МИКРОСХЕМА ИНТЕГРАЛЬНАЯ

1921ВК035

Руководство пользователя

2020

Содержание

Введение

32-разрядные микроконтроллеры с каждым днем набирают все большую популярность среди разработчиков различного оборудования и программистов. Они применяются при разработке и изготовлении электронной техники. Высокая вычислительная мощность и при этом относительно низкая стоимость делают эти устройства привлекательными для самого широкого круга разработчиков.

Микросхема 1921ВК035 представляет собой СБИС 32-разрядного микроконтроллера на базе RISC-ядра, предназначенного для промышленных и потребительских приложений, включая системы дистанционного мониторинга, контрольно-измерительные приборы, сетевые устройства, системы автоматизации производственных процессов, автомобильную электронику, системы управления электродвигателями.

В состав микроконтроллера входит широкий набор цифровой и аналоговой периферии, в связи с чем, 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 руководстве пользователя приведено описание архитектуры, функционального построения и периферии микроконтроллера 1921ВК035.

1 Область применения и особенности ИС 1921ВК035

Сфера применения интегральной микросхемы 1921ВК035 довольно широка: средства измерений, связи, наблюдения, безопасности, автоматизация производства, медицины, энергетики, промышленности, различных систем управления и системы электропривода.

Для эффективного управления в электромеханических системах была разработана дополнительная периферия: блоки ШИМ, блок АЦП с интерфейсом к контроллеру прямого доступа к памяти, модуль захвата/сравнения, блок импульсного квадратурного декодера, используемого для обработки сигналов датчиков положения ротора в высокопроизводительных системах для определения положения, направления и скорости вращения.

Разработанный микроконтроллер имеет встроенную Flash-память программ размером 64 Кбайт, которую можно использовать для хранения и загрузки пользовательского программного обеспечения. Во Flash-памяти существует особая защищенная область, которая может быть использована для хранения начального загрузчика.

Система тактирования микроконтроллера позволяет использовать различные источники тактового сигнала, что позволяет расширить набор применений и решаемых задач пользователя. Микроконтроллер может тактироваться от внутреннего RC-генератора с частотой 8 МГц, внутреннего осциллятора с внешним кварцевым резонатором, а также сигналом синтезатор частоты PLL. 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 или WFE).

2 Краткое техническое описание микроконтроллера

2.1 Функциональные параметры

Структурная схема микроконтроллера показана на рисунке 2.1.

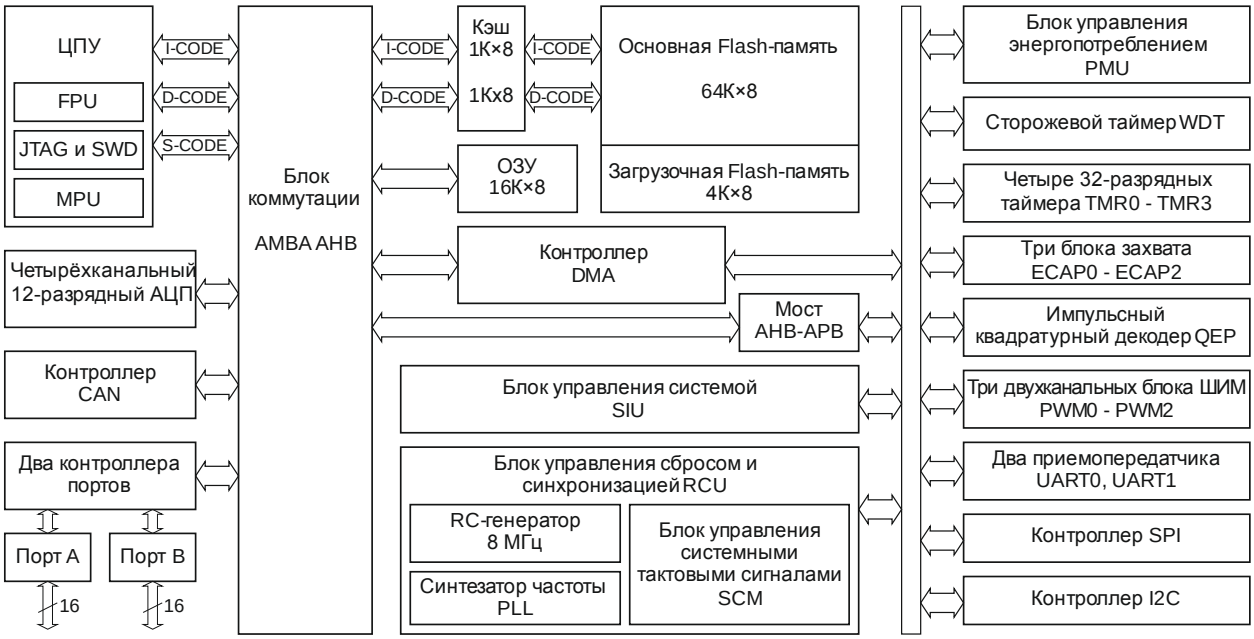


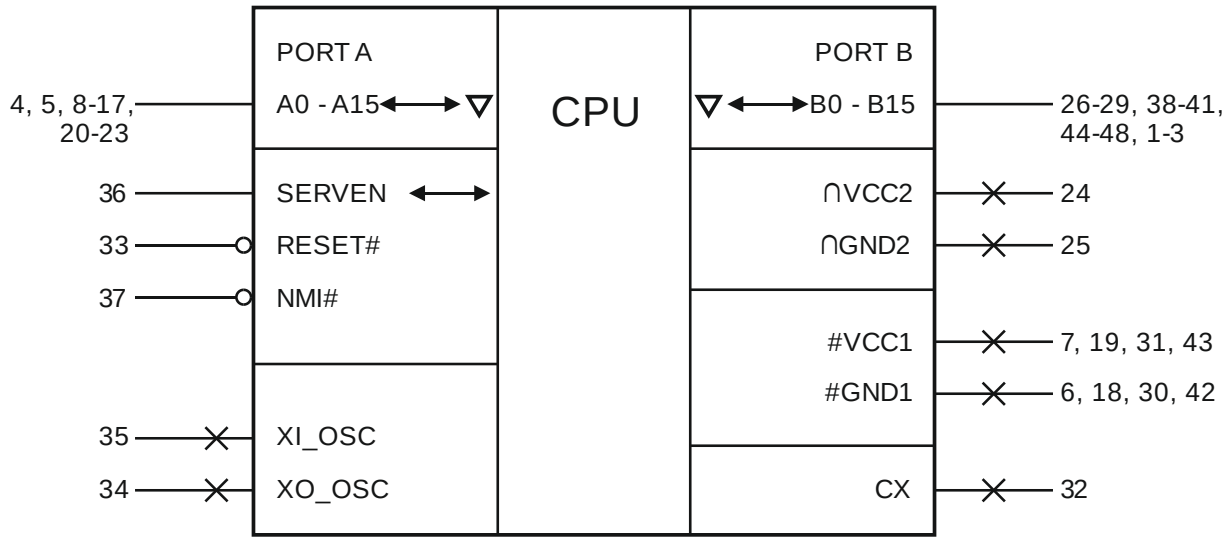
Рисунок 2.1 – Структурная схема микроконтроллера

В состав микроконтроллера входят функциональные элементы:

- 32-разрядное ЦПУ с поддержкой набора одноцикловых команд умножения с накоплением, команд централизованного управления потоком данных, арифметических и логических команд, встроенным модулем обработки команд с плавающей запятой с одинарной точностью FPU, поддержкой отладочных интерфейсов JTAG и SWD и модулем защиты памяти MPU;
- блок управления сбросом и синхронизацией RCU, имеющий в своем составе RC-генератор (8 Гц), синтезатор частоты PLL и блок управлнения системными тактовыми сигналами SCM;
- блок управления системой SIU;
- блок коммутации AMBA AHB;
- основная Flash-память объемом 64 Кбайт;
- загрузочная Flash-память емкостью 4 Кбайт;
- кэш команд и данных объемом 1 Кбайт каждый;
- ОЗУ объемом 16 Кбайт;
- 16-канальный контроллер прямого доступа к памяти DMA;
- блок управления энергопотреблением PMU, позволяющий переводить системные блоки в режим Powerdown;
- два контроллера портов A и B, управляющих 16-разрядными портами ввода-вывода;
- четырехканальный 12-разрядный АЦП с режимами цифрового компаратора для каждого из каналов (равно или больше, равно или меньше, попадание в диапазон, выход из диапазона) и функцией автоматического запуска модулей ШИМ по событию «окончание преобразования»;
- три двухканальных блока ШИМ PWM0 – PWM2;

- импульсный квадратурный декодер QEP для обработки сигналов датчиков положения ротора, позволяющий определить положение, направление и скорость вращения;
- три блока захвата ECAP0 – ECAP2;
- четыре 32-разрядных таймера TMR0 – TMR3;
- сторожевой таймер WDT;
- два приемопередатчика UART0, UART1;
- контроллер CAN (протокол 2.0b);
- контроллер I2C;
- контроллер SPI.

Условное графическое обозначение микроконтроллера приведено на рисунке 2.2.



Примечание – Альтернативные функции выводов порта А и порта В указаны в таблице 2.1.

Рисунок 2.2 – Условное графическое обозначение микросхемы

Назначение выводов

Функциональное назначение выводов микроконтроллера приведено в таблице 2.1. После сброса микроконтроллера выводы портов А и В конфигурируются как выводы общего назначения и находятся в третьем состоянии. Исключение составляют выводы А2 – А6: конфигурируются как выводы отладочного интерфейса JTAG. Включение/ отключение режима альтернативной функции осуществляется посредством регистров ALTFUNCSET/ALTFUNCCLR блоков GPIO. Выводы портов А и В имеют схемы «pull-up» и «pull-down», подключаемые программно (регистр PULLMODE блока GPIO). Выводы NMI#, RESET# имеют постоянную схему «pull-up», вывод SERVEN – постоянную схему «pull-down». При использовании отладочного интерфейса SWD его вывод SWCLK соответствует выводу JTAG_TCK, вывод SWDIO соответствует выводу JTAG_TMS, а вывод блока трассировки SWO соответствует выводу JTAG_TDO.

В таблице 2.1 используются условные обозначения: I – вход, O – выход, I/O – вход/выход, Z – третье состояние.

Микросхемы выполнены в металлокерамическом корпусе МК 5162.48-1.

Масса микросхемы – не более 1,0 г.

Таблица 2.1 – Функциональное назначение выводов ИС 1921ВК035

Обозначение вывода	Обозначение альтернативной функции вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
A0		4	I/O/Z	Вход/выход порта A, разряд 0
	I2C_SCL		I/O	Вход/выход синхросигнала I2C
A1		5	I/O/Z	Вход/выход порта A, разряд 1
	I2C_SDA		I/O	Вход/выход данных I2C
A2		8	I/O/Z	Вход/выход порта A, разряд 2
	JTAG_TCK		I	Вход тактового сигнала порта JTAG
A3		9	I/O/Z	Вход/выход порта A, разряд 3
	JTAG_TMS		I	Вход режима порта JTAG
A4		10	I/O/Z	Вход/выход порта A, разряд 4
	JTAG_TDI		I	Вход данных порта JTAG
	ECAP0_IO		I/O	Вход/выход блока захвата 0
A5		11	I/O/Z	Вход/выход порта A, разряд 5
	JTAG_TDO		O	Выход данных порта JTAG
	ECAP1_IO		I/O	Вход/выход блока захвата 1
A6		12	I/O/Z	Вход/выход порта A, разряд 6
	JTAG_TRST		I	Вход сброса порта JTAG
	ECAP2_IO		I/O	Вход/выход блока захвата 2
A7		13	I/O/Z	Вход/выход порта A, разряд 7
	PWM_TZ		I	Вход сигнала аварии блоков ШИМ
A8		14	I/O/Z	Вход/выход порта A, разряд 8
	PWM0_A		O	Выход линии A блока ШИМ0
A9		15	I/O/Z	Вход/выход порта A, разряд 9
	PWM0_B		O	Выход линии B блока ШИМ0
A10		16	I/O/Z	Вход/выход порта A, разряд 10
	PWM1_A		O	Выход линии A блока ШИМ1
A11		17	I/O/Z	Вход/выход порта A, разряд 11
	PWM1_B		O	Выход линии B блока ШИМ1
A12		20	I/O/Z	Вход/выход порта A, разряд 12
	PWM2_A		O	Выход линии A блока ШИМ2
A13		21	I/O/Z	Вход/выход порта A, разряд 13
	PWM2_B		O	Выход линии B блока ШИМ2
A14		22	I/O/Z	Вход/выход порта A, разряд 14
	TMR0_IN		I	Вход внешней синхронизации таймера 0
A15		23	I/O/Z	Вход/выход порта A, разряд 15
	TMR1_IN		I	Вход внешней синхронизации таймера 1
B0		26	I/O/Z	Вход/выход порта B, разряд 0
	ADC_CH0		I	Вход АЦП, разряд 0
B1		27	I/O/Z	Вход/выход порта B, разряд 1
	ADC_ CH1		I	Вход АЦП, разряд 1
B2		28	I/O/Z	Вход/выход порта B, разряд 2
	ADC_ CH2		I	Вход АЦП, разряд 2

Окончание таблицы 2.1

Обозначение вывода	Обозначение альтернативной функции вывода	Номер вывода	Тип вывода	Функциональное назначение вывода
B3		29	I/O/Z	Вход/выход порта B, разряд 3
	ADC_ CH3		I	Вход АЦП, разряд 3
B4		38	I/O/Z	Вход/выход порта B, разряд 4
	SPI_FSS		I/O	Вход/выход выбора ведомого устройства SPI
B5		39	I/O/Z	Вход/выход порта B, разряд 5
	SPI_SCK		I/O	Вход/выход синхросигнала SPI
B6		40	I/O/Z	Вход/выход порта B, разряд 6
	SPI_RX		I	Вход данных SPI
B7		41	I/O/Z	Вход/выход порта B, разряд 7
	SPI_TX		O	Выход данных SPI
B8		44	I/O/Z	Вход/выход порта B, разряд 8
	UART1_TX		O	Выход данных блока UART1
B9		45	I/O/Z	Вход/выход порта B, разряд 9
	UART1_RX		I	Вход данных блока UART1
B10		46	I/O/Z	Вход/выход порта B, разряд 10
	UART0_TX		O	Выход данных блока UART0
	QEP_S		I/O	Вход/выход стробирования квадратурного декодера
B11		47	I/O/Z	Вход/выход порта B, разряд 11
	UART0_RX		I	Вход данных блока UART0
	QEP_I		I/O	Индексный вход/выход квадратурного декодера
B12		48	I/O/Z	Вход/выход порта B, разряд 12
	CAN1_RX		I	Вход данных CAN1
	QEP_B		I	Вход B квадратурного декодера
B13		1	I/O/Z	Вход/выход порта B, разряд 13
	CAN1_TX		O	Выход данных CAN1
	QEP_A		I	Вход A квадратурного декодера
B14		2	I/O/Z	Вход/выход порта B, разряд 14
	CAN0_RX		I	Вход данных CAN0
B15		3	I/O/Z	Вход/выход порта B, разряд 15
	CAN0_TX		O	Выход данных CAN0
CX	–	32	–	Вывод регулятора напряжения питания для подключения внешнего конденсатора 4,7 мкФ
SERVEN	–	36	I/ O	Вход выбора сервисного режима/ Выход программируемого тактового сигнала CLKOUT
NMI#	–	37	I	Вход внешнего немаскируемого прерывания (активный ноль)
RESET#	–	33	I	Вход внешнего сброса (активный ноль)
XI_OSC	–	35	–	Вывод внешнего тактового сигнала/ Вывод для подключения кварцевого резонатора
XO_OSC	–	34	–	Вывод для подключения кварцевого резонатора
∩VCC2	–	24	–	Аналоговое питание 3,3 В
∩GND2	–	25	–	Аналоговая земля
#VCC1	–	7, 19, 31, 43	–	Цифровое питание 3,3 В
#GND1	–	6, 18, 30, 42	–	Цифровая земля

Особенности системы питания

На плате вывод питания ∩VCC2 может быть объединен с выводами питания #VCC1 (при этом должны быть приняты меры для снижения помех по питанию).

Выключение питания микроконтроллера должно проводиться путем полного снятия напряжения, как с выводов питания, так и со всех остальных выводов микросхемы. Подача напряжения на функциональные выводы микросхемы при выключенном питании недопустима.

2.2 Электрические параметры

Номинальное значение напряжения питания по выводам #VCC1, ∩VCC2 должно быть 3,3 В ± 0,3В.

Значение суммарного максимального тока по выводам портов A и B не должно превышать 50 мА.

Допустимая максимальная разность значений напряжений питания по выводам #VCC1, ∩VCC2 составляет 50 мВ.

Амплитудное значение пульсации напряжения питания должно быть не более 50 мВ.

Электрические параметры микросхем при приемке и поставке соответствуют нормам, приведенным в таблице 2.2.

Таблица 2.2 – 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 по выводам A0 – A15, B0 – B15, В, U_{CC1} = 3,0 В, I_{OL} = 6 мА	U_{OL}	–	0,4	−60 ± 3 25 ± 10 125 ± 5
2 Выходное напряжение высокого уровня по выводам A0 – A15, B0 – B15, В, U_{CC1} = 3,0 В, I_{OH} = –6 мА	U_{OH}	U_{CC1} – 0,9	–	
3 Ток утечки низкого уровня по входам A0 – A15, B0 – B15 с отключенными «pull-up» и «pull-down», мкА, U_{CC1} = 3,6 В, U_{IL} = 0 В	I_{ILL}	–10	–	
4 Ток утечки высокого уровня по входам A0 – A15, B0 – B15 с отключенными «pull-up» и «pull-down», мкА, U_{CC1} = 3,6 В, U_{IH} = U_{CC1}	I_{ILH}	–	10	
5 Входной ток низкого уровня по выводам «pull-up» A0 – A15, B0 – B15, NMI#, RESET#, мкА U_{CC1} = 3,6 В, U_{IL} = 0 В	I_{IL1}	–200	–	
6 Входной ток высокого уровня по выводам «pull-up» NMI#, RESET#, мкА, U_{CC1} = 3,6 В, U_{IH} = U_{CC1}	I_{IH1}	–	10	
7 Входной ток низкого уровня по выводу «pull-down» SERVEN, мкА, U_{CC1} = 3,6 В, U_{IL} = 0 В	I_{IL2}	–10	–	
8 Входной ток высокого уровня по выводам «pull-down» A0 – A15, B0 – B15, SERVEN, мкА, U_{CC1} = 3,6 В, U_{IH} = U_{CC1}	I_{IH2}	–	200	
9 Входной ток низкого уровня по выводу XI_OSC, мкА, U_{CC1} = 3,6 В, U_{IL} = 0 В	I_{IL3}	–40	–	

Окончание таблицы 2.2

1	2	3	4	5
10 Входной ток высокого уровня по выводу XI_OSC, мкА, U_{CC1} = 3,6 В, $U_{IH} = U_{CC1}$	I_{IH3}	–	40	
11 Динамический ток потребления по выводам #VCC1 в активном режиме, мА, U_{CC1} = 3,6 В, U_{CC2} = 3,6 В, f_{CI} = 100 МГц	I_{OCC1}	–	150	
12 Интегральная нелинейность АЦП, EMP, U_{CC1} = 3,6 В, U_{CC2} = 3,6 В	E_L	–3	3	−60 ± 3 25 ± 10 125 ± 5
13 Дифференциальная нелинейность АЦП, EMP, U_{CC1} = 3,6 В, U_{CC2} = 3,6 В	E_{LD}	–2	2	
14 Функциональный контроль, U_{CC1} = (3,0; 3,6) В, U_{CC2} = (3,0; 3,6) В, f_{CI} = (0,001; 100) МГц	ФК	–	–	

Примечания

1 Параметры I_{ILL}, I_{ILH}, I_{IH1}, I_{IL2} при температуре минус 60 ºС не измеряются, а гарантируются нормами при температуре (25 ± 10) ºС.

2 При функциональном контроле АЦП значения напряжений питания изменяются синхронно.

Значения предельно допустимых электрических режимов эксплуатации в диапазоне рабочих температур среды соответствуют нормам, приведенным в таблице 2.3.

Таблица 2.3 – Значения предельно допустимых электрических режимов эксплуатации микросхем в диапазоне температур среды

Наименование параметра режима, единица измерения		Буквенное обозначение параметра	Предельно допустимый режим		Предельный режим	
			не менее	не более	не менее	не более
1 Напряжение питания цифровой части, В[1)]		U_{CC1}	3,0	3,6	–0,3	5,2
2 Напряжение питания аналоговой части, В[1)]		U_{CC2}	3,0	3,6	–0,3	5,2
3 Входное напряжение низкого уровня, В[1)]		U_{IL}	–0,5	0,8	–0,6	–
4 Входное напряжение высокого уровня, В[1)]		U_{IH}	$0{,}2U_{CC1}$ +1	U_{CC1}	–	U_{CC1} + 0,6
5 Входное напряжение высокого уровня по выводу XI_OSC, В[1)]		U_{IHC}	$0{,}7U_{CC1}$	U_{CC1}	–	U_{CC1} + 0,6
6 Выходной ток низкого уровня, мА[1)]		I_{OL}	–	6	–	10
7 Выходной ток высокого уровня, мА[1)]		I_{OH}	–6	–	–10	–
8 Частота следования импульсов тактового сигнала процессорного ядра, МГц		f_{CI}	0,001	100	–	–
9 Частота следования импульсов тактового сигнала по выводу XI_OSC, МГц	при работе с внешним тактовым генератором	f_C	0,001	40	–	–
	при работе с кварцевым резонатором		8	24	–	–
10 Емкость нагрузки, пФ		C_L	–	40	–	–

[1)] Время работы в одном из предельных режимов должно быть не более 5 с.

3 Архитектура изделия

Микроконтроллер 1921ВК035 структурно представляет собой мультистадийный RISC процессор. Ядро полностью реализует наборы команд Thumb и Thumb2.

Поддержка DSP-инструкций и наличие модуля операций с плавающей запятой существенно ускоряет обработку потоковых данных, что в свою очередь делает микроконтроллер весьма привлекательным для использования в системах управления и обработки информации.

Микроконтроллер способен параллельно выполнять четыре операции сложения/вычитания с 8-разрядными операндами или две операции сложения/вычитания с 16-разрядными операндами. Также реализовано умножение за один цикл, при этом для 16-разрядных чисел возможно параллельное исполнение двух операций. Из особенностей следует отметить аппаратное умножение 32-разрядных чисел за 1 цикл, а также деление 32-разрядных чисел, занимающее от 2 до 12 циклов.

Блок коммутации микроконтроллера

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

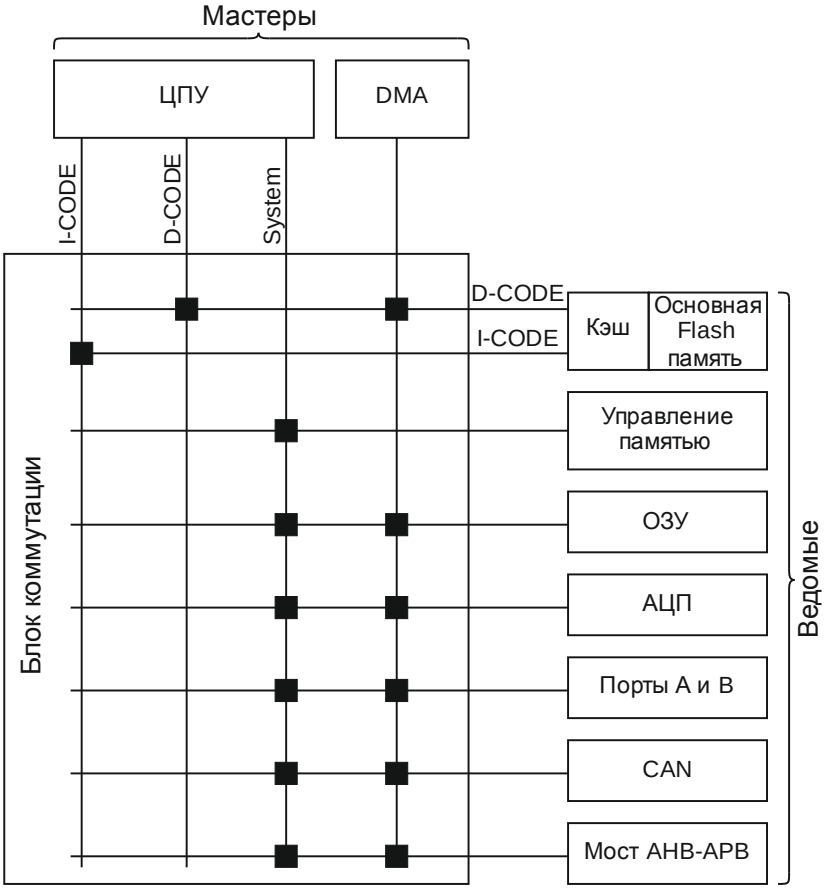


Рисунок 3.1 – Схема соединения блоков внутри блока коммутации

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:

- OSICLK – тактовый сигнал внутреннего RC-генератора;
- OSECLK – внешний тактовый сигнал по выводам XI_OSC и XO_OSC;
- PLLCLK – тактовый сигнал с выхода PLL;
- PLLDIVCLK – тактовый сигнал с выхода PLL, прошедший через дополнительный делитель;
- SYSCLK – системный тактовый сигнал, определяющий частоту работы процессорного ядра ($f_{SYSCLK} = f_{CI}$);
- PCLK – периферийный тактовый сигнал APB шины;
- HCLK – системный тактовый сигнал AHB шины;
- FCLK – неотключаемый системный тактовый сигнал.

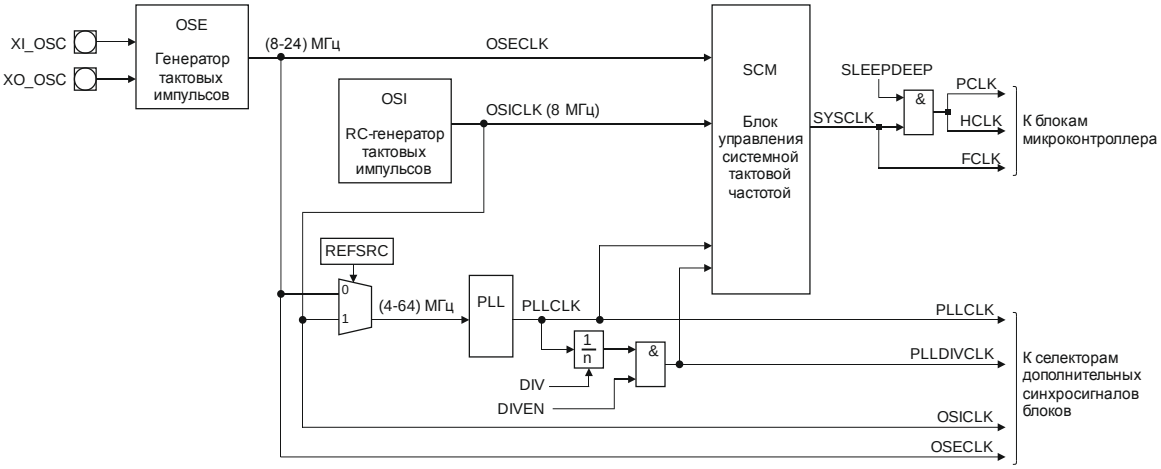


Рисунок 4.1 – Схема синхронизации

4.2 Синтезатор частоты PLL

В качестве опорного сигнала блока PLL можно выбрать один из источников, выбор осуществляется полем REFSRC регистра PLLCFG.

Выходная частота блока PLL определяется согласно формуле

$$f_{OUT} = f_{IN} \times \frac{M}{N} \times \frac{1}{2^{OD}} \quad (4.1)$$

Коэффициенты N, M, OD задаются соответствующими полями регистра PLLCFG. Существуют следующие ограничения параметров:

- коэффициент деления $1 \leq N \leq 63$;
- коэффициент деления $2 \leq M \leq 63$;
- входная частота должна находиться в диапазоне (4 – 64) МГц;
- значение частоты f_{IN}/N должно быть в диапазоне (4 – 20) МГц;
- значение частоты $f_{IN} \times M/N$ должно быть в диапазоне (120 – 200) МГц;
- значение выходной частоты f_{OUT} должно быть в диапазоне (15 – 200) МГц.

Настройку блока PLL необходимо делать до того, как его тактовый сигнал будет выбран в качестве рабочей частоты системы или одного из блоков. При правильной установке всех значений и выходе блока PLL на рабочий режим, будет установлен бит LOCK в регистре PLLCFG. После установления рабочего режима можно разрешить выход частоты из блока PLL битом OUTEN. Также есть возможность прохождения сигнала сквозь блок PLL без изменений (bypass режим), для установления которого можно воспользоваться битом BYPASS.

Также присутствует внешний делитель частоты, управляемый битами регистра PLLDIV. Для тактирования различных блоков и модулей микроконтроллера можно выбрать как сигнал PLLCLK, так и PLLDIVCLK, 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 xOK и xERR регистра SYSCLKSTAT, где x – OSECLK, PLLCLK, PLLDIVCLK. Бит xOK будет установлен, если соответствующий тактовый сигнал стабильно работает и будет сброшен при его сбое. Бит xERR 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 OSICLK. Реакция на любое пропадание тактового сигнала на меньшее время будет отсутствовать.

Минимальная частота, за которой может осуществляться слежение – 20 кГц.

Значение минимально возможного времени реакции вычисляется по формуле (4.2) (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 OSICLK;

T_{CLK} – период контролируемой тактовой частоты.

Например, если необходима максимально быстрая реакция на пропадание частоты 100 МГц на выходе блока PLL (T_{CLK} = 10 нс) при сигнале OSICLK 8 МГц (T_{REF} = 125 нс), то необходимо в поле PLLCLK регистра SECPRD0 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 OSICLK) займет дополнительно не более 14 тактов сигнала OSICLK.

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 OSICLK, и будет выработано немаскируемое прерывание NMI# совместно с установкой флага прерывания SYSFAIL в регистре INTSTAT (флаг сбрасывается записью единицы).

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 OSICLK, 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 OSICLK), 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 xOK для него будет установлен), например, на сигнал OSICLK (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 как внешним сигналом с вывода RESET# (активный уровень сигнала сброса – низкий), так и с помощью внутренних источников – по запросу процессора и по переполнению сторожевого таймера.

Также в микроконтроллере есть возможность сброса по получению сигнала LOCKUP от ядра. Данный сигнал возникает в случае невосстановимого системного исключения (Unrecoverable exception), если микроконтроллер не может из него выйти. Установка бита LOCKUPEN в регистре SYSRSTCFG разрешит автоматический сброс после получения от ядра сигнала LOCKUP.

В регистре SYSRSTSTAT содержится набор флагов, по которым можно установить причину последнего произошедшего сброса. В случае нормальной загрузки после подачи питания будет установлен флаг POR, так как сброс вызван внешним сигналом.

4.5 Прерывания

В микроконтроллере есть два прерывания, связанных с работой блока RCU.

Первое прерывание – непосредственно RCU Interrupt. Выбрать источники, влияющие на него можно в регистре INTEN – события появления или пропадания одного из тактовых сигналов, выход блока PLL в рабочий режим. Все события сопровождаются флагами в регистре INTSTAT.

Второе прерывание связано с пропаданием текущего системного тактового сигнала SYSCLK. В случае если была включена система слежения за тактовым сигналом (бит SECEN регистра SYSCLKCFG) возникнет прерывание, соответствующее вектору немаскируемого прерывания NMI#, и будет установлен соответствующий флаг в регистре INTSTAT.

4.6 Тактирование и сброс периферийных блоков

Для некоторых периферийных блоков система тактирования предусматривает выбор независимого тактового источника. У блоков WATCHDOG, TRACE, UART, SPI, ADC 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 4.2.

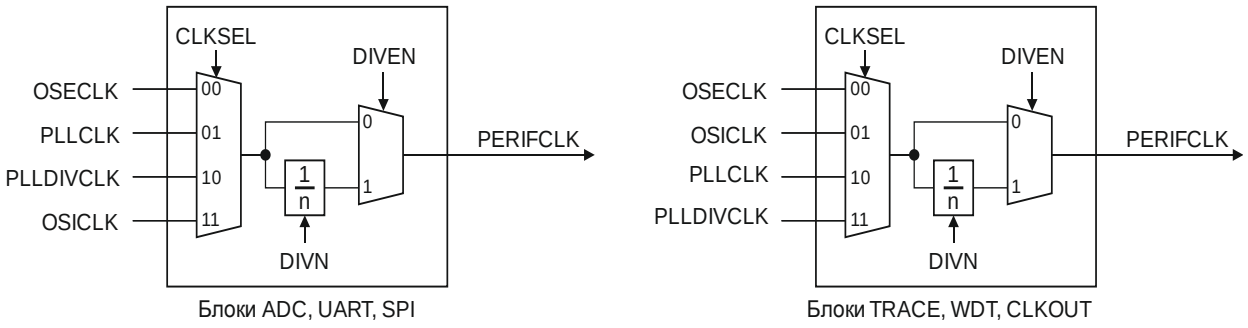


Рисунок 4.2 – Схемы селекторов дополнительных синхросигналов блоков

Для остальных блоков, а именно: CAP, TMR, I2C, QEP, PWM, и для блоков, относящихся к AHB периферии (GPIO, CAN), тактирование подается установкой соответствующего бита в CLKCFG, см. таблицу 4.1.

Таблица 4.1 – Тактовые сигналы периферийных блоков

Периферийный блок	Тактовый сигнал	Делитель	Управляющий регистр
ADC	ACLK	1-63	ADCCFG
SPI	SPICLK	1-63	SPICFG
UART	UARTCLK	1-63	UARTCFG
WATCHDOG	WDTCLK	1-63	WDTCFG
CLOCKOUT	CLKOUT	1-7	CLKOUTCFG
TRACE	TRACECLK	1-63	TRACECFG
TIMER	PCLK	-	PCLKCFG, PRSTCFG
PWM	PCLK	-	PCLKCFG, PRSTCFG
I2C	PCLK	-	PCLKCFG, PRSTCFG
QEP	PCLK	-	PCLKCFG, PRSTCFG
CAP	PCLK	-	PCLKCFG, PRSTCFG
GPIO	HCLK	-	HCLKCFG, HRSTCFG
CAN	HCLK	-	HCLKCFG, HRSTCFG

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 управления.

5 Блок управления энергопотреблением PMU

В микроконтроллере предусмотрены несколько режимов функционирования для уменьшения потребляемой энергии в тех задачах, где ядро не должно постоянно работать, например, во время ожидания прихода внешнего события. К ним относятся:

- режим Sleep (ядро остановлено, периферия работает);
- режим Deepsleep (ядро остановлено, периферия переведена в режим Powerdown).

Контроль режимов энергопотребления осуществляется с помощью блока управления энергопотреблением PMU.

5.1 Режим Sleep

Вход в режим Sleep производится при выполнении инструкции WFI (Wait For Interrupt) или WFE (Wait For Event) при сброшенном бите SLEEPDEEP регистра SCR блока SCB ядра. В этом режиме прекращается тактирование ядра, но вся периферия продолжает полностью функционировать.

В зависимости от состояния бита SLEEPONEXIT (регистр SCR блока SCB ядра) возможны два режима входа в этот режим:

- SLEEPONEXIT = 0 – вход в режим Sleep производится, как только была выполнена инструкция WFI или WFE, при выходе продолжает исполняться программа, прерванная одной из инструкций выше;
- SLEEPONEXIT = 1 – вход в режим Sleep осуществляется не только при выполнении соответствующих инструкций, но и каждый раз при выходе из последнего активного обработчика прерываний (ядро «просыпается» только для обработки прерываний).

Выход из режима Sleep производится различными путями, в зависимости от того, какая инструкция была использована для входа.

Если для входа в режим Sleep была использована инструкция WFI, то приход любого разрешенного прерывания в NVIC вызовет выход из режима, одновременно с переходом в соответствующий обработчик.

Если была использована инструкция WFE, то выход произойдет либо по разрешенному прерыванию NVIC (аналогично WFI), либо по событию, которое может быть сгенерировано несколькими путями.

Если установлен бит SEVONPEND регистра SCR блока SCB ядра, то событием будет являться приход любого запрещенного прерывания в NVIC (разрешенного лишь в периферийном модуле, а не в NVIC). При этом по выходе необходимо будет сбрасывать соответствующие флаги прерывания, как в периферийном модуле, так и в регистрах NVIC (регистры ICPR).

Также событием для выхода из режима Sleep является сигнал RXEV, который может быть сгенерирован в GPIOA или GPIOB. Дополнительно сигнал RXEV от необходимого порта должен быть разрешен установкой бита GPIOAEV или GPIOBEV в регистре RXEVEN блока PMU. В этом случае выход из режима Sleep будет произведен наиболее быстро, так как время на вход/выход в прерывание тратиться не будет.

5.2 Режим Deepsleep

В режиме Deepsleep тактирование ядра останавливается, а также происходит переход в режим Powerdown следующих периферийных блоков (по умолчанию): MFLASH, PLL, EXTOSC.

Вход в режим Deepsleep производится при выполнении инструкции WFI (Wait For Interrupt) или WFE (Wait For Event) при установленном бите SLEEPDEEP регистра SCR блока SCB ядра.

Если для входа в режим Deepsleep была использована инструкция WFI, то приход любого разрешенного прерывания в NVIC вызовет выход из режима, одновременно с переходом в соответствующий обработчик.

Если для входа в режим Deepsleep была использована инструкция WFE, то приход любого разрешенного события RXEV от портов ввода-вывода вызовет выход из режима.

Перед тем как осуществить вход в этот режим необходимо провести некоторые подготовительные операции.

Необходимо включить PMU, установив бит EN в регистре CFG блока PMU. В противном случае переход в режим Deepsleep будет невозможен.

С помощью сброса битов PLLPD, MFLASHPD, EXTOSCPD в регистре PDEN блока PMU можно отключить перевод соответствующих блоков в режим Powerdown при входе в режим Deepsleep – они продолжат свое функционирование. Например, если оставить активным внешний осциллятор, периферия, использующая его в качестве тактового сигнала, продолжит функционировать.

Необходимо настроить значение длительности выхода из режима Deepsleep – выход периферийных блоков из режима Powerdown требует некоторого времени (для каждого блока разное), поэтому в регистр PUDEL блока PMU должно быть внесено значение большее либо равное самой большой величине задержки среди блоков, для которых активен режим Powerdown. Значение вносится в тактах тактового сигнала OSICLK. В таблице 5.1 представлены типовые величины задержек. По умолчанию регистр PUDEL содержит значение 1680, т. к. возможность активации режима Powerdown включена для блока PLL.

Таблица 5.1 – Настройка времени выхода из режима Powerdown для указанных блоков

Периферийный блок	Время выхода из режима Powerdown, мкс	Значение регистра PUDEL в тактах сигнала OSICLK (f_{OSICLK} = 8 МГц)
PLL	210	1680
MFLASH	15	120
EXTOSC	0	0

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 четыре основные области, см. таблицу 6.1.

Таблица 6.1 – Общая организация памяти микроконтроллера

Адресное пространство	Описание
E000_0000h – FFFF_FFFFh	Системная область
4000_0000h – 400F_FFFFh	Регистры управления периферийными блоками
2000_0000h – 2000_3FFFh	Внутреннее ОЗУ 16 Кбайт
0000_0000h – 0000_FFFFh	Flash-память (шина выборки инструкций I-Code + шина данных D-Code)

Системная область делится на две части. Первая часть, объемом 1 Мбайт, занимает диапазон адресов E000_0000h – E007_FFFFh и зарезервирована для «личной» периферийной шины PPB (Private Peripheral Bus). Шина PPB используется для выборки/записи данных и отладочного доступа – для периферии. Эта область рассматривается как строго упорядоченная память. Часть ее адресов (E000_E000h – E000_EFFFh) занимает пространство управления системы SCS, в котором находятся регистры блока управления системой SCB, контроллера прерываний NVIC, системного таймера SysTick, средств отладки и другие. Байты регистров SCS всегда располагаются в порядке «младший – старший» независимо от того, какой формат представления данных принят для остальной памяти. Доступ к большинству регистров возможен только полными двойными словами (32 бита), все исключения из этого правила оговариваются специально. Попытка кода обратиться к области PPB обычно вызывает прерывание по ошибке шины; исключениями являются возможность доступа к регистру STIR, если это разрешено соответствующим битом регистра CCR, и доступность отладочных регистров. Вторая часть является памятью устройств, занимает область E010_0000h – FFFF_FFFFh и является системным регионом ядра.

Регистры периферийных блоков микроконтроллера доступны в адресном пространстве 4000_0000h – 4000_FFFFh. Карта размещения блоков микроконтроллера в памяти представлена в таблице 6.2.

Таблица 6.2 – Регистры периферийных блоков

Адресное пространство	Название блока	Описание
4005_3000h – 4005_3FFFh	ECAP2	Блок захвата 2
4005_2000h – 4005_2FFFh	ECAP1	Блок захвата 1
4005_1000h – 4005_1FFFh	ECAP0	Блок захвата 0
4005_0000h – 4005_0FFFh	I2C	Последовательный интерфейс I2C
4004_F000h – 4004_FFFFh	QEP	Квадратурный декодер QEP
4004_E000h – 4004_EFFFh	PWM2	Блок ШИМ2
4004_D000h – 4004_DFFFh	PWM1	Блок ШИМ1
4004_C000h – 4004_CFFFh	PWM0	Блок ШИМ0
4004_B000h – 4004_BFFFh	TMR3	Блок таймера 3
4004_A000h – 4004_AFFFh	TMR2	Блок таймера 2

Окончание таблицы 6.2

Адресное пространство	Название блока	Описание
4004_9000h – 4004_9FFFh	TMR1	Блок таймера 1
4004_8000h – 4004_8FFFh	TMR0	Блок таймера 0
4004_7000h – 4004_7FFFh	SPI	Последовательный интерфейс SPI
4004_6000h – 4004_6FFFh	UART1	Последовательный приемопередатчик UART1
4004_5000h – 4004_5FFFh	UART0	Последовательный приемопередатчик UART0
4004_4000h – 4004_4FFFh	DMA	Контроллер прямого доступа к памяти DMA
4004_3000h – 4004_3FFFh	WDT	Сторожевой таймер WDT
4004_2000h – 4004_2FFFh	PMU	Блок управления питанием PMU
4004_1000h – 4004_1FFFh	RCU	Блок управления тактированием и сбросом RCU
4004_0000h – 4004_0FFFh	SIU	Блок системных функций
4003_0000h – 4003_FFFFh	MFLASH	Flash-память
4002_0000h – 4002_FFFFh	CAN	Контроллер CAN
4001_1000h – 4001_1FFFh	GPIOB	Порт B
4001_0000h – 4001_0FFFh	GPIOA	Порт A
4000_0000h – 4000_FFFFh	ADC	Блок АЦП

7 Контроллер Flash-памяти

7.1 Основная Flash-память

Основная Flash-память может использоваться для хранения программ и данных пользователя. Размер основной Flash-памяти составляет 64 Кбайт (64 страницы по 1 Кбайт), и в адрессном пространстве она занимает диапазон с 0000h по 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-память занята (стирание, запись), транзакция проходит успешно с неопределенными данными на выходе.

Память доступна для чтения, записи, полного и постраничного стирания через регистры данных DATA1 и DATA0, адреса ADDR, команд CMD, статуса STAT блока MFLASH. Запись необходимо производить в предварительно очищенную ячейку памяти. Стирание памяти осуществляется полностью или постранично.

Минимальное время чтения данных из Flash-памяти составляет около 30 нс. Поэтому, исходя из выбраной рабочей частоты, следует задать определенное количество дополнительных тактов ожидания, необходимое для стабильного чтения из Flash-памяти. Данный параметр заносится в поле LAT регистра CTRL, см. таблицу 7.1.

Таблица 7.1 – Значения параметра (количество дополнительных тактов ожидания) в поле LAT регистра CTRL в зависимости от частоты сигнала SYSCLK

f_{SYSCLK}, МГц, не более	Количество дополнительных тактов ожидания в поле LAT регистра CTRL при чтении Flash-памяти
100	3
90	2
60	1
30	0
Примечание – Значение параметра после сброса равно 1.	

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 два двойных слова (64 бит) данных из Flash-памяти. Далее эти данные записываются во внутренний первый буфер.

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

4 Сразу после установки сигнала готовности, из Flash-памяти считываются два двойных слова 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, 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.

Операция кэширования

Основная Flash-память дополнена блоками кэш-памяти по 1 Кбайт на каждую из шин I-code и D-code. Доступ к этим блокам осуществляется за 1 такт системной частоты. При первом обращении к адресу, его данные параллельно заносятся и в кэш-память, и сохраняются там. Если в дальнейшем произойдет повторное обращение к тому же адресу, данные будут прочитаны уже из кэш, обращения к медленной Flash-памяти не будет. Но данные в кэш, которые были запрошены давно, могут быть вытесненны более свежими данными, обращение к которым было позже. Таким образом, кэшированная информация постоянно обновляется, но участки программного кода, вызываемые чаще всего, с большей вероятностью окажутся в кэш. За счет этого достигается повышенная скорость выполнения такого кода.

Предвыборка и кэш на каждую из шин I-code и D-code включаются отдельно в регистре CTRL. Перед включением кэша необходимо его сбросить, установив соответствующий бит FLUSH регистра CTRL. Во время процедуры сброса бит BUSY соответствующего регистра статуса кэш будет держаться в 1. После успешного завершения сброса будет сброшен и данный бит, после чего можно продолжать выполнение основной программы.

7.2 Загрузочная Flash-память

Загрузочная Flash-память (NVR область) может использоваться для хранения пользовательской программы-загрузчика, которая посредством одного из имеющихся интерфейсов получает данные для прошивки основной памяти. Размер загрузочной Flash-памяти составляет 4 Кбайт (4 страницы по 1 Кбайт каждая). Но для программы-загрузчика доступны только первые 3 Кбайт. Верхний килобайт отведен под хранение пользовательских данных, а в его первом байте (абсолютный адрес C00h) располагается конфигурационное слово CFGWORD, содержащее параметры загрузки и защиты контроллера. Описание битов CFGWORD представлено в таблице 7.2.

Примечание – параллельный доступ к NVR и основной области Flash-памяти невозможен. Так например, ядро, исполяющее код из основной области Flash будет вынужденно остановиться на время осуществления модификации NVR области, т.к. не может в это время считывать инструкции. Подобное справедливо и для обратного случая.

Таблица 7.2 – Конфигурационное слово CFGWORD

CFGWORD	+ C00h (смещение относительно начального адреса NVR области)	
Поле	Биты	Описание
BMODEDIS	4	Бит источника загрузки
	0	Загрузка из NVR области
	1	Загрузка из основной Flash-памяти (по умолчанию)
FLASHWE	3	Бит включения защиты основной Flash-памяти
	0	Запрет записи и стирания основного блока Flash-памяти
	1	Защита выключена (по умолчанию)
NVRWE	2	Бит включения защиты NVR области Flash-памяти
	0	Запрет записи и стирания NVR блока Flash-памяти
	1	Защита выключена (по умолчанию)
DEBUGEN	1	Бит разрешения работы системы отладки ядра
	0	Откладка отключена
	1	Откладка включена (по умолчанию)
JTAGEN	0	Бит разрешения работы пинов JTAG/SWD
	0	Работа запрещена
	1	Работа разрешена (по умолчанию)
–	7-5	Зарезервировано

При установленных битах NVRWE или FLASHWE запрещается любая модификация соответсвующей области памяти через регистровый интерфейс. Операции записи и стирания интерпретируются как операция чтения, при этом стандартным образом продолжают работать механизмы флагов из регистра STAT.

По умолчанию после стирания CFGWORD включена загрузка из основной памяти. Чтобы использовать загрузчик, необходимо сбросить в нем бит BMODEDIS. Тогда при загрузке произойдет подмена основной Flash-памяти на область загрузочной Flash. Т. е. при обращении по адресам 0000_0000h – 0000_0FFFh будут прочитаны данные из загрузочной Flash-памяти, см. рисунок 7.1.

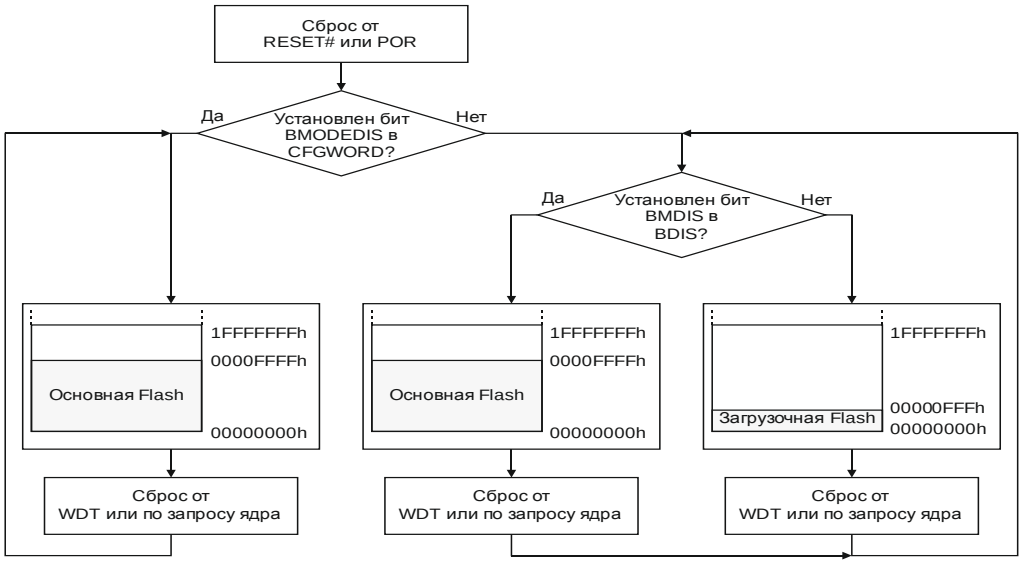


Рисунок 7.1 – Блок-схема алгоритма загрузки микроконтроллера

При этом обе области полностью доступны для чтения, записи, стирания через регистровый интерфейс MFLASH.

Есть возможность после отработки программы-загрузчика сразу перезапустить контроллер с области основной Flash-памяти. Для этого необходимо установить бит BMDIS регистра BDIS блока MFLASH и подать команду программной перезагрузки контроллера (установить бит SYSRESETREQ регистра AIRCR ядра). Устройство перезагрузится уже с основной Flash-памятью. Данная комбинация действий будет работать только до внешнего сброса или сброса по питанию контроллера. После внешней перезагрузки при сброшеном бите BMODEDIS загрузка опять начнется с загрузочной Flash-памяти. Чтобы отключить полностью старт с загрузочной Flash-памяти, необходимо вернуть бит BMODEDIS в 1. Это возможно осуществить, выполнив стирание соответствующей страницы загрузочной Flash-памяти или сервисное стирание всего контроллера. При этом обновлённое значение вступит в силу только после внешнего сброса или сброса по питанию. Подобное справедливо и для других битов CFGWORD.

7.3 Сервисный сброс всей Flash-памяти

1 Во время сброса микроконтроллера анализируется состояние вывода SERVEN. Если вывод находится в состоянии логической единицы (подтянут к 3,3 В), то загрузочная и основная Flash-памяти переводятся в режим, в котором чтение запрещено (при чтении возвращаются нули). При этом игнорируется состояние битов DEBUGEN и JTAGEN.

2 Далее по отладочному интерфейсу (SWD или JTAG) должна быть подана команда записи значения 0000_0001h в регистр SERVCTL блока SIU, после чего будет активировано полное стирание загрузочной и основной Flash-памяти. По завершении процесса стирания в этом же регистре выставится флаг DONE

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

8 Контроллер прямого доступа к памяти DMA

Основные свойства и отличительные особенности контроллера прямого доступа к памяти DMA:

- 16 каналов;
- каждый канал DMAi (где i от 0 до 15) 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;
- поддержка передачи данных различной разрядности;
- каждому каналу DMAi доступна первичная и альтернативная структура управляющих данных канала;
- все данные канала хранятся во внутреннем ОЗУ в структуре управляющих данных канала;
- разрядность данных приемника равна разрядности данных передатчика;
- количество передач в одном цикле контроллера DMA может программироваться от 1 до 1024;
- инкремент адреса передачи может быть больше, чем разрядность данных;
- возможность начать передачи по сигналам от блоков UART, SPI, ADC, TMR, PWM, QEP.

Аппаратные источники запросов каналов контроллера DMA указаны в таблице 8.1.

Таблица 8.1 – Аппаратные источники запросов каналов контроллера DMA

Номер канала	Аппаратный источник запросов	Описание
0	UART0_TX	Канал DMA от UART0 по передаче
1	UART1_TX	Канал DMA от UART1 по передаче
2	UART0_RX	Канал DMA от UART0 по приему
3	UART1_RX	Канал DMA от UART1 по приему
4	ADC_SEQ0	Канал DMA от секвенсора 0 блока АЦП
5	ADC_SEQ1	Канал DMA от секвенсора 1 блока АЦП
6	SPI_TX	Канал DMA от SPI по передаче
7	SPI_RX	Канал DMA от SPI по приему
8 – 15	В соответсвии с регистром DMAMUX блока SIU	Каналы DMA с конфигурируемыми источниками запросов

8.1 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 8.1 показана область памяти, необходимая контроллеру для структур управляющих данных каналов, при использовании всех 16 каналов и опциональной альтернативной структуры данных.

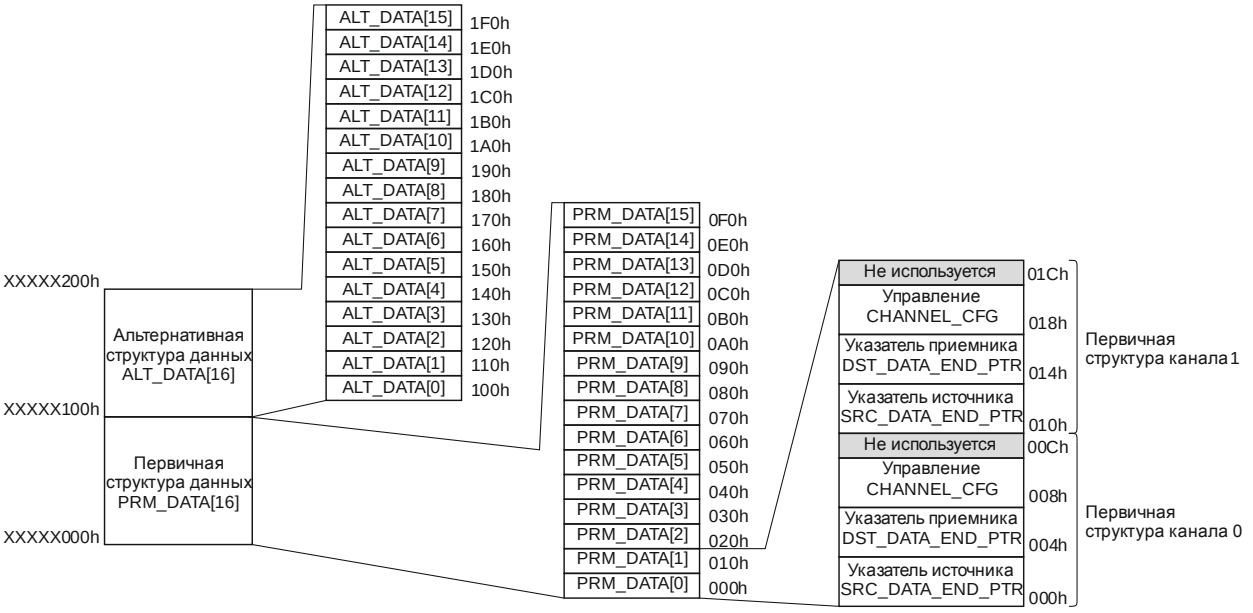


Рисунок 8.1 – Карта памяти для 16 каналов контроллера DMA, включая альтернативную структуру

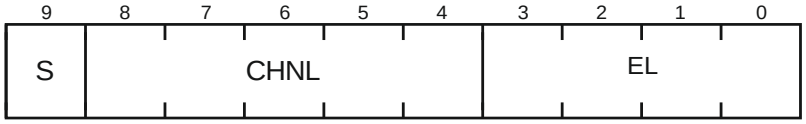
Объем структуры, показанной на рисунке 8.1, составляет 512 байт. Контроллер использует младшие разряды адреса для доступа ко всем элементам структуры управляющих данных, и поэтому разрешенные значения базового адреса для первичной структуры управляющих данных – XXXX_X000h, XXXX_X200h, XXXX_X400h и т. д.

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

В таблице 8.2 перечислены разряды адреса, обеспечивающие контроллеру DMA доступ к различным элементам структуры управляющих данных.

Таблица 8.2 – Разряды адреса, используемые для доступа к управляющим данным 16 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-18h	
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.

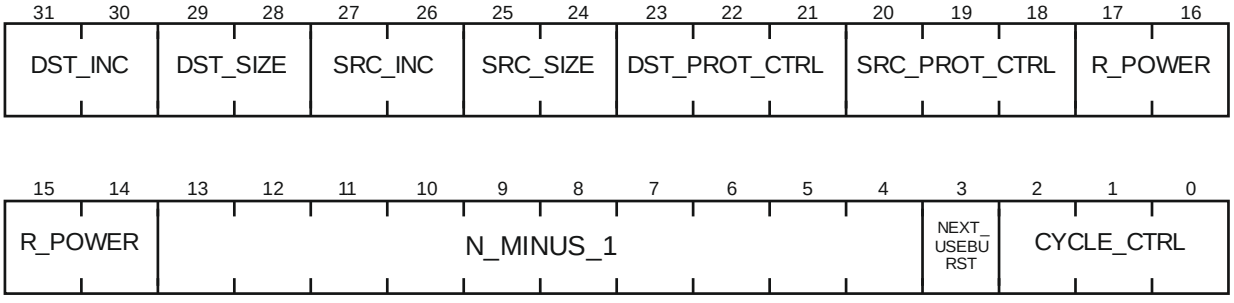
Любая из структур управляющих данных каждого канала состоит из двух указателей адреса (приемника и источника данных) и ячейки управления канала.

Управляющие данные канала CHANNEL_CFG

32-разрядная ячейка памяти, содержащая конфигурационную информацию для осуществления передач DMA (на рисунке 8.1 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 CHANNEL_CFG приведена в таблице 8.3.

Таблица 8.3 – Структура управляющих данных канала CHANNEL_CFG

CHANNEL_CFG



Продолжение таблицы 8.3

<table>
<tr><th>Поле</th><th>Биты</th><th colspan="4">Описание</th></tr>
<tr><td rowspan="8">DST_INC</td><td rowspan="8">31-30</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 rowspan="2">Код</td><td colspan="3">Разрядность данных источника</td></tr>
<tr><td>Байт</td><td>Слово (16 бит)</td><td>Двойное слово (32 бита)</td></tr>
<tr><td>00</td><td>Шаг – байт</td><td>Зарезервировано</td><td>Зарезервировано</td></tr>
<tr><td>01</td><td colspan="2">Шаг – слово (16 бит)</td><td>Зарезервировано</td></tr>
<tr><td>10</td><td colspan="3">Шаг – двойное слово (32 бита)</td></tr>
<tr><td>11</td><td colspan="3">Нет инкремента. Адрес остается равным значению ячейки DST_DATA_END_PTR</td></tr>
<tr><td></td><td></td><td colspan="4"></td></tr>
<tr><td>DST_SIZE</td><td>29-28</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–1 устанавливает значение поля DST_SIZE, равное SRC_SIZE.</td></tr>
<tr><td rowspan="8">SRC_INC</td><td rowspan="8">27-26</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 rowspan="2">Код</td><td colspan="3">Разрядность данных источника</td></tr>
<tr><td>Байт</td><td>Слово (16 бит)</td><td>Двойное слово (32 бита)</td></tr>
<tr><td>00</td><td>Шаг – байт</td><td>Зарезервировано</td><td>Зарезервировано</td></tr>
<tr><td>01</td><td colspan="2">Шаг – слово (16 бит)</td><td>Зарезервировано</td></tr>
<tr><td>10</td><td colspan="3">Шаг – двойное слово (32 бита)</td></tr>
<tr><td>11</td><td colspan="3">Нет инкремента. Адрес остается равным значению ячейки SRC_DATA_END_PTR</td></tr>
<tr><td></td><td></td><td colspan="4"></td></tr>
<tr><td rowspan="5">SRC_SIZE</td><td rowspan="5">25-24</td><td colspan="4">Разрядность данных источника</td></tr>
<tr><td>00</td><td colspan="3">Байт</td></tr>
<tr><td>01</td><td colspan="3">Слово (16 бит)</td></tr>
<tr><td>10</td><td colspan="3">Двойное слово (32 бита)</td></tr>
<tr><td>11</td><td colspan="3">Зарезервировано. Не использовать!</td></tr>
<tr><td rowspan="4">DST_PROT_CTRL</td><td rowspan="4">23-21</td><td colspan="4">Задает параметры защиты шины AHB при записи данных в приемник</td></tr>
<tr><td rowspan="2">Код</td><td colspan="3">Биты поля DST_PROT_CTRL</td></tr>
<tr><td>23</td><td>22</td><td>21</td></tr>
<tr><td>0</td><td>Доступ не кэшируется</td><td>Доступ не буферизуется</td><td>Доступ непривилегированный</td></tr>
<tr><td>1</td><td>Доступ кэшируется</td><td>Доступ буферизуется</td><td>Доступ привилегированный</td></tr>
<tr><td rowspan="4">SRC_PROT_CTRL</td><td rowspan="4">20-18</td><td colspan="4">Задает параметры защиты шины AHB при чтении данных из источника</td></tr>
<tr><td rowspan="2">Код</td><td colspan="3">Биты поля SRC_PROT_CTRL</td></tr>
<tr><td>20</td><td>19</td><td>18</td></tr>
<tr><td>0</td><td>Доступ не кэшируется</td><td>Доступ не буферизуется</td><td>Доступ непривилегированный</td></tr>
<tr><td>1</td><td>Доступ кэшируется</td><td>Доступ буферизуется</td><td>Доступ привилегированный</td></tr>
</table>

Продолжение таблицы 8.3

<table>
<tr><th>Поле</th><th>Биты</th><th>Описание</th></tr>
<tr><td>R_POWER</td><td>17-14</td><td>Параметр R.
Задает количество передач канала DMA до выполнения контроллером DMA процедуры арбитража (переарбитрации).
Количество передач равно 2^R
<table>
<tr><th>Код</th><th>Количество передач</th></tr>
<tr><td>0h</td><td>1 (арбитраж производится после каждой передачи DMA)</td></tr>
<tr><td>1h</td><td>2</td></tr>
<tr><td>2h</td><td>4</td></tr>
<tr><td>3h</td><td>8</td></tr>
<tr><td>4h</td><td>16</td></tr>
<tr><td>5h</td><td>32</td></tr>
<tr><td>6h</td><td>64</td></tr>
<tr><td>7h</td><td>128</td></tr>
<tr><td>8h</td><td>256</td></tr>
<tr><td>9h</td><td>512</td></tr>
<tr><td>Ah – Fh</td><td>1024 (арбитраж не производится, так как максимальное количество передач DMA равно 1024)</td></tr>
</table>
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>
<tr><td>N_MINUS_1</td><td>13-4</td><td>Перед выполнением цикла DMA эти разряды указывают общее количество передач DMA, из которых состоит цикл
<table>
<tr><th>Код</th><th>Количество передач</th></tr>
<tr><td>000h</td><td>1</td></tr>
<tr><td>001h</td><td>2</td></tr>
<tr><td>…</td><td>…</td></tr>
<tr><td>3FFh</td><td>1024</td></tr>
</table>
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>NEXT_
USEBURST</td><td>3</td><td>Контролирует установку соответствующий 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</td></tr>
</table>

Окончание таблицы 8.3

Поле	Биты	Описание	
NEXT_ USEBURST	3	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 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
CYCLE_ CTRL	2-0	Поле задания типа цикла DMA	
		000b	Недействительный. Структура управляющих данных канала в запрещенном состоянии
		001b	Основной
		010b	Авто-запрос
		011b	«Пинг-понг»
		100b	Работа с памятью в режиме «разборка-сборка» с использованием первичных управляющих данных канала
		101b	Работа с памятью в режиме «разборка-сборка» с использованием альтернативных управляющих данных канала
		110b	Работа с периферией в режиме «разборка-сборка» с использованием первичных управляющих данных канала
		111b	Работа с периферией в режиме «разборка-сборка» с использованием альтернативных управляющих данных канала
		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.	

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–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/передачи, и контроллер DMA выполняет сдвиг влево значения N–1 на количество разрядов, соответствующее полю DST_INC и затем вычитает получившееся значение от значения DST_DATA_END_PTR.

8.2 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 8.4, 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 8.4 – 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 активным 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–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

Окончание таблицы 8.4

Номер правила	Описание
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
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–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–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 8.5.

Таблица 8.5 – 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

8.3 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 8.6.

После окончания цикла DMA контроллер выбирает следующий для обслуживания канал из всех включенных каналов DMAi. Рисунок 8.2 иллюстрирует процесс выбора следующего канала для обслуживания.

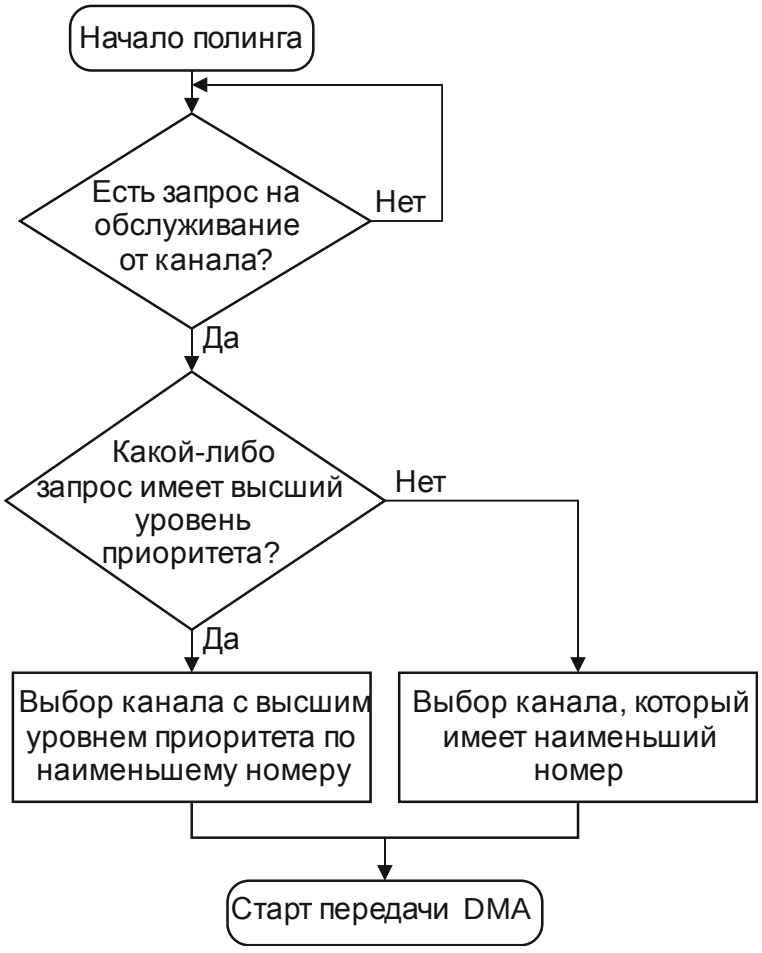


Рисунок 8.2 – Алгоритм выбора (полинга) следующего канала для обслуживания

Таблица 8.6 – 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	
…	…	…	
14	1	Высокий	
15	1	Высокий	
0	0	По умолчанию	
1	0	По умолчанию	
2	0	По умолчанию	
3	0	По умолчанию	
…	…	…	
14	0	По умолчанию	
15	0	По умолчанию	Низший

8.4 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 8.3 показан пример функционирования контроллера DMA в режиме «пинг-понг». Пояснения к рисунку 8.3 представлены в виде таблицы 8.7.

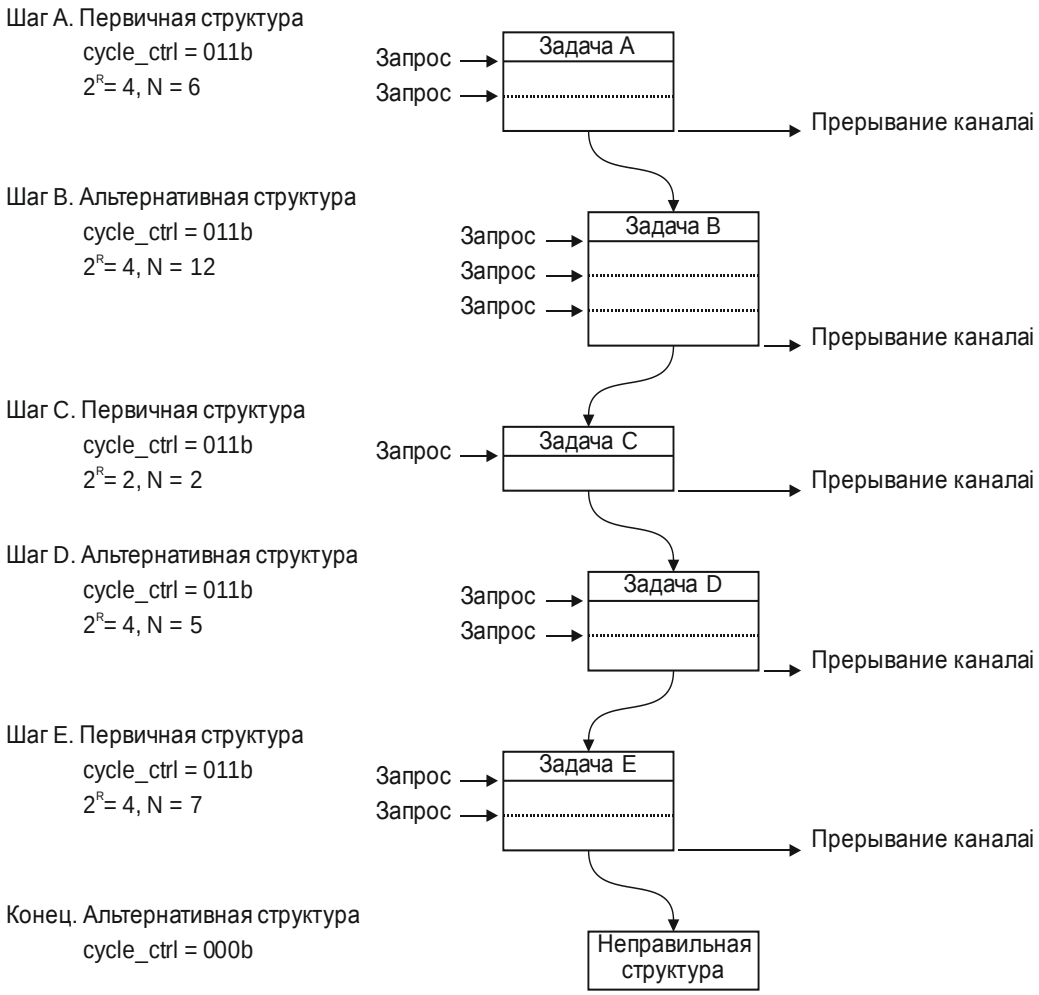


Рисунок 8.3 – Пример схемы функционирования контроллера DMA в режиме «пинг-понг»

Таблица 8.7 – Пояснения к схеме на рисунке 8.3

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 2 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.

Окончание таблицы 8.7

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:

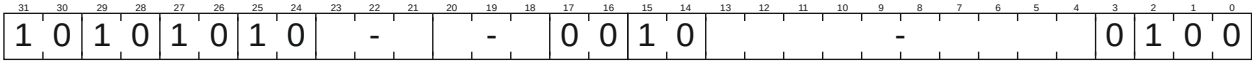
- передача с использованием альтернативной управляющей структуры будет выполнена в режиме цикла «основной»;
- контроллер считает «неправильную» структуру управляющих данных. После исполнения контроллером N передач с использованием первичных управляющих данных, он делает эти управляющие данные «неправильными» 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 8.8 указаны константы, которые должны быть записаны пользователем в регистр CHANNEL_CFG первичной структуры управляющих данных канала для работы с памятью в режиме «разборка-сборка».

Таблица 8.8 – Конфигурация первичной структуры управляющих данных канала для работы с памятью в режиме «разборка-сборка»

CHANNEL_CFG



Поле	Биты	Кон-станта	Пояснение
DST_INC	31-30	10b	Контроллер DMA производит инкремент адреса с шагом в двойное слово
DST_SIZE	29-28	10b	Контроллер DMA осуществляет передачу двойным словом
SRC_INC	27-26	10b	Контроллер DMA производит инкремент адреса с шагом в двойное слово
SRC_SIZE	25-24	10b	Контроллер DMA осуществляет передачу двойными словами
DST_PROT_CTRL	23-21	–	Управление защитой шины при записи данных в приемник. Задается пользователем
SRC_PROT_CTRL	20-18	–	Управление защитой шины при чтении данных из источника. Задается пользователем
R_POWER	17-14	0010b	Контроллер DMA выполняет четыре передачи ($2^R = 2^2 = 4$)
N_MINUS_1	13-4	–	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
NEXT_USEBURST	3	0	Для данного режима бит должен быть сброшен
CYCLE_CTRL	2-0	100b	Контроллер DMA работает с памятью в режиме «разборка-сборка» с использованием первичных управляющих данных канал.

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 8.4 показан пример схемы функционирования контроллера в режиме «разборка-сборка». Пояснения к рисунку 8.4 приведены ниже («Инициализация» и «Функционирование»).

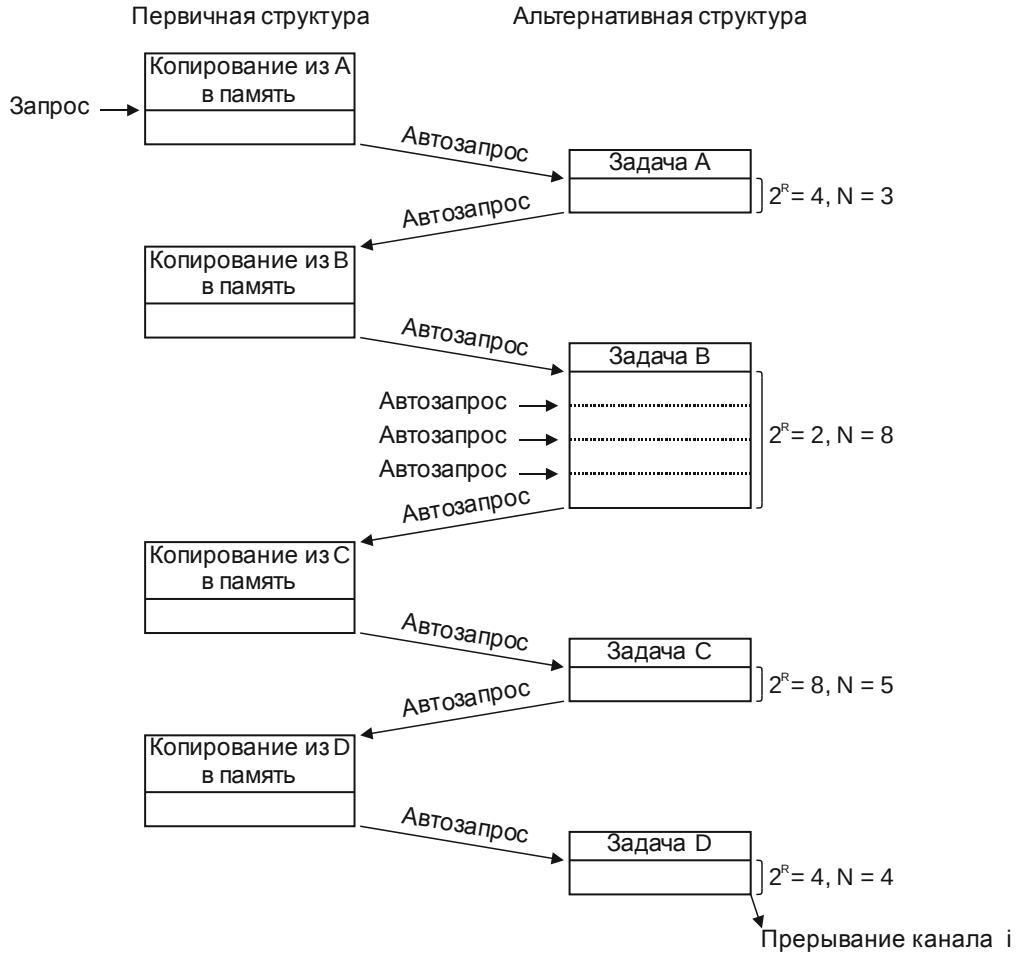


Рисунок 8.4 – Пример схемы функционирования контроллера DMA в режиме «разборка-сборка»

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–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 8.9. Исходя из примера, в регистр SRC_DATA_END_PTR первичной управляющей структуры необходимо занести значение 2000_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 8.9 – Пример размещения управляющих данных для альтернативной структуры

Тип данных	Адрес ОЗУ	Регистр	Значение
Управляющие данные для задачи D	2000_015Ch	Не используется	XXXX_XXXXh
	2000_0158h	CHANNEL_CFG	CYCLE_CTRL = 001b, 2^R = 4, N = 4
	2000_0154h	DST_DATA_END_PTR	2000_DE00h
	2000_0150h	SRC_DATA_END_PTR	2000_D000h
Управляющие данные для задачи C	2000_016Ch	Не используется	XXXX_XXXXh
	2000_0168h	CHANNEL_CFG	CYCLE_CTRL = 101b, 2^R = 8, N = 5
	2000_0164h	DST_DATA_END_PTR	2000_CE00h
	2000_0160h	SRC_DATA_END_PTR	2000_C000h
Управляющие данные для задачи B	2000_017Ch	Не используется	XXXX_XXXXh
	2000_0178h	CHANNEL_CFG	CYCLE_CTRL = 101b, 2^R = 2, N = 8
	2000_0174h	DST_DATA_END_PTR	2000_BE00h
	20000170h	SRC_DATA_END_PTR	2000_B000h
Управляющие данные для задачи A	2000_018Ch	Не используется	XXXX_XXXXh
	2000_0188h	CHANNEL_CFG	CYCLE_CTRL = 101b, 2^R = 4, N = 3
	2000_0184h	DST_DATA_END_PTR	2000_AE00h
	2000_0180h	SRC_DATA_END_PTR	2000_A000h

Функционирование

1 Первичная структура, копирование данных задачи A. По получении первого запроса на обслуживание контроллер DMA выполняет четыре передачи DMA. Эти передачи записывают альтернативные структуры 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:

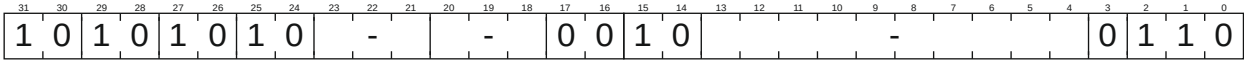
- передача с использованием альтернативной управляющей структуры будет выполнена в режиме цикла «основной»;
- контроллер DMA считает «неправильную» структуру управляющих данных. После исполнения DMA контроллером N передач с использованием первичных управляющих данных, он делает эти управляющие данные «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 8.10 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 8.10 – Конфигурация первичной структуры управляющих данных канала DMAi для работы с периферией в режиме «разборка-сборка»

CHANNEL_CFG



Поле	Биты	Конс-танта	Пояснение
DST_INC	31-30	10b	Контроллер DMA производит инкремент адреса с шагом в двойное слово
DST_SIZE	29-28	10b	Контроллер DMA осуществляет передачу двойным словом
SRC_INC	27-26	10b	Контроллер DMA производит инкремент адреса с шагом в двойное слово
SRC_SIZE	25-24	10b	Контроллер DMA осуществляет передачу двойными словами
DST_PROT_CTRL	23-21	–	Управление защитой шины при записи данных в приемник. Задается пользователем
SRC_PROT_CTRL	20-18	–	Управление защитой шины при чтении данных из источника. Задается пользователем
R_POWER	17-14	0010b	Контроллер DMA выполняет четыре передачи DMA ($2^R = 2^2 = 4$)
N_MINUS_1	13-4	–	Настраивает контроллер DMA на выполнение N передач DMA. Так как поле R_POWER задает значение, равное двум, то необходимо задавать значение N, кратное четырем. Число, равное N/4, это количество раз, которое нужно настраивать альтернативные управляющие данные. Задается пользователем
NEXT_USEBURST	3	0	Для данного режима бит должен быть сброшен
CYCLE_CTRL	2-0	110b	Контроллер работает с периферией в режиме «разборка-сборка» с использованием первичных управляющих данных канала DMAi

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 8.5 показан пример схемы функционирования контроллера DMA в режиме «разборка-сборка». Пояснения к рисунку 8.5 приведены ниже («Инициализация» и «Функционирование»).

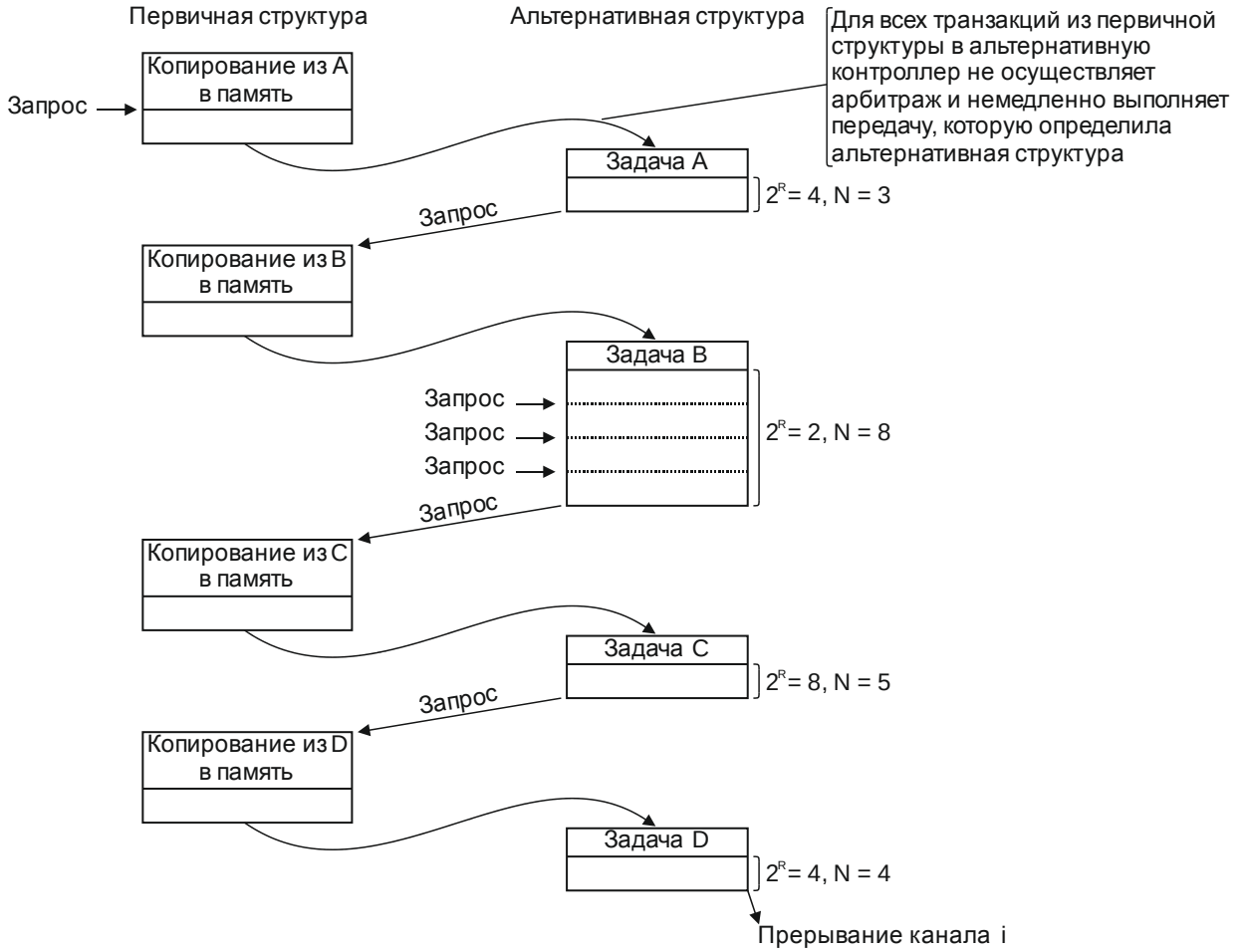


Рисунок 8.5 – Пример схемы функционирования контроллера DMA в режиме работы с периферией «разборка-сборка»

Инициализация

1 Первичная структура управляющих данных настраивается для работы с периферией в режиме «разборка-сборка» путем записи в CYCLE_CTRL значения 110b. 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–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 8.11. Исходя из примера, в регистр SRC_DATA_END_PTR необходимо занести значение 2000_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 8.11 – Пример размещения управляющих данных для альтернативной структуры

Тип данных	Адрес ОЗУ	Регистр	Значение
Управляющие данные для задачи D	2000_015Ch	Не используется	XXXX_XXXXh
	2000_0158h	CHANNEL_CFG	CYCLE_CTRL = 001b, 2^R = 4, N = 4
	2000_0154h	DST_DATA_END_PTR	2000_DE00h
	2000_0150h	SRC_DATA_END_PTR	2000_D000h
Управляющие данные для задачи C	2000_016Ch	Не используется	XXXX_XXXXh
	2000_0168h	CHANNEL_CFG	CYCLE_CTRL = 111b, 2^R = 8, N = 5
	2000_0164h	DST_DATA_END_PTR	2000_CE00h
	2000_0160h	SRC_DATA_END_PTR	2000_C000h
Управляющие данные для задачи B	2000_017Ch	Не используется	XXXXXXXXh
	2000_0178h	CHANNEL_CFG	CYCLE_CTRL = 111b, 2^R = 2, N = 8
	2000_0174h	DST_DATA_END_PTR	2000_BE00h
	2000_0170h	SRC_DATA_END_PTR	2000_B000h
Управляющие данные для задачи A	2000_018Ch	Не используется	XXXX_XXXXh
	2000_0188h	CHANNEL_CFG	CYCLE_CTRL = 111b, 2^R = 4, N = 3
	2000_0184h	DST_DATA_END_PTR	2000_AE00h
	2000_0180h	SRC_DATA_END_PTR	2000_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.

8.5 Индикация ошибок

Контроллер DMA может отключить i-й 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 нему 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.

9 Система прерываний

Таблица прерываний 9.1 представляет собой перечень адресов, соответствующих определенным обработчикам прерываний.

Таблица 9.1 – Таблица прерываний

Номер вектора	Смещение	Обозначение	Описание
-	0000h	SP	Вершина стека
-	0004h	Reset	Сброс
-	0008h	NMI#	Немаскируемое прерывание NMI#
-	000Ch	HardFault	Любой отказ, если соответствующий обработчик не может быть запущен
-	0010h	MemManage	Прерывание по отказу системы управления памятью
-	0014h	BusFault	Прерывание по отказу шины AHB
-	0018h	UsageFault	Прерывание по ошибке программы
-	001Ch-0028h	-	Зарезервировано
-	002Ch	SVCall	Обработка прерываний, вызванных инструкцией SVC
-	0030h	DebugMonitor	Прерывание монитора отладки
-	0034h	-	Зарезервировано
-	0038h	PendSV	Прерывание системного уровня. В приложении используется вызов «Супервизор», если этот запрос обслуживается базовой операционной системой
-	003Ch	SysTick	Прерывание системного уровня. Прерывание вызывается таймером SysTick
0	0040h	WDT	Прерывание блока сторожевого таймера
1	0044h	RCU	Прерывание блока RCU
2	0048h	MFLASH	Прерывание контроллера Flash-памяти
3	004Ch	GPIOA	Прерывание порта A
4	0050h	GPIOB	Прерывание порта B
5	0054h	DMA_CH0	Прерывания контроллера DMA по каналам
6	0058h	DMA_CH1	
7	005Ch	DMA_CH2	
8	0060h	DMA_CH3	
9	0064h	DMA_CH4	
10	0068h	DMA_CH5	
11	006Ch	DMA_CH6	
12	0070h	DMA_CH7	
13	0074h	DMA_CH8	
14	0078h	DMA_CH9	
15	007Ch	DMA_CH10	
16	0080h	DMA_CH11	
17	0084h	DMA_CH12	

Продолжение таблицы 9.1

Номер вектора	Смещение	Обозначение	Описание
18	0088h	DMA_CH13	Прерывания контроллера DMA по каналам
19	008Ch	DMA_CH14	
20	0090h	DMA_CH15	
21	0094h	TMR0	Прерывание таймера 0
22	0098h	TMR1	Прерывание таймера 1
23	009Ch	TMR2	Прерывание таймера 2
24	00A0h	TMR3	Прерывание таймера 3
25	00A4h	UART0_TD	Прерывание UART0 по окончанию передачи
26	00A8h	UART0_RX	Прерывание UART0 по заполнению FIFO
27	00ACh	UART0_TX	Прерывание UART0 по опустошению FIFO
28	00B0h	UART0	Общее прерывание блока UART0
29	00B4h	UART1_TD	Прерывание UART1 по окончанию передачи
30	00B8h	UART1_RX	Прерывание UART1 по заполнению FIFO
31	00BCh	UART1_TX	Прерывание UART1 по опустошению FIFO
32	00C0h	UART1	Общее прерывание блока UART1
33	00C4h	SPI	Общее прерывание контроллера SPI
34	00C8h	SPI_RX	Прерывание SPI по заполнению FIFO
35	00CCh	SPI_TX	Прерывание SPI по опустошению FIFO
36	00D0h	I2C	Прерывание контроллера I2C
37	00D4h	ECAP0	Прерывание блока захвата 0
38	00D8h	ECAP1	Прерывание блока захвата 1
39	00DCh	ECAP2	Прерывание блока захвата 2
40	00E0h	PWM0	Общее прерывание блока ШИМ0
41	00E4h	PWM0_HD	Прерывание схемы удержания блока ШИМ0
42	00E8h	PWM0_TZ	Прерывание детектора аварий блока ШИМ0
43	00ECh	PWM1	Общее прерывание блока ШИМ1
44	00F0h	PWM1_HD	Прерывание схемы удержания блока ШИМ1
45	00F4h	PWM1_TZ	Прерывание детектора аварий блока ШИМ1
46	00F8h	PWM2	Общее прерывание блока ШИМ2
47	00FCh	PWM2_HD	Прерывание схемы удержания блока ШИМ2
48	0100h	PWM2_TZ	Прерывание детектора аварий блока ШИМ2
49	0104h	QEP	Прерывание квадратурного декодера
50	0108h	ADC_SEQ0	Прерывание секвенсора 0 блока АЦП
51	010Ch	ADC_SEQ1	Прерывание секвенсора 1 блока АЦП
52	0110h	ADC_DC	Прерывание компараторов блока АЦП
53	0114h	CAN0	Прерывания контроллера CAN
54	0118h	CAN1	
55	011Ch	CAN2	
56	0120h	CAN3	
57	0124h	CAN4	
58	0128h	CAN5	
59	012Ch	CAN6	
60	0130h	CAN7	
61	0134h	CAN8	
62	0138h	CAN9	
63	013Ch	CAN10	

Окончание таблицы 9.1

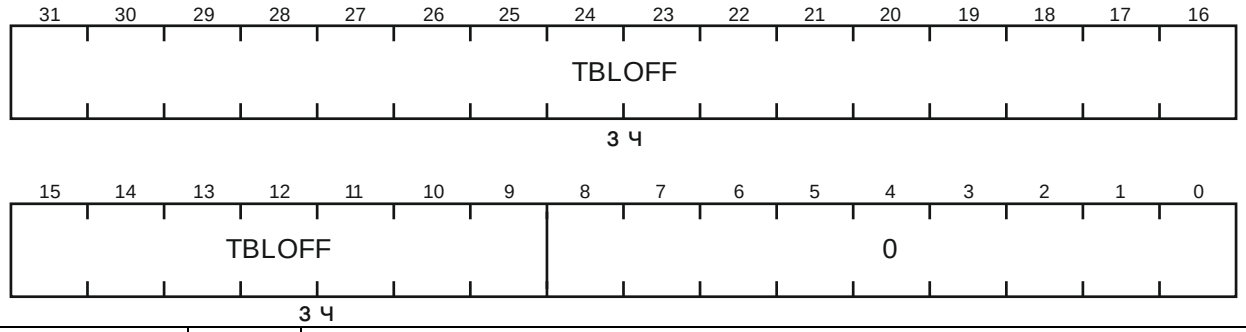
Номер вектора	Смещение	Обозначение	Описание
64	0140h	CAN11	Прерывания контроллера CAN
65	0144h	CAN12	
66	0148h	CAN13	
67	014Ch	CAN14	
68	0150h	CAN15	
69	0154h	FPU	Прерывание исключений блока FPU

Первоначально адрес начала таблицы прерываний 0000_0000h.

Таблица векторов может быть размещена по другому адресу в памяти программ или в ОЗУ. В случае размещения таблицы прерываний в области ОЗУ появляется возможность изменять обработчики прерываний в процессе выполнения программы. Положение таблицы векторов в памяти определяется регистром VTOR, см. таблицу 9.2.

Таблица 9.2 – Регистр смещения таблицы векторов

VTOR E000_ED08h Сброс: 0000_0000h



Поле	Биты	Описание
TBLOFF	31-9	Биты 31-9 адреса таблицы векторов
0	8-0	Данные биты всегда должны быть равны 0, т.к. адрес таблицы векторов должен быть вывровнен по границе 512 байт

Обработчики прерываний можно динамически менять, но при этом обязательно следует располагать следующие элементы:

- начальное значение основного указателя стека;
- вектор сброса Reset;
- вектор NMI;
- вектор исключения HardFault.

Остальные прерывания не могут генерироваться, пока не будут разрешены.

Контроллер прерываний NVIC

Контроллер прерываний NVIC обеспечивает:

- программное задание уровня приоритета независимо для каждого прерывания в диапазоне от 0 до 7 (прерывание с уровнем 0 имеет наивысший приоритет);
- генерирование сигнала прерывания по фронту и по уровню сигнала;
- динамическое изменение приоритета прерываний;
- разделение по группам с одинаковым приоритетом и по подгруппам внутри одной группы;

- передача управления из одного обработчика в другой без восстановления контекста.

Процессор автоматически сохраняет в стеке свое состояние при входе в обработчик прерывания и восстанавливает свое состояние после завершения обработки прерывания, т. е. без необходимости программирования этих операций.

Обработка прерываний по уровню и по фронту

Контроллер прерываний NVIC поддерживает прерывания, как по фронту, так и по уровню. Прерывание по фронту – импульсное прерывание, которое может иметь длительность большую или равную длительности такта системной частоты.

Прерывание по уровню возникает до тех пор, пока устройством удерживается заданный уровень сигнала. Если прерывание по уровню не было снято до завершения работы обработчика прерываний, то контроллер NVIC вновь начинает его обработку.

В случае прихода импульсного прерывания от любого источника в момент обработки предыдущего, в контроллере NVIC устанавливается флаг, сигнализирующий о приходе нового прерывания, которое будет обработано после завершения обработки текущего прерывания. В случае, если контроллер NVIC находится в состоянии ожидания и приходит импульсное прерывание от того же источника, обработка выполнится только один раз.

Для управления прерываниями используются пять групп регистров ISER, ICER, ISPR, ICPR и IABR. Подробное описание приведено в приложении В.

10 Порты ввода-вывода

В состав микроконтроллера входят два 16-разрядных порта ввода-вывода: порт A и порт B. 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). Исключением являются выводы B0 – B3 порта B, которые имеют аналоговую функцию – с ними мультиплексированы каналы АЦП.

По умолчанию порты находятся в сбросе и не тактируются. Активировать порт A и порт B можно с помощью соответсвующих бит регистров HCLKCFG, HRSTCFG блока RCU.

10.1 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.

На рисунке 10.1 приведена структурная схема вывода цифрового порта микроконтроллера. Схемы всех выводов идентичны.

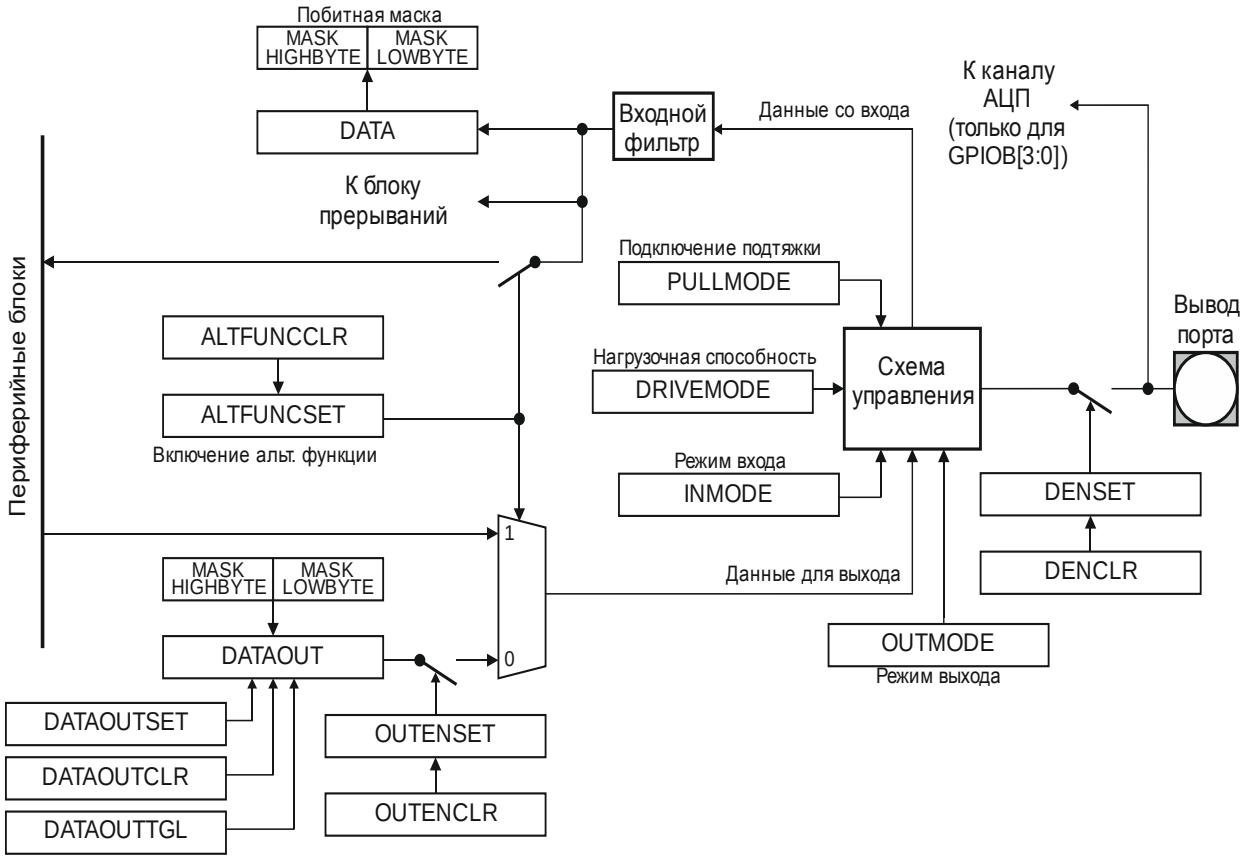


Рисунок 10.1 – Структурная схема вывода цифрового порта микроконтроллера

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

Для каждого вывода задается режим работы, 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, кроме выводов JTAG, конфигурируются как выводы общего назначения (режим GPIO) и находятся в третьем состоянии. Перед тем как взаимодействовать с выводом как цифровым входом, выходом или альтернативной функцией необходимо разрешить цифровую работу вывода, записав единицы в соответствующие разряды регистра DENSET. Для сброса установленных бит следует записать единицы в регистр DENCLR. При использовании аналоговой входной функции необходимо запретить цифровую работу всех соответствующих выводов, мультиплексированных с выбранными каналами АЦП.

Выбор нагрузочной способности и быстродействия вывода определяется полями регистра DRIVEMODE, а режим подтяжки 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.

Режим работы входной цепи настраивается с помощью регистра INMODE.

10.2 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.

Входы и выходы периферийных блоков в процессе работы коммутируются с выводами микроконтроллера при условии, что для этих выводов включен режим альтернативной функции.

10.3 Входные фильтры

Ко всем площадкам выводов подключены входные фильтры. На рисунке 10.2 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.

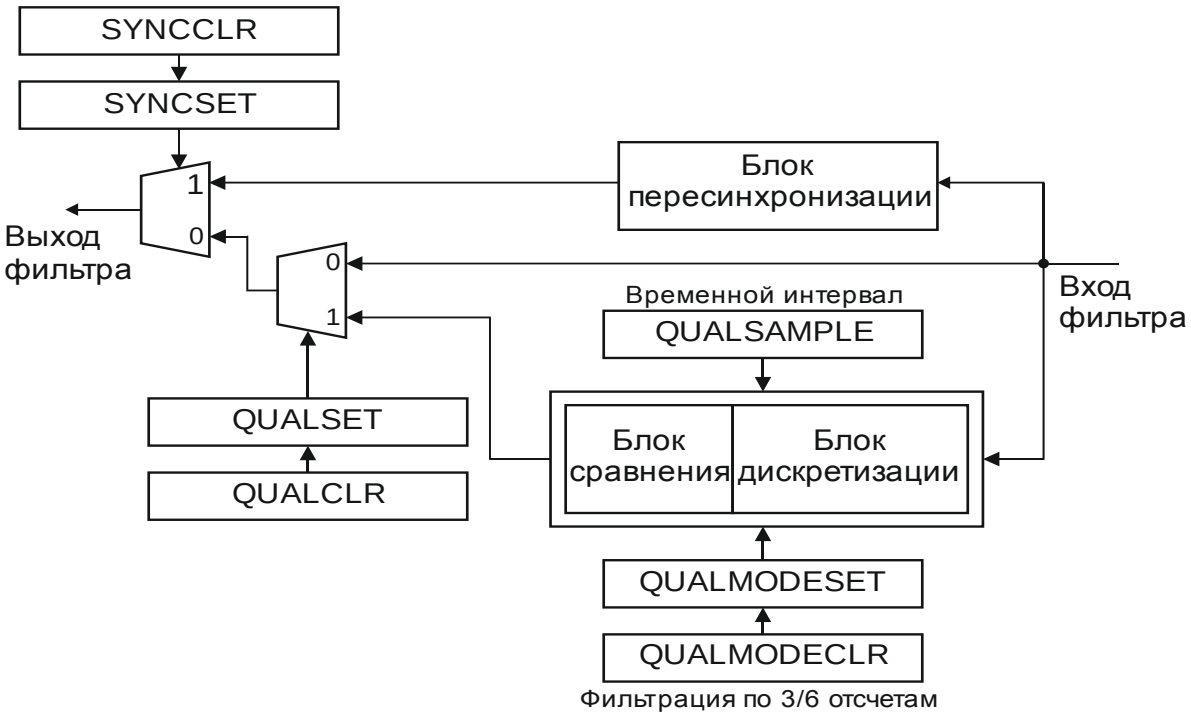


Рисунок 10.2 – Структурная схема фильтра вывода порта

10.4 Прерывания

Каждый из выводов способен генерировать прерывание. На рисунке 10.3 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 NVIC. Прерывание может быть сгенерировано программно записью единицы в соответствующий бит регистра INTSTATUS.

Прерывание может быть сброшено программно записью единицы в соответствующий бит регистра INTCLEAR. 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.

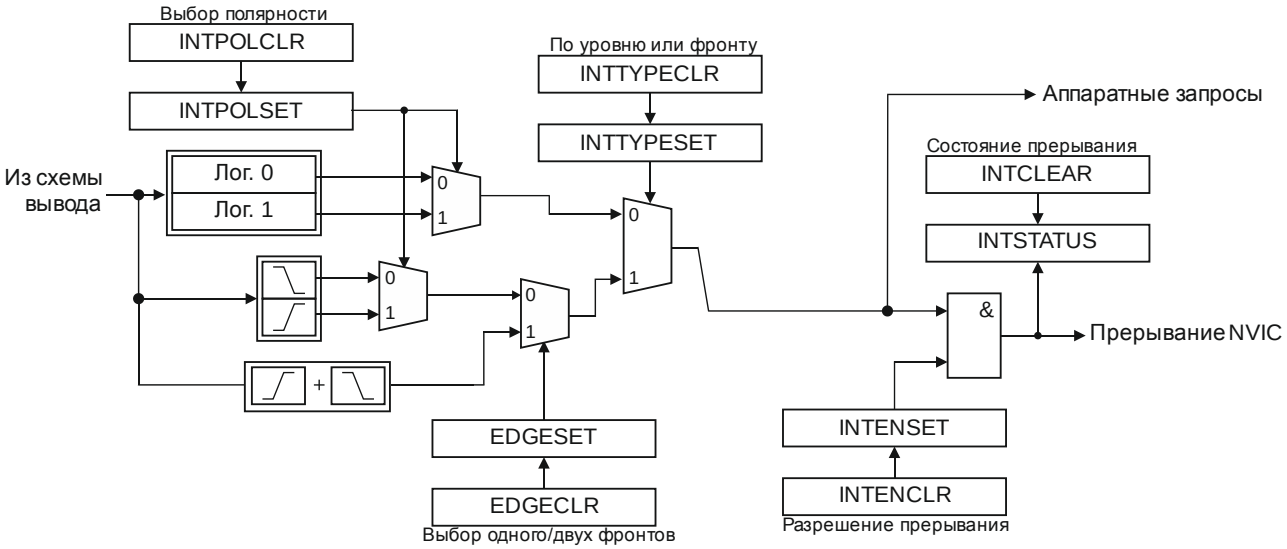


Рисунок 10.3 – Структурная схема блока генерации прерываний

10.5 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.

Для того чтобы в соответствии с настройками прерывания сгенерировался запрос RXEV к ядру, необходимо осуществить запись единиц в RXEVSET. Отключение генерации запросов RXEV к ядру осуществляется через RXEVCLR.

10.6 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.

10.7 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: 4001_0400h – 4001_07FCh для младшего байта и 4001_0800h – 4001_0BFCh для старшего байта. Биты с 9 по 2 адреса являются маской. Таким образом, адресу 4001_0400h соответствует маска 00h (MASKLB[0x00]), адресу 4001_0404h – маска 01h (MASKLB[0x01]) и т. д.

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

Разряды порта, закрытые «нулями» маски, останутся неизменными, а остальные примут новые значения. На рисунке 10.4 показан механизм маскирования младшего байта порта A.

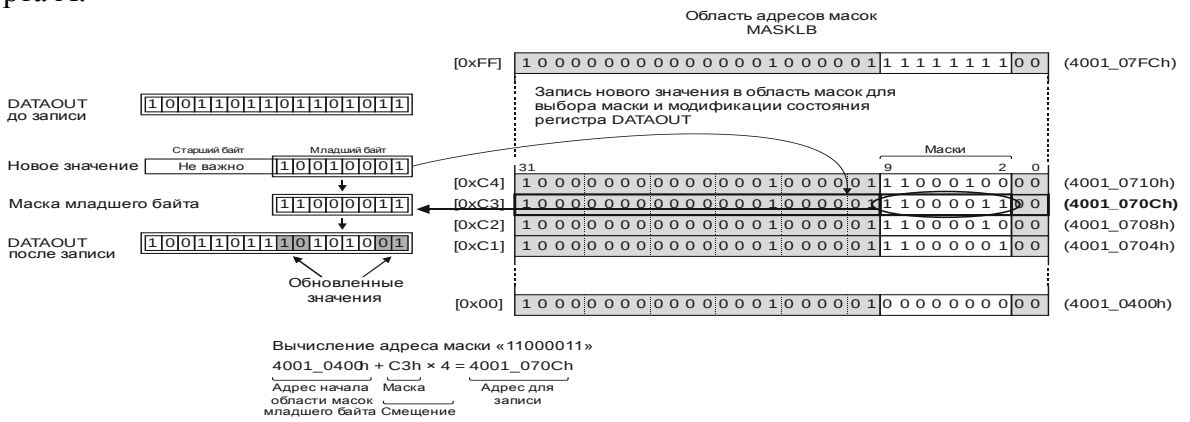


Рисунок 10.4 – Механизм изменения состояния младшего байта порта A с маскированием

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 4001_070Ch. Новое значение XX90h данных, которое требуется передать в порт (при этом старший байт числа не важен), нужно записать в ячейку с адресом 4001_070Ch. Далее это значение будет аппаратно маскировано и размещено в регистре порта DATAOUT.

Аналогично выполняется маскирование старшего байта, см. рисунок 10.5. Разница лишь в том, что в данном случае берется старший байт нового значения, а младший не важен.

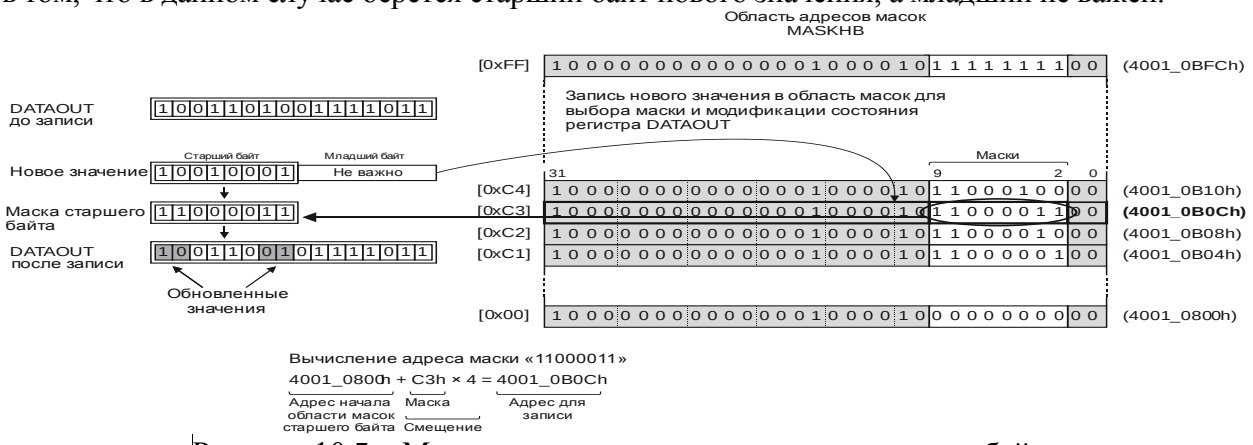


Рисунок 10.5 – Механизм изменения состояния старшего байта порта A с маскированием

11 Блоки таймеров

Микроконтроллер содержит четыре блока 32-разрядных таймеров. Все блоки идентичны.

Счетчик таймера работает по системному тактовому сигналу. Кроме этого, таймер может управляться внешним сигналом, а также синхронизироваться по внешнему сигналу. На рисунке 11.1 представлена функциональная схема таймера.

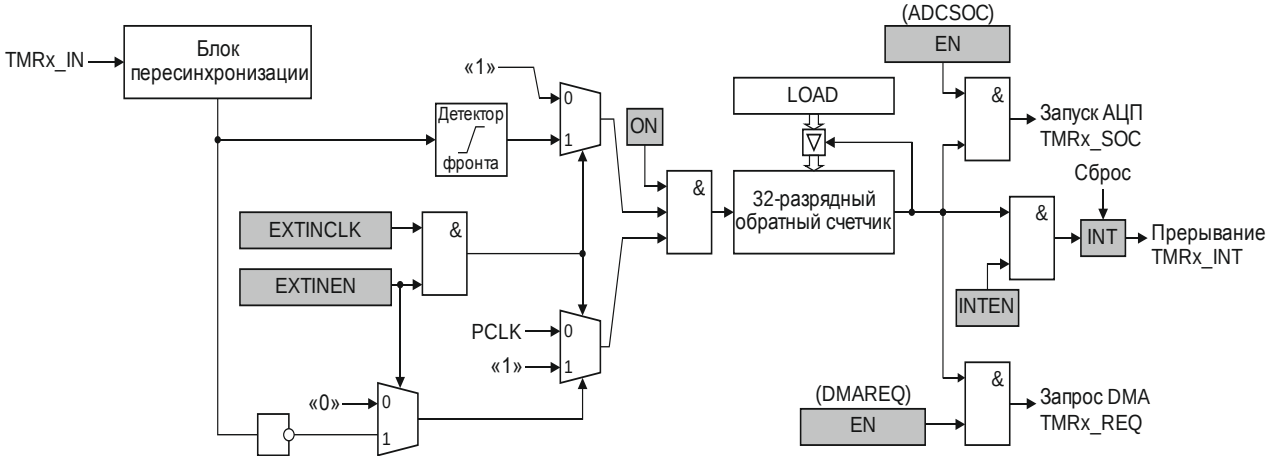


Рисунок 11.1 – Функциональная схема таймера

Управление таймером осуществляется посредством регистра CTRL. Начальное значение задается регистром VALUE. Для включения таймера нужно установить бит ON. Счетчик таймера декрементируется от значения заданного регистром VALUE до нуля на частоте тактового сигнала микроконтроллера. По достижении нуля счетчик таймера загружается значением, заданным регистром перезагрузки LOAD, и, если разрешено, битом INTEN генерируется прерывание. При возникновении прерывания устанавливается флаг INT в регистре INTSTATUS.

Для таймеров 0 и 1 доступен свой внешний вход синхронизации TMRx_IN (x = 0, 1), выведенный на альтернативную функцию порта микроконтроллера.

Если установлен бит EXTINEN, то счетчик таймера работает на частоте тактового сигнала микроконтроллера, только если сигнал на входе имеет уровень логической единицы.

Если одновременно установлены биты EXTINEN и EXTINCLK, то тактирование счетчика таймера происходит по положительному фронту внешнего сигнала, приходящего на вход TMRx_IN. При этом частота внешнего сигнала должна быть как минимум в два раза меньше частоты системного тактового сигнала.

С помощью таймера возможно генерирование запроса DMA. Если записать в поле EN регистра DMAREQ единицу, то по достижении счетчиком нуля будет генерироваться запрос по соответствующему каналу контроллера DMA. Выбор необходимого канала DMAi выполняется с помощью поля SRCSELx (x – номер канала от 8 до 15) регистра DMAMUX блока SIU.

Также таймер может выдавать строб для запуска измерения АЦП. Если записать в поле EN регистра ADCSOC единицу, то по достижении счетчиком нуля будет генерироваться сигнал запуска АЦП.

12 Блоки захвата

В микроконтроллере реализованы три блока захвата. Все блоки идентичны.

Блоки захвата используются для:

- вычисления скорости вращения вала ротора (с использованием датчиков Холла);
- вычисления промежутков времени между срабатыванием позиционных датчиков;
- вычисления периода и скважности импульсов.

Возможности блока захвата:

- 32-разрядный таймер, с разрешающей способностью 10 нс (на 100 МГц);
- четыре 32-разрядных регистра захвата времени;
- выбор полярности фронта для обработки каждого из четырех последовательных событий;
- источники прерываний по каждому из четырех событий;
- однократный захват значений времени до четырех событий;
- режим цикличной работы по событиям, с переписыванием значений (кольцевой буфер);
- режимы захвата абсолютного и относительного значений времени;
- альтернативный режим работы, если не задействована функция захвата времени – одноканальный выход ШИМ.

Функциональная схема блока захвата представлена на рисунке 12.1.

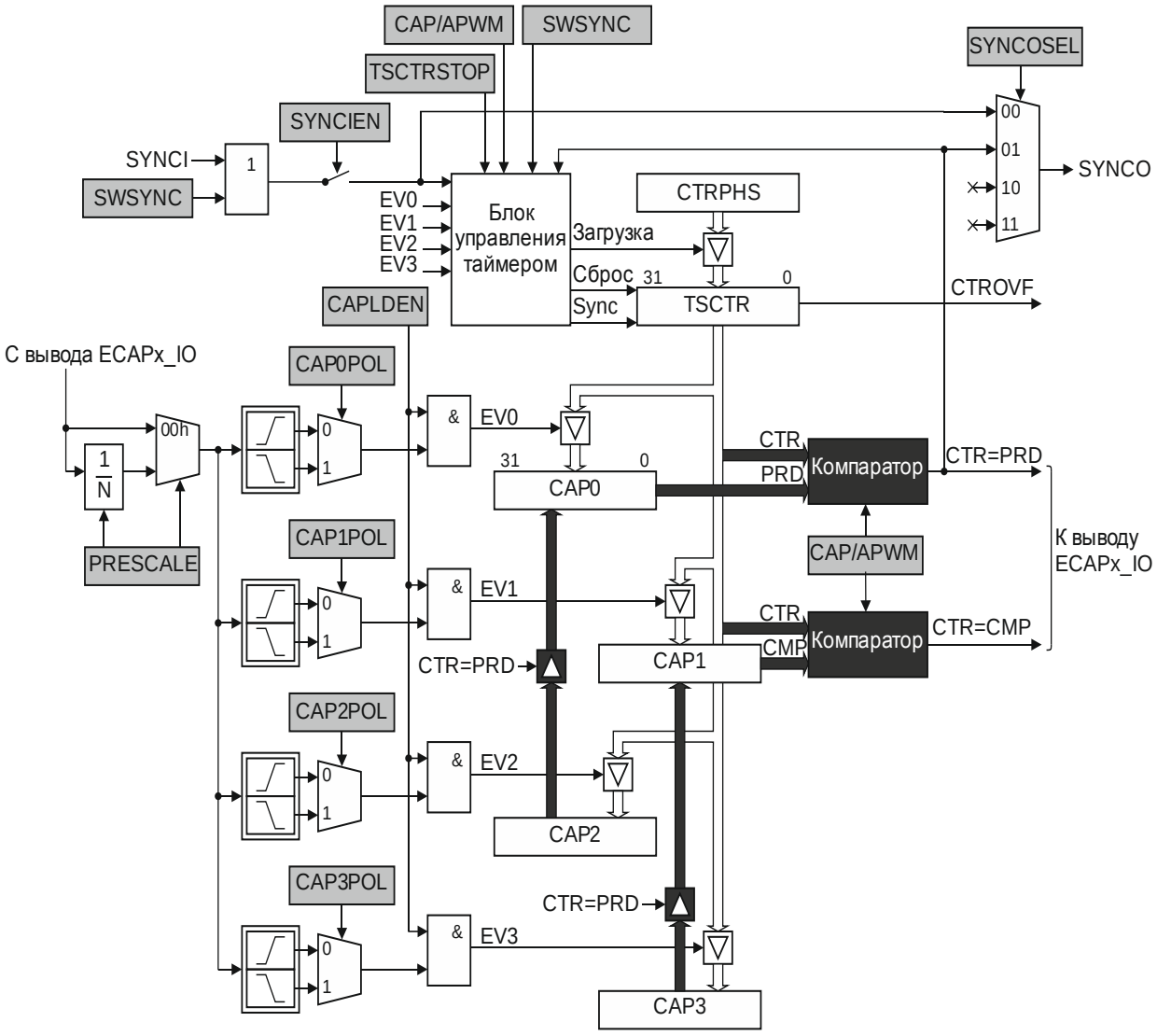


Рисунок 12.1 – Функциональная схема блока захвата

Для начала работы с блоками захвата необходимо подать тактирование – установить соответсвующие биты в регистре PCLKCFG блока RCU и снять сброс – установить соответствующие биты регистра PRSTCFG.

Каждый блок захвата ECAPx имеет один вывод ECAPx_IO (где x = 0, 1, 2), соединенный с выводом микроконтроллера (альтернативная функция). В зависимости от режима работы блока захвата вывод является входом внешнего события или выходом генерируемого сигнала ШИМ. Чтобы переключить альтернативную функцию на сигналы блоков ECAPx по соответсвующим выводам, необходимо установить бит ECAPxEN в регистре REMAPAF блока SIU.

12.1 Режим захвата времени

Режим захвата времени выбран по умолчанию. Вывод ECAPx_IO функционирует как вход.

Предварительный делитель

В случае если события на входе ECAPx_IO приходят слишком часто и требуется уменьшить их частоту, используется предварительный делитель событий (импульсов), состоящий из собственно делителя и мультиплексора. В предварительном делителе используется счетчик, который производит выборку одного события из каждых 2 – 63 входных. Значение делителя задается полем PRESCALE регистра ECCTL0. В случае если задано значение 00h, то делитель выключен и входной сигнал поступает на детекторы фронта напрямую.

Для примера на рисунке 12.2 показаны несколько вариантов сигналов на выходе делителя, в зависимости от заданного значения N.

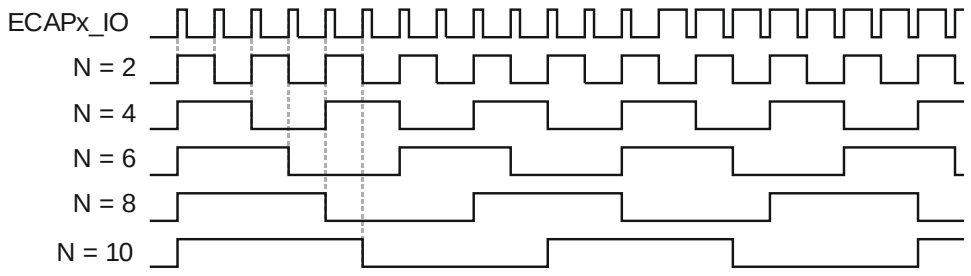


Рисунок 12.2 – Формы сигналов на выходе делителя в зависимости от значения N

Захват значения таймера

С выхода делителя сигнал поступает на четыре детектора фронта, каждый из которых управляется соответствующим битом CAPnPOL. Далее, если установлен бит CAPLDEN и обнаружен заданный фронт сигнала, формируется соответствующее событие. Возможно одновременное формирование до четырех событий (EV0 – EV3), и по переднему фронту каждого события происходит захват значения таймера TSCTR в соответствующий регистр захвата (CAP0 – CAP3).

Регистр захвата перезаписывается новым захваченным значением каждый раз при возникновении соответствующего события.

Однократный захват

Однократный захват выбирается битом CONTOST регистра ECCTL1 и включается записью единицы в бит REARM с последующей установкой бита CAPLDEN (установится аппаратно). В этом режиме происходит запуск двухразрядного счетчика событий EV0 – EV3. Количество подсчитываемых событий от одного до четырех задается полем STOPWRAP. Подсчитывается каждое из сформированных событий и одновременно происходит захват значения таймера в соответствующие регистры захвата. Как только

количество событий совпадет со значением STOPWRAP, события EV0 – EV3 больше не формируются, значение таймера захватывается регистрами CAP0 – CAP3, и далее регистры не перезаписываются.

Для повторного запуска следует записать единицу в бит REARM (это обнулит счетчик и включит режим), после чего – разрешить формирование событий EV0 – EV3 установкой бита CAPLDEN.

Циклический захват

Циклический захват выбран по умолчанию. После установки бита CAPLDEN начинается обработка событий EV0 – EV3 и захват значения таймера. Количество подсчитываемых событий от одного до четырех задается полем STOPWRAP.

Регистр захвата перезаписывается новым захваченным значением каждый раз при возникновении соответствующего события.

Примечение – В обоих режимах захвата значение поля STOPWRAP не оказывает никакого влияния на счетчик и состояние бита CAPLDEN.

Таймер

Таймер представляет собой 32-разрядный счетчик, работающий на системной частоте. Контроль работы таймера осуществляет блок управления таймером. Счетчик таймера включается битом TSCTRSTOP и инкрементируется, начиная со значения 0000_0000h до значения FFFF_FFFFh, после чего сбрасывается.

Чтобы синхронизировать работу таймера с другими блоками, счетчик таймера может быть в любой момент загружен новым значением, которое предварительно записывается в теневой регистр CTRPHS. Загрузка может быть активирована как программно – запись единицы в бит SWSYNC регистра ECCTL1, так и аппаратно – приход синхроимпульса по входу SYNCI. Разрешение синхронизации осуществляется установкой бита SYNCIEN.

Блок захвата также может генерировать сигнал синхронизации SYNCO. Источник выбирается полем SYNCOSEL либо используется SYNCI или SWSYNC, либо событие CTR = PRD генерации ШИМ.

Входы и выходы синхронизации блоков захвата подключены по цепочке, см. рисунок 12.3.

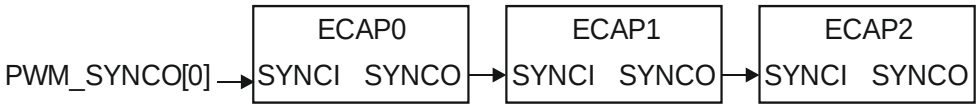


Рисунок 12.3 – Подключение сигналов синхронизации блоков захвата

Таймер может быть сброшен (с предварительным захватом его значения) при формировании событий EV0 – EV3. Указать событие можно установкой соответствующего бита CTRRSTn (где n от 0 до 3) в регистре EECTL0. Так, например, если установлен бит CTRRST2, то при формировании события EV2 произойдет захват значения таймера в регистр CAP2 и сброс таймера.

Регистры CAP0 – CAP3

32-разрядные регистры CAP0 – CAP3, сохраняющие (захватывающие) значение счетчика таймера в момент появления положительного фронта сигнала события EVn (где n от 0 до 3), доступны только для чтения.

12.2 Режим работы «генератор ШИМ»

Режим работы «генератор ШИМ» выбирается установкой бита CAPAPWM в регистре ECCTL1. Вывод ECAPx_IO (где x = 0, 1, 2) функционирует как выход. Блок захвата в этом случае используется как одноканальный 32-разрядный генератор сигнала ШИМ.

Таймер и регистры захвата

Таймер функционирует как 32-разрядный инкрементный счетчик, работающий на системной частоте. После включения счетчик таймера считает от значения 0000_0000h до значения, которое задается регистром CAP0. Как только значения счетчика и регистра совпадают, счетчик сбрасывается.

Регистр CAP0 является регистром периода таймера, а регистр CAP1 – регистром сравнения. Регистры CAP2 и CAP3 являются регистрами отложенной загрузки для регистров CAP0 и CAP1 соответственно. Все регистры доступны как для записи, так и для чтения.

Запись в регистр CAP0 является мгновенной загрузкой, которая аппаратно дублируется записью в регистр CAP2. Аналогично для пары CAP1, CAP3.

Запись в регистры CAP2 и CAP3 является отложенной загрузкой. Как только значение счетчика таймера достигает значения периода CAP0, возникает событие CTR = PRD, по которому происходит сброс таймера и перегрузка значений из CAP2 и CAP3 в регистры CAP0 и CAP1 (на рисунке 12.1 отмечено стрелками черного цвета).

Регистры CAP0 и CAP1 должны быть обязательно инициализированы до начала запуска таймера. При дальнейшей работе можно изменять значения только регистра отложенной загрузки.

Генерация ШИМ

После инициализации регистров CAP0 и CAP1 запускается счетчик таймера. Текущее значение счетчика CTR посредством двух компараторов сравнивается одновременно со значением PRD регистра периода CAP0 и значением CMP регистра сравнения CAP1 (на рисунке 12.1 отмечено черным цветом).

Как только возникает событие CTR = CMP, сигнал на выходе ECAPx_IO переводится в ноль. Далее сигнал удерживается в нуле до тех пор, пока счетчик таймера не достигнет значения периода. При возникновении события CTR = PRD сигнал переводится в единицу. Одновременно с этим происходит сброс таймера и перезагрузка регистров CAP0 и CAP1. Управлять полярностью сигнала можно битом APWMPOL. На рисунке 12.4 представлен пример формирования сигнала ШИМ с активным высоким уровнем сигнала (по умолчанию APWMPOL = 0).

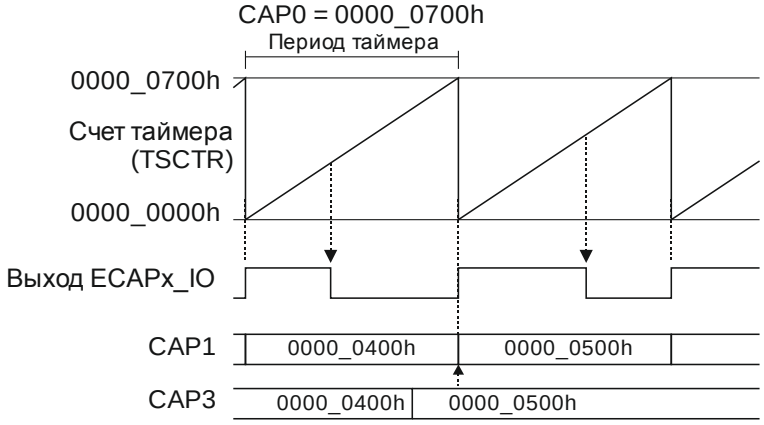


Рисунок 12.4 – Пример формирования сигнала ШИМ с активным высоким уровнем сигнала (по умолчанию APWMPOL = 0)

На рисунке 12.4 период таймера задан как CAP0 = 0000_0700h. Начальное значение сравнения CAP1 = 0000_0400h. Пока таймер считает, в регистр CAP3 загружается новое значение 0000_0500h для отложенной загрузки. По достижении значения сравнения сигнал на выходе ECAPx_IO переводится в низкий уровень. По окончании периода происходит сброс таймера и загрузка значения 0000_0500h (из регистра CAP3) в регистр CAP1 и перевод сигнала на выходе ECAPx_IO в высокий уровень.

Таким образом, можно достаточно гибко управлять как длительностью импульсов, изменяя период работы таймера, так и скважностью при постоянном периоде.

12.3 Прерывания

Источники прерываний блока захвата:
- события EV0 – EV3;
- переполнение счетчика таймера CTROVF;
- события CTR = PRD;
- события CTR = CMP.

Каждое из семи прерываний имеет бит маски в регистре ECEINT, флаг прерывания в регистре ECFLG, бит сброса флага в регистре ECCLR и бит программного прерывания в регистре ECFRC. На рисунке 12.5 показан пример для прерывания по событию EV1.

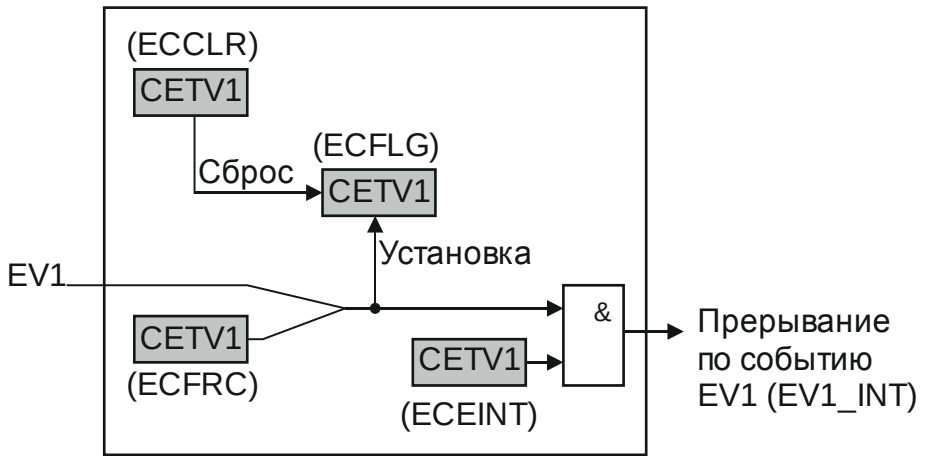


Рисунок 12.5 – Пример управления прерыванием EV1

Все прерывания по событиям поступают на блок управления прерываниями и обрабатываются, как сказано выше. При возникновении любого из этих прерываний в регистре PEINT устанавливается флаг PEINT, и генерируется прерывание блока захвата, см. рисунок 12.6.

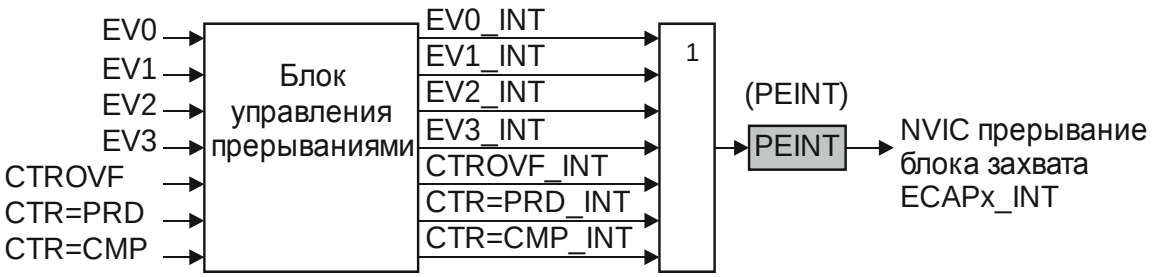


Рисунок 12.6 – Общая схема управления прерываниями

Примечание – Программа обслуживания прерывания должна сбрасывать флаг PEINT во избежание повторного обслуживания прерывания от блока захвата. Для сброса флага следует записать единицу в нулевой бит регистра PEINT.

13 Модуль квадратурного декодера QEP

Квадратурный декодер QEP преобразует цифровой сигнал с датчика положения вала, позволяя вычислять скорость, направление вращения, а также текущее положение вала.

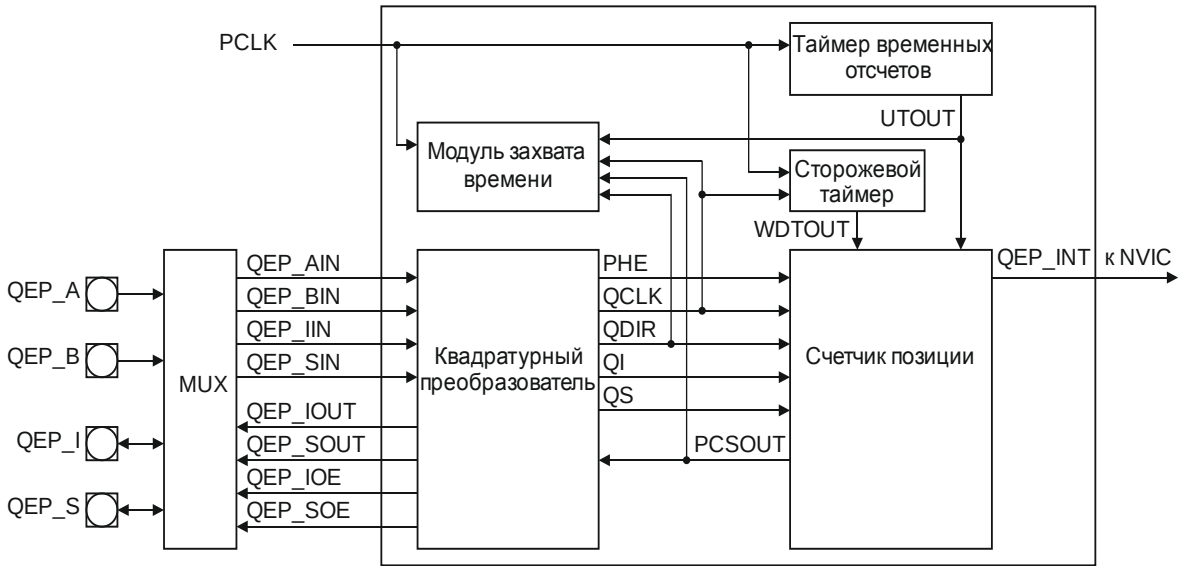


Рисунок 13.1 – Схема квадратурного декодера с мультиплексором входных/выходных сигналов

В состав квадратурного декодера QEP входят, см. рисунок 13.1:
- настраиваемый обработчик сигналов входов;
- квадратурный преобразователь;
- счетчик позиции/блок управления;
- модуль захвата времени;
- таймер временных отсчетов;
- сторожевой таймер.

По умолчанию модуль захвата времени находится в сбросе и не тактируется. Разрешить тактирование и снять сброс можно, установив соответсвующие биты в регистрах PCLKCFG и PRSTCFG блока RCU.

Чтобы переключить альтернативную функцию на сигналы QEP по соответсвующим выводам, необходимо установить бит QEPEN в регистре REMAPAF блока SIU.

13.1 Обработчик сигналов входов

Квадратурный декодер QEP использует два квадратурных вывода микроконтроллера, работающих на вход. Также, имеются специальный индексный вывод и вывод стробирования, которые могут работать на вход и выход.

QEP_A и QEP_B – в квадратурном режиме – это два входа. Сигналы на входах сдвинуты по фазе на 90 градусов и по ним можно определить скорость и направление вращения ротора, см. рисунок 13.2. В режиме счета/направления сигналы на входах используются как тактовый и сигнал направления вращения ротора, по которым также можно вычислить скорость вращения.

QEP_I – индексный вывод. Сигнал на входе сигнализирует о полном обороте ротора. Позволяет сбрасывать счетчик позиции поворота ротора.

QEP_S – пользовательский вывод стробирования. Сигнал на входе может сбросить или защелкнуть счетчик позиции. Применяется при использовании концевых выключателей.

Сигналы на входах могут быть проинвертированы. Инверсия включается установкой соответствующего бита в регистре QDECCTL.

Установкой бита SWAP можно включить обратный счет, т. е. программно подать сигнал с вывода QEP_A на вход QB квадратурного преобразователя, а сигнал с вывода QEP_B подать на вход QA (входы A и B меняются местами).

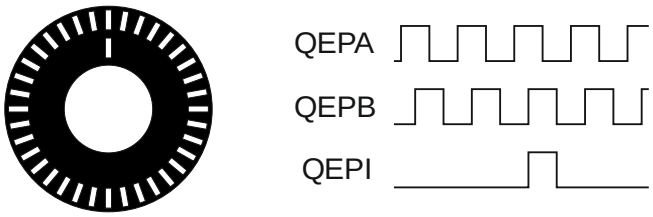


Рисунок 13.2 – Диаграммы входных сигналов

13.2 Квадратурный преобразователь

На рисунке 13.3 показана схема квадратурного преобразователя.

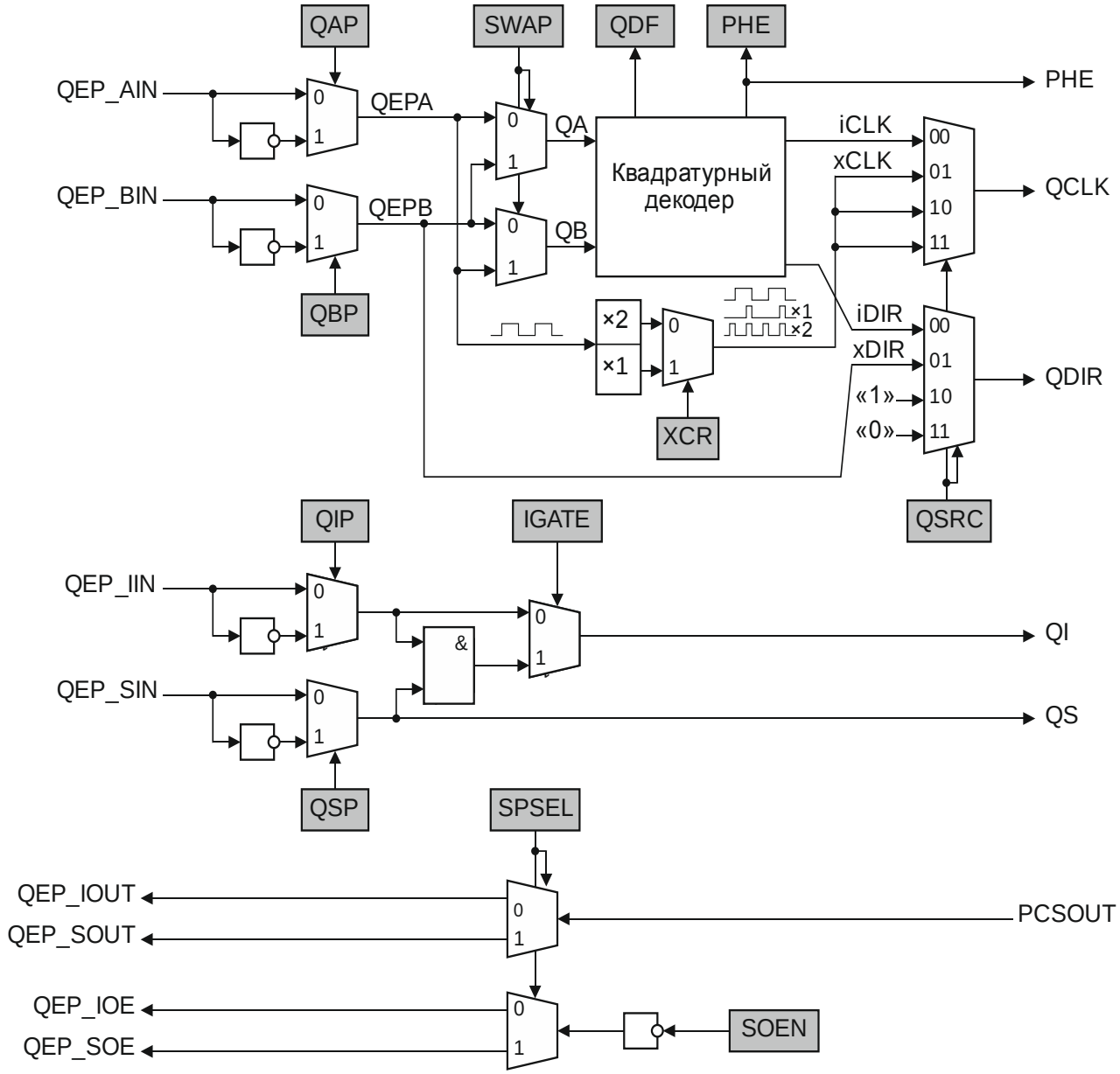


Рисунок 13.3 – Схема квадратурного преобразователя

Режимы работы

Квадратурный преобразователь поддерживает четыре режима работы:

- режим квадратурного счета;
- режим счета/направления;
- режим счета вверх;
- режим счета вниз.

Выбор режима зависит от значения поля QSRC регистра QDECCTL.

Режим квадратурного счета

Квадратурный преобразователь формирует сигнал направления вращения, тактовый сигнал и сигнал направления счета (вверх-вниз) для счетчика позиции.

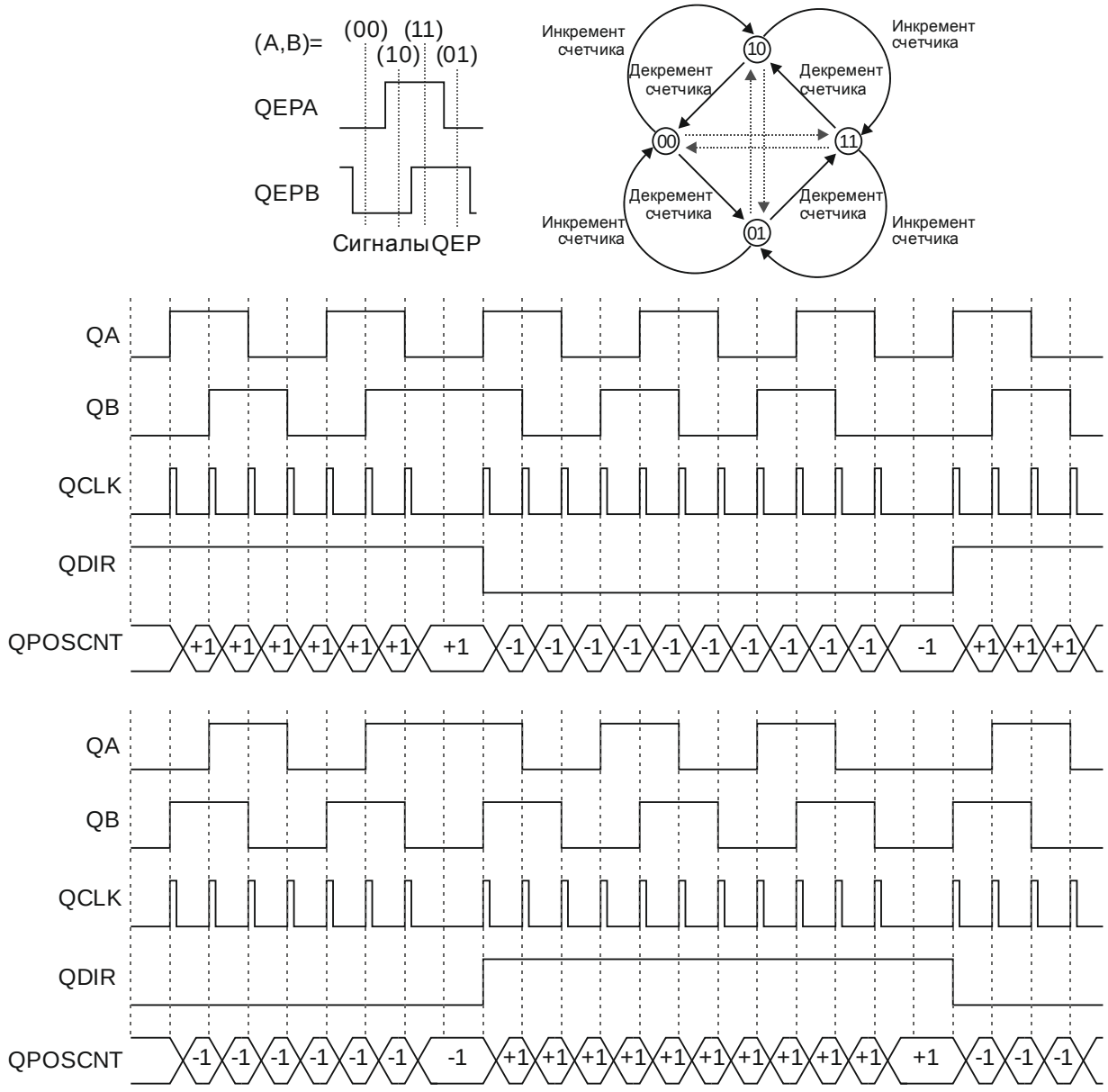


Рисунок 13.4 – Временные диаграммы и автомат состояний работы в квадратурном режиме счета

Направление вращения ротора определяется по порядку смены передних и задних фронтов на входах QEP_A и QEP_B. К примеру, если за передним фронтом сигнала на входе QEP_A следует передний фронт сигнала на входе QEP_B, см. рисунок 13.4, то направление вращения следует считать прямым, а счетчик позиции работает на увеличение. Если же за передним фронтом сигнала на входе QEP_B следует передний

фронт сигнала на входе QEP_A, то направление вращения следует считать инверсным, а счетчик позиции работает на уменьшение. Если на обоих выводах зафиксировано одновременно два фронта, то такое состояние считается ошибочным.

Квадратурный преобразователь выдает четыре счетных импульса на один период входного сигнала, поскольку использует для счета передний и задний фронты сигналов.

Режим счета/направления

В этом режиме вывод QEP_A работает как вход тактовых импульсов, а вывод QEP_B – как вход задания направления счета. Счетчик позиции в этом режиме работает по каждому переднему фронту сигнала на входе QEP_A.

Режим вверх

Режим используется для вычисления частоты следования импульсов на вывод QEP_A. Фронт задается битом XCR регистра QDECCTL. Счетчик всегда работает на увеличение.

Режим вниз

Режим используется для вычисления частоты следования импульсов на выводе QEP_A. Фронт задается битом XCR. Счетчик всегда работает на уменьшение.

13.3 Счетчик позиции

Работа счетчика позиции контролируется посредством регистров QEPCTL и QPOSCTL, которыми задается режим счета, сброса и хранения, а также логика для формирования внешнего сигнала синхронизации.

Режимы сброса счетчика позиции

Счетчик позиции может накапливать результат в течение многих оборотов вала, а может подсчитывать позицию только за один оборот, сбрасываясь каждый раз по событию прихода индексной метки. В зависимости от назначения могут использоваться следующие способы сброса счетчика позиции:

- по сигналу индексации;
- по переполнению;
- только по первому сигналу индексации;
- по таймеру временных отсчетов.

Режим задается полем PCRM регистра QEPCTL.

Счетчик сбрасывается в ноль при его переполнении или при превышении значения регистра максимального значения QPOSMAX. Флаг прерывания, возникающего при переполнении счетчика, устанавливается в регистре QFLG.

Режим сброса по сигналу индексации

Режим сброса по сигналу индексации включен по умолчанию.

При получении сигнала с индексного вывода QEP_I при счете вверх счетчик обнулится по следующему фронту сигнала тактирования QCLK. Если же сигнал индексации был получен при счете вниз, то в счетчик будет загружено значение QPOSMAX, см. рисунок 13.5.

При получении первого сигнала индексации, схема дожидается любого изменения на квадратурных входах и запоминает значение этого события – фронт, активный вывод QEP_A или QEP_B, а также направление вращения. Этот момент времени называется маркером индексации. При появлении этого события устанавливается бит FIMF регистра QEPSTS, а направление вращения сохраняется в бите FIDF. В дальнейшем, сброс счетчика будет производиться только в присутствии сигнала индексации и соответствия сохраненным значениям маркера индексации. Если направление вращения изменится, то

для формирования сброса сохраненное в маркере значение фронта (передний или задний фронт) меняется на обратное. Это сделано с целью привязки индексации к квадратурному сигналу QA/QB, а также для более точной обработки индексации, чтобы исключить влияние ширины импульса на индексный вывод. Сохраненные значения используются при сбросе по маркеру индексации (если поле IEL в регистре QEPCTL равно 11b).

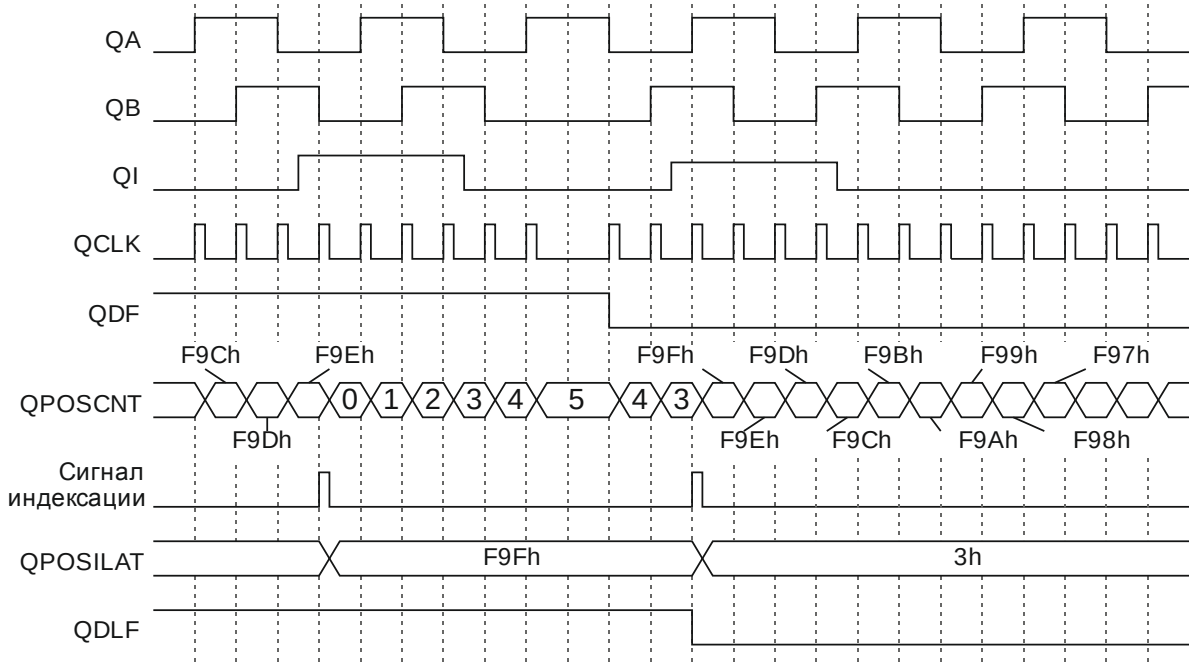


Рисунок 13.5 – Временные диаграммы сброса по сигналу индексации

По каждому сигналу индексации, включая маркер индексации, содержимое счетчика сохраняется в регистре QPOSILAT, а направление вращения – в бите QDLF регистра QEPSTS. Если при сохранении значение счетчика QPOSCNT не равно ни нулю, ни значению QPOSMAX, то выставляется флаг ошибки счетчика позиции (бит PCEF в регистре QEPSTS) и флаг прерывания (бит PCE в регистре QFLG). Флаг ошибки счетчика позиции обновляется с каждым индексом, а флаг прерывания может быть сброшен только программно.

Поле настройки события индексации для сохранения счетчика позиции IEL (регистр QEPCTL) игнорируется. Также только в этом режиме могут устанавливаться флаг ошибки счетчика позиции PCEF и флаг соответствующего прерывания.

Режим сброса по переполнению

Максимальное значение счетчика позиции задается регистром QPOSMAX. Если счетчик считает вверх и достигнуто максимальное значение, то со следующим тактом синхросигнала счетчик обнулится. Если счетчик считает вниз и достигнуто значение нуля, то со следующим тактом синхросигнала в счетчик будет загружено значение QPOSMAX. Сброс по событию индексации не производится.

Получение значений маркера индексации происходит аналогично тому, как это происходит в режиме сброса по сигналу индексации. Полученные значения могут использоваться при инициализации по маркеру индексации, если в поле IEL записано значение 11b в регистре QEPCTL.

Временные диаграммы сброса показаны на рисунке 13.6.

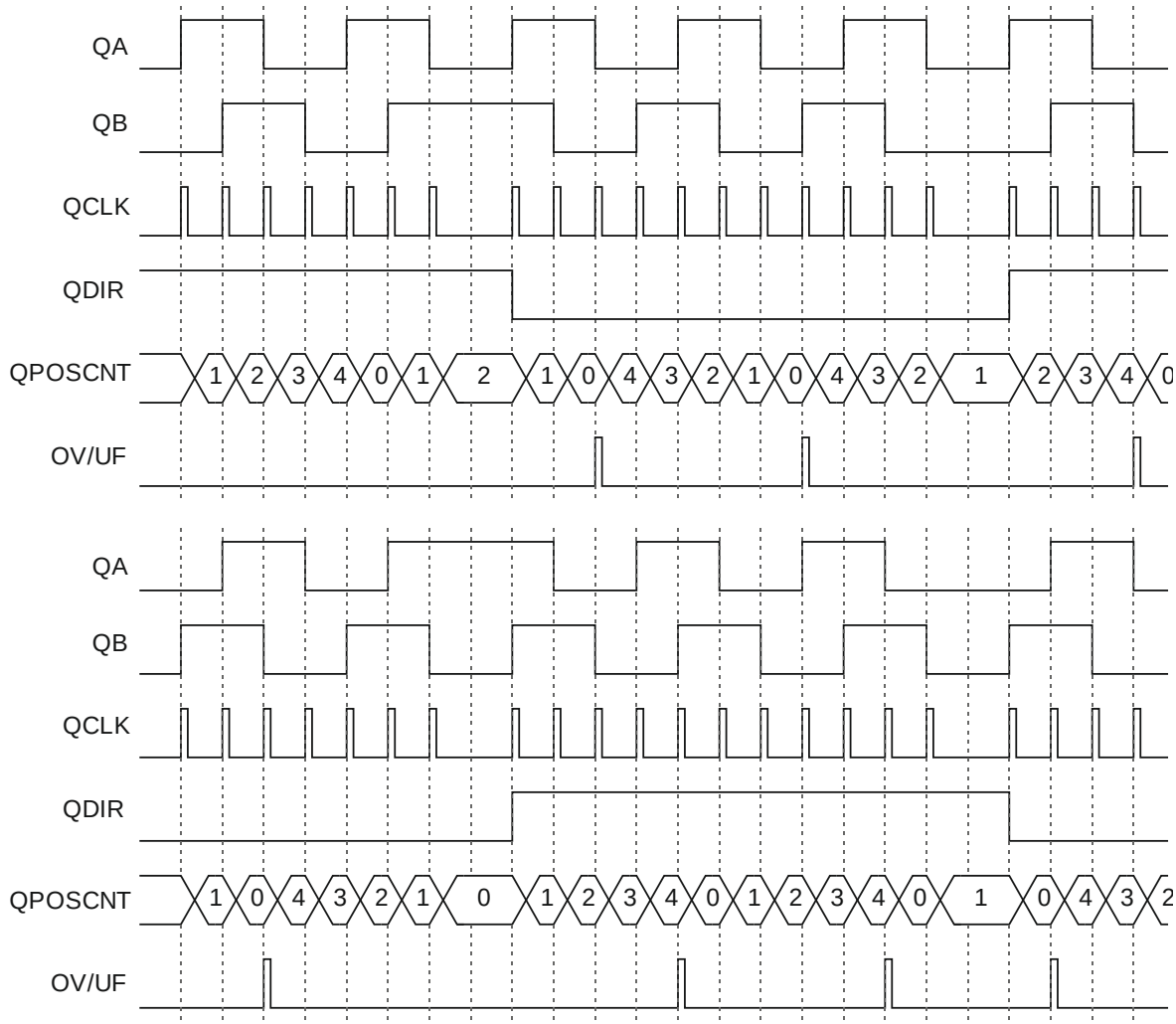


Рисунок 13.6 – Временные диаграммы сброса по сигналу переполнения

Режим сброса по первому сигналу индексации

Если было получено событие индексации при счете вверх, то счетчик обнулится со следующим тактом синхросигнала. Если же событие индексации было зафиксировано при счете вниз, то со следующим тактом синхросигнала в счетчик будет загружено значение QPOSMAX. При последующем счете сброс может произойти только при достижении нуля или значения QPOSMAX (т. е. аналогично режиму сброса по переполнению), а дальнейшие возможные события получения сигнала на выводе индексации влиять на сброс не будут.

Получение значений маркера индексации происходит аналогично тому, как это происходит в режиме сброса по сигналу индексации. Полученные значения могут использоваться при инициализации по маркеру индексации, если в поле IEL записано значение 11b.

Режим сброса по таймеру временных отсчетов

В этом режиме счетчик сбрасывается в ноль или загружается значением QPOSMAX, в зависимости от текущего режима счета (задается полем QSRC регистра QDECCTL), по событию срабатывания таймера временных отсчетов. В остальном режим аналогичен режиму сброса по переполнению.

Также возможно настроить сохранение значения счетчика QPOSCNT в регистр QPOSLAT перед сбросом, для этого необходимо включить модуль захвата, установив бит CEN в регистре QCAPCTL. Этот режим удобен для измерения частоты.

Сохранение счетчика позиции

Внешние входы индексации и стробирования можно запрограммировать на формирование событий для сохранения значения счетчика позиции в регистры QPOSILAT и QPOSSLAT.

Сохранение по событию индексации

В некоторых задачах не требуется сбрасывать счетчик позиции по каждому сигналу индексации, а вместо этого может потребоваться увеличить разрядность счетчика до 32 бит (режимы, задаваемые значениями PCRM равными 01b и 10b). В этом случае бит QDLF (направление вращения) в регистре QEPSTS будет перезаписываться по каждому сигналу индексации, а счетчик будет сохранять значение по следующим событиям индексации:

- по переднему фронту сигнала индексации при IEL = 01b;
- по заднему фронту сигнала индексации при IEL = 10b;
- по маркеру индексации при IEL = 11b.

Сохранение значения счетчика по маркеру индексации будет производиться только в присутствии сигнала индексации и по событию, эквивалентному сохраненному при первой индексации по маркеру. Если направление вращения изменится, то сохраненное в маркере значение типа фронта меняется на обратное значение. Это сделано с целью привязки индексации к квадратурному сигналу QA/QB, а также для более точной обработки индексации, чтобы исключить влияние ширины импульса на индексном выводе.

При сохранении значения счетчика в регистр QPOSILAT формируется флаг IEL прерывания индексации в регистре QFLG. В режиме сброса по сигналу индексации (PCRM = 00h) значение поля IEL в регистре QEPCTL игнорируется.

Сохранение по событию стробирования

Значение счетчика сохраняется в регистр QPOSSLAT по каждому переднему фронту сигнала на входе QEP_S, если сброшен бит SEL в регистре QEPCTL. Если же бит SEL установлен, то сохранение в регистре QPOSSLAT происходит по переднему фронту сигнала строба на входе QEP_S при прямом направлении вращения и по заднему фронту для обратного вращения. При каждом сохранении счетчика в регистре QPOSSLAT устанавливается флаг прерывания SEL.

Инициализация счетчика позиции

Счетчик событий может быть проинициализирован программно или по событиям:

- событие индексации;
- событие стробирования.

Входной сигнал индексации QEPI может использоваться для инициализации счетчика по переднему и заднему фронту. Если поле IEI = 10b, то счетчик загружается значением регистра QPOSINIT по переднему фронту сигнала индексации. Аналогично, если IEI = 11b, то счетчик загружается значением QPOSINIT по заднему фронту сигнала индексации.

Если поле SEI = 10b, то счетчик загружается значением регистра QPOSINIT по переднему фронту сигнала стробирования на входе QEP_S. Если SEI = 11b, то счетчик загружается значением регистра QPOSINIT по переднему фронту сигнала стробирования, если идет счет вверх, и по заднему – если вниз.

Программно счетчик инициализируется при записи единицы в бит SWI регистра QEPCTL. Бит не сбрасывается автоматически, но повторная запись единицы также приведет к инициализации счетчика.

Компаратор счетчика позиции

Компаратор, см. рисунок 13.7, сравнивает значение счетчика позиции с регистром QPOSCMP и при совпадении значений формирует прерывание, а также внешний синхросигнал, который может быть направлен на один из выводов: индексный вывод QEP_I или вывод стробирования QEP_S. Бит SPSEL в регистре QDECCTL определяет, на какой именно вывод будет направлен сигнал синхронизации, а бит SOEN в регистре QDECCTL разрешает этому выводу работать как выход.

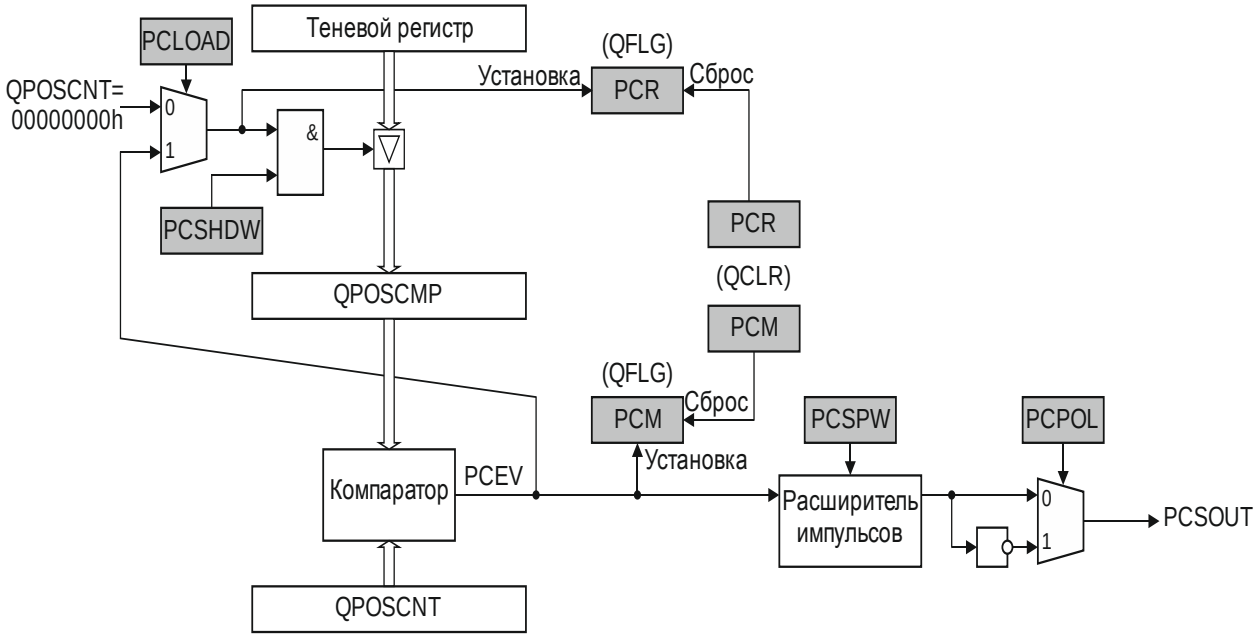


Рисунок 13.7 – Функциональная схема компаратора счетчика позиции

Регистр QPOSCMP может использовать режим отложенной загрузки, когда отложенное значение берется из теневого регистра, а если режим отложенной загрузки выключен, то запись в QPOSCMP производится сразу в активный регистр.

Отложенная загрузка происходит по следующим событиям:

- по совпадению результатов сравнения;
- по обнулению счетчика QPOSCNT.

Флаг успешного сравнения PCM устанавливается, когда выполняется условие QPOSCNT = QPOSCMP, при этом также формируется синхроимпульс требуемой длительности для извещения внешнего устройства (сигнал PCSOUT). Настраиваемая длительность синхроимпульса контролируется специальной схемой задержки.

Флаг PCR готовности компаратора к отложенной загрузке значения сравнения выставляется, когда выполняется условие для отложенной записи, заданное битом PCLOAD в регистре QPOSCTL. При этом генерация соответсвующего прерывания и установка флага PCR будет осуществляться лишь при включении режима отложенной загрузки PCSHDW (регистр QPOSCTL).

13.4 Таймер временных отсчетов

Таймер, используемый для оповещения программного обеспечения о необходимости начать измерение скорости, представляет собой 32-разрядный таймер, работающий на частоте системного тактового сигнала. Таймер включается установкой бита UTE в регистре QEPCTL. Когда значение таймера достигает порога (QUTMR = QUPRD), формируется прерывание и выставляется флаг UTO. Данный блок таймера может быть использован для вычисления скорости при высоком быстродействии, см. рисунок 13.8, а также для сохранения счетчика позиции, регистра таймера и регистра периода в регистрах QPOSLAT, QCTMRLAT и QCPRDLAT соответственно. Режим сохранения определяется состоянием бита QCLM в регистре QEPCTL.

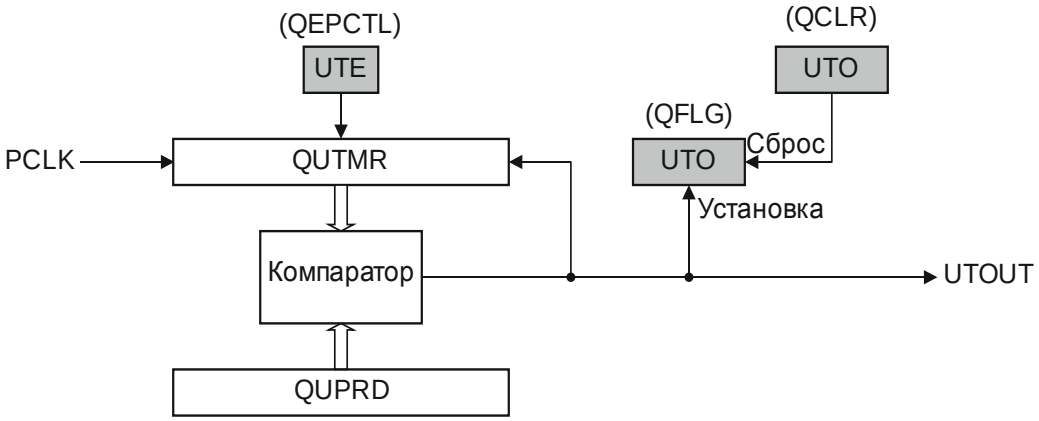


Рисунок 13.8 – Функциональная схема таймера временных отсчетов

13.5 Модуль захвата времени

Функциональная схема модуля захвата времени представлена на рисунке 13.9. Таймер использует тактовый сигнал и сигнал квадратурных событий с коэффициентами деления, программируемыми полями CCPS и UPPS в регистре QCAPCTL. Коэффициенты деления можно менять в процессе работы, но в этом случае может произойти событие захвата, содержащее неверные данные, которые следует игнорировать. Во избежание подобной ситуации перед изменением значений коэффициентов деления следует выключить модуль захвата времени (сбросить бит CEN), и снять все маски прерываний. После изменения коэффициентов проинициализировать таймер QCTMR (записать нулевое значение), сбросить все статусы, вновь разрешить прерывания и включить модуль захвата.

Также, существует возможность включить улучшенный режим теневой загрузки UPPS/CCPS. Если установить бит EPSLD в регистре QCAPCTL, то изменение UPPS/CCPS в процессе работы будет происходить следующим образом:

- при записи, значение UPPS/CCPS занесется в теневой регистр
- по ближайшему, после записи, событию QCLK: новое значение делителя примет силу, сбросится QCTMR, сбросится счетчик предделителя UPPS, сбросится UPEVNT.

Таким образом, в этом режиме – следующее событие (установка UPEVNT и захват QCTMR в QCPRD), которое будет следовать после изменения UPPS/CCPS, будет уже корректным. Без необходимости осуществлять пропуски данных или дополнительных программных манипуляций.

Если бит SELEVNT в регистре QCAPCTL сброшен, то по деленному квадратурному событию значение таймера QCTMR загружается в регистр периода QCPRD, после чего таймер сбрасывается, и устанавливается флаг UPEVNT в регистре QEPSTS, означающий обновление регистра QCPRD. Флаг сбрасывается программно записью единицы.

При установленном бите SELEVNT обновление регистра периода происходит по сигналу от выхода компаратора PCSOUT.

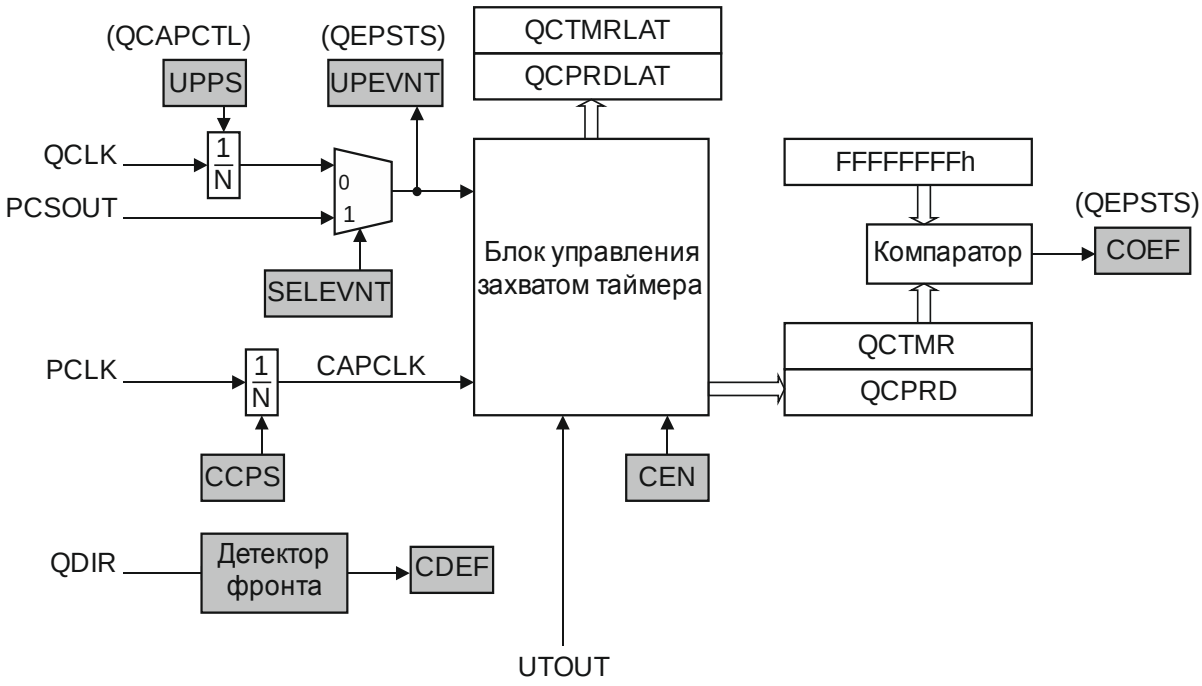


Рисунок 13.9 – Функциональная схема модуля захвата времени

Значение таймера можно использовать при измерениях скорости, если:

- его значение не превысило FFFF_FFFFh;
- направление вращения за время измерения не изменилось.

Если между двумя событиями UPEVNT (т. е. во время измерения) таймер QCTMR переполнился, устанавливается флаг ошибки COEF в регистре QEPSTS. Если между двумя событиями положения вала изменилось направление вращения, устанавливается флаг ошибки CDEF.

Значения таймера QCTMR и регистра периода QCPRD могут быть сконфигурированы для захвата в регистры QCTMRLAT и QCPRDLAT по событиям:

- прочитан регистр QPOSCNT;
- сработал сторожевой таймер.

Если бит QCLM сброшен, то при каждом чтении регистра счетчика позиции QPOSCNT регистры QCTMR и QCPRD загружаются в регистры QCTMRLAT и QCPRDLAT соответственно.

Если бит QCLM установлен, то при каждом срабатывании таймера временных отсчетов счетчик позиции, регистр таймера и регистр периода захватываются в регистры QPOSLAT, QCTMRLAT и QCPRDLAT соответственно.

Измерения на малых скоростях вращения (низкая частота квадратурного сигнала) производятся следующим образом – таймер QCTMR, тактирующийся от системного тактового сигнала с делителем CCPS, по событию UPEVNT, сохраняет свое значение времени в регистре QCPRD, одновременно сбрасывается и выставляет флаг UPEVNT в регистре QEPSTS, чтобы сообщить программе об окончании измерения. Событие UPEVNT возникает каждые несколько тактов QCLK, в соответствии с запрограммированным коэффициентом деления UPPS в регистре QCAPCTL. Таким образом, зная количество квадратурных событий за измеренный отрезок времени, а также такой параметр, как количество квадратурных событий за полный оборот вала, можно вычислить скорость вращения.

Измерения на высоких скоростях, см. рисунок 13.10, могут производиться иначе. Таймер временных отсчетов формирует общую длительность измерения, счетчик позиции подсчитывает количество импульсов QCLK. Зная количество импульсов QCLK за один полный оборот, можно вычислить скорость вращения вала.

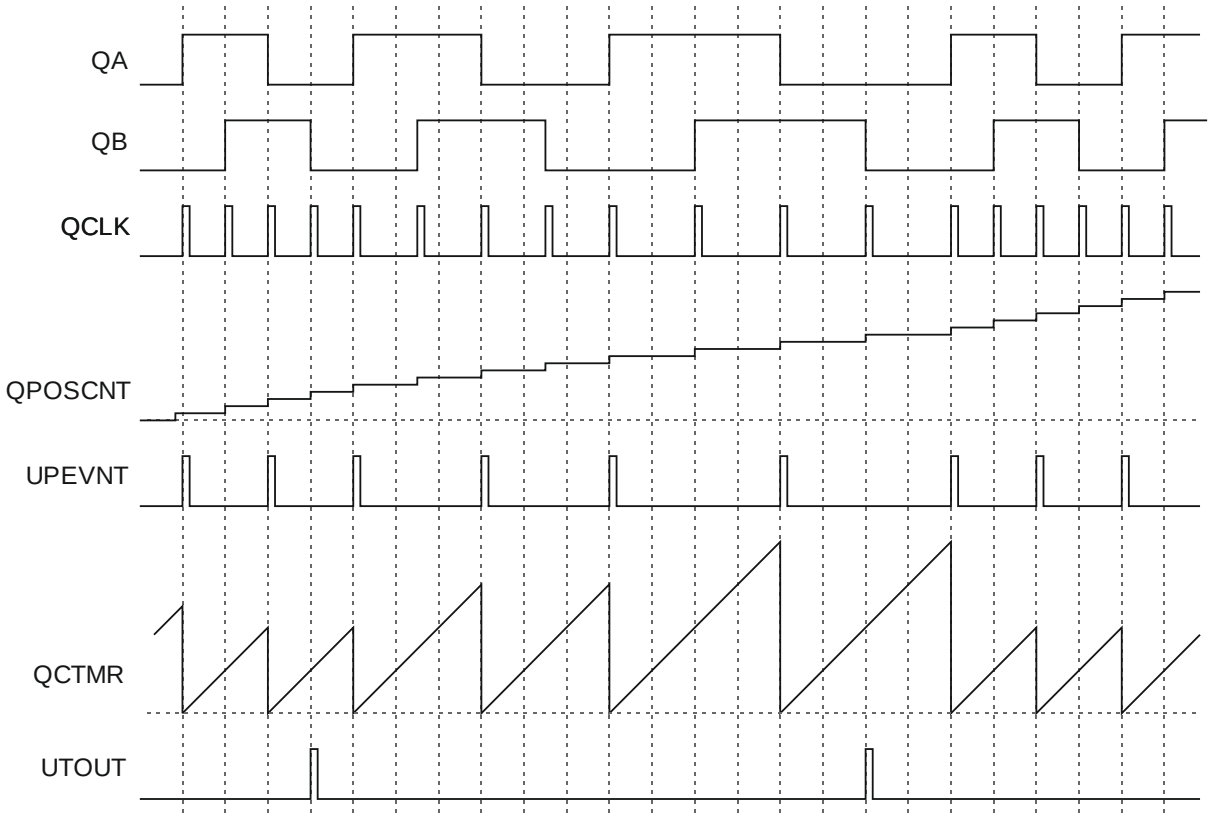


Рисунок 13.10 – Работа на высоких скоростях

Также существует и смешанный способ измерения скорости – по заданному значению счетчика позиции, с помощью компаратора счетчика позиции, можно сформировать событие UPEVNT (необходимо установить бит SELEVNT в регистре QCAPCTL), которое, так же как и при измерениях на малых скоростях, позволит получить значение таймера QCTMR. Для использования этого способа измерения скорости необходимо разрешить прерывание по событию PCSOUT компаратора и устанавливать в этом прерывании каждый раз порог сравнения компаратора QPOSCMP на заданное количество меток вперед по сравнению с текущей позицией счетчика QPOSCNT (в зависимости от направления вращения). Тогда, устанавливая QPOSCMP дальше от QPOSCNT с увеличением скорости вращения, можно поддерживать оптимальное захватываемое время, обеспечивающее максимальную точность измерения времени для всех диапазонов вращения. Этот способ измерения наиболее сложен, но и наиболее универсален.

Дополнительно, модуль захвата времени способен генерировать запросы DMA. Чтобы разрешить генерацию необходимо установить бит DMAEN в регистре DMAREQ. Затем, каждый раз, когда флаг UPEVNT (регистра EPSTS) переходит из 0 в 1, будет генерироваться запрос. Если UPEVNT будет оставаться несброшенным, то запросы генерироваться не будут. При этом, когда происходит чтение QCPRD при DMAEN = 1, флаг UPEVNT сбрасывается автоматически.

13.6 Сторожевой таймер блока QEP

Блок квадратурного декодера QEP содержит 32-битный сторожевой таймер, который тактируется системным тактовым сигналом, деленным на 64, и сбрасывается любым квадратурным событием (перепад на выводе QEP_A/QEP_B). Если ни одного квадратурного события не было зафиксировано до события QWDTMR = QWDPRD, сторожевой таймер формирует флаг прерывания WTO в регистре QFLG. Регистр QWDPRD содержит значение срабатывания сторожевого таймера. Функциональная схема сторожевого таймера представлена на рисунке 13.11.

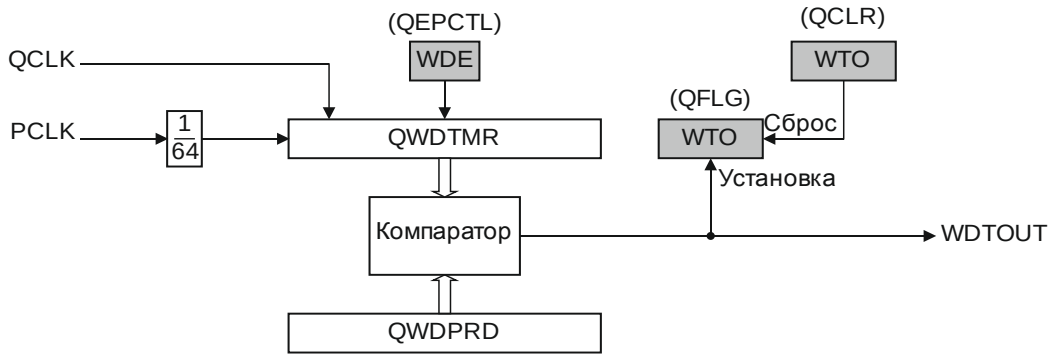


Рисунок 13.11 – Функциональная схема сторожевого таймера

13.7 Прерывания

Блок квадратурного декодера QEP содержит 11 источников прерываний, см. рисунок 13.12. Система прерываний состоит из регистра маски прерываний QEINT, регистра флагов прерываний QFLG, а также схемы формирования внешнего прерывания QEP_INT по наличию активных флагов. Прерывание INT также может быть маскировано в контроллере прерывания NVIC. Сброс флагов прерываний осуществляется через регистр QCLR. Сброс флага активности прерывания INT осуществляется записью в регистр INTCLR. Также, прерывание можно сформировать программной записью в регистр QFRC, но для этого необходимо предварительно включить счетчик позиции, установив бит QPEN в регистре QEPCTL.

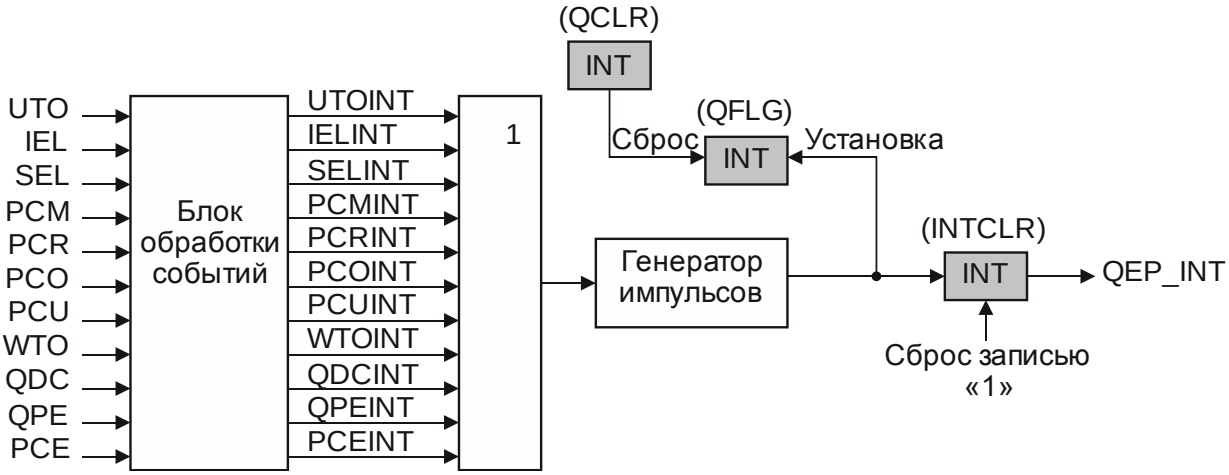


Рисунок 13.12 – Схема системы прерываний

На рисунке 13.13 показана схема формирования прерывания внутри блока обработки событий для события UTO (срабатывание таймера временных отсчетов). Схемы формирования прерываний для остальных событий идентичны.

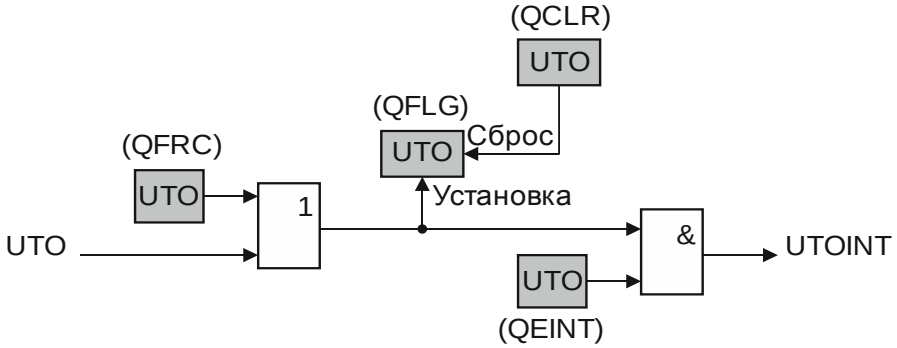


Рисунок 13.13 – Схема формирования прерывания UTOINT

14 Блоки ШИМ

Архитектура блоков ШИМ разработана по принципу минимальной нагрузки на процессор, что достигается автоматизацией формирования выходных импульсов с настраиваемыми пользователем параметрами. Так, после минимальных настроек, эти блоки способны работать самостоятельно, формируя выходные сигналы на выводах PWMx_A и PWMx_B (где x = 0, 1, 2) микроконтроллера.

Микроконтроллер содержит три блока ШИМ, объединенных схемой синхронизации.

Каждый блок ШИМ поддерживает следующую функциональность:

- 16-разрядный таймер;
- выходы PWMx_A и PWMx_B могут работать в режиме фронтовой и центрированной модуляции как полностью независимо, так и комплементарно с разделением генератором «мертвого» времени;
- выходы PWMx_A и PWMx_B могут управляться в зависимости от событий цифровых компараторов блока АЦП, а также от событий блока аналоговых компараторов, обеспечивая автоматический релейный режим поддержания заданной величины;
- программное управление выходами ШИМ;
- программное задание фазы счетчиков таймера для координации работы нескольких блоков ШИМ;
- аппаратный контроль фазы при координации работы нескольких блоков ШИМ;
- предотвращение наложения фронтов за счет генератора «мертвого» времени с независимой схемой задержки переднего и заднего фронтов выходного сигнала;
- сигнал аварии может переводить выходы PWMx_A и PWMx_B в высокое, низкое или Z-состояние;
- однократная и циклическая обработка сигналов аварии;
- все события могут инициировать прерывания;
- при обработке прерываний программируемый предделитель событий позволяет снизить нагрузку на процессор;
- сигнал ШИМ может модулироваться высокочастотным сигналом при использовании драйверов ключей с импульсным трансформатором.

Для начала работы с блоками ШИМ необходимо снять с них сброс и разрешить тактирование – необходимо установить биты PWMEN регистов PCLKCFG и PRSTCFG.

Описание сигналов и выводов блока ШИМ:

- PWMx_A и PWMx_B (где x = 0, 1, 2) – выходы ШИМ;
- PWM_TZ – вход, с которого принимается сигнал аварии (общий для всех блоков ШИМ, и каждый блок может использовать или не использовать этот сигнал);
- PWM_SYNCI – вход, служащий для приема синхросигнала;
- PWM_SYNCO – выход, использующийся для синхронизации блока ШИМ;
- PWM_TZINT – прерывание по сигналу аварии;
- PWM_HDINT – прерывание от порогового выключателя.

Функциональная схема блока ШИМ показана на рисунке 14.1.

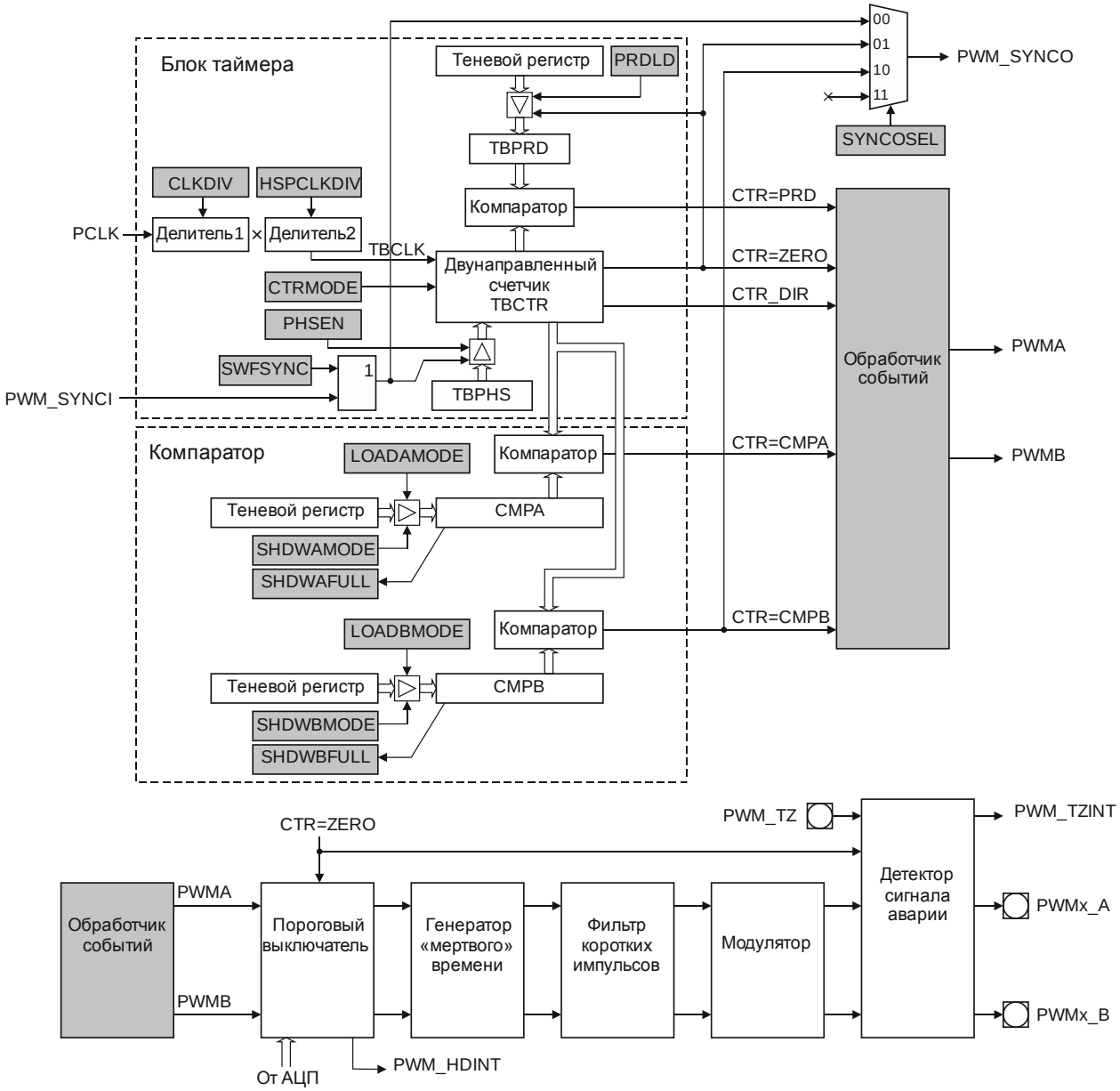


Рисунок 14.1 – Функциональная схема бока ШИМ

14.1 Таймер

Таймер представляет собой двунаправленный счетчик TBCTR, тактируемый сигналом TBCLK, который формируется на основе синхросигнала PCLK. Частота сигнала TBCLK задается произведением коэффициентов двух делителей. Коэффициенты задаются полями CLKDIV и HPCLKDIV регистра TBCTL. По умолчанию, работа делителей остановлена - TBCLK не генерируется. Чтобы таймер начал счет по TBCLK, необходимо записью единиц в поле PRESCRST регистра PWMSYNC блока SIU разрешить работу делителей.

Для работы других блоков ШИМ счетчик позволяет формировать события такие, как совпадение по периоду CTR = PRD (TBCTR = TBPRD), совпадение с нулем CTR = Zero (TBCTR = 0000h), совпадение с регистрами CTR = CMPA и CTR = CMPB (TBCTR = CMPA и TBCTR = CMPB соответственно). Событие TBCTR = FFFFh влияет только на флаг CTRMAX регистра TBSTS.

Всеми настройками работы счетчика таймера управляет регистр TBCTL.

Состояние счетчика отражают флаги регистра TBSTS.

На выходе первого блока ШИМ формируется сигнал SYNCO, который является, синхросигналом для остальных блоков ШИМ, см. рисунок 14.3а. Сигнал SYNCO имеет три источника – программно сгенерированный синхроимпульс, посредством записи единицы в бит SWFSYNC, события CTR = Zero и CTR = CMPB. Выбор источника осуществляется посредством поля SYNCOSEL.

В блоке таймера находятся регистры начальной фазы счета TBPHS и периода (максимального значения счетчика) TBPRD. Регистр периода имеет теневой регистр для синхронной загрузки значения, до которого счетчик осуществляет счет. Управление загрузкой осуществляется битом PRDLD.

Счетчик может работать в трех режимах счета, см. рисунок 14.2:

- вверх (от 0000h до значения TBPRD, затем сброс в 0000h и т. д.);
- вниз (от значения TBPRD до 0000h, затем загрузка значения TBPRD и т. д.);
- вверх-вниз (от 0000h до значения TBPRD, затем от значения TBPRD до 0000h и т. д.).

Параметры счета задаются полем CTRMODE.

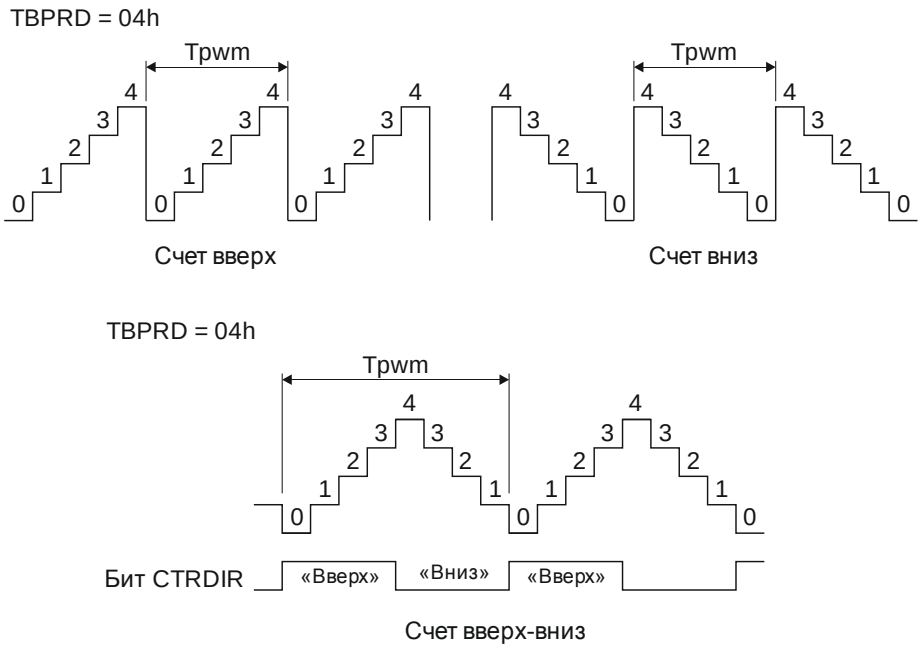


Рисунок 14.2 – Режимы работы счетчика при значении периода 0004h (Tpwm); для режима счета «вверх-вниз» дополнительно указано поведение флага CTRDIR

Теневая загрузка

Регистры TBPRD, CMPA, CMPAHR, CMPB имеют соответствующие теневые регистры и управляющие биты, регулирующие режим и событие загрузки. Помимо индивидуальной настройки теневого режима есть бит SHDWGLOB регистра TBCTL. По умолчанию он установлен, и теневая загрузка работает согласно настройкам. Но возможны ситуации, когда необходимо обеспечить одновременную загрузку всех теневых регистров. Тогда перед загрузкой необходимо сбросить SHDWGLOB, в результате значения будут попадать в теневые регистры, но перезапись в активные будет блокирована. Когда все необходимые регистры будут записаны, следует установить этот бит, и тогда все активные регистры будут перезаписаны в соответствии с заданными настройками по соответствующим событиям. Данный бит не оказывает влияния, если для регистров выбрана прямая загрузка, без участия теневых.

Синхронизация таймеров блоков ШИМ

Реализована схема синхронизациии блоков ШИМ, см. рисунок 14.3а.

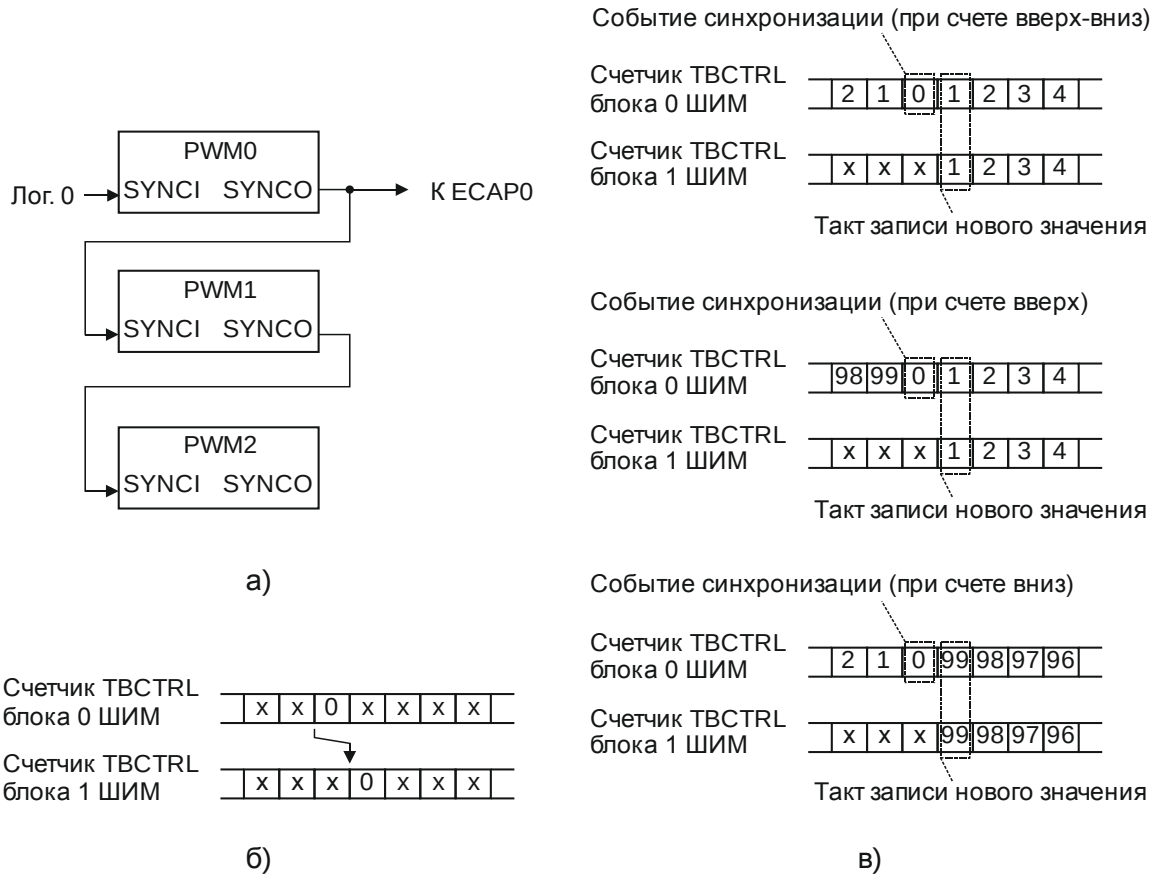


Рисунок 14.3 – Схема синхронизации модулей ШИМ

Система синхронизации таймеров включает в себя таймеры всех блоков ШИМ. Каждый блок ШИМ имеет вход синхронизации PWMx_SYNCI и выход синхронизации PWMx_SYNCO (где x = 0, 1, 2).

Если бит PHSEN установлен, то в счетчик таймера будет автоматически загружаться значение регистра TBPHS, при выполнении каждого из условий:

- изменение входного сигнала PWMx_SYNCI (в этом случае загрузка значения TBPHS в регистр TBCTR происходит на следующий такт TBCLK после поступления импульса на вход PWMx_SYNCI (где x = 0, 1, 2) с задержкой в два системных такта, если TBCLK = PCLK, или один такт, если TBCLK ≠ PCLK);
- запись единицы в бит SWFSYNC (программная синхронизация), которая генерирует импульс синхронизации, аналогичный импульсу с входа PWMx_SYNCI.

В режиме счета вверх-вниз необходимо запрограммировать бит PHSDIR, чтобы задать направление счета таймера после синхронизации.

Если бит PHSEN сброшен, блок ШИМ не будет реагировать на входной сигнал синхронизации, а только передавать напрямую этот сигнал на выход PWM_SYNCO, чтобы тактировать другие блоки ШИМ. Следующая особенность схемы: генерация и распространение сигнала синхронизации от блока ШИМ, – занимает один такт TBCLK.

К примеру, если по событию синхронизации блока 0 в счетчик блока 1 должен быть записан ноль, то этот ноль запишется только на следующий такт после события, см. рисунок 14.3б. Таким образом, при синхронизации от другого блока ШИМ нужно всегда учитывать этот такт и записывать значение фазы, следующее по порядку, в соответствии с режимом счета, см. рисунок 14.3в.

14.2 Компаратор

Компаратор – это блок, сравнивающий значение счетчика таймера с заданными значениями порогов срабатывания. Значения хранятся в регистрах CMPA и CMPB. Значения, записываемые по адресам регистров CMPA и CMPB, предварительно размещаются в теневых регистрах. Это нужно для синхронной загрузки новых значений. Управление загрузкой регистров CMPA и CMPB осуществляется битами SHDWAMODE и SHDWBMODE, а также полями LOADAMODE и LOADBMODE регистра CMPCTL.

Блок компаратора формирует на выходах два события CTR = CMPA и CTR = CMPB, возникающие в случае совпадения значения счетчика с регистром CMPA и/или регистром CMPB соответственно.

Для каждого компаратора событие может возникать:

- один раз за период, если счетчик считает вверх или вниз;
- один раз за период, если счетчик считает вверх-вниз, но при этом значение в регистре CMPA/CMPB равно 0000h или значению TBPRD;
- два раза за период, если счетчик считает вверх-вниз и при этом значение в регистре CMPA/CMPB лежит в диапазоне 0001h – (TBPRD – 1).

На рисунках 14.4 – 14.7 приведены примеры формирования сигналов событий.

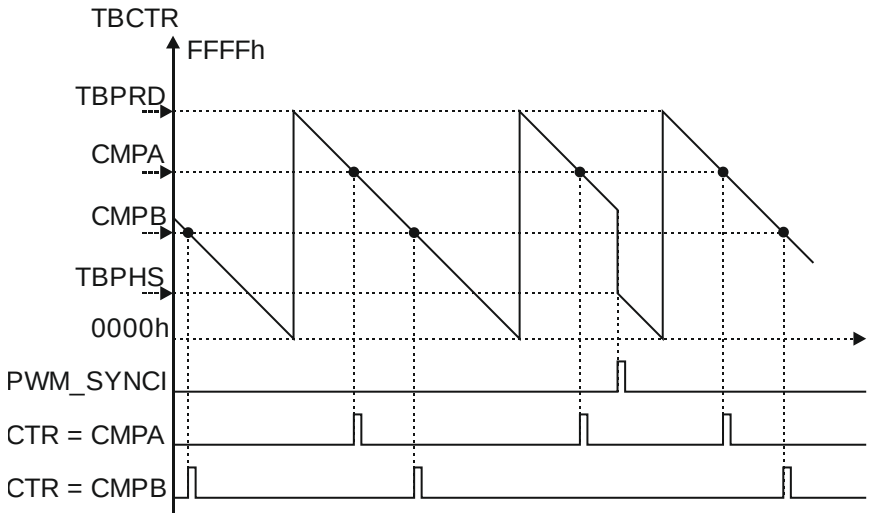


Рисунок 14.4 – Диаграммы работы при счете вниз

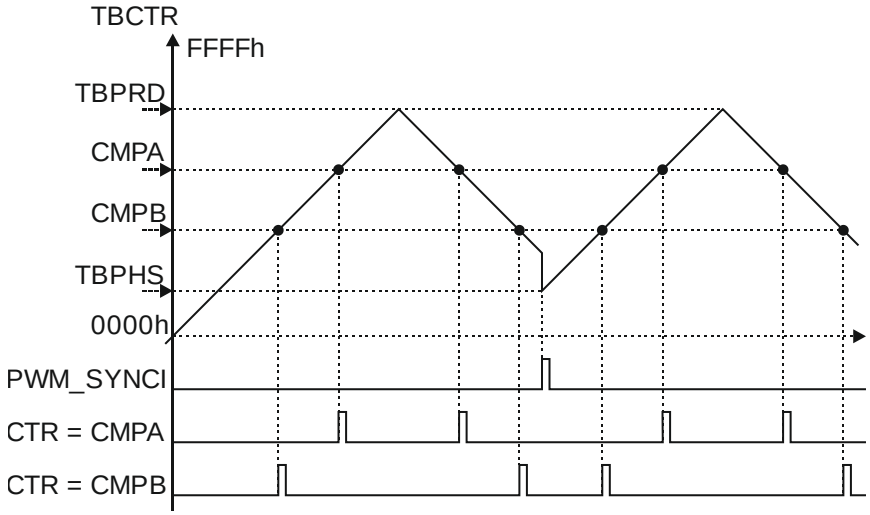


Рисунок 14.5 – Диаграммы работы при счете вверх-вниз. Синхронизация при счете вверх

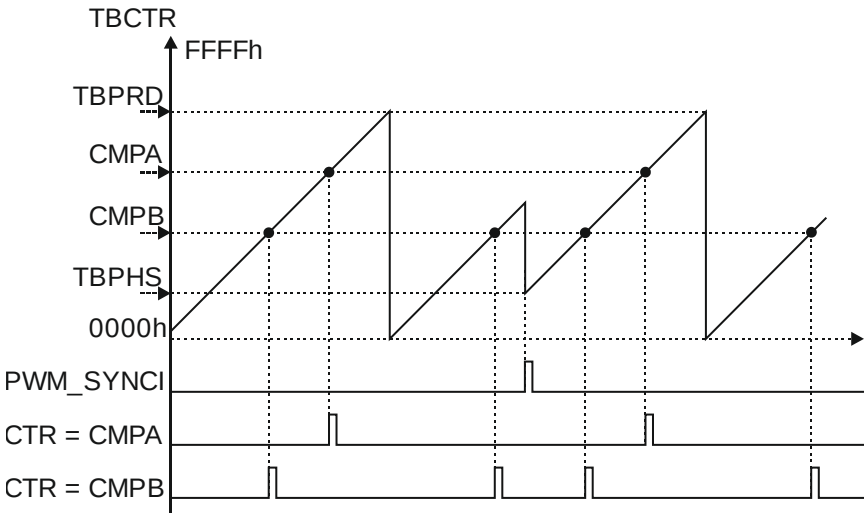


Рисунок 14.6 – Диаграммы работы при счете вверх

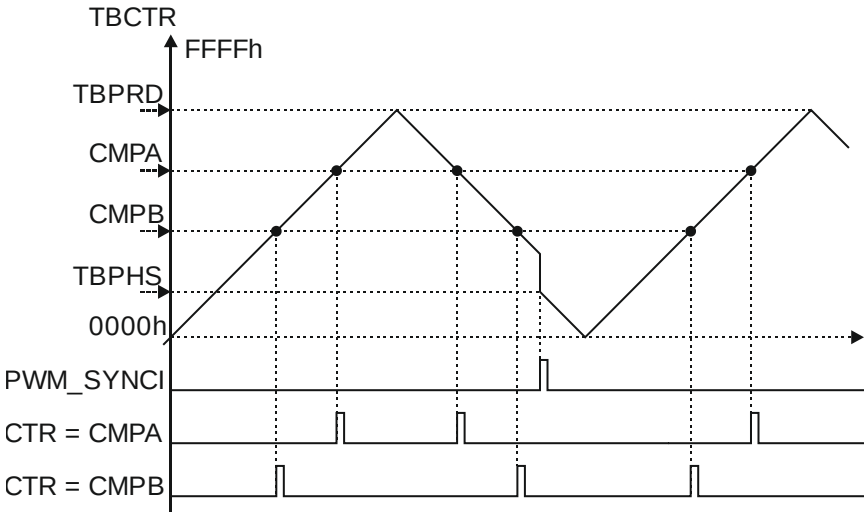


Рисунок 14.7 – Диаграммы работы при счете вверх-вниз.
Синхронизация при счете вниз

14.3 Обработчик событий

Обработчик событий – блок, управляющий поведением сигналов на линиях PWMA и PWMB, см. рисунок 14.1, в зависимости от возникающих событий на входе блока и направления счета счетчика таймера. На поведение выходных сигналов влияют импульсы входных сигналов при возникновении событий: CTR = PRD, CTR = Zero, CTR = CMPA, CTR = CMPB.

Основные действия с сигналами PWMA и PWMB:

- переключение в единицу или ноль;
- инверсия (переключение в противоположное состояние);
- сохранение без изменений.

Поведение сигналов задается независимо друг от друга. Кроме этого, обработчик событий позволяет программно задавать состояние сигналов PWMA и PWMB и величину «мертвого» времени ШИМ. Управление работой блока производится посредством регистров AQCTLA, AQCTLB, AQSFRC, AQCSFRC.

Существует вероятность того, что несколько событий могут произойти одновременно. Для таких ситуаций обработчик событий использует систему приоритетов событий при счете вверх, при счете вниз, при счете вверх-вниз, см. таблицы 14.1 – 14.3.

Таблица 14.1 – Распределение приоритетов событий при счете вверх

Событие	Приоритет
Программное	1 (самый высокий)
CTR = TBPRD	2
CTR = CMPB (счет вверх) при счете вверх	3
CTR = CMPA (счет вверх) при счете вверх	4 (самый низкий)

Таблица 14.2 – Распределение приоритетов событий при счете вниз

Событие	Приоритет
Программное	1 (самый высокий)
CTR = Zero	2
CTR = CMPB (счет вниз) при счете вниз	3
CTR = CMPA (счет вниз) при счете вниз	4 (самый низкий)

Таблица 14.3 – Распределение приоритетов событий при счете вверх-вниз

Событие	Приоритет
Программное	1 (самый высокий)
CTR = CMPB (счет вверх) при счете вверх или CTR = CMPB (счет вниз) при счете вниз	2
CTR = CMPA (счет вверх) при счете вверх или CTR = CMPA (счет вниз) при счете вниз	3
CTR = Zero или CTL = PRD	4
CTR = CMPB (счет вверх) при счете вниз или CTR = CMPB (счет вниз) при счете вверх	5
CTR = CMPA (счет вверх) при счете вниз или CTR = CMPA (счет вниз) при счете вверх	6 (самый низкий)

В режиме счета вверх:

- если компаратор запрограммирован так, что CMPA/CMPB ≤ TBPRD (счет вверх), то событие произойдет при CTR = CMPA/CMPB;
- если компаратор запрограммирован так, что CMPA/CMPB > TBPRD (счет вверх), то событие не произойдет;
- если компаратор запрограммирован на срабатывание при счете вниз, то событие не произойдет.

В режиме счета вниз:

- если компаратор запрограммирован так, что CMPA/CMPB ≤ TBPRD (счет вниз), то событие произойдет при CTR = CMPA/CMPB;
- если компаратор запрограммирован так, что CMPA/CMPB ≥ TBPRD (счет вниз), то событие произойдет при CTR = TBPRD;
- если компаратор запрограммирован на срабатывание при счете вверх, то событие не произойдет.

В режиме счета вверх-вниз, см. рисунок 14.8:

- если счетчик считает вверх, а компаратор запрограммирован так, что CMPA/CMPB < TBPRD (счет вверх), то событие произойдет при CTR = CMPA/CMPB;
- если CMPA/CMPB ≥ TBPRD (счет вверх), то событие произойдет при CTR = TBPRD;
- если счетчик считает вниз, а компаратор запрограммирован так, что CMPA/CMPB < TBPRD (счет вниз), то событие произойдет при CTR = CMPA/CMPB;
- если CMPA/CMPB ≥ TBPRD (счет вверх), то событие произойдет при CTR = TBPRD.

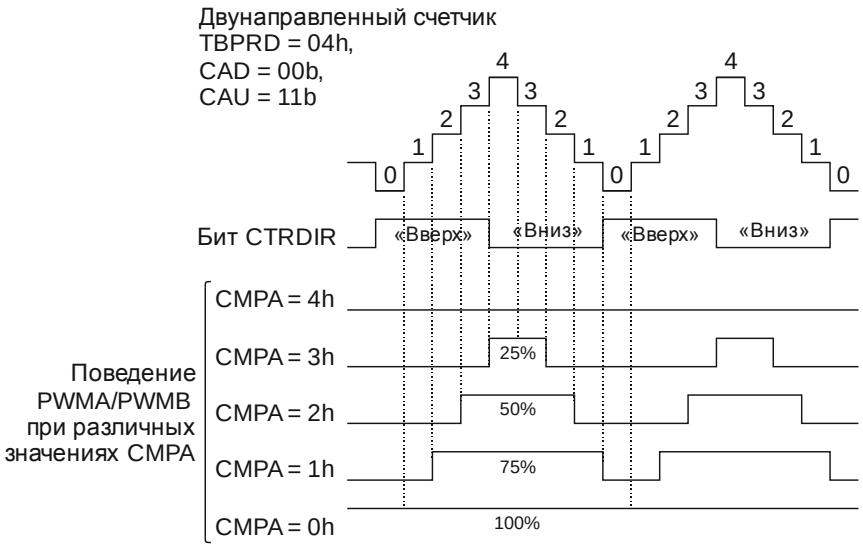


Рисунок 14.8 – Работа таймера при счете вверх-вниз с симметричным выходом (центрированная модуляция)

На рисунках 14.9 – 14.11 показано поведение линий PWMA и PWMB при различных видах модуляции, принятые обозначения и пояснения приведены в таблице 14.4.

Таблица 14.4 – Пояснения к обозначениям на рисунках 14.9 – 14.11

<table>
<tr><th colspan="3">Обозначение</th><th colspan="2">Пояснение</th></tr>
<tr><td>P
×</td><td>CA
×</td><td>CB
×</td><td colspan="2">События CTR = PRD, CTR = CMPA, CTR = CTRB соответственно. Символ «×» указывает на то, что при возникновении этого события сигнал на линии PWMA/PWMB остается без изменений. Пунктирными линиями отмечены моменты возникновения события.
Так, например, см. рисунок 14.9, при возникновении события CTR = CTRB, сигнал на линии PWMA остается без изменения, а сигнал на линии PWMB переключается в ноль</td></tr>
<tr><td colspan="2">Z
↑</td><td>Z
↓</td><td>Событие CTR = Zero</td><td rowspan="3">Символы «↑»/«↓» указывают на то, что при возникновении этого события сигнал на линии PWMA/PWMB переключается в единицу/ноль</td></tr>
<tr><td colspan="2">CA
↑</td><td>CA
↓</td><td>Событие CTR = CMPA</td></tr>
<tr><td colspan="2">CB
↑</td><td>CB
↓</td><td>Событие CTR = CMPB</td></tr>
</table>

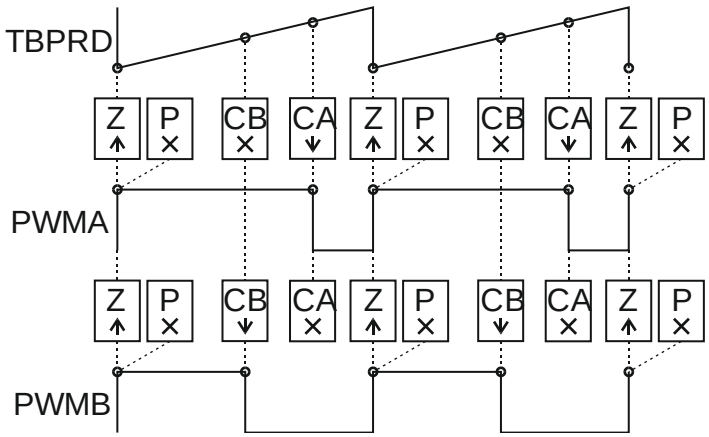


Рисунок 14.9 – Независимый режим работы выходов (фронтовая модуляция)

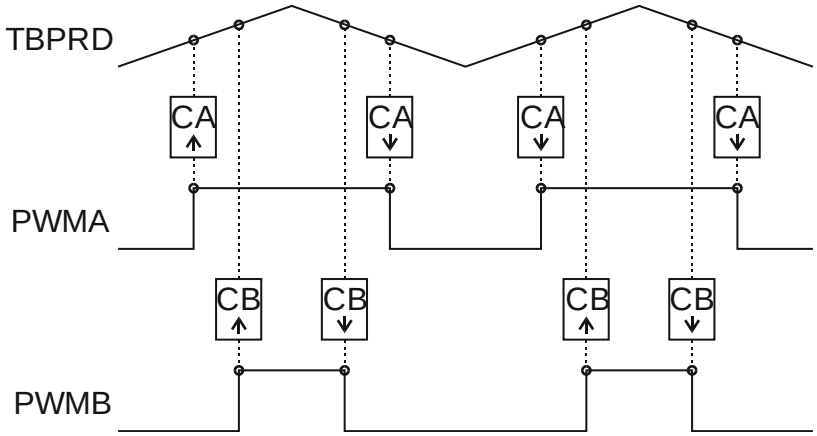


Рисунок 14.10 – Симметричный режим работы при счете вверх-вниз (центрированная модуляция)

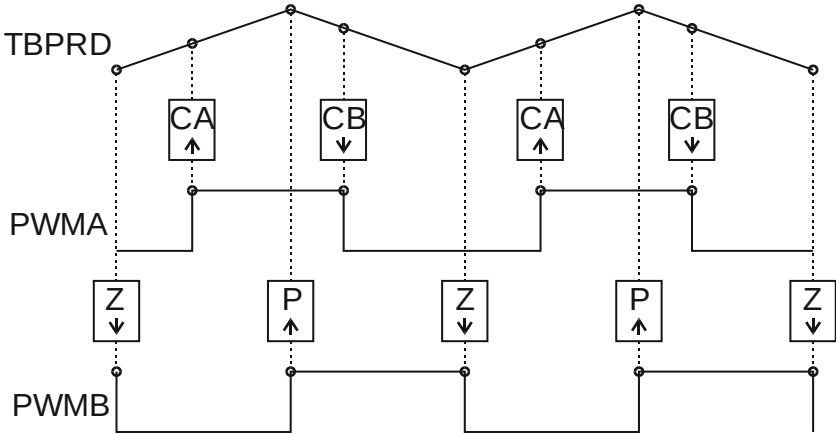


Рисунок 14.11 – Ассиметричный режим работы при счете вверх-вниз

14.4 Пороговый выключатель

Пороговый выключатель контролирует выходные сигналы PWMA и PWMB обработчика событий и позволяет удерживать их в определенном заданном пользователем состоянии в случае прихода сигнала триггера от цифровых компараторов блока АЦП. Этот блок удобен для организации релейного поддержания заданного уровня какой-либо физической величины, например для организации контура тока. В этом случае цифровой

компаратор, к каналу АЦП которого подключен сигнал датчика тока контура, формирует сигнал о превышении током задания, а соответствующий пороговый выключатель реагирует на это превышение и включает/отключает соответствующий силовой транзистор посредством влияния на выход ШИМ.

Функциональные возможности:

- входные события от компараторов блока АЦП могут использоваться всеми блоками ШИМ;
- при регистрации события от компаратора блока АЦП выходные сигналы обработчика событий могут быть переведены в состояние логической единицы, нуля или оставлены без изменений;
- поддерживаются однократное и циклическое срабатывания для удержания выхода;
- входное событие от компаратора блока АЦП может анализироваться в однократном и циклическом режимах;
- событие срабатывания компаратора блока АЦП может быть сгенерировано программно;
- пороговый выключатель может быть отключен, если он не требуется;
- может генерироваться прерывание по событиям порогового выключателя.

Управление пороговым выключателем осуществляется посредством регистров HDSEL, HDCTL и HDFRC.

Функционирование

Когда выходные сигналы компараторов блока АЦП переходят в состояние высокого уровня, формируется событие. Каждый пороговый выключатель блока ШИМ может использовать, а может не использовать эти события в своей работе; выбор, по сигналу какого компаратора блока АЦП формировать событие удержания, задается с помощью регистра HDSEL. Длительность импульса на входном сигнале от компаратора блока АЦП не должна быть меньше периода системного синхросигнала. Каждый входной сигнал компаратора блока АЦП должен быть настроен на однократное или циклическое формирование события, выбор режима задается битами CBC и OST, а источник события – полем ADCDC.

При получении события от компаратора блока АЦП в режиме циклической обработки немедленно формируется реакция на основе содержимого регистра HDCTL, в результате чего меняется состояние сигналов на выходе порогового выключателя взамен полученных от обработчика событий PWMA и/или PWMB на заданное пользователем в регистре HDCTL. Дополнительно устанавливается флаг CBC в регистре HDFLG, и генерируется прерывание PWM_HDINT. Удержание выходных сигналов PWMA и PWMB заканчивается по событию TBCTR = 0000h, при условии, что событие компаратора блока АЦП уже не активно. Таким образом, в режиме циклической обработки состояние удержания сбрасывается в каждом периоде ШИМ. При этом флаг CBC остается активным до его программного сброса. Если после сброса флага CBC вновь будет получено событие компаратора блока АЦП, то флаг установится вновь.

При получении события компаратора блока АЦП в режиме однократной обработки, также немедленно формируется реакция на основе содержимого регистра HDCTL, которая меняет состояние выходных сигналов PWMA и/или PWMB. В дополнение, устанавливается флаг OST, и генерируется прерывание PWM_HDINT. Удержание выходных сигналов будет производиться до программного сброса записью единицы в бит OST регистра HDCLR.

Способ удержания выходных сигналов при получении события компаратора блока АЦП программируется индивидуально для выходных сигналов PWMA и PWMB в регистр HDCTL.

14.5 Генератор задержки ШИМ

Блок имеет на входе сигналы PWMA и PWMB с выходов обработчика событий, а на выходах повторяет эти сигналы, но со вставкой задержки («мертвое» время) в момент переключения сигналов (если это необходимо).

Задержку можно учесть при программировании обработчика событий, но чтобы с высокой вероятностью избежать ошибок, желательно использовать генератор задержки ШИМ.

Основные функции генератора задержки ШИМ:

- генерация пары выходных сигналов PWMA и PWMB с выдержкой интервалов (задержек) времени относительно входных сигналов PWMA, PWMB;
- программирование задержки для активного высокого и активного низкого уровня сигналов каналов PWMA и PWMB;
- добавление программируемой задержки передних фронтов сигналов;
- добавление программируемой задержки для задних фронтов сигналов;
- возможность передачи сигналов с входов на выходы без изменений.

Генератор задержки ШИМ программируется посредством регистров DBCTL, DBRED и DBFED. Структурная схема генератора «мертвого» времени ШИМ представлена на рисунке 14.12.

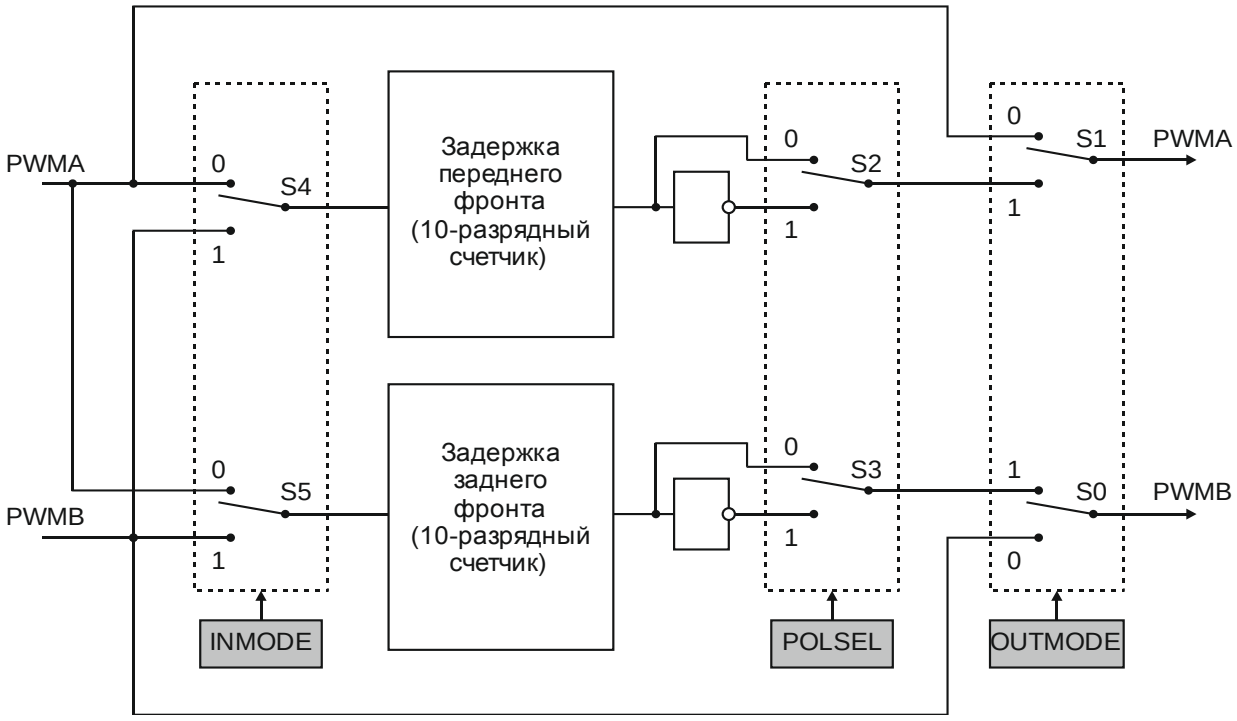


Рисунок 14.12 – Структурная схема генератора «мертвого» времени ШИМ

Функционирование

Генератор задержки ШИМ может работать с четырьмя источниками (фронты сигналов PWMA и PWMB). Выбор источника задается полем MODE регистра DBCTL.

Поле POLSEL позволяет задать инверсию (переключение значения на противоположное) сигнала после внесения задержки, см. рисунок 14.13.

Величины задержек по переднему и заднему фронту программируются раздельно посредством регистров DBRED и DBFED соответственно.

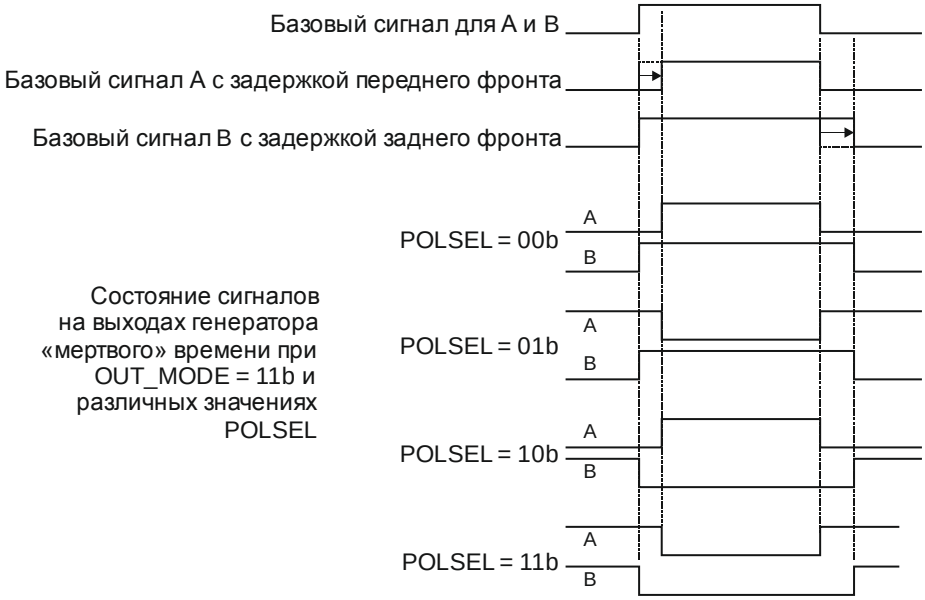


Рисунок 14.13 – Временные диаграммы работы генератора «мертвого» времени в типовой конфигурации

14.6 Фильтр коротких импульсов

Фильтр коротких импульсов предназначен для блокирования прохождения на выход импульсов с длительностью меньше заданной. Этот блок может применяться, если драйвер силового ключа инвертора не имеет такой функции, а для обеспечения правильного режима работы транзистора необходимо запретить открытие/закрытие транзистора на очень короткое время.

Основные функции фильтра:

- программируемая ширина минимального пропускаемого импульса;
- фильтр может быть отключен, если он не требуется.

Ширина минимального импульса, допускаемого к прохождению на выход, задается в регистре FWDTH в тактах PCLK и может принимать значение от 00h (фильтр выключен) до FFh. Импульсы длительностью меньше заданной пропускаться не будут.

14.7 Модулятор

Блок позволяет модулировать выходной сигнал ШИМ с помощью высокочастотных импульсов программируемой скважности. Модулирование требуется для управления силовыми ключами через импульсный трансформатор.

Основные функции модулятора:

- программируемая частота;
- программируемая ширина первого импульса;
- программируемый коэффициент заполнения второго и последующего импульсов;
- модулятор может быть отключен (бит CHPEN регистра PCCTL).

Модулятор программируется посредством регистра PCCTL. Структурная схема модулятора приведена на рисунке 14.14.

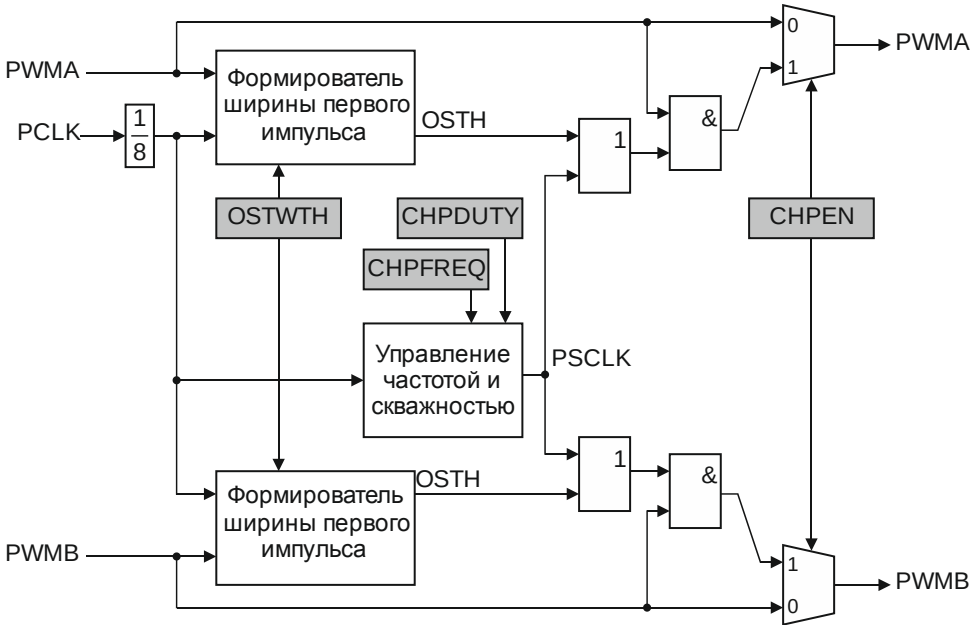


Рисунок 14.14 – Структурная схема модулятора

Функционирование

Частота модуляции формируется на основе системной частоты при помощи делителя, программируемого полем CHPFREQ.

На рисунке 14.15 приведен пример временных диаграмм работы модулятора.

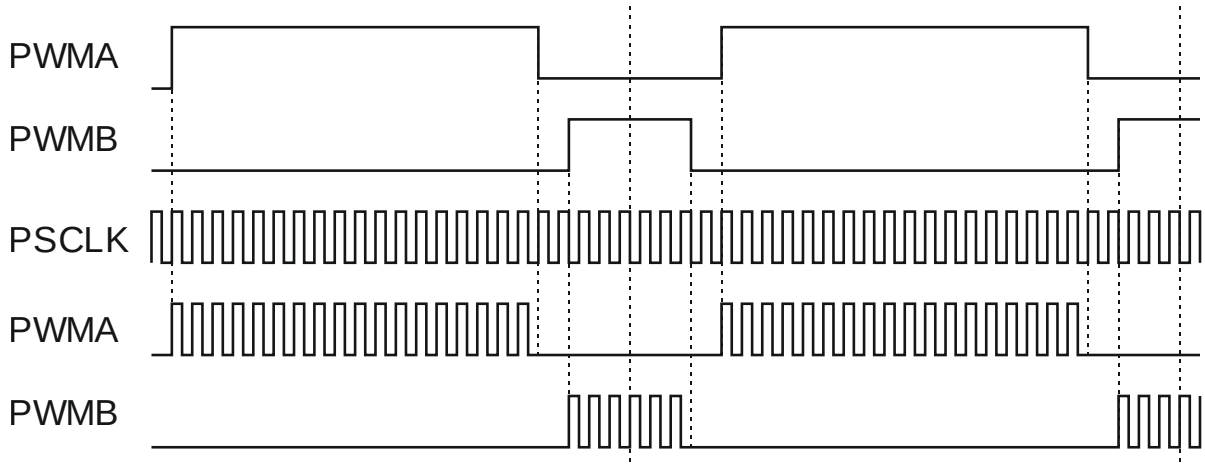


Рисунок 14.15 – Временные диаграммы работы модулятора

Ширина первого импульса может программироваться независимо с помощью поля OSTWTH, это требуется для открывания ключа. Значения поля OSTWTH лежат в диапазоне 0h – Fh.

Ширина L первого импульса определяется по формуле

$$L = T \times 8 \times (OSTWTH + 1), \qquad (14.1)$$

где T – период синхросигнала PCLK.

На рисунке 14.16 приведен пример формирования расширенного первого импульса.

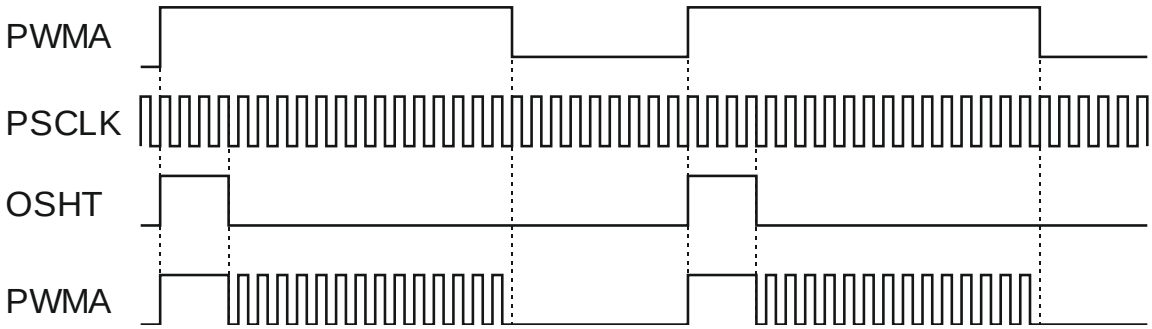


Рисунок 14.16 – Временные диаграммы работы модулятора с раширенным первым импульсом

Также существует возможность регулировать коэффициент заполнения импульсов посредством поля CHPDUTY. Значения поля CHPDUTY лежат в диапазоне 0h – 7h.

Коэффициент заполнения D (с шагом 12,5 %) последующих импульсов определяется по формуле

$$D = 12{,}5 \times (CHPDUTY + 1). \qquad (14.2)$$

На рисунке 14.17 приведена иллюстрация регулировки коэффициента заполнения.

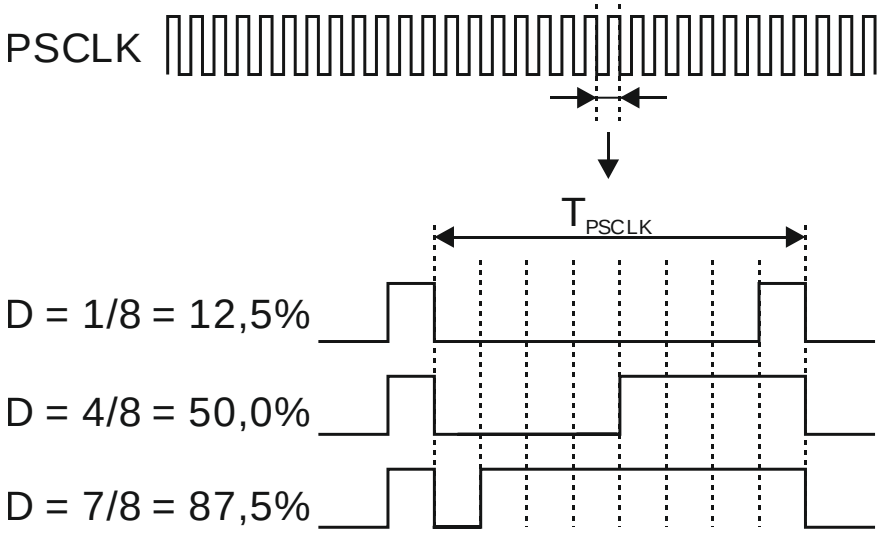


Рисунок 14.17 – Регулировка коэффициента заполнения D

14.8 Детектор сигнала аварии

Детектор сигнала аварии контролирует выходы PWMx_A и PWMx_B и может переводить их в определенное (запрограммированное) состояние в случае, если поступит сигнал аварии.

Основные функции детектора сигнала аварии:

- входной сигнал аварии с вывода микроконтроллера PWM_TZ может использоваться любым блоком ШИМ;
- в случае если поступит сигнал аварии, выводы ШИМ могут быть переведены в одно из состояний: логического нуля, логической единицы, высокоимпедансное или оставлены без изменения;
- поддерживается однократная блокировка выводов для ситуации короткого замыкания или перегрузки по току;
- поддерживается циклическая блокировка для режима ограничения тока;
- каждый входной источник сигнала аварии может быть обработан в однократном и циклическом режимах;
- поддерживается программная генерация сигнала аварии;
- детектор сигнала аварии может быть отключен, если он не требуется.

Детектор сигнала аварии программируется посредством регистров TZSEL, TZCTL, TZEINT, TZFLG, TZCLR и TZFRC.

Функционирование

Структурная схема детектора сигналов аварии показана на рисунке 14.18.

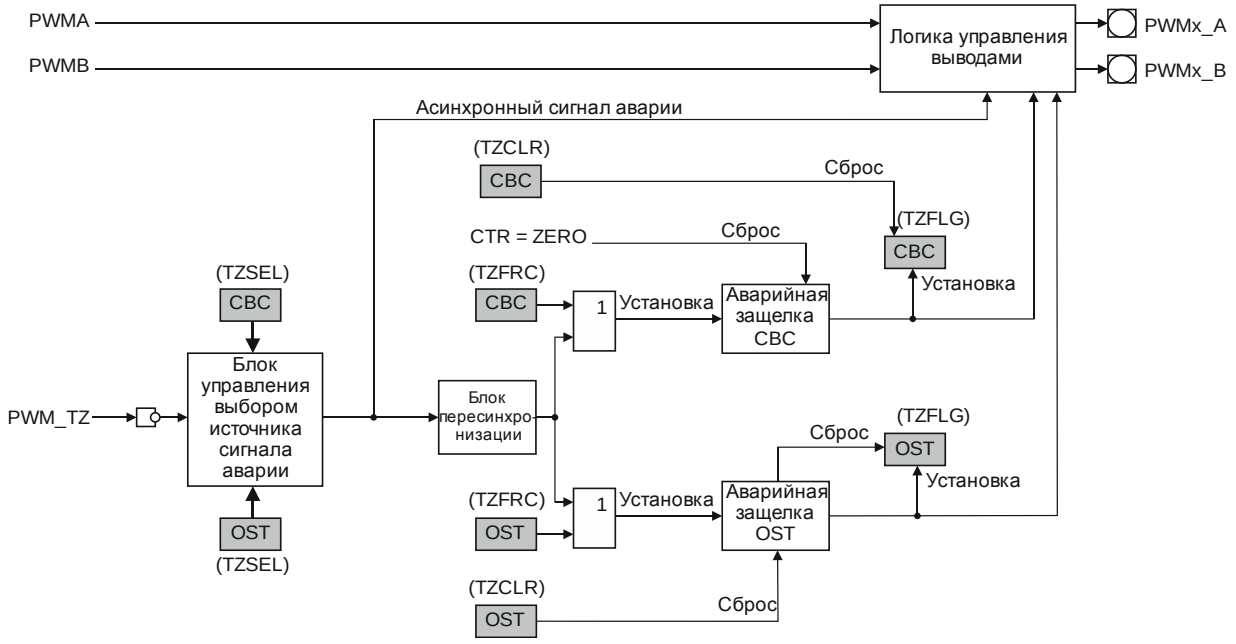


Рисунок 14.18 – Структурная схема детектора сигналов аварии

Переход входного сигнала аварии PWM_TZ из состояния логической единицы в состояние логического нуля формирует событие аварии. Каждый блок ШИМ может использовать или не использовать это событие в своей работе (программируется посредством регистра TZSEL). События могут формироваться синхронно (с цифровым фильтром помех) или асинхронно (программируется через регистры GPIO микроконтроллера). При синхронной обработке длительность импульса на входном сигнале сбоя должна быть не меньше периода синхросигнала TBCLK. Если же обработка производится в асинхронном режиме, то событие формируется и обрабатывается даже в том случае, если по какой-либо причине отключилось тактирование. Каждый входной сигнал аварии должен быть настроен на однократное или циклическое формирование события аварии (программируется посредством регистра TZSEL).

При получении события аварии в режиме циклической обработки немедленно выполняется действие, заданное регистром TZCTL, и устанавливается флаг CBC в регистре TZFLG, а также генерируется прерывание PWM_TZINT, если разрешено в регистре TZEINT и контроллером прерываний. Аварийное удержание выводов заканчивается по событию TBCTR = 0000h при условии, что событие аварии уже неактивно. Таким образом, в режиме циклической обработки событие аварии сбрасывается в каждом периоде ШИМ, хотя флаг аварии CBC остается установленным до принудительного программного сброса. Если после сброса регистра флага CBC вновь будет получено событие аварии, то флаг установится вновь. На рисунке 14.19 показана схема формирования прерывания.

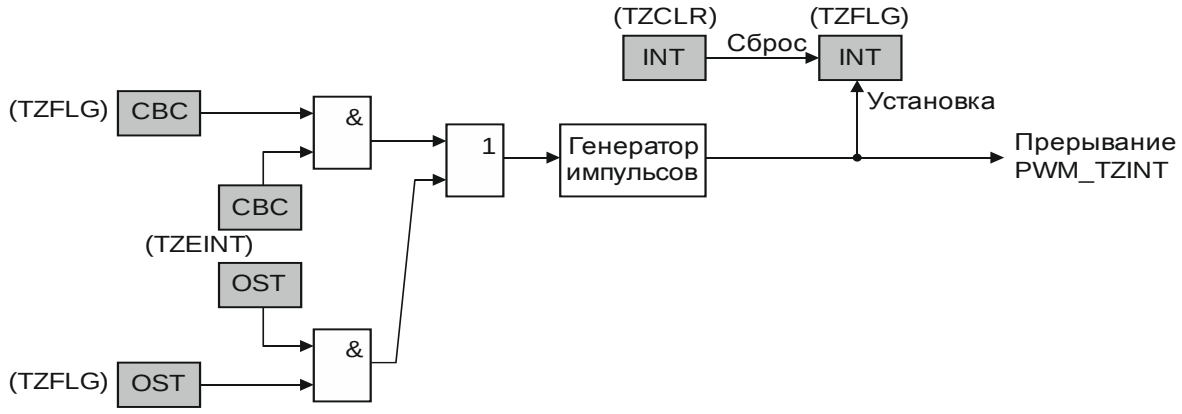


Рисунок 14.19 – Схема формирования прерывания

При получении события аварии в режиме однократной обработки немедленно выполняется действие, заданное регистром TZCTL, и устанавливается флаг OST в регистре TZFLG, а также генерируется прерывание PWM_TZINT, если разрешено в регистре TZEINT и контроллером прерываний. Аварийное удержание выводов заканчивается после принудительного программного сброса записью в бит OST регистра TZCLR.

Аварийное состояние выводов при получении события сбоя программируется индивидуально для выходов PWM_A и PWM_B полями TZA и TZB регистра TZCTL.

14.9 Триггер событий

Основные функции триггера событий:

- получение событий, сформированных таймером и компаратором;
- использование информации о направлении счета (вверх-вниз);
- использование делителя событий для формирования сигнала прерывания, запросов к блокам контроллера DMA и АЦП;
- предоставление доступа процессора к содержимому регистра флагов событий и счетчикам событий.

Триггер событий программируется посредством регистров ETSEL, ETPS, ETFLG, ETCLR и ETFRC.

Прерывания

Функциональная схема триггера событий для генерации прерываний показана на рисунке 14.20. Триггер может генерировать прерывания (если разрешено битом INTEN регистра ETSEL) по каждому событию, а также в два, три и четыре раза реже. Коэффициент деления событий задается полем INTPRD. Источник события выбирается полем INTSEL. Количество возникших событий отражается в поле INTCNT. Счетчик INTCNT считает от 00b до INTPRD и сбрасывается только вместе с отправкой активного прерывания.

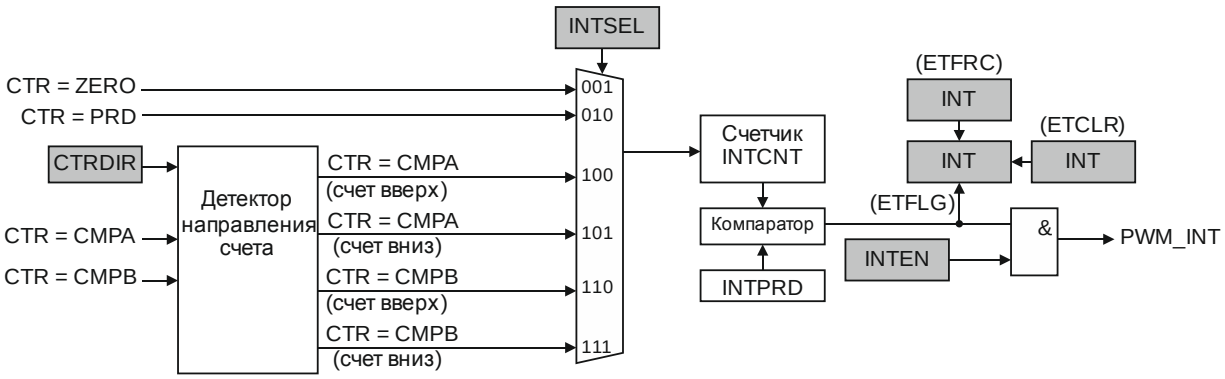


Рисунок 14.20 – Структурная схема триггера событий для генерации прерываний

Когда возникает совпадение значений счетчиков INTCNT и INTPRD, то возможны варианты:

- если прерывание разрешено и сброшен флаг INT (регистр ETFLG), то генерируется прерывание и устанавливается флаг INT, а счетчик INTCNT сбрасывается в 00b и начинает считать заново;
- если прерывание запрещено или флаг INT установлен, то счетчик перестает считать события;
- если прерывание разрешено, но флаг от предыдущего прерывания еще не сброшен, то счетчик хранит свое максимально достигнутое значение INTCNT = INTPRD до сброса флага INT. Это позволяет обработать еще прерывание, пришедшее за то время, пока обрабатывалось предыдущее.

Каждая запись в INTPRD сбрасывает счетчик INTCNT. Запись единицы в бит INT регистра ETFRC увеличит значение счетчика на единицу.

Сопряжение с блоком DMA

Механизм сопряжения модуля ШИМ с блоком DMA реализован аналогично механизму сопряжения модуля ШИМ с триггером событий. Управление запросами блока DMA: сопряжение с сигналами PWM_DRQA, PWM_DRQB осуществляется через регистры ETSEL, ETPS, ETFRC аналогично механизму управления сигналом прерывания PWM_INT. Функциональная схема триггера событий для генерации запросов блока DMA показана на рисунке 14.21.

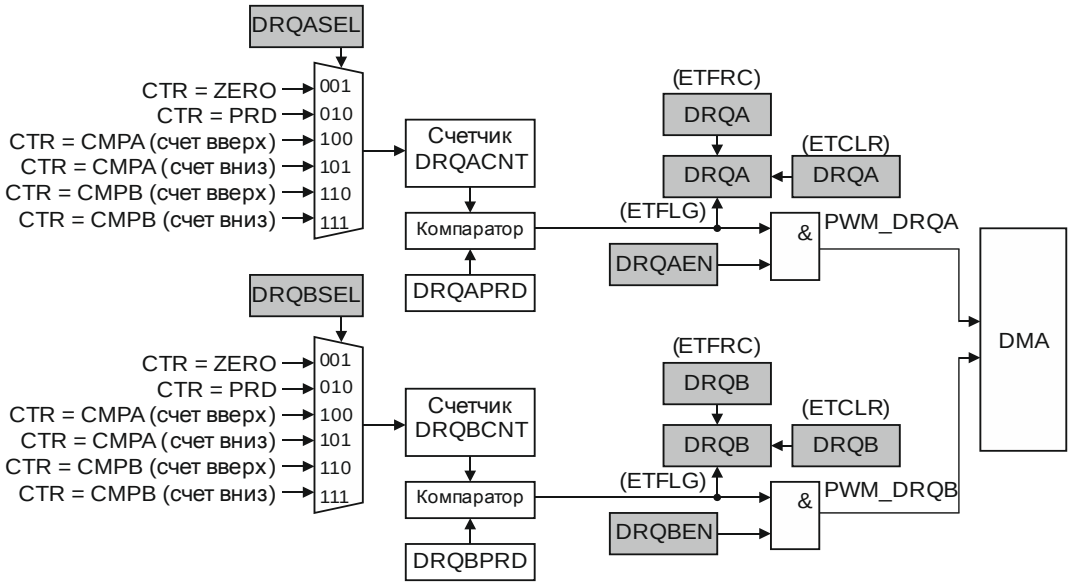


Рисунок 14.21 – Функциональная схема триггера событий для генерации запросов контроллера DMA

Сопряжение с блоком АЦП

Управление сигналами запуска блока АЦП PWM_SOCA, PWM_SOCB осуществляется аналогично сигналу прерывания PWM_INT и запросам PWM_DRQA, PWM_DRQB через регистры ETSEL, ETPS, ETFRC.

Функциональная схема триггера событий для генерации сигнала запуска АЦП показана на рисунке 14.22.

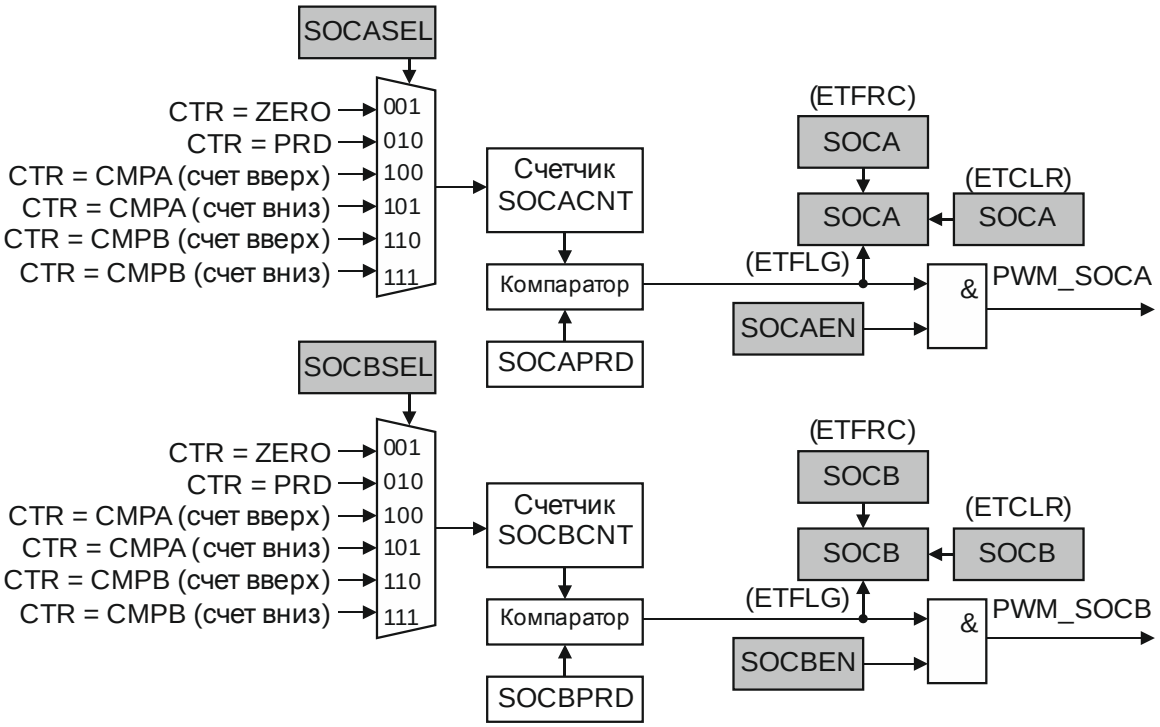


Рисунок 14.22 – Функциональная схема триггера событий для генерации сигнала запуска АЦП

Выходы всех трёх блоков ШИМ0 – ШИМ2 (PWMx_SOCA и PWMx_SOCB каждого блока, где x = 0, 1, 2) объединяются по ИЛИ, как показано на рисунке 14.23. Сигналы с выходов элементов ИЛИ защелкиваются в триггерах и формируют импульсы запуска секвенсоров блока АЦП.

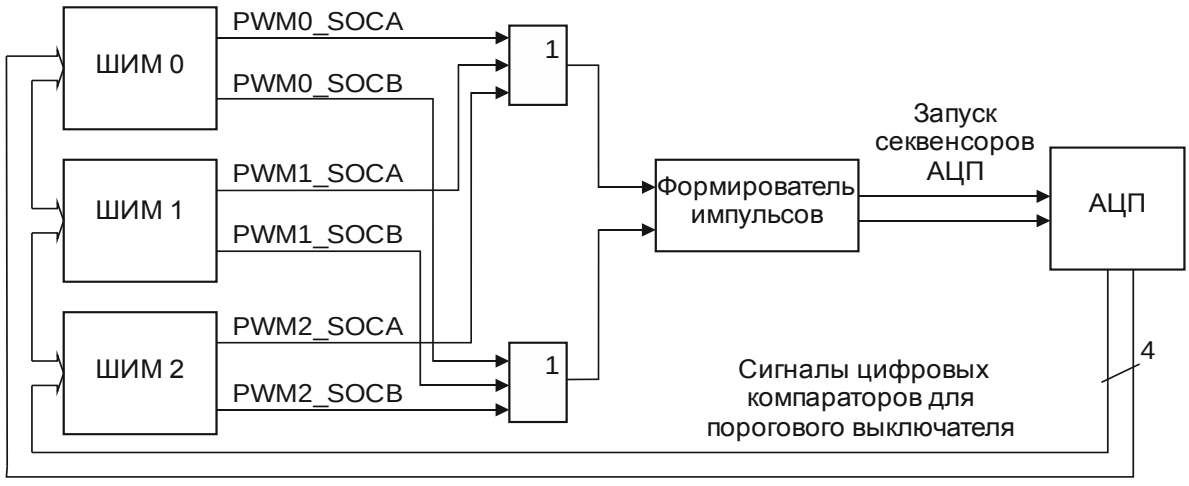


Рисунок 14.23 – Схема соединений между блоками ШИМ и АЦП

15 Приемопередатчик UART

В состав микроконтроллера входят два идентичных универсальных асинхронных приемопередатчика UART0, UART1.

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, а также в случае обнаружения ошибки;
- возможность деления тактовой частоты в диапазоне от 1 до 65 535 (допускается использование нецелых коэффициентов деления, что позволяет использовать любой опорный генератор с частотой более 3,6864 МГц);
- поддержку прямого доступа к памяти.

Приемопередатчик реализует:

- передачу данных длиной от 5 до 8 бит со скоростью до 921 600 бит/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), от передатчика, а также 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.

15.1 Функционирование блока UART

На рисунке 15.1 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.

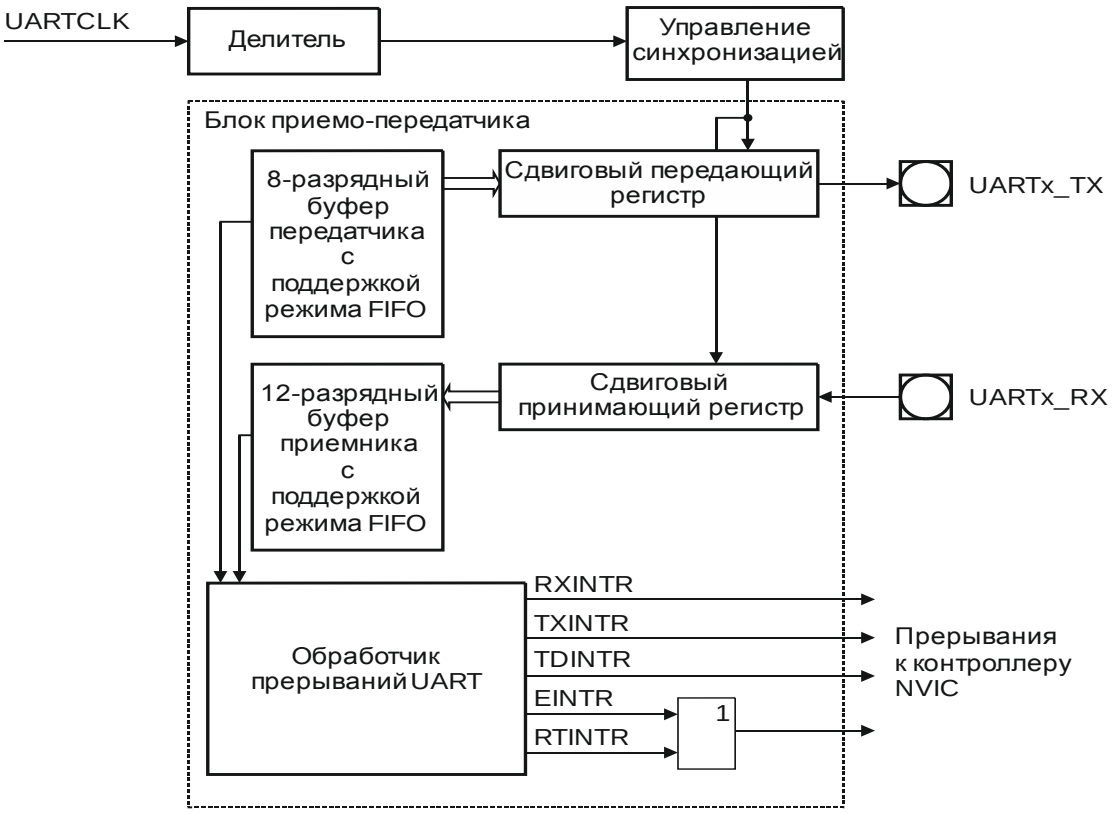


Рисунок 15.1 – Упрощенная функциональная схема приемопередатчика

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 NVIC.

Сброс модуля

Приемопередающая логика модуля, управляющие регистры и FIFO по умолчанию находятся в сбросе. Снять сброс можно путём установки бита RSTDIS в соответсвующем регистре UARTCFGi (i – номер модуля 0 или 1) блока RCU.

Синхронизация

Существует ограничение на соотношение между частотами тактовых сигналов PCLK и UARTCLK:

$$f_{UARTCLK} \leq 5/3 \times f_{PCLK} \qquad (15.1)$$

Например, для достижения максимальной скорости передачи данных 921600 бод (при $f_{UARTCLK}$ = 921600 × 16 = 14,7456 МГц) частота сигнала PCLK 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

$$K = f_{UARTCLK}/(16 \times baudrate), \quad (15.2)$$

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 230400 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:

$K = 1 + 5/64 = 1{,}078$,

$baudrate = (4 \times 10^6)/(16 \times 1{,}078) = 231911$ 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 UART_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 UART_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 UART_RX). После этого байт заносится в буфер приемника вместе с тремя битами признаков ошибки, см. рисунок 15.2, 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 при приеме 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.

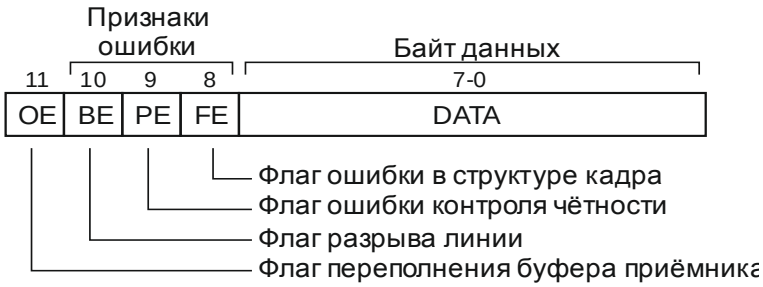


Рисунок 15.2 – 12-разрядная ячейка принимающего буфера

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.

15.2 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.

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.

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.

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.

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.

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.

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, 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 бита 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 15.1 указаны значения параметров срабатывания запросов блочного обмена RXDMABREQ и TXDMABREQ в зависимости от порога заполнения буфера.

Таблица 15.1 – 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.

15.3 Прерывания

В модуле предусмотрено пять маскируемых источников прерывания.

Сигналы запросов на прерывания:

- RXINTR – от приемного FIFO;
- TXINTR – от передающего FIFO;
- RTINTR – по таймауту приемника;
- TDINTR – по окончании передачи в линии;
- EINTR – по ошибке.

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 прерывания, 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 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.

На контроллере прерываний NVIC заведено 4 линии прерываний:

- UART_RX_INT – запрос RXINTR;
- UART_TX_INT – запрос TXINTR;
- UART_TD_INT – запрос TDINTR;
- UART_E_RT_INT – объединенные по ИЛИ запросы сигналов RTINTR и EINTR.

15.4 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.

16 Контроллер интерфейса SPI

Контроллер интерфейса SPI реализует 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).

16.1 Структура контроллера SPI

Упрощенная функциональная схема контроллера SPI с блоком синхронизации показана на рисунке 16.1.

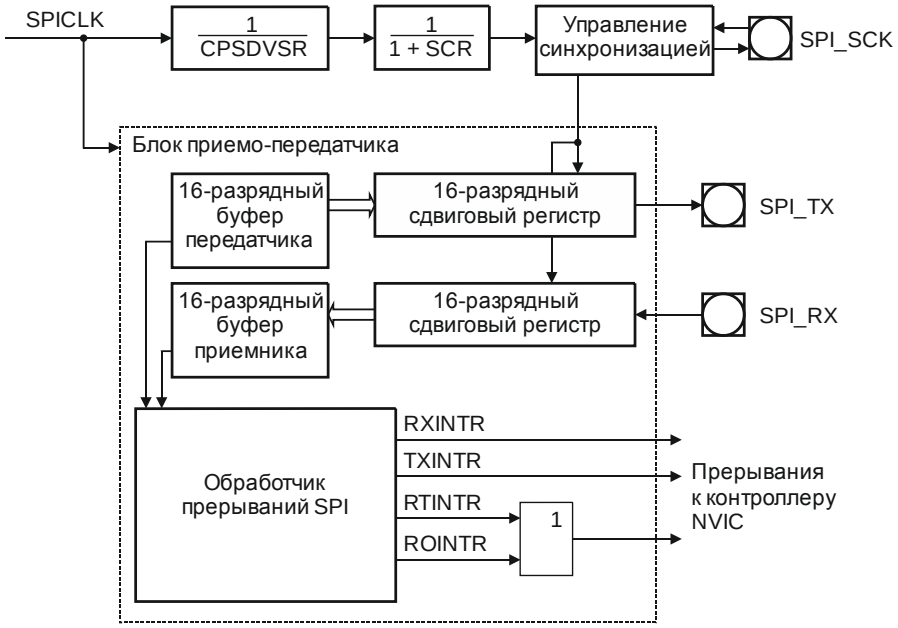


Рисунок 16.1 – Упрощенная функциональная схема контроллера SPI

Синхронизация

Тактирование контроллера SPI осуществляется тактовым сигналом SPICLK, который формируется на основе одного из четырех базовых синхросигналов (см. раздел 4).

Существует ограничение на соотношение между частотами тактовых сигналов PCLK и SPICLK:

$$f_{SPICLK} \leq f_{PCLK} \quad (16.1)$$

В режиме мастера на основе сигнала SPICLK посредством двух последовательно стоящих делителей формируется сигнал тактирования передачи и приема данных с частотой f_{SPI_SCK}, которую можно вычислить по формуле

$$f_{SPI\_SCK} = f_{SPICLK}/(CPSDVSR \times (1+SCR)), \quad (16.2)$$

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 SPI_SCK микроконтроллера и далее к подключенным внешним ведомым устройствам.

В режиме ведомого – значения коэффициентов делителей не важны. Внешний синхросигнал подается на вывод SPI_SCK и тактирует прием и передачу данных.

Для корректной работы всегда должны соблюдаться условия:

- в режиме мастера для формируемого синхросигнала

$$f_{SPI\_SCK} \leq f_{PCLK}/2; \quad (16.3)$$

- в режиме ведомого для входящего внешнего синхросигнала

$$f_{SPI\_SCK} \leq f_{PCLK}/12. \quad (16.4)$$

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 RFFF 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.

16.2 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.

16.3 Функционирование

Приемопередающая логика модуля, управляющие регистры и FIFO по умолчанию находятся в сбросе. Снять сброс можно путём установки бита RSTDIS в соответсвующем регистре SPICFG 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 SPI_TX (передача данных к внешнему устройству) и SPI_RX (прием данных от внешнего устройства).

В зависимости от режима работы сигнал на линии SPI_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 SPI_SCK формируется только тогда, когда приемопередатчик готов к обмену данными. Перевод сигнала SPI_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 SPI_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 SPI_SCK, две линии приема и передачи данных SPI_RX и SPI_TX, а также линию выбора устройства (для режима ведомого) SPI_FSS.

Если устройство функционирует в режиме ведомого, то на его вход SPI_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 SPI_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 выставление данных на линиях SPI_TX и SPI_RX происходит по переднему фронту сигнала синхронизации, а выборка – по заднему.

Передним всегда считается тот фронт сигнала, который является началом передачи первого бита, см. рисунок 16.2.

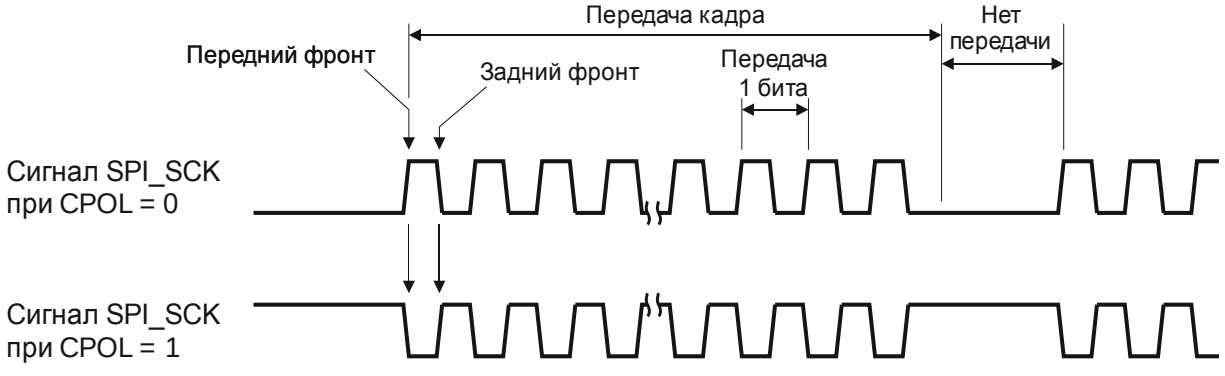


Рисунок 16.2 – Сигнал синхронизации SPI_SCK при разных состояниях бита CPOL

Комбинации битов SPO и SPH задают четыре режима обмена данными, см. рисунок 16.3.

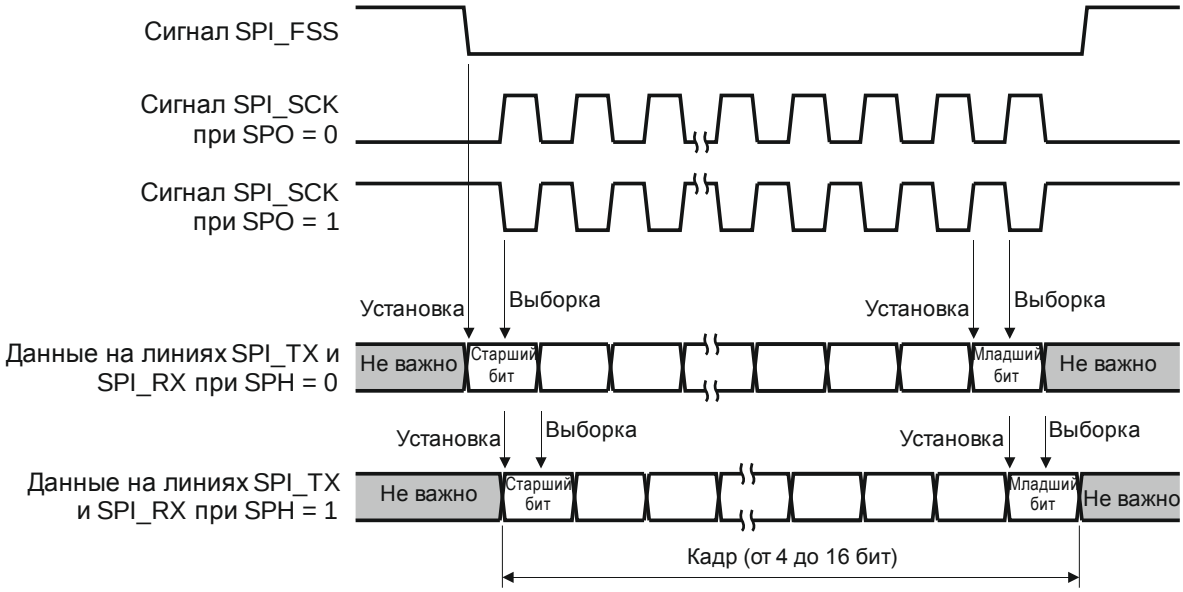


Рисунок 16.3 – Передача кадров данных в интерфейсе SPI

На рисунке 16.4 показано поведение сигналов при непрерывной передаче кадров данных.

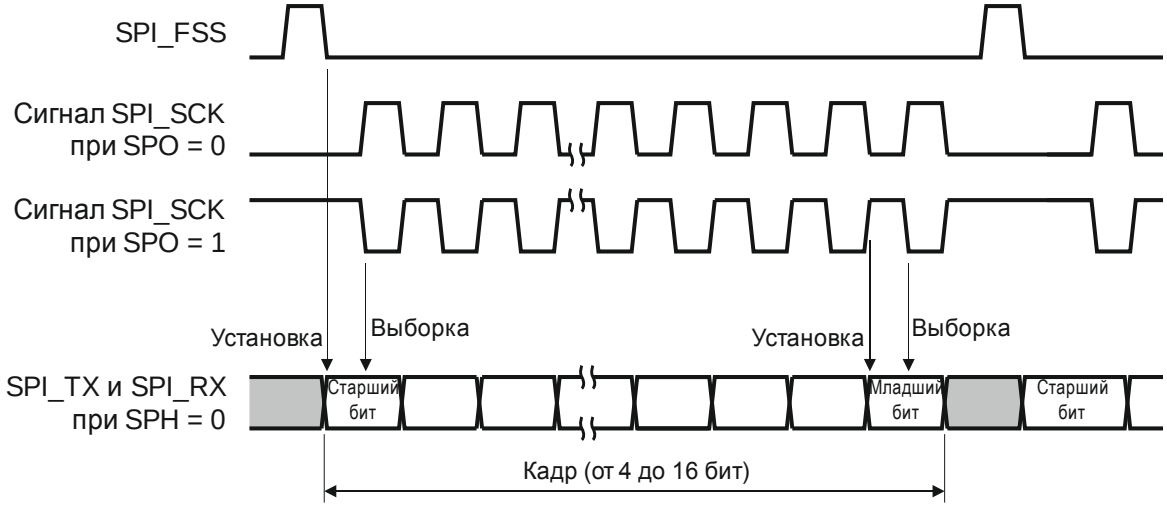


Рисунок 16.4 – Поведение сигналов при непрерывной передаче кадров данных

В режиме непрерывной передачи данных при условии SPH = 0 на линии SPI_FSS должны формироваться импульсы между передачами кадров данных. Это связано с тем, что в этом режиме низкий уровень сигнала на линии SPI_FSS ведомого устройства блокирует запись в сдвиговый регистр. Поэтому мастер должен переводить линию SPI_FSS в высокий уровень по окончании передачи каждого кадра, разрешая, таким образом, запись новых данных. По окончании приема последнего бита кадра линия SPI_FSS переводится в состояние логической единицы по истечении одного такта сигнала SPI_SCK.

В режиме непрерывной передачи данных при условии SPH = 1 низкий уровень сигнала на линии SPI_FSS не блокирует запись в сдвиговый регистр. Линия SPI_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 SPI_SCK, две линии приема и передачи данных SPI_RX и SPI_TX, а также линию выбора устройства (для режима ведомого) SPI_FSS.

Если устройство функционирует в режиме ведомого, то на его вход SPI_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 SPI_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 16.5.

Выставление данных происходит по заднему фронту сигнала SPI_SCK, а считывание – по переднему.

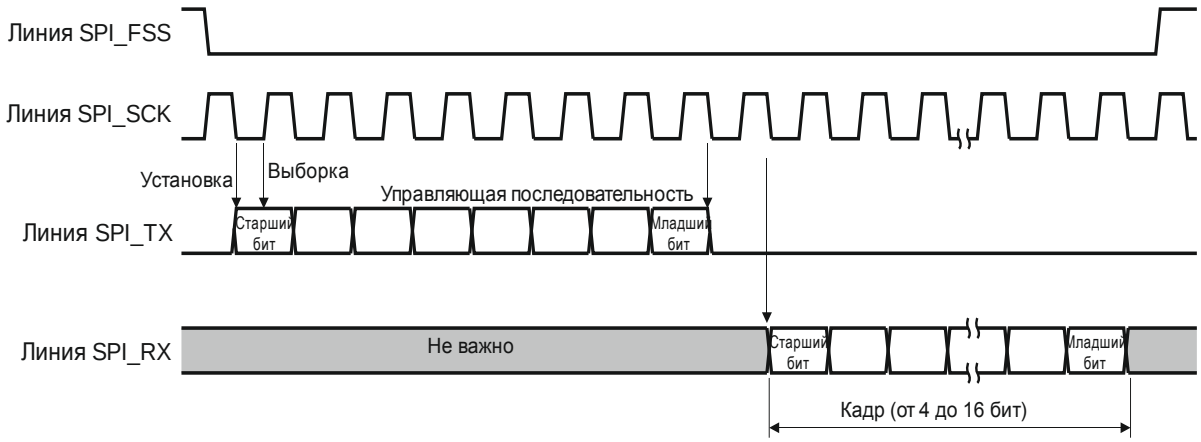


Рисунок 16.5 – Передача кадра данных в интерфейсе Microwire

По окончании приема данных линия SPI_FSS переводится в высокое состояние.

Примечание – В течение времени, когда передается управляющая последовательность и между передачами, линия SPI_RX может находиться в третьем состоянии.

В режиме непрерывной передачи начало и завершение передачи нескольких кадров данных аналогично передаче одного кадра. Линия SPI_FSS удерживается в нуле в течение всего сеанса передачи. По окончании передачи одного кадра данных начинается передача управляющей последовательности без паузы, см. рисунок 16.6.

Примечание – Буферы FIFO приема и передачи данных не очищаются автоматически, даже в случае запрещения работы сбросом бита SSE.

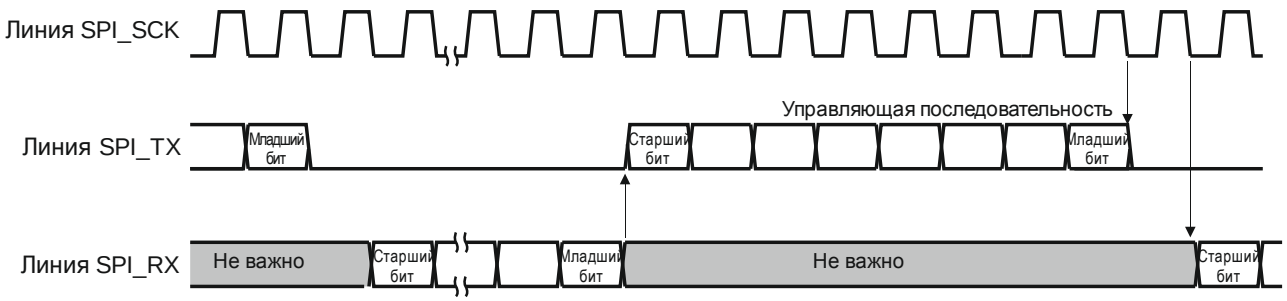


Рисунок 16.6 – Передача кадров данных в интерфейсе Microwire

Интерфейс SSI

Интерфейс SSI реализует полнодуплексный режим передачи данных.

Включает одну линию синхронизации SPI_SCK, две линии приема и передачи данных SPI_RX и SPI_TX, а также линию выбора устройства SPI_FSS.

Перед началом передачи каждого кадра на линии SPI_FSS формируется импульс длительностью в один период сигнала SPI_SCK. Далее мастер и ведомый передают данные. Установка данных производится по переднему фронту синхросигнала, а выборка – по заднему, см. рисунок 16.7. Весь цикл передачи начинается сразу же после появления хотя бы одного элемента в буфере FIFO передатчика.

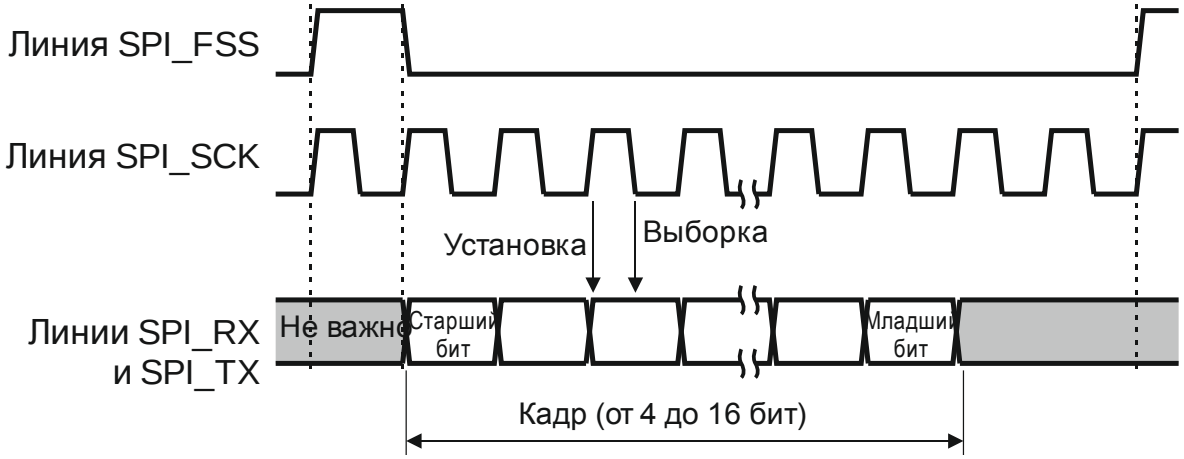


Рисунок 16.7 – Передача кадра данных в интерфейсе SSI

Режим непрерывной передачи кадров данных показан на рисунке 16.8.

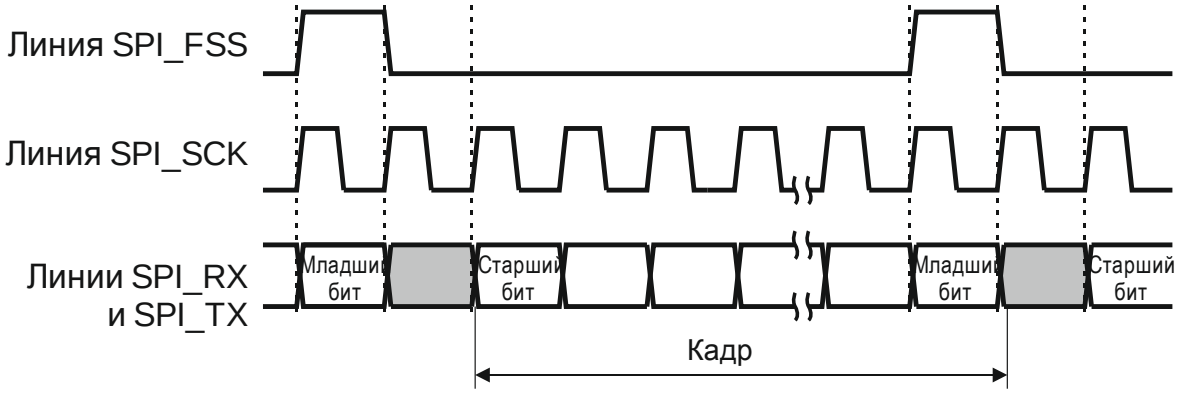


Рисунок 16.8 – Непрерывная передача кадров данных в интерфейсе SSI

16.4 Прерывания

Буфер приема и буфер передачи могут генерировать четыре 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 прерываний 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 контроллере прерываний NVIC заведено 3 линии прерываний:

- SPI_RX_INT – запрос RXINTR;
- SPI_TX_INT – запрос TXINTR;
- SPI_RO_RT_INT – объединенные по ИЛИ запросы ROINTR и RTINTR.

17 Контроллер интерфейса I2C

Модуль контроллера I2C обеспечивает 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.

17.1 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 функцию «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 17.1.

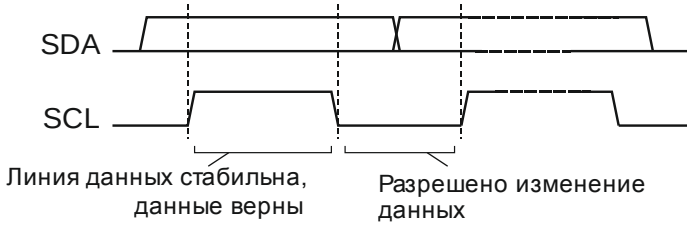


Рисунок 17.1 – Передача бита данных

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 17.2.

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 17.2.

Состояния старта и стопа формирует только мастер.

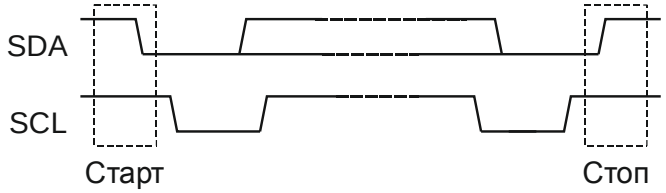


Рисунок 17.2 – Состояния старта и стопа

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 17.3.

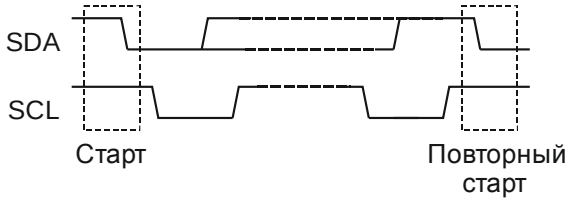


Рисунок 17.3 – Состояние повторного старта

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 17.4 приведен пример арбитража.

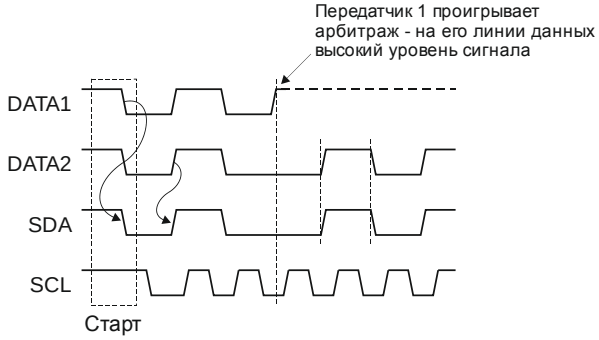


Рисунок 17.4 – Арбитраж на линии SDA

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 17.5.

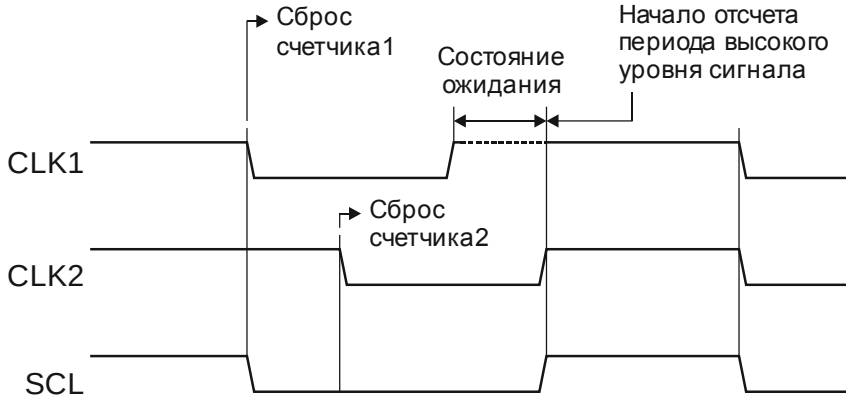


Рисунок 17.5 – Синхронизация

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 17.5). 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 17.5).

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 17.5. 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 17.6.

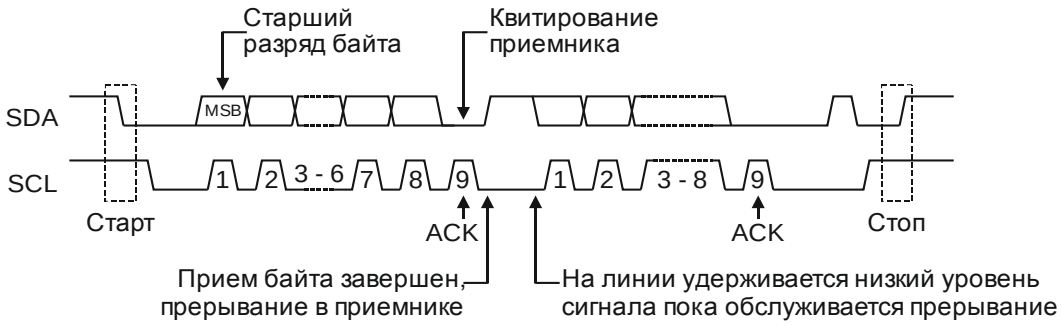


Рисунок 17.6 – Передача данных

Квитирование

Каждый байт посылки должен быть завершен квитированием, т. е. ответом на прием сигнала запроса подтверждения приема (бит ACK). На рисунках 17.6 и 17.7 показано положение момента квитирования в пределах посылки.

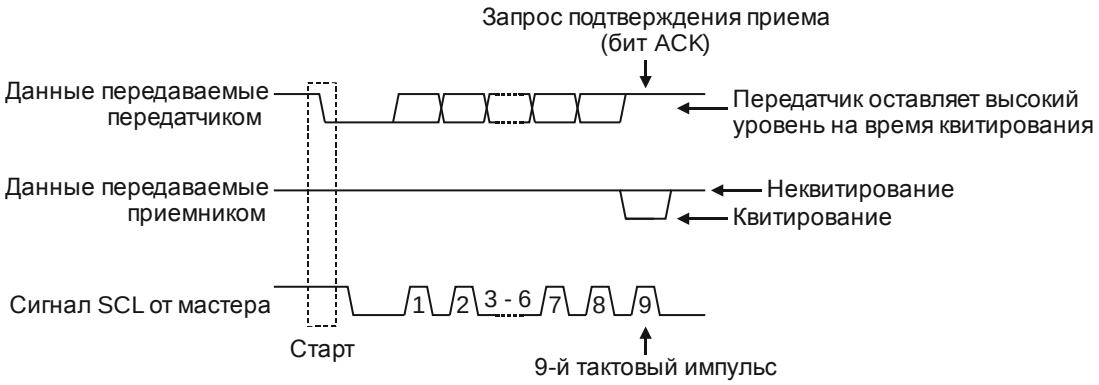


Рисунок 17.7 – Квитирование и неквитирование бита подтверждения ACK

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 17.7. 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 17.8 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»).

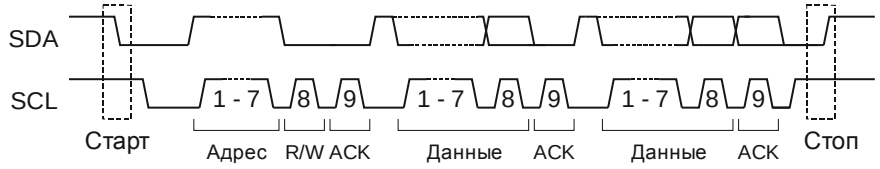


Рисунок 17.8 – Передача данных с 7-битной адресацией

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 17.9.

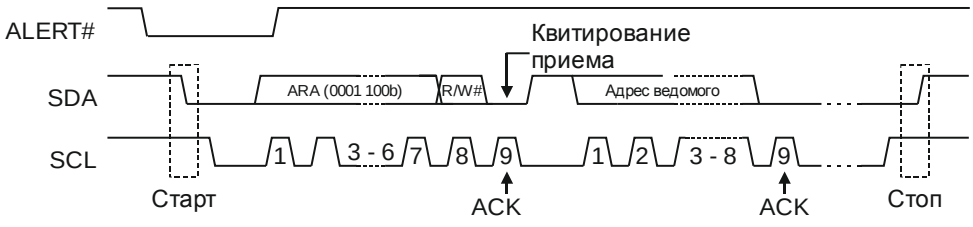


Рисунок 17.9 – Передача адреса отклика на сигнал предупреждения

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 17.10 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 17.10.

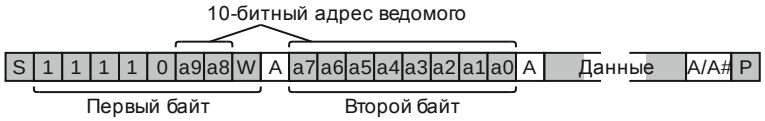


Рисунок 17.10 – Передача данных ведомому с 10-битным адресом (для расшифровки применяемых обозначений следует обратиться к таблице 17.1)

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 17.11.

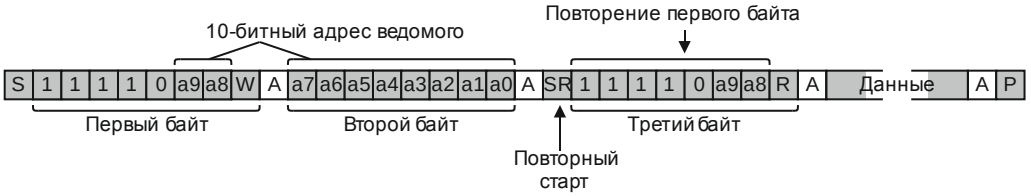


Рисунок 17.11 – Получение данных от ведомого с 10-битным адресом

На рисунках 17.10 и 17.11 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 17.1 с подробными пояснениями.

Таблица 17.1 – 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

17.2 Функциональное описание

Структурная схема модуля I2C представлена на рисунке 17.12. Далее приводится краткое описание назначения блоков модуля.

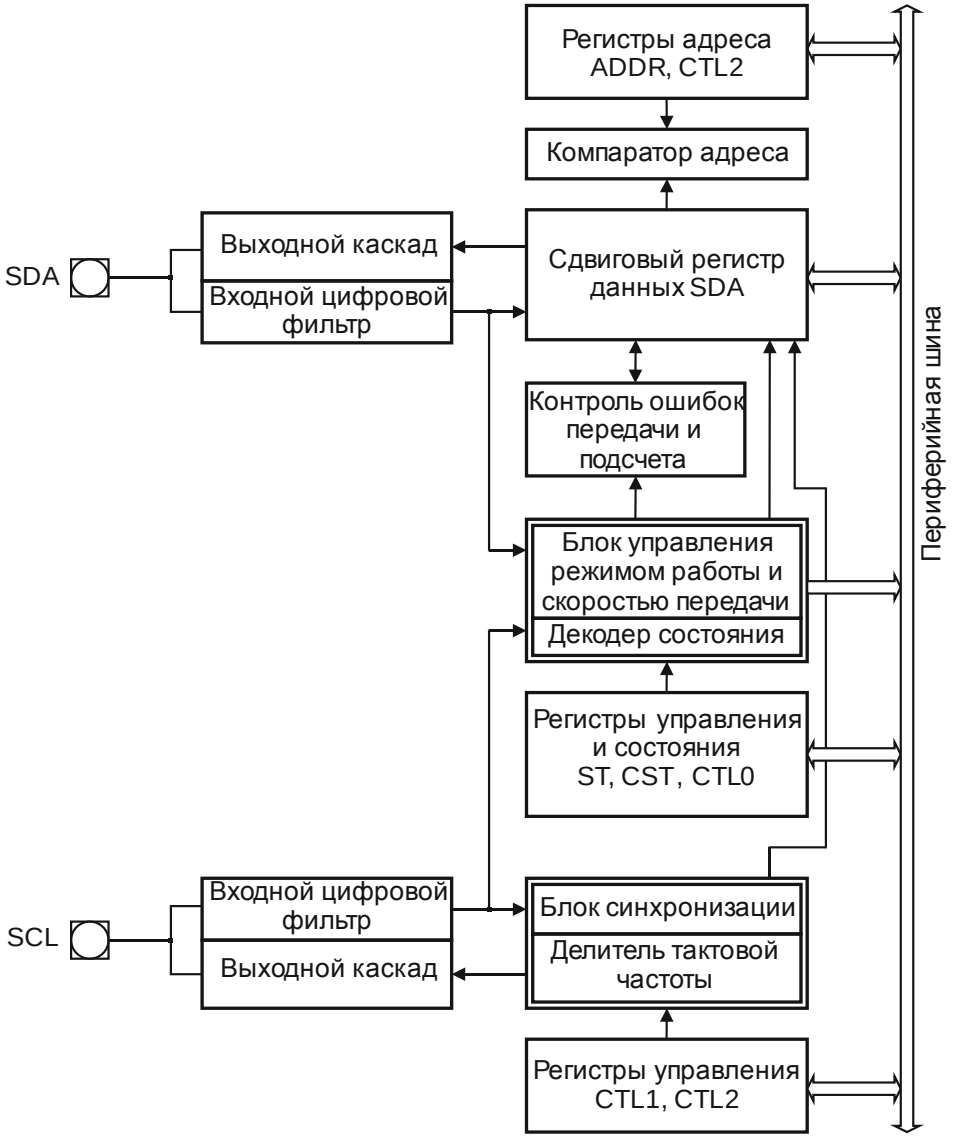


Рисунок 17.12 – Структурная схема модуля I2C

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 1111_0b, 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 17.13.

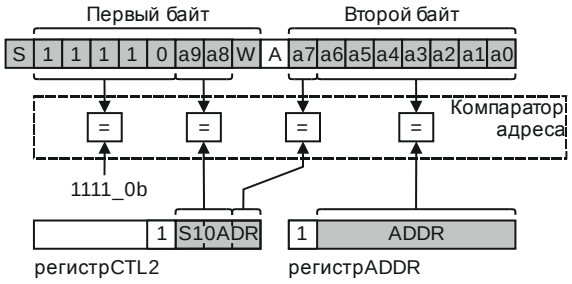


Рисунок 17.13 – Компаратор адреса в режиме 10-битной адресации

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 17.14.

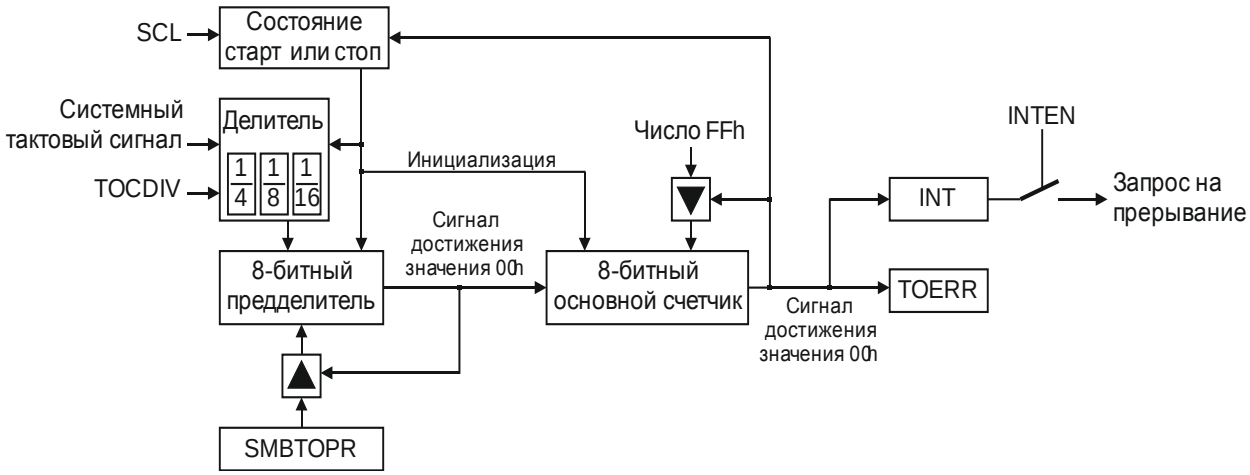


Рисунок 17.14 – Функциональная схема счетчика времени ожидания

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

$$T_{ожид} = T_{PCLK} \times TOCDIV \times (SMBTOPR + 1) \times 256 , \qquad (17.1)$$

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.

17.3 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 17.15. Расшифровка обозначений, принятых на рисунке 17.15, приведена в таблице 17.1.

Все передачи в режиме HS по формату идентичны передачам режима FS, что делает эти два режима полностью совместимыми.

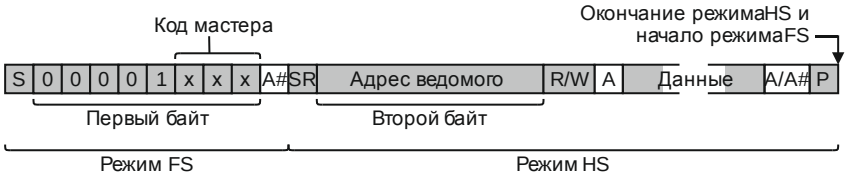


Рисунок 17.15 – Переход в режим HS и обратно в режим FS

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 17.2 и 17.3 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 17.16 – 17.23, поясняющих работу модуля I2C в том или ином режиме, приняты обозначения, расшифровка которых приводится в таблице 17.1. Для получения дополнительной информации и понимания работы модуля I2C можно воспользоваться приложением Б.

Таблица 17.2 – 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 17.2

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 17.3 – 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 17.16 представлено графическое пояснение к описанию режима.

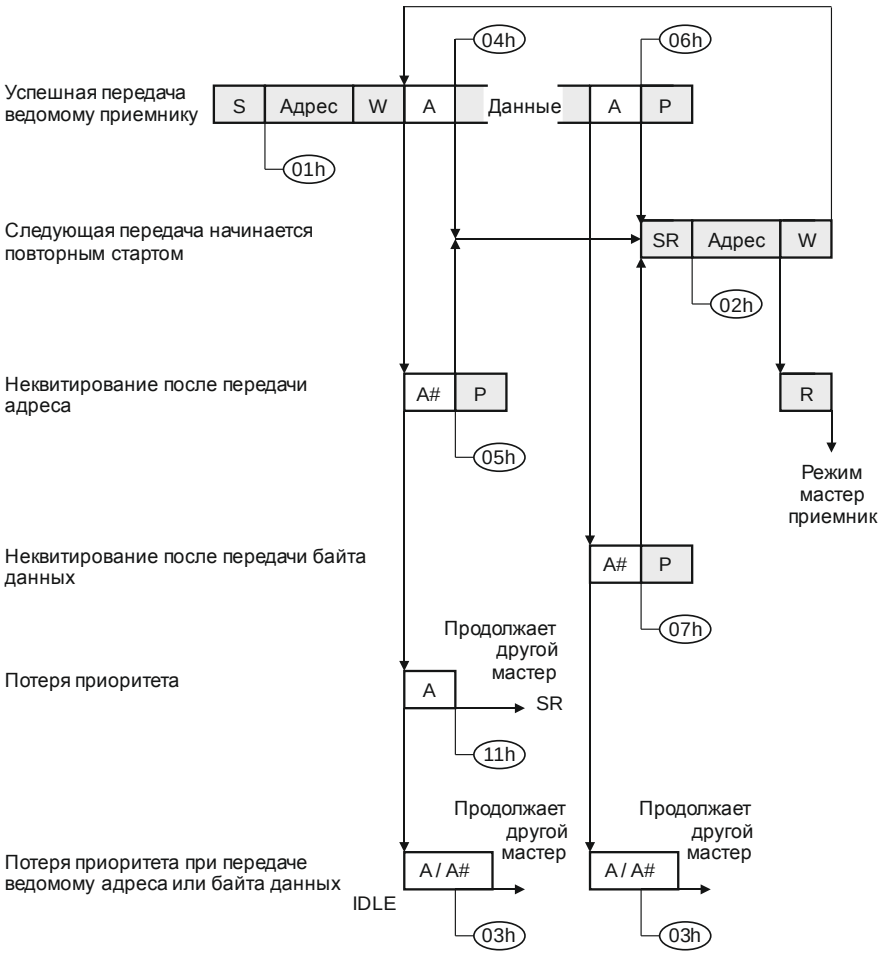


Рисунок 17.16 – Режим FS мастера передатчика

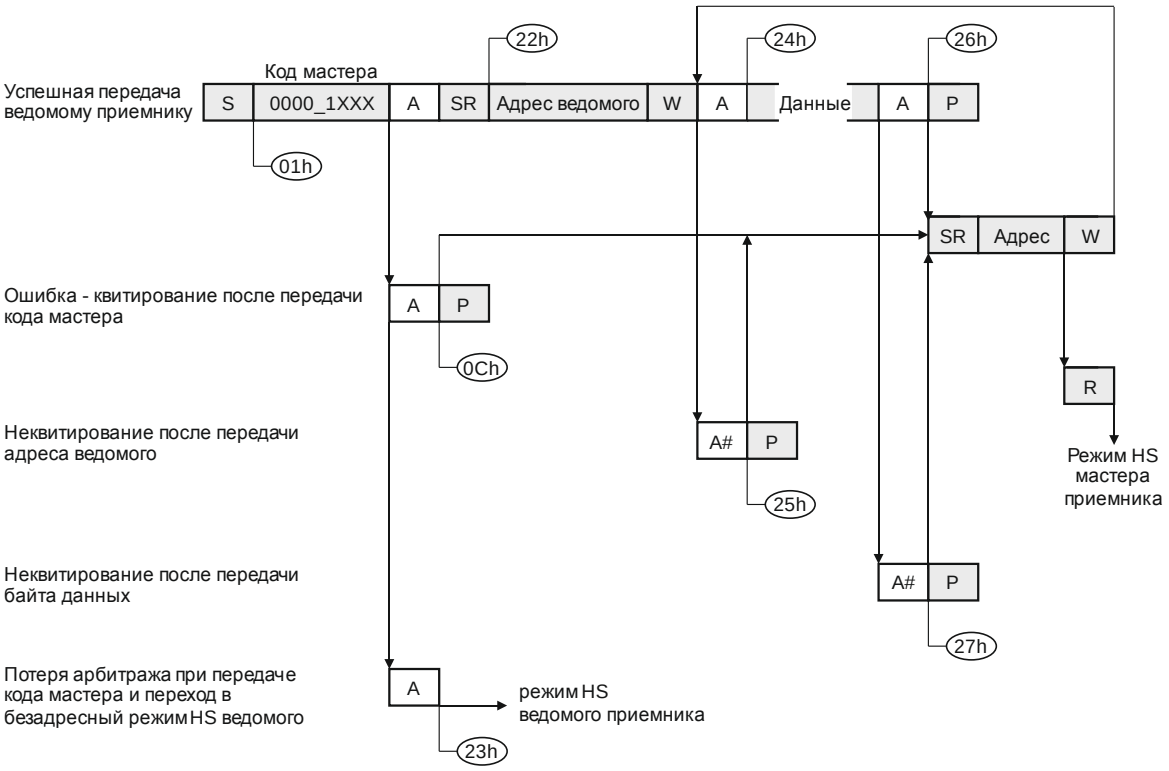


Рисунок 17.17 – Режим HS мастера передатчика

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 17.17 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 17.18 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 17.19 представлено графическое пояснение к описанию режима.

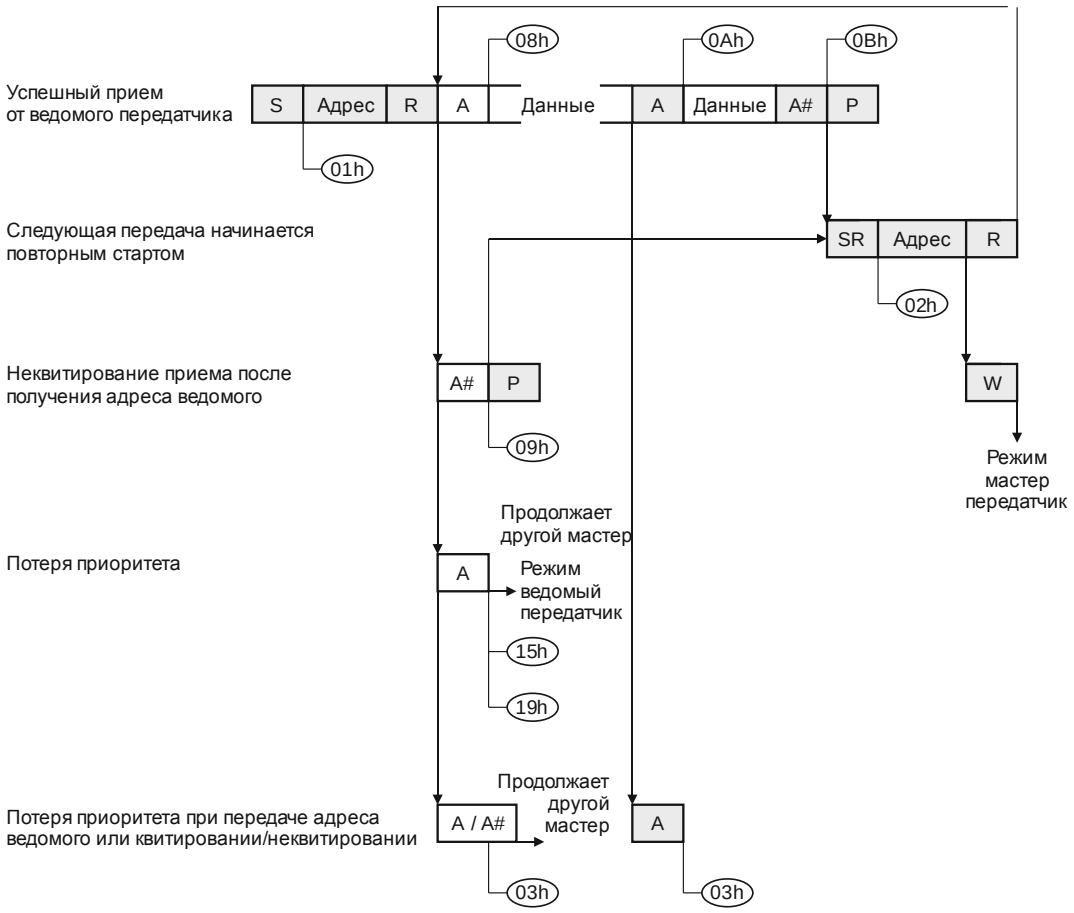


Рисунок 17.18 – Режим FS мастера приемника

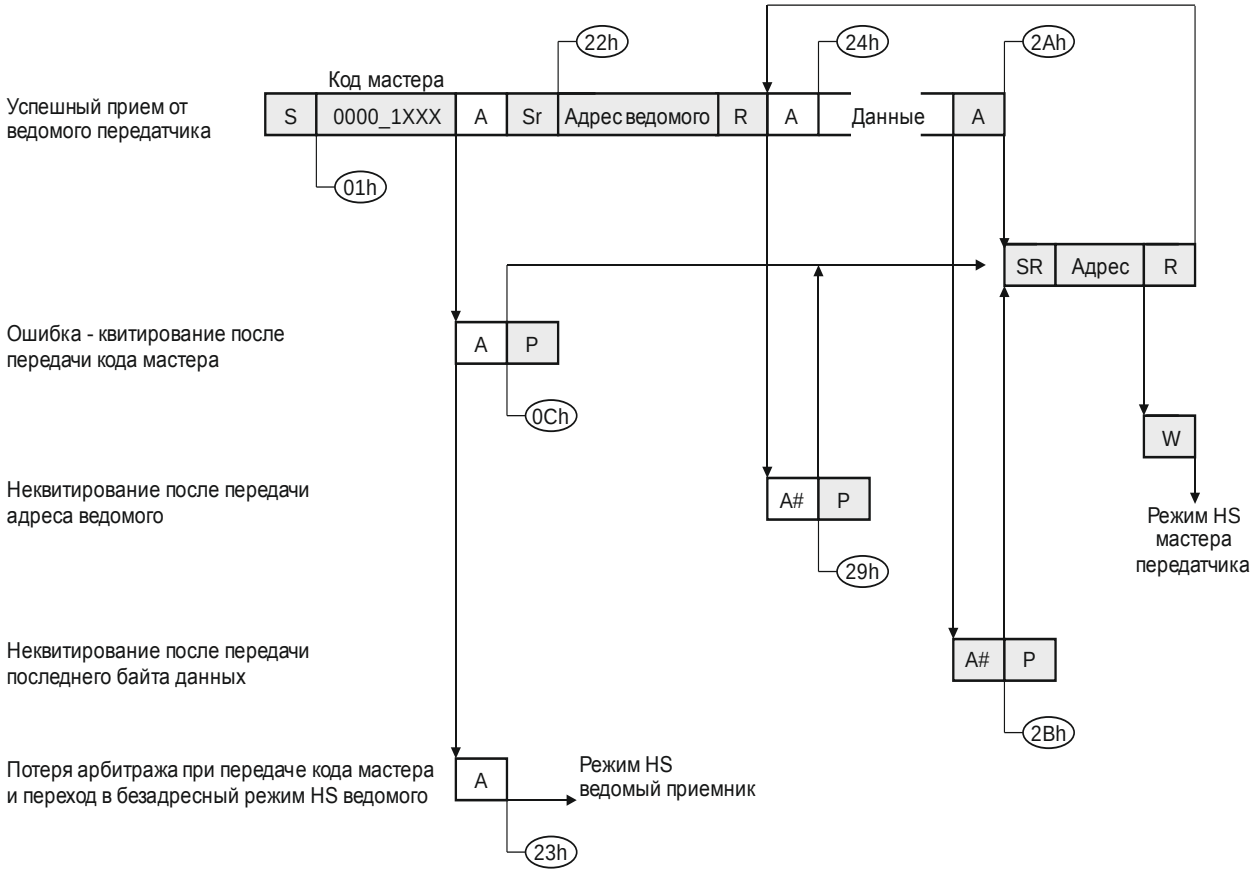


Рисунок 17.19 – Режим HS мастера приемника

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 17.1 настоящего ТО.

Механизм распознавания адреса изложен в подразделе 17.2 настоящего ТО и показан на рисунке 17.13.

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 17.20 представлено графическое пояснение к описанию режима.

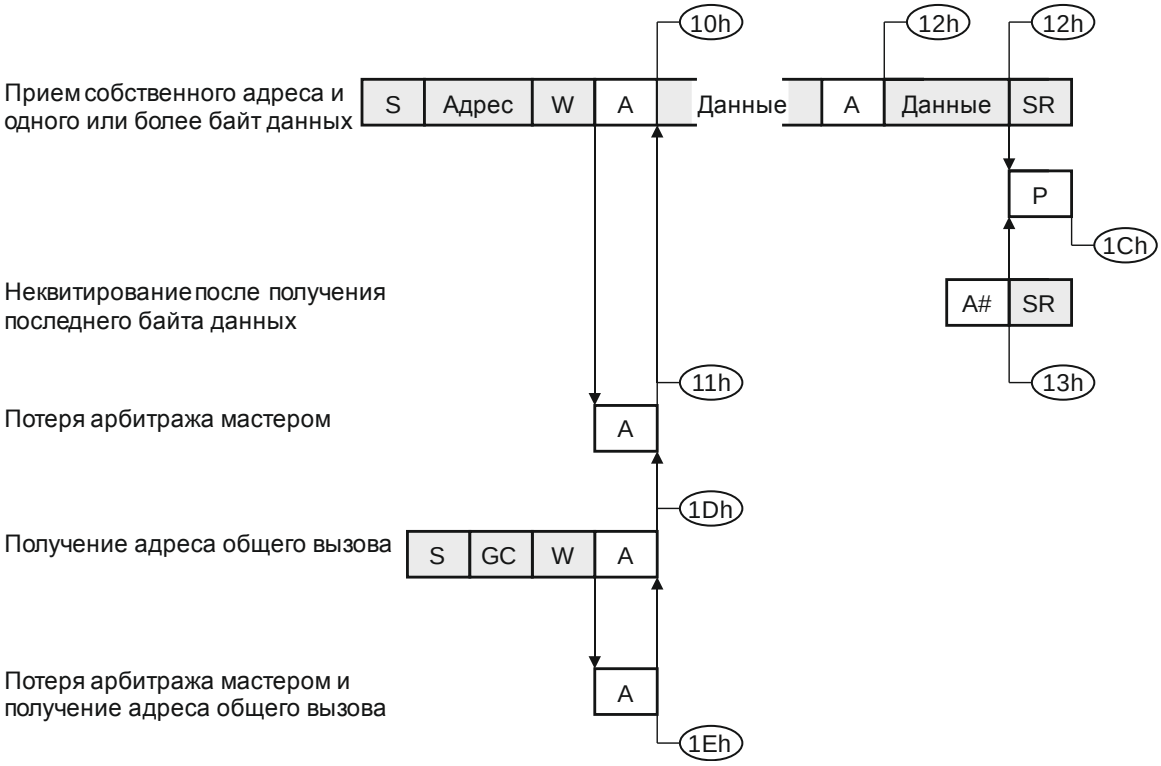


Рисунок 17.20 – Режим FS ведомого приемника

Рисунок 17.21 – 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 17.21 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 «не желает» далее принимать данные.

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 17.22 представлено графическое пояснение к описанию режима.

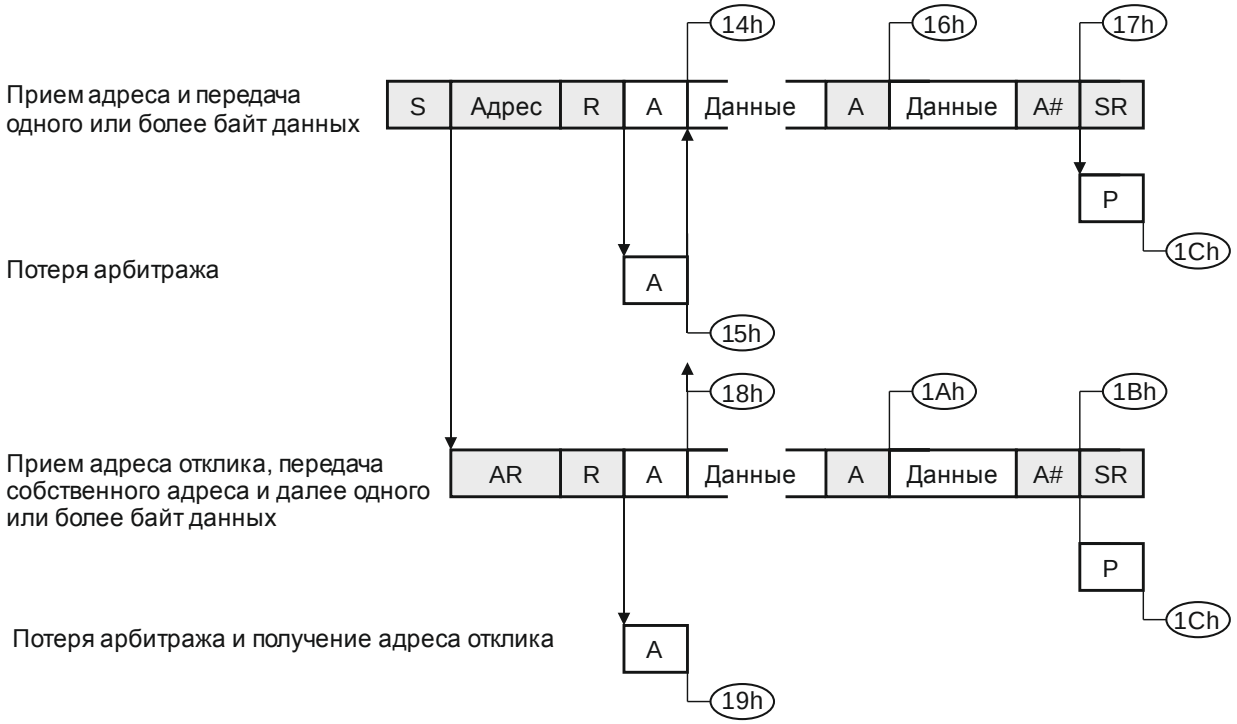


Рисунок 17.22 – Режим FS ведомого передатчика

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 17.23 представлено графическое пояснение к описанию режима.

Рисунок 17.23 – 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.

18 Контроллер интерфейса CAN

18.1 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 электропроводов 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 18.1.

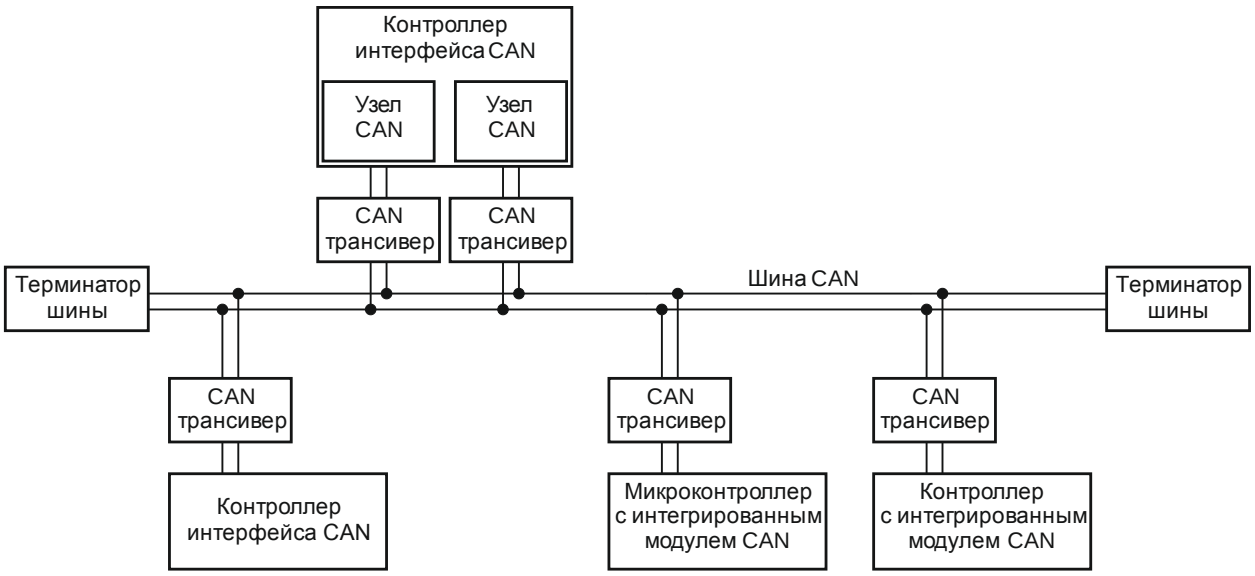


Рисунок 18.1 – Общая структура CAN сети

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

Формируется, когда узел желает передать данные. Формат сообщения показан на рисунке 18.2.

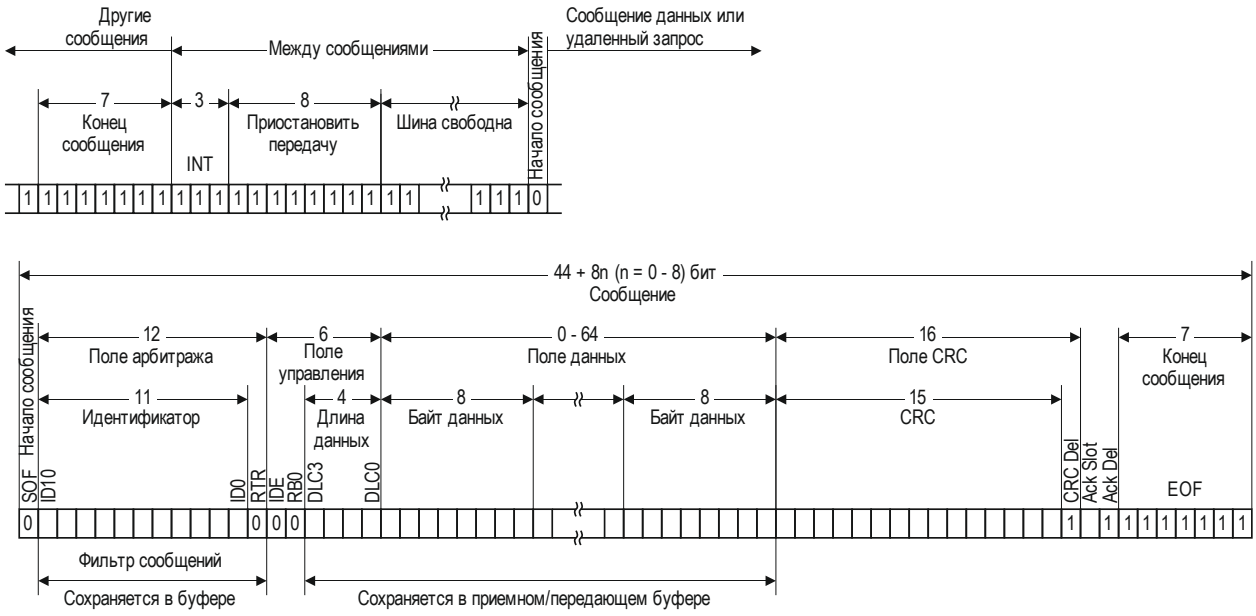


Рисунок 18.2 – Стандартное сообщение данных

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 – число 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 18.3.

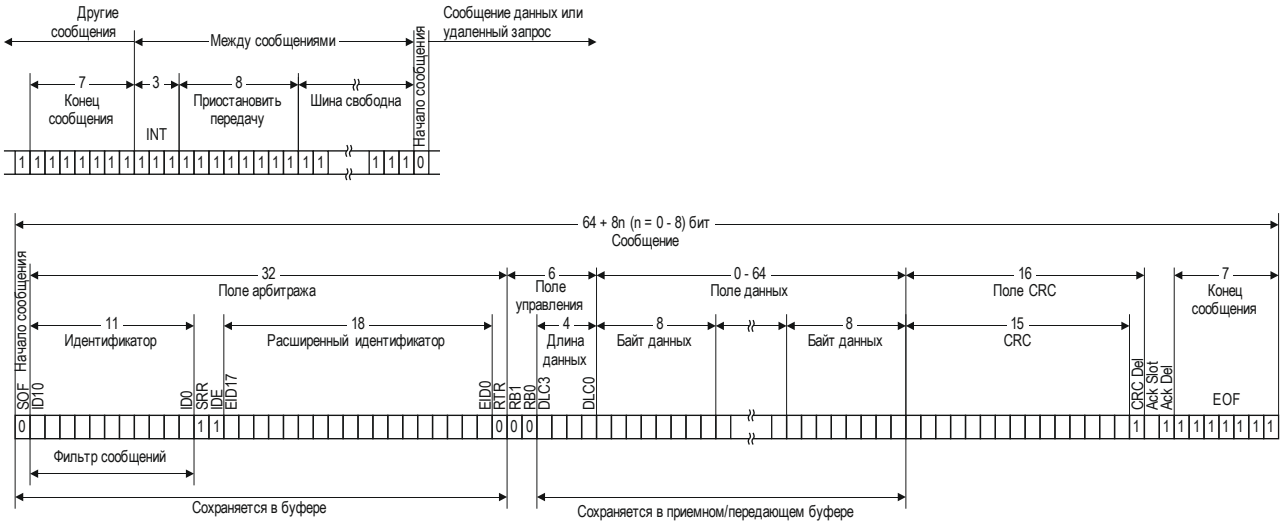


Рисунок 18.3 – Расширенное сообщение данных

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 – число 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 18.4 представлен вариант удаленного запроса со стандартным идентификатором.

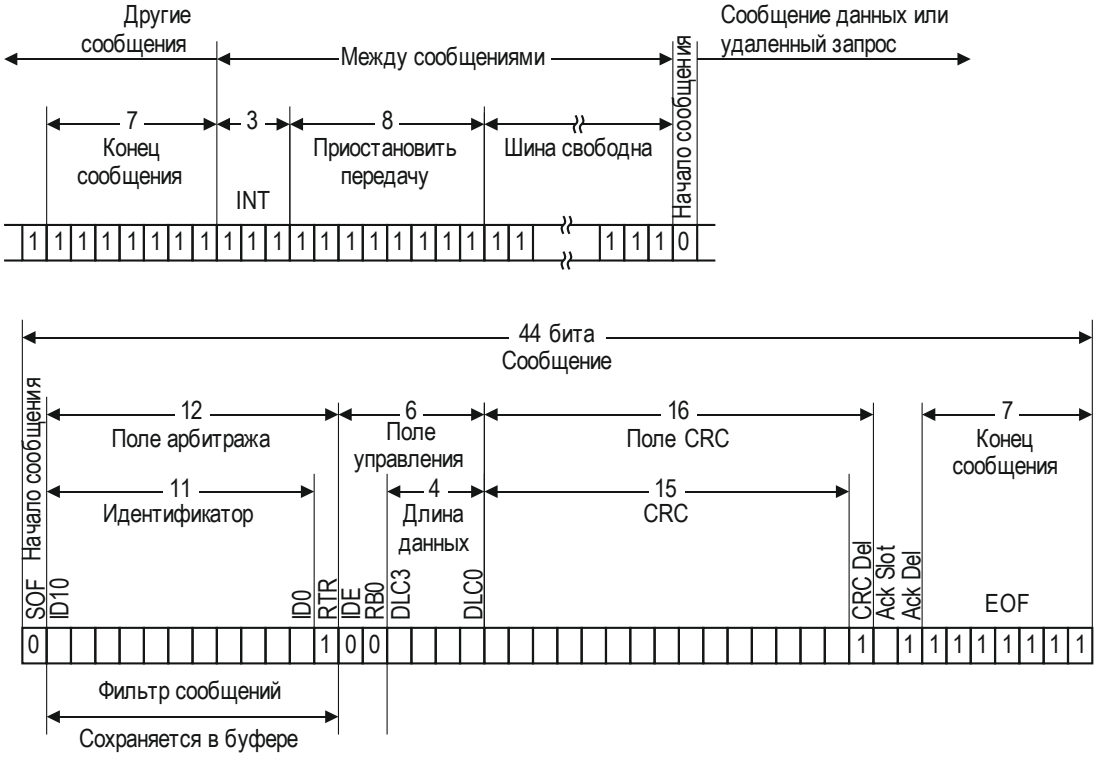


Рисунок 18.4 – Удаленный запрос данных (стандартный формат)

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 18.5.

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 18.5), 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.

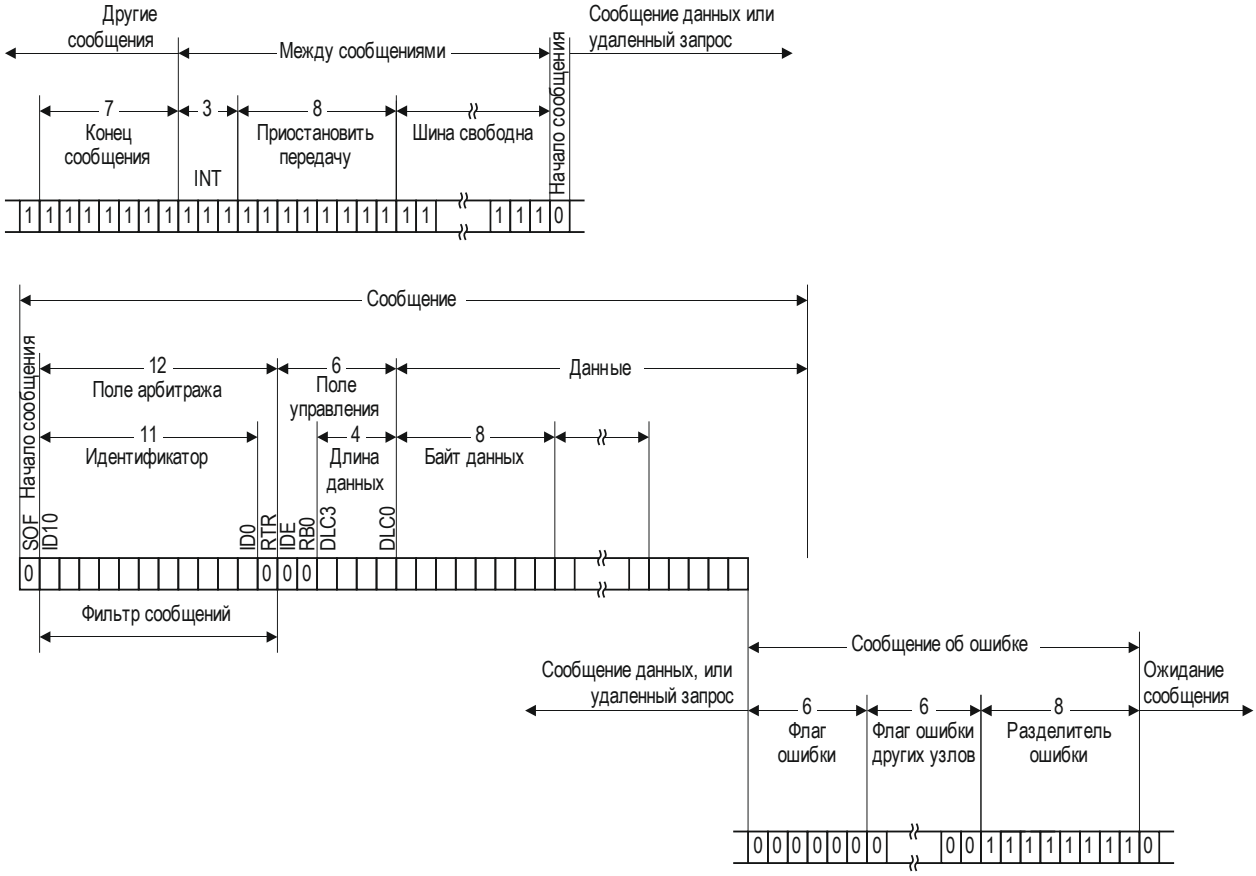


Рисунок 18.5 – Сообщение об ошибке

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-стаффинга» 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 18.6.

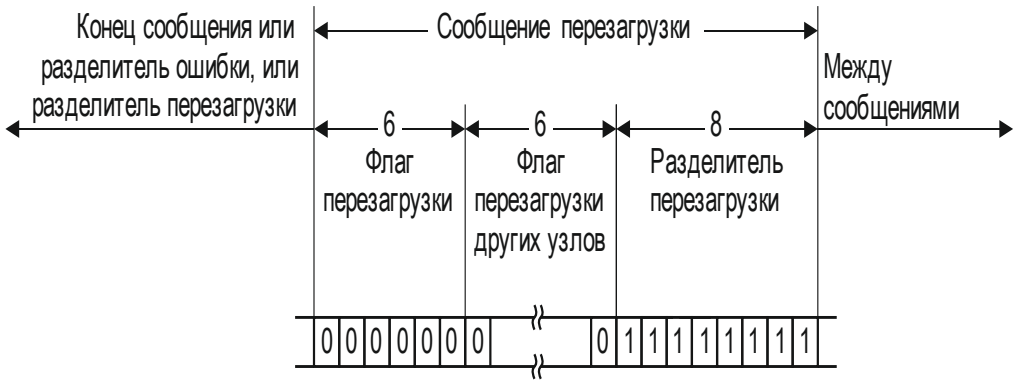


Рисунок 18.6 – Сообщение о перезагрузке

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.

18.2 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 16 линий прерываний и 64 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 64 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 18.7.

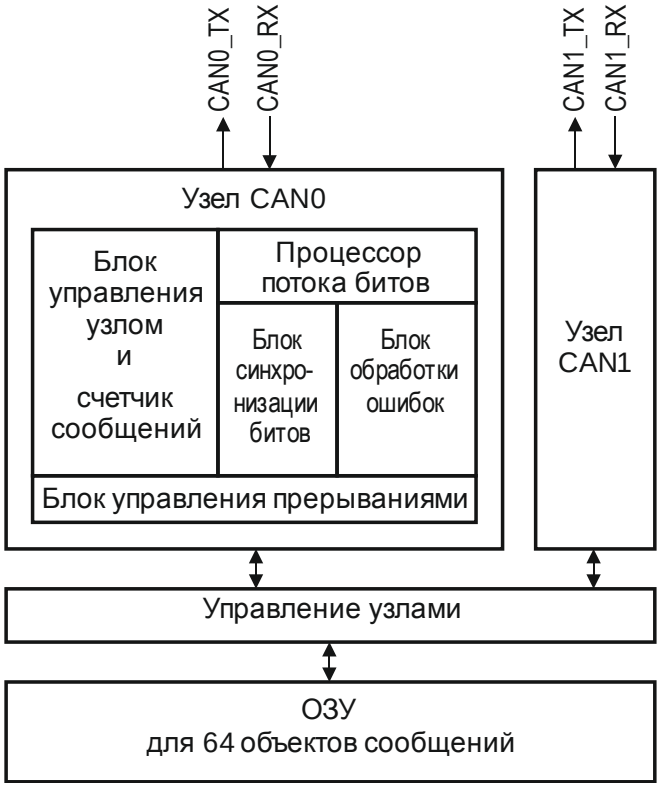


Рисунок 18.7 – Общая структура контроллера CAN

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 были получены 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.

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 18.8. 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.

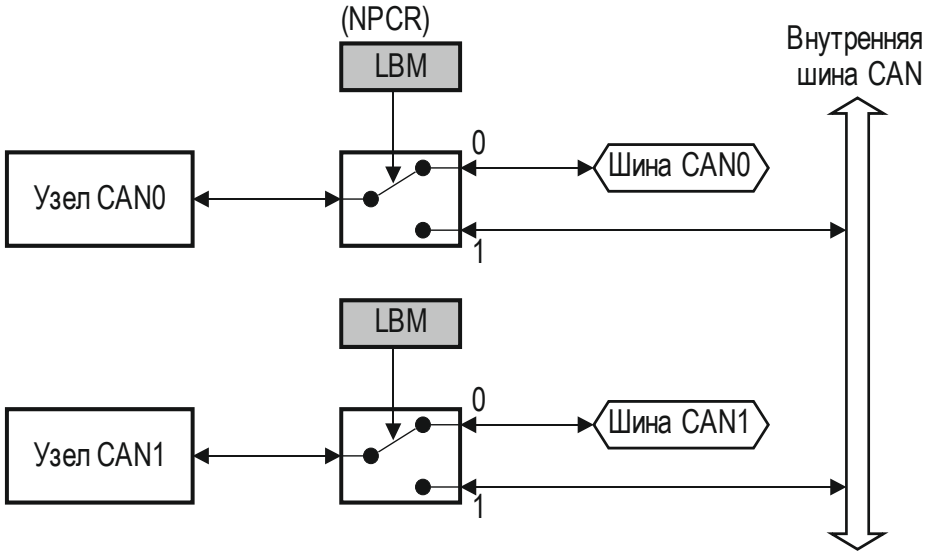


Рисунок 18.8 – Режим внутренней петли

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, см. рисунок 18.9, позволяет генерировать тактовый сигнал FOUT из входного FIN (HCLK) путем программирования делителя посредством регистра FDR.

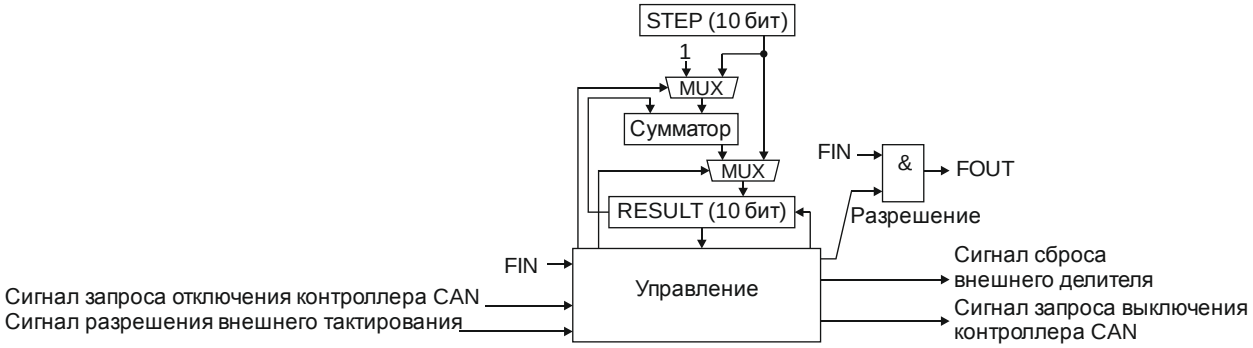


Рисунок 18.9 – Схема дробного делителя

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 сиганал FIN в МГц следует пользоваться формулой:

$$f_{IN} = n \times Ntq, \qquad (18.1)$$

где Ntq – количество квантов времени tq;

n – целое число, начиная с 1 (для задания кратности).

Дробный делитель делит f_{IN} 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 18.9 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

$$f_{OUT} = f_{IN} \times 1/(1024 - STEPd), \qquad (18.2)$$

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 18.10 показано формирование сигнала FOUT при значении STEP = 3FDh (1021d).

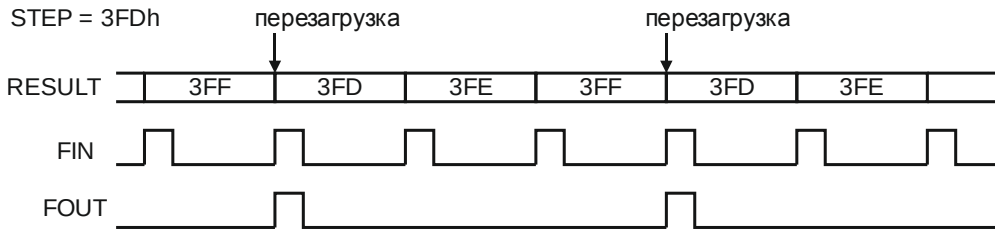


Рисунок 18.10 – Формирование сигнала FOUT в нормальном режиме

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

$$f_{OUT} = f_{IN} \times STEPd/1024d\ . \qquad (18.3)$$

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 18.11 показано формирование сигнала FOUT при значении STEP = 234h (564d). $f_{OUT} = f_{IN} \times 564/1024 = 0{,}55 \times f_{IN}$.

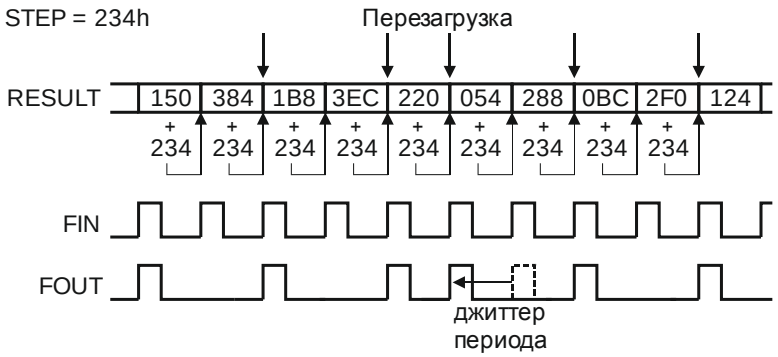


Рисунок 18.11 – Формирование сигнала FOUT в режиме дробного деления

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 18.12 показана структура формирования запроса на прерывание.

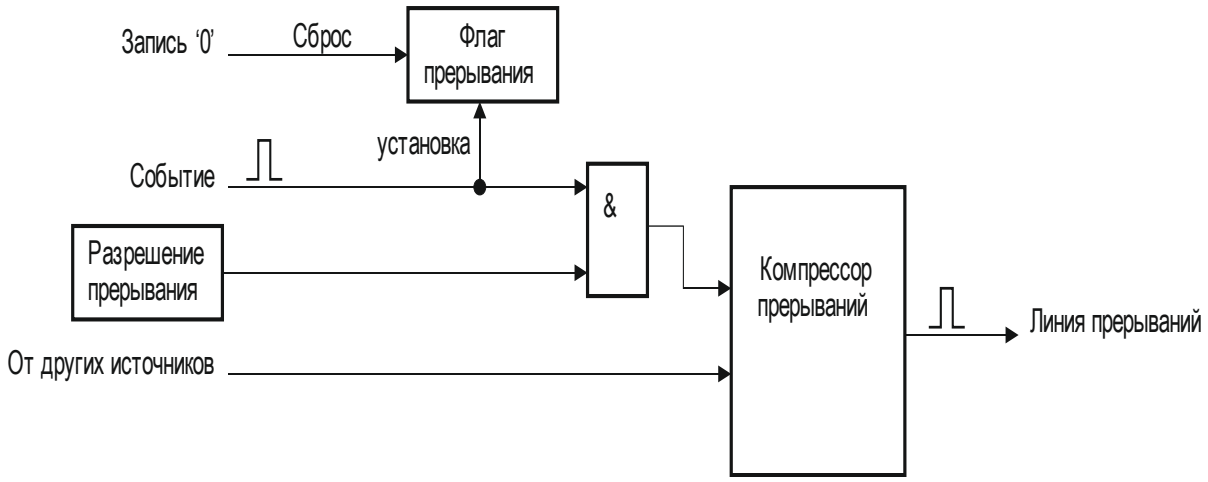


Рисунок 18.12 – Структура формирования запроса на прерывание

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 18.13 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.

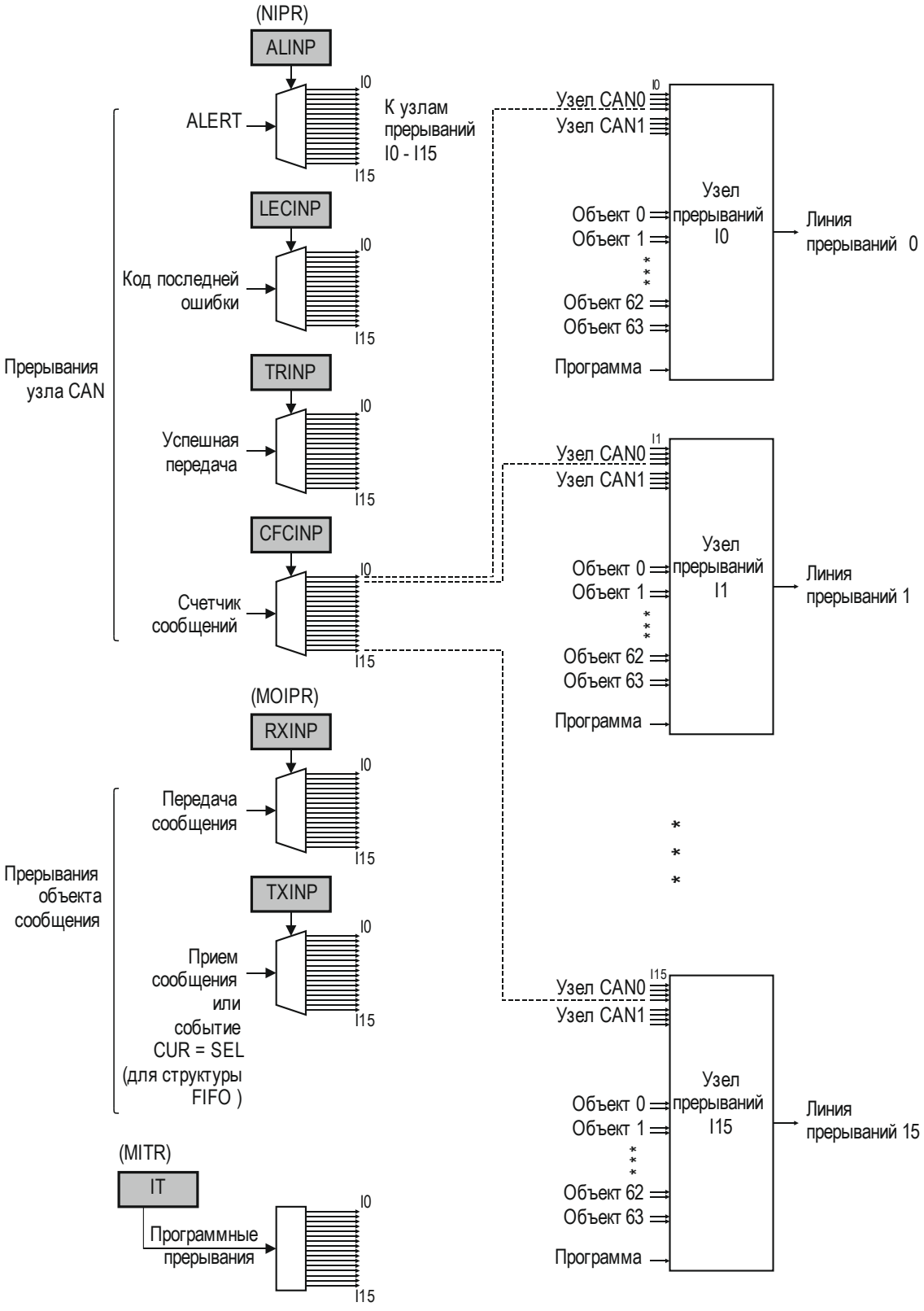


Рисунок 18.13 – Схема коммутации линий прерываний

18.3 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 18.14. 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.

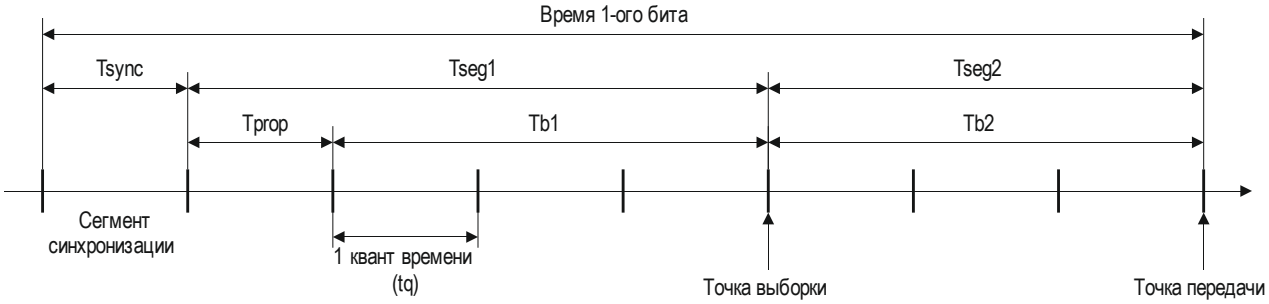


Рисунок 18.14 – Структура одного бита

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

$$tq = (BRP + 1)/f_{OUT}; \quad (18.4)$$

- при DIV8 = 1 значение кванта времени

$$tq = 8 \times (BRP + 1)/f_{OUT}; \quad (18.5)$$

- $Tsync = 1 \times tq$;
- $Tseg1 = (TSEG1 + 1) \times tq \geq 3tq$;
- $Tseg2 = (TSEG2 + 1) \times tq \geq 2tq$;
- $Tbit = Tsync + Tseg1 + Tseg2 \geq 8tq$.

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

$$T_{SJW} = (SJW + 1) \times tq\ . \quad (18.6)$$

Помимо прочего, должны соблюдаться следующие правила:

$$Tseg1 \geq T_{SJW} + Tprop \quad (18.7)$$

$$\text{и } Tseg2 \geq T_{SJW}. \quad (18.8)$$

Соотношения между максимальным отклонением частоты сигнала FOUT и сегментами буферов фаз и шириной перехода ресинхронизации следующие:

$$-\Delta f_{OUT} \leq T/2 \times (13 \times Tbit - Tb2); \quad (18.9)$$

$$-\Delta f_{OUT} \leq T_{SJW}/20 \times Tbit, \quad (18.10)$$

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;
- отключенным 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 64. 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 18.15.

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 16) 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.

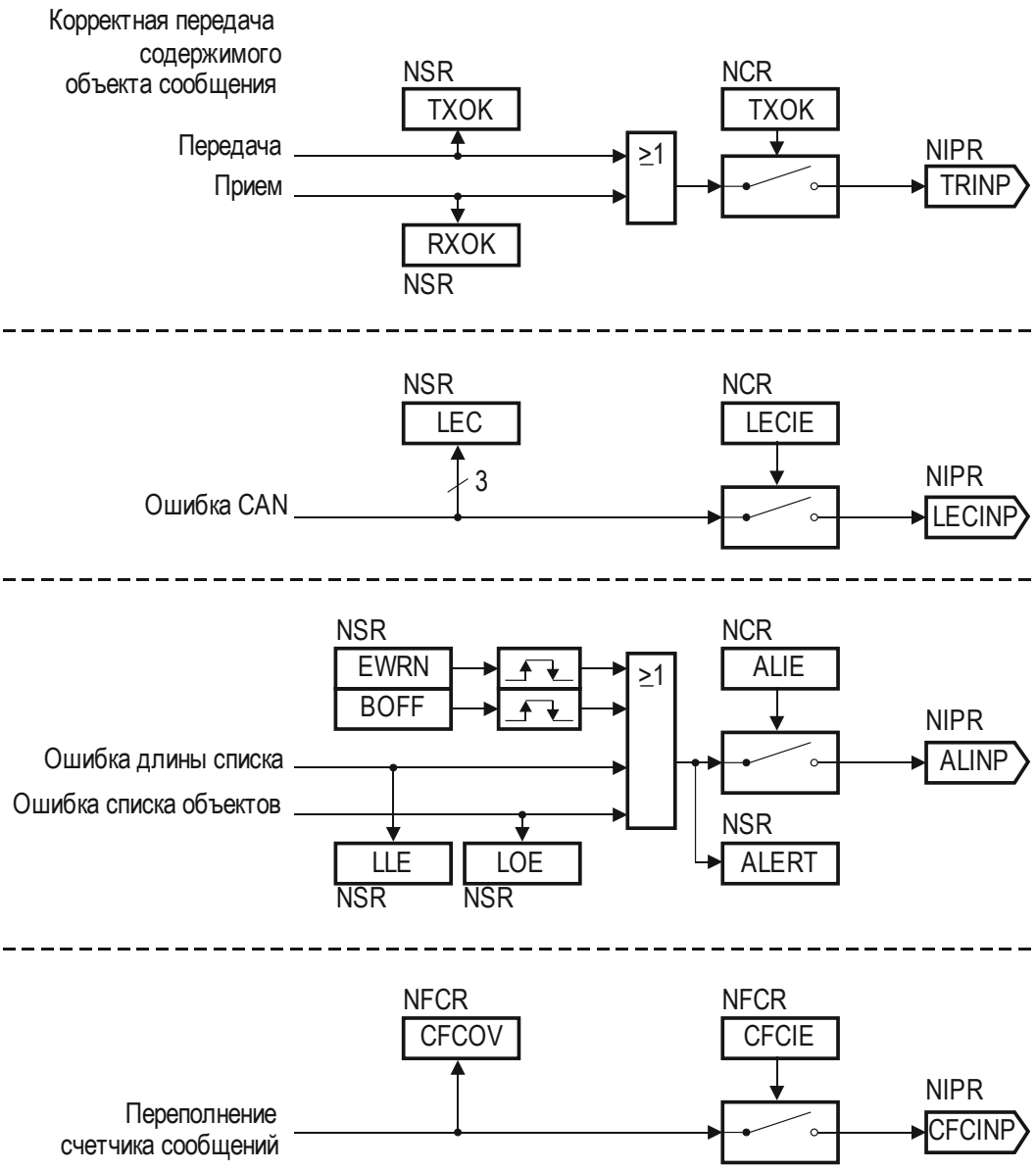


Рисунок 18.15 – Прерывания CAN узла

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 16). Установка нескольких битов приводит к параллельному генерированию запросов прерываний на соответствующих установленным битам линиях прерываний.

18.4 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 18.16, где для примера взят пятый объект сообщения.

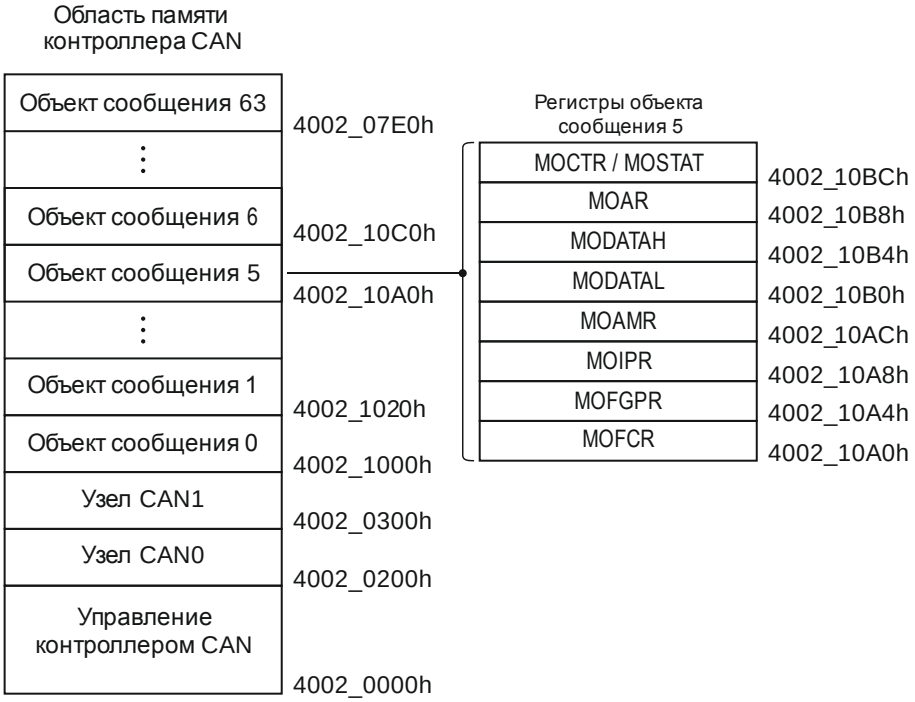


Рисунок 18.16 – Структура памяти регистров

Объекты сообщений контроллера CAN могут быть организованы в восемь списков, см. рисунок 18.17.

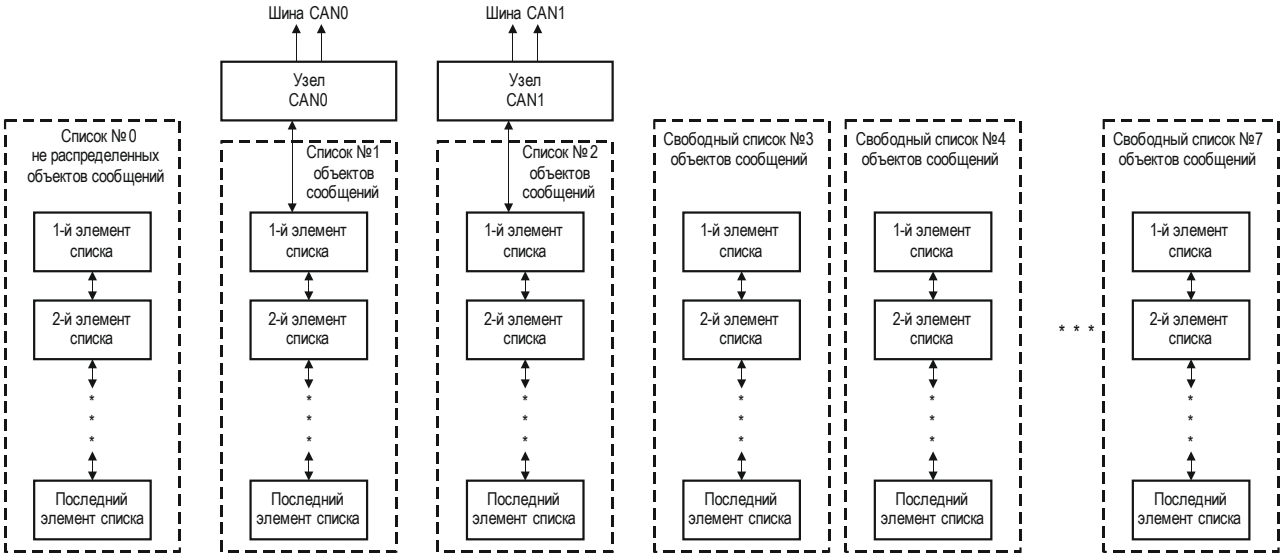


Рисунок 18.17 – Списки контроллера CAN

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 18.18 представлен вариант, когда объекты сообщений с номерами 3, 5 и 16 занесены в список №2, принадлежащий узлу CAN1. Состояние списка отражено в регистре LIST2.

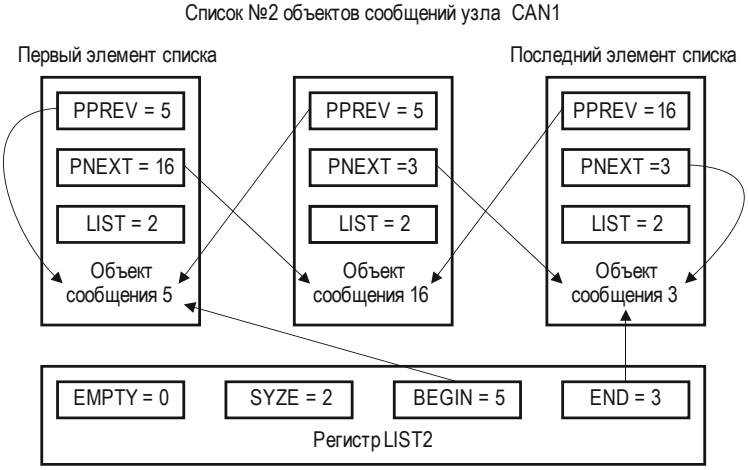


Рисунок 18.18 – Пример списка объектов сообщений

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 18.18 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).

18.5 Прием и передача сообщений

Прием сообщения

После завершения приема сообщение сохраняется в объекте сообщения в соответствии с установленным алгоритмом, см. рисунок 18.19.

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.

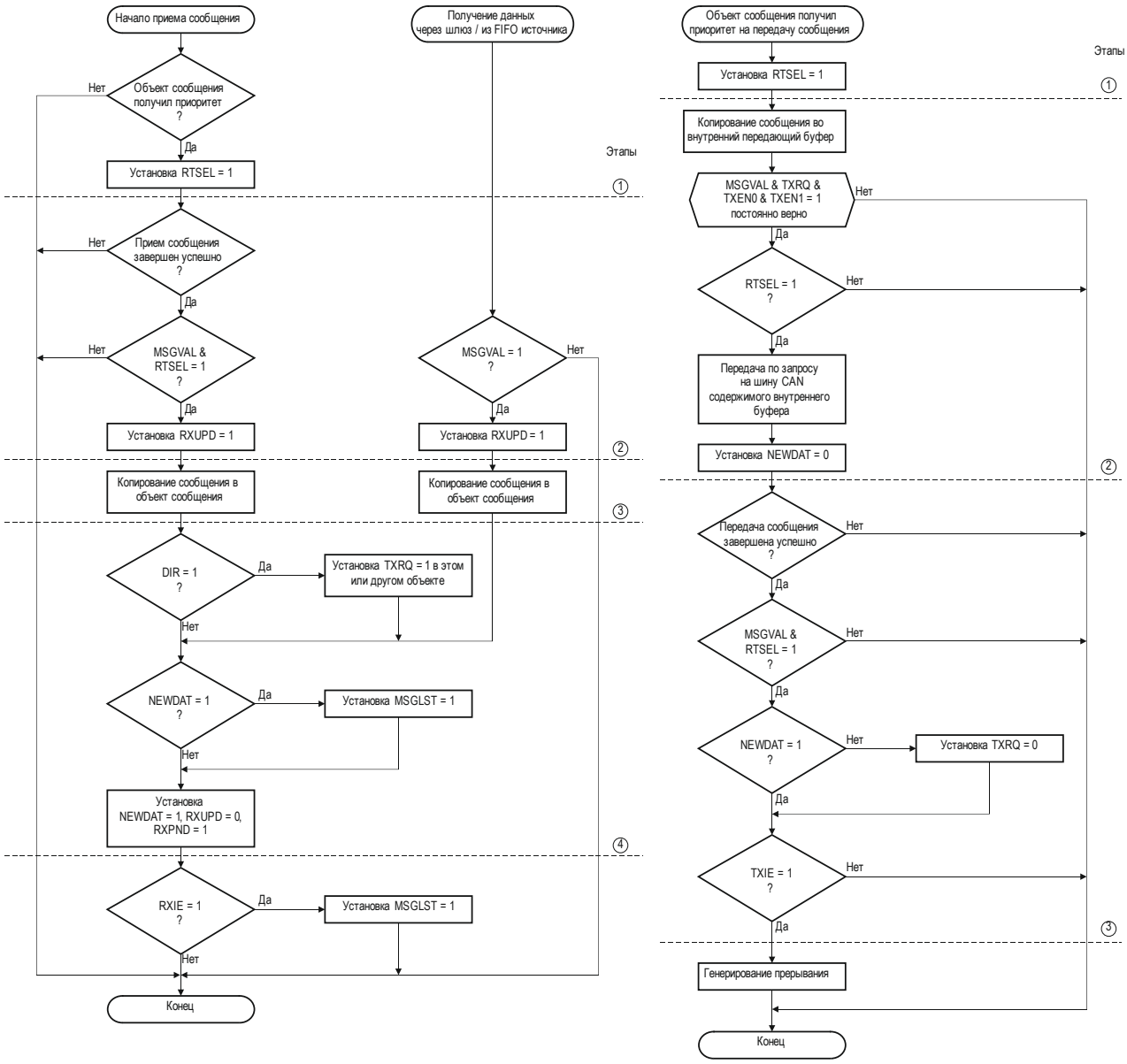


Рисунок 18.19 – Алгоритмы приема и передачи сообщения

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 18.19. 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.

18.6 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 18.20 показан пример проверки идентификатора.

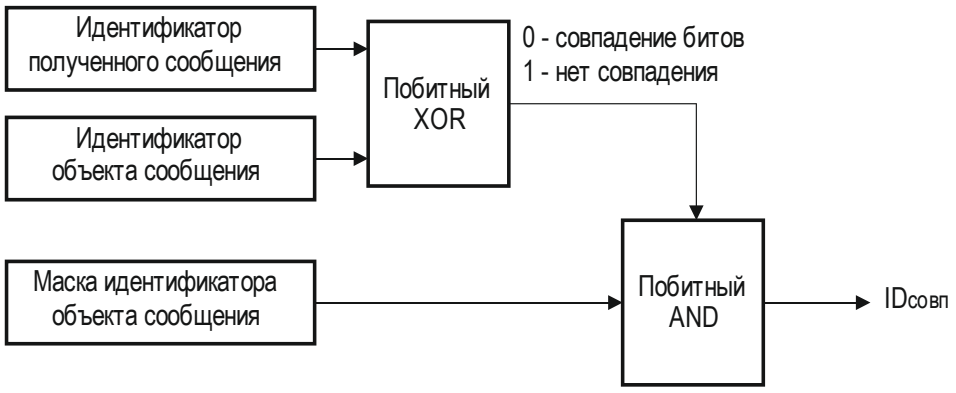


Рисунок 18.20 – Проверка идентификатора полученного сообщения

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 18.21 показано формирование запроса на передачу объекта сообщения.

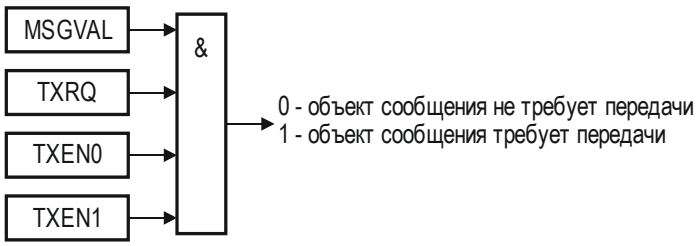


Рисунок 18.21 – Формирование запроса на передачу объекта сообщения

18.7 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 отвечать на запрос будет сам), 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).

18.8 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 повторений

Режим передачи данных с защитой от повторений 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.

18.9 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 18.22 представлена основная FIFO структура. Она состоит из одного базового объекта и n-ого числа вспомогательных объектов.

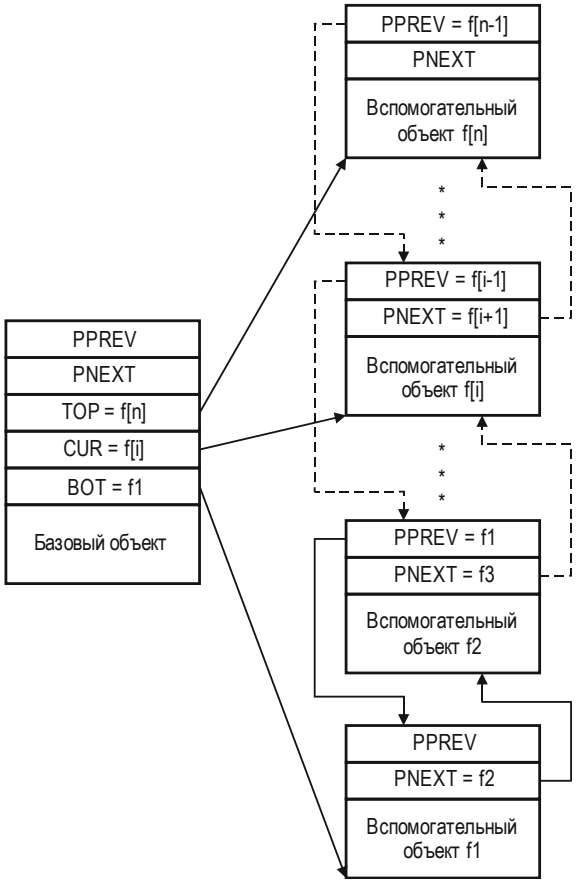


Рисунок 18.22 – FIFO структура с базовым объектом и n вспомогательными объектами

Вспомогательные объекты передающей FIFO структуры объединяются последовательно в списки (подобно спискам объектов сообщений). Базовый объект может

быть занесен в любой список. Хотя на рисунке 18.22 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, 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 64 объекта 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 для 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.

18.10 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 18.23, 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.

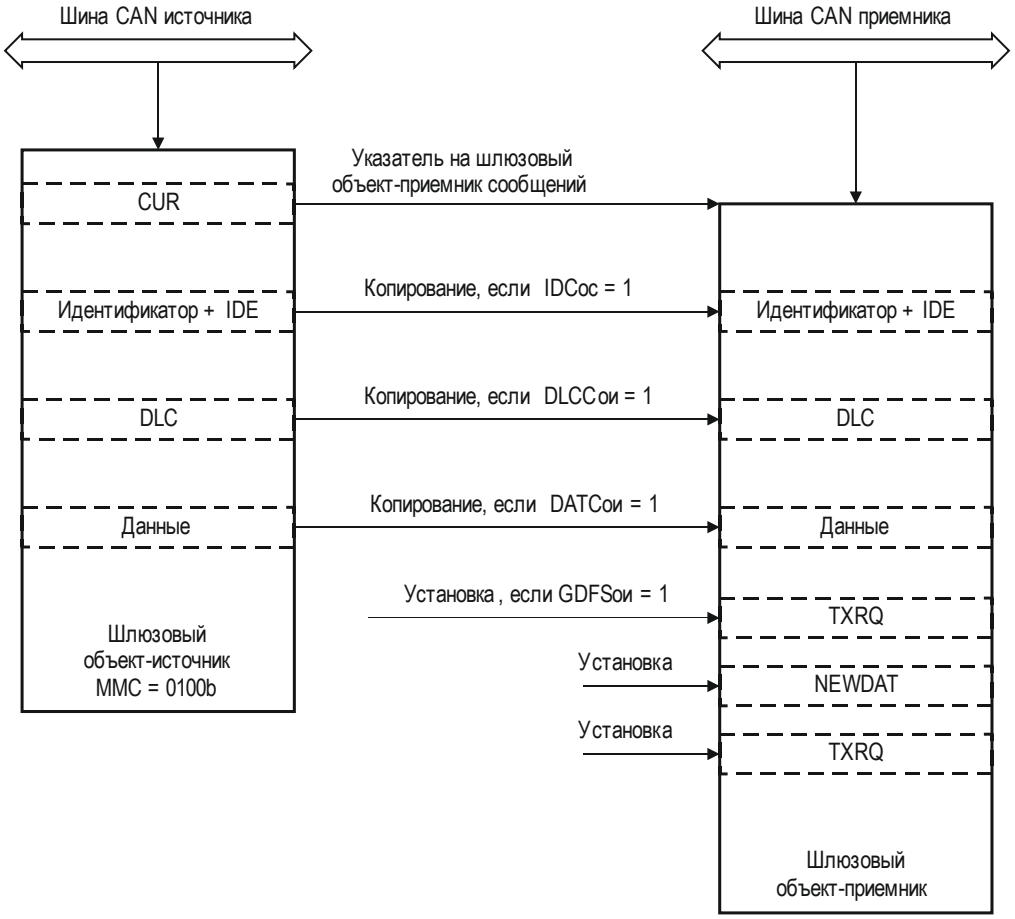


Рисунок 18.23 – Передача через шлюз от источника к приемнику

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.

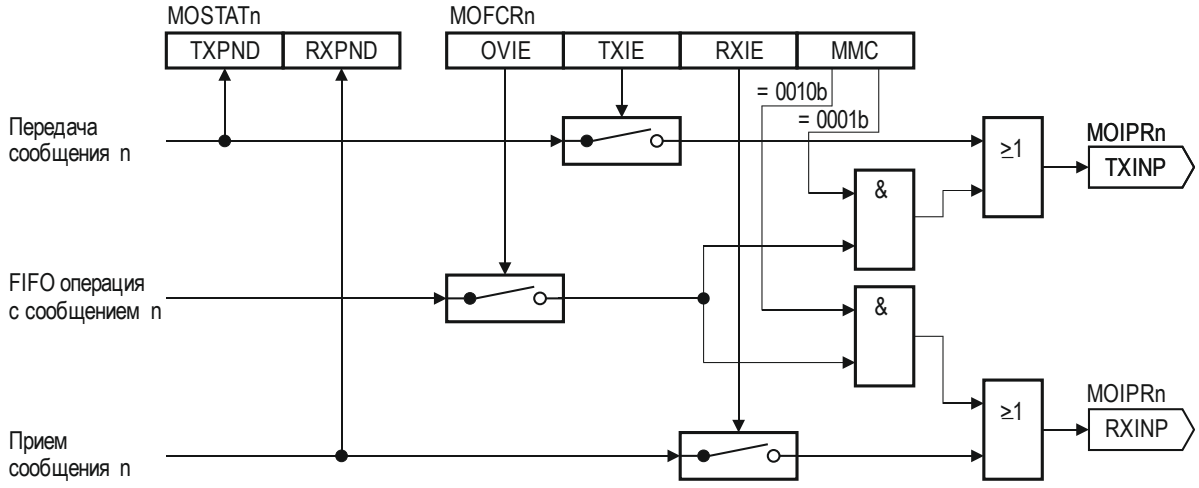
После успешного приема/передачи сообщения ЦП получает уведомление о завершении операции для задания дальнейших действий, связанных с объектом сообщения.

18.11 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 18.24:

- 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 18.24 – Распределение прерываний

Ждущие сообщения

Когда генерируется запрос на прерывание (после приема/передачи сообщения), в одном из четырех регистров ждущих прерываний MSPNDx (x от 0 до 3) выставляется флаг ждущего сообщения. Четыре регистра образуют область из 32 × 4 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 18.25 и 18.26.

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:

- 6 и 5 биты поля MPN выбирают регистр MSPNDx, в котором будет установлен флаг седьмого ждущего сообщения;

- пять младших бит поля MPN (на рисунке 18.25 выделены серым цветом) выбирают позицию флага (от 0 до 31), который будет установлен в выбранном регистре MSPNDx.

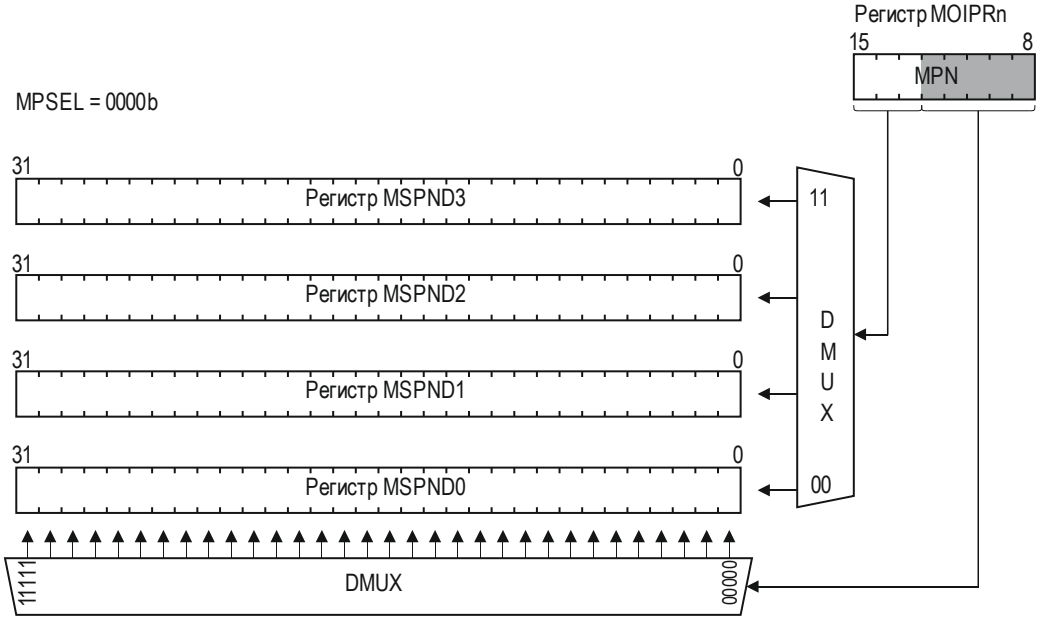


Рисунок 18.25 – Режим выбора и установки флагов при MPSEL = 0h

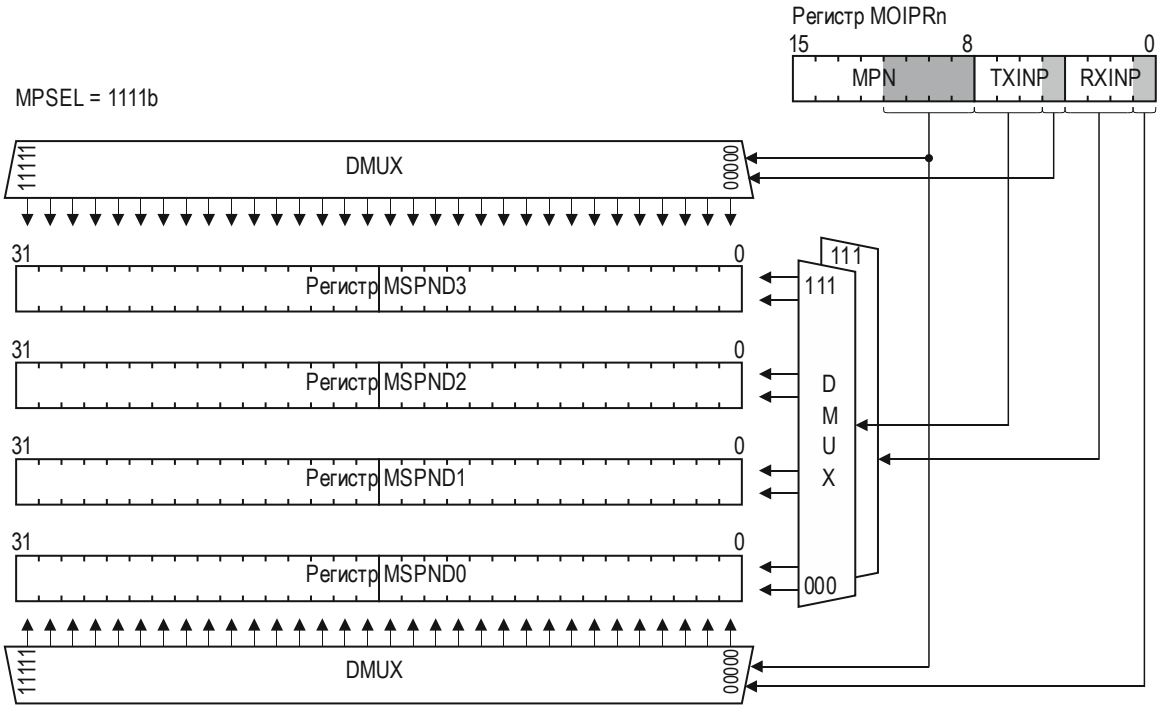


Рисунок 18.26 – Режим выбора и установки флагов при MPSEL = Fh

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:

- 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 18.26 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.

18.12 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 = 00000000h и после этого записать регистр FDR.

Далее для конфигурирования узла CAN с номером x (x от 0 до 3) 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.

19 Блок АЦП

Блок АЦП объединяет один модуль АЦП последовательного приближения (архитектура SAR), схему управления, буферы результатов измерений и схему управления прерываниями. Структурная схема блока АЦП показана на рисунке 19.1.

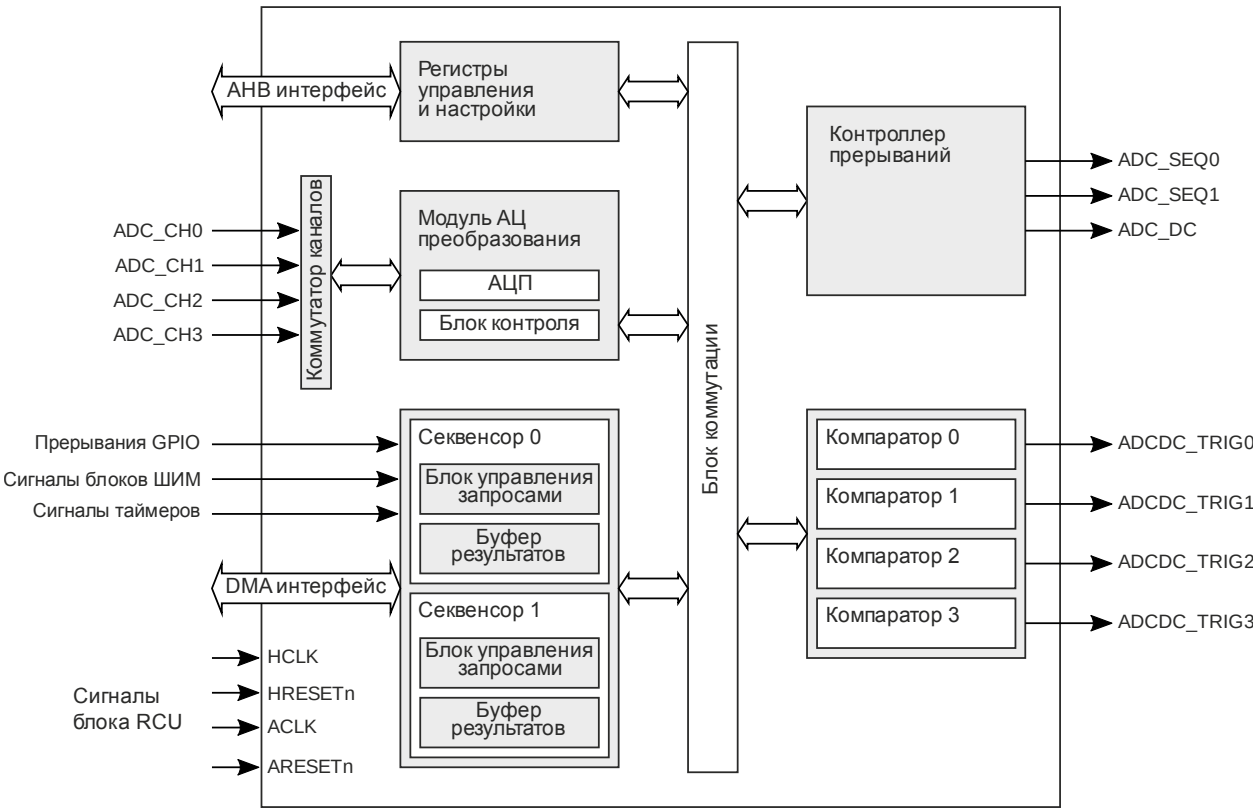


Рисунок 19.1 – Структурная схема блока АЦП

В блок АЦП входят:

- четырехканальный модуль АЦП разрядностью 12 бит и скоростью измерения по одному каналу до 1 миллиона измерений в секунду;
- два секвенсора, каждый из которых позволяет независимо произвести запуск измерений по необходимым каналам АЦП и сгенерировать прерывание;
- четыре независимых цифровых компаратора, 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 имеет четыре входных канала. Диапазон измеряемых напряжений ограничен ∩VCC2 и ∩GND2.

Настройка тактирования и сброса блока АЦП и его модулей осуществляется посредством регистра ADCCFG блока управления тактовыми сигналами RCU. Вся внутренняя логика блока АЦП тактируется частотой ACLK, но запись/чтение контрольно-статусных регистров осуществляется на частоте HCLK.

Для правильной работы блока необходимо обеспечить тактирование модулей АЦП частотой ACLK от 300 кГц до 32 МГц, которую можно получить, выбрав источник тактового сигнала полем CLKSEL, а также, при необходимости, включив и настроив делитель полями DIVEN и DIVN (поля регистра ADCCFG блока RCU). Частота ACLK не должна быть больше частоты системного тактового сигнала SYSCLK (на тактовый вход HCLK блока АЦП подается системный тактовый сигнал).

19.1 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 s секвенсоров (s = 0, 1). Структурная схема секвенсора представлена на рисунке 19.2.

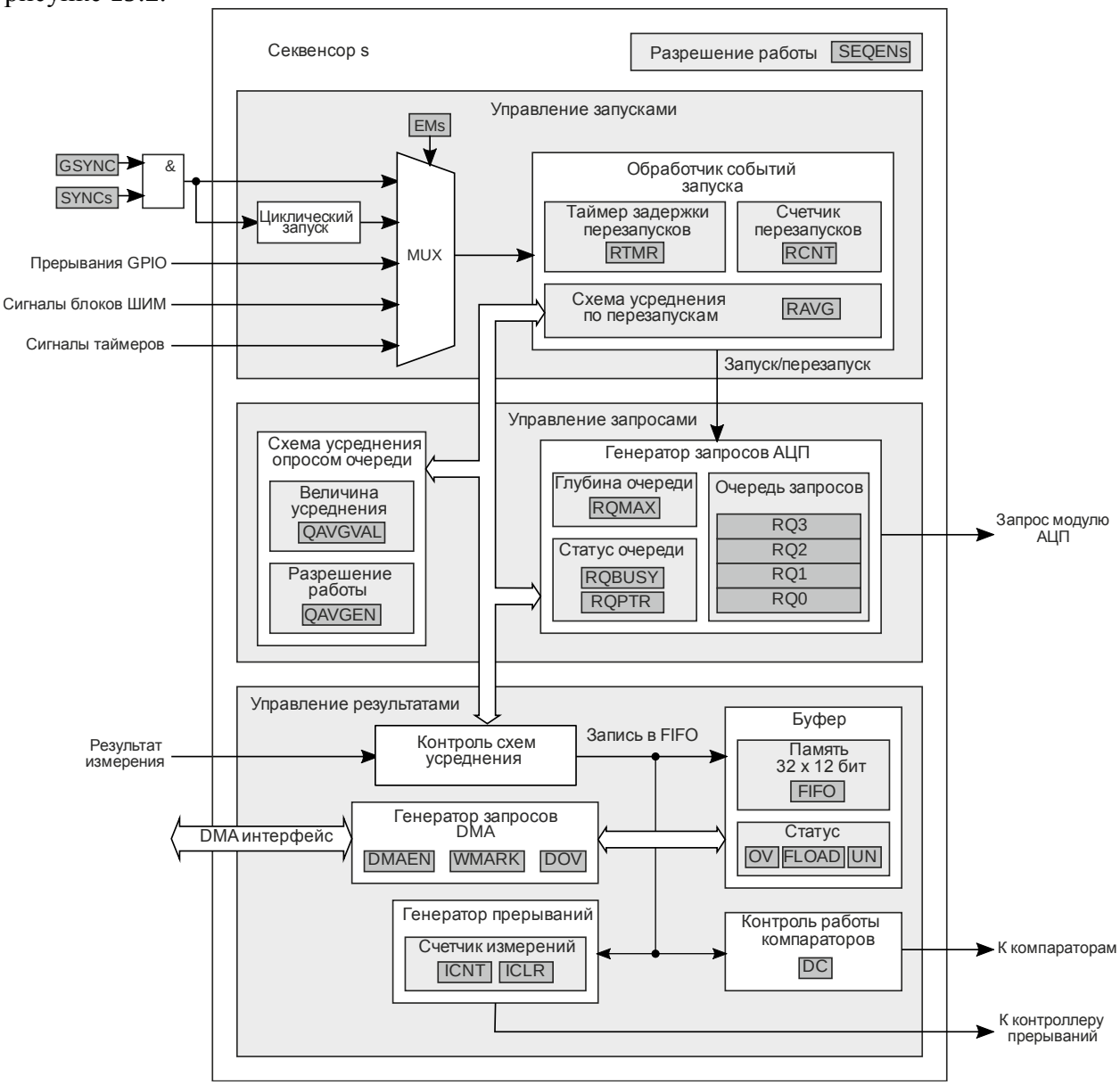


Рисунок 19.2 – Структурная схема секвенсора

Одиночные запуски по событиям

Разрешение работы секвенсора осуществляется установкой соответствующего бита SEQENs (s = 0, 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;
- сигналы от блоков ШИМ;
- сигнал прерывания 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 19.3.

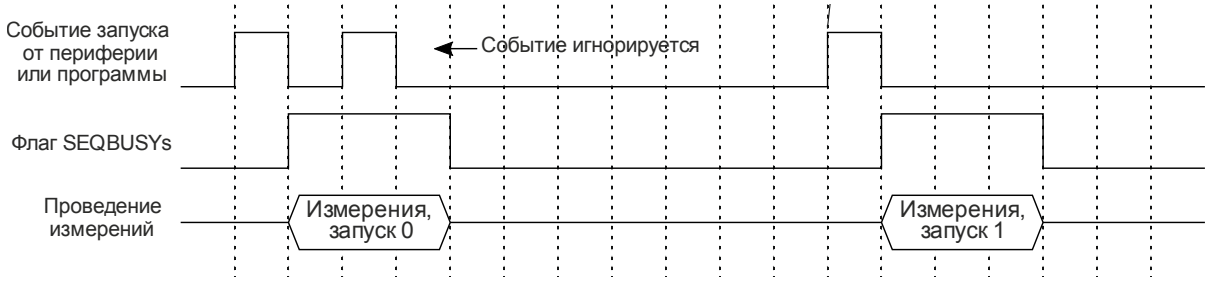


Рисунок 19.3 – Одиночные запуски секвенсора по событиям

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 19.4.

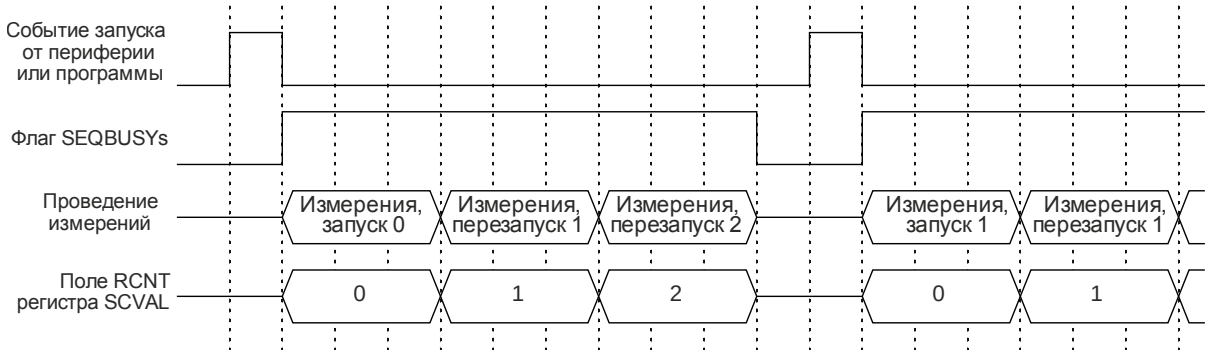


Рисунок 19.4. – Одиночные запуски по событиям с немедленными перезапусками при RCNT = 2 (регистр SCCTL)

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 19.5.

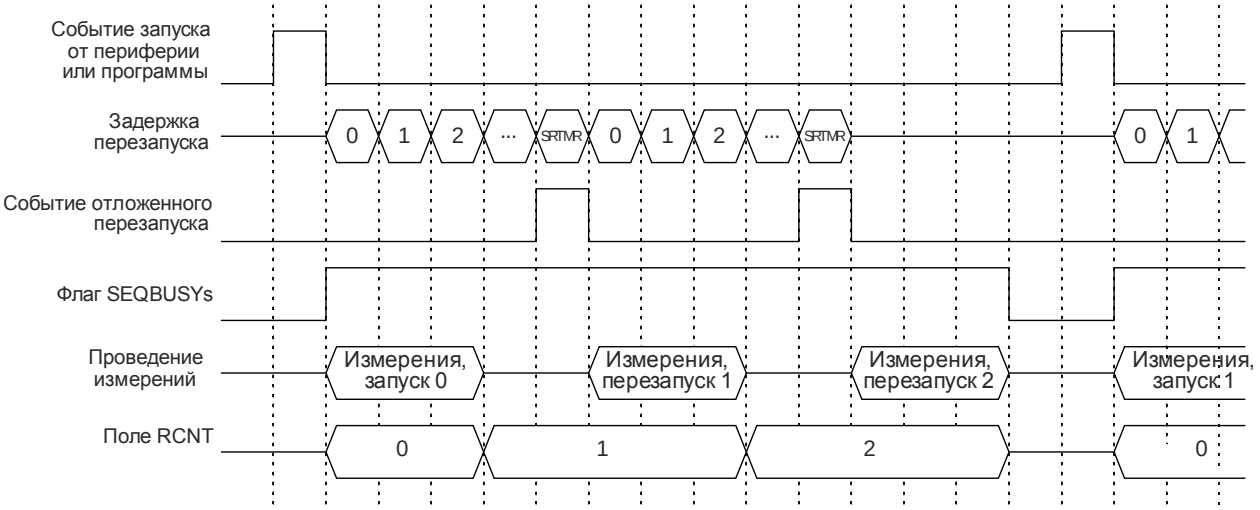


Рисунок 19.5. – Одиночные запуски по событиям с отложенными перезапусками через время SRTMR при RCNT = 2 (регистр SCCTL)

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 = 1, 2, ..., 8. Значение 2^p 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 19.6. При отложенных перезапусках данный вид усреднения работает аналогично, позволяя распределить равномерно измерения на длительном промежутке времени и получить среднее значение сигнала в конце него.

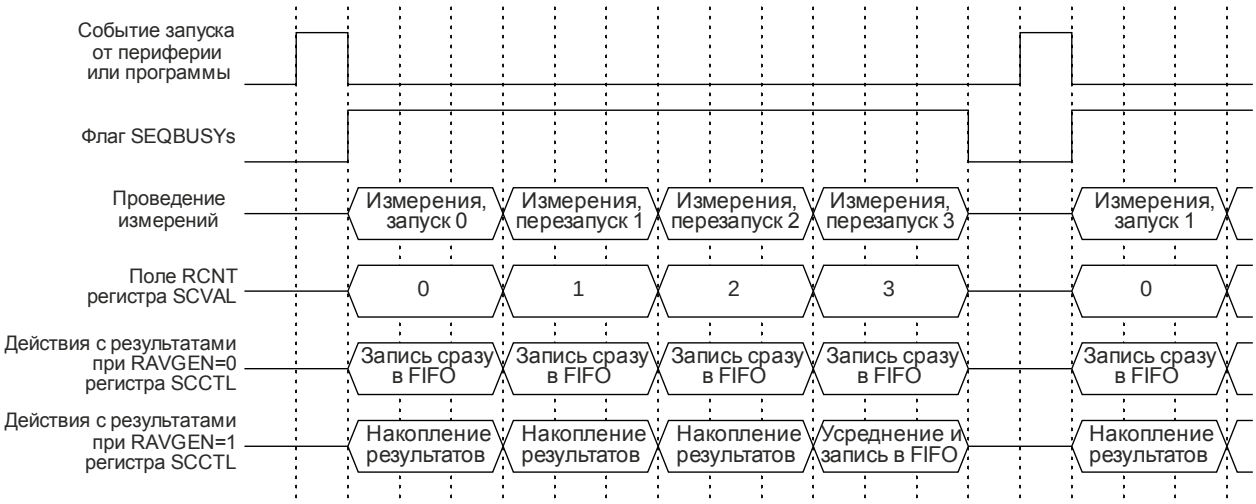


Рисунок 19.6. – Одиночные запуски по событиям усреднением по перезапускам при RCNT = 3, RAVGEN = 1 (регистр SCCTL)

Например, если измерения проводились по всем 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 19.7.

Рисунок 19.7 – 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 SRTMR). Диаграммы работы секвенсора при циклическом отложенном запуске показаны на рисунке 19.8.

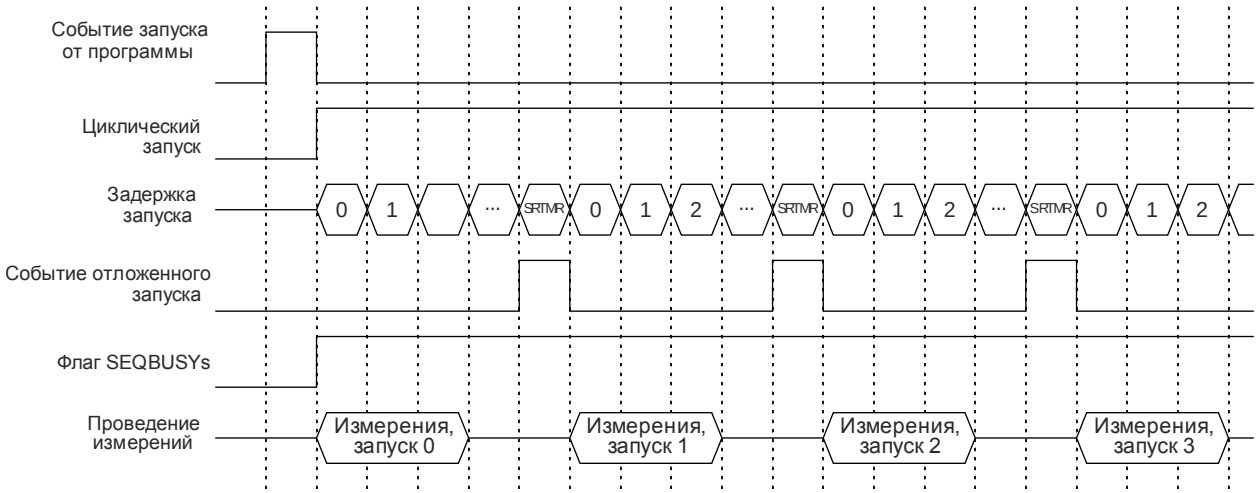


Рисунок 19.8 – Диаграммы работы секвенсора при циклическом отложенном запуске через определённое время (значение регистра SRTMR)

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 19.9.

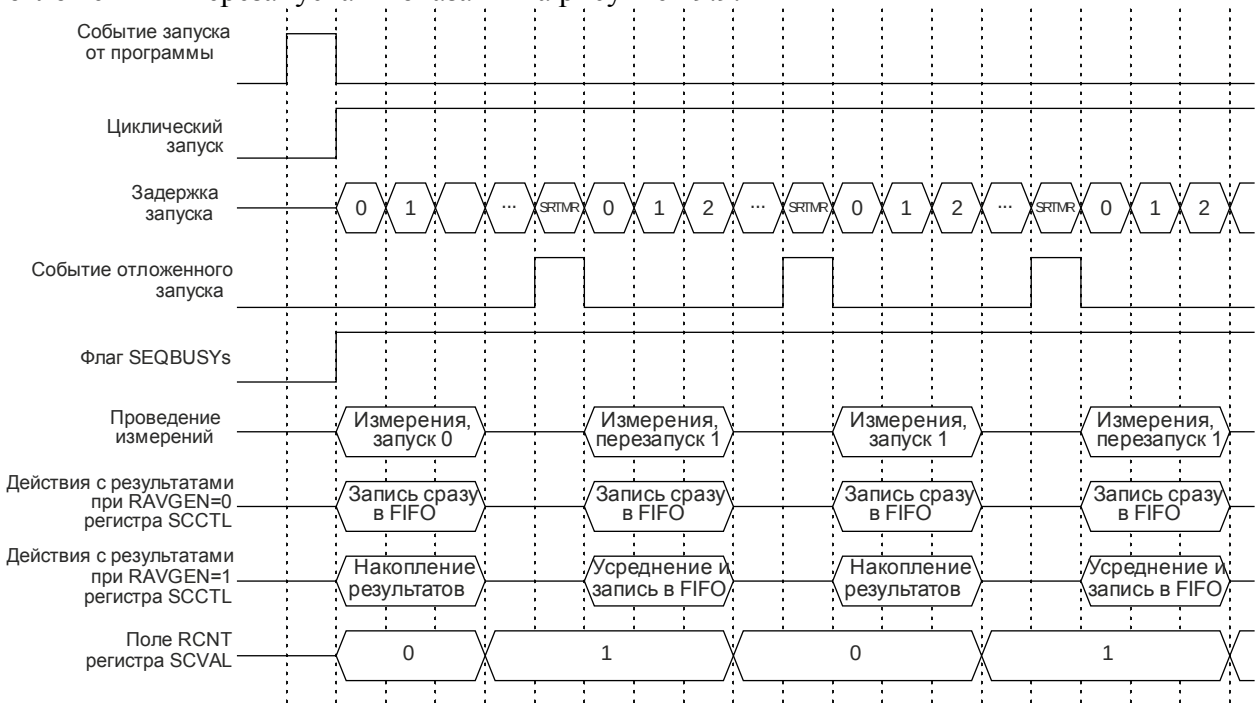


Рисунок 19.9 – Диаграммы работы секвенсора в циклическом режиме запуска с усреднением по отложенным перезапускам определённое время (значение регистра SRTMR) при RCNT=1 (регистр SCCTL)

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 19.10.

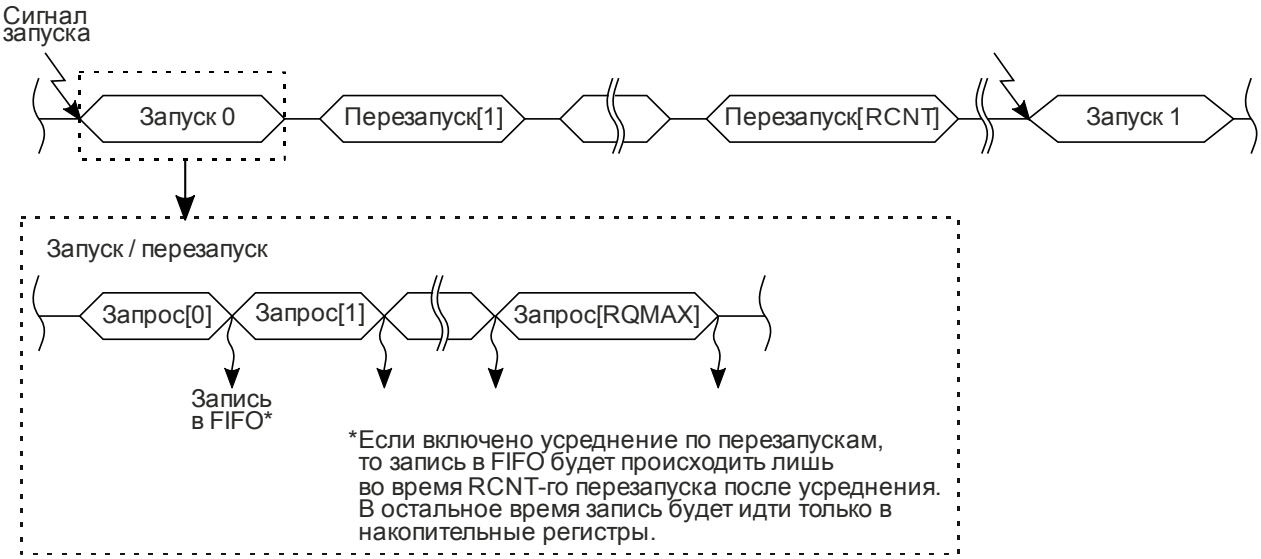


Рисунок 19.10 – Генерация секвенсором запросов на измерение

Пояснения к рисунку 19.10

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 (пункт 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 измерений. 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, см. рисунок 19.10, 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 режима усреднения сканированием показаны на рисунке 19.11.

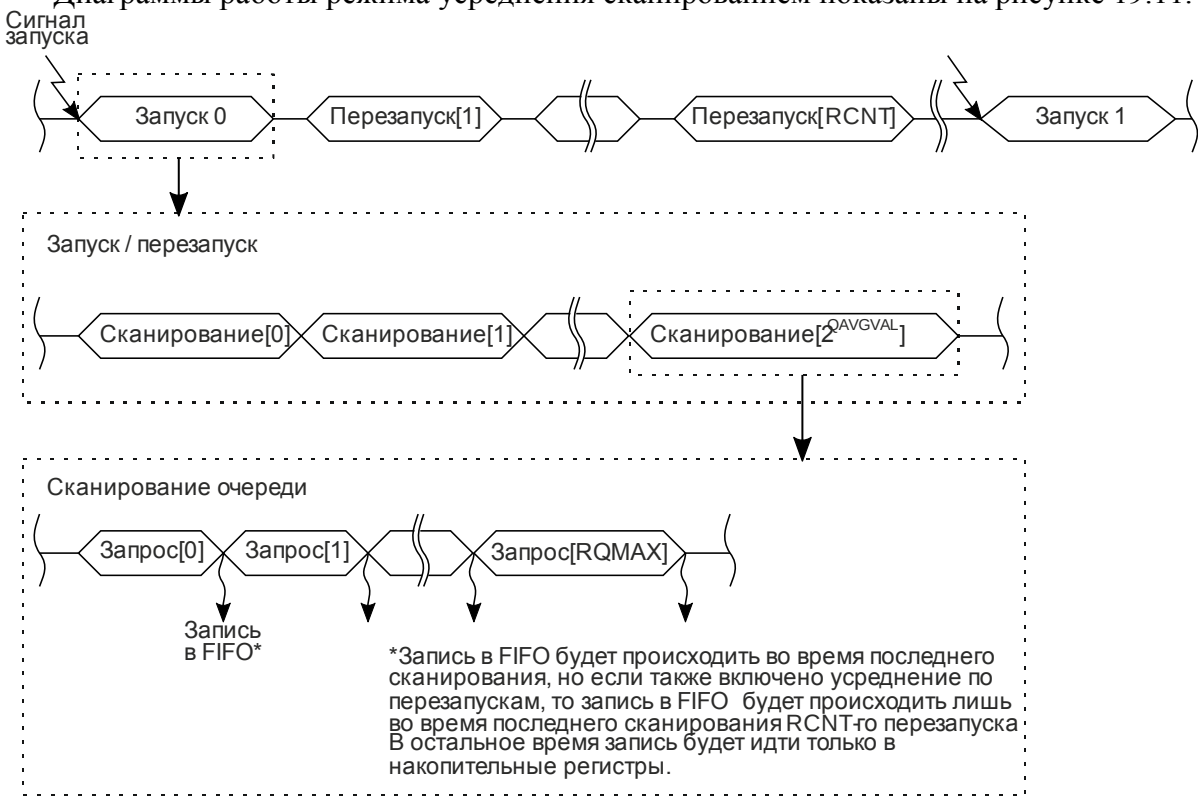


Рисунок 19.11 – Диаграммы работы режима усреднения сканированием

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 будут и 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_SEQ0 и ADC_SEQ1 соответственно.

Использование прямого доступа к памяти

Для разрешения использования контроллера DMA секвенсором, необходимо установить бит DMAEN в регистре SDMACTL. Поле WMARK того же регистра задает уровень заполнения буфера секвенсора, по достижении которого будет запущен контроллер DMA. Перенос данных будет выполняться, пока не будет передано число результатов измерений, соответствующее состоянию поля WMARK.

Если очередной запрос на запуск контроллера DMA пришел раньше, чем закончился предыдущий цикл DMA от того же секвенсора, то будет выставлен флаг ошибки DOVs в регистре FSTAT.

19.2 Модуль АЦП

Структурная схема четырехканального модуля АЦП показана на рисунке 19.12.

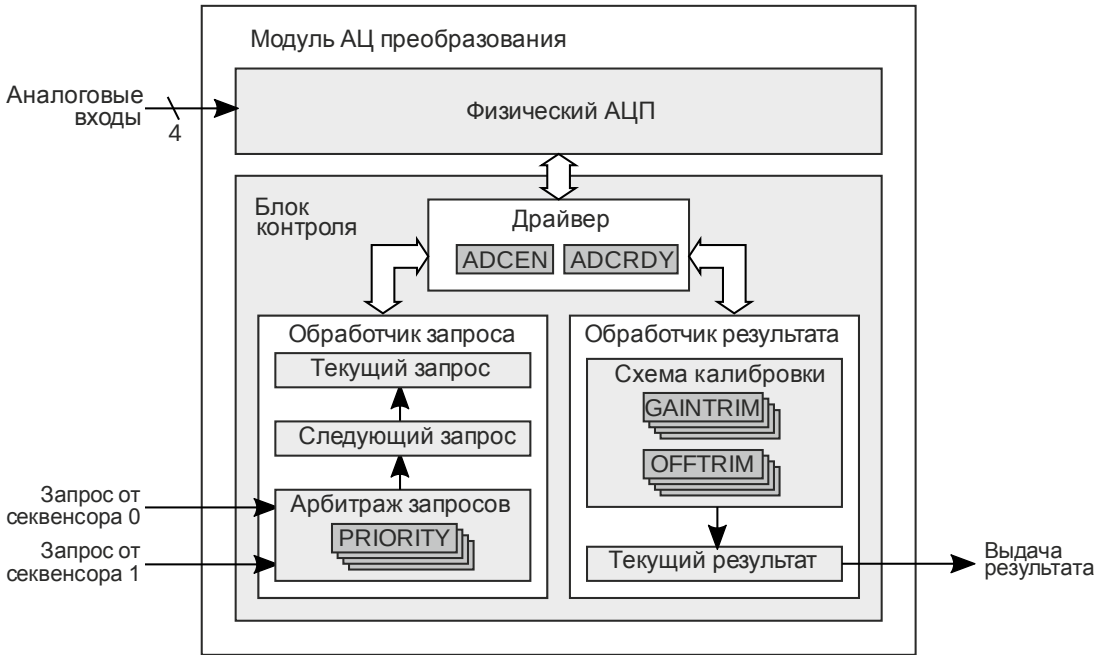


Рисунок 19.12 – Структурная схема четырехканального модуля АЦП

Разрешение работы модуля АЦП осуществляется установкой бита ADCEN в регистре ACTL. При каждом переключении ADCEN в единицу запускается процедура инициализации АЦП, которая завершается установкой бита ADCRDY регистра ACTL. АЦП готов к работе, когда ADCRDY = 1. Для правильной работы блока, необходимо обеспечить тактирование модулей АЦП частотой ACLK от 300 кГц до 32 МГц. Время преобразования одного канала равняется 14 тактам частоты ACLK.

Правила арбитража

Когда модуль 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, n = 0, 1, 2, 3) 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 19.13, где серым выделены запросы по каналам с установленным битом PRIORITY.

.

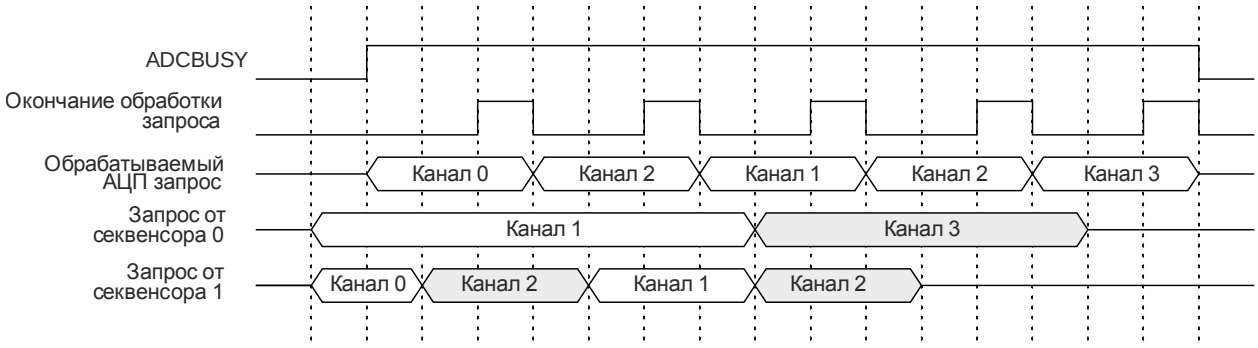


Рисунок 19.13 – Пример схемы арбитража запросов

Коррекция результатов измерений

Результат преобразования передается на схему коррекции, которая нивелирует ошибку усиления и смещения нуля и работа которой описывается формулой:

$$D_C = \frac{D_R \times (4096 + GAINTRIM)}{4096} + OFFTRIM, \quad (19.1)$$

где D_R – данные, полученные непосредственно с модуля АЦП;

D_C – скорректированные данные, выдаваемые секвенсору;

GAINTRIM – коэффициент корректировки усиления, имеет диапазон (–256, …, 255);

OFFTRIM – коэффициент корректировки смещения нуля, имеет диапазон (–256, …, 255).

Значения GAINTRIM и OFFTRIM заносятся в одноименные поля регистров CHCTLn (где n – номер канала от 0 до 3) 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 4095, то результат равен 4095.

19.3 Цифровой компаратор

В состав блока АЦП входят d компараторов (d = 0, 1, 2, 3). Структурная схема компаратора показана на рисунке 19.14.

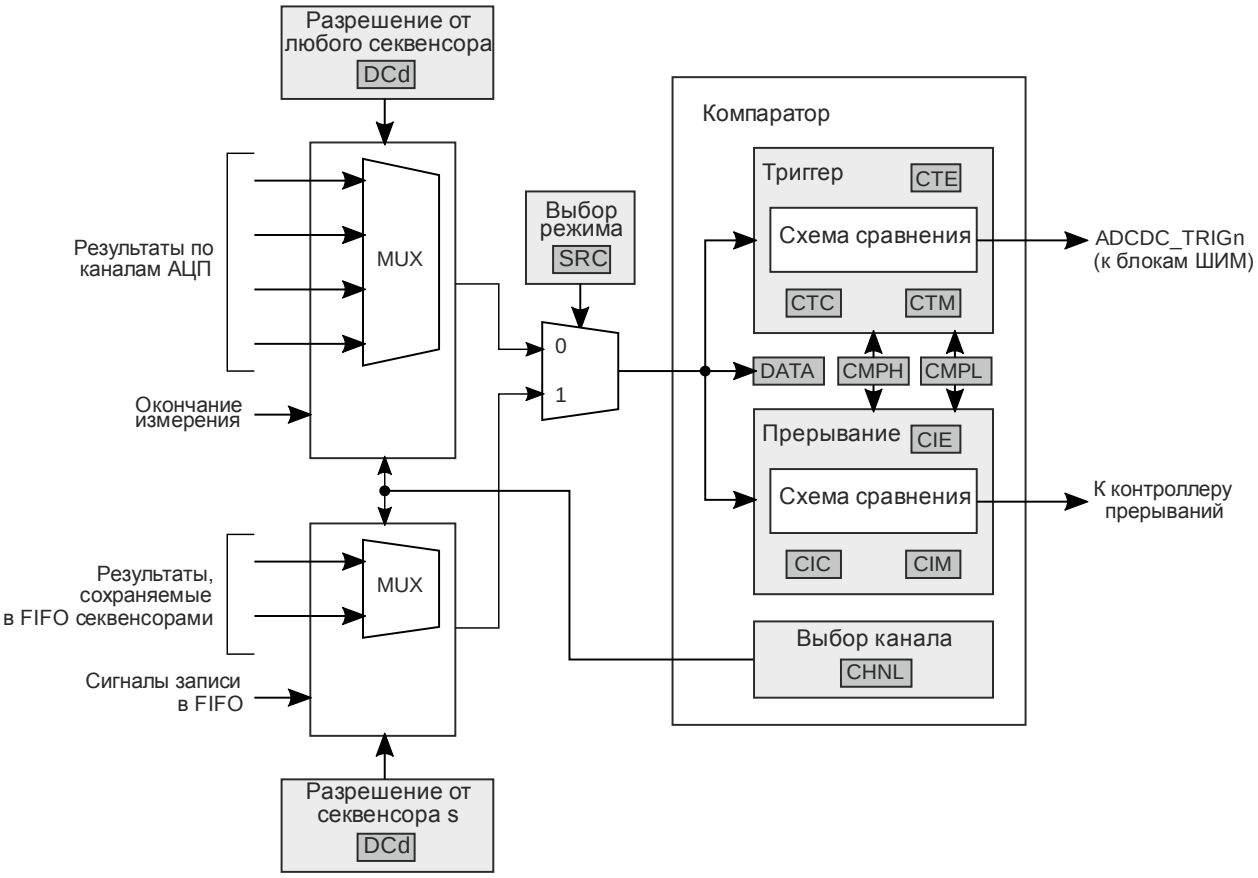


Рисунок 19.14 – Структурная схема компаратора

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):

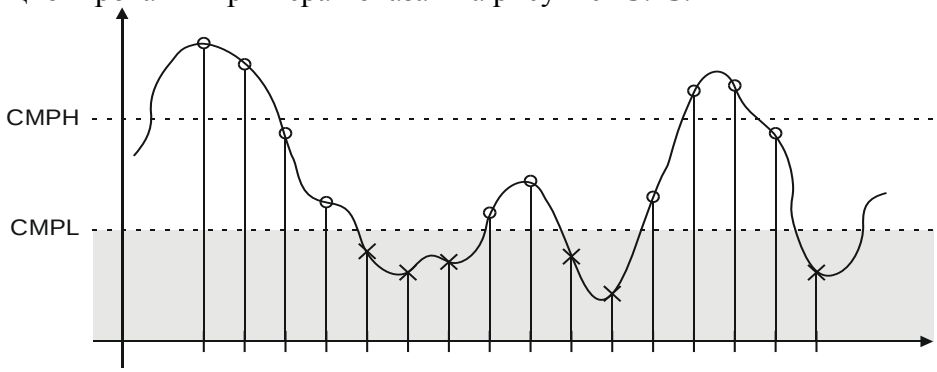
- В однократном режиме (CTM = 01b) выходной триггер переключится в единицу только в случае, если результат сравнения окажется положительным при том, что результат предыдущего сравнения был отрицательным. В остальных случаях состояние триггера – ноль.

- 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 19.15.



Многократный -	0	0	0	0	1	1	1	0	0	1	1	0	0	0	0	1
Однократный -	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1
Многократный с гистерезисом -	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	1
Однократный с гистерезисом -	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1

Рисунок 19.15 – Функционирование триггера при CTC = 00b

Сравнение по условию «CMPL ≤ Измерение ≤ CMPH» (CTC = 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 19.16.

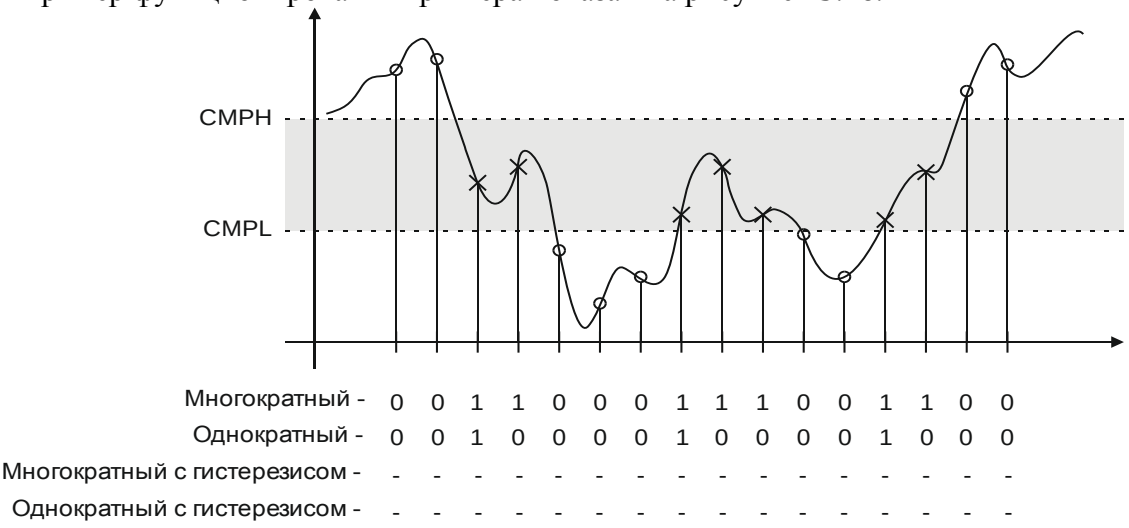


Рисунок 19.16 – Функционирование триггера при CTC = 01b

Сравнение по условию «CMPH ≤ Измерение» (CTC = 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 19.17.

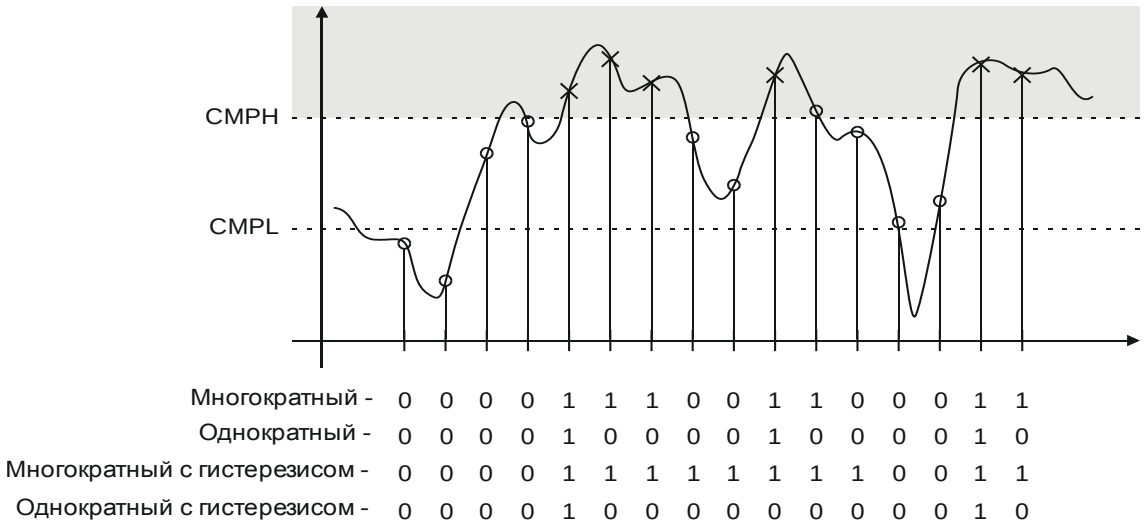


Рисунок 19.17 – Функционирование триггера при CTC = 10b

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.

19.4 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 19.18.

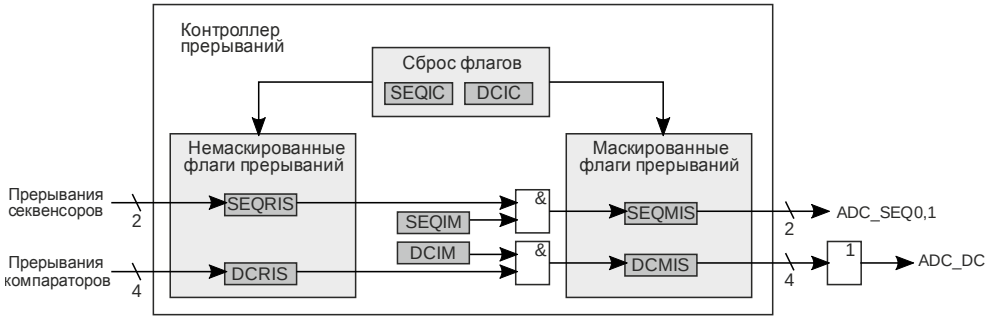


Рисунок 19.18 – Структурная схема контроллера прерываний

19.5 Примеры работы блока АЦП

В этом подразделе представлены разнообразные примеры настройки блока АЦП для осуществления измерений в различных режимах. Для всех примеров подразумевается, что блок АЦП тактируется частотой 25 МГц.

Настройка тактирования

Один из примеров настройки тактирования блока АЦП совместно с настройкой системного тактового сигнала представлен ниже.

1 С помощью регистра PLLCFG блока RCU настраиваем выходную частоту PLL 100 МГц. Осуществляем процедуру перевода системной частоты на PLL. Таким образом, частота системного тактового сигнала SYSCLK будет равна 100 МГц.

2 Настройка рабочей частоты блока АЦП ACLK производится с помощью регистра ADCCFG блока RCU. Выбираем в качестве источника выходную частоту PLL (поле CLKSEL = 1), включаем делитель на 4 (поле DIVN = 1, бит DIVEN = 1). Таким образом, ACLK = 25 МГц.

```
RCU->ADCCFG_bit.CLKSEL = 1;
RCU->ADCCFG_bit.DIVN = 1;
RCU->ADCCFG_bit.DIVEN = 1;
```

3 Включаем тактирование блока (бит CLKEN = 1) и снимаем сброс (бит RSTDIS = 1). Блок АЦП готов к дальнейшим конфигурациям.

```
RCU->ADCCFG_bit.CLKEN = 1;
RCU->ADCCFG_bit.RSTDIS = 1;
```

Пример 1 – Программный запуск одного секвенсора

Требуемый режим работы:

- программный запуск;
- секвенсор 0;
- однократное измерение всех четырех каналов;
- без прерываний (опрос флагов).

Код, соответствующий настройке и запуску необходимого режима, представлен ниже.

```
// Настройка
    ADC->ACTL_bit.ADCEN = 1;
ADC->EMUX_bit.EM0 = 0;
ADC->SEQ[0].SRQCTL_bit.RQMAX = 3;
ADC->SEQ[0].SRQSEL_bit.RQ0 = 0;
ADC->SEQ[0].SRQSEL_bit.RQ1 = 1;
ADC->SEQ[0].SRQSEL_bit.RQ2 = 2;
ADC->SEQ[0].SRQSEL_bit.RQ3 = 3;
ADC->SEQEN_bit.SEQEN0 = 1;
// Запуск
while(!ADC->ACTL_bit.ADCRDY);
ADC->SEQSYNC_bit.SYNC0 = 1;
ADC->SEQSYNC_bit.GSYNC = 1;
```

1 Разрешаем работу модуля АЦП – необходимо установить бит ADCEN в регистре ACTL.

2 Для работы выберем 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 всем четырем каналам.

4 Настраиваем каналы для запросов. Допустим, необходимо опросить каналы последовательно от нулевого к третьему, значит, в регистр SRQSEL 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 19.19.

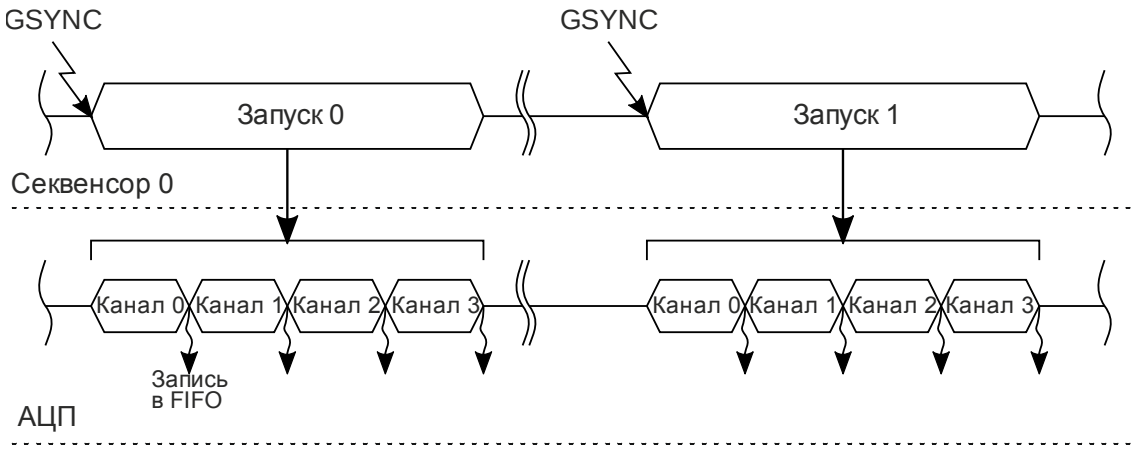


Рисунок 19.19 – Диаграммы программного запуска одного секвенсора

Пример 2 – Циклический опрос канала с задержками

Требуемый режим работы:

- циклическая работа;
- секвенсор 0;
- опрос канала номер 2 каждую 1 мс;
- 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
ADC->ACTL_bit.ADCEN = 1;
ADC->CHCTL[2].CHCTL_bit.GAINTRIM = 5; // значения для примера
ADC->CHCTL[2].CHCTL_bit.OFFTRIM = (uint32_t)(-5);
ADC->EMUX_bit.EM0 = 0xF;
ADC->SEQ[0].SCCTL_bit.ICNT = 0;
ADC->SEQ[0].SRTMR = 24999;
ADC->SEQ[0].SRQCTL_bit.RQMAX = 0;
ADC->SEQ[0].SRQCTL_bit.QAVGVAL = 4;
ADC->SEQ[0].SRQCTL_bit.QAVGEN = 1;
ADC->SEQ[0].SRQSEL_bit.RQ0 = 2;
ADC->SEQEN_bit.SEQEN0 = 1;
// NVIC прерывание
```

```
ADC->IM_bit.SEQIM0 = 1;
NVIC_EnableIRQ(ADC_SEQ0_IRQn);
// Запуск
while(!ADC->ACTL_bit.ADCRDY);
ADC->SEQSYNC_bit.SYNC0 = 1;
ADC->SEQSYNC_bit.GSYNC = 1;
```

1 Разрешаем работу модуля АЦП – необходимо установить бит ADCEN в регистре ACTL.

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 SRQSEL необходимо внести значение RQ0 = 2h.

6 Включаем усреднение сканированием по 16 опросам очереди. В регистре SRQCTL нужно установить поля QAVGVAL = 4, QAVGEN = 1.

7 Для того чтобы опрос проводился каждую 1 мс (при ACLK = 25 МГц), необходимо внести в регистр SRTMR значение (1000000 нс/40 нс) – 1 = 24999.

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 19.20.

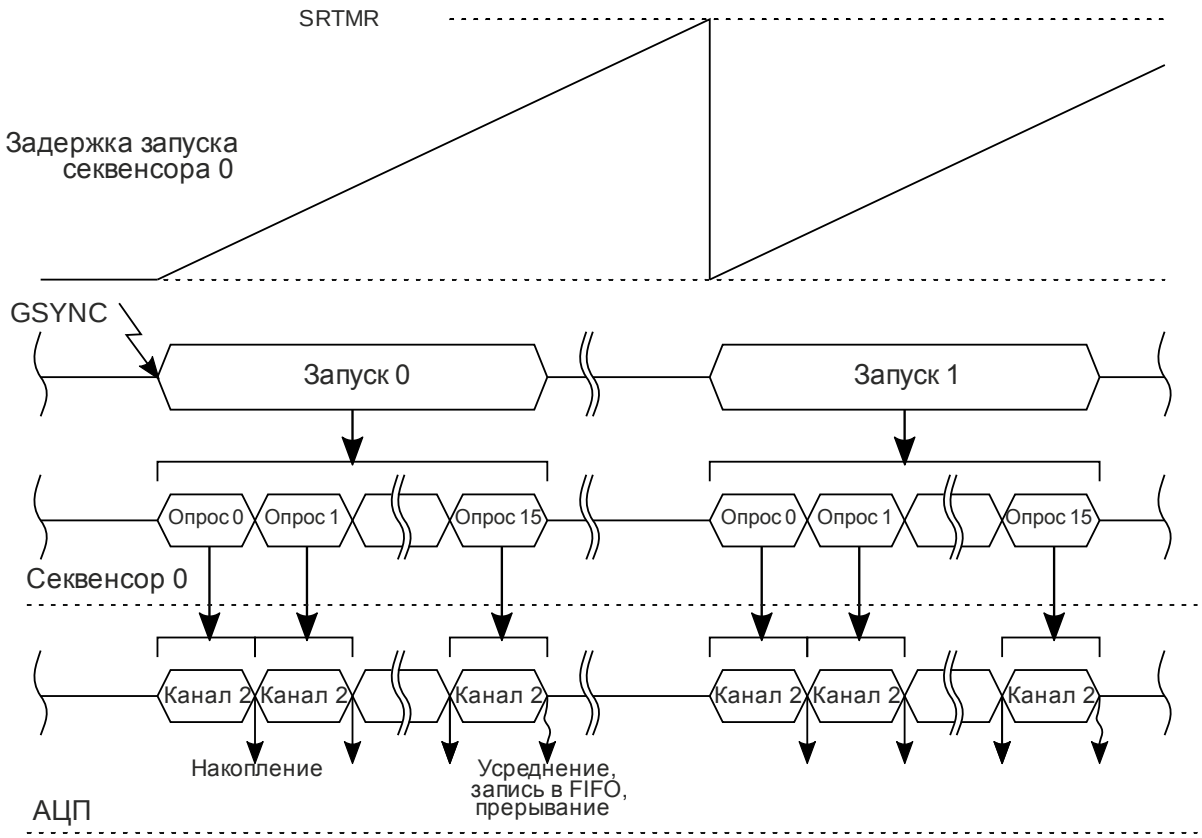


Рисунок 19.20 – Диаграммы циклического опроса канала

Пример 3 – Равномерно распределенные измерения по периоду таймера

Требуемый режим работы:

- запуск по таймеру 0 с частотой 10 кГц;
- 4 точки измерения, равномерно распределенные по периоду;
- секвенсор 0;
- опросы по каналам 3 и 0;
- каждый опрос из 4 последовательных измерений и усреднения;
- прерывание каждые 8 записей в FIFO (по окончании 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 = 9999;
TMR0->ADCSOC_bit.EN = 1;
// Настройка АЦП
ADC->ACTL_bit.ADCEN = 1;
ADC->EMUX_bit.EM0 = 3;
ADC->SEQ[0].SCCTL_bit.ICNT = 7;
ADC->SEQ[0].SCCTL_bit.RCNT = 3;
ADC->SEQ[0].SRTMR = 624;
ADC->SEQ[0].SRQCTL_bit.RQMAX = 1;
ADC->SEQ[0].SRQCTL_bit.QAVGVAL = 2;
ADC->SEQ[0].SRQCTL_bit.QAVGEN = 1;
ADC->SEQ[0].SRQSEL_bit.RQ0 = 3;
ADC->SEQ[0].SRQSEL_bit.RQ1 = 0;
```

```
ADC->SEQEN_bit.SEQEN0 = 1;
// NVIC прерывание
ADC->IM_bit.SEQIM0 = 1;
NVIC_EnableIRQ(ADC_SEQ0_IRQn);
// Запуск
while(!ADC->ACTL_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 SYSCLK = 100 МГц), необходимо внести в регистр LOAD таймера значение (100000 кГц/10 кГц) – 1 = 9999.

3 Разрешаем генерацию таймером запросов на старт преобразования, установив бит EN в регистре ADCSOC таймера.

4 Разрешаем работу модуля АЦП – необходимо установить бит ADCEN в регистре ACTL.

5 Для работы выберем секвенсор 0. Настроим его источник запуска – поле EM0 регистра EMUX должно быть равно 3h, т. к. запуск по опустошению таймера 0.

6 Необходимое количество записей в FIFO для генерации прерывания – 8h, поэтому запишем в поле ICNT регистра SCCTL значение 8 – 1 = 7h.

7 По каждому запуску должно совершиться 3 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

25000 кГц/(10 кГц×4) – 1 = 624.

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 попадает по два усредненных результата, то прерывание будет сгенерировано после записи последнего результата в последнем третьем перезапуске.

Работа режима проиллюстрирована на рисунке 19.21.

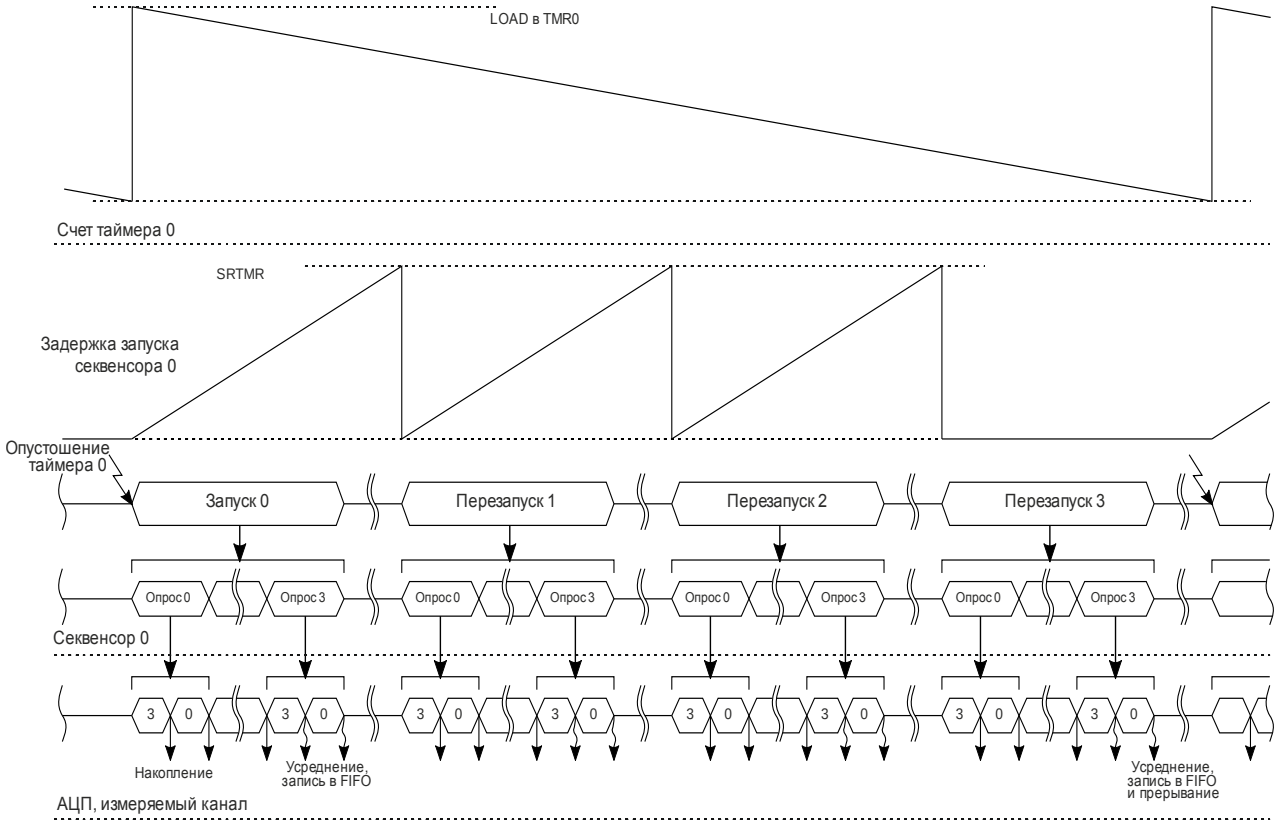


Рисунок 19.21 – Диаграммы равномерно распределенных измерений по периоду таймера

Пример 4 – Запуск нескольких секвенсоров

Требуемый режим работы:

- таймер 0 опустошается с частотой 10 кГц;
- ШИМ0 считает в двунаправленном режиме, достигает нуля с частотой 10 кГц;
- секвенсор 0 – запускается по ШИМ;
- секвенсор 0 – усреднение результатов по четырем точкам измерений, равномерно распределенным по периоду;
- секвенсор 0 – измерения по каналам 0 и 1, каналы имеют повышенный приоритет;
- секвенсор 0 – каждое измерение из двух последовательных опросов и усреднения;
- секвенсор 0 – прерывание каждые 2 записи в FIFO (по окончании измерений в текущем периоде);
- секвенсор 1 – запускается по сигналу таймера 0;
- секвенсор 1 – измерение канала 2;
- секвенсор 1 – каждое измерение из 4 последовательных опросов и усреднения;
- секвенсор 1 – прерывание при каждой записи в FIFO.

Код, соответствующий настройке и запуску необходимого режима, представлен ниже.

```
// Настройка таймера
RCU->PCLKCFG0_bit.TMR0EN = 1;
RCU->PRSTCFG0_bit.TMR0EN = 1;
TMR0->LOAD = 9999;
TMR0->ADCSOC_bit.EN = 1;
//Инициализация ШИМ
RCU->PCLKCFG0_bit.PWM0EN = 1;
RCU->PRSTCFG0_bit.PWM0EN = 1;
PWM0->TBCTL_bit.CLKDIV = 1;
PWM0->TBCTL_bit.HSPCLKDIV = 5;
```

```
PWM0->TBPRD = 250;
PWM0->TBCTL_bit.CTRMODE = 2;
PWM0->ETSEL_bit.SOCASEL = 1;
PWM0->ETSEL_bit.SOCAEN = 1;
//Настройка АЦП
ADC->ACTL_bit.ADCEN = 0x1;
ADC->CHCTL[0].CHCTL_bit.PRIORITY = 1;
ADC->CHCTL[1].CHCTL_bit.PRIORITY = 1;
//Настройка секвенсора 0
ADC->EMUX_bit.EM0 = 7;
ADC->SEQ[0].SCCTL_bit.ICNT = 1;
ADC->SEQ[0].SCCTL_bit.RCNT = 3;
ADC->SEQ[0].SCCTL_bit.RAVGEN = 1;
ADC->SEQ[0].SRTMR = 624;
ADC->SEQ[0].SRQCTL_bit.RQMAX = 1;
ADC->SEQ[0].SRQCTL_bit.QAVGVAL = 1;
ADC->SEQ[0].SRQCTL_bit.QAVGEN = 1;
ADC->SEQ[0].SRQSEL_bit.RQ0 = 0;
ADC->SEQ[0].SRQSEL_bit.RQ1 = 1;
ADC->SEQEN_bit.SEQEN0 = 1;
ADC->IM_bit.SEQIM0 = 1;
NVIC_EnableIRQ(ADC_SEQ0_IRQn);
//Настройка секвенсора 1
ADC->EMUX_bit.EM1 = 3;
ADC->SEQ[1].SRQCTL_bit.RQMAX = 0;
ADC->SEQ[1].SRQCTL_bit.QAVGVAL = 2;
ADC->SEQ[1].SRQCTL_bit.QAVGEN = 1;
ADC->SEQ[1].SRQSEL_bit.RQ0 = 2;
ADC->SEQEN_bit.SEQEN1 = 1;
ADC->IM_bit.SEQIM1 = 1;
NVIC_EnableIRQ(ADC_SEQ1_IRQn);
// Запуск
while(!ADC->ACTL_bit.ADCRDY);
TMR0->CTRL_bit.ON = 1;
SIU->PWMSYNC_bit.PRESCRST = 1<<0;
```

1 Включаем тактирование таймера 0 и снимаем сброс – установим бит TMR0EN в регистрах PCLKCFG0 и PRSTCFG0 соответственно.

2 Для того чтобы таймер 0 опустошался с частотой 10 кГц (при SYSCLK = 100 МГц), необходимо внести в регистр LOAD таймера 0 значение (100000 кГц/10 кГц) – 1 = 9999.

3 Разрешаем генерацию таймером 0 запросов на старт преобразования, установив бит EN в регистре ADCSOC таймера 0.

4 Включаем тактирование ШИМ0 и снимаем сброс – установим бит PWM0EN в регистрах PCLKCFG0 и PRSTCFG0 соответственно.

5 Настроим делители тактового сигнала ШИМ таким образом, чтобы частота счета TBCLK была равна 5 МГц (при SYSCLK = 100 МГц). Для этого запишем в поля регистра ШИМ TBCTL значения CLKDIV = 1 (коэффициент 1/2), HSPCLKDIV = 5 (коэффициент 1/10).

6 Режим счета двусторонний – поле CTRMODE = 2 регистра TBCTL, а период счета (регистр TBPRD), соответственно, равен 5000 кГц/(10 кГц)/2 = 250.

7 Разрешаем генерацию ШИМ0 запросов по каналу A на старт преобразования, установив бит SOCAEN в регистре ETSEL и записав единицу в поле ETSEL того же регистра.

8 Разрешаем работу модуля АЦП – необходимо установить бит ADCEN в регистре ACTL.

9 Устанавливаем высокий приоритет для каналов, которые будут обрабатываться секвенсором 0 – каналов 0, 1, установив бит PRIRORITY в соответствующих регистрах CHCTL.

10 Проинициализируем секвенсор 0. Настроим его источник запуска – поле EM0 регистра EMUX должно быть равно 7h, т. к. запуск по сигналу канала А ШИМ.

11 Необходимое количество записей в FIFO для генерации прерывания – 2, поэтому запишем в поле ICNT регистра SEQ0->SCCTL значение 2 – 1 = 1h.

12 По каждому запуску должно совершиться 3 перезапуска с паузой в четверть периода ШИМ. В поле количества перезапусков RCNT регистра SEQ0->SCCTL внесем 3h.

В регистр задержки перезапуска SEQ0->SRTMR внесем значение 25000 кГц/(10 кГц×4) – 1 = 624.

13 В поле RQMAX регистра SEQ0->SRQCTL необходимо внести значение 1h, т. к. измерения будут проводиться по двум каналам.

14 Настраиваем каналы для запроса. В регистр SEQ0->SRQSEL необходимо внести значения RQ0 = 0h, RQ1 = 1h.

15 Включаем усреднение сканированием по двум опросам очереди. В регистре SEQ0->SRQCTL нужно установить поля QAVGVAL = 1, QAVGEN = 1.

16 Разрешаем генерацию прерываний: нужно установить бит SEQIM0 в регистре IM.

17 Разрешаем работу секвенсора. Для этого необходимо установить бит SEQEN0 в регистре SEQEN.

18 Проинициализируем секвенсор 1. Настроим его источник запуска: поле EM1 регистра EMUX должно быть равно 3h, т. к. запуск по сигналу таймера 0.

19 В поле RQMAX регистра SEQ1->SRQCTL необходимо внести значение 0h, т. к. измерения будут проводиться по одному каналу.

20 Настраиваем каналы для запроса. В регистр SEQ1->SRQSEL необходимо внести значение RQ0 = 2h.

21 Включаем усреднение сканированием по четырем опросам очереди. В регистре SEQ1->SRQCTL нужно установить поля QAVGVAL = 2, QAVGEN = 1.

22 Разрешаем генерацию прерываний: нужно установить бит SEQIM1 в регистре IM.

23 Разрешаем работу секвенсора. Для этого необходимо установить бит SEQEN1 в регистре SEQEN.

24 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. А также разрешить работу предделителя ШИМ, записав 1h в поле PRESCRST регистра PWMSYNC блока SIU.

25 Измерения секвенсора 0 будут запускаться по каждому равенству нулю счетного регистра ШИМ0. После каждого запуска будет проведено по три отложенных перезапуска. Т. к. дополнительно включено усреднение по перезапускам, то два усреднённых результата попадут в FIFO в последнем третьем перезапуске и будет вызвано прерывание. Измерения секвенсора 1 будут производиться по каждому опустошению таймера 0. После усреднения сканированием результат будет записан в FIFO и будет вызвано прерывание.

Работа режима проиллюстрирована на рисунке 19.22.

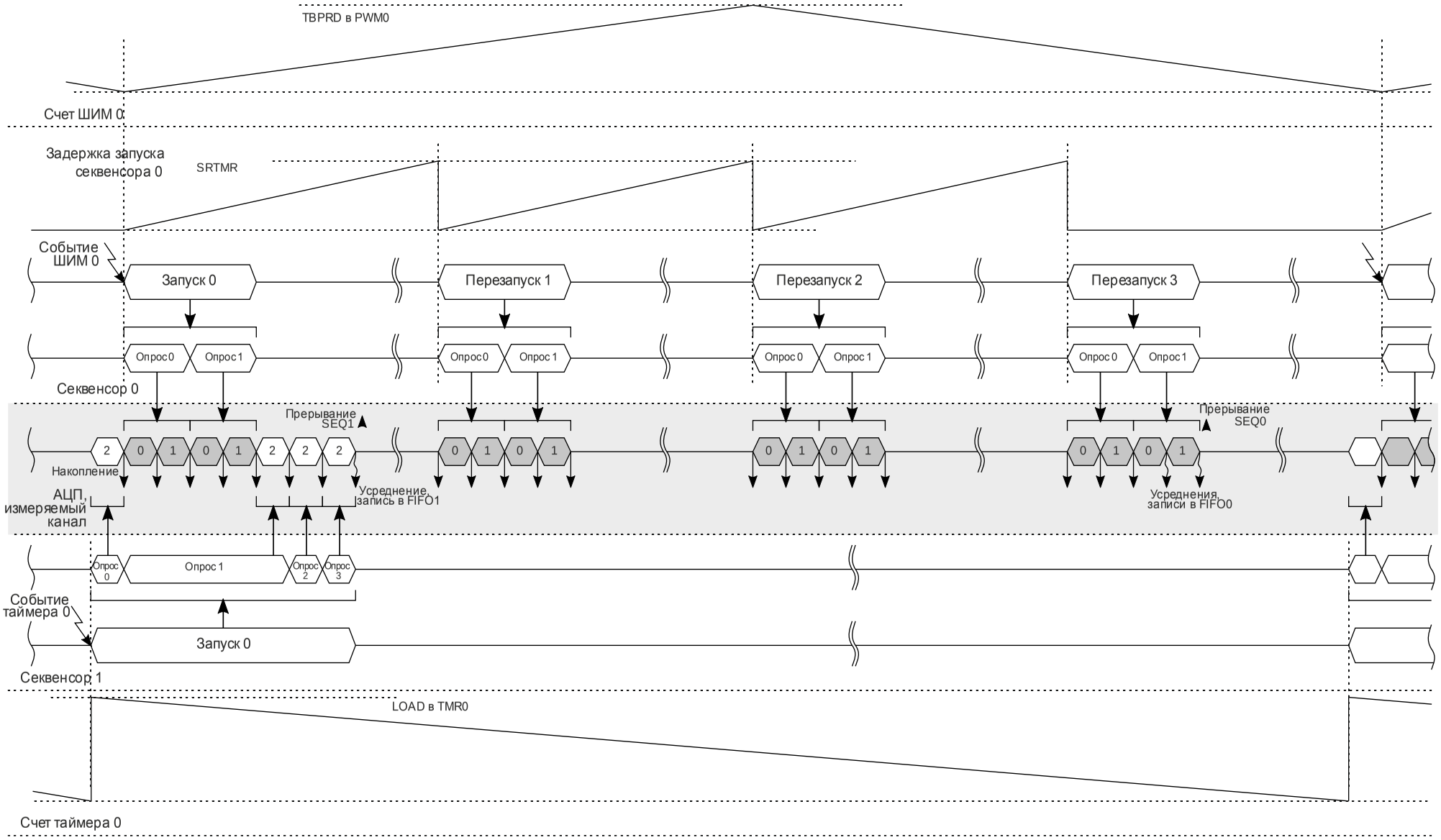


Рисунок 19.22 – Диаграммы запуска нескольких секвенсоров

20 Сторожевой таймер

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.

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. Алгоритм работы сторожевого таймера показан на рисунке 20.1.

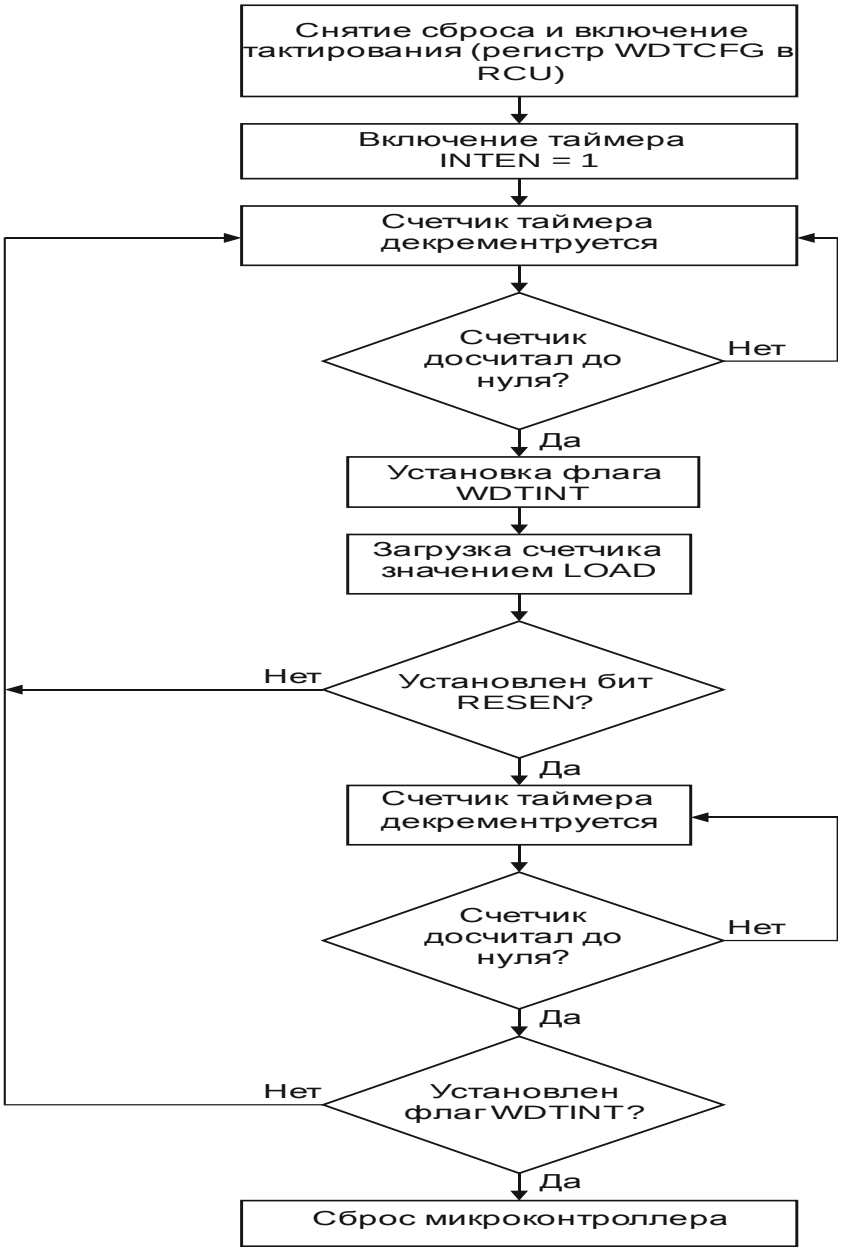


Рисунок 20.1 – Алгоритм работы сторожевого таймера

21 Программно-аппаратные средства отладки

Для освоения и изучения 32-разрядного микроконтроллера 1921ВК035, 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.

Описание макетно-отладочной платы приведено в КФДЛ.441461.018РЭ.

Для создания программного обеспечения рекомендуется использовать программный продукт CodeMaster++[ARM], который представляет собой набор программно-аппаратных средств для разработки и отладки систем на базе микроконтроллера 1921ВК035.

Среда разработки CodeMaster++[ARM] включает в себя менеджер проектов, редактор исходных кодов, компилятор (C, C++), средства отладки и симуляции микроконтроллера 1921ВК035. Среда позволяет осуществлять отладку программ, а также программирование микроконтроллера посредством JTAG эмулятора JEM-NT-CM4.

Адаптер JEM-NT-CM4 обеспечивает взаимодействие между интегрированной средой разработки CodeMaster++[ARM], установленной на персональном компьютере, и отладочными ресурсами, встроенными в микроконтроллер 1921ВК035, а также выполнение отладочных функций. Информационный обмен с микроконтроллером осуществляется по одному из отладочных портов JTAG или SWD.

Отладка пользовательской программы предполагает два основных режима работы:

- выполнение программы в режиме реального времени RUN;
- останов программы на определенном адресе или в определенный момент выполнения HALT.

Большинство отладочных функций доступно исключительно в режиме останова. В этом режиме отладчик JEM-NT-CM4 во взаимодействии с CodeMaster++[ARM] позволяет анализировать и изменять ход исполнения пользовательской программы между отдельными участками программы, исполняемыми в режиме RUN.

При работе с пользовательской программой адаптер JEM-NT-CM4 обеспечивает выполнение следующих отладочных действий:

- сброс микроконтроллера с остановом пользовательской программы на начальном адресе;
- запуск на выполнение программы в режиме реального времени RUN;
- останов программы в произвольный момент времени STOP;
- определение и изменение адреса выполнения программы (чтение и запись счетчика команд);
- чтение и запись доступных ресурсов микроконтроллера (ОЗУ, Flash-память, SFR и т. д.);
- установку и снятие точек останова по адресу выполнения программы;
- запуск на выполнение программы до определенного места в исходном коде (до курсора, до адреса);
- пошаговое исполнение программы: шаги низкого и высокого уровней, с заходом и без захода в подпрограммы.

Заключение

В настоящем руководстве пользователя представлено описание архитектуры, функционального построения и периферии микроконтроллера 1921ВК035.

Приложение А
(обязательное)
Регистры микроконтроллера

В данном приложении представлены регистры с указанием их назначений, мнемонических названий и адресов.

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 контроллера АЦП

Базовый адрес: 4000_0000h

Смещение:

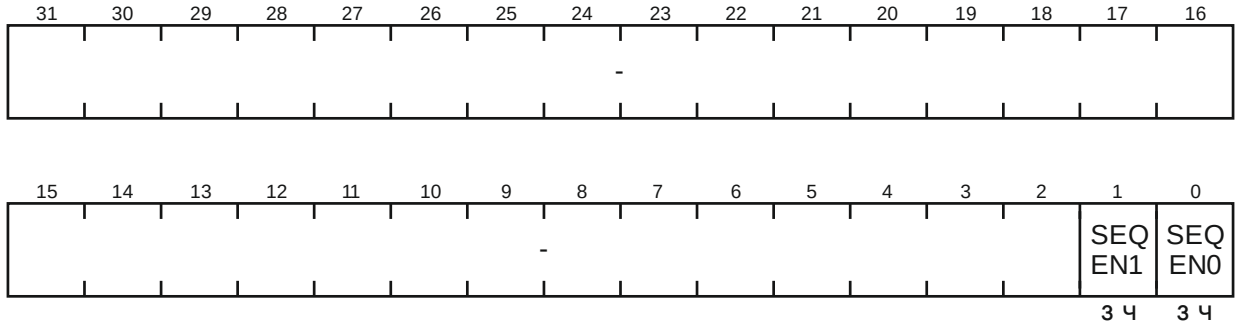
+ 40h (SEQ0)	Регистры секвенсора 0
+ 74h (SEQ1)	Регистры секвенсора 1
+ 200h (DC0)	Регистры цифрового компаратора 0
+ 20Ch (DC1)	Регистры цифрового компаратора 1
+ 218h (DC2)	Регистры цифрового компаратора 2
+ 224h (DC3)	Регистры цифрового компаратора 3
+ 500h (CHCTL)	Регистры настройки каналов

Мнемоника: SEQs;
DCd

Примечание – s – номер секвенсора 0 или 1;
d – номер по порядку цифрового компаратора от 0 до 3.

SEQEN – регистр включения секвенсоров

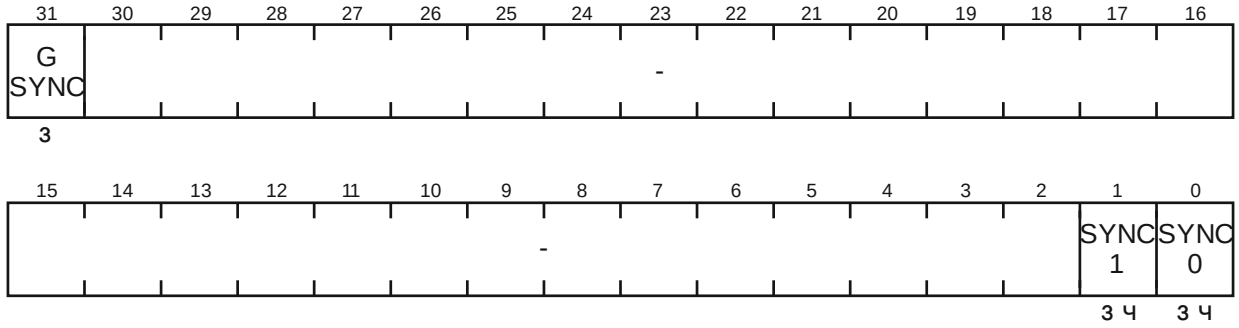
Смещение: + 00h.
Сброс: 0h



Поле	Биты	Описание	
SEQENs	1-0	Бит разрешения работы секвенсора s	
		0	Запрещено
		1	Разрешено
–	31-2	Зарезервировано	

SEQSYNC – регистр программной синхронизации секвенсоров

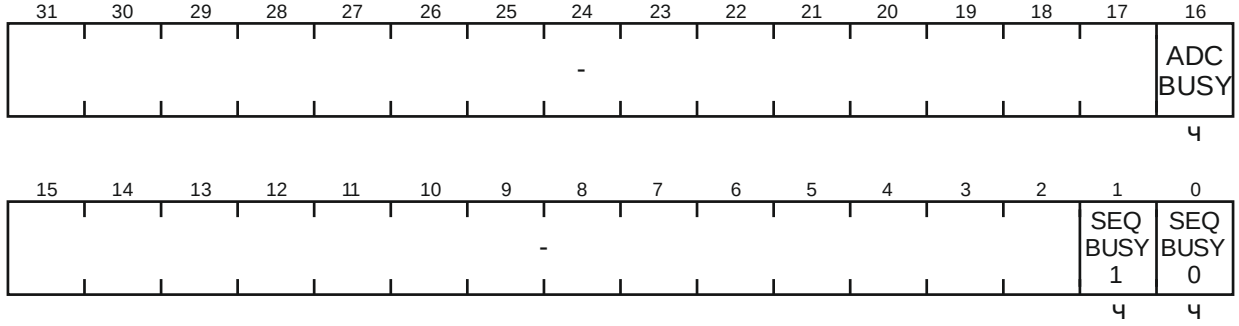
Смещение: + 04h.
Сброс: 0h



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

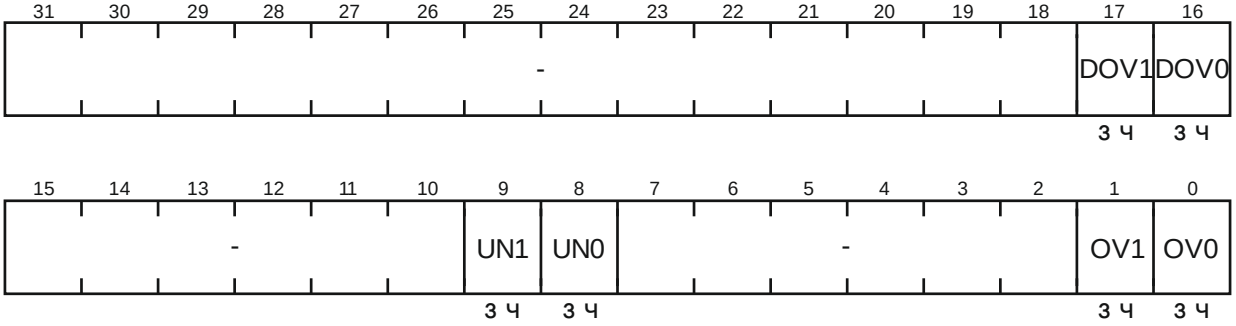
Смещение: + 08h
Сброс: 0h



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

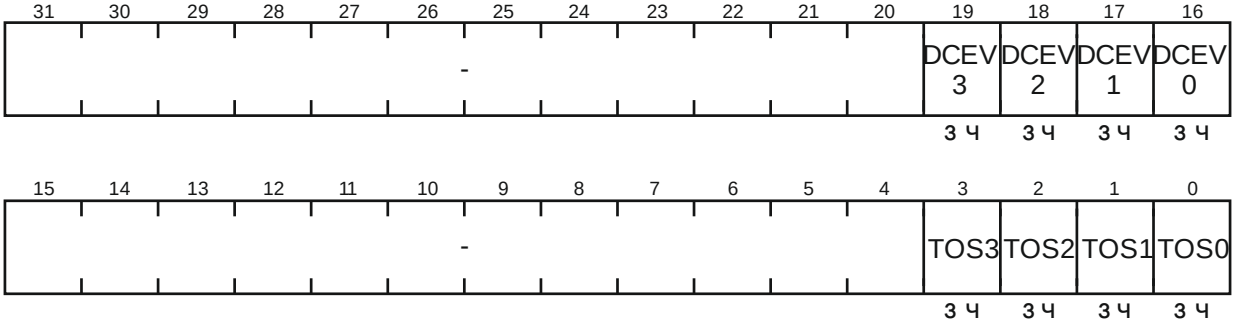
Смещение: + 0Ch
Сброс: 0h



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 компараторов

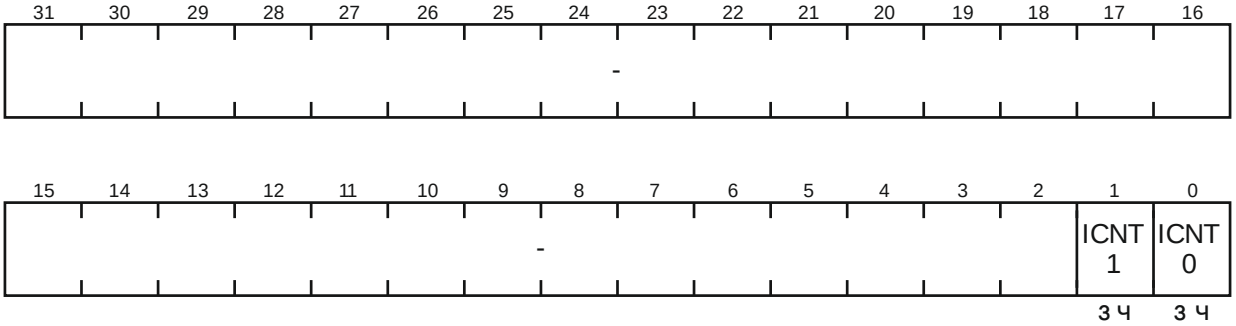
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание	
DCEVd	19-16	Флаг прохождения события сравнения	
		0	Сравнение не выполнялось
		1	Сравнение выполнялось
		Флаг сбрасывается записью единицы	
TOSd	3-0	Флаг состояния выходного триггера компаратора d	
		0	Триггер не сработал
		1	Триггер сработал
		Флаг сбрасывается записью единицы	
–	31-20, 15-4	Зарезервировано	

CICNT – регистр настройки режима сброса счетчика прерываний

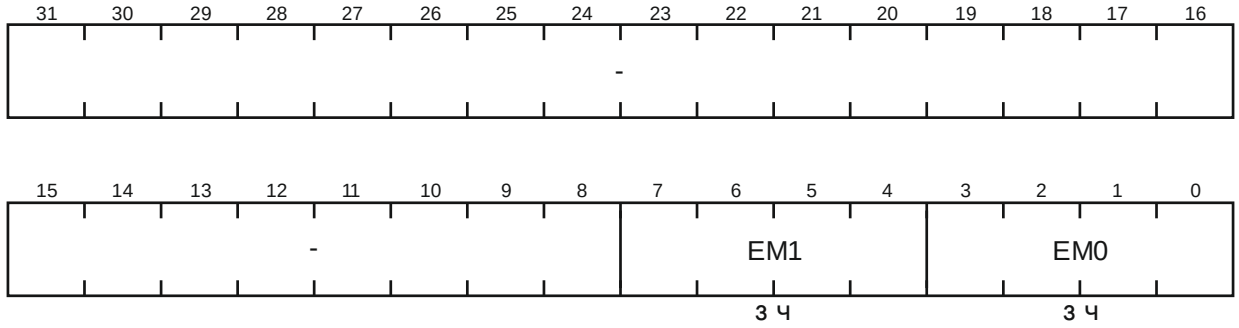
Смещение: + 14h
Сброс: 0h



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

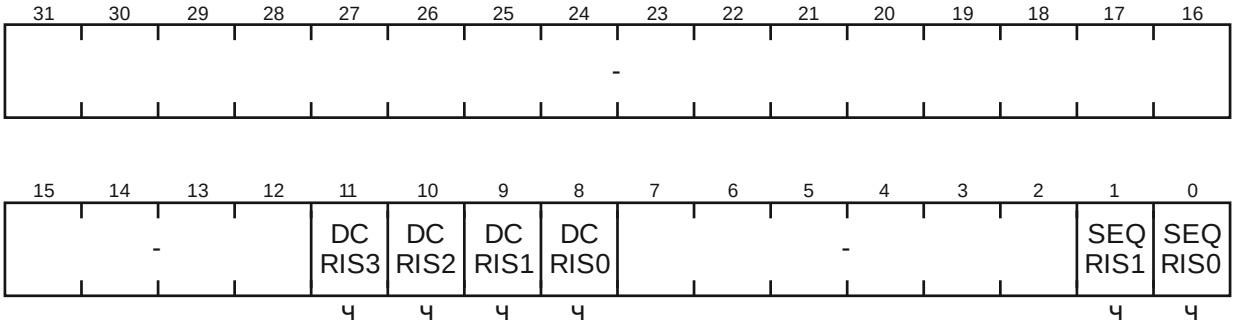
Смещение: + 18h
Сброс: 0h



Поле	Биты	Описание	
EMs	7-0	Поле выбора события для запуска секвенсора s	
		0h	Установка бита GSYNC в регистре SEQSYNC
		1h	Прерывание от GPIOA
		2h	Прерывание от GPIOB
		3h	Сигнал от блока TMR0
		4h	Сигнал от блока TMR1
		5h	Сигнал от блока TMR2
		6h	Сигнал от блока TMR3
		7h	Сигналы от блоков PWM0, PWM1, PWM2 – канал A
		8h	Сигналы от блоков PWM0, PWM1, PWM2 – канал B
		9h-Eh	Зарезервировано
		Fh	Циклическая работа. Активируется после установки бита GSYNC в регистре SEQSYNC
–	31–8	Зарезервировано	

RIS – регистр флагов немаскированных прерываний

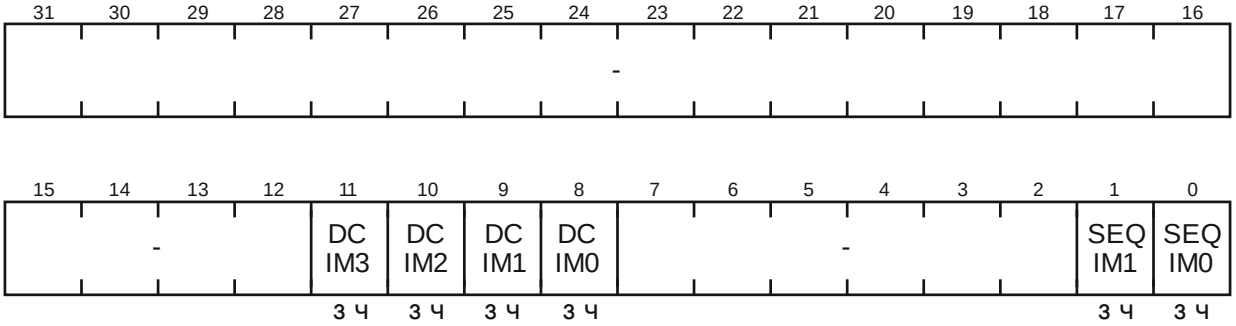
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание	
DCRISd	11-8	Флаг прерывания компаратора d	
		0	Нет прерывания или флаг сброшен
		1	Поступил запрос на прерывание
SEQRISs	1-0	Флаг прерывания секвенсора s	
		0	Нет действий
		1	Запрос секвенсора завершился и счетчик прерываний досчитал до значения ICNT регистра SCCTL
–	31-12, 7-2	Зарезервировано	

IM – регистр маски прерываний

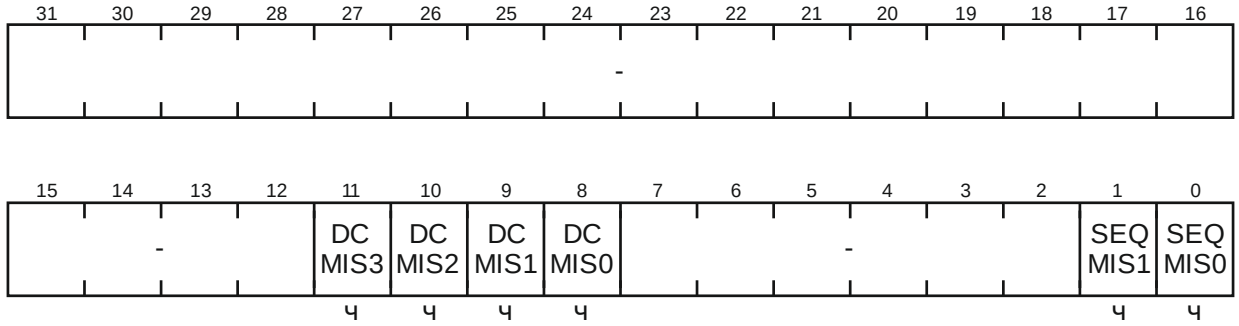
Смещение: + 20h
Сброс: 0h



Поле	Биты	Описание	
DCIMd	11-8	Маска прерывания компаратора d	
		0	Маскировано
		1	Разрешено
SEQIMs	1-0	Маска прерывания секвенсора s	
		0	Маскировано
		1	Разрешено
–	31-12, 7-2	Зарезервировано	

MIS – регистр флагов маскированных прерываний

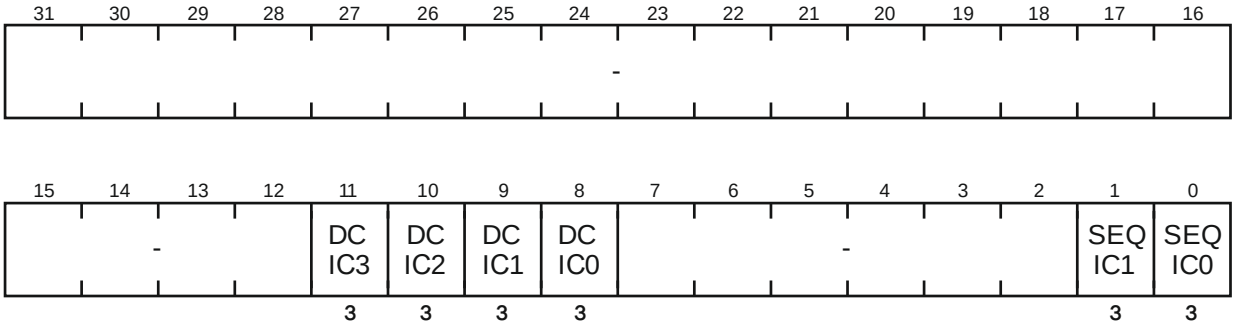
Смещение: + 24h
Сброс: 0h



Поле	Биты	Описание	
DCMISd	11-8	Флаг маскированного прерывания компаратора d	
		0	Нет прерывания или флаг сброшен
		1	Поступил запрос на прерывание
SEQMISs	1-0	Флаг маскированного прерывания секвенсора s	
		0	Нет действий
		1	Запрос секвенсора завершился и счетчик прерываний досчитал до значения ICNT регистра SCCTL
–	31-12, 7-2	Зарезервировано	

IC – регистр сброса флагов прерываний

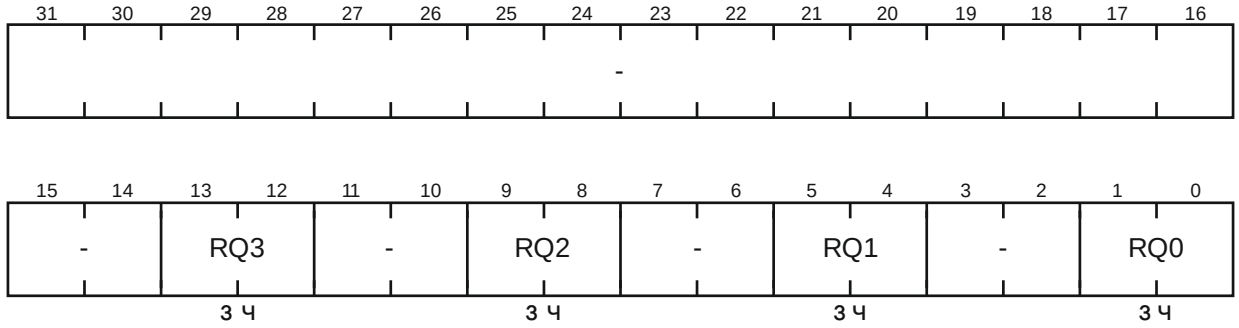
Смещение: + 28h
Сброс: 0h



Поле	Биты	Описание	
DCICd	11-8	Сброс маскированного и немаскированного флага прерывания компаратора d	
		0	Нет действий
		1	Сброс флагов
SEQICs	1-0	Сброс маскированного и немаскированного флага прерывания секвенсора s	
		0	Нет действий
		1	Сброс флагов
–	31-12, 7-2	Зарезервировано	

SRQSEL – регистр выбора каналов для запросов секвенсора

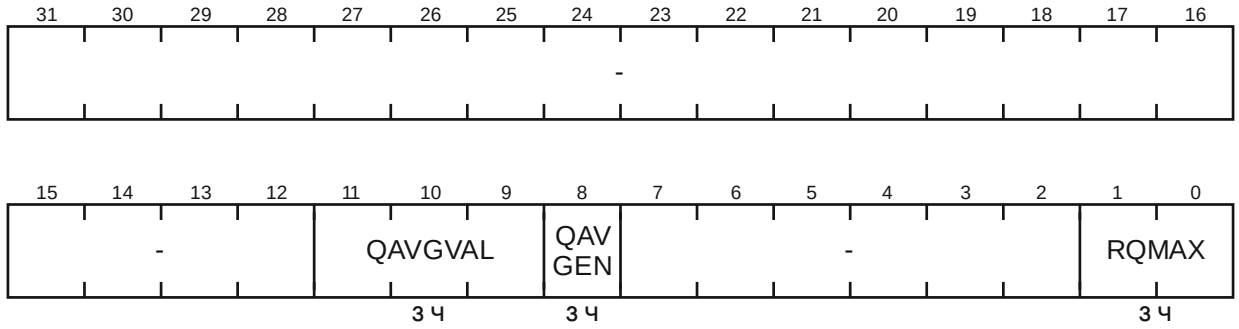
Смещение: SEQs + 00h
Сброс: 0h



Поле	Биты	Описание
RQn	13-12, 9-8, 5-4, 1-0	Номер канала АЦП для запроса секвенсора s. Допустимые значения от 0h до 3h.
–	31-14, 11-10, 7-6, 3-2	Зарезервировано

SRQCTL – регистр управления очередью запросов секвенсора

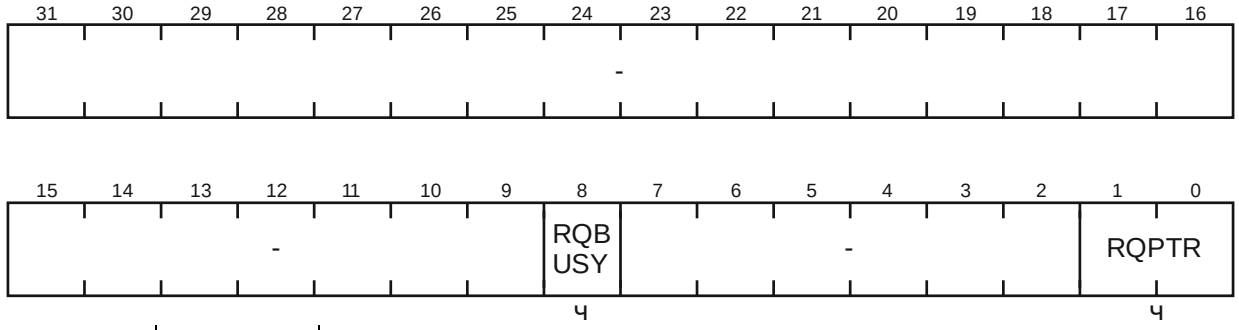
Смещение: SEQs + 10h
Сброс: 0h



Поле	Биты	Описание	
QAVGVAL	11-9	Поле задания количества необходимых опросов для усреднения сканированием	
		000	Зарезервировано
		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	1-0	Поле задания номера последнего элемента секвенсора s	
–	31-12, 7-2	Зарезервировано	

SRQSTAT – регистр статуса очереди запросов секвенсора

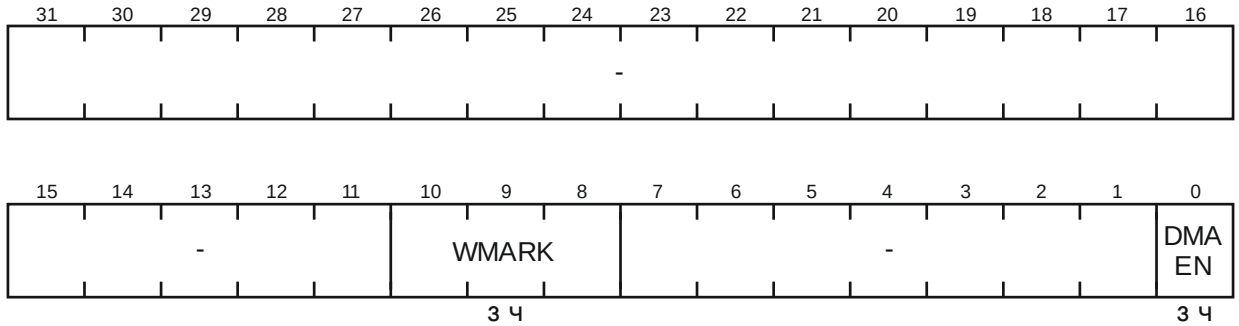
Смещение: SEQs + 14h
Сброс: 0h



Поле	Биты	Описание	
RQBUSY	8	Флаг активного запроса секвенсора на измерение	
		0	Текущий запрос неактивен
		1	Выставленый запрос в состоянии обработки или ожидания
RQPTR	1-0	Номер текущего запроса в очереди	
–	31-9, 7-2	Зарезервировано	

SDMACTL – регистр управления запросами DMA секвенсора

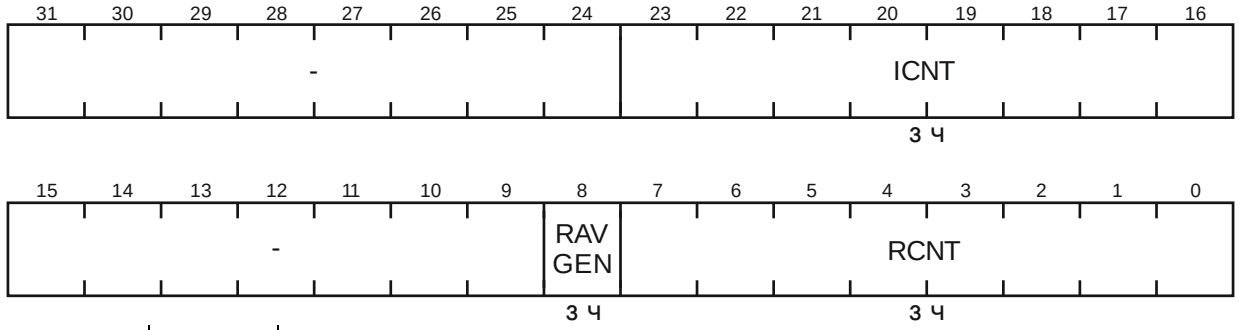
Смещение: SEQs + 18h
Сброс: 0h



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

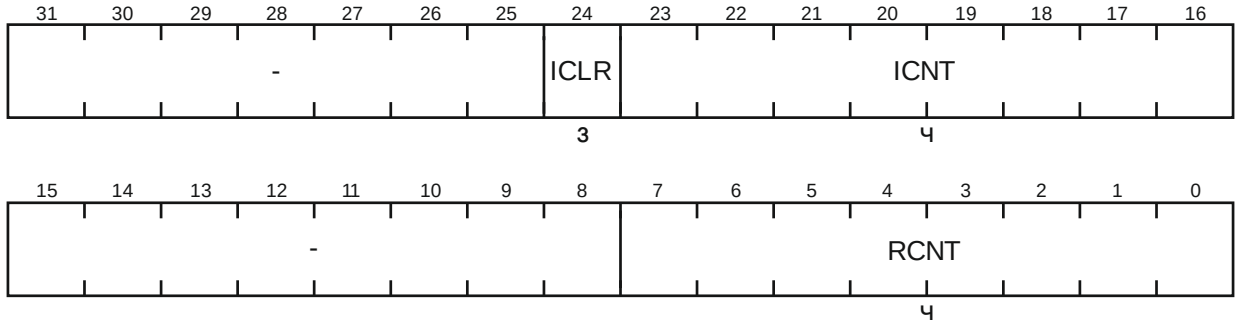
Смещение: SEQs + 1Ch
Сброс: 0h



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 от 1 до 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

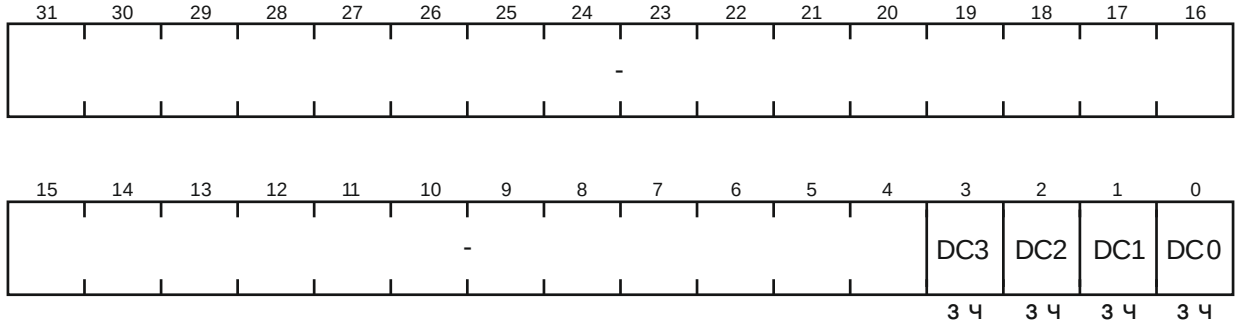
Смещение: SEQs + 20h
Сброс: 0h



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

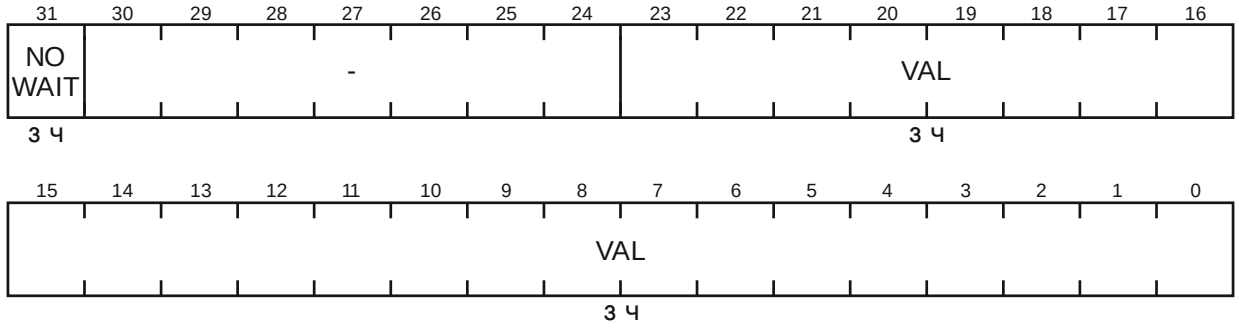
Смещение: SEQs + 24h
Сброс: 0h



Поле	Биты	Описание	
DCd	3-0	Бит разрешения работы компаратора d секвенсором s	
		0	Запрещен
		1	Разрешен
–	31-4	Зарезервировано	

SRTMR – регистр задержки перезапусков секвенсора

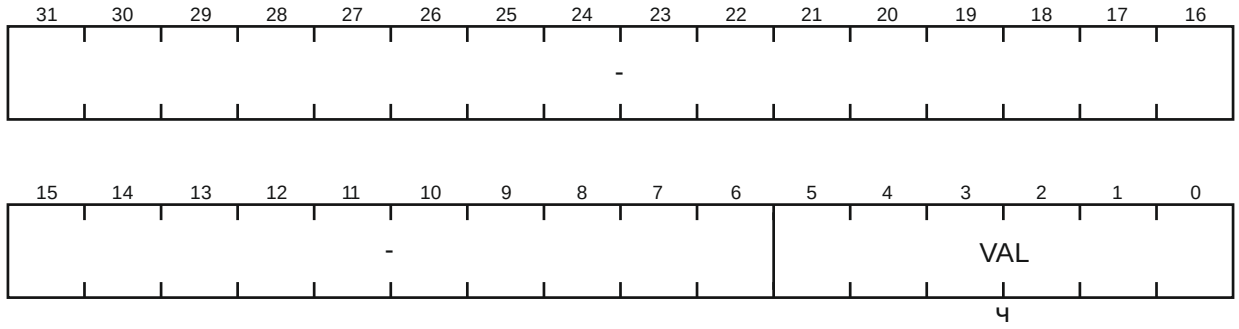
Смещение: SEQs + 28h
Сброс: 0h



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

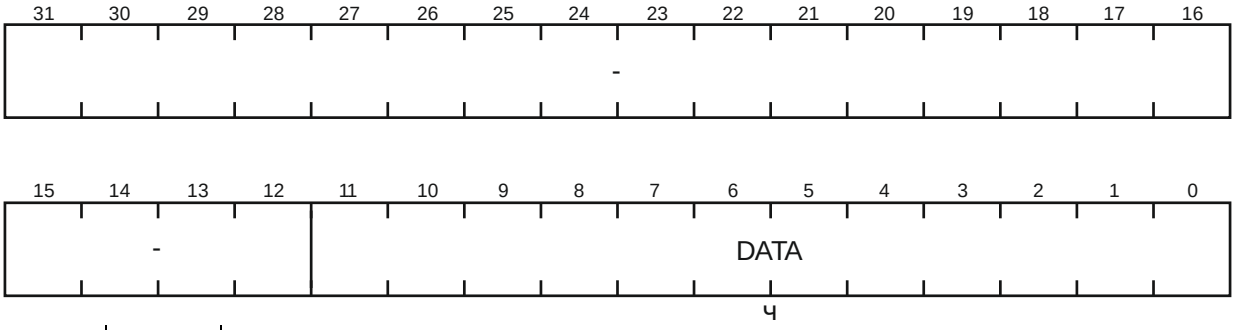
Смещение: SEQs + 2Ch
Сброс: 0h



Поле	Биты	Описание
VAL	5-0	Значение количества результатов измерений, сохраненных в буфере секвенсора
–	31-6	Зарезервировано

SFIFO – регистр результата измерения секвенсора

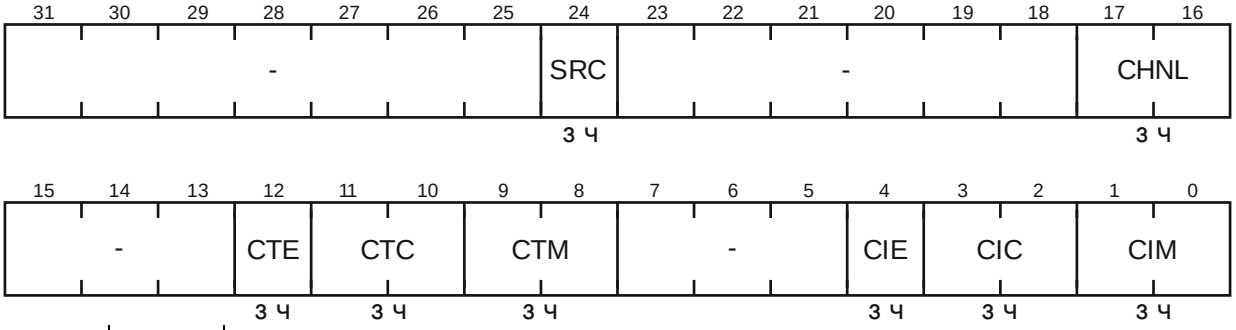
Смещение: SEQs + 30h
Сброс: 0h



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: 0h

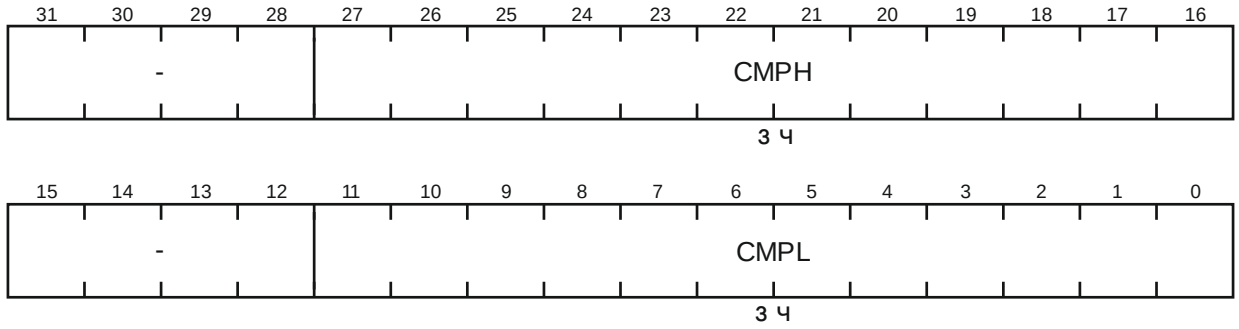


Поле	Биты	Описание	
SRC	24	Выбор источника значения для сравнения	
		0	Окончание измерения АЦП
		1	Запись результата в FIFO секвенсором
CHNL	17-16	Поле номера канала, результат измерения которого будет передан на компаратор. Допустимые значения от 0h до 3h	
CTE	12	Бит разрешения срабатывания выходного триггера компаратора	
		0	Запрещено
		1	Разрешено
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	

Поле	Биты	Описание	
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
–	31-25, 23-18, 15-13, 7-5	Зарезервировано	

DCMP – регистр диапазона компаратора

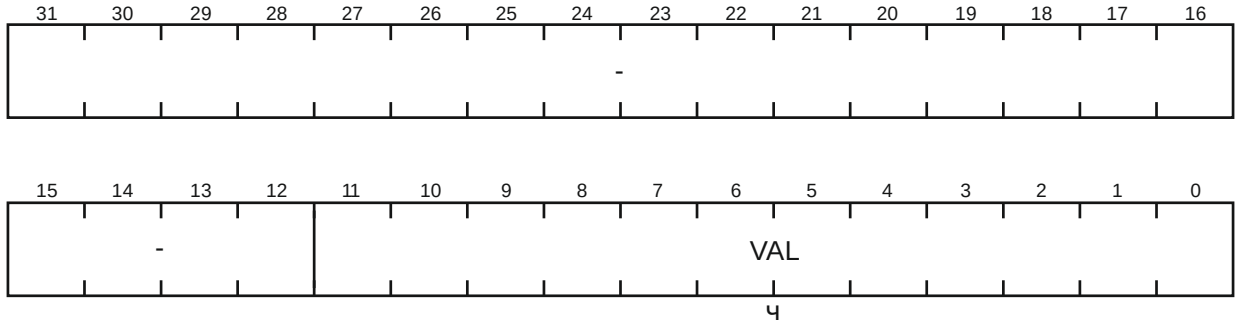
Смещение: DCd + 04h
Сброс: 0h



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

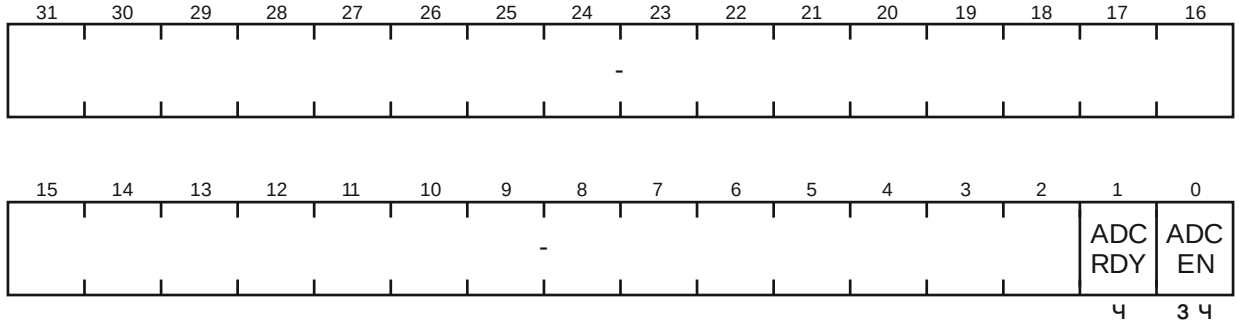
Смещение: DCd + 08h
Сброс: 0h



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

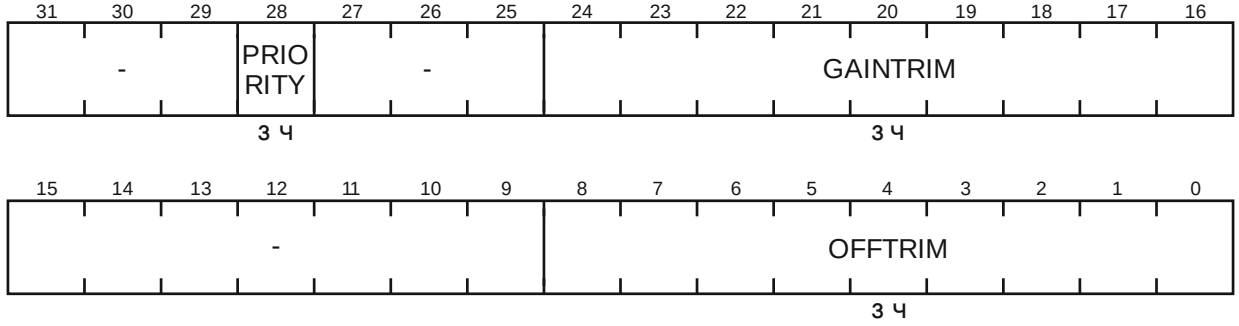
Смещение: + 400h
Сброс: 0h



Поле	Биты	Описание	
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
–	31-2	Зарезервировано	

CHCTL – массив регистров настройки каналов

Смещение: CHCTL + (04*n)h, где n – номер канала от 0 до 3
Сброс: 0h



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, величина вносится в дополнительном коде: 100h соответствует –256, 000h – 0, 0FFh – 255	
OFFTRIM	8-0	Поле задания коэффициента для коррекции смещения нуля. Диапазон значений –256, …, 255, величина вносится в дополнительном коде: 100h соответствует –256, 000h – 0, 0FFh – 255	
–	31-29, 27-25, 15-9	Зарезервировано	

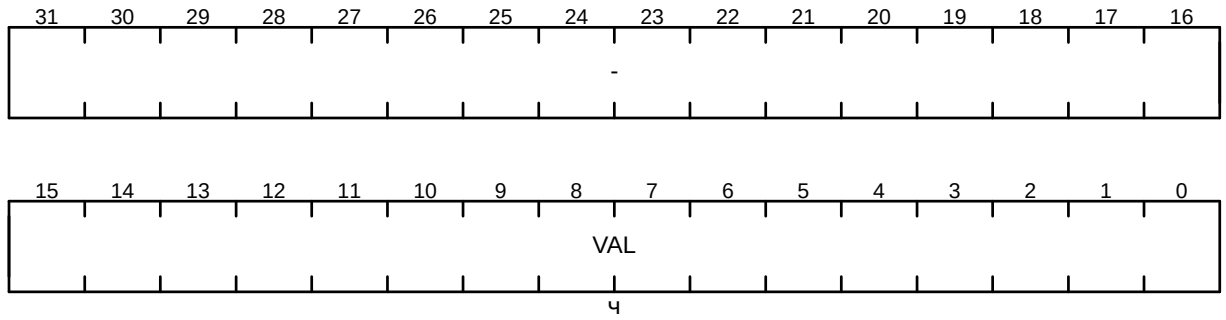
A.2 Регистры портов (GPIO)

Базовый адрес:	4001_0000h (GPIOA)	Регистры порта A
	4001_1000h (GPIOB)	Регистры порта B
Смещение:	+ 400h (MASKLB)	Массив регистров масок младшего байта порта
	+ 800h (MASKHB)	Массив регистров масок старшего байта порта
Мнемоника:	GPIOp	

Примечание – p – имя порта A или B;
n – номер по порядку вывода порта от 0 до 15.

DATA – регистр входных данных порта

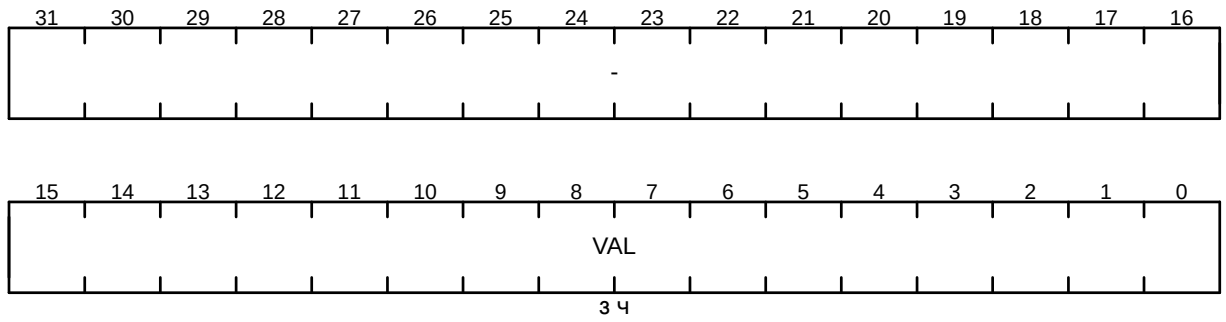
Смещение: + 00h
Сброс: 0000_xxxxh



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает текущее состояние порта
		Запись	Не выполняется
–	31-16	Зарезервировано	

DATAOUT – регистр выходных данных порта

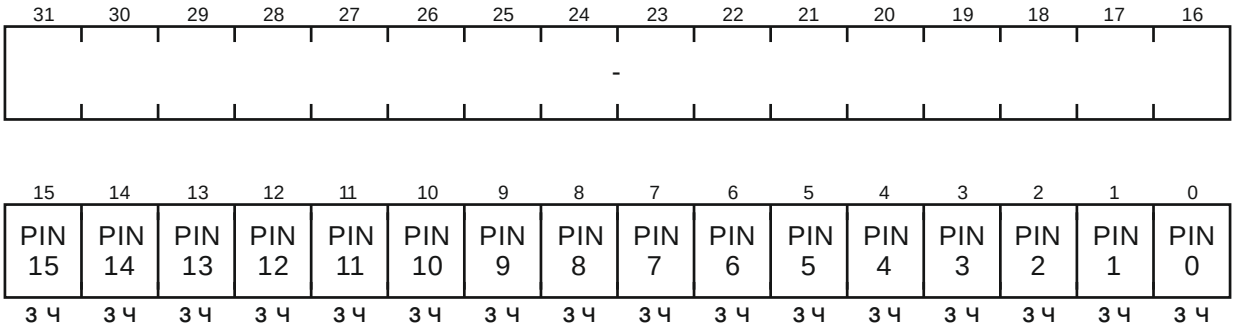
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание	
VAL	15-0	Чтение	Возвращает текущее состояние порта
		Запись	Устанавливает на выводах порта уровень сигнала, соответствующий записанному в биты значению
–	31-16	Зарезервировано	

DATAOUTSET – регистр установки битов порта

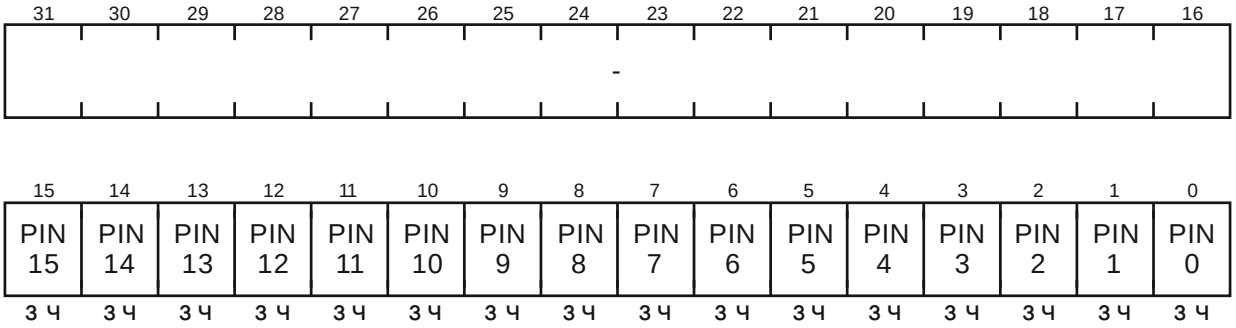
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Не допускается
		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

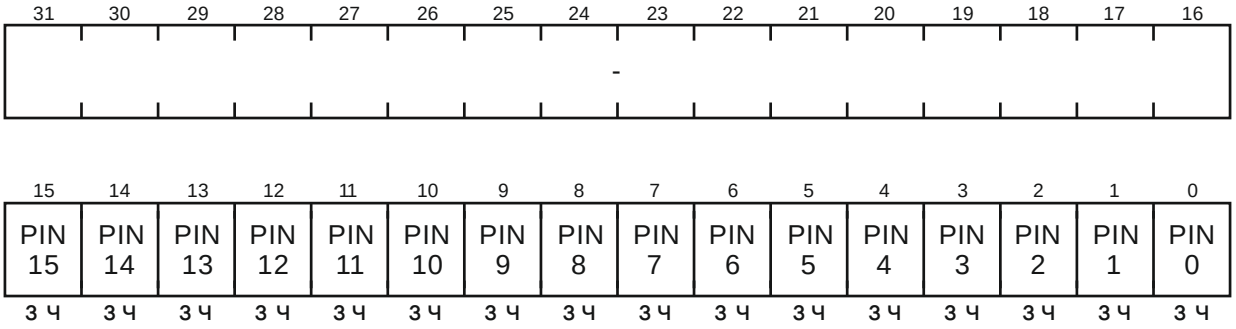
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Сбрасывает соответствующий бит n в регистре DATAOUT и, как следствие, устанавливает низкий уровень сигнала на выводе n порта
–	31-16	Зарезервировано	

DATAOUTTGL – регистр переключения битов порта

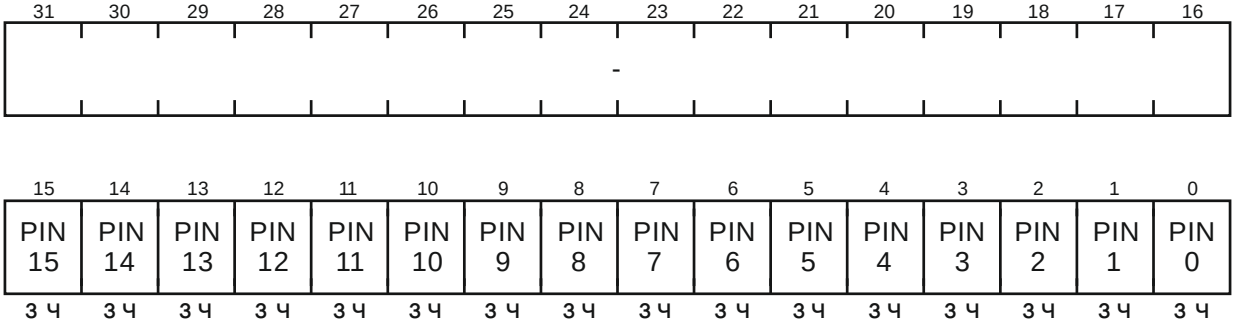
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Изменяет состояние соответствующего бита n в регистре DATAOUT на противоположное и, как следствие, состояние сигнала на выводе n порта
–	31-16	Зарезервировано	

DENSET – регистр разрешения цифровой функции порта

Смещение: + 14h
Сброс: 7Ch (для порта A), 0h (для порта B)

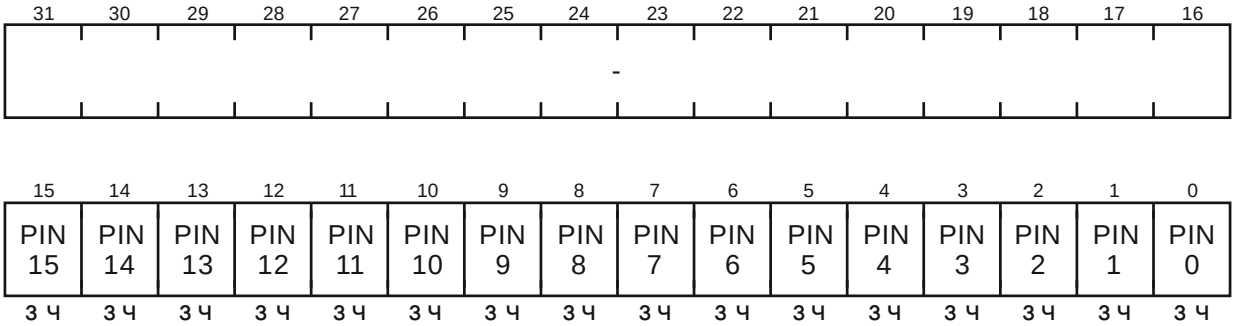


Поле	Биты	Описание		
PINn	15-0	Чтение	0	Вывод n отключен
			1	Вывод n подключен
		Запись нуля	Не выполняется	
		Запись единицы	Подключает вывод n к его цифровой схеме, т. е. разрешает функционирование вывода как входа/выхода порта	
–	31-16	Зарезервировано		

Примечание – Выводы A2 – A6 порта A, которым соответствуют биты PIN2 – PIN6 подключены по умолчанию, поскольку имеют альтернативные функции, связанные с портом JTAG.

DENCLR – регистр запрещения цифровой функции порта

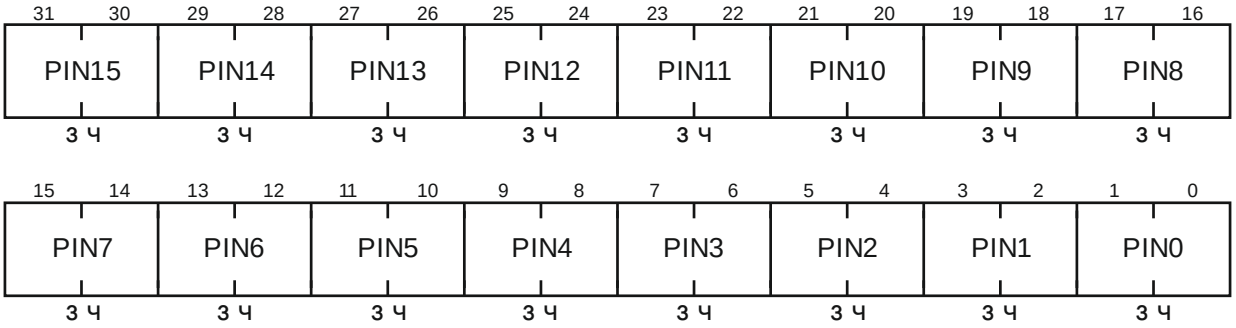
Смещение: + 18h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Отключает вывод n от его цифровой схемы, т. е. запрещает функционирование вывода как входа/выхода порта
–	31-16	Зарезервировано	

INMODE – регистр выбора режима входа порта

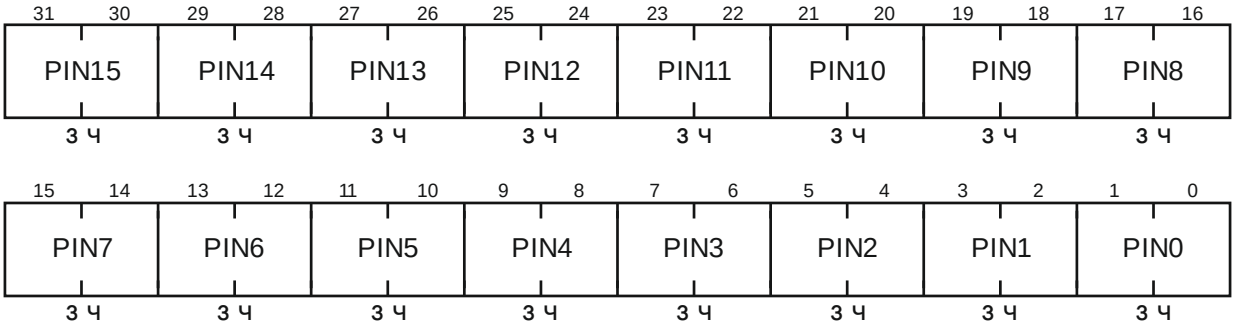
Смещение: + 1Ch
Сброс: 0h



Поле	Биты	Описание	
PINn	31-0	Поле выбора режима входа n	
		00	Триггер Шмитта
		01	КМОП-буфер
		10	Зарезервировано
		11	Входной буфер отключен

PULLMODE – регистр выбора режима подтяжки порта

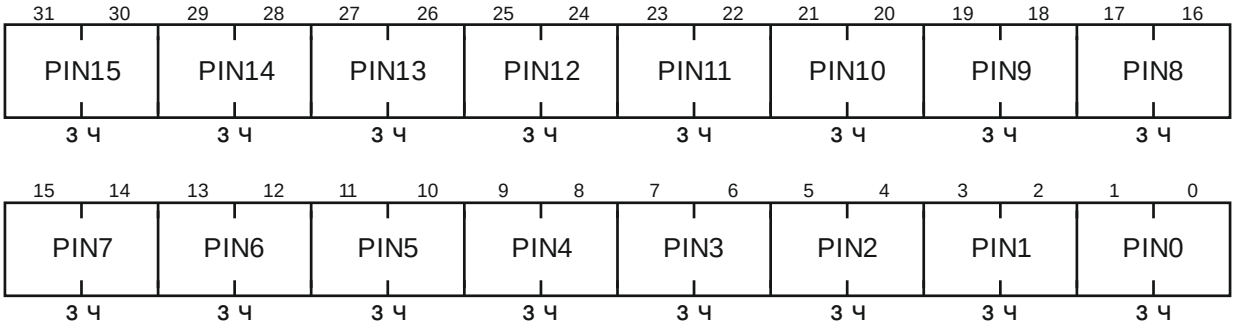
Смещение: + 20h
Сброс: 0h



Поле	Биты	Описание	
PINn	31-0	Поле выбора режима подтяжки вывода n	
		00	Подтяжка отключена
		01	Подтяжка к уровню логической единицы (Pull-up)
		10	Подтяжка к уровню логического нуля (Pull-down)
		11	Зарезервировано

OUTMODE – регистр выбора режима выхода порта

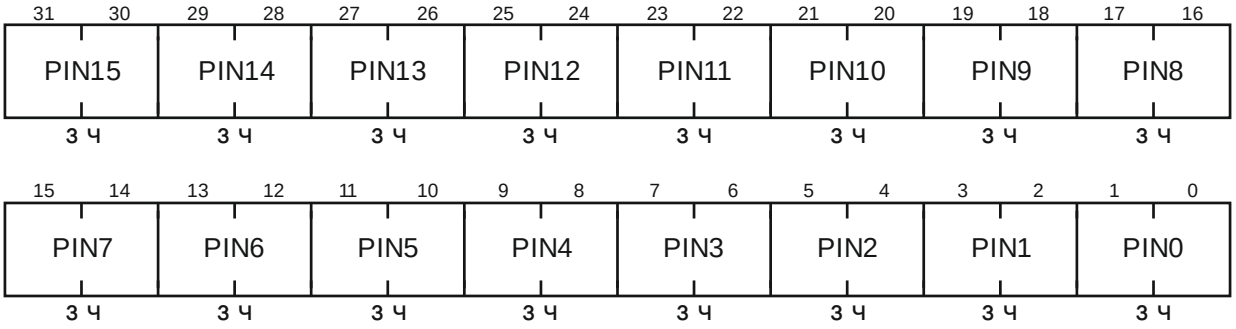
Смещение: + 24h
Сброс: 0h



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

DRIVEMODE – регистр выбора параметров выхода порта

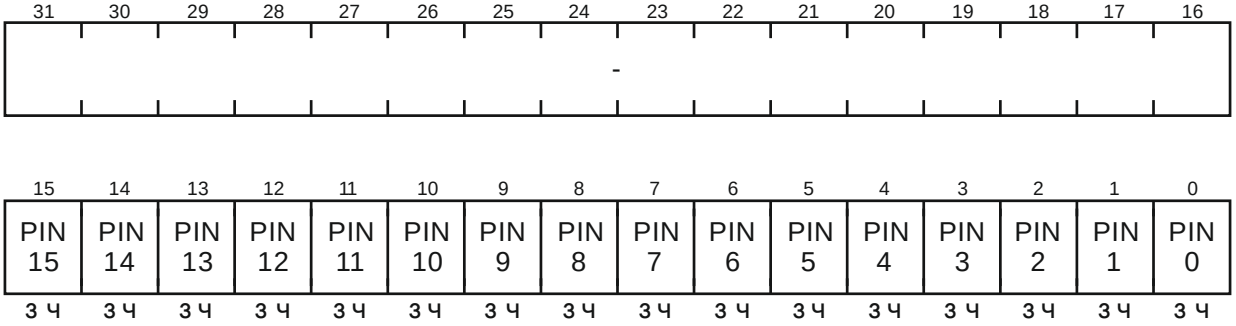
Смещение: + 28h
Сброс: 0h



Поле	Биты	Описание	
PINn	31-0	Поле выбора параметров выхода порта	
		00	Высокая нагрузочная способность и высокая скорость
		01	Высокая нагрузочная способность и низкая скорость
		10	Низкая нагрузочная способность и высокая скорость
		11	Низкая нагрузочная способность и низкая скорость

OUTENSET – регистр разрешения управления выходом порта

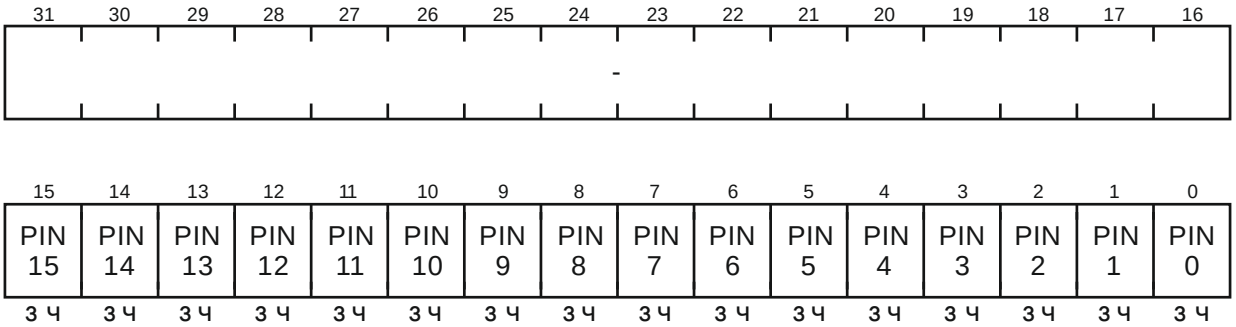
Смещение: + 2Ch
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Источник выходного сигнала (n-ый бит регистра DATAOUT) отключен от вывода n.
			1	Состояние вывода n задается источником выходного сигнала (бит n регистра DATAOUT)
		Запись нуля	Не выполняется	
		Запись единицы	Разрешает управлять состоянием вывода с помощью соответствующего бита регистра DATAOUT	
–	31-16	Зарезервировано		

OUTENCLR – регистр запрещения управления выходом порта

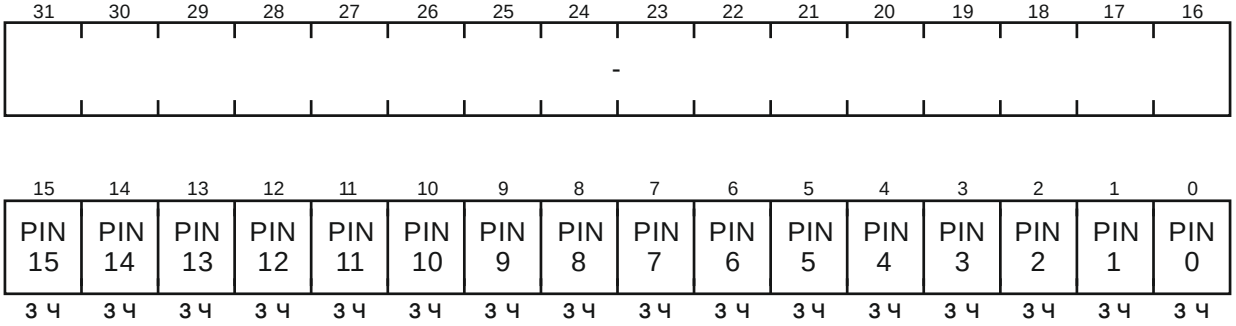
Смещение: + 30h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Сбрасывает соответствующий бит n регистра OUTENSET и запрещает управлять состоянием вывода с помощью соответствующего бита регистра DATAOUT
–	31-16	Зарезервировано	

ALTFUNCSET– регистр включения альтернативной функции порта

Смещение: + 34h
Сброс: 7Ch (для порта A), 0h (для порта B)

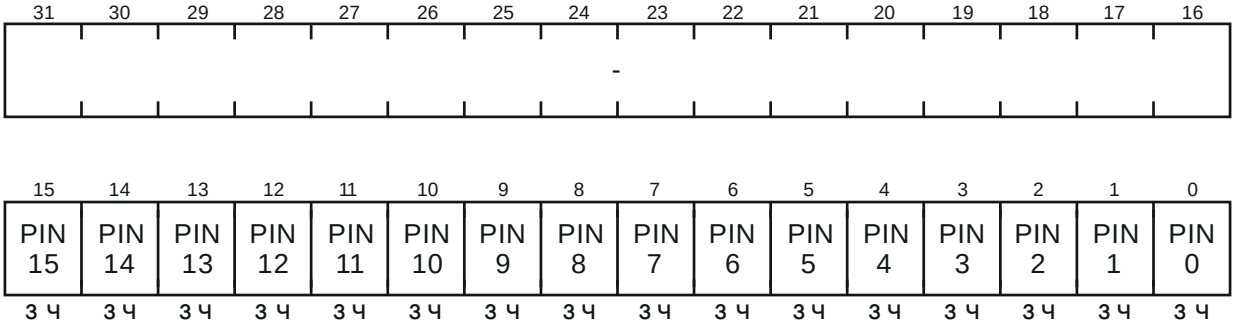


Поле	Биты	Описание		
PINn	15-0	Чтение	0	Вывод n в режиме входа/выхода общего назначения
			1	Вывод n в режиме альтернативной функции
		Запись нуля	Не выполняется	
		Запись единицы	Включает режим альтернативной функции вывода n	
–	31-16	Зарезервировано		

Примечание – Режим альтернативной функции выводов A2 – A6 (выводы JTAG) порта A, которому соответствуют биты PIN2 – PIN6, включен по умолчанию.

ALTFUNCCLR – регистр выключения альтернативной функции порта

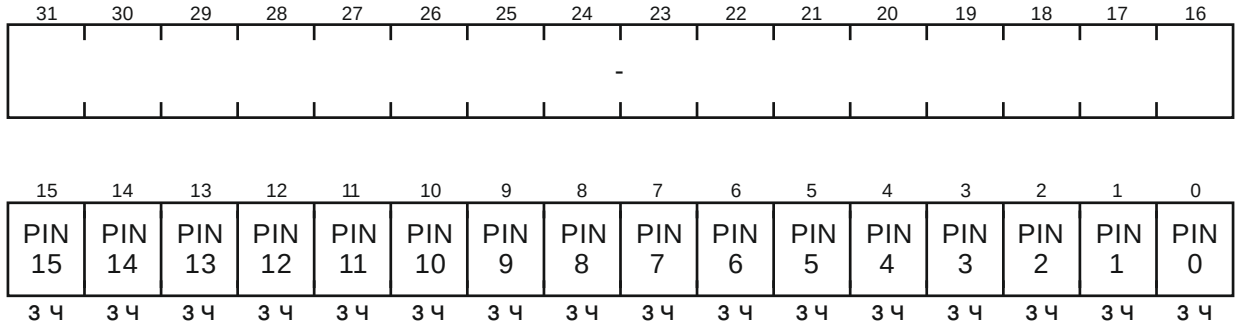
Смещение: + 38h
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	Возвращает ноль	
		Запись нуля	0	Не выполняется
		Запись единицы	1	Включает режим входа/выхода общего назначения
–	31-16	Зарезервировано		

SYNCSET – регистр включения дополнительной синхронизации входов портов

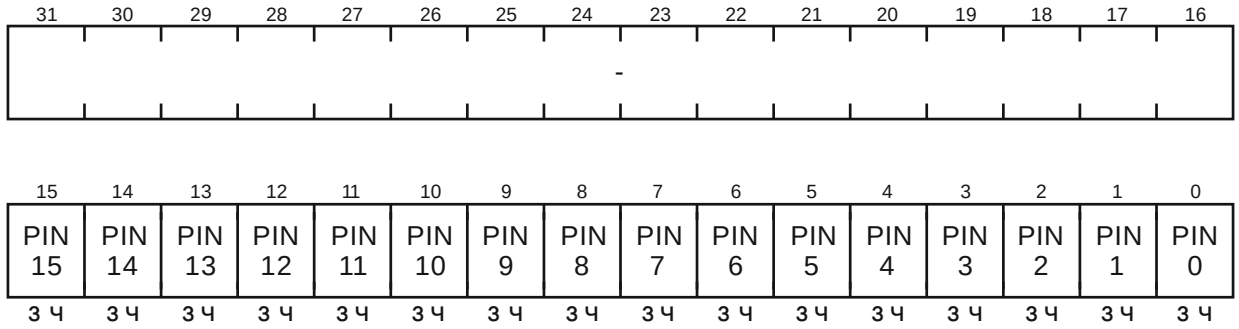
Смещение: + 44h
Сброс: 0h



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

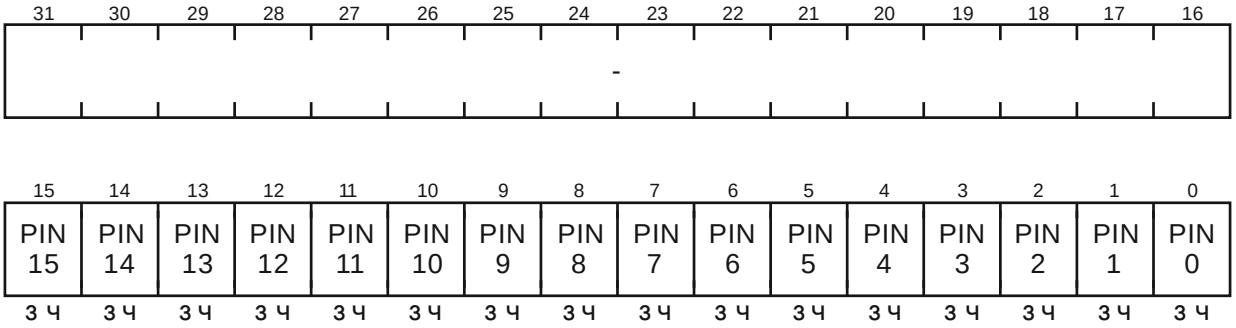
Смещение: + 48h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Отключает дополнительную схему синхронизации вывода n
–	31-16	Зарезервировано	

QUALSET – регистр включения фильтров портов

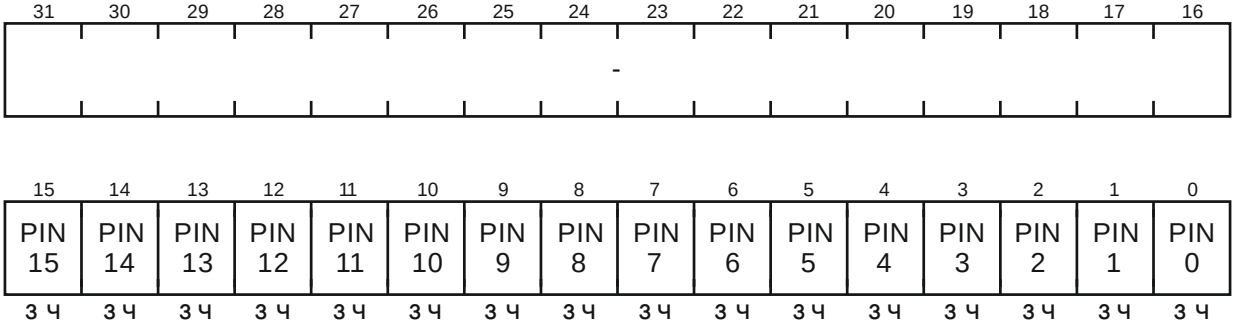
Смещение: + 4Ch
Сброс: 0h



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

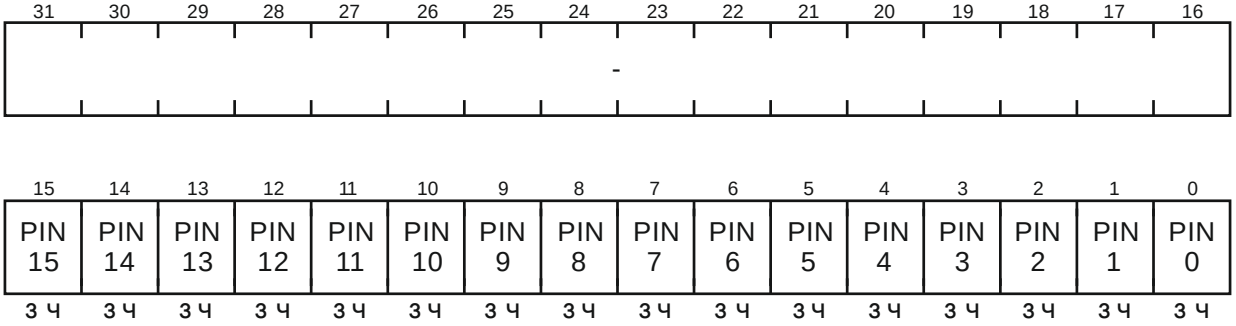
Смещение: + 50h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Отключает входной фильтр вывода n
–	31-16	Зарезервировано	

QUALMODESET – регистр режима фильтра порта

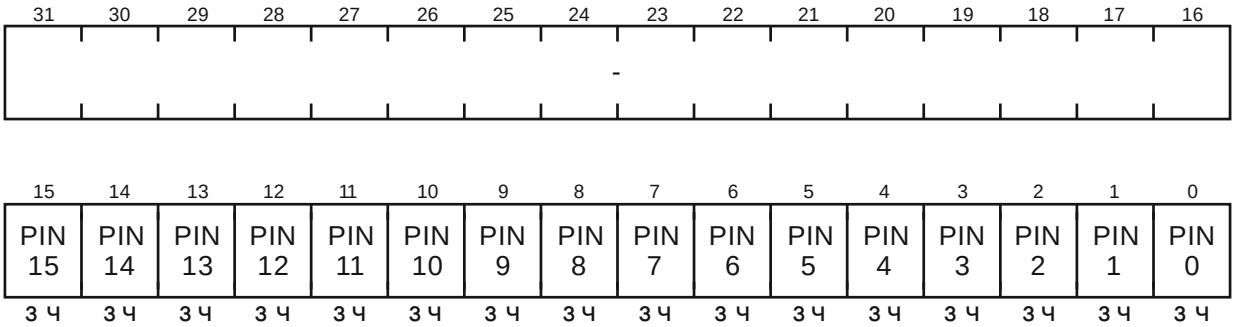
Смещение: + 54h
Сброс: 0h



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

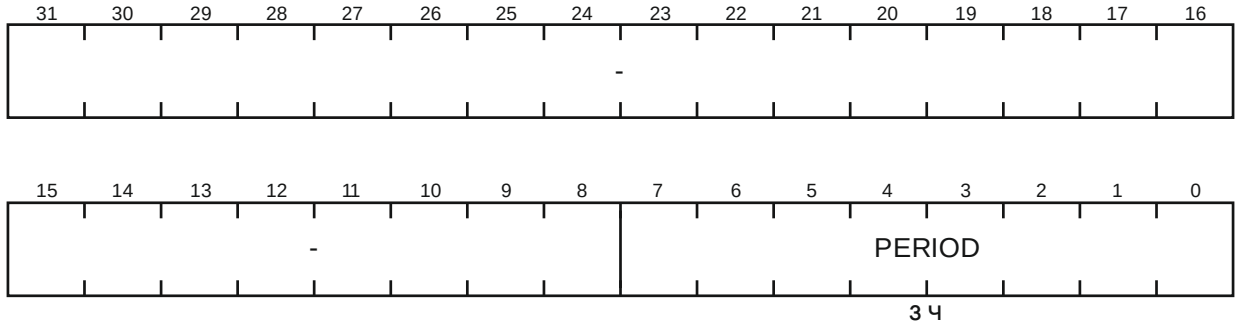
Смещение: + 58h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Сбрасывает соответствующий бит n в регистре QUALMODESET и включает режим измерения уровня сигнала на выводе n по трем отсчетам
–	31-16	Зарезервировано	

QUALSAMPLE – регистр настройки фильтра порта

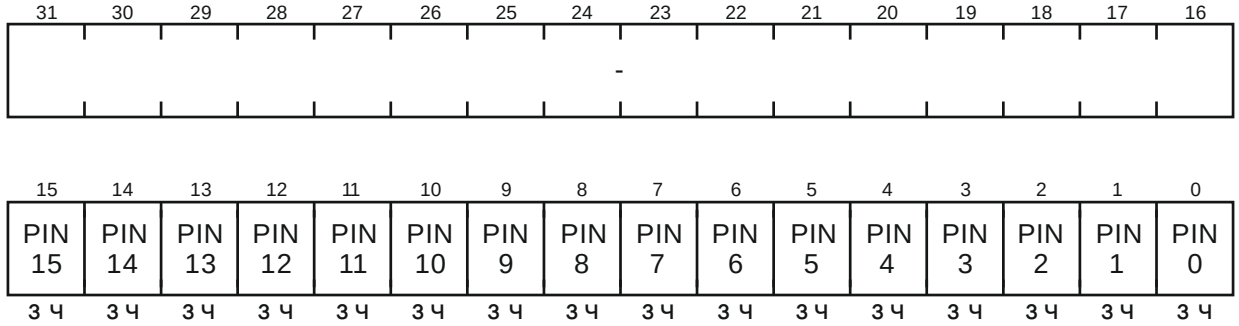
Смещение: + 5Ch
Сброс: 0h



Поле	Биты	Описание
PERIOD	7-0	Временной интервал (в тактах FCLK) между отсчетами при измерениях уровней сигналов на выводах порта. Заданное значение является единым для всех выводов порта
–	31-8	Зарезервировано

INTENSET – регистр разрешения прерываний порта

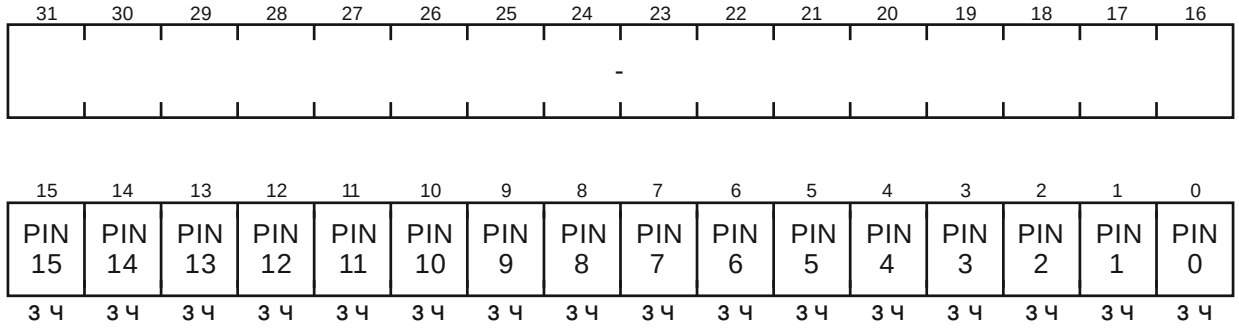
Смещение: + 60h
Сброс: 0h



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

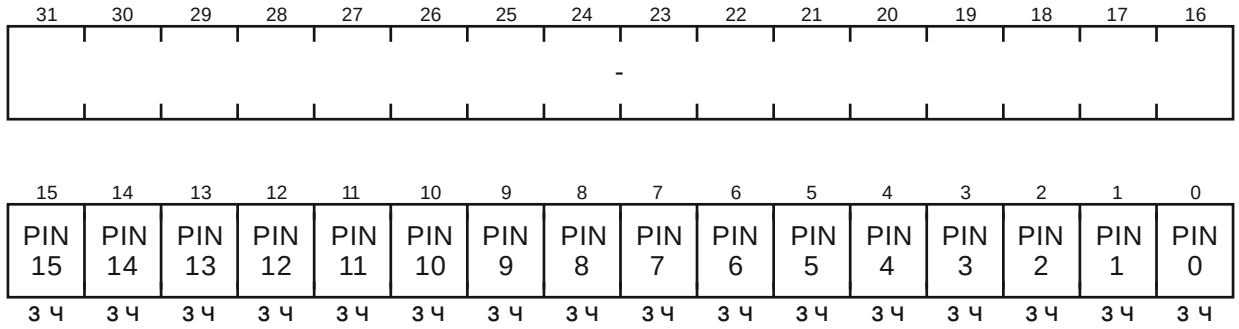
Смещение: + 64h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Всегда читаются нули
		Запись нуля	Не выполняется
		Запись единицы	Сбрасывает соответствующий бит n в регистре INTENSET и запрещает прерывания вывода n
–	31-16	Зарезервировано	

INTTYPESET – регистр типа прерываний порта

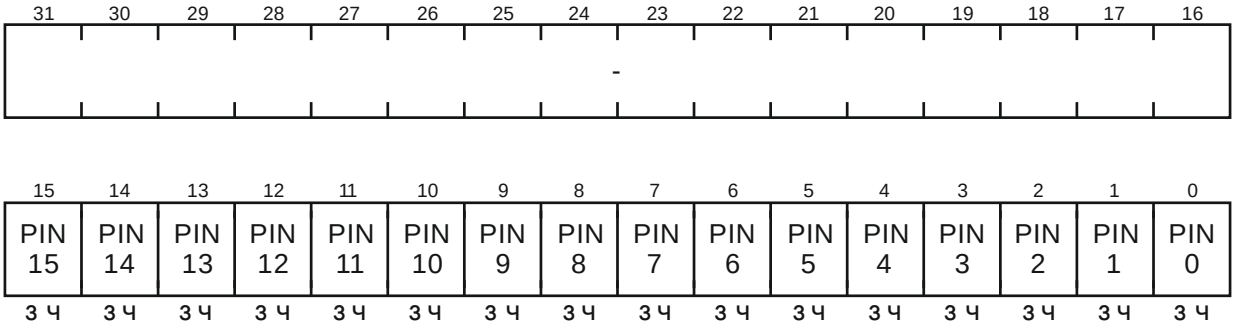
Смещение: + 68h
Сброс: 0h



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

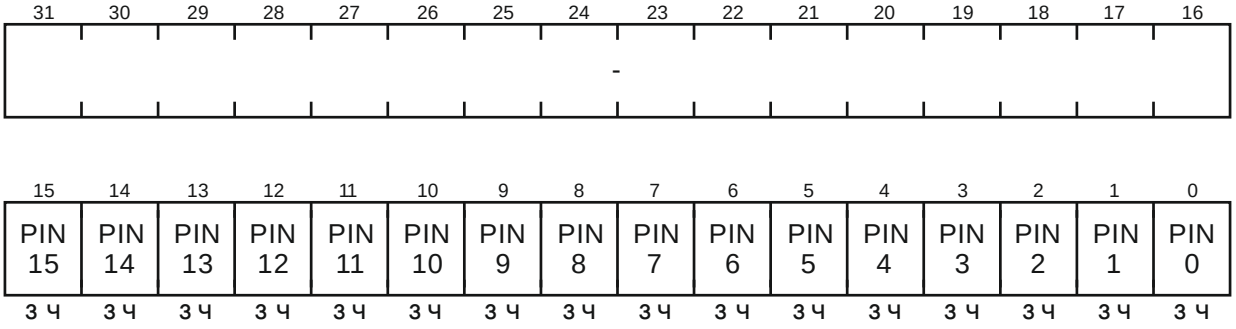
Смещение: + 6Ch
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает ноль
		Запись нуля	Нет реакции
		Запись единицы	Сбрасывает соответствующий бит n в регистре INTTYPESET и выбирает прерывания по уровню сигнала на выводе n
–	31-16	Зарезервировано	

INTPOLSET – регистр полярности события прерывания порта

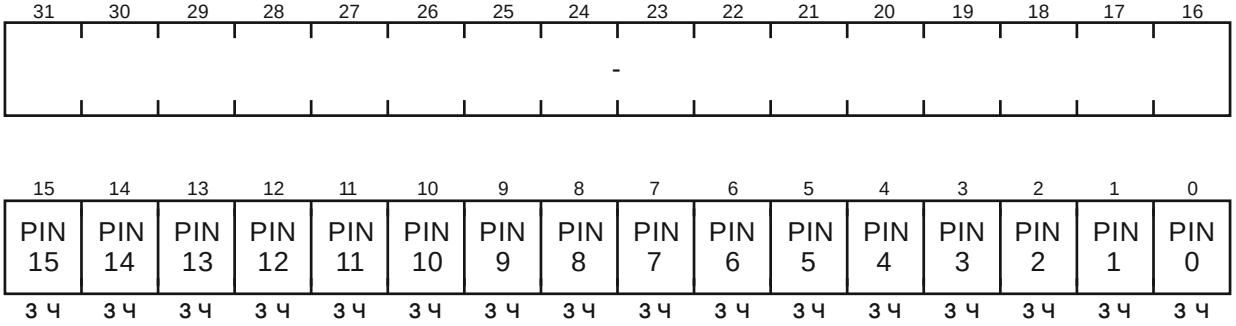
Смещение: + 70h
Сброс: 0h



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

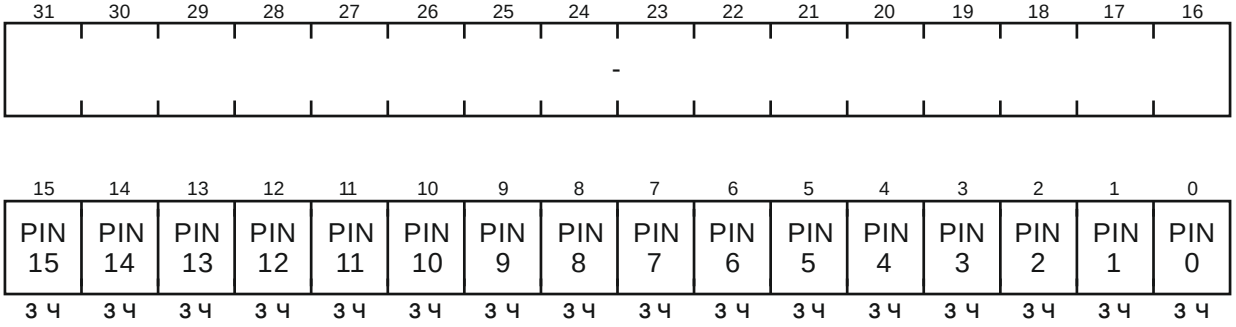
Смещение: + 74h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Сбрасывает соответствующий бит n в регистре INTPOLCLR и выбирает прерывания по низкому уровню или отрицательному фронту
–	31-16	Зарезервировано	

INTEDGESET – регистр включения прерывания по любому перепаду

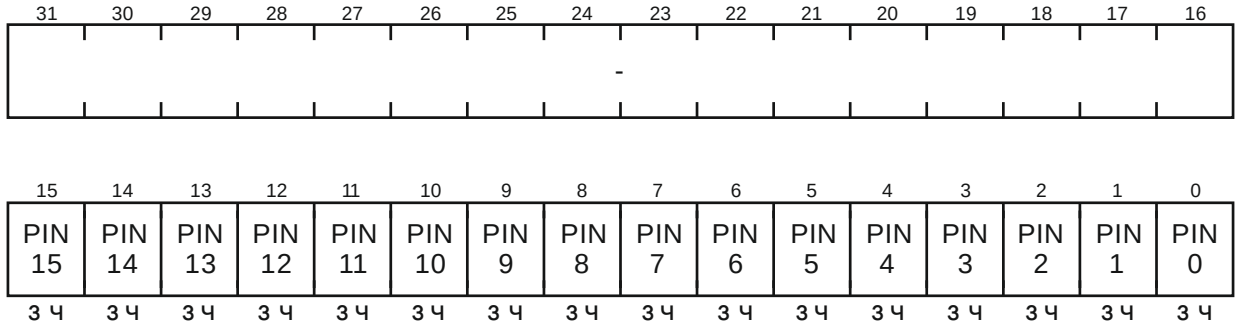
Смещение: + 78h
Сброс: 0h



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

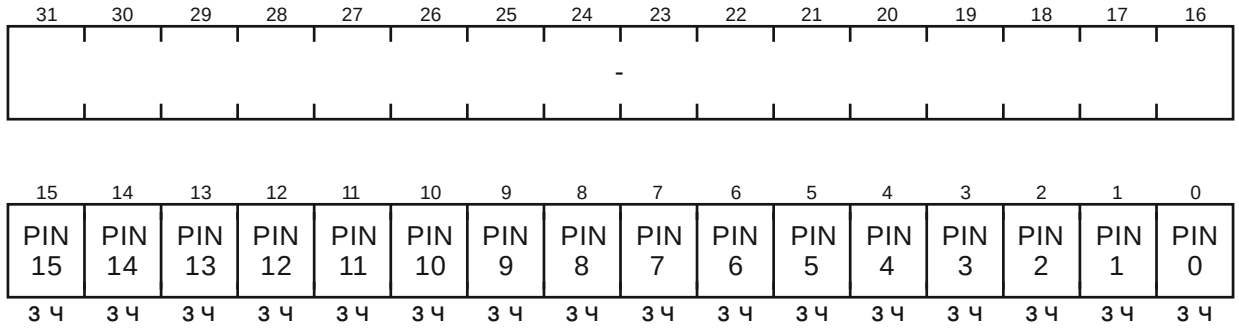
Смещение: + 7Ch
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Всегда читаются нули
		Запись нуля	Не выполняется
		Запись единиц	Сбрасывает соответствующий бит n в регистре INTEDGESET и запрещает прерывания по любому фронту сигнала на выводе n
–	31-16	Зарезервировано	

INTSTATUS – регистр состояния и сброса прерываний порта

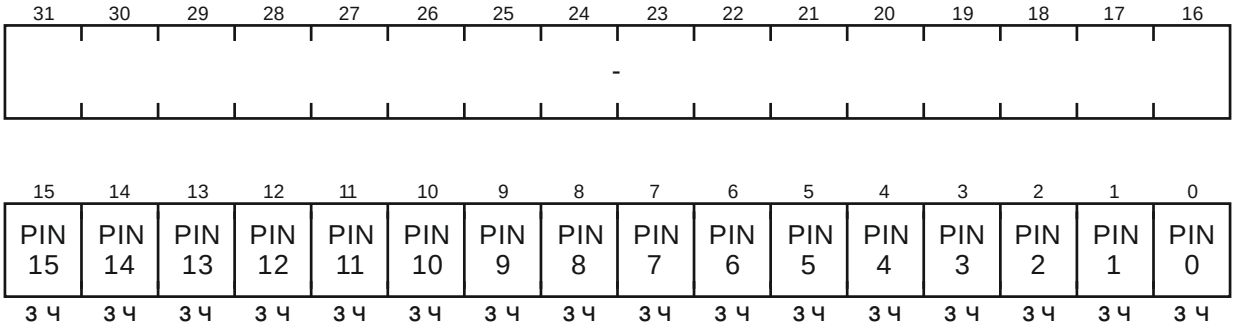
Смещение: + 80h
Сброс: 0h



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

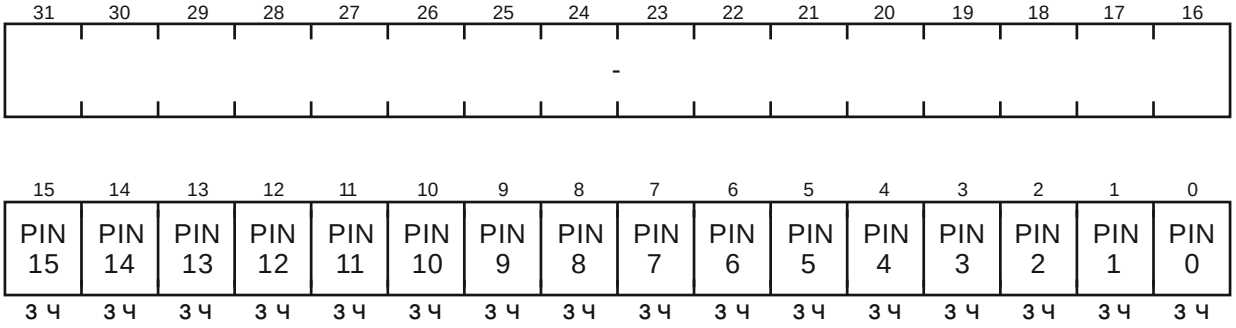
Смещение: + 84h
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Запрос DMA не генерируется
			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

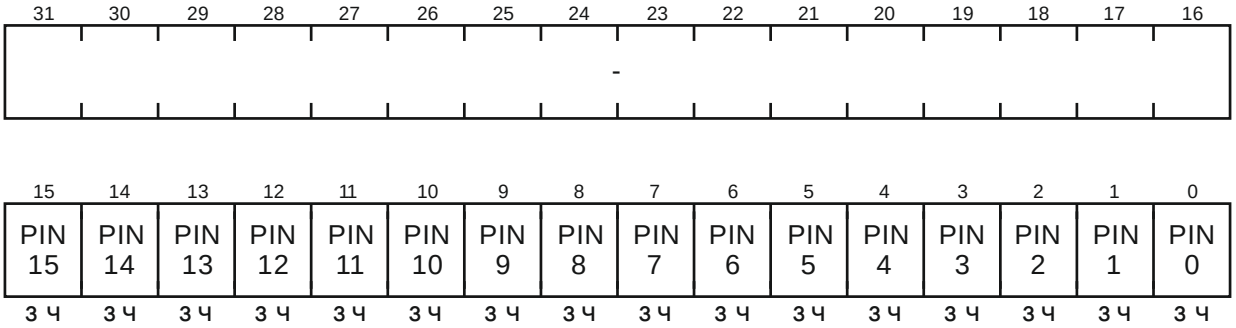
Смещение: + 88h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Сбрасывает соответствующий бит n в регистре DMAREQSET и запрещает генерирование запросов DMA по прерыванию вывода n
–	31-16	Зарезервировано	

ADCSOCSET – регистр включения генерации запросов начала преобразования АЦП по прерыванию порта

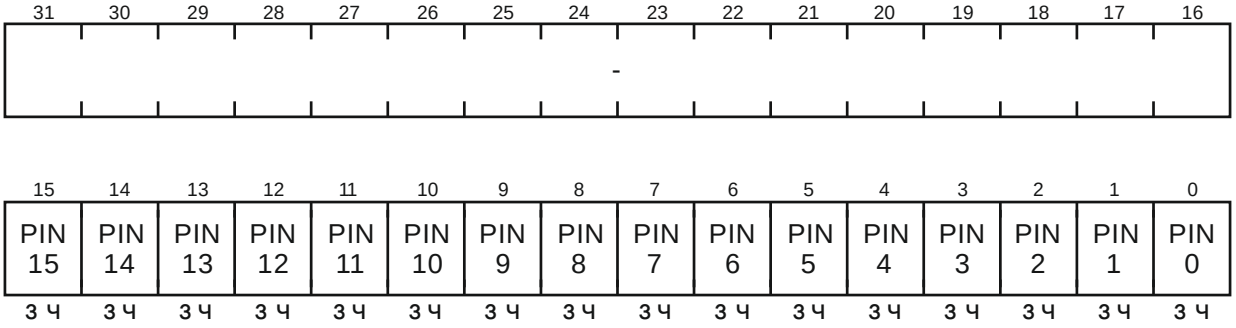
Смещение: + 8Ch
Сброс: 0h



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

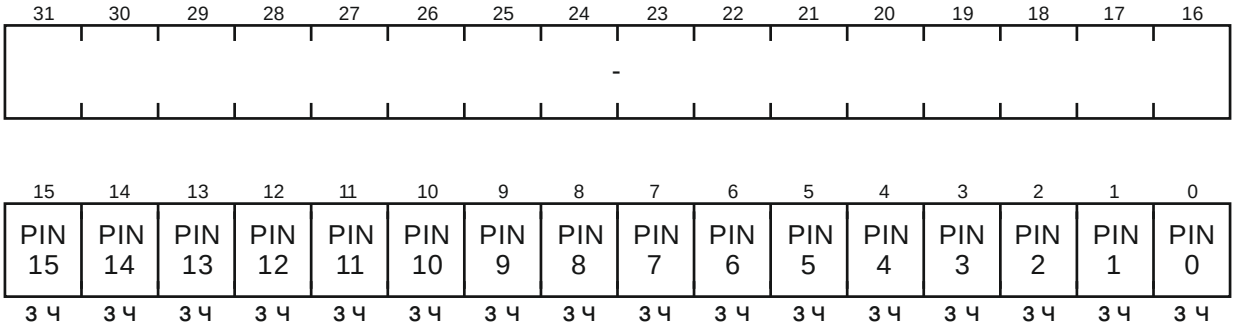
Смещение: + 90h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Сбрасывает соответствующий бит n в регистре ADCSOCSET и запрещает генерирование запросов АЦП по прерыванию вывода n
–	31-16	Зарезервировано	

RXEVSET – регистр включения генерации запросов RXEV к ядру по прерыванию порта

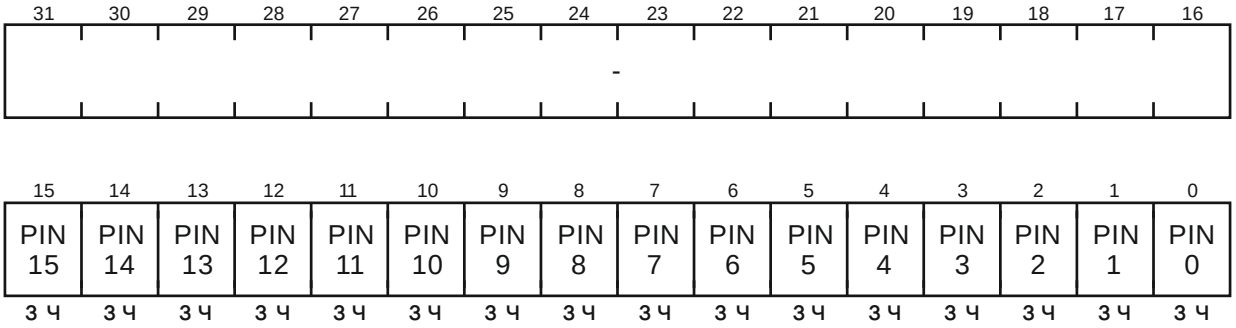
Смещение: +94h
Сброс: 0h



Поле	Биты	Описание		
PINn	15-0	Чтение	0	Запрос RXEV к ядру не генерируется
			1	Запрос RXEV к ядру генерируется по прерыванию (в том числе немаскированному)
		Запись нуля	Не выполняется	
		Запись единицы	Разрешает генерирование запросов RXEV к ядру по прерыванию вывода n (в том числе немаскированному)	
–	31-16	Зарезервировано		

RXEVCLR – регистр отключения генерации запросов RXEV к ядру по прерыванию порта

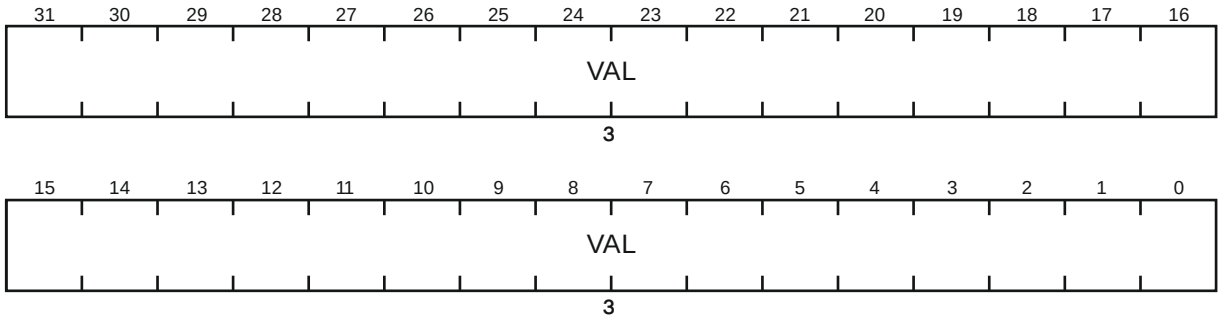
Смещение: + 98h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Сбрасывает соответствующий бит n в регистре RXEVSET и запрещает генерирование запросов RXEV к ядру по прерыванию вывода n
–	31-16	Зарезервировано	

LOCKKEY – регистр ключа блокировки

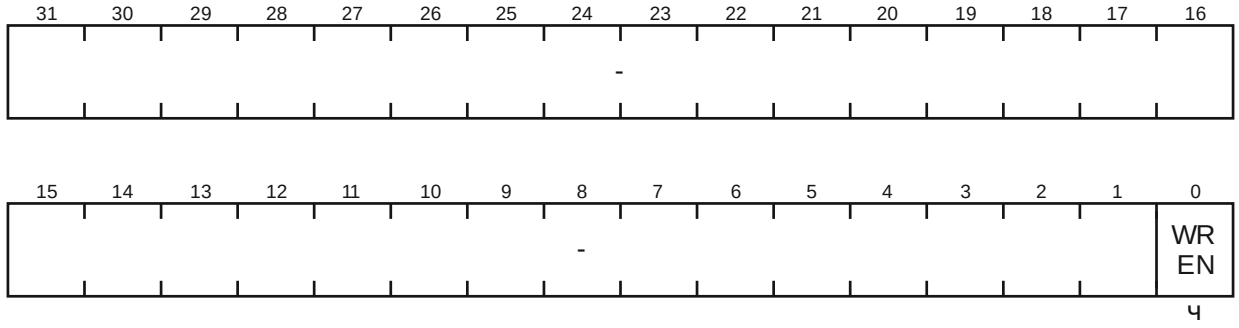
Смещение: + 9Ch
Сброс: 0h



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

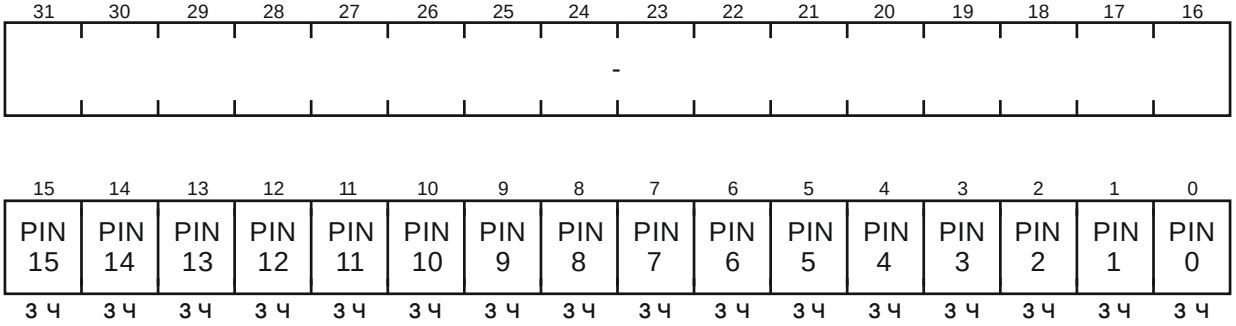
Смещение: + 9Ch
Сброс: 0h



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: 7Ch (для порта A), 0h (для порта B)

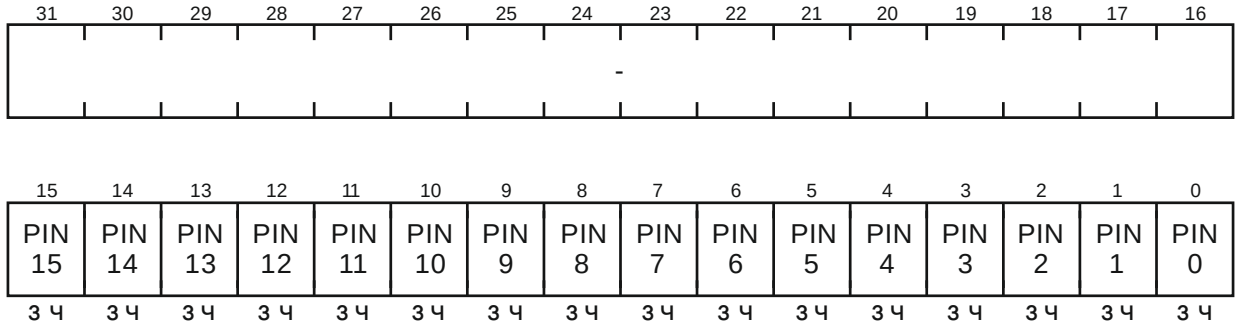


Поле	Биты	Описание		
PINn	15-0	Чтение	0	Блокировка отключена
			1	Блокировка включена
		Запись нуля	Не выполняется	
		Запись единицы (возможна после установки ключа в регистре LOCKKEY)	Блокирует изменение конфигурации вывода n. Соответствующие биты n регистров становятся недоступными для записи. Исключения: - регистр флагов прерываний INTSTATUS; - регистр временного интервала измерений QUALSAMPLE	
–	31-16	Зарезервировано		

Примечание – Выводы A2 – A6 порта A, которым соответствуют биты PIN2 – PIN6, заблокированы по умолчанию, поскольку имеют альтернативные функции, связанные с портом JTAG. Для изменения конфигурации этих выводов следует записать ключ в регистр LOCKKEY, а затем отключить блокировку посредством регистра LOCKCLR.

LOCKCLR – регистр отключения блокировки изменения конфигурации вывода

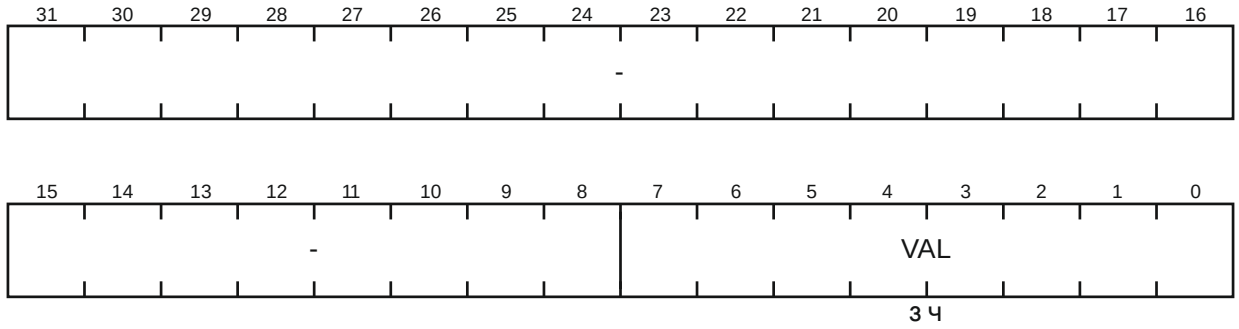
Смещение: + A4h
Сброс: 0h



Поле	Биты	Описание	
PINn	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы (возможна после установки ключа в регистре LOCKKEY)	Сбрасывает соответствующий бит n в регистре LOCKSET и разрешает изменения конфигурации вывода n
–	31-16	Зарезервировано	

MASKLB – регистр массива масок младшего байта порта

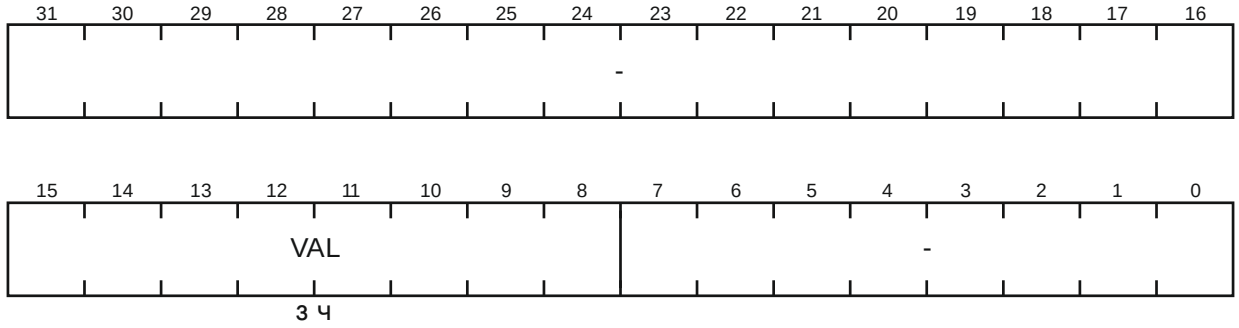
Адрес: GPIOp + MASKLB + (4*i)h (i = 0h, …, FFh)
Сброс: xxh



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: xx00h



Поле	Биты	Описание
VAL	15-8	Доступ по маске для старших восьми бит порта
–	31-16, 7-0	Зарезервировано

А.3 Регистры контроллера CAN

Базовый адрес: 4002_0000h

Смещение:

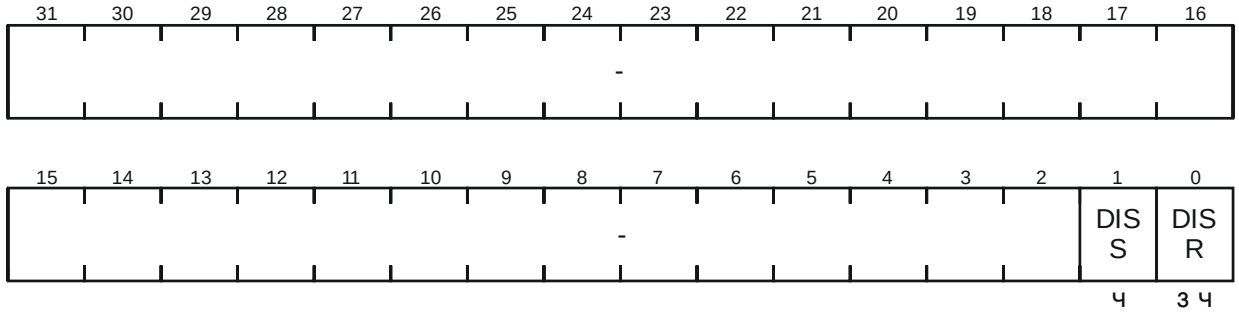
+ 100h (LIST)	Регистры свободного списка
+ 140h (MSPND)	Регистры ждущих прерываний
+ 180h (MSID)	Регистры индексов сообщений
+ 200h (Node_0)	Регистры узла 0
+ 300h (Node_1)	Регистры узла 1
+ 1000h + 20h*m (Msg_m)	Регистры объекта сообщений x

Мнемоника: Node_n;
Msg_m

Примечание – n – номер узла 0 или 1;
m – номер объекта сообщения от 0 до 63;
mo – номер объекта сообщения в шестнадцатеричном формате от 0h до 3Fh (что соответствует диапазону от 0 до 63).

CLC – регистр управления частотой

Смещение: + 00h
Сброс: 3h

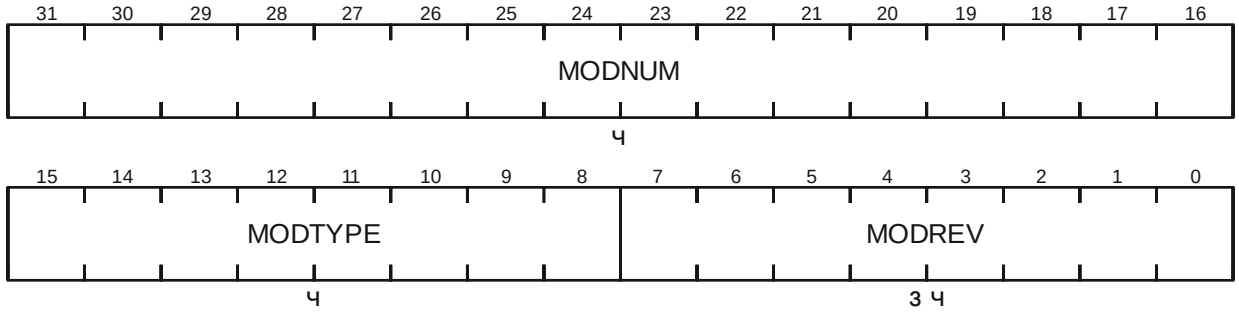


Поле	Биты	Описание	
DISS	1	Бит состояния контроллера CAN	
		0	Включен
		1	Выключен
DISR	0	Бит выключения контроллера CAN	
		0	Нет действий
		1	Запись единицы запускает механизм выключения
–	31-2	Зарезервировано	

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

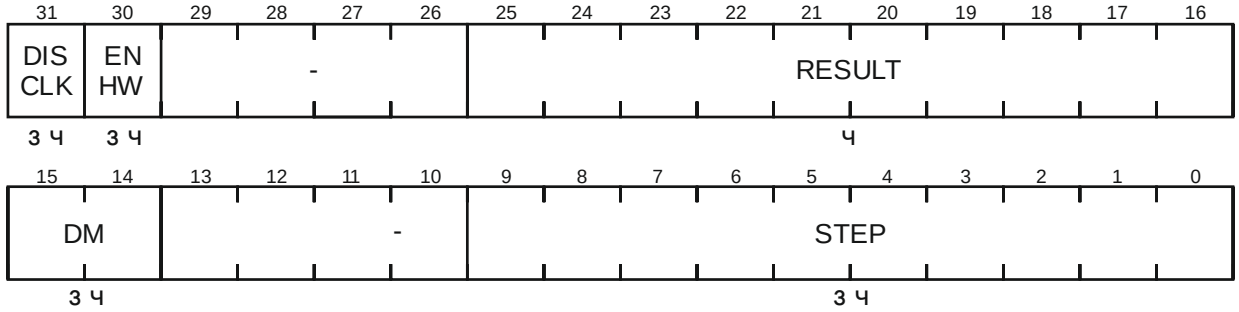
Смещение: + 08h
Сброс: 2B_C051h



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

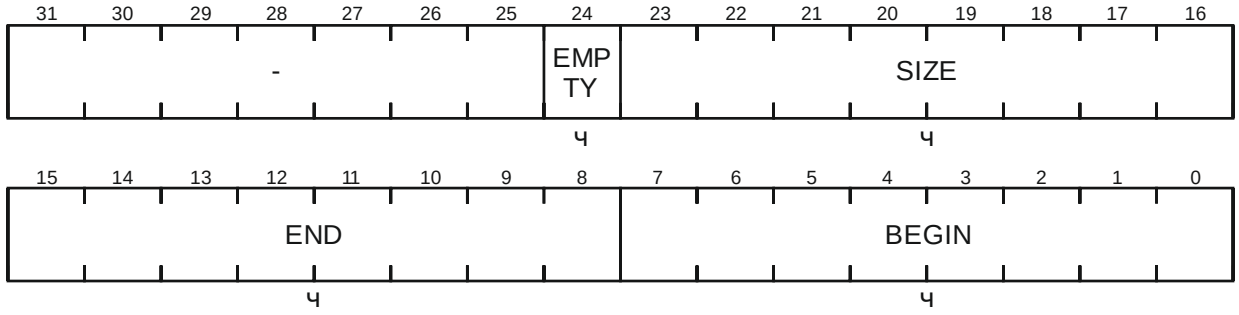


Поле	Биты	Описание	
DISCLK	31	Бит запрета внутреннего тактирования	
		0	Генерирование сигнала FCAN разрешено
		1	Генерирование сигнала FCAN запрещено
ENHW	30	Бит контроля синхронизации. Это бит аппаратно удерживается в сброшенном состоянии и не может быть установлен	
RESULT	25-16	Счетчик делителя частоты	
DM	15-14	Поле задания режима делителя частоты	
		00, 11	Счетчик выключен. Синхросигнал FOUT не генерируется. Сигнал сброса внешнего делителя в состоянии логической единицы. Поле RESULT не меняется

Поле	Биты	Описание	
DM	15-14	01	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.
		10	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.
STEP	9-0	Шаг делителя. Поле хранит значение, которое загружается в RESULT при переполнении счетчика делителя	
–	29-26, 13-10	Зарезервировано	

LIST – массив регистров свободного списка

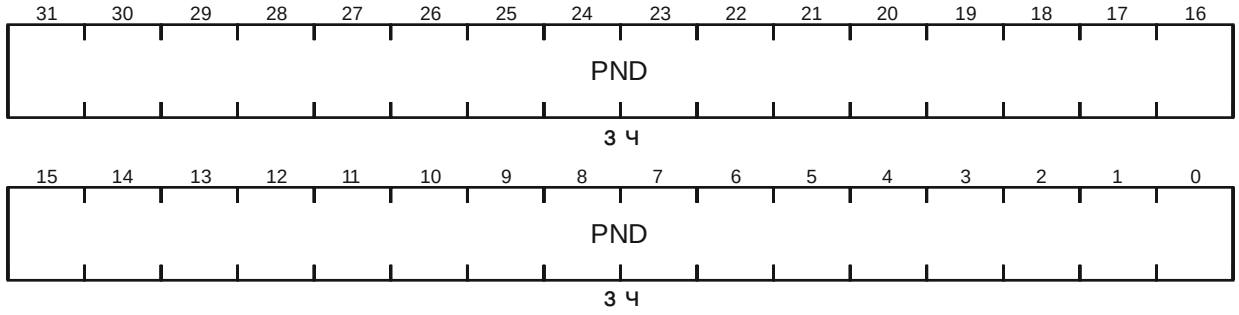
Смещение: LIST + 4*ln, где ln – номер списка от 0 до 7
Сброс: 7F_7F00h (для списка 0), 100_0000h (для списков 1 – 7)



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 FFh, согласно количеству объектов сообщений (64)	
BEGIN	7-0	Номер объекта сообщения, находящегося первым в списке. Поле может принимать значения от 00h до FFh, согласно количеству объектов сообщений	
–	31-25	Зарезервировано	

MSPND – массив регистров ждущих прерываний

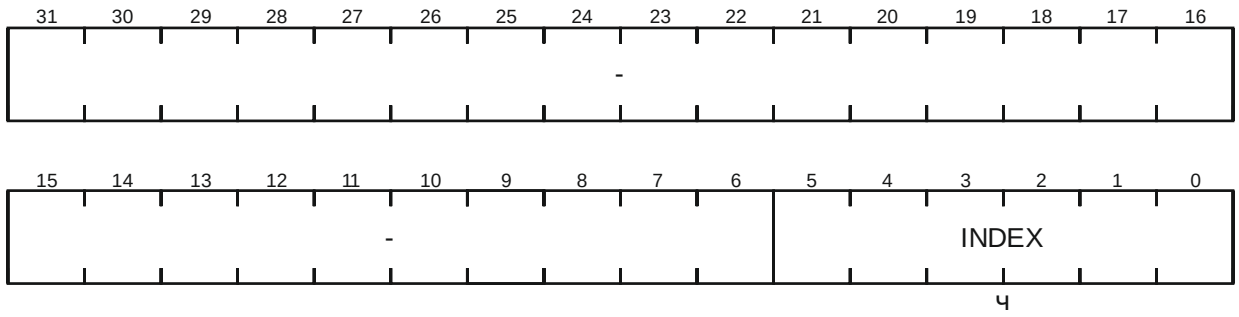
Смещение: MSPND + 4*i, где i от 0 до 3
Сброс: 0h



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

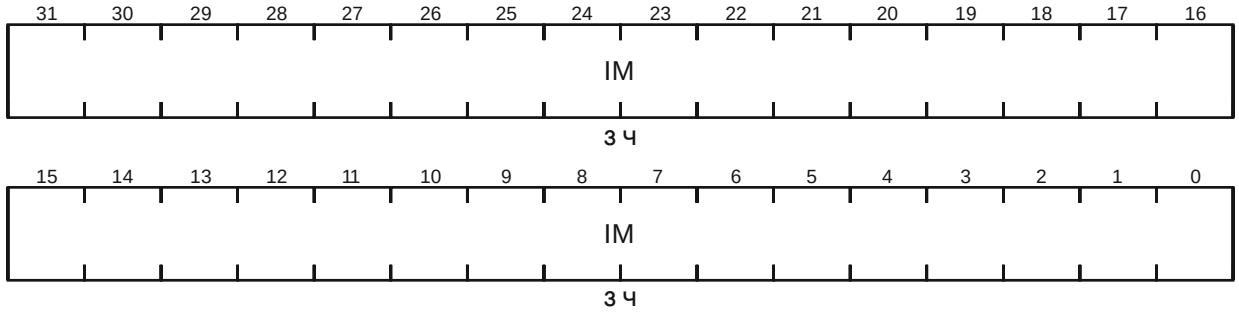
Смещение: MSID + 4* i, где i от 0 до 3
Сброс: 20h



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

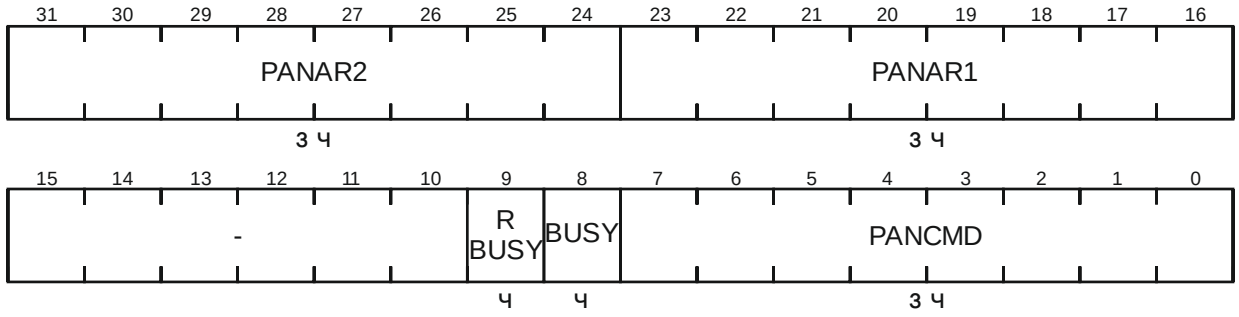
Смещение: + 1C0h
Сброс: 0h



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.3.1	
PANAR1	23-16	Панель аргумента 1, см. таблицу А.3.1	
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.3.1. После выполнения команды в это поле записывается 00h	
–	15-10	Зарезервировано	

Таблица А.3.1 – Коды команд работы со списками

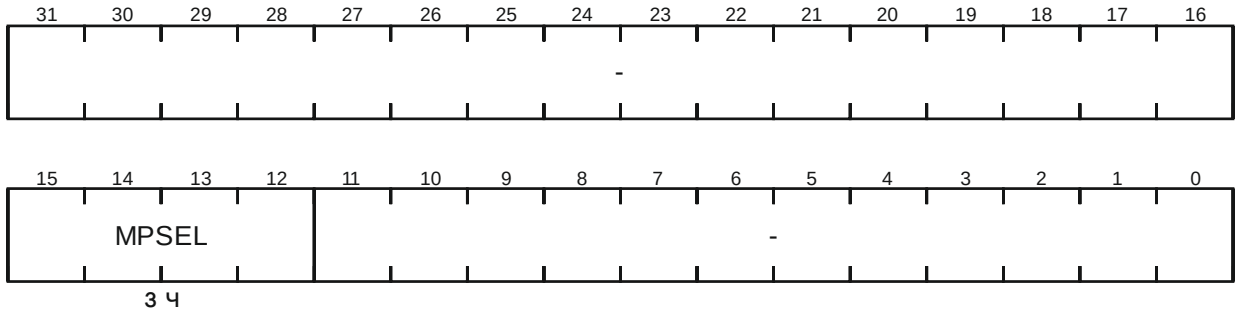
Поле PANCMD	Поле PANAR2	Поле PANAR1	Описание команды
00h	–	–	Нет операции. Никаких действий не выполняется
01h	Результат: бит 7 – ошибка, бит 6 – не определен	–	Инициализация списков. Запуск инициализации для очистки битовых полей CTRL и LIST всех объектов сообщений. Регистры LIST0 – LIST8 устанавливаются 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

Окончание таблицы А.3.1

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

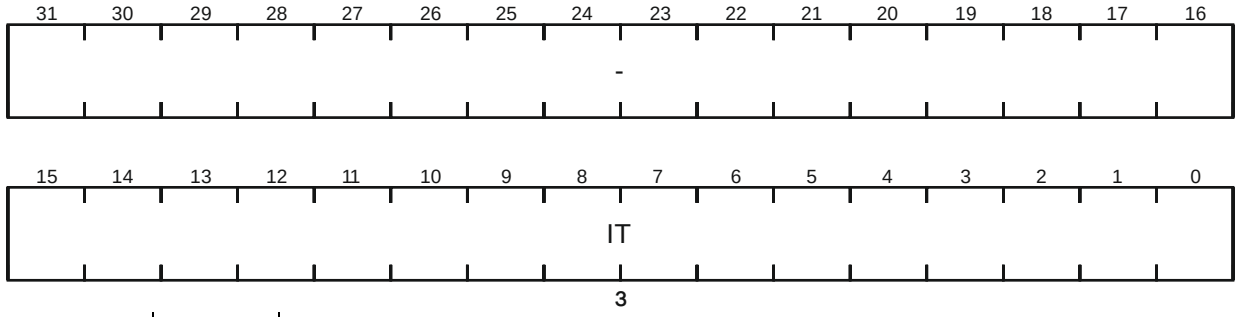
Смещение: + 1C8h
Сброс: 0h



Поле	Биты	Описание
MPSEL	15-12	Поле задания позиции ждущего бита сообщения после приема/ передачи сообщения
–	31-16, 11-0	Зарезервировано

MITR – регистр прерываний

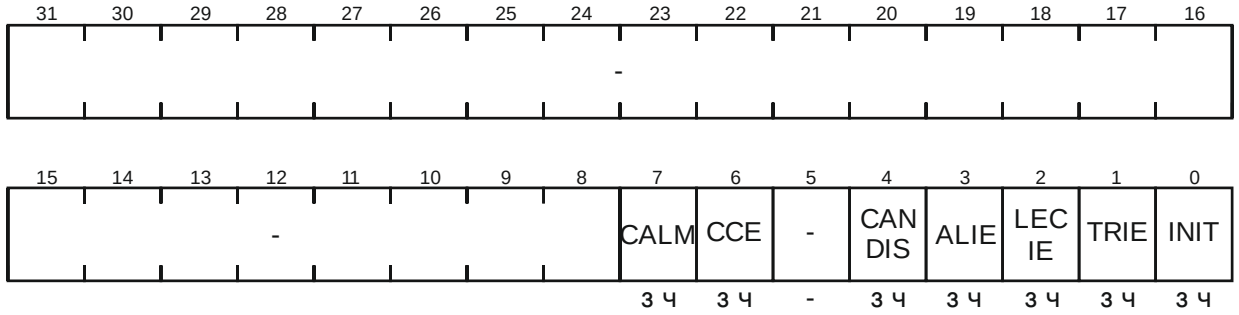
Смещение: + 1CCh
Сброс: 0h



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

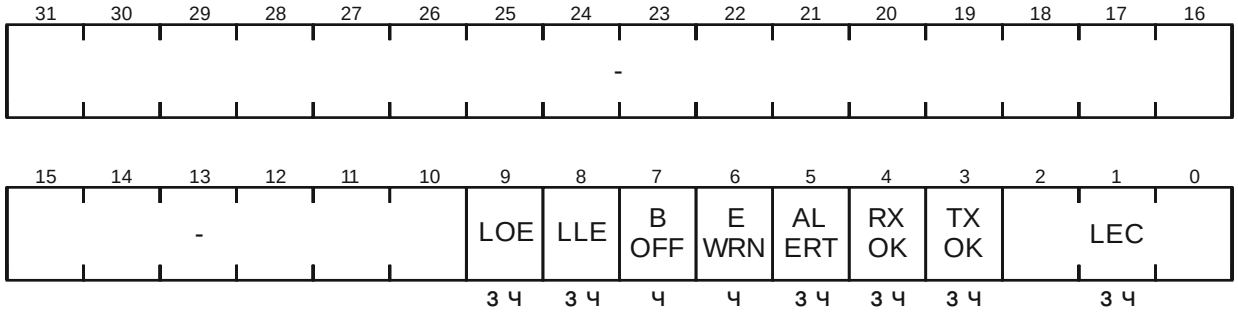


Поле	Биты	Описание	
CALM	7	Бит включения режима анализа узла	
		0	Режим выключен
		1	Установка бита включает режим анализа узла. В этом режиме сообщения могут только приниматься, бит подтверждения не посылается после успешного приема сообщения, флаг активной ошибки посылается рецессивным вместо доминантного. 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

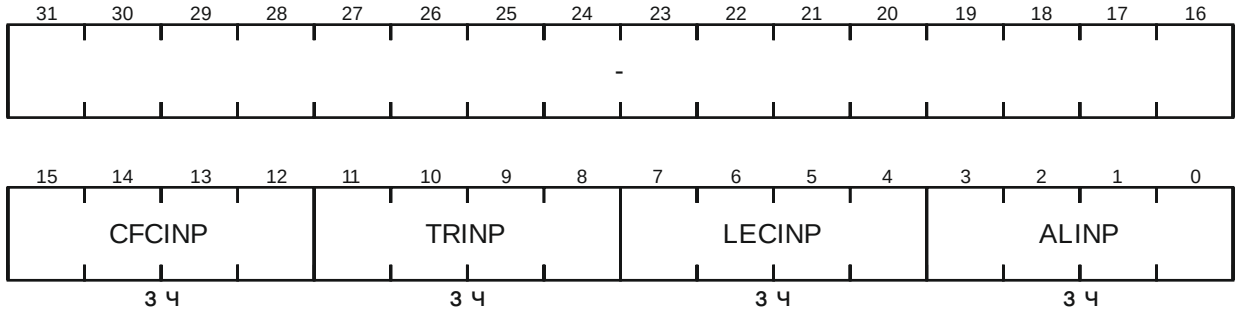


Поле	Биты	Описание	
LOE	9	Флаг ошибки номера списка	
		0	Ошибок не обнаружено
		1	Обнаружена ошибка при фильтрации принимаемого сообщения. В регистре MOSTAT объекта сообщения обнаружен неверный номер списка
		Бит должен сбрасываться программно записью нуля	
LLE	8	Флаг ошибки списка	
		0	Ошибок не обнаружено
		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 достигнул
		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 (заполнения, 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	Пользовательский код
		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

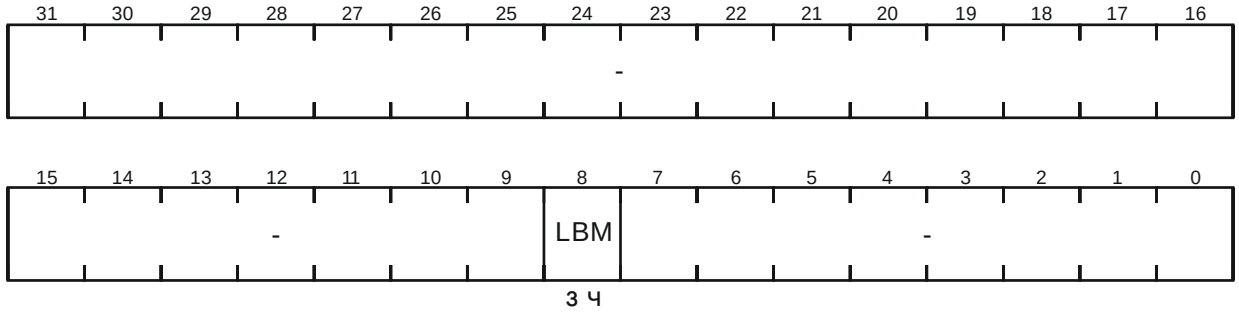


Поле	Биты	Описание
CFCINP	15-12	Указатель для прерывания при переполнении счетчика фреймов узла
TRINP	11-8	Указатель для прерывания по окончании передачи/приема сообщения
LECINP	7-4	Указатель для прерывания при записи кода последней ошибки
ALINP	3-0	Указатель для прерывания ALERT
–	31-16	Зарезервировано

Примечание – Каждый из указателей задает номер одной из 16 линий прерываний. Значение 0h соответствует нулевой линии, значение Fh – линии 15.

NPCR – регистр управления портом узла

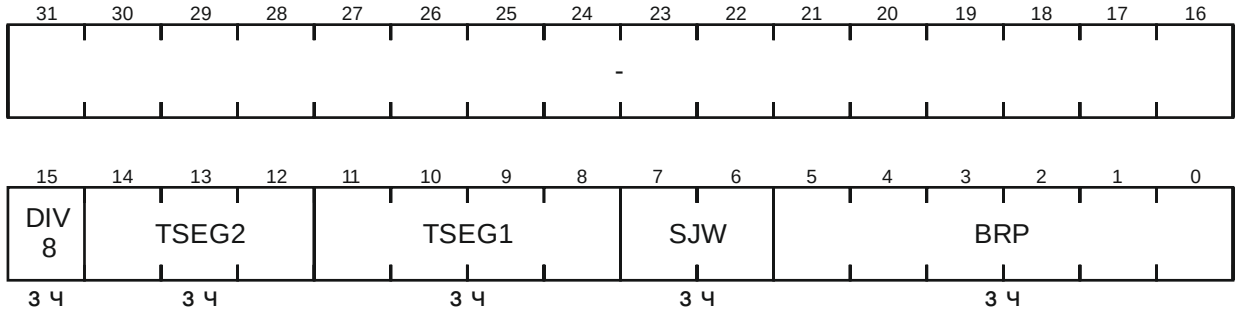
Смещение: Node_n + 0Ch
Сброс: 0h



Поле	Биты	Описание	
LBM	8	Бит включения режима обратной петли (Loop-Back)	
		0	Выключен
		1	Включен. 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

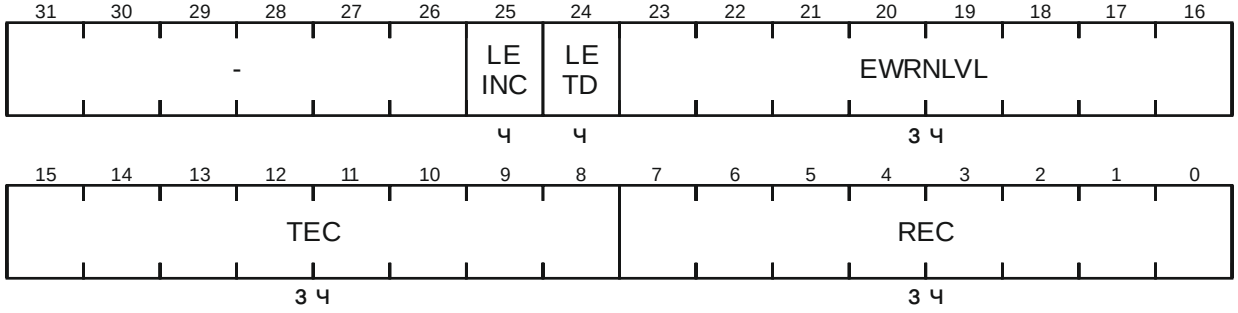
Смещение: Node_n + 10h
Сброс: 0h



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

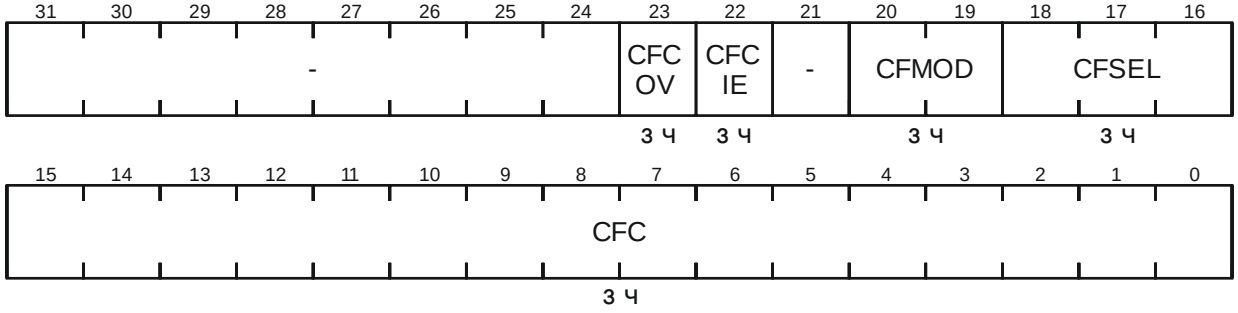
Смещение: Node_n + 14h
Сброс: 60_0000h



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

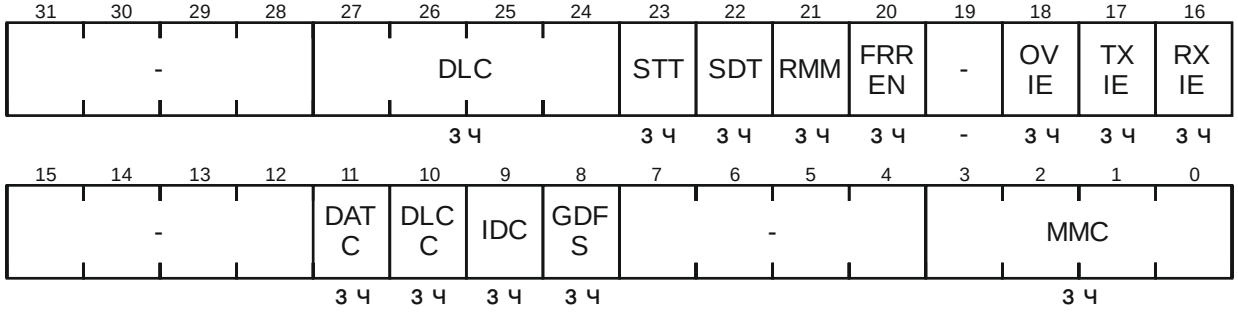
Смещение: Node_n + 18h
Сброс: 0h



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_m + 00h
Сброс: 0h



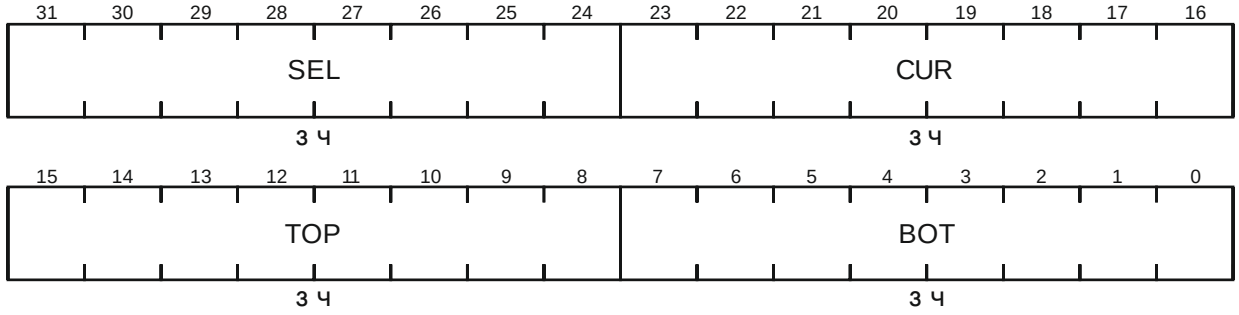
Поле	Биты	Описание	
DLC	27–24	Код длины данных. Показывает количество байт данных, находящихся в объекте сообщения. Диапазон – значение от 0 до 8. Если значение поля DLC больше 8, это автоматически указывает на 8 байт. Значение поля DLC полученного сообщения сохраняется таким, каким было получено	
STT	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	Выключен. Идентификатор, бит IDE и поле DLC объекта сообщения m остаются без изменений до получения корректного фрейма удаленного запроса
		1	Включен. 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 16 линий прерываний. Если объект сообщения m является объектом передачи FIFO, то поле RXINP (регистр MOIPR) указывает на одну из 16 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 16 линий прерываний
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 16 линий прерываний
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

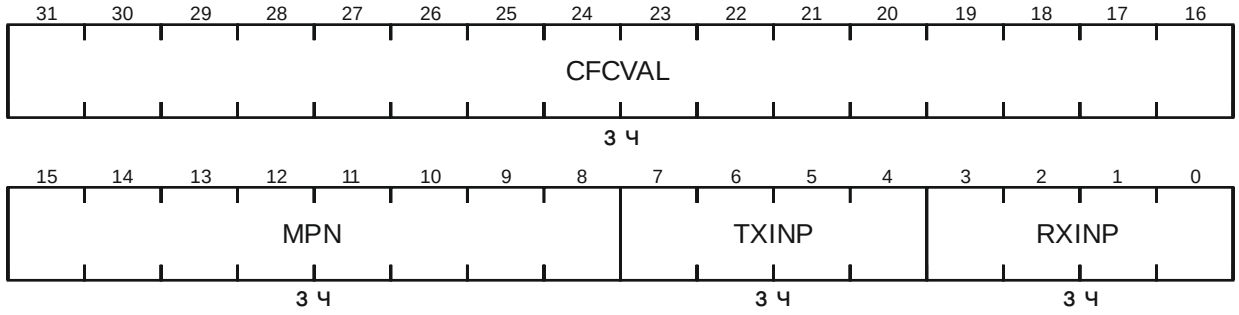
Смещение: Msg_m + 04h
Сброс: 0h



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

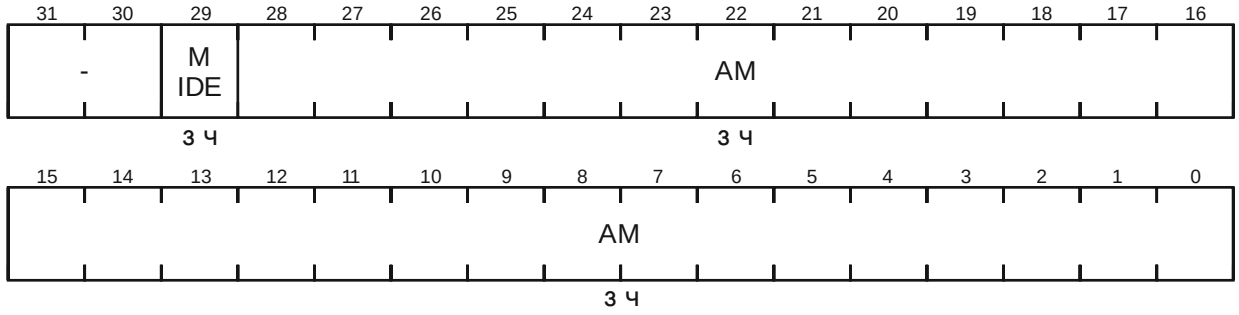
Смещение: Msg_m + 08h
Сброс: 0h



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 16 линий прерываний с номерами от 0 до 15. Значение 0000b, записанное в TXINP, выбирает нулевую линию прерываний, 0001b – первую, 0010b – вторую и т. д. Дополнительно бит TXINP используется для выбора позиции ждущего бита объекта сообщения m
RXINP	3-0	Указатель линии прерываний для прерывания после приема. Всего доступно 16 линий прерываний с номерами от 0 до 15. Значение 0000b, записанное в TXINP, выбирает нулевую линию прерываний, 0001b – первую, 0010b – вторую и т. д. Дополнительно бит RXINP используется для выбора позиции ждущего бита объекта сообщения m

MOAMR – регистр маски объекта сообщения

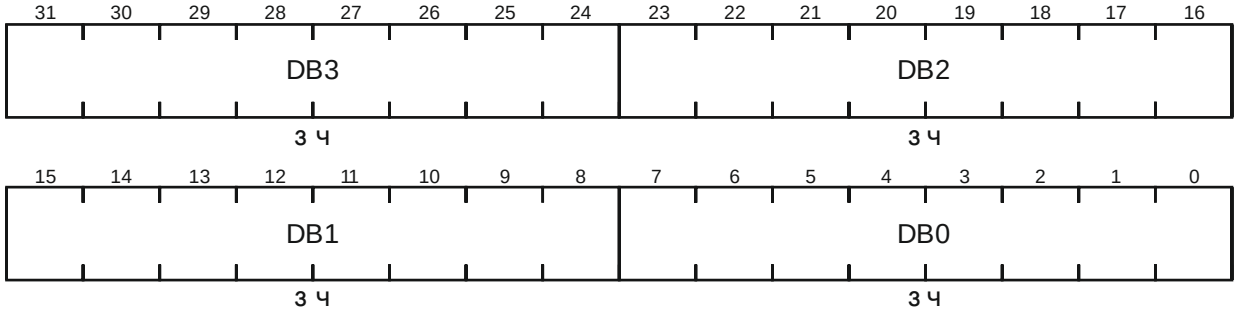
Смещение: Msg_m + 0Ch
Сброс: 3FFF_FFFFh



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

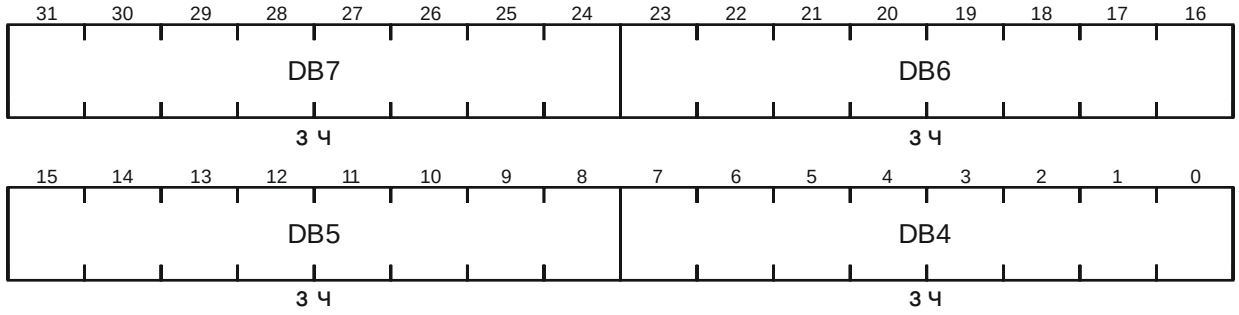
Смещение: Msg_m + 10h
Сброс: 0h



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

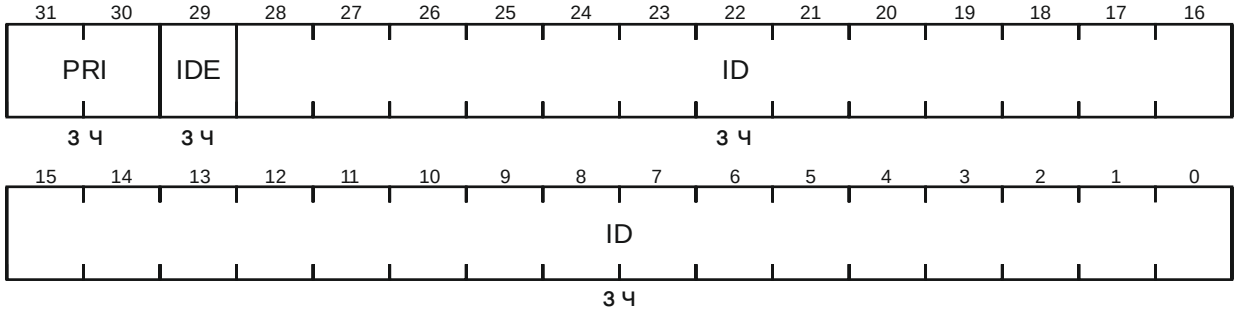
Смещение: Msg_m + 14h
Сброс: 0h



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_m + 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

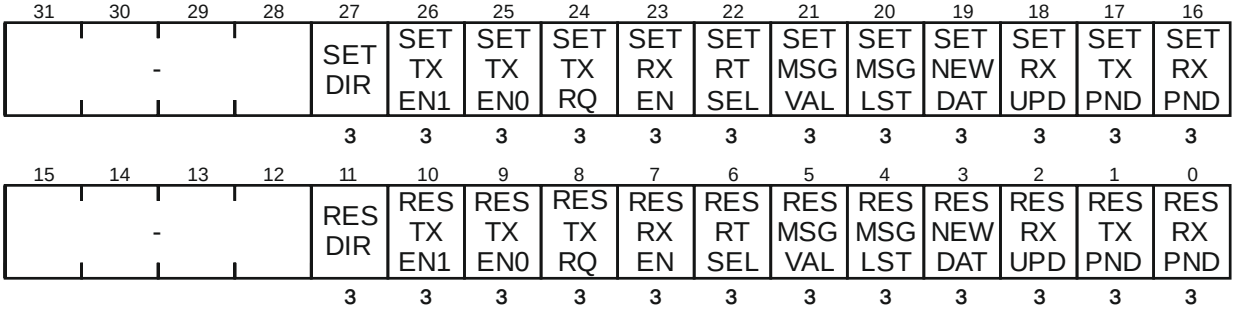
Поле	Биты	Описание	
PRI	31-30	10	Фильтрация в зависимости от значения идентификатора. Объект сообщения m получает приоритет на передачу сообщения только в случае, если в списке нет других объектов сообщений с «Идентификатор + IDE + DIR» более высокого приоритета (согласно правилам арбитража в таблице А.3.2)
		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.3.2 – 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_m + 1Ch
Сброс: 0h



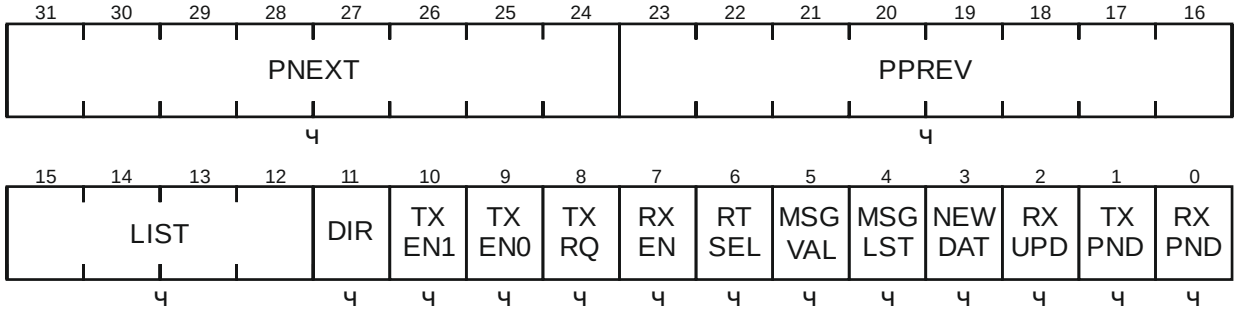
Поле	Биты	Описание
RESRXPND, SETRXPND	0, 16	Сброс/установка бита RXPND
RESTXPND, SETTXPND	1, 17	Сброс/установка бита TXPND
RESRXUPD, SETRXUPD	2, 18	Сброс/установка бита RXUPD
RESNEWDAT, SETNEWDAT	3, 19	Сброс/установка бита NEWDAT
RESMSGLST, SETMSGLST	4, 20	Сброс/установка бита MSGLST
RESMSGVAL, SETMSGVAL	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_m + 1Ch
Сброс:
- 0100_0000h для объекта сообщения 0;
- [mo + 1][mo - 1][0000]h для объектов сообщений с номерами от 1 до 62, (соответственно, mo от 01h до 3Eh);
- FFFE_0000h для объекта сообщения 255;



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

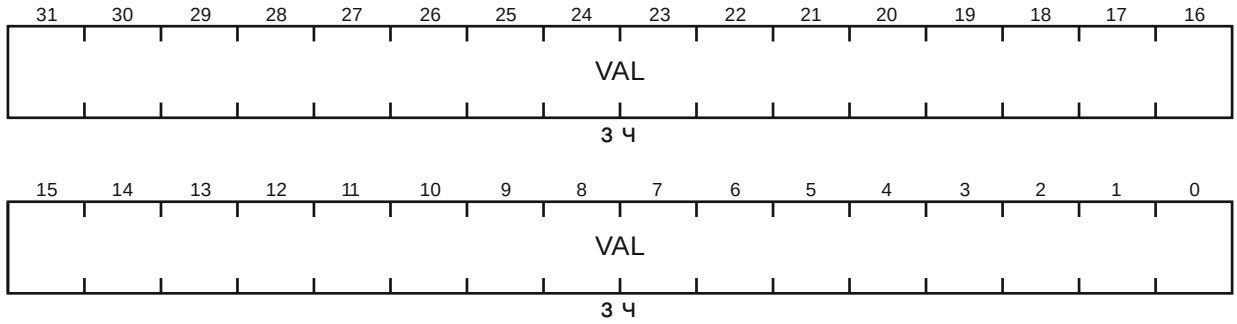
A.4 Регистры контроллера Flash-памяти

Базовый адрес: 4003_0000h

Смещение: + 04h (DATA) Регистры данных

ADDR – регистр адреса Flash-памяти

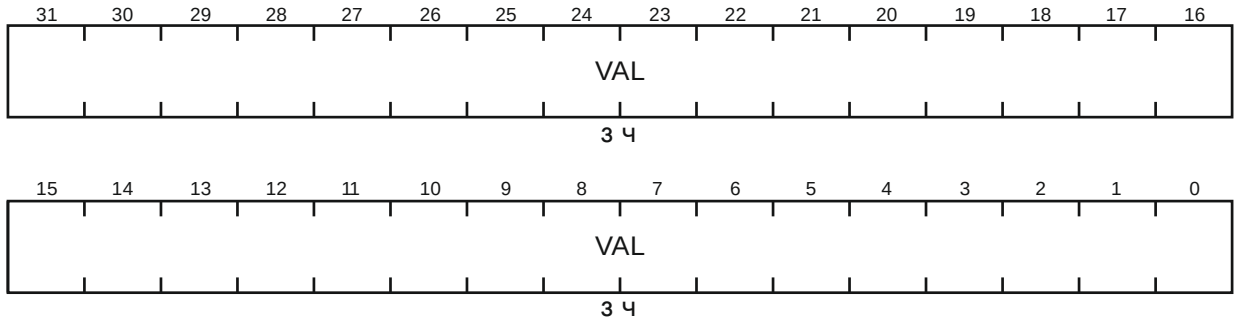
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Адрес, используемый при командах записи, чтения и постраничного стирания.

DATA – массив регистров данных Flash-памяти

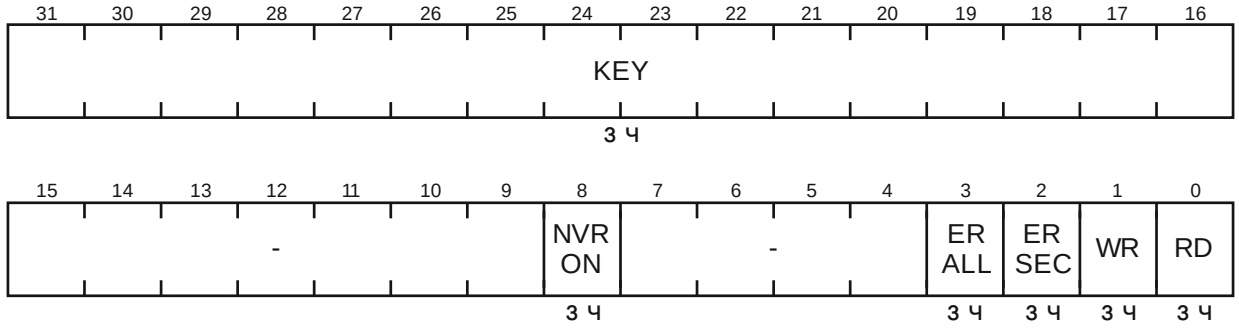
Смещение: DATA + (4*d)h, где d = 0 или 1
Сброс: FFFF_FFFFh



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

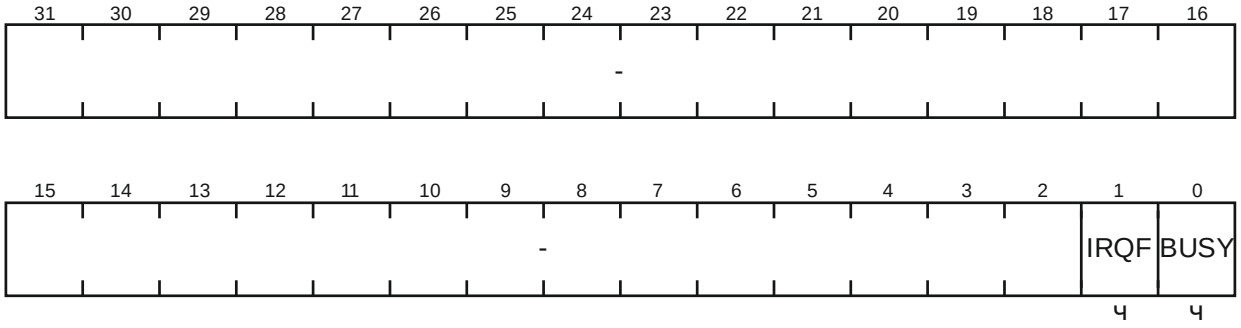
Смещение: + 24h
Сброс: DEC0_0000h



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
ERALL	3	Бит активации команды полного стирания области
ERSEC	2	Бит активации команды стирания страницы области. Адрес страницы вычисляется на основе значения регистра ADDR.
WR	1	Бит активации команды записи данных DATA0, DATA1 по адресу ADDR в области
RD	0	Бит активации команды чтения данных в DATA0, DATA1 по адресу ADDR в области
–	15-9, 7-4	Зарезервировано

STAT – регистр статуса Flash-памяти

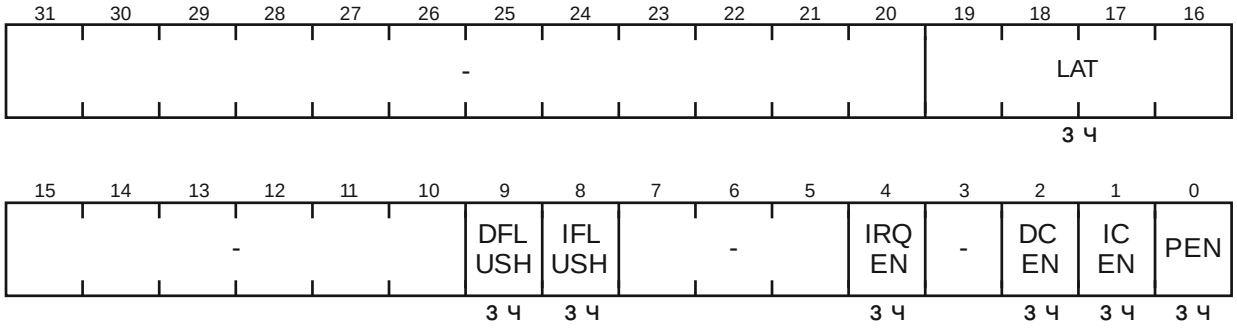
Смещение: + 28h
Сброс: 0h



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

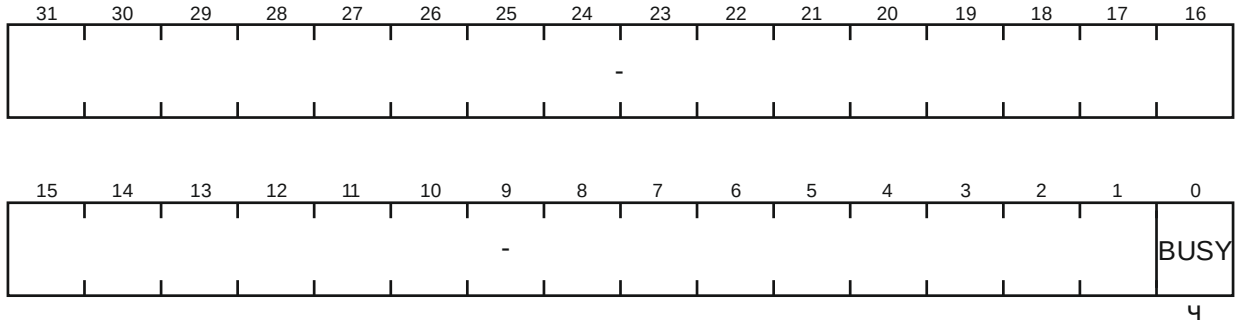
Смещение: + 2Ch
Сброс: 1_0000h



Поле	Биты	Описание	
LAT	19-16	Поле задания количества дополнительных тактов ожидания при чтении из Flash-памяти	
DFLUSH	9	Бит запуска очистки кэш данных	
		0	Нет реакции
		1	Запуск очистки
IFLUSH	8	Бит запуска очистки кэш инструкций	
		0	Нет реакции
		1	Запуск очистки
IRQEN	4	Бит управления прерыванием по завершении выполнения команды регистра CMD	
		0	Нет реакции
		1	Прерывание разрешено
DCEN	2	Бит управления кэш данных	
		0	Выключено
		1	Включено
ICEN	1	Бит управления кэш инструкций	
		0	Выключено
		1	Включено
PEN	0	Бит управления предвыборкой	
		0	Выключено
		1	Включено
–	31-20, 15-10, 7-5, 3	Зарезервировано	

ICSTAT – регистр статуса кэш инструкций

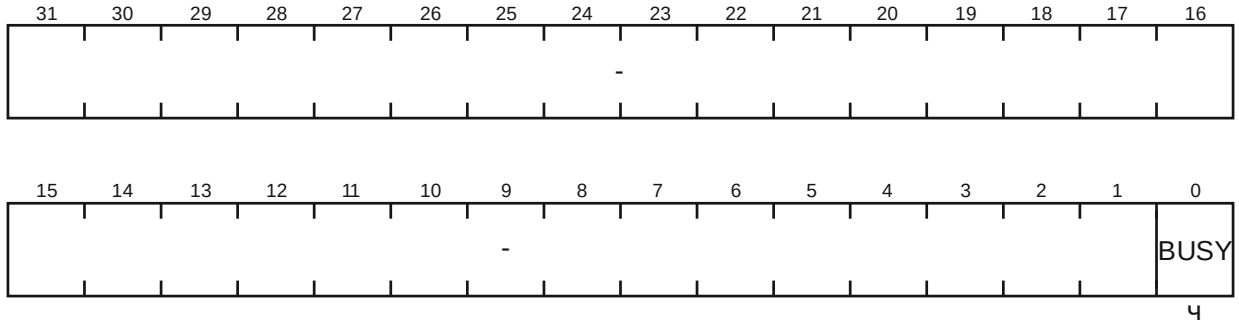
Смещение: + 34h
Сброс: 0h



Поле	Биты	Описание
BUSY	0	Флаг устанавливается при запуске команды очистки кэш данных, сбрасывается после их окончания
–	31-1	Зарезервировано

DCSTAT – регистр статуса кэш данных

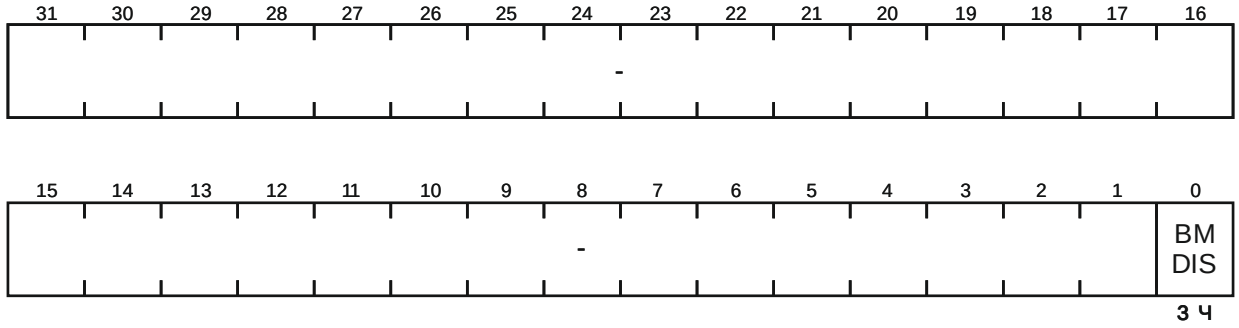
Смещение: + 38h
Сброс: 0h



Поле	Биты	Описание
BUSY	0	Флаг устанавливается при запуске команды очистки кэш данных, сбрасывается после их окончания
–	31-1	Зарезервировано

BDIS – регистр управления загрузкой

Смещение: + 78h
Сброс: 0h



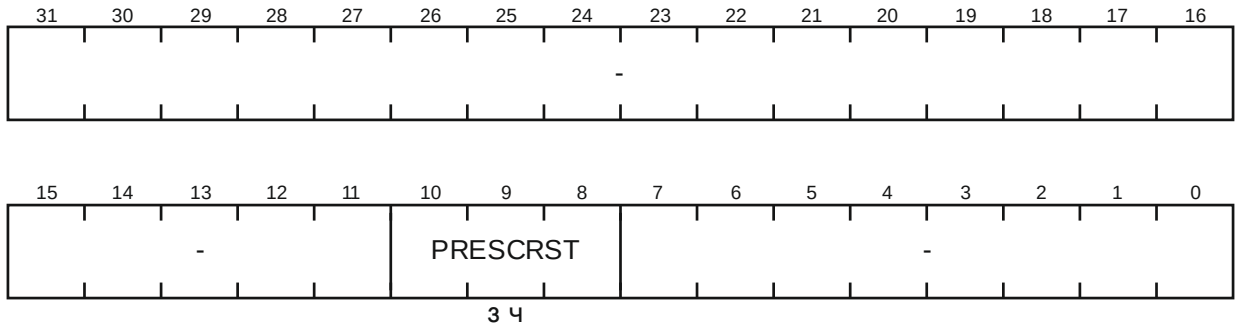
Поле	Биты	Описание	
BMDIS	0	Бит отключения старта из загрузчоной памяти после следующего программного сброса. Оказывает влияние, только если в CFGWORD активирован старт из загрузочной области	
		0	Старт из загрузочной области будет происходить каждый раз после любого из сбросов
		1	Старт из загрузочной области будет происходить только при внешнем или POR сбросе. Программный сброс будет приводить к загрузке из основной области
–	31-1	Зарезервировано	

A.5 Регистры блока управления системой

Базовый адрес: 4004_0000h

PWMSYNC – регистр настройки синхронизации PWM

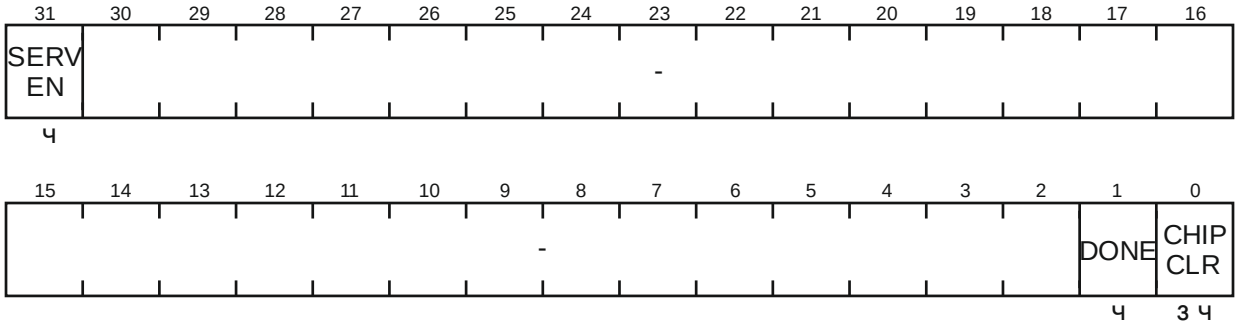
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
PRESCRST	10-8	Биты сброса счетчиков предварительных делителей блоков ШИМ. Запись нуля в младший бит поля сбрасывает счетчик блока ШИМ0, в первый бит – счетчик блока ШИМ1, в старший бит – счетчик блока ШИМ2. Запись единицы разрешает счет
–	31-11, 7-0	Зарезервировано

SERVCTL – регистр настройки сервисного режима

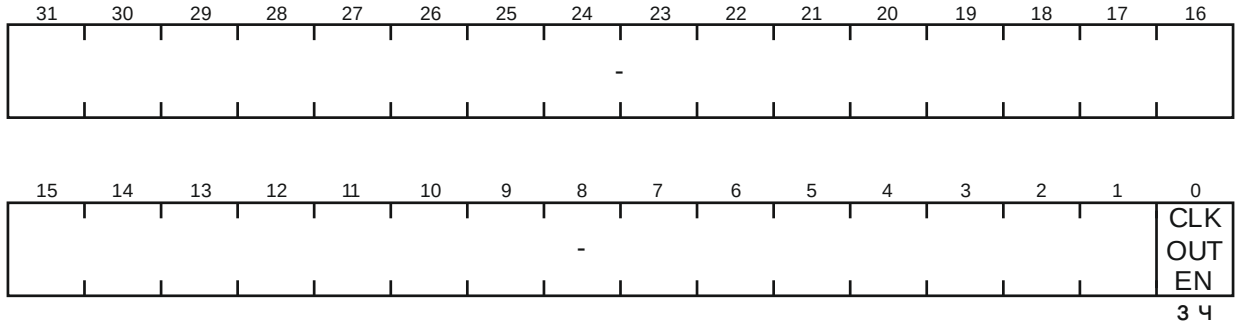
Смещение: + 14h
Сброс: 0h



Поле	Биты		Описание
SERVEN	31	0	Обычный режим работы
		1	Сервисный режим. Во время сброса на выводе SERVEN была единица
DONE	1	Флаг завершения команды сервисного стирания	
		0	Команда не завершена
		1	Команда завершена
CHIPCLR	0	Бит стирания. Запись единицы активирует полное стирание	
–	30-2	Зарезервировано	

CLKOUTCTL – регистр настройки выдачи тактового сигнала

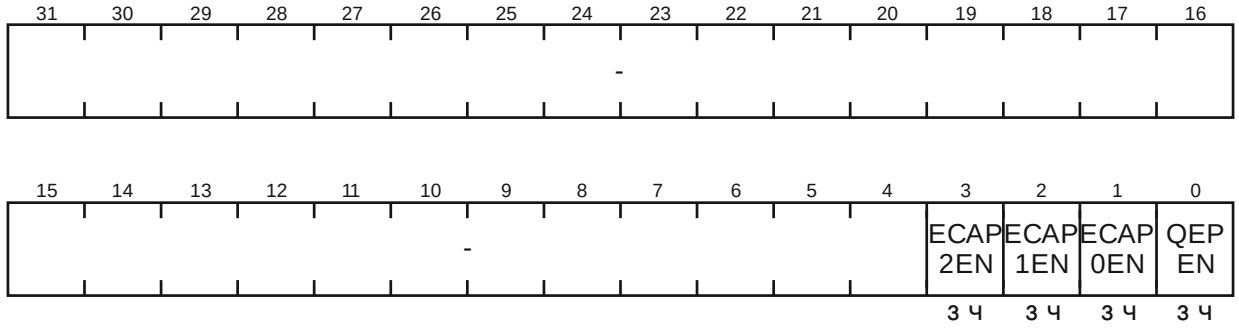
Смещение: + 18h
Сброс: 0h



Поле	Биты	Описание	
CLKOUTEN	0	Бит включения функции выдачи тактового сигнала на выводе SERVEN	
		0	Выключено
		1	Включено
–	31-1	Зарезервировано	

REMAPAF – регистр переключения альтернативных функций

Смещение: + 1Ch
Сброс: 0h

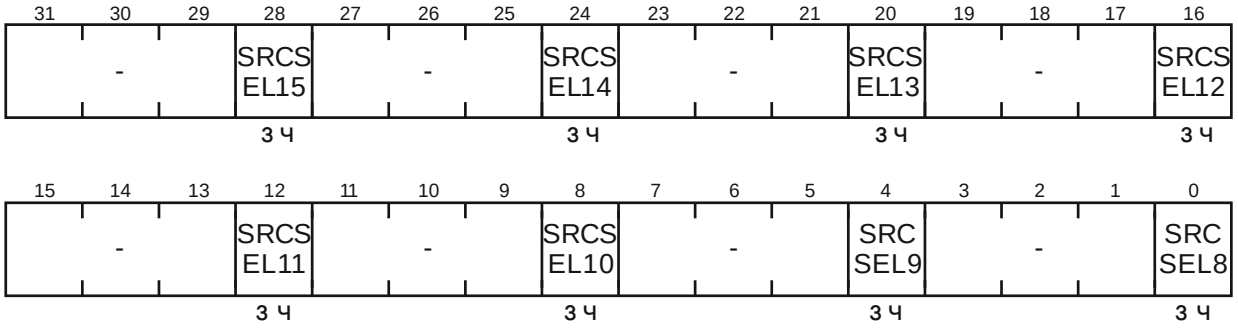


Поле	Биты	Описание
ECAP2EN	3	Включение функции блока ECAP2 на выводе A6
ECAP1EN	2	Включение функции блока ECAP1 на выводе A5
ECAP0EN	1	Включение функции блока ECAP0 на выводе A4
QEPEN	0	Включение функций блока QEP на выводах B10 – B13
–	31-4	Зарезервировано

Примечание – Для всех бит регистра:
0 – функция выключена;
1 – функция включена.

DMAMUX – регистр настройки конфигурируемых каналов DMA

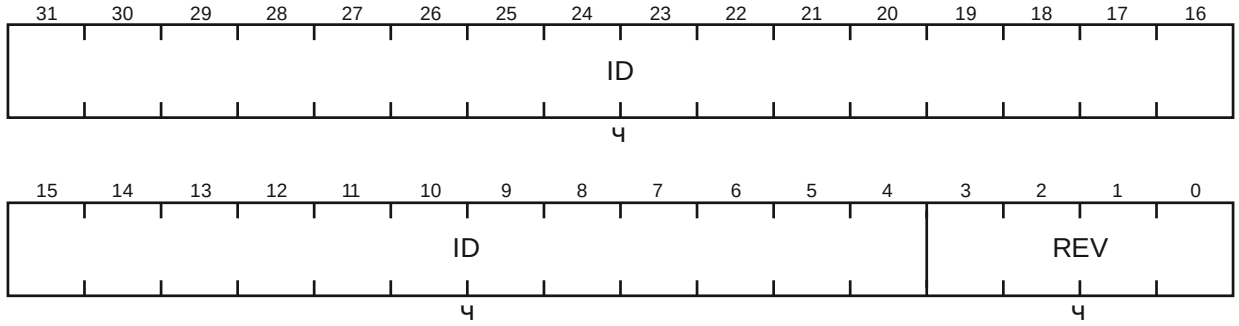
Смещение: + 20h
Сброс: 0h



Поле	Биты	Описание	
SRCSEL15	28	Выбор источника для канала 15 контроллера DMA	
		0	Запрос от блока PWM2 канал A
		1	Нет аппаратного источника
SRCSEL14	24	Выбор источника для канала 14 контроллера DMA	
		0	Запрос от блока PWM1 канал A
		1	Нет аппаратного источника
SRCSEL13	20	Выбор источника для канала 13 контроллера DMA	
		0	Запрос от блока PWM0 канал A
		1	Нет аппаратного источника
SRCSEL12	16	Выбор источника для канала 12 контроллера DMA	
		0	Запрос от блока TMR3
		1	Запрос от блока PWM2 канал B
SRCSEL11	12	Выбор источника для канала 11 контроллера DMA	
		0	Запрос от блока TMR2
		1	Запрос от блока PWM1 канал B
SRCSEL10	8	Выбор источника для канала 10 контроллера DMA	
		0	Запрос от блока TMR1
		1	Запрос от блока PWM0 канал B
SRCSEL9	4	Выбор источника для канала 9 контроллера DMA	
		0	Запрос от блока TMR0
		1	Запрос от порта B
SRCSEL8	0	Выбор источника для канала 8 контроллера DMA	
		0	Запрос от блока QEP
		1	Запрос от порта A
–	31-29, 27-25, 23-21, 19-17, 15-13, 11-9, 7-5, 3-1	Зарезервировано	

CHIPID – регистр идентификации системы

Смещение: + FFCh
Сброс: 5A29_8FE1h



Поле	Биты	Описание
ID	31-4	Номер модели
REV	3-0	Номер ревизии

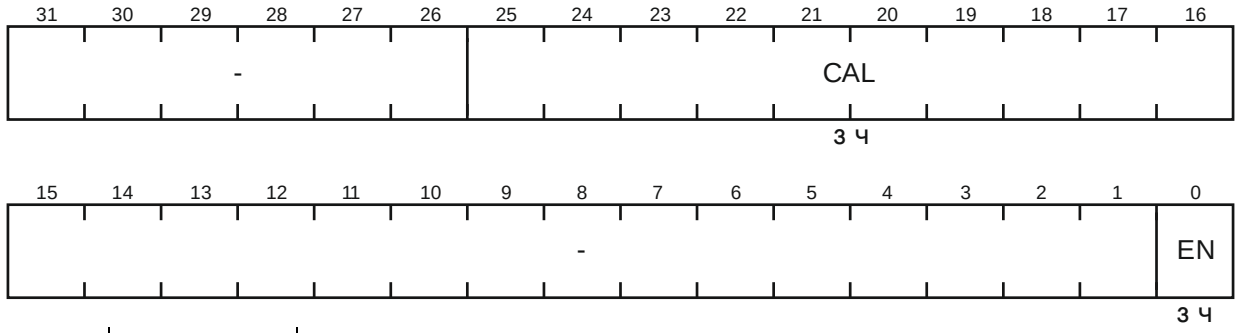
A.6 Регистры системы управления тактированием и сбросом

Базовый адрес: 4004_1000h

Смещение: + 60h (UARTCFG) Регистры настройки тактирования UART

OSICFG – регистр конфигурации внутреннего осциллятора

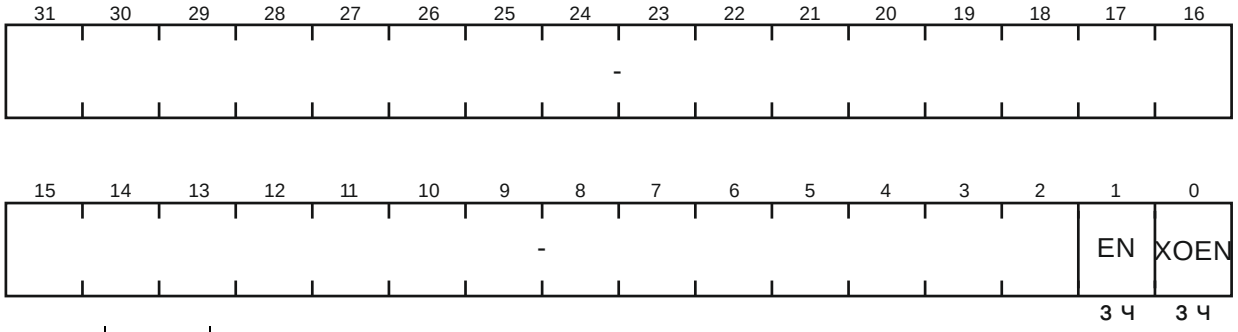
Смещение: + 00h
Сброс: 220_0001h



Поле	Биты	Описание	
CAL	25-16	Значение для подстройки частоты внутреннего осциллятора	
EN	0	Бит включения внутреннего осциллятора	
		0	Выключено
		1	Включено
–	31-26, 15-1	Зарезервировано	

OSECFG – регистр конфигурации внешнего осциллятора

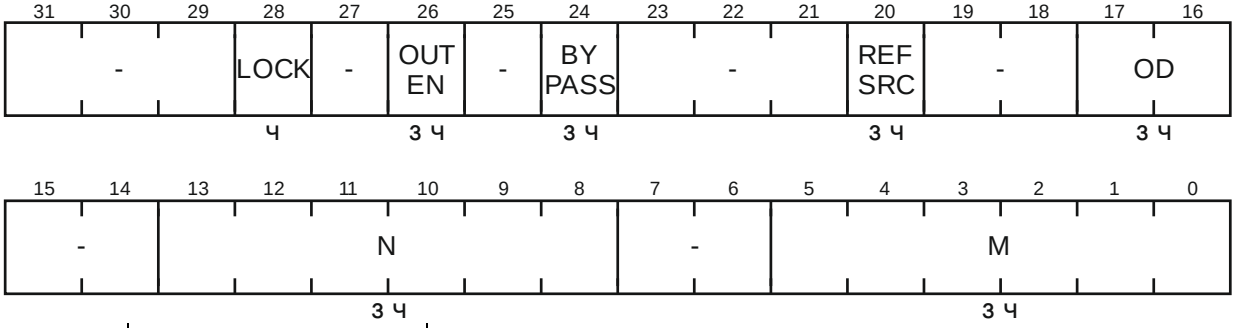
Смещение: + 04h
Сброс: 3h



Поле	Биты	Описание	
EN	1	Разрешение синхросигнала от внешнего осциллятора	
		0	Запрещено
		1	Разрешено
XOEN	0	Разрешение работы вывода XO_OSC	
		0	Запрещено
		1	Разрешено
–	31-2	Зарезервировано	

PLLCFG – регистр конфигурации PLL

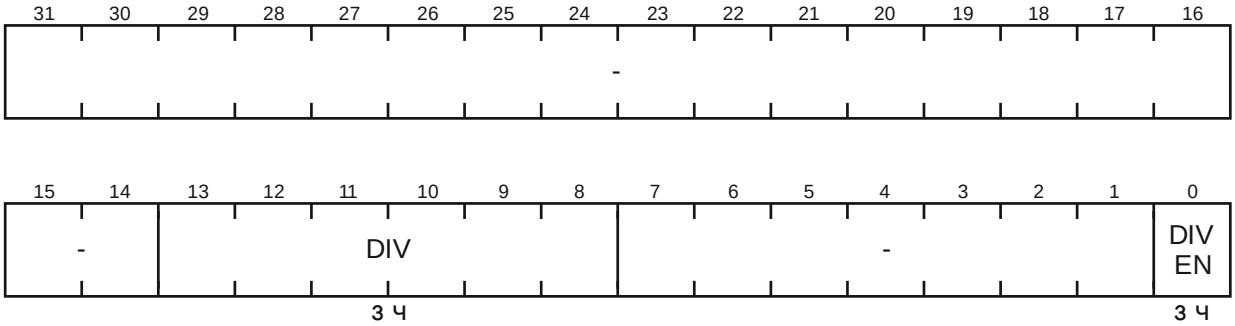
Смещение: + 08h
Сброс: 102h



Поле	Биты	Описание	
LOCK	28	Устанавливается, если блок PLL стабилен	
OUTEN	26	Разрешение вывода частоты с блока PLL	
BYPASS	24	Режим сквозного прохождения тактового сигнала	
REFSRC	20	Выбор источника входной частоты f_{IN}	
		0	Внешний тактовый сигнал OSECLK
		1	Внутренний тактовый сигнал OSICLK
OD	17-16	Коэффициент деления выходного сигнала PLL	
		00	1
		01	1/2
		10	1/4
		11	1/8
N	13-8	Коэффициент деления N_PLL. Значения от 1h до 3Fh	
M	5-0	Коэффициент деления M_PLL. Значения от 2h до 3Fh	
–	31-29, 27, 25, 23-21, 19-18, 15-14, 7- 6	Зарезервировано	

PLLDIV – регистр внешнего делителя PLL

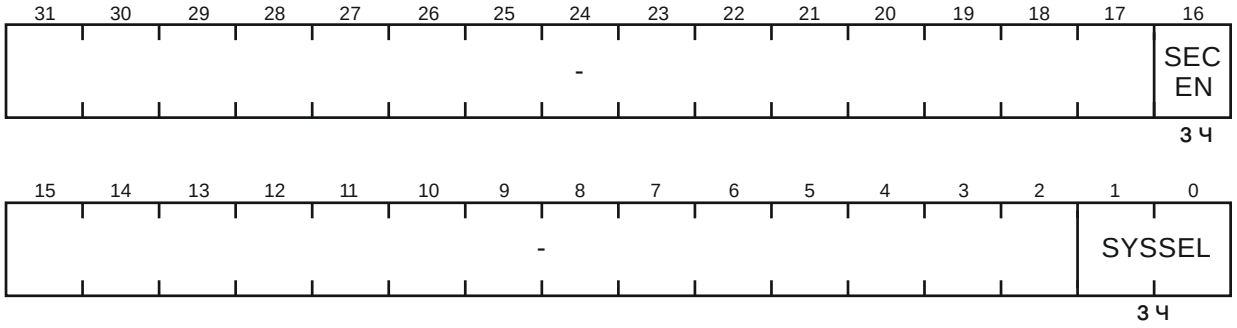
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание	
DIV	13-8	Коэффициент выходного делителя PLL, вычисляемый как DIV+1	
DIVEN	0	Бит разрешения выходного делителя частоты блока PLL	
		0	Запрещено
		1	Разрешено
–	31-14, 7-1	Зарезервировано	

SYSCLKCFG – регистр конфигурации системного тактового сигнала

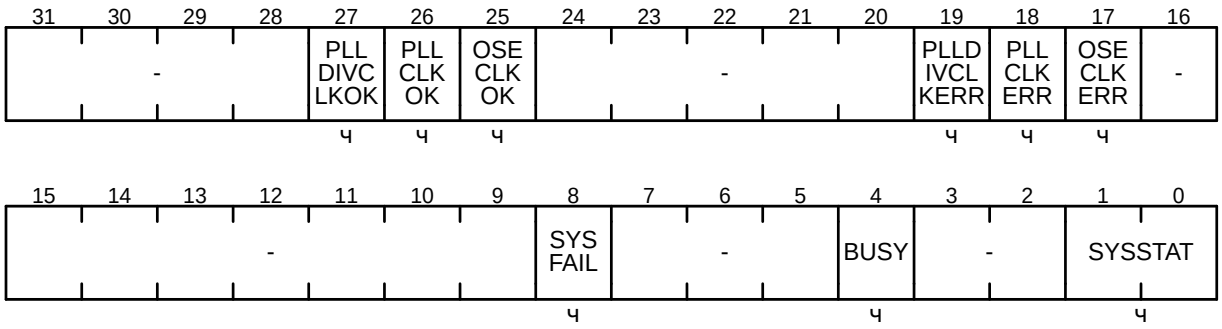
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание	
SECEN	16	Бит включения системы слежения за тактовым сигналом	
		0	Выключено
		1	Включено
SYSSEL	1-0	Выбор источника системного тактового сигнала	
		00	Сигнал OSICLK
		01	Сигнал OSECLK
		10	Сигнал с выхода блока PLL
		11	Сигнал после внешнего делителя PLL
–	31-17, 15-2	Зарезервировано	

SYSCLKSTAT – регистр статуса системного тактового сигнала

Смещение: + 14h
Сброс: C100_0000h

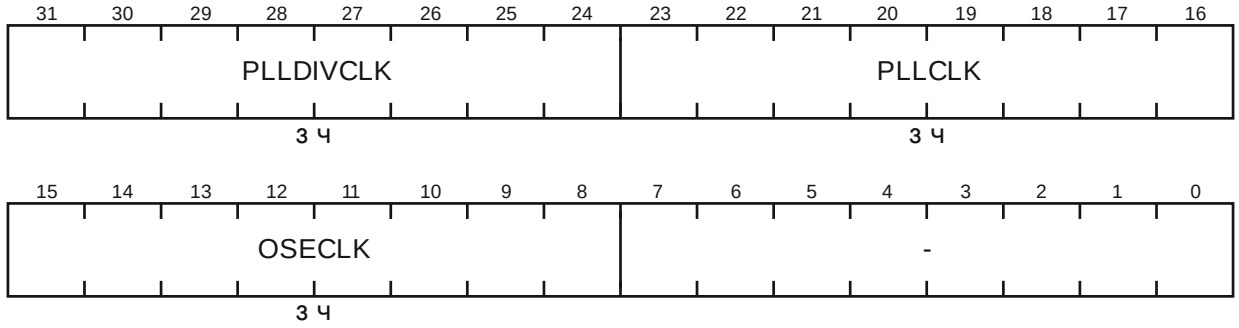


Поле	Биты	Описание
PLLDIV CLKOK	27	Флаг наличия тактового сигнала на выходе внешнего делителя PLL
PLLCLKOK	26	Флаг наличия тактового сигнала на выходе PLL
OSECLKOK	25	Флаг наличия выбранного тактового сигнала OSECLK
PLLDIV CLKERR	19	Флаг отсутствия тактового сигнала на выходе внешнего делителя PLL
PLLCLKERR	18	Флаг отсутствия тактового сигнала на выходе PLL

Поле	Биты	Описание	
OSECLKERR	17	Флаг отсутствия выбранного тактового сигнала OSECLK	
SYSFAIL	8	Флаг ошибки при пропадании тактового сигнала, который был выбран в качестве системного	
BUSY	4	Флаг занятости блока RCU (например, во время смены тактового сигнала)	
SYSSTAT	1-0	Текущий источник системного тактового сигнала	
		00	Сигнал OSICLK
		01	Сигнал OSECLK
		10	Сигнал с выхода PLL
		11	Сигнал после внешнего делителя PLL
–	31-28, 24-20, 16-9, 7-5, 3-2	Зарезервировано	

SECPRD – регистр слежения за состоянием системного тактового сигнала

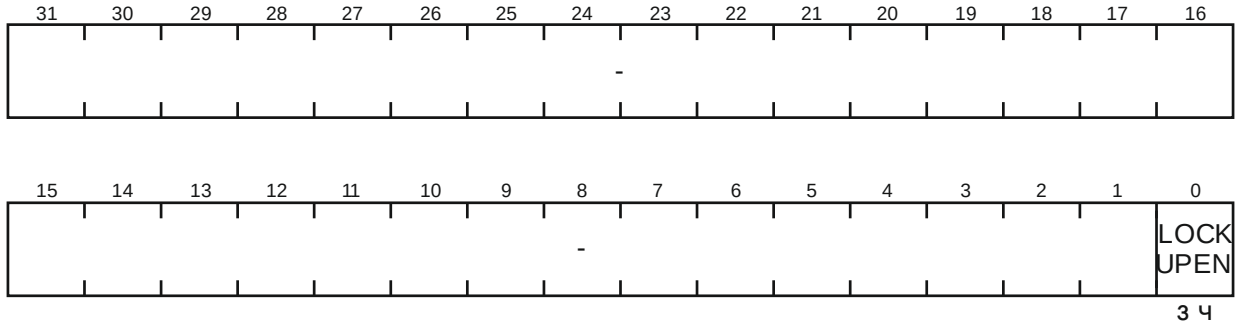
Смещение: + 18h
Сброс: FFFF_FF00h



Поле	Биты	Описание
PLLDIVCLK	31-24	Максимальное значение счетчика периодов для детектирования пропадания сигнала PLLDIVCLK
PLLCLK	23-16	Максимальное значение счетчика периодов для детектирования пропадания сигнала PLLCLK
OSECLK	15-8	Максимальное значение счетчика периодов для детектирования пропадания сигнала OSECLK
–	7-0	Зарезервировано

SYSRSTCFG – регистр настройки системного сброса

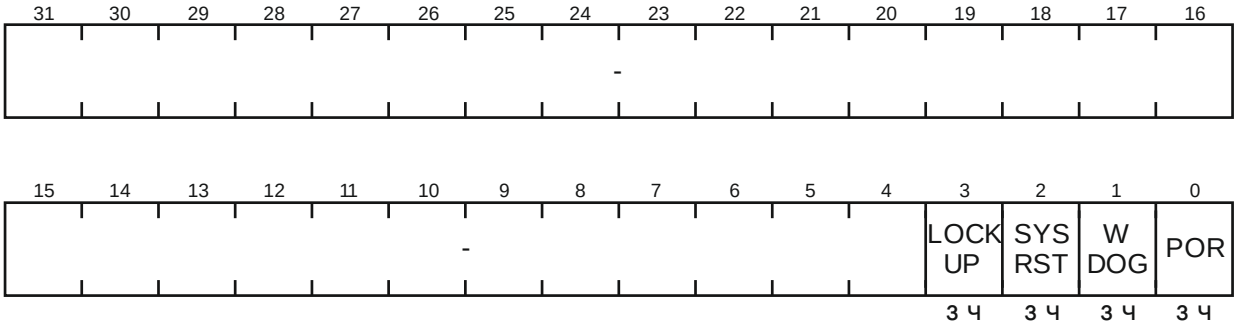
Смещение: + 1Ch
Сброс: 0h



Поле	Биты	Описание	
LOCKUPEN	0	Бит разрешения сброса, когда процессор в состоянии LOCKUP	
		0	Запрещено
		1	Разрешено
–	31-1	Зарезервировано	

SYSRSTSTAT – регистр статуса системного сброса

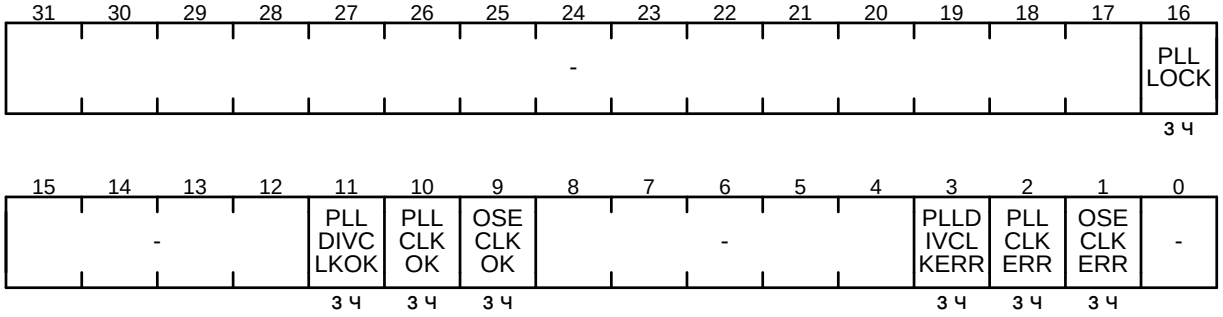
Смещение: + 20h
Сброс: 1h



Поле	Биты	Описание
LOCKUP	3	Флаг устанавливается, если последний сброс произошел из-за входа процессора в состояние LOCKUP
SYSRST	2	Флаг устанавливается, если последний сброс произошел в результате подачи сигнала системного сброса
WDOG	1	Флаг устанавливается, если последний сброс был активирован сторожевым таймером
POR	0	Флаг устанавливается, если последний сброс произошел из-за срабатывания схемы POR
–	31-4	Зарезервировано

INTEN – регистр разрешения прерываний

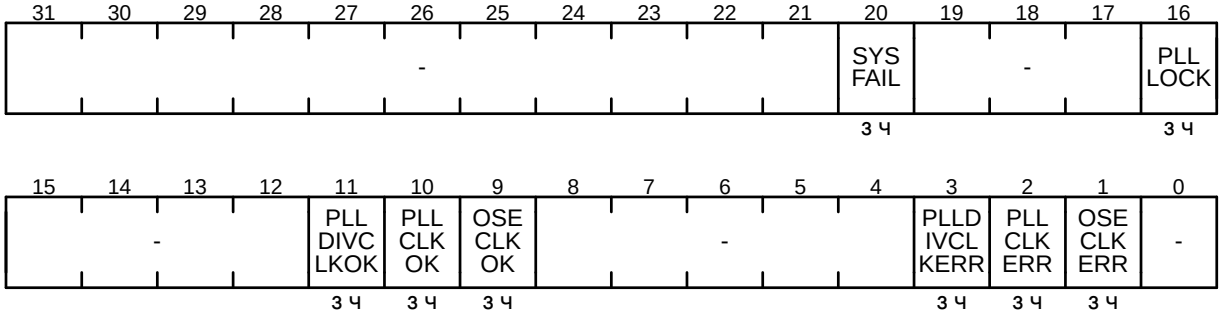
Смещение: + 24h
Сброс: 0h



Поле	Биты	Описание
PLLLOCK	16	Разрешение прерывания по сигналу LOCK
PLLDIVCLKOK	11	Разрешение прерывания при появлении стабильного тактового сигнала на выходе внешнего делителя PLL
PLLCLKOK	10	Разрешение прерывания при появлении стабильного тактового сигнала на выходе PLL
OSECLKOK	9	Разрешение прерывания при появлении стабильного выбранного тактового сигнала OSECLK
PLLDIVCLKERR	3	Разрешение прерывания при пропадании стабильного тактового сигнала на выходе внешнего делителя PLL
PLLCLKERR	2	Разрешение прерывания при пропадании стабильного тактового сигнала на выходе PLL
OSECLKERR	1	Разрешение прерывания при пропадании стабильного выбранного тактового сигнала OSECLK
–	31-17, 15-12, 8-4, 0	Зарезервировано

INTSTAT – регистр статуса прерываний

Смещение: + 28h
Сброс: 0h

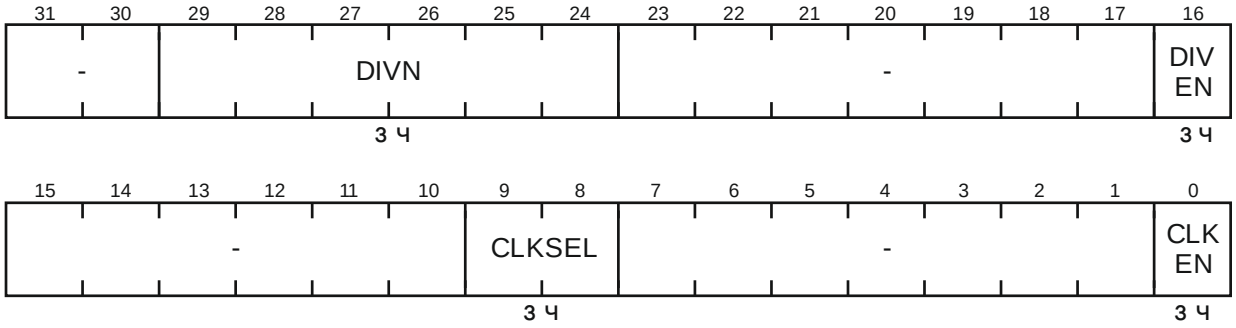


Поле	Биты	Описание
SYSFAIL	20	Флаг устанавливается при попытке перейти на отсутствующий источник тактирования (прерывание, выставляющее данный флаг, соответствует вектору NMI)
PLLLOCK	16	Флаг прерывания по сигналу LOCK
PLLDIVCLKOK	11	Флаг прерывания при появлении стабильного тактового сигнала на выходе внешнего делителя PLL
PLLCLKOK	10	Флаг прерывания при появлении стабильного тактового сигнала на выходе PLL
OSECLKOK	9	Флаг прерывания при появлении стабильного выбранного тактового сигнала OSECLK
PLLDIVCLKERR	3	Флаг прерывания при пропадании стабильного тактового сигнала на выходе внешнего делителя PLL
PLLCLKERR	2	Флаг прерывания при пропадании стабильного тактового сигнала на выходе PLL
OSECLKERR	1	Флаг прерывания при пропадании стабильного выбранного тактового сигнала OSECLK
–	31-21, 19-17, 15-12, 8-4, 0	Зарезервировано

Примечание – Флаги сбрасываюся только программно. Для сброса флага следует записать единицу в соответствующий бит.

TRACECFG – регистр настройки интерфейса трассировки

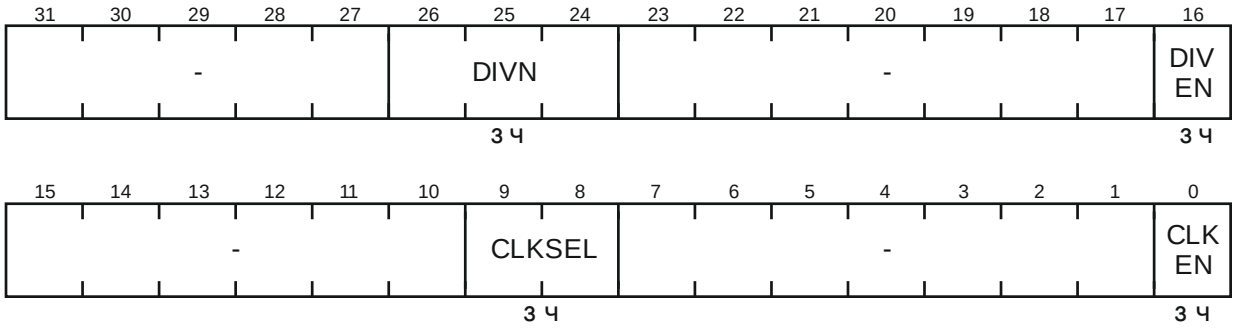
Смещение: + 2Ch
Сброс: 1h



Поле	Биты	Описание	
DIVN	29-24	Коэффициент деления входной частоты интерфейса трассировки. Результирующий коэффициент деления N = 2 × (DIVN + 1)	
DIVEN	16	Разрешение делителя входной частоты интерфейса трассировки	
CLKSEL	9-8	Выбор источника входной частоты интерфейса трассировки	
		00	OSICLK
		01	OSECLK
		10	PLLCLK
		11	PLLDIVCLK
CLKEN	0	Разрешение тактирования	
–	31-30, 23-17, 15-10, 7-1	Зарезервировано	

CLKOUTCFG – регистр настройки выходного тактового сигнала

Смещение: + 30h
Сброс: 0h

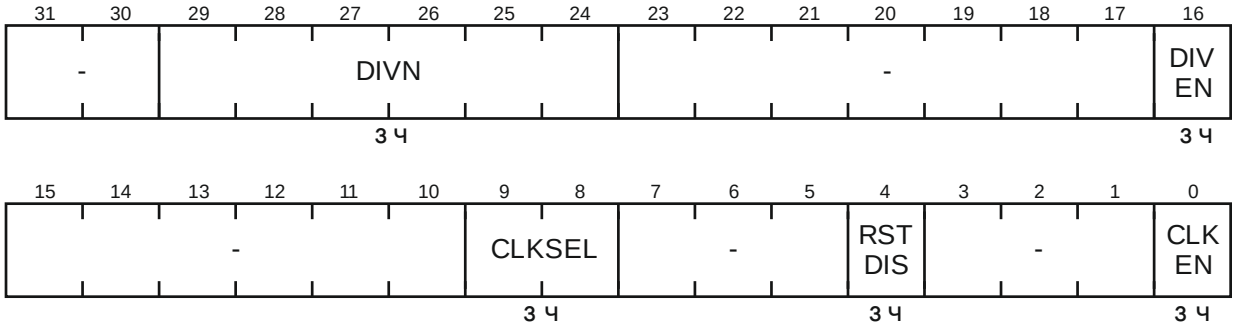


Поле	Биты	Описание
DIVN	26-24	Коэффициент деления выходного сигнала. Результирующий коэффициент деления N = 2×(DIVN+1)
DIVEN	16	Разрешение делителя выходного сигнала

Поле	Биты	Описание	
CLKSEL	9-8	Выбор источника выходного сигнала	
		00	OSICLK
		01	OSECLK
		10	PLLCLK
		11	PLLDIVCLK
CLKEN	0	Разрешение выходного сигнала	
–	31-27, 23-17, 15-10, 7-1	Зарезервировано	

WDTCFG – регистр настройки сторожевого таймера

Смещение: + 34h
Сброс: 301_0000h

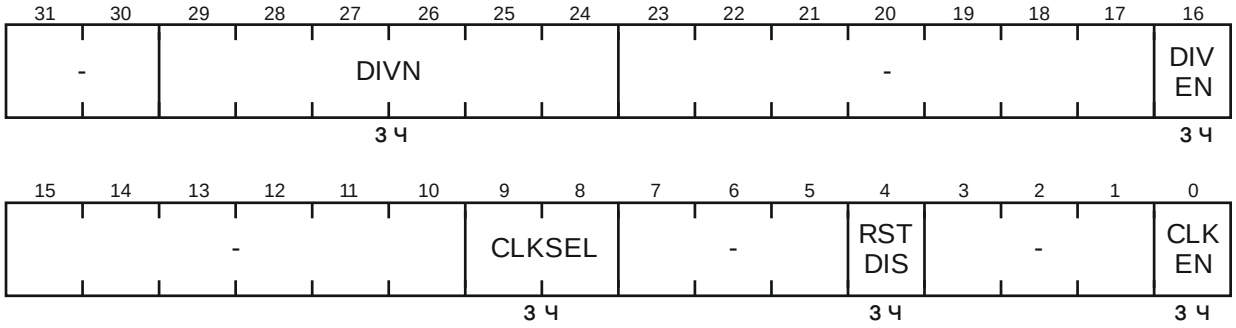


Поле	Биты	Описание	
DIVN	29-24	Коэффициент деления входного сигнала. Результирующий коэффициент деления N = 2×(DIVN+1).	
DIVEN	16	Разрешение делителя входного сигнала	
CLKSEL	9-8	Выбор источника тактового сигнала	
		00	OSICLK
		01	OSECLK
		10	PLLCLK
		11	PLLDIVCLK
RSTDIS	4	Снятие сигнала сброса. Когда сигнал в нуле, блок в состоянии сброса	
CLKEN	0	Разрешение тактирования	
–	31-30, 23-17, 15-10, 7-5, 3-1	Зарезервировано	

Примечание – Частота тактирования сторожевого таймера должна быть как минимум в 2 раза ниже системной.

UARTCFG – массив регистров настройки UART

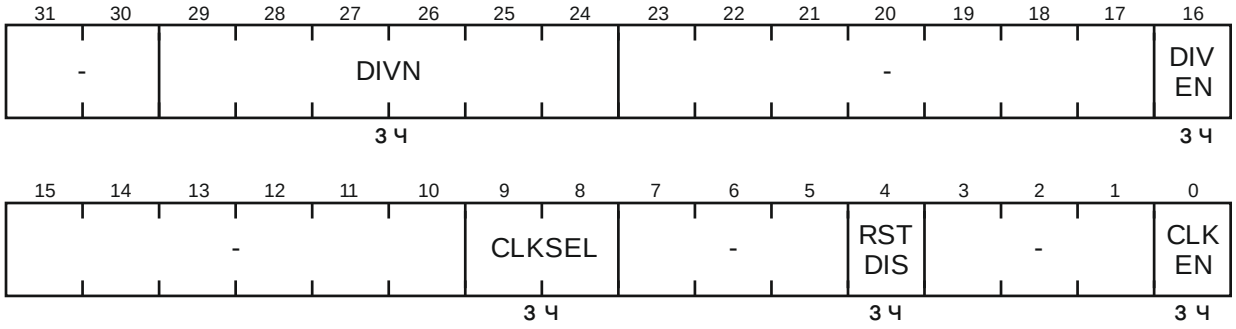
Смещение: UARTCFG + (4*n)h, где n – номер блока 0, 1
Сброс: 0h



Поле	Биты	Описание	
DIVN	29-24	Коэффициент деления входного сигнала. Результирующий коэффициент деления N = 2×(DIVN+1)	
DIVEN	16	Разрешение делителя входного сигнала	
CLKSEL	9-8	Выбор источника тактового сигнала	
		00	OSECLK
		01	PLLCLK
		10	PLLDIVCLK
		11	OSICLK
RSTDIS	4	Снятие сигнала сброса. Когда сигнал в нуле, блок в состоянии сброса	
CLKEN	0	Разрешение тактирования	
–	31-30, 23-17, 15-10, 7-5, 3-1	Зарезервировано	

SPICFG – регистр настройки SPI

Смещение: + 80h
Сброс: 0h

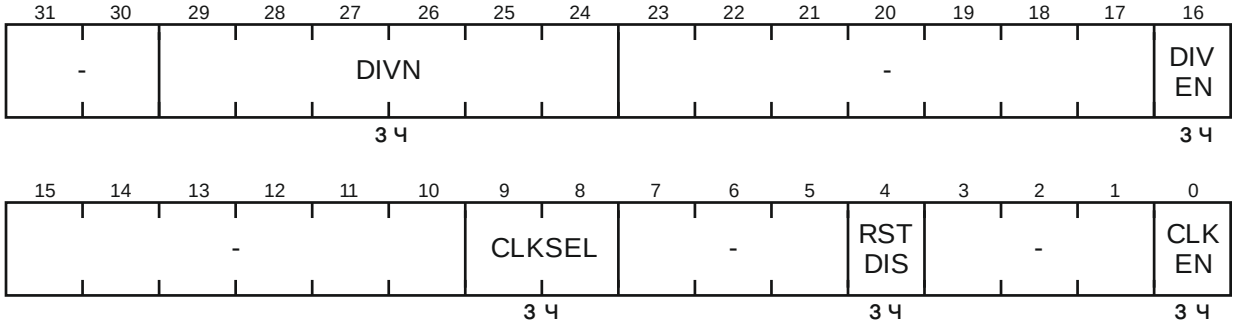


Поле	Биты	Описание
DIVN	29-24	Коэффициент деления входного сигнала. Результирующий коэффициент деления N = 2 × (DIVN + 1)
DIVEN	16	Разрешение делителя входного сигнала

Поле	Биты	Описание	
CLKSEL	9-8	Выбор источника тактового сигнала	
		00	OSECLK
		01	PLLCLK
		10	PLLDIVCLK
		11	OSICLK
RSTDIS	4	Снятие сигнала сброса. Когда сигнал в нуле, блок в состоянии сброса	
CLKEN	0	Разрешение тактирования	
–	31-30, 23-17, 15-10, 7-5, 3-1	Зарезервировано	

ADCCFG – регистр настройки блока АЦП

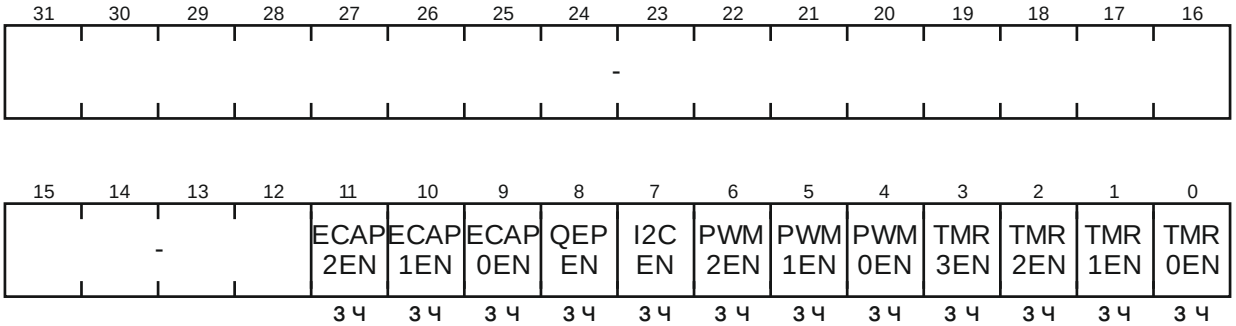
Смещение: + A0h
Сброс: 0h



Поле	Биты	Описание	
DIVN	29-24	Коэффициент деления входного сигнала. Результирующий коэффициент деления N = 2 × (DIVN + 1)	
DIVEN	16	Разрешение делителя входного сигнала	
CLKSEL	9-8	Выбор источника тактового сигнала	
		00	OSECLK
		01	PLLCLK
		10	PLLDIVCLK
		11	OSICLK
RSTDIS	4	Снятие сигнала сброса. Когда сигнал в нуле, блок в состоянии сброса	
CLKEN	0	Разрешение тактирования	
–	31-30, 23-17, 15-10, 7-5, 3-1	Зарезервировано	

PCLKCFG – регистр разрешения тактовых сигналов периферийных блоков шины APB

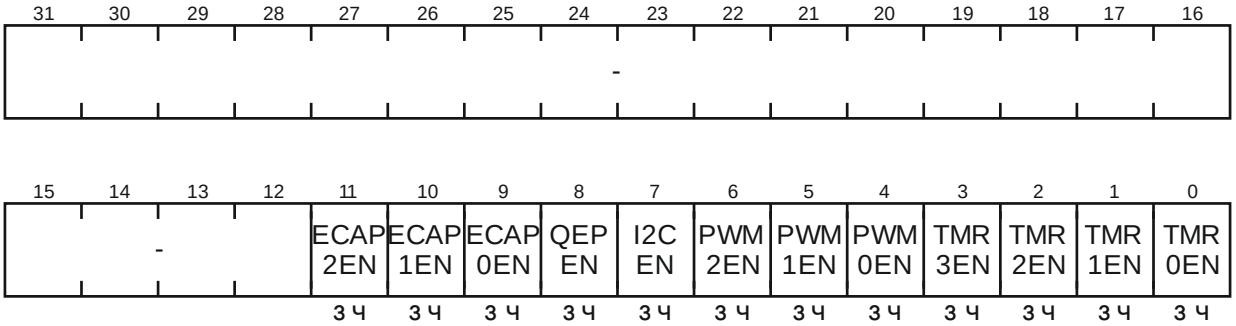
Смещение: + E0h
Сброс: 0h



Поле	Биты	Описание
ECAP2EN	11	Бит разрешения тактирования блока ECAP2
ECAP1EN	10	Бит разрешения тактирования блока ECAP1
ECAP0EN	9	Бит разрешения тактирования блока ECAP0
QEPEN	8	Бит разрешения тактирования блока QEP
I2CEN	7	Бит разрешения тактирования контроллера I2C
PWM2EN	6	Бит разрешения тактирования блока PWM2
PWM1EN	5	Бит разрешения тактирования блока PWM1
PWM0EN	4	Бит разрешения тактирования блока PWM0
TMR3EN	3	Бит разрешения тактирования блока TMR3
TMR2EN	2	Бит разрешения тактирования блока TMR2
TMR1EN	1	Бит разрешения тактирования блока TMR1
TMR0EN	0	Бит разрешения тактирования блока TMR0
–	31-12	Зарезервировано

PRSTCFG – регистр включения периферийных блоков шины APB

Смещение: + F0h
Сброс: 0h

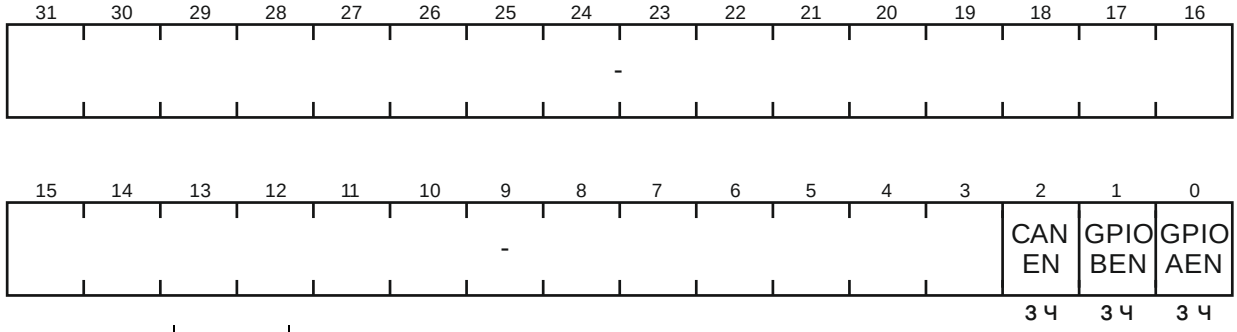


Поле	Биты	Описание
ECAP2EN	11	Бит включения блока ECAP2
ECAP1EN	10	Бит включения блока ECAP1
ECAP0EN	9	Бит включения блока ECAP0
QEPEN	8	Бит включения блока QEP

Поле	Биты	Описание
I2CEN	7	Бит включения контроллера I2C
PWM2EN	6	Бит включения блока PWM2
PWM1EN	5	Бит включения блока PWM1
PWM0EN	4	Бит включения блока PWM0
TMR3EN	3	Бит включения блока TMR3
TMR2EN	2	Бит включения блока TMR2
TMR1EN	1	Бит включения блока TMR1
TMR0EN	0	Бит включения блока TMR0
–	31-12	Зарезервировано

HCLKCFG – регистр разрешения тактирования периферийных блоков шины AHB

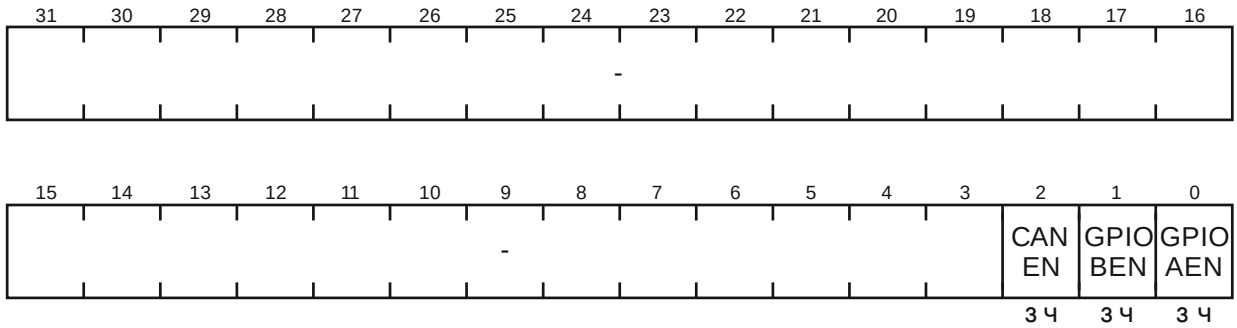
Смещение: + 100h
Сброс: 0h



Поле	Биты	Описание
CANEN	2	Бит разрешения тактирования CAN
GPIOBEN	1	Бит разрешения тактирования порта B
GPIOAEN	0	Бит разрешения тактирования порта A
–	31-3	Зарезервировано

HRSTCFG – регистр включения периферийных блоков шины AHB

Смещение: + 104h
Сброс: 0h



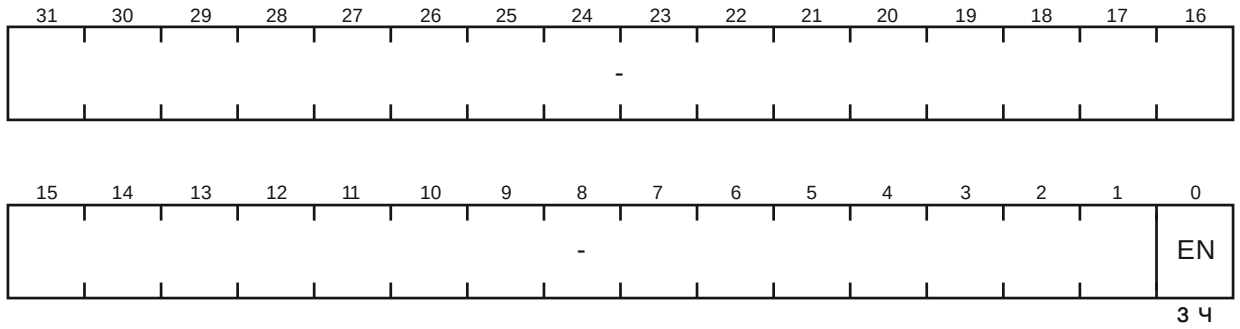
Поле	Биты	Описание
CANEN	2	Бит включения CAN
GPIOBEN	1	Бит включения порта B
GPIOAEN	0	Бит включения порта A
–	31-3	Зарезервировано

A.7 Регистры системы управления питанием

Базовый адрес: 4004_2000h

CFG – регистр настройки PMU

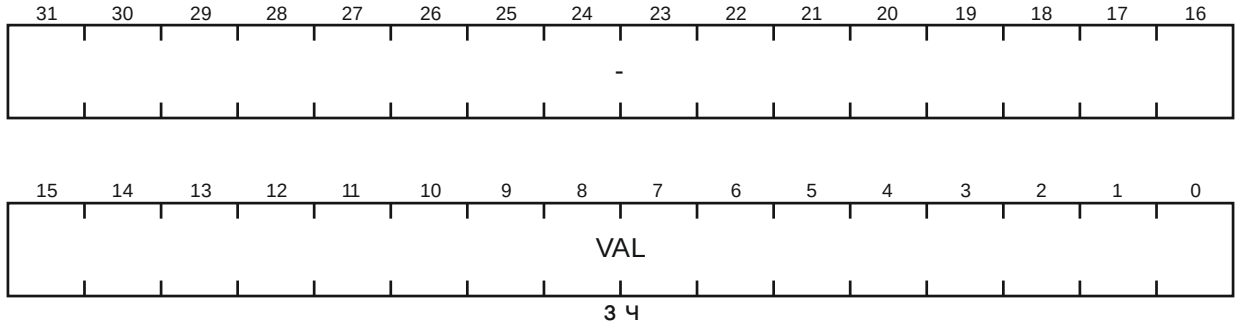
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание	
EN	0	Бит включения PMU	
		0	Выключено
		1	Включено
–	31-1	Зарезервировано	

PUDEL – регистр задания задержки пробуждения блоков

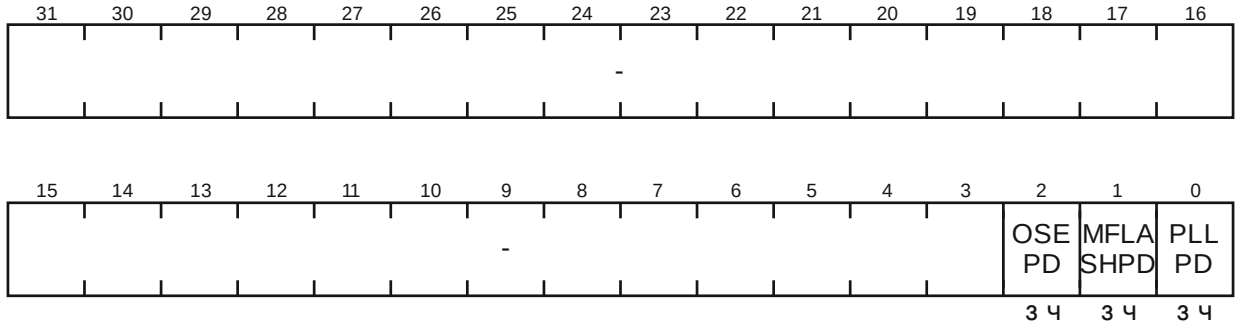
Смещение: + 04h
Сброс: xxxx_0690h



Поле	Биты	Описание
VAL	15-0	Значение задержки пробуждения для периферийных блоков в тактах OSICLK
–	31-16	Зарезервировано

PDEN – регистр управления режимом Powerdown

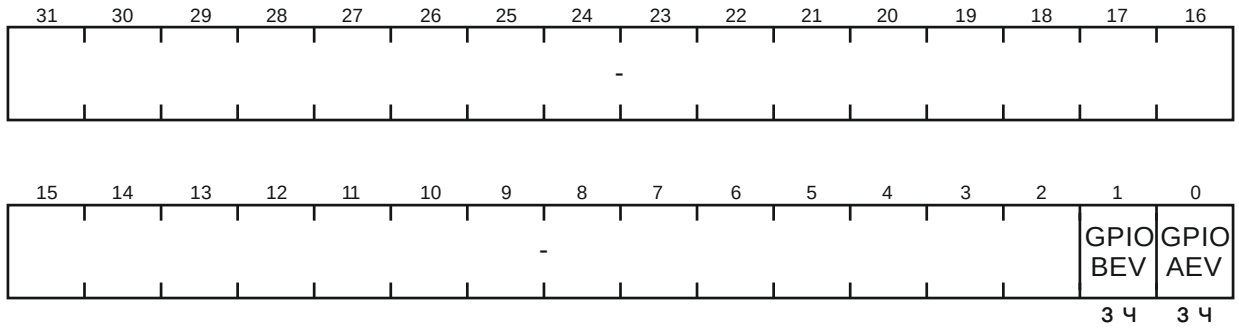
Смещение: + 08h
Сброс: 7h



Поле	Биты	Описание	
OSEPD	2	Бит включения режима Powerdown для внешнего осциллятора	
		0	Выключено
		1	Включено
MFLASHPD	1	Бит включения режима Powerdown для MFLASH	
		0	Выключено
		1	Включено
PLLPD	0	Бит включения режима Powerdown для PLL	
		0	Выключено
		1	Включено
–	31-3	Зарезервировано	

RXEVEN – регистр разрешения событий RXEV

Смещение: + 0Ch
Сброс: 0h



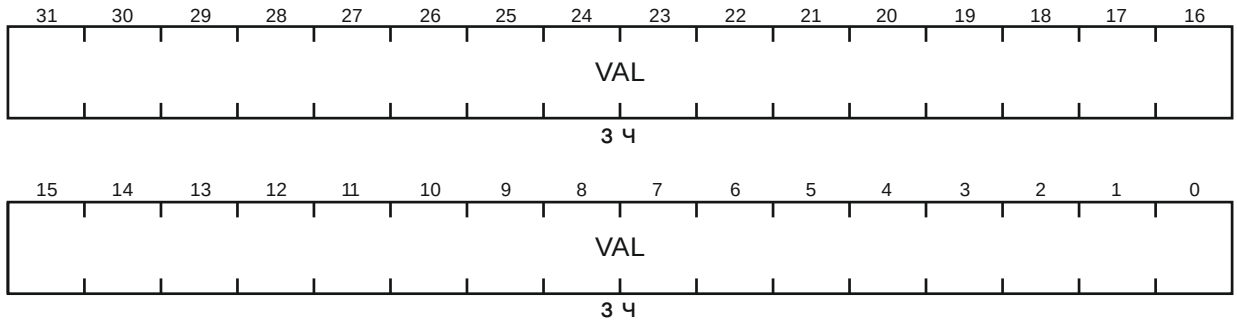
Поле	Биты	Описание	
GPIOBEV	1	Бит разрешения выхода из режима сна по RXEV от GPIOB	
		0	Выключено
		1	Включено
GPIOAEV	0	Бит разрешения выхода из режима сна по RXEV от GPIOA	
		0	Выключено
		1	Включено
–	31-2	Зарезервировано	

A.8 Регистры сторожевого таймера

Базовый адрес: 4004_3000h

LOAD – регистр загрузки сторожевого таймера

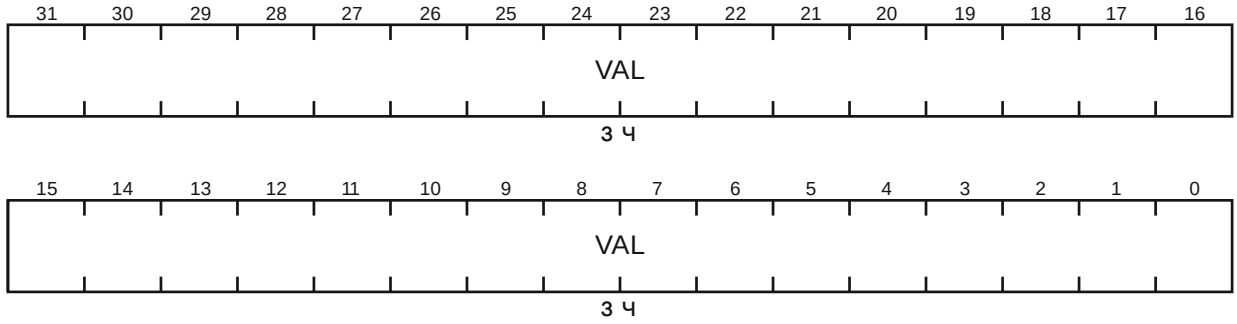
Смещение: + 00h
Сброс: FFFF_FFFFh



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

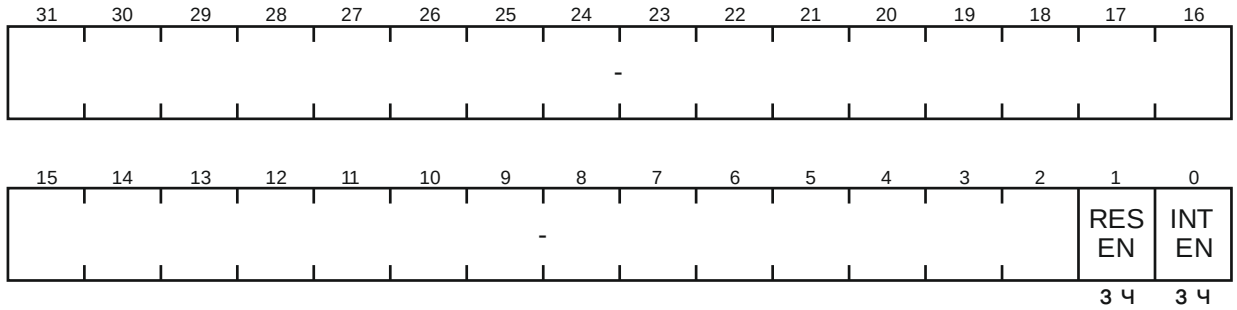
Смещение: + 04h
Сброс: FFFF_FFFFh



Поле	Биты	Описание
VAL	31-0	32-разрядный регистр текущего значения счетчика

CTRL – регистр управления сторожевого таймера

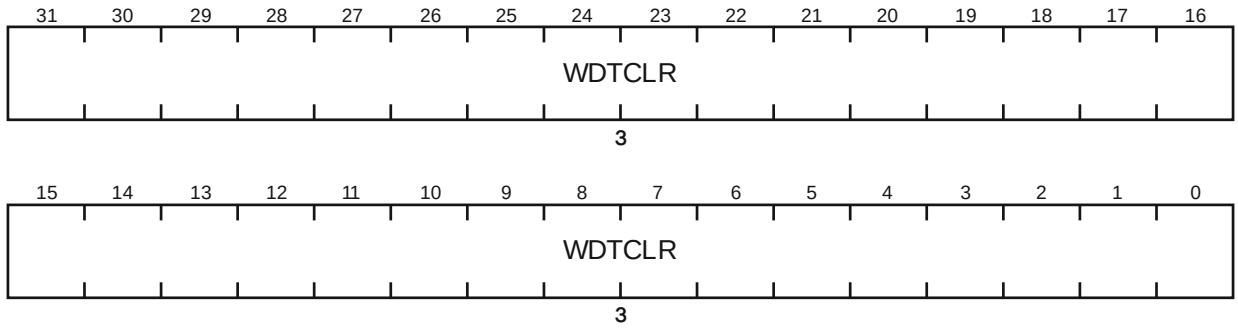
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание	
RESEN	1	Бит разрешения сброса микроконтроллера по сторожевому таймеру. Работает по функции «Логическое И» с битом INTEN регистра WDTCTRL	
		0	Сброс бита выключает сброс
		1	Установка бита 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 генерирует прерывание. Если счетчик был включен на момент установки бита, то он инициируется значением из регистра LOAD
–	31-2	Зарезервировано	

INTCLR – регистр сброса сторожевого таймера

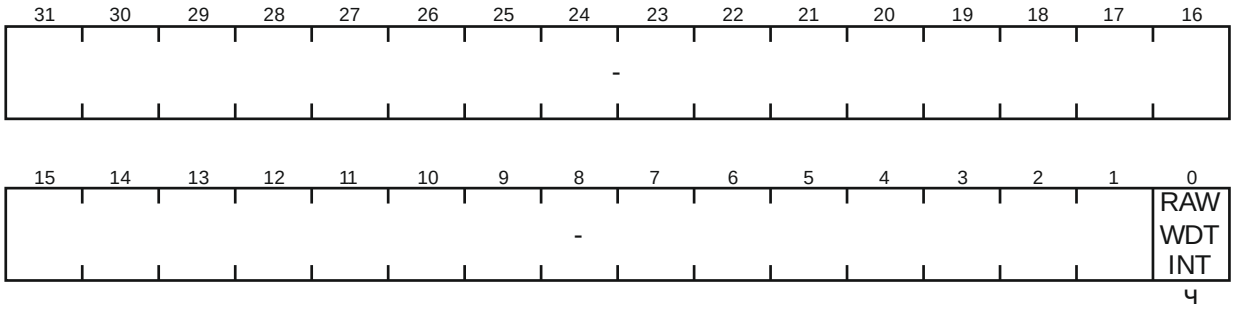
Смещение: + 0Ch
Сброс: 0h



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

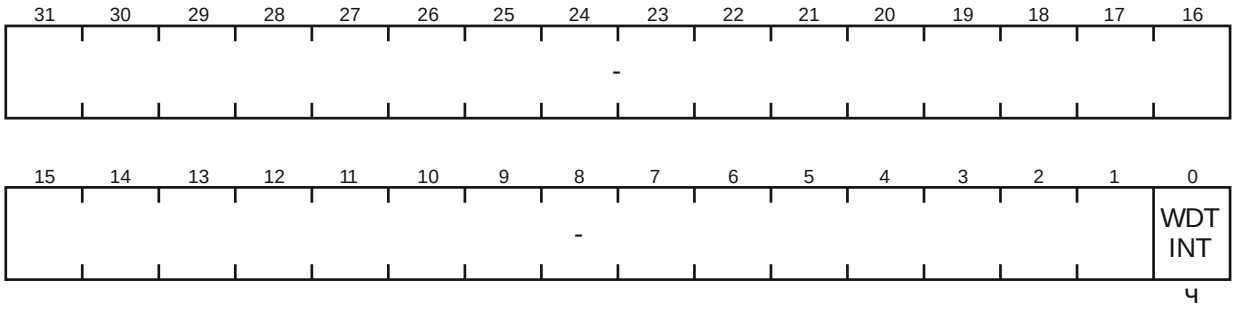
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание	
RAWWDTINT	0	Индикатор состояния немаскированного бита прерывания	
		0	Сброшено
		1	Установлено
–	31-1	Зарезервировано	

MIS – регистр маскированного прерывания сторожевого таймера

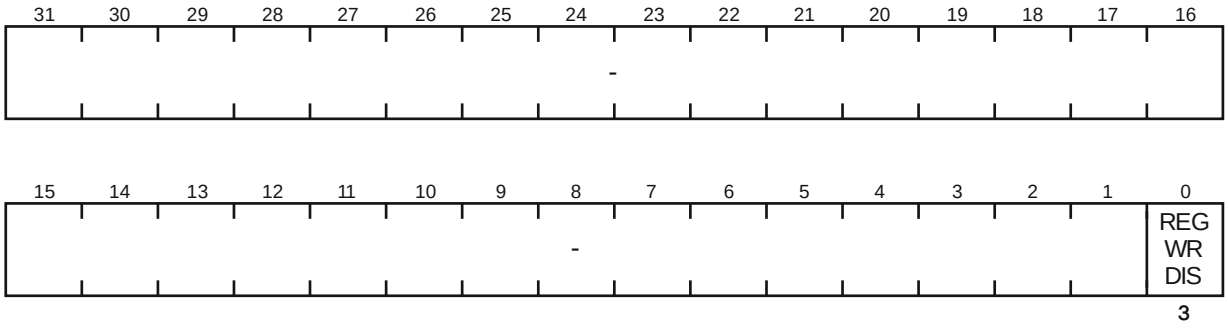
Смещение: + 14h
Сброс: 0h



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, 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 1ACC_E551h
		1	Запрещено. Для установки бита следует записать в регистр LOCK любое значение, кроме 1ACC_E551h
–	31-1	Зарезервировано	

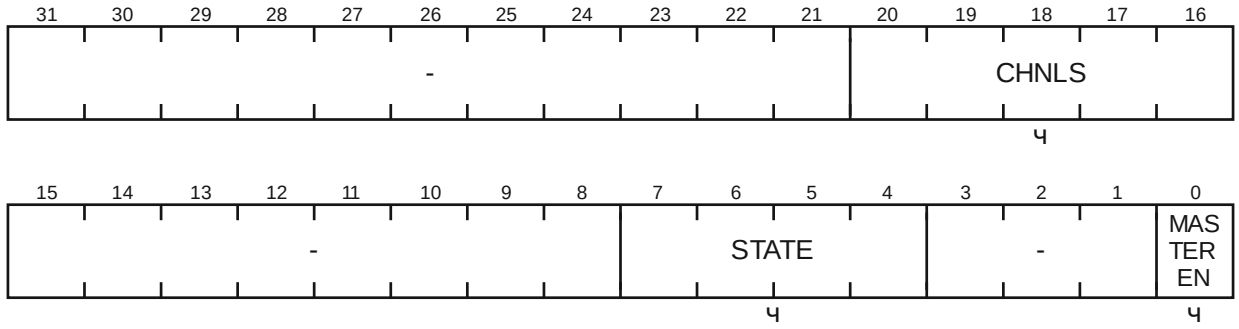
A.9 Регистры контроллера DMA

Базовый адрес: 4004_4000h

Примечание – i – номер по порядку канала от 0 до 15.

STATUS – регистр статуса DMA

Смещение: + 00h
Сброс: 0xxx_0000h

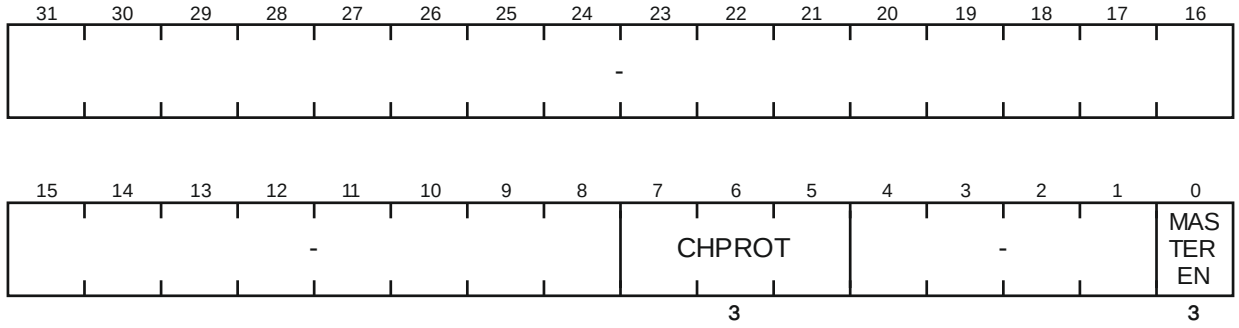


Поле	Биты	Описание	
CHNLS	20-16	Количество доступных каналов DMA	
		00h	1 канал
		01h	2 канала
		02h	3 канала
		…	…
		0Fh	16 каналов
		Значения 10h-1Fh зарезервированы	
STATE	7-4	Текущее состояние конечного автомата управления контроллера	
		0h	В покое
		1h	Чтение управляющих данных канала
		2h	Чтение 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	Приостановлено
		9h	Выполнено
		Ah	Режим работы с периферией «разборка-сборка»
		Bh-Fh	Зарезервировано
MASTEREN	0	Состояние контроллера DMA	
		0	Работа контроллера запрещена
		1	Работа контроллера разрешена
–	31-21, 15-8, 3-1	Зарезервировано	

Примечание – Регистр доступен только для чтения. Возвращает состояние контроллера DMA. Во время сброса чтение регистра запрещено.

CFG – регистр конфигурации контроллера DMA

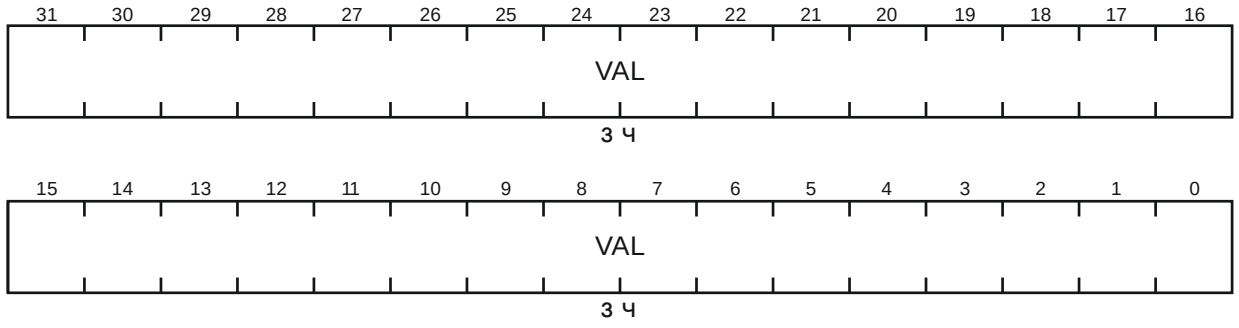
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание			
CHPROT	7-5	Задает параметры защиты шины AHB при обращении контроллера DMA к структурам управляющих данных каналов			
			Биты поля CHPROT		
			7	6	5
		0	Доступ не кэшируется	Доступ не буферизуется	Доступ непривилегированный
		1	Доступ кэшируется	Доступ буферизуется	Доступ привилегированный
MASTEREN	0	Бит разрешения работы контролера DMA			
		0	Запрещено		
		1	Разрешено		
–	31-8, 4-1	Зарезервировано			

BASEPTR – регистр базового адреса управляющих данных каналов

Смещение: + 08h
Сброс: 0h

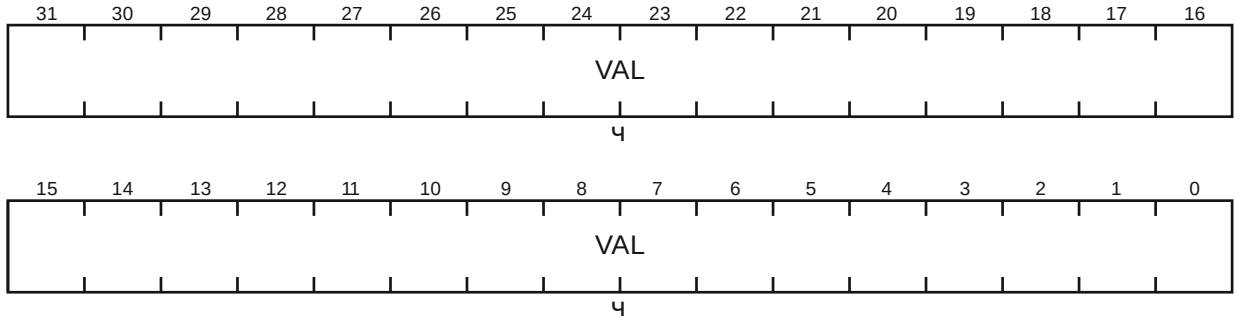


Поле	Биты	Описание
VAL	31-0	Указатель на базовый адрес первичной структуры управляющих данных

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

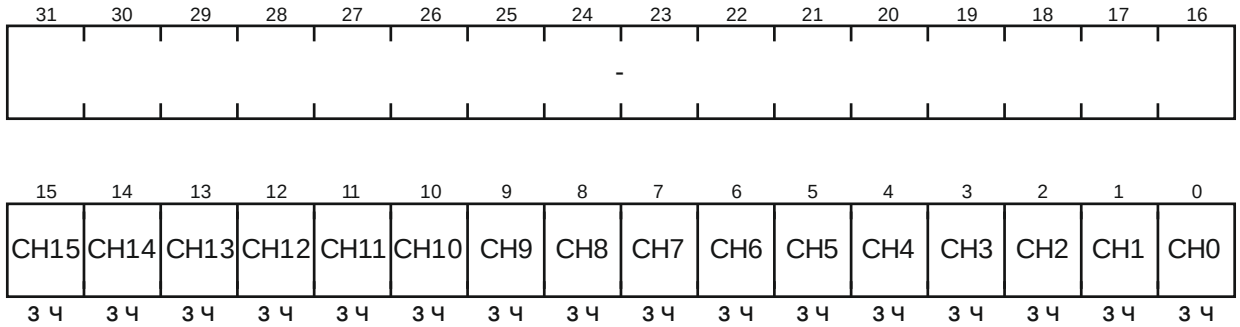


Поле	Биты	Описание
VAL	31-0	Указатель базового адреса альтернативной структуры управляющих данных каналов.

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

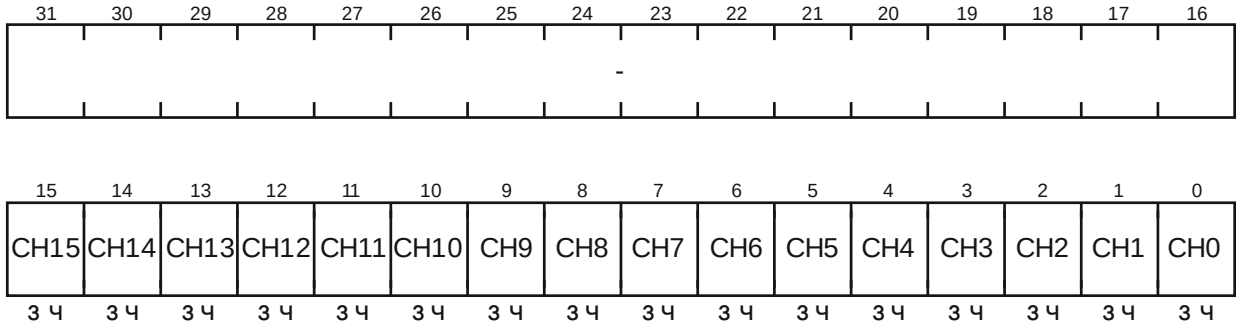
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание		
CHi	15-0	Чтение	0	Доступны только BREQ запросы от периферии
			1	Доступны BREQ и SREQ запросы от периферии
		Запись	Не выполняется	
–	31-16	Зарезервировано		

SWREQ – регистр программного запроса на обработку каналов DMA

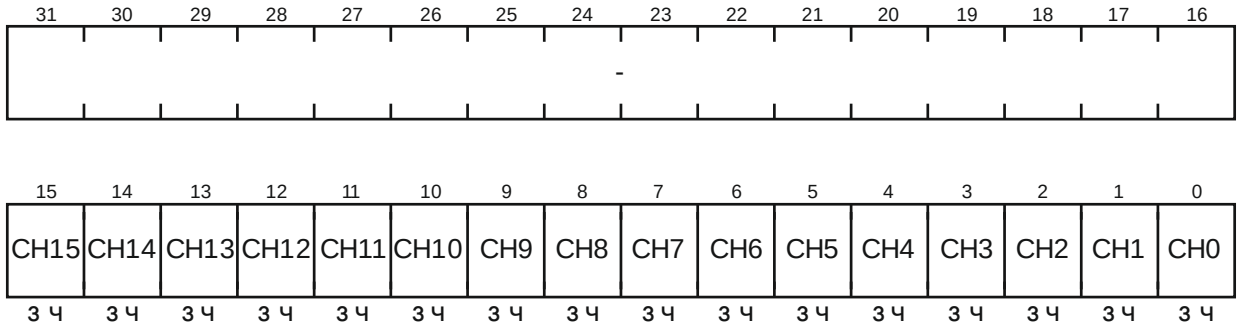
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание	
CHi	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Устанавливает запрос на выполнение цикла DMA по каналу i
–	31-16	Зарезервировано	

USEBURSTSET – регистр установки пакетного обмена каналов DMA

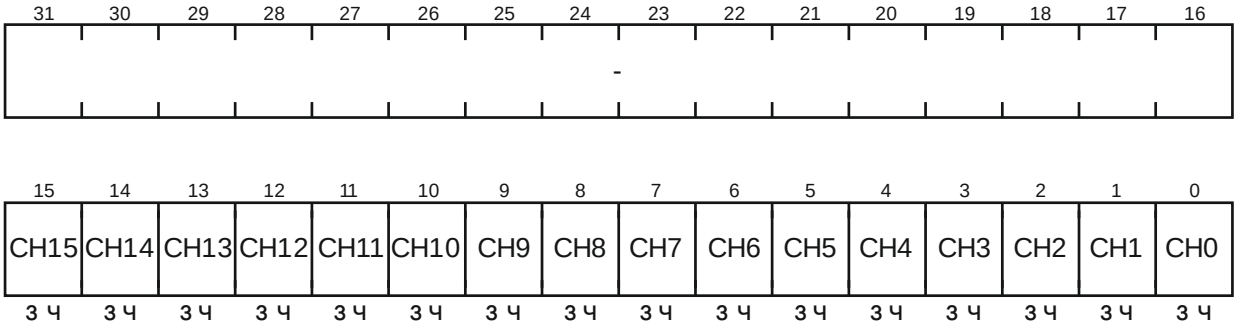
Смещение: + 18h
Сброс: 0h



Поле	Биты	Описание		
CHi	15-0	Бит отключения выполнения одиночных запросов SREQ. При активации режима будут обрабатываться и исполняться только пракетные 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	
–	31-16	Зарезервировано		

USEBURSTCLR – регистр сброса пакетного обмена каналов DMA

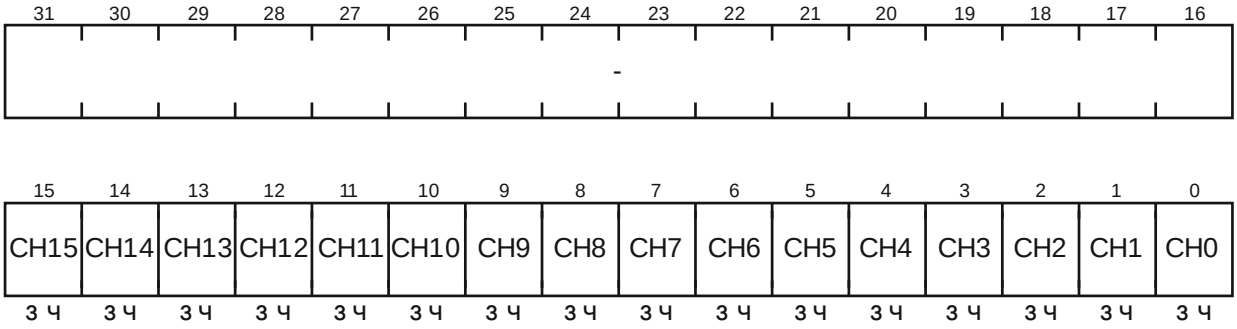
Смещение: + 1Ch
Сброс: 0h



Поле	Биты	Описание	
CHi	15-0	Чтение	Читаются нули
		Запись нуля	Не выполняется
		Запись единицы	Разрешает обрабатывать одиночные запросы SREQ на выполнение циклов DMA
–	31-16	Зарезервировано	

REQMASKSET – регистр маскирования запросов от периферии на обслуживание каналов DMA

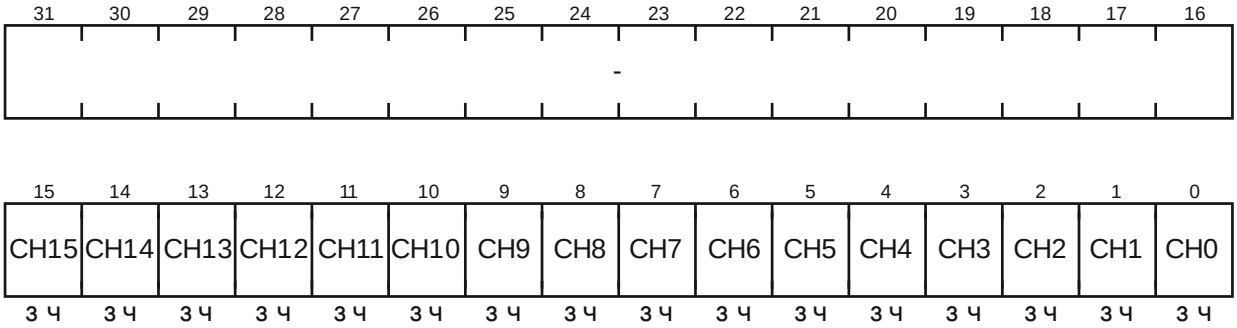
Смещение: + 20h
Сброс: 0h



Поле	Биты	Описание		
CHi	15-0	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	
–	31-16	Зарезервировано		

REQMASKCLR – регистр сброса маскирования запросов на обслуживание каналов DMA

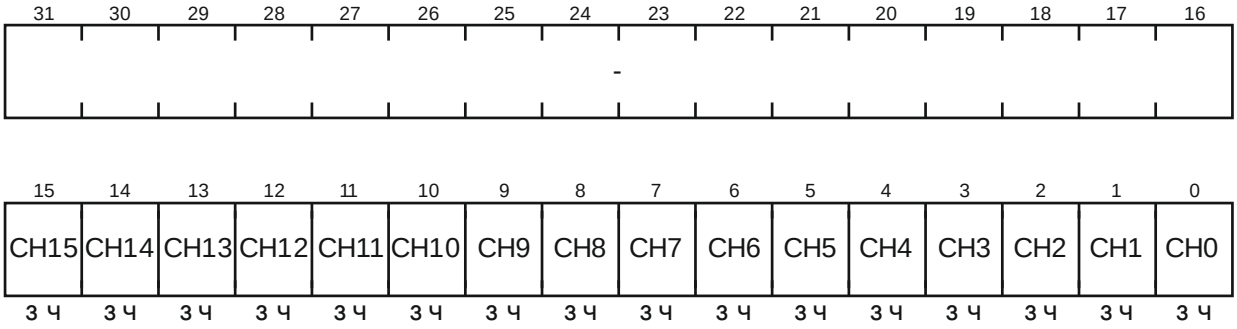
Смещение: + 24h
Сброс: 0h



Поле	Биты	Описание	
CHi	15-0	Чтение	Читаются нули
		Запись нуля	Не выполняется
		Запись единицы	Разрешает выполнение циклов DMA по запросам SREQ и BREQ периферии
–	31-16	Зарезервировано	

ENSET – регистр установки разрешения работы каналов DMA

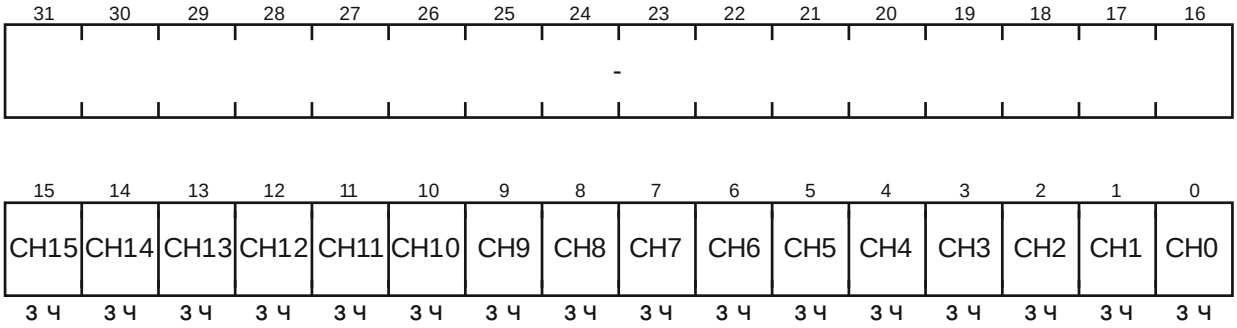
Смещение: + 28h
Сброс: 0h



Поле	Биты	Описание		
CHi	15-0	Чтение	0	Канал i отключен
			1	Работа канала i разрешена
		Запись нуля	Не выполняется	
		Запись единицы	Разрешает работу канала i	
–	31-16	Зарезервировано		

ENCLR – регистр сброса разрешения работы каналов DMA

Смещение: + 2Ch
Сброс: 0h

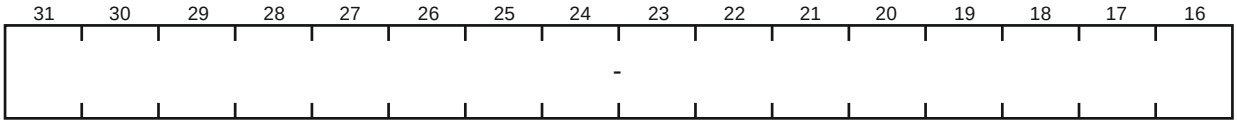


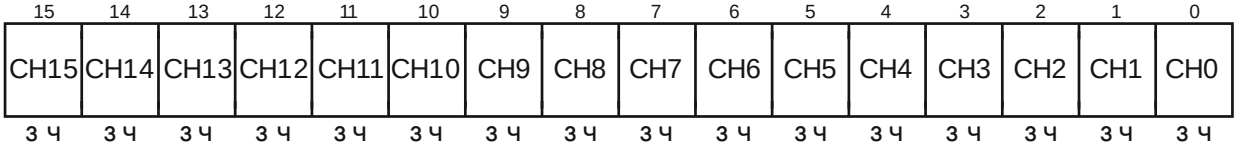
Поле	Биты	Описание	
CHi	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Запрещает работу канала i
–	31-16	Зарезервировано	

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





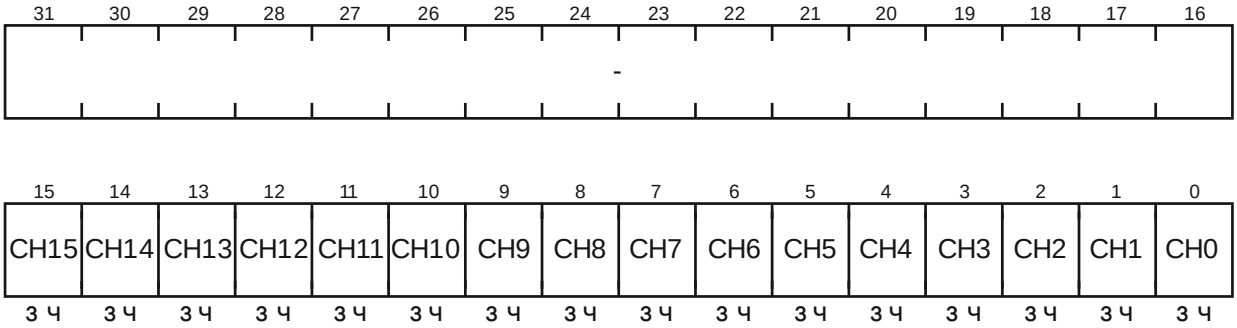
Поле	Биты	Описание		
CHi	15-0	Бит включения использования каналом i альтернативной структуры управляющих данных		
		Чтение	0	Используется первичная структура
			1	Используется альтернативная структура
		Запись нуля	Не выполняется	
		Запись единицы	Включает использование альтернативной структуры управляющих данных	
–	31-16	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

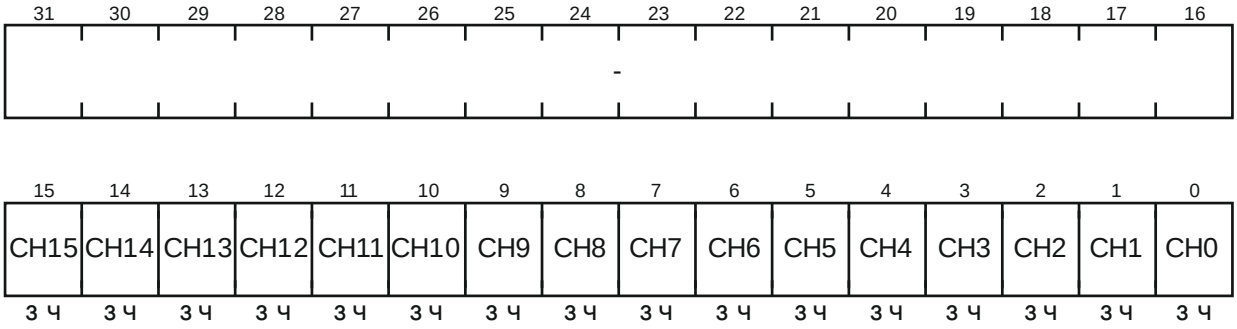
Смещение: + 34h
Сброс: 0h



Поле	Биты	Описание	
CHi	15-0	Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Включает использование первичной структуры управляющих данных канала i
–	31-16	Зарезервировано	

PRIORITYSET – регистр установки приоритета каналов DMA

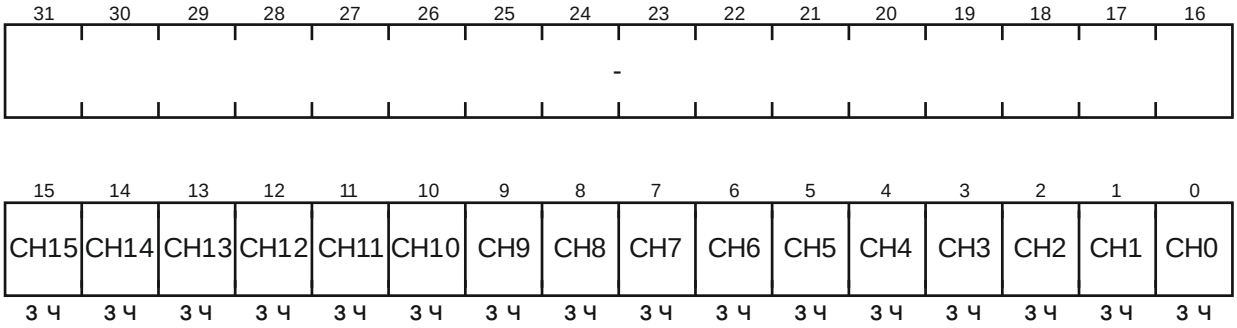
Смещение: + 38h
Сброс: 0h



Поле	Биты	Описание		
CHi	15-0	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	
–	31-16	Зарезервировано		

PRIORITYCLR – регистр сброса установок приоритета каналов DMA

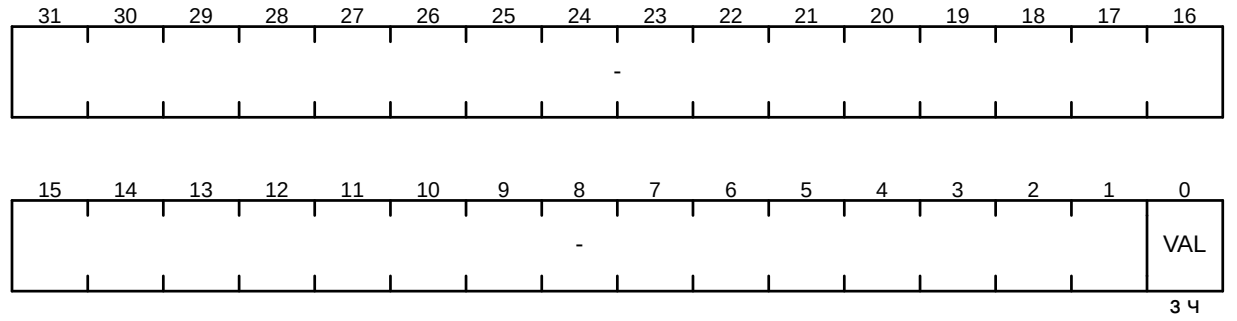
Смещение: + 3Ch
Сброс: 0h



Поле	Биты	Описание	
CHi	15-0	Чтение	Читаются нули
		Запись нуля	Не выполняется
		Запись единицы	Присваивает каналу i уровень приоритета по умолчанию
–	31-16	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	Сбросывает флаг ошибки	
–	31-1	Зарезервировано		

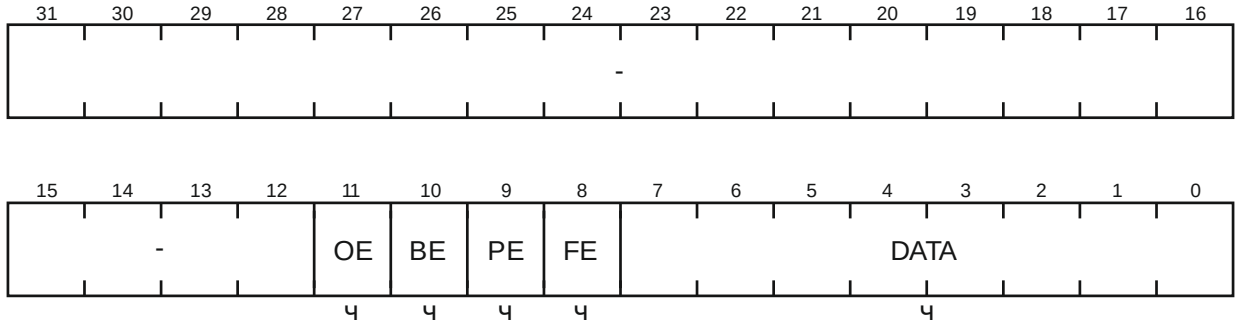
Примечание – При одновременном сбросе флага ERRCLR и появлении ошибки на шине AHB приоритет отдается ошибке, и бит ERRCLR остается установленным.

A.10 Регистры блока UART

Базовый адрес: 4004_5000h (UART0) Регистры приемопередатчика UART0
4004_6000h (UART1) Регистры приемопередатчика UART1

DR – регистр данных

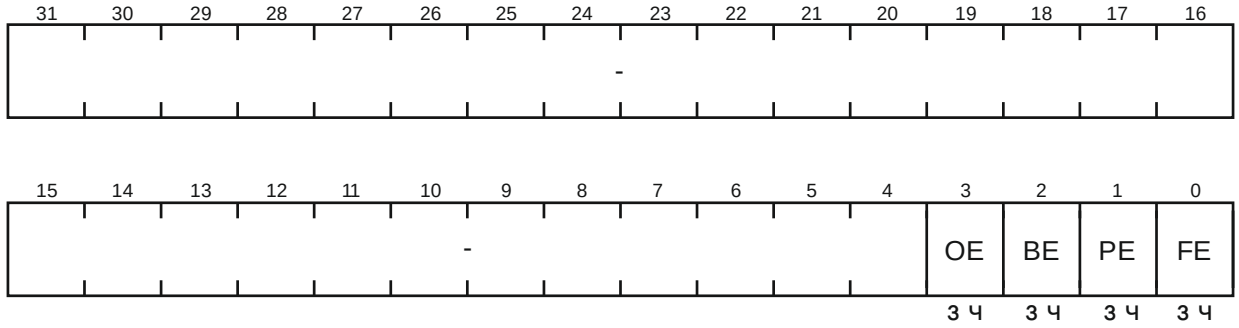
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание
OE, BE, PE, FE	11, 10, 9, 8	См. 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

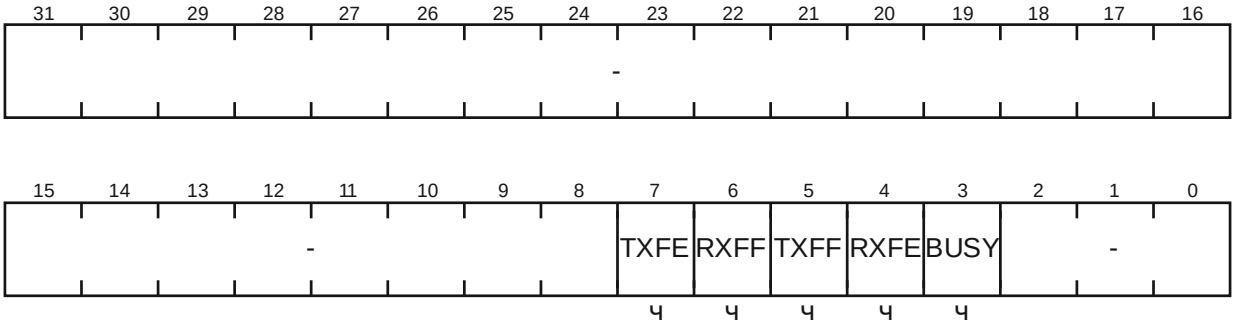


Поле	Биты	Описание	
OE	3	Флаг переполнения буфера приемника	
		0	В буфере есть свободное место или бит был сброшен после записи в регистр RSR. Содержимое буфера остается верным, так как был перезаписан только сдвиговый регистр. Центральный процессор должен считать данные для того, чтобы освободить буфер
		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

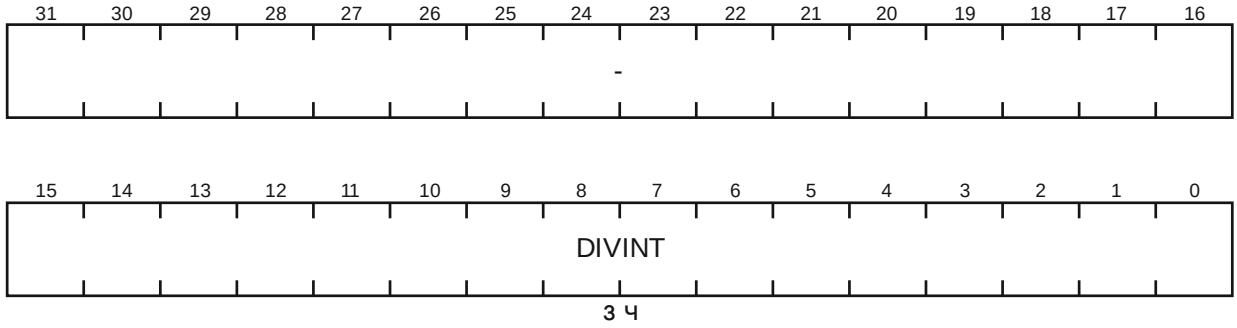
Смещение: + 18h
Сброс: 0h



Поле	Биты	Описание	
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 LCR_H (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)
–	31-8, 2-0	Зарезервировано	

IBRD – регистр целой части делителя скорости обмена данными

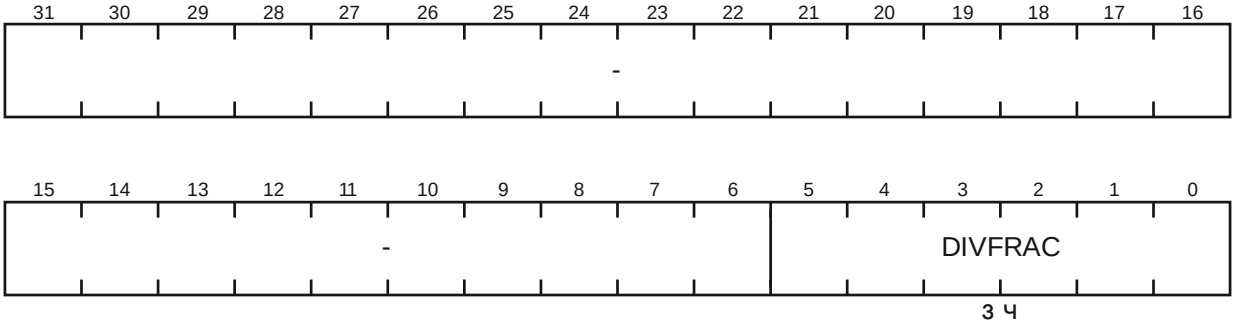
Смещение: + 24h
Сброс: 0h



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 целой дробной части делителя скорости обмена данными

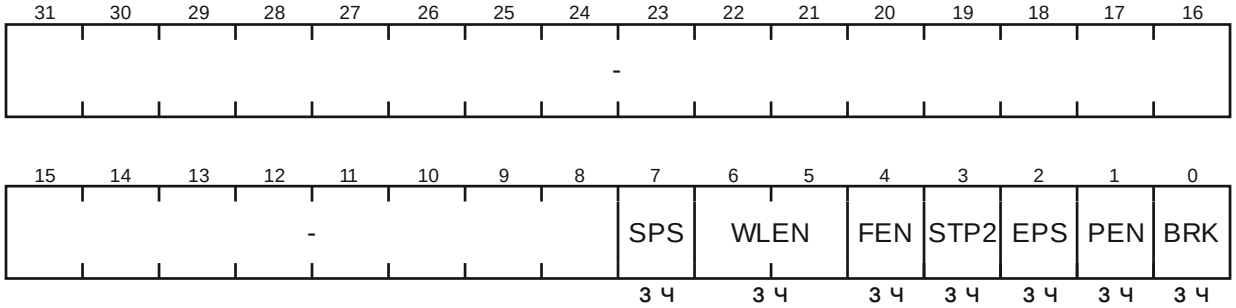
Смещение: + 28h
Сброс: 0h



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
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	

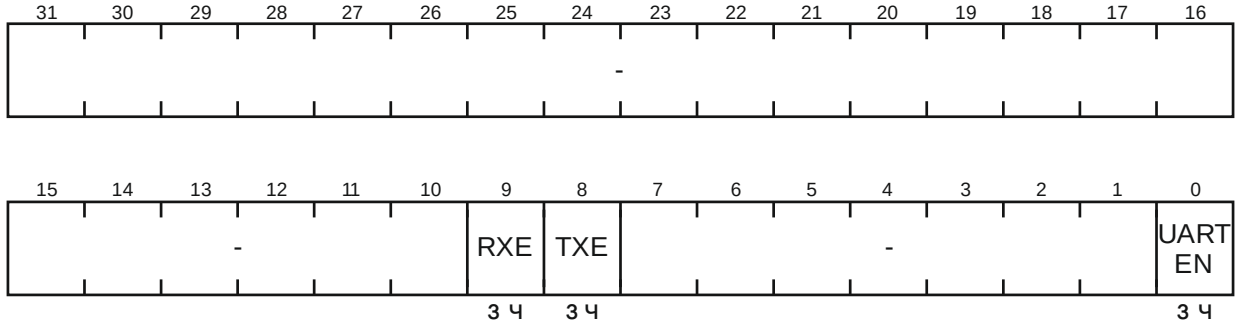
Примечание – Дополнительная информация о бите паритета кадра в таблице А.10.1.

Таблица А.10.1 – 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

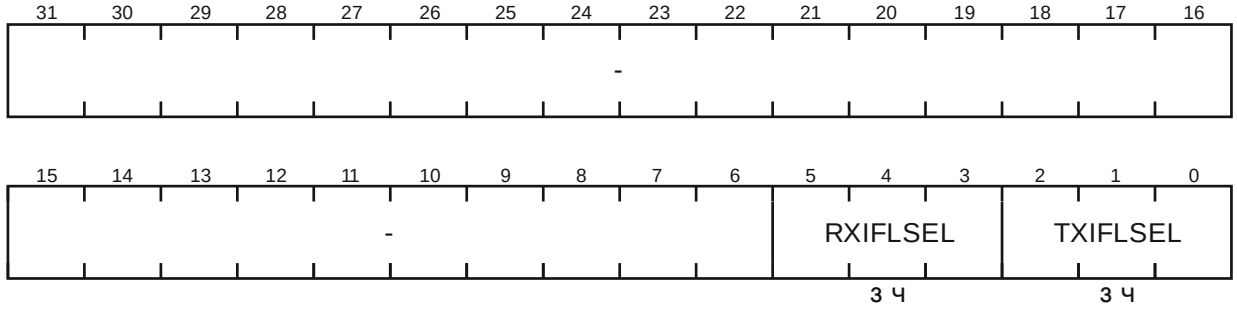
Смещение: + 30h
Сброс: 300h



Поле	Биты	Описание	
RXE	9	Бит разрешения приема	
		0	Запрещено
		1	Разрешено
TXE	8	Бит разрешения передачи	
		0	Запрещено
		1	Разрешено
UARTEN	0	Бит разрешения работы приемопередатчика	
		0	Запрещено
		1	Разрешено
–	31-10, 7-1	Зарезервировано	

IFLS – регистр порога прерывания по заполнению буфера в режиме FIFO

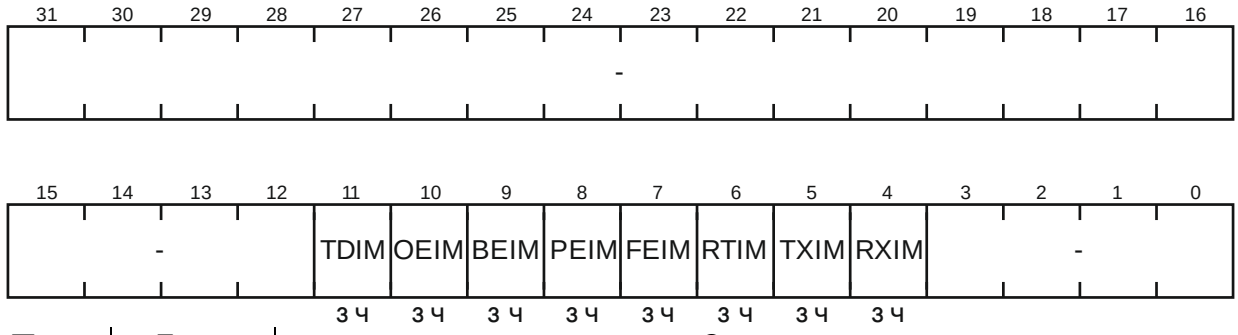
Смещение: + 34h
Сброс: 12h



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

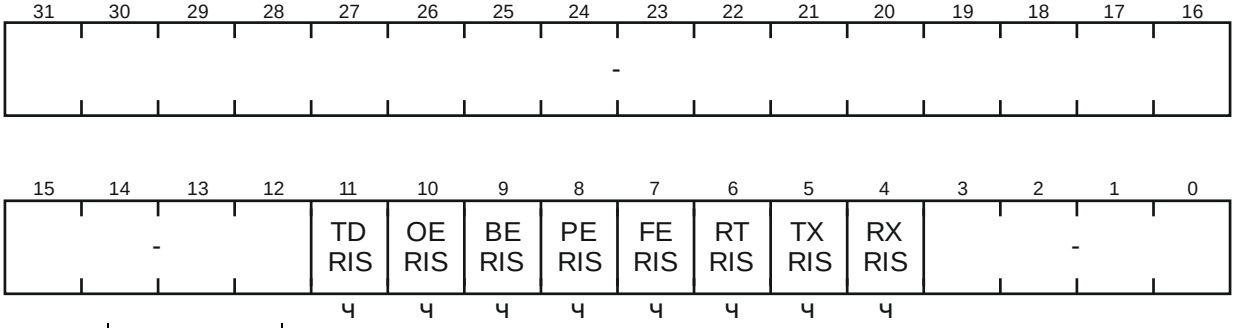


Поле	Биты	Описание
TDIM	11	Окончание передачи в линии
OEIM	10	Переполнение буфера приемника
BEIM	9	Разрыв линии
PEIM	8	Ошибка контроля четности
FEIM	7	Ошибка в структуре кадра
RTIM	6	Таймаут приема данных
TXIM	5	Порог опустошения буфера передатчика
RXIM	4	Порог переполнения буфера приемника
–	31-12, 3-0	Зарезервировано

Примечание – Маска прерываний формируется установкой бит.

RIS – регистр состояния прерываний

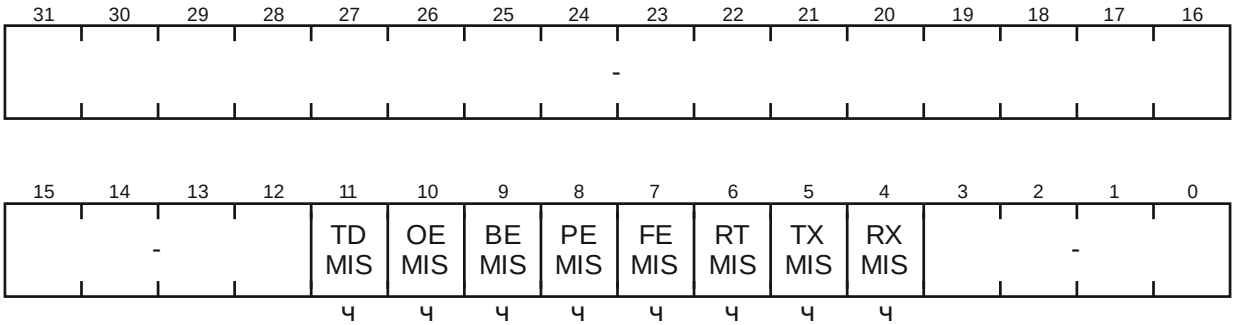
Смещение: + 3Ch
Сброс: 0xh



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 буфера передатчика
RXRIS	4	Флаг порога переполнения буфера приемника
–	31-12, 3-0	Зарезервировано

MIS – регистр состояния прерываний с маскированием

Смещение: + 40h
Сброс: 0xh

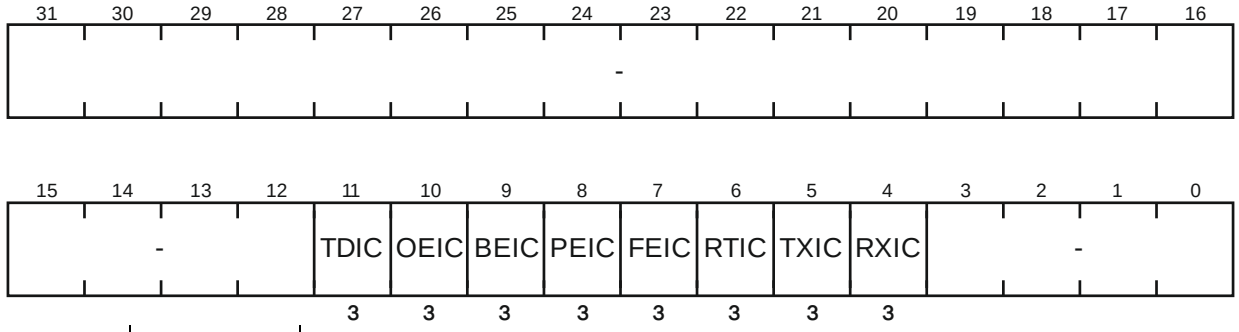


Поле	Биты	Описание
TDMIS	11	Флаг окончания передачи в линии
OEMIS	10	Флаг переполнения буфера приемника
BEMIS	9	Флаг разрыва линии
PEMIS	8	Флаг ошибки контроля четности
FEMIS	7	Флаг ошибки в структуре кадра
RTMIS	6	Флаг таймаута приема данных
TXMIS	5	Флаг порога опустошения буфера передатчика
RXMIS	4	Флаг порога переполнения буфера приемника
–	31-12, 3-0	Зарезервировано

Примечание – Устанавливаются флаги, которые закрыты маской регистра IMSC.

ICR – регистр сброса прерываний

Смещение: + 44h
Сброс: 0h

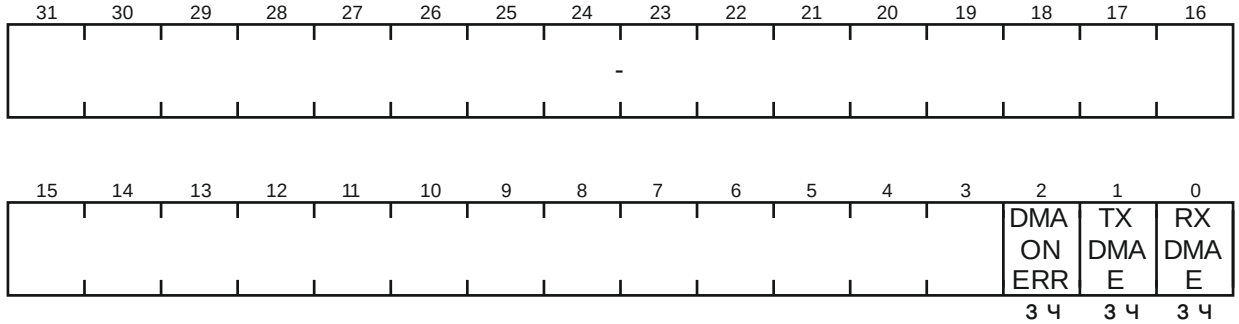


Поле	Биты	Описание
TDIC	11	Бит сброса флага окончания передачи в линии
OEIC	10	Бит сброса флага переполнения буфера приемника
BEIC	9	Бит сброса флага разрыва линии
PEIC	8	Бит сброса флага ошибки контроля четности
FEIC	7	Бит сброса флага ошибки в структуре кадра
RTIC	6	Бит сброса флага таймаута приема данных
TXIC	5	Бит сброса флага порога опустошения буфера передатчика
RXIC	4	Бит сброса флага порога переполнения буфера приемника
–	31-12, 3-0	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



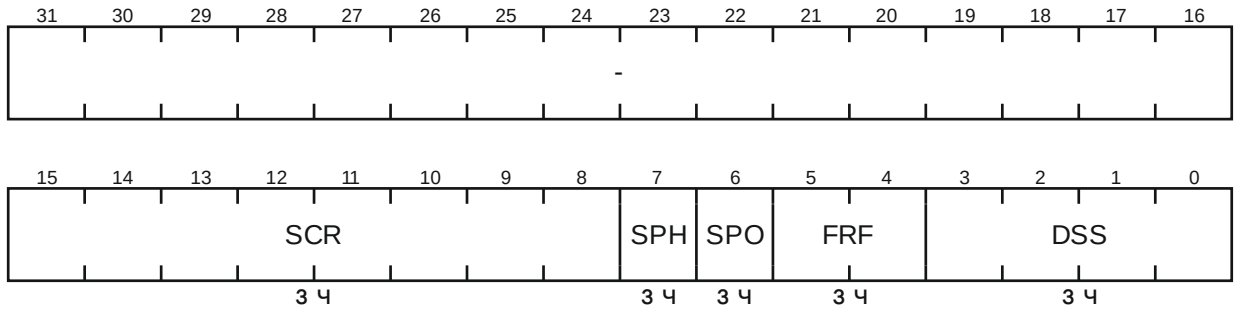
Поле	Биты	Описание
DMAONERR	2	Блокирование запросов UARTRXDMASREQ и UARTRXDMABREQ от приемника в случае прерывания по ошибке
		0 Выключено
		1 Включено
TXDMAE/ RXDMAE	1/ 0	Бит разрешения формирования запросов блока DMA для обслуживания буфера передатчика/приемника
		0 Запрещено
		1 Разрешено
–	31-3	Зарезервировано

A.11 Регистры контроллера SPI

Базовый адрес: 4004_7000h

CR0 – регистр управления 0

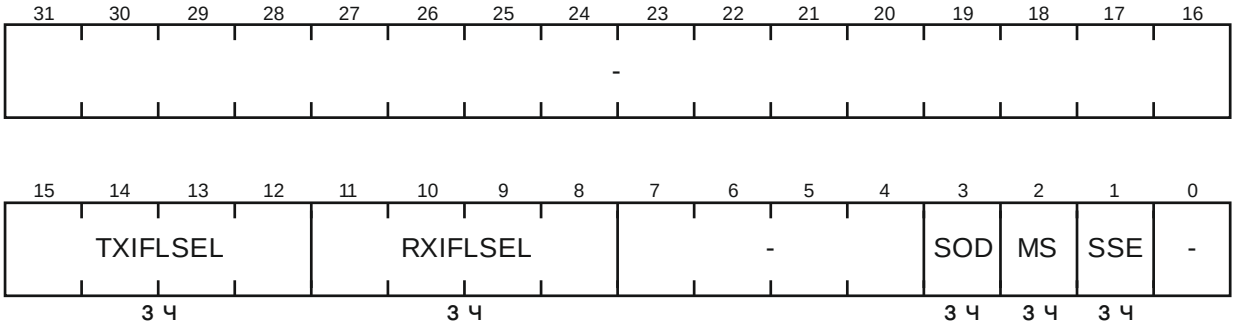
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание	
SCR	15-8	Коэффициент деления второго делителя. Может принимать значения 00h до FFh	
SPH	7	Фаза сигнала SPI_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 SPI_SCK (только для протокола обмена SPI)	
		0	В режиме ожидания линия SPI_SCK удерживается в состоянии логического нуля
		1	В режиме ожидания линия SPI_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

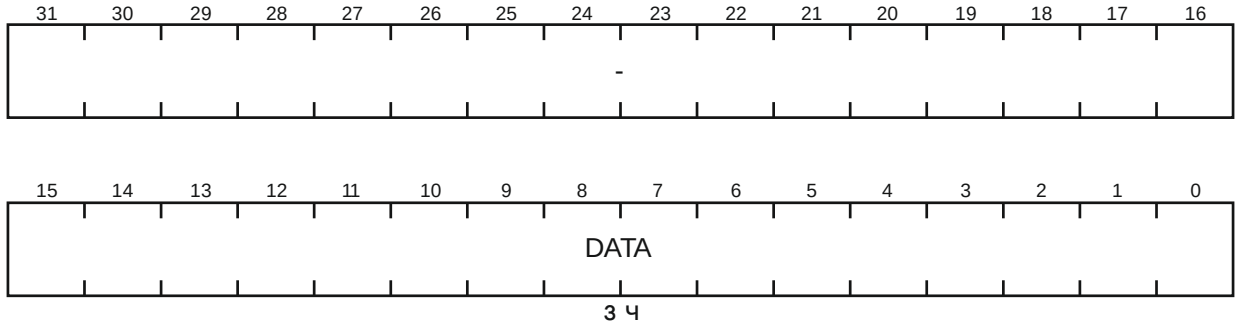
Смещение: + 04h
Сброс: 4400h



Поле	Биты	Описание
TXIFLSEL	15-12	Велична 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	Велична 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 SPI_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

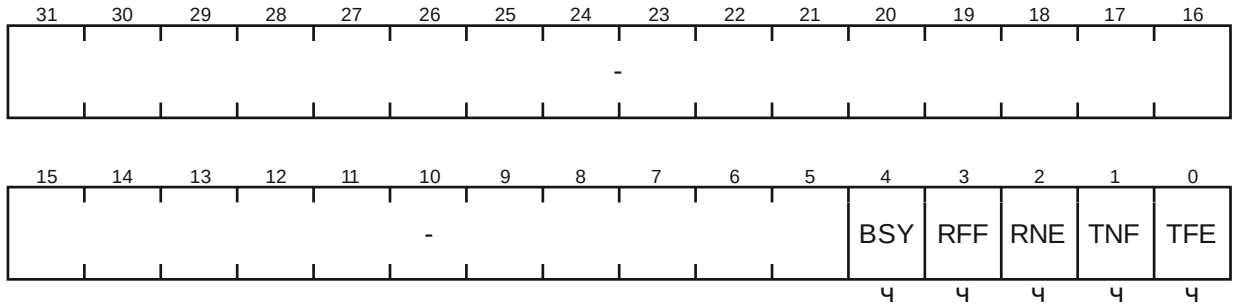
Смещение: + 08h
Сброс: 0h



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

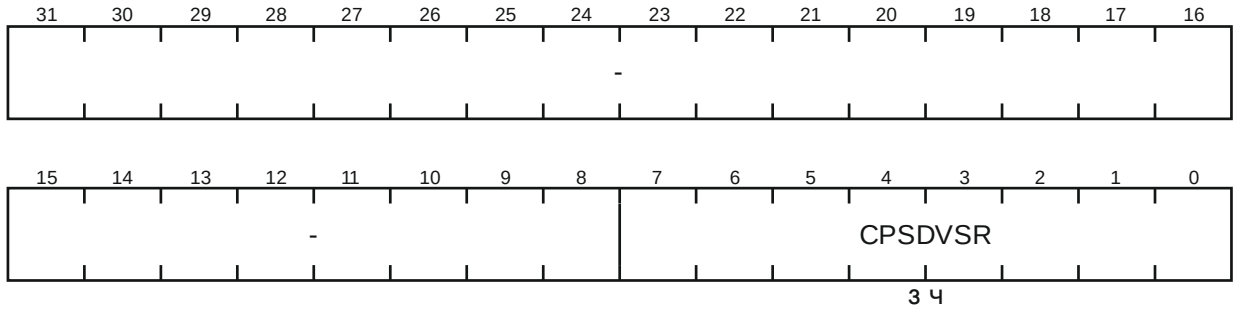
Смещение: + 0Ch
Сброс: 3h



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

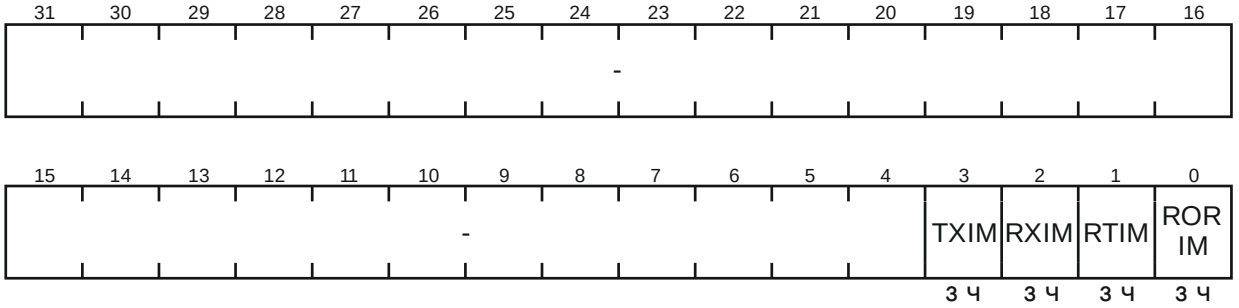
Смещение: + 10h
Сброс: 0h



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

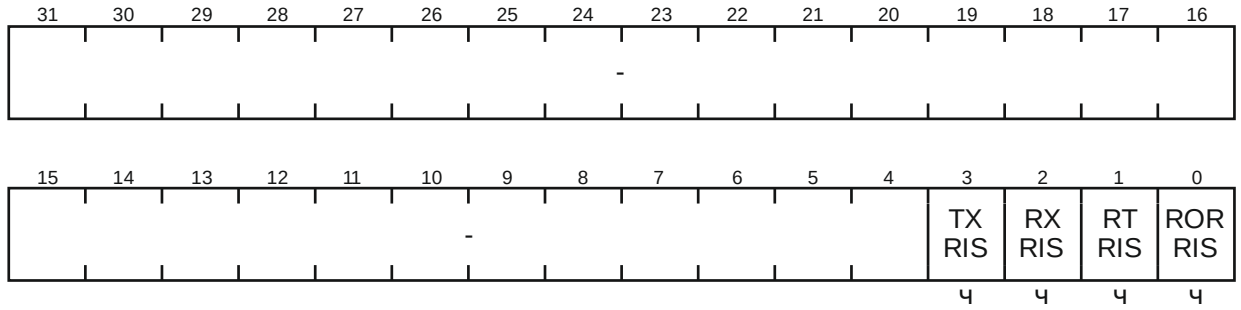


Поле	Биты	Описание
TXIM	3	Буфер передатчика опустошен до величины порога или ниже
RXIM	2	Буфер приемника заполнен на величину порога или выше
RTIM	1	Таймаут приема данных
RORIM	0	Переполнение буфера приемника
–	31-4	Зарезервировано

Примечание – Установка/сброс бит формирует маску. По умолчании все биты сброшены и установка флагов запрещена.

RIS – регистр состояния прерываний

Смещение: + 18h
Сброс: 8h

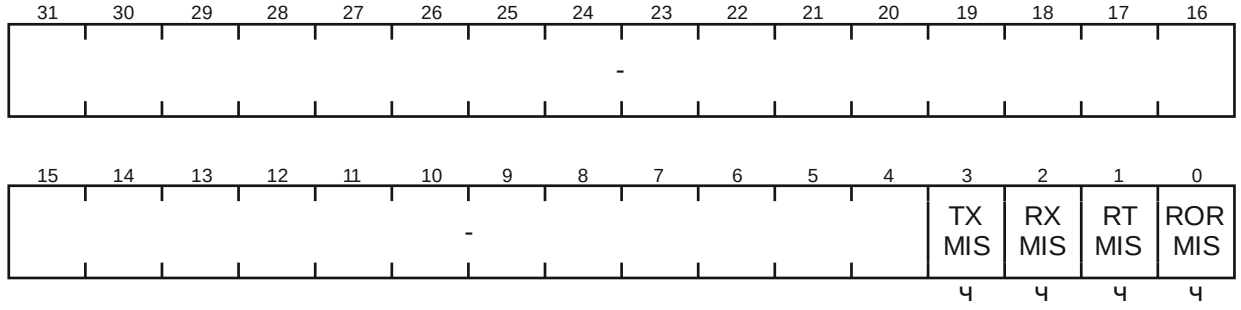


Поле	Биты	Описание
TXRIS	3	Буфер передатчика опустошен до величины порога или ниже
RXRIS	2	Буфер приемника заполнен на величину порога или выше
RTRIS	1	Таймаут приема данных
RORRIS	0	Переполнение буфера приемника
–	31-4	Зарезервировано

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

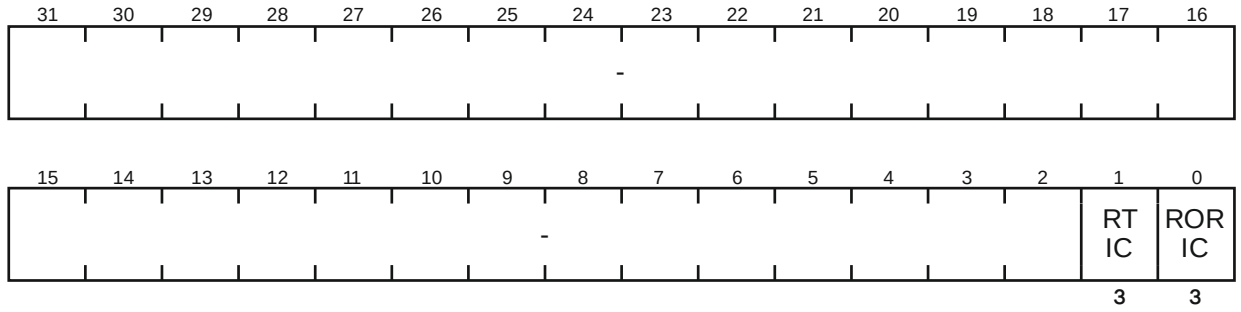


Поле	Биты	Описание
TXMIS	3	Буфер передатчика опустошен до величины порога или ниже
RXMIS	2	Буфер приемника заполнен на величину порога или выше
RTMIS	1	Таймаут приема данных
RORMIS	0	Переполнение буфера приемника
–	31-4	Зарезервировано

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

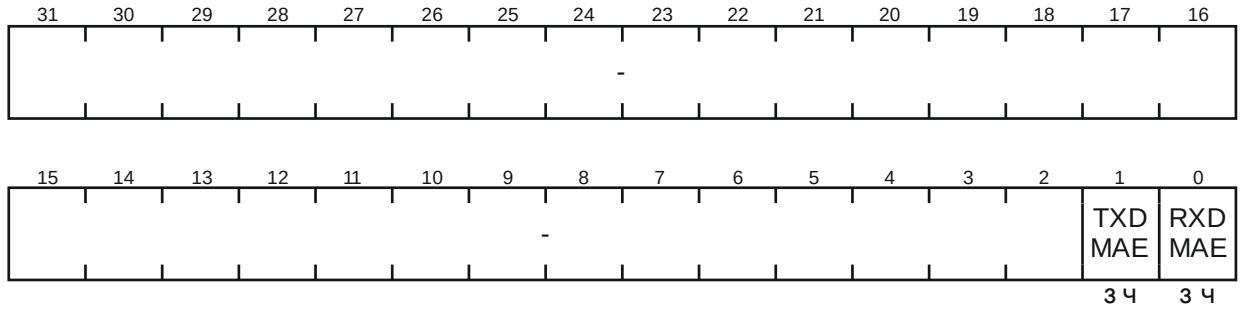


Поле	Биты	Описание
RTIC	1	Таймаут приема данных
RORIC	0	Переполнение буфера приемника
–	31-4	Зарезервировано

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	

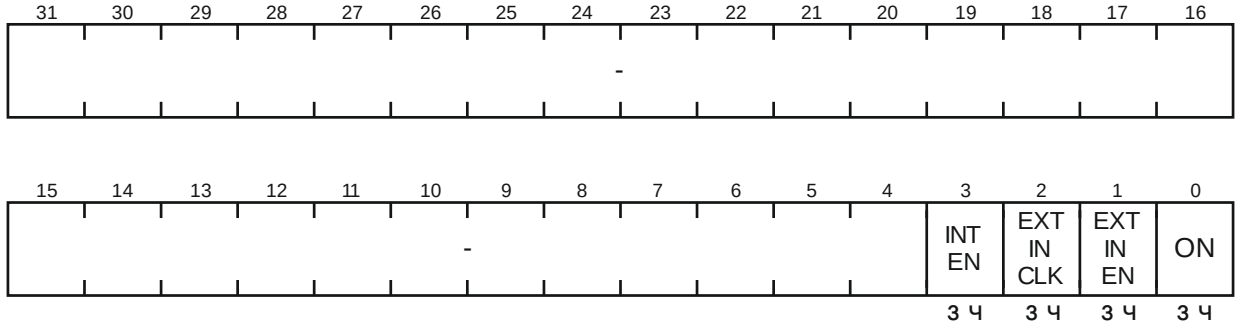
A.12 Регистры таймеров

Базовый адрес:

4004_8000h	Регистры таймера 0
4004_9000h	Регистры таймера 1
4004_A000h	Регистры таймера 2
4004_B000h	Регистры таймера 3

CTRL – регистр управления блока таймера

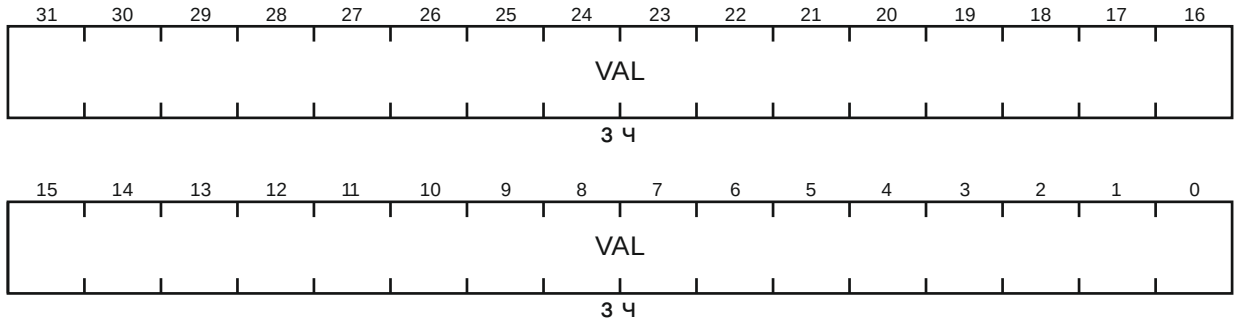
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание
INTEN	3	Бит разрешения прерывания таймера
		0 Запрещено
		1 Разрешено
EXTINCLK	2	Бит включения внешнего входа синхронизации как тактового
		0 Нет действий
		1 Сигнал на входе TMR_IN является тактовым
EXTINEN	1	Бит разрешения работы таймера, если сигнал на внешнем входе равен единице
		0 Запрещено
		1 Таймер декрементируется с частотой PCLK
ON	0	Бит включения таймера
		0 Выключено
		1 Таймер декрементируется с частотой PCLK
–	31-4	Зарезервировано

VALUE – регистр текущего значения таймера

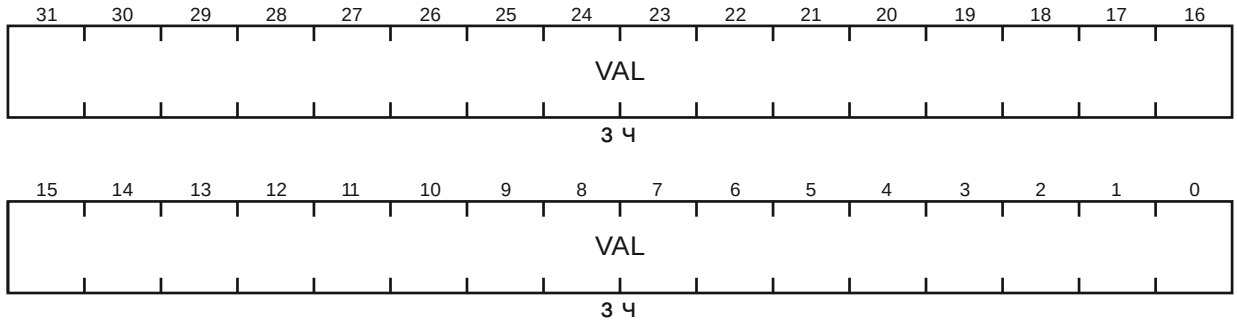
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Текущее значение таймера

LOAD – регистр начального значения счетчика таймера

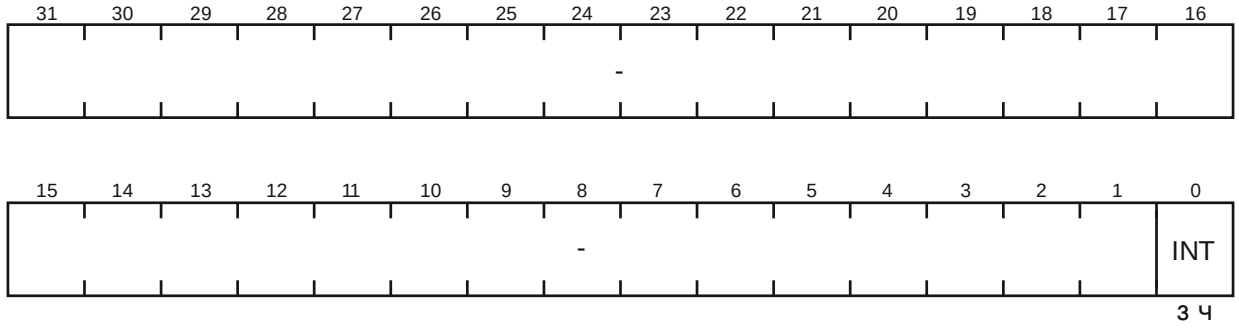
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение перезагрузки таймера

INTSTATUS – регистр прерывания таймера

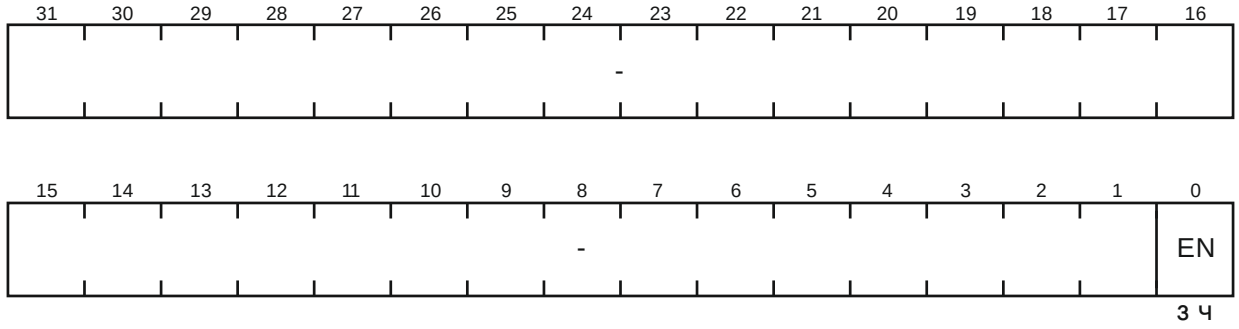
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание	
INT	0	Флаг прерывания таймера	
		0	Нет прерывания
		1	Запрос на прерывание
		Флаг сбрасывается записью единицы	
–	31-1	Зарезервировано	

DMAREQ – регистр управления запросом DMA

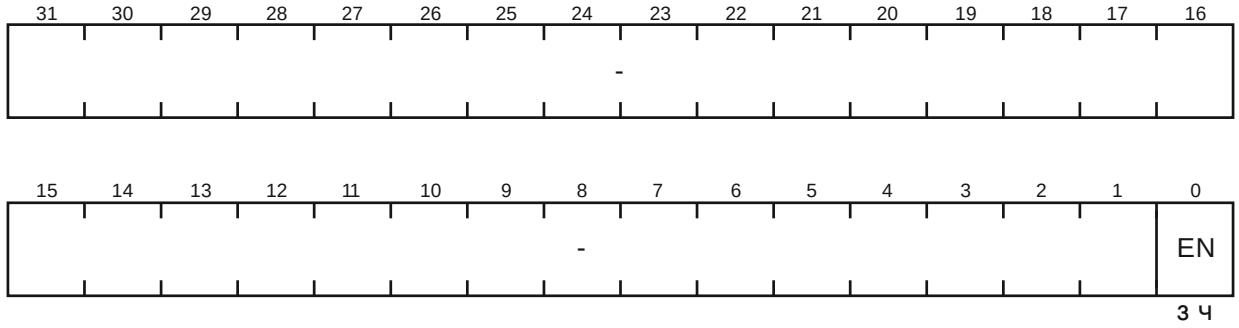
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание	
EN	0	Разрешение генерации запроса контроллера DMA по опустошению таймера	
		0	Нет запроса
		1	Запрос разрешен
–	31-1	Зарезервировано	

ADCSOC – регистр управления запуском АЦП

Смещение: + 14h
Сброс: 0h



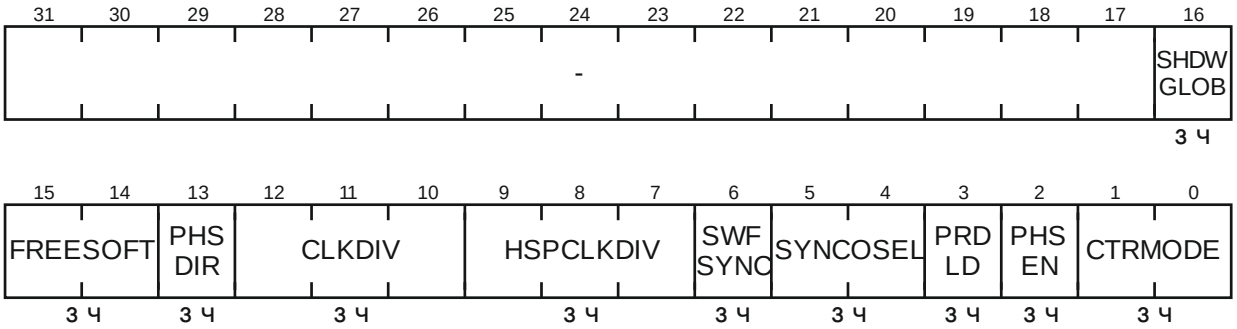
Поле	Биты	Описание	
EN	0	Разрешение генерации сигнала запуска АЦП по опустошению таймера	
		0	Нет сигнала
		1	Генерация разрешена
–	31-1	Зарезервировано	

A.13 Регистры блока ШИМ

Базовый адрес: 4004_C000h Регистры блока ШИМ0
4004_D000h Регистры блока ШИМ1
4004_E000h Регистры блока ШИМ2

TBCTL – регистр управления таймером

Смещение: + 00h
Сброс: 0001_8000h

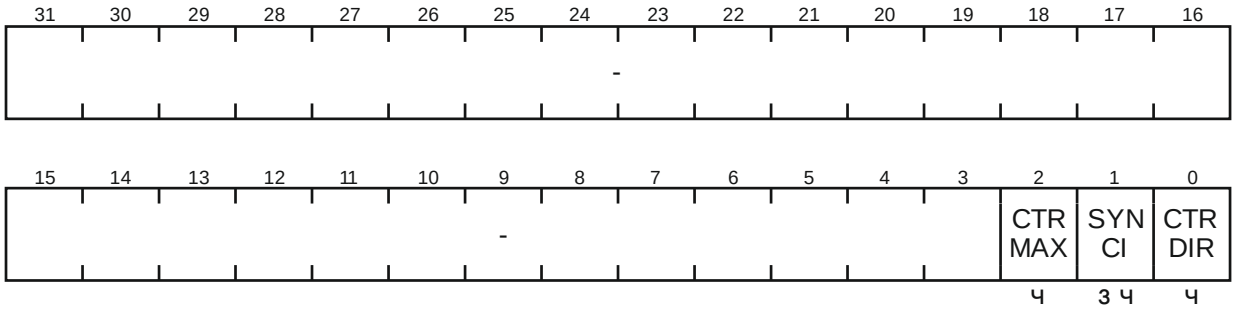


Поле	Биты	Описание	
SHDWGLOB	16	Глобальное разрешение всех теневых загрузок	
		0	Все теневые регистры пишутся, но события перезаписи в активные блокируются
		1	Теневая загрузка происходит согласно настройкам соответствующих полей
FREESOFT	15-14	Поле задания поведения счетчика ШИМ после перехода в режим останова во время отладки	
		00	Счетчик будет остановлен на следующий такт TBCLK
		01	Счетчик будет остановлен по достижении события: - TBCTR = TBPRD (при счете вверх); - TBCTR = 0000h (при счете вниз или вверх-вниз)
		1x	Счетчик продолжит работу
PHSDIR	13	Бит задания фазового направления (используется только при двунаправленном счете). Задает направление счета после синхронизации. Загружается вместе с регистром фазы TBPHS	
		0	Вниз
		1	Вверх
CLKDIV	12-10	Поле задания первого делителя тактовой частоты	
		000	1
		001	1/2
		010	1/4
		011	1/8
		100	1/16
		101	1/32
		110	1/64
		111	1/128

Поле	Биты	Описание	
HSPCLKDIV	9-7	Поле задания второго делителя тактовой частоты. Конечное значение делителя является произведением значений делителей, задаваемых полями CLKDIV и HSPCLKDIV	
		000	1
		001	1/2
		010	1/4
		011	1/6
		100	1/8
		101	1/10
		110	1/12
		111	1/14
SWFSYNC	6	Бит программной эмуляции появления синхроимпульса	
		0	Нет действий
		1	Запись единицы вызывает появление синхроимпульса в цепи PWM_SYNCI
SYNCOSEL	5-4	Поле выбора источника для выходного сигнала синхронизации PWM_SYNCO	
		00	PWM_SYNCI
		01	CTR = 0000h
		10	CTR = CMPB
		11	Выдача синхроимпульса запрещена
PRDLD	3	Бит управления загрузкой регистра TBPRD	
		0	Режим отложенной загрузки регистра TBPRD разрешен
		1	Запись в TBPRD будет произведена сразу в активный регистр
PHSEN	2	Бит разрешения загрузки счетчика таймера	
		0	Запрещено
		1	Разрешена загрузка счетчика TBCTR значением регистра фазы TBPHS при получении события синхронизации (импульс на входе PWM_SYNCI или запись в бит SWFSYNC)
CTRMODE	1-0	Поле задания направления счета	
		00	Вверх
		01	Вниз
		10	Вверх-вниз
		11	Счетчик остановлен
–	31-17	Зарезервировано	

TBSTS – регистр статуса таймера

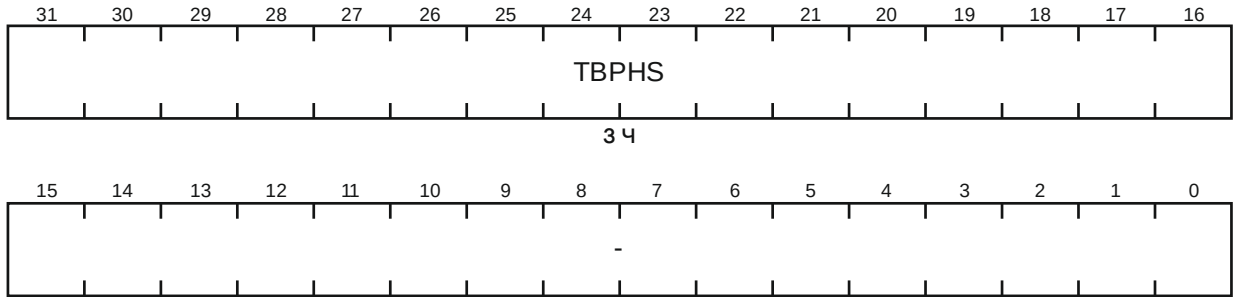
Смещение: + 04h
Сброс: 0h



Поле	Бит	Описание	
CTRMAX	2	Флаг достижения счетчиком таймера своего максимального значения FFFFh	
		0	Значение не достигнуто или флаг был сброшен
		1	Значение было достигнуто
		Запись единицы сбрасывает флаг	
SYNCI	1	Флаг синхронизации	
		0	Синхронизация не достигнута или флаг был сброшен
		1	Синхронизация произошла
		Запись единицы сбрасывает флаг	
CTRDIR	0	Флаг направления счета таймера	
		0	Вниз
		1	Вверх
–	31-3	Зарезервировано	

TBPHS – регистр фазы таймера

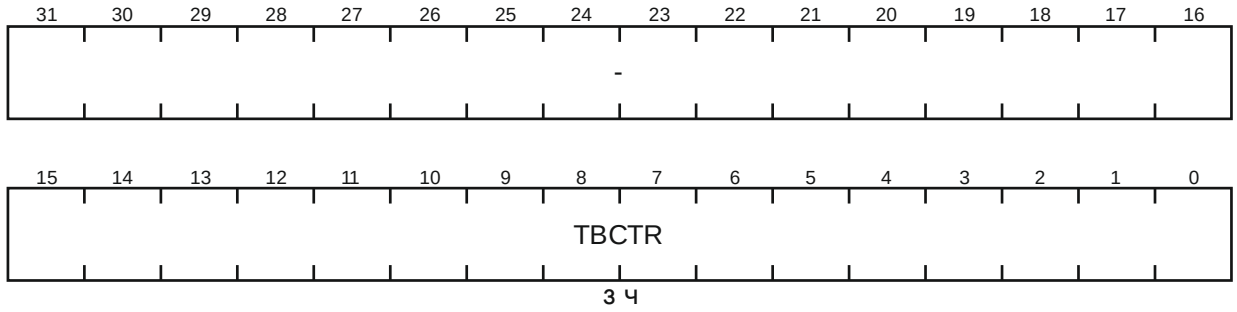
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание
TBPHS	31-16	Поле задания начальной фазы таймера при получении сигнала синхронизации
–	15-0	Зарезервировано

TBCTR – регистр текущего значения таймера

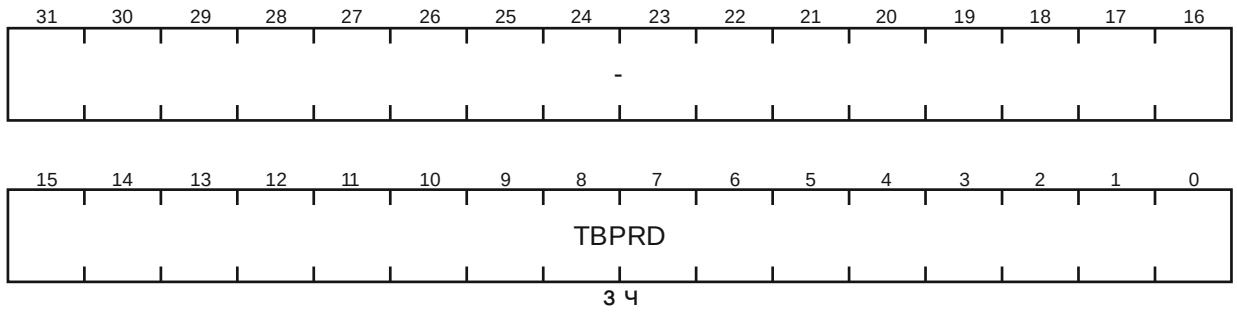
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание
TBCTR	15-0	Текущее значение счетчика таймера. Запись в регистр изменяет значение таймера. Запись происходит асинхронно с TBCLK и не использует отложенный механизм загрузки
–	31-16	Зарезервировано

TBPRD – регистр периода таймера

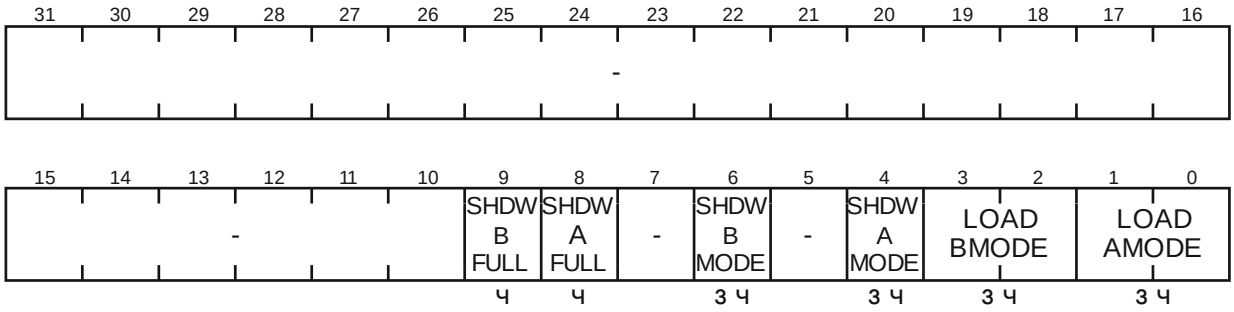
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
TBPRD	15-0	Период таймера (максимальное значение счета таймера). Отложенная загрузка в этот регистр программируется битом PRDLD регистра TBCTL. По умолчанию бит PRDLD сброшен и запись в регистр TBPRD приводит к записи в теневой регистр. Активный регистр будет загружен по событию TBCTR = Zero. Если бит PRDLD установлен, то запись выполняется напрямую в активный регистр
–	31-16	Зарезервировано

CMPCTL – регистр управления компаратором

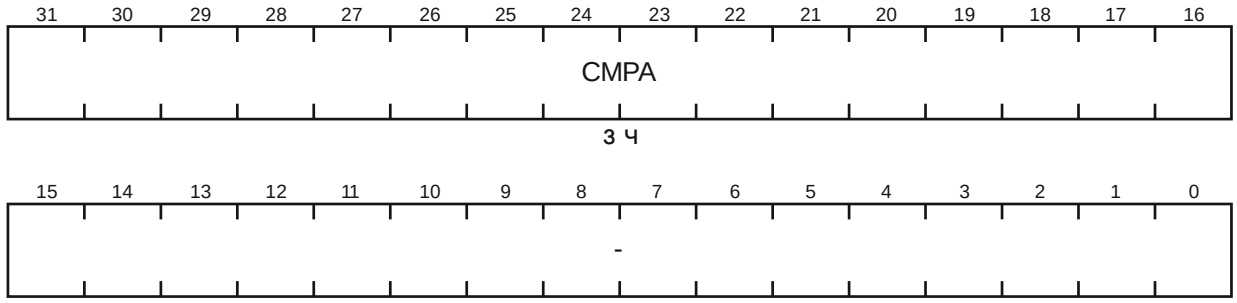
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание	
SHDWBFULL	9	Флаг отложенной загрузки в регистр CMPB	
		0	Нет действий
		1	Активный регистр загружен значением из теневого регистра
SHDWAFULL	8	Флаг отложенной загрузки в регистр CMPA	
		0	Нет действий
		1	Активный регистр загружен значением из теневого регистра
SHDWBMODE	6	Бит управления загрузкой регистра CMPB	
		0	Значение, записываемое в регистр CMPB, размещается в теневом регистре (отложенная загрузка)
		1	Производится загрузка напрямую в активный регистр
SHDWAMODE	4	Бит управления загрузкой регистра CMPA	
		0	Значение, записываемое в регистр CMPA, размещается в теневом регистре (отложенная загрузка)
		1	Производится загрузка напрямую в активный регистр
LOADBMODE	3-2	Поле задания события загрузки отложенного значения в регистр CMPB (при условии, что бит SHDWBMODE сброшен)	
		00	CTR = Zero
		01	CTR = PRD
		10	CTR = Zero или CTR = PRD
		11	Загрузка запрещена
LOADAMODE	1-0	Поле задания события загрузки отложенного значения в регистр CMPA (при условии, что бит SHDWAMODE сброшен)	
		00	CTR = Zero
		01	CTR = PRD
		10	CTR = Zero или CTR = PRD
		11	Загрузка запрещена
–	31-10, 7, 5	Зарезервировано	

CMPA – регистр порога срабатывания канала A

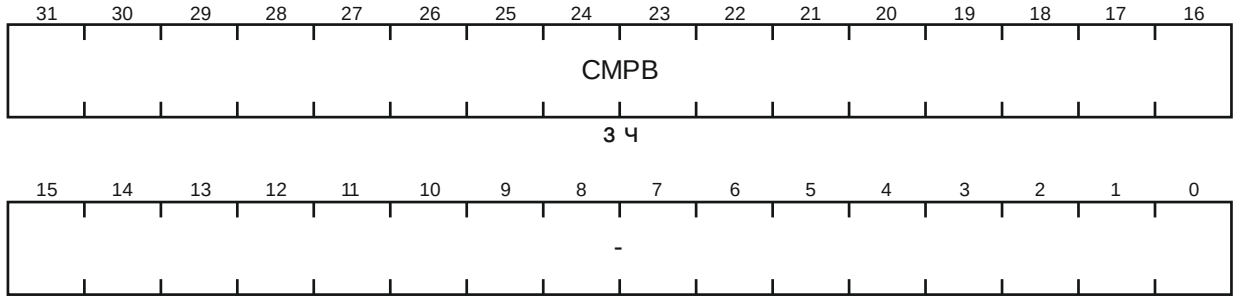
Смещение: + 18h
Сброс: 0h



Поле	Биты	Описание
CMPA	31-16	Активное значение порога срабатывания канала A, которое сравнивается со значением счетчика таймера. При совпадении значений формируется событие CTR = CMPA, которое влияет на поведение сигналов на линиях PWMx_A и PWMx_B
–	15-0	Зарезервировано

CMPB – регистр порога срабатывания канала B

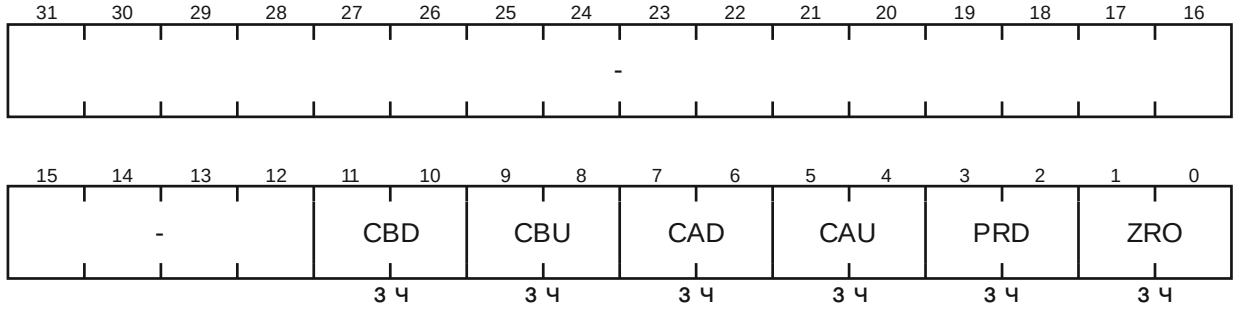
Смещение: + 1Ch
Сброс: 0h



Поле	Биты	Описание
CMPB	31-16	Активное значение порога срабатывания канала B, которое сравнивается со значением счетчика таймера. При совпадении значений формируется событие CTR = CMPB, которое влияет на поведение сигналов на линиях PWMx_A и PWMx_B
–	15-0	Зарезервировано

AQCTLA – регистр обработчика для выхода A

Смещение: + 20h
Сброс: 0h

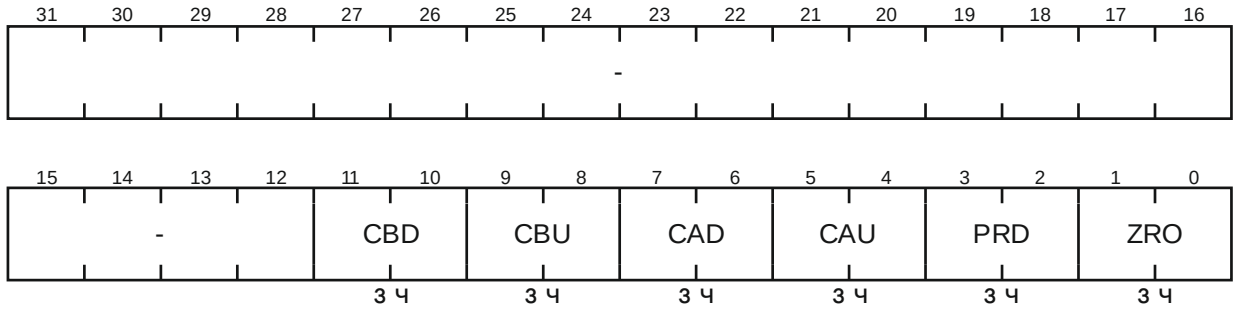


Поле	Биты	Описание
CBD	11-10	Действие на выводе PWMx_A при CTR = CMPB при счете вниз
CBU	9-8	Действие на выводе PWMx_A при CTR = CMPB при счете вверх
CAD	7-6	Действие на выводе PWMx_A при CTR = CMPA при счете вниз
CAU	5-4	Действие на выводе PWMx_A при CTR = CMPA при счете вверх
PRD	3-2	Действие на выводе PWMx_A при CTR = PRD
ZRO	1-0	Действие на выводе PWMx_A при CTR = Zero
–	31-12	Зарезервировано

Примечание – Для каждого события может быть задано одно из четырех действий:
00 – нет реакции;
01 – переключение вывода PWMx_A в ноль;
10 – переключение вывода PWMx_A в единицу;
11 – инверсия вывода PWMx_A.

AQCTLB – регистр обработчика для выхода B

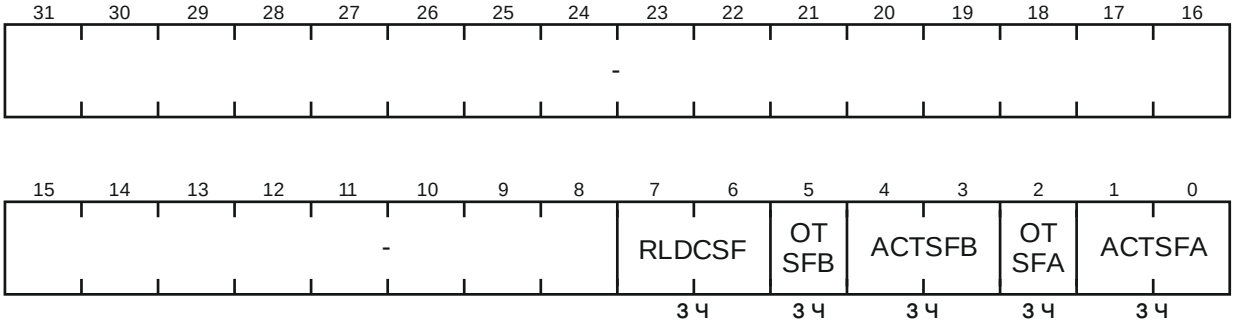
Смещение: + 24h
Сброс: 0h



Примечание – Назначения битовых полей аналогичны назначению полей регистра AQCTLA с той разницей, что относятся они к выводу PWMx_B.

AQSFRC – регистр программного управления однократным действием

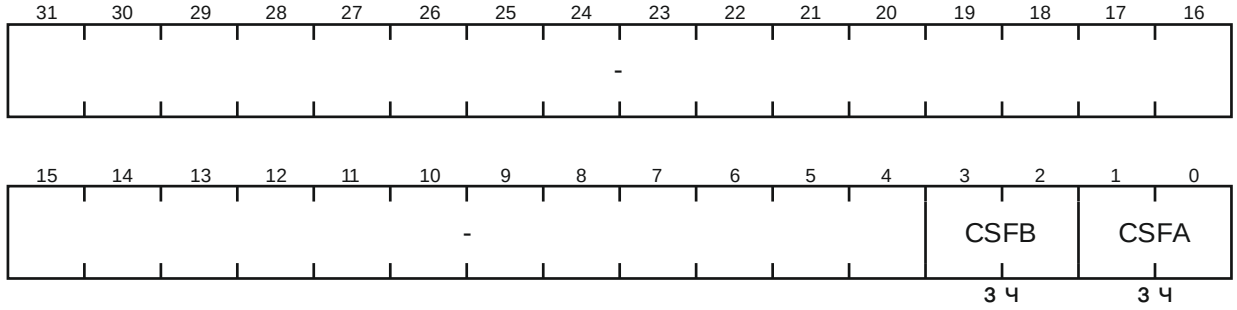
Смещение: + 28h
Сброс: 0h



Поле	Биты	Описание	
RLDCSF	7-6	Управление событием теневой загрузки полей OTSFA, OTSFB, а также полей CSFA, CSFB регистра AQCSFRC	
		00	CTR = Zero
		01	CTR = PRD
		10	CTR = Zero или CTR = PRD
		11	Без теневой загрузки, прямая запись в регистр
OTSFB	5	Запись единицы приводит к однократному переключению вывода в состояние, согласно ACTSFB	
ACTSFB	4-3	Выбор действия с выходным сигналом на выводе	
		00	Нет действий
		01	PWMx_B = 0
		10	PWMx_B = 1
		11	Инверсия вывода PWMx_B
OTSFA	2	Запись единицы приводит к однократному переключению вывода в состояние согласно ACTSFA	
ACTSFA	1-0	Выбор действия с выходным сигналом на выводе	
		00	Нет действий
		01	PWMx_A = 0
		10	PWMx_A = 1
		11	Инверсия вывода PWMx_A
–	31-8	Зарезервировано	

AQCSFRC – регистр обработчика для циклического программного управления

Смещение: + 2Ch
Сброс: 0h

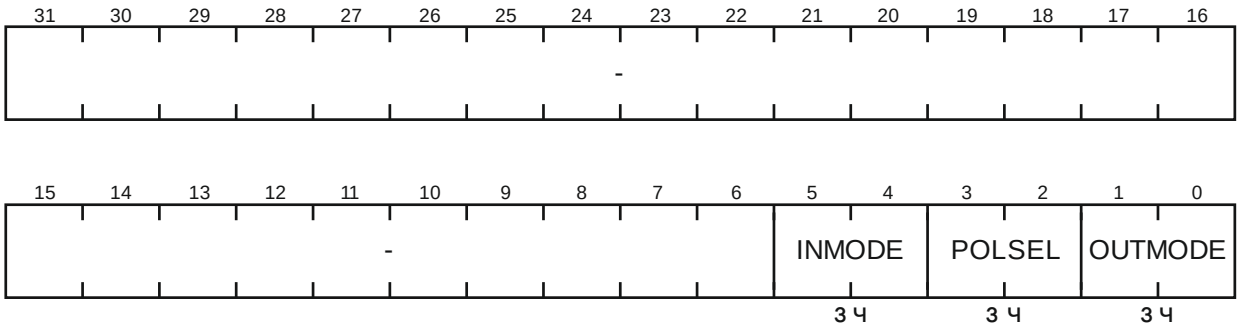


Поле	Биты	Описание
CSFB/ CSFA	3-2/ 1-0	Поле задания циклического воздействия на выход PWMx_B/PWMx_A
–	31-4	Зарезервировано

Примечание – Может быть задано одно из четырех воздействий:
00, 11 – нет реакции;
01 – значение 0 на выходе PWMx_B/PWMx_A;
10 – значение 1 на выходе PWMx_B/PWMx_A.

DBCTL – регистр управления генератором «мертвого» времени ШИМ

Смещение: + 30h
Сброс: 0h

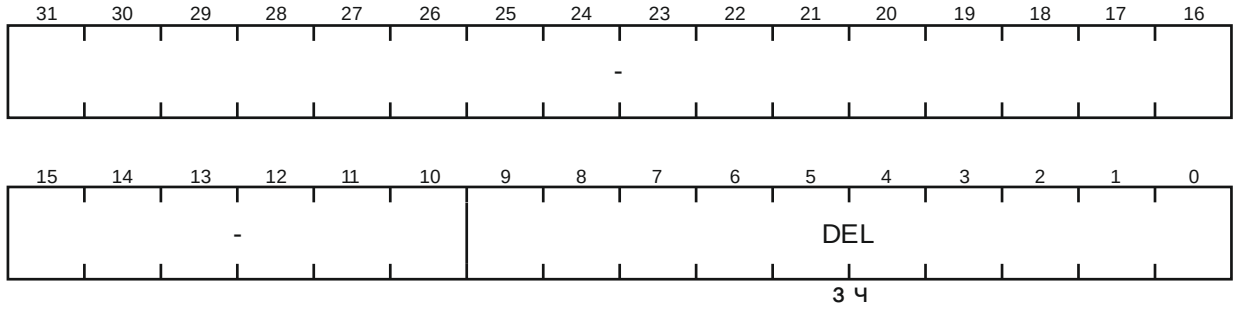


Поле	Биты	Описание	
INMODE	5-4	Поле выбора источника для контроля по фронту и срезу. Старший бит поля управляет ключом S5, младший – ключом S4, см. рисунок 14.12	
		00	Входной сигнал PWMA используется для контроля по переднему и заднему фронтам
		01	Входной сигнал PWMA используется для контроля по заднему фронту, а сигнал PWMB – по переднему
		10	Входной сигнал PWMA используется для контроля по переднему фронту, а сигнал PWMB – по заднему
		11	Входной сигнал PWMB используется для контроля по переднему и заднему фронтам

Поле	Биты	Описание	
POLSEL	3-2	Поле задания полярности сигнала на выходе. Старший бит поля управляет ключом S3, а младший – ключом S2, см. рисунок 14.12	
		00	Инверсия запрещена
		01	Инверсия только на выходе PWMx_A
		10	Инверсия только на выходе PWMx_B
		11	Инверсия на выходах PWMx_A и PWMx_B
OUTMODE	1-0	Поле выбора фронта, для которого включена задержка («мертвое» время). Старший бит поля управляет ключом S1, а младший – ключом S0, см. рисунок 14.12	
		00	Не задано
		01	Задний фронт выхода PWMx_B
		10	Передний фронт выхода PWMx_A
		11	Передний фронт выхода PWMx_A и задний фронт выхода PWMx_B
–	31-6	Зарезервировано	

DBRED – регистр управления «мертвым» временем переднего фронта

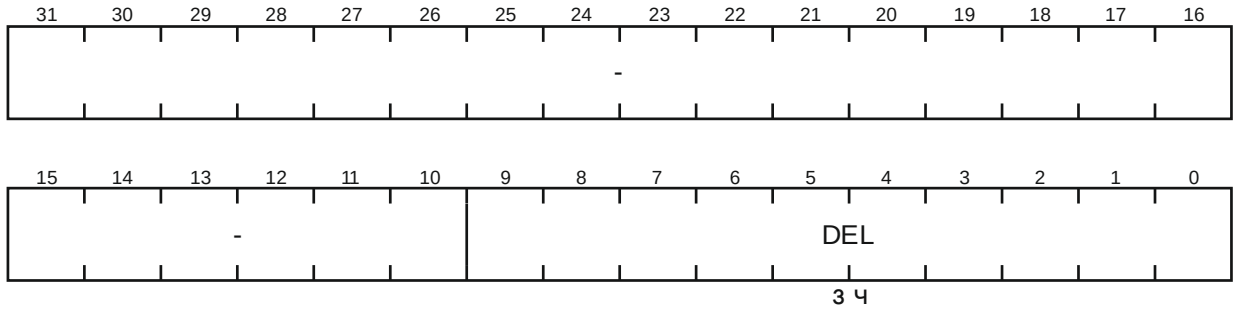
Смещение: + 34h
Сброс: 0h



Поле	Биты	Описание
DEL	9-0	Величина задержки переднего фронта для генератора «мертвого времени» ШИМ (в периодах тактового сигнала TBCLK)
–	31-10	Зарезервировано

DBFED – регистр управления «мертвым» временем заднего фронта

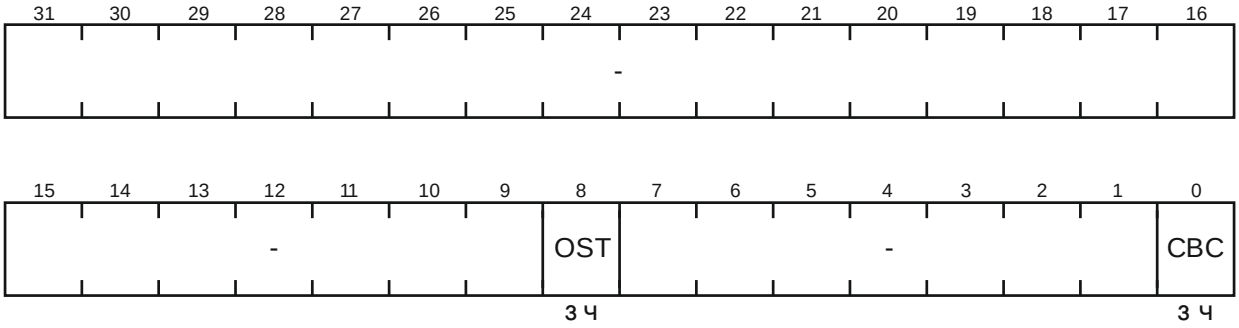
Смещение: + 38h
Сброс: 0h



Поле	Биты	Описание
DEL	9-0	Величина задержки заднего фронта для генератора «мертвого» времени ШИМ (в периодах тактового сигнала TBCLK)
–	31-10	Зарезервировано

TZSEL – регистр источника сигнала аварии

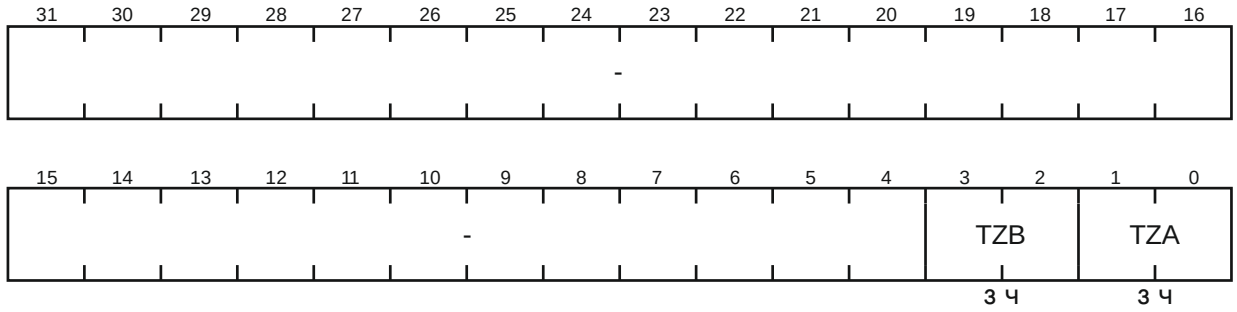
Смещение: + 3Ch
Сброс: 0h



Поле	Биты	Описание	
OST	8	Бит разрешения источника сигнала аварии с вывода PWM_TZ в однократном режиме	
		0	Запрещено
		1	Разрешено
CBC	0	Бит разрешения источника сигнала аварии с вывода PWM_TZ в циклическом режиме	
		0	Запрещено
		1	Разрешено
–	31-9, 7-1	Зарезервировано	

TZCTL – регистр управления детектором сигнала аварии

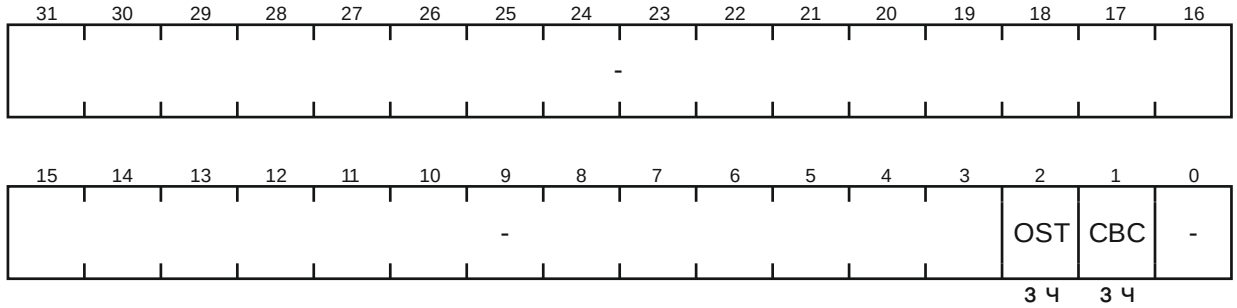
Смещение: + 40h
Сброс: 0h



Поле	Биты	Описание	
TZB/ TZA	3-2/ 1-0	Поле задания поведения вывода PWMx_B/PWMx_A в случае получения сигнала аварии. Источник сигнала аварии при этом определяется регистром TZSEL	
		00	Переключение в третье состояние
		01	Переключение в единицу
		10	Переключение в ноль
		11	Нет действий
–	31-4	Зарезервировано	

TZEINT – регистр маски прерывания детектора сигнала аварии

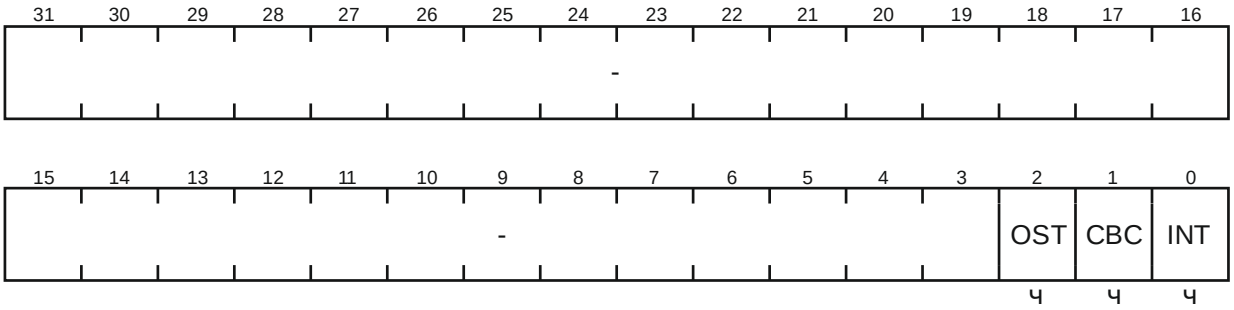
Смещение: + 44h
Сброс: 0h



Поле	Биты	Описание	
OST	2	Бит разрешения генерации прерывания в однократном режиме обработки аварии	
		0	Запрещено
		1	Разрешено
CBC	1	Бит разрешения генерации прерывания в циклическом режиме обработки аварии	
		0	Запрещено
		1	Разрешено
–	31-3, 0	Зарезервировано	

TZFLG – регистр флагов прерывания детектора сигнала аварии

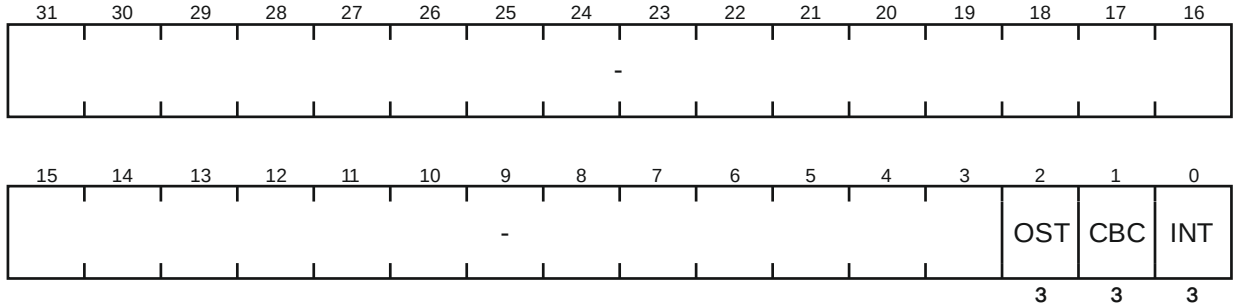
Смещение: + 48h
Сброс: 0h



Поле	Биты	Описание
OST	2	Флаг прерывания в однократном режиме обработки аварии
		0 Нет прерывания
		1 Запрос на прерывание
		При этом действие на выходе продолжается вплоть до обнуления счетчика таймера, если сигнал аварии не перестал быть активным к этому моменту. Если флаг сброшен, а источник сигнала аварии остался, флаг установится снова
CBC	1	Флаг прерывания в циклическом режиме обработки аварии
		0 Нет прерывания
		1 Запрос на прерывание
		При этом действие на выходе продолжается вплоть до обнуления счетчика таймера, если сигнал аварии не перестал быть активным к этому моменту. Если флаг сброшен, а источник сигнала аварии остался, флаг установится снова
INT	0	Флаг внешнего прерывания NVIC
		0 Нет прерывания
		1 Запрос на прерывание
		Если флаг был сброшен, а один из флагов CBC или OST установлен, флаг установится снова
–	31-3	Зарезервировано

TZCLR – регистр сброса флагов прерываний детектора сигнала аварии

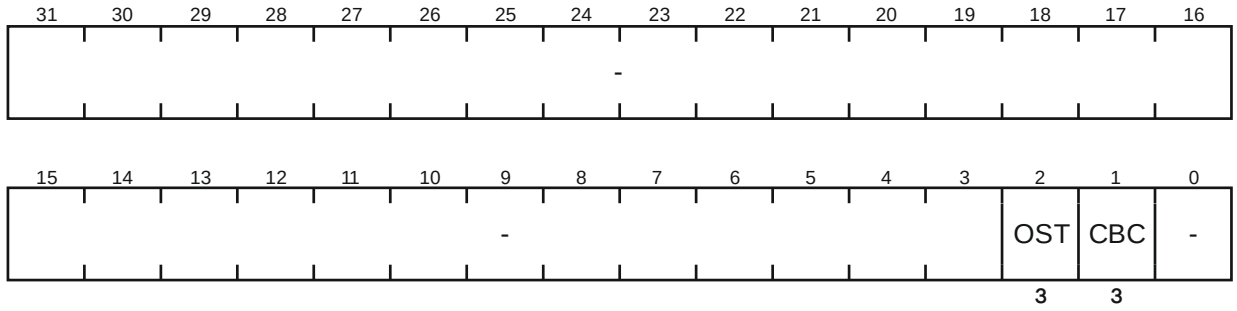
Смещение: + 4Ch
Сброс: 0h



Поле	Биты	Описание
OST	2	Бит сброса флага прерывания в однократном режиме обработки аварии. Запись единицы сбрасывает бит OST в регистре TZFLG
CBC	1	Бит сброса флага прерывания в циклическом режиме обработки аварии. Запись единицы сбрасывает бит CBC в регистре TZFLG
INT	0	Бит сброса флага внешнего прерывания NVIC. Запись единицы сбрасывает бит INT в регистре TZFLG
–	31-3	Зарезервировано

TZFRC – регистр программной эмуляции сигнала аварии

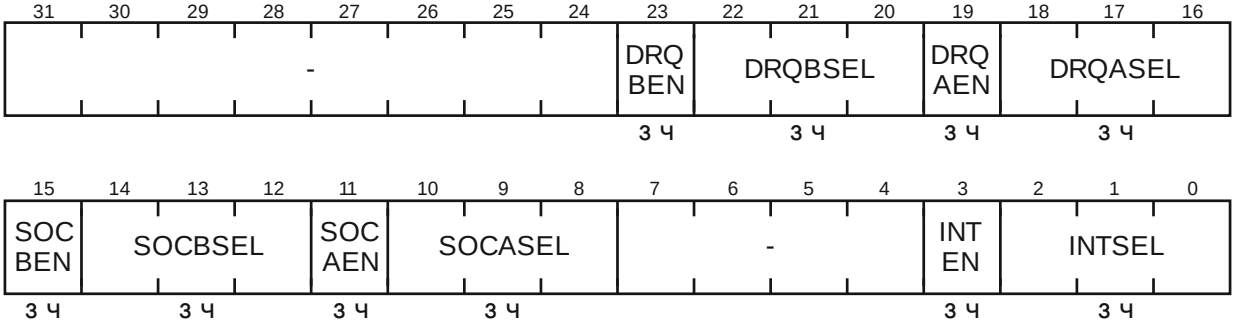
Смещение: + 50h
Сброс: 0h



Поле	Биты	Описание
OST	2	Бит программной генерации сигнала аварии в однократном режиме. Запись единицы устанавливает бит OST в регистре TZFLG Чтение бита возвращает ноль
CBC	1	Бит программной генерации сигнала аварии в циклическом режиме. Запись единицы устанавливает бит CBC в регистре TZFLG Чтение бита возвращает ноль
–	31-3, 0	Зарезервировано

ETSEL – регистр источника триггера событий

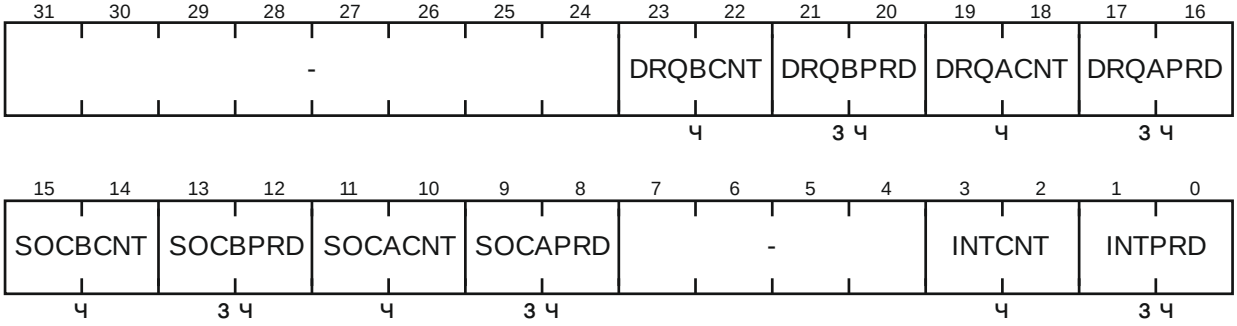
Смещение: + 54h
Сброс: 0h



Поле	Биты	Описание	
DRQBEN/ DRQAEN	23/ 19	Бит разрешения генерации запроса DMA	
		0	Запрещено
		1	Разрешено
SOCBEN/ SOCAEN	15/ 11	Бит разрешения генерации внешнего сигнала PWM_SOCB/PWM_SOCA для запуска АЦП	
		0	Запрещено
		1	Разрешено
DRQBSEL/ DRQASEL/ SOCBSEL/ SOCASEL/ INTSEL	22-20/ 18-16 14-12/ 10-8/ 2-0	Поле выбора события, по которому будет сформирован импульс DMA_REQB/DMA_REQA/PWM_SOCB/PWM_SOCA/PWM_INT	
		000	Зарезервировано
		001	CTR = Zero
		010	CTR = PRD
		011	Зарезервировано
		100	CTR = CMPA при счете вверх
		101	CTR = CMPA при счете вниз
		110	CTR = CMPB при счете вверх
		111	CTR = CMPB при счете вниз
INTEN	3	Бит разрешения генерации внешнего прерывания PWM_INT	
		0	Запрещено
		1	Разрешено
–	31-24, 7-4	Зарезервировано	

ETPS – регистр предделителя триггера событий

Смещение: + 58h
Сброс: 0h

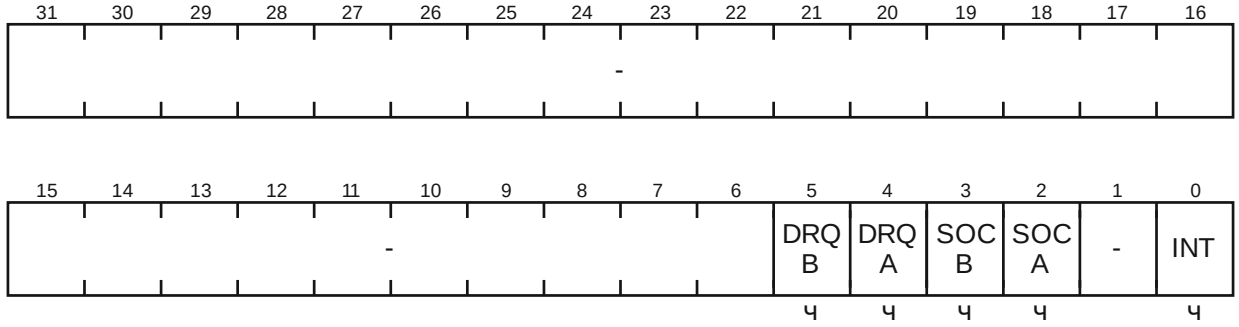


Поле	Биты	Описание	
DRQBCNT/ DRQACNT/ SOCBCNT/ SOCACNT	23-22/ 19-18/ 15-14/ 11-10	Счетчик событий сигнала DMA_REQB/DMA_REQA/PWM_SOCB/ PWM_SOCA. Счетчик автоматически сбрасывается, когда сформировано прерывание и перестает считать, когда достигает значения DRQBPRD/DRQAPRD/SOCBPRD/SOCAPRD соответсвенно.	
DRQBPRD/ DRQAPRD	21-20/ 17-16	Поле задания количества событий, выбранных полем DRQBSEL/ DRQASEL регистра ETSEL, по которым будет сформирован сигнал запуска DMA_REQB/DMA_REQA. Для разрешения генерации сигнала нужно установить бит DRQBEN/DRQAEN регистра ETSEL. Сигнал будет сформирован, даже если флаг DRQB/DRQA (регистр ETFLG) предыдущего сигнала не был сброшен. Как только сигнал DMA_REQB/DMA_REQA отправлен, счетчик DRQBCNT/DRQACNT автоматически сбрасывается	
		00	Выдача сигнала по каждому событию
		01	Выдача сигнала каждые два события
		10	Выдача сигнала каждые три события
		11	Выдача сигнала каждые четыре события
SOCBPRD/ SOCAPRD	13-12/ 9-8	Поле задания количества событий, заданных полем SOCBSEL/ SOCASEL регистра ETSEL, по которым будет сформирован сигнал запуска АЦП PWM_SOCB/PWM_SOCA. Для разрешения генерации сигнала нужно установить бит SOCBEN/SOCAEN регистра ETSEL. Сигнал будет сформирован, даже если флаг SOCB/SOCA (регистр ETFLG) предыдущего сигнала не был сброшен. Как только сигнал PWM_SOCB/PWM_SOCA отправлен, счетчик SOCBCNT/SOCACNT автоматически сбрасывается	
		00	Выдача сигнала по каждому событию
		01	Выдача сигнала каждые два события
		10	Выдача сигнала каждые три события
		11	Выдача сигнала каждые четыре события
INTCNT	3-2	Значение счетчика событий прерываний. Счетчик автоматически сбрасывается, когда сформировано прерывание и перестает считать, когда достигает значения INTPRD	

Поле	Биты	Описание	
INTPRD	1-0	Поле задания количества событий, выбранных полем INTSEL регистра ETSEL, по которым будет сформировано внешнее прерывание PWM_INT. Для разрешения генерации прерывания нужно установить бит INTEN в регистре ETSEL. Если флаг прерывания INT (регистр ETFLG) установлен от предыдущего прерывания, то текущее прерывание не будет активировано до сброса этого флага (сбрасывается записью единицы в бит INT регистра ETCLR). Такой механизм позволяет обрабатывать одно прерывание, в то время как другое ждет своей очереди	
		00	Прерывания по каждому событию (INTCNT = 00b)
		01	Прерывания каждые два события (INTCNT = 01b)
		10	Прерывания каждые три события (INTCNT = 10b)
		11	Прерывания каждые четыре события (INTCNT = 11b)
–	31-24, 7-4	Зарезервировано	

ETFLG – регистр флагов триггера событий

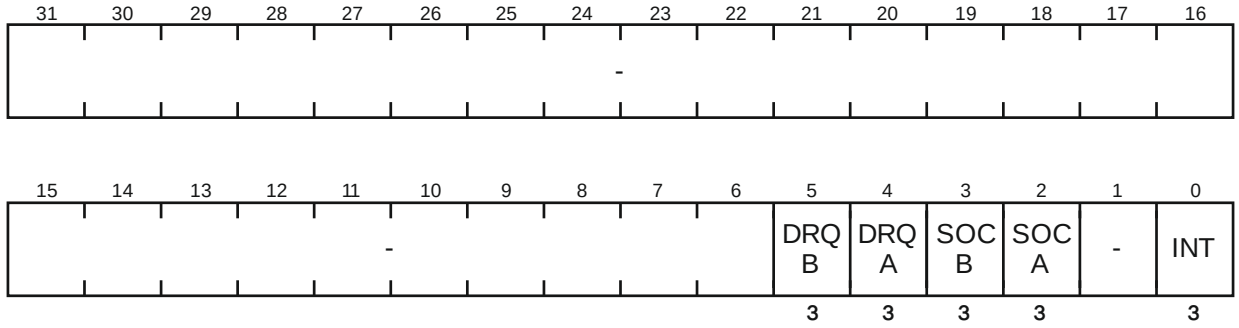
Смещение: + 5Ch
Сброс: 0h



Поле	Битыы	Описание	
DRQB/ DRQA	5/ 4	Флаг запроса DMA_REQB/DMA_REQA	
		0	Не установлено или сброшено
		1	Установлено
SOCB/ SOCA/ INT	3/ 2/ 0	Флаг внешнего сигнала АЦП PWM_SOCB/PWM_SOCA/PWM_INT	
		0	Не установлено или сброшено
		1	Установлено
–	31-6, 1	Зарезервировано	

ETCLR – регистр сброса флагов триггера событий

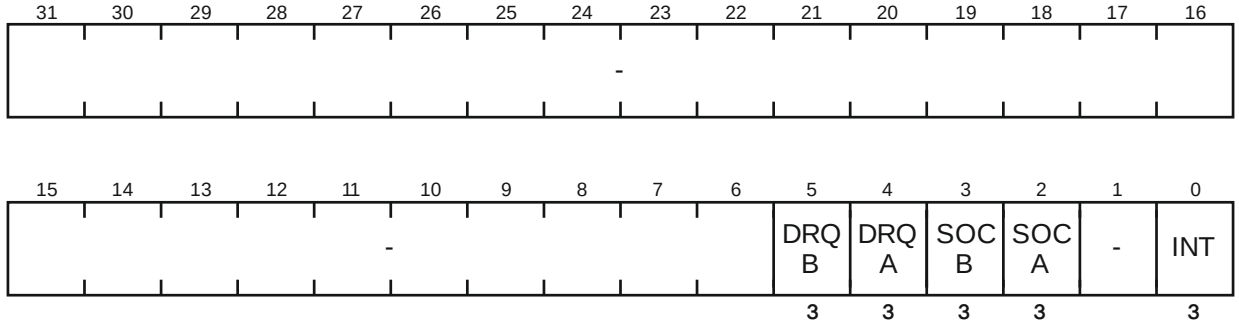
Смещение: + 60h
Сброс: 0h



Поле	Бит	Описание	
DRQB/ DRQA	5/ 4	Бит сброса флага DRQB/DRQA в регистре ETFLG	
		0	Нет действий
		1	Запись единицы сбрасывает флаг
SOCB/ SOCA/ INT	3/ 2/ 0	Бит сброса флага SOCB/SOCA/INT в регистре ETFLG	
		0	Нет действий
		1	Запись единицы сбрасывает флаг
–	31-6, 1	Зарезервировано	

ETFRC – регистр программной эмуляции флагов триггера событий

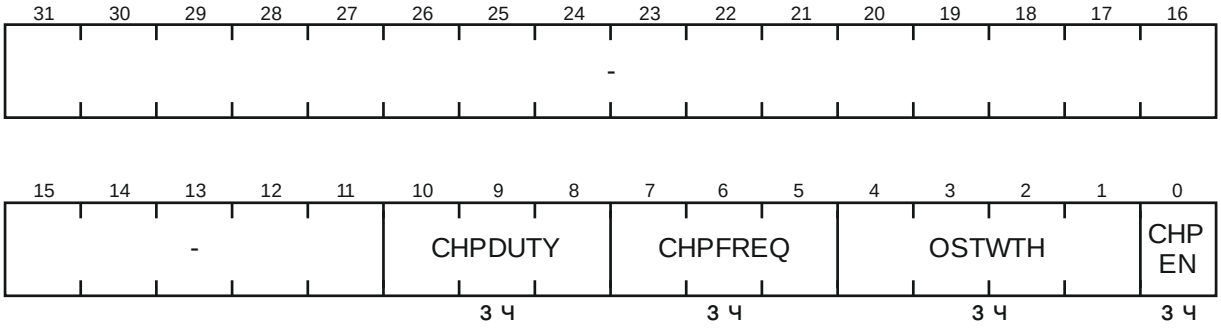
Смещение: + 64h
Сброс: 0h



Поле	Биты	Описание	
DRQB/ DRQA	5/ 4	Бит программной установки флага DRQB/DRQA в регистре ETFLG	
		0	Нет действий
		1	Запись единицы устанавливает флаг
SOCB/ SOCA/ INT	3/ 2/ 0	Бит программной установки флага SOCB/SOCA/INT в регистре ETFLG	
		0	Нет действий
		1	Запись единицы устанавливает флаг
–	31-6, 1	Зарезервировано	

PCCTL – регистр управления модулятором

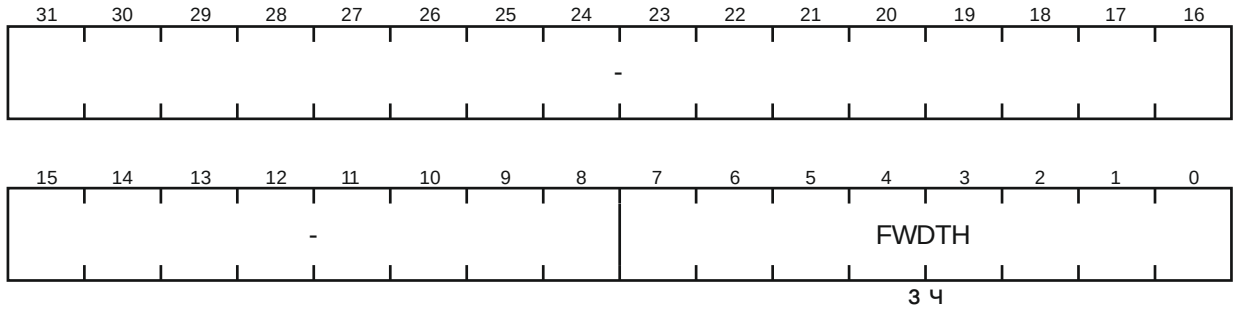
Смещение: + 68h
Сброс: 0h



Поле	Биты	Описание	
CHPDUTY	10-8	Поле задания коэффициента заполнения второго и последующих импульсов	
		000	1/8 (12,5 %)
		001	2/8 (25,0 %)
		…	…
		110	7/8 (87,5 %)
		111	Зарезервировано
CHPFREQ	7-5	Поле выбора делителя частоты синхронизации для задания частоты второго и последующих импульсов	
		000	1
		001	1/2
		…	…
		110	1/7
		111	1/8
OSTWTH	4-1	Поле задания ширины первого импульса	
		0h	$1 \times T_{PCLK} \times 8$
		1h	$2 \times T_{PCLK} \times 8$
		…	…
		Eh	$15 \times T_{PCLK} \times 8$
		Fh	$16 \times T_{PCLK} \times 8$
CHPEN	0	Бит разрешения работы модулятора	
		0	Запрещено
		1	Разрешено
–	31-11	Зарезервировано	

FWDTH – регистр ширины фильтрации

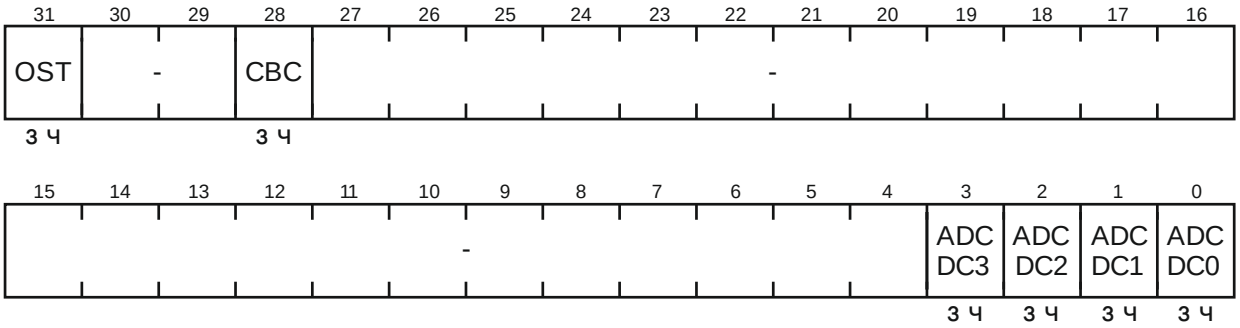
Смещение: + 70h
Сброс: 0h



Поле	Биты	Описание	
FWDTH	7-0	Поле задания ширины фильтрации коротких импульсов в тактах PCLK. Если f_{PCLK} = 100 МГц, то фильтруются импульсы до 2,55 мкс с шагом 0,01 мкс.	
		0h	Фильтр выключен
		1h	1 такт PCLK
		…	…
		FFh	255 тактов PCLK
–	31-8	Зарезервировано	

HDSEL – регистр источника сигнала события удержания

Смещение: + 88h
Сброс: 0h

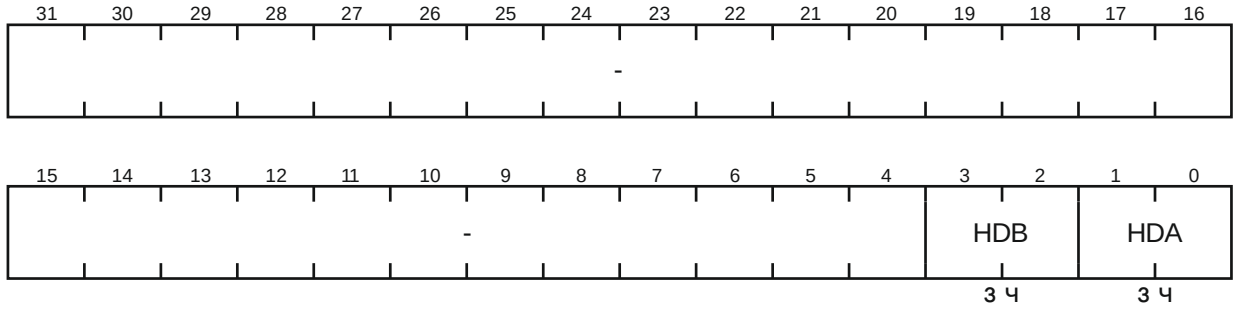


		Описание	
OST	31	Бит разрешения события по источнику ADCDC в однократном режиме обработки аварии	
		0	Запрещено
		1	Разрешено
CBC	28	Бит разрешения события по источнику ADCDC в циклическом режиме обработки аварии	
		0	Запрещено
		1	Разрешено

Поле	Биты	Описание	
ADCDC3–ADCDC0	3-0	Бит выбора цифрового компаратора (0 – 3) блока АЦП, с выхода которого берется сигнал для формирования события удержания	
		0	Не выбрано
		1	Выбрано
–	30-29, 27-4	Зарезервировано	

HDCTL – регистр управления детектором событий удержания

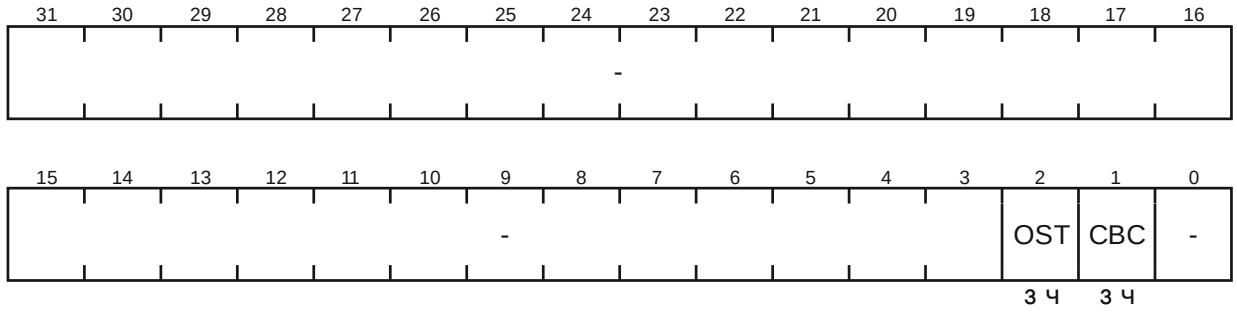
Смещение: + 8Ch
Сброс: 0h



Поле	Биты	Описание	
HDB/ HDA	3-2/ 1-0	Поле задания поведения сигнала PWMx_B/PWMx_A в случае сбоя (аварии). (Источник сбоя определяется регистром HDSEL)	
		00	Зарезервировано
		01	Переключается в состояние единицы
		10	Переключается в состояние нуля
		11	Остается без изменений
–	31-4	Зарезервировано	

HDEINT – регистр маски прерывания порогового выключателя

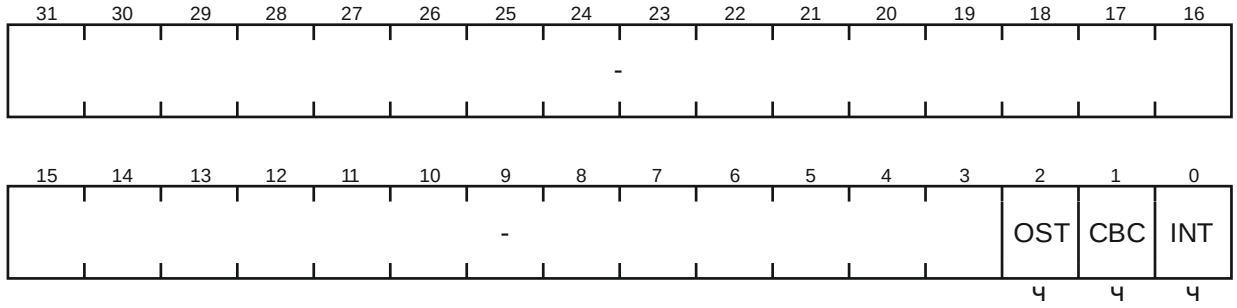
Смещение: + 90h
Сброс: 0h



Поле	Биты	Описание	
OST	2	Бит разрешения генерации прерывания в однократном режиме	
		0	Запрещено
		1	Разрешено
CBC	1	Бит разрешения генерации прерывания в циклическом режиме	
		0	Запрещено
		1	Разрешено
–	31-3, 0	Зарезервировано	

HDFLG – регистр флагов прерывания порогового выключателя

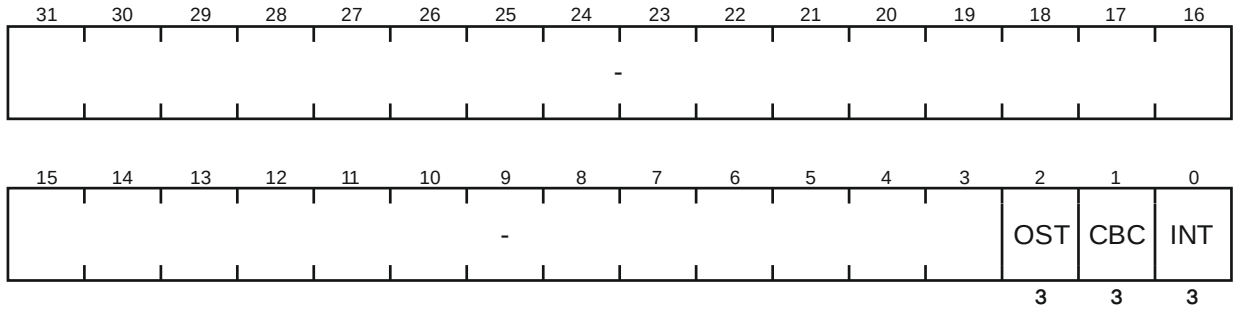
Смещение: + 94h
Сброс: 0h



Поле	Биты	Описание
OST, CBC	2, 1	Флаг прерывания в однократном режиме, флаг прерывания в циклическом режиме. При этом действие на выходе продолжается вплоть до обнуления счетчика таймера, если сигнал аварии не перестал быть активным к этому моменту. Если флаг сброшен, а источник сигнала аварии остался, флаг установится снова
INT	0	Флаг внешнего прерывания NVIC. Если флаг был сброшен, а один из флагов CBC или OST установлен, флаг INT установится снова
–	31-3	Зарезервировано

HDCLR – регистр сброса флагов порогового выключателя

Смещение: + 98h
Сброс: 0h

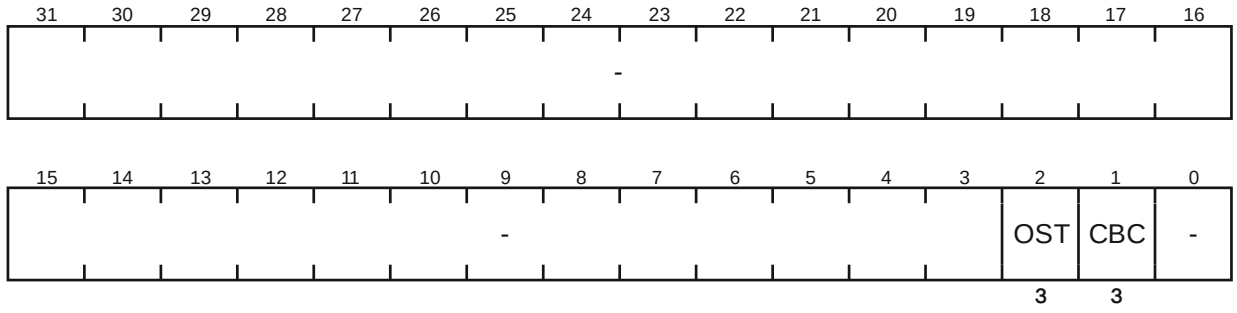


Поле	Биты	Описание
OST	2	Бит сброса флага прерывания в однократном режиме
CBC	1	Бит сброса флага прерывания в циклическом режиме
INT	0	Бит сброса флага внешнего прерывания NVIC
–	31-3	Зарезервировано

Примечание – Запись единицы в бит регистра сбрасывает соответствующий бит в регистре HDFLG.

HDFRC – регистр программной активации порогового выключателя

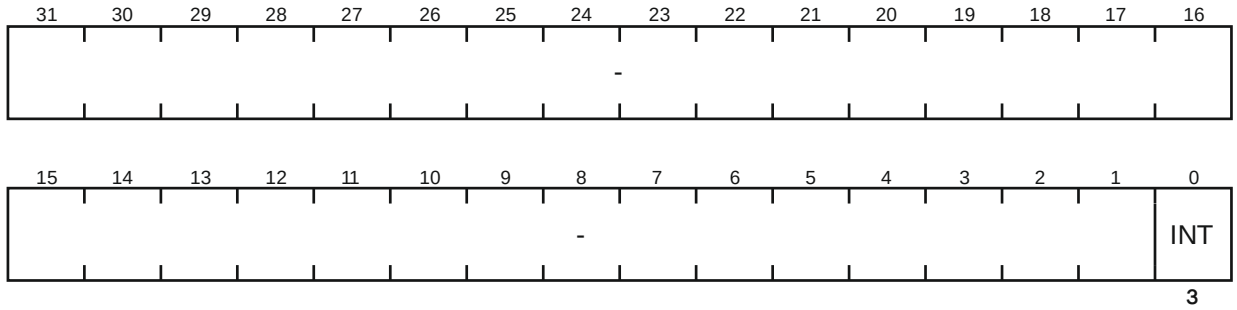
Смещение: + 9Ch
Сброс: 0h



Поле	Биты	Описание	
OST	2	Бит активации порогового выключателя в однократном режиме обработки аварии	
		0	Нет действий
		1	Запись единицы активирует выключатель и устанавливает флаг OST в регистре HDFLG
CBC	1	Бит активации порогового выключателя в циклическом режиме обработки аварии	
		0	Нет действий
		1	Запись единицы активирует выключатель и устанавливает флаг CBC в регистре HDFLG
–	31-3, 0	Зарезервировано	

HDINTCLR – регистр сброса прерывания порогового выключателя

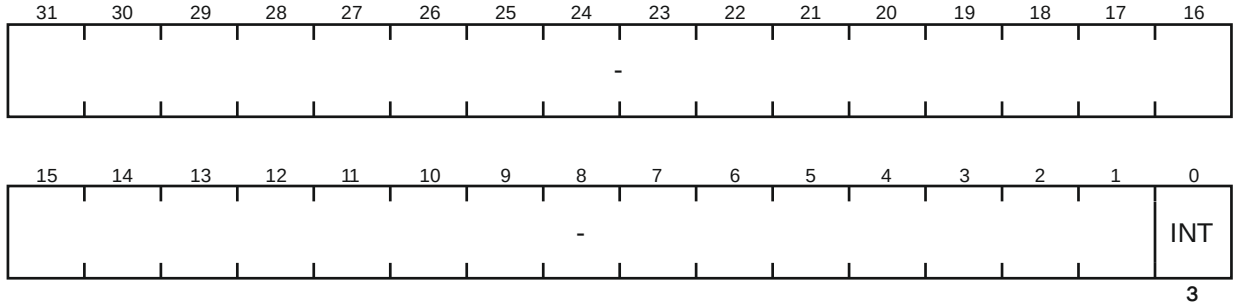
Смещение: + A0h
Сброс: 0h



Поле	Биты	Описание
INT	0	Бит сброса прерывания. Сброс запроса на прерывание должен производиться программой обработки прерывания, во избежание повторного запуска обработчика
–	31-1	Зарезервировано

TZINTCLR – регистр сброса прерывания детектора событий аварии

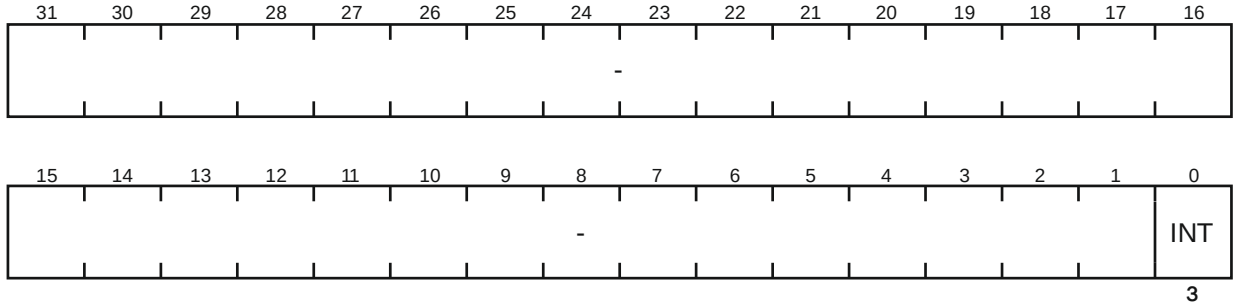
Смещение: + A4h
Сброс: 0h



Примечание – Назначение и функционирование аналогично регистру HDINTCLR.

INTCLR – регистр сброса прерывания таймера блока ШИМ

Смещение: + A8h
Сброс: 0h



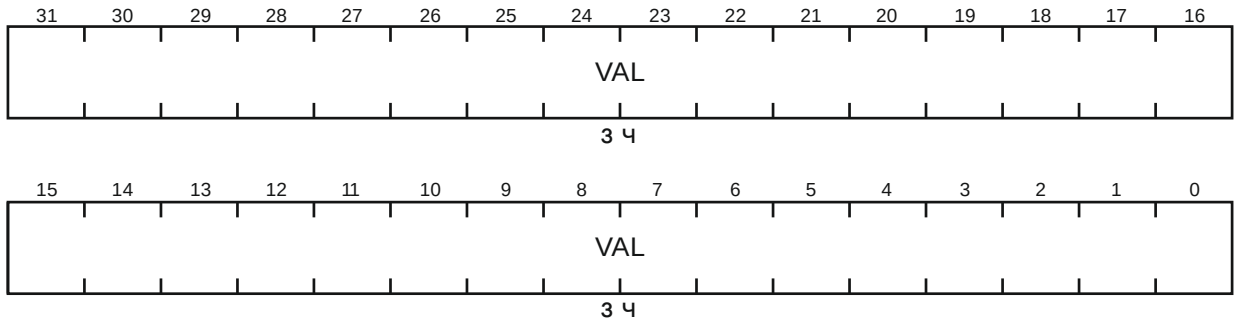
Примечание – Назначение и функционирование аналогично регистру HDINTCLR.

A.14 Регистры квадратурного декодера QEP

Базовый адрес: 4004_F000h

QPOSCNT – регистр счета счетчика позиции

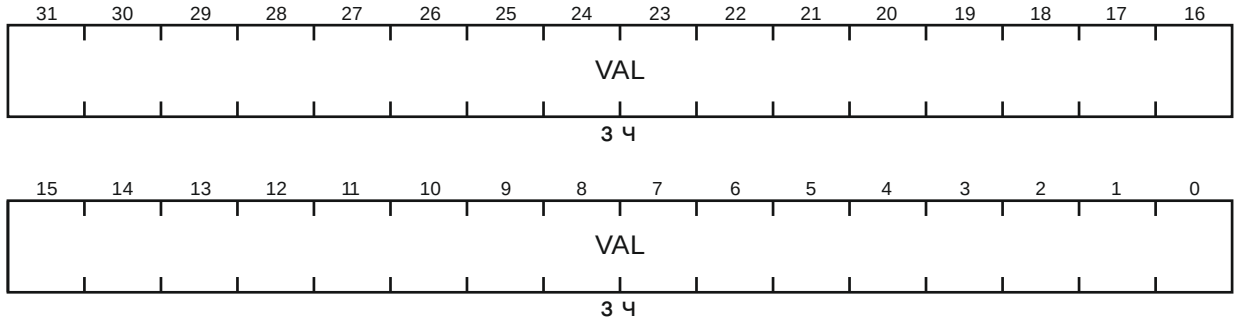
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение счета счетчика позиции

QPOSINIT – регистр инициализации счетчика позиции

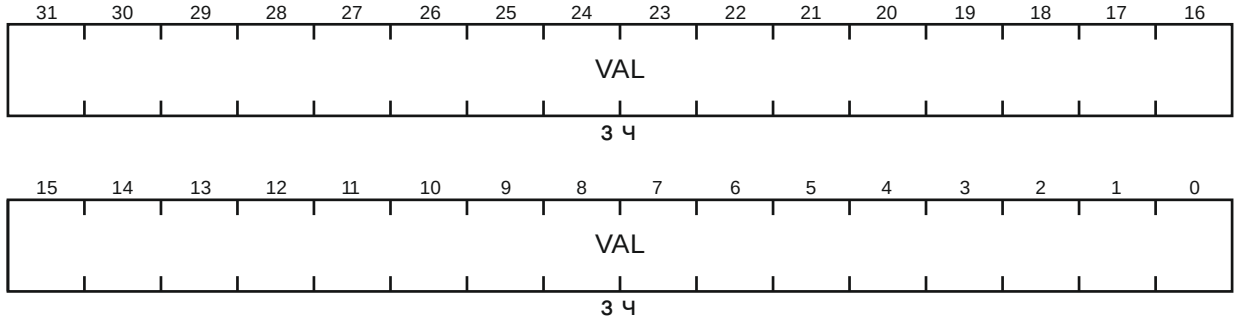
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение инициализации счетчика позиции

QPOSMAX – регистр максимального значения счетчика позиции

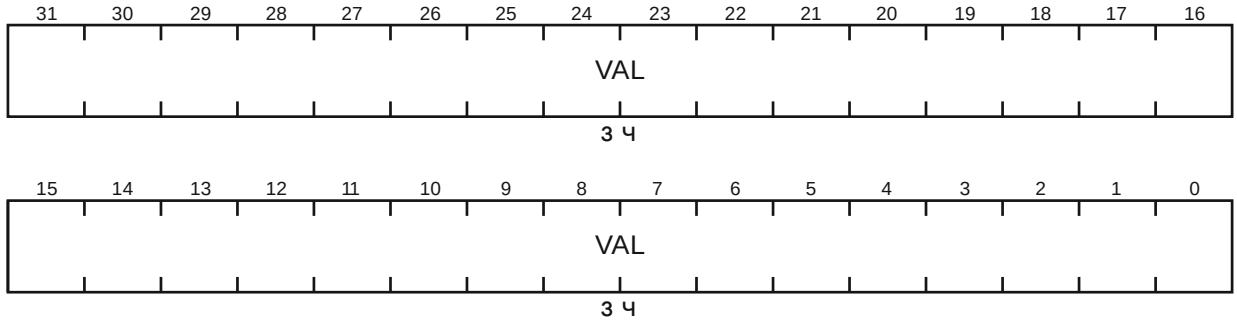
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение максимального значения счетчика позиции

QPOSCMP – регистр сравнения счетчика позиции

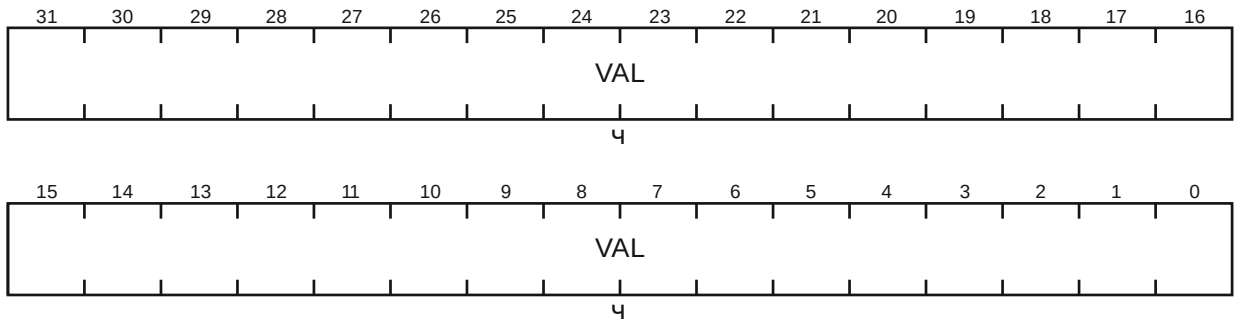
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение сравнения счетчика позиции

QPOSILAT – регистр хранения позиции по индексации

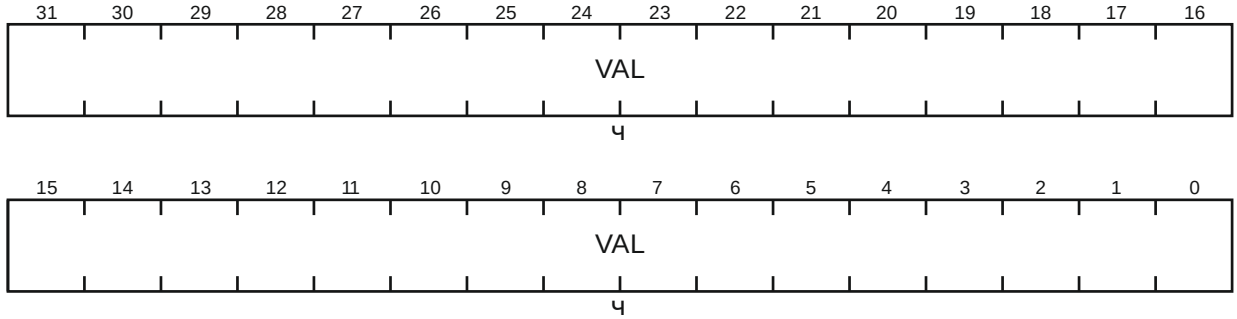
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение хранения позиции по индексации

QPOSSLAT – регистр хранения позиции по стробу

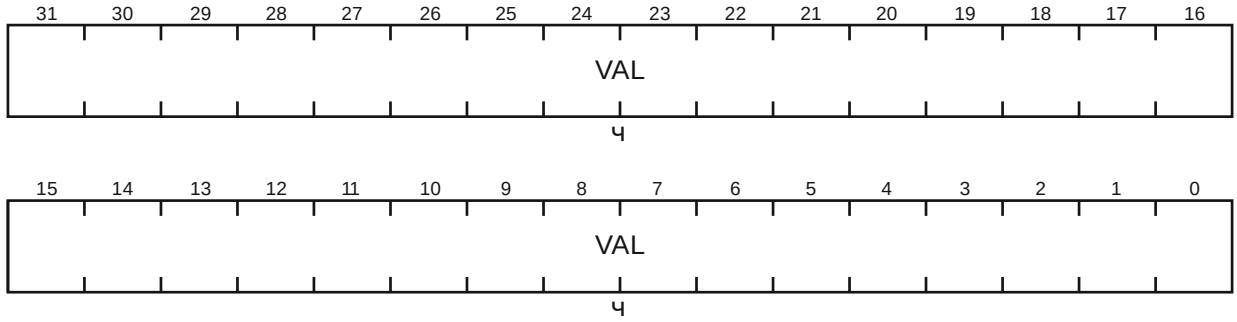
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение хранения позиции по стробу

QPOSLAT – регистр хранения позиции по таймеру временных отсчетов

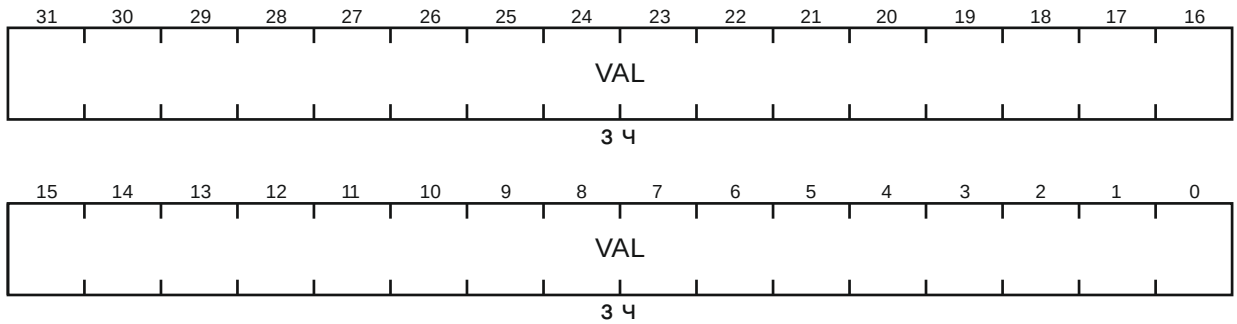
Смещение: + 18h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение хранения позиции по таймеру временных отсчетов

QUTMR – регистр таймера временных отсчетов

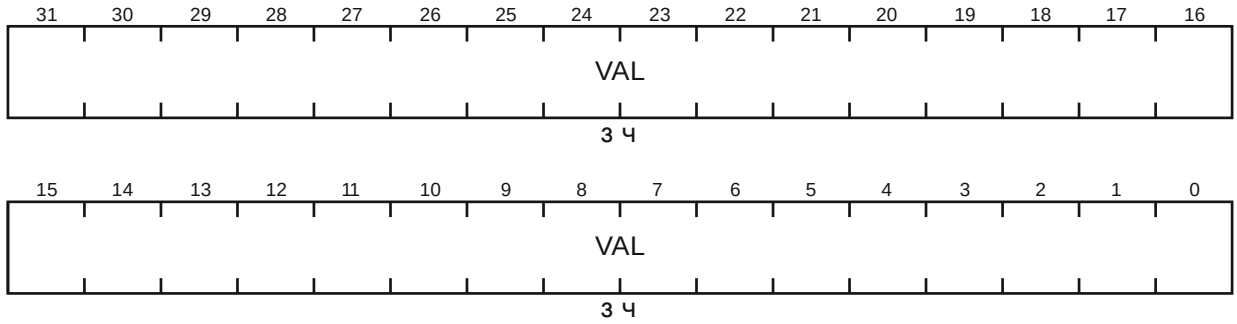
Смещение: + 1Ch
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение таймера временных отсчетов

QUPRD – регистр длительности счета таймера временных отсчетов

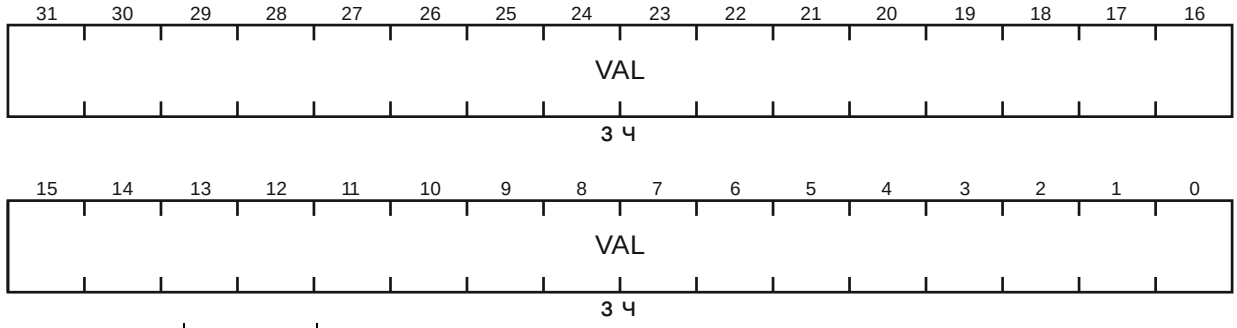
Смещение: + 20h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение длительности счета таймера временных отсчетов

QWDTMR – регистр счета сторожевого таймера

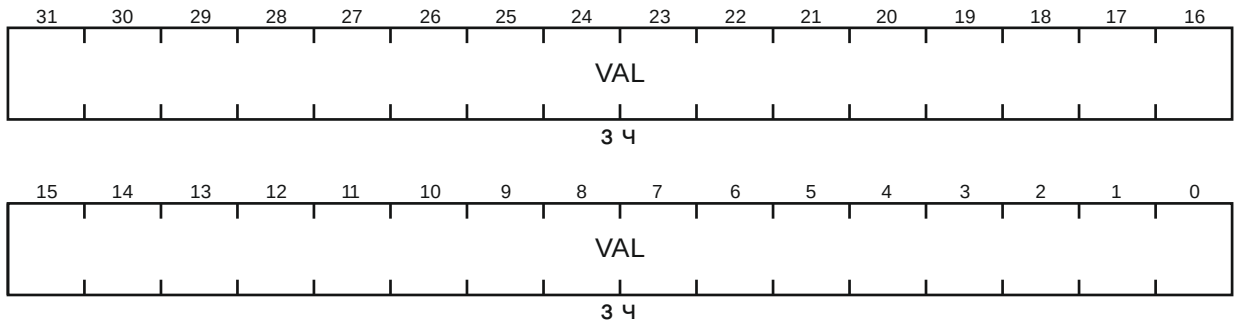
Смещение: + 24h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение счета сторожевого таймера

QWDPRD – регистр длительности счета сторожевого таймера

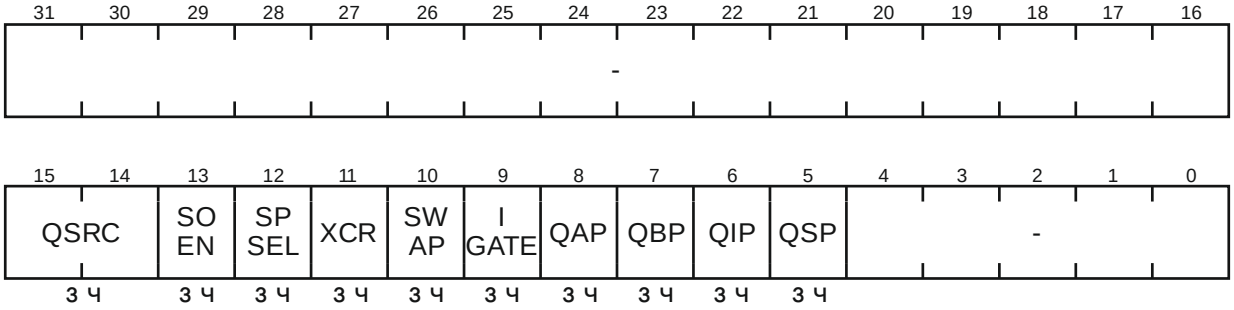
Смещение: + 28h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение длительности счета сторожевого таймера

QDECCTL – регистр управления входами

Смещение: + 2Ch
Сброс: 0h

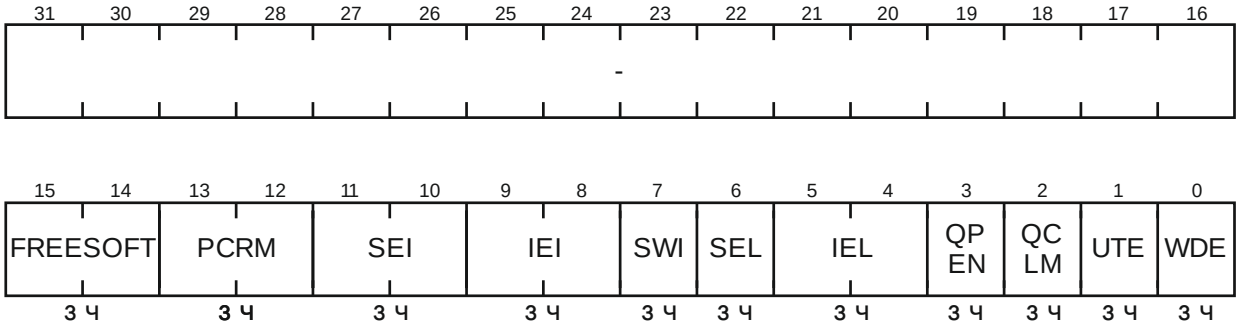


Поле	Биты	Описание	
QSRC	15-14	Режим работы	
		00	Квадратурный
		01	Счета/направления
		10	Счет вверх (QCLK = xCLK, QDIR = 1)
		11	Счет вниз (QCLK = xCLK, QDIR = 0)
SOEN	13	Бит разрешения выдачи выходного сигнала компаратора	
		0	Запрещено
		1	Разрешено
SPSEL	12	Бит выбора вывода для выдачи выходного сигнала компаратора	
		0	Стробирующий вывод
		1	Индексный вывод
XCR	11	Бит выбора фронта квадратурного входа	
		0	Передний фронт
		1	Передний и задний фронты
SWAP	10	Бит обмена входов QEP_A и QEP_B	
		0	Нет действий
		1	Входы QEP_A и QEP_B меняются местами
IGATE	9	Бит включения стробирования входного сигнала индексации	
QAP	8	Бит включения инвертирования входного сигнала с QEP_A	
QBP	7	Бит включения инвертирования входного сигнала с QEP_B	
QIP	6	Бит включения инвертирования входного сигнала с QEP_I	
QSP	5	Бит включения инвертирования входного сигнала с QEP_S	
–	31-16, 4-0	Зарезервировано	

Примечание – Для битов с 9 по 5 справедливо: 0 – выключено, 1 – включено.

QEPCTL – регистр управления квадратурного декодера

Смещение: + 30h
Сброс: 0h

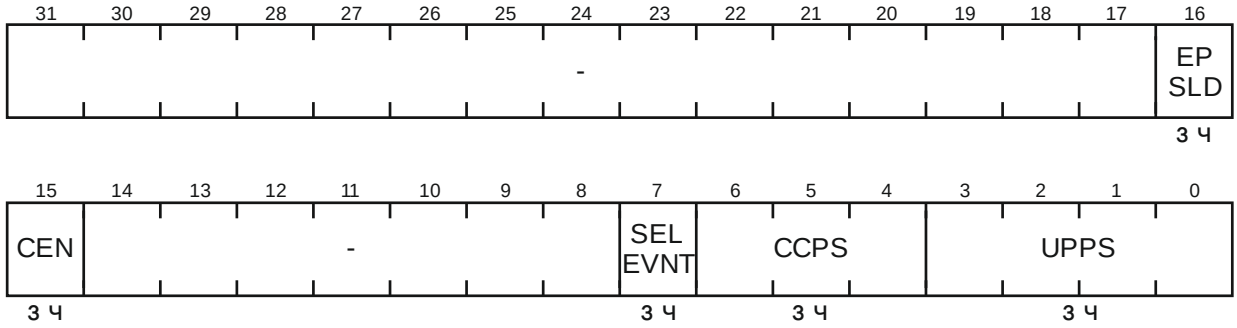


Поле	Биты	Описание	
FREESOFT	15-14	Поле управления счетчиками QPOSCNT, QWDTMR, QUTMR, QCTMR в режиме отладки	
		00	Принудительная блокировка счета
		01	Счет до переполнения
		10, 11	Разблокирование счета
PCRM	13-12	Поле задания события для сброса счетчика позиции	
		00	Событие индексации
		01	Достижение максимальной позиции
		10	Первое событие индексации
		11	Окончание временного отсчета
SEI	11-10	Поле задания события стробирования для инициализации счетчика позиции (QPOSCNT = QPOSINIT)	
		00, 01	Работа без инициализации
		10	Передний фронт сигнала QEPS
		11	Передний фронт QEPS при вращении по часовой стрелке или задний фронт QEPS при вращении против часовой стрелки
IEI	9-8	Поле задания события индексации для инициализации счетчика позиции (QPOSCNT = QPOSINIT)	
		00, 01	Работа без инициализации
		10	По переднему фронту сигнала QEPI
		11	По заднему фронту сигнала QEPI
SWI	7	Бит программной инициализации счетчика позиции. Не сбрасывается аппаратно	
		0	Нет действий
		1	Запись единицы загружает счетчик позиции QPOSCNT значением QPOSINIT
SEL	6	Бит задания события стробирования для сохранения значения счетчика позиции (QPOSSLAT = POSCNT)	
		0	По переднему фронту QEPS
		1	По переднему фронту QEPS при вращении по часовой стрелке или по заднему фронту QEPS при вращении против часовой стрелки

Поле	Биты	Описание	
IEL	5-4	Поле задания события индексации для сохранения значения счетчика позиции (QPOSILAT = POSCNT)	
		00	Без сохранения
		01	По переднему фронту сигнала индексации
		10	По заднему фронту сигнала индексации
		11	По маркеру индексации
QPEN	3	Бит разрешения работы счетчика позиции	
		0	Запись нуля останавливает счетчик и сбрасывает его
		1	Работа разрешена
QCLM	2	Бит задания события сохранения значения регистров модуля захвата	
		0	По чтению QPOSCNT регистры QCTMR и QCPRD сохраняются в регистры QCTMRLAT и QCPRDLAT соответственно.
		1	По окончании временного отсчета регистры QPOSCNT, QCTMR и QCPRD сохраняются в регистры QPOSLAT, QCTMRLAT и QCPRDLAT соответственно
UTE	1	Бит разрешения работы таймера временных отсчетов	
		0	Запрещено
		1	Разрешено
WDE	0	Бит разрешения работы сторожевого таймера	
		0	Запрещено
		1	Разрешено
–	31-16	Зарезервировано	

QCAPCTL – регистр управления захватом

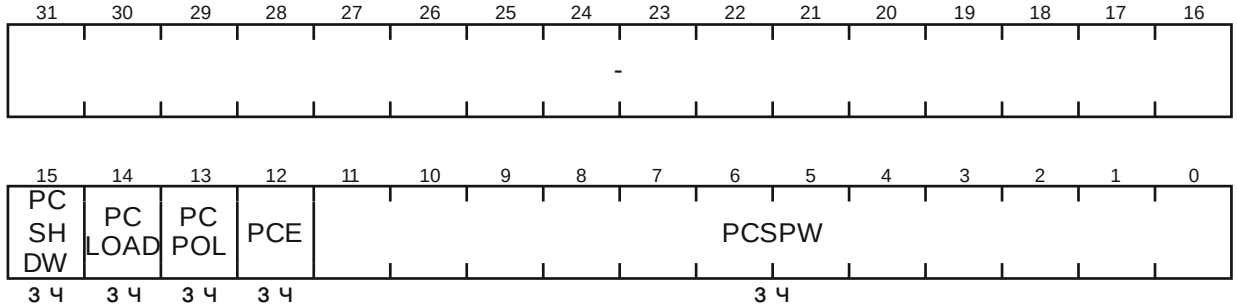
Смещение: + 34h
Сброс: 0h



Поле	Биты	Описание	
EPSLD	16	Бит включения улучшенного режима теневой загрузки UPPS/CCPS	
		0	Запрещено
		1	Разрешено
CEN	15	Бит разрешения работы модуля захвата времени	
		0	Запрещено
		1	Разрешено
SELEVNT	7	Бит сброса таймера	
		0	По деленному квадратурному событию
		1	По получении сигнала PCSOUT от компаратора
CCPS	6-4	Поле задания делителя системного такта	
		000	Нет деления
		001	1/2
		010	1/4
		011	1/8
		100	1/16
		101	1/32
		110	1/64
		111	1/128
UPPS	3-0	Поле задания делителя квадратурного сигнала	
		0h	Нет деления
		1h	1/2
		2h	1/4
		3h	1/8
		4h	1/16
		5h	1/32
		6h	1/64
		7h	1/128
		8h	1/256
		9h	1/512
		Ah	1/1024
		Bh	1/2048
		Ch-Fh	Зарезервировано
–	31-17, 14-8	Зарезервировано	

QPOSCTL – регистр управления счетчиком позиции

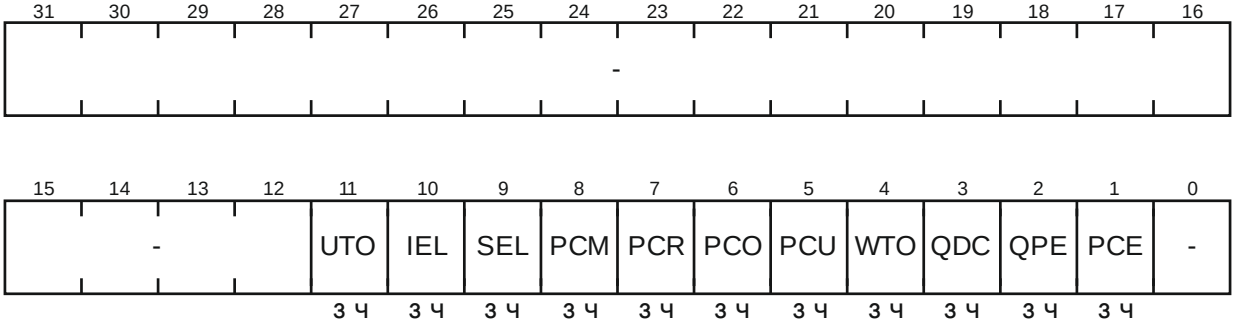
Смещение: + 38h
Сброс: 0h



Поле	Биты	Описание	
PCSHDW	15	Бит разрешения режима отложенной загрузки	
		0	Запрещено
		1	Разрешено
PCLOAD	14	Бит выбора события загрузки в режиме отложенной записи	
		0	Загрузка отложенного значения в активный регистр по событию QPOSCNT = 0
		1	Загрузка по QPOSCNT = QPOSCMP
PCPOL	13	Бит выбора полярности выхода синхронизации	
		0	Активная единица
		1	Активный ноль
PCE	12	Бит разрешения работы компаратора	
		0	Запрещено
		1	Разрешено
PCSPW	11-0	Поле задания ширины импульса выхода синхронизации	
		000h	Отсутствие импульса
		001h	2 × T
		…	…
		007h	8 × T
		…	…
		FFFh	4096 × T
		T – период тактового сигнала PCLK	
–	31-16	Зарезервировано	

QEINT – регистр масок прерываний

Смещение: + 3Ch
Сброс: 0h

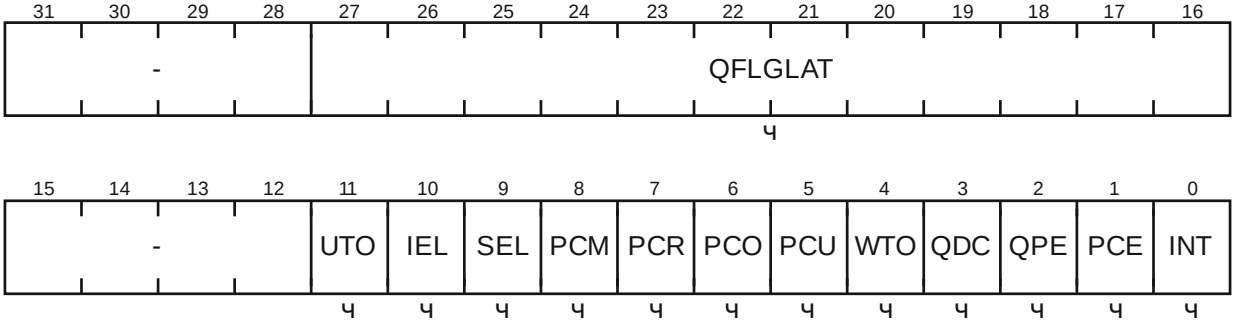


Поле	Биты	Описание
UTO	11	Бит разрешения прерывания по срабатыванию таймера временных отсчетов
IEL	10	Бит разрешения прерывания по событию индексации
SEL	9	Бит разрешения прерывания по событию стробирования
PCM	8	Бит разрешения прерывания по срабатыванию компаратора
PCR	7	Бит разрешения прерывания по готовности компаратора к загрузке значения сравнения из отложенного регистра
PCO	6	Бит разрешения прерывания по переполнению счетчиком позиции QPOSMAX при счете вверх
PCU	5	Бит разрешения прерывания по переходу счетчиком позиции через минимальное значение при счете вниз
WTO	4	Бит разрешения прерывания при срабатывании сторожевого таймера
QDC	3	Бит разрешения прерывания при смене направления вращения
QPE	2	Бит разрешения прерывания по ошибке фазы на квадратурном входе
PCE	1	Бит разрешения прерывания счетчика позиции
–	31-12, 0	Зарезервировано

Примечание – Установленный бит разрешает генерирование соответствующего прерывания, сброшенный – запрещает.

QFLG – регистр флагов прерываний

Смещение: + 40h
Сброс: 0h

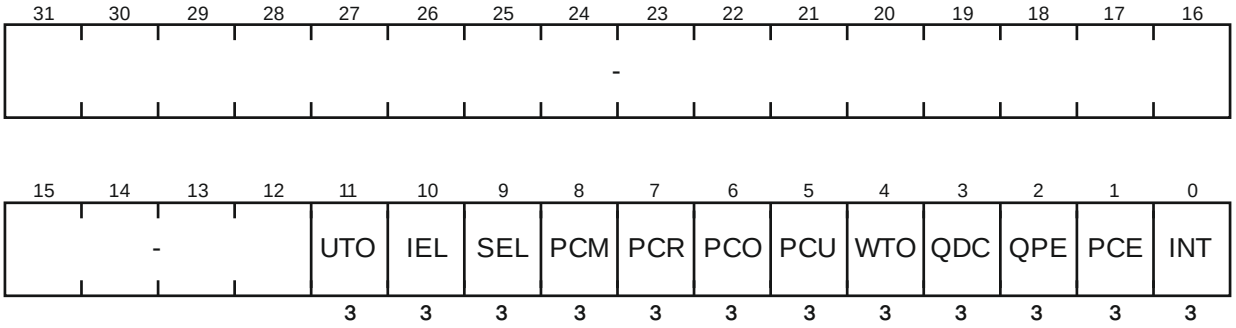


Поле	Биты	Описание
QFLGLAT	27-16	Защелкивает биты 11-0 регистра QFLG по событию чтения регистра QPOSCNT. Поле доступно только для чтения
UTO	11	Флаг прерывания по срабатыванию таймера временных отсчетов
IEL	10	Флаг прерывания по событию индексации
SEL	9	Флаг прерывания по событию стробирования
PCM	8	Флаг прерывания по срабатыванию компаратора
PCR	7	Флаг прерывания по готовности компаратора к загрузке значения сравнения из отложенного регистра
PCO	6	Флаг прерывания по переполнению счетчиком позиции QPOSMAX при счете вверх
PCU	5	Флаг прерывания по переходу счетчиком позиции через минимальное значение при счете вниз
WTO	4	Флаг прерывания при срабатывании сторожевого таймера
QDC	3	Флаг прерывания при смене направления вращения
QPE	2	Флаг прерывания по ошибке фазы на квадратурном входе
PCE	1	Флаг прерывания ошибки счетчика позиции
INT	0	Флаг выходного прерывания блока квадратурного декодера
–	31-28, 15-12	Зарезервировано

Примечание – Установленный бит является индикатором запроса соответствующего прерывания. Сброс флагов прерываний осуществляется посредством регистра QCLR.

QCLR – регистр сброса флагов прерываний

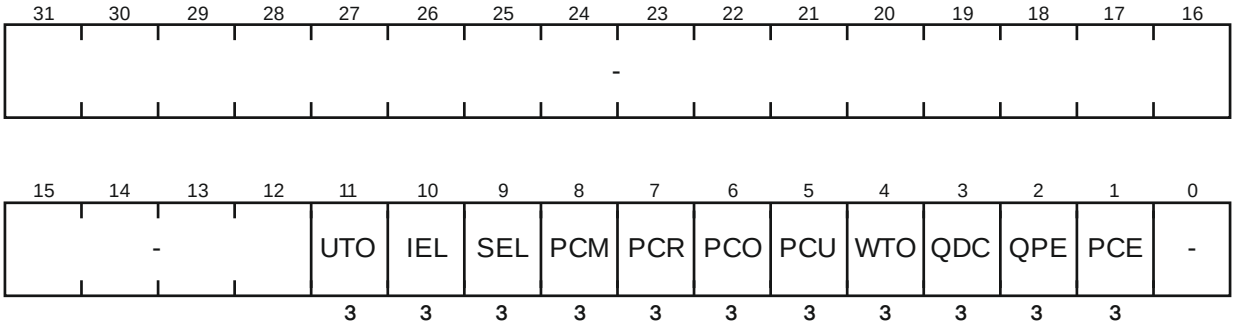
Смещение: + 44h
Сброс: 0h



Поле	Биты	Описание
UTO	11	Запись единицы сбрасывает флаг прерывания по срабатыванию таймера временных отсчетов
IEL	10	Запись единицы сбрасывает флаг прерывания по событию индексации
SEL	9	Запись единицы сбрасывает флаг прерывания по событию стробирования
PCM	8	Запись единицы сбрасывает флаг прерывания по срабатыванию компаратора
PCR	7	Запись единицы сбрасывает флаг прерывания по готовности компаратора к загрузке значения сравнения из отложенного регистра
PCO	6	Запись единицы сбрасывает флаг прерывания по переполнению счетчиком позиции QPOSMAX при счете вверх
PCU	5	Запись единицы сбрасывает флаг прерывания по переходу счетчиком позиции через минимальное значение при счете вниз
WTO	4	Запись единицы сбрасывает флаг прерывания при срабатывании сторожевого таймера
QDC	3	Запись единицы сбрасывает флаг прерывания при смене направления вращения
QPE	2	Запись единицы сбрасывает флаг прерывания по ошибке фазы на квадратурном входе
PCE	1	Запись единицы сбрасывает флаг прерывания ошибки счетчика позиции
INT	0	Запись единицы сбрасывает флаг выходного прерывания блока квадратурного декодера
–	31-12	Зарезервировано

QFRC – регистр эмуляции прерываний

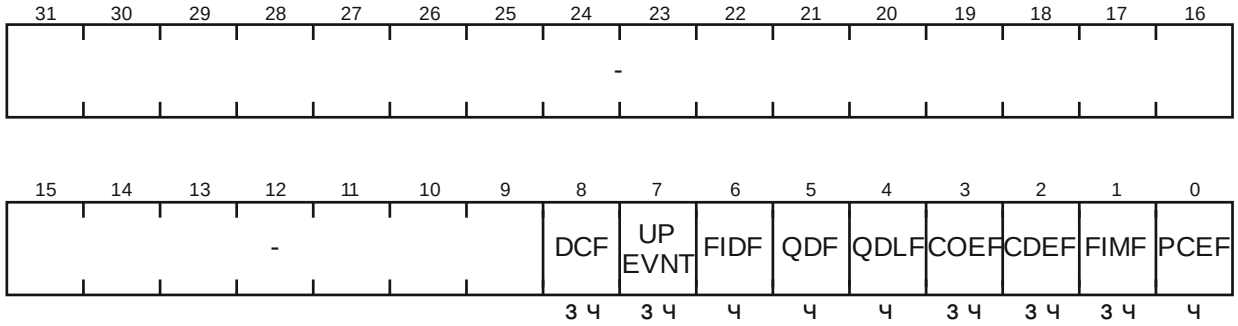
Смещение: + 48h
Сброс: 0h



Поле	Биты	Описание
UTO	11	Запись единицы сбрасывает флаг прерывания по срабатыванию таймера временных отсчетов
IEL	10	Запись единицы сбрасывает флаг прерывания по событию индексации
SEL	9	Запись единицы сбрасывает флаг прерывания по событию стробирования
PCM	8	Запись единицы сбрасывает флаг прерывания по срабатыванию компаратора
PCR	7	Запись единицы сбрасывает флаг прерывания по готовности компаратора к загрузке значения сравнения из отложенного регистра
PCO	6	Запись единицы сбрасывает флаг прерывания по переполнению счетчиком позиции QPOSMAX при счете вверх
PCU	5	Запись единицы сбрасывает флаг прерывания по переходу счетчиком позиции через минимальное значение при счете вниз
WTO	4	Запись единицы сбрасывает флаг прерывания при срабатывании сторожевого таймера
QDC	3	Запись единицы сбрасывает флаг прерывания при смене направления вращения
QPE	2	Запись единицы сбрасывает флаг прерывания по ошибке фазы на квадратурном входе
PCE	1	Запись единицы сбрасывает флаг прерывания ошибки счетчика позиции
–	31-12, 0	Зарезервировано

QEPSTS – регистр статуса

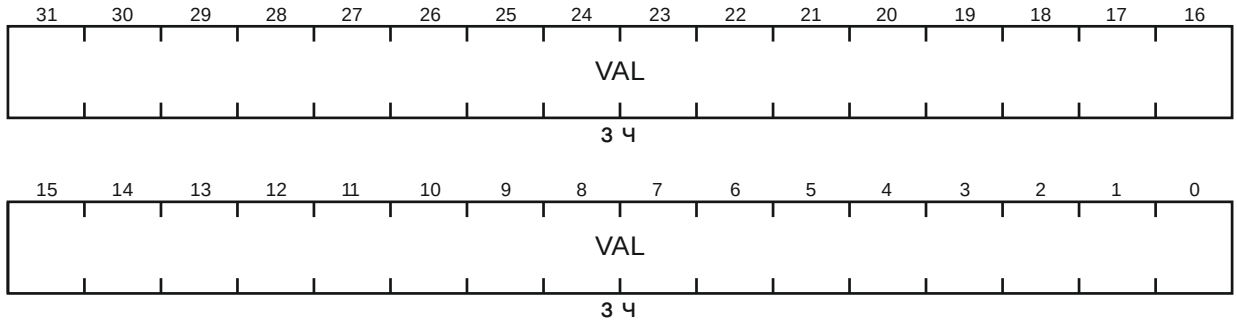
Смещение: + 4Ch
Сброс: 0h



Поле	Биты	Описание	
DCF	8	Флаг изменения направления вращения вала ротора. Сброс записью 1	
		0	Направление не изменялось
		1	Произошло изменение направления вращения
UPEVNT	7	Флаг сброса QCTMR и обновления QCPRD. Сброс записью 1	
		0	Нет событий
		1	Зафиксировано событие сброса и обновления
FIDF	6	Индикатор направления вращения по событию первого импульса индексации. Только чтение	
		0	Против часовой стрелки (счет вниз)
		1	По часовой стрелке (счет вверх)
QDF	5	Флаг направления вращения. Обновляется по каждому событию на входах квадратур. Только чтение	
		0	Вращение вала ротора против часовой стрелки
		1	Вращение вала ротора по часовой стрелке
QDLF	4	Флаг направления вращения. Обновляется по каждому сигналу индексации. Только чтение	
		0	Вращение вала ротора против часовой стрелки
		1	Вращение вала ротора по часовой стрелке
COEF	3	Флаг ошибки переполнения счетчика QCTMR модуля захвата. Сброс записью 1	
		0	Ошибка отсутствует
		1	Произошло переполнение
CDEF	2	Флаг ошибки изменения направления вращения вала ротора между двумя событиями UPEVNT. Сброс записью 1	
		0	Ошибка отсутствует
		1	Произошло изменение направления вращения во время измерения
FIMF	1	Флаг приема первого импульса сигнала индексации. Сброс записью 1	
		0	Импульсов нет либо первый импульс уже был принят
		1	Принят первый импульс сигнала индексации
PCEF	0	Флаг ошибки счетчика позиции. Обновляется по каждому сигналу индексации. Только чтение	
		0	Во время последнего сигнала индексации ошибки не возникло
		1	Ошибка счетчика позиции
–	31-9	Зарезервировано	

QCTMR – регистр таймера блока захвата

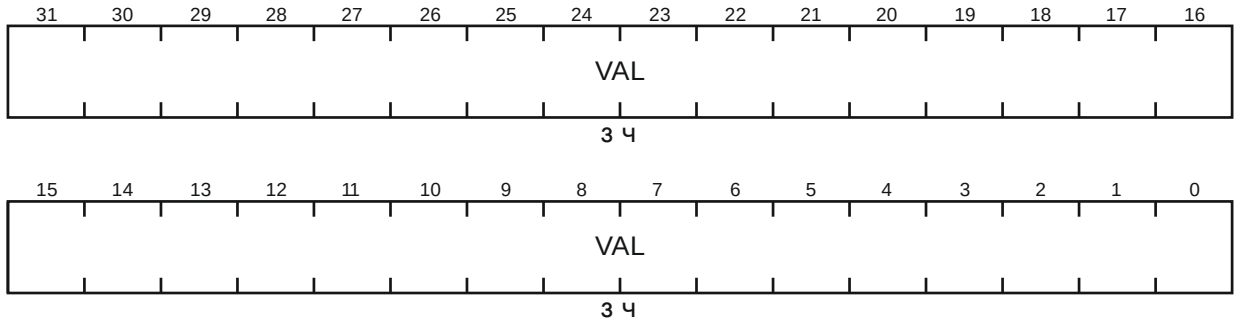
Смещение: + 50h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение таймера блока захвата

QCPRD – регистр длительности измерения блока захвата

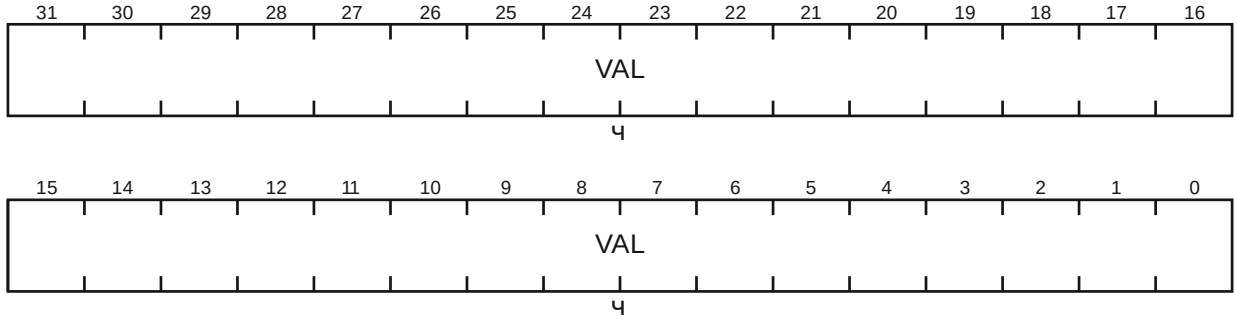
Смещение: + 54h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение длительности измерения блока захвата

QCTMRLAT – регистр хранения таймера блока захвата

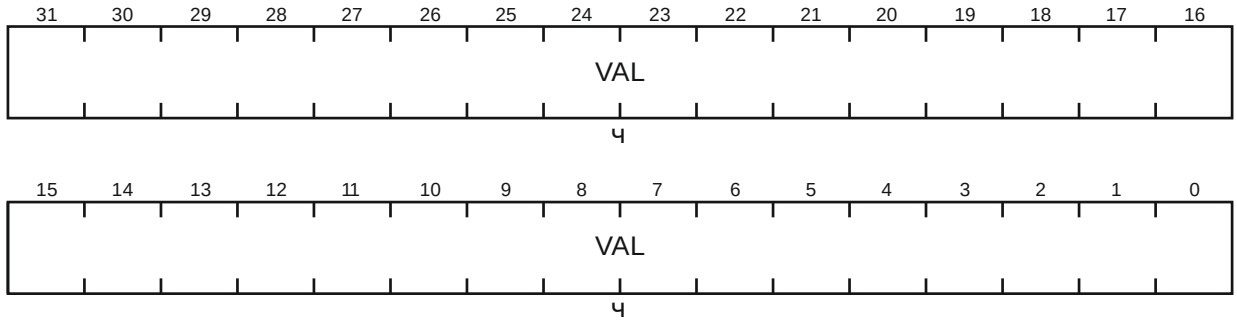
Смещение: + 58h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение хранения таймера блока захвата

QCPRDLAT – регистр хранения длительности измерения блока захвата

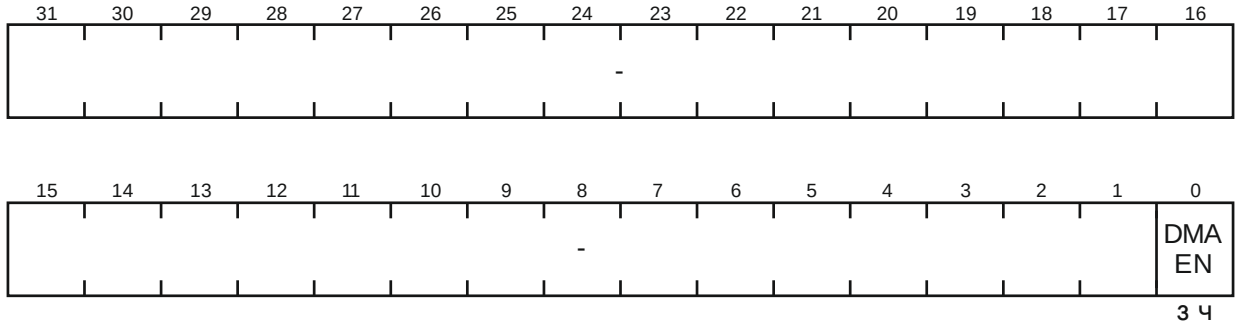
Смещение: + 5Ch
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение хранения длительности измерения блока захвата

DMAREQ – регистр управления запросом DMA

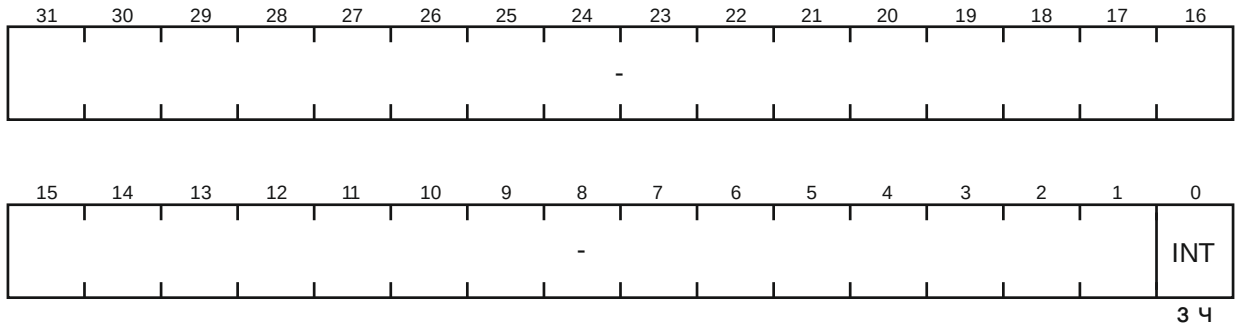
Смещение: + 60h
Сброс: 0h



Поле	Биты	Описание	
DMAEN	0	Разрешение генерации запроса DMA. Запрос генерируется каждый раз, когда флаг UPEVNT (регистра EPSTS) переходит из 0 в 1. Если UPEVNT будет оставаться несброшенным, то запросы генерироваться не будут. При этом, когда происходит чтение QCPRD при DMAEN = 1, флаг UPEVNT сбрасывается автоматически	
		0	Нет запроса
		1	Запрос разрешен
–	31-1	Зарезервировано	

INTCLR – регистр сброса прерывания

Смещение: + 70h
Сброс: 0h



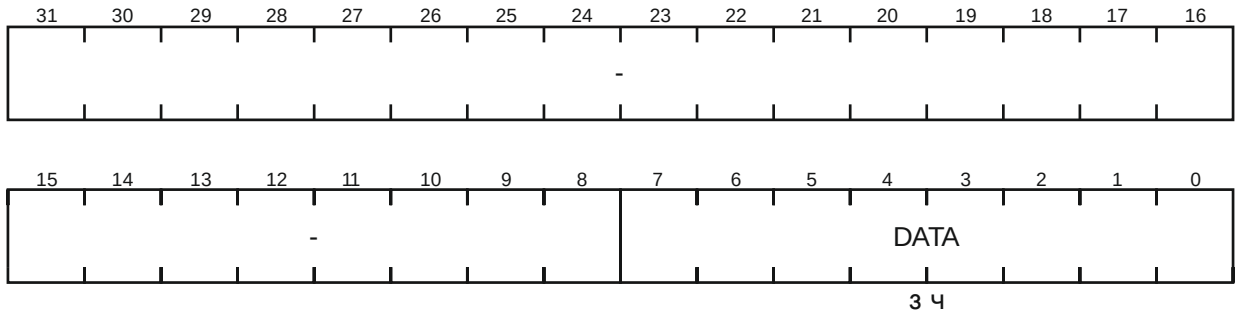
Поле	Биты	Описание
INT	0	Бит сброса прерывания. Чтение возвращает текущий статус активности прерывания. Запись единицы в бит сбрасывает запрос прерывания. Активный запрос способен вызывать повторные запуски обработчика прерывания
–	31-1	Зарезервировано

A.15 Регистры контроллера I2C

Базовый адрес: 4005_0000h

SDA – регистр данных

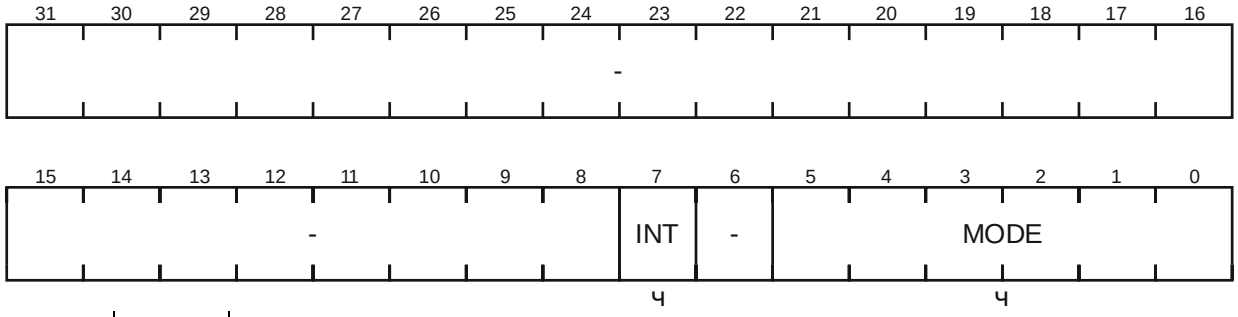
Смещение: + 00h
Сброс: 0xxh



Поле	Биты	Описание
DATA	7-0	Поле данных
–	31-8	Зарезервировано

ST – регистр состояния

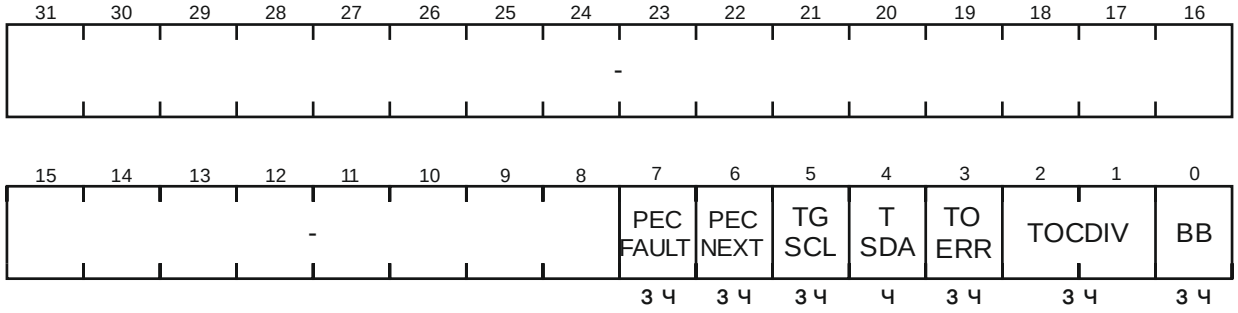
Смещение: + 04h
Сброс: 0h



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

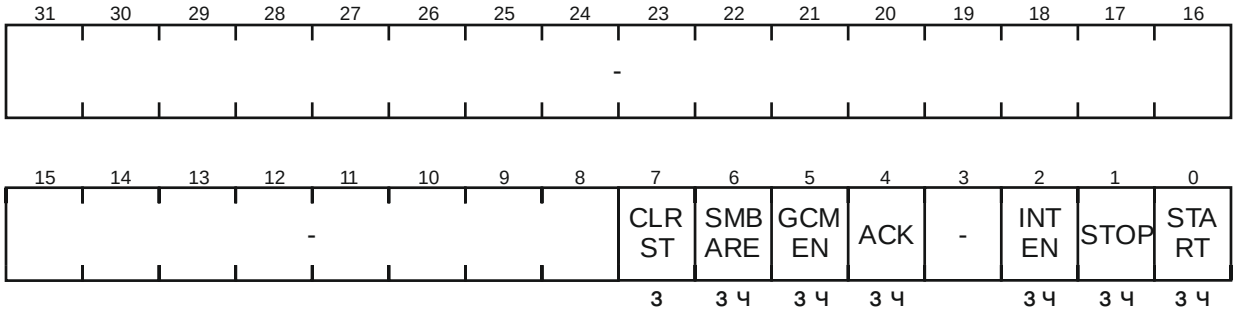


Поле	Биты	Описание
PECFAULT	7	Флаг ошибки. Устанавливается в случае, если после расчета контрольной суммы для пакета данных и сравнения ее с полученной суммой значение во внутреннем регистре ошибок не нулевое
PECNEXT	6	Бит управления отправкой байта контрольной суммы. Установка бита указывает на то, 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 I2C_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 I2C_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

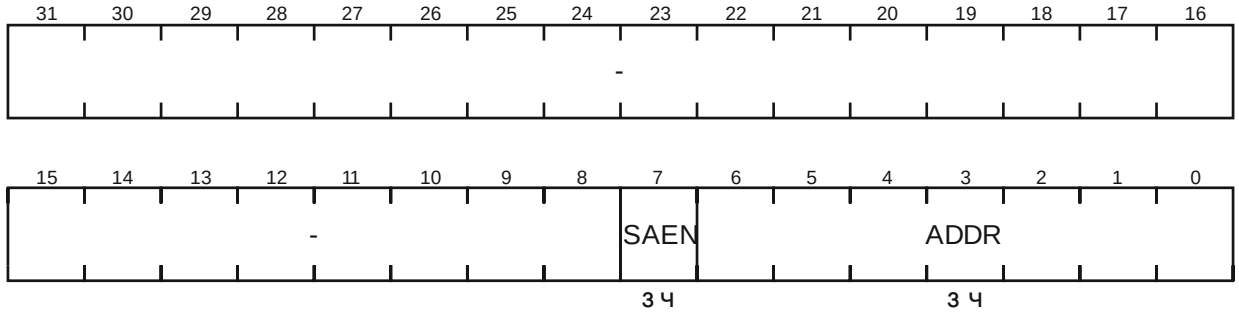


Поле	Биты	Описание	
CLRST	7	Бит сброса флага прерывания INT	
		Чтение	Возвращает ноль
		Запись нуля	Не выполняется
		Запись единицы	Сбрасывает флаг INT в регистре ST
SMBARE	6	Бит управления реакцией на получение адреса отклика	
		0	Полученный адрес не проверяется на совпадение с адресом отклика
		1	Адрес, полученный сразу после старта, проверяется на совпадение с адресом отклика (0001_100b)
		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 (0000_000b)
		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

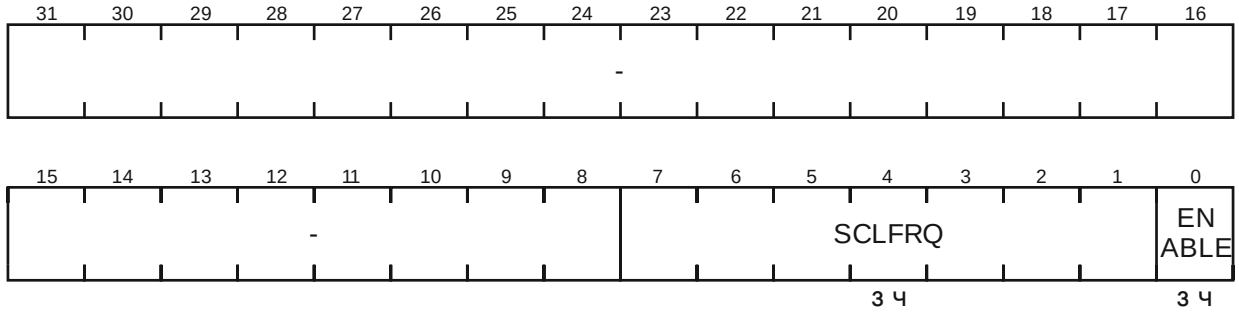
Смещение: + 10h
Сброс: 0h



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

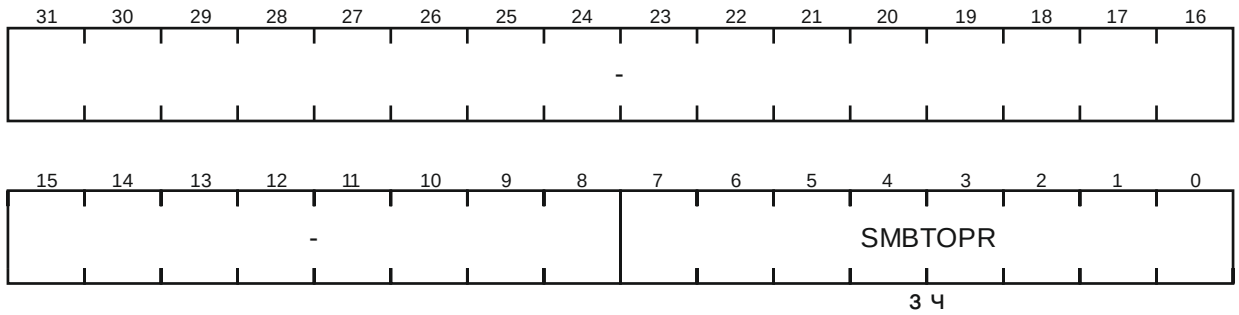
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание
SCLFRQ	7-1	Младшие разряды поля выбора частоты f_{SCL} сигнала на выводе I2C_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.15.1) Таким образом, частота сигнала на выводе I2C_SCL равна $f_{SCL} = 1/(T_{SCLH} + T_{SCLL})$. (A.15.2) 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

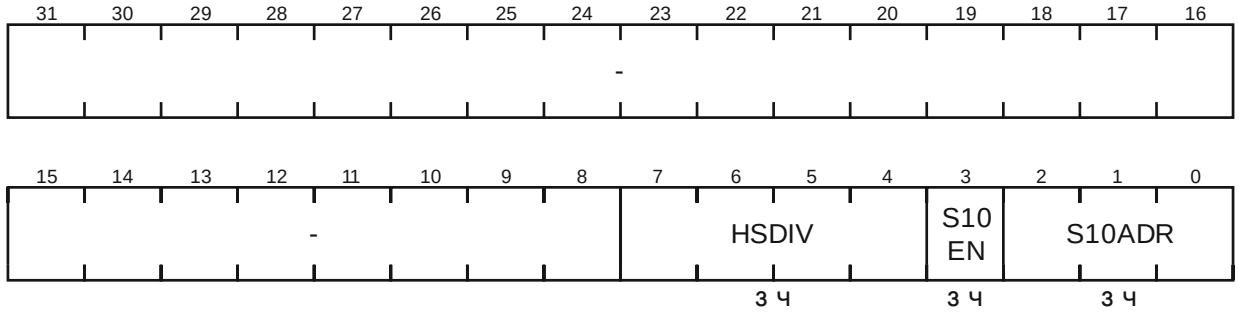
Смещение: + 18h
Сброс: 0h



Поле	Биты	Описание
SMBTOPR	7-0	Поле значения перезагрузки предделителя
–	31-8	Зарезервировано

CTL2 – регистр управления 2

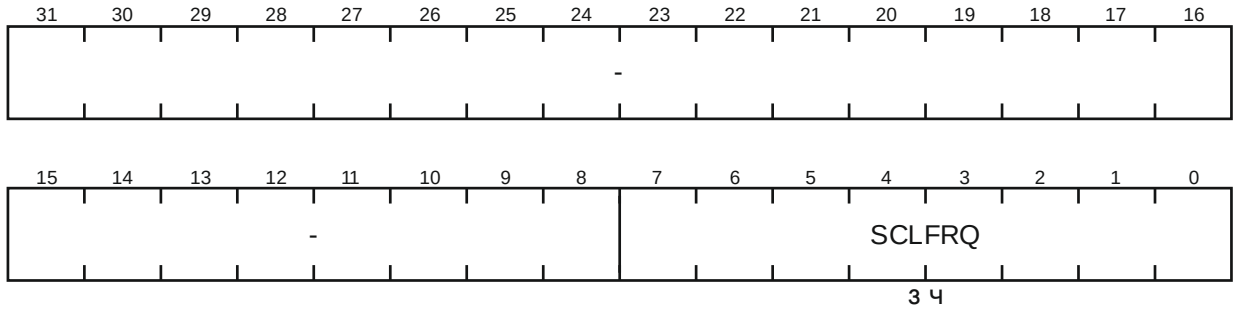
Смещение: + 1Ch
Сброс: 0h



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.15.3) $T_{HSCLL} = 2 \times HSDIV \times (1/f_{PCLK})$. (A.15.4) Таким образом, частота сигнала на выводе I2C_SCL равна $f_{SCL} = 1/(T_{HSCLH} + T_{HSCLL})$. (A.15.5) 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

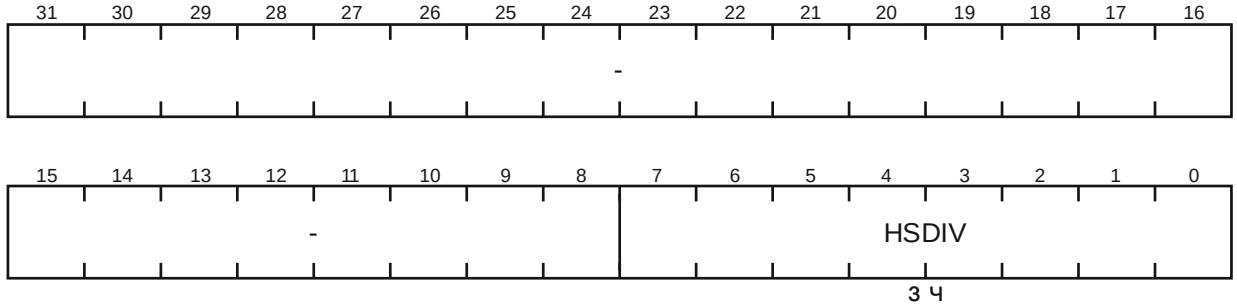
Смещение: + 20h
Сброс: 0h



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



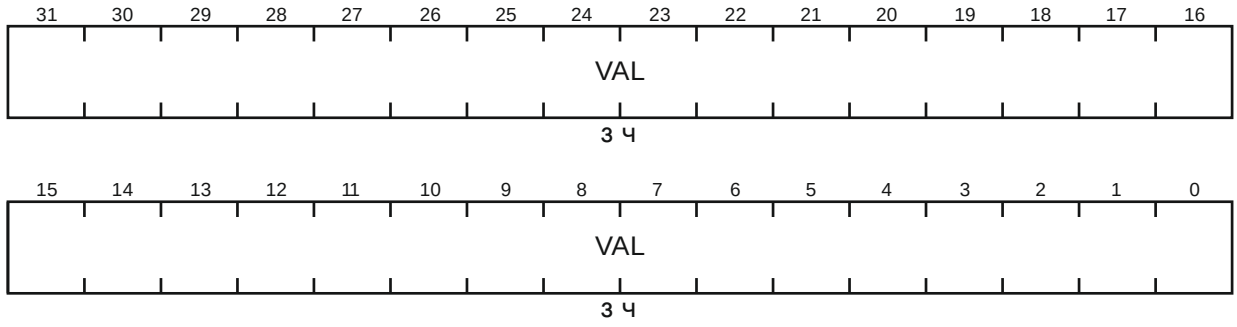
Поле	Биты	Описание
HSDIV	7-0	Старшие разряды делителя HSDIV. См. основное описание поля в регистре CTL2
–	31-8	Зарезервировано

A.16 Регистры блока захвата

Базовый адрес:	4005_1000h	Регистры блока захвата 0
	4005_2000h	Регистры блока захвата 1
	4005_3000h	Регистры блока захвата 2

TSCTR – регистр счетчика таймера

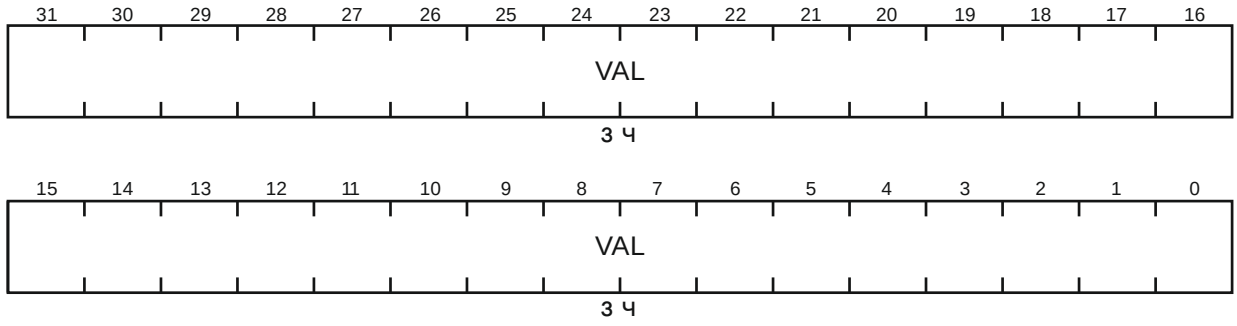
Смещение: + 00h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Запись задает начальное значение таймера. Чтение возвращает текущее значение таймера.

CTRPHS – регистр отложенной загрузки таймера

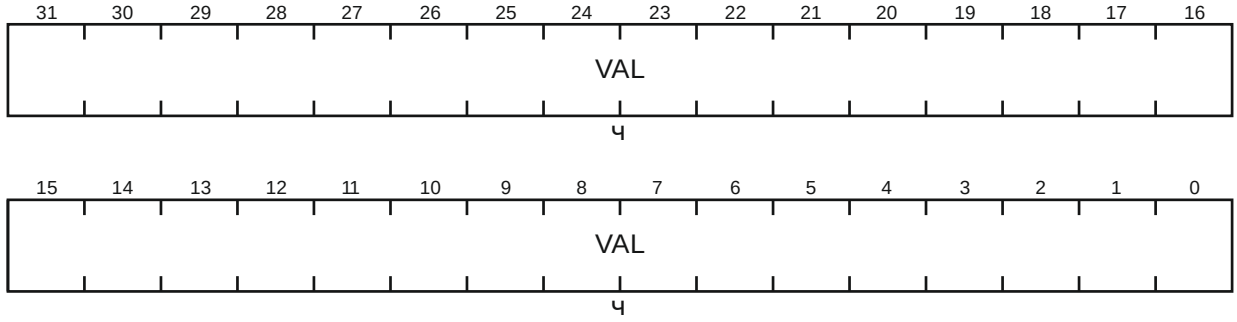
Смещение: + 04h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Значение из регистра загружается в таймер по событиям SYNCI или под управлением процессора. Регистр используется для синхронизации с другими блоками CAP/PWM.

CAP0 – регистр захвата 0

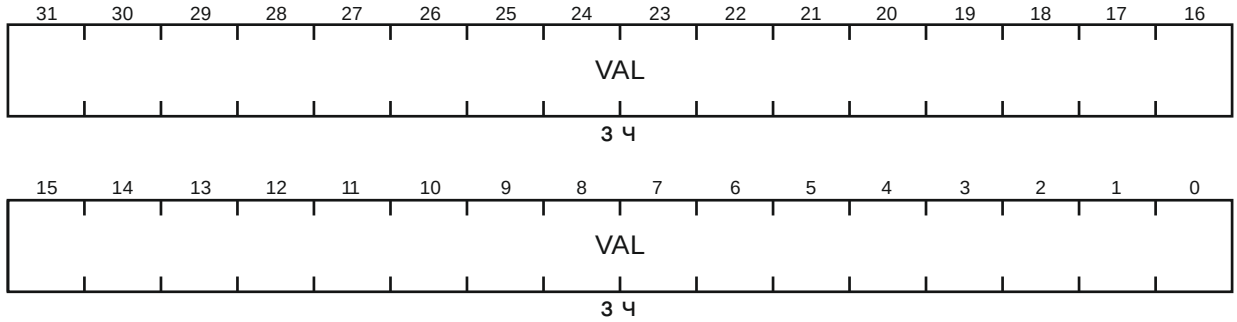
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Регистр используется только в режиме захвата и содержит значение таймера, сохраненное по событию EV0.

PRD – регистр периода

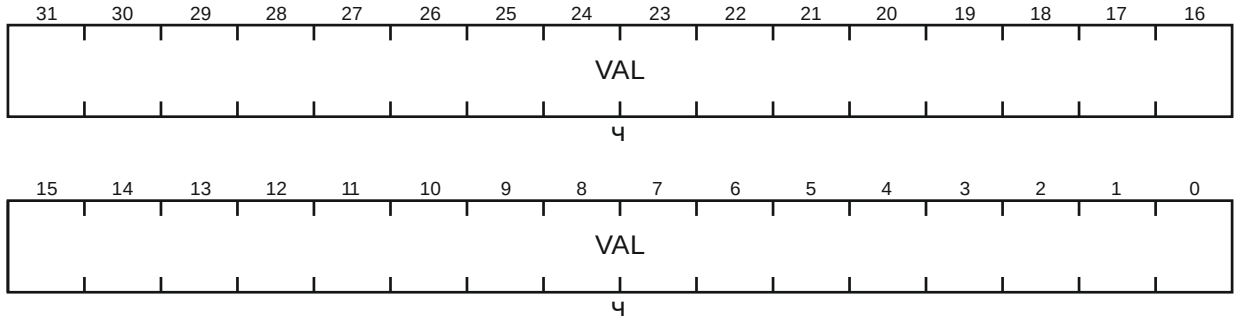
Смещение: + 08h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Регистр используется только в режиме APWM и содержит значение периода генерации. Величина может быть обновлена из регистра отложенной загрузки PRDSHDW.

CAP1 – регистр захвата 1

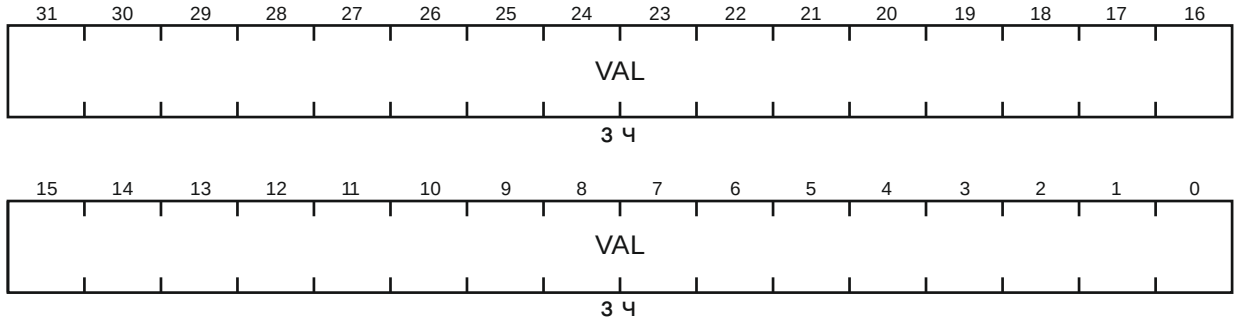
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Регистр используется только в режиме захвата и содержит значение таймера, сохраненное по событию EV1.

CMP – регистр сравнения

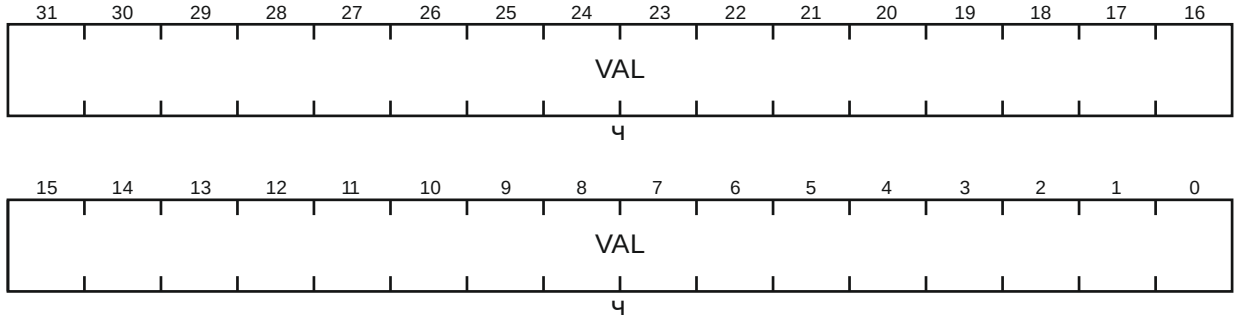
Смещение: + 0Ch
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Регистр используется только в режиме APWM и содержит значение сравнения. Величина может быть обновлена из регистра отложенной загрузки CMPSHDW.

CAP2 – регистр захвата 2

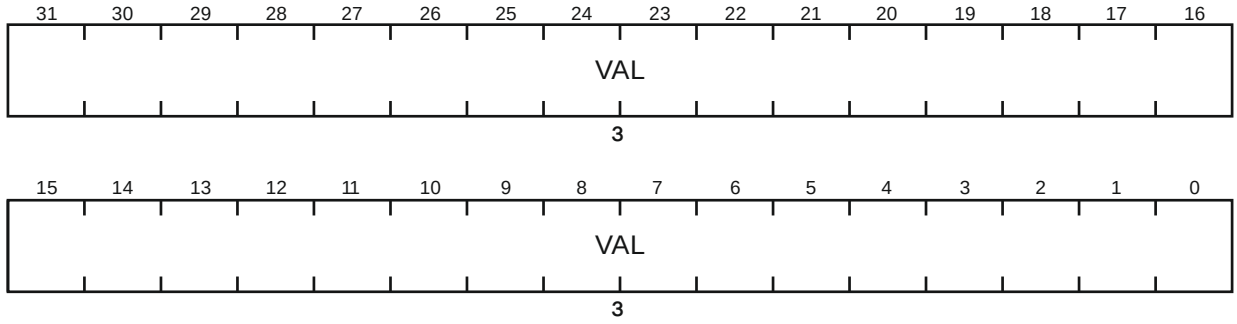
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Регистр используется только в режиме захвата и содержит значение таймера, сохраненное по событию EV2.

PRDSHDW – регистр отложенной загрузки периода

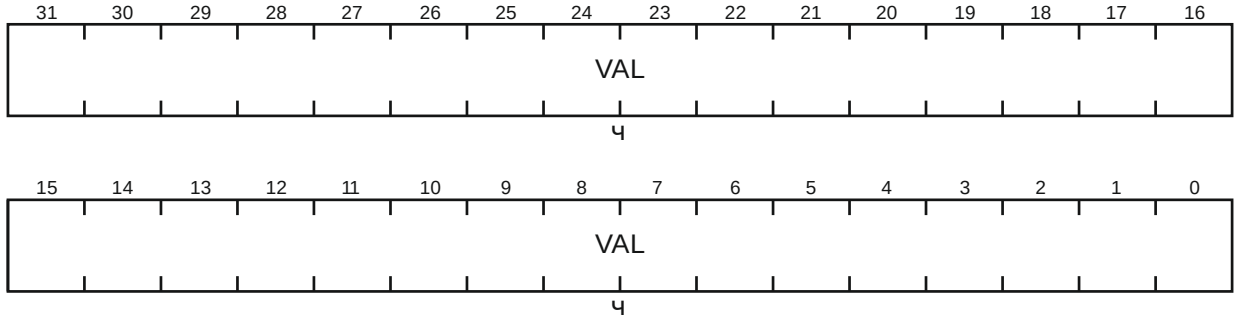
Смещение: + 10h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Регистр используется только в режиме APWM и содержит значение отложенной загрузки периода. По событию CTR = PRD значение будете перенесено в регистр PRD

CAP3 – регистр захвата 3

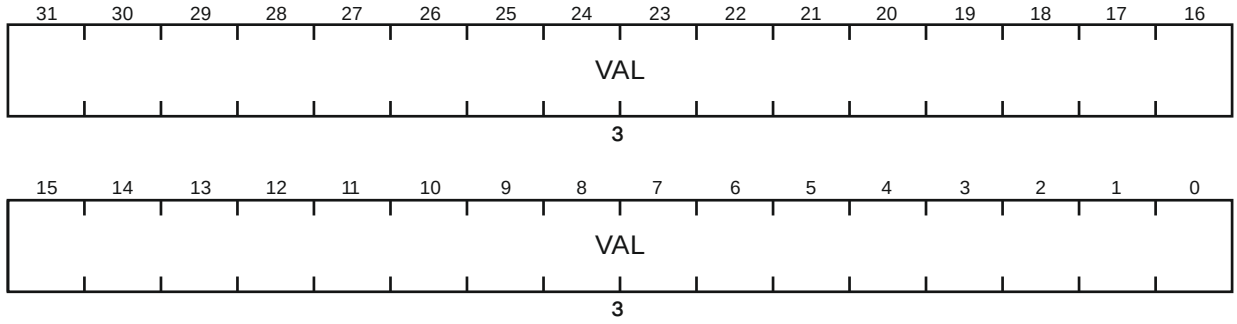
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Регистр используется только в режиме захвата и содержит значение таймера, сохраненное по событию EV3

CMPSHDW – регистр отложенной загрузки сравнения

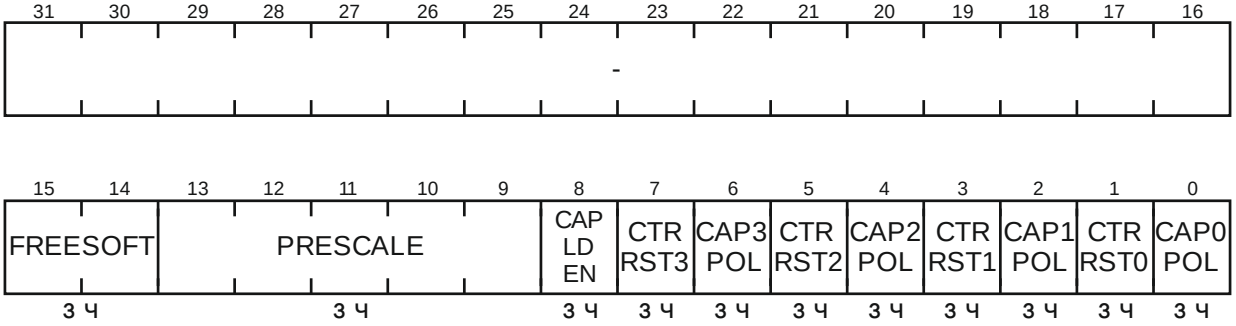
Смещение: + 14h
Сброс: 0h



Поле	Биты	Описание
VAL	31-0	Регистр используется только в режиме APWM и содержит значение отложенной загрузки сравнения. По событию CTR = PRD значение будете перенесено в регистр CMP.

ECCTL0 – регистр контроля 0

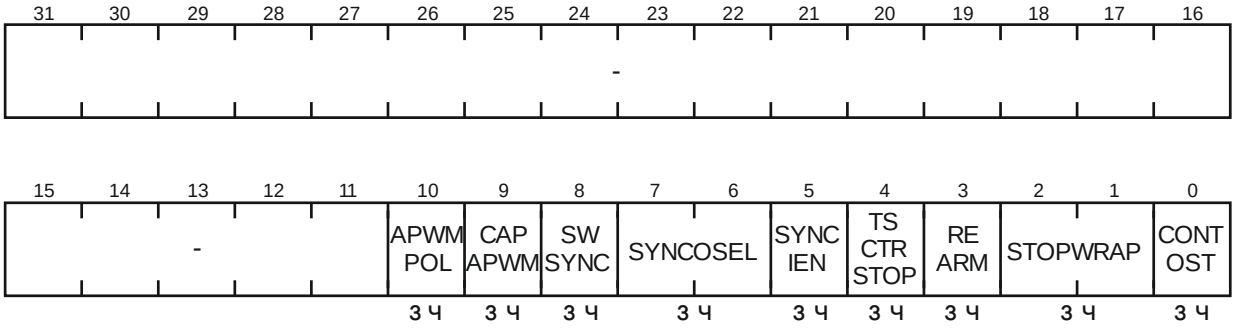
Смещение: + 28h
Сброс: 0h



Поле	Биты	Описание	
FREESOFT	15-14	Управление остановкой таймера в режиме отладки	
		00	Моментальная остановка таймера
		01	Остановка таймера по переполнению
		10, 11	Таймер не останавливается
PRESCALE	13-9	Предварительный делитель. Если записано значение 00h – делитель выключен.	
CAPLDEN	8	Бит разрешения захвата регистрами CAP0 – CAP3	
		0	Запрещено
		1	Разрешено
CTRRSTn	7, 5, 3, 1	Бит сброса таймера после события n (n от 0 до 3)	
		0	Нет действий
		1	Сброс таймера после события n
CAPnPOL	6, 4, 2, 0	Бит выбора фронта захвата (n от 0 до 3)	
		0	Захват по переднему фронту
		1	Захват по заднему фронту
–	31-16	Зарезервировано	

ECCTL1 – регистр контроля 1

Смещение: + 2Ch
Сброс: 0h

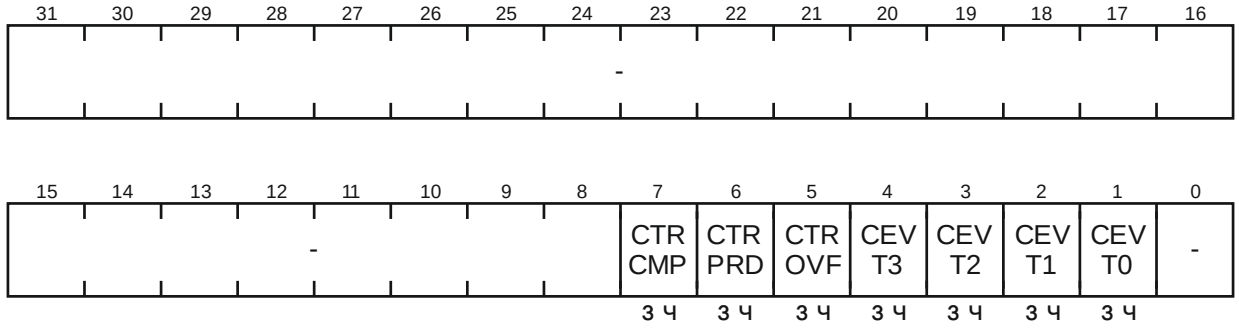


Поле	Биты	Описание	
APWMPOL	10	Бит задания активного уровня в режиме APWM	
		0	Высокий
		1	Низкий
CAPAPWM	9	Бит выбора режима	
		0	Работа в режиме захвата. Блокирование сброса таймера по событию CTR=PRD. Блокирование отложенной загрузки PRD, CMP. Разрешение захвата CAP0–CAP3. Внешний порт работает на вход.
		1	Работа в режиме APWM. Разрешение сброса таймера по CTR = PRD. Разрешение теневой загрузки PRD, CMP. Блокирование захвата CAP0–CAP3. Внешний порт работает на выход.
SWSYNC	8	Межблочная синхронизация таймеров	
		0	Нет действий
		1	Запись единицы: - загружает значение таймера из отложенного регистра при условии, что установлен бит SYNCIEN; - генерирует выходной сигнал синхронизации SYNCO при условии, что в поле SYNCOSEL записано 00b.
		Примечание – В режиме APWM синхронизация также происходит автоматически по событию CTR = PRD.	
SYNCOSEL	7-6	Выбор источника выходного синхросигнала SYNCO	
		00	Пропуск сигнала синхронизации с SYNCI или SWSYNC
		01	Передача события CTR = PRD в качестве выходного сигнала синхронизации
		10, 11	Запрет выходного сигнала синхронизации
SYNCIEN	5	Бит разрешения синхронизации	
		0	Запрещено
		1	Разрешено
TSCTRSTOP	4	Бит управления работой таймера	
		0	Остановлено
		1	Запущено
REARM	3	Запись единицы запускает следующую последовательность действий: сброс управляющего контроллера, разрешение работы управляющего контроллера и загрузку регистров захвата	

Поле	Биты	Описание	
STOPWRAP	2-1	Значение компаратора остановки в режимах захвата	
		00	Останов при значениии счетчика 00
		01	Останов при значениии счетчика 01
		10	Останов при значениии счетчика 10
		11	Останов при значениии счетчика 11
		Примечание – Остановка управляющего контроллера приводит также к блокировке загрузки регистров захвата.	
CONTOST	0	Режим работы захвата	
		0	Циклический
		1	Однократный
–	31-11	Зарезервировано	

ECEINT – регистр маски прерываний

Смещение: + 30h
Сброс: 0h

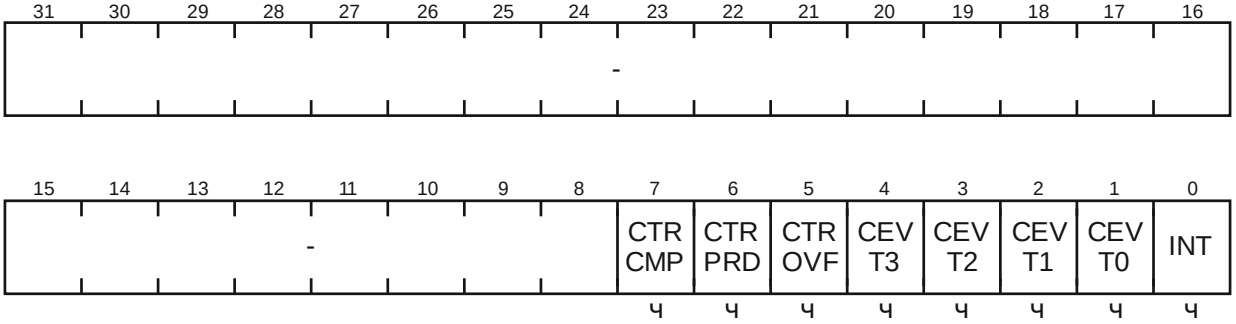


Поле	Биты	Описание
CTRCMP	7	Бит разрешения генерации прерывания по событию CTR = CMP
CTRPRD	6	Бит разрешения генерации прерывания по событию CTR = PRD
CTROVF	5	Бит разрешения генерации прерывания по событию CTROVF
CEVTn	4-1	Бит разрешения генерации прерывания по событию CEVTn (n от 0 до 3)
–	31-8, 0	Зарезервировано

Примечание – Установленный бит разрешает прерывание, сброшенный – запрещает.

ECFLG – регистр статуса прерываний

Смещение: + 34h
Сброс: 0h

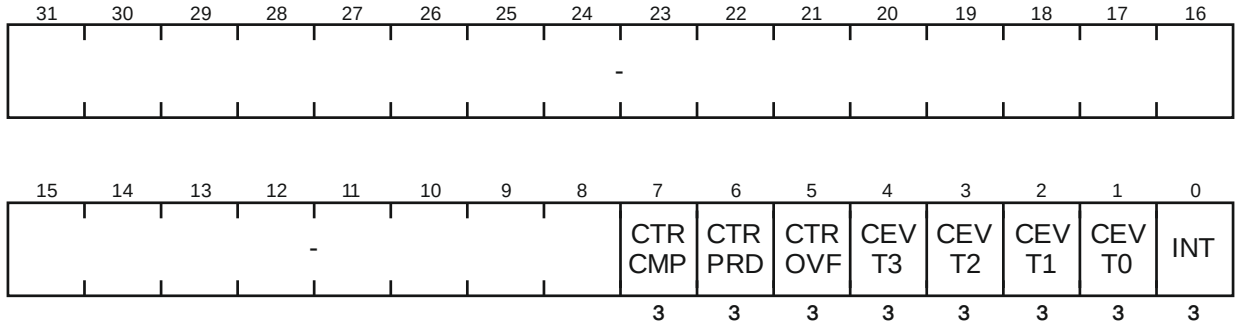


Поле	Биты	Описание	
CTRCMP	7	Флаг прерывания по событию CTR = CMP	
		0	Событие не произошло
		1	Событие произошло
CTRPRD	6	Флаг прерывания по событию CTR = PRD	
		0	Событие не произошло
		1	Событие произошло
CTROVF	5	Флаг прерывания по событию CTROVF	
		0	Событие не произошло
		1	Событие произошло
CEVTn	4-1	Флаг прерывания по событию CEVTn (n от 0 до 3)	
		0	Событие не произошло
		1	Событие произошло
INT	0	Флаг прерывания	
–	31-8	Зарезервировано	

Примечание – Все флаги сбрасываются записью единиц в биты регистра ECCLR.

ECCLR – регистр сброса прерываний

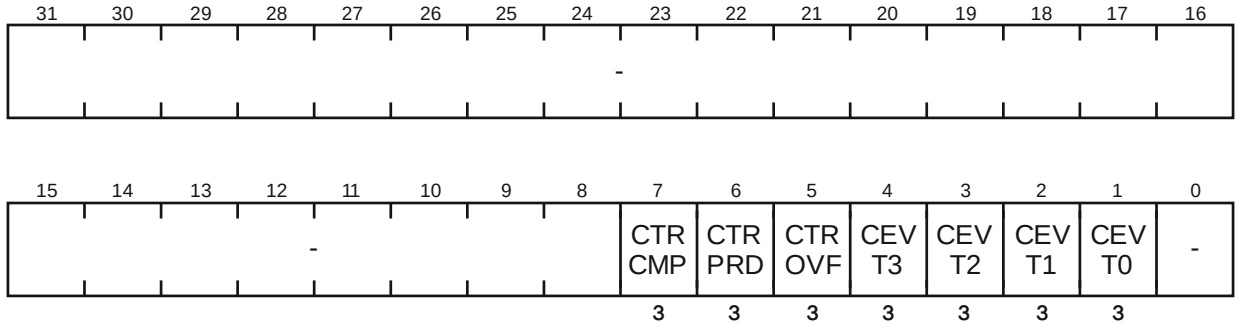
Смещение: + 38h
Сброс: 0h



Поле	Биты	Описание
CTRCMP	7	Запись единицы сбрасывает флаг прерывания по событию CTR = CMP
CTRPRD	6	Запись единицы сбрасывает флаг прерывания по событию CTR = PRD
CTROVF	5	Запись единицы сбрасывает флаг прерывания по событию CTROVF
CEVTn	4-1	Запись единицы сбрасывает флаг прерывания по событию CEVTn (n от 0 до 3)
INT	0	Запись единицы сбрасывает флаг прерывания
–	31-8	Зарезервировано

ECFRC – регистр программных прерываний

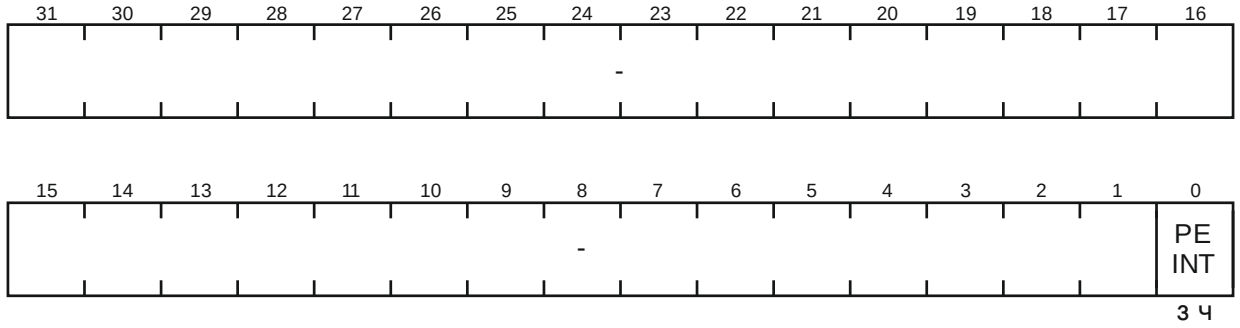
Смещение: + 3Ch
Сброс: 0h



Поле	Биты	Описание
CTRCMP	7	Запись единицы генерирует прерывание по событию CTR = CMP
CTRPRD	6	Запись единицы генерирует прерывание по событию CTR = PRD
CTROVF	5	Запись единицы генерирует прерывание по событию CTROVF
CEVTn	4-1	Запись единицы генерирует прерывание по событию CEVTn (n от 0 до 3)
–	31-8, 0	Зарезервировано

PEINT – регистр активного прерывания

Смещение: + 40h
Сброс: 0h



Поле	Биты	Описание
PEINT	0	Флаг активного прерывания. Устанавливается при возникновении прерывания блока. Сбрасывается только программно записью единицы
–	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-разрядная величина, состоящая 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)

Окончание 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)

Окончание 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 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)
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)

Окончание 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 освобожде-ния шины (00h, 01h)

Приложение В
(обязательное)
Регистры прерываний

Для управления прерываниями используются пять групп регистров ISERi, ICERi, ISPRi, ICPRi и IABRi, где i = 0, 1, 2, см. таблицу В.1. Группы имеют идентичную структуру. Набор прерываний, которыми управляет регистр группы, зависит от индекса i. На рисунке В.1 показана одна группа регистров и указано соответствие номеров векторов прерываний и бит регистров. Управление прерыванием осуществляется записью единицы или нуля в соответствующий бит. Допускается одновременное управление несколькими прерываниями.

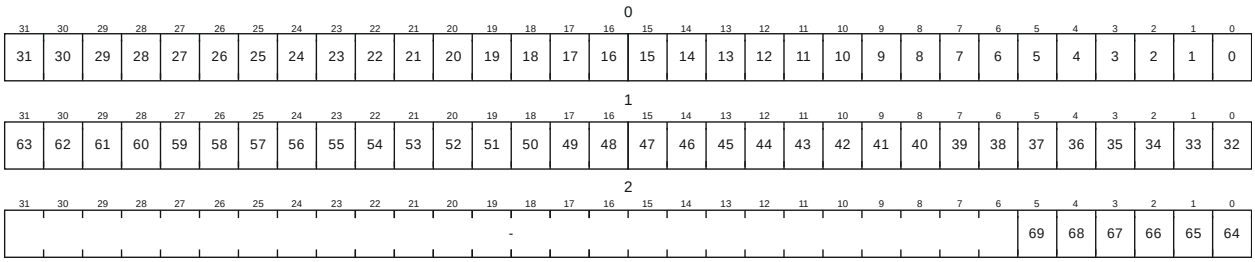


Рисунок В.1 – Соответствие векторов прерываний и бит управляющих регистров

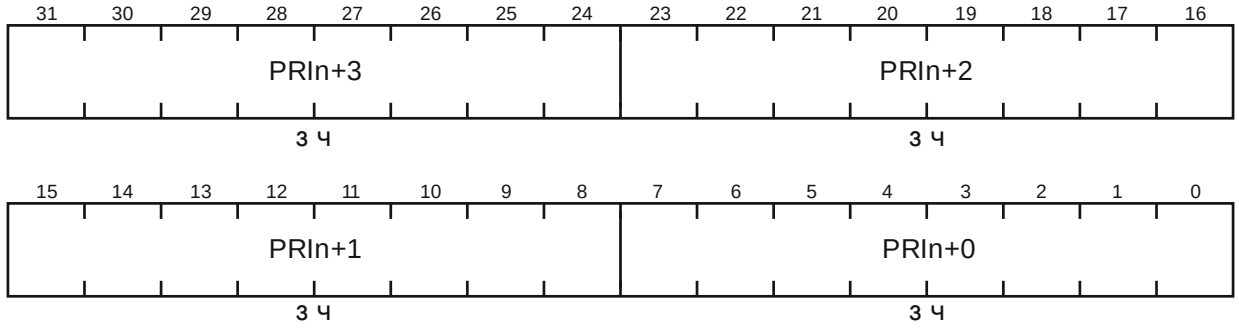
Таблица В.1 – 32-разрядные регистры управления прерываниями

Мнемоника и назначение	Операция над битом		Влияние на соответствующее прерывание
ISERi Регистр разрешения прерываний от источников	Чтение	0	Прерывание запрещено
		1	Прерывание разрешено
	Запись	0	Нет влияния
		1	Разрешение прерывания
ICERi Регистр сброса разрешения прерываний от источников	Чтение	0	Прерывание запрещено
		1	Прерывание разрешено
	Запись	0	Нет влияния
		1	Запрет прерывания
ISPRi Регистр ждущих прерываний	Чтение	0	Ждущего прерывания нет
		1	Есть ждущее прерывание
	Запись	0	Нет влияния
		1	Установка ждущего прерывания (программное прерывание)
ICPRi Регистр сброса ждущих прерываний	Чтение	0	Ждущего прерывания нет
		1	Есть ждущее прерывание
	Запись	0	Нет влияния
		1	Сброс ждущего прерывания
IABRi Регистр флагов прерываний	Чтение	0	Флага прерывания нет
		1	Флаг прерывания установлен. Сбрасывается аппаратно по окончании обслуживания прерывания

Для задания приоритетов прерываний используются регистры IPRi, где i от 0 до 33. В таблице В.2 представлен формат регистра IPRi.

Таблица В.2 – Регистр приоритетов

IPRi Сброс: 0000000h



Поле	Биты	Описание
PRIn+3	31-24	Поле задания приоритета (n+3) вектора прерывания
PRIn+2	23-16	Поле задания приоритета (n+2) вектора прерывания
PRIn+1	15-8	Поле задания приоритета (n+1) вектора прерывания
PRIn+0	7-0	Поле задания приоритета n-ого вектора прерывания $n = 4 \times i$

Примечание – запись в PRIn[4:0] игнорируется, читаются - нули. Диапазон допустимых значений для битов PRIn[7:5] - 0-7. Подробнее о том, как приоритеты разделяются на группы и подгруппы в таблице В.3.

Таблица В.3 – Распределение приоритетов на группы и подгруппы

Поле PRIGROUP регистра AIRCR ядра	PRIn[7:5]			Количество	
	Распределение на группы и подгруппы	Биты группы	Биты подгруппы	Группы	Подгруппы
0-4	0bxxx	[7:5]	Отсутствуют	8	0
5	0bxx.y	[7:6]	[5]	4	2
6	0bx.yy	[7]	[6:5]	2	4
7	0byyy	Отсутствуют	[7:5]	0	8

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в документе	Номер документа	Подпись	Дата
	изме-ненных	замененных	новых	анну-лиро-ванных				