МИКРОСХЕМА ИНТЕГРАЛЬНАЯ

К1921ВГ015

Руководство пользователя

от 19.02.2025

К1921ВГ015 – ультра низко-потребляющий микроконтроллер RISC-V архитектуры с 1Мб Flash памяти, 256 Кб (+64Кб) ОЗУ, ADC, CMP, CRYPTO, TRNG, SPI, QSPI, UART, I2C, CAN, AntiTAMPER, WakeUp.

Особенности

- Ультра низкое потребление:
  - напряжение питания: 1,62 В – 3,60В;
  - температурный диапазон: (-40 – 85) °С;
  - до 8 мкА в режиме POWERDOWN (GPR, RTC, внутренний RC-осциллятор, 3 WAKEUP);
  - до 190 мкА в режиме STOP (GPR, RTC, внутренний RC-осциллятор, 3 WAKEUP, SRAM1, CMP);
  - до 500 мкА / МГц в режиме RUN;
  - до 40 мкс время выхода из режима IDLE;
  - до 1800 мкс время выхода из режимов STOP и POWERDOWN.

- Сброс:
  - ультранизкопотребляющий POR;

- Ядро: RISC-V 32bit CPU
  - система команд: RV32IMFCN_ZBA_ZBB_ZBC_ZBS;
  - от 32 КГц до 50 МГц;
  - 1.35 DMIPS/MHz (Dhrystone 2.1)

- Тактовые сигналы:
  - внутренний осциллятор HSE для подключения внешнего резонатора от 2 МГц до 30 МГц;
  - внутренний осциллятор RTC: 32 КГц с программной калибровкой;
  - внутренний RC генератор HSI: 1МГц;
  - внутренний RC генератор LSI (RTC): 32 КГц;
  - внутренний осциллятор LSE для подключения внешнего резонатора 32768 Гц;
  - системная PLL: до 50 МГц.

- CRYPTO: AES-128; AES-256, Кузнечик, Магма.
- HASH: SHA-1, SHA-224, SHA-256, MD5.
- 2 CRC, 128 бит уникальный ID.
- До 48 выводов I/O.
- Память:
  - 1 Мб Flash памяти;
  - 256 Кб памяти ОЗУ0;
  - 64 Кб памяти ОЗУ1 в батарейном домене;
  - 64 байта регистров GPR.

- Аналоговая периферия:
  - 8-канальный 12-разрядный быстродействующий АЦП с режимами цифрового компаратора;
  - 8-канальный 16-разрядный сигма-дельта АЦП;
  - два аналоговых компаратора.

- Контроллер DMA: 24 канала.
- Контроллеры интерфейсов:
  - пять приемопередатчиков UART;
  - CAN 2.0b;
  - USB 2.0 FullSpeed (Device);
  - один контроллер I2C;
  - один контроллер QSPI;
  - два контроллера SPI.

- 6 таймеров: один 32-разрядный, три 16-разрядных, два сторожевых таймера (один независимый сторожевой таймер).
- Интерфейс отладки: JTAG.
- Корпус: LQFP100

Содержание

Введение

Микросхема К1921ВГ015 представляет собой СБИС 32-разрядного микроконтроллера на базе ядра RISC-V, предназначенного для промышленных и потребительских приложений, включая системы дистанционного мониторинга, контрольно-измерительные приборы, системы автоматизации производственных процессов, автомобильную электронику, а также устройства с батарейным питанием.

В состав микроконтроллера входит широкий набор цифровой и аналоговой периферии, поэтому, он может применяться в различных системах цифровой обработки сигналов, в том числе, требующих точных аналогово-цифровых преобразований, в системах управления и сбора информации.

В настоящем техническом описании приведено описание архитектуры, функционального построения и периферии микроконтроллера К1921ВГ015. Техническое описание может служить практическим руководством по применению микроконтроллера для разработчиков систем на его основе и программистов.

1 Область применения и особенности микроконтроллера

Сфера применения микросхемы К1921ВГ015 довольно широка: средства измерений, связи, наблюдения, безопасности, автоматизация производства, энергетики, промышленности, различных систем, работающих от батарейного питания.

В состав микросхемы К1921ВГ015 входят: блоки АЦП сигма-дельта, АЦП последовательного приближения с интерфейсом к контроллеру прямого доступа к памяти, модуль захвата/сравнения, блоки кодирования информации, блок RTC, два блока ОЗУ на 256 кБ и 64 кБ, блок контроля вскрытия и др.

Микроконтроллер содержит домен батарейного питания, в который входят периферийные блоки: RTC, аналоговые компараторы, блок контроля вскрытия, дополнительное ОЗУ 64 КБ, независимый сторожевой таймер. Периферийные блоки, входящие в состав домена батарейного питания, позволяют осуществлять контроль вскрытия корпуса системы, отсчитывать временные промежутки и сохранять информацию при отсутствии основного питания.

Разработанный микроконтроллер имеет встроенную Flash-память программ объемом 1 Мбайт, которую можно использовать для хранения и загрузки пользовательского программного обеспечения.

Каждый микроконтроллер содержит уникальный идентификационный номер ID, состоящий из 128 бит.

Система тактирования микроконтроллера позволяет использовать различные источники тактового сигнала, что позволяет расширить набор применений и решаемых задач пользователя. Существует возможность гибкой настройки тактовых сигналов для блоков периферии.

Для снижения энергопотребления микросхемы предусмотрена возможность отключения тактовых сигналов отдельных блоков периферии в случае, если они не используются пользователем. При переходе процессора в режим пониженного энергопотребления возможно отключение тактового сигнала ядра (команда WFI), а также возможность выбора периферийных блоков, которые будут отключены/включены при переходе/выходе из данного режима.

2 Краткое техническое описание микроконтроллера

2.1 Функциональные параметры

Структурная схема микроконтроллера показана на рисунке 2.1.

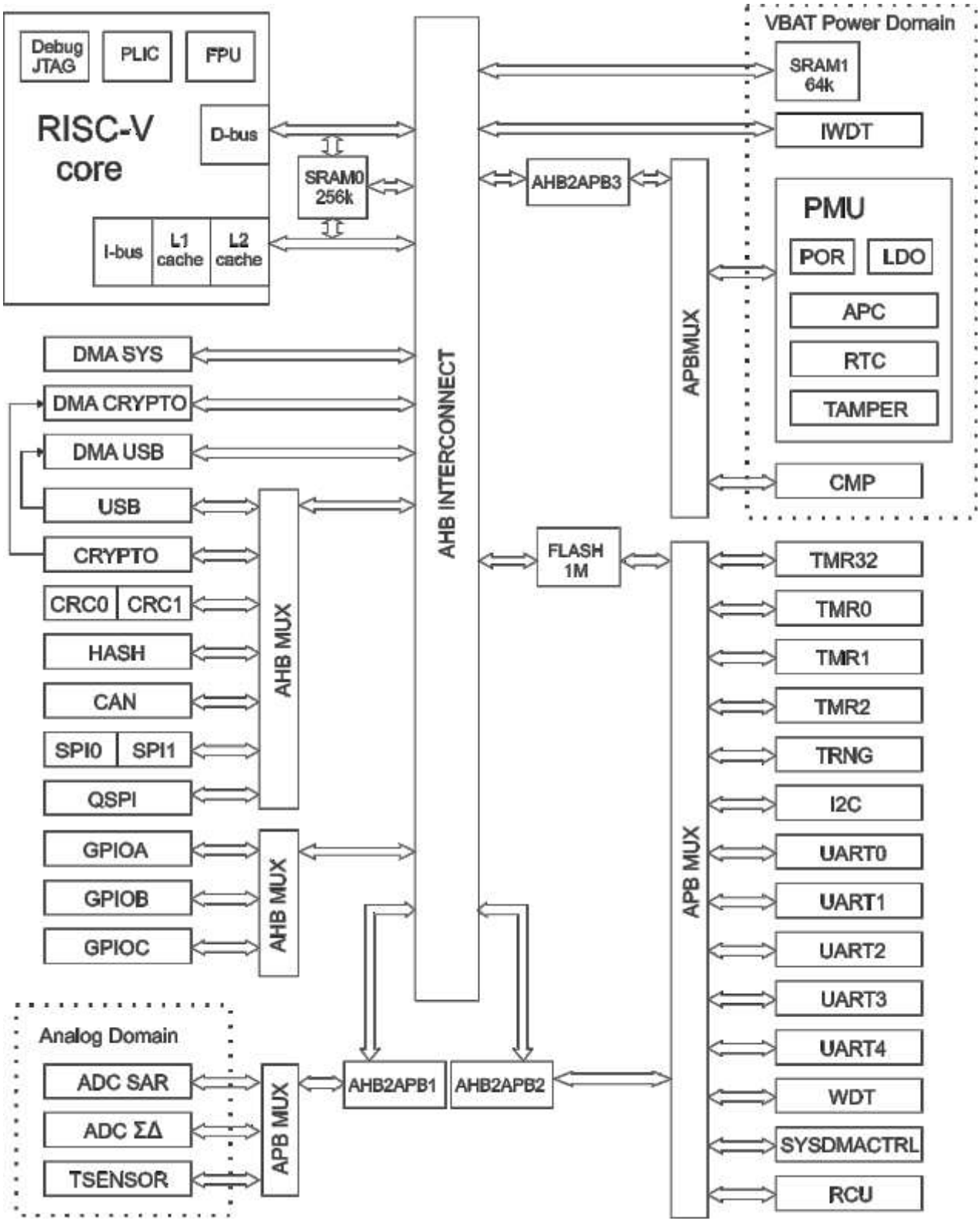


Рисунок 2.1 – Структурная схема микроконтроллера

В состав микроконтроллера входят функциональные элементы:

- 32-разрядное ядро архитектуры RISC-V с поддержкой системы команд RV32IMFCN_ZBA_ZBB_ZBC_ZBS, набора команд умножения, арифметических и логических команд, встроенным модулем обработки команд с плавающей запятой с одинарной точностью FPU, кэшем команд и поддержкой отладочного интерфейса JTAG;
- блок управления сбросом и синхронизацией RCU, имеющий в своем составе RC-генератор (1 МГц), синтезатор частоты SYSPLL и блок управления системными тактовыми сигналами SCM;
- системный блок управления энергопотреблением PMUSYS;
- блок управления энергопотреблением в составе с RTC модулем (PMURTC);
- блок коммутации AXI AHB;
- основная Flash-память объемом 1 Мбайт;
- SRAM0 (ОЗУ0) объемом 256 Кбайт;
- SRAM1 (ОЗУ1), подключенное к домену батарейного питания, объемом 64 Кбайт;
- 24-канальный контроллер прямого доступа к памяти DMA;
- блок часов реального времени RTC со входами контроля целостности;
- датчик температуры TSENSOR;
- сторожевой таймер WDT;
- независимый сторожевой таймер IWDT;
- один 8-канальный 12-разрядный быстродействующий АЦП с режимами цифрового компаратора для каждого из каналов (ADCSAR);
- один 8-канальный (16-канальный в бескорпусном исполнении) 16-разрядный сигма-дельта АЦП (ADCSD);
- три 16-разрядных порта ввода-вывода A, B, C;
- восемь аналоговых входов, подключенных к каналам АЦП (ADCSD и ADCSAR);
- один 32-разрядный таймер TMR32;
- три 16-разрядных таймеров TMR0 – TMR2;
- пять приемопередатчиков UART0 – UART4;
- блок криптографии CRYPTO;
- два блока вычисления CRC (CRC0, CRC1);
- генератор случайных чисел (TRNG);
- HASH процессор;
- контроллеры интерфейсов:
  - CAN 2.0b;
  - USB 2.0 FullSpeed (Device);
  - один контроллер I2C;
  - один контроллер QSPI;
  - два контроллера SPI (SPI0 – SPI1).

Особенности выводов микроконтроллера

Все выводы микроконтроллера по их функциональному назначению, организованы в группы:

- 16-разрядные порты ввода-вывода A, B, C;
- 8 аналоговых входов, подключаемых к каналам АЦП;
- порт тестирования JTAG;
- питание микроконтроллера;
- питание аналогового домена, включающего блоки:
  - АЦП последовательного приближения (ADC);
  - АЦП сигма-дельта (ADCSD);
  - внутренний датчик температуры (TSENSOR), подключаемый к одному из входов АЦП;
- питание батарейного домена, включающего блоки:

- ОЗУ1;
- PMU;
- контроля вскрытия TAMPER;
- часов реального времени RTC;
- аналоговые компараторы (CMP0- CMP1);
- IWDT.

Каждый цифровой вывод порта микроконтроллера может использоваться как двунаправленный вывод общего назначения (режим GPIO). Помимо этого, все выводы имеют альтернативную функцию (или функции). Режим работы, альтернативная функция, нагрузочная способность и быстродействие, а также подтяжка к высокому уровню и функционирование в режиме выхода с открытым стоком/истоком могут быть заданы для каждого вывода независимо от других.

Примечание – После сброса микроконтроллера выводы портов ввода-вывода конфигурируются как выводы общего назначения и находятся в третьем состоянии.

Порт JTAG, предназначенный для внутрисхемного программирования микроконтроллера, тестирования и отладки программ пользователя, включает в свой состав пять выводов TCK, TMS, TDI, TDO, TRST для подключения JTAG эмулятора.

Выводы ADC_CH0 – ADC_CH7 (входные каналы АЦП), не подключенные к внешним источникам напряжения, находятся в плавающем состоянии.

Вывод RST является входом сигнала сброса микроконтроллера и должен находиться в состоянии логической единицы. Внешний сброс осуществляется подачей на вывод логического нуля в течение минимум 10 мкс.

Микроконтроллер имеет конфигурационный вывод SERVEN. При подаче на SERVEN логической единицы во время сброса, микроконтроллер переходит в сервисный режим, в котором запрещаются любые операции со всей Flash-памятью, кроме полного стирания.

Выводы XI_RTC и XO_RTC предназначены для подключения внешнего источника тактового сигнала (с частотой 32 кГц) блока RTC.

Выводы XI_OSC и XO_OSC предназначены для подключения внешнего источника тактового сигнала микроконтроллера с частотой (2 – 30) МГц.

Вывод V_BAT предназначен для подключения внешнего источника питания (батарейки). Номинальное значение напряжения должно находиться в диапазоне от 1,7 до 3,6 В.

Выводы VCC1 и GND1 предназначены для подключения внешних источников питания ядра, Flash, PLL и периферии микроконтроллера. Номинальное значение напряжения должно находиться в диапазоне от 1,7 до 3,6 В. Максимальный ток потребления с учётом нагрузки по GPIOA, GPIOB, GPIOC - до 150 мА.

Вывод № 12 из группы выводов VCC1 предназначен для подключения входного напряжения АРС.

Вывод № 16 из группы выводов VCC1 предназначен для подключения LDO1 питания ядра.

Вывод № 61 из группы выводов VCC1 предназначен для подключения питания Flash и PLL.

Выводы VCC2 и GND2 предназначены для подключения внешних источников питания АЦП и датчика температуры. Номинальное значение напряжения 3,3 В.

Вывод AREF предназначен для подключения внешнего источника опорного напряжения АЦП. Номинальное значение напряжения должно находиться в диапазоне от 1,2 до 3,3 В. Максимальный ток - до 0,1 мА.

Выводы VAON_CAP, VCORE_CAP, VBAT_CAP предназначены для подключения внешних конденсаторов емкостью 22 мкФ. Конденсаторы, подключаемые к выводам VAON_CAP, VCORE_CAP, VBAT_CAP вторым выводом подключаются к GND1.

Вывод SDADC_CAP предназначен для подключения внешнего конденсатора емкостью 0,1 мкФ. Конденсатор, подключаемый к выводу SDADC_CAP, вторым выводом подключается к GND2.

Примечание – На плате вывод питания GND1 может быть объединен с выводами питания GND2 (при этом должны быть приняты меры для снижения помех по питанию).

Выключение питания микроконтроллера подразумевает полное снятие напряжения как с выводов питания VCC1, VCC2, V_BAT, AREF, так и со всех остальных выводов микроконтроллера.

Примечание – Запрещено подавать напряжение на функциональные выводы при выключенном питании микроконтроллера.

УГО и назначение выводов микроконтроллера

Условное графическое обозначение микроконтроллера приведено на рисунке 2.2.

Функциональное назначение выводов микроконтроллера приведено в таблицах 2.1 – 2.3, в которых приняты обозначения: I – вход, O – выход, I/O – вход/выход, Z – третье состояние.

Примечание – Альтернативные функции выводов портов A, B, C указаны в таблице 2.1.

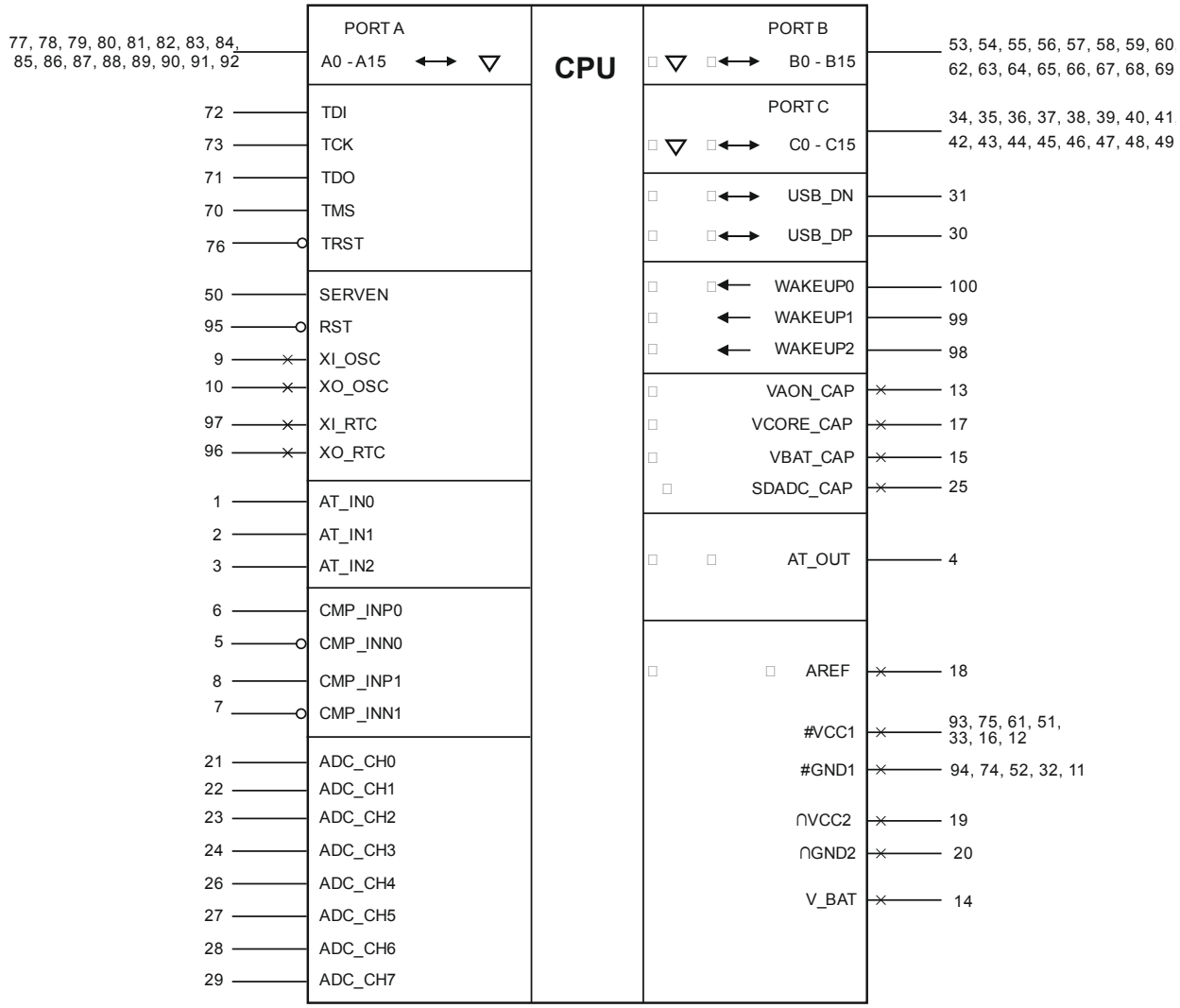


Рисунок 2.2 – Условное графическое обозначение микросхемы

Таблица 2.1 – Функциональное назначение выводов, имеющих альтернативные функции

Обозначение вывода	Обозначение альтернативной функции вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
1	2	3	4	5
A0		77	I/O/Z	Вход/выход порта A, разряд 0
	UART0_RX		I	Вход данных UART0
	TMR2_OUT0		O	Выход TMR2
A1		78	I/O/Z	Вход/выход порта A, разряд 1
	UART0_TX		O	Выход данных UART0
	TMR2_OUT1		O	Выход TMR2
A2		79	I/O/Z	Вход/выход порта A, разряд 2
	UART1_RX		I	Вход данных UART1
	TMR2_OUT2		O	Выход TMR2
A3		80	I/O/Z	Вход/выход порта A, разряд 3
	UART1_TX		O	Выход данных UART1
	TMR2_OUT3		O	Выход TMR2
A4		81	I/O/Z	Вход/выход порта A, разряд 4
	UART2_RX		I	Вход данных UART2
	TMR1_CCIA		I	Вход захвата TMR1
	QSPI_CLK		I/O	Вход/выход синхронизации QSPI
A5		82	I/O/Z	Вход/выход порта A, разряд 5
	UART2_TX		O	Выход данных UART2
	TMR1_CCIB		I	Вход захвата TMR1
	QSPI_FSS		I/O	Вход/выход выбора устройства QSPI
A6		83	I/O/Z	Вход/выход порта A, разряд 6
	UART3_RX		I	Вход данных UART3
	TMR1_OUT0		O	Выход TMR1
	QSPI_IO0		I/O	Вход/выход данных QSPI, разряд 0
A7		84	I/O/Z	Вход/выход порта A, разряд 7
	UART3_TX		O	Выход данных UART3
	TMR1_OUT1		O	Выход TMR1
	QSPI_IO1		I/O	Вход/выход данных QSPI, разряд 1
A8		85	I/O/Z	Вход/выход порта A, разряд 8
	UART4_RX		I	Вход данных UART4
	TMR1_OUT2		O	Выход TMR1
	QSPI_IO2		I/O	Вход/выход данных QSPI, разряд 2
A9		86	I/O/Z	Вход/выход порта A, разряд 9
	UART4_TX		O	Выход данных UART4
	TMR1_OUT3		O	Выход TMR1
	QSPI_IO3		I/O	Вход/выход данных QSPI, разряд 3
A10		87	I/O/Z	Вход/выход порта A, разряд 10
	QSPI_CLK		I/O	Вход/выход синхронизации QSPI
	UART0_CTS		I	Вход сигнала готовности к приему UART0
	UART1_RX		I	Вход данных UART1

Продолжение таблицы 2.1

1	2	3	4	5
A11		88	I/O/Z	Вход/выход порта A, разряд 11
	QSPI_FSS		I/O	Вход/выход выбора устройства QSPI
	UART0_DCD		I	Вход отклика при обнаружении информационного сигнала UART0
	UART1_TX		O	Выход данных UART1
A12		89	I/O/Z	Вход/выход порта A, разряд 12
	QSPI_IO0		I/O	Вход/выход данных QSPI, разряд 0
	UART0_DSR		I	Вход сигнала готовности источника данных UART0
	UART2_RX		I	Вход данных UART2
A13		90	I/O/Z	Вход/выход порта A, разряд 13
	QSPI_IO1		I/O	Вход/выход данных QSPI, разряд 1
	UART0_RI		I	Вход сигнала-индикатора вызова UART0
	UART2_TX		O	Выход данных UART2
A14		91	I/O/Z	Вход/выход порта A, разряд 14
	QSPI_IO2		I/O	Вход/выход данных QSPI, разряд 2
	UART0_RTS		O	Выход сигнала запроса на передачу UART0
	UART3_RX		I	Вход данных UART3
A15		92	I/O/Z	Вход/выход порта A, разряд 15
	QSPI_IO3		I/O	Вход/выход данных QSPI, разряд 3
	UART0_DTR		O	Выход сигнала готовности приемника данных UART0
	UART3_TX		O	Выход данных UART3
B0		53	I/O/Z	Вход/выход порта B, разряд 0
	SPI0_CLK		I/O	Вход/выход синхронизации SPI0
	UART1_CTS		I	Вход сигнала готовности к приему UART1
	UART4_RX		I	Вход данных UART4
B1		54	I/O/Z	Вход/выход порта B, разряд 1
	SPI0_FSS		I/O	Вход/выход выбора устройства SPI0
	UART1_DCD		I	Вход отклика при обнаружении информационного сигнала UART1
	UART4_TX		O	Выход данных UART4
B2		55	I/O/Z	Вход/выход порта B, разряд 2
	SPI0_RX		I	Вход данных SPI0
	UART1_DSR		I	Вход сигнала готовности источника данных UART1
	TMR32_EXTIN		I	Вход синхронизации TMR32
B3		56	I/O/Z	Вход/выход порта B, разряд 3
	SPI0_TX		O	Выход данных SPI0
	UART1_RI		I	Вход сигнала-индикатора вызова UART1
	TMR0_EXTIN		I	Вход синхронизации TMR0

Продолжение таблицы 2.1

1	2	3	4	5
B4		57	I/O/Z	Вход/выход порта B, разряд 4
	SPI1_CLK		I/O	Вход/выход синхронизации SPI1
	UART1_RTS		O	Выход сигнала запроса на передачу UART1
	TMR1_EXTIN		I	Вход синхронизации TMR1
B5		58	I/O/Z	Вход/выход порта B, разряд 5
	SPI1_FSS		I/O	Вход/выход выбора устройства SPI1
	UART1_DTR		O	Выход сигнала готовности приемника данных UART1
	TMR2_EXTIN		I	Вход синхронизации TMR2
B6		59	I/O/Z	Вход/выход порта B, разряд 6
	SPI1_RX		I	Вход данных SPI1
	UART2_CTS		I	Вход сигнала готовности к приему UART2
	UART0_RX		I	Вход данных UART0
B7		60	I/O/Z	Вход/выход порта B, разряд 7
	SPI1_TX		O	Выход данных SPI1
	UART2_DCD		I	Вход отклика при обнаружении информационного сигнала UART2
	UART0_TX		O	Выход данных UART0
B8		62	I/O/Z	Вход/выход порта B, разряд 8
	CAN0_RX		I	Вход данных CAN0
	UART2_DSR		I	Вход сигнала готовности источника данных UART2
	TMR1_OUT0		O	Выход TMR1
B9		63	I/O/Z	Вход/выход порта B, разряд 9
	CAN0_TX		O	Выход данных CAN0
	UART2_RI		I	Вход сигнала-индикатора вызова UART2
	TMR1_OUT1		O	Выход TMR1
B10		64	I/O/Z	Вход/выход порта B, разряд 10
	CAN1_RX		I	Вход данных CAN1
	UART2_RTS		O	Выход сигнала запроса на передачу UART2
	TMR1_OUT2		O	Выход TMR1
B11		65	I/O/Z	Вход/выход порта B, разряд 11
	CAN1_TX		O	Выход данных CAN1
	UART2_DTR		O	Выход сигнала готовности приемника данных UART2
	TMR1_OUT3		O	Выход TMR1
B12		66	I/O/Z	Вход/выход порта B, разряд 12
	CMP_OUT0		O	Выход компаратора, нулевой канал
	UART3_CTS		I	Вход сигнала готовности к приему UART3
	TMR1_CCIA		I	Вход захвата TMR1
B13		67	I/O/Z	Вход/выход порта B, разряд 13
	CMP_OUT1		O	Выход компаратора, первый канал
	UART3_DCD		I	Вход отклика при обнаружении информационного сигнала UART3
	TMR1_CCIB		I	Вход захвата TMR1

Продолжение таблицы 2.1

1	2	3	4	5
B14		68	I/O/Z	Вход/выход порта B, разряд 14
	TMR32_CCIA		I	Вход захвата TMR32
	UART3_DSR		I	Вход сигнала готовности источника данных UART3
	CMP_OUT0		O	Выход компаратора, нулевой канал
B15		69	I/O/Z	Вход/выход порта B, разряд 15
	TMR32_CCIB		I	Вход захвата TMR32
	UART3_RI		I	Вход сигнала-индикатора вызова UART3
	CMP_OUT1		O	Выход компаратора, первый канал
C0		34	I/O/Z	Вход/выход порта C, разряд 0
	TMR32_OUT0		O	Выход TMR32
	UART3_RTS		O	Выход сигнала запроса на передачу UART3
C1		35	I/O/Z	Вход/выход порта C, разряд 1
	TMR32_OUT1		O	Выход TMR32
	UART3_DTR		O	Выход сигнала готовности приемника данных UART3
C2		36	I/O/Z	Вход/выход порта C, разряд 2
	TMR32_OUT2		O	Выход TMR32
	UART4_CTS		I	Вход сигнала готовности к приему UART4
C3		37	I/O/Z	Вход/выход порта C, разряд 3
	TMR32_OUT3		O	Выход TMR32
	UART4_DCD		I	Вход отклика при обнаружении информационного сигнала UART4
C4		38	I/O/Z	Вход/выход порта C, разряд 4
	TMR32_EXTIN		I	Вход синхронизации TMR32
	UART4_DSR		I	Вход сигнала готовности источника данных UART4
C5		39	I/O/Z	Вход/выход порта C, разряд 5
	TMR0_EXTIN		I	Вход синхронизации TMR0
	UART4_RI		I	Вход сигнала-индикатора вызова UART4
C6		40	I/O/Z	Вход/выход порта C, разряд 6
	TMR0_OUT0		O	Выход TMR0
	UART4_RTS		O	Выход сигнала запроса на передачу UART4
C7		41	I/O/Z	Вход/выход порта C, разряд 7
	TMR0_OUT1		O	Выход TMR0
	UART4_DTR		O	Выход сигнала готовности приемника данных UART4
	CLKOUT		O	Выход тактового сигнала CLKOUT

Продолжение таблицы 2.1

1	2	3	4	5
C8		42	I/O/Z	Вход/выход порта C, разряд 8
	TMR0_OUT2		O	Выход TMR0
	CAN0_RX		I	Вход данных CAN0
C9		43	I/O/Z	Вход/выход порта C, разряд 9
	TMR0_OUT3		O	Выход TMR0
	CAN0_TX		O	Выход данных CAN0
C10		44	I/O/Z	Вход/выход порта C, разряд 10
	TMR0_CCIA		I	Вход захвата TMR0
	CAN1_RX		I	Вход данных CAN1
C11		45	I/O/Z	Вход/выход порта C, разряд 11
	TMR0_CCIB		I	Вход захвата TMR0
	CAN1_TX		O	Выход данных CAN1
C12		46	I/O/Z	Вход/выход порта C, разряд 12
	I2C_SCL		I/O	Вход/выход синхронизации I2C
	TMR1_EXTIN		I	Вход синхронизации TMR1
C13		47	I/O/Z	Вход/выход порта C, разряд 13
	I2C_SDA		I/O	Вход/выход данных I2C
	TMR2_EXTIN		I	Вход синхронизации TMR2
C14		48	I/O/Z	Вход/выход порта C, разряд 14
	TMR2_CCIA		I	Вход захвата TMR2
	I2C_SCL		I/O	Вход/выход синхронизации I2C
C15		49	I/O/Z	Вход/выход порта C, разряд 15
	TMR2_CCIB		I	Вход захвата TMR2
	I2C_SDA		I/O	Вход/выход данных I2C

Примечание – В графе «Тип вывода» принятые условные обозначения:
- I – вход,
- O – выход,
- Z – третье состояние.

Таблица 2.2 – Функциональное назначение выводов, не имеющих альтернативных функций

Обозначение вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
1	2	3	4
XI_OSC	9	–	Вывод внешнего тактового сигнала/ Вывод для подключения кварцевого резонатора

XO_OSC	10	–	Вывод для подключения кварцевого резонатора

Продолжение таблицы 2.2

1	2	3	4
RST	95	I	Вход сигнала сброса
TCK	73	I	Вход тактового сигнала порта JTAG
TDI	72	I	Вход данных порта JTAG
TDO	71	O	Выход данных порта JTAG
TMS	70	I	Вход режима порта JTAG
TRST	76	I	Вход сброса порта JTAG
SERVEN	50	I	Вход сигнала активации сервисного режима
XI_RTC	97	–	Вывод для подключения кварцевого резонатора (32 кГц)
XO_RTC	96	–	Вывод для подключения кварцевого резонатора (32 кГц)
USB_DN	31	I/O	Вход/выход «USB Dn»
USB_DP	30	I/O	Вход/выход «USB Dp»
CMP_INP0	6	I	Прямой вход компаратора, канал 0
CMP_INN0	5	I	Инверсный вход компаратора, канал 0
CMP_INP1	8	I	Прямой вход компаратора, канал 1
CMP_INN1	7	I	Инверсный вход компаратора, канал 1
AT_IN0	1	I	Вход датчика вскрытия
AT_IN1	2	I	Вход датчика вскрытия
AT_IN2	3	I	Вход датчика вскрытия
AT_OUT	4	O	Выход генерации сигнала ПСП для датчиков вскрытия
WAKEUP0	100	I	Вход 0 для пробуждения из режима энергосбережения
WAKEUP1	99	I	Вход 1 для пробуждения из режима энергосбережения
WAKEUP2	98	I	Вход 2 для пробуждения из режима энергосбережения
ADC_CH0	21	I	Вход ADCSAR/ADCSD, канал 0
ADC_CH1	22	I	Вход ADCSAR/ADCSD, канал 1
ADC_CH2	23	I	Вход ADCSAR/ADCSD, канал 2
ADC_CH3	24	I	Вход ADCSAR/ADCSD, канал 3
ADC_CH4	26	I	Вход ADCSAR/ADCSD, канал 4
ADC_CH5	27	I	Вход ADCSAR/ADCSD, канал 5
ADC_CH6	28	I	Вход ADCSAR/ADCSD, канал 6
ADC_CH7	29	I	Вход ADCSAR/ADCSD, канал 7
GND1	94, 74, 52, 32, 11	–	Цифровая земля
VCC1	93, 75,61, 51, 33, 16, 12	–	Цифровое питание
V_BAT	14	–	Батарейное питание
VAON_CAP	13	–	Вывод подключения конденсатора
VCORE_CAP	17	–	Вывод подключения конденсатора
VBAT_CAP	15	–	Вывод подключения конденсатора

VCC2	19	–	Аналоговое питание
GND2	20	–	Аналоговая земля

Продолжение таблицы 2.2

1	2	3	4
AREF	18	–	Опорное напряжение ADCSAR
SDADC_CAP	25	–	Вывод подключения конденсатора

Примечание – В графе «Тип вывода» принятые условные обозначения:
- I – вход,
- O – выход,
- Z – третье состояние.

Микросхемы выполнены в корпусе LQFP100.
Масса микросхемы – не более 4 г.

2.2 Электрические параметры

Номинальное значение напряжения питания по выводам:

- VCC1 должно быть в диапазоне от 1,7 до 3,6 В с допустимой разностью значений между выводами не более 50 мВ и с амплитудным значением пульсации напряжения не более 30 мВ;

Значение суммарного максимального тока по выводам портов ввода-вывода A, B, C, не должно превышать 150 мА.

Электрические параметры микросхем при приемке и поставке соответствуют нормам, приведенным в таблице 2.4.

Таблица 2.4 – Электрические параметры микросхем при приемке и поставке

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение параметра	Норма параметра		Температура среды, ºС
		не менее	не более	
1	2	3	4	5
1 Выходное напряжение низкого уровня буферов ввода-вывода, В, $U_{CC1} = U_{CC2} = 3,0$ В, $I_{OL} = 4,0$ мА	U_{OL}	–	0,4	
2 Выходное напряжение высокого уровня буферов ввода-вывода, В, $U_{CC1} = U_{CC2} = 3,0$ В, $I_{OH} = –4,0$ мА	U_{OH}	U_{CC1}-0,4	–	
3 Ток утечки низкого уровня по входам с отключенными схемами «Pull-up» и «Pull-down», $U_{CC1} = U_{CC2} = 3,6$ В, $U_{IL} = 0$ В	I_{ILL}	–2	–	
4 Ток утечки высокого уровня по входам с отключенными схемами «Pull-up» и «Pull-down», $U_{CC1} = U_{CC2} = 3,6$ В, $U_{IH} = U_{CC1}$	I_{ILL}	–	2	
Входной ток низкого уровня по выводам с подключенной схемой «Pull-up», мкА, $U_{CC1} = U_{CC2} = 3,6$ В, $U_{IL} = 0$ В	I_{IL1}	–200	–10	–40±3 25±10 85±5
Входной ток высокого уровня по выводам с подключенной схемой «Pull-up», мкА, $U_{CC1} = U_{CC2} = 3,6$ В, $U_{IH} = U_{CC1}$	I_{IH1}	–	2	
Входной ток низкого уровня по выводам с подключенной схемой «Pull-down», мкА, $U_{CC1} = U_{CC2} = 3,6$ В, $U_{IL} = 0$ В	I_{IL2}	10	200	
Входной ток высокого уровня по выводам с подключенной схемой «Pull-down», мкА, $U_{CC1} = U_{CC2} = 3,6$ В, $U_{IH} = U_{CC1}$	I_{IH2}	–2	–	
Входной ток низкого уровня по выводу тактового сигнала XI_OSC, мкА, $U_{CC1} = U_{CC2} = 3,6$ В, $U_{IL} = 0$ В	I_{IL3}	–40	–	
В $U_{CC1} = U_{CC2} = 3,6$ В, $U_{IH} = U_{CC1}$	I_{IH3}	–	40	

Продолжение таблицы 2.4

1	2	3	4	5
Входной ток низкого уровня по выводу сброса, мкА, $U_{CC1} = U_{CC2} = 3{,}6$ В, $U_{IL} = 0$ В	I_{IL4}	−800	−50	
12 Потребление в нормальном режиме работы RUN (вся периферия включена), мА $U_{CC1} = U_{CC2} = 3{,}6$ В, $f_{CI} = 50$ МГц, $U_{LDO1} = U_{LDO0} = 1{,}2$ В	$I_{CC1RUN1}$	–	25	
13 Потребление в нормальном режиме работы RUN (вся периферия выключена), мА $U_{CC1} = U_{CC2} = 3{,}6$ В, $f_{CI} = 50$ МГц, $U_{LDO1} = U_{LDO0} = 1{,}2$ В	$I_{CC1RUN2}$	–	20	
14 Потребление в нормальном режиме работы RUN (вся периферия включена, исполнение программы из ОЗУ), мА $U_{CC1} = U_{CC2} = 3{,}6$ В, $f_{CI} = 50$ МГц, $U_{LDO1} = U_{LDO0} = 1{,}2$ В	$I_{CC1RUN3}$	–	25	
15 Потребление в нормальном режиме работы RUN (вся периферия выключена, исполнение программы из ОЗУ), мА $U_{CC1} = U_{CC2} = 3{,}6$ В, $f_{CI} = 50$ МГц, $U_{LDO1} = U_{LDO0} = 1{,}2$ В	$I_{CC1RUN4}$	–	20	
16 Потребление в режиме STOP (домен ядра выключен, включен домен RTC и батареи CMP), мкА $f_{CI} = 1$ МГц (от HSI) , $U_{LDO0} = 1{,}2$ В	$I_{CC1STOP}$	–	190	−40±3 25±10 85±5
17 Потребление в режиме POWERDOWN (домен ядра выключен, домен батареи выключен, включен домен RTC (GPR, часы реального времени, работа от внутреннего RC-осциллятора 32 кГц), мкА U	I_{CC1PDN}	–	8	
18 Потребление блока АЦП SAR по линии U_{CC2}, мкА $U_{CC1} = U_{CC2} = 3{,}3$ В, Fs = 1 Msps	I_{CC2SAR}	–	1000	
Интегральная нелинейность АЦП SAR, МР $U_{CC1} = U_{CC2} = 3{,}3$ В, $f_{CI} = 50$ МГц, fs = 250 кГц	E_L	–	±3,0	
20 Дифференциальная нелинейность АЦП SAR, MI $U_{CC1} = U_{CC2} = 3{,}3$ В, $f_{CI} = 50$ МГц, fs=250 кГц	E_{LD}	–	±2,0	
Отношение сигнал/(шум + искажения) в канале АЦП SAR, дБ, $U_{CC1} = U_{CC2} = 3{,}3$ В, $f_{CI} = 50$ МГц, fs = 1 МГц	SINAD	60	–	
22 Гистерезис компаратора, мВ $U_{CC1} = U_{CC2} = 3{,}3$ В	U_{HCMP}	–	40	
23 Входной ток утечки по входам компаратора, мкА $U_{CC1} = U_{CC2} = 3{,}6$ В	I_{LCMP}	–	10	

Продолжение таблицы 2.4

1	2	3	4	5
24 Функциональный контроль $U_{CC1} = U_{CC2} = U_{CC3}$ = (1,7; 3,6) В; f_{CI} = (2; 50) МГц	ФК	–	–	−40±3 25±10 85±5
Примечания 1 Норма для тока потребления I_{CC1PDN} в режиме POWERDOWN приведена для диапазона температур среды от 25 ºС до 85 ºС.				

Необходимо соблюдать следующий порядок подачи напряжения питания: сначала подаются напряжения питания UCC1, UCC2, а затем входные напряжения низкого уровня UIL и высокого уровня UIH.

Порядок снятия напряжений с выводов микросхемы при выключении должен быть обратным подаче: первыми снимаются входные сигналы, а затем напряжения питания.

Допускается возможность одновременной подачи и снятия напряжений питания и напряжений входных сигналов.

Значения предельно допустимых электрических режимов эксплуатации в диапазоне рабочих температур среды соответствуют нормам, приведенным в таблице 2.5.

Таблица 2.5 – Значения предельно допустимых электрических режимов эксплуатации

Наименование параметра режима, единица измерения		Буквенное обозначение параметра	Предельно допустимый режим	
			не менее	не более
1		2	3	4
1 Напряжение питания цифровой части[1)], В		U_{CC1}	1,62	3,6
2 Напряжение питания аналоговой части[1)], В		U_{CC2}	1,62	3,6
3 Напряжение питания по выводу V_BAT, В		U_{CC3}	1,7	3,6
4 Напряжение источника опорного напряжения АЦП, В		U_{AREF}	1,2	3,3
5 Входное напряжение низкого уровня[1)], В		U_{IL}	–0,5	0,8
6 Входное напряжение высокого уровня[1)], В		U_{IH}	$0,7U_{CC1}$	U_{CC1}
7 Входное напряжение высокого уровня по выводу XI_OSC, В		U_{IHCI}	$0,7U_{CC1}$	U_{CC1}
8 Входное напряжение высокого уровня сигнала сброса RST#, В		U_{IHRST}	$0,6U_{CC1}$	U_{CC1}
9 Выходной ток низкого уровня[1)], мА		I_{OL}	–	4,0
10 Выходной ток высокого уровня[1)], мА		I_{OH}	–4,0	–
11 Системная частота следования импульсов тактового сигнала процессорного ядра SYSCLK, МГц		f_{CI}	–	50
12 Частота следования импульсов тактового сигнала по выводу OSC, МГц	при работе с внешним тактовым генератором	f_C	2	30
	при работе с кварцевым резонатором		8	24
13 Емкость нагрузки, пФ		C_L	–	40
14 Напряжение питания по выводу VCC1 61 при работе блоков FLASH и PLL, В		$U_{CC61FPLL}$	2,25	3,6
15 Напряжение питания по выводу VCC1 61 при работе только блока FLASH, В		U_{CC61F}	1,7	3,6

Продолжение таблицы 2.5

1	2	3	4
16 Диапазон уровня входного сигнала SAR АЦП, В	U_{SARIN}	–0,5	U_{CC2}
17 Входной диапазон напряжений компаратора, В При $U_{CC1} \geq U_{BAT}$	U_{CMPIN1}	–0,5	U_{CC1}
18 Входной диапазон напряжений компаратора, В При $U_{CC1} < U_{BAT}$	U_{CMPIN2}	–0,5	U_{BAT}
19 Максимальный входной уровень сигнала по линиям данных USB, В	U_{USBIN}	–	U_{CC1}
[1] Время работы в одном из предельных режимов должно быть не более 5 с.			

Таблица 2.6 – Значения параметров предельных электрических режимов эксплуатации

Наименование параметра режима, единица измерения	Буквенное обозначение параметра	Предельный режим	
		не менее	не более
1 Напряжение питания цифровой части[1], В	U_{CC1}	–	5,0
2 Напряжение питания аналоговой части[1], В	U_{CC2}	–	5,0
3 Входное напряжение низкого уровня[1], В	U_{IL}	–0,6	–
4 Входное напряжение высокого уровня[1], В	U_{IH}	–	U_{CC1}+0,6
5 Входное напряжение высокого уровня по выводу XI_OSC, В	U_{IHCI}	–	U_{CC1}+0,6
6 Входное напряжение высокого уровня сигнала сброса RST#, В	U_{IHRST}	–	U_{CC1}+0,6
7 Выходной ток низкого уровня[1], мА	I_{OL}	–	10
8 Выходной ток высокого уровня[1], мА	I_{OH}	–10	–
Примечание – Нормы на параметры приведены для температуры 25 ºС. [1] Время работы в одном из предельных режимов должно быть не более 5 с.			

3 Архитектура изделия

Микроконтроллер К1921ВГ015 спроектирован на базе RISC-V ядра ВМ-310S6.

Процессорное ядро ВМ-310S6 поддерживает систему команд RV32IMFCN_ZBA_ZBB_ZBC_ZBS. ВМ-310S6 поддерживает два режима привилегированности: machine и user. Режим user предоставляет механизм изоляции процессов друг от друга и от доверенного кода, исполняемого в machine режиме.

Конвейер ВМ-310S6 состоит из двух стадий, на которых выполняются следующие операции:

1 Генерация запроса в подсистему памяти программ (PMS).

2 Чтение фрагмента кода из PMS и извлечение команд. Декодирование команды. Исполнение команды.

Подсистема предварительной обработки команд (FE) отвечает за выполнение операций первой стадии конвейера и частично второй стадии конвейера.

Блок FE состоит из следующих блоков:

1 IFU - Instruction Fetch Unit - содержит логику формирования адреса следующего фрагмента кода.

2 IDU - Instruction Decode Unit - блок предварительного декодирования инструкций.

3 IFQ - Instruction Fetch Queue - очередь фрагментов кода.

3.1 Блок коммутации микроконтроллера

Все устройства микроконтроллера соединены между собой через блок коммутации. На рисунке 3.1 приведена схема соединения основных и периферийных блоков микроконтроллера.

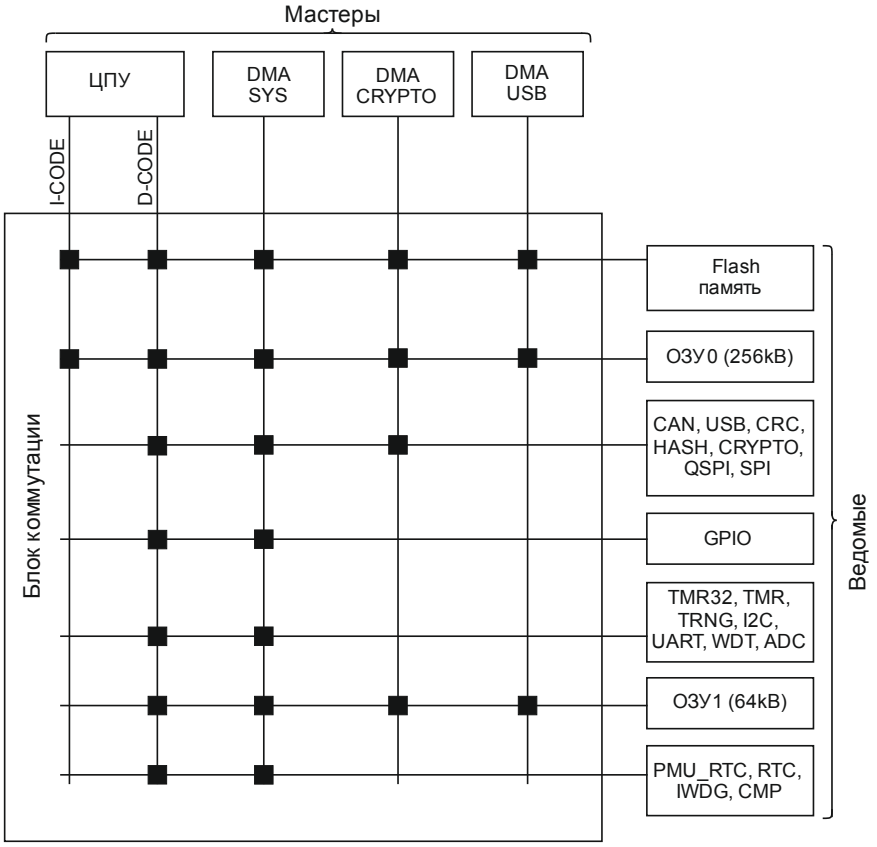


Рисунок 3.1 – Схема коммутации периферийных блоков микроконтроллера

4 Блок управления сбросом и синхронизацией RCU

4.1 Общая система тактирования

В микроконтроллере предусмотрена развитая подсистема управления тактовыми сигналами и сигналами сброса, см. рисунок 4.1.

Основными тактовыми сигналами, которые используются в микросхеме, являются:

- HSECLK – внешний тактовый сигнал с выводов XI_OSC и XO_OSC;
- HSICLK – внутренний высокочастотный тактовый сигнал частотой 1 МГц (тактовый сигнал внутреннего RC-генератора);
- LSICLK – внутренний низкочастотный тактовый сигнал частотой 32,768 кГц с блока RTC;
- REFCLK – опорный тактовый сигнал (зависит от режима PowerDown);
- SYSPLL0CLK – тактовый сигнал с выхода 0 PLLSYS;
- SYSPLL1CLK – тактовый сигнал с выхода 1 PLLSYS;
- SYSCLK – системный тактовый сигнал, определяющий частоту работы процессорного ядра. Источник сигала SYSCLK выбирается битовым полем CLKSRC регистра SYSCLKCFG (см. Приложение А.1);
- CLKOUT – выходной сигнал блока формирования внешнего тактового сигнала;
- USBPLL0CLK – тактовый сигнал с выхода 0 PLLUSB;
- TMR32_EXTINCLK – внешний тактовый сигнал таймера TMR32, подключенный к выводу TMR32_EXTIN;
- TMR0_EXTINCLK – внешний тактовый сигнал таймера TMR0, подключенный к выводу TMR0_EXTIN;
- TMR1_EXTINCLK – внешний тактовый сигнал таймера TMR1, подключенный к выводу TMR1_EXTIN;
- TMR2_EXTINCLK – внешний тактовый сигнал таймера TMR2, подключенный к выводу TMR2_EXTIN.

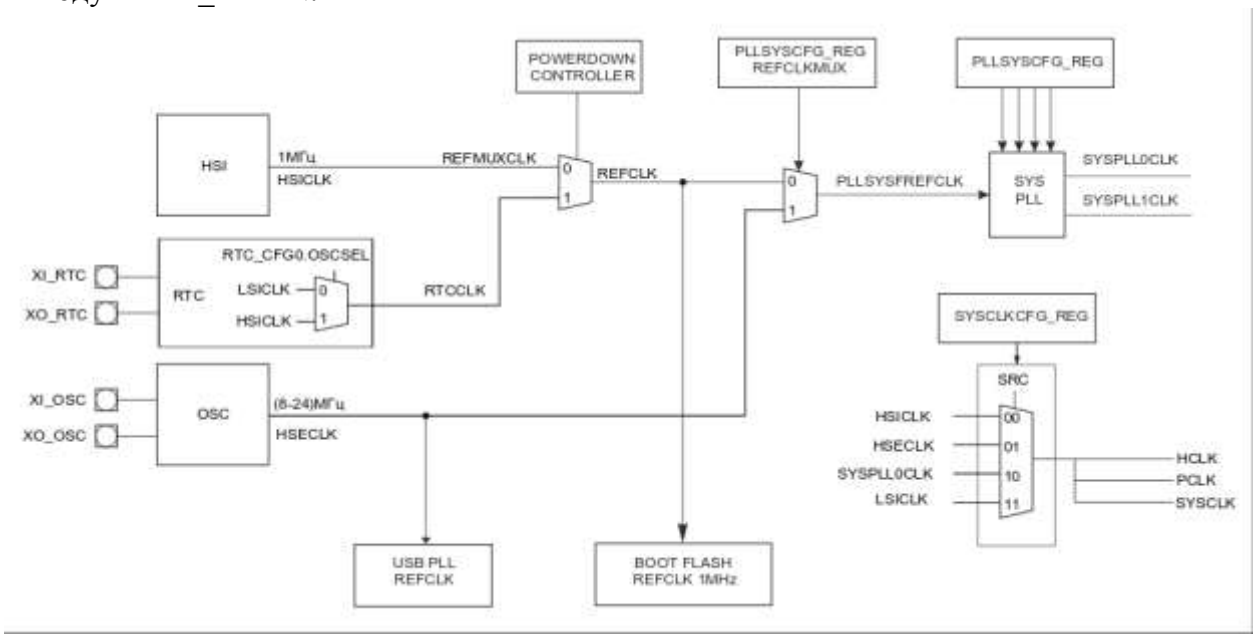


Рисунок 4.1 – Схема синхронизации

4.2 Синтезатор частоты PLL

Общая функциональная схема блока PLL показана на рисунке 4.2.

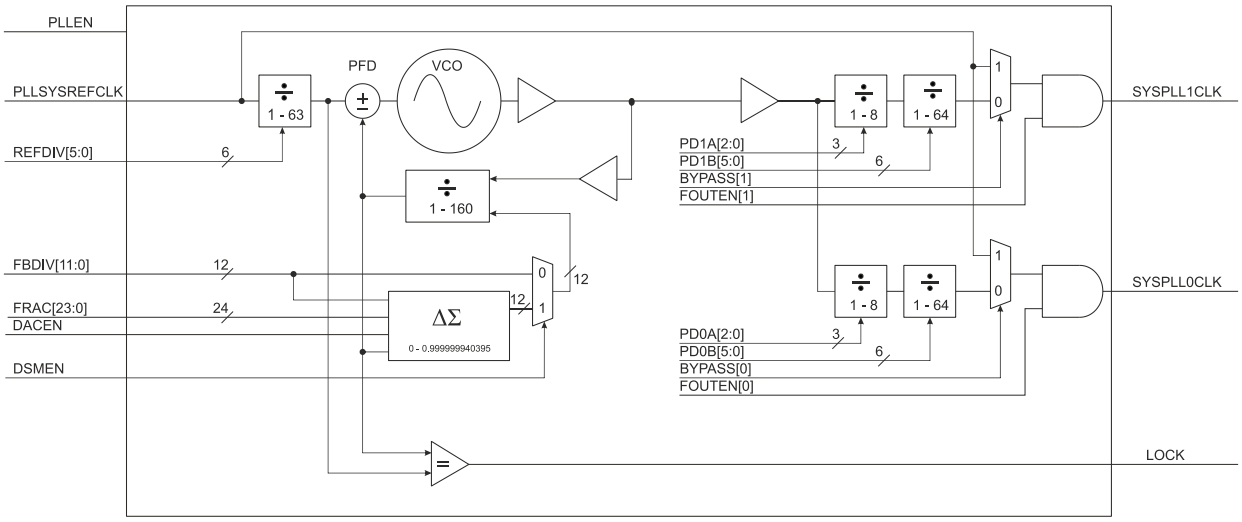


Рисунок 4.2 – Функциональная схема блока PLL

В качестве опорной частоты f_{REF} блока PLL используется PLLSYSREFCLK.

DSMEN – включение дельта-сигма модулятора дробного делителя.

DACEN – включение ЦАП с дробным шумоподавлением в режиме дробного делителя.

Выходная частота F_{OUT0} блока PLL вычисляется согласно формуле 4.1, выходная частота F_{OUT1} блока PLL вычисляется согласно формуле 4.2.

$$f_{OUT0} = \frac{f_{REF}}{REFDIV} \times \frac{FBDIV + \frac{FRAC}{2^{24}} \times DSMEN}{(1+PD0A) \times (1+PD0B)} \quad (4.1)$$

$$f_{OUT1} = \frac{f_{REF}}{REFDIV} \times \frac{FBDIV + \frac{FRAC}{2^{24}} \times DSMEN}{(1+PD1A) \times (1+PD1B)} \quad (4.2)$$

Значение частоты f_{VCO} вычисляется по формуле 4.3.

$$f_{VCO} = \frac{f_{REF} \times (FBDIV + \frac{FRAC}{2^{24}} \times DSMEN)}{REFDIV} \quad (4.3)$$

Коэффициенты REFDIV, FBDIV, FRAC, PD0A, PD0B, PD1A, PD1B задаются соответствующими полями регистра PLLCFG. Существуют следующие ограничения параметров:

- коэффициент деления входной частоты $1 \leq REFDIV \leq 63$;
- коэффициент деления обратной связи (без дробного делителя) $16 \leq FBDIV \leq 160$;
- коэффициент деления обратной связи (с дробным делителем) $20 \leq FBDIV \leq 160$;
- коэффициент деления от 1 до 8 ($0 \leq PD0A$ (PD1A) ≤ 7);
- коэффициент деления от 1 до 64 ($0 \leq PD0B$ (PD1B) ≤ 63);
- входная частота f_{REF} должна находиться в диапазоне от 10 МГц до 30 МГц;
- значение частоты f_{VCO} должно быть в диапазоне (200 – 1600) МГц;
- значение выходной частоты f_{OUT} должно быть в диапазоне 390 кГц – 60 МГц.

Примечание – Настоятельно рекомендуется максимизировать значение PD0A в паре делителей PD0A, PD0B. Аналогично и для пары делителей PD1A, PD1B.

Настройку блока PLL необходимо делать до того, как его тактовый сигнал будет выбран в качестве рабочей частоты системы или одного из блоков. Сначала необходимо отключить прохождение входного сигнала сквозь блок PLL путём сброса BYP бита в регистре PLLSYSCFG0. Затем установить делители REFDIV и FBDIV, соблюдая указанные выше ограничения. Если необходим дробный делитель, то необходимо установить бит DSMEN регистра PLLSYSCFG0 и записать коэффициент FRAC в регистр PLLSYSCFG1. Если дробный делитель не используется, то бит DSMEN должен быть сброшен.

Следующим шагом следует установить желаемые коэффициенты деления PD*A и PD*B, и разрешить генерацию выходной частоты путем установки бита FOUTEN* и PLLEN.

При правильной установке всех значений и выходе блока PLL на рабочий режим будет установлен бит LOCK в регистре PLLSYSSTAT.

Для тактирования некоторых периферийных блоков микроконтроллера (WDT, UART, QSPI, SPI, ADCSAR и ADCSD) можно выбрать как сигнал SYSPLL0CLK, так и SYSPLL1CLK, независимо друг от друга.

4.3 Система слежения за тактовыми сигналами

Система слежения осуществляет контроль источников тактовых сигналов и позволяет обрабатывать исключительные ситуации, связанные с их пропаданием (срыв генерации блока PLL, ненадежный контакт с внешним резонатором и т. п.).

Текущий статус тактовых сигналов можно установить, прочитав соответствующие биты CLKGOODx и CLKERRx регистра CLKSTAT, где x – принимает значения от 0 до 3 (0 – HSICLK, 1 – HSECLK, 2 – SYSPLL0CLK, 3 – LSICLK). Бит CLKGOODx будет установлен, если соответствующий тактовый сигнал стабильно работает, и будет сброшен при его сбое. Бит CLKERRx имеет обратное поведение – при сбое устанавливается, сбрасывается при нормальной работе тактового сигнала.

Время реакции системы слежения на исчезновение тактового сигнала настраивается с помощью полей регистра SECPRD, и по умолчанию равно 256 тактам сигнала REFCLK (HSICLK). Реакция на любое пропадание тактового сигнала на меньшее время будет отсутствовать.

Минимальная частота, за которой может осуществляться слежение – 16 Гц.

Значение минимально возможного времени реакции вычисляется по формуле (результат округляется до целого в большую сторону)

$$SECPRD\_MIN = (4 \times T_{REF} + 6 \times T_{CLK})/T_{REF}, \quad (4.2)$$

где T_{REF} – период опорной тактовой частоты сигнала HSICLK;

T_{CLK} – период контролируемой тактовой частоты.

Например, если необходима максимально быстрая реакция на пропадание частоты 50 МГц на выходе блока PLL (T_{CLK} = 10 нс) при частоте сигнала REFCLK 1 МГц (T_{REF} = 1 мкс), то необходимо в поле PLLCLK регистра SECPRD записать значение SECPRD_MIN = 5, вычисленное по формуле (4.2).

Стоит отметить, что SECPRD_MIN является лишь временем детектирования сбоя. Сам переход в аварийный режим (переключение системной частоты на сигнал HSICLK) займет дополнительно не более 14 тактов сигнала HSICLK.

Кроме слежения за каждым из источников тактового сигнала по отдельности, система отслеживания позволяет контролировать текущий системный тактовый сигнал SYSCLK.

Включение контроля сигнала SYSCLK осуществляется установкой бита SECEN регистра SYSCLKCFG.

При сбое будет осуществлен аварийный переход системной частоты на сигнал HSICLK.

Текущий статус системного тактового сигнала доступен в бите SYSFAIL регистра SYSCLKSTAT: при сбое этот бит установится и будет держаться установленным, пока пользователь вручную не перейдет на любой из доступных стабильных источников, записав в поле SYSSEL регистра SYSCLKCFG новое значение (при сбое поле сохраняет старое значение источника системной частоты). В том числе возможен переход и на сигнал HSICLK, несмотря на то, что в аварийном режиме схема уже тактируется этим сигналом – это позволит сбросить бит SYSFAIL регистра SYSCLKSTAT.

Прямой переход из аварийного состояния к тактированию от вызвавшего сбой, но уже восстановившегося источника – невозможен. Если это необходимо сделать, то сначала нужно перейти на один из стабильных источников (например, сигнал HSICLK), и лишь затем начать переход на бывший «сбойным», но восстановившийся источник.

При переходе микроконтроллера в аварийное состояние по тактированию требуется обязательно перейти на стабильный источник тактирования (бит xGOOD для него будет установлен), например, на сигнал HSICLK (SYSSEL = 0). Если аварийный источник возобновил тактирование, то снова перейти на него можно, записав поле SYSSEL регистра SYSCLKCFG на номер соответствующего источника.

4.4 Сигналы сброса

Сброс может осуществляться внешним сигналом с вывода RST (активный уровень сигнала сброса – низкий). При включении питания сброс контроллера осуществляется с помощью блока POR, входящего в состав домена батарейного питания и домена питания ядра.

Примечание – В случае перехода в режим глубокого сна (PowerDown), в котором отключается питание батарейного домена, будет неактивен сброс микроконтроллера внешним сигналом с вывода RST. Из режима глубокого сна можно выйти только с помощью настроенных внешних входов пробуждения WAKEUP0 – WAKEUP2, нарушения целостности входов датчиков вскрытия AT_IN0 - AT_IN2 или сигнала пробуждения ALARM блока RTC.

4.5 Тактирование и сброс периферийных блоков

Для некоторых периферийных блоков система тактирования предусматривает выбор независимого тактового источника. У блоков WDT, IWDT, UART, QSPI, SPI, ADCSAR, ADCSD и блока формирования внешнего сигнала CLKOUT есть соответствующие регистры, в которых можно включать тактирование, выбирать его источник и настраивать отключаемый делитель. Источник тактирования выбирается независимо для каждого из блоков с помощью его собственного селектора, показанного на рисунке 4.4.

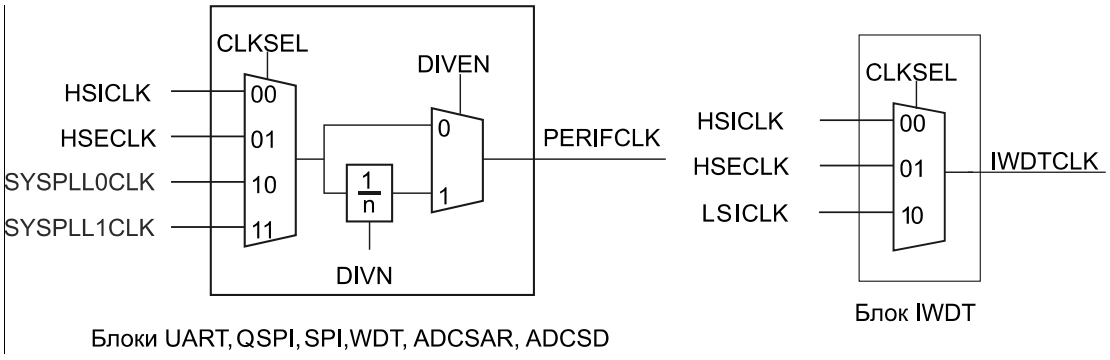


Рисунок 4.4 – Схема селектора дополнительного синхросигнала блока

Для остальных блоков, а именно CMP, TMR32, TMR, TRNG, I2C и для блоков, относящихся к AHB периферии (GPIO, CAN, CRC0, CRC1, HASH, CRYPTO, USB),

тактирование подается установкой соответствующего бита в CGCFGAPB (CGCFGAHB), см. таблицу 4.3.

Таблица 4.3 – Тактовые сигналы периферийных блоков

Периферийный блок	Тактовый сигнал	Делитель	Управляющий регистр
ADCSAR	ADCSARCLK	1-63	ADCSARCFG
ADCSD	ADCSDCLK	1-63	ADCSDCFG
QSPI	QSPICLK	1-63	QSPICFG
SPI	SPICLK	1-63	SPICFG
UART	UARTCLK	1-63	UARTCFG
WDT	WDTCLK	1-63	WDTCFG
CLOCKOUT	CLKOUT	1-65535	CLKOUTCFG
TMR32	PCLK	-	CGCFGAPB, RSTDISAPB
TMR	PCLK	-	CGCFGAPB, RSTDISAPB
I2C	PCLK	-	CGCFGAPB, RSTDISAPB
CMP	PCLK	-	CGCFGAPB, RSTDISAPB
GPIO	HCLK	-	CGCFGAHB, RSTDISAHB
CAN	HCLK	-	CGCFGAHB, RSTDISAHB
USB	HCLK	-	CGCFGAHB, RSTDISAHB
CRC0, CRC1	HCLK	-	CGCFGAHB, RSTDISAHB
HASH	HCLK	-	CGCFGAHB, RSTDISAHB
CRYPTO	HCLK	-	CGCFGAHB, RSTDISAHB

Тактирование блока USB осуществляется отдельным блоком PLL, аналогичным SYSPLL. Регистры настройки USBPLL представлены в приложении А.23. Для работы блока USB выходную частоту блока USBPLL необходимо настроить на частоту 60 МГц.

По умолчанию на всех периферийных блоках отключено тактирование, и все они находятся в состоянии сброса. Для начала работы нужно подать тактовый сигнал, а также вывести блок из состояния сброса, осуществив запись единиц в соответствующие биты регистров CGCFGAPB (CGCFGAHB) и, RSTDISAPB (RSTDISAHB).

5 Блок управления энергопотреблением PMU

В микроконтроллере предусмотрена возможность конфигурирования режимов функционирования для уменьшения потребляемой энергии в тех задачах, где ядро не должно постоянно работать, например, во время ожидания прихода внешнего события. В связи с тем, что в микроконтроллере отдельно выделен батарейный домен питания, то управление режимами блоков, входящих в его состав осуществляется регистрами PMU_RTC (см. приложение А.3). Управление режимом функционирования остальных блоков осуществляется регистрами PMU_SYS (см. Приложение А.2).

Помимо этого, предусмотрено два способа перехода в режим пониженного энергопотребления:

- использование команды WFI (Wait For Interrupt);
- непосредственно через запись в регистры.

Выход из режима пониженного энергопотребления может быть осуществлен по одному из событий:

- события wake (WAKEUP0-WAKEUP2, IWDT, CMP0, CMP1);
- событие монитора вскрытия (нарушение целостности цепей AT_IN0 - AT_IN2, расхождение частот RC-генератора);
- событие сигнализатора Alarm;

Переключением питания ядра и периферии, входящей в состав домена батарейного питания между основным питанием и батарейным управляет контроллер питания APC.

5.1 Расширенный контроллер питания APC

Особенности:

- широкий диапазон входного напряжения от 1,7 до 3,6 В;
- работа в режимах: обычное, пониженное и ультранизкое потребление;
- возможность задания верхнего и нижнего уровней питающих напряжений в режимах пониженного и ультранизкого потребления;
- детектор снижения уровня напряжения VAON ниже 1,80 В с формированием события WKUP[4] контроллера WAKE;
- возможность работы от основного и батарейного источника питания.

На рисунке 5.1 показана структурная схема питания микроконтроллера.

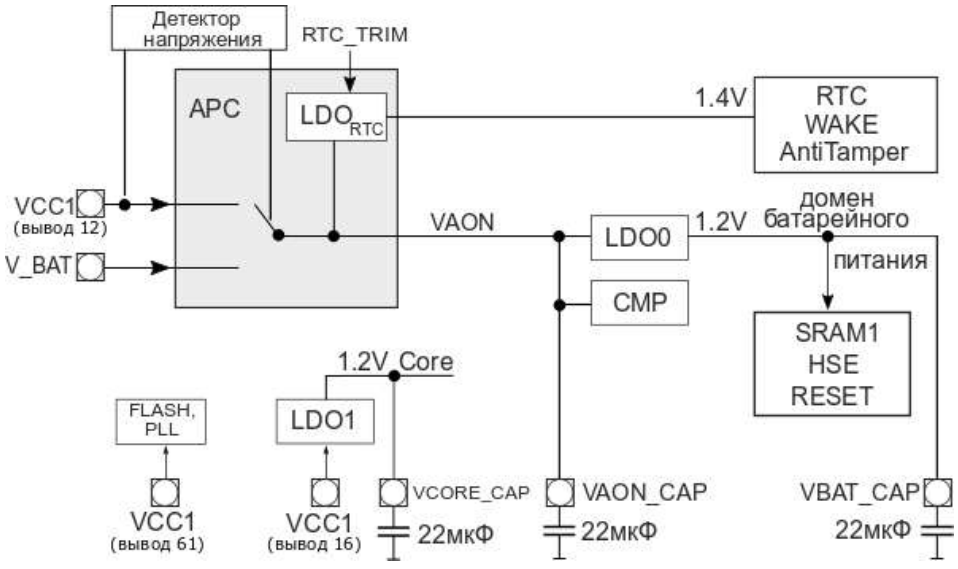


Рисунок 5.1 – Структурная схема питания микроконтроллера

Контроллер питания включает в себя все необходимые вспомогательные функции, выходы опорного напряжения и контроль питания. Возможности APC делают его основой блока управления питанием (PMU). Контроллер питания имеет высокий коэффициент подавления нестабильности питания (PSRR), мониторинг входного питания (VI) и независимую развязку опорного напряжения без необходимости использования дополнительных выводов корпуса.

При включении микроконтроллера контроллер APC переключает внутренне питание VAON на вход батарейного питания V_BAT. При наличии на выводе VCC1(№ 12) напряжения, превышающего V_BAT, контроллер APC переключает внутренне питание VAON на вход питания VCC1(№ 12). При условии V_BAT > VCC1(вывод № 12) формируется событие WKUP[5] контроллера WAKE.

Регуляторы напряжения LDO0 и LDO1

Для обеспечения стабилизированного напряжения питания ядра и периферийных блоков, входящих в домен батарейного питания, предусмотрены регуляторы напряжения LDO1 и LDO0, соответственно. Регуляторы напряжения могут работать в трех режимах: обычном, пониженного потребления и ультранизкого потребления. В обычном режиме регуляторы напряжения работают непрерывно, поддерживая заданный уровень напряжения. В режимах пониженного и ультранизкого потребления регуляторы включаются при снижении уровня выходного напряжения ниже заданного битовым полем LDOLV и выключаются по достижению уровня заданного битовым полем LDOHV. После выключения регуляторов LDO0 и LDO1 периферийные блоки работают за счет энергии, накопленной во внешних конденсаторах, подключенных к выводам VCORE_CAP и VBAT_CAP.

Регулятор напряжения LDO0 обеспечивает питанием домен батарейного питания, включающий следующие блоки: HSI (1МГц), CMP, IWDT, SRAM1 (64K), HSE, RESET.

Регулятор напряжения LDO1 обеспечивает питанием домен питания Core, включающий следующие блоки: PLL, SRAM0 (256K), DMA, FLASH (1Mb), TMR32, TMR(0-2), USB(PLL_USB), GPIO, CRC, HASH, UART, I2C, CAN, Risc-V Core, CRYPTO, SPI, QSPI, TRNG, WDT.

Значение напряжений на выходе обоих LDO в обычном режиме потребления по умолчанию составляет 1,2 В. Управление уровнем напряжения регуляторов в обычном режиме осуществляется битовым полем VPROG регистра RTC_WAKECFG.

Выбор режима пониженного потребления для LDO0 и LDO1 осуществляется записью в битовое поле LP регистра RTC_CFG1. Для переключения в режим пониженного потребления регулятора LDO0 необходимо записать значение 0x1, для LDO1 – значение 0x2, для переключения LDO0 и LDO1 - значение 0x3. На рисунке 5.2 представлена диаграмма работы регуляторов в режиме пониженного потребления.

Выбор режима ультранизкого потребления для LDO0 и LDO1 осуществляется записью в битовое поле ULP регистра RTC_CFG1. Для переключения в режим ультранизкого потребления регулятора LDO0 необходимо записать значение 0x1, для LDO1 – значение 0x2, для переключения LDO0 и LDO1 - значение 0x3. На рисунке 5.3 представлена диаграмма работы регуляторов в режиме ультранизкого потребления.

Для управления верхней и нижней границей напряжения LDO при работе в режимах низкого и ультранизкого потребления предусмотрена возможность записи значений в битовые поля LDOHV и LDOLV регистра RTC_CFG1 соответственно.

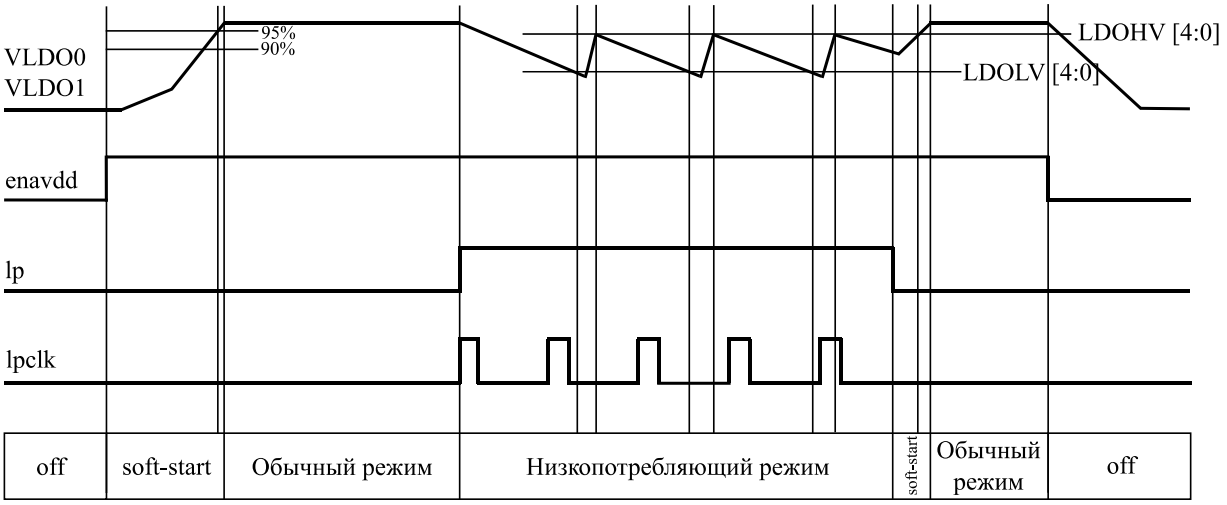


Рисунок 5.2 – Диаграмма работы регуляторов в режиме пониженного потребления

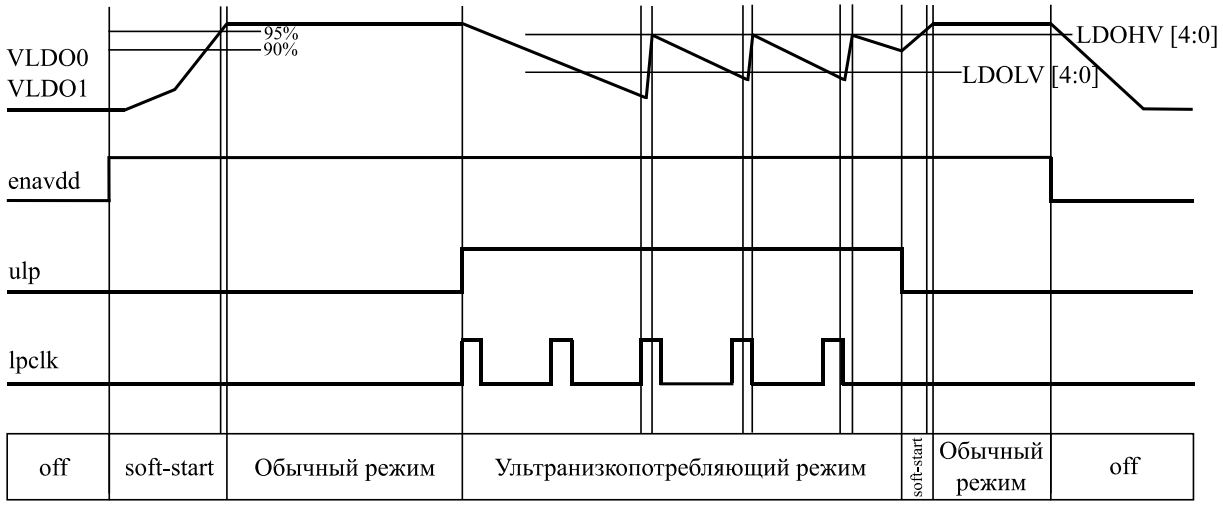


Рисунок 5.3 – Диаграмма работы регуляторов в режиме ультранизкого потребления

Описание сигналов на рисунках 5.2 и 5.3:

enavdd – внутренний сигнал разрешения работы регулятора;

lp, ulp – сигналы перехода в режим пониженного и ультранизкого потребления, соответственно;

lpclk – сигнал синхронизации для активации регулировки уровня выходного напряжения регуляторов в режиме ультранизкого потребления.

Соответствие значений битовых полей LDOHV, LDOLV и VPROG уровню напряжения регуляторов приведено в таблице 5.1.

Воздействие на регистр RTC_CFG1 также происходит при переходе в режим WFI и при выходе из него. Для конфигурации перехода в режим WFI, а также выхода из него предусмотрены регистры WFI_ENTR и WFI_EXIT.

В режиме ультранизкого энергопотребления для снижения собственного потребления регуляторов LDO используется сигнал с высоким активным уровнем lpclk, формируемый блоком RTC. Во время высокого уровня сигнала lpclk функция регулировки уровня выходного напряжения активна. Длительность высокого уровня имеет возможность регулировки с помощью бита LS регистра RTC_CFG0 блока RTC. В момент низкого уровня сигнала lpclk функция регулировки выходного напряжения не активна, а уровень напряжения фиксирован.

Таблица 5.1 – Соответствие уровня напряжения регуляторов значению битовых полей LDOHV, LDOLV и VPROG

Значение битового поля	Уровень напряжения, В	Значение битового поля	Уровень напряжения, В	Значение битового поля	Уровень напряжения, В
0	0,800	10	1,000	20	1,200
1	0,820	11	1,020	21	1,220
2	0,840	12	1,040	22	1,240
3	0,860	13	1,060	23	1,260
4	0,880	14	1,080	24	1,280
5	0,900	15	1,100	25	1,300
6	0,920	16	1,120	26	1,320
7	0,940	17	1,140		
8	0,960	18	1,160		
9	0,980	19	1,180		
Примечание: корректная работа микроконтроллера во всем диапазоне напряжений и температур гарантируется при значении битового поля не менее 5, что соответсвует напряжению 0,9 В.					

Регулятор напряжения LDO_RTC

Регулятор напряжения LDO_RTC обеспечивает питанием модуль RTC (Alarm, RC генератор 32 КГц (LSI), RTC_XTAL), WAKEUP, монитор вскрытия, 16 регистров RTC, регистры конфигурации PMU_RTC. Регулятор напряжения LDO_RTC работает всегда при наличии напряжения VCC1 (вывод № 12) или VBAT. Регулятор напряжения LDO_RTC формирует фиксированное выходное напряжение номиналом 1,4 В.

Предусмотрена температурная коррекция выходного напряжения регулятора LDO_RTC с помощью битового поля TEMPTRIM регистра RTC_TRIM.

5.2 Управление энергопотреблением батарейного домена

Блок PMU_RTC находится в домене питания VBAT, и позволяет осуществлять процесс подсчета временных интервалов, пробуждение контроллера по внешним событиям, а также производить управление питанием доменов VBAT и CORE.

В основе блока PMU_RTC находится контроллер RTC-APC. Сам контроллер RTC-APC обеспечивает работу часов реального времени, управление линейными стабилизаторами и пробуждение по внешним событиям. Контроль за режимами потребления осуществляет блок PMU.

PMU_RTC позволяет (без участия пользователя) осуществлять вход в режимы пониженного энергопотребления по команде ядра WFI. Для активации управления энергопотреблением по команде WFI, необходимо записать бит WFI_PDEN.EN = 1. Без включения этого режима команда WFI будет только приостанавливать работу ядра (режим IDLE).

После установки бита WFI_PDEN.EN и поступления команды WFI, контроллер постепенно будет выключать периферийные блоки, которые указаны в регистре PMU_VBATPER_WFI:

- DACPD – выключение опорного источника для компараторов;
- CMPxPD – выключение компараторов 0 и 1.

Если работа какого-либо блока, описанного выше, не требуется, то его можно перевести принудительно в неактивное состояние записью соответствующих битов регистра PMU_VBATPER_FORCE. Структура битовых полей регистра PMU_VBATPER_FORCE аналогична структуре битовых полей регистра PMU_VBATPER_WFI.

Для управления питанием периферийных блоков, находящихся в домене питания ядра, используют регистры PDEN, PDEN_FORCE, находящиеся в блоке PMUSYS. Регистр PDEN имеет следующие поля:

- PLL – при поступлении команды WFI, при установленном бите PLL переходит в режим пониженного энергопотребления;
- USB – отключается PHY блок USB для экономии электроэнергии;
- ADC – блок АЦП переводится в режим пониженного потребления;
- FLASH – контроллер памяти переводится в режим пониженного энергопотребления;
- SCM – системный менеджер тактовой частоты переводится на источник опорной частоты (на внутренний RC-генератор 1 МГц).

Для автоматического управления доменами питания при использовании команды WFI используются регистры WFI_ENTR и WFI_EXIT. Содержимое регистра WFI_ENTR используется для управления LDO при входе в режим пониженного энергопотребления, а содержимое WFI_EXIT, соответственно, при выходе. Регистры имеют идентичные поля:

- LDOxEN – бит включения/выключения соответствующего LDOx;
- LDOxLP – бит включения/выключения пониженного потребления, соответствующего LDOx;

LDOxULP – бит включения/выключения ультра-пониженного потребления соответствующего LDOx;

VL – минимальное значение напряжения, выдаваемое обоими LDO в режимах пониженного напряжения;

VH – максимальное значение напряжения, выдаваемое обоими LDO в режимах пониженного напряжения.

Данные значения битов переписываются, при входе или выходе в режим пониженного потребления, в регистр CONFIG1 контроллера PMURTC.

Для пробуждения контроллера используются события:

- событие TIME = ALARM;
- WAKEUP[2:0] – внешние входы контроллера для отслеживания перепадов сигналов;
- AT_IN[2:0] – внешние выводы контроля целостности линий;
- VBAKOVERVI – событие показывающее, что напряжение на батарее больше внешнего напряжения питания;
- UVLOZ – событие понижения напряжения батареи и внешнего питания ниже порога 1,8 В;
- WKCOMP – источник событий, объединяющий события от периферии VBAT домена: CMPx – компараторов, IWDT – сторожевого таймера домена VBAT, а также внешнего сигнала сброса EXTRST.

Для управления реакцией на события от периферийных блоков VBAT домена используются регистры PMU_WK3EN и PMU_WK3STAT. В данных регистрах имеются поля для соответствующей периферии CMPx, IWDT и внешнего сигнала сброса EXTRST. При записи в соответствующий бит регистра PMU_WK3EN разрешается реакция от соответствующей периферии, а в регистре PMU_WK3STAT устанавливается соответствующий бит от периферии, вызвавшей событие. Бит необходимо сбросить после программной обработки события путем записи логической единицы в соответствующий бит регистра.

Регистр PMU_WCYC необходим для задания циклов ожидания при обращении к регистрам блока RTC-APC. Так как RTC-APC не может работать на частотах обращения к нему больше 1 МГц, то для успешного доступа к регистрам необходимо ожидание определенных временных интервалов, заданных в тиках системной частоты. Если контроллер работает на частоте 50 МГц, то для доступа необходимо записать в регистр PMU_WCYC значение 50 перед началом работы с регистрами RTC-APC.

Все WAKE события отображаются в регистрах RTC_HISTORY.

5.3 Настройка регистров для режимов работы с WFI

Возможны пять основных режимов работы с WFI:
- режим ожидания (IDLE_MODE);
- режим ожидания низкопотребляющий (LP_IDLE_MODE, ULP_IDLE_MODE);
- режим остановки ядра (STOP_MODE);
- режим остановки ядра низкопотребляющий (LP_STOP_MODE, ULP_STOP_MODE);
- режим сна (POWEROFF_MODE).

После выхода из режимов IDLE_MODE, LP_IDLE_MODE, STOP_MODE, LP_STOP_MODE микроконтроллер восстановить работу регуляторов в соответствии с конфигурацией регистра WFI_EXIT.

Основные режимы работы микроконтроллера представлены в таблице 5.2.

Таблица 5.2 – Таблица режимов работы микроконтроллера

Блоки, входящие в состав микроконтроллера		Режимы работы микроконтроллера					
		RUN	IDLE	LP(ULP)_IDL	STOP	LP(ULP)_STO	POWEROFF
Функционирование блоков	RTC (Alarm, RC генератор 32 КГц (LSI), RTC_XTAL), WAKEUP, монитор вскрытия, 16 регистров RTC, регистры PMU_RTC	✓	✓	✓	✓	✓	✓
	Risc-V Core	✓	-	-	-	-	-
	SRAM0 (256K)	✓	✓	✓	-	-	-
	SRAM1 (64K)	✓	✓	✓	✓	✓	-
	PLL[2)]	✓	✓	-	-	-	-
	FLASH (1Mb) [2)]	✓	✓	✓[1)]	-	-	-
	ADCSAR, ADCSD, DMA, TMR32, TMR(0-2), USB(PLL_USB), GPIO, CRC, HASH, UART, I2C, CAN, CRYPTO, SPI, QSPI, TRNG, WDT	✓	✓	✓	-	-	-
	HSI (1МГц), CMP, IWDT, HSE, RESET[3)]	✓	✓	✓	✓	✓	-
События пробуждения	Любое разрешенное прерывание от периферийного блока	✓	✓	-	-	-	-
	События периферии домена батарейного питания (CMP, IWDT, секундный таймер, RESET) [4)]	✓	✓	✓	✓	✓	-
	Внешние события WAKEUP, события монитора вскрытия, событие TIME = ALARM (событие разрешено по умолчанию) [4)].	✓	✓	✓	✓	✓	✓
Время выхода из режима, мкс[5)]		-	40	60	1750	1750	1800

Примечания

1 В режиме LP(ULP)_IDLE Flash доступна только для чтения.

2 Блоки FLASH и PLL могут быть переведены в режим PowerDown через регистр PMUSYS->PDENFORCE.

3 Блоки HSI, HSE, CMP могут быть переведены в режим PowerDown через регистр PMURTC->PMU_VBATPER_FORCE.

4 В режимах IDLE и LP(ULP)_IDLE пробуждение от событий периферии домена батарейного питания и внешних событий WAKEUP будет функционировать только при разрешенном прерывании PMURTC.

5 Время выхода из режима – время от события пробуждения до начала выполнения команд ядра. Измерения для режимов IDLE и LP(ULP)_IDLE проводились при системной частоте HSE =16 МГц и PMURTC->PMU_WCYC = 8.

Режим ожидания (IDLE_MODE)

Необходимо установить следующие значения битовых полей: WFI_PDEN.EN = 1, WFI_ENTR.LDO0EN = 1, WFI_ENTR.LDO1EN = 1. Остальные значения полей регистров равны нулю.

Разрешение перехода в PowerDown по команде WFI, оба LDO будут включены, питание подается и на ядро, и на периферию, рабочее напряжение 1.2 В (в соответствии со значением битового поля VPROG регистра RTC_WAKECFG).

Микроконтроллер может возобновить работу по любому из разрешенных прерываний, событий WAKE или по внешнему сигналу RESET.

Режим ожидания низкопотребляющий (LP_IDLE_MODE, ULP_IDLE_MODE)

Для перехода в режим LP_IDLE_MODE необходимо установить следующие значения битовых полей: WFI_PDEN.EN = 1, WFI_ENTR.LDO0EN = 1, WFI_ENTR.LDO1EN = 1, WFI_ENTR.LDO0LP = 1, WFI_ENTR.LDO1LP = 1, WFI_ENTR.VL = 5 (0,9 В), WFI_ENTR.VH = 10 (1,0 В).

Для перехода в режим ULP_IDLE_MODE необходимо установить следующие значения битовых полей: WFI_PDEN.EN = 1, WFI_ENTR.LDO0EN = 1, WFI_ENTR.LDO1EN = 1, WFI_ENTR.LDO0ULP = 1, WFI_ENTR.LDO1ULP = 1, WFI_ENTR.VL = 5 (0,9 В), WFI_ENTR.VH = 10 (1,0 В).

Необходимо программно переключить системную частоту на HSE, HSI или LSI, и выключить PLL. После выполнения команды WFI произойдет изменение рабочих напряжений регуляторов LDO0 и LDO1. В данном режиме регуляторы работают в соответствии с диаграммой, представленной на рисунке 5.2 (режим LP_IDLE_MODE) или на рисунке 5.3 (режим ULP_IDLE_MODE).

Микроконтроллер может возобновить работу по любому из разрешенных прерываний, событий WAKE или по внешнему сигналу RESET.

Режим остановки ядра (STOP_MODE)

Необходимо установить следующие значения битовых полей: WFI_PDEN.EN = 1, WFI_ENTR.LDO0EN = 1. Остальные значения полей регистров равны нулю.

После выполнения команды WFI выключается домен ядра с соответствующей периферией. Остается работать домен батарейного питания VBAT (HSI, CMP (DAC), IWDT, SRAM1, HSE) с номинальным напряжением 1.2 Вольт (в соответствии со значением битового поля VPROG регистра RTC_WAKECFG).

Микроконтроллер может возобновить работу по одному из разрешенных событий:
- внешние события WAKEUP;
- событие TIME = ALARM (событие разрешено по умолчанию);
- события монитора вскрытия;
- события периферии домена батарейного питания;
- RESET.

Примечание – После пробуждения микроконтроллера из режима остановки ядра будет включен регулятор LDO1 в режиме, соответствующем регистру WFI_EXIT и выполнение программы начнется с адреса 0x80000000.

Режим остановки ядра низкопотребляющий (LP_STOP_MODE, ULP_STOP_MODE)

Для перехода в режим LP_STOP_MODE необходимо установить следующие значения битовых полей: WFI_PDEN.EN = 1, WFI_ENTR.LDO0EN = 1, WFI_ENTR.LDO0LP = 1, WFI_ENTR.VL = 5 (0,9 В), WFI_ENTR.VH = 10 (1,0 В).

Для перехода в режим ULP_STOP_MODE необходимо установить следующие значения битовых полей: WFI_PDEN.EN = 1, WFI_ENTR.LDO0EN = 1, WFI_ENTR.LDO0ULP = 1, WFI_ENTR.VL = 5 (0,9 В), WFI_ENTR.VH = 10 (1,0 В).

После выполнения команды WFI произойдет выключение домена ядра с соответствующей периферией и изменение рабочих напряжений регулятора LDO0. В данном режиме регулятор LDO0 работает в соответствии с диаграммой, представленной на рисунке 5.2 (режим LP_STOP_MODE) или на рисунке 5.3 (режим ULP_STOP_MODE).

Микроконтроллер может возобновить работу по одному из разрешенных событий:

- внешние события WAKEUP;
- событие TIME = ALARM (событие разрешено по умолчанию);
- события монитора вскрытия;
- события периферии домена батарейного питания;
- RESET.

Примечание – После пробуждения микроконтроллера из низкопотребляющего режима остановки ядра будет включен регулятор LDO1 в режиме, соответствующем регистру WFI_EXIT и выполнение программы начнется с адреса 0x80000000.

Режим сна (POWEROFF_MODE)

Необходимо установить следующие значения битовых полей: WFI_PDEN.EN = 1, WFI_ENTR.LDO0EN = 0, WFI_ENTR.LDO1EN = 0.

Выключаются: LDO0, LDO1, PLL, HSI, HSE, SRAM0, SRAM1. Остается работать только блок RTC, 16 регистров RTC_REG[] сохраняют значания и отслеживаются внешние события WAKEUP, монитора вскрытия и сигнализатор RTC (Alarm). Микроконтроллер может возобновить работу по одному из разрешенных событий:

- внешние события WAKEUP;
- событие TIME = ALARM (событие разрешено по умолчанию);
- события монитора вскрытия.

Примечание – После пробуждения микроконтроллера из режима сна будут включены оба регулятора LDO0 и LDO1 в обычном режиме и выполнение программы начнется с адреса 0x80000000.

6 Организация памяти

Память микроконтроллера имеет предопределенное 32-разрядное адресное пространство с областями: программ, данных, периферии и внутренних ресурсов, жестко соединенных с процессором. Адресное пространство разбито на несколько областей, см. таблицу 6.1.

Таблица 6.1 – Общая организация памяти микроконтроллера

Адресное пространство	Описание
8000_0000h – 801F_FFFFh	Flash-память (FLASH) 1 Мбайт
4000_0000h – 4003_FFFFh	Внутреннее ОЗУ0 256 Кбайт
2000_0000h – 3801_FFFFh	Регистры управления периферийными блоками
1000_0000h – 1000_FFFFh	Внутреннее ОЗУ1 64 Кбайт
0C00_0000h – 0CFF_FFFFh	Регистры PLIC
0200_0000h – 0200_FFFFh	Регистры CLINT
0000_0000h – 0000_0FFFh	Регистры Debug
Примечание – Младшие 128 Кбайт ОЗУ0 подключены к интерфейсу TCM-A, а старшие 128 Кбайт - к TCM-B. Для получения максимальной производительности при исполнении из ОЗУ0 рекомендуется переменные и исполняемый код располагать в регионах, подключенных к разным интерфейсам TCM.	

Внутреннее ОЗУ1 размером 64 Кбайт подключено к батарейному домену питания. Регистры периферийных блоков микроконтроллера доступны в адресном пространстве 2000_0000h – 3801_FFFFh. Карта размещения блоков микроконтроллера в памяти представлена в таблице 6.2.

Таблица 6.2 – Карта регистров периферийных блоков

Адресное пространство	Обозначение блока	Описание
1	2	3
3801_2000h – 3801_2FFFh	IWDT	Сторожевой таймер
3801_1000h – 3801_1FFFh	PMURTC	Блок управления питанием RTC
3801_0000h – 3801_0FFFh	CMP	Блок аналоговых компараторов
3001_2000h – 3001_2FFFh	ADCSD	Блок сигма-дельта АЦП
3001_0000h – 3001_0FFFh	ADCSAR	Блок АЦП последовательного приближения
3000_F000h – 3000_FFFFh	PMUSYS	Блок управления питанием системы
3000_E000h – 3000_EFFFh	RCU	Блок управления сбросом и синхронизацией
3000_D000h – 3000_DFFFh	FLASH	Контроллер Flash-памяти
3000_C000h – 3000_CFFFh	DMA	Контроллер прямого доступа к памяти
3000_B000h – 3000_BFFFh	WDT	Сторожевой таймер
3000_A000h – 3000_AFFFh	UART4	Последовательный приемопередатчик TUART0
3000_9000h – 3000_9FFFh	UART3	Последовательный приемопередатчик UART3
3000_8000h – 3000_8FFFh	UART2	Последовательный приемопередатчик UART2
3000_7000h – 3000_7FFFh	UART1	Последовательный приемопередатчик UART1

3000_6000h – 3000_6FFFh	UART0	Последовательный приемопередатчик UART0

Продолжение таблицы 6.2

1	2	3
3000_5000h – 3000_5FFFh	I2C	Последовательный интерфейс I2C
3000_4000h – 3000_4FFFh	TRNG	Блок генератора случайных чисел
3000_3000h – 3000_3FFFh	TMR2	Таймер 16-разрядный TMR2
3000_2000h – 3000_2FFFh	TMR1	Таймер 16-разрядный TMR1
3000_1000h – 3000_1FFFh	TMR0	Таймер 16-разрядный TMR0
3000_0000h – 3000_0FFFh	TMR32	Таймер 32-разрядный TMR32
2800_2000h – 2800_2FFFh	GPIOC	Порт C
2800_1000h – 2800_1FFFh	GPIOB	Порт B
2800_0000h – 2800_0FFFh	GPIOA	Порт A
2006_0000h – 2006_FFFFh	SPI1	Последовательный интерфейс SPI1
2005_0000h – 2005_FFFFh	SPI0	Последовательный интерфейс SPI0
2004_0000h – 2004_FFFFh	QSPI	Последовательный интерфейс QSPI
2003_2000h – 2003_2FFFh	HASH	Хеш процессор
2003_1000h – 2003_1FFFh	CRC1	Блок подсчета циклического избыточного кода CRC1
2003_0000h – 2003_0FFFh	CRC0	Блок подсчета циклического избыточного кода CRC0
2002_0000h – 2002_FFFFh	CRYPTO	Блок криптографии
2001_0000h – 2001_FFFFh	USB	Последовательный интерфейс USB
2000_1000h – 2000_1FFFh	CANMSG	Объекты сообщений контроллера CAN
2000_0000h – 2000_0FFFh	CAN	Контроллер CAN

7 Контроллер Flash-памяти

7.1 Flash-память

Flash-память может использоваться для хранения программ и данных пользователя. Размер основной Flash-памяти составляет 1 Мбайт (256 страниц по 4 Кбайт), и в адресном пространстве она занимает диапазон с 8000_0000h по 800F_FFFFh.

Чтение Flash-памяти осуществляется через две шины AHB: I-code (для команд) и D-code (для данных). Чтение D-code шины имеет приоритет. На обеих шинах при попытке записи в любую область, чтении из несуществующей области, чтении во время, когда Flash занята (стирание, запись), транзакция проходит успешно с неопределенными данными на выходе.

Память доступна для чтения, записи, полного и постраничного стирания через регистры данных DATAn (n от 0 до 3), адреса ADDR, команд CMD, статуса STAT блока FLASH. Запись необходимо производить в предварительно очищенную ячейку памяти. Стирание памяти осуществляется полностью или постранично.

Минимальное время чтения данных из Flash-памяти составляет до 60 нс (типовое значение задержки – от 30 нс). Поэтому, исходя из выбранной рабочей частоты, следует задать определенное количество дополнительных тактов ожидания, необходимое для стабильного чтения из Flash-памяти.

В таблице 7.1 представлены значения дополнительных тактов ожидания Flash памяти при значении выходного напряжения регулятора LDO1 – 1,2 В. Значения дополнительных тактов ожидания Flash памяти при значении выходного напряжения регулятора LDO1 0,9 В представлены в таблице 7.2. При использовании низкопотребляющего или ультра низкопотребляющего режима для регулятора LDO1 и снижении уровня выходного напряжения до 0,9 В необходимо в регистр е LP установить бит LPEN. В этом случае при обращении к Flash возможны только операции чтения.

Таблица 7.1 – Значения параметра (количество дополнительных тактов ожидания) в зависимости от частоты сигнала SYSCLK при напряжении 1,2 В LDO1

f_{SYSCLK}, не более, МГц	Количество дополнительных тактов ожидания при чтении Flash-памяти
60	1
30	0
Примечание – Значение параметра после сброса равно 1.	

Таблица 7.2 – Значения параметра (количество дополнительных тактов ожидания) в зависимости от частоты сигнала SYSCLK при напряжении 0,9 В LDO1

f_{SYSCLK}, не более, МГц	Количество дополнительных тактов ожидания при чтении Flash-памяти
60	3
45	2
30	1
15	0
Примечание – Значение параметра после сброса равно 1.	

Помимо основной области памяти (1 Мбайт), доступной как через глобальное адресное пространство, так и через регистры контроллера, существует дополнительная NVR область (две страницы по 4 Кбайт в диапазоне 0x0000 – 0x1FFF). В последней ячейке второй страницы NVR области располагается конфигурационное слово микроконтроллера CFGWORD. NVR область доступна для чтения, записи и постраничного стирания только через регистры блока FLASH.

Операция предвыборки

При запросе данных на шине по адресу, по которому не осуществлялась предвыборка, выполняются следующие действия:

1 Сигнал готовности на шине устанавливается в ноль и задерживает транзакцию.

2 По запрашиваемому адресу считываются 4 32-битных слов (128 бит) данных из Flash-памяти. Далее эти данные записываются во внутренний первый буфер.

3 Требуемое слово передается на внутреннюю шину, и сигнал готовности устанавливается в единицу.

4 Сразу после установки сигнала готовности из Flash-памяти считываются 128 бит данных по следующему адресу. Данные сохраняются во втором буфере. Если во время считывания этих данных появляются запросы по адресам, сохраненным в первом буфере, ответ возникает мгновенно, если по другим адресам, то готовность на шине устанавливается в ноль и происходит ожидание завершения считывания во второй буфер и далее возврат к действию 2.

5 Если приходят запросы по адресам, сохраненным в первом буфере, ответ возникает мгновенно, если по адресам, находящимся во втором буфере, ответ также возникает мгновенно. Далее переписывается первый буфер значением второго и считывается следующий адрес из Flash-памяти. Если приходят запросы по адресам не из первого и второго буферов, то возврат к действию 1.

Конфигурационное слово CFGWORD

В дополнительной области Flash-памяти расположено конфигурационное слово CFGWORD, содержащее параметры микроконтроллера. Описание битов представлено в таблице 7.3. Чтение конфигурационного слова контроллером происходит каждый раз после сброса по включению (POR).

Таблица 7.3 – Конфигурационное слово CFGWORD

CFGWORD +1FF0h (смещение относительно начального адреса CFG дополнительной области FLASH)			
Поле	Биты	Описание	
JTAGEN	2	Бит разрешения работы интерфейса JTAG	
		0	Откладка отключена
		1	Откладка включена (по умолчанию)
CFGWE	1	Бит включения защиты CFG области Flash-памяти	
		0	Запрет записи и стирания CFG области
		1	Защита выключена (по умолчанию)
FLASHWE	0	Бит включения защиты основной области Flash-памяти	
		0	Запрет записи основного блока
		1	Защита выключена (по умолчанию)
–	31-3	Зарезервировано	

При установленном бите FLASHWE запрещается запись в соответствующую область памяти через регистровый интерфейс. Операция записи интерпретируется как операция чтения, при этом стандартным образом продолжают работать механизмы флагов из регистра STAT.

7.2 Сервисный сброс всей Flash-памяти

Во время сброса микроконтроллера анализируется состояние вывода SERVEN. Если вывод находится в состоянии логической единицы (подтянут к 3,3 В), то Flash-память

переводится в режим, в котором чтение запрещено (при чтении возвращаются нули). При этом игнорируется состояние бита JTAGEN.

Далее по отладочному интерфейсу JTAG должна быть подана команда записи значения 0000_0100h в регистр SERVCTL блока PMUSYS, после чего будет активировано полное стирание всех областей основной и загрузочной памяти. По завершении процесса стирания в этом же регистре выставится флаг DONE.

Примечание – Если полное стирание не требуется, во время сброса на выводе SERVEN должен удерживаться логический ноль.

8 Микропроцессорное ядро CloudBEAR

Процессорное ядро BM-310 поддерживает систему команд: RV32IMFCN_ZBA_ZBB_ZBC_ZBS. Ядро CloudBEAR BM-310S6 также поддерживает два режима привилегированности: machine и user. Режим user предоставляет механизм изоляции процессов друг от друга и от доверенного кода, исполняемого в режиме machine.

Конвейер BM-310 состоит из двух стадий, на которых выполняются следующие операции:

1 Генерация запроса в подсистему памяти программ (PMS).

2 Чтение фрагмента кода из PMS и извлечение команд. Декодирование команды. Исполнение команды.

Подсистема предварительной обработки команд (FE) отвечает за выполнение операций первой стадии конвейера и частично второй стадии конвейера.

Блок FE состоит из следующих блоков:

1 IFU - Instruction Fetch Unit - содержит логику формирования адреса следующего фрагмента кода.

2 IDU - Instruction Decode Unit - блок предварительного декодирования инструкций.

3 IFQ - Instruction Fetch Queue - очередь фрагментов кода.

На первой стадии блок IFU формирует запросы для считывания фрагментов кода в подсистему памяти программ (PMS), используя информацию о перенаправлении подкачки от IDU, а также подсистемы исполнения команд (BE).

На второй стадии происходит чтение фрагмента кода из памяти программ (PMS). Фрагмент кода может содержать одну или две команды. IDU выполняет предварительное декодирование и помещает фрагмент кода, соответствующий одной команде, в очередь команд (IFQ). Подсистема исполнения команд (BE) получает фрагмент кода из очереди команд (или из байпасса), выполняет полное декодирование команды, считывает операнды из регистрового файла, осуществляет проверку возможности выполнения и выполняет команду на соответствующем исполнительном устройстве.

Выполнение всех команд RV32IMC занимает один такт, кроме команд умножения/деления:

- MUL – 2 такта;
- MULH – 2 такта;
- MULHS– 2 такта;
- MULHSU– 2 такта;
- DIV – от 2 до 16 тактов;
- DIVU – от 2 до 16 тактов;
- REM – от 2 до 16 тактов;
- REMU – от 2 до 16 тактов.

Исполнительное устройство умножения является конвейеризованным и отвечает за исполнение команд MUL/MULH[[S]U]. Исполнительное устройство деления является итеративным и отвечает за исполнение команд DIV[U] и REM[U]. Команды доступа к CSR регистрам исполняются в пустом конвейере.

8.1 Система команд

Поддерживаемые команды представлены в таблице 8.1.

Таблица 8.1 – Система команд процессора

Мнемокод команды	Формат	Операнды	Краткое описание
1	2	3	4
LB	I	rd, rs1, imm	Загрузка байта
LH	I	rd, rs1, imm	Загрузка полуслова
LW	I, Cx	rd, rs1, imm	Загрузка слова
LBU	I	rd, rs1, imm	Загрузка байта без знака
LHU	I	rd, rs1, imm	Загрузка слова без знака
SB	S	rs1, rs2, imm	Сохранение байта
SH	S	rs1, rs2, imm	Сохранение полуслова
SW	S, Cx	rs1, rs2, imm	Сохранение слова
ADD	R, Cx	rd, rs1, rs2	Арифметическое сложение
ADDI	I, Cx	rd, rs1, imm	Арифметическое сложение с непосредственным значением
SUB	R, Cx	rd, rs1, rs2	Арифметическое вычитание
LUI	U	rd, imm	Загрузка верхней части непосредственным значением
AUIPC	U	rd, imm	Арифметическое сложение верхней части PC с непосредственным значением
XOR	R	rd, rs1, rs2	Логическое исключающее “ИЛИ”
XORI	I	rd, rs1, imm	Логическое исключающее “ИЛИ” с непосредственным значением
OR	R, Cx	rd, rs1, rs2	Логическое “ИЛИ”
ORI	I	rd, rs1, imm	Логическое “ИЛИ” с непосредственным значением
AND	R, Cx	rd, rs1, rs2	Логическое “И”
ANDI	I	rd, rs1, imm	Логическое “И” с непосредственным значением
SLL	R	rd, rs1, rs2	Логический сдвиг влево
SLLI	I, Cx	rd, rs1, shamt	Логический сдвиг влево на непосредственное значение
SRL	R	rd, rs1, rs2	Логический сдвиг вправо
SRLI	I	rd, rs1, shamt	Логический сдвиг вправо на непосредственное значение
SRA	R	rd, rs1, rs2	Арифметический сдвиг вправо
SRAI	I	rd, rs1, shamt	Арифметический сдвиг вправо на непосредственное значение
SLT	R	rd, rs1, rs2	Сравнение “<”
SLTI	I	rd, rs1, imm	Сравнение “<” с непосредственным значением
SLTU	R	rd, rs1, rs2	Сравнение “<” с беззнаковым
SLTIU	I	rd, rs1, imm	Сравнение “<” с беззнаковым с непосредственным значением
BEQ	SB, Cx	rs1, rs2, imm	Переход в случае равенства
BNE	SB, Cx	rs1, rs2, imm	Переход в случае неравенства
BLT	SB	rs1, rs2, imm	Переход в случае меньше

Продолжение таблицы 8.1

1	2	3	4
BGE	SB	rs1, rs2, imm	Переход в случае больше или равно
BLTU	SB	rs1, rs2, imm	Переход в случае меньше беззнакового
BGEU	SB	rs1, rs2, imm	Переход в случае больше или равно беззнакового
JAL	UJ, Cx	rd, imm	Переход & link
JALR	UJ, Cx	rd, rs1, imm	Переход & link register
FENCE.I	I	-	Синхронизация потока подкачки команд и потока чтения/записи данных
RDINSTRET	I	rd	Псевдоинструкция (alias) на команду чтения счетчика числа исполненных инструкций
ECALL	I	-	Запрос на выполнение переменного окружения операционной системы
EBREAK	I	-	Передача управления в переменное окружение отладки
RDCYCLE	I	rd	Псевдоинструкция (alias) на команду чтения счетчика количества выполненных циклов процессора
RDCYCLEH	I	rd	Псевдоинструкция (alias) на команду чтения старших бит счетчика количества выполненных циклов процессора
RDTIME	I	rd	Псевдоинструкция (alias) на команду чтения значения низкочастотного системного таймера ядра
RDTIMEH	I	rd	Псевдоинструкция (alias) на команду чтения значения старших бит низкочастотного системного таймера ядра
RDINSTRET	I	rd	Псевдоинструкция (alias) на команду чтения счетчика числа исполненных инструкций
RDINSTRETH	I	rd	Псевдоинструкция (alias) на команду чтения старших бит счетчика числа исполненных инструкций
MUL	R	rd, rs1, rs2	Умножение
MULH	R	rd, rs1, rs2	Умножение с возвратом старших бит результата
MULHSU	R	rd, rs1, rs2	Умножение знакового на беззнаковое с возвратом старших бит результата
MULHU	R	rd, rs1, rs2	Умножение беззнакового на беззнаковое с возвратом старших бит результата
DIV	R	rd, rs1, rs2	Деление
DIVU	R	rd, rs1, rs2	Беззнаковое деление
REM	R	rd, rs1, rs2	Остаток от деления
REMU	R	rd, rs1, rs2	Беззнаковый остаток от деления
BLT	SB	rs1, rs2, imm	Переход в случае меньше
BGE	SB	rs1, rs2, imm	Переход в случае больше или равно
BLTU	SB	rs1, rs2, imm	Переход в случае меньше беззнакового
BGEU	SB	rs1, rs2, imm	Переход в случае больше или равно беззнакового

Продолжение таблицы 8.1

1	2	3	4
JAL	UJ, Cx	rd, imm	Переход & link
JALR	UJ, Cx	rd, rs1, imm	Переход & link register
FENCE.I	I	-	Синхронизация потока подкачки команд и потока чтения/записи данных
RDINSTRET	I	rd	Псевдоинструкция (alias) на команду чтения счетчика числа исполненных инструкций
ECALL	I	-	Запрос на выполнение переменного окружения операционной системы
EBREAK	I	-	Передача управления в переменное окружение отладки
RDCYCLE	I	rd	Псевдоинструкция (alias) на команду чтения счетчика количества выполненных циклов процессора
RDCYCLEH	I	rd	Псевдоинструкция (alias) на команду чтения старших бит счетчика количества выполненных циклов процессора
RDTIME	I	rd	Псевдоинструкция (alias) на команду чтения значения низкочастотного системного таймера ядра
RDTIMEH	I	rd	Псевдоинструкция (alias) на команду чтения значения старших бит низкочастотного системного таймера ядра
RDINSTRET	I	rd	Псевдоинструкция (alias) на команду чтения счетчика числа исполненных инструкций
RDINSTRETH	I	rd	Псевдоинструкция (alias) на команду чтения старших бит счетчика числа исполненных инструкций
MUL	R	rd, rs1, rs2	Умножение
MULH	R	rd, rs1, rs2	Умножение с возвратом старших бит результата
MULHSU	R	rd, rs1, rs2	Умножение знакового на беззнаковое с возвратом старших бит результата
MULHU	R	rd, rs1, rs2	Умножение беззнакового на беззнаковое с возвратом старших бит результата
DIV	R	rd, rs1, rs2	Деление
DIVU	R	rd, rs1, rs2	Беззнаковое деление
REM	R	rd, rs1, rs2	Остаток от деления
REMU	R	rd, rs1, rs2	Беззнаковый остаток от деления
Примечание – rs – регистр источник; rd – регистр приёмник; imm или shamt – непосредственное значение.			

Подробное описание каждой команды и форматы команд приведены в документе «The RISC-V Instruction Set Manual Volume I: User-Level ISA Document Version 2.2».

9 Контроллер прерываний

9.1 Локальный контроллер прерываний (CLINT)

За генерацию запросов (сигналов) программных прерываний (machine software interrupt) и прерываний таймера (machine timer interrupt) отвечает локальный контроллер прерываний (Core Local Interrupt Controller, CLINT). CLINT содержит регистры для управления программными прерываниями и прерываниями таймера. Карта регистров CLINT приведена в таблице 9.1.

Таблица 9.1 – Карта регистров CLINT

Адрес	Название	Описание	Доступ
0x0000	CLINT_MSIP0	Регистр запроса/снятия программного прерывания	R/W
0x4000	CLINT_MTIMECMP	Регистр сравнения таймера	R/W
0xBFF8	CLINT_MTIME	Регистр значения таймера	R

Запрос на программное прерывание может быть установлен/снят через соответствующий регистр CLINT_MSIP0, нулевой бит которого (MSIP) отражает ждущее программное прерывание.

Регистр CLINT_MTIMECMP0 содержит 64-битное значение таймера. Значение увеличивается по сигналу timer_pulse внешнего интерфейса процессорного комплекса. Структура регистра CLINT_MTIMECMP представлена в таблице 9.2.

Таблица 9.2 – Структура регистра CLINT_MTIMECMP

Биты	Название	Описание	Доступ
[31:0]	MTIMECMPL	Младшая часть значения регистра сравнения	R
[63:32]	MTIMECMPH	Старшая часть значения регистра сравнения	R

Структура регистра CLINT_MTIME представлена в таблице 9.3.

Таблица 9.3 – Структура регистра CLINT_MTIME

Биты	Название	Описание	Доступ
[31:0]	MTIMEL	Младшая часть значения таймера	R
[63:32]	MTIMEH	Старшая часть значения таймера	R

9.2 Контроллер обработки внешних прерываний (PLIC)

Структурная схема контроллера обработки внешних прерываний представлена на рисунке 9.1.

Внешние источники прерываний подключаются к входному интерфейсу процессорного комплекса через сигнал int_global. Сигналы, подключаемые к int_global, должны быть синхронны с тактовым сигналом clk. Запросы прерываний int_global являются edge-sensitive сигналами.

Сигналы int_global подключаются к блоку IGW (interrupt gateway). При появлении активного уровня в бите сигнала int_global на один такт синхросигнала clk блок IGW формирует в следующем такте запрос к PLIC. Запрос в PLIC будет удерживаться до прихода от PLIC подтверждения о завершении выполнения прерывания (interrupt completion). Значение бита сигнала int_global игнорируется после выставления запроса и до подтверждения о завершении выполнения прерывания.

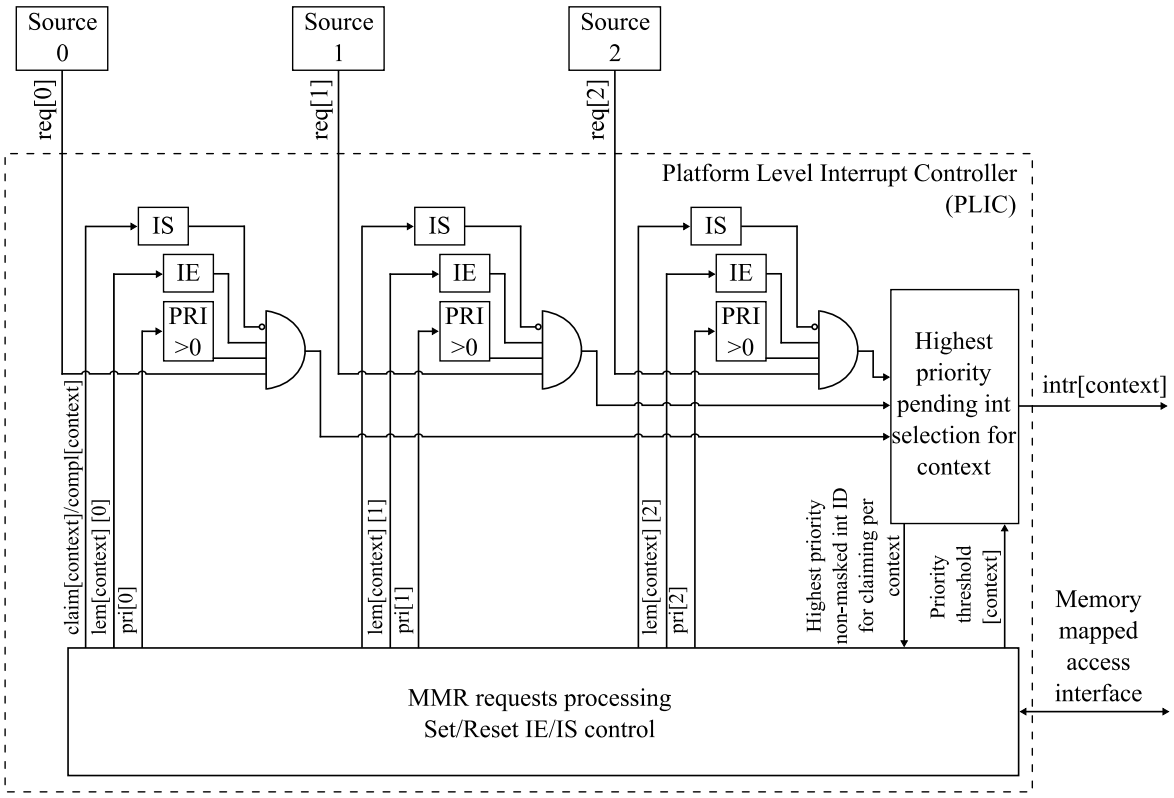


Рисунок 9.1 – Структурная схема контроллера обработки внешних прерываний

Блок PLIC получает входные запросы прерываний от IGW и формирует запросы к ядрам комплекса, которые сигнализируют о наличии или отсутствии хотя бы одного прерывания, ожидающего обработки ядром в machine режиме (сигнал int_meip) и/или user режиме (сигнал int_ueip). Сигналы int_meip и int_ueip формируются для каждого ядра отдельно.

Существуют следующие контексты обработки и управления прерываниями в PLIC:

1 Контекст machine уровня привилегированности (M-контекст) ядра.

2 Контекст user уровня привилегированности (U-контекст) ядра.

9.3 Обзор функционирования

Цикл прерывания

1 Состояние линии прерывания соответствует условию, определяемому настройкой режима MODE прерывания.

2 Устанавливается бит ожидания прерывания. Бит ожидания прерывания может быть очищен путем записи в соответствующий исходный регистр MODE или по событию «start of interrupt».

3 Устанавливается сигнал EIP для связанного приемника.

4 Запрос «Start of interrupt» обработчика прерываний вызывает переход в состояние «в обработке» («in service») ждущего прерывания с наивысшим приоритетом.

5 Когда обработчик завершает обработку прерывания, он отправляет сообщение «конец прерывания» («end of interrupt»).

Запросы прерываний

Каждый приёмник прерываний имеет EIP сигнал, который уведомляет о существовании разрешённого ждущего обработки прерывания. Сигнал EIP активируется

только в случае, если значение приоритета ждущего прерывания строго выше установленного порога.

Таблица прерываний представляет собой перечень векторов, соответствующих определенным обработчикам прерываний, см. таблицу 9.1.

Таблица 9.1 – Таблица прерываний

Номер вектора	Обозначение	Описание
1	WDT	Прерывание блока сторожевого таймера
2	CAN_IRQ0	Прерывания блока CAN, линия 0
3	CAN_IRQ1	Прерывания блока CAN, линия 1
4	USB	Прерывания блока USB
5	GPIO	Прерывания портов GPIO
6	TMR32	Прерывание 32-разрядного таймера
7	TMR0	Прерывание таймера 0
8	TMR1	Прерывание таймера 1
9	TMR2	Прерывание таймера 2
10	QSPI	Общее прерывание контроллера QSPI
11	SPI0	Общее прерывание контроллера SPI0
12	SPI1	Общее прерывание контроллера SPI1
13	DMA_CH_0_1_2	Прерывания контроллера DMA по каналам 0,1,2
14	DMA_CH_3_4_5	Прерывания контроллера DMA по каналам 3,4,5
15	DMA_CH_6_7_8	Прерывания контроллера DMA по каналам 6,7,8
16	DMA_CH_9_10_11	Прерывания контроллера DMA по каналам 9,10,11
17	DMA_CH_12_13_14	Прерывания контроллера DMA по каналам 12,13,14
18	DMA_CH_15_16_17	Прерывания контроллера DMA по каналам 15,16,17
19	DMA_CH_18_19_20	Прерывания контроллера DMA по каналам 18,19,20
20	DMA_CH_21_22_23	Прерывания контроллера DMA по каналам 21,22,23
21	I2C	Прерывание контроллера I2C
22	UART0	Общее прерывание блока UART0
23	UART1	Общее прерывание блока UART1
24	UART2	Общее прерывание блока UART2
25	UART3	Общее прерывание блока UART3
26	UART4	Общее прерывание блока UART4
27	CRYPTO_HASH_CRC	Прерывание блоков CRYPTO и HASH, CRC0, CRC1
28	TRNG	Прерывание блока TRNG
28	ADC	Прерывание блока АЦП ADCSAR и ADCSD
30	CMP	Прерывание блока CMP
31	PMU_RTC	Прерывание блока PMU_RTC

9.4 Конфигурация прерываний

Карта регистров контроллера PLIC показана в таблице 9.2

Таблица 9.2 – Карта регистров контроллера прерываний PLIC

Адрес	Обозначение	Описание
0x0C000004	PRI[1]	Регистр уровня приоритета для источника прерываний 1 (WDT)
0x0C000008	PRI[2]	Регистр уровня приоритета для источника прерываний 2 (CAN_IRQ0)
0x0C00000C	PRI[3]	Регистр уровня приоритета для источника прерываний 3 (CAN_IRQ1)
…		
0x0C00007C	PRI[31]	Регистр уровня приоритета для источника прерываний 31 (PMU_RTC)
…		
0x0C001000	IPM0	Регистр-маска 0 наличия прерываний для источников 1 – 31 (бит 0 считывается всегда 0) (только для чтения)
…		
0x0C002000	MIEM0	Регистр-маска разрешения прерываний в machine режиме для источников 1 – 31 (бит 0 считывается всегда 0)
…		
0x0C002080	UIEM0	Регистр-маска разрешения прерываний в user режиме для источников 1 – 31 (бит 0 считывается всегда 0)
…		
0x0C004000	NINT	Регистр количества прерываний (только для чтения)
0x0C004004	NPRI	Регистр количества уровней приоритетов (только для чтения)
…		
0x0C200000	MTHR	Регистр порога приоритета прерываний в machine режиме
0x0C200004	MICC	Регистр взятия/завершения обработки прерывания в machine режиме
…		
0x0C201000	UTHR	Регистр порога приоритета прерываний в user режиме
0x0C201004	UICC	Регистр взятия/завершения обработки прерывания в user режиме

Для каждого источника прерываний n существует отдельный регистр PRI[n], позволяющий задать его приоритет. Контроллер прерываний поддерживает семь уровней приоритета прерываний. Приоритеты прерываний нумеруются с первого (низкий) по седьмой (высокий). После сброса приоритет прерывания имеет нулевое значение, что означает прерывание запрещено. Поэтому перед разрешением прерывания, необходимо обязательно сконфигурировать его приоритет.

Текущее состояние битов ожидания источника прерывания в ядре PLIC может быть считано из регистра IPM0. Бит ожидания для номера прерывания N хранится в бите N. Бит 0 слова 0, который представляет собой несуществующий источник прерывания 0, всегда аппаратно обнулён.

Биты ожидания, расположенные в регистре IPM0, доступны только для чтения. Бит ожидания в ядре PLIC очищается после запуска обработки прерывания (по событию «Start of interrupt») для соответствующего источника. Модуль gateway направляет новый запрос прерывания в ядро контроллера PLIC только после получения уведомления о том, что обработка предыдущего запроса прерывания из того же источника, завершена. Если источник прерывания, чувствительный к уровню, отключает прерывание после того, как ядро PLIC установит бит IP и прежде, чем прерывание будет обслужено, IP бит остается установленным и источник прерывания будет обслуживаться обработчиком, который затем

должен будет определить, что устройство вызвавшее прерывание больше не требует обслуживания.

Регистры MIEM0 и UIEM0 представляют собой битовую маску. Единица в i-бите указывает на разрешение отправки прерывания от (i − 1) источника в аппаратный поток исполнения; ноль – на запрещение. После сброса PLIC значения всех битов в регистре MIEM0/UIEM0 сброшены в 0. По готовности принимать прерывания, ПО должно разрешить обработку желаемых прерываний путем записи 1 в соответствующие биты регистра MIEM0/UIEM0. Рекомендуется устанавливать в 1 только те биты MIEM0/UIEM0, которые соответствуют действительно существующим источникам, подключенным к ядру, чтобы избежать появления прерываний от несуществующих источников.

9.5 Программная модель обработки внешних прерываний

Источник прерываний (interrupt source) через PLIC сигнализирует в ядро о наличии прерывания. После выполнения необходимых аппаратных проверок о допустимости выполнения прерываний и выбора наиболее приоритетного прерывания (при готовности нескольких прерываний), PLIC формирует сигнал int_meip к ядру, указывающий на наличие прерывания, готового к обработке. Сигнал int_meip подключается в CSR регистр mip в бит MEIP (machine external interrupt pending, бит 11.

По умолчанию все прерывания во всех уровнях привилегированности обрабатываются в machine режиме. Если прерывания не запрещены (в регистре mie установлен бит MEIE и в регистре mstatus установлен бит MIE), управление передается в обработчик прерывания (выполняется переход на адрес обработчика прерываний), а в регистр mcause записывается «причина», т.е. тип источника, вызвавшего прерывание. Для внешних прерываний тип устанавливается в значение Machine external interrupt. Обработчик прерывания в machine режиме может перенаправить обработку прерывания в user режим путем установки бита UEIP (user external interrupt pending, бит 8) в CSR регистре uip (см. [3]) и последующего вызова команды MRET. При этом прерывания в user режиме должны быть разрешены, т.е. установлен бит UEUE (user external interrupt enable, бит 8) в CSR регистре uie (см. [3]).

Для повышения производительности программное обеспечение может использовать CSR регистры medeleg и mideleg для закрепления обработки прерываний непосредственно за user режимом. При использовании данной схемы прерывания должны быть разрешены в U-контексте ядра через соотвествующие регистры PLIC. При этом PLIC будет уведомлять ядро о наличии такого прерывания через сигнал int_ueip.

Для «взятия» внешнего прерывания на исполнение необходимо выполнить команду чтения по адресу MMR-регистра ICC. Эта транзакция сигнализирует в PLIC о том, что аппаратный поток принял прерывание к исполнению. После завершения обработки прерывания необходимо выполнить команду записи по адресу регистра ICC, что сигнализирует в PLIC о том, что обработка прерывания завершена.

Помимо использования модели обработки прерываний, описанной выше, для отслеживания наличия прерываний может быть использована модель опроса (poll model), в которой вместо «ожидания» появления прерывания и «заброса» исполнения программы в обработчик прерывания, ПО может самостоятельно проверять наличие запросов на прерывания путем чтения регистра IPM. ПО также может выполнять «взятие» прерывания на обработку путем чтения регистра ICC.

При использовании данной модели программирования надо учитывать, что чтение IPM не гарантирует, что последующее чтение ICC вернет прерывание с тем же номером. Последующее чтение ICC может вернуть прерывание с тем же номером или с другим номером, а также вернуть номер 0, указывающий на отсутствие прерывания.

10 Контроллер прямого доступа к памяти DMA

Основные свойства и отличительные особенности:

- 24 канала контроллера прямого доступа к памяти DMA;
- каждый канал DMAi (где i от 0 до 23) имеет свои сигналы управления передачей данных и программируемый уровень приоритета;
- каждый уровень приоритета обрабатывается исходя из уровня приоритета, определяемого номером канала DMAi;
- поддержка различного типа передачи данных в пределах внутреннего ОЗУ: память – память, память – периферия, периферия – память;
- поддержка различных типов циклов DMA;
- поддержка циклического режима передачи;
- поддержка передачи данных различной разрядности;
- каждому каналу DMAi доступна первичная и альтернативная структура управляющих данных канала;
- все данные канала хранятся в ОЗУ в структуре управляющих данных канала;
- разрядность данных приемника равна разрядности данных передатчика;
- количество передач в одном цикле может программироваться от 1 до 1024;
- инкремент адреса передачи может быть больше, чем разрядность данных;
- возможность начать передачи по сигналам от блоков: GPIO, TMR32, TMR, UART, SPI, ADCSAR, HASH.

Аппаратные источники запросов каналов контроллера DMA указаны в таблице 10.1.

Таблица 10.1 – Аппаратные источники запросов каналов контроллера DMA

Номер канала	Аппаратный источник запросов	Описание
0	GPIOA	Канал DMA от порта GPIOA
1	GPIOB	Канал DMA от порта GPIOB
2	GPIOC	Канал DMA от порта GPIOC
3	-	-
4	TMR32	Канал DMA от TMR32
5	TMR0	Канал DMA от TMR0
6	TMR1	Канал DMA от TMR1
7	TMR2	Канал DMA от TMR2
8	UART0_TX	Канал DMA от UART0 по передаче
9	UART1_TX	Канал DMA от UART1 по передаче
10	UART2_TX	Канал DMA от UART2 по передаче
11	UART0_RX	Канал DMA от UART0 по приему
12	UART1_RX	Канал DMA от UART1 по приему
13	UART2_RX	Канал DMA от UART2 по приему
14	ADCSAR_SEQ0	Канал DMA от секвенсора 0 блока ADCSAR
15	ADCSAR_SEQ1	Канал DMA от секвенсора 1 блока ADCSAR
16	-	-
17	SPI0_TX	Канал DMA от SPI0 по передаче
18	SPI1_TX	Канал DMA от SPI1 по передаче
19	QSPI_RX	Канал DMA от QSPI по приему
20	SPI0_RX	Канал DMA от SPI0 по приему
21	SPI1_RX	Канал DMA от SPI1 по приему
22	HASH	Канал DMA от HASH
23	-	-

10.1 Программное управление контроллером DMA

Контроллер DMA выполняет передачи 8-, 16- и 32-разрядных данных. Разрядность данных источника и приемника должны быть одинаковыми.

Контроллер DMA позволяет управлять инкрементом адреса при чтении данных из источника и при записи данных в приемник. Инкремент адреса зависит от разрядности передаваемых данных: минимальная величина инкремента адреса всегда соответствует разрядности передаваемых данных; максимальная величина – одно слово. Контроллер DMA может быть настроен на работу с фиксированным адресом (например, для работы с FIFO).

Контроллер DMA имеет возможность обслуживать сигналы запроса на одиночный обмен SREQ и сигналы запроса на пакетный обмен BREQ блоков UART, SPI. Блок ADC генерирует только запросы на пакетный обмен BREQ.

Каждому каналу контроллера DMA соответствуют две структуры управляющих данных: первичная и альтернативная. В ОЗУ должна быть отведена область для хранения этих структур.

На рисунке 10.1 показана область памяти, необходимая контроллеру для структур управляющих данных каналов при использовании всех 24 каналов и опциональной альтернативной структуры данных.

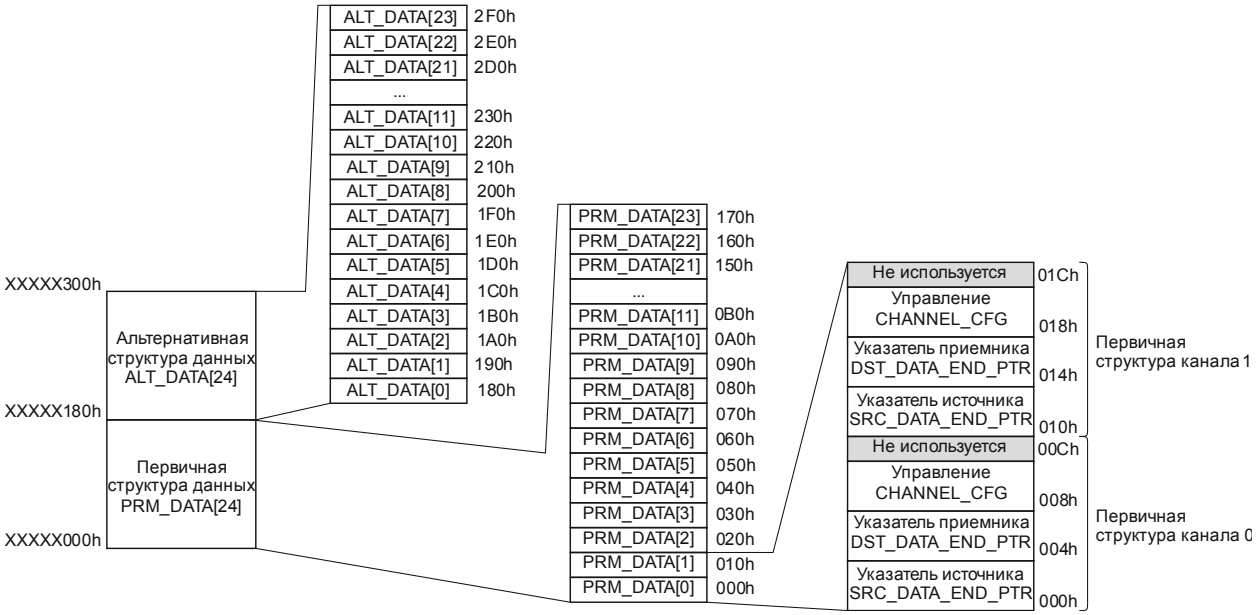


Рисунок 10.1 – Карта памяти для 24 каналов контроллера DMA, включая альтернативную структуру

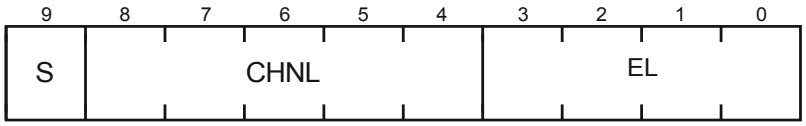
Объем структуры, показанной на рисунке 10.1, составляет 768 байт. Контроллер использует младшие разряды адреса для доступа ко всем элементам структуры управляющих данных, и поэтому разрешенные значения базового адреса для первичной структуры управляющих данных – XXXX_X000h, XXXX_X300h, XXXX_X600h и т. д.

Базовый адрес для первичной структуры управляющих данных возможно установить путем записи соответствующего значения в регистр BASEPTR.

В таблице 10.2 перечислены разряды адреса, обеспечивающие контроллеру DMA доступ к различным элементам структуры управляющих данных.

Таблица 10.2 – Разряды адреса, используемые для доступа к управляющим данным 24 каналов контроллера DMA

Разряды адреса



Обозначение	Биты	Действие	
S	9	Выбор структуры управляющих данных:	
		0	Первичная
		1	Альтернативная
CHNL	8-4	Выбор канала. Допустимые значения 0h-17h	
EL	3-0	Выбор управляющего элемента:	
		0h	Указатель конца данных источника
		4h	Указатель конца данных приемника
		8h	Конфигурация структуры управляющих данных
		Ch	Не используется. Контроллер не имеет доступа к этому адресу

Не обязательно вычислять базовый адрес альтернативной структуры управляющих данных, он вычисляется автоматически и помещается в регистр ALTBASEPTR.

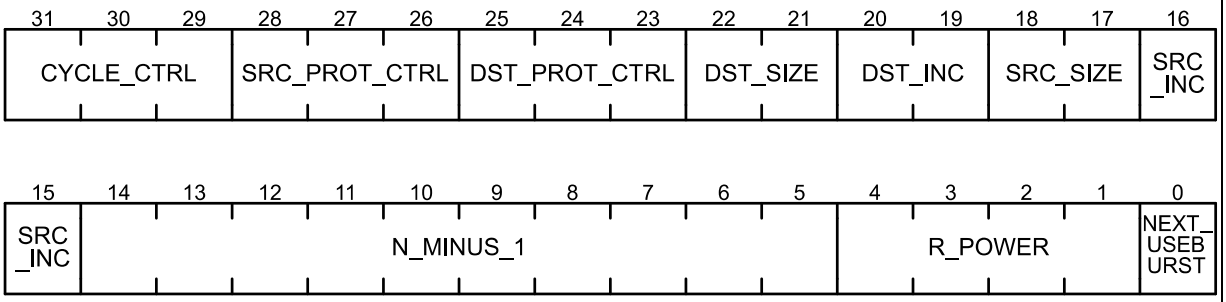
Любая из структур управляющих данных каждого канала состоит из двух указателей адреса (приемника и источника данных) и ячейки управления канала.

Управление канала CHANNEL_CFG

32-разрядная ячейка памяти, содержащая конфигурационную информацию для осуществления передач DMA (на рисунке 10.1 отмечена как «Управление …»). В начале цикла DMA или в начале 2^R передачи контроллер DMA считывает значение этой ячейки. После выполнения 2^R или N передач он сохраняет обновленное ее значение обратно в память. Структура регистра CHANNEL_CFG приведена в таблице 10.3.

Таблица 10.3 – Структура управляющих данных канала

CHANNEL_CFG



Продолжение таблицы 10.3

<table>
<tr><th>Поле</th><th>Биты</th><th colspan="4">Описание</th></tr>
<tr><td>CYCLE_
CTRL</td><td>31-29</td><td colspan="4">Поле задания типа цикла DMA</td></tr>
<tr><td></td><td></td><td>000b</td><td colspan="3">Недействительный.
Структура управляющих данных канала в запрещенном состоянии</td></tr>
<tr><td></td><td></td><td>001b</td><td colspan="3">Основной</td></tr>
<tr><td></td><td></td><td>010b</td><td colspan="3">Авто-запрос</td></tr>
<tr><td></td><td></td><td>011b</td><td colspan="3">«Пинг-понг»</td></tr>
<tr><td></td><td></td><td>100b</td><td colspan="3">Работа с памятью в режиме «разборка-сборка» с использованием первичных управляющих данных канала</td></tr>
<tr><td></td><td></td><td>101b</td><td colspan="3">Работа с памятью в режиме «разборка-сборка» с использованием альтернативных управляющих данных канала</td></tr>
<tr><td></td><td></td><td>110b</td><td colspan="3">Работа с периферией в режиме «разборка-сборка» с использованием первичных управляющих данных канала</td></tr>
<tr><td></td><td></td><td>111b</td><td colspan="3">Работа с периферией в режиме «разборка-сборка» с использованием альтернативных управляющих данных канала</td></tr>
<tr><td></td><td></td><td colspan="4">Примечание – После завершения всего цикла передач контроллер DMA устанавливает значение поля CYCLE_CTRL в 000b, переводя тем самым тип цикла в «недействительный». Это позволяет избежать повторения выполненной передачи DMA</td></tr>
<tr><td>SRC_PROT
_CTRL</td><td>28-26</td><td colspan="4">Задает параметры защиты шины AHB при чтении данных из источника</td></tr>
<tr><td></td><td></td><td rowspan="2">Код</td><td colspan="3">Биты поля SRC_PROT_CTRL</td></tr>
<tr><td></td><td></td><td>28</td><td>27</td><td>26</td></tr>
<tr><td></td><td></td><td>0</td><td>Доступ не кэшируется</td><td>Доступ не буферизуется</td><td>Доступ непривилегированный</td></tr>
<tr><td></td><td></td><td>1</td><td>Доступ кэшируется</td><td>Доступ буферизуется</td><td>Доступ привилегированный</td></tr>
<tr><td>DST_PROT
_CTRL</td><td>25-23</td><td colspan="4">Задает параметры защиты шины AHB при записи данных в приемник</td></tr>
<tr><td></td><td></td><td rowspan="2">Код</td><td colspan="3">Биты поля DST_PROT_CTRL</td></tr>
<tr><td></td><td></td><td>25</td><td>24</td><td>23</td></tr>
<tr><td></td><td></td><td>0</td><td>Доступ не кэшируется</td><td>Доступ не буферизуется</td><td>Доступ непривилегированный</td></tr>
<tr><td></td><td></td><td>1</td><td>Доступ кэшируется</td><td>Доступ буферизуется</td><td>Доступ привилегированный</td></tr>
<tr><td>DST_SIZE</td><td>22-21</td><td colspan="4">Разрядность данных приемника.
Значение этого поля должно быть равно значению поля SRC_SIZE.
Примечание – Если контроллер обнаруживает неравные значения этих полей, он при ближайшем обновлении поля N_MINUS_1 устанавливает значение поля DST_SIZE, равное SRC_SIZE.</td></tr>
<tr><td>DST_INC</td><td>20-19</td><td colspan="4">Шаг инкремента адреса приемника.
Код, записанный в поле DST_INC, задает шаг, который в свою очередь зависит от разрядности данных источника.</td></tr>
<tr><td></td><td></td><td rowspan="2">Код</td><td colspan="3">Разрядность данных источника</td></tr>
<tr><td></td><td></td><td>Байт</td><td>Слово (16 бит)</td><td>Двойное слово (32 бита)</td></tr>
<tr><td></td><td></td><td>00</td><td>Шаг – байт</td><td>Зарезервировано</td><td>Зарезервировано</td></tr>
<tr><td></td><td></td><td>01</td><td colspan="2">Шаг – слово (16 бит)</td><td>Зарезервировано</td></tr>
</table>

Продолжение таблицы 10.3

<table>
<tr><th>Поле</th><th>Биты</th><th colspan="4">Описание</th></tr>
<tr><td></td><td></td><td>10</td><td colspan="3">Шаг – двойное слово (32 бита)</td></tr>
<tr><td></td><td></td><td>11</td><td colspan="3">Нет инкремента. Адрес остается равным значению ячейки DST_DATA_END_PTR</td></tr>
<tr><td>SRC_SIZE</td><td>18-17</td><td colspan="4">Разрядность данных источника</td></tr>
<tr><td></td><td></td><td>00</td><td colspan="3">Байт</td></tr>
<tr><td></td><td></td><td>01</td><td colspan="3">Слово (16 бит)</td></tr>
<tr><td></td><td></td><td>10</td><td colspan="3">Двойное слово (32 бита)</td></tr>
<tr><td></td><td></td><td>11</td><td colspan="3">Зарезервировано. Не использовать!</td></tr>
<tr><td>SRC_INC</td><td>16-15</td><td colspan="4">Шаг инкремента адреса источника.
Код, записанный в поле SRC_INC, задает шаг, который в свою очередь зависит от разрядности данных источника.</td></tr>
<tr><td></td><td></td><td rowspan="2">Код</td><td colspan="3">Разрядность данных источника</td></tr>
<tr><td></td><td></td><td>Байт</td><td>Слово (16 бит)</td><td>Двойное слово (32 бита)</td></tr>
<tr><td></td><td></td><td>00</td><td>Шаг – байт</td><td>Зарезервировано</td><td>Зарезервировано</td></tr>
<tr><td></td><td></td><td>01</td><td colspan="2">Шаг – слово (16 бит)</td><td>Зарезервировано</td></tr>
<tr><td></td><td></td><td>10</td><td colspan="3">Шаг – двойное слово (32 бита)</td></tr>
<tr><td></td><td></td><td>11</td><td colspan="3">Нет инкремента. Адрес остается равным значению ячейки SRC_DATA_END_PTR</td></tr>
<tr><td>N_MINUS_1</td><td>14-5</td><td colspan="4">Перед выполнением цикла DMA эти разряды указывают общее количество передач DMA, из которых состоит цикл</td></tr>
<tr><td></td><td></td><td>Код</td><td colspan="3">Количество передач</td></tr>
<tr><td></td><td></td><td>000h</td><td colspan="3">1</td></tr>
<tr><td></td><td></td><td>001h</td><td colspan="3">2</td></tr>
<tr><td></td><td></td><td>…</td><td colspan="3">…</td></tr>
<tr><td></td><td></td><td>3FFh</td><td colspan="3">1024</td></tr>
<tr><td></td><td></td><td colspan="4">Примечание – Контроллер DMA обновит это поле перед тем, как произвести процесс арбитража. Это позволяет контроллеру хранить количество оставшихся передач DMA до завершения всего цикла DMA.</td></tr>
<tr><td>R_POWER</td><td>4-1</td><td colspan="4">Параметр R.
Задает количество передач канала DMA до выполнения контроллером DMA процедуры арбитража (переарбитрации).
Количество передач равно 2R</td></tr>
<tr><td></td><td></td><td>Код</td><td colspan="3">Количество передач</td></tr>
<tr><td></td><td></td><td>0h</td><td colspan="3">1 (арбитраж производится после каждой передачи DMA)</td></tr>
<tr><td></td><td></td><td>1h</td><td colspan="3">2</td></tr>
<tr><td></td><td></td><td>2h</td><td colspan="3">4</td></tr>
<tr><td></td><td></td><td>3h</td><td colspan="3">8</td></tr>
<tr><td></td><td></td><td>4h</td><td colspan="3">16</td></tr>
<tr><td></td><td></td><td>5h</td><td colspan="3">32</td></tr>
<tr><td></td><td></td><td>6h</td><td colspan="3">64</td></tr>
<tr><td></td><td></td><td>7h</td><td colspan="3">128</td></tr>
<tr><td></td><td></td><td>8h</td><td colspan="3">256</td></tr>
<tr><td></td><td></td><td>9h</td><td colspan="3">512</td></tr>
<tr><td></td><td></td><td>Ah – Fh</td><td colspan="3">1024 (арбитраж не производится, так как максимальное количество передач DMA равно 1024)</td></tr>
<tr><td></td><td></td><td colspan="4">Примечание – Необходимо с осторожностью устанавливать большие значения R для низкоприоритетных каналов, так как это может привести к невозможности обслуживать запросы по высокоприоритетным каналам.</td></tr>
</table>

Окончание таблицы 10.3

Поле	Биты	Описание	
NEXT_ USEBURST	0	Контролирует установку соответствующего каналу бита в регистре USEBURSTSET, если контроллер работает в периферийном режиме «разборка-сборка» и завершает цикл DMA, используя альтернативные управляющие данные. Примечание – Перед завершением цикла DMA, использующего альтернативные управляющие данные, контроллер DMA сбрасывает соответствующий каналу бит в регистре USEBURSTSET, если количество оставшихся передач DMA меньше, чем 2^R. Программирование бита NEXT_USEBURST определяет, будет ли контроллер DMA дополнительно переопределять состояние бита в регистре USEBURSTSET. Если контроллер выполняет цикл DMA в периферийном режиме «разборка-сборка», то после окончания цикла, использующего альтернативные управляющие данные, дальнейшие действия будут зависеть от состояния бита NEXT_USEBURST	
		0	Контроллер DMA не изменяет значение соответствующего каналу бита в регистре USEBURSTSET. Если бит CHi в USEBURSTSET сброшен, то при выполнении циклов DMA с использованием альтернативных управляющих данных контроллер DMA отвечает как на запросы BREQ, так и на запросы SREQ от периферии
		1	Контроллер DMA изменяет значение соответствующего каналу бита в регистре USEBURSTSET, а именно – устанавливает бит. Поэтому для оставшихся циклов DMA с использованием альтернативных управляющих данных контроллер DMA реагирует только на запросы BREQ от периферии

Указатель конца данных источника SRC_DATA_END_PTR и указатель конца данных приемника DST_DATA_END_PTR

Указатель конца данных источника SRC_DATA_END_PTR и указатель конца данных приемника DST_DATA_END_PTR – 32-разрядные ячейки памяти, которые содержат адрес месторасположения конца данных источника и приемника соответственно. Перед тем как контроллер DMA выполнит передачу, необходимо определить их значения. Контроллер DMA считывает значения этих областей перед началом 2^R передач.

Для вычисления адреса источника передачи контроллер DMA выполняет сдвиг влево значения N_MINUS_1 на количество разрядов, соответствующее полю SRC_INC, и затем вычитает получившееся значение от значения SRC_DATA_END_PTR.

Подобным образом вычисляется начальный адрес приемника/источника, и контроллер DMA выполняет сдвиг влево значения N_MINUS_1 на количество разрядов, соответствующее полю DST_INC, и затем вычитает получившееся значение от значения DST_DATA_END_PTR.

10.2 Правила обмена данными

Следует избегать адресации к зарезервированным или неиспользованным адресам, так как это может привести к непредсказуемым результатам.

Необходимо заполнять неиспользуемые или зарезервированные разряды регистров нулями при записи и игнорировать значения таких разрядов при считывании.

Системный сброс или сброс по установке питания сбрасывает все регистры в состояние 0000_0000h, если не указано иное.

Контроллер DMA использует правила обмена данными, см. таблицу 10.4, при соблюдении следующих условий:

- канал контроллера DMA включен (установлен соответствующий каналу бит в регистре ENSET и бит MASTEREN в регистре CFG);
- запросы канала не замаскированы (сброшен соответствующий каналу бит в регистре REQMASKSET).

Таблица 10.4 – Перечень правил, при которых передачи по каналам DMAi разрешены и запросы не маскируются (i – номер канала)

Номер правила	Описание
1	Если канал не активен (передача не идет в данный момент), то установка бита CHi в регистре SWREQ или запрос от соответствующей периферии инициирует передачу по каналу i
2	Одновременно активен может быть только один канал
3	Если запрос от периферии происходит в момент, когда канал активен, то контроллер обслужит этот запрос после завершения текущей передачи
4	Если приходит сразу несколько запросов от периферии для одного канала в момент, когда канал активен, то контроллер обслужит только первый запрос после завершения текущей передачи
5	Для циклов DMA, отличных по типу от периферийного режима «разборка-сборка», по окончании 2^R передач контроллер DMA сбрасывает бит CHi в регистре USEBURSTSET, если количество оставшихся передач меньше, чем 2^R, позволяя периферии завершить передачи, используя как SREQ запросы, так и BREQ. В периферийном режиме «разборка-сборка» контроллер сбрасывает бит CHi в регистре USEBURSTSET, только если количество оставшихся передач с использованием альтернативной структуры управляющих данных меньше, чем 2^R
6	Контроллер DMA игнорирует запрос SREQ, если бит CHi регистра WAITONREQ сброшен или установлен бит CHi регистра USEBURSTSET
7	Необходимо с осторожностью устанавливать разряды регистра USEBURSTSET. Если значение, указанное в регистре N_MINUS_1 меньше, чем значение 2^R, то контроллер DMA не очистит разряды USEBURSTSET, и поэтому одиночные запросы SREQ будут запрещены. Если программные запросы через регистр SWREQ не генерируются, и периферия не осуществляет запросов на пакетную обработку BREQ, то контроллер DMA никогда не выполнит необходимых передач
8	Для типов циклов DMA, отличных от периферийного режима «разборка-сборка», если придет запрос SREQ, то контроллер DMA выполнит одну передачу. В периферийном режиме «Исполнение с изменением конфигурации», если придет запрос SREQ, контроллер DMA выполняет 2^R передач с использованием первичной структуры управляющих данных. Затем без осуществления арбитража выполняет одну передачу, используя альтернативную структуру управляющих данных

Продолжение таблицы 10.4

Номер правила	Описание
9	Для типов циклов DMA, отличных от периферийного режима «разборка-сборка», если одновременно пришли запросы SREQ и BREQ, то приоритет предоставляется BREQ, и контроллер DMA выполняет 2^R передач (или число передач, указанное в поле N_MINUS_1). В периферийном режиме «разборка-сборка», если одновременно пришли запросы SREQ и BREQ, то приоритет также предоставляется BREQ, и контроллер DMA выполняет 2^R передач с использованием первичной структуры управляющих данных. Затем без осуществления арбитража выполняет 2^R передач (или число передач, указанное в поле N_MINUS_1), используя альтернативную структуру управляющих данных
10	В периферийном режиме «разборка-сборка», если бит NEXT_USEBURST в CHANNEL_CFG установлен, то контроллер DMA устанавливает соответствующий каналу бит в регистре USEBURSTSET после окончания цикла DMA, использующего альтернативные управляющие данные
11	Когда установлен бит CHi регистра REQMASKSET, контроллер DMA игнорирует запросы SREQ и BREQ

При отключении канала (бит CHi регистра ENSET сброшен) контроллер DMA осуществляет передачи согласно правилам, представленным в таблице 10.5.

Таблица 10.5 – Перечень правил осуществления передач для запрещенных каналов

Номер правила	Описание
1	Если приходит запрос на пакетную обработку BREQ от периферии, то происходит вызов прерывания канала контроллера DMA (если было включено). Это позволяет сигнализировать о запросе, даже если канал выключен
2	Если приходит запрос на одиночную передачу SREQ от периферии, то происходит вызов прерывания канала контроллера DMA (если было включено) при условии, что бит CHi регистра WAITONREQ установлен, а бит CHi регистра USEBURSTSET сброшен. Это позволяет сигнализировать о запросе, даже если канал выключен

10.3 Правила арбитража

Контроллер DMA имеет возможность настройки момента арбитража при передачах DMA. Эта возможность позволяет уменьшить время отклика при обслуживании каналов с высоким приоритетом.

Контроллер DMA имеет настройки, которые определяют количество передач по шине AHB до повторения арбитража (переарбитрации). Это значение задается параметром R (поле R_POWER в регистре CHANNEL_CFG структуры управляющих данных канала). Количество транзакций одного канала до переарбитрации при этом равно 2^R. Например, если R = 4, то арбитраж будет проводиться через каждые 16 передач DMA.

Необходимо с осторожностью устанавливать большие значения R для низкоприоритетных каналов, так как это может привести к невозможности обслуживать запросы по высокоприоритетным каналам.

При $N > 2^R$ (N – номер передачи) и если результат деления 2^R на N не целое число, контроллер всегда выполняет последовательность из 2^R передач до тех пор, пока не выполнится условие $N < 2^R$. Контроллер выполняет оставшиеся N передач в конце цикла DMA.

Приоритет

При проведении арбитража определяется канал для обслуживания в следующем цикле DMA. На выбор следующего канала влияют:

- номер канала;
- уровень приоритета, присвоенного каналу.

Каждому каналу может быть присвоен уровень приоритета по умолчанию или высокий. Изменение уровня приоритета осуществляется установкой соответствующего бита CHi (i – номер канала) в регистрах PRIORITYSET и PRIORITYCLR. Канал 0 имеет наивысший уровень приоритета.

Порядок каналов по уменьшению уровня приоритета представлен в таблице 10.6.

После окончания цикла DMA контроллер выбирает следующий для обслуживания канал из всех включенных каналов DMAi. Рисунок 10.2 иллюстрирует процесс выбора следующего канала для обслуживания.

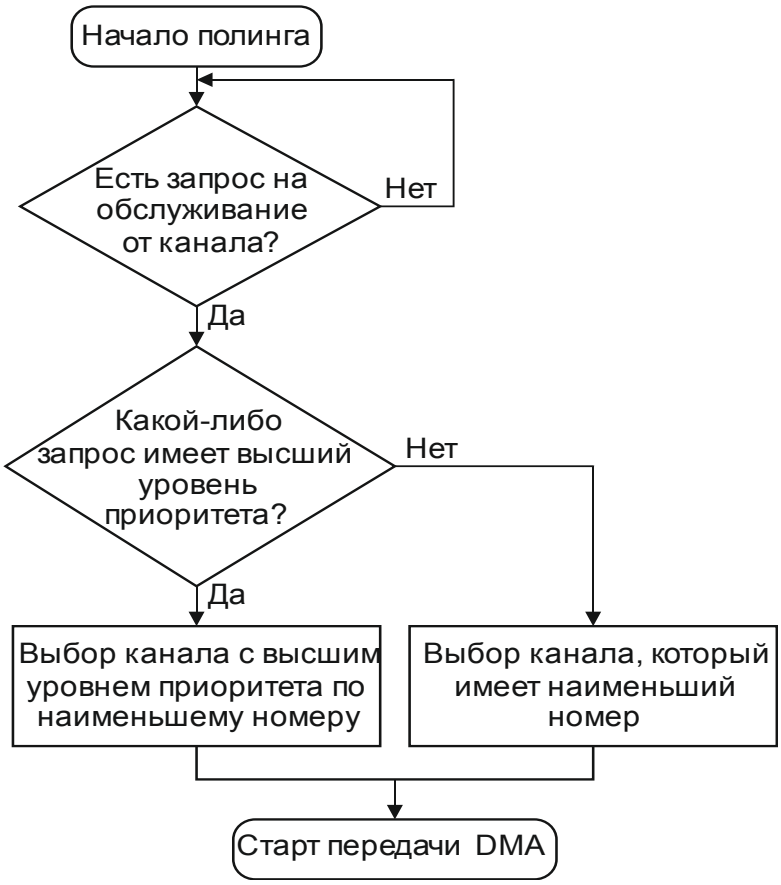


Рисунок 10.2 – Алгоритм выбора (полинга) следующего канала для обслуживания

Таблица 10.6 – Распределение приоритетов

Номер канала CHi	Состояние бита CHi в регистре PRIORITYSET	Уровень приоритета	Порядок изменения уровня приоритета
0	1	Высокий	Наивысший
1	1	Высокий	Снижение уровня приоритета
2	1	Высокий	
3	1	Высокий	
…	…	…	
22	1	Высокий	
23	1	Высокий	
0	0	По умолчанию	
1	0	По умолчанию	
2	0	По умолчанию	
3	0	По умолчанию	
…	…	…	
22	0	По умолчанию	
23	0	По умолчанию	Низший

10.4 Типы циклов

Для всех типов циклов DMA повторный арбитраж происходит после 2^R передач DMA. Если установить длинный период арбитража на низкоприоритетном канале, то это заблокирует все запросы на обработку от других каналов до тех пор, пока не будут выполнены 2^R передач DMA по данному каналу. Поэтому, устанавливая значение R, необходимо учитывать, что это может привести к повышенному времени отклика на запрос на обработку от высокоприоритетных каналов.

Поддерживаются следующие типы циклов DMA:

- недействительный (структура управляющих данных канала в запрещенном состоянии);
- основной;
- авто-запрос;
- «пинг-понг»;
- работа с памятью в режиме «разборка-сборка» (scatter-gather);
- работа с периферией в режиме «разборка-сборка».

Задание типа цикла DMA осуществляется программированием поля CYCLE_CTRL регистра CHANNEL_CFG структуры управляющих данных канала.

Недействительный цикл

После окончания цикла контроллер DMA устанавливает тип цикла в значение «недействительный» для предотвращения повтора выполненного цикла DMA.

Основной цикл

В данном режиме контроллер DMA работает либо с первичными, либо с альтернативными управляющими данными канала, совершая по 2^R передач по каждому запросу.

Перед началом работы необходимо включить контроллер DMA и разрешить работу канала: установить соответствующий каналу бит в регистре ENSET и бит MASTEREN в регистре CFG, а также проверить, что запросы канала не замаскированы (сброшен соответствующий каналу бит в регистре REQMASKSET).

После того как разрешена работа канала, цикл DMA выглядит следующим образом:

1 Контроллер DMA ожидает получения запроса (программного либо от периферии) на обработку. Если запрос получен, то контроллер DMA переходит к шагу 2.

2 Контроллер DMA выполняет 2^R передач. Если число оставшихся передач 0, контроллер DMA переходит к шагу 4, иначе выполняется шаг 3.

3 Происходит осуществление арбитража: если высокоприоритетный канал выдает запрос на обработку, то контроллер начинает обслуживание этого канала, иначе происходит ожидание очередного запроса на обработку по каналу, и если периферийный блок или программа его выдает, то контроллер DMA переходит к шагу 2.

4 Контроллер DMA указывает центральному процессору на завершение цикла DMA. Вызывается соответствующее каналу прерывание (если было включено).

Авто-запрос

Контроллеру DMA необходим лишь одиночный запрос для разрешения работы и выполнения цикла DMA. Такая работа позволяет выполнять передачу больших пакетов данных без существенного увеличения времени отклика на обслуживание высокоприоритетных запросов и не требует множественных запросов на обработку от программы или периферийных блоков. Контроллер DMA позволяет выбрать для использования либо первичную, либо альтернативную структуру управляющих данных канала.

Перед началом работы необходимо включить контроллер DMA и разрешить работу канала: установить соответствующий каналу бит в регистре ENSET и бит MASTEREN в регистре CFG, а также проверить, что запросы канала не замаскированы (сброшен соответствующий каналу бит в регистре REQMASKSET).

После того как разрешена работа канала, цикл DMA выглядит следующим образом:

1 Контроллер DMA ожидает получения запроса (программного либо от периферии) на обработку. Если запрос получен, то контроллер DMA переходит к шагу 2.

2 Контроллер выполняет 2^R передач. Если число оставшихся передач 0, контроллер DMA переходит к шагу 4, иначе выполняется шаг 3.

3 Осуществление арбитража: если высокоприоритетный канал выдает запрос на обработку, то контроллер DMA начинает обслуживание этого канала, иначе контроллер переходит к шагу 2.

4 Контроллер DMA указывает центральному процессору на завершение цикла DMA. Вызывается соответствующее каналу прерывание (если было включено).

Отличие от режима «основной» состоит в том, что в режиме «авто-запрос» контроллер DMA позволит осуществить все N транзакций по одному запросу, в то время как в основном режиме по каждому запросу будет выполняться лишь 2^R передач.

Режим «пинг-понг»

Контроллер DMA выполняет цикл DMA, используя одну из первичных структур управляющих данных, а затем выполняет еще один цикл DMA, используя альтернативную структуру управляющих данных. Контроллер выполняет циклы DMA с переключением структур до тех пор, пока не считает режим «недействительный» или «основной», или пока процессор не запретит работу канала.

На рисунке 10.3 показан пример функционирования контроллера DMA в режиме «пинг-понг». Пояснения к рисунку 10.3 представлены в виде таблицы 10.7.

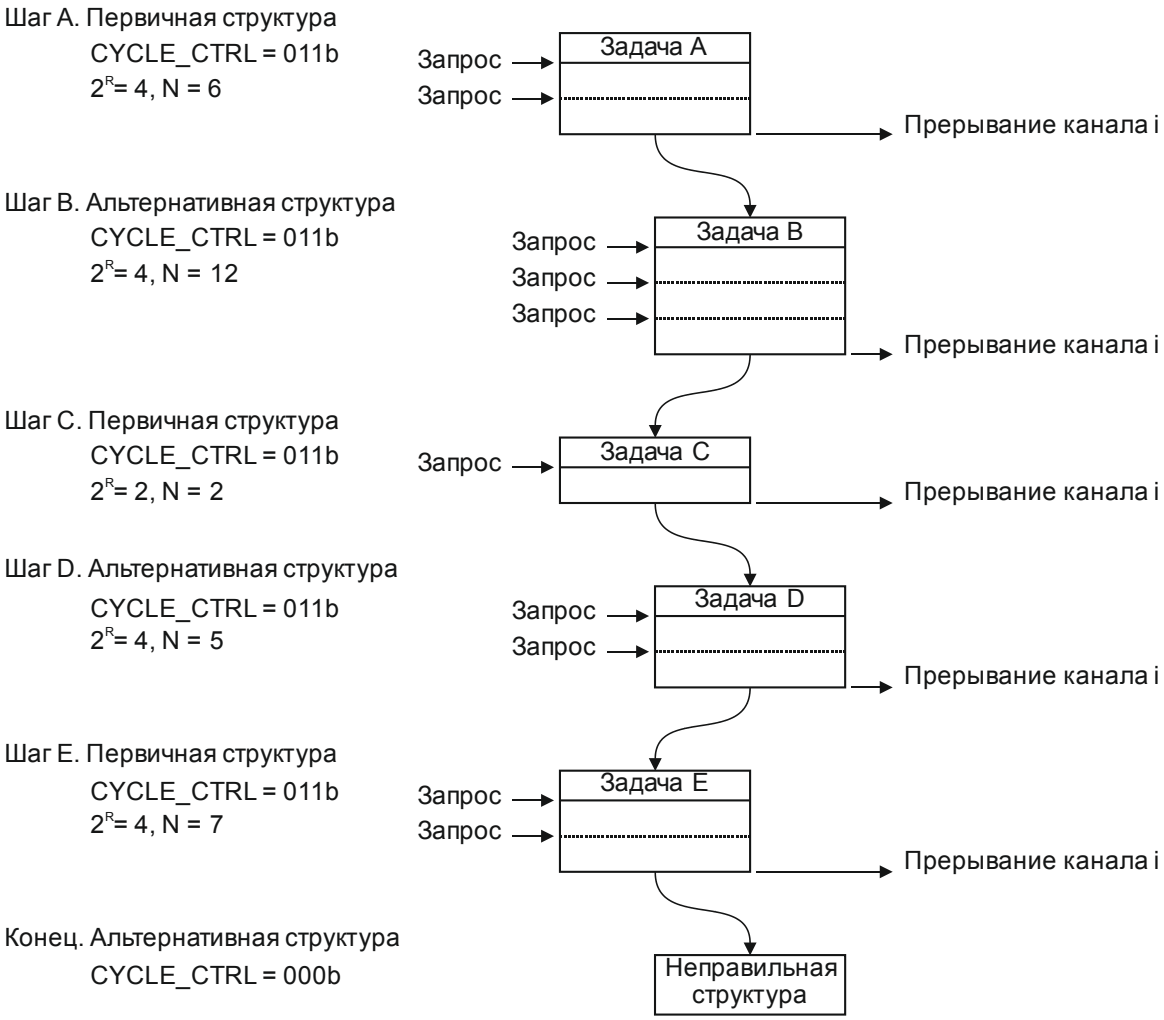


Рисунок 10.3 – Пример функционирования контроллера DMA в режиме «пинг-понг»

Таблица 10.7 – Пояснения к схеме на рисунке 10.3

Шаг	Действия процессора и контроллера DMA
A	Процессор включает контроллер DMA и разрешает работу канала. В программе устанавливаются первичная структура управляющих данных для шага A и альтернативная структура управляющих данных для шага B. Это позволит контроллеру DMA переключиться к шагу B незамедлительно после выполнения шага A, при условии, что контроллер DMA не получит запрос на обработку от высокоприоритетного канала. Контроллер DMA получает запрос и выполняет четыре передачи DMA. Контроллер DMA выполняет арбитраж. После получения запроса на обработку от этого же канала, контроллер DMA продолжает цикл в ситуации отсутствия высокоприоритетных запросов. Контроллер DMA выполняет оставшиеся две передачи DMA. Контроллер DMA входит в процедуру арбитража. Если прерывание включено, то произойдет его вызов. Примечание – После выполнения шага A процессор может установить первичные управляющие данные канала для шага C. Это позволит контроллеру переключиться к шагу C незамедлительно после выполнения шага B, при условии, что контроллер DMA не получит запрос на обработку от высокоприоритетного канала. После получения нового запроса на обработку от канала при условии его наивысшего приоритета исполняется шаг B.

Продолжение таблицы 10.7

Шаг	Действия процессора и контроллера DMA
B	Контроллер DMA выполняет четыре передачи. Контроллер DMA выполняет арбитраж. После получения запроса на обработку от этого же канала контроллер DMA продолжает цикл в ситуации отсутствия высокоприоритетных запросов. Контроллер DMA выполняет четыре передачи. Контроллер DMA выполняет арбитраж. После получения запроса на обработку от этого же канала контроллер DMA продолжает цикл в ситуации отсутствия высокоприоритетных запросов. Контроллер DMA выполняет оставшиеся четыре передачи. Контроллер DMA входит в процедуру арбитража. Если прерывание включено, то произойдет его вызов. Примечание – После выполнения шага B процессор может установить альтернативные управляющие данные канала для шага D. После получения нового запроса на обработку от канала при условии его наивысшего приоритета исполняется шаг C.
C	Контроллер DMA выполняет две передачи. Контроллер DMA входит в процедуру арбитража. Если прерывание включено, то произойдет его вызов. Примечание – После выполнения шага C процессор может установить первичные управляющие данные канала для шага E. После получения нового запроса на обработку от канала при условии его наивысшего приоритета исполняется шаг D.
D	Контроллер DMA выполняет четыре передачи. Контроллер DMA выполняет арбитраж. После получения запроса на обработку от этого же канала контроллер DMA продолжает цикл в ситуации отсутствия высокоприоритетных запросов. Контроллер DMA выполняет оставшуюся передачу. Контроллер DMA входит в процедуру арбитража. Если прерывание включено, то произойдет его вызов. Примечание – После получения нового запроса на обработку от канала при условии его наивысшего приоритета исполняется шаг E.
E	Контроллер DMA выполняет четыре передачи. Контроллер DMA выполняет арбитраж. После получения запроса на обработку от этого же канала контроллер DMA продолжает цикл в ситуации отсутствия высокоприоритетных запросов. Контроллер DMA выполняет оставшиеся три передачи. Контроллер DMA входит в процедуру арбитража. Если прерывание включено, то произойдет его вызов

Если контроллер DMA получит новый запрос на обработку от данного канала, и этот запрос будет самым приоритетным, то контроллер предпримет попытку выполнения следующего шага. Однако из-за того, что процессор не установил альтернативные управляющие данные и по окончании шага D контроллер DMA установил поле CYCLE_CTRL альтернативной управляющей структуры в состояние 000b, передачи DMA прекращаются.

Работа с памятью в режиме «разборка-сборка»

Алгоритм работы данного режима является оптимальным именно для работы с памятью, несмотря на это, его использование возможно для любого типа передачи данных: память – память, периферия – память, память – периферия, – с помощью как программных запросов, так и запросов от периферии.

В данном режиме контроллер DMA использует первичные управляющие данные для программирования альтернативных управляющих данных.

Контроллер DMA, получая начальный запрос на обработку, выполняет четыре передачи DMA, заполняя альтернативную структуру канала данными, доступными для первичной управляющей структуры. По окончании этих передач контроллер DMA входит в процедуру арбитража, и если более высокоприоритетных запросов не обнаружено, начинает цикл DMA, используя обновленные альтернативные управляющие данные, после – арбитраж, затем контроллер DMA выполняет еще четыре передачи DMA, вновь заполняя альтернативную структуру данными с помощью первичной структуры.

Контроллер DMA продолжает выполнять циклы DMA, меняя структуры управляющих данных, пока не произойдет одно из следующих условий:

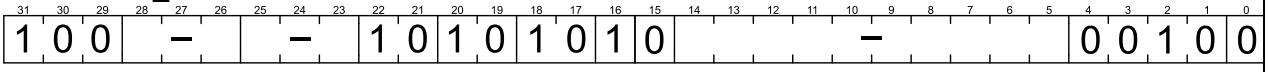
- передача с использованием альтернативной управляющей структуры будет выполнена в режиме цикла «основной»;
- контроллер считает «неправильную» структуру управляющих данных. После исполнения контроллером N передач с использованием первичных управляющих данных, он делает эти управляющие данные «неправильными» путем записи в поле CYCLE_CTRL значения 000b.

Контроллер DMA устанавливает прерывание канала DMAi в этом режиме работы только тогда, когда последний цикл передач DMA выполняется с использованием режима «основной». Также необходимо помнить, что для режима «основной» авто-запросы не действуют.

В таблице 10.8 указаны константы, которые должны быть записаны пользователем в регистр CHANNEL_C «разборка-сборка».

Таблица 10.8 – Конфигурация первичной структуры управляющих данных канала для работы с памятью в режиме «разборка-сборка»

CHANNEL_CFG



Поле	Биты	Конс-танта	Пояснение
CYCLE_CTRL	31-29	100b	Контроллер DMA работает с памятью в режиме «разборка-сборка» с использованием первичных управляющих данных канала.
SRC_PROT_CTRL	28-26	–	Управление защитой шины при чтении данных из источника. Задается пользователем
DST_PROT_CTRL	25-23	–	Управление защитой шины при записи данных в приемник. Задается пользователем
DST_SIZE	22-21	10b	Контроллер DMA осуществляет передачу двойным словом
DST_INC	20-19	10b	Контроллер DMA производит инкремент адреса с шагом в двойное слово
SRC_SIZE	18-17	10b	Контроллер DMA осуществляет передачу двойными словами
SRC_INC	16-15	10b	Контроллер DMA производит инкремент адреса с шагом в двойное слово
N_MINUS_1	14-5	–	Настраивает контроллер на выполнение N передач DMA. Так как поле R_POWER задает значение 2, то необходимо задавать значение N, кратное 4. Число, равное N/4, это количество раз, которое нужно настраивать альтернативные управляющие данные. Задается пользователем

Продолжение таблицы 10.8

Поле	Биты	Конс-танта	Пояснение
R_POWER	4-1	0010b	Контроллер DMA выполняет четыре передачи ($2^R = 2^2 = 4$)
NEXT_USEBURST	0	0	Для данного режима бит должен быть сброшен

В указатель конца данных источника SRC_DATA_END_PTR первичной структуры необходимо записать адрес конца области памяти, в которой последовательно расположено нужное количество наборов управляющих данных для программирования альтернативной структуры канала.

В указатель конца данных приемника DST_DATA_END_PTR первичной структуры необходимо записать адрес конца альтернативной управляющей структуры используемого канала.

На рисунке 10.4 показан пример функционирования контроллера DMA в режиме «разборка-сборка». Пояснения к рисунку 10.4 приведены ниже («Инициализация» и «Функционирование»).

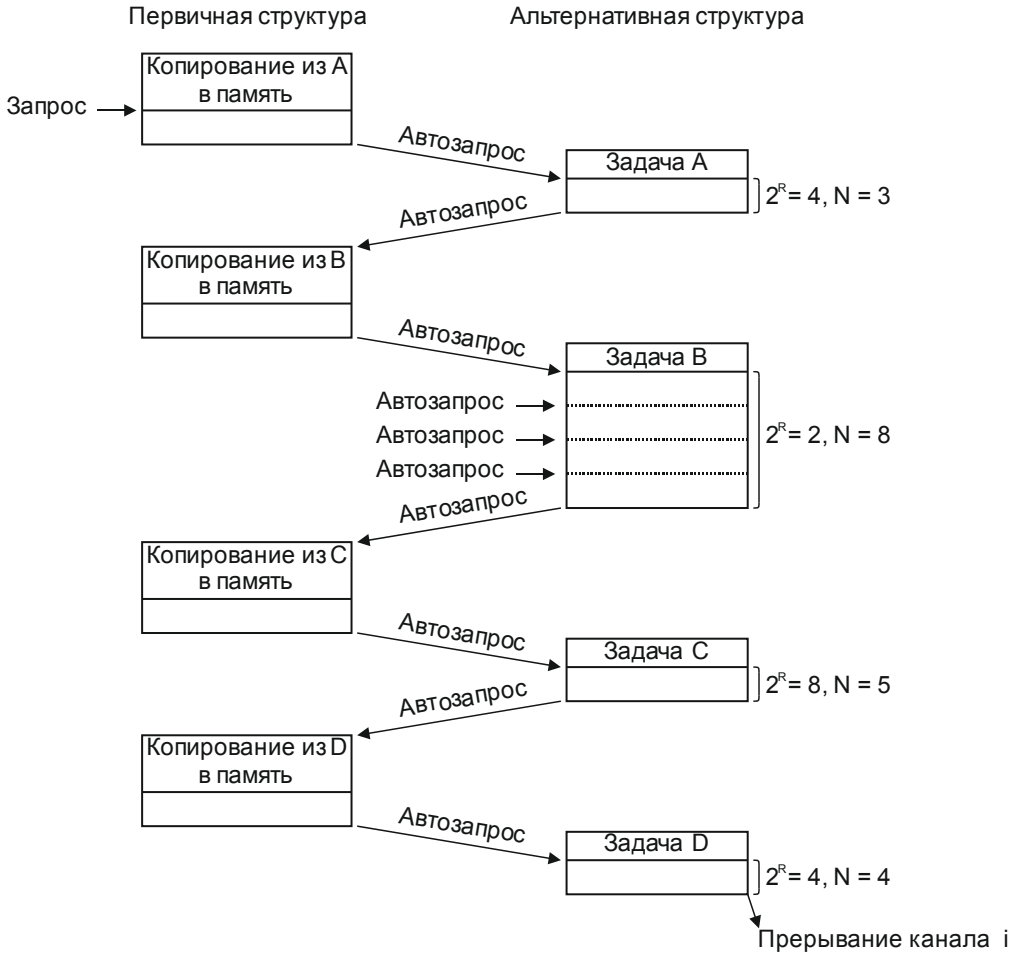


Рисунок 10.4 – Пример функционирования контроллера DMA в режиме «разборка-сборка»

Инициализация

1 Первичная структура управляющих данных настраивается для работы с памятью в режиме «разборка-сборка» путем записи в CYCLE_CTRL значения 100b. Так как

управляющие данные канала состоят из четырех слов, R_POWER = 0010b. Поскольку количество задач равно четырем, то N = 16, т. е. значение поля N_MINUS_1 = 00Fh.

2 Управляющие данные для шагов A, B, C, D располагаются в области ОЗУ. Адрес конца этой области заносится в регистр SRC_DATA_END_PTR первичных управляющих данных. Пример размещения и заполнения управляющих данных для альтернативной структуры показан в таблице 10.9. Согласно примеру, в регистр SRC_DATA_END_PTR первичной управляющей структуры необходимо записать значение 4000_015Ch.

В регистр DST_DATA_END_PTR первичной структуры необходимо записать адрес конца альтернативной структуры управляющих данных используемого канала. Например, при использовании канала 9 в регистр DST_DATA_END_PTR необходимо записать значение XXXX_X29Ch.

3 Включается контроллер DMA и разрешается работа канала, путем установки соответствующего каналу бита в регистре ENSET и бита MASTEREN в регистре CFG. Также необходимо удостовериться, что запросы канала не замаскированы (сброшен соответствующий каналу бит в регистре REQMASKSET).

Таблица 10.9 – Пример размещения управляющих данных для альтернативной структуры

Тип данных	Адрес ОЗУ	Регистр	Значение
Управляющие данные для задачи D	4000_015Ch	Не используется	XXXX_XXXXh
	4000_0158h	CHANNEL_CFG	CYCLE_CTRL = 001b, 2^R = 4, N = 4
	4000_0154h	DST_DATA_END_PTR	4000_DE00h
	4000_0150h	SRC_DATA_END_PTR	4000_D000h
Управляющие данные для задачи C	4000_016Ch	Не используется	XXXX_XXXXh
	4000_0168h	CHANNEL_CFG	CYCLE_CTRL = 101b, 2^R = 8, N = 5
	4000_0164h	DST_DATA_END_PTR	4000_CE00h
	4000_0160h	SRC_DATA_END_PTR	4000_C000h
Управляющие данные для задачи B	4000_017Ch	Не используется	XXXX_XXXXh
	4000_0178h	CHANNEL_CFG	CYCLE_CTRL = 101b, 2^R = 2, N = 8
	4000_0174h	DST_DATA_END_PTR	4000_BE00h
	40000170h	SRC_DATA_END_PTR	4000_B000h
Управляющие данные для задачи A	4000_018Ch	Не используется	XXXX_XXXXh
	4000_0188h	CHANNEL_CFG	CYCLE_CTRL = 101b, 2^R = 4, N = 3
	4000_0184h	DST_DATA_END_PTR	4000_AE00h
	4000_0180h	SRC_DATA_END_PTR	4000_A000h

Функционирование

1 Первичная структура, копирование данных задачи A. По получении первого запроса на обслуживание контроллер DMA выполняет четыре передачи DMA. Эти передачи записывают альтернативную структуру управляющих данных для задачи A. Контроллер DMA генерирует авто-запрос для канала, после чего проводит процедуру арбитража.

Далее контроллер DMA выполняет задачу A с использованием альтернативных данных, по окончании генерирует авто-запрос для канала и проводит процедуру арбитража.

2 Первичная структура, копирование данных задачи B. По получении авто-запроса контроллер DMA выполняет четыре передачи DMA. Эти передачи записывают альтернативную структуру управляющих данных для задачи B. Контроллер DMA генерирует авто-запрос для канала, после чего проводит процедуру арбитража.

Далее контроллер DMA выполняет задачу B с использованием альтернативных данных, по окончании генерирует авто-запрос для канала и проводит процедуру арбитража.

3 Первичная структура, копирование данных задачи C. По получении авто-запроса на обслуживание контроллер DMA выполняет четыре передачи DMA. Эти передачи записывают альтернативную структуру управляющих данных для задачи C. Контроллер DMA генерирует авто-запрос для канала, после чего проводит процедуру арбитража.

Далее контроллер DMA выполняет задачу C с использованием альтернативных данных, по окончании генерирует авто-запрос для канала и проводит процедуру арбитража.

4 Первичная структура, копирование данных задачи D. По получении авто-запроса на обслуживание контроллер DMA выполняет четыре передачи DMA. Эти передачи записывают альтернативную структуру управляющих данных для задачи D. Контроллер DMA записывает в CYCLE_CTRL первичных данных значение 000b для индикации о том, что эта структура управляющих данных является «неправильной». Далее контроллер DMA генерирует авто-запрос для канала, после чего проводит процедуру арбитража.

Далее контроллер DMA выполняет задачу D, используя тип цикла «основной». По завершении задачи генерирует прерывание канала DMA (если было включено) и входит в процедуру арбитража. Цикл работы с памятью в режиме «разборка-сборка» завершен.

Работа с периферией в режиме «разборка-сборка»

Алгоритм работы данного режима является оптимальным именно для работы с периферией, несмотря на это, его использование возможно для любого типа передачи данных: память – память, периферия – память, память – периферия, – с помощью как программных запросов, так и запросов от периферии.

В данном режиме контроллер DMA использует первичные управляющие данные для программирования альтернативных управляющих данных.

Контроллер DMA, получая начальный запрос на обработку, выполняет четыре передачи DMA, заполняя альтернативную структуру канала данными, доступными для первичной управляющей структуры. По окончании этих передач контроллер DMA без осуществления арбитража начинает цикл DMA, используя обновленные альтернативные управляющие данные, после – арбитраж, затем контроллер DMA выполняет еще четыре передачи DMA, вновь заполняя альтернативную структуру данными с помощью первичной структуры. Это единственный случай, при котором контроллер DMA не осуществляет процедуру арбитража после выполнения передачи DMA, используя первичные управляющие данные.

Контроллер DMA продолжает выполнять циклы DMA, меняя структуры управляющих данных, пока не произойдет одно из следующих условий:

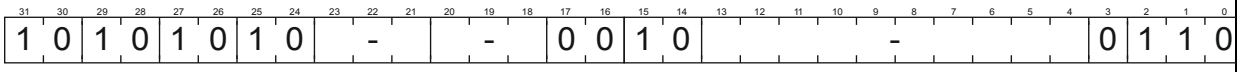
- передача с использованием альтернативной управляющей структуры будет выполнена в режиме цикла «основной»;
- контроллер DMA считает «неправильную» структуру управляющих данных. После исполнения DMA контроллером N передач с использованием первичных управляющих данных, он делает эти управляющие данные «неправильными» путем записи в поле CYCLE_CTRL значения 000b.

Контроллер DMA устанавливает прерывание канала DMAi в этом режиме работы только тогда, когда последний цикл передач DMA выполняется с использованием режима «основной». Также необходимо помнить, что для режима «основной» авто-запросы не действуют.

В таблице 10.10 указаны константы, которые должны быть записаны пользователем в регистр CHANNEL_CFG первичной структуры управляющих данных канала для работы с периферией в режиме «разборка-сборка».

Таблица 10.10 – Конфигурация первичной структуры управляющих данных канала DMAi для работы с периферией в режиме «разборка-сборка»

CHANNEL_CFG



Поле	Биты	Конс-танта	Пояснение
CYCLE_CTRL	31-29	110b	Контроллер работает с периферией в режиме «разборка-сборка» с использованием первичных управляющих данных канала DMAi
SRC_PROT_CTRL	28-26	–	Управление защитой шины при чтении данных из источника. Задается пользователем
DST_PROT_CTRL	25-23	–	Управление защитой шины при записи данных в приемник. Задается пользователем
DST_SIZE	22-21	10b	Контроллер DMA осуществляет передачу двойным словом
DST_INC	20-19	10b	Контроллер DMA производит инкремент адреса с шагом в двойное слово
SRC_SIZE	18-17	10b	Контроллер DMA осуществляет передачу двойными словами
SRC_INC	16-15	10b	Контроллер DMA производит инкремент адреса с шагом в двойное слово
N_MINUS_1	14-5	–	Настраивает контроллер на выполнение N передач DMA. Так как поле R_POWER задает значение 2, то необходимо задавать значение N, кратное 4. Число, равное N/4, это количество раз, которое нужно настраивать альтернативные управляющие данные. Задается пользователем
R_POWER	4-1	0010b	Контроллер DMA выполняет четыре передачи ($2^R = 2^2 = 4$)
NEXT_USEBURST	0	0	Для данного режима бит должен быть сброшен

В указатель конца данных источника SRC_DATA_END_PTR первичной структуры необходимо записать адрес конца области памяти, в которой последовательно расположено нужное количество наборов управляющих данных для программирования альтернативной структуры канала DMAi.

В указатель конца данных приемника DST_DATA_END_PTR первичной структуры необходимо записать адрес конца альтернативной управляющей структуры используемого канала DMAi.

На рисунке 10.5 показан пример функционирования контроллера DMA в режиме «разборка-сборка». Пояснения к рисунку 10.5 приведены ниже («Инициализация» и «Функционирование»).

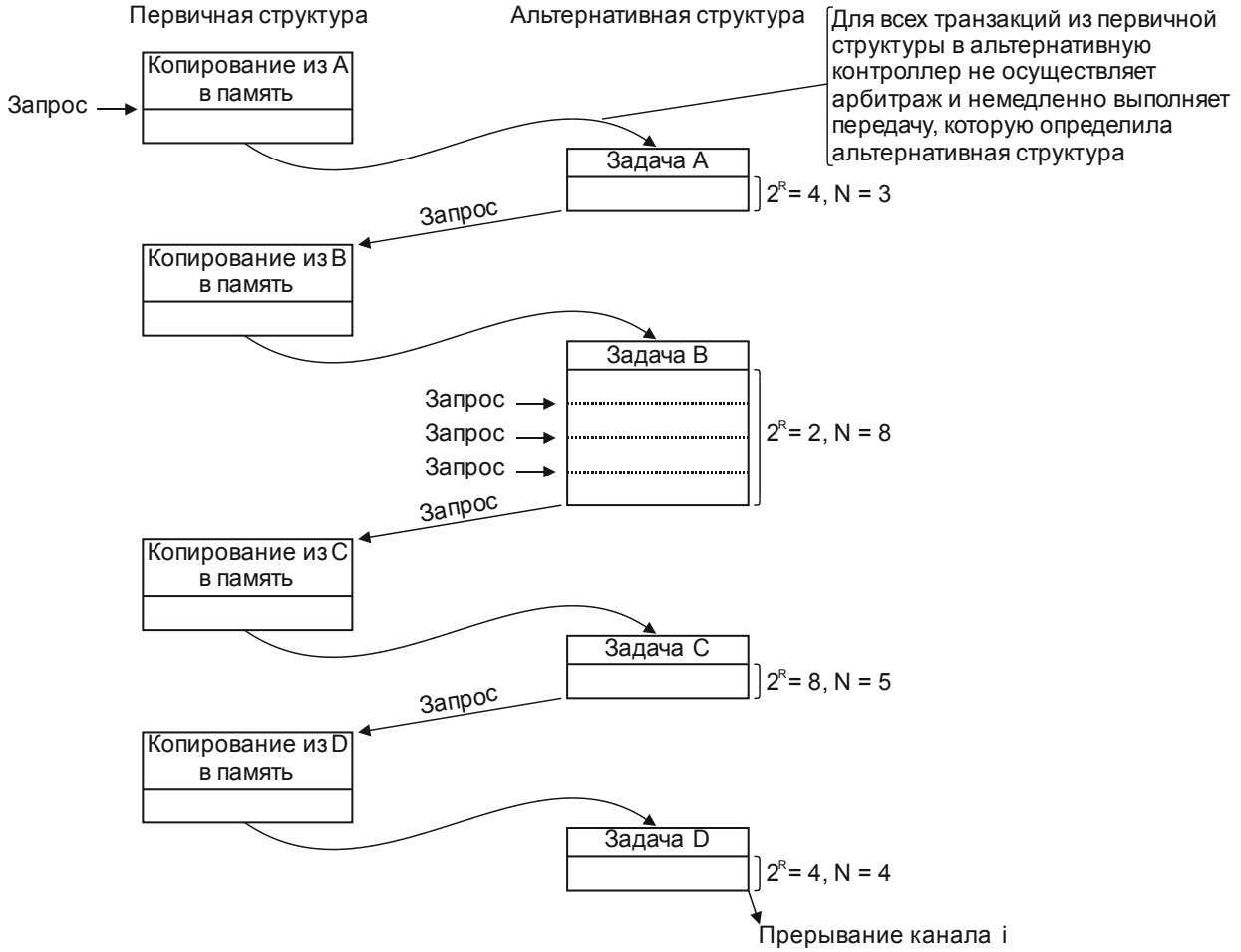


Рисунок 10.5 – Пример функционирования контроллера DMA в режиме работы с периферией «разборка-сборка»

Инициализация

1 Первичная структура управляющих данных настраивается для работы с периферией в режиме «разборка-сборка» путем записи в CYCLE_CTRL значения 110b. Поскольку управляющие данные канала состоят из четырех слов, R_POWER = 0010b. Так как количество задач равно четырем, то N = 16, т. е. значение поля N_MINUS_1 = 00Fh.

2 Управляющие данные для шагов A, B, C, D располагаются в области ОЗУ. Адрес конца этой области заносится в регистр SRC_DATA_END_PTR первичных управляющих данных. Пример размещения и заполнения управляющих данных для альтернативной структуры, показан в таблице 10.11. Исходя из примера, в регистр SRC_DATA_END_PTR необходимо занести значение 4000_015Ch.

В регистр DST_DATA_END_PTR первичной структуры необходимо занести адрес конца альтернативной структуры управляющих данных используемого канала. Например, при использовании канала 9 в регистр DST_DATA_END_PTR необходимо занести значение XXXX_X29Ch.

3 Включается контроллер DMA и разрешается работа канала, путем установки соответствующего каналу бита в регистре ENSET и бита MASTEREN в регистре CFG.

Также необходимо удостовериться, что запросы канала не замаскированы (сброшен соответствующий каналу бит в регистре REQMASKSET).

Таблица 10.11 – Пример размещения управляющих данных для альтернативной структуры

Тип данных	Адрес ОЗУ	Регистр	Значение
Управляющие данные для задачи D	4000_015Ch	Не используется	XXXX_XXXXh
	4000_0158h	CHANNEL_CFG	CYCLE_CTRL = 001b, 2^R = 4, N = 4
	4000_0154h	DST_DATA_END_PTR	4000_DE00h
	4000_0150h	SRC_DATA_END_PTR	4000_D000h
Управляющие данные для задачи C	4000_016Ch	Не используется	XXXX_XXXXh
	4000_0168h	CHANNEL_CFG	CYCLE_CTRL = 111b, 2^R = 8, N = 5
	4000_0164h	DST_DATA_END_PTR	4000_CE00h
	4000_0160h	SRC_DATA_END_PTR	4000_C000h
Управляющие данные для задачи B	4000_017Ch	Не используется	XXXX_XXXXh
	4000_0178h	CHANNEL_CFG	CYCLE_CTRL = 111b, 2^R = 2, N = 8
	4000_0174h	DST_DATA_END_PTR	4000_BE00h
	4000_0170h	SRC_DATA_END_PTR	4000_B000h
Управляющие данные для задачи A	4000_018Ch	Не используется	XXXX_XXXXh
	4000_0188h	CHANNEL_CFG	CYCLE_CTRL = 111b, 2^R = 4, N = 3
	4000_0184h	DST_DATA_END_PTR	4000_AE00h
	4000_0180h	SRC_DATA_END_PTR	4000_A000h

Функционирование

1 Первичная структура, копирование данных для задачи A. По получении запроса на обслуживание контроллер выполняет четыре передачи DMA. Эти передачи записывают альтернативную структуру управляющих данных для задачи A.

Далее контроллер сразу же выполняет задачу A и по окончании проводит процедуру арбитража. После выставления нового запроса на обслуживание, при условии, что этот запрос является наиболее приоритетным, процесс продолжается.

2 Первичная структура, копирование данных для задачи B. Контроллер выполняет четыре передачи DMA. Эти передачи записывают альтернативную структуру управляющих данных для задачи B.

Далее контроллер выполняет задачу B. Для завершения задачи необходимо три запроса (программных или от периферии). По окончании контроллер проводит процедуру арбитража. После выставления нового запроса на обслуживание, при условии, что этот запрос является наиболее приоритетным, процесс продолжается.

3 Первичная структура, копирование данных для задачи C. Контроллер выполняет четыре передачи DMA. Эти передачи записывают альтернативную структуру управляющих данных для задачи C.

Далее контроллер выполняет задачу C и по окончании проводит процедуру арбитража.

После выставления периферией нового запроса на обслуживание, при условии, что этот запрос является наиболее приоритетным, процесс продолжается.

4 Первичная структура, копирование данных для задачи D. Контроллер выполняет четыре передачи DMA. Эти передачи записывают альтернативную структуру управляющих данных для шага D. Контроллер записывает в CYCLE_CTRL первичных данных значение 000b для индикации о том, что эта структура управляющих данных является «неправильной». Далее контроллер выполняет задачу D, используя основной цикл DMA, входит в прерывание канала DMAi (если включено) и запускает процедуру арбитража. Цикл работы с периферией в режиме «разборка-сборка» завершен.

10.5 Циклический режим

По умолчанию контроллер DMA выполняет циклы однократно, т. е. чтобы выполнить исполненный цикл еще раз необходимо обновить поля N_MINUS_1 и CYCLE_CTRL, а также включить соответствующий канал путём записи в регистр ENSET.

Однако существует возможность активировать циклический режим для основного цикла или авто-запроса. В этом режиме поле CYCLE_CTRL и бит разрешения работы канала в регистре ENSET не будут сбрасываться по окончании цикла. Также значение N_MINUS_1 будет сохранено перед первой передачей и затем восстановлено, когда текущий цикл будет завершен. Таким образом, можно осуществлять циклические передачи, например, для генерации сигналов сложной формы и других задач, не задействуя при этом прерывания DMA.

Активация режима производится установкой соответсующего бита в регистре CIRCULARSET. Сброс режима – записью единицы в CIRCULARCLR.

10.6 Индикация ошибок

Контроллер DMA может отключить i-канал в следующих случаях:

- при завершении цикла DMA;
- при чтении режима канала «недействительный»;
- при появлении ошибки на шине AHB.

Как только контроллер DMA получает сообщение об ошибке по шине AHB, он отключает канал, в котором обнаружена ошибка и устанавливает флаг VAL в регистре ERRCLR. Для того чтобы определить канал контроллера DMA, в котором произошла ошибка, программа, выполняемая процессором, должна всегда хранить данные о каналах, которые недавно вызывали прерывания, т. е. завершали работу и отключались.

Алгоритм определения канала контроллера DMA с ошибкой:

- необходимо прочитать регистр ENSET с целью создания текущего списка отключенных каналов контроллера DMA;
- процессор должен сравнить список выключенных каналов контроллера DMA, полученный в результате чтения регистра ENSET, с данными о каналах, которые недавно вызывали прерывания. Канал контроллера DMA, который отключился и по которому отсутствуют данные о вызове прерывания, является каналом, связанным с ошибкой.

В контроллере DMA присутствует возможность использования режимов защиты шины AHB: при записи в приемник, при чтении из источника и при обращении к структурам управляющих данных каналов. Защита шины в каждой из ситуаций настраивается индивидуально. Доступными режимами защиты являются: кэширование, буферизация, привилегированный доступ.

Защита шины при записи в приемник настраивается полем DST_PROT_CTRL в ячейке CHANNEL_CFG структуры управляющих данных канала контроллера DMA.

Защита шины при чтении из источника настраивается полем SRC_PROT_CTRL в ячейке CHANNEL_CFG структуры управляющих данных канала контроллера DMA.

Защита шины при обращении контроллера DMA к структурам управляющих данных каналов DMAi настраивается полем CHPROT в регистре CFG.

10.7 Прерывания

В контроллер PLIC поступает только 8 линий прерывания от каналов DMA, объединенных по 3 канала:

- прерывания от каналов DMA с 0 по 2 поступают на линию прерывания №13;
- прерывания от каналов DMA с 3 по 5 поступают на линию прерывания №14;
- прерывания от каналов DMA с 6 по 8 поступают на линию прерывания №15;
- прерывания от каналов DMA с 9 по 11 поступают на линию прерывания №16;
- прерывания от каналов DMA с 12 по 14 поступают на линию прерывания №17;
- прерывания от каналов DMA с 15 по 17 поступают на линию прерывания №18;
- прерывания от каналов DMA с 18 по 20 поступают на линию прерывания №19;
- прерывания от каналов DMA с 21 по 23 поступают на линию прерывания №20.

Источник прерывания можно определить, считав состояние регистра IRQSTAT. Сброс прерывания осуществляется программно путём установки соответствующего бита в регистре IRQSTATCLR.

11 Порты ввода-вывода

В состав микроконтроллера входят три 16-разрядных порта ввода-вывода: A, B, C. Структуры портов и функционирования – идентичны.

Каждый цифровой вывод порта микроконтроллера может использоваться как двунаправленный вывод общего назначения (режим GPIO). Помимо этого, все выводы имеют альтернативную функцию (или функции).

По умолчанию порты находятся в сбросе и не тактируются. Активировать порты можно с помощью соответствующих бит регистров CGCFGAHB, RSTDISAHB блока RCU.

11.1 Функционирование порта

Полученные данные сохраняются в регистре DATA порта. Данные для передачи записываются в регистр DATAOUT порта. Существует возможность модификации состояния регистра DATAOUT путем записи единиц в регистр DATAOUTSET для установки соответствующих бит, в регистр DATAOUTCLR – для сброса бит, в регистр DATAOUTTGL – для инверсии бит.

Примечание – Регистр DATAOUTTGL, а также все регистры xxxCLR предназначены для изменения конфигурации порта одной записью. Использование операции чтение-модификация-запись с любым из этих регистров может привести к нежелательным результатам, влияющим на другие биты регистра.

На рисунке 11.1 приведена структурная схема вывода цифрового порта микроконтроллера. Схемы всех выводов идентичны.

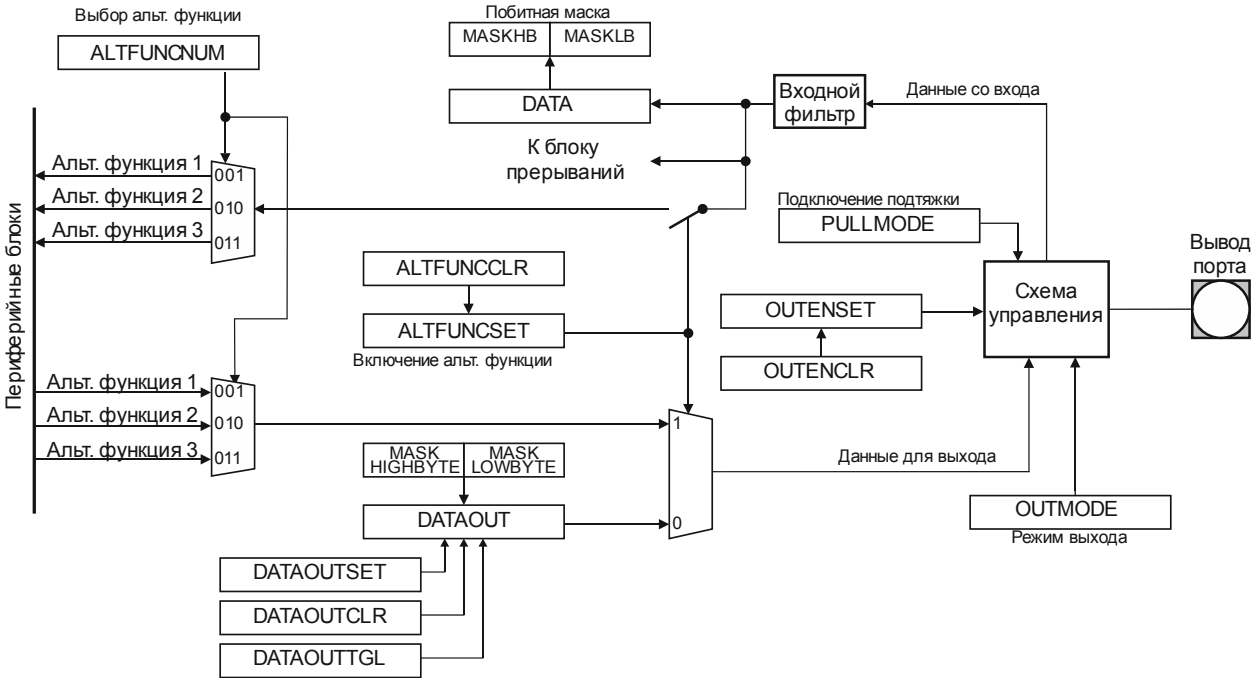


Рисунок 11.1 – Структурная схема вывода цифрового порта микроконтроллера

Схема состоит из двунаправленной площадки вывода, фильтра входных сигналов, мультиплексора выбора номера альтернативной функции, мультиплексора выбора режима работы (режим GPIO либо режим альтернативной функции).

Для каждого вывода задается режим работы, номер альтернативной функции, нагрузочная способность и быстродействие вывода, режим подтяжки, а также производится настройка порта на работу в режиме с открытым стоком/истоком. Для работы с периферийными блоками включается режим альтернативной функции. Входной сигнал

может подаваться для дальнейшей обработки как напрямую (асинхронный вход), так и проходить обработку через фильтр.

После сброса все выводы конфигурируются как выводы общего назначения (режим GPIO) и находятся в третьем состоянии.

Включение подтяжки вывода конфигурируется регистром PULLMODE.

Разрешение работы выходных каскадов определяется состоянием бит регистра OUTENSET (для сброса установленных бит следует записать единицы в регистр OUTENCLR), а их режимы («push-pull», открытый сток/исток) состоянием полей в регистре OUTMODE. В режимах открытый сток/исток управление состоянием вывода осуществляется регистрами DATAOUT, DATAOUTCLR, DATAOUTSET, DATAOUTTGL, при этом режим выхода должен быть отключен записью в регистр OUTENCLR.

11.2 Режим альтернативных функций

Для перевода желаемого вывода порта в режим альтернативной функции необходимо установить соответствующий бит в регистре ALTFUNCSET порта. Для отключения альтернативной функции нужно записать единицу в соответствующий бит регистра ALTFUNCCLR. Выбор номера альтернативной функции осуществляется с помощью регистра ALTFUNCNUM. Каждому выводу соответствуют два бита регистра.

Входы и выходы периферийных блоков в процессе работы коммутируются с выводами микроконтроллера при условии, что для этих выводов включен режим альтернативной функции. В связи с этим периферийный блок может передавать информацию на несколько выводов одновременно. В то же время прием информации может осуществляться только с одного вывода, во избежание конфликтов уровней сигналов (для этого дополнительно предусмотрена система приоритета альтернативных функций). Количество выводов, сигналы с которых могут быть переданы на периферийный блок, для каждого блока различно.

Распределение приоритетов для входных функций происходит следующим образом: наивысший приоритет имеют функции с наименьшим номером, наинизший приоритет – функции с наибольшим номером. Но при этом, если на разных выводах встречаются идентичные функции с одинаковыми номерами, то их приоритеты распределяются по букве порта и номеру пина: чем номер пина меньше, а буква порта ближе к началу алфавита, тем приоритет больше.

Указанные правила программирования выводов для приема данных распространяются на все периферийные блоки.

11.3 Входные фильтры

Ко всем площадкам выводов подключены входные фильтры. На рисунке 11.2 показана структурная схема фильтра вывода порта.

Входной сигнал с вывода порта может приниматься как напрямую (асинхронный режим), так и пересинхронизироваться (синхронизироваться с тактовой частотой работы микроконтроллера). Управление осуществляется регистрами SYNCSET/SYNCLCR.

Дополнительно есть возможность включения накопления трех или шести отсчетов входного сигнала для помехоустойчивости вывода. Если результаты всех отсчетов совпадают, сигнал передается дальше по схеме, в противном случае состояние сигнала не меняется. Временные интервалы между отсчетами задаются в количестве тактов системной частоты посредством регистра QUALSAMPLE. Временной интервал задается один для всех выводов порта.

Включение фильтра и задание режима его работы осуществляется посредством регистров QUALSET/QUALCLR и QUALMODESET/QUALMODECLR соответственно.

Одновременно оба режима активными быть не могут – при установленных единицах в одних и тех же разрядах SYNCSET и QUALSET сигнал будет проходить напрямую с входа фильтра на его выход.

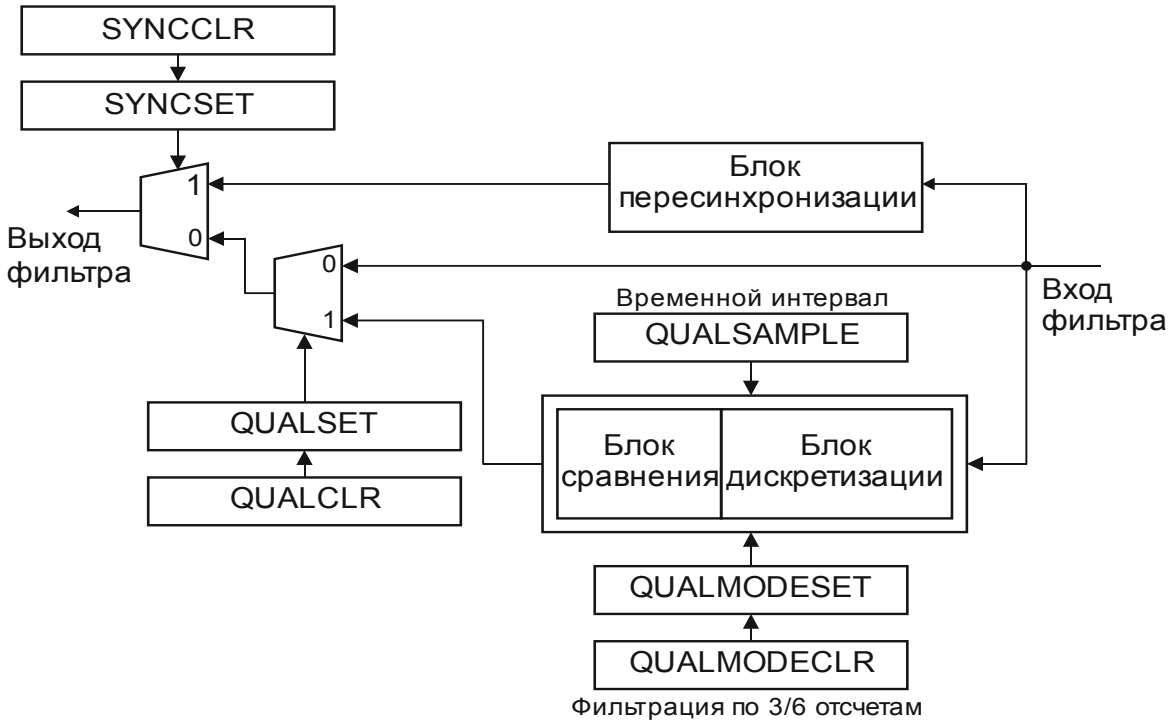


Рисунок 11.2 – Структурная схема фильтра вывода порта

11.4 Прерывания

Каждый из выводов способен генерировать прерывание. На рисунке 11.3 показана структурная схема блока генерации прерываний.

Схема вывода позволяет также осуществлять гибкое управление прерываниями и задавать, по какому аппаратному событию генерировать прерывание (по какому фронту или уровню). При возникновении прерывания в регистре INTSTATUS устанавливается соответствующий флаг, и выставляется прерывание в контроллере прерываний PLIC.

Прерывание может быть сброшено программно записью единицы в соответствующий бит регистра INTSTATUS. Для разрешения прерывания вывода порта следует записать единицу в соответствующий выводу бит регистра INTENSET, а для запрета прерывания – единицу в бит регистра INTENCLR.

Для задания типа события (уровень или фронт), по которому генерируется прерывание, используется регистр INTTYPESET, для задания полярности (низкий/высокий уровень или положительный/отрицательный фронт) используется INTPOLSET, а для сброса настроек – INTTYPECLR и INTPOLCLR соответственно.

Существует возможность организации прерывания по обоим фронтам – сначала необходимо задать тип прерывания – фронт (запись в INTTYPESET), а затем записать соответствующие единицы в INTEDGESET. В этом режиме состояние регистра полярности INTPOLSET игнорируется. Отключить режим генерации прерывания по обоим фронтам можно записью в INTEDGECLR, в таком случае для генерации в дальнейшем будет использована текущая настройка полярности (регистр INTPOLSET).

В режиме прерывания по уровню состояние регистра INTEDGESET не влияет на их генерацию.

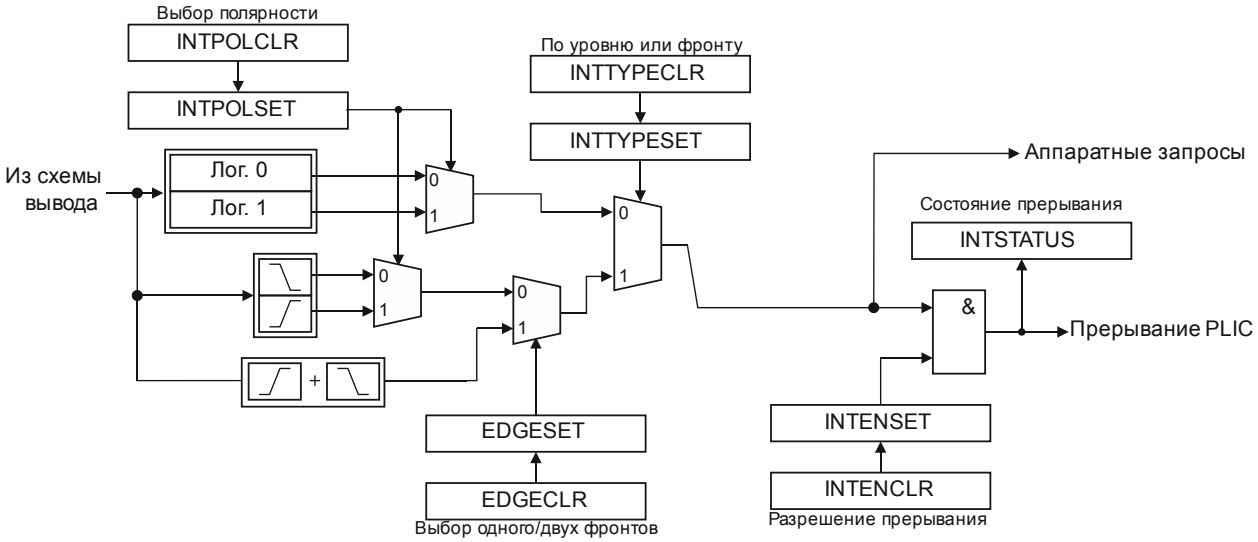


Рисунок 11.3 – Структурная схема блока генерации прерываний

11.5 Генерация аппаратных запросов

Кроме генерации прерываний, порты также имеют возможность генерации запросов к другой периферии. Генерация запроса происходит по условиям возникновения прерывания на выбранном выводе, при этом не важно, маскировано само прерывание или нет (состояние INTENSET), влияние оказывают только настройки генерации прерывания (INTTYPESET, INTPOLSET, INTEDGESET).

Для того чтобы в соответствии с настройками прерывания генерировался BREQ запрос к контроллеру DMA, необходимо осуществить запись единиц в DMAREQSET. Отключение генерации запросов к контроллеру DMA осуществляется через DMAREQCLR.

Для того чтобы в соответствии с настройками прерывания сгенерировался запрос начала преобразования АЦП, необходимо осуществить запись единиц в ADCSOCSET. Отключение генерации запросов начала преобразования АЦП осуществляется через регистр ADCSOCLR.

11.6 Механизм блокировки конфигурации

Для защиты конфигурации выводов от несанкционированного изменения реализован механизм блокировки.

Блокировка конфигурации осуществляется записью соответствующих единиц в регистр LOCKSET, сброс блокировки – в регистр LOCKCLR. При включенной блокировке игнорируется запись во все биты и поля настройки маскированных выводов – доступны для записи лишь регистры INTSTATUS и QUALSAMPLE.

По умолчанию все выводы являются разблокированными, а запись в регистры LOCKSET/LOCKCLR игнорируется. Для того чтобы сделать доступными для изменения регистры LOCKSET/LOCKCLR, необходимо в регистр LOCKKEY занести значение ключа ADEADBEEh. Если необходимо снова защитить от изменений регистры LOCKSET/LOCKCLR, то достаточно записать в LOCKKEY любое значение, отличное от ключа.

Текущий статус защиты регистров LOCKSET/LOCKCLR можно узнать, прочитав регистр LOCKKEY – значение 0000_0001h соответствует снятой защите, 0000_0000h – установленной.

11.7 Механизм маскирования

Для управления состоянием выводов порта дополнительно используется механизм маскирования. Он позволяет устанавливать желаемый уровень сигнала на нужном выводе,

не затрагивая состояние других выводов. 16-разрядный порт условно разбивается на старший байт и младший байт. Для доступа по маске к младшему байту используется массив регистров MASKLB, а к старшему – MASKHB.

Каждый массив состоит из 256 регистров, каждый регистр имеет порядковый номер (от 00h до FFh), который является маской. Так, например, для порта A выделены две области памяти с адресами: 2800_0400h – 2800_07FCh для младшего байта и 2800_0800h – 2800_0BFCh для старшего байта. Биты с 9 по 2 адреса являются маской. Таким образом, адресу 2800_0400h соответствует маска 00h (MASKLB[0x00]), адресу 2800_0404h – 01h (MASKLB[0x01]) и т. д.

Для того чтобы изменить состояние выводов порта с использованием маски, нужно записать новое значение в соответствующий элемент массива (MASKLB или MASKHB) с порядковым номером, совпадающим со значением маски.

Разряды порта, закрытые «нулями» маски, останутся неизменными, а остальные примут новые значения. На рисунке 11.4 показан механизм маскирования младшего байта порта A.

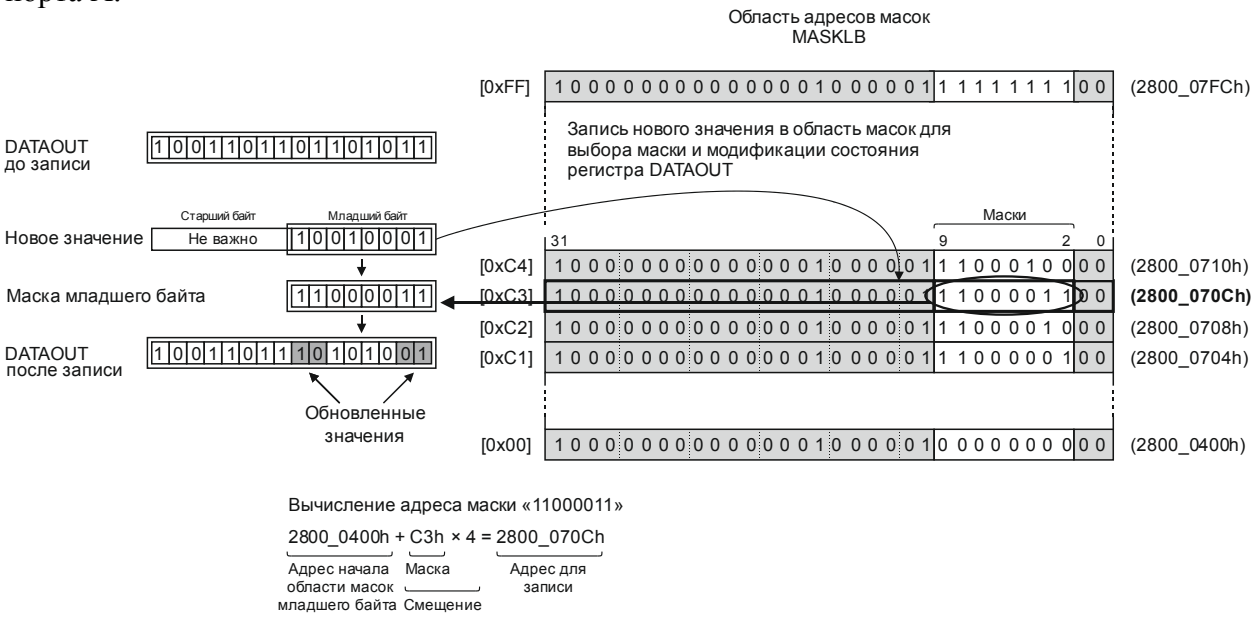


Рисунок 11.4 – Механизм изменения состояния младшего байта порта A с маскированием

Для изменения 0, 1, 6 и 7 битов регистра порта нужно использовать маску 1100_0011b. Эта маска является частью (биты с 9 по 2) адреса 2800_070Ch. Новое значение XX90h данных, которое требуется передать в порт (при этом старший байт числа не важен), нужно записать в ячейку с адресом 2800_070Ch. Далее это значение будет аппаратно маскировано и размещено в регистре порта DATAOUT.

Аналогично выполняется маскирование старшего байта, см. рисунок 11.5. Разница лишь в том, что в данном случае берется старший байт нового значения, а младший не важен.

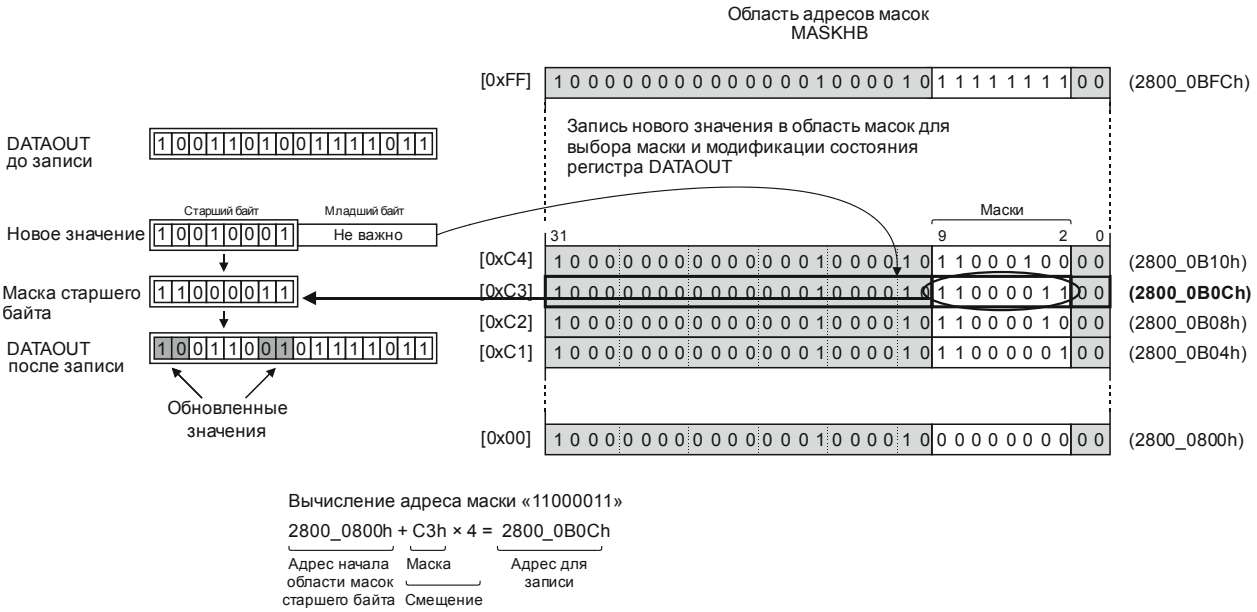


Рисунок 11.5 – Механизм изменения состояния старшего байта порта A с маскированием

12 Таймер TMR32

Микроконтроллер содержит один блок 32-разрядного таймера TMR32.

TMR32 – 32-разрядный таймер/счетчик с четырьмя устройствами захвата/сравнения. Таймер поддерживает несколько захватов/сравнений, выход ШИМ и выдержку временных интервалов. Обладает обширными возможностями прерываний, которые могут быть сгенерированы как при переполнении счетчика, так и от регистров захвата/сравнения. Таймер TMR32 имеет внешние выводы, подключаемые через альтернативные функции портов GPIO. Так, вход TMR32_EXTIN используется в качестве внешнего тактового сигнала, сигналы TMR32_CCIA и TMR32_CCIB – входы блоков CAPCOM, а сигналы TMR32_OUT0 – TMR32_OUT3 – выходы блоков CAPCOM, входящих в состав таймера.

Характеристики Таймера:

- 32-разрядный таймер/счетчик с четырьмя режимами работы;
- конфигурируемый источник тактового сигнала;
- четыре конфигурируемых регистра захвата/сравнения.

Структурная схема таймера представлена на рисунке 12.1.

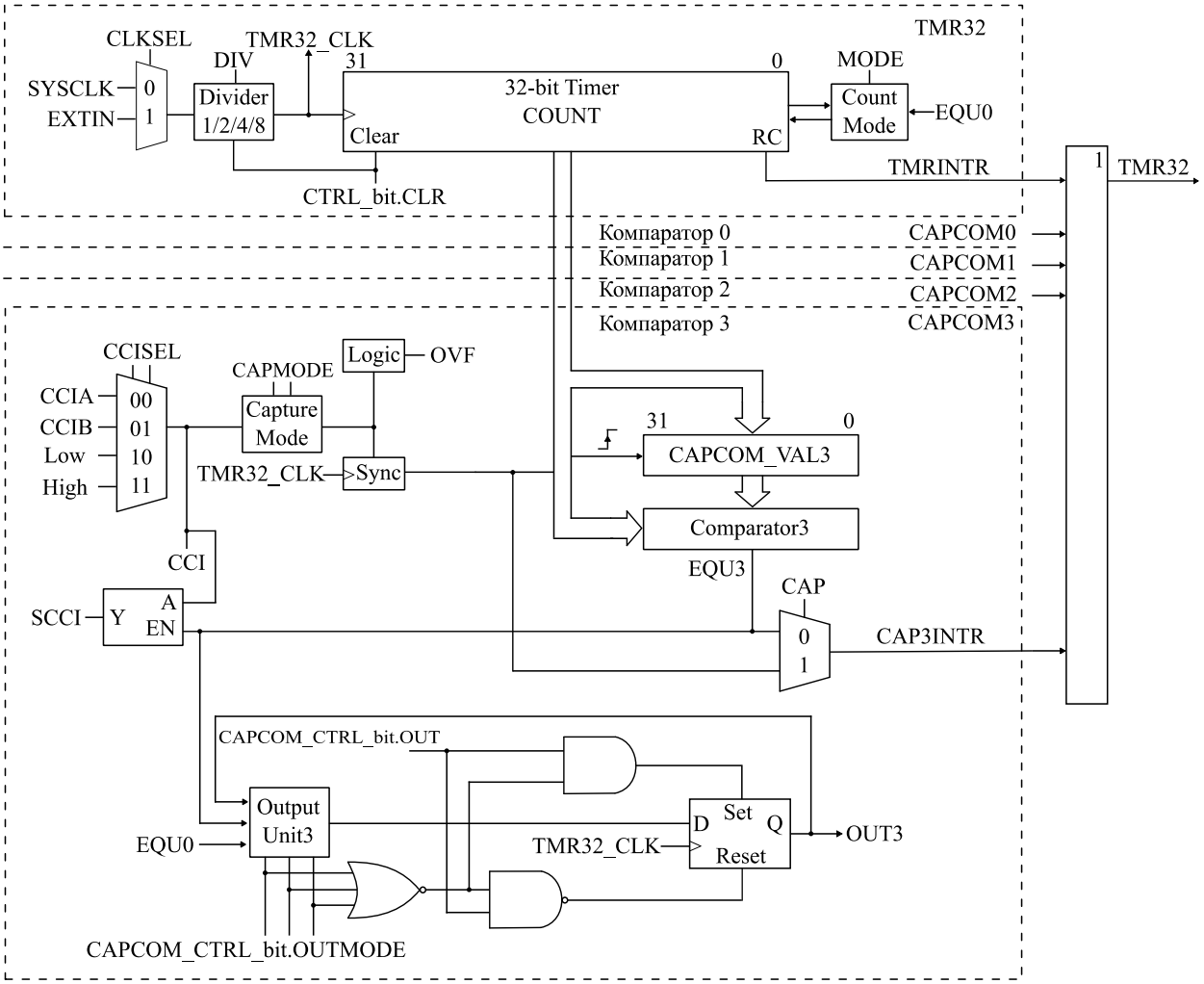


Рисунок 12.1 – Структурная схема таймера

12.1 Функционирование таймера

32-разрядный таймер-счетчик COUNT

Регистр таймера-счетчика COUNT инкрементируется или декрементируется (в зависимости от режима работы) по переднему фронту тактового сигнала. Счетчик может быть считан или записан программно. Кроме того, таймер может генерировать прерывания при переполнении. Регистр COUNT очищается установкой бита CLR регистра CTRL. Бит CLR также очищает делитель тактовой частоты и направление счета для режима вверх/вниз.

Примечание – Рекомендуется останавливать таймер перед изменением режима его работы (за исключением разрешения прерываний, операций с флагом прерываний и битом CLR) во избежание некорректной работы. В случае если тактовый сигнал таймера асинхронен тактовому сигналу SYSCLK, любое чтение регистра COUNT должно происходить при остановленном таймере, иначе результат может быть непредсказуем. В качестве альтернативы возможно считывание регистра в рабочем состоянии несколько раз и выбор корректного значения программно (наиболее частое значение). Любая запись в регистр COUNT приводит к изменению счетчика таймера.

Выбор источника тактирования и делитель.

Тактирование таймера осуществляется тактовыми сигналом SYSCLK либо от внешнего источника через вывод TMR32_EXTIN. Источник тактирования определяется битом CLKSEL. Выбранный сигнал синхронизации передается либо напрямую, либо через делитель DIV регистра CTRL. При выставлении бита CLR делитель сбрасывается.

Запуск таймера.

Таймер запускается или перезапускается при соблюдении следующих условий:

1 Таймер считает, если в регистре CTRL значение поля MODE > 0 (режим работы) и источник тактового сигнала активен.

2 В режиме работы «вверх» или «вверх/вниз» таймер может быть остановлен путем записи 0 в регистр сравнения CAPCOM[0].VAL. Его можно запустить повторно посредством записи ненулевого значения в регистр CAPCOM[0].VAL. В этом случае таймер инкрементируется с нуля.

12.2 Режимы счёта

Управление режимом таймера

Таймер имеет 4 режима работы, описанных в таблице 12.1: «стоп», «вверх», «непрерывный» и «вверх/вниз». Режимы выбираются с помощью поля MODE регистра CTRL.

Таблица 12.1 - Режимы работы таймера

MODE	Режим	Описание
00	Стоп	Остановка таймера
01	Вверх	Многократный счет от 0 до значения в регистре CAPCOM[0].VAL.
10	Непрерывный	Многократный счет от 0 до значения FFFF_FFFFh.
11	Вверх/вниз	Многократный счет от 0 вверх до значения в CAPCOM[0].VAL и обратно до 0.

Режим «вверх»

Режим «вверх» используется при значении периода таймера, отличном от значения FFFF_FFFFh. Таймер циклично считает до значения регистра сравнения CAPCOM[0].VAL, задающего период, как показано на рисунке 12.2. Количество отсчетов в периоде равно (CAPCOM[0].VAL + 1). Когда значение таймера становится равным значению регистра сравнения (COUNT = CAPCOM[0].VAL), таймер перезапускается, начиная счет с нуля. Если режим «вверх» выбран, когда COUNT > CAPCOM[0].VAL, таймер перезапускается немедленно.

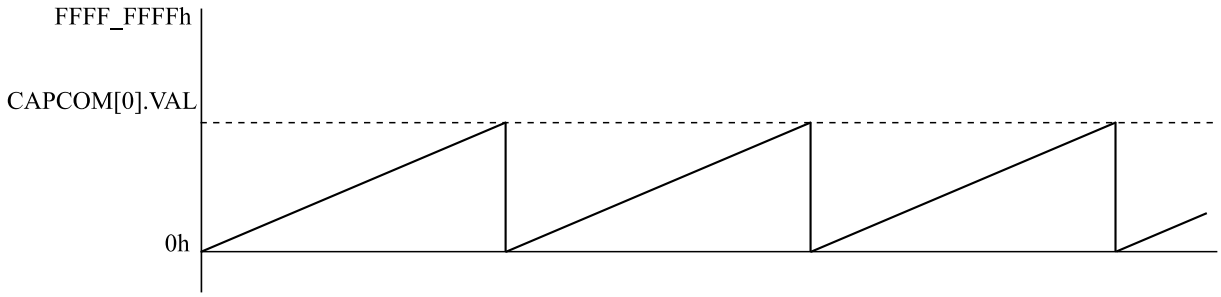


Рисунок 12.2 — Режим «вверх»

Флаг прерывания CAP0 регистра RIS устанавливается, когда значение счетчика COUNT становится равным значению регистра сравнения CAPCOM[0].VAL. Флаг прерывания TMR устанавливается, когда значение счетчика COUNT , равное значению CAPCOM[0].VAL, переключается в 0.

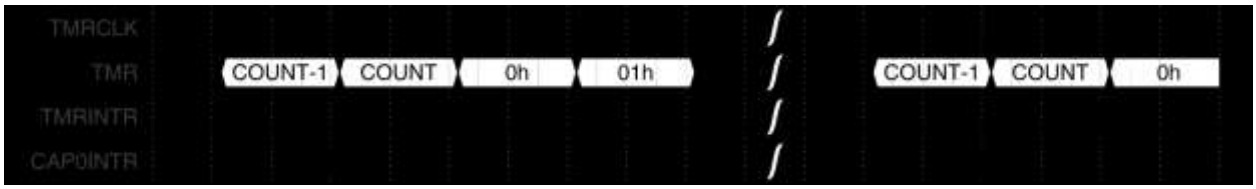


Рисунок 12.3 — Установка флагов прерываний в режиме «вверх»

Изменение регистра периода (захвата/сравнения) CAPCOM[0].VAL .

При изменении регистра CAPCOM[0].VAL во время работы таймера, таймер продолжает отсчет вверх до новой границы периода, если новый период больше или равен старому или больше текущего значения счетчика. Если новый период меньше текущего значения счетчика, таймер обнуляется. Однако до обнуления счетчика может произойти один дополнительный отсчет.

Непрерывный режим

В непрерывном режиме таймер циклично считает вверх до значения FFFF_FFFFh и перезапускается с 0, как показано на рисунке 12.4. Регистр захвата/сравнения CAPCOM[0].VAL работает так же, как и остальные регистры захвата/сравнения.

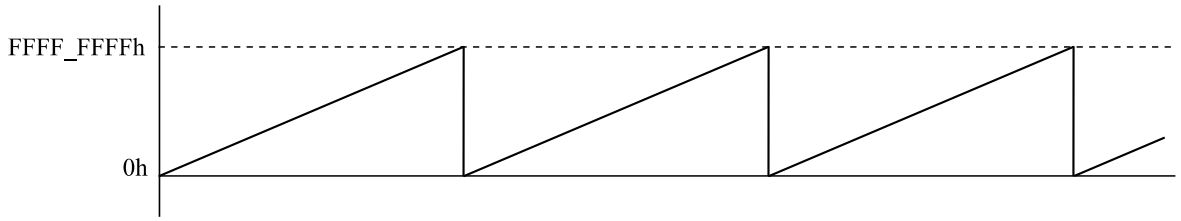


Рисунок 12.4 — Непрерывный режим

Флаг прерывания TMR устанавливается при переключении счетчика из значения FFFF_FFFFh в значение 0. Цикл выставления флага представлен на рисунке 12.5.

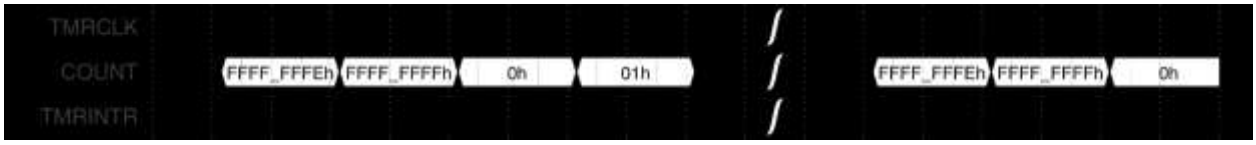


Рисунок 12.5 — Установка флага прерываний в непрерывном режиме

Использование непрерывного режима

Непрерывный режим может использоваться для генерации независимых временных интервалов и выходных частот. Каждый раз по завершении интервала генерируется прерывание. Затем временной интервал добавляется к регистру CAPCOM[n].VAL (где n - номер блока CAPCOM) в подпрограмме обработки прерывания. На рисунке 12.6 показаны два разных временных интервала t0 и t1, записываемые в регистры захвата / сравнения. В этом случае временной интервал контролируется аппаратно, а не программно, без влияния задержки обработки прерывания.

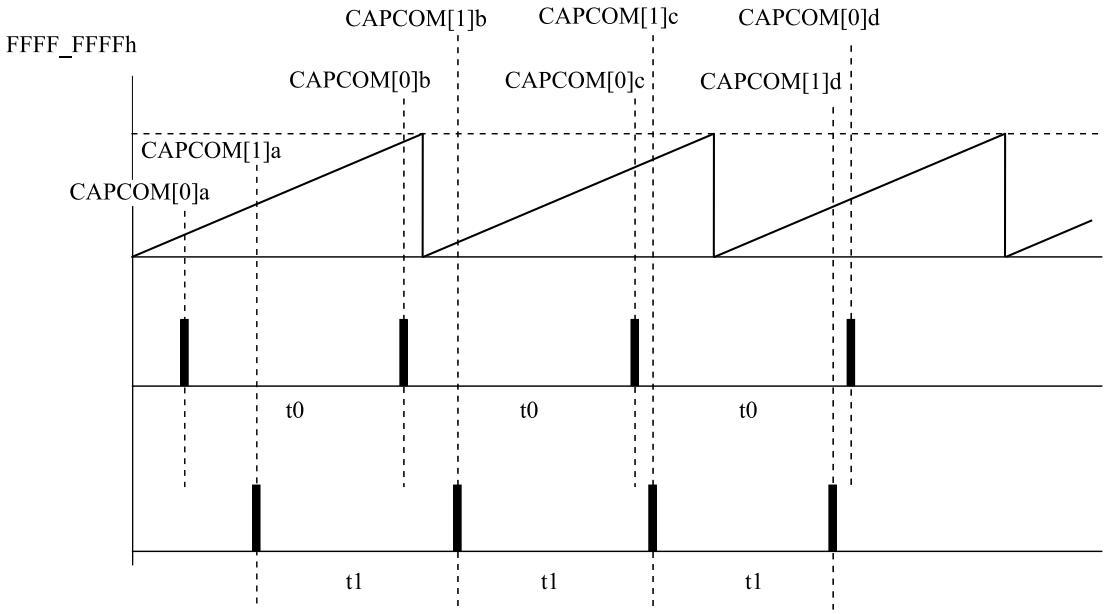


Рисунок 12.6 — Временные интервалы в непрерывном режиме

Временные интервалы могут быть сформированы и в других режимах, где в качестве регистра периода используется CAPCOM[0].VAL. Их обработка более сложная, поскольку сумма старых данных CAPCOM[n].VAL и нового периода может быть выше значения CAPCOM[0].VAL. Если предыдущее значение CAPCOM[n].VAL плюс tx больше, чем

данные CAPCOM[0].VAL, значение CAPCOM[0].VAL необходимо вычесть, чтобы получить правильный временной интервал.

Режим «вверх/вниз»

Режим «вверх/вниз» используется при необходимости генерации симметричных импульсов в периоде таймера, отличном от FFFF_FFFFh. Таймер непрерывно считает до значения регистра захвата / сравнения CAPCOM[0].VAL и назад к 0, как показано на рисунке 12.7. Период равен удвоенному значению регистра CAPCOM[0].VAL.

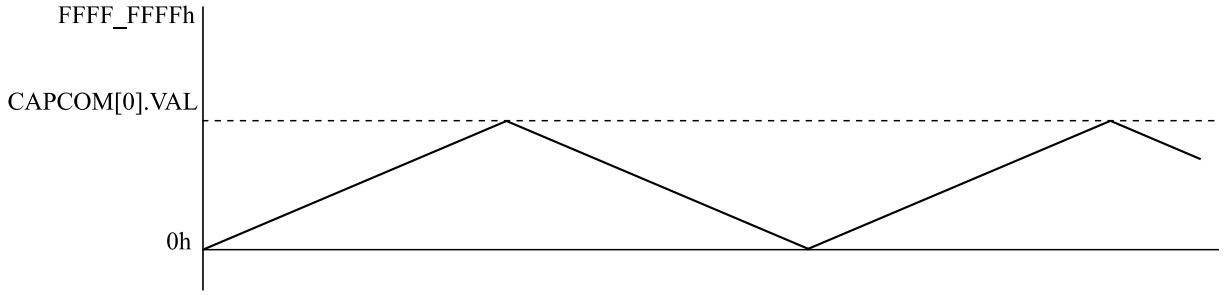


Рисунок 12.7 — Режим «вверх/вниз»

Направление счета запоминается. Это позволяет таймеру останавливаться и запускаться в том же направлении отсчета, которое было до останова. Очистить направление можно выставлением бита CLR, который очищает также значения регистра COUNT и биты делителя тактовой частоты DIV регистра CTRL.

В режиме «вверх/вниз» флаги прерываний CAP0 и TMR устанавливаются лишь единожды за период с интервалом ½ периода. Флаг CAP0 устанавливается при переключении значения счетчика таймера COUNT из значения COUNT = CAPCOM[0].VAL – 1 в значение COUNT = CAPCOM[0].VAL. Флаг TMR устанавливается при переключении значения COUNT из 1 в 0. Цикл выставления флагов представлен на рисунке 12.8.

Рисунок 12.8 – Установка флагов прерываний в режиме «вверх/вниз»

Изменение регистра периода CAPCOM[0].VAL

При изменении CAPCOM[0].VAL во время работы таймера и отсчете в направлении вниз таймер продолжает свой спуск до тех пор, пока не достигнет нуля. Новый период вступает в силу после того, как счетчик начнет обратный отсчет до нуля.

Когда таймер ведет отсчет в направлении вверх, и новый период больше или равен старому периоду, или больше текущего значения отсчета, таймер ведет отсчет до нового периода, прежде чем начать обратный отсчет. Когда таймер ведет отсчет в направлении вверх, и новый период меньше текущего значения отсчета, таймер начинает обратный отсчет. Однако может произойти еще один подсчет, прежде чем счетчик начнет обратный отсчет.

Использование режима «Вверх/вниз»

Режим вверх/вниз необходим, когда требуется время простоя между выходными сигналами. Например, чтобы избежать перегрузки, два выхода, управляющие Н-мостом, никогда не должны находиться в высоком состоянии одновременно. В примере, показанном на рисунке 9, t_{dead} является:

$$t_{dead} = t_{timer} (CAPCOM[1].VAL – CAPCOM[2].VAL)$$

где t_{dead} - Время, в течение которого оба выхода должны быть неактивны;

t_{timer} - Время цикла работы таймера;

CAPCOM[n].VAL - Содержимое регистра захвата/сравнения.

Регистры CAPCOM[n].VAL не буферизуются. Они обновляются сразу же после записи. Таким образом, любое требуемое время простоя не будет поддерживаться автоматически.

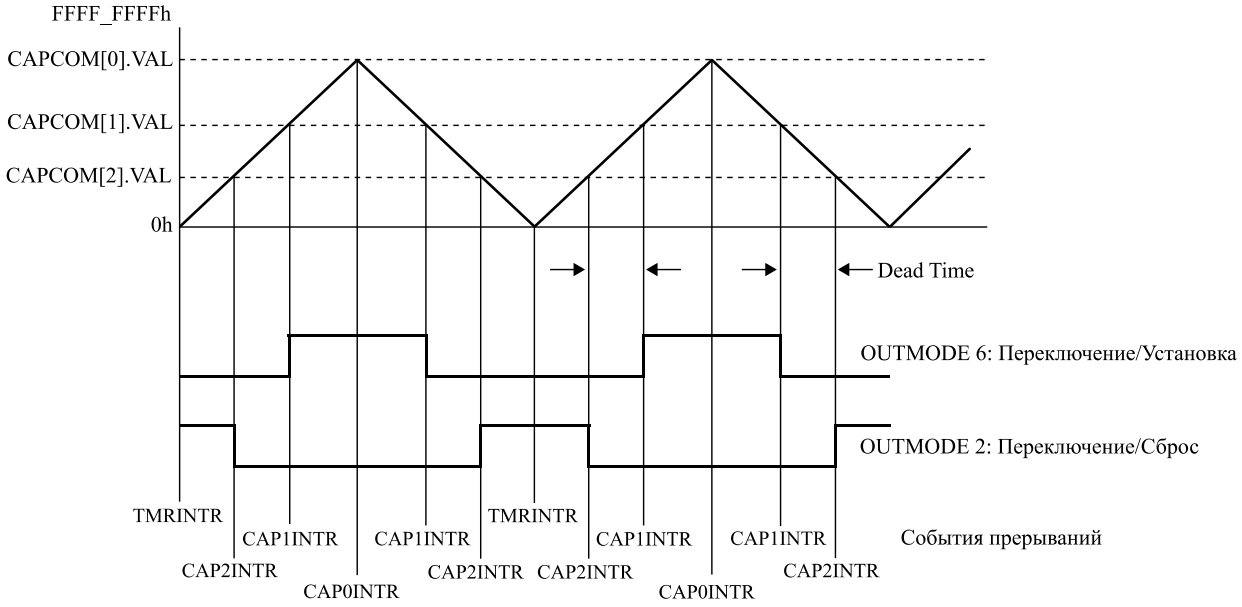


Рисунок 12.9 – Блок вывода в режиме «вверх/вниз»

12.3 Блоки захвата/сравнения

В TMR32 присутствуют четыре идентичных блока захвата/сравнения, CAPCOM[n].VAL. Любой из блоков может быть использован для сбора данных таймера или для генерации выходного сигнала с заданными временными интервалами.

Режим захвата

Режим захвата выбирается, при установке бита CAP = 1 в регистре CAPCOM[n].CTRL. Режим захвата используется для записи временных событий. Он может быть использован для вычисления скорости или измерения времени. Входы захвата TMR32_CCIA и TMR32_CCIB подключены к внешним выводам или внутренним сигналам (Low – уровень логического нуля, High – уровень логической единицы) и выбираются с помощью битов CCISEL. Биты CAPMODE выбирают фронт захвата входного сигнала как нарастающий, спадающий или и то, и другое вместе. Захват происходит на выбранном фронте входного сигнала.

В случае захвата:

- значение таймера копируется в регистр CAPCOM[n].VAL;
- устанавливается флаг прерывания CAPn в регистре RIS.

Уровень входного сигнала может быть считан в любое время с помощью бита CCI регистра CAPCOM[n].CTRL. Сигналы захвата TMR32_CCIA и TMR32_CCIB синхронизируются с частотой таймера. Синхронизация внешнего сигнала захвата изображена на рисунке 12.10.

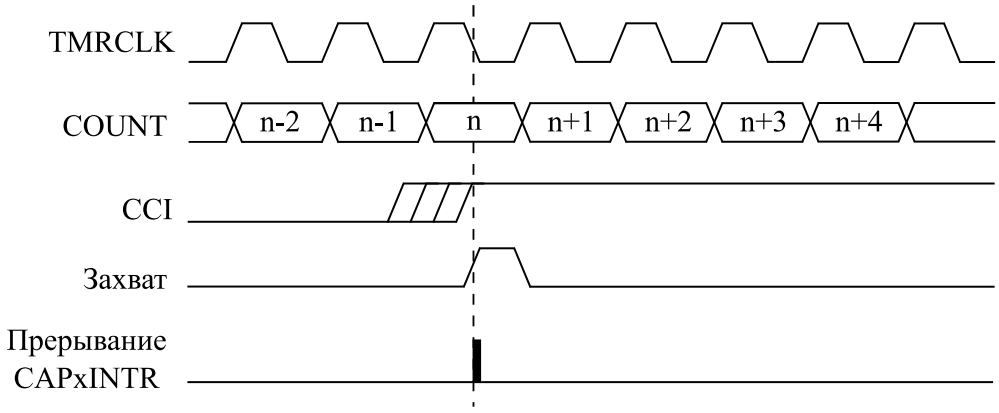


Рисунок 12.10 – синхронизация внешнего сигнала захвата

В случае захвата нового события без считывания предыдущего значения из регистра CAPCOM[n].VAL происходит перезапись регистра новым значением и устанавливается бит OVF регистра CAPCOM[n].CTRL как проиллюстрировано на рисунке 12.11. Сброс бита OVF производится программно.

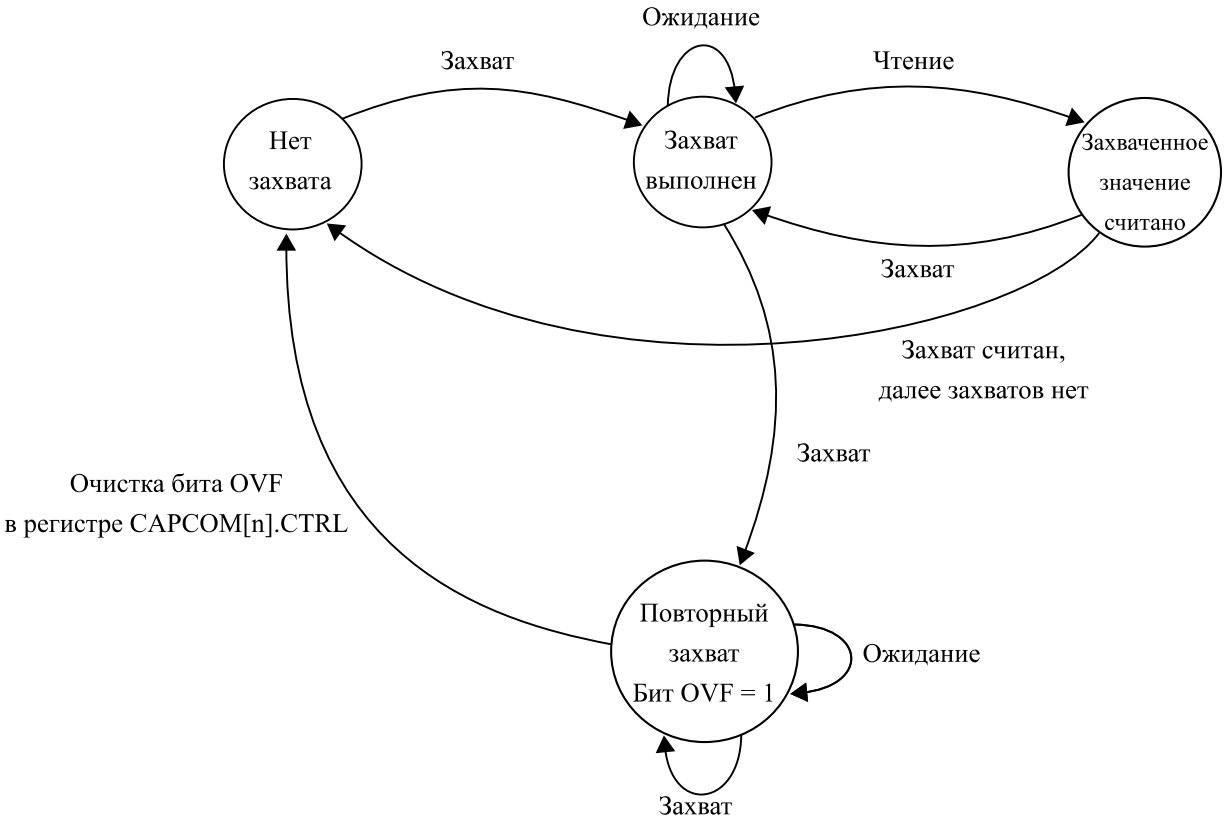


Рисунок 12.11 – Цикл захвата

Программная инициализация захвата

Захват может быть инициализирован программно. Биты CAPMODE могут быть установлены для захвата на обоих фронтах. Затем программа устанавливает CCISEL[1] = 1 и переключает бит CCISEL[0] для переключения сигнала захвата между High и Low.

Событие захвата (захват состояния внешнего сигнала) будет происходить при переключении битового поля CCISEL[0] между значениями High и Low.

Режим сравнения

Режим сравнения выбирается при CAP = 0 в регистре CAPCOM[n].CTRL. Режим сравнения используется для генерации выходных сигналов ШИМ или прерываний через определенные промежутки времени. Когда COUNT досчитывает до значения в CAPCOM[n].VAL:

- устанавливается флаг прерывания CAPn;
- внутренний сигнал EQUn = 1;
- EQUn влияет на выход в соответствии с режимом вывода;
- входной сигнал CCI фиксируется в SCCI.

12.4 Выходной модуль

Каждый блок захвата/сравнения содержит блок вывода. Блок вывода используется для генерации выходных сигналов, таких как ШИМ-сигналы. Каждый выходной блок имеет восемь режимов работы, которые генерируют сигналы на основе сигналов EQU0 и EQUn.

Режимы вывода

Режимы вывода определяются битами OUTMODE регистра CAPCOM[n].CTRL, они описаны в таблице 12.2. Выходной сигнал изменяется по переднему фронту тактов таймера для всех режимов, кроме режима 0. Режимы вывода 2, 3, 6 и 7 непригодны для блока вывода 0, поскольку EQUn = EQU0.

Таблица 12.2 - Режимы вывода

№	OUTMODE	Режим	Описание
0	000	Вывод	Выходной сигнал TMR32_OUTn определяется битом OUT регистра CAPCOM[n].CTRL. Сигнал TMR32_OUTn обновляется немедленно при обновлении бита OUT
1	001	Установка	Выходной сигнал устанавливается, когда таймер отсчитывает значение CAPCOM[n].VAL. Он остается установленным до тех пор, пока не произойдет сброс таймера или пока не будет выбран другой режим вывода, который повлияет на выходной сигнал
2	010	Переключение/сброс	Выход переключается, когда таймер отсчитывает значение CAPCOM[n].VAL. Он сбрасывается, когда таймер отсчитывает значение CAPCOM[0].VAL
3	011	Установка/сброс	Выходной сигнал устанавливается, когда таймер отсчитывает значение CAPCOM[n].VAL. Он сбрасывается, когда таймер отсчитывает значение CAPCOM[0].VAL
4	100	Переключение	Выход переключается, когда таймер отсчитывает значение CAPCOM[n].VAL. Период вывода в два раза превышает период таймера

Продолжение таблицы 12.2

№	OUTMODE	Режим	Описание
5	101	Сброс	Выходной сигнал сбрасывается, когда таймер отсчитывает значение CAPCOM[n].VAL. Он остается сброшенным до тех пор, пока не будет выбран другой режим вывода, который повлияет на выходной сигнал
6	110	Переключение/ установка	Выход переключается, когда таймер отсчитывает значение CAPCOM[n].VAL. Он устанавливается, когда таймер отсчитывает значение CAPCOM[0].VAL
7	111	Сброс/установка	Выходной сигнал сбрасывается, когда таймер отсчитывает значение CAPCOM[n].VAL. Он устанавливается, когда таймер отсчитывает значение CAPCOM[0].VAL
Примечание – n – номер блока CAPCOM.			

Пример вывода — таймер в режиме счета вверх

Сигнал TMR32_OUTn изменяется, когда таймер досчитывает до значения CAPCOM[n].VAL, и переключается со значения CAPCOM[0].VAL в ноль, в зависимости от режима вывода. Пример показан на рисунке 12.12 с использованием CAPCOM[0].VAL и CAPCOM[1].VAL.

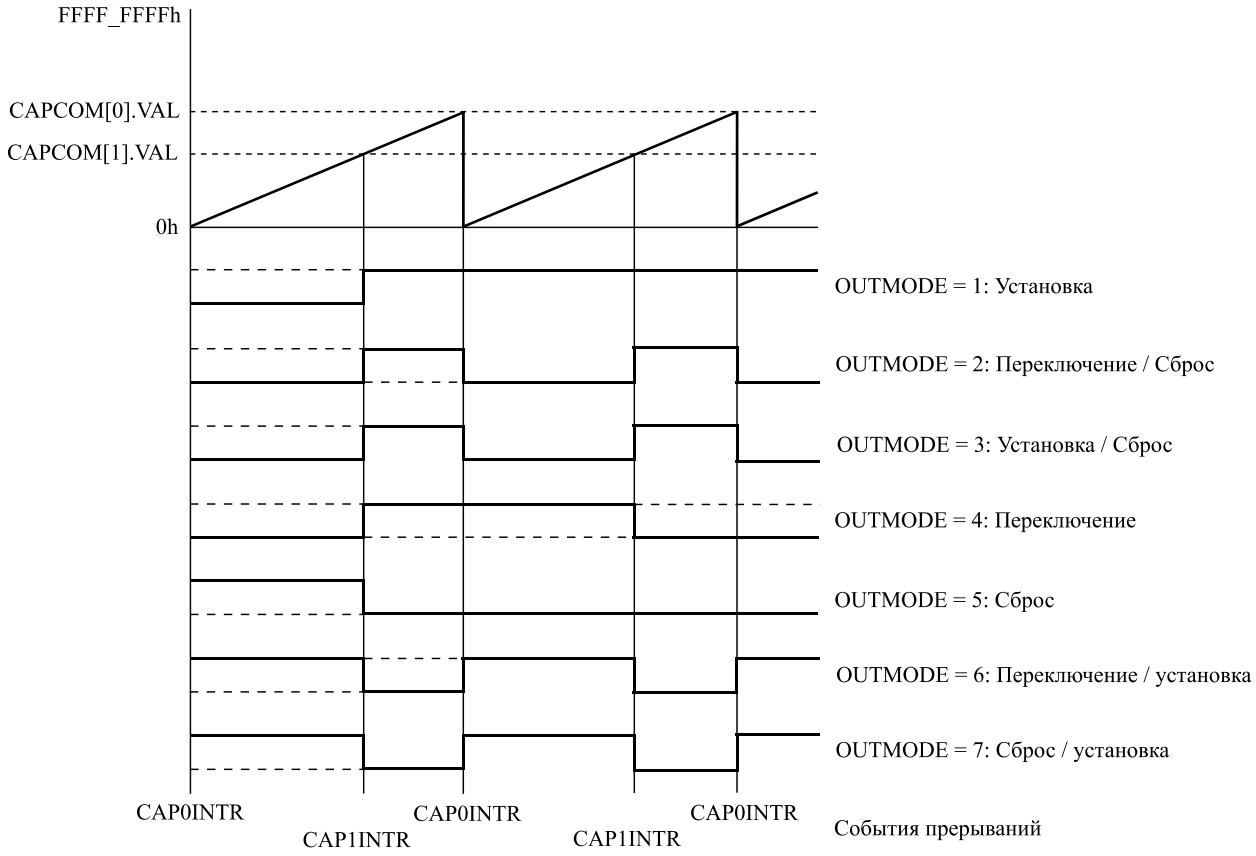


Рисунок 12.12 – Таймер в режиме счета вверх (пример вывода)

Пример вывода — Таймер в непрерывном режиме

Сигнал TMR32_OUTn изменяется, когда таймер достигает значений CAPCOM[n].VAL и CAPCOM[0].VAL, в зависимости от режима вывода. Пример показан на рисунке 12.13 с использованием CAPCOM[0].VAL и CAPCOM[1].VAL.

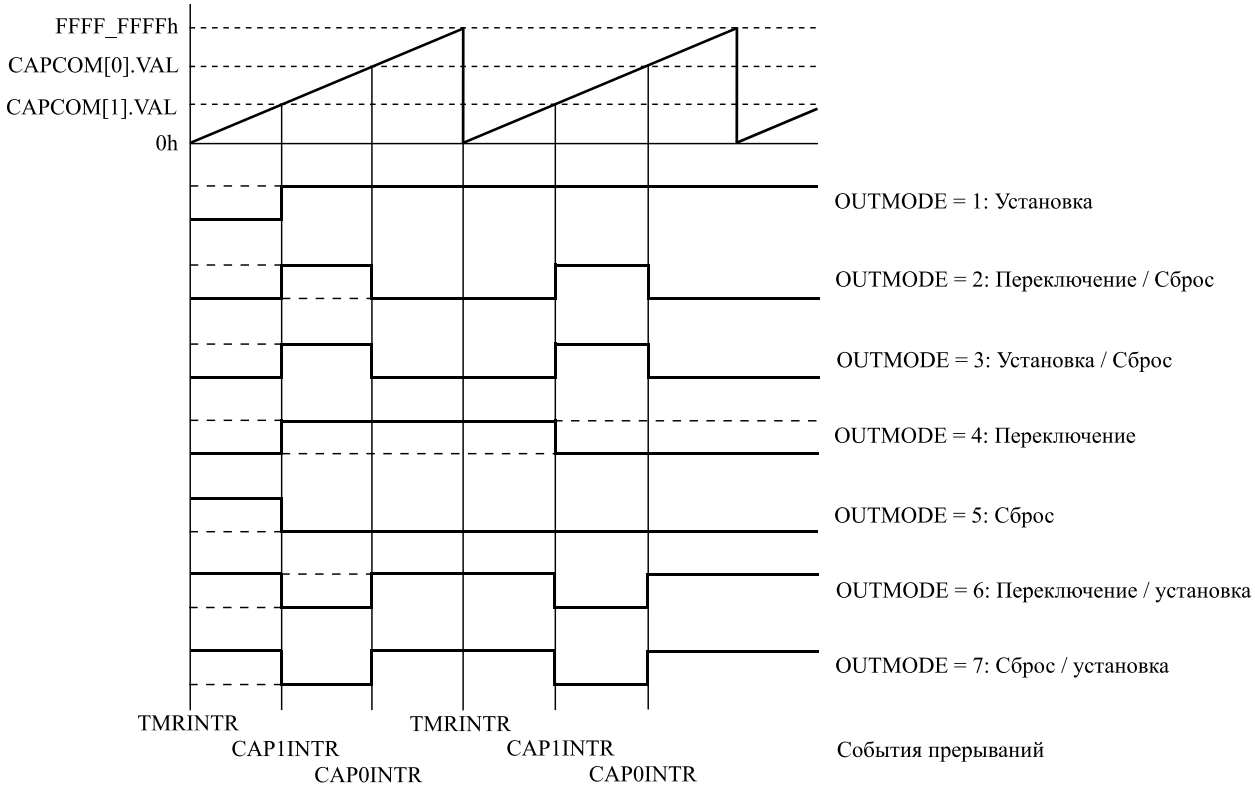


Рисунок 12.13 – Таймер в непрерывном режиме (пример вывода)

Пример вывода — Таймер в режиме счета «вверх/вниз»

Сигнал TMR32_OUTn изменяется, когда таймер равен CAPCOM[n].VAL в любом направлении счета и когда таймер равен CAPCOM[0].VAL, в зависимости от режима вывода. Пример показан на рисунке 12.14 с использованием CAPCOM[0].VAL и CAPCOM[2].VAL.

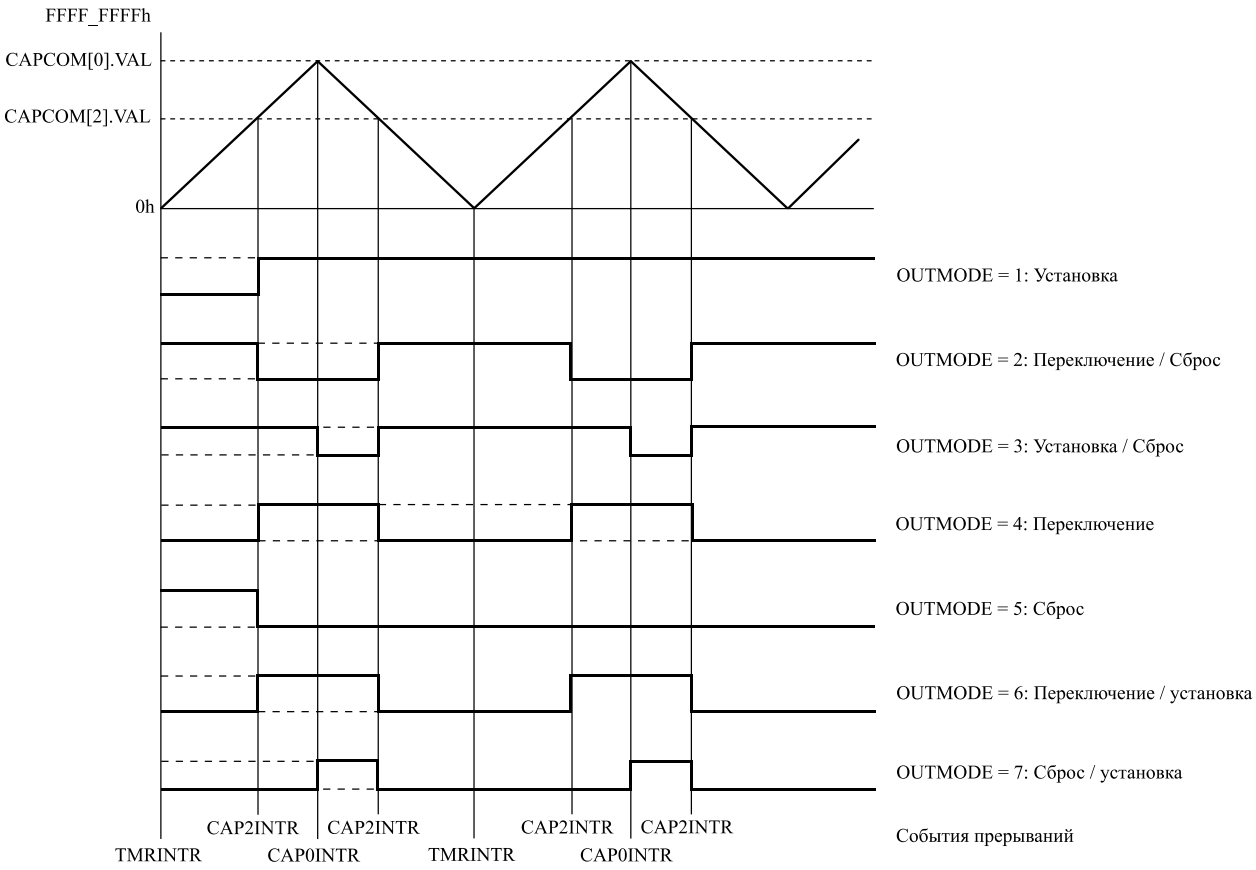


Рисунок 12.14 – Таймер в режиме счета «вверх/вниз» (пример вывода)

Переключение между режимами вывода

При переключении между режимами вывода во избежание сбоя один из битов OUTMODE регистра CAPCOM[n].CTRL должен оставаться установленным, кроме переключения в режим 0. Безопасным методом переключения между режимами вывода является использование режима вывода 7 в качестве переходного состояния.

12.5 Прерывания таймера

В модуле таймера предусмотрено пять маскируемых источников прерываний.

Сигналы запросов на прерывания:

- TMRINTR – сигнал прерывания от таймера;
- CAP0INTR – от 0 модуля CAPCOM;
- CAP1INTR – от 1 модуля CAPCOM;
- CAP2INTR – от 2 модуля CAPCOM;
- CAP3INTR – от 3 модуля CAPCOM.

Каждый из сигналов может быть маскирован путем установки соответствующего бита в регистре маски IM.

Источник прерывания также можно определить, считав состояние регистра RIS или регистра MIS (маскированные прерывания). Сброс прерывания осуществляется программно путём установки соответствующего бита в регистре IC.

Запрос TMRINTR возникает, когда значение счетчика таймера COUNT принимает значение равное нулю. Периодичность запросов равна периоду счета таймера.

В режиме захвата запросы CAPxINTR возникают при наступлении события захвата внешнего сигнала в соответствии с конфигурацией модуля CAPCOM[n].

В режиме сравнения запросы CAPxINTR возникают, когда значение счетчика таймера COUNT становится равным значению регистра сравнения (COUNT = CAPCOM[n].VAL).

В контроллер PLIC поступает только одна линия прерывания TMR32, являющаяся логическим ИЛИ сигналов TMRINTR, CAP0INTR - CAP3INTR.

12.6 Взаимодействие с DMA

Модуль таймера может формировать запросы к контроллеру прямого доступа к памяти DMA. Выбор события для формирования запросов к DMA производится с помощью регистра DMA_IM. Биты CAP0 – CAP3 отвечают за разрешение запросов от модулей CAPCOM, бит TMR отвечает за разрешение запросов от таймера. В контроллер DMA поступает одна линия запроса на обработку данных TMR32.

12.7 Взаимодействие с АЦП

Модуль таймера может формировать запросы на запуск секвенсоров АЦП последовательного приближения. Выбор события для формирования запросов к АЦП производится с помощью регистра ADC_IM.

13 Таймеры TMR

Микроконтроллер содержит три идентичных блока 16-разрядных таймеров TMR.

TMRx – 16-разрядный таймер/счетчик с четырьмя устройствами захвата/сравнения. Каждый таймер поддерживает по четыре модуля захвата/сравнения, выход ШИМ и выдержку временных интервалов. Обладает обширными возможностями прерываний, которые могут быть сгенерированы как при переполнении счетчика, так и от регистров захвата/сравнения. Таймеры TMR имеют внешние выводы, подключаемые через альтернативные функции портов GPIO. Так вход TMRx_EXTIN используется в качестве внешнего тактового сигнала, а сигналы TMRx_CCIA, TMRx_CCIB, TMRx_OUT0 – TMRx_OUT3 – выводы блоков CAPCOM, входящих в состав таймера.

Характеристики таймера:
- 16-разрядный таймер/счетчик с четырьмя режимами работы;
- конфигурируемый источник тактового сигнала;
- четыре конфигурируемых регистра захвата/сравнения.

Структурная схема таймера представлена на рисунке 13.1.

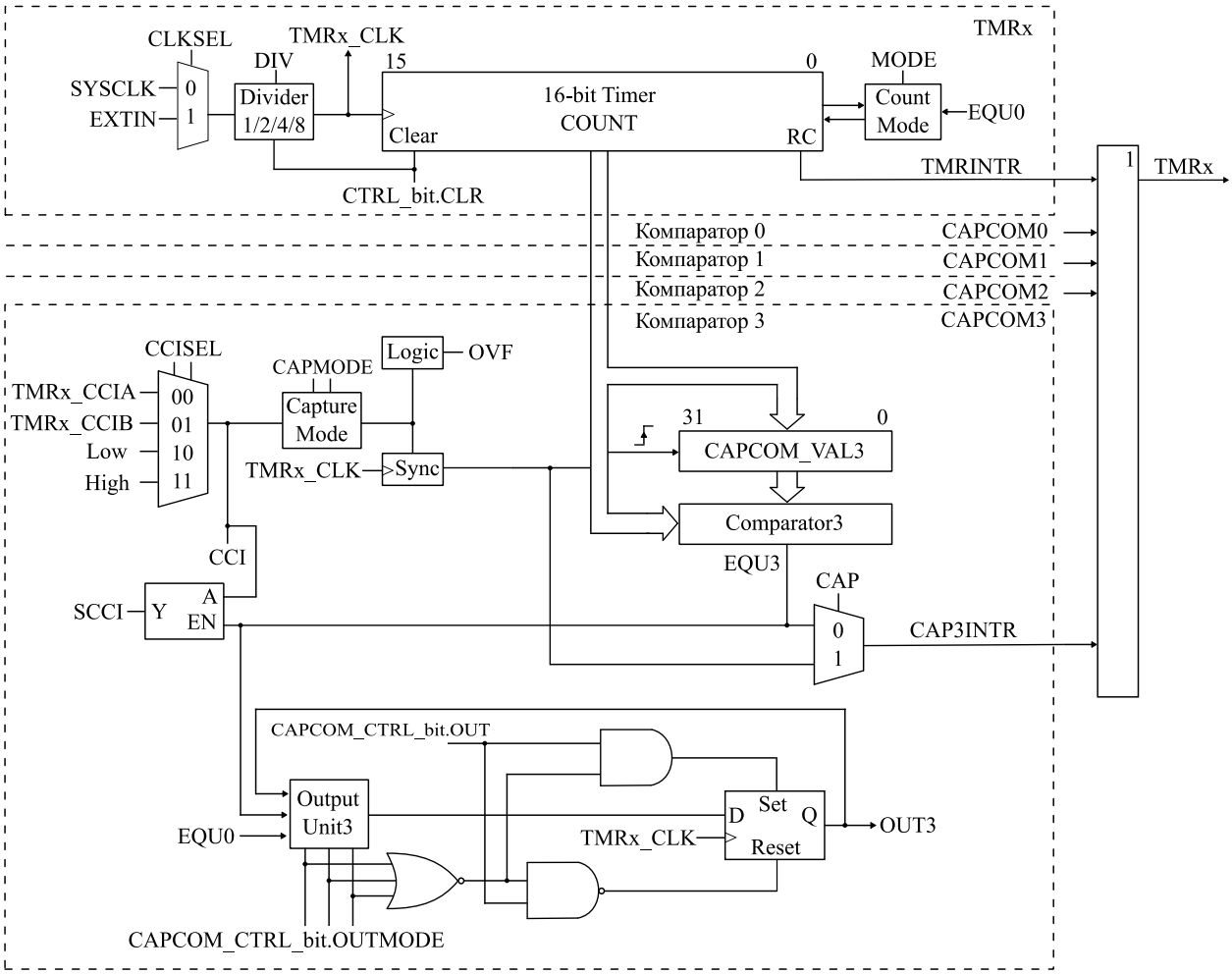


Рисунок 13.1 – Структурная схема таймера

13.1 Функционирование таймера

16-разрядный таймер-счетчик COUNT

Регистр таймера-счетчика COUNT инкрементируется или декрементируется (в зависимости от режима работы) по переднему фронту тактового сигнала. Счетчик может быть считан или записан программно. Кроме того, таймер может генерировать прерывания при переполнении. Регистр COUNT очищается установкой бита CLR. Бит CLR также очищает делитель тактовой частоты и направление счета для режима вверх/вниз.

Примечание – Рекомендуется останавливать таймер перед изменением режима его работы (за исключением разрешения прерываний, операций с флагом прерываний и битом CLR) во избежание некорректной работы. В случае если тактовый сигнал таймера асинхронен тактовому сигналу ЦПУ, любое чтение регистра COUNT должно происходить при нефункционирующем таймере, иначе результат может быть непредсказуем. В качестве альтернативы возможно считывание регистра в рабочем состоянии несколько раз и выбор корректного значения программно (наиболее частое). Любая запись в регистр COUNT производит немедленный эффект.

Выбор источника тактирования и делитель.

Тактирование таймера осуществляется тактовыми сигналом SYSCLK либо от внешнего источника через вывод TMRx_EXTIN. Источник тактирования определяется битом CLKSEL. Выбранный сигнал синхронизации передается либо напрямую, либо через делитель DIV регистра CTRL. При выставлении бита CLR делитель сбрасывается.

Запуск таймера.

Таймер запускается или перезапускается при соблюдении следующих условий:

1 Таймер считает, если в регистре CTRL значение поля MODE > 0 (режим работы) и источник тактового сигнала активен.

2 В режиме работы «вверх» или «вверх/вниз» таймер может быть остановлен путем записи 0 в регистр сравнения CAPCOM[0].VAL . Его можно запустить повторно посредством записи ненулевого значения в регистр CAPCOM[0].VAL. В этом случае таймер инкрементируется с 0.

13.2 Режимы счёта

Управление режимом таймера

Таймер имеет четыре режима работы, описанных в таблице 13.1: «стоп», «вверх», «непрерывный» и «вверх/вниз». Режимы выбираются с помощью поля MODE регистра CTRL.

Таблица 13.1 - Режимы работы таймера

MODE	Режим	Описание
00	Стоп	Остановка таймера
01	Вверх	Многократный счет от 0 до значения в регистре CAPCOM[0].VAL.
10	Непрерывный	Многократный счет от 0 до значения FFFFh.
11	Вверх/вниз	Многократный счет от 0 вверх до значения в CAPCOM[0].VAL и обратно до 0.

Режим «вверх»

Режим «вверх» используется при значении периода таймера, отличном от значения FFFFh. Таймер циклично считает до значения регистра сравнения CAPCOM[0].VAL, задающего период, как показано на рисунке 13.2. Количество отсчетов в периоде равно (CAPCOM[0].VAL + 1). Когда значение таймера становится равным значению регистра сравнения (COUNT = CAPCOM[0].VAL), таймер перезапускается, начиная счет с нуля. Если режим «вверх» выбран, когда COUNT > CAPCOM[0].VAL, таймер перезапускается немедленно.

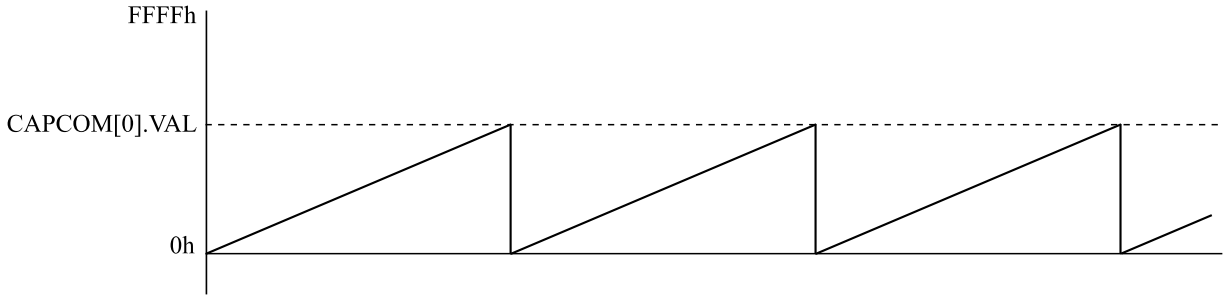


Рисунок 13.2 — Режим «вверх»

Флаг прерывания CAP0 регистра RIS устанавливается, когда значение счетчика COUNT становится равным значению регистра сравнения CAPCOM[0].VAL. Флаг прерывания TMR устанавливается, когда значение счетчика COUNT , равное значению CAPCOM[0].VAL, переключается в 0.

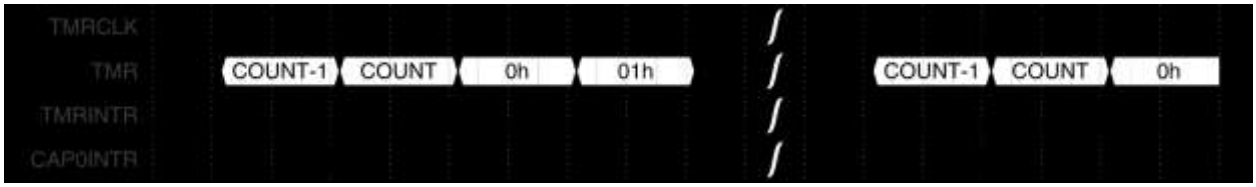


Рисунок 13.3 — Установка флагов прерываний в режиме «вверх»

Изменение регистра периода (захвата/сравнения) CAPCOM[0].VAL.

При изменении регистра CAPCOM[0].VAL во время работы таймера таймер продолжает отсчет вверх до новой границы периода, если новый период больше или равен старому или больше текущего значения счетчика. Если новый период меньше текущего значения счетчика, таймер обнуляется. Однако до обнуления счетчика может произойти один дополнительный отсчет.

Непрерывный режим

В непрерывном режиме таймер циклично считает вверх до значения FFFFh и перезапускается с 0, как показано на рисунке 13.4. Регистр захвата/сравнения CAPCOM[0].VAL работает так же, как и остальные регистры захвата/сравнения.

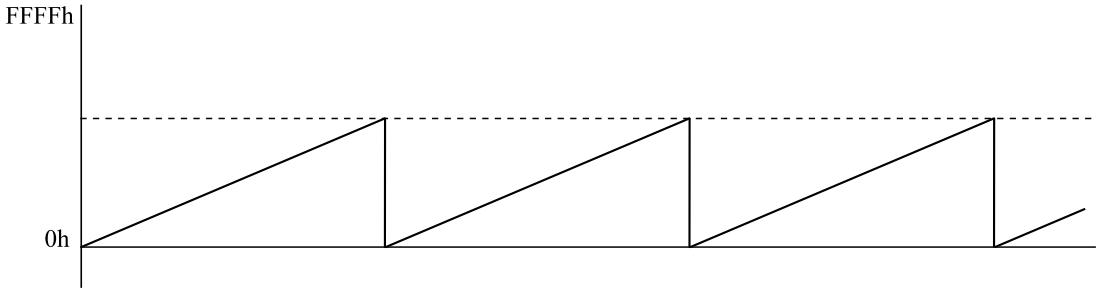


Рисунок 13.4 — Непрерывный режим

Флаг прерывания TMR устанавливается при переключении счетчика со значения FFFFh в 0. Цикл выставления флага представлен на рисунке 13.5.

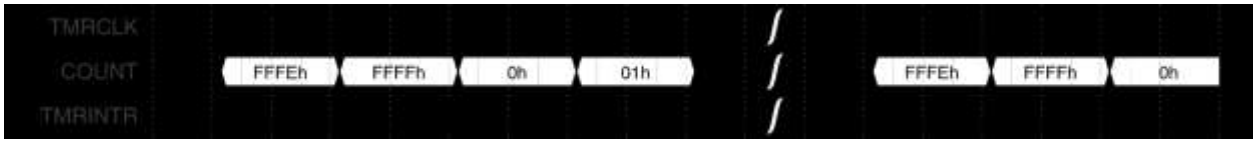


Рисунок 13.5 — Установка флага прерываний в непрерывном режиме

Использование непрерывного режима

Непрерывный режим может использоваться для генерации независимых временных интервалов и выходных частот. Каждый раз по завершении интервала генерируется прерывание. Затем временной интервал добавляется к регистру CAPCOM[n].VAL (где n - номер блока CAPCOM) в подпрограмме обработки прерывания. На рисунке 13.6 показаны два разных временных интервала t0 и t1, записываемые в регистры захвата / сравнения. В этом случае временной интервал контролируется аппаратно, а не программно, без влияния задержки обработки прерывания.

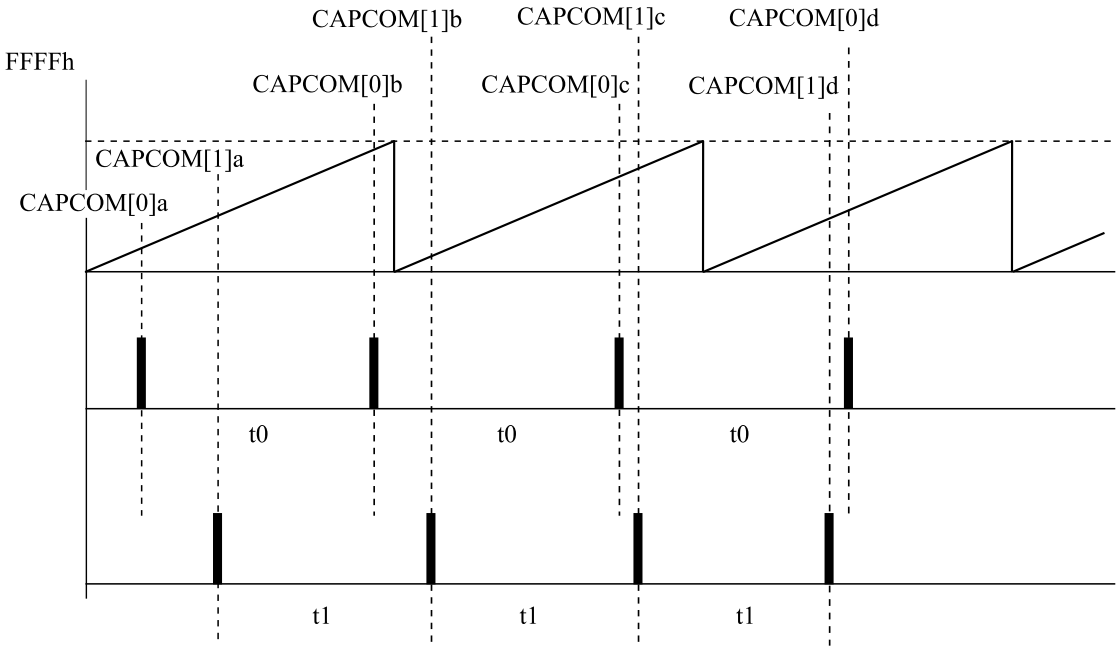


Рисунок 13.6 — Временные интервалы в непрерывном режиме

Временные интервалы могут быть сформированы и в других режимах, где в качестве регистра периода используется CAPCOM[0].VAL. Их обработка более сложная, поскольку

сумма старых данных CAPCOM[n].VAL и нового периода может быть выше значения CAPCOM[0].VAL. Если предыдущее значение CAPCOM[n].VAL плюс tx больше, чем данные CAPCOM[0].VAL, значение CAPCOM[0].VAL необходимо вычесть, чтобы получить правильный временной интервал.

Режим «вверх/вниз»

Режим «вверх/вниз» используется при необходимости генерации симметричных импульсов и периоде таймера, отличном от FFFFh. Таймер непрерывно считает до значения регистра захвата / сравнения CAPCOM[0].VAL и назад к 0, как показано на рисунке 13.7. Период равен удвоенному значению регистра CAPCOM[0].VAL.

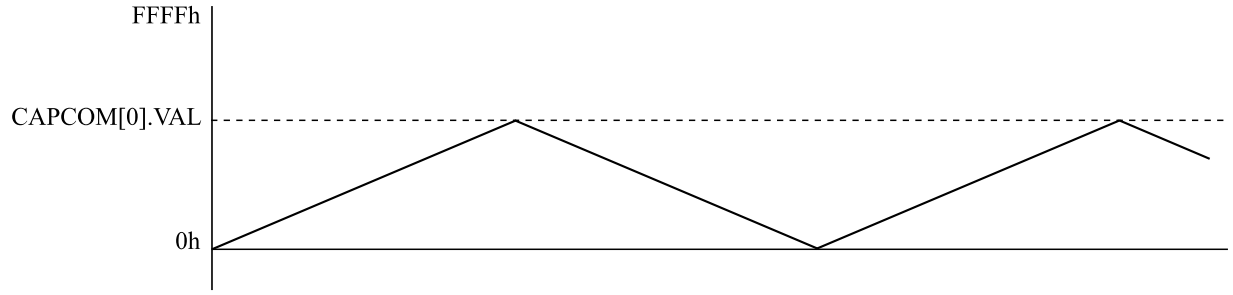


Рисунок 13.7 — Режим «вверх/вниз»

Направление счета запоминается. Это позволяет таймеру останавливаться и запускаться в том же направлении отсчета, которое было до останова. Очистить направление можно выставлением бита CLR, который очищает также значения регистра COUNT и биты делителя тактовой частоты DIV регистра CTRL.

В режиме «вверх/вниз» флаги прерываний CAPC0 и TMR устанавливаются лишь однократно за период с интервалом ½ периода. Флаг CAP0 устанавливается при переключении значения счетчика таймера COUNT из значения COUNT = CAPCOM[0].VAL – 1 в значение COUNT = CAPCOM[0].VAL. Флаг TMR устанавливается при переключении значения COUNT из 1 в 0. Цикл выставления флагов представлен на рисунке 13.8.

Рисунок 13.8 – Установка флагов прерываний в режиме «вверх/вниз»

Изменение регистра периода CAPCOM[0].VAL

При изменении CAPCOM[0].VAL во время работы таймера и отсчете в направлении вниз таймер продолжает свой спуск до тех пор, пока не достигнет нуля. Новый период вступает в силу после того, как счетчик начнет обратный отсчет до нуля.

Когда таймер ведет отсчет в направлении вверх, и новый период больше или равен старому периоду, или больше текущего значения отсчета, таймер ведет отсчет до нового периода, прежде чем начать обратный отсчет. Когда таймер ведет отсчет в направлении вверх, и новый период меньше текущего значения отсчета, таймер начинает обратный отсчет. Однако может произойти еще один подсчет, прежде чем счетчик начнет обратный отсчет.

Использование режима «Вверх/вниз»

Режим вверх/вниз необходим, когда требуется время простоя между выходными сигналами. Например, чтобы избежать перегрузки, два выхода, управляющие Н-мостом, никогда не должны находиться в высоком состоянии одновременно. В примере, показанном на рисунке 13.9, t_{dead} является:

$$t_{dead} = t_{timer} (CAPCOM[1].VAL – CAPCOM[2].VAL)$$

где t_{dead} - время, в течение которого оба выхода должны быть неактивны;

t_{timer} - время цикла работы таймера;

CAPCOM[n].VAL - содержимое регистра захвата/сравнения.

Регистры CAPCOM[n].VAL не буферизуются. Они обновляются сразу же после записи. Таким образом, любое требуемое время простоя не будет поддерживаться автоматически.

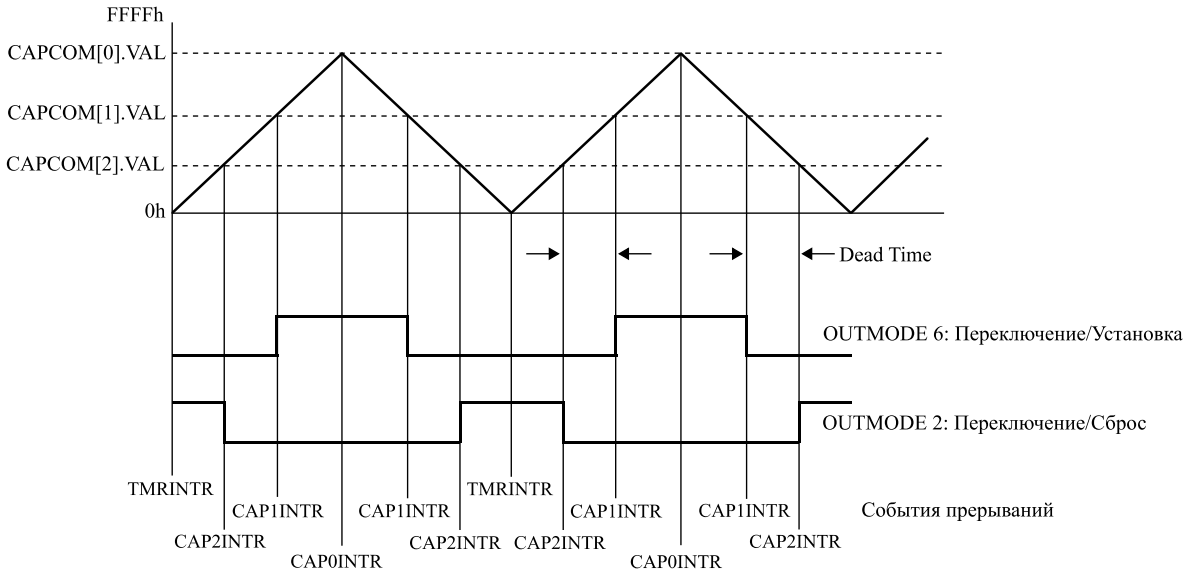


Рисунок 12.9 – Блок вывода в режиме «вверх/вниз»

13.3 Блоки захвата/сравнения

В TMR присутствуют четыре идентичных блока захвата/сравнения, CAPCOM[n].VAL. Любой из блоков может быть использован для сбора данных таймера или для генерации выходного сигнала с заданными временными интервалами.

Режим захвата

Режим захвата выбирается, при установке бита CAP = 1 в регистре CAPCOM[n].CTRL. Режим захвата используется для записи временных событий. Он может быть использован для вычисления скорости или измерения времени. Входы захвата TMRx_CCIA и TMRx_CCIB подключены к внешним выводам или внутренним сигналам (Low – уровень логического нуля, High – уровень логической единицы) и выбираются с помощью битов CCISEL. Биты CAPMODE выбирают фронт захвата входного сигнала как нарастающий, спадающий или и то, и другое вместе. Захват происходит на выбранном фронте входного сигнала.

В случае захвата:

- значение таймера копируется в регистр CAPCOM[n].VAL;
- устанавливается флаг прерывания CAPx в регистре RIS.

Уровень входного сигнала может быть считан в любое время с помощью бита CCI регистра CAPCOM[n].CTRL. Сигналы захвата TMRx_CCIA и TMRx_CCIB синхронизируются с частотой таймера. Синхронизация внешнего сигнала захвата изображена на рисунке 13.10.

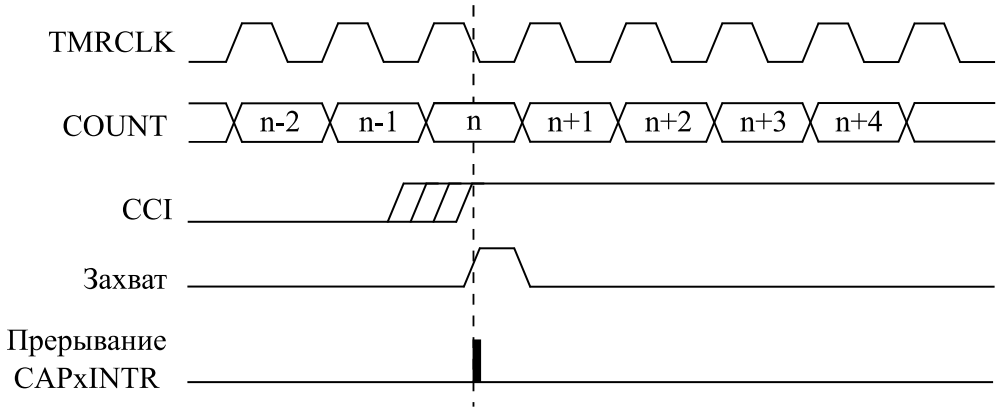


Рисунок 13.10 – синхронизация внешнего сигнала захвата

В случае захвата нового события без считывания предыдущего значения из регистра CAPCOM[n].VAL происходит перезапись регистра новым значением и устанавливается бит OVF регистра CAPCOM[n].CTRL как проиллюстрировано на рисунке 13.11. Сброс бита OVF производится программно.

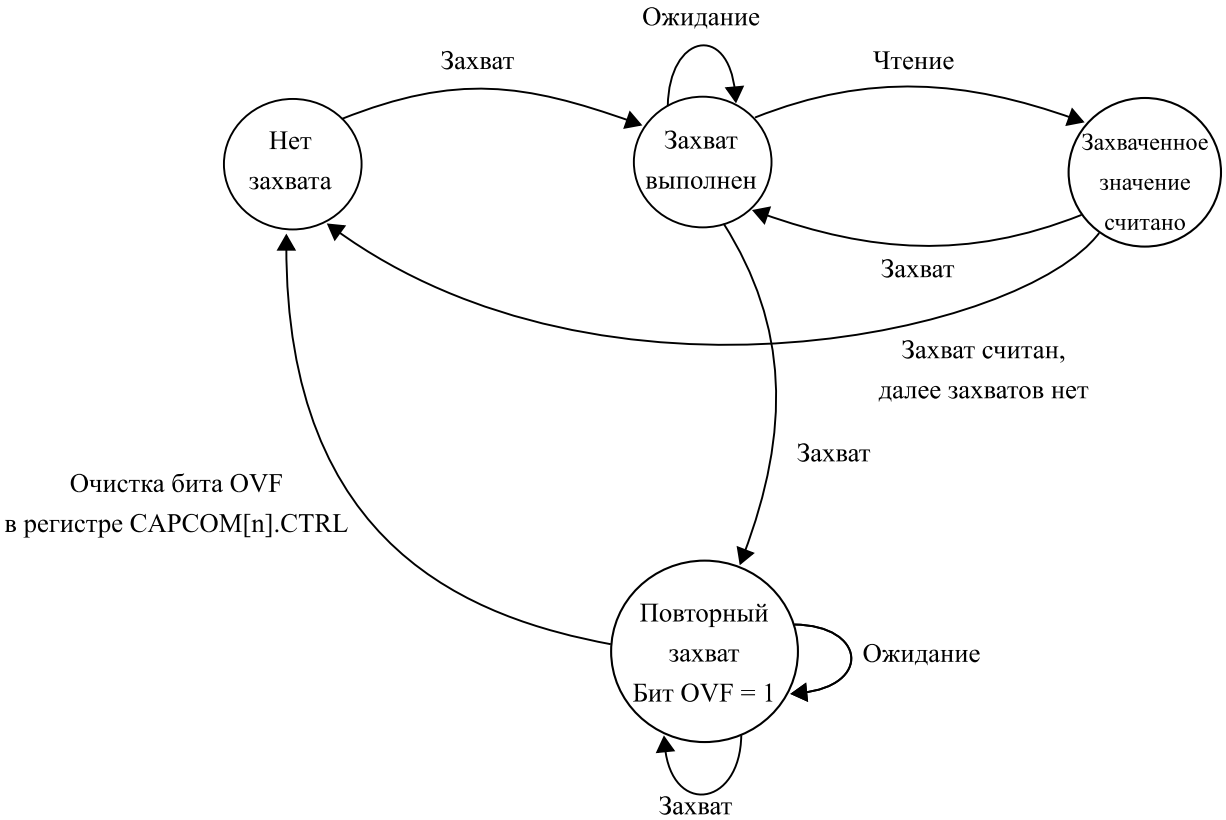


Рисунок 13.11 – Цикл захвата

Программная инициализация захвата

Захват может быть инициализирован программно. Биты CAPMODE могут быть установлены для захвата на обоих фронтах. Затем программа устанавливает CCISEL = 1 в регистре CAPCOM[1].CTRL и переключает биты CCISEL в регистре CAPCOM[0].CTRL

для переключения сигнала захвата между High и Low. Событие захвата (захват состояния внешнего сигнала) будет происходить при переключении битового поля CCISEL регистра CAPCOM[0].CTRL между значениями High и Low.

Режим сравнения

Режим сравнения выбирается при CAP = 0 в регистре CAPCOM[n].CTRL. Режим сравнения используется для генерации выходных сигналов ШИМ или прерываний через определенные промежутки времени. Когда COUNT досчитывает до значения в CAPCOM[n].VAL:

- устанавливается флаг прерывания CAPn;
- внутренний сигнал EQUn = 1;
- EQUn влияет на выход в соответствии с режимом вывода;
- входной сигнал CCI фиксируется в SCCI.

13.4 Выходной модуль

Каждый блок захвата/сравнения содержит блок вывода. Блок вывода используется для генерации выходных сигналов, таких как ШИМ-сигналы. Каждый выходной блок имеет восемь режимов работы, которые генерируют сигналы на основе сигналов EQU0 и EQUn.

Режимы вывода

Режимы вывода определяются битами OUTMODE регистра CAPCOM[n].CTRL, они описаны в таблице 13.2. Выходной сигнал изменяется по переднему фронту тактов таймера для всех режимов, кроме режима 0. Режимы вывода 2, 3, 6 и 7 непригодны для блока вывода 0, поскольку EQUn = EQU0.

Таблица 13.2 - Режимы вывода

№	OUTMODE	Режим	Описание
0	000	Вывод	Выходной сигнал TMRx_OUTn определяется битом OUT регистра CAPCOM[n].CTRL. Сигнал TMRx_OUTn обновляется немедленно при обновлении бита OUT
1	001	Установка	Выходной сигнал устанавливается, когда таймер отсчитывает значение CAPCOM[n].VAL. Он остается установленным до тех пор, пока не произойдет сброс таймера или пока не будет выбран другой режим вывода, который повлияет на выходной сигнал
2	010	Переключение/сброс	Выход переключается, когда таймер отсчитывает значение CAPCOM[n].VAL. Он сбрасывается, когда таймер отсчитывает значение CAPCOM[0].VAL
3	011	Установка/сброс	Выходной сигнал устанавливается, когда таймер отсчитывает значение CAPCOM[n].VAL. Он сбрасывается, когда таймер отсчитывает значение CAPCOM[0].VAL

Продолжение таблицы 13.2

№	OUTMODE	Режим	Описание
4	100	Переключение	Выход переключается, когда таймер отсчитывает значение CAPCOM[n].VAL. Период вывода в два раза превышает период таймера.
5	101	Сброс	Выходной сигнал сбрасывается, когда таймер отсчитывает значение CAPCOM[n].VAL. Он остается сброшенным до тех пор, пока не будет выбран другой режим вывода, который повлияет на выходной сигнал.
6	110	Переключение/ установка	Выход переключается, когда таймер отсчитывает значение CAPCOM[n].VAL. Он устанавливается, когда таймер отсчитывает значение CAPCOM[0].VAL.
7	111	Сброс/установка	Выходной сигнал сбрасывается, когда таймер отсчитывает значение CAPCOM[n].VAL. Он устанавливается, когда таймер отсчитывает значение CAPCOM[0].VAL.
Примечание – n – номер блока CAPCOM.			

Пример вывода - таймер в режиме счета вверх

Сигнал TMRx_OUTn изменяется, когда таймер досчитывает до значения CAPCOM[n].VAL, и падает в ноль при CAPCOM[0].VAL, в зависимости от режима вывода. Пример показан на рисунке 13.12 с использованием CAPCOM[0].VAL и CAPCOM[1].VAL.

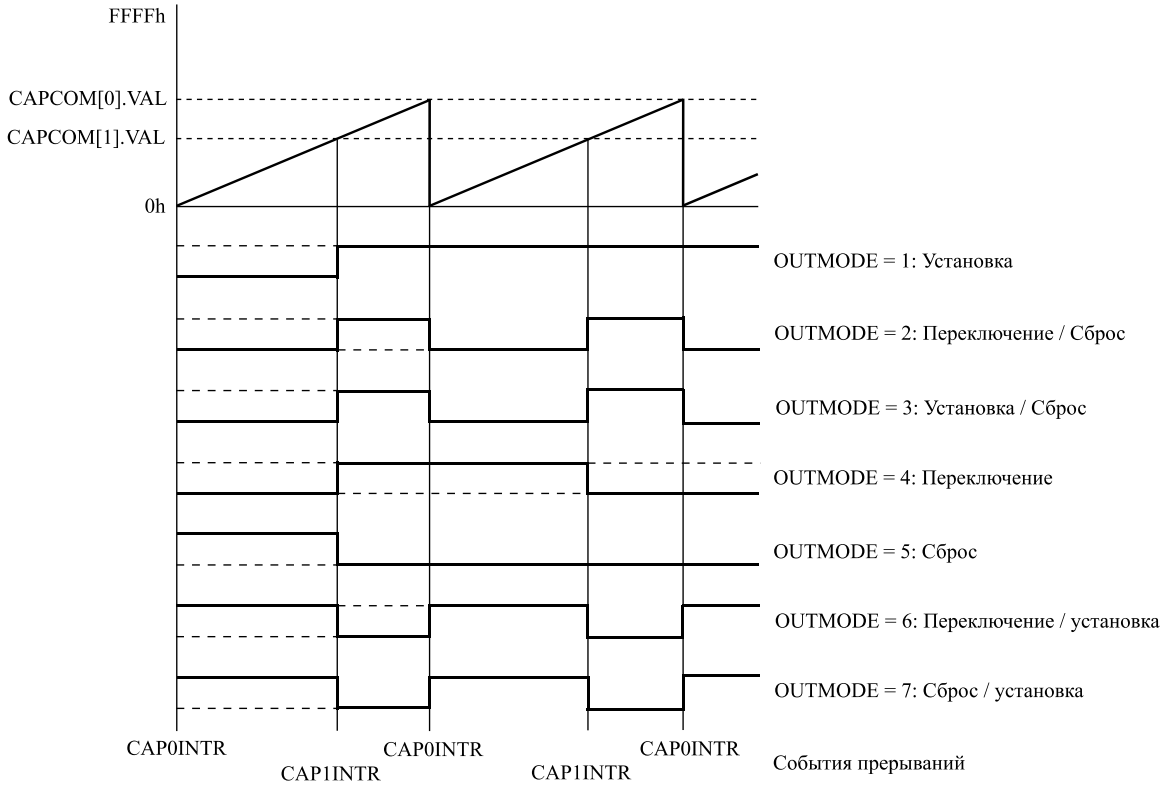


Рисунок 13.12 – Таймер в режиме счета вверх (пример вывода)

Пример вывода - Таймер в непрерывном режиме

Сигнал TMRx_OUTn изменяется, когда таймер достигает значений CAPCOM[n].VAL и CAPCOM[0].VAL, в зависимости от режима вывода. Пример показан на рисунке 13.13 с использованием CAPCOM[0].VAL и CAPCOM[1].VAL.

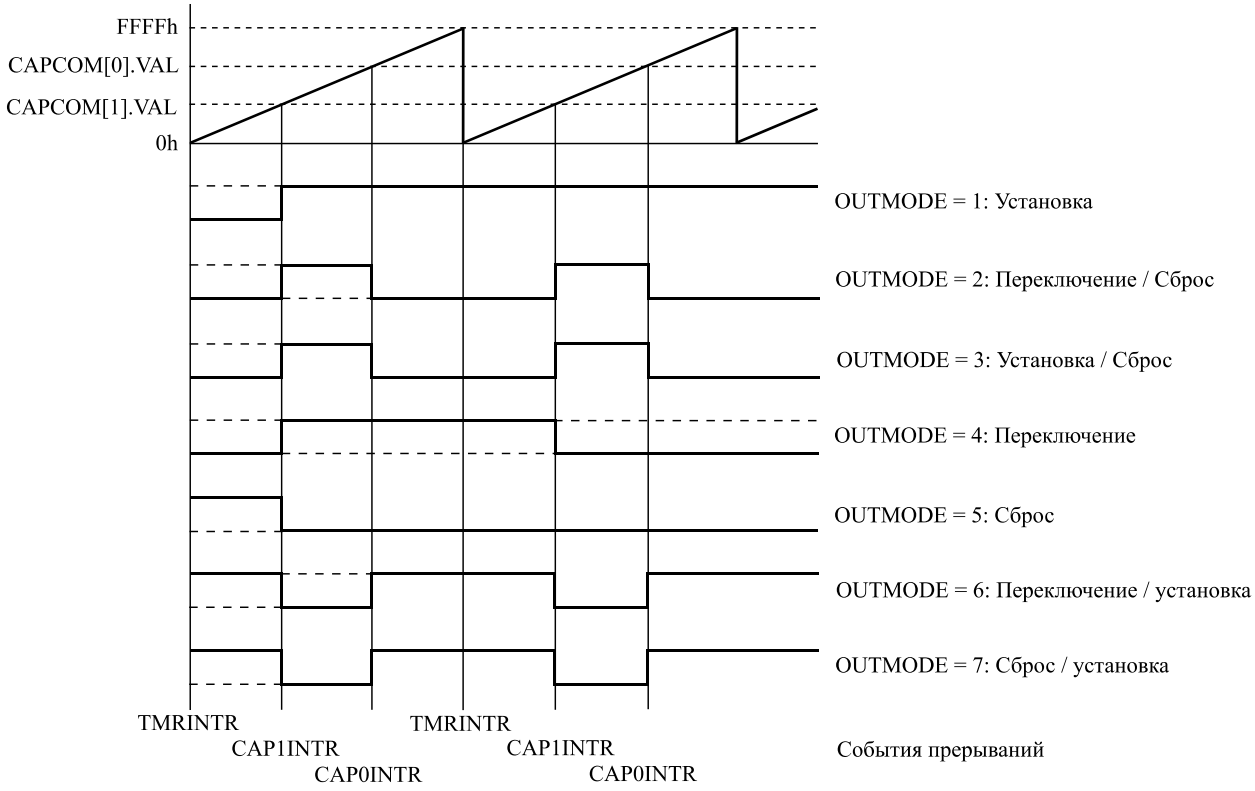


Рисунок 13.13 – Таймер в непрерывном режиме (пример вывода)

Пример вывода - Таймер в режиме счета «вверх/вниз»

Сигнал TMRx_OUTn изменяется, когда таймер равен CAPCOM[n].VAL в любом направлении счета и когда таймер равен CAPCOM[0].VAL, в зависимости от режима вывода. Пример показан на рисунке 13.14 с использованием CAPCOM[0].VAL и CAPCOM[2].VAL.

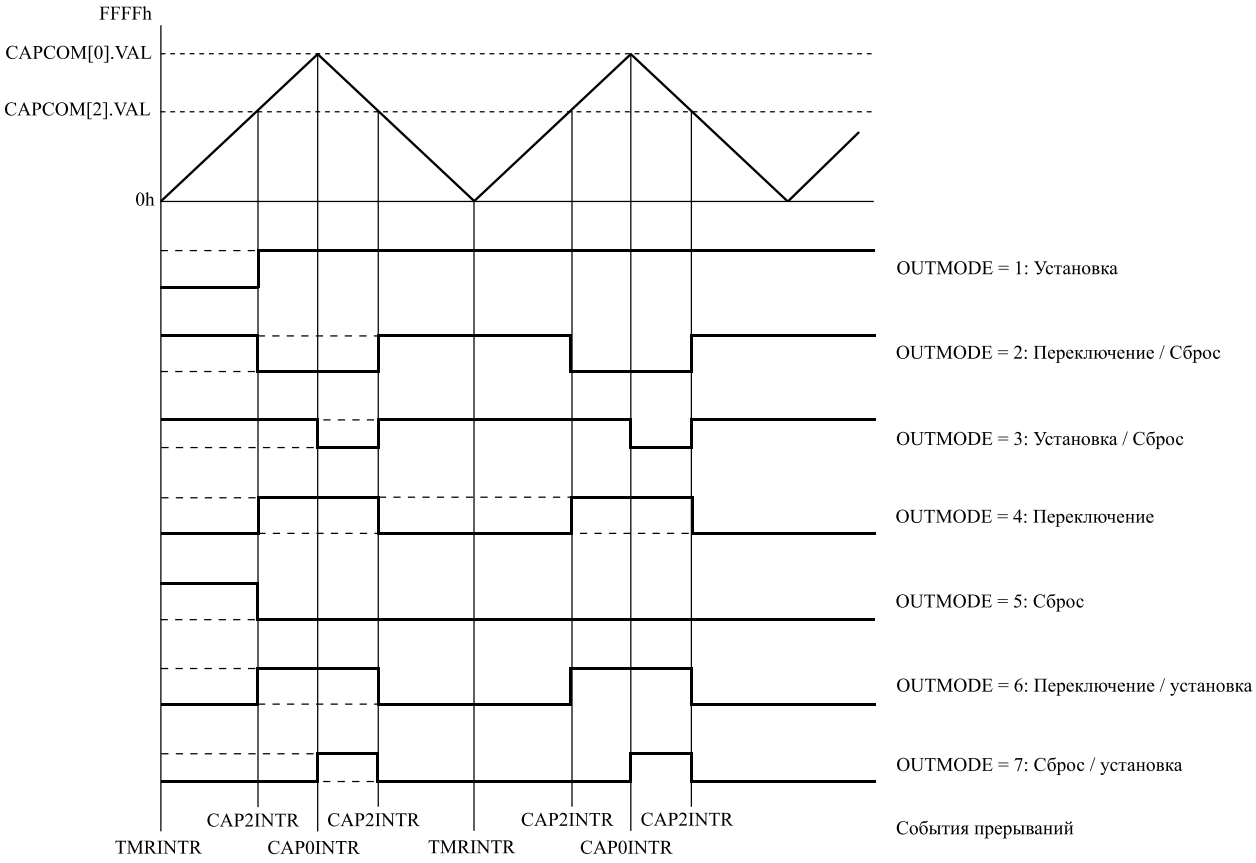


Рисунок 13.14 – Таймер в режиме счета «вверх/вниз» (пример вывода)

Переключение между режимами вывода

При переключении между режимами вывода один из битов OUTMODE регистра CAPCOM[n].CTRL должен оставаться установленным во время перехода, кроме переключения в режим 0. В противном случае может произойти сбой вывода, поскольку элемент NOR декодирует режим вывода 0. Безопасным методом переключения между режимами вывода является использование режима вывода 7 в качестве переходного состояния.

13.5 Прерывания таймера

В модуле таймера предусмотрено пять маскируемых источников прерываний.

Сигналы запросов на прерывания:

- TMRINTR – сигнал прерывания от таймера;
- CAP0INTR – от 0 модуля CAPCOM;
- CAP1INTR – от 1 модуля CAPCOM;
- CAP2INTR – от 2 модуля CAPCOM;
- CAP3INTR – от 3 модуля CAPCOM.

Каждый из сигналов может быть маскирован путем установки соответствующего бита в регистре маски IM.

Источник прерывания также можно определить, считав состояние регистра RIS или регистра MIS (маскированные прерывания). Сброс прерывания осуществляется программно путём установки соответствующего бита в регистре IC.

Запрос TMRINTR возникает, когда значение счетчика таймера COUNT принимает значение равное нулю. Периодичность запросов равна периоду счета таймера.

В режиме захвата запросы CAPxINTR возникают при наступлении события захвата внешнего сигнала в соответствии с конфигурацией модуля CAPCOM[n].

В режиме сравнения запросы CAPxINTR возникают, когда значение счетчика таймера COUNT становится равным значению регистра сравнения (COUNT = CAPCOM[n].VAL).

В контроллер PLIC поступает только одна линия прерывания TMR32, являющаяся логическим ИЛИ сигналов TMRINTR, CAP0INTR - CAP3INTR.

13.6 Взаимодействие с DMA

Модуль таймера может формировать запросы к контроллеру прямого доступа к памяти DMA. Выбор события для формирования запросов к DMA производится с помощью регистра DMA_IM. Биты CAP0 – CAP3 отвечают за разрешение запросов от модулей CAPCOM, бит TMR отвечает за разрешение запросов от таймера. В контроллер DMA поступает одна линия запроса на обработку данных TMRx.

13.7 Взаимодействие с АЦП

Модуль таймера может формировать запросы на запуск секвенсоров АЦП последовательного приближения. Выбор события для формирования запросов к АЦП производится с помощью регистра ADC_IM.

14 Блок подсчёта циклического избыточного кода (CRC)

Микроконтроллер содержит два идентичных блока подсчёта циклического избыточного кода CRC0 и CRC1.

Блок проверки подсчётом циклически избыточным кодом (CRC) используется для получения CRC из 8-, 16-, 32-битных слов данных и генератора полиномов.

Среди других применений, методы, основанные на подсчёте CRC, используются для определения корректности переданных данных или целостности хранимых данных. Блок вычисления CRC помогает посчитать контрольную сумму данных во время выполнения, для сравнения с эталонной суммой, созданной ранее и сохранённой в памяти.

14.1 Особенности блока CRC

Блок CRC имеет следующие особенности:

- программируемый генератор полиномов (7-, 8-, 16-, 32-бит);
- использует полином CRC32-IEEE-802.3 (0x04C1_1DB7) по умолчанию;
- оперирует с 8-, 16-, 32-битными входными данными;
- программируемое начальное значение CRC;
- один 32-битный регистр данных;
- вычисление CRC осуществляется за четыре такта HCLK для входных 32-битных данных;
- возможность обратимости входных/выходных данных для поддержки различных порядков байт данных, опциональный XOR для выходных данных.

14.2 Функции блока CRC

Структура блока CRC

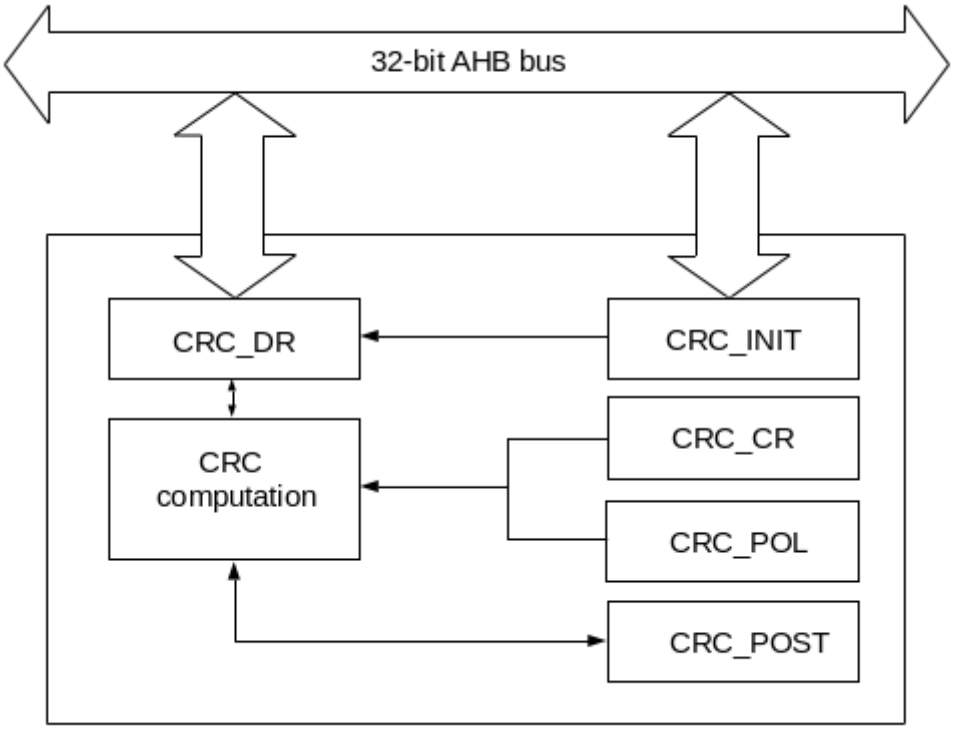


Рисунок 14.1 – Структура блока проверки подсчётом циклически избыточным кодом

Описание блока CRC

Аппаратный блок подсчёта CRC имеет один 32-разрядный регистр данных (DR) доступный для чтения и записи. Он используется для ввода новых входных данных при записи в регистр, и содержит результат предыдущего подсчёта CRC при чтении.

Каждая операция записи в регистр данных создаёт комбинацию предыдущего значения CRC (содержащегося в регистре DR) и нового. Подсчёт CRC выполняется над всеми 32-битными словами данных, полусловами или байтами, в зависимости от формата записываемых данных.

Доступ к регистру DR может быть по словам (32 бита), выровненным справа полусловам (16 бит) и выровненным справа байтам. Другие регистры блока CRC поддерживают любой доступ. Длительность вычисления зависит от ширины входных данных:

- 4 такта AHB для 32-битных входных данных;
- 2 такта AHB для 16-битных входных данных;
- 1 такт AHB для 8-битных входных данных.

Размер данных может быть динамически скорректирован для уменьшения количества операций записи в регистр данных DR в соответствии с необходимым количеством байт. Например, CRC для 5 байт исходных данных может быть подсчитана записью слова, а затем записью байта.

После завершения подсчёта CRC, результат помещается в регистр DR. Если используется пост-обработка (установлены биты XOROUT и/или REV_OUT в регистре CR), тогда результат вычисления CRC помещается в регистр DR, а результат пост-обработки помещается в регистр POST.

Порядок следования бит внутри каждого байта, полуслова или слова целиком входных данных может быть изменён в соответствии со значением, записанным в битовое поле REV_IN[1:0] регистра CR.

Например, входные данные 0x1A2B3C4D для подсчёта CRC преобразуются в:

- 0x58D43CB2 с побайтовым обращением;
- 0xD458B23C с обращением по два байта;
- 0xB23CD458 с обращением по четыре байта.

Выходные данные могут быть также обращены установкой бита REV_OUT в регистре CR. В этом случае, операция обращения исполняется на битовом уровне: к примеру, выходные данные 0x11223344 преобразуются в 0x22CC4488.

Бит XOR_OUT в регистре CR указывает, что результат будет подвергнут операции «XOR» (исключающее ИЛИ) со всеми нулями или всеми единицами. Операция «XOR» производится над результатом после обращения выходных данных, если был установлен соответствующий бит в регистре CR, к примеру: результат из предыдущего примера 0x22CC4488 преобразуется в 0xDD33BB77.

Бит MODE в регистре CR устанавливает режим работы блока подсчёта CRC. При установке его в «1», входные данные после операции изменения порядка следования байтов (если была выбрана соответствующая опция) создают комбинацию (операция XOR – «Исключающее ИЛИ») с предыдущим содержимым регистра данных.

Таблица 14.1 – Влияние бита MODE на режим работы блока CRC

Входные данные	Регистр данных после подсчёта	
	бит MODE сброшен	Бит MODE установлен
Первые: 0x0000 0000	0xc704 dd7b	0xc704 dd7b
Вторые: 0x1111 1111	0x7ab7 9290	0xbdb3 4feb

Параметры для расчета CRC указанные в таблице 14.1: REV_IN = 0x0, REV_OUT = = 0x0, XOR_OUT = 0x0, стандартные значения регистров порождающего полинома и инициализации. Запись данных производится словами (32-бит).

В случае MODE, установленном в «1», вторые данные для подсчета используются 0xc704 dd7b (предыдущее содержимое регистра данных) ^ 0x1111_1111(новые входные данные) и равны 0xd615 cc6a.

Блок подсчёта CRC может быть инициализирован запрограммированным значением используя управляющий бит RESET в регистре CR (стандартное значение 0xFFFFFFFF).

Начальное значение для алгоритма CRC может быть запрограммировано в регистре INIT. Регистр DR автоматически инициализируется при записи в регистр INIT.

Программирование порождающего полинома

Коэффициенты порождающего полинома задаются в регистре POL, а их размер может быть 7, 8, 16 или 32-битным в зависимости от поля POLY_SIZE[1:0] в регистре CR. Чётные порождающие полиномы не поддерживаются.

Примечание – Чётный порождающий полином имеет коэффициенты - X+X2+..+Xn, тогда как нечётный - 1+X+X2+..+Xn.

Если разрядность данных CRC меньше, чем 32-бит, значение может быть прочитано из младших разрядов регистра DR.

Для получения корректного значения CRC, во время вычисления не допускается изменение коэффициентов порождающего полинома или его размера. В результате, если вычисление CRC продолжается, необходимо выполнить сброс или прочитать регистр DR до изменения порождающего полинома. Начальное значение порождающего полинома CRC-32 (Ethernet) 0x4C11DB7.

15 Хеш-процессор (HASH)

15.1 Функционирование хеш-процессора

Хеш-процессор представляет собой полный набор реализаций алгоритмов безопасного хеширования (SHA-1, SHA-224, SHA-256), хеш-алгоритм MD5 (алгоритм хеш-суммы 5) и HMAC (хеш-код аутентификации сообщений) алгоритм, подходящий для различных приложений. Он вычисляет хеш-сумму (160 бит для алгоритма SHA-1, 224 бита для алгоритма SHA-224, 256 бит для алгоритма SHA-256 и 128 бит для алгоритма MD5) для сообщений длинной до 264 – 1 бит, когда как алгоритм HMAC обеспечивает способ аутентификации сообщений с использованием хеш функций. Алгоритмы HMAC заключаются в вызове SHA-1, SHA-224, SHA-256 или MD5 хеш функций дважды.

15.2 Основные особенности

Основные особенности хеш процессора:

- может применяться в системах аутентификации данных;
- быстрое вычисление алгоритмов SHA-1, SHA-224 и SHA-256, и MD5;
- 32-битные слова данных для входных данных, поддержка словного, полусловного, байтового и битовой строки с порядком от младшего к старшему;
- поддержка автоматического изменения порядка следования бит;
- автоматическая процедура заполнения: вставки ведущей единицы, длины сообщения до размера 512 (16 * 32) бит при формировании блока;
- автоматическая работа с потоком данных для прямого доступа к памяти.

Примечание – Автоматическое заполнение данных согласно определению в спецификациях к алгоритмам SHA-1, SHA-224 и SHA-256, состоит из добавления дополнительного бита «1» за которым следуют N-бит «0» для получения общей длины, равной 448 по модулю 512. После этого, сообщение завершается 64-битным целочисленным значением, которое является двоичным представлением длинны оригинального сообщения. Для хеш-процессора, передаваемые данные являются 32-битными словами, таким образом нужно предоставить дополнительную информацию в конце записи сообщения, а именно количество допустимых битов в последнем записанном 32-битном слове.

15.3 Описание хеш процессора

Блок-схема хеш процессора представлена на рисунке 15.1.

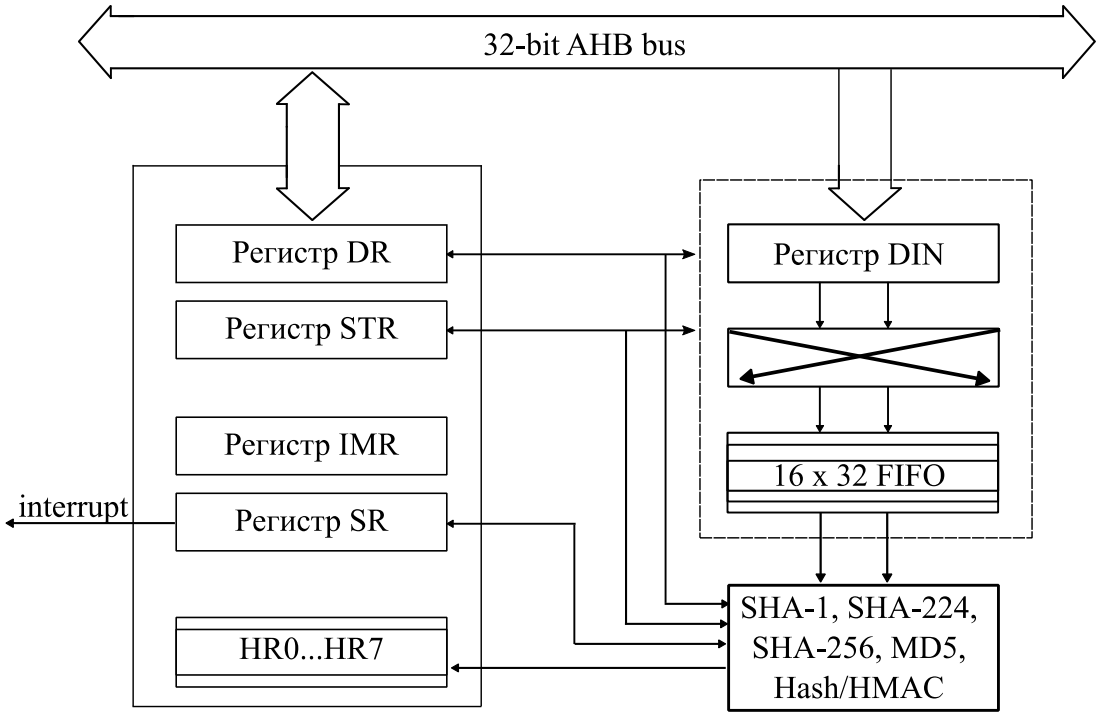


Рисунок 15.1 - Блок схема хеш-процессора

При передаче сообщения длинной до 264 бита на вход хеш-процессора, алгоритмы SHA-1, SHA-224 и SHA-256 и MD5 вычисляют 160-битную, 224-битную, 256-битную и 128-битную выходную строку, соответственно, называемую хеш-суммой сообщения. После этого хеш-сумму сообщения можно обработать с помощью алгоритма цифровой подписи, чтобы сгенерировать или проверить подпись для сообщения. Подписание хеш-суммы сообщения, а не самого сообщения, часто повышает эффективность процесса, поскольку хеш-сумма сообщения обычно намного меньше по размеру, чем само сообщение. При проверке цифровой подписи необходимо использовать тот же хеш-алгоритм, что и при формировании цифровой подписи.

Алгоритмы SHA-1, SHA-224, SHA-256 и MD5 квалифицируются как «безопасные», потому что с точки зрения вычислений невозможно найти сообщение, соответствующее данной хеш-сумме сообщения, или найти два разных сообщения, которые создают одну и ту же хеш-сумму сообщения. Любое изменение сообщения при передаче с очень высокой вероятностью приведет к вычислению другой хеш-суммы сообщения, и подпись не сможет быть проверена.

Сообщение или массив данных для обработки хеш-процессором следует воспринимать как битовую строку. Длинна этого сообщения определяется количеством битов в этом сообщении (пустое сообщение имеет длину 0). Так, 32-бита этой битовой строки формируют 32-битное слово.

Примечание – Битовая строка растёт слева направо, и биты могут группироваться как байты (8-бит) или слова (32-бит).

15.4 Длительность обработки

Время вычисления промежуточного блока сообщения занимает:

- 66 циклов тактового сигнала HCLK для алгоритма SHA-1;
- 50 циклов тактового сигнала HCLK для алгоритма SHA-224;
- 50 циклов тактового сигнала HCLK для алгоритма SHA-256;
- 50 циклов тактового сигнала HCLK для алгоритма MD5.

Помимо времени вычисления необходимо добавить время, необходимое для загрузки 16 слов блока в процессор (по крайней мере 16 циклов тактового сигнала для блока в 512-бит).

Время, необходимое для обработки последнего блока сообщения (или ключа в HMAC) может быть длиннее. Время зависит от длинны последнего блока и размера ключа (в режиме HMAC). В сравнении с обработкой промежуточного блока, оно может быть увеличено в:

- от 1 до 2,5 раз для хеш-функции;
- около 2,5 раз для входного ключа HMAC;
- от 1 до 2,5 раз для сообщения HMAC;
- около 2,5 раз для внутреннего ключа HMAC в случае короткого ключа, либо если используются одинаковые внутренний и внешний длинные ключи;
- от 3,5 до 5 для внешнего ключа HMAC в случае разных внутренних и внешних длинных ключей.

15.5 Типы данных

Данные передаются в хеш-процессор по 32 бита (слова) с помощью записи в регистр DIN. При этом, исходная битовая строка может быть организована в байты, полуслова или слова, или даже представлена в виде битов. Так как организация системной памяти имеет обратный порядок байтов, а вычисления SHA1, SHA-224 и SHA-256 выполняются с прямым порядком байтов, в зависимости от того, как сгруппирована исходная битовая строка, операция замены битов, байтов или полуслов выполняется автоматически хеш-процессором.

Тип обрабатываемых данных настраивается битовым полем DATATYPE в управляющем регистре CR:

- при представлении входных данных как битовой строки, все биты следует поменять местами (записать значение 0x3 в битовое поле DATATYPE), как показано на рисунке 15.2;
- при представлении входных данных как байтов, байты следует поменять местами (записать значение 0x2 в битовое поле DATATYPE), как показано на рисунке 15.3;
- при представлении входных данных как полуслово, то полуслова следует поменять местами (записать значение 0x1 в битовое поле DATATYPE), как показано на рисунке 15.4.

Младший значащий бит сообщения должен быть в позиции 0 (справа) в первом слове, введенном в хеш-процессор, 32-й бит битовой строки должен быть в позиции 0 во втором слове, введенном в хеш-процессор, и т.д.

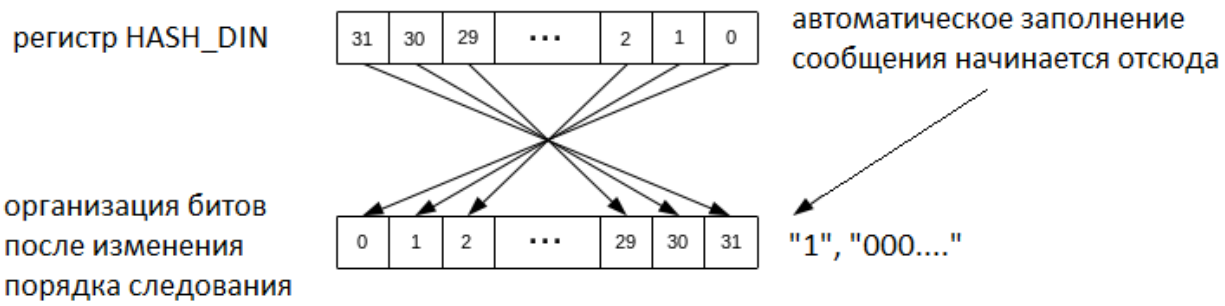


Рисунок 15.2 – Представление входных данных как битовой строки

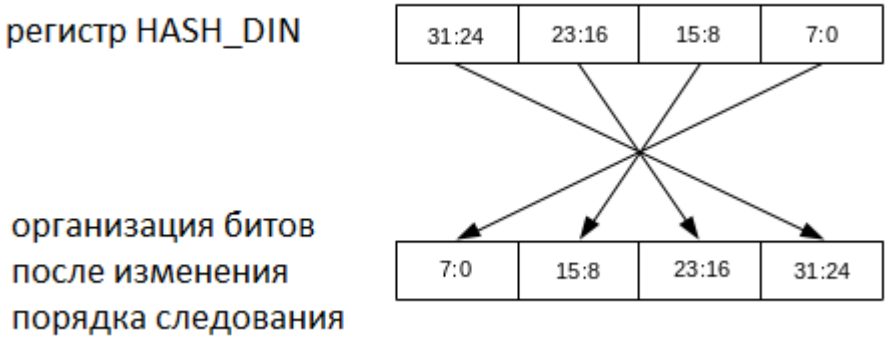


Рисунок 15.3 – Представление входных данных как байтов

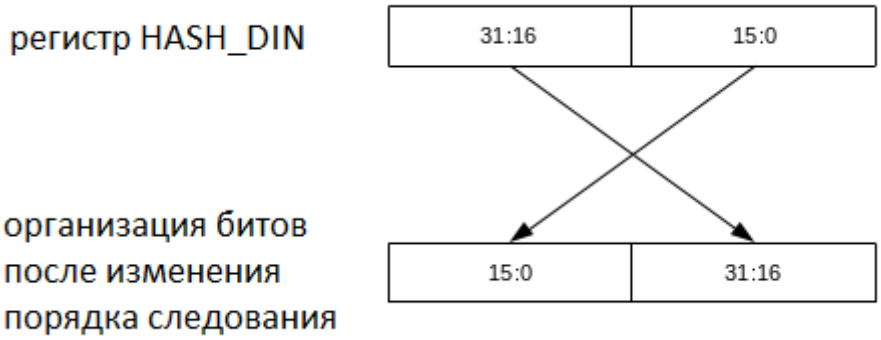


Рисунок 15.4 – Представление входных данных как полуслов

15.6 Вычисление хеш-суммы сообщения

Хеш-процессор последовательно обрабатывает блоки по 512 бит при вычислении хеш-суммы сообщения. Таким образом, каждый раз, когда DMA или ЦП записывают 16 × 32-битных слов (512 бит) в хеш-процессор, хеш-функция автоматически начинает вычислять хеш-сумму сообщения. Данная операция - частичное вычисление хеш-суммы.

Сообщение для обработки вводится в периферийное устройство 32-битными словами, записанными в регистр DIN. Текущее содержимое регистра DIN передается во входной буфер FIFO (IN FIFO) каждый раз, когда в регистр записываются новые данные. Регистр DIN и входной буфер FIFO образуют входной буфер длиной 17 слов.

Обработка блока может начаться только после того, как последнее значение блока войдет во входной буфер. Периферийное устройство должно получить информацию о том,

содержит ли регистр HASH_DIN последние биты сообщения или нет. Возможны два случая:

Прямой доступ к памяти не используется:

- вычисление частичной хеш-суммы сообщения выполняется путем записи слов блока сообщения в регистр DIN. Программное обеспечение должно дождаться, пока процессор снова будет готов (когда DINIS=1), прежде чем записывать новые данные следующего блока в DIN;

- вычисление окончательной хеш-суммы сообщения (последний введенный блок) делается путем записи бита DCAL в 1.

При использовании прямого доступа к памяти: содержимое регистра DIN автоматически определяется информацией, отправленной контроллером DMA.

- в случае одиночной передачи DMA: когда последний блок был передан в регистр DIN по каналу DMA, бит DCAL регистра STR будет автоматически установлен в 1 в регистре STR, чтобы запустить окончательный расчет хеш-суммы сообщения;

- в случае множественной передачи DMA: Бит множественной передачи DMA (MDMAT) должен быть установлен в 1 программным обеспечением, чтобы бит DCAL не устанавливался автоматически аппаратно, в этом случае окончательный расчет хеш-суммы сообщения для хеша и для каждой фазы алгоритма HMAC (для получения более подробной информации об этапах HMAC см. раздел операций HMAC) не будет запущен в конце запроса на передачу DMA, что позволит процессору переконфигурировать DMA. Во время последней передачи DMA бит множественной передачи DMA (MDMAT) может быть очищен программным обеспечением, чтобы автоматически установить бит DCAL в конце последнего блока и запустить окончательное вычисление хеш-суммы сообщения.

Процесс ввода данных и вычисление частичной хеш-суммы - продолжается до тех пор, пока последние биты исходного сообщения не будут записаны в регистр DIN. Поскольку длина (количество битов) сообщения может быть любым целым числом, последнее слово, записанное в хеш-процессор, может иметь допустимое количество битов от 1 до 32. Это количество допустимых битов в последнем слове (NBLW) должно быть записано в регистр STR, чтобы правильно выполнить заполнение сообщения перед окончательным вычислением хеш-суммы сообщения.

Как только это будет сделано, запись в STR с битом DCAL = 1 запускает обработку последнего введенного блока сообщения хеш-процессором. Эта обработка заключается в:

- автоматическое выполнение операции заполнения сообщения: цель этой операции – сделать общую длину дополненного сообщения кратной 512. Хеш-процессор последовательно обрабатывает блоки по 512 бит при вычислении хеш-суммы сообщения;

- вычисление окончательной хеш-суммы сообщения.

Когда DMA включен, он предоставляет информацию хеш-процессору при передаче последнего слова данных. Затем заполнение и вычисление хеш-суммы выполняются автоматически, как и при установке бита DCAL.

15.7 Автоматическое заполнение сообщения

Заполнение сообщения заключается в добавлении «1», за которой следуют m «0», за которым следует 64-битное целое число до конца исходного сообщения для создания дополненного блока сообщения длиной 512. «1» добавляется к последнему слову, записанному в регистр HASH_DIN в битовую позицию, определяемой битовым полем NBLW регистра HASH_STR, а оставшиеся старшие биты очищаются («0»).

Пример − Предположим, что исходное сообщение представляет собой двоично-кодированную форму ASCII «abc» длиной L = 24:

```
  байт 0   байт1    байт 2   байт 3
 01100001 01100010 01100011 UUUUUUUU
<-- первое слово записанное в HASH_DIN -->
```

Поле NBLW должно быть загружено со значением 24: «1» добавляется к битовой позиции 24 в битовой строке (начиная отсчета слева направо в приведенной выше битовой строке), что соответствует биту 31 в регистре HASH_DIN (порядок следования байтов от младшего к старшему):

```
01100001 01100010 01100011 1UUUUUUU
```

Поскольку L = 24, количество битов в приведенной выше битовой строке равно 25, и к ним добавляются 423 «0», что дает теперь 448. Это дает (в шестнадцатеричном формате с порядком следования битов от старшего к младшему):

```
61626380 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000
```

Значение L в двухсловном представлении (то есть 00000000 00000018) добавляется. Отсюда окончательное заполненное сообщение в шестнадцатеричном формате:

```
61626380 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000028
```

Если хеш-процессор запрограммирован на использование порядка следования байтов от младшего к старшему, указанное выше сообщение необходимо ввести, выполнив следующие шаги:

1 В регистр HASH_DIN записывается 0xUU636261 (где «U» означает «все равно»).

2 В регистр HASH_STR записывается 0x18 (количество допустимых битов в последнем слове, записанном в регистр HASH_DIN, равно 24, так как исходная длина сообщения составляет 24 бита).

3 0x10 записывается в регистр HASH_STR, чтобы начать заполнение сообщения и вычисляется хеш-сумма. Когда NBLW ≠ 0x00, заполнение сообщения помещает «1» в регистр HASH_DIN в битовой позиции, определяемой значением NBLW, и вставляет «0» в битовые позиции [31:(NBLW+1)]. Когда NBLW == 0x00, заполнение сообщения вставляет одно новое слово со значением 0x0000 0001. Затем добавляется полное нулевое слово (0x0000 0000) и длина сообщения в представлении из двух слов, чтобы получить блок размером 16 x 32 бит. слова.

4 Выполняется вычисление хеш-суммы, после чего хеш-сумма сообщения становится доступной в регистрах HASH_Hx. Например, для алгоритма SHA-1:

```
H0 = 0xA9993E36
H1 = 0x4706816A
H2 = 0xBA3E2571
H3 = 0x7850C26C
H4 = 0x9CD0D89D
```

15.8 Вычисление хеш-функции

Хеш-функция (SHA-1, SHA-224, SHA-256 и MD5) выбирается, когда бит INIT записывается в «1» в регистре HASH_CR, а бит MODE в «0» в HASH_CR. Алгоритм (

SHA-1, SHA-224, SHA-256 или MD5) выбирается одновременно (то есть, когда бит INIT установлен) с использованием битов ALGO.

Затем сообщение может начинать обрабатываться, записав его пословно в регистр HASH_DIN. Когда блок из 512 бит, то есть 16 слов, был записан, вычисление частичной хеш-суммы сообщения начинается с записи первых данных следующего блока. Хеш-процессор остается занятым в течение 66 тактов для алгоритма SHA-1 или 50 тактов для алгоритма MD5, алгоритма SHA-224 и алгоритма SHA-256.

Затем процесс можно повторять до последнего слова сообщения. Если используются передачи DMA, обратитесь к разделу «Процедура загрузки данных по DMA». В противном случае, если длина сообщения не кратна 512 битам, то для запуска вычисления окончательной хеш-суммы необходимо записать в регистр HASH_STR.

После вычисления хеш-сумма может быть прочитана из регистров HASH_H0...HASH_H7, где:

- HASH_H4..HASH_H7 не имеют значения, когда выбран алгоритм MD5;
- HASH_H5..HASH_H7 не имеют значения, когда выбран алгоритм SHA-1;
- HASH_H7 не имеет значения, если выбран алгоритм SHA-224.

15.9 Вычисление HMAC

Алгоритм HMAC используется для аутентификации сообщений путем необратимой привязки обрабатываемого сообщения к ключу, выбранному пользователем. Спецификации HMAC см. в документе «HMAC: хеширование с ключом для аутентификации сообщений», H. Krawczyk, M. Bellare, R. Canetti, февраль 1997 г.

По сути, алгоритм состоит из двух вложенных хеш-операций:

- pad представляет собой последовательность нулей, необходимую для расширения ключа до длины блока данных базовой хеш-функции (то есть 512 бит для алгоритмов хеширования SHA-1, SHA224, SHA-256 и MD5);
- представляет оператор конкатенации.

Для расчета HMAC требуются четыре различных этапа:

1 Блок инициализируется путем записи бита INIT в «1» с битом MODE в «1» и битами ALGO, установленными в значение, соответствующее желаемому алгоритму. Бит LKEY также должен быть установлен на этом этапе, если используемый ключ длиннее 64 байт (в этом случае спецификации HMAC указывают, что вместо реального ключа следует использовать хеш-ключа). Также, при использовании длинного ключа и, если внутренний и внешний ключ одинаковый, следует установить бит SAMK для ускорения вычисления HMAC.

2 Затем ключ (который будет использоваться для внутренней хеш-функции) передается ядру. Эта операция использует тот же механизм, что и для отправки сообщения в хеш-операции (то есть путем записи в HASH_DIN и, наконец, в HASH_STR).

3 После ввода последнего слова и начала вычисления хеш-процессор обрабатывает ключ. Затем он готов принять текст сообщения, используя тот же механизм, который использовался для отправки сообщения в операции хеширования.

4 После первого раунда хеширования хеш-процессор возвращает «готово», чтобы указать, что он готов получить ключ, который будет использоваться для внешней хеш-функции (обычно этот ключ совпадает с тем, который используется для внутренней хеш-функции). Когда вводится последнее слово ключа и начинается вычисление, результат HMAC становится доступным в регистрах HASH_H0...HASH_H7.

Примечания

1 Задержка вычисления HMAC зависит от длины ключей и сообщения. Вы можете использовать HMAC как две вложенные базовые хеш-функции с одинаковой длиной ключа (длинным или коротким).

2 При использовании длинного ключа и, если внутренний и внешний ключ одинаковый и бит SAMK в регистре HASH_CR установлен, не требуется подавать внешний ключ (см. п.4). Результат HMAC будет доступен после обработки текста сообщения. Хеш-процессор возвратит сигнал «готово» и вы можете прочитать регистры HASH_HRx.

Ниже приведен пример алгоритма HMAC SHA-1 (ALGO=00 и MODE=1 в HASH_CR), из спецификации NIST. Предположим, что исходное сообщение представляет собой двоично-кодированную форму ASCII «Sample message for keylen=blocklen» длиной L = 34 байта.

Пример

1 В регистр HASH_DIN вводится внутренний ключ (длина равна 64, т. е. без заполнения), указанный в примере NIST. Поскольку длина ключа равна 64, бит LKEY установлен в 0 в регистре HASH_CR.

00010203 04050607 08090A0B 0C0D0E0F 10111213 14151617

18191A1B 1C1D1E1F 20212223 24252627 28292A2B 2C2D2E2F

30313233 34353637 38393A3B 3C3D3E3F

2 В регистр HASH_DIN вводится сообщение (длина равна 34, т. е. требуется заполнение). HASH_STR должен быть установлен на 0x20, чтобы начать заполнение сообщения и вычисление внутреннего хеша («U» - не важно)

53616D70 6C65206D 65737361 67652066 6F72206B 65796C65

6E3D626C 6F636B6C 656EUUUU

3 В регистр HASH_DIN вводится внешний ключ (длина равна 64, т.е. без заполнения), идентичный внутреннему ключу.

4 Затем выполняется окончательное вычисление хеш-функции. Результат HMAC-SHA1 доступен в регистрах HRx (x равен от 0 до 4), как показано ниже:

```
HR0 = 0x5FD596EE
HR1 = 0x78D5553C
HR2 = 0x8FF4E72D
HR3 = 0x266DFD19
HR4 = 0x2366DA29
```

15.10 Переключение контекста

Можно прервать процесс хеширования/HMAC, чтобы выполнить другую обработку с более высоким приоритетом, и завершить прерванный процесс позже, когда будет завершена задача с более высоким приоритетом. Для этого, контекст прерванной задачи должен быть сохранен из хеш-регистров в память, а затем восстановлен из памяти в хеш-регистры. Процедуры, в которых поток данных управляется программным обеспечением или прямым доступом к памяти, описаны ниже.

Процедура загрузки данных программным обеспечением

Контекст можно сохранить только тогда, когда ни один блок в данный момент не обрабатывается. То есть вы должны дождаться, когда DINIS = 1 (последний блок был обработан и входной FIFO пуст) или NBW ≠ 0 (FIFO не заполнен и обработка не ведется).

Сохранение контекста. Сохраните в памяти содержимое следующих регистров:

– HASH_IMR,

– HASH_STR,

– HASH_CR,

– от HASH_CSR0 до HASH_CSR53.

Восстановление контекста. Контекст может быть восстановлен после выполнения высокоприоритетной задачи. Для этого, следуйте порядку последовательности ниже.

а) Запишите следующие регистры со значениями, сохраненными в памяти: HASH_IMR, HASH_STR и HASH_CR.

б) Инициализируйте хеш-процессор, установив бит INIT в регистре HASH_CR.

с) Запишите регистры HASH_CSR0 в HASH_CSR53 со значениями, сохраненными в памяти.

Теперь вы можете возобновить обработку с того места, где она была прервана.

Процедура загрузки данных с механизма прямого доступа к памяти (ПДП)

В этом случае невозможно понять, идет ли передача ПДП или хеш-процессор находится в состоянии вычисления. Таким образом, необходимо остановить механизм ПДП, а затем дождаться готовности хеш-процессора, чтобы прервать обработку сообщения.

Прерывание обработки:

1) Очистите бит DMAE, чтобы отключить интерфейс ПДП.

2) Дождитесь завершения текущей передачи ПДП (дождитесь, когда DMAS = 0 в регистре HASH_SR). Обратите внимание, что блок может быть не полностью перенесен в хеш-процессор.

3) Отключить соответствующий канал в контроллере ПДП.

4) Подождите, пока хеш-процессор не будет готов (блок не обрабатывается), то есть дождитесь DINIS = 1

Этапы сохранения и восстановления контекста такие же, как описано выше (см. «Процедура загрузки данных программным обеспечением»).

Перенастройте контроллер DMA так, чтобы он передавал конец сообщения. Теперь вы можете перезапустить обработку с того места, где она была прервана, установив бит DMAE.

Примечание – Если замена контекста не включает операции HMAC, регистры HASH_CSR38–HASH_CSR53 не нужно сохранять и восстанавливать. Если обмен контекста происходит между двумя блоками (последний блок был полностью обработан, а следующий блок еще не помещен в IN FIFO, NBW = 000 в регистре HASH_CR), регистры HASH_CSR22 - HASH_CSR37 не нужно сохранять и восстанавливать.

15.11 Прерывания

Хеш-процессор имеет два отдельных маскируемых источника прерываний, подключённых к одному и тому же вектору прерывания.

Вы можете включить или отключить источники прерывания по отдельности, изменив биты маски в регистре IMR. Установка соответствующего бита маски в 1 разрешает прерывание.

Статус отдельных источников прерываний можно прочитать из регистра HASH_SR.

На рисунке 15.5 представлена схема формирования прерываний.

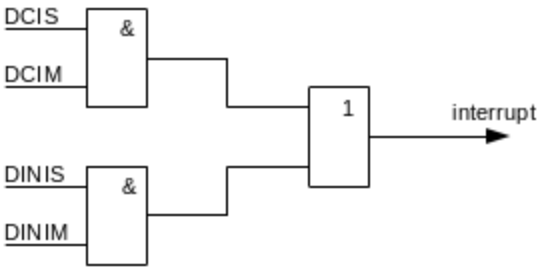


Рисунок 15.5 – Схема прерываний хеш-процессора.

16 Блок криптографии

16.1 Общие положения

Модуль реализует следующие алгоритмы шифрования:
- AES-128;
- AES-256;
- Кузнечик («4 Алгоритм блочного шифрования с длиной блока n = 128 бит» из ГОСТ 34.12-2018);
- Магма («5 Алгоритм блочного шифрования с длиной блока n = 64 бит» из ГОСТ 34.12-2018).

В определенный момент времени может выполняться либо шифрование, либо дешифрование блока текста одним из реализованных алгоритмов.

Модуль может формировать прерывание при возникновении одного или нескольких отслеживаемых событий, о чем информирует регистр IRQ. Прерывания маскируются с помощью регистра IRQ_ENABLE и бита INTERRUPT слова управления DMA текущего дескриптора.

AES-128 и AES-256 используют общий участок памяти внутри модуля для хранения раундовых ключей. Выполнение криптографической операции означает инвалидацию ранее рассчитанных раундовых ключей для остальных алгоритмов, которые совместно используют упомянутое хранилище раундовых ключей.

Важно отметить, что блоки открытого текста, шифртекста, ключа и инициализационного вектора сохраняются в регистрах с обратным порядком байт (Big-Endian).

16.2 Структура блока криптографии

Упрощенная функциональная схема блока криптографии показана на рисунке 16.1.

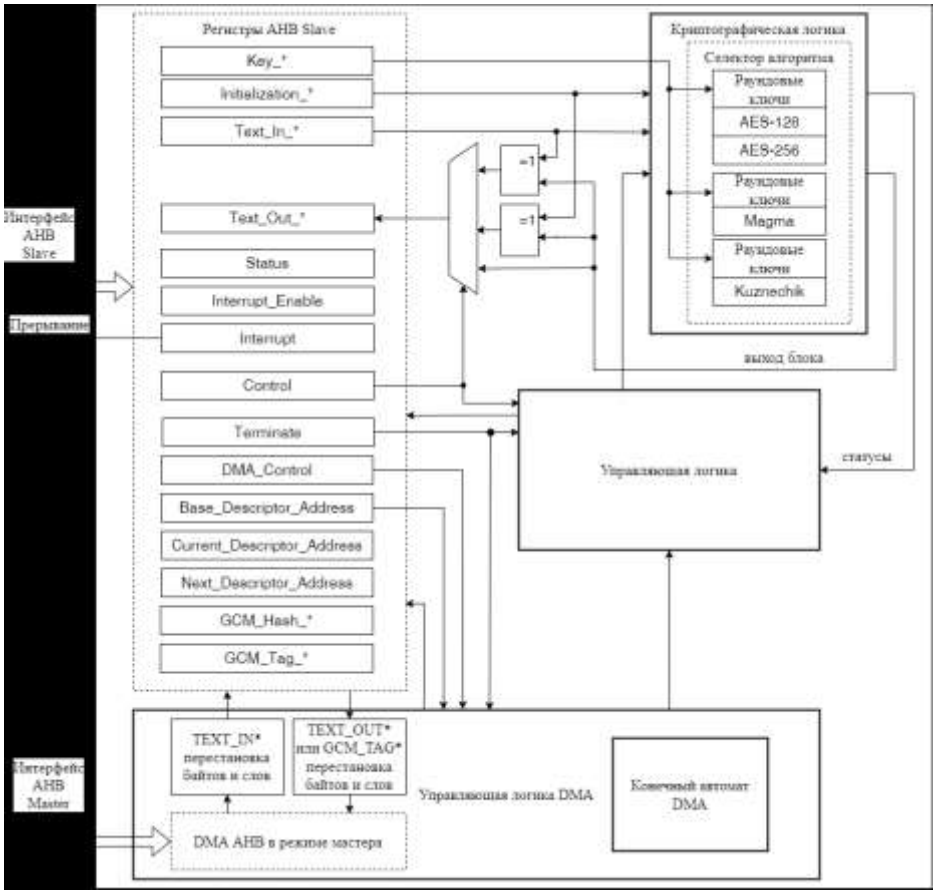


Рисунок 16.1 – Структурная схема блока криптографии

AES (FIPS PUB 197)

Реализованы две из трех приведенных в стандарте FIPS PUB 197 разновидностей алгоритма AES-128 и AES-256 с соответствующими разрядностями ключа. Длина шифруемого/дешифруемого блока – 128 бит.

Кузнечик (ГОСТ 34.12-2018)

«Кузнечик» («Kuznechik») – базовый блочный шифр, который описан в стандарте ГОСТ 34.12-2018, с длиной блока 128 бит и длиной ключа 256 бит.

Магма (ГОСТ 34.12-2018)

«Магма» («Magma») – базовый блочный шифр, который описан в стандарте ГОСТ 34.12-2018, с длиной блока 64 бит и длиной ключа 256 бит.

16.3 Режимы выполнения криптографических операций

Доступны следующие режимы:

- ECB – режим электронной кодовой книги (режим простой замены из ГОСТ 34.13-2018);
- CBC – режим простой замены с зацеплением» с параметром z = 1 ГОСТ 34.13-2018);
- CTR – режим гаммирования с параметром z = 1 из (ГОСТ 34.13-2018).

За выбор режима выполнения криптографической операции отвечает поле MODE регистра CONTROL.

Во всех случаях регистры KEY_* используются для формирования раундовых ключей выбранного алгоритма шифрования (если выбран алгоритм AES-128, то в формировании раундовых ключей участвуют только KEY_0, …, KEY_3).

При выполнении шифрования в регистры TEXT_IN_* записывается блок открытого текста, при дешифровании – блок зашифрованного текста. Если выбран алгоритм Magma, то значение TEXT_IN_2 и TEXT_IN_3 не влияют на результат выполнения криптографической операции.

При описании режимов выполнения криптографических операций используются следующие сокращения:

- P – открытый текст (plain text);
- C – зашифрованный текст (cipher text);
- IV – вектор инициализации текущего блока текста (формируется из значений регистров IV_*);
- Next_IV – вектор инициализации для следующего блока текста (данное значение автоматически будет записано в регистры IV_*);
- ^ – побитовая операция исключающего ИЛИ;
- E_in – входные данные криптографического блока в режиме шифрования (если используется алгоритм Magma, то в качестве входных данных будут использоваться старшие 64 бита, регистры с индексами 0 и 1);
- E_out – выходные данные криптографического блока в режиме шифрования (если используется алгоритм Magma, то результат будет дополнен справа 64 нулевыми битами);
- D_in – входные данные криптографического блока в режиме дешифрования;
- D_out – выходные данные криптографического блока в режиме дешифрования.

Режим электронной кодовой книги (ECB)

Режим электронной кодовой книги (ELECTRONIC_CODEBOOK, ECB). В ГОСТ 34.12-2018 он называется режимом простой замены.

Регистры IV_* не используются.

Шифрование (значение поля DIRECTION равно 0 – Encryption):
E_in = TEXT_IN = P;
C =TEXT_OUT = E_out.

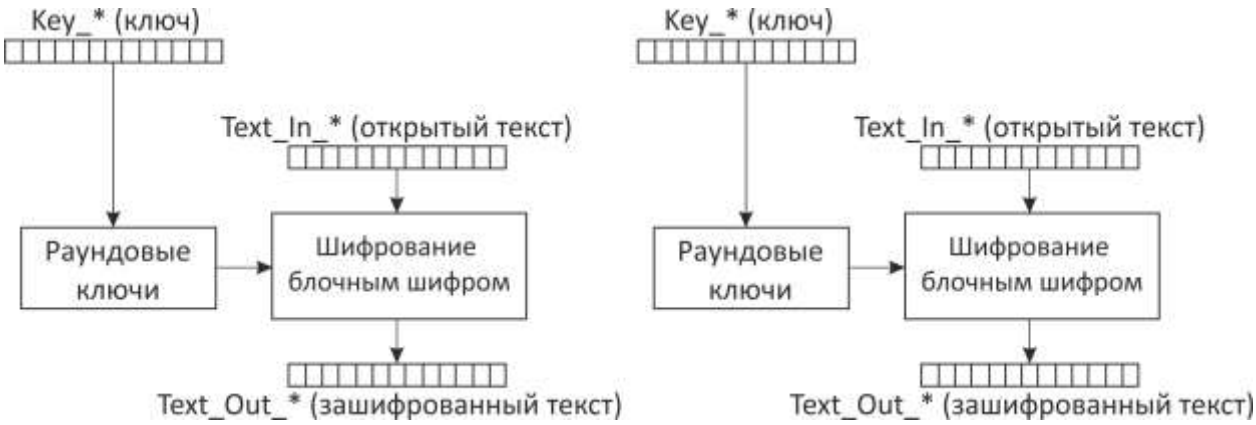


Рисунок 16.2 – Шифрование в режиме ECB

Дешифрование (значение поля DIRECTION равно 1 – Decryption):
D_in = TEXT_IN = C;
P = TEXT_OUT = D_out.

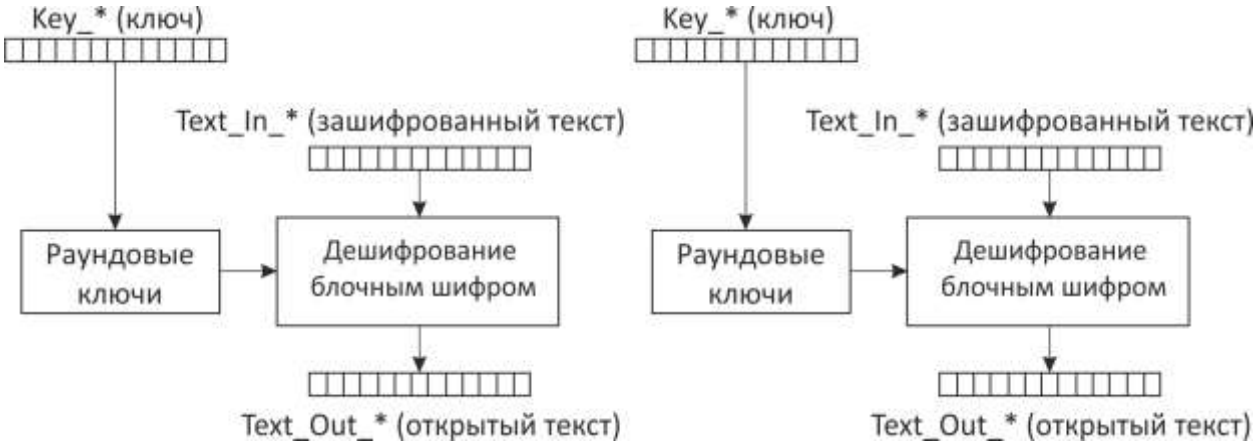


Рисунок 16.3 – Дешифрование в режиме ECB

Режим сцепления блоков шифротекста (CBC)

Режим сцепления блоков шифротекста (CIPHER_BLOCK_CHAINING, CBC). В ГОСТ 34.13-2018 он называется режимом простой замены с зацеплением (для случая, когда параметр z = 1).

Данный криптографический режим может быть использован для организации аутентификации сообщения (защиты его целостности), именуемой CBC-MAC (message authentication code). В качестве MAC, также называемого аутентификационным тегом или имитовставкой, берётся последний блок сообщения, зашифрованного блочным алгоритмом в режиме CBC.

Шифрование (значение поля DIRECTION равно 0 – Encryption):
E_in = TEXT_IN ^ IV = P ^ IV;
C =TEXT_OUT = E_out;
Next_IV = C = TEXT_OUT.

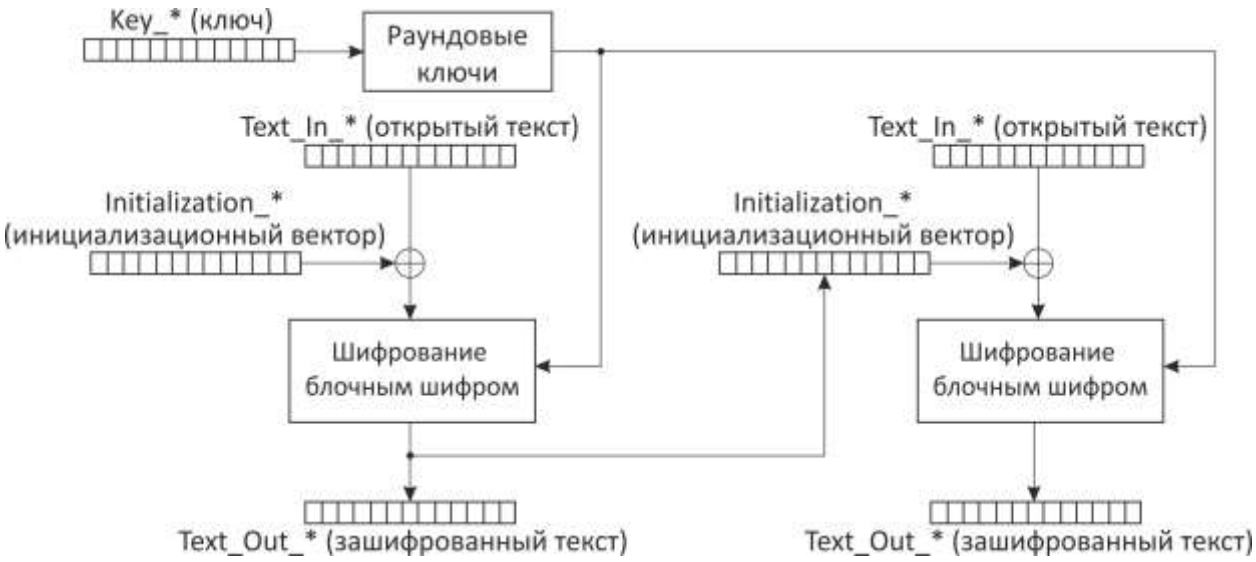


Рисунок 16.4 – Шифрование в режиме CBC

Дешифрование (значение поля DIRECTION равно 1 – Decryption):
D_in = TEXT_IN = C;
P = TEXT_OUT = D_out ^ IV;
Next_IV = C = TEXT_IN.

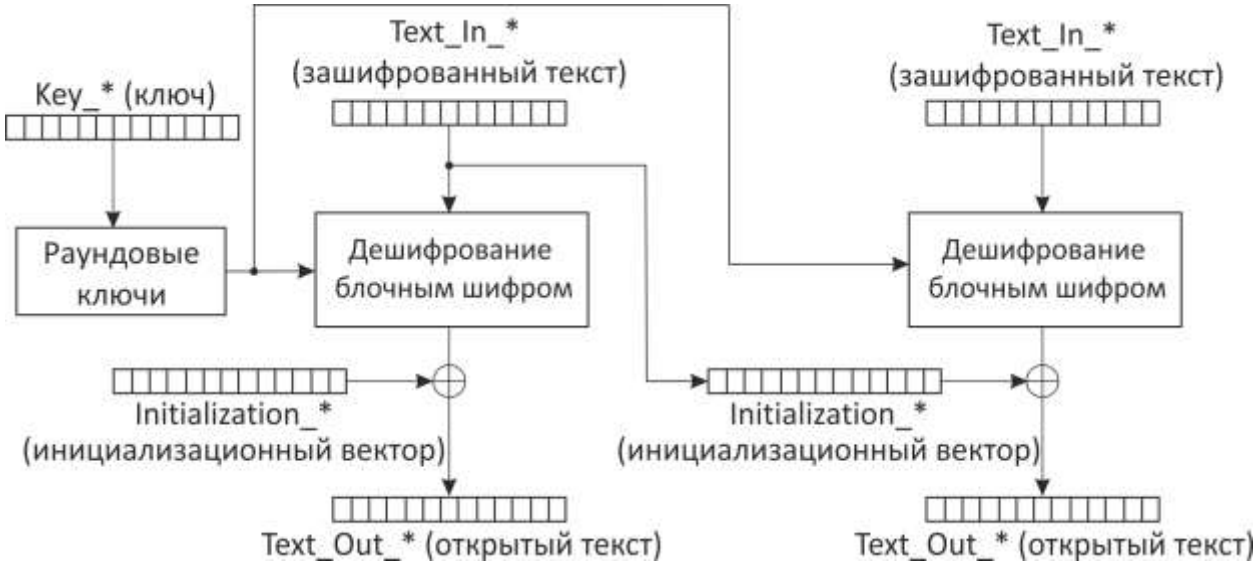


Рисунок 16.5 – Дешифрование в режиме CBC

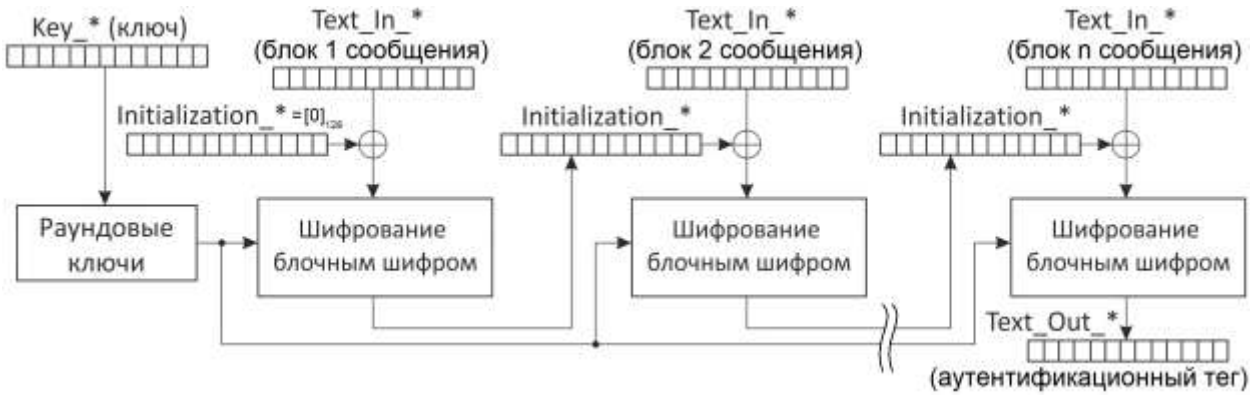


Рисунок 16.6 – Формирование аутентификационного тега сообщения (CBC-MAC)

Режим счётчика (CTR)

Режим счётчика (COUNTER_MODE, CTR). В ГОСТ 34.13-2018 он называется режимом гаммировани (для случая, когда параметр z = 1).

Шифрование (значение поля DIRECTION равно 0 – Encryption):

E_in = IV;

C = TEXT_OUT = E_out ^ TEXT_IN = E_out ^ P;

Next_IV = IV + 1 (если алгоритм – Magma, то инкрементируется IV_1, в остальных случаях IV_3).

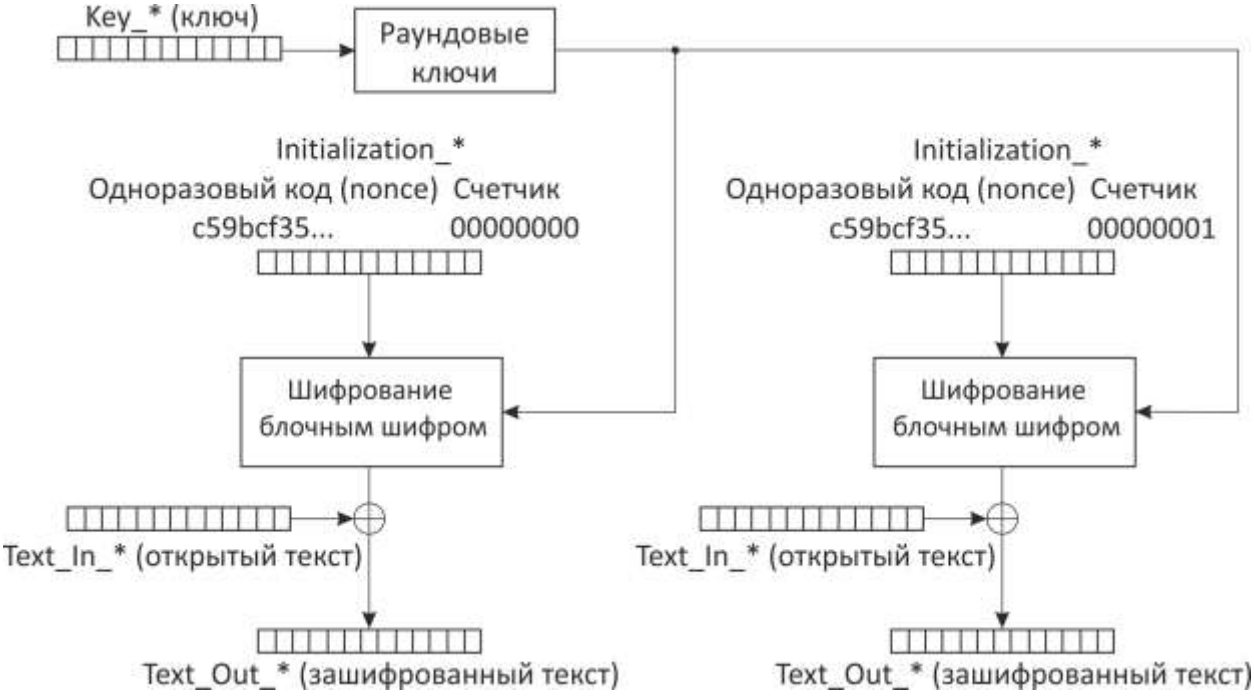


Рисунок 16.7 – Шифрование в режиме CNT

Дешифрование (значение поля DIRECTION равно 1 – Decryption):

E_in = IV;

P = TEXT_OUT = E_out ^ TEXT_IN = E_out ^ C;

Next_IV = IV + 1 (если алгоритм – Magma, то инкрементируется IV_1, в остальных случаях IV_3).

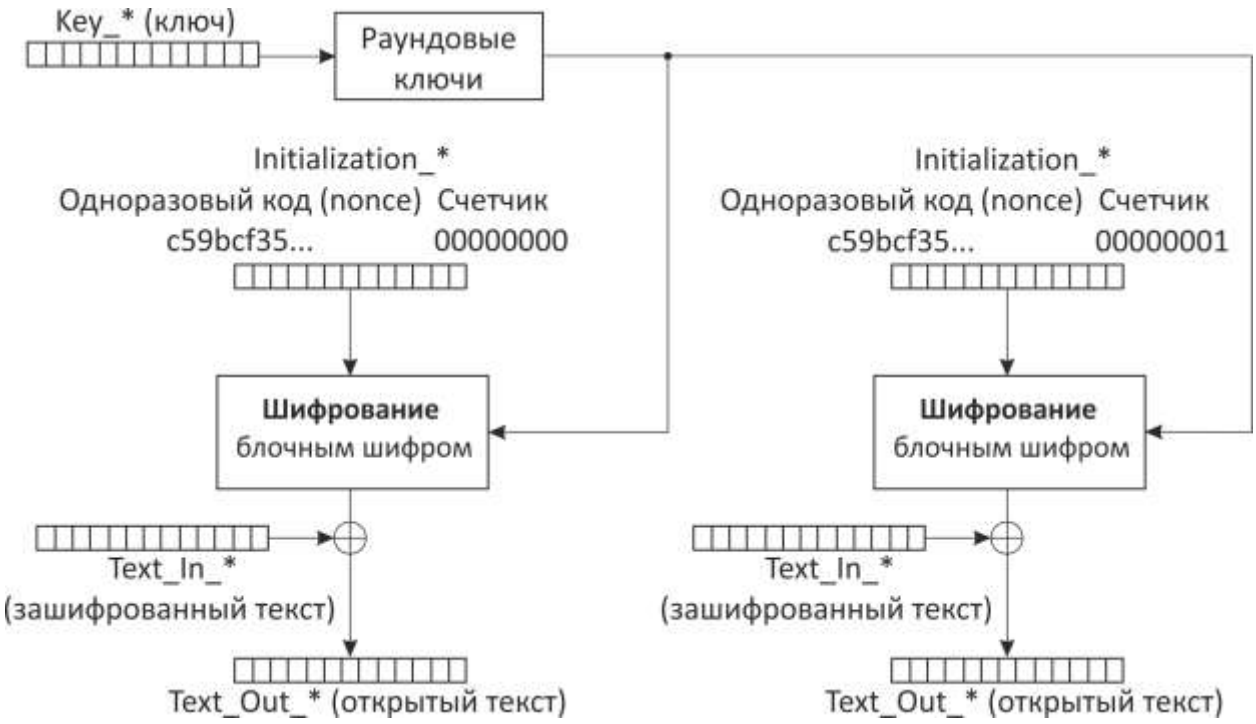


Рисунок 16.8 – Дешифрование в режиме CNT

Режим счётчика с аутентификацией Галуа (GCM)

Шифрование и дешифрования блоков текста в режиме GCM аналогично таковому в режиме счетчика (CTR). Данный режим является режимом аутентифицированного шифрования (Authenticated Encryption with Associated Data, AEAD). AEAD – класс блочных режимов шифрования, при котором часть сообщения шифруется, часть остается открытой, и всё сообщение целиком аутентифицировано (при совпадении значения аутентификационного тега сообщений гарантируется неизменность количества блоков текста, их содержимого и порядка следования в сообщении).

Выполнение расчета аутентификационного тега разбито на четыре фазы, которые должны выполняться последовательно:

– GCM_INIT. Обнуляется значения регистров GCM_Tag_*. На основе текущего состояния регистров KEY_* производится расчет хеш-ключа, который будет использован в качестве множителя при выполнении умножения в поле Галуа GF(2^{128}). Значение хеш-ключа отображается в GCM_Hash_*;

– GCM_HEADER. В регистры TEXT_IN_* помещается блок дополнительных аутентифицируемых данных (Additional authenticated data – AAD). В GCM_Tag_* попадает результат выполнения ИСКЛЮЧАЮЩЕГО ИЛИ над текущим значением тэга и произведением AAD на хеш-ключ;

– GCM_PAYLOAD. В регистры TEXT_IN_* помещаются открытый текст/шифрованный текст, который будет зашифрован/расшифрован соответственно. В GCM_Tag_* попадает результат выполнения ИСКЛЮЧАЮЩЕГО ИЛИ над текущим значением тэга и произведением блока шифрованного текста на хеш-ключ;

– GCM_LAST_BLOCK. В регистры TEXT_IN_0, TEXT_IN_2 заносятся нули. В TEXT_IN_1 – количество бит AAD, которые обрабатываются в фазе GCM_HEADER. В TEXT_IN_3 – количество шифруемых/дешифруемых бит полезной нагрузки, которые обрабатываются в фазе GCM_PAYLOAD. В GCM_Tag_* попадает результат выполнения ИСКЛЮЧАЮЩЕГО ИЛИ над текущим значением тэга, произведением TEXT_IN_* на хеш-ключ, результатом шифрования значения IV_*, в котором IV_3 подменяется на 0x00000001.

Ниже приведены рисунки, которые иллюстрируют состав сообщения, операции над которым производятся в режиме GCM согласно стандарту NIST SP 800-38D.

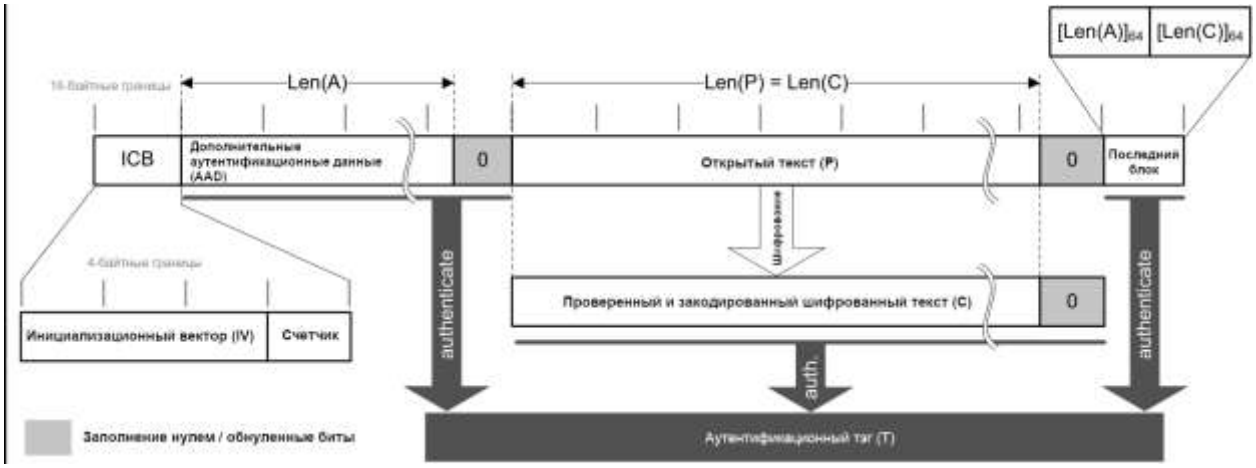


Рисунок 16.9 – Структура шифруемого сообщения в режиме GCM

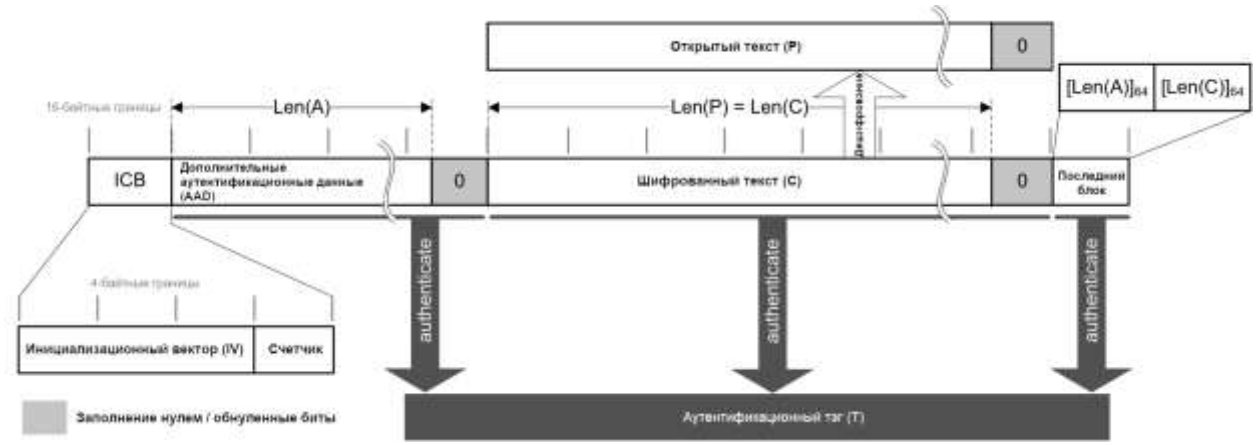


Рисунок 16.10 – Структура дешифруемого сообщения в режиме GCM

Регистры IV_* формируют начальное значение счетчика (initial counter block, ICB), состоящее из двух частей:

– Регистры IV_0, IV_1, IV_2 составляют 96-битное значение инициализационного вектора, которое должно вместе с ключом составлять уникальную для каждого шифруемого сообщения пару. Стоит отметить, что стандарт GCM поддерживает использование вектора инициализации меньшей длины, но в данном случае применяется точное значение в 96 бит;

– Регистр IV_3, который содержит счетчик, инкрементируемый каждый раз после завершения шифрования/дешифрования блока текста. Согласно стандарту NIST SP 800-38D значение счетчика при обработке первого блока текста – 0x00000002. Данное значение будет занесено в IV_3 после выполнения фазы GCM_INIT, если был установлен бит SELF_UPDATE регистра CONTROL или выполнение фазы является результатом работы в поточном режиме.

Неправильное использование инициализационного вектора (IV) может негативно сказаться на безопасности GCM. Например, повторное использование пары ключа и IV приведет к потере конфиденциальности сообщений, в которых использовалось одно и то же значение IV. Повторное использование IV отправителем может позволить

злоумышленнику идентифицировать ключ. Это может привести к подделке сообщений. Рекомендации по генерации инициализационного вектора приведены в NIST SP 800-38D (в разделе «8 Uniqueness Requirement on IVs and Keys»).

Если длина AAD не кратна 128 бит, пользователь должен дополнить нулями последний блок дополнительных аутентифицируемые данных.

Если длина дешифруемого текста не кратна 128 бит, пользователь должен дополнить нулями последний блок. Обнуление неактуальных бит последнего блока открытого текста не предусмотрено, их значение следует игнорировать.

Важно отметить, что модуль не накладывает маску на последний зашифрованный блок текста. Поэтому шифрование сообщений с длиной полезной нагрузки, которая не кратна 128 бит, не поддерживается (будет получен некорректный аутентификационный тег).

Ниже приведены рисунки, которые иллюстрируют процесс шифрования и дешифрования сообщения в режиме GCM.

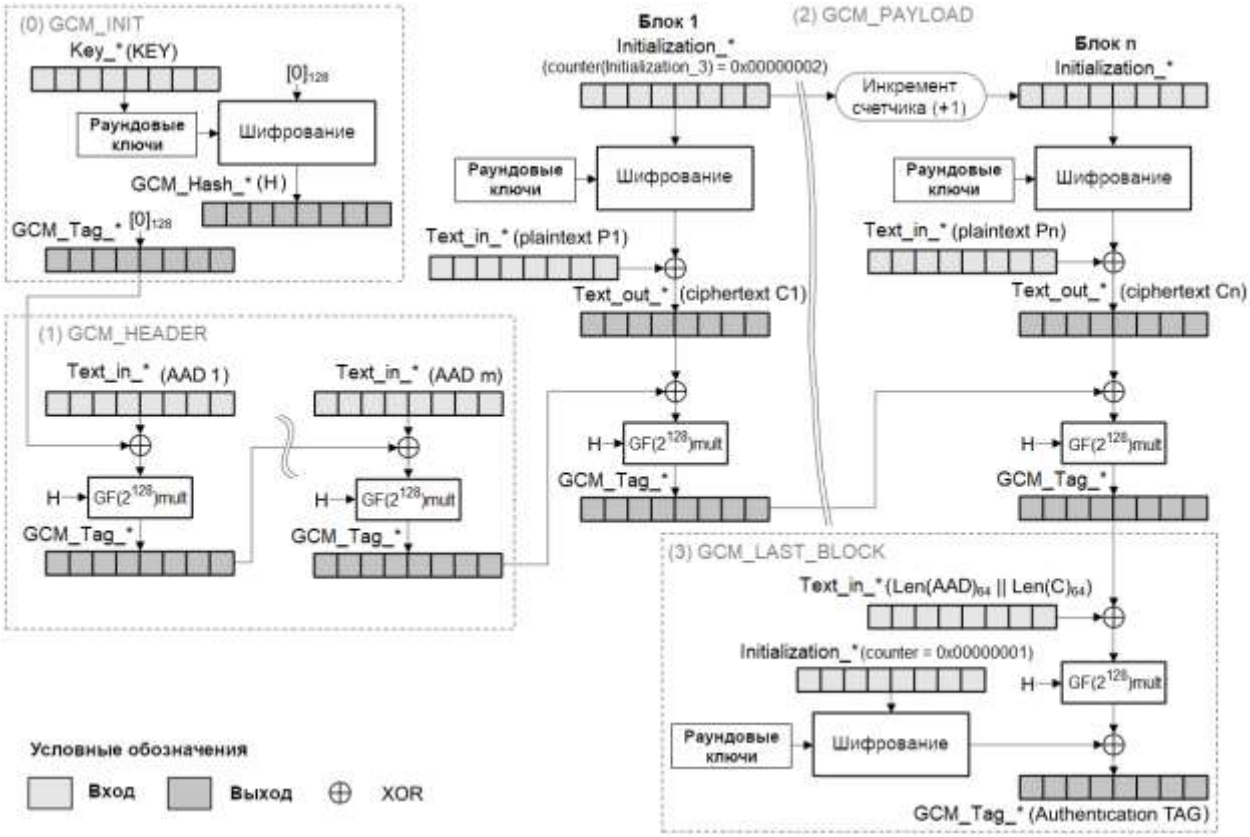


Рисунок 16.11 – Шифрование сообщения в режиме GCM

Дешифрование в фазе GCM_PAYLOAD выполняется параллельно с получением блока открытого текста. Поэтому суммарно на обработку сообщения будет потрачено меньше времени, чем при его шифровании.

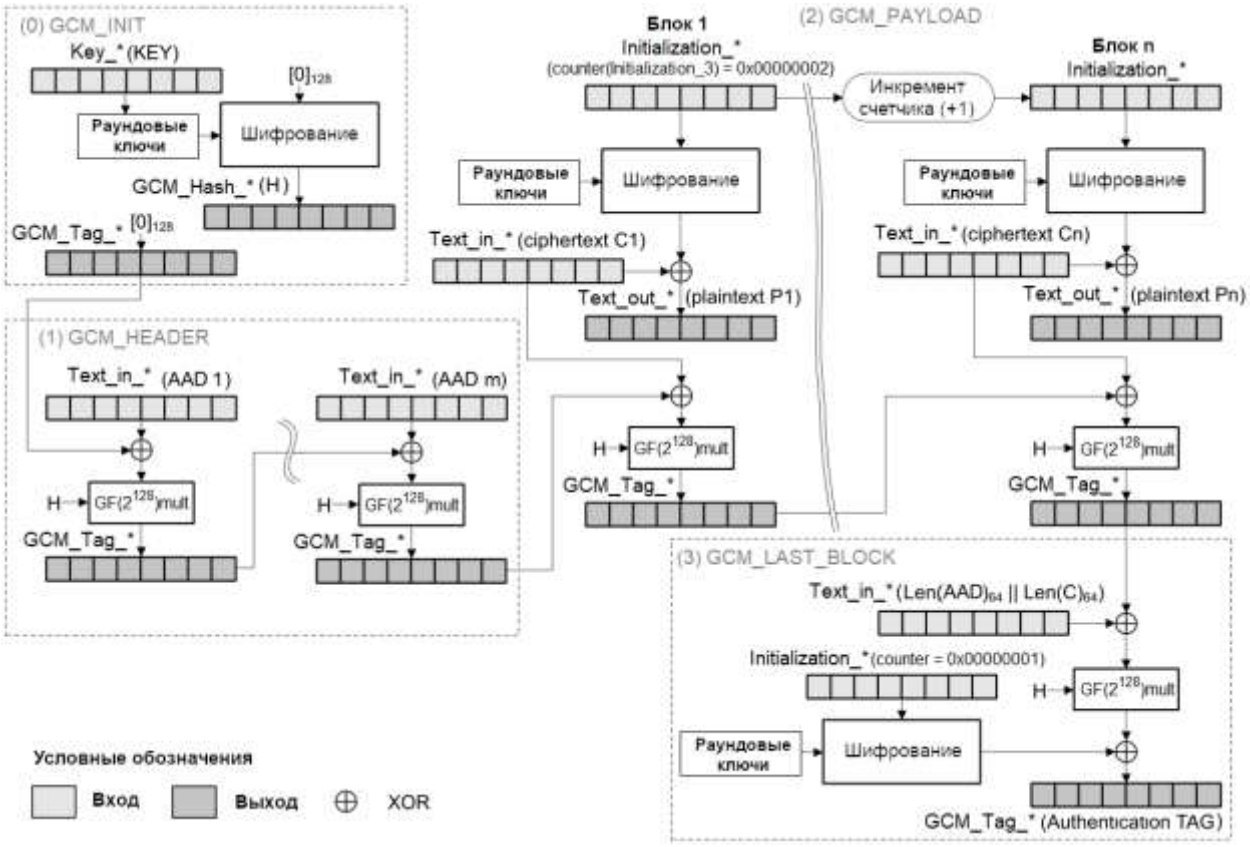


Рисунок 16.12 – Дешифрование сообщения в режиме GCM

Вырожденным случаем GCM является код аутентификации сообщения на основе полей Галуа (Galois message authentication code – GMAC). Процесс получения тега аналогичен таковому для обычного GCM, в котором пропущена фаза GCM_PAYLOAD.

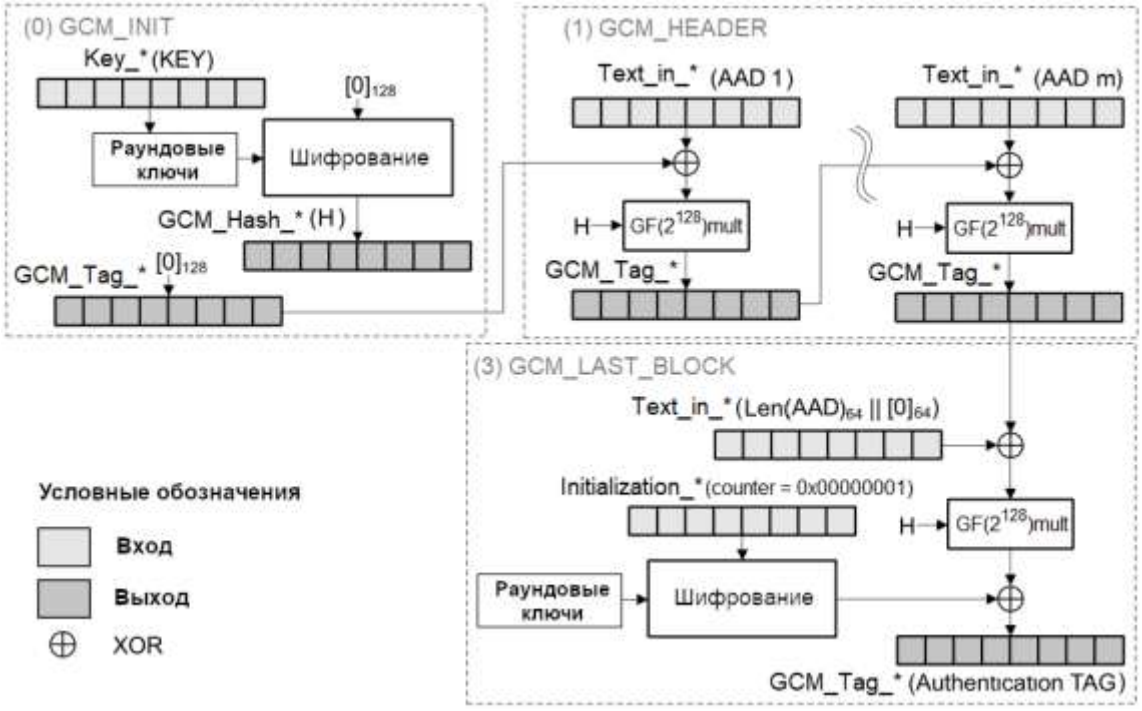


Рисунок 16.13 – Формирование аутентификационного тега в режиме GMAC

Неподдерживаемые конфигурации режима GCM

Запись в регистр CONTROL значения MODE равного GALOIS_COUNTER_MODE в сочетании с ALGORITHM соответствующим MAGMA блокирует запуск криптографической операции (установленный бит START регистра CONTROL). В случае использования режима с прямым доступом к памяти (DMA), если в слове управления DMA выставлен GALOIS_COUNTER_MODE в качестве MODE и значение ALGORITHM соответствует MAGMA, выполнение дескриптора завершится с ошибкой. О данном факте будет свидетельствовать значение поля BAD_DESCR равный 0001_2 (CONTROL) регистра STATUS и, если был установлен бит DMA_FAIL в регистре IRQ_ENABLE, установка бита прерывание DMA_FAIL в регистре IRQ.

Как оговорено ранее, в режиме GALOIS_COUNTER_MODE не поддерживается шифрование сообщений с длиной полезной нагрузки (блоков шифрованного текста), которая не кратна 128 бит.

16.4 Режимы работы

Общие положения

В криптографическом модуле реализованы режим работы с одним блоком текста и поточный режим. Последний из них также можно назвать режимом работы с использованием прямого доступа к памяти (DMA).

Регистры IV _*, KEY_*, STATUS, IRQ используются одинаково в обоих режимах. Регистры BASE_DESCRIPTOR, CURRENT_DESCRIPTOR, NEXT_DESCRIPTOR используются только в режиме работы с DMA.

Важно понимать, что в TEXT_OUT_* не хранится результат криптографической операции (зашифрованный текст/открытый текст). Отображаемое в данную область значение зависит от состояния полей MODE, DIRECTION и ALGORITHM регистра CONTROL (описание полей приведено в «Описании режима работы с одним блоком»). Для режимов криптографической операции отличных от ELECTRONIC_CODEBOOK корректность TEXT_OUT_* после завершения операции (зашифрованный текст для шифрования/открытый текст для дешифрования) гарантируется только при условии неизменности IV_*/TEXT_IN_* и вышеупомянутых полей регистра CONTROL. В таблице ниже приведена зависимость отображаемого значения от MODE и DIRECTION (см. «Режимы выполнения криптографических операций»).

Таблица 16.1 – Значение, отображаемое в регистры TEXT_OUT_*

Поля регистра CONTROL	Значение TEXT_OUT_*
MODE = 1 (CIPHER_BLOCK_CHAINING), DIRECION = 1 (Decrypt)	algorithm_result ^ IV_*
MODE = 2 (COUNTER_MODE) или MODE = 3 (GALOIS_COUNTER_MODE)	algorithm_result ^ TEXT_IN_*
В остальных случаях	algorithm_result

В регистре STATUS не хранится значение поля KEYS_READY, в него отображается соответствующий бит поля KEYS_STORED текущего выбранного алгоритма, который определяется значением поля CONTROL.ALGORITHM. Поле TEXT_OUT_VALID имеет схожую зависимость от CONTROL.ALGORITHM. Установленное значение TEXT_OUT_VALID означает, что выходные данные активного криптографического подмодуля не изменились с момента завершения последней операции шифрования/дешифрования.

Модуль в состоянии хранить до трех результатов шифрования/дешифрования входных данных (TEXT_IN_*, IV_* или TEXT_IN_* ^ IV _* в зависимости от режима выполнения криптографической операции, задаваемого CONTROL.MODE). По одному на

криптографический подмодуль. Их три, поскольку AES-128 и AES-256, объединены в один подмодуль.

Регистр STATUS (структура регистров и битовых полей блока криптографии приведена в приложении А.12).

DMA_LAST – Поле отображает значение бита LAST слова управления DMA последнего прочитанного дескриптора. Бит установлен, если прочитанный дескриптор является последним в цепочке дескрипторов. Значение обновляется при считывании нового дескриптора. Бит сбрасывается при экстренном прекращении выполнения операции.

DMA_COUNTER – Поле отображает количество обработанных DMA блоков текст. Значение обнуляется при считывании нового дескриптора. Значение сбрасывается при экстренном прекращении выполнения операции. Предупреждение: если в слове управления DMA в поле COUNT записано максимально возможное значение, то после обработки последнего блока текста произойдет переполнение счетчика, который отображается в DMA_COUNTER.

BAD_DESCR – Поле хранит сведения об обнаруженных некорректных словах в прочитанном дескрипторе DMA. Список возможных значений:

0000_2 – NO_ERROR;

0001_2 – CONTROL:

0010_2 – TEXT_IN_ADDRESS;

0100_2 – TEXT_OUT_ADDRESS;

1000_2 – NEXT_DESCRIPTOR.

После записи в регистр DMA_CONTROL любого значения поле будет переведено в состояние NO_ERROR.

AHB_ERROR – Поле хранит сведения об области памяти, на которую пришелся ответ ERROR на транзакцию по шине AHB. Список возможных значений:

000_2 – NO_ERROR;

001_2 – READ_DESCRIPTOR;

010_2 – READ_TEXT_IN;

100_2 – WRITE_TEXT_OUT.

После записи в регистр DMA_CONTROL любого значения поле будет переведено в состояние NO_ERROR.

DMA_ACTIVE – Если бит установлен, то конечный автомат DMA занят выполнением дескриптора. Бит READY будет находиться в сброшенном состоянии, пока DMA_ACTIVE установлен. Бит сбрасывается после выполнения цепочки дескрипторов, обнаружения некорректного дескриптора, получения ответа ERROR по шине AHB или при экстренном прекращении выполнения операции.

KEYS_STORED – Каждый бит данного поля соответствует заданному криптографическому алгоритму. Биты нумеруются справа налево. 0 – AES-128, 1 – AES-256, 2 – MAGMA, 3 – KUZNECHIK. Если бит, который относится к интересующему криптографическому алгоритму, установлен, то в модуле хранится его раундовые ключи. Бит сбрасывается при старте обновления раундовых ключей заданного алгоритма, в случае, когда одновременное хранение раундовых ключей не предусмотрено (это актуально для пары алгоритмов AES-128 и AES-256), при экстренном прекращении выполнения операции.

IRQ_PENDING – Если бит установлен, то регистр IRQ содержит хотя бы 1 не сброшенный бит.

TEXT_OUT_VALID – Если бит установлен, то на выходе криптографического подмодуля находится результат шифрования/дешифрования входных данных (TEXT_IN_*, IV_* или TEXT_IN_* ^ IV_* в зависимости от алгоритма и режима выполнения криптографической операции). Для операций кроме дешифрования в режиме CIPHER_BLOCK_CHAINING, режимах COUNTER_MODE и GALOIS_COUNTER_MODE

установленный бит TEXT_OUT_VALID эквивалентен тому, что регистры TEXT_OUT содержат результат выполнения предыдущей криптографической операции. Для обоих режимов счетчика для корректности результата требуется еще и неизменность значения TEXT_IN_*, для дешифрования в режиме CIPHER_BLOCK_CHAINING – IV_*. Выполнение обновления раундовых ключей не оказывает на данный бит влияния (например, запись в регистр CONTROL установленных бит UPDATE_KEY и START сбросит данный бит только после перерасчета раундовых ключей). Бит сбрасывается при старте выполнения криптографической операции (запись в регистр CONTROL установленного бита START) или при экстренном прекращении выполнения операции.

KEYS_READY – Если бит установлен, то раундовые ключи текущего активного криптографического алгоритма сгенерированы. Бит сбрасывается при старте обновления раундовых ключей или при экстренном прекращении выполнения операции.

READY – Если бит установлен, то модуль готов к выполнению очередной криптографической операции и/или обновлению раундовых ключей шифрования. Бит сбрасывается при старте обновления раундовых ключей, выполнения криптографической операции и считывания первого дескриптора цепочки. Бит устанавливается после завершения всех запланированных действий или при экстренном прекращении выполнения операции.

Регистр IRQ_ENABLE

DMA_FAIL – Если бит установлен, то функционал бита IRQ.DMA_FAIL разрешен.

DMA_DONE – Если бит установлен, то функционал бита IRQ.DMA_DONE разрешен. Бит доступен только для чтения и отображает значение бита INTERRUPT управляющего слова дескриптора DMA.

WR_IGNORED – Если бит установлен, то функционал бита IRQ.WR_IGNORED разрешен.

DONE – Если бит установлен, то функционал бита IRQ.DONE разрешен.

Регистр IRQ

DMA_FAIL – Бит устанавливается, если установлен соответствующий бит регистра IRQ_ENABLE, при преждевременном завершении работы DMA из-за обнаружения ошибки (см. поля AHB_ERROR и BAD_DESCRIPTOR регистра STATUS). Бит может быть сброшен записью 1.

DMA_DONE – Бит устанавливается, если бит INTERRUPT управляющего слова текущего дескриптора DMA был установлен (его состояние отображается в соответствующий бит регистра IRQ_ENABLE), при успешном завершении выполнения дескриптора DMA (поля STATUS.BAD_DESCRIPTOR и STATUS.AHB_ERROR содержат нулевое значение). Бит может быть сброшен записью 1.

WR_IGNORED – Бит устанавливается, если установлен соответствующий бит регистра IRQ_ENABLE, при выполнении записи в регистр CONTROL, TEXT_IN_* или IV_* пока STATUS.READY сброшен. Бит может быть сброшен записью 1.

DONE – Бит устанавливается, если установлен соответствующий бит регистра IRQ_ENABLE, при переходе в режим готовности (переключение STATUS.READY в 1). Если запись в регистр CONTROL была выполнена с одновременно установленными битами UPDATE_KEY и START, то прерывание будет сгенерировано однократно и только после готовности выходных данных. Если сброс бита STATUS.READY вызван записью установленного бита START регистра DMA_CONTROL, то прерывание будет сгенерировано однократно после успешного завершения всей цепочки дескрипторов, либо при обнаружении некорректного дескриптора (ненулевое значение STATUS.BAD_DESCR), либо в результате возникновения ошибки на шине AHB (ненулевое значение STATUS.AHB_ERROR). Бит может быть сброшен записью 1.

Описание режима работы с одним блоком

Запуск режима работы с одним блоком текста осуществляется с помощью регистра CONTROL.

Запись в регистр CONTROL, DMA_CONTROL, TEXT_IN_* или IV_* будет проигнорирована, если бит STATUS.READY сброшен. Если предварительно был установлен бит IRQ_ENABLE.WR_IGNORED, то будет сгенерировано соответствующее прерывание (INTERRUPT.WR_IGNORED).

Для получения корректного результата криптографической операции необходимо иметь предварительно рассчитанные для выбранного алгоритма (поле ALGORITHM регистра CONTROL) раундовые ключи. Их расчёт может быть выполнен как отдельное действие (CONTROL.START = 0, CONTROL.UPDATE_KEY = 1), либо как предварительный этап криптографической операции (CONTROL.START = 1, CONTROL.UPDATE_KEY = 1). Если для текущего алгоритма нет рассчитанных раундовых ключей (см. соответствующий бит поля KEYS_STORED регистра STATUS), то автоматически будет выполнен перерасчет раундовых ключей на основе текущего значения регистров KEY_*.

Регистр CONTROL

GCM_PHASE – Фаза обработки пакета в режиме GALOIS_COUNTER_MODE:

0 – GCM_INIT;

1 – GCM_HEADER;

2 – GCM_PAYLOAD;

3 – GCM_LAST_BLOCK.

Если режим выполнения криптографической операции не GALOIS_COUNTER_MODE, то значение поля игнорируется.

SELF_UPDATE – Разрешение автоматического обновления содержимого регистров IV_* при завершении выполнения криптографической операции. После шифрования в режиме CIPHER_BLOCK_CHAINING в IV_* будет записано содержимое TEXT_OUT_*, после дешифрования – TEXT_IN_* (что повлияет на считываемое из регистров TEXT_OUT_* значение, как это описано в замечании ниже). Для режима COUNTER_MODE на 1 будет увеличено значение или регистра IV_1 для алгоритма MAGMA, или IV_3 для остальных случаев. Для режима GALOIS_COUNTER_MODE в фазе GCM_INIT в регистр IV_3 будет занесено значение 0x00000002. Для режима GALOIS_COUNTER_MODE в фазах GCM_HEADER или GCM_PAYLOAD на 1 будет увеличено значение регистра IV_3.

Примечание − Если CONTROL.SELF_UPDATE был установлен при запуске дешифрования в режиме CIPHER_BLOCK_CHAINING, то для получения расшифрованного блока текста потребуется вычислить (TEXT_OUT_* ^ IV_* ^ предыдущее значение IV_*), где ^ – операция побитового ИЛИ (см. «Режим сцепления блоков шифротекста CBC»). Альтернативным способом будет являться переключение DIRECTION в 0 или MODE в 0 с последующим вычислением (TEXT_OUT_* ^ предыдущее значение IV_*).

MODE – Выбор режима выполнения криптографической операции:

0 – ELECTRONIC_CODEBOOK (ECB);

1 – CIPHER_BLOCK_CHAINING (CBC);

2 – COUNTER_MODE (CTR);

3 – GALOIS_COUNTER_MODE (GCM).

ALGORITHM – выбора алгоритма криптографической операции (00_2 – AES_128, 01_2 – AES_256, 10_2 – MAGMA, 11_2 – KUZNECHIK).

DIRECTION – направление криптографической операции (0 – Encryption, 1 - Decryption). При выполнении операции шифрования (Encryption) значение регистров {TEXT_IN_0, ..., TEXT_IN_3} или {TEXT_IN_0, TEXT_IN_1} для алгоритма Magma будут интерпретированы модулем как открытый текст (plain text), а при завершении выполнения в соответствующие регистры TEXT_OUT попадет шифрованный текст (cipher text). В

случае дешифрования (Decryption) TEXT_IN будет считаться шифрованным текстом, а в TEXT_OUT будет помещен дешифрованный открытый текст.

START – Запрос на выполнение криптографической операции. Записываемое в данный бит значение будет проигнорировано, если в поле MODE записывается значение, которое соответствует GALOIS_COUNTER_MODE, а в поле ALGORITHM – MAGMA. Непосредственный старт может быть отложен до завершения обновления раундовых ключей (спровоцированное установленным битом UPDATE_KEY или выполненное из-за отсутствия раундовых ключей для выбранного криптографического алгоритма). При чтении всегда возвращает 0.

UPDATE_KEY – Принудительно выполнить обновление раундовых ключей шифрования на основе текущего значения регистров {KEY_0, ..., KEY_7} или {KEY_0, ..., KEY_3} для алгоритма AES-128. Записываемое в данный бит значение будет проигнорировано, если в поле MODE записывается значение, которое соответствует GALOIS_COUNTER_MODE, а в поле ALGORITHM – MAGMA. При чтении всегда возвращает 0.

Описание поточного режима работы

Размещение в памяти любого из дескрипторов должно иметь выравнивание по границе в четыре слова. В регистр BASE_DESCRIPTOR можно записать только корректный адрес дескриптора, поскольку младшие четыре бита доступны только для чтения.

Запуск потокового режима работы осуществляется с помощью регистра DMA_CONTROL.

Запись в регистр DMA_CONTROL, CONTROL, TEXT_IN_* или IV_* будет проигнорирована, если бит STATUS.READY сброшен. Если предварительно был установлен бит IRQ_ENABLE.WR_IGNORED, то будет сгенерировано соответствующее прерывание (IRQ.WR_IGNORED).

Для получения корректного результата криптографической операции необходимо иметь предварительно рассчитанные для выбранного алгоритма раундовые ключи. Если таковых нет для заданного в слове управления DMA поля ALGORITHM, будет автоматически инициирован их расчет, даже при сброшенном бите UPDATE_KEY. Если набор операций, заданный дескриптором, должен быть осуществлен с использованием нового ключа, то необходимо установить бит UPDATE_KEY.

Регистр управления DMA (**DMA_CONTROL**)

WORDS_SWAP – Если бит сброшен, то заполнение регистров TEXT_IN_* считанными из памяти словами входного текста идет в порядке увеличения индекса регистра. То есть данные попадают в регистр TEXT_IN_0, следующее слово (адрес которого больше на четыре) в TEXT_IN_1 и так далее всех алгоритмов кроме «Магма» (регистры с индексами TEXT_IN_2 и TEXT_IN_3 не используются). Если бит установлен, то последовательность заполнения регистров TEXT_IN_* меняется на обратную (от TEXT_IN_1 к TEXT_IN_0 для алгоритма MAGMA, от TEXT_IN_3 к TEXT_IN_0 в остальных случаях). Аналогичное влияние оказывается на пересылку выходных слов текста из регистров TEXT_OUT_* в память. Данный бит не меняет порядок считывания слов дескриптора.

BYTES_SWAP – Если бит установлен, то перед записью в регистры TEXT_IN_* в прочитанном слове будет изменен порядок следования байт, пересылаемые из TEXT_OUT_* слова будут выставлены на шину AHB с инвертированным порядком байт. Данный бит не оказывает влияние на считываемые слова дескриптора

START – Установленный бит запускает последовательность операций в режиме прямого доступа к памяти. При чтении всегда возвращает 0.

Ниже приведены варианты размещения байт данных в регистрах TEXT_IN_* для оперативной памяти с порядком байт от младшего к старшему (в которой байт 0xD4 со смещением адреса 0x0, байт 0xC3 со смещением 0x1, байт 0xB2 со смещением 0x2 и байт 0xA1 со смещением 0x3 будут образовывать число 0xA1B2C3D4) и размера блока текста в 128 бит.

Размещение байт в TEXT_IN [127 : 0] / TEXT_OUT [127 : 0] при WORDS_SWAP = 0, BYTES_SWAP = 0:

TEXT_IN_0 [31:0]				TEXT_IN_1 [31:0]				TEXT_IN_2 [31:0]				TEXT_IN_3 [31:0]			
0x3	0x2	0x1	0x0	0x7	0x6	0x5	0x4	0xB	0xA	0x9	0x8	0xF	0xE	0xD	0xC

Размещение байт в TEXT_IN [127 : 0] / TEXT_OUT [127 : 0] при WORDS_SWAP = 0, BYTES_SWAP = 1:

TEXT_IN_0 [31:0]				TEXT_IN_1 [31:0]				TEXT_IN_2 [31:0]				TEXT_IN_3 [31:0]			
0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xA	0xB	0xC	0xD	0xE	0xF

Размещение байт в TEXT_IN [127 : 0] / TEXT_OUT [127 : 0] при WORDS_SWAP = 1, BYTES_SWAP = 0:

TEXT_IN_0 [31:0]				TEXT_IN_1 [31:0]				TEXT_IN_2 [31:0]				TEXT_IN_3 [31:0]			
0xF	0xE	0xD	0xC	0xB	0xA	0x9	0x8	0x7	0x6	0x5	0x4	0x3	0x2	0x1	0x0

Размещение байт в TEXT_IN [127 : 0] / TEXT_OUT [127 : 0] при WORDS_SWAP = 1, BYTES_SWAP = 1:

TEXT_IN_0 [31:0]				TEXT_IN_1 [31:0]				TEXT_IN_2 [31:0]				TEXT_IN_3 [31:0]			
0xC	0xD	0xE	0xF	0x8	0x9	0xA	0xB	0x4	0x5	0x6	0x7	0x0	0x1	0x2	0x3

Регистр базового адреса дескриптора DMA (**BASE_DESCRIPTOR**)

ADDRESS – Адрес, который указывает на первый дескриптор DMA в цепочке, который будет прочитан из памяти после записи 1 в бит START регистра DMA_CONTROL

Регистр активного дескриптора DMA (**CURRENT_DESCRIPTOR**)

ADDRESS – Адрес, который указывает на текущий выполняемый дескриптор. При записи 1 в бит START регистра DMA_CONTROL в данный регистр попадает значение из регистра BASE_DESCRIPTOR. Если значение бита LAST слова управления DMA текущего дескриптора сброшен, то при успешном завершении текущего дескриптора в CURRENT_DESCRIPTOR заносится значение NEXT_DESCRIPTOR

Регистр следующего дескриптора DMA (**NEXT_DESCRIPTOR**)

ADDRESS – Адрес, который указывает на следующий дескриптор цепочки. Значение регистра будет сброшено, когда в слове управления текущего дескриптора DMA будет установлен бит LAST

Из дескрипторов может быть составлен однонаправленный список (цепочка дескрипторов), который будет определять параметры выполняемых модулем действий. Начало цепочки задается значением регистра BASE_DESCRIPTOR. Ниже в виде таблиц приведена структура дескриптора данных DMA.

Таблица 16.2 – Структура дескриптора данных DMA

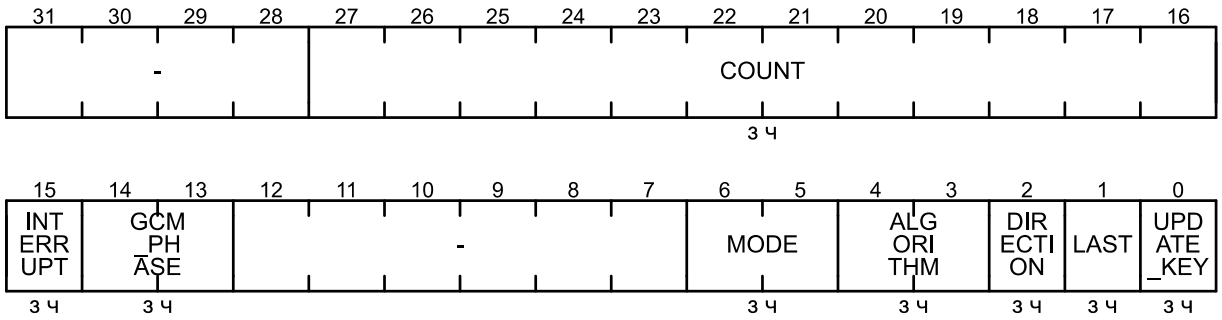
Индекс	Смещение адреса	Слова дескриптора
0	0x0	Слово управления DMA
1	0x4	32-битовый стартовый адрес источника
2	0x8	32-битовый стартовый адрес назначения
3	0xC	Следующий дескриптор данных DMA (32-битовый адрес, указывающий на следующий дескриптор данных)

Адреса источника данных должны быть выровнены по границе 32-разрядного слова. Бит LAST в управляющем слове дескриптора маркирует последний дескриптор цепочки. Если он сброшен, то адрес следующего дескриптора данных должен быть выровнен по границе в 4 слова. Если бит LAST установлен, последнее слово дескриптора не будет считано, а значение регистра NEXT_DESCRIPTOR будет сброшено.

В фазах GCM_INIT и GCM_HEADER режима GALOIS_COUNTER_MODE значение слова стартового адреса назначения игнорируется, поскольку результат их выполнения не подразумевает записи какого-либо результата в память. В фазе GCM_INIT режима GALOIS_COUNTER_MODE также будет проигнорировано значение слова стартового адреса источника, поскольку единственными изменяемыми входными данными является секретный ключ, хранимый в регистрах KEY_*.

Стоит отметить, что при считывании валидного значения слова управления DMA будут перезаписаны поля DIRECTION, ALGORITHM, MODE, GCM_PHASE регистра CONTROL.

Таблица 16.3 –Формат слова управления DMA



Поле	Биты	Описание
COUNT	27-16	Количество обрабатываемых блоков входного текста COUNT + 1 (размер блока зависит от значения поля TYPE: 64 бит для шифра Магма, 128 бит для остальных). В режиме GALOIS_COUNTER_MODE в фазах GCM_INIT и GCM_LAST_BLOCK данное поле должно иметь нулевое значение, в противном случае дескриптор будет считаться некорректным.
INTERRUPT	15	Разрешение генерации прерывания при завершении выполнения дескриптора.
GCM_PHASE	14-13	Фаза обработки пакета в режиме GALOIS_COUNTER_MODE. Если режим выполнения криптографической операции не GALOIS_COUNTER_MODE, то поле должно иметь нулевое значение (GCM_INIT), в противном случае дескриптор будет считаться некорректным.
		0 GCM_INIT

		1	GCM_HEADER
		2	GCM_PAYLOAD
		3	GCM_LAST_BLOCK
MODE	6-5	Выбор режима выполнения криптографической операции. Использование GALOIS_COUNTER_MODE в сочетании с криптографическим алгоритмом MAGMA не поддерживается, дескриптор будет считаться некорректным.	
		0	ELECTRONIC_CODEBOOK (ECB)
		1	CIPHER_BLOCK_CHAINING (CBC)
		2	COUNTER_MODE (CTR)
		3	GALOIS_COUNTER_MODE (GCM)
ALGORITHM	4-3	Тип криптографического алгоритма	
		0	AES-128
		1	AES-256
		2	Магма
		3	Кузнечик
DIRECTION	2	Режим работы	
		0	Шифрование (encryption)
		1	Дешифрование (decryption)
LAST	1	Установленный бит указывает на то, что данный дескриптор является последним в цепочке дескрипторов	
UPDATE_ KEY	0	После считывания корректного дескриптора данных DMA принудительно выполнить обновление раундовых ключей шифрования на основе текущего значения регистров {KEY_0, ..., KEY_7} или {KEY_0, ..., KEY_3} для алгоритма AES-128.	
-	31-28, 12-7	Зарезервированные биты. Должны быть равны 0, в противном случае дескриптор будет считаться некорректным	

Таблица 16.4 – Формат 32-битового стартового адреса источника (Source_Address)

Поле	Биты	Описание
ADDRESS	31-2	Старшие биты адреса, который указывает на начало массива слов входных данных
-	1-0	Младшие биты адреса. Должны быть равны 0, в противном случае дескриптор будет считаться некорректным

Таблица 16.5 – Формат 32-битового стартового адреса назначения (Destination_Address)

Поле	Биты	Описание
ADDRESS	31-2	Старшие биты адреса, который указывает на начало участка памяти, в который будет сохраняться результат выполнения криптографической операции
-	1-0	Младшие биты адреса. Должны быть равны 0, в противном случае дескриптор будет считаться некорректным

Таблица 16.6 – Формат адреса следующего дескриптора данных DMA (NEXT_DESCRIPTOR)

Поле	Биты	Описание
ADDRESS	31-4	Старшие биты адреса, который указывает на следующий дескриптор в цепочке. Если в слове управления DMA установлен бит LAST, то слово NEXT_DESCRIPTOR не будет прочитано
-	3-0	Младшие биты адреса. Если в слове управления DMA не установлен бит LAST, данные биты должны быть равны 0, в противном случае дескриптор будет считаться некорректным

В процессе чтения дескриптора производится анализ значений полей очередного слова дескриптора. Если оно признано некорректным, то работа поточного режима прекращается. Устанавливается бит DONE регистра IRQ, если был установлен соответствующий бит в регистре IRQ_ENABLE. В поле BAD_DESCRIPTOR регистра STATUS устанавливается значение, которое указывает на слово с недопустимым значением. Устанавливается бит DMA_FAIL регистра IRQ, если это разрешено через регистр IRQ_ENABLE.

Причины получения ненулевого значения STATUS.BAD_DESCR сведены в таблицу, приведенную ниже.

Таблица 16.7 – Причины получения ненулевого значения STATUS.BAD_DESCR

Индекс слова	Причины некорректности слова дескриптора
0	Обнаружен один или несколько ненулевых битов на позициях с 7 по 12 или с 28 по 31.
	Обнаружено неподдерживаемое сочетание значений полей MODE и ALGORITHM (GALOIS_COUNTER_MODE в паре с криптографическим алгоритмом MAGMA).
	Обнаружено ненулевое значение в поле GCM_PHASE, когда режим криптографической операции отличается от GALOIS_COUNTER_MODE.
	Обнаружено ненулевое значение в поле COUNT, когда MODE соответствует GALOIS_COUNTER_MODE, а значение GCM_PHASE – GCM_INIT или GCM_LAST_BLOCK.
1	Обнаружен один или несколько ненулевых битов на позициях с 0 по 1.
2	Обнаружен один или несколько ненулевых битов на позициях с 0 по 1.
3	Бит LAST слова управления DMA сброшен и обнаружен один или несколько ненулевых битов на позициях с 0 по 3.

К нештатному завершению выполнения цепочки дескрипторов может привести получение ответа ERROR на доступ чтения по шине AHB слова дескриптора, чтения входных блоков текста или записи выходных блоков текста. О данном факте будет свидетельствовать соответствующее значение поля AHB_ERROR регистра STATUS. Генерация прерываний аналогична таковой для ненулевого значения BAD_DESCRIPTOR.

Алгоритм функционирования поточного режима работы

Ниже приведен алгоритм работы машины конечных состояний, которая отвечает за реализацию поточного режима работы:

1 Ожидание записи DMA_CONTROL.START = 1;

2 В качестве текущего адреса дескриптора (отображается в CURRENT_DESCRIPTOR.ADDRESS) устанавливается значение из регистра BASE_DESCRIPTOR.ADDRESS;

3 Чтение слова управления DMA. Если по шине AHB получен ответ ERROR или слово признано некорректным, то происходит установка соответствующего статуса, если соответствующие прерывания разрешены в регистре IRQ_ENABLE, генерируются прерывания DMA_FAIL и DONE, происходит переход к пункту 1;

4 Значения полей DIRECTION, ALGORITHM, MODE, GCM_PHASE заносятся в регистр CONTROL. В STATUS.DMA_LAST отображается состояние бита LAST. В IRQ_ENABLE.DMA_DONE – значение бита INTERRUPT;

5 Чтение слова стартового адреса источника. Получение ответа ERROR или обнаружение ошибки в слове обрабатывается аналогично пункту 3;

6 Чтение слова стартового адреса назначения. Получение ответа ERROR или обнаружение ошибки в слове обрабатывается аналогично пункту 3;

7 Данный пункт выполняется, только если был сброшен бит LAST слова управления DMA. Чтение слова адреса следующего дескриптора (отображается в NEXT_DESCRIPTOR.ADDRESS). Получение ответа ERROR или обнаружение ошибки в слове обрабатывается аналогично пункту 3;

8 Если для текущего криптографического алгоритма нет рассчитанных раундовых ключей, активируется их расчет на основе текущего значения из регистров KEY_*;

9 Данный пункт не выполняется, если CONTROL.MODE равен GALOIS_COUNTER_MODE, а CONTROL.GCM_PHASE соответствует GCM_INIT. Чтение блока входного текста. Четыре или два слова, если выбран алгоритм «Магма», заносятся в регистры TEXT_IN_* (с учетом WORDS_SWAP и BYTES_SWAP регистра DMA_CONTROL). Получение ответа ERROR обрабатывается аналогично пункту 3;

10 Ожидание готовности раундовых ключей;

11 Запуск выполнения шифрования/дешифрования;

12 Ожидание готовности выходных данных;

13 Данный пункт не выполняется, если CONTROL.MODE равен GALOIS_COUNTER_MODE, а CONTROL.GCM_PHASE соответствует GCM_INIT или GCM_HEADER. В остальных случаях выполняется запись в память блока выходного текста (содержимое TEXT_OUT_*) или аутентификационного тега (содержимое GCM_Tag_*). Тег является результирующими данными, если CONTROL.MODE равен GALOIS_COUNTER_MODE, а CONTROL.GCM_PHASE соответствует GCM_LAST_BLOCK. В обоих случаях данные сохраняются с учетом WORDS_SWAP и BYTES_SWAP регистра DMA_CONTROL. Получение ответа ERROR обрабатывается аналогично пункту 3;

14 Если CONTROL.MODE не равен ELECTRONIC_CODEBOOK, выполняем обновление IV_*;

15 Если значение счетчика обработанных блоков STATUS.DMA_COUNTER меньше значения поля COUNT слова управления DMA, то переходим к пункту 9;

16 Если в слове управления DMA бит LAST был сброшен, генерируется прерывание DMA_DONE, если был установлен бит INTERRUPT в слове управления DMA, заносим в CURRENT_DESCRIPTOR значение из NEXT_DESCRIPTOR, возвращаемся к пункту 3;

17 Если в слове управления DMA бит LAST был установлен, генерируется прерывание DMA_DONE, если был установлен бит INTERRUPT в слове управления DMA, генерируется прерывание DONE, если оно разрешено в регистре IRQ_ENABLE, происходит переход к пункту 1.

16.5 Экстренное прекращение выполнения операции

Для организации экстренного прекращения нужно записать в поле CODE регистра TERMINATE значение 0xD0. На следующий такт после такой записи будет сформирован сигнал сброса состояния модулей криптографических операций и хранилища раундовых ключей. Данное событие вернет биты READY, KEYS_READY, TEXT_OUT_VALID, KEYS_STORED, DMA_ACTIVE, DMA_COUNTER и DMA_LAST регистра STATUS к значениям по умолчанию. Стоит отметить, что бит IRQ_PENDING, поля BAD_DESCRIPTOR и AHB_ERROR не изменят своего значения.

Регистр TERMINATE

CODE – Запись в данное поле значения 0xD0 приведет к сбросу внутреннего состояния модулей криптографических операций и хранилища раундовых ключей. Запись других значений будет проигнорирована. При чтении всегда возвращает 0.

16.6 Производительность

Таблица 16.8 – Количество тактов на операцию в режиме работы с одним блоком

Операция	Длительность в тактах			
	AES-128	AES-256	Magma	Kuznechik
Расчет раундовых ключей	10 + 1	14 + 1	1 + 1	32 + 1
Шифрование/дешифрование	10 + 1	14 + 1	8 + 1	10 + 1
Шифрование/дешифрование с новым ключом	20 + 1	28 + 1	9 + 1	42 + 1

В поточном режиме работы к длительности операции прибавляется задержка на считывание дескриптора, считывание входных блоков текста и запись выходных блоков текста.

В данной реализации на выполнение умножения в поле Галуа (GF(2^{128})mult), используемого в криптографическом режиме GALOIS_COUNTER_MODE, тратится 8 тактов. Длительность операций отличается для разных фаз:

1 В фазе GCM_INIT определяется длительностью шифрования выбранного алгоритма;

2 В фазе GCM_HEADER равняется длительности GF(2128)mult плюс 1 такт;

3 В фазе GCM_PAYLOAD:

а) при выполнении шифрования определяется суммой длительности шифрования выбранного алгоритма, длительности GF(2128)mult плюс 1 такт.

б) при выполнении дешифрования определяется длительностью шифрования выбранного алгоритма. Это обусловлено тем, что умножение на блок шифрованного текста, находящегося в TEXT_IN_*, на хэш-ключ и шифрование IV_* выполняются параллельно.

4 В фазе GCM_LAST_BLOCK определяется длительностью шифрования выбранного алгоритма. Это происходит, поскольку умножение блока длины на хэш-ключ и шифрование значения на основе IV_* (вместо IV_3 используется 0x00000001) выполняются параллельно.

17 Блок генератора случайных чисел

Аппаратный генератор случайных чисел – это устройство, которое генерирует последовательность случайных 32-разрядных чисел на основе измеряемых, хаотически изменяющихся, параметров протекающего физического процесса.

Криптографические модули могут использовать Генераторы Случайных Чисел (ГСЧ). В криптографии случайные последовательности играют определяющую роль. Они используются, в частности, для получения ключевой последовательности используемого алгоритма шифрования, для генерации гаммы поточных шифров, а также для выработки векторов инициализации (блочных шифров).

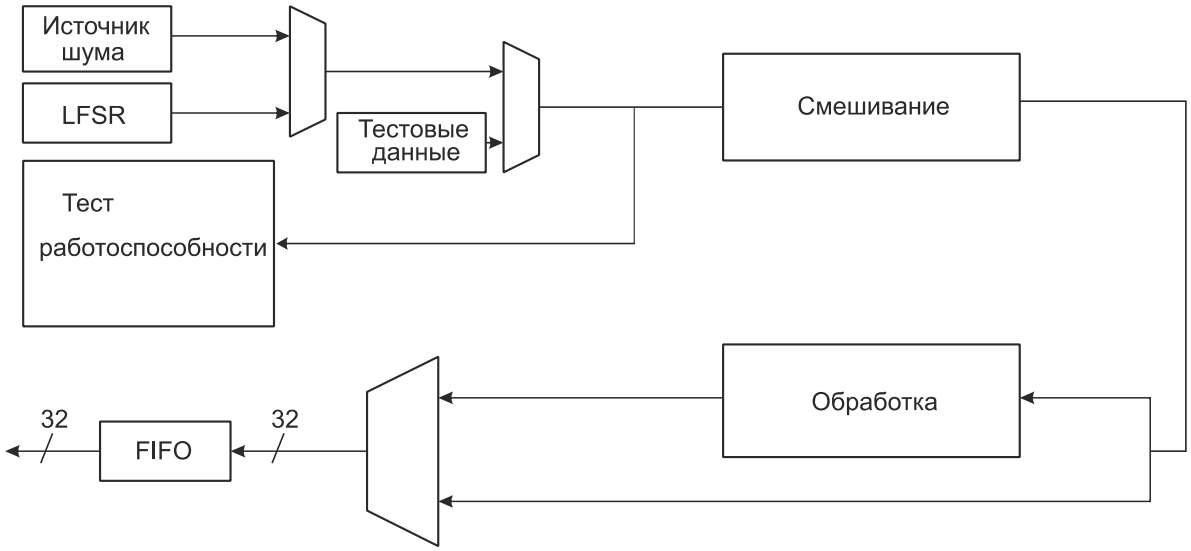


Рисунок 17.1 – Блок-диаграмма ИГСЧ

17.1 Источник энтропии

Фазовое дрожание цифрового сигнала данных (джиттер от англ. jitter) — нежелательные фазовые и/или частотные случайные отклонения передаваемого сигнала. Возникают вследствие нестабильности задающего генератора, изменений параметров линии передачи во времени и различной скорости распространения частотных составляющих одного и того же сигнала. Поскольку фазовое дрожание зависит от различных факторов, некоторые из которых полностью случайны, оно может быть использовано как источник случайных чисел.

В ГСЧ на базе такого явления сравниваются случайные задержки прохождения сигнала через кольцевые генераторы. Простейший кольцевой генератор состоит из нечетного числа инверторов, соединенных последовательно, при этом выход последнего соединен с входом первого инвертора, образуя линию обратной связи. Частота колебания такого генератора определяется суммой задержек всех его инверторов, это время зависит от множества параметров, включающих в себя тепловой шум в проводниках и полупроводниках и помехи в источниках питания.

Данная реализация Истинного Генератора Случайных Чисел (ИГСЧ) основана на использовании 32 отдельных колец генератора, каждое кольцо состоит из 5 инверторов. Выход аналогового сигнала преобразуется в цифровой сигнал путем выборки значений на постоянной тактовой частоте.

Структуру, реализованную в данном блоке, очень трудно имитировать. С помощью задания конфигурации битов возможно заставить данную структуру работать в качестве Псевдо-Случайного Генератора (ПСГ) для определенных задач моделирования.

Выборку из источника шума можно делать напрямую, в обход функции обработки (управляющий бит CONDBYPASS = 1).

17.2 Процесс дискретизации

Если выходной сигнал высокочастотного кольцевого генератора (RO) дискретизируется низкочастотным тактовым сигналом, то отобранный бит является случайным. Должно выполняться следующее утверждение $T_{OSC} \gg T_{RO}$, где T_{OSC} – период, в течение которого сигнал с кольцевого генератора (RO) находится в свободном режиме (независимо генерируется). Как показано на рисунках 17.2 и 17.3, поддерживаются два режима генерации:

1) В горячем режиме (регистр COOLDPERIOD = 0), сигнал с кольцевых генераторов (RO) свободно генерируется, пока FIFO заполняется. Результат берётся каждые D_{OSC} тактового сигнала, где D_{OSC} можно настроить с помощью регистра SAMPERIOD. При этом период колебаний $T_{OSC}=D_{OSC}*T_{sysclk}$.

2) В холодном режиме сигналы с кольцевых генераторов (RO) – запускаются установкой в значение «0» на время Drst тактового сигнала после каждой выборки результата, где Drst можно настроить через интерфейс APB (регистр COOLDPERIOD).

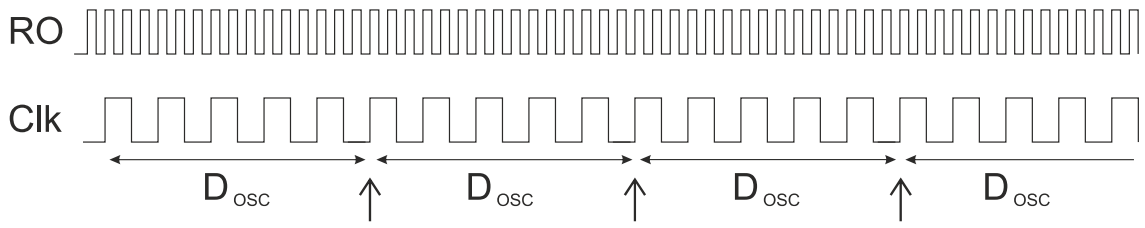


Рисунок 17.2 − Процесс выборки – горячий режим

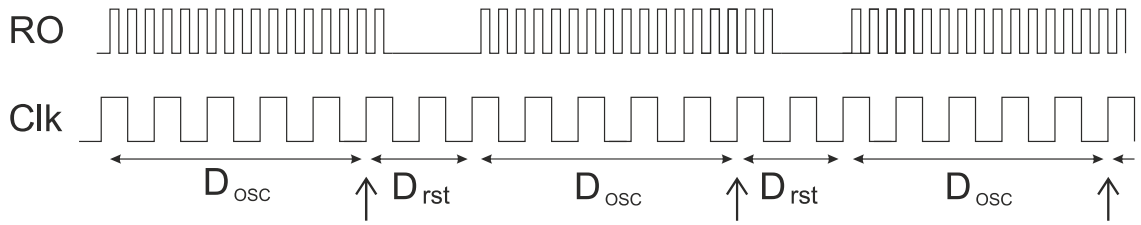


Рисунок 17.3 − Процесс выборки – холодный режим

В действительности, идеальное значение D_{OSC} обычно гораздо больше, чем в примере, указанном на рисунке выше. Также типичное время удержания T_{OSC} >> Trst.

Горячий режим, теоретически, может иметь некоторые временные корреляции, которые в холодном режиме не проявятся, поскольку он «сбрасывается» перед генерацией каждого случайного значения. В горячем режиме дополнительная эффективность может быть получена с помощью регулировки параметра T_{OSC} (регистр SAMPERIOD); тесты на работоспособность сработают в любом случае при обнаружении ошибок.

Холодный режим может быть более подверженным сторонним атакам, поскольку остановки генерации предоставляют точки синхронизации, то есть злоумышленнику легче узнать, когда генерируются данные. Горячий режим не полностью защищен от такой проблемы, поскольку AES генерирует сигнатуру мощности, которая может использоваться в качестве точек синхронизации. Горячий режим имеет лучшую пропускную способность.

17.3 Непрерывное тестирование

Выборки из источника энтропии постоянно контролируются с помощью тестов на повторения и пропорциональных тестов, проходящих параллельно. Опционально также могут использоваться тесты на корреляцию и автокорреляцию.

17.4 Тест на количество повторений

В ИГСЧ реализован тест подсчета количества повторений. Допустимый уровень ложноположительных результатов установлен на значении $\alpha = 2^{-20}$. Источник случайного шума выдаёт недвоичную выборку. Для минимальной энтропии $H = 8*0,5 = 4.0$, допустимое пороговое значение повторений по умолчанию равно $C = 1+(-\log2 (\alpha)/H) = 6$.

Пороговое значение программируется в регистре REPTSTCUTOFF. Тест может быть отключен с помощью бита REPTSTDIS в регистре управления (CR). О провале теста повторений сообщается с помощью битов REPTSTFAILSH в регистре статуса (STAT).

17.5 Тест на подсчет пропорций (окно 512-недвоичных выборок)

Тест на количество пропорций, был реализован с размером окна W=512 недвоичных выборок. Допустимый уровень ложноположительных результатов – $\alpha = 2^{-20}$. Источник случайного шума выдаёт однобитную выборку, и после выбранного метода смешивания создаются недвоичные выборки. Для минимальной энтропии $H = 8 * 0.5 = 4.0$, допустимое пороговое значение пропорций по умолчанию равно $C = 1+\text{CRITBINOM}(W, 2^{-H},1- \alpha) = 63$.

Пороговое значение программируется в регистре PROPTESTCUTOFF. Тест может быть отключен с помощью бита PROPTSTDIS в регистре управления (CR). О провале теста на подсчет пропорций сообщается с помощью битов PROPTESTFAIL в регистре статуса (STAT).

17.6 Тест корреляции

Когда два кольцевых генератора связаны между собой с помощью внешней синхронизации, их битовые потоки становятся коррелированными. Тест на подсчет корреляции может обнаружить это явление. Он использует мультиплексирование с временным разделением, и одновременно обрабатывает одну пару кольцевых генераторов. Окно размером W = 128 выборок используются с пороговыми значениями, определяемыми в регистрах CORRTESTCUTOFF. Отдельные проверки со смещением (или все проверки) могут быть отключены с помощью битов CORRTSTDIS в регистре управления (CR).

В каждом окне проверяется одна пара кольцевых генераторов (RO). Все возможные комбинации различных индексов генераторов (RO) будут проверены в порядке возрастания (1-0,...,7-0,0-1,2-1,...,7-1,...). Результаты проверок отключенных кольцевых генераторов будут замаскированы. Пара кольцевых генераторов, не прошедших тест, то есть их индексы будут установлены как битовые индексы в регистре CORRTESTFAILED.

17.7 Тест на подсчёт авто-корреляции

Когда кольцевой генератор непрерывно дискретизируется без накопления достаточного дрожания (jitter), создаваемый битовый поток демонстрирует периодическое поведение. Тест на подсчет автокорреляции способен обнаружить это поведение. Он использует мультиплексирование с временным разделением и поддерживает один кольцевой генератор. Окно размером W = 128 выборок используется с пороговыми значениями, определяемыми в регистрах AUTOCORRTESTCUTOFF. Отсрочки для отдельных проверок (или все проверки) могут быть отключены с помощью битов ACORRTSTDIS в регистре управления (CR).

В каждом окне проверяется один кольцевой генератор (RO). Все кольцевые генераторы поиндексно будут проверены в порядке возрастания. Результаты отключенных

генераторов будут замаскированы. Генератор не прошедший тест, то есть его индекс, будет установлен как битовый индекс в регистре AUTOCORRTESTFAILED.

17.8 Тест при запуске

Тест на количество повторений и тест на подсчет пропорций (окно из 512 выборок) должны быть пройдены, прежде чем какие-либо данные будут считаны из FIFO.

17.9 Смешивание XOR

Наиболее эффективным способом получения случайных данных, с самой низкой задержкой, является объединение выходных данных кольцевых генераторов (RO). Выходные данные кольцевых генераторов могут объединяться по методу XOR (Исключающего Или), чтобы увеличить уровень энтропии. Операции XOR могут быть выполнены до или после выборки, но в данном ИГСЧ сначала проводится выборка, а затем XOR.

Имеются два уровня смешивания XOR. При выборе XOR1 в качестве метода смешивания будет применено сокращение с 32 до 4 бит. Дальнейшее сокращение возможно с помощью метода смешивания XOR2 как с 4 до 1 бита.

Последний возможный метод смешивания – это использование алгоритма исключения систематической ошибки Фон-Неймана. Он опирается на предположение – если входные биты являются независимыми и идентично распределенными, выходные, вероятно, распределены равномерно.

17.10 Обработка

Функция обработки берет значение NbOfBlocks *128 бит от источника энтропии в качестве входных данных и генерирует n_{out} = 128 бит в качестве выходных. Предполагая, что минимальная энтропия H = 0.5 и количество блоков NbOfBlocks= 4, входная энтропия составляет h_{in} = 512 * 0.5 = 256 бит. Можно считать, что на выходе функции обработки имеем 128 бит полной энтропии. Функция обработки может быть протестирована путем отправки известных данных (используя TESTDATA с установкой контрольного бита TESTEN в значение 1).

17.11 FIFO

Выходные данные функции обработки разбиваются на 32-разрядные слова, перед сохранением в FIFO (программируемый размер 256 слов *32 бита).

Независимо от того пропущена ли функция обработки, и необработанные шумовые выборки, и обработанные выборки будут записываться в FIFO, если бит старта/включения установлен, а КА находится не в состоянии «Error». Тесты работоспособности не препятствуют записи в FIFO, даже на этапе запуска, если установлен FIFOFILLST. Пользователь самостоятельно должен определить, являются ли данные в FIFO действительными. 32-разрядные случайные числа доступны через интерфейс APB, в диапазоне адресов [0x80 - 0xFC].

17.12 LFSR

Для проверки корректности данных, произведенных ИГСЧ, выборки могут быть получены из регистра LFSR (сдвиговый регистр линейной обратной связи) вместо источника шума. Эта функция доступна через регистры.

17.13 Прерывания

ИГСЧ имеет два источника прерываний.

Таблица 17.1 - Источники прерывания

Имя	Описание
FIFOFULLINT	Этот бит прерывания устанавливается, когда FIFO заполняется. Он сбрасывается при записи регистра FIFOLEV или при записи ‘0’ в 7 бит регистра STAT (FIFOFULL) или при программном сбросе установкой 8-го бита SOFTRST регистра CR.
ANYTESTFAILINT	Этот бит прерывания устанавливается при обнаружении ошибки в любом из тестов работоспособности. Он сбрасывается при записи ‘0’ в 6 бит регистра STAT (ANYTESTFAIL) или при программном сбросе установкой 8-го бита SOFTRST регистра CR.

Для каждого источника прерывания имеется один бит в регистре состояния (STAT) и один бит разрешения в регистре управления (CR). Состояния в регистре состояния не зависят от значений битов разрешения прерывания (TFAILINTEN, FFULLINTEN) в регистре управления. Биты разрешения прерывания используются только для генерации внешнего сигнала прерывания IRQ. Если какое-то прерывание активно и включено, то формируется сигнал IRQ высокого уровня.

В контроллер PLIC поступает одна линия прерывания TRNGINT от модуля ИГСЧ объединяющая в себе по принципу «логического ИЛИ» оба сигнала прерывания FIFOFULLINT и ANYTESTFAILINT.

17.14 Программирование ИГСЧ

Управление Конечным Автоматом

ИГСЧ управляется Конечным Автоматом (КА), который обрабатывает все низкоуровневые взаимодействия между различными внутренними компонентами, указанными на рис.17.1. Во время нормальной работы КА обеспечивает управление источником энтропии. Диаграмма состояния КА показана на рисунке 17.4.

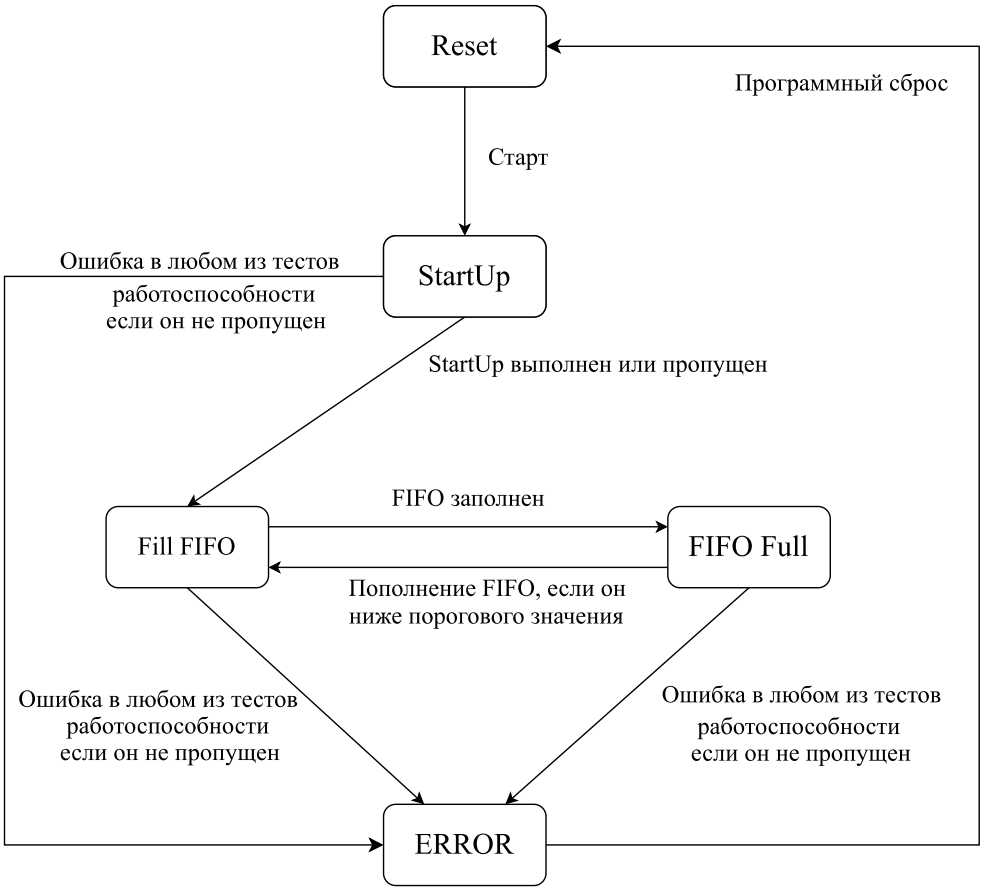


Рисунок 17.4 − Диаграмма состояний КА ИГСЧ

Диаграмма состояний не показывает, что аппаратный или программный сброс приводит к тому, что КА возвращает ИГСЧ обратно в состояние “Reset”.

Порядок действий при проверке функции обработки:

1. Запустить программный сброс;
2. Записать управляющий регистр;

Убедиться, что COUNTBLOCK = 4 и TESTEN = 1;

3. Записать регистр ключей;
4. Записать данные в регистр TESTDATA (после каждого ввода данных подождать, пока флаг занятости DATABUSYTEST не сбросится);
5. Прочитать результат из FIFO.

В таблице ниже приведен пример данных.

Таблица 17.2 – Тест с известным ответом для функции обработки

	Формат 128 бит	Формат 32 бита APB
Ключ	0x2B7E151628AED2A6ABF7158809CF4F3C	0x16157E2B 0xA6D2AE28 0x8815F7AB 0x3C4FCF09
Входной сигнал	0x6BC0BCE12A459991E134741A7F9E1925	0xE1BCC06B 0x9199452A 0x1A7434E1 0x25199E7F
	0xAE2D8A571E03AC9C9EB76FAC45AF8E51	0x578A2DAE 0x9CAC031E 0xAC6FB79E 0x518EAF45
	0x30C81C46A35CE411E5FBC1191A0A52EF	0x461CC830 0x11E45CA3 0x19C1FBE5 0xEF520A1A
	0xF69F2445DF4F9B17AD2B417BE66C3710	0x45249FF6 0x179B4FDF 0x7B412BAD 0x10376CE6
Ожидаемый результат	0x3FF1CAA1681FAC09120ECA307586E1A7	0xA1CAF13F 0x09AC1F68 0x30CA0E12 0xA7E18675

Порядок действий при проверке источника энтропии:

1. Запустить программный сброс;
2. Записать управляющий регистр, убедившись, что ни один из тестов работоспособности не будет пропущен, и установить бит запуска/включения;
3. Дождаться пропадания состояний “Сброс”, “Запуск” или “Ошибка”. В этом случае все тесты пройдены, и источник энтропии работает правильно;
4. Если состояние “Ошибка” не пропадает, выбрать другие значения для SAMPERIOD и WARMPERIOD и вернуться к шагу 1.

Порядок действий при программировании ключа:

1. Проверить регистр FIFOLEV, чтобы узнать количество сгенерированных случайных чисел, или дождаться FIFOFULL IRQ;
2. Когда FIFOLEV достигнет ожидаемого значения или когда будет получен FIFOFULL IRQ, считать случайные числа через интерфейс APB;
3. Используйте 4x32-битных случайных значения для программирования случайного ключа;
4. Применить программный сброс, для очистки FIFO.

Порядок действий для получения выборок:

1. Проверить регистр FIFOLEV, чтобы узнать количество сгенерированных случайных чисел, или дождаться FIFOFULL IRQ;
2. Когда FIFOLEV достигнет ожидаемого значения или когда будет получен FIFOFULL IRQ, считать случайные числа через интерфейс APB.

Порядок действий для изменения конфигурации:

1 Отключить ИГСЧ (сбросить бит [0] – START – управляющего регистра CR);

2 Настройте регистр управления (CR);

3 Выполнить сброс пути к данным, чтобы получить чистую начальную точку (установить бит [8] – SOFTRST – управляющего регистра);

4 Включить ИГСЧ обратно (установите бит [0] – START – управляющего регистра).

Формат данных – порядок байтов

Все криптографические данные обрабатывают в соответствии с форматом big-endian (стандарт AES). Это означает, что первый байт (младший адрес) данных является самым значимым байтом (MSB).

Шина APB использует формат little endian. Это означает, что наименее значимый байт (LSB) хранится по самому младшему адресу.

Например, 128-битный блок D[A] = 0x00112233445566778899AABBCCDDEEFF, хранящийся по адресу A (кратному 4), отображается на шине APB в виде следующих 32-разрядных слов:

- D[A+0] = 0x33221100;
- D[A+4] = 0x77665544;
- D[A+8] = 0xBBAA9988;
- D[A+12] = 0xFFDDEECC.

17.15 Производительность

Количество циклов (тактового сигнала), необходимых ИГСЧ для генерации данных, можно оценить по следующей формуле

$$Nbcycles \approx 1024 \times SAMPERIOD + WARMPERIOD, \quad (17.1)$$

где 1024 – начальное количество выборок;

SAMPERIOD и WARMPERIOD – два значения конфигурации (задаются с помощью одноименных регистров).

При типичном использовании, если установить SAMPERIOD = 30 и WARMPERIOD = 512, с момента установки управляющего регистра до появления первого случайного слова в выходном буфере FIFO потребуется приблизительно
1024 × 30 + 512 = 31 232 цикла.

18 Часы реального времени RTC

Часы реального времени RTC предназначены для отсчета времени в микроконтроллере. Данный блок продолжает работу после отключения питания на одном из выводов VCC1(вывод 12) или V_BAT, так как блок PMU_RTC запитан от выхода VAON модуля APC. Описание работы модуля APC представлено в подразделе 5.1.

Структура блока RTC представлена на рисунке 18.1.

В состав блока часов реального времени входят:

1) тактовый генератор, имеющий в своем составе:
   - низкопотребляющий осциллятор 32kHz со встроенными конденсаторами;
   - низкопотребляющий RC-генератор 32kHz;
   - выходной делитель тактового сигнала;
2) 32-разрядный счетчик времени (охватывает подсчет секунд до 136 лет);
3) 32-разрядный регистр сигнализации Alarm;
4) 16 32-разрядных регистров общего назначения;
5) 11-разрядный регистр цифровой подстройки частоты с разрешением 0,5 ppm для тактового выхода 1c;
6) 11-разрядный регистр подстройки частоты, получаемой от RC-генератора;
7) три входа мониторинга вскрытий AT_IN0-AT_IN2 с одним выходом генерации последовательности AT_OUT;
8) контроллер пробуждений с фиксацией событий от внешних выводов WAKEUP0 – WAKEUP2.

Блок RTC формирует тактовый сигнал lpclk для управления регуляторами напряжений (LDO0 и LDO1) и контроллером питания APC.

Длительность логического нуля сигнала lpclk составляет 30 мкс, а длительность логической единицы выбирается битом LS регистра RTC_CFG0 (15 мкс или 30 мкс). Включение сигнала lpclk осуществляется с помощью бита CD регистра RTC_CFG0.

Регистры блока RTC, в том числе TIME, ALARM, а также регистры конфигурации PMURTC сбрасываются только при включении микроконтроллера после полного отключения питания микроконтроллера по выводам VCC1 и V_BAT.

При первом включении питания, в связи с тем, что по умолчанию установлен бит ALARMEN регистра RTC_CFG1, и регистры RTC_TIME и RTC_ALARM имеют одинаковое нулевое значение, возникает событие ALARM. Поэтому, при разрешении прерывания блока PMU_RTC сразу же будет вызван соответствующий обработчик прерывания. Сброс запроса на прерывание PMU_RTC осуществляется записью единицы в бит IRQEVT регистра PMU_IRQEVT.

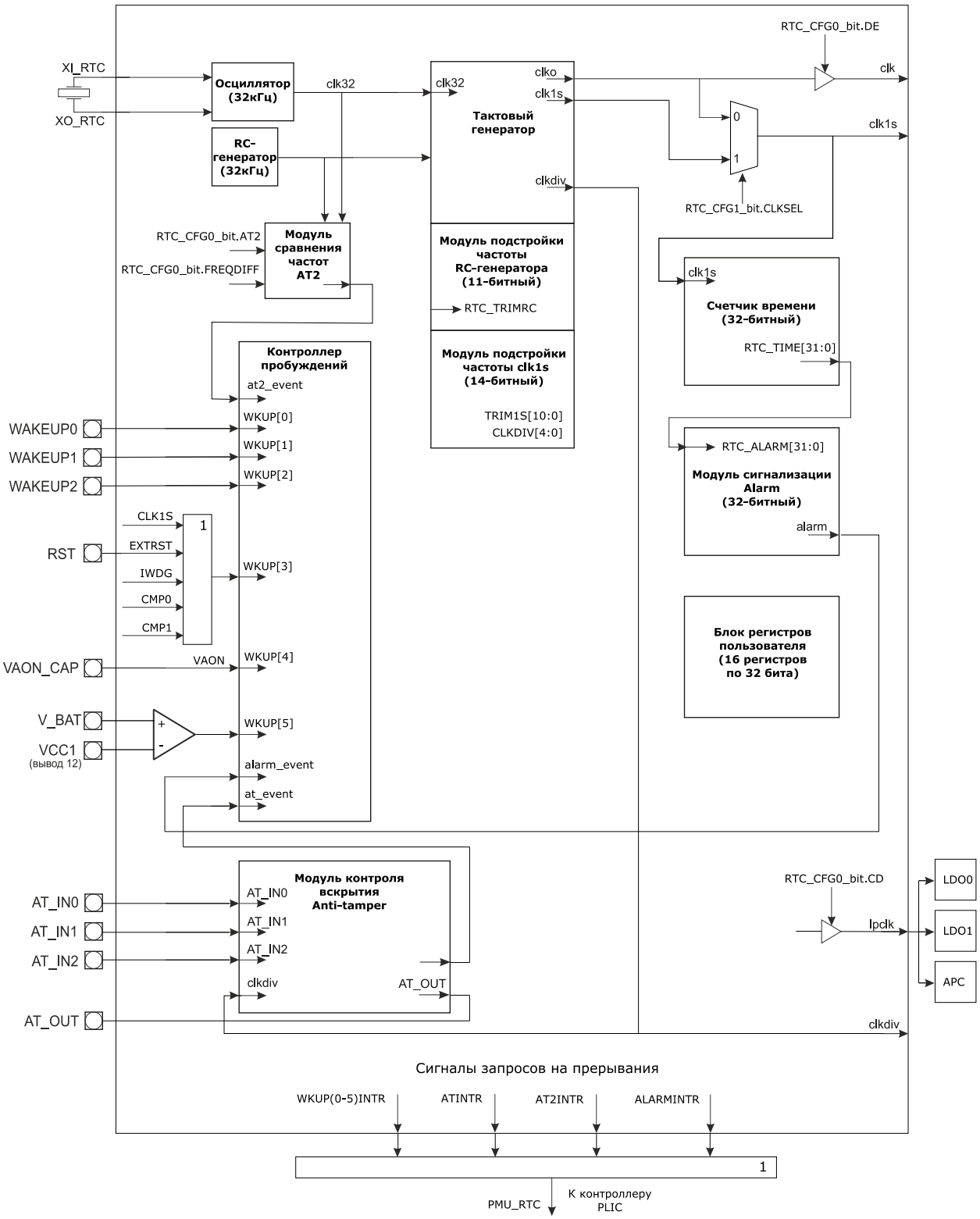


Рисунок 18.1 – Структура блока RTC

18.1 Особенности

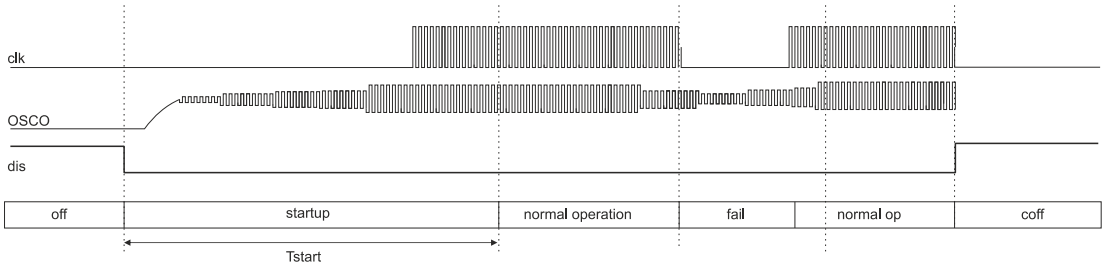


Рисунок 18.2 – Диаграмма старта осциллятора

Первый такт генератора соответствует падающему фронту такта clk32 - 32.768kHz. Отсчет клока производится по восходящему фронту 1c, clk1s. Сравнение падающего фронта 1 такта, clk1s. Сгенерированный первый такт внутреннего RC-генератора устанавливает все счетчики и регистры PMURTC в значения по умолчанию, когда выходное напряжение LDO_RTC повышается.

Включение контроллера осуществляется регуляторами, которые включают, сбрасывают и отключают сигналы. Сигнал оповещения обычно используется для определения режима работы: 1 – ассоциируется с высокой мощностью, 0 – работа на низкой мощности.

По умолчанию, при включении питания RTC или при обнаружении сигнала питания, устанавливается бит TIMEALARM регистра RTC_HISTORY и включаются оба регулятора (REGEN[1:0]=11b и REGDIS[1:0]=00b).

18.2 Контроллер пробуждения WAKE

Контроллер пробуждения содержит шесть источников событий WKUP[0] - WKUP[5], а также события от монитора вскрытия (нарушение целостности цепей AT_IN0 - AT_IN2, расхождение частот RC-генератора) и событие сигнализатора Alarm (совпадение значений регистров RTC_TIME и RTC_ALARM). Структурная схема контроллера пробуждения WAKE представлена на рисунке 18.3.

События WKUP[0] - WKUP[2] управляются внешними выводами микроконтроллера WAKEUP0 – WAKEUP2. Полярность регистрируемого события определяется битами WAKEPOL[2:0] регистра PMURTC->RTC_WAKECFG (Если соответствующий бит WAKEPOL сброшен, то активным для сигнала пробуждения считается высокий логический уровень сигнала. Если бит установлен, то низкий логический уровень). Отследить события WKUP[0]-WKUP[2], вызвавшие прерывание, можно в регистре PMURTC->RTC_HISTORY в битовых полях WAKE0-WAKE2. Контроллер Wake входит в состав блока RTC и подключен к регулятору LDO_RTC, поэтому при регистрации событий по выводам WAKEUP0 – WAKEUP2 высоким логическим уровнем является напряжение выше уровня 1,4 В (выходное напряжение регулятора LDO_RTC).

Событие WKUP[3] может происходить при формировании событий от следующих периферийных блоков, входящих в состав домена батарейного питания:

- CLK1S – событие периода тактового сигнала таймера REG_TIME. Если бит CLKSEL регистра RTC_CFG1 установлен, то частота тактового сигнала 1 Гц, если сброшен – то 32 КГц;
- EXTRST – сигнал внешнего сброса RST;
- IWDT – сигнал от независимого сторожевого таймера;
- CMP0 – сигнал триггера блока аналогового компаратора 0;
- CMP1 – сигнал триггера блока аналогового компаратора 1.

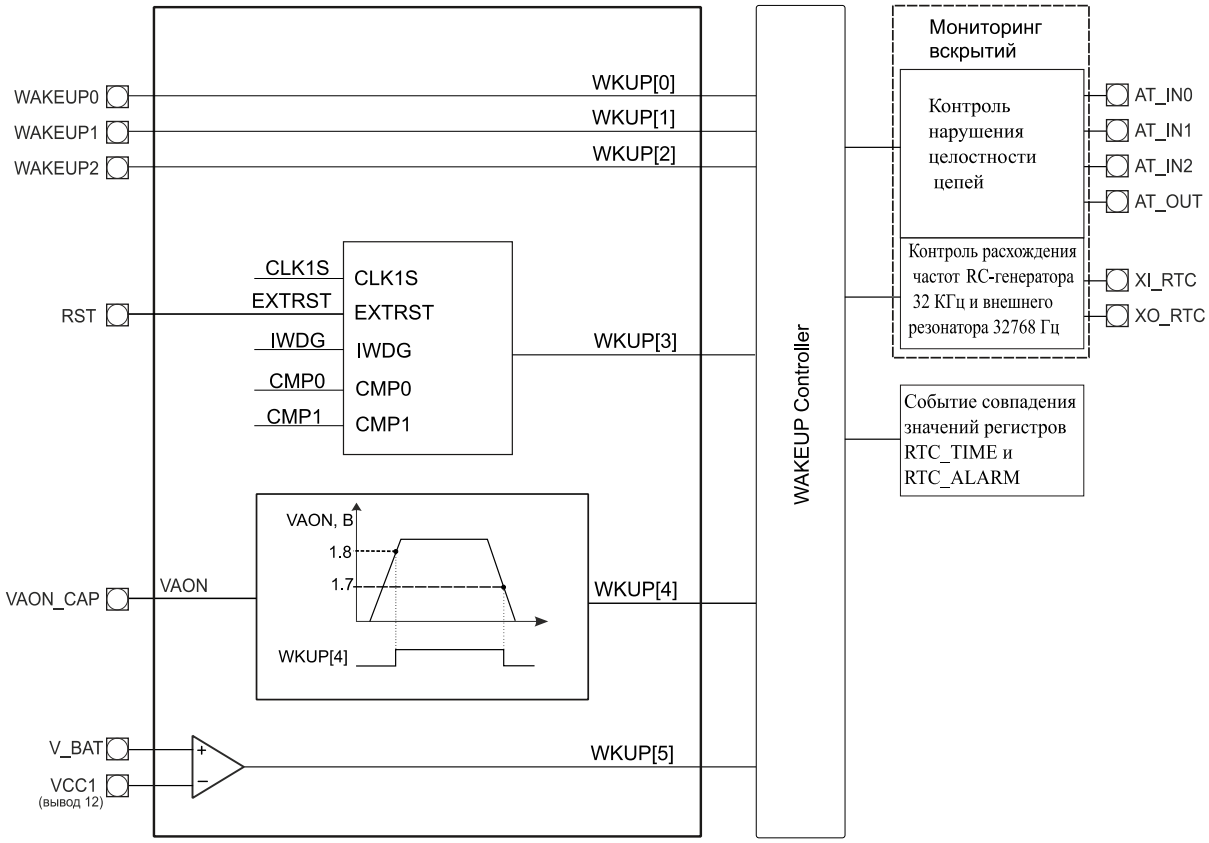


Рисунок 18.3 – Структурная схема контроллера пробуждения WAKE

Для того чтобы по событию WKUP[3] формировалось прерывание блока RTC и происходило пробуждение контроллера из режима сна, необходимо установить бит WAKEEN[3] регистра PMURTC->RTC_WAKECFG, при этом бит WAKEPOL[3] регистра PMURTC->RTC_WAKECFG должен быть сброшен. Выбор периферийных блоков, формирующих событие WKUP[3] осуществляется установкой соответствующих бит регистра PMURTC->PMU_WK3EN. Отследить событие WKUP[3], вызвавшее прерывание, можно в регистре PMURTC->RTC_HISTORY в битовом поле WKVBATPER. Определить, какие периферийные блоки вызвали событие можно в регистре статуса PMURTC->PMU_WK3STAT.

Сигнал события WKUP[4] устанавливается при уровне напряжения VAON 1,8 В при нарастании напряжения и сбрасывается при уровне напряжения VAON 1,7 В при снижении (см. рисунок 18.4).

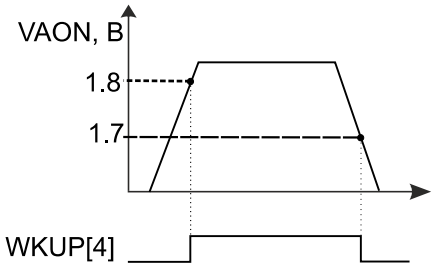


Рисунок 18.4 – Формирование сигнала WKUP[4]

Для того, чтобы по событию WKUP[4] формировалось прерывание блока RTC и происходило пробуждение контроллера из режима сна, необходимо установить бит

WAKEEN[4] регистра PMURTC->RTC_WAKECFG, при этом бит WAKEPOL[4] регистра PMURTC->RTC_WAKECFG должен быть сброшен. Статус события WKUP[4] отражается в битовом поле WKUVLO регистра PMURTC->RTC_HISTORY.

Сигнал события WKUP[5] представляет собой выход компаратора, к которому подключено напряжение V_BAT и входное напряжение APC – VCC1(вывод № 12). Так событие WKUP[5] будет установлено при условии V_BAT > VCC1(вывод № 12), т.е. при отсутствии основного напряжения и переключении на батарейное питание. Для того, чтобы по событию WKUP[5] формировалось прерывание блока RTC и происходило пробуждение контроллера из режима сна, необходимо установить бит WAKEEN[5] регистра PMURTC->RTC_WAKECFG, при этом бит WAKEPOL[5] регистра PMURTC->RTC_WAKECFG должен быть сброшен. Статус события WKUP[5] отражается в битовом поле WKVFVBAK регистра PMURTC->RTC_HISTORY.

События от монитора вскрытия также могут формировать прерывание блока RTC и пробуждать контроллер из режима сна. Для активации мониторинга входов AT_IN0, AT_IN1, AT_IN2 необходимо установить соответствующие биты TAMPER0EN – TAMPER2EN регистра PMURTC->RTC_WAKECFG.

В WAKECONFIG регистре можно настроить систему так, чтобы отдельные регуляторы оставались включенными, даже при сигнале оповещения = 0 или чтобы отдельные регуляторы оставались отключенными при сигнале оповещения = 1.

WKUP[4] (пониженное напряжение питания (VAON)) и WKUP[5] (сигнал от компаратора напряжений V_BAT и входного напряжения APC − VCC1(вывод № 12)) имеют дополнительную возможность настройки, поскольку они соответствуют событиям, связанным с ограниченным напряжением питания. В этом случае сигнал оповещения установлен, но уровень мощности может быть уменьшен, поскольку регуляторы, отключенные сигналом оповещения = 1, переопределяются при включении [4] или включении [5].

Биты LP[1:0] или ULP[1:0] регистра RTC_CFG1 могут быть установлены для принудительного переключения регуляторов в режим малой мощности или сверхнизкой мощности соответственно. Вход в эти режимы устанавливается сбросом бита ALARMRST регистра RTC_CFG1.

Для корректной регистрации событий в регистре RTC_HISTORY после перехода в режим RUN из режимов STOP (LP_STOP, ULP_STOP) и POWER_OFF необходимо вначале исполнения программы сбросить регистр RTC_HISTORY. После этого события будут корректно регистрироваться в регистре History.

На рисунке 18.5 представлена временная диаграмма последовательности возникновения события просыпания, запуска LDO, старта программы и сброса регистра RTC_HISTORY. Временная диаграмма представлена для случая перехода в режим RUN из режима POWER_OFF. В качестве сигнала пробуждения «WAKE» на временной диаграмме может быть любое из разрешенных событий, представленных на структурной схеме контроллера пробуждения WAKE (рисунок 18.3).

Для корректной регистрации, событие должно продолжаться не менее 5мс.

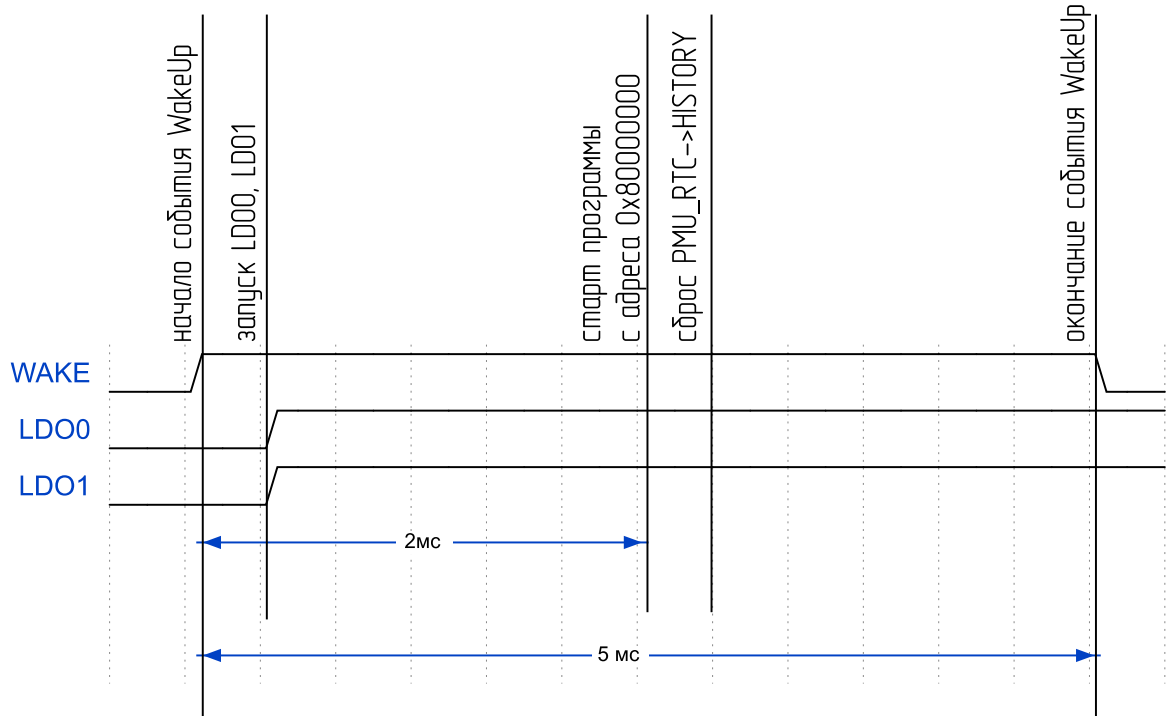


Рисунок 18.5 – Временная диаграмма последовательности возникновения события просыпания

18.3 Мониторинг вскрытий

Блок содержит два режима монитора:

- генерирование псевдослучайной последовательности (ПСП) на выходе AT_OUT с последующем сравнением с сигналом, полученным по входам AT_IN0, AT_IN1 и AT_IN2;
- мониторинг частоты, сравнивающий внешний осциллятор частотой 32768 Гц и внутренний RC генератор.

Мониторинг вскрытий входит в состав блока RTC и подключен к регулятору LDO_RTC, поэтому на выходе AT_OUT генерируется псевдослучайная последовательность амплитудой 1,4 В.

Мониторинг входов AT_IN0, AT_IN1, AT_IN2

Первая часть мониторинга вскрытий позволяет осуществлять наблюдение за целостностью внешних цепей, подключенных между выходом AT_OUT и входами AT_IN0, AT_IN1 и AT_IN2. При обнаружении разрыва устанавливается бит TIMEALARM регистра RTC_HISTORY и соответствующий бит TAMPER0EN - TAMPEREN2 регистра RTC_WAKECFG, позволяющие определить цепь, по которой обнаружен разрыв.

Событие считается действительным, если выполняются следующие условия:

- разрыв может быть обнаружен, если он происходит на входе, который разрешен соответствующим битом A2, A1, A0 регистра WAKECONFIG;
- разрыв каждой из цепей определяется сравнением значений на выходе ATOUT и соответствующим входом ATIN2, ATIN 1, ATIN 0.

На выход ATOUT выдается ПСП, определяемая регистром сдвига с линейной обратной связью (LFSR), синхронизируемым сигналом CLKDIV. Коэффициент деления для CLKDIV задаётся программно, что позволяет определить рабочую частоту сдвига с учетом паразитной емкости внешних соединений и потребляемой мощности. В таблице 18.1 представлены значения частоты CLKDIV в зависимости от значения битового поля CLKDIV регистра RTC_TRIM.

Таблица 18.1 – Значения делителя CLKDIV в зависимости от значения битового поля

Значение CLKDIV регистра	Частота CLKDIV, Гц	Значение CLKDIV регистра	Частота CLKDIV, Гц
0x00	32 768	0x0B	16
0x01	16 384	0x0C	8
0x02	8 192	0x0D	4
0x03	4 096	0x0E	2
0x04	2 048	0x0F	1
0x05	1 024	0x10	1/2
0x06	512	0x11	1/4
0x07	256	0x12	1/8
0x08	128	0x13	1/16
0x09	64	0x14	1/32
0x0A	32	0x15	1/64
Примечание – Значения битового поля 0x16 – 0x1F зарезервированы			

Сравнение частот внешнего осциллятора и внутреннего RC генератора

Вторая часть мониторинга вскрытий считает количество тактов внешнего осциллятора в течение 32 периодов внутреннего RC генератора (32768 Гц). При превышении количества посчитанных тактов внешнего осциллятора заданного порогового значения устанавливается флаг события AT2 битовом поле регистра RTC_CFG0. По умолчанию данный функционал отключен. Текущее значение разности количества тактов внешнего осциллятора отображено в битовом поле FREQDIFF регистра RTC_CFG0.

Предел отклонения частоты следования тактов задается битовым полем AT2 регистра RTC_CFG0, допустимые значения перечислены в таблице 18.2.

Мониторинг тактов внешнего осциллятора позволяет обнаружить отключение внешнего кварцевого резонатора. Каждый раз, при превышении разности частот внешнего осциллятора и RC генератора порогового значения вызывается событие пробуждения (ALARM).

Таблица 18.2 – Значения предела отклонения частоты следования тактов внешнего осциллятора

Значение AT2 регистра RTC_CFG0	Значение предела
0x00	не контролируется
0x01	до 4,7 %
0x02	до 7,8 %
0x03	до 10,9 %
0x04	до 14,1 %
0x05	до 17,2 %
0x06	до 20,3 %;
0x07	до 23,4 %

18.4 Калибровка сигнала с односекундным периодом

Амплитуда колебаний на выводе генератора контролируется схемой автоматической регулировки усиления (АРУ), которая управляет усилением генератора для поддержания постоянной амплитуды. Когда колебание обнаруживается схемой монитора, на внутреннем выводе сброса rstz устанавливается высокий уровень. Кварцевый генератор также имеет режим обхода (бит BYPASS регистра RTC_TRIM), получая тактовый сигнал напрямую с вывода XI_ RTC.

Калибровка представляет собой 11-битное слово, состоящее из знака и 10-разрядной подстройки (таблица 18.3).

Таблица 18.3 – Таблица калибровки односекундного периода

Значение поля TRIM1S регистра RTC_TRIM	Коррекция, ppm	Описание
011 1111 1111	512	Увеличение 1-секундного периода на 1024 такта частоты RTC каждые 64 секунды
011 1111 1110	511.5	Увеличение 1-секундного периода на 1023 такта частоты RTC каждые 64 секунды
000 0000 0001	1	Увеличение 1-секундного периода на 2 такта частоты RTC каждые 64 секунды
000 0000 0000	0.5	Увеличение 1-секундного периода на 1 такт частоты RTC каждые 64 секунды
111 1111 1111	0	Калибровка выключена (по умолчанию)
111 1111 1110	-0.5	Уменьшение 1-секундного периода на 1 такт частоты RTC каждые 64 секунды
100 0000 0001	-511	Уменьшение 1-секундного периода на 1022 такта частоты RTC каждые 64 секунды
100 0000 0000	-511.5	Уменьшение 1-секундного периода на 1023 такта частоты RTC каждые 64 секунды

18.5 Калибровка тактового сигнала RC-генератора

Поскольку RC-генераторы имеют низкую стабильность частоты генерируемых колебаний, в тактовом генераторе, входящем в состав блока RTC, предусмотрена возможность подстройки частоты сигнала, получаемого от RC-генератора. Подстройка частоты осуществляется с помощью записи поправочных значений в регистр RTC_TRIMRC.

Шаг подстройки меняется нелинейно. Зависимость изменения частоты от шага подстройки показана на рисунке 18.5.

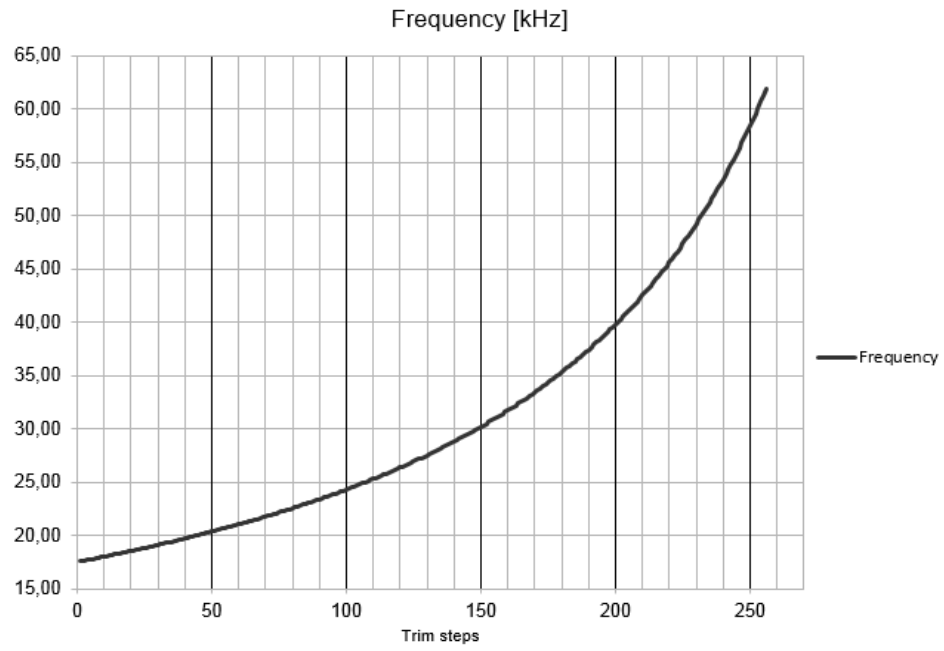


Рисунок 18.5 – График изменения частоты от шага подстройки

Из графика видно, что количество шагов для подстройки в сторону уменьшения частоты больше, а точность подстройки выше, чем в сторону увеличения частоты.

Примеры значений подстройки для записи в регистр RTC_TRIMRC указаны в таблице 18.4.

Таблица 18.4 – Таблица калибровки сигнала RC-генератора

Значение регистра RTC_TRIMRC	Коррекция, %	Описание
1000 0000	-45,51	Отрицательная коррекция частоты
1000 0001	-45,35	Отрицательная коррекция частоты
…		
1111 1111	-15,28	Отрицательная коррекция частоты
0000 0000	-14,91	Отрицательная коррекция частоты
0000 0001	-14,54	Отрицательная коррекция частоты
…		
0010 0001	-0,51	Отрицательная коррекция частоты
0010 0010	0	Значение по умолчанию, коррекция отсутствует
0010 0011	+0,52	Положительная коррекция частоты
…		
0111 1110	+90,20	Положительная коррекция частоты
0111 1111	+92,08	Положительная коррекция частоты

Для подстройки рекомендуется подход, когда для расчёта оптимального значения подстройки используется метод поиска с последовательным приближением.

18.6 Рекомендации по подключению и трассировке сигналов на печатной плате

Основные требования к подключению:

- минимизировать емкостные связи между сигналами RTC и другими сигналами;
- избегать параллельной трассировки сигналов RTC с другими высокоскоростными сигналами на печатной плате;
- поместить кварцевый резонатор как можно ближе к выводам микросхемы;
- использовать симметричную трассировку для сигналов XI_RTC и XO_RTC.

Даже если предполагается не использовать RTC, то питание и землю блока подключать необходимо. Вывод XO_RTC допускается никуда не подключать, а XI_RTC подтянуть к земле через резистор.

18.7 Генерация прерываний

В модуле предусмотрено несколько источников прерываний.

Сигналы запросов на прерывания:

- WKUP[0] - WKUP[2] – от событий с внешних выводов (WAKEUP0 - WAKEUP2);
- WKUP[3] – логическое ИЛИ событий периферийных блоков, входящих в состав домена батарейного питания (CLK1S, EXTRST, IWDT, CMP0, CMP1);
- WKUP[4] – событие изменения уровня напряжения VAON;
- WKUP[5] – событие V_BAT > VCC1(вывод №12);
- AT – событие нарушения целостности внешних цепей, подключенных между выходом AT_OUT и входами AT_IN0, AT_IN1 и AT_IN2;
- AT2 – событие превышения предела отклонения частоты следования тактов внешнего осциллятора и внутреннего RC генератора (LSI);
- ALARM – событие совпадения значений регистров RTC_TIME и RTC_ALARM.

19 Сторожевой таймер WDT

Сторожевой таймер позволяет сбросить систему в случае отказа программного обеспечения. Пользователь может включать или выключать сторожевой таймер по собственному усмотрению.

Сторожевой таймер представляет собой 32-битный обратный счетчик, который загружается значением из регистра LOAD. Счетчик уменьшается на единицу по каждому нарастающему фронту тактового сигнала WDTCLK.

По умолчанию сторожевой таймер не тактируется и находится в сбросе. Активировать таймер можно с помощью регистра WDTCFG блока RCU.

Сброс сторожевого таймера WDT осуществляется записью любого значения в регистр INTCLR.

Бит запрета записи (REGWRDIS) во все регистры сторожевого таймера, кроме регистра LOCK, необходим для предотвращения отключения сторожевого таймера сбойными программами. Чтобы узнать состояние бита блокировки необходимо прочитать бит REGWRDIS регистра LOCK. Для сброса бита следует записать в регистр LOCK значение 1ACCE551h. Для установки бита следует записать в регистр LOCK любое значение, кроме 1ACCE551h.

Включение счета сторожевого таймера и его прерывания осуществляются установкой бита INTEN в регистре CTRL. Когда счетчик таймера достигает нуля, устанавливается флаг WDTINT в регистре MIS, а в счетчик загружается значение из регистра LOAD.

Далее, если установлен бит RESEN, счетчик продолжает декрементироваться. Если на момент повторного достижения нуля флаг WDTINT установлен, производится сброс микроконтроллера.

Для предотвращения сброса микроконтроллера необходимо сбросить флаг WDTINT путем записи любого значения в регистр сброса INTCLR, что приводит к сбросу прерывания сторожевого таймера и загрузке счетчика значением из регистра LOAD.

Алгоритм работы сторожевого таймера показан на рисунке 19.1.

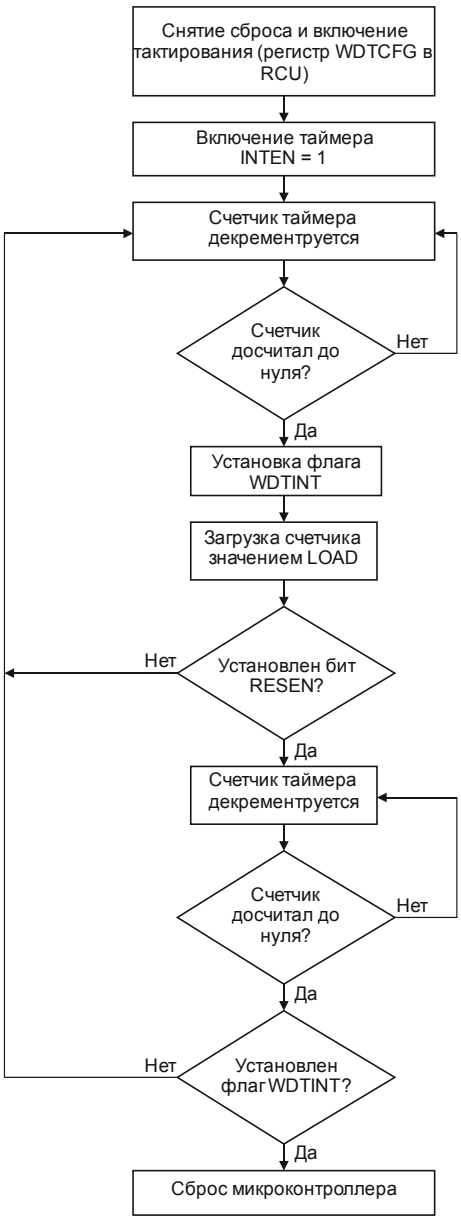


Рисунок 19.1 – Алгоритм работы сторожевого таймера

20 Независимый сторожевой таймер IWDT

Независимый сторожевой таймер позволяет сбросить систему в случае отказа программного обеспечения. Независимый сторожевой таймер входит в состав домена батарейного питания и продолжает функционирование даже при отключении основного питания.

Независимый сторожевой таймер представляет собой 32-битный обратный счетчик, который загружается значением из регистра LOAD. Счетчик уменьшается на единицу по каждому нарастающему фронту тактового сигнала IWDTCLK.

Независимый сторожевой таймер подключен к домену батарейного питания и может функционировать при отключении основного питания на выводах VCC1, VCC2.

Независимый сторожевой таймер может формировать сигнал пробуждения для вывода микроконтроллера из состояния сна.

По умолчанию сторожевой таймер не тактируется и находится в сбросе. Активировать таймер можно с помощью регистра IWDG_CFG блока PMURTC.

Разрешение на формирование сигнала пробуждения по сигналу прерывания от модуля IWDT осуществляется установкой бита IWDT регистра PMU_WK3EN блока PMURTC.

Бит BLK регистра CTRL отвечает за блокировку изменения источника тактирования IWDT и запрещения сброса IWDT при записи регистра PMU_RTC->IWDT_CFG.

Бит запрета записи (REGWRDIS) во все регистры сторожевого таймера, кроме регистра LOCK, необходим для предотвращения отключения сторожевого таймера сбойными программами. Чтобы узнать состояние бита блокировки необходимо прочитать бит REGWRDIS регистра LOCK. Для сброса бита следует записать в регистр LOCK значение 1ACCE551h. Для установки бита следует записать в регистр LOCK любое значение, кроме 1ACCE551h.

Включение счета сторожевого таймера и его прерывания осуществляются установкой бита INTEN в регистре CTRL. Когда счетчик таймера достигает нуля, устанавливается флаг WDTINT в регистре MIS, а в счетчик загружается значение из регистра LOAD.

Далее, если установлен бит RESEN, счетчик продолжает декрементироваться. Если на момент повторного достижения нуля флаг WDTINT установлен, производится сброс микроконтроллера.

Для предотвращения сброса микроконтроллера необходимо сбросить флаг WDTINT путем записи любого значения в регистр сброса INTCLR, что приводит к сбросу прерывания сторожевого таймера и загрузке счетчика значением из регистра LOAD.

Алгоритм работы сторожевого таймера показан на рисунке 20.1.

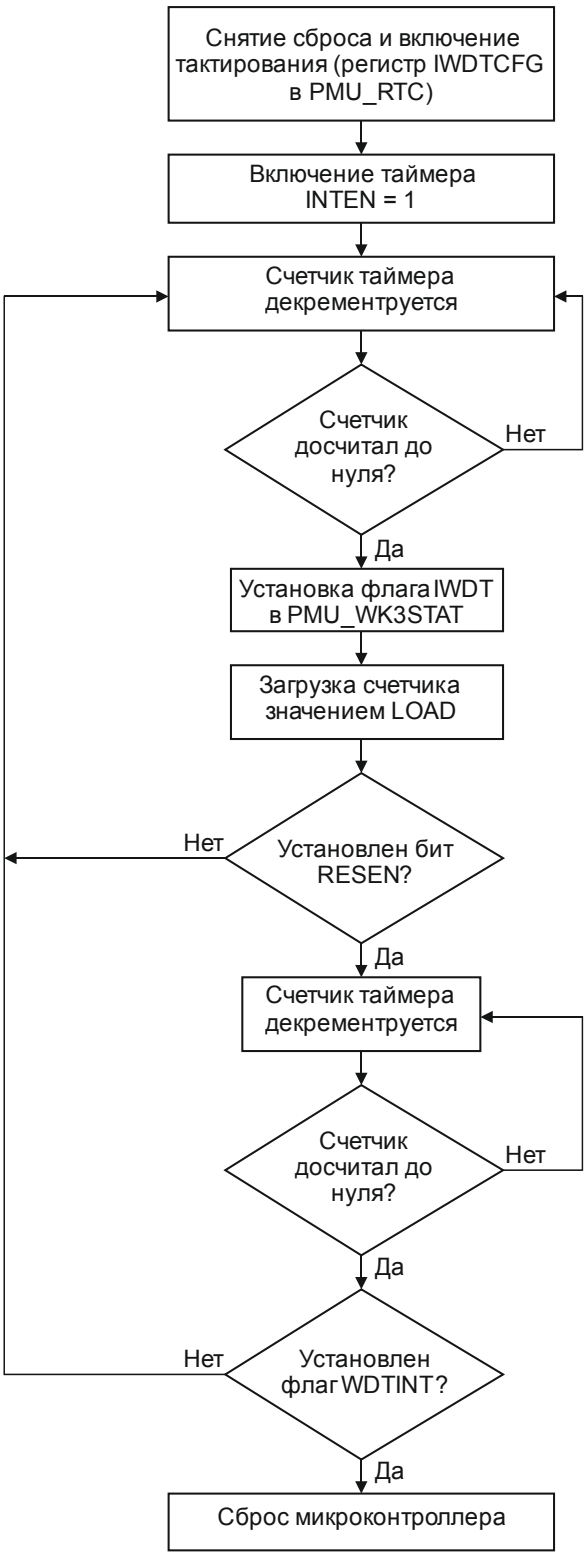


Рисунок 20.1 – Алгоритм работы независимого сторожевого таймера

21 Блок АЦП последовательного приближения

Блок быстродействующего АЦП включает АЦП последовательного приближения (архитектура SAR), схему управления, буферы результатов измерений и схему управления прерываниями. Структурная схема блока АЦП показана на рисунке 21.1.

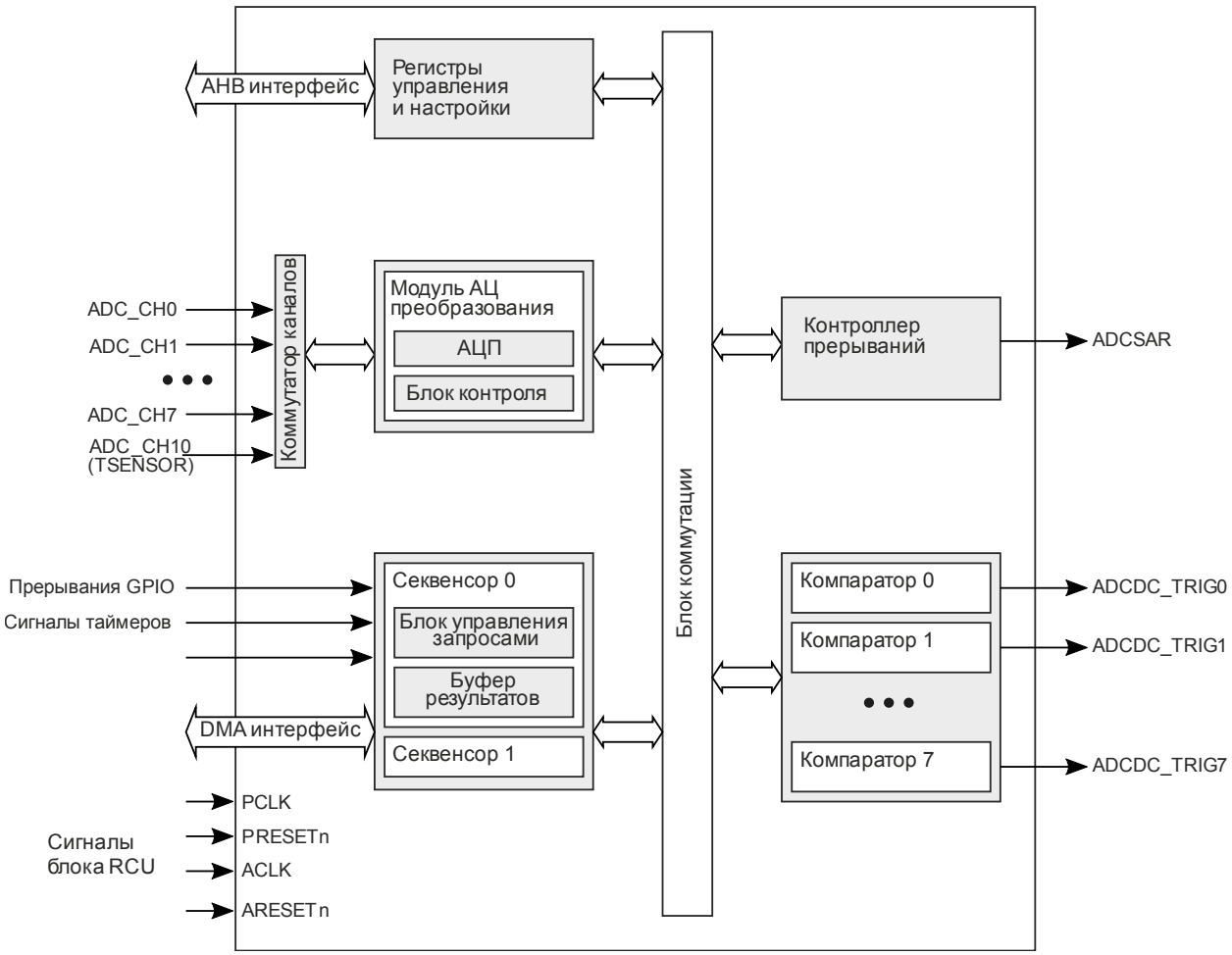


Рисунок 21.1 – Структурная схема блока АЦП

В блок АЦП входят:

- один 8-канальный (8 внешних каналов (ADC_CH0 - ADC_CH7) и канал № 10, подключенный к внутреннему датчику температуры) модуль АЦП с настраиваемой разрядностью 6, 8, 10, 12 бит и скоростью измерения по одному каналу в 6-битном режиме до 6 миллионов измерений в секунду;
- два секвенсора, каждый из которых позволяет независимо произвести запуск измерений по необходимым каналам АЦП и сгенерировать прерывание;
- 8 независимых цифровых компараторов, отслеживающих и сравнивающих измерения с пороговыми значениями для формирования прерываний и сигналов управления другими блоками микроконтроллера;
- два буфера результатов измерений (каждый организован по типу FIFO);
- блок управления прерываниями.

Блок АЦП суммарно имеет 8 внешних аналоговых входных каналов. Диапазон измеряемых напряжений ограничен опорным напряжением на выводе AREF и аналоговой землей GND2.

Аналоговый вход канала ADC_CH10 используется для подключения внутреннего датчика температуры и не имеет внешнего подключения.

Настройка тактирования и сброса блока АЦП и его модулей осуществляется посредством регистра ADCSARCFG блока управления тактовыми сигналами RCU. Вся внутренняя логика блока АЦП тактируется частотой ACLK, но запись/чтение контрольно-статусных регистров осуществляется на частоте PCLK.

Для правильной работы блока, необходимо обеспечить тактирование модулей АЦП частотой внутреннего сигнала ACLK от 140 кГц до 60 МГц, которую можно получить,

выбрав источник тактового сигнала полем CLKSEL, а также, при необходимости, включив и настроив делитель полями DIVEN и DIVN (поля регистра ADCSARCFG блока RCU). Частота ACLK не должна быть больше частоты тактового сигнала PCLK.

Время, на которое канал измерения подключается к внутренней зарядной емкости АЦП составляет период одного такта частоты ACLK. Предусмотрена возможность увеличения этого времени с помощью дополнительных тактов ожидания, которые задаются отдельно для каждого канала в массиве регистров ADCSAR->CHDELAY[]. В первую очередь это необходимо при подключении на вход канала АЦП высокоомной нагрузки.

21.1 Секвенсор

Секвенсор представляет собой управляющий блок, позволяющий разгрузить процессор от управления модулями АЦП. Секвенсор управляет запуском модулей АЦП, обработкой полученных результатов измерений и генерацией прерываний. В состав блока АЦП входят два секвенсора. Структурная схема секвенсора представлена на рисунке 21.2 (где секвенсор s от 0 до 1).

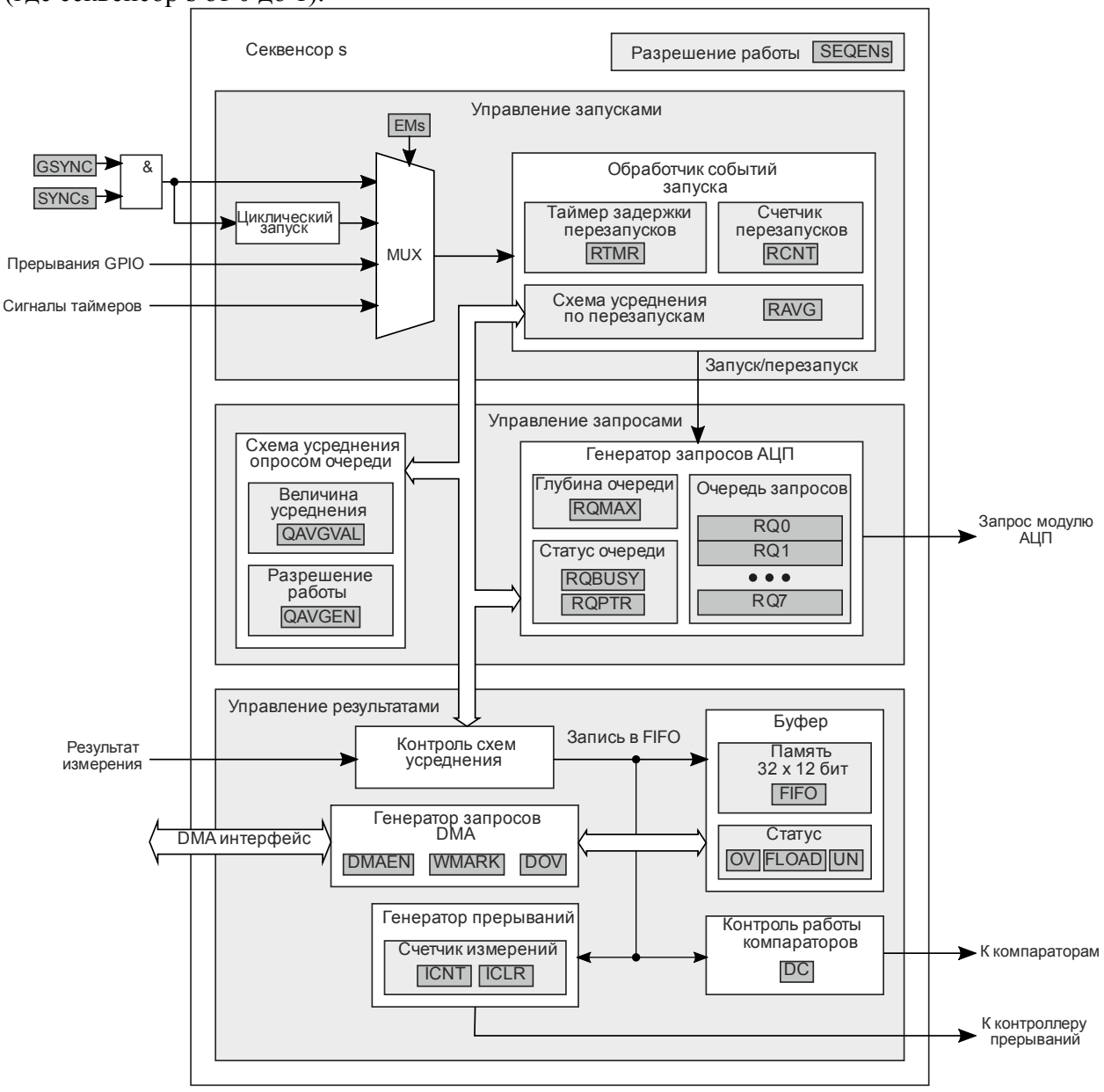


Рисунок 21.2 – Структурная схема секвенсора

Одиночные запуски по событиям

Разрешение работы секвенсора осуществляется установкой соответствующего бита

SEQENs (где s от 0 до 1) в регистре SEQEN.

Каждый секвенсор может совершать независимые однократные запуски по одному из событий, которое выбирается полем EMs регистра EMUX:

- установка бита GSYNC регистра SEQSYNC (запустятся только секвенсоры, для которых установлены биты SYNCs того же регистра);
- сигналы от таймеров TMR32, TMR0-TMR2;
- сигналы от GPIO.

Когда секвенсор s запускается по одному из сигналов событий, выставляется соответствующий флаг SEQBUSYs в регистре BSTAT. Также в это же время все настройки секвенсора сохраняются в теневых регистрах, и секвенсор начинает работу согласно полученным настройкам. Изменять настройки секвенсора во время его работы можно, но вступят в силу они лишь при следующем запуске. Флаг занятости держится установленным до тех пор, пока задача, инициированная событием, не будет полностью выполнена секвенсором (будет осуществлена запись последнего результата в FIFO).

События запуска не кэшируются, если секвенсор был занят выполнением текущей задачи (установлен SEQBUSYs), когда пришло очередное событие запуска, то оно будет проигнорировано. Необходимо учитывать это при настройке запуска по событиям от сторонних периферийных модулей так, чтобы время между возникновением событий было не меньше времени измерений. Диаграммы работы секвенсора при одиночных запусках по событиям показаны на рисунке 21.3.

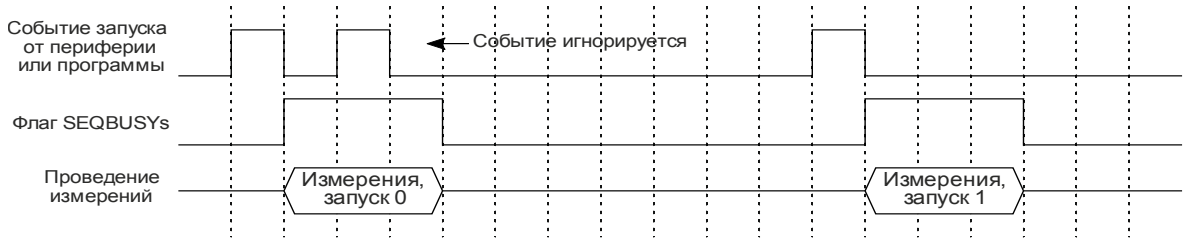


Рисунок 21.3 – Диаграммы работы секвенсора при одиночных запусках по событиям

Одиночные запуски по событиям с немедленными перезапусками

Здесь и далее, под перезапусками понимается внутренний механизм работы секвенсора, а под запусками – приход внешнего события, которое переводит секвенсор из ожидающего состояния в активное.

Секвенсор имеет возможность осуществлять как отложенные, так и немедленные автоматические перезапуски серий измерений, после прихода первого «инициирующего» события запуска от периферии. Перезапуск может выполняться до 255 раз (поле RCNT регистра SCCTL). Текущее состояние счетчика перезапусков можно узнать, прочитав поле RCNT регистра SCVAL. Диаграммы работы секвенсора при разрешенных, немедленных перезапусках показаны на рисунке 21.4.

Рисунок 21.4 – Диаграммы работы секвенсора при одиночных запусках по событиям с немедленными перезапусками при RCNT = 2 (регистр SCCTL)

Одиночные запуски по событиям с отложенными перезапусками

Отложенные перезапуски осуществляются спустя некоторое время от запуска,

задаваемое регистром SRTMR. Задержка задается в тактах ACLK и ведет счет независимо от текущего состояния секвенсора, поэтому, при её выборе необходимо учитывать, что текущие измерения должны завершиться до прихода сигнала перезапуска, иначе он будет пропущен.

В режиме одиночных запусков по событиям счетчик задержки перезапуска не будет считать, если поле RCNT регистра SCCTL равно нулю.

Как говорилось ранее, все настройки секвенсора сохраняются в теневых регистрах при его переходе в режим занятости и их изменение никак не повлияет на текущую работу. Однако существует возможность обновить задержку перезапуска еще во время работы секвенсора. Для этого надо записать новое значение задержки в регистр SRTMR с одновременно установленным последним битом NOWAIT. В этом случае, новая величина задержки вступит в силу после ближайшего события отложенного перезапуска.

Диаграммы работы секвенсора при активных отложенных перезапусках показаны на рисунке 21.5.

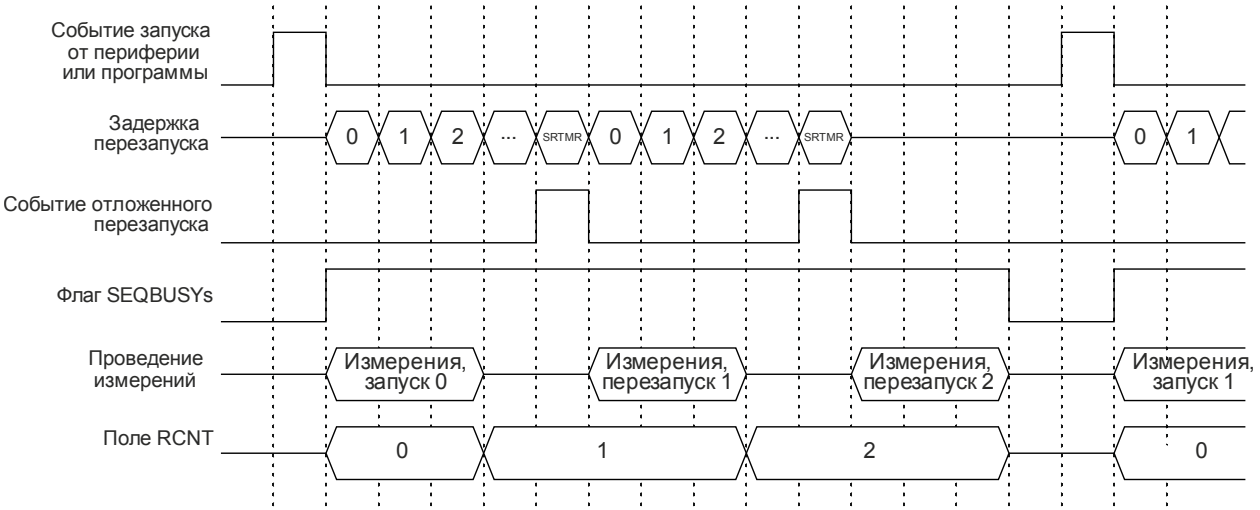


Рисунок 21.5 – Диаграммы работы секвенсора при одиночных запусках по событиям с отложенными перезапусками через определенное время (регистр SRTMR) при RCNT = 2 (регистр SCCTL)

Одиночные запуски с усреднением по перезапускам

Существует режим усреднения результатов по перезапускам, который включается установкой бита RAVGEN в регистре SCCTL. Главным условием работы этого режима является то, что поле RCNT регистра SCCTL должно содержать любое значение, соответствующее $2^p - 1$, где p равно от 1 до 8. Значение 2^p и является количеством серий измерений, которые будут усреднены (запуск и все перезапуски).

Работа этого режима заключается в том, что пока идут перезапуски, результаты попадут в буфер не сразу, а будут накапливаться во внутренних регистрах (каждому запросу на измерение соответствует такой регистр). Лишь во время последнего перезапуска, будут получены усреднённые значения по каждому из измерений, которые и будут помещаться в FIFO в порядке очереди.

Работа режима усреднения при немедленных перезапусках продемонстрирована на рисунке 21.6. При отложенных перезапусках данный вид усреднения работает аналогично, позволяя распределить равномерно измерения на длительном промежутке времени и получить среднее значение сигнала в конце него.

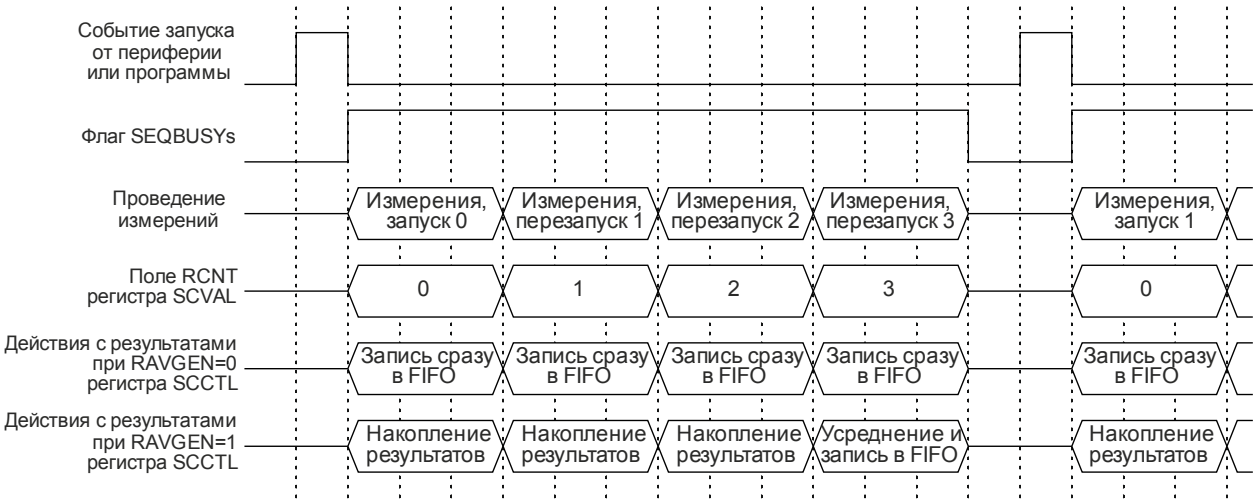


Рисунок 21.6 – Диаграммы работы секвенсора при одиночных запусках по событиям с усреднением по перезапускам при RCNT=3, RAVGEN=1 (регистр SCCTL)

Например, если измерения проводились по четырём каналам при RAVGEN = 0, то на момент снятия флага SEQBUSYs в FIFO было бы 16 результатов, но если усреднение по перезапускам было бы активно, то в FIFO находилось бы четыре усредненных результата по каждому из каналов.

Циклический запуск

Секвенсор может быть запрограммирован на циклический запуск – он будет запускаться снова каждый раз при завершении предыдущего запуска (имитация постоянно активного внешнего события). Чтобы начать работу в циклическом режиме, необходимо, после соответствующей конфигурации регистра EMUX, установить бит GSYNC регистра SEQSYNC. Чтобы завершить работу в циклическом режиме, необходимо выбрать в поле EMs регистра EMUX любое событие однократного запуска. Флаг занятости секвенсора SEQBUSYs в регистре BSTAT будет установлен сразу же по входу в циклический режим и будет сброшен только лишь по выходу из него. Диаграммы работы секвенсора при циклическом запуске показаны на рисунке 21.7.

Рисунок 21.7 – Диаграммы работы секвенсора при циклическом запуске

Циклический отложенный запуск

В отличие от режима одиночных запусков, циклический режим позволяет активировать счетчик задержки SRTMR, даже если поле RCNT регистра SCCTL равно нулю. В этом случае измерения будут запускаться не непрерывно, а с некоторой паузой (определяемой регистром SRTMR). Диаграммы работы секвенсора при циклическом запуске показаны на рисунке 21.8.

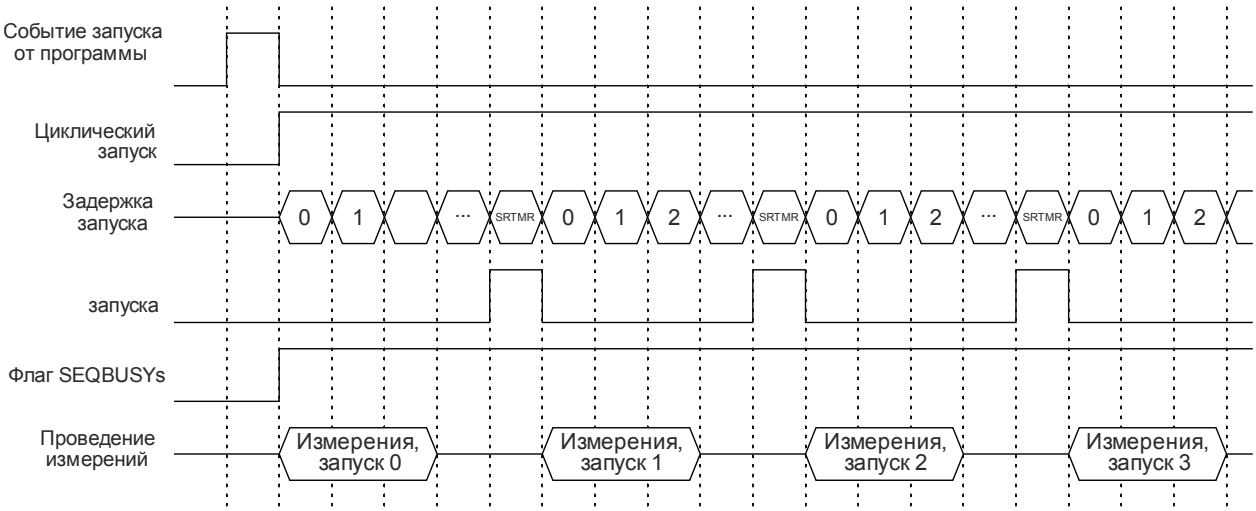


Рисунок 21.8 – Диаграммы работы секвенсора при циклическом отложенном запуске через время SRTMR

Циклический запуск с усреднением по перезапускам

Если в циклическом режиме установить поле RCNT регистра SCCTL значением, отличным от нуля, то секвенсор между запусками начнет делать нужное количество перезапусков. Но механика такого режима внешне никак не будет отличаться от обычной циклической работы, поэтому данный режим имеет смысл лишь в том случае, когда необходимо скомбинировать циклический режим с усреднением по перезапускам (как немедленным, так и отложенным).

Диаграммы работы секвенсора в циклическом режиме запуска с усреднением по отложенным перезапускам показаны на рисунке 21.9.

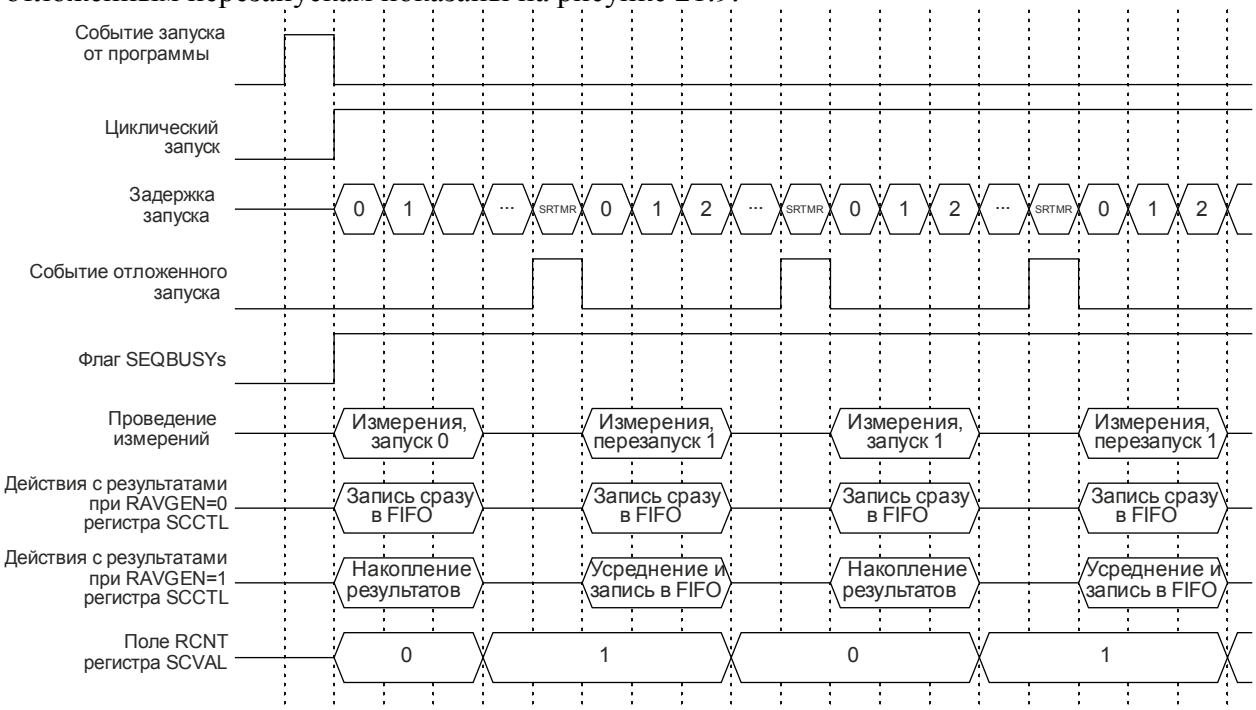


Рисунок 21.9 – Диаграммы работы секвенсора в циклическом запуске с отложенными перезапусками через определенное время (регистр SRTMR) при RCNT=1 (регистр SCCTL)

Генерация запросов на измерение

После запуска по событию или очередного перезапуска секвенсор начинает формировать запросы на измерения по каналам в порядке очереди, заданной регистром SRQSEL. Конец очереди или её «глубина» задается полем RQMAX регистра SRQCTL. Определить состояние очереди можно с помощью регистра SRQSTAT – в поле RQPTR

находится номер текущего запроса по порядку, а установленный флаг занятости RQBUSY говорит о том, что запрос выставлен и в состоянии обработки.

Иллюстрация к механизму генерации запросов на измерение представлена на рисунке 21.10.

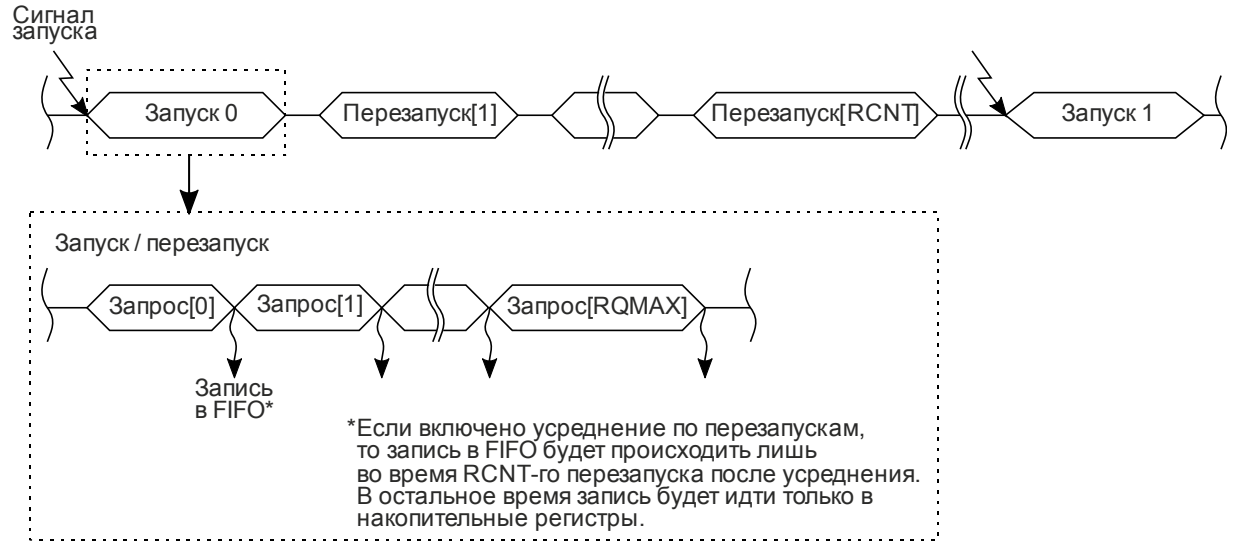


Рисунок 21.10 – Генерация секвенсором запросов на измерение

Пояснения к рисунку 21.10

1 Как только секвенсор получает сигнал запуска, он выставляет первый запрос из очереди и ожидает его принятия модулем АЦП.

2 Когда запрос будет принят, АЦП проведет измерение. По окончании измерения результат выполнения запроса будет передан секвенсору.

3 Секвенсор сохраняет результат в FIFO и продолжает выставлять запросы до окончания их очереди.

4 Если секвенсор настроен на проведение перезапусков (отложенных или немедленных), то по окончании очереди секвенсор перезапускается и снова выставляет первый в очереди запрос.

5 Лишь когда завершен последний перезапуск, то работа секвенсора, начатая по первичному событию запуска (пояснение 1), считается завершенной.

6 Секвенсор переходит в состояние ожидания следующего сигнала запуска.

Результат каждого запроса сохраняется в кольцевом буфере результатов секвенсора SFIFO. Буферы всех секвенсоров имеют емкость в 32 12-битных слова. Текущее количество слов в буфере можно узнать, прочитав регистр SFLOAD.

Контроль состояния буфера осуществляется посредством флагов регистра FSTAT. Установленный флаг OVs, указывает на то, что в FIFO не осталось свободных ячеек. Любая запись в буфер в таком случае будет игнорироваться до появления хотя бы одной свободной ячейки. Сброшенный флаг UNs свидетельствует о наличии в FIFO как минимум одного результата измерения. Соответственно, флаг UNs установится, когда FIFO будет полностью пуст, при этом результатом чтения пустого FIFO будут нули. Сброс флагов осуществляется путем записи в них единицы.

Усреднение сканированием очереди

Наряду с усреднением по перезапускам секвенсор имеет дополнительный механизм усреднения – усреднение сканированием (опросом) очереди измерений. Оба механизма могут работать как вместе, так и по отдельности.

Включается усреднение сканированием путем установки бита QAVGEN в регистре SRQCTL. Перед включением режима необходимо задать количество усредняемых опросов в поле QAVGVAL того же регистра – оно считается как 2^{QAVGVAL}.

Работа режима заключается в том, что очередь запросов выполняется не один раз (рисунок 21.10) с сохранением результатов в буфер после каждого запроса, а $2^{QAVGVAL}$ раз с сохранением результатов лишь на последней итерации сканирования после усреднения всех полученных измерений. Результаты измерений накапливаются и усредняются индивидуально для каждого запроса (аналогично усреднению по перезапускам).

Диаграммы работы усреднения сканированием показаны на рисунке 21.11.

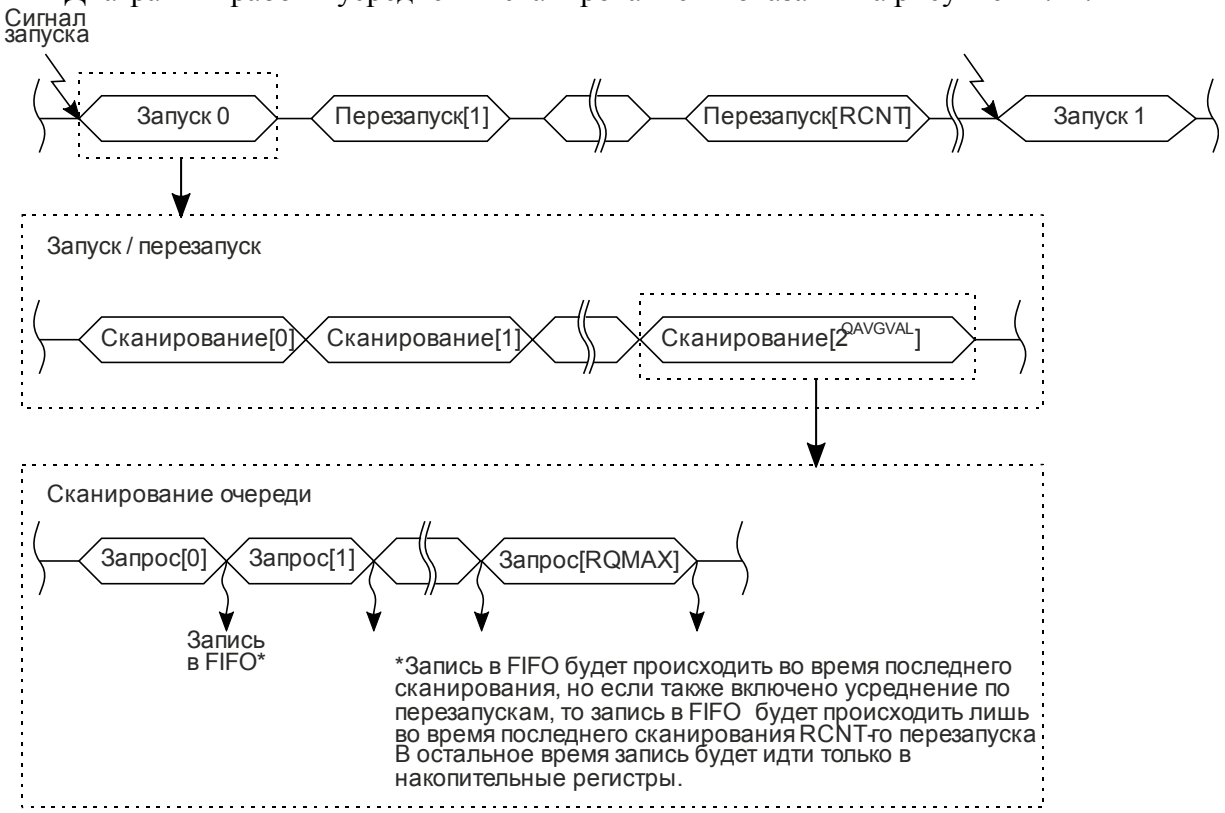


Рисунок 21.11 – Режим усреднения сканированием

Усреднение сканированием позволяет с относительно малой фазовой задержкой измерить уровень сигнала по нескольким каналам в пределах одной временной точки, а усреднение по перезапускам дает возможность усреднить значения, полученных таким образом точек, на достаточно большом интервале времени. В совокупности, оба механизма предлагают довольно гибкий инструментарий для автоматизации и усреднения измерений.

Генерация прерываний

Секвенсор может генерировать прерывания с заданной периодичностью. По завершении каждой записи в FIFO инкрементируется счетчик измерений. Как только было зафиксировано ICNT+1 записей (поле регистра SCCTL), генерируется прерывание. Стоит отметить, что даже если FIFO заполнено полностью (установлен OVs в регистре FSTAT), а измерения продолжают проводиться – счетчик измерений все равно будет считать последующие попытки записи секвенсора в FIFO (хотя они и будут игнорироваться самим FIFO). Текущее состояние счетчика можно узнать, прочитав поле ICNT регистра SCVAL. Сброс счетчика запросов происходит в следующих случаях:

- зафиксировано ICNT+1 записей;
- при запуске секвенсора по событию, если сброшен бит ICNTs в регистре CICNT;
- бит разрешения работы секвенсора ENs регистра SEQEN сброшен;
- программно – при каждой записи единицы в поле ICLR регистра SCVAL.

Для каждого секвенсора выделена линия прерываний – ADC_SEQs.

Использование прямого доступа к памяти

Для разрешения использования DMA секвенсором, необходимо установить бит DMAEN в регистре SDMACTL. Поле WMARK того же регистра задает уровень заполнения буфера секвенсора, по достижении которого будет запущен DMA. Перенос данных будет выполняться, пока не будет передано число результатов измерений, соответствующее состоянию поля WMARK.

Если очередной запрос на запуск DMA пришел раньше, чем закончился предыдущий цикл DMA от того же секвенсора, то будет выставлен флаг ошибки DOVs в регистре FSTAT.

21.2 Модуль АЦП

Структурная схема 8-канального модуля АЦП (каналы ADC_CH[0 – 7]) показана на рисунке 21.12.

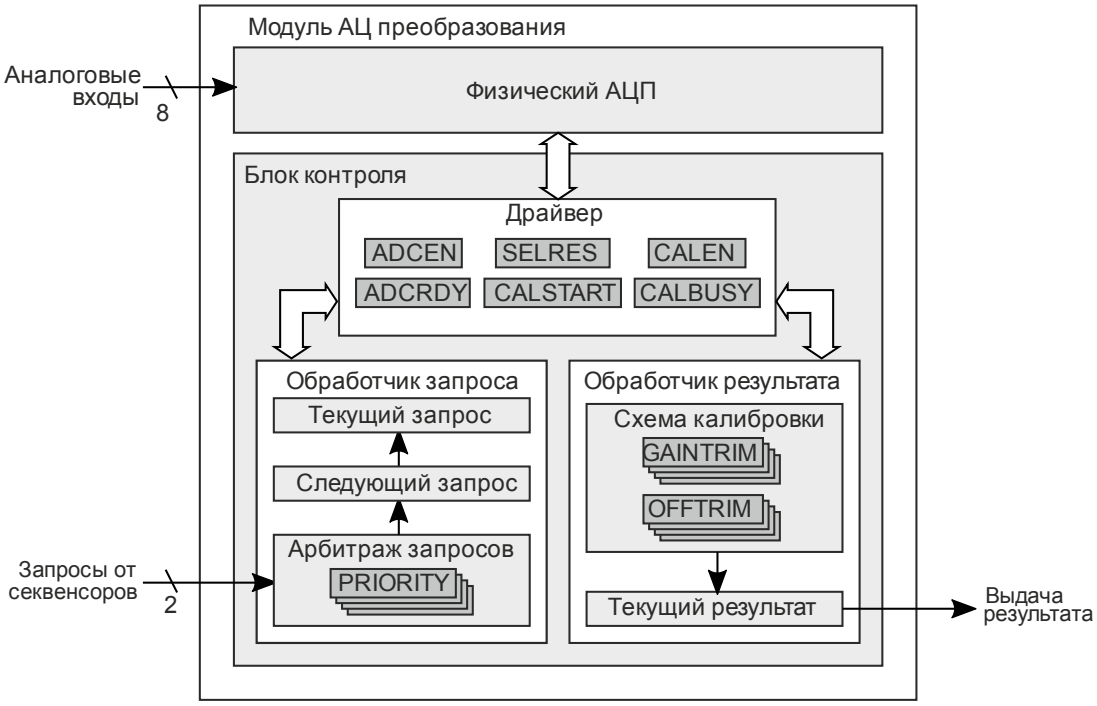


Рисунок 21.12 – Структурная схема 8-канального модуля АЦП

Поправочное значение можно загрузить вручную – величина записывается в поле CALIN регистра ACTL, затем устанавливается бит CALLOAD.

Процесс внутренней калибровки также может быть запущен автоматически в процессе инициализации АЦП, если на момент переключения ADCEN в единицу бит CALEN был установлен. Внутренняя калибровка происходит в течение 82 тактов сигнала ACLK.

Разрядность результата настраивается с помощью поля SELRES регистра ACTL – можно осуществить выбор между 6, 8, 10, 12 бит. Выравнивается результат по младшему биту. В таблице 21.1 показаны соответствия режима разрядности и длительности преобразования.

Таблица 21.1 – Зависимость скорости преобразования одного канала от разрядности данных

Поле SELRES регистра ACTL	Разрядность данных, бит	Количество тактов сигнала ACLK на преобразование	Максимальная скорость, выборок/с
00b	6	8	$6{,}25\times10^6$
01b	8	10	$5{,}00\times10^6$
10b	10	12	$4{,}1\times10^6$
11b	12	14	$3{,}50\times10^6$

Общая процедура включения модуля АЦП:

1 Включить LDO внутри АЦП, установив единицу в соответствующее поле LDOEN регистра PMUSYS – ADCSARPWRCFG.

2 Включить преобразователь уровня (Level-Shifter) сбросом бита LVLDIS в том же регистре.

3 Настроить и включить тактирование АЦП частотой ACLK от 140 кГц до 60 МГц, снять сброс (регистр RCU – ADCCFG).

4 Если необходимо, активировать автоматическую внутреннюю калибровку по включению – установить бит CALEN регистра ACTL.

5 Разрешить работу модуля АЦП с помощью установки бита ADCEN в регистре ACTL.

6 Если на момент разрешения работы был установлен CALEN, то запустится процедура калибровки. Если этот бит был сброшен, то выполнится одиночное пустое преобразование.

7 По окончании инициирующих процедур установится бит ADCRDY регистра ACTL. Модуль АЦП готов к работе.

Процедура деинициализации модуля АЦП для уменьшения потребления (например, при переходе в DEEPSLEEP):

1) Модуль АЦП должен находиться в состоянии ожидания – измерения не проводятся.

2) Сбрасывается бит ADCEN регистра ACTL.

3) Сбрасывается бит LDOEN регистра PMUSYS – ADCSARPWRCFG .

4) Устанавливаем бит LVLDIS регистра PMUSYS – ADCSARPWRCFG.

Правила арбитража

Когда АЦП начинает выполнять измерения по запросам, он устанавливает флаг ADCBUSY в регистре BSTAT. Флаг занятости будет сброшен лишь при полном отсутствии запросов, при последовательном выполнении запросов флаг не сбрасывается. Во время установки флага все настройки каналов АЦП сохраняются в теневых регистрах. Изменять их настройки во время работы можно, но вступят в силу они лишь при следующей установке флага ADCBUSY после сброса.

Секвенсоры могут выставлять запросы как одновременно, так и независимо друг от друга по разным событиям запуска. Для определения порядка выполнения запросов модулем АЦП реализована схема арбитража, со следующими правилами работы:

1) Если модуль АЦП был в режиме ожидания (включен и измерения не проводятся), то при поступлении запроса незамедлительно начинается его обслуживание.

2) Секвенсоры выставляют «ждущие» запросы – запрос будет активен до тех пор, пока модуль АЦП не начнет его обработку. Как только его обслуживание началось, запрос сбрасывается.

3) Как только текущий запрос секвенсора был сброшен, он сразу выставляет следующий запрос (если таковой имеется в наличии), чтобы успеть принять участие в

ближайшей процедуре арбитража, и чтобы АЦП не тратил такты на переход из активного режима в режим ожидания и обратно.

4) Секвенсор, во время работы, может не выставить следующий запрос после сброса текущего, но лишь в случае ожидания отложенного события перезапуска. Во всех остальных режимах запросы выставляются неразрывно друг за другом.

5) Если несколько секвенсоров выставили одинаковые запросы, то обслуживаться они будут параллельно. Результат запроса попадет в буферы всех соответствующих секвенсоров.

6) Арбитраж происходит перед началом обработки первого запроса (если АЦП был в ожидании) и в конце каждого обрабатываемого в текущий момент.

7) Если несколько секвенсоров одновременно выставят запросы, то запросы по каналам с меньшим номером имеют приоритет выше, чем каналы с большим.

8) Каналы с установленным битом PRIORITY в регистрах CHCTLn (где n – номер канала от 0 до 7) имеют более высокий приоритет, чем те, у которых бит сброшен.

9) Если одновременно будут выставлены запросы по каналам с установленным PRIORITY, то запросы по каналам с меньшим номером имеют приоритет выше, чем каналы с большим.

10) Необходимо с осторожностью производить частый опрос более высокоприоритетных каналов – это может значительно затруднить опрос остальных каналов.

Иллюстрация работы схемы арбитража запросов приведена на рисунке 21.13.

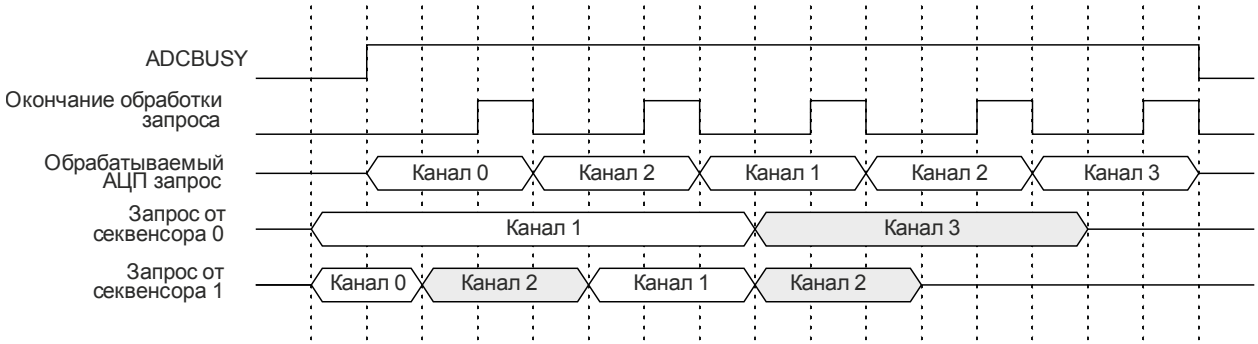


Рисунок 21.13 – Пример арбитража запросов, серым выделены запросы по каналам с установленным битом PRIORITY

Схема коррекции результатов измерений

Результат преобразования передается на схему коррекции, которая нивелирует ошибку усиления и смещения нуля и работа которой описывается формулой 21.1.

$$D_C = \frac{D_R \times (D_{MAX} + GAINTRIM)}{D_{MAX}} + OFFTRIM, \quad (21.1)$$

где D_R – данные, полученные непосредственно с АЦП;

D_C – скорректированные данные, выдаваемые секвенсору;

D_{MAX} – максимальная величина данных в зависимости от разрядности результата;

GAINTRIM – коэффициент корректировки усиления;

OFFTRIM – коэффициент корректировки смещения нуля.

Значения GAINTRIM и OFFTRIM заносятся в одноименные поля регистров CHCTLn (где n – номер канала от 0 до 7) в дополнительном коде. По умолчанию результат измерения проходит через схему коррекции, не изменяясь, т. к. коэффициенты равны нулю.

Реализована математика «насыщения» – когда значение OFFTRIM отрицательное и больше дроби, то результат будет равен нулю, если сумма OFFTRIM и дроби больше $D_{MAX} - 1$, то результат равен $D_{MAX} - 1$.

В зависимости от разрядности результата диапазоны коэффициентов коррекции

отличаются. Все значения вносятся в дополнительном коде и выравниваются по младшему биту.

Таблица 21.2 – Зависимость диапазонов коэффициентов коррекции от разрядности данных

Поле SELRES регистра ACTL	Разрядность данных, бит	Диапазон значений коэффициентов коррекции OFFTRIM, GAINTRIM
00b	6	от –4 до 3
01b	8	от –16 до 15
10b	10	от –64 до 63
11b	12	от –256 до 255

21.3 Цифровой компаратор

В состав блока АЦП входят восемь компараторов. Структурная схема компаратора показана на рисунке 21.14 (где d от 0 до 7).

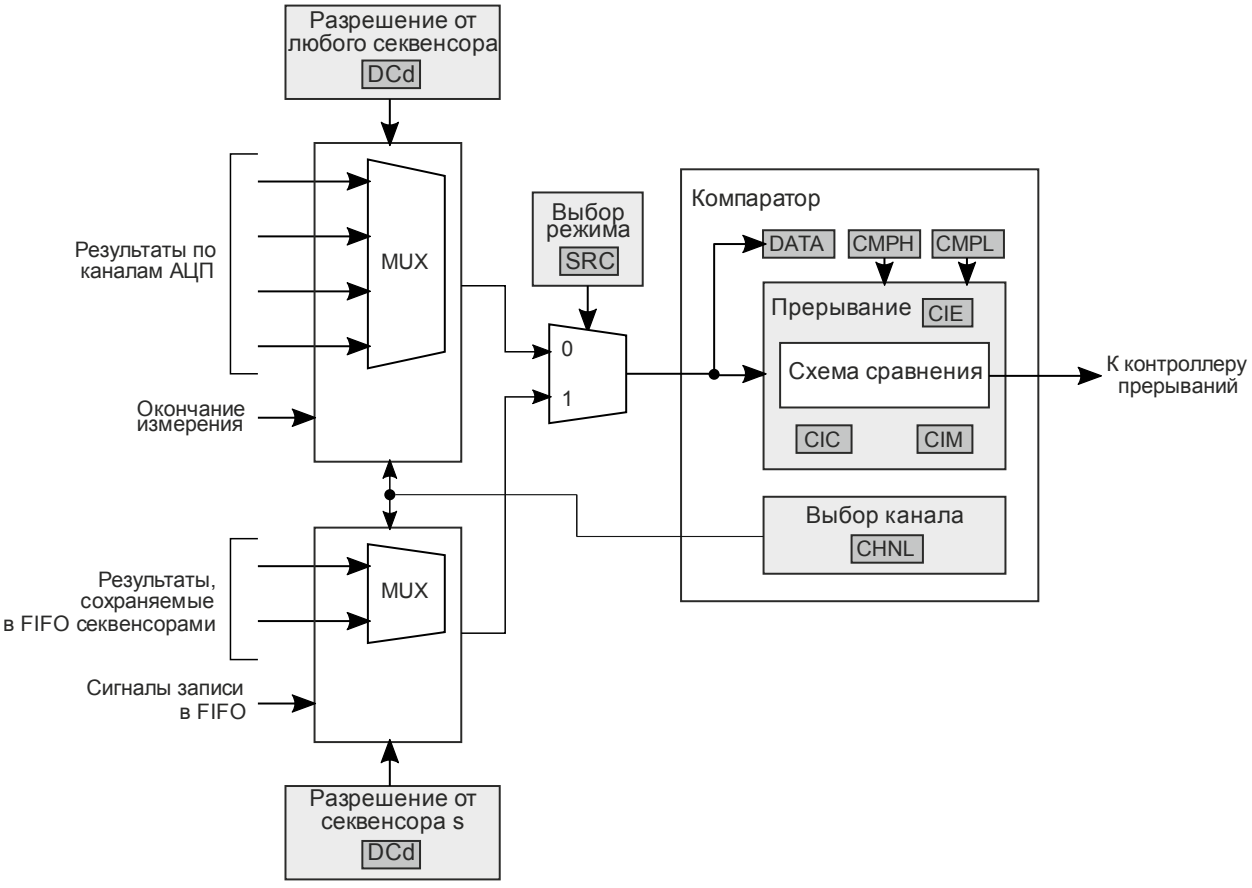


Рисунок 21.14 – Структурная схема компаратора

Все компараторы блока АЦП независимы. Каждый компаратор может обрабатывать результат измерения любого канала.

По умолчанию, когда модуль АЦП завершает обработку запроса по каналу, он выставляет результат, который захватывает как секвенсор, так и разрешенный (любым из секвенсоров) и настроенный на этот канал компаратор.

Возможен и другой режим работы, когда на компаратор будет подаваться результат, который секвенсор записывает в FIFO, но только в том случае, если канал, соответствующий сохраняемому результату, также совпадает с каналом, на который

настроен компаратор. Включение этого режима осуществляется установкой бита SRC в регистре DCTL.

Правила настройки цифровых компараторов:

1) Посредством регистра SRQSEL секвенсора s выбираются каналы для измерений.

2) Посредством регистра SDC выбираются (разрешаются) компараторы для обработки полученных результатов запросов (установкой битов DCd). Запрещенные компараторы не обрабатывают полученные результаты.

3) Для каждого компаратора d в его регистре DCTL в поле CHNL указывается номер канала, результат измерения которого будет передан на него. По умолчанию значение CHNL = 0h, т. е. все компараторы настроены на работу с нулевым каналом.

4) Битом SRC в регистре DCTL выбирается источник данных для компаратора – результаты непосредственно с АЦП или результаты, записываемые секвенсором в FIFO (которые могут быть уже усреднены).

Результат измерения, полученный компаратором d, передается во внутреннюю схему сравнения и одновременно с этим сохраняется в регистре DDATA. Схема сравнения выполняет проверку соответствия результата измерения заданному условию (поля CTC, CTM регистра DCTL и поля CMPL, CMPH регистра DCMP) и в зависимости от результата проверки переключает выходной триггер и устанавливает флаг события сравнения DCEVd в регистре DCTRIG. Работа триггера разрешается установкой бита CTE регистра DCTL.

Сравнение по условию «Измерение ≤ CMPL» (CTC = 00b):

- в однократном режиме (CTM = 01b) выходной триггер переключится в единицу только в случае, если результат сравнения окажется положительным при том, что результат предыдущего сравнения был отрицательным. В остальных случаях состояние триггера – ноль;

- в многократном режиме (CTM = 00b) выходной триггер будет переключаться в единицу каждый раз, когда результат сравнения будет положительным;

- в однократном режиме с гистерезисом (CTM = 11b) выходной триггер переключится в единицу только в случае, если после прекращения выполнения условия «CMPH ≤ Измерение», результат сравнения окажется положительным при том, что результат предыдущего сравнения был отрицательным. В остальных случаях состояние триггера – ноль;

- в многократном режиме с гистерезисом (CTM = 10b) выходной триггер переключится в единицу в случае, если после прекращения выполнения условия «CMPH ≤ Измерение», результат сравнения окажется положительным при том, что результат предыдущего сравнения был отрицательным, и далее триггер будет оставаться в состоянии единицы до тех пор, пока снова не выполнится условие «CMPH ≤ Измерение».

Пример функционирования триггера показан на рисунке 21.15.

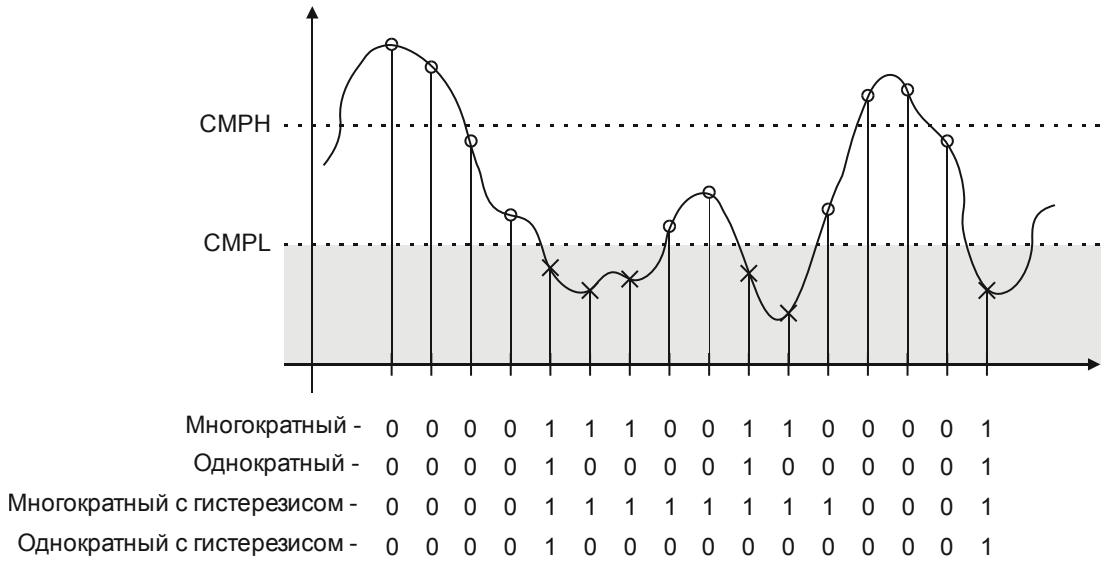


Рисунок 21.15 – Функционирование триггера при СТС = 00b

Сравнение по условию «CMPL ≤ Измерение ≤ CMPH» (СТС = 01b):

- в однократном режиме выходной триггер переключится в единицу только в случае, если результат сравнения окажется положительным при том, что результат предыдущего сравнения был отрицательным. В остальных случаях состояние триггера – ноль;

- в многократном режиме выходной триггер будет переключаться в единицу каждый раз, когда результат сравнения будет положительным;

- однократный и многократный режимы с гистерезисом не поддерживаются.

Пример функционирования триггера показан на рисунке 21.16.

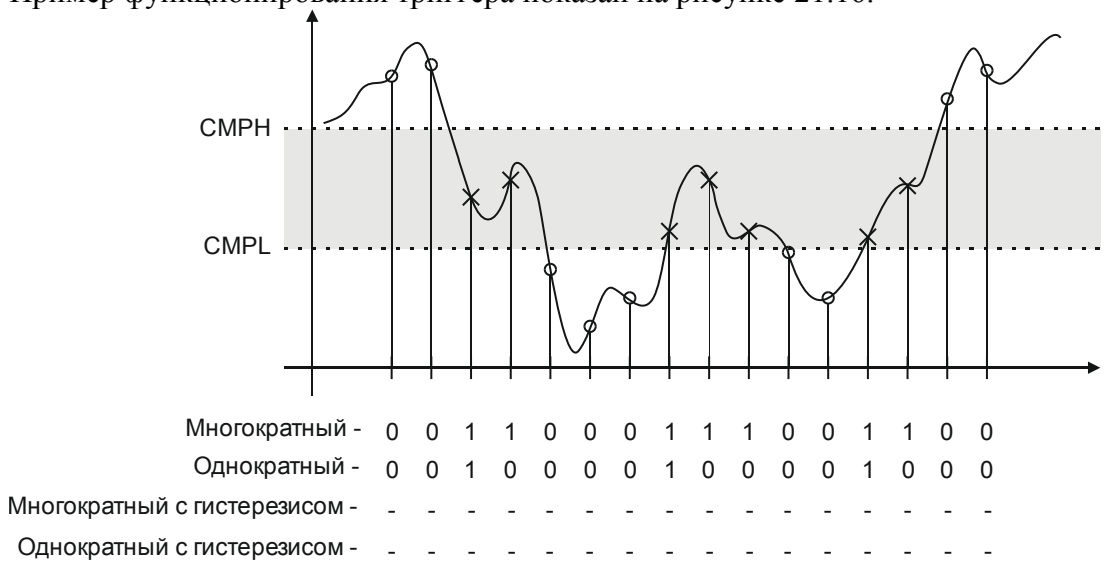


Рисунок 21.16 – Функционирование триггера при СТС = 01b

Сравнение по условию «CMPH ≤ Измерение » (СТС = 10b):

- в однократном режиме выходной триггер переключится в единицу только в случае, если результат сравнения окажется положительным при том, что результат предыдущего сравнения был отрицательным. В остальных случаях состояние триггера – ноль;

- в многократном режиме выходной триггер будет переключаться в единицу каждый раз, когда результат сравнения будет положительным;

- в однократном режиме с гистерезисом выходной триггер переключится в единицу только в случае, если после прекращения выполнения условия «Измерение ≤ CMPL», результат сравнения окажется положительным при том, что результат предыдущего

сравнения был отрицательным. В остальных случаях состояние триггера – ноль;

- в многократном режиме с гистерезисом (CTM = 10b) выходной триггер переключится в единицу в случае, если после прекращения выполнения условия «Измерение ≤ CMPL», результат сравнения окажется положительным при том, что результат предыдущего сравнения был отрицательным, и далее триггер будет оставаться в состоянии единицы до тех пор, пока снова не выполнится условие «Измерение ≤ CMPL».

Пример функционирования триггера показан на рисунке 21.17.

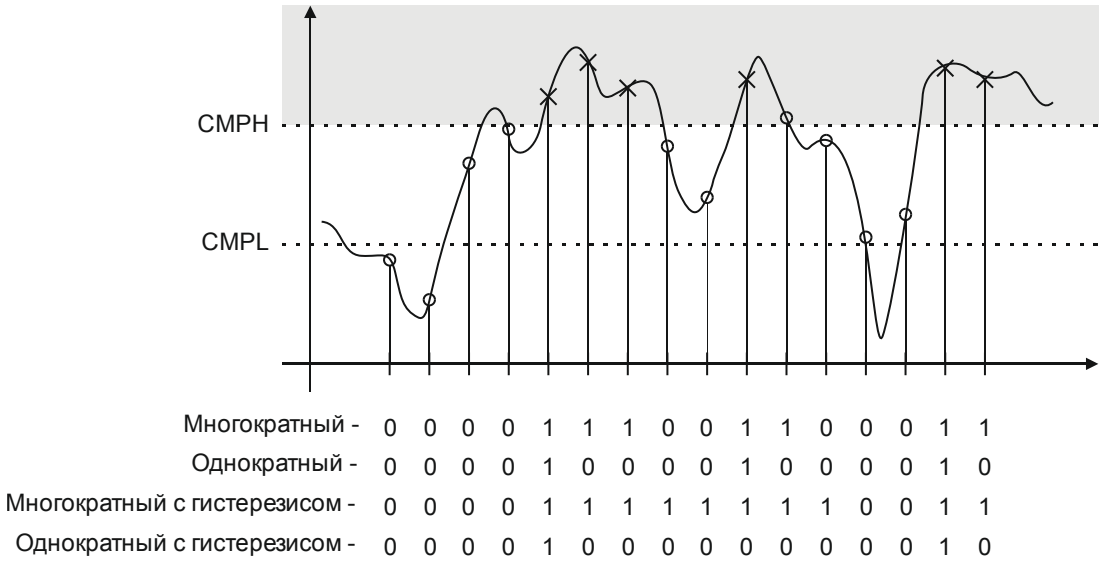


Рисунок 21.17 – Функционирование триггера при CTC = 10b

Переключение выходного триггера в единицу устанавливает соответствующий флаг TOSd в регистре DCTRIG и генерирует управляющий сигнал для пороговых выключателей блоков ШИМ. Вне зависимости от состояния флагов TOSd по каждому событию сравнения устанавливается соответствующий флаг DCEVd того же регистра. Флаги события сравнения сбрасываются записью единицы. Сброс самого триггера и его статусного флага выполняется также записью единицы в соответствующий бит TOSd регистра DCTRIG.

Независимо от состояния триггера (разрешен или запрещен) в случае положительного результата сравнения компаратор может генерировать прерывание. Для этого следует установить бит CIE регистра DCTL и задать условия CIC и CIM (аналогичны по функционалу CTC и CTM).

Примечание – Условия срабатывания выходного триггера компаратора и условия генерирования прерываний могут не совпадать.

21.4 Датчик температуры

В состав микроконтроллера входит датчик температуры, который подключен к каналу 10 АЦП. В связи с тем, что датчик температуры имеет высокое выходное сопротивление, для получения достоверных значений напряжения необходимо увеличить время подключения к цепи зарядной емкости АЦП путем увеличения количества дополнительных тактов в регистре CHDELAY[10].

21.5 Минимальное время измерения

Для получения корректных значений измерений необходимо выдерживать минимальное время измерения, которое, в свою очередь зависит от скорости канала.

В блоке АЦП последовательного приближения к быстрым относятся каналы с номерами от 0 до 5. Время измерения для быстрых каналов при использовании различной разрядности указано в таблице 21.2.

К медленным относятся каналы с номерами 6 и 7, а также канал 10, подключенный к датчику измерения температуры. Время измерения для медленных каналов при использовании различной разрядности указано в таблице 21.3.

Значения количества дополнительных циклов тактовой частоты в таблицах 21.2 и 21.3 указаны при работе АЦП на максимальной частоте 50 МГц.

Количество дополнительных циклов тактовой частоты АЦП задается для каждого канала индивидуально с помощью регистров CHDELAYn (где n – номер канала от 0 до 7).

Таблица 21.2 – Минимальное время измерения для быстрых каналов

Разрядность	Входное сопротивление, кОм	Минимальное время измерения, нс	Количество дополнительных циклов тактовой частоты АЦП
1	2	3	4
12 бит	0	5	0
	0,05	8	0
	0,1	11	0
	0,2	17	0
	0,5	33	1
	1	60	3
	5	271	13
	10	531	26
	20	1049	52
	50	2604	130
	100	5193	259
10 бит	0	4	0
	0,05	7	0
	0,1	9	0
	0,2	13	0
	0,5	27	1
	1	50	2
	5	225	11
	10	441	22
	20	873	43
	50	2166	108
	100	4321	216

Продолжение таблицы 21.2

1	2	3	4
8 бит	0	3	0
	0,05	5	0
	0,1	7	0
	0,2	10	0
	0,5	21	1
	1	39	1
	5	179	8
	10	352	17
	20	697	34
	50	1729	86
	100	3449	172
6 бит	0	3	0
	0,05	4	0
	0,1	5	0
	0,2	8	0
	0,5	16	0
	1	29	1
	5	134	6
	10	263	13
	20	521	26
	50	1292	64
	100	2578	128

Таблица 21.3 – Минимальное время измерения для медленных каналов

Разрядность	Входное сопротивление, кОм	Минимальное время измерения, нс	Количество дополнительных циклов тактовой частоты АЦП
1	2	3	4
12 бит	0	12	0
	0,05	15	0
	0,1	18	0
	0,2	24	1
	0,5	40	1
	1	67	3
	5	277	13
	10	537	26
	20	1055	52
	50	2608	130
	100	5195	259

Продолжение таблицы 21.3

1	2	3	4
10 бит	0	10	0
	0,05	12	0
	0,1	15	0
	0,2	19	0
	0,5	33	1
	1	55	2
	5	230	11
	10	446	22
	20	878	43
	50	2169	108
	100	4322	216
8 бит	0	8	0
	0,05	10	0
	0,1	12	0
	0,2	15	0
	0,5	26	1
	1	43	2
	5	183	9
	10	356	17
	20	700	34
	50	1731	86
	100	3449	172
6 бит	0	6	0
	0,05	7	0
	0,1	9	0
	0,2	11	0
	0,5	19	0
	1	32	1
	5	137	6
	10	266	13
	20	524	26
	50	1294	64
	100	2578	128

21.6 Прерывания

При генерировании прерываний устанавливаются флаги SEQRISs и DCRISd в регистре RIS. Если были установлены маски прерываний в регистре IM (поля SEQIMs и DCIMd), то также устанавливаются соответствующие маскированные флаги прерываний SEQMISs и DCMISd. Сброс флагов (маскированных и немаскированных) осуществляется записью единицы в соответствующие поля регистра IC (поля SEQICs и DCICd).

Установка маскированных флагов SEQMISs вызывает формирование соответствующих прерываний ADC_SEQs блока АЦП.

Флаги DCMISd компараторов объединены по ИЛИ, и установка любого из них вызывает формирование прерывания ADC_DC блока АЦП.

Структурная схема контроллера прерываний показана на рисунке 21.18.

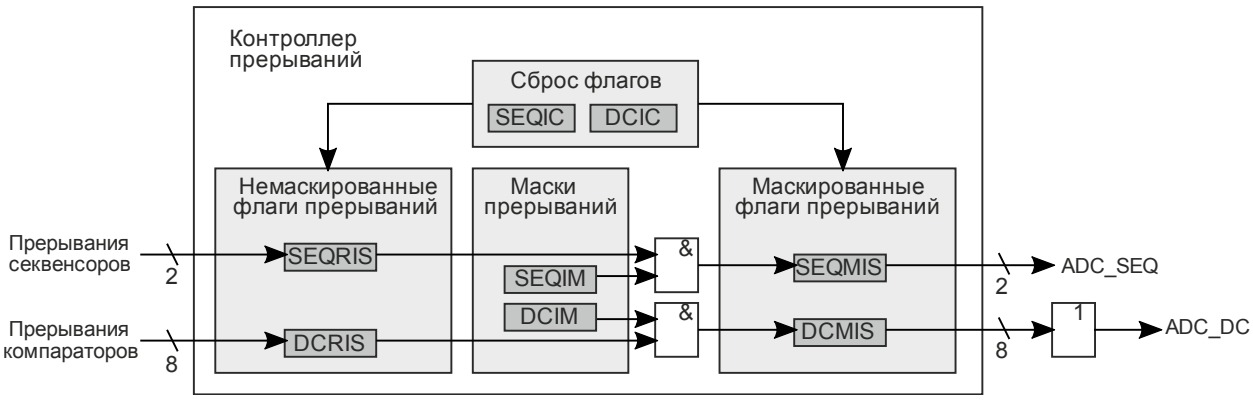


Рисунок 21.18 – Контроллер прерываний

21.7 Примеры работы блока АЦП

В этом подразделе представлены разнообразные примеры настройки блока АЦП для осуществления измерений в различных режимах. Для всех примеров подразумевается, что АЦП тактируется частотой в 33,3 МГц.

Включение LDO

Модуль АЦП имеет внутренний LDO, который должен быть включен в первую очередь.

1 Для включения LDO внутри АЦП необходимо установить единицу в соответствующее поле LDOEN регистра PMUSYS - ADCPWRCFG .

```
PMUSYS->ADCPWRCFG_bit.LDOEN0 = 1;
```

Настройка тактирования

Один из примеров настройки тактирования АЦП совместно с настройкой системного тактового сигнала представлен ниже.

1) С помощью регистра PLLCFG блока RCU настраиваем выходную частоту PLL 50 МГц. Осуществляем процедуру перевода системной частоты на источник PLL. Таким образом, частота системного тактового сигнала SYSCLK будет равна 50 МГц.

2) Настройка рабочего тактового сигнала АЦП ACLK производится с помощью регистра ADCCFG блока RCU. Выбираем в качестве источника выходную частоту PLL (поле CLKSEL = 1), включаем делитель на 6 (поле DIVN=2, бит DIVEN=1). Таким образом, f_{ACLK} = 33,3 МГц.

```
RCU->ADCSARCLKCFG_bit.CLKSEL = 1;
RCU->ADCSARCLKCFG _bit.DIVN = 2;
RCU->ADCSARCLKCFG _bit.DIVEN = 1;
```

3) Включаем тактирование блока (бит CLKEN=1) и снимаем сброс (бит RSTDIS=1). Блок АЦП готов к дальнейшим конфигурациям.

```
RCU-> ADCSARCLKCFG _bit.CLKEN = 1;
RCU-> ADCSARCLKCFG _bit.RSTDIS = 1;
```

Пример 1 – Программный запуск одного секвенсора
Требуемый режим работы:
- программный запуск;
- секвенсор 0;
- однократное измерение каналов 0-3;
- без прерываний (опрос флагов).

Код, соответствующий настройке и запуску необходимого режима, представлен ниже.

```
// Настройка модуля АЦП – 12 бит и калибровка при включении
ADCSAR->ADCCTL_bit.SELRES = 3;
ADCSAR->ADCCTL_bit.CALEN = 1;
ADCSAR->ADCCTL_bit.ADCEN = 1;
//Настройка секвенсора 0
ADCSAR->EMUX_bit.EM0 = 0;
ADCSAR->SEQ[0].RQCTL_bit.RQMAX = 0x3;
ADCSAR->SEQ[0].RQSEL0_bit.RQ0 = 0x0;
ADCSAR->SEQ[0].RQSEL0_bit.RQ1 = 0x1;
ADCSAR->SEQ[0].RQSEL0_bit.RQ2 = 0x2;
ADCSAR->SEQ[0].RQSEL0_bit.RQ3 = 0x3;
ADCSAR->SEQEN_bit.SEQEN0 = 1;
// Запуск
while(!ADCSAR->ADCCTL_bit.ADCRDY);
ADCSAR->SEQSYNC_bit.SYNC0 = 1;
ADCSAR->SEQSYNC_bit.GSYNC = 1;
```

1 Перед разрешением работы АЦП устанавливаем 12-битный режим измерений (SELRES=3) и включаем механизм калибровки (CALEN). Затем разрешаем работу модуля АЦП – необходимо установить бит ADCEN в регистре ADCCTL.

2 Для работы выбран секвенсор 0. Настроим его источник запуска – поле EM0 регистра EMUX должно быть равно 0, т. к. запуск программный.

3 В поле RQMAX регистра SRQCTL необходимо внести значение 3h, т. к. измерения будут проводиться по четырем каналам.

4 Настраиваем каналы для запросов. Допустим, необходимо опросить каналы последовательно от нулевого к третьему, значит, в регистр SRQSEL0 необходимо внести значения в поля: RQ0=0h, RQ1=1h, RQ2=2h, RQ3=3h.

5 Разрешаем работу секвенсора. Для этого необходимо установит бит SEQEN0 в регистре SEQEN.

6 Запускаем измерения. Перед запуском проверяем флаг ADCRDY, чтобы быть уверенными в том, что модуль АЦП провел необходимые инициализации. Для того чтобы начать измерения, необходимо установить бит SYNC0 в регистре SEQSYNC и записать в бит GSYNC единицу.

7 Проводим опрос флагов, чтобы установить окончание измерений. Например, можно опрашивать регистр SFLOAD, ожидая, пока он не станет равен 4h.

8 Считываем четыре результата измерения из буфера SFIFO.

Работа режима проиллюстрирована на рисунке 21.19.

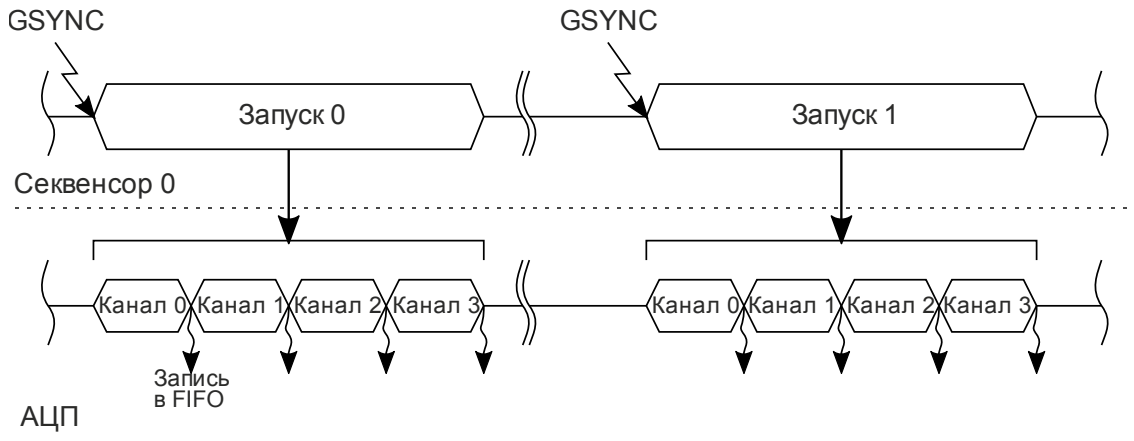


Рисунок 21.19 – Диаграммы программного запуска одного секвенсора

Пример 2 – Циклический опрос канала с задержками
Требуемый режим работы:
- циклическая работа;
- секвенсор 0;
- опрос канала номер 2 каждую 1мс;
- коррекция канала 2;
- каждый опрос из 16 последовательных измерений и усреднения;
- прерывание по каждой записи в FIFO.

Код, соответствующий настройке и запуску необходимого режима, представлен ниже.

```
//Настройка модуля АЦП – 12 бит и калибровка при включении
ADCSAR->ADCCTL_bit.SELRES = 3;
ADCSAR->ADCCTL_bit.CALEN = 1;
ADCSAR->ADCCTL_bit.ADCEN = 1;
//Коррекция канала 2
ADCSAR->CHCTL[2].CHCTL_bit.GAINTRIM = 5; // значения для примера
ADCSAR->CHCTL[2].CHCTL_bit.OFFTRIM = (uint32_t)(-5);
//Настройка секвенсора 0
ADCSAR->EMUX_bit.EM0 = 0xF;
ADCSAR->SEQ[0].CCTL_bit.ICNT = 0;
ADCSAR->SEQ[0].RTMR = 33332;
ADCSAR->SEQ[0].RQCTL_bit.RQMAX = 0;
ADCSAR->SEQ[0].RQCTL_bit.QAVGVAL = 4;
ADCSAR->SEQ[0].RQCTL_bit.QAVGEN = 1;
ADCSAR->SEQ[0].RQSEL0_bit.RQ0 = 2;
ADCSAR->SEQEN_bit.SEQEN0 = 1;
// PLIC прерывание
ADCSAR->IM_bit.SEQIM0 = 1;
PLIC_IntEnable(ADC_SEQ0_IRQn);
// Запуск
while(!ADCSAR ->ADCCTL_bit.ADCRDY);
ADCSAR->SEQSYNC_bit.SYNC0 = 1;
ADCSAR->SEQSYNC_bit.GSYNC = 1;
```

1 Перед разрешением работы АЦП устанавливаем 12-битный режим измерений (SELRES=3) и включаем механизм калибровки (CALEN). Затем разрешаем работу модуля АЦП – необходимо установить бит ADCEN в регистре ADCCTL.

2 Если предварительно была произведена процедура коррекции канала 2 и были вычислены поправочные коэффициенты, их необходимо внести в поля GAINTRIM и OFFTRIM регистра CHCTL2.

3 Для работы выберем секвенсор 0. Настроим его источник запуска – поле EM0 регистра EMUX должно быть равно Fh, т. к. запуск циклический.

4 В поле RQMAX регистра SRQCTL необходимо внести значение 0h, т. к. измерения будут проводиться по одному каналу.

5 Настраиваем канал для запроса. В регистр SRQSEL0 необходимо внести значение RQ0=2h.

6 Включаем усреднение сканированием по 16 опросам очереди. В регистре SEQ0_RQCTL нужно установить поля QAVGVAL=4, QAVGEN=1.

7 Для того чтобы опрос проводился каждую 1 мс (при f_{ACLK} = 33,3 МГц), необходимо внести в регистр SEQ0_RTMR значение (1000000 нс/30 нс) – 1 = 33332.

8 Разрешаем генерацию прерываний по каждой записи в FIFO – нужно установить бит SEQIM0 в регистре IM и проследить, чтобы поле ICNT регистра SCCTL оставалось равным нулю.

9 Разрешаем работу секвенсора. Для этого необходимо установит бит SEQEN0 в регистре SEQEN.

10 Запускаем измерения. Перед запуском проверяем флаг ADCRDY, чтобы быть уверенными в том, что модуль АЦП провел необходимые инициализации. Для того чтобы начать измерения, необходимо установить бит SYNC0 в регистре SEQSYNC и записать в бит GSYNC единицу.

11 Измерения будут сразу же запущены и будут запускаться каждую 1 мс далее. После каждого запуска будет проведено 16 измерений, результат усреднения которых будет записан в FIFO, и будет вызвано прерывание.

Работа режима проиллюстрирована на рисунке 21.20.

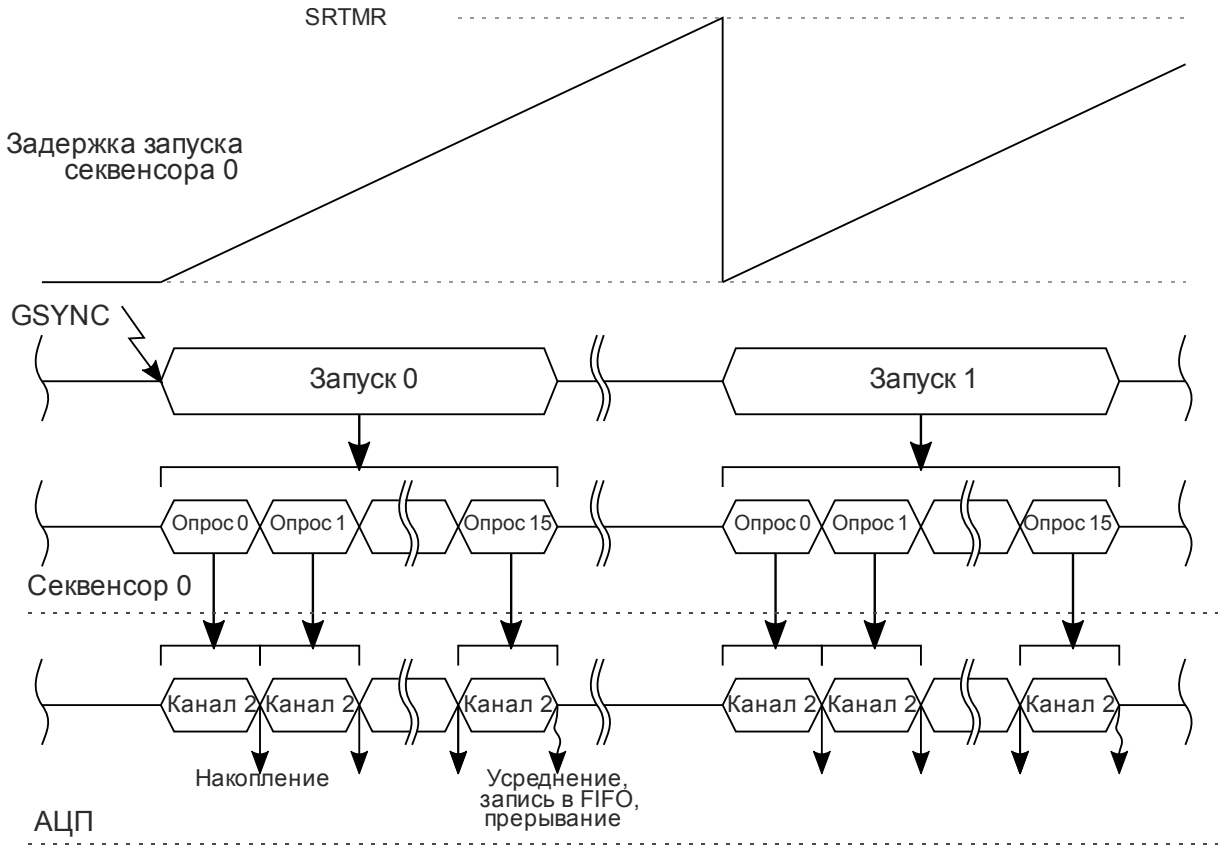


Рисунок 21.20 – Диаграммы циклического опроса канала

Пример 3 – Равномерно распределенные измерения по периоду таймера

Требуемый режим работы:

- запуск по таймеру 0 с частотой 10 кГц;

- четыре точки измерений равномерно распределенные по периоду;

- секвенсор 0;

- опросы по каналам 3 и 0;

- каждый опрос из четырех последовательных измерений и усреднения;

- прерывание каждые восемь записей в FIFO (по окончанию измерений в текущем периоде таймера).

Код, соответствующий настройке и запуску необходимого режима, представлен ниже.

```
// Настройка таймера
RCU->PCLKCFG0_bit.TMR0EN = 1;
RCU->PRSTCFG0_bit.TMR0EN = 1;
```

```
TMR0->LOAD = 4999;
TMR0->ADCSOC_bit.EN = 1;
//Настройка модуля АЦП – 12 бит и калибровка при включении
ADCSAR->ADCCTL_bit.SELRES = 3;
ADCSAR->ADCCTL_bit.CALEN = 1;
ADCSAR->ADCCTL_bit.ADCEN = 1;
//Настройка секвенсора
ADCSAR->EMUX_bit.EM0 = 5;
ADCSAR->SEQ[0].CCTL_bit.ICNT = 7;
ADCSAR->SEQ[0].CCTL_bit.RCNT = 3;
ADCSAR->SEQ[0].RTMR = 799;
ADCSAR->SEQ[0].RQCTL_bit.RQMAX = 1;
ADCSAR->SEQ[0].RQCTL_bit.QAVGVAL = 2;
ADCSAR->SEQ[0].RQCTL_bit.QAVGEN = 1;
ADCSAR->SEQ[0].RQSEL0_bit.RQ0 = 3;
ADCSAR->SEQ[0].RQSEL0_bit.RQ1 = 0;
ADCSAR->SEQEN_bit.SEQEN0 = 1;
// PLIC прерывание
ADCSAR->IM_bit.SEQIM0 = 1;
PLIC_IntEnable(ADC_SEQ0_IRQn);
// Запуск
while(!ADCSAR->ADCCTL_bit.ADCRDY);
TMR0->CTRL_bit.ON = 1;
```

1 Включаем тактирование таймера 0 и снимаем сброс – установим бит TMR0EN в регистрах PCLKCFG0 и PRSTCFG0 соответственно.

2 Для того чтобы таймер опустошался с частотой 10 кГц (при f_{PCLK}=50 МГц), необходимо внести в регистр таймера CAPCAOM[0].VAL значение (50000 кГц/10 кГц) – 1 = 4999.

3 Разрешаем генерацию таймером запросов на старт преобразования, установив бит EN в регистре ADC_IM таймера.

4 Перед разрешением работы АЦП устанавливаем 12-битный режим измерений (SELRES=3) и включаем механизм калибровки (CALEN). Затем разрешаем работу модуля АЦП – необходимо установить бит ADCEN в регистре ADCCTL.

5 Для работы выберем секвенсор 0. Настроим его источник запуска – поле EM0 регистра EMUX должно быть равно 2h, т. к. запуск по опустошению таймера 0.

6 Необходимое количество записей в FIFO для генерации прерывания – 8h, поэтому запишем в поле ICNT регистра SEQ0_CCTL значение 8 – 1=7h.

7 По каждому запуску должно совершиться три перезапуска с паузой в четверть периода опустошения таймера 0. В поле количества перезапусков RCNT регистра SCCTL внесем 3h.

В регистр задержки перезапуска SRTMR внесем значение 33333 кГц/(10 кГц×4) – 1 = 832.

8 В поле RQMAX регистра SRQCTL необходимо внести значение 1h, т. к. измерения будут проводиться по двум каналам.

9 Настраиваем каналы для запроса. В регистр SRQSEL необходимо внести значения RQ0=3h, RQ1=0h.

10 Включаем усреднение сканированием по четырем опросам очереди. В регистре SRQCTL нужно установить поля QAVGVAL=2, QAVGEN=1.

11 Разрешаем генерацию прерываний после каждой восьмой записи в FIFO – нужно установить бит SEQIM0 в регистре IM.

12 Разрешаем работу секвенсора. Для этого необходимо установить бит SEQEN0 в регистре SEQEN.

13 Запускаем измерения. Перед запуском проверяем флаг ADCRDY, чтобы быть

уверенными в том, что модуль АЦП провел необходимые инициализации. Для того чтобы их начать, необходимо включить таймер 0, установив бит ON в регистре CTRL таймера 0.

14 Измерения будут запускаться по каждому опустошению таймера 0. После каждого запуска будет проведено по три отложенных перезапуска. Т. к. после каждого перезапуска в FIFO будет попадать по 2 усредненных результата, то прерывание будет сгенерировано после записи последнего результата в последнем третьем перезапуске.

Работа режима проиллюстрирована на рисунке 21.21.

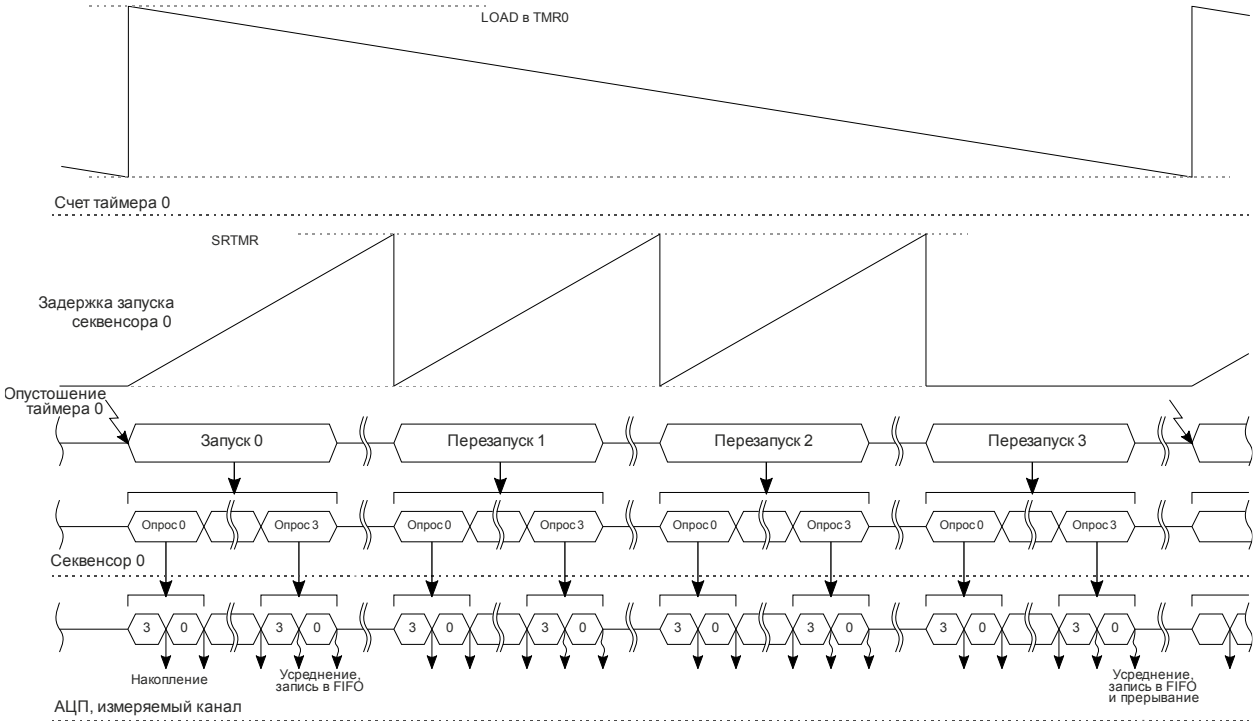


Рисунок 21.21 – Диаграммы равномерно распределенных измерений по периоду таймера

22 Модуль сигма-дельта АЦП

Модуль сигма-дельта АЦП включает в себя 8 отдельных каналов разрядностью 16 бит. Каждый канал включается и настраивается отдельно, со своим регистром результата преобразования. Модуль сигма-дельта АЦП входит в состав аналогового домена питания и использует выводы питания VCC2 и GND2. Структурная схема АЦП представлена на рисунке 22.1.

16-разрядный аналого-цифровой преобразователь (АЦП), предназначенный для использования в промышленных системах управления питанием, системах управления электродвигателями или в качестве многоканальной системы ввода данных. Каждый канал имеет программируемый входной усилитель с диапазоном коэффициента усиления от 0 до 38 дБ. Частота преобразования может быть запрограммирована на четыре значения: 64, 32, 16 или 8 кГц (при входной тактовой частоте 16,384 МГц).

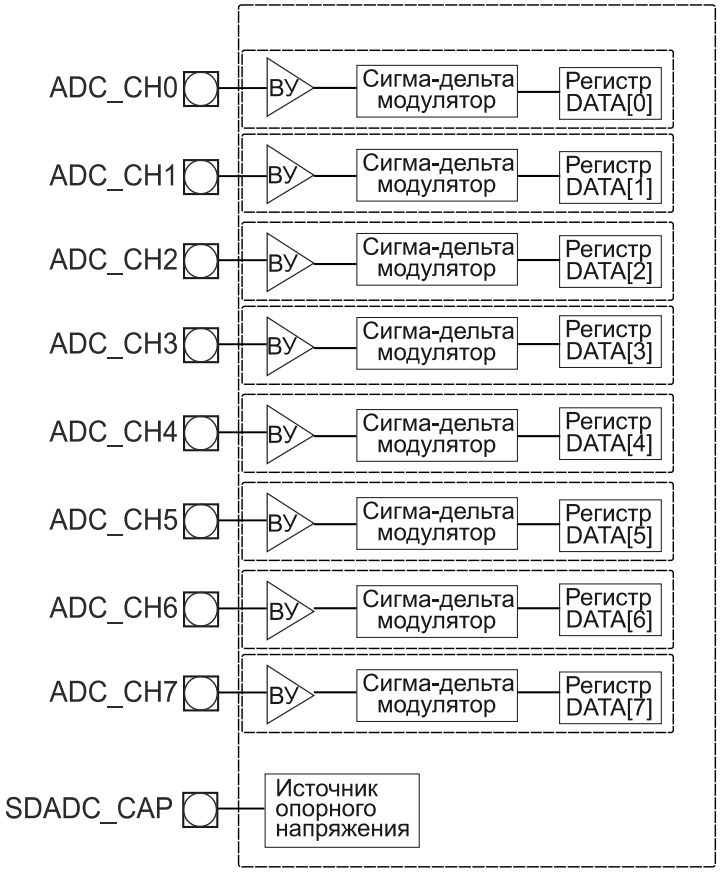


Рисунок 22.1 – Структурная схема сигма-дельта АЦП

22.1 Источник опорного напряжения

Источник опорного напряжения выполнен с использованием напряжения запрещенной зоны полупроводника. Напряжение внутреннего источника подается на вывод SDADC_CAP, который используется для подключения внешнего конденсатора. В случае использования этого вывода в качестве опорного источника, необходима обязательная буферизация. На выходе внутреннего источника опорного напряжения формируется напряжение (1,250±0,125) В. Допускается подключение внешнего источника опорного напряжения к выводу SDADC_CAP не более (VCC2 / 2) В.

22.2 Программируемый усилитель

Усилитель с программируемым коэффициентом усиления представляет собой схему на переключаемых конденсаторах и является частью сигма-дельта модулятора. Переключаемые конденсаторы коммутируются с частотой fADCCLK/8. Коэффициенты усиления усилителя приведены в таблице 22.1. Уровень выходного напряжения усилителя не должен превышать максимально допустимого уровня входного сигнала сигма-дельта модулятора.

Таблица 22.1– Установка программируемого входного усилителя

IxGS2	IxGS1	IxGS0	Усиление, дБ
0	0	0	0
0	0	1	6
0	1	0	12
0	1	1	18
1	0	0	20
1	0	1	26
1	1	0	32
1	1	1	38

22.3 Запуск преобразования

Для включения АЦП необходимо установить биты ENB и PUREF регистра CTRL. Они отвечают за включение самого АЦП и внутреннего источника опорного напряжения. После включения АЦП в регистре ENB можно выбрать каналы, по которым будет выполняться преобразование. Как только выбран хотя бы один канал, АЦП начинает генерировать внутренний стробирующий сигнал ECD с частотой 1/50 от частоты тактового сигнала АЦП ADCCLK(ACLK). Если АЦП было сброшено или выключены все каналы, генерация строба начнется заново при следующем включении. При включении канала до получения корректного результата преобразования необходимо пропустить несколько тактов стробирующего сигнала ECD. За количество тактов отвечает поле WTCYC регистра CTRL. По умолчанию оно равно 3, это значит, что будет пропущено 3 импульса строба, а на 4 данные преобразования занесены в регистр DATA соответствующего канала.

Для мониторинга состояния готовности каналов можно использовать регистр READY. Если АЦП включено и канал разрешен в регистре ENB, то по истечению 3 тактов ECD установится соответствующий бит в регистре READY.

Для отслеживания новых преобразований можно воспользоваться регистром DATAUPD. Если канал был разрешен, прошло нужное количество тактов после включения, выставлен требуемый режим канала, по приходу строба ECD данные будут занесены в соответствующий каналу регистр DATA. А в регистре DATAUPD будет установлен бит, соответствующий каналу. Сбросить биты регистра DATAUPD можно записав туда «1».

22.4 Режимы

Во время работы, активные каналы повышают величину потребляемой модулем мощности. В целях снижения энергопотребления предусмотрено несколько режимов включения каналов. Режим работы выбирается для каждого канала индивидуально в регистре MODE.

Режим «1» − одиночный запуск с последующим выключением канала. При выборе данного режима канал дождется сигнала READY после включения, затем выполнит одно преобразование (с занесением данных в регистр DATA и установкой соответствующего бита в регистре DATAUPD). По получению результата будут сброшены в «0» бит ENB и биты MODE соответствующие каналу. Для нового запуска необходимо заново устанавливать данные биты ENB и MODE. Соответственно между запусками канал будет

выключен, энергопотребление снижено, но для получения нового результата потребуется больше времени.

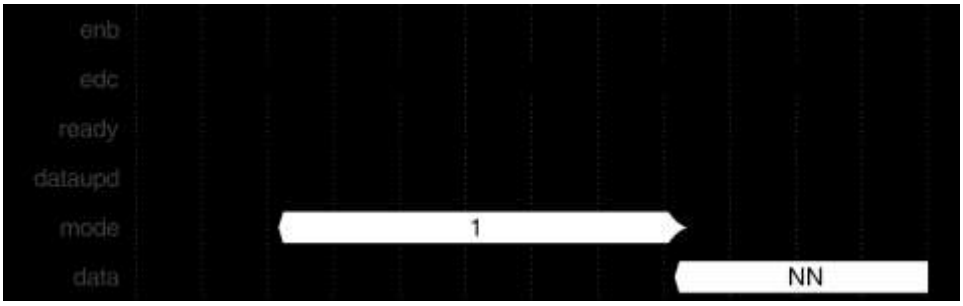


Рисунок 22.2 – Диаграмма переключения внутренних сигналов в режиме 1.

Режим «2» − одиночный запуск без выключения канала. Данный режим отличается от предыдущего режима только тем, что не будет сброшен бит ENB. Соответственно данные будут однократно записаны в регистр DATA, канал продолжит работу и при следующем запуске (при условии заново установленных бит режима канала в регистре MODE) результат будет получен при первом же стробе ECD.

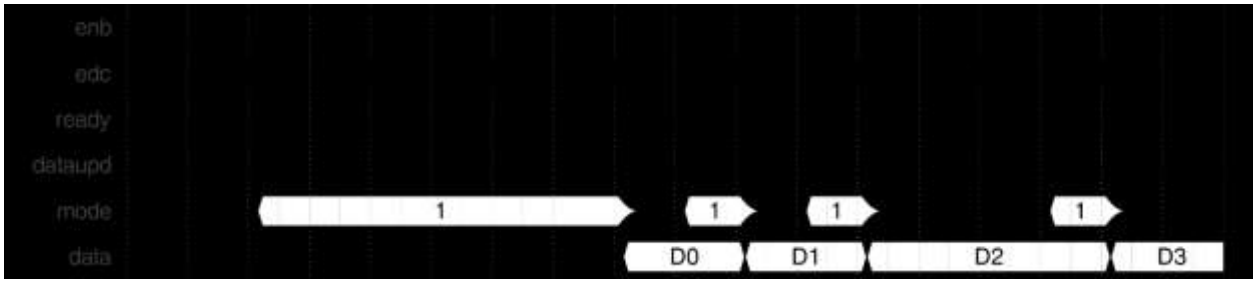


Рисунок 22.3 – Диаграмма переключения внутренних сигналов в режиме 2.

Режим «3» − постоянный режим работы. После получения данных канал не отключается, и обновление регистра DATA будет происходить по каждому стробу ECD.

Рисунок 22.4 – Диаграмма переключения внутренних сигналов в режиме 3.

22.5 Передаточная функция АЦП

На рисунках 22.5 и 22.6 приведены передаточные функции для дифференциального и однопроводного включения.

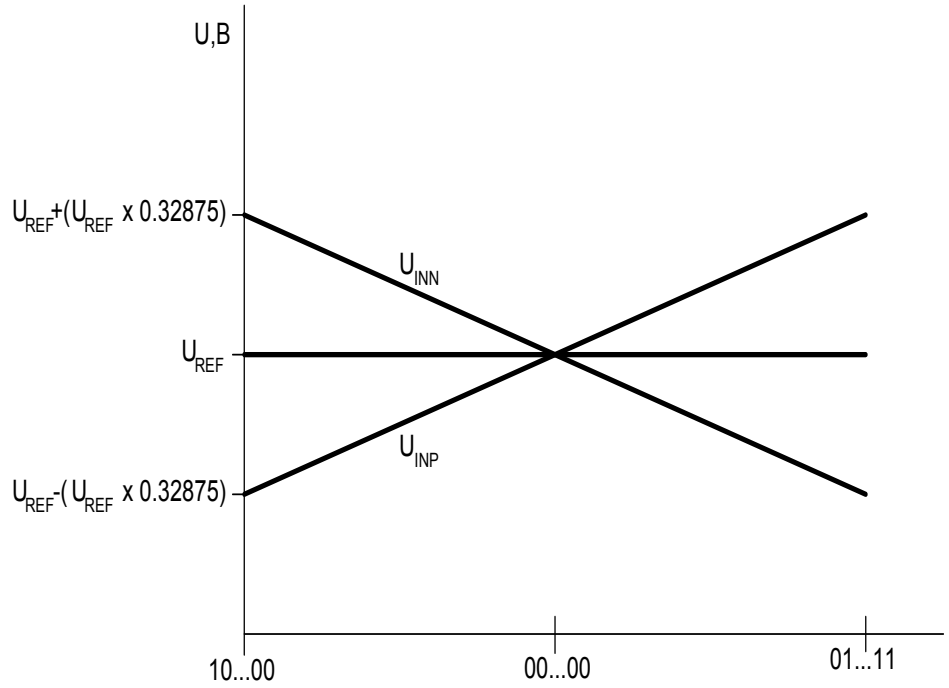


Рисунок 22.5 – Передаточная функция АЦП в режиме дифференциального входа

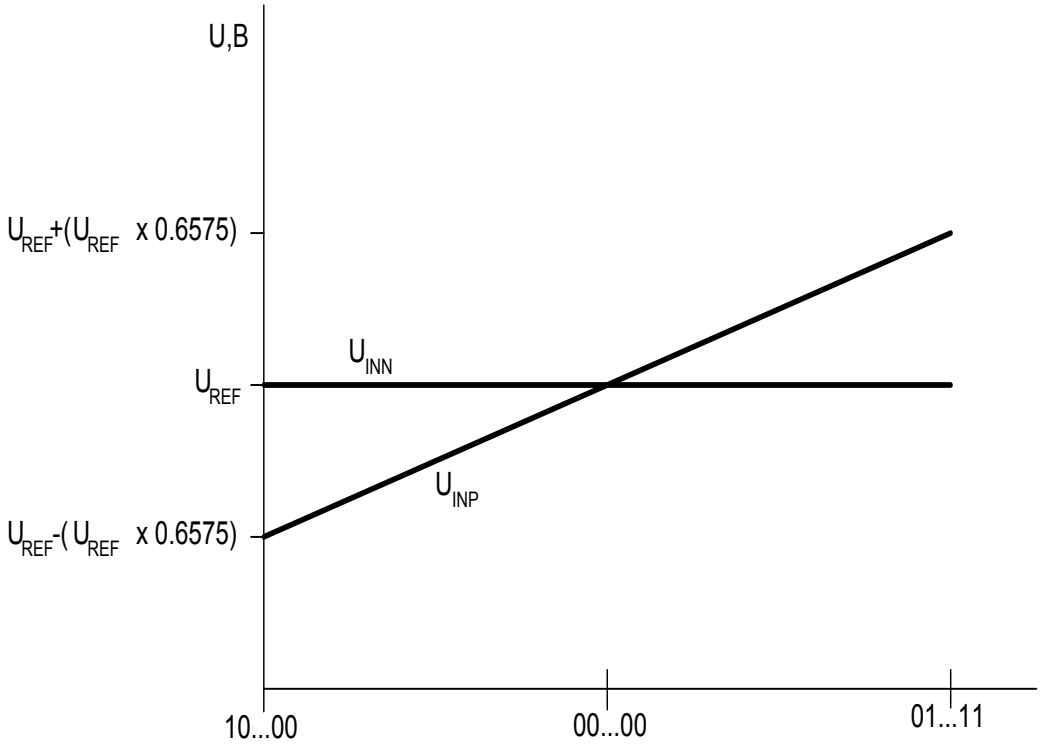


Рисунок 22.6 – Передаточная функция АЦП в режиме однопроводного входа

22.6 Прерывания

Модуль АЦП может генерировать прерывание по получению результата преобразования любым из каналов. Для этого необходимо указать маску каналов, которые активируют сигнал прерывания. Маска задается в регистре IM. Состояние сработавших прерываний можно прочитать в регистре MIS. Их сброс осуществляется путем записи «1»

в соответствующие разряды регистра IC.

23 Датчик температуры

Микроконтроллер содержит датчик температуры, позволяющий измерять температуру кристалла во время работы. Датчик температуры содержит элементы, напряжение на которых линейно убывает с ростом температуры (СТАТ). В качестве СТАТ элементов используется два последовательно соединенных диода. Выход датчика подключается на вход канала № 10 АЦП последовательного приближения.

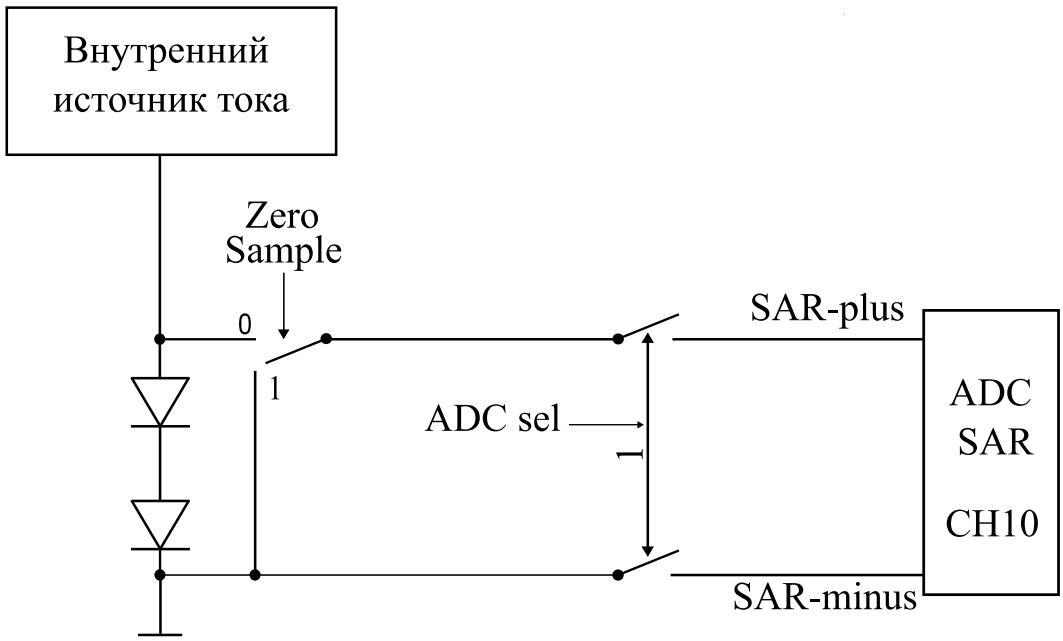


Рисунок 23.1 – Структурная схема датчика температуры

Характеристики

Основные характеристики датчика температуры:

- напряжение питания аналоговой части (VCC2): от 2,5 В до 3,6 В;
- напряжение питания цифровой части (VCC1): от 1,08 В до 1,32 В;
- погрешность определения температуры: не более 5 °С;
- диапазон измерения температуры: от – 40 °С до 85 °С;
- диапазон выходного напряжения: от 0,5 В до 1,6 В;
- выходное сопротивление: 40 кОм.

Зависимость напряжения на датчике от температуры представлена на рисунке 23.2. Данные получены при следующих значениях: VCC1 = 3,3 В, VCC2 = 3,3 В, AREF = 3,3.

Значение температуры (°С) вычисляется по формуле 23.1.

$$T = K \times (U_0 - U_{ADC}) \quad (23.1)$$

где K – угловой коэффициент, K = 250;

U_0 – значение напряжения на датчике температуры (В) при температуре 0°С, U_0 = 1.41;

U_{ADC} – значение напряжения на датчике температуры (В), вычисляемое по формуле 23.2.

$$U_{ADC} = \frac{U_{AREF} \times D_{ADC}}{2^n} \quad (23.2)$$

где U_{AREF} – значение опрного напряжения на выводе AREF;
D_{ADC} – результат АЦП канала датчика температуры;
n – разрядность АЦП.

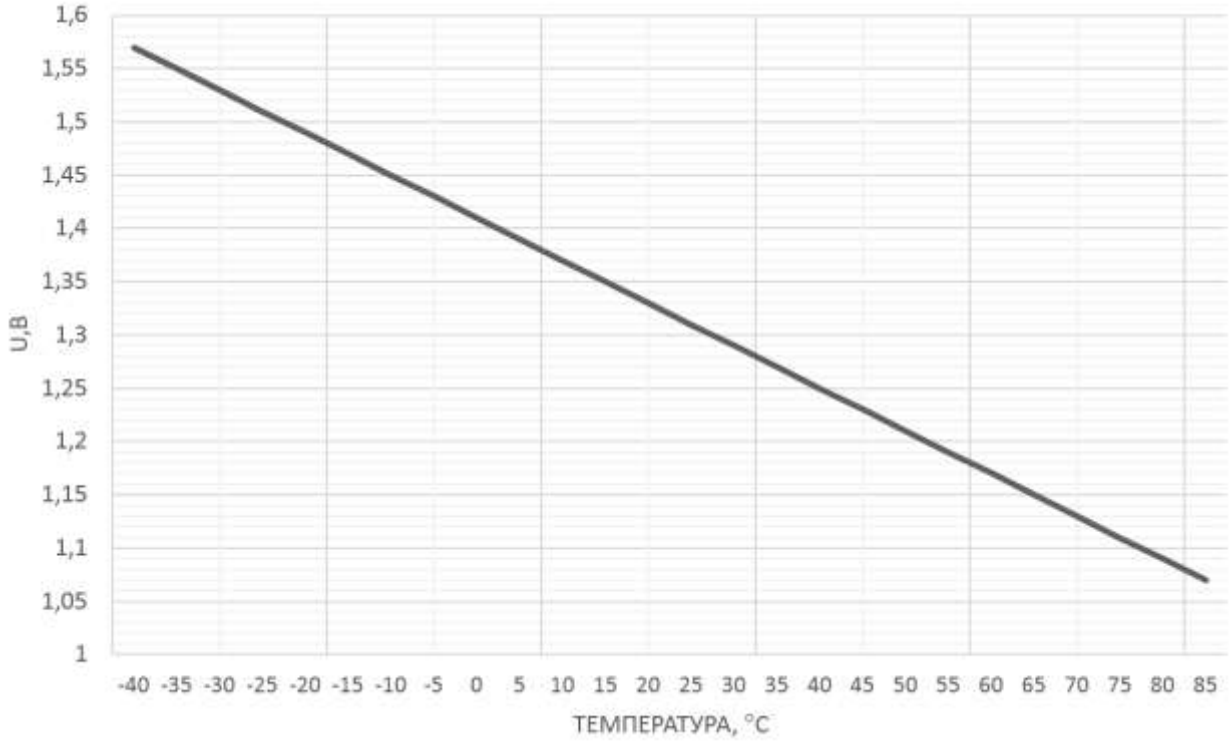


Рисунок 23.2 – График зависимости напряжения на выходе датчика от температуры

Для вычисления значения температуры по результату, полученному с канала АЦП, подключенного к датчику температуры, можно использовать формулу 23.3.

$$T = 250 \times \left(1{,}41 - \frac{U_{AREF} * D_{ADC}}{2^n}\right) \quad (23.3)$$

Калибровка формулы вычисления температуры

Для повышения точности вычисления температуры необходимо произвести калибровку значений K и U_0.

Для получения значения U_0 необходимо измерить значение датчика температуры в вольтах при темепратуре 0°C.

Для получения значения K необходимо измерить значение датчика температуры при температуре отличной от 0°C, например, при 20°C и выислить по формуле 23.4.

$$K = \frac{T}{U_0 - U_{ADC}} \quad (23.4)$$

Инициализация датчика температуры

Для включения датчика, необходимо в регистр CTRL записать значение 0x12.

В связи с тем, что датчик температуры имеет высокое выходное сопротивление, для получения достоверных значений напряжения необходимо увеличить время подключения к цепи зарядной емкости АЦП путем увеличения количества дополнительных тактов в регистре CHDELAY[10].

Количество дополнительных тактов необходимо выбирать из таблицы 21.3 учитывая выходное сопротивление датчика 40 кОм.

24 Модуль аналогового компаратора

Аналоговый компаратор сравнивает два аналоговых сигнала и формирует логический выходной сигнал с результатом сравнения. Выходной сигнал может быть использован как внутри контроллера, так и подан на внешний вывод. Также, компаратор может формировать прерывания и/или сигналы запуска секвенсоров модуля АЦП. Модуль компаратора содержит два идентичных аналоговых компаратора и один ЦАП с двумя настраиваемыми уровнями выходного напряжения.

Компаратор может сравнивать напряжение на внешнем выводе микросхемы с напряжением на другом выводе, а также с напряжением на встроенном ЦАП.

Аналоговый компаратор подключен к домену батарейного питания и может функционировать при отключении основного питания на выводах VCC1, VCC2.

Компаратор может формировать сигнал пробуждения для вывода микроконтроллера из состояния сна.

Структурная схема компаратора показана на рисунке 24.1.

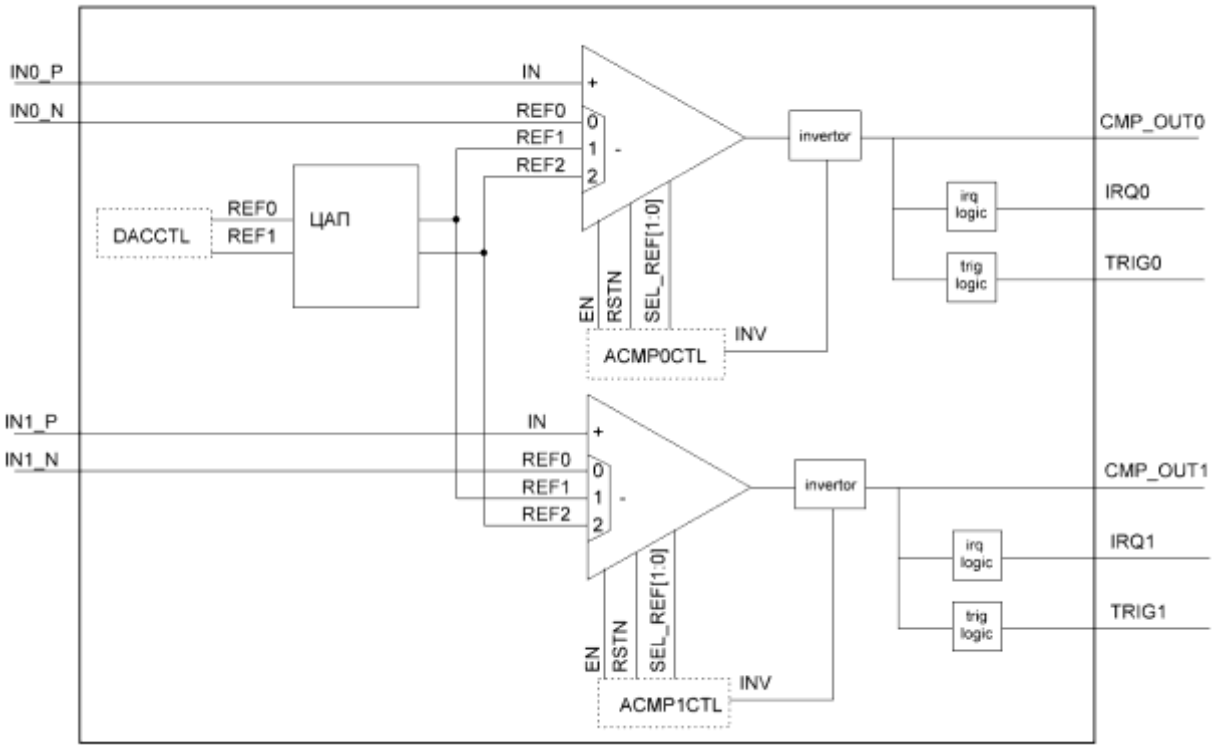


Рисунок 24.1 – Структурная схема компаратора

Логика сравнения уровней аналоговых сигналов осуществляется формулами 24.1 и 24.2.

$$VIN- < VIN+, CMP\_OUT = 1. \quad (24.1)$$

$$VIN- > VIN+, CMP\_OUT = 0. \quad (24.2)$$

В этой формуле VIN+ напряжение на выводе INn_P, а VIN − может выбираться из нескольких источников, заданных в регистре ACMPnCTL поле SELREF.

- вывод INn_N;
- опорное напряжение с 0 канала встроенного ЦАП;
- опорное напряжение с 1 канала встроенного ЦАП.

При необходимости можно инвертировать выход компаратора установив бит INV регистра ACMPnCTL.

Опорное напряжение встроенных ЦАП программируется через регистр DACCTL. Можно настроить два независимых выходных канала.

Опорное напряжение вычисляется по формуле 24.3.

$$Vref = \frac{AVDD \times REF_n}{16} \quad (24.3)$$

Блок может генерировать прерывание по каждому из каналов. Событие для генерации выбирается полем INTSRC регистра ACMPnCTL. Аналогично сигналу прерывания генерируется сигнал для запуска секвенсоров АЦП. Событие для генерации выбирается полем TRIGSRC регистра ACMPnCTL.

В качестве событий могут выступать следующие состояния выхода CMP_OUTn:

- положительный фронт;
- отрицательный фронт;
- любой фронт;
- логический уровень нуля;
- логический уровень единицы.

В регистре INTEN разрешается генерация соответствующего прерывания на системный уровень. Регистр RIS содержит статус прерываний без учета маски разрешения INTEN. Регистр MIS содержит статус прерываний с учетом маски разрешения INTEN. Для сброса прерывания необходимо записать «1» в соответствующий бит регистра ICLR.

Событие пробуждения микроконтроллера

Модуль аналоговых компараторов может формировать сигнал пробуждения микроконтроллера. Для этого необходимо в битовом поле TRIGSRC регистра ACMP0CTL и ACMP1CTL для выбора по каждому из каналов, одного из событий формирования сигнала пробуждения микроконтроллера. При возникновении данного события будет сформировано прерывание блока RTC.

Возможность пробуждения микроконтроллера модулем аналоговых компараторов можно использовать для отслеживания уровня питания микроконтроллера.

25 Приемопередатчик UART

В состав микроконтроллера входят пять идентичных универсальных асинхронных приемопередатчиков UART0 – UART4.

В состав приемопередатчика входят два буфера типа FIFO. Буфер приемника имеет разрядность 12, буфер передатчика – разрядность восемь. Каждый буфер может хранить до 32 байт данных, и каждый буфер может быть сконфигурирован (программно) как 32-байтный или как однобайтный.

Приемопередатчик обеспечивает:

- независимое маскирование прерываний от буфера передатчика, буфера приемника, по таймауту приемника, по изменению линий состояния модема, а также в случае обнаружения ошибки;
- возможность деления тактовой частоты в диапазоне от 1 до 65535 (допускается использование нецелых коэффициентов деления, что позволяет использовать любой опорный генератор с частотой более 3,6864 МГц);
- поддержку прямого доступа к памяти;
- поддержку функции управления модемом (сигналы CTS, DCD, DSR, RTS, DTR и RI).

Приемопередатчик реализует:

- передачу данных длиной от 5 до 8 бит со скоростью до 3000 Кбит/с;
- контроль четности (проверочный бит выставляется по четности, нечетности, имеет фиксированное значение либо не передается);
- формирование одного или двух стоповых бит;
- обнаружение ложных стартовых битов;
- формирование и обнаружение сигнала разрыва линии.

Функциональные возможности

Режим работы приемопередатчика и скорость обмена данными контролируются регистром LCRH и регистрами делителя IBRD и FBRD.

Устройство может формировать следующие сигналы:

- независимые маскируемые прерывания от приемника (в том числе по таймауту), передатчика, а также по изменению состояния модема и в случае обнаружения ошибки;
- общее прерывание, возникающее в случае, если возникло одно из независимых немаскированных прерываний;
- сигналы запроса на прямой доступ к памяти для совместной работы с контроллером DMA.

В случае возникновения ошибки в структуре сигнала, четности данных, а также разрыва линии, соответствующий бит ошибки устанавливается и сохраняется в буфере приемника. В случае переполнения буфера приемника также устанавливается соответствующий бит, а буфер становится недоступным для записи.

25.1 Функционирование блока UART

На рисунке 25.1 показана упрощенная функциональная схема приемопередатчика.

Генератор тактового сигнала приемопередатчика формирует синхросигнал последовательного обмена данными, который представляет собой последовательность импульсов с шириной, равной одному периоду сигнала UARTCLK, и частотой в 16 раз превышающей частоту передачи данных.

Буфер передатчика предназначен для хранения данных, полученных от ЦП, до тех пор, пока они не будут переданы внешнему устройству.

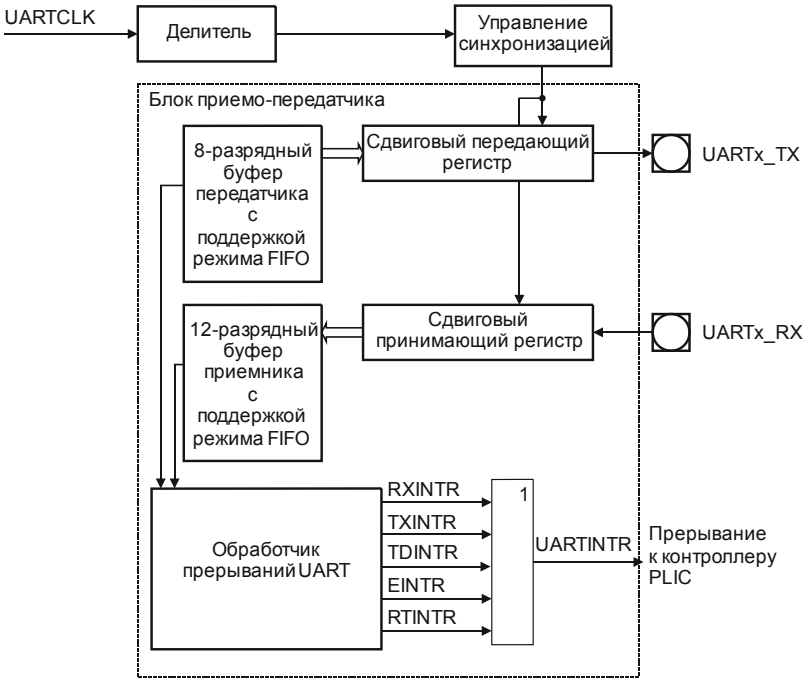


Рисунок 25.1 – Функциональная схема приемопередатчика

Буфер приемника предназначен для хранения данных и кодов ошибки (принятых от внешнего устройства) до тех пор, пока они не будут прочитаны ЦП.

Обработчик прерываний генерирует независимые маскируемые прерывания с активным высоким уровнем. Кроме того, формируется комбинированное прерывание путем объединения независимых прерываний по схеме ИЛИ. Сигнал прерывания передается на контроллер PLIC.

Сброс модуля

Приемопередающая логика модуля, управляющие регистры и FIFO по умолчанию находятся в сбросе. Снять сброс можно путём установки бита RSTDIS в соответствующем регистре UARTCFGi (i – номер модуля от 0 до 4) блока RCU.

Синхронизация

Существует ограничение на соотношение между частотами тактовых сигналов PCLK и UARTCLK, представленное формулой 25.1.

$$f_{UARTCLK} \leq 5/3 \times f_{PCLK} \qquad (25.1)$$

Например, для достижения максимальной скорости передачи данных 921600 бод (при $f_{UARTCLK} = 921600 \times 16 = 14{,}7456$ МГц) частота f_{SYSCLK} должна быть не менее 8,84736 МГц.

Для точной настройки частоты передачи данных используются два делителя – один управляется регистром UARTCFGx блока RCU, второй находится внутри модуля UART.

Коэффициент деления второго делителя имеет целую и дробную части, которые задаются регистрами IBRD и FBRD. Возможность задания нецелых коэффициентов деления позволяет осуществлять обмен данными на стандартных скоростях, используя в качестве источника тактовый сигнал с произвольной частотой более 3,6864 МГц.

Коэффициент деления К частоты сигнала UARTCLK рассчитывается по формуле 25.2.

$$K = f_{UARTCLK}/(16 \times baudrate), \quad (25.2)$$

где $f_{UARTCLK}$ – частота сигнала синхронизации блока UART – UARTCLK, Гц;

baudrate – скорость передачи, бод.

Получившееся дробное десятичное число следует разделить на две части – целую и дробную.

Целая часть после преобразования в двоичный формат записывается в регистр IBRD.

Дробная часть умножается на 64 и округляется до ближайшего целого числа. Полученное число преобразовывается в двоичный формат и записывается в регистр FBRD.

Для примера, пусть требуемая скорость передачи данных 230 400 бит/с и частота тактового сигнала UARTCLK равна 4 МГц. Тогда:

$$K = (4 \times 10^6)/(16 \times 230400) = 1{,}085.$$

Получившееся число разбивается на две части – 1 и 0,085.

В регистр IBRD записывается значение 0001h.

Значение (0,085 × 64) округляется и преобразовывается в 05h для записи в регистр FBRD.

Таким образом, реальные значения коэффициента деления частоты и скорости передачи будут следующими:

$$K = 1 + 5/64 = 1{,}078,$$

$$baudrate = (4 \times 10^6)/(16 \times 1{,}078) = 231911 \text{ бит/с}.$$

Ошибка установки скорости:

$$\Delta = ((231911 - 230400)/230400) \times 100\ \% = 0{,}656\ \%.$$

Максимальная ошибка установки скорости передачи данных:

$$\Delta = (1/64) \times 100\ \% = 1{,}56\ \%.$$

Такая ошибка возникает в случае K = 1, при этом разница накапливается в течение 64 тактовых интервалов.

Содержимое регистров LCRH, IBRD и FBRD обновляется при записи в регистр LCRH. Таким образом, для того, чтобы новые параметры коэффициента деления вступили в силу, после их записи в регистры IBRD и FBRD, необходимо осуществить запись в регистр LCRH и только в такой последовательности.

Примечание – Изменение содержимого регистров IBRD, FBRD и LCRH допускается только во время, когда приемопередатчик запрещен и не осуществляется передача/прием байта.

Передача и прием данных

Данные для передачи заносятся в буфер передатчика посредством записи в регистр DR. После записи хотя бы одного байта в буфер передатчика устанавливается флаг BUSY в регистре FR. Это состояние флага сохраняется, пока буфер передатчика не пуст (даже если работа приемопередатчика запрещена). Далее, если работа приемопередатчика разрешена

(установлены биты UARTEN и TXE регистра CR), начинается передача информационного кадра с параметрами, указанными в регистре управления линией LCRH. Передача данных продолжается до опустошения буфера передатчика (до окончания передачи всех байт). По окончании передачи сбрасывается флаг BUSY.

При приеме байта данных (установлены биты UARTEN и RXE регистра CR) для каждого бита производятся три выборки уровня, и решение о значении бита принимается по мажоритарному принципу.

В случае если приемник находился в неактивном состоянии (постоянный высокий уровень сигнала на линии UARTx_RX), и произошла смена уровня входного сигнала с высокого на низкий (стартовый бит), включается счетчик, тактируемый внутренним сигналом, после чего отсчеты сигнала на входе приемника регистрируются каждые восемь тактов.

Стартовый бит считается достоверным в случае, если сигнал на линии UARTx_RX сохраняет низкий логический уровень в течение восьми периодов внутреннего синхросигнала с момента включения счетчика. В противном случае переход в ноль рассматривается как ложный старт и игнорируется.

После обнаружения достоверного стартового бита очередной бит данных фиксируется каждые 16 отсчетов тактового сигнала. Производится регистрация всех бит данных (согласно запрограммированным параметрам) и бита четности (если включен режим контроля четности).

По окончании приема байта производится проверка присутствия корректного стопового бита (высокий логический уровень сигнала UARTx_RX). После этого байт заносится в буфер приемника вместе с тремя битами признаков ошибки, см. рисунок 25.2, и битом переполнения буфера.

В 12-разрядной ячейке буфера байт данных располагается в области младших восьми бит, три бита признаков ошибки – в битах с 8 по 10.

Флаг переполнения буфера приемника выставляется в том случае, если к моменту, когда очередной кадр данных полностью принят, буфер уже заполнен. В этом случае принятый кадр остается в сдвиговом принимающем регистре и, в случае приема следующего кадра данных, будет потерян.

Как только в буфере приемника освобождается место для записи, кадр данных, находящийся в сдвиговом регистре, переписывается в буфер, а флаг переполнения сбрасывается.

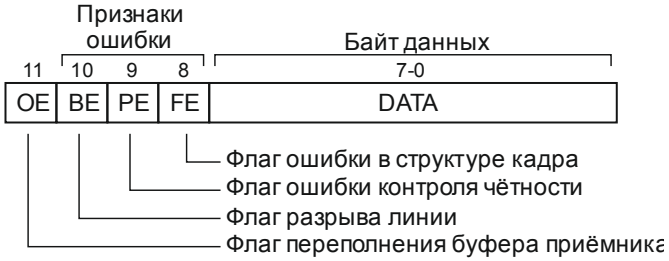


Рисунок 25.2 – 12-разрядная ячейка принимающего буфера

Данные из буфера приемника можно прочитать посредством регистра DR. Состояние признаков ошибки и флага переполнения определяется чтением регистра RSR и относится к последнему байту, считанному из регистра DR, в связи с этим регистр DR всегда должен считываться первым.

Все флаги сбрасываются одновременно записью любого значения в регистр RSR или после сброса устройства.

Примечания

1 Необходимо запрещать работу приемопередатчика перед перепрограммированием его регистров управления. Если приемопередатчик переводится в отключенное состояние во время передачи или приема, то перед остановкой он завершает выполняемую операцию.

2 Целостность данных в буферах передатчика и приемника не гарантируется, если установился флаг BRK (разрыв линии), или если программное обеспечение произвело остановку приемопередатчика после его повторного перевода в разрешенное состояние.

25.2 Режим модема

Приемопередатчик может использоваться как оконечное устройство или как оборудование передачи данных. Сигналы модема в режиме оконечного устройства и их назначение представлены в таблице 25.1.

Таблица 25.1 – Назначение сигналов в режиме модема

Сигнал	Назначение в зависимости от режима работы		Режим работы вывода
	Оконечное устройство	Оборудование передачи данных	
UARTx_RTS	Готов к передаче данных	Запрос передачи данных	Выход
UARTx_CTS	Запрос передачи данных	Готов к передаче данных	Вход
UARTx_DTR	Приемник данных готов	Источник данных готов	Выход
UARTx_DSR	Источник данных готов	Приемник данных готов	Вход
UARTx_DCD	Обнаружен информационный сигнал	–	Вход
UARTx_RI	Индикатор вызова	–	Вход

Аппаратное управление потоком данных

Программно активируемый режим аппаратного управления потоком данных позволяет контролировать (приостанавливать и возобновлять) информационный обмен с помощью выводов UARTx_RTS и UARTx_CTS. Иллюстрация взаимодействия двух устройств последовательной связи с аппаратным управлением потоком данных представлена рисунке 25.3.

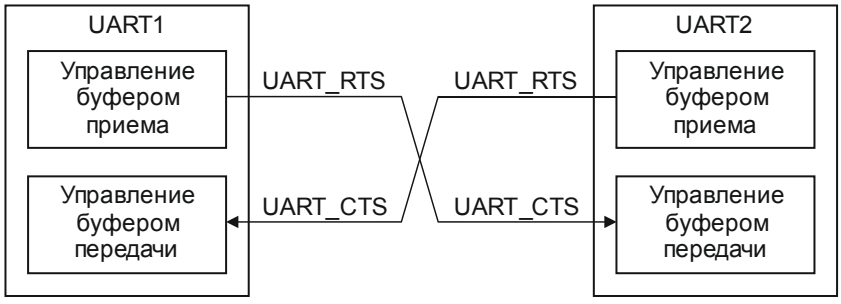


Рисунок 25.3 – Взаимодействие двух устройств последовательной связи с аппаратным управлением потоком данных

Если разрешено управление потоком данных по сигналу RTS, вывод UARTx_RTS переводится в активное состояние только после того, как в буфере приемника появляется заданное количество свободных ячеек.

Если разрешено управление потоком данных по сигналу CTS, передача данных осуществляется только после перевода вывода UARTx_CTS в активное состояние.

Режим аппаратного управления потоком данных задается путем задания битов RTSEN и CTSEN в регистре управления CR.

Примечание – В случае если выбран режим управления потоком данных по сигналу RTS, программное обеспечение не может использовать бит RTSEN для проверки состояния линии RTS.

Логика управления потоком данных по сигналу RTS использует данные о превышении уровня заполнения буфера приемника. Сигнал на выводе UARTx_RTS переводится в активное состояние только после того, как в буфере приемника появляется заданное количество свободных ячеек. После достижения порогового уровня заполнения буфера приемника сигнал RTS снимается (переводится в пассивное состояние), указывая, таким образом, на отсутствие свободного места для сохранения принятых данных. При этом дальнейшая передача данных должна быть прекращена по завершении передачи текущего кадра.

Обратно в активное состояние сигнал RTS переводится после считывания данных из буфера приемника в количестве, достаточном для того, чтобы заполнение буфера оказалось ниже порогового уровня.

В случае если управление потоком данных по сигналу RTS запрещено, но при этом работа приемопередатчика разрешена, прием будет осуществляться до полного заполнения буфера приемника либо до завершения передачи данных.

В случае выбора одного из режимов с управлением потоком данных по сигналу CTS передатчик осуществляет проверку состояния вывода UARTx_CTS перед началом передачи очередного байта данных. Передача осуществляется только в случае, если данная линия активна, и продолжается до тех пор, пока активное состояние линии сохраняется и буфер передатчика не пуст.

При переходе вывода UARTx_CTS в неактивное состояние модуль завершает выдачу текущего передаваемого кадра, после чего передача данных прекращается.

Если управление потоком данных по сигналу CTS запрещено, и при этом работа приемопередатчика UART разрешена, данные будут выдаваться до опустошения буфера передатчика.

25.3 Интерфейс прямого доступа к памяти

Приемопередатчик оснащен интерфейсом подключения к контроллеру прямого доступа к памяти. Работа в данном режиме контролируется регистром управления DMACR. Интерфейс DMA включает в себя шесть сигналов.

Сигнал RXDMASREQ (для приема) – запрос передачи отдельного символа, инициируемый приемопередатчиком. Размер символа в режиме приема данных – до 12 бит. Сигнал переводится в активное состояние в случае, если буфер приемника содержит, по меньшей мере, один символ.

Сигнал RXDMABREQ (для приема) – запрос блочного обмена данными, инициируемый приемопередатчиком. Сигнал переходит в активное состояние в случае, если заполнение буфера приемника превысило заданный порог. Порог программируется индивидуально для каждого буфера посредством полей регистра IFLS.

Сигнал RXDMACLR (для приема) – сброс запроса на DMA, инициируемый приемопередатчиком. В случае если был запрошен блочный обмен данными, сигнал сброса формируется в ходе передачи последнего символа данных в блоке.

Сигнал TXDMASREQ (для передачи) – запрос передачи отдельного символа, инициируемый приемопередатчиком. Размер символа в режиме передачи данных – до

восьми бит. Сигнал переводится в активное состояние в случае, если буфер передатчика содержит, по меньшей мере, одну свободную ячейку.

Сигнал TXDMABREQ (для передачи) – запрос блочного обмена данными, инициируемый приемопередатчиком. Сигнал переводится в активное состояние в случае, если заполнение буфера передатчика ниже заданного порога. Порог программируется индивидуально для каждого буфера посредством полей регистра IFLS.

Сигнал TXDMACLR (для передачи) – сброс запроса на DMA, инициируемый контроллером DMA с целью сброса принятого запроса. В случае если был запрошен блочный обмен данными, сигнал сброса формируется в ходе передачи последнего символа данных в блоке.

Сигналы блочного и одноэлементного обмена данными не являются взаимно исключающими, они могут быть инициированы одновременно. Например, в случае если заполнение данными буфера приемника превышает пороговое значение, то формируется как сигнал запроса одноэлементного обмена, так и сигнал запроса блочного обмена данными. В случае если количество данных в буфере приема меньше порогового значения, формируется только запрос одноэлементного обмена. Это бывает полезно в ситуациях, при которых объем данных меньше размера блока.

Пусть, например, нужно принять 19 символов, а порог заполнения буфера установлен равным четырем. Тогда контроллер DMA осуществит четыре передачи блоков по четыре символа, а оставшиеся три символа передаст в ходе трех одноэлементных обменов, поскольку для них блок UART не может инициировать процедуру блочного обмена.

Каждый инициированный приемопередатчиком сигнал запроса DMA остается активным до момента его сброса соответствующим сигналом DMACLR.

После снятия сигнала сброса модуль приемопередатчика вновь получает возможность сформировать запрос на DMA в случае выполнения описанных выше условий. Все запросы DMA снимаются после запрета работы приемопередатчика, а также в случае установки в ноль бит управления DMATXDMAE или RXDMAE в регистре управления DMACR.

В случае запрета буферов устройство способно передавать и принимать только одиночные символы, и, как следствие, контроллер может инициировать DMA только в одноэлементном режиме. При этом модуль в состоянии формировать только сигналы управления RXDMASREQ и TXDMASREQ.

Когда буферы включены, обмен данными может производиться в ходе как одноэлементных, так и блочных передач данных, в зависимости от установленной величины порога заполнения буферов и их фактического заполнения.

В таблице 25.2 указаны значения параметров срабатывания запросов блочного обмена RXDMABREQ и TXDMABREQ в зависимости от порога заполнения буфера.

Таблица 25.2 – Параметры срабатывания запросов блочного обмена данными в режиме DMA

Пороговый уровень	Количество незаполненных ячеек буфера передатчика	Количество заполненных ячеек буфера приемника
1/8	28	4
1/4	24	8
1/2	16	16
3/4	8	24
7/8	4	28

В регистре управления DMACR предусмотрен бит DMAONERR, который позволяет запретить DMA от приемника в случае активного состояния линии прерывания по обнаружению ошибки EINTR. При этом соответствующие линии запроса DMA – RXDMASREQ и RXDMABREQ – переводятся в неактивное состояние (маскируются) до

сброса EINTR. На линии запроса DMA, обслуживающие передатчик, состояние EINTR не влияет.

25.4 Прерывания

В модуле предусмотрено семь маскируемых источников прерываний.

Сигналы запросов на прерывания:

- RXINTR – от приемного FIFO;
- TXINTR – от передающего FIFO;
- RTINTR – по таймауту приемника;
- TDINTR – по окончанию передачи в линии;
- MSINTR – по состоянию модема;
- EINTR – по ошибке;
- UARTINTR – логическое ИЛИ сигналов запросов на прерывания.

Каждый из сигналов может быть маскирован путем установки соответствующего бита в регистре маски IMSC.

Источник прерывания также можно определить, считав состояние регистра RIS или регистра MIS (маскированные прерывания). Сброс прерывания осуществляется программно путём установки соответствующего бита в регистре ICR.

Сигнал RXINTR

Запрос возникает в случае обнаружения одного из событий:

- буфер приемника в режиме FIFO и его заполнение достигло заданного порогового значения;
- буфер приемника имеет одну ячейку (режим FIFO запрещен) и принят один кадр данных.

Линия прерывания переходит в высокое состояние и удерживается в нем до тех пор, пока из буфера не будет прочитан, как минимум, один байт или выполнен программный сброс прерывания (регистр ICR).

Сигнал TXINTR

Запрос возникает в случае обнаружения одного из событий:

- буфер передатчика в режиме FIFO и его опустошение достигло заданного порогового значения;
- буфер передатчика имеет одну ячейку (режим FIFO запрещен) и пуст.

Линия прерывания переходит в высокое состояние и удерживается в нем до тех пор, пока в буфер не будет записан, как минимум, один байт или выполнен программный сброс прерывания.

Запись данных в буфер передатчика допускается как перед разрешением работы приемопередатчика и прерываний, так и после разрешения.

Примечание – Прерывание передатчика работает по фронту, а не по уровню сигнала. В случае если работа приемопередатчика и прерывания от него разрешена до осуществления записи данных в буфер передатчика, прерывание не формируется. Прерывание возникает только при опустошении буфера.

Сигнал TDINTR

Запрос возникает в случае окончания передачи в линии.

Сигнал MSINTR

Прерывание возникает в случае изменения любой из линий состояний модема (UARTx_CTS, UARTx_DCD, UARTx_DSR, UARTx_RI). Прерывание сбрасывается программно.

Сигнал RTINTR

Запрос возникает в случае, если буфер приемника не пуст и на вход приемника не поступало новых данных в течение периода времени, необходимого для передачи 32 бит. Прерывание сбрасывается после считывания данных из буфера приемника до его опустошения или программно.

Сигнал EINTR

Запрос возникает в случае ошибки при приеме данных. Оно может быть вызвано:
- ошибкой в структуре кадра;
- ошибкой контроля четности;
- разрывом линии;
- переполнением буфера приемника.

Сигнал UARTINTR

Логическое ИЛИ сигналов RXINTR, TXINTR, TDINTR, MSINTR, RTINTR, EINTR.

В контроллер PLIC поступает по 1 линии прерывания UARTINTR от каждого блока UART.

25.5 Программирование

Для программирования рекомендуется следующая последовательность действий:
- запретить работу приемопередатчика;
- дождаться окончания приема и/или передачи текущего байта данных;
- сбросить буфер передатчика посредством сброса бита FEN регистра LCRH;
- изменить настройки регистра CR;
- разрешить работу приемопередатчика.

26 Контроллер интерфейса CAN 2.0b

Микроконтроллер содержит один контроллер интерфейса CAN с двумя узлами, 128 объектами сообщений и двумя линиями прерывания.

26.1 Протокол CAN

Последовательный интерфейс CAN (Controller Area Network) – интерфейс связи, эффективно поддерживающий распределенное управление в масштабе реального времени с высокой помехозащищенностью. Протокол связи определен в спецификации CAN 2.0B.

Протокол CAN оптимизирован для систем, в которых должно передаваться относительно небольшое количество информации (по сравнению с Ethernet или USB) к любому или всем узлам сети. Множественный доступ с опросом состояния шины позволяет каждому узлу получить доступ к шине с учетом приоритетов. Неадресная структура сообщений позволяет организовать многоабонентскую доставку данных с сокращением трафика шины. Быстрая устойчивая передача информации с системой контроля ошибок позволяет отключать неисправные узлы от шины, что гарантирует доставку критических по времени сообщений.

Область применения протокола CAN: от высокоскоростных сетей связи до применения в автомобиле. Высокая скорость передачи данных (до 1 Мбит/с), хорошая помехозащищенность протокола, защита от неисправности узлов – делают шину CAN подходящей для индустриальных приложений управления типа Device Net.

CAN имеет асинхронную последовательную структуру шины с одним логическим сегментом сети. CAN сеть может состоять из двух или более узлов с возможностью подключения/отключения узлов от шины без перенастройки других устройств, см. рисунок 26.1.

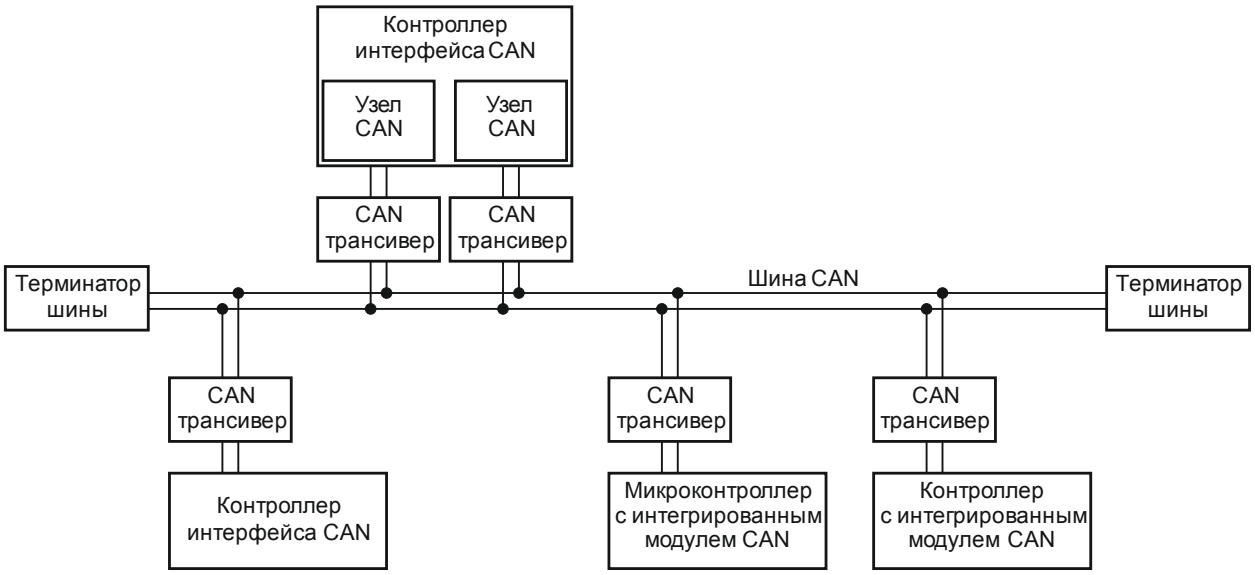


Рисунок 26.1 – Общая структура CAN сети

Логика шины работает по механизму «монтажного И», в котором рецессивный бит соответствует логической единице, а доминантный – логическому нулю. Пока ни один узел не формирует доминантный бит, шина находится в рецессивном состоянии. Появление на шине доминантного бита (выставленного одним или несколькими узлами) создает доминантное состояние шины. Отсюда следует, что при выборе среды передачи данных необходимо точно определить, какое состояние будет доминантным, а какое – рецессивным. Одним из наиболее распространенных и дешевых вариантов линии связи является витая пара. Линии шины тогда называются CANH и CANL и могут быть

подключены непосредственно к устройствам. Не существует никакого дополнительного стандарта на среду передачи данных.

При использовании в качестве линии связи витой пары с нагрузочными резисторами на концах можно получить максимальную скорость передачи данных 1 Мбит/с при длине линии до 40 м. Для линий связи протяженностью более 40 м необходимо снизить скорость передачи данных (для линии 1000 м скорость шины должна быть не более 40 Кбит/с). Из-за дифференциального характера линии связи, шина CAN малочувствительна к электромагнитным помехам. Экранирование шины значительно снизит воздействие внешнего электромагнитного поля, что особенно важно для высокоскоростных режимов работы.

Двоичная информация кодируется. Доминантным является низкий уровень, рецессивным – высокий уровень. Для гарантированной синхронизации данных всеми узлами шины используется принцип «бит-стаффинга». Это означает, что при последовательной передаче пяти бит одинаковой полярности передатчик вставляет один дополнительный бит противоположной полярности перед передачей остальных битов. Приемник также проверяет полярность и удаляет дополнительные биты.

В CAN протоколе при передаче данных приемные узлы не адресуются, а указывается идентификатор передатчика. С помощью идентификатора указывается содержание сообщения (например, применительно автомобиля – обороты, температура двигателя и т. д.) и степень приоритета сообщения. Более высокий приоритет у идентификатора, имеющего меньшее бинарное значение.

При коллективном доступе к шине используется неразрушающий арбитраж с опросом состояния шины. Перед началом передачи данных узел проверяет состояние шины (отсутствие активности на шине). При начале передачи сообщения узел становится управляющим шины, все остальные узлы переходят в режим приема. После приема сообщения (подтвержденного каждым узлом) каждый узел проверяет идентификатор в сообщении и сохраняет сообщение, если это требуется. В противном случае, сообщение сбрасывается. Если два или более узлов начинают передачу данных одновременно, поразрядный арбитраж позволяет избежать конфликта на шине. Каждый узел выдает на шину свой идентификатор (старший бит формируется первым) и контролирует ее состояние. Если узел посылает «1», а считывает «0», значит арбитраж потерян, и узел переключается в режим приема. Это происходит тогда, когда идентификатор конкурирующего узла имеет меньшее бинарное значение. Таким образом, узел с высоким приоритетом выигрывает арбитраж без необходимости повторять сообщение. Все остальные узлы будут пытаться передать сообщение после освобождения шины. Данный механизм не позволяет передавать сообщения одновременно разными узлами. Для этого программно должно быть обеспечено, чтобы узлы, передающие данные, не имели одинаковых идентификаторов. Оригинальная спецификация в версии CAN 2.0b (так называемая расширенная версия CAN) определяет возможность идентификатора иметь длину 11 или 29 бит.

Протокол CAN предусматривает следующие типы сообщений:

- сообщение данных (стандартное и расширенное);
- удаленный запрос данных;
- сообщение об ошибке;
- сообщение о перезагрузке.

Стандартное сообщение данных

Стандартное сообщение данных формируется, когда узел желает передать данные. Формат сообщения показан на рисунке 26.2.

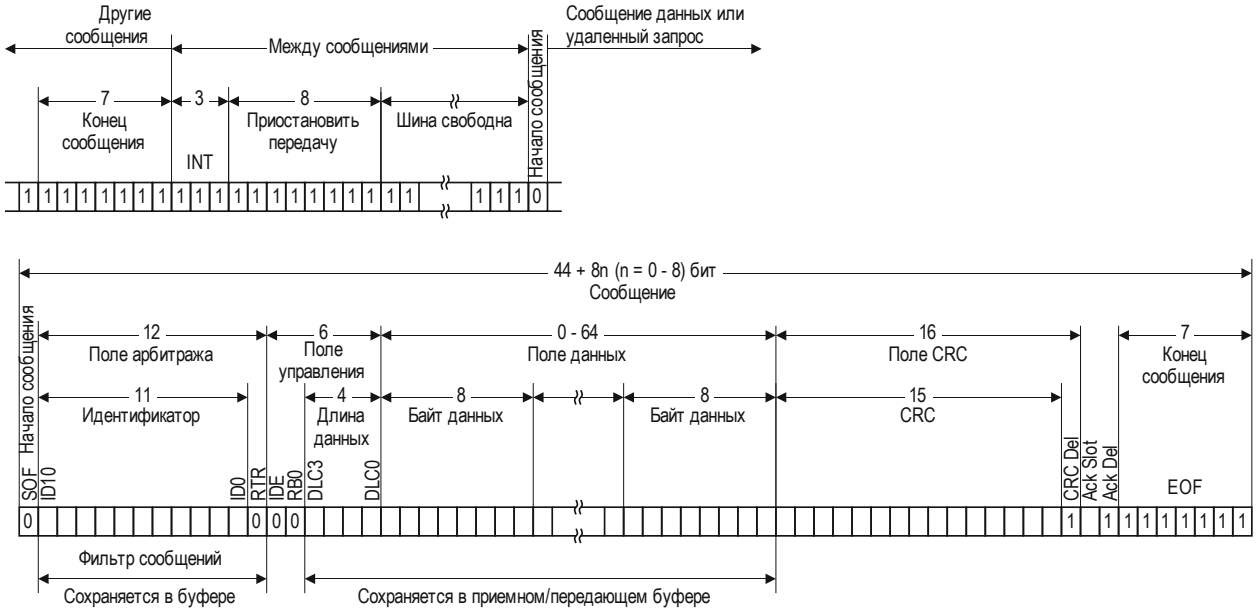


Рисунок 26.2 – Стандартное сообщение данных

Стандартное сообщение имеет в своем составе:

- бит SOF – доминантный («0») бит начала сообщения для жесткой синхронизации всех узлов;

- поле арбитража (12 бит), включающее поле ID идентификатора (11 бит), и бит RTR передачи по удаленному запросу (RTR = «0» соответствует сообщению данных, RTR = «1» соответствует удаленному запросу);

- поле управления (6 бит), включающее бит IDE, – указатель расширенного идентификатора (IDE = «0» соответствует стандартному идентификатору, IDE = «1» соответствует расширенному идентификатору), бит RB0 – резервный доминантный бит и поле DLC – числа байт данных (4 бита), которое указывает, сколько байт данных содержится в сообщении (допустимые значения от 0 до 8, другие значения использоваться не могут);

- поле данных (от 0 до 64 бит), содержащее целое число байт данных;

- поле контрольной суммы CRC (16 бит), включающее поле CRC (15 бит), используемое для обнаружения возможных ошибок передачи данных, и бит CRCDel рецессивный разделитель CRC;

- поле подтверждения (2 бита), включающее бит ACK Slot подтверждения передачи (передающий узел выдает рецессивный бит, а любой узел, который принял сообщение без ошибок, заменяет его сформированным доминантным битом), и бит ACKDel рецессивный разделитель подтверждения;

- поле EOF конца сообщения (7 бит).

Между передачами двух любых сообщений шина должна оставаться в рецессивном состоянии как минимум в течение времени появления 3 бит (поле INT простоя). Если после появления трех рецессивных бит (поле INT) ни один узел не начал передачу, шина переходит в состояние бездействия IDLE и находится в рецессивном состоянии до появления доминантного бита сообщения.

Расширенное сообщение данных

Расширенное сообщение данных формируется, когда узел желает передать данные. Формат сообщения показан на рисунке 26.3.

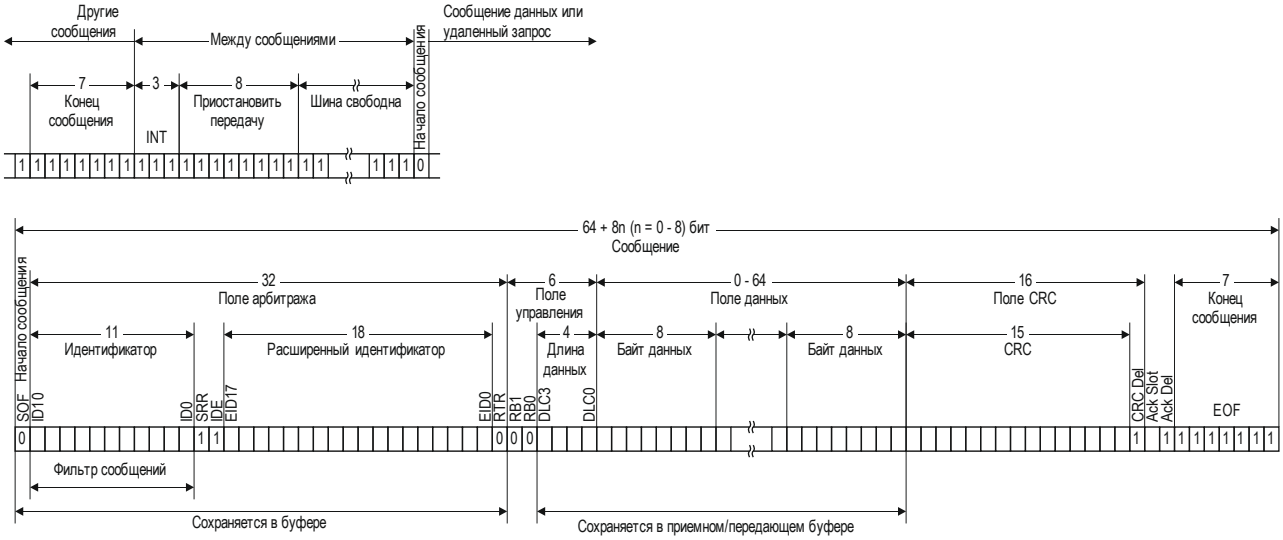


Рисунок 26.3 – Расширенное сообщение данных

Расширенное сообщение имеет в своем составе:

- бит SOF – доминантный («0») бит начала сообщения для жесткой синхронизации всех узлов;

- поле арбитража (38 бит), включающее поле стандартного идентификатора (11 бит), бит SRR заменитель удаленного запроса, бит IDE указатель расширенного идентификатора (рецессивный, что соответствует расширенному идентификатору) и поле расширенного идентификатора (18 бит);

- бит RTR передачи по удаленному запросу (RTR = «0» соответствует сообщению данных, RTR = «1» соответствует удаленному запросу);

- поле управления (6 бит), включающее бит RB0 – резервный доминантный бит, бит RB1 – резервный доминантный бит и поле DLC числа байт данных (4 бита), которое указывает, сколько байт данных содержится в сообщении (допустимые значения – от 0 до 8, другие значения использоваться не могут);

- поле данных (от 0 до 64 бит), содержащее целое число байт данных;

- поле контрольной суммы CRC (16 бит), включающее поле CRC (15 бит), используемое для обнаружения возможных ошибок передачи данных и бит CRCDel рецессивный разделитель CRC;

- поле подтверждения (2 бита), включающее бит ACK Slot подтверждения передачи (передающий узел выдает рецессивный бит, а любой узел, который принял сообщение без ошибок, заменяет его сформированным доминантным битом) и бит ACKDel рецессивный разделитель подтверждения;

- поле EOF конца сообщения (7 бит).

Удаленный запрос данных

Формируется, когда узлу требуются данные другого узла. Узел назначения посылает удаленный запрос с идентификатором источника. Соответствующий узел источника (распознавший свой идентификатор) посылает стандартное или расширенное сообщение в ответ на запрос.

Удаленный запрос данных существует в стандартном и расширенном вариантах. На рисунке 26.4 представлен вариант удаленного запроса со стандартным идентификатором.

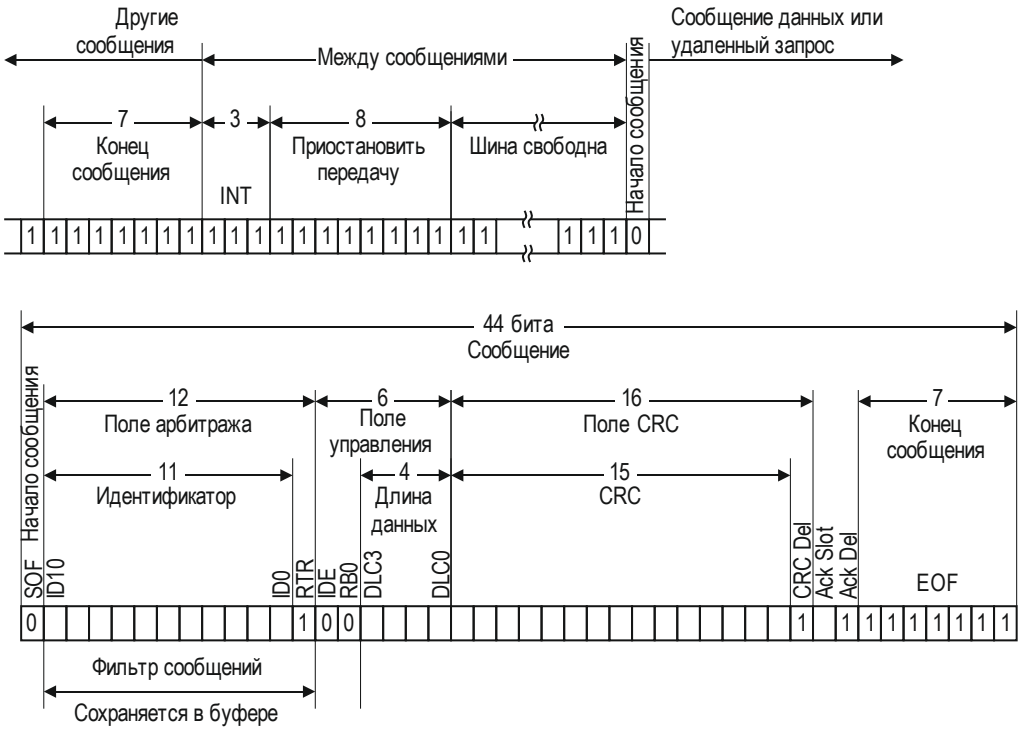


Рисунок 26.4 – Удаленный запрос данных (стандартный формат)

Имеются только два отличия содержимого удаленного запроса от сообщения данных:

- бит RTR в удаленном запросе передается в рецессивном состоянии;
- поле данных отсутствует (в сообщении не передается никаких данных, значение в поле DLC любое в пределах от 0 до 8).

В самом маловероятном случае, когда одновременно формируется удаленный запрос, и устройство пытается передать данные с одинаковыми идентификаторами, арбитраж будет выигран устройством, передающим данные, из-за доминантного состояния бита RTR.

Узел, который посылал запрос, получает данные немедленно.

Сообщение об ошибке

Сообщение об ошибке формируется любым узлом, который обнаруживает ошибку на шине. Формат сообщения показан на рисунке 26.5.

Сообщение об ошибке состоит из двух полей: поле разделителя ошибки и поле флага ошибки. Возможны два типа поля флага ошибки, в зависимости от вида ошибки узла, обнаружившего ее.

Если ошибку обнаружил активный узел (как в примере на рисунке 26.5), тогда он прерывает передачу текущего сообщения, формируя флаг активной ошибки. Флаг активной ошибки состоит из шести последовательных доминантных бит, которые нарушают правила «бит-стаффинга» (правила наполнения и передачи бит на шине). Остальные узлы также обнаруживают ошибку и начинают формировать сообщение об ошибке. Таким образом, поле флага ошибки может содержать от 6 до 12 доминантных бит (сформированных одним узлом или более). Поле флага ошибки дополняется разделителем ошибки, состоящим из восьми рецессивных бит и позволяющим перезапускать связь с шиной после обнаружения ошибки. После перехода шины в нормальное состояние узлы возобновляют передачу данных, остановленный узел повторяет передачу сообщения, переданного до этого с ошибкой.

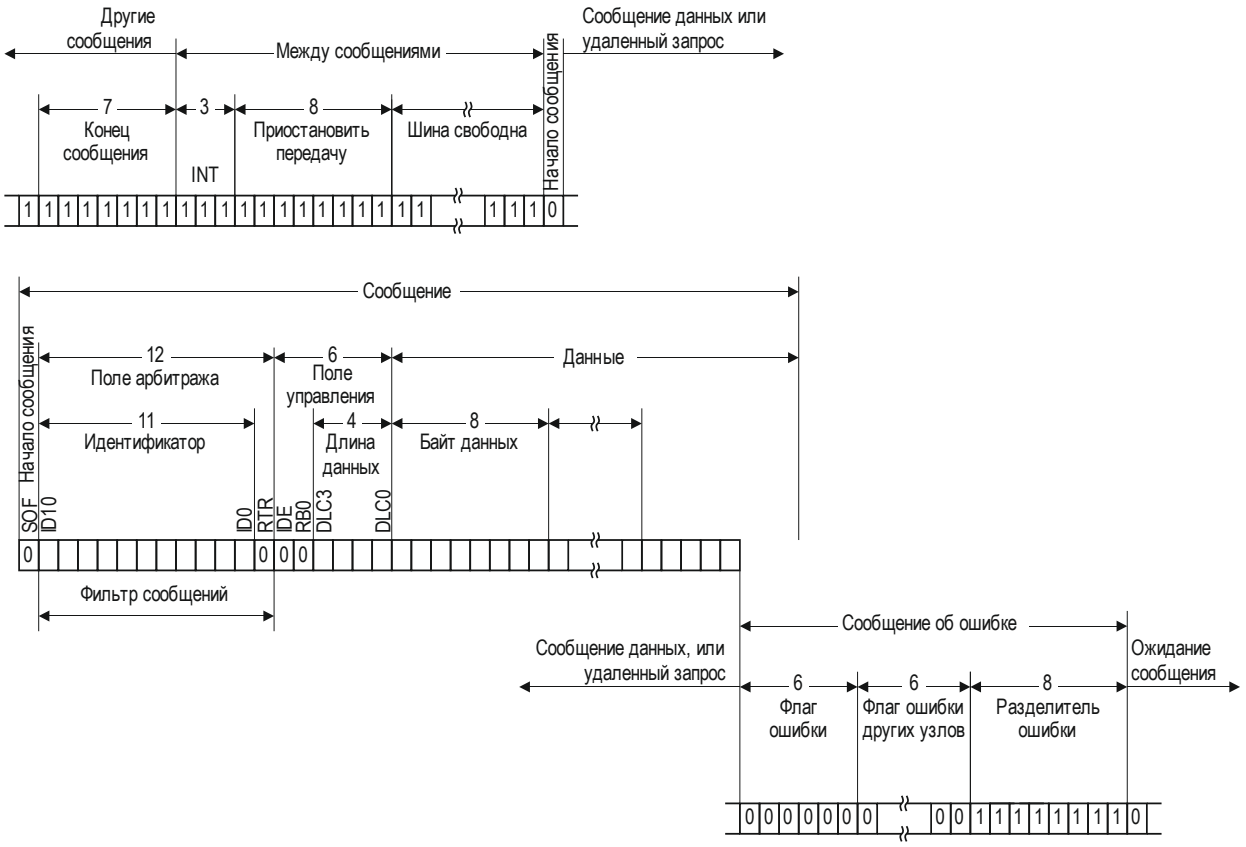


Рисунок 26.5 – Сообщение об ошибке

Если ошибку обнаружил пассивный узел, тогда он формирует флаг пассивной ошибки, состоящий из шести последовательных рецессивных бит, и затем разделитель ошибки. Таким образом, сообщение о пассивной ошибке состоит из 14 рецессивных бит. Это не нарушает правила «бит-стаффинга» на шине и не оказывает влияния на передачи других узлов. Исключение составляет узел, который передает данные узлу, обнаружившему ошибку. В этом случае правила «бит-стафинга» нарушаются, и передача данных прекращается. После передачи пассивной ошибки узел должен ожидать шесть последовательных рецессивных бит для восстановления связи с шиной.

Сообщение о перезагрузке

Формат сообщения о перезагрузке аналогичен формату сообщения об ошибке, но может быть сформирован, только когда шина простаивает.

Сообщение о перезагрузке показано на рисунке 26.6.

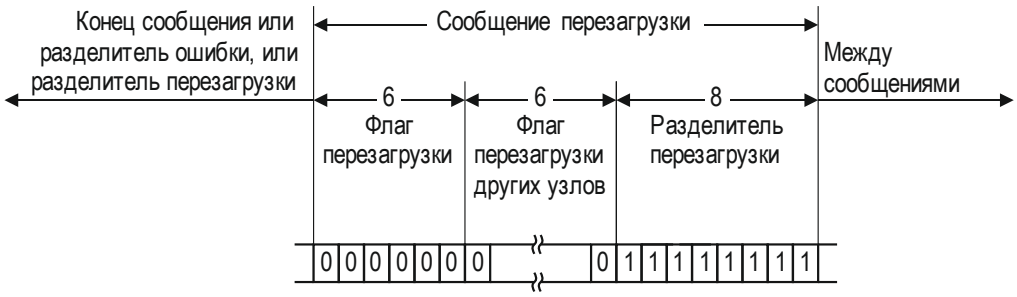


Рисунок 26.6 – Сообщение о перезагрузке

Разделитель перезагрузки состоит из восьми последовательных рецессивных бит.

Узел может сформировать сообщение о перезагрузке в двух случаях:

- между сообщениями обнаружен доминантный бит, что является ненормальным во время простоя шины;
- для задержки передачи нового сообщения.

Узел может последовательно сформировать не более двух сообщений перезагрузки.

Флаг перезагрузки состоит из шести последовательных доминантных бит. Другие узлы обнаруживают перезагрузку и начинают формировать ее самостоятельно. Поэтому на шине во время выполнения перезагрузки может быть до 12 доминантных бит.

26.2 Структура и функционирование контроллера CAN

В состав контроллера CAN входят два идентичных независимых узла CAN0 и CAN1, ОЗУ для хранения сообщений, которое является общим для узлов, и система управления. Контроллер CAN имеет следующие функциональные особенности:

- соответствие ISO 11898;
- функционирование согласно спецификации CAN 2.0b (активная версия);
- отдельные управляющие регистры для каждого из двух узлов;
- программируемая скорость передачи информации до 1 Мбит/с;
- гибкий и полный контроль передачи сообщений и обработки ошибок.

Контроллер CAN реализует 2 линии прерываний и 128 объектов сообщений для хранения сообщений и их параметров в ОЗУ. Каждый объект сообщения может быть привязан к любому из узлов, сконфигурирован для передачи или приема как стандартных, так и расширенных сообщений и удаленных запросов. Каждый объект имеет индивидуальную маску для фильтрации принимаемых сообщений. Объекты сообщений могут объединяться в классы, с разными уровнями приоритета, могут объединяться для построения структур FIFO произвольных размеров (до 128 объектов в одной структуре). Кроме того, реализована возможность попарного соединения объектов для формирования шлюзов для автоматической передачи сообщений между узлами. Параллельно с вышеуказанными свойствами объекты сообщений могут организовываться в списки с постоянно доступной реорганизацией (совместимость с TwinCan-устройствами, которые не имеют списков).

Структура контроллера CAN приведена на рисунке 26.7.

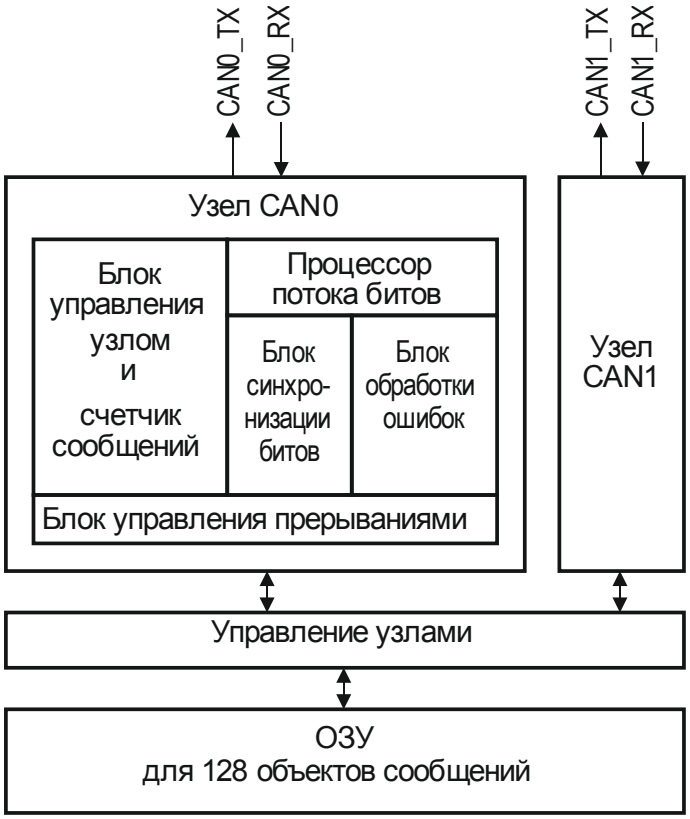


Рисунок 26.7 – Общая структура контроллера CAN

Синхронизация

Тактирующим сигналом контроллера CAN является сигнал FIN (HLCK), приходящий с генератора тактовых сигналов. На основе этого сигнала посредством программируемого дробного делителя частоты формируется внутренний синхросигнал FCAN (FOUT), синхронизирующий работу контроллера и являющийся базовым синхросигналом для передачи/приема сообщений по внешней шине CAN.

Включение контроллера CAN

По умолчанию после сброса микроконтроллера контроллер CAN выключен. На это также указывает состояние флага DISS регистра CLC. Когда контроллер выключен, этот флаг установлен.

Для включения контроллера CAN следует записать ноль в бит DISR регистра CLC. После этого флаг DISS сбросится. Рекомендуется проверять состояние флага DISS, перед началом программирования регистров контроллера, которые не доступны в выключенном состоянии.

Выключение контроллера CAN

Программно можно перевести контроллер CAN в режим выключения установкой бита DISR. Контроллер завершает все текущие операции, после чего устанавливает флаг DISS и отключает внутреннее тактирование, в связи с чем, все регистры становятся недоступными для обращения.

Простой шины

Между передачами сообщений шина CAN находится в рецессивном состоянии. Для выполнения условий простоя шины необходимо, чтобы было получено, как минимум, три рецессивных бита после завершения передачи/приема очередного сообщения.

Анализ работы контроллера CAN

Для анализа работы контроллера CAN доступны два режима – общего анализа и внутренней петли «loop-back».

Режим общего анализа включается установкой бита CALM регистра NCR узла и позволяет осуществлять независимый мониторинг работы узла, не затрагивая шину CAN. В этом режиме сообщения данных и удаленные запросы отслеживаются без участия узла в операциях на шине. Выходы узла находятся в рецессивном состоянии. Узел может получать сообщения данных, сообщения удаленных запросов и сообщения об ошибках, но работа узла на передачу запрещена. Полученные сообщения данных/удаленных запросов остаются без подтверждения (бит подтверждения остается в рецессивном состоянии), но принимаются и сохраняются (при совпадении идентификаторов) в соответствующих объектах сообщений. В ответ на входящие сообщения не выдается подтверждение, и не генерируются сообщения об ошибках. На удаленные запросы не выдаются сообщения данных, а сами сообщения данных не могут быть переданы установкой бита запроса передачи TXRQ регистра состояния объекта сообщения MOSTAT. Прерывания после приема генерируются (если это разрешено) для всех принятых сообщений, не содержащих ошибок.

Режим внутренней петли включается установкой бита LBM регистра NPCR и позволяет проводить внутреннее тестирование контроллера CAN, а также отладку управляющей программы без доступа к внешней шине CAN. Внутренняя петля состоит из внутренней шины CAN (внутри контроллера CAN) и переключателя выбора шины для каждого узла, см. рисунок 26.8. С помощью переключателя каждый узел CAN может быть подключен либо к внутренней шине (режим внутренней петли), либо к внешней шине (нормальный режим работы). Если выбран режим внутренней петли, то на внешнем передающем выводе узла CAN поддерживается рецессивный уровень сигнала, а состояние принимающего вывода игнорируется.

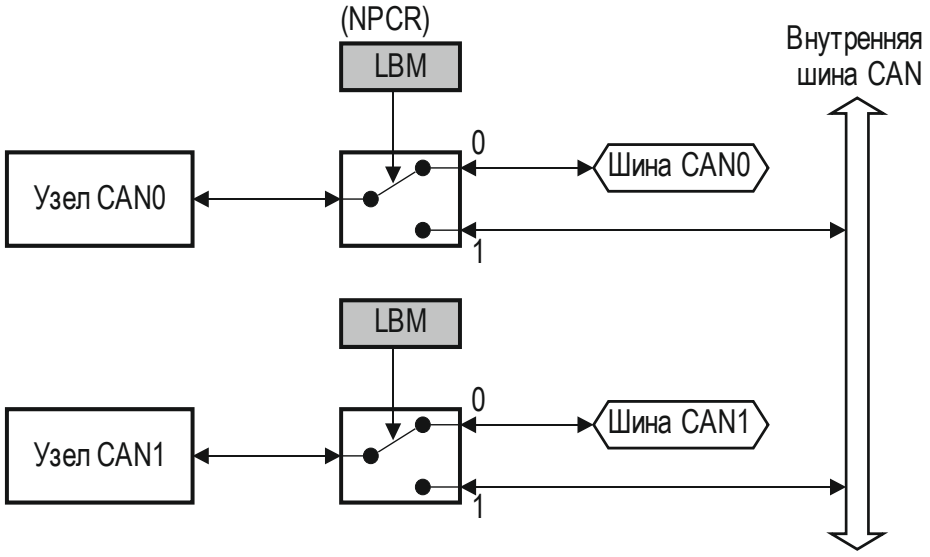


Рисунок 26.8 – Режим внутренней петли «loop-back»

Если оба узла CAN функционируют в режиме внутренней петли, они взаимодействуют друг с другом посредством внутренней шины CAN, не оказывая влияние на работу других модулей, функционирующих в нормальном режиме.

Дробный делитель

Дробный делитель позволяет генерировать тактовый сигнал FOUT из входного FIN (HCLK) путем программирования делителя посредством регистра FDR.

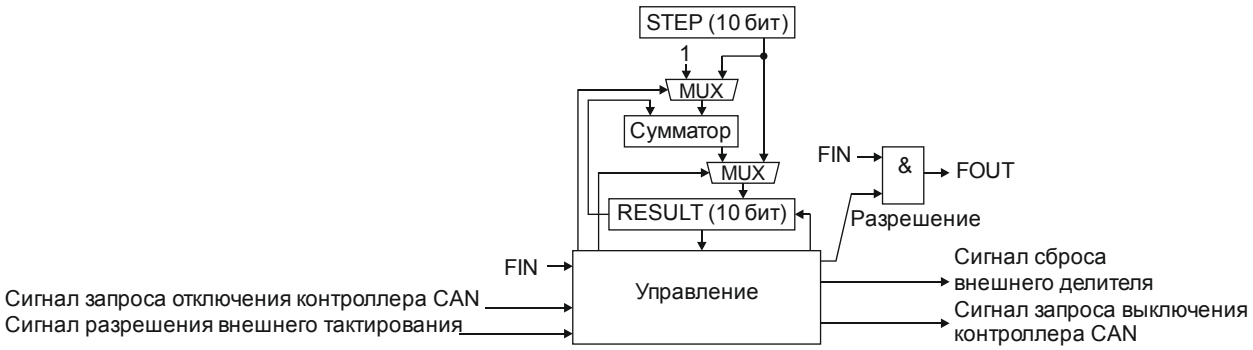


Рисунок 26.9 – Схема дробного делителя

Задаваемое значение частоты сигнала FIN зависит от длительности передачи одного бита информации и должно быть n-кратно ей. Поскольку длительность передачи бита определяется количеством квантов времени Ntq, (см. далее), то для расчета частоты сигнала FIN в МГц следует пользоваться формулой:

$$f_{FIN} = n \times Ntq, \qquad (26.1)$$

где Ntq – количество квантов времени tq;

n – целое число, начиная с 1 (для задания кратности).

Дробный делитель делит f_{FIN} путем умножения на величину 1/val или величину 1024/val для любого val от 0 до 1023, получая на выходе тактовый сигнал FOUT (FCAN).

На рисунке 26.9 показана блок-схема дробного делителя. Логика дробного делителя работает по-разному, в зависимости от режима, задаваемого полем DM.

В режиме нормального деления (DM = 01b) делитель работает как перегружаемый счетчик с шагом инкрементирования, равным единице. Состояние счетчика доступно посредством поля RESULT. Каждый раз, при переполнении (т. е. когда RESULT = 3FFh), формируется импульс сигнала FOUT, после чего в счетчик загружается значение из поля STEP.

Частота сигнала FOUT определяется по формуле

$$f_{FOUT} = f_{FIN} \times 1/(1024 - STEPd), \qquad (26.2)$$

где STEPd – значение поля STEP в десятичном формате.

Отсюда следует, что для получения сигнала FOUT с частотой, равной частоте сигнала FIN, значение STEP должно быть равно 3FFh. На рисунке 26.10 показано формирование сигнала FOUT при значении STEP = 3FDh (1021d).

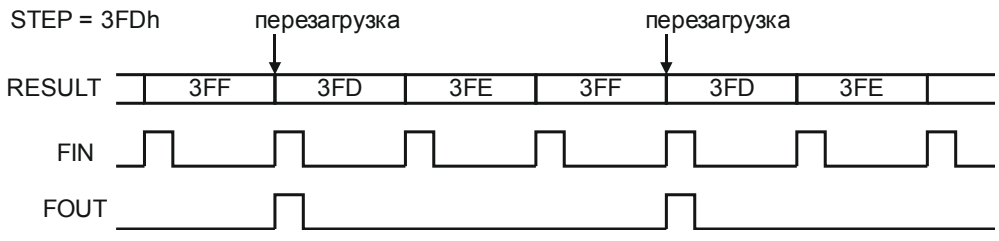


Рисунок 26.10 – Формирование сигнала FOUT в нормальном режиме

В режиме дробного деления (DM = 10b) делитель работает как перезагружаемый счетчик, но шаг инкрементирования в этом случае равен значению поля STEP. Если результат инкрементирования значения RESULT на величину STEP превышает 3FFh,

возникает переполнение счетчика, формируется импульс сигнала FOUT, после чего в счетчик загружается значение, на которое результат инкрементирования превысил 3FFh.

Частота выходного сигнала FOUT определяется по формуле

$$f_{FOUT} = f_{FIN} \times STEPd/1024d \; . \qquad (26.3)$$

В целом, режим дробного деления позволяет программировать частоту сигнала FOUT с более высокой точностью, чем нормальный режим, но сигнал может иметь джиттер периода, не превышающий одного периода FIN, в связи с чем, не рекомендуется использовать режим дробного деления при высоких скоростях передач.

На рисунке 26.11 показано формирование сигнала FOUT при значении STEP = 234h (564d). $f_{FOUT} = f_{FIN} \times 564/1024 = 0{,}55 \times f_{FIN}$.

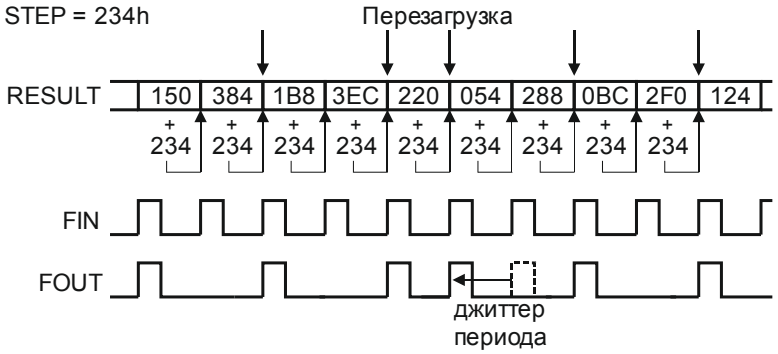


Рисунок 26.11 – Формирование сигнала FOUT в режиме дробного деления

Процесс выключения делителя начинается одновременно с возникновением запроса выключения контроллера CAN.

Контроллер сообщений

Контроллер сообщений управляет обменом сообщениями между CAN узлами и памятью сообщений и выполняет следующие функции:

- фильтрация входящих сообщений для определения корректного объекта сообщения для сохранения полученных данных;
- определение объекта сообщения, содержимое которого будет передано в первую очередь (для каждого узла индивидуально);
- передача содержимого объекта сообщения к CAN узлу с параллельной вставкой в сообщение битов управления и состояния;
- осуществление буферизации FIFO и функционирования шлюза;
- объединение битов уведомления ждущих обработки сообщений.

Управление прерываниями блока CAN

На рисунке 26.12 показана структура формирования запроса на прерывание.

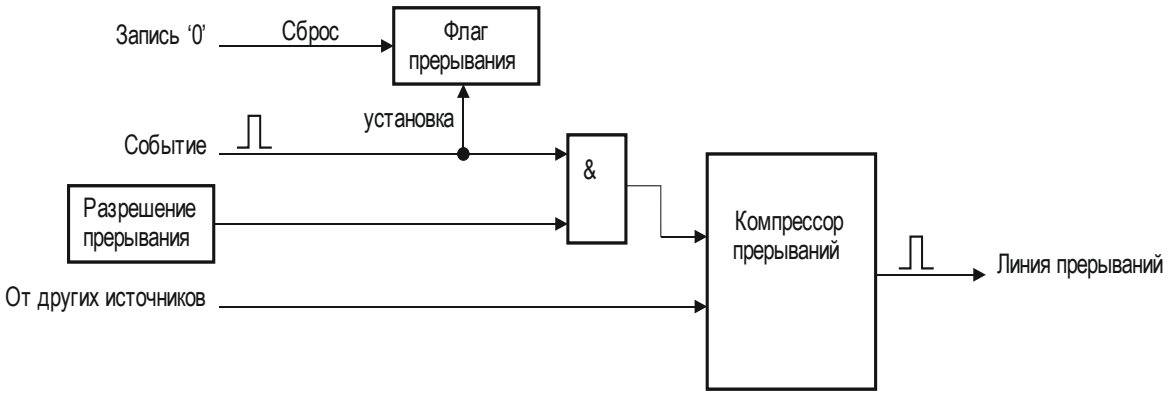


Рисунок 26.12 – Структура формирования запроса на прерывание

Событие, по которому должен быть сгенерирован запрос на прерывание, устанавливает флаг прерывания и (если разрешено) формирует запрос на прерывание на одной из 16 линий прерываний. Импульс запроса на прерывание генерируется независимо от состояния флага прерывания. Флаг прерывания может быть сброшен программно, записью нуля. Если к одной линии прерываний подключены несколько источников прерываний, то появление импульса от любого источника сформирует запрос на прерывание. Логика управления прерываниями использует схему компрессии прерываний.

Источниками прерываний являются:

- CAN узлы (восемь источников – по четыре для каждого узла);
- объекты сообщений (512 источников – по два для каждого объекта);
- программное прерывание (источник – регистр MITR).

Каждый аппаратный источник прерывания управляется четырьмя битами указателя прерываний, который определяет для него одну из 16 линий прерываний, что позволяет коммутировать на одну линию несколько источников прерываний. На рисунке 26.13 представлена схема коммутации линий прерываний.

Когда объект сообщения Msg_x генерирует запрос на прерывание по окончании приема или передачи сообщения, запрос передается на линию прерываний, выбранную в битовом поле RXINP или TXINP регистра MOIPR объекта сообщения Msg_x. Если количество объектов сообщений больше, чем количество линий прерываний, то на одну линию могут приходить несколько запросов прерываний. Для разрешения конфликтов на линиях прерываний в контроллере CAN предусмотрен механизм распределения приоритетов для объектов сообщений.

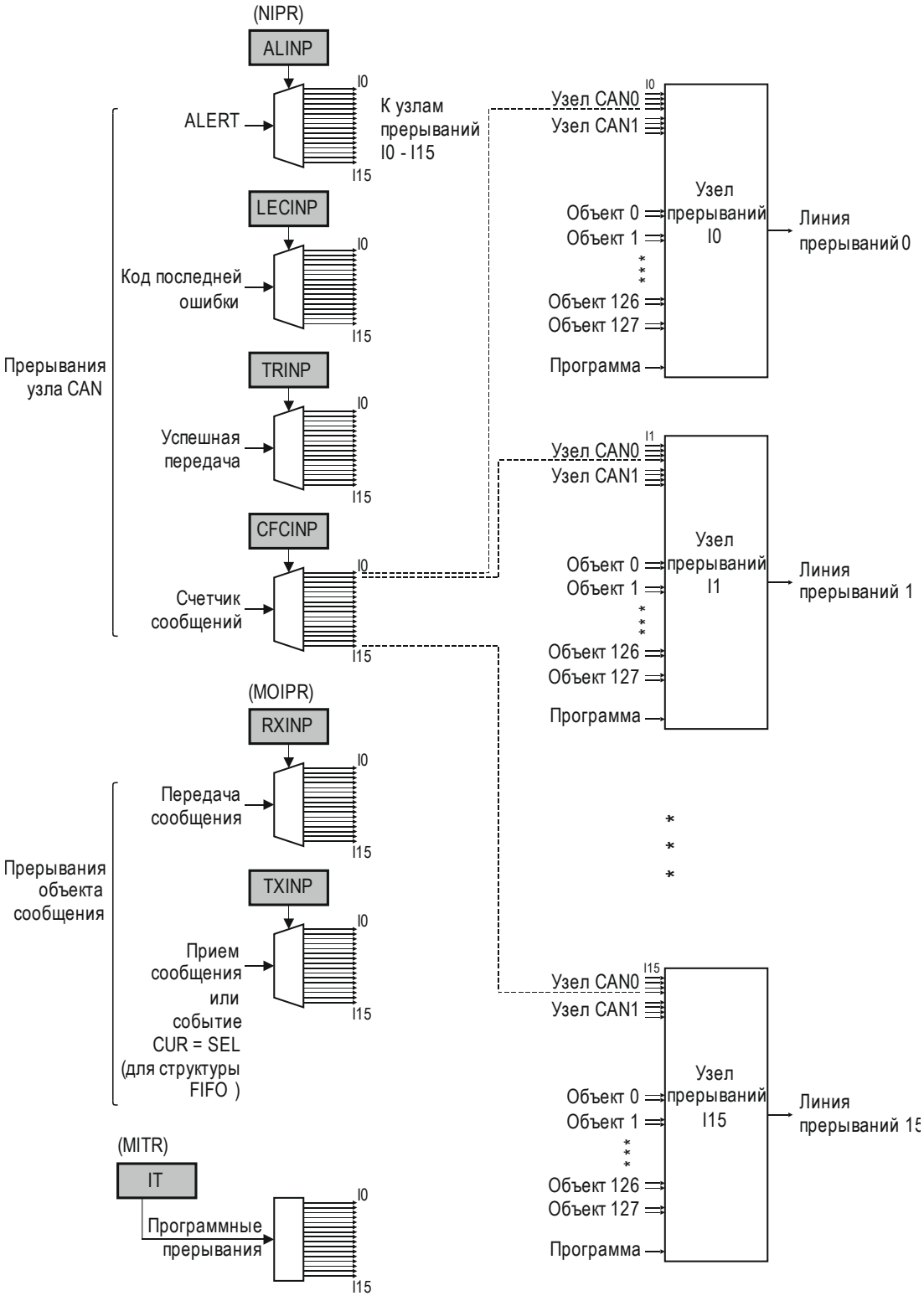


Рисунок 26.13 – Схема коммутации линий прерываний

26.3 Узел CAN

Каждый узел CAN имеет свою собственную логику управления и выдачи информации о состоянии и может быть сконфигурирован и работать независимо от другого узла.

Режим конфигурации включается установкой бита CCE регистра NCR. Режим конфигурации позволяет изменять параметры синхронизации битов и состояния счетчиков ошибок.

Конфигурация прерываний задается битами TRIE, ALIE и LECIE:

- бит TRIE управляет разрешением прерывания после передачи сообщения;
- бит ALIE управляет разрешением прерываний по ошибке;
- бит LECIE управляет разрешением прерывания по коду последней ошибки.

Регистр NSR отражает текущее состояние, содержит информацию о передачах и ошибках узла CAN.

Блок управления узлом CAN

Блок управления узлом координирует работу:

- разрешает/запрещает действия узла на шине;
- разрешает/запрещает и генерирует различные события, касающиеся работы узла (ошибка на шине, успешное завершение передачи сообщения), которые приводят к формированию запросов на прерывания;
- управляет счетчиком сообщений.

Блок синхронизации битов CAN

Согласно стандарту ISO 11898 время передачи одного бита разделено на сегменты, которые, в свою очередь, составлены из целочисленных отрезков времени, называемых квантами времени tq, см. рисунок 26.14. Квант времени – фиксированная единица времени, получаемая из частоты синхронизации и делителя контроллера CAN.

Сегмент синхронизации Tsync позволяет синхронизировать начало обмена данными между передатчиком и приемником. Длительность сегмента всегда равна одному кванту времени.

Сегмент распространения Tprop используется для компенсации физического времени запаздывания сигнала в пределах сети. Длительность сегмента рассчитывается с учетом времени прохождения сигнала от передатчика к приемнику и обратно, входной задержки компаратора и задержки выхода драйвера и может составлять от 1 до 8 квантов времени.

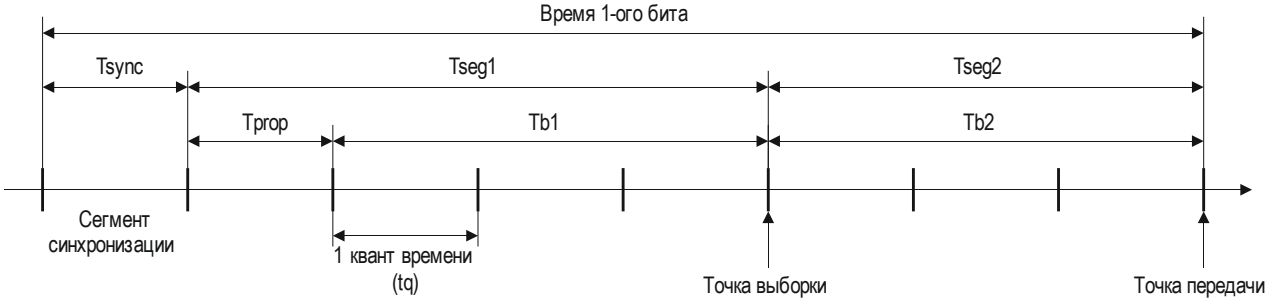


Рисунок 26.14 – Структура одного бита

Сегменты буфера фазы 1 и буфера фазы 2 – Tb1 и Tb2, расположенные до и после точки выборки, используются для компенсации смещения фазы тактовых частот источника и приемника, обнаруживаемой после появления сегмента синхронизации, а также для оптимального расположения точки выборки полученного бита.

Точка выборки – момент, когда читается состояние шины CAN для определения принятого бита. Как правило, длительность временного интервала от начала бита до точки выборки составляет (60 – 70) % времени бита, в зависимости от системных параметров.

Сегмент распространения и сегмент буфера фазы 1 вместе составляют сегмент параметра 1 (Tseg1), который определяется битовым полем TSEG1 регистра синхронизации битов NBTR (может быть записан, только если установлен бит CCE регистра NCR). Согласно стандарту ISO, минимальная длительность сегмента параметра 1 должна составлять три кванта времени.

Сегмент параметра 2 (Tseg2) определяется битовым полем TSEG2 регистра NBTR и охватывает сегмент буфера фазы 2. Минимальная длительность сегмента параметра 2 составляет два кванта времени.

Согласно стандарту ISO, минимальная длительность одного бита, получающаяся сложением сегментов Tsync, Tseg1 и Tseg2, не должна быть менее 8 квантов времени.

Максимальная длительность бита – 25 квантов времени.

Примечание – Минимальное номинальное время передачи одного бита составляет 1 мкс, что соответствует скорости передачи 1 Мбит/с.

Формулы вычисления значений сегментов и времени одного бита Tbit:

- при DIV8 = 0 значение кванта времени

$$tq = (BRP + 1)/f_{FOUT}; \quad (26.4)$$

- при DIV8 = 1 значение кванта времени

$$tq = 8 \times (BRP + 1)/f_{FOUT}; \quad (26.5)$$

- Tsync = 1 × tq;
- Tseg1 = (TSEG1 + 1) × tq ≥ 3tq;
- Tseg2 = (TSEG2 + 1) × tq ≥ 2tq;
- Tbit = Tsync + Tseg1 + Tseg2 ≥ 8tq.

Чтобы компенсировать смещение фазы между частотами генераторов различных узлов шины CAN, каждое устройство должно синхронизироваться по фронту смены уровня сигнала на шине CAN от рецессивного к доминантному. Как только фронт обнаруживается, логика синхронизации сравнивает его текущее положение с ожидаемым и выполняет настройку значений параметров Tseg1 и Tseg2.

Контроллер CAN использует два механизма синхронизации – аппаратный и ресинхронизацию (синхронизация с восстановлением тактовых интервалов).

Аппаратная синхронизация выполняется по каждому фронту смены уровня сигнала на шине CAN от рецессивного к доминантному. При аппаратной синхронизации временные интервалы сегментов, из которых складываются времена битов, не изменяются в течение всего сообщения.

Ресинхронизация выполняется автоматическим удлинением сегмента Tseg1 или укорачиванием сегмента Tseg2. Максимальное значение изменения сегментов колеблется в пределах от 1 до 4 квантов времени. Синхронизация выполняется только при появлении фронта смены уровня сигнала на шине CAN от рецессивного к доминантному. Фиксированное значение максимального числа последовательных бит одинаковой полярности гарантирует своевременное восстановление синхронизации. Смещение фазы фронта смены уровня сигнала на шине отслеживается относительно сегмента синхронизации и измеряется в квантах времени.

Если величина фазового смещения меньше или равна запрограммированному значению ширины перехода ресинхронизации T_{SJW}, выполняется аппаратная синхронизация.

Если величина смещения фазы больше, чем T_{SJW}, а фазовое смещение положительно, то удлиняется сегмент Tseg1, в случае отрицательного фазового смещения укорачивается сегмент Tseg2.

Значение T_{SJW} определяется полем SJW регистра NBTRx по формуле

$$T_{SJW} = (SJW + 1) \times tq. \quad (26.6)$$

Помимо прочего, должны соблюдаться следующие правила:

$$Tseg1 \geq T_{SJW} + Tprop \quad (26.7)$$

$$\text{и } Tseg2 \geq T_{SJW}. \quad (26.8)$$

Соотношения между максимальным отклонением частоты сигнала FOUT и сегментами буферов фаз, и шириной перехода ресинхронизации следующие:

$$-\Delta f_{FOUT} \leq T/2 \times (13 \times Tbit - Tb2); \quad (26.9)$$

$$-\Delta f_{FOUT} \leq T_{SJW}/20 \times Tbit, \quad (26.10)$$

где T – меньшее из Tb1 и Tb2.

В итоге:

- Tsync составляет 1 квант времени;
- Tprop – от 1 до 8 квантов времени;
- Tb1 – от 1 до 8 квантов времени;
- Tb2 – выбирается равным двум квантам времени или равным сегменту Tb1, если его значение более двух квантов времени;
- T_{SJW} может составлять максимально 4 кванта времени, однако, в типовых приложениях достаточно 1.

Корректные значения параметров синхронизации битов должны быть записаны в регистр NBTR (доступен, если установлен бит CCE) до окончания инициализации (до сброса бита INIT регистра NCR), т. е. до начала работы CAN узла.

Процессор потока битов

Процессор потока битов формирует (на основе содержимого объектов сообщений) сообщения данных и удаленные запросы непосредственно перед отправкой на шину CAN. Процессор потока управляет генератором CRC (генератор контрольной суммы) и добавляет контрольную сумму к сообщению. После вставки битов начала SOF и конца EOF сообщения, процессор потока начинает передачу сообщения по правилам арбитража шины CAN. В течение всего времени передачи сообщения процессор потока битов ведет мониторинг шины. Если обнаруживается несовпадение текущего (определяемого мониторингом) и ожидаемого (выдаваемого CAN узлом) уровня напряжения на шине CAN, генерируется ошибка и соответствующий ей запрос на прерывание. Код возникшей ошибки отражается в битовом поле LEC регистра NSR.

Корректность получаемых данных проверяется и подтверждается или не подтверждается кодом CRC. В случае отсутствия подтверждения возникает ошибка, генерируется запрос на прерывание и код ошибки выставляется в регистре NSR, кроме этого, на шину выдается сообщение об ошибке.

После получения сообщения, не содержащего ошибок, и разбиения его на идентификатор и пакет данных, полученная информация записывается в буфер блока обработки сообщений, формируется соответствующее прерывание, и обновляются регистры состояния.

Блок обработки ошибок

Блок обработки ошибок предназначен для выявления ошибок в работе устройств узла. В составе блока есть два счетчика: счетчик ошибок приема (поле REC в регистре NECNT) и счетчик ошибок передачи (поле TEC). Инкрементированием и декрементированием счетчиков управляет процессор потока битов.

Если процессор потока битов сам выявляет ошибку в процессе передачи, то счетчик TEC инкрементируется на 8. Инкрементирование на 1 происходит, если об ошибке сообщено внешним CAN-устройством путем генерирования сообщения об ошибке. Направление передачи с ошибочным сообщением и узел, сообщивший об ошибке передачи, указывают на соответствующие узлы CAN в регистрах NECNT, что используется для анализа ошибки.

В зависимости от значений счетчиков ошибок узел CAN может находиться в одном из трех состояний:

- активной ошибки;
- пассивной ошибки;
- отключен от шины.

Узел находится в состоянии активной ошибки, если значение каждого из счетчиков ошибок меньше 128. Узел в состоянии активной ошибки присоединен к шине CAN и посылает флаг активной ошибки при обнаружении ошибок.

Узел находится в состоянии пассивной ошибки, если значение хотя бы одного из счетчиков ошибок больше или равно 128. Узел подключен к шине, но при обнаружении ошибок посылает флаг пассивной ошибки. После передачи узел в состоянии пассивной ошибки будет ждать инициализации дальнейшей передачи.

Узел находится в состоянии отключения от шины, если значение счетчика ошибок TEC больше или равно 256. О том, что CAN узел находится в состоянии отключения от шины CAN, сигнализирует флаг BOFF регистра NSR. Узел в состоянии отключения от шины CAN не может работать с шиной CAN (выходные передатчики отключены).

Флаг EWRN регистра NSR устанавливается, когда хотя бы один из счетчиков достиг или превысил лимит ошибок, определенный в битовом поле EWRNLVL регистра NECNT. Как только значения обоих счетчиков перестанут превышать лимит ошибок, флаг EWRN сбросится.

Счетчик сообщений

Счетчик сообщений может использоваться для получения информации о завершении передачи/приема сообщения соответствующего узла CAN. Подсчет сообщений осуществляется 16-разрядным счетчиком, который управляется регистром NFCR. Битовые поля CFMOD и CFSEL определяют режим работы и событие для инкрементирования счетчика.

Каждый узел CAN имеет в своем составе 16-разрядный счетчик сообщений/синхросчетчик, который подсчитывает количество принятых и переданных сообщений. Битовое поле CFSEL определяет один из трех режимов работы счетчика.

В режиме подсчета сообщений после успешной передачи и/или приема сообщения, содержимое счетчика копируется в битовое поле CFCVAL регистра MOIPR объекта сообщения Msg_x, участвующего в пересылке данных. После чего счетчик сообщений инкрементируется.

Прерывания узла CAN

Коммутация линий запросов прерываний показана на рисунке 26.15.

Узел может генерировать запросы на прерывания в случае:

- успешной передачи/приема сообщения;
- обнаружения кода последней ошибки;
- переполнения счетчика сообщений;
- состояния ALERT (состояние, возникающее, когда хотя бы один из счетчиков ошибок узла достиг значения своего лимита, изменяется состояние «отключен от шины», возникает ошибка длины списка или ошибка списка объектов).

После каждой успешной передачи или успешного приема сообщения генерируется (если разрешено соответствующими битами TXOK и RXOK) прерывание. Битовое поле TRINP регистра NIPR задает одну (из шестнадцати) линию прерывания.

Прерывание узла при возникновении кода последней ошибки формируется (если разрешено битом LECIE), если после модификации поля LEC его значение больше нуля. Битовое поле LECINP задает линию прерывания.

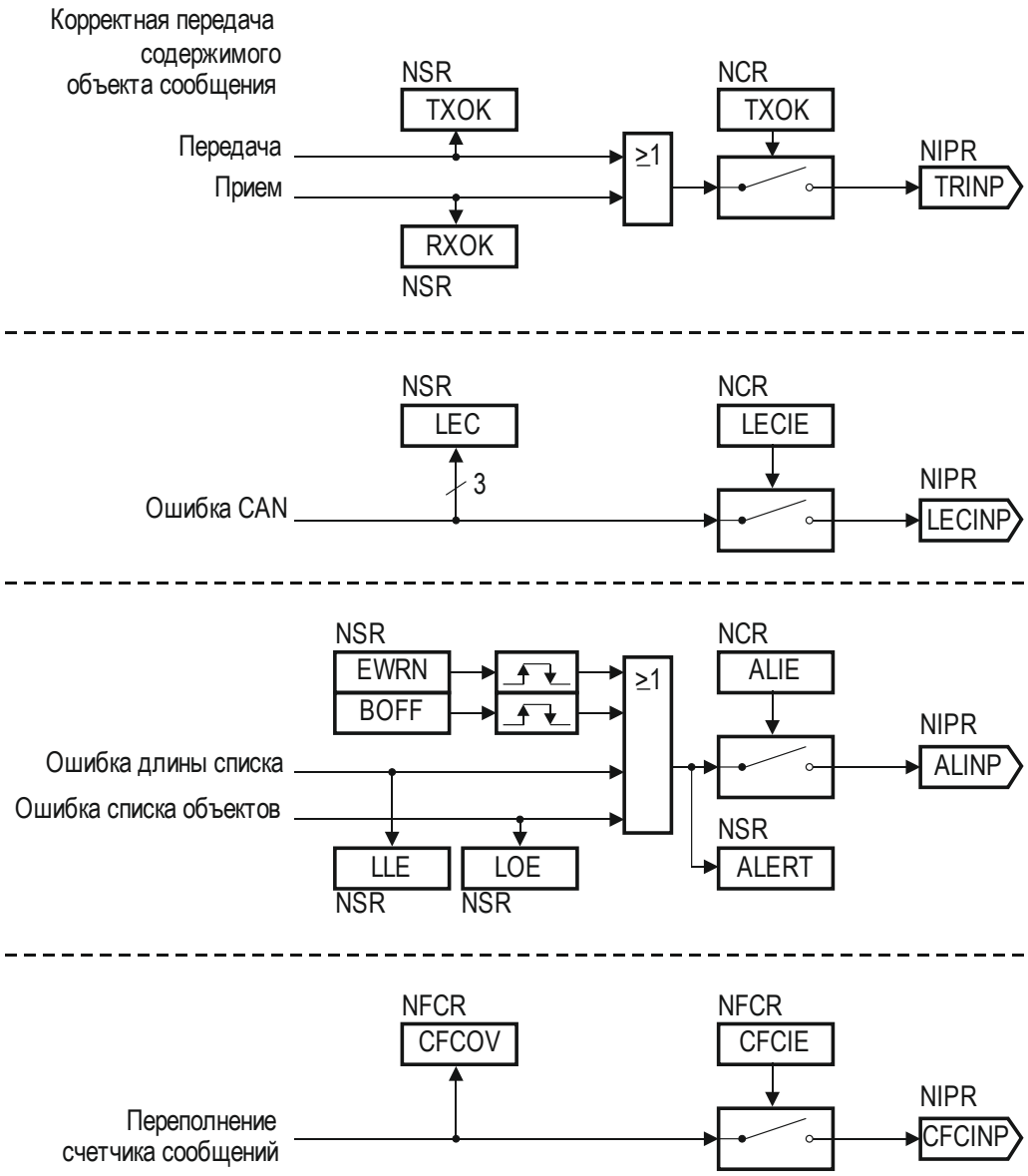


Рисунок 26.15 – Прерывания CAN узла

Прерывание узла при переполнении счетчика сообщений генерируется, если оно разрешено битом CFCIE регистра NFCR. Битовое поле CFCINP задает линию прерывания.

Прерывание ALERT может быть сформировано (если разрешено битом ALERT) любым из следующих событий:

- изменение состояния бита BOFF;
- изменение состояния бита EWRN;
- ошибка длины списка, которая также выставляет бит LLE;
- ошибка элемента списка, которая также выставляет бит LOE;
- бит INIT выставлен аппаратно.

Битовое поле ALINP задает линию прерывания.

В дополнение к аппаратным прерываниям есть возможность программного генерирования прерываний с использованием регистра прерываний MITR. Запись единицы в n-й разряд битового поля IT генерирует сигнал запроса прерывания на соответствующей ему n-ой линии прерываний (одной из шестнадцати). Установка нескольких битов приводит к параллельному генерированию запросов прерываний на соответствующих установленным битам линиях прерываний.

26.4 Объекты сообщений

Регистры управления и состояния объектов сообщений

В состав каждого объекта сообщения входят девять 32-разрядных регистров:

- управления и состояния – MOCTR (только запись) и MOSTAT (только чтение), доступные по одному адресу;
- арбитража – MOAR;
- данных – MODATAH и MODATAL;
- маски – MOAMR;
- указателя прерываний – MOIPR;
- указателя FIFO или шлюза – MOFGPR;
- управления функционированием – MOFCR.

Расположение регистров представлено на рисунке 26.16, где для примера взят пятый объект сообщения.

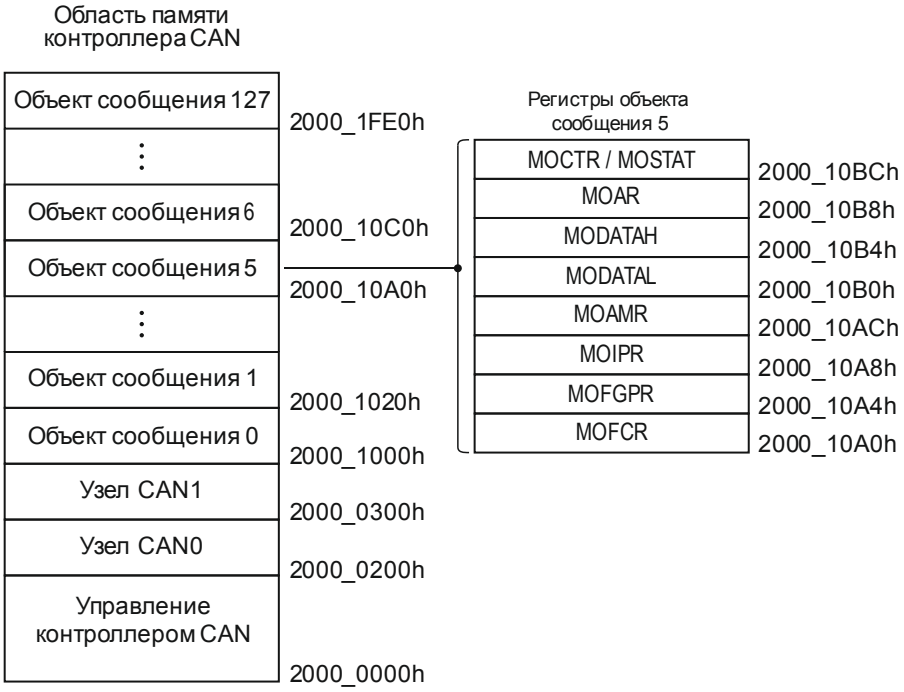


Рисунок 26.16 – Структура памяти регистров

Объекты сообщений контроллера CAN могут быть организованы в восемь списков, см. рисунок 26.17.

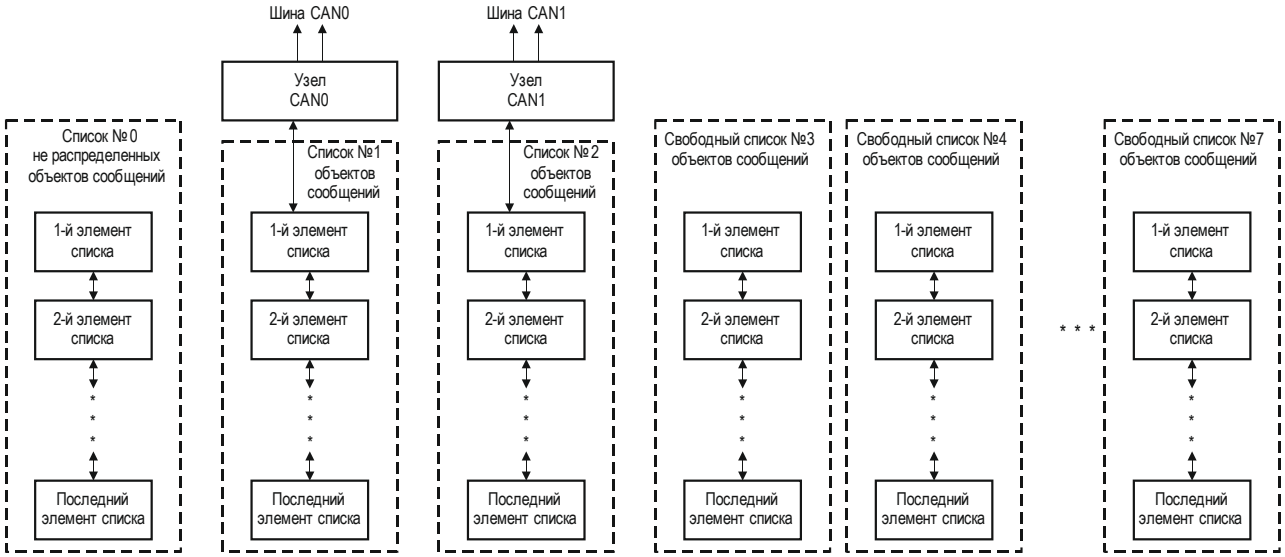


Рисунок 26.17 – Списки контроллера CAN

Каждый объект сообщения может быть добавлен в один из списков. Каждый узел CAN имеет свой список и соответствующий регистр списка. Регистр LIST1 отражает состояние списка №1 узла CAN0, регистр LIST2 – списка № 2 узла CAN1.

Примечание – Узел может оперировать только с теми объектами сообщений, которые занесены в принадлежащий ему список.

Положение объекта сообщения Msg_x в списке определяется посредством регистра MOSTAT, который содержит указатели на предшествующий ему и следующий за ним элементы списка (объекты). Нераспределенные между узлами CAN объекты сообщений по умолчанию организуются в отдельный список № 0, состояние которого отражается в регистре LIST0. Остальные пять списков с номерами от 3 до 7 являются свободными (не принадлежат ни одному узлу) и имеют соответствующие регистры LIST3 – LIST7.

Примечание – Объекты сообщений, распределенные в списки с 3 по 7, не могут быть использованы узлами CAN.

Механизмы FIFO и шлюза оперируют с объектами сообщений независимо от их распределения по спискам, что дает возможность работы со всеми восемью списками. Следовательно, при использовании механизмов FIFO и шлюза следует внимательно следить за содержимым списков.

На рисунке 26.18 представлен вариант, когда объекты сообщений с номерами 3, 5 и 16 занесены в список № 2, принадлежащий узлу CAN1. Состояние списка отражено в регистре LIST2.

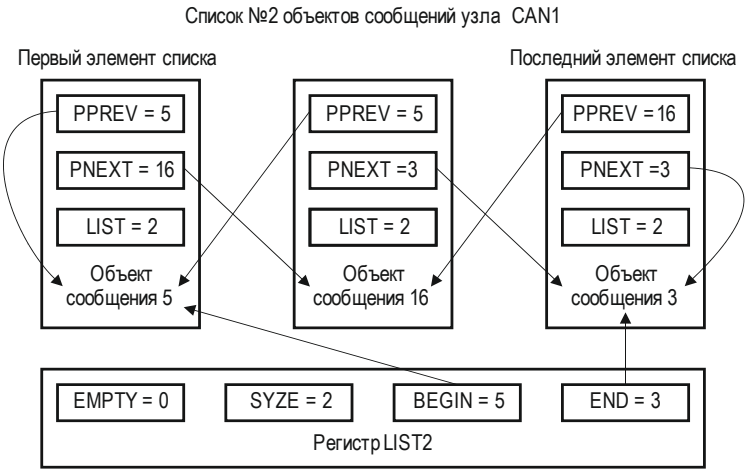


Рисунок 26.18 – Пример списка объектов сообщений

Значение поля BEGIN регистра LIST2 указывает на первый элемент списка (объект сообщения 5). Значение поля END указывает на последний элемент списка (объект сообщения 3). Количество элементов списка (количество объектов сообщений в списке) отражается в поле SIZE (значение SIZE всегда на единицу меньше количества элементов списка). Бит EMPTY является индикатором заполнения списка. Если список пуст, бит EMPTY установлен, в противном случае бит сброшен.

Каждый объект сообщения содержит номер списка (поле LIST), к которому он относится, а также указатели PNEXT и PPREV на следующий по списку объект сообщения и предшествующий соответственно. Поле PPREV первого по списку объекта сообщения должно указывать на этот же объект. Поле PNEXT последнего по списку объекта сообщения должно указывать на этот же объект.

На рисунке 26.18 указатель PPREV пятого объекта сообщения (первого в списке) имеет значение 5h, а указатель PNEXT третьего объекта сообщения (последнего в списке) имеет значение 3h. Значение поля LIST всех трех объектов сообщений равно 2h.

Объект сообщения, у которого LIST = 0h относится к нулевому списку нераспределенных объектов. После сброса все объекты сообщений считаются нераспределенными. По умолчанию порядок элементов списка № 0 следующий: объект сообщения (n – 1) является предыдущим объекта сообщения Msg_x, а объект сообщения (Msg_x + 1) – следующим.

Для просмотра структуры списка объектов сообщений узла достаточно обратиться к соответствующим регистрам LIST1/LIST2 и MOSTAT.

Структура списка управляется и изменяется посредством контроллера списка, который, в свою очередь, управляется панелью команд, основное назначение которой – упрощение внесения изменений в структуру списка, отслеживание этих изменений и проверка их корректности с помощью регистра PANCTR.

Панель команд запускается записью соответствующей команды в битовое поле PANCMD. До записи кода команды должны быть записаны соответствующие аргументы команды в битовые поля PANAR1 и PANAR2.

Примечание – Запись новых значений в поля PANAR1 и PANAR2 не изменяет сразу их содержимого. Новые значения сначала попадают в специальный теневой регистр. Далее, одновременно с записью кода команды в поле PANCMD, новые значения из теневого регистра переносятся в поля PANAR1 и PANAR2.

С записью корректного кода команды выставляется флаг BUSY, и в дальнейшем все попытки записи в регистр PANCTR игнорируются. Флаг BUSY остается активным, а панель команд заблокированной до тех пор, пока не завершится выполнение записанной команды.

После сброса микроконтроллера контроллер списка формирует список № 0 нераспределенных объектов сообщений. Во время этой операции флаг BUSY установлен, и все обращения к объектам сообщений запрещены. По окончании этой операции флаг BUSY сбрасывается, и объекты становятся доступными.

В случае появления команды динамического распределения, по которой какой-либо элемент забирается из списка № 0 и переносится в другой указанный список, наряду с битом BUSY, устанавливается бит RBUSY. Это указывает на то, что значения битовых полей PANAR1 и PANAR2 будут обновлены контроллером списка, следующим образом:

- номер объекта сообщения, переносимого из списка № 0 нераспределенных объектов сообщений, записывается в PANAR1;

- если установлен бит ERR (седьмой бит поля PANAR2), значит список № 0 пуст и выполнение команды завершается; если бит ERR сброшен – список № 0 не пуст и команда выполняется.

Результаты выполнения команды динамического распределения записываются до того, как контроллер списка начнет процесс распределения. Как только результаты станут доступны, бит RBUSY сбрасывается. Это позволяет пользователю запрограммировать настройки желаемого объекта сообщения, в то время как контроллер списка распределяет объекты. Во время операций со списками доступ к объектам сообщений не запрещен, но

следует помнить, что любой доступ к регистрам объектов сообщений в течение процесса распределения объектов вносит задержку (в процесс), равную длительности доступа.

Код команды «нет операции» автоматически записывается в битовое поле PANCMD.

Новая команда может быть записана в любое время, когда бит BUSY сброшен.

Все битовые поля регистра PANCTR, исключая биты BUSY и RBUSY, могут быть записаны программно, что делает возможным сохранять и восстанавливать значения регистра PANCTR, если панель команд используется независимой подпрограммой обработки прерываний. Если возникает такая ситуация, то любые задачи, которые используют панель команд и которые могут прерывать выполнение других задач, тоже использующих панель команд, будут опрашивать состояние флага BUSY. До тех пор, пока флаг BUSY будет оставаться установленным, содержимое регистра PANCTR будет сохранено в соответствующей области памяти до операции восстановления. Как только подпрограмма обработки прерываний закончится, содержимое регистра PANCTR будет восстановлено.

До того, как объект сообщения, занесенный в список активного узла CAN, будет перенесен на другую позицию этого же списка или перенесен в другой список, бит MSGVAL регистра MOSTAT объекта сообщения Msg_x должен быть очищен.

Примечание – Если требуется перераспределить объекты сообщений в списки повторно, необходимо приостановить работу узлов CAN (установить бит INIT регистра NCR), а после занесения объектов в списки возобновить ее (сбросить бит INIT).

26.5 Прием и передача сообщений

Прием сообщения

После завершения приема сообщение сохраняется в объекте сообщения в соответствии с установленным алгоритмом, см. рисунок 26.19.

Помимо сохранения данных в объекте сообщения, контроллер CAN осуществляет обмен данными с ЦП.

При приеме сообщения информация сохраняется в объекте сообщения только в том случае, если установлен бит MSGVAL регистра MOSTAT. Если ЦП очищает бит MSGVAL, контроллер CAN останавливает запись в объект сообщения, и далее объект может быть реконфигурирован центральным процессором с последующей записью в него информации без участия контроллера CAN.

Полученное с шины сообщение может быть сохранено в объекте сообщения только в случае, если установлен бит RXEN. Контроллер CAN проверяет состояние бита RXEN только во время фильтрации принимаемого сообщения. После того, как сообщение принято, состояние бита не имеет значения и не оказывает влияния на дальнейшее сохранение данных в объекте сообщения.

Бит RXEN позволяет управлять блокированием объекта сообщения – после сброса бита RXEN полученное сообщение сохраняется в объекте сообщения, который получил приоритет, но в сохранении последующих сообщений этот объект не принимает участия.

Реконфигурация объекта сообщения центральным процессором во время работы контроллера CAN (например, сброс бита MSGVAL, изменение объекта сообщения и повторная установка бита MSGVAL) происходят следующим образом:

- объект сообщения получает приоритет;
- ЦП очищает бит MSGVAL для реконфигурации объекта сообщения;
- после реконфигурации ЦП снова устанавливает бит MSGVAL;
- завершается получение сообщения;
- если установлен бит MSGVAL, полученные данные сохраняются в объекте сообщения, генерируется запрос на прерывание, устанавливается соответствующий флаг;
- если сконфигурировано, производятся шлюзовые и FIFO операции.

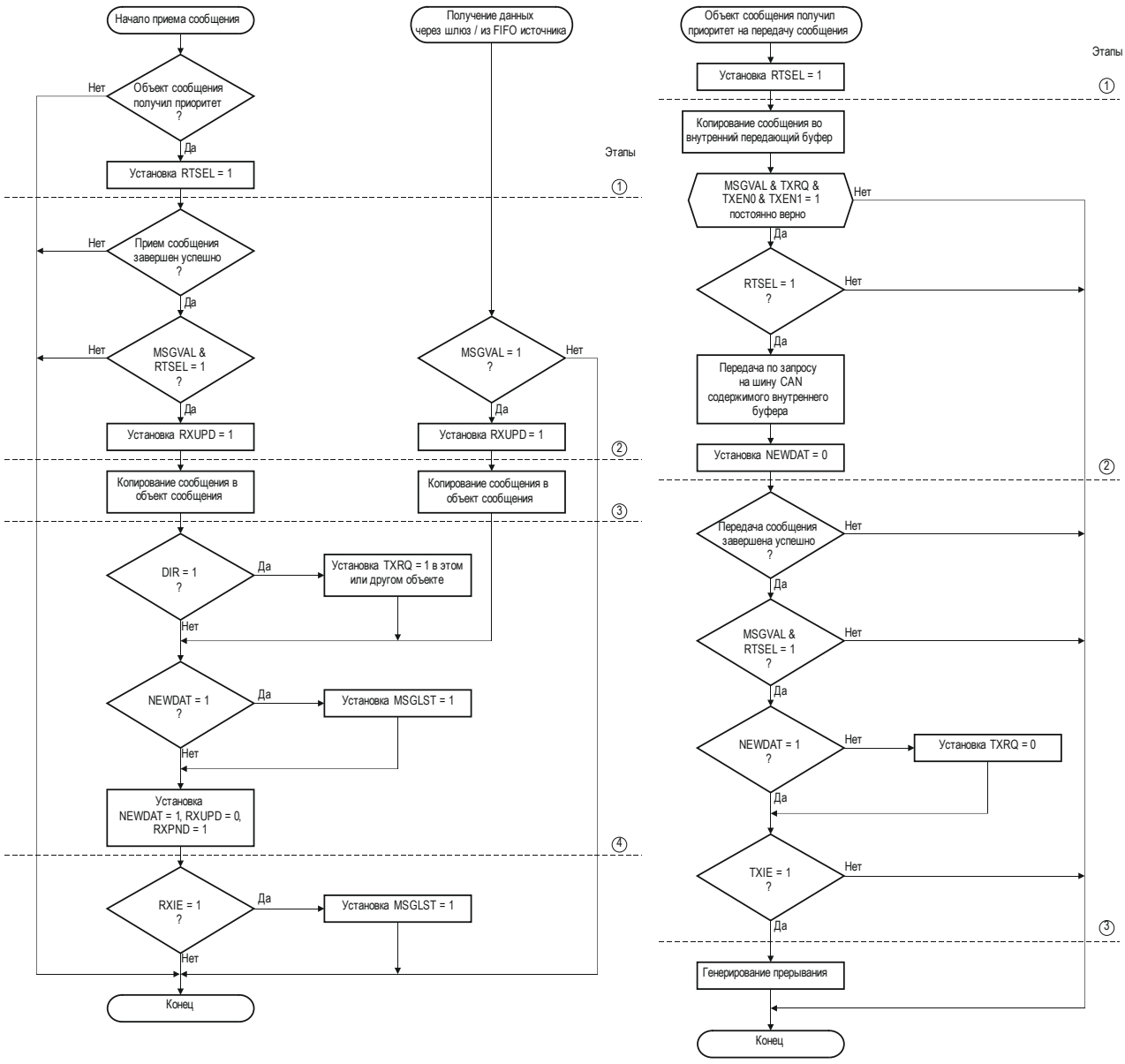


Рисунок 26.19 – Алгоритмы приема и передачи сообщения

Примечание – После реконфигурации объекта сохранение данных по завершении получения сообщения может быть нежелательным. Запретить запись данных в объект сообщения можно посредством бита RTSEL.

После получения объектом сообщения приоритета, его бит RTSEL устанавливается контроллером CAN, открывая, таким образом, объект сообщения для записи. После приема сообщения контроллер CAN дополнительно проверяет возможность записи в объект сообщения, а именно – установлен ли все еще бит RTSEL. И только в том случае, если бит RTSEL установлен, полученные данные сохраняются в объекте сообщения (вместе со всеми последующими действиями, которые указаны выше).

Если во время операций контроллера CAN объект сообщения становится некорректным (сброс бита MSGVAL), бит RTSEL должен быть сброшен до того, как бит MSGVAL будет установлен снова, или, по крайней мере, одновременно с ним. Это необходимо для предотвращения сохранения старой информации в объекте сообщения.

Реконфигурация объекта сообщения должна происходить следующим образом:

- сброс бита MSGVAL;
- реконфигурация объекта сообщения, пока бит MSGVAL сброшен;
- сброс бита RTSEL и далее установка бита MSGVAL.

Индикатором процесса сохранения (изменения) данных в объекте сообщения является флаг RXUPD, который выставляется с началом процесса сохранения (изменения) и сбрасывается с его окончанием.

После сохранения полученного сообщения (идентификатора, бита IDE, кода длины данных, поля данных, в случае сообщения данных) выставляется флаг NEWDAT. Если к моменту выставления (завершение сохранения/изменения данных) флаг NEWDAT был уже установлен, выставляется флаг MSGLST, который говорит о том, что произошла потеря данных.

Флаги RXUPD и NEWDAT позволяют произвести чтение корректных данных из объекта сообщения во время текущих операций контроллера CAN. Рекомендуемая последовательность действий следующая:

- сброс флага NEWDAT;
- чтение данных (идентификатор, данные и т. д.) из объекта сообщения;
- проверка флагов NEWDAT и RXUPD – оба флага должны быть сброшены. В случае невыполнения этого условия – возвращение к первому действию;
- если флаги NEWDAT и RXUPD сброшены, то содержимое объекта сообщения корректно и не используется контроллером CAN в течение операции чтения.

Поведение флагов RXUPD, NEWDAT и MSGLST идентично как для сообщений данных, так и для сообщений удаленных запросов.

Передача сообщения

Алгоритм передачи сообщений показан на рисунке 26.19. Одновременно с копированием данных (идентификатора, бита IDE, бита RTR, равного биту DIR, кода длины данных и собственно данных) из объекта сообщения, содержимое которого должно быть передано во внутренний передающий буфер соответствующего узла CAN, для контроля соблюдения четкой последовательности выполнения всех операций устанавливаются биты состояния.

Сообщение может быть передано только в случае, когда все четыре бита MSGVAL, TXEN0, TXEN1 и TXRQ установлены.

Бит RTSEL выставляется после того, как объект сообщения получает приоритет для передачи своего содержимого. Когда данные объекта сообщения копируются в передающий буфер, бит RTSEL проверяется и, если он установлен, сообщение передается. После успешной передачи сообщения бит RTSEL проверяется снова и, если он установлен, осуществляются дальнейшие операции.

Для полной и завершенной реконфигурации корректного объекта сообщения должны быть выполнены следующие шаги:

- очистка бита MSGVAL;
- реконфигурация объекта сообщения, пока бит MSGVAL сброшен;
- сброс бита RTSEL и установка бита MSGVAL.

Сброс бита RTSEL гарантирует как полное отключение объекта сообщения от текущей передачи, так и то, что никакие операции (копирование данных в передающий буфер, включая сброс бита NEWDAT, очистка бита TXRQ, прерывание сообщения и т. д.), относящиеся к старой конфигурации этого объекта сообщения, не повлияют на новую конфигурацию после установки бита MSGVAL.

После завершения передачи содержимого объекта сообщения в передающий буфер узла CAN, флаг NEWDAT аппаратно сбрасывается, тем самым обозначая, что объект сообщения открыт для записи новых данных.

Если после успешной передачи сообщения (на шину CAN) флаг NEWDAT все еще остается сброшенным (в объект сообщения не были записаны новые данные), флаг TXRQ аппаратно сбрасывается. Если же флаг NEWDAT был установлен программно (в связи с необходимостью передачи новых данных), флаг TXRQ не сбрасывается, тем самым разрешая передачу новых данных.

26.6 Фильтрация сообщений

Фильтрация при получении сообщений

При получении узлом CAN сообщения определяется объект сообщения, в котором будут сохранены получаемые данные в случае успешного приема.

Объект сообщения считается корректным для приема, если одновременно соблюдаются условия:

- объект сообщения распределен в список объектов сообщений узла, который принимает сообщение;
- бит MSGVAL установлен;
- бит RXEN установлен;
- бит DIR равен биту RTR принимаемого сообщения. Если бит DIR установлен, объект сообщения (объект передачи) может принять только сообщение удаленного запроса. Если бит DIR сброшен (объект приема), объект сообщения может принять только сообщение данных;
- если бит MIDE установлен, то бит IDE получаемого сообщения оказывает следующее влияние:
  - если бит IDE (регистр MOAR) установлен, то бит IDE принимаемого сообщения должен быть равен единице (расширенный идентификатор);
  - если бит IDE сброшен, бит IDE принимаемого сообщения должен быть равен нулю (стандартный идентификатор);
  - если бит MIDE сброшен, значение бита IDE принимаемого сообщения не важно, т. е. допускаются сообщения как со стандартным, так и с расширенным идентификатором;
- идентификатор полученного сообщения полностью (побитно) совпадает с идентификатором, хранящимся в регистре MOARn объекта сообщения, за исключением битов, закрытых маской регистра MOAMRn, значение которых не важно.

На рисунке 26.20 показан пример проверки идентификатора.

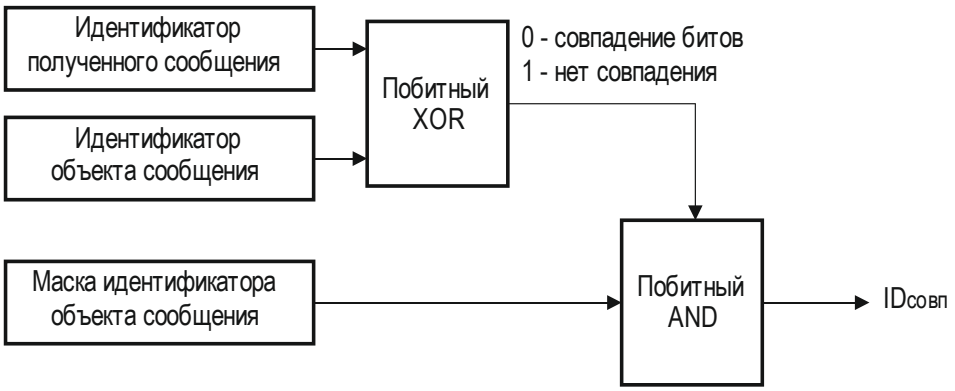


Рисунок 26.20 – Проверка идентификатора полученного сообщения

Среди всех объектов сообщений, которые отвечают указанным выше критериям, для сохранения полученного сообщения выбирается объект с наивысшим приоритетом. Для задания приоритета используется поле PRI в регистре MOAR. Объект сообщения, у которого значение поля PRI меньше, имеет больший приоритет. При равенстве значений поля PRI приоритетным считается объект сообщения, который предшествует следующему в списке.

Фильтрация при передаче сообщений

Когда требуется передача содержимого какого-либо объекта сообщения, в соответствующих управляющих регистрах выставляются флаги, указывающие на необходимость передачи. Объект сообщения считается корректным для передачи, если одновременно соблюдаются условия:

- объект сообщения распределен в список объектов сообщений узла CAN;
- флаг MSGVAL установлен;
- флаг TXRQ установлен;
- флаги TXEN0 и TXEN1 установлены.

Может возникнуть ситуация, когда передачи требуют одновременно несколько объектов сообщений. Среди всех объектов, которые отвечают указанным выше критериям, для передачи выбирается объект с наивысшим приоритетом.

Объект сообщения, у которого значение поля PRI меньше, имеет больший приоритет. При равенстве значений поля PRI разных объектов приоритет определяется следующим образом:

- при PRI = 10b – согласно правилам арбитража передачи сообщения;
- при PRI = 01b/11b приоритет имеет объект сообщения, который предшествует следующему в списке.

Объект сообщения, являющийся корректным для передачи и имеющий приоритет, будет осуществлять передачу первым. Остальные объекты сообщений будут переданы по очереди, согласно их приоритетам.

Объект сообщения определяется как стандартный объект сообщения, если в регистре MOFCR значение битового поля MMC равно нулю. Стандартный объект сообщения может принимать и передавать сообщения, согласно правилам, описанным выше.

На рисунке 26.21 показано формирование запроса на передачу объекта сообщения.

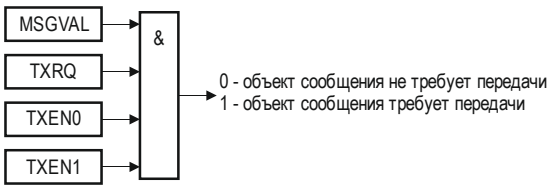


Рисунок 26.21 – Формирование запроса на передачу объекта сообщения

26.7 Удаленные запросы

После получения узлом CAN сообщения удаленного запроса и сохранения его в объекте сообщения, выставляется бит запроса передачи для ответа на удаленный запрос (отправка сообщения данных) или для автоматического повторения запроса.

В зависимости от состояния бита FRREN объекта сообщения, который принял сообщение удаленного запроса, возможны два варианта действий:

- если бит FRREN сброшен, то устанавливается флаг TXRQ этого объекта;
- если бит FRREN установлен, то устанавливается флаг TXRQ того объекта, на который указывает поле CUR объекта, принявшего удаленный запрос. При этом поле CUR не меняет своего значения.

Состояние регистров объекта сообщения, передающего сообщение удаленного запроса

У объекта сообщения, передающего сообщение удаленного запроса, в регистре MOSTAT должен быть сброшен бит DIR (объект передает сообщение данных) и установлены биты TXEN0, TXEN1, MSGVAL и TXRQ. Значение идентификатора в регистре MOAR передающего объекта сообщения должно быть равно значению идентификатора принимающего объекта сообщения (или совместно с регистром MOAMR

обеспечивать успешное прохождение фильтрации), чтобы сообщение удаленного запроса было принято принимающим объектом другого узла. Само сообщение удаленного запроса должно содержать идентификатор принимающего объекта сообщения, поэтому значение регистра MODATAL передающего объекта сообщения должно быть равно значению регистра MOAR принимающего объекта.

Состояние регистров объекта сообщения, принимающего сообщение удаленного запроса при FRREN = 0

У объекта сообщения, принимающего сообщение удаленного запроса, должны быть установлены биты DIR (объект принимает сообщение удаленного запроса), TXEN0 и TXEN1 (если ответ на запрос будет отправлен объектом), RXEN и MSGVAL. Регистры MODATAL и MODATAH должны содержать данные, которые будут переданы в ответ на запрос.

Состояние регистров объекта сообщения, принимающего сообщение удаленного запроса (при FRREN = 1) и содержащего данные для ответа на запрос

У объекта сообщения, принимающего сообщение удаленного запроса, должны быть установлены биты DIR, RXEN и MSGVAL. Битовое поле CUR должно указывать на номер объекта сообщения (должен находиться в том же узле, что и объект принявший сообщение удаленного запроса), содержащего данные, предназначенные для передачи в ответ на поступивший удаленный запрос.

В свою очередь у объекта сообщения, хранящего данные для отправки в ответ на запрос, должны быть установлены биты DIR, (объект передает сообщение данных), TXEN0, TXEN1 и MSGVAL. Бит TXRQ устанавливается автоматически при приеме сообщения удаленного запроса принимающим объектом сообщения.

Прием ответа на запрос (переданного сообщения данных) осуществляется стандартным объектом сообщения запрашивающего узла CAN (обмен данными происходит между объектом сообщения, хранящим данные для отправки в ответ на запрос, и объектом сообщения запрашивающего узла).

26.8 Дополнительные режимы передачи

Дополнительно имеются два режима, каждый из которых может быть выбран индивидуально:

- режим передачи данных с защитой от повторений;
- режим однократной пересылки данных.

Режим передачи данных с защитой от повторения

Режим передачи данных с защитой от повторения выбирается установкой бита SDT регистра MOFCR.

После приема сообщения данных и сохранения его в объекте с установленным битом SDT, бит MSGVAL этого объекта аппаратно сбрасывается, чтобы исключить возможность повторного приема и записи в этот объект. Этот режим нельзя использовать для базового объекта FIFO структуры.

В ответ на сообщение удаленного запроса, принятое объектом с установленным битом SDT, будут отправлены данные из объекта сообщения, на который указывает поле CUR объекта, принявшего удаленный запрос. После этого бит MSGVAL объекта, принявшего сообщение удаленного запроса, сбросится.

Примечание – Объект, принявший сообщение удаленного запроса, не может быть источником данных, передаваемых в ответ на запрос. Это означает, что в данном режиме бит FRREN объекта, принявшего удаленный запрос, обязательно должен быть установлен.

Режим однократной пересылки данных

Режим однократной пересылки данных выбирается установкой бита STT.

Бит TXRQ сбрасывается, когда содержимое объекта сообщения копируется в передающий буфер узла CAN. Таким образом, в дальнейшем, при неудачной (вследствие ошибок) пересылке сообщения по CAN-шине, повторной передачи не будет.

26.9 FIFO структура объектов сообщений

Регистр MOFGPR объекта сообщения Msg_x содержит установки указателей на объекты сообщений, которые используются при операциях FIFO и шлюзовых операциях.

В случае сильной загрузки ЦП обработка серии сообщений может быть затруднена – например, вследствие получения и/или передачи большого числа сообщений за малые промежутки времени. Для таких случаев предусмотрена система буферов быстрого ввода-вывода, так называемая FIFO структура, которая может функционировать автоматически, и позволяет избежать потери принимаемых сообщений, минимизировать время подготовки сообщений к отправке, а также генерировать прерывания по окончании операций.

Допускается организация нескольких параллельных FIFO структур. Число структур и их составляющих зависит только от количества доступных объектов сообщений. FIFO структура может быть создана, изменена и удалена в любой момент времени, даже во время операций контроллера CAN.

На рисунке 26.22 представлена основная FIFO структура. Она состоит из одного базового объекта и n числа вспомогательных объектов.

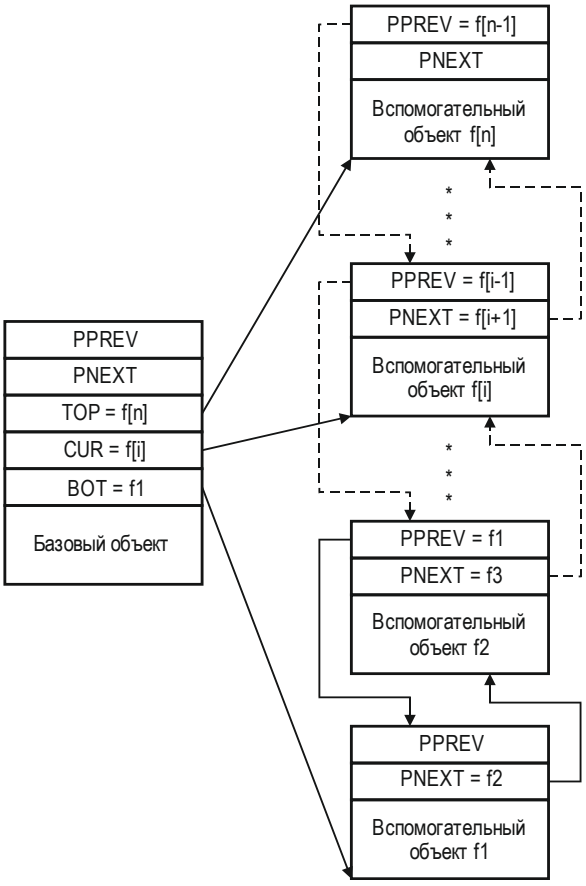


Рисунок 26.22 – FIFO структура с базовым объектом и n вспомогательными объектами

Вспомогательные объекты передающей FIFO структуры объединяются последовательно в списки (подобно спискам объектов сообщений). Базовый объект может быть занесен в любой список. Хотя на рисунке 26.22 базовый объект не относится ни к одному из списков, он может быть вставлен в любую последовательность вспомогательных объектов. Это означает, что базовый объект одновременно является и вспомогательным объектом (шлюзовые операции не возможны). Порядковые номера объектов сообщений (0, 1, 2 и т. д.) не имеют никакого значения при FIFO операциях с объектами.

Вспомогательные объекты должны быть определены в общий список (так как они последовательно связаны). С помощью указателей (битовые поля BOT, CUR и TOP) можно присоединять базовый объект к вспомогательному объекту, независимо от того, принадлежат базовый и вспомогательный объекты одному списку или разным спискам, но базовый должен быть первым в списке в таком случае.

Минимальная FIFO структура может состоять из одного объекта сообщения, который будет одновременно являться и базовым, и вспомогательным (фактически не используется). Максимальная FIFO структура может включать в себя все 256 объектов сообщений.

В базовом объекте FIFO границы установлены: поле BOT указывает на самый младший элемент FIFO структуры, поле TOP – на самый старший элемент, поле CUR – на вспомогательный объект, который в настоящий момент выбран контроллером CAN для передачи сообщения. Как только начинается передача, в CUR записывается номер следующего по списку вспомогательного объекта сообщения (CUR = PNEXT используемого объекта). Если значение битового поля CUR достигло номера старшего элемента списка (CUR = TOP), то следующим значением будет BOT (реализация автоматического перехода в начало списка). Таким образом, реализуется замкнутая FIFO структура, в которой битовые поля TOP и BOT устанавливают связь между началом и концом списка.

Битовое поле SEL позволяет определить вспомогательный объект, в пределах списка, для которого генерируется прерывание всякий раз, когда указатель CUR достигает значения указателя SEL. Также битовое поле SEL позволяет отследить окончание запланированной передачи серии сообщений или выдать прерывание, предупреждающее о том, что FIFO структура становится заполненной.

Вспомогательные объекты приемной FIFO структуры могут принадлежать списку любого узла.

FIFO структура для приема

FIFO структура для приема используется для буферизации входящих сообщений данных и удаленных запросов.

FIFO структура для приема активируется записью значения 0001b в битовое поле MMC регистра MOFCR базового объекта. Эта запись автоматически определяет объект как базовый объект приема FIFO. Типы вспомогательных объектов FIFO не имеют значения при операциях.

Когда базовый объект FIFO получает сообщение, оно сохраняется не в этом базовом объекте, а во вспомогательном объекте сообщения, на который указывает битовое поле CUR. При этом по умолчанию предполагается, что для вспомогательного объекта MMC = 0000b (действительное значение MMC игнорируется) и никаких операций фильтрации принимаемого сообщения не производится.

Одновременно с приемом сообщения текущее значение указателя CUR базового объекта меняется на номер следующего по списку вспомогательного объекта FIFO структуры. Этот вспомогательный объект будет использован для приема следующего сообщения.

Если установлен флаг OVIE регистра MOFCR базового объекта и значение указателя CUR становится равным значению указателя SEL, генерируется прерывание переполнения. Это прерывание генерируется на узле прерываний с указателем TXINP базового объекта

сразу после сохранения полученного сообщения во вспомогательном объекте. Прерывания генерируются, если это разрешено битом TXIE.

Следует помнить, что сообщение сохраняется в базовом и вспомогательном объектах FIFO, только если установлен бит MSGVAL.

Во избежание непосредственного приема сообщения вспомогательным объектом, как если бы он был независимым объектом и не принадлежал FIFO структуре, флаги RXEN всех вспомогательных объектов должны быть сброшены. Состояние флага RXEN неважно в случае, когда вспомогательный объект занесен в список, не связанный с узлом CAN.

FIFO структура для передачи

FIFO структура для передачи используется с целью буферизации серий сообщений данных или удаленных запросов, которые должны быть отправлены. FIFO структура для передачи состоит из базового объекта и одного или более вспомогательных объектов.

FIFO структура для передачи активируется записью значения 0010b в поле MMC регистра MOFCR базового объекта. В отличие от FIFO структуры для приема, в битовые поля MMC вспомогательных объектов (FIFO структуры для передачи) должно быть записано значение 0011b. Указатели CUR всех вспомогательных объектов должны указывать на базовый объект FIFO передачи (чтобы инициализироваться программно).

Флаги TXEN1 всех вспомогательных объектов сообщений, за исключением одного, на который указывает указатель CUR базового объекта, должны быть программно сброшены. Флаг TXEN1 указанного объекта должен быть установлен. Указатель CUR базового объекта может быть инициализирован для любого вспомогательного объекта.

При определении корректности объектов сообщений FIFO структуры для начала FIFO операций базовый объект должен быть определен первым как корректный, т. е. MSGVAL должен быть установлен.

В случае необходимости удаления FIFO структуры, прежде чем начнется операция удаления, все вспомогательные объекты, принадлежащие этой FIFO структуре, должны быть определены как некорректные (биты MSGVAL должны быть сброшены).

FIFO структура для передачи использует флаги TXEN1 всех своих объектов для выбора сообщения для передачи. В результате фильтрации право передавать сообщение получает тот объект, у которого выставлен флаг TXEN1. После передачи сообщения флаг TXEN1 аппаратно сбрасывается, а в указатель CUR записывается номер следующего объекта, требующего отправки сообщения, для которого уже выставлен (аппаратно) свой флаг TXEN1, и так далее – для всей FIFO структуры.

Если установлен флаг OVIE регистра MOFCRn базового объекта и значение указателя CUR становится равным значению указателя SEL, генерируется прерывание переполнения. Это прерывание генерируется на узле прерываний с указателем RXINP базового объекта после завершения операций получения сообщения. Прерывания приема базового объекта генерируются, если это разрешено битом RXIE.

Программирование регистров для FIFO структуры

1 Для передающего базового объекта:

- сбросить бит MSGVAL;
- задать поля CUR, BOT, TOP, SEL;
- записать значение 0010b в поле MMC, задать DLC, установить биты OVIE и RXIE (если необходимо).

Примечание – Состояние регистров MOAR и MOAMR передающего базового объекта не важно, поскольку в передаче участвуют передающие вспомогательные объекты и принимающий базовый объект. Поле RXINP указывает линию, на которую будет выдаваться прерывание переполнения (CUR = SEL).

2 Для передающих вспомогательных объектов:

- сбросить бит MSGVAL;

- установить биты DIR, TXEN1 (только для того вспомогательного объекта, на который указывает поле CUR передающего базового объекта, у остальных вспомогательных объектов бит TXEN1 должен быть сброшен), TXEN0;

- записать в поле CUR номер передающего базового объекта;

- записать значение 0011b в поле MMC, задать DLC.

Примечание – Значение регистров MOAR передающих вспомогательных объектов должно совпадать (или совместно с регистрами MOAMR обеспечивать успешное прохождение фильтрации) со значением регистра MOAR принимающего базового объекта, так как процесс передачи фактически происходит между ними (или иного принимающего объекта, если на приеме используется не FIFO структура).

3 Для принимающего базового объекта:

- установить бит RXEN;

- задать поля CUR, BOT, TOP, SEL;

- записать значение 0001b в поле MMC, задать DLC, установить биты OVIE и TXIE (если необходимо).

Примечание – Значение регистра MOAR принимающего базового объекта должно быть равно значению регистров MOAR передающих вспомогательных объектов передачи (или совместно с регистром MOAMR обеспечивать успешное прохождение фильтрации). Поле TXINP указывает, на какую линию будет выдаваться прерывание переполнения (прерывание после операции сохранения полученного сообщения во вспомогательных объектах при CUR = SEL).

4 Для принимающих вспомогательных объектов:

- сбросить бит RXEN (не требуется, если вспомогательные объекты занесены в список, не связанный с узлом CAN);

- задать поле DLC (состояние поля MMC не важно).

Примечание – Состояние регистров MOAR, принимающих вспомогательные объекты, не важно.

5 Установить бит MSGVAL в первую очередь у передающего базового объекта, а затем у всех остальных объектов.

6 Установить бит TXRQ для всех передающих вспомогательных объектов, начиная с того, на который указывает поле CUR передающего базового объекта.

26.10 Режим шлюза

Режим позволяет реализовывать автоматическую передачу информации через шлюз между двумя независимыми шинами CAN без участия ЦП.

Шлюз можно сформировать на уровне объектов сообщений и осуществлять передачу информации между узлами CAN. Шлюз может быть сформирован между двумя любыми объектами сообщений, принадлежащими разным узлам CAN. Количество шлюзов зависит только от количества объектов сообщений, допускающих формирование шлюзов.

Режим шлюза активируется записью значения 0100b в битовое поле MMC регистра MOFCR объекта сообщения Msg_x, инициализирует его как шлюзовый объект-источник. Объект сообщения, который будет являться шлюзовым объектом-приемником, выбирается указателем CUR объекта-источника. Для формирования шлюза достаточно, чтобы объект-приемник был корректным (установлен бит MSGVAL). Остальные параметры не влияют на возможность осуществления передачи между объектами от источника к приемнику.

Шлюзовый объект-источник, см. рисунок 26.23, функционирует как обычный объект сообщения, с тем отличием, что возможны дополнительные действия контроллера CAN при приеме и сохранении сообщения в объекте-приемнике:

1) Если установлен флаг DLCC регистра MOFCRn объекта-источника, код длины данных DLC копируется из шлюзового объекта-источника в шлюзовый объект-приемник.

2) Если установлен флаг IDC объекта-источника, идентификатор ID и расширение IDE копируются из шлюзового объекта-источника в шлюзовый объект-приемник.

3) Если установлен флаг DATC объекта-источника, байты данных, хранящиеся в двух регистрах MODATAL и MODATAH объекта-источника, копируются из шлюзового

объекта-источника в шлюзовый объект-приемник. Копируются все восемь байт данных, вне зависимости от значения поля DLC.

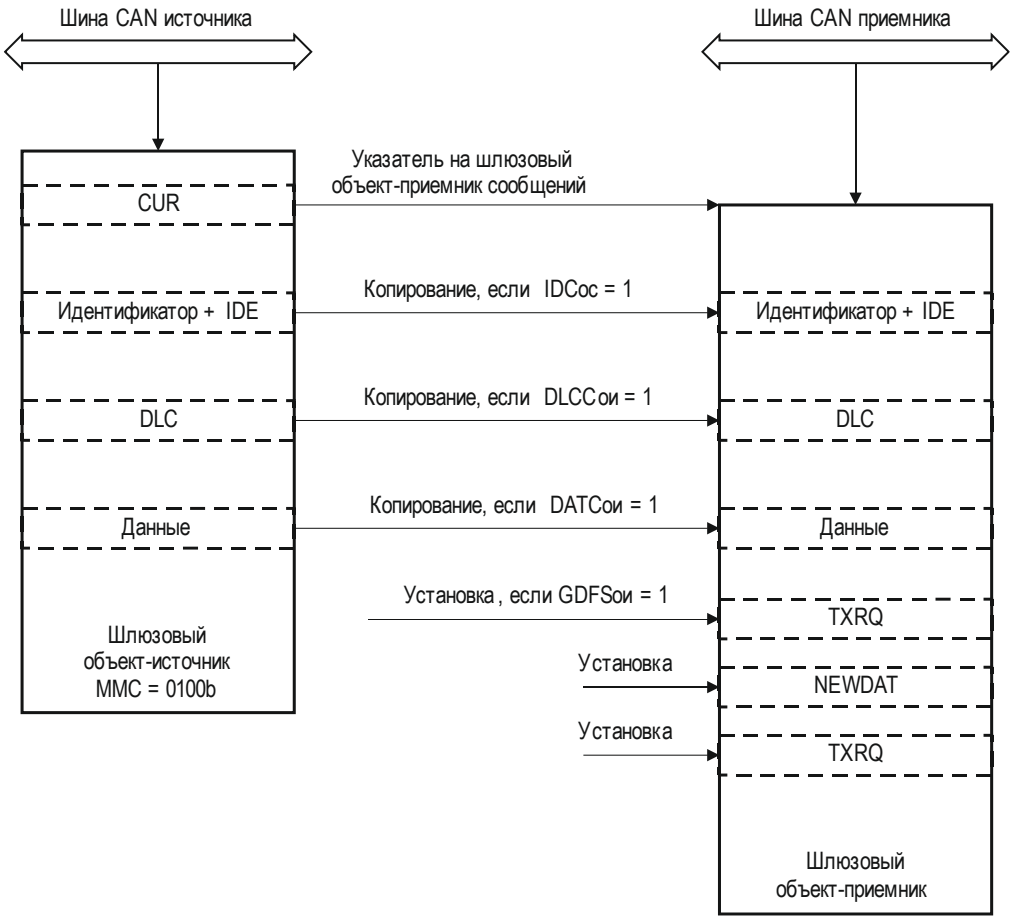


Рисунок 26.23 – Передача через шлюз от источника к приемнику

4) Если установлен флаг GDFS объекта-источника, то устанавливается бит запроса передачи TXRQ объекта-приемника.

5) Устанавливаются флаги RXPND и NEWDAT регистра MOSTAT объекта-приемника.

6) Если установлен флаг RXIE регистра MOSTAT объекта-приемника, то генерируется запрос на прерывание.

7) Указатель CUR объекта-источника переводится на следующий объект-приемник по правилам FIFO структуры. Сформировать шлюз между объектом-источником и одним объектом-приемником (значение указателя CUR будет оставаться неизменным) возможно программированием:

TOP = BOT = CUR = номер объекта-приемника.

Организация шлюза «объект-источник – объект-приемник» аналогична организации FIFO структуры «базовый объект – вспомогательный объект», что указывает на возможность формирования шлюза с интегрированным FIFO приемником. При получении сообщения данных (объект-источник является объектом приема, т. е. его бит DIR сброшен) и при получении удаленного запроса (объект-источник является объектом передачи) через шлюз используется один и тот же механизм.

Несмотря на то, что механизм удаленных запросов работает независимо от типа объекта сообщения, он наиболее полезен при использовании шлюзов, для формирования удаленных запросов на шине шлюзового объекта-источника после получения удаленного запроса на шине шлюзового объекта-приемника. В зависимости от значения бита FRREN шлюзового объекта-приемника, есть два варианта обработки удаленного запроса,

возникшего с той стороны шлюза, где расположен объект-приемник (при условии, что происходит передача из объекта-источника в объект-приемник, т. е. DIR (источника) = 0 и DIR (приемника) = 1):

1) Обработка запроса шлюзового объекта-приемника с FRREN = 0b:

- сообщение удаленного запроса принимается шлюзовым объектом-приемником;
- бит TXRQ шлюзового объекта-приемника устанавливается автоматически;
- сообщение данных с текущей информацией, хранящейся в объекте-приемнике, передается на шину приемника.

2) Обработка запроса шлюзового объекта-приемника с FRREN = 1b:

- сообщение удаленного запроса принимается шлюзовым объектом-приемником;
- бит TXRQ шлюзового объекта-источника (объект должен быть указан в поле CUR объекта-приемника), устанавливается автоматически;
- сообщение данных передается объектом-источником на шину CAN источника;
- получатель удаленного запроса в ответ выдает сообщение данных на шину источника;
- сообщение данных сохраняется в объекте-источнике;
- сообщение данных копируется в объект-приемник (через шлюз);
- выставляется бит TXRQ объекта-приемника (при условии, что GDFS источника = 1);
- новые данные, сохраненные в объекте-приемнике, передаются на шину приемника, в ответ на удаленный запрос на шине приемника.

Рекомендации по записи в регистры в режиме шлюза при передаче удаленного запроса с FRREN = 1

Обмен запрос – данные происходит в данном случае между стандартным объектом сообщения одного узла и объектом-приемником шлюза другого узла. Но при этом данные для ответа на запрос в шлюзовый объект-приемник поступают по шлюзу от объекта-источника. При получении удаленного запроса от объекта сообщения объектом-приемником флаг TXRQ устанавливается не у самого объекта-приемника, а у объекта-источника, благодаря установленному биту FRREN и битовому полю CUR (указывает на объект-источник) объекта-приемника. Данные из MODATAL и MODATAH объекта-источника копируются в MODATAL и MODATAH объекта-приемника (установлен бит DATC регистра MOFCR объекта-источника), вследствие чего автоматически устанавливается бит TXRQ регистра MOCTR объекта-приемника (установлен бит GDFS объекта-источника шлюза), и осуществляется передача сообщения данных (ответ на запрос) запрашивающему объекту сообщения.

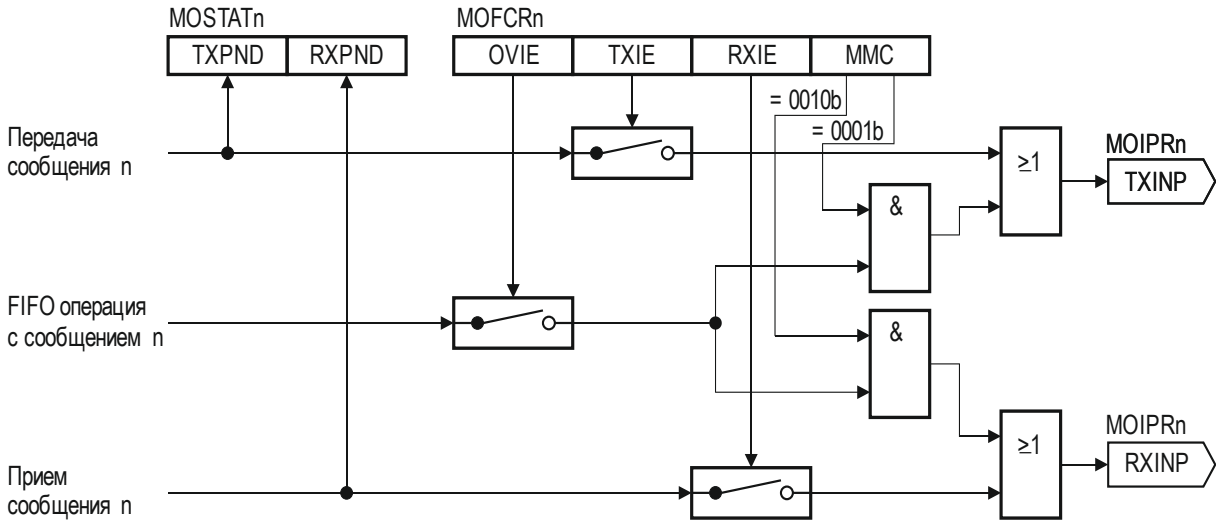
После успешного приема/передачи сообщения ЦП получает уведомление о завершении операции для задания дальнейших действий, связанных с объектом сообщения.

26.11 Прерывания объектов сообщений

После сохранения принятого сообщения в объект сообщения или успешной передачи формируется соответствующее прерывание. Каждый объект сообщения может формировать прерывания. Каждое прерывание направляется на одну из 16 выходных линий прерываний. Прерывания приема (после сохранения сообщения) также формируются после операций FIFO и шлюзовых операций. Флаги TXPND и RXPND всегда устанавливаются после успешной операции передачи/приема, независимо от состояния соответствующих флагов разрешения прерываний.

Объект сообщения может формировать FIFO прерывания. Если флаг OVIE регистра MOFCR установлен, то формирование FIFO прерывания будет зависеть от типа объекта сообщений, см. рисунок 26.24:

- если объект сообщения является принимающим базовым объектом, то выходная линия прерываний для этого объекта определяется битовым полем TXINP регистра MOIPR;
- если объект сообщения является передающим базовым объектом, то выходная линия прерываний определяется битовым полем RXINP.



MMC = 0001b : объект сообщения n является базовым объектом приема FIFO
MMC = 0010b : объект сообщения n является базовым объектом передачи FIFO

Рисунок 26.24 – Распределение прерываний

Ждущие сообщения

Когда генерируется запрос на прерывание (после приема/передачи сообщения), в одном из восьми регистров ждущих прерываний MSPNDx (x от 0 до 7) выставляется флаг ждущего сообщения. Восемь регистров образуют область из 32 × 8 бит – по два бита (один бит для операций приема и один бит для операций передачи) для каждого из объектов сообщений. Позиция флага ждущего сообщения определяется демультиплексорами DMUX, см. рисунки 26.25 и 26.26.

В зависимости от значения поля MPSEL регистра MCR, реализуется один из двух режимов выбора и установки флагов, ждущих сообщения:

- режим 1 в случае MPSEL = 0h;
- режим 2 в случае MPSEL = Fh.

Если нет необходимости в определении источника прерывания (прием или передача сообщения), то можно использовать любой из двух режимов, в противном случае, следует использовать второй режим.

В первом режиме установка флага ждущего сообщения происходит следующим образом:

- 7, 6 и 5 биты поля MPN выбирают регистр MSPNDx, в котором будет установлен флаг седьмого ждущего сообщения;

- пять младших бит поля MPN (на рисунке 26.25 выделены серым цветом) выбирают позицию флага (от 0 до 31), который будет установлен в выбранном регистре MSPNDx.

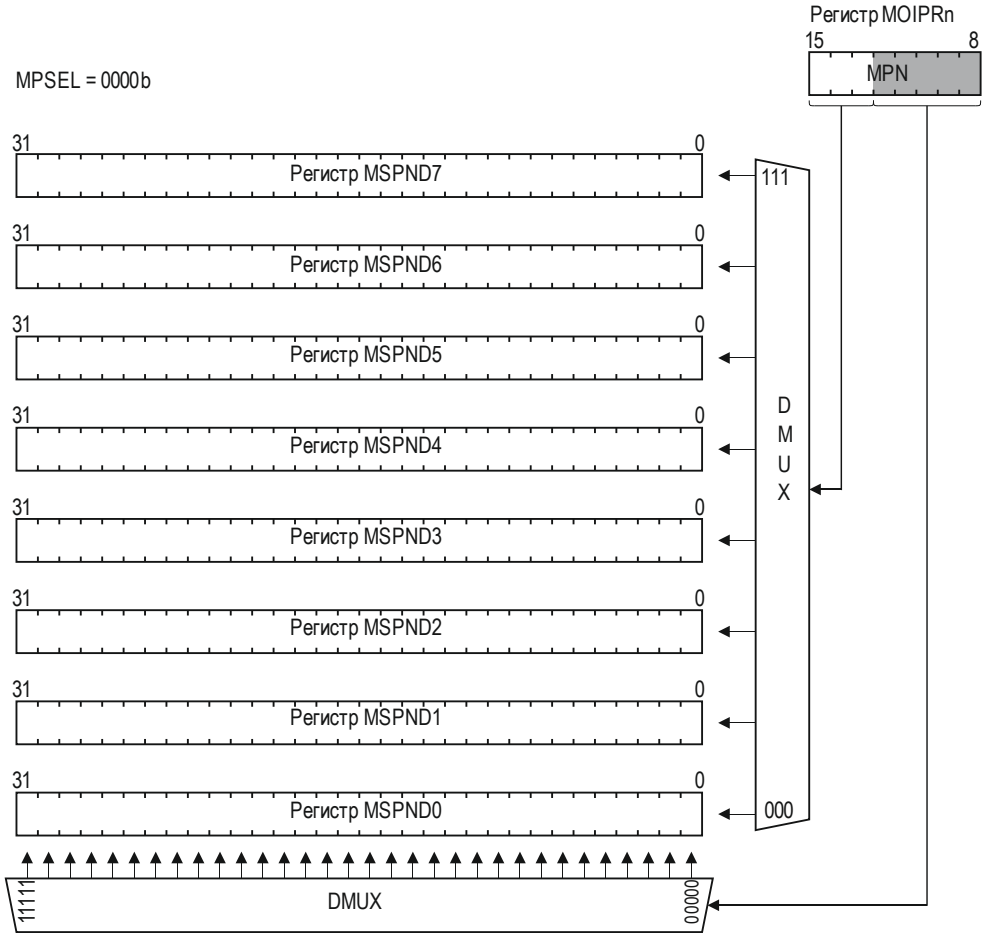


Рисунок 26.25 – Режим выбора и установки флагов при MPSEL = 0h

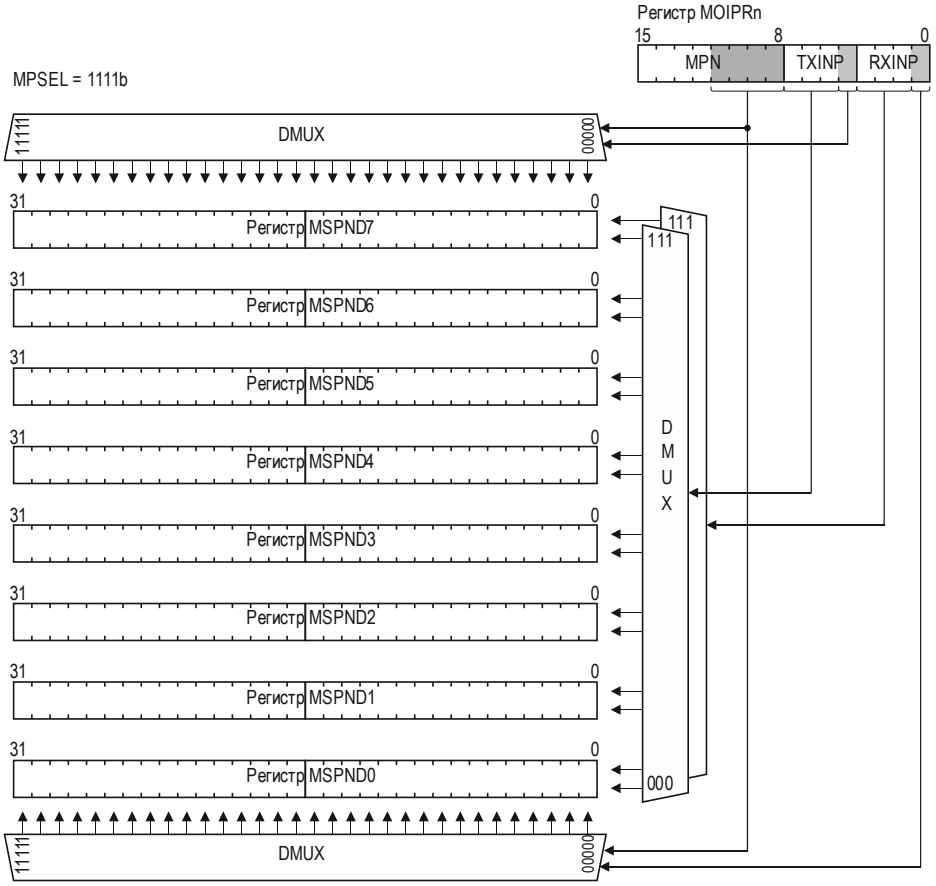


Рисунок 26.26 – Режим выбора и установки флагов при MPSEL = Fh

Во втором режиме при определении позиции флага ждущего сообщения принимаются в расчет значения поля MPN, полей RXINP (для приема) и TXINP (для передачи). При этом для флагов могут использоваться любые биты выбранного регистра MSPNDx. Установка флага ждущего сообщения происходит следующим образом:

- 3, 2 и 1 биты полей TXINP/RXINP выбирают регистр MSPNDx, в котором будет установлен флаг по окончанию передачи/приема сообщения;

- четыре младших бита поля MPN (на рисунке 26.26 выделены серым цветом) совместно с нулевыми битами полей TXINP и RXINP выбирают позицию флага (от 0 до 31). Фактически нулевой бит полей TXINP/RXINP выбирает старшее или младшее слово выбранного регистра MSPNDx, а четыре бита поля MPN задают позицию в выбранном слове.

Регистры MSPNDx могут быть записаны программно. Биты, в которые записываются единицы, остаются без изменений, а биты, в которые записываются нули, очищаются. Такой механизм записи позволяет избежать конфликта между одновременной аппаратной установкой и программной очисткой битов регистра.

Каждый регистр MSPNDx связан с соответствующим регистром индекса сообщения MSIDx, который отражает позицию самого младшего бита из всех установленных в регистре MSPNDx. Регистры MSIDx доступны только для чтения и обновляются незамедлительно после изменения (как аппаратного, так и программного) содержимого соответствующих регистров MSPNDx.

Регистр маски индекса сообщения MSIMASK содержит маску для регистров MSPNDx. Только незакрытые маской биты могут обслуживаться. Регистр MSIMASK используется одновременно для всех регистров MSPNDx и соответствующих им регистров MSIDx.

26.12 Программирование контроллера CAN

Для корректной работы контроллера CAN следует соблюдать порядок программирования регистров.

Для запуска контроллера CAN необходимо:

- записать регистр CLC;
- проверить, что сброшен бит DISR, регистр PANCTR = 0000_0000h, и после этого записать регистр FDR.

Далее для конфигурирования узла CAN с номером x (x от 0 до 1) выполнить:

- в регистре узла NCRx установить биты INIT и CCE, после чего регистры NBTRx и NPCRx станут доступны для записи и чтения, а регистр NECNTx – только для чтения;
- записать регистр NPCRx;
- записать регистр NIPRx;
- записать регистр NBTRx;
- записать регистр NFCRx (если необходимо);
- в регистре NCRx сбросить биты INIT и CCE, после чего регистры NBTRx и NPCRx будут не доступны для записи;
- распределить объекты сообщений в списки посредством регистра PANCTR.

Для корректной работы объектов сообщений регистры каждого из них должны быть проинициализированы. Для объектов, использование которых не предусматривается, достаточно записать ноль в бит MSGVAL регистра MOCTR.

Рекомендуемый порядок инициализации регистров объекта сообщения:

- установить бит DIR в регистре MOSTAT для передачи сообщения данных/приема удаленного запроса или сбросить бит DIR для приема сообщения данных/передачи удаленного запроса; установить биты TXEN0 и TXEN1 (для передачи) или RXEN (для приема) в регистре MOCTR;
- записать регистр MOFCR;
- записать регистр MOAR;
- записать регистр MOAMR (если необходимо);
- записать регистр MOFGPR (если будут использоваться FIFO структуры);
- записать регистр MOIPR;
- записать регистры MODATAL и MODATAH;
- установить бит MSGVAL корректности объекта сообщения в регистре MOCTR (для неиспользуемых объектов этот бит должен быть сброшен);
- для активирования передачи установить бит TXRQ регистра MOCTR.

26.13 Пример расчета скорости передачи 1Мбит/с

При частоте Fout = 48 МГц получить скорость передачи 1 Мбит/с можно, задавая напрямую значения полей регистра NBTR.

Если у нас Fin = 48 МГц, то можно оставить Fout = 48 МГц, для этого значение поля STEP регистра FDR должно быть 3FFh (см. раздел «Дробный делитель»).

Для скорости передачи 1 Мбит/с при частоте Fout = 48 МГц получаем:

Согласно стандарту ISO, минимальная длительность одного бита (Tbit), являющаяся суммой сегментов Tsync, Tseg1 и Tseg2, не должна быть менее 8 квантов времени (максимальная длительность бита – 25 квантов времени):

$$Tbit = 1 / (1\ Мбит/с) = 1\ мкс. \quad (26.11)$$

$$Tbit = Tsync + Tseg1 + Tseg2 \geq 8tq. \quad (16.12)$$

В соответствии с формулами вычисления значений сегментов и времени одного бита Tbit (формулы 26.4 и 26.5):

$$Tsync = 1 \times tq; \quad (26.13)$$

$$Tseg1 = (TSEG1 + 1) \times tq \geq 3tq. \quad (26.14)$$

$$Tseg2 = (TSEG2 + 1) \times tq \geq 2tq. \quad (26.15)$$

Опираясь на формулы 26.13, 26.14 и 26.15 подбираем значения длительности сегментов и рассчитываем Tbit:

$$Tsync = 1\ tq. \quad (26.16)$$

$$Tseg1 = 4\ tq. \quad (26.17)$$

$$Tseg2 = 3\ tq. \quad (26.18)$$

$$Tbit = Tsync + Tseg1 + Tseg2 = 8\ tq. \quad (26.19)$$

Опираясь на выражения (26.11) и (26.19), рассчитываем квант времени tq:

Tbit = 8 tq = 1 мкс;

Тогда: tq = 1 мкс / 8 = 125 нс;

Из формулы 26.4:

tq = (BRP + 1) / Fout;

найдем значение поля BRP:

$$BRP = tq * Fout - 1 = 125\ нс * 48\ МГц - 1 = 5. \quad (26.20)$$

Выводим итоговые значения полей для регистра синхронизации битов NBTR:

- TSEG1 = 3, исходя из (26.14) и (26.17);
- TSEG2 = 2, исходя из (26.15) и (26.18);
- BRP = 5 из выражения (26.20).

Следовательно, значение регистра NBTR = 2305h.

Если изменить значение поля STEP регистра FDR в соответствии с примером из описания (было - 3FFh, стало - 3FDh) при прочих значениях полей, указанных выше, мы получим скорость передачи в 3 раза медленнее (было 1 Мбит/с, стало 0,33(3) Мбит/с).

27 Контроллер интерфейса I2C

В состав микроконтроллера входит один контроллер интерфейса I2C, обеспечивающий полную поддержку двухпроводного последовательного синхронного интерфейса I2C/SMBus. Результат такой совместимости – легкое соединение со многими запоминающими устройствами и устройствами ввода-вывода, включая EEPROM, SRAM, счетчики, АЦП, ЦАП, периферийные устройства.

Функциональные возможности модуля контроллера I2C:

- совместимость с протоколами SMBus 1.1 и SMBus 2.0, ACCESS.Bus, I2C 2.1;
- поддержка скоростного/стандартного (FS) и высокоскоростного (HS) режимов;
- программирование действий мастера/ведомого;
- возможность подключения к шине нескольких ведущих устройств, т. е. поддержка режима мультимастер (MM);
- один программно задаваемый адрес;
- 7- или 10-битная адресация ведомого;
- поддержка адреса общего вызова.

Особые возможности протокола SMBus:

- отслеживание времени простоя линии SCL;
- наличие функции отслеживания ошибок в пакетах данных PEC с использованием метода расчета контрольной суммы CRC;
- поддержка адреса отклика мастера;
- поддержка полинга и контроля прерываний.

27.1 Протокол шины

Протокол I2C использует двухпроводной интерфейс для двухсторонней связи между устройствами, подключенными к шине. Двунаправленная шина состоит из двух линий: данных SDA и тактового сигнала SCL. Эти линии подключены к источнику питания через подтягивающие резисторы. Шинные формирователи любых устройств, подключаемых к шине, выполняются по схеме с открытым коллектором или открытым стоком. Устройства могут выставить только низкий уровень на соответствующей линии. Следовательно, обе линии SDA и SCL реализуют правило «монтажное И».

Протокол поддерживает режим мультимастер, в котором шина может контролироваться одним или несколькими устройствами из подключенных к шине. Каждое устройство, подключенное к шине, имеет свой адрес и может быть как приемником, так и передатчиком (некоторые только приемниками).

Операции с данными

Устройство, которое начинает передачу данных, становится мастером. Мастер генерирует тактовый сигнал SCL, а также инициирует и завершает передачу данных по шине. За один такт сигнала SCL передается один бит данных по линии SDA, см.рисунок 27.1.

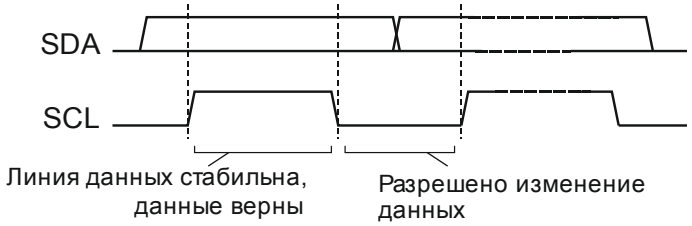


Рисунок 27.1 – Передача бита данных

Данные валидны (верны), пока уровень сигнала на линии SCL высокий. Когда на линии SCL низкий уровень сигнала, данные могут меняться.

Старт и стоп

Состояние старта формируется тогда, когда на линии SCL держится высокий уровень сигнала, а на линии SDA возникает перепад уровня сигнала из высокого уровня в низкий, см. рисунок 27.2.

Состояние стопа (останова) формируется тогда, когда на линии SCL держится высокий уровень сигнала, а на линии SDA возникает перепад уровня сигнала из низкого уровня в высокий, см. рисунок 26.2.

Состояния старта и стопа формирует только мастер.

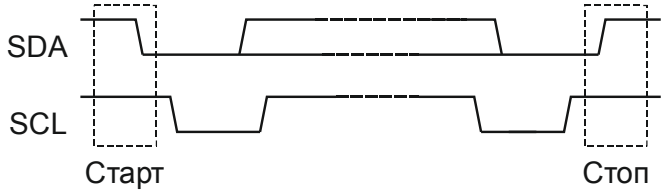


Рисунок 27.2 – Состояния старта и стопа

После того, как сформировано состояние старта, шина считается занятой и другие устройства не должны пытаться управлять ею. Шина считается занятой до тех пор, пока не будет сформировано состояние стопа. В середине передачи может быть сформировано состояние повторного старта, если мастеру нужно обратиться к другому ведомому или если требуется изменение направления передачи данных без потери контроля над шиной, см. рисунок 27.3.

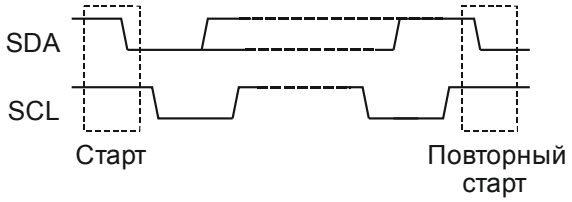


Рисунок 27.3 – Состояние повторного старта

Арбитраж

Арбитраж выполняется в момент времени, когда на линии SCL находится «1». Два устройства могут сгенерировать стартовое состояние в одно и то же время. Далее арбитраж будет продолжаться до тех пор, пока одно из устройств сформирует «0», а другое – «1» на линии SDA. Устройство, которое установило «1» на линии SDA, проигрывает арбитраж. На рисунке 27.4 приведен пример арбитража.

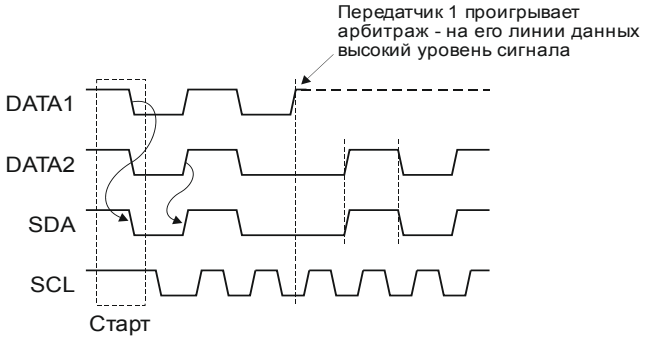


Рисунок 27.4 – Арбитраж на линии SDA

Два устройства передают свои данные DATA1 и DATA2 на линию SDA. В момент времени, когда очередной бит данных DATA1 равен «1», а бит данных DATA2 равен «0», второе устройство выигрывает арбитраж и продолжает передачу своих данных, а первое устройство прекращает передачу.

Если устройство проигрывает арбитраж во время передачи первого байта после старта (во время передачи адреса ведомого), оно становится ведомым приемником и мониторит передаваемый адрес на случай совпадения. Арбитраж также может быть проигран в режиме мастера приемника во время квитирования или в режиме ведомого передатчика во время ответа на адрес отклика на сигнал предупреждения.

В случае проигрывания арбитража в битовом поле MODE регистра ST устанавливается соответствующий код и генерируется прерывание.

Синхронизация

Синхронизация тактовых сигналов разных устройств, подключенных к шине I2C, реализуется в случаях, когда несколько устройств являются мастерами, и выполняется с использованием той особенности, что линия SCL реализована как монтажное «И» линий тактовых сигналов этих устройств. Для примера рассмотрим синхронизацию двух мастеров с линиями тактовых сигналов CLK1 и CLK2, см. рисунок 27.5.

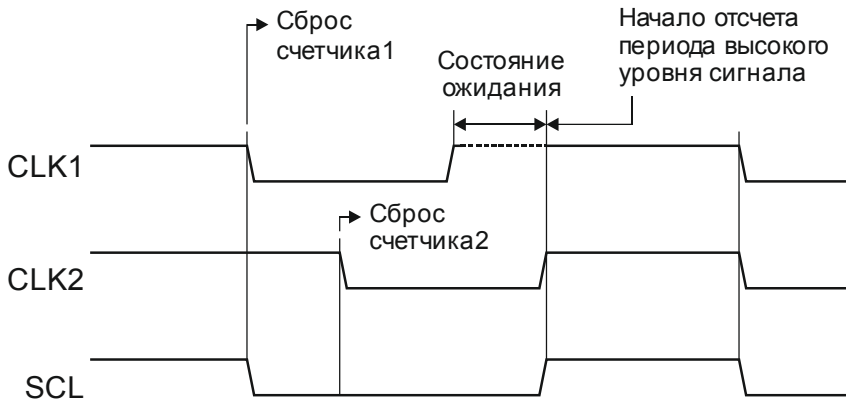


Рисунок 27.5 – Синхронизация

Линия SCL переводится в состояние «0» сразу, как только один из мастеров выставляет на своей линии тактового сигнала низкий уровень сигнала (сигнал CLK1 на рисунке 27.5). При этом его внутренний счетчик длительности низкого уровня сигнала сбрасывается и начинает отсчет. Второй мастер выставляет низкий уровень позже, и его счетчик также сбрасывается (сигнал CLK2 на рисунке 27.5).

Как только внутренний счетчик первого мастера переполнится, мастер выставит на линии CLK1 высокий уровень сигнала. Тем не менее, линия SCL будет по-прежнему оставаться в состоянии «0», удерживаемая вторым мастером. В связи с этим, первый мастер перейдет в состояние ожидания, см. рисунок 27.5. Когда переполнится счетчик второго мастера, он выставит на линии CLK2 высокий уровень сигнала, и в этот момент линия SCL перейдет в состояние «1». С этого момента внутренние счетчики длительности высокого уровня сигнала обоих мастеров начнут синхронный отсчет.

Каждая передача данных состоит из начального состояния «старт», состояний передач битов и состояния «стоп». Данные передаются старшим битом MSB вперед. Передача каждого байта завершается квитированием, т. е. приемник подтверждает окончание приема сигналом подтверждения ACK. Ведомое устройство может увеличивать паузу между тактовыми импульсами, удерживая на линии SCL сигнал низкого уровня, пока происходит

обработка принятых данных или подготовка данных для следующей передачи. Этот процесс может происходить после передачи любого бита/байта, см. рисунок 27.6.

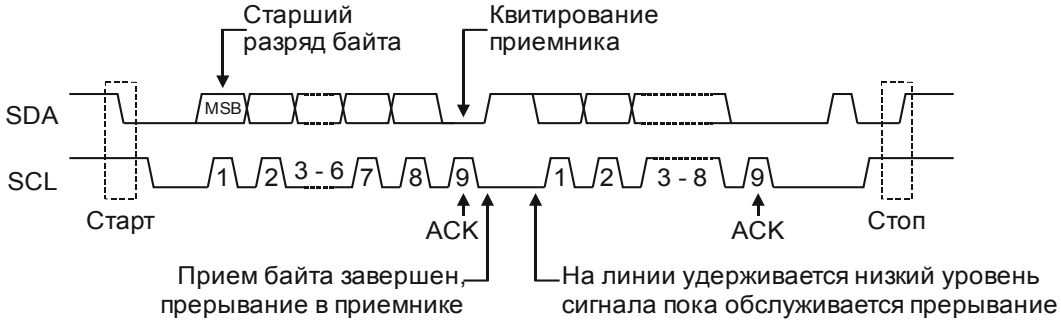


Рисунок 27.6 – Передача данных

Квитирование

Каждый байт посылки должен быть завершен квитированием, т. е. ответом на прием сигнала запроса подтверждения приема (бит ACK). На рисунках 27.6 и 27.7 показано положение момента квитирования в пределах посылки.

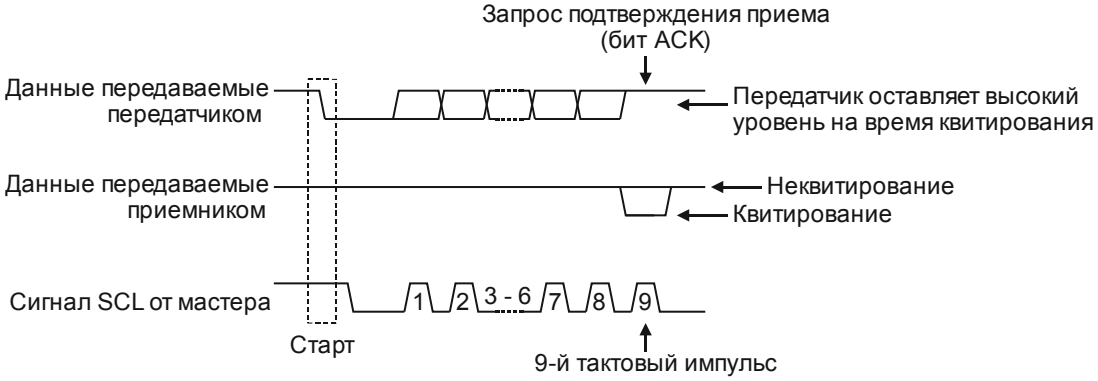


Рисунок 27.7 – Квитирование и неквитирование бита подтверждения ACK

Бит запроса подтверждения приема генерируется мастером. Передатчик (мастер или ведомый) в момент девятого такта синхросигнала оставляет линию SDA в состоянии «1» (бит ACK). В свою очередь, приемник должен сбросить линию SDA в «0» в течение времени, пока на линии SCL удерживается высокий уровень девятого импульса тактового сигнала, т. е. квитировать прием, см. рисунок 27.7. Если приемник не отвечает на запрос подтверждения и не подтверждает прием байта, то он оставляет линию SDA без изменений в состоянии «1», т. е. не квитирует прием.

Примечание – Все устройства, подсоединенные к шине I2C, в обязательном порядке должны квитировать бит ACK при получении байта с их собственным адресом. Этот механизм используется для отслеживания наличия отключившихся (самостоятельно или по каким-то причинам) от шины устройств.

Ведомое устройство имеет право не квитировать бит ACK в следующих случаях:

- если ведомый не может принять данные или он занят. Мастер, обнаружив неквитирование байта, должен сгенерировать состояние стопа и прервать передачу. Как альтернатива, ведомый может затянуть период низкого уровня сигнала тактирования на линии SCL для завершения своих операций и продолжить передачу;
- если ведомый обнаружил некорректную команду или некорректные данные. В этом случае ведомый должен неквитировать принятый байт. Мастер, обнаружив неквитирование байта, должен сгенерировать состояние стопа и повторить передачу;
- если мастер функционирует как приемник, то, приняв байт, он должен сообщить ведомому об окончании данных неквитированием бита ACK, посланного ведомым. После

этого ведомый передатчик должен освободить линию SDA для того, чтобы мастер смог сгенерировать состояние завершения передачи (состояние стопа).

Формат передачи данных с 7-битной адресацией

На рисунке 27.8 показана передача адреса и двух байт данных. Каждому устройству, подключенному к шине, присваивается уникальный 7-битный адрес. Первые семь бит, передаваемые после старта, представляют собой адрес ведомого, восьмой бит R/W# определяет направление передачи – от ведомого (чтение, если R/W# = «1») или к ведомому (запись, если R/W# = «0»).

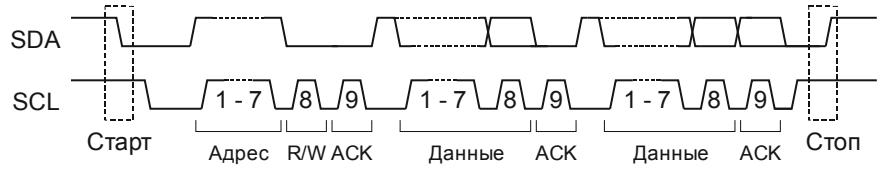


Рисунок 27.8 – Передача данных с 7-битной адресацией

Каждый ведомый, получивший байт адреса, сравнивает его со своим собственным адресом. Если адрес распознается как «свой», ведомый квитирует прием и далее, в зависимости от состояния бита R/W#, становится передатчиком или приемником.

Протокол SMBus/I2C позволяет генерировать адрес общего вызова для одновременного обращения ко всем устройствам, подключенным к шине. Первым передается адрес общего вызова (00h), затем следует байт назначения общего вызова. Ведомые, которые ожидают данные, квитируют этот байт и становятся приемниками, остальные игнорируют общий вызов.

Протокол SMBus/I2C поддерживает уникальную функцию – распознавание адреса отклика на сигнал предупреждения (Alert Response Address – ARA). В системах с несколькими ведомыми каждое устройство может послать мастеру сигнал предупреждения. Для этого используется дополнительная третья линия ALERT#, физически идентичная линиям SDA и SCL, реализованная по принципу монтажное «И». К этой линии также подключаются все устройства. Когда какому-то ведомому (или нескольким ведомым) необходимо обратиться к мастеру, он (или они) выставляет (выставляют) на линии ALERT# низкий уровень сигнала – это сигнал предупреждения, см. рисунок 27.9.

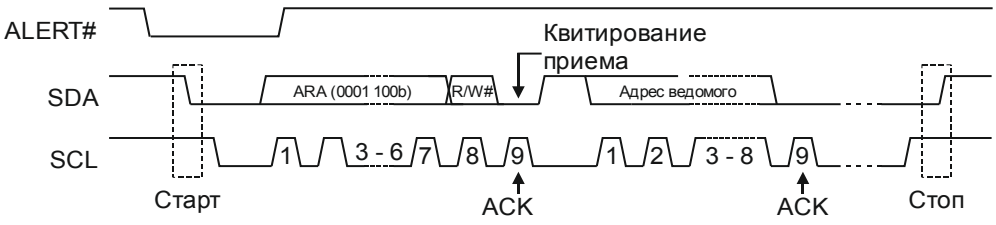


Рисунок 27.9 – Передача адреса отклика на сигнал предупреждения

Мастер, обнаружив «0» на линии ALERT#, обращается ко всем ведомым, посылая адрес отклика на сигнал предупреждения ARA. Адрес состоит из семи битов (000_1100b) и бита R/W# = «1» (чтение). Ведомый, который отправил сигнал предупреждения, получив сигнал ARA, квитирует его и затем отправляет свой 7-битный адрес (восьмой бит может быть как «0», так и «1»), сообщая, таким образом, ведомому, какое именно устройство посылало сигнал предупреждения. Кроме этого, ведомый, который выставлял «0» на линии ALERT#, должен перестать удерживать линию, чтобы на ней установился высокий уровень сигнала. В том случае, если несколько устройств посылали сигнал предупреждения, то после получения ARA, свой адрес передает то устройство, которое захватывает шину по стандартным правилам арбитража. Если после обслуживания ведомого мастер все еще

обнаруживает на линии ALERT# низкий уровень сигнала, он понимает это как то, что сигнал предупреждения посылался несколькими ведомыми. Мастер снова отправляет сигнал ARA и затем общается со следующим ведомым. Появление на линии ALERT# высокого уровня сигнала означает, что все ведомые, которые требовали обращения, обслужены.

Примечание – Описываемый в настоящем техническом описании модуль I2C не имеет выделенной линии ALERT#. При необходимости, пользователь может задействовать свободный вывод микроконтроллера и программно реализовать возможность передачи сигнала предупреждения от ведомого к мастеру. В свою очередь, функция распознавания адреса отклика ARA и последующей отправки собственного адреса реализована полностью. Включить функцию можно установкой бита SMBARE в регистре CTL0.

Формат передачи данных с 10-битной адресацией

10-битная адресация позволяет адресовать до 1024 ведомых устройств с использованием резервной комбинации 1111_0xxb, которая передается по линии SDA сразу после старта. 10-битный формат полностью совместим с 7-битным форматом и может использоваться одновременно с ним, что позволяет соединять по шине I2C устройства с разной адресацией.

Основной идеей формата является передача 10-битного адреса в двух первых байтах, следующих сразу после старта. В первом байте передается значение 1111_0xxb, где «xx» – это два старших бита адреса и бит R/W# (на рисунке 27.10 обозначен символом «W» – запись), который должен быть равен «0», чтобы ведомый понял, что в следующем байте будут переданы остальные 8 бит адреса. Во втором байте передаются 8 бит адреса, см. рисунок 27.10.

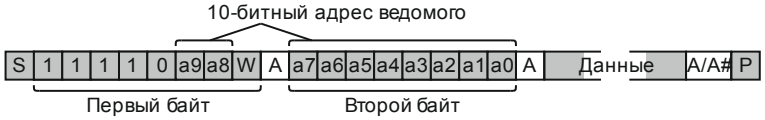


Рисунок 27.10 – Передача данных ведомому с 10-битным адресом (для расшифровки применяемых обозначений следует обратиться к таблице 27.1)

Чтобы осуществить чтение ведомого, которого адресует мастер после второго байта адреса, следует отправить бит повторного старта и затем комбинацию 1111_0xxb и бит R/W# (обозначен символом «R» – чтение), который на этот раз равен «1», см. рисунок 27.11.

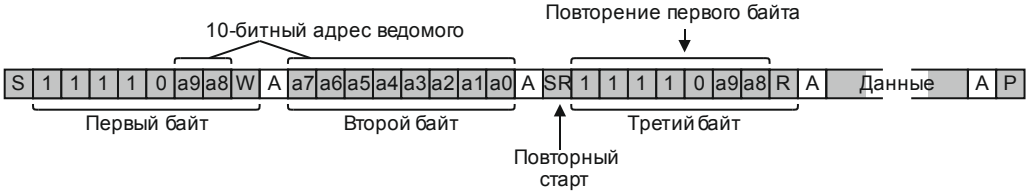


Рисунок 27.11 – Получение данных от ведомого с 10-битным адресом

На рисунках 27.10 и 27.11 биты посылки условно обозначены буквами S, W и др. или состояния битов указаны непосредственно «0» или «1». В дальнейшем на подобных рисунках, поясняющих содержимое посылки при передаче или приеме данных, будут применяться такие же и другие обозначения. Все обозначения, которые будут использоваться, указаны в таблице 27.1 с подробными пояснениями.

Таблица 27.1 – Условные обозначения, принятые на рисунках, показывающих содержимое посылок данных или адресов на линии SDA

Обозначение	Расшифровка обозначения
S	Состояние старта. Символом «S» обозначается стартовый бит посылки
SR	Состояние повторного старта
R/W	Бит указания направления передачи. В тексте настоящего описания он упоминается как R/W#. Наличие этого обозначения в бите посылки указывает на то, что этот бит может быть равен как «0», так и «1»
R	Частный случай обозначения бита направления передачи R/W#. Если в обозначении бита стоит символ «R», то это указывает на то, что в данной посылке бит R/W# должен быть равен «1», т. е. направление передачи данных происходит от ведомого к мастеру (чтение)
W	Частный случай обозначения бита направления передачи R/W#. Если в обозначении бита стоит символ «W», то это указывает на то, что в данной посылке бит R/W# должен быть равен «0», т. е. направление передачи данных происходит от мастера к ведомому (запись)
A/A#	Бит квитированого/неквитированого приема, посылаемый приемником в ответ на запрос передатчика подтвердить прием. Наличие этого обозначения в бите посылки указывает на то, что этот бит может быть равен как «0», так и «1»
A	Частный случай обозначения бита A/A#. Если в обозначении бита стоит символ «A», то это указывает на то, что в данной посылке в ответ на запрос подтверждения приема байта произошло квитирование, т. е. приемник установил линию SDA в «0». В тексте настоящего описания квитированный бит запроса подтверждения приема обозначается как ACK
A#	Частный случай обозначения бита A/A#. Если в обозначении бита стоит символ «A#», то это указывает на то, что в данной посылке в ответ на запрос подтверждения приема байта произошло неквитирование, т. е. приемник не изменил линию SDA и оставил ее в состоянии «1». В тексте настоящего описания, неквитированный бит запроса подтверждения приема обозначается как NACK
P	Состояние окончания передачи. Символом «P» обозначается стоповый бит посылки
Код мастера	8-битный код мастера. Значение 0000_1xxxb, где «xxx» – уникальный код каждого мастера в системе нескольких устройств
Адрес	7-битный адрес ведомого, передаваемый мастером
Адрес ведомого	Адрес ведомого, передаваемый во втором байте посылки. В режиме HS это 7-битный адрес, на что указывает идущий следом бит R/W#, в остальных случаях это восемь младших бит 10-битного адреса
Данные	Байт или несколько байт данных
GC	Байт адреса общего вызова (0000_0000b)
AR	Адрес отклика (0001_100b)
	Изображение числа в овале с линией, прикрепляющей его к изображению передачи битов, обозначает код операции и указывает момент, в который этот код записывается в поле MODE регистра ST
Цветное поле	Серым цветом обозначены биты, передаваемые от мастера к ведомому
Белое поле	Белым цветом обозначены биты, передаваемые от ведомого к мастеру

27.2 Функциональное описание

Структурная схема модуля I2C представлена на рисунке 27.12. Далее приводится краткое описание назначения блоков модуля.

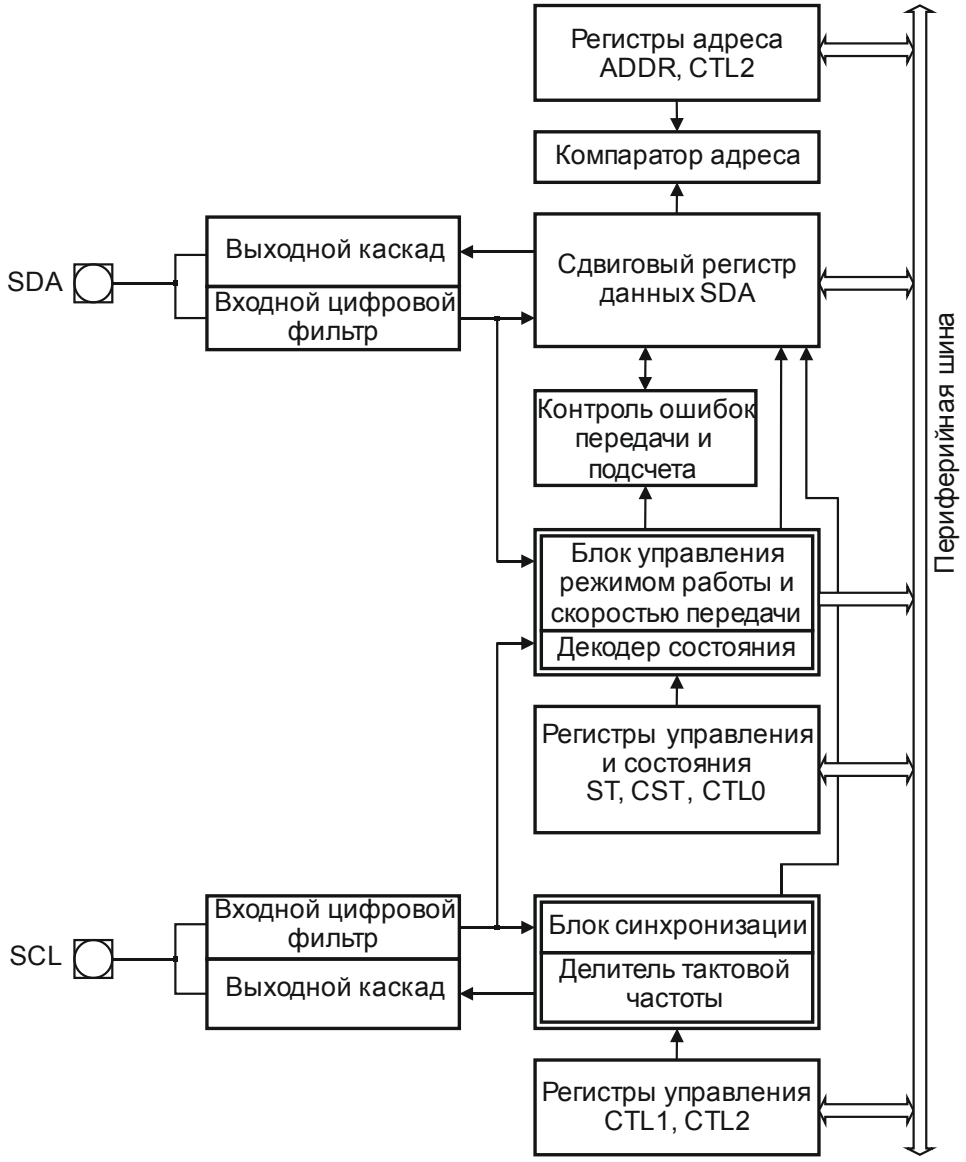


Рисунок 27.12 – Структурная схема модуля I2C

Входные и выходные каскады линий SDA и SCL

Для обеих линий используются входные шумовые фильтры. В режиме FS эти фильтры подавляют любые импульсы входного сигнала, длительность которых не превышает один такт системного синхросигнала. Выходные каскады включают в себя понижающие (до уровня «0») устройства с открытым стоком. Функционирование входных и выходных каскадов зависит от состояния модуля I2C, т. е. модуль включен или выключен.

Управление режимом работы и опрос состояния

Управление модулем I2C осуществляют блоки управления режимом работы и скоростью передачи регистров управления и состояния. В состав этих блоков входят следующие регистры:

- ST – содержит биты, отражающие текущую конфигурацию модуля I2C (мастер или ведомый, передатчик или приемник) и бит флага прерывания;

- CST – является одновременно регистром управления шиной и регистром состояния шины;

- CTL0 – управляет генерированием состояний старта, повторного старта и останова, а также квитированием;

- CTL1 и CTL2 – устанавливают параметры тактового сигнала в режиме мастера и контролируют режим 10-битной адресации.

Регистры адреса и компаратор адреса

В регистр адреса ADDR может быть записан 7-битный адрес, который является адресом устройства при работе его в режиме ведомого. Распознавание адреса включается установкой бита SAEN.

Компаратор адреса сравнивает принятый 7-битный адрес со значением, хранящимся в поле ADDR. Если разрешено распознавание адреса общего вызова (установлен бит GCMEN регистра CTL0), то компаратор сравнивает принятый адрес со значением 0000_000b. Если разрешено распознавание адреса отклика на сигнал предупреждения (установлен бит SMBARE регистра CTL0), то компаратор сравнивает принятый адрес со значением 0001_100b.

Если включен режим 10-битной адресации (одновременно установлены биты SAEN и S10EN регистров ADDR и CTL2 соответственно), компаратор сравнивает старшие пять битов первого полученного байта со значением 11110b, а следующие два бита – со значением второго и первого битов поля S10ADR регистра CTL2. Старший бит второго полученного байта сравнивается со значением нулевого бита поля S10ADR, а оставшиеся семь битов – со значением битового поля ADDR регистра ADDR, см. рисунок 27.13.

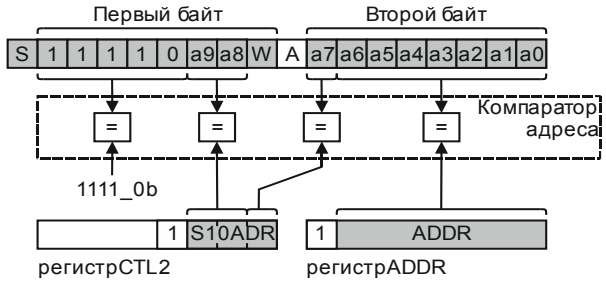


Рисунок 27.13 – Компаратор адреса в режиме 10-битной адресации

Сдвиговый регистр данных

Регистр SDA представляет собой сдвиговый регистр, используемый для приема и передачи данных. Старший бит регистра передается/принимается первым, младший бит – последним. Запись в регистр SDA возможна только, если установлен бит INT регистра ST. Регистр может быть прочитан в любой момент времени, но прочитанные данные будут гарантированно достоверными только при установленном бите INT. Регистр SDA не очищается при сбросе и хранит случайные данные до тех пор, пока не будет перезаписан программно или аппаратно после приема байта.

Генерация тактового сигнала и синхронизация

Последовательный тактовый сигнал (выходной сигнал модуля I2C в режиме мастера) формируется генератором на базе системного тактового сигнала (с частотой f_{PCLK}).

Модуль I2C может функционировать в двух глобальных режимах – стандартном/скоростном FS и высокоскоростном HS.

В режиме FS используется 15-битный предделитель. Значение младших 6 бит определяется значением битового поля SCLFRQ регистра CTL1, а нулевой бит всегда равен нулю (деление производится только на четное число). Значение старших 8 бит задается полем SCLFRQ регистра CTL3. Блок синхронизации тактового сигнала производит

синхронизацию генератора тактового сигнала и выходного сигнала SCL с тактовым сигналом других устройств, подключенных к шине.

В режиме HS используется 12-битный предделитель. Значение младших 4 бит определяется значением битового поля HSDIV регистра CTL2, нулевой бит при этом всегда равен нулю. Значение старших 8 бит задается полем HSDIV регистра CTL4.

Определяемое спецификацией протокола SMBus наименьшее время ожидания на линии SCL составляет 25 мс. Если пауза между двумя тактовыми импульсами превысила 25 мс, то устройство должно прервать текущую передачу. Мастер должен сформировать состояние старта в процессе передачи или после ее окончания. Ведомый должен освободить шину. Устройства, обнаружившие данное состояние, должны восстановить свои соединения и ожидать формирования состояния старта в пределах 10 мс.

Для отслеживания периодов ожиданий на шине в модуле I2C имеется счетчик времени ожидания. Функциональная схема счетчика показана на рисунке 27.14.

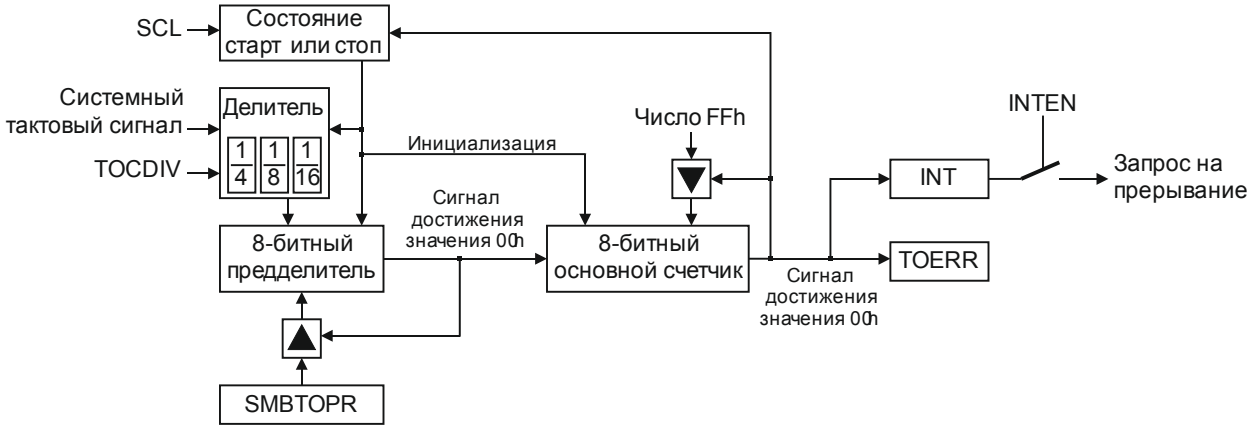


Рисунок 27.14 – Функциональная схема счетчика времени ожидания

Счетчик времени ожидания состоит из делителя, 8-битного программируемого предделителя и 8-битного основного счетчика. Все элементы счетчика времени ожидания начинают работу по отрицательному фронту сигнала на линии SCL (если работа счетчика разрешена). Положительный фронт сигнала на линии SCL сбрасывает значения делителя, предделителя и основного счетчика. Предделитель считает вниз, начиная со значения, записанного в регистр TOPR. После достижения нуля счетчиком предделителя, он загружается значением из регистра TOPR. Основной счетчик считает вниз от значения FFh. Каждое достижение нуля предделителем декрементирует значение основного счетчика. Обнуление основного счетчика и загрузка его значением FFh вызывает остановку основного счетчика предделителя и делителя и установку флага TOERR в регистре CST. Дополнительно устанавливается флаг INT, и если разрешено, генерируется прерывание.

Период времени ожидания определяется следующим выражением

$$T_{ожид} = T_{PCLK} \times TOCDIV \times (SMBTOPR + 1) \times 256 , \qquad (27.1)$$

где T_{PCLK} – период системного тактового сигнала с частотой f_{PCLK}.

Арбитраж и обнаружение ошибок на шине I2C

Арбитраж в режиме мастера передатчика может быть потерян в случае, когда два мастера одновременно формируют состояние старта и начинают передачу данных. Потеря арбитража может происходить как во время передачи адреса, так и во время передачи данных.

В случае потери приоритета при передаче байта адреса, мастер переходит в режим ведомого приемника и начинает принимать адрес. Если принятый адрес оказался «своим»,

модуль I2C далее функционирует в режиме ведомого. Если принятый адрес не оказался «своим», то модуль I2C переходит в режим безадресного ведомого.

В случае потери приоритета при передаче байта данных модуль I2C сразу переходит в режим безадресного ведомого.

Обнаружение и исправление ошибок на шине I2C

Состояние ошибки на шине I2C возникает в том случае, если во время передачи адреса/данных или во время квитирования на шине I2C обнаруживаются состояния старта или стопа. При обнаружении ошибки на шине I2C выполняются действия:

- в поле MODE регистра ST записывается код ошибки 1Fh;
- генерируется прерывание (если разрешено);
- модуль I2C переходит в режим безадресного ведомого;
- линии SDA и SCL освобождаются.

Обнаружение ошибки на шине I2C может вызвать у простой шины некорректное формирование состояния старта и отключение модуля I2C. Поэтому для возврата к нормальной работе следует выполнить действия:

- выключить и снова включить модуль I2C (бит ENABLE в регистре CTL1);
- в течение времени простоя проверить, не подключен ли другой активный мастер к шине I2C (бит BB регистра CST должен быть обнулен);
- в режиме мастера сформировать состояние старта, передать адрес и затем сформировать состояние останова, таким образом, проведя синхронизацию всех ведомых устройств (в том числе и тех, которые не обнаружили ошибку на шине I2C).

Режим IDLE

Переход в режим IDLE происходит при отключении внешнего сигнала тактирования модуля I2C записью нуля в бит I2CEN регистра APB_CLK. Переход в режим IDLE подобен программному выключению модуля I2C (очистка бита ENABLE в регистре CTL1). Регистры CTL0, ST и CST очищаются, чтобы гарантировать нормальный старт после возобновления функционирования модуля.

Выход из режима IDLE осуществляется записью единицы в бит I2CEN и включением модуля I2C битом ENABLE.

27.3 Инициализация и функционирование

В целом модуль I2C поддерживает два базовых режима – режим FS и режим HS.

Стандартный/скоростной режим или режим FS – стандартный режим работы, в котором модуль функционирует по умолчанию. Диапазон частот сигнала на линии SCL – от 763 Гц до 6,25 МГц (при f_{PCLK} = 100 МГц).

Высокоскоростной режим или режим HS – режим работы, который включается программно. Режим HS значительно превосходит режим FS по скорости – диапазон частот сигнала на линии SCL от 8,14 кГц до 16,67 МГц (при f_{PCLK} = 100 МГц).

Все операции режима HS начинаются в режиме FS в следующем порядке:

- стартовое состояние;
- 8-битный код мастера (значение 0000_1xxxb, где «xxx» – уникальный код каждого мастера в системе нескольких устройств);
- неквитирование.

Арбитраж на шине I2C происходит в момент передачи несколькими мастерами своих уникальных кодов. Выигравший арбитраж мастер захватывает шину. В связи с такой организацией режима HS дальнейший арбитраж и синхронизация на шине не реализуются.

После выполнения вышеуказанных шагов устройства, поддерживающие режим HS, переключаются в этот режим. Мастер генерирует состояние повторного старта SR, а затем передает адрес ведомого и бит направления передачи R/W#, см. рисунок 27.15. Расшифровка обозначений, принятых на рисунке 27.15, приведена в таблице 27.1.

Все передачи в режиме HS по формату идентичны передачам режима FS, что делает эти два режима полностью совместимыми.

Рисунок 27.15 – Переход в режим HS и обратно в режим FS

Выход из режима HS происходит генерированием состояния окончания передачи P, после которого все устройства переключаются обратно в режим FS. В каждом из двух базовых режимов – FS и HS – модуль I2C может функционировать как мастер или ведомый, получать или передавать информацию.

Далее все режимы работы модуля I2C будут рассмотрены подробно.

Инициализация

Для начала работы следует произвести инициализацию:

1 Включить модуль I2C установкой бита ENABLE в регистре CTL1.

2 Если активен режим мастера, записать нужный коэффициент деления в битовое поле SCLFRQ (регистры CTL1, CTL3) для выбора периода тактового сигнала SCL (для режима HS записать коэффициент деления в поле HSDIV регистров CTL2, CTL4).

3 Если активен режим ведомого, необходимо:

- записать «собственный» адрес ведомого в битовое поле ADDR и установить бит SAEN регистра ADDR;
- для реализации 10-битной адресации записать старшие биты адреса в битовое поле S10AD и установить бит S10EN регистра CTL2;
- для включения функции распознавания адреса общего вызова установить бит GCMEN в регистре CTL0;
- для включения функции распознавания адреса отклика установить бит SMBARE в регистре CTL0.

4 При необходимости отслеживания периодов ожидания на шине I2C записать желаемые значения в регистр TOPR и в битовое поле TOCDIV (регистр CST) для отсчета времени ожидания на линии SCL. Для автоматического отслеживания времени ожидания записать ненулевое значение в битовое поле TOCDIV регистра CST.

5 Для разрешения формирования запроса на прерывание установить бит INTEN в регистре CTL0.

Функционирование

Модуль I2C может работать в режиме мастера или ведомого. Также он может функционировать как передатчик или приемник, т. е. модуль I2C поддерживает девять режимов:

- безадресный ведомый;
- мастер передатчик в режиме FS;
- мастер передатчик в режиме HS;
- мастер приемник в режиме FS;
- мастер приемник в режиме HS;
- ведомый передатчик в режиме FS;
- ведомый передатчик в режиме HS;
- ведомый приемник в режиме FS;
- ведомый приемник в режиме HS.

Передача информации по шине I2C состоит из последовательности различных действий (начало передачи, прием данных и др.). Каждое действие называется состоянием (состояние старта, состояние останова и др.). После того, как то или иное состояние сформировано, его код аппаратно записывается в регистр ST в битовое поле MODE и может

быть прочитано программно. В таблицах 27.2 и 27.3 приводятся все возможные состояния, их мнемонические обозначения и коды. На квитирование или неквитирование приема указывает запись «ACK» или «NACK» соответственно. Так, например, если мастер отправил байт адреса ведомому, который после получения квитировал прием, то на это будет указывать «ACK», а в поле MODE регистра ST будет записан код 04h, соответствующий состоянию с мнемоническим обозначением «MTADPA». Более подробно каждый режим работы модуля I2C будет рассмотрен далее. На рисунках 27.16 – 27.23, поясняющих работу модуля I2C в том или ином режиме, приняты обозначения, расшифровка которых приводится в таблице 27.1. Для получения дополнительной информации и понимания работы модуля I2C можно воспользоваться приложением Б.

Таблица 27.2 – Коды функционирования модуля I2C в режиме FS

Режим		Код	Мнемоника	Описание состояния на момент записи кода в поле MODE регистра ST	ACK/ NACK
Общий		00h	IDLE	IDLE, нет доступной валидной информации о статусе	–
Мастер в режиме FS	–	01h	STDONE	Сформировано состояние старта	–
		02h	RSDONE	Сформировано состояние повторного старта	–
		03h	IDLARL	Потеря арбитража, переход в режим безадресного ведомого	–
	Передача	04h	MTADPA	Отправлен адрес ведомого	ACK
		05h	MTADNA	Отправлен адрес ведомого	NACK
		06h	MTDAPA	Отправлен байт данных	ACK
		07h	MTDANA	Отправлен байт данных	NACK
	Прием	08h	MRADPA	Отправлен адрес ведомого	ACK
		09h	MRADNA	Отправлен адрес ведомого	NACK
		0Ah	MRDAPA	Принят байт данных	ACK
		0Bh	MRDANA	Принят байт данных	NACK
	–	0Ch	MTMCER	Отправлен код мастера, обнаружена ошибка	ACK
–		0Dh – 0Fh		Зарезервировано. Не использовать!	–
Ведомый в режиме FS	Прием	10h	SRADPA	Принят адрес	ACK
		11h	SRAAPA	Принят адрес после потери арбитража	ACK
		12h	SRDAPA	Принят байт данных	ACK
		13h	SRDANA	Принят байт данных	NACK
	Передача	14h	STADPA	Принят адрес	ACK
		15h	STAAPA	Принят адрес после потери арбитража	ACK
		16h	STDAPA	Отправлен байт данных	ACK
		17h	STDANA	Отправлен байт данных	NACK
	Передача адреса отклика	18h	SATADP	Принят адрес отклика на предупреждение	ACK
		19h	SATAAP	Принят адрес отклика на предупреждение после потери арбитража	ACK
		1Ah	SATDAP	Отправлены данные в ответ на получение адреса отклика	ACK
		1Bh	SATDAN	Отправлены данные в ответ на получение адреса отклика	NACK

Окончание таблицы 27.2

Режим		Код	Мнемоника	Описание состояния на момент записи кода в поле MODE регистра ST	ACK/ NACK
Ведомый в режиме FS	–	1Ch	SSTOP	Обнаружено состояние останова ведомого	–
		1Dh	SGADPA	Принят адрес общего вызова	ACK
		1Eh	SDAAPA	Принят адрес общего вызова после потери арбитража	ACK
Общий		1Fh	BERROR	Обнаружена ошибка на шине (некорректное состояние старта или останова)	–
Примечание – Диапазон значений кодов 0Dh–0Fh зарезервирован и не доступен для использования. Дополнительная информация находится в приложении Б данного ТО.					

Таблица 27.3 – Коды функционирования модуля I2C в режиме HS

Режим		Код	Мнемоника	Описание состояния на момент записи кода в поле MODE регистра ST	ACK/ NACK
Мастер в режиме HS	–	21h	HMTMCOK	Код мастера передан успешно, переход в режим HS	–
		22h	HRSDONE	Сформировано состояние повторного старта	–
		23h	HIDLARL	Потеря арбитража, переход в режим HS безадресного ведомого	–
	Передача	24h	HMTADPA	Отправлен адрес ведомого	ACK
		25h	HMTADNA	Отправлен адрес ведомого	NACK
		26h	HMTDAPA	Отправлен байт данных	ACK
		27h	HMTDANA	Отправлен байт данных	NACK
	Прием	28h	HMRADPA	Отправлен адрес ведомого	ACK
		29h	HMRADNA	Отправлен адрес ведомого	NACK
		2Ah	HMRDAPA	Принят байт данных	ACK
		2Bh	HMRDANA	Принят байт данных	NACK
Ведомый в режиме HS	Прием	30h	HSRADPA	Принят адрес	ACK
		32h	HSRDAPA	Принят байт данных	ACK
		33h	HSRDANA	Принят байт данных	NACK
	Передача	34h	HSTADPA	Принят адрес	ACK
		36h	HSTDAPA	Отправлен байт данных	ACK
		37h	HSTDANA	Отправлен байт данных	NACK
Примечание – Диапазоны значений кодов 2Ch–2Fh и 38h–3Fh, а также коды 20h, 31h, 35h зарезервированы и недоступны для использования. Дополнительная информация находится в приложении Б данного ТО.					

Режим безадресного ведомого

Режим работы по умолчанию (MODE = 00h). После включения модуль I2C начинает функционировать в режиме безадресного ведомого и непрерывно мониторит шину. При обнаружении состояния старта или повторного старта переходит в режим ведомого приемника. Для перехода в режим мастера передатчика нужно сформировать корректное состояние старта.

Переключение в режим безадресного ведомого происходит в случаях:

- стартовое состояние не было успешно сформировано, так как другое устройство удерживало на линии SCL низкий уровень сигнала;

- произошла потеря арбитража во время передачи байта данных в режиме мастера передатчика или во время передачи бита R/W# в режиме мастера приемника;
- произошла потеря арбитража во время ответа на полученный адрес отклика;
- неквитирование принятого адреса в режиме ведомого приемника (адрес не совпал со «своим» или запрещен);
- неквитирование в конце переданного байта в режиме ведомого передатчика;
- обнаружено состояние останова;
- обнаружена ошибка на шине;
- модуль I2C был сброшен;
- модуль I2C был выключен.

Режим FS мастера передатчика

Включение режима FS мастера передатчика:

1 Переход в режим мастера передатчика происходит после успешного формирования состояния старта. Первый байт, передаваемый мастером сразу после старта, состоит из адреса ведомого и бита направления.

2 В зависимости от состояния бита направления (R/W#), модуль I2C далее функционирует как мастер передатчика (если R/W# = «0») или как мастер приемника (если R/W# = «1»). Для перехода в режим HS мастер может передать код мастера (0000_1xxxb) вместо первого байта адреса.

3 Переход в режим мастера произойдет после установки бита START в регистре CTL0. Если бит BB в регистре CST сброшен, т. е. шина свободна, будет сгенерировано состояние старта. Если бит BB = 1b, то бит START останется установленным, а состояние старта будет сгенерировано по истечении времени, равного одному такту сигнала тактирования на линии SCL, после освобождения шины.

4 Как только стартовое состояние будет сгенерировано успешно, бит START сбросится, модуль I2C перейдет в состояние STDONE (в поле MODE запишется значение 01h), установится флаг INT, и линия SCL будет удерживаться в «0» до тех пор, пока флаг INT не будет сброшен. Если разрешено битом INTEN (регистр CTL0), сгенерируется прерывание.

Передача адреса и данных:

1 Пока удерживается флаг INT, программа записывает адрес ведомого и бит направления передачи в регистр данных SDA (адрес записывается в биты с седьмого по первый).

2 После записи в регистр SDA флаг INT сбрасывается программно установкой бита CLRST в регистре CTL0.

3 После сброса флага INT и по истечении времени, требуемого для установки данных, на линии SCL появляется тактовый сигнал и данные, хранящиеся в регистре SDA, начинают передаваться по линии SDA.

4 После завершения передачи байта и получения ответа на запрос подтверждения передачи (ACK), т. е. после девятого такта сигнала тактирования на линии SCL, аппаратная часть анализирует квитирование/неквитирование передачи и устанавливает соответствующий код в поле MODE.

5 Во время передачи линии SCL и SDA постоянно мониторятся с целью выявления возможных конфликтов с другими устройствами, подключенными к шине. В случае обнаружения конфликта передача прерывается и в поле MODE записывается код 11b (состояние SRAAPA – переход в режим ведомого приемника после потери арбитража) или код 03h (состояние IDLARL – переход в режим безадресного ведомого после потери арбитража).

6 Если бит направления равен единице и не обнаружено ошибок на шине, модуль I2C переходит в режим мастера приемника.

7 Если бит направления равен нулю и передача адреса ведомого завершена успешно (значение кода в поле MODE не равно 05h/1Fh), устанавливается флаг INT, указывая на то, что ожидается запись первого байта данных в регистр SDA для дальнейшей передачи, и, если разрешено, генерируется прерывание. Пока флаг INT будет оставаться установленным, линия SCL будет удерживаться в «0».

8 Байт данных записывается программно в регистр SDA и передача продолжается.

9 Если ведомый приемник не квитирует отправленный ему байт данных, в поле MODE записывается код 0Bh (состояние MRDANA). На линии SCL будет установлен низкий уровень сигнала и, если разрешено, сгенерировано прерывание.

Для отслеживания ошибок в пакетах данных применяется механизм вычисления контрольной суммы (CRC) для нескольких байт данных. В режиме мастера передатчика установка бита PECNEXT в регистре CST вызовет перенос содержимого регистра ошибок (не доступен программно) в регистр SDA и инициирует передачу байта CRC (байт контрольной суммы) ведомому. Передача байта CRC должна выполняться после передачи последнего байта данных и перед формированием состояния останова или повторного старта.

Мастер передатчика контролирует шину и может адресовать любое ведомое устройство и изменять направление передачи без потери контроля над шиной, используя возможность формирования состояния повторного старта. Для формирования состояния следует:

1 Установить бит START.

2 В режиме мастера приемника прочитать последний полученный байт из регистра SDA.

3 Сбросить флаг прерывания INT.

После этих действий будет освобождена линия SCL, сгенерировано состояние повторного старта и сгенерировано прерывание. В поле MODE будет записан код 02h (состояние RSDONE).

Модуль I2C может быть выведен из режима мастера передатчика генерированием состояния останова. Для этого необходимо:

1 Установить бит STOP в регистре CTL0.

2 В режиме мастера приемника прочитать последний полученный байт из регистра SDA.

3 Сбросить флаг INT.

Вышеуказанные действия приведут к незамедлительному формированию состояния останова и очистке бита STOP.

Состояние останова может быть сформировано только если модуль I2C функционирует как мастер и контролирует шину (в поле MODE находится любое значение кода из диапазона 01h – 0Bh).

Дополнительно можно обратиться к приложению Б данного технического описания.

На рисунке 27.16 представлено графическое пояснение к описанию режима.

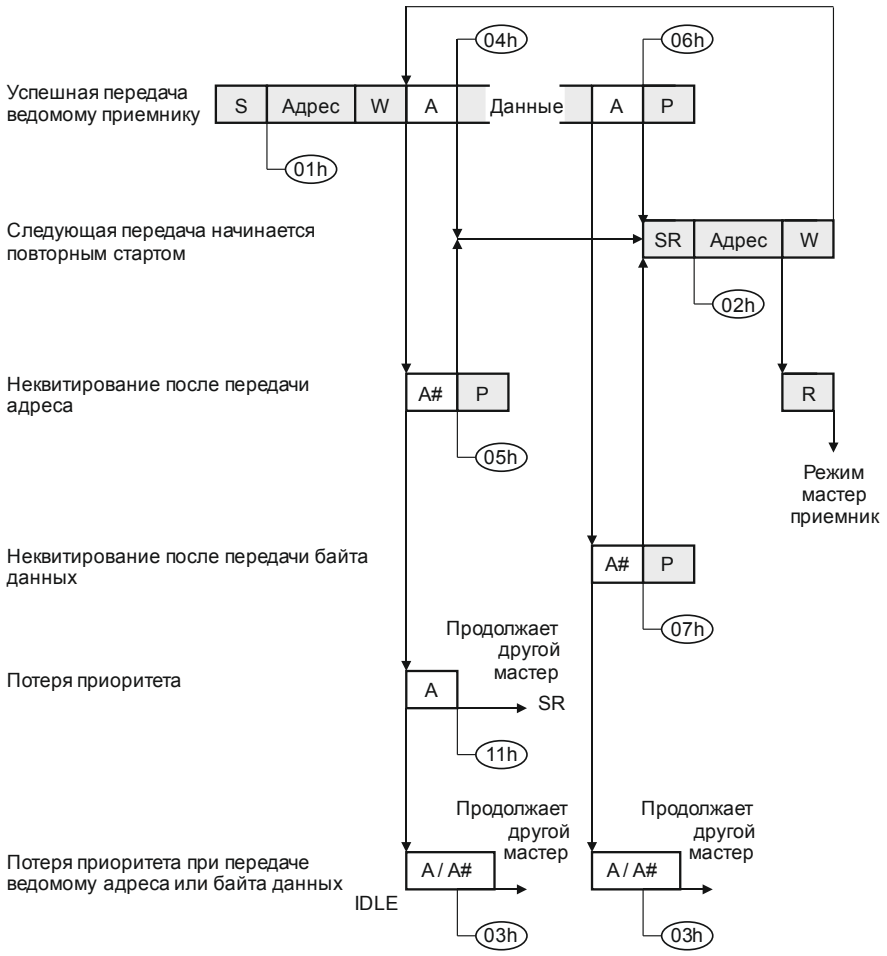


Рисунок 27.16 – Режим FS мастера передатчика

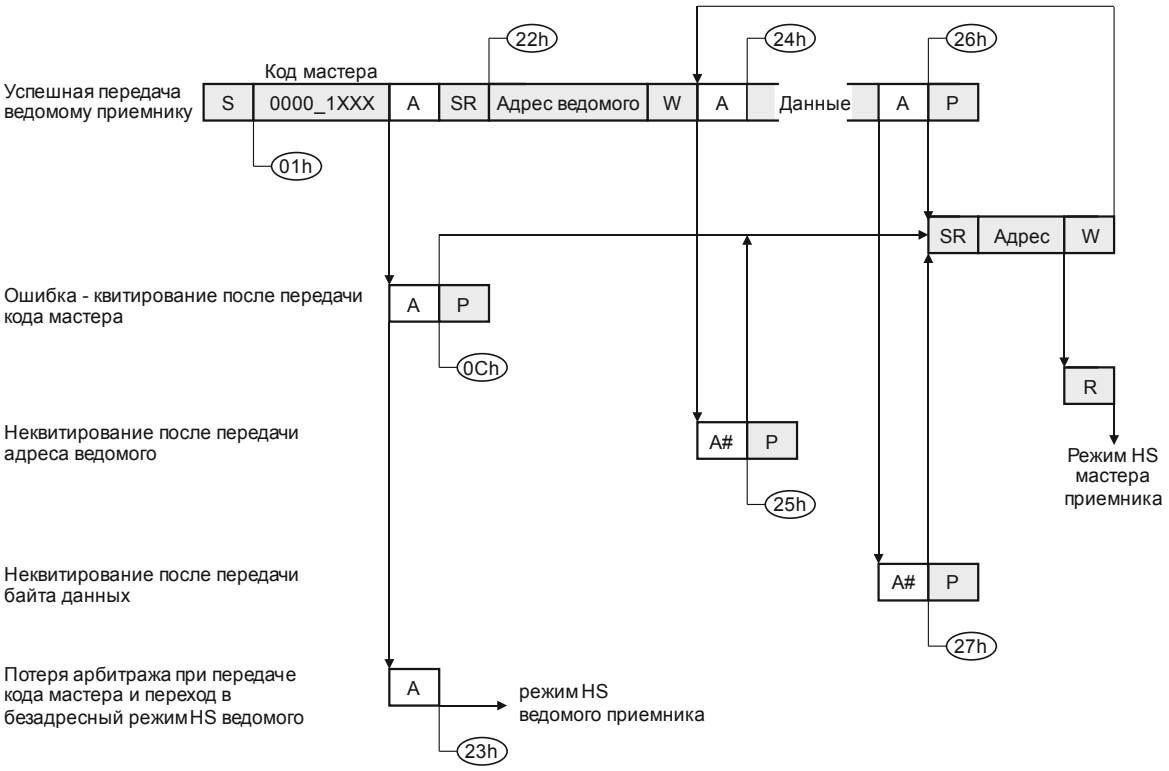


Рисунок 27.17 – Режим HS мастера передатчика

Режим HS мастера передатчика

Переход в режим HS мастера передатчика происходит в том случае, если после состояния старта мастер передает код мастера (0000_1xxxb) вместо адреса ведомого. По окончании передачи кода мастера устанавливается флаг INT и, если разрешено, генерируется прерывание. Вслед за успешной передачей кода мастера в поле MODE записывается код 21h (состояние HMTMCOK), и мастер переходит в режим HS.

Далее необходимо сформировать состояние повторного старта, записав единицу в бит START, и сбросить флаг INT, записью единицы в бит CLRST.

После сгенерированного состояния повторного старта устанавливается флаг INT, и в поле MODE записывается код 22h (состояние HRSDONE). Дальнейший порядок действий по передаче адреса и данных аналогичен описанному режиму FS мастера передатчика.

Дополнительно можно обратиться к приложению Б.

На рисунке 27.17 представлено графическое пояснение к описанию режима.

Режим FS мастера приемника

Переход в режим мастера приемника происходит после успешной передачи адреса ведомого с единичным битом направления (R/W# = «1»). В режиме мастера приемника модуль I2C получает данные от ведомого устройства, поэтому теряет контроль над шиной SDA. В тоже время мастер продолжает тактировать передачу и должен отвечать на бит ACK каждого принятого байта.

После каждого принятого байта устанавливается флаг INT, и пользовательская программа читает полученные данные из регистра SDA. Линия SCL удерживается в «0», пока установлен флаг INT. После сброса флага INT может стартовать прием следующего байта. После этого (согласно протоколу SMBus) состояния повторного старта или стопа не должны генерироваться мастером, поскольку мастер теперь не является единственным контролером линии SDA. В конце приема каждого байта мастер не квитирует прием, сообщая, таким образом, ведомому об успешном приеме.

После приема предпоследнего байта перед сбросом флага INT следует записать ноль в бит ACK регистра CTL0. В тоже время, если требуется отправка байта CRC, следует установить бит PECNEXT в регистре CST. После сброса флага INT будет принят последний байт данных и не квитирован. По окончании приема мастер возвращается в режим передатчика и теперь может сгенерировать состояние повторного старта или останова.

Если механизм отслеживания ошибок включен, то последний переданный от ведомого байт будет байтом CRC. В случае если результат вычисления контрольной суммы не нулевой, то установится флаг ошибки PECFAULT в регистре CST.

Дополнительно можно обратиться к приложению Б.

На рисунке 27.18 представлено графическое пояснение к описанию режима.

Режим HS мастера приемника

Переход в режим HS мастера приемника происходит, если после переданного кода мастера и последовавшего за ним состояния повторного старта, производится передача адреса ведомого с битом направления R/W# = «1». Модуль I2C переходит в режим HS мастера приемника, устанавливается флаг INT, а в поле MODE записывается соответствующий код из диапазона 28h – 2Bh.

Дополнительно можно обратиться к приложению Б.

На рисунке 27.19 представлено графическое пояснение к описанию режима.

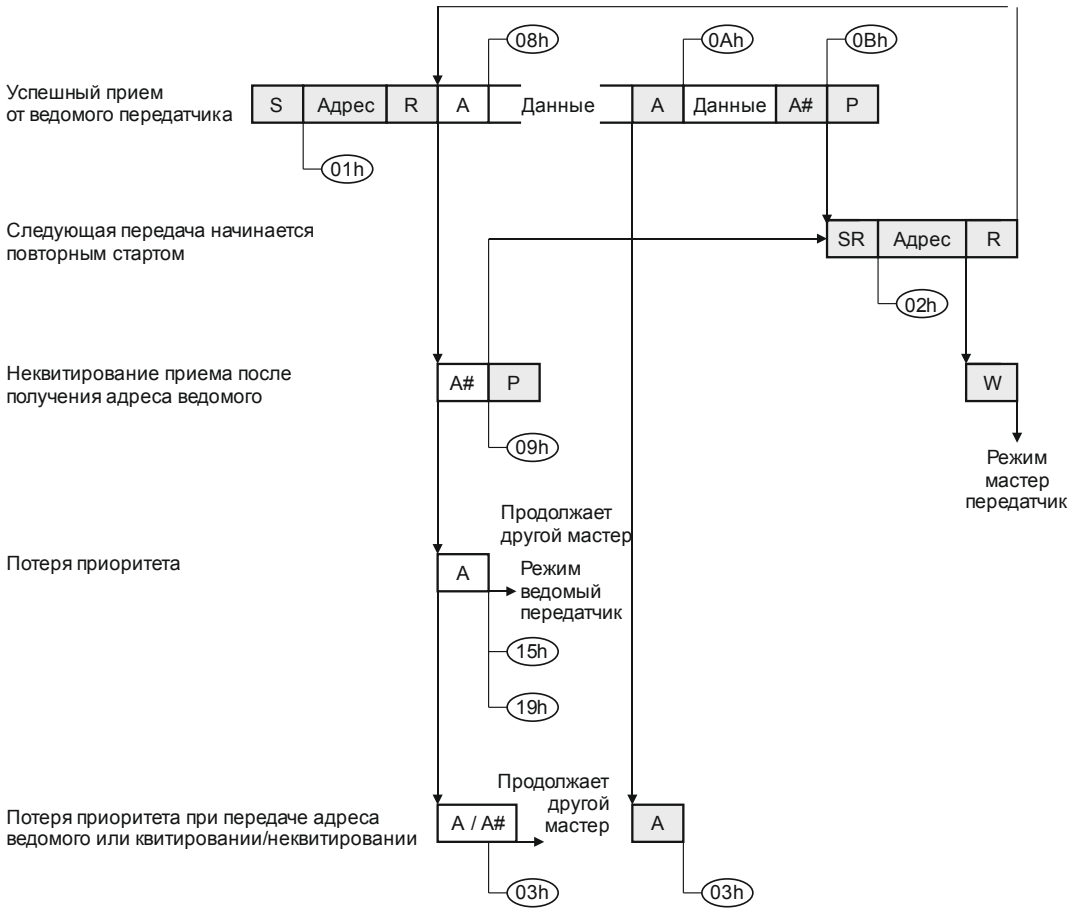


Рисунок 27.18 – Режим FS мастера приемника

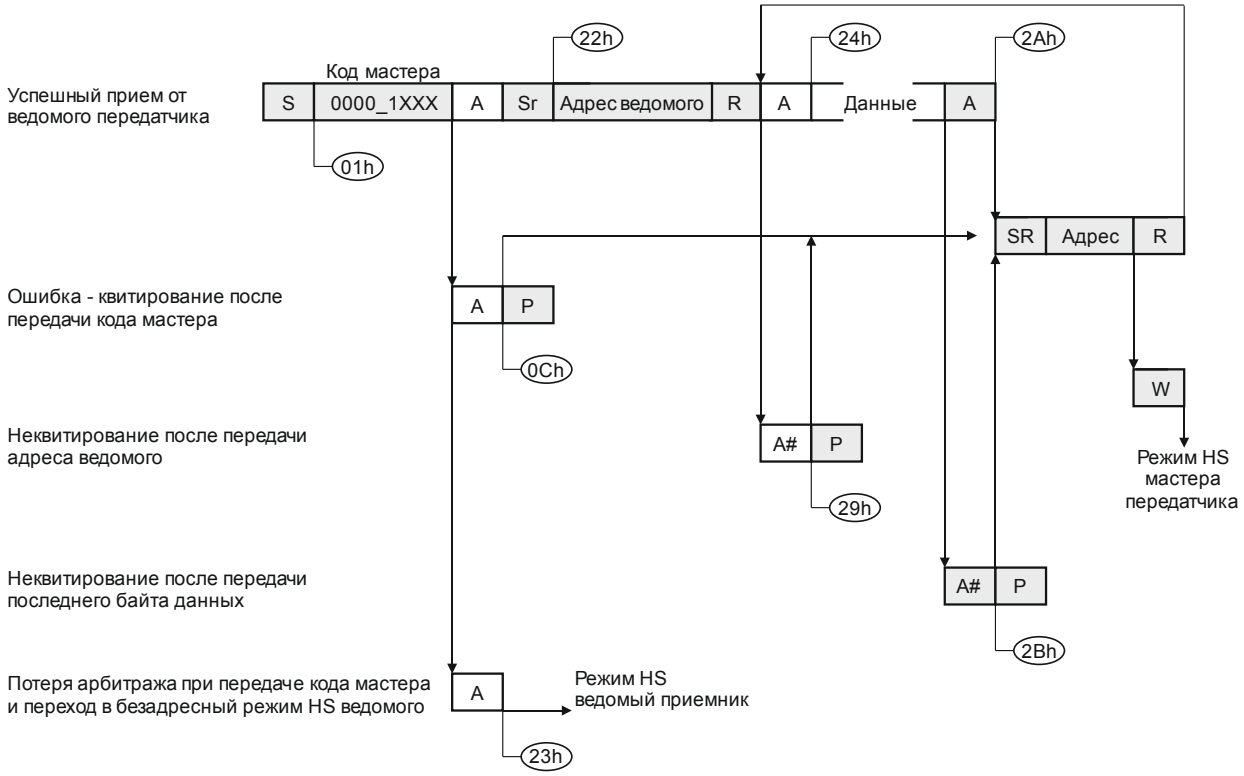


Рисунок 27.19 – Режим HS мастера приемника

Режим FS ведомого приемника

В этом режиме данные принимаются от мастера передатчика. Ведомый квитирует или не квитирует прием каждого байта.

После включения модуль I2C мониторит шину. При обнаружении состояния старта, модуль I2C переключается в режим ведомого приемника и начинает принимать семь бит адреса и бит направления передачи от мастера. Мастер передатчика может переключиться в режим ведомого приемника вследствие потери арбитража при передаче адреса.

После получения байта адреса ведомый сравнивает полученный адрес:

- полем ADDR регистра ADDR, если установлен бит SAEN;
- со значением 0000_000b (адрес общего вызова), если установлен бит GCMEN;
- со значением 0001_100b (адрес отклика), если установлен бит SMBARE.

Квитирование приема производится, если принятый адрес совпал с «собственным» (запрограммированным пользователем) адресом общего вызова или адресом отклика. После обнаружения совпадения адреса и квитирования в поле MODE записывается соответствующий код и устанавливается флаг INT. Также, если разрешено битом INTEN, генерируется прерывание. Принятый байт (адрес и бит направления) переписывается в регистр SDA.

В зависимости от состояния бита направления, модуль I2C переходит в режим ведомого передатчика (если R/W# = «1») или остается в режиме ведомого приемника (R/W# = «0»).

После каждого принятого байта устанавливается флаг INT, указывающий на то, что необходимо прочитать данные из регистра SDA, а линия SCL удерживается в «0». После программного чтения регистра SDA флаг INT сбрасывается (записью единицы в бит CLRST), и линия SCL освобождается.

Установка битов SAEN и S10EN включает режим 10-битной адресации ведомого приемника. После обнаружения состояния старта ведомый последовательно принимает два байта, в которых содержится адрес.

Последовательность передачи бит в посылке при 10-битной адресации была рассмотрена ранее в подразделе 27.1 настоящего ТО.

Механизм распознавания адреса изложен в подразделе 27.2 настоящего ТО и показан на рисунке 27.13.

После корректного приема ведомым двух байтов и совпадения принятого адреса с собственным, байты сохраняются в регистре SDA и сдвиговом регистре, прием квитируется, устанавливается флаг INT, а в поле MODE записывается соответствующий код состояния – 10h или 17h.

Если включен механизм обнаружения ошибок, последний байт, принятый от мастера передатчика, будет байтом CRC. Если результат вычисления контрольной суммы не нулевой, устанавливается флаг ошибки PECFAULT и передача не квитируется. Программа пользователя должна «знать» о количестве передаваемых мастером байт и устанавливать бит PECNEXT перед чтением предпоследнего байта из регистра SDA и потом сбрасывать флаг INT. В результате будет аппаратно рассчитана контрольная сумма, и результат отправлен мастеру в момент передачи бита ACK. Если ошибок нет, будет выполнено квитирование (отправлен «0» в ответ на запрос ACK), если ошибки есть – неквитирование (отправлена «1» в ответ на запрос ACK).

Если ведомому приемнику нужно сообщить мастеру, что он не может более принимать данные, следует сначала установить бит ACK, а затем – бит CLRST (для сброса флага INT). Далее будет принят последний байт данных, который не будет квитирован (бит ACK = 1b), и установится флаг INT. После этого программа может прочитать последний полученный байт из регистра SDA и сбросить флаг INT, после чего модуль I2C освободит шину.

Дополнительно можно обратиться к приложению Б.

На рисунке 27.20 представлено графическое пояснение к описанию режима.

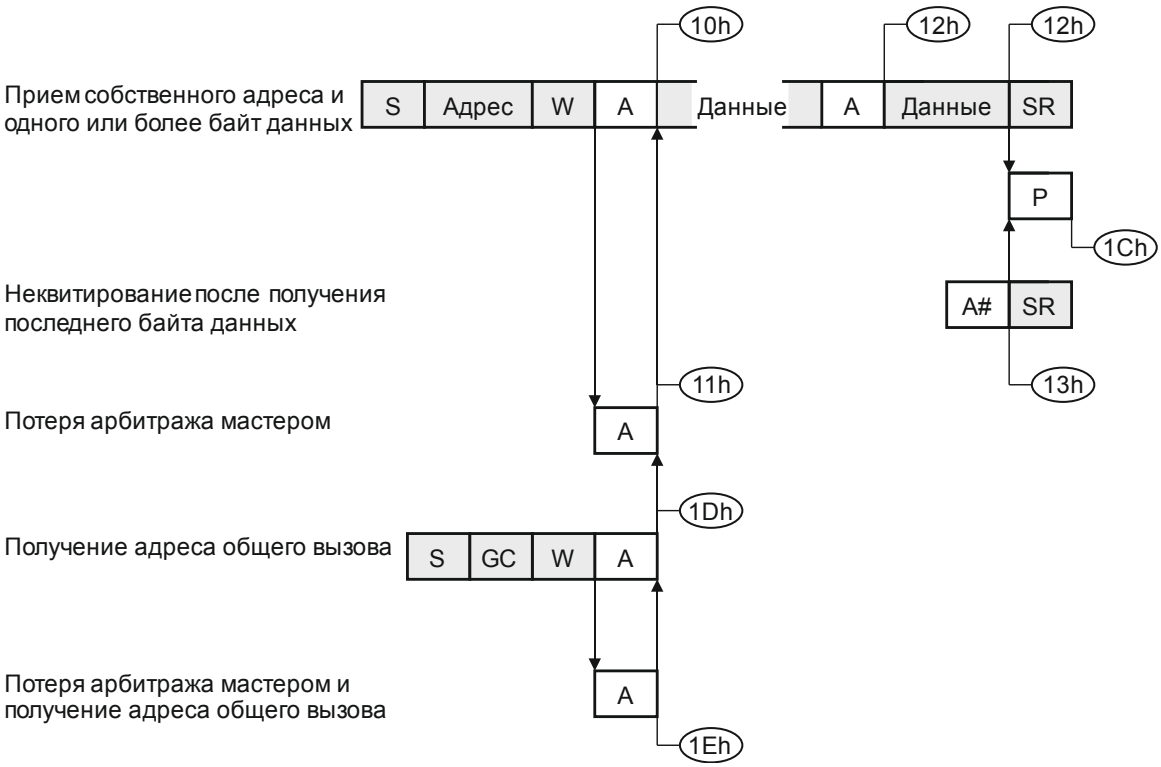


Рисунок 26.20 – Режим FS ведомого приемника

Рисунок 27.21 – Режим HS ведомого приемника

Режим HS ведомого приемника

Включение режима происходит после получения валидного кода мастера (0000_1xxxb). После передачи кода мастера формируется состояние повторного старта, а затем передается адрес ведомого с нулевым битом направления (R/W# = «0»). После получения байта адреса ведомый проверяет его на совпадение (см. ранее «Режим FS ведомого приемника»).

Дополнительно можно обратиться к приложению Б.

На рисунке 27.21 представлено графическое пояснение к описанию режима.

Режим FS ведомого передатчика

В этом режиме данные передаются от ведомого передатчика к мастеру приемника. Ведомый проверяет ответ мастера на бит ACK.

Переход в режим передатчика происходит из режима ведомого приемника. После получения собственного адреса и бита направления, равного единице (R/W# = «1»), ведомый становится передатчиком. Флаг INT устанавливается, указывая на то, что в регистр SDA следует записать данные.

Пока установлен флаг INT, линия SCL удерживается в «0». После записи данных в регистр SDA следует сбросить флаг INT. После этого, по истечении времени, необходимого для установки данных на линии SDA, линия SCL освобождается, и данные начинают передаваться.

Передача данных аналогична передаче в режиме мастера передатчика. После каждого успешного приема байта устанавливается флаг INT, а в поле MODE записывается соответствующий код. Линия SCL удерживается в состоянии «0» до тех пор, пока флаг INT остается установленным. Флаг INT должен сбрасываться только после записи данных в регистр SDA. Каждый последующий байт должен записываться в регистр SDA до тех пор, пока в поле MODE не появится код 17h (состояние STDANA), указывающий на то, что мастер закончил прием и более не ожидает новых данных.

Вывод ведомого из режима передатчика осуществляется только мастером приемника. Мастер приемника должен не квитировать последний (согласно запланированному количеству) полученный байт данных. При обнаружении неквитирования переданных данных, модуль I2C переходит в режим безадресного ведомого, и в поле MODE записывается код 00h (состояние IDLE). Далее ведомый мониторит шину в ожидании состояния старта или повторного старта.

Для работы в режиме с 10-битной адресацией следует осуществить действия аналогичные описанным для режима FS ведомого приемника.

Сначала модуль I2C переходит в режим ведомого приемника и получает 10-битный адрес. Если программно не требуется никаких действий, то флаг INT не устанавливается, линия SCL не удерживается в «0», и поле MODE содержит соответствующую информацию о состоянии. Далее (см. ранее «Формат передачи данных с 10-битной адресацией»), вслед за вторым байтом адреса, может последовать состояние повторного старта, и затем повторная передача первого байта адреса, с той лишь разницей, что бит направления содержит единицу (R/W# = «1»). Таким образом, после приема трех байт, если принятый 10-битный адрес окажется «своим», установится флаг INT, и ведомый переключится в режим передатчика. В поле MODE запишется один из двух кодов – 14h или 15h.

Если включен механизм распознавания ошибок, то последний отправленный ведомым передатчиком байт будет байтом CRC. Программа должна «знать» количество байт, посылаемых в пакете данных, и после отправки всех байт устанавливать бит PECNEXT (вместо записи очередных данных в регистр SDA) для того, чтобы в регистр SDA записался байт контрольной суммы.

В модуле I2C поддерживается функция распознавания адреса отклика, который передается мастером шины ко всем ведомым. Ведомое устройство, получившее адрес отклика (0001_100b), переключается в режим передатчика и начинает передавать свой собственный адрес (подробнее – см. «Формат передачи данных с 7-битной адресацией»).

Для включения функции распознавания адреса отклика следует установить бит SMBARE в регистре CTL0.

Модуль I2C реагирует на адрес отклика только при работе в режиме ведомого. В ответ на получение адреса отклика начать передачу адресов могут несколько ведомых. Ведомый, выигравший арбитраж, продолжает передачу, остальные – освобождают шину.

Дополнительно можно обратиться к приложению Б.

На рисунке 27.22 представлено графическое пояснение к описанию режима.

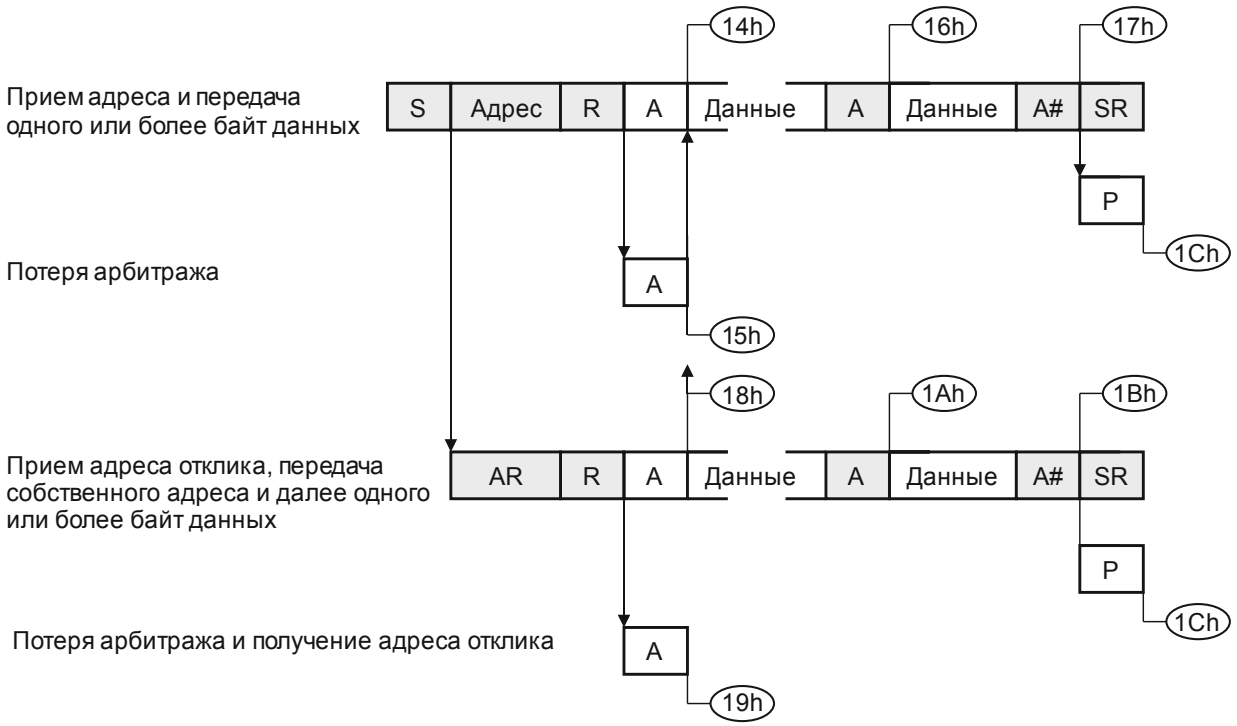


Рисунок 27.22 – Режим FS ведомого передатчика

Режим HS ведомого передатчика

Модуль I2C переходит в режим HS ведомого после получения валидного кода мастера (0000_1xxxb). Далее следует состояние повторного старта и передача адреса ведомого с единичным битом направления (R/W# = «1»). После этого ведомый переключается в режим HS ведомого передатчика. Функционирование в этом режиме в целом идентично режиму FS ведомого передатчика, с теми отличиями, что поддерживается более высокая скорость передачи, а значения кодов состояний (поле MODE) находятся в диапазоне 34h – 37h.

Дополнительно можно обратиться к приложению Б.

На рисунке 27.23 представлено графическое пояснение к описанию режима.

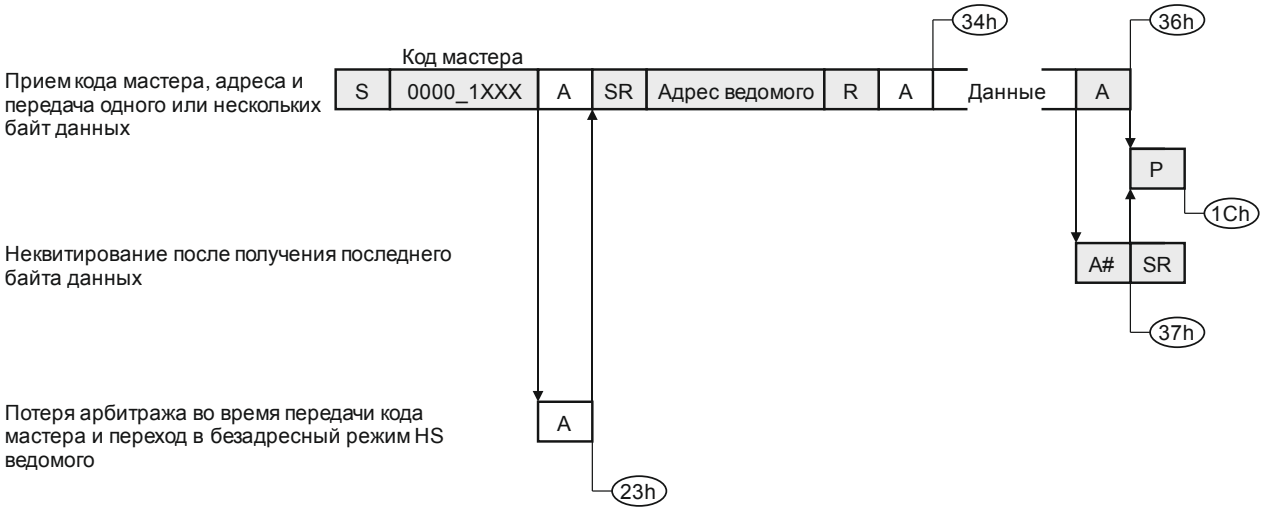


Рисунок 27.23 – Режим HS ведомого передатчика

Дополнительная информация о работе модуля

1 Когда модуль I2C выключен, бит BB регистра CST очищен. Включение модуля в системе, с более чем одним мастером, может произойти в момент времени, когда по шине идет передача. Бит BB не сможет это показать. Во избежание создания ошибок на шине модуль I2C должен синхронизироваться с сигналами на шине прежде, чем сделать попытку стать мастером. Для этого следует дождаться момента, когда на шине не будет выявлена активность, т. е. периодически проверять бит BB через периоды времени, равные периоду ожидания на шине.

2 Бит BB позволяет мониторить шину и не допускать формирования ошибочных состояний старта в процессе передачи между другими устройствами на шине.

3 В некоторых случаях шина может «зависать» при активных (с нулевым уровнем) сигналах на линиях SDA и/или SCL. Источниками таких состояний могут быть необнаруженные ошибочные стартовые или стоповые состояния, сформировавшиеся в течение приема ведомых данных. Если считать, что причиной зависания явился модуль I2C, то возможны следующие два варианта развития событий:

а) если зависла линия SCL, ничего не будет происходить, а мастер, захвативший шину, должен освободить ее;

б) если зависла линия SDA, мастер должен освободить шину. Следует помнить, что в нормальном состоянии удерживать линию SCL может только текущий мастер шины. Последовательность действий для выхода из зависания следующая (при условии, что на шине только один мастер):

- выключить и включить модуль I2C для перевода его в режим безадресного ведомого;
- установить бит START для создания состояния старта;
- проверить, удерживается ли линия SDA в «0» (активное состояние) чтением бита TSDA регистра CST. Если линия активна, отправить одиночный импульс по линии SCL, установив бит TGSCL в регистре CST;
- проверить, что в поле MODE записан код 01b (состояние STDONE), который укажет на то, что состояние старта сформировано. Если нет, то повторять предыдущий и этот шаги до тех пор, пока линия SDA не освободится.

28 Контроллер интерфейса QSPI

Контроллер интерфейса QSPI реализует интерфейс последовательной синхронной связи в режиме ведущего (мастера) устройства и обеспечивает обмен данными с подключенным ведомым периферийным устройством в соответствии с одним из двух протоколов: стандартный протокол передачи SPI и интерфейс памяти QuadSPI.

Основные характеристики контроллера QSPI:
- поддержка SDR и DDR;
- программируемый код операции для непрямого режима;
- программируемый формат кадра для непрямого режима;
- встроенный FIFO для приема и передачи;
- доступ к 8-, 16- и 32-битным данным;
- генерация прерывания по событиям:
    - порогового заполнения/опустошение буфера передатчика/приемника;
    - заполнение/опустошение буфера передатчика/приемника;
    - останов передачи;
    - завершение передачи.

28.1 Структура контроллера QSPI

Упрощенная функциональная схема контроллера QSPI с блоком синхронизации показана на рисунке 28.1.

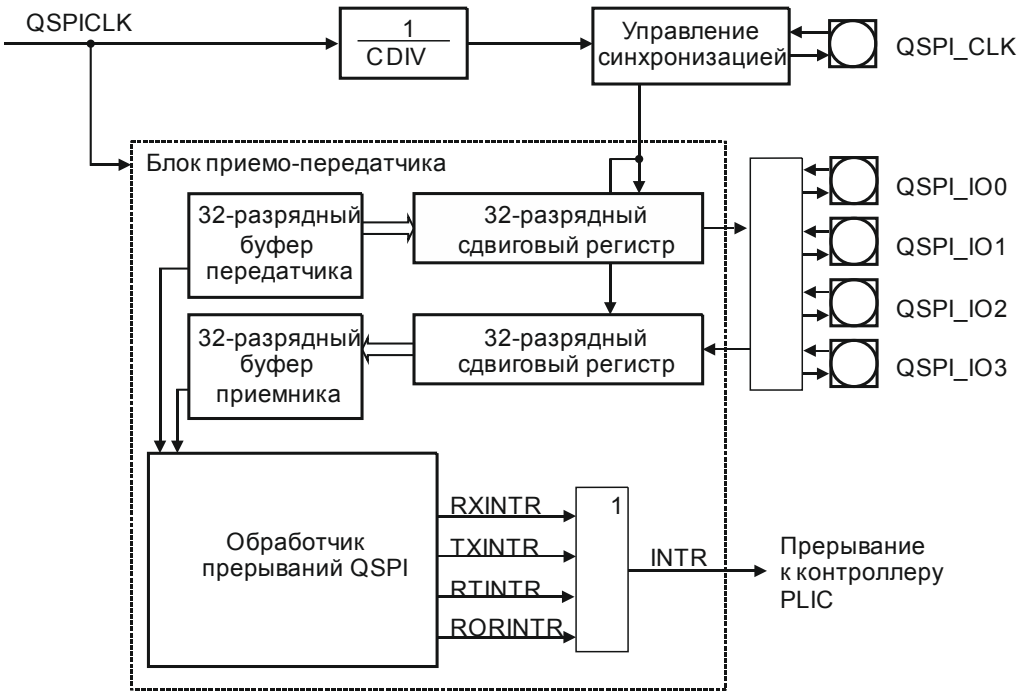


Рисунок 28.1 – Структурная схема контроллера QSPI

28.2 Режим работы SPI

QSPI может работать как стандартный SPI в режиме мастера. Выбор данного режима осуществляется с помощью битового поля FMODE регистра DCR. Схема подключения внешнего устройства приведена на рисунке 28.2.

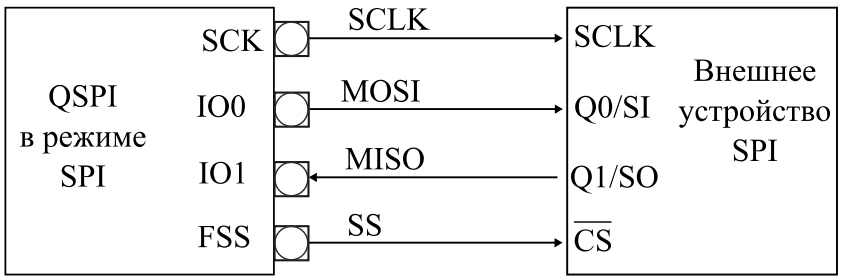


Рисунок 28.2 – Подключение внешнего устройства в режиме SPI

28.3 Последовательность команд QSPI

QSPI взаимодействует с флэш-памятью с помощью команд. Каждая команда может включать 5 фаз:

- инструкция;
- адрес;
- дополнительные данные;
- цикл ожидания;
- данные.

Любую из этих фаз можно опционально пропускать, но по крайней мере одна из фаз инструкций, адресов, альтернативный байт или фаза данных должна присутствовать.

Сигнал FSS переключается в ноль перед началом каждой команды и снова повышается после завершения каждой команды.

Фаза инструкции

На этом этапе 8-битная инструкция, записанная в поле INST регистра QCC[7:0], отправляется во флэш-память с указанием типа операции. Несмотря на то, что большинство интегральных схем флэш-памяти могут получать инструкции только по одному биту за раз от сигнала IO0 (режим одиночного SPI), фаза инструкции может опционально отправлять 2 бита за раз (через IO0/IO1 в режиме двойного SPI) или 4 бита за раз (через IO0/IO1/IO2/IO3 в режиме Quad SPI). Режим инструкции выбирается с помощью битового поля IMOD[1:0] регистра QCC[9:8]. Когда битовое поле IMOD = 00, фаза инструкции пропускается, и начинается последовательность команд с фазой адреса, если она присутствует.

Фаза адреса

В адресной фазе 1-4 байта отправляются во флэш-память, чтобы указать адрес операции. Количество отправляемых байтов адреса настраивается в поле ADSIZ[1:0] регистра QCC[13:12]. Байты адреса для отправки записываются в регистр QAD[31:0].

Фаза адреса может отправлять 1 бит за раз (через SO в режиме одиночного SPI), 2 бита за раз (через IO0/IO1 в режиме двойного SPI) или 4 бита за раз (через IO0/IO1/IO2/IO3 в режиме Quad SPI). Режим передачи адреса выбирается с помощью битового поля ADMOD[1:0] регистра QCC[11:10].

Когда ADMOD = 00, фаза адреса пропускается, и последовательность команд продолжается следующим этапом, если таковой имеется.

Фаза дополнительных данных

В фазе дополнительных данных 1-4 байта отправляются во флэш-память, как правило, для управления режимом работы. Количество дополнительных данных для отправки настраивается в битовом поле ABSIZ регистра QCC. Отправляемые дополнительные данные записываются в регистр QAB.

Фаза дополнительных данных может отправлять 1 бит за раз (через SO в режиме одиночного SPI), 2 бита за раз (через IO0/IO1 в режиме двойного SPI) или 4 бита за раз (через IO0/IO1/IO2/IO3 в режиме Quad SPI). Режим передачи адреса выбирается с помощью битового поля ABMOD регистра QCC.

Когда ABMOD = 00, фаза дополнительных данных пропускается, и последовательность команд переходит непосредственно к следующей фазе, если таковая имеется.

Фаза циклов ожидания

В фазе циклов ожидания предоставляется 0-31 цикла (периодов сигнала QSPI_CLK) без отправки или получения каких-либо данных, чтобы дать флэш-памяти время подготовиться к фазе данных в случае обмена на высоких частотах. Количество циклов ожидания задается в битовом поле DCYCS регистра QCC. В обоих режимах SDR и DDR продолжительность указывается как количество полных циклов QSPI_CLK.

Когда в битовое поле DCYCS записано нулевое значение, фаза циклов ожидания пропускается, и последовательность команд переходит непосредственно к фазе данных, если она присутствует.

Режим работы фазы циклов ожидания определяется битовым полем DMOD регистра QCC. В случае использования двойного или Quad SPI режима приема данных из флэш-памяти для обеспечения достаточного времени ожидания при переключении режима линий IO (из режима ввода в режим вывода) ведомого необходимо использовать хотя бы один цикл ожидания.

Фаза данных

Во время фазы данных любое количество байт может быть отправлено или получено из флэш-памяти.

Количество отправляемых/принимаемых слов данных указывается в регистре TDS.

Данные, которые должны быть отправлены во флэш-память, должны быть записаны в регистр TDR, данные, полученные из Flash-памяти можно получить, прочитав регистр TDR.

28.4 Режимы протокола интерфейса сигнала QSPI

Режим передачи по одной линии данных

Устаревший режим SPI позволяет последовательно отправлять/принимать только один бит. В этом режиме данные отправляются во флэш-память через сигнал SO, подключенный к выводу IO0. Данные, полученные из флэш-памяти, поступают через сигнал SI, подключенный к выводу IO1.

Каждая из фаз команды может быть сконфигурирована отдельно для использования однобитового режима записью значения 0x01 в битовые поля IMOD/ADMOD/ ABMOD/DMOD регистра QCC.

В каждой фазе, настроенной на передачу по одной линии, выводы контроллера QSPI функционируют в следующих режимах:

- IO0 (SO) находится в режиме вывода;
- IO1 (SI) находится в режиме ввода (высокий импеданс);
- IO2 находится в режиме вывода и установлен на «0»;
- IO3 находится в режиме вывода и принудительно установлен на «1» (для деактивации функции «удержания»).

Данная конфигурация выводов QSPI активна и в фазе циклов ожидания, если DMOD = 01.

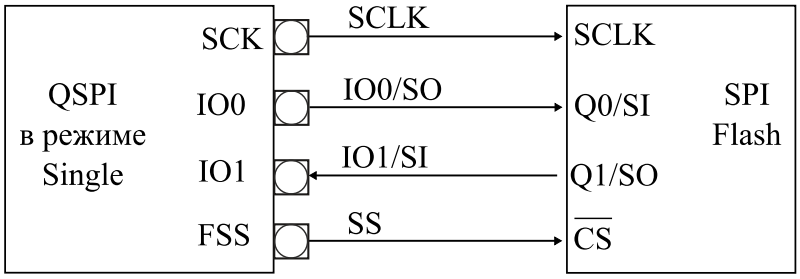


Рисунок 28.3 – Подключение внешней flash-памяти в режиме single QSPI

Режим передачи по двум линиям данных

В режиме передачи по двум линиям (DIO), по выводам IO0/IO1 одновременно отправляются/принимаются два бита.

Каждая из фаз команды настраивается отдельно для использования режима передачи по двум линиям записью значения 0x02 в битовые поля IMOD/ADMOD/ABMOD/DMOD регистра QCC.

В режиме передачи по двум линиям во всех фазах команды, выводы контроллера QSPI функционируют в следующих режимах:

- IO0/IO1 имеют высокий импеданс (вход) во время фазы данных для операций чтения, и выходы во всех остальных случаях;
- IO2 находится в режиме вывода и установлен на «0»;
- IO3 находится в режиме вывода и принудительно установлен на «1».

В фазе циклов ожидания, при записи в битовое поле DMOD значения 1, выводы IO0/IO1 всегда имеют высокий импеданс.

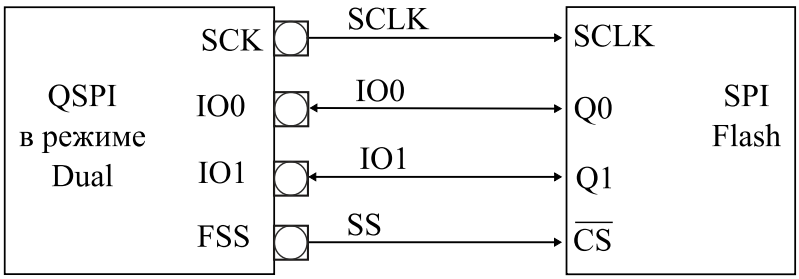


Рисунок 28.4 – Подключение внешней flash-памяти в режиме dual QSPI

Режим передачи по четырем линиям данных

В режиме передачи по четырем линиям (QIO) по выводам IO0/IO1/IO2/IO3 одновременно отправляются/принимаются четыре бита сигнала.

Каждая из фаз команды настраивается отдельно для использования режима передачи по четырем линиям записью значения 0x03 в битовые поля IMOD/ADMOD/ABMOD/DMOD регистра QSPI_QCC.

В режиме передачи по четырем линиям во всех фазах команды, выводы контроллера QSPI IO0/IO1/IO2/IO3 находятся в состоянии высокого импеданса во время фазы данных для операций чтения и в режиме выхода во всех остальных фазах команды.

В фазе циклов ожидания, при записи в битовое поле DMOD значения 3, все выводы IO0/IO1/IO2/IO3 имеют высокий импеданс.

Выводы IO2 и IO3 используются только в режиме передачи по четырем линиям. Если ни одна из фаз команды не настроена на использование режима передачи по четырем линиям, тогда выводы микроконтроллера, соответствующие выводам интерфейса IO2 и IO3, можно использовать для других функций, даже когда QSPI активен.

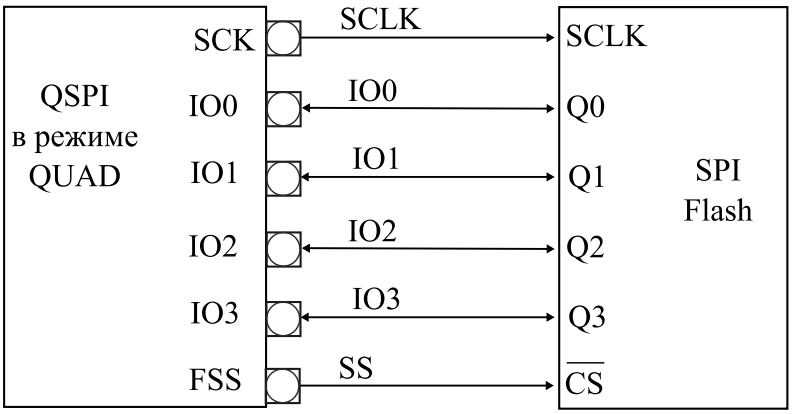


Рисунок 28.5 – Подключение внешней flash-памяти в режиме quad QSPI

Режим SDR

По умолчанию бит DDR регистра TCR равен 0, и QSPI работает в обычном режиме передачи данных (изменение данных происходит по одному фронту сигнала QSPI_CLK).

В режиме SDR, когда QSPI управляет сигналами IO0/SO, IO1, IO2, IO3, эти сигналы переключаются только по одному фронту сигнала QSPI_CLK.

При приеме данных в режиме SDR QSPI предполагает, что флэш-память также отправит данные по одному фронту QSPI_CLK.

Режим DDR

Когда бит DDR регистра TCR установлен в 1, QSPI работает в режиме двойной скорости передачи данных (DDR).

В режиме DDR, когда QSPI управляет сигналами IO0/SO, IO1, IO2, IO3 в фазах адреса/дополнительных данных/данных, биты отправляются по обоим фронтам сигнала QSPI_CLK.

Фаза инструкции не зависит от состояния бита DDR. Инструкция всегда передается по одному фронту сигнала QSPI_CLK. При приеме данных в режиме DDR контроллер QSPI предполагает, что флэш-память также отправляет данные по обоим фронтам сигнала QSPI_CLK.

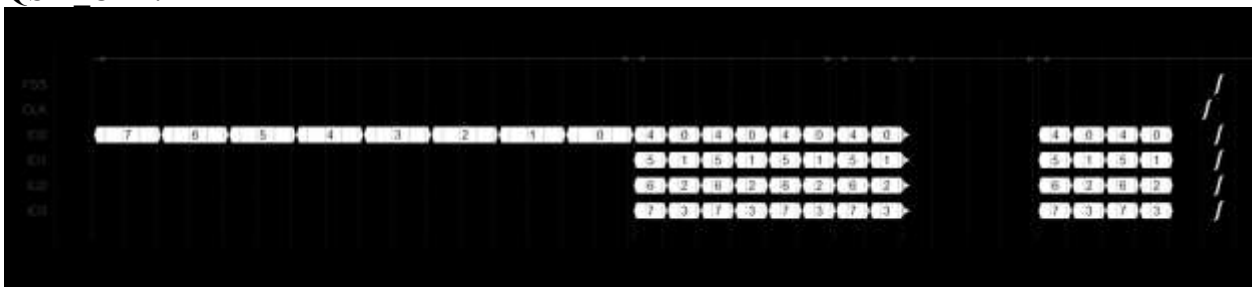


Рисунок 28.6 – Пример передачи команды DDR в режиме передачи данных по четырем линиям

28.5 Программирование QSPI

Режим работы выбирается с помощью FMOD регистра DCR.

Для выбора стандартного режима работы в битовое поле FMOD необходимо записать значение 0, а для выбора режима работы QSPI – значение 2.

Порядок действий при настройке работы в стандартном режиме SPI:

1 Указать значение делителя, полярность и фазу сигнала в регистре DCR.

2 Разрешить прием/передачу данных, указать размер слова данных и порядок передачи битов в слове данных (MSB/LSB) регистре TCR.

3 Чтение/запись данных из/в FIFO осуществляется с помощью регистра TDR.

Порядок действий при настройке работы в режиме QSPI:

1 Указать количество слов данных для чтения или записи в регистре TDS (размер слова данных задается битовым полем LEN регистра TCR).

2 Указать значение делителя, полярность и фазу сигнала в регистре DCR, а также режим и код инструкции в регистре QCC.

3 Указать необязательные дополнительные данные, которые будут отправлены сразу после фазы адреса в регистре QAB.

4 Указать целевой адрес в регистре QAD.

5 Чтение/запись данных из/в FIFO осуществляется с помощью регистра TDR.

При записи управляющих регистров TCR, FWM, IMR необходимо указать следующие настройки:

1 Установить бит разрешения работы DEN в регистре HCR и биты RXE, TXE в регистре TCR;

2 Указать пороговый уровень FIFO записав значения в битовые поля RHEC и TLEC регистра FWM.

3 Разрешить прерывания в регистре IMR

При записи регистра конфигурации обмена QCC необходимо указать следующие параметры:

- байт инструкции, записав в битовое поле INST;
- режим инструкции, записав в битовое поле IMOD;
- режим передачи адреса, записав в битовое поле ADMOD;
- размер адреса, записав в битовое поле ADSIZ;
- режим передачи дополнительных данных, записав в битовое поле ABMOD;
- количество байт дополнительных данных, записав в битовое поле ABSIZ;
- количество циклов ожидания, записав в битовое поле DCYCS.
- режим отправки/получения данных, записав в битовое поле DMOD.

Если ни регистр адреса (QAD), ни регистр данных (TDR) не требуется обновлять для конкретной команды, то последовательность команд запускается, как только записывается регистр QCC. Это происходит только в том случае, когда в битовые поля ADMOD и DMOD записаны нулевые значения, т.е. пропускаются фазы адреса и данных.

Когда при передаче необходимо передать только фазу адреса (в битовое поле ADMOD записано ненулевое значение) и при этом пропустить фазу данных (в битовое поле DMOD записано значение 0), то передача последовательности команд начинается сразу после записи значения адреса в регистр QAD.

28.6 Бит занятости QSPI и функция прерывания

Как только QSPI начинает операцию с флэш-памятью, бит BUSY автоматически устанавливается в DSR.

Бит BUSY сбрасывается после того, как QSPI завершит запрошенную последовательность команд и FIFO опустеет.

Любую операцию можно приостановить, установив бит DRS в регистре HCR. Когда передача приостановлена, биты BUSY и TST автоматически сбрасываются. Для очистки буферов FIFO по приему/передаче необходимо установить биты RXCLR и TXCLR регистра HCR.

28.7 Поведение сигнала FSS

По умолчанию сигнал FSS имеет высокий уровень, что означает неактивное состояние выбора внешней флэш-памяти. Сигнал FSS сбрасывается перед началом операции и устанавливается после ее завершения.

28.8 Прерывания QSPI

Прерывание может быть вызвано по следующим событиям:
- пороговое переполнение буфера приемника;
- пороговое опустошение буфера передатчика;
- полное заполнение буфера приемника;
- полное опустошение буфера приемника;
- полное заполнение буфера передатчика;
- полное опустошение буфера передатчика;
- останов передачи;
- завершение передачи.

29 Контроллер интерфейса SPI

В состав микроконтроллера входят два идентичных контроллера интерфейса SPI, реализующих интерфейс последовательной синхронной связи в режиме ведущего (мастера) и ведомого устройства и обеспечивает обмен данными с подключенным ведомым или ведущим периферийным устройством в соответствии с одним из трех протоколов фирм Motorola, National Semiconductor, Texas Instruments.

Последовательный синхронный интерфейс SPI фирмы Motorola обеспечивает полнодуплексный обмен данными по четырехпроводной линии и программное задание фазы и полярности тактового сигнала.

Интерфейс Microwire фирмы National Semiconductor обеспечивает полудуплексный обмен данными с использованием 8-битных управляющих последовательностей.

Интерфейс SSI фирмы Texas Instruments обеспечивает полнодуплексный обмен данными по четырехпроводной линии и возможность перевода линии передачи данных в третье (высокоимпедансное) состояние.

Выбор интерфейса осуществляется посредством поля FRF регистра CR0.

В режиме мастера и в режиме ведомого устройства контроллер SPI обеспечивает:

- передачу данных, размещенных в буфере передатчика (восемь 16-разрядных ячеек);
- прием данных и размещение их в буфере приемника (восемь 16-разрядных ячеек).

Контроллер SPI формирует сигналы прерываний по следующим событиям:

- необходимость обслуживания буферов приемника и/или передатчика;
- переполнение буфера приемника;
- наличие данных в буфере приемника по истечении времени таймаута.

Основные характеристики контроллера SPI:

- программное управление скоростью обмена;
- программируемая длительность информационного кадра от 4 до 16 бит;
- независимое маскирование прерываний от буферов передатчика и приемника;
- поддержка прямого доступа к памяти (блок DMA).

29.1 Структура контроллера SPI

Упрощенная функциональная схема контроллера SPI с блоком синхронизации показана на рисунке 29.1.

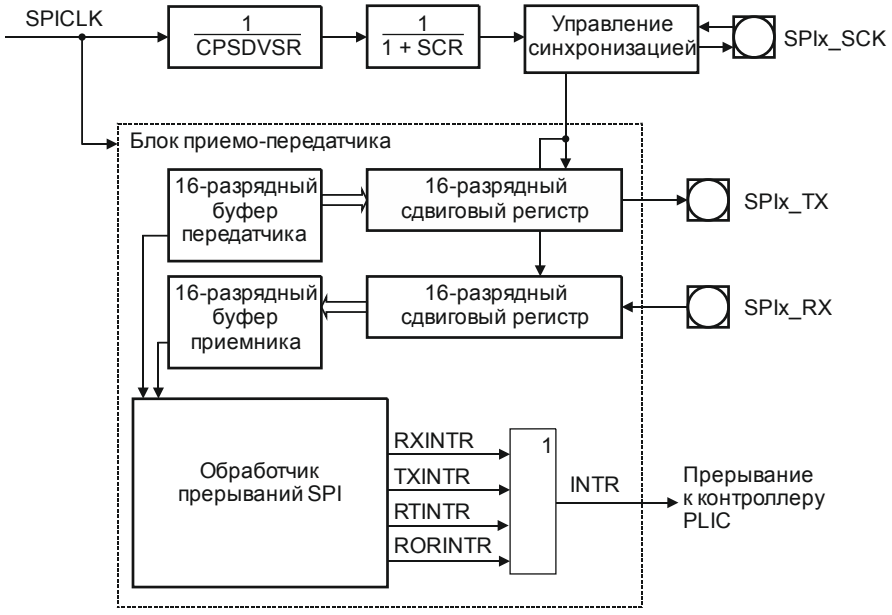


Рисунок 29.1 – Упрощенная функциональная схема контроллера SPI

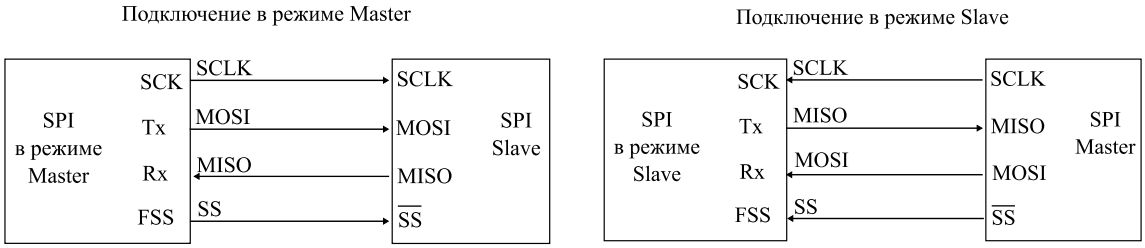


Рисунок 29.2 – Примеры схем включения контроллера SPI в режимах Master и Slave

Синхронизация

Тактирование контроллера SPI осуществляется тактовым сигналом SPICLK, который формируется на основе одного из нескольких базовых синхросигналов (см. раздел 4).

Существует ограничение на соотношение между частотами тактовых сигналов SYSCLK и SPICLK:

$$f_{SPICLK} \le f_{SYSCLK}. \quad (29.1)$$

В режиме мастера на основе сигнала SPICLK посредством двух последовательно стоящих делителей формируется сигнал тактирования передачи и приема данных с частотой f_{SPIx_SCK}, которую можно вычислить по формуле

$$f_{SPIx\_SCK} = f_{SPICLK}/(CPSDVSR \times (1+SCR)), \quad (29.2)$$

где f_{SPICLK} – частота входного синхросигнала SPICLK;

CPSDVSR – коэффициент первого делителя частоты (задается в регистре CPSR);

SCR – коэффициент второго делителя частоты (задается в регистре CR0).

Сформированный синхросигнал подается на вывод SPIx_SCK (где x – номер контролера 0, 1) микроконтроллера и далее к подключенным внешним ведомым устройствам.

В режиме ведомого значения коэффициентов делителей не важны. Внешний синхросигнал подается на вывод SPIx_SCK и тактирует прием и передачу данных.

Для корректной работы всегда должны соблюдаться условия:

- в режиме мастера для формируемого синхросигнала:

$$f_{SPIx\_SCK} \le f_{SYSCLK}/2; \quad (29.3)$$

- в режиме ведомого для входящего внешнего синхросигнала:

$$f_{SPIx\_SCK} \le f_{SYSCLK}/12. \quad (29.4)$$

Буферы приема и передачи

Для хранения передаваемых и принятых данных в контроллере SPI имеются два 16-разрядных буфера, организованных по типу FIFO. Каждый буфер может хранить до восьми слов данных. Буфер для передаваемых данных доступен только для записи, а буфер принятых данных – только для чтения.

Данные для передачи записываются в буфер через регистр DR. Допускается заранее заполнить буфер или записывать в него данные в течение работы контроллера. Состояние буфера можно контролировать с помощью битов TNF и TFE регистра SR. Если контроллер выключен (сброшен бит SSE регистра CR1), то запись в регистр DR приведет к тому, что данные будут размещены в буфере и будут переданы после включения контроллера. Если контроллер включен и выбран режим мастера, то в случае отсутствия данных в буфере

запись в регистр DR приведет к немедленному началу передачи. Если запись данных в регистр DR происходит во время текущей передачи, то данные размещаются в буфере.

Полученные данные автоматически сохраняются в буфере принятых данных. Извлечь данные из буфера возможно чтением регистра DR. Состояние буфера можно контролировать с помощью битов RFF и RNE регистра SR.

Размер передаваемого кадра данных может быть от 4 до 16 бит, что задается полем DSS регистра CR0. Если выбран размер кадра менее 16 бит, данные выравниваются по правой границе; неиспользуемые биты игнорируются.

29.2 Интерфейс прямого доступа к памяти

Контроллер SPI имеет интерфейс подключения к контроллеру прямого доступа к памяти. Работа в данном режиме контролируется регистром DMACR.

Сигналы для приема:

- RXDMASREQ – запрос передачи отдельного слова, инициируется приемопередатчиком. Сигнал переводится в активное состояние в случае, если буфер приемника содержит, по меньшей мере, одно слово;
- RXDMABREQ – запрос блочного обмена данными, инициируется приемопередатчиком. Сигнал переходит в активное состояние в случае, если заполнение буфера приемника превысило заданный порог. Порог программируется индивидуально посредством поля RXIFLSEL регистра CR1;
- RXDMACLR – сброс запроса на DMA, инициируется контроллером DMA с целью сброса принятого запроса. В случае если был запрошен пакетный обмен данными, сигнал сброса формируется в ходе передачи последнего слова данных в пакете.

Сигналы для передачи:

- TXDMASREQ – запрос передачи отдельного слова, инициируется приемопередатчиком. Сигнал переводится в активное состояние в случае, если буфер передатчика содержит, по меньшей мере, одну свободную ячейку;
- TXDMABREQ – запрос блочного обмена данными, инициируется приемопередатчиком. Сигнал переводится в активное состояние в случае, если заполнение буфера передатчика ниже заданного порога. Порог программируется индивидуально посредством поля TXIFLSEL регистра CR1;
- TXDMACLR – сброс запроса на DMA, инициируется контроллером DMA с целью сброса принятого запроса. В случае если был запрошен пакетный обмен данными, сигнал сброса формируется в ходе передачи последнего слова данных в пакете.

Сигналы блочного и одноэлементного обмена данными не являются взаимно исключающими, они могут быть инициированы одновременно. Например, в случае если заполнение данными буфера приемника превышает пороговое значение, формируется как сигнал запроса одноэлементного обмена, так и сигнал запроса блочного обмена данными. В случае если количество данных в буфере приема меньше порогового значения, формируется только запрос одноэлементного обмена. Это бывает полезно в ситуациях, при которых объем данных меньше размера блока.

Например, нужно принять 19 слов, а порог заполнения буфера установлен равным четырем. Тогда контроллер DMA осуществит четыре пакетных передачи блоков по четыре слова, а оставшиеся три слова – в ходе трех одиночных обменов, поскольку для них контроллер SPI не инициирует процедуру пакетного обмена.

Каждый инициированный приемопередатчиком сигнал запроса DMA остается активным до момента его сброса соответствующим сигналом от контроллера DMA.

После снятия сигнала сброса приемопередатчик вновь получает возможность сформировать запрос на DMA в случае выполнения описанных выше условий. Все запросы DMA снимаются после запрета работы приемопередатчика, а также в случае снятия сигнала разрешения DMA.

29.3 Функционирование

Приемопередающая логика модуля, управляющие регистры и FIFO по умолчанию находятся в сбросе. При подготовке к работе необходимо подать тактирование установкой бита CLKEN, а затем снять сброс путём установки бита RSTDIS в соответствующем регистре SPICLKCFG блока RCU.

Прежде чем разрешить работу битом SSE регистра CR1, следует сконфигурировать контроллер SPI посредством регистров CR0 и CR1, а также, если это необходимо, запрограммировать маски прерываний.

Динамическое изменение конфигурации устройства не допускается.

Для протокола SPI дополнительно задаются полярность и фаза сигнала (биты SPH и SPO регистра CR0).

После разрешения работы приемопередатчик готов к обмену данными с внешними устройствами по линиям SPIx_TX (передача данных к внешнему устройству) и SPIx_RX (прием данных от внешнего устройства).

В зависимости от режима работы сигнал на линии SPIx_FSS используется либо для кадровой синхронизации (интерфейс SSI, активное состояние – высокий уровень), либо для выбора устройства в режиме ведомого (интерфейсы SPI и Microwire, активное состояние – низкий уровень).

Во всех трех режимах SPI, Microwire и SSI синхросигнал SPIx_SCK формируется только тогда, когда приемопередатчик готов к обмену данными. Перевод сигнала SPIx_SCK в неактивное состояние используется как признак таймаута приемника, то есть наличия в буфере приемника необработанных данных по истечении заданного интервала времени.

Установка бита MS регистра CR1 включает режим ведомого устройства. В этом режиме разрешение или запрещение передачи данных через выход SPIx_TX контролируется битом SOD. На прием синхросигнала и данных состояние этого бита влияния не оказывает.

Интерфейс SPI

Интерфейс SPI реализует полнодуплексный режим передачи данных.

Включает одну линию синхронизации SPIx_SCK, две линии приема и передачи данных SPIx_RX и SPIx_TX, а также линию выбора устройства (для режима ведомого) SPIx_FSS.

Если устройство функционирует в режиме ведомого, то на его вход SPIx_FSS должен подаваться низкий уровень сигнала в течение всей передачи кадра (последовательность передаваемых бит данных длиною от 4 до 16 бит).

Передача данных может быть одиночной (один кадр) или непрерывной (более одного кадра подряд). Данные передаются старшим битом вперед.

Особенностью интерфейса SPI является то, что в нем реализована возможность задания полярности и фазы тактового сигнала. Бит SPO регистра CR0 задает полярность тактового сигнала, т. е. определяет, какой уровень сигнала будет удерживаться на линии SPIx_SCK в то время, когда линия не активна.

Бит SPH задает фазу тактового сигнала. Фактически, он задает порядок считывания и выставления данных. По умолчанию бит SPH сброшен и выборка данных на линиях SPIx_TX и SPIx_RX происходит по переднему фронту сигнала синхронизации, а установка – по заднему.

Передним всегда считается тот фронт сигнала, который является началом передачи первого бита, см. рисунок 29.3.

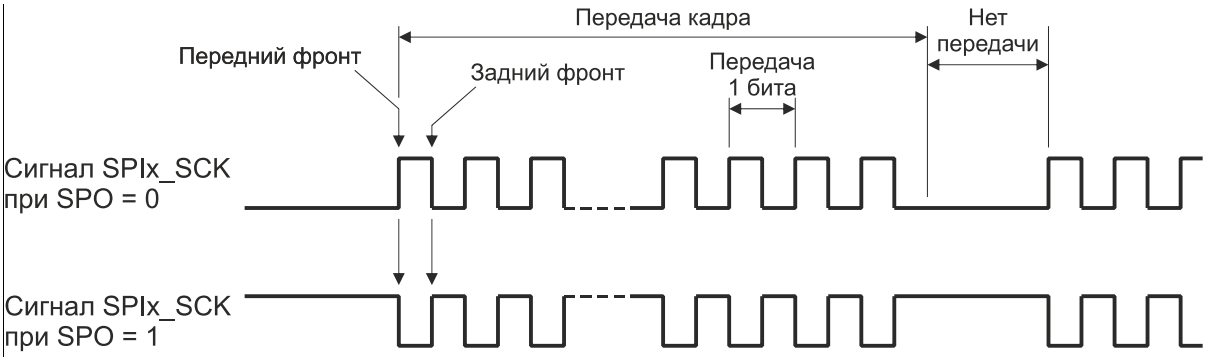


Рисунок 29.3 – Сигнал синхронизации SPIx_SCK при разных состояниях бита SPO

Комбинации битов SPO и SPH задают четыре режима обмена данными, см. рисунок 29.4.

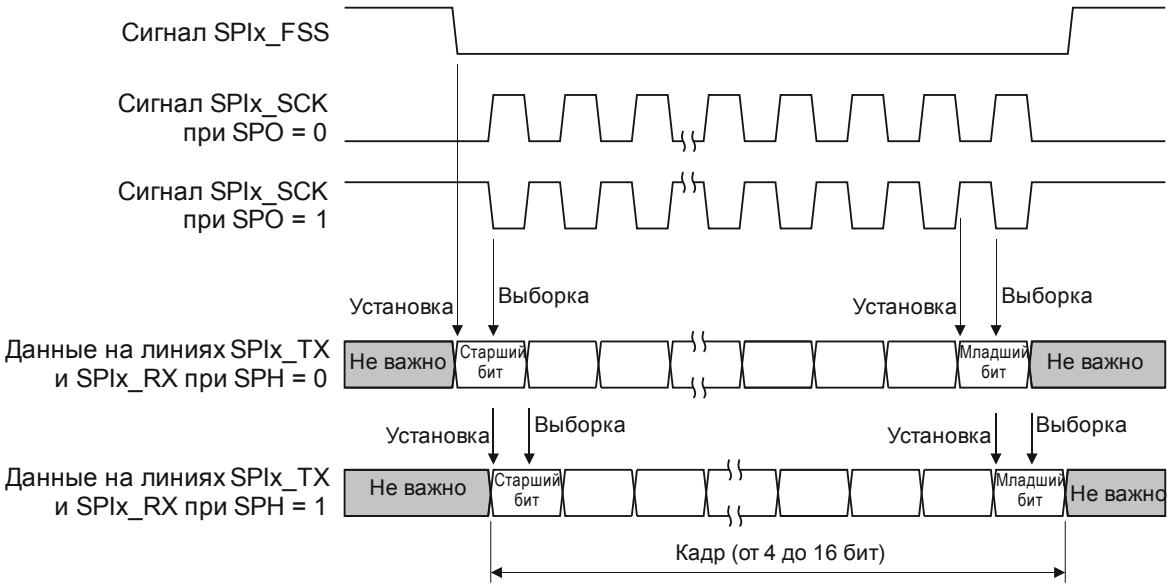


Рисунок 29.4 – Передача кадров данных в интерфейсе SPI

На рисунке 29.5 показано поведение сигналов при непрерывной передаче кадров данных.

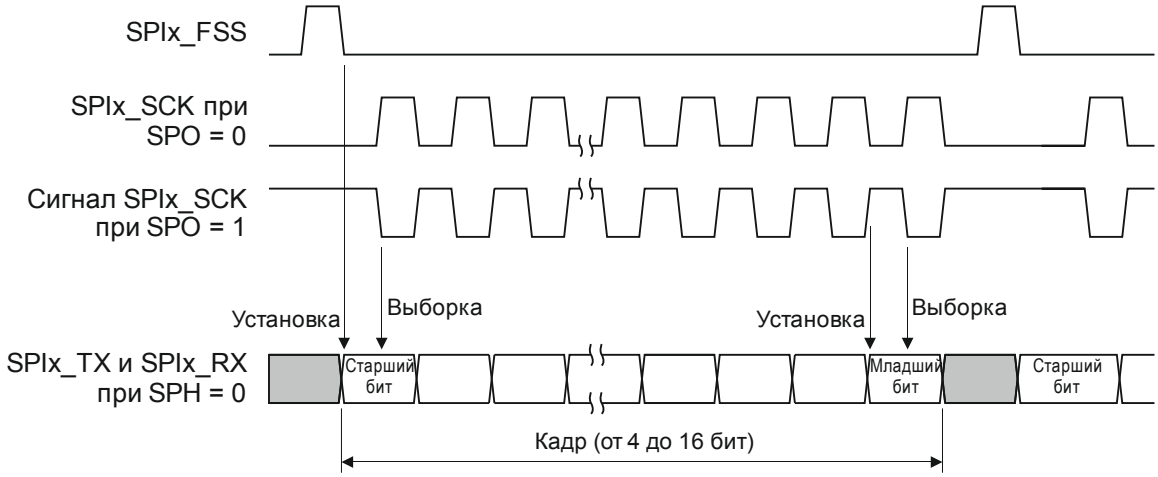


Рисунок 29.5 – Поведение сигналов при непрерывной передаче кадров данных

В режиме непрерывной передачи данных при условии SPH = 0 на линии SPIx_FSS должны формироваться импульсы между передачами кадров данных. Это связано с тем, что в этом режиме низкий уровень сигнала на линии SPIx_FSS ведомого устройства блокирует запись в сдвиговый регистр. Поэтому мастер должен переводить линию SPIx_FSS в высокий уровень по окончании передачи каждого кадра, разрешая, таким образом, запись новых данных. По окончании приема последнего бита кадра линия SPIx_FSS переводится в состояние логической единицы по истечении одного такта сигнала SPIx_SCK.

В режиме непрерывной передачи данных при условии SPH = 1 низкий уровень сигнала на линии SPIx_FSS не блокирует запись в сдвиговый регистр. Линия SPIx_FSS может оставаться в состоянии нуля в течение передачи всех кадров. Только по окончании передачи линия может быть переведена в состояние логической единицы.

Интерфейс Microwire

Интерфейс Microwire реализует полудуплексный режим передачи данных.

Включает линию синхронизации SPIx_SCK, две линии приема и передачи данных SPIx_RX и SPIx_TX, а также линию выбора устройства (для режима ведомого) SPIx_FSS.

Если устройство функционирует в режиме ведомого, то на его вход SPIx_FSS должен подаваться низкий уровень сигнала в течение всей передачи кадра (последовательность передаваемых бит данных длиною от 4 до 16 бит).

Передача данных может быть одиночной (один кадр) или непрерывной (более одного кадра подряд). Данные передаются старшим битом вперед.

Перед началом передачи линия SPIx_FSS переводится в низкое состояние.

Каждая передача начинается с передачи от мастера к ведомой 8-битной управляющей последовательности. В течение передачи этой последовательности приемник мастера не обрабатывает входящие данные. После того как управляющая последовательность передана и декодирована одним из ведомых устройств, этот ведомый выдерживает паузу в один такт синхросигнала и начинает передавать мастеру кадр данных, см. рисунок 29.6.

Выставление данных происходит по заднему фронту сигнала SPIx_SCK, а считывание – по переднему.

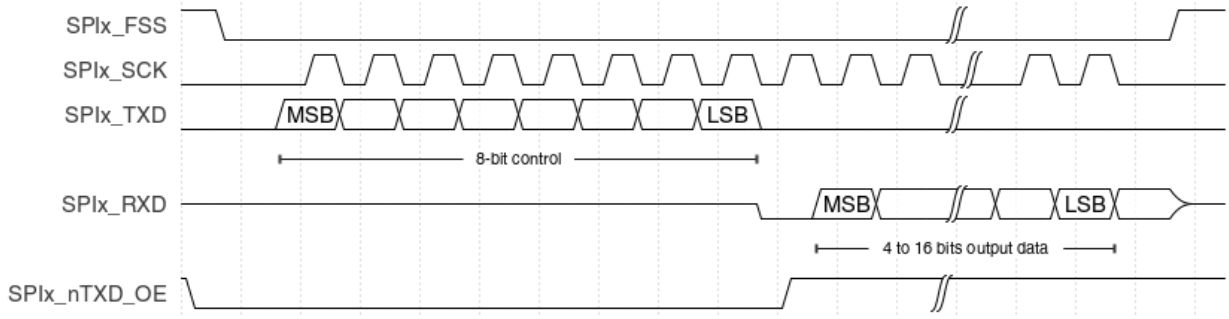


Рисунок 29.6 – Передача кадра данных в интерфейсе Microwire

По окончании приема данных линия SPIx_FSS переводится в высокое состояние.

Примечание – В течение времени, когда передается управляющая последовательность и между передачами, линия SPIx_RX может находиться в третьем состоянии.

В режиме непрерывной передачи начало и завершение передачи нескольких кадров данных аналогично передаче одного кадра. Линия SPIx_FSS удерживается в нуле в течение всего сеанса передачи. По окончании передачи одного кадра данных начинается передача управляющей последовательности без паузы, см. рисунок 29.7.

Примечание – Буферы FIFO приема и передачи данных не очищаются автоматически, даже в случае запрещения работы сбросом бита SSE.

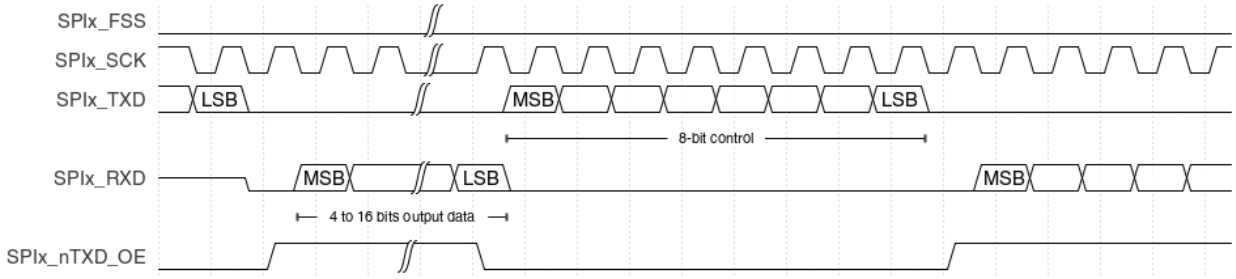


Рисунок 29.7 – Передача кадров данных в интерфейсе Microwire

Для реализации возможности объединения двух линий TxD и RxD при работе в режиме MICROWIRE модуль интерфейса имеет в составе сигнал nSSPOE. Это внутренний сигнал управления выходом линии TxD интерфейса SPI. В фазе чтения по линии RxD на нем устанавливается высокий уровень, и он переводит вывод порта TxD в высокоимпедансное состояние.

Интерфейс SSI

Интерфейс SSI реализует полнодуплексный режим передачи данных.

Включает одну линию синхронизации SPIx_SCK, две линии приема и передачи данных SPIx_RX и SPIx_TX, а также линию выбора устройства SPIx_FSS.

Перед началом передачи каждого кадра на линии SPIx_FSS формируется импульс длительностью в один период сигнала SPIx_SCK. Далее мастер и ведомый передают данные. Установка данных производится по переднему фронту синхросигнала, а выборка – по заднему, см. рисунок 29.8. Весь цикл передачи начинается сразу же после появления хотя бы одного элемента в буфере FIFO передатчика.

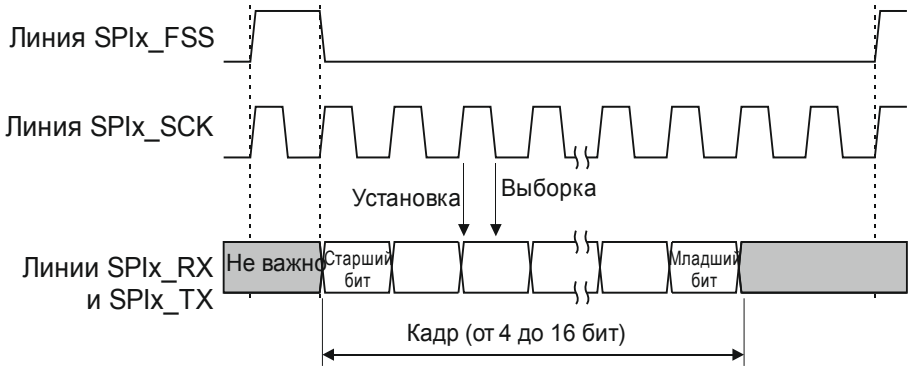


Рисунок 29.8 – Передача кадра данных в интерфейсе SSI

Режим непрерывной передачи кадров данных показан на рисунке 29.9.

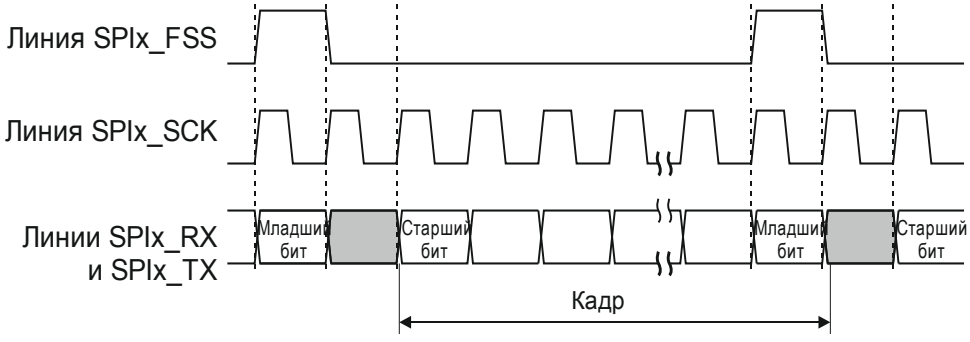


Рисунок 29.9 – Непрерывная передача кадров данных в интерфейсе SSI

29.4 Прерывания

Буфер приема и буфер передачи могут генерировать 4 независимых маскируемых запроса на прерывания:

- TXINTR – запрос на обслуживание буфера передатчика (опустошение буфера равно или ниже порога);
- RXINTR – запрос на обслуживание буфера приемника (заполнение буфера равно или выше порога);
- RTINTR – таймаут ожидания чтения данных из буфера приемника (буфер приемника не пуст и не было попыток обращения к нему в течение времени, равного передаче 32 бит);
- RORINTR – переполнение буфера приемника.

Пороги программируются индивидуально для каждого буфера посредством полей TXIFLSEL и RXIFLSEL регистра CR1.

Каждый из сигналов может быть маскирован путем установки соответствующего бита в регистре маски IMSC.

Источник прерывания также можно определить, считав состояние регистра RIS или регистра MIS (маскированные прерывания). Сброс прерывания осуществляется программно путём установки соответствующего бита в регистре ICR.

На контроллер прерываний PLIC подключена одна линия прерывания, объединяющая по логическому «ИЛИ» запросы: TXINTR, RXINTR, RTINTR и RORINTR.

30 Контроллер интерфейса USB FullSpeed

USB (Universal Serial Bus) – последовательная шина, которая позволяет передавать данные между хост-компьютером и различными периферийными устройствами. Наиболее важным свойством USB является возможность подключения и отключения периферийных устройств без выключения всей системы (один из классов механизма «Plug and Play»). USB-хост имеет соединения типа точка-точка с периферийными устройствами через поярусную топологию типа «звезда», в узлах которой содержится USB-устройство, называемое хабом. USB позволяет соединить до 127 устройств.

Упрощенная функциональная схема контроллера USB на рисунке 30.1.

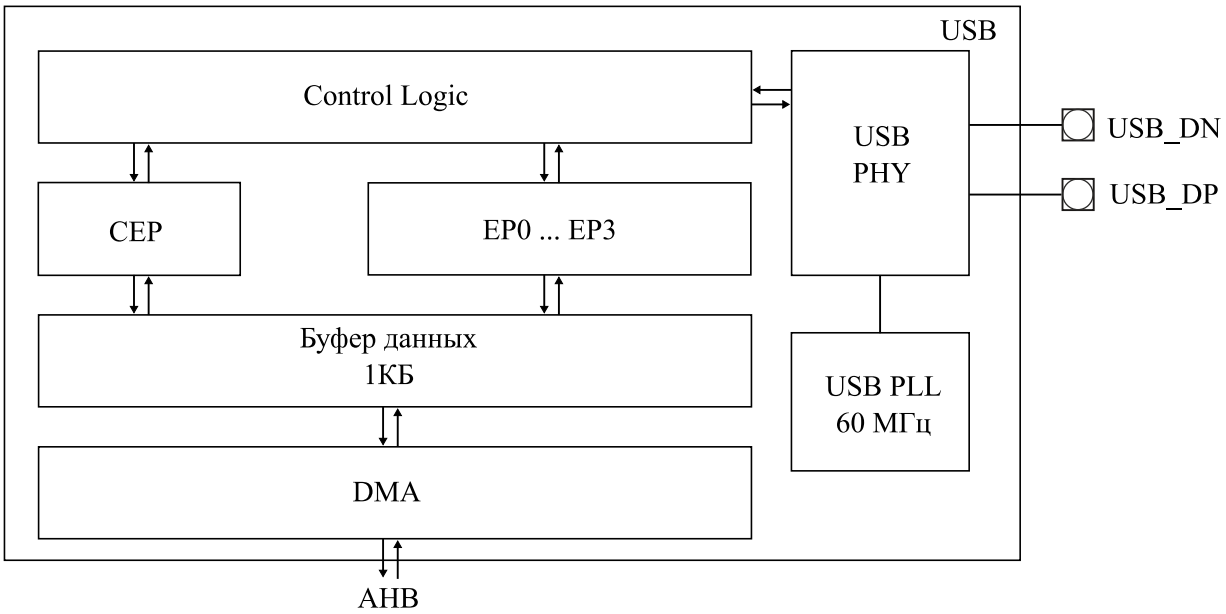


Рисунок 30.1 – Структурная схема контроллера USB

Источник тактирования USB выбирается битовым полем USBCLKSEL регистра USBCLKCFG. Схема выбора синхросигнала блока USB показана на рисунке 30.2.

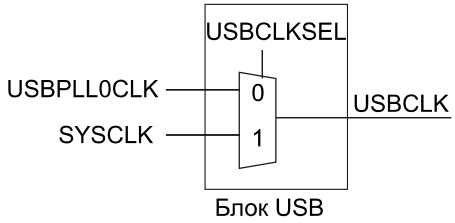


Рисунок 30.2 – Схема селектора синхросигнала блока USB

30.1 Функции устройства

Блок USB содержит одну контрольную точку (CEP) и четыре конечных точки (EP0 – EP3).

Для хранения данных контрольной и конечных точек имеется отдельный блок ОЗУ объемом 1Кбайт.

30.2 Работа устройства

Работа при управляющих (Control) передачах

Возможны четыре типа управляющих передач:
- управляющая запись с передачей данных;
- управляющая запись без передачи данных;
- управляющее чтение с передачей данных;
- управляющее чтение без передачи данных.

Этап передачи команды (Setup stage)

Передача команды – это Setup-токен с восемью байтами управляющих данных, которые содержат запрашиваемую информацию от хоста к устройству (см. рисунок 30.3).

Рисунок 30.3 – Передача команды

Для этого этапа доступны два прерывания: по приему Setup-токена и по приему Setup-пакета (после приема 8 байт данных без каких-либо ошибок).

ЦП должен сбросить биты соответствующих прерываний, если они возникли, обработать принятые на этом этапе данные и подготовиться к опциональному этапу передачи данных (Data stage). Четыре регистра SETUP1_0, SETUP3_2, SETUP5_4 и SETUP7_6 хранят данные, принятые контрольной точкой на этапе передачи команды.

Этап передачи данных (Data stage)

В случае управляющей записи с передачей данных, когда происходит OUT-передача, принятые данные записываются в буфер контрольной точки, генерируется прерывание и устанавливается флаг DATAPKTRECINT в регистре USB_EP_x_IRQ_STAT (x – номер контрольной точки). ЦП должен обработать принятые данные, количество байт которых указывается в регистре OUT_TRNSFR_CNT. Передача данных показана на рисунке 30.4.

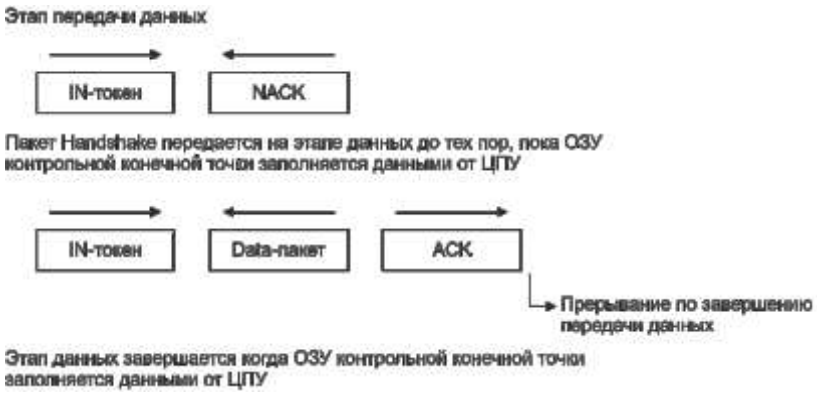


Рисунок 30.4 – Передача данных

В случае управляющего чтения с передачей данных, когда происходит IN-передача, ЦП должен загрузить буфер контрольной точки данными для передачи хосту, затем загрузить регистр IN_TRNSFR_CNT значением количества передаваемых данных. Запись значения количества байт в регистр разрешает USB-устройству передать данные хосту на

приходящий IN-токен. После, ЦП может ожидать прерывание, которое будет сигнализировать о завершении этапа передачи данных и установки флага DATAPKTTRINT в регистре USB_EP_x_IRQ_STAT.

В случае если требуется послать STALL-Handshake хосту, ЦП должен установить бит STALL в регистре CEP_CTRL_STAT.

Этап подтверждения (Status stage)

После того, как микроконтроллер завершил этап передачи данных, бит NAK_CLEAR должен быть сброшен (см. рисунок 30.5). Сбрасывая этот бит, ЦП позволяет хосту перейти к следующей управляющей передаче. Этот бит устанавливается USB-устройством после приема Setup-токена.

В случае если требуется послать STALL-Handshake хосту, ЦП должен установить бит STALL и сбросить бит NAK_CLEAR.

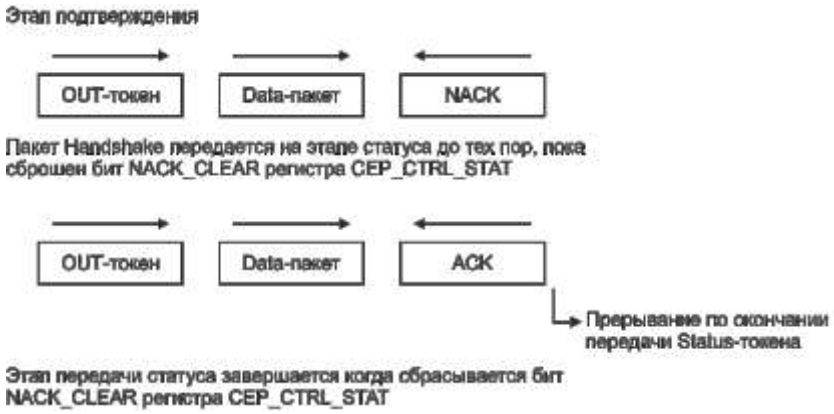


Рисунок 30.5 – Подтверждение

Работа при приеме IN-токенов

Данные для любой IN-посылки записываются в буфер микроконтроллером в момент приема устройством IN-токена от хоста. Есть три различных режима подтверждения данных, при которых они будут посланы хосту:

- автоподтверждения;
- подтверждения вручную;
- подтверждения «на лету».

Режим автоподтверждения

Если конечная точка настроена на работу в режиме автоподтверждения, то она будет отсылать количество байт данных, равное значению EP_MPS регистра USB_EP_x_MPS. Контроллер конечной точки будет ждать, пока количество байт данных в буфере станет равным EP_MPS, после чего подтвердит отправку. Если микроконтроллеру требуется отправить короткий пакет в конце передачи, следует установить бит PKTEND регистра USB_EP_x_RSP_SC. Когда бит будет установлен, оставшееся количество байт данных будет отправлено хосту на следующий IN-токен.

Этот режим требует минимум вмешательства центрального процессора, т. к. большинство работы будет сделано на аппаратном уровне. Режим может быть выбран, когда размер пересылаемых данных в хост всегда равен значению EP_MPS. Более подробно действия контроллера точки отображены в таблице 30.1.

Таблица 30.1 – Действия Endpoint в режиме автоподтверждения

Бит PacketEnd	Количество доступных данных в буфере	Действие
0	< EP_MPS	Ответ NAK
0	≥ EP_MPS	Ответ количеством данных, равных EP_MPS
1	< EP_MPS	Ответ количеством данных, доступных в буфере
1	≥ EP_MPS	Ответ количеством данных, равных EP_MPS

Режим подтверждения вручную

Если конечная точка настроена на работу в режиме подтверждения вручную, то она будет отсылать данные тогда, когда это будет разрешено микроконтроллером. ЦП сначала записывает данные в буфер, а затем записывает количество байт данных (значение EP_CNT) в регистр USB_EP_x_CNT. Как только будет записано верное количество передаваемых байт, данные будут отправлены хосту на следующий входной IN-токен.

Этот режим требует вмешательства ЦП каждую посылку и может быть использован, если количество передаваемых байт каждый раз непостоянное и решается ЦП. Более подробно действия контроллера точки отображены в таблице 30.2.

Таблица 30.2 – Действия Endpoint в режиме подтверждения вручную

EP_CNT записан	Количество доступных данных в буфере	Действие
Нет	-	Ответ NAK
Да	EP_CNT	Ответ количеством данных, равных EP_CNT

Режим подтверждения «на лету»

Простейший режим работы, где не требуется процедуры подтверждения. Буфер заполняется центральным процессором. Если хост отправил IN-токен, данные в буфере автоматически подтверждаются и отправляются хосту. Если количество данных в буфере превышает размер одного пакета EP_MPS, контроллер точки автоматически упаковывает данные в цепочку пакетов, размером не более EP_MPS, и отправляет их хосту.

Этот режим требует минимум вмешательства ЦП и лучше всего подходит для изохронных передач, где скорость доставки данных более важна, чем размер пакета. Более подробно действия контроллера точки отображены в таблице 30.3.

Таблица 30.3 – Действия Endpoint в режиме подтверждения «на лету»

Количество доступных данных в буфере	Количество отправленных данных
>= EP_MPS	Ответ количеством данных, равных EP_MPS
< EP_MPS	Ответ количеством данных, доступных в буфере

30.3 Рекомендации по программированию

Настройка PLLUSB

Логика модуля USB использует выделенное тактирование от блока PLLUSB. Данный блок должен быть настроен на выходную частоту 60 МГц. Настройка блока PLLUSB осуществляется через регистры PLLUSBCFGx аналогично PLLSYS (подраздел 4.2). Выбор тактового сигнала для USB осуществляется битовым полем USBCLKSEL регистра USBCLKCFG.

Инициализация конечной точки

Первым шагом для инициализации конечной точки является запись соответствующих значений в регистры USB_EP_x.START_ADDR и USB_EP_x.END_ADDR, указывая начальный и конечный адрес буфера конечной точки x. Выделяемые диапазоны адресов для различных конечных точек, в том числе и для контрольной (CEP), не должны пересекаться.

Биты EP_TYPE в регистре USB_EP_x.CONFIG должны быть установлены в соответствии с типом конечной точки. После установки всех необходимых конфигурационных бит регистра, конечная точка должна быть включена с помощью бита EP_VALID в регистре USB_EP_x.CONFIG. Выбор режима направления передачи осуществляется с помощью бита EP_DIR. Также необходимо дополнительно выбрать режим работы конечной точки IN с помощью битового поля MODE в регистре USB_EP_x.RSP_SC. В регистр USB_EP_x.MPS должен быть записан максимальный размер пакета.

Как только передача и/или прием будут разрешены, регистры USB_EP_x.CONFIG, USB_EP_x.MPS, USB_EP_x.START_ADDR и USB_EP_x.END_ADDR не должны изменяться программным обеспечением.

Процесс подготовки данных для передачи

При передаче данных к хосту (IN-транзакции) необходимо сообщить DMA адрес размещения передаваемых данных в общем адресном пространстве с помощью регистра AHB_DMA_ADDR, затем записать количество данных для пересылки в регистр DMA_CNT. Далее с помощью битового поля DMA_EP_ADDR в регистре DMA_CTRL_STS задается адрес конечной точки (для управляющей конечной точки адрес всегда равен нулю), в буфер которой DMA будет пересылать данные. Записью единицы в битовое поле DMARW регистра DMA_CTRL_STS производится выбор операции DMA (запись в буфер USB) и с помощью бита DMAEN активируется запуск DMA на пересылку данных. По окончанию операции пересылки данных DMA будет установлен бит DMACMPL в регистре INTSTAT1. Далее необходимо задать количество передаваемых байтов в регистре CEP_IN_XFRCNT (для управляющей конечной точки) либо в регистре USB_EP_x.CNT (для остальных конечных точек).

Процесс получения данных при приеме

При передаче данных от хоста (OUT-транзакции) количество принятых байтов отображается регистре CEP_OUT_XFRCNT (для управляющей конечной точки) либо в регистре USB_EP_x.AVAIL_CNT (для остальных конечных точек). Для DMA необходимо сообщить адрес размещения принимаемых данных в общем адресном пространстве с помощью регистра AHB_DMA_ADDR, затем записать количество данных для пересылки в регистр DMA_CNT. Далее с помощью битового поля DMA_EP_ADDR в регистре DMA_CTRL_STS задается адрес конечной точки (для управляющей конечной точки адрес всегда равен нулю), из буфера которой DMA будет забирать данные. Записью нуля в битовое поле DMARW регистра DMA_CTRL_STS производится выбор операции DMA (чтение из буфера USB) и с помощью бита DMAEN активируется запуск DMA на пересылку данных. По окончанию операции пересылки данных DMA будет установлен бит DMACMPL в регистре INTSTAT1.

31 Программно-аппаратные средства отладки

Для освоения и изучения 32-разрядного микроконтроллера К1921ВГ015, а также для макетирования и отладки систем пользователя на его основе, следует использовать макетно-отладочную плату, которая позволяет подключать внешние элементы к портам микроконтроллера, работать с внешними интерфейсами, а также программировать встроенную Flash-память и выполнять отладку и оценку работы прикладных программ.

Для создания программного обеспечения можно использовать интегрированную среду разработки Eclipse, компилятор GCC для RISC-V и систему отладки OpenOCD.

Адаптер Olimex ARM-USB-OCD-H или J-Link обеспечивает взаимодействие между интегрированной средой разработки Eclipse (с GCC компилятором для RISC-V и системой отладки OpenOCD), установленной на персональном компьютере, и отладочными ресурсами, встроенными в микроконтроллер К1921ВГ015, а также выполнение отладочных функций. Информационный обмен с микроконтроллером осуществляется по отладочному порту JTAG.

Заключение

В настоящем техническом описании КФДЛ.431295.078ТО представлено описание архитектуры, функционального построения и периферии микроконтроллера К1921ВГ015. Техническое описание может служить практическим руководством по применению микроконтроллера для разработчиков систем на его основе и программистов.

Приложение А
(обязательное)
Регистры микроконтроллера

Для каждого блока отдельно приведены абсолютные базовые адреса, а в описаниях регистров показаны смещения относительно его базового адреса, если не указано иное. Абсолютный адрес регистра вычисляется путем сложения абсолютного базового адреса блока и смещения этого регистра.

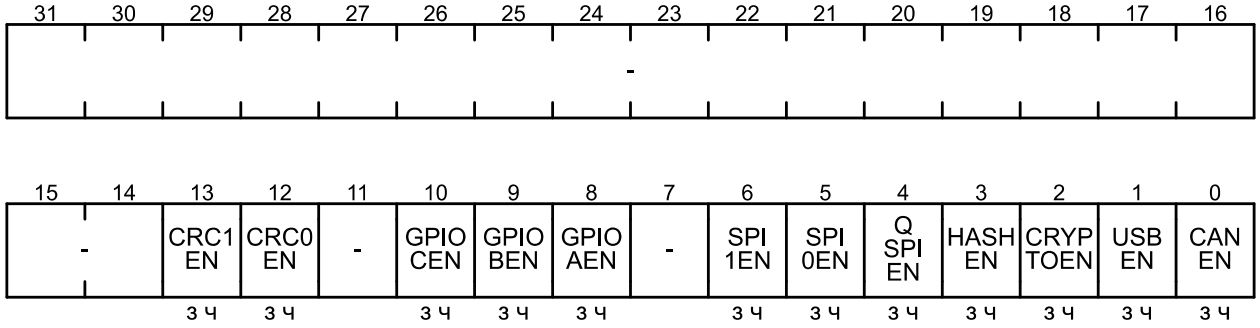
А.1 Регистры блока управления сбросом и синхронизацией RCU

Базовый адрес: 3000_E000h

Смещение:	+ 70h (UARTCLKCFG)	Регистры настройки тактирования UART
	+ 94h (SPICLKCFG)	Регистры настройки тактирования SPI

CGCFGAHB – регистр разрешения тактирования периферийных блоков шины AHB

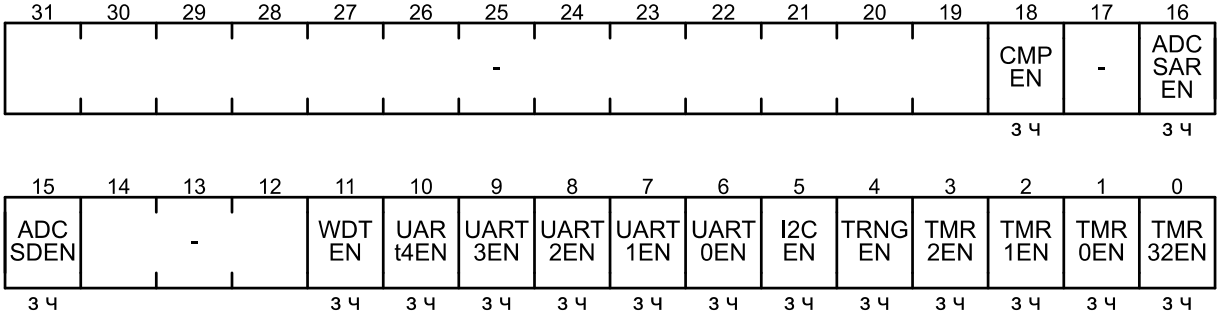
Смещение: + 0h
Сброс: 0h



Поле	Биты	Описание
CRC1EN	13	Бит разрешения тактирования CRC1
CRC0EN	12	Бит разрешения тактирования CRC0
GPIOCEN	10	Бит разрешения тактирования порта C
GPIOBEN	9	Бит разрешения тактирования порта B
GPIOAEN	8	Бит разрешения тактирования порта A
SPI1EN	6	Бит разрешения тактирования SPI1
SPI0EN	5	Бит разрешения тактирования SPI0
QSPIEN	4	Бит разрешения тактирования QSPI
HASHEN	3	Бит разрешения тактирования HASH
CRYPTOEN	2	Бит разрешения тактирования CRYPTO
USBEN	1	Бит разрешения тактирования USB
CANEN	0	Бит разрешения тактирования CAN
–	31-14, 11, 7	Зарезервировано

CGCFGAPB – регистр разрешения тактирования периферийных блоков шины APB

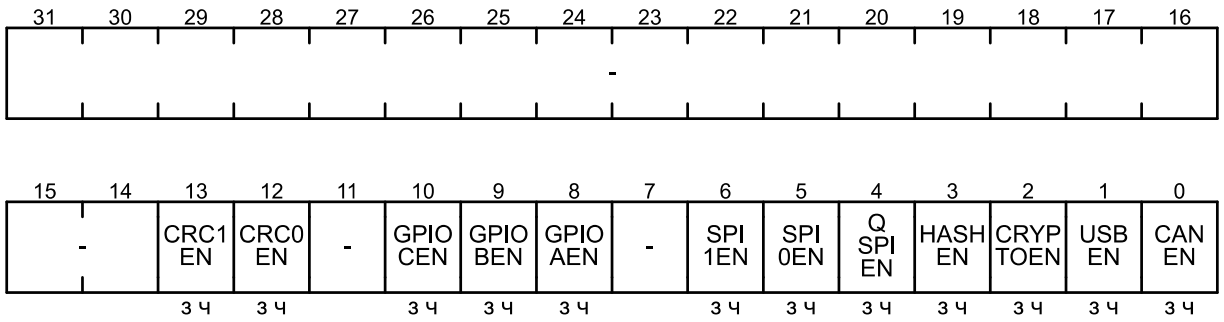
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание
CMPEN	18	Бит разрешения тактирования CMP
ADCSAREN	16	Бит разрешения тактирования ADCSAR
ADCSDEN	15	Бит разрешения тактирования ADCSD
WDTEN	11	Бит разрешения тактирования WDT
UART4EN	10	Бит разрешения тактирования UART4
UART3EN	9	Бит разрешения тактирования UART3
UART2EN	8	Бит разрешения тактирования UART2
UART1EN	7	Бит разрешения тактирования UART1
UART0EN	6	Бит разрешения тактирования UART0
I2CEN	5	Бит разрешения тактирования I2C
TRNGEN	4	Бит разрешения тактирования TRNG
TMR2EN	3	Бит разрешения тактирования TMR2
TMR1EN	2	Бит разрешения тактирования TMR1
TMR0EN	1	Бит разрешения тактирования TMR0
TMR32EN	0	Бит разрешения тактирования TMR32
–	31-19, 17, 14-12	Зарезервировано

RSTDISAHB – регистр включения периферийных блоков шины AHB

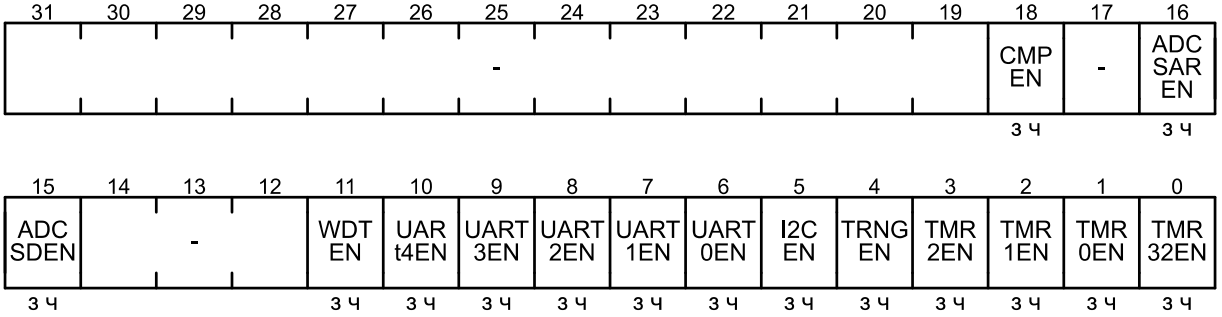
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
CRC1EN	13	Бит включения CRC1
CRC0EN	12	Бит включения CRC0
GPIOCEN	10	Бит включения порта C
GPIOBEN	9	Бит включения порта B
GPIOAEN	8	Бит включения порта A
SPI1EN	6	Бит включения SPI1
SPI0EN	5	Бит включения SPI0
QSPIEN	4	Бит включения QSPI
HASHEN	3	Бит включения HASH
CRYPTOEN	2	Бит включения CRYPTO
USBEN	1	Бит включения USB
CANEN	0	Бит включения CAN
–	31-14, 11, 7	Зарезервировано

RSTDISAPB – регистр включения периферийных блоков шины APB

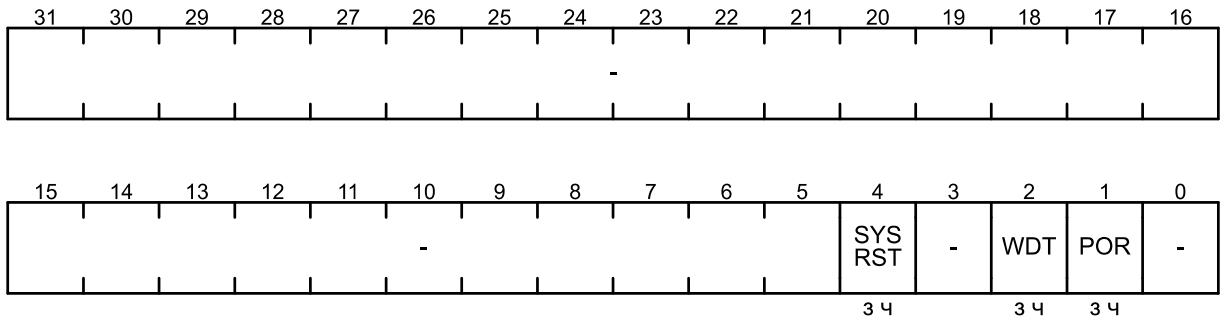
Смещение: + 18h
Сброс: 0h



Поле	Биты	Описание
CMPEN	18	Бит включения CMP
ADCSAREN	16	Бит включения ADCSAR
ADCSDEN	15	Бит включения ADCSD
WDTEN	11	Бит включения WDT
UART4EN	10	Бит включения UART4
UART3EN	9	Бит включения UART3
UART2EN	8	Бит включения UART2
UART1EN	7	Бит включения UART1
UART0EN	6	Бит включения UART0
I2CEN	5	Бит включения I2C
TRNGEN	4	Бит включения TRNG
TMR2EN	3	Бит включения TMR2
TMR1EN	2	Бит включения TMR1
TMR0EN	1	Бит включения TMR0
TMR32EN	0	Бит включения TMR32
–	31-19, 17, 14-12	Зарезервировано

RSTSTAT – регистр статуса сброса системных блоков

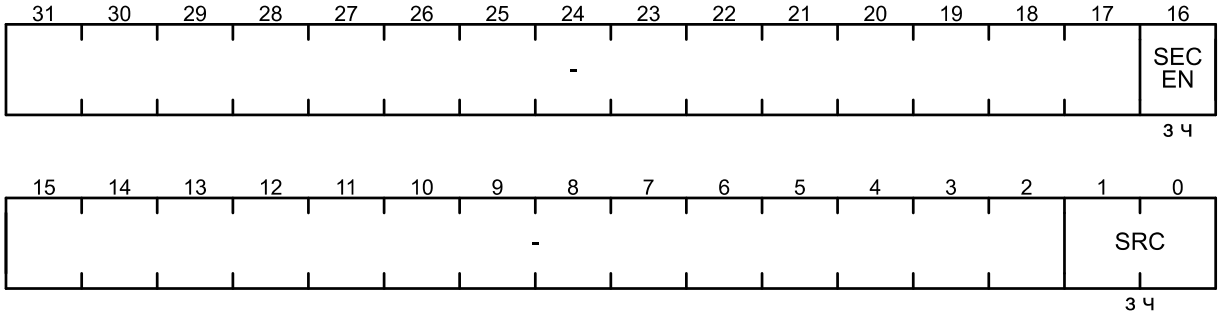
Смещение: + 20h
Сброс: 0h



Поле	Биты	Описание
SYSRST	4	Бит состояния сброса от системного сброса
WDT	2	Бит состояния сброса от WDT
POR	1	Бит состояния сброса от POR
–	31-5, 3, 0	Зарезервировано

SYSCLKCFG – регистр конфигурации системной частоты

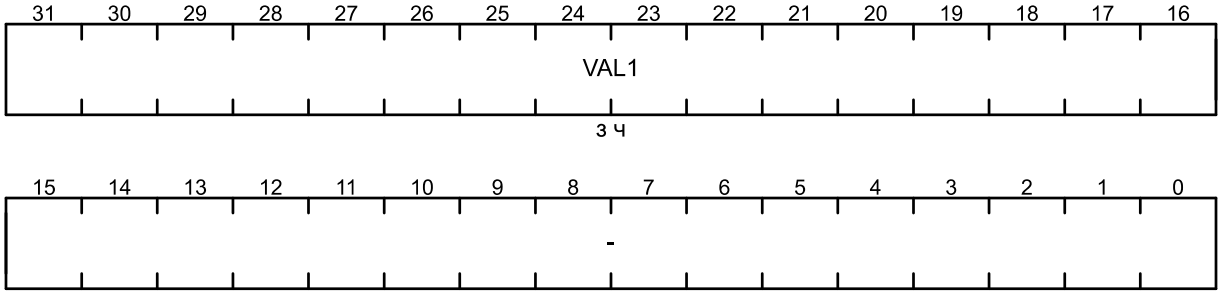
Смещение: + 30h
Сброс: 0h



Поле	Биты	Описание	
SECEN	16	Включение системы слежения тактового сигнала	
SRC	1-0	Выбор источника тактового сигнала	
		00b	Сигнал HSICLK (1МГц)
		01b	Сигнал HSECLK (XTAL)
		10b	Сигнал SYSPLL0CLK
		11b	Сигнал LSICLK (32кГц) от блока RTC
–	31-17, 15-2	Зарезервировано	

SECCNT0 – регистр 0 счетчика слежения системной частоты

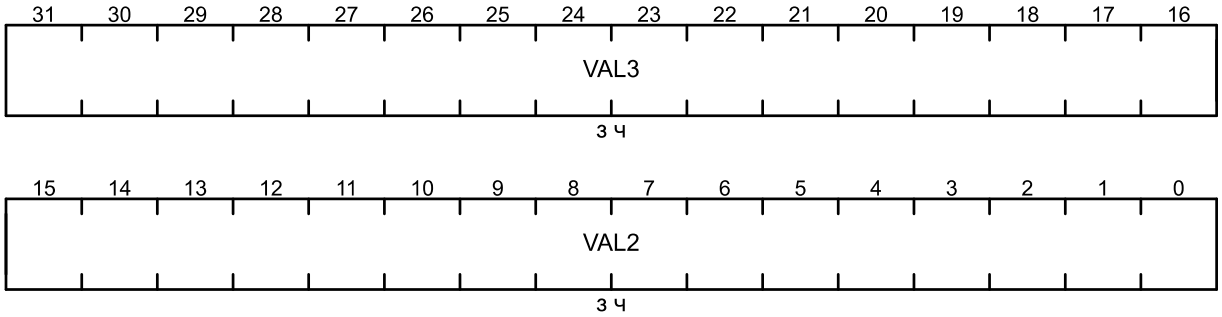
Смещение: + 34h
Сброс: 0h



Поле	Биты	Описание
VAL1	31-16	Максимальное значение счетчика периодов для детектирования пропадания сигнала HSECLK
–	15-0	Зарезервировано

SECCNT1 – регистр 1 счетчика слежения системной частоты

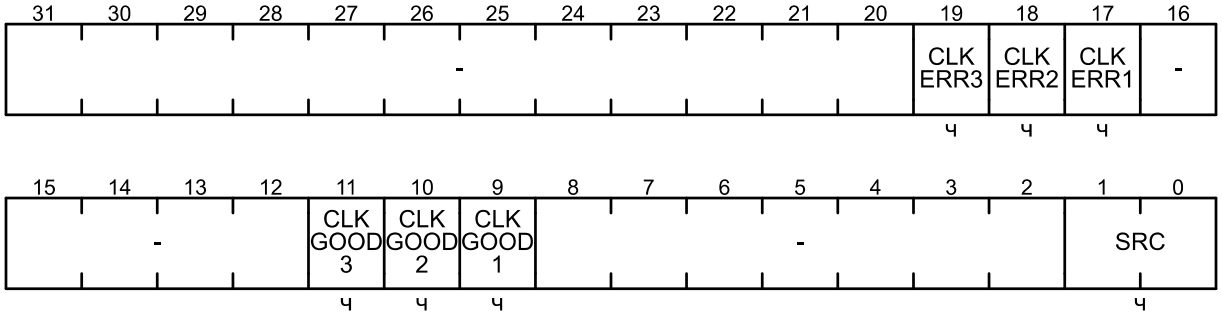
Смещение: + 38h
Сброс: 0h



Поле	Биты	Описание
VAL3	31-16	Максимальное значение счетчика периодов для детектирования пропадания сигнала LSICLK
VAL2	15-0	Максимальное значение счетчика периодов для детектирования пропадания сигнала SYSPLL0CLK

CLKSTAT – регистр статуса системной частоты

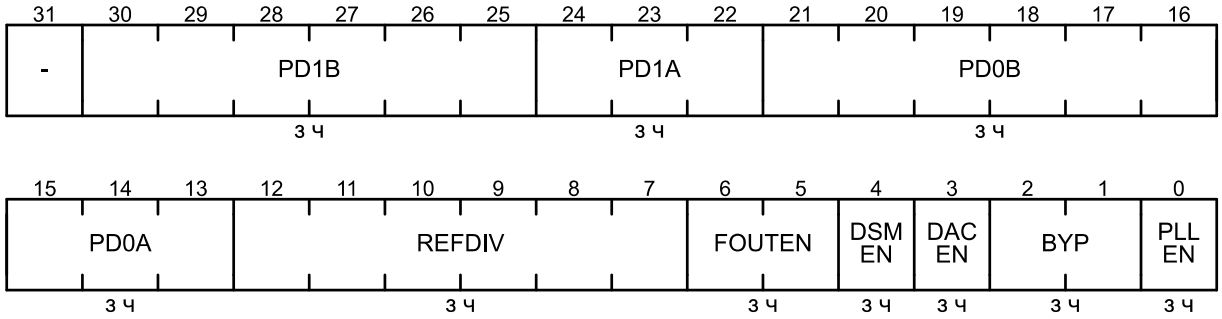
Смещение: + 3Ch
Сброс: 0h



Поле	Биты	Описание	
CLKERR3	19	Ошибка при работе источника тактового сигнала LSICLK	
CLKERR2	18	Ошибка при работе источника тактового сигнала SYSPLL0CLK	
CLKERR1	17	Ошибка при работе источника тактового сигнала HSECLK	
CLKGOOD3	11	Источник тактового сигнала LSICLK корректный	
CLKGOOD2	10	Источник тактового сигнала SYSPLL0CLKкорректный	
CLKGOOD1	9	Источник тактового сигнала HSECLK корректный	
SRC	1-0	Текущий источник тактового сигнала	
		00b	Сигнал HSICLK (1МГц)
		01b	Сигнал HSECLK (XTAL)
		10b	Сигнал SYSPLL0CLK
		11b	Сигнал LSICLK (32кГц) от блока RTC
–	31-20, 16-12, 8-2	Зарезервировано	

PLLSYSCFG0 – регистр 0 конфигурации PLL

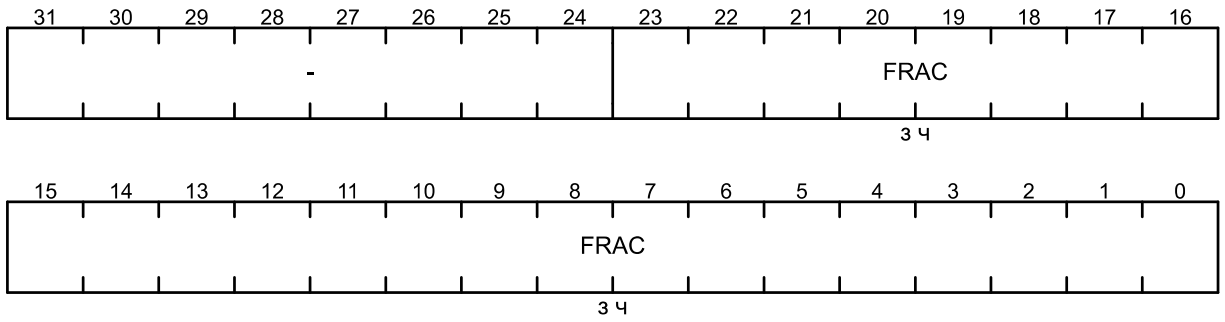
Смещение: + 50h
Сброс: 0h



Поле	Биты	Описание
PD1B	30-25	Коэффициент деления PD1B. Значения делителя PD1B+1
PD1A	24-22	Коэффициент деления PD1A. Значения делителя PD1A+1
PD0B	21-16	Коэффициент деления PD0B. Значения делителя PD0B+1
PD0A	15-13	Коэффициент деления PD0A. Значения делителя PD0A+1
REFDIV	12-7	Коэффициент деления входной частоты REFDIV. Значения от 1 до 63
FOUTEN	6-5	Бит разрешения выходов PLL
DSMEN	4	Бит разрешения дельта-сигма модулятора дробного делителя
DACEN	3	Бит включения DAC
BYP	2-1	Бит разрешения сквозного прохождения FREF на выход PLL
PLLEN	0	Бит включения PLL
–	31	Зарезервировано

PLLSYSCFG1 – регистр 1 конфигурации PLL

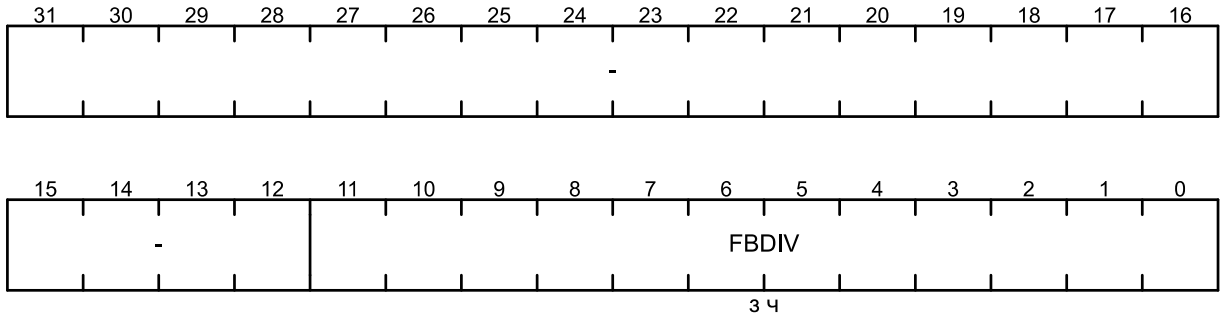
Смещение: + 54h
Сброс: 0h



Поле	Биты	Описание
FRAC	23-0	Значение дробного делителя
–	31-24	Зарезервировано

PLLSYSCFG2 – регистр 2 конфигурации PLL

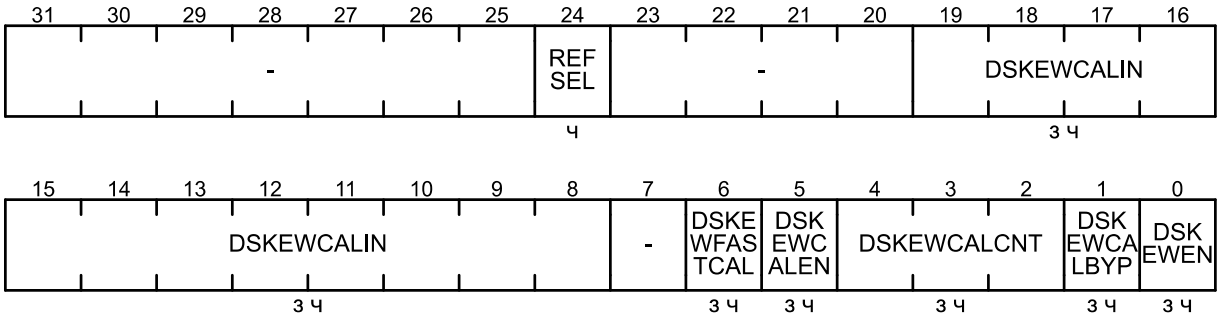
Смещение: + 58h
Сброс: 0h



Поле	Биты	Описание
FBDIV	11-0	Значение делителя
–	31-12	Зарезервировано

PLLSYSCFG3 – регистр 3 конфигурации PLL

Смещение: + 5Ch
Сброс: 0h

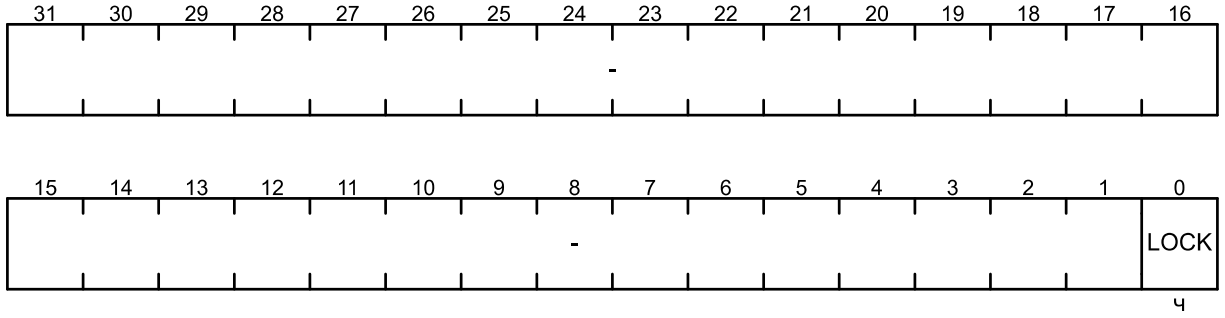


Поле	Биты	Описание	
REFSEL	24	Бит выбора источника тактирования для PLL	
		0	REFCLK
		1	SRCCLK
DSKEWCALIN	19-8	В данном поле содержится целое число со знаком, положительные значения которого задерживают сброс более быстрого пути, а отрицательные значения – сброс более медленного пути. 0 – минимальное значение, при котором каждый отсчет увеличивает время сброса на одну буферную задержку. Если бит DSKEWCALEN установлен, это значение может быть использовано для принудительной коррекции значения фазового сдвига на основе предыдущих показаний DSKEWCALOUT. Если бит DSKEWCALBYP установлен, то это значение вводится непосредственно в логику калибровки. Если бит DSKEWCALBYP сброшен, тогда это значение является начальным состоянием для калибровочной последовательности.	
DSKEWFASTCAL	6	Бит включения быстрой калибровки компенсации фазового сдвига	
		0	Для нормальной работы должно быть установлено нулевое значение
		1	Устанавливается для первичной калибровки, если начальное значение еще не известно
DSKEWCALEN	5	Включить калибровку компенсации фазового сдвига для динамичной подстройки отклонений входного сигнала.	
		0	Калибровка компенсации фазового сдвига отключена. Статическое смещение фазы определяется только путем аналогового сопоставления.
		1	Калибровка компенсации фазового сдвига включена. Статическое смещение фазы регулируется путем измерения фазы на входе.
DSKEWCALCNT	4-2	Программируемый счетчик для циклической калибровки компенсации фазового сдвига.	

		В поле указывается количество фронтов PFD, ожидаемых после каждого шага калибровки. Значение счетчика определяется как $2^{DSKEWCALCNT+5}$ (например, если DSKEWCALCNT = 5, тогда цикл будет ожидать 1024 периода PFD, перед попыткой новой настройки). Значение по умолчанию – 2.	
DSKEWCALBYP	1	Бит обхода калибровки компенсации фазового сдвига	
		0	Для установки фазовой коррекции используется выход калибровки по сдвигу (при установленном бите DSKEWCALEN).
		1	Для установки фазовой коррекции используется значение поля DSKEWCALIN (при установленном бите DSKEWCALEN).
DSKEWEN	0	Бит включения компенсации фазового сдвига	
		0	Выбор внутреннего тактирования с обратной связью
		1	Выбор внешнего тактирования
–	31-25, 23-20, 7	Зарезервировано	

PLLSYSSTAT – регистр статуса PLL

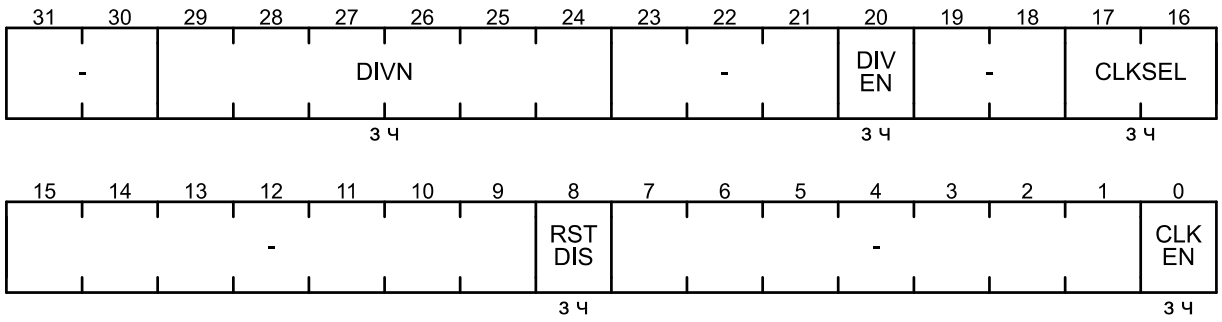
Смещение: + 60h
Сброс: 0h



Поле	Биты	Описание
LOCK	0	Устанавливается, если выходная частота блока PLLSYS стабильна
–	31-1	Зарезервировано

UARTCLKCFG – массив регистров настройки UART

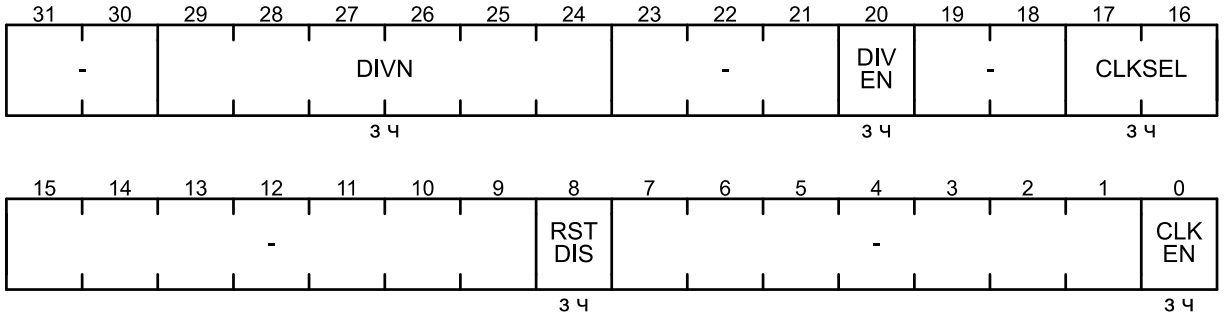
Смещение: UARTCFG + (4*n)h, где n – номер блока 0-4
Сброс: 0h



Поле	Биты	Описание	
DIVN	29-24	Коэффициент деления входного сигнала. Результирующий коэффициент деления N = 2×(DIVN+1)	
DIVEN	20	Разрешение делителя входного сигнала	
CLKSEL	17-16	Выбор источника тактового сигнала	
		00b	Сигнал HSICLK (1МГц)
		01b	Сигнал HSECLK (XTAL)
		10b	Сигнал SYSPLL0CLK
		11b	Сигнал SYSPLL1CLK
RSTDIS	8	Снятие сигнала сброса. Когда сигнал в нуле, блок в состоянии сброса	
CLKEN	0	Разрешение тактирования	
–	31-30, 23-21, 19-18, 15-9, 7-1	Зарезервировано	

QSPICLKCFG – регистр настройки QSPI

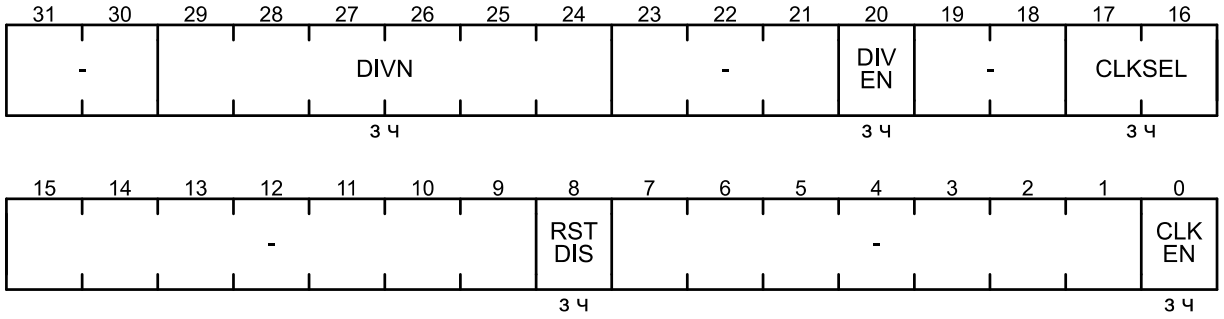
Смещение: 90h
Сброс: 0h



Поле	Биты	Описание	
DIVN	29-24	Коэффициент деления входного сигнала. Результирующий коэффициент деления N = 2 × (DIVN + 1).	
DIVEN	20	Разрешение делителя входного сигнала	
CLKSEL	17-16	Выбор источника тактового сигнала	
		00b	Сигнал HSICLK (1МГц)
		01b	Сигнал HSECLK (XTAL)
		10b	Сигнал SYSPLL0CLK
		11b	Сигнал SYSPLL1CLK
RSTDIS	8	Снятие сигнала сброса. Когда сигнал в нуле, блок в состоянии сброса	
CLKEN	0	Разрешение тактирования	
–	31-30, 23-21, 19-18, 15-9, 7-1	Зарезервировано	

SPICLKCFG – регистр настройки SPI

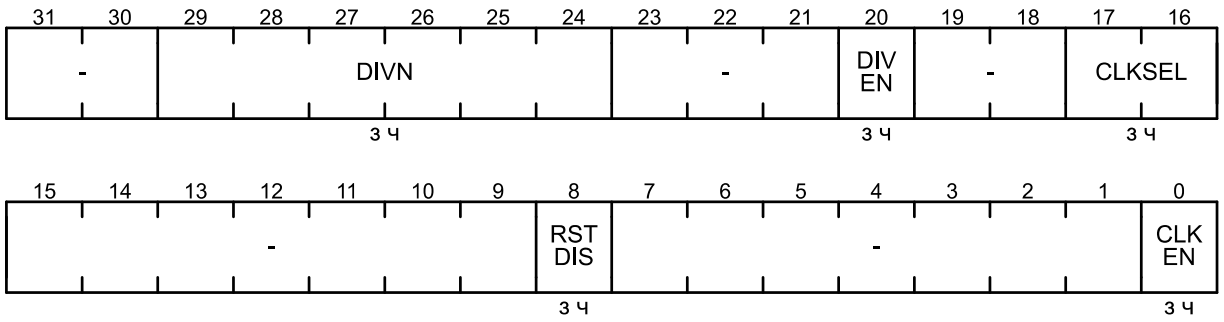
Смещение: SPICFG+ (4*n)h, где n – номер блока 0-1
Сброс: 0h



Поле	Биты	Описание	
DIVN	29-24	Коэффициент деления входного сигнала. Результирующий коэффициент деления N = 2 × (DIVN + 1).	
DIVEN	20	Разрешение делителя входного сигнала	
CLKSEL	17-16	Выбор источника тактового сигнала	
		00b	Сигнал HSICLK (1МГц)
		01b	Сигнал HSECLK (XTAL)
		10b	Сигнал SYSPLL0CLK
		11b	Сигнал SYSPLL1CLK
RSTDIS	8	Снятие сигнала сброса. Когда сигнал в нуле, блок в состоянии сброса	
CLKEN	0	Разрешение тактирования	
–	31-30, 23-21, 19-18, 15-9, 7-1	Зарезервировано	

ADCSARCLKCFG – регистр настройки ADCSAR

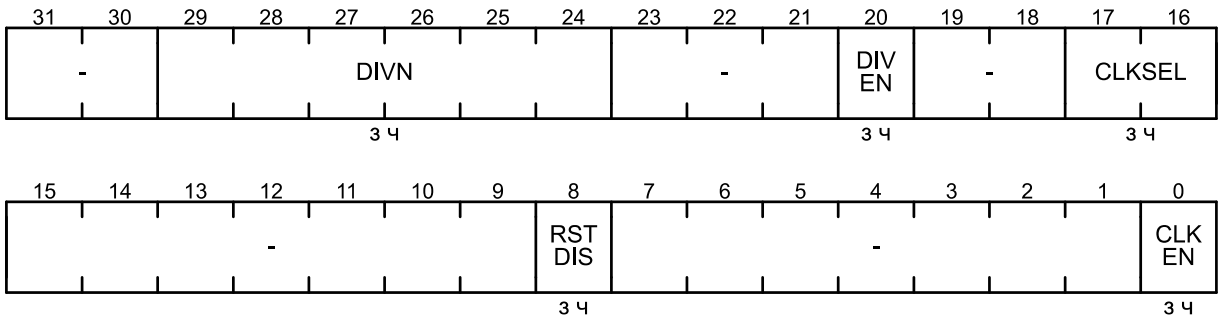
Смещение: + B0h
Сброс: 0h



Поле	Биты	Описание	
DIVN	29-24	Коэффициент деления входного сигнала. Результирующий коэффициент деления N = 2 × (DIVN + 1).	
DIVEN	20	Разрешение делителя входного сигнала	
CLKSEL	17-16	Выбор источника тактового сигнала	
		00b	Сигнал HSICLK (1МГц)
		01b	Сигнал HSECLK (XTAL)
		10b	Сигнал SYSPLL0CLK
		11b	Сигнал SYSPLL1CLK
RSTDIS	8	Снятие сигнала сброса. Когда сигнал в нуле, блок в состоянии сброса	
CLKEN	0	Разрешение тактирования	
–	31-30, 23-21, 19-18, 15-9, 7-1	Зарезервировано	

ADCSDCLKCFG – регистр настройки ADCSD

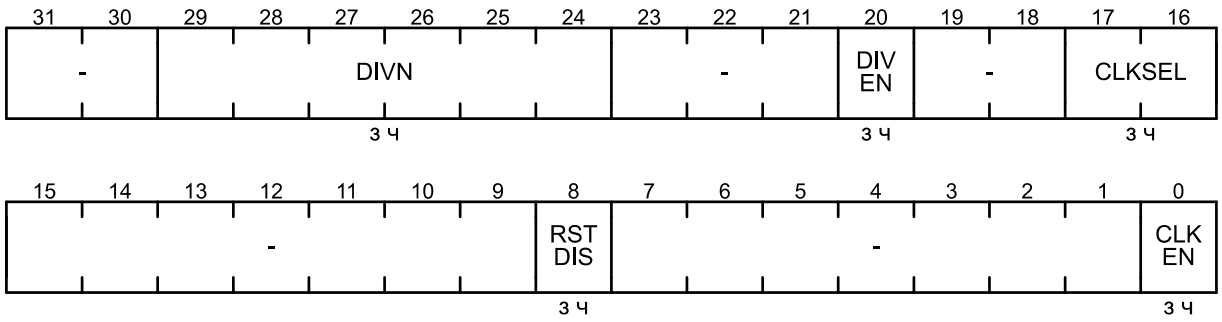
Смещение: + B4h
Сброс: 0h



Поле	Биты	Описание	
DIVN	29-24	Коэффициент деления входного сигнала. Результирующий коэффициент деления N = 2 × (DIVN + 1).	
DIVEN	20	Разрешение делителя входного сигнала	
CLKSEL	17-16	Выбор источника тактового сигнала	
		00b	Сигнал HSICLK (1МГц)
		01b	Сигнал HSECLK (XTAL)
		10b	Сигнал SYSPLL0CLK
		11b	Сигнал SYSPLL1CLK
RSTDIS	8	Снятие сигнала сброса. Когда сигнал в нуле, блок в состоянии сброса	
CLKEN	0	Разрешение тактирования	
–	31-30, 23-21, 19-18, 15-9, 7-1	Зарезервировано	

WDTCLKCFG – регистр настройки сторожевого таймера

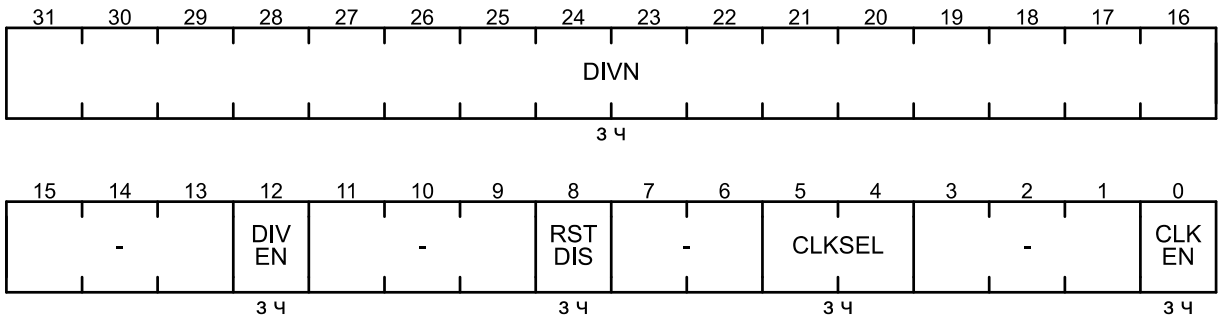
Смещение: + B8h
Сброс: 0h



Поле	Биты	Описание	
DIVN	29-24	Коэффициент деления входного сигнала. Результирующий коэффициент деления N = 2 × (DIVN + 1).	
DIVEN	20	Разрешение делителя входного сигнала	
CLKSEL	17-16	Выбор источника тактового сигнала	
		00b	Сигнал HSICLK (1МГц)
		01b	Сигнал HSECLK (XTAL)
		10b	Сигнал SYSPLL0CLK
		11b	Сигнал SYSPLL1CLK
RSTDIS	8	Снятие сигнала сброса. Когда сигнал в нуле, блок в состоянии сброса	
CLKEN	0	Разрешение тактирования	
–	31-30, 23-21, 19-18, 15-9, 7-1	Зарезервировано	

CLKOUTCFG – регистр настройки

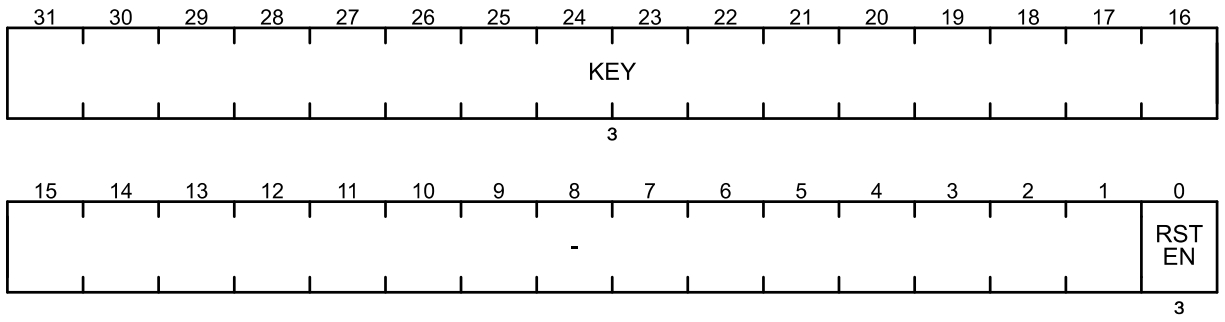
Смещение: + BCh
Сброс: 0h



Поле	Биты	Описание	
DIVN	31-16	Коэффициент деления входного сигнала. Результирующий коэффициент деления N = 2 × (DIVN + 1).	
DIVEN	12	Разрешение делителя входного сигнала	
RSTDIS	8	Снятие сигнала сброса. Когда сигнал в нуле, блок в состоянии сброса	
CLKSEL	5-4	Выбор источника тактового сигнала	
		00b	Сигнал HSICLK (1МГц)
		01b	Сигнал HSECLK (XTAL)
		10b	Сигнал SYSPLL0CLK
		11b	Сигнал LSICLK (32кГц) от блока RTC
CLKEN	0	Разрешение тактирования	
–	15-13, 11-9, 7-6, 3-1	Зарезервировано	

RSTSYS – регистр системного сброса

Смещение: + C0h
Сброс: 0h



Поле	Биты	Описание
KEY	31-16	Ключ сброса. Необходимо записывать значение A55Ah.
RSTEN	0	Разрешение системного сброса
–	15-1	Зарезервировано

A.2 Регистры блока управления энергопотреблением PMUSYS

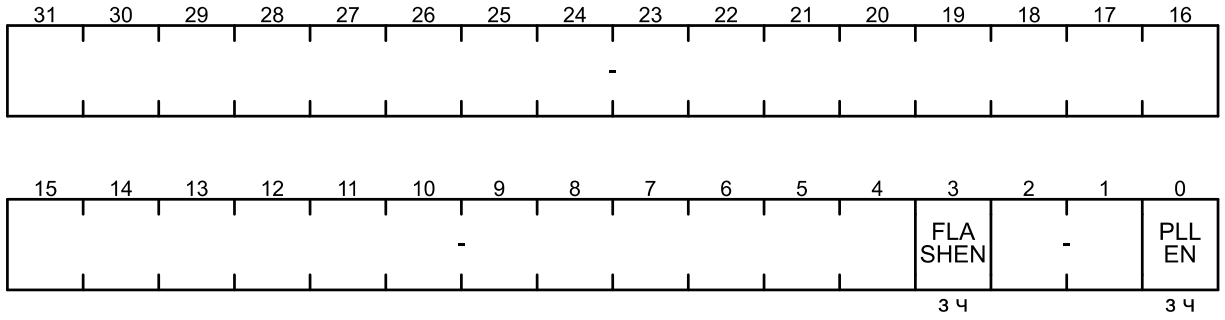
Базовый адрес: 3000_F000h

Смещение: + 110h (UID) Регистры уникального идентификатора

Примечание – n – номер регистра уникального идентификатора от 0 до 3.

PDEN – регистр разрешения перехода в PowerDown по команде WFI

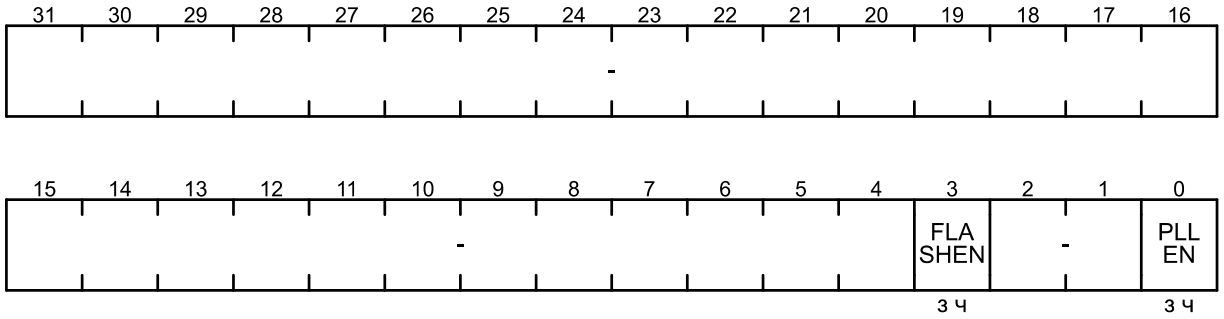
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание
FLASHEN	3	Бит разрешения перехода в режим Powerdown для Flash при выполнении команды WFI
PLLEN	0	Бит разрешения перехода в режим Powerdown для PLL при выполнении команды WFI
–	31- 4, 2 - 1	Зарезервировано

PDENFORCE – регистр управления принудительным режимом Powerdown блоков

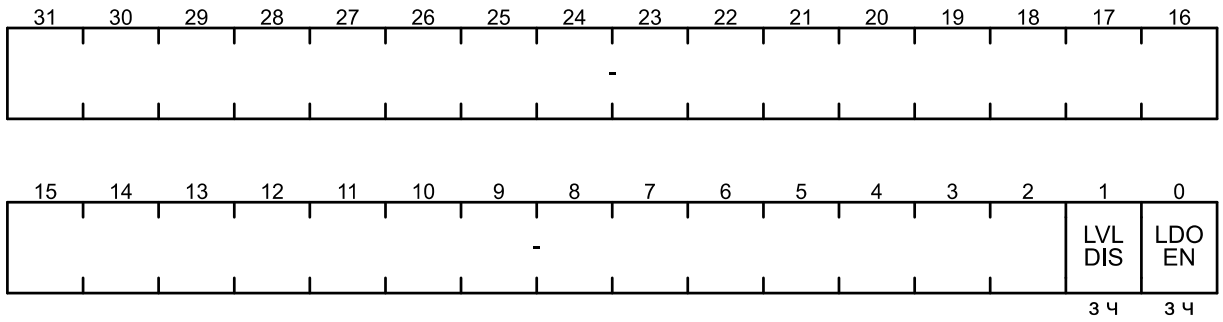
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание
FLASHEN	3	Бит принудительного включения режима Powerdown для Flash
PLLEN	0	Бит принудительного включения режима Powerdown для PLL
–	31- 4, 2 - 1	Зарезервировано

ADCPWRCFG – регистр управления питанием АЦП SAR

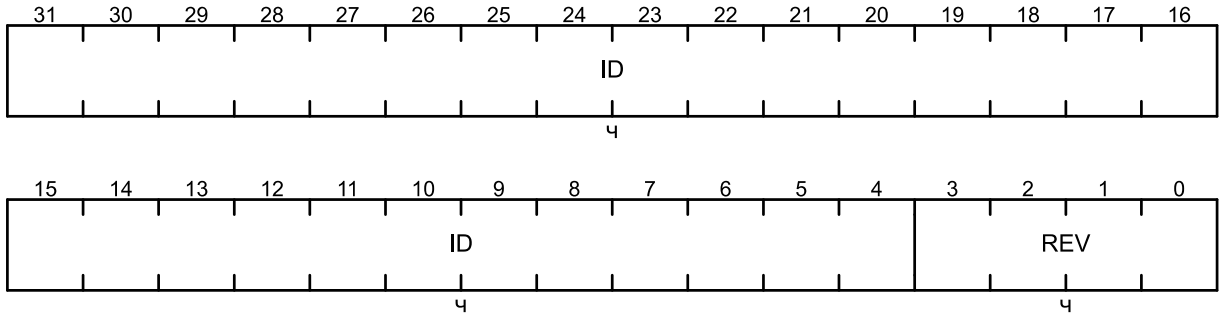
Смещение: + 10h
Сброс: 03h



Поле	Биты	Описание
LVLDIS	1	Бит выключения АЦП Level-Shifter Примечание – Level-Shifter используется для согласования логических уровней между доменами питания ядра и аналоговых блоков.
LDOEN	0	Бит включения LDO ADCSAR
–	31-2	Зарезервировано

CHIPID – регистр идентификации системы

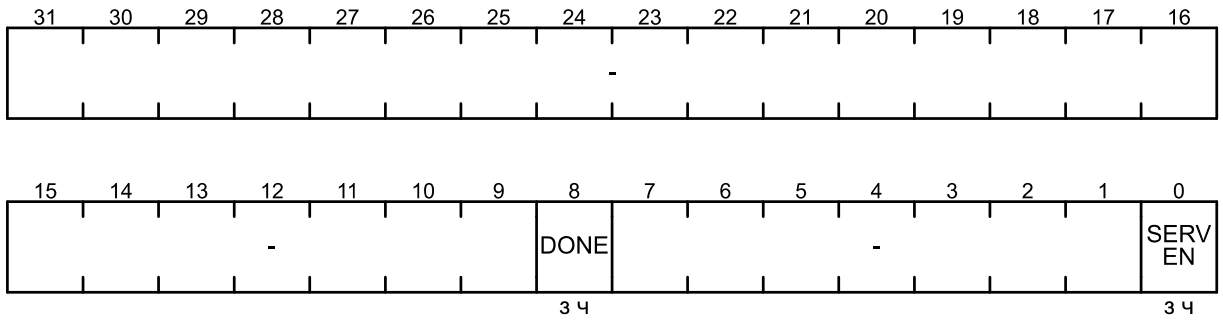
Смещение: + 100h
Сброс: DEAD_BEE1h



Поле	Биты	Описание
ID	31-4	Номер модели
REV	3-0	Номер ревизии

SERVCTL – регистр настройки сервисного режима

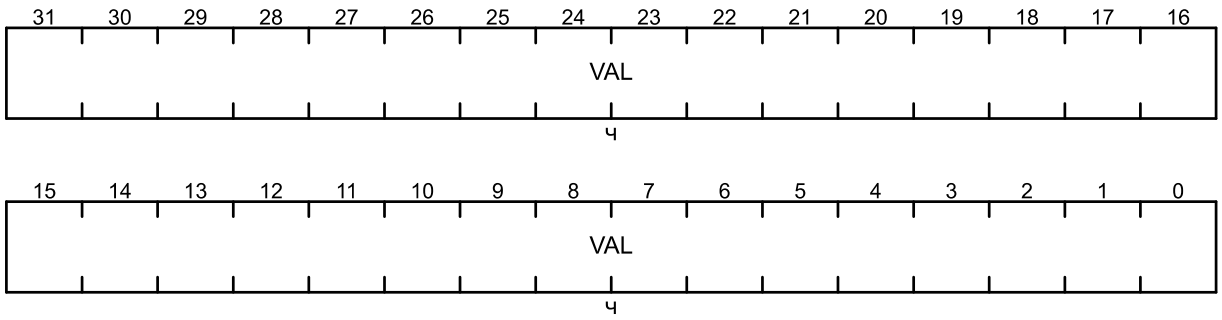
Смещение: + 104h
Сброс: 0h



Поле	Биты	Описание	
DONE	8	Флаг завершения команды сервисного стирания	
		0	Команда не завершена
		1	Команда завершена
		Запись единицы активирует полное стирание всей Flash- памяти микроконтроллера	
SERVEN	0	Бит статуса сервисного режима /Бит стирания.	
		0	Обычный режим работы
		1	Сервисный режим. Во время сброса на выводе SERVEN была единица
–	31-9, 7-1	Зарезервировано	

UID – массив регистров уникального 128-битного идентификатора

Смещение: UID + (4*n)h
Сброс: XXXX_XXXXh



Поле	Биты	Описание
UID	31-0	Уникальный 128-битный идентификатор микросхемы

A.3 Регистры блока управления энергопотреблением PMURTC

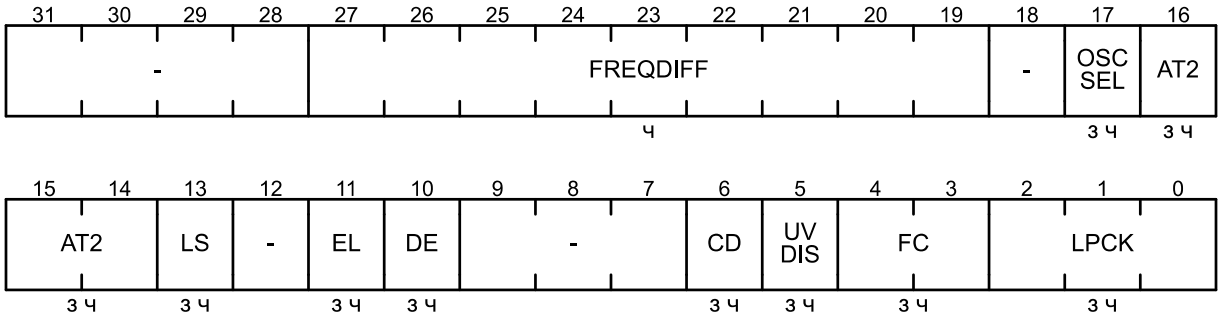
Базовый адрес: 3801_1000h

Смещение: + 020h (REG) Массив регистров пользователя

Примечание – n – номер регистра 0 – 15.

RTC_CFG0 – регистр 0 настройки RTC

Смещение: + 00h
Сброс: F0002280h

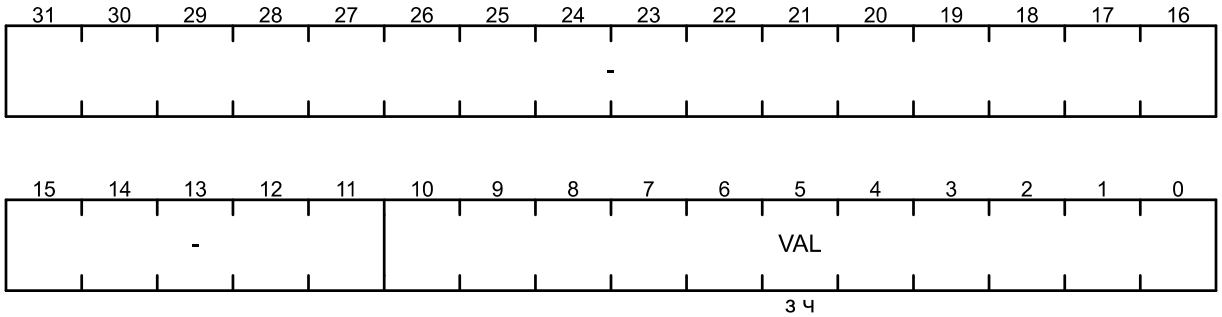


Поле	Биты	Описание
FREQDIFF	27-19	Разность количества тактов второго механизма защиты от несанкционированного доступа AT2 (вход защиты от вскрытия 2) относительно ожидаемого значения
OSCSEL	17	Выбор в качестве источника тактирования кварцевого резонатора и отключение RC-генератора (за исключением случая, когда активен второй механизм защиты от несанкционированного доступа)
	0	RC-генератор
	1	Кварцевый резонатор
AT2	16-14	Предел отклонения частоты следования тактов второго механизма защиты от несанкционированного доступа (AT2)
	0	не контролируется
	1	до 4,7 %
	2	до 7,8 %
	3	до 10,9 %
	4	до 14,1 %
	5	до 17,2 %
	6	до 20,3 %;
	7	до 23,4 %
LS	13	Выбор длительности высокого логического уровня сигнала lpclk (Длительность влияет на время работы регуляторов LDO0, LDO1 в режиме LP и ULP). Чем меньше длительность, тем меньше собственное потребление регуляторов, но больше нестабильность по выходным напряжениям.
	0	15 мкс
	1	30 мкс
EL	11	Принудительное разрешение работы регуляторов уровня LDO0
DE	10	Разрешение выхода сигнала (clkdiv) защиты от вскрытия

CD	6	Запрещение генерации сигнала lpclk	
UVDIS	5	Запрет на обнаружение пониженного напряжения	
FC	4-3	Принудительный перевод режима энергопотребления:	
		0	работа по умолчанию
		1	принудительно перевести в низкопотребляющий режим
		2	принудительно перевести в высокопотребляющий режим
		3	принудительно перевести в высокопотребляющий режим из низкопотребляющего режима
LPCK	2-0	Выбор периода обновления компараторов в режимах низкого потребления (LP) и ультранизкого потребления (ULP). Для поддержания неизменного коэффициента заполнения сигнала периоды сокращаются вдвое, если бит LS сброшен	
		0	всегда активен
		1	122 мкс
		2	488 мкс
		3	2 мс
		4	8 мс
		5	32 мс
		6	128 мс
		7	512 мс
–	31-28, 18, 12, 9-7	Зарезервировано	

RTC_TRIMRC – регистр подстройки RC генератора

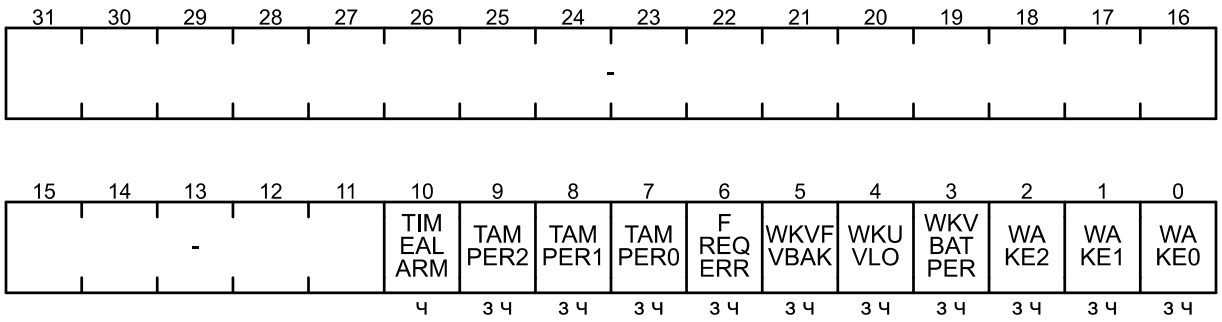
Смещение: + 04h
Сброс: FFFFF822h



Поле	Биты	Описание
VAL	10-0	Поправочное значение для корректировки частоты RC-генератора Значение 0x22 является значением по умолчанию (калибровка выключена). Соответствие значений битового поля значениям корректировки приведено в таблице 18.4.
-	31-11	Зарезервировано

RTC_HISTORY – регистр событий RTC

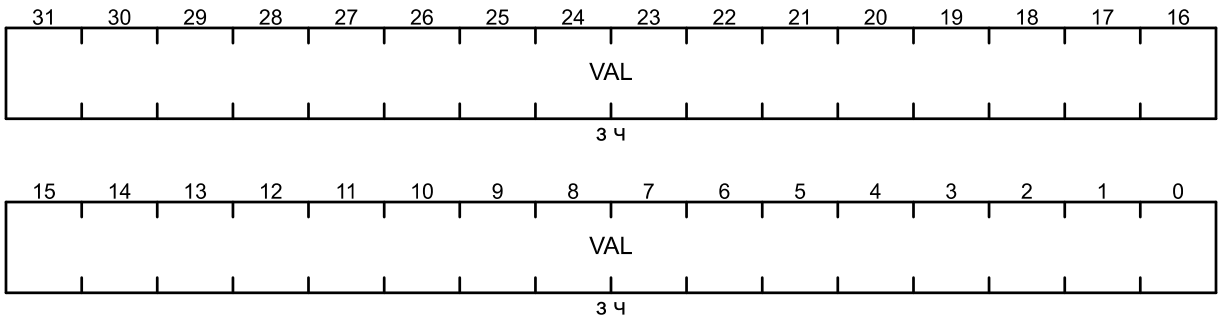
Смещение: + 08h
Сброс: FFFFFC00h



Поле	Биты	Описание
TIMEALARM	10	Сигнализация о срабатывании пробуждения. Устанавливается при равенстве значений регистров RTC_TIME и RTC_ALARM
TAMPER2	9	С помощью вывода AT_IN2 обнаружен несанкционированный доступ
TAMPER1	8	С помощью вывода AT_IN1 обнаружен несанкционированный доступ
TAMPER0	7	С помощью вывода AT_IN0 обнаружен несанкционированный доступ
FREQERR	6	Обнаружено превышение отклонения частоты на входе внешнего тактового сигнала RTC и внутреннего RC-генератора 32кГц более чем задано в битовом поле AT2 регистра RTC_CFG0
WKVFVBAK	5	Пробуждение произошло из-за события превышения напряжения на выводе V_BAT относительно напряжения на выводе VCC1 и установленного бита WAKEEN[5] регистра RTC_WAKECFG
WKUVLO	4	Пробуждение произошло из-за события понижения уровня напряжения VAON и установленного бита WAKEEN[4] регистра RTC_WAKECFG
WKVBATPER	3	Пробуждение произошло из-за одного из событий от периферийных блоков домена питания VBAT и установленного бита WAKEEN[3] регистра RTC_WAKECFG. Точная причина пробуждения будет указана в регистре PMU_WK3STAT
WAKE2	2	Пробуждение произошло из-за активного уровня сигнала на выводе WAKEUP2 и установленного бита WAKEEN[2] регистра RTC_WAKECFG
WAKE1	1	Пробуждение произошло из-за активного уровня сигнала на выводе WAKEUP1 и установленного бита WAKEEN[1] регистра RTC_WAKECFG
WAKE0	0	Пробуждение произошло из-за активного уровня сигнала на выводе WAKEUP0 и установленного бита WAKEEN[0] регистра RTC_WAKECFG
-	31-11	Зарезервировано

RTC_TIME – регистр счетчика времени RTC

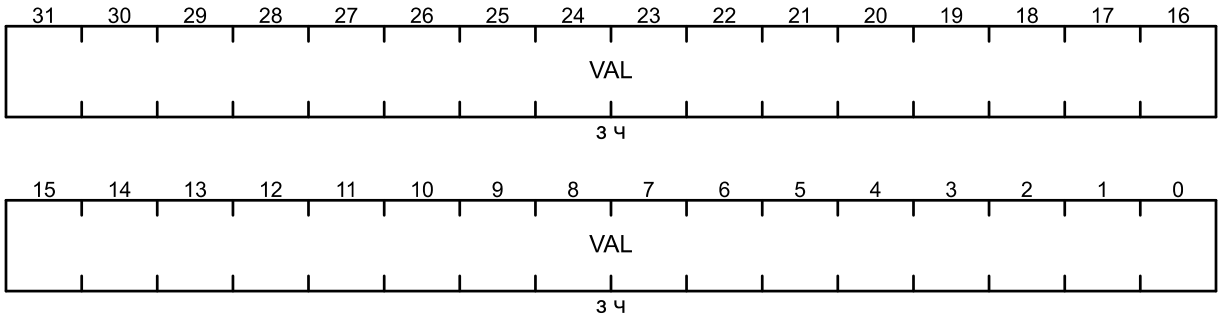
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Текущее значение таймера пробуждения

RTC_ALARM – регистр счетчика пробуждения RTC

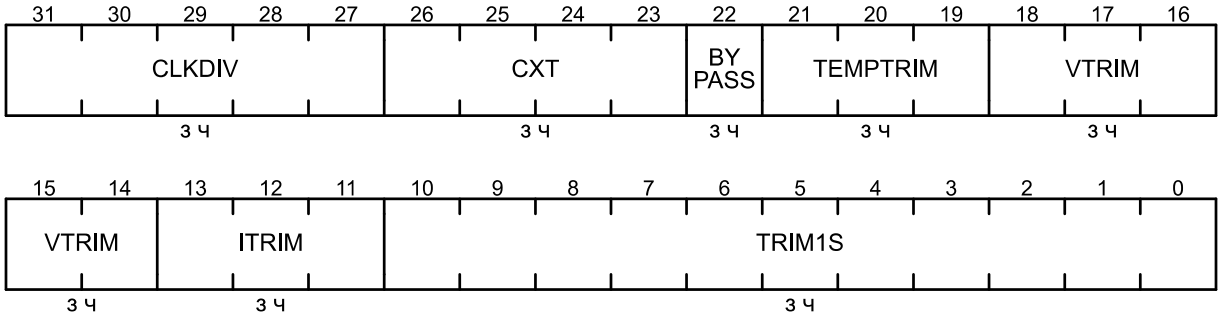
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Пороговое значение срабатывания таймера пробуждения. Если установлен бит ALARMEN регистра RTC_CFG1, то при равенстве значений регистра RTC_TIME и данного регистра сработает сигнализация и будет сгенерировано соответствующее прерывание

RTC_TRIM – регистр подстройки RTC

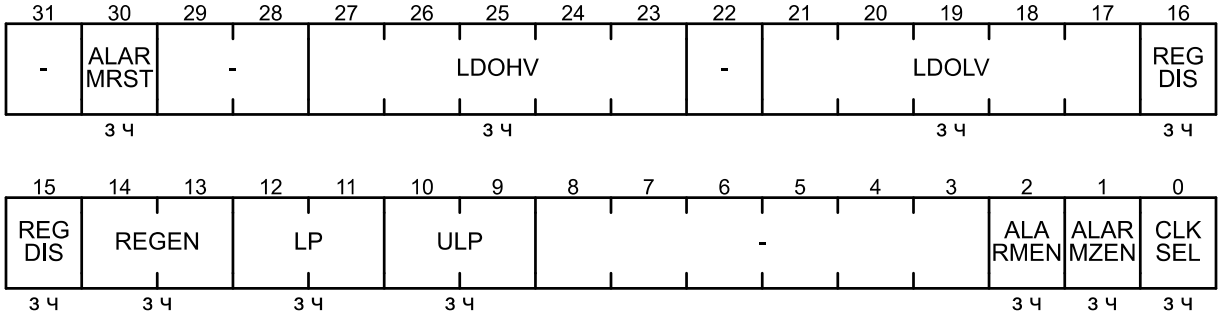
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание	
CLKDIV	31-27	Значение делителя частоты тактового сигнала (32кГц) функционала защиты от вскрытия	
CXT	26-23	Биты зарезервированы и должны иметь нулевые значения	
BYPASS	22	Режим обхода кварцевого резонатора. Установленный бит позволяет использовать вывод XI_RTC для непосредственной подачи тактового сигнала	
TEMPTRIM	21-19	Коэффициент температурной коррекции напряжения LDO_RTC	
		000	Коррекция при температуре 0 ºС
		001	Коррекция при температуре +25 ºС
		010	Коррекция при температуре +50 ºС
		011	Коррекция при температуре +75 ºС
		100	Коррекция при температуре -60 ºС
		101	Коррекция при температуре -50 ºС
		110	Коррекция при температуре -40 ºС
		111	Коррекция при температуре -25 ºС
VTRIM	18-14	Коэффициент коррекции для смещения уровня опорного напряжения LDO_RTC. Поле зарезервировано и должно иметь нулевое значение.	
ITRIM	13-11	Коэффициент коррекции для тока смещения LDO_RTC. Поле зарезервировано и должно иметь нулевое значение.	
TRIM1S	10-0	Значение увеличивает или уменьшает 1-секундный период сигнала clk1s на количество тактов частоты RTC каждые 64 секунды. Так, значение 0x7FF задает нулевую калибровку (калибровка выключена). Значение 0x3FF задает увеличение 1-секундного периода на 1024 такта частоты RTC каждые 64 секунды (+512ppm). Значение 0x400 задает уменьшение 1-секундного периода на 1023 такта частоты RTC каждые 64 секунды (-511.5ppm). Соответствие значений битового поля приведено в таблице 18.3.	

RTC_CFG1 – регистр 1 настройки RTC

Смещение: + 18h
Сброс: AD0A0117h

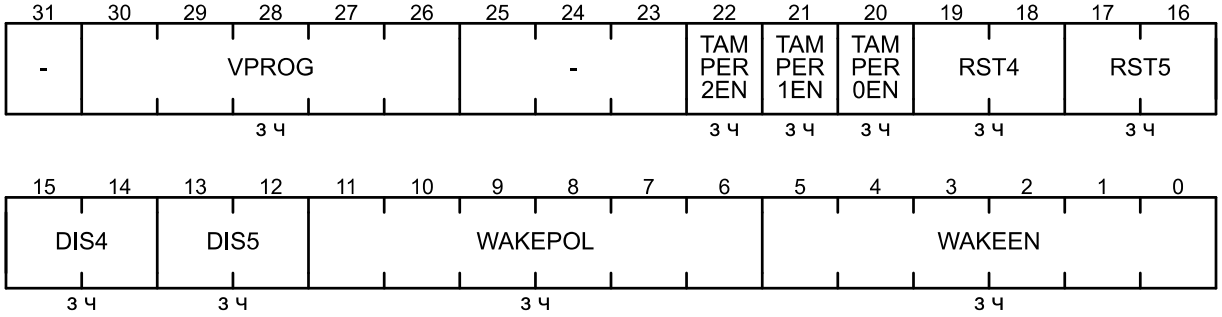


Поле	Биты	Описание
ALARMRST	30	Сброс флага сигнала пробуждения (RTC_HISTORY.TIMEALARM)
LDOHV	27-23	Верхняя граница напряжения LDO в низкопотребляющем и ультранизкопотребляющем режимах. Значение по умолчанию 1Ah соотвествует 1.32 В
LDOLV	21-17	Нижняя граница напряжения LDO в низкопотребляющем и ультранизкопотребляющем режимах. Значение по умолчанию 05h соотвествует 0.9 В
REGDIS	16-15	Когда бит REGDIS[k] установлен, регулятор k останется выключен даже при активном сигнале тревоги
REGEN	14-13	Когда бит REGEN[k] установлен, регулятор k останется включен даже при не активном сигнале тревоги
LP	12-11	Когда бит LP[k] установлен, регулятор k работает в низко потребляющем режиме
ULP	10-9	Когда бит ULP[k] установлен, регулятор k работает в ультра-низко потребляющем режиме
ALARMEN	2	Разрешение на активацию сигнала тревоги при равенстве значений в регистрах RTC_TIME и RTC_ALARM
ALARMZEN	1	Бит зарезервирован и должен иметь нулевое значение
CLKSEL	0	Выбор источника тактирования для таймера RTC_TIME
		0 – 32 КГц
		1 – 1 Гц (по умолчанию)
-	31, 29-28, 22, 8-3	Зарезервировано

RTC_WAKECFG – регистр настройки пробуждения RTC

Смещение: + 1Ch

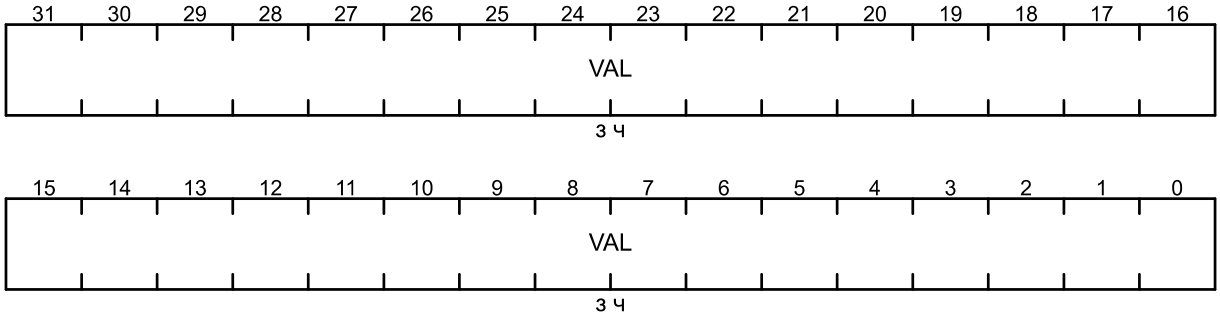
Сброс: 5380_0000h



Поле	Биты	Описание
VPROG	30-26	Значение задает напряжение обоих LDO в обычном режиме потребления. Значение по умолчанию 10100_2 соответствует напряжению 1,2 В
TAMPER2EN	22	Разрешение мониторинга вывода AT_IN2 в качестве источника пробуждения
TAMPER1EN	21	Разрешение мониторинга вывода AT_IN1 в качестве источника пробуждения
TAMPER0EN	20	Разрешение мониторинга вывода AT_IN0 в качестве источника пробуждения
RST4	19-18	Принудительный сброс домена питания LDO1, LDO0 при возникновении события WKUP[4] (пониженный уровень напряжения VAON)
RST5	17-16	Принудительный сброс домена питания LDO1, LDO0 при возникновении события WKUP[5] (превышение напряжения V_BAT относительно VCC1)
DIS4	15-14	Значение будет скопировано в RTC_CFG1.REGDIS при возникновении события WKUP[4]
DIS5	13-12	Значение будет скопировано в RTC_CFG1.REGDIS при возникновении события WKUP[5]
WAKEPOL	11-6	Когда бит WAKEPOL[k] сброшен, активным для k-го сигнала пробуждения считается высокий логический уровень. Когда бит установлен – низкий логический уровень. Соответствие индекса k сигналу приведено в описании поля WAKEEN
WAKEEN	5-0	Когда бит WAKEEN[k] сброшен соответствующее событие не будет использоваться для пробуждения: 0 – активный уровень сигнала на выводе WAKEUP0; 1 – активный уровень сигнала на выводе WAKEUP1; 2 – активный уровень сигнала на выводе WAKEUP2; 3 – событие WKUP[3] от одного из периферийных модулей домена питания VBAT (конкретный источник отображается в регистре PMU_WK3STAT); 4 – событие WKUP[4], обнаружение пониженного уровня напряжения VAON; 5 – событие WKUP[5], обнаружение превышения напряжения на выводе V_BAT относительно напряжения на выводе VCC1 (вывод №12).
–	31, 25-23	Зарезервировано

RTC_REG – массив регистров пользователя

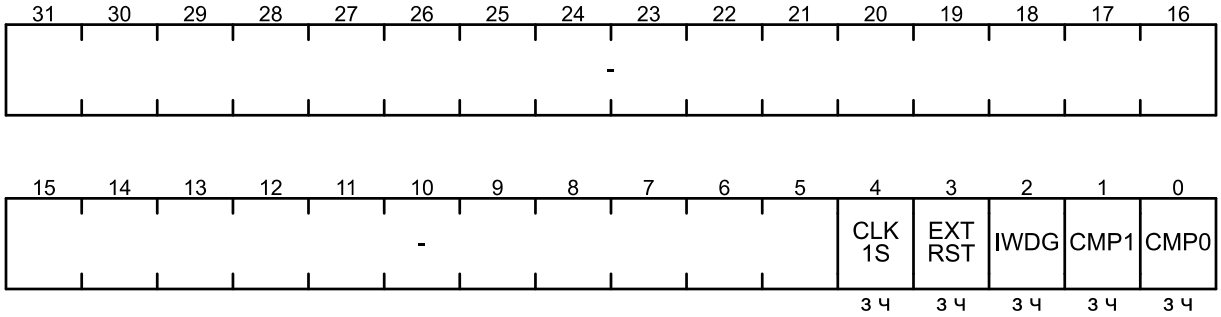
Смещение: REG + (4*n)h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение регистра n общего назначения.

PMU_WK3EN – регистр разрешения блоков для формирования сигнала события WKVBATPER

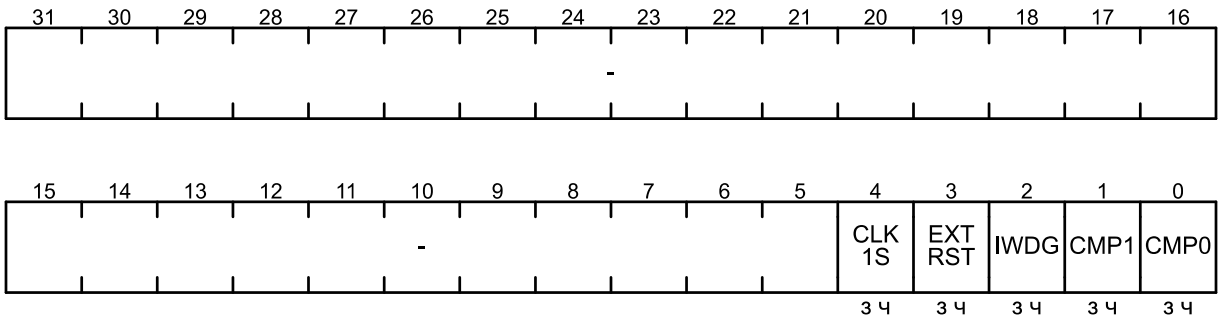
Смещение: + 80h
Сброс: 0h



Поле	Биты	Описание
CLK1S	4	Разрешение на формирование сигнала пробуждения по положительному фронту тактового сигнала CLK1S
EXTRST	3	Разрешение на формирование сигнала пробуждения по внешнему сбросу
IWDT	2	Разрешение на формирование сигнала пробуждения по сигналу прерывания от модуля IWDT
CMP1	1	Разрешение на формирование сигнала пробуждения по сигналу события от модуля CMP1
CMP0	0	Разрешение на формирование сигнала пробуждения по сигналу события от модуля CMP0
-	31-5	Зарезервировано

PMU_WK3STAT – регистр статуса блоков батарейного домена

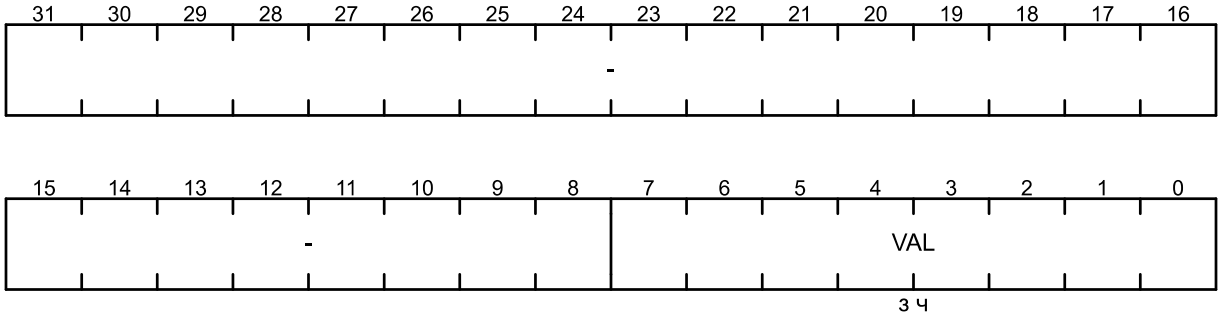
Смещение: + 84h
Сброс: 0h



Поле	Биты	Описание
CLK1S	4	Сигнал пробуждения был сформирован по положительному фронту тактового сигнала CLK1S
EXTRST	3	Сигнал пробуждения был сформирован из-за внешнего сброса
IWDT	2	Сигнал пробуждения был сформирован из-за запроса на прерывания от IWDT
CMP1	1	Сигнал пробуждения был сформирован из-за запроса на прерывания от CMP1
CMP0	0	Сигнал пробуждения был сформирован из-за запроса на прерывания от CMP0
-	31-5	Зарезервировано

PMU_WCYC – регистр длительности доступа к регистрам RTC_*

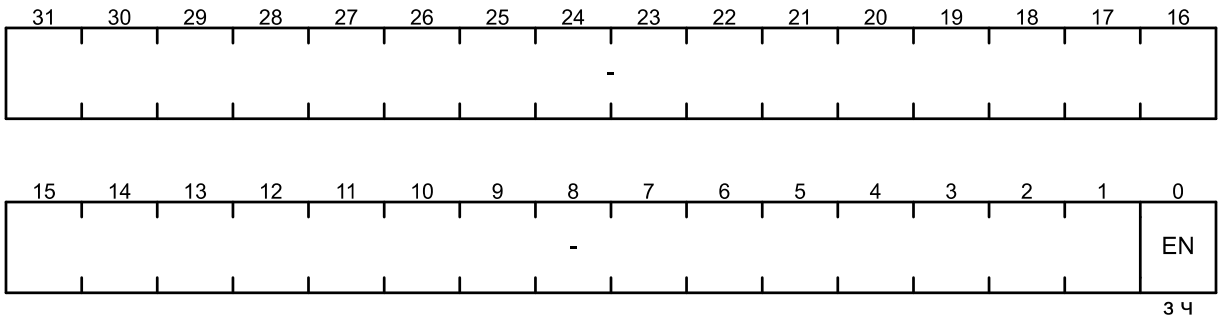
Смещение: + 88h
Сброс: 0000_0014h



Поле	Биты	Описание
VAL	7-0	Количество дополнительных тактов REFCLK, на которые будет удлинен доступ к регистрам RTC_* данного блока. Длительность обращения к регистру из данной группы должна быть не меньше (0,5 – 1) мкс. В противном случае корректность выполнения доступа не гарантируется
-	31-8	Зарезервировано

WFI_PDEN – регистр разрешения перехода в PowerDown по команде WFI

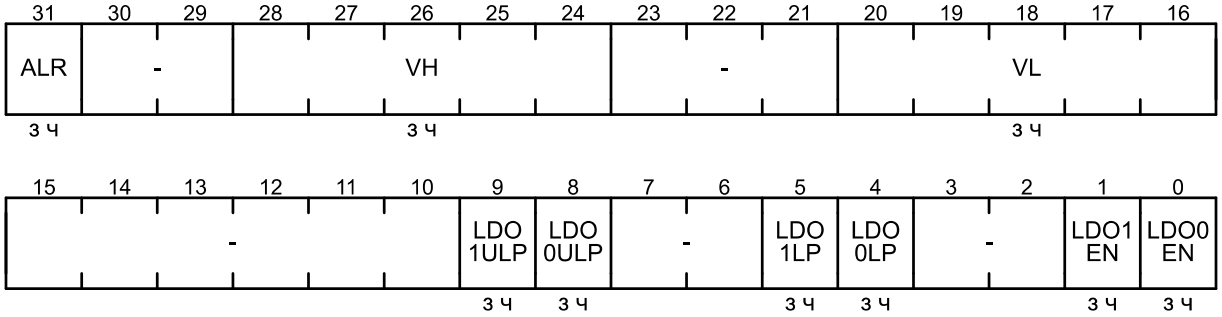
Смещение: + 8Ch
Сброс: 0h



Поле	Биты	Описание
EN	0	Бит разрешения перехода периферийных блоков в PowerDown по команде WFI. Разрешение перехода в PowerDown для каждого периферийного блока настраивается в регистрах PMU_VBATPER_WFI и PMUSYS->PDEN.
-	31-1	Зарезервировано

WFI_ENTR – регистр конфигурации регуляторов напряжения при переходе в режим ожидания (WFI)

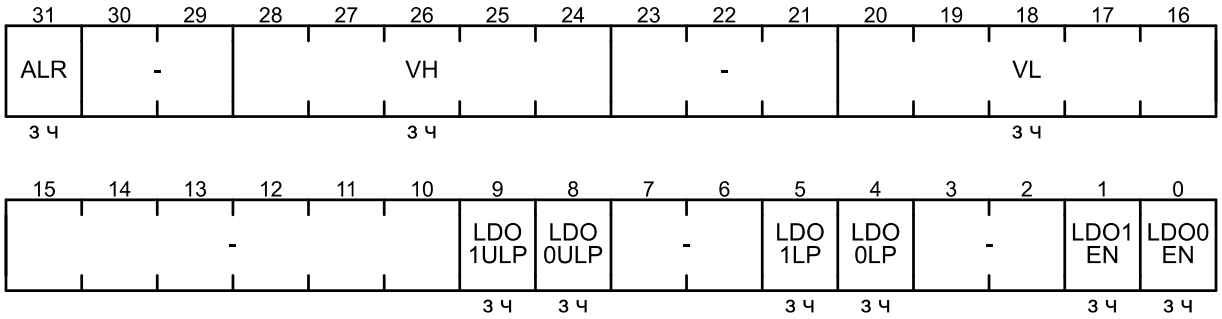
Смещение: + 90h
Сброс: 9A05_0003h



Поле	Биты	Описание
ALR	31	При переходе в режим ожидания прерывания (WFI) сбросить флаг тревожного таймера
VH	28-24	При входе в режим WFI установить верхнюю границу напряжения LDO в низкопотребляющем и ультранизкопотребляющем режимах. Значение будет записано в RTC_CFG1.LDOHV. Значение по умолчанию 1Ah соответствует 1.32 В
VL	20-16	При входе в режим WFI установить нижнюю границу напряжения LDO в низкопотребляющем и ультранизкопотребляющем режимах. Значение будет записано в RTC_CFG1.LDOLV. Значение по умолчанию 05h соответствует 0.9 В
LDO1ULP	9	При входе в режим WFI перевести LDO1 в режим ультранизкого потребления. Значение будет записано в RTC_CFG1.ULP[1]
LDO0ULP	8	При входе в режим WFI перевести LDO0 в режим ультранизкого потребления. Значение будет записано в RTC_CFG1.ULP[0]
LDO1LP	5	При входе в режим WFI перевести LDO1 в режим низкого потребления. Значение будет записано в RTC_CFG1.LP[1]
LDO0LP	4	При входе в режим WFI перевести LDO0 в режим низкого потребления. Значение будет записано в RTC_CFG1.LP[0]
LDO1EN	1	Если бит установлен, то при входе в режим WFI LDO1 останется во включенном состоянии или будет включен. Сброшенный бит соответствует отключению LDO1. Значение будет записано в RTC_CFG1.REGEN[1]
LDO0EN	0	Если бит установлен, то при входе в режим WFI LDO0 останется во включенном состоянии или будет включен. Сброшенный бит соответствует отключению LDO0. Значение будет записано в RTC_CFG1.REGEN[0]
-	30-29, 23-21, 15-10, 7-6, 3-2	Зарезервировано

WFI_EXIT – регистр конфигурации регуляторов напряжения при выходе из режима ожидания прерывания (WFI)

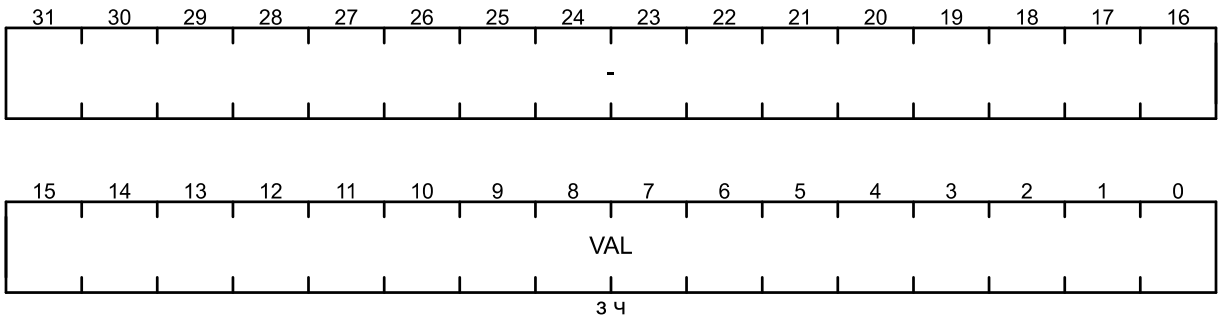
Смещение: + 94h
Сброс: 9A05_0003h



Поле	Биты	Описание
ALR	31	При выходе из режима ожидания прерывания (WFI) сбросить флаг тревожного таймера
VH	28-24	При выходе из режима WFI установить верхнюю границу напряжения LDO в низкопотребляющем и ультранизкопотребляющем режимах. Значение будет записано в RTC_CFG1.LDOHV. Значение по умолчанию 1Ah соответствует 1.32 В
VL	20-16	При выходе из режима WFI установить нижнюю границу напряжения LDO в низкопотребляющем и ультранизкопотребляющем режимах. Значение будет записано в RTC_CFG1.LDOLV. Значение по умолчанию 05h соответствует 0.9 В
LDO1ULP	9	При выходе из режима WFI перевести LDO1 в режим ультранизкого потребления. Значение будет записано в RTC_CFG1.ULP[1]
LDO0ULP	8	При выходе из режима WFI перевести LDO0 в режим ультранизкого потребления. Значение будет записано в RTC_CFG1.ULP[0]
LDO1LP	5	При выходе из режима WFI перевести LDO1 в режим низкого потребления. Значение будет записано в RTC_CFG1.LP[1]
LDO0LP	4	При выходе из режима WFI перевести LDO0 в режим низкого потребления. Значение будет записано в RTC_CFG1.LP[0]
LDO1EN	1	Если бит установлен, то при выходе из режима WFI LDO1 останется во включенном состоянии или будет включено. Сброшенный бит соответствует отключению LDO1. Значение будет записано в RTC_CFG1.REGEN[1]
LDO0EN	0	Если бит установлен, то при выходе из режима WFI LDO0 останется во включенном состоянии или будет включено. Сброшенный бит соответствует отключению LDO0. Значение будет записано в RTC_CFG1.REGEN[0]
-	30-29, 23-21, 15-10, 7-6, 3-2	Зарезервировано

WFI_DELENTR – регистр задержки при переходе в режим ожидания (WFI)

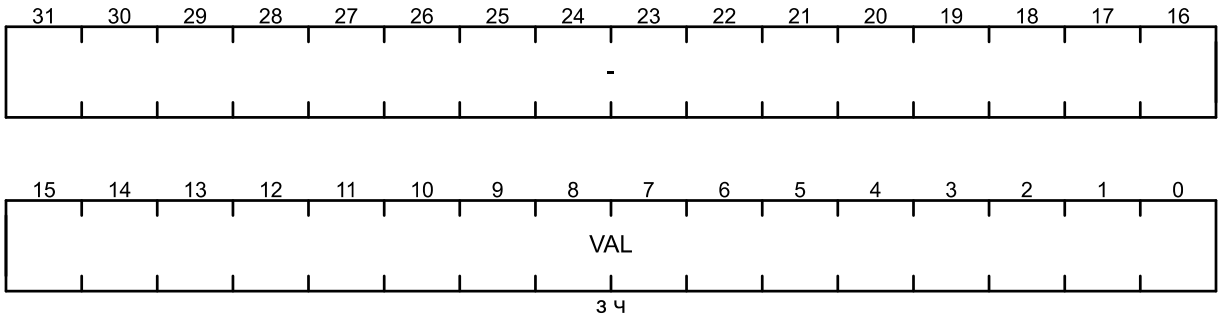
Смещение: + 98h
Сброс: 0000_0064h



Поле	Биты	Описание
VAL	15-0	Значение задержки
-	31-16	Зарезервировано

WFI_DELEXIT – регистр задержки при выходе из режима ожидания прерывания (WFI)

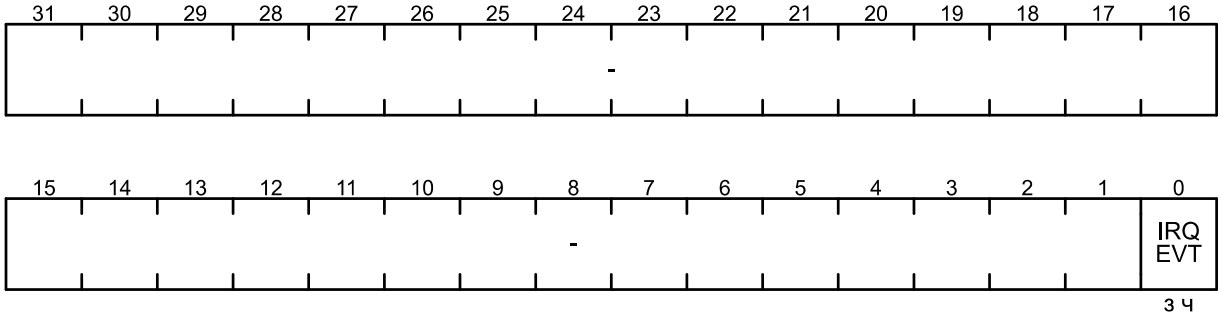
Смещение: + 9Ch
Сброс: 0000_0064h



Поле	Биты	Описание
VAL	15-0	Значение задержки
-	31-16	Зарезервировано

PMU_IRQEVT – регистр прерывания PMURTC

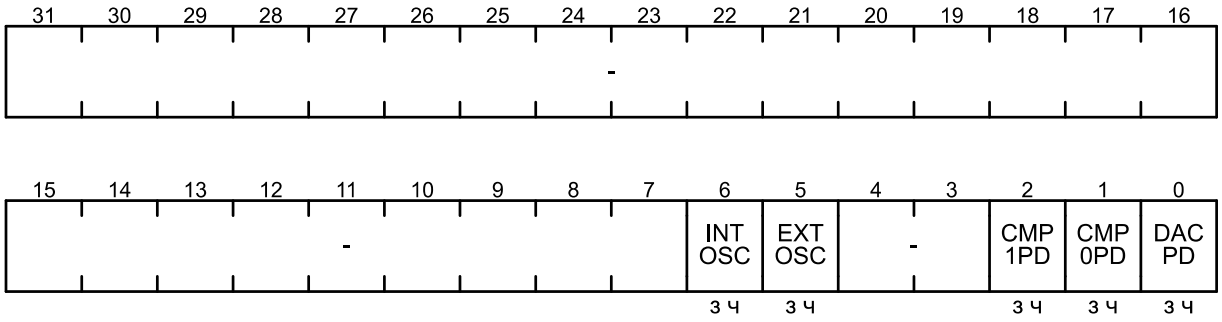
Смещение: + 0A0h
Сброс: 0000_0001h



Поле	Биты	Описание
IRQEVT	0	Бит запроса на прерывание от блока PMURTC. Запись 1 сбросит данный бит, запись 0 будет проигнорирована
-	31-1	Зарезервировано

PMU_VBATPER_FORCE – регистр переключения периферии в PowerDown

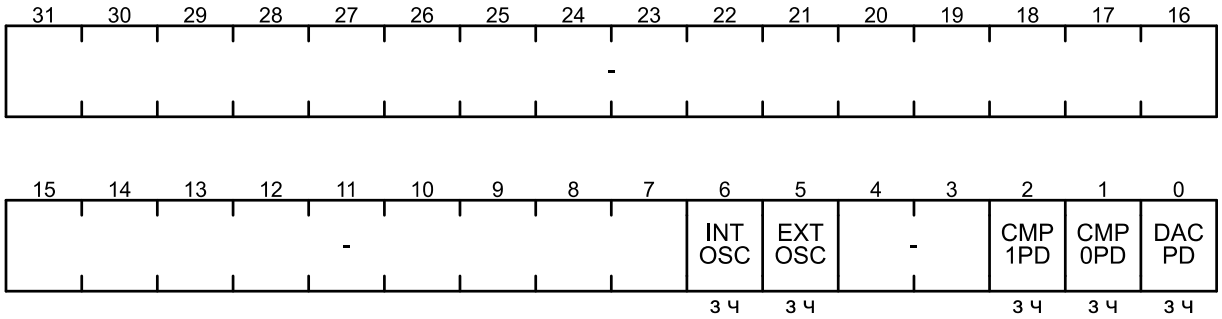
Смещение: + 0A4h
Сброс: 0h



Поле	Биты	Описание
INTOSC	6	Остановить генерацию тактового сигнала внутреннего RC-генератора (HSI)
EXTOSC	5	Остановить генерацию тактового сигнала внешнего кварцевого резонатора (HSE)
CMP1PD	2	Перевести блок CMP1 в режим пониженного потребления
CMP0PD	1	Перевести блок CMP0 в режим пониженного потребления
DACPD	0	Перевести блок DAC компараторов в режим пониженного потребления
-	31-18, 15-7	Зарезервировано

PMU_VBATPER_WFI – регистр переключения периферии в PowerDown по команде WFI

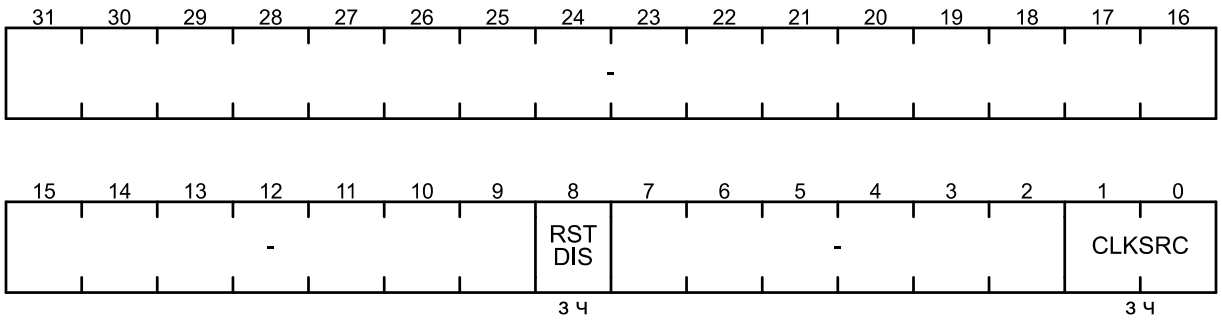
Смещение: + 0A8h
Сброс: 0h



Поле	Биты	Описание
INTOSC	17	Остановить генерацию тактового сигнала внутреннего RC-генератора (HSI) при выполнении команды WFI
EXTOSC	16	Остановить генерацию тактового сигнала внешнего кварцевого резонатора (HSE) при выполнении команды WFI
CMP1PD	2	Перевести блок CMP1 в режим пониженного потребления при выполнении команды WFI
CMP0PD	1	Перевести блок CMP0 в режим пониженного потребления при выполнении команды WFI
DACPD	0	Перевести блок DAC в режим пониженного потребления при выполнении команды WFI
-	31-18, 15-7	Зарезервировано

IWDTCFG – регистр настройки сторожевого таймера IWDT

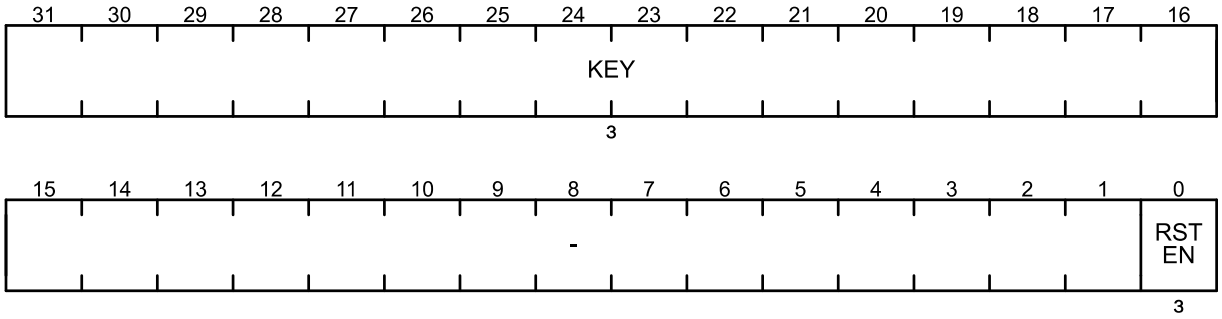
Смещение: + 0ACh
Сброс: 0h



Поле	Биты	Описание	
RSTDIS	8	Снятие сигнала сброса. Когда сигнал в нуле, блок в состоянии сброса	
CLKSRC	1-0	Выбор источника тактового сигнала	
		0	Сигнал HSICLK (1МГц)
		1	Сигнал HSECLK (XTAL)
		2	Сигнал LSICLK от блока RTC
-	31-9, 7-2	Зарезервировано	

VBATRST – регистр управления сбросом домена VBAT

Смещение: + 0C0h
Сброс: 0h



Поле	Биты	Описание
KEY	31-16	Код запуска команды сброса. Для активации команды сброса необходимо записать в поле KEY значение C0DEh.
RSTEN	0	Бит активации сброса блоков IWDT и CMP домена VBAT
-	15-1	Зарезервировано

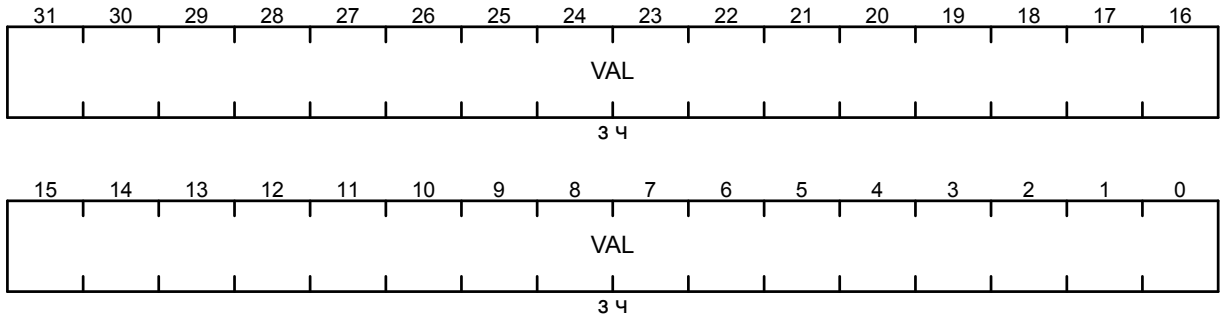
A.4 Регистры контроллера Flash-памяти

Базовый адрес: 3000_D000h

Смещение: + 04h (DATA) Регистры данных

ADDR – регистр адреса Flash-памяти

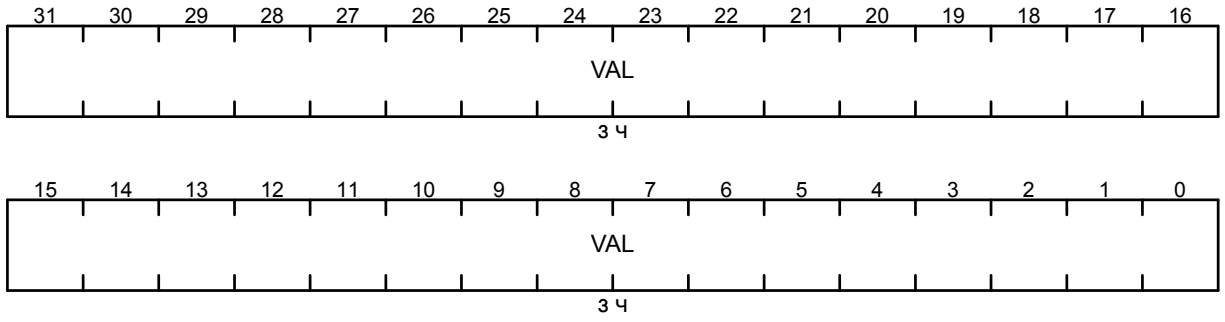
Смещение: + 0h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Адрес, используемый при командах записи, чтения и постраничного стирания. Должен быть выровнен по 16 байт. Не выровненные адреса выравниваются автоматически

DATA – массив регистров данных Flash-памяти

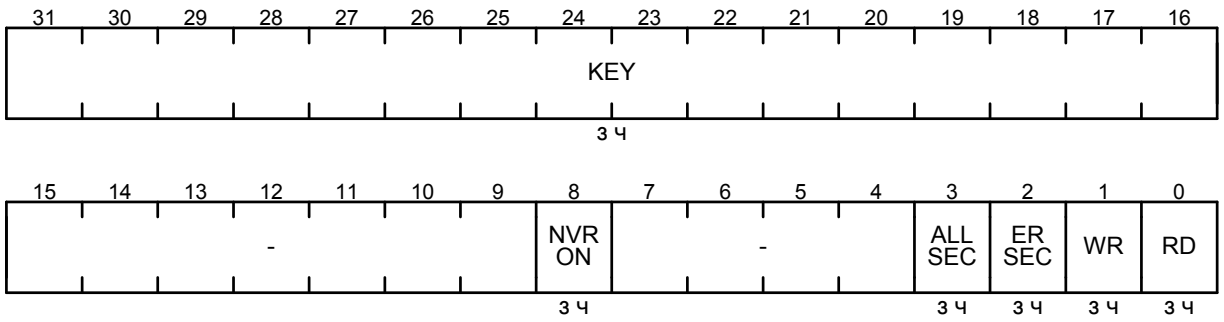
Смещение: DATA + (4*d)h, где d = 0, …, 3
Сброс: FFFF_FFFFh



Поле	Биты	Описание
VAL	31-0	32-разрядные регистры слов данных. Все слова данных должны быть загружены в регистры до установки бита команды записи. Читаемые данные будут доступны в регистрах после сброса флага BUSY

CMD – регистр команд Flash-памяти

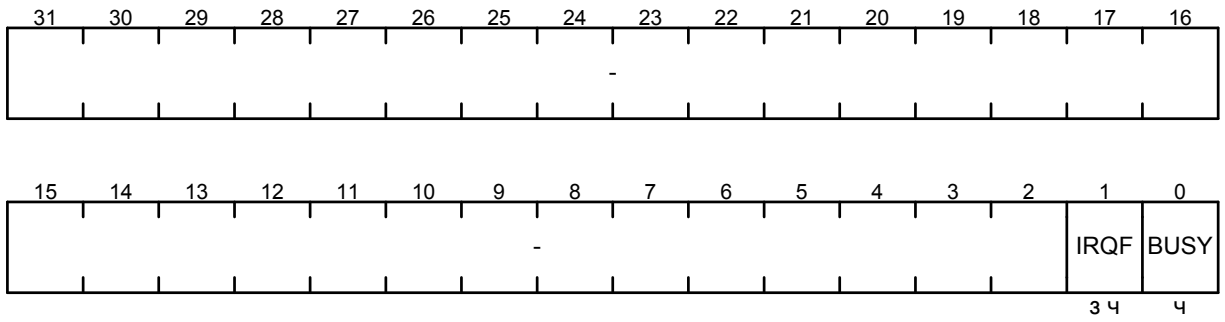
Смещение: + 44h
Сброс: DEC0_0000h



Поле	Биты	Описание
KEY	31-16	Код запуска команды. Все команды для вступления в силу должны сопровождаться записью в поле KEY значения C0DEh. Команды должны выполняться по одной, т. е. запись следующей команды разрешена, только после завершения предыдущей. При одновременной записи нескольких команд будет выполнена та, номер бита которой меньше. Чтение поля KEY всегда возвращает DEC0h
NVRON	8	Бит модификации команды для работы с NVR областью
		0 | Команда выполняется для основной области Flash-памяти
		1 | Команда выполняется для NVR области Flash-памяти.
ALLSEC	3	Бит активации команды стирания всех страниц. При установленных битах ALLSEC и ERSEC происходит полное стирание области
ERSEC	2	Бит активации команды стирания страницы области. Адрес страницы вычисляется на основе значения регистра ADDR
WR	1	Бит активации команды записи данных DATAd (d от 0 до 3), начиная с адреса ADDR в области
RD	0	Бит активации команды чтения данных в DATAd (d от 0 до 3), начиная с адреса ADDR в области
–	15-9, 7-4	Зарезервировано

STAT – регистр статуса Flash-памяти

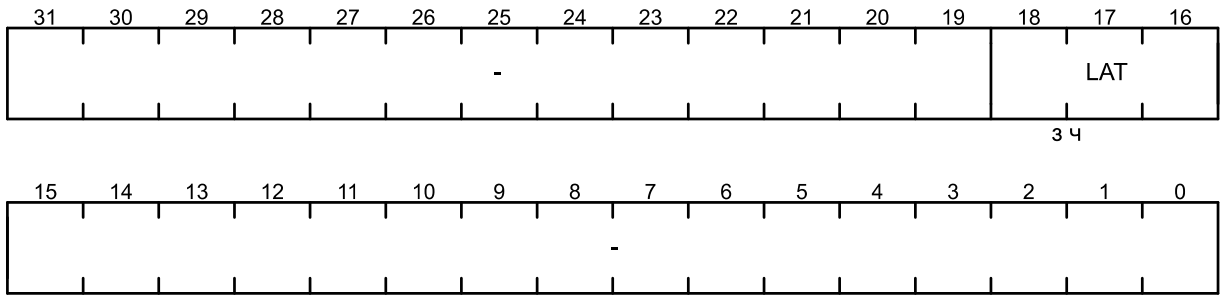
Смещение: + 48h
Сброс: 0h



Поле	Биты	Описание	
IRQF	1	Флаг прерывания по окончании выполнения команды. Устанавливается только если бит IRQEN установлен	
		0	Нет информации
		1	Команда выполнена
		Сбрасывается записью «1»	
BUSY	0	Статус работы контроллера Flash-памяти	
		0	Нет активной команды
		1	Выполняется команда
		Примечание – В связи с особенностями пересинхронизации, при работе на высоких частотах ядра необходимо добавлять задержку между записью регистра CMD и чтением флага BUSY, например, 5 NOP команд	
–	31-2	Зарезервировано	

CTRL – регистр настройки контроллера Flash-памяти

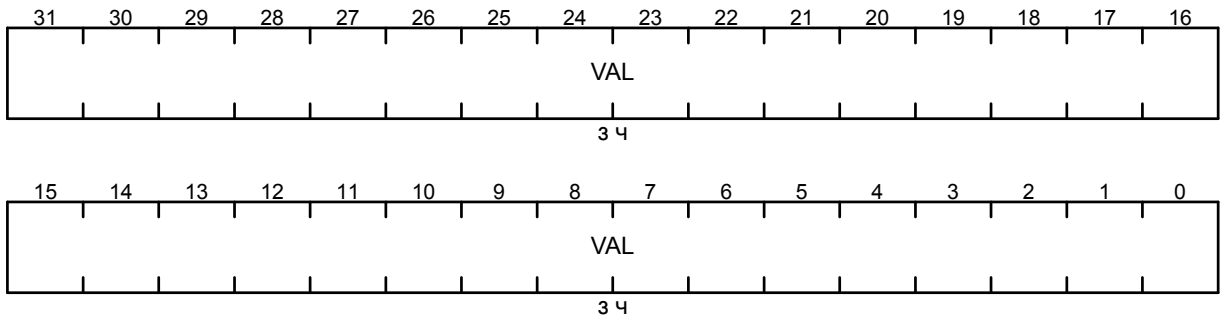
Смещение: + 4Ch
Сброс: 1_0000h



Поле	Биты	Описание
LAT	18-16	Поле задания количества дополнительных тактов ожидания при чтении из Flash-памяти
–	31-19, 15-0	Зарезервировано

TACCR – регистр установки времени доступа к Flash

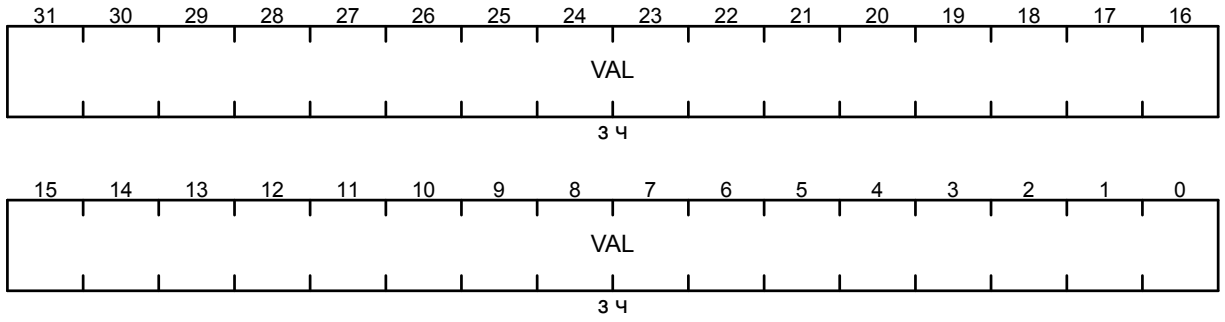
Смещение: + 1Ch
Сброс: 02h



Поле	Биты	Описание
VAL	31-0	Содержит количество циклов clk для 20 нс Запись заблокирована, когда STAT->BUSY = 1

TNVSR – регистр установки времени доступа к NVR области Flash

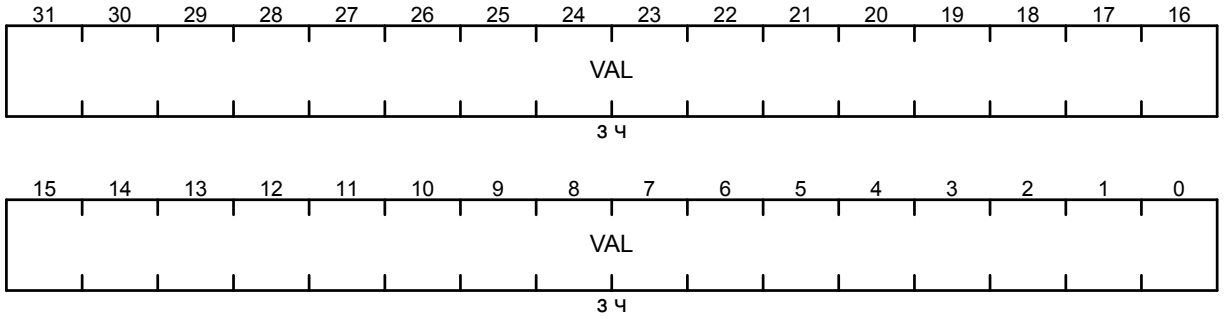
Смещение: + 20h
Сброс: 500



Поле	Биты	Описание
VAL	31-0	Содержит количество циклов clk для 5 мс Запись заблокирована, когда STAT->BUSY = 1

TERSR – регистр установки времени стирания Flash

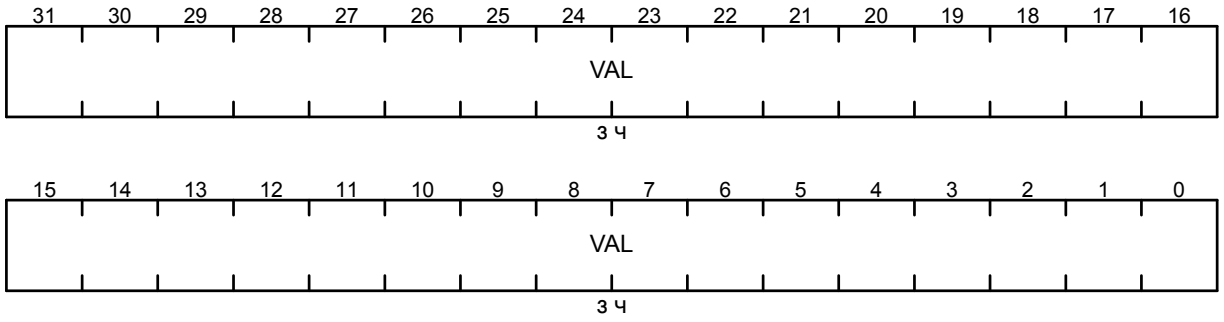
Смещение: + 24h
Сброс: 10000000



Поле	Биты	Описание
VAL	31-0	Содержит количество циклов clk для 100 мс Запись заблокирована, когда STAT->BUSY = 1

TNVHR – регистр установки времени доступа к NVR области Flash

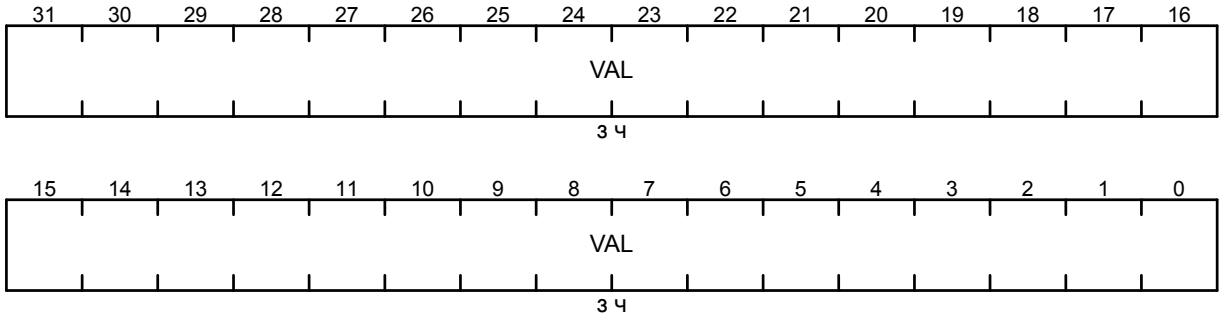
Смещение: + 28h
Сброс: 500



Поле	Биты	Описание
VAL	31-0	Содержит количество циклов clk для 5 мкс Запись заблокирована, когда STAT->BUSY = 1

TNVH1R – регистр установки времени стирания NVR области Flash

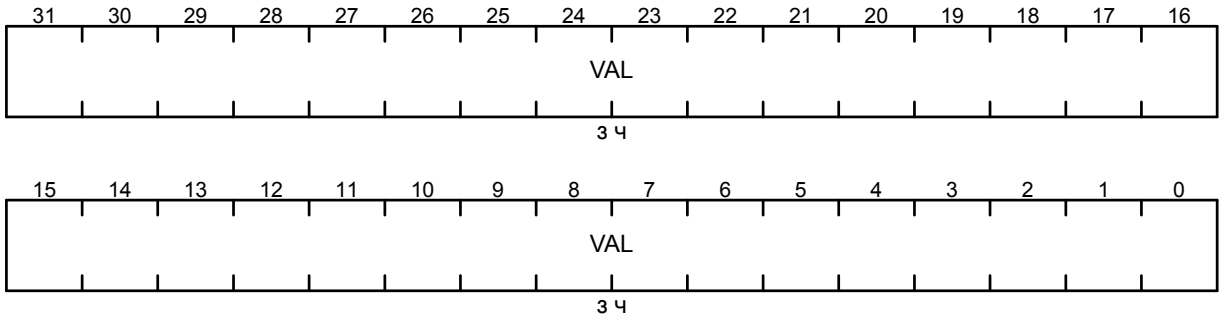
Смещение: + 2Ch
Сброс: 10000



Поле	Биты	Описание
VAL	31-0	Содержит количество циклов clk для 100 мкс Запись заблокирована, когда STAT->BUSY = 1

TRCVR – регистр установки времени восстановления Flash

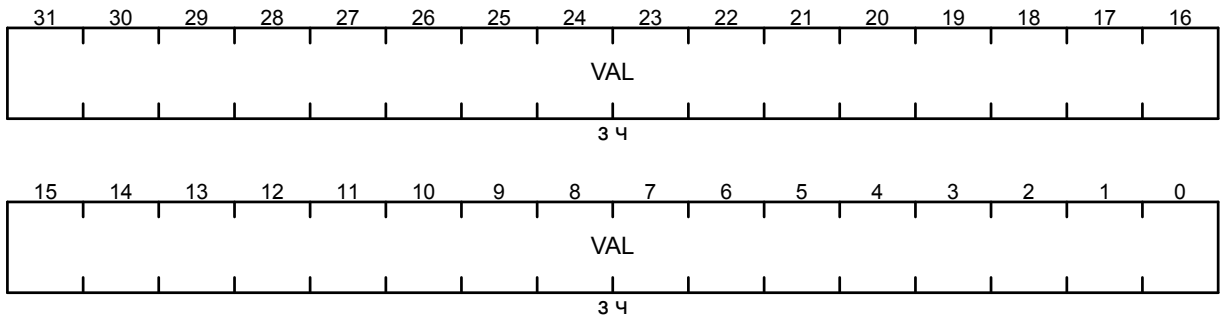
Смещение: + 30h
Сброс: 1000



Поле	Биты	Описание
VAL	31-0	Содержит количество циклов clk для 10 мкс Запись заблокирована, когда STAT->BUSY = 1

TPGSR – регистр установки времени программирования NVR области Flash

Смещение: + 34h
Сброс: 1000



Поле	Биты	Описание
VAL	31-0	Содержит количество циклов clk для 10 мкс Запись заблокирована, когда STAT->BUSY = 1

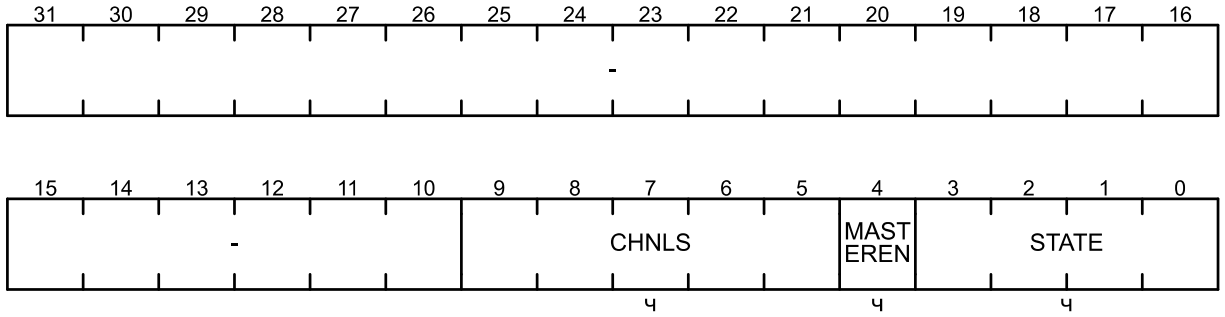
A.5 Регистры контроллера прямого доступа к памяти DMA

Базовый адрес: 3000_C000h

Примечание**:** i – порядковый номер канала от 0 до 23

STATUS – регистр статуса DMA

Смещение: + 00h
Сброс: 0000_0000h

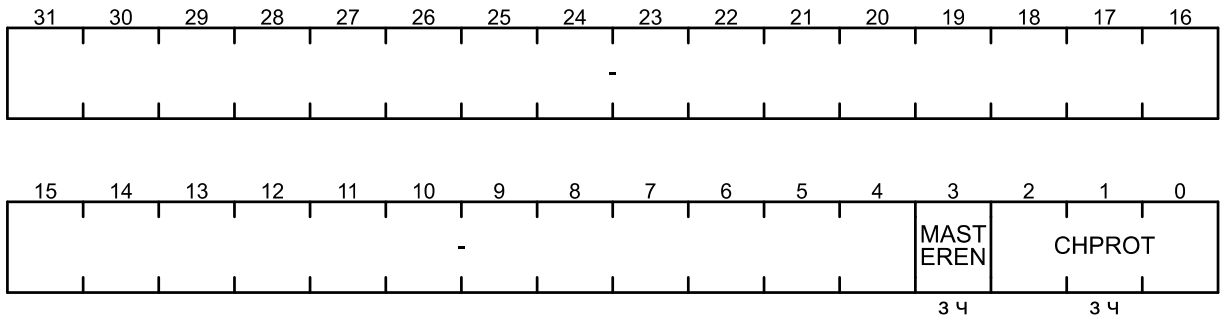


Поле	Биты	Описание	
CHNLS	5-9	Количество доступных каналов DMA	
		00h	1 канал
		01h	2 канала
		02h	3 канала
		…	…
		17h	24 канала
MASTEREN	4	Состояние контроллера DMA	
		0	Работа контроллера запрещена
		1	Работа контроллера разрешена
STATE	3-0	Текущее состояние конечного автомата управления контроллера	
		0h	В покое
		1h	Чтение управляющих данных канала
		2h	Чтение указателя конца данных источника
		3h	Чтение указателя конца данных приемника
		4h	Чтение данных источника
		5h	Запись данных в приемник
		6h	Ожидание запроса на выполнение прямого доступа
		7h	Запись управляющих данных канала
		8h	Приостановлен
		9h	Выполнен
		Ah	Режим работы с периферией «разборка-сборка»
		Bh-Fh	Зарезервировано
–	31-10	Зарезервировано	

Примечание – Регистр доступен только для чтения. Возвращает состояние контроллера DMA. Во время сброса чтение регистра запрещено.

CFG – регистр конфигурации контроллера DMA

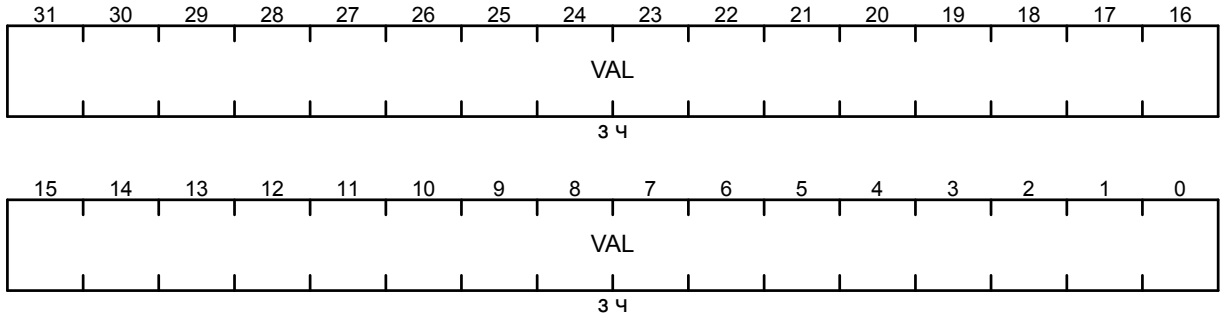
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание			
MASTEREN	3	Бит разрешения работы контролера DMA			
		0	Запрещено		
		1	Разрешено		
CHPROT	2-0	Задает параметры защиты шины AHB при обращении контроллера DMA к структурам управляющих данных каналов			
			Биты поля CHPROT		
			2	1	0
		0	Доступ не кэшируется	Доступ не буферизуется	Доступ непривилегированный
		1	Доступ кэшируется	Доступ буферизуется	Доступ привилегированный
–	31-4	Зарезервировано			

BASEPTR – регистр базового адреса управляющих данных каналов

Смещение: + 08h
Сброс: 0h

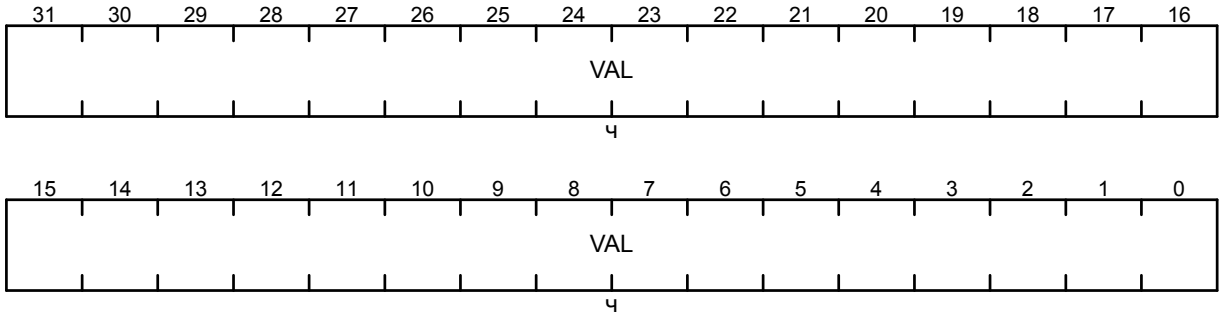


Поле	Биты	Описание
VAL	31-0	Указатель на базовый адрес первичной структуры управляющих данных

Примечание – Регистр определяет базовый адрес системной памяти размещения управляющих данных каналов. Во время сброса чтение регистра запрещено.

ALTBASEPTR – регистр базового адреса альтернативных управляющих данных каналов

Смещение: + 0Ch
Сброс: 0h

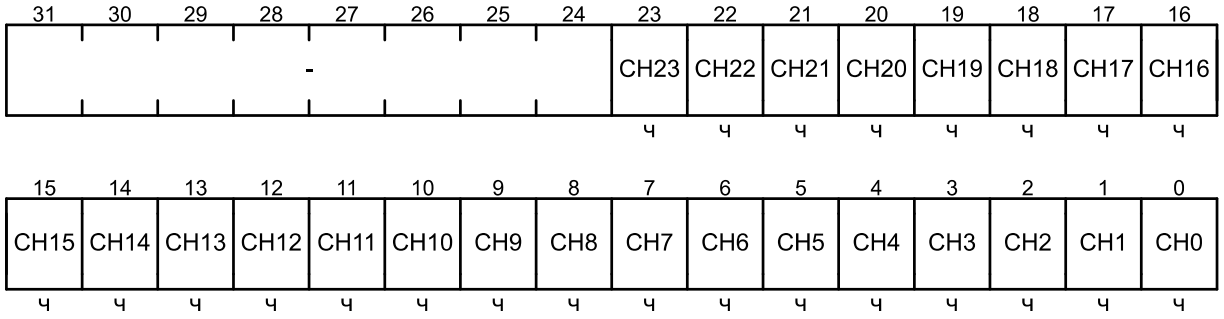


Поле	Биты	Описание
VAL	31-0	Указатель базового адреса альтернативной структуры управляющих данных каналов

Примечание – Регистр позволяет не производить вычисления базового адреса альтернативных управляющих данных каналов. Доступен только для чтения и возвращает указатель базового адреса альтернативных управляющих данных каналов. Во время сброса чтение регистра запрещено.

WAITONREQ – регистр статуса ожидания запросов для передачи

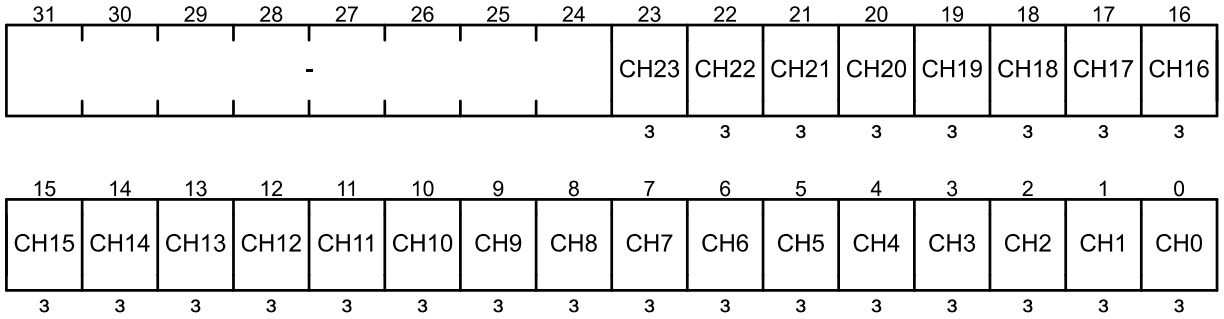
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание		
CHi	23-0	Чтение	0	Доступны только BREQ запросы от периферии
			1	Доступны BREQ и SREQ запросы от периферии
		Запись	Не выполняется	
–	31-24	Зарезервировано		

SWREQ – регистр программного запроса на обработку каналов DMA

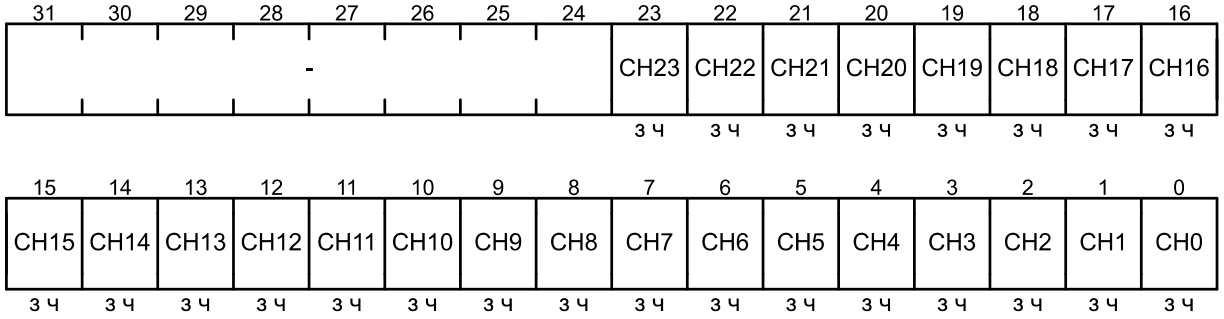
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание	
CHi	23-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Устанавливает запрос на выполнение цикла DMA по каналу i
–	31-24	Зарезервировано	

USEBURSTSET – регистр установки пакетного обмена каналов DMA

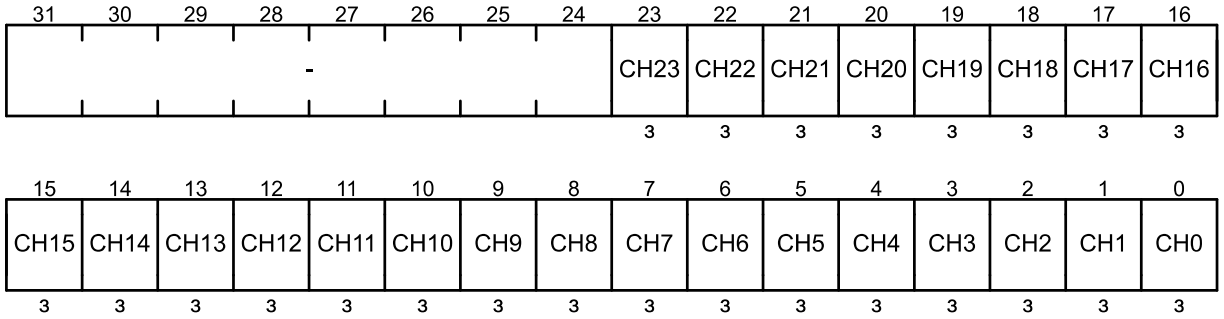
Смещение: + 18h
Сброс: 0h



Поле	Биты	Описание		
CHi	23-0	Бит отключения выполнения одиночных запросов SREQ. При активации режима будут обрабатываться и исполняться только пакетные запросы BREQ		
		Чтение	0	Выполнение циклов DMA в ответ на SREQ и BREQ
			1	Выполнение циклов DMA только в ответ на BREQ
		Запись нуля	Не выполняется	
		Запись единицы	Отключает выполнение запросов SREQ	
–	31-24	Зарезервировано		

USEBURSTCLR – регистр сброса пакетного обмена каналов DMA

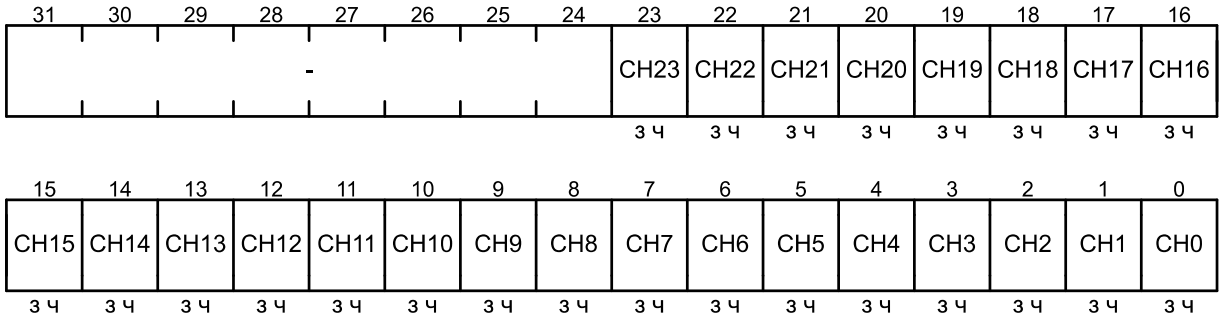
Смещение: + 1Ch
Сброс: 0h



Поле	Биты	Описание	
CHi	23-0	Чтение	Читаются нули
		Запись нуля	Не выполняется
		Запись единицы	Разрешает обрабатывать одиночные запросы SREQ на выполнение циклов DMA
–	31-24	Зарезервировано	

REQMASKSET – регистр маскирования запросов от периферии на обслуживание каналов DMA

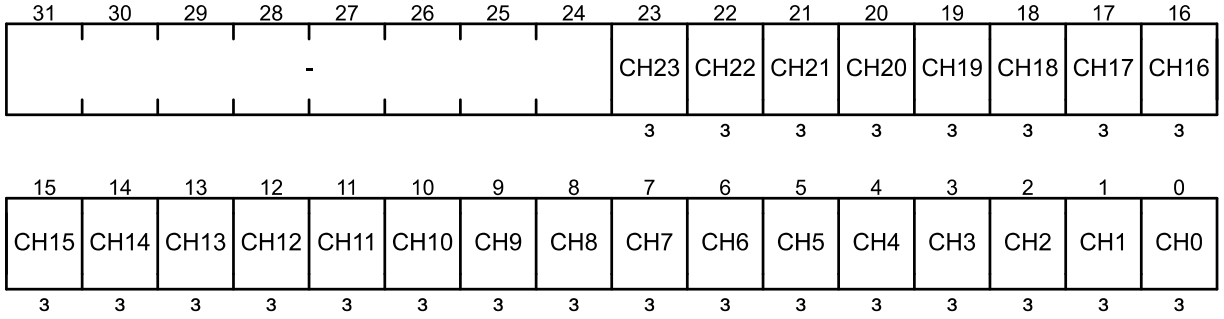
Смещение: + 20h
Сброс: 0h



Поле	Биты	Описание		
CHi	23-0	Бит отключения выполнения каналом i циклов DMA в ответ на поступающие запросы SREQ и BREQ периферии		
		Чтение	0	Канал i выполняет циклы
			1	Канал i не выполняет циклы
		Запись нуля	Не выполняется	
		Запись единицы	Отключает выполнение циклов	
–	31-24	Зарезервировано		

REQMASKCLR – регистр сброса маскирования запросов на обслуживание каналов DMA

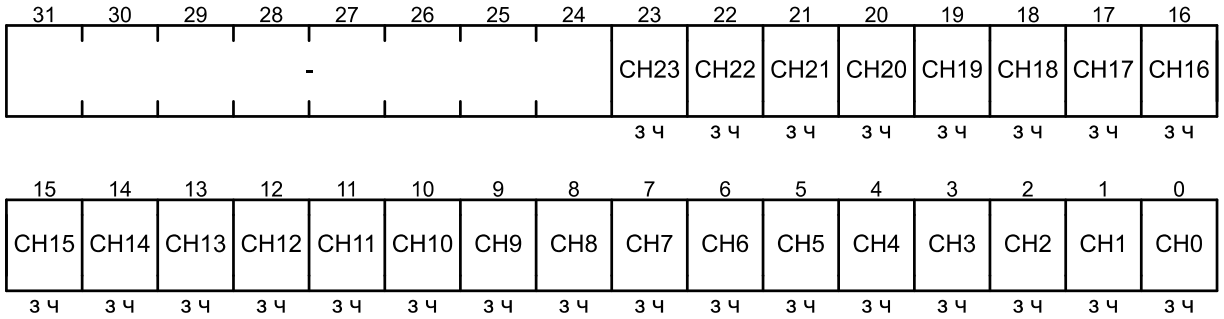
Смещение: + 24h
Сброс: 0h



Поле	Биты	Описание	
CHi	23-0	Чтение	Читаются нули
		Запись нуля	Не выполняется
		Запись единицы	Разрешает выполнение циклов DMA по запросам SREQ и BREQ периферии
–	31-24	Зарезервировано	

ENSET – регистр установки разрешения работы каналов DMA

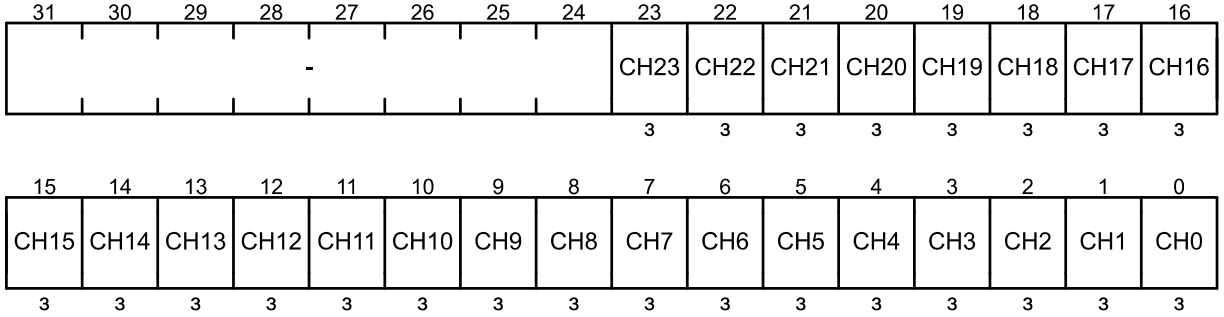
Смещение: + 28h
Сброс: 0h



Поле	Биты	Описание		
CHi	23-0	Чтение	0	Канал i отключен
			1	Работа канала i разрешена
		Запись нуля	Не выполняется	
		Запись единицы	Разрешает работу канала i	
–	31-24	Зарезервировано		

ENCLR – регистр сброса разрешения работы каналов DMA

Смещение: + 2Ch
Сброс: 0h

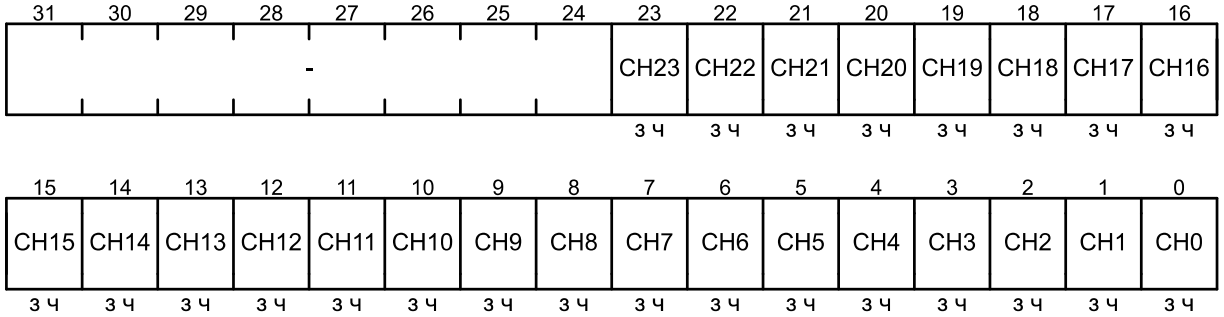


Поле	Биты	Описание	
CHi	23-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Запрещает работу канала i
–	31-24	Зарезервировано	

Примечание – Контроллер может отключить канал i в случае:
- завершения цикла DMA;
- чтения CHANNEL_CFG с полем CYCLE_CTRL, значение которого 000b;
- появления ошибки на шине AHB.

PRIALTSET – регистр установки первичной/альтернативной структуры управляющих данных каналов DMA

Смещение: + 30h
Сброс: 0h



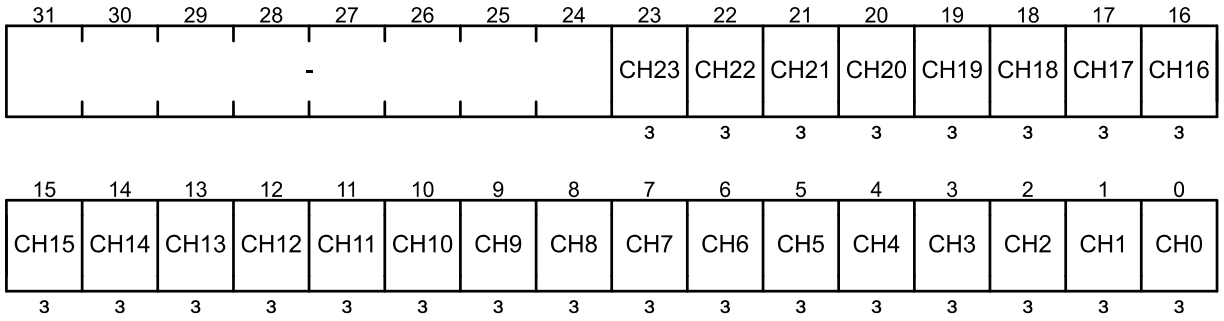
Поле	Биты	Описание		
CHi	23-0	Бит включения использования каналом i альтернативной структуры управляющих данных		
		Чтение	0	Используется первичная структура
			1	Используется альтернативная структура
		Запись нуля	Не выполняется	
		Запись единицы	Включает использование альтернативной структуры управляющих данных	
–	31-24	Зарезервировано		

Примечание – Контроллер может переключить состояние бита CHi в случае:

- завершения четырех передач DMA, указанных в первичной структуре управляющих данных при выполнении цикла DMA в режимах работы с памятью или периферией «разборка-сборка»;
- завершения всех передач DMA, указанных в первичной структуре управляющих данных при выполнении цикла DMA в режиме «пинг-понг»;
- завершения всех передач DMA, указанных в альтернативной структуре управляющих данных при выполнении цикла DMA в режимах «пинг-понг» и «разборка-сборка».

PRIALTCLR – регистр сброса первичной/альтернативной структуры управляющих данных каналов DMA

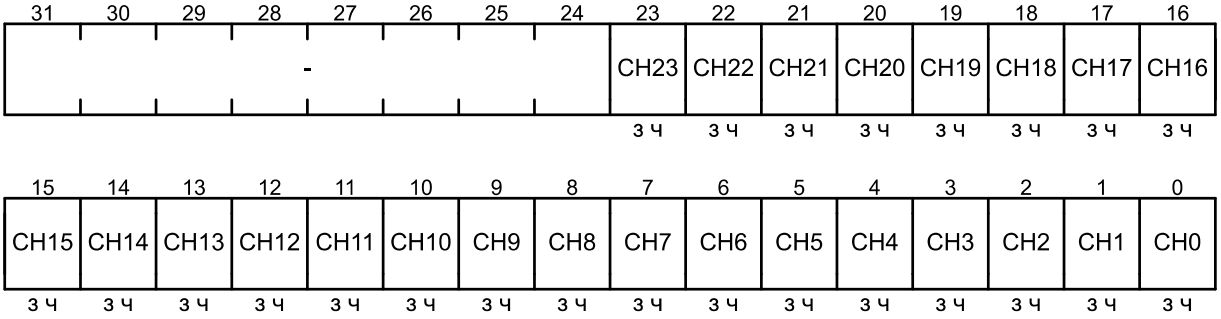
Смещение: + 34h
Сброс: 0h



Поле	Биты	Описание	
CHi	23-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Включает использование первичной структуры управляющих данных канала i
–	31-24	Зарезервировано	

PRIORITYSET – регистр установки приоритета каналов DMA

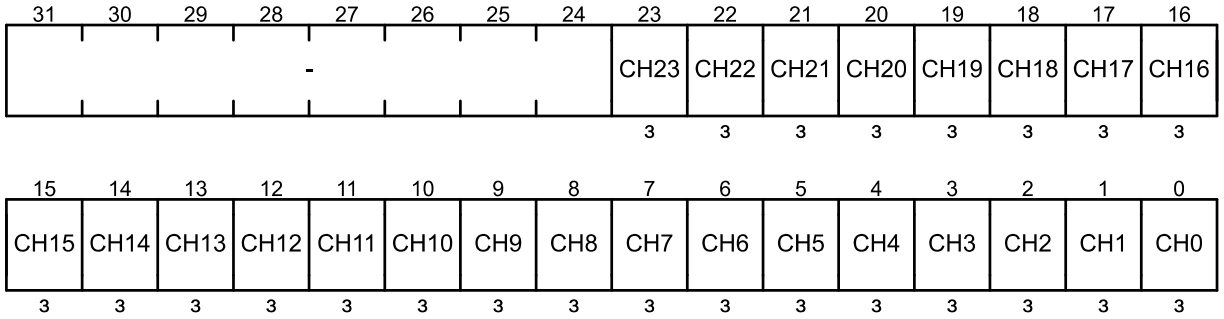
Смещение: + 38h
Сброс: 0h



Поле	Биты	Описание		
CHi	23-0	Чтение	0	Каналу i присвоен уровень приоритета по умолчанию
			1	Каналу i присвоен высокий уровень приоритета
		Запись нуля	Не выполняется	
		Запись единицы	Присваивает каналу i высокий уровень приоритета	
–	31-24	Зарезервировано		

PRIORITYCLR – регистр сброса установок приоритета каналов DMA

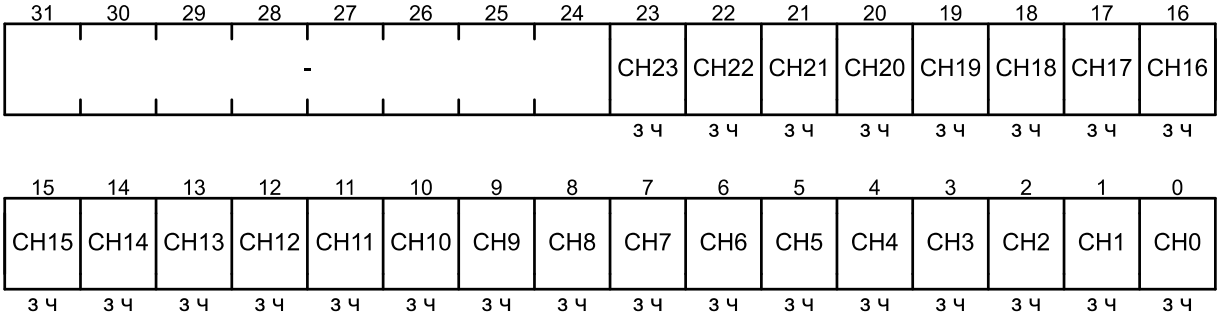
Смещение: + 3Ch
Сброс: 0h



Поле	Биты	Описание	
CHi	23-0	Чтение	Читаются нули
		Запись нуля	Не выполняется
		Запись единицы	Присваивает каналу i уровень приоритета по умолчанию
–	31-24	Зарезервировано	

CIRCULARSET– регистр установки циклического режима каналов DMA

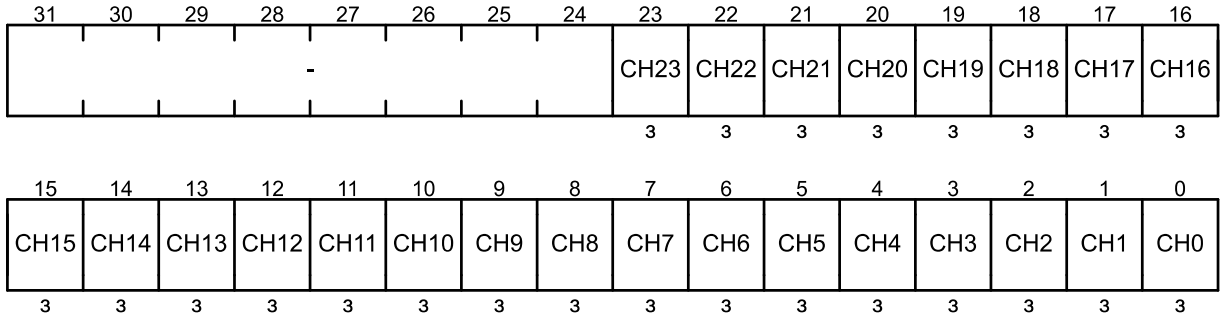
Смещение: + 40h
Сброс: 0h



Поле	Биты	Описание		
CHi	23-0	Чтение	0	Канал i функционирует в обычном режиме
			1	Канал i функционирует в циклическом режиме
		Запись нуля	Не выполняется	
		Запись единицы	Устанавливает циклический режим работы по каналу i	
–	31-24	Зарезервировано		

CIRCULARCLR – регистр сброса циклического режима каналов DMA

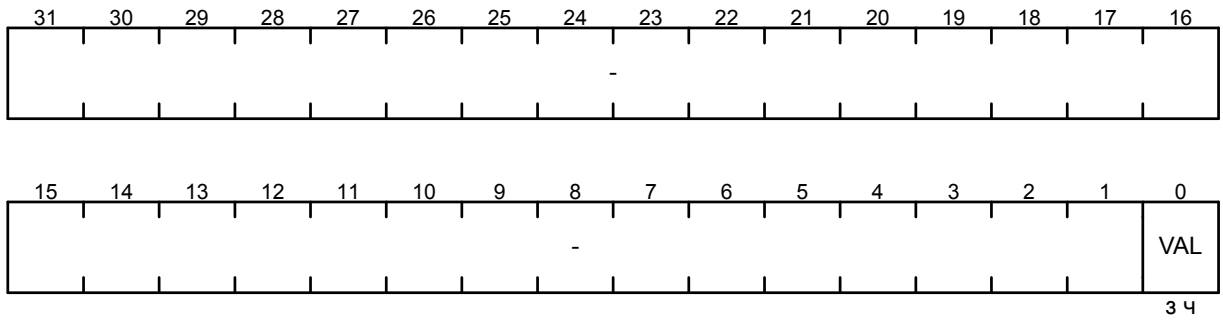
Смещение: + 44h
Сброс: 0h



Поле	Биты	Описание	
CHi	31-0	Чтение	Читаются нули
		Запись нуля	Не выполняется
		Запись единицы	Отключает циклический режим работы по каналу i
–	31-24	Зарезервировано	

ERRCLR – регистр сброса флага ошибки DMA

Смещение: + 4Ch
Сброс: 0h

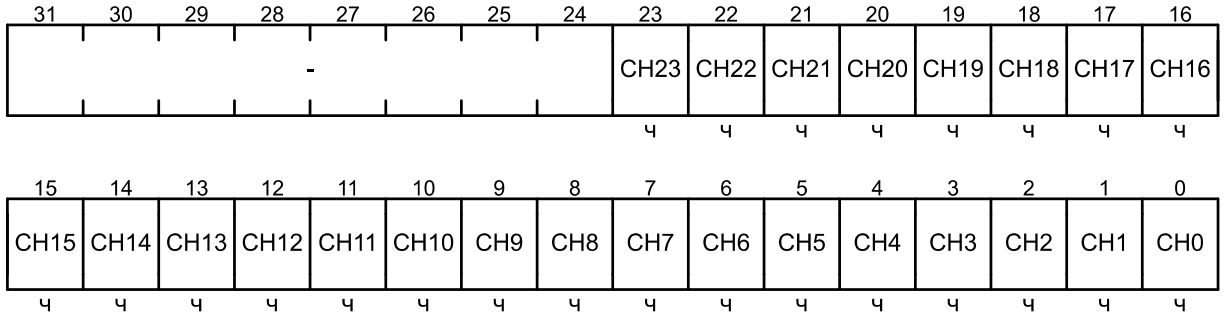


Поле	Биты	Описание		
VAL	0	Флаг ошибки на шине AHB		
		Чтение	0	Ошибок не обнаружено
			1	Произошла ошибка
		Запись нуля	Не выполняется	
		Запись единицы	Сбрасывает флаг ошибки VAL	
–	31-1	Зарезервировано		

Примечание – При одновременном сбросе флага VAL и появлении ошибки на шине AHB приоритет отдается ошибке, и бит VAL остается установленным

IRQSTAT – регистр статуса прерывания каналов DMA

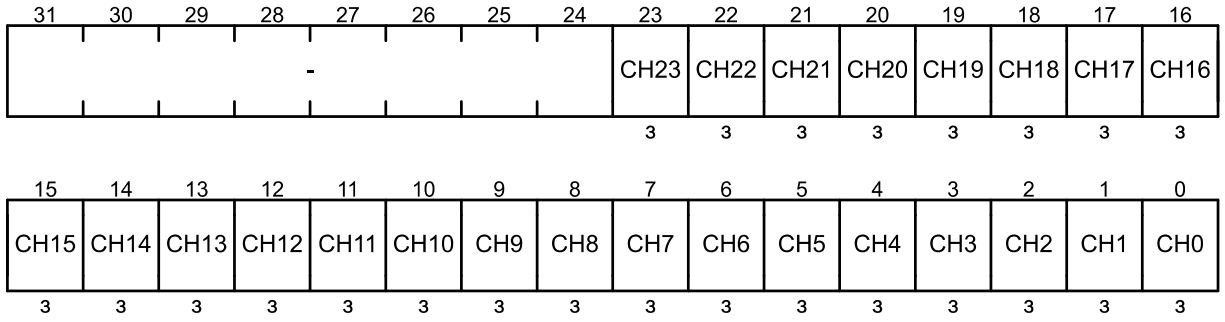
Смещение: + 50h
Сброс: 0h



Поле	Биты	Описание		
CHi	23-0	Чтение	0	Нет запроса на прерывание от канала i
			1	Активный запрос на прерывание от канала i
		Запись	Не выполняется	
–	31-24	Зарезервировано		

IRQSTATCLR – регистр сброса статуса прерывания каналов DMA

Смещение: + 54h
Сброс: 0h



Поле	Биты	Описание	
CHi	23-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Сбрасывает активный запрос на прерывание от канала i
–	31-24	Зарезервировано	

A.6 Регистры портов ввода-вывода

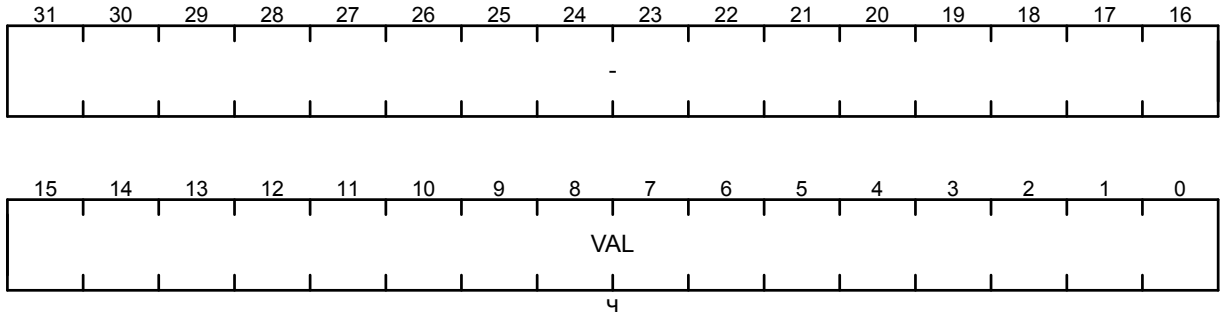
Базовый адрес:	2800_0000h	Регистры порта GPIOA;
	2800_1000h	Регистры порта GPIOB;
	2800_2000h	Регистры порта GPIOC;
Смещение:	+ 400h (MASKLB)	Массив регистров масок младшего байта порта;
	+ 800h (MASKHB)	Массив регистров масок старшего байта порта

Мнемоника: GPIOp

Примечание– p – имя порта A, B, или C;
n – порядковый номер вывода порта от 0 до 15.

DATA – регистр входных данных порта

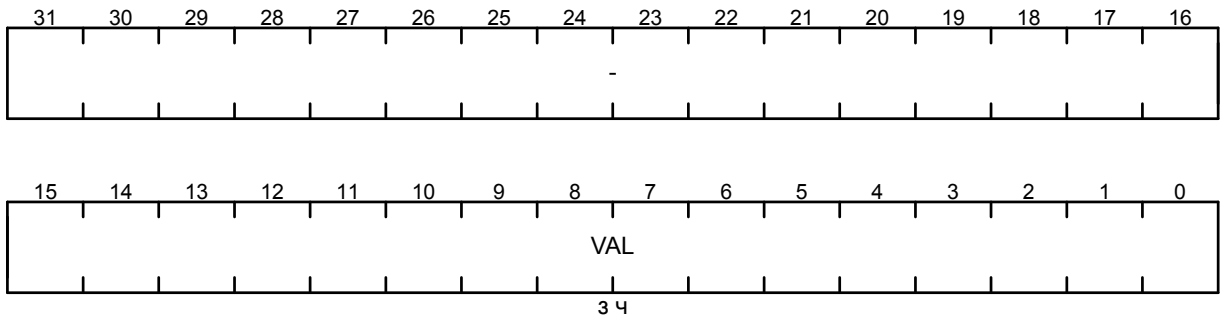
Смещение: + 00h
Сброс: 0000_xxxxh



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает текущее состояние порта
		Запись	Не выполняется
–	31-16	Зарезервировано	

DATAOUT – регистр выходных данных порта

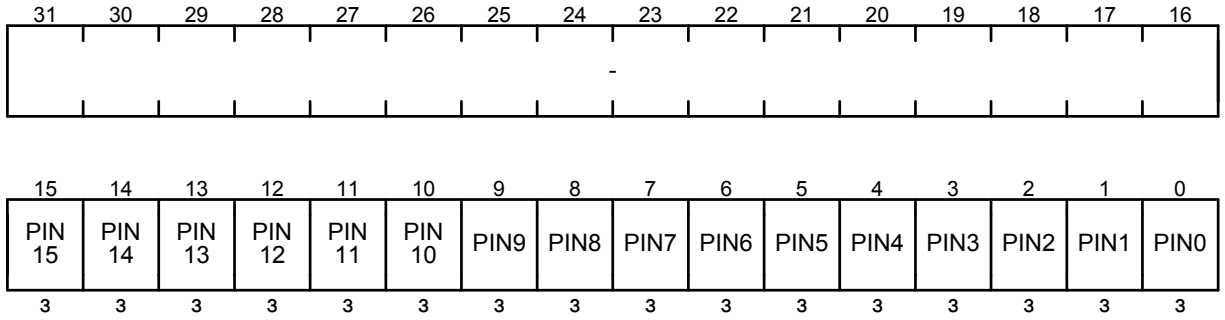
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние порта
		Запись	Устанавливает на выводах порта уровень сигнала, соответствующий записанному в биты значению
–	31-16	Зарезервировано	

DATAOUTSET – регистр установки битов порта

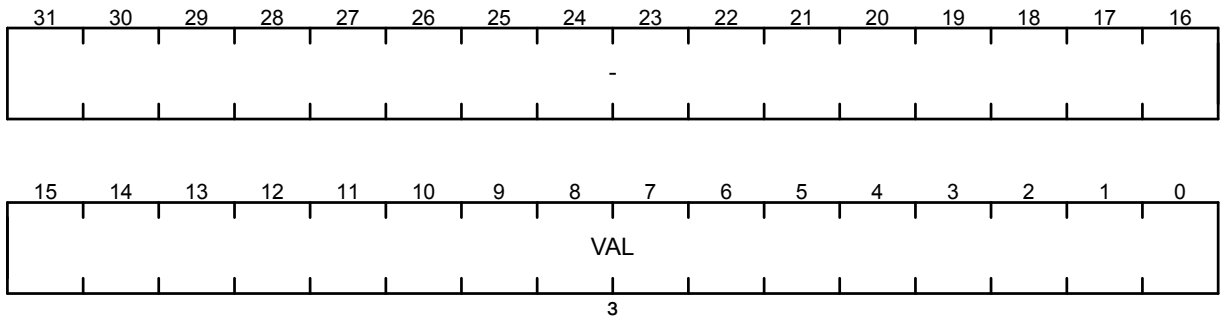
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает состояние порта (регистр DATAOUT)
		Запись нуля	Не выполняется
		Запись единицы	Устанавливает соответствующий бит n в регистре DATAOUT и, как следствие, высокий уровень сигнала на выводе n порта
–	31-16	Зарезервировано	

DATAOUTCLR – регистр сброса битов порта

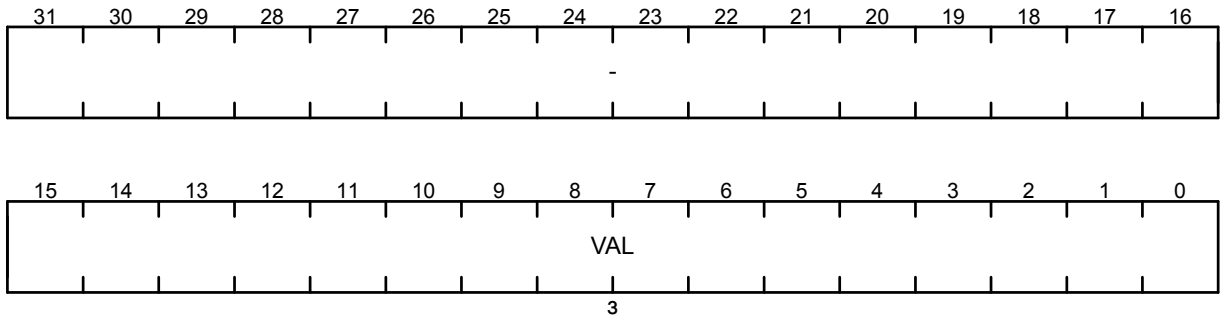
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние порта (регистр DATAOUT). Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Запись	Соответсвенно установленным битам сбрасывает биты в регистре DATAOUT и, как следствие, устанавливает низкий уровень сигнала на выводах порта
–	31-16	Зарезервировано	

DATAOUTTGL – регистр переключения битов порта

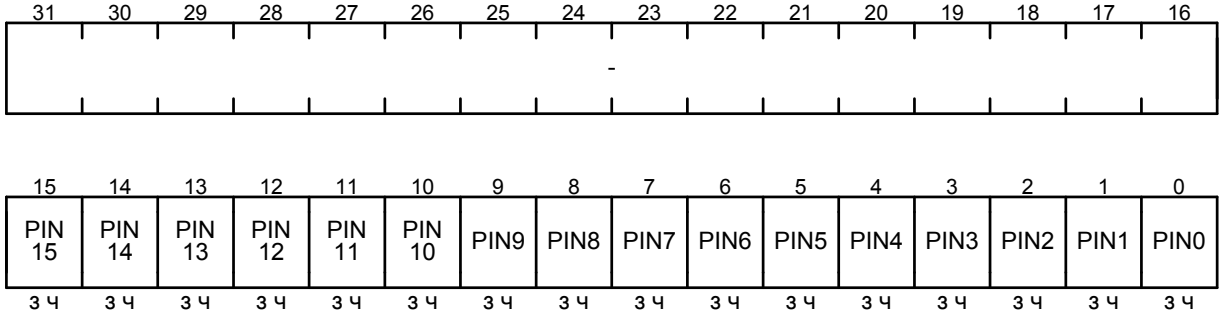
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние порта (регистр DATAOUT). Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Запись	Соответсвенно установленным битам изменяет состояние битов в регистре DATAOUT на противоположное и, как следствие, состояние сигналов на выводе порта
–	31-16	Зарезервировано	

PULLMODE – регистр выбора режима подтяжки порта

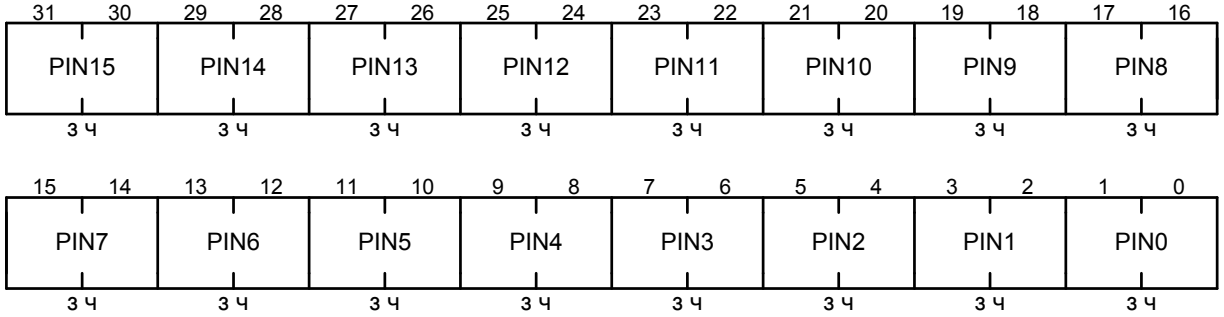
Смещение: + 20h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Бит включения режима подтяжки вывода n	
		0	Подтяжка отключена
		1	Подтяжка к уровню логической единицы (pull-up)
–	31-16	Зарезервировано	

OUTMODE – регистр выбора режима выхода порта

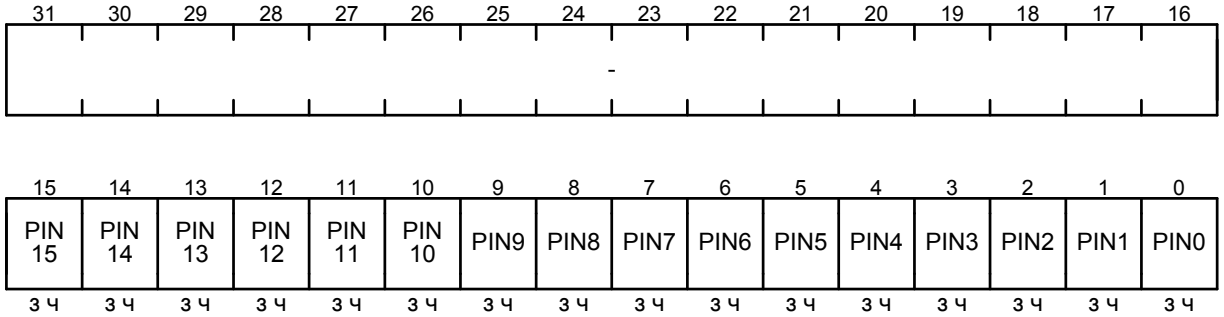
Смещение: + 24h
Сброс: 0h



Поле	Биты	Описание	
PINn	31-0	Поле выбора режима выхода n	
		00	Двухтактный выход (Push-Pull)
		01	Выход с открытым стоком (Open Drain)
		10	Выход с открытым истоком (Open Source)
		11	Зарезервировано

OUTENSET – регистр разрешения управления выходом порта

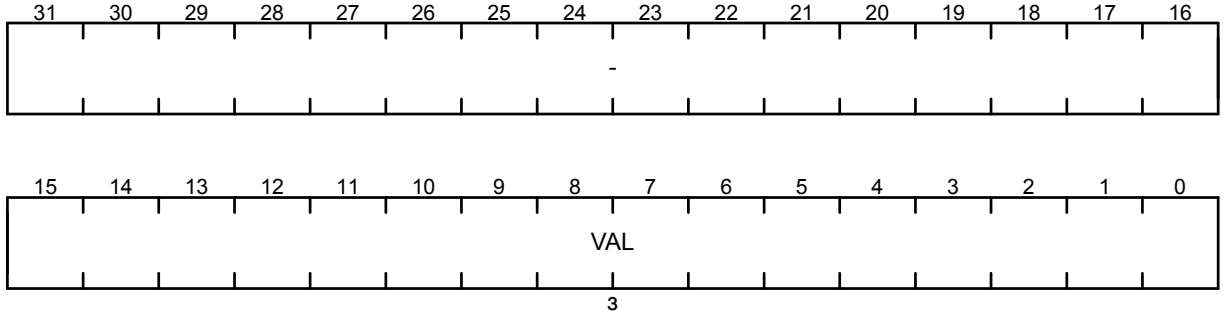
Смещение: + 2Ch
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Источник выходного сигнала (соответсвующий бит регистра DATAOUT) отключен от вывода n
			1	Состояние вывода n может задаваться источником выходного сигнала (соответсвующий бит регистра DATAOUT)
		Запись нуля	Не выполняется	
		Запись единицы	Разрешает управлять состоянием вывода с помощью соответсвующего бита регистра DATAOUT	
–	31-16	Зарезервировано		

OUTENCLR – регистр запрещения управления выходом порта

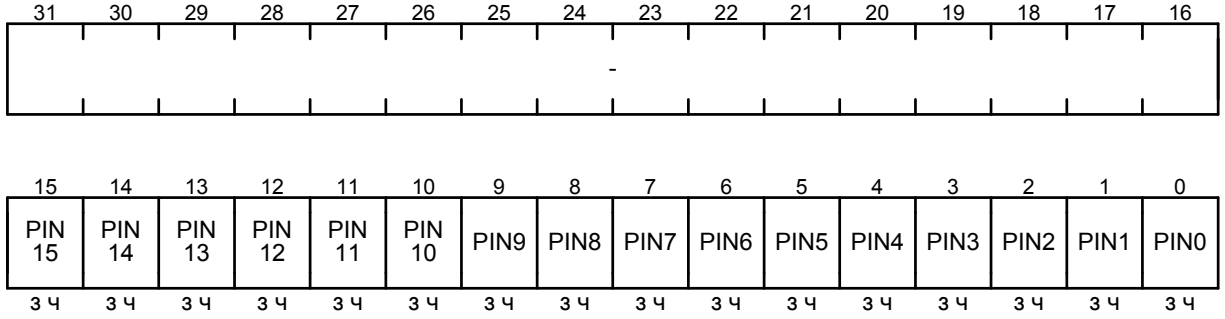
Смещение: + 30h
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние регистра OUTENSET. Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Запись	Соответсвенно установленным битам сбрасывает соответствующие биты регистра OUTENSET и запрещает управлять состоянием выводов с помощью соответсвующих битов регистра DATAOUT
–	31-16	Зарезервировано	

ALTFUNCSET– регистр включения альтернативной функции порта

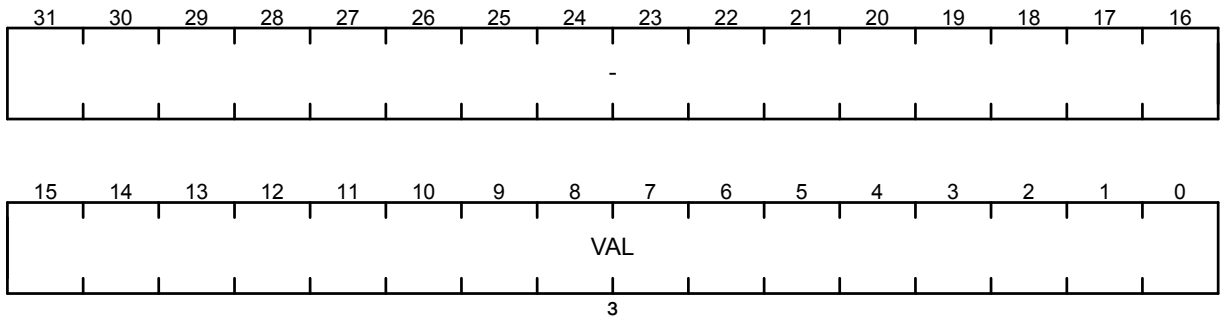
Смещение: + 34h
Сброс: 0h (для порта B)



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Вывод n в режиме входа/выхода общего назначения
			1	Вывод n в режиме альтернативной функции
		Запись нуля	Не выполняется	
		Запись единицы	Включает режим альтернативной функции вывода n	
–	31-16	Зарезервировано		

ALTFUNCCLR – регистр выключения альтернативной функции порта

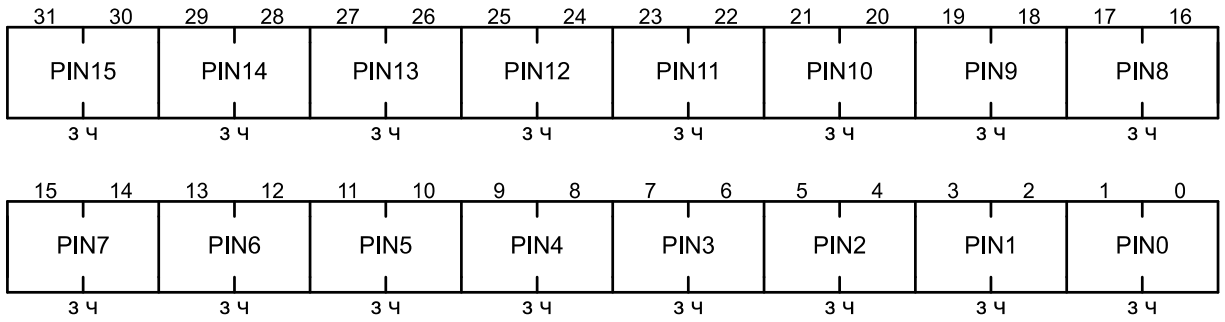
Смещение: + 38h
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние регистра ALTFUNCSET. Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Запись	Соответсвенно установленным битам включает режим входов/выходов общего назначения на выводах порта
–	31-16	Зарезервировано	

ALTFUNCNUM – регистр выбора альтернативной функции порта

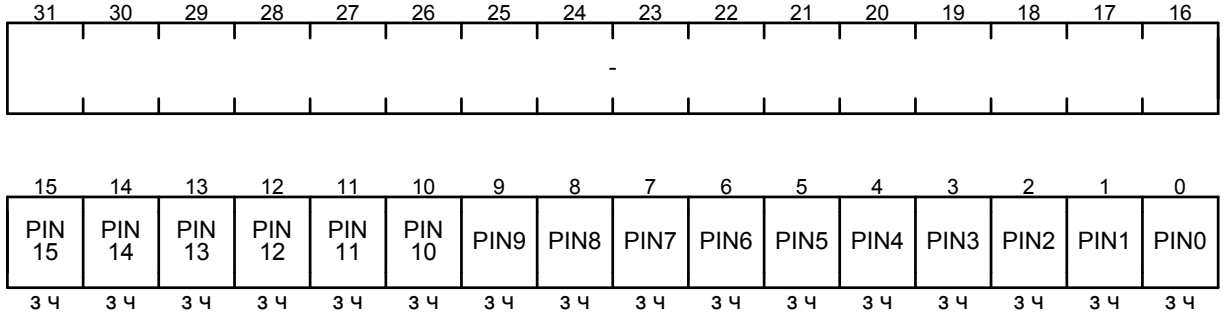
Смещение: + 3Ch
Сброс: 0h



Поле	Биты	Описание	
PINn	31-0	Поле выбора альтернативной функции пина	
		0h	Альтернативная функция пина не выбрана
		1h	Альтернативная функция 1
		2h	Альтернативная функция 2
		3h	Альтернативная функция 3

SYNCSET – регистр включения дополнительной синхронизации входов портов

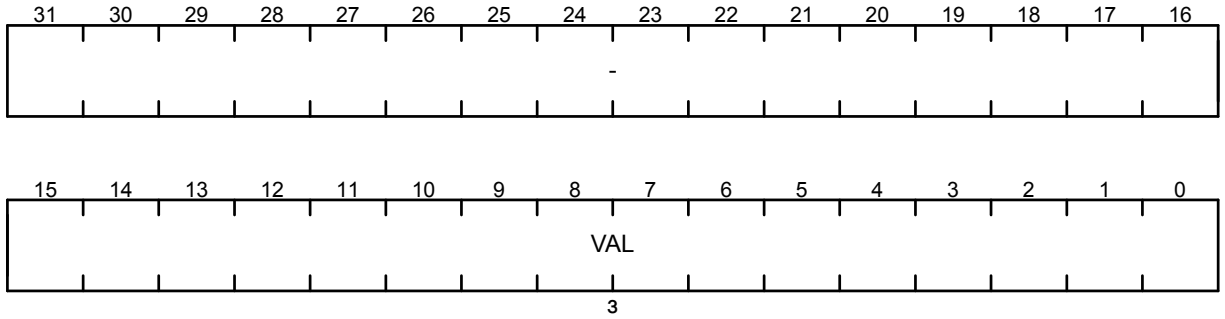
Смещение: + 44h
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Сигнал с вывода n передается в регистр DATA после двухтактной синхронизации (базовая)
			1	Сигнал с вывода n передается в регистр DATA после четырехтактной синхронизации (базовая и дополнительная)
		Запись нуля	Не выполняется	
		Запись единицы	Подключает дополнительную схему для получения четырехтактной синхронизации	
–	31-16	Зарезервировано		

SYNCCLR – регистр выключения дополнительной синхронизации входов портов

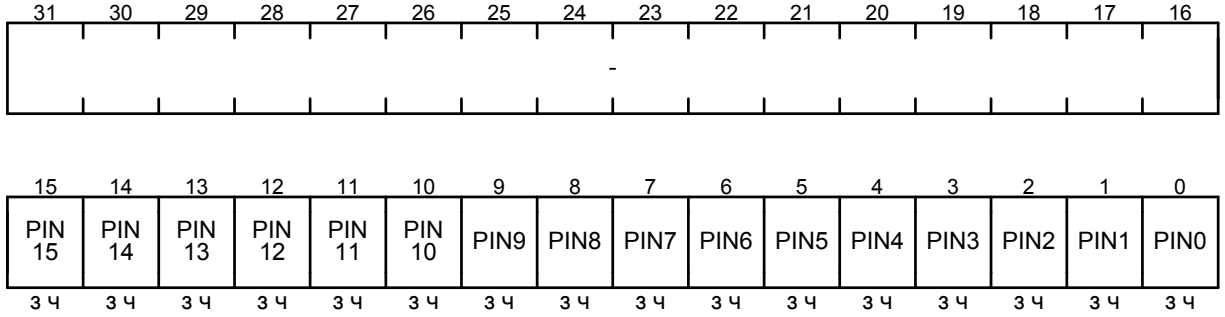
Смещение: + 48h
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние регистра SYNCSET. Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Запись	Соответсвенно установленным битам отключает дополнительные схемы синхронизации выводов
–	31-16	Зарезервировано	

QUALSET – регистр включения фильтров портов

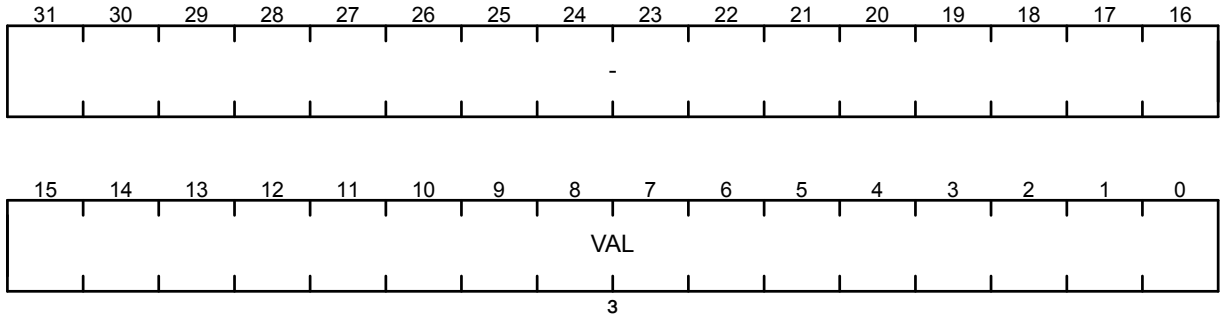
Смещение: + 4Ch
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	0 Входной фильтр вывода n отключен
			1 Входной фильтр вывода n включен
		Запись нуля	Не выполняется
		Запись единицы	Подключает входной фильтр вывода n
–	31-16	Зарезервировано	

QUALCLR – регистр отключения фильтров портов

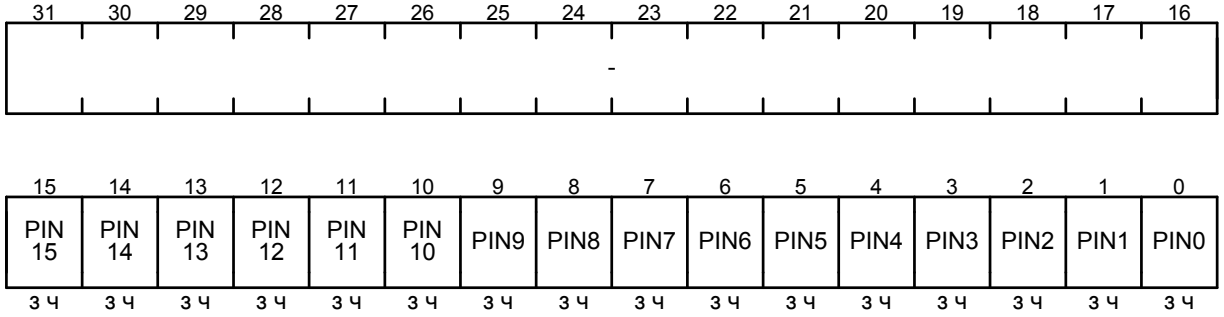
Смещение: + 50h
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние регистра QUALSET. Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Запись	Соответсвенно установленным битам отключает входные фильтры выводов
–	31-16	Зарезервировано	

QUALMODESET – регистр режима фильтра порта

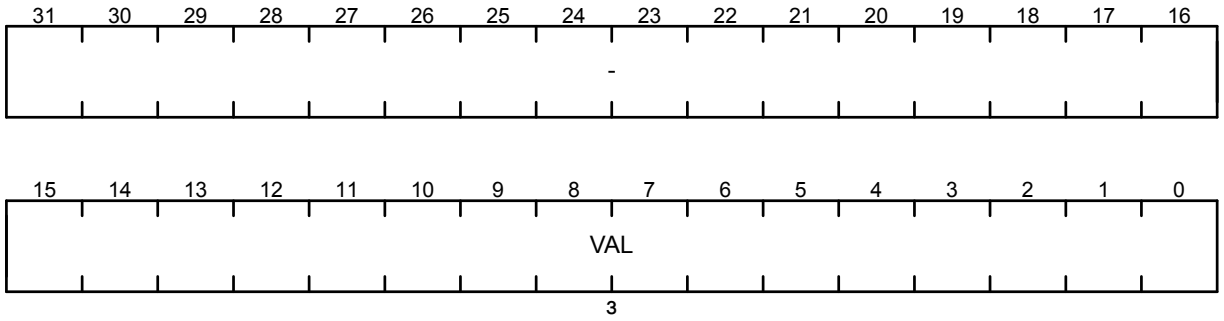
Смещение: + 54h
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Включен режим измерения уровня сигнала на выводе n по трем отсчетам
			1	Включен режим измерения уровня сигнала на выводе n по шести отсчетам
		Запись нуля	Не выполняется	
		Запись единицы	Включает режим измерения уровня сигнала на выводе n по шести отсчетам	
–	31-16	Зарезервировано		

QUALMODECLR – регистр сброса режима фильтра порта

Смещение: + 58h
Сброс: 0h

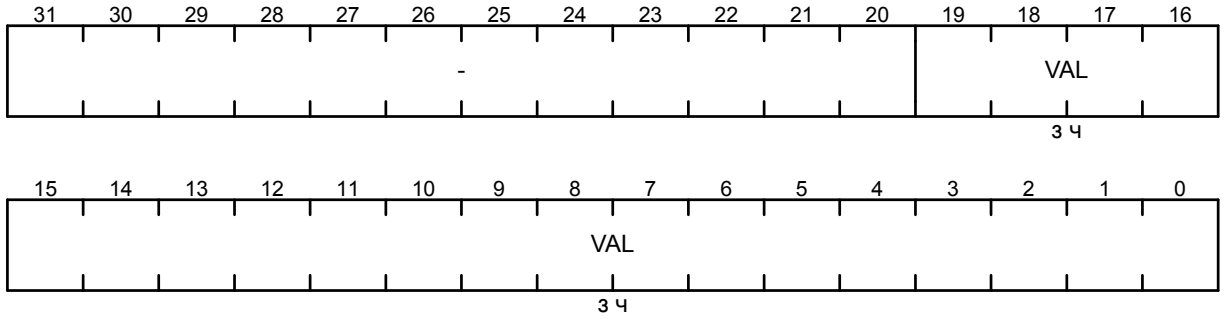


Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние регистра QUALMODESET. Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Запись	Соответсвенно установленным битам сбрасывает биты в регистре QUALMODESET и включает режим измерения уровня сигнала на выводах по трем отсчетам

–	31-16	Зарезервировано

QUALSAMPLE – регистр настройки фильтра порта

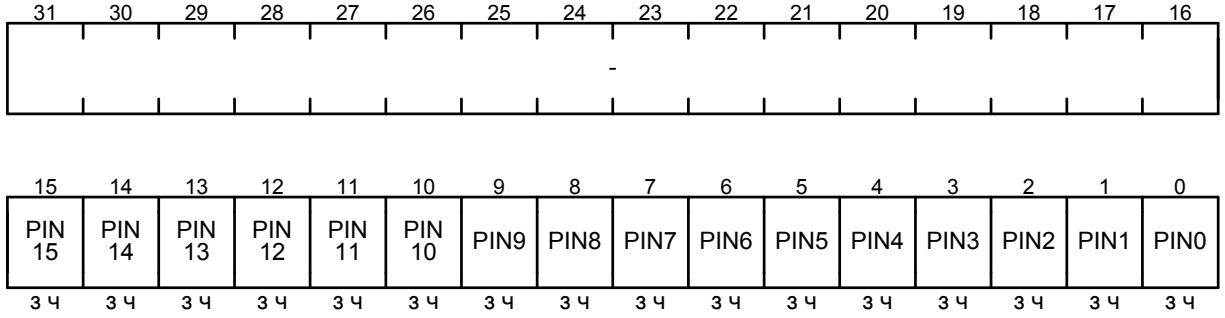
Смещение: + 5Ch
Сброс: 0h



Поле	Биты	Описание
PERIOD	19-0	Временной интервал (в тактах FCLK) между отсчетами при измерениях уровней сигналов на выводах порта. Заданное значение является единым для всех выводов порта
–	31-20	Зарезервировано

INTENSET – регистр разрешения прерываний порта

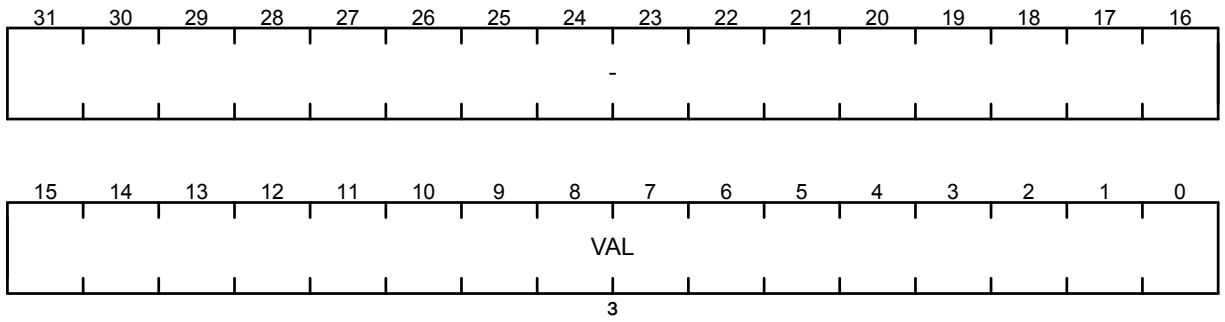
Смещение: + 60h
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Прерывания вывода n запрещены
			1	Прерывания вывода n разрешены
		Запись нуля	Не выполняется	
		Запись единицы	Разрешает прерывания вывода n	
–	31-16	Зарезервировано		

INTENCLR – регистр сброса разрешения прерываний порта

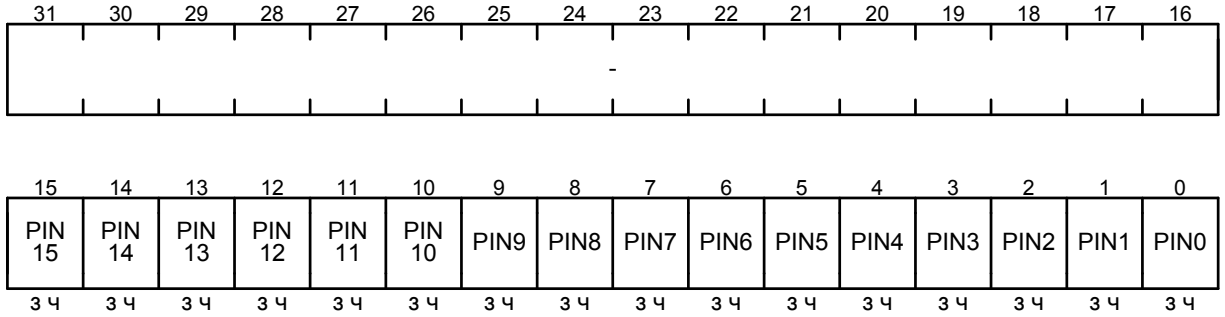
Смещение: + 64h
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние регистра INTENSET. Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Запись	Соответсвенно установленным битам сбрасывает биты в регистре INTENSET и запрещает прерывания выводов
–	31-16	Зарезервировано	

INTTYPESET – регистр типа прерываний порта

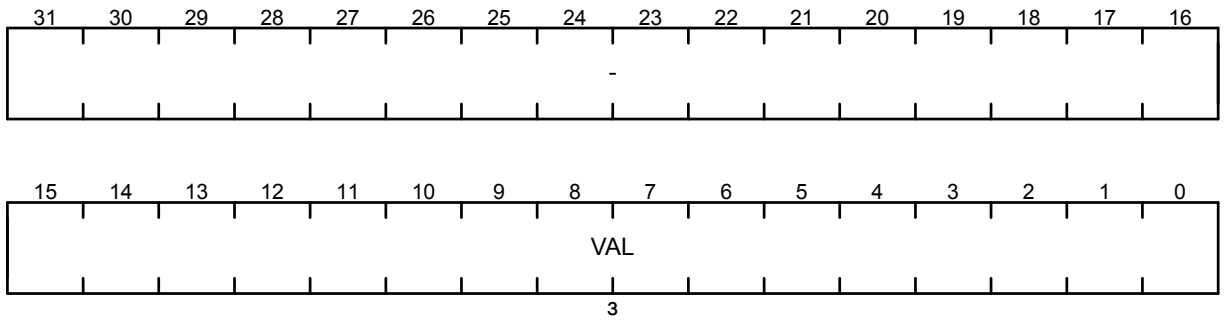
Смещение: + 68h
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Прерывания по уровню сигнала на выводе n
			1	Прерывания по фронту сигнала на выводе n
		Запись нуля	Не выполняется	
		Запись единицы	Выбирает прерывания по фронту сигнала на выводе n	
–	31-16	Зарезервировано		

INTTYPECLR – регистр сброса типа прерываний порта

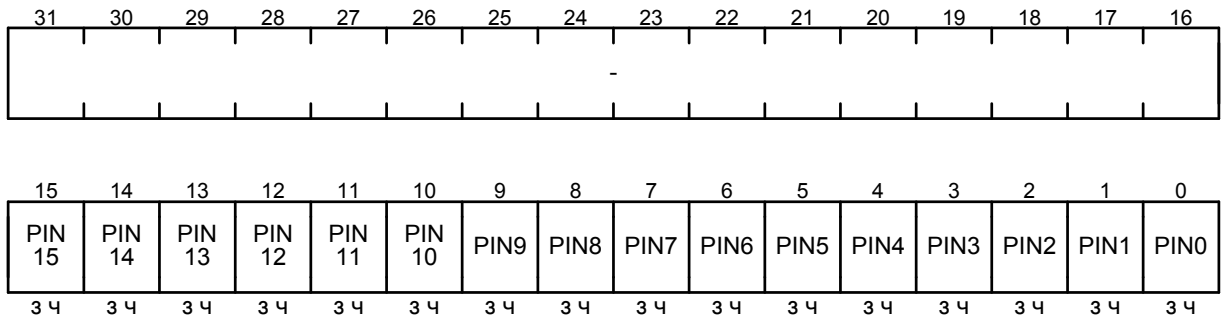
Смещение: + 6Ch
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние регистра INTTYPESET. Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Запись	Соответсвенно установленным битам сбрасывает биты в регистре INTTYPESET и выбирает прерывания по уровню сигнала на выводах
–	31-16	Зарезервировано	

INTPOLSET – регистр полярности события прерывания порта

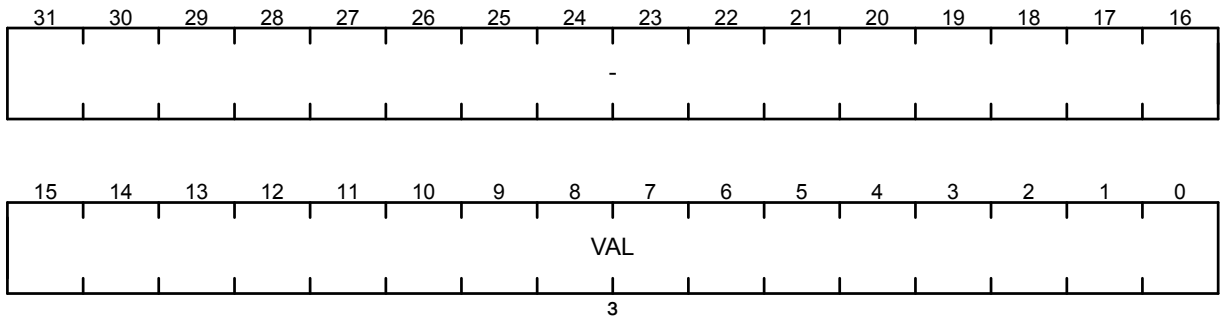
Смещение: + 70h
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Прерывания по низкому уровню или отрицательному фронту сигнала на выводе n
			1	Прерывания по высокому уровню или положительному фронту сигнала на выводе n
		Запись нуля	Не выполняется	
		Запись единицы	Выбирает прерывания по высокому уровню или положительному фронту	
–	31-16	Зарезервировано		

INTPOLCLR – регистр сброса полярности события прерывания порта

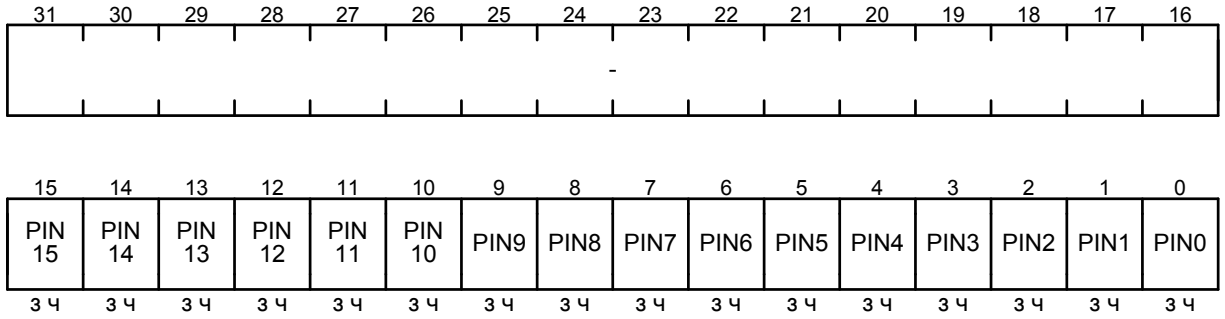
Смещение: + 74h
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние регистра INTPOLSET. Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Запись	Соответсвенно установленным битам сбрасывает биты в регистре INTPOLCLR и выбирает прерывания по низкому уровню или отрицательному фронту
–	31-16	Зарезервировано	

INTEDGESET – регистр включения прерывания по любому перепаду

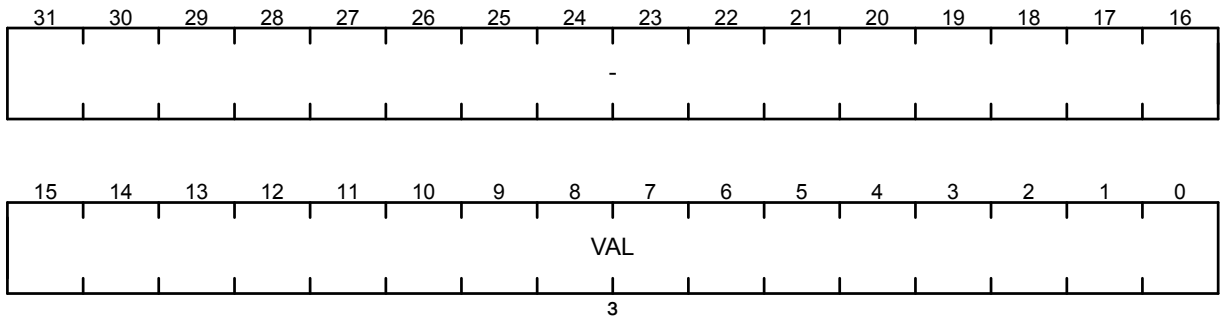
Смещение: + 78h
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Прерывания по любому фронту сигнала на выводе n запрещены
			1	Прерывания по любому фронту сигнала на выводе n разрешены
		Запись нуля	Не выполняется	
		Запись единицы	Разрешает прерывания по любому фронту сигнала на выводе n	
–	31-16	Зарезервировано		

INTEDGECLR – регистр отключения прерывания по любому перепаду

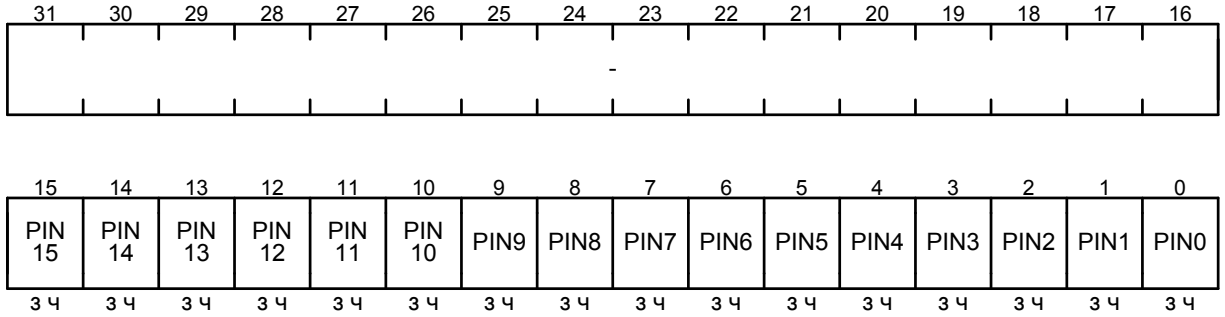
Смещение: + 7Ch
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние регистра INTEDGESET. Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Запись	Соответсвенно установленным битам сбрасывает биты в регистре INTEDGESET и запрещает прерывания по любому фронту сигнала на выводах
–	31-16	Зарезервировано	

INTSTATUS – регистр состояния и сброса прерываний порта

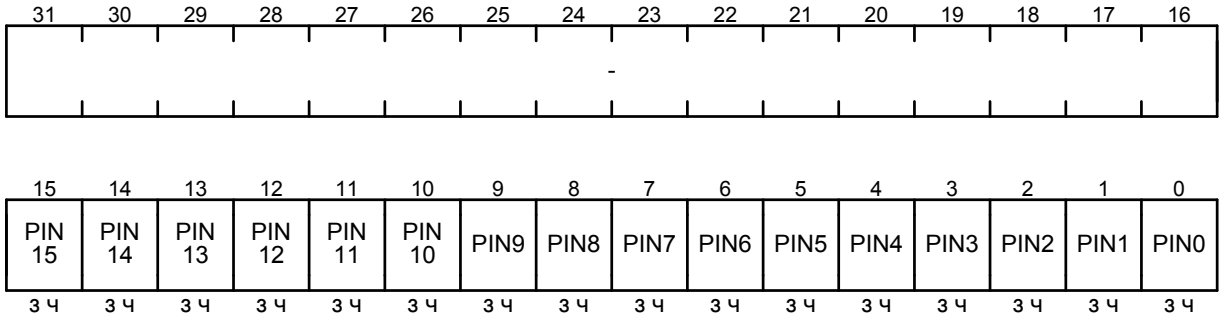
Смещение: + 80h
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Нет прерывания
			1	Флаг запроса на прерывание. Бит не сбрасывается аппаратно
		Запись нуля	Не выполняется	
		Запись единицы	Обнуляет бит PINn и таким образом сбрасывает флаг прерывания, если он был установлен	
–	31-16	Зарезервировано		

DMAREQSET – регистр включения генерации запросов DMA по прерыванию порта

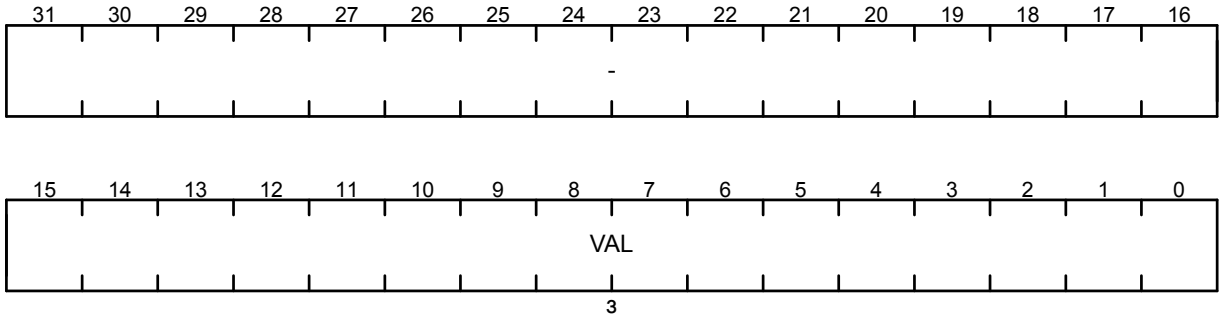
Смещение: + 84h
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Запрос DMA не генерироуется
			1	Запрос DMA генерируется по прерыванию (в том числе немаскированному)
		Запись нуля	Не выполняется	
		Запись единицы	Разрешает генерирование запросов DMA по прерыванию вывода n (в том числе немаскированному)	
–	31-16	Зарезервировано		

DMAREQCLR – регистр отключения генерации запросов DMA по прерыванию порта

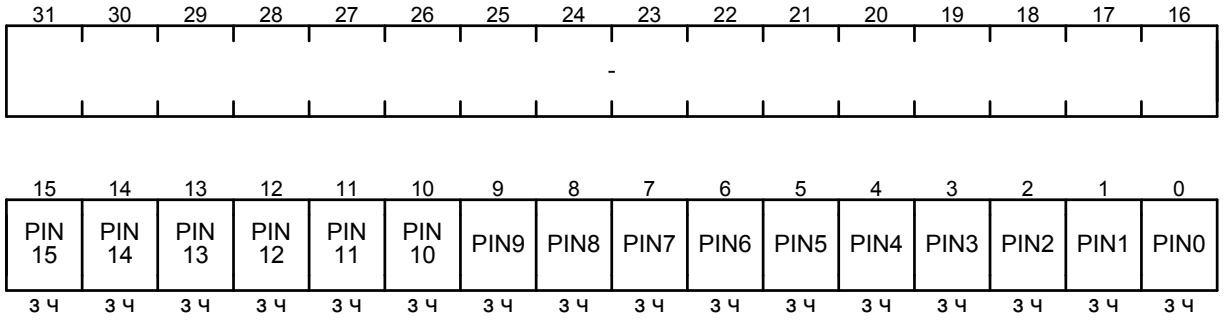
Смещение: + 88h
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние регистра DMAREQSET. Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Заспись	Соответсвенно установленным битам сбрасывает биты в регистре DMAREQSET и запрещает генерирование запросов DMA по прерыванию выводов
–	31-16	Зарезервировано	

ADCSOCSET – регистр включения генерации запросов начала преобразования АЦП по прерыванию порта

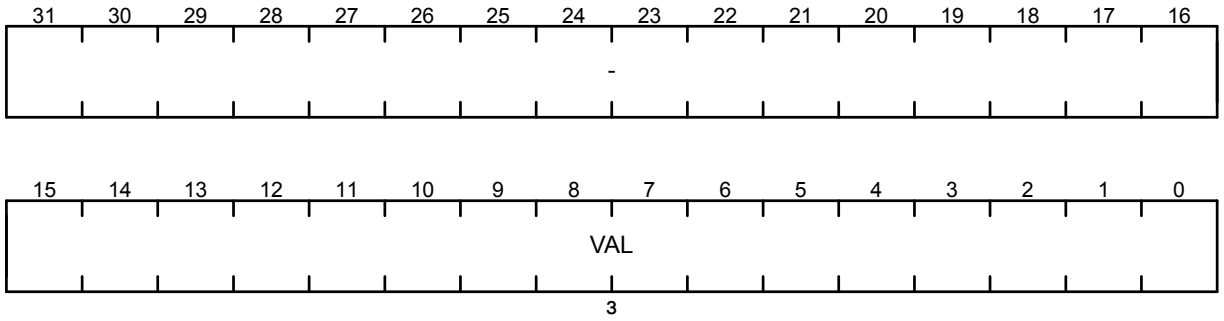
Смещение: + 8Ch
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Запрос начала преобразования АЦП не генерируется
			1	Запрос начала преобразования АЦП генерируется по прерыванию (в том числе немаскированному)
		Запись нуля	Не выполняется	
		Запись единицы	Разрешает генерирование запросов начала преобразования АЦП по прерыванию вывода n (в том числе немаскированному)	
–	31-16	Зарезервировано		

ADCSOCCLR – регистр отключения генерации запросов начала преобразования АЦП по прерыванию порта

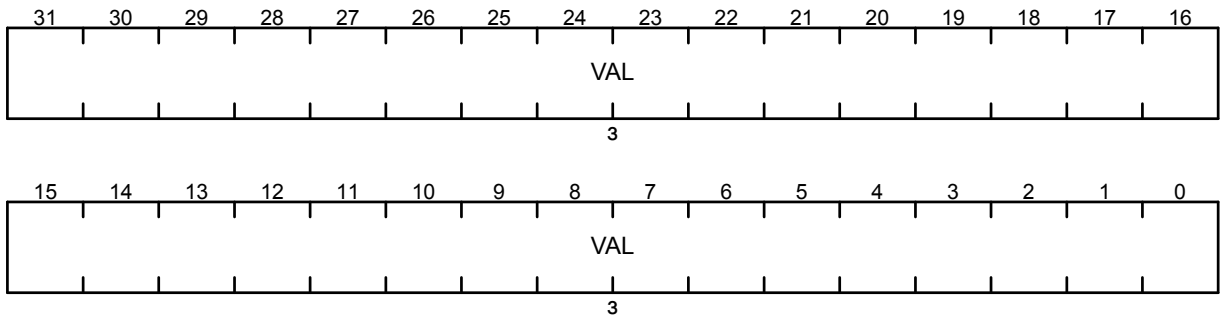
Смещение: + 90h
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние регистра ADCSOCSET. Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Запись	Соответсвенно установленным битам сбрасывает биты в регистре ADCSOCSET и запрещает генерирование запросов АЦП по прерыванию выводов
–	31-16	Зарезервировано	

LOCKKEY – регистр ключа блокировки

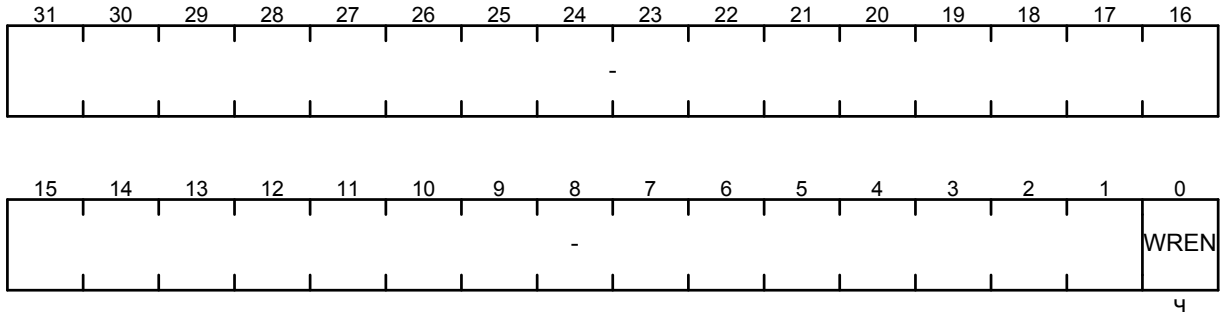
Смещение: + 9Ch
Сброс: 0h



Поле	Биты	Описание	
VAL	31-0	Значение ключа – ADEADBEEh	
		Чтение	Не выполняется
		Запись любого значения отличного от ключа	Блокирует доступ к регистрам LOCKSET и LOCKCLR
		Запись значения ключа	Разрешает доступ к регистрам LOCKSET и LOCKCLR

LOCKSTAT– регистр статуса блокировки

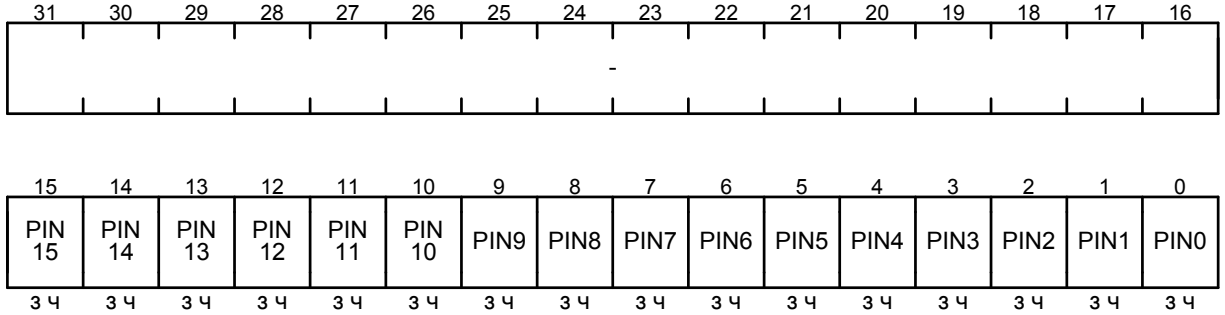
Смещение: + 9Ch
Сброс: 0h



Поле	Биты	Описание	
WREN	0	Индикатор доступа регистров LOCKSET и LOCKCLR для записи	
		0	Запрещено
		1	Разрешено
		Бит не доступен для записи	
–	31-1	Зарезервировано	

LOCKSET – регистр включения блокировки изменения конфигурации вывода

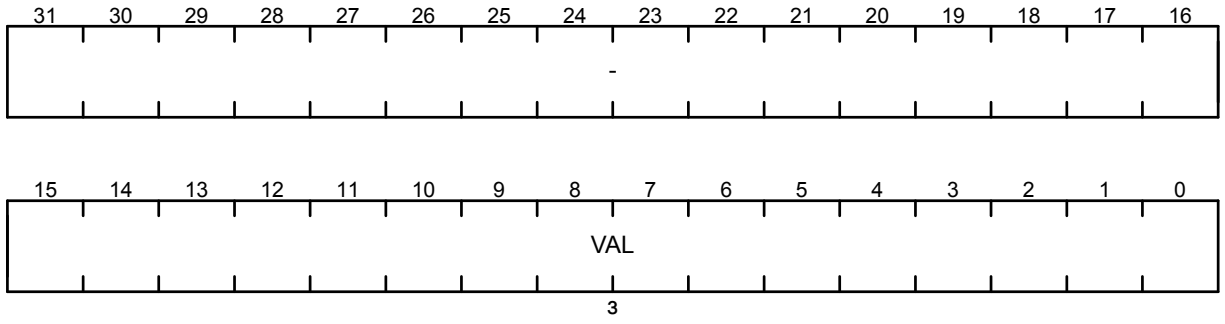
Смещение: + A0h
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Блокировка отключена
			1	Блокировка включена
		Запись нуля	Не выполняется	
		Запись единицы (возможна после установки ключа в регистре LOCKKEY)	Блокирует изменение конфигурации вывода n. Соответствующие биты n регистров становятся недоступными для записи. Исключения: - регистр флагов прерываний INTSTATUS; - регистр временного интервала измерений QUALSAMPLE	
–	31-16	Зарезервировано		

LOCKCLR – регистр отключения блокировки изменения конфигурации вывода

Смещение: + A4h
Сброс: 0h

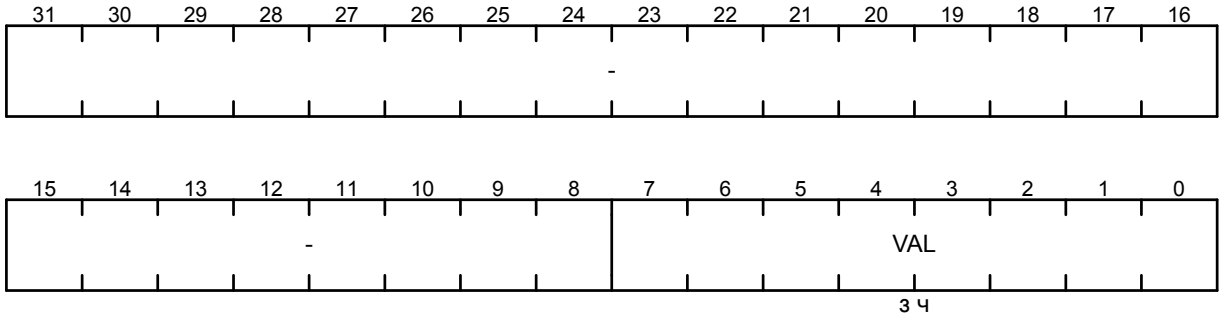


Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает состояние регистра LOCKSET. Примечание – Использование операции чтение-модификация-запись может привести к нежелательным результатам, влияющим на другие биты регистра.
		Запись (возможна после установки ключа в регистре LOCKKEY)	Соответсвенно установленным битам сбрасывает биты в регистре LOCKSET и разрешает изменения конфигурации выводов

–	31-16	Зарезервировано

MASKLB – регистр массива масок младшего байта порта

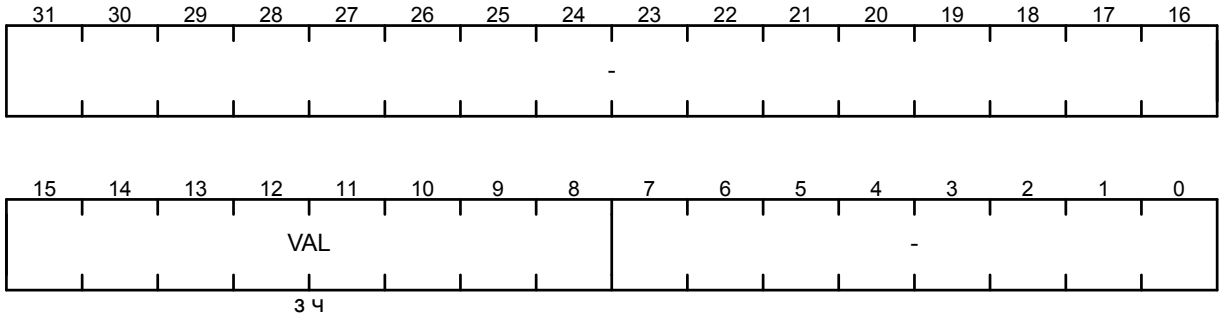
Адрес: GPIOp + MASKLB + (4*i)h, i = 0h, …, FFh
Сброс: 000000xxh



Поле	Биты	Описание
VAL	7-0	Доступ по маске для младших восьми бит порта
–	31-8	Зарезервировано

MASKHB – регистр массива масок старшего байта порта

Адрес: GPIOp + MASKHB + (4*i)h, i = 0h, …, FFh
Сброс: 0000_xx00h



Поле	Биты	Описание
VAL	15-8	Доступ по маске для старших восьми бит порта
–	31-16, 7-0	Зарезервировано

A.7 Регистры 32-разрядного таймера TMR32

Базовый адрес: 3000_0000h

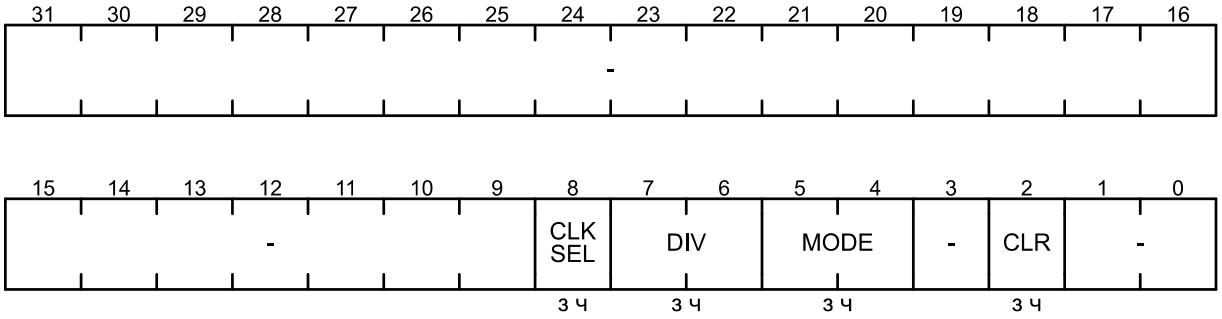
Смещение:	+ 18h (CAPCOM0)	Регистры захвата и сравнения CAPCOM0
	+ 20h (CAPCOM1)	Регистры захвата и сравнения CAPCOM1
	+ 28h (CAPCOM2)	Регистры захвата и сравнения CAPCOM2
	+ 30h (CAPCOM3)	Регистры захвата и сравнения CAPCOM3

Мнемоника: CAPCOMn;

Примечание – n – номер блока захвата и сравнения от 0 до 3.

CTRL – регистр конфигурации таймера

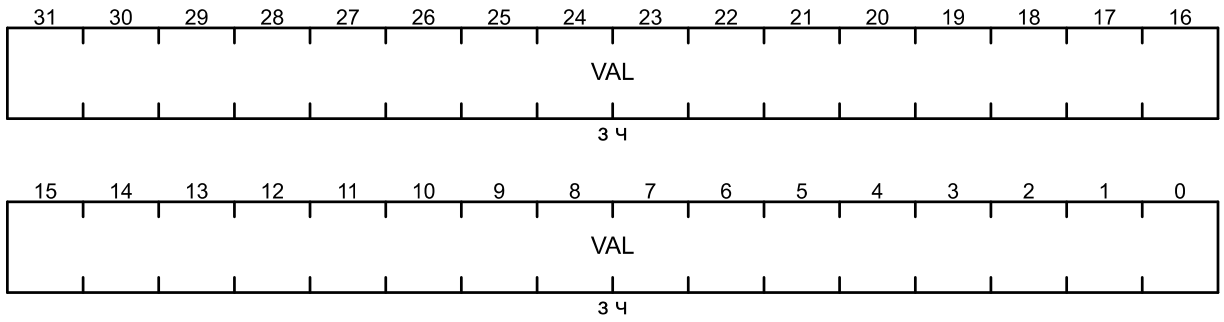
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание	
CLKSEL	8	Бит выбора источника тактирового сигнала	
		0	тактовый сигнал SYSCLK
		1	внешний источник синхронизации
DIV	7-6	Выбор входного делителя частоты таймера	
		0	1
		1	2
		2	4
		3	8
MODE	5-4	Выбор режима счета	
		0	останов
		1	счет вверх (многократно от 0 до значения CAPCOM0_VAL)
		2	непрерывно (многократно от 0 до значения FFFF_FFFFh)
		3	вверх/вниз (многократно от 0 вверх до значения CAPCOM0_VAL и обратно до 0).
CLR	2	Бит сброса счётчика (а также делителя и направления счета в режиме вверх/вниз)	
–	31-21, 13-12	Зарезервировано	

COUNT – регистр счетчика таймера

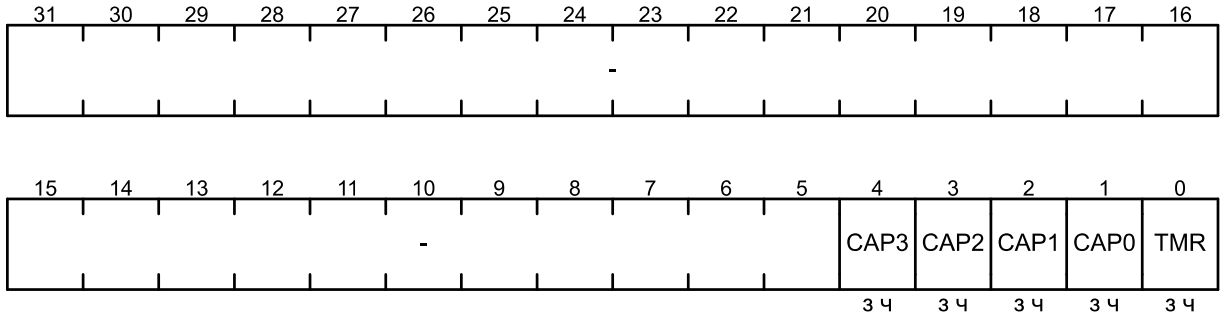
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	32-разрядное текущее значение счетчика таймера

IM – регистр маски прерываний

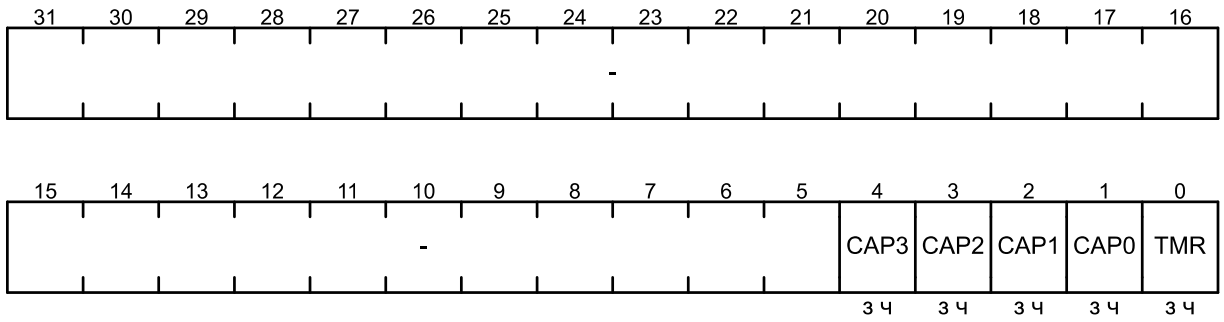
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание
CAP3	4	Маска прерывания CAPCOM3
CAP2	3	Маска прерывания CAPCOM2
CAP1	2	Маска прерывания CAPCOM1
CAP0	1	Маска прерывания CAPCOM0
TMR	0	Маска прерывания таймера
–	31-5	Зарезервировано

RIS – регистр состояния прерываний

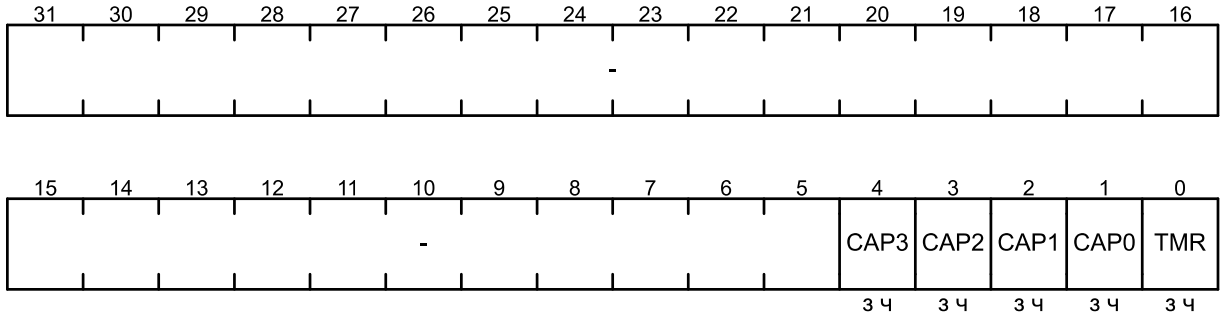
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание
CAP3	4	Флаг прерывания CAPCOM3
CAP2	3	Флаг прерывания CAPCOM2
CAP1	2	Флаг прерывания CAPCOM1
CAP0	1	Флаг прерывания CAPCOM0
TMR	0	Флаг прерывания таймера
–	31-5	Зарезервировано

MIS – регистр состояния прерываний с маскированием

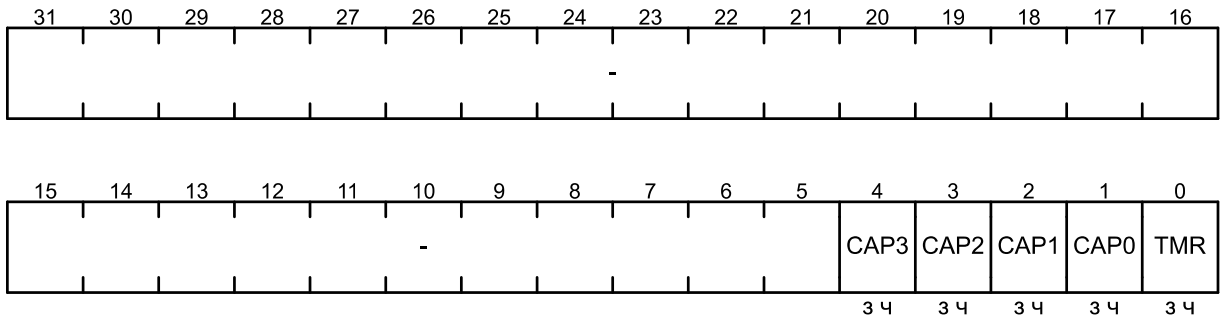
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
CAP3	4	Флаг маскированного прерывания CAPCOM3
CAP2	3	Флаг маскированного прерывания CAPCOM2
CAP1	2	Флаг маскированного прерывания CAPCOM1
CAP0	1	Флаг маскированного прерывания CAPCOM0
TMR	0	Флаг маскированного прерывания таймера
–	31-5	Зарезервировано

IC – регистр сброса прерываний

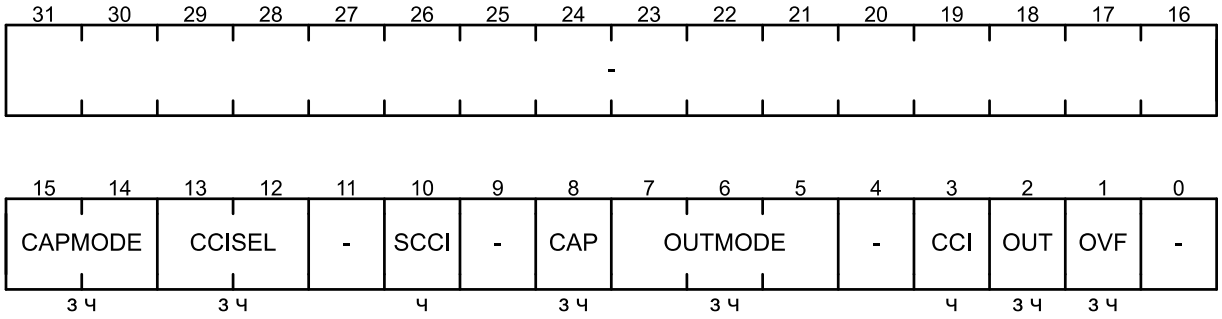
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание
CAP3	4	Сброс маскированного и немаскированного флага прерывания CAPCOM3
CAP2	3	Сброс маскированного и немаскированного флага прерывания CAPCOM2
CAP1	2	Сброс маскированного и немаскированного флага прерывания CAPCOM1
CAP0	1	Сброс маскированного и немаскированного флага прерывания CAPCOM0
TMR	0	Сброс маскированного и немаскированного флага прерывания таймера
–	31-5	Зарезервировано

CAPCOM_CTRL – регистр настройки захвата и сравнения

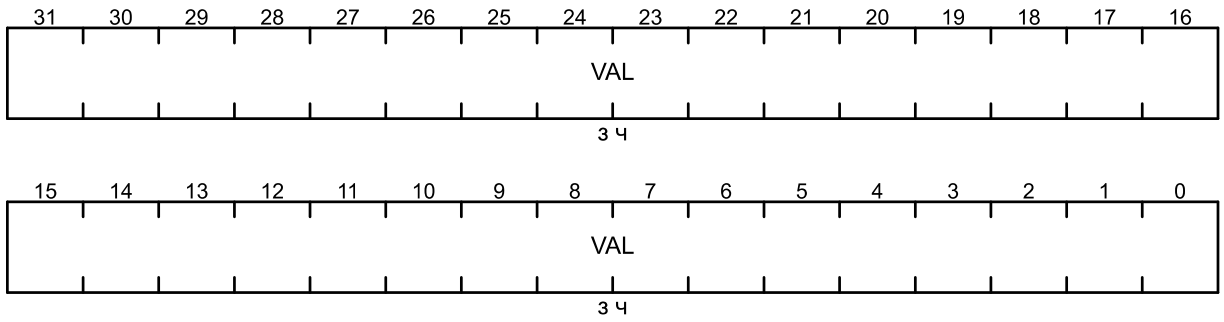
Смещение: CAPCOMn + 00h
Сброс: 0h



Поле	Биты	Описание
CAPMODE	15-14	Выбор режима захвата
		0 захвата нет
		1 захват по переднему фронту тактового сигнала
		2 захват по заднему фронту тактового сигнала
		3 захват по обоим фронтам тактового сигнала
CCISEL	13-12	Выбор входа захвата/сравнения
		0 CCIA
		1 CCIB
		2 Уровень логического нуля (Low)
		3 Уровень логической единицы (High)
SCCI	10	Бит состояния синхронизированного входа захвата/сравнения. Примечание – Выбранный входной сигнал защелкивается по сигналу EQUx и может быть прочитан через этот бит.
CAP	8	Выбор режима захвата/сравнения
		0 режим сравнения
		1 режим захвата (CAPMODE)
OUTMODE	7-5	Выбор режима выхода
		0 значение бита OUT
		1 установка (Set);
		2 переключение/сброс (Toggle/Reset);
		3 установка/сброс (Set/Reset);
		4 переключение (Toggle);
		5 сброс (Reset);
		6 переключение/установка (Toggle/Set);
		7 сброс/установка (Reset/Set).
CCI	3	Флаг состояния входа захвата/сравнения. Отображает значение выбранного входного сигнала
OUT	2	Выбор состояния выхода. Для режима выхода (OUTMODE = 0) бит напрямую определяет значение выходного сигнала.
		0 низкий уровень выходного сигнала
		1 высокий уровень выходного сигнала
OVF	1	Флаг переполнения захвата
		0 переполнение отсутствует
		1 произошло переполнение
		Данный флаг должен быть сброшен программно, записью единицы.
–	31-16, 11, 9, 4, 0	Зарезервировано

CAPCOM_VAL – регистр значения захвата и сравнения

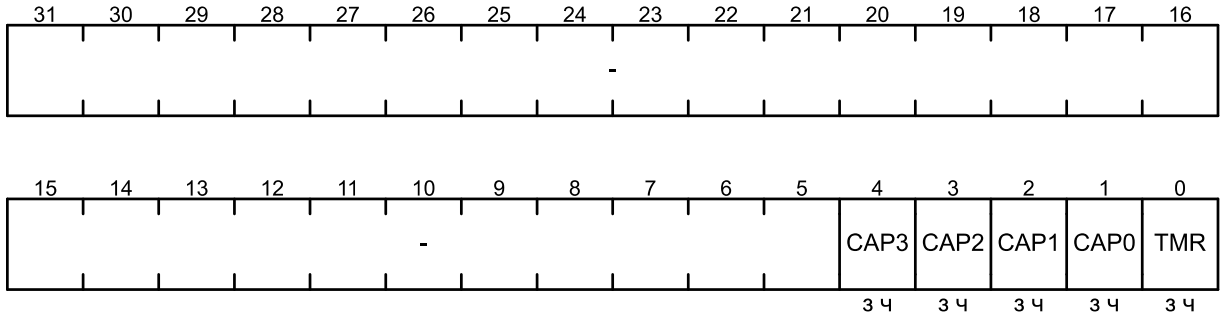
Смещение: CAPCOMn + 04h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	32-разрядное значение регистра захвата и сравнения

DMA_IM – регистр формирования запросов DMA

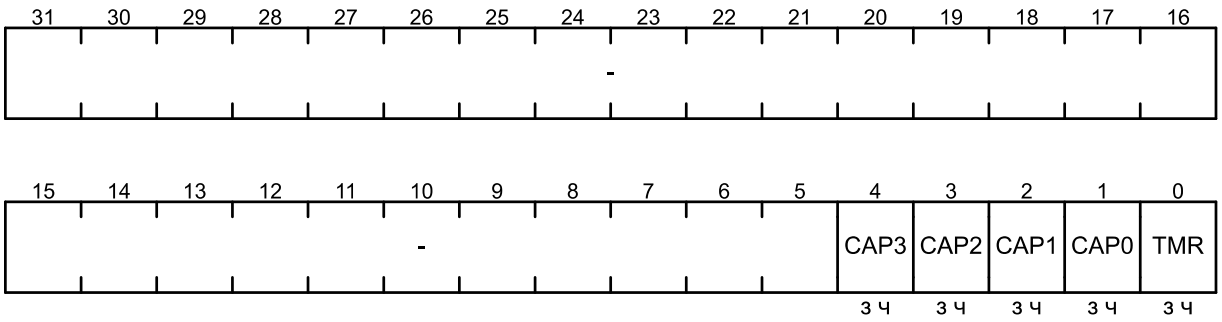
Смещение: + 38h
Сброс: 0h



Поле	Биты	Описание
CAP3	4	Бит разрешения формирования запросов DMA от CAPCOM3
CAP2	3	Бит разрешения формирования запросов DMA от CAPCOM2
CAP1	2	Бит разрешения формирования запросов DMA от CAPCOM1
CAP0	1	Бит разрешения формирования запросов DMA от CAPCOM0
TMR	0	Бит разрешения формирования запросов DMA при переполнении таймера
–	31-5	Зарезервировано

ADC_IM – регистр запуска АЦП

Смещение: + 3Ch
Сброс: 0h



Поле	Биты	Описание
CAP3	4	Бит разрешения запуска АЦП от CAPCOM3
CAP2	3	Бит разрешения запуска АЦП от CAPCOM2
CAP1	2	Бит разрешения запуска АЦП от CAPCOM1
CAP0	1	Бит разрешения запуска АЦП от CAPCOM0
TMR	0	Бит разрешения запуска АЦП при переполнении таймера
–	31-5	Зарезервировано

A.8 Регистры 16-разрядного таймера TMR

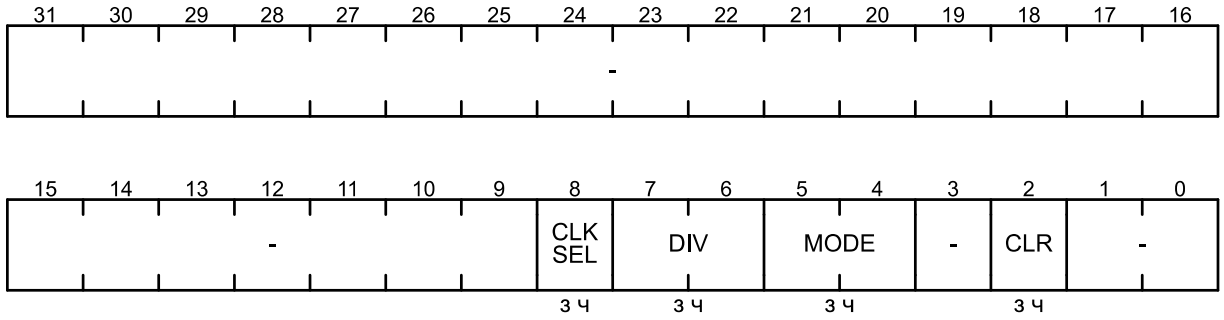
Базовый адрес:	3000_1000h	Регистры таймера TMR0
	3000_2000h	Регистры таймера TMR1
	3000_3000h	Регистры таймера TMR2
Смещение:	+ 18h (CAPCOM0)	Регистры захвата и сравнения CAPCOM0
	+ 20h (CAPCOM1)	Регистры захвата и сравнения CAPCOM1
	+ 28h (CAPCOM2)	Регистры захвата и сравнения CAPCOM2
	+ 30h (CAPCOM3)	Регистры захвата и сравнения CAPCOM3

Мнемоника: CAPCOMn;

Примечание – n – номер блока захвата и сравнения от 0 до 3.

CTRL – регистр конфигурации таймера

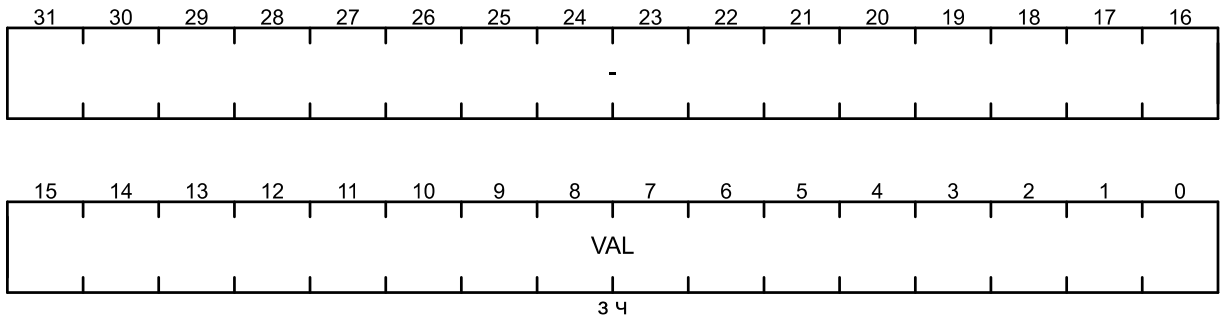
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание	
CLKSEL	8	Бит выбора источника тактирового сигнала	
		0	тактовый сигнал SYSCLK
		1	внешний источник синхронизации
DIV	7-6	Выбор входного делителя частоты таймера	
		0	1
		1	2
		2	4
		3	8
MODE	5-4	Выбор режима счета	
		0	останов
		1	счет вверх (многократно от 0 до значения CAPCOM0_VAL)
		2	непрерывно (многократно от 0 до значения FFFF_FFFFh)
		3	вверх/вниз (многократно от 0 вверх до значения CAPCOM0_VAL и обратно до 0).
CLR	2	Бит сброса счётчика (а также делителя и направления счета в режиме вверх/вниз)	
–	31-9, 3, 1-0	Зарезервировано	

COUNT – регистр счетчика таймера

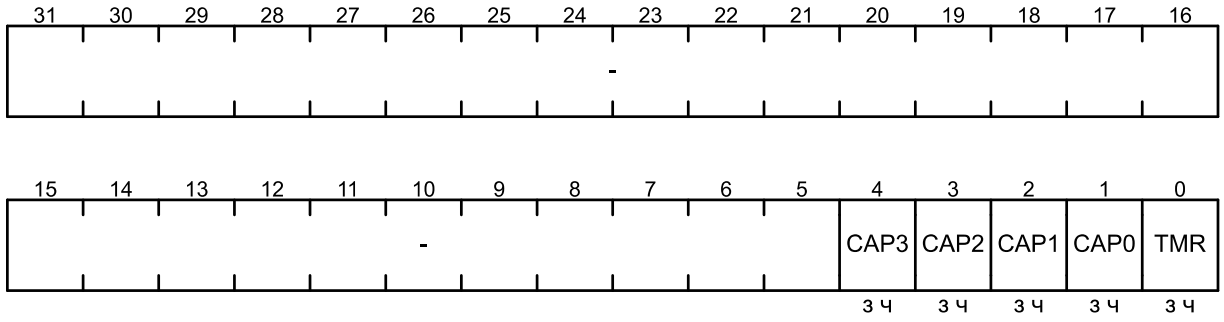
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание
VAL	15-0	16-разрядное текущее значение счетчика таймера
–	31-16	Зарезервировано

IM – регистр маски прерываний

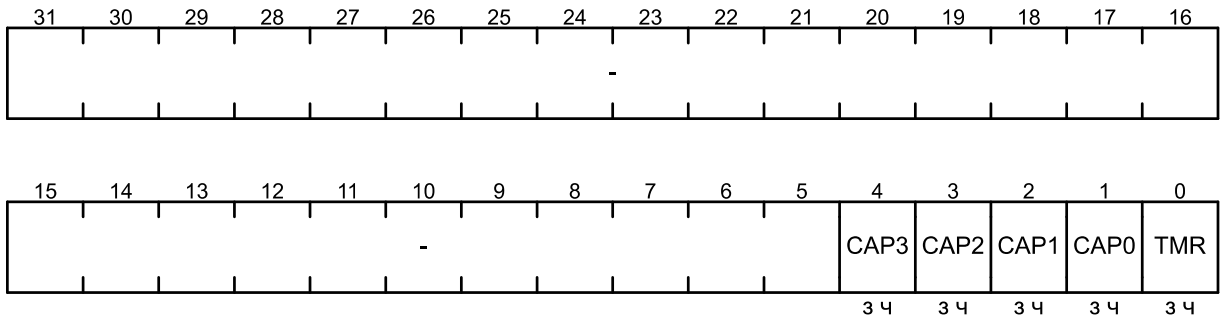
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание
CAP3	4	Маска прерывания CAPCOM3
CAP2	3	Маска прерывания CAPCOM2
CAP1	2	Маска прерывания CAPCOM1
CAP0	1	Маска прерывания CAPCOM0
TMR	0	Маска прерывания таймера
–	31-5	Зарезервировано

RIS – регистр состояния прерываний

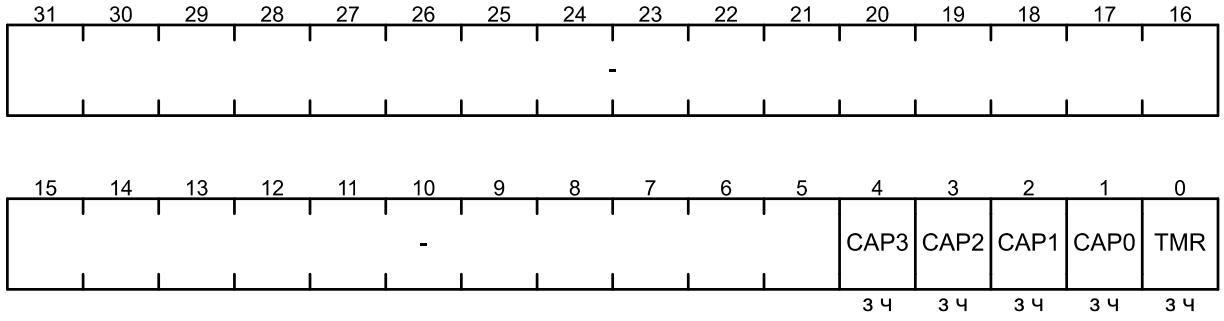
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание
CAP3	4	Флаг прерывания CAPCOM3
CAP2	3	Флаг прерывания CAPCOM2
CAP1	2	Флаг прерывания CAPCOM1
CAP0	1	Флаг прерывания CAPCOM0
TMR	0	Флаг прерывания таймера
–	31-5	Зарезервировано

MIS – регистр состояния прерываний с маскированием

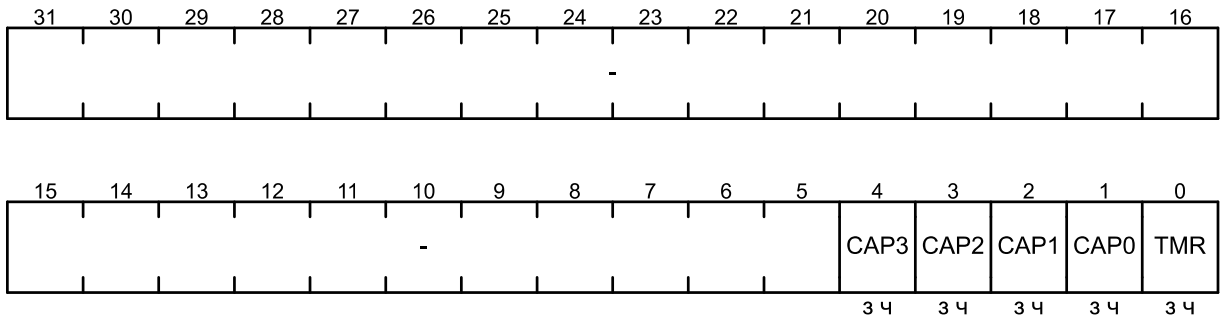
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
CAP3	4	Флаг маскированного прерывания CAPCOM3
CAP2	3	Флаг маскированного прерывания CAPCOM2
CAP1	2	Флаг маскированного прерывания CAPCOM1
CAP0	1	Флаг маскированного прерывания CAPCOM0
TMR	0	Флаг маскированного прерывания таймера
–	31-5	Зарезервировано

IC – регистр сброса прерываний

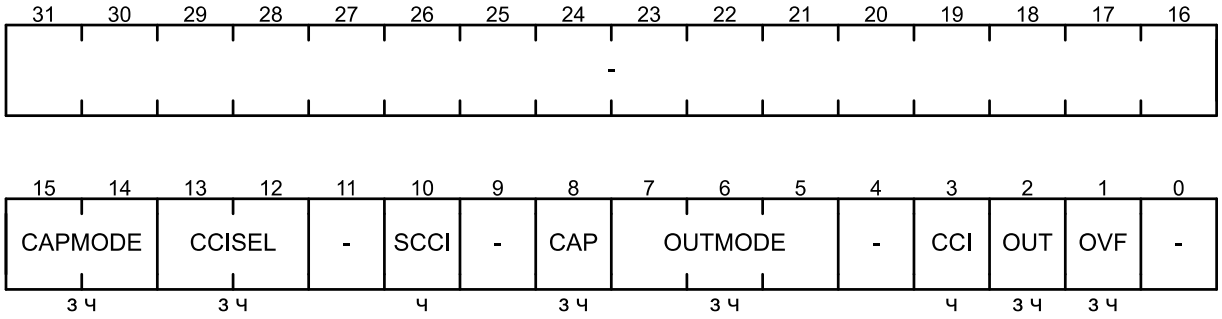
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание
CAP3	4	Сброс маскированного и немаскированного флага прерывания CAPCOM3
CAP2	3	Сброс маскированного и немаскированного флага прерывания CAPCOM2
CAP1	2	Сброс маскированного и немаскированного флага прерывания CAPCOM1
CAP0	1	Сброс маскированного и немаскированного флага прерывания CAPCOM0
TMR	0	Сброс маскированного и немаскированного флага прерывания таймера
–	31-5	Зарезервировано

CAPCOM_CTRL – регистр настройки захвата и сравнения

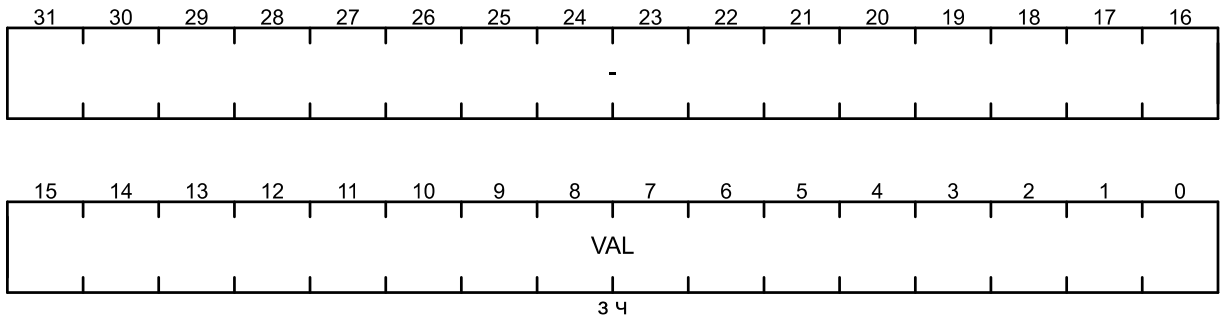
Смещение: CAPCOMn + 00h
Сброс: 0h



Поле	Биты	Описание
CAPMODE	15-14	Выбор режима захвата
		0 захвата нет
		1 захват по переднему фронту тактового сигнала
		2 захват по заднему фронту тактового сигнала
		3 захват по обоим фронтам тактового сигнала
CCISEL	13-12	Выбор входа захвата/сравнения
		0 CCInA
		1 CCInB
		2 Уровень логического нуля (Low)
		3 Уровень логической единицы (High)
SCCI	10	Бит состояния синхронизированного входа захвата/сравнения. Примечание – Выбранный входной сигнал защелкивается по сигналу EQUx и может быть прочитан через этот бит.
CAP	8	Выбор режима захвата/сравнения
		0 режим сравнения
		1 режим захвата
OUTMODE	7-5	Выбор режима выхода
		0 значение бита OUT
		1 установка (Set);
		2 переключение/сброс (Toggle/Reset);
		3 установка/сброс (Set/Reset);
		4 переключение (Toggle);
		5 сброс (Reset);
		6 переключение/установка (Toggle/Set);
		7 сброс/установка (Reset/Set).
CCI	3	Флаг состояния входа захвата/сравнения. Отображает значение выбранного входного сигнала
OUT	2	Выбор состояния выхода. Для режима выхода (OUTMODE = 0) бит напрямую определяет значение выходного сигнала.
		0 низкий уровень выходного сигнала
		1 высокий уровень выходного сигнала
OVF	1	Флаг переполнения захвата
		0 переполнение отсутствует
		1 произошло переполнение
		Данный флаг должен быть сброшен программно, записью единицы.
–	31-16, 11, 9, 4, 0	Зарезервировано

CAPCOM_VAL – регистр значения захвата и сравнения

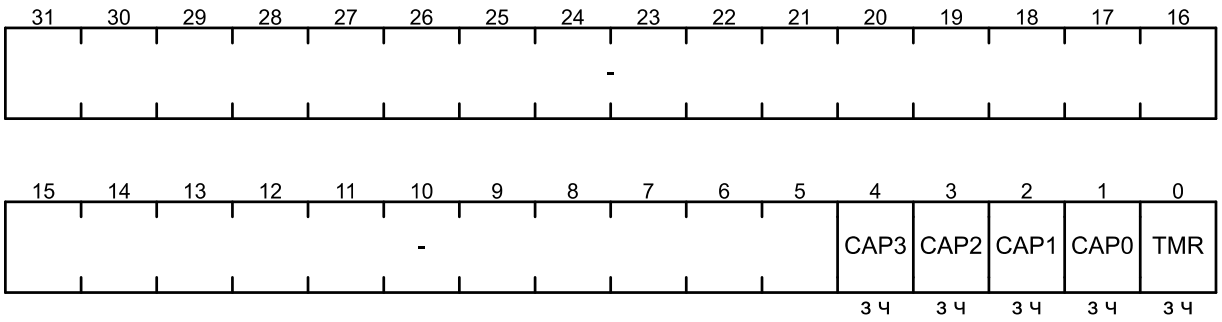
Смещение: CAPCOMn + 04h
Сброс: 0h



Поле	Биты	Описание
VAL	15-0	16-разрядное значение регистра захвата и сравнения
–	31-16	Зарезервировано

DMA_IM – регистр формирования запросов DMA

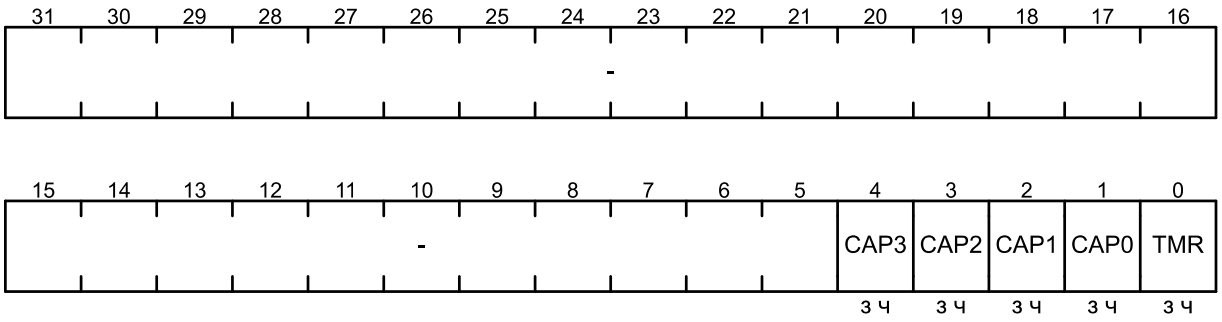
Смещение: + 38h
Сброс: 0h



Поле	Биты	Описание
CAP3	4	Бит разрешения формирования запросов DMA от CAPCOM3
CAP2	3	Бит разрешения формирования запросов DMA от CAPCOM2
CAP1	2	Бит разрешения формирования запросов DMA от CAPCOM1
CAP0	1	Бит разрешения формирования запросов DMA от CAPCOM0
TMR	0	Бит разрешения формирования запросов DMA при переполнении таймера
–	31-5	Зарезервировано

ADC_IM – регистр запуска АЦП

Смещение: + 3Ch
Сброс: 0h



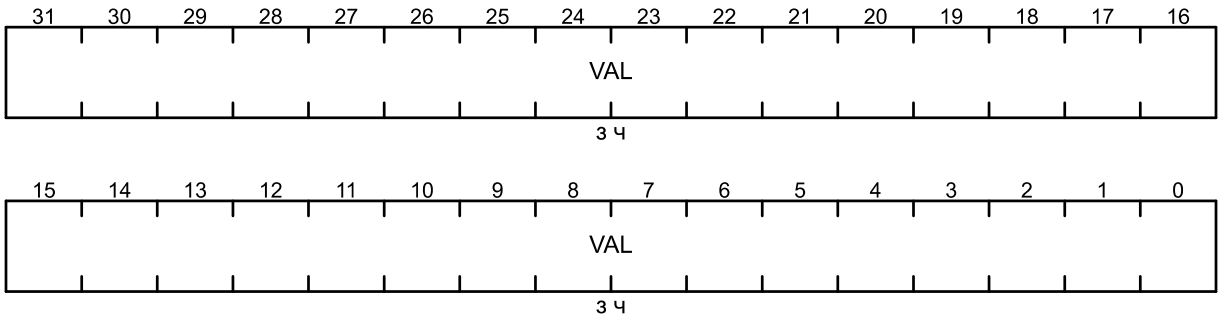
Поле	Биты	Описание
CAP3	4	Бит разрешения запуска АЦП от CAPCOM3
CAP2	3	Бит разрешения запуска АЦП от CAPCOM2
CAP1	2	Бит разрешения запуска АЦП от CAPCOM1
CAP0	1	Бит разрешения запуска АЦП от CAPCOM0
TMR	0	Бит разрешения запуска АЦП при переполнении таймера
–	31-5	Зарезервировано

A.9 Регистры блока подсчета циклического избыточного кода CRC

Базовый адрес: 2003_0000h Регистры CRC0
2003_1000h Регистры CRC1

DR - регистр данных

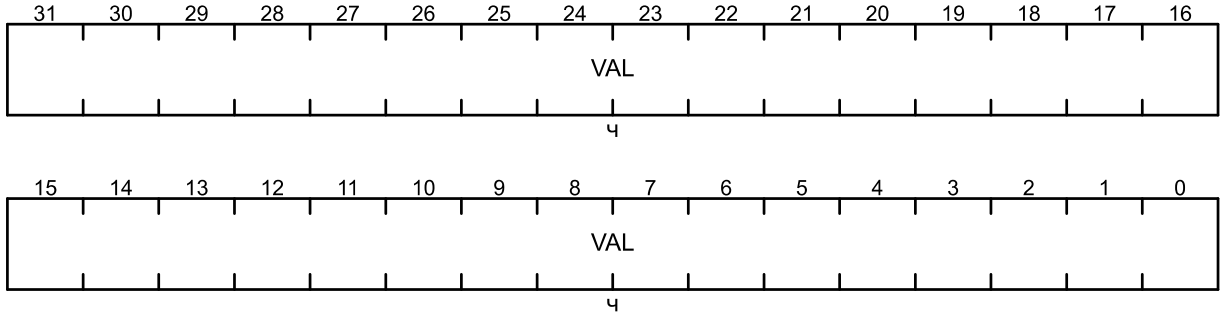
Смещение: 0h
Сброс: FFFF_FFFFh



Поле	Биты	Описание
VAL	31-0	Данные для записи в вычислительный блок CRC записываются в регистр. Вычисленное значение CRC возвращаются при чтении регистра. Если размер данных меньше 32-бит, младшие значащие биты используются для чтения/записи корректного значения.

POST – регистр пост-обработки

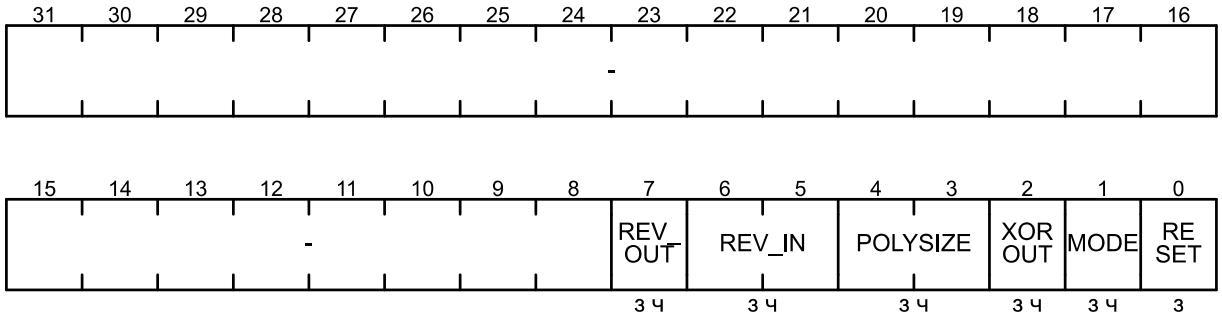
Смещение: 04h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение пост-обработки посчитанного значения CRC результата согласно установленным битам в регистре управления CR.

CR – регистр управления

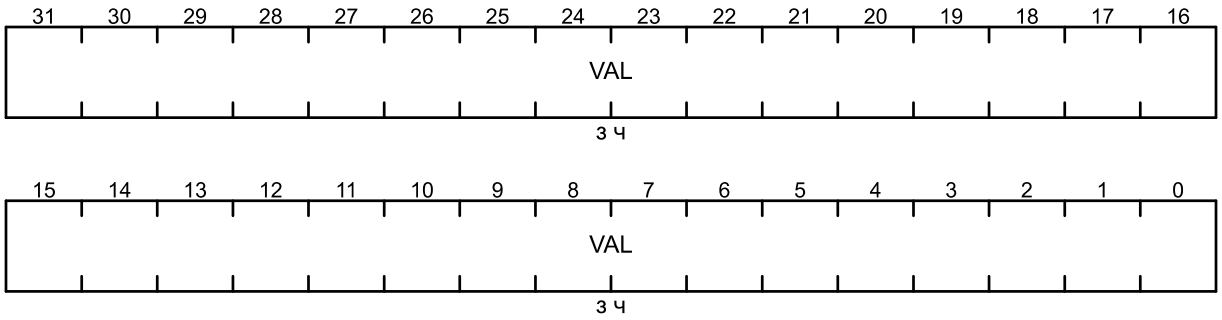
Смещение: 08h
Сброс: 0h



Поле	Биты	Описание	
REV_OUT	7	Обратить выходные данные. Этот бит управляет порядком следования бит выходных данных.	
		0	Порядок битов не изменяется
		1	Бит-реверсный формат выходных данных
REV_IN	6-5	Обращение входных данных Эти биты управляют порядком следования бит входных данных	
		0	Порядок битов не изменяется
		1	Бит-реверсный формат, внутри каждого байта выходных данных
		2	Бит-реверсный формат, внутри каждого полуслова выходных данных
		3	Бит-реверсный формат, внутри слова выходных данных
POLYSIZE	4-3	Размер полинома. Эти биты управляют размером порождающего полинома	
		0	32 – бит
		1	16 – бит
		2	8 – бит
		3	7 – бит
XOR_OUT	2	Операция «Исключающее ИЛИ» (XOR) над выходными данными «Исключающее ИЛИ» производится над выходными данными после обращения, если оно имело место быть. Пост-обработка.	
		0	Не выполняется
		1	Выходные данные подвергаются операции XOR со значением 0xFFFFFFFF
MODE	1	Режим работы блока подсчёта CRC. Бит определяет возможный режим работы блока подсчёта CRC:	
		0	Входные данные данные для расчёта поступают напрямую
		1	Каждая операция записи в регистр данных создаёт комбинацию (операция «Исключающее ИЛИ» (XOR)) с предыдущим содержимым регистра данных CRC
RESET	0	Бит сброса блока подсчёта CRC и установки значения регистра данных DR значением сохранённым в INIT.	
-	31-8	Зарезервировано	

INIT – регистр начального значения

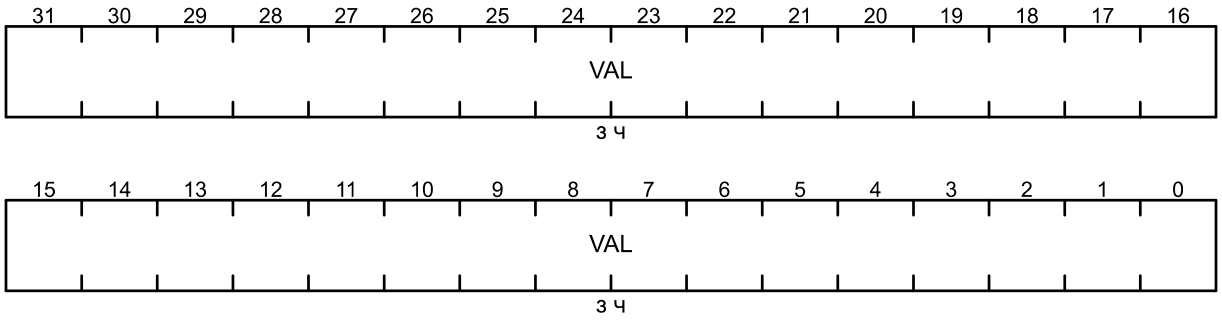
Смещение: 0Ch
Сброс: FFFF_FFFFh



Поле	Биты	Описание
VAL	31-0	Значение для установки начального значения CRC.

POL – регистр полинома

Смещение: 10h
Сброс: 04C1_1DB7h



Поле	Биты	Описание
VAL	31-0	Значение коэффициентов порождающего полинома, который используется при подсчёте CRC.

A.10 Регистры блока HASH

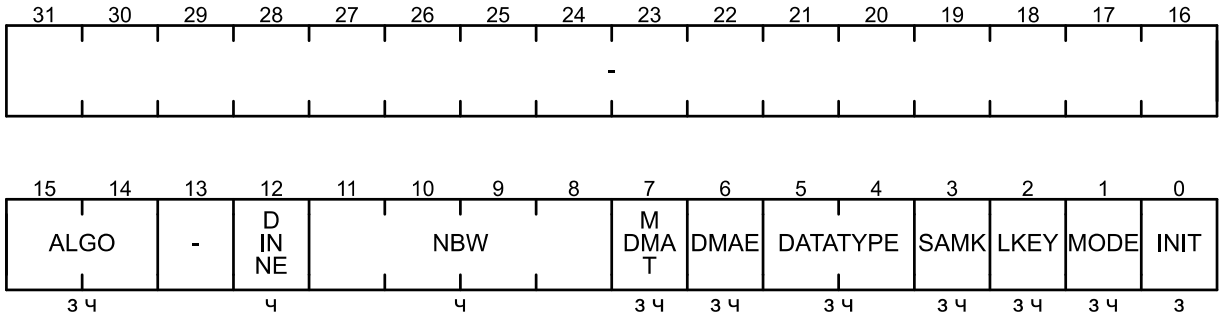
Базовый адрес: 2003_2000h

Смещение: + 040h (HR) Регистры хеш-суммы

Примечание – n – номер регистра хеш-суммы 0 – 7.

CR - регистр управления

Смещение: + 00h
Сброс: 0000_0000h



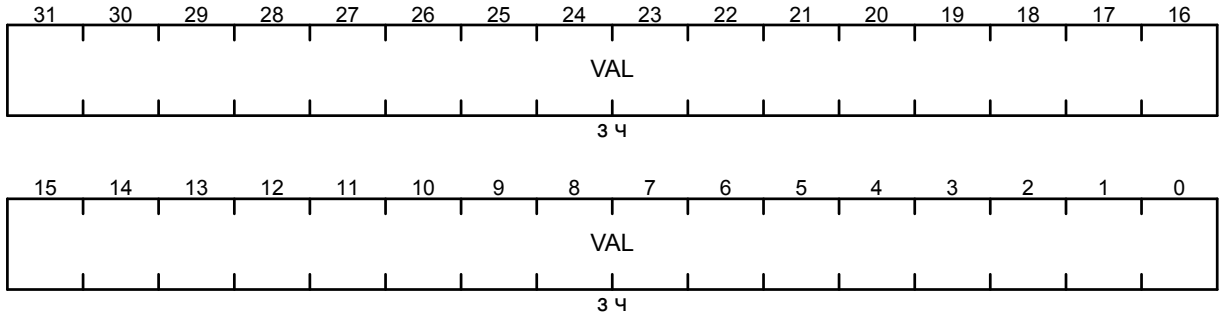
Поле	Биты	Описание	
ALGO	15-14	Выбор алгоритма для хеш-функции:	
		0	SHA-1
		1	MD5
		2	SHA-224
		3	SHA-256
DINNE	12	Регистр DIN не пуст Этот бит устанавливается, когда регистр HASH_DIN содержит допустимые данные (означает, что он хотя бы единожды был записан). Очищается либо установкой бита INIT (при инициализации) или при установке бита DCAL (окончание обработки сообщения) в «1».	
		0	Нет допустимых данных в DIN регистре
		1	Входной буфер содержит хотя бы одно слово данных
NBW	11-8	Количество слов в FIFO Это битовое поле отображает количество слов, в сообщении которые были помещены во входное FIFO. NBW инкрементируется (+1) когда происходит запись в DIN регистр при установленном DINNE в «1». Поле сбрасывается в «0000» при установке бита INIT в «1» или, когда начинается окончательный подсчёт хеш-суммы сообщения (DCAL устанавливается в «1» или DMA заканчивает последнюю транзакцию).	
		0	DINNE=0: Нет доступных слов в буфере DIN (буфер пуст, ни в DIN регистре, ни во входном FIFO нет доступных данных)
			DINNE=1: 1 слово было записано в буфер DIN (DIN регистр содержит 1 слово, входное FIFO пусто)

		1	2 слова записано в буфер DIN (HASH_DIN регистр и входное FIFO содержат по 1 слову)
		…	
		15	16 слов было записано в буфер DIN
MDMAT	7	Множественная передача ПДП. Бит устанавливается при подсчёте хеш-сумм больших файлов используя механизм прямого доступа к памяти.	
		0	DCAL автоматически устанавливается после окончания передачи данных с использованием ПДП
		1	DCAL автоматически не устанавливается после окончания передачи данных с использованием ПДП
DMAE	6	Включение механизма DMA 0: Передачи DMA выключены. 1: Включение передач DMA. Запрос DMA будет послан сразу, как только хеш-процессор будет готов принять данные.	
DATATYPE	5-4	Выбор формата данных Определяет формат данных для загрузки в регистр DIN:	
		0	32-битные данные. Эти данные пишутся в регистр DIN и напрямую используются в вычислении, без изменения порядка следования битов.
		1	16-битные данные, полуслова. Данные, записываемые в регистр DIN подразумеваются как 2 полуслова, и меняются местами до вычисления.
		2	8-битные данные, байты. Данные, записываемые в регистр DIN подразумеваются как 4 байта, и меняются местами до вычисления.
		3	битовые данные или битовая строка. Данные, записываемые в регистр DIN подразумеваются как 32 бита (первый бит строки на позиции 0), и меняются местами до вычисления (первый бит строки на позиции 31).
SAMK	3	Одинаковый ключ. Бит позволяет хеш-процессору использовать одинаковые внутренние и внешние ключи при использовании длинного ключа.	
		0	Используются разные ключи
		1	Используются одинаковые ключи
LKEY	2	Выбор длинного ключа В режиме HMAC бит определяет выбор длины ключа.	
		0	Короткий ключ (менее 64 байт)
		1	Длинный ключ (более 64 байт)
MODE	1	Выбор режима Бит выбирает HASH или HMAC режим алгоритма:	
		0	Режим HASH
		1	Режим HMAC. Бит LKEY должен быть установлен если длина используемого ключа больше 64 байт. Бит SAMK должен быть установлен если для длинного ключа используется одинаковый внутренний и внешний ключи.
INIT	0	Инициализация хеш-процессора. Запись этого бита в «1» производит сброс хеш-процессора и производит захват основных параметров (ALGO, MODE, LKEY и т. д.), таким образом хеш-процессор становится готов для вычисления новой хеш-суммы сообщения.	

–	31-16, 13	Зарезервировано

DIN – регистр ввода данных

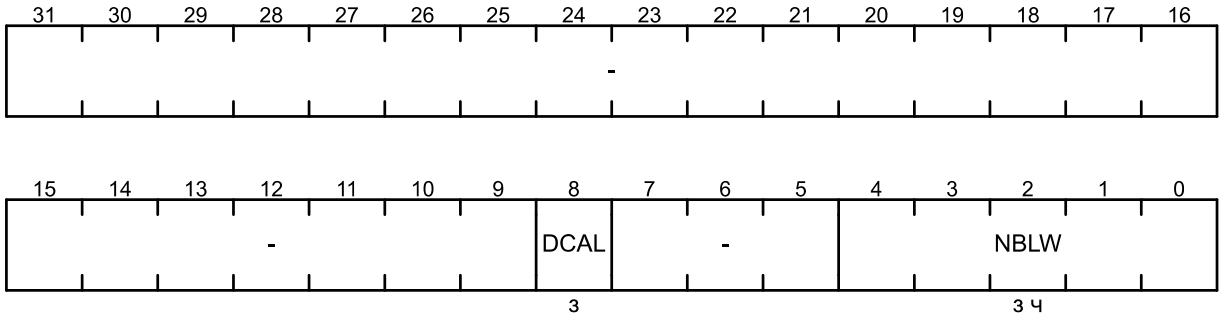
Смещение: + 04h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	Значение данных для ввода сообщения блоками по 512 бит. Биты DATATYPE должны быть предварительно настроены в регистре HASH_CR, чтобы получить правильное представление сообщения. Когда в регистр HASH_DIN записан блок из 16 слов, запускается вычисление промежуточной хеш-суммы: - путем записи новых данных в регистр DIN (первое слово следующего блока), если DMA не используется; - автоматически, если используется DMA. Когда последний блок был записан в регистр HASH_DIN, запускается окончательный расчет хеш-суммы (включая заполнение): - путем записи бита DCAL в 1 в регистре STR; - автоматически, если используется DMA и бит MDMAT установлен в «0». Когда выполняется расчет хеш-суммы сообщения (промежуточный или окончательный), любой новый доступ для записи в регистр DIN задерживается до тех пор, пока не завершится вычисление. При чтении регистра DIN осуществляется доступ к последнему слову, записанному в этом месте (ноль после сброса).

STR – регистр запуска

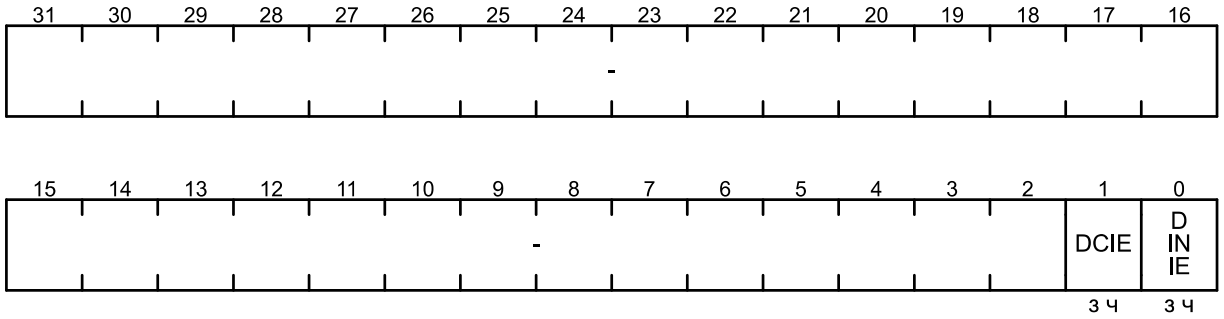
Смещение: + 08h
Сброс: 0000_0000h



Поле	Биты	Описание	
DCAL	8	Вычисление хеш-суммы Запись 1 в этот бит запускает заполнение сообщения с использованием ранее записанного значения NBLW и запускает вычисление окончательной хеш-суммы сообщения со всеми словами данных, записанными во входное FIFO с момента последней записи бита INIT в 1. Чтение этого бита всегда возвращает 0.	
NBLW	4-0	Количество допустимых битов в последнем слове сообщения в организации битовой строки хеш-процессора. При записи в эти биты с DCAL равным «0», они принимают значение с шины данных AHB: Когда эти биты пишутся, с DCAL равным «1», битовое поле не изменяется. Их чтение возвращает последнее значение, записанное в NBLW. Примечание – Эти биты необходимо настроить до установки бита DCAL, иначе они не будут учитываться. В частности, невозможно одновременно настроить NBLW и установить DCAL.	
		0	Все 32 бита последних данных, записанных в организацию битовой строки хеш-процессора (после замены данных), действительны.
		1	Действителен только бит [31] последних данных, записанных в организации битовой строки хеш-процессора (после замены данных).
		2	Действительны только биты [31:30] последних данных, записанных в организации битовой строки хеш-процессора (после замены данных).
		3	Действительны только биты [31:29] последних данных, записанных в организации битовой строки хеш-процессора (после замены данных).
		…	
		31	Действительны только бит [0] последних данных, записанных в организации битовой строки хеш-процессора (после замены данных).
-	31-9, 7-5	Зарезервировано	

IMR – регистр разрешения прерывания

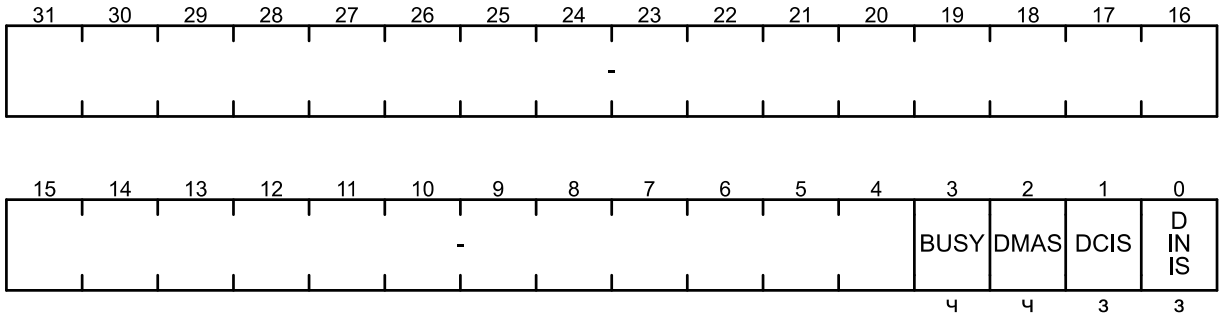
Смещение: + 0Ch
Сброс: 0000_0000h



Поле	Биты	Описание
DCIE	1	Разрешение прерывания по окончанию вычисления хеш-суммы сообщения
DINIE	0	Разрешение прерывания по готовности принятия нового блока данных
–	31-2	Зарезервировано

SR – регистр статуса

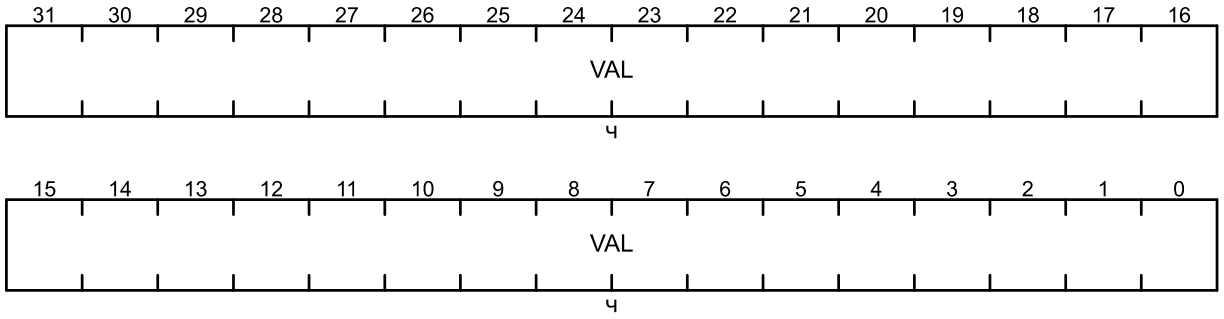
Смещение: + 10h
Сброс: 0000_0000h



Поле	Биты	Описание	
BUSY	3	Бит занятости	
		0	блок данных в данный момент не обрабатывается
		1	хеш-процессор обрабатывает блок данных
DMAS	2	Статус ПДП Этот бит предоставляет информацию об активности интерфейса ПДП. Он устанавливается при записи в HASH_CR.DMAE и сбрасывается, когда DMAE=0 и передача ПДП не выполняется. С этим битом не связано прерывание.	
		0	интерфейс ПДП отключен (DMAE=0) и передача не ведется
		1	интерфейс ПДП включен (DMAE=1) или идет передача
DCIE	1	Статус прерывания завершения вычисления сводки сообщения Этот бит устанавливается аппаратно при вычислении хеш-суммы (обработано все сообщение). Он очищается записью его в «0» или записью бита INIT в «1» в регистре HASH_CR.	
		0	Нет доступных результатов в регистрах HASH_HRx
		1	Расчет завершен, результат доступен в регистрах HASH_HRx. Прерывание генерируется, если бит DCIE установлен в регистре HASH_IMR.
DINIE	0	Состояние прерывания ввода данных. Этот бит устанавливается аппаратно, когда входной буфер готов принять новый блок (16 ячеек свободны). Он очищается записью в «0» или записью в регистр HASH_DIN. 0: Во входном буфере свободно менее 16 ячеек. 1: Во входной буфер можно ввести новый блок. Прерывание генерируется, если бит DINIE установлен в регистре HASH_IMR.	
–	31-4	Зарезервировано	

HR – массив регистров хеш-суммы

Смещение: HR + (4*n)h
Сброс: 0000_0000h



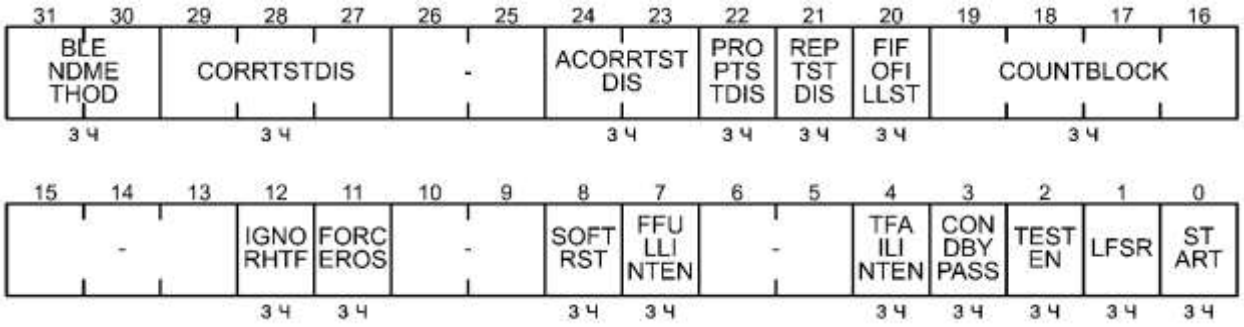
Поле	Биты	Описание
VAL	31-0	Регистры содержат результат вычисления хеш-суммы: 1 В регистры HR0, HR1, HR2, HR3 и HR4 помещается результат вычисления хеш-суммы с использованием алгоритма SHA1. Обратите внимание, что в этом случае регистры с HR5 по HR7 не используются и читаются как ноль. 2 В регистры HR0, HR1, HR2 и HR3 помещается результат вычисления хеш-суммы с использованием алгоритма MD5. Обратите внимание, что в этом случае регистры с HR4 по HR7 не используются и читаются как ноль. 3 В регистры от HR0 до HR6 помещается результат вычисления хеш-суммы с использованием алгоритма SHA224. Обратите внимание, что в этом случае регистр HR7 не используется и читается как ноль. 4 В регистры от HR0 до HR7 помещается результат вычисления хеш-суммы с использованием алгоритма SHA256. Если происходит чтение к одному из этих регистров, когда ядро хеш-процессора вычисляет промежуточную или окончательную хеш-сумму сообщения (то есть, когда бит DCAL был записан в «1»), то чтение приостанавливается (возникает состояние ожидания на шине AHB) до завершения расчётов.

A.11 Регистры генератора случайных чисел ИГСЧ

Базовый адрес: 3000_4000h

CR – регистр конфигурации блока ИГСЧ

Смещение: + 00h
Сброс: 0004_0000h

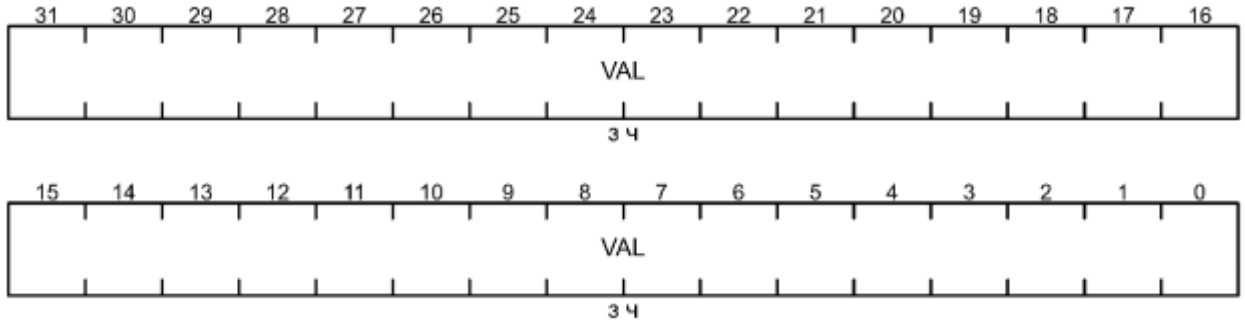


Поле	Биты	Описание	
BLENDMETHOD	31-30	Выбор метода смешивания	
		00	Объединение
		01	XOR_LEVEL_1
		10	XOR_LEVEL_2
		11	Исключение систематической ошибки Фон-Неймана
CORRTSTDIS	29-27	Отключение отдельных проверок со смещением в тесте на корреляцию – разные кольцевые генераторы (RO):	
		xx1	по сравнению с такой же выборкой другого кольцевого генератора RO→(0) нулевое смещение
		x1x	по сравнению с предыдущей выборкой другого кольцевого генератора RO→(-1) смещение -1
		1xx	по сравнению со следующей выборкой другого кольцевого генератора RO→(+1) смещение +1
ACORRTSTDIS	24-23	Отключение отдельных проверок со смещением в тесте на автокорреляцию – один и тот же кольцевой генератор (RO):	
		x1	по сравнению со следующим образцом того же самого RO→(0) нулевое смещение
		1x	по сравнению с более поздним образцом того же самого RO→(+1) смещение +1
PROPTSTDIS	22	Все тесты на пропорции отключаются с помощью этого бита	
REPTSTDIS	21	Все тесты на повторения отключаются с помощью этого бита	
FIFOFILLST	20	Разрешение записи выборок в FIFO во время запуска	
COUNTBLOCK	19-16	Число 128-битных блоков, используемых в функции последующей обработки. Нулевое значение не допускается	
IGNORHTF	12	Результаты тестов работоспособности во время запуска и во время работы не влияют на смену состояния КА (конечного автомата). Он также обходит фазу StartUp КА.	
FORCEROS	11	Принудительный запуск генераторов, когда FIFO заполнен	

SOFTRST	8	Очистка содержимого информационного канала и переход КА в состояние сброса	
		0	Нормальный режим
		1	Компоненты информационного канала – тесты работоспособности, функция обработки и FIFO – очищаются. Этот бит не очищается автоматически. Регистры конфигурации сохраняют свои значения. Отправляет КА в состояние сброса.
FFULLINTEN	7	Включить прерывание, если FIFO заполнен.	
TFAILINTEN	4	Включить прерывание, если какой-либо из тестов работоспособности провален	
CONDBYPASS	3	0	Используется функция обработки (обычный режим)
		1	Функция обработки пропускается (для непосредственного наблюдения за источником энтропии)
TESTEN	2	Выбор входных данных для функции обработки и непрерывных тестов:	
		0	Источник шума (обычный режим)
		1	Регистр с тестовыми данными (тестовый режим)
LFSR	1	Выбор между ИГСЧ с асинхронными независимыми генераторами (при значении «0») и псевдослучайным генератором с синхронными генераторами для целей моделирования (при значении «1»).	
START	0	Запуск/включение ИГСЧ	
-	26-25, 15-13, 10-9, 6-5	Зарезервировано	

FIFOLEV – регистр уровня заполнения FIFO

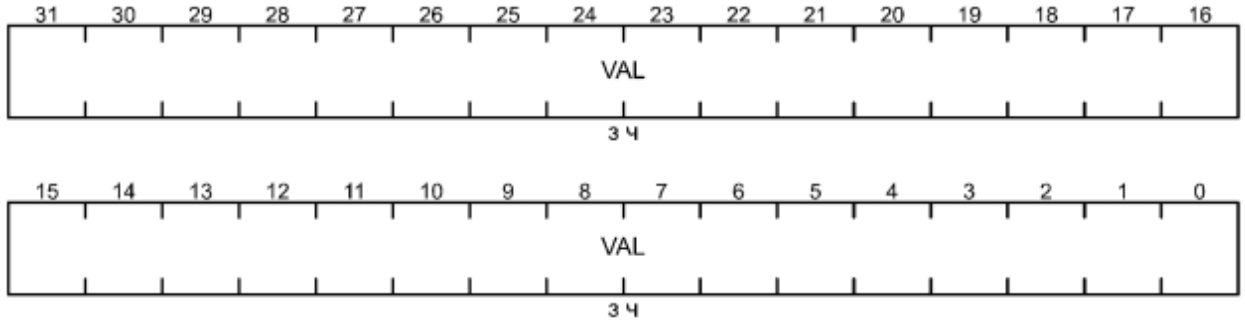
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Количество доступных 32-битных случайных слов в буфере FIFO. Любая запись в этот регистр очищает флаг FifoFull в регистре состояния.

FIFOTHR – регистр порогового значения FIFO

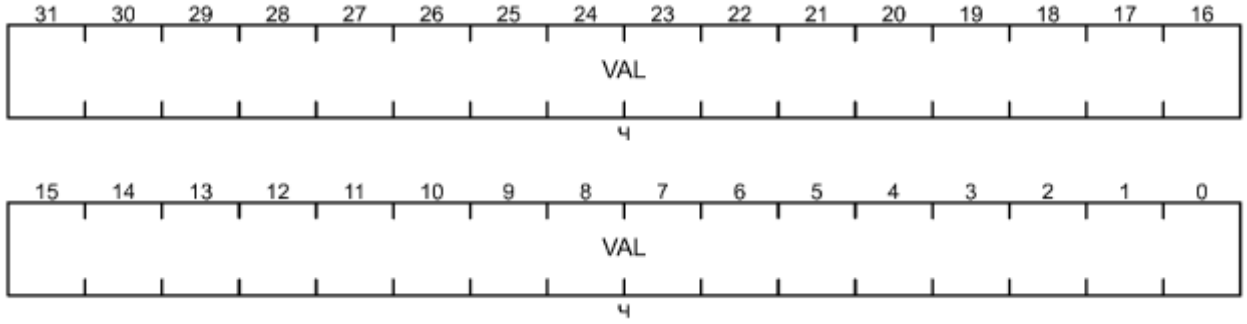
Смещение: + 08h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	Пороговый уровень буфера FIFO, ниже которого модуль начинает пополнять FIFO Выражен в количестве 128-битных блоков.

FIFODEP – регистр максимального количества слов в FIFO

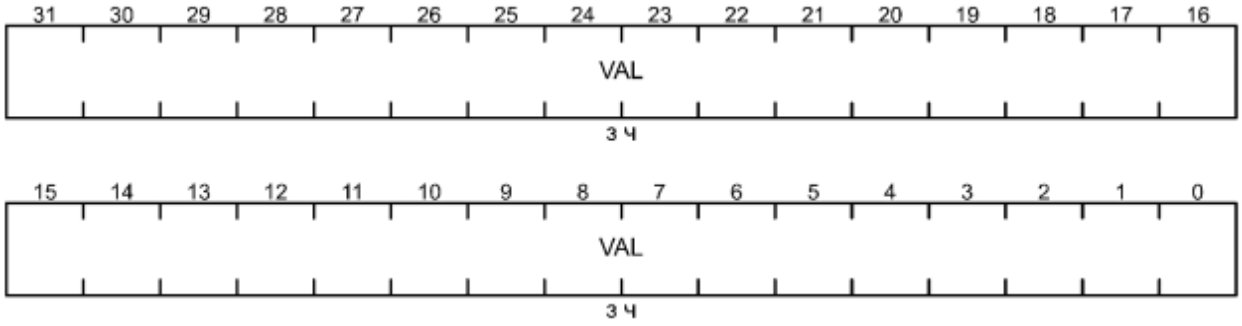
Смещение: + 0Ch
Сброс: 0000_0100h



Поле	Биты	Описание
VAL	31-0	Максимальное количество 32-разрядных слов, которые могут быть сохранены в FIFO. При чтении возвращается значение 256.

KEY0 – регистр ключа 0

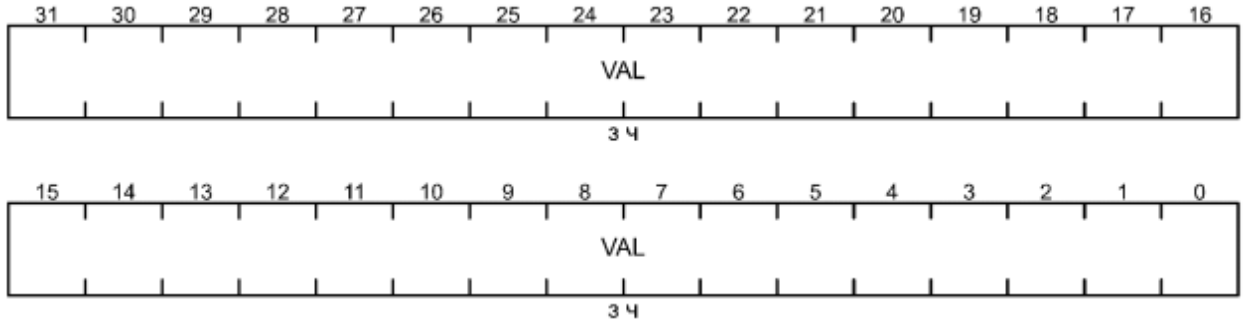
Смещение: + 10h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	Ключ 0. Байты ключа AES [0-3] (Набор битов четырех регистров ключей формирует 128-битный ключ AES, используемый для функции обработки)

KEY1 – регистр ключа 1

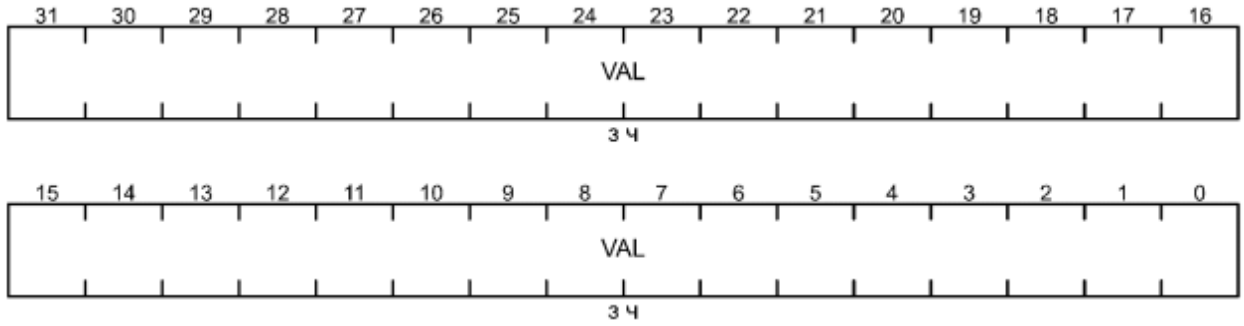
Смещение: + 14h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	Ключ 1. Байты ключа AES [4-7] (Набор битов четырех регистров ключей формирует 128-битный ключ AES, используемый для функции обработки)

KEY2 – регистр ключа 2

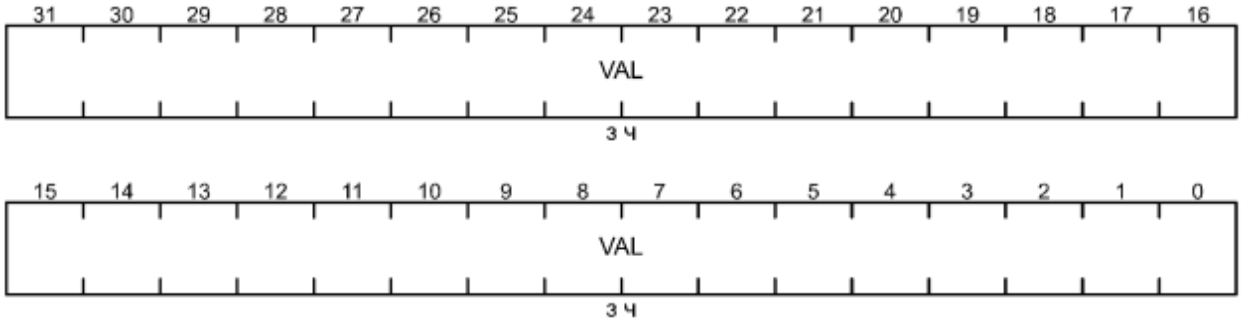
Смещение: + 18h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	Ключ 2. Байты ключа AES [8-11] (Набор битов четырех регистров ключей формирует 128-битный ключ AES, используемый для функции обработки)

KEY3 – регистр ключа 3

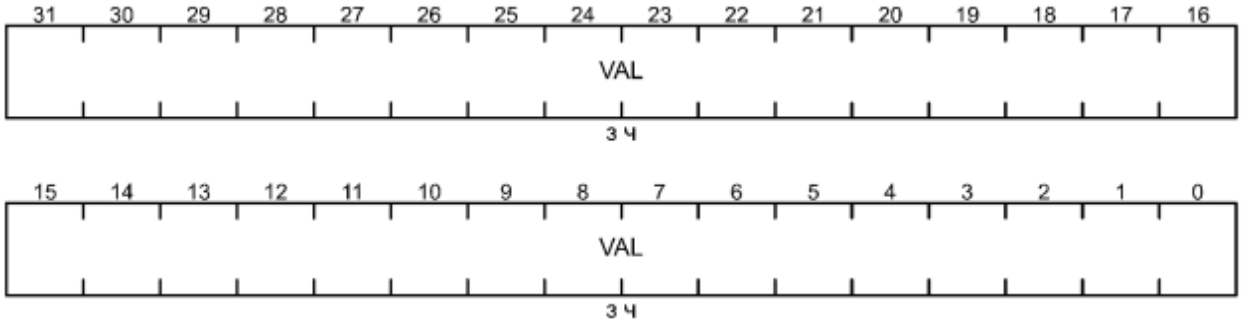
Смещение: + 1Ch
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	Ключ 3. Байты ключа AES [12-15] (Набор битов четырех регистров ключей формирует 128-битный ключ AES, используемый для функции обработки)

TESTDATA – регистр с тестовыми данными

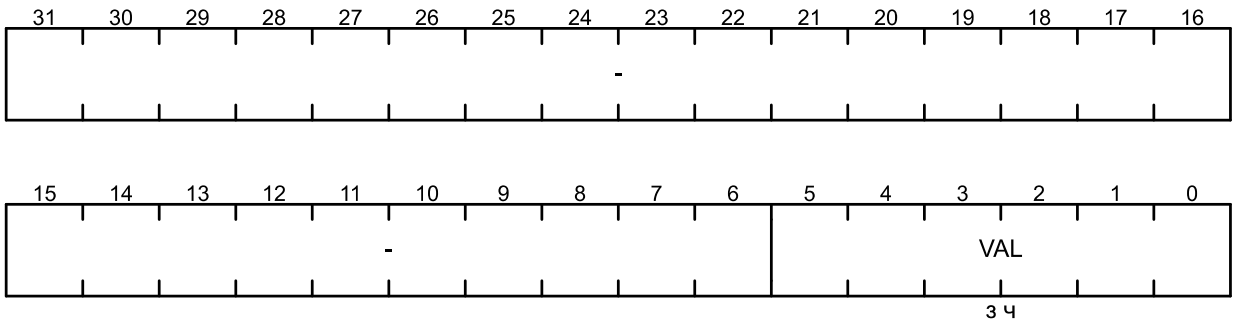
Смещение: + 20h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Передача известных данных в функцию обработки или в непрерывные тесты

REPTSTCUTOFF – регистр предельного значения для теста на повторение

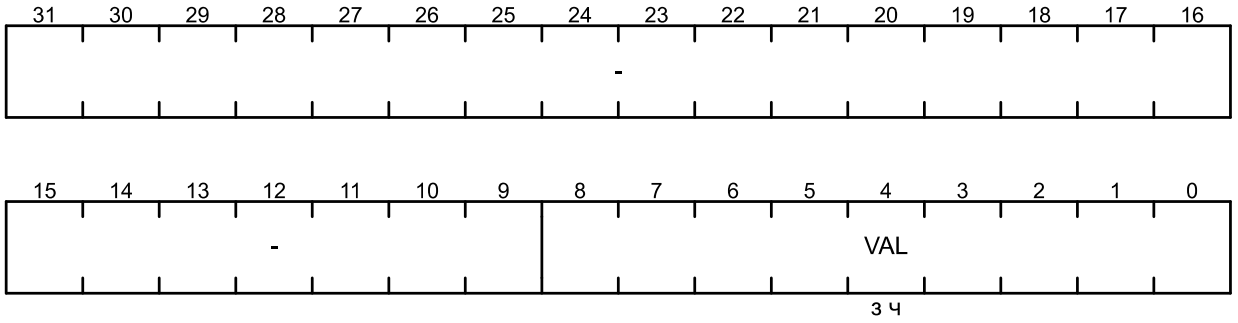
Смещение: + 24h
Сброс: 0000_0006h



Поле	Биты	Описание
VAL	5-0	Предельное значение для теста на повторение
-	31-6	Зарезервировано

PROPTESTCUTOFF - регистр предельного значения для теста на пропорции

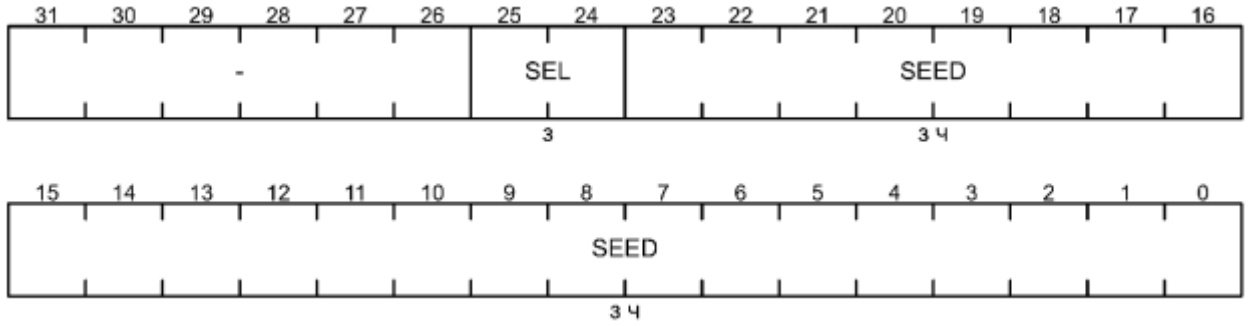
Смещение: + 28h
Сброс: 0000_003Eh



Поле	Биты	Описание
VAL	8-0	Предельное значение для тестов на пропорции
-	31-9	Зарезервировано

LFSRSEED – регистр сдвига с линейной обратной связью

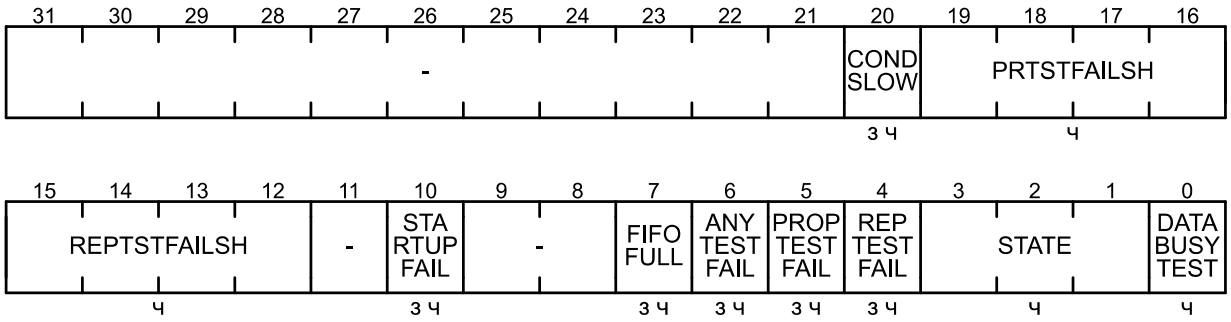
Смещение: + 2Ch
Сброс: 00FF_FFFFh



Поле	Биты	Описание
SEL	25-24	Общий индекс LFSR, для которого следует использовать значение инициализации
SEED	23-0	Начальное значение LFSR при инициализации
-	31-26	Зарезервировано

STAT – регистр состояния ИГСЧ

Смещение: + 30h
Сброс: 0000_0000h

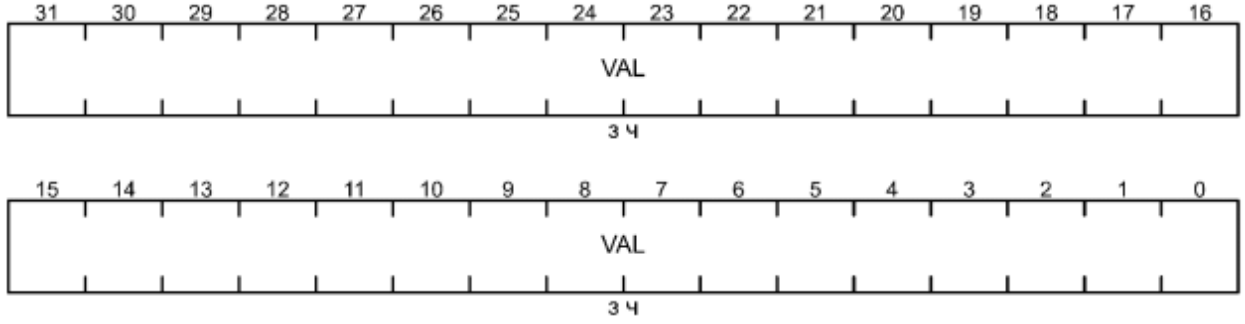


Поле	Биты	Описание	
CONDSLOW	20	Функция обработки прорабатывает данные медленнее, чем они ей выдаются	
PRTSTFAILSH	19-16	Доля проваленных тестов на пропорции	
REPTSTFAILSH	15-12	Доля проваленных тестов на повторения	
STARTUPFAIL	10	Сбой теста (тестов) запуска	
FIFOFULL	7	Статус полного буфера FIFO	
ANYTESTFAIL	6	Любой из включенных тестов работоспособности завершился провалом	
PROPTESTFAIL	5	Провал теста на пропорции	
REPTESTFAIL	4	Провал теста (тестов) на повторения	
STATE	3-1	Состояние контрольного автомата КА	
		000	Сброс
		001	Запуск

		010	Холостой ход / FIFO заполнен
		011	Не используется
		100	Заполннение FIFO
		101	Ошибка
		110	Не используется
		111	Не используется
DATABUSYTEST	0	Устанавливается во время процесса записи данных в регистр TESTDATA	
-	31-21, 11, 9-8	Зарезервировано	

WARMPERIOD – регистр времени прогрева

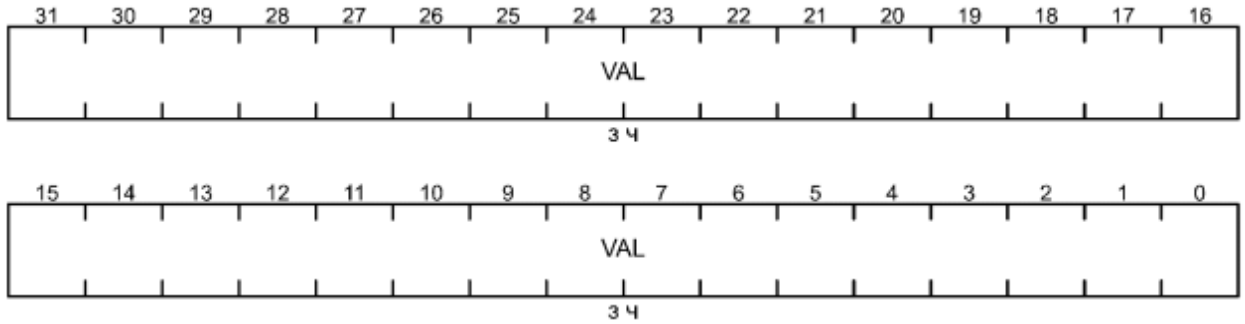
Смещение: + 34h
Сброс: 0000_0200h



Поле	Биты	Описание
VAL	31-0	Количество тактов, ожидаемых во время процесса прогрева

DISOSC – регистр отключения колец генератора

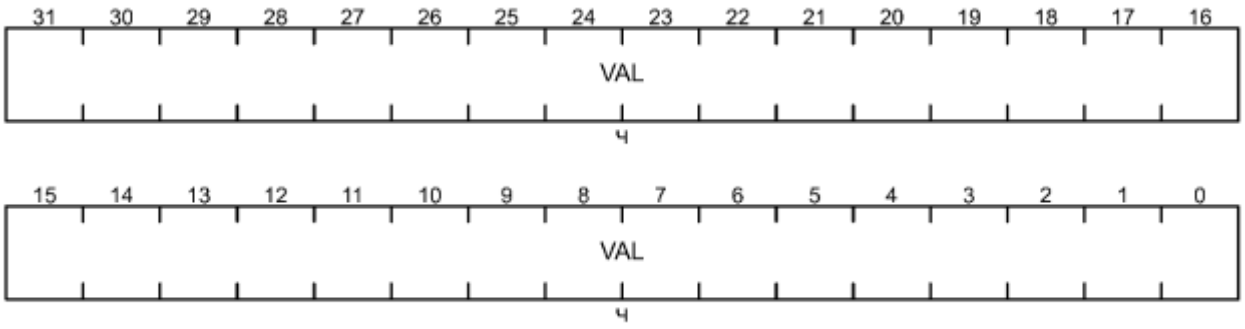
Смещение: + 38h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	Независимое отключение кольца генератора от 0 до N-1. По одному биту на каждое кольцо.

SAMPERIOD – регистр периода дискретизации

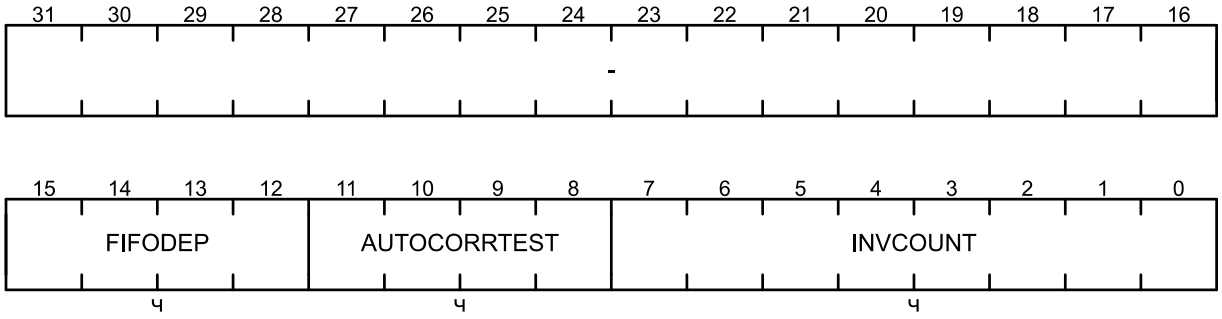
Смещение: + 44h
Сброс: 0000_0FFFh



Поле	Биты	Описание
VAL	31-0	Количество тактов ожидания между моментами выборки (дискретизации).

HWCFG – регистр аппаратной конфигурации

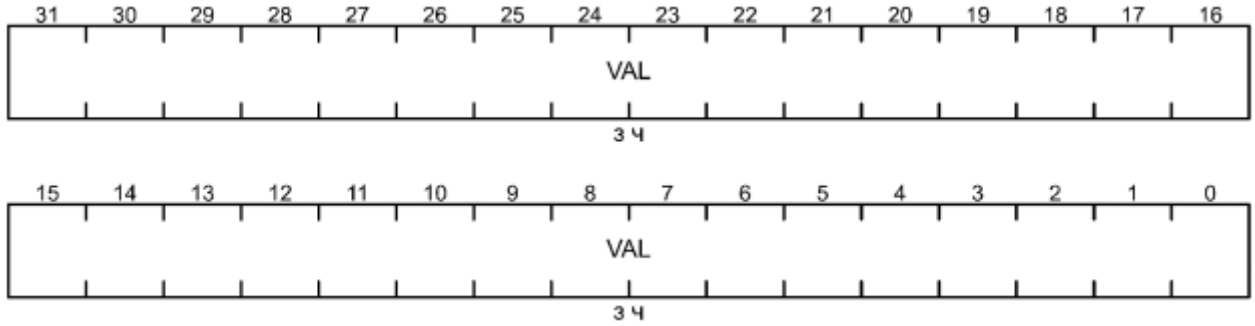
Смещение: + 58h
Сброс: 0000_8105h



Поле	Биты	Описание
FIFODEPTH	15-12	Значение глубины буфера FIFO. При чтении возвращается значение 8
AUTOCORRTEST	11-8	Количество тестов на автокорреляцию. При чтении возвращается значение 1
INVCOUNT	7-0	Количество инверторов. При чтении возвращается значение 5
-	31-16	Зарезервировано

COOLDPERIOD – регистр значения периода ожидания в холодном режиме

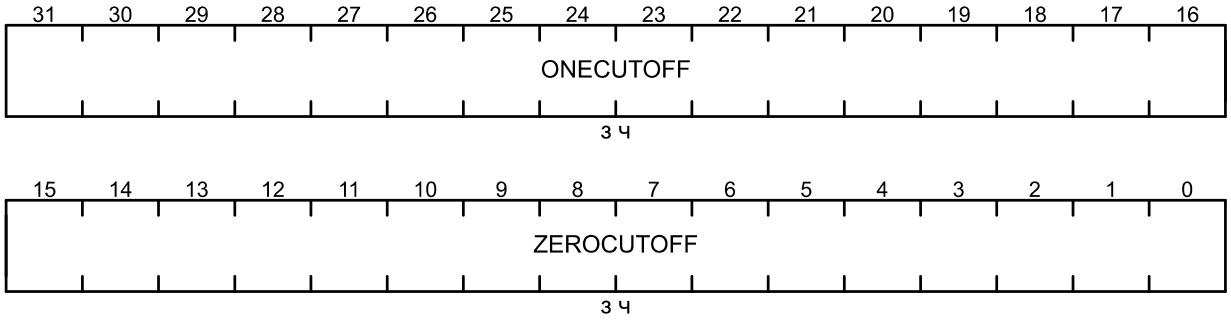
Смещение: + 5Ch
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	Количество тактов во время цикла ожидания в холодном режиме

AUTOCORRTESTCUTOFF0 – регистр граничных значений для теста на автокорреляцию

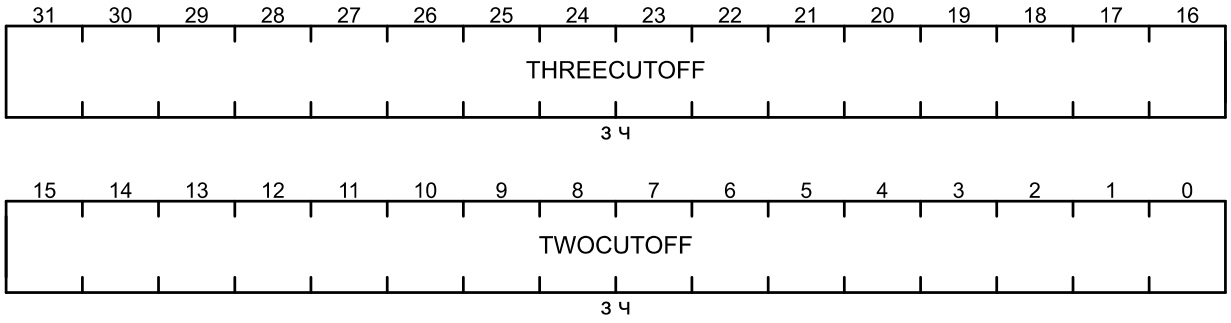
Смещение: + 60h
Сброс: 007F_007Fh



Поле	Биты	Описание
ONECUTOFF	31-16	Граничное значение для задержки выборки +1 в тесте на автокорреляцию
ZEROCUTOFF	15-0	Граничное значение для задержки выборки 0 в тесте на автокорреляцию

AUTOCORRTESTCUTOFF1 - регистр граничных значений для теста на автокорреляцию

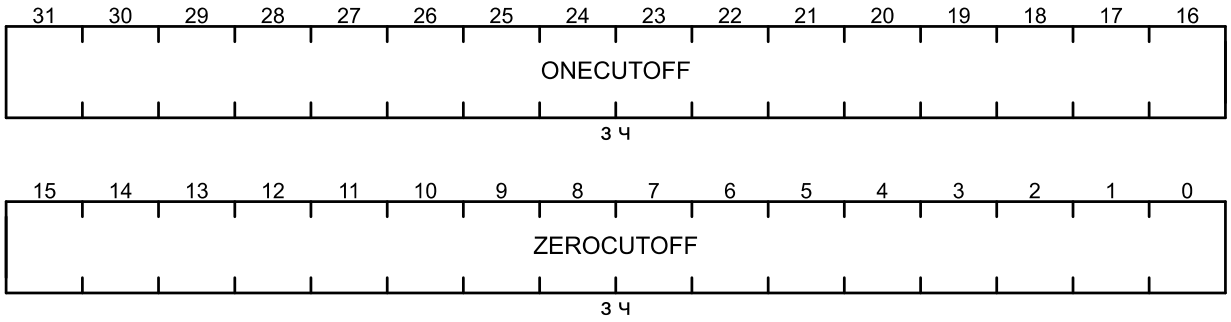
Смещение: + 64h
Сброс: 007F_007Fh



Поле	Биты	Описание
THREECUTOFF	31-16	Граничное значение для задержки выборки +3 в тесте на автокорреляцию
TWOCUTOFF	15-0	Граничное значение для задержки выборки +2 в тесте на автокорреляцию

CORRTESTCUTOFF0 - регистр граничных значений для теста на корреляцию

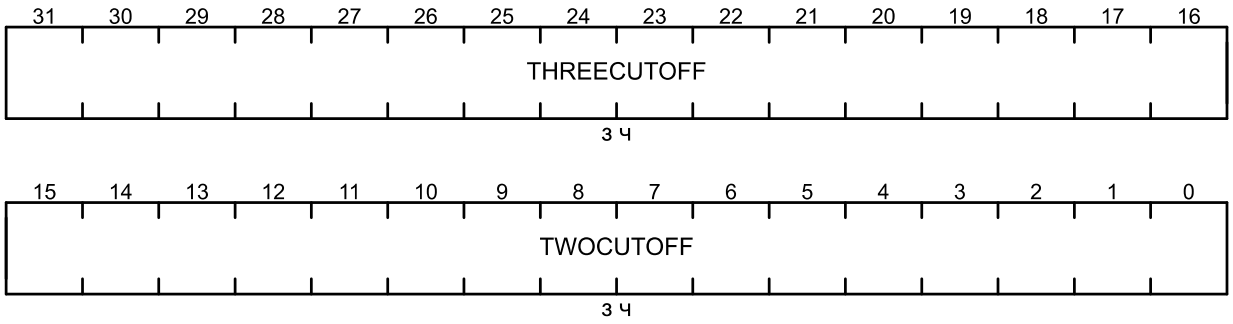
Смещение: + 68h
Сброс: 007F_007Fh



Поле	Биты	Описание
ONECUTOFF	31-16	Граничное значение для сдвига/задержки выборки +/-1 в тесте на корреляцию
ZEROCUTOFF	15-0	Граничное значение для сдвига/задержки выборки 0 в тесте на корреляцию

CORRTESTCUTOFF1 - регистр граничных значений для теста на корреляцию

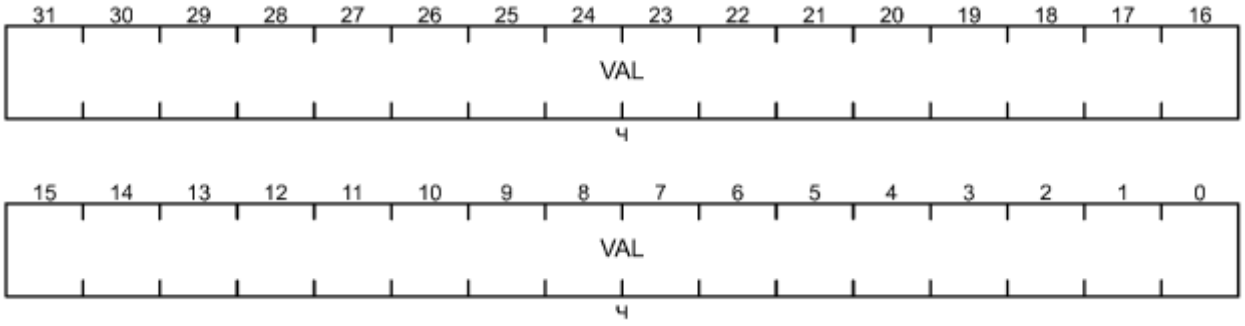
Смещение: + 6Ch
Сброс: 007F_007Fh



Поле	Биты	Описание
THREECUTOFF	31-16	Граничное значение для сдвига/задержки выборки +/-3 в тесте на корреляцию
TWOCUTOFF	15-0	Граничное значение для сдвига/задержки выборки +/-2 в тесте на корреляцию

AUTOCORRTESTFAILED – регистр, хранящий значения индексов кольцевых генераторов, проваливших тест на автокорреляцию

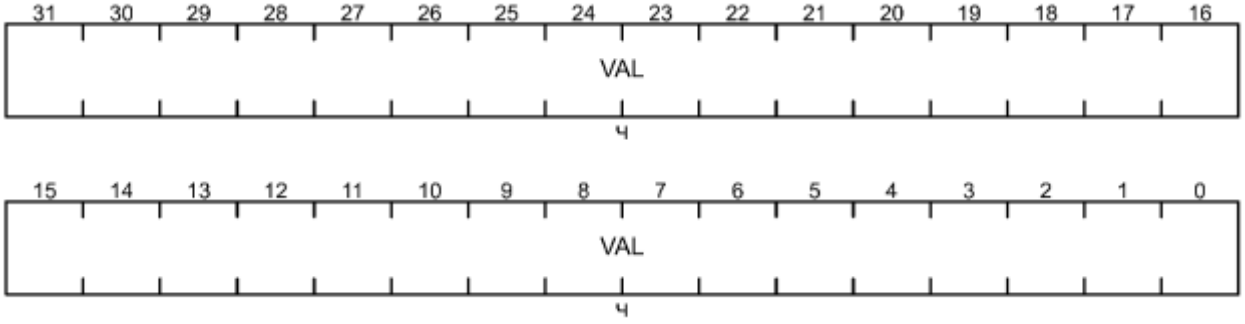
Смещение: + 70h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	Кольцевой генератор/генераторы, провалившие тест на автокорреляцию

CORRTESTFAILED - регистр, хранящий значения индексов кольцевых генераторов, проваливших тест на корреляцию

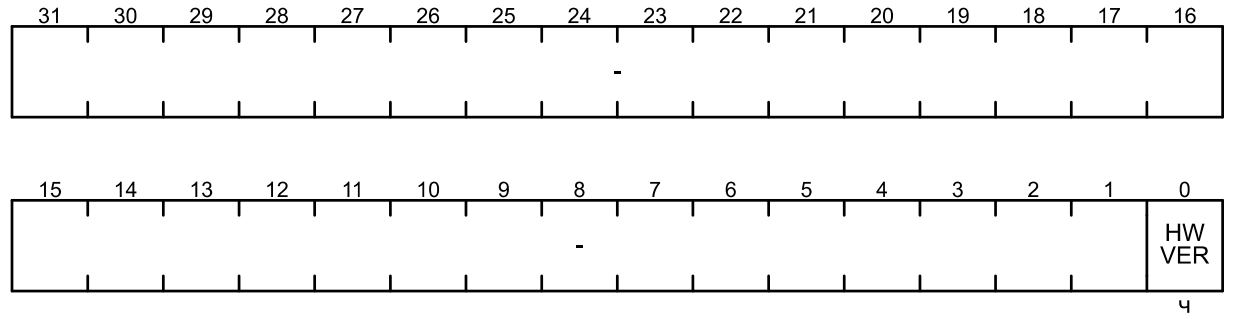
Смещение: + 74h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	Кольцевые генераторы, провалившие тест на корреляцию

HWVERSION – регистр версии оборудования

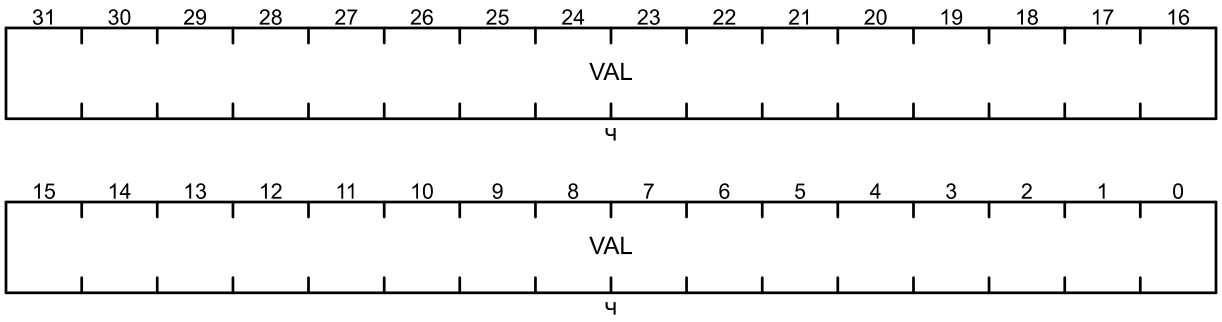
Смещение: + 7Ch
Сброс: 0000_0001h



Поле	Биты	Описание
HWVER	0	Установлено в «1» для этой версии
-	31-1	Зарезервировано

FIFO<i> – регистр буфера FIFO

Смещение: +80 +(4*i)h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	Ячейка буфера FIFO

Примечание – i – номер регистра FIFO от 0 до 31.

A.12 Регистры блока CRYPTO

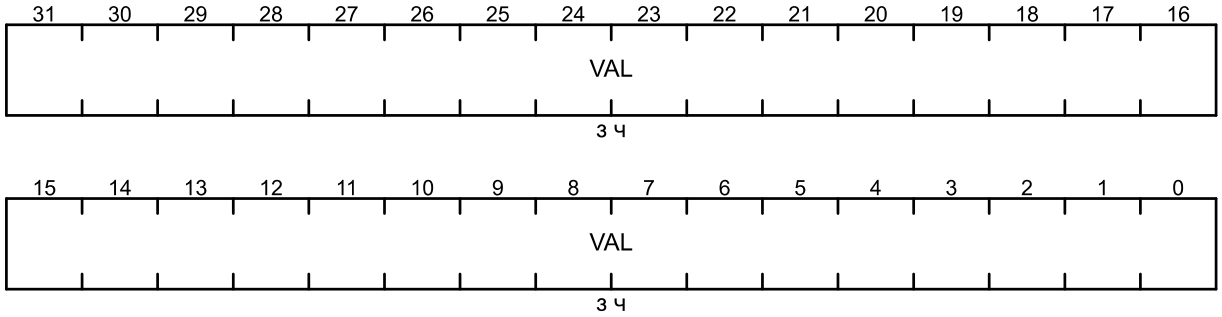
Базовый адрес: 2002_0000h

Смещение:		
	+ 00h (IV)	Массив регистров вектора инициализации
	+ 10h (TEXT_IN)	Массив регистров входного текста
	+ 20h (KEY)	Массив регистров раундовых ключей
	+ 70h (TEXT_OUT)	Массив регистров выходного текста

Примечание – n – номер регистра от 0 до 3;
k – номер регистра ключа от 0 до 7.

IV – массив регистров вектора инициализации

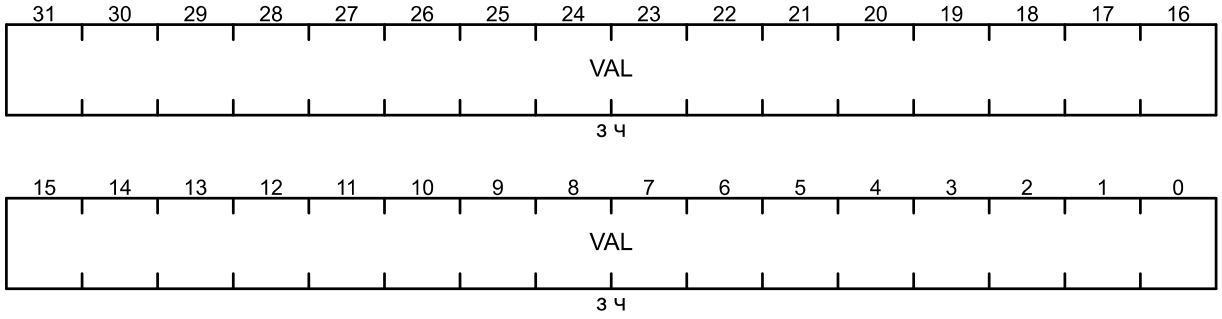
Смещение: IV +(4*n)h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	32-разрядный регистр, хранящий часть значения инициализационного вектора.

TEXT_IN – массив регистров входного текста

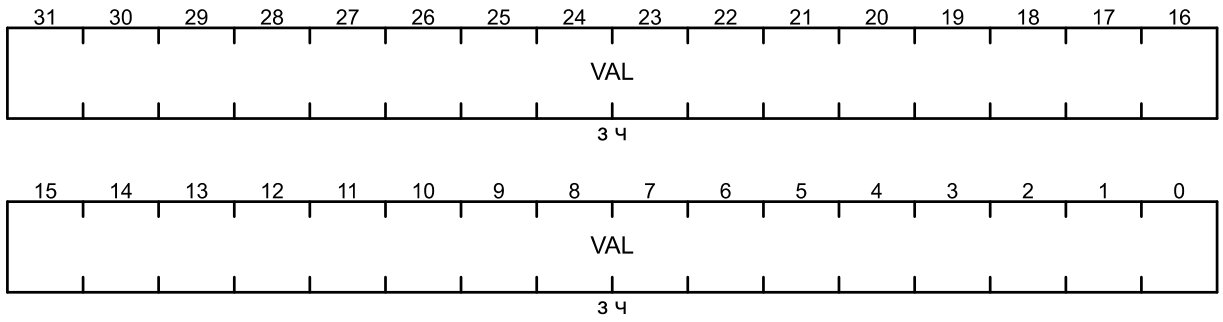
Смещение: TEXT_IN +(4*n)h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	32-разрядный регистр, хранящий часть значения входного текста.

KEY – массив регистров раундовых ключей

Смещение: KEY + (4*k)h
Сброс: 0000_0000h

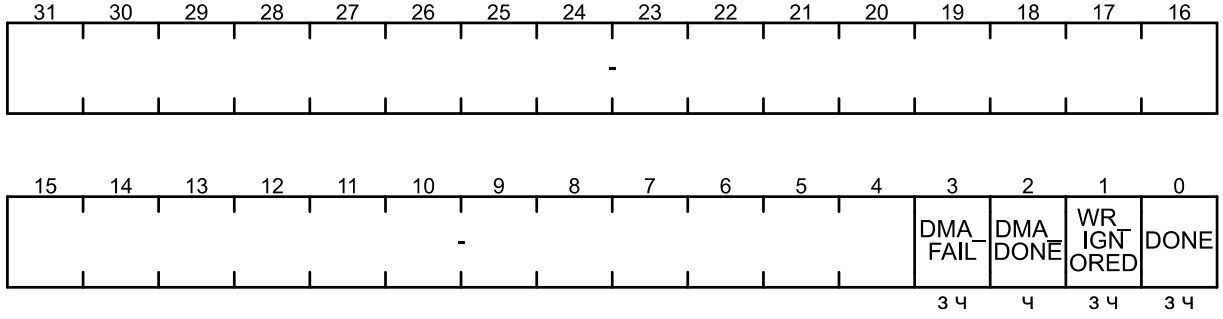


Поле	Биты	Описание
VAL	31-0	32-разрядный регистр, хранящий значение ключа.

Примечание – Данный набор регистров используется для формирования раундовых ключей выбранного алгоритма шифрования (если выбран алгоритм AES-128, то в формировании раундовых ключей участвуют только KEY_0, …, KEY_3).

IRQ_ENABLE – регистр разрешения прерываний

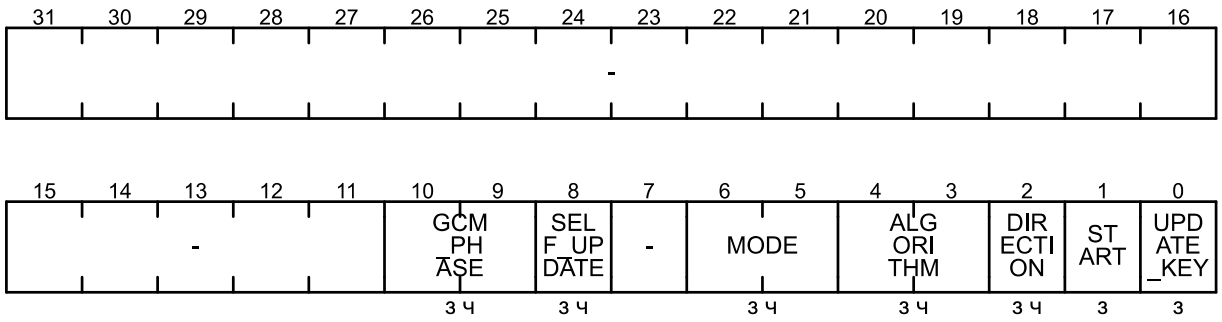
Смещение: + 40h
Сброс: 0000_0000h



Поле	Биты	Описание
DMA_FAIL	3	Если бит установлен, то функционал бита IRQ.DMA_FAIL разрешен.
DMA_DONE	2	Если бит установлен, то функционал бита IRQ.DMA_DONE разрешен. Бит доступен только для чтения и отображает значение бита INTERRUPT управляющего слова дескриптора DMA.
WR_IGNORED	1	Если бит установлен, то функционал бита IRQ.WR_IGNORED разрешен.
DONE	0	Если бит установлен, то функционал бита IRQ.DONE разрешен.
-	31-4	Зарезервировано

CONTROL – регистр управления функционированием

Смещение: + 44h
Сброс: 0000_0000h

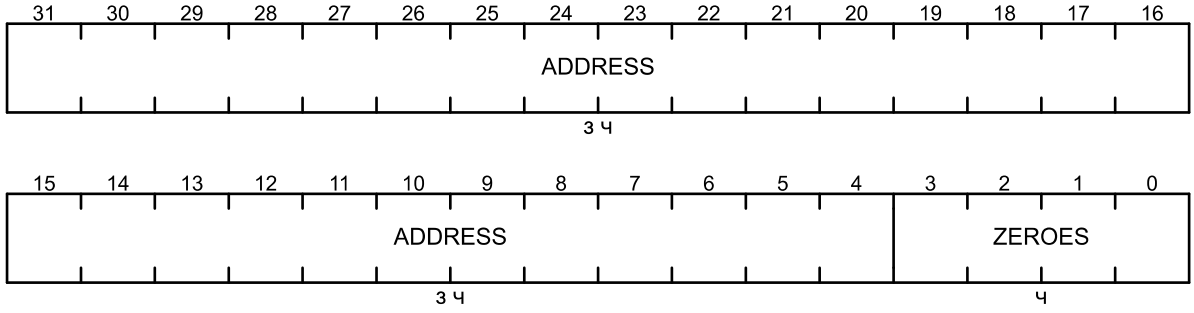


Поле	Биты	Описание	
GCM_PHASE	10-9	Фаза обработки пакета в режиме GALOIS_COUNTER_MODE. Если режим выполнения криптографической операции не GALOIS_COUNTER_MODE, то значение поля игнорируется.	
		0	GCM_INIT;
		1	GCM_HEADER;
		2	GCM_PAYLOAD;
		3	GCM_LAST_BLOCK.
SELF_UPDATE	8	Разрешение автоматического обновления содержимого регистров IV_* при завершении выполнения криптографической операции.	
GCM_PHASE	10-9	Фаза обработки пакета в режиме GALOIS_COUNTER_MODE. Если режим выполнения криптографической операции не GALOIS_COUNTER_MODE, то значение поля игнорируется.	
MODE	6-5	Выбор режима выполнения криптографической операции:	
		0	ELECTRONIC_CODEBOOK (ECB);
		1	CIPHER_BLOCK_CHAINING (CBC);
		2	COUNTER_MODE (CTR);
		3	GALOIS_COUNTER_MODE (GCM).
ALGORITHM	4-3	Выбор алгоритма криптографической операции	
		0	AES_128
		1	AES_256
		2	MAGMA
		3	KUZNECHIK
DIRECTION	2	Направление криптографической операции (0 – Encryption, 1 - Decryption)	
START	1	Запрос на выполнение криптографической операции. Непосредственный старт может быть отложен до завершения обновления раундовых ключей (спровоцированное установленным битом UPDATE_KEY или выполненное из-за отсутствия раундовых ключей для выбранного криптографического алгоритма). При чтении всегда возвращает 0.	

UPDATE_KEY	0	Принудительно выполнить обновление раундовых ключей шифрования на основе текущего значения регистров {KEY_0, ..., KEY_7} или {KEY_0, ..., KEY_3} для алгоритма AES-128. При чтении всегда возвращает 0.
-	31-11, 7	Зарезервировано

BASE_DESCRIPTOR – регистр адреса базового дескриптора

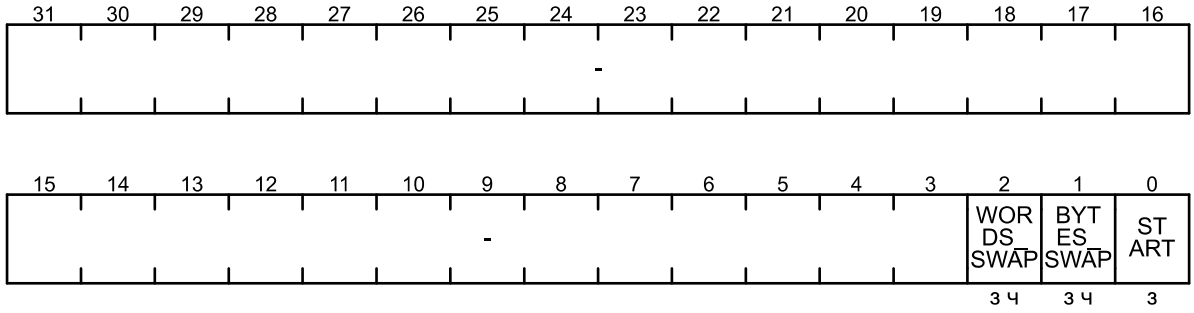
Смещение: + 48h
Сброс: 0000_0000h



Поле	Биты	Описание
ADDRESS	31-4	Адрес, который указывает на первый дескриптор DMA в цепочке, который будет прочитан из памяти после записи 1 в бит START регистра DMA_CONTROL
ZEROES	4-0	Младшие биты адреса, должны быть равны 0. Для обеспечения корректности записываемого адреса дескриптора доступны только для чтения.

DMA_CONTROL – регистр управления функционированием DMA

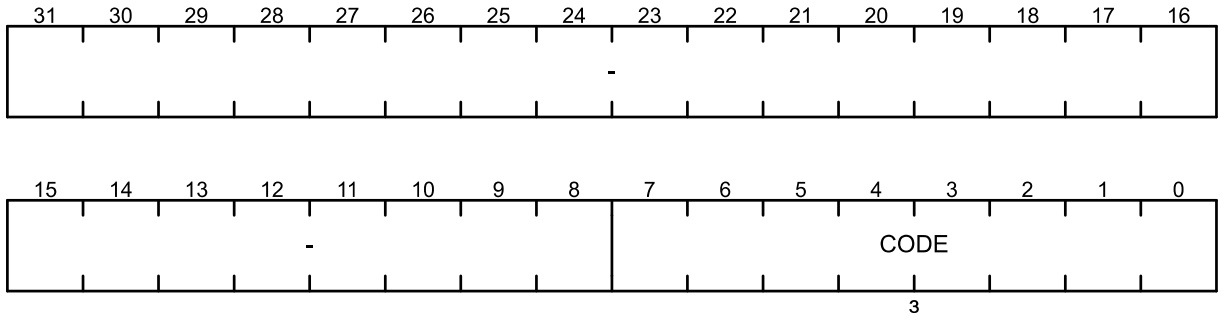
Смещение: + 4Ch
Сброс: 0000_0000h



Поле	Биты	Описание
WORDS_SWAP	2	Если бит сброшен, то заполнение регистров TEXT_IN_* считанными из памяти словами входного текста идет в порядке увеличения индекса регистра. Если бит установлен, то последовательность заполнения регистров TEXT_IN_* меняется на обратную. Аналогичное влияние оказывается на пересылку выходных слов текста из регистров TEXT_OUT_* в память. Данный бит не меняет порядок считывания слов дескриптора.
BYTES_SWAP	1	Если бит установлен, то перед записью в регистры TEXT_IN_* в прочитанном слове будет изменен порядок следования байт. Пересылаемые из TEXT_OUT_* слова будут выставлены на шину AHB с инвертированным порядком байт. Данный бит не оказывает влияние на считываемые слова дескриптора
START	0	Установленный бит запускает последовательность операций в режиме прямого доступа к памяти. При чтении всегда возвращает 0.
-	31-3	Зарезервировано

TERMINATE – регистр экстренного завершения операции

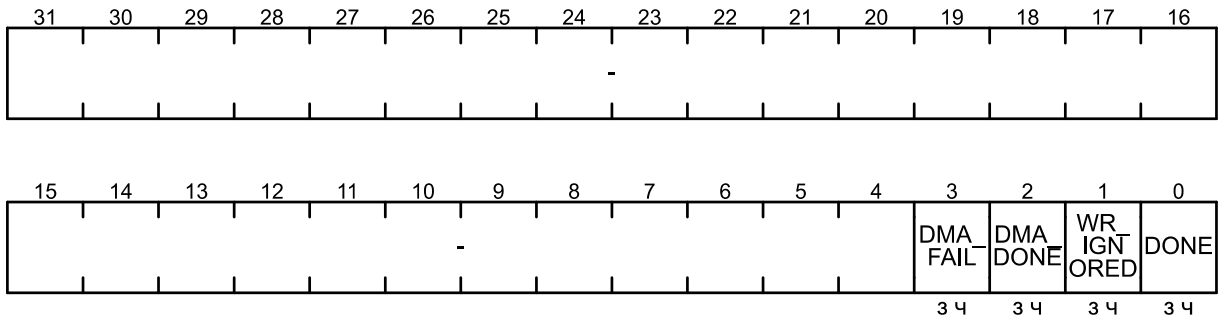
Смещение: + 50h
Сброс: 0000_0000h



Поле	Биты	Описание
CODE	7-0	Запись в данное поле значения 0xD0 приведет к сбросу внутреннего состояния модулей криптографических операций и хранилища раундовых ключей. Запись других значений будет проигнорирована. При чтении всегда возвращает 0.
-	31-8	Зарезервировано

IRQ – регистр прерываний, ожидающих обработки

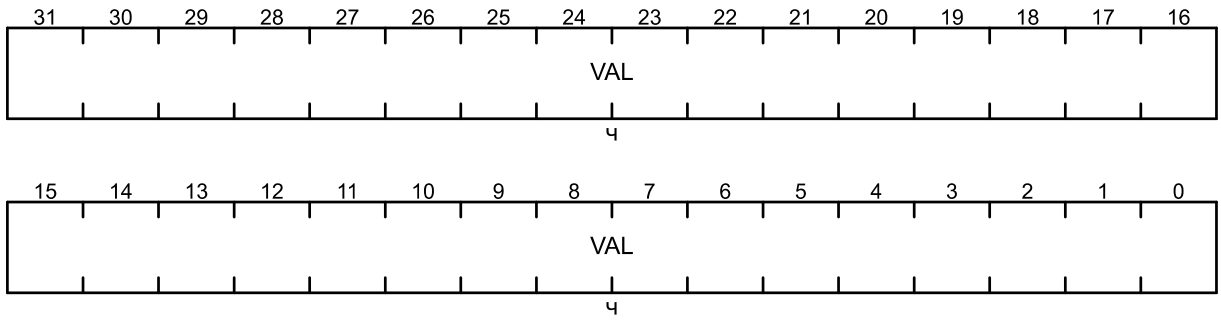
Смещение: + 60h
Сброс: 0000_0000h



Поле	Биты	Описание
DMA_FAIL	3	Бит устанавливается, если установлен соответствующий бит регистра IRQ_ENABLE, при преждевременном завершении работы DMA из-за обнаружения ошибки. Бит может быть сброшен записью 1.
DMA_DONE	2	Бит устанавливается, если бит INTERRUPT управляющего слова текущего дескриптора DMA был установлен (его состояние отображается в соответствующий бит регистра IRQ_ENABLE), при успешном завершении выполнения дескриптора DMA (поля STATUS.BAD_DESCRIPTOR и STATUS.AHB_ERROR содержат нулевое значение). Бит может быть сброшен записью 1.
WR_IGNORED	1	Бит устанавливается, если установлен соответствующий бит регистра IRQ_ENABLE, при выполнении записи в регистр CONTROL, TEXT_IN_* или IV_* пока STATUS.READY сброшен. Бит может быть сброшен записью 1.
DONE	0	Бит устанавливается, если установлен соответствующий бит регистра IRQ_ENABLE, при переходе в режим готовности (переключение STATUS.READY в 1). Бит может быть сброшен записью 1.
-	31-4	Зарезервировано

TEXT_OUT – массив регистров выходного текста

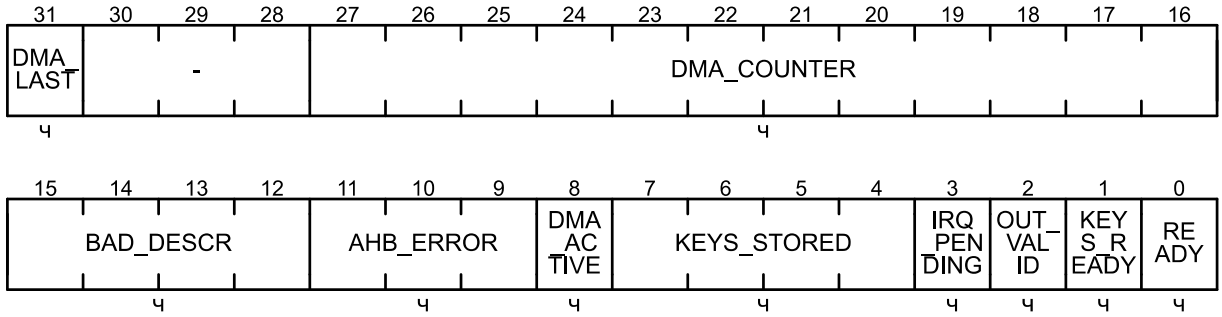
Смещение: TEXT_OUT +(4*n)h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	32-разрядный регистр, хранящий часть значения выходного текста.

STATUS – регистр статуса операций

Смещение: + 80h
Сброс: 0000_0001h

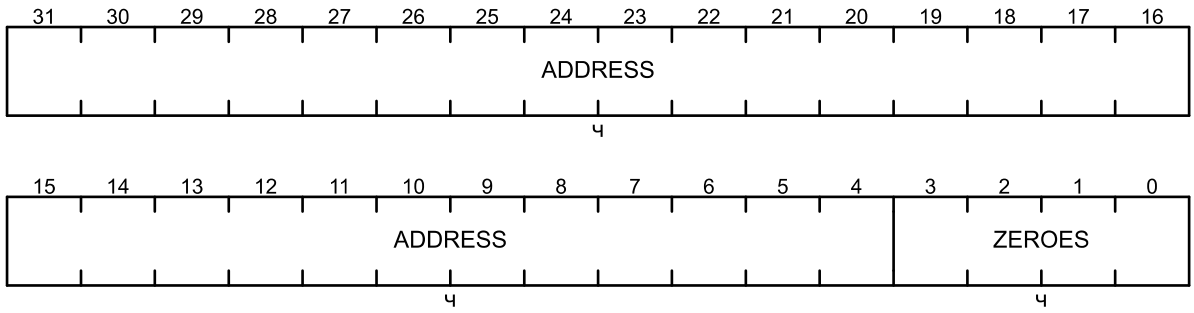


Поле	Биты	Описание
DMA_LAST	31	Поле отображает значение бита LAST слова управления DMA последнего прочитанного дескриптора. Бит установлен, если прочитанный дескриптор является последним в цепочке дескрипторов. Значение обновляется при считывании нового дескриптора. Бит сбрасывается при экстренном прекращении выполнения операции.
DMA_ COUNTER	27-16	Поле отображает количество обработанных DMA блоков текст. Значение обнуляется при считывании нового дескриптора. Значение сбрасывается при экстренном прекращении выполнения операции.

<table>
<tr><td>BAD_DESCR</td><td>15-12</td><td>Поле хранит сведения об обнаруженных некорректных словах в прочитанном дескрипторе DMA. После записи в регистр DMA_CONTROL любого значения поле будет переведено в состояние NO_ERROR.
Список возможных значений поля:<table><tr><td>0000b</td><td>NO_ERROR</td></tr><tr><td>0001b</td><td>CONTROL</td></tr><tr><td>0010b</td><td>TEXT_IN_ADDRESS</td></tr><tr><td>0100b</td><td>TEXT_OUT_ADDRESS</td></tr><tr><td>1000b</td><td>NEXT_DESCRIPTOR</td></tr></table></td></tr>
<tr><td>AHB_ERROR</td><td>11-9</td><td>Поле хранит сведения об области памяти, на которую пришелся ответ ERROR на транзакцию по шине AHB. После записи в регистр DMA_CONTROL любого значения поле будет переведено в состояние NO_ERROR.
Список возможных значений:<table><tr><td>000b</td><td>NO_ERROR</td></tr><tr><td>001b</td><td>READ_DESCRIPTOR</td></tr><tr><td>010b</td><td>READ_TEXT_IN</td></tr><tr><td>100b</td><td>WRITE_TEXT_OUT</td></tr></table></td></tr>
<tr><td>DMA_
ACTIVE</td><td>8</td><td>Если бит установлен, то конечный автомат DMA занят выполнением дескриптора. Бит READY будет находиться в сброшенном состоянии, пока DMA_ACTIVE установлен. Бит сбрасывается после выполнения цепочки дескрипторов, обнаружения некорректного дескриптора, получения ответа ERROR по шине AHB или при экстренном прекращении выполнения операции.</td></tr>
<tr><td>KEYS_
STORED</td><td>7-4</td><td>Каждый бит данного поля соответствует заданному криптографическому алгоритму. Биты нумеруются справа налево. 0 – AES-128, 1 – AES-256, 2 – MAGMA, 3 – KUZNECHIK.</td></tr>
<tr><td>IRQ_
PENDING</td><td>3</td><td>Если бит установлен, то регистр IRQ содержит хотя бы 1 не сброшенный бит.</td></tr>
<tr><td>TEXT_OUT_
VALID</td><td>2</td><td>Если бит установлен, то на выходе криптографического подмодуля находится результат шифрования/дешифрования входных данных (TEXT_IN_*, IV_* или TEXT_IN_* ^ IV_* в зависимости от алгоритма и режима выполнения криптографической операции).</td></tr>
<tr><td>KEYS_
READY</td><td>1</td><td>Если бит установлен, то раундовые ключи текущего активного криптографического алгоритма сгенерированы. Бит сбрасывается при старте обновления раундовых ключей или при экстренном прекращении выполнения операции.</td></tr>
<tr><td>READY</td><td>0</td><td>Если бит установлен, то модуль готов к выполнению очередной криптографической операции и/или обновлению раундовых ключей шифрования. Бит сбрасывается при старте обновления раундовых ключей, выполнения криптографической операции и считывания первого дескриптора цепочки. Бит устанавливается после завершения всех запланированных действий или при экстренном прекращении выполнения операции.</td></tr>
<tr><td>-</td><td>30-28</td><td>Зарезервировано</td></tr>
</table>

CURRENT_DESCRIPTOR – регистр адреса активного дескриптора

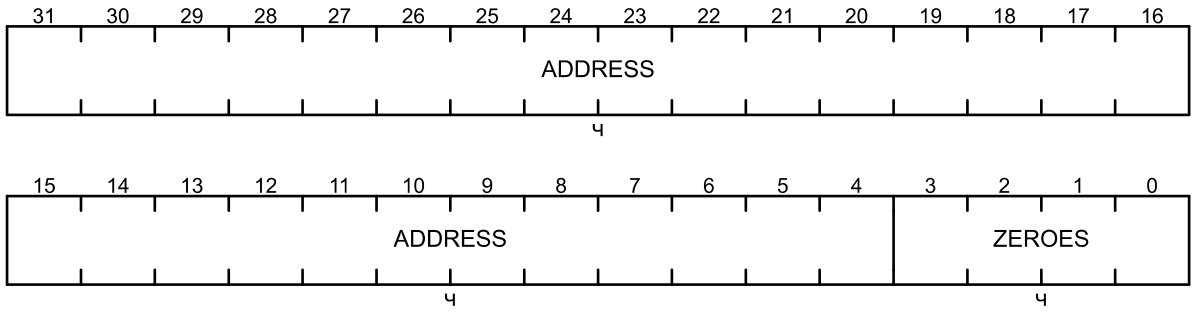
Смещение: + 84h
Сброс: 0000_0000h



Поле	Биты	Описание
ADDRESS	31-4	Адрес, который указывает на текущий выполняемый дескриптор. При записи 1 в бит START регистра DMA_CONTROL в данный регистр попадает значение из регистра BASE_DESCRIPTOR. Если значение бита LAST слова управления DMA текущего дескриптора сброшен, то при успешном завершении текущего дескриптора в CURRENT_DESCRIPTOR заносится значение NEXT_DESCRIPTOR
ZEROES	4-0	Младшие биты адреса, должны быть равны 0.

NEXT_DESCRIPTOR – регистр адреса следующего дескриптора

Смещение: + 88h
Сброс: 0000_0000h



Поле	Биты	Описание
ADDRESS	31-4	Адрес, который указывает на следующий дескриптор цепочки. Значение регистра будет сброшено, когда в слове управления текущего дескриптора DMA будет установлен бит LAST
ZEROES	4-0	Младшие биты адреса, должны быть равны 0.

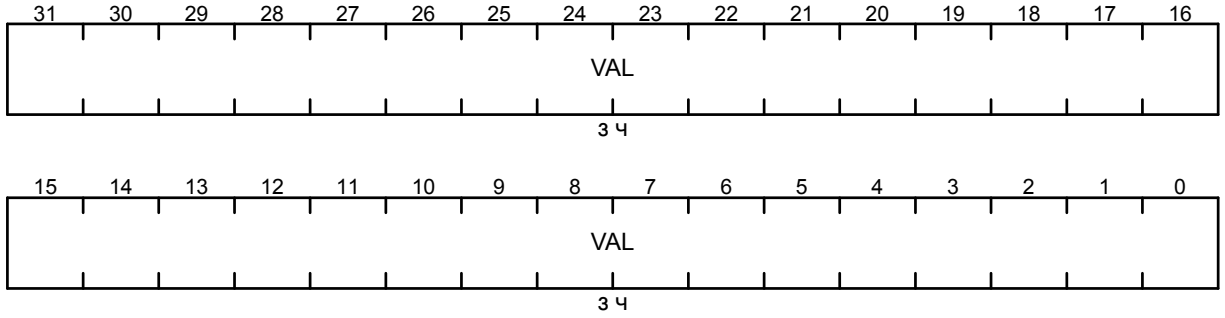
A.13 Регистры сторожевых таймеров WDT и IWDT

Базовый адрес:	3000_B000h	Регистры сторожевого таймера WDT
	3801_2000h	Регистры независимого таймера IWDT

Примечание – Регистры WDT и IWDT идентичны, за исключением регистра CTRL.

LOAD – регистр загрузки сторожевого таймера

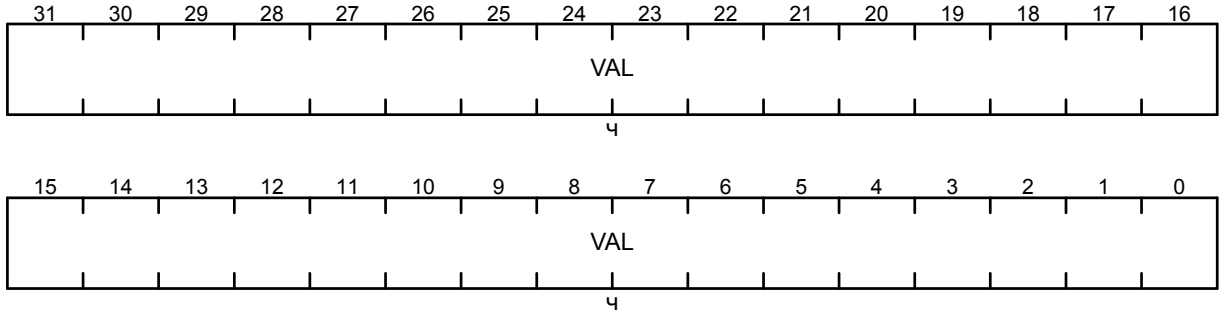
Смещение: + 00h
Сброс: FFFF_FFFFh



Поле	Биты	Описание
VAL	31-0	32-разрядный регистр, хранящий начальное значение счетчика. Когда происходит запись в этот регистр, счетчик сразу инициируется этим новым значением. Минимальное допустимое значение 0000_0001h

VALUE – регистр значения счетчика сторожевого таймера

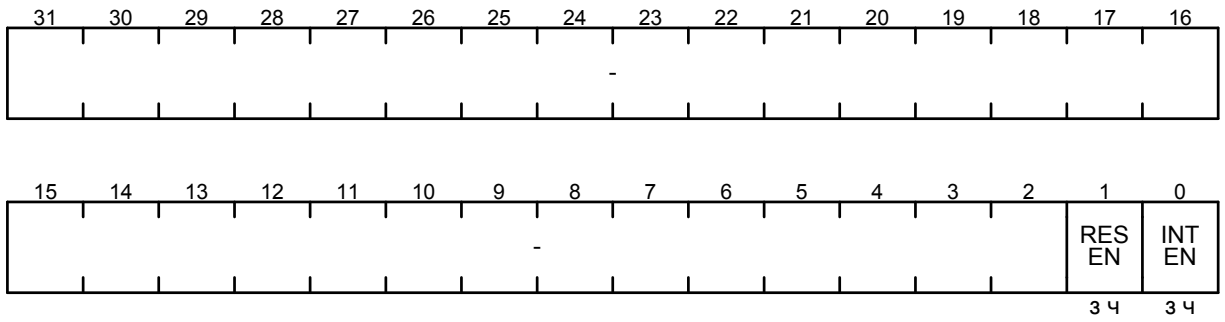
Смещение: + 04h
Сброс: FFFF_FFFFh



Поле	Биты	Описание
VAL	31-0	32-разрядный регистр текущего значения счетчика

CTRL – регистр управления сторожевого таймера

Смещение: + 08h
Сброс: 0h

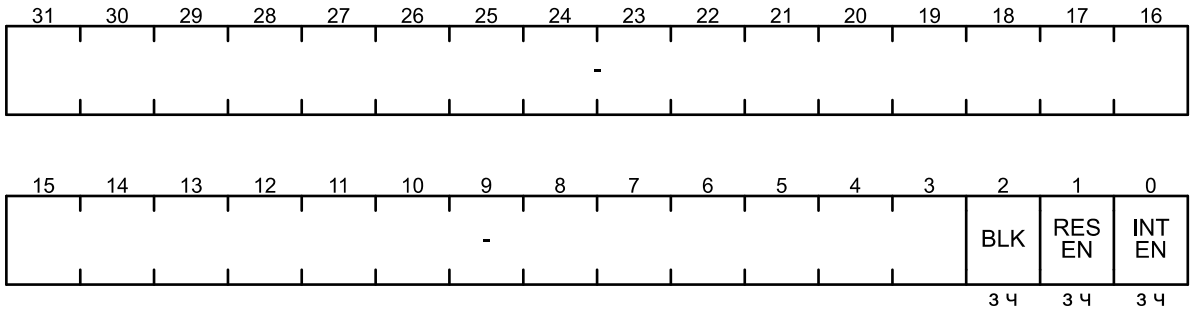


Поле	Биты	Описание	
RESEN	1	Бит разрешения сброса микроконтроллера по сторожевому таймеру. Работает по функции «Логическое И» с битом INTEN данного регистра	
		0	Сброс бита выключает сброс
		1	Установка включает сброс
INTEN	0	Бит включения счета и разрешения прерывания сторожевого таймера	
		0	Сброс бита выключает счетчик и снимает прерывание
		1	Установка бита включает счетчик и разрешает генерацию прерывания. Если счетчик был включен на момент установки бита, то он инициируется значением из регистра LOAD
–	31-2	Зарезервировано	

Примечание – Данный регистр относится только к таймеру WDT.

CTRL – регистр управления независимого сторожевого таймера

Смещение: + 08h
Сброс: 0h

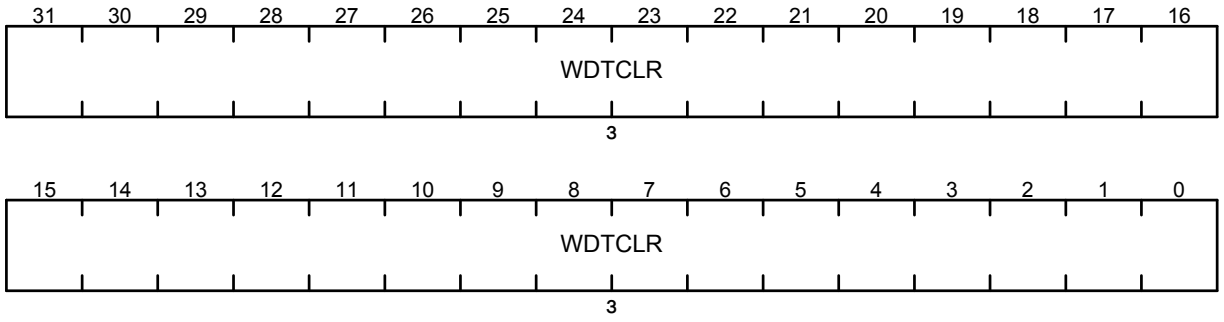


Поле	Биты	Описание	
BLK	2	Бит блокировки изменения источника тактирования IWDT и запрещения сброса IWDT при записи регистра PMU_RTC->IWDT_CFG	
RESEN	1	Бит разрешения сброса микроконтроллера по сторожевому таймеру. Работает по функции «Логическое И» с битом INTEN данного регистра	
		0	Сброс бита выключает сброс
		1	Установка включает сброс
INTEN	0	Бит включения счета и разрешения прерывания сторожевого таймера	
		0	Сброс бита выключает счетчик и снимает прерывание
		1	Установка бита включает счетчик и разрешает генерацию прерывания. Если счетчик был включен на момент установки бита, то он инициируется значением из регистра LOAD
–	31-3	Зарезервировано	

Примечание – Данный регистр относится только к таймеру IWDT.

INTCLR – регистр сброса сторожевого таймера

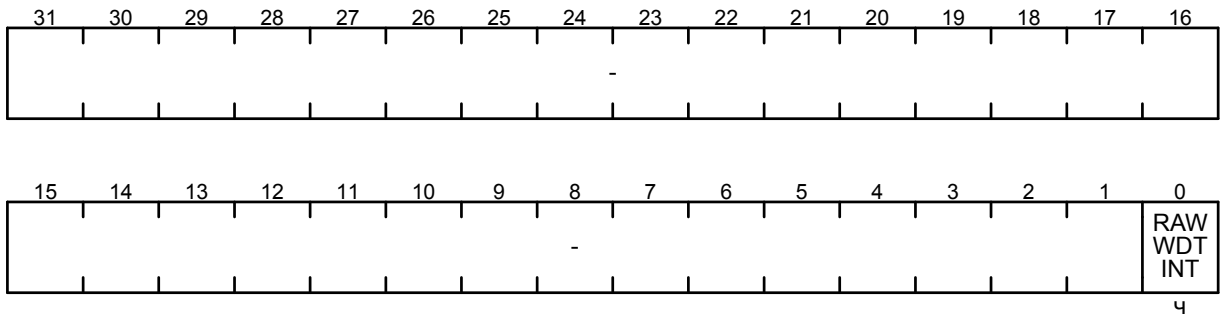
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание
WDTCLR	31-0	32-разрядный регистр сброса сторожевого таймера. Запись любого значения в этот регистр приводит к сбросу прерывания сторожевого таймера и загрузке счетчика значением из регистра LOAD

RIS – регистр прерывания сторожевого таймера

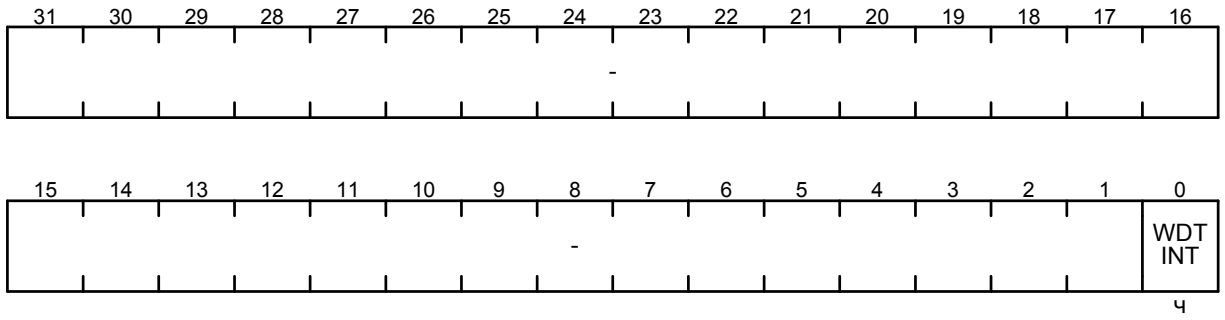
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание	
RAWWDTINT	0	Индикатор состояния немаскированного бита прерывания	
		0	Сброшено
		1	Установлено
–	31-1	Зарезервировано	

MIS – регистр маскированного прерывания сторожевого таймера

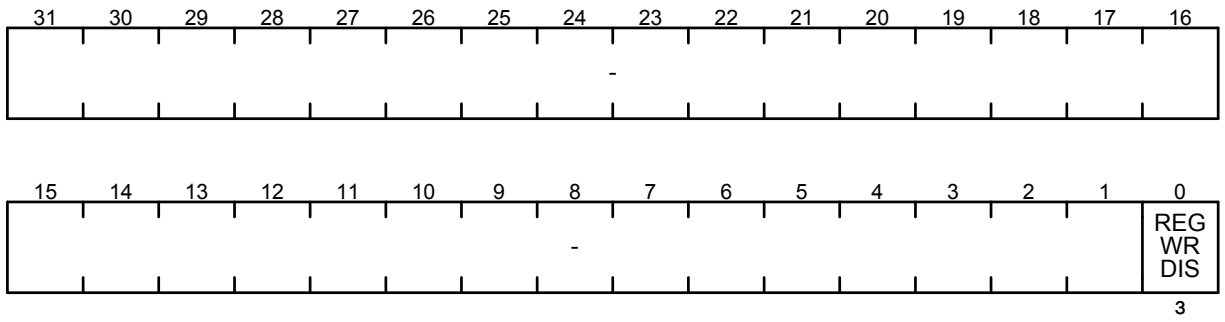
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание	
WDTINT	0	Индикатор состояния маскированного бита прерывания. Сигнализирует о появлении маскированного прерывания от счетчика. Состояние бита WDTINT – это «логическое И» битов RAWWDTINT и INTEN	
		0	Сброшено
		1	Установлено
–	31-1	Зарезервировано	

LOCK – регистр блокировки сторожевого таймера

Смещение: + C00h
Сброс: 0h



Поле	Биты	Описание	
REGWRDIS	0	Бит запрета записи во все регистры сторожевого таймера (кроме регистра LOCK). Функция необходима для предотвращения отключения сторожевого таймера сбойными программами	
		0	Разрешено (по умолчанию). Для сброса бита следует записать в регистр LOCK значение 1ACCE551h
		1	Запрещено. Для установки бита следует записать в регистр LOCK любое значение, кроме 1ACCE551h
–	31-1	Зарезервировано	

A.14 Регистры блока АЦП последовательного приближения

Базовый адрес: 3001_0000h

Смещение:

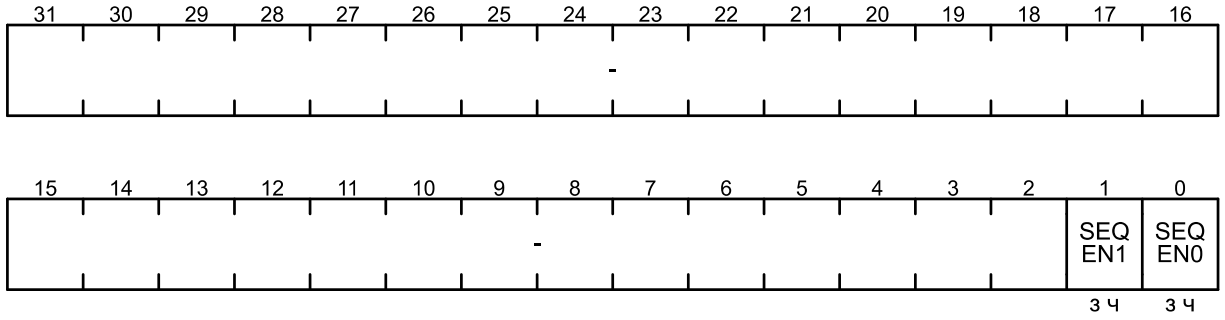
+ 040h (SEQ0)	Регистры секвенсора 0
+ 07Ch (SEQ1)	Регистры секвенсора 1
+ 400h (DC0)	Регистры цифрового компаратора 0
+ 40Ch (DC1)	Регистры цифрового компаратора 1
+ 418h (DC2)	Регистры цифрового компаратора 2
+ 424h (DC3)	Регистры цифрового компаратора 3
+ 430h (DC4)	Регистры цифрового компаратора 4
+ 43Ch (DC5)	Регистры цифрового компаратора 5
+ 448h (DC6)	Регистры цифрового компаратора 6
+ 454h (DC7)	Регистры цифрового компаратора 7
+ 580h (CHCTL)	Массив регистров настройки каналов
+ 680h (CHDELAY)	Массив регистров настройки количества дополнительных циклов для каждого канала

Мнемоника: SEQs;
DCd

Примечание – s – номер секвенсора 0 – 1;
d – номер по порядку цифрового компаратора от 0 до 7;
n – номер канала АЦП от 0 до 7 и 10.

SEQEN – регистр включения секвенсоров

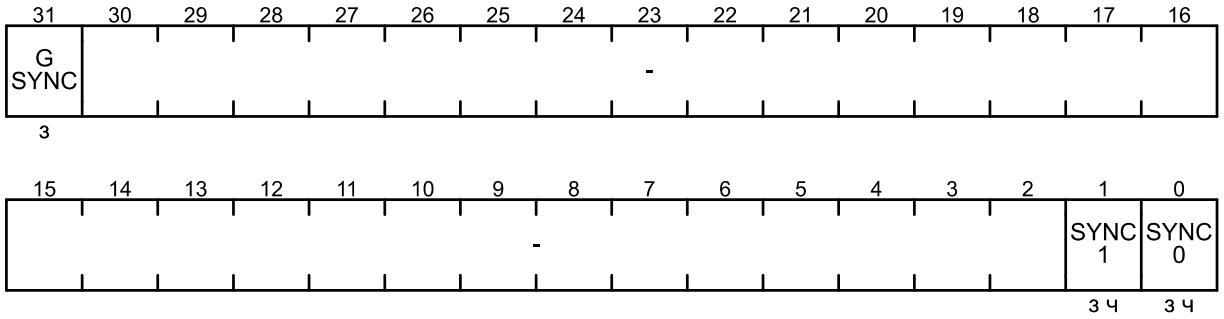
Смещение: + 00h
Сброс: 0000_0000h



Поле	Биты	Описание	
SEQENs	1-0	Бит разрешения работы секвенсора s	
		0	Запрещено
		1	Разрешено
–	31-2	Зарезервировано	

SEQSYNC – регистр программной синхронизации секвенсоров

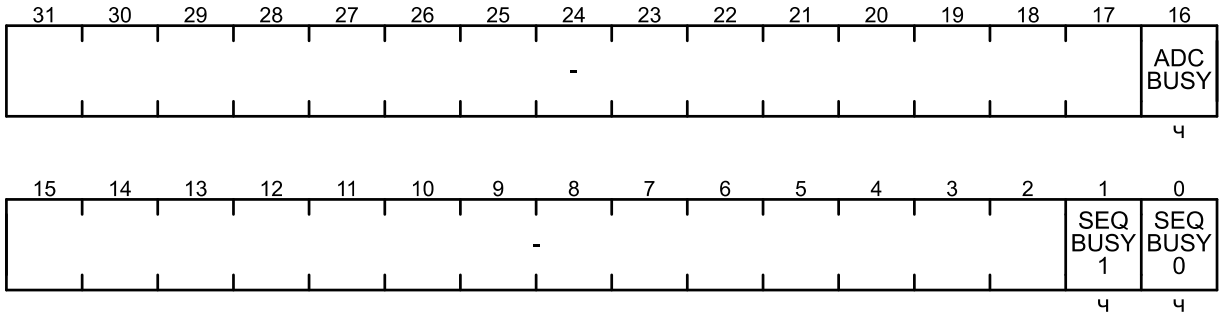
Смещение: 04h
Сброс: 0000_0000h



Поле	Биты	Описание	
GSYNC	31	Бит запуска секвенсоров. Запись единицы запускает секвенсоры, работа которых разрешена и для которых установлены биты SYNCs	
SYNCs	1-0	Бит разрешения запуска секвенсора s	
		0	Запрещено
		1	Разрешено (если установлен бит SEQENs в регистре SEQEN)
–	30-2	Зарезервировано	

BSTAT – регистр флагов занятости

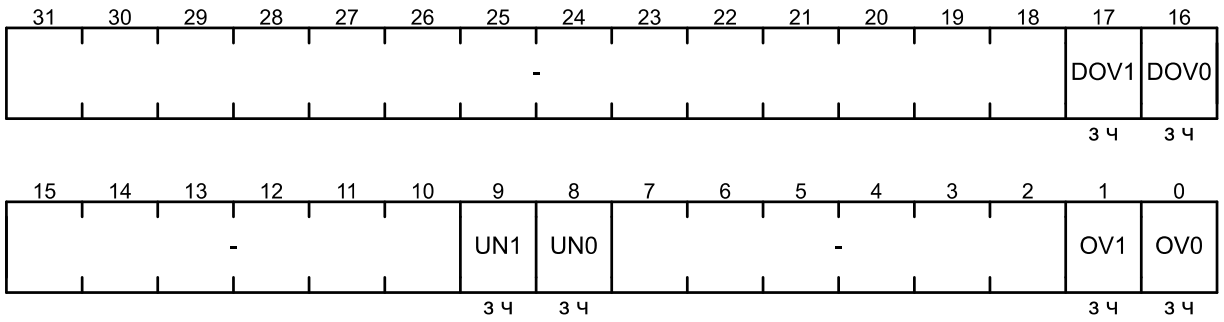
Смещение: 08h
Сброс: 0000_0000h



Поле	Биты	Описание	
ADCBUSY	16	Флаг занятости модуля АЦП	
		0	АЦП выключен или в режиме ожидания запроса
		1	АЦП проводит измерения по активным запросам
SEQBUSYs	1-0	Флаг занятости секвенсора s	
		0	Секвенсор выключен или в режиме ожидания сигнала запуска
		1	Секвенсор производит запуск/перезапуск или выполняет задержку перезапуска
–	31-17, 15-2	Зарезервировано	

FSTAT – регистр флагов буферов результатов и блока DMA

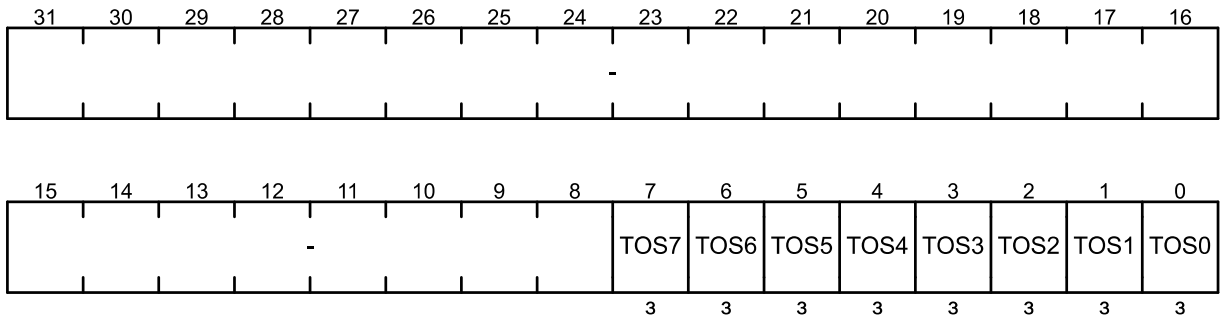
Смещение: 0Ch
Сброс: 0000_0000h



Поле	Биты	Описание	
DOVs	17-16	Флаг ошибки DMA	
		0	Нет ошибки
		1	При наличии обрабатываемого запроса DMA от секвенсора s, пришел еще один запрос, который не может быть обработан
		Флаг сбрасывается записью единицы	
UNs	9-8	Флаг пустоты буфера секвенсора s	
		0	Буфер не пуст
		1	Буфер пуст
		Флаг может быть сброшен программно записью единицы	
OVs	1-0	Флаг заполнения буфера секвенсора s	
		0	В буфере есть как минимум одна свободная ячейка
		1	Буфер заполнен. Все последующие записи в буфер блокируются до появления как минимум одной свободной ячейки
		Флаг сбрасывается записью единицы	
–	31-18, 15-10, 7-2	Зарезервировано	

DCTRIG – регистр сброса флагов выходных триггеров компараторов

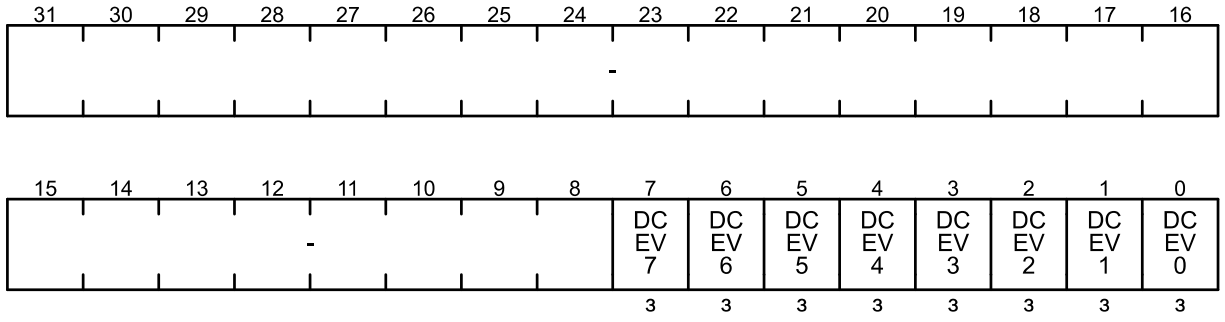
Смещение: 10h
Сброс: 0000_0000h



Поле	Биты	Описание	
TOSd	7-0	Флаг состояния выходного триггера компаратора d	
		0	Триггер не сработал
		1	Триггер сработал
		Флаг сбрасывается записью единицы	
–	31-8	Зарезервировано	

DCEV – регистр сброса флагов компараторов

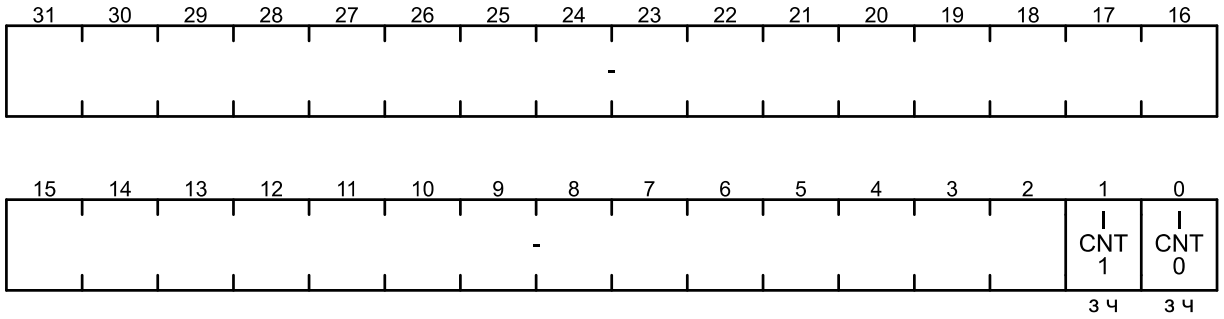
Смещение: 14h
Сброс: 0000_0000h



Поле	Биты	Описание	
DCEVd	7-0	Флаг события сравнения компаратора d	
		0	Сравнение не выполнялось
		1	Сравнение выполнялось
		Флаг сбрасывается записью единицы	
–	31-8	Зарезервировано	

CICNT – регистр настройки режима сброса счетчика прерываний

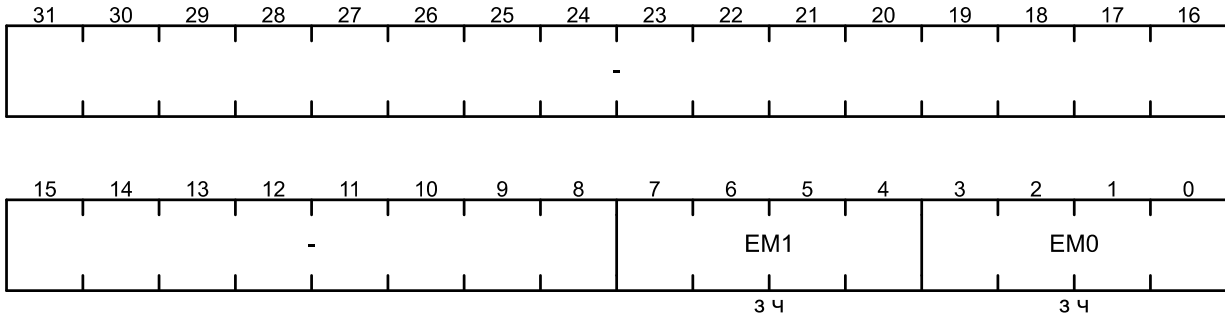
Смещение: 18h
Сброс: 0000_0000h



Поле	Биты	Описание	
ICNTs	1-0	Бит выбора режима сброса счетчика, используемого для генерации прерываний секвенсора s	
		0	Счетчик будет сбрасываться по запуску секвенсора
		1	Запрет сброса счетчика по запуску секвенсора
–	31-2	Зарезервировано	

EMUX – регистр выбора событий запуска секвенсоров

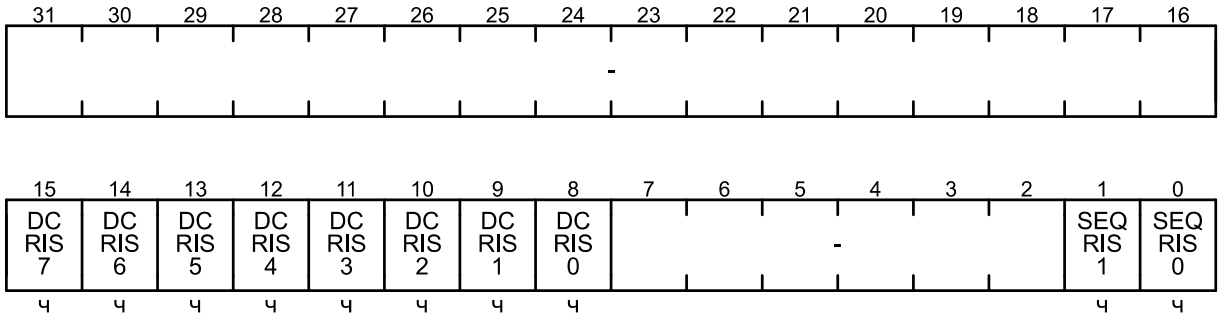
Смещение: 1Ch
Сброс: 0000_0000h



Поле	Биты	Описание	
EMs	3-0	Поле выбора события для запуска секвенсора s	
		0h	Установка бита GSYNC в регистре SEQSYNC
		1h	Сигнал от блока TMR32
		2h	Сигнал от блока TMR0
		3h	Сигнал от блока TMR1
		4h	Сигнал от блока TMR2
		5h	Сигнал от GPIOA
		6h	Сигнал от GPIOB
		7h	Сигнал от GPIOC
		Fh	Циклическая работа. Активируется после установки бита GSYNC в регистре SEQSYNC
		8h-Eh	Зарезервировано
–	31–8	Зарезервировано	

RIS – регистр флагов немаскированных прерываний

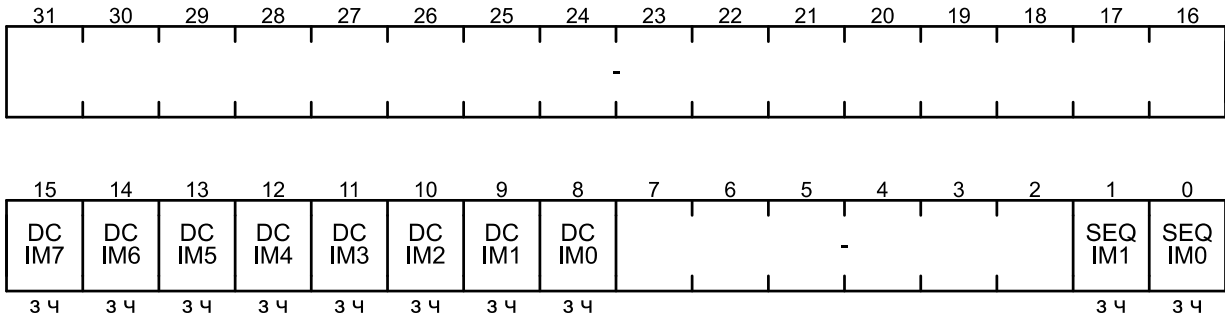
Смещение: 20h
Сброс: 0000_0000h



Поле	Биты	Описание	
DCRISd	31-8	Флаг прерывания компаратора d	
		0	Нет прерывания или флаг сброшен
		1	Поступил запрос на прерывание
SEQRISs	7-0	Флаг прерывания секвенсора s	
		0	Нет действий
		1	Запрос секвенсора завершился и счетчик прерываний досчитал до значения ICNT регистра SCCTL

IM – регистр маски прерываний

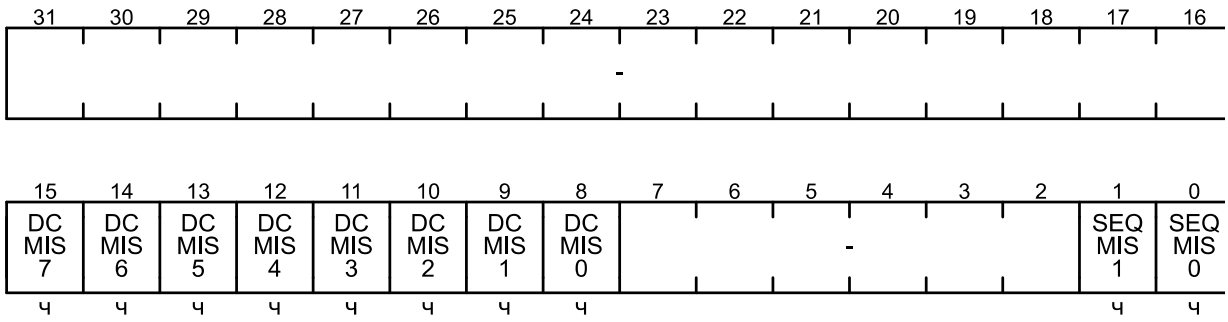
Смещение: 24h
Сброс: 0000_0000h



Поле	Биты	Описание	
DCIMd	31-8	Маска прерывания компаратора d	
		0	Маскировано
		1	Разрешено
SEQIMs	7-0	Маска прерывания секвенсора s	
		0	Маскировано
		1	Разрешено

MIS – регистр флагов маскированных прерываний

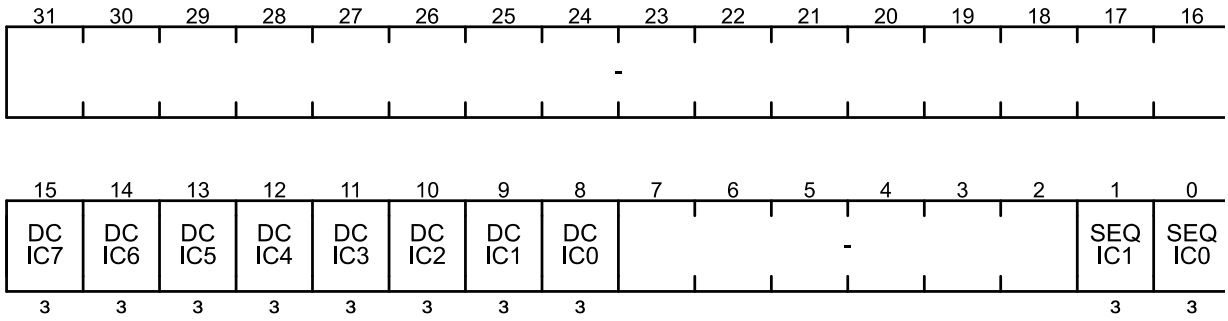
Смещение: 28h
Сброс: 0000_0000h



Поле	Биты	Описание	
DCMISd	31-8	Флаг маскированного прерывания компаратора d	
		0	Нет прерывания или флаг сброшен
		1	Поступил запрос на прерывание
SEQMISs	7-0	Флаг маскированного прерывания секвенсора s	
		0	Нет действий
		1	Запрос секвенсора завершился и счетчик прерываний досчитал до значения ICNT регистра SCCTL

IC – регистр сброса флагов прерываний

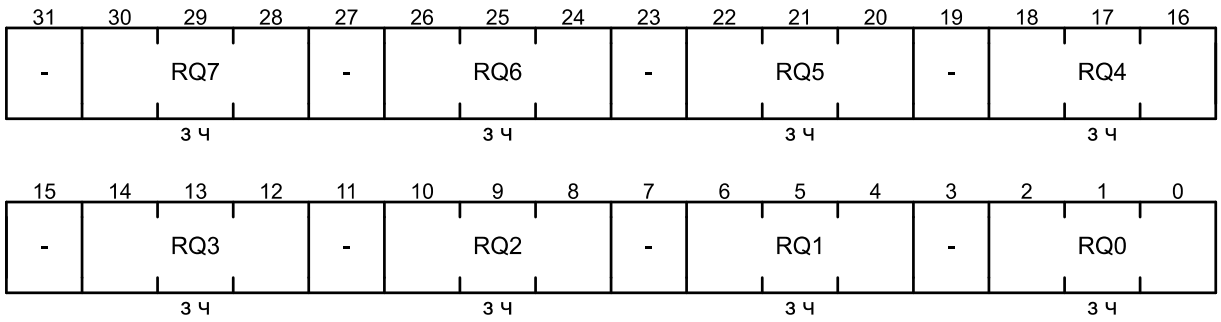
Смещение: 2Ch
Сброс: 0000_0000h



Поле	Биты	Описание	
DCICd	31-8	Сброс маскированного и немаскированного флага прерывания компаратора d	
		0	Нет действий
		1	Сброс флагов
SEQICs	7-0	Сброс маскированного и немаскированного флага прерывания секвенсора s	
		0	Нет действий
		1	Сброс флагов

SRQSEL – регистр выбора каналов для запросов секвенсора

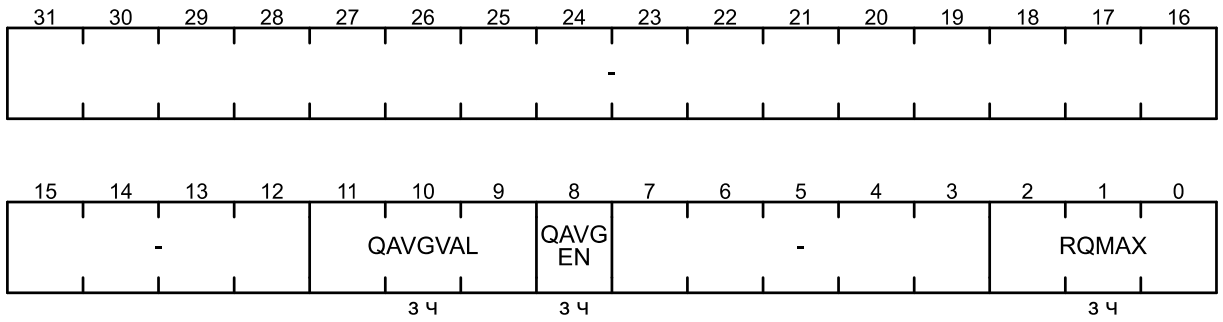
Смещение: SEQs + 00h
Сброс: 0000_0000h



Поле	Биты	Описание
RQn	30-28, 26-24, 22-20, 18-16, 14-12, 10-8, 6-4,2-0	Номер канала АЦП для запроса (n от 0 до 7) секвенсора s. Допустимые значения 0 – 7
–	31, 27, 23, 19, 15, 11, 7, 3	Зарезервировано

SRQCTL – регистр управления очередью запросов секвенсора

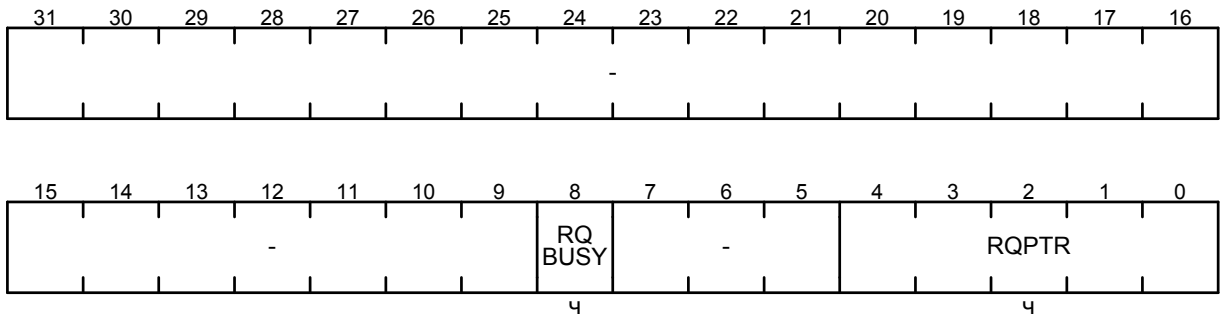
Смещение: SEQs + 18h
Сброс: 0000_0000h



Поле	Биты	Описание	
QAVGVAL	11-9	Поле задания количества опросов для усреднения сканированием	
		000	Без усреднения
		001	2
		010	4
		011	8
		100	16
		101	32
		110	64
		111	Зарезервировано
QAVGEN	8	Бит управления режимом усреднения сканированием	
		0	Усреднение сканированием очереди запросов отключено
		1	Усреднение сканированием очереди запросов включено
RQMAX	2-0	Поле задания «глубины» очереди запросов секвенсора s (номер последнего элемента)	
–	31-12, 7-3	Зарезервировано	

SRQSTAT – регистр статуса очереди запросов секвенсора

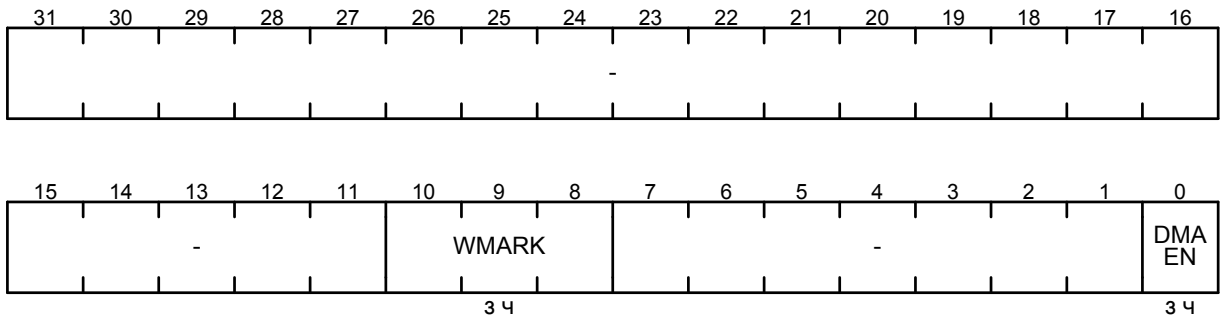
Смещение: SEQs + 1Ch
Сброс: 00000000h



Поле	Биты	Описание	
RQBUSY	8	Флаг активного запроса секвенсора	
		0	Текущий запрос неактивен
		1	Запрос на измерение выставлен и находится в состоянии обработки или ожидания
RQPTR	4-0	Номер текущего запроса в очереди	
–	31-9, 7-5	Зарезервировано	

SDMACTL – регистр управления запросами DMA секвенсора

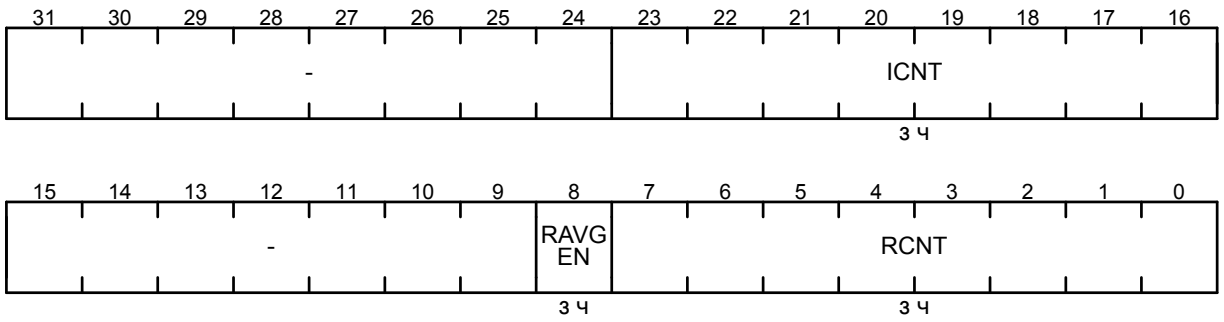
Смещение: SEQs + 20h
Сброс: 0000_0000h



Поле	Биты	Описание	
WMARK	10-8	Поле задания количества результатов измерений, записанных в буфер секвенсора, по достижении которого вызывается DMA	
		000	Зарезервировано
		001	Одна запись в буфер
		010	2
		011	4
		100	8
		101	16
		110	32
		111	Зарезервировано
DMAEN	0	Бит разрешения использования блока DMA	
		0	Запрещено
		1	Разрешено
–	31-11, 7-1	Зарезервировано	

SCCTL – регистр управления счетчиками прерывания и перезапуска секвенсора

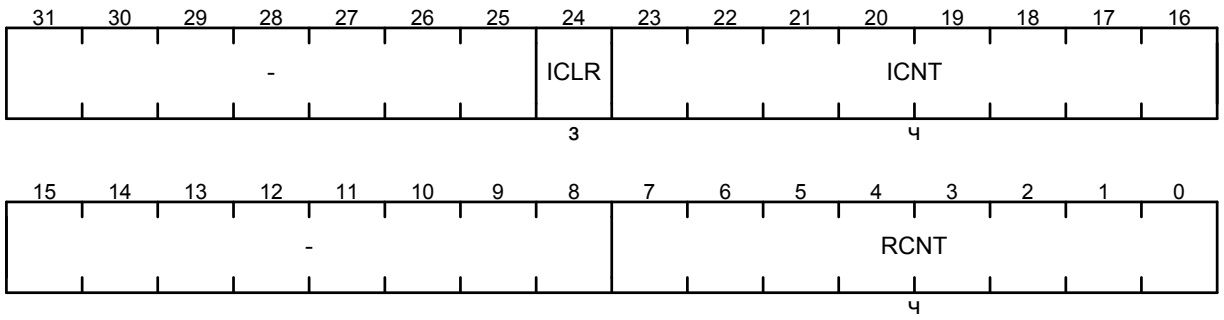
Смещение: SEQs + 24h
Сброс: 0000_0000h



Поле	Биты	Описание	
ICNT	23-16	Цикличность прерываний. Поле задания количества запросов секвенсором s модуля АЦП, по достижении которого генерируется прерывание. Значение 00h означает выставление прерывания по каждому запросу модуля АЦП, значение FFh – каждые 256 запросов	
RAVGEN	8	Бит разрешения режима усреднения результатов по перезапускам	
		0	Запрещено
		1	Разрешено
		Примечание – Для корректной работы этого режима поле RCNT регистра SCCTL должно содержать любое значение, соответствующее $2^p - 1$, где p = 1, ..., 8.	
RCNT	7-0	Количество перезапусков очереди запросов секвенсора s. Поле задания количества перезапусков очереди запросов секвенсора s после его запуска по событию. Значение 00h соответствует режиму без перезапусков, значение 01h – один перезапуск, FFh – 255 перезапусков	
–	31-24, 15-9	Зарезервировано	

SCVAL – регистр состояния счетчиков прерывания и перезапуска секвенсора

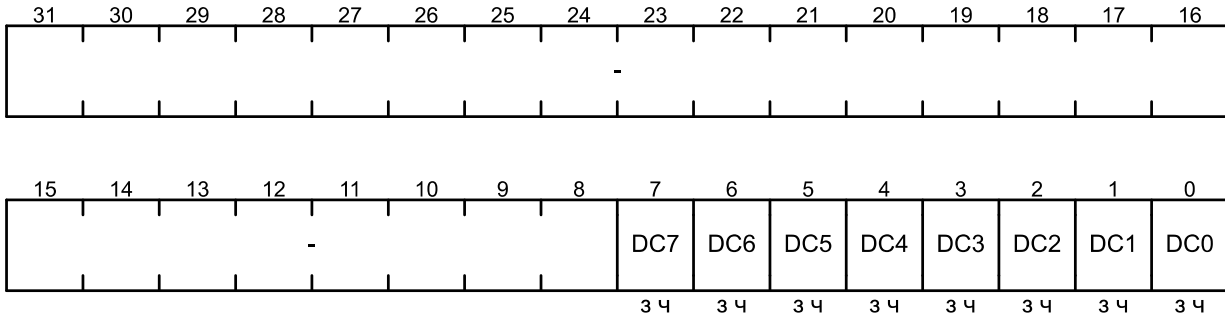
Смещение: SEQs + 28h
Сброс: 0000_0000h



Поле	Биты	Описание	
ICLR	24	Бит сброса счетчика запросов для генерации прерывания	
		0	Нет действий
		1	Сброс счетчика ICNT
ICNT	23-16	Текущее состояние счетчика запросов, используемого для генерации прерываний	
RCNT	7-0	Текущее количество совершенных перезапусков	
–	31-25, 15-8	Зарезервировано	

SDC – регистр выбора компаратора секвенсором

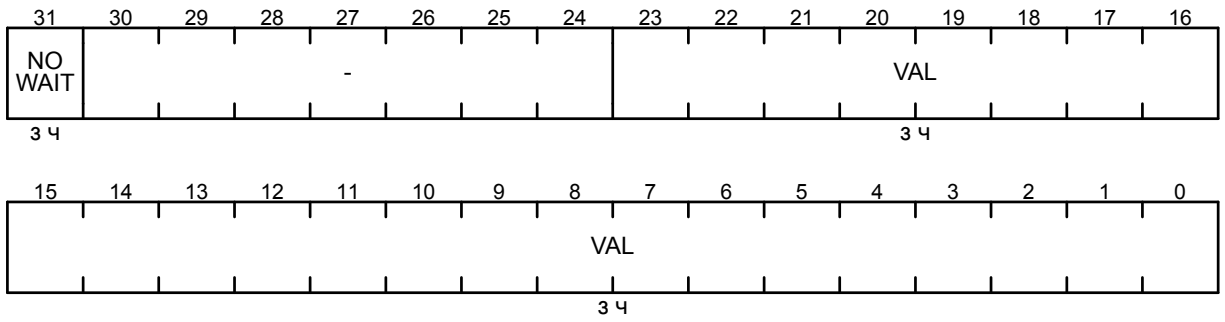
Смещение: SEQs + 2Ch
Сброс: 0000_0000h



Поле	Биты	Описание	
DCd	23-0	Бит разрешения работы компаратора d секвенсором	
		0	Запрещено
		1	Разрешено
–	31-24	Зарезервировано	

SRTMR – регистр задержки перезапусков секвенсора

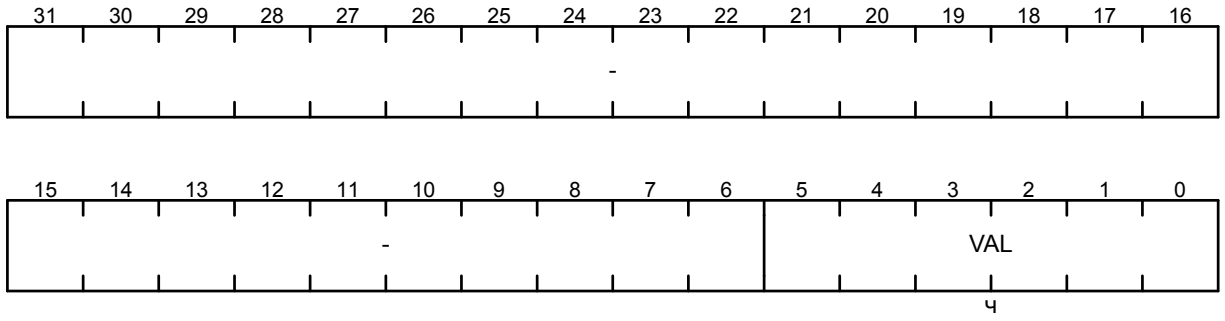
Смещение: SEQs + 30h
Сброс: 0000_0000h



Поле	Биты	Описание	
NOWAIT	31	Бит управления теневой загрузкой значения задержки перезапуска	
		0	Значение обновится по ближайшему событию запуска секвенсора
		1	Значение обновится по ближайшему событию перезапуска
VAL	23-0	Поле задания задержки перезапуска очереди секвенсора. Значение TMR = 000000h задает немедленный перезапуск (если он включен)	
–	30-24	Зарезервировано	

SFLOAD – регистр загрузки буфера секвенсора

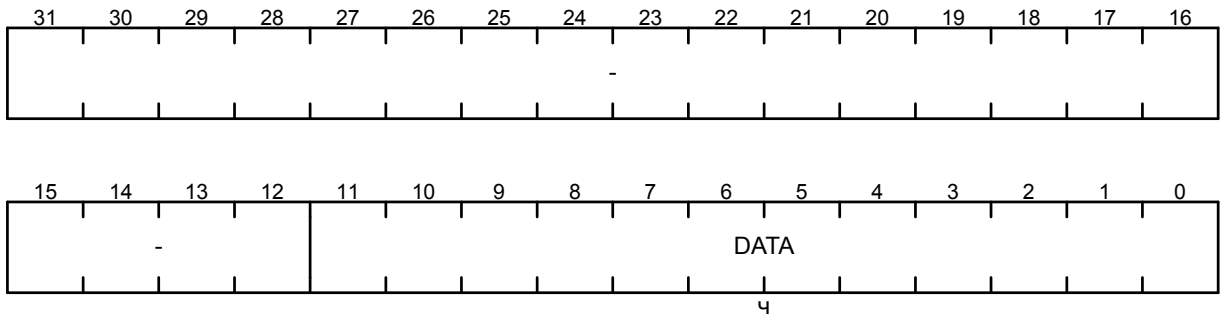
Смещение: SEQs + 34h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	5-0	Значение количества результатов измерений, сохраненных в буфере секвенсора
–	31-6	Зарезервировано

SFIFO – регистр результата измерения секвенсора

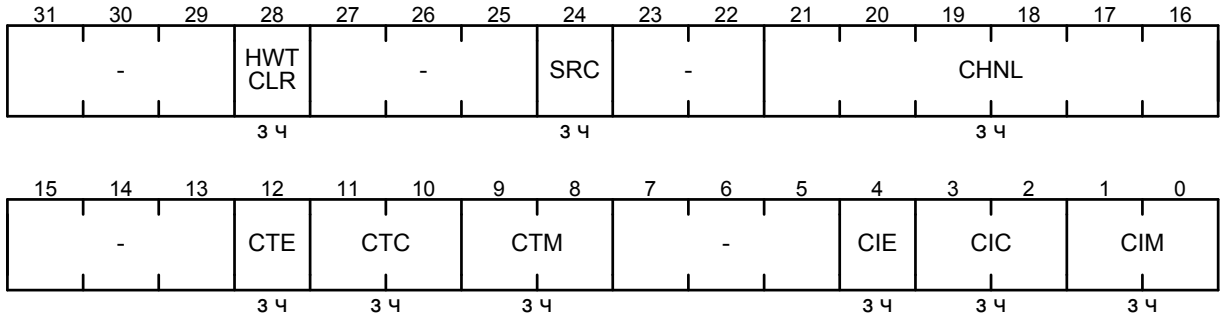
Смещение: SEQs + 38h
Сброс: 0000_0000h



Поле	Биты	Описание
DATA	11-0	Результат измерения. Чтение поля DATA возвращает результат измерения из буфера секвенсора
–	31-12	Зарезервировано

DCTL – регистр управления компаратора

Смещение: DCd + 00h
Сброс: 0000_0000h

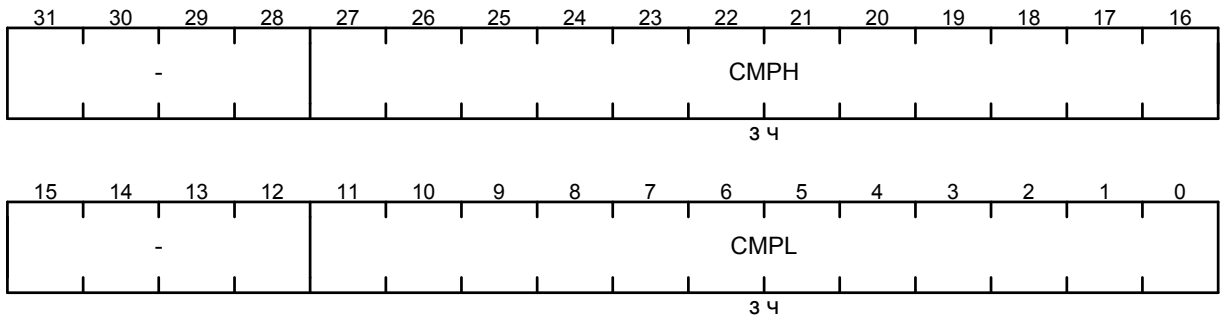


Поле	Биты	Описание	
HWTCLR	28	Управление аппаратным сбросом выходного триггера компаратора	
		0	Сброс происходит через флаги DCTRIG
		1	Сброс происходит аппаратно при первом отрицательном сравнении
SRC	24	Выбор источника значения для сравнения	
		0	Окончание измерения АЦП
		1	Запись результата в FIFO секвенсором
CHNL	21-16	Номер канала. Поле выбирает канал, результат измерения которого будет передан на компаратор	
CTE	12	Бит разрешения срабатывания выходного триггера компаратора	
		0	Запрещено
		1	Разрешено

Поле	Биты	Описание	
CTC	11-10	Поле задания условия срабатывания выходного триггера. Если для значения, полученного в результате измерения, выполняется условие, то состояние триггера единица, в противном случае – ноль	
		00	Измерение ≤ CMPL
		01	CMPL ≤ Измерение ≤ CMPH
		10	CMPH ≤ Измерение
		11	Зарезервировано
		Параметры CMPL и CMPH задаются в регистре DCMP	
CTM	9-8	Поле задания режима срабатывания выходного триггера	
		00	Многократный
		01	Однократный
		10	Многократный с гистерезисом
		11	Однократный с гистерезисом
CIE	4	Бит разрешения прерывания компаратора	
		0	Запрещено
		1	Разрешено. Прерывание генерируется каждый раз при одновременном выполнении условий CIC и CIM
CIC	3-2	Поле задания условия генерирования прерывания	
		00	Измерение ≤ CMPL
		01	CMPL ≤ Измерение ≤ CMPH
		10	CMPH ≤ Измерение
		11	Зарезервировано
CIM	1-0	Поле задания режима генерирования прерывания	
		00	Многократный
		01	Однократный
		10	Многократный с гистерезисом
		11	Однократный с гистерезисом
–	31-29, 27-25, 23-22, 15-13, 7-5	Зарезервировано	

DCMP – регистр диапазона компаратора

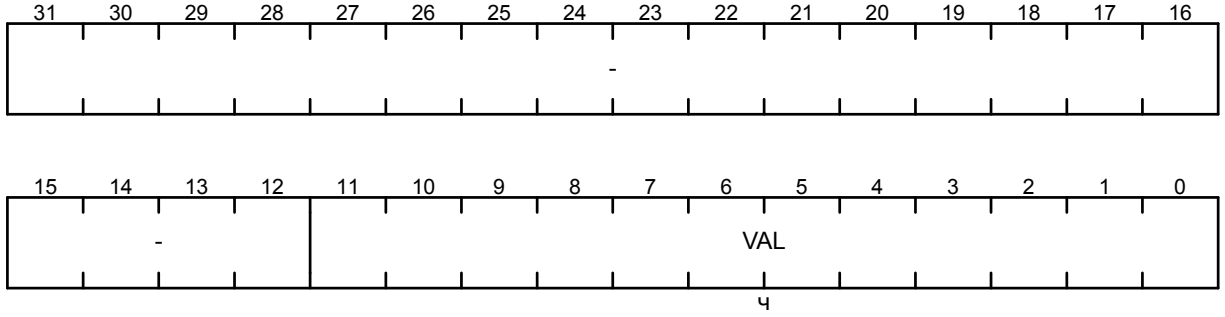
Смещение: DCd + 04h
Сброс: 0000_0000h



Поле	Биты	Описание
CMPH	27-16	Поле значения верхнего порога диапазона измерений
		Всегда должно выполняться условие CMPL ≤ CMPH
CMPL	11-0	Поле значения нижнего порога диапазона измерений
–	31-28, 15-12	Зарезервировано

DDATA – регистр результата измерения компаратора

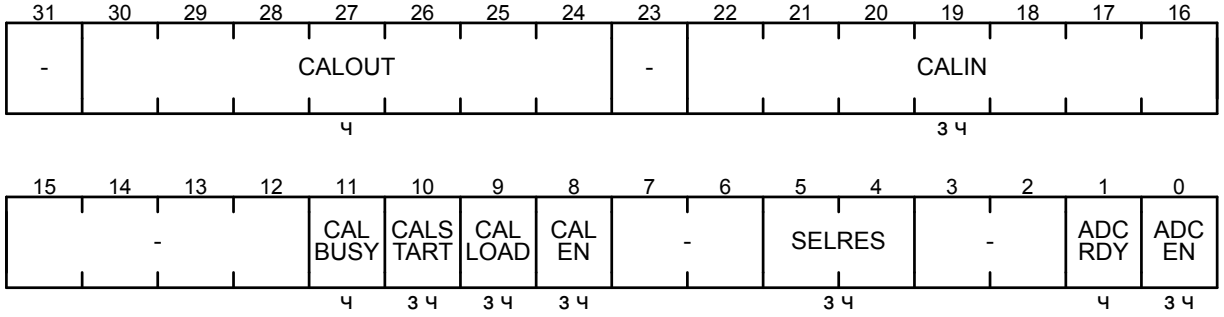
Смещение: DCd + 08h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	11-0	Значение результата измерения, которое последним использовалось компаратором при проверке на соответствие условиям CTC и CTM (см. регистр DCTL)
–	31-12	Зарезервировано

ACTL – регистр управления модулем АЦП

Смещение: + 540h
Сброс: 0000_0000h

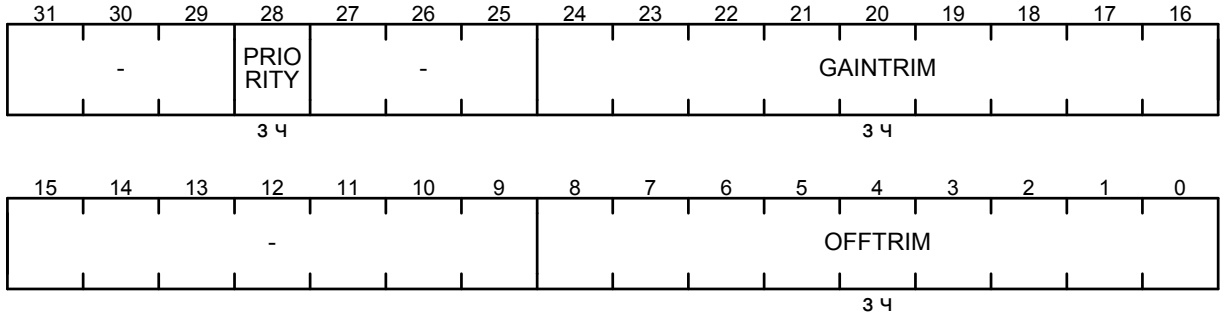


Поле	Биты	Описание
CALOUT	30-24	Поправочное значение, полученное в результате калибровки. Диапазон значений (–64, …, 63), величина в дополнительном коде
CALIN	22-16	Поправочное значение для ручного занесения в схему калибровки с помощью бита CALLOAD. Диапазон значений (–64, …, 63), величина вносится в дополнительном коде
CALBUSY	11	Флаг активности процедуры калибровки. Сбрасывается автоматически
	0	Калибровка не проводится
	1	Проводится калибровка
CALSTART	10	Старт проведения схемы внутренней калибровки АЦП
	0	Нет реакции
	1	Старт калибровки
CALLOAD	9	Внесение значения поля CALIN во внутреннюю схему калибровки
	0	Нет реакции
	1	Загрузка значения
CALEN	8	Включение схемы внутренней калибровки
	0	Выключено
	1	Включено
SELRES	5-4	Выбор разрядности результата модуля АЦП
	00b	6 бит
	01b	8 бит
	10b	10 бит
	11b	12 бит
ADCRDY	1	Флаг готовности АЦП к проведению измерений
	0	АЦП выключено (ADCEN=0) либо в состоянии инициализации
	1	АЦП включено и готово к преобразованиям
ADCEN	0	Включение АЦП. При каждом включении запускается процедура инициализации, которая завершается установкой ADCRDY
	0	Модуль АЦП выключен
	1	Модуль АЦП включен

–	31, 23, 15-12, 7-6, 3-2	Зарезервировано

CHCTL – массив регистров настройки каналов

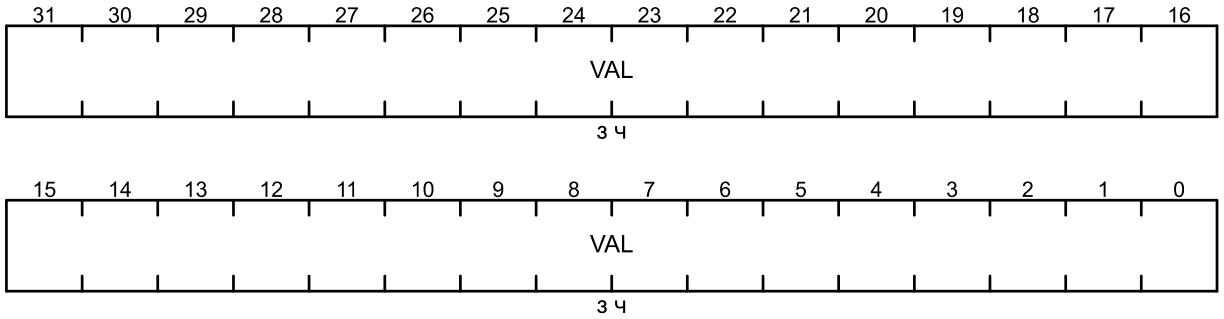
Смещение: CHCTL + (4*n)h
Сброс: 0000_0000h



Поле	Биты	Описание	
PRIORITY	28	Бит задания приоритета канала	
		0	Канал имеет стандартный приоритет
		1	Канал имеет повышенный приоритет
GAINTRIM	24-16	Поле задания коэффициента для коррекции усиления. Диапазон значений -256, …, 255 в 12-разрядном режиме, величина вносится в дополнительном коде: 100h соответствует -256, 000h – 0, 0FFh – 255. При других разрядностях результата диапазон сужается – подробнее в таблице 21.2. Примечание – Отрицательные числа записываются в дополнительном двоичном коде	
OFFTRIM	8-0	Поле задания коэффициента для коррекции смещения нуля. Диапазон значений –256, …, 255 в 12-разрядном режиме, величина вносится в дополнительном коде: 100h соответствует -256, 000h – 0, 0FFh – 255. При других разрядностях результата диапазон сужается – подробнее в таблице 21.2. Примечание – Отрицательные числа записываются в дополнительном двоичном коде	
–	31-29, 27-25, 15-9	Зарезервировано	

CHDELAY – массив регистров настройки количества дополнительных циклов для каждого канала

Смещение: CHDELAY + (4*n)h
Сброс: 0000_0000h



Поле	Биты	Описание
VAL	31-0	Поле задания количества дополнительных циклов (время, на которое канал измерения подключается к внутренней зарядной емкости АЦП) для канала n.

A.15 Регистры сигма-дельта АЦП

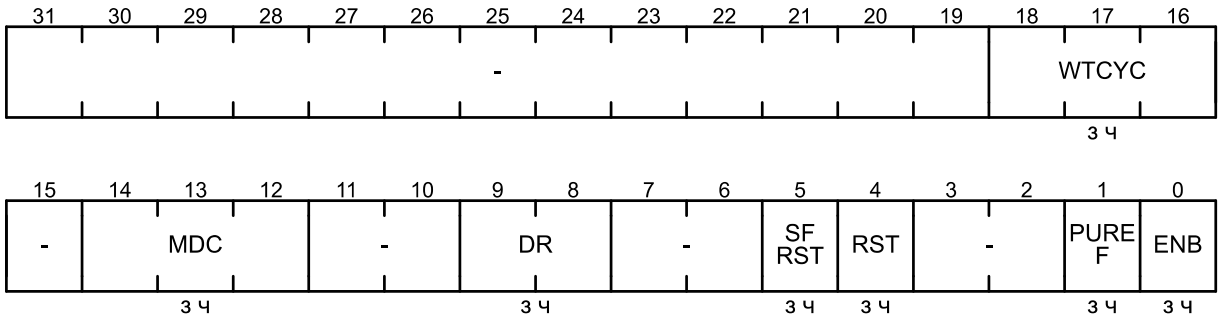
Базовый адрес: 3001_2000h

Смещение: + 040h (DATA) Массив регистров настройки каналов

Примечание – n – номер канала АЦП от 0 до 7

CTRL – регистр конфигурации АЦП

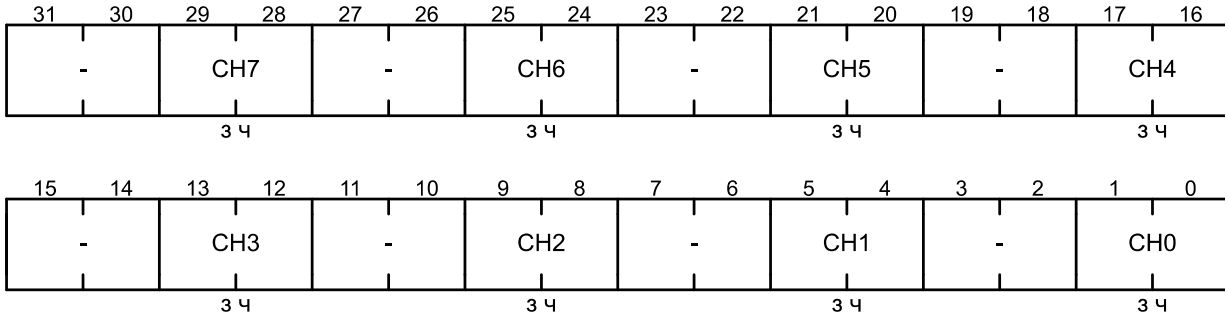
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание	
WTCYC	18-16	Количество тактов ожидания	
MDC	14-12	Значение основного делителя частоты	
		0	Делитель выключен
		1	1/2
		2	1/3
		3	1/4
		4	1/5
		5 6 7	Делитель выключен
DR	9-8	Значение делителя коэффициента прореживания	
		0	1/2048
		1	1/1024
		2	1/512
		3	1/256
SFRST	5	Бит программного сброса АЦП	
RST	4	Бит сброса АЦП	
PUREF	1	Бит включения внутреннего источника опорного напряжения	
ENB	0	Бит разрешения работы АЦП	
–	31-19, 15, 11-10, 7-6, 3-2	Зарезервировано	

MODE – регистр выбора режима каналов

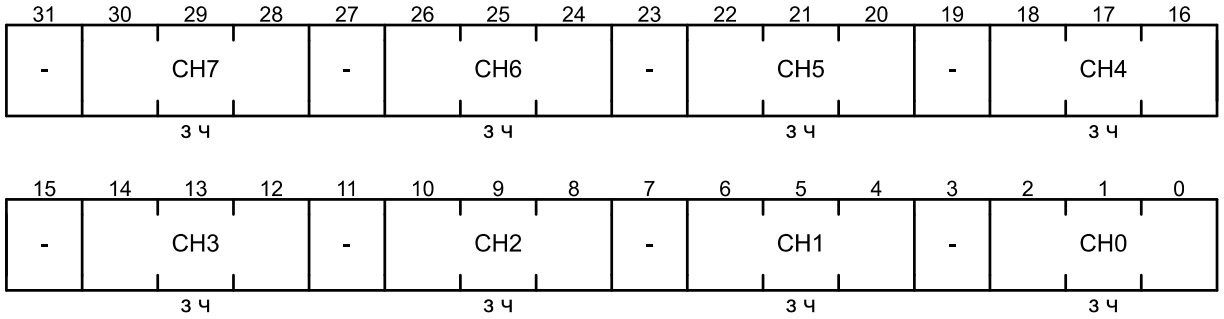
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание	
CHn	29-28, 25-24, 21-20, 17-16, 13-12, 9-8, 5-4, 1-0	Выбор режима работы канала n	
		0	Нет запуска
		1	Однократный запуск с последующим отключением канала
		2	Однократный запуск без отключения канала
		3	Циклический запуск
–	31-19, 15, 11-10, 7-6, 3-2	Зарезервировано	

AMPL – регистр усиления каналов

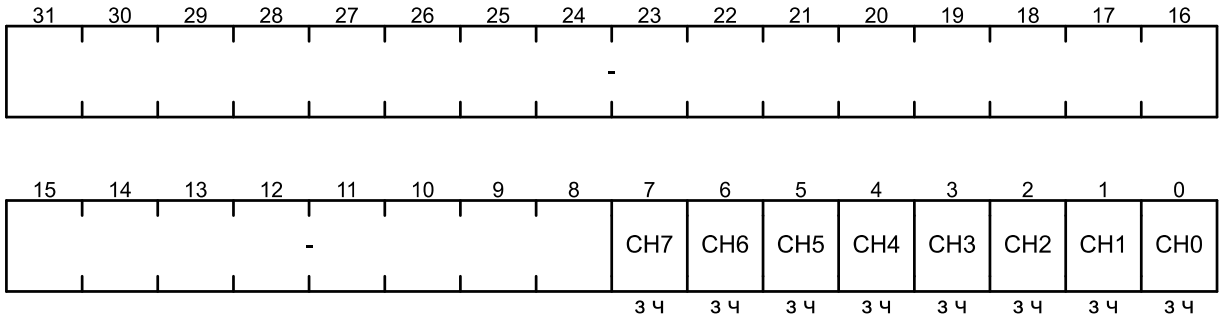
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание	
CHn	18-16	Выбор коэффициента усиления канала n	
		0	0 дБ
		1	6 дБ
		2	12 дБ
		3	18 дБ
		4	20 дБ
		5	26 дБ
		6	32 дБ
		7	38 дБ
-	31, 27, 23, 19, 15, 11, 7, 3	Зарезервировано	

ENB – регистр запуска преобразования

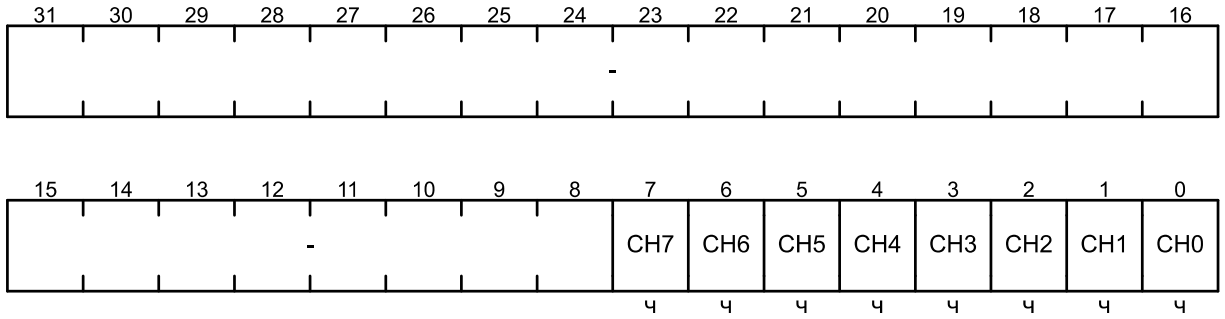
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание
CHn	7-0	Бит включения канала n
-	31-8	Зарезервировано

READY – регистр готовности каналов

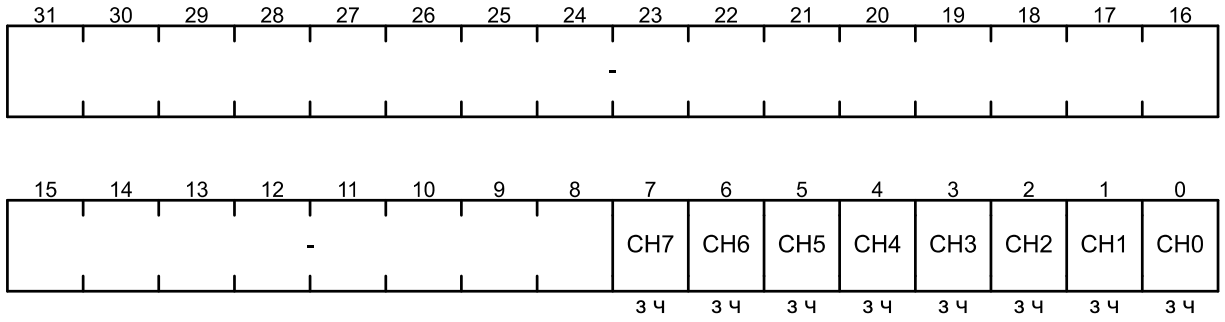
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
CHn	7, 6, 5, 4, 3, 2, 1, 0	Бит готовности канала n
-	31-8	Зарезервировано

DATAUPD – регистр окончания преобразования

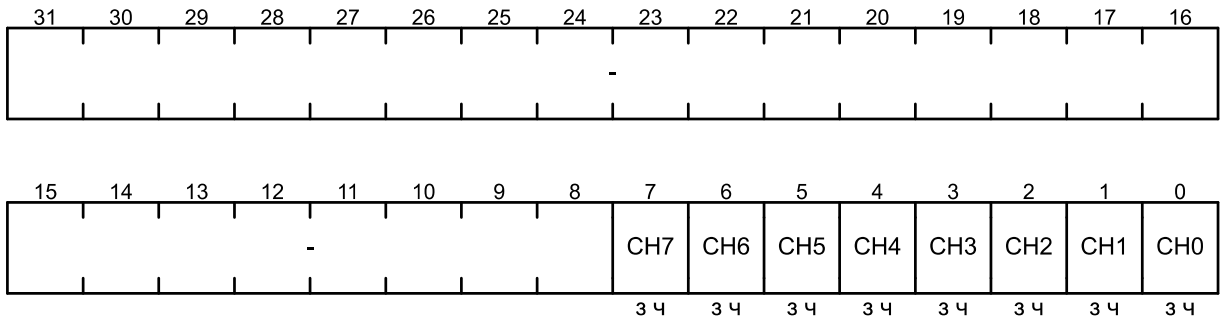
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание
CHn	7, 6, 5, 4, 3, 2, 1, 0	Бит окончания преобразования канала n
-	31-8	Зарезервировано

IM – регистр маски прерываний

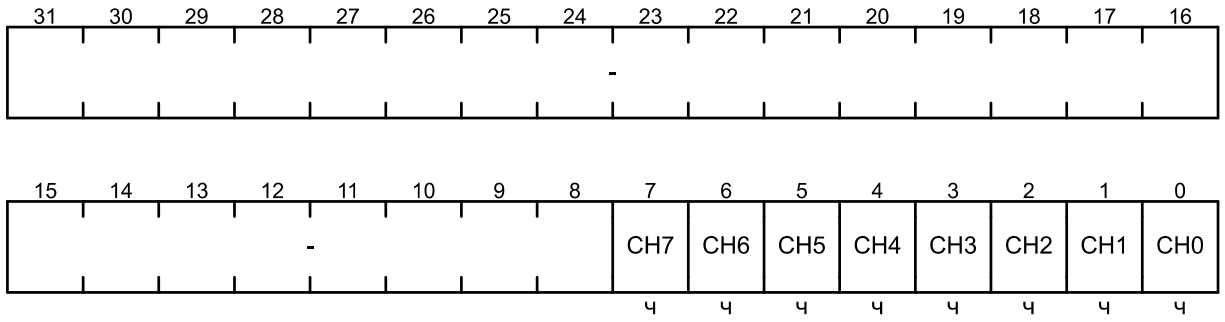
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
CHn	7, 6, 5, 4, 3, 2, 1, 0	Бит маски прерывания по окончанию преобразования канала n
-	31-8	Зарезервировано

MIS – регистр состояния прерываний с маскированием

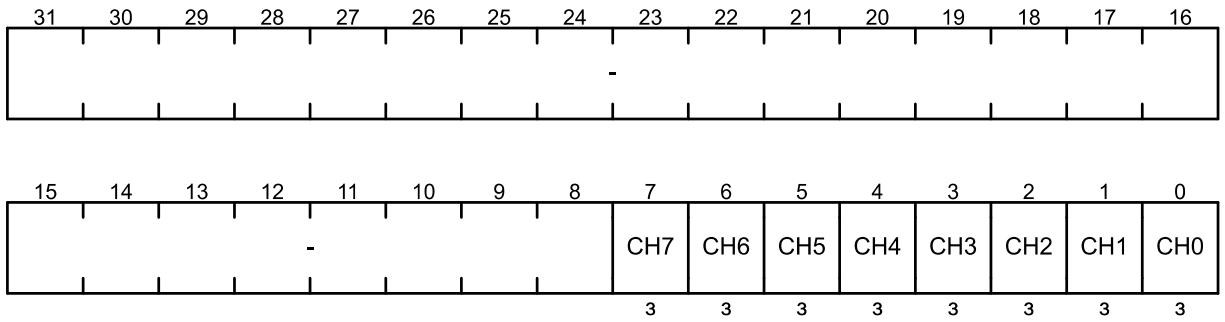
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
CHn	7, 6, 5, 4, 3, 2, 1, 0	Флаг маскированного прерывания по окончанию преобразования канала n
-	31-8	Зарезервировано

IC – регистр сброса прерываний

Смещение: + 14h
Сброс: 0h

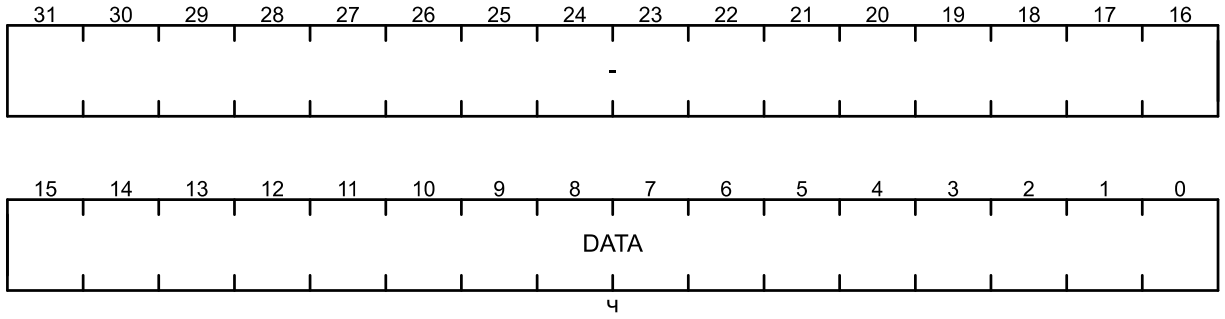


Поле	Биты	Описание
CHn	7, 6, 5, 4, 3, 2, 1, 0	Бит сброса прерывания по каналу n
-	31-8	Зарезервировано

Примечание – Запись единиц в биты регистра сбрасывает соответствующие им флаги в регистрах RIS и MIS, а также прерывания, вызвавшие установку этих флагов.

DATA – массив регистров результатов преобразования каналов

Смещение: DATA + (4*n)h
Сброс: 0h



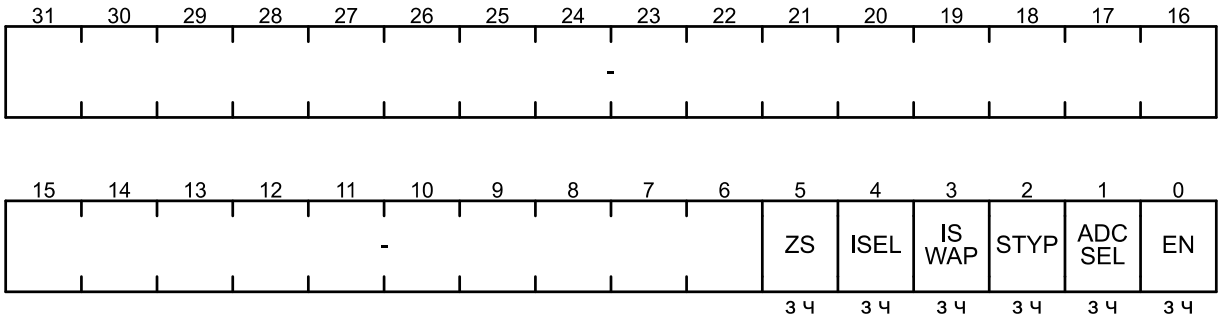
Поле	Биты	Описание
DATA	15-0	Значение результата преобразования канала n
–	31-16	Зарезервировано

A.16 Регистры управления датчиком температуры

Базовый адрес: 3001_1000h

CTRL – регистр настройки датчика температуры

Смещение: + 0h
Сброс: 0h



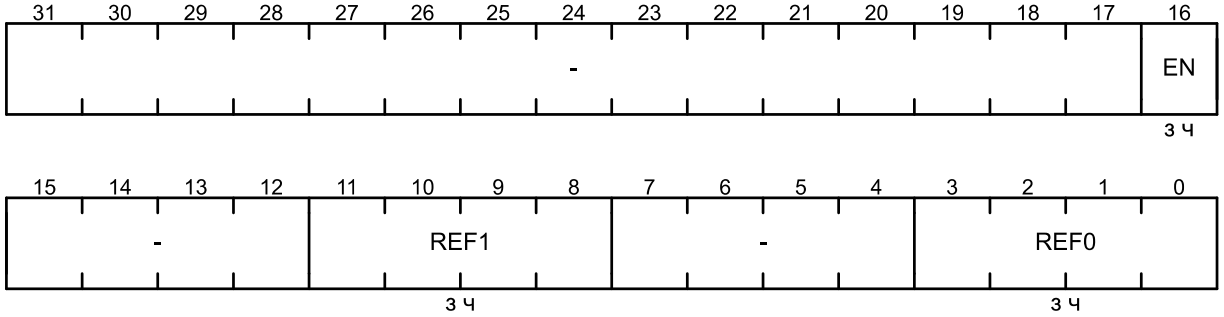
Поле	Биты	Описание	
ZS	5	Нулевое измерение для калибровки датчика	
ISEL	4	Выбор источника опорного тока	
		0	Внешний источник (сигма-дельта АЦП)
		1	Внутренний источник
ISWAP	3	Выбор варианта коммутации источников тока	
		0	Схема №1
		1	Схема №2
STYPE	2	Выбор схемы датчика	
		0	CTAT на двух последовательных диодах, однополярное включение
		1	PTAT на двух параллельных диодах, дифференциальное включение
ADCSEL	1	Выбор АЦП для подключения	
		0	Датчик подключен к сигма-дельта АЦП
		1	Датчик подключен к АЦП последовательного приближения
EN	0	Включение датчика температуры	
		0	Включен
		1	Выключен
–	31-6	Зарезервировано	

A.17 Регистры управления аналоговым компаратором

Базовый адрес: 3801_0000h

DACCTL – регистр настройки ЦАП компаратора

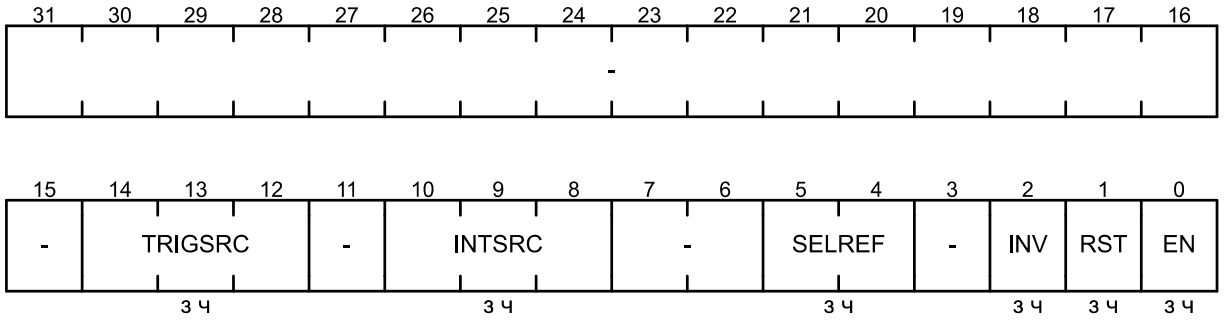
Смещение: + 0h
Сброс: 0h



Поле	Биты	Описание	
EN	16	Включение ЦАП	
		0	Включен
		1	Выключен
REF1	11-8	Уровень напряжения на выходе ЦАП REF1	
REF0	3-0	Уровень напряжения на выходе ЦАП REF0	
–	31-17, 15-12, 7-4	Зарезервировано	

ACMP0CTL – регистр настройки компаратора 0

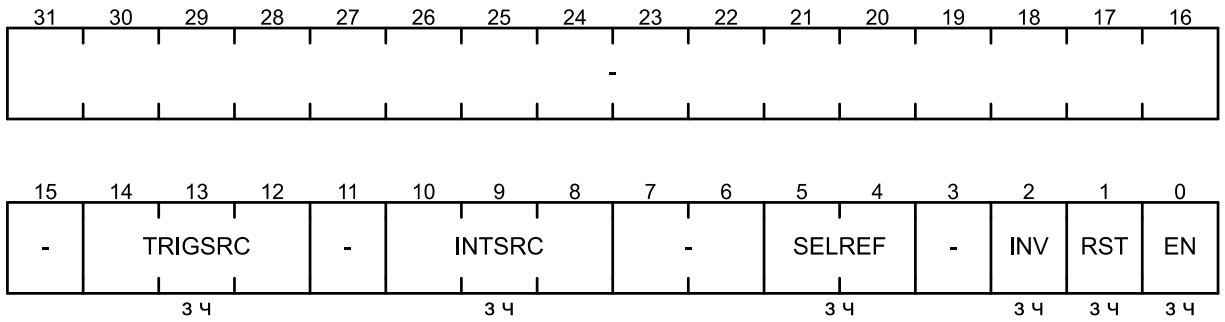
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание	
TRIGSRC	14-12	Выбор события для формирования сигнала TRIG0	
		0	Сигнал TRIG0 не формируется
		1	Положительный фронт
		2	Отрицательный фронт
		3	Любой фронт
		4	Уровень логической единицы
		5	Уровень логического нуля
INTSRC	10-8	Выбор события для формирования прерывания	
		0	Прерывание запрещено
		1	Положительный фронт
		2	Отрицательный фронт
		3	Любой фронт
		4	Уровень логической единицы
		5	Уровень логического нуля
SELREF	5-4	Выбор сигнала, подключаемого на отрицательный вход компаратора	
		0	Вывод микроконтроллера IN0_N
		1	Выход ЦАП REF0
		2	Выход ЦАП REF1
INV	2	Инверсия выходного сигнала компаратора	
		0	На выходе прямой сигнал
		1	На выходе инверсный сигнал
RST	1	Сброс компаратора 0	
		0	Состояние сброса
		1	Состояние работы
EN	0	Бит разрешения работы компаратора 0	
		0	Запрещено
		1	Разрешено
–	31-15, 11, 7-6, 3	Зарезервировано	

ACMP1CTL – регистр настройки компаратора 1

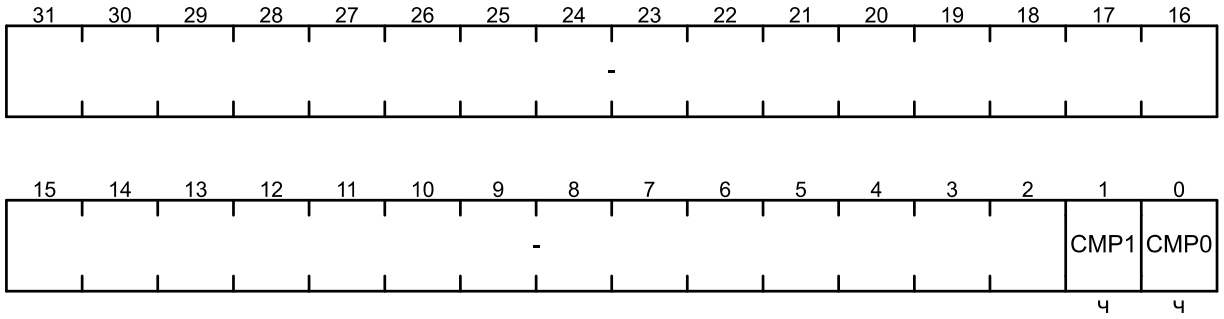
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание	
TRIGSRC	14-12	Выбор события для формирования сигнала TRIG1	
		0	Сигнал TRIG1 не формируется
		1	Положительный фронт
		2	Отрицательный фронт
		3	Любой фронт
		4	Уровень логической единицы
		5	Уровень логического нуля
INTSRC	10-8	Выбор события для формирования прерывания	
		0	Прерывание запрещено
		1	Положительный фронт
		2	Отрицательный фронт
		3	Любой фронт
		4	Уровень логической единицы
		5	Уровень логического нуля
SELREF	5-4	Выбор сигнала, подключаемого на отрицательный вход компаратора	
		0	Вывод микроконтроллера IN0_N
		1	Выход ЦАП REF0
		2	Выход ЦАП REF1
INV	2	Инверсия выходного сигнала компаратора	
		0	На выходе прямой сигнал
		1	На выходе инверсный сигнал
RST	1	Сброс компаратора 1	
		0	Состояние сброса
		1	Состояние работы
EN	0	Бит разрешения работы компаратора 1	
		0	Запрещено
		1	Разрешено
–	31-15, 11, 7-6, 3	Зарезервировано	

STATUS – регистр статуса компаратора

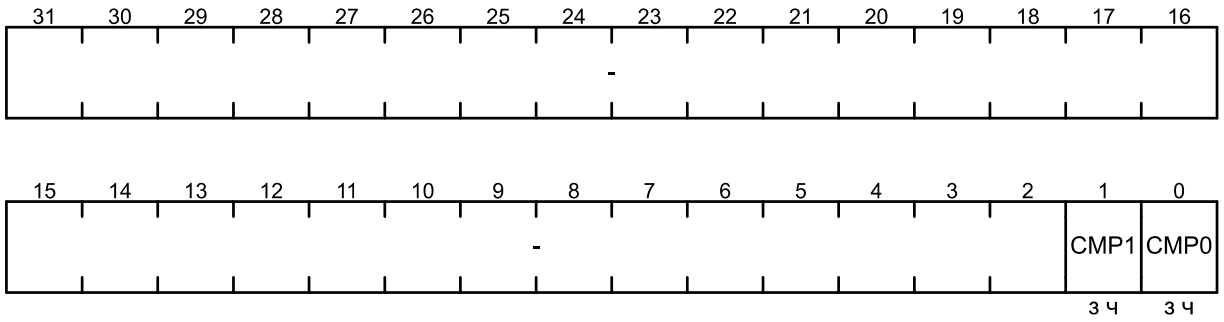
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание
CMP1	1	Состояние выхода компаратора 1
CMP0	0	Состояние выхода компаратора 0
–	31-2	Зарезервировано

INTEN – регистр разрешения прерываний компаратора

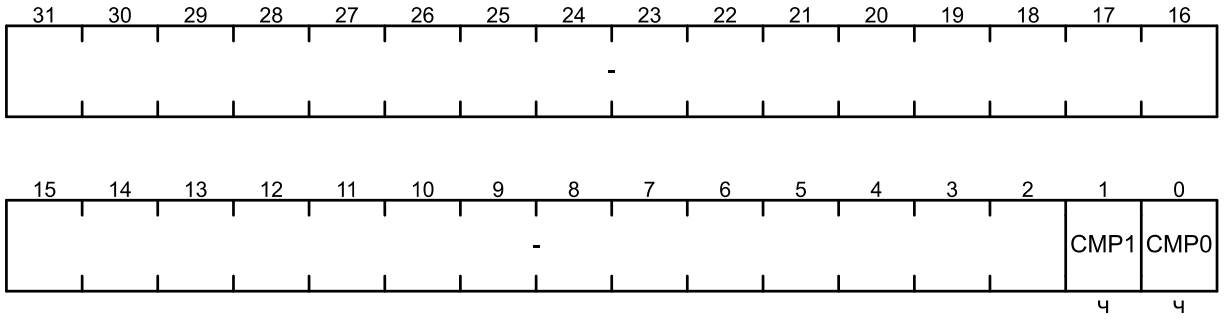
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
CMP1	1	Разрешение прерывания от компаратора 1
CMP0	0	Разрешение прерывания от компаратора 0
–	31-2	Зарезервировано

RIS – регистр состояний прерываний

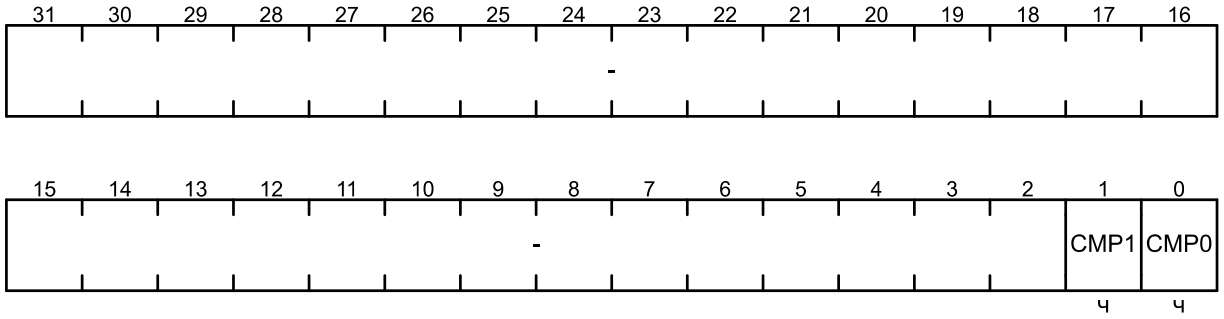
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание
CMP1	1	Запрос на прерывание от компаратора 1
CMP0	0	Запрос на прерывание от компаратора 0
–	31-6	Зарезервировано

MIS – регистр состояний прерываний с маскированием

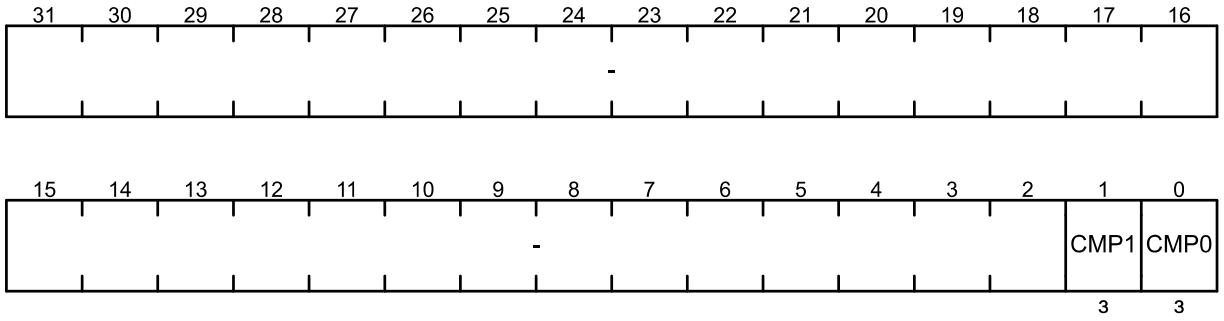
Смещение: + 18h
Сброс: 0h



Поле	Биты	Описание
CMP1	1	Запрос на прерывание от компаратора 1
CMP0	0	Запрос на прерывание от компаратора 0
–	31-2	Зарезервировано

ICR – регистр сброса прерываний

Смещение: + 1Ch
Сброс: 0h



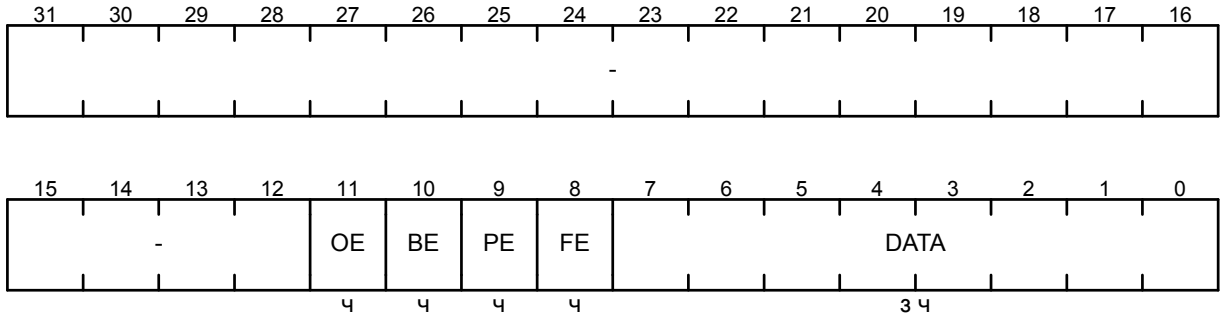
Поле	Биты	Описание
CMP1	1	Сброс прерывания от компаратора 1
CMP0	0	Сброс прерывания от компаратора 0
–	31-2	Зарезервировано

A.18 Регистры приемопередатчика UART

Базовый адрес:	3000_6000h	Регистры приемопередатчика UART0
	3000_7000h	Регистры приемопередатчика UART1
	3000_8000h	Регистры приемопередатчика UART2
	3000_9000h	Регистры приемопередатчика UART3
	3000_A000h	Регистры приемопередатчика UART4

DR – регистр данных

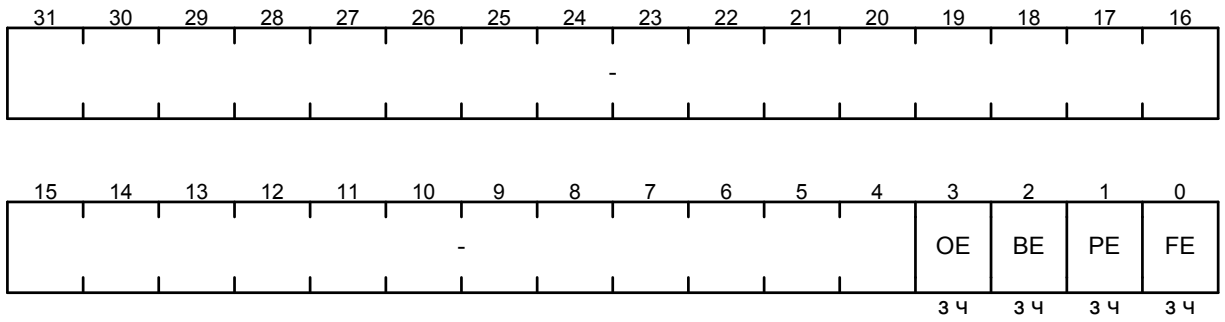
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание
OE, BE, PE, FE	11, 10, 9, 8	См. далее описание бит в регистре RSR
DATA	7-0	Поле данных. Результатом записи в поле DATA является размещение байта в буфере передатчика, а результатом чтения – считывание байта из буфера приемника
–	31-12	Зарезервировано

RSR – регистр состояния приемника и сброса ошибки приемника

Смещение: + 04h
Сброс: 0h

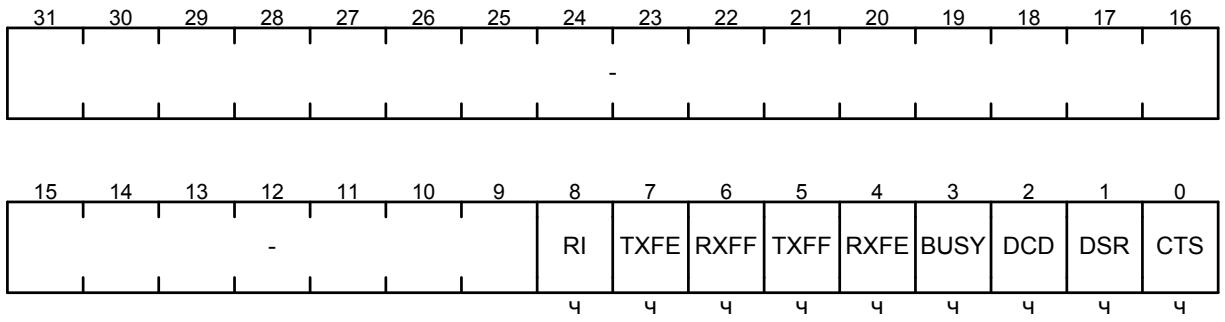


Поле	Биты	Описание	
OE	3	Флаг переполнения буфера приемника	
		0	В буфере есть свободное место или бит был сброшен после записи в регистр RSR. Содержимое буфера остается верным, так как перезаписан был только сдвиговый регистр. Центральный процессор должен считать данные для того, чтобы освободить буфер
		1	Буфер заполнен, а данные продолжают поступать
BE	2	Флаг разрыва линии	
		0	Нормальная работа или бит был сброшен после записи в регистр RSR
		1	Обнаружен признак разрыва линии, то есть наличие низкого логического уровня на входе приемника в течение времени, большего, чем длительность передачи полного кадра данных (включая стартовый, стоповый биты и бит проверки на четность). При включенном режиме FIFO данная ошибка ассоциируется с последним байтом, поступившим в буфер. В случае обнаружения разрыва линии в буфер загружается только один нулевой кадр. Прием данных возобновляется только после перехода линии в логическую единицу и последующего обнаружения корректного стартового бита
PE	1	Флаг ошибки контроля четности	
		0	Нормальная работа или бит был сброшен после записи в регистр RSR
		1	Четность принятого кадра данных не соответствует установкам битов EPS и SPS в регистре управления линией LCRH. При включенном режиме FIFO данная ошибка ассоциируется с байтом, находящимся на вершине буфера
FE	0	Флаг ошибки в структуре кадра	
		0	Нормальная работа или бит был сброшен после записи в регистр RSR
		1	В принятом символе не обнаружен корректный стоповый бит (единица). При включенном режиме FIFO данная ошибка ассоциируется с байтом, находящимся на вершине буфера
–	31-4	Зарезервировано	

Примечание – Все флаги сбрасываются записью единицы.

FR – регистр флагов

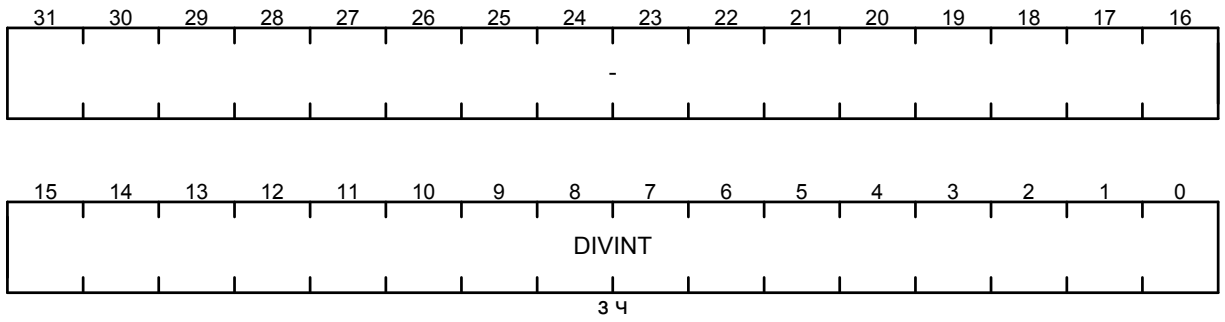
Смещение: + 18h
Сброс: 1h



Поле	Биты	Описание	
RI	8	Флаг состояния линии UARTx_RI	
		0	Линия неатктивна
		1	Линия активна
TXFE/ RXFE	7/ 4	Флаг пустоты буфера передатчика/приемника. Установка флага зависит от состояния бита FEN регистра LCRH	
		0	Буфер не пуст
		1	Буфер пуст
		Примечание – Бит TXFE/RXFE не дает никакой информации о наличии данных в сдвиговом передающем регистре.	
RXFF/ TXFF	6/ 5	Флаг заполнения буфера приемника/передатчика. Установка флага зависит от состояния бита FEN регистра LCRH (т. е. включен режим FIFO или нет)	
		0	Буфер не заполнен
		1	Если режим FIFO запрещен, бит устанавливается, когда буферный регистр приемника/передатчика занят. Если режим FIFO разрешен бит устанавливается, если заполнен буфер приемника/ передатчика
BUSY	3	Бит занятости блока UART	
		0	Блок не занят
		1	Блок передает данные на линию. Бит остается установленным до тех пор, пока данные, включая стоповые биты, не будут полностью переданы. Также бит устанавливается при наличии данных в буфере передатчика, вне зависимости от состояния приемопередатчика (даже если он запрещен)
DCD	2	Флаг состояния линии UARTx_DCD	
		0	Линия неатктивна
		1	Линия активна
DSR	1	Флаг состояния линии UARTx_DSR	
		0	Линия неатктивна
		1	Линия активна
CTS	0	Флаг состояния линии UARTx_CTS	
		0	Линия неатктивна
		1	Линия активна
–	31-9	Зарезервировано	

IBRD – регистр целой части делителя скорости обмена данными

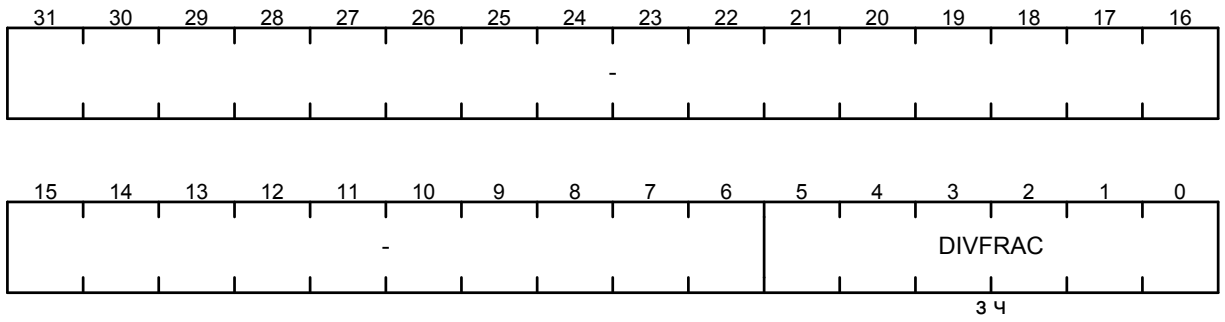
Смещение: + 24h
Сброс: 0h



Поле	Биты	Описание
DIVINT	15-0	Целая часть коэффициента деления частоты для формирования тактового сигнала передачи данных. Минимальное значение 0001h
–	31-16	Зарезервировано

FBRD – регистр дробной части делителя скорости обмена данными

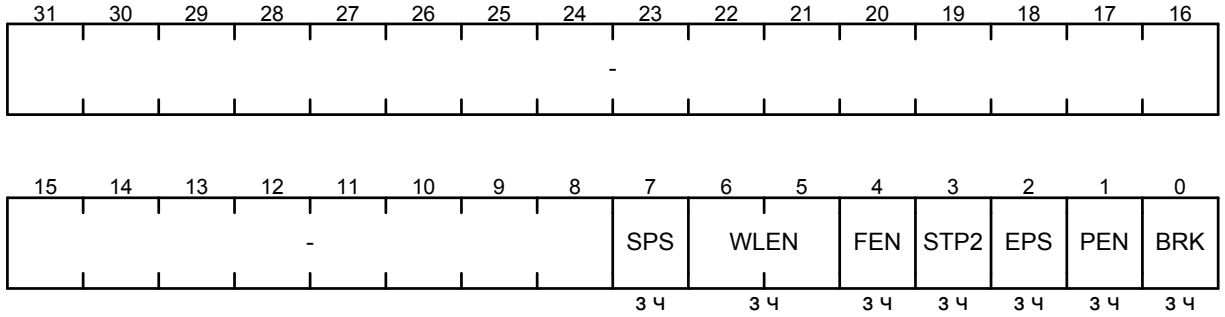
Смещение: + 28h
Сброс: 0h



Поле	Биты	Описание
DIVFRAC	5-0	Дробная часть коэффициента деления частоты для формирования тактового сигнала передачи данных. При DIVINT = FFFFh, значение DIVFRAC может быть только 00h. Примечание – Невыполнение этого условия приведет к прерыванию приема/передачи.
–	31-6	Зарезервировано

LCRH – регистр управления линией

Смещение: + 2Ch
Сброс: 0h



Поле	Биты	Описание	
SPS	7	Бит разрешения передачи бита четности с фиксированным значением	
		0	Запрещено
		1	На месте бита четности передается инверсное значение бита EPS, оно же проверяется при приеме данных. При EPS = 0 на месте бита четности передается единица. При EPS = 1 на месте бита четности передается ноль
WLEN	6-5	Поле количества передаваемых/принимаемых информационных бит	
		00	5 бит
		01	6 бит
		10	7 бит
		11	8 бит

Поле	Биты	Описание	
FEN	4	Бит включения режима FIFO буфера приемника и передатчика	
		0	Выключено
		1	Включено
STP2	3	Бит выбора режима передачи стопового бита	
		0	Один стоповый бит
		1	Два стоповых бита (следует помнить, что приемник не проверяет наличие дополнительного стопового бита в кадре)
EPS	2	Бит паритета. Определяет четность числа, до которого дополняется количество единиц в информационной части кадра	
		0	Нечетное число
		1	Четное число
PEN	1	Бит включения проверки четности	
		0	Выключено. Кадр не содержит бита четности
		1	Включено. Бит четности передается в кадре и проверяется
BRK	0	Флаг разрыва линии	
		0	Нормальная работа
		1	По завершении передачи текущего символа на выходе передатчика устанавливается низкий уровень сигнала. Для правильного выполнения этой операции программное обеспечение должно обеспечить передачу сигнала разрыва в течение, как минимум, времени передачи двух информационных кадров
–	31-8	Зарезервировано	

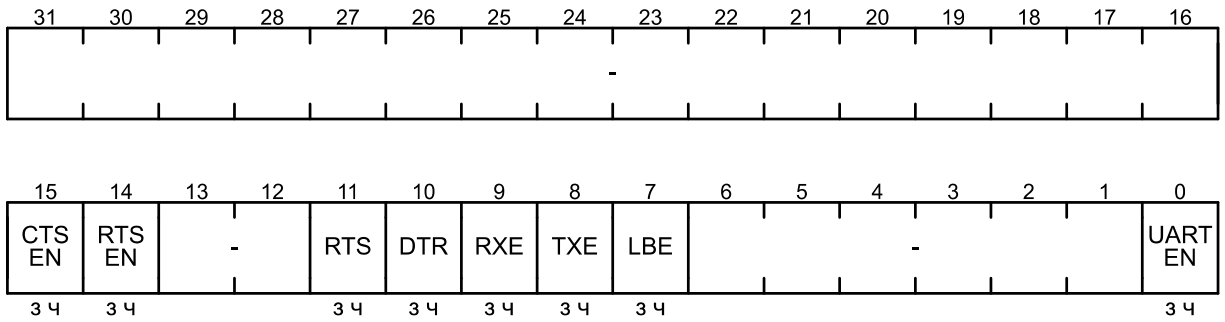
Примечание – Дополнительная информация о бите паритета кадра в таблице А.21.1.

Таблица А.21.1 – Зависимость бита паритета в кадре от состояния битов регистра LCRH

Биты регистра LCRH			Наличие и состояние бита паритета
SPS	EPS	PEN	
Не важно	Не важно	0	Не передается, не проверяется
0	0	1	Проверка нечетности слова данных
0	1	1	Проверка четности слова данных
1	0	1	Бит четности постоянно равен единице
1	1	1	Бит четности постоянно равен нулю

CR – регистр управления

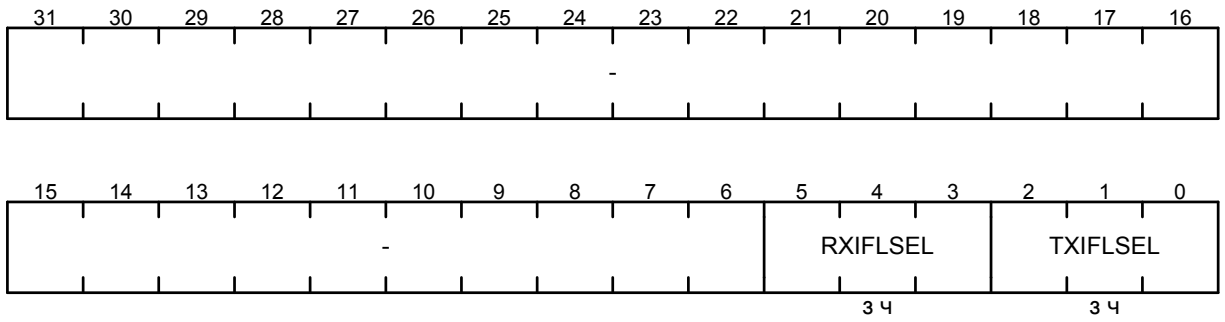
Смещение: + 30h
Сброс: 300h



Поле	Биты	Описание	
CTSEN	15	Бит разрешения аппаратного управления потоком данных по линии CTS	
		0	Запрещено
		1	Разрешено. Данные передаются в линию только при активном значении сигнала UARTx_CTS
RTSEN	14	Бит разрешения аппаратного управления потоком данных по линии UARTx_RTS	
		0	Запрещено
		1	Разрешено. Запрос данных от внешнего устройства осуществляется только при наличии свободного места в буфере приемника
RTS	11	Бит программного управления состоянием линии модема UARTx_RTS	
		0	Высокий уровень
		1	Низкий уровень
DTR	10	Бит программного управления состоянием линии модема UARTx_DTR	
		0	Высокий уровень
		1	Низкий уровень
RXE	9	Бит разрешения приема	
		0	Запрещено
		1	Разрешено
TXE	8	Бит разрешения передачи	
		0	Запрещено
		1	Разрешено
LBE	7	Бит включения режима «петли» - выход передатчика соединяется со входом приёмника	
		0	Запрещено
		1	Разрешено
UARTEN	0	Бит разрешения работы приемопередатчика	
		0	Запрещено
		1	Разрешено
–	31-16, 13-12, 6-1	Зарезервировано	

IFLS – регистр порога прерывания по заполнению буфера в режиме FIFO

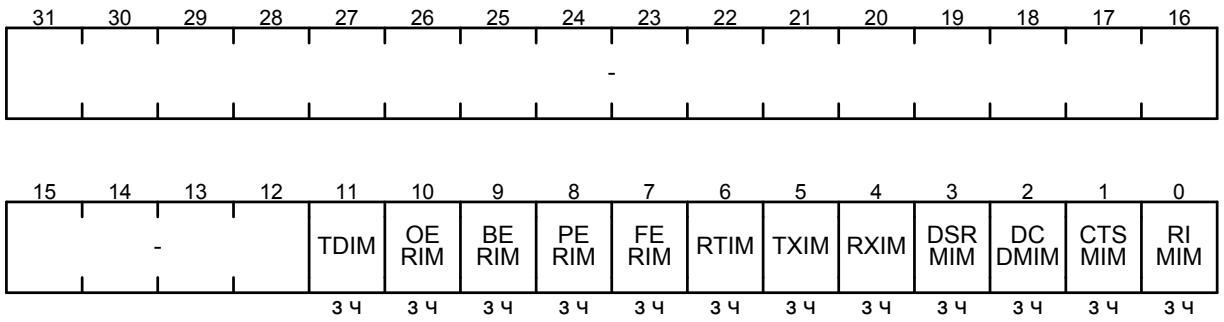
Смещение: + 34h
Сброс: 12h



Поле	Биты	Описание	
RXIFLSEL/ TXIFLSEL	5-3/ 2-0	Порог заполнения/опустошения буфера приемника/передатчика, по достижении которого будет генерироваться прерывание	
		000	Заполнение/опустошение на 1/8
		001	Заполнение/опустошение на 1/4
		010	Заполнение/опустошение на 1/2 (по умолчанию)
		011	Заполнение/опустошение на 3/4
		100	Заполнение/опустошение на 7/8
		Комбинации 101, 110 и 111 зарезервированы	
–	31-6	Зарезервировано	

IMSC – регистр маски прерываний

Смещение: + 38h
Сброс: 0h

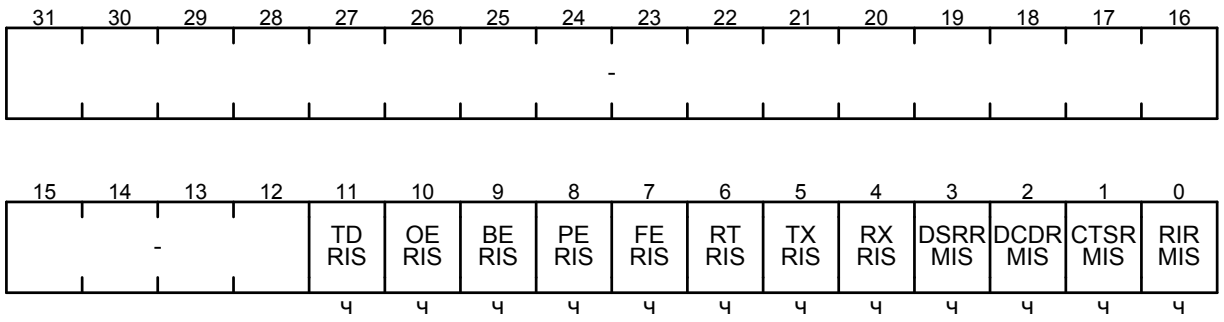


Поле	Биты	Описание
TDIM	11	Окончание передачи в линии
OERIM	10	Переполнение буфера приемника
BERIM	9	Разрыв линии
PERIM	8	Ошибка контроля четности
FERIM	7	Ошибка в структуре кадра
RTIM	6	Таймаут приема данных
TXIM	5	Порог опустошения буфер передатчика
RXIM	4	Порог переполнения буфера приемника
DSRMIM	3	Флаг изменения состояния линии UARTx_DSR
DCDMIM	2	Флаг изменения состояния линии UARTx_DCD
CTSMIM	1	Флаг изменения состояния линии UARTx_CTS
RIMIM	0	Флаг изменениясостояния линии UARTx_RI
–	31-12	Зарезервировано

Примечание – Маска прерываний формируется установкой бит.

RIS – регистр состояния прерываний

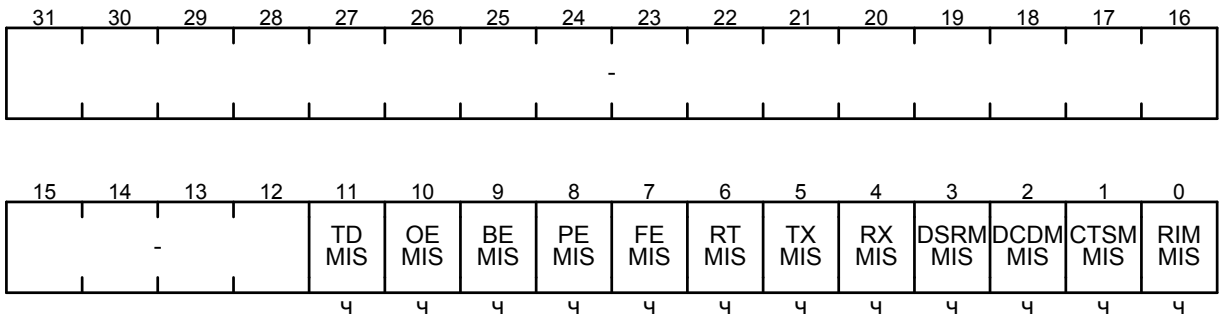
Смещение: + 3Ch
Сброс: 0h



Поле	Биты	Описание
TDRIS	11	Флаг окончания передачи в линии
OERIS	10	Флаг переполнения буфера приемника
BERIS	9	Флаг разрыва линии
PERIS	8	Флаг ошибки контроля четности
FERIS	7	Флаг ошибки в структуре кадра
RTRIS	6	Флаг таймаута приема данных
TXRIS	5	Флаг порога опустошения буфер передатчика
RXRIS	4	Флаг порога переполнения буфера приемника
DSRRMIS	3	Флаг изменения состояния линии UARTx_DSR
DCDRMIS	2	Флаг изменения состояния линии UARTx_DCD
CTSRMIS	1	Флаг изменения состояния линии UARTx_CTS
RIRMIS	0	Флаг изменения состояния линии UARTx_RI
–	31-12	Зарезервировано

MIS – регистр состояния прерываний с маскированием

Смещение: + 40h
Сброс: 0h

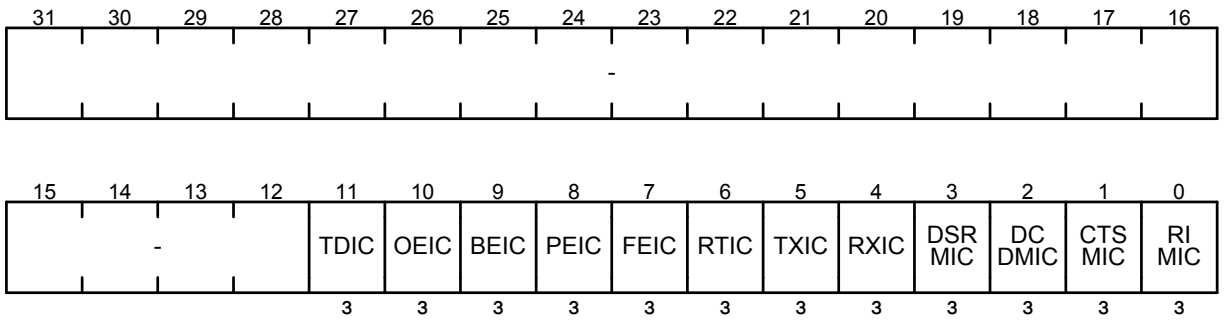


Поле	Биты	Описание
TDMIS	11	Флаг окончания передачи в линии
OEMIS	10	Флаг переполнения буфера приемника
BEMIS	9	Флаг разрыва линии
PEMIS	8	Флаг ошибки контроля четности
FEMIS	7	Флаг ошибки в структуре кадра
RTMIS	6	Флаг таймаута приема данных
TXMIS	5	Флаг порога опустошения буфера передатчика
RXMIS	4	Флаг порога переполнения буфера приемника
DSRMMIS	3	Флаг изменения состояния линии UARTx_DSR
DCDMMIS	2	Флаг изменения состояния линии UARTx_DCD
CTSMMIS	1	Флаг изменения состояния линии UARTx_CTS
RIMMIS	0	Флаг изменения состояния линии UARTx_RI
–	31-12	Зарезервировано

Примечание – Устанавливаются флаги, которые закрыты маской регистра IMSC.

ICR – регистр сброса прерываний

Смещение: + 44h
Сброс: 0h

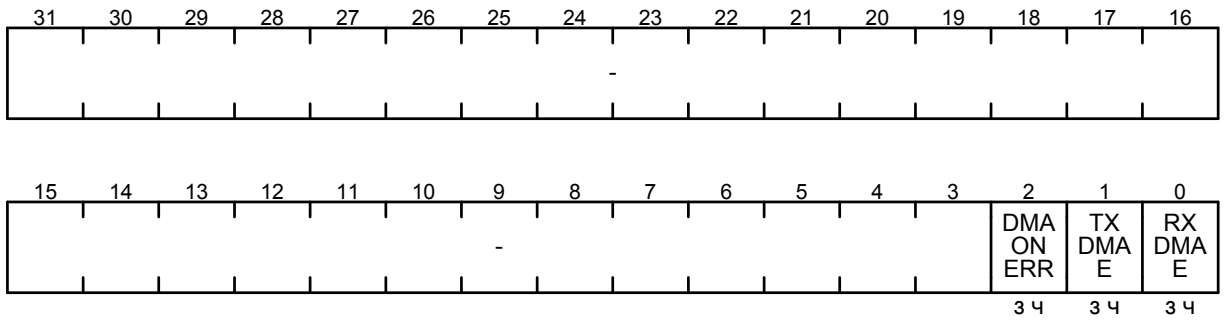


Поле	Биты	Описание
TDIC	11	Бит сброса флага окончания передачи в линии
OEIC	10	Бит сброса флага переполнения буфера приемника
BEIC	9	Бит сброса флага разрыва линии
PEIC	8	Бит сброса флага ошибки контроля четности
FEIC	7	Бит сброса флага ошибки в структуре кадра
RTIC	6	Бит сброса флага таймаута приема данных
TXIC	5	Бит сброса флага порога опустошения буфера передатчика
RXIC	4	Бит сброса флага порога переполнения буфера приемника
DSRMIC	3	Бит сброса флага изменения состояния линии UARTx_DSR
DCDMIC	2	Бит сброса флага изменения состояния линии UARTx_DCD
CTSMIC	1	Бит сброса флага изменения состояния линии UARTx_CTS
RIMIC	0	Бит сброса флага изменения состояния линии UARTx_RI
–	31-12	Зарезервировано

Примечание – Запись единиц в биты регистра сбрасывает соответствующие им флаги в регистрах RIS и MIS, а также прерывания, вызвавшие установку этих флагов.

DMACR – регистр управления прямым доступом к памяти

Смещение: + 48h
Сброс: 0h



Поле	Биты	Описание
DMAONERR	2	Блокирование запросов UARTRXDMASREQ и UARTRXDMABREQ от приемника в случае прерывания по ошибке
		0 Выключено
		1 Включено
TXDMAE/ RXDMAE	1/ 0	Бит разрешения формирования запросов блока DMA для обслуживания буфера передатчика/приемника
		0 Запрещено
		1 Разрешено
–	31-3	Зарезервировано

A.19 Регистры контроллера интерфейса CAN 2.0b

Базовый адрес: 2000_0000h

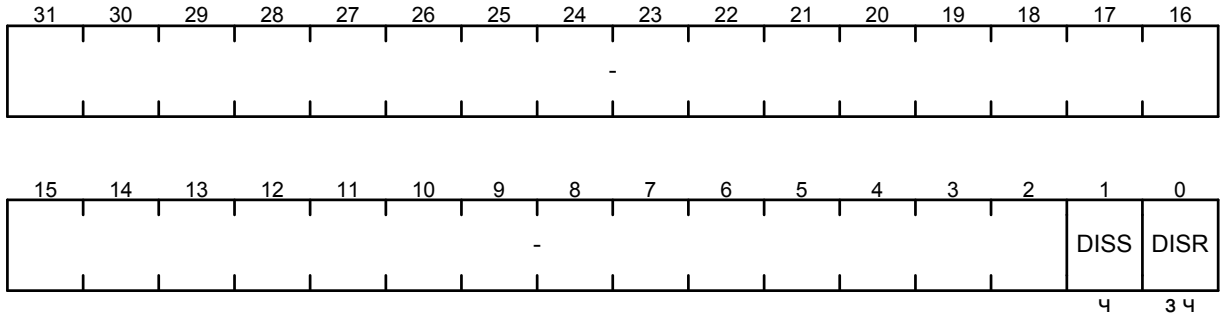
Смещение:		
	+ 100h (LIST)	Регистры свободного списка
	+ 140h (MSPND)	Регистры ждущих прерываний
	+ 180h (MSID)	Регистры индексов сообщений
	+ 200h (Node_0)	Регистры узла 0
	+ 300h (Node_1)	Регистры узла 1
	+ 1000h + 20h*m (Msg_m)	Регистры объекта сообщений m

Мнемоника: Node_n;
Msg_m

Примечание – ln – номер списка от 0 до 2;
n – номер узла 0 или 1;
m – номер объекта сообщения от 0 до 127;
mo – номер объекта сообщения в шестнадцатеричном формате от 0h до 7Fh (что соответствует диапазону от 0 до 127).

CLC – регистр управления частотой

Смещение: + 00h
Сброс: 3h

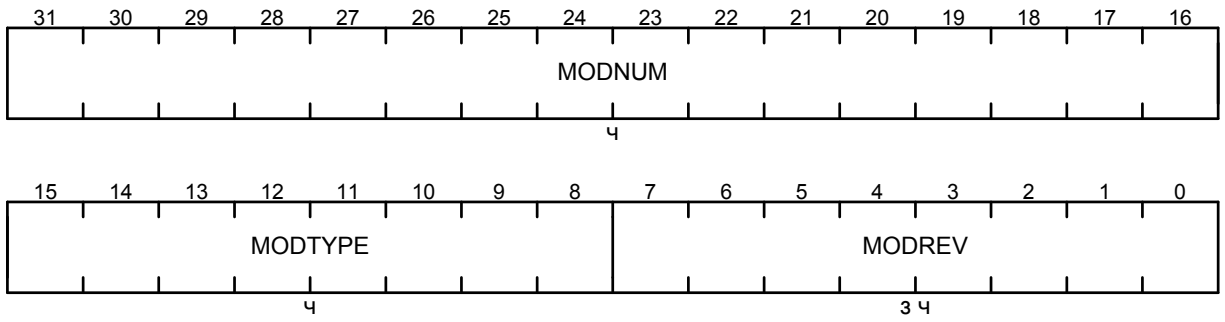


Поле	Биты	Описание	
DISS	1	Бит состояния контроллера CAN	
		0	Включен
		1	Выключен
DISR	0	Бит выключения контроллера CAN	
		0	Нет действий
		1	Запись единицы запускает механизм выключения
–	31-2	Зарезервировано	

Примечание – Когда контроллер CAN находится в выключенном состоянии, только регистр CLC доступен для записи и чтения, доступ к остальным регистрам не возможен.

ID – регистр идентификации

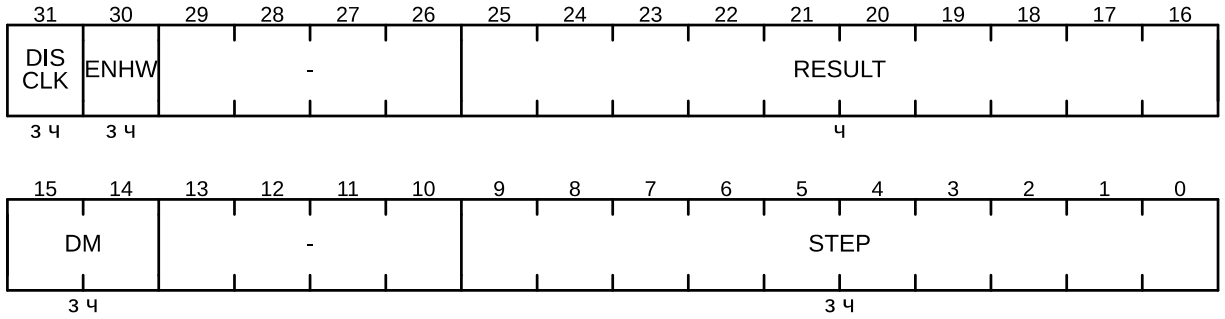
Смещение: + 08h
Сброс: 2B_C051h



Поле	Биты	Описание
MODNUM	31-16	Идентификационный номер контроллера CAN
MODTYPE	15-8	Разрядность контроллера CAN
MODREV	7-0	Число модификаций контроллера CAN

FDR – регистр делителя

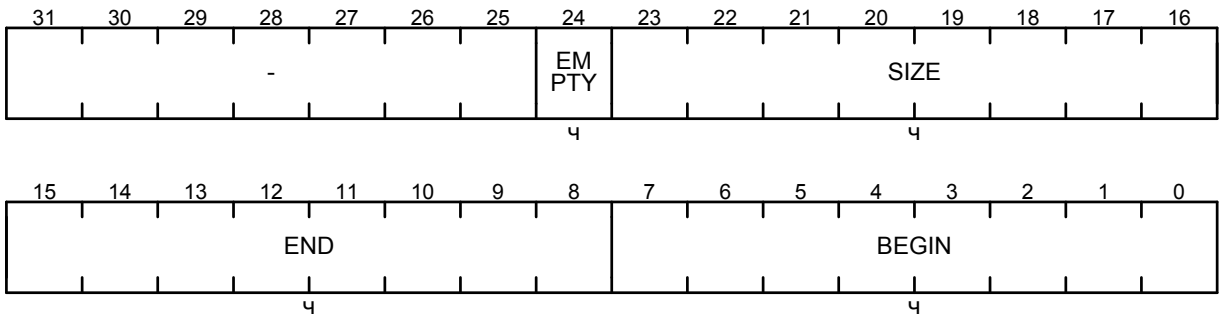
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание	
DISCLK	31	Бит запрета внутреннего тактирования	
		0	Генерирование сигнала FCAN разрешено
		1	Генерирование сигнала FCAN запрещено
ENHW	30	Бит контроля синхронизации. Это бит аппаратно удерживается в сброшенном состоянии и не может быть установлен	
RESULT	25-16	Счетчик делителя частоты	
DM	15-14	Поле задания режима делителя частоты	
		00b	Счетчик выключен
		01b	Нормальный режим работы. Синхросигнал FOUT формируется. Сигнал сброса внешнего делителя в состоянии логического нуля. При активации режима поле RESULT загружается значением 3FFh. Далее периодически загружается значением из STEP
		10b	Режим дробного деления. Синхросигнал FOUT формируется. Сигнал сброса внешнего делителя в состоянии логического нуля. При активации режима поле RESULT загружается значением 3FFh. Далее периодически загружается значением из STEP
		11b	Синхросигнал FOUT не генерируется. Сигнал сброса внешнего делителя в состоянии логической единицы. Поле RESULT не меняется
STEP	9-0	Шаг делителя. Поле хранит значение, которое загружается в RESULT при переполнении счетчика делителя	
–	29-26, 10	Зарезервировано	

LIST – массив регистров свободного списка

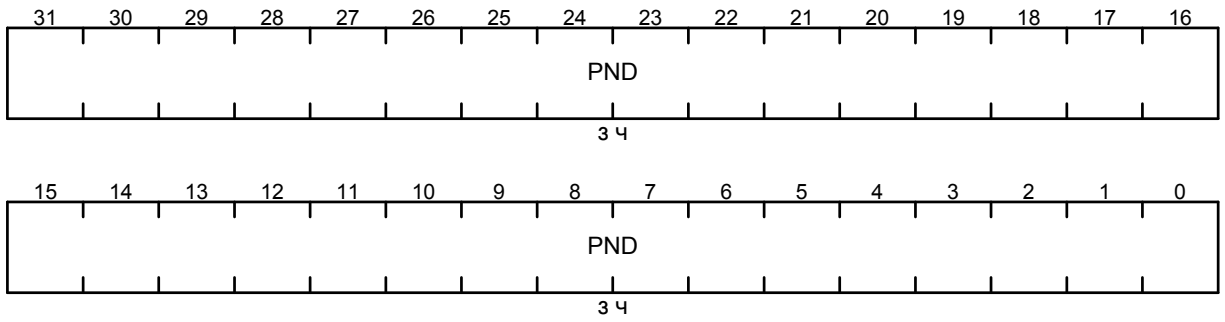
Смещение: LIST + 4×ln
Сброс: 7F_7F00h (для списка 0), 100_0000h (для списков 1 – 2)



Поле	Биты	Описание	
EMPTY	24	Индикатор пустого списка	
		0	В списке есть как минимум один элемент
		1	Список пуст
SIZE	23-16	Размер списка. Количество элементов (объектов сообщений) в списке. Значение поля SIZE всегда на единицу меньше числа элементов. Если список пуст, SIZE = 00h	
END	15-8	Номер объекта сообщения, находящегося последним в списке. Поле может принимать значения от 00h до 7Fh, согласно количеству объектов сообщений (128)	
BEGIN	7-0	Номер объекта сообщения, находящегося первым в списке. Поле может принимать значения от 00h до 7Fh, согласно количеству объектов сообщений	
–	31-25	Зарезервировано	

MSPND – массив регистров ждущих прерываний

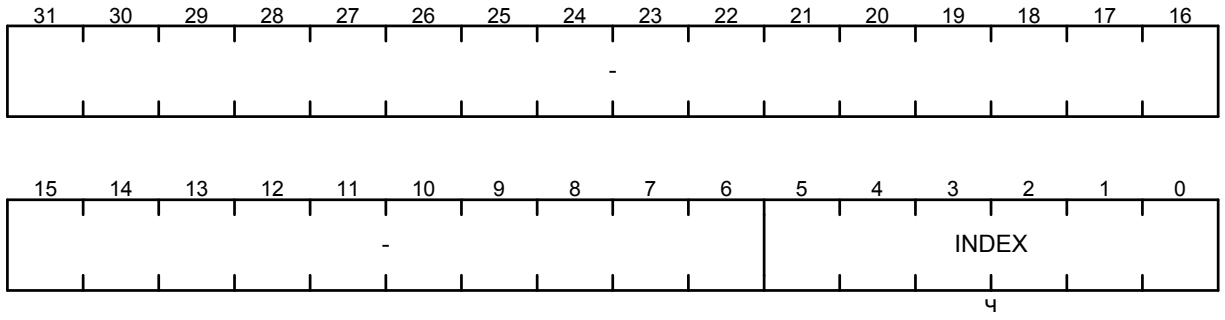
Смещение: MSPND + 4×ln
Сброс: 0h



Поле	Биты	Описание
PND	31-0	Поле ждущих битов сообщений. Каждому объекту сообщения выделяется один бит. Биты устанавливаются только аппаратно. Установленные биты сбрасываются аппаратно по окончании обслуживания запроса прерывания или могут быть сброшены в любой момент программно

MSID – массив регистров индекса сообщения

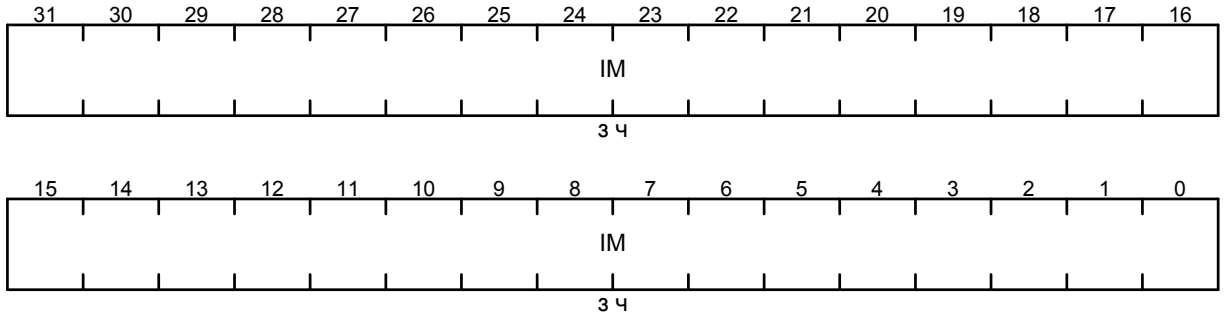
Смещение: MSID + 4×ln
Сброс: 20h



Поле	Биты	Описание
INDEX	5-0	Поле номера ждущего бита. Если в регистре MSPND есть установленные биты, которые не маскируются соответствующими битами регистра MSIMASK, то поле INDEX будет указывать на самый младший из них. Если в регистре MSPND нет установленных битов или они замаскированы, то в поле INDEX будет находиться значение 20h, указывающее на бит 31 регистра MSPND
–	31-6	Зарезервировано

MSIMASK – регистр маски индекса сообщения

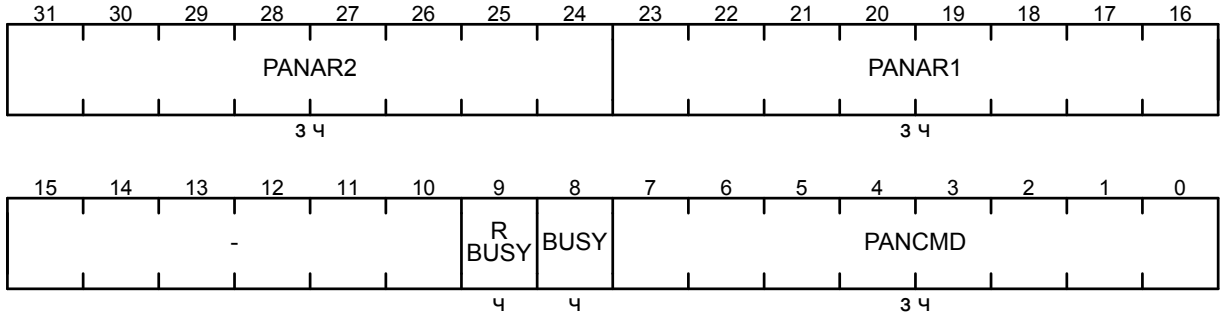
Смещение: + 1C0h
Сброс: 0h



Поле	Биты	Описание
IM	31-0	Маска для ждущих битов сообщений. Учитывается состояние только тех бит регистра MSPND, для которых в поле IM установлены соответствующие биты

PANCTR – регистр панели команд

Смещение: + 1C4h
Сброс: 301h



Поле	Биты	Описание	
PANAR2	31-24	Панель аргумента 2 (см. таблицу А.19.1)	
PANAR1	23-16	Панель аргумента 1 (см. таблицу А.19.1)	
RBUSY	9	Флаг занятости панелей аргументов	
		0	Нет действий
		1	Выполняется команда списка, результат выполнения которой будет записан в поле PANAR1 и поле PANAR2
BUSY	8	Флаг занятости панелей аргументов	
		0	Панели готовы для записи
		1	Панели заняты – ожидают записи по окончании выполнения команды
PANCMD	7-0	Поле команды (см. таблицу А.19.1). После выполнения команды в это поле записывается 00h	
–	15-10	Зарезервировано	

Таблица А.19.1 – Коды команд работы со списками

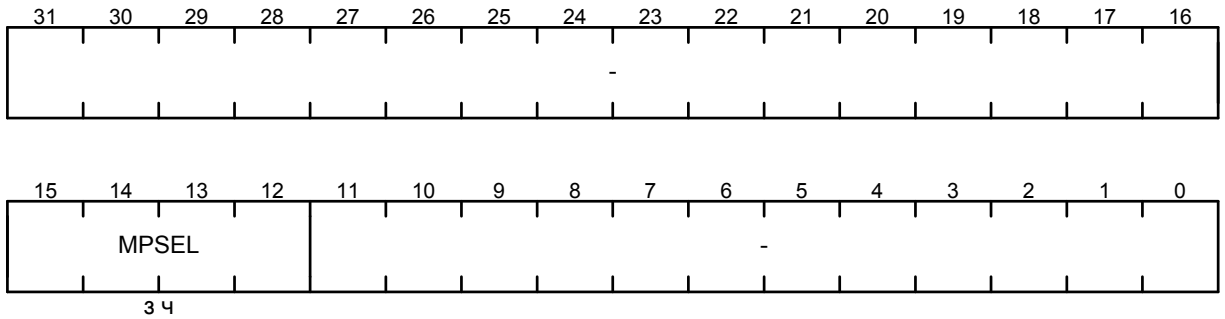
Поле PANCMD	Поле PANAR2	Поле PANAR1	Описание команды
00h	–	–	Нет операции. Никаких действий не выполняется
01h	Результат: бит 7 – ошибка, бит 6 – не определен	–	Инициализация списков. Запуск инициализации для очистки битовых полей CTRL и LIST всех объектов сообщений. Регистры LIST0 – LIST8 устанавливаются в свои значения после сброса. Это приводит к переносу всех объектов сообщений в список № 0 (список нераспределенных объектов сообщений). Инициализация списков требует, чтобы биты INIT и CCE регистра NCR были установлены для обоих узлов. Бит 7 поля PANAR2 сигнализирует о результате операции: - 0 – инициализация завершена успешно; - 1 – инициализация не завершена, поскольку не все биты INIT и CCE были установлены. Команда инициализации списков автоматически запускается при каждом сбросе контроллера CAN, за исключением случая, когда все регистры объектов сообщений уже сброшены
02h	Аргумент: номер списка	Аргумент: номер объекта сообщения	Статическое занесение объекта сообщения в список. Объект сообщения переносится из текущего списка в список, указанный полем PANAR2 и добавляется в его конец. Эта команда также используется для дераспределения объекта сообщения, т. е. переноса его в список № 0 (если поле PANAR2 равно 00h)
03h	Аргумент: номер списка. Результат: бит 7 – ошибка, бит 6 – не определен	Результат: номер объекта сообщения	Динамическое занесение объекта сообщения в список. Первый объект сообщения списка № 0 переносится в список, указанный полем PANAR2, и добавляется в его конец. Номер объекта сообщения возвращается полем PANAR1. Бит 7 поля PANAR2 сигнализирует о результате операции: - 0 – операция выполнена; - 1 – операция не выполнена – список № 0 пуст
04h	Аргумент: номер объекта сообщения	Аргумент: текущий номер объекта сообщения	Перемещение по списку вверх. Перенос объекта сообщения с номером PANAR1 на одну позицию выше, чем расположен объект сообщения с номером PANAR2

Продолжение таблицы А.19.1

Поле PANCMD	Поле PANAR2	Поле PANAR1	Описание команды
05h	Аргумент: номер объекта сообщения Результат: бит 7 – ошибка, бит 6 – не определен	Результат: номер добавлен-ного объекта сообщения	Динамическая вставка в список. Первый объект сообщений списка № 0 вставляется на одну позицию выше, чем расположен объект сообщения с номером PANAR2. Номер добавленного объекта сообщения возвращается полем PANAR1. Бит 7 поля PANAR2 сигнализирует о результате операции: - 0 – операция выполнена; - 1 – операция не выполнена – список № 0 пуст
06h	Аргумент: номер объекта сообщения	Аргумент: текущий номер объекта сообщения	Перемещение по списку вниз. Перенос объекта сообщения с номером PANAR1 на одну позицию ниже, чем расположен объект сообщения с номером PANAR2
07h	Аргумент: номер объекта сообщения. Результат: бит 7– ошибка, бит 6 – не определен	Результат: номер добавлен-ного объекта сообщения	Динамическая вставка в список. Первый объект сообщения списка № 0 вставляется на одну позицию ниже, чем расположен объект сообщения с номером PANAR2. Номер добавленного объекта сообщения возвращается полем PANAR1. Бит 7 поля PANAR2 сигнализирует о результате операции: - 0 – операция выполнена; - 1 – операция не выполнена – список № 0 пуст
08h – FFh	–	–	Зарезервировано

MCR – регистр управления

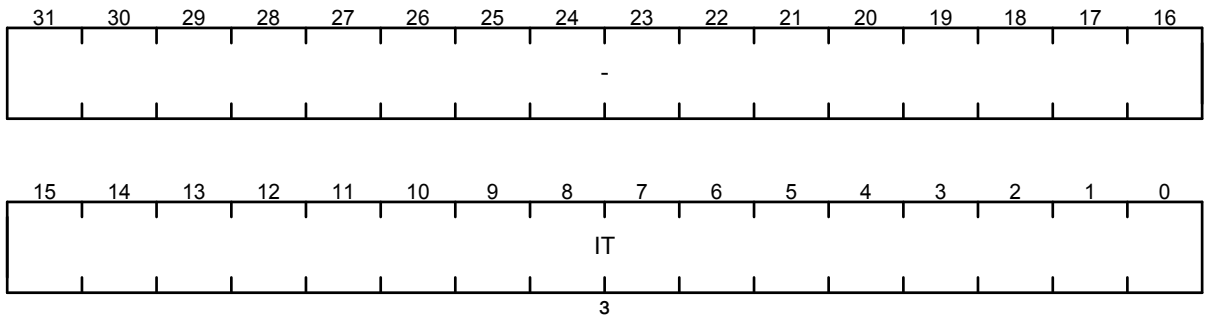
Смещение: + 1C8h
Сброс: 0h



Поле	Биты	Описание
MPSEL	15-12	Поле задания позиции ждущего бита сообщения после приема/ передачи сообщения
–	31-16, 11-0	Зарезервировано

MITR – регистр прерываний

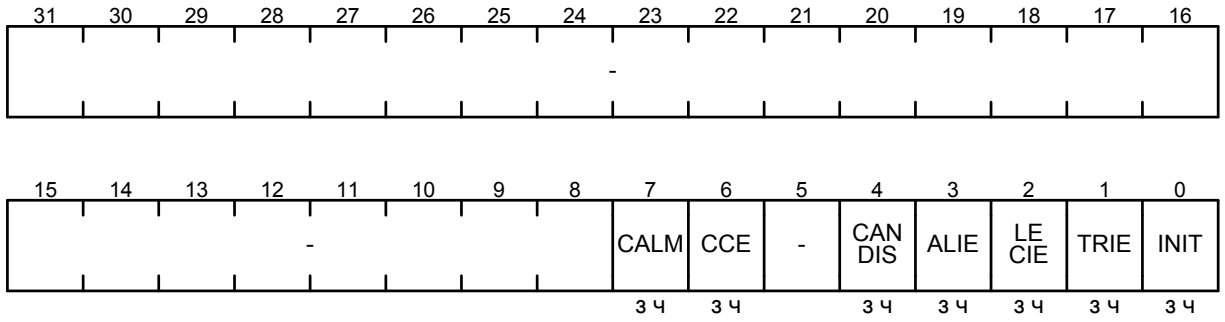
Смещение: + 1CCh
Сброс: 0h



Поле	Биты	Описание
IT	15-0	Поле генератора прерываний. Каждый бит поля связан с одной из линий прерываний. Номера битов от 0 до 15 соответствуют номерам линий прерываний. Для того, чтобы сгенерировать одно или несколько прерываний, следует установить соответствующие биты. Установленные биты сбрасываются аппаратно
–	31-16	Зарезервировано

NCR – регистр управления узла

Смещение: Node_n + 00h
Сброс: 1h

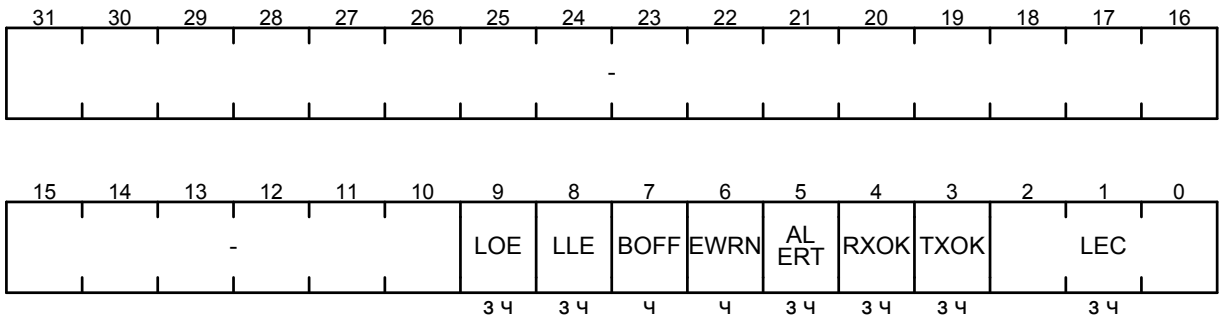


Поле	Биты	Описание	
CALM	7	Бит включения режима анализа узла	
		0	Режим выключен
		1	Установка бита включает режим анализа узла. В этом режиме сообщения могут только приниматься, бит подтверждения не посылается после успешного приема сообщения, флаг активной ошибки посылается рецессивным вместо доминантного. На линии отправки сообщений поддерживается высокий уровень сигнала
		Бит может быть установлен только, если установлен бит INIT	
CCE	6	Бит разрешения изменения конфигурации узла. Управляет доступом к регистрам NBTR, NPCR и NECNT	
		0	Только чтение
		1	Полный доступ

Поле	Биты	Описание	
CANDIS	4	Бит выключения узла	
		0	Сброс бита включает узел
		1	Установка бита выключает узел. Сначала узел переходит в состояние «простоя» или «отключен от шины», далее аппаратно устанавливается бит INIT, и, если разрешено, генерируется прерывание ALERT
ALIE	3	Бит разрешения прерывания ALERT от узла	
		0	Запрещено
		1	Разрешено
LECIE	2	Бит разрешения прерывания от узла при обнаружении кода последней ошибки	
		0	Запрещено
		1	Разрешено
TRIE	1	Бит разрешения прерывания от узла по окончании передачи/приема	
		0	Запрещено
		1	Разрешено
INIT	0	Инициализация узла	
		0	Сброс бита разрешает участие узла в работе CAN шины. Узел ожидает последовательность из 11 рецессивных бит на шине и включается в трафик. Если на момент сброса бита INIT узел находился в состоянии «отключен от шины», начинается процесс выхода из этого состояния в следующем порядке – получение 128 последовательностей бит (каждая из 11 рецессивных бит), выход из состояния «отключен от шины», включение в трафик
		1	Установка бита INIT прекращает участие узла в трафике. Все текущие передачи останавливаются, линии передач переходят в рецессивное состояние. Если на момент установки бита INIT узел находился в состоянии «отключен от шины», процесс выхода из этого состояния продолжается до его завершения. Далее узел остается неактивным до тех пор, пока установлен бит INIT
–	31-8, 5	Зарезервировано	

NSR – регистр состояния узла

Смещение: Node_n + 04h
Сброс: 0h

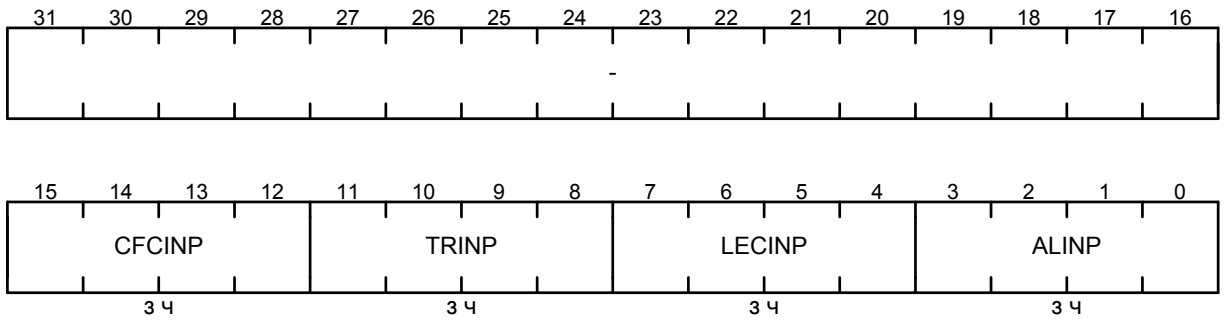


Поле	Биты	Описание	
LOE	9	Флаг ошибки номера списка	
		0	Ошибок не обнаружено
		1	Обнаружена ошибка при фильтрации принимаемого сообщения. В регистре MOSTAT объекта сообщения обнаружен неверный номер списка
		Бит должен сбрасываться программно записью нуля	
LLE	8	Флаг ошибки списка	
		0	Ошибок не обнаружено
		1	Обнаружена ошибка при фильтрации принимаемого сообщения. Количество элементов списка, принадлежащего узлу, отличается от указанного в поле SIZE соответствующего регистра списка
		Бит должен сбрасываться программно записью нуля	
BOFF	7	Флаг состояния «отключен от шины»	
		0	Узел не находится в состоянии «отключен от шины»
		1	Узел находится в состоянии «отключен от шины»
EWRN	6	Флаг критического количества ошибок	
		0	Лимит ошибок еще не достигнут
		1	По крайней мере, один из счетчиков ошибок (REC, TEC) достиг лимита ошибок, заданного полем EWRNLVL регистра NECNT узла
ALERT	5	Флаг предупреждения ALERT	
		0	Нет событий
		1	Произошло одно или несколько не взаимоисключающих событий: - модификация бита BOFF; - модификация/установка бита LOE; - установка бита LLE; - аппаратная установка бита INIT
		Бит должен сбрасываться программно записью нуля	
RXOK	4	Флаг успешного приема сообщения	
		0	Полученных сообщений нет
		1	Сообщение получено
		Бит должен сбрасываться программно записью нуля	
TXOK	3	Флаг успешной передачи сообщения	
		0	Переданных сообщений нет
		1	Сообщение передано без ошибок с получением подтверждения
		Бит должен сбрасываться программно записью нуля	

Поле	Биты	Описание	
LEC	2-0	Поле кода последней из обнаруженных ошибок работы узла	
		000	Ошибок нет
		001	Ошибка стаффинга (STUFF ERROR). Может быть обнаружена во время передачи шестого бита из последовательности шести одинаковых бит в поле сообщения, которое должно быть кодировано методом разрядного заполнения (заключается в том, что после передачи пяти битов одинаковой полярности, шестой бит должен иметь противоположную полярность и вставляться передатчиком в поток данных автоматически, приемник пропускает этот бит)
		010	Ошибка формы (FORM ERROR). Обнаруживается, если: - в битовом поле фиксированного формата содержится количество битов, отличающееся от установленного; - на месте рецессивного бита находятся доминантный или наоборот. Исключение – для приемника доминантный бит в течение последнего бита поля «конец кадра» не интерпретируется как ошибка формы
		011	Ошибка подтверждения (ACKNOWLEGMENT ERROR). Обнаруживается передатчиком всякий раз, когда он не обнаруживает доминантный бит ACK в «области подтверждения»
		100	Разрядная ошибка или ошибка бита 1 (BIT 1 ERROR). Узел, который передает данные на шину, осуществляет мониторинг шины. Ошибка бита 1 имеет место, если при передаче рецессивного «1» бита (за исключением битов полей арбитража и подтверждения) на шине обнаруживается доминантный «0» бит
		101	Разрядная ошибка или ошибка бита 0 (BIT 0 ERROR). Ошибка возникает в случаях: - во время передачи сообщения (или бита подтверждения, флага активной ошибки, флага перезагрузки), узел передает доминантный бит «0», но на шине обнаруживается рецессивный «1»; - во время выхода из состояния «отключен от шины» при каждом обнаружении последовательности из 11 рецессивных битов. В этом случае, ЦП может использовать код 101 для отслеживания длительного простоя шины
		110	Ошибка циклического избыточного кода (CRC ERROR). Передатчик по установленному алгоритму вычисляет значение контрольной суммы (CRC) для передаваемых данных и вставляет ее в сообщение. Приемник, после получения данных, вычисляет CRC по тому же алгоритму, что и передатчик, и сравнивает вычисленное значение с принятым значением. В случае несовпадения фиксируется ошибка
		111	Код разрешения аппаратной записи в поле LEC
		Поле LEC обновляется аппаратно после каждой передачи (успешной или с ошибкой) – в нём устанавливается код ошибки, либо 000b, если её не было. Прерывание по LEC происходит каждый раз при обнаружении ошибки при условии, что оно разрешено битом LECIE регистра NCR. Программно в поле можно записать только 111b, если требуется. Это значение будет находиться в поле до завершения текущей передачи	
–	31-10	Зарезервировано	

NIPR – регистр указателя прерываний узла

Смещение: Node_n + 08h
Сброс: 0h

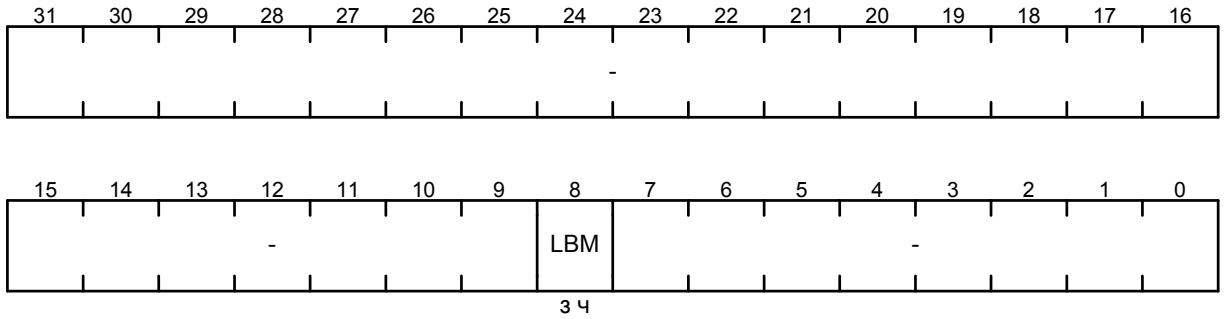


Поле	Биты	Описание
CFCINP	15-12	Указатель для прерывания при переполнении счетчика фреймов узла
TRINP	11-8	Указатель для прерывания по окончании передачи/приема сообщения
LECINP	7-4	Указатель для прерывания при записи кода последней ошибки
ALINP	3-0	Указатель для прерывания ALERT
–	31-16	Зарезервировано

Примечание – Каждый из указателей задает номер одной из 2 линий прерываний. Значение 0h соответствует нулевой линии, значение 1h – линии 1.

NPCR – регистр управления портом узла

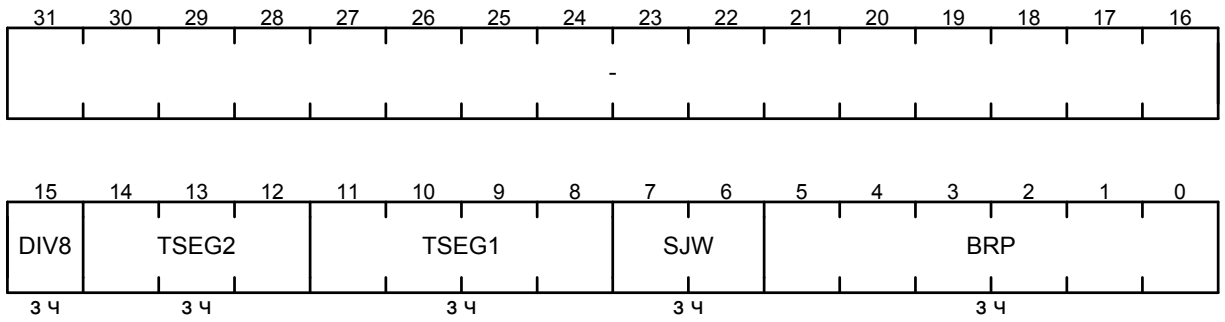
Смещение: Node_n + 0Ch
Сброс: 0h



Поле	Биты	Описание	
LBM	8	Бит включения режима обратной петли (Loop-Back)	
		0	Выключено
		1	Включено. В этом режиме узел подсоединяется к внутренней виртуальной CAN шине. Если для обоих узлов включен режим обратной петли, то они объединяются виртуальной CAN шиной и могут взаимодействовать друг с другом. При этом на внешних выводах узлов, соединенных с внешней физической CAN шиной, поддерживается рецессивный уровень сигнала, т. е. узлы не активны
–	31-9, 7-0	Зарезервировано	

NBTR – регистр синхронизации битов

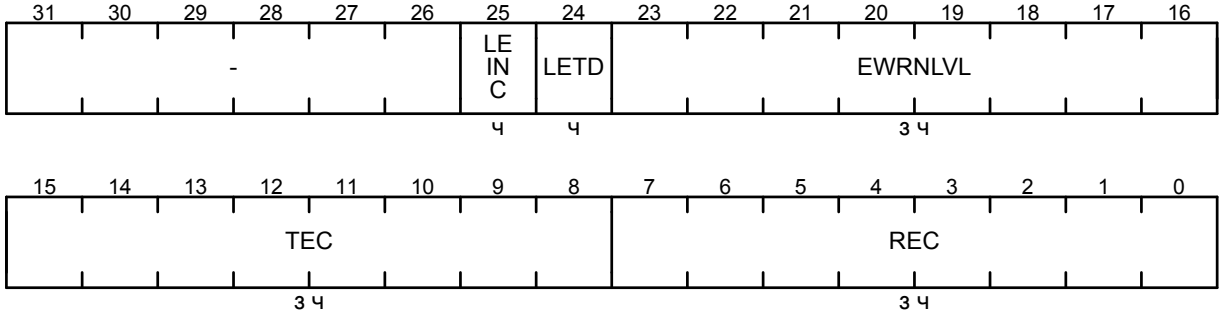
Смещение: Node_n + 10h
Сброс: 0h



Поле	Биты	Описание	
DIV8	15	Делитель частоты на восемь	
		0	Выключено
		1	Включено
TSEG2	14-12	Параметр 2. Временной промежуток от точки выборки до точки передачи, определяемый пользователем. Длительность сегмента равна tq × (TSEG2 + 1) и может быть уменьшена за счет ресинхронизации. Диапазон допустимых значений от 1h до 7h	
TSEG1	11-8	Параметр 1. Временной промежуток от сегмента синхронизации до точки выборки, определяемый пользователем и включающий в себя сегмент распространения. Длительность равна tq × (TSEG1 + 1) и может быть увеличена за счет ресинхронизации. Допустимые значения от 2h до Fh	
SJW	7-6	Ширина перехода ресинхронизации. Длительность равна tq × (SJW + 1)	
BRP	5-0	Предделитель скорости передачи. Длительность одного кванта времени (в тактах частоты): - (BRP + 1), если DIV8 = 0; - 8 × (BRP + 1), если DIV8 = 1	
–	31-16	Зарезервировано	

NECNT – регистр счетчика ошибок узла

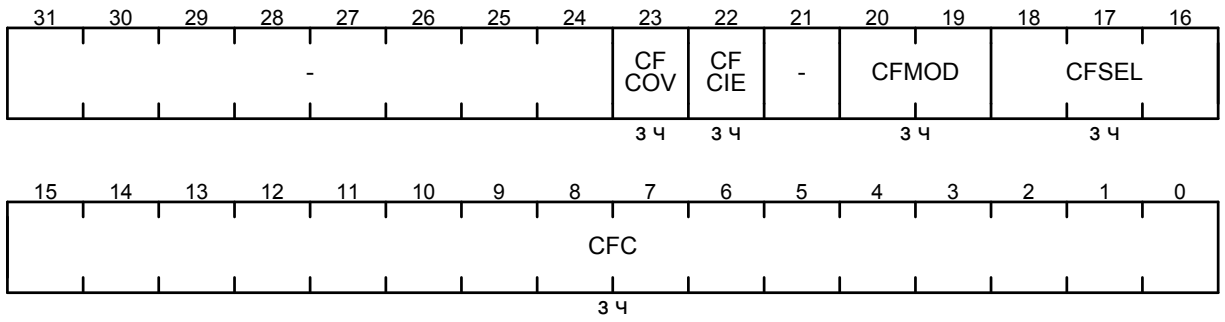
Смещение: Node_n + 14h
Сброс: 60_0000h



Поле	Биты	Описание	
LEINC	25	Индикатор инкрементирования при последней ошибке	
		0	Обнаруженная ошибка приводит к инкрементированию счетчика ошибок на единицу
		1	Обнаруженная ошибка приводит к инкрементированию счетчика ошибок на восемь
LETD	24	Флаг последней ошибки передачи	
		0	При приеме сообщения обнаружена ошибка, и произошло инкрементирование поля REC
		1	При передаче сообщения обнаружена ошибка, и произошло инкрементирование поля TEC
EWRNLVL	23-16	Поле задания лимита ошибок, по достижении которого выставляется флаг EWRN в регистре NSR (по умолчанию, количество ошибок – 96)	
TEC	15-8	Поле счетчика ошибок передачи сообщений	
REC	7-0	Поле счетчика ошибок приема сообщений	
–	31-26	Зарезервировано	

NFCR – регистр счетчика сообщений узла

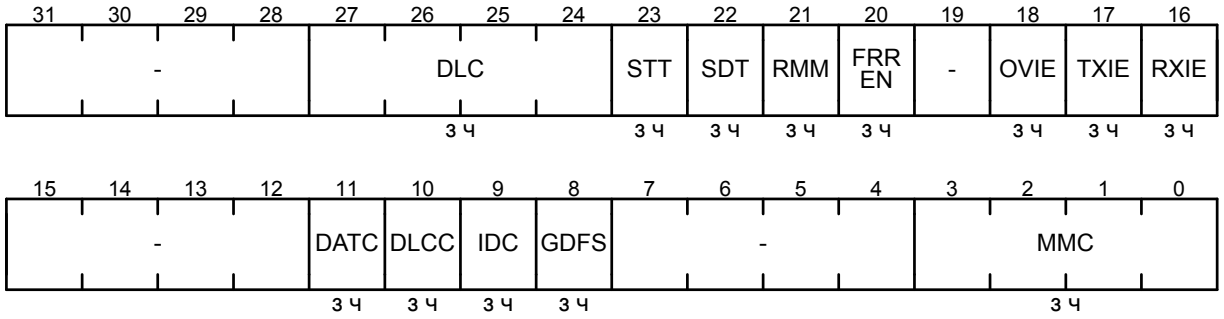
Смещение: Node_n + 18h
Сброс: 0h



Поле	Биты	Описание	
CFCOV	23	Флаг переполнения счетчика сообщений	
		0	Счетчик не переполнен
		1	Счетчик переполнился. В режиме синхросчетчика этот флаг устанавливается при изменении поля CFC, и, если установлен бит CFCIE, формируется прерывание
		Бит сбрасывается программно	
CFCIE	22	Бит разрешения прерывания от счетчика сообщений	
		0	Запрещено
		1	Разрешено
CFMOD	20-19	Режим работы счетчика сообщений	
		00	Счетчик сообщений инкрементируется после каждого успешного приема/передачи сообщения
		01-11	Зарезервировано. Не использовать!
CFSEL	18-16	Поле задания условия инкрементирования счетчика сообщений	
		**1	При получении сообщения, не имеющего объекта сообщения
		1	При получении сообщения, имеющего соответствующий объект сообщения
		1**	При успешной отправке сообщения
		000	Зарезервировано. Не использовать!
		Состояние бита, отмеченного символом *, неважно. Условия могут комбинироваться между собой (например, 110b или 101b)**	
CFC	15-0	Поле счетчика сообщений. Хранит значение счетчика сообщений при CFMOD = 00b	
–	31-24, 21	Зарезервировано	

MOFCR – регистр управления функционированием объекта сообщения

Смещение: Msg_mo + 00h
Сброс: 0h



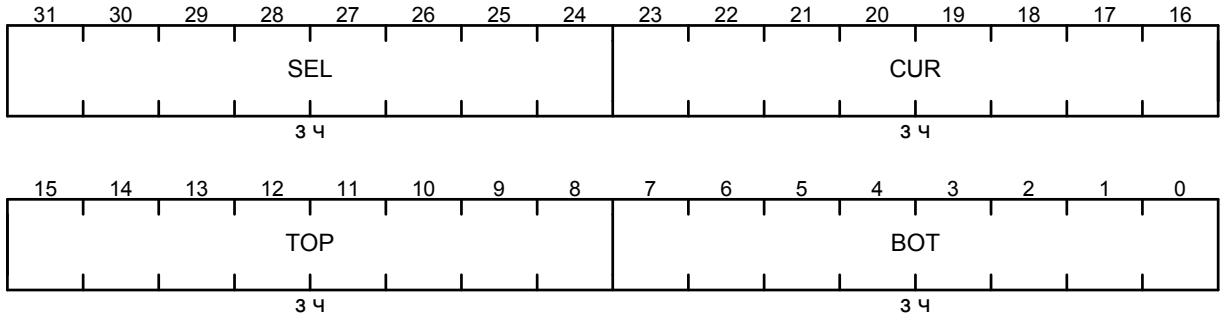
Поле	Биты	Описание	
DLC	27–24	Код длины данных. Показывает количество байт данных, находящихся в объекте сообщения. Диапазон – значение от 0 до 8. Если значение поля DLC больше 8, это автоматически указывает на 8 байт. Значение поля DLC полученного сообщения сохраняется таким, каким было получено	
STT	23	Бит задания однократной пересылки данных	
		0	Нет действий
		1	Если бит установлен, тогда бит TXRQ сбрасывается после начала передачи объекта сообщения X. В связи с этим, в случае неудачной передачи, повторной передачи сообщения не будет
SDT	22	Бит задания однократного участия объекта сообщения m в пересылке	
		0	Нет действий
		1	Если бит установлен, и объект сообщения m не является объектом FIFO, тогда бит MSGVAL сбрасывается после успешного приема данных
RMM	21	Бит включения удаленного мониторинга объекта передачи	
		0	Выключено. Идентификатор, бит IDE и поле DLC объекта сообщения m остаются без изменений до получения корректного фрейма удаленного запроса
		1	Включено. Идентификатор, бит IDE и поле DLC корректного фрейма удаленного запроса копируются в объект передачи n в порядке получения битов фрейма удаленного запроса монитора
		Состояние бита оказывает влияние только на объекты передач	
FRREN	20	Бит разрешения удаленного запроса. Определяет, будет ли устанавливаться бит TXRQ в объекте сообщения m или в другом объекте сообщения, на который указывает CUR	
		0	Бит TXRQ объекта сообщения m устанавливается после получения корректного фрейма удаленного запроса
		1	Бит TXRQ другого объекта сообщения (на который указывает CUR) устанавливается после получения им корректного фрейма удаленного запроса

Поле	Биты	Описание	
OVIE	18	Бит разрешения прерывания по заполнению FIFO объекта сообщения m. Прерывание генерируется, когда указатель CUR (указатель на текущий объект) достигает значения SEL регистра MOFGPR	
		0	Запрещено
		1	Разрешено
		Если объект сообщения m является объектом приема FIFO, то поле TXINP (регистр MOIPR) указывает на одну из двух линий прерываний. Если объект сообщения m является объектом передачи FIFO, то поле RXINP (регистр MOIPR) указывает на одну из двух линий прерываний. Для всех других режимов объекта сообщения состояние бита OVIE не важно	
TXIE	17	Бит разрешения прерывания по окончании передачи сообщения	
		0	Запрещено
		1	Разрешено. Прерывание генерируется, если сообщение из объекта сообщения m было успешно передано. Поле TXINP (регистр MOIPR) указывает на одну из двух линий прерывания
RXIE	16	Бит разрешения прерывания по окончании приема сообщения	
		0	Запрещено
		1	Разрешено. Прерывание генерируется, если сообщение было успешно принято объектом сообщения m (напрямую или через шлюз). Поле RXINP (регистр MOIPR) указывает на одну из двух линий прерывания
DATC	11	Индикатор копирования данных	
		0	Данные не копируются
		1	Данные в регистрах MODATAH и MODATAL объекта-источника шлюза (после сохранения принятого фрейма в источнике) копируются через шлюз в объект-приемник
		Бит DATC используется только объектом-источником в режиме шлюза. Во всех остальных случаях бит не функционирует	
DLCC	10	Индикатор копирования кода длины данных DLC	
		0	Код не копируется
		1	Код длины данных объекта-источника шлюза (после сохранения принятого фрейма в источнике) копируется через шлюз в объект-приемник
		Бит DLCC используется только объектом-источником в режиме шлюза. Во всех остальных случаях бит не функционирует	
IDC	9	Индикатор копирования идентификатора	
		0	Идентификатор не копируется
		1	Идентификатор объекта-источника шлюза (после сохранения принятого фрейма в источнике) копируется через шлюз в объект-приемник
		Бит IDC используется только объектом-источником в режиме шлюза. Во всех остальных случаях бит не функционирует	
GDFS	8	Индикатор отправки фрейма через шлюз	
		0	Состояние бита TXRQ объекта-приемника без изменений
		1	Установлен бит TXRQ объекта-приемника после внутренней передачи из объекта-источника
		Бит GDFS используется только объектом-источником в режиме шлюза. Во всех остальных случаях бит не функционирует	

Поле	Биты	Описание	
MMC	3-0	Задание режима объекта сообщения m	
		0000	Стандартный объект сообщения
		0001	Базовый объект приемной структуры FIFO
		0010	Базовый объект передающей структуры FIFO
		0011	Вспомогательный объект передающей структуры FIFO
		0100	Объект-источник шлюза
		Остальные комбинации зарезервированы	
–	31-28, 19, 15-12, 7-4	Зарезервировано	

MOFGPR – регистр указателя FIFO или шлюза объекта сообщения

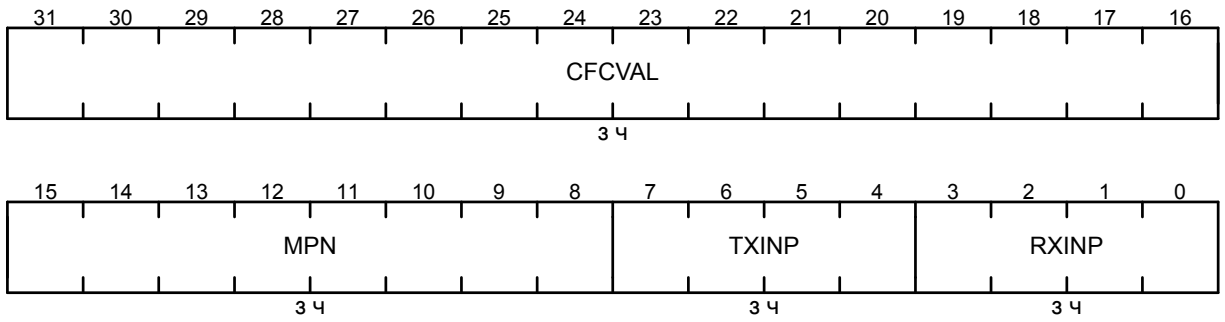
Смещение: Msg_mo + 04h
Сброс: 0h



Поле	Биты	Описание
SEL	31-24	Указатель объекта сообщения. Второй (программный) указатель в дополнение к аппаратному указателю CUR при работе с FIFO. Поле SEL используется для общего мониторинга (генерирование прерываний FIFO)
CUR	23-16	Указатель на текущий объект в пределах FIFO или шлюза. После каждой операции FIFO или передачи через шлюз указатель CUR обновляется – в него заносится номер следующего объекта сообщения в списке (поле PNEXT регистра MOSTAT) – до тех пор, пока не будет достигнут верхний элемент FIFO (поле TOP), после чего CUR сбрасывается, и в него загружается номер нижнего элемента списка (из поля BOT)
TOP	15-8	Указатель верхнего элемента FIFO. В поле находится номер последнего элемента
BOT	7-0	Указатель нижнего элемента FIFO. В поле находится номер первого элемента

MOIPR – регистр указателя прерываний объекта сообщения

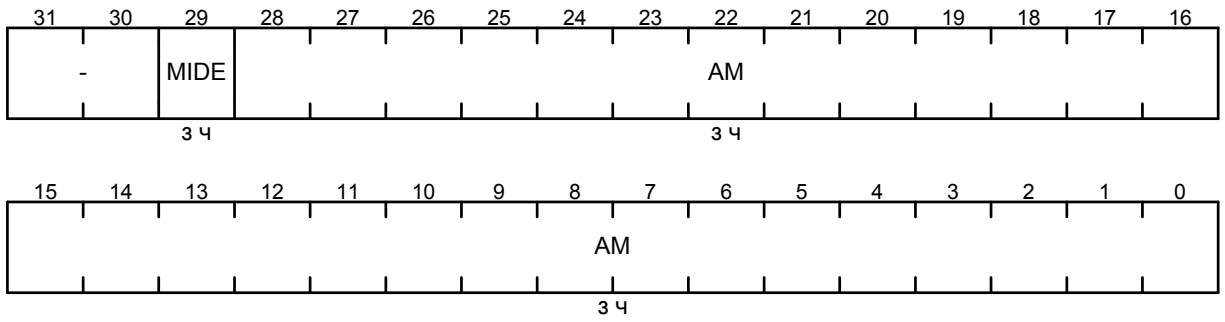
Смещение: Msg_mo + 08h
Сброс: 0h



Поле	Биты	Описание
CFCVAL	31-16	Количество фреймов. Каждый раз после записи принятого сообщения в объект сообщения m или успешной передачи объекта сообщения m, значение счетчика фреймов CFC (регистр NFCR) копируется в CFCVAL
MPN	15-8	Номер ждущего бита сообщения. Указывает позицию бита, соответствующего объекту сообщения m в регистре MSPND
TXINP	7-4	Указатель линии прерываний для прерывания после передачи. Всего доступно две линии прерывания с номерами от 0 до 1. Значение 0000b, записанное в TXINP, выбирает нулевую линию прерываний, 0001b – первую. Дополнительно бит TXINP используется для выбора позиции ждущего бита объекта сообщения m
RXINP	3-0	Указатель линии прерываний для прерывания после приема. Всего доступно две линии прерывания с номерами от 0 до 1. Значение 0000b, записанное в RXINP, выбирает нулевую линию прерываний, 0001b – первую. Дополнительно бит RXINP используется для выбора позиции ждущего бита объекта сообщения m

MOAMR – регистр маски объекта сообщения

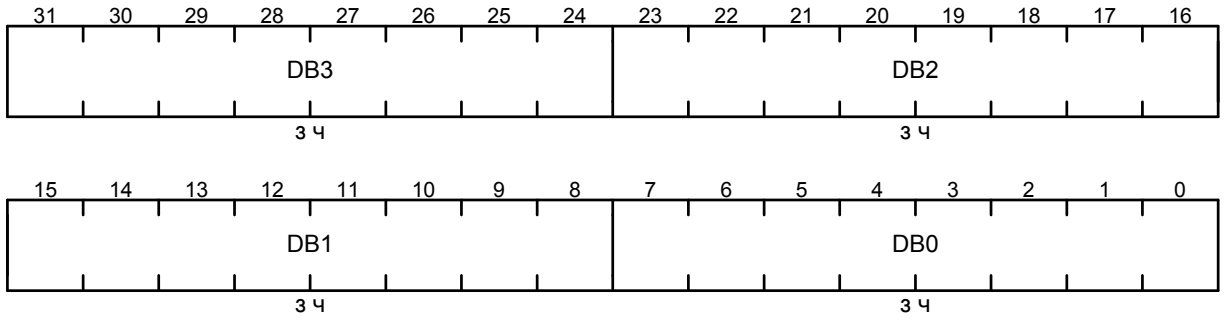
Смещение: Msg_mo + 0Ch
Сброс: 3FFF_FFFFh



Поле	Биты	Описание	
MIDE	29	Маска бита IDE сообщения	
		0	Объект сообщения m может принимать как стандартные, так и расширенные фреймы
		1	Объект сообщения m может принимать только те фреймы, у которых состояние бита IDE совпадает с его битом IDE
AM	28-0	Маска идентификатора. При приеме расширенного сообщения используется вся маска. При приеме стандартного сообщения используются биты 28-18, при этом состояние битов 17-0 не важно	
–	31-30	Зарезервировано	

MODATAL – младший регистр данных объекта сообщения

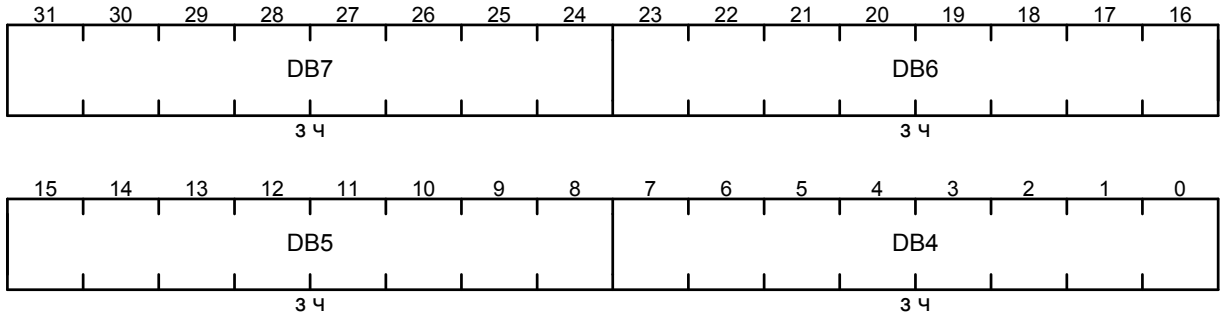
Смещение: Msg_mo + 10h
Сброс: 0h



Поле	Биты	Описание
DB3	31-24	Третий байт данных
DB2	23-16	Второй байт данных
DB1	15-8	Первый байт данных
DB0	7-0	Нулевой байт данных

MODATAH – старший регистр данных объекта сообщения

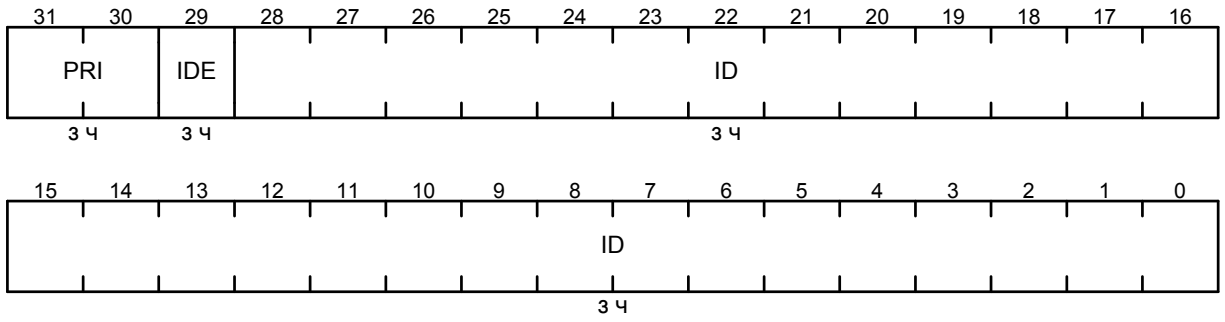
Смещение: Msg_mo + 14h
Сброс: 0h



Поле	Биты	Описание
DB7	31-24	Седьмой байт данных
DB6	23-16	Шестой байт данных
DB5	15-8	Пятый байт данных
DB4	7-0	Четвертый байт данных

MOAR – регистр арбитража объекта сообщения

Смещение: Msg_mo + 18h
Сброс: 0h



Поле	Биты	Описание	
PRI	31-30	Класс приоритета. Поле определяет один из четырех классов (0, 1, 2 и 3) приоритета объекта сообщения m. Нулевой класс устанавливает наивысший приоритет. Объекты сообщений с нулевым классом всегда выигрывают арбитраж при передаче и приеме сообщений. Фильтрация сообщений на основе идентификатора (маскируемого) и позиции в списке организуются только для объектов сообщений с равным приоритетом. Кроме этого, поле PRI определяет метод фильтрации	
		00	Зарезервировано
		01	Фильтрация в зависимости от положения объекта сообщения в списке. Объект сообщения m получает приоритет на передачу сообщения только в случае, если нет других объектов сообщений с установленными битами MSGVAL, TXEN0 и TXEN1, стоящих выше по списку

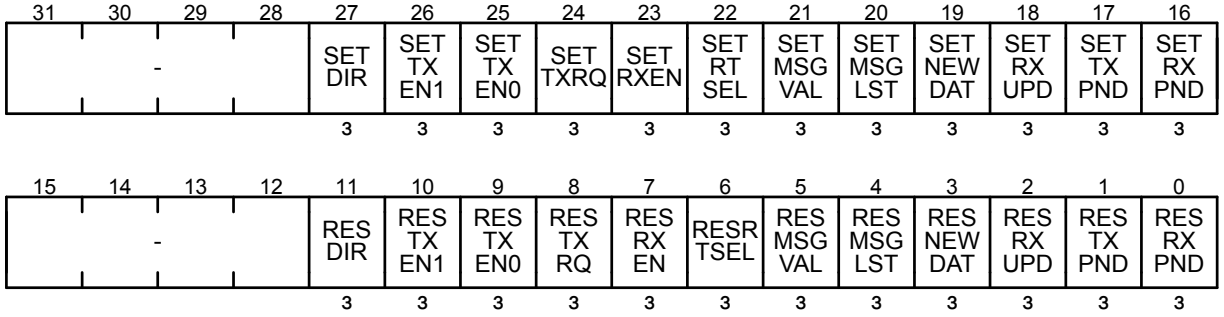
Поле	Биты	Описание	
		10	Фильтрация в зависимости от значения идентификатора. Объект сообщения m получает приоритет на передачу сообщения только в случае, если в списке нет других объектов сообщений с «Идентификатор + IDE + DIR» более высокого приоритета (согласно правилам арбитража в таблице А.22.2)
		11	Фильтрация в зависимости от положения объекта сообщения в списке (как при PRI = 01b)
IDE	29	Бит расширения идентификатора объекта сообщения m	
		0	Объект сообщения m оперирует с фреймами со стандартным 11-битным идентификатором
		1	Объект сообщения m оперирует с фреймами с расширенным 29-битным идентификатором
ID	28-0	Идентификатор объекта сообщения m. При оперировании с расширенными фреймами используются биты 28-0. При оперировании со стандартными фреймами используются биты 28-18, при этом состояние битов 17-0 не важно	

Таблица А.22.2 – Распределение приоритета между объектами сообщений согласно правилам арбитража

Установки для объектов сообщений 0 и 1, которые участвуют в арбитраже (приоритет объекта 0 выше приоритета объекта 1)	Пояснение
MOAR0[28:18] < MOAR1[28:18] (11-битный стандартный идентификатор объекта 0 меньше по числовому значению, чем 11-битный идентификатор объекта 1)	Стандартный фрейм с идентификатором, имеющим меньшее значение, обладает более высоким приоритетом
MOAR0[28:18] = MOAR1[28:18]. В регистре MOAR0 бит IDE = 0. В регистре MOAR1 бит IDE = 1.	При равенстве значений стандартных идентификаторов, стандартный фрейм имеет приоритет перед расширенным
MOAR0[28:18] = MOAR1[28:18]. Биты IDE обоих объектов сброшены. В регистре MOSTAT0 бит DIR = 1. В регистре MOSTAT1 бит DIR = 0.	При равенстве значений идентификаторов стандартный фрейм данных имеет приоритет перед стандартным фреймом удаленного запроса
MOAR0[28:0] = MOAR1[28:0] Биты IDE обоих объектов установлены. В регистре MOSTAT0 бит DIR = 1. В регистре MOSTAT1 бит DIR = 0.	При равенстве значений идентификаторов расширенный фрейм данных имеет приоритет перед расширенным фреймом удаленного запроса
MOAR0[28:0] < MOAR1[28:0] Биты IDE обоих объектов установлены. (29-битный идентификатор объекта 0 меньше по числовому значению, чем 29-битный идентификатор объекта 1)	Расширенный фрейм с идентификатором, имеющим меньшее значение, обладает более высоким приоритетом

MOCTR – регистр управления объектом сообщения

Смещение: Msg_mo + 1Ch
Сброс: 0h



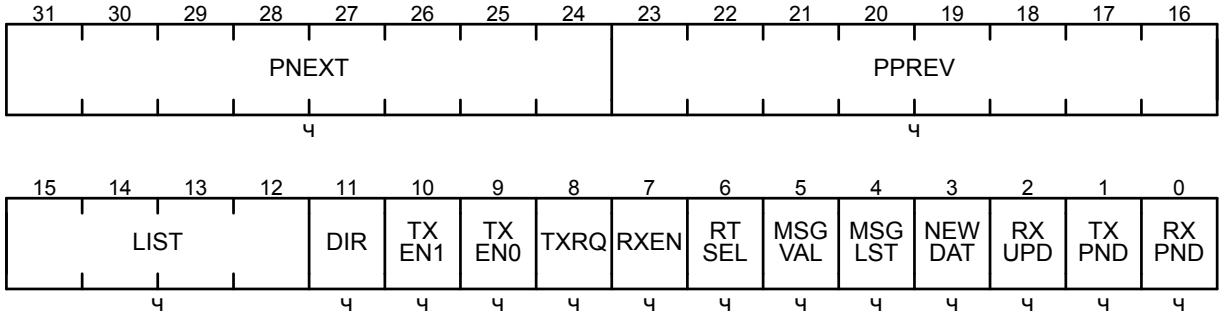
Поле	Биты	Описание
RESRXPND, SETRXPND	0, 16	Сброс/установка бита RXPND
RESTXPND, SETTXPND	1, 17	Сброс/установка бита TXPND
RESRXUPD, SETRXUPD	2, 18	Сброс/установка бита RXUPD
RESNEWDAT, SETNEWDAT	3, 19	Сброс/установка бита NEWDAT
RESMSGLST, SETMSGLST	4, 20	Сброс/установка бита MSGLST
RESMSGVAL, SETMSGVAL	5, 21	Сброс/установка бита MSGVAL
RESRTSEL, SETRTSEL	6, 22	Сброс/установка бита RTSEL
RESRXEN, SETRXEN	7, 23	Сброс/установка бита RXEN
RESTXRQ, SETTXRQ	8, 24	Сброс/установка бита TXRQ
RESTXEN0, SETTXEN0	9, 25	Сброс/установка бита TXEN0
RESTXEN1, SETTXEN1	10, 26	Сброс/установка бита TXEN1
RESDIR, SETDIR	11, 27	Сброс/установка бита DIR
–	15-12, 31-28	Зарезервировано. При чтении возвращаются нули. При записи следует писать 0h.

Примечание – Биты с префиксом SET и RES работают попарно. Комбинация состояний этих бит оказывает влияние на соответствующий бит регистра MOSTAT:

- SET* = 1, RES* = 0 устанавливает бит *;
- SET* = 0, RES* = 1 сбрасывает бит *;
- SET* = RES* = 0 или SET* = RES* = 1 не изменяет состояние бита *.

MOSTAT – регистр состояния объекта сообщения

Смещение: Msg_mo + 1Ch
Сброс:
- 0100_0000h для объекта сообщения 0;
- [mo + 1][mo - 1][0000]h для объектов сообщений с номерами от 1 до 126, (соответственно mo от 01h до 7Eh);
- FFFE_0000h для объекта сообщения 127.



Поле	Биты	Описание	
PNEXT	31-24	Указатель на следующий элемент списка. В поле находится номер объекта сообщения, расположенного выше по списку относительно текущего	
PPREV	23-16	Указатель на предыдущий элемент списка. В поле находится номер объекта сообщения, расположенного ниже по списку относительно текущего	
LIST	15-12	Номер списка, которому принадлежит объект сообщения m. Поле обновляется аппаратно при распределении/перераспределении объекта сообщения	
DIR	11	Бит распределения	
		0	Объект приема сообщения данных. Объект принимает сообщение данных. При установленном бите TXRQ объект формирует сообщение удаленного запроса с идентификатором объекта сообщения m, а затем передает его. Полученное в ответ сообщение данных с соответствующим идентификатором сохраняется в объекте сообщения m
		1	Объект передачи сообщения данных. При установленном бите TXRQ объект формирует, а затем передает сообщение данных. Если объект сообщения m получает сообщение удаленного запроса с соответствующим идентификатором, то устанавливается флаг TXRQ его регистра MOSTAT, после чего в ответ передается сообщение данных, содержащихся в объекте сообщения m
TXEN1	10	Бит разрешения передачи фрейма	
		0	Запрещено
		1	Передача фрейма разрешена. Объект сообщения m может участвовать в передаче только, если установлены оба бита – TXEN1 и TXEN0. Контроллер CAN использует бит TXEN1 для выбора активного объекта передачи сообщения из FIFO

Поле	Биты	Описание	
TXEN0	9	Бит разрешения передачи фрейма	
		0	Запрещено
		1	Передача фрейма разрешена. Объект сообщения m может участвовать в передаче, только если установлены оба бита TXEN0 и TXEN1. Контроллер CAN использует бит TXEN1 для выбора активного объекта передачи сообщения из FIFO. Можно программно очищать бит TXEN0 для запрета передачи сообщения, которое в настоящий момент формируется, или для запрета автоматической передачи в ответ на удаленный запрос
TXRQ	8	Бит инициации передачи	
		0	Нет действий
		1	Установка бита инициирует передачу фрейма из объекта сообщения m. Инициация передачи фрейма возможна только в случае, если установлены биты TXRQ, TXEN0, TXEN1 и MSGVAL. Также бит TXRQ устанавливается аппаратно при получении фрейма удаленного запроса. Бит сбрасывается аппаратно при успешном завершении передачи, и если при этом не был повторно программно установлен бит NEWDAT
RXEN	7	Бит разрешения приема	
		0	Запрещено
		1	Объект сообщения может принимать сообщения
		Состояние бита учитывается только при фильтрации принимаемых сообщений	
RTSEL	6	Индикатор возможности приема/передачи	
		0	Объект сообщения не может принимать/передавать сообщения
		1	Объект сообщения может принимать/передавать сообщения
		Прием фрейма. Бит RTSEL устанавливается аппаратно после того, как выбран объект сообщения m для сохранения только что принятого фрейма. Прежде, чем записать принятые данные в объект сообщения m, аппаратная часть проверяет состояние бита RTSEL. ЦПУ может сбрасывать этот бит, чтобы запретить запись принятого фрейма в объект сообщения m. Передача фрейма. Бит RTSEL устанавливается аппаратно после того, как выбран следующий объект сообщения m для передачи фрейма. Аппаратная часть перед началом передачи проверяет: установлен ли бит RTSEL и сброшен ли бит NEWDAT. Бит RTSEL должен оставаться установленным до окончания передачи. Проверка состояния бита RTSEL производится только при попытке изменения содержимого объекта сообщения m во избежание одновременного выполнения операций передачи фрейма и его изменения. Бит не участвует в фильтрации сообщений и не сбрасывается аппаратно	
MSGVAL	5	Бит активности объекта сообщения m	
		0	Не активен
		1	Активен
		Только те объекты сообщений, для которых установлен этот бит, могут использоваться для операций приема и передачи	

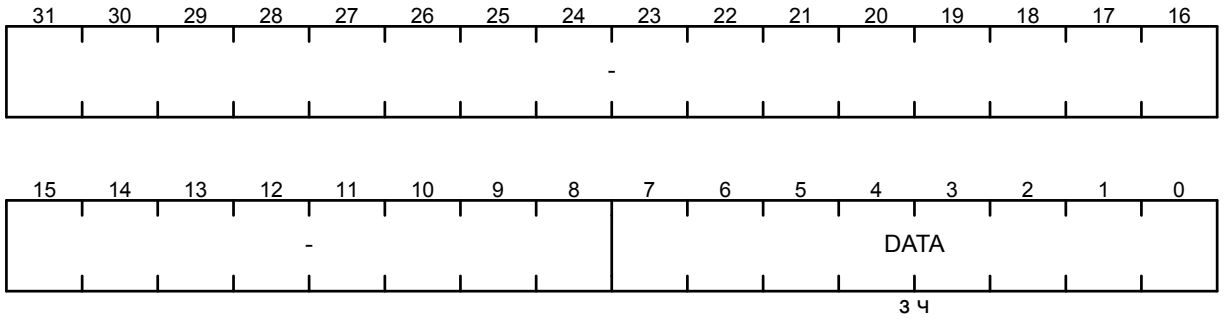
Поле	Биты	Описание	
MSGLST	4	Бит потери сообщения	
		0	Ни одно сообщение не потеряно
		1	Принятое сообщение потеряно вследствие того, что контроллер CAN попытался установить бит NEWDAT по окончании приема сообщения при том, что флаг NEWDAT уже был установлен ранее после записи другого сообщения
NEWDAT	3	Индикатор новых данных	
		0	С момента сброса бита NEWDAT никаких изменений объекта сообщения m не обнаружено
		1	Объект сообщения был изменен. Бит устанавливается аппаратно после того, как принятое сообщение было сохранено в объекте сообщения m. Бит сбрасывается аппаратно после начала передачи объекта сообщения m. Бит NEWDAT следует устанавливать программно после того, как новые данные для передачи будут сохранены в объекте сообщения m для предотвращения автоматического сброса бита TRXQ в конце текущей передачи
RXUPD	2	Индикатор изменений	
		0	Нет текущих изменений
		1	Идентификатор сообщения, поле длины данных DLC и данные в объекте сообщения изменяются
TXPND	1	Индикатор окончания передачи	
		0	Переданных сообщений нет
		1	Сообщение объекта m было успешно передано
RXPND	0	Индикатор окончания приема	
		0	Принятых сообщений нет
		1	Сообщение было успешно принято объектом сообщения m (напрямую или через шлюз). Бит должен сбрасываться программно

A.20 Регистры контроллера интерфейса I2C

Базовый адрес: 3000_5000h Регистры блока I2C

SDA – регистр данных

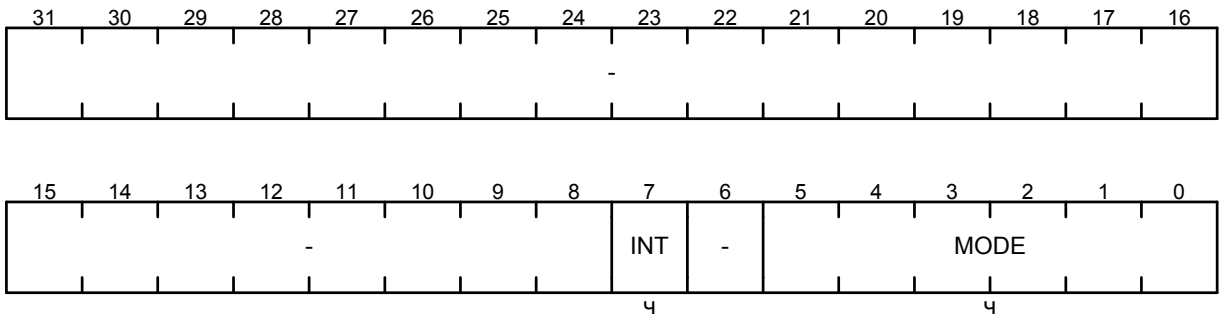
Смещение: + 00h
Сброс: 0xxh



Поле	Биты	Описание
DATA	7-0	Поле данных
–	31-8	Зарезервировано

ST – регистр состояния

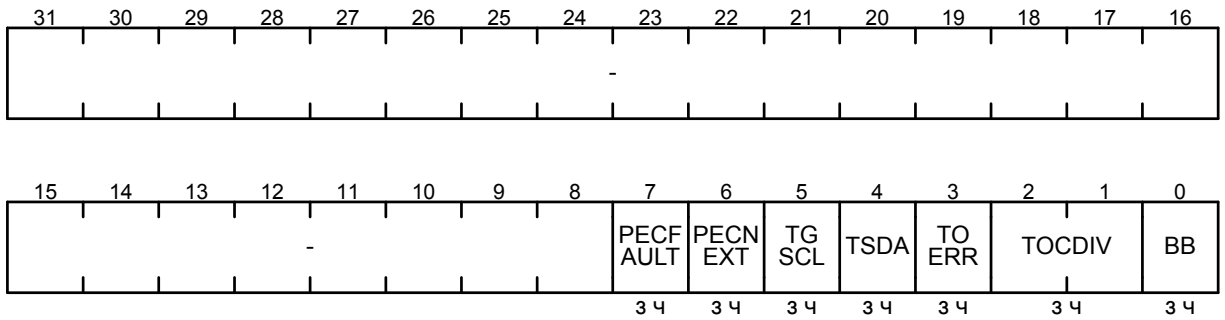
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание
INT	7	Флаг прерывания. Устанавливается после девятого такта сигнала SCL (когда SCL = 0) в любое запрограммированное время. Условия выставления флага INT: - во время приема/передачи, как в режиме мастера, так и в режиме ведомого; - при совпадении адреса (адреса ведомого, адреса отклика или адреса общего вызова) содержимое регистра SDA должно контролироваться программно для определения типа полученного адреса; - после успешного формирования стартового состояния или состояния повторного старта; - в случае неквитирования переданной информации; - при потере арбитража во время передачи последнего бита; - при обнаружении валидного состояния останова или состояния повторного старта; - при обнаружении ошибки на шине. Пока установлен флаг INT, на линии SCL удерживается низкий уровень сигнала. Флаг INT может быть сброшен установкой бита CLRST в регистре CTL0 или выключением модуля I2C (обнуление бита ENABLE в регистре CTL1). Условия выставления флага INT (не влияющие на уровень сигнала на линии SCL): - простой на линии SCL; - состояние останова в режиме ведомого (MODE = 1Ch); - потеря арбитража, вследствие чего ведомый переключился в безадресный режим (MODE = 03h или MODE = 23h); - неквитированная передача байта данных (MODE = 17h)
MODE	5-0	Код состояния. Возникновение того или иного состояния в течение функционирования модуля I2C сопровождается записью соответствующего кода в поле MODE
–	31-8, 6	Зарезервировано

CST – регистр управления и статуса

Смещение: + 08h
Сброс: 0h

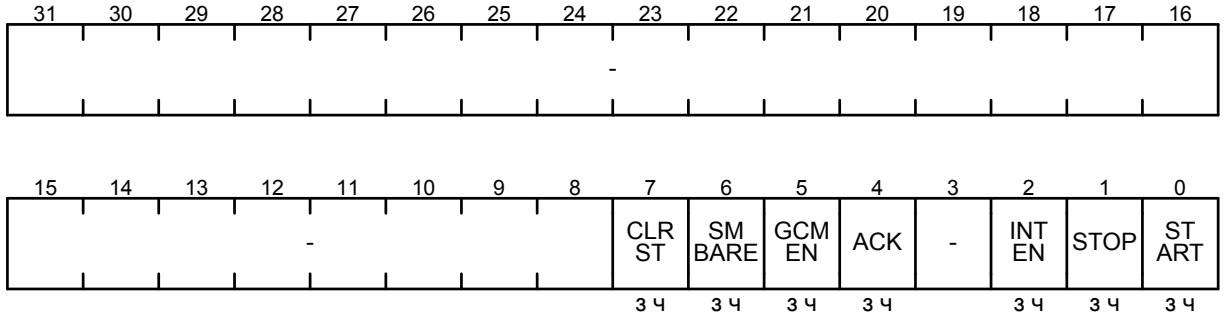


Поле	Биты	Описание
PECFAULT	7	Флаг ошибки. Устанавливается в случае, если после расчета контрольной суммы для пакета данных и сравнения ее с полученной суммой, значение во внутреннем регистре ошибок не нулевое
PECNEXT	6	Бит управления отправкой байта контрольной суммы. Установка бита указывает на то, что следующий передаваемый байт будет байтом CRC (байт контрольной суммы). Реакция на установку бита PECNEXT зависит от режима работы. В режиме мастера передатчика установка бита PECNEXT вызовет загрузку результата вычисления CRC в регистр SDA. После сброса флага INT начнется передача байта CRC. В режиме приемника установка этого бита будет указывать логике управления на то, что следующий байт, который будет принят, будет байтом CRC. В режиме ведомого приемника модуль I2C автоматически будет квитировать или не квитировать прием байта CRC, в зависимости от того, будет ли выявлена ошибка пакета данных или нет. В режиме мастера приемника по окончании приема байта CRC будет отправлено значение бита ACK регистра CTL0
TGSCL	5	Бит переключения SCL. Бит позволяет переключать вывод I2C_SCL во время восстановления после ошибки. Когда на выводе I2C_SDA – низкий уровень сигнала, запись «1» в бит TGSCL переключит вывод SCL на один такт. Когда на выводе I2C_SDA высокий уровень сигнала, запись «1» в бит TGSCL игнорируется. Бит очищается аппаратно по окончании такта
TSDA	4	Бит тестирования SDA. Содержит текущее значение SDA. Этот бит можно использовать для отслеживания окончания процесса восстановления после ошибки, в течение которого ведомый постоянно поддерживает низкий уровень сигнала на выводе SDA
TOERR	3	Флаг ошибки простоя на шине. Если TOERR = 1b, это указывает на то, что на линии SCL был обнаружен простой. Флаг TOERR выставляется по обнулению основного счетчика времени простоя и может быть сброшен

		записью «1» в бит CLRST регистра CTL0	
Поле	Биты	Описание	
TOCDIV	2-1	Поле коэффициента делителя. Устанавливает коэффициент деления системного тактового сигнала, подаваемого на предделитель времени простоя линии SCL	
		00	Тактовый сигнал отсутствует
		01	Деление на 4
		10	Деление на 8
		11	Деление на 16
BB	0	Флаг занятости шины. Если BB = 1b, это указывает на то, что шина занята. Устанавливается, как только шина переходит в активное состояние (одновременное появление низкого уровня сигнала на выводах I2C_SDA и I2C_SCL или хотя бы на одном из них) или в стартовое состояние. Сбрасывается при выключении интерфейса I2C либо при обнаружении состояния останова	
–	31-8	Зарезервировано	

CTL0 – регистр управления 0

Смещение: + 0Ch
Сброс: 0h



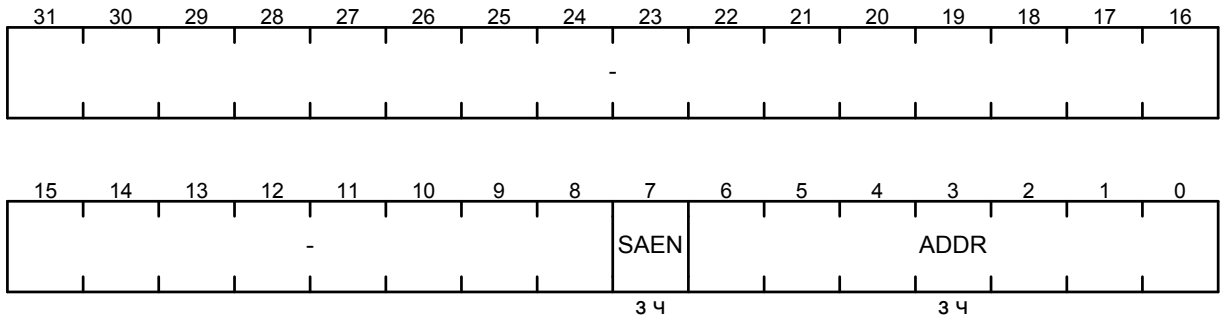
Поле	Биты	Описание	
CLRST	7	Бит сброса флага прерывания INT	
		Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Сбрасывает флаг INT в регистре ST
SMBARE	6	Бит управления реакцией на получение адреса отклика	
		0	Полученный адрес не проверяется на совпадение с адресом отклика
		1	Адрес, полученный сразу после старта, проверяется на совпадение с адресом отклика (0001100b)
		Бит очищается при выходе ведомого из режима IDLE	

GCMEN	5	Бит управления реакцией на получение адреса общего вызова	
		0	Полученный адрес не проверяется на совпадение с адресом общего вызова
		1	Адрес, полученный сразу после старта, проверяется на совпадение с адресом общего вызова (000_0000b)
		Бит очищается при выходе ведомого из режима IDLE	

Поле	Биты	Описание	
ACK	4	Бит квитирования приема. В режиме передатчика не используется. В режиме приемника (мастера/ведомого) содержит значение, которое передается в течение цикла отклика на запрос передатчика подтвердить прием. Передача нуля по окончании передачи байта (квитирование) означает, что данные успешно получены. Передача единицы (неквитирование) означает, что приемник не может продолжать работу по каким-либо причинам. Бит ACK очищается аппаратно по окончании цикла отклика	
INTEN	2	Бит разрешения прерывания	
		0	Запрещено
		1	Разрешено
STOP	1	Бит останова. В режиме мастера установка бита STOP генерирует состояние останова, которое завершает или прерывает текущую передачу. После прекращения передачи бит STOP очищается аппаратно	
START	0	Бит старта. Этот бит устанавливается, когда требуется сформировать стартовое состояние на шине. Бит START очищается аппаратно по окончании цикла стартового состояния, а также при обнаружении ошибки на шине (состояние с кодом 1Fh)	
–	31-8, 3	Зарезервировано	

ADDR – регистр собственного адреса

Смещение: + 10h
Сброс: 0h

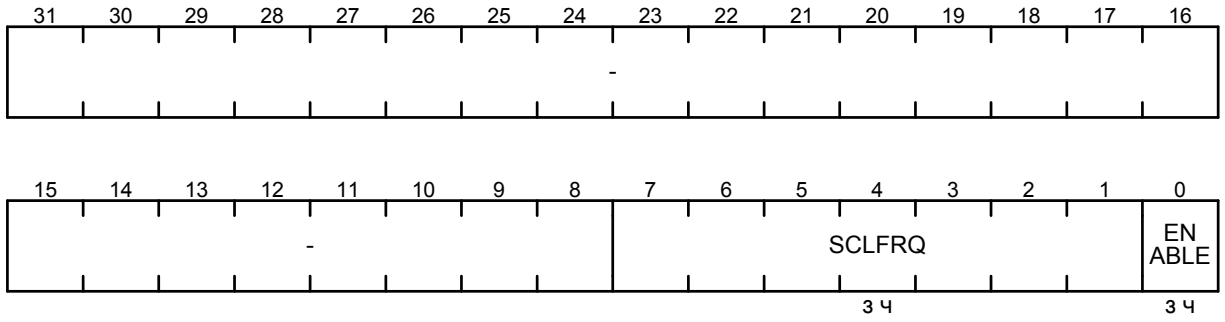


Поле	Биты	Описание	
SAEN	7	Бит разрешения распознавания адреса	
		0	Безадресный режим

		1	Включена функция распознавания принятого адреса
ADDR	6–0	Поле собственного 7-битного адреса. При работе в режиме ведомого первые 7 бит, принятые после стартового состояния, сравниваются со значением ADDR. Если обнаружено совпадение и установлен бит SAEN, ведомый переходит в режим приемника или передатчика (в зависимости от состояния бита направления R/W#)	
–	31-8	Зарезервировано	

CTL1 – регистр управления 1

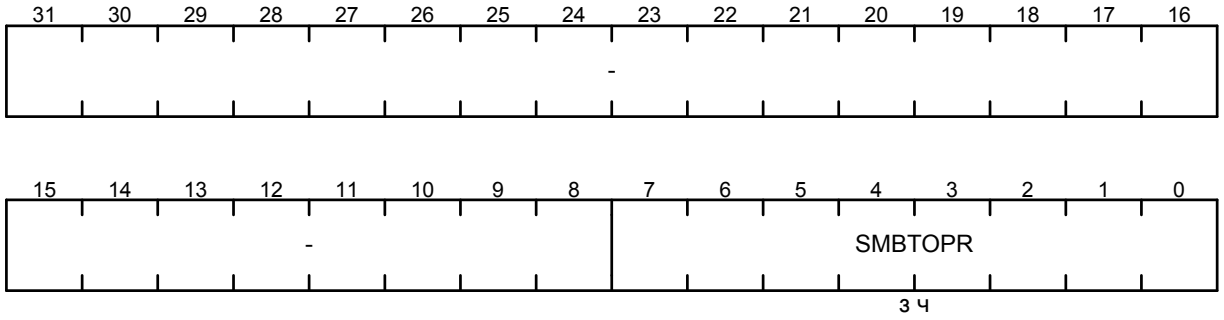
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание
SCLFRQ	7-1	Младшие разряды поля выбора частоты f_{SCL} сигнала на выводе I2Cx_SCL в режиме мастера. Длительности высокого T_{SCLH} и низкого T_{SCLL} уровней сигнала SCL зависят от тактовой частоты f_{PCLK} модуля I2C и рассчитываются по формуле $T_{SCLH} = T_{SCLL} = 2 \times SCLFRQ \times (1/f_{PCLK})$. (A.25.1) Таким образом, частота сигнала на выводе I2Cx_SCL равна $f_{SCL} = 1/(T_{SCLH} + T_{SCLL})$. (A.25.2) В поле SCLFRQ можно записать любое значение в диапазоне от 4h до 7FFFh (старшие разряды находятся в регистре CLT3). При попытке записи любого значения меньше 4h, оно будет записано со смещением 4h. Например, при записи числа 2h, к нему будет аппаратно добавлено смещение 4h и, в итоге, в поле SCLFRQ окажется значение 6h
ENABLE	0	Бит включения модуля I2C
		0 – Модуль выключен. Тактирование не осуществляется. Регистры CTL0, ST, CST сброшены
		1 – Модуль включен
–	31-8	Зарезервировано

TOPR – регистр загрузки предделителя

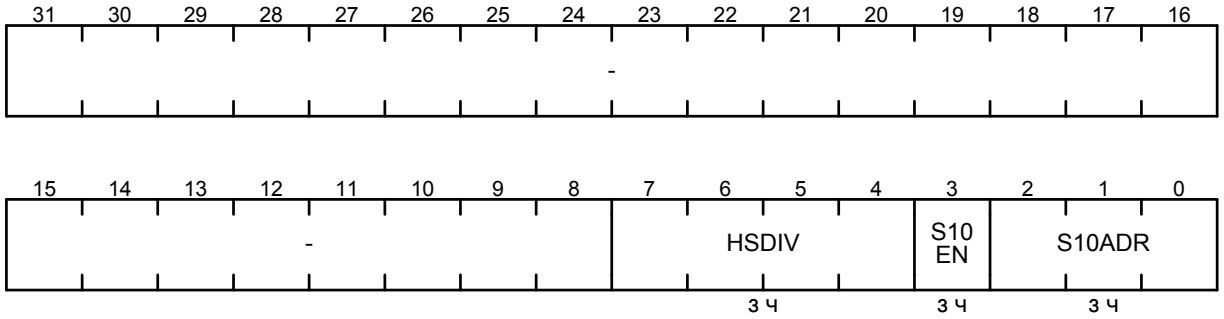
Смещение: + 18h
Сброс: 0h



Поле	Биты	Описание
SMBTOPR	7-0	Поле значения перезагрузки предделителя
–	31-8	Зарезервировано

CTL2 – регистр управления 2

Смещение: + 1Ch
Сброс: 0h

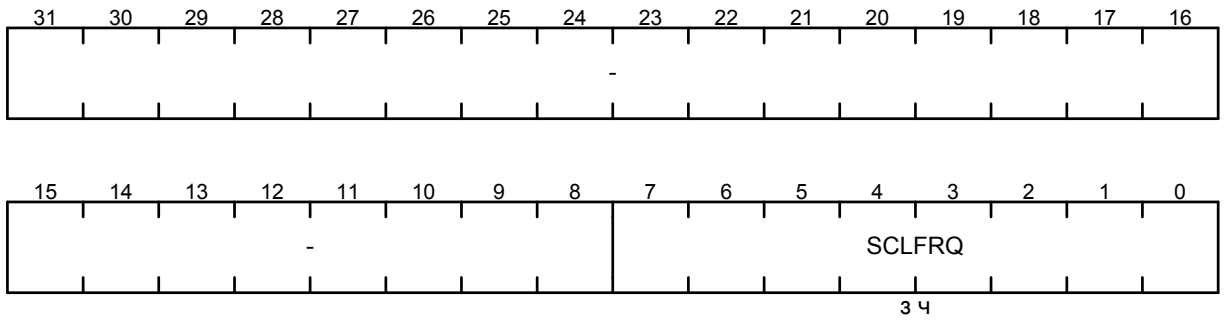


Поле	Биты	Описание	
HSDIV	7-4	Младшие разряды поля выбора частоты f_{SCL} сигнала на выводе I2C_SCL в режиме HS мастера. Длительности высокого(T_{HSCLH}) и низкого(T_{HSCLL}) уровней сигнала на выводе I2C_SCL зависят от тактовой частоты f_{PCLK} модуля I2C и рассчитываются по формулам $T_{HSCLH} = HSDIV \times (1/ f_{PCLK})$, (A.25.3) $T_{HSCLL} = 2 \times HSDIV \times (1/ f_{PCLK})$. (A.25.4) Таким образом, частота сигнала на выводе I2C_SCL равна $f_{SCL} = 1/(T_{HSCLH} + T_{HSCLL})$. (A.25.5) В поле HSDIV можно записать любое значение в диапазоне от 2h до 1000h (старшие разряды находятся в регистре CTL4). При попытке записи любого значения меньше 2 в поле HSDIV, оно будет записано со смещением 2. Например, при записи числа 1 к нему будет аппаратно добавлено смещение 2 и, в итоге, в поле HSDIV окажется значение 3	
S10EN	3	Бит разрешения 10-битной адресации ведомого	
		0	Запрещено
		1	Разрешено при условии, что установлен бит SAEN в регистре

		ADDR
S10ADR	2-0	Поле старших битов 10-битного адреса ведомого. Поле содержит старшие три разряда адреса ведомого при 10-битной адресации. Первый принятый байт адреса сравнивается со значением [11110b, S10ADR[2:1]], второй байт адреса – со значением [S10ADR[0], ADDR]
–	31-8	Зарезервировано

CTL3 – регистр управления 3

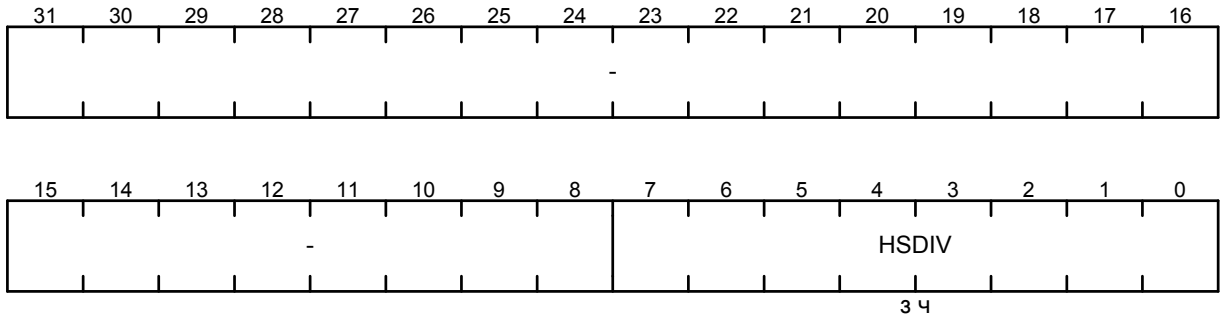
Смещение: + 20h
Сброс: 0h



Поле	Биты	Описание
SCLFRQ	7-0	Старшие разряды делителя SCLFRQ. См. основное описание поля в регистре CTL1
–	31-8	Зарезервировано

CTL4 – регистр управления 4

Смещение: + 24h
Сброс: 0h



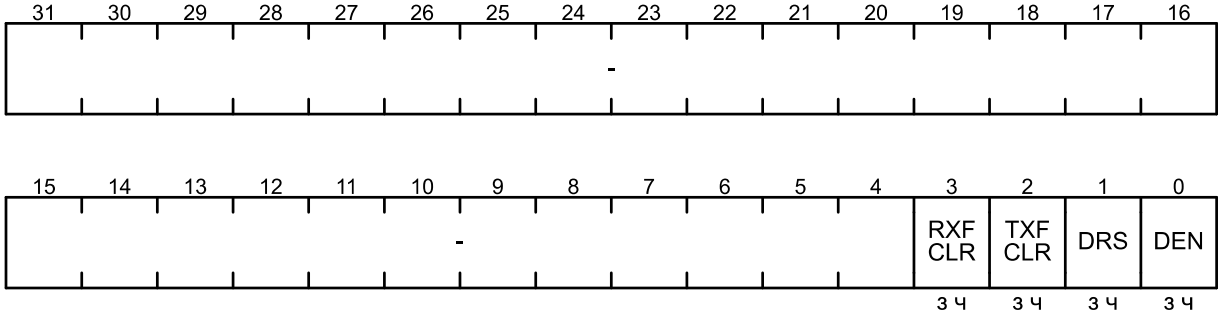
Поле	Биты	Описание
HSDIV	7-0	Старшие разряды делителя HSDIV. См. основное описание поля в регистре CTL2
–	31-8	Зарезервировано

A.21 Регистры контроллера интерфейса QSPI

Базовый адрес: 2004_0000h Регистры контроллера QSPI

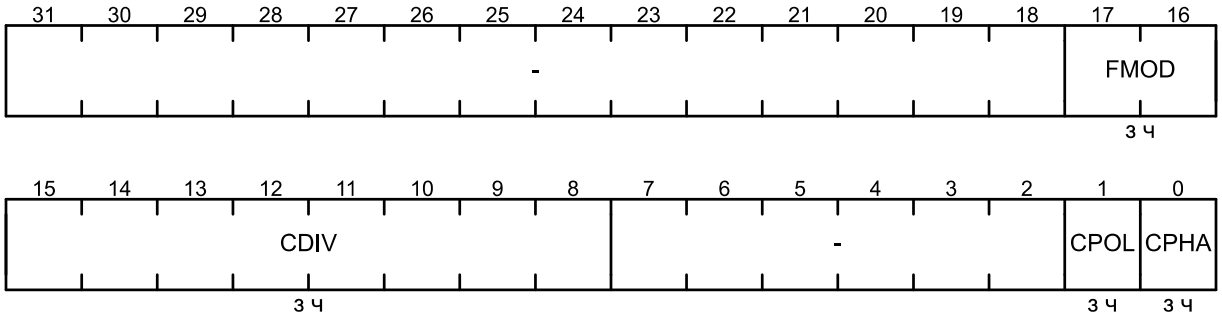
HCR – регистр управления

Смещение: + 00h
Сброс: 00h



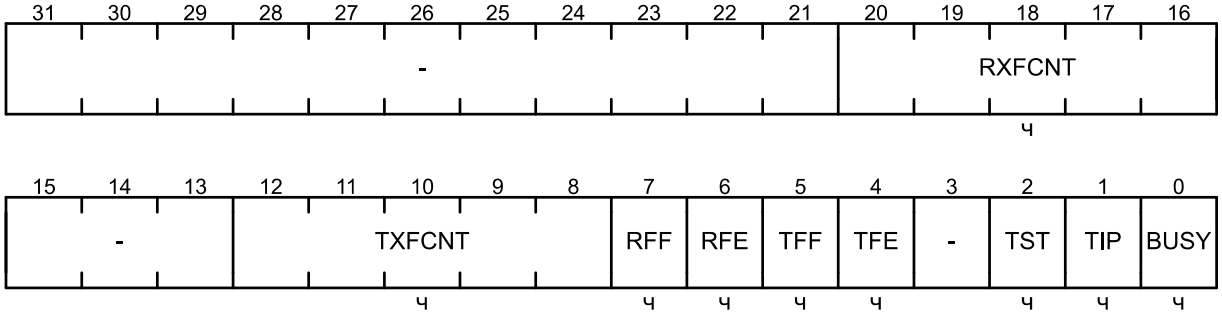
Поле	Биты	Описание
RXCLR	3	Бит очистки приемного буфера FIFO
TXCLR	2	Бит очистки передающего буфера FIFO
DRS	1	Бит сброса устройства (останова активной передачи)
DEN	0	Бит разрешения работы
–	31-4	Зарезервировано

DCR – регистр конфигурации устройства
Смещение: + 04h
Сброс: 0000_0100h



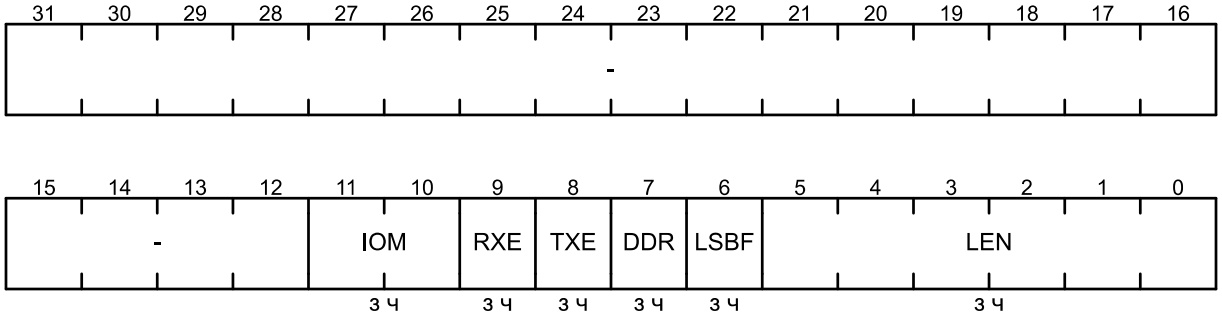
Поле	Биты	Описание	
FMOD	17-16	Выбор режима работы	
		0	Стандартный режим SPI
		1	Зарезервировано
		2	Интерфейс памяти QSPI
		3	Зарезервировано
CDIV	15-8	Значение делителя	
CPOL	1	Полярность сигнала QSPI_SCK	
CPHA	0	Фаза сигнала QSPI_SCK	
–	31-19, 7-2	Зарезервировано	

DSR – регистр статуса
Смещение: + 08h
Сброс: 0000_0050h



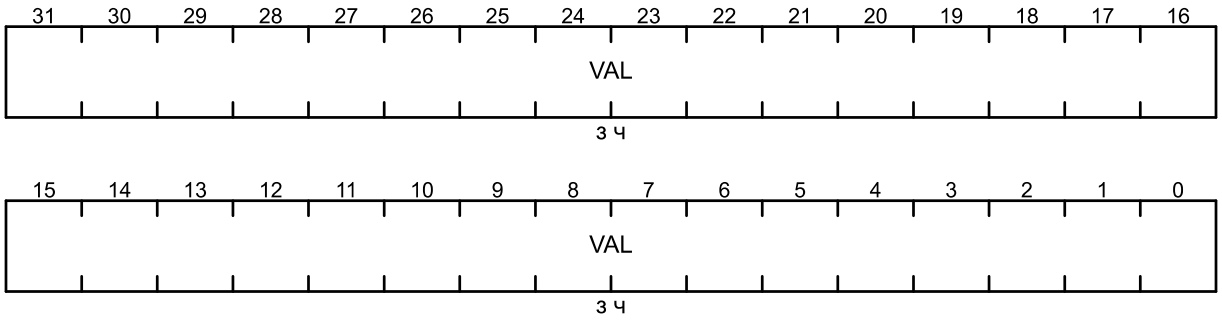
Поле	Биты	Описание	
RXFCNT	20-16	Количество элементов в буфере FIFO приемника	
TXFCNT	12-8	Количество элементов в буфере FIFO передатчика	
RFF	7	Буфер FIFO приемника заполнен	
RFE	6	Буфер FIFO приемника пуст	
TFF	5	Буфер FIFO передатчика заполнен	
TFE	4	Буфер FIFO передатчика пуст	
TST	2	Передача приостановлена	
TIP	1	Передача в процессе	
BUSY	0	0	Приемопередатчик не активен
		1	Передача/прием данных
–	31-21, 15-13, 3	Зарезервировано	

TCR – регистр конфигурации передачи
Смещение: + 0Ch
Сброс: 00h



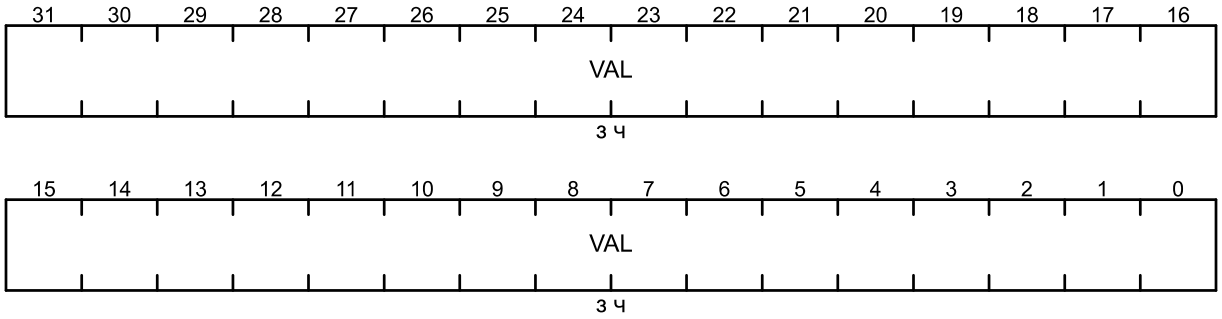
Поле	Биты	Описание	
IOM	11-10	Выбор количества линий данных	
		0	Одна линия передачи (Single)
		1	
		2	Две линии передачи (Double)
		3	Четыре линии передачи (Quad)
RXE	9	Бит разрешения приема	
TXE	8	Бит разрешения передачи	
DDR	7	0	Обычная скорость передачи данных (изменение данных по одному фронту QSPI_SCK)
		1	Двойная скорость передачи данных (изменение данных по обоим фронтам QSPI_SCK)
LSBF	6	0	Передача старшим битом вперед
		1	Передача младшим битом вперед
LEN	5-0	Размер слова данных (количество бит). Значение от 1 до 32	
–	31-12	Зарезервировано	

TDR – регистр передачи данных
Смещение: + 10h
Сброс: 00h



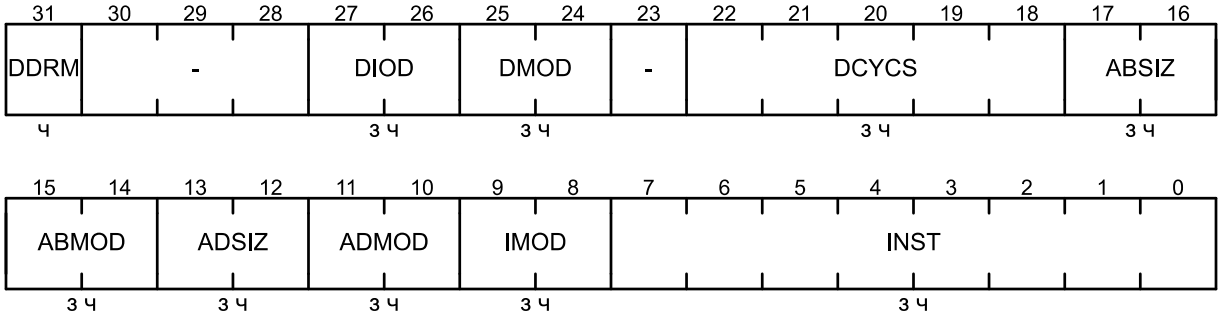
Поле	Биты	Описание
VAL	31-0	32-разрядный буфер FIFO приемника и передатчика. Данные для передачи записываются в регистр. Принятые данные возвращаются при чтении регистра.

TDS – регистр количества слов данных
Смещение: + 14h
Сброс: 00h



Поле	Биты	Описание
VAL	31-0	Количество слов данных для приема/передачи

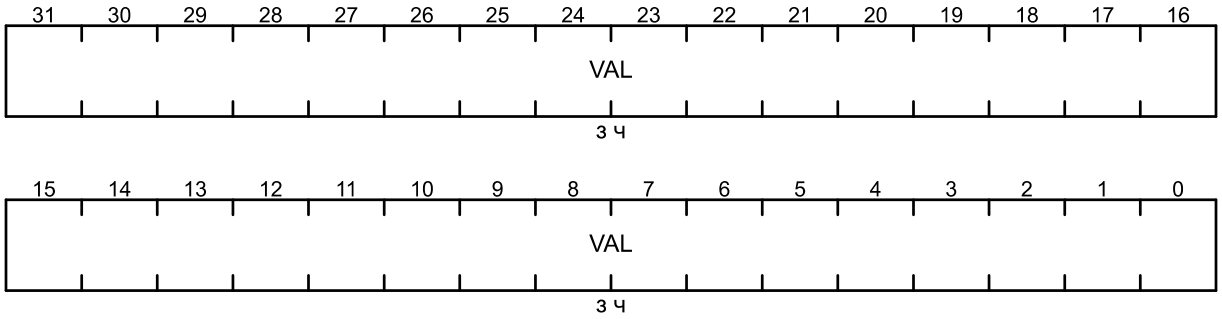
QCC – регистр конфигурации обмена QSPI
Смещение: + 18h
Сброс: 00h



Поле	Биты	Описание	
DDRM	31	Выбор режима передачи по обоим фронтам. Бит оказывает влияние в режиме QSPI только для адреса, дополнительных данных и данных.	
		0	Обычная скорость передачи данных (изменение данных по одному фронту QSPI_SCK)
		1	Двойная скорость передачи данных (изменение данных по обоим фронтам QSPI_SCK)
DIOD	27-26	Выбор направления передачи данных	
		0	Передача данных во внешнее устройство
		1	Прием данных из внешнего устройства
		2	Зарезервировано
		3	Зарезервировано
DMOD	25-24	Выбор режима данных	
		0	Отсутствует передача данных
		1	Передача данных по одной линии данных (SIO)
		2	Передача данных по двум линиям данных (DIO)
		3	Передача данных по четырем линиям данных (QIO)
DCYCS	22-18	Количество циклов ожидания (число периодов QSPI_CLK). Значения от 0 до 31	
ABSIZ	17-16	Выбор размера дополнительных данных	
		0	8 бит
		1	16 бит
		2	24 бита
		3	32 бита
ABMOD	15-14	Выбор режима передачи дополнительных данных	
		0	Отсутствует передача дополнительных данных
		1	Передача дополнительных данных по одной линии данных (SIO)
		2	Передача дополнительных данных по двум линиям данных (DIO)
		3	Передача дополнительных данных по четырем линиям данных (QIO)
ADSIZ	13-12	Выбор размера адреса	
		0	8-битный адрес
		1	16-битный адрес
		2	24-битный адрес
		3	32-битный адрес

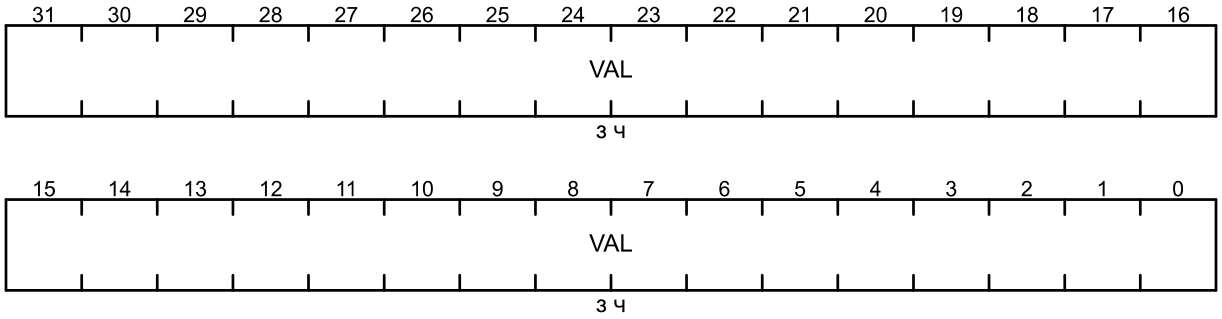
ADMOD	11-10	Выбор режима передачи адреса	
		0	Отсутствует передача адреса
		1	Передача адреса по одной линии данных (SIO)
		2	Передача адреса по двум линиям данных (DIO)
		3	Передача адреса по четырем линиям данных (QIO)
IMOD	9-8	Выбор режима инструкции	
		0	Нет иструкции
		1	Инструкция для режима одной линии данных (SIO)
		2	Инструкция для режима двух линий данных (DIO)
		3	Инструкция для режима четырех линий данных (QIO)
INST	7-0	Код инструкции, отправляемый во внешнее устройство по QSPI	
–	30-28, 23	Зарезервировано	

QAD – регистр адреса
Смещение: + 1Ch
Сброс: 00h



Поле	Биты	Описание
VAL	31-0	Значение адреса, передаваемого во внешнюю Flash память

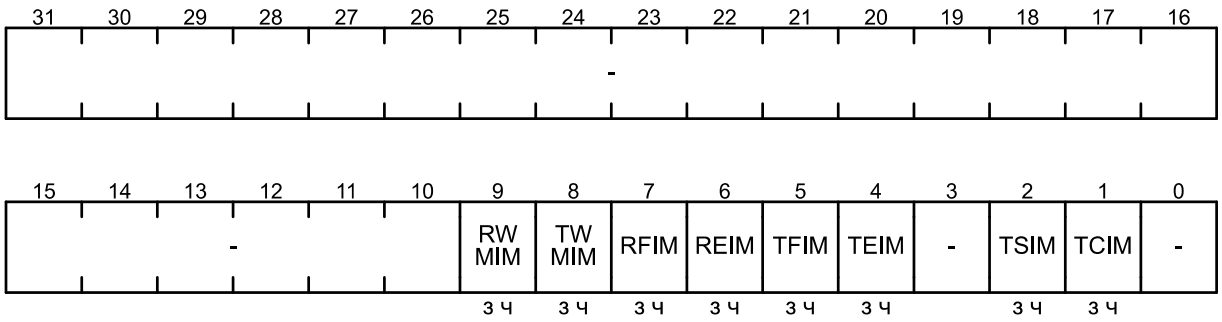
QAB – регистр дополнительных данных
Смещение: + 20h
Сброс: 00h



Поле	Биты	Описание
VAL	31-0	Значение дополнительных данных, передаваемых во внешнее устройство

IMR – регистр маски прерываний

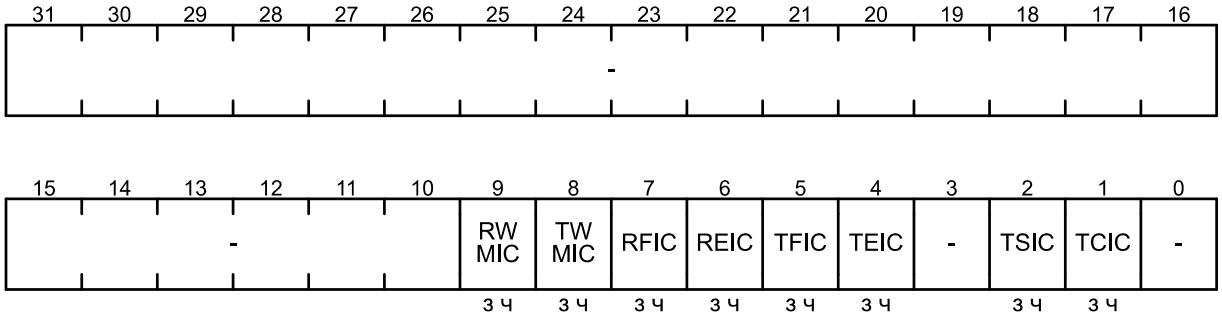
Смещение: + 24h
Сброс: 00h



Поле	Биты	Описание
RWIM	9	Порог переполнения буфера приемника
TWIM	8	Порог опустошения буфер передатчика
RFIM	7	Буфер FIFO приемника заполнен
REIM	6	Буфер FIFO приемника пуст
TFIM	5	Буфер FIFO передатчика заполнен
TEIM	4	Буфер FIFO передатчика пуст
TSIM	2	Останов передачи
TCIM	1	Завершение передачи
–	31-10, 3, 0	Зарезервировано

ICR – регистр сброса прерываний

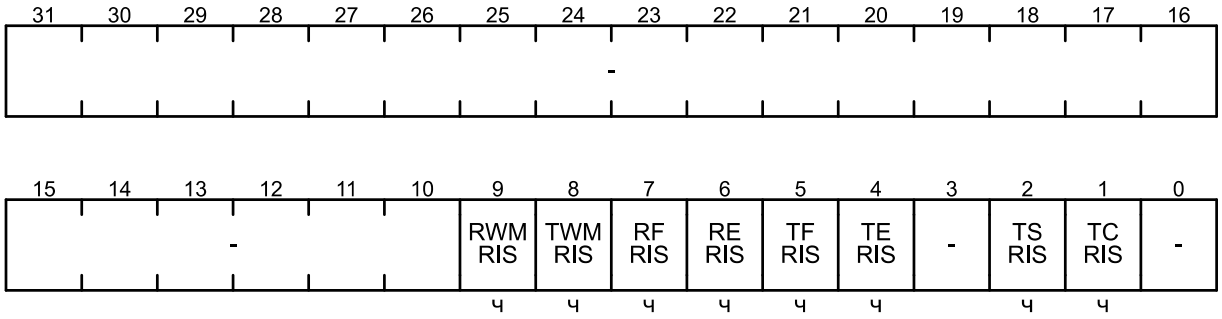
Смещение: + 28h
Сброс: 00h



Поле	Биты	Описание
RWIC	9	
TWIC	8	
RFIC	7	
REIC	6	
TFIC	5	
TEIC	4	
TSIC	2	
TCIC	1	
–	31-10, 3, 0	Зарезервировано

RIS – регистр состояния прерываний

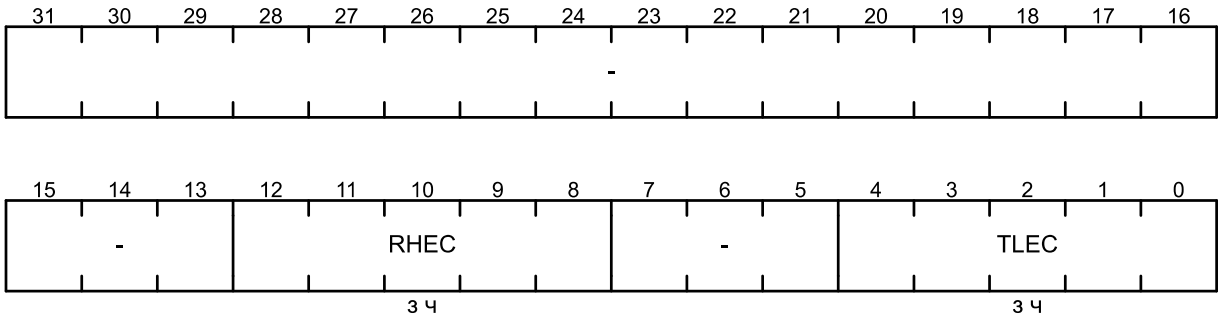
Смещение: + 2Ch
Сброс: 00h



Поле	Биты	Описание
RWRIS	9	Флаг порогового заполнения буфера приемника
TWRIS	8	Флаг порогового опустошения буфера передатчика
RFRIS	7	Флаг заполнения буфера приемника
RERIS	6	Флаг опустошения буфера приемника
TFRIS	5	Флаг заполнения буфера передатчика
TERIS	4	Флаг опустошения буфера передатчика
TSRIS	2	Флаг останова передачи
TCRIS	1	Флаг завершение передачи
–	31-10, 3, 0	Зарезервировано

FWM – регистр пороговых значений FIFO

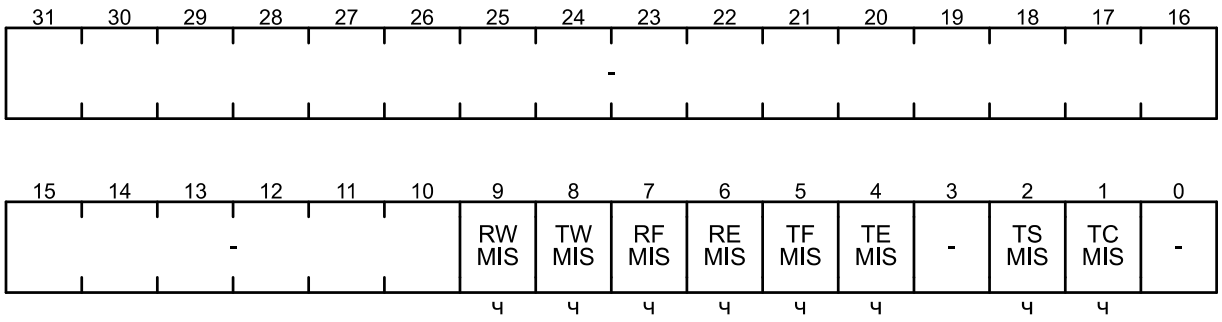
Смещение: + 30h
Сброс: 0000_0E02h



Поле	Биты	Описание
RHEC	12-8	Уровень порогового заполнения буфера приемника. При достижении заданного значения будет выставлен флаг RWRIS регистра RIS
TLEC	4-0	Уровень порогового опустошения буфера передатчика. При достижении заданного значения будет выставлен флаг TWRIS регистра RIS
–	31-13, 7-6	Зарезервировано

MIS – регистр состояния прерываний с маскированием

Смещение: + 34h
Сброс: 00h



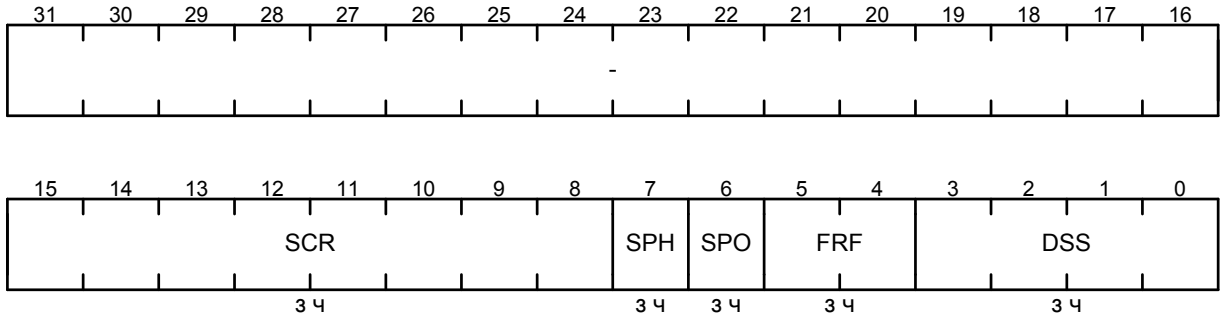
Поле	Биты	Описание
RWMIS	9	Флаг порогового заполнения буфера приемника
TWMIS	8	Флаг порогового опустошения буфера передатчика
RFMIS	7	Флаг заполнения буфера приемника
REMIS	6	Флаг опустошения буфера приемника
TFMIS	5	Флаг заполнения буфера передатчика
TEMIS	4	Флаг опустошения буфера передатчика
TSMIS	2	Флаг останова передачи
TCMIS	1	Флаг завершение передачи
–	31-10, 3, 0	Зарезервировано

A.22 Регистры контроллера интерфейса SPI

Базовый адрес: 2005_0000h Регистры контроллера SPI0
2006_0000h Регистры контроллера SPI1

CR0 – регистр управления 0

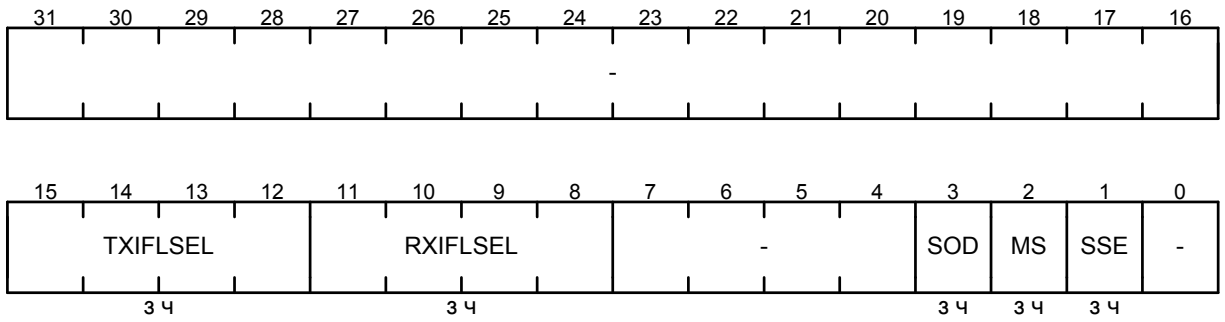
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание	
SCR	15-8	Коэффициент деления второго делителя. Может принимать значения 00h до FFh	
SPH	7	Фаза сигнала SPIx_SCK (только для протокола обмена SPI)	
		0	Выборка данных по переднему фронту синхросигнала, а установка – по заднему
		1	Выборка данных по заднему фронту синхросигнала, а установка – по переднему
SPO	6	Полярность сигнала SPIx_SCK (только для протокола обмена SPI)	
		0	В режиме ожидания линия SPIx_SCK удерживается в состоянии логического нуля
		1	В режиме ожидания линия SPIx_SCK удерживается в состоянии логической единицы
FRF	5-4	Поле выбора протокола обмена информацией	
		00	SPI
		01	SSI
		10	Microwire
		11	Зарезервировано
DSS	3-0	Размер слова данных	
		0h-2h	Зарезервировано
		3h	4 бита
		4h	5 бит
		...	…
		Eh	15 бит
		Fh	16 бит
–	31-16	Зарезервировано	

CR1 – регистр управления 1

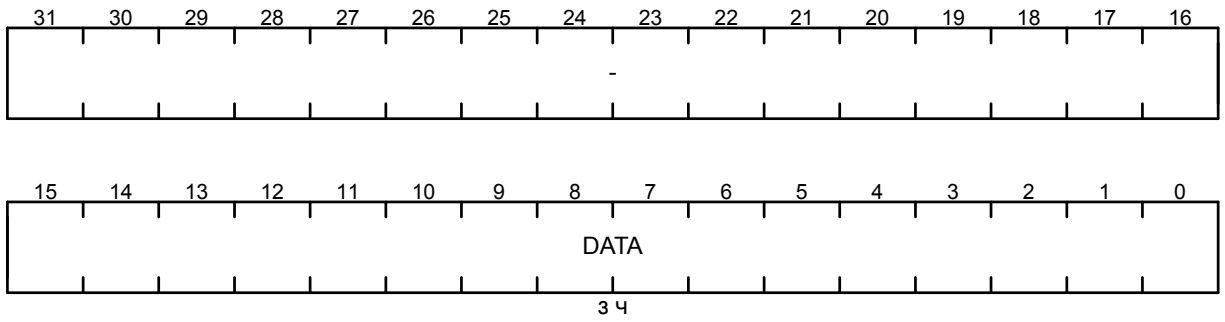
Смещение: + 04h
Сброс: 4400h



Поле	Биты	Описание
TXIFLSEL	15-12	Величина порога опустошения передающего FIFO. При опустошении до уровня порога или ниже может быть сгенерировано прерывание или запрос DMA. Допустимый диапазон значений 0h-8h (по умолчанию – 4h)
RXIFLSEL	11-8	Величина порога наполнения принимающего FIFO. При заполнении до уровня порога или выше может быть сгенерировано прерывание или запрос DMA. Допустимый диапазон значений 0h-8h (по умолчанию – 4h)
SOD	3	Бит запрета передачи данных. В режиме мастера значение бита игнорируется. В режиме ведомого бит контролирует выход данных. Пока бит сброшен передача и прием данных разрешены. Установка бита блокирует передачу данных и переводит вывод SPIx_TX в состояние слабой логической единицы, при этом прием тактового сигнала и прием данных не блокируется
MS	2	Бит выбора режима работы
		0 Мастер
		1 Ведомый
SSE	1	Бит разрешения работы приемопередатчика
		0 Запрещено
		1 Разрешено
–	31-16, 7-4, 0	Зарезервировано

DR – регистр данных

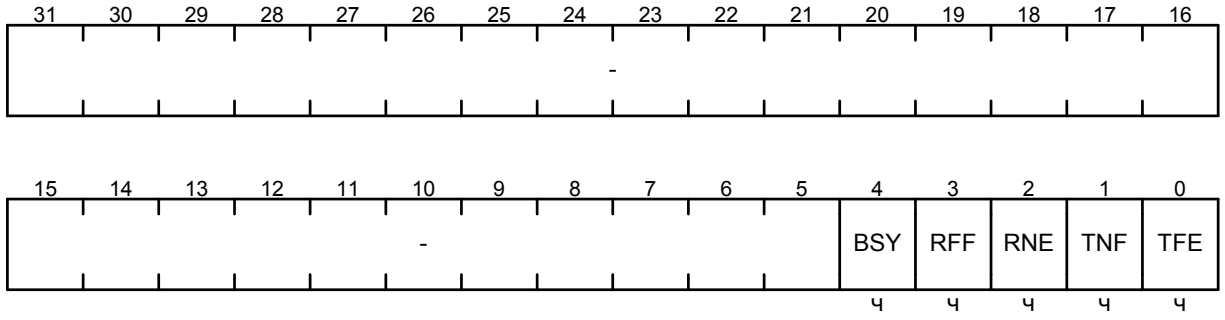
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание
DATA	15-0	16-разрядный буфер FIFO приемника и передатчика. Данные для передачи записываются в регистр. Если размер данных менее 16 бит, они должны быть выравнены по правой границе. Принятые данные автоматически выравниваются по правой границе. Принятые данные возвращаются при чтении регистра.
–	31-16	Зарезервировано

SR – регистр состояния

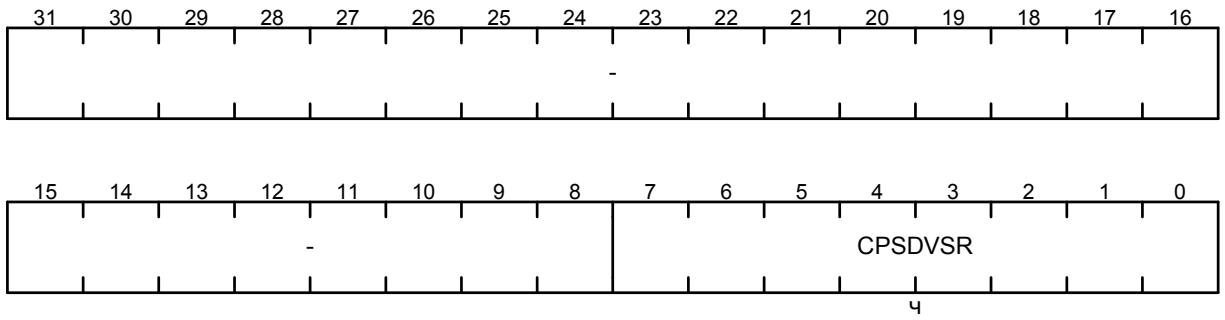
Смещение: + 0Ch
Сброс: 3h



Поле	Биты		Описание
BSY	4	0	Приемопередатчик не активен
		1	Передача/прием данных, либо буфер FIFO не пуст
RFF	3	0	Буфер FIFO приемника не заполнен
		1	Буфер FIFO приемника заполнен
RNE	2	0	Буфер FIFO приемника пуст
		1	Буфер FIFO приемника не пуст
TNF	1	0	Буфер FIFO передатчика заполнен
		1	Буфер FIFO передатчика не заполнен
TFE	0	0	Буфер FIFO передатчика не пуст
		1	Буфер FIFO передатчика пуст
–	31-5	Зарезервировано	

CPSR – регистр делителя тактовой частоты

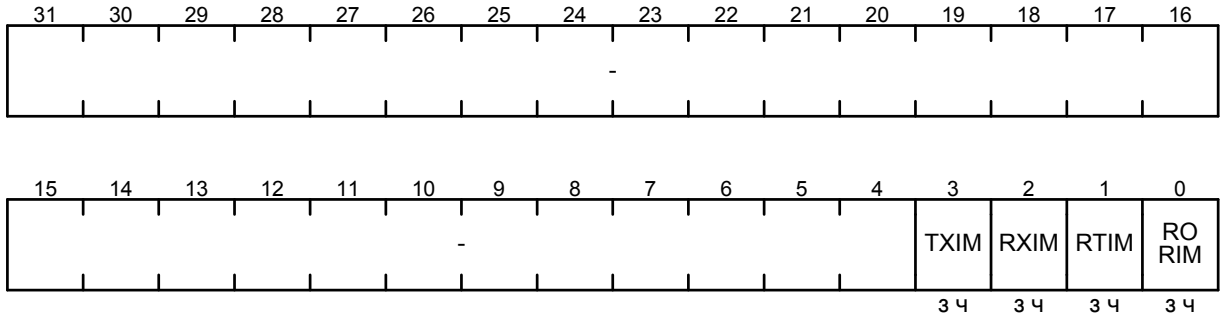
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
CPSDVSR	7-0	Коэффициент деления первого делителя. Может принимать четные значения от 02h до FEh
–	31-8	Зарезервировано

IMSC – регистр маски прерываний

Смещение: + 14h
Сброс: 0h

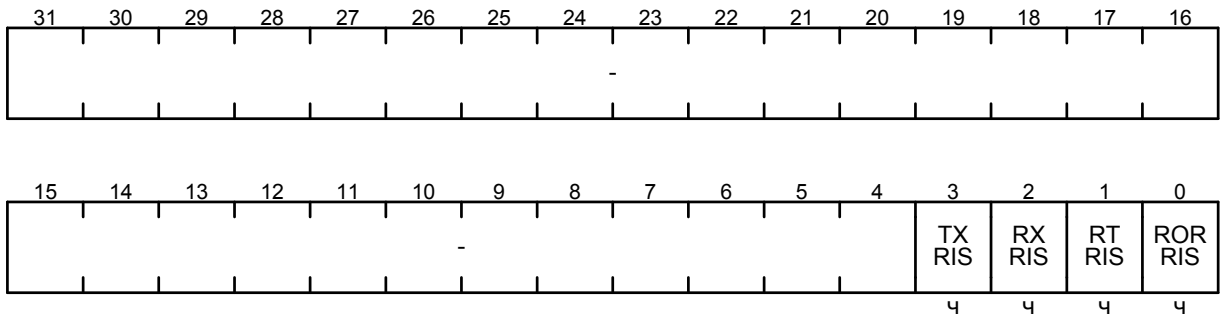


Поле	Биты	Описание
TXIM	3	Буфер передатчика опустошен до величины порога или ниже
RXIM	2	Буфер приемника заполнен на величину порога или выше
RTIM	1	Таймаут приема данных
RORIM	0	Переполнение буфера приемника
–	31-4	Зарезервировано

Примечание – Установка/сброс бит формирует маску. По умолчанию все биты сброшеныи установка флагов запрещена.

RIS – регистр состояния прерываний

Смещение: + 18h
Сброс: 8h

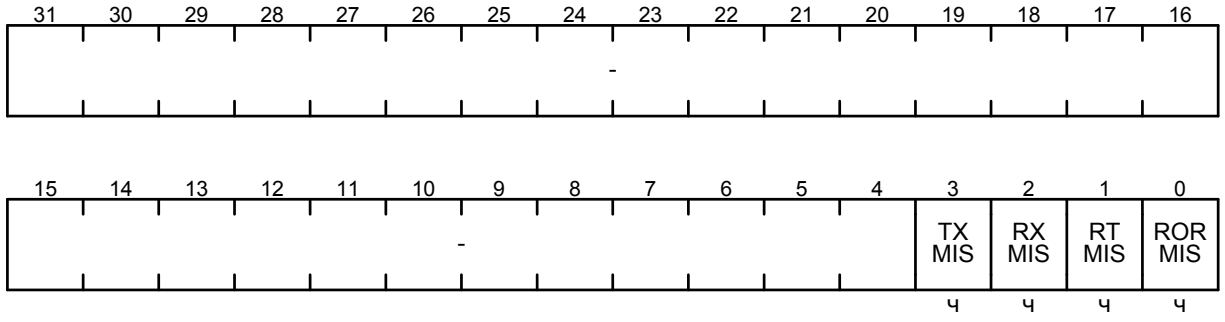


Поле	Биты	Описание
TXRIS	3	Буфер передатчика опустошен до величины порога или ниже
RXRIS	2	Буфер приемника заполнен на величину порога или выше
RTRIS	1	Таймаут приема данных
RORRIS	0	Переполнение буфера приемника
–	31-4	Зарезервировано

Примечание – При возникновении прерываний устанавливаются соответствующие им немаскируемые флаги. Биты RTRIS также сбрасываются после чтения буфера приемника.

MIS – регистр состояния прерываний с маскированием

Смещение: + 1Ch
Сброс: 0h

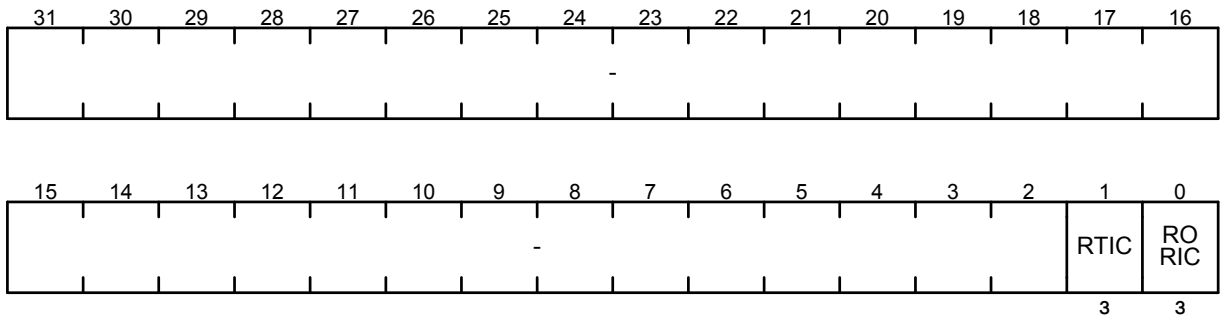


Поле	Биты	Описание
TXMIS	3	Буфер передатчика опустошен до величины порога или ниже
RXMIS	2	Буфер приемника заполнен на величину порога или выше
RTMIS	1	Таймаут приема данных
RORMIS	0	Переполнение буфера приемника
–	31-4	Зарезервировано

Примечание – В регистре устанавливаются только те флаги, которые закрыты маской регистра IMSC. Бит RTMIS также сбрасывается после чтения буфера приемника.

ICR – регистр сброса прерываний

Смещение: + 20h
Сброс: 0h

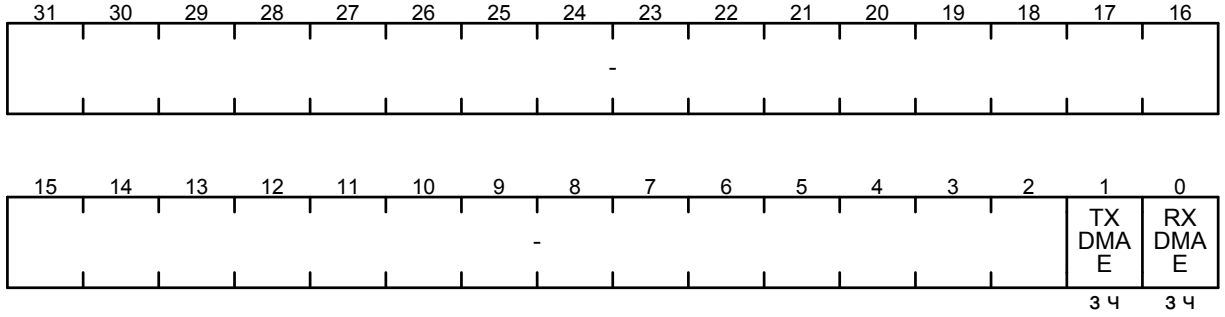


Поле	Биты	Описание
RTIC	1	Таймаут приема данных
RORIC	0	Переполнение буфера приемника
–	31-4	Зарезервировано

Примечание – Запись единиц в биты регистра сбрасывает соответствующие им флаги в регистрах RIS и MIS, а также прерывания, вызвавшие установку этих флагов.

DMACR – регистр управления прямым доступом к памяти

Смещение: + 24h
Сброс: 0h



Поле	Биты	Описание	
TXDMAE	1	Бит разрешения использования контроллера DMA при передаче	
		0	Не используется
		1	Разрешено формирование запросов контроллера DMA для обслуживания буфера FIFO передатчика
RXDMAE	0	Бит разрешения использования контроллера DMA при приеме	
		0	Не используется
		1	Разрешено формирование запросов контроллера DMA для обслуживания буфера FIFO приемника
–	31-2	Зарезервировано	

A.23 Регистры контроллера интерфейса USB

Базовый адрес: 2001_0000h

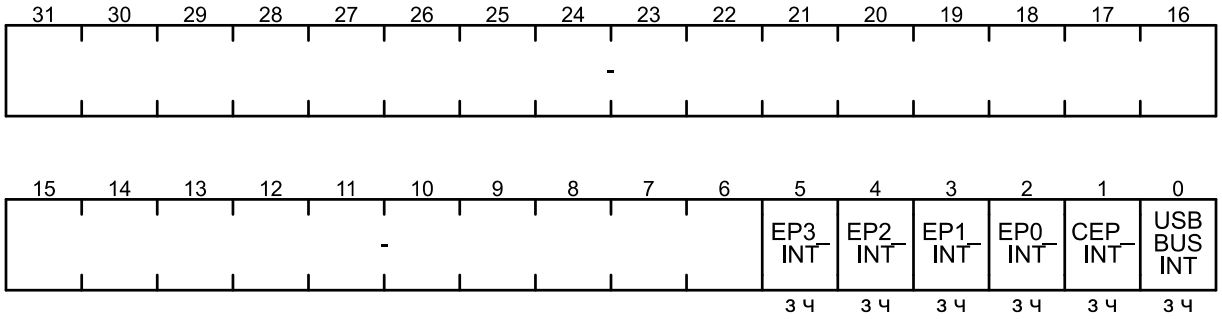
Смещение:	+ 28h (CEP)	Регистры настройки контрольной конечной точки CEP
	+ 70h (EP0)	Регистры настройки конечной точки EP0
	+ B0h (EP1)	Регистры настройки конечной точки EP1
	+ F0h (EP2)	Регистры настройки конечной точки EP2
	+ 130h (EP3)	Регистры настройки конечной точки EP3

Мнемоника: EPn;
DCd

Примечание – n – номер конечной точки от 0 до 3.

INTSTAT0 – регистр статуса прерываний 0

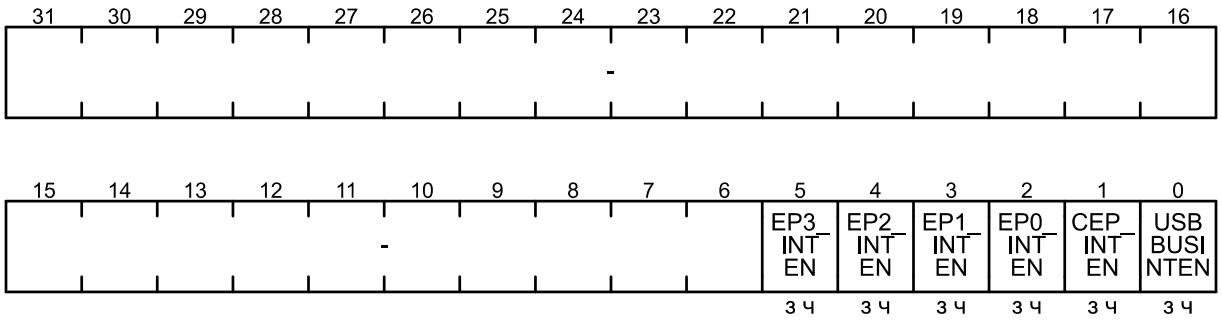
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание
EP3_INT	5	Прерывание от конечной точки 3
EP2_INT	4	Прерывание от конечной точки 2
EP1_INT	3	Прерывание от конечной точки 1
EP0_INT	2	Прерывание от конечной точки 0
CEP_INT	1	Прерывание от управляющей конечной точки
USBBUSINT	0	Прерывание от шины USB
–	31-6	Зарезервировано

INTEN0 – регистр разрешения прерываний 0

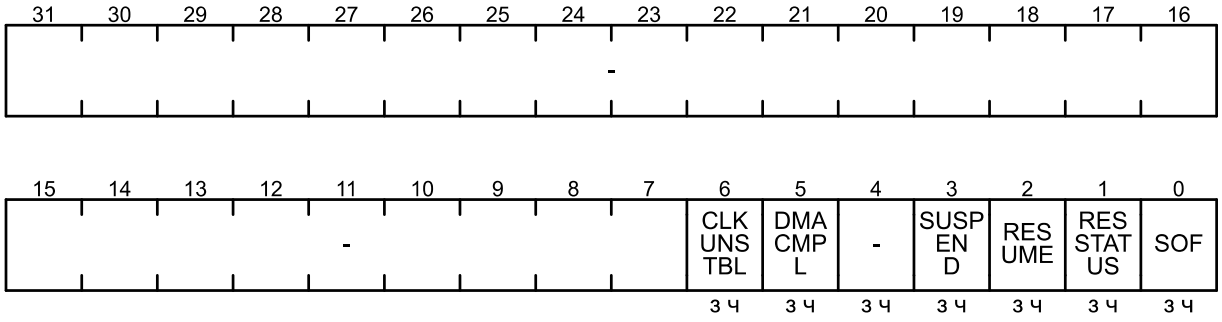
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание
EP3_INTEN	5	Разрешение прерывания от конечной точки 3
EP2_INTEN	4	Разрешение прерывания от конечной точки 2
EP1_INTEN	3	Разрешение прерывания от конечной точки 1
EP0_INTEN	2	Разрешение прерывания от конечной точки 0
CEP_INTEN	1	Разрешение прерывания от управляющей конечной точки
USBBUSINTEN	0	Разрешение прерывания от шины USB
–	31-6	Зарезервировано

INTSTAT1 – регистр статуса прерываний 1

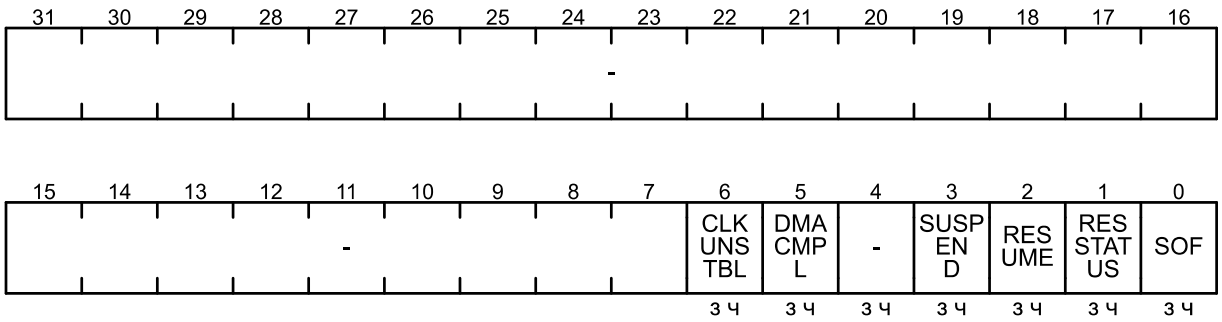
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
CLKUNSTBL	6	Устанавливается при наличии стабильной тактовой частоты блока USB.(при низкой тактовой частоте работы контроллера есть вероятность пропуска этого прерывания)
DMACMPL	5	Флаг завершения передачи данных по DMA
SYSPEND	3	Бит запроса режима SUSPEND. По умолчанию, бит установлен и должен быть сброшен перед сбросом USB. Также бит устанавливается, при приходе запроса остановки от хоста
RESUME	2	Флаг возобновления работы устройства
RESTATUS	1	Флаг завершения сброса корневого порта
SOF	0	Флаг приема пакета SOF
–	31-7, 4	Зарезервировано

INTEN1 – регистр разрешения прерываний 1

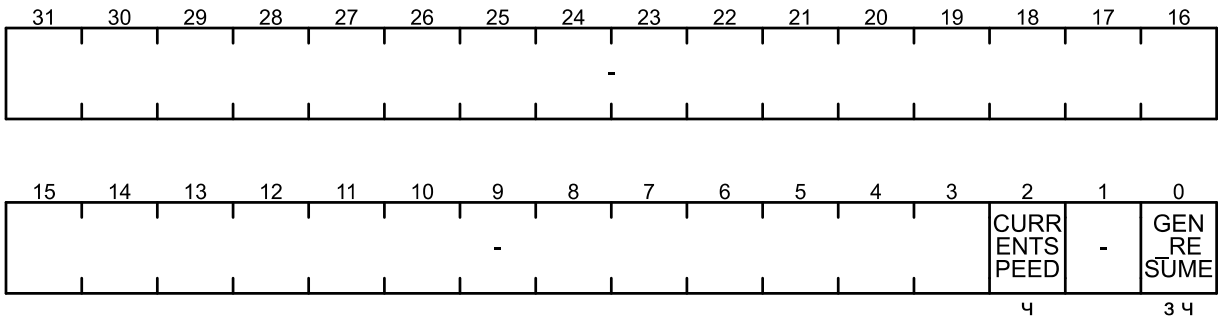
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание
CLKUNSTBL	6	Разрешение прерывания по нестабильной тактовой частоте блока USB.
DMACMPL	5	Разрешение прерывания по завершению передачи данных по DMA
SYSPEND	3	Разрешение прерывания по запросу режима SUSPEND
RESUME	2	Разрешение прерывания по возобновлению работы устройства
RESTATUS	1	Разрешение прерывания по завершению сброса корневого порта
SOF	0	Разрешение прерывания по приему пакета SOF
–	31-7, 4	Зарезервировано

OPERATIONS – регистр операций

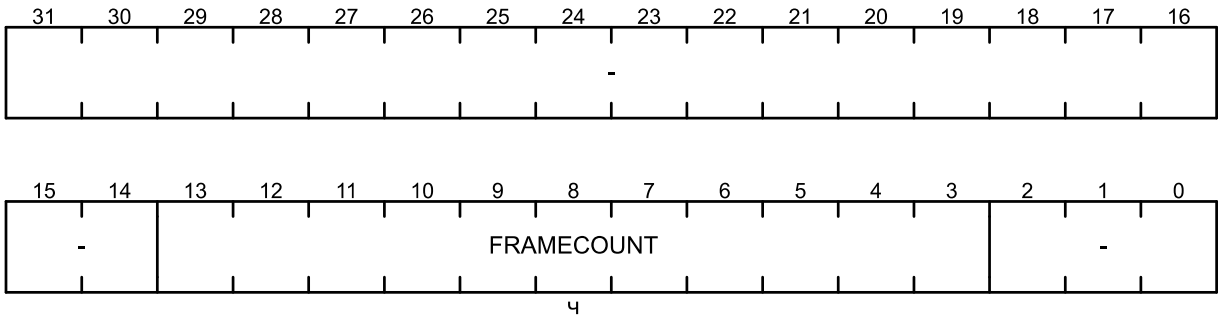
Смещение: + 18h
Сброс: 0h



Поле	Биты	Описание	
CURRENTSPEED	2	Индикатор скорости работы контроллера устройства	
		0	Full Speed
		1	Зарезервировано
GEN_RESUME	0	Бит запуска восстановления работы	
		0	Нет действий
		1	Установка бита запускает последовательность восстановления работы, если удаленное восстановление разрешено
–	31-3, 1	Зарезервировано	

FRAMECNT – регистр счетчика кадров

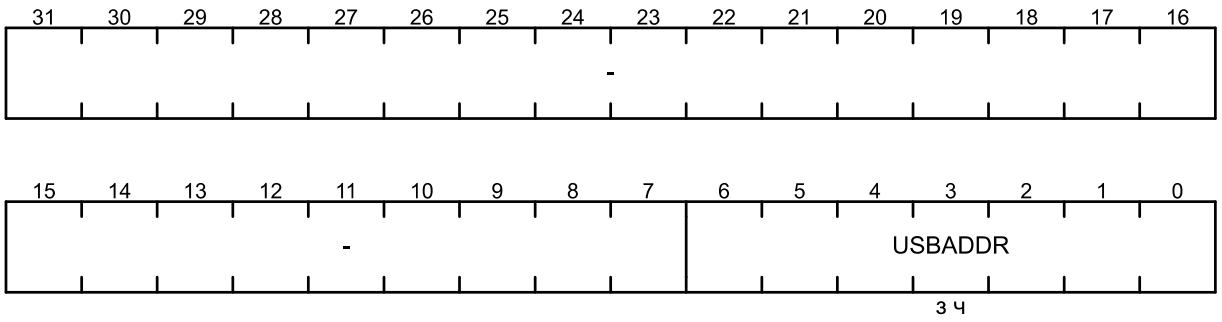
Смещение: + 1Ch
Сброс: 0h



Поле	Биты	Описание
FRAMCOUNT	13-3	Поле счетчика кадров с момента последнего пакета SOF
–	31-14, 2-0	Зарезервировано

USBADDR – регистр адреса на шине USB

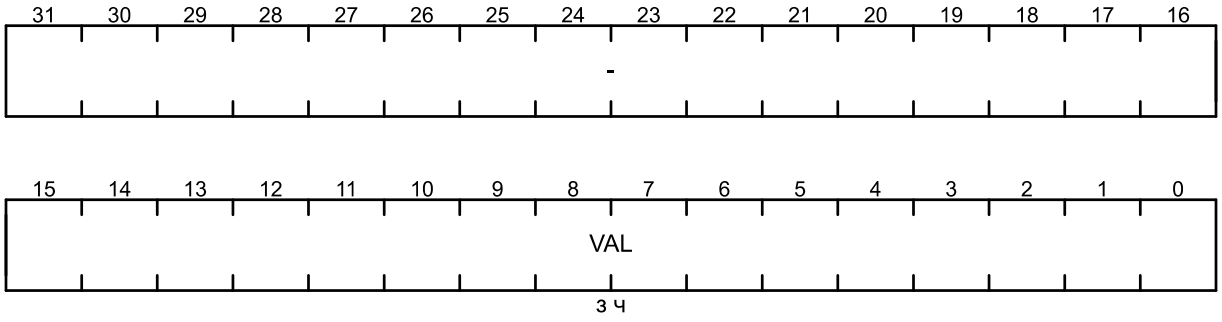
Смещение: + 20h
Сброс: 0h



Поле	Биты	Описание
USBADDR	6-0	Поле текущего адреса устройства. Поле сбрасывается при сбросе корневого порта
–	31-7	Зарезервировано

CEP_DATA_BUF – регистр буфера данных управляющей конечной точки

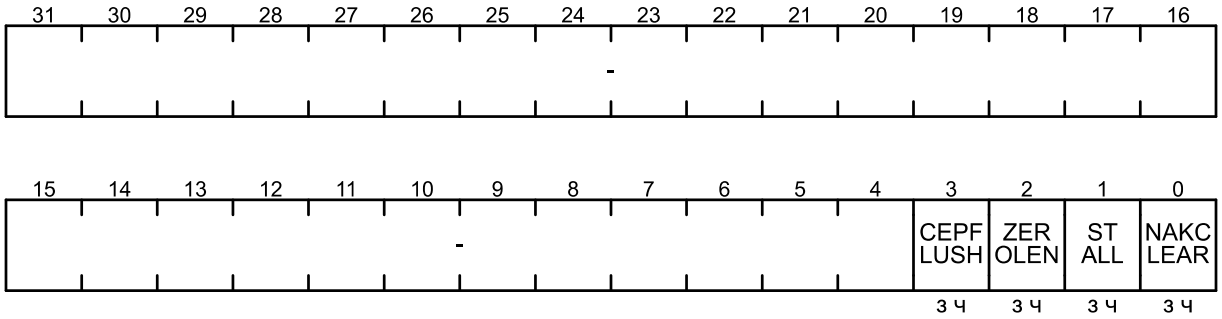
Смещение: + 28h
Сброс: 0h



Поле	Биты	Описание
VAL	15-0	Буфер данных для передачи
–	31-16	Зарезервировано

CEP_CTRL_STAT – регистр управления и состояния управляющей конечной точки

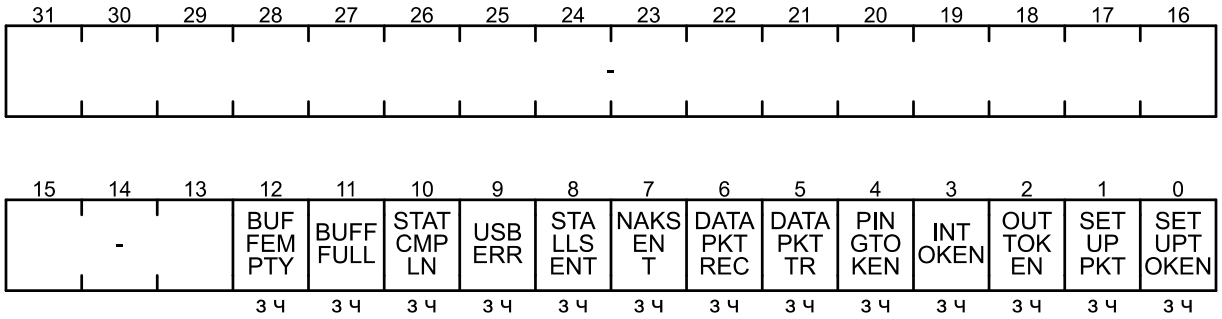
Смещение: + 2Ch
Сброс: 0h



Поле	Биты	Описание	
CEPFLUSH	3	Бит сброса буфера Установка бита сбрасывает содержимое буфера	
ZEROLEN	2	Бит разрешения передачи пакета нулевой длины	
		0	Запрещено
		1	При работе в режиме «Auto Validation» установка этого бита разрешает посылку пакета нулевой длины в ответ на метку IN. Сразу после отправки пакета бит сбрасывается
STALL	1	Бит разрешения отправки «Stall» После записи единицы в бит, управляющий буфер отправляет подтверждение «Stall» в ответ на любую метку IN, или OUT. При этом бит NAK_CLEAR должен быть сброшен, т. к. он имеет более высокий приоритет	
NAKCLEAR	0	Флаг приема метки SETUP Устанавливается контроллером каждый раз при приеме метки SETUP. Пока бит установлен, контроллер будет отвечать меткой NAK на все запросы	
–	31-4	Зарезервировано	

CEP_INTEN – регистр разрешения прерываний буфера управляющей конечной точки

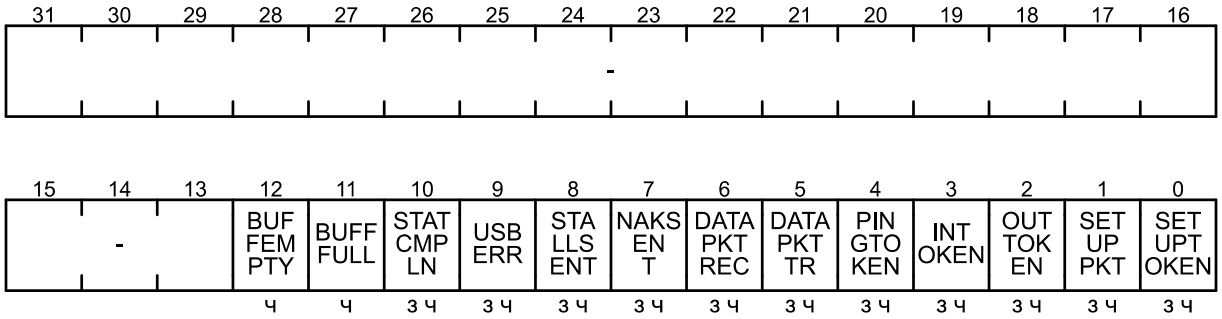
Смещение: + 30h
Сброс: 0h



Поле	Биты	Описание
BUFFEMPTY	12	Разрешение прерывания по опустошению управляющего буфера
BUFFULL	11	Разрешение прерывания по заполнению управляющего буфера
STATCMPLN	10	Разрешение прерывания по успешному завершению стадии «Status» операции на шине USB
USBERR	9	Флаг ошибки при проведении операции
STALLSENT	8	Разрешение прерывания по окончанию посылки метки STALL в ответ на метку IN или OUT
NAKSENT	7	Разрешение прерывания по окончанию посылки метки NAK в ответ на метку IN или OUT
DATAPKTREC	6	Разрешение прерывания по успешному приму пакета данных, следующего за меткой OUT, в ответ на который была отправлена метка ACK
DATAPKTTR	5	Разрешение прерывания по успешной отправке пакета данных в ответ на метку IN с получением подтверждающей метки ACK
PINGTOKEN	4	Разрешение прерывания по окончанию прием метки PING от хоста
INTOKEN	3	Разрешение прерывания по окончанию прием метки IN от хоста
OUTTOKEN	2	Разрешение прерывания по окончанию прием метки OUT от хоста
SETUPPKT	1	Разрешение прерывания по окончанию приема пакета Setup от хоста.
SETUPTOKEN	0	Разрешение прерывания по окончанию приема метки SETUP от хоста
–	31-13	Зарезервировано

CEP_INTSTAT – регистр флагов прерываний буфера управляющей конечной точки

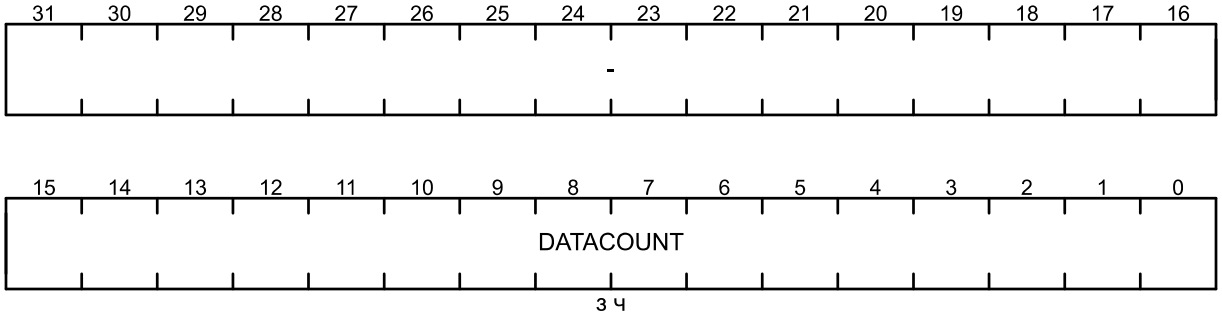
Смещение: + 34h
Сброс: 0h



Поле	Биты	Описание
BUFFEMPTY	12	Флаг опустошения управляющего буфера
BUFFULL	11	Флаг заполнения управляющего буфера
STATCMPLN	10	Флаг успешного завершения стадии «Status» операции на шине USB
USBERR	9	Флаг ошибки при проведении операции
STALLSENT	8	Флаг окончания посылки метки STALL в ответ на метку IN или OUT
NAKSENT	7	Флаг окончания посылки метки NAK в ответ на метку IN или OUT
DATAPKTREC	6	Флаг успешного прима пакета данных, следующего за меткой OUT, в ответ на который была отправлена метка ACK
DATAPKTTR	5	Флаг успешной отправки пакета данных в ответ на метку IN с получением подтверждающей метки ACK
PINGTOKEN	4	Флаг окончания прием метки PING от хоста
INTOKEN	3	Флаг окончания прием метки IN от хоста
OUTTOKEN	2	Флаг окончания прием метки OUT от хоста
SETUPPKT	1	Флаг окончания приема пакета Setup от хоста. Флаг должен быть сброшен до приема следующего пакета Setup. В противном случае последующие пакеты будут записываться в буфер поверх предыдущих
SETUPTOKEN	0	Флаг окончания приема метки SETUP от хоста
–	31-13	Зарезервировано

CEP_IN_XFRCNT – регистр счетчика для передачи данных

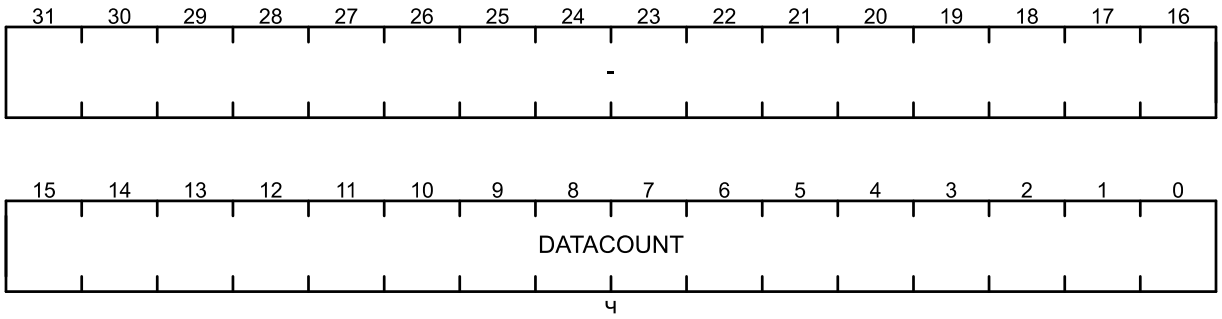
Смещение: + 38h
Сброс: 0h



Поле	Биты	Описание
DATACOUNT	15-0	Количество байт для отправки хосту
–	31-16	Зарезервировано

CEP_OUT_XFRCNT – регистр счетчика принятых данных

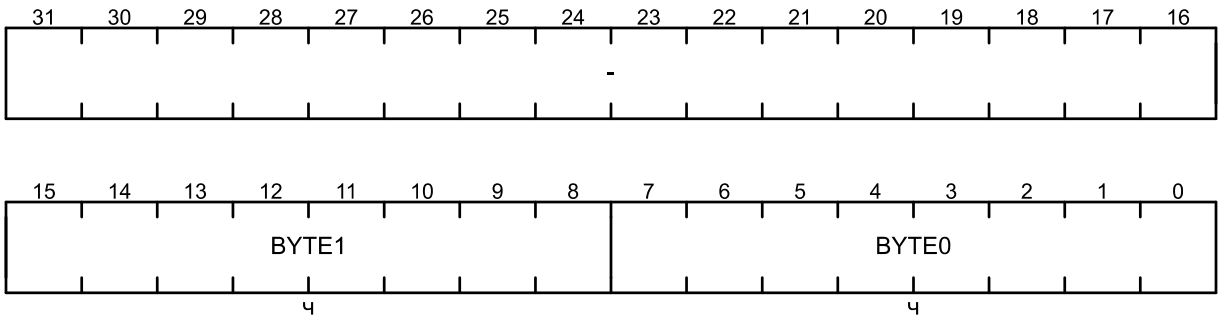
Смещение: + 3Ch
Сброс: 0h



Поле	Биты	Описание
DATACOUNT	15-0	Количество байт полученных от хоста
–	31-16	Зарезервировано

CEP_SETUP1_0 – регистр нулевого и первого байтов пакета Setup

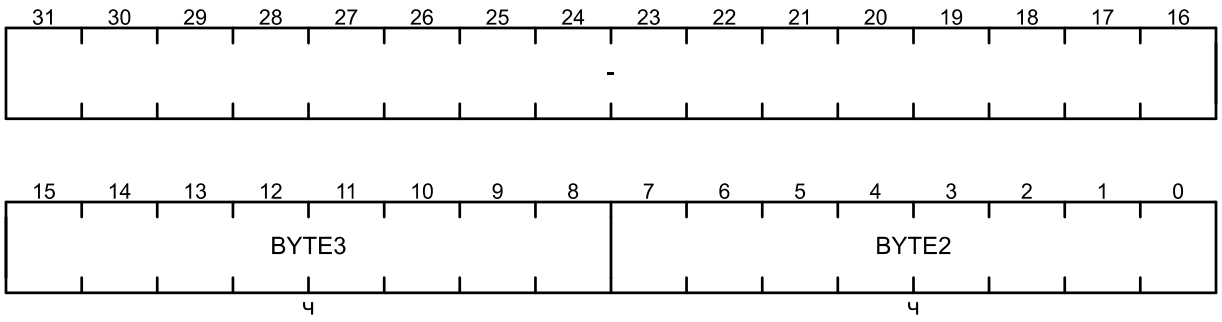
Смещение: + 44h
Сброс: 0h



Поле	Биты	Описание	
BYTE1	15-8	Старший байт пакета Setup	
		00h	Получить статус
		01h	Сбросить параметр
		02h	Зарезервировано
		03h	Установить параметр
		04h	Зарезервировано
		05h	Установить адрес
		06h	Установить дескриптор
		07h	Получить дескриптор
		08h	Установить конфигурацию
		09h	Получить конфигурацию
		0Ah	Установить интерфейс
		0Bh	Получить интерфейс
		0Ch	Синхрокадр
BYTE0	7-0	Младший байт пакета Setup	
	7	Бит направления	
		0	От хоста к устройству
		1	От устройства к хосту
	6-5	Тип	
		00	Standart
		01	Class
		10	Vendor
		11	Зарезервировано
	4-0	Получатель	
		0h	Устройство
		1h	Интерфейс
		2h	Буфер
		3h	Другое
		4h-1Fh	Зарезервировано
–	31-16	Зарезервировано	

CEP_SETUP3_2 – регистр второго и третьего байтов пакета Setup

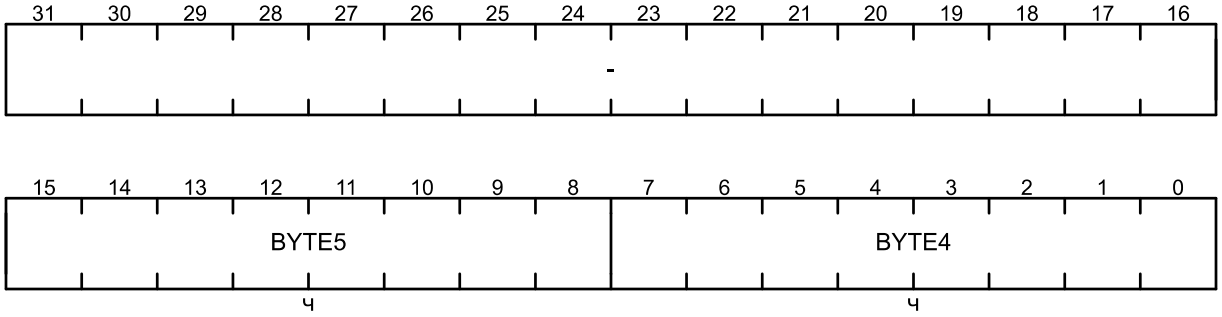
Смещение: + 48h
Сброс: 0h



Поле	Биты	Описание
BYTE3	15-8	Третий байт пакета Setup
BYTE2	7-0	Второй байт пакета Setup
–	31-16	Зарезервировано

CEP_SETUP5_4 – регистр четвертого и пятого байтов пакета Setup

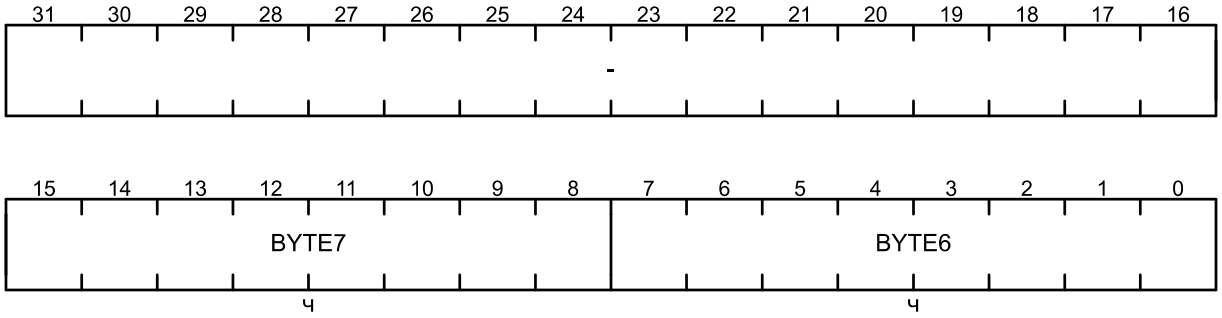
Смещение: + 4Ch
Сброс: 0h



Поле	Биты	Описание
BYTE5	15-8	Пятый байт пакета Setup
BYTE4	7-0	Четвертый байт пакета Setup
–	31-16	Зарезервировано

CEP_SETUP7_6 – регистр шестого и седьмого байтов пакета Setup

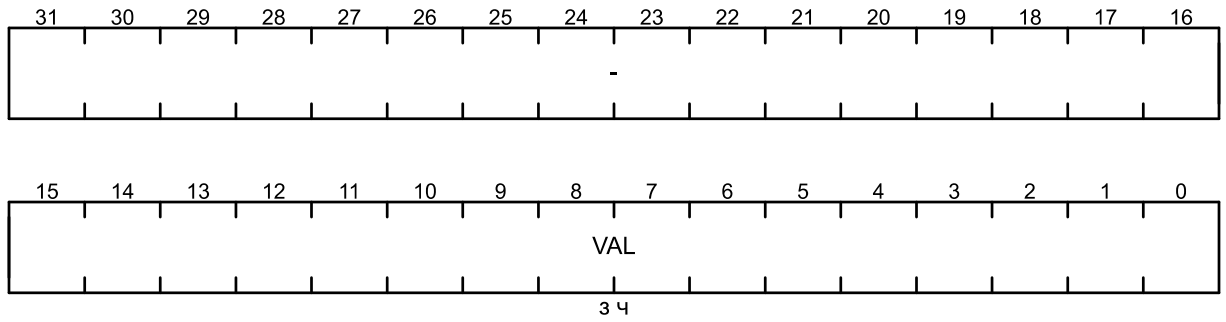
Смещение: + 50h
Сброс: 0h



Поле	Биты	Описание
BYTE7	15-8	Седьмой байт пакета Setup
BYTE6	7-0	Шестой байт пакета Setup
–	31-16	Зарезервировано

CEP_START_ADDR – регистр начального адреса управляющего буфера

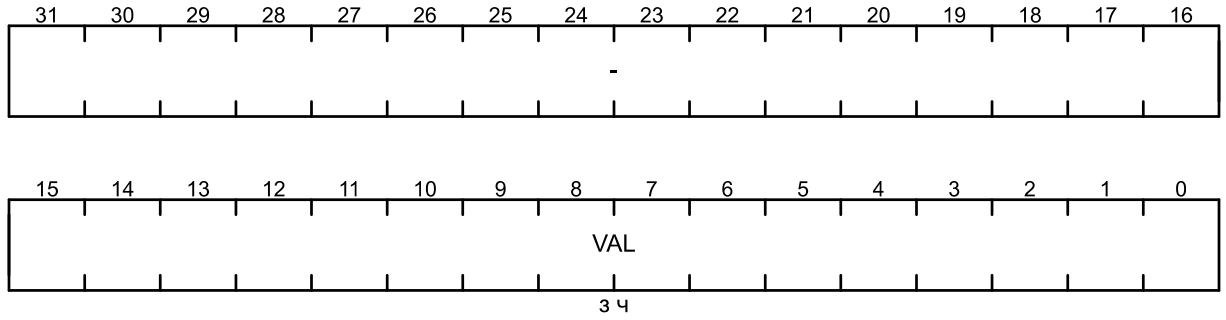
Смещение: + 54h
Сброс: 0h



Поле	Биты	Описание
VAL	15-0	Значение начального адреса в буфере USB
-	31-16	Зарезервировано

CEP_END_ADDR – регистр конечного адреса управляющего буфера

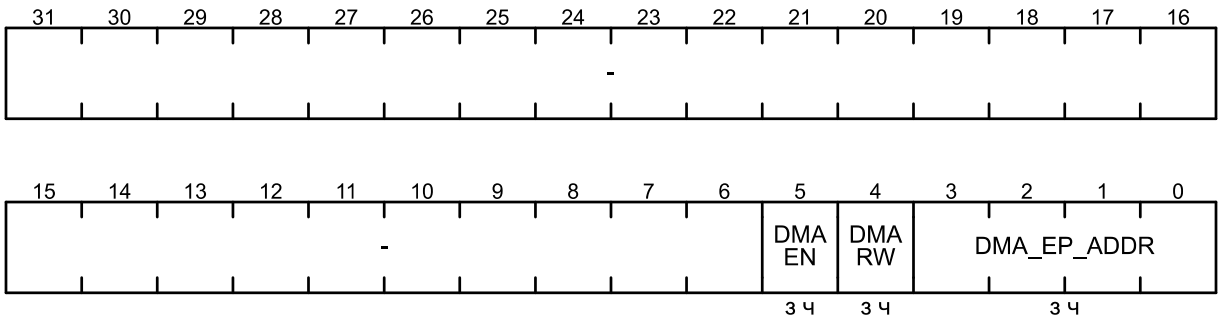
Смещение: + 58h
Сброс: 0h



Поле	Биты	Описание
VAL	15-0	Значение конечного адреса в буфере USB
-	31-16	Зарезервировано

DMA_CTRL_STS – регистр управления и статуса DMA

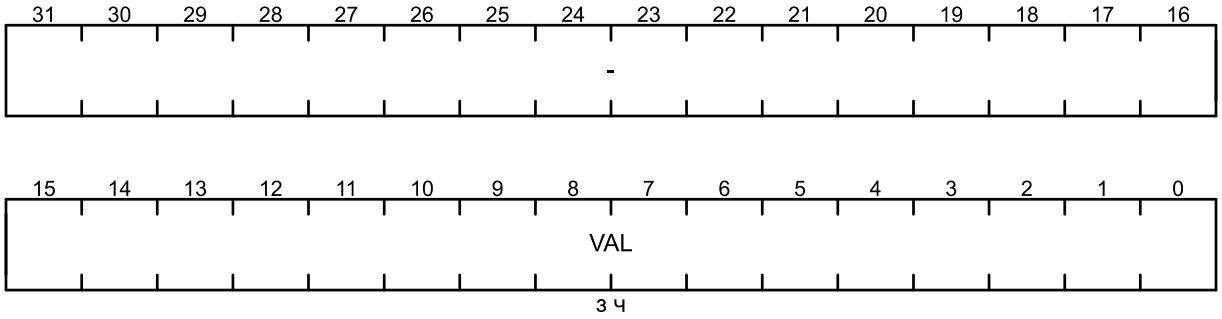
Смещение: + 5Ch
Сброс: 0h



Поле	Биты	Описание	
DMAEN	5	Разрешение работы DMA	
DMARW	4	Выбор операции DMA	
		0	Операция чтения из буфера USB
		1	Операция записи в буфер USB
DMA_EP_ADDR	3-0	Адрес конечной точки DMA	
-	31-6	Зарезервировано	

DMA_CNT – регистр количества байт для обмена DMA

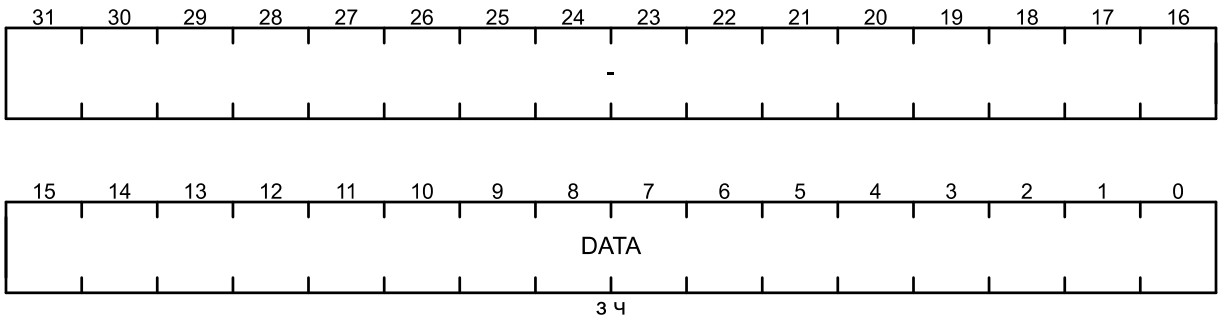
Смещение: + 60h
Сброс: 0h



Поле	Биты	Описание
VAL	15-0	Значение количества байт для обмена через DMA
-	31-16	Зарезервировано

USB_EP.DATA_BUF – регистр обмена данными с буфером n

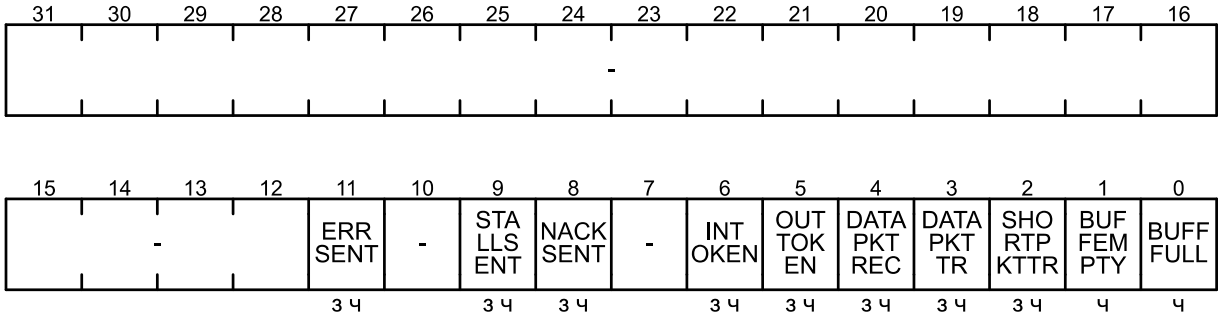
Смещение: EPn + 00h
Сброс: 0h



Поле	Биты	Описание
DATA	15-0	Данные обмена с буфером
-	31-16	Зарезервировано

USB_EP_INTSTAT – регистр статуса прерываний конечной точки

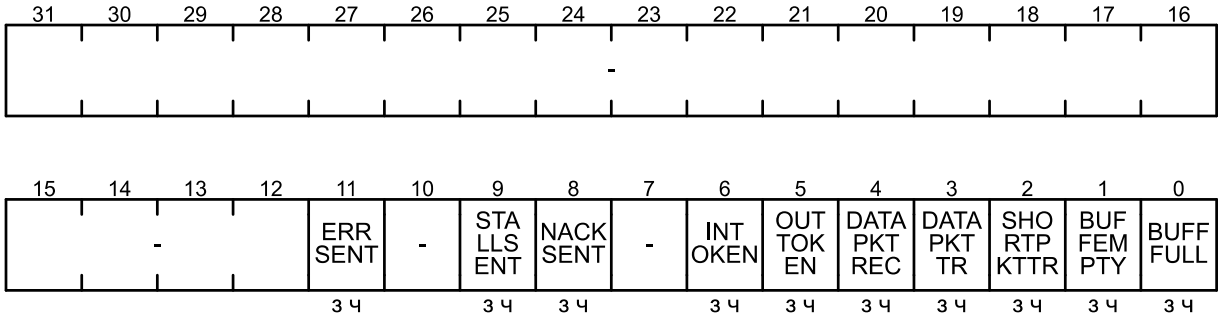
Смещение: EPn + 04h
Сброс: 0h



Поле	Биты	Описание
ERRSENT	11	Флаг прерывания по ошибке
STALLSENT	9	Флаг прерывания по рукопожатию STALL
NACKSENT	8	Флаг прерывания по рукопожатию NACK
INTOKEN	6	Флаг прерывания по получению пакета Token In
OUTTOKEN	5	Флаг прерывания по получению пакета Token Out
DATAPKTREC	4	Флаг прерывания по окончанию приема пакета данных
DATAPKTTR	3	Флаг прерывания по окончанию передачи пакета данных
SHORTPKTTR	2	Флаг прерывания по передаче «короткого» пакета
BUFFEMPTY	1	Флаг прерывания при опустошении буфера конечной точки
BUFFFULL	0	Флаг прерывания при заполнении буфера конечной точки
-	31-12, 10, 7	Зарезервировано

USB_EP_INTEN – регистр разрешения прерываний конечной точки

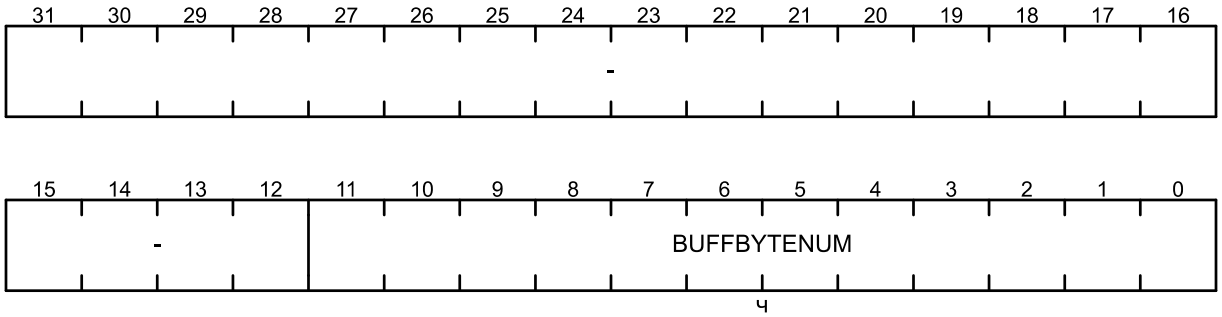
Смещение: EPn + 08h
Сброс: 0h



Поле	Биты	Описание
ERRSENT	11	Разрешение прерывания по ошибке
STALLSENT	9	Разрешение прерывания по рукопожатию STALL
NACKSENT	8	Разрешение прерывания по рукопожатию NACK
INTOKEN	6	Разрешение прерывания по получению пакета Token In
OUTTOKEN	5	Разрешение прерывания по получению пакета Token Out
DATAPKTREC	4	Разрешение прерывания по окончанию приема пакета данных
DATAPKTTR	3	Разрешение прерывания по окончанию передачи пакета данных
SHORTPKTTR	2	Разрешение прерывания по передаче «короткого» пакета
BUFFEMPTY	1	Разрешение прерывания при опустошении буфера конечной точки
BUFFFULL	0	Разрешение прерывания при заполнении буфера конечной точки
-	31-12, 10, 7	Зарезервировано

USB_EP_AVAIL_CNT – регистр количества байт в буфере

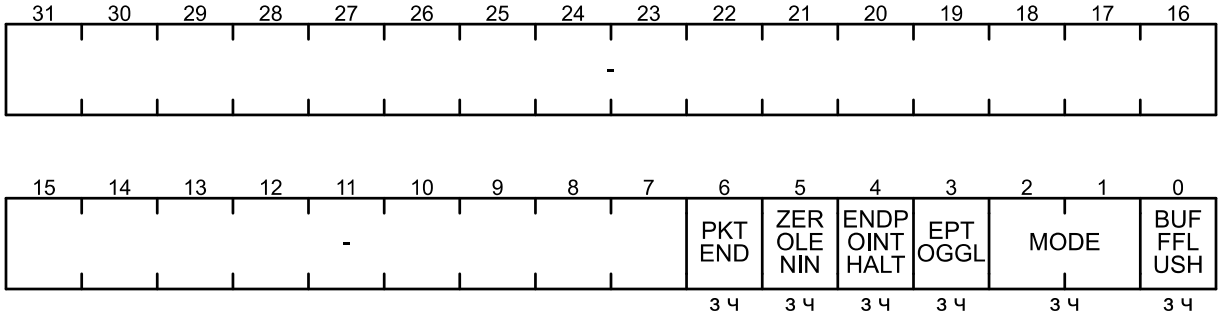
Смещение: EPn + 0Ch
Сброс: 0h



Поле	Биты	Описание
BUFFBYTENUM	11-0	Значение количества байт, находящихся в буфере
-	31-12	Зарезервировано

USB_EP_RSP_SC – регистр управления и состояния конечной точки

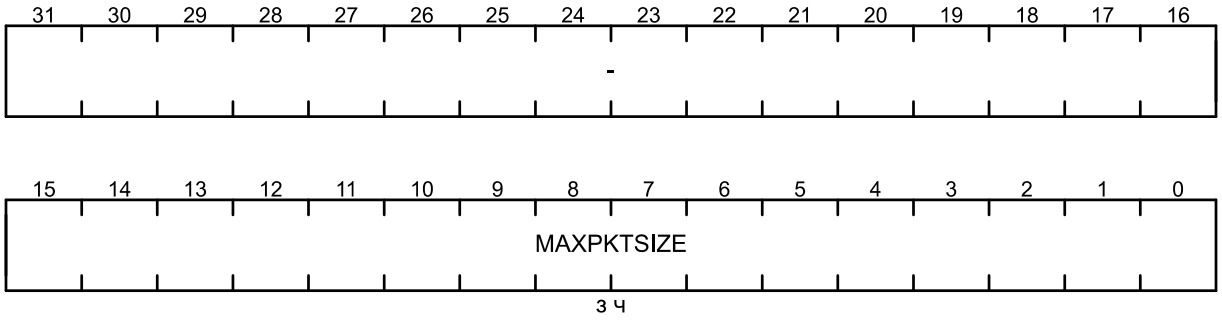
Смещение: EPn + 10h
Сброс: 0h



Поле	Биты	Описание	
PKTEND	6	Конец пакета	
ZEROLENIN	5	Бит разрешения передачи пакета нулевой длины	
ENDPOINTHALT	4	Конечная точке в режиме останова	
ENDPOINTTOGGL	3		
MODE	2-1	Выбор режима работы конечной точки IN	
		0	Режим автоматической проверки длины пакета
		1	Режим проверки длины пакета вручную
		2	Плавающий режим
		3	Зарезервировано
BUFFFLUSH	0	Бит сброса буфера Установка бита сбрасывает содержимое буфера	
-	31-12	Зарезервировано	

USB_EP_MPS – регистр максимального размера пакета

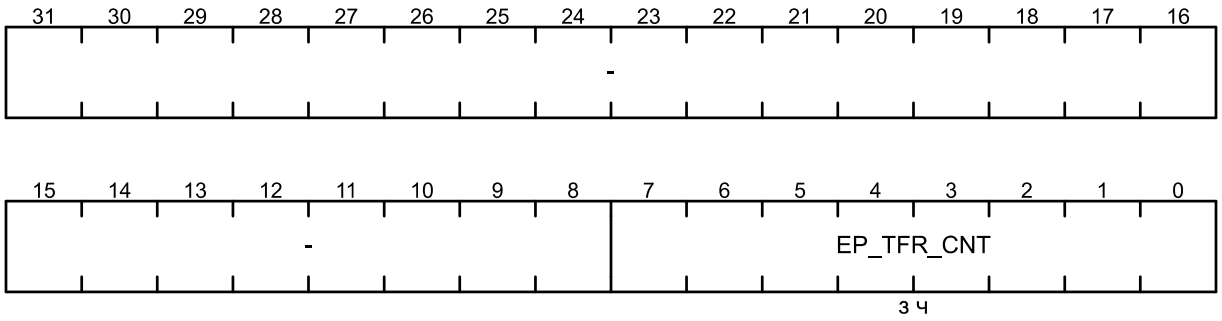
Смещение: EPn + 14h
Сброс: 0h



Поле	Биты	Описание
MAXPKTSIZE	15-0	Значение максимального размера пакета данных
-	31-16	Зарезервировано

USB_EP_CNT – регистр количества передач

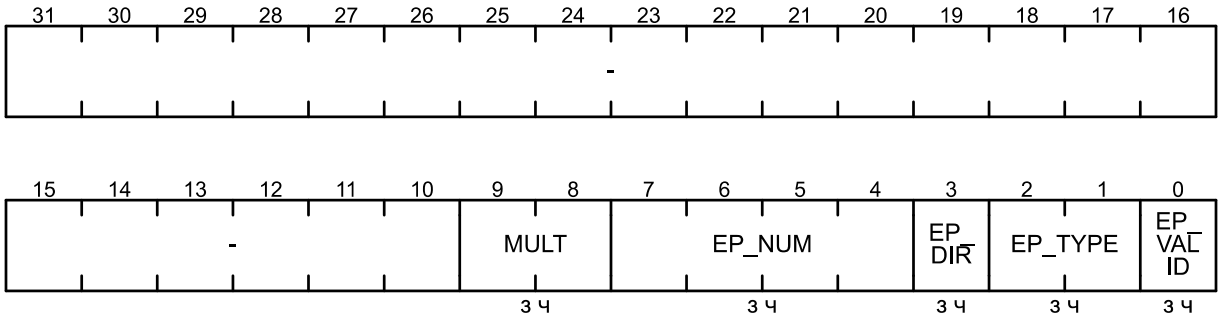
Смещение: EPn + 18h
Сброс: 0h



Поле	Биты	Описание
EP_TFR_CNT	7-0	Значение количества передач конечной точки
-	31-8	Зарезервировано

USB_EP_CFG – регистр конфигурации конечной точки

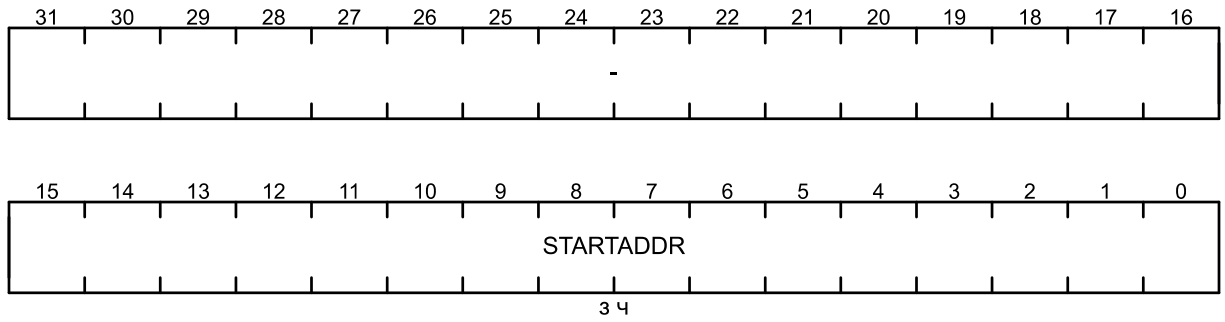
Смещение: EPn + 1Ch
Сброс: 0h



Поле	Биты		Описание
MULT	9-8		Поле числа операций, выполняемых в одном микрокадре
		0	Одна
		1	Две
		2	Три
		3	Зарезервировано
EP_NUM	7-4		Номер конечной точки. Допустимые значения от 1 до 15
EP_DIR	3		Выбор направления работы конечной точки
		0	OUT EP
		1	IN EP
EP_TYPE	2-1		Выбор типа конечной точки
		0	Зарезервировано
		1	Isochronous
		2	Bulk
		3	Interrupt
EP_VALID			Бит включения конечной точки
-	31-10		Зарезервировано

USB_EP_START_ADDR – регистр начального адреса буфера

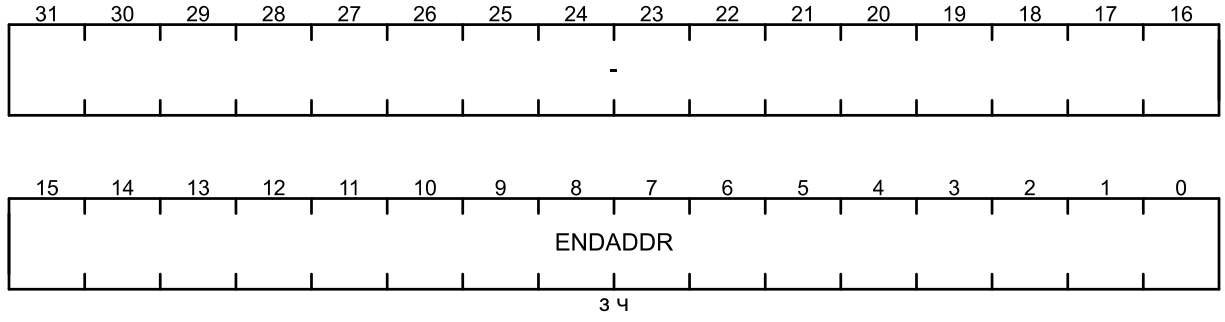
Смещение: EPn + 20h
Сброс: 0h



Поле	Биты	Описание
STARTADDR	15-0	Начальный адрес буфера конечной точки n
-	31-16	Зарезервировано

USB_EP_END_ADDR – регистр конечного адреса управляющего буфера

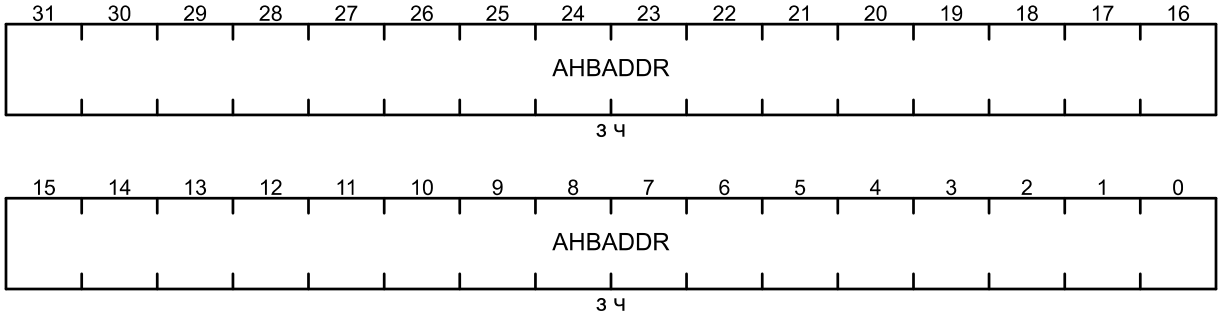
Смещение: EPn + 24h
Сброс: 0h



Поле	Биты	Описание
ENDADDR	15-0	Конечный адрес буфера конечной точки n
-	31-16	Зарезервировано

AHB_DMA_ADDR – регистр адреса записи и чтения в режиме DMA

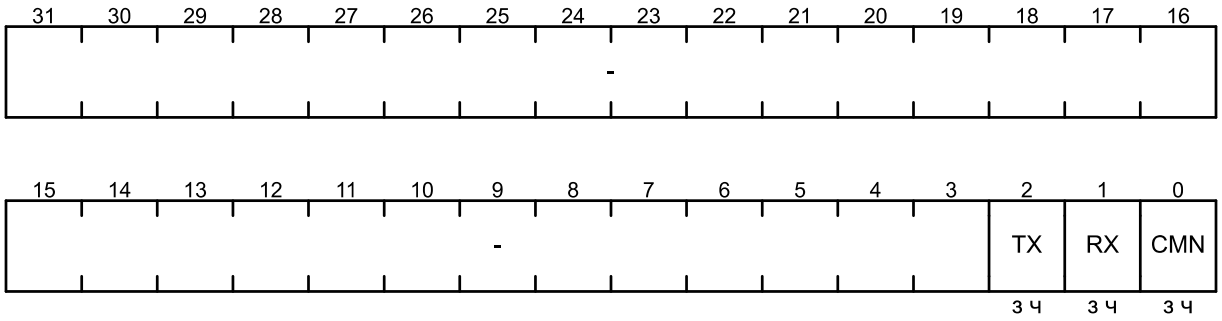
Смещение: + 700h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Адрес DMA

PHY_PD – регистр управления PowerDown PHY

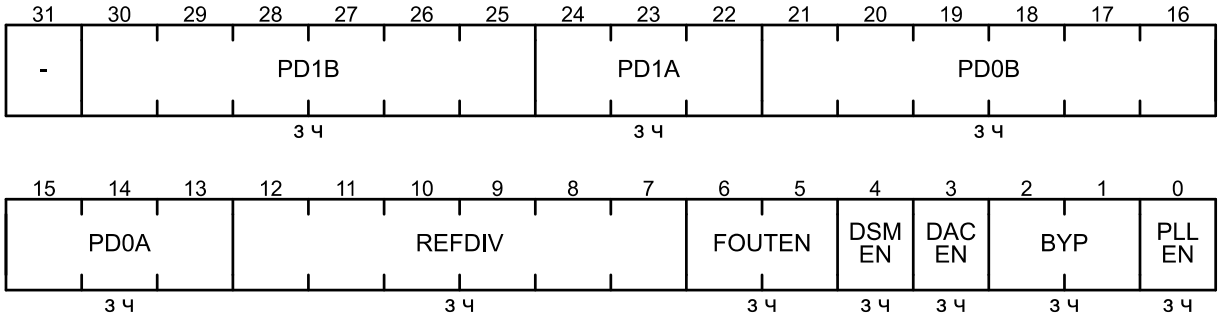
Смещение: + 7C0h
Сброс: 0h



Поле	Биты	Описание
TX	2	Бит включения режима Powerdown для передающей логики
RX	3	Бит включения режима Powerdown для принимающей логики
CMN	0	Бит включения режима Powerdown для цифровой части USB PHY
-	31-3	Зарезервировано

PLLUSBCFG0 – регистр 0 конфигурации PLL

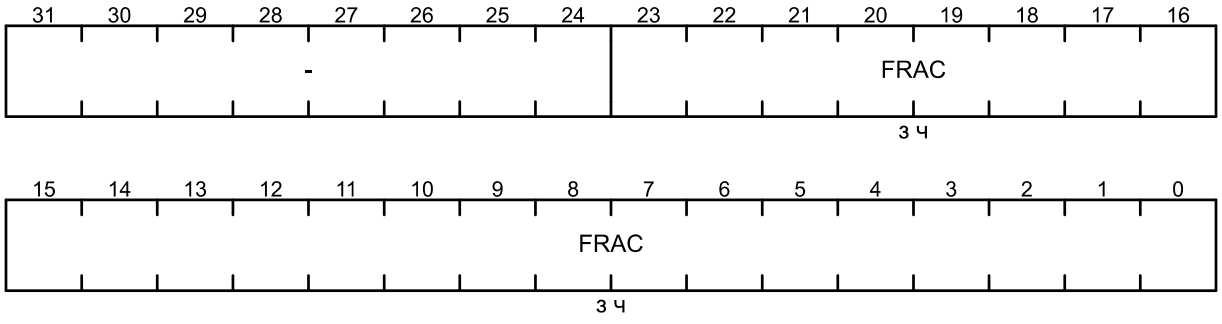
Смещение: + 800h
Сброс: 0h



Поле	Биты	Описание
PD1B	30-25	Зарезервировано
PD1A	24-22	Зарезервировано
PD0B	21-16	Коэффициент деления PD0B. Значения от 1 до 63
PD0A	15-13	Коэффициент деления PD0A. Значения от 1 до 7
REFDIV	12-7	Коэффициент деления входной частоты REFDIV. Значения от 1 до 63
FOUTEN	6-5	Бит разрешения выхода PLL
DSMEN	4	Бит включения дельта-сигма модулятора
DACEN	3	Бит включения ЦАП
BYP	1	Бит разрешения сквозного прохождения FREF на выход PLL
PLLEN	0	Бит включения PLL
–	19, 15	Зарезервировано

PLLUSBCFG1 – регистр 1 конфигурации PLL

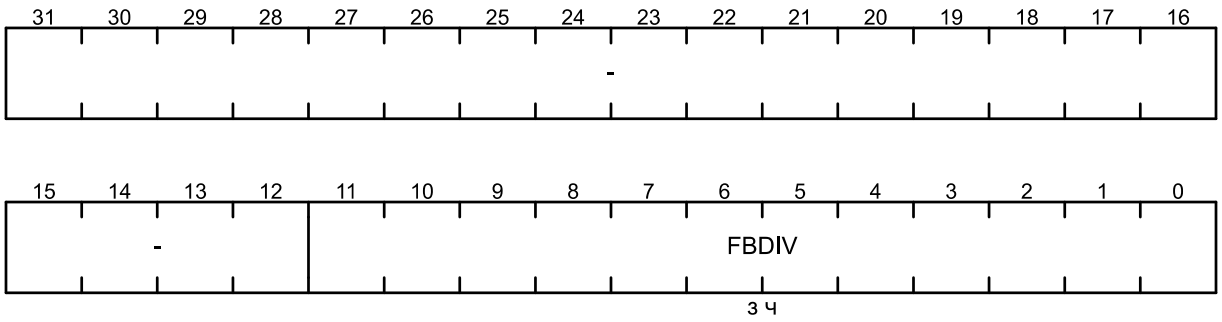
Смещение: + 804h
Сброс: 0h



Поле	Биты	Описание
FRAC	23-0	Значение дробного делителя
-	31-24	Зарезервировано

PLLUSBCFG2 - регистр 2 конфигурации PLL

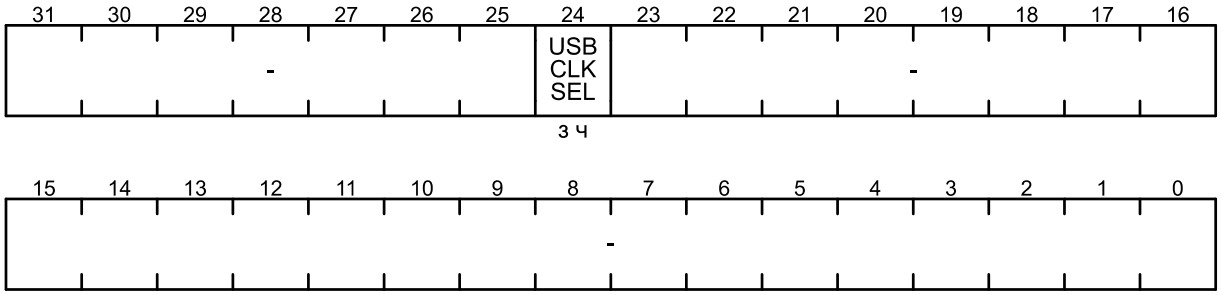
Смещение: + 808h
Сброс: 0h



Поле	Биты	Описание
FBDIV	11-0	Значение целочисленного делителя
-	31-12	Зарезервировано

USBCLKCFG - регистр выбора тактового сигнала

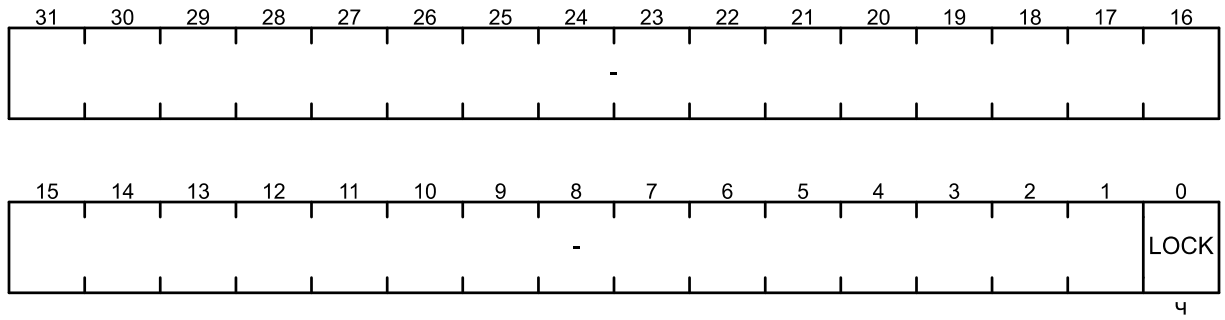
Смещение: + 80Ch
Сброс: 0h



Поле	Биты	Описание	
USBCLKSEL	24	Выбор источника тактового сигнала USB	
		0	PLLUSB0CLK
		1	SYSCLK
-	31-25, 23-0	Зарезервировано	

PLLUSBSTAT - регистр статуса PLL

Смещение: + 810h
Сброс: 0h



Поле	Биты	Описание
LOCK	0	Устанавливается, если выходная частота блока PLLUSB стабильна
-	31-1	Зарезервировано

Приложение Б
(обязательное)
Коды состояний функционирования блока I2C

В таблицах Б.1 – Б.11 представлена информация о соответствии кодов и операций.

Условные обозначения, принятые в данных таблицах:

- [ADR, 0], [ADR, 1] – 8-разрядное значение, состоящее из 7-разрядного адреса ADR и бита направления передачи R/W#, значение которого «0» или «1» указывается непосредственно;
- DAT – байт данных;
- код мастера – 8-разрядное значение 0000_1xxxb, где «xxx» – уникальный код каждого мастера в системе нескольких устройств;
- «с ACK» – выражение, обозначающее, что после передачи адреса/байта в ответ на запрос подтверждения передачи (бит ACK) передатчик получает подтверждение передачи от ведомого (квитирование);
- «с NACK» – выражение, обозначающее, что после передачи адреса/байта в ответ на запрос подтверждения передачи (бит ACK) передатчик получает неподтверждение передачи от ведомого (неквитирование);
- X – бит может быть установленным (1b) или сброшенным (0b), в зависимости от режима работы, состояния и дальнейших действий модуля I2C.

Таблица Б.1 – Исключительные состояния

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
00h	IDLE	–	–	–	–	–	Ожидать завершения текущей передачи байта
1Fh	Ошибка на шине	–	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)

Таблица Б.2 – Режим FS мастера передатчика (дополнительно см. таблицу Б.4)

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1	2	3	4	5	6	7	8
01h	Старт	Код мастера	1	0	0	0	Передать код мастера и перейти в режим HS (0Ch/21h)
		[ADR, 0]					Передать адрес ведомого (04h/05h)
02h	Повторный старт	[ADR, 0]	1	0	0	0	Передать адрес ведомого (04h/05h)
		[ADR, 1]					Передать адрес ведомого, после чего перейти в режим приемника (08h/09h)

Продолжение таблицы Б.2

1	2	3	4	5	6	7	8
03h	Потеря арбитража, мастер перешел в режим безадресного ведомого	–	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
04h	Отправлен адрес ведомого с ACK	DAT	1	0	0	0	Передать байт данных (06h/07h)
		–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
05h	Отправлен адрес ведомого с NACK	–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
06h	Отправлен байт данных с ACK	DAT	1	0	0	0	Передать байт данных (06h/07h)
		–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
07h	Отправлен байт данных с NACK	–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)

Таблица Б.3 – Режим FS мастера приемника

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1	2	3	4	5	6	7	8
08h	Отправлен адрес ведомого с ACK	–	1	0	0	0	Получить байт данных, квитировать прием (0Ah)
			1	1	0	0	Получить байт данных, не квитировать прием (0Bh)
09h	Отправлен адрес ведомого с NACK	–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
0Ah	Принят байт данных и квити-рован	DAT	1	0	0	0	Получить байт данных, квитировать прием (0Ah)
			1	1	0	0	Получить байт данных, не квитировать прием (0Bh)

Продолжение таблицы Б.3

1	2	3	4	5	6	7	8
0Bh	Принят байт данных и не квити-рован	DAT	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)

Таблица Б.4 – Режим FS мастера передатчика (дополнительно см. таблицу Б.2)

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
0Ch	Отправлен код мастера, обнаружена ошибка (ACK)	–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)

Таблица Б.5 – Режим FS ведомого приемника (дополнительно см. таблицу Б.7)

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
10h	Принят адрес и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)
11h	Принят адрес пос-лепотери арбитража и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)
12h	Принят байт данных и квити-рован	DAT	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)
13h	Принят байт данных и не квити-рован	DAT	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	0	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)

Таблица Б.6 – Режим FS ведомого передатчика

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1	2	3	4	5	6	7	8
14h	Принят адрес и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/ не квитировать (16h/17h)
15h	Принят адрес после потери арбитража и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/ не квитировать (16h/17h)
16h	Отправлен байт данных с ACK	DAT	1	X	0	0	Передать байт данных, квитировать/ не квитировать (16h/17h)
17h	Отправлен байт данных с NACK	–	1	X	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	X	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)
18h	Принят адрес отклика и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/ не квитировать (1Ah/1Bh)
19h	Принят адрес отклика после потери арбитража и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/ не квитировать (1Ah/1Bh)
1Ah	Отправлен байт данных в ответ на получение адреса отклика с ACK	DAT	1	X	0	0	Передать байт данных, квитировать/ не квитировать (1Ah/1Bh)

Продолжение таблицы Б.6

1	2	3	4	5	6	7	8
1Bh	Отправлен байт данных в ответ на получение адреса отклика с NACK	–	1	X	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	X	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)

Таблица Б.7 – Режим FS ведомого приемника (дополнительно см. таблицу Б.5)

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
1Ch	Стоп	–	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	0	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)
1Dh	Принят адрес общего вызова и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)
1Eh	Принят адрес общего вызова после потери арбитража и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (12h)
			1	1	0	0	Получить байт данных, не квитировать прием (13h)

Таблица Б.8 – Режим HS мастера передатчика

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
21h	Успешно отправлен код мастера, мастер перешел в режим HS	–	1	0	0	1	Сделать повторный старт (22h)
22h	Повторный старт	[ADR, 0]	1	0	0	0	Передать адрес ведомого (28h/29h)
		[ADR, 1]					Передать адрес ведомого, после квитирования/не квитирования переключиться в режим мастера приемника (28h/29h)
23h	Потеря арбитража, мастер перешел в режим HS безадресного ведомого	–	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
24h	Отправлен адрес ведомого с ACK	DAT	1	0	0	0	Передать байт данных (26h/27h)
		–	1	0	0	1	Сделать повторный старт (22h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
25h	Отправлен адрес ведомого с NACK	–	1	0	0	1	Сделать повторный старт (22h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
26h	Отправлен байт данных с ACK	DAT	1	0	0	0	Передать байт данных (26h/27h)
		–	1	0	0	1	Сделать повторный старт (22h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
27h	Отправлен байт данных с NACK	–	1	0	0	1	Сделать повторный старт (22h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)

Таблица Б.9 – Режим HS мастера приемника

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
28h	Отправлен адрес ведомого с ACK	–	1	0	0	0	Получить байт данных, квитировать прием (2Ah)
			1	1	0	0	Получить байт данных, не квитировать прием (2Bh)
29h	Отправлен адрес ведомого с NACK	–	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)
2Ah	Принят байт данных и квити-рован	DAT	1	0	0	0	Получить байт данных, квитировать прием (2Ah)
			1	1	0	0	Получить байт данных, не квитировать прием (2Bh)
2Bh	Принят байт данных и не квити-рован	DAT	1	0	0	1	Сделать повторный старт (02h)
			1	0	1	0	Остановить передачу (00h)
			1	0	1	1	Остановить передачу, а затем сделать повторный старт (01h)

Таблица Б.10 – Режим HS ведомого приемника

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
30h	Принят адрес и квити-рован	–	1	0	0	0	Получить байт данных, квитировать прием (32h)
			1	1	0	0	Получить байт данных, не квитировать прием (33h)
32h	Принят байт данных и квити-рован	DAT	1	0	0	0	Получить байт данных, квитировать прием (32h)
			1	1	0	0	Получить байт данных, не квитировать прием (33h)
33h	Принят байт данных и не квити-рован	DAT	1	0	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	0	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)

Таблица Б.11 – Режим HS ведомого передатчика

Код	Описание состояния	Регистр SDA	Биты регистра CTL0				Возможные дальнейшие действия и коды результатов их выполнения
			CLRST	ACK	STOP	START	
34h	Принят адрес и квити-рован	DAT	1	X	0	0	Передать байт данных, квитировать/ не квитировать (36h/37h)
36h	Отправлен байт данных с ACK	DAT	1	X	0	0	Передать байт данных, квитировать/ не квитировать (36h/37h)
37h	Отправлен байт данных с NACK	–	1	X	0	0	Функционировать в режиме безадресного ведомого (00h)
			1	X	0	1	Функционировать в режиме безадресного ведомого, сделать старт после освобождения шины (00h, 01h)

Приложение В
(обязательное)
Токи потребления микроконтроллера

Значения, представленные в таблицах В.1 и В.2, были получены на макетных образцах, имеющих особенности, представленные в ERRATA, и будут скорректированы при измерениях опытных образцов.

Измерения параметров проводились при Fhse = 16 МГц, VCC1 = 3,3 В, VCC2 = 3,3 В, T = 25°C. Описания режимов работы 2 – 6, представленных в таблицах В.1 и В.2, представлены в подразделе 5.3.

Таблица В.1 – Токи потребления микроконтроллера при работе всех периферийных блоков

№ п/п	Режим работы	Ток потребления, мА				Описание
		HSI (1МГц)	HSE (16МГц)	PLL (50МГц)	LSI (32КГц)	
1	RUN	4,9000	11,4400	25,0000	4,9000	Все работает
2	IDLE	4,8800	6,4500	10,7800	4,8800	Команда WFI
3	LP(ULP)_IDLE	3,5400	5,0200	5,7100	3,5400	В режиме LowPower LDO0 (RTC_CFG1_bit. LDOHV = 0x19 (1.3В), RTC_CFG1_bit.LDOLV = 0x16 (1.24В)), команда WFI
4	STOP	1,5700	1,5700	1,5700	1,5700	Отключение LDO1, команда WFI
5	LP(ULP)_STOP	1,5200	1,5100	1,5100	1,5100	Отключение: LDO1, в режиме LowPower: LDO0 (RTC_CFG1_bit. LDOHV = 0x19 (1.3В), RTC_CFG1_bit.LDOLV = 0x16 (1.24В)), команда WFI
6	POWEROFF	0,0057*	0,0057*	0,0057*	0,0057*	Отключение: LDO0 и LDO1, команда WFI

Таблица В.2 – Токи потребления микроконтроллера при отключенных периферийных блоках

№ п/п	Режим работы	Ток потребления, мА		Описание
		HSI (1МГц)	LSI (32КГц)	
1	RUN	3,0500	2,9600	Все работает
2	IDLE	1,450	1,3500	Команда WFI
3	LP(ULP)_IDLE	1,2500	1,0800	В режиме LowPower LDO0 (RTC_CFG1_bit. LDOHV = 0x19 (1.3В), RTC_CFG1_bit.LDOLV = 0x16 (1.24В)), команда WFI
4	STOP	0,0730	0,0730	Отключение LDO1, команда WFI
5	LP(ULP)_STOP	0,0165	0,0165	Отключение: LDO1, в режиме LowPower: LDO0 (RTC_CFG1_bit. LDOHV = 0x19 (1.3В), RTC_CFG1_bit.LDOLV = 0x16 (1.24В)), команда WFI
6	POWEROFF	0,0057*	0,0057*	Отключение: LDO0 и LDO1, команда WFI
Примечание – При измерениях параметров все периферийные блоки были отключены (в режиме PowerDown – внешний осциллятор EXTOSC(HSE)).				

Приложение Г
(обязательное)
Назначение выводов в корпусе