

МИКРОСХЕМА ИНТЕГРАЛЬНАЯ
1874BE66T

Руководство пользователя

Содержание

1 Введение.....	4
2 Назначение и основные характеристики микросхемы 1874BE66T.....	5
2.1 Архитектурные характеристики микросхемы.....	5
2.2 Конструктивные характеристики микросхемы.....	6
2.3 Функциональная характеристика микроконтроллера.....	12
2.4 Электрические параметры микросхем.....	18
3 Структура и организация ИС 1874BE66T.....	21
3.1 Ключевые особенности микроконтроллера.....	21
3.2 Системные ресурсы микроконтроллера.....	22
3.3 Синхронизация.....	23
3.4 Встроенные периферийные устройства.....	25
3.5 Специальные режимы работы.....	26
4 Система команд ИС 1874BE66T.....	28
4.1 Общая характеристика системы команд.....	28
4.2 Режимы адресации.....	30
4.3 Выборы режима адресации ассемблера.....	33
4.4 Стандарты и соглашения программного обеспечения.....	33
4.5 Защита и руководящие принципы программного обеспечения.....	35
5 Организация памяти.....	36
5.1 Карта памяти микроконтроллеров.....	36
5.2 Работа с окнами.....	43
6 Стандартные и PTS прерывания.....	47
6.1 Краткий обзор прерываний.....	47
6.2 Сигналы и регистры прерываний.....	48
6.3 Источники и приоритеты прерывания.....	50
6.4 Время ожидания прерывания.....	54
6.5 Программирование прерывания.....	56
6.6 Инициализация блоков управления PTS.....	63
7 Порты ввода-вывода.....	89
7.1 Описание портов ввода-вывода.....	89
7.2 Входные порты 0 и 1.....	89
7.3 Двухнаправленные порты 2, 5, 7.....	91
7.4 Двухнаправленные порты 3, 4 (шина адресов/данных).....	98
7.5 Стандартный выходной порт 6.....	101
8 Генератор формы сигнала.....	103
8.1 Краткий обзор функций генератора формы сигнала.....	103
8.2 Сигналы и регистры генератора формы сигнала.....	104
8.3 Работа генератора формы сигнала.....	106
8.4 Программирование ГФС.....	111
8.5 Определение состояния ГФС.....	117
8.6 Разрешение прерываний генератора формы сигнала.....	117
9 Широтно-импульсный модулятор.....	119
9.1 Краткий обзор функционирования блока ШИМ.....	119
9.2 Сигналы и регистры блока ШИМ.....	119
9.3 Работа широтно-импульсного модулятора.....	120
9.4 Программирование частоты и периода.....	121
9.5 Программирование скважности (Duty cycle).....	122
10 Процессор событий ERA.....	126
10.1 Краткий обзор функционирования ERA.....	126
10.2 Регистры и сигналы таймера/счетчика ERA.....	127
10.3 Краткий обзор функционирования таймеров/счетчиков.....	130

10.4	Краткое описание функционирования канала ЕРА.....	133
10.5	Программирование ЕРА и таймеров/счетчиков	137
10.6	Разрешения прерывания ЕРА	144
10.7	Определение состояния события	144
11	Генератор программируемого тактового сигнала.....	145
11.1	Функциональное описание	145
11.2	Применение генератора программируемого тактового сигнала.....	147
12	Аналого-цифровой преобразователь АЦП	148
12.1	АЦП. Краткий функциональный обзор	148
12.2	Сигналы и регистры АЦП.....	148
12.3	Работа АЦП	149
12.4	Программирование АЦП.....	151
12.5	Определение состояния АЦП и результатов преобразования	154
12.6	Примеры схем внешнего интерфейса и ошибок АЦ преобразования.....	155
13	Специальные режимы работы.....	163
13.1	Сигналы и регистры управления специальными режимами	163
13.2	Ограничение потребления мощности	165
13.3	Режим холостого хода IDLE.....	165
13.4	Режим низкого потребления POWERDOWN	166
13.5	Режим внутрисхемной эмуляции ONCE	170
13.6	Зарезервированные тестовые режимы.....	170
14	Интерфейс внешней памяти	171
14.1	Сигналы и регистры интерфейса внешней памяти	171
14.2	Регистры и байты конфигурации кристалла	175
14.3	Мультиплексирование и разрядность шины.....	178
14.4	Управление длительностью шинного цикла сигналом READY	180
14.5	Режимы управления шиной	181
15	Программирование постоянной памяти.....	193
15.1	Режимы программирования.....	193
15.2	Карта распределения памяти EPROM	193
15.3	Защита памяти.....	194
15.4	Ширина программирующего импульса.....	197
15.5	Модифицированный QUICK-PULSE алгоритм.....	198
15.6	Вывода в режиме программирования	199
15.7	Вход в режимы программирования	201
15.8	Режим программирования SLAVE.....	202
15.9	Режим автопрограммирования AUTO	210
15.10	Режим вывода содержимого памяти ROM-DUMP	214
15.11	Режим RUN-TIME	215
16	Рекомендации по отладочным средствам ИС 1874ВЕ66Т.....	216
16.1	Программаторы для программируемого варианта ИС 1874ВЕ66Т.....	216
16.2	Описание инструментальных средств для ИС 1874ВЕ66Т	216
16.3	Интегрированный пакет разработки и отладки систем на базе микроконтроллера (Project-96).....	216
16.4	Отладчик-симулятор микроконтроллеров семейства MCS-196 PDS-96.....	218
16.5	Кросс-макроассемблер МСА-96.....	220
16.6	Кросс-компилятор языка Си МСС-96.....	220
17	Заключение.....	222
	Приложение А (обязательное) Система команд.....	223
	Приложение Б (обязательное) Описание сигналов	263
	Приложение В (обязательное) Регистры микроконтроллера.....	276
	Лист регистрации изменений	324

1 Введение

В настоящем руководстве КФДЛ.431295.033 приведено описание архитектуры, функционального построения, системы команд и особенностей применения микросхемы 1874ВЕ66Т, которая представляет собой ИС однокристалльного 16-разрядного микроконтроллера с тактовой частотой 16 МГц, ОЗУ 488×8 бит, 14-канальным 8/10-разрядным АЦП, с функцией управления электроприводом и энергонезависимой памятью программ типа EPROM объемом 16К×8.

Разработанная микросхема будет служить основой для создания перспективных систем управления.

Одной из быстроразвивающихся областей встраиваемых систем управления являются системы реального времени, строящиеся на базе современных микроконтроллеров. Остро стоит необходимость реализовывать сложные алгоритмы управления, использующие на входе большое количество цифровых и аналоговых сигналов, когда сигналы отклика должны генерироваться с минимально возможной временной задержкой. Встроенные системы управления часто являются критичными по площади монтажа, рассеиваемой мощности и стоимости системы в целом, поэтому применение современных высокопроизводительных микроконтроллеров позволяет решить эти проблемы.

Для встраиваемых систем управления необходимо использовать микроконтроллеры, которые обладают следующими свойствами:

- высокой степенью системной интеграции;
- уменьшенной необходимостью в дополнительных периферийных устройствах и, следовательно, накладных расходах программно-аппаратного обеспечения;
- обеспечение стабильности и надежности систем.

Непрекращающееся увеличение сложности встраиваемых систем управления требует более существенного увеличения производительности центрального процессорного устройства ЦПУ и периферии микроконтроллера по сравнению с традиционными 8-разрядными микроконтроллерами.

16-разрядный однокристалльный КМОП микроконтроллер 1874ВЕ66Т был разработан для удовлетворения высоких требований к рабочим характеристикам систем управления в реальном времени.

В настоящее время одним из основных факторов обеспечения конкурентоспособности отечественной радиоэлектронной аппаратуры и ее живучести является применение при ее разработке и производстве импортонезависимой элементной базы. Особенно это касается аппаратуры для вооружения, военной и специальной техники. Импортозамещение электронных компонентов наиболее эффективно в случае использования полных функциональных аналогов изделий микроэлектроники.

Появление отечественного варианта высокопроизводительного 16-разрядного микроконтроллера для систем управления и призвано обеспечить импортонезависимость проектируемых на его основе устройств.

Настоящее руководство может служить практическим пособием по применению микроконтроллеров для разработчиков систем на основе ИС 1874ВЕ66Т.

2 Назначение и основные характеристики микросхемы 1874BE66T

Микросхема 1874BE66T – 16-разрядный микроконтроллер с расширенной периферией, функцией управления электроприводом и энергонезависимой, однократно программируемой памятью программ типа EPROM.

Архитектура ИС ориентирована на создание управляющих систем, функционирующих в режиме реального времени с возможностью адаптации и модификации под конкретные приложения. Изделия могут служить элементной базой для систем управления различной аппаратурой, в том числе спецаппаратурой, силовой электроникой, бытовой техникой, теле-, аудио- и видеоаппаратурой, в компьютерной технике, автомобильной технике и т.д., также изделия предназначены для применения в системах приема, передачи и обработки информации, встроенного управления и автономных необслуживаемых аппаратов специального назначения.

Применение ИС 1874BE66T позволит обеспечить требуемые тактико-технические данные важнейших систем В и ВТ и выполнить требования ТЗ на аппаратуру (комплексы) по назначению, энергопотреблению и массогабаритным показателям. Это даст возможность обеспечить комплектование аппаратуры специального назначения отечественными ИС взамен аналогичных импортных изделий.

Наличие средств инструментальной отладки обеспечивает как эффективное проектирование систем на основе микроконтроллера, так и возможность смены алгоритма работы при создании модификаций систем.

Микросхема представляет собой высокопроизводительный 16-разрядный микроконтроллер 1874BE66T со встроенной памятью ЭПЗУ 16К×8 типа EPROM и является функциональным аналогом микросхемы 87C196MD фирмы Intel, США.

2.1 Архитектурные характеристики микросхемы

Микросхема 1874BE66T имеет следующие архитектурные характеристики:

- разрядность АЛУ, бит	16
- разрядность данных, бит	16
- тактовая частота, МГц	от 8 до 16
- динамически конфигурируемая шина данных, бит	8 или 16
- встроенная память программ (типа EPROM), бит	16К×8
- регистровое ОЗУ, бит	488×8
- число источников прерывания	18
- число параллельных 8-разрядных портов ввода-вывода	8
в том числе:	
- портов ввода-вывода	5
- портов ввода	2
- портов вывода	1
- разрядность сторожевого таймера	16
- число таймеров/счетчиков	2
- разрядность таймеров/счетчиков	16
- число каналов аналого-цифрового преобразователя	14
- число разрядов аналого-цифрового преобразователя	8, 10
- процессор событий ЕРА, вход/выход	6/12
- периферийный сервер PTS	1
- генератор программируемой частоты	1

- трехфазный генератор сигналов (функция управления двигателем)	1
- блок ШИМ сигналов	2
- номинальное напряжение питания цифровой части, В	5,0
- напряжение питания аналоговой части, В	4,0...5,5

2.2 Конструктивные характеристики микросхемы

Микросхема разработана в металлокерамическом планарном корпусе с четырехсторонним расположением выводов 4235.88-1.

Герметизация микросхем осуществляется шовной роликовой сваркой. Показатель герметичности микросхем по эквивалентному нормализованному потоку – не более $6,65 \times 10^{-3}$ Па·см³/с.

Номинальное значение напряжения питания микросхем плюс 5 В. Допустимое отклонение напряжения питания от номинального ± 10 %. Амплитуда пульсаций напряжения питания не более 50 мВ.

Напряжение источника опорного напряжения от 4,0 до 5,5 В. Допустимое отклонение напряжения питания от крайних значений минус 1 % для напряжения 4,0 В и плюс 1 % для напряжения 5,5 В.

Микросхема не имеет собственных резонансных частот ниже 100 Гц.

Функциональные возможности микросхемы 1874BE66T соответствуют функциональным возможностям микросхемы-аналога 87C196MD.

Более подробные конструктивные характеристики микросхемы 1874BE66T приведены в технических условиях АЕЯР.431280.713.

Структурная схема приведена на рисунке 2.1.

Условное графическое обозначение микросхем приведено на рисунке 2.2.

Назначение выводов микросхемы дано в таблице 2.1.

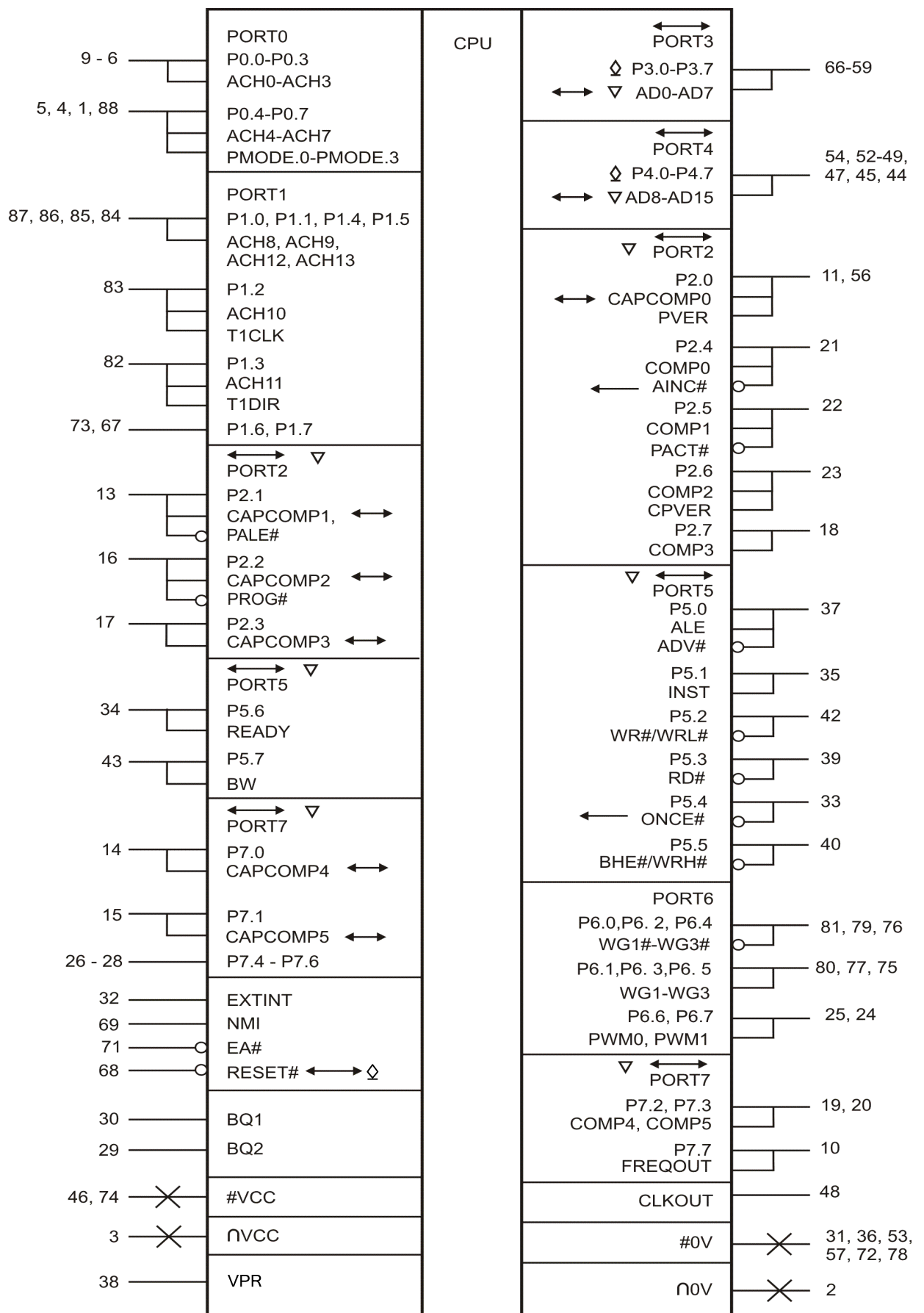


Рисунок 2.2 – Условно-графическое обозначение ИС 1874BE66Т

Т а б л и ц а 2.1 – Назначение выводов микросхемы 1874BE66T

Обозначение вывода	Номер вывода	Функциональное назначение	Тип вывода	Обозначение альтернативной функции вывода
1	2	3	4	5
P0.0	9	Вход «порт 0», 0 разряд Вход АЦП, канал 0	I I	ACH0
P0.1	8	Вход «порт 0», 1 разряд Вход АЦП, канал 1	I I	ACH1
P0.2	7	Вход «порт 0», 2 разряд Вход АЦП, канал 2	I I	ACH2
P0.3	6	Вход «порт 0», 3 разряд Вход АЦП, канал 3	I I	ACH3
P0.4	5	Вход «порт 0», 4 разряд Вход АЦП, канал 4 Вход «режим программирования», 0 разряд	I I I	ACH4, PMODE.0
P0.5	4	Вход «порт 0, 5 разряд» Вход АЦП, канал 5 Вход «режим программирования», 1 разряд	I I I	ACH5, PMODE.1
P0.6	1	Вход «порт 0, 6 разряд» Вход АЦП, канал 6 Вход «режим программирования», 2 разряд	I I I	ACH6, PMODE.2
P0.7	88	Вход «порт 0, 7 разряд» Вход АЦП, канал 7 Вход «режим программирования», 3 разряд	I I I	ACH7, PMODE.3
P1.0	87	Вход «порт 1», 0 разряд Вход АЦП, канал 8	I I	ACH8
P1.1	86	Вход «порт 1», 1 разряд Вход АЦП, канал 9	I I	ACH9
P1.2	83	Вход «порт 1», 2 разряд Вход АЦП, канал 10 Вход «синхронизация таймера 1»	I I I	ACH10 T1CLK
P1.3	82	Вход «порт 1», 3 разряд Вход АЦП, канал 11 Вход «режим таймера 1»	I I I	ACH11 T1DIR
P1.4	85	Вход «порт 1», 4 разряд Вход АЦП, канал 12	I I	ACH12
P1.5	84	Вход «порт 1», 5 разряд Вход АЦП, канал 13	I I	ACH13
P1.6	73	Вход «порт 1», 6 разряд	I	
P1.7	67	Вход «порт 1», 7 разряд	I	
P2.0	11, 56	Вход/выход «порт 2», 0 разряд Вход/выход «выборка- сравнение», 0 канал Выход «верификация»	I/O/Z I/O O	CAPCOMP0 PVER
P2.1	13	Вход/выход «порт 2», 1 разряд Вход/выход «выборка- сравнение», 1 канал Вход «строб записи»	I/O/Z I/O I	CAPCOMP1 PALE#
P2.2	16	Вход/выход «порт 2», 2 разряд Вход/выход «выборка-сравнение», 2 канал Вход «программирование»	I/O/Z I/O I	CAPCOMP2, PROG#

Продолжение таблицы 2.1

1	2	3	4	5
P2.3	17	Вход/выход «порт 2», 3 разряд Вход/выход «выборка-сравнение, 3 канал»	I/O/Z I/O	CAPCOMP3
P2.4	21	Вход/выход «порт 2», 4 разряд Выход «сравнение», 0 канал Вход «автоинкремент»	I/O/Z O I	COMP0 AINC#
P2.5	22	Вход/выход «порт 2», 5 разряд Выход «сравнение», 1 канал Выход «подтверждение программирования»	I/O/Z O O	COMP1 PACT#
P2.6	23	Вход/выход «порт 2», 6 разряд Выход «сравнение», 2 канал Выход «ошибка программирования»	I/O/Z O O	COMP2 CPVER
P2.7	18	Вход/выход «порт 2», 7 разряд Выход «сравнение», 3 канал	I/O/Z O	COMP3
P3.0	66	Вход/выход «порт 3», 0 разряд Вход/выход «адрес/данные», 0 разряд	I/O/2 I/O/Z	AD0
P3.1	65	Вход/выход «порт 3», 1 разряд Вход/выход «адрес/данные», 1 разряд	I/O/2 I/O/Z	AD1
P3.2	64	Вход/выход «порт 3», 2 разряд Вход/выход «адрес/данные», 2 разряд	I/O/2 I/O/Z	AD2
P3.3	63	Вход/выход «порт 3», 3 разряд Вход/выход «адрес/данные», 3 разряд	I/O/2 I/O/Z	AD3
P3.4	62	Вход/выход «порт 3», 4 разряд Вход/выход «адрес/данные», 4 разряд	I/O/2 I/O/Z	AD4
P3.5	61	Вход/выход «порт 3», 5 разряд Вход/выход «адрес/данные», 5 разряд	I/O/2 I/O/Z	AD5
P3.6	60	Вход/выход «порт 3», 6 разряд Вход/выход «адрес/данные», 6 разряд	I/O/2 I/O/Z	AD6
P3.7	59	Вход/выход «порт 3», 7 разряд Вход/выход «адрес/данные», 7 разряд	I/O/2 I/O/Z	AD7
P4.0	54	Вход/выход «порт 4», 0 разряд Вход/выход «адрес/данные», 8 разряд	I/O/2 I/O/Z	AD8
P4.1	52	Вход/выход «порт 4», 1 разряд Вход/выход «адрес/данные», 9 разряд	I/O/2 I/O/Z	AD9
P4.2	51	Вход/выход «порт 4», 2 разряд Вход/выход «адрес/данные», 10 разряд	I/O/2 I/O/Z	AD10
P4.3	50	Вход/выход «порт 4», 3 разряд Вход/выход «адрес/данные», 11 разряд	I/O/2 I/O/Z	AD11
P4.4	49	Вход/выход «порт 4», 4 разряд Вход/выход «адрес/данные», 12 разряд	I/O/2 I/O/Z	AD12
P4.5	47	Вход/выход «порт 4», 5 разряд Вход/выход «адрес/данные», 13 разряд	I/O/2 I/O/Z	AD13
P4.6	45	Вход/выход «порт 4», 6 разряд Вход/выход «адрес/данные», 14 разряд	I/O/2 I/O/Z	AD14
P4.7	44	Вход/выход «порт 4», 7 разряд Вход/выход «адрес/данные», 15 разряд	I/O/2 I/O/Z	AD15
P5.0	37	Вход/выход «порт 5», 0 разряд Выход «разрешение записи адреса» Выход «адрес действителен»	I/O/Z O O	ALE ADV#

Продолжение таблицы 2.1

1	2	3	4	5
P5.1	35	Вход/выход «порт 5», 1 разряд Выход «чтение команды»	I/O/Z O	INST
P5.2	42	Вход/выход «порт 5», 2 разряд Выход «запись» Выход «запись младшего байта»	I/O/Z O O	WR# WRL#
P5.3	39	Вход/выход «порт 5», 3 разряд Выход «чтение»	I/O/Z O	RD#
P5.4	33	Вход/выход «порт 5», 4 разряд Вход «внутрисхемная изоляция»	I/O/Z I	ONCE#
P5.5	40	Вход/выход «порт 5», 5 разряд Выход «выбор старшего байта» Выход «запись старшего байта»	I/O/Z O O	BHE# WRH#
P5.6	34	Вход/выход «порт 5», 6 разряд Вход «готовность»	I/O/Z I	READY
P5.7	43	Вход/выход «порт 5», 7 разряд Вход «разрядность шины данных»	I/O/Z I	BW
P6.0	81	Выход «порт 6», 0 разряд Выход «фаза 1 инверсная»	O O	WG1#
P6.1	80	Выход «порт 6», 1 разряд Выход «фаза 1»	O O	WG1
P6.2	79	Выход «порт 6», 2 разряд Выход «фаза 2 инверсная»	O O	WG2#
P6.3	77	Выход «порт 6», 3 разряд Выход «фаза 2»	O O	WG2
P6.4	76	Выход «порт 6», 4 разряд Выход «фаза 3 инверсная»	O O	WG3#
P6.5	75	Выход «порт 6», 5 разряд Выход «фаза 3»	O O	WG3
P6.6	25	Выход «порт 6», 6 разряд Выход «ШИМ 0»	O O	PWM0
P6.7	24	Выход «порт 6», 7 разряд Выход «ШИМ 1»	O O	PWM1
P7.0	14	Вход/выход «порт 7, 0 разряд» Вход/выход «выборка-сравнение», 4 канал	I/O/Z I/O	CAPCOMP4
P7.1	15	Вход/выход «порт 7, 1 разряд» Вход/выход «выборка-сравнение», 5 канал	I/O/Z I/O	CAPCOMP5
P7.2	19	Вход/выход «порт 7, 2 разряд» Выход «сравнение», 4 канал	I/O/Z O	COMP4
P7.3	20	Вход/выход «порт 7, 3 разряд» Выход «сравнение», 5 канал	I/O/Z O	COMP5
P7.4	26	Вход/выход «порт 7, 4 разряд»	I/O/Z	
P7.5	27	Вход/выход «порт 7, 5 разряд»	I/O/Z	
P7.6	28	Вход/выход «порт 7, 6 разряд»	I/O/Z	
P7.7	10	Вход/выход «порт 7, 7 разряд» Выход «программируемый тактовый сигнал»	I/O/Z O	FREQOUT
EXTINT	32	Вход «внешнее прерывание»	I	

Окончание таблицы 2.1

1	2	3	4	5
NMI	69	Вход «немаскируемое прерывание»	I	
EA#	71	Вход «обращение к внешней памяти»	I	
CLKOUT	48	Выход «системный тактовый сигнал»	O	
RESET#	68	Вход/выход «сброс»	I/O/2	
BQ1	30	Выход для подключения кварцевого резонатора/ вход тактового сигнала	– I	
BQ2	29	Выход для подключения кварцевого резонатора	–	
VPR	38	Выход «напряжение программирования» Вход «возврат из режима пониженного потребления»	– I	
#VCC	46,74	Выводы питания цифровой части микросхемы	–	
#0V	31, 36, 53, 57, 72, 78	Общие выводы цифровой части микросхемы	–	
Π VCC	3	Вывод питания аналоговой части микросхемы	–	
Π 0V	2	Общий вывод аналоговой части микросхемы	–	
<p>Примечания</p> <p>1 Выводы 12, 41, 55, 58, 70 не задействованы, могут быть подключены к общим выводам цифровой части микросхемы (#0V).</p> <p>2 Выводы 44, 45, 47, 49–52, 54, 59–66 при использовании их в качестве входов/выходов портов 3 и 4 функционируют или как комплементарные, или как выводы с открытым стоком. При использовании в качестве шины адреса/данных (AD0 – AD15) они функционируют как входы/выходы с тремя состояниями.</p> <p>3 В графе «Тип вывода»: I – вход, O – выход, Z – третье состояние, 2 – режим открытого стока.</p>				

2.3 Функциональная характеристика микроконтроллера

В таблице 2.2 приведены основные функциональные параметры, характеризующие ИС с точки зрения применения в аппаратуре.

Т а б л и ц а 2.2 – Основные функциональные параметры ИС 1874BE66T

Наименование параметра, единица измерения	Значение параметра
Объем встроенной программной памяти, Кбайт	16 (EPROM)
Тактовая частота, МГц	от 8 до 16
Регистровое ОЗУ, бит	488 × 8
Источники прерываний/векторы вызова подпрограмм	18/33
Умножение (16×16 бит), мкс	1,75
Деление (32/16 бит), мкс	3,0
Режимы энергосбережения	2
Порты ввода-вывода	5×8
Порты ввода	2×8
Порты вывода	1×8

Окончание таблицы 2.2

Наименование параметра, единица измерения	Значение параметра
16-разрядный сторожевой таймер	1
Реконфигурируемая 8/16-разрядная шина данных	да
Процессор событий (ЕРА)	
высокоскоростные модули захвата/сравнения	6
высокоскоростные модули сравнения	6
Сервер периферийных транзакций PTS	да
трехфазный генератор сигналов	да
16-разрядные таймеры	2
Блоки ШИМ	2
АЦП 10- или 8-разрядный/14 каналов	да
Количество команд	112

- Особенности архитектуры микросхемы с точки зрения применения в аппаратуре:
- быстродействующая архитектура типа «регистр-регистр»;
 - регистровое ОЗУ емкостью до 488 байт;
 - однократно программируемое ПЗУ емкостью 16 Кбайт;
 - динамически конфигурируемая разрядность шины данных - 8 или 16 бит;
 - протокол захвата шины HOLD/HLDA;
 - 12-канальная матрица процессоров событий (ЕРА);
 - сервер периферийных транзакций (PTS);
 - два 16-разрядных таймера/счетчика с предделителями и режимами квадратурного счета;
 - 16-разрядный сторожевой таймер;
 - 14 каналов аналого-цифрового преобразователя с разрешением в 8 или 10 бит;
 - трехфазный генератор формы сигналов с широтно-импульсной модуляцией ГФС;
 - генератор программируемой частоты;
 - двухканальный блок ШИМ;
 - пять 8-разрядных портов ввода-вывода, два 8-разрядных порта ввода и один 8-разрядный порт вывода;
 - режимы холостого хода IDLE и пониженного энергопотребления POWERDOWN;
 - режим электрической изоляции выводов микросхемы ONCE.

Центральное процессорное устройство ЦПУ

Микроконтроллер построен по Фон Неймановской архитектуре. Он имеет внешнюю системную магистраль для обмена данными с внешней памятью и дополнительными периферийными устройствами. Система команд микроконтроллера поддерживает широкий набор методов адресации, в т.ч. битовую адресацию.

Архитектура микроконтроллеров, называемая архитектурой типа «регистр-регистр», обеспечивает достижение высокой производительности и упрощает работу с периферией.

Главным отличительным признаком архитектуры микроконтроллеров является ядро, базирующееся на регистровом файле. Большое число универсальных легкодоступных регистров исключает «узкие места», свойственные архитектуре, использующей специальные регистры-аккумуляторы, и обеспечивает быстрое переключение контекста. ЦПУ микроконтроллера реализует инструкции с байтами, словами, несколько инструкций с 32-разрядными операндами, а также команды перехода по битам.

Микроконтроллер имеет регистровую организацию. Память данных обменивается данными с процессорным ядром по внутренней магистрали микроконтроллера. Поэтому обмен данными с внутренней памятью данных осуществляется быстрее, чем с внешней памятью.

ЦПУ осуществляет обмен информацией с внутренней и (или) внешней памятью программ и данных с помощью встроенного контроллера памяти, а с периферийными устройствами – через регистры специальных функций и имеет общее адресное пространство размером 64 Кбайт.

Система команд обеспечивает обработку данных длиной байт или 16-разрядное слово. Часть команд поддерживает инструкции над двойными словами, в том числе с такими же числами осуществляются и инструкции умножения и деления.

Встроенный контроллер памяти работает с 8- и 16-разрядной внешней шиной. Обеспечена возможность работы с внешней медленной памятью, с автоматическим формированием циклов ожидания.

Все внутренние регистры микроконтроллера могут выполнять функции «аккумулятора», что обеспечивает более эффективное программирование и повышение реальной производительности.

Несколько определенных ячеек памяти программ используются для хранения байта конфигурации кристалла (векторов подпрограмм обработки прерываний), а также содержат сигнатурный код доступа встроенного ПЗУ для считывания, что обеспечивает защиту программных продуктов от несанкционированного доступа.

Структура внешней шины программ/данных обеспечивает режим непосредственного доступа к памяти и многопроцессорный режим, а четырехбайтная очередь инструкций – оптимальный режим выборки команд без потери производительности.

Слово состояния программы содержит информацию о текущем состоянии работы. В нем имеется два байта: старший байт текущего слова состояния и младший байт масок прерывания.

Реализована развитая система прерываний. Система прерываний микроконтроллера поддерживается двумя устройствами: программируемым контроллером прерываний (IC) и сервером периферийных транзакций PTS, обслуживает 18 источников прерываний (16 уровней приоритета для IC и 15 уровней для PTS) и формирует 18 адрес-векторов подпрограмм обслуживания через IC и 15 – через PTS. Это существенно повышает эффективность применения микроконтроллера для управления реальными объектами.

Кроме того, микроконтроллер содержит встроенные периферийные блоки – многоканальный аналого-цифровой преобразователь, широтно-импульсные модуляторы, таймеры и т. п., которые могут функционировать автономно, практически не загружая центральный процессор. Характерные особенности контроллера шины – программное задание числа тактов ожидания, конфигурируемая шина данных (8- или 16-разрядная), а также протокол HOLD/HLDA для многопроцессорных систем.

На частоте 16 МГц ЦПУ выполняет два миллиона операций в секунду при выполнении элементарных инструкций над знаковыми/беззнаковыми данными длиной 1 или 2 байт. Для этих чисел имеются также и инструкции умножения и деления (быстродействие: 580 тысяч умножений и 330 тысяч делений в секунду).

Память и внешняя шина ЦПУ имеет одно адресное пространство размером 64 Кбайт, в котором находятся регистры общего назначения (488 байт), регистры специального назначения, встроенная программная память, внешняя память для программы и данных. IC имеет объем памяти 16 Кбайт и программную защиту от несанкционированного доступа.

Интерфейс и встроенные функциональные устройства

Микроконтроллер позволяет реализовать эффективное управление различного рода устройствами, в том числе и электродвигателями. В него встроены периферийные устройства, оптимизированные для управления трехфазными индукционными двигателями переменного тока, бесколлекторными трехфазными двигателями постоянного тока, четырехфазными шаговыми двигателями.

Периферийные устройства управляются и опрашиваются через специальные функциональные регистры SFR, которые могут быть доступны косвенно или окнами и как CPU-аккумуляторы.

Генератор формы сигналов ГФС

В микроконтроллере имеется уникальное периферийное устройство, называемое генератором формы сигналов ГФС (Waveform Generator – WFG), которое используется для формирования трехфазных широтно-модулированных сигналов.

Блок ГФС генерирует три неперекрывающиеся во времени комплементарные последовательности импульсов с управляемой длительностью (PWM сигналы) с разрешением в 125 нс (при запуске по фронту) или 250 нс (для «центрированной» оценки). Особенности генератора являются программируемые частота, время рабочего цикла и время блокировки. Для каждой фазы блок ГФС имеет по два выхода с повышенной нагрузочной способностью, при этом полярность выходных сигналов может программироваться. Схема защиты позволяет отключать все выходы ГФС одновременно в ответ на внешние события.

Каждый из выходов программируется независимо.

Блок ГФС позволяет обеспечить высокоэффективное управление трехфазными двигателями переменного тока, а также бесколлекторными двигателями постоянного тока.

Блок широтно-импульсных модуляторов ШИМ

Изделие содержит два аппаратно реализованных широтно-импульсных модулятора для которых используется общий программируемый источник опорной частоты, но длительность рабочего цикла для каждого выхода программируется независимо.

Встроенный ШИМ предназначен для генерации широтно-модулированного сигнала на выходе микросхемы без участия процессора.

На таймер подается системный синхросигнал (возможно через предделитель). В регистр периода процессор записывает число, соответствующее периоду ШИМ. При совпадении содержимого этого регистра и таймера, последний сбрасывается в нулевое состояние и на соответствующем выходе микросхемы формируется положительный перепад. В регистре длительности процессор записывает число, соответствующее длительности импульса. При совпадении его содержимого с содержимым таймера на выходе микросхемы формируется отрицательный перепад.

Система ввода-вывода

Микроконтроллер имеет 64 линии ввода-вывода, которые могут быть использованы встроенными периферийными устройствами.

Микроконтроллер имеет пять 8-разрядных портов ввода-вывода, два 8-разрядных порта ввода и один 8-разрядный порт вывода, часть из них имеет программно управляемые альтернативные функции для работы с встроенными устройствами.

Процессор событий EPA (event processor array)

Блок аналогичен ранее применявшимся устройствам HSIO. EPA имеет более простую архитектуру, чем HSIO, обладая при этом лучшей разрешающей способностью. В HSIO все входные каналы имеют общую память (семиуровневое FIFO), в которой запоминаются временные отметки, соответствующие событиям на входах. То же касается выходных линий HSIO – все они имеют общую память (восемь ячеек), в которую процессор записывает команды для всех выходных каналов HSIO. Поэтому за один такт HSIO может обработать только один входной и один выходной канал. В EPA каждый канал имеет свой собственный буфер, а выдача и прием сигналов производятся одновременно по всем каналам. Поэтому разрешающая способность EPA выше, чем у HSIO, в четыре раза. Кроме того, EPA – более гибкий функциональный блок: каждый его канал может служить и входом, и выходом, тогда как HSIO имеет четыре выходных,

два входных и два двунаправленных линии. ЕРА выполняет функции ввода-вывода, связанные с таймерами 1, 2.

ЕРА используется для работы с событиями. Матрица процессора событий состоит из шести модулей захвата/сравнения и шести модулей сравнения. Блок ЕРА имеет разрешение в 125 нс. Эти модули предназначены для быстрой генерации события на выходе микросхемы либо быстрой реакции на событие на её входе без участия процессора.

Таймер модуля считает импульсы, поступающие с постоянной частотой, соответствующей сигналу синхронизации (возможно деленной на предделителе). В регистр времени события предварительно записывается по команде определенное число. Компаратор сравнивает каждый момент времени содержимое таймера и содержимое регистра времени события. В момент, когда их содержимое становится равным друг другу, происходит заданное событие на соответствующем выходе микросхемы, и выставляется запрос на прерывание. Таким образом, можно запрограммировать определенное выходное событие в заданное время и осуществить его без участия процессора. Этот режим используется в частности для генерации ШИМ.

Периферийный сервер PTS

Этот функциональный блок предназначен для аппаратной обработки прерываний. Он содержит набор встроенных алгоритмов, исходные данные для которых должны быть размещены программой пользователя во встроенном ОЗУ кристалла. Алгоритмы PTS охватывают, в основном, пересылки данных. Прерывания, обслуживаемые PTS, обрабатываются быстрее, чем те, которые обслуживаются обычным способом.

Блок PTS предназначен для обеспечения механизма обработки событий в заданное время без участия процессора. Заданные события выполняются на микропрограммном уровне. Такими блоками событий могут быть:

- передача блока информации из одного места памяти (или устройства ввода-вывода) в другое;
- последовательный опрос нескольких каналов АЦП;
- передача информации по последовательному каналу связи.

Использование такого механизма обработки событий позволяет снизить загрузку процессора и распараллелить процесс обработки информации.

Сервер периферийных транзакций поддерживает обработку запросов на прерывание на микропрограммном уровне, не требуя вмешательства процессора. Специальный режим работы PTS обеспечивает поддержку функций последовательного ввода-вывода (SIO).

Таймеры-счетчики

Микроконтроллер имеет два независимых 16-разрядных многофункциональных таймера-счетчика. В одном из режимов используется внутренняя синхронизация (режим работы – таймер реального времени), в другом – внешняя (счетчик внешних событий). Содержимое таймеров может быть в любое время считано или программно модифицировано, а также сброшено программно или внешним сигналом. Таймер-счетчик внешних событий подсчитывает положительные или отрицательные фронты входных сигналов и может работать как в режиме прямого счета (инкремента), так и обратного (декремента).

Источником счетных импульсов может быть либо системный синхросигнал с фиксированной частотой (возможно предварительно деленной на заданное число раз), либо один из входов микроконтроллера. В первом случае устройство выполняет функции таймера, так как фактически считает интервалы времени постоянной длительности.

Во втором случае устройство выполняет функции счетчика событий (отрицательных или положительных перепадов) на входе микросхемы. Счет может производиться либо в одном, либо в обоих направлениях. В последнем случае направление счета определяется

уровнем сигнала на соответствующем входе микросхемы. При переходе содержимого счетчика из наибольшего состояния в наименьшее и наоборот, могут генерироваться соответствующие внутренние запросы на прерывания. Конкретные режимы работы и структура таймеров устанавливаются программно.

Сторожевой таймер (Watchdog timer – WDT)

Сторожевой таймер позволяет восстанавливать нормальную работу при сбоях программ. Если WDT разрешен, он будет вызывать аппаратный сброс, если программа не очищает его каждые 64К машинных цикла.

Когда таймер переполняется, он переводит сигнал RESET# в состояние низкого уровня не менее, чем на два машинных цикла, сбрасывая микроконтроллер и другие устройства, подключенные к его выводу RESET#.

Генератор программируемого тактового сигнала

Микроконтроллер 1874BE66T имеет возможность формировать выходной тактовый сигнал FREQOUT на выводе P7.7 с программируемой частотой от 4 кГц до 1 МГц со скважностью 50 %. Частота входного тактового сигнала при этом составляет 16 МГц.

Аналого-цифровой преобразователь (АЦП)

В составе микроконтроллера имеется встроенный 8-, 10-разрядный 14-входовый АЦП с диапазоном входного напряжения от $(U_{#0V} - 0,5) В$ до $(U_{NVCC} + 0,5) В$. Модуль АЦП предназначен для преобразования входной аналоговой информации в цифровую и передачи ее в процессор для дальнейшей обработки.

Время преобразования (измерения) составляет:

- 10,0 мкс – в режиме 10-разрядного преобразования;
- 7,0 мкс – в режиме 8-разрядного преобразования.

Схема выборки/хранения и отдельные входы опорного напряжения и аналоговой земли обеспечивают повышение точности преобразования.

Преобразование информации на том или ином канале инициализируется соответствующей командой.

Преобразование начинается выбором требуемого канала, который осуществляется записью номера канала в специальный служебный регистр. Аналоговый мультимплексор пропускает на выход сигнал выбранного канала. После этого подается сигнал фиксации на устройство выборки-хранения. Оно фиксирует уровень выбранного аналогового сигнала на весь период преобразования. Затем подается сигнал начала преобразования. АЦП представляет собой АЦП последовательных приближений. После окончания преобразования АЦП выдает сигнал готовности, по которому его выходная информация записывается в буферный регистр, с которого она затем может быть считана процессором по соответствующей команде. При этом вырабатывается запрос на прерывание.

Энергосберегающие режимы работы

В МК реализованы режимы работы с пониженным энергопотреблением:

- режим IDLE (HALT), когда работают только встроенные функциональные устройства, а микроконтроллер находится в режиме ожидания разрешенного прерывания от внешнего или внутреннего устройства;

- POWERDOWN (STOP), когда все устройства выключены до появления сигнала внешнего прерывания. В этом случае ток потребления – всего несколько микроампер.

Система команд

Система команд включает в себя полный набор арифметических, логических команд, а также инструкций управления и перехода над 8-битовыми операндами и 16-битными словами с различными способами адресации.

Система команд MCS-96 содержит следующие типы адресации: прямая регистровая (register - direct), косвенная (indirect), косвенная с автоувеличением (indirect with autoincrement), непосредственная (immediate), короткая индексная (short-indexed) и длинная индексная (long-indexed). Эти типы адресации увеличивают гибкость и скорость выполнения команд устройствами MCS-96. Каждая команда использует, по крайней мере, один из способов адресации.

Прямая регистровая адресация и непосредственная адресация выполняются наиболее быстро. Прямая регистровая адресация обеспечивает доступ к файлу регистров и SFR. Непосредственная адресация использует информацию, следующую за кодом команды, как операнд. Оба режима косвенной адресации используют значение слова в регистре как адрес операнда.

Длинная индексная адресация обеспечивает прямой доступ к любой ячейке 64К адресного пространства. Этот способ формирует адрес операнда добавлением 16-битного значения к регистровому слову. Индексирование с нулевым регистром позволяет иметь прямую адресацию к любой ячейке. Короткая индексная адресация формирует адрес операнда добавлением 8-битного значения к регистровому слову.

Косвенная адресация с автоинкрементом увеличивает адресное слово на единицу после операции с байтом, и на два после операции со словом. Этот способ адресации обеспечивает легкий доступ к справочным таблицам. Типы данных двойное слово (DOUBLE-WORD) и длинное целое (LONG) (32 битные) поддерживаются для результатов 16×16 умножения и для делимого при делении 32 на 16, а также для сдвиговых инструкций и 32-разрядном сравнении.

Кроме того имеются расширенные версии команд прерываемой и непрерываемой передачи блока. Множество способов адресации микроконтроллера 1874BE66T делает легким программирование на языке ассемблера и обеспечивает отличную взаимосвязь с языками высокого уровня. Команды ассемблера состоят из мнемоники, за которой следуют адрес или данные.

Остальные инструкции с 32-разрядными операндами могут быть выполнены комбинацией инструкций по 16 бит. Общее число команд – 112. Обеспечивается защита против невыполнимых кодов операций.

2.4 Электрические параметры микросхем

Электрические параметры микросхем при приёмке и поставке и предельно допустимые значения параметров приведены в таблицах 2.3 и 2.4, соответственно.

Номинальное значение напряжения питания микросхем 5,0 В. Допустимое отклонение напряжения питания $\pm 10\%$. Амплитуда пульсаций напряжения питания не более 50 мВ.

Микросхемы должны быть устойчивы к воздействию механических, климатических, биологических факторов и специальных сред, соответствующих группе унифицированного исполнения 4У по ГОСТ РВ 20.39.414.1-97 и таблицам 3, 4 ОСТ В 11 0998-99, в том числе:

- пониженной рабочей температуры среды – минус 60 °С;
- повышенной рабочей температуры среды – 85 °С;
- повышенной предельной температуры – 150 °С.

Т а б л и ц а 2.3 – Электрические параметры при приёмке и поставке

Наименование параметра, единица измерения, режим измерения	Буквенное обозначе- ние параметра	Норма параметра		Темпера- тура среды, °С
		не менее	не более	
1	2	3	4	5
1 Выходное напряжение высокого уровня по выводам P2.0 – P2.7, P3.0 – P3.7, P4.0 – P4.7, P5.0 – P5.7, P6.0 – P6.7, P7.0 – P7.7, CLKOUT, B, $U_{\#VCC} = 4,5 \text{ В}, U_{\text{NVCC}} = 4,0 \text{ В}, I_{\text{OH}} = -7,0 \text{ мА}$	U_{OH}	$U_{\#VCC}-1,5$	–	–60 ± 3 25 ± 10 85 ± 3
2 Выходное напряжение низкого уровня по выводам P2.0 – P2.7, P5.0 – P5.7, P6.6, P6.7, P7.0 – P7.7, CLKOUT, RESET#, B, $U_{\#VCC} = 4,5 \text{ В}, U_{\text{NVCC}} = 4,0 \text{ В}, I_{\text{OL}} = 3,2 \text{ мА}$	U_{OL1}	–	0,45	
3 Выходное напряжение низкого уровня по выводам портов P3, P4, B, $U_{\#VCC} = 4,5 \text{ В}, U_{\text{NVCC}} = 4,0 \text{ В}, I_{\text{OL}} = 15,0 \text{ мА}$	U_{OL2}	–	1,0	
4 Выходное напряжение низкого уровня по выводам P6.0 – P6.5, B, $U_{\#VCC} = 4,5 \text{ В}, U_{\text{NVCC}} = 4,0 \text{ В}, I_{\text{OL}} = 10,0 \text{ мА}$	U_{OL3}	–	0,45	
5 Выходной ток в состоянии «Выключено» по входам/выходам P2.0 – P2.7, P3.0 – P3.7, P4.0 – P4.7, P5.0 – P5.7, P7.0 – P7.7, мкА, $U_{\#VCC} = 5,5 \text{ В}, U_{\text{NVCC}} = 5,5 \text{ В},$ $0 < U_I < (U_{\#VCC} - 0,3) \text{ В}$	I_{OZ}	–1,0	1,0	
6 Токи утечки на входах P0.0 – P0.7, P1.0 – P1.4, мкА ¹⁾ $U_{\#VCC} = 5,5 \text{ В}, U_{\text{NVCC}} = 5,5 \text{ В}, 0 < U_I < U_{\text{NVCC}}$	I_{LI}	–1,0	1,0	
7 Динамический ток потребления по выводу #VCC, мА ¹⁾ , $U_{\text{PR}} = U_{\#VCC} = U_{\text{NVCC}} = 5,5 \text{ В}, f_{\text{CI}} = 16 \text{ МГц}$	I_{OCC1}	–	70	
8 Динамический ток потребления по выводу $\cap VCC$, мА, $U_{\text{PR}} = U_{\#VCC} = U_{\text{NVCC}} = 5,5 \text{ В}, f_{\text{CI}} = 16 \text{ МГц}$	I_{OCC2}	–	5	
9 Ток потребления в режиме ожидания, мА, $U_{\text{PR}} = U_{\#VCC} = U_{\text{NVCC}} = 5,5 \text{ В}, f_{\text{CI}} = 16 \text{ МГц}$	I_{CCS}	–	30	
10 Дифференциальная нелинейность, LSB ²⁾ , $U_{\#VCC} = 5,0 \text{ В}, U_{\text{NVCC}} = 5,12 \text{ В}, f_{\text{CI}} = 16 \text{ МГц}$	E_{LD}	–2,0	2,0	
11 Нелинейность относительно минимаксной прямой, LSB ²⁾ , $U_{\#VCC} = 5,0 \text{ В}, U_{\text{NVCC}} = 5,12 \text{ В}, f_{\text{CI}} = 16 \text{ МГц}$	E_{LBS}	–4,0	4,0	25 ± 10
12 Погрешность преобразования, LSB ²⁾ , $U_{\#VCC} = 5,0 \text{ В}, U_{\text{NVCC}} = 5,12 \text{ В}, f_{\text{CI}} = 16 \text{ МГц}$	E_{OK}	–4,0	4,0	
13 Смещение нуля, LSB ²⁾ , $U_{\#VCC} = 5,0 \text{ В}, U_{\text{NVCC}} = 5,12 \text{ В}, f_{\text{CI}} = 16 \text{ МГц}$	$U_{\text{Ю}}$	–1,5	1,5	
14 Функциональный контроль $U_{\#VCC} = (4,5; 5,5) \text{ В}, U_{\text{NVCC}} = (4,0; 5,5) \text{ В},$ $f_{\text{CI}} = (8; 16) \text{ МГц}$	ФК	–	–	–60 ± 3 25 ± 10 85 ± 3

Окончание таблицы 2.3

<p>Примечание – Параметры I_{OZ}, I_{LI} при температуре минус 60 °С не измеряются, а гарантируются нормой при температуре (25 ± 10) °С.</p> <p>¹⁾ Параметр проверяется при низком уровне сигнала на входе RESET#.</p> <p>²⁾ Параметры АЦП измеряются при температуре (25 ± 10) °С, т.к. в диапазоне температур недостаточна точность измерений, связанная со сложностью изготовления оснастки.</p>

Т а б л и ц а 2.4 – Значения предельно допустимых электрических режимов эксплуатации в диапазоне рабочих температур от минус 60 до 85 °С

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 ИС, В ¹⁾	$U_{\#VCC}$	4,5	5,5	-0,3	7,5
2 Напряжение питания аналоговой части ИС, В ¹⁾	$U_{\#VCC}$	4,0	5,5	-0,3	7,5
3 Напряжение программирования внутреннего EPROM, В	U_{PR}	12,25	12,75	-0,3	13,0
4 Входное напряжение низкого уровня, В ¹⁾	U_{IL}	-0,3	$0,3U_{\#VCC}$	-0,5	-
5 Входное напряжение высокого уровня, В ¹⁾	U_{IH}	$0,7U_{\#VCC}$	$U_{\#VCC}+0,3$	-	$U_{\#VCC}+0,5$
6 Выходной ток низкого уровня по выводам, мА ²⁾	P2.0 – P2.7, P5.0 – P5.7, P6.6, P6.7, P7.0 – P7.7, CLKOUT, RESET#	-	3,2	-	-
	P3.0 – P3.7, P4.0 – P4.7	-	15,0	-	-
	P6.0 – P6.5	-	10,0	-	-
7 Выходной ток высокого уровня, мА ³⁾	I_{OH}	-7,0	-	-	-
8 Емкость нагрузки, пФ	C_L	-	100	-	200
9 Частота следования импульсов тактового сигнала, МГц	f_{CI}	8	16	-	-
10 Длительность фронтов тактового сигнала, нс	t_{LH}	-	5	-	500
	t_{HL}	-	5	-	500
11 Диапазон входных напряжений АЦП, В	ΔU_I	-0,5	$U_{\#VCC}+0,5$	-0,5	$U_{\#VCC}+0,5$

¹⁾ Время действия предельных режимов – не более 10 мс.

²⁾ Значения суммарных предельных токов низкого уровня не должны превышать значений: для выводов P6.0 – P6.5: 40 мА; P3.0 – P3.7: 90 мА; P4.0 – P4.7: 90 мА; P5.0 – P5.7, CLKOUT: 35 мА; P2.0 – P2.7, P6.6, P6.7, P7.0 – P7.7: 63 мА.

³⁾ Значения суммарных предельных токов высокого уровня не должны превышать значений: для выводов P6.0 – P6.5: 28 мА; P3.0 – P3.7: 42 мА; P4.0 – P4.7: 42 мА; P5.0 – P5.7, CLKOUT: 35 мА; P2.0 – P2.7, P6.6, P6.7, P7.0 – P7.7: 63 мА.

3 Структура и организация ИС 1874ВЕ66Т

16-разрядный микроконтроллер 1874ВЕ66Т предназначен для выполнения вычислительных инструкций с повышенным быстродействием и высокоскоростных команд ввода-вывода. Он имеет общие архитектуру и набор команд с другими микросхемами серии 1874.

Микроконтроллер 1874ВЕ66Т разработан для использования в быстродействующих системах управления в режиме реального времени.

3.1 Ключевые особенности микроконтроллера

Структурная схема микроконтроллера 1874ВЕ66Т приведена на рисунке 2.1.

Микроконтроллер является полным функциональным аналогом изделий 87С196МD, реализующим базовую архитектуру MCS-196 фирмы Intel (США).

ИС 1874ВЕ66Т – это 16-разрядная быстродействующая микросхема высокой степени интеграции, ориентированная на решение задач управления процессами в реальном масштабе времени.

По общепринятому среди разработчиков аппаратуры мнению, однокристалльные микроконтроллеры семейства MCS-196 фирмы Intel являются своего рода индустриальным стандартом для 16-разрядных встроенных систем управления. Это объясняется с одной стороны сочетанием высоких технических показателей и экономической эффективности указанных семейств, а с другой – совершенством архитектуры, обеспечивающей эффективность и, частично, преемственность программных наработок при проектировании систем на их основе.

Архитектура микроконтроллера идеально ориентирована для создания модификаций систем для реализации функций управления и вычисления в реальном режиме времени под конкретные приложения. Этому способствует высокая производительность микроконтроллера, развитая система встроенных функциональных блоков, внутренняя программная память объемом 16 Кбайт и мощная система команд. МК Может служить элементной базой для систем управления различной аппаратурой, в том числе может применяться в спецаппаратуре, бытовой технике, теле-, аудио- и видеоаппаратуре, телефонной и радиосвязи, компьютерной технике, автомобильной и силовой электронике, особенно в части эффективного управления моторами и многих других отраслях.

Ключевые особенности микроконтроллеров семейства:

- трехфазный генератор сигналов широтно-импульсной модуляции;
- функционирование на частотах до 16 МГц;
- быстродействующая архитектура типа «регистр-регистр»;
- регистровое ОЗУ емкостью до 488 байт;
- однократно программируемое ПЗУ емкостью 16 Кбайт;
- динамически конфигурируемая разрядность шины данных – 8 или 16 бит;
- протокол захвата шины HOLD/HLDA;
- 8-канальная матрица процессоров событий;
- два 16-разрядных таймера/счетчика с предделителями и режимами квадратурного счета;
- 16-разрядный сторожевой таймер;
- до 14 каналов аналого-цифрового преобразователя с разрешением в 8 или 10 бит;
- восемь 8-разрядных портов ввода-вывода, ввода, вывода;
- режимы холостого хода (IDLE) и пониженного энергопотребления (POWERDOWN);
- сервер периферийных транзакций (PTS);
- двухканальный модуль сигналов ШИМ.

3.2 Системные ресурсы микроконтроллера

Микроконтроллер 1874BE66T обеспечен большим количеством мощных системных средств, расположенных вокруг ЦПУ.

Центральный процессор управляется контроллером микрокода, который выдает команды RALU на исполнение команд, используя байты, слова или двойные слова 256 байтами младшего файла регистров или через «окно», которое непосредственно имеет доступ к старшему файлу регистров. Команды центрального процессора сдвигаются из 4-байтовой очереди команд. Контроллер микрокода расшифровывает инструкции и затем выдает последовательность микрокоманд, управляющих выполнением заданной инструкции (подробно – в приложении А «Система команд»).

Файл регистров

Файл регистров разделен на старший и младший файл. В младшем файле регистров младшие 24 байта отведены регистрам специальных функций центрального процессора SFR и указателю стека, в то время как остальные доступны как регистры общего назначения РОН. Старший файл регистров содержит только регистры общего назначения. К ним можно обратиться как к байтам, словам или двойным словам. RALU доступны старшие и младшие файлы регистров отдельно. Младший файл регистров всегда доступен с прямой адресацией. Старший файл регистров доступен с прямой адресацией только при использовании режима «окна».

Регистровое арифметико-логическое устройство RALU

RALU содержит контроллер микрокода, арифметико-логическое устройство ALU, главный программный счетчик PC, слово состояния процессора PSW и служебные регистры. Регистры RALU: регистр инструкций, регистр констант, регистр выбора бит, счетчик циклов и три временных регистра (старшее слово, младшее слово и регистр второго операнда).

PSW содержит один бит PSW.1, который глобально разрешает или запрещает обслуживание всех маскируемых прерываний, один бит PSW.2, который разрешает или запрещает периферийный сервер PTS, и шесть булевых флагов, которые отражают состояние программы пользователя.

Все регистры, кроме 3-разрядного регистра выбора бита и 6-разрядного счетчика циклов, являются 16- или 17-разрядными (16 бит плюс расширение знака). Некоторые из этих регистров могут уменьшить загрузку ALU, выполняя простые действия.

RALU использует старшие и младшие регистры слов совместно для 32-разрядных инструкций и как временные регистры для большинства инструкций. Эти регистры имеют собственную логику сдвига и используются для операций, которые требуют логических сдвигов, включая нормализацию, умножение и деление. 6-разрядный счетчик циклов считает повторяющиеся сдвиги. Регистр второго операнда хранит второй операнд для инструкций с двумя операндами, включая множитель в течение умножения, и делитель в течение действия деления. При вычитании выходы этого регистра дополняются перед пересылкой в ALU.

RALU ускоряет вычисления, храня константы (например, 0, 1 и 2) в регистре констант так, чтобы они были доступны при дополнении, инкременте или декременте байт или слов. Кроме того, регистр констант генерирует одноразрядные маски на базе регистра битового выбора для инструкций проверки бит.

Выполнение инструкций

RALU исполняет большинство вычислений автономно, без использования аккумулятора. Вместо этого RALU работает непосредственно с младшим файлом регистров, которые фактически являются 256 «аккумуляторами». Поскольку данные не

проходят через единственный аккумулятор, команды микроконтроллера выполняются быстрее и более эффективно.

Формат команд

Микроконтроллеры сочетают большой набор регистров общего назначения с форматом трехоперандных инструкций. Этот формат позволяет в одной команде определить два регистра источника и отдельно регистр назначения. Например, следующая команда умножает две 16-битные переменные и сохраняет 32-битный результат в третьей переменной.

```
MUL RESULT, FACTOR_1, FACTOR_2; умножает FACTOR_1 и FACTOR_2
; и помещает в RESULT
; (RESULT) ← (FACTOR_1 × FACTOR_2)
```

Интерфейс памяти

РАЛУ связывается со всем объемом памяти, кроме регистрового файла и периферийных SFR, через контроллер памяти. (РАЛУ оперирует со старшим файлом регистров через контроллер памяти кроме случаев, когда используются «окна»; смотри раздел 5 «Распределение памяти»). Контроллер памяти содержит очередь команд, подчинённый программный счётчик, регистры адресов и данных и контроллер шины.

Контроллер шины управляет шиной памяти, которая состоит из внутренней шины памяти и внешней шины адреса/данных. Контроллер шины получает запросы доступа к памяти или от РАЛУ, или от очереди команд; запросы очереди всегда имеют приоритет. Эта очередь прозрачна к РАЛУ и программному обеспечению.

При использовании логического анализатора для отладки программы необходимо помнить, что инструкции предзагружены в очередь команд и не обязательно выполняются немедленно после того, как они загружены.

Обслуживание прерываний

Гибкая система обработки прерываний микроконтроллера имеет два главных компонента: программируемый контроллер прерываний IC и периферийный сервер PTS. Программируемый контроллер прерываний имеет аппаратную схему приоритетов PTS, которая может быть изменена программным обеспечением. Прерывания, которые выполняются контроллером прерывания, обслуживаются сервисными программами пользователя. Периферийный сервер транзакций PTS – микропрограммный аппаратный процессор, обеспечивающий быстрое обслуживание прерываний. Большинство прерываний (кроме NMI, TRAP и неисполняемого кода) можно запрограммировать на обслуживание их PTS вместо контроллера прерываний. PTS может передавать байты или слова индивидуально либо блоками между любыми ячейками памяти, управлять многократными аналого-цифровыми (АЦ) преобразованиями и генерировать ШИМ сигналы. Микроконтроллер 1874BE66T имеет дополнительные режимы, которые обеспечивают асинхронную или синхронную последовательную связь. PTS прерывания имеют более высокий приоритет, чем стандартные прерывания, и могут временно приостанавливать программы обслуживания прерываний.

3.3 Синхронизация

На схему синхроимпульсов (рисунок 3.1) поступает входной тактовый сигнал по выводу BQ1, задаваемый внешним кварцевым резонатором, и делится по частоте на два. Генератор синхроимпульсов принимает входную частоту от схемы «делитель на два» и формирует два неперекрывающихся внутренних сигнала PH1 и PH2.

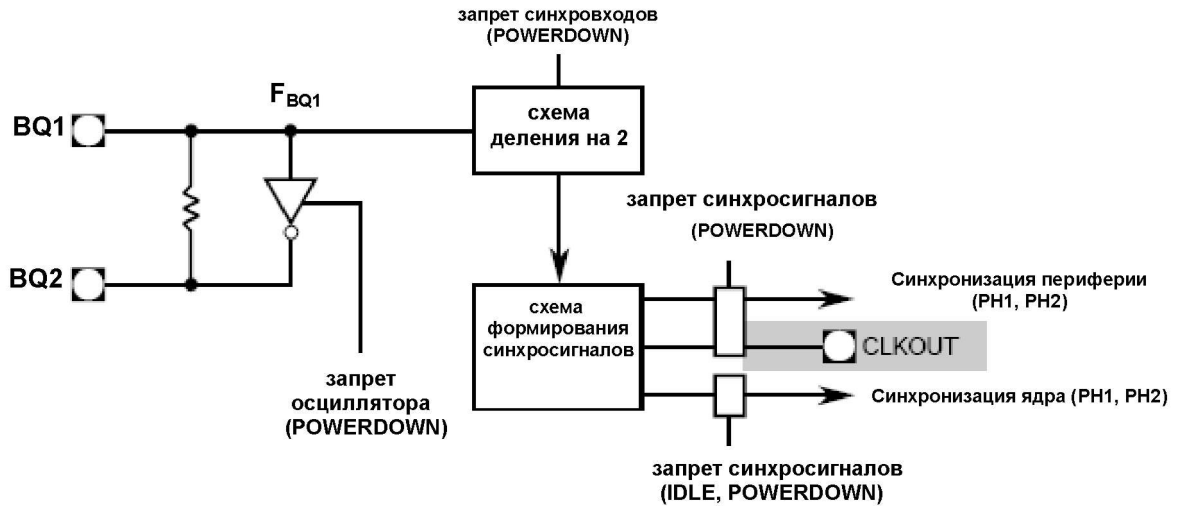


Рисунок 3.1 – Схема формирователя синхроимпульсов

Передние фронты PH1 и PH2 формируют внутренний CLKOUT сигнал. Схема синхроимпульсов отдельно формирует внутренние сигналы синхроимпульсов для центрального процессора и периферийных устройств, чтобы обеспечить гибкость в управлении мощностью потребления от источника питания.

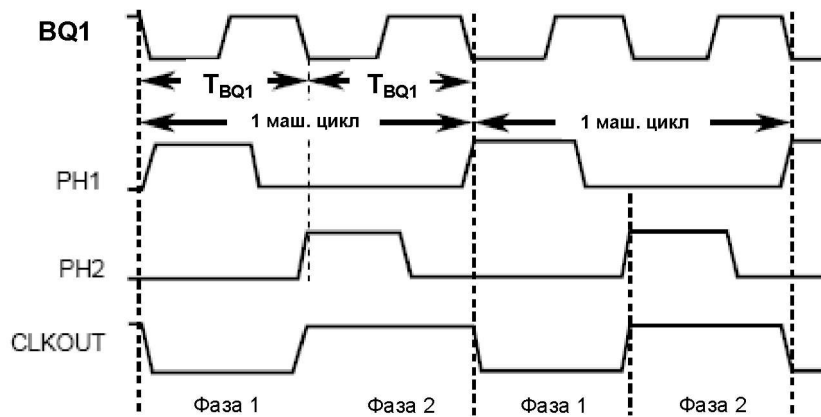


Рисунок 3.2 – Внутренние фазы синхроимпульсов

Комбинация периодов фазы 1 и фазы 2 внутреннего CLKOUT сигнала определяет основную единицу времени микроконтроллера, называемую «машинным циклом». В таблице 3.1 приведены параметры временной диаграммы (длительность машинного цикла при различной тактовой частоте, см. рисунок 3.2) на различных частотах.

Т а б л и ц а 3.1

F_{BQ1} (МГц)	Длительность машинного цикла (нс)
8	250
12	167
16	125

Следующие формулы позволяют вычислять частоты PH1 и PH2 – формула (3.1) (в МГц), длительность машинного цикла – формула (3.2) (в мкс), период синхроимпульсов (T_{BQ1}) – формула (3.3).

$$PH1 = \frac{F_{BQ1}}{2} = PH2, \quad (3.1)$$

$$\text{Длительность машинного цикла} = \frac{2}{F_{\text{BQ1}}}, \quad (3.2)$$

$$T_{\text{BQ1}} = \frac{1}{F_{\text{BQ1}}} \quad (3.3)$$

Поскольку микроконтроллер может работать на многих частотах, эти вычисления времени позволяют определять время выполнения инструкции в машинных циклах.

3.4 Встроенные периферийные устройства

Внутренние периферийные модули обеспечивают дополнительные функции для разнообразных специализированных применений. В данном подразделе приведен краткий обзор функциональности периферийных устройств, а в последующих разделах дается более подробное описание каждого устройства.

Порты ввода-вывода

Микроконтроллер 1874VE66T имеет восемь портов ввода-вывода (порты 0 – 7). Выводы портов мультиплексированы, чтобы служить стандартным вводом-выводом или обеспечивать специальные функции, связанные с периферийным устройством или компонентом системы. Если определённый сигнал специальной функции не используется, вывод может быть индивидуально сконфигурирован, чтобы служить «стандартным» для ввода-вывода. Порты 3 и 4 – исключения; они управляются на уровне порта, не на уровне вывода. Когда контроллер шины должен использовать шину адреса/данных, он осуществляет управление портами. Когда шина адреса/данных не используется, можно использовать порты для ввода-вывода. Подробно о сигналах – в приложении Б.

Порт 0 – порт только для ввода, который является также аналоговым входом для АЦП.

Порт 1 – порт только для ввода, который обслуживает аналоговые входы для АЦП.

Порт 2 – стандартный двунаправленный порт ввода-вывода, который обслуживает выходы ЕРА и таймеров.

Порты 3, 4 и 5 – распределенные в памяти, двунаправленные порты ввода-вывода. Порты 3 и 4 служат внешней шиной адреса/данных, в то время как порт 5 обеспечивает сигналы шинного управления.

Порт 6 – стандартный порт только для вывода, который обслуживает выходы блока ШИМ и генератора формы сигналов. В разделе 7 “Порты ввода-вывода” описываются порты ввода-вывода более подробно.

Порт 7 – стандартный двунаправленный порт, который обслуживает блок ЕРА и генератор программируемого тактового сигнала FREQOUT. Более подробно это описано в разделе 7.

Процессор событий ЕРА и таймеры/счетчики

Процессор событий выполняет быстродействующий ввод-вывод сигналов на базе таймеров/счетчиков. Во входном режиме ЕРА контролирует переключение входных сигналов. Когда происходит событие, ЕРА делает запись соответствующего значения из таймера. Это захват события. В режиме вывода ЕРА контролирует таймер до совпадения его значения с хранимым значением. При совпадении ЕРА запускает выходное событие, это может быть установка, очистка или переключение выходного сигнала. Это сравнение события. И захват, и сравнение событий могут вызвать прерывания, которые обслуживаются или контроллером прерываний, или периферийным сервером.

Таймер 1 и таймер 2 – это 16-разрядные реверсивные таймеры/счетчики, которые могут тактироваться внутренним или внешним сигналом. Каждый таймер/счетчик называют таймером, если такт внутренний, и счетчиком, если такт внешний.

Примечание – Дополнительная информация приведена в разделе 10 «Процессор событий ERA».

Блок ШИМ

Сигналы каждого ШИМ канала – импульсы с переменной скважностью. Некоторые типы двигателей требуют ШИМ сигналы для эффективной работы. После фильтра ШИМ сигнал формируется постоянным уровнем, длительность которого может меняться с дискретным шагом (256 в рабочем цикле). В ШИМ период также программируем (8 бит). Более подробно блок описан в разделе 9 «Широтно-импульсный модулятор».

Генератор формы сигнала ГФС

Генератор формы сигнала решает задачу формирования и синхронизации ШИМ сигналов. Генератор ГФС оптимизирован для применений в управлении трехфазными двигателями постоянного тока, трехфазными безколлекторными двигателями, четырехфазными шаговыми двигателями. ГФС может формировать три независимых пары ШИМ сигналов, которые имеют общий период, так называемое время «простоя» (dead time) и операционный режим. После программирования и инициализации генератор формы сигнала работает без вмешательства центрального процессора до тех пор, пока не программируется новая скважность (более подробно – в разделе 8 «Генератор формы сигнала»).

Генератор программируемого тактового сигнала

Микроконтроллер 1874BE66T имеет возможность формировать выходной тактовый сигнал FREQOUT на выводе P7.7 с программируемой частотой от 4 кГц до 1 МГц со скважностью 50 %. Частота входного тактового сигнала при этом составляет 16 МГц (раздел 11 «Генератор программируемого тактового сигнала»).

Аналого-цифровой преобразователь

Аналого-цифровой преобразователь АЦП преобразует аналоговое входное напряжение в цифровой эквивалент. Точность преобразования (8 или 10 битов), время выборки, время преобразования программируются. Преобразования могут быть выполнены с аналоговой землей и опорным напряжением, и результаты могут быть использованы, чтобы вычислить ошибки нулевого смещения и усиления. Внутренняя схема компенсации нулевого смещения позволяет автоматически регулировать нулевое смещение. АЦП также имеет режим порогового детектирования, который может использоваться, чтобы вызвать прерывания, когда программируемое пороговое напряжение превышено в любом направлении. Режим АЦ сканирования блоком PTS облегчает автоматизацию АЦ преобразования и сохранения его результатов (раздел 12 «Аналого-цифровой преобразователь АЦП»).

Сторожевой таймер

Сторожевой таймер – внутренний 16-разрядный таймер, который сбрасывает микроконтроллер, если программное обеспечение не в состоянии работать должным образом из-за сбоя или «зависания».

3.5 Специальные режимы работы

В дополнение к нормальному режиму включения, микроконтроллер работает в нескольких режимах специального назначения. Режимы холостого хода понижают мощность, когда микроконтроллер не активен. Режим внутрисхемной эмуляции ONCE электрически изолирует микроконтроллер от системы, а некоторые другие режимы обеспечивают варианты программирования для энергонезависимого запоминающего устройства.

Сокращение потребления мощности

В режиме холостого хода IDLE центральный процессор прекращает выполнять инструкции, но периферийные устройства остаются активными. Потребление мощности понижается приблизительно до 40 % нормального потребления. Аппаратный сброс или любое разрешенное прерывание выводят из режима холостого хода.

В режиме пониженного энергопотребления POWERDOWN все внутренние фазы установлены в логическом нуле и внутренний генератор отключен. Файл регистров и большинство периферийных устройств сохраняют их данные, если поддерживается $U_{\#VCC}$. Потребление мощности понижается до микроамперного диапазона. Более подробно это описано в разделе 13 «Специальные режимы работы».

Режим электрической изоляции ONCE

Режим внутрисхемной эмуляции ONCE электрически изолирует микроконтроллер от системы. Устанавливая режим ONCE, можно проверить схемы печатной платы, когда микроконтроллер впаян в плату. Более подробно это описано в разделе 13 «Специальные режимы работы».

Программирование внутренней памяти программ EPROM

Микроконтроллер, который имеет внутреннюю, однократно программируемую память программ EPROM, поддерживает несколько вариантов программирования.

Программирование в подчиненном режиме позволяет программатору стираемой программируемой постоянной памяти программировать и проверять один или более микроконтроллеров.

Автопрограммирование позволяет микроконтроллеру программировать себя кодом и данными, расположенными во внешнем запоминающем устройстве.

Программирование в ходе выполнения программы позволяет программировать отдельные ячейки EPROM в течение нормального выполнения кода под управлением программного обеспечения.

Режим чтения позволяет при необходимости сосчитать содержание энергонезависимого запоминающего устройства микроконтроллера.

Более подробно это описано в разделе 15 «Программирование постоянной памяти».

4 Система команд ИС 1874ВЕ66Т

В этом разделе приведено краткое описание системы команд микроконтроллера 1874ВЕ66Т и рекомендации по разработке программ (более подробно – в приложении А).

4.1 Общая характеристика системы команд

Система команд поддерживает различные типы операндов, необходимые для разработки программ в системах управления (таблица 4.1).

Типы переменных операндов указаны заглавными буквами во избежание путаницы. Например, BYTE – 8-битная беззнаковая переменная в инструкции, в то время как byte – любая 8-битная единица данных (со знаком или без знака).

Т а б л и ц а 4.1 – Определения типа операнда

Тип операнда	Число бит	Знак	Возможные значения	Ограничения адресации
BIT	1	нет	TRUE (1) или FALSE (0)	как компоненты байтов
BYTE	8	нет	от 0 до (2^8-1) (от 0 до 255)	нет
SHORT INTEGER	8	да	от (-2^7) до $(+2^7-1)$ от (-128) до (+127)	нет
WORD	16	нет	от 0 до $(2^{16}-1)$ (от 0 до 65 535)	адрес четного байта
INTEGER	16	да	от (-2^{15}) до $(+2^{15}-1)$ (от -32 768 до +32 767)	адрес четного байта
DOUBLE WORD ¹⁾	32	нет	от 0 до $(2^{32}-1)$ (от 0 до 4 294 967 295)	адрес в младшем регистровом файле, кратный четырем ²⁾
LONG INTEGER ¹⁾	32	да	от (-2^{31}) до $(+2^{31}-1)$ (от -2 147 483 648 до +2 147 483 647)	адрес в младшем регистровом файле, кратный четырем ²⁾

¹⁾ 32-битные переменные поддерживаются только как операнд в инструкциях сдвига, как делимое при инструкции деления 32 на 16 и как результат действия умножения 16 на 16.

²⁾ Для совместимости с программными средствами сторонних производителей, необходимо придерживаться правил, принятых в программировании на «Си» для адресации 32-битных операндов.

Т а б л и ц а 4.2 – Эквивалентные типы операндов для ассемблера и «Си»

Тип операнда	Эквивалент ассемблера	Эквивалент на языке «Си»
BYTE	BYTE	unsigned char
SHORT INTEGER	BYTE	char
WORD	WORD	unsigned int
INTEGER	WORD	int
DOUBLE WORD	LONG	unsigned long
LONG INTEGER	LONG	long

Операнды BIT

BIT – одноразрядная переменная, которая может иметь булевы значения "true" и "false". Архитектура требует, чтобы к BIT обращались как к компонентам BYTE или WORD, так как прямая адресация BIT не поддерживается.

Операнды BYTE

BYTE – 8-битная переменная без знака, которая может иметь значения от 0 до 255 (28–1). Арифметические и сравнение операторы могут быть применены к операндам BYTE, но результат должен интерпретироваться по модулю 256 арифметически. Логические действия с BYTE осуществляются побитно. Биты в пределах BYTE помечены от 0 до 7, бит 0 – младший бит. Нет никаких ограничений выравнивания для BYTE, так что они могут быть помещены по любому адресу памяти.

Операнды SHORT INTEGER

SHORT INTEGER – 8-битная переменная со знаком, которая может принимать значения от минус 128 (–27) до плюс 127 (27–1). Арифметические действия, которые производят результаты вне диапазона SHORT INTEGER, устанавливают флаги переполнения в слове состояния процессора (PSW). Числовой результат тот же самый, что и результат эквивалентного действия с переменной BYTE. Нет никаких ограничений выравнивания для SHORT INTEGER, так что они могут быть помещены по любому адресу памяти.

Операнды WORD

WORD – 16-битная беззнаковая переменная, которая может иметь значения от 0 до 6535 (216–1). Арифметические и сравнение операторы могут быть применены к операндам WORD, но результат должен интерпретироваться по модулю 6536 арифметически. Логические действия с WORD осуществляются побитно. Биты в пределах WORD помечены от 0 до 15, бит 0 – младшего значения бит.

WORD должны быть выровнены по границе четного байта адреса пространства. Младший байт WORD находится по четному адресу байта, а старший байт находится в следующем (нечетном) адресе. Адрес WORD – адрес младшего байта. Действия WORD по нечетным адресам не гарантируются.

Операнды INTEGER

INTEGER – 16-битная переменная со знаком, которая может принимать значения от минус 32 768 (–215) до плюс 32 767 (+215–1). Арифметические действия, которые производят результаты вне диапазона INTEGER, устанавливают флаги переполнения в слове состояния процессора (PSW). Числовой результат тот же самый, как результат эквивалентного действия на переменных WORD.

INTEGER должен быть выровнен по границе четного байта в адресном пространстве. Младший байт INTEGER находится по четному адресу байта, а старший байт находится в следующем выше нечетном адресе. Адрес INTEGER – адрес его младшего байта. Действия INTEGER по нечетным адресам не гарантируются.

Операнды DOUBLE WORD

DOUBLE WORD – 32-битная переменная без знака, которая может принимать значения от 0 до 4 294 967 295 (232–1). Архитектура непосредственно поддерживает операнды DOUBLE WORD только как операнды в командах сдвига, делимого при делении 32/16 и произведения при умножении 16×16. Для этих действий переменная DOUBLE WORD должна находиться в младшем файле регистров и должна быть выровнена по адресу, который является кратным четырем. Адрес DOUBLE WORD – это адрес младшего байта (четный адрес байта). Младшее слово DOUBLE WORD находится всегда в младшем адресе, даже когда данные находятся в стеке. Это означает, что старшее слово должно быть выдвинуто в стек первым.

Действия DOUBLE WORD, которые непосредственно не поддерживаются, могут быть легко осуществлены с двумя операндами WORD. Например, следующие

последовательности инструкций с 16 битами исполняют 32-битное сложение и 32-битное вычитание соответственно.

ADD REG1, REG3; (двухоперандное сложение)
ADDC REG2, REG4
SUB REG1, REG3; (двухоперандное вычитание)
SUBC REG2, REG4

Операнды LONG INTEGER

LONG INTEGER – 32-битная переменная со знаком, которая может принимать значения от минус 2 147 483 648 (–231) до плюс 2 147 483 647 (231–1). Архитектура непосредственно поддерживает операнды LONG INTEGER только как операнды в командах сдвига, как делимое при делении 32/16 и как произведение при умножении 16×16. Для этих действий переменная LONG INTEGER должна быть в младшем файле регистров и должна быть выровнена по адресу, который является кратным четырем. Адрес LONG INTEGER – это его младший байт (адрес четного байта).

Инструкции с LONG INTEGER, которые непосредственно не поддерживаются, могут быть легко осуществлены с двумя операндами INTEGER. Смотри пример в «Операнды DOUBLE WORD».

Преобразование операндов

Система команд поддерживает преобразования между разными типами операндов. LDBZE (load byte, zero extended), конвертирует BYTE в WORD. CLR (очистка) конвертирует WORD в DOUBLE WORD, очищая (записывая ноль) в старшее слово в DOUBLE WORD. LDBSE (load byte, sign extended) конвертирует SHORT INTEGER в INTEGER. EXT (расширение знака) конвертирует INTEGER в LONG INTEGER.

Условные переходы

Команды для сложения, вычитания и сравнения не различают операнды без знака (BYTE, WORD) и со знаком (SHORT INTEGER, INTEGER). Однако условные команды перехода позволяют рассматривать результаты этих действий как числа со знаком или без знака. Например, команда CMP (сравнение) используется, чтобы сравнить 16-битные числа со знаком или без знака. После инструкции сравнения можно использовать JN (переход, если больше) команду для операндов без знака или JGT (переход, если больше чем) команда для операндов со знаком.

Инструкции с плавающей точкой

Аппаратные средства непосредственно не поддерживают действия с переменными REAL (плавающей точкой). Эти действия поддерживаются только с использованием библиотеки с плавающей точкой. Производительность этих инструкций значительно повышена при использовании NORML инструкции и бита ST в слове состояния процессора (PSW). Инструкция NORML нормализует 32-битную переменную. Флаг ST может использоваться в комбинации с флагом переноса C, чтобы достигнуть более высокой точности при округлении.

4.2 Режимы адресации

Система команд использует четыре основных режима адресации:

- прямой;
- непосредственный;
- косвенный (с или без автоинкремента);
- индексный (короткий, длинный или с нулевым индексом).

Указатель стека может использоваться с косвенной адресацией, чтобы получить доступ к вершине стека, и может также использоваться с коротко индексированной

адресацией к данным в пределах стека. Нулевой регистр может использоваться с длинной индексной адресацией для доступа к любой ячейке запоминающего устройства.

Инструкция может содержать только одну косвенную или индексную ссылку; остальные операнды должны быть прямо адресованы.

Т а б л и ц а 4.3 – Описание временных регистров

Рабочий регистр	Описание
AX	выровненный словом 16 битный регистр; AH – старший байт AX и AL – младший байт
BX	выровненный словом 16 битный регистр; BH – старший байт BX и BL – младший байт
CX	выровненный словом 16 битный регистр; CH – старший байт CX и CL – младший байт
DX	выровненный словом 16 битный регистр; DH – старший байт DX и DL – младший байт

Прямая адресация

Прямая адресация непосредственно обеспечивает доступ к ячейкам (256 байтов) в младшем файле регистров, не загружая контроллер памяти. «Окна» позволяют перераспределять другие секции запоминающего устройства в младший файл регистров для прямого доступа. Нужно определять регистры как операнды в пределах инструкции. Адрес регистра должен соответствовать правилам выравнивания для типа операнда. В вычислении может принять участие до трех регистров в зависимости от инструкции. Следующие инструкции используют прямую адресацию:

```
ADD      AX, BX, CX ; AX ← BX + CX
ADDB    AL, BL, CL ; AL ← BL + CL
MULB    AX, BL     ; AX ← AX × BL
INCB    CL         ; CL ← CL + 1
```

Непосредственная адресация

Непосредственный режим адресации определяет одну непосредственную величину как операнд в инструкции. Непосредственное значение помечается символом «#». Инструкция может содержать только одну непосредственную величину. Остальные операнды должны быть прямо адресованы. Следующие инструкции используют непосредственную адресацию:

```
ADD AX, #340; AX ← AX + 340
PUSH #1234H ; SP ← SP – 2 ; MEM_WORD(SP) ← 1234H
DIBV AX, #10 ; AL ← AX/10 ; AH ← AX MOD 10
```

Косвенная адресация

Косвенный режим адресации обеспечивает доступ к операнду, получая его адрес из регистра WORD в младшем файле регистров. Регистр, содержащий косвенный адрес, помечается квадратными скобками []. Косвенный адрес может обратиться к любой ячейке в пределах памяти, включая файл регистров. Регистр, который содержит косвенный адрес, должен быть выровнен словом, и косвенный адрес должен соответствовать правилам для типа операнда. Инструкция может содержать только одну косвенную ссылку, остальные операнды должны быть прямо адресованы. Следующие инструкции используют косвенную адресацию:

```
LD      AX, [BX] ; AX ← MEM_WORD(BX)
ADDB AL, BL, [CX]; AL ← BL + MEM_BYTE(CX)
POP     [AX] ; MEM_WORD(AX) ← MEM_WORD(SP); SP ← SP + 2
```

Косвенная адресация с автоинкрементом

Можно автоматически увеличивать косвенный адрес после текущего доступа к памяти. При определении автоинкремента в конце косвенной ссылки добавляется символ плюс «+». В этом случае инструкция автоматически увеличивает косвенный адрес (на единицу, если регистр результата 8-битный или на два, если это 16-битный регистр). Следующие инструкции используют косвенную адресацию с автоинкрементом:

```
LD    AX, [BX]+      ; AX ← MEM_WORD(BX); BX ← BX + 2
ADDB  AL, BL, [CX]+  ; AL ← BL + MEM_BYTE(CX); CX ← CX + 1
PUSH  [AX] + ; SP ← SP - 2; MEM_WORD(SP) ← MEM_WORD(AX); AX ← AX + 2
```

Косвенная адресация с указателем стека

Можно также использовать косвенную адресацию, чтобы получить доступ к вершине стека, используя указатель стека как регистр WORD в косвенной ссылке. Следующая инструкция использует косвенную адресацию с указателем стека:

```
PUSH [SP] ; duplicate top of stack; SP ← SP + 2
```

Индексная адресация

Индексная адресация вычисляет адрес, добавляя поле смещения к основному адресу. Есть три разновидности индексной адресации: короткая индексная, длинная индексная и адресация с нулевым индексом. И короткая, и длинная индексные адресации используются, чтобы получить доступ к определенному элементу в пределах структуры. Короткая индексная адресация может получить доступ к 255 байтам, длинная индексная адресация может получить доступ к 65 535 байтам, адресация с нулевым индексом может получить доступ к одному байту. Инструкция может содержать только одну индексную ссылку, остальные операнды должны быть прямо адресованы.

Короткая индексная адресация

В короткой индексной инструкции смещение определяется как 8-битная константа, и основной адрес – как косвенный адрес регистра (WORD). Следующие инструкции используют короткую индексную адресацию.

```
LD AX, 12H [BX]; AX ← MEM_WORD (BX + 12H)
MULB AX, BL, 3 [CX]; AX ← BL × MEM_BYTE (CX + 3)
```

Инструкция LD AX, 12H [BX] загружает AX содержимым ячейки по адресу BX+12H. То есть инструкция добавляет BX+12H к содержимому [BX], затем загружает AX содержимым. Например, если [BX] содержит 1000H, то AX загружается содержимым ячейки 1012H. Короткая индексная адресация обычно используется для доступа, когда [BX] содержит базовый адрес и константа (12H в этом примере) – смещение определенного элемента в структуре.

Можно также использовать указатель стека в короткой индексной инструкции и получить доступ к ячейке в пределах стека, как показано в следующей инструкции:

```
LD AX, 2 [SP]
```

Длинная индексная адресация

В длинной индексной инструкции основной адрес определяется как 16-битная переменная и смещение как косвенный адрес регистра (WORD). Следующие инструкции используют длинную индексную адресацию.

```
LD AX, TABLE [BX]; AX ← MEM_WORD (TABLE + BX)
AND AX, BX, TABLE [CX]; AX ← BX AND MEM_WORD (TABLE + CX)
ST AX, TABLE [BX]; MEM_WORD (TABLE + BX) ← AX
ADDB AL, LOOKUP [CX]; AL ← BL + MEM_BYTE (LOOKUP + CX)
```

Инструкция LD AX, TABLE [BX] загружает AX содержимым ячейки памяти по адресу TABLE+BX. То есть инструкция добавляет содержание BX к константе TABLE

(основной адрес), затем загружает AX содержанием результирующего адреса. Например, если TABLE равняется 4000H и BX содержит 12H, то AX загружается содержимым ячейки 4012H. Длинная индексная адресация обычно используется для обращения к элементам в таблице, где константа TABLE является основным адресом структуры, и BX – масштабированное смещение ($n \times$ размер элемента, в байтах) в структуре.

Адресация с нулевым адресом

В команде с нулевым индексом определяется адрес как 16-битная переменная, смещение – ноль, и можно записать это одним из трех способов: [0], [ZERO_REG] или без указания. Каждая из следующих инструкций загружает AX содержимым переменной THISVAR.

```
LD AX, THISVAR [0]
LD AX, THISVAR [ZERO_REG]
LD AX, THISVAR
```

Следующие инструкции также используют адресацию с нулевым индексом:
ADD AX, 1234H [ZERO_REG]; AX ← AX + MEM_WORD (1234H)
POP 5678H [ZERO_REG]; MEM_WORD (5678H) ← MEM_WORD (SP); SP ← SP + 2

4.3 Выборы режима адресации ассемблера

Ассемблер упрощает выбор режимов адресации. Рекомендуется использовать это везде, где возможно.

Прямая адресация

Ассемблер выбирает между прямой адресацией и адресацией с нулевым индексом в зависимости от местоположения операнда в памяти. Необходимо обратиться к операнду его символическим именем. Если операнд находится в младшем файле регистров, ассемблер выбирает прямую адресацию. Если операнд находится в другой области памяти, ассемблер выбирает адресацию с нулевым индексом.

Индексная адресация

Ассемблер выбирает между короткой индексной и длинной индексной адресацией, в зависимости от значения индекса. Если значение может быть выражено в восьми битах, язык ассемблера выбирает короткую индексную адресацию. Если значение больше 8-битного, выбирается длинная индексная адресация.

4.4 Стандарты и соглашения программного обеспечения

При разработке программного обеспечения рекомендуется чтобы использовались соглашения, принятые для редактирования на основе языка «Си». Эти стандарты применимы и для ассемблера, и для программной среды «Си», и они обеспечивают совместимость между этими средами.

Использование регистров

В 256-байтном младшем файле регистров содержатся регистры специальных функций центрального процессора и указатель стека. Остаток младшего файла регистров и старший файл регистров доступны для использования. Периферийные регистры специальных функций (SFR) и распределенные в карте памяти SFR расположены в верхней области адресного пространства. Периферийные SFR могут быть перемещены через «окна» в младший файл регистров для прямого доступа. Распределенные в карте памяти SFR не могут быть использованы в режиме «окон». Можно использовать косвенную или индексную адресацию, чтобы получить доступ к ним. Все SFR оперируют как BYTE или WORD, если не определено иначе.

Язык программирования «Си» адаптирован к простой эффективной стратегии, размещает восемь байтов, начинающихся в адресе $1C_H$, во временное хранение и рассматривает оставшуюся область в файле регистров как сегмент запоминающего устройства, размещённый там, где необходимо.

Примечание – Использование любого SFR как базового или индексного регистра для косвенных или индексных инструкций может вызвать непредсказуемые результаты, потому что внешние события могут изменить содержимое SFR. Так как некоторые SFR очищаются при чтении, допустимо использование SFR как операнда в инструкциях «чтение-модификация-запись» (например, XORB).

Адресация 32-битных операндов

32-битные операнды (DOUBLE WORD и LONG INTEGER) сформированы двумя смежными 16-битными словами в запоминающем устройстве. Младшее слово DOUBLE WORD находится всегда в младшем адресе, даже когда данные находятся в стеке (это означает, что старшее слово должно быть выдвинуто в стек первым). Адрес 32-битного операнда – адрес его младшего байта.

Аппаратные средства поддерживают 32-разрядные типы данных как операнды в инструкциях сдвига, делимое в делении 32/16 и произведение в умножении 16×16 . Для этих действий 32-битный операнд должен располагаться в младшем файле регистров и должен быть выровнен по адресу, кратному четырем.

Соединение подпроцедур

Параметры передаются к подпроцедурам через стек. Параметры выдвигаются в стек от самого правого параметра налево. Параметры на 8 битов выдвигаются в стек со старшим неопределённым байтом. 32-битные параметры выдвигаются в стек как два 16-битных; старшая часть параметра выдвигается в стек первой. Как пример рассмотрим следующую процедуру:

```
Void example_procedure (char param1, long param2, int param3)
```

Когда эта процедура выполняется, стек будет содержать параметры в следующем порядке:

```
param3  
low word of param2  
high word of param2  
undefined; param1  
return address ← Stack Pointer
```

Если процедура возвращает значение к коду запроса (в противоположность изменению более глобальных переменных), результат возвращается в область временного хранения (TMPREG0 в этом примере) начинающуюся в $1C_H$. TMPREG0 рассматривается или как 8-, 16- или 32-битная переменная, в зависимости от типа процедуры.

Стандарт вызова, принятый языком «Си», имеет несколько следующих ключевых особенностей.

Процедуры могут всегда предполагать, что восемь байтов файла регистров, начинающиеся в $1C_H$, могут использоваться для временного хранения в пределах тела процедуры.

Код, который вызывает процедуру, должен учитывать, что процедура изменяет восемь байтов файла регистров, начинающиеся в $1C_H$.

Код, который вызывает процедуру, должен предполагать, что процедура изменяет слово состояния процессора PSW, потому что процедуры не сохраняют и не восстанавливают PSW. Результат процедур всегда возвращается в переменную TMPREG0.

Язык «Си» позволяет определять прерывание процедур, которые выполняются. Прерывание процедур не соответствует правилам нормальных процедур. Параметры нельзя передать к этим процедурам, и они не могут вернуть результаты. Так как

прерывание процедуры можно выполнить по существу в любое время, они должны сохранить и восстанавливать значения PSW и TMPREG0.

4.5 Защита и руководящие принципы программного обеспечения

В микроконтроллере реализовано несколько механизмов восстановления после ошибок программного обеспечения и аппаратных средств. Прерывание по невыполнимому коду инструкции обеспечивает защиту от выполнения некорректного кода инструкции. Команда аппаратного сброса RST может вызвать сброс, если программный счетчик выходит за границы. Код инструкции RST – FF_H, поэтому процессор сбросит себя, если попытается выполнить инструкцию от незапрограммированных ячеек в энергонезависимом запоминающем устройстве или из шины, линии которой подключены к высокому уровню. Сторожевой таймер WDT может также сбросить микроконтроллер в случае аппаратной или программной ошибки.

Рекомендуется заполнить неиспользованные области с кодом NOP и периодическими переходами к процедуре обслуживания ошибок или RST инструкции. Это особенно важно для кодов окружения таблиц поиска. Везде, где место позволяет, необходимо окружить каждую таблицу семью NOP (потому что самая длинная инструкция устройства имеет семь байтов) и RST или переходами к процедуре обслуживания ошибок. Так как RST – однобайтовая инструкция, NOP не нужны, если RST используются вместо переходов к процедуре ошибки. Это гарантирует быстрое восстановление от ошибки программного обеспечения.

При использовании сторожевого таймера WDT для защиты программного обеспечения рекомендуется сбрасывать WDT только в одном месте программы, сокращая возможность нежелательного сброса WDT. Секция кода, который сбрасывает WDT, должна контролировать другие секции кода для правильного выполнения инструкций. Это может быть сделано проверкой переменных, чтобы удостовериться, что они – в пределах разумных значений. Использование программного таймера, чтобы сбросить WDT каждые 10 мс, обеспечивает защиту только от катастрофических ошибок.

5 Организация памяти

В разделе описывается адресное пространство, его основные разделы, методика работы с окнами, а также порядок обращения к старшему файлу регистров и периферийным SFR с использованием команд с прямым методом адресации.

5.1 Карта памяти микроконтроллеров

В таблице 5.1 приведена карта памяти ИС 1874ВЕ66Т.

Внешние устройства (память или устройства ввода-вывода)

Несколько областей памяти отведены для внешних устройств (таблица 5.1). Данные могут быть сохранены в любой части этой памяти. Эти области могут также использоваться для связи с внешней периферией, подключенной к шине адреса/данных.

Т а б л и ц а 5.1 – Карта памяти микроконтроллера 1874ВЕ66Т

Шестнадцатеричный диапазон адресов	Описание	Способы адресации
FFFF 6000	Внешнее устройство (память или устройство ввода-вывода), подключенное к шине адреса/данных	Косвенный или индексный
5FFF 2080	Память программ (внутренняя энергонезависимая или внешняя память) ¹⁾	Косвенный или индексный
207F 2000	Память специального назначения (внутренняя энергонезависимая или внешняя память)	Косвенный или индексный
1FFF 1FE0	Картированные в памяти SFR	Косвенный или индексный
1FDF 1F00	Периферийные SFR	Косвенный, индексный или прямой через окно
1EFF 0200	Внешние устройства (память или ввод-вывод), подключенные к шине адреса/данных (будущее SFR расширение) ²⁾	Косвенный или индексный
01FF 0100	Старший файл регистров (оперативная память регистров общего назначения)	Косвенный, индексный или прямой через окно
00FF 0000	Младший регистровый файл (оперативная память регистров общего назначения, указатель стека и центрального процессора SFR)	Прямой, косвенный или индексный

¹⁾ После сброса ИС 1874ВЕ66Т выбирает первую команду из ячейки 2080_H.
²⁾ Содержимое или функции этих ячеек могут измениться в будущих версиях микроконтроллера.

Программная память и память специального назначения

Внутренняя энергонезависимая, однократно программируемая память – дополнительный компонент микроконтроллера 1874ВЕ66Т (память EPROM). Энергонезависимая память разделяется на память специального назначения и память программ (ячейки 2000_H и выше; таблица 5.1). Сигнал EA# управляет доступом к этой области памяти. Доступ осуществляется к внутренней памяти, если сигнал EA# поддерживается в высоком уровне, и к внешней памяти, если сигнал EA# поддерживается

в низком уровне. Захват сигнала EA# осуществляется при сбросе микроконтроллера при переходе сигнала RESET# из состояния низкого уровня в высокий.

Память программ

Память программ расположена в области, начинающейся с адреса 2080_H. Эта область памяти доступна для хранения выполняемых кодов и данных. Высокий уровень сигнала EA# служит для разрешения доступа к памяти программ.

Примечание – Рекомендуется запись кода FF_H (код инструкции для команды RST) в используемые ячейки памяти программ. Это вызовет сброс устройства, если программа начинает выполнять коды из неиспользованной памяти.

Память специального назначения

Память специального назначения находится по адресам 2000_H–207F_H (таблица 5.2). Она содержит несколько зарезервированных ячеек памяти, байты конфигурации (CCB0, CCB1) и векторы для периферийного сервера (PTS) и стандартных прерываний. Эта область адресов доступна во внутренней памяти ИС 1874BE66T, если сигнал EA# установлен в высокий уровень, и во внешней памяти, если сигнал EA# установлен в низкий уровень.

Т а б л и ц а 5.2 – Адреса памяти специального назначения

Адрес (шестнадцатеричный код)	Описание
207F – 205E	Зарезервированы (каждый байт должен содержать FF _H)
205D – 2040	Векторы PTS
203F – 2030	Старшие векторы прерывания
202F – 2020	Ключ защиты
201F	Зарезервирован (должен содержать 20 _H)
201E	Зарезервирован (должен содержать FF _H)
201D	Зарезервирован (должен содержать 20 _H)
201C	Зарезервирован (должен содержать FF _H)
201B	Зарезервирован (должен содержать 20 _H)
201A	CCB1 (байт конфигурации 1)
2019	Зарезервирован (должен содержать 20 _H)
2018	CCB0 (байт конфигурации 0)
2017 – 2014	Зарезервированы (каждый байт должен содержать FF _H)
2013 – 2000	Младшие векторы прерывания

Резервные ячейки памяти

Несколько ячеек памяти зарезервировано. Не рекомендуется считывать или записывать эти ячейки, за исключением инициализации. Инициализировать резервные ячейки следует устанавливая значения, приведенные в таблице 5.2.

Прерывания и векторы PTS

Старшие и младшие векторы прерываний содержат адреса подпрограмм обслуживания прерываний. Векторы периферийного сервера PTS содержат адреса блоков управления PTS.

Ключ защиты

Ключ защиты предотвращает несанкционированное программирование энергонезависимой памяти (раздел 15 «Программирование постоянной памяти»).

Байты конфигурации (ССВ0, ССВ1)

Байты конфигурации (ССВ0, ССВ1) определяют режим работы микроконтроллера 1874ВЕ66Т. Они задают ширину шины, режим управления шиной и состояния ожидания. Они также управляют режимом пониженного энергопотребления, сторожевым таймером и защитой энергонезависимой памяти.

ССВ – первые байты, записываемые из памяти, когда устройство выходит из состояния сброса. После сброса последовательность ССВ загружается в регистры конфигурации ССР. После записи содержимое ССР не может быть изменено до следующего сброса устройства. Как правило, ССВ программируются один раз, когда пользовательская программа компилируется и не переопределяются в течение нормального режима работы.

Для устройств с программируемой пользователем энергонезависимой памятью ССВ загружены для нормальных операций, но РССВ загружаются в ССР, если ИС 1874ВЕ66Т входит в режимы программирования (раздел 15 «Программирование постоянной памяти»).

Специальные функциональные регистры SFR

Эти регистры или картированные в памяти SFR или периферийные SFR. К картированным в памяти SFR необходимо обращаться используя режимы косвенной или индексной адресации, и они не могут выбираться в режиме окон. Периферийные SFR физически расположены на кристалле ИС 1874ВЕ66Т, и они могут выбираться в режиме «окон» (подраздел 5.2 «Работа с окнами»). Не рекомендуется использовать резервные SFR, необходимо записать в них нули или оставить их в заданном по умолчанию состоянии. При чтении резервные биты и SFR принимают неопределенные значения.

Примечание – Использование любого SFR как базового или индексного регистра для косвенных или индексных инструкций может вызвать непредсказуемые результаты, потому что внешние события могут изменить содержимое SFR. Также, потому что некоторые SFR очищаются при чтении, считается возможным использование SFR в инструкциях «чтение-модификация-запись» (например, XORB).

Картированные в памяти SFR

Ячейки 1FE0_H–1FFF_H содержат картированные в памяти SFR (см. таблицу 5.3). Ячейки в этом диапазоне, пропущенные в таблице 5.3, резервные. К SFR необходимо обращаться в режимах косвенной или индексной адресации, и они не могут быть выбраны в режимах «окон». К картированным SFR доступ осуществляется через контроллер памяти, т.к. команды, которые работают с этим SFR, выполняются так, как если бы они были из внешней памяти с нулевым временем ожидания.

Т а б л и ц а 5.3 – Картированные в памяти SFR

SFR для портов 3, 4, 5 и UPROM		
Адрес	Старший (нечетный) байт	Младший (четный) байт
1FFE _H	P4_PIN	P3_PIN
1FFC _H	P4_REG	P3_REG
...
1FF6 _H	P5_PIN	USFR
1FF4 _H	P5_REG	Зарезервировано
1FF2 _H	P5_DIR	Зарезервировано
1FF0 _H	P5_MODE	Зарезервировано

Периферийные SFR

Ячейки 1F00_H–1FDF_H обеспечивают доступ к периферийным SFR. В таблице 5.4 содержится список периферийных SFR микроконтроллера. Ячейки, которые пропущены в таблице, резервные. Периферийные SFR – регистры управления ввода-вывода; они физически расположены на кристаллах ИС 1874BE66T. К периферийным SFR возможен доступ в режиме окон и к ним можно обратиться как к слову или байту (кроме случаев, отдельно оговорённых в примечаниях к таблицам данного раздела).

К периферийным SFR обращение непосредственное, без использования контроллера памяти, и команды, которые обращаются к этим SFR, выполняются так, как если бы они оперировали с регистровым файлом.

Т а б л и ц а 5.4 – Периферийные SFR

SFR порта 2 и порта 7		
Адрес (шестнадцатеричный)	Старший (нечетный) байт	Младший (четный) байт
1FDE	Зарезервирован	Зарезервирован
...
1FD6	P7_PIN	P2_PIN
1FD4	P7_REG	P2_REG
1FD2	P7_DIR	P2_DIR
1FD0	P7_MODE	P2_MODE
SFR генератора формы сигнала		
Адрес (шестнадцатеричный)	Старший (нечетный) байт	Младший (четный) байт
1FCE	Зарезервирован	WG_PROTECT
1FCC	WG_CONTROL (H)	WG_CONTROL (L)
1FCA	WG_COUNTER (H)	WG_COUNTER (L)
1FC8	WG_RELOAD (H)	WG_RELOAD (L)
1FC6	WG_COMP3 (H)	WG_COMP3 (L)
1FC4	WG_COMP2 (H)	WG_COMP2 (L)
1FC2	WG_COMP1 (H)	WG_COMP1 (L)
1FC0	WG_OUTPUT (H)	WG_OUTPUT (L)
SFR периферийных прерывания и PWM		
Адрес (шестнадцатеричный)	Старший (нечетный) байт	Младший (четный) байт
1FBE	Зарезервирован	PI_PEND
1FBC	Зарезервирован	PI_MASK
1FBA	Зарезервирован	FREQ_CNT
1FB8	Зарезервирован	FREQ_GEN
1FB6	Зарезервирован	PWM_COUNT
1FB4	Зарезервирован	PWM_PERIOD
1FB2	Зарезервирован	PWM1_CONTROL
1FB0	Зарезервирован	PWM0_CONTROL
SFR АЦП		
Адрес (шестнадцатеричный)	Старший (нечетный) байт	Младший (четный) байт
1FAE	AD_TIME	AD_TEST
1FAC	Зарезервирован	AD_COMMAND
1FAA	AD_RESULT (H)	AD_RESULT (L)
1FA8	P1_PIN	P0_PIN
1FA6	Зарезервирован	Зарезервирован
...
1F80	Зарезервирован	Зарезервирован

Окончание таблицы 5.4

Адрес (шестнадцатеричный)	Старший (нечетный) байт	Младший (четный) байт
SFR EPA и таймеров		
1F7E*	TIMER2 (H)	TIMER2 (L)
1F7C	Зарезервирован	T2CONTROL
1F7A*	TIMER1 (H)	TIMER1 (L)
1F78	Зарезервирован	T1CONTROL
1F76	Зарезервирован	Зарезервирован
1F74	Зарезервирован	Зарезервирован
1F72	T1RELOAD (H)	T1RELOAD (L)
1F70	Зарезервирован	Зарезервирован
1F6E*	COMP5_TIME (H)	COMP5_TIME (L)
1F6C	Зарезервирован	COMP5_CON
1F6A	COMP4_TIME (H)	COMP4_TIME (L)
1F68*	Зарезервирован	COMP4_CON
1F66*	COMP3_TIME (H)	COMP3_TIME (L)
1F64	Зарезервирован	COMP3_CON
1F62*	COMP2_TIME (H)	COMP2_TIME (L)
1F60	Зарезервирован	COMP2_CON
1F5E*	COMP1_TIME (H)	COMP1_TIME (L)
1F5C	Зарезервирован	COMP1_CON
1F5A*	COMP0_TIME (H)	COMP0_TIME (L)
1F58	Зарезервирован	COMP0_CON
1F56*	EPA5_TIME (H)	EPA5_TIME (L)
1F54	Зарезервирован	EPA5_CON
1F52*	EPA4_TIME (H)	EPA4_TIME (L)
1F50	Зарезервирован	EPA4_CON
1F4E*	EPA3_TIME (H)	EPA3_TIME (L)
1F4C	Зарезервирован	EPA3_CON
1F4A*	EPA2_TIME (H)	EPA2_TIME (L)
1F48	Зарезервирован	EPA2_CON
1F46*	EPA1_TIME (H)	EPA1_TIME (L)
1F44*	Зарезервирован	EPA1_CON
1F42*	EPA0_TIME (H)	EPA0_TIME (L)
1F40	Зарезервирован	EPA0_CON
* Адресуется как слово.		

Регистровый файл

Регистровый файл (рисунок 5.1) разделен на старший файл регистров и младший файл регистров. Старший файл регистров состоит из регистров общего назначения оперативной памяти. Младший файл регистров содержит регистры общего назначения оперативной памяти наряду с указателем стека SP и регистрами специальных функций центрального процессора SFR.

В таблице 5.5 перечислены адреса памяти регистрового файла. РАЛУ обращается к младшему файлу регистров непосредственно, без использования контроллера памяти. Он также доступен в режиме окон прямой адресацией (подраздел 5.2 «Работа с окнами»). Старший регистровый файл и периферийные SFR могут быть доступны в режиме окон. К регистрам в младшем файле регистров и регистрам в режиме окон можно обратиться прямой адресацией.

Примечание – Регистровый файл не должен содержать коды команд. Попытка выполнять команду из ячейки в файле регистров заставляет контроллер памяти принять команду из внешней памяти.

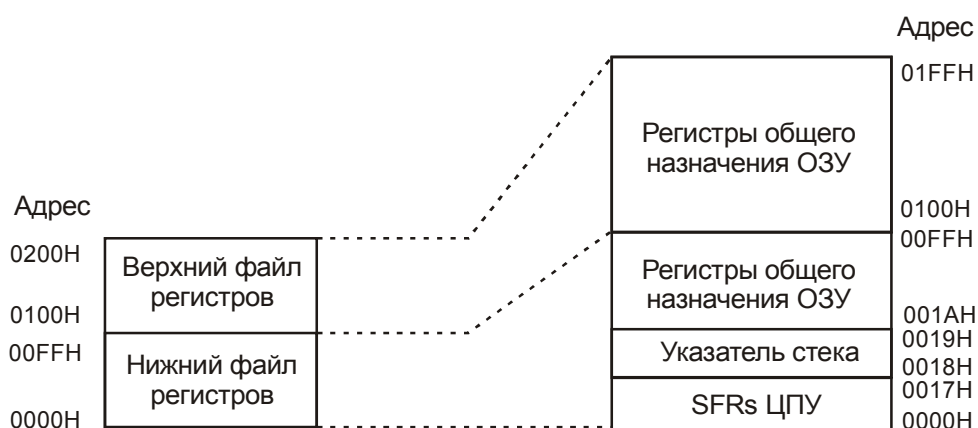


Рисунок 5.1 – Карта памяти файла регистров

Таблица 5.5 – Адреса памяти файла регистров

Диапазон адресов (шестнадцатеричные)	Описание	Способы адресации
01FF - 0100	Старший файл регистров (регистры оперативной памяти)	Косвенный, индексный или прямой в режиме окон
00FF - 001A	Младший регистровый файл (регистры оперативной памяти)	Прямой, косвенный или индексный
0019 - 0018	Младший регистровый файл (указатель стека)	Прямой, косвенный или индексный
0017 - 0000	Младший регистровый файл (SFR ЦПУ)	Прямой, косвенный или индексный

Регистры общего назначения оперативной памяти

Младший файл регистров содержит оперативную память регистров общего назначения. Ячейки указателя стека могут также использоваться как оперативная память регистров общего назначения, когда операции стека не выполняются. РАЛУ может обратиться к этой памяти непосредственно, используя прямую регистровую адресацию.

Старший файл регистров также содержит оперативную память регистров общего назначения. РАЛУ обычно использует косвенную или индексную адресацию, чтобы обратиться к оперативной памяти в старшем файле регистров. Работа с окнами дает возможность РАЛУ использовать прямую регистровую адресацию, чтобы обратиться к этой памяти.

Работа с окнами может обеспечить быстрое переключение контекста задач прерывания и более быстрого выполнения программы (подраздел 5.2 «Работа с окнами»). Блоки PTS и стек наиболее эффективны, когда расположены в старшем файле регистров.

Указатель стека SP

Ячейки памяти 0018_H и 0019_H содержат указатель стека SP. Он содержит адрес стека, должен указывать адрес слова (чётный), который на два байта больше, чем желаемый стартовый адрес. Прежде, чем центральный процессор выполняет вызов подпрограммы или процедуры обслуживания прерывания, он декрементирует SP на два и копирует адрес следующей команды из счетчика программы в стек. ЦП загружает адрес подпрограммы или процедуры обслуживания прерывания в счетчик программы. Когда выполняется команда RET в конце подпрограммы или программы обслуживания прерывания, ЦП загружает POP содержимое вершины стека (то есть адрес возврата) в счетчик программы и увеличивает SP на два.

Подпрограммы могут быть вложены. То есть, каждая подпрограмма может вызвать другие подпрограммы. Центральный процессор помещает содержание счетчика программы в стек каждый раз, когда выполняется вызов подпрограммы. Стек становится нисходящим, поскольку увеличивается число записей. Единственное ограничение на глубину вложения – количество доступной памяти. Поскольку центральный процессор возвращается из каждой вложенной подпрограммы, он выталкивает адрес из вершины стека и следующий адрес возврата перемещается в вершину стека.

Программа должна загрузить выровненный по слову (чётный) адрес в указатель стека. Необходимо выбрать адрес, который на два байта больше, чем желательный стартовый адрес, потому что центральный процессор автоматически уменьшает указатель стека прежде, чем помещает первый байт адреса возврата в стек. Стек становится нисходящим, что позволяет отвести достаточно места для максимального числа обращений к стеку. Стек должен быть расположен или во внутреннем файле регистров или во внешней оперативной памяти. Стек может использоваться наиболее эффективно, когда он расположен в файле регистров.

Следующий пример инициализирует вершину старшего файла регистров микроконтроллера как стек:

LD SP, #200_H; загрузить указатель стека

Следующий пример показывает, как позволить устройству ввода позиций компоновщика определять, где стек соответствует карте памяти:

LD SP, #STACK

Регистры специальных функций SFR центрального процессора

Ячейки 0000_H–0017_H в младшем файле – регистры специальных функций центрального процессора (таблица 5.6).

Т а б л и ц а 5.6 – Регистры SFR центрального процессора

Адрес	Старший (нечетный) байт	Младший (четный) байт
0016 _H	Резервный	Резервный
0014 _H	Резервный	WSR
0012 _H	INT_MASK1	INT_PEND1
0010 _H	Резервный	Резервный
000E _H	Резервный	Резервный
000C _H	Резервный	Резервный
000A _H	Резервный	WATCHDOG
0008 _H	INT_PEND	INT_MASK
0006 _H	PTSSRV (H)	PTSSRV (L)
0004 _H	PTSSEL (H)	PTSSEL (L)
0002 _H	ONES_REG (H)	ONES_REG (L)
0000 _H	ZERO_REG (H)	ZERO_REG (L)

Окончание таблицы 5.6

Примечание – Использование любого SFR как базового или индексного регистра для косвенных или индексных инструкций может вызвать непредсказуемые результаты, потому что внешние события могут изменить содержание SFR. Так как некоторые SFR очищаются при чтении, допускается использование SFR как операнды в инструкциях «чтение-модификация-запись» (например, XORB).

5.2 Работа с окнами

Работа с окнами расширяет объем памяти, который является доступным с прямой регистровой адресацией. Прямая регистровая адресация может обратиться к младшему файлу регистров с короткими, быстро выполняющимися командами. При работе с окнами прямая регистровая адресация может также обратиться к старшему файлу регистров и периферийным SFR.

Работа с окнами отображает сегмент памяти (верхний файл регистров или периферийные SFR) в младший файл регистров. Регистр выбора окна WSR выбирает 32-, 64- или 128-байтный сегмент старшей памяти для создания окон в вершине младшего файла регистров. На рисунке 5.2 показано 128-байтное окно.

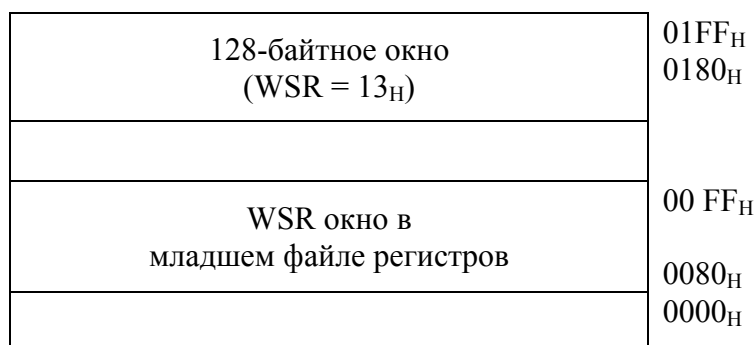


Рисунок 5.2 – Работа с окнами

Примечание – Картированные в памяти SFR могут быть доступны в режимах косвенной или индексной адресации; к ним нельзя обратиться через окно. Чтение картированного в памяти SFR через окно дает FF_H (все «1»), а запись в картированный в памяти SFR через окно не имеет никакого эффекта.

Выбор окна

Регистр выбора окна (рисунок 5.3, таблица 5.7) выбирает окно, которое будет отображено в вершине младшего файла регистров.

В таблице 5.8 содержатся значения WSR для работы с окнами периферийных SFR. В таблице 5.9 содержатся значения WSR для работы с окнами старшего регистрового файла.

Регистр WSR

Адрес: 0014_H

Состояние сброса: 00_H

Регистр выбора окна (WSR) помещает секции оперативной памяти в вершину младшего регистрового файла с 32-, 64- или 128-байтовым приращением. PUSHA сохраняет этот регистр в стеке, а POPA восстанавливает его.

7	0						
–	W6	W5	W4	W3	W2	W1	W0

Рисунок 5.3 – Регистр выбора окна (WSR)

Т а б л и ц а 5.7 – Разряды регистра выбора окна (WSR)

Номер бита	Мнемоника бита	Функция
7	–	Зарезервирован; для совместимости с будущими устройствами записывается «0»
6 – 0	W6 – W0	Выбор окна. Эти биты определяют размер окна и их число. Смотреть таблицу 5.8 для периферийных SFR или таблицу 5.9 для старшего файла регистра

Т а б л и ц а 5.8 – Выбор окна периферийных SFR

Внешние устройства	WSR значение для 32-байтного окна (00E0 _H –00FF _H)	WSR значение для 64-байтного окна (00C0 _H –00FF _H)	WSR значение для 128-байтного окна (0080 _H –00FF _H)
Порт 2 ГФС	7E _H	3F _H	1F _H
Периферийные прерывания ШИМ АЦП	7D _H	3E _H	
Таймеры 1–2 ЕРА сравнение 0–5	7B _H	3D _H	1E _H
ЕРА захват/сравнение 0–5	7A _H		

Т а б л и ц а 5.9 – Выбор окна старшего файла регистров

Оперативная память РОН	WSR значение для 32-байтового окна (00E0 _H –00FF _H)	WSR значение для 64-байтового окна (00C0 _H –00FF _H)	WSR значение для 128-байтового окна (0080 _H –00FF _H)
01E0 _H –01FF _H	4F _H	27 _H	13 _H
01C0 _H –01DF _H	4E _H		
01A0 _H –01BF _H	4D _H		
0180 _H –019F _H	4C _H	26 _H	12 _H
0160 _H –017F _H	4B _H	25 _H	
0140 _H –015F _H	4A _H		
0120 _H –013F _H	49 _H	24 _H	
0100 _H –011F _H	48 _H		

Адресация к ячейкам памяти через окно

После того, как выбрано желательное окно, необходимо знать в окне прямой адрес ячейки памяти (адрес в младшем файле регистра). Вычисляется прямой адрес следующим образом:

- вычесть базовый адрес области, которая будет повторно отображена (таблица 5.10) из адреса выбранной ячейки. Это дает смещение адреса ячейки;

- добавляется смещение к базовому адресу окна (таблица 5.11). Результат – прямой адрес в окне.

Т а б л и ц а 5.10 – Адресные окна

Базовый адрес	WSR значение для 32-байтового окна (00E0 _H – 00FF _H)	WSR значение для 64-байтового окна (00C0 _H –00FF _H)	WSR значение для 128-байтового окна (0080 _H –00FF _H)
Периферийные SFR			
1FE0 _H	7F _H	3F _H	1F _H
1FC0 _H	7E _H		
1FA0 _H	7D _H	3E _H	
1F80 _H	7C _H		
1F60 _H	7B _H	3D _H	1E _H
1F40 _H	7A _H		
1F20 _H	79 _H	3C _H	
1F00 _H	78 _H		
02E0 _H	57 _H	2B _H	15 _H
02C0 _H	56 _H		
02A0 _H	55 _H	2A _H	
0280 _H	54 _H		
0260 _H	53 _H	29 _H	14 _H
0240 _H	52 _H		
0220 _H	51 _H	28 _H	
0200 _H	50 _H		
01E0 _H	4F _H	27 _H	13 _H
01C0 _H	4E _H		
01A0 _H	4D _H	26 _H	
0180 _H	4C _H		
0160 _H	4B _H	25 _H	12 _H
0140 _H	4A _H		
0120 _H	49 _H	24 _H	
0100 _H	48 _H		
Пр и м е ч а н и е – Ячейки 1FE0 _H –1FFF _H содержат картированные в памяти SFR, к которым нельзя обратиться через окно. Чтение по этим адресам через окно устанавливает FF _H ; запись по этим адресам через окно не имеет никакого эффекта.			

Т а б л и ц а 5.11 – Базовые адреса окон

Размер окна	WSR базовый адрес (базовый адрес в младшем файле регистров)
32 байта	00E0 _H
64 байта	00C0 _H
128 байтов	0080 _H

Пример работы с 32-байтными окнами

Предположим, необходимо обратиться к ячейке 014B_H (в старшем файле регистров, используемом для оперативной памяти регистров общего назначения) с прямой регистровой адресацией через 32-байтовое окно. В таблице 5.10 находим, что нужно записать 4A_H в регистр выбора окна. Базовый адрес 32-байтовой области памяти – 0140_H. Чтобы определить смещение, необходимо вычесть этот базовый адрес из адреса, к которому будет обращение (014B_H – 0140_H = 000B_H). Добавить смещение к базовому адресу окна в младшем файле регистров (00E0_H, таблица 5.11). Прямой адрес – 00EB_H (000B_H + 00E0_H).

Пример работы с 64-байтными окнами

Предположим, необходимо обратиться к регистру WG_CONTROL (ячейка 1FCC_H) с прямой адресацией через 64-байтовое окно. В таблице 5.9 указано, что необходимо записать 3F_H в регистр выбора окна. Базовый адрес 64-байтовой области памяти – 1FC0_H. Чтобы определить смещение, надо вычесть этот базовый адрес из адреса, к которому будет обращение (1FCC_H – 1FC0_H = 000C_H) и добавить смещение к базовому адресу окна в младшем файле регистров (00C0_H, см. таблицу 5.10). Прямой адрес – 00CC_H (000C_H + 00C0_H).

Пример работы со 128-байтными окнами

Предположим, необходимо обратиться к ячейке 1F42_H (регистр EPA0_TIME) с прямой регистровой адресацией через 128-байтовое окно. В таблице 5.9 указывается, что необходимо написать 1E_H в регистр выбора окна. Базовый адрес 128-байтовой области памяти – 1F00_H. Чтобы определить смещение, надо вычесть этот базовый адрес из адреса, к которому будет обращение (1F42_H – 1F00_H = 0042_H) и добавить смещение к базовому адресу окна в младшем файле регистров (0080_H, см. таблицу 5.10). Прямой адрес – 00C2_H (0042_H + 0080_H).

Пример работы с окнами в области памяти, не поддерживаемой режимом окон

Предположим, необходимо обратиться к ячейке 1FF1_H (регистр P5_MODE, картированный в памяти SFR) с прямой регистровой адресацией через 128-байтовое окно. Эта ячейка находится в диапазоне адресов (1FE0_H–1FFF_H), который не поддерживается режимом окон. Хотя можно выбрать окно, записать 1F_H в WSR, чтение этой ячейки через окно установит FF_H (все 1), и запись не изменит содержимое. Однако возможно обратиться к периферийным SFR в диапазоне 1F80_H–1FDF_H по их прямым адресам в окне.

Работа с окнами и способы адресации

При установке режима окна доступ к соответствующим ячейкам осуществляется через окно, используя прямую (8 битов) адресацию и обычную адресацию на 16 битов. Ячейки младшего файла регистров, которые покрыты окном, всегда доступны косвенными или индексными инструкциями. Повторное разрешение прямого доступа ко всему младшему регистровому файлу очищает WSR. Для разрешения прямого доступа к определенной ячейке в младшем файле регистров необходимо выбрать меньшее окно, которое не покрывает эту ячейку.

Когда работа с окнами разрешена:

- прямая регистровая команда, использующая адрес в пределах младшего файла регистров, фактически обращается к окну в старшем файле регистров;
- команды косвенные, индексированные или с нулевым регистром, использующие адрес или в пределах младшего файла регистров, или в пределах старшего регистрового файла, обращаются к ячейкам памяти.

Следующая типовая программа иллюстрирует различие между прямой регистровой и индексной адресацией при использовании работы с окнами:

```
PUSHA          ; помещает содержание WSR в стек
LDB WSR, #12H  ; выбор окна 12H, 128-байтовый блок
                ; следующая команда использует прямую регистровую адресацию
ADD 40H, 80H  ; mem_word (40H) ← mem_word (40H) + mem_word (380H)
                ; следующие две команды используют косвенную адресацию
ADD 40H, 80H [0] ; mem_word (40H) ← mem_word (40H) + mem_word (80H + 0)
ADD 40H, 380H [0]; mem_word (40H) ← mem_word (40H) + mem_word (380H + 0)
POPA          ; перезагружает предыдущее содержание в WSR
```

6 Стандартные и PTS прерывания

В этом разделе описывается схема управления прерываниями, схема приоритетов и синхронизации для стандартных прерываний и прерываний периферийного сервера транзакций PTS. Описаны три специальных прерывания и семь режимов PTS, четыре из которых используются с ЕРА, чтобы обеспечить последовательный программный канал ввода-вывода для синхронных и для асинхронных передач и приемов. Описано программирование и управление прерываниями.

6.1 Краткий обзор прерываний

Схема управления прерываниями микроконтроллера разрешает событиям в реальном времени управлять процессом выполнения программы. Когда событие генерирует прерывание, устройство приостанавливает выполнение текущего потока команд и начинает выполнять процедуру обслуживания в ответ на прерывание. Когда обслуживание прерывания завершается, выполнение программы продолжается в точке, где произошло прерывание. Периферийное устройство, внешний сигнал или команда могут произвести запрос прерывания. В самом простом случае устройство получает запрос, исполняет обслуживание и возвращается к задаче, которая была прервана.

Гибкая система обработки прерываний микроконтроллера имеет два главных компонента: программируемый контроллер прерываний и периферийный сервер PTS. Программируемый контроллер прерываний имеет аппаратную схему приоритета, которая может быть изменена вашим программным обеспечением. Прерывания, которые проходят через контроллер прерываний, обслуживаются программами обслуживания прерываний. Старшие и младшие векторы в памяти специального назначения (раздел 5 «Организация памяти») содержат адреса программ обслуживания прерываний. Периферийный сервер PTS обеспечивает быстродействующую обработку, не изменяя стек или PSW. Большинство прерываний (кроме NMI, TRAP и невыполняемого кода) можно обслуживать в PTS вместо контроллера прерываний.

PTS поддерживает семь специальных микропрограмм, которые позволяют ему выполнять определенные задачи значительно быстрее, чем эквивалентные процедуры обслуживания прерываний. PTS может передать байты или слова (индивидуально или в блоках) между любыми ячейками запоминающего устройства, управлять многократным аналого-цифровым (АЦ) преобразованием и передавать и принимать последовательные данные или в асинхронном, или в синхронном режиме. PTS прерывания имеют более высокий приоритет, чем стандартные прерывания, и могут временно приостановить стандартные процедуры обслуживания прерываний.

Блок данных, называемый блоком управления PTS (PTSCB), содержит определенные данные для каждой процедуры PTS. Когда формируется PTS прерывание, кодирующее устройство приоритета выбирает соответствующий вектор и захватывает блок управления PTS (PTSCB).

Т а б л и ц а 6.1 – Сигналы прерываний

Сигнал прерывания	Тип	Описание
EXTINT	Вход	Внешнее прерывание. Это программируемое прерывание управляется регистром WG_PROTECT. Регистр определяет какое событие вызовет прерывание: передний фронт/высокий уровень сигнала или задний фронт/низкий уровень сигнала. В режиме пониженного энергопотребления удержание сигнала EXTINT не менее 50 нс возвращает микроконтроллер в режим нормальной работы. Прерывание при этом не нуждается в разрешении. Если прерывание EXTINT разрешено, процессор вызывает программу обслуживания, иначе центральный процессор выполняет инструкцию, которая немедленно следует за командой, которая вызвала режим пониженного энергопотребления. В режиме холостого хода любое разрешенное прерывание возвращает контроллер в режим нормальной работы
NMI	Вход	Немаскируемое прерывание. NMI имеет самый высокий приоритет из всех приоритетных прерываний. Для гарантированного распознавания необходимо удерживать длительность сигнала NMI более одного машинного цикла

Т а б л и ц а 6.2 – Регистры управления и состояния прерываний и PTS

Мнемоника	Адрес	Описание
INT_MASK INT_MASK1	0008 _H , 0013 _H	Регистры маски прерываний. Эти регистры разрешают/запрещают каждое маскируемое прерывание, кроме MNI, программных TRAP и прерываний по некорректному коду инструкций
INT_PEND INT_PEND1	0009 _H , 0012 _H	Регистры задержки прерываний. Биты в этих регистрах, установленные аппаратными средствами, указывают, какие прерывания задержаны в обслуживании
PI_MASK	1FBC _H	Регистр маски периферийных прерываний. Биты в этом регистре разрешают запросы прерываний, переполнение/антипереполнение таймеров 1, 2 и запросы прерываний генератора формы сигнала
PI_PEND	1FBE _H	Регистр задержки прерываний периферийных устройств. Любой установленный бит указывает на задержку обслуживания соответствующего запроса прерываний
PSW	Нет прямого доступа	Слово состояния процессора. Этот регистр содержит один бит, который глобально разрешает/запрещает обслуживание всех маскируемых прерываний, и другой, который разрешает/запрещает PTS. Эти биты устанавливаются или сбрасываются командами EI, DI, EPTS и DPTS, соответственно
PTSSEL	0004 _H , 0005 _H	Регистр выбора PTS. Этот регистр выбирает или процедуру PTS или стандартную процедуру обслуживания прерывания для каждого из маскируемых запросов прерываний
PTSSRV	0006 _H , 0007 _H	Регистр обслуживания PTS. Биты в этом регистре, установленные аппаратными средствами, запрашивают обслуживание окончания прерываний PTS (т.н. end-of-PTS)

6.3 Источники и приоритеты прерывания

В таблице 6.3 содержится перечень источников прерываний, их заданные по умолчанию приоритеты (30 – самый высокий и 00 – самый низкий) и их векторы-адреса. Прерывания по некорректному коду инструкции и программные TRAP не имеют приоритетов. Они непосредственно поступают для обслуживания в контроллер прерываний. Кодирование устройства приоритета определяет приоритет всех остальных задержанных прерываний. NMI имеет самый высокий приоритет. PTS прерывания имеют следующий самый высокий приоритет, а стандартные прерывания – самый низкий. Кодирование устройства приоритета выбирает самый высокий приоритет удержанного запроса, и контроллер прерываний выбирает соответствующую вектору ячейку в запоминающем устройстве специального назначения. Этот вектор содержит стартовый адрес соответствующего блока управления PTS (PTSCB) или программу обслуживания. PTSCB должен быть расположен на границе, кратной четырем словам, во внутреннем файле регистров.

Т а б л и ц а 6.3 – Источники прерываний, векторы и приоритеты

Источник прерывания	Мнемоника	Обслуживание контроллера прерываний			Обслуживание PTS		
		Название	Вектор	Приоритет	Название	Вектор	Приоритет
Немаскируемое прерывание	NMI	INT15	203E _H	30	–	–	–
Вывод EXTINT	EXTINT	INT14	203C _H	14	PTS14	205C _H	29
WF генератор и ERA COMP5	PI *	INT13	203A _H	13	PTS13	205A _H	28
ERA захват/сравнение 5	CAPCOMP5	INT12	2038 _H	12	PTS12	2058 _H	27
ERA сравнение 4	COMP4	INT11	2036 _H	11	PTS11	2056 _H	26
ERA захват/сравнение 4	CAPCOMP4	INT10	2034 _H	10	PTS10	2054 _H	25
ERA сравнение 3	COMP3	INT09	2032 _H	09	PTS09	2052 _H	24
ERA захват/сравнение 3	CAPCOMP3	INT08	2030 _H	08	PTS08	2050 _H	23
Некорректный код команды	–	–	2012 _H	–	–	–	–
Команды TRAP программного обеспечения	–	–	2010 _H	–	–	–	–
ERA сравнение 2	COMP2	INT07	200E _H	07	PTS07	204E _H	22
ERA захват/сравнение 2	CAPCOMP2	INT06	200C _H	06	PTS06	204C _H	21
ERA сравнение 1	COMP1	INT05	200A _H	05	PTS05	204A _H	20
ERA захват/сравнение 1	CAPCOMP1	INT04	2008 _H	04	PTS04	2048 _H	19
ERA сравнение 0	COMP0	INT03	2006 _H	03	PTS03	2046 _H	18
ERA захват/сравнение 0	CAPCOMP0	INT02	2004 _H	02	PTS02	2044 _H	17
АЦ преобразование завершено	AD_DONE	INT01	2002 _H	01	PTS01	2042 _H	16
Переполнение таймеров 1 или 2	OVRTM *	INT00	2000 _H	00	PTS00	2040 _H	15

* Обслуживание PTS некорректно для мультиплексных прерываний, потому что PTS не может определить источник этих прерываний.

Специальные прерывания

Микроконтроллеры имеют три специальных источника прерываний, которые всегда разрешены: некорректный код инструкции, программный TRAP и NMI. Эти прерывания не разрешаются или запрещаются командами EI или DI, и они не могут быть замаскированы. Все они обслуживаются контроллером прерываний и не могут быть обслужены на PTS. Из них только NMI проходит детектор перехода и дешифратор приоритета. Другие два идут непосредственно к контроллеру прерываний для обслуживания.

Некорректный код инструкции

Если центральный процессор пытается выполнить некорректный код инструкции, вырабатывается косвенный вектор через ячейку 2012_H. Это предотвращает выполнение произвольной программы в течение отказов программного обеспечения и аппаратных средств. Вектор прерывания должен содержать стартовый адрес программы обслуживания ошибки, который не будет далее развивать ошибочную ситуацию. Прерывание по некорректному коду инструкции предотвращает запрос других прерываний и разрешение их обслуживания, пока следующая инструкция не будет выполнена.

Программный TRAP

Инструкция TRAP (код инструкции F7_H) вызывает прерывание по вектору в ячейке 2010_H. Инструкция TRAP обеспечивает прерывание, которое является полезным при отладке программного обеспечения или генерации прерываний программного обеспечения. Инструкция TRAP запрещает обслуживание других запросов прерывания, пока следующая инструкция не выполнена.

NMI

Внешний вывод NMI производит немаскируемое прерывание для реализации программ обслуживания критических ситуаций. NMI имеет самый высокий приоритет среди приоритизированных прерываний. Запрос передается непосредственно от детектора перехода в дешифратор приоритета и вызывает вектор косвенно через ячейку 203E_H. NMI сканируется в течение фазы 2 (CLKOUT высокий), захватывается микроконтроллером. Так как прерывание производится фронтом, только один запрос прерывания может быть сгенерирован, даже если вывод поддерживается в высоком уровне.

Если система не использует NMI прерывание, необходимо соединить вывод NMI с #0V, чтобы предотвратить несанкционированное прерывание.

Вывод внешнего прерывания

Схема защиты в генераторе формы сигнала (рисунок 6.2) контролирует внешний сигнал EXTINT. Когда она обнаруживает действительное событие на входе, то одновременно запрещает выходы генератора формы сигнала и генерирует запрос прерывания EXTINT. Биты 2 и 3 регистра защиты генератора формы сигнала (WG_PROTECT) (рисунок 8.9) выбирают тип внешнего события, которое произведет запрос прерывания: задний или передний фронт, низкий или высокий уровень.

Когда принимается «событие – уровень», внешний сигнал должен остаться постоянным в течение, по крайней мере, 24 периодов T_{BQ1} (24/F_{BQ1}), чтобы быть распознанным как действительное прерывание. Когда сигнал установлен, устройство выборки уровня принимает уровень сигнала три раза в течение 24 периодов T_{BQ1}. Когда соответствующий уровень распознан, устройство выборки уровня генерирует один импульс на выходе. Этот импульс производит запрос прерывания EXTINT. Режим «событие – уровень» полезен в оборудовании с высоким уровнем помех, где выброс помехи может вызвать незапланированный запрос прерывания.

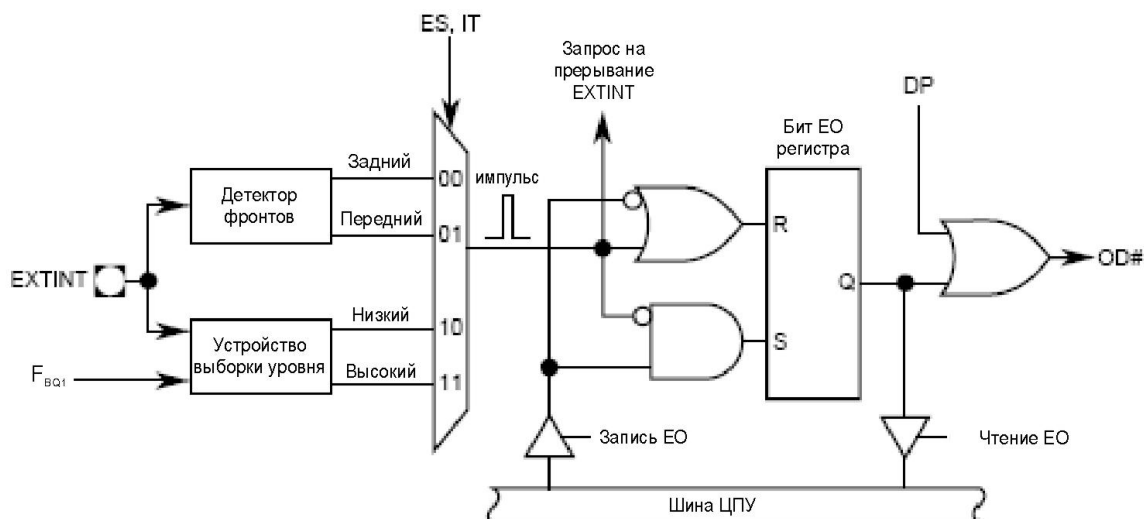


Рисунок 6.2 – Схема защиты генератора формы сигнала

Когда установлен режим «событие-фронт», сигнал на выходе должен остаться постоянным, по крайней мере, в течение двух T_{BQ1} ($2/F_{BQ1}$), чтобы быть распознанным как действительное прерывание. Когда действительный переход уровня произошел, детектор перехода формирует одиночный импульс. Импульс генерирует запрос прерывания EXTINT.

Мультиплексированные источники прерывания

Прерывания PI и OVRTM имеют мультиплексные источники (см. таблицу 6.3). Индивидуальный источник произведет прерывание, только если программное обеспечение разрешает источник прерывания и мультиплексное прерывание. Мультиплексное прерывание разрешается установкой соответствующего бита в регистре маски (рисунки 6.7 и 6.8). Чтобы разрешить источник прерывания, необходимо установить соответствующий бит в регистре PI_MASK (рисунок 6.9). На рисунке 6.3 приведена процедура обработки по переполнению таймеров (OVRTM).

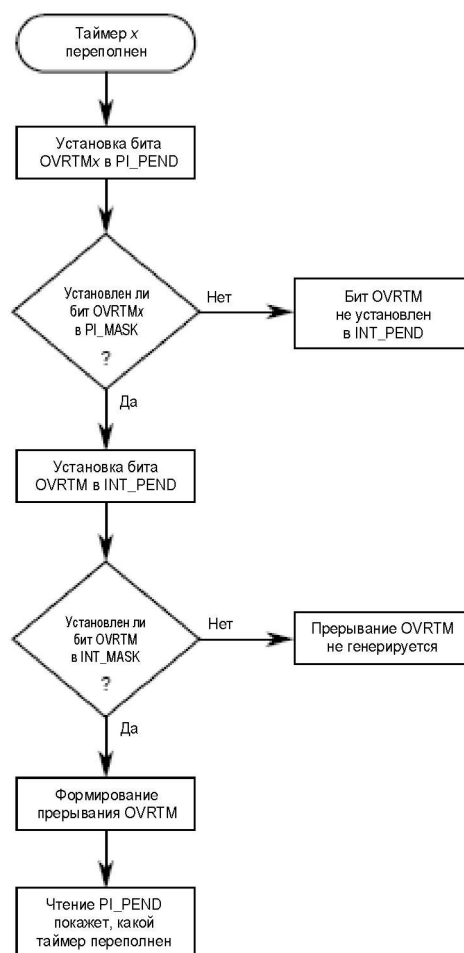


Рисунок 6.3 – Процедура обработки OVRTM прерывания

Программа обслуживания прерывания должна считать PI_PENDING регистр (рисунок 6.12), чтобы определить источник прерывания. Перед выполнением инструкции возврата программа обслуживания прерывания должна проверить наличие других задержанных в обслуживании источников прерываний. Обслуживание прерываний PTS не рекомендуется для мультиплексных прерываний, потому что PTS не может определить источник прерывания.

Прерывания по обслуживанию PTS (end-of-PTS)

Когда содержимое регистра PTSCOUNT становится равно нулю в конце однократной передачи, передачи блока, АЦП сканирования или при процедуре последовательного ввода-вывода, аппаратные средства очищают соответствующий бит в регистре PTSSEL (рисунок 6.6), который запрещает PTS обслуживание этого прерывания. Также устанавливается соответственно PTSSRV бит, запрашивающий end-of-PTS прерывание. End-of-PTS прерывание имеет тот же самый приоритет, что и соответствующее стандартное прерывание. Контроллер прерываний обслуживает его программой обработки прерываний, находящейся в ячейке запоминающего устройства, на которую указывает вектор стандартного прерывания. Например, PTS обслуживает CAPCOMP0 прерывание, если установлен PTSSEL.2. Векторы прерываний находятся в ячейке 2044_H, но вектор end-of-PTS прерывания в ячейке 2004_H – стандартный вектор CAPCOMP0 прерывания. Когда end-of-PTS прерывание обслуживается программой обработки прерывания, аппаратные средства очищают PTSSRV бит. Программа обслуживания end-of-PTS прерывания должна повторно инициализировать PTSCB, если требуется, и устанавливать соответствующий PTSSEL бит, разрешающий обслуживание PTS прерывания.

6.4 Время ожидания прерывания

Время ожидания прерывания – полная задержка между временем, когда запрос прерывания произведен, и временем, когда устройство начинает выполнение стандартной программы обслуживания прерывания или программы обслуживания прерывания PTS. Задержка происходит между временем, когда запрос прерывания обнаружен, и временем, когда он подтвержден. Запрос прерывания подтвержден, когда текущая инструкция заканчивает выполняться. Если запрос прерывания происходит в течение одного из последних четырех тактов инструкции, он не может быть подтвержден до окончания следующей инструкции. Эта дополнительная задержка происходит, потому что инструкции предвыбираются за несколько машинных циклов прежде, чем они выполняются. Таким образом, максимальная задержка между запросом прерывания и подтверждением – четыре машинных цикла плюс время выполнения следующей инструкции.

Когда запрос стандартного прерывания подтвержден, аппаратные средства очищают бит задержки прерывания и производят вызов по адресу, содержащемуся в соответствующем векторе прерывания. Когда запрос прерывания PTS подтвержден, аппаратные средства немедленно обращаются по вектору PTSCB, и начинается выполнение цикла PTS.

Ситуации, которые увеличивают время ожидания прерывания

Если запрос прерывания происходит в то время, когда любая из следующих инструкций выполняется, прерывание не будет подтверждено, пока не выполнится последующая инструкция:

- знаковый префикс кода инструкции (FE) для двухбайтовых команд умножения и деления со знаком;
- любая из этих восьми защищенных инструкций: DI, EI, APTS, EPTS, POPA, POPF, PUSHA, PUSHF;
- любая из инструкций «чтение-модификация-запись»: AND, ANDB, OR, ORB, XOR, XORB.

Прерывания по несуществующему коду команды и прерывания программного TRAP запрещают другие запросы прерываний, пока не выполнена последующая инструкция.

Каждый цикл PTS в пределах процедуры PTS не может быть прерван. Цикл PTS – отклик PTS на отдельный запрос прерывания. В режиме передачи блока цикл PTS состоит из передачи полного блока байтов или слов. Это означает худший случай времени ожидания – 500 машинных циклов, если происходит передача блока 32 слов из одной области внешней памяти в другую (см. таблицу 6.4 для информации о времени выполнения циклов PTS).

Вычисление времени ожидания

Максимальное время ожидания получается, когда запрос прерывания происходит слишком поздно для подтверждения во время текущей инструкции. Приведенное ниже вычисление худшего случая предполагает, что текущая инструкция – не защищенная. При вычислении времени ожидания используются нижеследующие термины.

Время окончания выполнения текущей инструкции (четыре цикла) – если инструкция защищенная, следующая за ней должна также выполняться перед подтверждением прерывания. Необходимо добавить время выполнения инструкции, которая следует за защищенной инструкцией.

Время выполнения следующей инструкции. Самая длинная инструкция NORML занимает 39 циклов. Однако BMOV инструкция может выполняться дольше, если передает большой блок данных. Если программа содержит процедуры, которые передают большие блоки данных, можно получить более точное значение худшего случая, если используется инструкция BMOV вместо NORML, см. приложение А.

Для стандартных прерываний время отклика на получение вектора и осуществление вызова – 11 циклов для внутреннего стека или 13 – для внешнего стека (считая шину адреса данных с нулевым состоянием ожидания).

Время ожидания стандартных прерываний

Худший случай задержки для стандартных прерываний – 56 циклов (4+39+11+2), если стек находится во внешнем запоминающем устройстве (см. рисунок 6.4). Это время задержки не включает время выполнения первой инструкции в программе обслуживания или выполнения инструкции после защищенной инструкции.

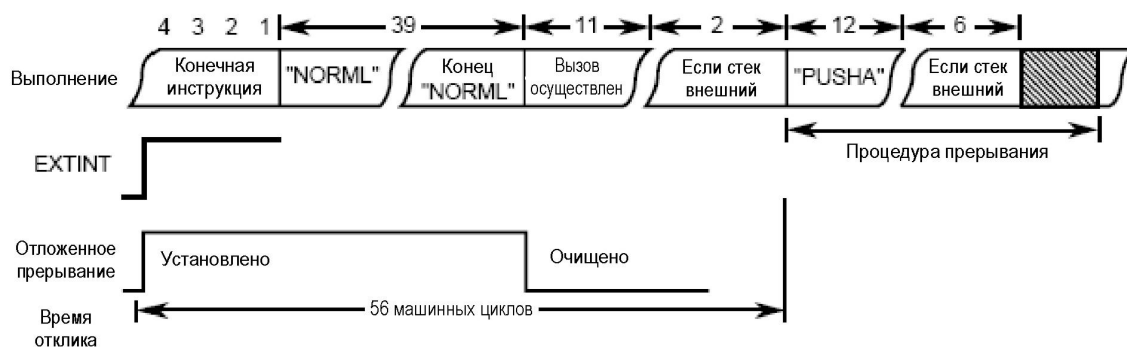


Рисунок 6.4 – Время отклика на стандартное прерывание

Время ожидания PTS прерывания

Максимальная задержка для PTS прерывания – 43 цикла (4 + 39) как показано на рисунке 6.5. Это время задержки не включает добавленную задержку, если выполняется защищенная инструкция или если запрос PTS уже принят (см. таблицу 6.4).

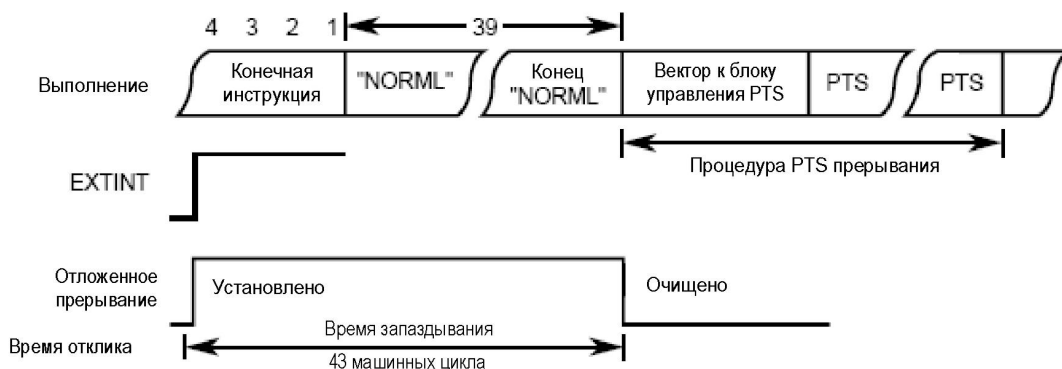


Рисунок 6.5 – Время отклика PTS прерывания

Т а б л и ц а 6.4 – Время выполнения циклов PTS

Режим PTS	Время выполнения (в машинных циклах)
1	2
Режим одиночной передачи регистр/регистр *	18 на передачу байта или слова + 1
память/регистр *	21 на передачу байта или слова + 1
память/память *	24 на передачу байта или слова + 1
Режим передачи блока	
регистр/регистр *	13 + 7 на передачу байта или слова (1 минимум)
память/регистр *	16 + 7 на передачу байта или слова (1 минимум)
память/память *	19 + 7 на передачу байта или слова (1 минимум)
Режим сканирования АЦП	
регистр/регистр *	21
регистр/память *	25

Окончание таблицы 6.4

1	2
ASIO режим приема Мажоритарность запрещена	24 + 2 (если контроль четности разрешен) 36 + время выборки (2-я выборка)
Мажоритарность разрешена	36 + 7 + время выборки (3-я выборка) 36 + 2 (если контроль четности разрешен)
ASIO режим передачи	29 + 3 (если контроль четности разрешен)
SSIO режим приема	29 (прием бита) 21 (нет приема)
SSIO режим передачи	30 (передача бита данных) 20 (нет передачи)
<p>* «Регистр» указывает доступ к файлу регистров или периферийному SFR. «Память» указывает доступ к занесенному в карту памяти регистру, устройству ввода-вывода или запоминающему устройству. См. таблицу 5.1 для информации о распределении памяти.</p>	

6.5 Программирование прерывания

Регистр выбора PTS (PTSSEL) выбирает или обслуживание PTS или стандартную процедуру обслуживания прерывания для каждого из маскируемых запросов прерываний (см. рисунок 6.6). Биты в регистрах маски прерываний INT_MASK и INT_MASK1 позволяют или запрещают (маскируют) индивидуальные прерывания (см. рисунки 6.7 и 6.8). Для источников мультиплексных прерываний биты в регистре PI_MASK (см. рисунок 6.9) разрешают или запрещают (маскируют) индивидуальные источники прерываний. Установка бита разрешает соответствующий источник прерывания и очистку бита запрещения источника за исключением бита немаскируемого прерывания NMI (INT_MASK1.7).

Для запрета любого прерывания необходимо очистить его бит маски. Чтобы разрешить обслуживание стандартного прерывания, устанавливается его маскирующий бит и очищается его бит выбора PTS. Чтобы разрешить прерывание для обслуживания PTS, необходимо установить и маскирующий бит, и бит выбора PTS.

Когда назначается обслуживание прерываниями PTS, необходимо установить блок управления PTS (PTSCB) для каждого источника прерываний (см. подраздел 6.6 «Инициализация блоков управления PTS») и использовать EPTS инструкцию, чтобы глобально разрешить PTS. Для назначения обслуживания прерываний стандартными процедурами используют инструкцию EI, чтобы глобально разрешить обслуживание прерываний.

Примечание – Инструкция DI (запрет прерывания) не запрещает обслуживание PTS. Однако она запрещает обслуживание для запросов прерываний end-of-PTS. Если запрос прерываний происходит в то время, когда прерывания запрещены, соответствующий бит ожидания устанавливается в регистре INT_PEND1 или INT_PEND.

Обслуживание PTS не рекомендовано для мультиплексных прерываний, потому что PTS не может определить источник прерываний.

Регистр PTSSEL

Адрес: 0004_H

Состояние сброса: 0000_H

Регистр выбора PTS (PTSSEL) выбирает или PTS микропрограмму, или процедуру обслуживания стандартного прерывания для каждого запроса прерывания. Установка бита выбирает PTS микропрограмму; очистка бита выбирает процедуру обслуживания стандартного прерывания. Когда PTSCOUNT становится равным нулю, аппаратные

средства очищают соответственно PTSSEL бит. PTSSEL бит должен быть установлен в соответствии с техническим описанием для перевыбора канала PTS.

15	14	13	12	119	10	9	8
–	EXTINT	PI	CAPCOMP5	COMP4	CAPCOMP4	COMP3	CAPCOMP3
7	6	5	4	3	2	1	0
COMP2	CAPCOMP2	COMP1	CAPCOMP1	COMP0	CAPCOMP0	AD	OVRTM

Рисунок 6.6 – Регистр выбора PTS (PTSSEL)

Т а б л и ц а 6.5 – Разряды регистра выбора PTS (PTSSEL)

Номер бита	Функция			
15	Зарезервированный; для совместимости с будущими устройствами записать ноль в этот бит			
14 – 0	Установка бита разрешает обслуживание соответствующего прерывания микропрограммой PTS. Адреса ячеек векторов прерываний PTS			
	Мнемоника	Вектор PTS	Мнемоника	Вектор PTS
	EXTINT	205C _H	COMP2	204E _H
	PI *	205A _H	CAPCOMP2	204C _H
	CAPCOMP5	2058 _H	COMP1	204A _H
	COMP4	2056 _H	CAPCOMP1	2048 _H
	CAPCOMP4	2054 _H	COMP0	2046 _H
	COMP3	2052 _H	CAPCOMP0	2044 _H
	CAPCOMP3	2050 _H	AD	2042 _H
			OVRTM *	2040 _H
* Обслуживание PTS некорректно для мультиплексных прерываний, так как PTS не может определить источник прерываний.				

Регистр INT_MASK

Адрес: 0008_H

Состояние сброса: 00_H

Регистр маски прерываний (INT_MASK) разрешает или запрещает индивидуальные запросы прерываний. Команды EI и DI разрешают и запрещают обслуживание всех маскируемых прерываний. INT_MASK – младший байт слова состояния процессора (PSW). PUSHF или PUSHA сохранит содержимое этого регистра в стеке и затем очистит этот регистр. Вызов прерывания не может произойти немедленно после этих инструкций. POPF или POPA восстанавливают содержимое регистра.

7	6	5	4	3	2	1	0
COMP2	CAPCOMP2	COMP1	CAPCOMP1	COMP0	CAPCOMP0	AD	OVRTM

Рисунок 6.7 – Регистр маски прерываний (INT_MASK)

Т а б л и ц а 6.6 – Разряды регистра маски прерываний (INT_MASK)

Номер бита	Функция		
7 – 0	Установка бита разрешает соответствующее прерывание		
	Адреса ячеек векторов стандартных прерываний		
	Мнемоника бита	Прерывание	Стандартный вектор
	COMP2	ЕРА сравнение канал 2	200E _H
	CAPCOMP2	ЕРА захват/сравнение канал 2	200C _H
	COMP1	ЕРА сравнение канал 1	200A _H
	CAPCOMP1	ЕРА захват/сравнение канал 1	2008 _H
	COMP0	ЕРА сравнение канал 0	2006 _H
	CAPCOMP0	ЕРА захват/сравнение канал 0	2004 _H
	AD	АЦ преобразование завершено	2002 _H
OVRTM *	Переполнение/антипереполнение таймера	2000 _H	
<p>* Таймер 1 и таймер 2 могут производить мультиплексное прерывание по переполнению/антипереполнению таймера. Записывается PI_MASK, чтобы разрешить источники прерываний. Считается PI_PEND, чтобы определить источник, который вызвал прерывание.</p>			

Регистр INT_MASK1

Адрес: 0013_H

Состояние сброса: 00_H

Регистр маски прерывания (INT_MASK1) позволяет или запрещает индивидуальные запросы прерываний. (Инструкции EI и DI позволяют и запрещают обслуживание всех маскируемых прерываний.) INT_MASK1 может считываться или записываться как байтовый регистр. PUSHA сохраняет содержимое регистра в стеке, а POPA восстанавливает его содержимое.

7	6	5	4	3	2	1	0
NMI	EXTINT	PI	CAPCOMP5	COMP4	CAPCOMP4	COMP3	CAPCOMP3

Рисунок 6.8 – Регистр маски прерываний 1 (INT_MASK1)

Т а б л и ц а 6.7 – Разряды регистра маски прерываний 1 (INT_MASK1)

Номер бита	Функция		
7 – 0 *	Установка бита разрешает соответствующее прерывание		
	Мнемоника бит	Прерывание	Стандартный вектор
	NMI	Немаскируемое прерывание	203E _H
	EXTINT	Вывод EXTINT	203C _H
	PI *	Мультиплексное периферийное прерывание	203A _H
	CAPCOMP5	ЕРА захват/сравнение канал 5	2038 _H
	COMP4	ЕРА сравнение канал 4	2036 _H
	CAPCOMP4	ЕРА захват/сравнение канал 4	2034 _H
	COMP3	ЕРА сравнение канал 3	2032 _H
CAPCOMP3	ЕРА захват/сравнение канал 3	2030 _H	

Окончание таблицы 6.7

* ГФС и канал 5 блока ЕРА в режиме сравнения могут генерить это прерывание. Запись в PI_MASK разрешает источники прерываний, чтение PI_PEND определяет вызывающий прерывание источник.

Регистр PI_MASK

Адрес: 1FBC_H

Состояние сброса: AA_H

Регистр маски периферийных прерываний (PI_MASK) разрешает или запрещает запросы прерываний, связанные с периферийными прерываниями (PI), и прерываниями по переполнению/антипереполнению таймеров (OVRTM).

7	6	5	4	3	2	1	0
–	COMP5	–	WG	–	OVRTM2	–	OVRTM1

Рисунок 6.9 – Регистр маски прерываний периферийных устройств (PI_MASK)

Т а б л и ц а 6.8 – Разряды регистра маски прерываний периферийных устройств (PI_MASK)

Номер бита	Мнемоника бита	Функция
7, 5, 3, 1	–	Зарезервированы для совместимости с будущими устройствами, записываются «0»
6	COMP5	ЕРА сравнение. Канал 5 Установка этого бита разрешает прерывание для канала 5 блока ЕРА в режиме сравнения. Прерывания блока ГФС и ЕРА сравнение Канал 5 ассоциируются с периферийными прерываниями (PI). Установка INT_MASK.5 разрешает PI
4	WG	Генератор формы сигнала. Установка бита разрешает прерывание генератора формы сигнала. Прерывания блока ГФС и ЕРА сравнение Канал 5 ассоциируются с периферийными прерываниями (PI). Установка INT_MASK.5 разрешает PI
2	OVRTM2	Переполнение/антипереполнение таймера 2. Установка этого бита разрешает прерывание переполнения/антипереполнения таймера 2. Прерывания переполнения/антипереполнения таймера 2 и таймера 1 связаны с прерыванием переполнения/антипереполнения таймера (OVRTM). Установка INT_MASK.0 разрешает OVRTM
0	OVRTM1	Переполнение/антипереполнение таймера 1. Установка этого бита разрешает прерывание переполнения/антипереполнения таймера 1. Прерывания переполнения/антипереполнения таймера 2 и таймера 1 связаны с прерыванием переполнения/антипереполнения таймера (OVRTM). Установка INT_MASK.0 разрешает OVRTM

Изменение приоритетов прерываний

Программное обеспечение может изменять заданные по умолчанию приоритеты маскируемых прерываний, управляя регистрами маски прерываний (INT_MASK и INT_MASK1). Например, можно определить какое прерывание, если таковые вообще имеются, может прервать программу обслуживания.

Следующая программа показывает один из способов запретить всем прерываниям, кроме EXTINT (приоритет 14), прерывать процедуру обслуживания АЦ-преобразования (приоритет 01).

SERIAL_RI_ISR:

```
PUSHA ; Сохранить PSW, INT_MASK, INT_MASK1 и WSR
; (Запрет всех прерываний)
LDB INT_MASK1, *0100000B ; Разрешено только EXTINT
EI ; Разрешено обслуживание прерывания
; Обслужить AD_DONE прерывание
POPA ; Восстановить PSW, INT_MASK, INT_MASK1 и
; WSR регистры
RET
CSEG B 02002H ; Заполнить таблицу прерываний
DCW AD_DONE_ISR END
```

Необходимо заметить, что ячейка 2002_H в таблице векторов прерываний должна быть загружена значением AD_DONE_ISR прежде, чем происходит запрос на прерывание и что прерывание завершения АЦ преобразования должно быть разрешено для выполнения этой программы.

Эта программа, подобно всем программам обслуживания прерываний, составлена следующим образом:

1 После того как аппаратные средства обнаруживают и располагают по приоритетам запрос прерывания, они производят вызов прерывания. Это выталкивает программный счетчик в стек и затем загружает в него содержимое вектора, соответствующего самому высокому приоритету ожидаемого немаскированного прерывания. Аппаратные средства не будут позволять вызов другого прерывания, пока первая инструкция процедуры обслуживания прерывания не выполнена.

2 PUSHA-инструкция помещает содержимое PSW, INT_MASK, INT_MASK1 и регистра выбора окна (WSR) в стек и затем очищает PSW, INT_MASK и регистры INT_MASK1. В дополнение к арифметическим флагам PSW содержит бит разрешения глобального прерывания (I) и бит разрешения PTS (PSE). Очищая PSW и регистры масок прерывания, PUSHA фактически запрещает все маскируемые прерывания, запрещает стандартное обслуживание прерывания и запрещает PTS. Поскольку PUSHA – защищенная инструкция, она также запрещает вызовы прерывания, пока следующая инструкция не выполнится.

3 LDB INT_MASK1-инструкция разрешает те прерывания, которые определены для прерывания процедуры обслуживания. В этом примере только EXTINT может прервать принятую процедуру обслуживания прерываний. Разрешая или запрещая прерывания, программное обеспечение устанавливает свои собственные приоритеты обслуживания прерываний.

4 Инструкция EI снова позволяет обработку прерываний и запрещает вызовы прерываний, пока следующая инструкция не выполнится.

5 Реально процедура обслуживания прерываний выполняется в соответствии с приоритетами, установленными программным обеспечением.

6 В конце процедуры обслуживания инструкция POPA восстанавливает первоначальное содержимое регистров PSW, INT_MASK, INT_MASK1 и WSR; любые изменения, сделанные в этих регистрах в течение процедуры обслуживания прерываний, перезаписываются. Поскольку запросы прерываний не могут произойти немедленно после инструкции POPA, последняя инструкция RET выполнится прежде, чем может произойти другой запрос прерывания.

«Преамбула» и исполняемый код для этой процедуры не сохраняются и не восстанавливаются в регистровом ОЗУ. Процедура обслуживания прерываний предполагает размещение ее собственного набора регистров в младшем регистровом файле. Кроме того, память в верхнем файле регистров доступна через «окна».

Определение источника прерываний

Когда аппаратные средства обнаруживают прерывание, они устанавливают соответствующий бит в INT_PEND или регистре INT_PEND1 (рисунки 6.10 и 6.11). Установка бита происходит, хотя индивидуальное прерывание может быть замаскировано. Аппаратные средства очищают бит ожидания, когда программа обращается к процедуре обслуживания. Процедура обслуживания может читать INT_PEND и INT_PEND1, чтобы определить, которое из прерываний ожидается.

Программное обеспечение может генерировать прерывание установкой бита в регистре INT_PEND1 или INT_PEND. Мы рекомендуем использование инструкций «чтение-модификация-запись» типа AND и OR для модификации этих регистров.

ANDB INT_PEND, #1111110_B; очистка бита OVRTM

XOR INT_PEND, #00000001_B; установка бита OVRTM

Другие методы также могут быть использованы в цикле прерываний. Например, прерывание могло произойти в течение последовательности инструкций, которые загружают содержимое регистра ожидаемого прерывания во временный регистр, изменяют содержимое временного регистра и затем записывают содержимое временного регистра обратно в регистр ожидания прерываний. Если прерывание происходит в течение одного из последних четырех машинных циклов второй инструкции, оно не будет подтверждено до окончания завершения третьей инструкции. Поскольку третья инструкция переписывает содержимое регистра ожидания прерываний, то переход по вектору не будет происходить.

OVRTM прерывание имеет мультиплексные источники прерываний. Смотреть PI_PEND (рисунок 6.12), чтобы определить, какой источник произвел запрос прерывания. Чтение PI_PEND очищает все биты. PI_PEND – только читаемый регистр.

Регистр INT_PEND

Адрес: 0009_H

Состояние сброса: 00_H

Когда аппаратные средства обнаруживают запрос прерывания, устанавливается соответствующий бит в регистрах INT_PEND или INT_PEND1. Когда вектор выбран, то аппаратные средства очищают бит ожидания. Программное обеспечение может генерировать прерывание установкой соответствующего бита ожидания прерывания.

7	6	5	4	3	2	1	0
COMP2	EPA2	COMP1	EPA1	COMP0	EPA0	AD	OVRTM

Рисунок 6.10 – Регистр ожидания прерывания (INT_PEND)

Т а б л и ц а 6.9 – Разряды регистра ожидания прерывания (INT_PEND)

Номер бита	Функция		
7 – 0	Любой установленный бит указывает, что ожидается обслуживание прерывания. Бит прерывания очищается, когда процесс передается по соответствующему вектору прерывания		
	Мнемоника бита	Прерывание	Стандартный вектор
7	COMP2	EPA сравнение канал 2	200E _H
6	COMP2	EPA захват/сравнение канал 2	200C _H
5	COMP1	EPA сравнение канал 1	200A _H

Окончание таблицы 6.9

Номер бита	Функция		
	Мнемоника бита	Прерывание	Стандартный вектор
4	CAPCOMP1	ЕРА захват/сравнение канал 1	2008 _H
3	COMP0	ЕРА сравнение канал 0	2006 _H
2	CAPCOMP0	ЕРА захват/сравнение канал 0	2004 _H
1	AD	АЦ-преобразование завершено	2002 _H
0	OVRTM *	Переполнение/антипереполнение таймера	2000 _H

* Таймер 1 и таймер 2 могут произвести мультиплексное прерывание по переполнению/антипереполнению. Записав в PI_MASK разрешение источника прерывания, считайте PI_PEND, чтобы определить источник, который вызвал прерывание.

Регистр INT_PEND1Адрес: 0012_HСостояние сброса: 00_H

Когда аппаратные средства обнаруживают запрос прерывания, они устанавливают соответствующий бит в регистрах INT_PEND или INT_PEND1. Когда вектор выбран, аппаратные средства очищают соответствующий бит. Программное обеспечение может произвести прерывание установкой бита ожидания прерывания.

7	6	5	4	3	2	1	0
NMI	EXTINT	PI	CAPCOMP5	COMP4	CAPCOMP4	COMP3	CAPCOMP3

Рисунок 6.11 – Регистр ожидания прерывания 1 (INT_PEND1)

Т а б л и ц а 6.10 – Разряды регистра ожидания прерывания 1 (INT_PEND1)

Номер бита	Функция		
7 – 0	Установленный бит разрешает соответствующее прерывание		
	Мнемоника бита	Прерывание	Стандартный вектор
	NMI	Немаскируемое прерывание	203E _H
	EXTINT	Вывод EXTINT	203C _H
	PI *	Мультиплексное периферийное прерывание	203A _H
	CAPCOMP5	ЕРА захват/сравнение канал 5	2038 _H
	COMP4	ЕРА сравнение канал 4	2036 _H
	CAPCOMP4	ЕРА захват/сравнение канал 4	2034 _H
	COMP3	ЕРА сравнение канал 3	2032 _H
	CAPCOMP3	ЕРА захват/сравнение канал 3	2030 _H

* ГФС и канал 5 блока ЕРА в режиме сравнения могут генерить это прерывание. Запись в PI_MASK разрешает источники прерываний, чтение PI_PEND определяет вызывающий прерывание источник.

Регистр PI_PEND

Адрес: 1FBЕ_H

Состояние сброса: АА_H

Когда аппаратные средства обнаруживают ожидаемое периферийное прерывание или прерывание таймера, они устанавливают соответствующий бит в регистрах ожидания прерывания (INT_PEND или INT_PEND1) и в регистре ожидания периферийного прерывания (PI_PEND). Когда вектор выбран, аппаратные средства очищают соответствующий бит INT_PEND/INT_PEND1. Чтение этого регистра очищает все биты PI_PEND. Программное обеспечение может произвести прерывание установкой бита PI_PEND.

7	6	5	4	3	2	1	0
-	COMP5	-	WG -	-	OVRTM2	-	OVRTM1

Рисунок 6.12 – Регистр ожидания прерывания периферийных устройств

Т а б л и ц а 6.11 – Разряды регистра ожидания прерывания периферийных устройств PI_PEND

Номер бита	Мнемоника бита	Функция
7, 5, 3, 1	–	Резервные. Эти биты не определены
6	COMP5	ЕРА сравнение «канал 5». Установка этого бита показывает ожидание прерывания для канала 5 блока ЕРА в режиме сравнения. Прерывания блока ГФС и ЕРА сравнение «канал 5» ассоциируются с периферийными прерываниями (PI). Установка INT_MASK.5 разрешает PI. Установка PI_MASK.6 позволяет COMP5
4	WG	Генератор формы сигнала. Установка бита указывает, что ожидается прерывание
2	OVRTM2	Переполнение/антипереполнение таймера 2. Установленный бит указывает на ожидаемое прерывание по переполнению/антипереполнению таймера 2. Прерывания по переполнению/антипереполнению таймера 2 и таймера 1 связаны с прерыванием по переполнению/антипереполнению таймера (OVRTM). Установка INT_MASK.0 позволяет OVRTM. Установка PI_MASK.2 позволяет OVRTM2
0	OVRTM1	Переполнение/антипереполнение таймера 1. Установленный бит указывает на ожидаемое прерывание по переполнению/ антипереполнению таймера 1 и таймера 2. Они связаны с прерыванием по переполнению/антипереполнению таймера (OVRTM). Установка INT_MASK.0 позволяет OVRTM. Установка PI_MASK.0 позволяет OVRTM1

6.6 Инициализация блоков управления PTS

Каждое PTS прерывание запрашивает блок данных в регистровом ОЗУ, называемый блоком управления PTS (PTSCB). PTSCB определяет PTS микропрограмму, которая вызывает и устанавливает определенные параметры для программы. Необходимо

установить PTSCB для каждого источника прерываний перед разрешением соответствующего PTS прерывания.

Адрес первого младшего байта PTSCB находится в таблице векторов PTS в области памяти специального назначения. На рисунке 6.13 приводятся PTSCB для каждого режима PTS. Неиспользованные байты PTSCB могут использоваться как дополнительное ОЗУ.

Примечание – PTSCB должен быть расположен во внутреннем файле регистров. Ячейки первого байта PTSCB должны быть выровнены по границе слова кратного четырем (адрес, делимый на 8).

	Одиночная передача	Передача блока	АЦП сканирование	SIO #1	SIO #2
	*	*	*	PTSVEC1 (H)	*
	*	PTSBLOCK	*	PTSVEC1 (L)	SAMPTIME
	PTSDST (H)	PTSDST (H)	PTSPTR2 (H)	BAUD (H)	DATA (H)
	PTSDST (L)	PTSDST (L)	PTSPTR2 (L)	BAUD (L)	DATA (L)
	PTSSRC (H)	PTSSRC (H)	PTSPTR1 (H)	EPAREG (H)	PTSCON1
	PTSSRC (L)	PTSSRC (L)	PTSPTR1 (L)	EPAREG (L)	PORTMASK
	PTSCON	PTSCON	PTSCON	PTSCON	PORTREG (H)
PTSVECT	PTSCOUNT	PTSCOUNT	PTSCOUNT	PTSCOUNT	PORTREG (L)

* Не используется.

Рисунок 6.13 – Блоки управления PTS

Счетчик циклов PTS

Первая ячейка PTSCB, содержащая 8-битную величину, называется PTSCOUNT. Это значение определяет количество прерываний, которые обслуживаются процедурой PTS. Декремент PTS PTSCOUNT происходит после каждого цикла PTS. Когда PTSCOUNT достигает нуля, аппаратные средства очищают соответствующий PTSSEL бит и устанавливают PTSSRV бит (рисунок 6.6), который запрашивает прерывание end-of-PTS. Программа, обслуживающая end-of-PTS, должна повторно инициализировать PTSCB, если требуется, и устанавливать соответствующий PTSSEL бит, вновь разрешающий обслуживание прерывания PTS.

Регистр PTSSRV

Адрес: 0006_H

Состояние сброса: 0000_H

Регистр обслуживания PTS (PTSSRV) используется аппаратными средствами, чтобы указать, что последнее PTS прерывание обслужено программой PTS. Когда PTSCOUNT равен нулю, аппаратные средства очищают соответствующий бит PTSSEL и устанавливают бит PTSSRV, который запрашивает end-of-PTS прерывание. Когда прерывание end-of-PTS вызвано, аппаратные средства очищают PTSSRV бит. Бит в регистре PTSSEL надо установить для повторного разрешения канала PTS.

15	14	13	12	11	10	9	8
–	EXTINT	PI	CAP COMP5	COMP4	CAP COMP4	COMP3	CAP COMP3
7	6	5	4	3	2	1	0
COMP2	CAP COMP2	COMP1	CAP COMP1	COMP0	CAP COMP0	AD	OVRTM

Рисунок 6.14 – Регистр PTSSRV обслуживания PTS

Т а б л и ц а 6.12 – Разряды регистра PTSSRV обслуживания PTS

Номер бита	Функция			
15	Резервный. Этот бит не определен			
14 – 0	Бит для соответствующего прерывания устанавливается аппаратно через стандартный вектор запроса прерывания end-of-PTS:			
	Мнемоника бит	Вектор PTS	Мнемоника бит	Вектор PTS
	EXTINT	205C _H	COMP2	204E _H
	PI *	205A _H	CAPCOMP2	204C _H
	CAPCOMP5	2058 _H	COMP1	204A _H
	COMP4	2056 _H	CAPCOMP1	2048 _H
	CAPCOMP4	2054 _H	COMP0	2046 _H
	COMP3	2052 _H	CAPCOMP0	2044 _H
	CAPCOMP3	2050 _H	AD	2042 _H
		OVRTM *	2040 _H	
* Обслуживание PTS не применяется для мультиплексных прерываний, потому что PTS не может определить источники этих прерываний.				

Выбор режима PTS

Второй байт каждого PTSCB – всегда 8-битная величина PTSCON. Биты 5 – 7 выбирают режим PTS (рисунок 6.15). Функция битов 0 – 4 отличается для каждого режима PTS. Время выполнения цикла для каждого режима PTS приведено в таблице 6.4.

Регистр PTSCON

Адрес: PTSPCB + 1

Регистр управления PTS (PTSCON) выбирает режим PTS и устанавливает функции управления для этого режима.

7	6	5	4	3	2	1	0
M2	M1	M0	*	*	*	*	*

* Функция этого бита зависит от выбранного режима.

Рисунок 6.15 – Регистр выбора режима PTS (PTSCON биты 7 – 5)

Т а б л и ц а 6.13 – Разряды регистра

Номер бита	Мнемоника	Функция			
7 – 5	M2 – M0	Режим PTS			
		Эти биты выбирают режим PTS:			
		M2	M1	M0	
		0	0	0	передача блока
		0	0	1	последовательный прием
		0	1	0	зарезервирован
		0	1	1	последовательная передача
		1	0	0	однократная передача
		1	0	1	зарезервирован
		1	1	0	АЦ сканирование
		1	1	1	зарезервирован

Режимы одиночной передачи и блочной передачи

В однократном режиме передачи прерывание задает PTS передачу отдельного байта или слова, выбранного битом BW в PTSCON, из одной ячейки запоминающего устройства в другую. Этот режим типично используется с EPA, чтобы переместить захваченные значения времени из регистра времени события во внутреннее ОЗУ для дальнейшей обработки. На рисунке 6.16 приведен управляющий блок PTS для однократного режима передачи.

В однократном режиме передачи блок управления PTS содержит источник и адрес назначения (PTSSRC и PTSDST), регистр управления (PTSCON) и счетчик передач (PTSCOUNT).



Рисунок 6.16 – Блок управления PTS в режиме однократной передачи

Т а б л и ц а 6.14 – Регистры, адреса и функции блока управления PTS в режиме однократной передачи

Регистр	Адрес	Функция
1	2	3
PTSDST	PTSCB +4	Адрес назначения PTS. Содержит адрес ячейки запоминающего устройства назначения. Действительный адрес – любая незарезервированная ячейка запоминающего устройства; однако это должен быть четный адрес, если выбрана передача слова
PTSSRC	PTSCB +2	Адрес источника PTS. Адрес ячейки источника запоминающего устройства. Действительный адрес – любая незарезервированная ячейка запоминающего устройства; однако необходимо указывать четный адрес, если выбрана передача слова

Окончание таблицы 6.14

1	2	3	
PTSCON	PTSCB +1	Биты управления PTS	
		M2-0	Режим PTS M2 M1 M0 1 0 0 режим однократной передачи
		BW	Передача байта/слова; 0 передача слова; 1 передача байта
		SU *	Обновление PTSSRC; 0 перезагружает оригинальный адрес источника PTS после того, как каждый байт или слово передано; 1 сохраняет текущий адрес источника передачи PTS байта или передачи слова
		DU *	Обновление PTSDST; 0 перезагружает оригинальный адрес назначения PTS после того, как каждый байт или слово передано; 1 сохраняет текущий адрес источника передачи PTS байта или передачи слова
		SI *	Автоинкремент PTSSRC; 0 не увеличивает содержимое PTSSRC после того, как каждый байт или слово передано; 1 увеличивает содержание PTSSRC после каждого байта или передачи слова
		DI *	Автоинкремент PTSDST; 0 не увеличивает содержимое PTSDST после того, как каждый байт или слово передано; 1 увеличивает содержание PTSDST после каждого байта или передачи слова
PTSCOUNT	PTSCB +0	Последовательная передача слов или байтов. Определяют число слов или байтов, которые будут переданы в течение отдельной процедуры передачи. Каждая передача слова или байта – один цикл PTS. Максимальное значение – 255	
* Биты DU/DI и биты SU/SI спарены в режиме однократной передачи. Каждая пара должна быть установлена или очищена вместе. Однако эти две пары, DU/DI и SU/SI, не обязательно должны быть равны.			

PTSCB в таблице 6.15 определяет девять циклов PTS. Каждый цикл перемещает единственное слово из ячейки 20_H во внешнюю ячейку запоминающего устройства. PTS передает первое слово в ячейку 6000_H . Затем он увеличивает и обновляет адрес назначения и декрементирует регистр PTSCOUNT, но не увеличивает адрес источника. Когда второй цикл начинается, PTS перемещает второе слово из ячейки 20_H в 6002_H ячейку. Когда PTSCOUNT равняется нулю, PTS заполнит ячейку $6000-600F_H$ и end-of-PTS прерывание генерируется.

Т а б л и ц а 6.15 – Однократный режим передачи PTSCB

Неиспользованный
Неиспользованный
PTSDST (H) = 60_H
PTSDST (L) = 00_H
PTSSRC (H) = 00_H
PTSSRC (L) = 20_H
PTSCON = 85_H (режим = 100, BW = 0, SI/SU = 0, DI/DU = 1)
PTSCOUNT = 09_H

Режим блочной передачи

В режиме блочной передачи прерывание заставляет PTS перемещать блок байтов или слов из одних ячеек запоминающего устройства в другие. На рисунке 6.17 показан блок управления PTS в режиме передачи блока.

В этом режиме каждый цикл PTS состоит из передачи определённого блока байтов или слов. Поскольку цикл PTS не может быть прерван, режим передачи блока может создать самое длительное время ожидания прерывания. Время ожидания в худшем случае может быть 500 циклов, если передача блока 32 слов из одних ячеек внешнего запоминающего устройства в другие использует 8-битную шину без циклов ожидания. Время выполнения циклов PTS приведено в таблице 6.4.

PTSCB (таблица 6.16) настраивает три цикла PTS, которые передадут 5 байтов от ячеек запоминающего устройства: 2024_H в $6000_H - 6004_H$ (цикл 1), $6005_H - 6009_H$ (цикл 2) и $600A_H - 600E_H$ (цикл 3). Источники адреса и назначения увеличиваются после каждой передачи байта, но только адрес назначения обновляется в PTSSRC в конце каждого цикла передачи блока. В этой процедуре PTS всегда получает первый байт из ячейки 20_H .

Т а б л и ц а 6.16 – Режим передачи блока PTSCB

Неиспользованный
PTSBLOCK = 05_H
PTSDST (H) = 60_H
PTSDST (L) = 00_H
PTSSRC (H) = 00_H
PTSSRC (L) = 20_H
PTSCON = 17_H (режим = 000; DI, SI, DU, BW = 1; SU = 0)
PTSCOUNT = 03_H

В режиме передачи блок управления PTS содержит размер блока (PTSBLOCK), источник и адрес назначения (PTSSRC и PTSDST), регистр управления (PTSCON) и счетчик передач (PTSCOUNT).

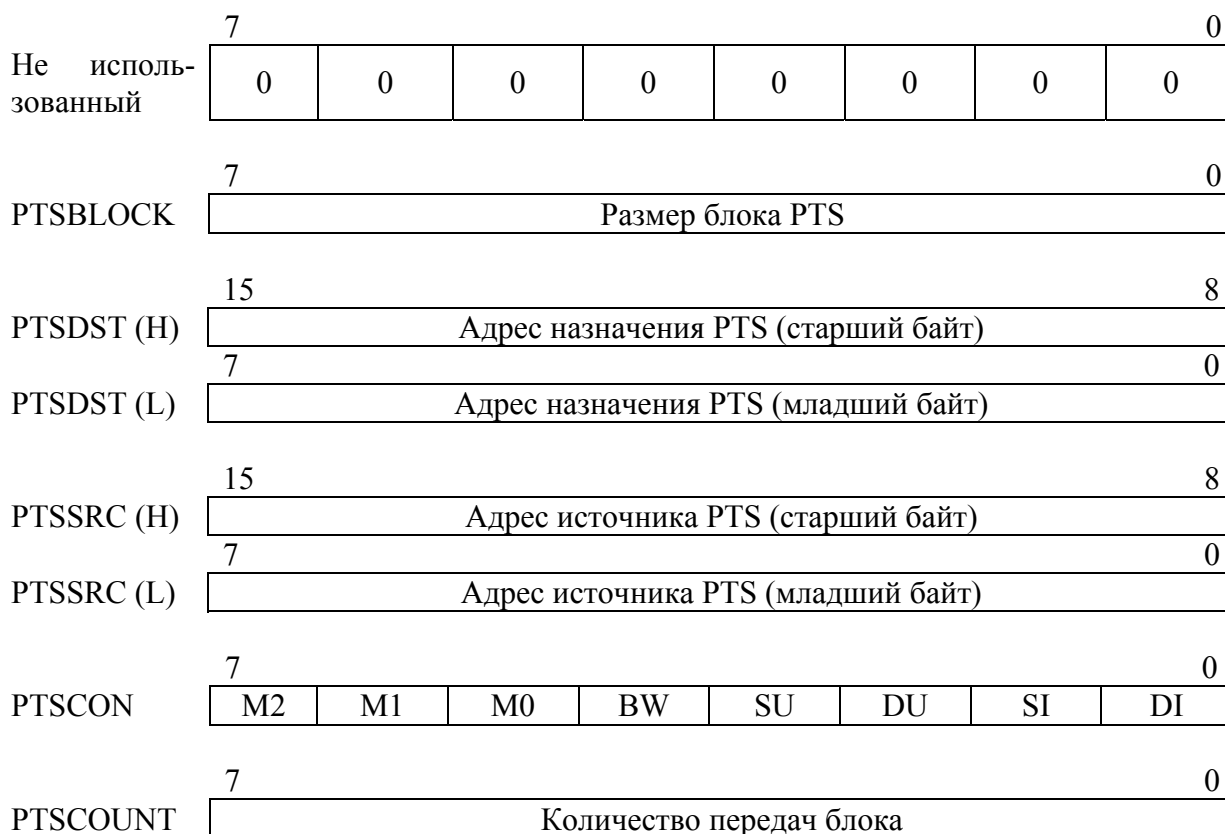


Рисунок 6.17 – Блок управления PTS в режиме передачи блока

Т а б л и ц а 6.17 – Регистры, адреса и функции блока управления PTS в режиме передачи блока

Регистр	Адрес	Функция	
PTSBLOCK	PTSCB +6	Размер блока PTS. Определяет количество байтов или слов в каждом блоке. Действительные значения 1–32	
PTSDST	PTSCB +4	Адрес назначения PTS. В этот регистр записывается адрес ячейки назначения запоминающего устройства. Действительный адрес - любая нерезервированная ячейка запоминающего устройства; однако, это должен быть четный адрес, если выбрана передача слова	
PTSSRC	PTSCB +2	Адрес источника PTS. В этот регистр записывается адрес ячейки источника запоминающего устройства. Действительный адрес – любая незарезервированная ячейка запоминающего устройства; однако, это должен быть четный адрес, если выбрана передача слова	
PTSCON	PTSCB +1	Биты управления PTS	
		M2– M0	Режим PTS. Эти биты выбирают режим PTS: M2 M1 M0 0 0 0 режим передачи блоков.
		BW	Передача байта/слова (0 = передача слова; 1 = передача байта)
		SU	Перезагрузка PTSSRC. 0 перезагружает первоначальный адрес источника PTS после того, как каждая передача блока завершена. 1 сохраняет текущий адрес источника PTS после того, как каждая передача блока завершена
		DU	Перезагрузка PTSDST. 0 перезагружает первоначальный адрес назначения PTS после того, как каждая передача блока завершена 1 сохраняет текущий адрес назначения PTS после того, как каждая передача блока завершена
		SI	Автоинкремент PTSSRC. 0 не увеличивает содержание PTSSRC после того, как каждый байт или слово переданы; 1 увеличивает содержание PTSSRC на «1» после каждого байта или передачи слова
		DI	Автоинкремент PTSDST. 0 не увеличивает содержание PTSDST после того, как каждый байт или слово переданы; 1 увеличивает содержание PTSDST на «1» после каждого байта или передачи слова
PTSCOUNT	PTSCB +0	Последовательная передача блоков. Определяет число блоков, которые будут переданы в течение процедуры поблочной передачи. Передача каждого блока – один цикл PTS. Максимальное число – 255	

Режим АЦ сканирования

В режиме АЦ сканирования PTS заставляет АЦП исполнять многократные преобразования на одном или более каналах и затем хранит результаты в таблице в запоминающем устройстве. На рисунке 6.18 показан блок управления PTS в режиме АЦ сканирования.

Блок управления содержит указатели для регистра AD_RESULT (PTSPTR1) и таблицы команд аналого-цифрового преобразования и результатов (PTSPTR2), регистр управления (PTSCON) и счетчик АЦ преобразований (PTSCOUNT).

Не использованный	7	0	0	0	0	0	0	0	0
Не использованный	7	0	0	0	0	0	0	0	0
PTSPTR2 (H)	15	Указатель 2 значение (старший байт)							8
PTSPTR2 (L)	7	Указатель 2 значение (младший байт)							0
PTSPTR1 (H)	15	Указатель 1 значение (старший байт)							8
PTSPTR1 (L)	7	Указатель 1 значение (младший байт)							0
PTSCON	7	M2	M1	M0	0	UPDT	0	1	0
PTSCOUNT	7	Количество циклов АЦ преобразований							0

Рисунок 6.18 – Блок управления PTS – режим АЦ сканирования

Т а б л и ц а 6.18 – Регистры, адреса и функции блока управления PTS в режиме АЦ сканирования

Регистр	Адрес	Функция	
PTSPTR2	PTSCB +4	Значение указателя 2. Этот регистр содержит адрес регистра результата АЦ преобразования (AD_RESULT)	
PTSPTR1	PTSCB +2	Значение указателя 1. Этот регистр содержит адрес таблицы команд АЦП и результаты преобразований	
PTSCON	PTSCB +1	Биты управления PTS	
		M2:0	Режим PTS. Эти биты выбирают режим PTS. M2 M1 M0 1 1 0 режим сканирования АЦП
		UPDT	Коррекция. 0 перезагружает начальное значение PTSPTR1 после каждого АЦ сканирования; 1 сохраняют текущее значение PTSPTR1 после каждого АЦ сканирования
PTSCOUNT	PTSCB +0	Последовательное аналого-цифровое преобразование. Определяет число АЦ преобразований, которые будут закончены в течение АЦ сканирования. Каждый цикл состоит из передачи результатов АЦ преобразований из PTS в таблицу команд/данных и загрузки новой команды в регистр AD_COMMAND. Максимальный номер – 255	

Используя режим АЦ-сканирования, необходимо установить таблицу команд/данных в запоминающем устройстве (таблица 6.19). Таблица команд/данных содержит АЦ команды, которые записываются в свободные ячейки запоминающего устройства. PTS хранит результаты преобразований в этих свободных ячейках. Только объем свободных ячеек запоминающего устройства ограничивает размер таблицы; они могут храниться во внутреннем или внешнем ОЗУ.

Т а б л и ц а 6.19 – Таблица команд/данных режима АЦ сканирования

Адрес	Содержание	
XXXX + A _H	A/D результат 2	
XXXX + 8 _H	Неиспользованный	A/D команда 3 *
XXXX + 6 _H	A/D результат 1	
XXXX + 4 _H	Неиспользованный	A/D команда 2
XXXX + 2 _H	A/D результат 0 **	
XXXX	Неиспользованный	A/D команда 1
<p>* Записывается 0000_H, чтобы предотвратить новое преобразование в конце процедуры.</p> <p>** Результат АЦ преобразования, которое инициализировало процедуру PTS.</p>		

Чтобы начать режим АЦ сканирования, разрешают прерывание по законченному циклу и разрешают его обслуживание через PTS. Программное обеспечение должно начать первое преобразование. Когда АЦП заканчивает первое преобразование и генерирует прерывание завершения АЦ преобразования, прерывание адресуется к PTSCB и инициирует процедуру АЦ сканирования. PTS хранит результаты преобразования, загружает новую команду в AD_COMMAND и декрементирует значение в PTSCOUNT. При каждом повторном PTS прерывании повторяется цикл просмотра A/D; он сохраняет результаты преобразования, загружает команду следующего преобразования в регистр AD_COMMAND и декрементирует PTSCOUNT. Процедура продолжается до тех пор, пока PTSCOUNT не уменьшится до нуля. Когда это произойдет, аппаратные средства очистят бит разрешения в регистре PTSSEL, который запрещает обслуживание PTS, и установят бит PTSSRV, который запрашивает end-of-PTS прерывание.

PTS циклы в режиме АЦ сканирования

Программное обеспечение начинает первое АЦ преобразование. После того, как прерывание по завершению A/D инициализирует программу PTS, выполняется следующая последовательность инструкций.

1 PTS читает первую команду (из адреса XXXX), загружает ее во временную ячейку и инкрементирует регистр PTSPTR1 дважды. PTSPTR1 теперь указывает на первую свободную ячейку в таблице команд/данных (адрес XXXX + 2).

2 PTS читает регистр AD_RESULT, хранит результаты первого преобразования в ячейке (XXXX + 2) в таблице команд/данных и увеличивает регистр PTSPTR1 на два. PTSPTR1 теперь указывает на (XXXX + 4).

3 PTS загружает команду из временной ячейки в регистр AD_COMMAND. Это завершает первый цикл АЦ сканирования и начинает следующее АЦ преобразование.

4 Если UPDT (PTSCON.3) очищен, первоначальный адрес перезагружается в регистр PTSPTR1. Следующий цикл использует ту же самую команду и перезаписывает предыдущие данные. Если UPDT установлен, то обновленный адрес остается в PTSPTR1 и следующий цикл использует новую команду и хранит преобразованные результаты в новом адресе.

5 Содержимое PTSCOUNT декрементируется, и центральный процессор возобновляет выполнение программы. Когда формируется следующее прерывание по

завершению АЦ преобразования, цикл повторяется. Когда PTSCOUNT достигает нуля, аппаратные средства очищают бит PTSSEL и устанавливают бит PTSSRV, который формирует запрос на прерывание end-of-PTS.

Режим АЦ сканирования (пример 1)

Таблица команд/данных, показанная в таблице 6.20, задаёт ряд аналого-цифровых преобразований, начиная с канала 7 и заканчивая каналом 4. Каждый вход в таблицу – слово (два байта). В таблице 6.21 приводится соответствующий PTSCB.

Программное обеспечение начинает преобразование с канала 7. После завершения преобразования прерывание по завершению АЦ-преобразования инициализирует программу PTS сканирования. Шаг 1 записывает команды из канала 6 во временную ячейку и увеличивает PTSPTR1 до 3002_H. Шаг 2 записывает результат преобразований из канала 7 в ячейку 3002_H и увеличивает PTSPTR1 до 3004_H. Шаг 3 загружает содержимое канала 6 из временной ячейки в AD_COMMAND, чтобы начать следующее преобразование. Шаг 4 обновляет PTSPTR1 (PTSPTR1 теперь указывает на 3004_H), и шаг 5 увеличивает PTSCOUNT до 3. Следующий цикл начинается с сохранения содержимого канала 5 команд во временной ячейке. В течение последнего цикла (PTSCOUNT = 1) «пустая» команда загружается в регистр AD_COMMAND, и преобразование не выполняется. PTSCOUNT – уменьшается до нуля, и происходит запрос на end-of-PTS прерывание.

Т а б л и ц а 6.20 – Таблица команд/данных (пример 1)

Адрес	Содержание	
300E _H	AD_RESULT для ACH4	
300C _H	Не используется	0000 _H (холостая команда)
300A _H	AD_RESULT для ACH5	
3008 _H	Не используется	AD_COMMAND для ACH4
3006 _H	AD_RESULT для ACH6	
3004 _H	Не используется	AD_COMMAND для ACH5
3002 _H	AD_RESULT для ACH7	
3000 _H	Не используется	AD_COMMAND для ACH6

Т а б л и ц а 6.21 – Режим АЦ сканирования PTSCB (пример 1)

Не используется
Не используется
PTSPTR2 (H) = 1F _H
PTSPTR2 (L) = AA _H
PTSPTR1 (H) = 30 _H
PTSPTR1 (L) = 00 _H
PTSCON = CB _H (Режим = 110, UPDT = 1)
PTSCOUNT = 04 _H

Режим АЦ сканирования (пример 2)

В таблице 6.23 приводится серия из десяти циклов PTS, каждый из которых читает один канал АЦП и записывает результат в ячейку (3002_H). UPDT бит (PTSCON.3) очищается так, чтобы после цикла было восстановлено первоначальное содержание PTSPTR1. Таблица команд/данных представлена в таблице 6.22.

Т а б л и ц а 6.22 – Таблица команд/данных (пример 2)

Адрес	Содержание	
3002 _H	AD_RESULT for ACHx	
3000 _H	Не используется	AD_COMMAND для ACHx

Т а б л и ц а 6.23 – Режим просмотра АЦ сканирования PTSCB (пример 2)

Не используется
Не используется
PTSPTR2 (H) = 1F _H
PTSPTR2 (L) = AA _H
PTSPTR1 (H) = 30 _H
PTSPTR1 (L) = 00 _H
PTSCON = C3 _H (Режим = 110, UPDT = 0)
PTSCOUNT = 0A _H

Программа начинает преобразование на канале (x). Когда преобразование закончено и сгенерировано прерывание завершения АЦ преобразования, начинается режим АЦП сканирования. PTS считывает команду в ячейке 3000_H и записывает её во временную ячейку. Затем увеличивает регистр PTSPTR1 на два и записывает значение регистра AD_RESULT в ячейку 3002_H. Заключительный шаг – копирование команды преобразования из временной ячейки в регистр AD_COMMAND. Центральный процессор обработает или перенесет данные результатов преобразования из таблицы прежде, чем следующее преобразование закончится и начнется новый цикл PTS. Когда начинается следующий цикл, PTSPTR1 снова обращается в 3000_H, и повторяются шаги первого цикла. Значение регистра AD_RESULT записывается в ячейку 3002_H, и команда из ячейки 3000_H выполняется снова.

Режим последовательного ввода-вывода

Микроконтроллер не имеет последовательного порта ввода-вывода, но режимы последовательного ввода-вывода PTS обеспечивают программируемый канал последовательного ввода-вывода для синхронной и асинхронной передачи и приема. Есть четыре основных режима работы: синхронная передача, синхронный прием, асинхронная передача и асинхронный прием. Стандартный сигнал порта ввода-вывода конфигурируется как для передачи данных (TXD), так и для получения данных (RXD). Канал EPA устанавливает скорость двоичной передачи (baud rate). Возможно конфигурирование любого сигнала порта 2 как TXD или как RXD. В синхронных режимах канал EPA может или производить последовательные синхроимпульсы (SCK) сигнала, или использовать внешний синхросигнал. Может быть передано и принято до 16 битов, включая биты четности и останова в асинхронных режимах. Последовательные режимы ввода-вывода требуют два блока управления PTS для формирования всех вариантов (см. рисунки 6.19 и 6.20). Эти блоки не обязательно должны быть смежными, но каждый из них должен располагаться в границах четырех слов в регистровом файле.

PTS блок управления 1 в режиме последовательного ввода-вывода

Блок управления 1 PTS содержит указатели на второй блок управления PTS (PTSVEC) и регистр времени EPA, который устанавливает скорость двоичной передачи (EPAREG). Он содержит также 16-битную величину, используемую для вычисления скорости двоичной передачи – регистр управления (PTSCON) и счетчик последовательных циклов PTS (PTSCOUNT).

PTSVEC (H)	15	8					
	SIO PTSCB2 Указатель основного адреса (старший байт)						
PTSVEC (L)	7	0					
	SIO PTSCB2 Указатель основного адреса (младший байт)						
BAUD (H)	15	8					
	Значение BAUD (старший байт)						
BAUD (L)	7	0					
	Значение BAUD (младший байт)						
EPAREG (H)	15	8					
	Адрес регистра EPA (старший байт)						
EPAREG (L)	7	0					
	Адрес регистра EPA (младший байт)						
PTSCON	7	0					
	M2	M1	M0	SA1	0	0	SA0
PTSCOUNT	7	0					
	Последовательные циклы PTS (счётчик)						

Рисунок 6.19 – Режим управления блоком 1 последовательного ввода-вывода PTS

Т а б л и ц а 6.24 – Регистры, адреса и функции PTS в режиме управления блоком 1 последовательного ввода-вывода PTS

Регистр	Адрес	Функция
1	2	3
PTSVEC	PTSCB1 + 6	Указатель базового адреса SIO PTSCB2. Этот регистр содержит основной адрес второго блока управления PTS для последовательного режима ввода-вывода
BAUD	PTSCB1 + 4	Значение BAUD. Этот регистр содержит 16-битную величину, используемую PTS для задания желаемой скорости передачи. Формула для вычисления значения для загрузки регистра BAUD: $\text{Baud_value} = \frac{F_{\text{BQ1}}}{\text{Multiplier}(\text{Baud_rate} \times \text{EPA_prescale})}$ где Baud_value – целое 16-битное число, которое загружено в регистр BAUD; F _{BQ1} – входная частота на BQ1, в МГц; Multiplier равно четырем в асинхронных режимах и равно восьми в синхронных режимах; Baud_rate – скорость передачи в битах в секунду; EPA_prescale – число предварительного деления частоты таймера EPA, 1–64
EPAREG	PTSCB1 + 2	Адрес регистра времени EPA. Этот регистр содержит 16-битный адрес регистра COMPx_TIME или EPAx_TIME

Окончание таблицы 6.24

1	2	3	
PTSCON	PTSCB1 + 2	Биты управления PTS	
		M2–M0	Режим M2 M1 M0 0 0 1 PTS SIO режим приема 0 1 1 SIO режим передачи
		SA1, SA0	Асинхронный/синхронный режим. SA1 SA0 * 0 0 разрешает асинхронные режимы последовательного ввода-вывода 1 1 разрешает синхронные режимы последовательного ввода-вывода
		MAJ	Мажоритарная выборка. 0 запрещает мажоритарную выборку в асинхронном режиме приёма; всегда очищен во всех других режимах; 1 позволяет мажоритарную выборку в асинхронном режиме приёма
PTSCOUNT	PTSCB1 + 0	Последовательные циклы PTS. Определяет число битов, которые будут переданы или получены, включая биты четности и останова, но без старт-бита. Для асинхронных режимов программируют значения от 1 до 16. Для синхронных режимов программируют удвоенное количество передаваемых (принимаемых) битов – от 2 до 32. PTSCOUNT декрементируется в конце каждого цикла PTS. Когда он достигает нуля, аппаратные средства очищают соответствующий бит PTSSEL и устанавливают бит PTSSRV, который разрешает end-of-PTS прерывание	
* В оба бита всегда заносят одинаковые значения.			

PTS блок управления 2 в режиме последовательного ввода-вывода

Блок управления 2 PTS содержит указатели и на регистр порта (PORTREG), и на регистр данных (DATA). Он также содержит 16-битную величину, которая используется для вычисления времени выборки для асинхронного приема при мажоритарной выборке (SAMPTIME), регистр управления (PTSCON1) и 16-битное значение, используемое для выбора разряда порта в качестве TXD или RXD сигнала (PORTMASK).



Рисунок 6.20, лист 1 – Режим управления блоком 2 последовательного ввода-вывода PTS

PTSCON1 (синхронный)	7	0	0	0	0	0	0	TRC	0	0	
PTSCON1 (асинхронный)	7	0	RPAR	PEN	0	0	0	FE	TPAR	0	
PORTMASK	7	Регистр маски порта									0
PORTREG (H)	15	Указатель адреса порта (старший байт)								8	
PORTREG (L)	7	Указатель адреса порта (младший байт)								0	

Рисунок 6.20, лист 2

Т а б л и ц а 6.25 – Регистры, адреса и функции PTS в режиме управления блоком 2 последовательного ввода-вывода PTS

Регистр	Адрес	Функция
1	2	3
SAMPTIME	PTSCB2 + 6	<p>Значение времени выборки.</p> <p>Этот регистр управляет временем между выборками в течение режима асинхронного приема при мажоритарной выборке. Для вычисления значения, загружаемого в регистр SAMPTIME, используют следующую формулу</p> $\text{Sample_time} = \frac{T_{\text{SAM}} \times F_{\text{BQ1}}}{2} - 9,$ <p>где Sample_time – целое число (1–31), загружаемое в регистр SAMPTIME; T_{SAM} – желательное время между выборками, в мкс; F_{BQ1} – входная частота на BQ1, в МГц</p>
DATA	PTSCB2 + 4	<p>Регистр данных.</p> <p>16-битный регистр содержит данные для передачи или данные, которые будут приняты. Во время передачи младший (нулевой) бит передается первым. Данные сдвигаются вправо с каждой успешной передачей. В течение приема первый бит загружается в старший (15-й) разряд. Данные сдвигаются вправо с каждым успешным приемом</p>
PTSCON1	PTSCB2 + 3	Биты управления PTS.
		Синхронный режим
		<p>TRC</p> <p>Управление режимом передачи/приема. 0 передают или получают данные в течение четных циклов PTS; 1 передают или получают данные в течение нечетных циклов PTS. Этот бит инициализируют при старте каждой передачи или приема</p>

Окончание таблицы 6.25

1	2	3
		Асинхронный режим
		RPAR Статус и контроль четности при приеме. Инициализируется перед началом приема. 0 бит TPAP установлен для проверки на четность; 1 бит TPAP очищен для проверки на нечетность
		PEN Разрешение контроля чётности. 0 запрет контроля чётности, 1 разрешение контроля чётности
		FE Флаг ошибки кадрирования. 0 бит останов 1, 1 бит останов 0. Очищать бит в начале каждого приема
		TPAR Управление чётностью передачи. 0 четность, 1 нечетность
PORTMASK	PTSCB2 + 2	Регистр маски порта. Выбирает сигнал порта, который будет функционировать на передачу данных (TXD) или на прием данных (RXD), установкой соответствующего бита
PORTREG	PTSCB2 + 0	Указатель адреса порта. 16 битный регистр, содержащий адрес порта, который будет использоваться при приеме и передаче данных

Пример режима синхронной передачи

В синхронном режиме передачи SSIO канал EPA управляет скоростью двоичной передачи, генерируя или захватывая сигнал синхроимпульсов SCK. Чтобы генерировать SCK сигнал, канал EPA конфигурируется в режиме сравнения, и выставляется опция переключения вывода. Если внешний источник задаёт SCK сигнал, канал EPA конфигурируется в режиме захвата по каждому фронту. В этом случае EPA канал генерирует прерывание по каждому переключению сигнала SCK. По каждому прерыванию EPA периферийный сервер PTS сдвигает бит данных на выход порта, который сконфигурирован как TXD. PTSCON1 (рисунок 6.19) задаёт передачу данных в чётных или нечётных циклах PTS. Поскольку передачи происходят только по переднему или заднему фронту синхроимпульсов, два цикла PTS требуются для передачи каждого бита данных (рисунок 6.21). Требуется 16 циклов PTS, чтобы передать восемь битов данных. В SSIO режиме передачи только биты данных могут быть переданы; биты чётности и стоп-биты исключены.

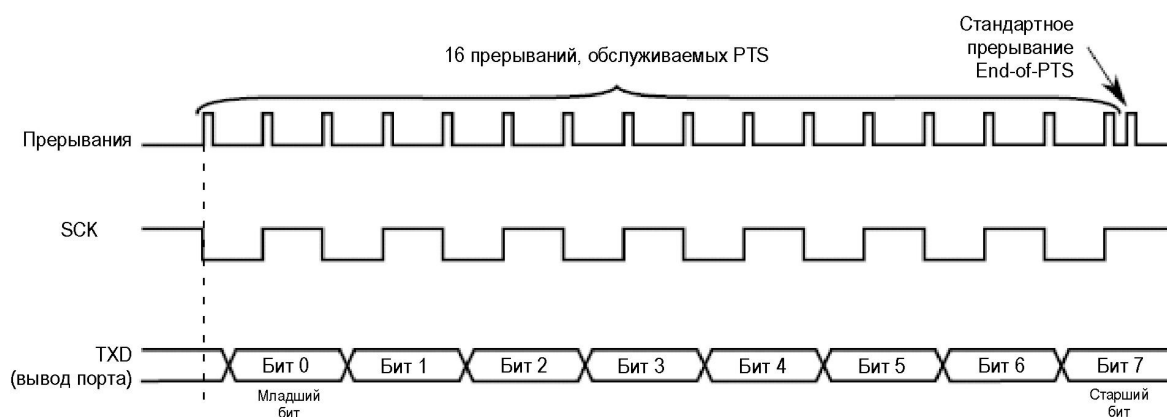


Рисунок 6.21 – Временные диаграммы синхронной последовательной передачи

Если SCK сигнал генерирован каналом EPA, первый цикл PTS должен быть начат следующим образом:

- вывод TXD порта и SCK сигнал инициализируются к требуемому системой логическому уровню перед началом передачи;

- добавляется содержание регистра таймера к Baud_value (рисунок 6.19) и результат сохраняется в регистре времени EPA. Это устанавливает время первого прерывания и передача первого разряда осуществляется с нужной двоичной скоростью.

Следующий пример использует CAPCOMP0 (P2.0) для управления скоростью двоичной передачи и выводом SCK сигнала и P2.2 для вывода данных (TXD). Он устанавливает процедуру синхронного последовательного ввода-вывода PTS, которая передает 16 байтов с восемью битами данных со скоростью 9600 бод. Этот пример использует несколько определенных пользователем регистров. T_COUNT определяет количество байтов для передачи, а TXDDONE – флаг, который устанавливается, когда все байты переданы.

1 Запрет прерываний и PTS.

Используется инструкция DI для запрета стандартных прерываний и инструкция DPTS для запрета PTS.

2 Установка указателя стека.

3 Сброс всех регистров маски прерываний.

Очистка INT_MASK, INT_MASK1 и PI_MASK.

4 Инициализация вывода P2.0 как выход CAPCOMP0 (SCK) и P2.2 как выход CAPCOMP2 (TXD).

Очистка P2_DIR (выбор вывода).

Установка P2_MODE.0 (выбор специальной функции).

Очистка P2_MODE.2 (выбор функции LSIO).

Установка битов 0 и 2 P2_REG инициализирует выходы TXD и SCK в «1».

5 Инициализация и разрешение таймера. Выбирают прямой счет, внутреннюю синхронизацию и запрещают предварительное деление частоты.

Установка битов 6 и 7 T1CONTROL (рисунок 10.8).

6 Инициализация PTSCB как показано в таблице 6.26.

Т а б л и ц а 6.26 – SSIO режим передачи PTSCBs

PTSCB1	PTSCB2
PTSVEC (H) = указатель на PTSCB2	Неиспользованный
PTSVEC (L) = указатель на PTSCB2	SAMPTIME = неиспользованный
BAUD (H) = 00 _H (9600 бод в 16 МГц)	DATA (H) = неиспользованный
BAUD (L) = D0 _H (9600 бод в 16 МГц)	DATA (L) = nn _H (8 битов данных)
EPAREG (H) = 1F _H (EPA0_TIME)	PTSCON1 = 02 _H (передача данных по нечетным циклам PTS)
EPAREG (L) = 42 _H (EPA0_TIME)	PORTMASK = 04 _H (P2.2 = TXD)
PTSCON = 72 _H (режим передачи SSIO)	PORTREG (H) = 1F _H (P2_REG)
PTSCOUNT = 10 _H (8 бит данных × 2)	PORTREG (L) = D4 _H (P2_REG)

7 Разрешение EPA0 прерывания.

Установка INT_MASK.2.

8 Загрузка количества передаваемых байтов в счетчик T_COUNT и очистка флага окончания передачи (TXDDONE).

LD T_COUNT, #16.

CLRB TXDDONE.

9 Выбор PTS обслуживания для CAPCOMP0.

Установка PTSSEL.2.

10 Установка CAPCOMP0 как канала только сравнения.

Установка EPA0_CON.6.

11 Начать операции канала CAPCOMP0, записав время первого прерывания в EPA0_TIME. Для установки корректного значения прибавляют baud_value (0D0_H) к текущему значению TIMER1 и сохраняют результат в EPA0_TIME. Baud_value определяет время первого PTS прерывания и первое переключение сигнала SCK. PTS передает первый бит данных по первому фронту SCK в этом примере. Baud_value = 0D0_H выбирает скорость двоичной передачи 9600 бод.

12 Разрешение PTS и стандартных прерываний.

Используют инструкцию EI для разрешения всех стандартных прерываний и инструкцию EPTS для разрешения PTS прерывания.

13 Начало передачи. Данные сдвигаются, начиная с младшего (самого правого) бита. Каждый раз, когда происходит совпадение между EPA0_TIME и TIMER1, канал CAPCOMP0 производит прерывание и переключает SCK сигнал. PTS выводит следующий бит данных на вывод, функционирующий как TXD по нечетным циклам PTS. Когда PTSCOUNT уменьшится до нуля, PTS вызывает end-of-PTS прерывание (рисунок 6.22). Процедура обслуживания прерывания запрещает канал EPA, потому что в конце цикла PTS следующее время переключения SCK загружается в регистр времени события. Если переключение происходит, полярность синхроимпульсов изменится из-за нечетного числа переключений. Процедура обслуживания прерывания должна загрузить следующий байт данных, перезагрузить PTSCOUNT и PTSCON1, выбрать обслуживание PTS для CAPCOMP0 и регистров EPA0_TIME и EPA0_CONTROL.

14 Чтобы определить, когда все байты переданы, надо создать циклическую подпрограмму, чтобы проверить состояние TXDDONE флага.



Рисунок 6.22 – Режим синхронного последовательного вывода. Блок-схема прерывания end-of-PTS

Пример режима синхронного приёма

В режиме синхронного приёма данных канал ЕРА управляет двоичной скоростью приёма, генерируя или захватывая синхроимпульсы (сигнал SCK). Для генерации SCK сигнала канал ЕРА формируется для сравнения. Всякий раз, когда происходит совпадение между регистром события-времени ЕРА и регистром таймера, канал ЕРА переключает SCK и производит прерывание. Если SCK сигнал обеспечивается внешним источником, канал ЕРА конфигурируется в режиме захвата с захватом по любому фронту. По каждому следующему ЕРА прерыванию PTS принимает бит данных с входа порта, функционирующего как RXD. PTSCON1 (рисунок 6.19) управляет режимом приёма (по чётным или нечётным циклам PTS). Поскольку прием происходит только по передним или задним фронтам синхроимпульсов, два цикла PTS требуется для приёма каждого бита данных (рисунок 6.23). Требуется 16 циклов PTS, чтобы получить восемь битов данных. SSIO прием не включает биты чётности или стоп-биты.

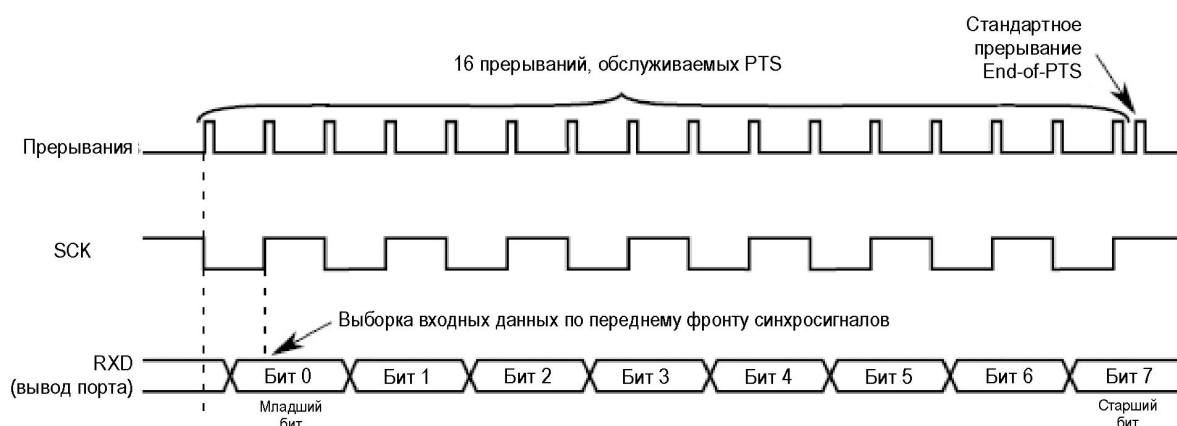


Рисунок 6.23 – Временные диаграммы синхронного последовательного приёма

Если SCK сигнал генерируется каналом ЕРА, первый цикл PTS должен быть начат следующим образом:

- вход RXD и сигнал SCK инициализируются к требуемому системой логическому уровню;

- прибавить содержимое регистра таймера к Baud_value (рисунок 6.19) и сохранить результат в регистре времени ЕРА. Это устанавливает время для первого прерывания и появление первого прерывания в соответствии со скоростью приёма данных.

В следующем примере CAPCOMP0 используется для захвата сигнала SCK, а P2.3 (RXD) – для получения данных. Процедура синхронного последовательного ввода PTS настроена на приём 16 байтов с восемью битами данных. Поскольку этот пример использует внешний последовательный ввод синхроимпульсов, TIMER1 и регистр BAUD не используются. Внешний источник синхроимпульсов управляет скоростью приёма. В этом примере используется несколько определяемых пользователем регистров. R_COUNT определяет количество принимаемых байтов, а RXDDONE – флаг, который устанавливается после получения всех байтов.

- 1 Запретить прерывания и PTS.

Использовать инструкцию DI, чтобы запретить все стандартные прерывания и инструкцию DPTS, чтобы запретить PTS.

- 2 Установить указатель стека.

- 3 Сбросить все регистры маски прерывания.

Очистить INT_MASK, INT_MASK1 и PI_MASK.

- 4 Инициализировать P2.0 для функционирования как вход CAPCOMP0 (SCK) и P2.3 для функционирования как RXD.

Установить биты 0 и 3 P2_DIR (выбирает ввод).

Установить P2_MODE.0 (выбирает специальную функцию).

- Очистить P2_MODE.3 (выбирает функцию LSIO).
- Установить биты 0 и 3 P2_REG (инициализирует SCK и RXD в «1»).
- 5 Инициализировать PTSCB как показано в таблице 6.27.

Т а б л и ц а 6.27 –Режим SSIO приёма PTSCBs

PTSCB1	PTSCB2
PTSVEC (H) = указатель на PTSCB2	Неиспользованный
PTSVEC (L) = указатель на PTSCB2	SAMPTIME = неиспользованный
BAUD (H) = неиспользованный	DATA (H) = неиспользованный
BAUD (L) = неиспользованный	DATA (L) = 00 _H (очистка регистра, чтобы получить данные)
EPAREG (H) = 1F _H (EPA0_TIME)	PTSCON1 = 00 _H (получение данных по чётным циклам PTS)
EPAREG (L) = 42 _H (EPA0_TIME)	PORTMASK = 08 _H (P2.3 = RXD)
PTSCON = 32 _H (режим SSIO приёма)	PORTREG (H) = 1F _H (P2_REG)
PTSCOUNT = 10 _H (8 бит данных x 2)	PORTREG (L) = D4 _H (P2_REG)

- 6 Разрешить прерывание CAPCOMP0.
- Установить INT_MASK.2.

7 Загрузить число байтов к передаче в определённый пользователем регистр счета передачи (R_COUNT) и очистить определенный пользователем флаг завершения приема (RXDDONE).

- LD R_COUNT, #16.
- CLRB RXDDONE.

- 8 Выбрать PTS-сервис для CAPCOMP0.
- Установить PTSSEL.2.

- 9 Установить CAPCOMP0 для захвата и передних, и задних фронтов.
- Установить биты 4 и 5 EPA0_CON (рисунок 10.10).

- 10 Установить PTS и стандартные прерывания.

Использовать инструкцию EI для разрешения всех стандартных прерываний и EPTS инструкцию для разрешения PTS.

11 Переключить вход SCK, чтобы начать прием. Данные перемещаются в наиболее значащий (крайний левый) бит и сдвигаются вправо с каждым полученным битом. EPA вырабатывает прерывание при каждом переключении на входе SCK. PTS получает следующий бит данных на вход, сконфигурированный как RXD, в четных циклах PTS. Когда PTSCOUNT декрементируется до нуля, PTS вызывает end-of-PTS прерывание (рисунок 6.24). Процедура обслуживания прерываний должна запретить канал EPA, очистить регистр данных (DATA), перезагрузить регистры PTSCOUNT и PTSCON1, перезагрузить EPA0_CON и выбрать обслуживание PTS для CAPCOMP0. Если EPA генерирует сигнал SCK, процедура обслуживания end-of-PTS прерывания также должна будет перезагрузить регистр EPA0_TIME.

12 Чтобы определить, когда все байты были переданы, надо создать циклическую подпрограмму, чтобы проверить состояние RXDDONE флага.

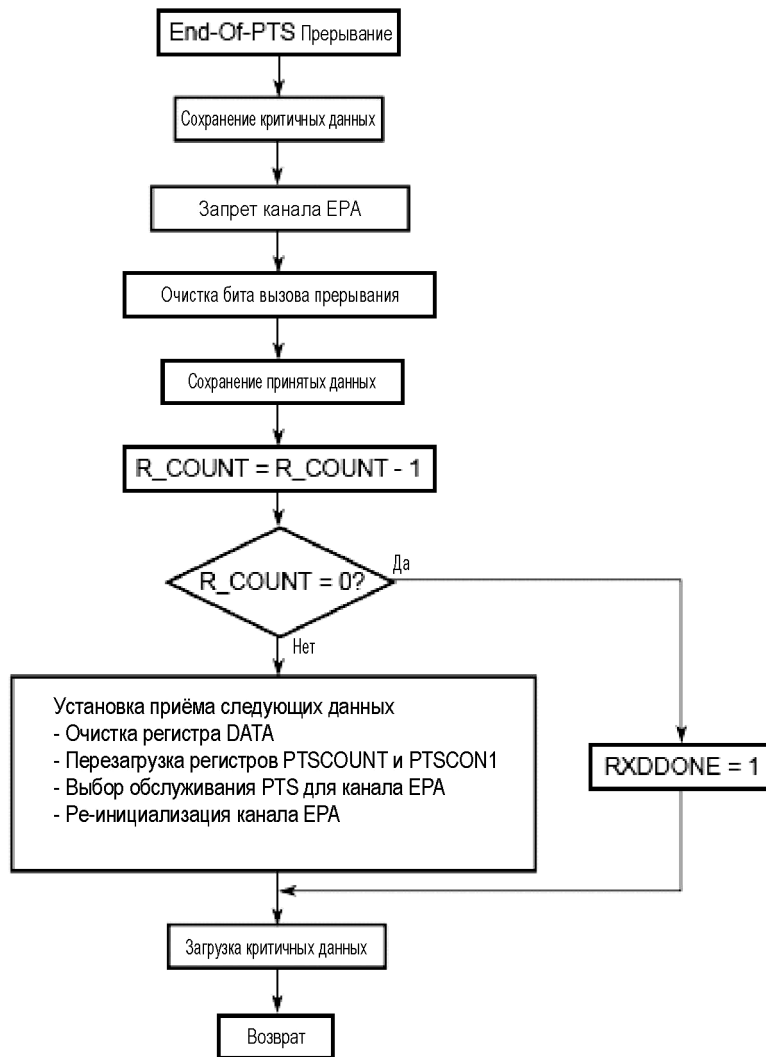


Рисунок 6.24 – Режим синхронного SIO приёма. Блок-схема процедуры прерывания end-of-PTS

Пример режима асинхронной SIO передачи

В режиме асинхронной передачи (ASIO) канал EPA управляет скоростью двоичной передачи, вырабатывая прерывание всякий раз, когда происходит совпадение между регистром времени события EPA и регистром таймера. PTS сдвигает бит данных на вывод порта, сконфигурированного для функционирования как сигнал передачи данных (TXD), когда выбранный канал EPA производит прерывания сравнения (рисунок 6.25). В режиме передачи ASIO PTS автоматически передает до 16 битов (данные + 1 опциональный бит четности + 1 стоп-бит). Максимальное число битов данных – 14 с паритетом или 15 без.

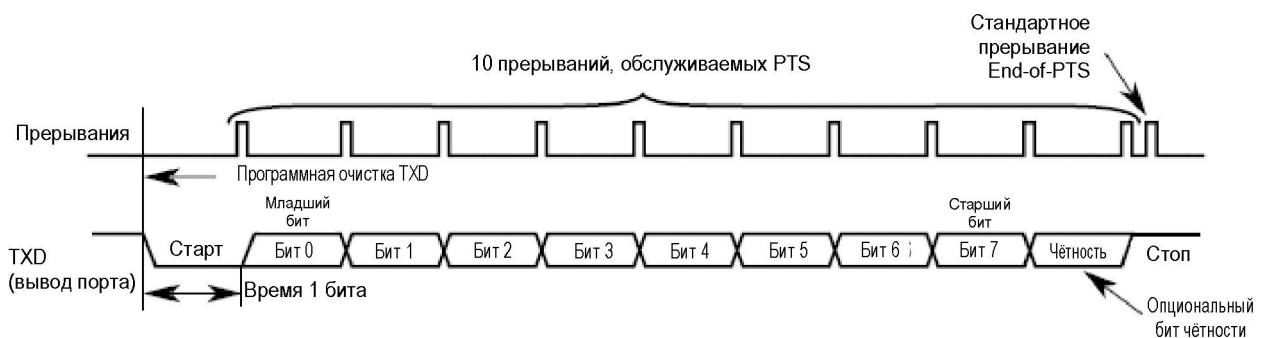


Рисунок 6.25 – Временная диаграмма асинхронной передачи SIO

Первый цикл PTS должен быть начат путем генерации стартового бита с последующей установкой времени для первого EPA прерывания:

- инициализировать вывод порта TXD в единицу перед стартом передачи;
- записать ноль в вывод порта TXD, чтобы начать передачу. PTS использует это как стартовый бит;

- добавить содержимое регистра таймера к Baud_value (рисунок 6.19) и сохранить результат в регистре времени EPA. Это устанавливает время для первого прерывания и задаёт надлежащую скорость передачи первого бита.

Следующий пример использует P2.0 как вывод данных (TXD), а CAPCOM0 для управления скоростью двоичной передачи. Он устанавливает процедуру асинхронного последовательного вывода PTS, которая передает 16 байтов с восемью битами данных, одним паритетным битом и одним стоп-битом со скоростью 9600 бод. В этом примере используется несколько определяемых пользователем регистров. T_COUNT определяет число байтов для передачи, а TXDDONE – флаг, который устанавливается, когда все байты переданы.

1 Запретить прерывания и PTS.

Использовать инструкцию DI, чтобы запретить все стандартные прерывания и инструкцию DPTS, чтобы запретить PTS.

2 Установить указатель стека.

3 Сбросить все регистры маски прерывания. Очистить INT_MASK, INT_MASK1 и PI_MASK.

4 Инициализировать P2.0 как TXD:

- очистить P2_DIR.0 (выбор вывода);
- очистить P2_MODE.0 (выбор функции LSIO);
- установить P2_REG.0 (инициализирует TXD вывод в «1»).

5 Инициализировать и разрешить таймер; выбрать прямой счет, внутренние синхроимпульсы и запретить делитель:

- установить биты 6 и 7 T1CONTROL (рисунок 10.8).

6 Инициализировать PTSCB как показано в таблице 6.28.

Т а б л и ц а 6.28 – Режим передачи ASIO PTSCB

PTSCB1	PTSCB2
PTSVEC (H) = указатель на PTSCB2	Неиспользуемый
PTSVEC (L) = указатель на PTSCB2	SAMPTIME = неиспользуемый
BAUD (H) = 01 _H (9600 бод в 16 МГц)	DATA (H) = неиспользуемый
BAUD (L) = A0 _H (9600 бод в 16 МГц)	DATA (L) = NN _H (8 битов данных)
EPAREG (H) = 1F _H (EPA0_TIME)	PTSCON1 = 21 _H (разрешить контроль нечетности)
EPAREG (L) = 42 _H (EPA0_TIME)	PORTMASK = 01 _H (P2.0 = TXD)
PTSCON = 60 _H (ASIO режим передачи)	PORTREG (H) = 1F _H (P2_REG)
PTSCOUNT = 0A _H (8 битов данных, 1 бит четности и 1 стоп-бит)	PORTREG (L) = D4 _H (P2_REG)

7 Разрешить прерывание CAPCOMP0:

- установить INT_MASK.2.

8 Загрузить число байтов для передачи в определённый пользователем регистр счета передачи (T_COUNT), и очистить определённый пользователем флаг окончания передачи (TXDDONE):

- LD T_COUNT, #16;
- CLRB TXDDONE.

9 Выбрать PTS-сервис для CAPCOMP0:

- установить PTSSEL.2.

10 Установить стартовый бит передачи:

- очистить P2.0.

11 Установить CAPCOMP0 как канал только сравнения:

- установить EPA0_CON.6 (рисунок 10.10).

12 Начать работу CAPCOMP0 канала, записав время первого прерывания в EPA0_TIME. Для установки правильного значения прибавить baud_value (1A0H) к текущему значению TIMER1 и сохранить результат в EPA0_TIME. Baud_value определяет время первого PTS прерывания. Когда прерывание происходит, PTS передает первый бит данных. Baud_value = 1A0_H выбирает скорость двоичной передачи 9600 бод.

13 Разрешить PTS и стандартные прерывания:

- использовать инструкцию EI для разрешения всех стандартных прерываний и EPTS инструкцию для разрешения PTS.

14 Передача стартует. Данные сдвигаются, начиная с наименее значимого (самого правого) бита. Каждый раз, когда происходит совпадение в таймере между EPA0_TIME и TIMER1, CAPCOMP0 канал производит прерывание, и PTS выводит следующий бит данных на вывод, сконфигурированный как TXD. Когда PTSCOUNT декрементируется до нуля, PTS вызывает прерывание end-of-PTS. Процедура обслуживания прерываний должна загрузить следующий байт данных, перезагрузить регистры PTSCOUNT и PTSCON1, очистить TXD бит, чтобы создать стартовый бит для следующего слова, которое будет передано, выбрать обслуживание PTS для CAPCOMP0 и перезагрузить регистры EPA0_TIME и EPA0_CONTROL.

15 Чтобы определить когда все байты будут переданы, надо создать циклическую подпрограмму, чтобы проверять состояние TXDDONE флага.

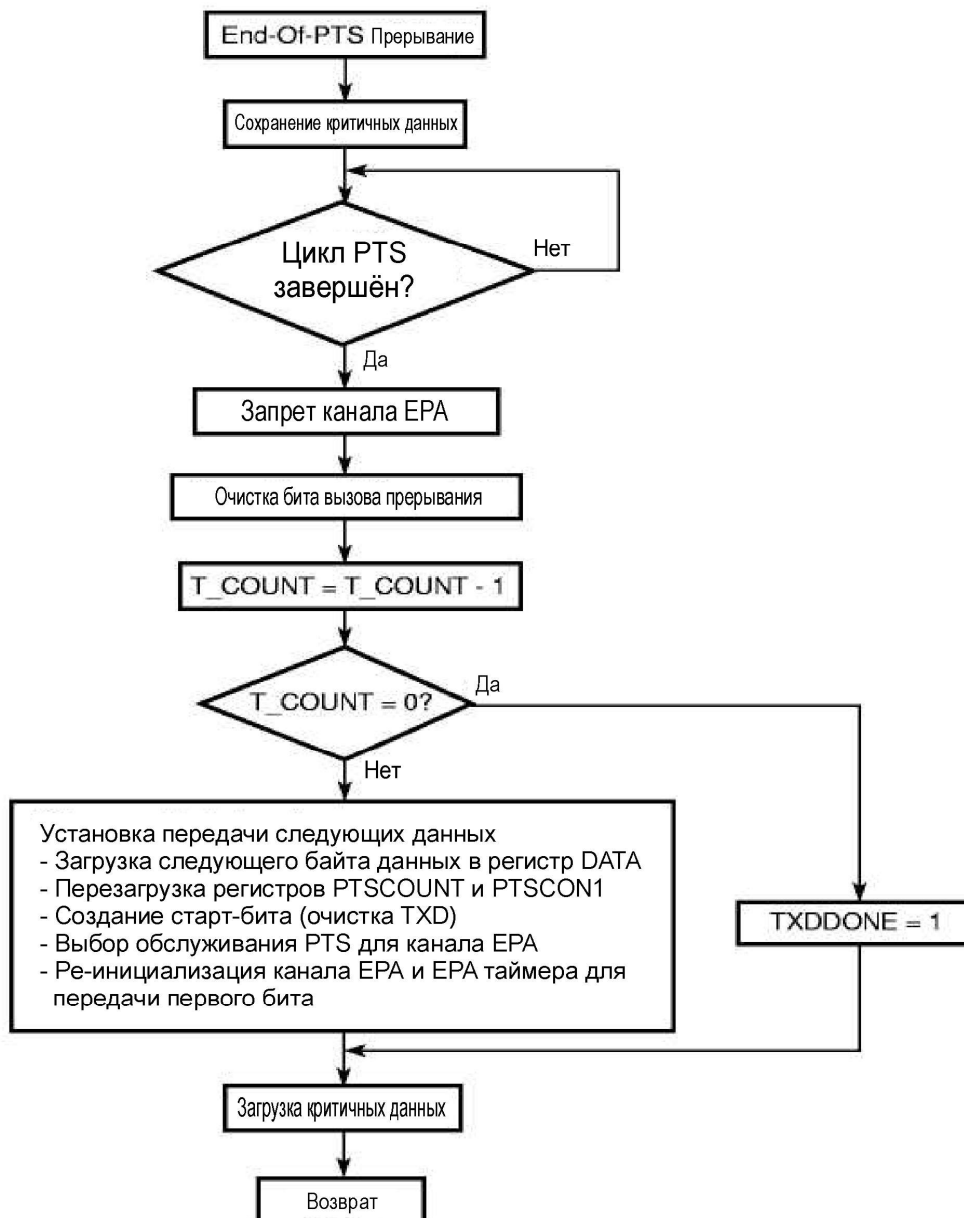


Рисунок 6.26 – Режим асинхронной SIO передачи. Блок-схема процедуры прерывания end-of-PTS

Пример режима асинхронного SIO приёма

В режиме асинхронного приёма (ASIO) канал EPA перехватывает задний фронт старт-бита данных на входе, функционирующем как RXD. При перехвате EPA генерирует стандартное прерывание, стартующее процесс асинхронного приёма. Процедура обслуживания для стандартного прерывания та же, что и для прерывания end-of-PTS. Она переводит канал EPA в режим сравнения и выставляет время следующего сравнения – время полутора битов и разрешает PTS. Через время полутора битов с начала старт-бита первый цикл PTS принимает входные данные с RXD и сдвигает их в регистр DATA (рисунок 6.27). Если разрешена мажоритарность, производится дополнительная выборка. Если две выборки различны, производится третья, чтобы определить, какая из первых двух верна. Регистр SAMPTIME (см. рисунок 6.19) задаёт временной интервал между выборками. Мажоритарность существенно увеличивает время выполнения цикла PTS (см. таблицу 6.4).

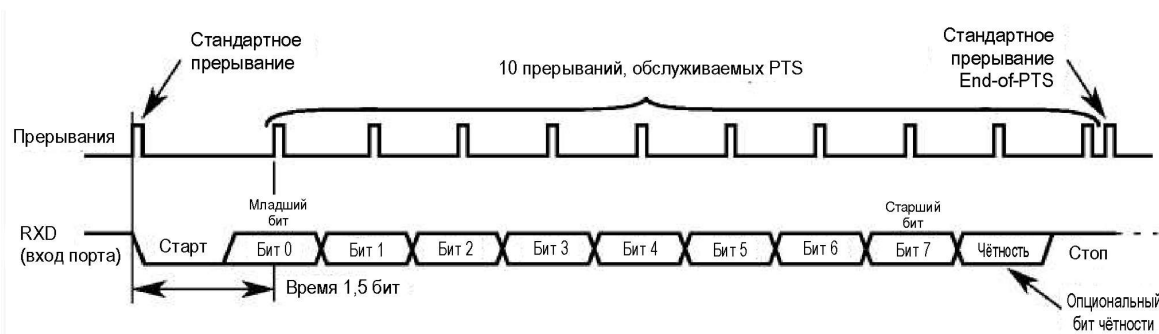


Рисунок 6.27 – Временная диаграмма асинхронного приёма SIO

Первый цикл PTS должен запущен следующим образом:

- вход RXD и сигнал SCK инициализируются к требуемому системой логическому уровню;

- прибавить содержимое регистра таймера к `Baud_value` (рисунок 6.19) и сохранить результат в регистре времени `EPA`. Это устанавливает время для первого прерывания и появление первого прерывания в соответствии со скоростью приёма данных.

В следующем примере `CAPCOMP0` используется для захвата старт-бита, а `P2.3 (RXD)` – для получения данных. Процедура синхронного последовательного ввода PTS настроена на приём 16 байтов с восемью битами данных и битом чётности каждый. В этом примере используется несколько определяемых пользователем регистров. `R_COUNT` определяет количество принимаемых байтов, а `RXDDONE` – флаг, который устанавливается после получения всех байтов.

1 Запретить прерывания и PTS:

- использовать инструкцию `DI`, чтобы запретить все стандартные прерывания и инструкцию `DPTS`, чтобы запретить PTS.

2 Установить указатель стека.

3 Сбросить все регистры маски прерывания:

- очистить `INT_MASK`, `INT_MASK1` и `PI_MASK`.

4 Инициализировать `P2.0` как RXD:

- очистить `P2_DIR.0` (выбор входа);

- очистить `P2_MODE.0` (выбор функции LSIO);

- установить `P2_REG.0` (инициализирует RXD вывод в «1»).

5 Инициализировать и разрешить таймер; выбрать прямой счет, внутреннюю синхронизацию и запретить предделитель.

Установить биты 6 и 7 в `T1CONTROL` (рисунок 10.8).

6 Инициализировать `PTSCB` как показано в таблице 6.29.

Т а б л и ц а 6.29 – PTSCBs в режиме приёма ASIO PTSCBs

PTSCB1	PTSCB2
<code>PTSVEC (H) =</code> указатель на <code>PTSCB2</code>	Неиспользованный
<code>PTSVEC (L) =</code> указатель на <code>PTSCB2</code>	<code>SAMPTIME = 01_H</code>
<code>BAUD (H) = 01_H</code>	<code>DATA (H) = 00_H</code> (очистка регистра для приёма данных)
<code>BAUD (L) = A0_H</code>	<code>DATA (L) = 00_H</code> (очистка регистра для приёма данных)
<code>EPAREG (H) = 1F_H</code> (<code>EPA0_TIME</code>)	<code>PTSCON1 = 60_H</code> (разрешить контроль нечетности)
<code>EPAREG (L) = 42_H</code> (<code>EPA0_TIME</code>)	<code>PORTMASK = 01_H</code> (<code>P2.0 = TXD</code>)
<code>PTSCON = 21_H</code> (SSIO режим передачи, мажоритарность)	<code>PORTREG (H) = 1F_H</code> (<code>P2_PIN</code>)
<code>PTSCOUNT = 0A_H</code> (8 битов данных, 1 бит четности)	<code>PORTREG (L) = D6_H</code> (<code>P2_PIN</code>)

7 Разрешить прерывание CAPCOMP0.

Установить INT_MASK.2.

8 Загрузить число байтов к передаче в определённый пользователем регистр счета передачи (R_COUNT) и очистить определенный пользователем флаг завершения приема (RXDDONE):

- LD R_COUNT, #16;

- CLRB RXDDONE.

9 Выбрать PTS-сервис для CAPCOMP0.

Установить PTSSEL.2.

10 Установить CAPCOMP0 для захвата задних фронтов.

- Установить EPA0_CON.4 (рисунок 10.10).

11 Разрешить PTS и стандартные прерывания:

- использовать инструкцию EI для разрешения всех стандартных прерываний и EPTS инструкцию для разрешения PTS.

12 Переключить вход RXD, чтобы начать прием. EPA генерирует стандартное прерывание. Процедура обслуживания для стандартного прерывания та же, что и для прерывания end-of-PTS. Она различает прерывание старта приёма от прерывания end-of-PTS чтением регистра EPA0_CON. Если регистр установлен в режим захвата, тогда это прерывание старта приёма.

Процедура обслуживания прерываний состоит из следующих действий:

1 Перевод канала EPA в режим сравнения.

Установка EPA0_CON.6 (выбор режима сравнения).

2 Инициализировать время первого цикла PTS, записав время первого прерывания в EPA0_TIME. Для установки корректного значения надо умножить baud_value ($1A0_H$) на 1,5, сложить результат с текущим значением TIMER1 и сохранить результат в EPA0_TIME. Baud_value задает время первого прерывания PTS, по которому PTS передаёт первый бит данных. Baud_value = $1A0_H$ задаёт скорость 9600 бод.

3 Выбор PTS-сервиса для CAPCOMP0.

Установить PTSSEL.2.

4 PTS начинает загружать данные через интервалы, в 1,5 раза больше времени длительности бита данных. Когда PTSCOUNT уменьшается до нуля, PTS вызывает прерывание end-of-PTS (рисунок 6.28). Процедура обслуживания прерывания проверяет ошибки фрейминга и чётности (рисунок 6.19), очищает регистры DATA для следующего приёма, перезагружает регистры PTSCOUNT и PTSCON1, переводит канал EPA в режим захвата по заднему фронту. Необходимо помнить, что в это время обслуживание PTS для канала EPA запрещено, т.к. следующее прерывание должно быть стандартным.

5 Чтобы определить, когда все байты были переданы, надо создать циклическую подпрограмму, чтобы проверять состояние флага RXDDONE.

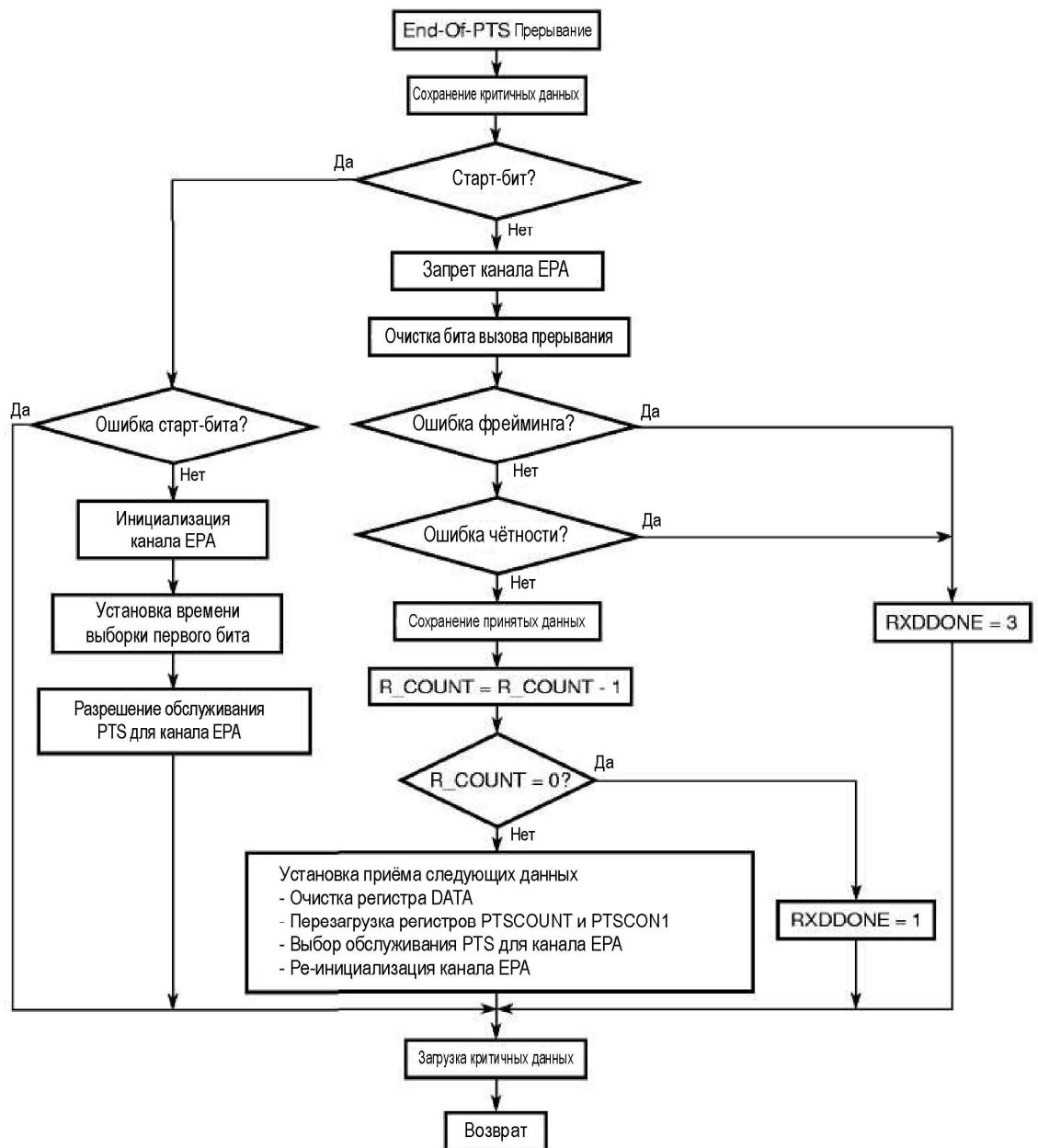


Рисунок 6.28 – Режим асинхронного SIO приёма. Блок-схема процедуры прерывания end-of-PTS

7 Порты ввода-вывода

Порты ввода-вывода обеспечивают механизм передачи информации между устройством и схемами, входящими в периферийное окружение. Они могут считывать состояние системы, осуществлять мониторинг системы, считывать состояние выходных устройств системы, конфигурировать систему, формировать сигналы управления, обеспечивать связь по последовательному порту и так далее.

7.1 Описание портов ввода-вывода

Регистры стандартных портов ввода-вывода расположены в адресном пространстве SFR, к которым обращаются косвенно или через окна. Регистры нестандартных портов ввода-вывода расположены по адресам, соответствующим карте памяти контроллера. К этим регистрам можно обратиться с использованием косвенной или индексной адресации. Через окна в регистровом файле к ним нет доступа. Все порты могут обеспечить медленные инструкции ввода/вывода или обслужить дополнительные функции. В таблице 7.1 приведен краткий обзор портов ввода-вывода устройства. Далее порты описаны более подробно и с объяснением. В подразделах, которые описывают соответствующие периферийные устройства, приводится использование выводов для их специальных функций.

Т а б л и ц а 7.1 – Порты ввода-вывода

Порт	Биты	Тип	Направление	Связанные периферийные устройства
Порт 0	8	Стандартный	Только для входа	АЦП
Порт 1	8	Стандартный	Только для входа	АЦП, ЕРА P1.7, P1.6 – без альтернативных функций
Порт 2	8	Стандартный	Двунаправленный	ЕРА и таймеры
Порт 3	8	Нестандартный	Двунаправленный	Шина адреса/данных
Порт 4	8	Нестандартный	Двунаправленный	Шина адреса/данных
Порт 5	8	Нестандартный	Двунаправленный	Управление шиной
Порт 6	8	Стандартный	Только для выхода	ШИМ, ГФС
Порт 7	8	Стандартный	Двунаправленный	ЕРА и генератор программного тактового сигнала. P7.6, P7.5, P7.4 – без альтернативных функций

7.2 Входные порты 0 и 1

Порт 0 – 8-битный, высокоимпедансный входной порт, который обеспечивает аналоговые и цифровые входы. Входные выводы могут использоваться как цифровые входы, а также как входы АЦП. Порты 0 и 1 отличаются от других стандартных портов, т. к. их выводы могут использоваться только как входы к цифровой или аналоговой схеме. Поскольку порт 0 и порт 1 постоянно формируются как входной порт, они не имеют регистров конфигурации. У каждого порта имеется свой единственный регистр P_x_PIN (x = 0, 1). Он может читаться, чтобы определить текущее состояние вывода порта. К регистру можно обратиться с косвенно-регистровой и индексной адресацией, а также с помощью окна в регистровом файле. В таблице 7.2 приведены стандартные выводы входных портов. В таблице 7.3 описаны регистры состояния P_x_PIN (x = 0, 1).

Т а б л и ц а 7.2 – Стандартные выводы входных портов 0 и 1

Вывод порта	Сигнал специальной функции	Тип сигнала специальной функции	Связанное периферийное устройство
P0.0 – P0.3	ACH0 – ACH3	Вход	АЦП
P0.4	ACH4	Вход	АЦП
	PMODE.0	Вход	EPROM*
P0.5	ACH5	Вход	АЦП
	PMODE.1	Вход	EPROM*
P0.6	ACH6	Вход	АЦП
	PMODE.2	Вход	EPROM*
P0.7	ACH7	Вход	АЦП
	PMODE.3	Вход	EPROM*
P1.0	ACH8	Вход	АЦП
P1.1	ACH9	Вход	АЦП
P1.2	ACH10	Вход	АЦП
	T1CLK	Вход	ЕРА
P1.3	ACH11	Вход	АЦП
	T1DIR	Вход	ЕРА
P1.4	ACH12	Вход	АЦП
P1.5	ACH13	Вход	АЦП
P1.6	–	Вход	–
P1.7	–	Вход	–

* EPROM – внутренняя однократно программируемая память.

Т а б л и ц а 7.3 – Регистры входных портов 0 и 1

Мнемоника	Адрес	Описание
P0_PIN	1FA8 _H	Каждый бит P0_PIN отражает текущее состояние соответствующего вывода порта 0
P1_PIN	1FA9 _H	Каждый бит P1_PIN отражает текущее состояние соответствующего вывода порта 1

Стандартное решение буферов входных портов 0 и 1

На рисунке 7.1 представлено схемное решение вывода входного порта. Транзисторы Q1 и Q2 служат для защиты от статического электричества, они подключены к $\cap V_{CC}$ и $\cap 0V$. Транзистор Q3 – дополнительное устройство защиты от статического электричества, подключенный к #0V. Резистор R1 ограничивает текущий ток через Q3 до приемлемой величины. Из этого узла входной сигнал идет к аналоговому мультиплексору и на цифровой буфер трансляции уровня. Буфер трансляции уровня конвертирует входные сигналы в рабочие сигналы, совместимые с цифровыми уровнями напряжения $U_{\#V_{CC}}$ и $U_{\#0V}$, используемыми ядром центрального процессора. Этот буфер является триггером Шмитта. Он необходим для улучшенной помехоустойчивости. Сигналы защелкиваются в P0_PIN или P1_PIN регистрах и выводятся на внутреннюю шину, когда регистры P0_PIN или P1_PIN читаются.

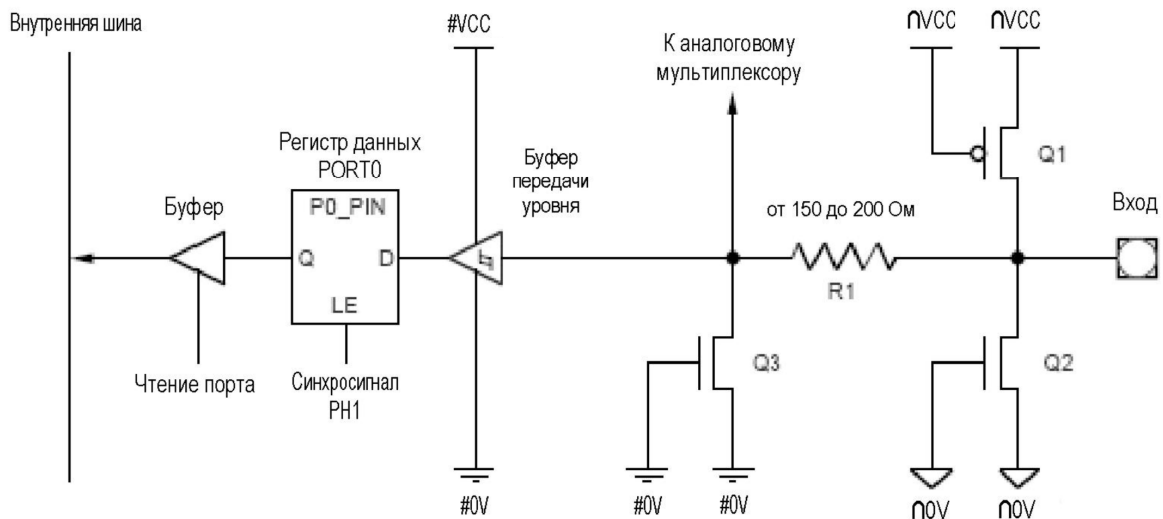


Рисунок 7.1 – Структура входного порта

Анализ стандартных входных портов 0 и 1

Выходы портов 0 и 1 являются уникальными в том, что они могут индивидуально использоваться как цифровые входы и аналоговые входы в одно время. Однако чтение порта усиливает влияние помех на АЦП, уменьшая точность любого аналого-цифрового преобразования. Поэтому строго рекомендуется не читать порты 0, 1 при выполнении аналого-цифрового преобразования. Для уменьшения помех регистры P0_PIN и P1_PIN тактируются только во время чтения порта.

Эти выводы порта запитаны аналоговым напряжением U_{nVCC} и аналоговой землей U_{n0V} . Если выводы порта должны функционировать или как аналоговые, или как цифровые входы, то выводы $nVCC$ и $n0V$ должны обеспечить требуемую мощность. Если напряжение, приложенное к аналоговому входу, превышает U_{nVCC} или U_{n0V} более, чем на 0,5 В, будут открываться защитные транзисторы Q1 или Q2, уменьшая точность всех аналоговых преобразований.

Значение на выводе порта фиксируется за один такт перед разрешением чтения буфера. Фиксация уровня на входе происходит в течение фазы 1 (в то время как CLKOUT имеет низкий уровень) и передается на внутреннюю шину. Чтобы гарантировать правильное чтение, сигнал должен быть установлен не менее чем за 45 нс до переднего фронта CLKOUT и удерживаться до прохождения заднего фронта CLKOUT.

Как цифровой вход, вывод функционирует как вход с высоким импедансом. В то же время, как аналоговый вход, вывод должен обеспечить ток в течение короткого времени для зарядки внутреннего конденсатора, когда начинается аналого-цифровое преобразование. Это означает, что если аналого-цифровое преобразование имеет место на выводе порта, его входные характеристики изменяются моментально.

7.3 Двухнаправленные порты 2, 5, 7

Хотя двухнаправленные порты очень похожи и по схеме, и по конфигурации, порт 5 отличается от других. Порт 5, имеющий адрес из адресного пространства, имеет КМОП-буфер, необходимый для выполнения быстрого управления внешней периферией. Порт 2 использует входные буферы на базе триггера Шмитта для улучшения помехоустойчивости.

В таблице 7.4 приведен список выводов двухнаправленных портов 2, 5, 7 с их сигналами специальной функции и связанными периферийными устройствами.

Т а б л и ц а 7.4 – Выводы двунаправленных портов 2, 5, 7

Вывод порта	Сигнал специальной функции	Тип сигнала специальной функции	Связанное периферийное устройство
P2.0	CAPCOMP0	Вход/выход	ЕРА
	PVER	Выход	EPROM
P2.1	CAPCOMP1	Вход/выход	ЕРА
	PALE#	Выход	EPROM
P2.2	CAPCOMP2	Вход/выход	ЕРА
	PROG#	Выход	EPROM
P2.3	CAPCOMP3	Вход/выход	ЕРА
P2.4	COMP0	Выход	ЕРА
P2.5	COMP1	Выход	ЕРА
P2.6	COMP2	Выход	ЕРА
P2.7	COMP3	Выход	ЕРА
P5.0	ALE/ADV#	Выход	Контроллер памяти
P5.1	INST	Выход	Контроллер памяти
P5.2	WR#/WRL#	Выход	Контроллер памяти
P5.3	RD#	Выход	Контроллер памяти
P5.4	ONCE#	Вход	Контроллер памяти
P5.5	BHE#/WRH#	Выход	Контроллер памяти
P5.6	READY	Вход	Контроллер памяти
P5.7	BW	Вход	Контроллер памяти
P7.0	CAPCOMP4	Вход/выход	ЕРА
P7.1	CAPCOMP5	Вход/выход	ЕРА
P7.2	COMP4	Выход	ЕРА
P7.3	COMP5	Выход	ЕРА
P7.4	–	–	–
P7.5	–	–	–
P7.6	–	–	–
P7.7	FREQOUT	Выход	Генератор программируемого тактового сигнала

П р и м е ч а н и е – EPROM – внутренняя однократно программируемая память.

В таблице 7.5 приведено описание регистров управления портов 2, 5, 7. Каждый порт имеет три регистра управления (Px_MODE, Px_DIR и Px_REG); они могут и читаться, и записываться. Регистры Px_PIN – регистры состояния, повторяющие логический уровень на выводе портов. Они могут только читаться. Регистры порта 2 используют побайтовую адресацию и могут быть доступны через окна. Регистры порта 5 используют 16-битную адресацию и не могут быть доступны через окна.

Т а б л и ц а 7.5 – Управление портами 2, 5, 7 и регистры состояния

Мнемоника	Адрес	Описание
P2_DIR P5_DIR P7_DIR	1FD2 _H 1FF3 _H 1FD3 _H	Направление порта x. Каждый бит Px_DIR управляет направлением соответствующего вывода. 0 комплементарный выход (только выход). 1 вход или выход с открытым стоком (вход, выход или вход/выход). Выходы с открытым стоком требуют внешний pull-up
P2_MODE P5_MODE P7_MODE	1FD0 _H 1FF1 _H 1FD1 _H	Режим порта x. Каждый бит Px_MODE определяет, функционирует ли соответствующий вывод как стандартный вывод порта ввода-вывода или как сигнал специальной функции. 0 стандартный вывод порта ввода-вывода. 1 сигнал специальной функции
P2_PIN P5_PIN P7_PIN	1FD6 _H 1FF7 _H 1FD7 _H	Вход порта x. Каждый бит Px_PIN отражает текущее состояние соответствующего вывода независимо от конфигурации вывода
P2_REG P5_REG P7_REG	1FD4 _H 1FF5 _H 1FD5 _H	Выход данных порта x. Для входа необходимо установить соответствующий Px_REG бит. Для выхода написать данные, которые будут выведены каждым выводом в соответствующий бит Px_REG. Когда вывод конфигурируется как стандартный вход/выход (Px_MODE.y = 0), результат записи центрального процессора в Px_REG немедленно виден на выводе. Когда вывод формируется как сигнал специальной функции (Px_MODE.y = 1), соответствующая периферия на МК или компонент вне МК управляет выводом. Центральный процессор может все еще писать Px_REG, но состояние вывода не меняется, пока он не переключен к его стандартной функции входа/выхода. Эта особенность позволяет программному обеспечению формировать вывод как стандартный вход/выход (очистить Px_MODE.y), инициализировать или переписывать значение вывода, затем формировать вывод как сигнал специальной функции (необходимо установить Px_MODE.y). Этим способом инициализация, восстановление ошибки, обработка исключения и т. д. могут быть сделаны без изменения работы связанного периферийного устройства

Работа двунаправленных портов 2, 5, 7

На рисунке 7.2 показывается логика управления выходных транзисторов Q1 и Q2. Q1 может выдать ток до минус 3 мА при выходном напряжении ($U_{\#VCC} - 0,7$) В. Q2 может пропустить ток не менее 3 мА при выходном напряжении 0,45 В. В режиме ввода-вывода (выбирается обнулением Px_MODE.y) состояние Px_REG и Px_DIR (x=2, 5, 7) подается на мультиплексоры. Это позволяет управлять транзисторами Q1 и Q2 так, чтобы вывод находился в состоянии высокого или низкого уровня или в высокоимпедансном состоянии. В таблице 7.6 приведена логическая таблица для функционирования ввода-вывода этих портов.

В режиме специальной функции (выбирается установкой Px_MODE.y) SFDIR и SFDATA – управляющие сигналы для мультиплексов. Этими сигналами производится управление транзисторами Q1 и Q2 так, чтобы можно было установить на выводе или высокий, или низкий уровень сигнала, или высокоимпедансное состояние. Для выходного сигнала специальной функции обнуляют SFDIR. Для входного сигнала специальной функции устанавливают SFDIR. Таблица 7.6 – логическая таблица для выполнения специальной функции этих портов. Даже если вывод должен использоваться в режиме специальной функции, необходимо инициализировать вывод как вход или выход, записывая Px_DIR.

Резистор R1 обеспечивает защиту от статического электричества. Входные сигналы буферизуются. Стандартные порты используют в качестве буфера триггер Шмитта для улучшенной помехоустойчивости. Порт 5 использует стандартный входной буфер для быстрых переключений, требуемых для управления шиной. Сигналы защелкиваются в регистре Px_PIN и выводятся на внутреннюю шину, когда регистр Px_PIN читается.

Спадающий фронт сигнала RESET# открывает транзистор Q3, который остается открытым приблизительно на 300 нс, заставляя вывод быстро изменить его состояние на состояние в режиме сброса. Активный низкий уровень RESET# включает транзистор Q4, который осуществляет слабую поддержку высокого уровня. Q4 осуществляет эту слабую поддержку, пока не будет изменен Px_MODE.

Примечание – После сброса по выводу P2.7 выдается сигнал CLKOUT, вместо поддержки высокого уровня.

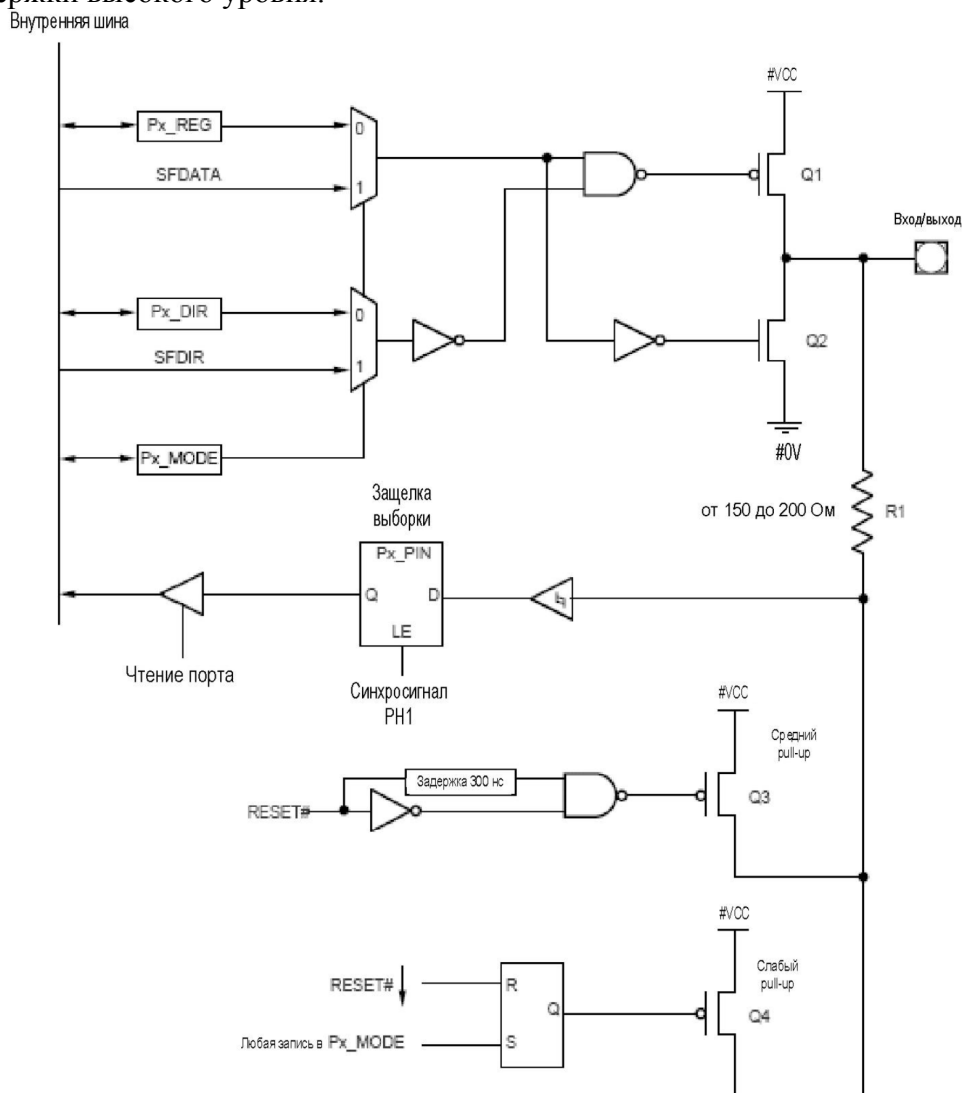


Рисунок 7.2 – Структура двунаправленного порта

Т а б л и ц а 7.6 – Таблица логических уровней для двунаправленных портов 2, 5, 7 в режиме входа/выхода

Конфигурация	Дополнительный выход		Выход с открытым стоком	Вход
	0	1		
Px_MODE	0	0	0	0
Px_DIR	0	0	1	1
SFDIR	X ¹⁾	X	X	X
SFDATA	X	X	X	X
Px_REG	0	1	0, 1 ²⁾	1
Q1	закрыт	открыт	закрыт	закрыт
Q2	открыт	закрыт	открыт, закрыт ²⁾	закрыт
Px_PIN	0	1	X ³⁾	высокий импеданс ⁴⁾

¹⁾ X = любой уровень.
²⁾ Если Px_REG очищен, то Q2 открыт (пропускает ток); если Px_REG установлен, то Q2 закрыт (x = 2, 5, 7).
³⁾ Px_PIN содержит текущее значение на выводе.
⁴⁾ В течение сброса и до первой записи в Px_MODE Q4 открыт (пропускает ток).

Конфигурация двунаправленных портов 2, 5, 7

Каждый двунаправленный вывод порта может быть индивидуально сконфигурирован, чтобы работать или как вывод входа/выхода, или как вывод сигнала специальной функции. В режиме специальной функции сигнал управляется периферийным устройством МК или компонентом вне МК. В любой конфигурации возможны два режима:

- комплементарный выход (только выход);
- высокоимпедансный вход или выход с открытым стоком (вход, выход или двунаправленный).

Для предотвращения плавающего уровня на выводе порта выводы двунаправленных портов имеют слабую поддержку высокого уровня, пока программное обеспечение не запишет Px_MODE (x = 2, 5, 7). Заданные по умолчанию значения регистров управления после начальной установки МК (RESET#) конфигурируют выводы портов как высокоимпедансные входы со слабой поддержкой высокого уровня (pull-up). Чтобы гарантировать, что порты инициализируются правильно и что слабая поддержка высокого уровня отключена, выполняют следующую последовательность шагов инициализации:

1 Производится запись в Px_DIR, чтобы установить конкретные выводы как входы или как выходы (выводы будут управлять данными, которые определяются в шаге 3).

Для комплементарного выхода очистить его Px_DIR бит.

Для высокоимпедансного входа или выхода с открытым стоком установить его Px_DIR бит (выходы с открытым стоком требуют внешней слабой поддержки высокого уровня).

2 Производится запись в Px_MODE, чтобы выбрать или режим специальной функции, или вход/выход. Запись Px_MODE (независимо от значения) выключает поддержку высокого уровня. Даже если порт должен использоваться как вход/выход (это выполняется после начальной установки), необходимо записать Px_MODE, чтобы гарантировать, что слабая поддержка высокого уровня была выключена.

Для стандартного входа/выхода очистить его бит в Px_MODE. В этом режиме вывод устанавливаются, как определено в шагах 1 и 3.

Для сигнала специальной функции установить его бит в Px_MODE. В этом режиме связанное периферийное устройство управляет выводом.

3 Производится запись в Px_REG.

Для выводов, работающих как выходы (задается в шаге 1), записать данные в соответствующие биты Px_REG, чтобы получить эти данные на соответствующих выводах. Для выходов в режиме специальной функции значение является несущественным, потому что периферийное устройство управляет выводом. Тем не менее, необходимо сделать запись в Px_REG, чтобы инициализировать вывод.

Для выводов, установленных как входы (определяется в шаге 1), установить соответствующие биты Px_REG.

В таблице 7.7 приведены значения управляющих регистров для каждой возможной конфигурации. Для выходов в режиме специальной функции значение Px_REG является несущественным, потому что связанное периферийное устройство управляет выводом в режиме специальной функции. Однако необходимо записывать в Px_REG, чтобы инициализировать вывод. Для двунаправленного вывода, чтобы он функционировал как вход (или вывод специальной функции, или вывод порта), необходимо установить Px_REG.

Т а б л и ц а 7.7 – Значения управляющих регистров для каждой конфигурации

Необходимая конфигурация вывода	Назначения регистра конфигурации		
	Px_DIR	Px_MODE *	Px_REG
Стандартный сигнал входа - выхода	Px_DIR	Px_MODE *	Px_REG
Комплементарный выход, вывод 0	0	0	0
Комплементарный выход, вывод 1	0	0	1
Выход с открытым стоком, вывод сильного 0	1	0	0
Выход с открытым стоком, высокий импеданс	1	0	1
Вход	1	0	1
Сигнал специальной функции	Px_DIR	Px_MODE *	Px_REG
Комплементарный выход, выходное значение, управляемое периферийным устройством	0	1	X
Выход с открытым стоком, выходное значение, управляемое периферийным устройством	1	1	X
Вход	1	1	1

* В течение сброса и до первой записи Px_MODE выходы имеют слабую поддержку высокого уровня (x = 2, 5, 7).

Пример конфигурации выводов двунаправленного порта

Предположим, что необходимо сконфигурировать выходы двунаправленного порта, как показано в таблице 7.8.

Т а б л и ц а 7.8 – Пример конфигурации порта

Выводы порта	Конфигурация	Данные
Px.0, Px.1	высокоимпедансный вход	высокий импеданс
Px.2, Px.3	выход с открытым стоком	0
Px.4	выход с открытым стоком	1 (при условии внешней слабой поддержки высокого уровня)
Px.5, Px.6	комплементарный выход	0
Px.7	комплементарный выход	1
Примечание – x равен 2, 5, 7.		

Можно использовать следующий набор команд. В таблице 7.9 показывается состояние каждого вывода после того, как прошла начальная установка МК (сброс) и после выполнения ниже приведенных команд:

```
LDB Px_DIR, #00011111B
LDB Px_MODE, #00000000B
LDB Px_REG, #10010011B
```

Т а б л и ц а 7.9 – Состояние выводов после сброса и после выполнения команд

Действие или код	Итоговые состояния выводов							
	Px.7	Px.6	Px.5	Px.4	Px.3	Px.2	Px.1	Px.0
RESET	wk1	wk1	wk1	wk1	wk1	wk1	wk1	wk1
LDB Px_DIR, #00011111B	1	1	1	wk1	wk1	wk1	wk1	wk1
LDB Px_MODE, #00000000B	1	1	1	HZ1	HZ1	HZ1	HZ1	HZ1
LDB Px_REG, #10010011B	1	0	0	HZ1	0	0	HZ1	HZ1
<p>П р и м е ч а н и я</p> <p>1 wk1 – слабая поддержка высокого уровня.</p> <p>2 HZ1 – высокий импеданс (фактически «1» с внешней поддержкой высокого уровня).</p> <p>3 x равен 2, 5, 7.</p>								

Анализ двунаправленных портов

Ниже приведен специальный анализ использования портов.

Порт 2. После сброса программное обеспечение должно сконфигурировать устройство в соответствии с внешней системой. Это достигается записью соответствующих данных конфигурации в P2_MODE. Запись в P2_MODE не только конфигурирует выводы, но также и выключает транзистор, который осуществляет слабую поддержку высокого уровня (Q4 на рисунке 7.2). По этой причине, даже если порт 2 должен использоваться, как это формируется при сбросе, необходимо все равно записать данные в P2_MODE.

P2.7. Значение, записанное в P2_REG.7 удерживается в буфере пока P2_MODE.7 не очищен, а затем это значение загружается в P2_REG.7. Значение, читаемое из P2_REG.7, это значение регистра, а не буфера. Любое изменение P2_REG.7 может быть прочитано только после очистки P2_MODE.7.

Порт 5. После сброса программное обеспечение должно сконфигурировать устройство в соответствии с внешней системой. В дальнейшем будут описаны состояния выводов порта 5 после сброса до записи в P5_MODE. Запись P5_MODE не только конфигурирует выводы, но также и выключает транзистор, который осуществляет слабую поддержку высокого уровня (Q4 на рисунке 7.2). По этой причине, даже если порт 5 должен использоваться, как это формируется при сбросе, необходимо все равно записать данные в P5_MODE.

P5.0/ALE. Если удерживать высокий уровень на EA# при сбросе (доступ внутренней памяти программ), вывод имеет слабую поддержку высокого уровня, пока не будет записан P5_MODE. Если удерживать низкий уровень на EA# при сбросе (доступ внешней памяти программ), то ALE или ADV# активируется как системный управляющий вывод в зависимости от бита ALE в CCR0. В любом случае, вывод становится настоящим комплементарным выходом.

P5.1/INST – этот вывод имеет слабую поддержку высокого уровня, пока программное обеспечение не запишет конфигурацию в P5_MODE.

P5.2/WR#/WRL# – этот вывод имеет слабую поддержку высокого уровня, пока программное обеспечение не запишет конфигурацию в P5_MODE.

P5.3/RD#. Если удерживать высокий уровень на EA# при сбросе (доступ к внутренней памяти программ), вывод имеет слабую поддержку высокого уровня, пока не

будет записан P5_MODE. Если удерживать низкий уровень на EA# при сбросе (доступ внешней памяти программ), то RD# активируется как системный управляющий вывод и становится настоящим комплементарным выходом.

P5.4 – этот вывод имеет слабую поддержку высокого уровня, пока программное обеспечение не запишет конфигурацию в P5_MODE. P5.4 – вывод, разрешающий режим ONCE. Поскольку низкий уровень сигнала в течение сброса может заставить устройство войти в режим ONCE, необходимо поддерживать высокий уровень на выводе в течение сброса, чтобы предотвратить случайный вход в режим ONCE.

P5.5/BHE#/WRN# – этот вывод имеет слабую поддержку высокого уровня, пока запись CCB не закончена. Затем, состояние этого вывода зависит от значения BW0 бита CCR0. Если BW0 обнулен, вывод имеет слабую поддержку высокого уровня, пока не будет записан P5_MODE. Если BW0 установлен, BHE# активизирован как вывод управления системы, и вывод становится комплементарным выходом.

P5.6/READY – этот вывод имеет слабую поддержку высокого уровня, пока запись CCB не закончена. В это время состояние этого вывода зависит от значения битов CCR0,1 в IRC0–IRC2. Если все IRC0–IRC2 установлены (111_B), активизирован READY как системный управляющий вывод. Это предотвращает вставку бесконечных циклов ожидания до первого обращения к внешней памяти. При других значениях IRC0–IRC2 вывод конфигурируется после сброса как вход/выход.

Примечание – Если IRC0–IRC2 CCB все установлены (READY активируется как системный управляющий вывод) и P5_MODE.6 очищен (конфигурирование вывода как вход/выход), при обращении к внешней памяти микроконтроллер может заблокироваться.

P5.7/BW – этот вывод имеет слабую поддержку высокого уровня, пока программное обеспечение не запишет конфигурацию в P5_MODE.

7.4 Двухнаправленные порты 3, 4 (шина адресов/данных)

Порты 3 и 4 – 8-битные, двухнаправленные, имеющие адреса из общего адресного пространства порты ввода-вывода. К ним можно обратиться только с косвенной или индексной адресацией и нельзя обратиться через регистровые окна. Порты 3 и 4 обеспечивают мультиплексную шину адреса/данных. В режиме программирования порты 3 и 4 служат шиной программирования (PBUS). Порт 5 обеспечивает сигналы управления шины.

В течение внешних циклов шины запоминающего устройства МК забирает под свой контроль порты 3 и 4 и автоматически формирует их как комплементарные порты вывода для того, чтобы сконфигурировать их для вывода адреса/данных или как входы для того, чтобы сосчитать данные. По этой причине эти порты не имеют регистров режима.

Микроконтроллер при неактивном EA# (высокий) имеет нерабочее время между внешними шинными циклами. Когда шина адреса/данных простаивает, можно использовать эти порты для ввода-вывода. Подобно порту 5, эти порты используют стандартные входные CMOS буферы. Однако порты 3 и 4 должны полностью функционировать или как комплементарные или порты с открытым стоком. Их выводы не могут конфигурироваться индивидуально. Микроконтроллер при активном EA# (низкий) не могут использовать порты 3 и 4 как стандартные входы/выходы. Если EA# активный (низкий), эти порты будут функционировать только как шина адреса/данных.

В таблице 7.10 приведен список выводов портов 3 и 4 с их сигналами специальной функции и связанными периферийными устройствами. В таблице 7.11 приведен список регистров, которые определяют функцию и указывают состояние портов 3 и 4.

Т а б л и ц а 7.10 – Выводы портов 3 и 4

Выводы порта	Сигнал специальной функции	Тип сигнала специальной функции	Связанное периферийное устройство
P3.7 – P3.0	AD7 – AD0	Вход/выход	Шина адреса/данных, младший байт
	PBUS7 – PBUS0	Вход/выход	Шина программирования, младший байт
P4.7 – P4.0	AD15 – AD8	Вход/выход	Шина адреса/данных, старший байт
	PBUS15 – PBUS8	Вход/выход	Шина программирования старший байт

Т а б л и ц а 7.11 – Регистры управления и состояния портов 3, 4

Мнемоника	Адрес	Описание
P3_PIN P4_PIN	1FFE _H 1FFF _H	Вход порта x (x = 3, 4). Каждый бит P _x _PIN отражает текущее состояние соответствующего вывода независимо от конфигурации вывода
P3_REG P4_REG	1FFC _H 1FFD _H	Выход порта x (x = 3, 4). Каждый бит P _x _REG содержит данные, которые будут выданы соответствующим выводом. Когда устройство требует доступа к внешнему запоминающему устройству, МК берет управление портом и управляет выдачей адреса/данных бит на вывод. Биты адреса/данных выводятся в это время. Когда доступ к внешней памяти закончен, устройство восстанавливает данные на выводе

Работа двунаправленных портов 3, 4 (шины адреса/данных)

На рисунке 7.3 показывается логика портов 3, 4. В течение сброса активный низкий уровень RESET# выключает Q1 и Q2 и включает транзистор Q3, который осуществляет слабую поддержку высокого уровня. (Q1 может пропускать ток до 3 мА при (U_{#VCC}–0,7) В на выходе; Q2 может пропускать ток до 3 мА при 0,45 В на выходе; и Q3 может пропускать ток около 10 мкА при (U_{#VCC} – 1,0) В на выходе.) В течение нормальной работы (как порта) внутренний сигнал управления BUS CONTROL SELECT управляет портом.

Когда микроконтроллеру необходим доступ к внешней памяти, он обнуляет сигнал BUS CONTROL SELECT, который подает адрес/данные на вход мультиплексора. Адрес/данные выдаются, управляя транзисторами Q1 и Q2.

Когда доступ к внешней памяти не требуется, микроконтроллер устанавливает сигнал BUS CONTROL SELECT, который выбирает P_x_REG как источник данных для мультиплексора. P_x_REG управляет транзисторами Q1 и Q2 как выходом с открытым стоком. (Выходы с открытым стоком требуют внешних резисторов поддержки высокого уровня.) В этой конфигурации вывод порта может использоваться как вход. Сигнал на выводе записывается в P_x_PIN. В таблице 7.12 приведены данные для конфигурации порта как вход/выход с открытым стоком.

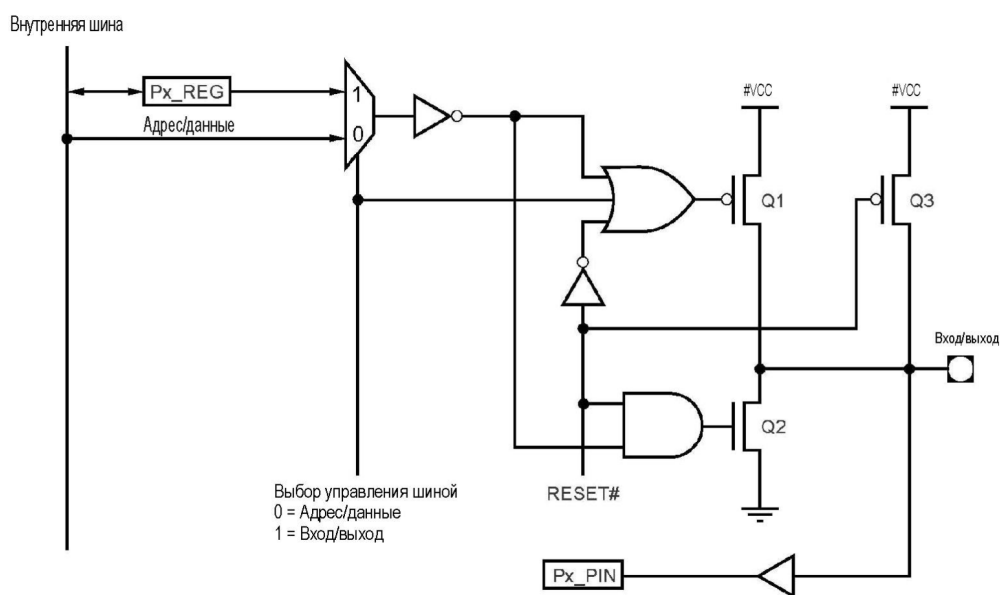


Рисунок 7.3 – Структура портов 3 и 4, шина адресов/данных

Т а б л и ц а 7.12 – Логическая таблица для портов 3, 4 в режиме входа/выхода с открытым стоком

Конфигурация	Открытый сток	
	0	1
Px_REG	0	1
Q1	закрит	закрит
Q2	открыт	закрит
Px_PIN	0	высокий импеданс

Использование портов 3 и 4 для входа/выхода

Чтобы использовать вывод порта как выход, необходимо записать выходные данные в соответствующий Px_REG бит. Когда устройство требует доступа к внешней памяти, оно берет под свое управление порт и выводит бит адреса/данных на вывод. Бит адреса/данных выдает выходные данные в течение этого времени. Когда внешний доступ завершен, микроконтроллер восстанавливает ваши данные на выводе.

Чтобы использовать вывод порта как вход, необходимо установить соответствующий бит Px_REG для перевода вывода в высокоимпедансное состояние. После этого можно читать входное значение вывода в регистр Px_PIN. Когда МК требует доступа к внешней памяти, он берет под свое управление порт. Необходимо сконфигурировать источник входных сигналов порта так, чтобы избежать конфликтных ситуаций в шине.

Анализ работы портов 3 и 4

Когда сигнал EA# активный (низкий уровень), порты 3 и 4 будут функционировать только как шина адреса/данных. В этом случае инструкция, которая работает с P3_REG или P4_REG, организует шинный цикл, в течение которого происходит чтение из или запись во внешнюю память по соответствующему адресу SFR. Например, запись в P4_REG организует шинный цикл, в результате которого во внешнюю память попадает содержимое ячейки 1FFD_H. Поскольку P3_REG и P4_REG не формируют выходное состояние на выводе, когда EA# активный, шина будет иметь плавающий уровень в течение многих периодов простоя (например, в течение выполнения инструкций BMOV или TJJMP).

Когда сигнал EA# не активный (высокий уровень), выводы портов 3 и 4 отображают содержание регистров P3_REG и P4_REG, которые при сбросе устанавливаются в FF_H.

Выводы переходят в состояние высокого импеданса. Уровень напряжения на выводах портов 3 и 4 будет плавать, если через внешние резисторы не организовать слабую поддержку высокого уровня или не записать нули в регистры P4_REG и P3_REG.

7.5 Стандартный выходной порт 6

Порт 6 – выходной порт, который обеспечивает выходы для генератора формы сигнала и сигналов широтно-импульсной модуляции (ШИМ). Выводы порта 6 могут конфигурироваться, чтобы работать или как выводы порта или как выводы генератора формы сигнала или выводы широтно-импульсного модулятора (ШИМ). В таблице 7.13 приведен список выводов с соответствующими сигналами специальной функции и связанными периферийными устройствами.

Т а б л и ц а 7.13 – Стандартные выводы выходного порта 6

Вывод порта	Сигнал специальной функции	Тип сигнала специальной функции	Связанное периферийное устройство
P6.0	WG1#	Выход	Генератор формы сигнала
P6.1	WG1	Выход	Генератор формы сигнала
P6.2	WG2#	Выход	Генератор формы сигнала
P6.3	WG2	Выход	Генератор формы сигнала
P6.4	WG3#	Выход	Генератор формы сигнала
P6.5	WG3	Выход	Генератор формы сигнала
P6.6	PWM0	Выход	Широтно-импульсный модулятор
P6.7	PWM1	Выход	Широтно-импульсный модулятор

Т а б л и ц а 7.14 – Регистр управления выходного порта 6

Мнемоника	Адрес	Описание
WG_OUTPUT	1FC0 _H	Управляющий регистр выходного порта 6. Этот регистр управляет выводами порта 6 в режиме входа/выхода. Его функции различны в режиме генератора формы сигнала и в режиме широтно-импульсного модулятора

Работа выходного порта 6

На рисунке 7.4 показана упрощенная схема порта 6. Порт 6 имеет собственную конфигурацию и регистр управления WG_OUTPUT. Транзистор Q1 может пропускать ток не менее 7 мА при выходном уровне ($U_{\#VCC} - 1,5$) В. Для выводов P6.0 – P6.5 транзистор Q2 может пропустить ток не менее 10 мА при выходном уровне 0,45 В. Для выводов P6.6 и P6.7 транзистор Q2 может пропустить ток не менее 3,2 мА при выходном уровне 0,45 В.

Конфигурирование выводов выходного порта 6

Порт 6 имеет собственный регистр конфигурации WG_OUTPUT (таблицы 7.14, 7.15, рисунок 7.5). Этот регистр управляет функциями вывода, значениями и полярностью вывода. К этому регистру можно обратиться или как к слову, или как к отдельным байтам, а также можно обращаться через регистровые окна. Функции этого регистра по формированию выходов общего назначения отличны от тех, что формируются для работы в качестве выводов генератора формы сигналов и широтно-импульсного модулятора.

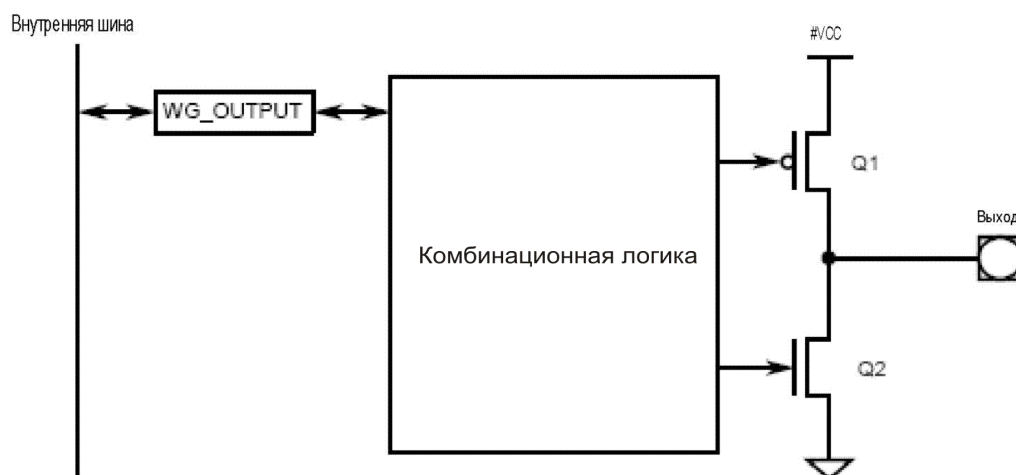


Рисунок 7.4 – Структура выходного порта

WG_OUTPUT (порт 6)

Адрес: 1FC0_H

Состояние сброса: 0000_H

Регистр WG_OUTPUT управляет функциями порта 6. Если используется порт 6 для выходов общего назначения, то необходимо записать C0_H (для активно-высоких выводов) или 00_H (для активно-низких выводов) в старший байт WG_OUTPUT и записать желаемые значения выводов в младший байт. Более подробно регистр WG_OUTPUT описан в разделе 8 «Генератор формы сигнала» и в разделе 9 «Широтно-импульсный модулятор».

15	OP1	OP0	–	M7	M6	M5, M4	M3, M2	M1, M0	8
7	D7	D6	D5	D4	D3	D2	D1	D0	0

Рисунок 7.5 – Регистр управления выходного порта WG_OUTPUT

Т а б л и ц а 7.17 – Поля регистра управления выходного порта WG_OUTPUT

Разряд	Мнемоника	Функция
15, 14	OP1, OP0	Полярность вывода. Эти биты выбирают полярность вывода. 0 – активный низкий уровень, 1 – активный высокий уровень
13	–	Зарезервирован. Для совместимости с будущими устройствами необходимо записать ноль в этот бит
12 – 8	M7 – M0	Режим. Эти биты выбирают или периферийную функцию, или функцию выхода общего назначения. Необходимо очистить эти биты для выхода общего назначения
7 – 0	D7 – D0	Данные. В режиме выхода общего назначения эти биты выставляются на выводах. Желаемые значения записывают в эти биты (биты 7–0 соответствуют выводам P6.7 – P6.0)

8 Генератор формы сигнала

Генератор формы сигнала ГФС упрощает задачу формирования синхронизированного широтно-импульсно-модулированного ШИМ выходного сигнала. Данный ГФС оптимизирован для приложений управления двигателями такими, как трехфазные асинхронные двигатели переменного тока, трехфазные бесконтактные двигатели постоянного тока или четырехфазные шаговые двигатели. ГФС может выдавать три независимых пары комплементарных ШИМ сигналов, которые совместно используют общий период несущей, «время простоя» (Dead Time) и рабочий режим. После инициализации генератор формы сигнала работает без вмешательства центрального процессора, пока программа не изменит скважность.

В этом разделе описывается работа генератора формы сигнала и объясняется, как его конфигурировать.

8.1 Краткий обзор функций генератора формы сигнала

Генератор формы сигнала (рисунок 8.1) имеет три основных части: задающий генератор, формирователи фазированных сигналов и схему управления. Задающий генератор устанавливает период опорного сигнала («несущей»), формирователи фазированных сигналов определяют скважность, а схема управления задаёт режим и управляет выработкой прерываний. Максимальная частота ГФС – 15,625 кГц для режимов выравнивания по центру и 31,250 кГц для режимов выравнивания по фронту.

Формируются три независимых фазы, каждая из которых имеет два программируемых комплементарных вывода. Программируемый генератор «времени простоя» запрещает активацию комплементарных выводов в определенное время. Период «несущей», время «простоя» (Dead Time) и рабочий режим, одни и те же для всех трех фаз; скважность программируется независимо.

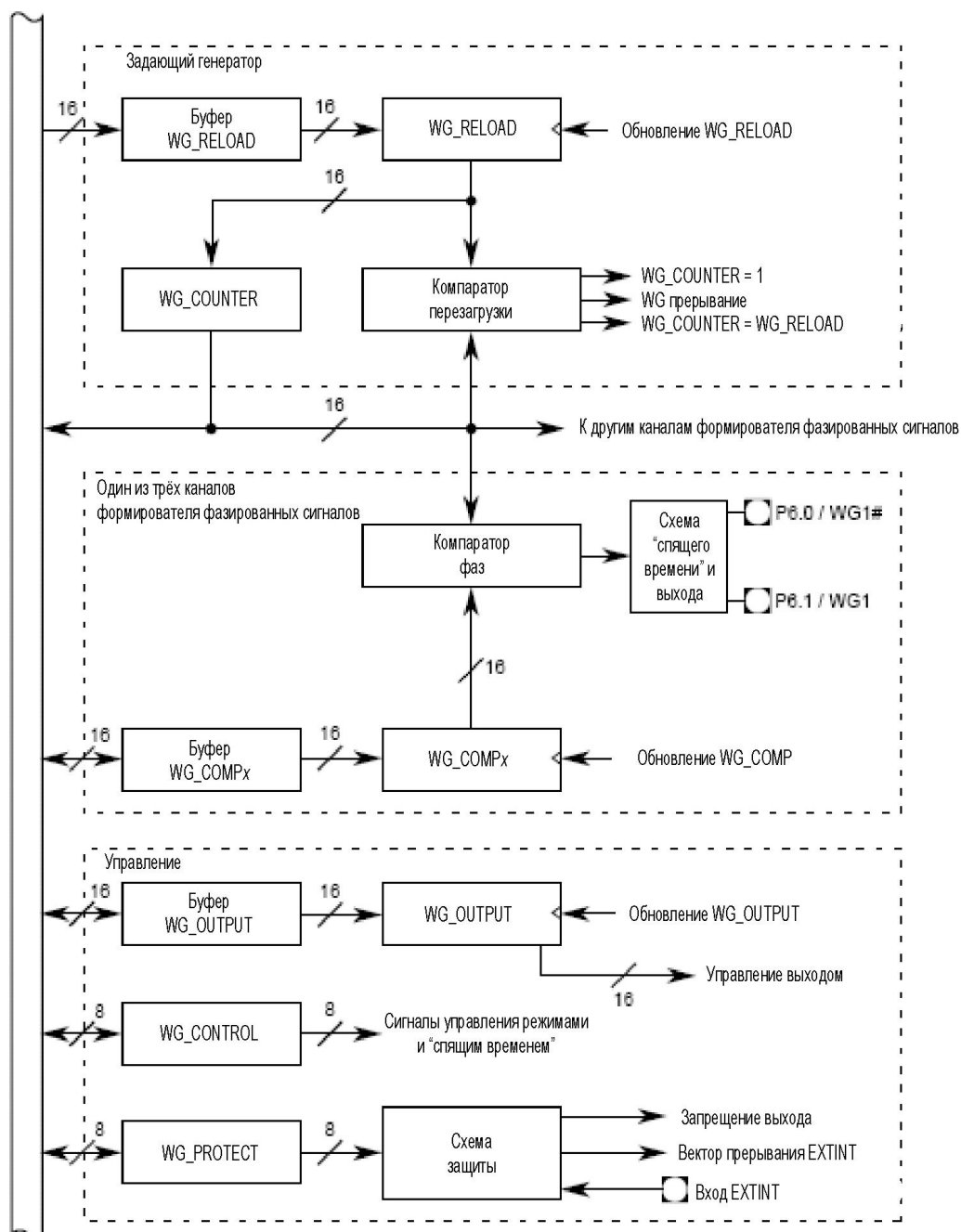


Рисунок 8.1 – Блок-схема генератора формы сигнала

8.2 Сигналы и регистры генератора формы сигнала

В таблице 8.1 описываются сигналы ГФС, а в таблице 8.2 кратко описываются регистры состояния и управления.

Т а б л и ц а 8.1 – Сигналы генератора формы сигнала

Выход порта	Сигнал ГФС	Тип	Описание
P6.0	WG1#	Выход	Инверсный выход фазы 1 ГФС
P6.1	WG1	Выход	Прямой выход фазы 1 ГФС
P6.2	WG2#	Выход	Инверсный выход фазы 2 ГФС
P6.3	WG2	Выход	Прямой выход фазы 2 ГФС
P6.4	WG3#	Выход	Инверсный выход фазы 3 ГФС
P6.5	WG3	Выход	Прямой выход фазы 3 ГФС
–	EXTINT	Вход	Вход схемы защиты ГФС

Т а б л и ц а 8.2 – Регистры управления и состояния ГФС

Мнемоника	Адрес	Описание
INT_MASK1	0013 _H	Маска 1 прерывания. EXTINT бит разрешает или запрещает прерывание EXTINT. PI бит разрешает или запрещает мультиплексированное периферийное прерывание. Соответствующий бит в регистре PI_MASK разрешает или запрещает индивидуальные источники периферийного прерывания
INT_PEND1	0014 _H	Ожидание 1 прерывания. Любой установленный бит индицирует ожидаемый запрос прерывания
PI_MASK	1FBC _H	Маска периферийного прерывания. WG бит разрешает или запрещает прерывание ГФС как один из возможных источников мультиплексированного периферийного прерывания. В PI бите в INT_MASK1 должно быть установлено разрешение мультиплексированного периферийного прерывания
PI_PEND	1FBE _H	Ожидание периферийного прерывания. Любой установленный бит индицирует ожидаемый запрос прерывания
WG_COMP1 WG_COMP2 WG_COMP3	1FC2 _H 1FC4 _H 1FC6 _H	Буферы сравнения ГФС. Каждый буфер сравнения фаз содержит значение, которое сравнивается со значением счётчика. Действие, выполняемое в случае соответствия значений, зависит от операционного режима
WG_CONTROL	1FCC _H	Управление ГФС. Регистр управления определяет операционный режим ГФС, стартует и останавливает счетчик, определяет «время простоя» для всех фаз и указывает текущее направление счета
WG_COUNTER	1FCA _H	Значение счетчика ГФС. Предназначенный только для чтения регистр-счётчик отражает текущее значение счётчика
WG_OUTPUT	1FC0 _H	Управление выходами ГФС. Регистр управления выходами конфигурирует выходы ГФС и выбирает их активную полярность
WG_PROTECT	1FCE _H	Защита ГФС. Защитный регистр разрешает и запрещает схему защиты и выходы, выбирает прерывания по уровню или по фронту и контролирует, какое значение фронта или уровня включит запрос на прерывание
WG_RELOAD	1FC8 _H	Значение перезагрузки ГФС. Регистр перезагрузки содержит значение, которое сравнивается со значением счётчика. Действия, выполняемые в случае соответствия значений, зависят от операционного режима

8.3 Работа генератора формы сигнала

В этом подразделе описываются главные компоненты ГФС: задающий генератор, формирователи фазированных сигналов, схемы управления и защиты. Далее объясняется, как обновляются буферные регистры, и описываются сходство и различие между «центрированным» и «фронтным» операционными режимами. Наконец, здесь описывается два типа запросов на прерывание, которые может вырабатывать ГФС, и объясняется, как разрешать прерывания.

Задающий генератор

Задающий генератор устанавливает период опорного сигнала выводов ШИМ. Определяется этот период записью значения в регистр перезагрузки (WG_RELOAD). Это значение загружается в счётный регистр (WG_COUNTER) при инициализации системы и периодически (в зависимости от операционного режима) впоследствии. Необходимо считать счётный регистр, чтобы определить текущее значение счётчика, и необходимо записывать в регистр перезагрузки, чтобы изменить значение перезагрузки в любое время.

16-битный счетчик опорного сигнала переключается каждый машинный цикл. Регистр управления (WG_CONTROL) разрешает и отключает счетчик, управляет режимом счета и отражает направление счета. Когда счетчик разрешён, он непрерывно считает между 0001_H и значением перезагрузки. Запись 0000_H в регистр перезагрузки или очистка разрешающего бита в регистре управления останавливает счетчик.

Формирователи фазированных сигналов

Формирователи фазированных сигналов определяют скважность выходных сигналов. Определяется скважность, записывая значение в регистр сравнения каждой фазы (WG_COMPx). Во всех операционных режимах выходы установлены при инициализации, и они остаются установленными, пока значение счётчика (WG_COUNTER) не будет соответствовать значению регистра сравнения фазы (WG_COMPx). В этом случае выходы сброшены и остаются сброшенными, пока не произойдёт другое событие. Событие, которое заставит выходы быть установленными снова, зависит от операционного режима.

Схема генератора «времени простоя» (рисунок 8.2) предотвращает выход и его инверсный выход от того, чтобы быть установленным в то же самое время. Схема использует два внутренних сигнала WFG и DT, чтобы вырабатывать неперекрывающиеся выходные сигналы. Схема обнаружения фронта формирует сигнал WFG, в то время как 10-битный счетчик «времени простоя» формирует сигнал DT (Dead Time). Когда обнаружен заданный фронт, в счетчик «времени простоя» (Dead Time) загружается 10-битное значение «времени простоя» из регистра управления, и DT устанавливается в низкий уровень. Каждый машинный цикл содержимое счётчика уменьшается на единицу, пока не достигнет нуля; в этом случае счётчик останавливается и сигнал DT выставляется в высокий уровень. Сигнал WFG логически умножается (AND) с DT, чтобы получить сигнал WG_EVEN; сигнал WFG# логически умножается (AND) с DT, чтобы получить сигнал WG_ODD. Выходы генератора формы сигнала могут быть соединены с сигналами WG_ODD и WG_EVEN.

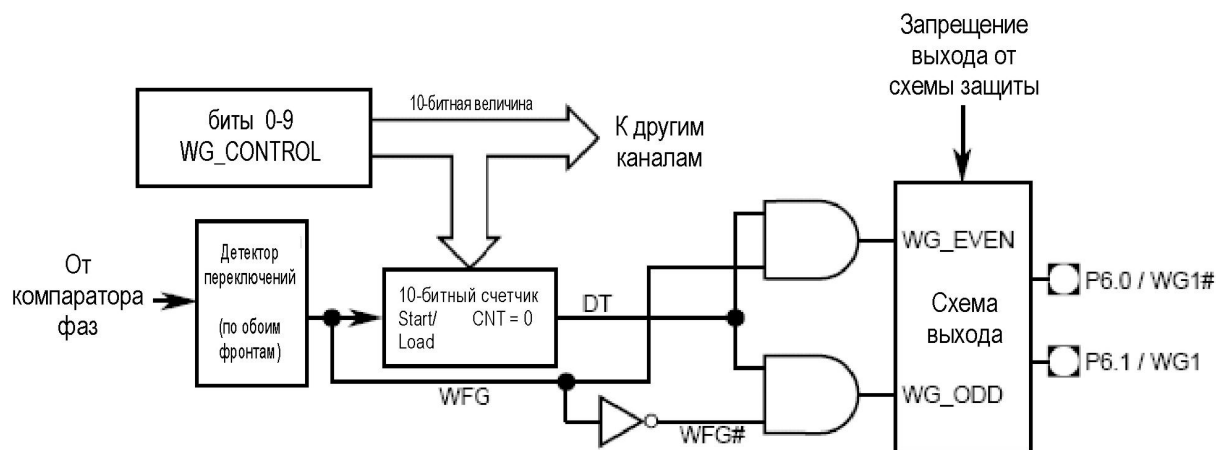


Рисунок 8.2 – Схема генератора «времени простоя»

Схема управления и защиты

Схема управления содержит регистры управления (WG_CONTROL) и выходные регистры (WG_OUTPUT). Регистр управления разрешает или запрещает счетчик, определяет направление счёта, управляет операционным режимом и определяет «время простоя» для всех трех фаз. Выходной регистр конфигурирует выходы, определяет полярность выходных сигналов (активный высокий или активный низкий) и задает, как обновляются выходные сигналы (немедленно или синхронно с событием).

Схема защиты (рисунок 8.3) контролирует вход EXTINT. Когда обнаруживается заданное событие на входе, схема одновременно запрещает выходы и вырабатывает запрос на прерывание EXTINT. Программное обеспечение также может запретить выходы, очистив бит разрешения выходов OE в регистре защиты (WG_PROTECT).

Запрещённые выходы переходят в неактивные состояния, заданные запрограммированной полярностью.

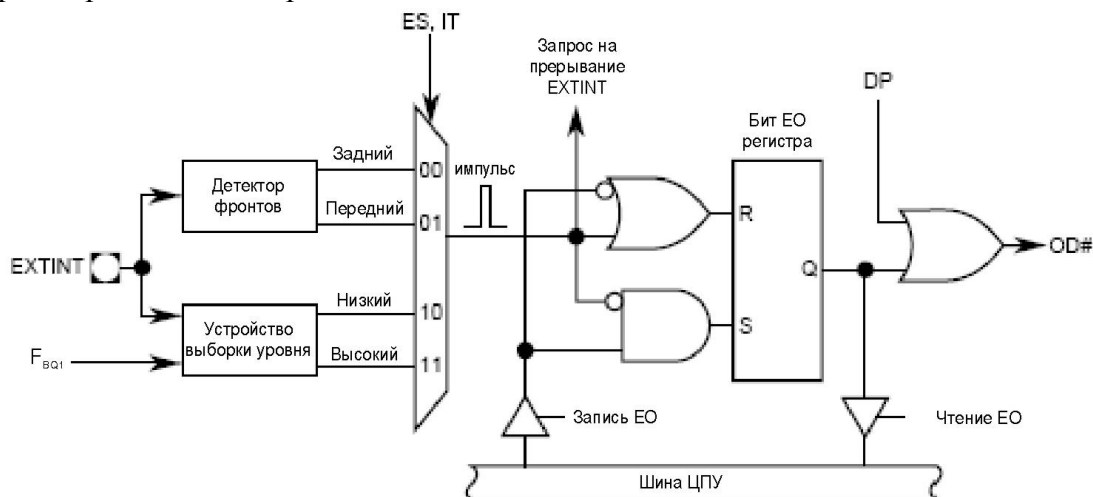


Рисунок 8.3 – Схема защиты

Буферизация и синхронизация регистров

Регистры WG_RELOAD, WG_COMPx и WG_OUTPUT буферизированы. ГФС обновляет регистры синхронно во избежание ошибочных или несимметричных рабочих циклов.

При записи в буферы WG_COMPx при остановленном счетчике (или когда в счётном регистре – ноль, или когда очищен бит разрешения счёта в регистре управления) содержимое регистров обновляется спустя половину машинного цикла.

Регистр WG_RELOAD обновляется, когда значение счётчика достигает значения перезагрузки. В регистр WG_COUNTER загружается обновленное значение

WG_RELOAD, так что новое значение перезагрузки вступает в силу для следующего цикла. В режиме 3 регистр WG_RELOAD может быть обновлён, когда происходит событие EPA. Для этого необходимо, чтобы была разрешена периферийная функция канала EPA.

Регистр WG_OUTPUT содержит бит синхронизации, который задаёт, как изменения отразятся на выходных сигналах – немедленно или синхронно с событием. Бит синхронизации не буферизирован, поэтому изменения в нём происходят немедленно. Необходимо инициализировать синхронизацию в ноль (изменения отражаются немедленно), чтобы гарантировать, что выходы находятся в желаемых состояниях при старте счётчика.

Режимы работы

Генератор формы сигнала может работать в «центрированном» или «фронтвом» режимах. В «центрированных» режимах счётчик считает и вверх, и вниз; во «фронтвых» режимах он считает только вверх. «Центрированные» режимы формируют выходные сигналы ШИМ, более эффективные для управления трехфазными индукционными двигателями переменного тока, в то время как «фронтвые» режимы формируют стандартные сигналы ШИМ. «Центрированные» выходные сигналы имеют меньше гармоник с двойным периодом «несущей», чем «фронтвые».

Начальное состояние ГФС одинаково для всех режимов работы. После системного включения питания или сброса счётчик остановлен, выводы сброшены, и все регистры очищены. Значения, записываемые в регистры, заносятся спустя половину такта.

Главные различия между «центрированными» и «фронтвыми» режимами: значение счётчика после инициализации, направление счета и условия, которые влекут за собой изменение в состоянии выходов. В таблице 8.3 приведены операции блока для «центрированного» и «фронтвого» режимов.

Т а б л и ц а 8.3 – Работа в «центрированных» и «фронтвых» режимах

Шаг	«Центрированные» режимы	«Фронтвые» режимы
1	Загрузка в WG_COUNTER значения WG_RELOAD. Выходы остаются сброшенными	Загрузка в WG_COUNTER значения 0001H. Выходы остаются сброшенными
2	Когда счетчик разрешён, он начинает обратный отсчёт. Когда WG_COUNTER достигает 1, ждёт 1 такт, затем начинает считать вверх. С началом счёта вверх выходы установлены	Когда счетчик разрешён, он начинает считать вверх. С началом счёта вверх выходы установлены
3	Когда WG_COUNTER достигает значения WG_COMPx при счёте вверх, выходы соответствующих фаз сбрасываются, и счёт вверх продолжается	Когда WG_COUNTER достигает значения WG_COMPx, выходы соответствующих фаз сбрасываются и счёт вверх продолжается
4	Когда WG_COUNTER достигает значения WG_RELOAD, начинается обратный счёт	Когда WG_COUNTER достигает значения WG_RELOAD, WG_RELOAD обновляется и происходит переход к шагу 1
5	Когда WG_COUNTER достигает значения WG_COMPx при обратном счёте, выходы соответствующих фаз устанавливаются, и обратный счёт продолжается	
6	Когда WG_COUNTER достигает 1, выходы сбрасываются, WG_RELOAD обновляется, и происходит переход к шагу 1	

Главные различия между «центрированными» и «фронтowymi» режимами – события, управляющие обновлениями регистра управления. В таблице 8.4 приведён список событий, которые могут повлечь обновление регистров, и регистры, которые обновляются в каждом режиме.

Т а б л и ц а 8.4 – Обновление регистров

Событие	«Центрированные» режимы		«Фронтowe» режимы	
	Режим 0	Режим 1	Режим 2	Режим 3
	Обновляемые регистры		Обновляемые регистры	
WG_COUNTER = =WG_RELOAD	WG_RELOAD WG_COUNTER WG_COMPx WG_OUTPUT *	WG_RELOAD WG_COUNTER WG_COMPx WG_OUTPUT *	WG_RELOAD WG_COUNTER WG_COMPx WG_OUTPUT *	WG_RELOAD WG_COUNTER WG_COMPx WG_OUTPUT *
WG_COUNTER = 1	–	WG_COMPx	–	–
Событие EPA	WG_OUTPUT *	WG_OUTPUT *	WG_OUTPUT *	WG_RELOAD WG_COUNTER WG_COMPx WG_OUTPUT *

* Регистр WG_OUTPUT обновляется при этих условиях, если его бит синхронизации установлен, иначе изменения выполняются немедленно.

«Центрированные» режимы

В «центрированных» режимах счетчик считает вниз от значения WG_RELOAD до единицы, затем меняет направление счёта и считает вверх от единицы до WG_RELOAD. Когда идет запись в регистр WG_RELOAD, в WG_COUNTER загружается значение перезагрузки. Когда установлен бит разрешения в регистре управления, счетчик начинает считать вниз и продолжает счёт до достижения единицы, ждёт один такт и начинает считать вверх вплоть до достижения значения WG_RELOAD. В этот момент WG_RELOAD обновляется, а в WG_COUNTER загружается обновлённое значение, так что новое значение перезагрузки вступает в силу в следующем такте. Счетчик продолжает обратный счёт от WG_RELOAD до единицы. Он производит «симметричный» счёт вверх и вниз, иллюстрированный треугольным сигналом на рисунке 8.4, с периодом равным удвоенному значению WG_RELOAD. На рисунке 8.5 показывается поведение выходов и прерываний в «центрированных» режимах.

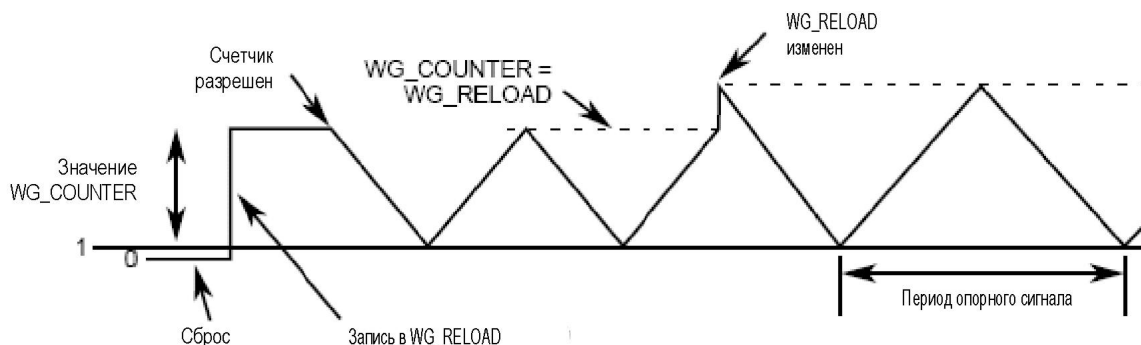


Рисунок 8.4 – Работа счётчика в «центрированных» режимах

В режиме 0 WG_COMPx и регистры WG_OUTPUT обновляются только однажды в течение периода опорного сигнала, когда счетчик достигает значения перезагрузки.

В режиме 1 эти регистры обновляются дважды в течение периода несущей: сначала, когда счетчик установлен в единицу, и второй раз, когда он достигает значения перезагрузки.

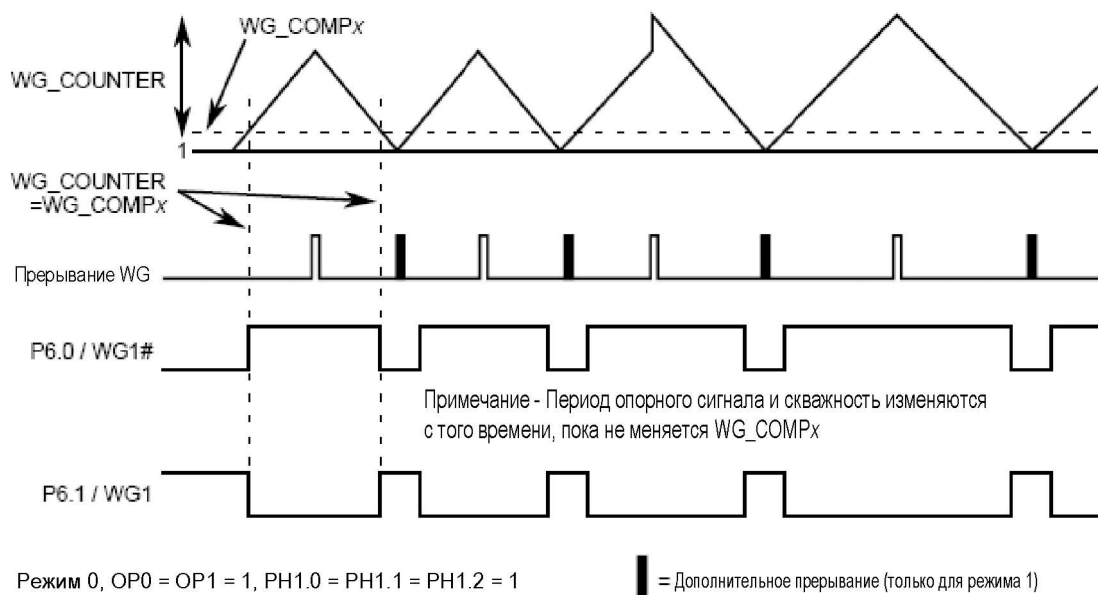


Рисунок 8.5 – Диаграмма выходных сигналов в «центрированных» режимах

«Фронтные» режимы

Во «фронтных» режимах счетчик считает вверх, начиная с единицы до значения **WG_RELOAD**. Когда записывается в регистр **WG_RELOAD**, в **WG_COUNTER** загружается 0001_H . Когда установлен бит разрешения в регистре управления, счетчик начинает считать вверх и продолжает считать, пока не достигает значения **WG_RELOAD** или (только в режиме 3) пока не произойдет событие EPA. В этот момент в **WG_COUNTER** вновь загружается 0001_H , и **WG_RELOAD** обновляется, так что новое значение перезагрузки вступает в силу в следующем такте. Счетчик продолжает считать вверх от 0001_H до **WG_RELOAD**. Он производит «гладко возрастающий» счет, иллюстрированный пилообразным сигналом на рисунке 8.6, с периодом равным значению **WG_RELOAD**. На рисунке 8.7 показывается поведение выходов и прерываний во «фронтных» режимах.

В режиме 2 регистры обновляются только однажды в течение периода опорного сигнала, когда счетчик достигает значения перезагрузки. В режиме 3 регистры также обновляются, когда происходит событие внешней функции EPA. (Необходимо конфигурировать канал EPA для этой функции; см. раздел 10 «Процессор событий EPA» для информации).

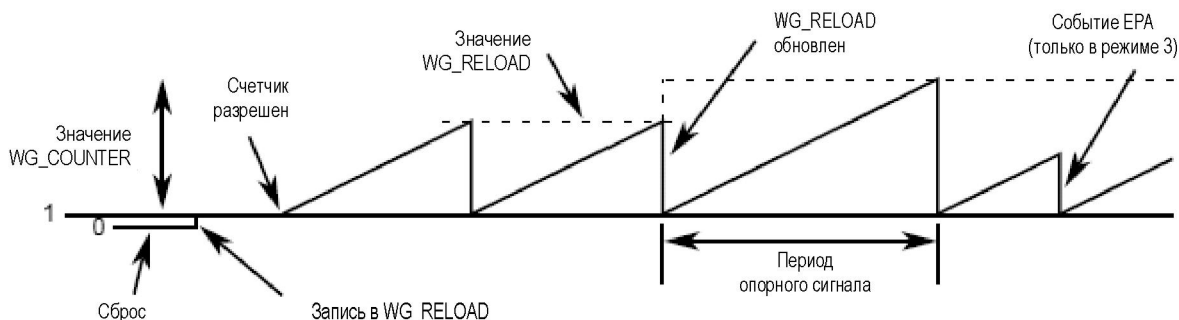
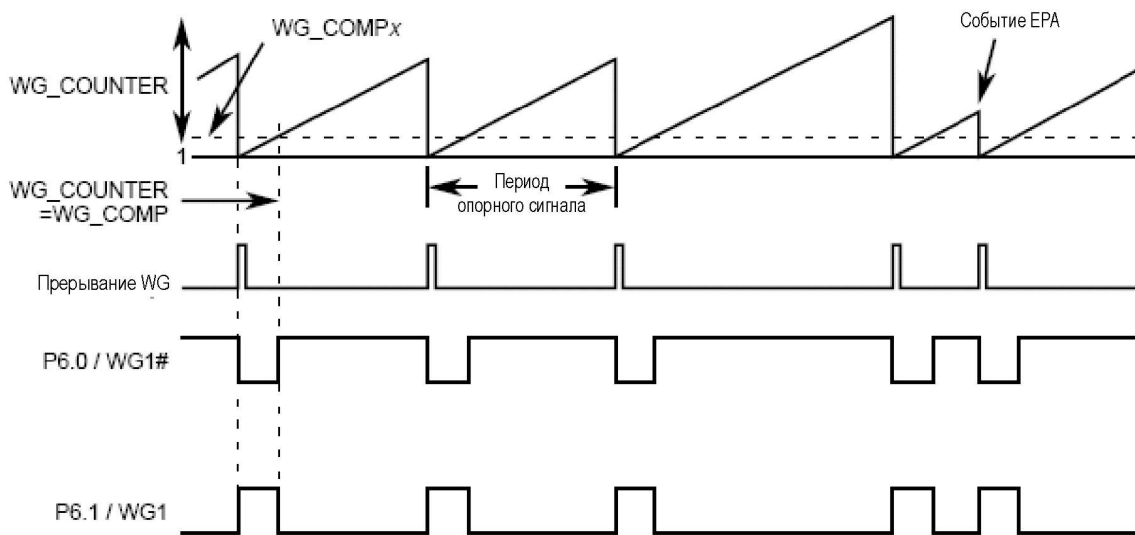


Рисунок 8.6 – Работа счётчика во «фронтных» режимах



Режим 3, OP0 = OP1 = 1, PH1.0 = PH1.1 = PH1.2 = 1
 Примечание - Период опорного сигнала и скважность изменяются с того времени, пока не меняется WG_COMPx

Рисунок 8.7 – Диаграмма выходных сигналов во «фронтных» режимах

8.4 Программирование ГФС

В этом подразделе объясняется, как конфигурировать ГФС и устанавливать его состояние.

Конфигурирование выходов

Выходы генератора формы сигнала мультиплексированы с выходом общего назначения порта 6 таким образом, что необходимо конфигурировать их как сигналы специальных функций, чтобы использовать их как выходы ГФС. Регистр WG_OUTPUT (рисунок 8.8) конфигурирует выходы, устанавливает полярность выхода и управляет, как изменяются выходные сигналы, синхронно с событием или немедленно.

Четыре бита WG_OUTPUT независимы от ГФС. Они конфигурируют выходы для широтно-импульсного модулятора (ШИМ), который также совместно использует выходы с портом 6. P6 и PE6 биты управляют выводом P6.6/PWM0, а P7 и PE7 биты управляют выводом P6.7/PWM1. Их размещение в этом регистре позволяет конфигурировать все выходы порта 6 единственной записью в WG_OUTPUT.

В таблице 8.5 показываются комбинации битов, необходимые для задания высокого или низкого уровня выходов ГФС или подключения их к сигналам WG_EVEN или WG_ODD. Необходимо обратить внимание, что PHx.2 всегда устанавливается для выбора функции сигнала ГФС (очистка PHx.2 выбирает функцию общего назначения порта ввода-вывода). В графе «Полярность выхода» показывается полярность выходных сигналов. На рисунках в таблице 8.5 показана скважность приблизительно 15 %, и в этих случаях длительность высокого уровня сигналов увеличивается с увеличением «времени простоя».

Т а б л и ц а 8.5 – Конфигурация выходов ГФС

PHx.2	PHx.1	PHx.0	Значение выхода		Полярность выхода	
			WGx	WGx#	WGx	WGx#
1	0	0	Низкий	Низкий	Всегда низкий	Всегда низкий
1	0	1	Низкий	WG_EVEN#	Всегда низкий	
1	1	0	WG_ODD	Низкий		Всегда низкий
1	1	1	WG_ODD	WG_EVEN		

Примечание – В этой таблице подразумеваются активные высокие выходные сигналы (OP1 = OP0 = 1).

Регистр WG_OUTPUT (генератор формы сигнала)

Адрес: 1FC0_H

Состояние после сброса: 0000_H

Регистр конфигурации выходов ГФС (WG_OUTPUT) управляет конфигурацией выводов генератора формы сигнала и модуля ШИМ. ГФС и модуль ШИМ совместно используют выводы порта 6. Наличие этих битов управления в одном регистре дает возможность конфигурировать весь порт 6 одной записью в WG_OUTPUT.

15							8
OP1	OP0	SYNC	PE7	PE6	PH3.2	PH2.2	PH1.2
7							0
P7	P6	PH3.1	PH3.0	PH2.1	PH2.0	PH1.1	PH1.0

Рисунок 8.8 – Регистр конфигурации выходов ГФС (WG_OUTPUT)

Т а б л и ц а 8.6 – Разряды регистра конфигурации выходов ГФС (WG_OUTPUT)

Номер бита	Мнемоника	Функция
1	2	3
15	OP1	Полярность выхода. Выбирает полярность выхода для сигналов инверсных фаз WG1#, WG2# и WG3#. 0 активно-низкие выходы, 1 активно-высокие выходы
14	OP0	Полярность выхода. Выбирает полярность выхода для сигналов прямых фаз WG1, WG2 и WG3. 0 активно-низкие выходы, 1 активно-высокие выходы
13	SYNC	Синхронизация. Выбирает, как происходит обновление регистра WG_OUTPUT: синхронно ли с другим событием или немедленно после того, как он изменится. 0 обновляет WG_OUTPUT немедленно, 1 обновляет WG_OUTPUT синхронно с событием. Чтобы гарантировать, что выходы находятся в нужных состояниях при запуске ГФС, необходимо при инициализации очистить этот бит, затем установить его позже, чтобы в последующем WG_OUTPUT обновлялся синхронно с событием. (В таблице 8.4 перечисляются события, которые обновляют WG_OUTPUT в каждом режиме)
12	PE7	P6.7/PWM1. Выбирает функцию порта или функцию выхода ШИМ на выводе P6.7/PWM1. 0 P6.7 1 PWM1
11	PE6	P6.6/PWM0. Выбирает функцию порта или функцию выхода ШИМ на выводе P6.6/PWM0. 0 P6.6 1 PWM0

Окончание таблицы 8.6

1	2	3
10	RH3.2	Функция фазы 3. Выбирает или функцию порта, или функцию выхода ГФС для выводов P6.4/WG3# и P6.5/WG3. 0 P6.4, P6.5 1 WG3#, WG3
9	RH2.2	Функция фазы 2. Выбирает или функцию порта, или функцию выхода ГФС для выводов P6.2/WG2# и P6.3/WG2. 0 P6.2, P6.3 1 WG2#, WG2
8	RH1.2	Функция фазы 1. Выбирает или функцию порта или функцию выхода ГФС для выводов P6.0/WG1# и P6.1/WG1. 0 P6.0, P6.1 1 WG1#, WG1
7	P7	P6.7/PWM1 значение. Нужное значение P6.7/PWM1 записывают в этот бит
6	P6	P6.6/PWM0 значение. Нужное значение P6.6/PWM0 записывают в этот бит
5, 4	RH3.1, H3.0	P6.4/WG3#, P6.5/WG3 значения. Нужные значения выходов записывают в эти биты, (см. таблицу 8.5)
3, 2	RH2.1, H2.0	P6.2/WG2#, P6.3/WG2 значения. Нужные значения выходов записывают в эти биты, (см. таблицу 8.5)
1, 0	RH1.1, H1.0	P6.0/WG1#, P6.1/WG1. Нужные значения выходов записывают в эти биты, (см. таблицу 8.5)

Управление схемой защиты и выработкой прерывания EXTINT

Регистр защиты (рисунок 8.9) управляет схемой защиты и запросами прерывания EXTINT.

Регистр WG_PROTECT

Адрес: 1FCE_H

Состояние после сброса: F0_H

Регистр защиты ГФС (WG_PROTECT) разрешает и запрещает выходы и схему защиты. Он также выбирает прерывания EXTINT или по уровню, или по фронту и выбирает, который уровень или фронт произведет запрос на прерывание EXTINT.

7	–	–	–	–	ES	IT	DP	EO	0
---	---	---	---	---	----	----	----	----	---

Рисунок 8.9 – Регистр защиты ГФС (WG_PROTECT)

Т а б л и ц а 8.7 – Разряды регистра защиты ГФС (WG_PROTECT)

Номер бита	Мнемоника	Функция
7 – 4	–	Зарезервирован. Для совместимости с будущими устройствами записывать нули в эти биты
3, 2	ES, IT	Разрешение выборки и тип прерывания. Бит ES выбирает, что схема защиты производит выборку по уровню сигнала EXTINT или детектирует перепад сигнала (фронт), в то время как бит IT задаёт, какое значение фронта или уровня вызывает запрос на прерывание. Возможные комбинации следующие: ES IT Событие 0 0 задний фронт 0 1 передний фронт 1 0 низкий уровень 1 1 высокий уровень
1	DP	Запрещение защиты. Этот бит разрешает и запрещает схему защиты. 0 разрешение защиты 1 запрещение защиты
0	EO	Разрешение выходов. Этот бит разрешает и запрещает выходы. 0 запрещение выходов 1 разрешение выходов

Задание периода опорного сигнала и скважности

Регистр перезагрузки (WG_RELOAD) и регистры сравнения фаз (WG_COMPx) управляют периодом опорного сигнала (несущей) и скважностью. Чтобы установить период опорного сигнала, его значение записывается в регистр перезагрузки (рисунок 8.10). Чтобы определить отрезок времени, в течение которого соответствующие выходы останутся установленными, его значение записывается в каждый регистр сравнения фаз.

Регистр WG_RELOAD

Адрес: 1FC8_H

Состояние сброса: 0000_H

Регистр перезагрузки ГФС (WG_RELOAD) и регистры сравнения фаз (WG_COMPx) управляют периодом опорного сигнала (несущей) и скважностью. Запись значения в регистр перезагрузки устанавливает период опорного сигнала.

При изменении значения WG_RELOAD изменяются и период опорного сигнала, и скважность, потому что выводы останутся установленными в течение постоянного отрезка времени, в то время как счетчик требует большего времени, чтобы закончить цикл. Чтобы изменить период «несущей», не изменяя скважность, необходимо пропорционально изменить и WG_RELOAD, и WG_COMPx в одно и то же время, немедленно после прерывания.

15	0
Перегружаемые данные	

Рисунок 8.10 – Регистр WG_RELOAD перезагрузки ГФС

Т а б л и ц а 8.8 – Разряды регистра WG_RELOAD перезагрузки ГФС

Номер бита	Функция
15 – 0	<p>Перезагрузка. Этот регистр определяет период опорного сигнала (несущего). Используются следующие формулы, чтобы вычислить период «несущей» и скважность:</p> $T_{\text{CARRIER}} = (\text{множитель} \times \text{WG_RELOAD}) / F_{\text{BQ1}},$ $\text{скважность} = (\text{WG_COMP}_x / \text{WG_RELOAD}) \times 100 \%,$ <p>где T_{CARRIER} – период несущей, мкс; F_{BQ1} – частота входного сигнала на входе BQ1, МГц; <i>множитель</i> равен четырем для «центрированных» режимов; равен двум для «фронтowych» режимов; WG_RELOAD – 16-битное значение $\text{WG_RELOAD} \geq \text{WG_COMP}_x$; WG_COMP_x – 16-битное значение $\text{WG_COMP}_x \leq \text{WG_RELOAD}$</p>

Регистр WG_COMP_x (x = 1, 2, 3)

Адреса: FC2_H, 1FC4_H, 1FC6_H

Состояние сброса: 0000_H

Регистр сравнения фаз (WG_COMP_x) управляет скважностью сигнала каждой фазы. Записывается значение в каждый регистр сравнения фаз, чтобы определить отрезок времени, в течение которого соответствующие выходы останутся установленными.

Изменение значения WG_RELOAD изменяет и период несущей и скважность, потому что выводы остаются установленными в течение постоянного отрезка времени, в то время как счетчик требует большего времени, чтобы закончить цикл. Чтобы изменить период опорного сигнала, не изменяя скважность, необходимо пропорционально изменить WG_RELOAD и WG_COMP_x в одно и то же время, немедленно после прерывания.



Рисунок 8.11 – Регистр сравнения фаз (WG_COMP_x)

Т а б л и ц а 8.9 – Разряды регистра сравнения фаз (WG_COMP_x)

Номер бита	Функция
15 – 0	<p>Сравнение. Эти биты определяют отрезок времени, в течение которого соответствующие выходы установлены. Используются следующие формулы, чтобы вычислить время установленного выхода и скважность:</p> $T_{\text{OUTPUT}} = (\text{множитель} \times \text{WG_RELOAD}) / F_{\text{BQ1}};$ $\text{скважность} = (\text{WG_COMP}_x / \text{WG_RELOAD}) \times 100 \%,$ <p>где T_{OUTPUT} – полное время, когда выход установлен, мкс; F_{BQ1} – частота входного сигнала на входе BQ1, МГц; <i>множитель</i> равен четырем для «центрированных» режимов и двум для «фронтowych» режимов; WG_RELOAD – 16-битное значение $\text{WG_RELOAD} \geq \text{WG_COMP}_x$; WG_COMP_x – 16-битное значение $\text{WG_COMP}_x \leq \text{WG_RELOAD}$</p>

Определение режима работы, «времени простоя» и старт счётчика

Регистр управления ГФС (WG_CONTROL) (рисунок 8.12) определяет «время простоя» и режим работы, разрешает и запрещает счётчики. Бит только для чтения CS указывает текущее направление счёта.

Регистр WG_CONTROL

Адрес: 1FCC_H

Состояние сброса: 00C0_H

15	8						
–	M2	M1	M0	CS	EC	DT9	DT8
7	0						
DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0

Рисунок 8.12 – Регистр управления ГФС (WG_CONTROL)

Т а б л и ц а 8.10 – Разряды регистра управления ГФС (WG_CONTROL)

Номер бита	Мнемоника	Функция
15	–	Зарезервирован. Для совместимости с последующими устройствами записывается ноль в этот бит
14– 12	M2– M0	Режим работы. Это поле управляет режимом работы ГФС. M2 M1 M0 Режим 0 0 0 0 «центрированный»; регистры обновляются однажды 0 0 1 1 «центрированный»; регистры обновляются дважды 0 1 0 2 «фронтной»; регистры обновляются однажды 0 1 1 3 «фронтной»; регистры обновляются дважды
11	CS	Состояние счётчика. Этот бит только для чтения указывает, считает ли счетчик вверх или в обратном порядке. 0 счет вниз 1 счет вверх
10	EC	Разрешение счетчика. Этот бит запускает или останавливает счетчик. 0 запретить (остановить) счетчик 1 разрешить (запустить) счетчик
9 – 0	DT9 – DT 0	«Время простоя» (Dead Time). Это поле определяет «время простоя» для всех трех фаз. Используется следующая формула, чтобы вычислить соответствующее значение DT_VALUE: $DT_VALUE = T_{DEAD} \times F_{BQ1} / 2$, где T_{DEAD} – «время простоя», мкс; F_{BQ1} – частота входного сигнала на входе BQ1, МГц

8.5 Определение состояния ГФС

Читается регистр WG_CONTROL (рисунок 8.12), чтобы определить текущее значение «времени простоя», состояние счётчика, направление счета и режим работы. Читается WG_COUNTER (рисунок 8.13), чтобы определить текущее значение счётчика.

Регистр WG_COUNTER

Адрес: 1FCA_H

Состояние сброса: XXXX_H

Читается регистр-счетчик генератора формы сигнала (WG_COUNTER), чтобы определить текущее значение счётчика.

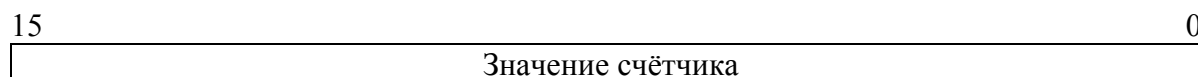


Рисунок 8.13 – Регистр-счётчик WG_COUNTER генератора формы сигнала

Т а б л и ц а 8.11 – Разряды регистра-счётчика WG_COUNTER генератора формы сигнала

Номер бита	Функция
15 – 0	Значение счётчика. Этот регистр отражает текущее значение счётчика

8.6 Разрешение прерываний генератора формы сигнала

Генератор формы сигнала может производить два типа запросов на прерывание: запрос на прерывание WG вызывается счетчиком, в то время как прерывание EXTINT вызывается внешним событием.

В режиме 0 запрос на прерывание WG производится однажды в период, когда счетчик достигает значения WG_RELOAD. В режиме 1 запрос на прерывание WG производится дважды в течение периода: сначала – когда счетчик достигает единицы и ещё раз – когда он достигает значения WG_RELOAD. Во «фронтных» режимах запрос на прерывание WG производится однажды в конце каждого периода, когда в счетчик загружается «1».

Схема защиты управляет прерыванием EXTINT. Два бита в регистре защиты управляют, какой тип внешнего события произведет запрос на прерывание: задний или передний фронт или низкий или высокий уровень.

Схема обнаружения фронта требует, чтобы сигнал оставался установленным в течение, по крайней мере, 2 тактов, которые будут отсчитаны от нужного фронта. Схема выборки требует, чтобы сигнал оставался установленным в течение, по крайней мере, 24 тактов, которые будут отсчитаны от нужного уровня. Схема производит выборки входного уровня три раза в течение этого 24-тактового периода и признает сигнал «правильным», только если это подтверждено для каждой выборки. Осуществление выборки уровня полезно для сред, в которых шумовые пики могли бы причинить нежелательные прерывания, если бы использовалось обнаружение фронта.

Чтобы разрешить прерывания, необходимо установить соответствующие биты масок в регистре маски и выполнить команду EI, чтобы разрешить обслуживание прерываний. Необходимо сосчитать регистр ожидания прерываний, чтобы определить любые отложенные прерывания.

Время простоя (Dead Time) и скважность

Короткое «время простоя» имеет небольшое влияние на скважность, если импульс относительно широк. Однако, более длинное «время простоя» с короткими импульсами может изменить скважность, как показано на рисунке 8.14. Нельзя использовать минимальную ширину импульса, так как возможно пропадание импульсов в течение периода, если «время простоя» больше, чем ширина импульса. Из этих соображений время импульса должно быть не меньше $3 \times T_{DEAD}$ (T_{DEAD} - «время простоя»).

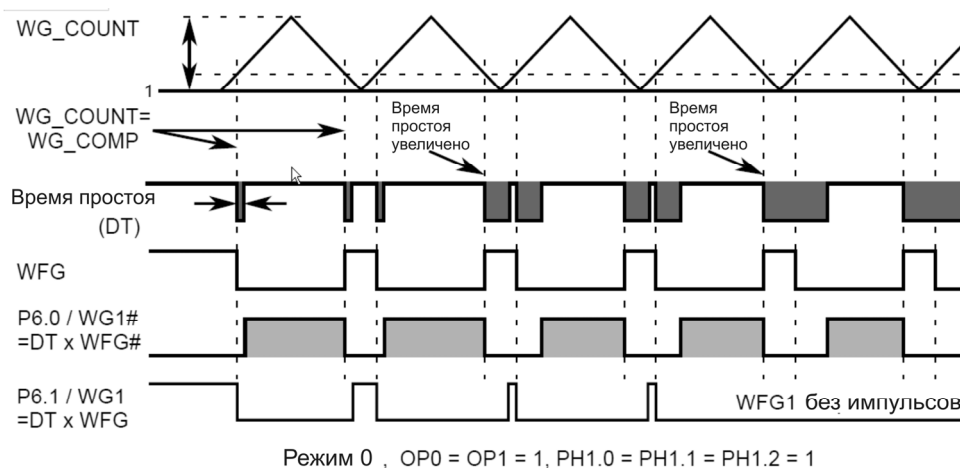


Рисунок 8.14 – «Время простоя» и скважность

9 Широтно-импульсный модулятор

Широтно-импульсный модулятор ШИМ (PWM – Pulse Width Modulator) имеет два выхода, каждый из которых может выводить ШИМ сигнал с фиксированной программируемой частотой и переменной скважностью. Эти выходы могут использоваться, чтобы управлять двигателями, которым необходимы нефильТРованные цифровые сигналы ШИМ для оптимальной эффективности, или они могут быть фильтрованы, чтобы получить сглаженный аналоговый сигнал.

В этом разделе проводится краткий функциональный обзор широтно-импульсного модулятора, описано как он программируется и обеспечивается типовая схема для преобразования ШИМ сигналов в аналоговые сигналы.

9.1 Краткий обзор функционирования блока ШИМ

Блок ШИМ имеет два канала, каждый из которых состоит из регистра управления (PWMx_CONTROL), буфера, компаратора, RS-триггера и выходного блока. Два других компонента, счетчик на восемь битов (PWM_COUNT) и регистр периода (PWM_PERIOD) для двух каналов модуля ШИМ, дополняют схему (рисунок 9.1).

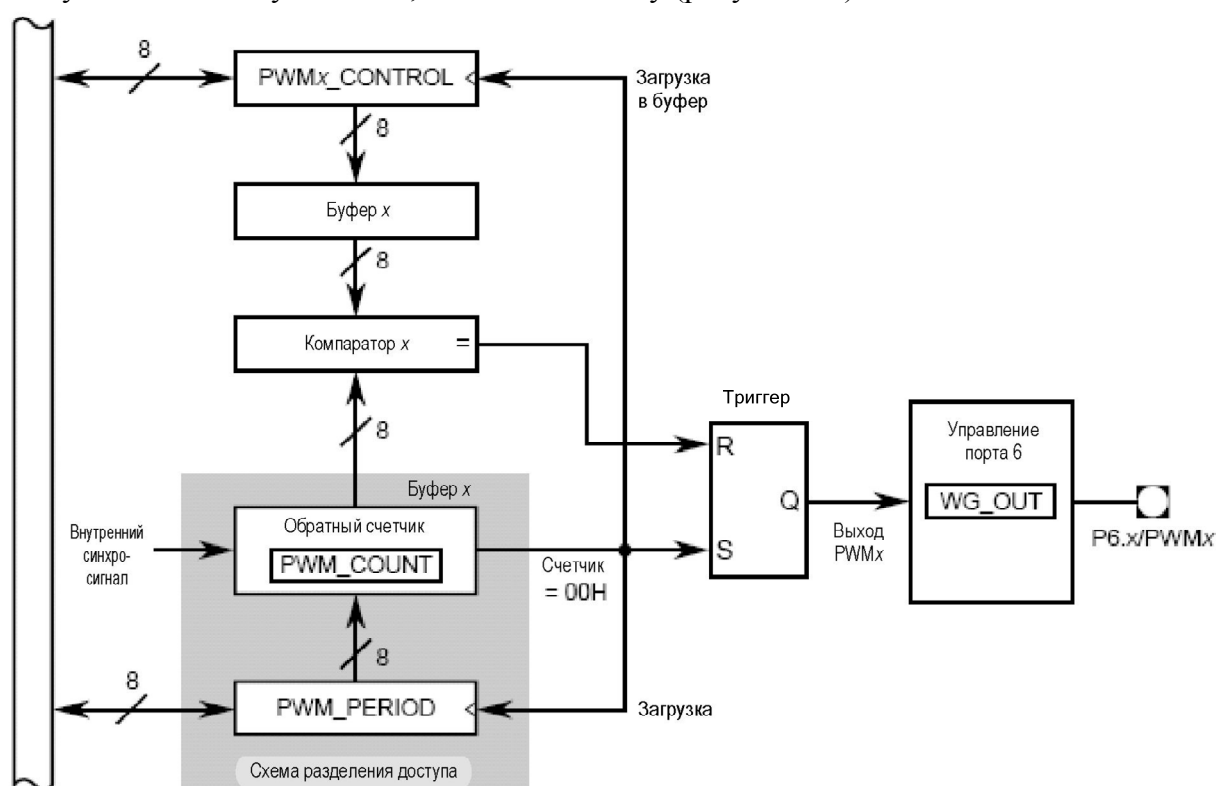


Рисунок 9.1 – Блок-схема блока ШИМ

9.2 Сигналы и регистры блока ШИМ

В таблице 9.1 описываются ШИМ сигналы, а в таблице 9.2 кратко описаны управляющие регистры и регистры состояния.

Т а б л и ц а 9.1 – ШИМ сигналы

Вывод порта	ШИМ сигнал	Тип ШИМ сигнала	Описание
P6.6	PWM0	Выход	Широтно-импульсный выход 0 канала с повышенной нагрузочной способностью
P6.7	PWM1	Выход	Широтно-импульсный выход 1 канала с повышенной нагрузочной способностью

Т а б л и ц а 9.2 – Регистры управления и состояния ШИМ

Мнемонический	Адрес	Описание												
PWM0_CONTROL PWM1_CONTROL	1FB0 _H 1FB2 _H	Скважность ШИМ. Этот регистр управляет скважностью ШИМ. Обнуление этого регистра ШИМ приводит к формированию сигнала со скважностью, близкой к нулю (0 %). FF _H в этом регистре заставит ШИМ сформировать сигнал со скважностью, близкой к единице (99,6 %)												
PWM_PERIOD	1FB4 _H	Период ШИМ. Этот регистр содержит запрограммированное значение, которое определяет период сигнала ШИМ. Значение перегружается в счетчик каждый раз, когда счетчик переустанавливается в FF _H												
PWM_COUNT	1FB6 _H	Счетчик ШИМ. Это доступный только для чтения регистр, содержит текущее значение счетчика обратного счета												
WG_OUTPUT	1FC0 _H	Выход генератора формы сигнала. Биты 11 и 12 (PE6 и PE7) определяют, функционирует соответствующий вывод как стандартный вывод порта ввода-вывода или как выход ШИМ. Биты 6 и 7 (P6 и P7) определяют выходное состояние вывода, когда выбрана функция выходного порта. <table style="margin-left: 40px;"> <tr> <td>PE_x</td> <td>P_x</td> <td>на выходе</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>X</td> <td>ШИМ выход</td> </tr> </table>	PE _x	P _x	на выходе	0	0	0	0	1	1	1	X	ШИМ выход
PE _x	P _x	на выходе												
0	0	0												
0	1	1												
1	X	ШИМ выход												

9.3 Работа широтно-импульсного модулятора

Регистр периода (PWM_PERIOD) управляет частотой обоих выходных ШИМ сигналов. Каждый регистр управления (PWM_x_CONTROL) управляет скважностью (отношением ширины импульса к периоду, в процентах) ШИМ сигнала на соответствующем выводе. Каждый регистр управления содержит 8-битное значение, которое загружается в буфер, когда 8-битный счетчик меняет значение из 00_H в FF_H. Компараторы сравнивают содержимое буферов со значением счетчика. Так как значение, записанное в регистр управления, сохраняется в буфере, необходимо записать новое 8-битное значение в PWM_x_CONTROL в любое время. Однако компараторы не определяют новое значение, пока счетчик не завершил обратный счет по предыдущему значению. Новое значение используется в течение следующего периода ШИМ сигнала.

Счетчик считает в обратном порядке к 00_H, в это время ШИМ выход переключается в состояние высокого уровня, счетчик загружает содержимое регистра PWM_PERIOD, и содержимое регистров управления загружается в буферы. На выводе ШИМ уровень остается высоким, пока значение счетчика соответствует значению в буфере, иначе на выводе будет низкий уровень. Необходимо читать регистр PWM_COUNT, чтобы знать текущее значение счетчика. Когда счетчик переустанавливается снова (то есть, когда происходит переполнение), вывод переключается в высокий уровень. Загрузка в PWM_x_CONTROL значения 00_H переводит вывод в состояние низкого уровня. На рисунке 9.2 показаны типичные формы сигналов на ШИМ выходах.

Примечание – Значение регистра PWM_x_CONTROL и соответствующая результирующая скважность (Duty Cycle) на рисунке 9.2 являются верными только тогда, когда значение регистра PWM_PERIOD равно FF_H.

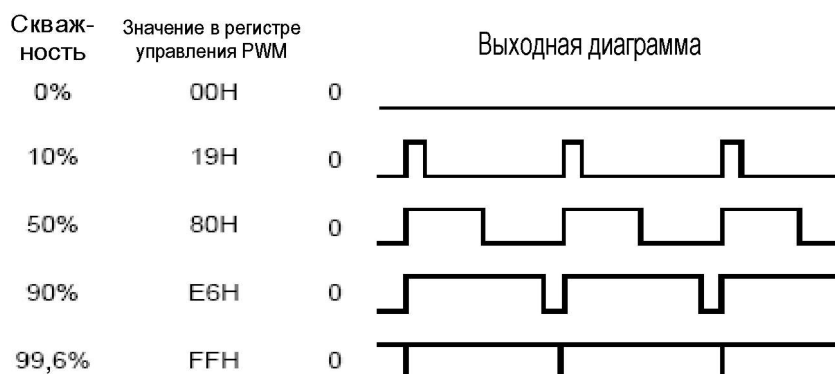


Рисунок 9.2 – Форма ШИМ сигнала

9.4 Программирование частоты и периода

Частота входного сигнала на BQ1 (F_{BQ1}) и содержимое регистра PWM_PERIOD определяет частоту сигнала (F_{PWM}) и период (T_{PWM}) на выводе ШИМ. В таблице 9.3 показывается частота сигнала на выводе ШИМ при определенном значении F_{BQ1} с различными значениями PWM_PERIOD. Чтобы вычислить желательные значения периода и частоты для сигнала ШИМ вывода и записать их в регистр PWM_PERIOD, используют следующие формулы:

$$T_{PWM}(\text{мкс}) = \frac{512 \times (\text{PWM_PERIOD} + 1)}{F_{BQ1}}, \quad (9.1)$$

$$F_{PWM}(\text{МГц}) = \frac{F_{BQ1}}{512 \times (\text{PWM_PERIOD} + 1)}, \quad (9.2)$$

где PWM_PERIOD – 8-битное значение для загрузки в регистр PWM_PERIOD;
 F_{BQ1} – частота входного сигнала на выводе BQ1 в МГц;
 T_{PWM} – период выходного сигнала на выводе ШИМ в мкс;
 F_{PWM} – частота выходного сигнала на выводе ШИМ в МГц.

Т а б л и ц а 9.3 – Частота выходного сигнала ШИМ (F_{PWM})

PWM_PERIOD	Частота выходного сигнала ШИМ (F_{PWM})		
	$F_{BQ1}=8$ МГц	$F_{BQ1}=10$ МГц	$F_{BQ1}=16$ МГц
00H	15,6 кГц	19,5 кГц	31,2 кГц
0FH	976,6 Гц	1220,7 Гц	1953,1 Гц
1FH	488,3 Гц	610,3 Гц	976,6 Гц
2FH	325,5 Гц	406,9 Гц	651,0 Гц
3FH	244,1 Гц	395,2 Гц	488,3 Гц
4FH	195,3 Гц	244,1 Гц	390,6 Гц
5FH	162,8 Гц	203,4 Гц	325,5 Гц
6FH	139,5 Гц	174,4 Гц	279,0 Гц
7FH	122,1 Гц	152,6 Гц	244,1 Гц
8FH	108,5 Гц	135,6 Гц	217,0 Гц
9FH	97,7 Гц	122,1 Гц	195,3 Гц
AFH	88,8 Гц	111,0 Гц	177,6 Гц
BFH	81,4 Гц	101,7 Гц	162,8 Гц
CFH	75,1 Гц	93,9 Гц	150,2 Гц
DFH	69,7 Гц	87,2 Гц	139,5 Гц
EFH	65,1 Гц	81,4 Гц	130,2 Гц
FFH	61,0 Гц	76,0 Гц	122,0 Гц

Регистр PWM_PERIOD

Адрес: 1FB4_H

Состояние сброса: 00_H

Регистр периода ШИМ (PWM_PERIOD) управляет периодом выходных ШИМ сигналов. Он содержит значение, которое определяет число состояний счета, необходимое для инкрементирования счетчика ШИМ. Значение PWM_PERIOD загружается в регистр счетчика периода ШИМ всякий раз, когда значение в счетчике равно нулю.

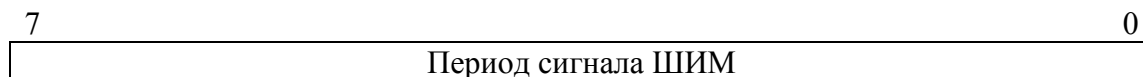


Рисунок 9.3 – Регистр периода ШИМ PWM_PERIOD

Т а б л и ц а 9.4 – Разряды регистра периода ШИМ PWM_PERIOD

Номер разряда	Функция
7 – 0	Период сигнала ШИМ. Этот регистр управляет периодом сигнала на выводах ШИМ. Значение PWM_PERIOD загружается в регистр счетчика периода ШИМ всякий раз, когда значение в счетчике равно нулю

9.5 Программирование скважности (Duty cycle)

Значения, записанные в PWM_x_CONTROL и в регистр PWM_PERIOD, управляют шириной импульса высокого уровня, фактически управляя скважностью. 8-битное значение, записанное в регистр управления, загружается в буфер, и это значение используется в течение следующего периода. Для вычисления желаемой скважности по данным значениям PWM_x_CONTROL и PWM_PERIOD и последующей записи этих значений в соответствующие регистры применяют следующие формулы:

$$\text{скважность (в \%)} = \frac{\text{PWM}_x_CONTROL}{\text{PWM_PERIOD} + 1} \times 100\% ; \quad (9.3)$$

$$\text{ширина импульса (в мкс)} = \frac{\text{Скважность} \times T_{\text{PWM}}}{100} , \quad (9.4)$$

где PWM_x_CONTROL – 8-битное значение для загрузки регистра PWM_x_CONTROL;
PWM_PERIOD – 8-битное значение для загрузки регистра PWM_PERIOD;
ширина импульса равна ширине каждого импульса высокого уровня;
T_{PWM} – период сигнала на выводе ШИМ в мкс.

Регистр PWM_x_CONTROL, (x = 0, 1)

Адреса: 1FB0_H, 1FB2_H

Состояние сброса: 00_H

Регистр управления ШИМ (PWM_x_CONTROL) определяет скважность сигнала ШИМ для канала x. Ноль, загруженный в этот регистр ШИМ, устанавливает состояние низкого уровня сигнала (постоянное, скважность равна 0 %). FF_H в этом регистре устанавливает для ШИМ максимальную скважность (99,6 %) и состояние постоянного высокого уровня сигнала.

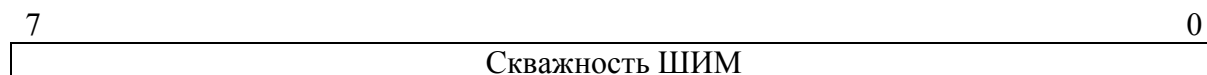


Рисунок 9.4 – Регистр управления ШИМ (PWM_x_CONTROL)

Т а б л и ц а 9.5 – Разряды регистра управления ШИМ (PWMx_CONTROL)

Номер разряда	Функция
7 – 0	Скважность ШИМ. Этот регистр управляет скважностью ШИМ. Ноль, загруженный в этот регистр ШИМ, устанавливает состояние постоянного низкого уровня сигнала (скважность равна 0 %). FF _H в этом регистре устанавливает для ШИМ максимальную скважность (99,6 %) и состояние постоянного высокого уровня сигнала

Примеры вычислений

Предположим, что F_{BQ1} равняется 16 МГц и значение, записанное в регистр PWM_PERIOD – FF_H; таким образом, ожидаемый период сигнала на выходе ШИМ равен 8,19 мс. Если значение в PWMx_CONTROL равняется 8A_H (десятичное 138), ширина высокого уровня сигнала равна 4,42 мс и низкого уровня сигнала 3,77 мс, что и составляет 8,19 мс для полного периода. Результирующая скважность равна 54 %.

Чтение текущего значения обратного счётчика

Можно считать значение регистра PWM_COUNT, чтобы найти текущее значение обратного счетчика (таблица 9.6, рисунок 9.5).

Регистр PWM_COUNT (только чтение)

Адрес: 1FB6_H

Состояние сброса: 00_H

Регистр периода счета ШИМ (PWM_COUNT) обеспечивает текущее значение периода счетчика обратного счета.

7	0
Значение счета PWM	

Рисунок 9.5 – Регистр периода счета ШИМ (PWM_COUNT)

Т а б л и ц а 9.6 – Разряды регистра периода счета ШИМ (PWM_COUNT)

Номер разряда	Функция
7 – 0	Значение периода счета PWM. Регистр содержит текущее значение периода счетчика обратного счета

Разрешение ШИМ функции выводов

Каждый ШИМ выход мультиплексирован с выводом порта, так что необходимо конфигурировать его как сигнал вывода специальной функции перед использованием функции ШИМ. Чтобы определить, функционирует ли соответствующий вывод как стандартный вывод порта ввода-вывода или как ШИМ вывод, необходимо записать число в регистр WG_OUTPUT (см. таблицу 9.8 и рисунок 9.6). В таблице 9.7 показывается выбор альтернативной функции порта (ШИМ) записью регистра WG_OUTPUT.

Т а б л и ц а 9.7 – Альтернативные функции ШИМ вывода

Выход ШИМ	Дополнительная функция порта	Выход ШИМ разрешен
PWM0	P6.6	когда WG_OUT.11 = 1
PWM1	P6.7	когда WG_OUT.12 = 1

Регистр WG_OUTPUT (генератор формы сигнала)Адрес: 1FC0_HСостояние сброса: 0000_H

Регистр управления функцией выхода генератора формы сигнала (WG_OUTPUT) управляет функционированием выходов генератора формы сигнала ГФС и блока ШИМ. Генератор формы сигнала и блок ШИМ совместно используют выводы порта 6. Наличие битов управления в отдельном регистре позволяет формировать все выводы порта записью единственного регистра WG_OUTPUT.

15	8						
OP1	OP0	SYNC	PE7	PE6	PH3.2	PH2.2	PH1.2
7	0						
P7	P6	PH3.1	PH3.0	PH2.1	PH2.0	PH1.1	PH1.0

Рисунок 9.6 – Регистр конфигурации выводов порта 6 (WG_OUTPUT)

Т а б л и ц а 9.8 – Регистр генератора формы сигнала (WG_OUTPUT) – конфигурация выводов порта 6

Номер разряда	Мнемоника	Функция
15	OP1	Полярность выхода
14	OP0	Полярность выхода
13	SYNC	Синхронизация
12	PE7	P6.7/PWM1 функция. Выбирает или функцию порта, или ШИМ функцию вывода P6.7/PWM1. 1 PWM1 0 P6.7
11	PE6	P6.6/PWM0 функция. Выбирает или функцию порта, или ШИМ функцию вывода P6.6/PWM0. 1 PWM0 0 P6.6
10	PH3.2	Функция фазы 3
9	PH2.2	Функция фазы 2
8	PH1.2	Функция фазы 1
7	P7	P6.7/PWM1 значение. В этот бит записывают желаемое значение P6.7/PWM1
6	P6	P6.6/PWM0 значение. В этот бит записывают желаемое значение P6.6/PWM0
5, 4	PH3.1, PH3.0	P6.4/WG3#, P6.5/WG3 значения
3, 2	PH2.1, PH2.0	P6.2/WG2 #, P6.3/WG2 значения
1, 0	PH1.1, PH1.0	P6.0/WG1 #, P6.1/WG1 значения

Создание аналоговых выходов

Блоки ШИМ вырабатывают прямоугольные сигналы, которые имеют изменяемые период и скважность. Подключение фильтра к этим выводам позволит создать сглаженный аналоговый сигнал. Чтобы сформировать сигнал, изменяющийся с желаемыми аналоговыми параметрами, сначала надо поставить буфер для формирования сигнала и затем фильтр. Это или простая емкостно-резистивная цепь или с активным фильтром. На рисунке 9.7 представлена типовая блок-схема устройства, необходимого для создания сглаженного аналогового сигнала.

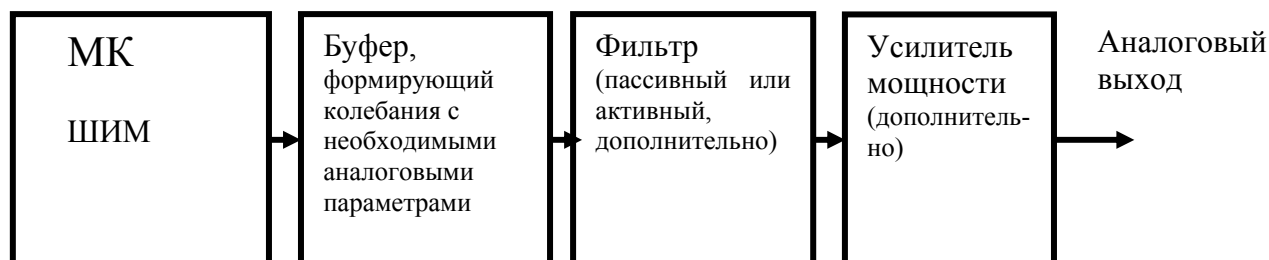


Рисунок 9.7 – Блок-схема преобразования цифрового выхода в аналоговый

На рисунке 9.8 показана типовая схема, используемая для малых выходных токов (меньше, чем 100 мкА). Необходимо рассчитать температуру и дрейф источника питания при отборе компонентов для внешней цифро-аналоговой схемы. С надлежащими параметрами, используя ШИМ, можно сделать очень точные 8-битные цифро-аналоговые преобразователи.

Необходимые номиналы R и C рассчитываются для конкретного применения.

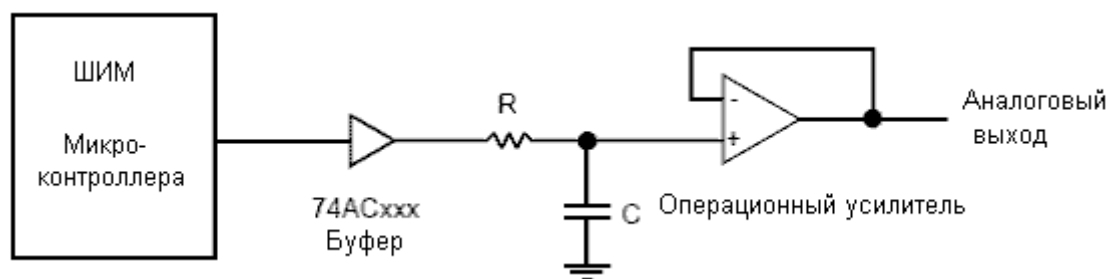


Рисунок 9.8 – Подключение ШИМ к аналоговой конверсионной схеме

10 Процессор событий ЕРА

Управление устройствами часто требует быстродействующего управления событиями. Например, может возникнуть необходимость, чтобы микроконтроллер периодически вырабатывал ШИМ сигналы или прерывание. В другом приложении микроконтроллер может контролировать входной сигнал для определения состояния внешнего устройства. Процессор событий ЕРА был спроектирован, чтобы уменьшить нагрузку на центральный процессор по управлению этими типами событий. Этот раздел описывает ЕРА, его таймеры и объясняет, как конфигурировать и программировать их.

10.1 Краткий обзор функционирования ЕРА

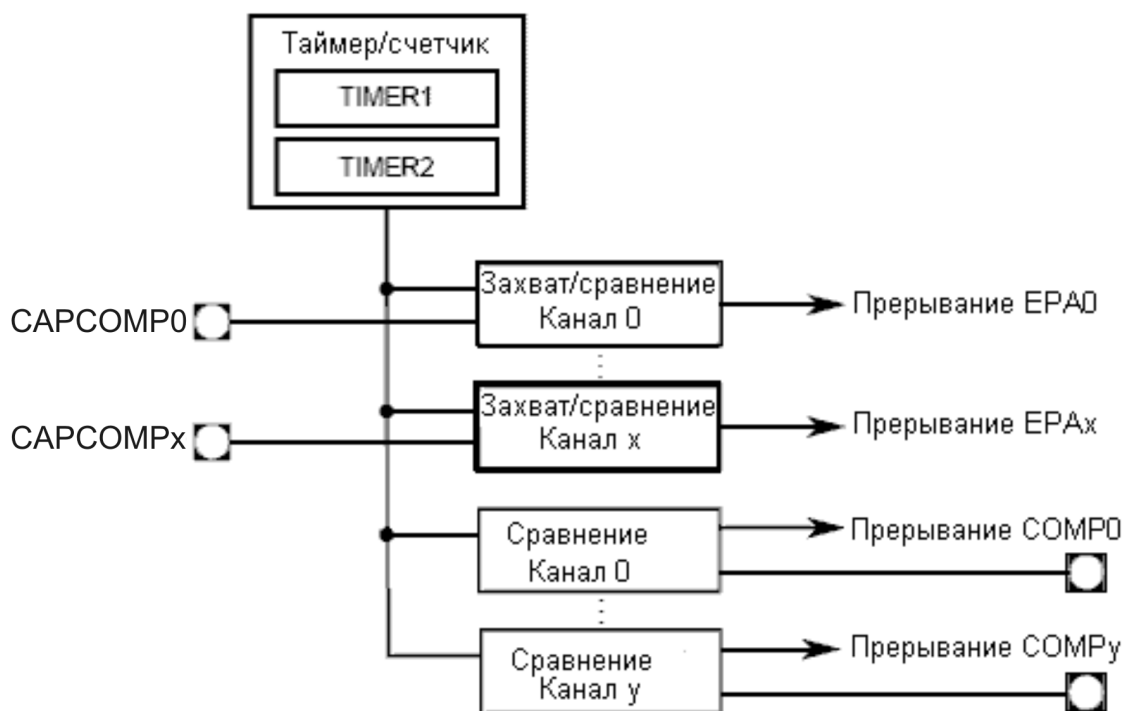
ЕРА осуществляет входные и выходные функции в соответствии с двумя таймерами/счетчиками – таймером 1 и таймером 2 (см. рисунок 10.1). Во входном режиме ЕРА фиксирует на выводе входные события: передний фронт, задний фронт или оба фронта входного импульса. Когда событие происходит, ЕРА делает запись значения таймера/счетчика в регистре и вырабатывает запрос на прерывание, таким образом, событие фиксируется по времени, когда оно произошло. Это называется «захватом по входу». Входные захваты буферизуются, чтобы позволить выполнить два захвата прежде, чем произойдет переполнение.

В выходном режиме ЕРА контролирует таймер/счетчик и сравнивает его значение со значением, сохраненным в специальном регистре. Когда значение таймера/счетчика соответствует сохраненному значению, ЕРА может вызвать событие: сброс таймера, преобразование АЦП, перезагрузку генератора формы сигнала или событие на выходе (установить выход в состояние высокого уровня, установить выход в состояние низкого уровня, изменить значение на выходе или не выполнять никакого действия). Такое выходное событие называют выходным сигналом по совпадению или «выходом сравнения».

Каждый захват по входу или выход сравнения устанавливает бит запроса прерывания. Этот бит может быть причиной прерывания. В таблице 10.1 приведен список каналов захвата/сравнения для микроконтроллера.

Т а б л и ц а 10.1 – Каналы ЕРА

Устройство	Каналы захвата/сравнения	Каналы только сравнения
1874ВЕ66Т	САРСОМР5 – САРСОМР0	СОМР5 – СОМР0



$x = y = 5$

Рисунок 10.1 – Структурная схема блока EPP

10.2 Регистры и сигналы таймера/счетчика EPP

В таблице 10.2 описаны сигналы EPP, входные и выходные сигналы таймера/счетчика. Каждый сигнал мультиплексирован с выводом порта, как показано в первой колонке. В таблице 10.3 кратко описаны регистры каналов захвата/сравнения EPP, каналов только сравнения EPP и таймеров/счетчиков.

Т а б л и ц а 10.2 – Сигналы EPP и таймера/счетчика

Вывод порта	Сигнал EPP	Тип сигнала EPP	Описание
P1.2	T1CLK	Вход	Внешний источник синхроимпульсов для таймера 1
P1.3	T1DIR	Вход	Внешнее управление направлением счета таймера 1
P2.0 P2.1 P2.2 P2.3 P7.0 P7.1	CAPCOMP0 CAPCOMP1 CAPCOMP2 CAPCOMP3 CAPCOMP4 CAPCOMP5	Вход/выход	Быстродействующий ввод-вывод для каналов захвата/сравнения
P2.4 P2.5 P2.6 P2.7 P7.2 P7.3	COMP0 COMP1 COMP2 COMP3 COMP4 COMP5	Выход	Выход каналов только сравнения

Т а б л и ц а 10.3 – Регистры управления и состояния ЕРА

Мнемоника	Адрес	Описание
1	2	3
COMP0_CON COMP1_CON COMP2_CON COMP3_CON COMP4_CON COMP5_CON	1F58 _H 1F5C _H 1F60 _H 1F64 _H 1F68 _H 1F6C _H	Управление сравнения COMP _x . Эти регистры управляют функциями каналов только сравнения
COMP0_TIME COMP1_TIME COMP2_TIME COMP3_TIME COMP4_TIME COMP5_TIME	1F5A _H 1F5E _H 1F62 _H 1F66 _H 1F6A _H 1F6E _H	Время сравнения COMP _x . Эти регистры содержат время, в которое произошло событие на каналах сравнения
ЕРА0_CON ЕРА1_CON ЕРА2_CON ЕРА3_CON ЕРА4_CON ЕРА5_CON	1F40 _H 1F44 _H 1F48 _H 1F4C _H 1F50 _H 1F54 _H	Управление захвата/сравнения CAPCOMP _x . Эти регистры управляют функциями каналов захвата/сравнения. ЕРА1_CON и ЕРА3_CON требуют дополнительного байта, потому что они содержат дополнительный бит для режима генерации импульсов ШИМ (с помощью ЕРА). К этим двум регистрам нужно обращаться как к словам; к другим можно обратиться как к байтам
ЕРА0_TIME ЕРА1_TIME ЕРА2_TIME ЕРА3_TIME ЕРА4_TIME ЕРА5_TIME	1F42 _H 1F46 _H 1F4A _H 1F4E _H 1F52 _H 1F56 _H	Время захвата/сравнения CAPCOMP _x . В режиме захвата эти регистры содержат захваченное значение таймера. В режиме сравнения эти регистры содержат время, в течение которого должно произойти событие. В режиме захвата эти регистры буферизуются, чтобы позволить два захвата прежде, чем происходит переполнение. Однако они не буферизуются в режиме сравнения
INT_MASK	0008 _H	Маска прерываний. Биты в этом 8-битном регистре разрешают и запрещают (маскируют) прерывания, связанные с соответствующими битами в регистре INT_PEND
INT_MASK1	0013 _H	Маска прерываний 1. Биты в этом 8-битном регистре разрешают и запрещают (маскируют) прерывания, связанные с соответствующими битами в регистре INT_PEND1
INT_PEND	0009 _H	Ожидание прерывания. Любой бит, установленный в этом 8-битном регистре, указывает, что ожидается запрос прерывания
INT_PEND1	0012 _H	Ожидание прерывания 1. Любой бит, установленный в этом 8-битном регистре, указывает, что ожидается запрос прерывания
P2_DIR P7_DIR	1FD2 _H 1FD3 _H	Порт_x Направление. Каждый бит P _x _DIR управляет направлением соответствующего вывода. Очистка бита формирует вывод как комплементарный выход; установка бита формирует вывод как выход с открытым стоком или вход. (Выходы с открытым стоком требуют внешней поддержки высокого уровня)

Продолжение таблицы 10.3

1	2	3
P2_MODE P7_MODE	1FD0 _H 1FD1 _H	Порт_x Режим. Каждый бит P _x _MODE определяет, функционирует ли соответствующий вывод как стандартный вывод порта ввода-вывода или как сигнал специальной функции. Установка бита формирует вывод как сигнал специальной функции, очистка бита формирует вывод как стандартный выход порта ввода-вывода
P0_PIN P1_PIN P2_PIN P7_PIN	1FA8 _H 1FA9 _H 1FD6 _H 1FD7 _H	Порт_x Вход. Каждый бит P _x _PIN отражает текущее состояние соответствующего вывода независимо от конфигурации вывода
P2_REG P7_REG	1FD4 _H 1FD5 _H	Порт_x Выход данных. Для входа надо установить соответствующий бит P _x _REG. Для выхода – записать данные, которые будут выведены каждым выводом из соответствующего бита P _x _REG. Когда вывод сконфигурирован как стандартный вход/выход (P _x _MODE.y = 0), результат центрального процессора пишется в P _x _REG и немедленно отображается на выводе. Когда вывод формируется как сигнал специальной функции (P _x _MODE.y = 1), связанный периферийный блок МК или компонент вне МК управляет выводом. Центральный процессор может продолжать запись в P _x _REG, но состояние вывода не изменяется, пока не сконфигурирована его стандартная функция входа/выхода. Эта особенность позволяет программному обеспечению формировать вывод как стандартный вход-выход (очистив P _x _MODE.y), инициализировать или переустановить значение вывода, затем сконфигурировать вывод как сигнал специальной функции (установив P _x _MODE.y). Этим способом инициализация, восстановление ошибки, обработка особых ситуаций и т.д. могут быть выполнены без изменения работы связанного периферийного устройства
PI_MASK	1FBC _H	Маска прерывания периферийного устройства. Биты в этом регистре разрешают и запрещают (маскируют) запрос прерывания по переполнению/антипереполнению таймеров 1 и 2, запросы прерывания генератора формы сигналов
PI_PEND	1FBE _H	Ожидание прерывания от периферийного устройства. Любой установленный бит указывает на ожидание запроса прерывания
T1CONTROL	1F78 _H	Управление таймера 1. Этот регистр разрешает/запрещает таймер 1, управляет направлением счета, выбором источника синхросигнала и направления, устанавливает коэффициент деления частоты синхросигналов
T1RELOAD	1F72 _H	Перезагрузка таймера 1. Этот регистр содержит значение инициализации для таймера 1. При переполнении/антипереполнении таймера 1 загружается значение T1RELOAD в регистр TIMER1, если и квадратурная синхронизация, и перезагрузка разрешены (T1CONTROL.5:0 = 1)

Окончание таблицы 10.3

1	2	3
T2CONTROL	1F7C _H	Управление таймера 2. Этот регистр разрешает/запрещает таймер 2, управляет направлением счета, выбирает источник синхросигнала и направления, устанавливает коэффициент деления частоты синхросигналов
TIMER1	1F7A _H	Значение таймера 1. Этот регистр содержит текущее значение таймера 1
TIMER2	1F7E _H	Значение таймера 2. Этот регистр содержит текущее значение таймера 2

10.3 Краткий обзор функционирования таймеров/счетчиков

ЕРА имеет два 16-битных реверсивных (прямого и обратного счета) таймера/счетчика: таймер 1 и таймер 2, которые могут быть синхронизированы внутренним или внешним синхросигналом. Каждый из них называют таймером, если синхронизация внутренняя, и счетчиком, если синхронизация внешняя. На рисунке 10.2 иллюстрирована структура таймера/счетчика.

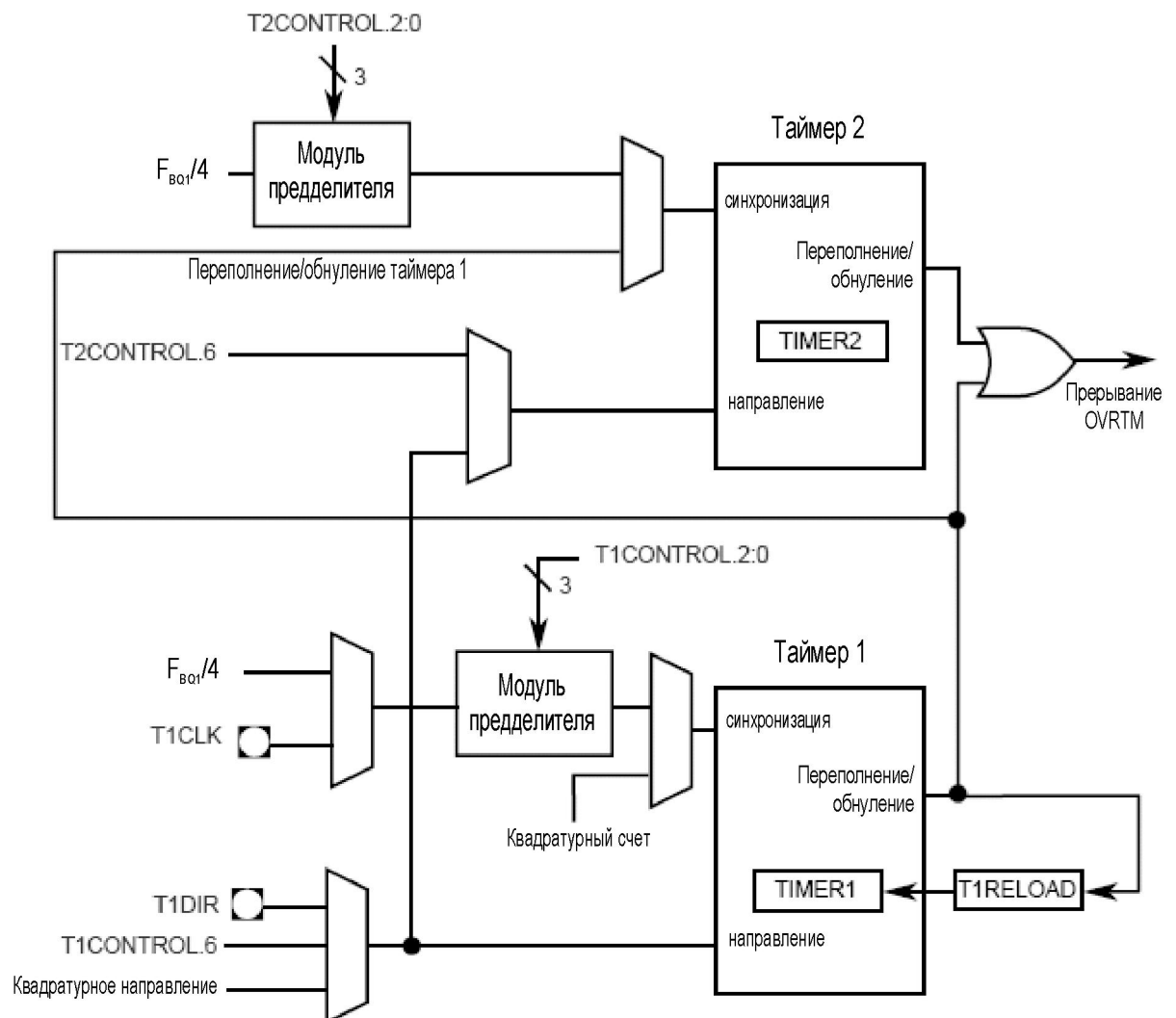


Рисунок 10.2 – Таймеры/счетчики ЕРА

Таймеры/счетчики могут использоваться как времязадающие блоки для входных захватов, выходных сравнений и программных прерываний (таймеры программного обеспечения). Когда счетчик наращивается от $FFFЕ_H$ до $FFFF_H$ или уменьшается от 0001_H до 0000_H , устанавливается бит ожидания запроса прерывания по переполнению/антипереполнению. Этот бит может быть причиной прерывания. Источник синхроимпульсов, источник направления счета и разрешение захвата по входу или выхода сравнения программируются. Максимальная скорость счета – половина частоты внутреннего синхросигнала или $F_{BQ1}/4$. Это обеспечивает минимальное разрешение для входного захвата или выходного сравнения 250 нс (при 16 МГц:

$$\text{разрешение} = \frac{4 \times \text{prescaler_divisor}}{F_{BQ1}}, \quad (10.1)$$

где prescaler_divisor – коэффициент деления частоты из регистров TxCONTROL;
 F_{BQ1} – частота входного синхросигнала на выводе BQ1.

Каскадный режим (только таймер 2)

Таймер 2 может использоваться в каскадном режиме. В этом режиме сигнал переполнения таймера 1 используется как сигнал синхронизации таймера 2. При этом направление счета определяется управляющим битом регистра управления таймера 1 или таймера 2. Этот метод, называемый каскадным, может обеспечить синхроимпульсы низкой частоты для управления таймаутом режима холостого хода или для формирования низкочастотных ШИМ сигналов.

Режим квадратурного счета

Таймер 1 может использоваться в двух режимах синхронизации квадратурного счета. Оба режима используют выходы T1CLK и T1DIR как квадратурные входы, как показано на рисунке 10.3. Внешние квадратурно кодированные сигналы (два сигнала одной частоты, которые отличаются по фазе на 90°) определяют инкремент или декремент таймера при поступлении положительного или отрицательного фронта. Поскольку входы T1CLK и T1DIR тактируются внутренними синхросигналами, переключения должны быть разделены временем, равным не менее двух периодов системного тактового сигнала для правильного выполнения операций. Счет синхронизирован сигналом PH2, который является сигналом PH1, задержанным на половину периода. Последовательность фронтов сигнала и уровней управляет направлением счета, смотри рисунок 10.4 и таблицу 10.4 для информации о направлении счета.

Типичный источник сигналов квадратурного счета – датчик угловых перемещений, показанный на рисунке 10.3. Его выходные сигналы X и Y – входы для T1CLK и T1DIR, которые в свою очередь формируют выходные сигналы X_internal и Y_internal. Эти сигналы приведены на рисунке 10.4 и в таблице 10.4, чтобы описать направление поворота диска.

В заданном по умолчанию режиме квадратурного счета программное обеспечение должно перезагрузить регистр TIMER1, когда таймер 1 переполняется или антипереполняется. В режиме 2 (T1CONTROL.2:0=1) таймер 1 автоматически загружает значение из регистра T1RELOAD в регистр TIMER1, когда происходит переполнение или антипереполнение. Режим 2 полезен для интерфейса с инкрементным датчиком поворота, при вращении только в одном направлении. Для этого случая инициализируется T1RELOAD со значением, которое на единицу меньше, чем разрешение кодирующего устройства. Этот метод позволяет таймеру 1 отслеживать абсолютное положение устройства кодирования поворота без программных издержек.

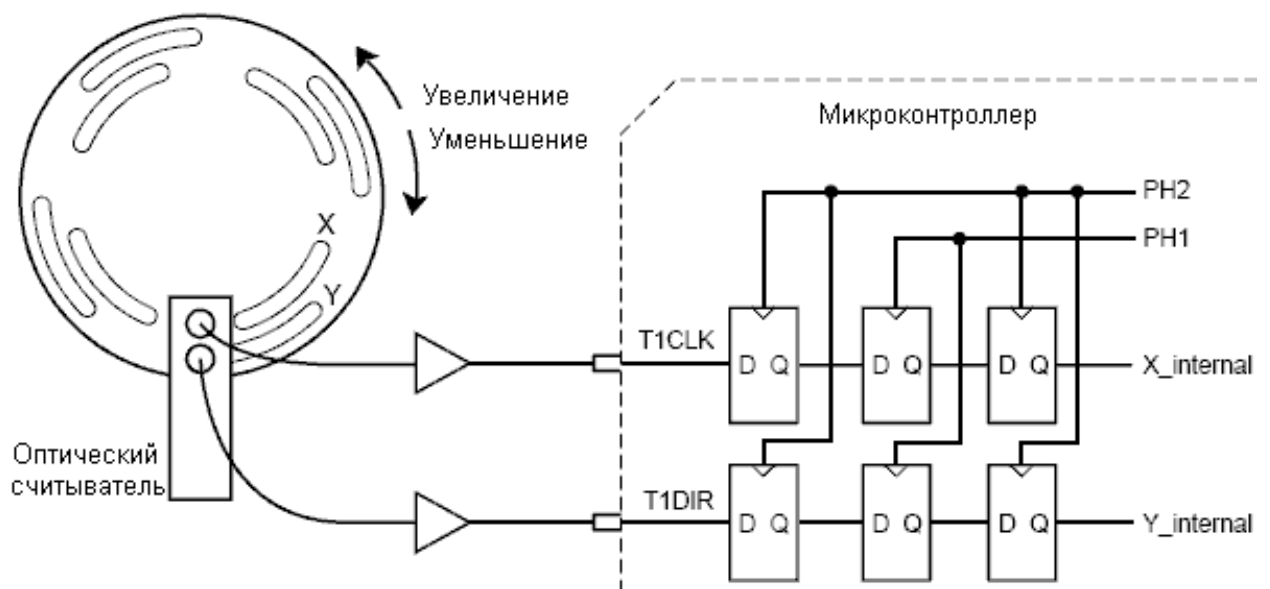


Рисунок 10.3 – Интерфейс режима квадратурного счета

Т а б л и ц а 10.4 – Таблица направления счета в зависимости от T1CLK и T1DIR

Состояние X_internal (T1CLK)	Состояние Y_internal (T1DIR)	Направление счета
↑	0	Увеличение
↓	1	Увеличение
0	↓	Увеличение
1	↑	Увеличение
↓	0	Уменьшение
↑	1	Уменьшение
0	↑	Уменьшение
1	↓	Уменьшение

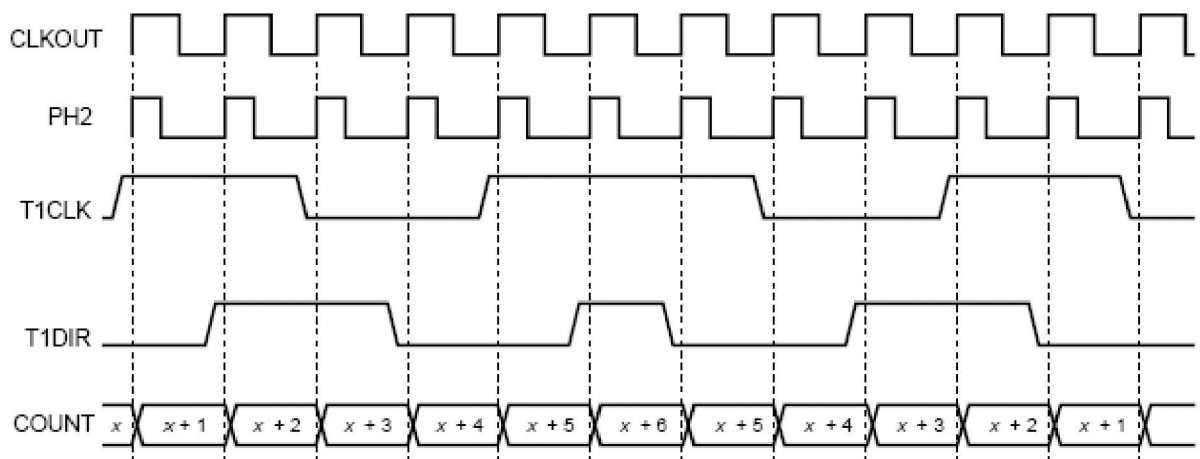


Рисунок 10.4 – Временные диаграммы для квадратурного счета

10.4 Краткое описание функционирования канала ЕРА

Канал ЕРА имеет и программируемые каналы захвата/сравнения, и только сравнения. Каждый канал захвата/сравнения может выполнять следующие задачи (каналы только сравнения имеют те же самые функциональные возможности за исключением того, что они не могут захватить внешнее событие):

- захват значения таймера, когда указанное переключение происходит на выводе ЕРА;
- старт АЦ преобразования или перезагрузка генератора формы сигнала, когда событие захвачено, или значение таймера соответствует запрограммированному значению в регистре времени события;
- очистка, установка или переключение на выводе ЕРА, когда значение таймера соответствует запрограммированному значению в регистре времени события;
- генерация прерывания, когда происходит захват или сравнение;
- сброс собственного базового таймера в режиме сравнения;
- сброс другого таймера и в режиме сравнения, и в режиме захвата.

Каждый канал ЕРА имеет регистр управления – ЕРА_x_CON (каналы захвата/сравнения) или COMP_x_CON (только каналы сравнения), регистр времени события, ЕРА_x_TIME (каналы захвата/сравнения) или COMP_x_TIME (только каналы сравнения) и вход таймера (рисунок 10.5). Регистр управления выбирает таймер, режим и событие, которое будет захвачено или событие, которое должно произойти. Регистр времени события удерживает захваченное значение таймера в режиме захвата и время события в режиме сравнения.

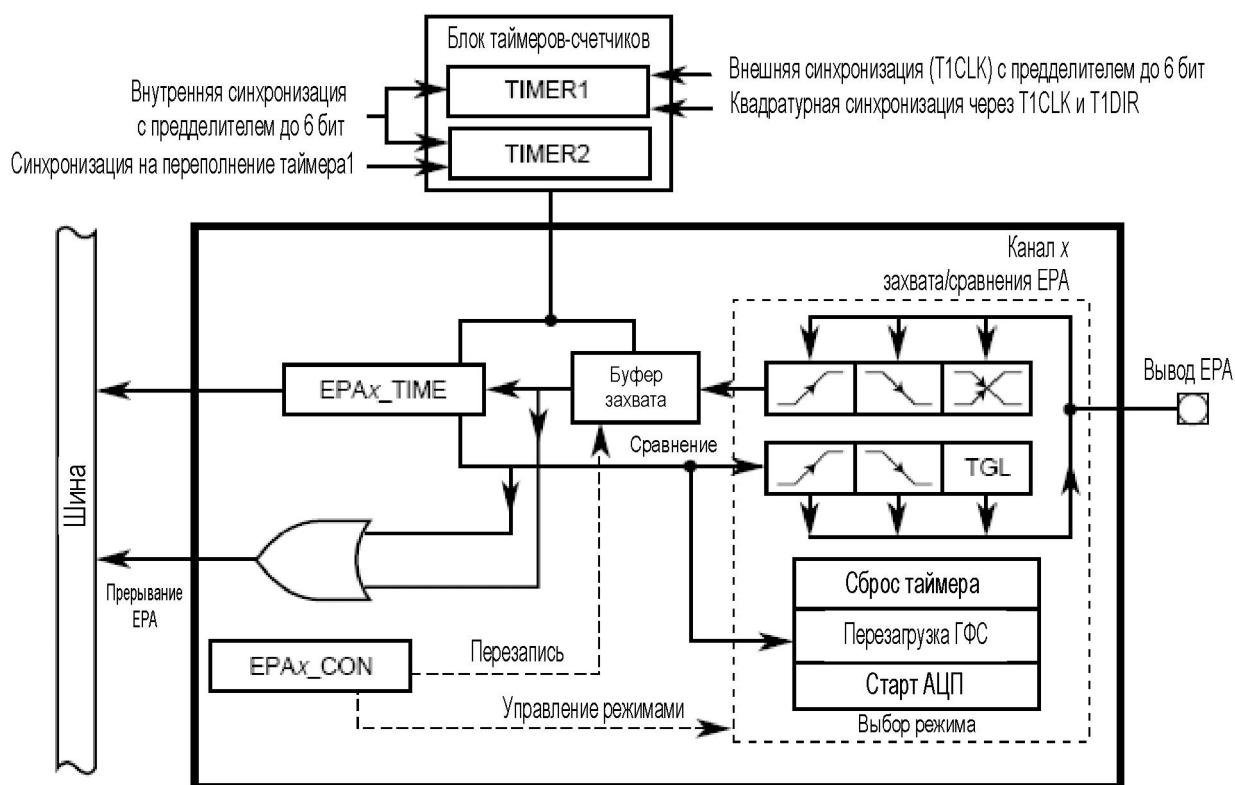


Рисунок 10.5 – Канал захвата/сравнения ЕРА

Работа в режиме захвата

В режиме захвата, когда действительное событие на выводе имеет место, значение выбранного таймера захватывается в буфер. Значение таймера затем передается из буфера в регистр ЕРА_x_TIME, который устанавливает бит ожидания прерывания ЕРА, как показано на рисунке 10.6. При разрешении происходит прерывание. Если второе событие

происходит прежде, чем центральный процессор читает первое значение таймера в EРАх_TIME, текущее значение таймера загружается в буфер и удерживается там. После того, как центральный процессор читает регистр EРАх_TIME, содержимое буфера захвата автоматически перемещается в EРАх_TIME, и бит ожидания прерывания EРА устанавливается.

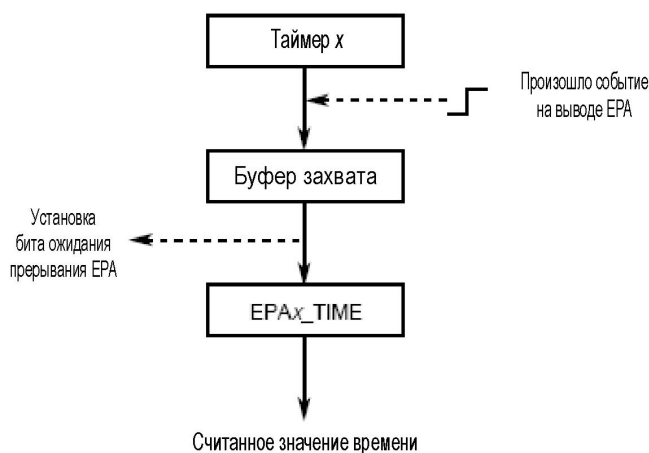


Рисунок 10.6 – Упрощенная структура EРА при захвате

Если третье событие происходит прежде, чем центральный процессор считывает регистр времени события, бит перезаписи (EРАх_CON.0) определяет, как EРА будет обрабатывать событие. Если бит очищен, EРА игнорирует третье событие. Если бит установлен, время третьего события перезаписывает в буфере захвата время второго события. В таблице 10.5 представлены возможные варианты обработки действительного события.

Примечание – Для того, чтобы событие было захвачено, сигнал должен быть установлен в течение, по крайней мере, двух машинных циклов до и после переключения (смотри рисунок 10.7).



Рисунок 10.7 – Действующие события на выводе EРА

Т а б л и ц а 10.5 – Операции канала EРА, когда действующий фронт сигнала появляется на выводе

Бит перезаписи (EРАх_CON.0)	Состояние буфера захвата и EРАх_TIME	Операция EРА, когда на выводе действующий фронт сигнала
0	Пустой	Фронт захвачен, и время события загружено в буфер захвата и регистр EРАх_TIME
0	Полный	Новые данные игнорируются, нет захвата, прерывания EРА или передачи времени события
1	Пустой	Фронт захвачен, и время события загружено в буфер захвата и регистр EРАх_TIME
1	Полный	Старые данные переписаны в буфер захвата

Захват входного события не устанавливает бит ожидаемого прерывания, пока захваченное значение времени не перемещается фактически из буфера захвата в регистр EРАх_TIME. Если буфер содержит данные, и используется PTS для обслуживания прерывания, то два прерывания PTS происходят почти одно за другим (то есть с одной инструкцией, выполненной между прерываниями).

Переполнение ЕРА

Переполнение имеет место, когда частота переключения на входах ЕРА превышает скорость обработки процедурой обслуживания прерывания ЕРА. Если не осуществлять стратегию обработки переполнений и если одновременно выполняются указанные ниже три условия, может возникнуть ситуация, когда и буфер захвата, и регистр EРАх_TIME содержат данные, а ЕРА прерывание не происходит:

- входной сигнал с достаточно высокой частотой, чтобы вызвать переполнение, присутствует на выбранном выводе ЕРА;
- бит перезаписи установлен (EРАх_CON.0 = 1, старые данные переписаны при переполнении);
- регистр EРАх_TIME читается в тот момент, когда ЕРА признает захваченный фронт как действующий.

Входная частота события зависит от длительности процедуры обслуживания прерывания так же, как других факторов. Если процедура обслуживания прерывания не включает проверку на переполнение, эта ситуация сохраняется до тех пор, пока МК не будет сброшен или регистр EРАх_TIME сосчитан. Выполнение чтения EРАх_TIME позволяет буферизованному значению времени быть перемещенным в EРАх_TIME. Это очищает буфер и позволяет другому событию быть захваченным. Необходимо помнить, что процедура передачи содержания буфера регистру EРАх_TIME состоит в том, что фактически установлен бит ожидания прерывания EРАх и начинается прерывание.

Предотвращение переполнения ЕРА

Любой из следующих методов может использоваться, чтобы предотвратить или исправить ситуацию переполнения ЕРА:

- очистить EРАх_CON.0. Когда бит перезаписи (EРАх_CON.0) – ноль, ЕРА не оценивает захваченный фронт, пока регистр EРАх_TIME не считан и данные не переданы из буфера захвата в EРАх_TIME. Это предотвращает переполнение, игнорируя новые входные события для захвата, когда буфер захвата и EРАх_TIME содержат действительные времена захвата;
- сделать проверку ожидаемых запросов прерываний до выполнения процедуры обслуживания прерываний EРАх. Другой способ избежать переполнения ЕРА состоит в проверке ожидаемых запросов прерываний перед выходом из процедуры обслуживания прерываний EРАх. Это – легкий способ обнаружения переполнения и дополнительных прерываний. Это может также сохранить время цикла, устраняя время ожидания, необходимое для обслуживания ожидаемых прерываний. Однако этот метод не может использоваться с периферийным сервером PTS.

Работа в режиме сравнения

Когда отобранное значение таймера соответствует значению времени события, происходит действие, указанное в регистре управления (то есть установка, очистка или переключение выхода, начало АЦП преобразования или перезагрузка генератора формы сигнала). Если бит повторного разрешения (EРАх_CON.3) установлен, действие происходит повторно при каждом совпадении значения таймера. Если очищен бит повторного разрешения, действие не происходит повторно, пока новое значение не будет записано в регистр времени события. В режиме сравнения можно использовать ЕРА, чтобы формировать широтно-импульсно модулированный ШИМ сигнал на выводе ЕРА.

Формирование низкочастотного сигнала ШИМ

Можно сформировать низкочастотный ШИМ сигнал на выводе одного канала ЕРА со стандартной процедурой обслуживания прерывания. Формируется канал ЕРА следующим образом: режим сравнения, переключение на выводе и функция повторного сравнения разрешены. Выбирается процедура обслуживания стандартных прерываний, разрешается прерывание ЕРА и глобальное разрешение прерываний инструкцией EI. Когда значение таймера/счетчика соответствует значению в регистре времени события, ЕРА переключает сигнал на выводе и вызывает прерывание. Процедура обслуживания прерывания загружает новое значение в ЕРАx_TIME.

Максимальная выходная частота зависит от полной задержки прерываний и от времени выполнения обслуживания прерывания процедурой, используемой системой. Поскольку дополнительные каналы ЕРА и другие функции микроконтроллера используются, получается уменьшение максимальной частоты ШИМ, потому что общее время ожидания прерываний и обслуживания прерываний увеличивается. Для определения максимума низкочастотного ШИМ сигнала в системе вычисляют худший случай времени задержки прерываний и наихудшее время выполнения обслуживания прерывания и затем складывают их вместе. Наихудший случай времени ожидания прерываний – полное время ожидания всех прерываний (обычных и PTS), используемых в системе. Наихудший случай времени выполнения обслуживания прерываний – полное время выполнения всех процедур обслуживания прерываний и процедур PTS.

Предположим, что система имеет один канал ЕРА, одно разрешенное прерывание и следующую процедуру обслуживания прерывания:

```
    ; если ЕРА0-х прерывание произошло
    ЕРА0-х_ISR:
    PUSHA
    LD ЕРАx_CON, #toggle_command
    ADD ЕРАx_TIME, TIMERx, [next_duty_ptr]; загрузка времени следующего
;события
    POPA
    RET
```

Худший случай времени ожидания прерывания для системы с единственным прерыванием – 56 периодов системного тактового сигнала (тактов) при использовании внешнего стека и 54 такта при использовании внутреннего стека. Для определения времени обслуживания прерываний складывают время выполнения инструкций сервисной программы.

Полное время выполнения процедуры обслуживания прерывания, которая обслуживает прерывание ЕРА – 79 тактов при использовании внешнего стека или 71 такт при использовании внутреннего стека. Поэтому один канал захвата/сравнения может быть обновлен каждые 125 тактов при использовании внутреннего стека (54 + 71). Каждый ШИМ период требует два обновления (одна установка и одна очистка), так что период ШИМ равняется 250 тактам. Когда входная частота на BQ1 – 16 МГц, ШИМ период – 31,25 мкс и максимальная частота ШИМ равна 32 кГц.

Формирование высокоскоростного ШИМ сигнала

Можно сформировать высокоскоростной импульсный сигнал с модулированной шириной импульса из двух каналов ЕРА и выделенного таймера/счетчика. Первый канал переключает сигнал на выходе, когда значение таймера соответствует ЕРАx_TIME, и через некоторое время второй канал переключает сигнал на выходе снова, а затем сбрасывает таймер/счетчик. Это перезапускает цикл. Прерывания не запрашиваются, в результате возможна максимальная скорость. Программное обеспечение вычисляет соответствующие значения ЕРАx_TIME и загружает их в соответствующее время в цикле в соответствии с изменением частоты или скважности.

Этот метод настройки ЕРА (устанавливаемый регистрами ТхCONTROL) позволяет получить максимум частоты ШИМ сигнала на выводе. Разрешение – минимальное время, требуемое между последовательными захватами или сравнениями. Когда входная частота на ВQ1 – 16 МГц, разрешение – 250 нс при максимальной частоте сигнала ШИМ 4 МГц.

10.5 Программирование ЕРА и таймеров/счетчиков

В этом подразделе приводится конфигурирование выводов порта для ЕРА и таймеров/счетчиков, описывается, как программировать таймеры, каналы захвата/сравнения и каналы только сравнения и объясняется, как разрешить прерывания ЕРА.

Конфигурирование ЕРА и сигналов таймера/счетчика

Прежде чем использовать ЕРА, необходимо сформировать соответствующие сигналы порта как сигналы специальных функций для ЕРА, так и дополнительно источник синхронизации и сигналы управления направлением счета для таймера/счетчика.

Сигналы, которые не используются для канала ЕРА или таймера/счетчика, могут формироваться как стандартный вход/выход.

Программирование таймеров

Регистры управления таймерами – T1CONTROL (рисунок 10.8) и T2CONTROL (рисунок 10.9). Запись значений в эти регистры конфигурирует таймеры. Запись в регистры TIMER1 и TIMER2 (таблица 10.3) задает определенные значения в регистрах таймеров.

Регистр T1CONTROL

Адрес: 1F78_H

Состояние после сброса: 00_H

Регистр управления таймера 1 (T1CONTROL) определяет источник синхроимпульсов, направление счета и период счета для таймера 1.

7							0
CE	UD	M2	M1	M0	P2	P1	P0

Рисунок 10.8 – Регистр управления таймера 1 (T1CONTROL)

Т а б л и ц а 10.6 – Разряды регистра управления таймера 1 (T1CONTROL)

Номер разряда	Мнемоника	Функция
1	2	3
7	CE	Разрешение счета. Этот бит разрешает или запрещает таймеры. После сброса таймеры запрещены и не разрешен их запуск. 0 запрещает таймеры 1 разрешает таймеры
6	UD	Прямой/обратный счет. Этот бит определяет направление счета таймера в выбранных режимах (см. биты режима M2 – M0). 0 счет в обратном направлении 1 счет в прямом направлении

Окончание таблицы 10.6

1	2	3
5 – 3	M2 – M0	<p>Биты управления синхронизацией и направлением счета ЕРА. Эти биты определяют источник синхроимпульсов и источник сигнала, управляющего направлением счета.</p> <p>M2 M1 M0 Источник синхроимпульсов Источник направления</p> <p>0 0 0 F_{BQ1}/4 UD бит (T1CONTROL.6)</p> <p>X 0 1 T1CLK вход¹⁾ UD бит (T1CONTROL.6)</p> <p>0 1 0 F_{BQ1}/4 T1DIR вход</p> <p>0 1 1 T1CLK вход¹⁾ T1DIR вход</p> <p>1 1 1 квадратурная синхронизация, использующая T1CLK и T1DIR</p>
2 – 0	P2 – P0	<p>Биты коэффициента деления частоты синхроимпульсов ЕРА (Prescaler)</p> <p>Эти биты определяют prescaler_divisor для синхроимпульсов.</p> <p>P2 P1 P0 Деление Разрешение²⁾</p> <p>0 0 0 на 1 (запрещение) 250 нс</p> <p>0 0 1 на 2 500 нс</p> <p>0 1 0 на 4 1 мкс</p> <p>0 1 1 на 8 2 мкс</p> <p>1 0 0 на 16 4 мкс</p> <p>1 0 1 на 32 8 мкс</p> <p>1 1 0 на 64 16 мкс</p> <p>1 1 1 разрешение T1RELOAD –</p>
<p>¹⁾ Если выбрана внешняя синхронизация, таймер считает и на положительный, и на отрицательный фронт синхроимпульсов.</p> <p>²⁾ Для 16 МГц. Для других частот используют формулу</p> $\text{разрешение} = \frac{4 \times \text{prescaler_divisor}}{F_{BQ1}}$		

Регистр T2CONTROL

Адрес: 1F7C_H

Состояние после сброса: 00_H

Регистр управления таймера 2 (T2CONTROL) определяет источник синхроимпульсов, направление и диапазон счёта для таймера 2.

7								0
CE	UD	M2	M1	M0	P2	P1	P0	

Рисунок 10.9 – Регистр управления таймера 2 (T2CONTROL)

Т а б л и ц а 10.7 – Разряды регистра управления таймера 2 (T2CONTROL)

Номер бита	Мнемоника	Функция
7	CE	Разрешение счета. Этот бит разрешает или запрещает таймеры. После сброса таймеры запрещены и не разрешен запуск. 0 запрещает таймер 1 разрешает таймер
6	UD	Прямой/обратный счет. Этот бит определяет направление счета таймера в выбранных режимах (биты режима M2:0). 0 счет в обратном направлении 1 счет в прямом направлении
5 – 3	M2 – M0	Биты управления синхронизацией и направлением счета ЕРА. Эти биты определяют источник синхроимпульсов и источник направления счета. M2 M1 M0 Источник синхроимпульсов Источник направления 0 0 0 F _{BQ1} /4 UD бит (T2CONTROL.6) X 0 1 зарезервирован – 0 1 0 зарезервирован – 0 1 1 зарезервирован – 1 0 0 переполнение таймера 1 UD бит (T2CONTROL.6) 1 1 0 переполнение таймера 1 как у таймера 1 1 1 1 зарезервирован –
2 – 0	P2 – P0	Биты коэффициента деления частоты синхроимпульсов ЕРА (Prescaler_divisor). Эти биты определяют prescaler_divisor для синхроимпульсов. P2 P1 P0 Деление Разрешение ¹⁾ 0 0 0 на 1 (запрещение) 250 нс 0 0 1 на 2 500 нс 0 1 0 на 4 1 мкс 0 1 1 на 8 2 мкс 1 0 0 на 16 4 мкс 1 0 1 на 32 8 мкс 1 1 0 на 64 16 мкс 1 1 1 зарезервировано –
<p>¹⁾ Для 16 МГц. Для других частот используют формулу</p> $\text{разрешение} = \frac{4 \times \text{prescaler_divisor}}{F_{BQ1}}$		

Программирование каналов захвата/сравнения

Регистр ЕРАх_CON управляет функционированием соответствующего канала захвата/сравнения. Регистры идентичны за исключением бита 2. Для каналов ЕРА САРСОМР0, САРСОМР2, САРСОМР4 установка этого бита позволяет ЕРА по событию вызывать перезагрузку генератора формы сигнала. Для каналов ЕРА САРСОМР1, САРСОМР3, САРСОМР5 установка этого бита позволяет событию ЕРА вызывать АЦ преобразование. Чтобы запрограммировать событие сравнения, всегда записывают ЕРАх_CON (рисунок 10.10) первым для конфигурации ЕРА канала захвата/сравнения, и затем загружают время события в ЕРАх_TIME. Чтобы запрограммировать захват события,

достаточно записать EРAx_CON. В таблице 10.8 показывается действие от различных комбинаций значений битов в EРAx_CON для каналов 1, 3.

Т а б л и ц а 10.8 – Пример установок битов в управляющем регистре EРA

Режим захвата								Действие
TВ	CE	MODE		PE	WGR/ AD	ROT	ON/ RT	
7	6	5	4	3	2	1	0	
X	0	0	0	–	–	–	0	Нет
X	0	0	1	–	X	X	X	Захват на отрицательном фронте
X	0	1	0	–	X	X	X	Захват на положительном фронте
X	0	1	1	–	X	X	X	Захват на обоих фронтах
X	0	X	1	–	X	1	X	Захват на отрицательном фронте и сброс противоположного таймера
X	0	1	X	–	X	1	X	Захват на положительном фронте и сброс противоположного таймера
X	0	0	1	–	1	X	X	Старт АЦ преобразования (САРСОМР1, САРСОМР3, САРСОМР5) или перезагрузка генератора формы сигнала (САРСОМР0, САРСОМР2, САРСОМР4) на отрицательном фронте
X	0	1	0	–	1	X	X	Старт АЦ преобразования (САРСОМР1, САРСОМР3, САРСОМР5) или перезагрузка генератора формы сигнала (САРСОМР0, САРСОМР2, САРСОМР4) на положительном фронте
Режим сравнения								Действие
TВ	CE	MODE		PE	WGR/ AD	ROT	ON/ RT	
7	6	5	4	3	2	1	0	
X	1	0	0	X	–	–	0	Нет
X	1	0	0	X	0	X	0	Выработка только прерывания (таймер программного обеспечения)
X	1	0	1	X	X	X	X	Обнуление на выходе
X	1	1	0	X	X	X	X	Установка (высокий уровень) на выходе
X	1	1	1	X	X	X	X	Переключение на выходе
X	1	X	X	X	X	0	1	Сброс текущего таймера
X	1	X	X	X	X	1	1	Сброс противоположного таймера
X	1	X	X	X	1	X	X	Старт АЦ преобразования (САРСОМР1, САРСОМР3, САРСОМР5) или перезагрузка генератора формы сигнала (САРСОМР0, САРСОМР2, САРСОМР4)
Примечания								
1 «–» – неиспользуемый бит.								
2 «X» – бит может использоваться, но нет никакого действия.								

Регистр EРAx_CON (x = 0–5)

Адреса: 1F40_H, 1F44_H, 1F48_H, 1F4C_H, 1F50_H, 1F54_H

Состояние сброса: 00_H

Регистры управление EРA (EРAx_CON) управляют функциями соответствующих каналов захвата/сравнения.

x = 0, 2, 4

7							0
TB	CE	M1	M0	RE	WGR	ROT	ON/RT

x = 1, 3, 5

7							0
TB	CE	M1	M0	RE	AD	ROT	ON/RT

Рисунок 10.10 – Регистр управления ЕРА (ЕРА_x_CON)

Т а б л и ц а 10.9 – Разряды регистра управления ЕРА (ЕРА_x_CON)

Номер разряда	Название	Функция																														
1	2	3																														
7	TB	<p>Выбор временной базы. Определяет текущий таймер. 0 таймер 1 – текущий таймер, а таймер 2 – противоположный таймер 1 таймер 2 – текущий таймер, а таймер 1 – противоположный таймер Событие сравнения (старт АЦ преобразования, очистка, установка или переключение выхода, и/или перезагрузка любого таймера) происходит, когда значение в текущем таймере соответствует времени, запрограммированному в регистре времени события. Когда происходит захват (по заднему, по переднему фронту или по любому изменению фронта на выводе блока ЕРА), соответствующее значение таймера сохраняется в регистре ЕРА_x_TIME</p>																														
6	CE	<p>Разрешение сравнения. Разрешает работать каналу ЕРА в режиме захвата или сравнения. 0 режим захвата 1 режим сравнения</p>																														
5, 4	M1, M0	<p>Выбор режима ЕРА. В режиме захвата определяет тип события, которое будет захватываться на входе. В режиме сравнения определяет действие, выполняемое блоком ЕРА в момент сравнения значения таймера и времени события.</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 10%;">M1</td> <td style="width: 10%;">M0</td> <td>Действие в режиме захвата</td> </tr> <tr> <td>0</td> <td>0</td> <td>нет изменений</td> </tr> <tr> <td>0</td> <td>1</td> <td>захват по заднему (отрицательному) фронту</td> </tr> <tr> <td>1</td> <td>0</td> <td>захват по переднему (положительному) фронту</td> </tr> <tr> <td>1</td> <td>1</td> <td>захват по любому фронту</td> </tr> <tr> <td>M1</td> <td>M0</td> <td>Действие в режиме сравнения</td> </tr> <tr> <td>0</td> <td>0</td> <td>нет изменений</td> </tr> <tr> <td>0</td> <td>1</td> <td>переключение в низкий уровень</td> </tr> <tr> <td>1</td> <td>0</td> <td>переключение в высокий уровень</td> </tr> <tr> <td>1</td> <td>1</td> <td>переключение уровня</td> </tr> </table>	M1	M0	Действие в режиме захвата	0	0	нет изменений	0	1	захват по заднему (отрицательному) фронту	1	0	захват по переднему (положительному) фронту	1	1	захват по любому фронту	M1	M0	Действие в режиме сравнения	0	0	нет изменений	0	1	переключение в низкий уровень	1	0	переключение в высокий уровень	1	1	переключение уровня
M1	M0	Действие в режиме захвата																														
0	0	нет изменений																														
0	1	захват по заднему (отрицательному) фронту																														
1	0	захват по переднему (положительному) фронту																														
1	1	захват по любому фронту																														
M1	M0	Действие в режиме сравнения																														
0	0	нет изменений																														
0	1	переключение в низкий уровень																														
1	0	переключение в высокий уровень																														
1	1	переключение уровня																														

Окончание таблицы 10.9

1	2	3
3	RE	<p>Повторное разрешение. Повторное разрешение применимо только к режиму сравнения. Это позволяет продолжать выполнять сравнение события каждый раз, когда регистр (EPAx_TIME) соответствует текущему таймеру, а не только до первого сравнения. 0 сравнение, функция запрещена после первого события 1 сравнение, функция всегда разрешена</p>
2	WGR AD	<p>Перезагрузка генератора формы сигнала, АЦ преобразование. Функция этого бита зависит от канала EPA. Для канала захвата/сравнения CAPCOMP0, CAPCOMP2, CAPCOMP4: бит WGR позволяет использовать действия EPA, чтобы вызвать перезагрузку новых значений в генератор формы сигнала. 0 нет действия 1 разрешена перезагрузка генератора формы сигнала Для каналов захвата/сравнения CAPCOMP1, CAPCOMP3, CAPCOMP5: бит AD разрешения АЦ преобразования позволяет использовать действия EPA, чтобы начать АЦ преобразование, которое было предварительно подготовлено в регистрах управления АЦП. 0 не начинать АЦ преобразование 1 старт АЦ преобразования при сравнении</p>
1	ROT	<p>Сброс противоположного таймера. Управление разными функциями для режимов захвата и сравнения. Выбирает таймер, который должен быть сброшен, если RT бит установлен. В режиме захвата: 0 нет действия 1 сбрасывает противоположный таймер В режиме сравнения: 0 выбирает текущий таймер для возможного сброса 1 выбирает противоположный таймер для возможного сброса. Состояние бита ТВ определяет, какой таймер является текущим таймером и какой таймер является противоположным таймером</p>
0	ON/RT	<p>Перезапись/сброс таймера. Этот бит перезаписывает новое значение в режиме захвата и сбрасывает таймер в режиме сравнения. В режиме захвата (ON): ошибка переполнения происходит во время входного захвата, в то время как регистр (EPAx_TIME) и его буфер переполнены. Когда происходит переполнение, бит ON определяет: будут ли перезаписаны старые данные или новые данные игнорируются. 0 новые данные игнорируются 1 старые данные в буфере перезаписываются В режиме сравнения (RT): 0 запрещает функцию сброса 1 сброс таймера, выбираемого битом ROT</p>

Программирование каналов сравнения

Чтобы запрограммировать события сравнения, необходимо сначала записать в регистр COMPx_CON (рисунок 10.11), чтобы формировать канал только сравнения и затем загрузить время события в COMPx_TIME. COMPx_CON имеет те же самые биты и назначения как ERAx_CON. COMPx_TIME функционально идентичен ERAx_TIME.

Регистр COMPx_CON (x = 0–5)

Адрес: 1F58_H, 1F5C_H, 1F60_H, 1F64_H, 1F68_H, 1F6C_H

Состояние после сброса: 00_H

Регистры управления ERA сравнением (COMPx_CON) определяют функции сравнения каналов ERA.

x = 0, 2, 4

7	0
TB	CE M1 M0 RE WGR ROT RT

x = 1, 3, 5

7	0
TB	CE M1 M0 RE AD ROT RT

Рисунок 10.11 – Регистр управления сравнением ERA (COMPx_CON)

Т а б л и ц а 10.10 – Разряды регистра управления сравнением ERA (COMPx_CON)

Номер разряда	Название	Функция
1	2	3
7	TB	Выбор временной базы. Определяет текущий таймер. 0 таймер 1 – текущий таймер, а таймер 2 – противоположный таймер 1 таймер 2 – текущий таймер, а таймер 1 – противоположный таймер Событие сравнения (старт АЦ преобразования, очистка, установка или переключение выхода, и/или перезагрузка любого таймера) происходит, когда значение в текущем таймере соответствует времени, запрограммированному в регистре времени события
6	CE	Разрешение сравнения. Разрешает работать каналу ERA в режиме сравнения. 0 режим сравнения запрещен 1 режим сравнения разрешен
5, 4	M1, M0	Выбор режима ERA. Определяет тип сравниваемого события. M1 M0 Действие в режиме сравнения 0 0 нет изменений 0 1 переключение в низкий уровень 1 0 переключение в высокий уровень 1 1 переключение уровня
3	RE	Повторное разрешение. Повторное разрешение применимо только к режиму сравнения. Это позволяет продолжать выполнять сравнение события каждый раз, когда регистр (ERAx_TIME) соответствует текущему таймеру, а не только до первого сравнения. 0 сравнение, функция запрещена после первого события 1 сравнение, функция всегда разрешена

Окончание таблицы 10.10

1	2	3
2	WGR AD	<p>Перезагрузка генератора формы сигнала, АЦ преобразование. Функция этого бита зависит от канала ЕРА.</p> <p>Для каналов захвата/сравнения CAPCOMP0, CAPCOMP2, CAPCOMP4: бит WGR позволяет использовать действия ЕРА, чтобы вызвать перезагрузку новых значений в генератор формы сигнала. 0 нет действия 1 разрешена перезагрузка генератора формы сигнала</p> <p>Для каналов захвата/сравнения CAPCOMP1, CAPCOMP3, CAPCOMP5: AD бит разрешения АЦ преобразования позволяет использовать действия ЕРА, чтобы начать АЦ преобразование, которое было предварительно подготовлено в регистрах управления АЦП. 0 не начинать АЦ преобразование 1 старт АЦ преобразования при сравнении</p>
1	ROT	<p>Сброс противоположного таймера. Выбирает таймер, который должен быть сброшен, если RT бит установлен. 0 выбирает текущий таймер для возможного сброса 1 выбирает противоположный таймер для возможного сброса. Состояние ТВ бита определяет, какой таймер является текущим таймером и какой таймер является противоположным таймером</p>
0	RT	<p>Сброс таймера. Этот бит управляет, будет ли таймер, выбранный битом ROT, сброшен. 1 сбрасывает таймер, выбранный битом ROT 0 запрещает функцию сброса</p>

10.6 Разрешения прерывания ЕРА

Для разрешения прерывания необходимо установить соответствующие биты в регистре INT_MASK. Для разрешения индивидуальных источников периферийных прерываний устанавливать соответствующие биты в регистре PI_MASK. В разделе 6 подробнее описываются эти прерывания.

10.7 Определение состояния события

В режиме сравнения бит ожидания прерывания устанавливается каждый раз, когда происходит совпадение при разрешенном событии (даже если прерывание специально замаскировано в регистре маски). В режиме захвата бит запроса прерывания устанавливается каждый раз, когда запрограммированное событие захвачено, и время события перемещается из буфера захвата в регистр ERAx_TIME.

Биты ожидания расположены в регистрах INT_PEND и INT_PEND1. Биты запроса для мультиплексированных прерываний (те, которые входят в периферийные прерывания) расположены в регистре PI_PEND. Переполнение/обнуление таймеров также вызывает установку битов запроса прерывания. Даже если прерывание замаскировано, программное обеспечение может опрашивать регистры запросов прерываний, чтобы определить, было ли событие.

11 Генератор программируемого тактового сигнала

ИС 1874BE66T имеет в составе своей периферии специфическое устройство – генератор сигнала с программируемой частотой. Этот сигнал имеет скважность 50 % и программируемую частоту в пределах от 4 кГц до 1 МГц при частоте тактового сигнала 16 МГц. Одним из применений для этого генератора является дистанционное управление передачей данных и управляющих сигналов для приводов на инфракрасных диодах.

11.1 Функциональное описание

Генератор программируемого тактового сигнала (рисунок 11.1) имеет в своем составе регистр частоты (FREQ_GEN), счетный регистр (FREQ_CNT), выходной формирователь сигнала. Выходной формирователь сигнала является одновременно выходным формирователем портового вывода P7.7, потому перед использованием генератора вывод конфигурируется как выход.

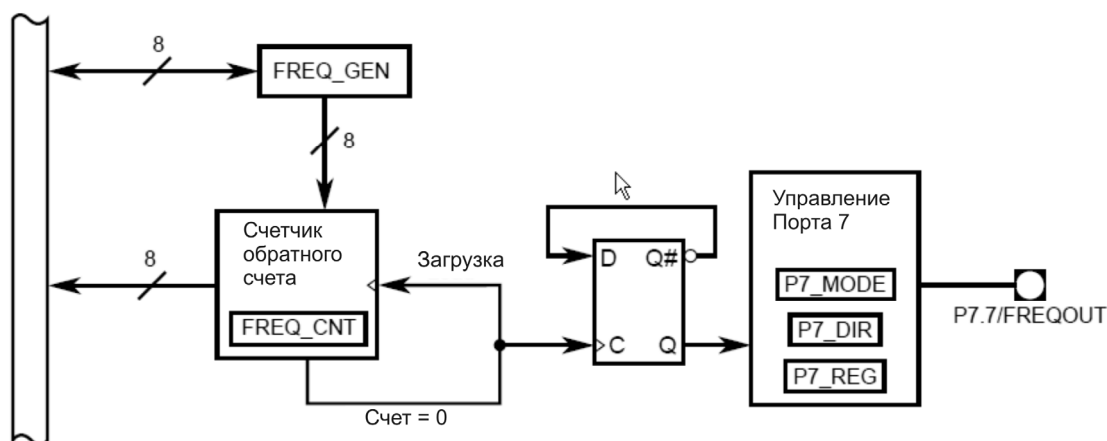


Рисунок 11.1 – Структурная схема генератора программируемого тактового сигнала

Регистр частоты (FREQ_GEN) управляет выходной частотой. Генератор загружает содержимое регистра FREQ_GEN в счетчик. Счетчик осуществляет обратный счет до нуля, и затем загружает свое новое значение из регистра FREQ_GEN. Каждая загрузка включает D-триггер, создающий сигнал со скважностью 50 %. Регистр счета (FREQ_CNT) отражает текущее значение счетчика обратного счета. В таблице 11.1 и в таблице 11.2 описаны выходной сигнал и регистры управления и состояния генератора.

Т а б л и ц а 11.1 – Сигнал генератора программируемого тактового сигнала

Выход порта	Сигнал генератора	Тип сигнала	Описание
P7.7	FREQOUT	Выход	Выходной сигнал генератора программируемого тактового сигнала

Т а б л и ц а 11.2 – Регистры управления и состояния генератора программируемого тактового сигнала

Наименование	Адрес	Описание
1	2	3
FREQ_GEN	1FB8 _H	Регистр частоты. Регистр удерживает запрограммированное значение и определяет выходную частоту. Это значение перегружается в счетчик обратного счета каждый раз при достижении нулевого значения в этом счетчике
FREQ_CNT	1FBA _H	Регистр счета. Регистр отображает текущее значение счетчика. Регистр только для чтения

Окончание таблицы 11.2

1	2	3
P7_DIR	1FBA _H	Направление порта 7. Разряд 7 управляет направлением P7.7/FREQOUT. Обнуление этого разряда конфигурирует P7.7/FREQOUT как комплементарный выход
P7_MODE	1FD1 _H	Режим порта 7. Разряд 7 управляет режимом (портовый вход-выход или сигнал специальной функции) вывода P7.7/FREQOUT. Установка этого бита конфигурирует вывод как FREQOUT
P7_PIN	1FD7 _H	Вход порта 7. Разряд отображает текущее состояние вывода P7.7/FREQOUT независимо от конфигурации
P7_REG	1FD5 _H	Выход данных порта 7. Разряд 7 содержит данные выводимые на P7.7. В режиме специальной функции (FREQOUT) выводом управляет генератор программируемого тактового сигнала

Конфигурирование выхода

Выход генератора программируемого тактового сигнала мультиплексирован с P7.7, поэтому необходимо конфигурировать вывод как выход сигнала специальной функции. Для этого необходимо последовательно выполнить следующие действия:

- 1 Очистить разряд 7 регистра P7_DIR.
- 2 Установить разряд 7 регистра P7_MODE.
- 3 Очистить разряд 7 регистра P7_REG.

Программирование частоты

Программирование частоты на выходе генератора осуществляется регистром FREQ_GEN. Этот регистр удерживает записанное в него значение, которое определяет выходную частоту. Значение (FREQ) из регистра перегружается в счетчик обратного счета при достижении счетчиком нулевого значения.

Регистр FREQ_GEN

Адрес: 1FB8_H

Состояние сброса: 00_H

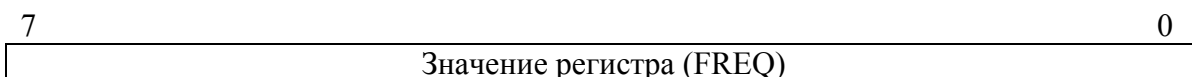


Рисунок 11.2 – Регистр частоты генератора программируемого тактового сигнала

Т а б л и ц а 11.3 – Разряды регистра частоты генератора программируемого тактового сигнала FREQ_GEN

Номер разряда	Функция
7 – 0	<p>Выходная частота Необходимо использовать следующую формулу для расчета значения (FREQ) регистра FREQ_GEN:</p> $FREQ = \frac{F_{BQ1}}{16 \times FREQ_OUT} - 1,$ <p>где FREQ – 8-разрядное значение регистра FREQ_GEN; F_{BQ1} – частота тактового сигнала на входе BQ1, в МГц; FREQ_OUT – частота выходного сигнала на выводе FREQOUT, МГц</p>

Определение текущего значения счетчика обратного счета

Для определения текущего значения счетчика обратного счета необходимо произвести чтение регистра `FREQ_CNT`.

Регистр `FREQ_CNT`

Адрес: `1FBAH`

Состояние сброса: `00H`



Рисунок 11.3 – Регистр счета `FREQ_CNT` генератора программируемого тактового сигнала

Т а б л и ц а 11.4 – Разряды регистра счета `FREQ_CNT` генератора программируемого тактового сигнала

Номер разряда	Функция
7 – 0	Текущее значение счетчика обратного счета

11.2 Применение генератора программируемого тактового сигнала

Одним из применений генератора программируемого тактового сигнала является управление инфракрасными диодами для передачи дистанционного управления данными и управляющими сигналами. Как показано на рисунках 11.4 и 11.5, генератор конфигурирован для сигнала частотой 40 кГц и включается и выключается записью седьмого разряда регистр `P7_MODE`. Информация передается сериями. Нулевой бит информации представлен несущей частотой длительностью 1 мс и паузой 1 мс, а единичный бит информации представлен несущей частотой длительностью 2 мс и паузой 2 мс. Фотодиод принимает световые импульсы, фильтр отсекает низкочастотные колебания и позволяет пройти несущей частоте 40 кГц. Несущая частота усиливается и детектируется для производства оригинальной последовательности импульсов.

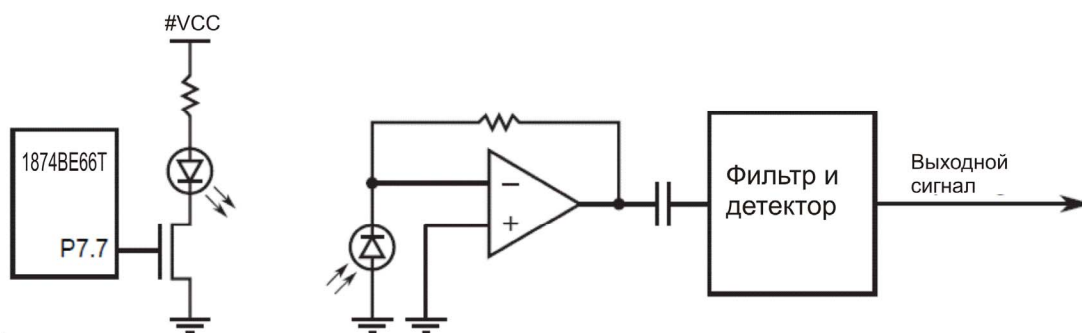


Рисунок 11.4 – Инфракрасное дистанционное управление

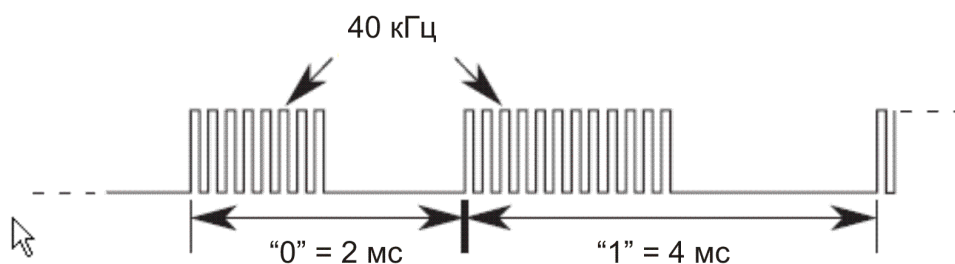


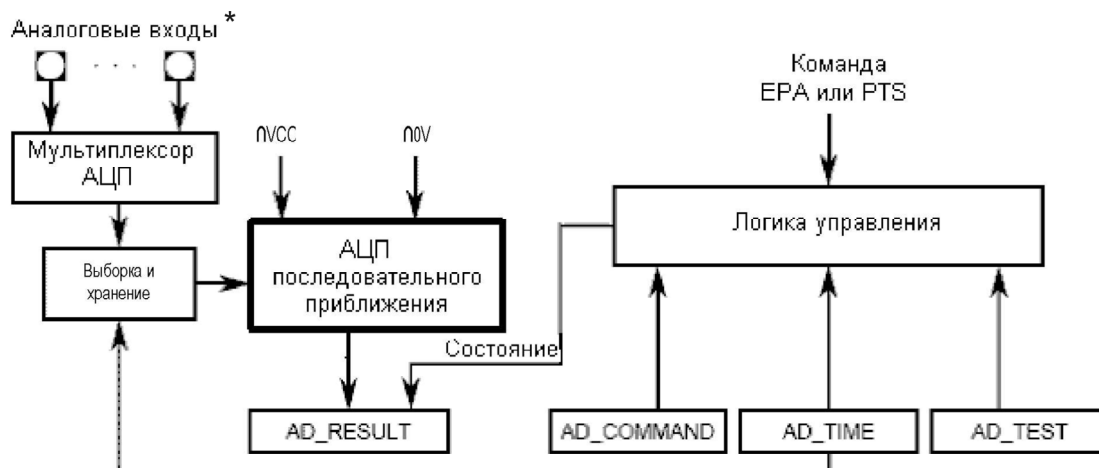
Рисунок 11.5 – Образец кодирования данных

12 Аналого-цифровой преобразователь АЦП

В этом разделе описывается работа АЦП и объясняется, как его программировать.

12.1 АЦП. Краткий функциональный обзор

Аналого-цифровой преобразователь АЦП преобразует аналоговое входное напряжение в 8- или 10-битное цифровое значение и устанавливает бит отложенного прерывания при сохранении результата. Преобразователь также контролирует вход, устанавливая бит отложенного прерывания, когда входное напряжение становится выше или ниже запрограммированного порогового значения. Структурная схема АЦП приведена на рисунке 12.1.



* Входы АЦП мультиплексированы с входами портов 0 и 1.

Рисунок 12.1 – Структурная схема АЦП

12.2 Сигналы и регистры АЦП

В таблице 12.1 приведен список сигналов АЦП, в таблице 12.2 описаны регистры состояния и управления. Хотя аналоговые входы мультиплексированы с выводами порта ввода-вывода, дополнительной конфигурации для этого не требуется.

Т а б л и ц а 12.1 – Выводы АЦП

Вывод порта	Сигнал АЦП	Тип сигнала АЦП	Описание
P1.5 – 0	АСН13 – 8	Вход	Аналоговые входы
P0.7 – 0	АСН7 – 0	Вход	Аналоговые входы
–	0V	Аналоговая земля	Аналоговая земля должна быть подключена для работы АЦП и порта
–	VCC	Аналоговое питание	Опорное напряжение должно быть подключено для работы АЦП и порта

Т а б л и ц а 12.2 – Регистры управления и состояния АЦП

Мнемоника	Адрес	Описание
AD_COMMAND	1FAC _H	Регистр команд АЦП. Этот регистр выбирает АЦ канал, определяет, начинается ли АЦ преобразование немедленно или по команде ЕРА и выбирает режим преобразования
AD_RESULT	1FAA _H 1FAB _H	Регистр результата АЦ преобразования. Для АЦ преобразования старший байт содержит восемь старших битов преобразованного кода. Младший байт содержит два младших бита при 10-битном преобразовании (не определены при 8-битном преобразовании), показывает, какой АЦ канал использовался и не занят ли канал. Для определения пороговой величины вычисляют значение для регистра последовательного приближения и записывают значение в старший байт AD_RESULT. Младший байт очищают или оставляют его в заданном по умолчанию состоянии
AD_TEST	1FAE _H	Регистр проверки АЦ преобразования. Регистр осуществляет коррекцию ошибок смещения нуля
AD_TIME	1FAF _H	Регистр времени АЦ преобразования. Этот регистр определяет время выборки и время преобразования для каждого бита
INT_MASK	0008 _H	Регистр маски прерывания. Бит AD этого регистра разрешает или запрещает прерывание АЦП. Установка AD бита разрешает запрос этого прерывания
INT_PEND	0009 _H	Регистр отложенного прерывания. Установленный бит AD этого регистра показывает, что запрос АЦ прерывания отложен
P0_PIN	1FA8 _H	Регистр состояния выводов порта 0. Чтение P0_PIN показывает текущие значения выводов порта 0. Чтение порта вызывает помехи в АЦП, уменьшая точность текущего преобразования. Настоятельно рекомендуется не производить чтение порта во время АЦ преобразования. Для уменьшения помех регистр P0_PIN тактируется только тогда, когда порт читается
P1_PIN	1FA9 _H	Регистр состояния выводов порта 1. Чтение P1_PIN показывает текущие значения выводов порта 1. Чтение порта вызывает помехи в АЦП, уменьшая точность текущего преобразования. Настоятельно рекомендуется не производить чтение порта, во время АЦ преобразования. Для уменьшения помех регистр P1_PIN тактируется только тогда, когда порт читается

12.3 Работа АЦП

АЦП преобразует аналоговое входное напряжение в цифровое значение, хранит результат в регистре AD_RESULT и устанавливает бит отложенного прерывания АЦП. 8-битное преобразование обеспечивает разрешение 20 мВ, 10-битное преобразование обеспечивает разрешение 5 мВ. 8-битное преобразование требует меньше времени, чем 10-битное, потому что используется на два бита меньше для преобразования и компаратор

требует меньшего времени установки для выполнения преобразования с шагом 20 мВ, чем с шагом 5 мВ.

Имеется возможность преобразовывать напряжение на аналоговом входном канале или тестовое напряжение. Преобразование тестовых входов позволяет вычислить ошибку смещения нуля, а коррекция смещения нуля позволяет ее компенсировать. Эта особенность позволяет уменьшить или устранить внешнюю аппаратную компенсацию. Обычно преобразование тестового напряжения используется для коррекции ошибки смещения нуля перед выполнением преобразованиями на входном канале. Регистр AD_TEST позволяет программировать коррекцию смещения нуля.

Пороговый детектор сравнивает входное напряжение с запрограммированным значением опорного напряжения и устанавливает бит ожидания прерывания, когда входное напряжение становится ниже или выше опорного.

Преобразование может быть начато записью в регистр AD_COMMAND или вызвано EРА, который может обеспечивать выборки с равными интервалами или синхронно с внешними событиями. Режим АЦ сканирования сервера периферийных транзакций PTS позволяет выполнять многократные преобразования и хранить их результаты.

Как только АЦП получает команду начать преобразование, образуется время задержки до начала выборки (преобразования, запущенные модулем EРА, начинаются после события захвата/сравнения, преобразования, вызванные прямой записью в AD_COMMAND, начинаются в течение трех циклов после завершения команды.) В течение времени этой задержки выборки аппаратные средства очищают регистр последовательного приближения и выбирают заданный канал входного мультиплексора. После этой задержки устройство подключает выход мультиплексора к запоминающей емкости на указанное время выборки. После завершения этого времени устройство отключает выход мультиплексора от запоминающей емкости так, чтобы изменения на входном выводе не изменили хранимый заряд, во время процесса преобразования. Затем устройство обнуляет компаратор и начинает преобразование.

При выполнении аналого-цифрового преобразования используется алгоритм последовательного приближения. В состав аппаратуры преобразователя входят резистивное дерево с 256 резисторами, компаратор, конденсаторы и 10-битный регистр последовательного приближения (SAR) с логикой управления процессом. Резистивное дерево обеспечивает шаг в 20 мВ ($U_{NVCC} = 5,12 \text{ В}$), в то время как цепочка емкостей создает шаг в 5 мВ внутри шага в 20 мВ. Таким образом, имеется 1024 уровня опорного напряжения для сравнения с аналоговым входом для генерации 10-битного двоичного результата. В 8-битном режиме преобразования используется только резистивное дерево, обеспечивающее 256 уровней опорного напряжения.

Алгоритм последовательного приближения осуществляет сравнение последовательности заданных напряжений резистивного дерева с величиной на аналоговом входе методом половинного деления, чтобы обеспечить лучшее приближение. Первое сравнение – с 1/2 значения полного опорного напряжения. Оно дает 10-битный двоичный код, в котором старший значащий бит равен нулю, остальные – единицы (01111111_B). Если на аналоговом входе значение меньше, чем контрольное значение, бит 10 в SAR останется нулевым, и вход будет сравниваться с новым контрольным напряжением, равным 1/4 полной шкалы (00111111_B). Если значение на аналоговом входе больше тестового напряжения, устанавливается 9 бит SAR. 8 бит затем очищаются для следующего теста (01011111_B). Такой бинарный поиск продолжается, пока не будут произведены 10 (или 8) сравнений, к этому времени результат преобразования будет располагаться в регистре AD_RESULT, откуда может быть программно считан. Результат равен отношению входного напряжения к напряжению аналогового источника. Двоичный результат будет содержать все единицы, если отношение равно 1,00.

12.4 Программирование АЦП

Программируются следующие параметры АЦП:

- вход преобразователя – выбор входного канала;
 - подстройка смещения нуля – нет коррекции, плюс 2,5 мВ, минус 2,5 мВ, минус 5,0 мВ;
 - временные параметры преобразования – время выборки и время преобразования для каждого бита;
 - режим преобразования – 8- или 10-битное преобразование, детектирование низкого или высокого порога;
 - запуск преобразования – непосредственный или запуск модулем ЕРА.
- Далее приводится описание регистров АЦП и способы их программирования.

Программирование регистра проверки АЦ преобразования

Регистр AD_TEST (рисунок 12.2, таблица 12.3) определяет напряжение смещения, которое будет приложено к резистивному дереву. Чтобы использовать коррекцию смещения нуля, сначала выполняют два преобразования: одно на $\Pi 0V$, другое на ΠVCC . Результаты этих преобразований используются для программного вычисления ошибки смещения нуля. Корректировка смещения нуля производится с помощью записи соответствующего значения в AD_TEST. Это напряжение смещения добавляется к резистивному дереву и используется со всеми входными каналами.

Регистр AD_TEST

Адрес: 1FAE_H

Состояние сброса: C0_H

Регистр проверки АЦ преобразования (AD_TEST) определяет установки корректировки ошибок смещения по постоянному току.

7	–	–	–	OFF1	–	OFF0	–	0
---	---	---	---	------	---	------	---	---

Рисунок 12.2 – Регистр проверки АЦ преобразования (AD_TEST)

Т а б л и ц а 12.3 – Разряды регистра проверки АЦ преобразования (AD_TEST)

Номер разряда	Мнемоника	Функция															
7 – 5	–	Зарезервированы; для совместимости с будущими устройствами в эти биты записывают нули															
4	OFF1	Смещение. Этот бит, наряду с OFF0 (бит 2), позволяет устанавливать точку смещения нуля. <table style="margin-left: 20px;"> <tr> <td>OFF1</td> <td>OFF0</td> <td></td> </tr> <tr> <td>0</td> <td>0</td> <td>нет коррекции</td> </tr> <tr> <td>0</td> <td>1</td> <td>прибавление 2,5 мВ</td> </tr> <tr> <td>1</td> <td>0</td> <td>вычитание 2,5 мВ</td> </tr> <tr> <td>1</td> <td>1</td> <td>вычитание 5,0 мВ</td> </tr> </table>	OFF1	OFF0		0	0	нет коррекции	0	1	прибавление 2,5 мВ	1	0	вычитание 2,5 мВ	1	1	вычитание 5,0 мВ
OFF1	OFF0																
0	0	нет коррекции															
0	1	прибавление 2,5 мВ															
1	0	вычитание 2,5 мВ															
1	1	вычитание 5,0 мВ															
3	–	Зарезервирован; для совместимости с будущими устройствами в этот бит записывают ноль															
2	OFF0	Смотри бит 4 (OFF1)															
1, 0	–	Зарезервированы; для совместимости с будущими устройствами в эти биты записывают нули															

Программирование регистра результата АЦ преобразования (только для детектирования порога)

Чтобы использовать режим детектирования порога сначала необходимо записать в старший байт регистра AD_RESULT значение для установки желаемого опорного (порогового) напряжения.

Регистр AD_RESULT (запись)

Адрес: 1FAA_H

Состояние сброса: FFC0_H

В старший байт регистра результата АЦ преобразования (AD_RESULT) может быть записано значение для установки опорного напряжения в режиме детектирования порога.

15	8
REFV7	REFV0
7	0
-	-

Рисунок 12.3 – Формат записи регистра результата АЦ преобразования (AD_RESULT)

Т а б л и ц а 12.4 – Разряды регистра результата АЦ преобразования (AD_RESULT)

Номер разряда	Мнемоника	Функция
15 – 8	REFV7 – REFV0	Опорное напряжение. Эти биты определяют пороговое значение. Они выбирают опорное напряжение для сравнения с аналоговым входным напряжением. Когда напряжение на аналоговом входе становится выше или ниже порогового значения, бит отложенного прерывания по завершению АЦ преобразования устанавливается. Для определения значения, записываемого в регистр, необходимо воспользоваться следующей формулой Опорное напряжение = $\frac{(\text{желаемое пороговое напряжение}) \times 256}{(U_{\text{ПВСС}} - U_{\text{П0V}})}$
7 – 0	-	Зарезервированы; для совместимости с будущими устройствами в эти биты записывают нули

Программирование регистра времени АЦ преобразования

Два параметра – время выборки и время преобразования, управляют временем, требуемым для АЦ преобразования. Время выборки – это время, в течение которого аналоговый вход подключен к запоминающей емкости. Если это время слишком коротко, емкость не зарядится полностью. Если время выборки слишком затянуто, входное напряжение может измениться, что вызовет ошибку преобразования. Время преобразования – это время, требуемое для преобразования величины напряжения, сохраненной на запоминающей емкости, в цифровое значение. Время преобразования должно быть достаточным, чтобы компаратор и схема преобразования успели установиться и вычислить напряжение. Слишком большое время преобразования позволяет запоминающей емкости разрядиться и уменьшает точность.

Регистр AD_TIME (рисунок 12.4, таблица 12.5) определяет время выборки и время АЦ преобразования.

Регистр AD_TIME

Адрес: 1FAFH

Состояние сброса: FFH

Регистр времени АЦП (AD_TIME) программирует время выборки и время преобразования для каждого бита. Этот регистр программирует скорость, с которой может идти АЦ преобразование, но не скорость, с которой это преобразование будет проходить корректно. Необходимо инициализировать регистр AD_TIME перед инициализацией регистра AD_COMMAND. Нельзя записывать значения в этот регистр во время преобразования – результаты непредсказуемы.

7								0
SAM2	SAM1	SAM0	CONV4	CONV3	CONV2	CONV1	CONV0	

Рисунок 12.4 – Регистр времени АЦ преобразования (AD_TIME)

Т а б л и ц а 12.5 – Разряды регистра времени АЦ преобразования (AD_TIME)

Номер разряда	Мнемоника бита	Функция
7 – 5	SAM2 – SAM0	Время выборки АЦП. Эти разряды определяют время выборки. Для вычисления времени выборки используется следующая формула: $SAM = \frac{T_{SAM} \times F_{BQ1} - 2}{8},$ где SAM – от 1 до 7; T _{SAM} – время выборки, мкс; F _{BQ1} – входная частота на BQ1, МГц
4 – 0	CONV4 – CONV0	Время АЦ преобразования. Эти биты определяют время преобразования для каждого бита. Для вычисления времени преобразования используется следующая формула: $CONV = \frac{T_{CONV} \times F_{BQ1} - 3}{(2 \times B)} - 1,$ где CONV – от 2 до 31; T _{CONV} – время преобразования, мкс; F _{BQ1} – входная частота на BQ1, МГц; B – число битов, которые будут преобразованы (8 или 10)

Программирование регистра команд АЦП (AD_COMMAND)

Регистр команд АЦП управляет режимом преобразования, аналоговым входным каналом и запуском преобразования.

Регистр AD_COMMAND

Адрес: 1FACH

Состояние сброса: 80H

Регистр команд АЦП (AD_COMMAND) выбирает номер АЦ канала, который будет преобразован, определяет, начинается ли АЦ преобразование немедленно или вызывается командой EPA, и выбирает режим преобразования.

7							0
-	M1	M0	GO	ACH3	ACH2	ACH1	ACH0

Рисунок 12.5 – Регистр команд АЦП (AD_COMMAND)

Т а б л и ц а 12.6 – Разряды регистра команд АЦП (AD_COMMAND)

Номер разряда	Мнемоника	Функция															
7	–	Зарезервирован, для совместимости с будущими устройствами записывают в этот бит ноль															
6, 5	M1, M0	Режим АЦ преобразования ¹⁾ . Эти биты определяют режим АЦП. <table style="margin-left: 20px;"> <tr> <td>M1</td> <td>M0</td> <td>Режим</td> </tr> <tr> <td>0</td> <td>0</td> <td>10-битное преобразование</td> </tr> <tr> <td>0</td> <td>1</td> <td>8-битное преобразование</td> </tr> <tr> <td>1</td> <td>0</td> <td>обнаружение напряжения выше порогового</td> </tr> <tr> <td>1</td> <td>1</td> <td>обнаружение напряжения ниже порогового</td> </tr> </table>	M1	M0	Режим	0	0	10-битное преобразование	0	1	8-битное преобразование	1	0	обнаружение напряжения выше порогового	1	1	обнаружение напряжения ниже порогового
M1	M0	Режим															
0	0	10-битное преобразование															
0	1	8-битное преобразование															
1	0	обнаружение напряжения выше порогового															
1	1	обнаружение напряжения ниже порогового															
4	GO	Запуск АЦ преобразования ²⁾ . Запись этого бита запускает АЦП. Значение, записанное в него, определяет, когда должно начаться преобразование. 0 ЕРА запускает преобразование, 1 начинается немедленно															
3 – 0	АСН3 – АСН0	Выбор канала АЦП. В эти биты записывается номер канала АЦП															

¹⁾ Когда выбран режим детектора порога для аналогового входа, никакое другое преобразование не может быть начато. Если другое значение загружено в AD_COMMAND, режим детектора порога запрещен и выполняется новая команда.

²⁾ Преобразование запускается записью в бит GO, а не его значением. Даже если бит GO имеет нужное значение, его необходимо установить снова, чтобы начать преобразование немедленно или же повторно очистить его для запуска преобразования модулем ЕРА.

Разрешение прерывания АЦП

АЦП может установить бит ожидания прерывания АЦП, когда преобразование заканчивается или когда входное напряжение переходит пороговое значение. Для разрешения прерывания необходимо установить соответствующий маскирующий бит регистра маскирования прерываний и выполнить EI инструкцию, чтобы полностью разрешить обслуживание прерываний. Прерывание АЦП может заставить PTS начать новое преобразование.

12.5 Определение состояния АЦП и результатов преобразования

Чтобы определить состояние АЦП, осуществляется чтение из регистра AD_RESULT (рисунок 12.6). Регистр AD_RESULT очищается, когда новое преобразование начато. Поэтому, чтобы предотвратить потерю данных, необходимо прочитать оба байта прежде, чем начнется новое преобразование. Если чтение из AD_RESULT происходит прежде, чем преобразование завершено, не может быть гарантирована точность результата.

Результат преобразования – отношение входного напряжения к эталонному напряжению:

$$\text{RESULT (8-бит)} = \frac{255 \times (V_{\text{IN}} - U_{\text{POV}})}{U_{\text{PVCC}} - U_{\text{POV}}}, \quad (12.1)$$

$$\text{RESULT (10-бит)} = \frac{1023 \times (V_{\text{IN}} - U_{\text{POV}})}{U_{\text{PVCC}} - U_{\text{POV}}}. \quad (12.2)$$

Можно также читать регистр отложенного прерывания INT_PEND (бит AD) для определения состояния АЦ прерывания.

Регистр AD_RESULT (Чтение)

Адрес: 1FAA_H

Состояние сброса: FFC0_H

15								8
ADRLT9	ADRLT8	ADRLT7	ADRLT6	ADRLT5	ADRLT4	ADRLT3	ADRLT2	
7								0
ADRLT1	ADRLT0	–	STATUS	ACH3	ACH2	ACH1	ACH0	

Рисунок 12.6 – Формат чтения регистра результата АЦП (AD_RESULT)

Т а б л и ц а 12.7 – Разряды регистра результата АЦП (AD_RESULT)

Номер разряда	Мнемоника	Функция
15 – 6	ADRLT9 – ADRLT0	Результат АЦП. Эти биты содержат результат АЦ преобразования
5	–	Зарезервирован. Этот бит не определён
4	STATUS	Состояние АЦП. Показывает состояние АЦП. До 8 тактов требуется, чтобы установить этот бит после начала команды. При проверке этого бита необходимо ждать, по крайней мере, 8 тактов. 0 АЦ преобразование не выполняется. 1 АЦ преобразование выполняется
3 – 0	ACH3 – ACH0	Номер АЦ канала. Эти биты указывают номер АЦ канала, который использовался для преобразования

12.6 Примеры схем внешнего интерфейса и ошибок АЦ преобразования

Этот подраздел описывает примеры внешней схемы интерфейса и ошибки, которые могут произойти в любом АЦП.

Проектирование внешней схемы интерфейса

Внешняя схема интерфейса с аналоговым входом определяется конкретным применением и влияет на характеристики преобразователя. При разработке внешней схемы такие факторы, как входной ток утечки, запоминающая емкость, последовательное сопротивление мультимплексора от вывода до конденсатора выборки должны быть правильно выбраны. Эти факторы (с идеализированными значениями и схемой) приведены на рисунке 12.7.

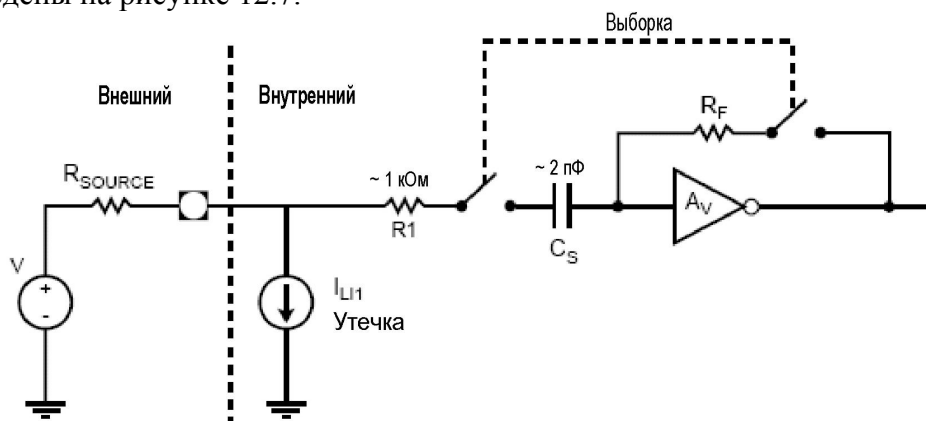


Рисунок 12.7 – Идеализированная схема интерфейса АЦП

В течение времени выборки внешняя входная схема должна быть способна зарядить типовой конденсатор (C_s) через последовательное сопротивление источника (R_{SOURCE}), входное сопротивление схемы (R_I) и сопротивление обратной связи компаратора (R_F). Полное эффективное последовательное сопротивление R_T рассчитывается с использованием следующей формулы:

$$R_T = R_{SOURCE} + R_I + \frac{R_F}{A_V + 1}, \quad (12.3)$$

где A_V – коэффициент усиления схемы компаратора.

Уменьшение эффекта высокого входного исходного сопротивления

При некоторых условиях входное сопротивление источника (R_{SOURCE}) может быть достаточно большим, чтобы повлиять на измерение. Можно минимизировать этот эффект, увеличивая время выборки или подключая внешний конденсатор (C_{EXT}) между входным выводом и $\cap 0V$. Внешний сигнал зарядит C_{EXT} до уровня напряжения источника. Когда канал выбран, C_{EXT} действует как источник с низким импедансом при заряде конденсатора выборки C_s . Малая часть заряда C_{EXT} передается C_s , происходит снижение выбранного напряжения. Снижение напряжения (в %) рассчитывается исходя из следующей формулы:

$$\text{снижение выбранного напряжения} = \frac{C_s}{C_{EXT} + C_s} \times 100. \quad (12.4)$$

Если C_{EXT} равно 0,005 мкФ или больше, ошибка будет меньше, чем ($-0,4$) LSB при 10-битном режиме преобразования. Использование C_{EXT} в соединении с R_{SOURCE} формирует фильтр низких частот на входе, снижая помеху на входе.

Высокое R_{SOURCE} может также стать причиной ошибки из-за входного тока утечки (I_{LII}). Ток I_{LII} обычно намного ниже, чем его нормы в технической документации. Объединенный эффект тока утечки и высокого R_{SOURCE} рассчитывается по следующей формуле:

$$\text{ошибка (LSBs)} = \frac{R_{SOURCE} \times I_{LII} \times 1024}{U_{\cap VCC}}, \quad (12.5)$$

где R_{SOURCE} – входное сопротивление источника, Ом;

I_{LII} – ток утечки на входе, А;

$U_{\cap VCC}$ – опорное напряжение, В.

Во внешних схемах с сопротивлением R_{SOURCE} , равным 1 кОм или ниже, и $U_{\cap VCC}$, равным 5,0 В из-за импеданса источника будет происходить ошибка 0,6 LSB или меньше.

Примеры входных схем АЦП

Предложенная схема входа АЦП показана на рисунке 12.8, она обеспечивает защиту против перенапряжения на аналоговом вводе. При входном напряжении ниже $U_{\cap 0V}$ или выше $U_{\cap VCC}$ диод D2 или D1 осуществляет прямое смещение около 0,8 В. Защита входа устройства начинает включаться при отклонении приблизительно 0,5 В от напряжения $U_{\cap 0V}$ или $U_{\cap VCC2}$. Резистор 270 Ом ограничивает входной ток на аналоговом входе до безопасного значения, меньше 1 мА.

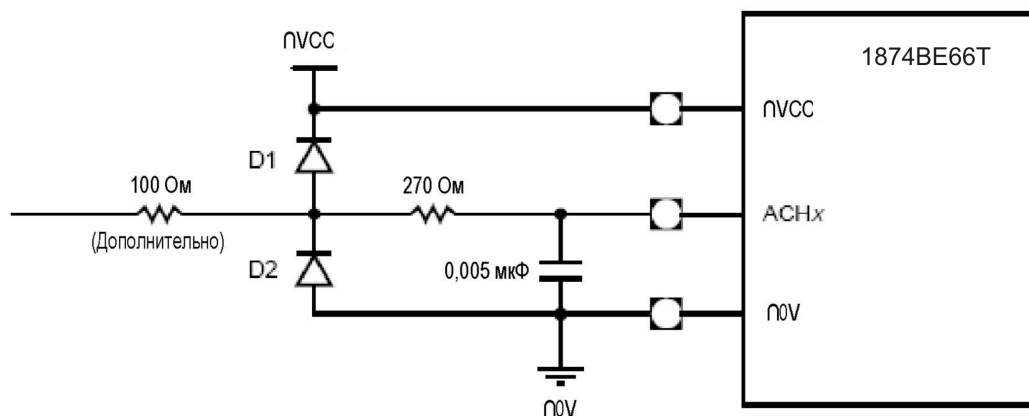


Рисунок 12.8 – Рекомендуемая входная схема АЦП

Примечание – При подключении любого аналогового входа к напряжению, более чем на 0,5 В превышающему U_{0V} или U_{VCC} , включается входное защитное устройство. Появляющийся при этом ток во внутренней цепи источника опорного напряжения существенно уменьшает точность АЦ преобразований на всех каналах.

Аналоговая земля и источник опорного напряжения

Пульсации опорного напряжения сильно влияют на абсолютную точность преобразования. По этой причине рекомендуется соединять вывод $0V$ с выводом $\#0V$ как можно ближе к устройству, используя минимальную длину связей. При внешних помехах настоятельно рекомендуется использовать отдельную шину $0V$, которая соединяется с $\#0V$ в единственной точке, как можно ближе к устройству. I_{OCC2} может измениться между 2 мА и 5 мА в течение преобразования. Для минимизации этого эффекта следует использовать фильтрующий керамический или танталовый конденсатор 1,0 мкФ между VCC и $0V$. Потенциал U_{0V} должен быть в пределах ± 50 мВ по отношению к $U_{\#0V}$. VCC должен быть стабильным и использоваться только для АЦП. Питание U_{VCC} может быть между 4,5 В и 5,5 В и должно обеспечивать ток приблизительно 5 мА. U_{VCC} должно быть приблизительно равно $U_{\#VCC}$. Большие отрицательные пики токов на выводе $0V$ относительно $\#0V$ могут вызвать эффект «защёлки» в аналоговой цепи. Это является дополнительной причиной, из-за которой следует тщательно заземлять устройство.

Аналоговое опорное напряжение (U_{VCC}) является положительным напряжением, с которым происходит сравнение при АЦ преобразовании. Это также питание и порта 0, и порта 1, если АЦП не используется. Если высокая точность не требуется, VCC может быть соединен с $\#VCC$. Если необходима высокая точность преобразования, VCC должен быть очень стабильным. Один из путей достижения этого – использование прецизионных источников напряжения или отдельного стабилизатора напряжения. Эти устройства должны быть связаны с $0V$, а не с $\#0V$, чтобы гарантировать, что VCC имеет отдельную землю $0V$, а не соединяется с $\#0V$.

Совместное использование аналоговых и цифровых входов

Порт 0 и порт 1 могут использоваться и для аналоговых, и для цифровых входных сигналов одновременно. Однако чтение порта может наводить помехи в аналоговую цепь. По этой причине необходимо гарантировать, что не идет процесс АЦ преобразования, когда требуется читать порт. В разделе 7 «Порты ввода-вывода» описано использование порта как цифровых входов.

Передающая функция и источники ошибок АЦ преобразования

Результат преобразования – это отношение входного напряжения к опорному.

$$\text{RESULT (8-бит)} = \frac{255 \times (V_{\text{IN}} - U_{\text{0V}})}{U_{\text{VCC}} - U_{\text{0V}}}, \quad (12.6)$$

$$\text{RESULT (10-бит)} = \frac{1023 \times (V_{\text{IN}} - U_{\text{0V}})}{U_{\text{VCC}} - U_{\text{0V}}}. \quad (12.7)$$

Получается ступенчатая передающая функция, когда выходной код отображает значение входного напряжения. Полученный цифровой код может трактоваться как информация о простом соотношении напряжений, абсолютном значении напряжения на входе или его относительном изменении.

Чем больше требований предъявляется к преобразователю, тем более важно полное понимание операции преобразования. Для простого применения достаточно знания абсолютной погрешности преобразования. Однако чтобы реализовать следящую обратную связь с аналоговыми входами, требуется полное понимание процесса преобразования и ошибок АЦП.

Во многих применениях менее критична абсолютная точность на входе, чем изменение на входе. Этот подход допустим, пока идет преобразование непрерывного сигнала, и нет ошибочных кодов.

Увеличение входных напряжений приводит к тому, что выходные коды также увеличиваются. Уменьшение входных напряжений приводит к тому, что выходные коды также уменьшаются. Другими словами, существует определенный диапазон входных напряжений для каждого 10-битного выходного кода, который производится с ошибкой только $\pm 0,25 \text{ LSBs}$ (1,5 мВ).

Внутренние ошибки АЦ преобразования следующие: ошибка дискретизации, ошибка смещения нуля, ошибка усиления, дифференциальная и интегральная нелинейности.

Кроме того, должны учитываться температурные коэффициенты, флуктуации $U_{\text{#VCC}}$, перекрытие выборки и хранения, взаимное влияние каналов и случайный шум. Обычно абсолютная ошибка включает в себя общую сумму ошибок и все расхождения между реальным процессом преобразования и идеальным. Однако, отдельные составляющие ошибки важны в конкретных разработках.

Неизбежная ошибка следует из преобразования непрерывного напряжения к целому числу в цифровом представлении. Эта ошибка, называемая ошибкой квантования, всегда $\pm 0,5 \text{ LSB}$. Ошибка квантования – единственная ошибка, замеченная в идеальном АЦП, и она, очевидно, присутствует в реальных преобразователях. На рисунке 12.9 приведена передающая функция идеального 3-битного АЦП.

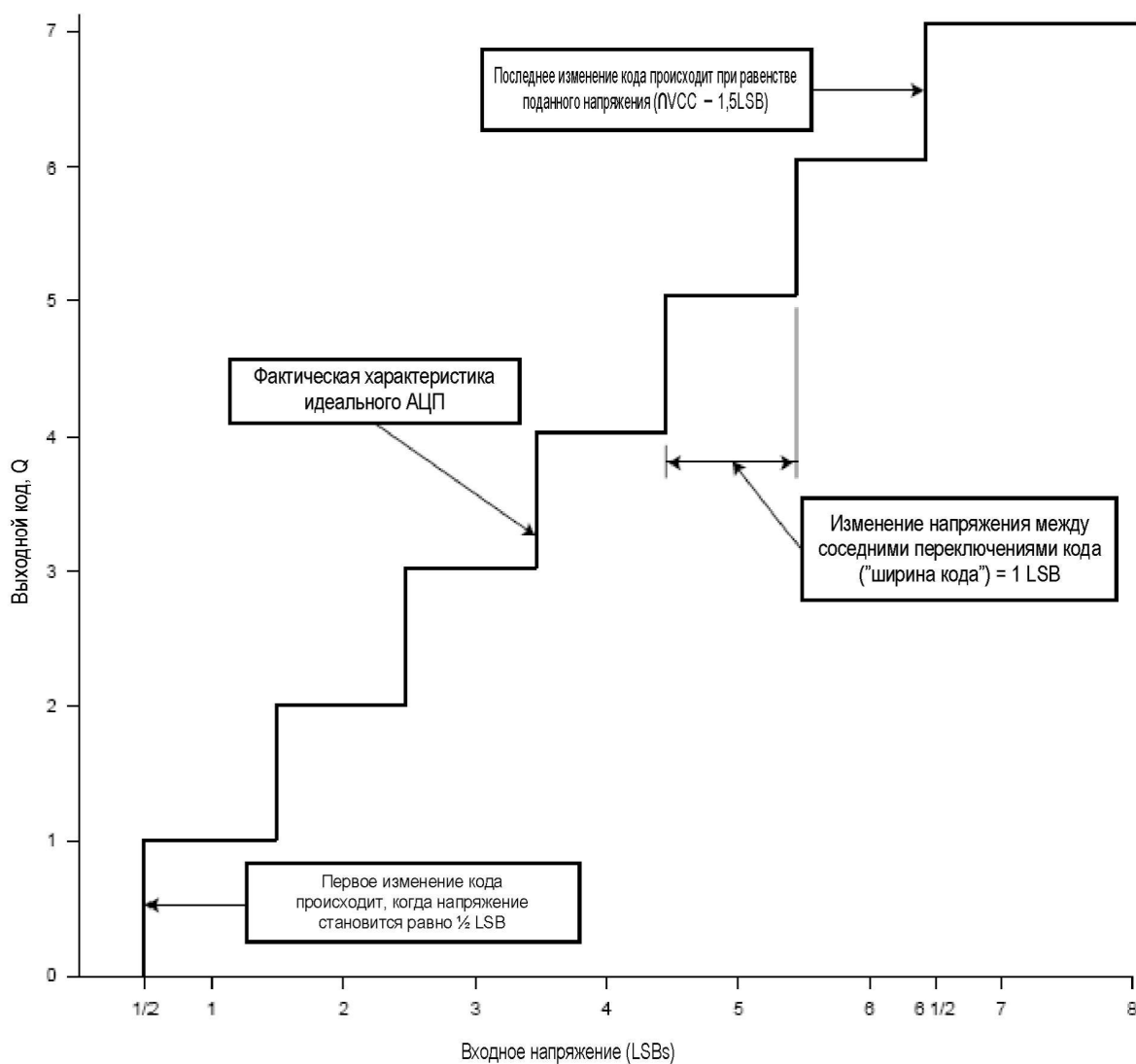


Рисунок 12.9 – Идеальная передаточная характеристика АЦП

Идеальная характеристика обладает следующими особенностями:

- появление младшего разряда кода происходит при входном напряжении 0,5 LSB;
- максимальное значение кода достигается, когда входное напряжение равно ($U_{NVCC} - 1,5 \text{ LSB}$);
- величина интервала (кванта) равна 1 LSB.

Эта характеристика не учитывает смещение нуля, ошибок полного диапазона и нелинейностей. Другими словами, это идеальное преобразование.

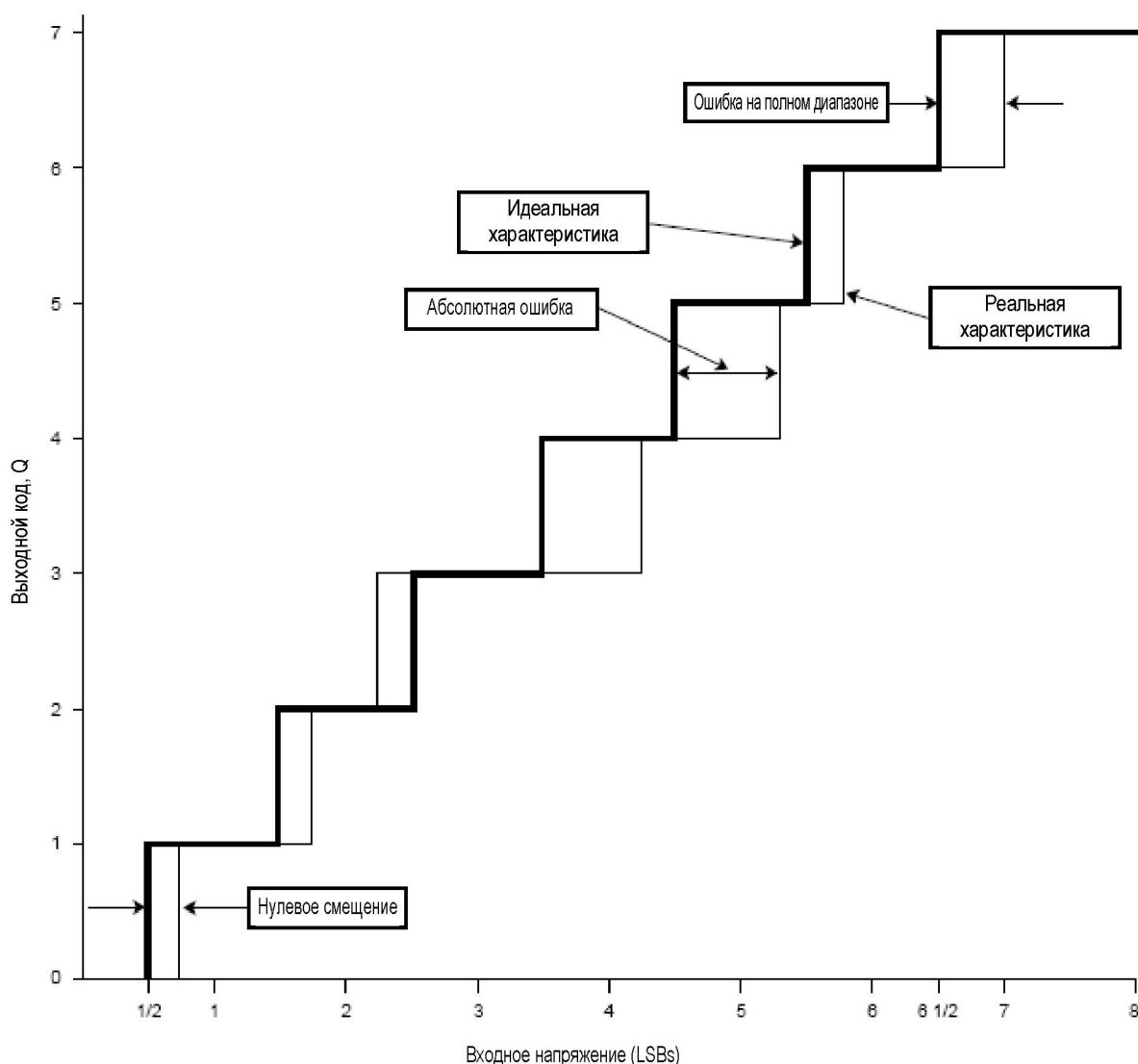


Рисунок 12.10 – Реальная и идеальная передаточные характеристики АЦП

Реальная характеристика 3-битного преобразователя отличается от идеальной. Если идеальную характеристику сопоставить с реальной, то реальная характеристика, как показано на рисунке 12.10, имеет как ошибки в расположении минимального и максимального кодового напряжения, так и в величине интервала (кванта). Смещение минимального значения кода от идеального называется ошибкой нулевого смещения, смещение максимального кодового значения от идеального называется ошибкой на полном диапазоне (fullscale error). Отклонение величины интервала (кванта) от идеальной характеристики вызвано двумя типами ошибок: дифференциальной и интегральной нелинейностью

Интегральная нелинейность определяется максимальным отклонением передаточной характеристики от идеальной прямолинейной характеристики при нулевых значениях смещения нуля и ошибки на полной шкале.

Дифференциальная нелинейность – это отклонение величины одного из квантов от его идеального значения. Если дифференциальная нелинейность превышает 1 LSB, то в выходном сигнале может отсутствовать одна из кодовых комбинаций (выпадающий код). В 10-битном преобразователе идеальный шаг квантования 5 мВ ($U_{NVCC}/1024$). Если такой преобразователь будет иметь максимальную дифференциальную нелинейность 2 LSB (10 мВ), тогда максимальная величина кванта будет не более чем на 10 мВ больше идеальной или 15 мВ.

Реальная величина кванта в этом преобразователе обычно изменяется от 2,5 мВ до 7,5 мВ. Ошибка дифференциальной нелинейности в ширине одного кванта компенсируется шириной другого кванта, так что сохраняется 1024 шага квантования.

Интегральная нелинейность вызывает в худшем случае отклонение реального кода от соответствующего кода идеальной характеристики. Интегральная нелинейность характеризует, насколько накопление дифференциальных нелинейностей по отдельным квантам может увеличить общий уход от линейной характеристики. Если ошибки дифференциальной нелинейности слишком велики, возможно выпадение кода или нарушение монотонности. Ни то, ни другое нежелательно в контурах управления. Преобразователь не имеет выпадающих кодов, если для каждого выходного кода существует входной диапазон напряжения, который характеризуется только этим кодом. Преобразователь является монотонным, если каждое последующее изменение кода вызвано изменением входного напряжения в том же направлении.

Дифференциальная нелинейность и интегральная нелинейность определяются при измерении ошибок общей (Terminal Based) линейности. Базовую выходную (Terminal Based) характеристику получают из реальной характеристики, преобразуя и масштабируя ее для уничтожения ошибок смещения нуля и ошибки на полном диапазоне, как показано на рисунке 12.11. Terminal Based характеристика подобна реальной, в которой ошибки нулевого смещения и ошибка на полном диапазоне компенсированы подстройкой. На практике это достигается использованием входных цепей, которые обеспечивают подстройку усиления и компенсацию смещения нуля. К тому же U_{HVCC} может точно регулироваться в заданном диапазоне для уменьшения ошибки полного диапазона.

Другие факторы, которые воздействуют на реальную систему АЦП, включают: температурный дрейф, импульсные помехи, взаимное влияние каналов мультиплексора и случайный шум. Но обычно эти влияния являются незначительными. Температурный дрейф – это скорость изменения характеристик с изменением температуры. Эти изменения отражаются в температурных коэффициентах.

Основные источники паразитных сигналов – это помехи по питанию, изменение входного сигнала в канале во время преобразования (после заряда эталонной емкости) и сигналы на каналах, не выбранных мультиплексором.

Входное сопротивление мультиплексора немного отличается на разных каналах, что приводит к появлению при одинаковом входном сигнале различных значений на разных каналах и на том же канале при повторных преобразованиях. Различия в токах утечки в разных каналах и случайный шум вызывают ошибки повторного измерения.

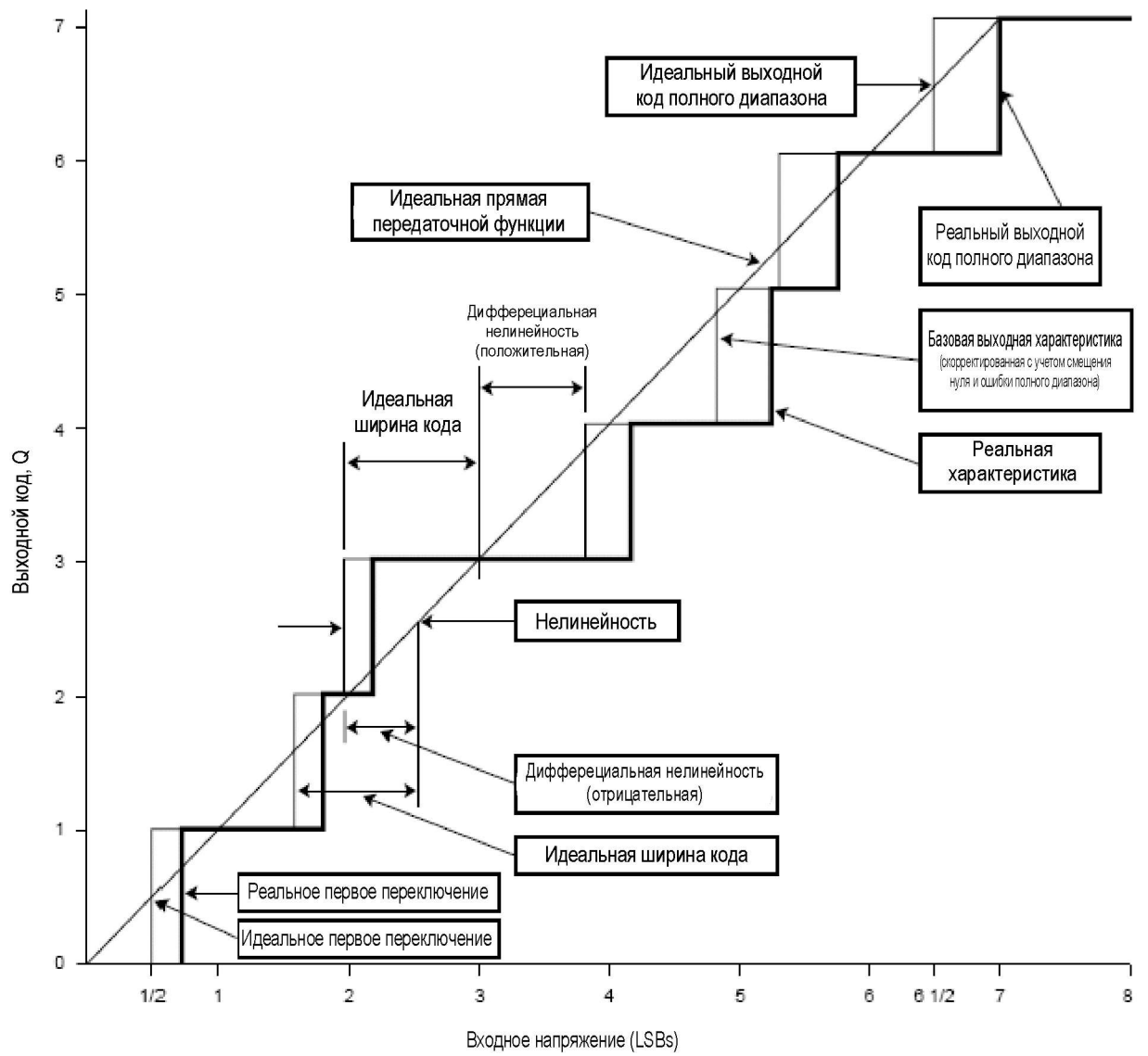


Рисунок 12.11 – Выходная передаточная характеристика АЦП

13 Специальные режимы работы

В микроконтроллере 1874BE66T предусмотрены два режима экономии мощности: холостой ход IDLE и низкое потребление POWERDOWN. Также предусмотрен режим внутрисхемного эмулятора ONCE, который электрически изолирует устройство от других компонентов системы. В этом разделе описываются указанные режимы, и вход, и выход из них.

13.1 Сигналы и регистры управления специальными режимами

В таблице 13.1 приведен список сигналов, в таблице 13.2 приведен список регистров, работа которых в специальных режимах рассматривается в этом разделе.

Т а б л и ц а 13.1 – Режим работы сигналов управления

Вывод порта	Название сигнала	Тип	Описание
1	2	3	4
P2.7	CLKOUT	Выход	Выход системного тактового сигнала. Выход внутреннего генератора синхроимпульсов. Частота CLKOUT – 1/2 частоты на выводе генератора (F _{BQ1}). CLKOUT имеет 50 % скважность
–	EXINT	Вход	Внешнее прерывание. Это программируемое прерывание управляется регистром WG_PROTECT. Этот регистр определяет, вызывается прерывание фронтом или уровнем и передним фронтом, высоким уровнем, или задним фронтом, низким уровнем. В режиме POWERDOWN он используется как вход, чувствительный к уровню. Установка сигнала EXINT, по крайней мере, на 50 нс вызывает выход из режима. Прерывание при этом может быть запрещено. Если прерывание от EXINT разрешено, начинает выполняться процедура обслуживания прерывания. Иначе, центральный процессор выполняет инструкцию, которая сразу следует за командой, которая вызвала режим POWERDOWN. В режиме холостого хода установление любого разрешенного прерывания вызывает возобновление нормальной работы
P5.4	ONCE#	Вход	Внутрисхемная эмуляция. Удержание низкого уровня на выводе во время нарастающего (положительного) фронта сигнала RESET# вызывает режим внутрисхемной эмуляции ONCE. Этот режим переводит все выходы, кроме BQ1 и BQ2, в состояние высокого импеданса, таким образом, изолируя МК от других компонентов в системе. Значение ONCE# захватывается, когда на вывод RESET# подается неактивный уровень. В то время как микроконтроллер находится в режиме ONCE, можно производить отладку системы. Для выхода из режима ONCE надо сбросить устройство низким уровнем сигнала RESET#. Чтобы предотвратить незапланированный вход в режим ONCE, можно сконфигурировать его как выход или подавать высокий уровень при выполнении сброса

Окончание таблицы 13.1

1	2	3	4
P2.6	Вход в тестовый режим	Вход/выход	Вход в тестовый режим. Если этот вывод удерживать в низком уровне в течение сброса (RESET#), устройство войдет в резервный тестовый режим, поэтому необходимо соблюдать меры предосторожности при использовании этого вывода как входа. Если необходимо сформировать этот вывод как вход, необходимо всегда удерживать высокий уровень на выводе в течение сброса для предотвращения случайного входа в тестовый режим
–	RESET#	Вход/выход	Сброс. Чувствительный к уровню вход сброса и выход с открытым стоком системного сброса микроконтроллера. Задний фронт RESET# и низкий уровень или внутренний сброс, включающий обнуляющий транзистор, связанный с выводом RESET#, удерживаются не менее 16-ти тактов. Для режимов POWERDOWN и IDLE выполнение сброса переводит МК в нормальный операционный режим. После сброса устройства, адрес первой инструкции 2080H
–	VPR	Питание	Напряжение программирования. В течение программирования на вывод VPR подается типовое напряжение (+12,5 В). Превышение максимума напряжения на выводе (+13,0 В) может повредить устройство. Вывод можно использовать для выхода из режима POWERDOWN. Используется этот метод для выхода из режима POWERDOWN только при использовании внешнего источника синхроимпульсов. На устройствах без внутреннего энергонезависимого запоминающего устройства надо соединить VPR с #VCC

Т а б л и ц а 13.2 – Регистры состояния и управления режимом

Мнемоника	Адрес	Описание
1	2	3
CCR0	2018 _H	Регистр конфигурации кристалла 0. Бит 0 этого регистра разрешает и запрещает режим POWERDOWN
INT_MASK1	0013 _H	Маска прерывания 1. Бит 6 этого 8-битного регистра разрешает и запрещает (маскирует) внешнее прерывание (EXINT)
P2_DIR P5_DIR P7_DIR	1FD2 _H 1FF3 _H 1FD3 _H	Управление порта <i>x</i> . Каждый бит P _x _DIR управляет конфигурацией соответствующего вывода. Очистка бита формирует вывод как комплементарный выход; установка формирует вывод как выход с открытым стоком или вход. (Выводы с открытым стоком требуют внешнего напряжения поддержки)
P2_MODE P5_MODE P7_MODE	1FD0 _H 1FF1 _H 1FD1 _H	Режим порта <i>x</i> . Каждый бит P _x _MODE управляет функцией соответствующего вывода как стандартного входа-выхода или как сигнала специальной функции. Установка бита формирует вывод как сигнал специальной функции; очистка бита формирует вывод как стандартный вход/выход

Окончание таблицы 13.2

1	2	3
P2_REG P5_REG P7_REG	1FD4 _H 1FF5 _H 1FD5 _H	<p>Порт_x. Вывод данных.</p> <p>Для входа устанавливают соответствующий P_x_REG бит. Для вывода записывают данные, которые будут выданы каждым выходом в соответствующий бит P_x_REG. Когда вывод формируется как стандартный вход/выход (P_x_MODE.y = 0), результат записи центрального процессора в P_x_REG немедленно появляется на выходе. Когда вывод формируется как сигнал специальной функции (P_x_MODE.y = 1), соответствующий периферийный блок МК или компонент вне МК управляет выводом. Центральный процессор может продолжать запись в P_x_REG, но вывод не меняет свое состояние, пока он не будет переключен назад к его стандартной функции ввода-вывода. Эта особенность позволяет программному обеспечению формировать вывод как стандартный вход/выход (очистить P_x_MODE.y), инициализировать или переписывать значение на выходе, затем формировать вывод как сигнал специальной функции (установить P_x_MODE.y). В этом случае инициализация, восстановление ошибки, обработка исключительных ситуаций и т. д. могут быть сделаны без изменения операций связанного периферийного устройства</p>

13.2 Ограничение потребления мощности

Оба режима сохранения мощности уменьшают потребляемую мощность, запрещая часть сигналов внутренней схемы синхроимпульсов (рисунок 13.1). В подразделах 13.3, 13.4 описываются оба режима подробно.

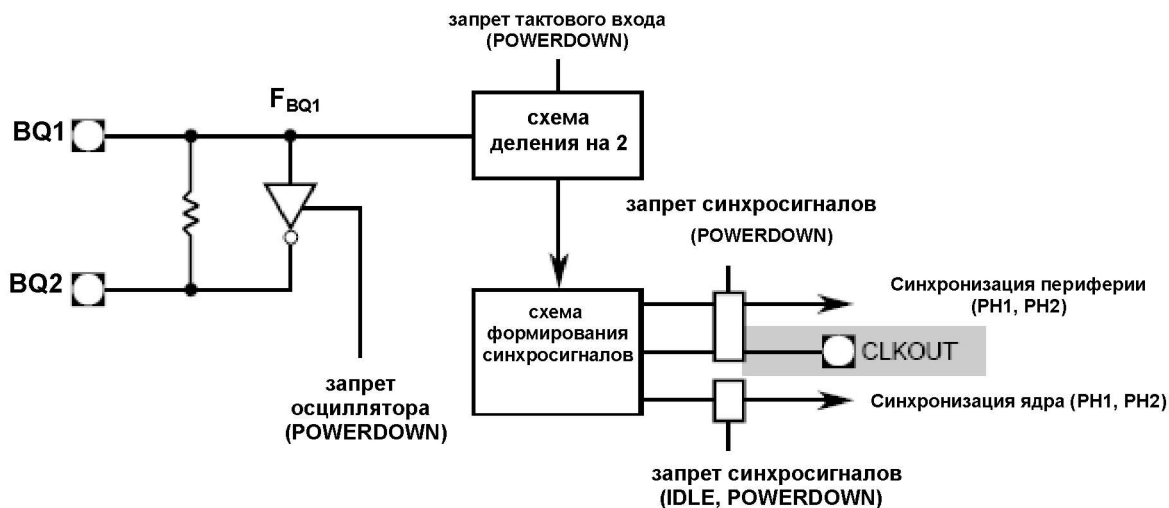


Рисунок 13.1 – Управление синхроимпульсами в течение режимов экономии мощности

13.3 Режим холостого хода IDLE

В режиме холостого хода потребление мощности устройства уменьшается приблизительно на 40 % от нормального потребления. Внутренняя логика удерживает сигналы синхронизации центрального процессора в логическом нуле, останавливая выполнение инструкций. Ни периферийные сигналы синхронизации, ни CLKOUT не

отключаются, так что регистры специальной функции SFR и регистры RAM сохраняют данные, периферийные устройства и система прерываний остаются активными. МК входит в режим холостого хода после выполнения инструкции IDLPD #1. Любой разрешенный источник прерывания, внутренний или внешний, или аппаратный сброс может вывести устройство из режима холостого хода. Когда происходит прерывание, сигналы синхронизации центрального процессора перезапускаются, и центральный процессор выполняет стандартную процедуру обслуживания прерывания или процедуру обслуживания прерывания PTS. Когда процедура завершена, конвейер центрального процессора восстанавливается и выполняется инструкция, которая следовала за IDLPD #1.

Примечание – Если разрешено, сторожевой таймер продолжает работать в режиме холостого хода. Устройство должно быть выведено из режима холостого хода прежде, чем счетчик переполнится. Иначе таймер сбросит МК. Интервал таймера – всегда 64К тактов.

Чтобы предотвратить случайное возвращение в рабочий режим, необходимо удерживать сигнал EXINT в состоянии низкого уровня, в то время как устройство находится в режиме холостого хода.

13.4 Режим низкого потребления POWERDOWN

Режим POWERDOWN переводит устройство в состояние очень низкого потребления мощности, выключая внутренний осциллятор и генератор синхроимпульсов. Внутренняя логика удерживает сигналы синхронизации периферии и центрального процессора в логическом нуле, заставляя центральный процессор прекращать выполнять инструкции, сигналы шинного управления системы становятся неактивными, CLKOUT устанавливается высоким уровнем, и периферийные устройства выключаются. Ток потребления устройства уменьшается до тока утечки. Если напряжение питания $U_{#VCC}$ не опускается ниже минимального, регистры специальных функций (SFR) и регистры RAM сохраняют свои данные.

Разрешение и запрещение режима POWERDOWN

Бит PD в регистре конфигурации кристалла 0 (CCR0.0) разрешает или запрещает режим POWERDOWN. Поскольку к CCR0 нельзя обратиться по команде, значение бита PD определено в байте конфигурации кристалла 0 (CCB0.0). Установка бита PD разрешает режим POWERDOWN, а очистка запрещает режим. CCR0 загружается из CCB0, когда устройство выполняет рестарт после сброса.

Вход в режим POWERDOWN

Перед входом в режим низкого потребления надо завершить следующие задачи:

1 Закончить все передачи или приемы последовательного порта. Иначе, когда устройство выходит из режима низкого потребления, работа последовательного порта продолжится с того места, где эта работа прервалась, и могут быть переданы или получены неправильные данные.

2 Закончить все аналоговые преобразования. Если режим низкого потребления начинается в течение преобразования, результат будет неправильным.

3 Если сторожевой таймер (WDT) разрешен, очистка WATCHDOG регистра производится перед инструкцией входа в режим низкого потребления. Это гарантирует, что устройство может выйти из режима низкого потребления корректно. Иначе, WDT может сбросить устройство прежде, чем осциллятор стабилизирует частоту. (WDT не может сбросить устройство в течение режима низкого потребления, потому что сброшены синхроимпульсы).

4 Все другие периферийные устройства перевести в неактивное состояние.

После завершения этих задач выполнить инструкцию IDLPD #2 для входа в режим низкого потребления.

Примечание – Чтобы предотвратить случайное возвращение в режим полной мощности, необходимо удерживать сигнал внешнего прерывания EXINT в состоянии низкого уровня, в то время как устройство находится в режиме низкого потребления.

Выход из режима низкого потребления POWERDOWN

Устройство выйдет из режима низкого потребления, когда произойдет любое из следующих событий:

- внешнее устройство на выводе VPR выставляет низкий уровень – не менее чем на 50 нс;
- произведен системный сброс (RESET);
- происходит переключение на выводе внешнего прерывания.

Задание низкого уровня на выводе VPR

Если устройство использует внешний тактовый сигнал, а не генерируемый на осцилляторе, то самый быстрый путь к выходу из режима низкого потребления – выставить низкий уровень на выводе VPR не менее чем на 50 нс. Используется этот метод только при использовании внешнего тактового сигнала, потому что синхронизация центрального процессора и периферийных устройств будет разрешена, а внутренний генератор будет заблокирован.

Выполнение аппаратного сброса (RESET)

Устройство выйдет из режима низкого потребления, если RESET# установлен. Если используется внешний тактовый сигнал, а не осциллятор на кристалле, RESET# должен остаться низким в течение, по крайней мере, 16 тактов. Если используется осциллятор на кристалле, то RESET# должен быть удержан в состоянии низкого уровня до тех пор, пока частота осциллятора не стабилизируется.

Установка сигнала внешнего прерывания (EXTINT)

Последний способ выхода из режима низкого потребления состоит в том, чтобы установить сигнал EXTINT не менее чем на 50 нс. Хотя EXTINT обычно простой вход, схема режима низкого потребления использует его, как чувствительный к уровню вход. Прерывание может не быть разрешено, но вывод должен формироваться как вход специальной функции. На рисунке 13.2 показана диаграмма последовательности включения и выключения режима низкого потребления при использовании вывода внешнего прерывания.

Когда внешнее прерывание выводит устройство из режима низкого потребления, соответствующий бит устанавливается в регистре ожидания прерывания. Если прерывание разрешено, устройство выполняет процедуру обслуживания прерывания, затем восстанавливает конвейер команд и выполняет инструкцию, следующую после инструкции IDLPD #2. Если прерывание запрещено (замаскировано), устройство восстанавливает конвейер команд и выполняет инструкцию, следующую после инструкции IDLPD #2, а бит ожидания прерывания остается установленным, пока не выполнится обслуживание прерывания или программное обеспечение не очистит бит ожидания прерывания.

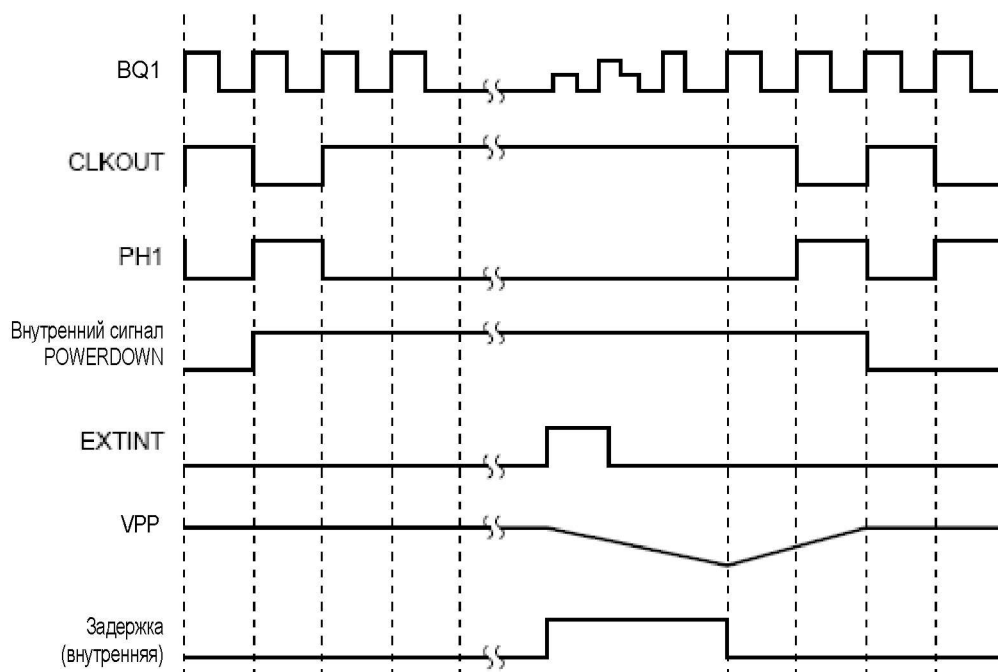


Рисунок 13.2 – Включение-выключение режима экономии мощности с использованием вывода внешнего прерывания

Когда используется сигнал внешнего прерывания для выхода из режима низкого потребления (POWERDOWN), рекомендуется использовать схему, показанную на рисунке 13.3 для вывода VPP. Разряд конденсатора вызывает задержку, которая позволяет осцилятору стабилизироваться прежде, чем будет разрешена синхронизация центрального процессора и периферийная синхронизация.

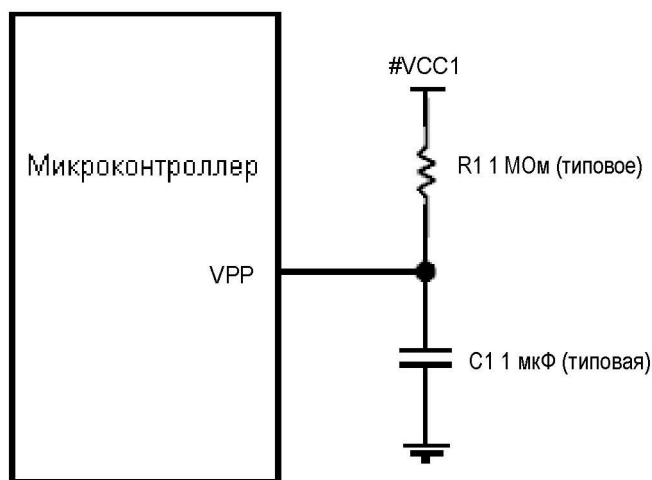


Рисунок 13.3 – Внешняя RC цепь на выводе VPP

В течение нормального режима работы (перед входом в режим POWERDOWN) напряжение на выводе VPP внутренне поддерживается до напряжения $U_{\#VCC}$. Когда устанавливается сигнал внешнего прерывания, работа генератора разрешена и включается слабое обнуление на VPP. Эта обнуляющая цепочка вызывает разряд внешнего конденсатора током порядка 200 мкА. При снижении напряжения вывода VPP ниже порогового напряжения (приблизительно 2,5 В), внутренняя синхронизация начинает работать и ЦП начинает выполнять коды команд.

В это время внутренний транзистор поддержки напряжения включается и быстро подтягивает высокий уровень до 3,5 В. После этого внутренняя поддержка становится неэффективной, и внешний резистор (R1) доводит напряжение до $U_{\#VCC}$ (смотри время восстановления на рисунке 13.4). Постоянная времени дает экспоненциально изменяющуюся кривую. Если $R1 = 1 \text{ МОм}$ и $C1 = 1 \text{ мкФ}$, время восстановления будет одна секунда.

Подбор R1 и C1

Значения R1 и C1 на рисунке 13.3 не являются критическими. Необходимо выбрать компоненты, которые обеспечат достаточное время разряда, чтобы разрешить внутренней схеме генератора стабилизироваться. Поскольку многие факторы могут влиять на требования ко времени разряда, необходимо всегда определять время разряда для наихудших условий работы.

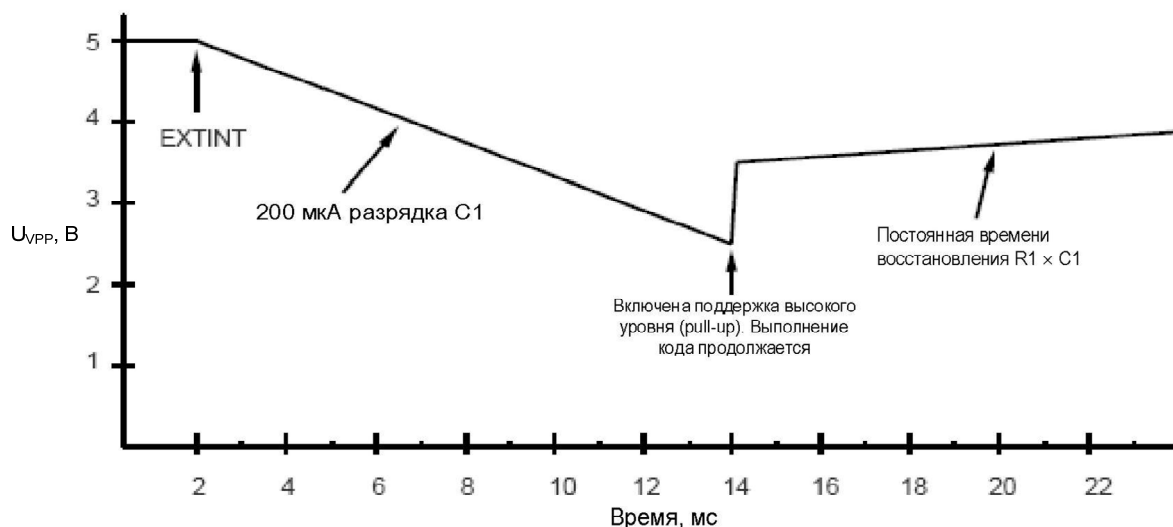


Рисунок 13.4 – Напряжение на выводе VPR при выходе из режима POWERDOWN

Необходимо выбрать резистор, который будет влиять на ток разряда. В большинстве случаев значения его сопротивления – между 200 кОм и 1 МОм.

При подборе конденсатора надо определить наихудшее время разряда, необходимое для генератора, чтобы стабилизировать частоту, затем использовать формулу (13.1), чтобы вычислить соответствующее значение для C1.

$$C1 = \frac{T_{DIS} \times I}{V_T}, \quad (13.1)$$

где C1 – номинал конденсатора в фарадах;
 T_{DIS} – худшее время разряда в секундах;
 I – ток разряда в амперах;
 V_T – пороговое напряжение в вольтах.

Примечание – Если устройство повторно вошло в POWERDOWN и вышло, прежде чем C1 зарядился до $U_{\#VCC}$, будет требоваться меньше времени для уменьшения напряжения до порога. Поэтому устройству понадобится меньше времени для выхода из POWERDOWN.

Например, предположим, что генератору необходимо по крайней мере 12,5 мс для разряда ($T_{DIS} = 12,5 \text{ мс}$), $V_T = 2,5 \text{ В}$ и ток разряда $I = 200 \text{ мкА}$. Номинал C1 около 1 мкФ:

$$C1 = \frac{(0,0125)(0,0002)}{2,5} = 1 \text{ мкФ}. \quad (13.2)$$

При использовании внешнего генератора значение C1 может быть очень маленьким, позволяя быстрый выход из POWERDOWN. Например, емкость в 100 пФ дает время разрядки 1,25 мкс.

$$T_{DIS} = \frac{C1 \times V_T}{I} = \frac{(1,0 \times 10^{-10})(2,5)}{0,0002} = 1,25 \text{ мкс} . \quad (13.3)$$

13.5 Режим внутрисхемной эмуляции ONCE

Режим внутрисхемной эмуляции ONCE изолирует устройство от других компонентов в системе, чтобы позволить проверять печатные платы или проводить отладку с поставочным эмулятором. В течение режима ONCE все выходы кроме BQ1, BQ2, #0V и #VCC имеют слабую поддержку до высокого или до низкого уровня. В течение режима ONCE вывод RESET# должен оставаться в состоянии высокого уровня, иначе устройство выйдет из режима ONCE и войдет в состояние сброса.

Удержание сигнала ONCE# в состоянии низкого уровня в течение положительного фронта сигнала RESET# вводит МК в режим ONCE. Чтобы предотвратить случайный вход в режим ONCE, рекомендуется конфигурирование этого вывода как выхода. Если необходимо сформировать этот вывод как вход, надо на нем держать высокий уровень в течение сброса, чтобы предотвратить случайный вход в режим ONCE.

Выход из режима ONCE происходит при установке сигнала RESET# и при плавающем уровне или слабой поддержке до высокого уровня на выводе ONCE#. Нормальная работа возобновится, когда сигнал RESET# перейдет в высокий уровень.

13.6 Зарезервированные тестовые режимы

Специальный вывод P2.6 предназначен для тестового режима, который недоступен потребителю. Чтобы предотвратить случайный вход в тестовый режим, рекомендуется конфигурирование вывода P2.6 как выхода. Если надо сформировать этот вывод как вход, необходимо держать этот вывод в состоянии высокого уровня во время сброса.

14 Интерфейс внешней памяти

Микроконтроллер 1874BE66T осуществляют поддержку интерфейса с различными типами устройств внешней памяти. Реализована программно-аппаратная поддержка конфигурации шины данных: 8-разрядная фиксированная, 16-разрядная фиксированная, 8/16-разрядная динамически переключаемая; внутреннее управление длительностью циклов обращения к «медленной» внешней памяти; несколько режимов управления шиной адресов/данных.

В разделе описаны сигналы интерфейса внешней памяти, регистры управления интерфейса, байты конфигурации и режимы работы внешней шины, временные диаграммы работы микросхемы.

14.1 Сигналы и регистры интерфейса внешней памяти

В таблице 14.1 приведены сигналы интерфейса внешней памяти, в таблице 14.2 приведены регистры интерфейса, в таблице 14.3 – установка регистров порта 5 для конфигурации сигналов интерфейса внешней памяти.

Т а б л и ц а 14.1 – Сигналы интерфейса внешней памяти

Обозначение сигнала	Вывод порта	Тип сигнала	Описание
1	2	3	4
AD15 – AD0	P4.7–P4.0, P3.7–P3.0	Вход/выход	Шина адресов/данных. Выводы конфигурируют мультиплексированную шину адресов/данных. В течение адресной фазы шинного цикла биты 0 – 15 адреса выводятся в шину и могут быть зафиксированы внешним интерфейсом сигналами ALE или ADV#. В течение фазы данных происходит передача 8- или 16-разрядных данных
ADV#	P5.0	Выход	Адрес действителен. Активный сигнал низкого уровня устанавливается только в течение доступа к внешней памяти. ADV# указывает, что адресная информация выведена в шину адресов/данных. Сигнал сохраняется в низком уровне в продолжении шинного цикла и переходит в высокий уровень после его завершения. Внешний интерфейс может использовать его для формирования сигналов CS («выбор кристалла»), используемых микросхемами памяти
ALE	P5.0	Выход	Разрешение записи адреса. Активный высокий уровень сигнала устанавливается в течение цикла обращения к внешней памяти. Сигналом ALE начинается внешний цикл шины и его активный уровень указывает на наличие адресной информации во внешней шине адресов/данных. Отличие ALE от ADV# в том, что его активный уровень не поддерживается до окончания цикла шины. Внешний интерфейс может использовать этот сигнал для демultipлексирования адреса из шины адресов/данных

Продолжение таблицы 14.1

1	2	3	4																				
ВНЕ#	P5.5	Выход	<p>Выбор старшего байта.</p> <p>В течение 16-разрядного цикла шины активный низкий уровень сигнала устанавливается для чтения или записи слова и старшего байта. ВНЕ# указывает, что действительна передача данных в старшем байте шины. Дешифрация ВНЕ# совместно с AD0 определяет, какой конкретно байт слова передается по шине:</p> <table> <tr> <td>ВНЕ#</td> <td>AD0</td> <td>Доступность байтов</td> </tr> <tr> <td>0</td> <td>0</td> <td>оба</td> </tr> <tr> <td>0</td> <td>1</td> <td>старший</td> </tr> <tr> <td>1</td> <td>0</td> <td>младший</td> </tr> </table> <p>Выбор ВНЕ# задается битом CCR0.2 = 1 в регистре конфигурации 0</p>	ВНЕ#	AD0	Доступность байтов	0	0	оба	0	1	старший	1	0	младший								
ВНЕ#	AD0	Доступность байтов																					
0	0	оба																					
0	1	старший																					
1	0	младший																					
BW	P5.7	Вход	<p>Разрядность шины данных.</p> <p>Регистры конфигурации 0 и 1 совместно с BW управляют шириной шины данных.</p> <p>CCR0.1 CCR1.2 BW</p> <table> <tr> <td></td> <td></td> <td></td> <td>фиксированная:</td> </tr> <tr> <td>0</td> <td>1</td> <td>X</td> <td>8-разрядная шина данных</td> </tr> <tr> <td>1</td> <td>0</td> <td>X</td> <td>16-разрядная шина данных, управляемая по BW</td> </tr> <tr> <td>1</td> <td>1</td> <td>H</td> <td>16-разрядная шина данных</td> </tr> <tr> <td>1</td> <td>1</td> <td>L</td> <td>8-разрядная шина данных</td> </tr> </table>				фиксированная:	0	1	X	8-разрядная шина данных	1	0	X	16-разрядная шина данных, управляемая по BW	1	1	H	16-разрядная шина данных	1	1	L	8-разрядная шина данных
			фиксированная:																				
0	1	X	8-разрядная шина данных																				
1	0	X	16-разрядная шина данных, управляемая по BW																				
1	1	H	16-разрядная шина данных																				
1	1	L	8-разрядная шина данных																				
CLKOUT	–	Выход	<p>Системный тактовый сигнал.</p> <p>Выход внутреннего тактового генератора. Частота F_{CO} равна 1/2 входной частоты F_{BQ1}. Скважность сигнала равна 50 %</p>																				
EA#	–	Вход	<p>Обращение к внешней памяти.</p> <p>Доступ к внутренней памяти программ разрешен установкой EA# в высокий уровень. Доступ к внешней памяти разрешен установкой EA# в низкий уровень. Если $U_{\#EA}$ устанавливается напряжением, равным U_{VPR} (типовое плюс 12,5 В) на фронте нарастания RESET#, микроконтроллер с EPROM входит в режим программирования. Дальнейшее изменение уровня напряжения на входе EA# не оказывает влияние на режим. Если МК не имеет встроенного EPROM, необходимо подключить вывод EA# к #VCC</p>																				
INST	P5.1	Выход	<p>Чтение команды.</p> <p>Активный высокий уровень сигнала указывает, что команда считана в МК из внешней памяти. Сигнал сохраняется в высоком уровне до окончания цикла чтения команды. При обращении к данным сигнал устанавливается в низкий уровень, включая чтение векторов прерываний и байтов конфигурации. Также сигнал всегда в низком уровне при чтении внутренней памяти</p>																				

Окончание таблицы 14.1

1	2	3	4
RD#	P5.3	Выход	Чтение. Устанавливается активным уровнем только в цикле чтения внешней памяти
READY	P5.6	Вход	Готовность. Активный сигнал совместно с регистрами конфигурации определяет число тактов задержки в шинном цикле. Регистры конфигурации задают число тактов задержки (0, 1, 2, 3, ..., бесконечность). При низком READY в шинный цикл вводится запрограммированная задержка. Если READY переключается в высокий уровень раньше, чем достигается запрограммированное число тактов задержки, «затяжка» цикла прекращается
WR#	P5.2	Выход	Запись. Активный низкий уровень указывает на запись во внешнюю память. Используется только в циклах записи. Выбор WR# задается битом CCR0.2 = 1
WRH#	P5.5	Выход	Запись старшего байта. В течение 16-разрядного машинного цикла активный низкий уровень устанавливается для записи старшего байта или слова во внешнюю память. В 8-разрядном цикле устанавливается во всех операциях записи. Выбор WRH# задается битом CCR0.2 = 0
WRL#	P5.2	Выход	Запись младшего байта. В течение 16-разрядного машинного цикла активный низкий уровень устанавливается для записи младшего байта или слова во внешнюю память. В 8-разрядном цикле устанавливается во всех операциях записи. Выбор WRL# задается битом CCR0.2 = 0

Т а б л и ц а 14.2 – Регистры интерфейса внешней памяти

Обозначение	Адрес	Описание
1	2	3
CCR0	2018 _H	Конфигурация кристалла 0. Управляет режимом пониженного потребления мощности, сигналами интерфейса шины, защитой внутренней памяти. Три бита регистра совместно с двумя битами CCR1 управляют «затяжкой» шинного цикла и разрядностью шины данных
CCR1	201A _H	Конфигурация кристалла 1. Разрешает сторожевой таймер, выбирает временной режим работы шины. Два бита совместно с тремя битами CCR0 управляют «затяжкой» шинного цикла и разрядностью шины
P5_DIR	1FF3 _H	Управление порта 5. Каждый очищенный бит конфигурирует соответствующий вывод как комплементарный выход, установленный бит – как вход или выход с открытым стоком

Окончание таблицы 14.2

1	2	3
P5_MODE	1FF1 _H	Режим порта 5. Каждый установленный бит конфигурирует соответствующий вывод как сигнал специальной функции, очищенный бит – как стандартный вход/выход
P5_PIN	1FF7 _H	Входы порта 5. Каждый бит регистра отражает текущее состояние соответствующего вывода, невзирая на конфигурацию
P5_REG	1FF5 _H	Выходы порта 5. При выводе данных записанные значения разрядов P5_REG выдаются на соответствующие выводы порта. При конфигурации выводов как стандартные входы-выходы (P5_MODE.y = 0) записанные данные появляются на выходах непосредственно. При конфигурации выводов как сигналов специальных функций (P5_MODE.y = 1) управление выводами осуществляется соответствующей внутренней или внешней периферией ИС 1874BE66T. Записываемые в P5_REG значения не будут влиять на состояние выводов, пока они не переключатся на функции стандартных входов/выходов. Это позволяет программному обеспечению конфигурировать выводы как стандартные, инициализировать или изменять значения на выводах, затем конфигурировать их как сигналы специальных функций, что позволяет производить коррекцию ошибок, обработку исключительных ситуаций и т. д., не нарушая режим работы периферийных устройств

Т а б л и ц а 14.3 – Установка регистров для конфигурации сигналов интерфейса внешней памяти

Вывод порта	Обозначение сигнала интерфейса	Тип сигнала	Значение установки регистров
P5.0	ALE/ADV#	выход	P5_DIR = 110X0000 _B
P5.1	INST	выход	P5_MODE = 111X 1111 _B
P5.2	WR#/WRL#*	выход	P5_REG = 11XX XXXX _B
P5.3	RD#	выход	
P5.5	BHE#/WRH#*	выход	
P5.6	READY	вход	
P5.7	BW	вход	

* Установка в регистре конфигурации CCR0 бита CCR0.2 = 1 выбирает функции BHE# и WR#, а CCR0.2 = 0 выбирает функции WRL# и WRH#.

14.2 Регистры и байты конфигурации кристалла

Настройка параметров операций МК и внешних шинных циклов осуществляется двумя регистрами конфигурации. Регистры программно недоступны и их загрузка осуществляется двумя байтами конфигурации, размещаемыми во внутренней энергонезависимой или внешней памяти по адресам 2018_H (CCB0) и 201A_H (CCB1). Загрузка осуществляется при выполнении микроконтроллером выхода из сброса (каждый раз). Значения разрядов регистров приведены на рисунках 14.1, 14.2, в таблицах 14.4, 14.5.

Регистр CCR0

Нет прямого доступа.

Регистр конфигурации 0 (CCR0) управляет режимом пониженного потребления мощности, сигналами управления шины, защитой встроенной памяти, три разряда в комбинации с двумя разрядами регистра CCR1 управляют «затягиванием» шинного цикла и разрядностью шины данных.

7	0						
LOC1	LOC0	IRC1	IRC0	ALE	WR	BW0	PD

Рисунок 14.1 – Регистр конфигурации кристалла 0 (CCR0)

Т а б л и ц а 14.4 – Разряды регистра конфигурации кристалла 0 (CCR0)

Номер бита	Мнемоника	Функция
1	2	3
7, 6	LOC1, LOC0	Биты защиты. Управляют доступом чтения или записи EPROM в режиме нормальных операций LOC1 LOC0 Доступ EPROM 0 0 запрет чтения и записи 0 1 запрет чтения 1 0 запрет записи 1 1 нет защиты
5, 4	IRC1, IRC0	Управление «затягиванием» шинного цикла. Два бита совместно с IRC1.1 (CCR1) и READY управляют «затягиванием» шинного цикла. Если READY удерживается в низком уровне, цикл шины «затягивается» в соответствии с запрограммированным числом тактов (машинных). Если READY переключается в высокий уровень раньше, чем достигнуто запрограммированное число тактов, «затягивание» цикла прекращается. При выборе варианта неопределённого времени «затягивания» P5.6 конфигурируется как сигнал READY. Рекомендуется дополнить аппаратное обеспечение счётчиком необходимого числа тактов «затягивания» и установки READY высоким уровнем, т. к. дефектная ИМС памяти может занимать шину неопределённое время. IRC2 IRC1 IRC0 «Затягивание» времени цикла шины 0 0 0 нет 0 X 1 запрещённые комбинации 0 1 X комбинации 1 0 0 1 такт 1 0 1 2 такта 1 1 0 3 такта 1 1 1 определяется сигналом READY

Окончание таблицы 14.4

1	2	3
3	ALE	Разрешение записи адреса и запись данных. Определяют комбинацию сигналов управления шиной при внешних циклах чтения и записи. ALE WR режимы 0 0 стробирование действительного адреса и записи (ADV#, RD#, WRL#, WRH#)
2	WR	0 1 стробирование действительного адреса (ADV#, RD#, WR#, BHE#) 1 0 стробирование записи (ADV#, RD#, WRL#, WRH#) 1 1 стандартный (ALE, RD#, WR#, BHE#)
1	BW0	Управление разрядностью шины. Осуществляется совместно с BW1 (CCR1.2) BW1 BW0 разрядность 0 0 запрещённая комбинация 0 1 только 16-разрядная 1 0 только 8-разрядная 1 1 управление по BW
0	PD	Выбор режима пониженного потребления мощности. Разрешает команде IDLPD #2 перевести МК в режим пониженного потребления мощности. PD = 0 – запрет пониженного потребления мощности; PD = 1 – разрешение пониженного потребления мощности

Регистр CCR1

Нет прямого доступа.

Регистр конфигурации 1 (CCR1) разрешает сторожевой таймер, выбирает временной режим шины, совместно с CCR1 управляет «затягиванием» шинного цикла и разрядностью шины данных.

7	1	1	0	1	WDE	BW1	IRC2	0
---	---	---	---	---	-----	-----	------	---

Рисунок 14.2 – Регистр конфигурации кристалла 1 (CCR1)

Т а б л и ц а 14.5 – Разряды регистра конфигурации кристалла 1 (CCR1)

Номер бита	Мнемоника	Функция
7, 6	1	Записываются «1»
5	0	Записывается «0»
4	1	Записывается «1»
3	WDE	Разрешение сторожевого таймера. 0 – разрешён всегда, 1 – запрещён до очистки регистра WATCHDOG
2	BW1	Управление разрядностью шины. Осуществляется совместно с BW0 (CCR0.1) BW1 BW0 разрядность 0 0 запрещённая комбинация 0 1 только 16-разрядная 1 0 только 8-разрядная 1 1 управление по BW
1	IRC2	Управление затягиванием шинного цикла. Бит совместно с IRC0, IRC1 (CCR0.4, CCR0.5) и READY управляют затягиванием шинного цикла. Если READY удерживается в низком уровне, цикл шины затягивается в соответствии с запрограммированным числом тактов (машинных). Если READY переключается в высокий уровень раньше, чем достигнуто запрограммированное число тактов, затягивание цикла прекращается. При выборе варианта неопределённого времени затягивания P5.6 конфигурируется как сигнал READY. Рекомендуется дополнить аппаратное обеспечение счётчиком необходимого числа тактов «затягивания» и установки READY высоким уровнем, т. к. дефектная ИС памяти может занимать шину неопределённое время. IRC2 IRC1 IRC0 «Затягивание» времени цикла шины 0 0 0 нет 0 X 1 запрещённые 0 1 X комбинации 1 0 0 1 такт 1 0 1 2 такта 1 1 0 3 такта 1 1 1 определяется сигналом READY
0	0	Записывается «0»

14.3 Мультиплексирование и разрядность шины

Внешняя шина имеет два режима работы: 16-разрядная мультиплексированная шина адресов/данных, мультиплексированная 16-разрядная шина адреса/8-разрядная шина данных (рисунок 14.3).

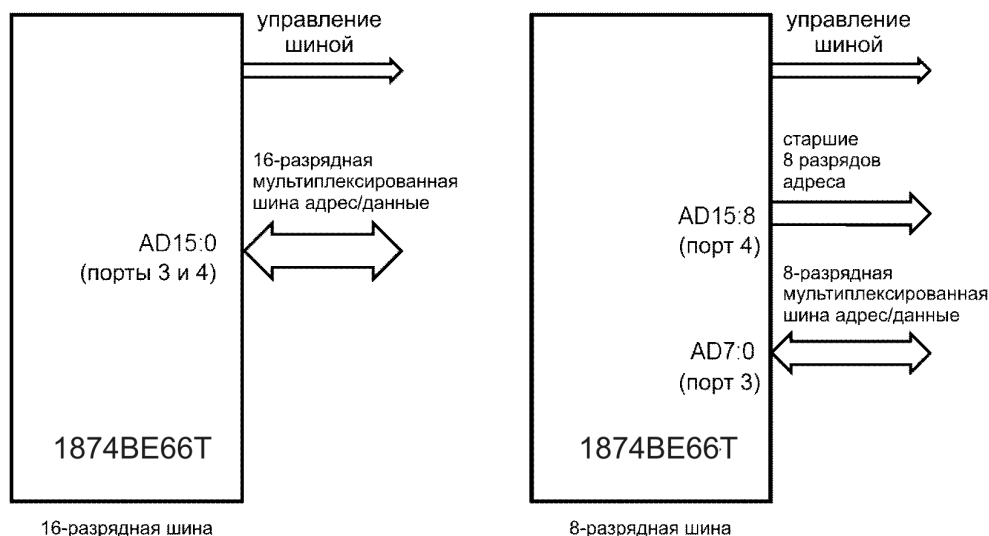


Рисунок 14.3 – Конфигурация шины адресов/данных

После сброса, но до записи ССВ, микроконтроллер конфигурируется в 8-разрядный режим шины независимо от состояния входа BW. Старшие линии шины (AD15–8) управляются маломощным выходным формирователем сигналов в течение циклов ССВ0 и ССВ1. Для предотвращения конфликтных ситуаций в шине не следует подключать к этим линиям нагрузки по #VCC и #0V. Старшие байты слов ССВ (ячейки 2019_H и 201B_H) должны быть загружены величиной 20_H. При чтении из памяти значения 20_H в старшем байте конфликтов в шине не произойдет. После загрузки ССВ в ССР значения BW0 и BW1 будут определять конфигурацию шины (рисунки 14.1, 14.2).

При установлении BW0 и BW1 разрядностью шины управляет сигнал BW: высокий уровень сигнала выбирает 16-разрядную шину, низкий уровень выбирает 8-разрядную шину. Сигнал BW действителен после появления адреса в шине (рисунок 14.4)

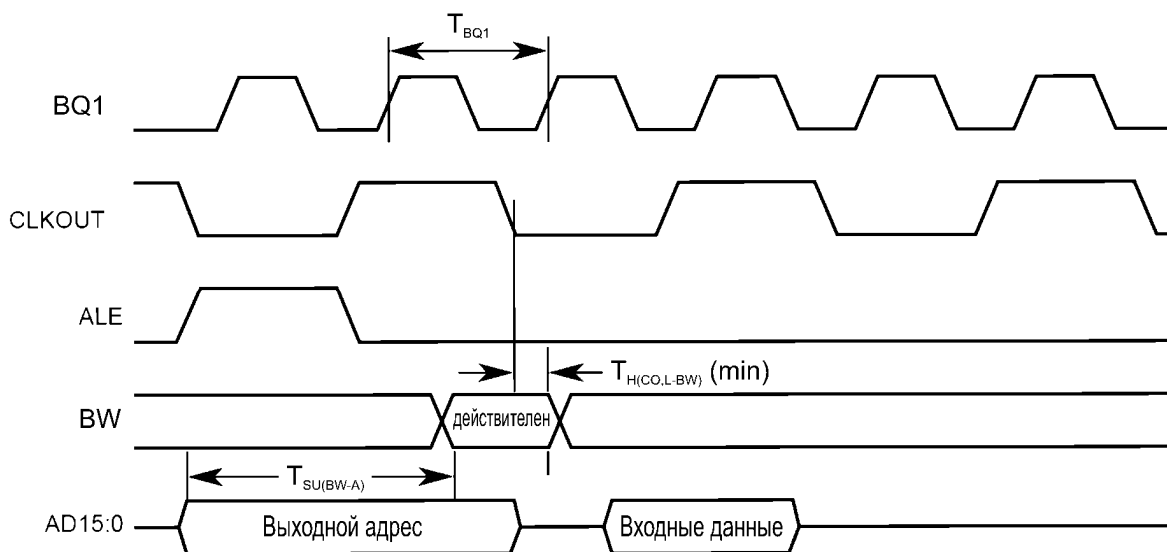


Рисунок 14.4 – Временная диаграмма сигнала BW

Т а б л и ц а 14.6 – Обозначения временных параметров сигнала BW

Обозначение	Определение
$T_{SU(BW-A)}$	Время установки сигналов адреса относительно сигнала BW
$T_{H(CO,L-BW)}$	Время сохранения сигнала BW относительно низкого уровня сигнала CO (минимальное)
T_{BQ1}	$1/F_{BQ1}$. Период следования импульсов входного тактового сигнала

Временные диаграммы 16-разрядной шины

На рисунке 14.5 приведены идеализированные временные диаграммы сигналов внешней шины в 16-разрядной конфигурации.

Передний фронт сигнала ALE устанавливается перед выдачей микроконтроллером адреса в шину (AD15–0). МК удерживает адрес до переключения ALE в низкий уровень (минимум). Сигнал ALE используется схемами «защёлки» адреса для его записи из шины и хранения до вывода данных в шину.

В 16-разрядном цикле чтения микроконтроллер устанавливает шину в третье состояние и выдает низкий уровень сигнала RD# для чтения данных. Внешняя память должна вывести данные в шину раньше переключения RD# в высокий уровень. При установке микроконтроллером сигнала INST производится чтение команд.

В 16-разрядном цикле записи микроконтроллер устанавливает низкий уровень сигнала WR# и выводит данные в шину. Задний фронт сигнала индицирует действительность данных. В этот момент внешняя система должна «защёлкнуть» данные.

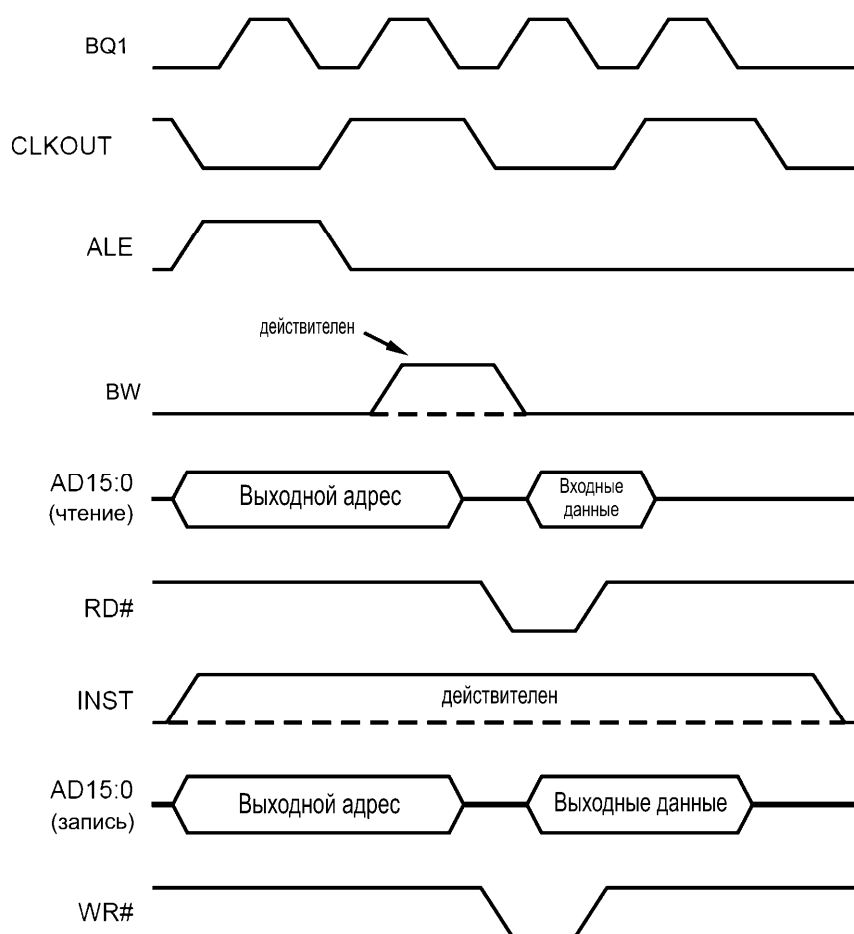


Рисунок 14.5 – Временные диаграммы сигналов 16-разрядной шины

Временные диаграммы 8-разрядной шины

В режиме 8-разрядной шины выходы AD7 – AD0 формируют мультиплексированную шину младшего байта адреса/данных. Выводы AD15 – AD8 не мультиплексированы; в течение цикла на них устанавливается и удерживается старший байт адреса. На рисунке 14.6 приведены идеализированные временные диаграммы сигналов внешней шины в 8-разрядной конфигурации.

Для чтения или записи байта требуется один цикл шины, для слова – два цикла. Первый цикл – доступ к младшему байту, второй цикл – к старшему байту. Временные параметры 16- и 8-разрядных циклов идентичны. ALE используется для демultipлексирования младшего байта адреса.

В цикле 8-разрядного чтения слова сигнал RD# устанавливается дважды: сначала читается младший байт, затем – старший. В цикле 8-разрядной записи слова сигнал WR# также устанавливается дважды: сначала из микроконтроллера выводится младший байт, затем – старший.

Входные и выходные данные должны быть стабильны во время переключения сигналов RD# и WR# соответственно из низкого уровня в высокий уровень.

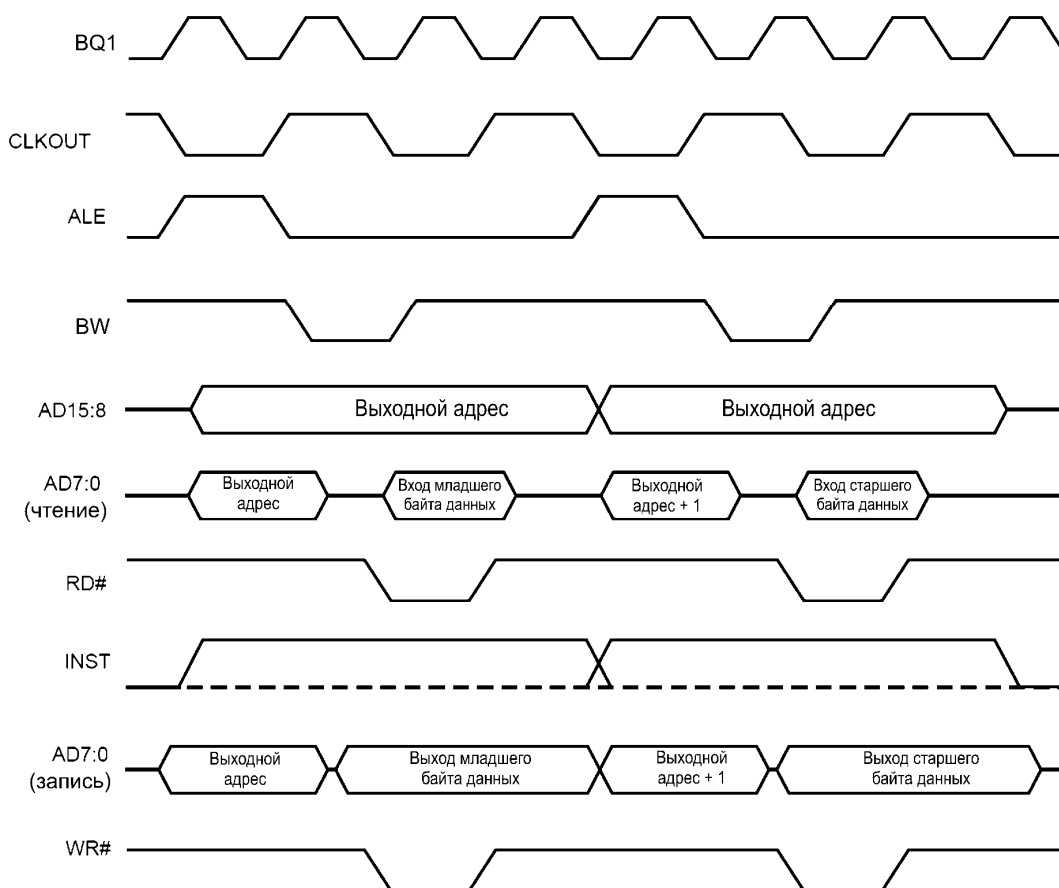


Рисунок 14.6 – Временные диаграммы сигналов 8-разрядной шины

14.4 Управление длительностью шинного цикла сигналом READY

Внешние устройства могут использовать вход READY для управления длительностью шинного цикла. Когда адрес выведен в шину, а устройство внешней памяти не готово для доступа, оно должно установить сигнал READY низким уровнем и удерживать его до готовности завершить операцию. В ответ микроконтроллер увеличивает длительность цикла, добавляя состояние ожидания ($2T_{BQ1}$) до тех пор, пока READY не переключится в высокий уровень. После сброса и до приёма CCB1 микроконтроллер всегда добавляет три состояния ожидания в шинный цикл. Пока P5.6 сконфигурирован как вход READY, биты IRC2:0 регистров конфигурации управляют

длительностью цикла (таблица 14.5, таблица 14.6), а при установке неопределённой длительности ($IRC2 - 0 = 111_B$) – управление осуществляется внешним аппаратным обеспечением (сигналом READY). На рисунке 14.7 приведены временные диаграммы для сигнала READY с одним добавочным состоянием ожидания. В таблице 14.7 приведены определения временных параметров сигнала READY.

Т а б л и ц а 14.7 – Обозначения временных параметров сигнала READY

Обозначение	Определение
$T_{SU(RDY,L-A)}$	Время установки сигнала адреса относительно установления сигнала READY в низкий уровень
$T_{H(CO,L-RDY,L)}$	Время удержания сигнала READY относительно системного тактового сигнала (минимальное, максимальное)

Параметр $T_{SU(RDY,L-A)}$ определяет время, необходимое внешней аппаратуре для дешифрации адреса и формирования низкого уровня сигнала READY. Параметр $T_{H(CO,L-RDY,L)}$ определяет минимально необходимое время фиксации микроконтроллером уровня сигнала READY. Норма на максимальное значение параметра не устанавливается.

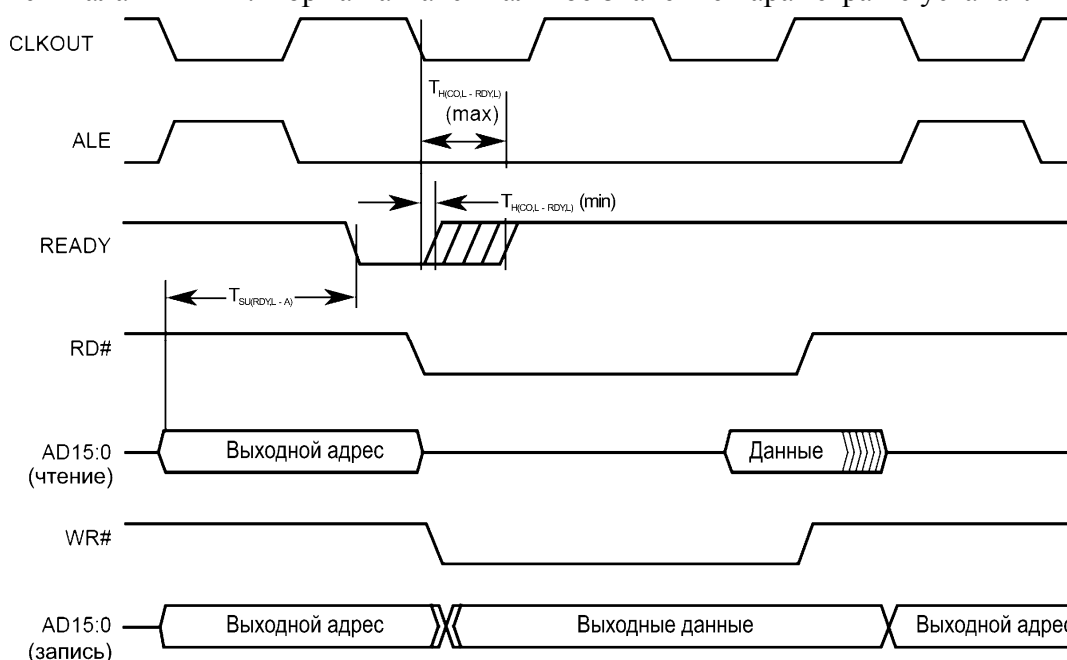


Рисунок 14.7 – Временные диаграммы сигнала READY

14.5 Режимы управления шиной

Биты ALE и WR (CCR0.3 и CCR0.2) определяют комбинацию сигналов управления шиной, формируемую во внешних циклах чтения и записи. В таблице 14.8 приведены устанавливаемые значения битов CCR0.3 и CCR0.2 для четырёх возможных режимов управления шиной.

Т а б л и ц а 14.8 – Режимы управления шиной

Режим	Сигналы управления	CCR0.3 (ALE)	CCR0.2 (WR)
стандартный	ALE, RD#, WR#, BHE#	1	1
стробирование записи	ALE, RD#, WRL#, WRH#	1	0
стробирование действительного адреса	ADV#, RD#, WR#, BHE#	0	1
стробирование действительного адреса и записи	ADV#, RD#, WRL#, WRH#	0	0

Стандартный режим управления шиной

В стандартном режиме управления шиной микроконтроллер формирует набор сигналов, приведённый в таблице 14.8 и на рисунке 14.8. ALE устанавливается при выводе адреса и может использоваться для фиксации адреса внешними схемами. RD# устанавливается в циклах чтения внешней памяти, WR# – в циклах записи. При установлении BHE# выбирается банк памяти, адресуемый старшим байтом шины данных.

В конфигурации 16-разрядной шины микроконтроллер может формировать отдельные сигналы записи младших и старших байтов для байтовых операций записи. На рисунке 14.9 приведена упрощённая схема дешифрации сигналов WRH# и WRL#. Для циклов чтения подобная пара сигналов не нужна, т. к. при чтении байта из 16-разрядной шины микроконтроллер игнорирует незатребованный байт.

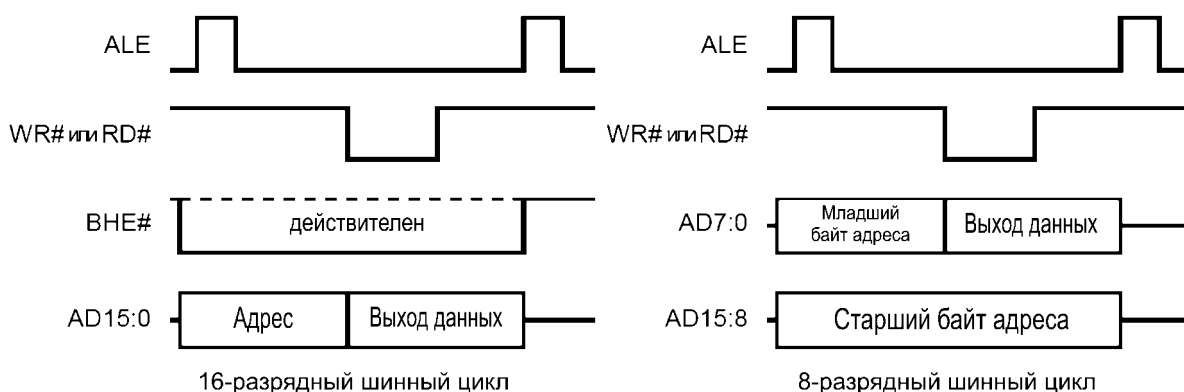


Рисунок 14.8 – Стандартный режим управления шиной

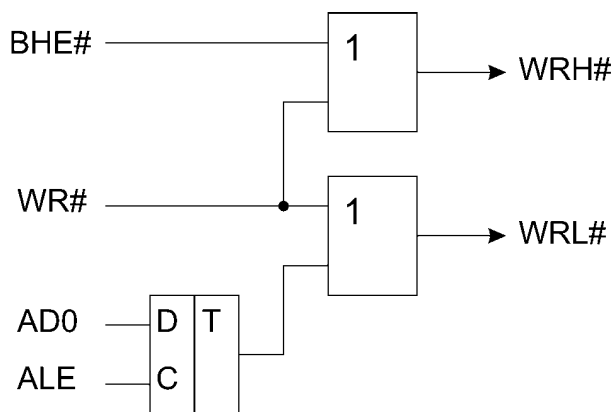


Рисунок 14.9 – Схема дешифрации сигналов WRH# и WRL#

На рисунке 14.10 приведён вариант 8-разрядной системы, использующий схемы ЭППЗУ (типа «флэш») и ОЗУ для младшего и старшего блоков памяти. Старший разряд адреса A15 используется как сигнал «выбор кристалла», сигнал ALE для фиксации адреса.

Режим стробирования записи

Режим стробирования записи устраняет необходимость внешнего декодирования записи старшего и младшего байтов во внешние 16-разрядное ОЗУ или флеш-память в режиме 16-разрядного шинного цикла.

При выборе режима стробирования записи микроконтроллер формирует вместо сигналов WR# и WE# сигналы WRL# и WRH# (внешний дешифратор по типу, приведённому на рисунке 14.9, не используется). WRL# устанавливается для записи младших байтов (чётные адреса) и слов. WRH# устанавливается для записи старших байтов (нечётные адреса) и слов. В 8-разрядном режиме шины сигналы WRL# и WRH# устанавливаются и для чётных и для нечётных адресов. На рисунке 14.12 приведены временные диаграммы сигналов режима стробирования записи.

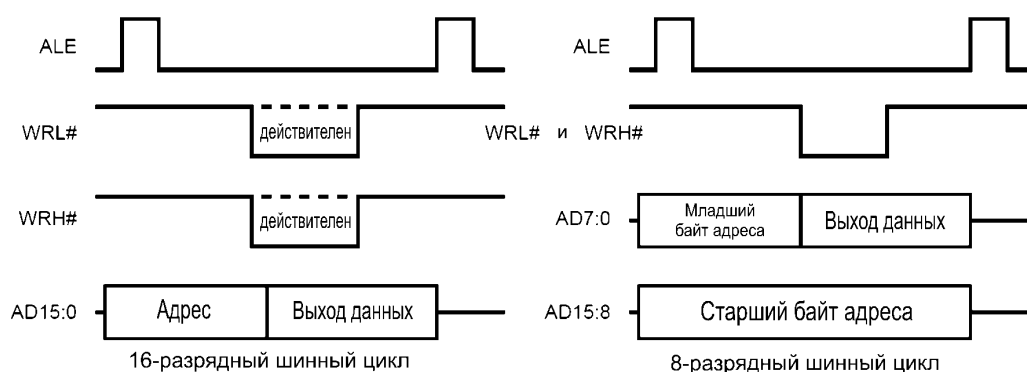


Рисунок 14.12 – Режим стробирования записи

На рисунке 14.13 приведён вариант 16-разрядной системы с двумя устройствами ЭПЗУ и двумя устройствами ОЗУ, сконфигурированными для использования режима стробирования записи. Сигнал ALE фиксирует адрес. Сигнал AD15 является сигналом «выбор кристалла» для микросхем памяти. WRL# устанавливается для записи младшего байта и слова, WRH# – для записи старшего байта и слова. A0 не используется (нет внешней дешифрации).

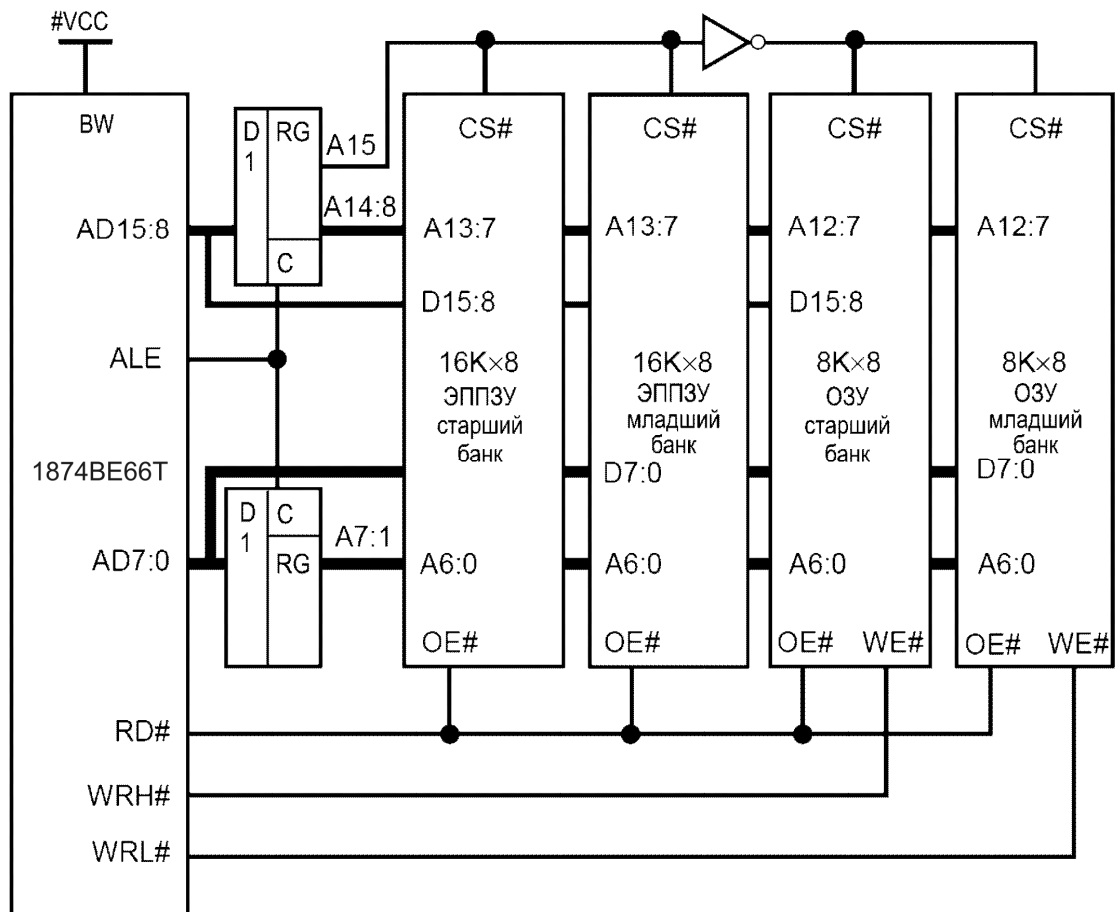


Рисунок 14.13 – 16-разрядная система с записью в 8-разрядное ОЗУ

Режим стробирования действительного адреса

При выборе режима стробирования действительного адреса микроконтроллер формирует сигнал $ADV\#$ вместо ALE . Сигнал $ADV\#$ устанавливается после вывода адреса в шину. Сигнал может использоваться для фиксации адреса и разрешать работу микросхем внешней памяти. Разница между сигналами ALE и $ADV\#$ в том, что $ADV\#$ устанавливается на полный цикл шины, а не только для фиксации адреса. Временные диаграммы режима стробирования действительного адреса приведены на рисунке 14.14, а на рисунке 14.15 показана разница временных диаграмм ALE и $ADV\#$ для одного цикла чтения или записи.

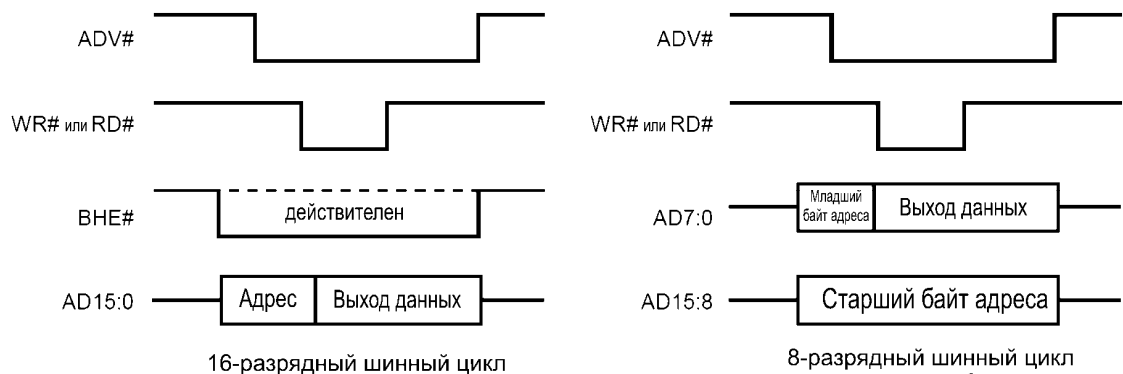


Рисунок 14.14 – Режим стробирования действительного адреса

Если циклы шины следуют один за другим, действие ADV# будет выглядеть идентично действию ALE. Различие заметно при наличии интервалов холостого хода шины. Т. к. ADV# поддерживается в этих интервалах в высоком уровне, внешняя память запрещена и снижается потребление мощности в системе.

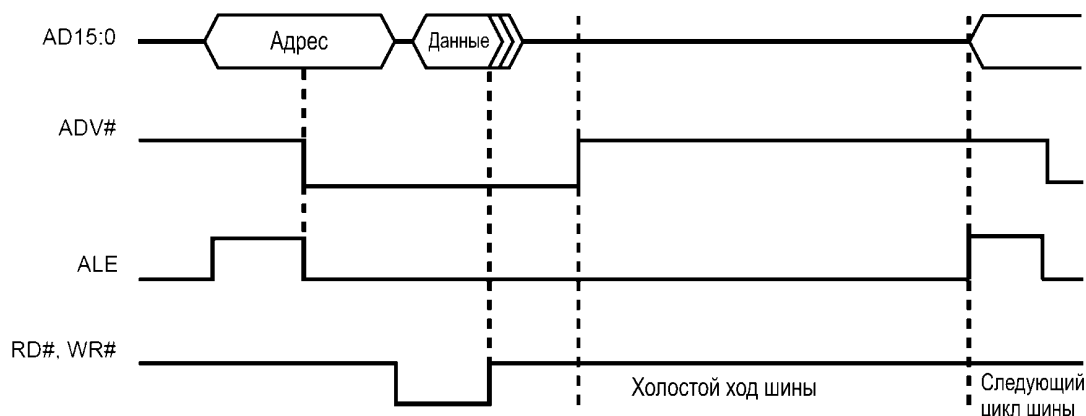


Рисунок 14.15 – Сравнение шинных циклов ALE и ADV#

На рисунках 14.16 и 14.17 показаны упрощённые схемы использования режима стробирования действительного адреса. На рисунке 14.16 показана 8-разрядная система с одной микросхемой ЭППЗУ (типа «флэш»), использующая ADV# для фиксации адреса и в качестве сигнала «выбор кристалла». На рисунке 14.17 показана 16-разрядная система с двумя микросхемами СППЗУ (типа EPROM) и аналогичным использованием сигнала ADV#.

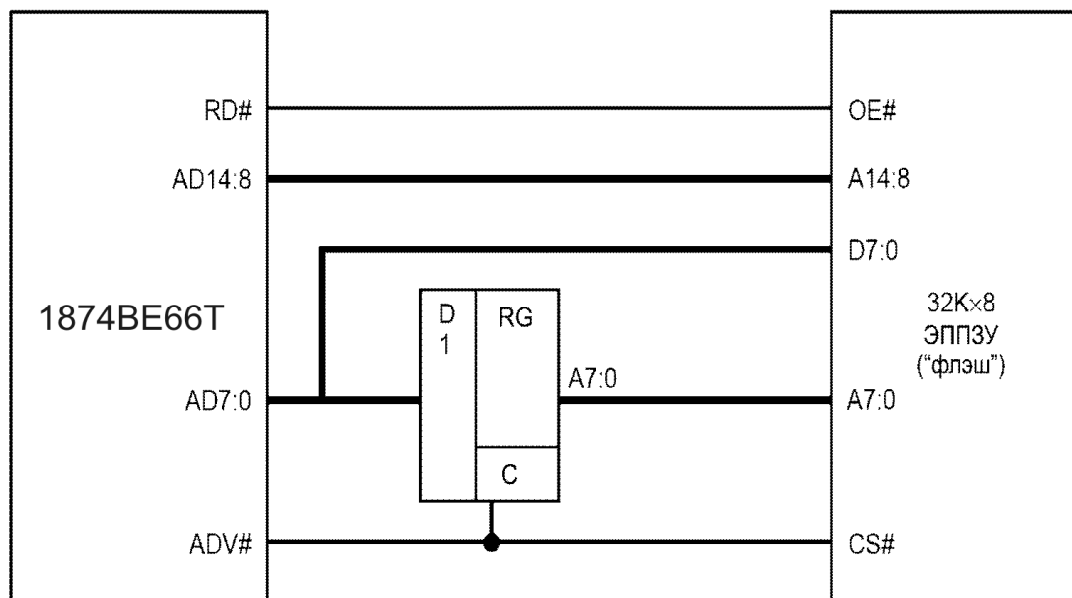


Рисунок 14.16 – 8-разрядная система с ЭППЗУ («флэш»)

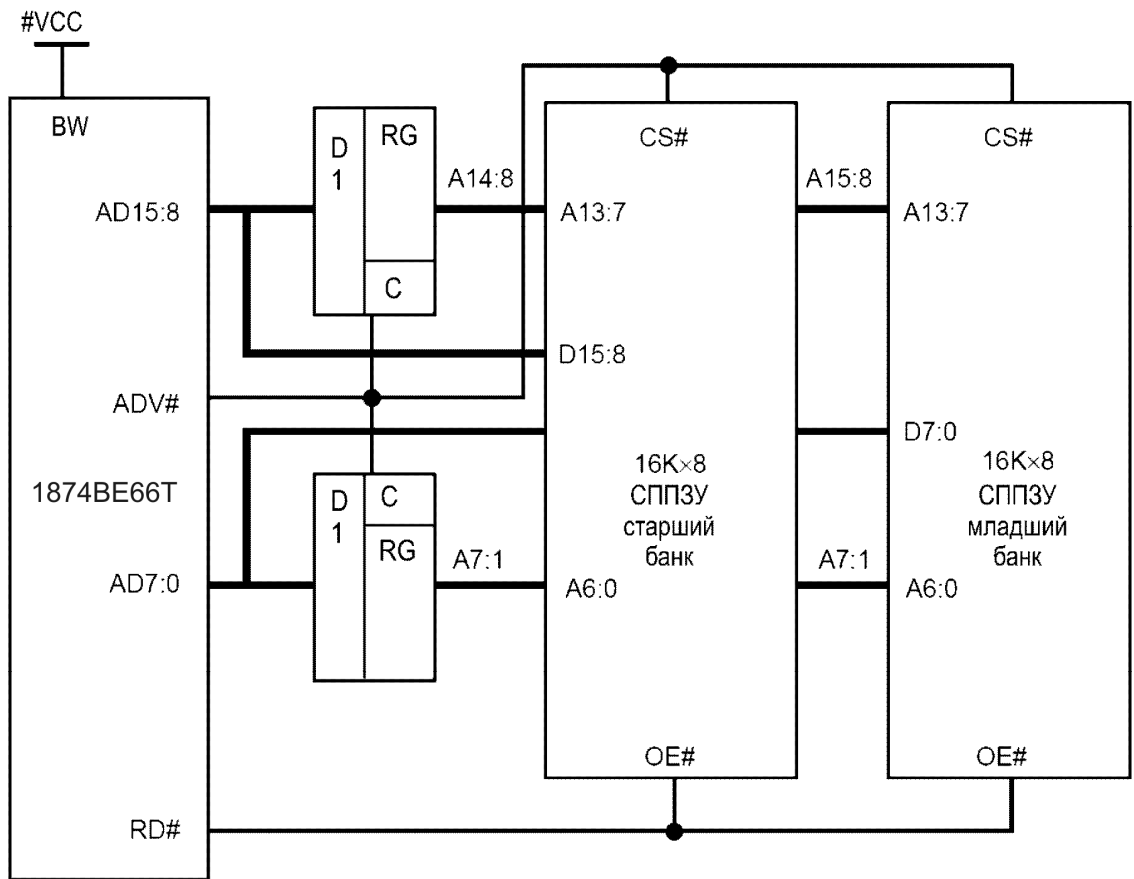


Рисунок 14.17 – 16-разрядная система с СППЗУ

Режим стробирования действительного адреса и записи

В данном режиме микроконтроллер формирует сигналы ADV#, RD#, WRL#, WRH#. Режим обычно используется в простых системах с 16-разрядной шиной. На рисунке 14.18 приведены временные диаграммы сигналов режима стробирования действительного адреса и записи. Сигнал RD# (не показан) аналогичен WRL# и WRH#. На рисунке 14.19 приведен пример системы, использующей режим стробирования действительного адреса и записи для доступа к 8-разрядному ОЗУ с 16-разрядной шиной данных.

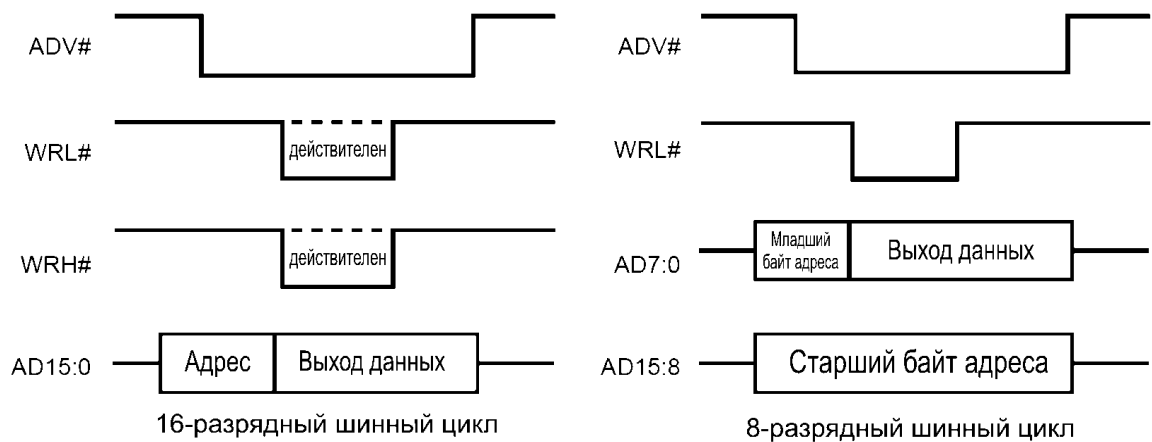


Рисунок 14.18 – Временные диаграммы сигналов в режиме стробирования действительного адреса и записи

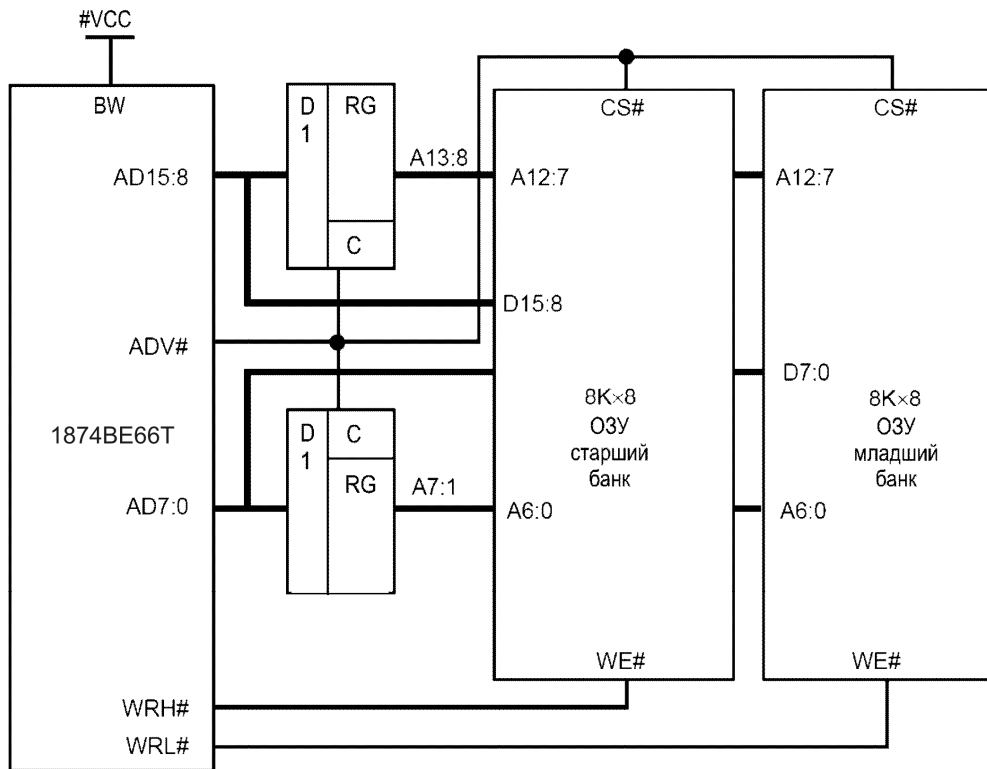


Рисунок 14.19 – 16-разрядная система с 8-разрядными микросхемами ОЗУ

На рисунках 14.20 – 14.22 приведены временные диаграммы работы микросхемы, в таблице 14.9 – значения динамических параметров сигналов интерфейса внешней памяти.

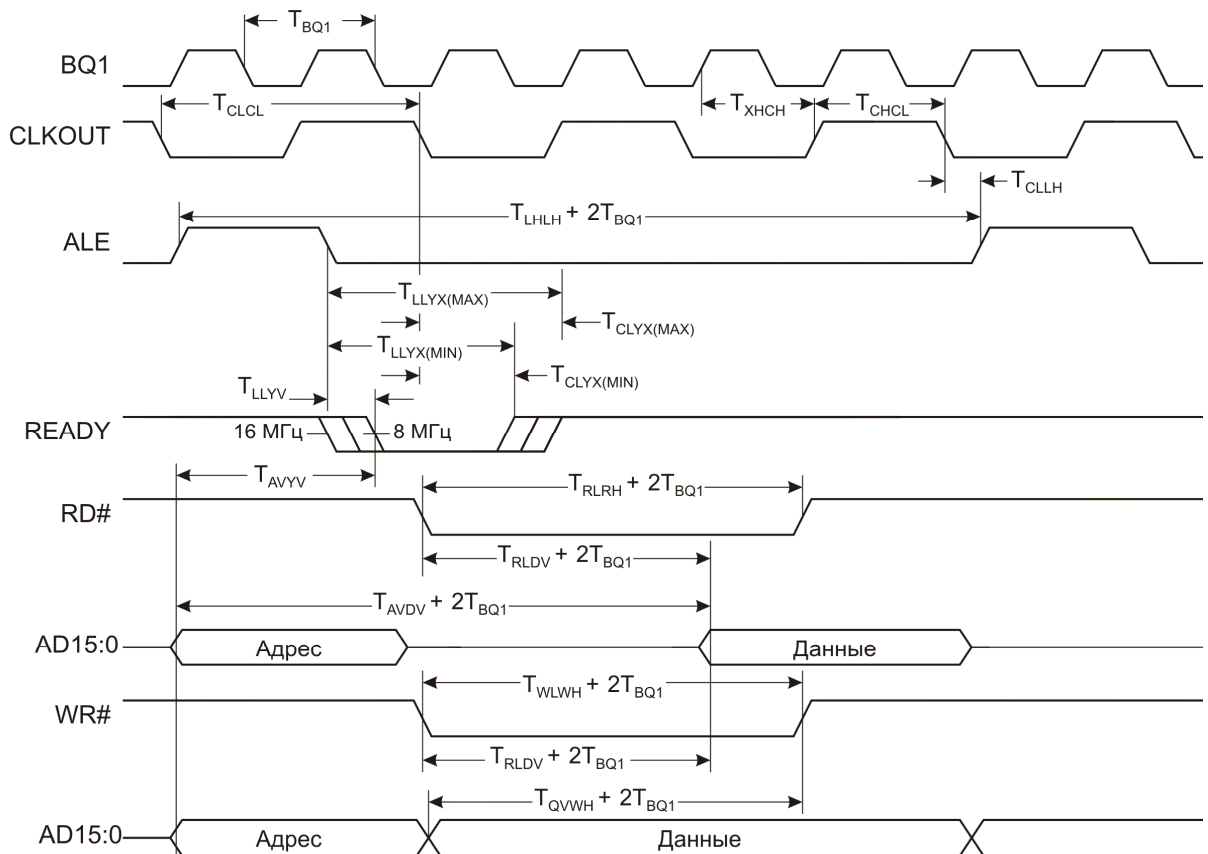


Рисунок 14.20 – Временная диаграмма для READY с одним циклом ожидания

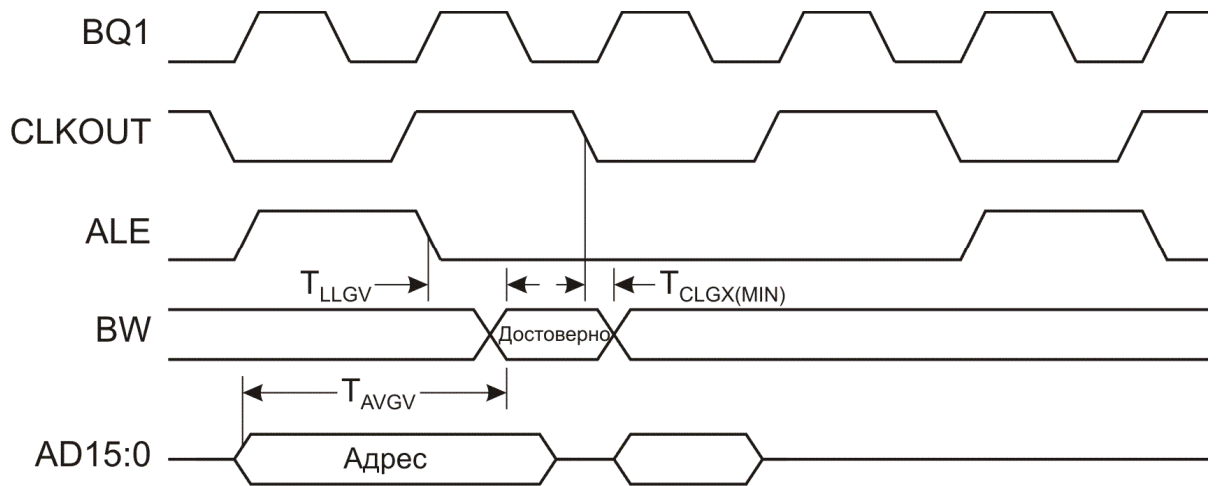


Рисунок 14.21 – Временная диаграмма для сигнала BW

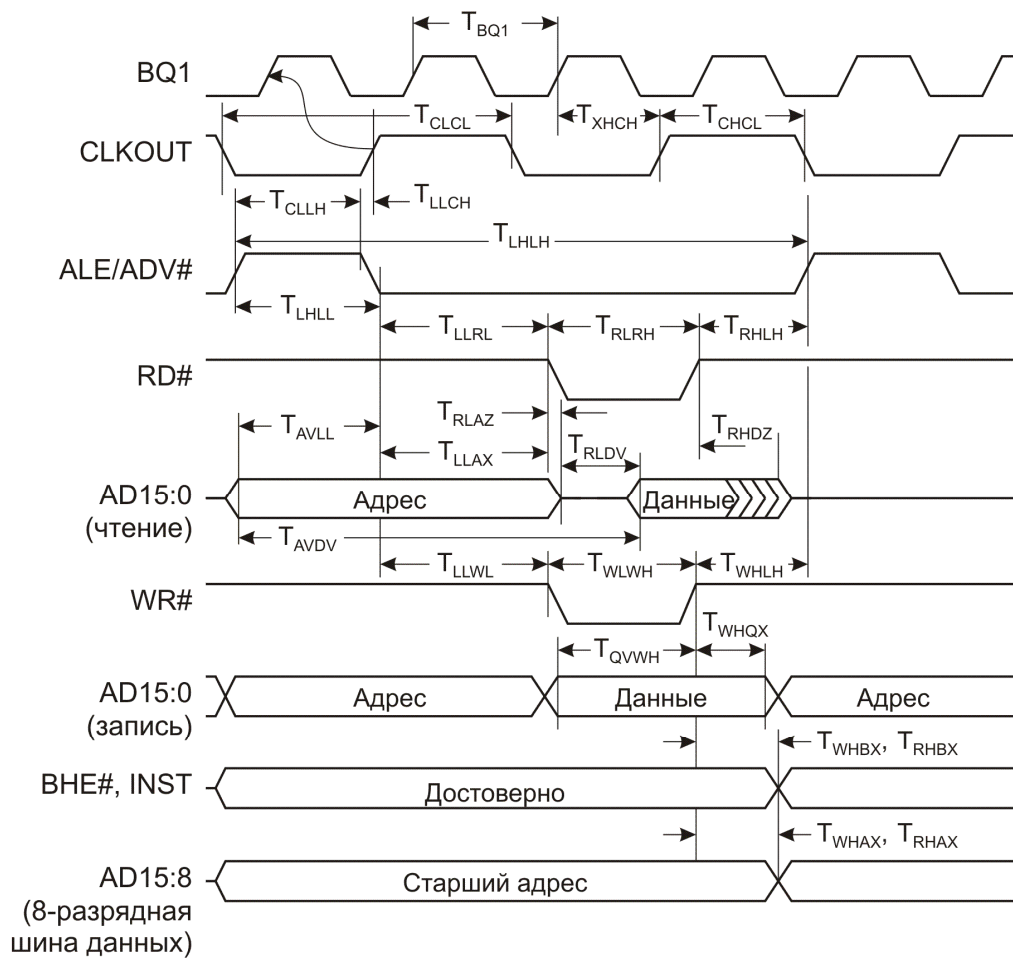


Рисунок 14.22 – Временная диаграмма системной шины

Т а б л и ц а 14.9 – Динамические (временные) параметры сигналов интерфейса внешней памяти

Наименование параметра	Обозначение на временной диаграмме	Описание параметра	Значение параметра		Единица измерения
			не менее	не более	
1	2	3	4	5	6
F_{BQ1}	F_{BQ1}	Частота тактового сигнала ¹⁾	8	16	МГц
T_{BQ1}	T_{BQ1}	Период тактового сигнала	62,5	125	нс
$T_{SU(RDY,L-A)}$	T_{AVYV}	Время установления низкого уровня сигнала READY относительно действительного сигнала адреса	–	$2T_{BQ1}$	нс
$T_{SU(RDY,L-ALE,L)}$	T_{LLYV}	Время установления низкого уровня сигнала READY относительно низкого уровня сигнала ALE	–	$T_{BQ1}-75$	нс
$T_{H(CO,L-RDY,L)}$	T_{CLYX}	Время удержания низкого уровня сигнала READY относительно низкого уровня сигнала CLKOUT ²⁾	0	$T_{BQ1}-30$	нс
$T_{H(ALE,L-RDY,L)}$	T_{LLYX}	Время удержания низкого уровня сигнала READY относительно низкого уровня сигнала ALE	$T_{BQ1}-15$	$2T_{BQ1}-40$	нс
$T_{SU(BW-A)}$	T_{AVGV}	Время установления действительного сигнала BW относительно действительного сигнала адреса	–	$2T_{BQ1}-75$	нс
$T_{SU(BW-ALE,L)}$	T_{LLGV}	Время установления сигнала BW относительно низкого уровня сигнала ALE	–	$T_{BQ1}-60$	нс
$T_{H(CO,L-BW)}$	T_{CLGX}	Время удержания сигнала BW относительно низкого уровня сигнала CLKOUT	0	–	нс
$T_{d(A-DI)}$	T_{AVDV}	Время задержки действительных входных данных относительно действительного адреса ³⁾	–	$3T_{BQ1}-55$	нс
$T_{SU(DI-RD,L)}$	T_{RLDV}	Время установления действительных входных данных относительно низкого уровня сигнала RD# ³⁾	–	$T_{BQ1}-22$	нс
$T_{SU(DI,Z-RD,H)}$	T_{RHDZ}	Время установления третьего состояния входных данных относительно высокого уровня сигнала RD#	–	T_{BQ1}	нс
$T_{d(BQ1,H-CO)}$	T_{XHCH}	Время задержки сигнала CLKOUT относительно высокого уровня сигнала BQ1	30	110	нс
$T_{CY(CO)}$	T_{CLCL}	Период сигнала CLKOUT	$2T_{BQ1}$		нс
$T_{WH(CO)}$	T_{CHCL}	Длительность высокого уровня сигнала CLKOUT	$T_{BQ1}-10$	$T_{BQ1}+15$	нс
$T_{d(CO,HL-ALE,LH)}$	T_{CLLH}	Время задержки нарастающего фронта сигнала ALE относительно спадающего фронта сигнала CLKOUT	–5	15	нс

Продолжение таблицы 14.9

1	2	3	4	5	6
$T_{d(CO,HL-ALE,HL)}$	T_{LLCH}	Время задержки спадающего фронта сигнала ALE относительно нарастающего фронта сигнала CLKOUT	-20	15	нс
$T_{CY(ALE)}$	T_{LHLH}	Период сигнала ALE ³⁾	$4T_{BQ1}$		нс
$T_{WH(ALE)}$	T_{LHLL}	Длительность высокого уровня сигнала ALE	$T_{BQ1}-10$	$T_{BQ1}+10$	нс
$T_{SU(A-ALE,HL)}$	T_{AVLL}	Время установления адреса относительно спадающего фронта сигнала ALE	$T_{BQ1}-15$	-	нс
$T_{H(ALE,HL-A)}$	T_{LLAX}	Время удержания адреса относительно спадающего фронта сигнала ALE	$T_{BQ1}-40$	-	нс
$T_{d(ALE,HL-RD,L)}$	T_{LLRL}	Время задержки низкого уровня сигнала RD# относительно спадающего фронта сигнала ALE	$T_{BQ1}-30$	-	нс
$T_{SU(RD,HL-CO,HL)}$	T_{RLCL}	Время установления низкого уровня сигнала RD# относительно спадающего фронта сигнала CLKOUT	4	30	нс
$T_{WL(RD)}$	T_{RLRH}	Длительность низкого уровня сигнала RD# ³⁾	$T_{BQ1}-5$	$T_{BQ1}+25$	нс
$T_{d(ALE,LH-RD,LH)}$	T_{RHLH}	Время задержки нарастающего фронта сигнала RD# относительно нарастающего фронта сигнала ALE ⁴⁾	T_{BQ1}	$T_{BQ1}+25$	нс
$T_{d(RD,LH-A,Z)}$	T_{RLAZ}	Время задержки третьего состояния адреса относительно нарастающего фронта сигнала RD#	-	5	нс
$T_{d(ALE,HL-WR,HL)}$	T_{LLWL}	Время задержки спадающего фронта сигнала WR# относительно спадающего фронта сигнала ALE	$T_{BQ1}-10$	-	нс
$T_{d(CO,HL-WR,HL)}$	T_{CLWL}	Время задержки спадающего фронта сигнала WR# относительно спадающего фронта сигнала CLKOUT	0	25	нс
$T_{SU(DI-WR,LH)}$	T_{QVWH}	Время установления сигнала данных относительно нарастающего фронта сигнала WR#	$T_{BQ1}-23$	-	нс
$T_{WL(WR)}$	T_{WLWH}	Длительность низкого уровня сигнала WR# ³⁾	$T_{BQ1}-30$	-	нс
$T_{H(WR,LH-D)}$	T_{WHQX}	Время удержания сигнала данных относительно нарастающего фронта сигнала WR#	$T_{BQ1}-25$	-	нс
$T_{V(ALE,LH-WR,H)}$	T_{WHLH}	Время сохранения высокого уровня сигнала WR# относительно нарастающего фронта ALE ⁴⁾	$T_{BQ1}-10$	$T_{BQ1}+15$	нс
$T_{H(BHIN-WR,LH)}$	T_{WHBX}	Время удержания сигналов BHE#, INST относительно нарастающего фронта сигнала WR#	$T_{BQ1}-10$	-	нс
$T_{H(ADH-WR,LH)}$	T_{WHAX}	Время удержания сигналов AD15:8 относительно нарастающего фронта сигнала WR# ⁵⁾	$T_{BQ1}-30$	-	нс

Окончание таблицы 14.9

1	2	3	4	5	6
$T_{H(BHIN-RD, LH)}$	T_{RHVX}	Время удержания сигналов BHE#, INST относительно нарастающего фронта сигнала RD#	$T_{BQ1}-10$	–	нс
$T_{H(ADH-RD, LH)}$	T_{RHAX}	Время удержания сигналов AD15:8 относительно нарастающего фронта сигнала RD# ⁵⁾	$T_{BQ1}-30$	–	нс

1) Параметры измерены на частоте 8 МГц.

2) Добавляются циклы ожидания, если превышает максимальное значение.

3) Если используются циклы ожидания, то временная добавка равна $2 \times T_{BQ1} \times N$, где N – число циклов ожидания.

4) Для циклов следующих друг за другом.

5) 8-битная шина.

Примечания

1 Емкости нагрузок по всем выводам $C_L = 100$ пФ, $F_{BQ1} = 16$ МГц.

2 Нормы динамических параметров приведены в диапазоне питающих напряжений $U_{#VCC} = (4,5 - 5,5)$ В и диапазоне рабочих температур среды.

3 Условные обозначения сигналов:

A – сигнал адреса;

DI – сигнал входных данных;

BHIN – сигналы BHE# и INST;

ADH – старший байт шины адреса-данных.

15 Программирование постоянной памяти

Контроллер содержит 16 Кбайт запоминающего устройства EPROM типа OTPROM, так как не имеет окна в корпусе микросхемы для стирания ультрафиолетовым излучением. В этом разделе описываются процедуры и принципы, которые могут помочь запрограммировать устройство. В данном разделе содержится следующая информация:

- краткий обзор режимов программирования;
- карта распределения памяти EPROM;
- защита памяти;
- программирование ширины программирующего импульса;
- модифицированный алгоритм быстрого программирования;
- выводы в режиме программирования;
- введение устройства в режимы программирования;
- программирование в режиме «SLAVE»;
- автопрограммирование «AUTO»;
- «RUN-TIME» программирование.

15.1 Режимы программирования

Когда выбирается режим программирования, устройство отзывается на него выполнением алгоритма, который находится в области тестового ПЗУ. Устройство поддерживает несколько режимов программирования.

Режим программирования SLAVE. В этом режиме для программирования EPROM необходимо использовать специальное устройство – программатор. Можно запрограммировать или верифицировать отдельные слова или наборы слов в EPROM.

Режим автопрограммирования AUTO. В этом режиме EPROM программируется из внешней памяти без использования программатора. Используя этот режим, невозможно запрограммировать биты UPROM и PCCB (для их записи необходимо использовать SLAVE режим). После программирования возможно использование режима ROM-DUMP для записи всей EPROM во внешнюю память для последующей верификации. Режим автопрограммирования обычно используется для программирования небольшого количества микроконтроллеров после того, как закончены этапы разработки и тестирования программного обеспечения (ПО).

Также возможно программирование ячеек EPROM без ввода контроллера в режим программирования. Такой режим называется RUN-TIME режимом. При этом ПО само контролирует количество и длительность программирующих импульсов.

15.2 Карта распределения памяти EPROM

EPROM содержит ячейки памяти специального назначения и память программ (таблица 15.1). 128 байт памяти специального назначения используются для хранения векторов прерываний, байтов конфигурации кристалла (CCB) и ключа безопасности. Также некоторые ячейки зарезервированы для тестирования. Необходимо записывать значения (20_H и FF_H) в зарезервированные ячейки согласно таблице 15.1. Остальная часть EPROM может быть использована для хранения кода.

Т а б л и ц а 15.1 – Карта распределения памяти

Диапазон адресов	Описание
1	2
$5FFF_H$ 2080_H	Память программ
$207F_H$ $205E_H$	Зарезервированы (должны содержать FF_H)

Продолжение таблицы 15.1

1	2
205D _H 2040 _H	PTS векторы
203F _H 2030 _H	Старшие векторы прерываний
202F _H 2020 _H	Ключ безопасности
201F _H 201С _H	Зарезервированы (должны содержать FF _H)
201В _H	Зарезервирован (должен содержать 20 _H)
201А _H	ССВ1
2019 _H	Зарезервирован (должен содержать 20 _H)
2018 _H	ССВ0
2017 _H 2014 _H	Зарезервированы (должны содержать FF _H)
2013 _H 2000 _H	Младшие векторы прерываний

15.3 Защита памяти

Существуют несколько механизмов, обеспечивающих ограничение доступа к внутренней и внешней памяти. Комбинирование битов защиты чтения и записи в байте конфигурации кристалла (CCR0) с ключом безопасности обеспечивают множество разнообразных уровней защиты памяти. Два UPROM бита запрещают выборку команд и данных из внешней памяти (рисунок 15.1).

Управление доступом к внутренней памяти

Биты защиты в регистре конфигурации кристалла (CCR0) управляют доступом к EPROM. Процедура сброса загружает CCR из CCB в нормальном режиме и из PCCB в режиме программирования. Необходимо запрограммировать CCB, используя любой из режимов программирования, но для записи PCCB нужно использовать режим SLAVE.

Доступ к EPROM в нормальном режиме функционирования

В нормальном режиме программирования биты защиты CCB0 управляют разрешением чтения и записи в EPROM. В таблице 15.2 описываются варианты защиты. Необходимо запрограммировать CCB, используя любой из режимов программирования.

Т а б л и ц а 15.2 – Защита памяти в нормальном режиме функционирования

Защита чтения LOC1 (CCR0.7)	Защита записи LOC0 (CCR0.6)	Режим защиты
1	1	Защиты нет. EPROM может читаться. Разрешён Run-time режим
1	0	Защита записи. RUN-TIME программирование запрещено. EPROM может читаться
0	1	Защита чтения. RUN-TIME программирование запрещено. Если идет выполнение программы из внешней памяти, то могут читаться только векторы прерываний и CCB. Ключ безопасности защищен от записи
0	0	Защита записи и чтения. RUN-TIME программирование запрещено. Если идет выполнение программы из внешней памяти, то могут читаться только векторы прерываний и CCB

Очистка ССВ0.6 обеспечивает защиту всей EPROM от случайной или преднамеренной записи. С разрешением защиты записи попытка записи заставляет контроллер шины выполнять шинный цикл последовательной записи, однако не запрещает VPR или запись данных в EPROM.

Очистка ССВ0.7 обеспечивает защиту чтения EPROM и защиту ключа безопасности от записи. С этой защитой контроллер шины не будет читать защищенную область EPROM. Попытка загрузить вторичный программный счетчик внешним адресом приводит к сбросу контроллера. Поскольку значение вторичного программного счетчика может опережать главный счётчик на четыре байта, то контроллер блокирует чтение кодов из последних четырех байт внутренней памяти. Векторы прерываний и ССВs не защищены от чтения, так как прерывания могут возникать и при выполнении команд из внешней памяти.

Доступ к EPROM в режиме программирования

Существуют три уровня защиты в режиме программирования:

- полный запрет программирования;
- полный запрет программирования, но разрешение режима ROM-DUMP после проверки ключа безопасности;
- разрешение после проверки ключа безопасности режимов ROM-DUMP, AUTO (автопрограммирования) и SLAVE.

Эти уровни защиты обеспечиваются битами защиты PCCB0, битами защиты ССВ0 и внутренним ключом безопасности (таблица 15.3). При входе в режимы программирования процедура сброса загружает PCCB в регистр конфигурации кристалла. Также во внутреннюю RAM загружается ССВ0 для обеспечения дополнительного уровня безопасности.

Можно запрограммировать байты ССВ, используя любой из режимов, но только в режиме SLAVE разрешается доступ к байтам PCCBs, и только в режимах SLAVE и автопрограммирования AUTO возможна запись внутреннего ключа безопасности.

Т а б л и ц а 15.3 – Варианты защиты памяти в режиме программирования

LOC1 (CCR0.7)		LOC0 (CCR0.6)		Программирование, ключ безопасности	Состояние защиты
PCCB	CCB	PCCB	CCB		
1	1	1	1	Нет	Защиты нет. Разрешены все режимы программирования
1	X	0	X	Да	Запрещены все режимы программирования. Режим ROM-DUMP разрешен после проверки ключа безопасности
1	0	1	0	Да	Режимы AUTO и SLAVE разрешены после проверки ключа безопасности
0	X	0	X	X	Полный запрет программирования

Если необходимо полностью запретить программирование, следует очистить оба бита защиты PCCB0. Если эти биты очищены, то запрещен вход в любой из режимов программирования.

Если необходимо запретить программирование, но разрешить ROM-DUMP, надо оставить бит защиты чтения PCCB0.7 незапрограммированным и очистить бит защиты записи PCCB0.6. Для защиты от неавторизованного чтения надо записать ключ безопасности. В режиме ROM-DUMP производится сравнение ключа безопасности, записанного во внутреннюю память, с ключом во внешней памяти, независимо от битов

защиты ССВ0. Если проверка прошла успешно, то продолжается выполнение последующих инструкций, если нет, контроллер входит в бесконечный цикл.

Если необходимо разрешить режимы SLAVE, автопрограммирования и ROM-DUMP нужно оставить оба бита защиты РССВ0 незапрограммированными. Чтобы защитить контроллер от неавторизованного программирования, надо очистить биты защиты ССВ0 и записать ключ безопасности. После того, как устройство войдет в SLAVE режим или режим автопрограммирования, соответствующая процедура из тестовой области ROM читает биты защиты ССВ0. Если любой бит защиты памяти установлен, процедура сравнивает ключ безопасности, расположенный во внутренней памяти, с внешним ключом. Если проверка прошла успешно, процедура продолжается; в противном случае, устройство входит в бесконечный внутренний цикл.

Можно программировать внутренний ключ безопасности либо в режиме AUTO, либо в режиме SLAVE. Один раз запрограммировав ключ секретности, необходимо обеспечить сравнение ключей при любом режиме программирования. Для режимов AUTO и ROM-DUMP сравнение ключей происходит во внешней памяти. Для режима SLAVE необходимо программировать ключ секретности по соответствующим адресам последовательность PROGRAM WORD. Сравнение с ключем секретности происходит даже если его не программировать.

Примечание – Если ключ безопасности остается незапрограммированным (заполненным FFFF_H), возможен неавторизованный доступ к EPROM, используя внешнюю память с незапрограммированным внешним ключом или используя режим SLAVE.

Управление выборкой из внешней памяти

Два UPROM (нестираемых) бита запрещают выборку команд и данных из внешней памяти. Если UPROM биты записаны, попытка обращения к внешней памяти приведет к сбросу контроллера. UPROM биты можно программировать, используя SLAVE режим.

Программируя бит DEI, можно запретить выборку команд из внешней памяти. Попытка загрузить внешний адрес во вторичный программный счетчик приведет к сбросу устройства. Поскольку значение вторичного программного счетчика может опережать значение главного счётчика на четыре байта, то контроллер блокирует чтение кодов из последних четырех байт внутренней памяти. Автоматический сброс также дает дополнительную защиту от выполнения неконтролируемых команд.

Программируя бит DED, можно запретить как запись, так и чтение данных из внешней памяти. Попытка обратиться к внешней памяти приведёт к сбросу контроллера. Установка этого бита запрещает режим ROM-DUMP.

Для правильного программирования этих битов необходимо использовать таблицу 15.5. Программирование осуществляется в SLAVE режиме. В нормальном режиме можно определить значения этих битов, читая регистр UPROM битов USFR (рисунок 15.1, таблица 15.4).

Регистр USFR

Адрес: 1FF6_H

Состояние после сброса: 02_H

7	–	–	–	–	DEI	DED	–	–	0
---	---	---	---	---	-----	-----	---	---	---

Рисунок 15.1 – Нестираемый регистр USFR

Т а б л и ц а 15.4 – Разряды регистра USFR

Номер бита	Обозначение	Функция
7 – 4	–	Зарезервированы. Всегда записывать нули
3	DEI	Установка этого бита запрещает выборку команд из внешней памяти. Любая попытка загрузить внешний адрес приводит к сбросу устройства
2	DED	Установка этого бита запрещает обращение к внешним данным. Любая попытка записи или чтения из внешней памяти приведет к сбросу
1, 0	–	Зарезервированы. Всегда записывать нули

Пользователь может проверить UPROM биты, чтобы удостовериться что они запрограммированы, но не может стереть их.

Т а б л и ц а 15.5 – Данные и адреса для установки битов UPROM (DED, DEI) в SLAVE режиме

Бит	Данные для установки бита	Адрес записи данных для установки бита
DEI	08 _H	0718 _H
DED	04 _H	0758 _H

15.4 Ширина программирующего импульса

Ширина программирующего импульса задается различными способами, в зависимости от режима программирования. В SLAVE режиме ширина импульса управляется PALE# сигналом. В режиме автопрограммирования значение программирующего импульса загружается в PPW из внешней памяти (EPROM). (В RUN-TIME режиме программное обеспечение управляет шириной импульса.)

Регистр PPW

Нет прямого доступа.

Чтобы правильно определить PPW_VALUE для нужной частоты устройства, используют следующую формулу. Результат округляют до следующего старшего целого числа.

$$PPW_VALUE = 62,5 \times 2 \times F_{BQ1} , \quad (15.1)$$

где PPW_VALUE – 16-битное слово;

F_{BQ1} – входная частота в МГц.

PPW не имеет прямого доступа.

Регистр ширины программирующего импульса загружается из внешней EPROM (адреса 14_H и 15_H) в режиме автопрограммирования. PPW_VALUE определяет ширину программирующего импульса.

15							8
PPW15	PPW14	PPW13	PPW12	PPW11	PPW10	PPW9	PPW8
7							0
PPW7	PPW6	PPW5	PPW4	PPW3	PPW2	PPW1	PPW0

Рисунок 15.2 – Регистр PPW программирования ширины импульса

Т а б л и ц а 15.6 – Разряды регистра PPW

Номер бита	Обозначение	Функция
15 – 0	PPW15 – PPW0	PPW_VALUE Это значение устанавливает ширину программирующего импульса для режима автопрограммирования. Чтобы вычислить PPW_VALUE, используют формулу (15.1). В таблице 15.7 приведены значения для различных частот

В таблице 15.7 представлены вычисления и значения PPW_VALUE для частот 8 МГц и 16 МГц. Минимальная ширина программирующего импульса составляет 500 мкс.

Т а б л и ц а 15.7 – Пример вычисления PPW_VALUE

F_{BQ1}	Требуются два импульса по 500 мкс
8 МГц	$PPW_VALUE = 62,5 \times 2 \times 8 = 1000 = 03E8_H$
16 МГц	$PPW_VALUE = 62,5 \times 2 \times 16 = 2000 = 07D0_H$

15.5 Модифицированный QUICK-PULSE алгоритм

Режимы SLAVE и автопрограммирования используют модифицированный QUICK-PULSE алгоритм (рисунок 15.3). В соответствии с модифицированным алгоритмом QUICK-PULSE формируется программирующий импульс для каждого слова EPROM. После необходимого количества программирующих импульсов происходит верификация записанных данных. Ошибка верификации переключает PVER сигнал, но не приводит к остановке процесса программирования. Этот процесс будет повторяться до тех пор, пока каждое слово EPROM не запишется и не верифицируется.

П р и м е ч а н и е – Контроллер использует два программирующих импульса.

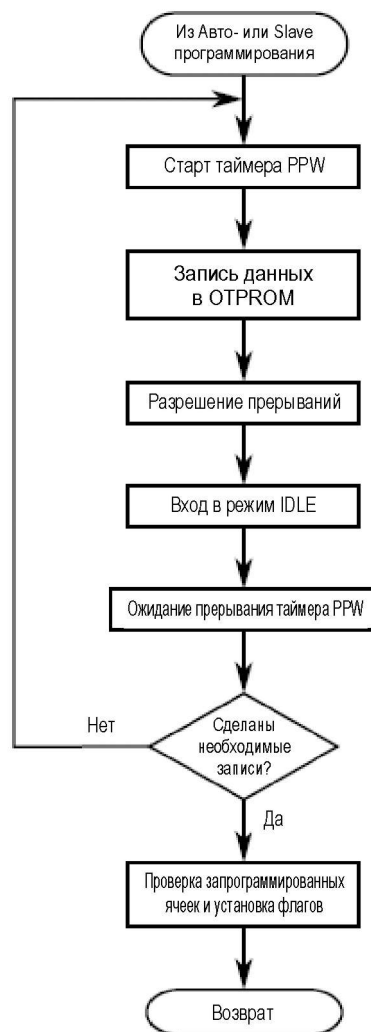


Рисунок 15.3 – Модифицированный QUICK-PULSE алгоритм

В режиме автопрограммирования программирующий импульс повторяется дважды с той длительностью, которая задается во внешней EPROM. В SLAVE режиме ширина программирующего импульса управляется сигналом PALE#. Во всех случаях, для успешного программирования ширина импульса должна быть не менее 500 мкс.

15.6 Вывода в режиме программирования

На рисунке 15.4 показаны сигналы, используемые для программирования, а в таблице 15.8 они описываются. EA#, VPR и PMODE выходы обеспечивают вход контроллера в режим программирования. Необходимо задать PMODE (P0.7:4) выходы таким образом, чтобы выбрать нужный режим программирования (таблица 15.9). В каждой процедуре программирования выходы порта 2 работают в режиме специальной функции сигнала. Порты 3 и 4 автоматически используют PBUS в течение программирования.

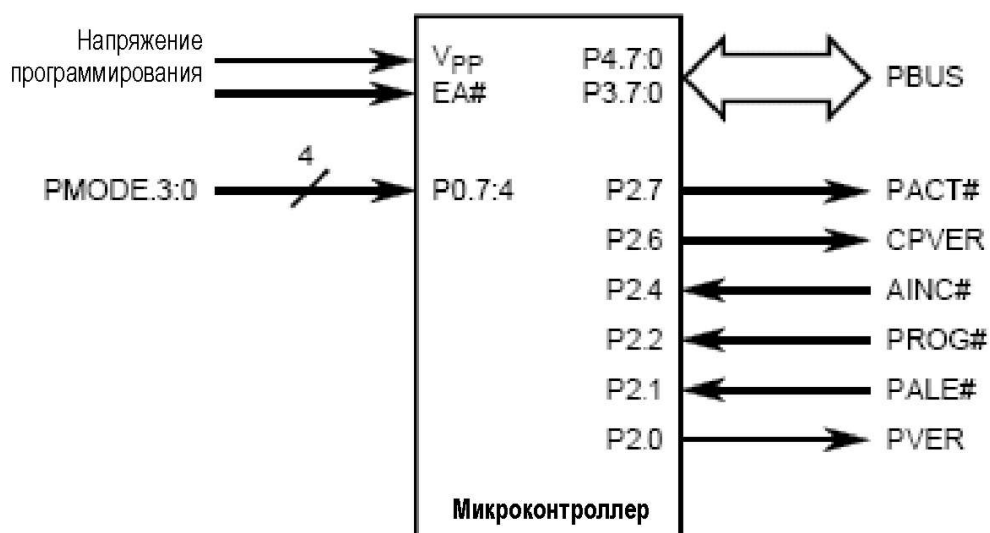


Рисунок 15.4 – Функции выводов в режиме программирования

Т а б л и ц а 15.8 – Описание выводов в режиме программирования

Вывод порта	Сигнал специальной функции	Тип	Режим программирования	Описание
1	2	3	4	5
P0.7 – P0.4	PMODE.3– PMODE.0	Вход	Все	Выбор режима программирования. Определяет режим программирования. Значения PMODE читаются после сброса устройства и не должны изменяться при программировании
P2.0	PVER	Выход	AUTO, SLAVE	Флаг верификации. В течение SLAVE или автопрограммирования, PVER обновляется после каждого импульса программирования. Высокий уровень показывает, что программирование ячейки прошло успешно, низкий – говорит об ошибке
P2.1	PALE#	Вход	SLAVE	Вход ALE в режиме программирования. В SLAVE режиме спадающий фронт заставляет контроллер читать команду и адрес с шины PBUS
P2.2	PROG#	Вход	SLAVE	Программирование. В режиме программирования по спаду сигнала фиксируются данные на PBUS и начинается программирование. По нарастающему фронту программирование заканчивается. В режиме чтения слова (ROM-DUMP) по спаду сигнала содержимое ячейки EPROM выводится в PBUS, по нарастанию – заканчивается передача данных
P2.4	AINC#	Вход	SLAVE	Автоинкремент. В течение SLAVE режима активный низкий уровень разрешает автоинкремент. (Автоинкремент позволяет считывать или записывать ячейки последовательно без передачи адреса при каждом обмене)

Окончание таблицы 15.8

1	2	3	4	5
P2.6	CPVER	Выход	SLAVE	Полная верификация программирования. Высокий уровень на этом выводе в режиме SLAVE показывает, что все ячейки запрограммированы верно
P2.7	PACT#	Выход	Авто, ROM-dump	Программирование активно. В режиме программирования AUTO или ROM-DUMP низкий уровень на выходе показывает, что идет процесс программирования или чтения, а высокий уровень индицирует окончание операции
P4.7–P4.0, P3.7–P3.0	PBUS	Вход/выход	SLAVE	Шина адрес/команды/данные. В течение SLAVE режима порты 3 и 4 служат двунаправленными портами с открытыми стоками для передачи команд, адресов и данных. В SLAVE режиме необходимо наличие внешних pull-up резисторов (поддержки высокого уровня)
P4.7–P4.0, P3.7–P3.0	PBUS	Вход/выход	Авто ROM-dump	Шина адрес/команды/данные. В течение автопрограммирования и ROM-DUMP порты 3 и 4 служат системной шиной для доступа к внешней памяти
–	EA#	Вход	Все	Внешний доступ. Управляет переходом контроллера в режим программирования. Если напряжение на EA# равно U_{VPR} , то при нарастающем фронте RESET# устройство переходит в режим программирования. EA# фиксируется только по нарастающему фронту сигнала RESET#
–	VPR	Вход	Все	Вывод «напряжение программирования». В течение программирования типовое значение напряжения на выводе VPR равно 12,5 В

15.7 Вход в режимы программирования

Для правильного выполнения программ необходимо обеспечить минимально необходимую аппаратную конфигурацию: вывод BQ1 подключен, неиспользованные входы объединены и подключены к питанию или земле. Перевод устройства в режим программирования производится подачей напряжения U_{PR} на EA# (плюс 12,5 В) во время нарастающего RESET#.

Выбор режима программирования

Значение PMODE (P0.7 – 4) управляет выбором режима программирования. PMODE фиксируется при нарастающем фронте сигнала RESET#. Для переключения режимов программирования необходимо сбросить устройство. В таблице 15.9 содержатся значения PMODE для каждого из режимов. Все другие значения PMODE зарезервированы.

Т а б л и ц а 15.9 – Значения PMODE

Значение PMODE	Режим программирования
5 _H	SLAVE
6 _H	ROM-DUMP (вывод содержимого памяти)
С _H	AUTO (автопрограммирование)

Включение и выключение питания при программировании

При начале программирования необходимо следовать следующим инструкциям по включению и выключению питания:

- Нельзя прикладывать напряжение к выводу VPR при низком $U_{\#VCC}$.
 - U_{PR} напряжение должно быть в пределах 1 В от $U_{\#VCC}$, если $U_{\#VCC}$ меньше 4,5 В. U_{PR} не должно превышать 4,5 В, пока $U_{\#VCC}$ не достигло 4,5 В.
 - Нельзя превышать максимальное значение U_{PR} .
 - $U_{EA\#}$ должен достигать напряжения программирования раньше U_{PR} .
 - Выводы PMODE (P0.7 – P0.4) должны быть установлены до начала переключения сигнала RESET# из состояния низкого уровня в состояние высокого уровня.
 - Все напряжения должны быть в пределах диапазонов, указанных в ТУ, частота генератора должна стабилизироваться до начала переключения сигнала RESET# из состояния низкого уровня в состояние высокого уровня.
 - Напряжения, прикладываемые к выводам #VCC, VPR, EA# и RESET#, должны быть стабильны, без шумов, выбросов и помех.
 - Все выводы #0V и $\cap 0V$ должны быть хорошо заземлены.
- Примечание** – Невыполнение этих требований может привести к выходу устройства из строя.

Последовательность включения питания

- 1 Удерживать низкий уровень на RESET#, пока напряжение $U_{\#VCC}$ стабилизируется. В это время выводы EA# и VPR могут быть не подключены.
- 2 После стабилизации $U_{\#VCC}$ и генератора удерживать RESET# в низком уровне и подать напряжение U_{PR} с вывода VPR на EA#.
- 3 После стабилизации $U_{EA\#}$ подать напряжение программирования на вывод VPR (U_{PR}).
- 4 Установить значение PMODE для выбора алгоритма программирования.
- 5 Подать высокий уровень на RESET#.
- 6 Закончить работу с выбранным алгоритмом программирования.

Последовательность выключения питания

- 1 Установить и удерживать RESET# в низком уровне в процессе выключения питания.
- 2 Снять напряжение программирования с вывода VPR и оставить его неподключенным.
- 3 Снять напряжение программирования с вывода EA# и оставить его неподключенным.
- 4 Снять питание с вывода #VCC и подождать, пока установится 0 В.

15.8 Режим программирования SLAVE

SLAVE режим позволяет записывать и считывать EPROM, включая биты PCCB и UPROM, используя EPROM программатор.

В этом режиме порты 3 и 4 работают как PBUS, передавая команды, адреса и данные. Наименьший значащий бит PBUS (P3.0) управляет типом команды («1» – программирование слова; «0» – чтение слова). Остальные 15 битов содержат адрес слова, которое будет запрограммировано или прочитано. Некоторые выводы порта 2 обеспечивают сигналы квитирования. AINC# сигнал управляет автоинкрементом, позволяя программировать или считывать последовательность ячеек EPROM. Это ускоряет процесс программирования, так как при этом не требуется формировать и декодировать последовательные адреса.

Примечание – Если во время программирования ключа безопасности произойдет сбой или возникнет состояние сброса, возможна запись неправильного

значения. Поэтому вначале необходимо записать всю программу и только потом биты защиты ССВ (ССВ0.6 и ССВ0.7).

Чтение слова-сигнатуры и значения программирующего напряжения

Слово-сигнатура идентифицирует устройство; напряжение программирования определяет напряжения U_{PR} и $U_{\#VCC}$, требуемые для программирования. Эта информация содержится в тестовом ROM по адресам 2070_H, 2072_H и 2073_H, однако непосредственный доступ к ней возможен по адресам 0070_H, 0072_H и 0073_H. Можно использовать команду чтения слова в SLAVE режиме для чтения сигнатуры и напряжения программирования. Внешний программатор может использовать эту информацию, чтобы определить тип устройства и эксплуатационные режимы. Никогда нельзя записывать в эти ячейки. Напряжения рассчитывают используя следующие уравнения (после преобразования значения ROM в десятичное число):

$$\text{напряжение} = 20 \times \text{значение ROM} / 256, \quad (15.2)$$

$$U_{\#VCC}(40_{\text{H}}) = \frac{20 \times 64}{256} = 5 \text{ (В)}, \quad (15.3)$$

$$U_{VPP}(0A0_{\text{H}}) = \frac{20 \times 160}{256} = 12,5 \text{ (В)}. \quad (15.4)$$

Т а б л и ц а 15.10 – Слово-сигнатура

Слово-сигнатура		$U_{\#VCC}$		U_{PR}	
Адрес	Значение	Адрес	Значение	Адрес	Значение
0070 _H	8794 _H	0072 _H	40 _H	0073 _H	0A0 _H

Схема включения и распределение памяти в SLAVE режиме

На рисунке 15.5 показана схема включения, а в таблице 15.11 показано распределение памяти для SLAVE режима.

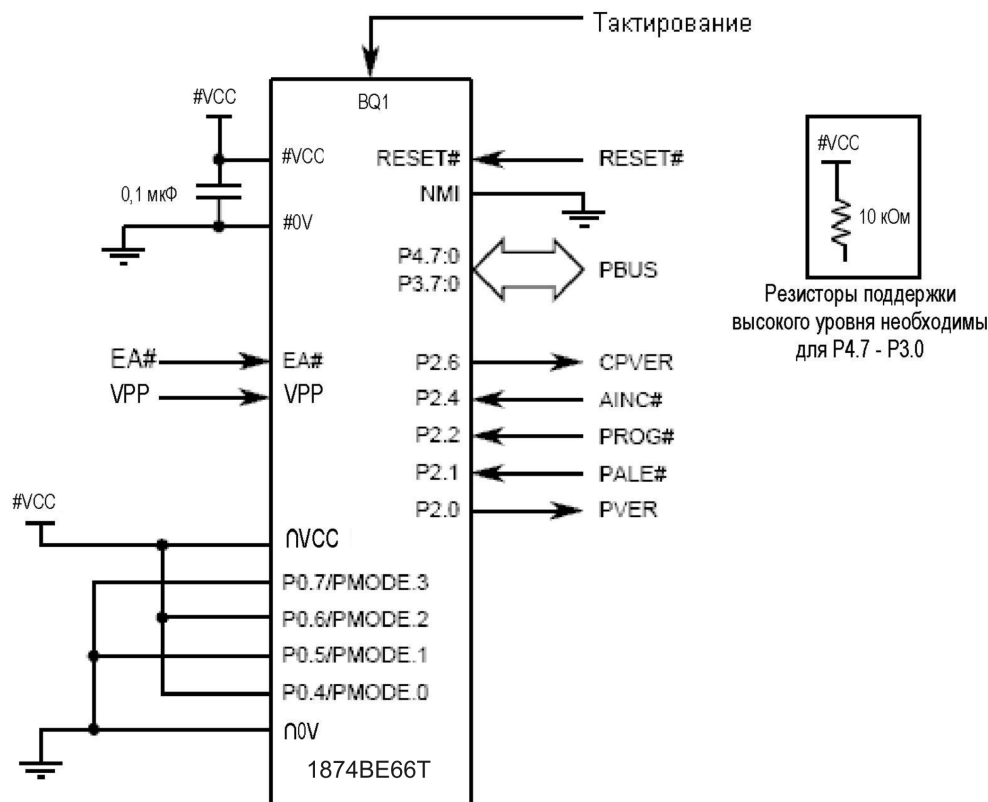


Рисунок 15.5 – Схема включения микроконтроллера для режима SLAVE

Т а б л и ц а 15.11 – Распределение памяти в режиме SLAVE

Описание	Адрес	Комментарии
EPROM	2000–5FFF _H	EPROM ячейки
DED	0758 _H	UPROM ячейка
DEI	0718 _H	UPROM ячейка
PCCB	0218 _H	Тестовая EPROM
Напряжения программирования (таблица 15.10)	0072 _H , 0073 _H	Только чтение
Слово-сигнатура	0070 _H	Только чтение

Оперативное окружение

Регистры конфигурации кристалла (CCR) определяют операционные режимы системы. Так как конфигурация в режиме программирования не обязательно совпадает с конфигурацией в обычном режиме работы, МК поддерживает эти различные конфигурации. Конфигурацию определяют, задав необходимые значения в байтах конфигурации кристалла (CCB), расположенных в EPROM. Конфигурацию в режиме программирования определяют, задав необходимые значения в регистрах PCCB, расположенных в тестовом EPROM.

На рисунке 15.6 представлено схематичное описание CCR. Во время сброса контроллер загружает байты CCB в CCR в нормальном режиме функционирования и PCCB – в режимах программирования. Можно программировать CCB, используя любой из методов программирования, но PCCB возможно программировать только в режиме SLAVE.

Регистры конфигурации кристалла CCR управляют циклами ожидания, режимом пониженного энергопотребления и внутренней защитой памяти. Эти регистры загружаются байтами PCCB в режимах программирования и байтами CCB в нормальном режиме.

Регистры CCR1, CCR0

Нет прямого доступа.

7								0
1	1	0	1	WDE	BW1	IRC2	0	
7								0
	LOC1	LOC0	IRC1	IRC0	ALE	WR	BW0	PD

Рисунок 15.6 – Регистры конфигурации кристалла (CCR1, CCR0)

Т а б л и ц а 15.12 – Разряды регистров конфигурации кристалла

Обозначение	Функция
1	2
WDE	Разрешение сторожевого таймера. По умолчанию PCCB запрещает функционирование сторожевого таймера
BW1	Выбор ширины шины. По умолчанию PCCB содержит бит, выбирающий управление выводом BUSWIDTH
IRC2	Управление сигналом READY. По умолчанию PCCB выбирает управление выводом READY

Окончание таблицы 15.12

1	2
LOC1 – LOC0	Биты защиты. По умолчанию ПССВ не активирует защиту
IRC1 – IRC0	Управление сигналом READY. По умолчанию ПССВ выбирает управление выводом READY
ALE	Выбор режима стробирования адреса. По умолчанию ПССВ выбирает сигнал ALE для стробирования
WR	Выбор режима стробирования записи. По умолчанию ПССВ выбирает сигналы WR# и ВНЕ# для стробирования
BW0	Выбор ширина шины. По умолчанию ПССВ содержит бит, выбирающий управление выводом BUSWIDTH
PD	Режим пониженного энергопотребления. По умолчанию ПССВ разрешает режим пониженного энергопотребления

Алгоритм программирования в режиме SLAVE

Алгоритм данного режима программирования состоит из трех процедур: процедуры декодирования адреса/команды, процедуры записи слова и процедуры чтения слова.

Процедура декодирования адреса/команды (рисунок 15.7) читает информацию с PBUS и передает управление процедурам записи или чтения слова, в зависимости от значения P3.0. Если P3.0 – единица, то происходит программирование слова, а остающиеся биты определяют адрес. Если P3.0 – ноль, то происходит чтение слова. Остающиеся биты также определяют адрес.

Процедура записи слова (Program Word) показанная на рисунке 15.8, проверяет биты защиты ССВ. Если любой из битов защиты был запрограммирован (ССВ0.6 или ССВ0.7), то необходимо иметь соответствующий ключ безопасности, чтобы получить доступ к устройству. Используя команду записи слова, записывают восемь последовательных слов, начиная с адреса 2020_H и кончая 202F_H. Процедура сохраняет эти восемь слов во внутреннем регистре и сравнивает их значение с внутренним ключом. Если ключи совпадают, то возможна дальнейшая работа с EPROM, иначе устройство входит в бесконечный цикл.

Процедура чтения слова (рисунок 15.10) также проверяет биты защиты ССВ. Если биты защиты незапрограммированы, то возможна выдача данных из EPROM на шину PBUS. Если любой бит запрограммирован, то возможно проводить циклы записи без чтения данных из EPROM.

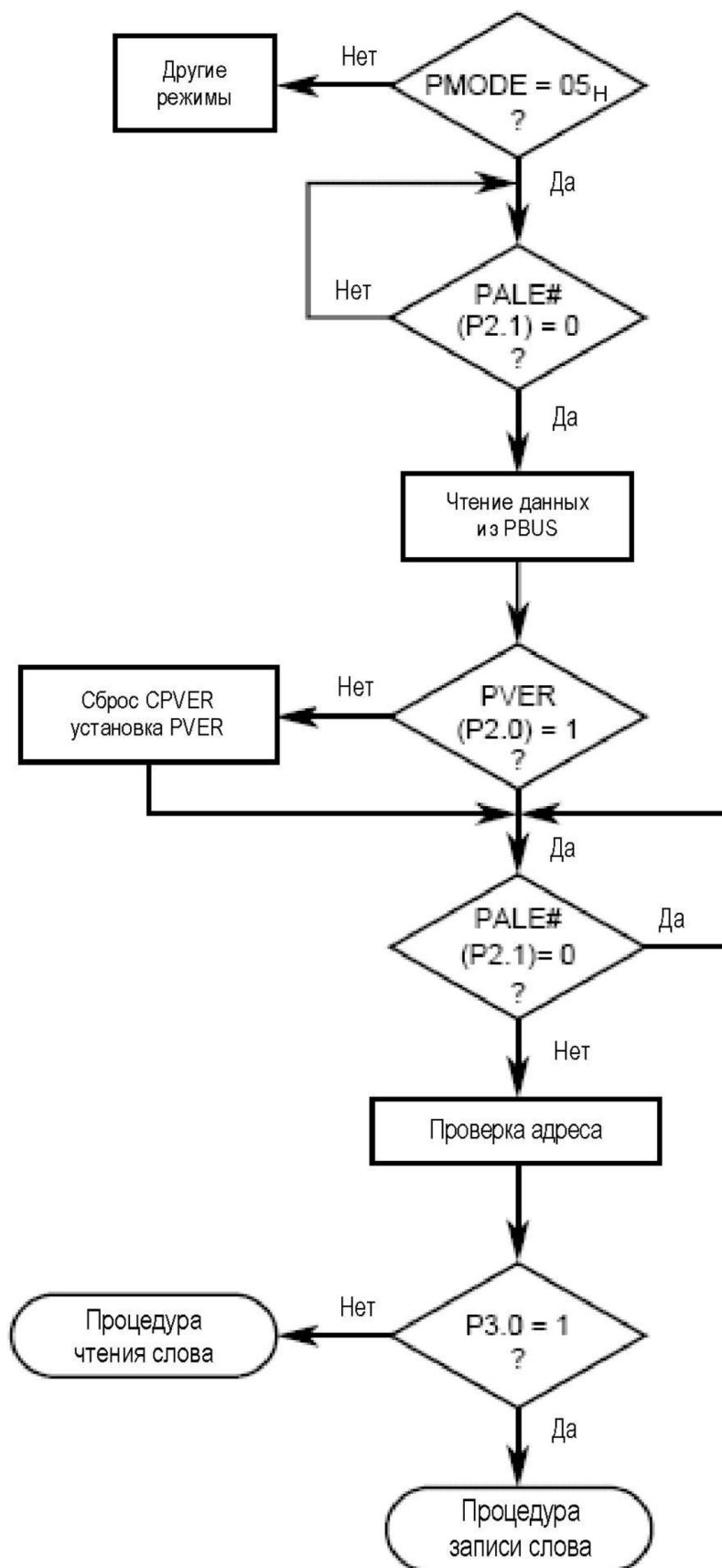


Рисунок 15.7 – Процедура декодирования адреса/команды

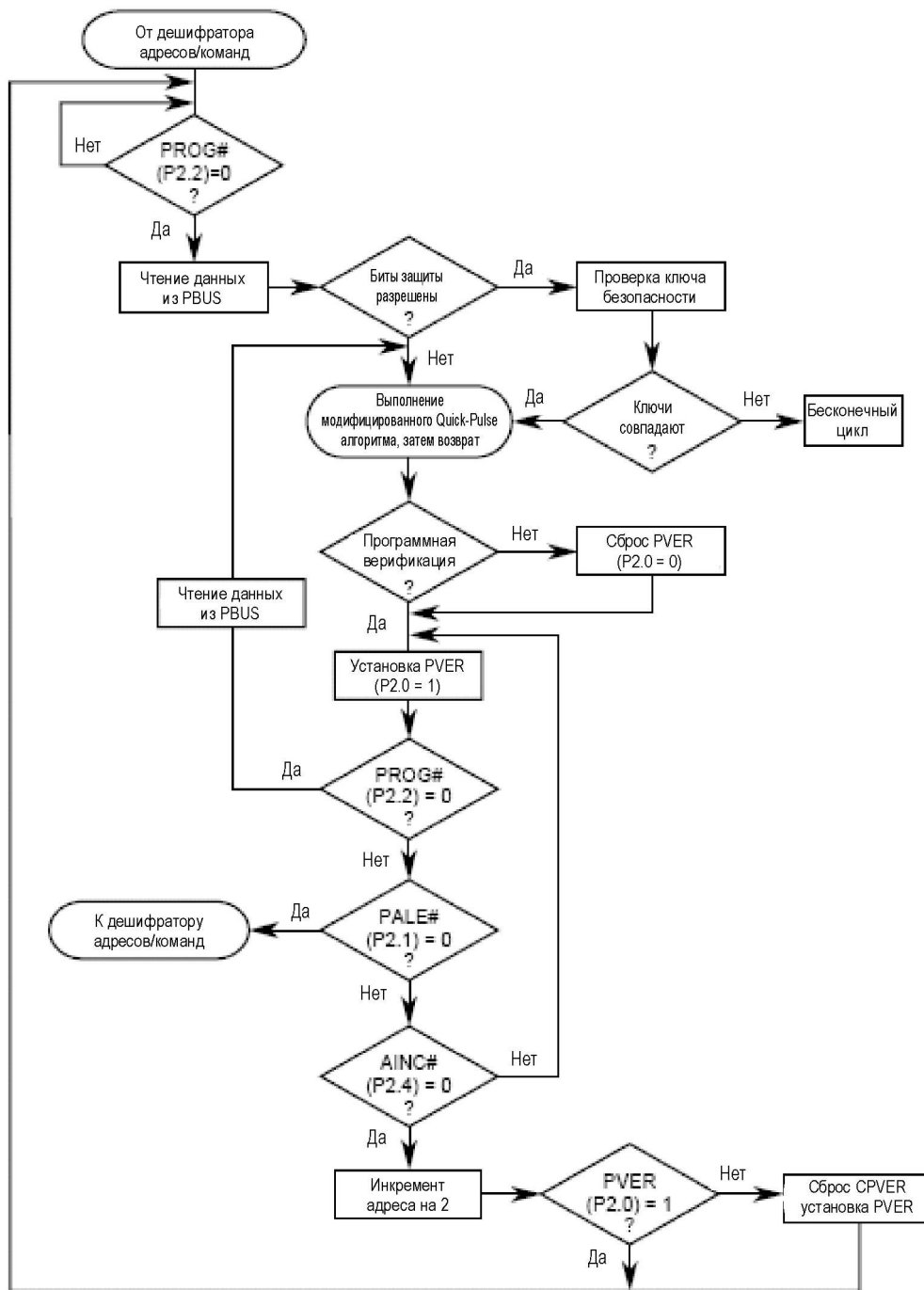


Рисунок 15.8 – Процедура программирования слов (Program Word)

На рисунке 15.9 показываются временные диаграммы для режима записи слова с автоинкрементом. По спадающему фронту сигнала PALE# происходит чтение команд и адресов с PBUS. По спадающему фронту сигнала PROG# происходит фиксация данных и начинается программирование. PROG# сигнал управляет длительностью программирующего импульса. По возрастающему фронту PROG# происходит верификация записанного значения и выставляется сигнал PVER. AINC# сигнал является дополнительным и служит для указания режима автоинкремента адреса. Если не используется сигнал AINC#, то необходимо послать новую команду для указания нового адреса.

На рисунке 15.11 показаны диаграммы в режиме чтения слов из памяти (Program Dump). Сигнал PROG# управляет шиной. В режиме выдачи слова вывод AINC# может остаться активным. Сигнал PROG# автоматически увеличивает адрес.

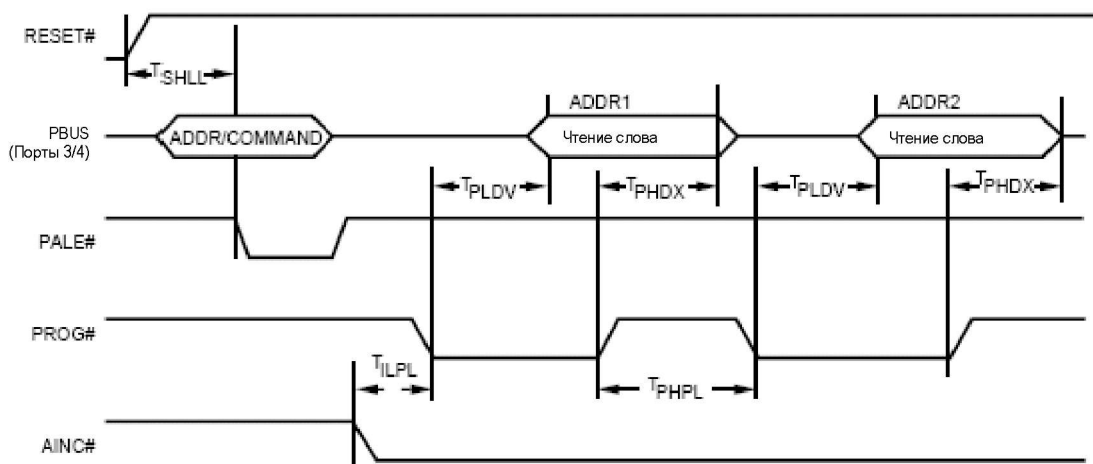


Рисунок 15.11 – Диаграммы чтения слова

Временные параметры и их обозначения

В таблице 15.13 представлены обозначения параметров, использующихся в диаграммах, а также их значения.

Т а б л и ц а 15.13 – Временные параметры в режимах программирования

Наименование параметра	Обозначение на временной диаграмме	Описание параметра	Значение параметра		Единица измерения
			не менее	не более	
1	2	3	4	5	6
$T_{SU(RDY,L-ALE,L)}$	T_{SHLL}	Время установления высокого уровня сигнала сброса относительно первого низкого уровня сигнала PALE#	1100	–	T_{BQ1}
$T_{WL(PALE)}$	T_{LLLH}	Длительность низкого уровня сигнала PALE#	50	–	T_{BQ1}
$T_{SU(A-PALE,L)}$	T_{AVLL}	Время установки адреса относительно низкого уровня сигнала PALE#	0	–	T_{BQ1}
$T_{H(PALE,L-A)}$	T_{LLAX}	Время удержания адреса относительно низкого уровня сигнала PALE#	100	–	T_{BQ1}
$T_{d(PROG,L-D)}$	T_{PLDV}	Время задержки вывода слова достоверных данных относительно низкого уровня сигнала PROG#	–	50	T_{BQ1}
$T_{H(PROG,H-D)}$	T_{PHDX}	Время удержания достоверного слова данных относительно высокого уровня сигнала PROG#	–	50	T_{BQ1}

Продолжение таблицы 15.13

1	2	3	4	5	6
$T_{SU(D-PROG,L)}$	T_{DVPL}	Время установки данных относительно низкого уровня сигнала PROG#	0	–	T_{BQ1}
$T_{H(PROG,L-D)}$	T_{PLDX}	Время удержания данных относительно низкого уровня сигнала PROG#	400	–	T_{BQ1}
$T_{WL(PROG)}$	T_{PLPH}	Длительность импульса PROG#	50 *		T_{BQ1}
$T_{SU(PROG,H-PALE,L)}$	T_{PHLL}	Время установки высокого уровня сигнала PROG# относительно следующего низкого уровня сигнала PALE#	220	–	T_{BQ1}
$T_{SU(PALE,H-PROG,L)}$	T_{LHPL}	Время установки высокого уровня сигнала PALE# относительно низкого уровня сигнала PROG#	220	–	T_{BQ1}
$T_{WH(PROG)}$	T_{PHPL}	Время от высокого PROG# до следующего низкого уровня сигнала PROG#	220	–	T_{BQ1}
$T_{SU(PROG,H-AINC,L)}$	T_{PHIL}	Время установления высокого уровня сигнала PROG# относительно низкого уровня сигнала AINC #	0	–	T_{BQ1}
$T_{WL(AINC)}$	T_{ILIH}	Длительность импульса AINC#	240	–	T_{BQ1}
$T_{H(AINC,L-PVER,L)}$	T_{ILVH}	Время удержания низкого уровня сигнала PVER относительно низкого уровня сигнала AINC#	50	–	T_{BQ1}
$T_{SU(AINC,L-PROG,L)}$	T_{ILPL}	Время установления низкого уровня сигнала AINC# относительно низкого уровня сигнала PROG#	170	–	T_{BQ1}
$T_{SU(PROG,H-PVER,L)}$	T_{PHVL}	Время установления высокого уровня сигнала PROG# относительно действительного низкого уровня PVER	–	220	T_{BQ1}
* Спецификация для режима Word Dump. Для режима программирования значение не менее 500 мкс.					

15.9 Режим автопрограммирования AUTO

Режим автопрограммирования – альтернатива программированию с помощью программатора. Используя этот режим программирования контроллер самостоятельно программирует себя, используя данные из внешней EPROM (адреса 4000_H и старше, таблица 15.14).

Схема включения в режиме автопрограммирования и распределение памяти

На рисунке 15.12 представлена рекомендованная схема для режима автопрограммирования. В таблице 15.14 показывается распределение памяти контроллера. Автопрограммирование определено для частот (6 – 8) МГц. На 8 МГц используют EPROM с $t_{ACC} = 250$ нс и $t_{OE} = 100$ нс или более быстродействующие устройства.

Т а б л и ц а 15.14 – Распределение памяти для режима автопрограммирования

Выходной адрес	Адрес внутреннего EPROM	Адрес в схеме на рисунке 15.12 (A15–0)	Описание
4014 _H	N/A	14 _H	PPW младшего байта
4015 _H	N/A	15 _H	PPW старшего байта
4020 _H –402F _H	2020 _H –202F _H	0020 _H –002F _H	Ключ безопасности
4000 _H –7FFF _H	2000 _H –5FFF _H	4000 _H –7FFF _H	Команды, данные и зарезервированные байты

Операционное окружение

В режиме автопрограммирования байты PCCB загружаются в регистры конфигурации кристалла. Так как устройство получает данные для программирования через внешнюю шину, устройство памяти в программирующей системе должно соответствовать конфигурации по умолчанию (рисунок 15.6). Автопрограммирование требует ширину шины 8 бит, так что в схеме сигнал BUSWIDTH должен быть заземлен. По умолчанию значение PCCB позволяет использовать любую стандартную EPROM, которая удовлетворяет требованиям к временным параметрам, приведенным в ТУ на микросхемы.

В режиме автопрограммирования также загружается байт CCB0 во внутреннюю RAM и проверяются биты защиты. Если любой бит защиты запрограммирован, процедура автопрограммирования сравнивает внутренний ключ безопасности с внешним. Если ключи не совпали, устройство входит в бесконечный внутренний цикл. Если ключи совпали, то процедура продолжается. Процедура автопрограммирования использует модифицированный QUICK-PULSE алгоритм с длительностью программирующих импульсов, записанный во внешнем EPROM.

Алгоритм процедуры автопрограммирования

На рисунке 15.13 представлен алгоритм процедуры программирования в авторежиме. Эта процедура проверяет биты защиты в CCB0; если любой из битов запрограммирован, то происходит сравнение внутреннего ключа безопасности с внешним. Если ключи совпадают, то процедура продолжается, иначе устройство входит в бесконечный цикл.

Если проверка ключа безопасности прошла успешно, то загружается значение длительности программирующих импульсов (PPW) из внешней EPROM во внутренний регистр PPW. После этого устанавливается сигнал PACT#, показывая, что программирование началось. (PACT# также активен во время сброса, хотя процесс программирования не идет.) В начальном состоянии сигнал PVER установлен. В случае обнаружения ошибки программирования, PVER переключится.

После этого процедура считывает данные из внешней EPROM, начиная с адресов 4000_H. При этом пропускаются любые слова, содержащие FFFF_H (незапрограммированное состояние). Если слово отлично от FFFF_H, начинается выполнение модифицированного QUICK-PULSE алгоритма, который записывает это значение в EPROM, после чего проводится проверка записи. Так продолжается до тех пор, пока не запишется вся EPROM, после чего сбрасывается сигнал PACT#, и устройство входит в бесконечный цикл.

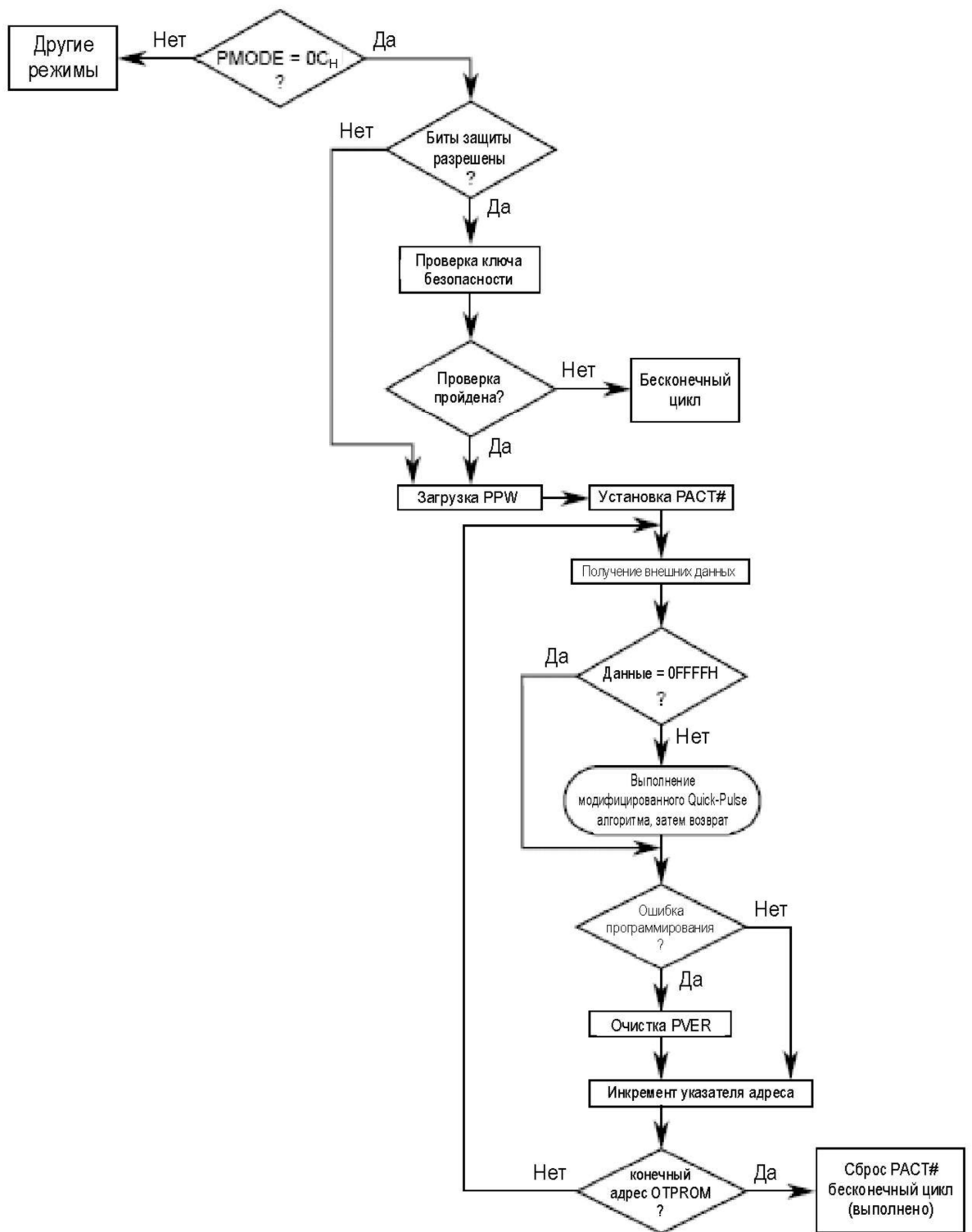


Рисунок 15.13 – Процедура автопрограммирования

Процедура автопрограммирования

В случае сбоя или сброса устройства во время программирования ключа безопасности и битов защиты, дальнейшее программирование невозможно. Чтобы этого не произошло, необходимо соблюдать нижеследующие рекомендации.

1 Используя «чистую» EPROM, необходимо следовать следующим инструкциям, чтобы пропустить программирование ССВ0 и выполнить программирование оставшейся части EPROM, включая ключ безопасности:

- поместить значение длительности программирующего импульса (PPW) во внешнюю EPROM по адресам 14_H и 15_H;
- оставить байт CCB0 во внешней памяти (4018_H) незапрограммированным (0FFFF_H);
- поместить значение CCB1 во внешней памяти по адресу 401A_H;
- поместить ключ безопасности, который будет запрограммирован, во внешней EPROM по адресам 4020_H-402F_H;
- поместить значение 20_H во внешней памяти по адресам 4019_H и 401B_H (это требуется для зарезервированных ячеек EPROM);
- разместить требуемые коды в EPROM по оставшимся адресам 4000_H-7FFF_H;
- выполнить последовательность включения питания, чтобы начать автопрограммирование;
- когда программирование закончено, выполнить последовательность выключения питания перед переходом к шагу 2.

2 Используя другую чистую EPROM, необходимо следовать следующим инструкциям для программирования только CCB0:

- поместить значение длительности программирующего импульса (PPW) во внешнюю EPROM по адресам 14_H и 15_H;
- поместить соответствующее значение CCB0 по адресу 4018_H;
- поместить ключ безопасности во внешней EPROM по адресам 0020_H-002F_H. Это значение должно соответствовать ключу безопасности, запрограммированному в шаге 1;
- оставить остальные ячейки EPROM незапрограммированными (0FFFF_H);
- выполнить последовательность включения питания, чтобы начать автопрограммирование;
- когда программирование закончено, выполнить последовательность выключения питания.

П р и м е ч а н и е – Все адреса приводятся для схемы, показанной на рисунке 15.12. Если используется другая схема, то придется соответственно изменить адреса.

15.10 Режим вывода содержимого памяти ROM-DUMP

Режим ROM-DUMP обеспечивает простой способ вывода содержимого EPROM после автопрограммирования. Используют ту же самую схему, которая используется при автопрограммировании, но с измененным подключением выводов PMODE (P0.7:4). Чтобы выбрать ROM-DUMP режим (PMODE=6_H), надо соединить P0.6 и P0.5 с #VCC и заземлить P0.7 и P0.4. Примерная схема включения (рисунок 15.12) не показывает необходимые подключения WR# и VPR для обеспечения записи в EPROM. И хотя в примере используется EPROM, можно также использовать RAM.

П р и м е ч а н и е – Если запрограммирован бит DED (USFR.2), режим ROM-DUMP невозможен.

Чтобы войти в режим ROM-DUMP, надо следовать рекомендациям по включению питания, приведенным выше. Режим ROM-DUMP проверяет ключ безопасности, независимо от битов защиты CCR. Если запрограммирован ключ безопасности, соответствующий ключ должен находиться во внешней памяти, иначе устройство войдет в бесконечный цикл. Если ключ безопасности пройдет проверку, режим ROM-DUMP фиксирует PPW, после чего вся EPROM записывается во внешнюю память. PACT# остается в низком уровне, пока идет выдача содержимого EPROM, а после ее окончания переключается в высокий уровень.

15.11 Режим RUN-TIME

Можно программировать ячейки EPROM во время обычного выполнения программы. Чтобы сделать EPROM доступной, надо установить на EA# напряжение #VCC во время сброса устройства. Подать напряжение программирования на вывод VPR на время всего процесса программирования. Затем просто записывать информацию в ячейку, которую необходимо запрограммировать.

Примечание – Запись любого бита защиты в CCB0 делает RUN-TIME режим недоступным.

Сразу же после окончания записи EPROM устройство должно войти в IDLE режим или выполнить команду из внешней памяти. Доступ к EPROM прекратит текущий цикл программирования. Каждый цикл программирования начинается, когда слово записывается в EPROM и заканчивается, когда происходит следующее обращение к EPROM. Каждое слово требует пять программирующих циклов, каждый из которых должен быть приблизительно 500 мкс. Ниже приведен пример программы программирования в режиме RUN-TIME:

PROGRAM:

```
PUSHA ;очистить PSW, WSR, INT_MASK, INT_MASK1
LD WSR,#7BH ;выбрать 32-байтное окно с EPA0_CON
LD COUNT,#5 ;задать пять программирующих циклов
ANDB INT_PEND,#CLEAR_EPA0 ;очистить бит задержки прерывания EPA0
LDB INT_MASK,#ENABLE_EPA0 ;разрешить прерывание CAPCOMP0
LDB EPA0_CON,#EPA0_TIMER ;установить CAPCOMP0 как программный таймер
```

LOOP:

```
LD TEMP0,TIMER1 ;загрузить значение TIMER1 в TEMP0
ADD EPA0_TIME,TEMP0,#PGM_PULSE ;загрузить в EPA0_TIME значение
;TIMER1 + PGM_PULSE
EI ;разрешить немаскируемое прерывание (CAPCOMP0)
ST DATA_TEMP,[ADDR_TEMP] ;запомнить данные по адресу
IDLDP #1 ;войти в режим IDLE
DJNZ COUNT,LOOP ;декремент COUNT и переход к метке LOOP если не 0
;для завершения 5 циклов программирования
;восстановить PSW, WSR и INT_MASKs
POPA
RET
```

```
EPA0_ISR:
RET
```

16 Рекомендации по отладочным средствам ИС 1874ВЕ66Т

16.1 Программаторы для программируемого варианта ИС 1874ВЕ66Т

Могут использоваться программаторы любой фирмы, обеспечивающие программирование аналогов разработанной ИС (87С196МД фирмы Intel), например, типа PicProg+, ChipProg, ChipProg+ ООО «Фирма Фитон», г. Москва.

16.2 Описание инструментальных средств для ИС 1874ВЕ66Т

Для проектирования и отладки систем на основе микроконтроллера могут использоваться инструментальные средства, поставляемые ООО «Фирма ФИТОН».

Ниже приводится краткая информация о средствах отладки, поставляемых ООО «Фирма ФИТОН», г. Москва, обеспечивающих отладку систем на базе разработанного микроконтроллера.

Реквизиты ООО «Фирма ФИТОН»:

Адрес: Россия, Москва, 123458, ул. Твардовского, д. 8, стр. 1, кор. Б, оф. 208, ТехноПарк «Строгино»

Тел/факс: (495) 730-75-84(многоканальный)

E-mail: PHYTON@phyton.ru

Сайт: www.phyton.ru

16.3 Интегрированный пакет разработки и отладки систем на базе микроконтроллера (Project-96).

Пакет Project-96 – набор программно-аппаратных средств, предназначенный для разработки и отладки систем на базе микроконтроллеров семейства MCS-196 фирмы Intel, являющихся функциональным аналогом изделий.

Концепция Project-96 – объединение внутрисхемного эмулятора, программного отладчика-симулятора, компиляторов, текстового редактора, менеджера проектов и программатора в рамках единой интеллектуальной среды разработки.

При наличии одного из программаторов PicProg+, ChipProg, ChipProg+ пакет поддерживает работу и с программатором. Программный интерфейс пакета унифицирован и поддерживает все этапы разработки программного обеспечения – от написания исходного текста программы до ее компиляции и отладки.

Пакет Project-96 ориентирован на отладку программ на языке высокого уровня по исходному тексту. Встроенные многооконный редактор, менеджер проектов и большое количество сервисных возможностей существенно облегчают труд разработчика, избавляя его от рутинных инструкций.

Редактор предназначен для написания исходных текстов программ и поддерживает редакцию с блоками текста, поиск/замену, цветовое выделение синтаксических конструкций языка Си и ассемблера.

Встроенный менеджер проектов поддерживает автоматическую компиляцию программ, написанных для компилятора Си и ассемблера. Переход от редактирования исходного текста к отладке и обратно происходит прозрачно, т. е. менеджер проектов автоматически запускает компиляцию изменившихся исходных текстов, активизирует отладчик, осуществляет загрузку программ.

Полная конфигурация пакета называется Project-96/ESCA и включает в себя:

- менеджер проектов;
- кросс-компилятор языка ассемблер MCA-96;
- кросс-компилятор языка «Си» MCC-96;
- отладчик-симулятор PDS-96;
- внутрисхемный эмулятор PICE-196.

Внутрисхемный эмулятор PICE-196

PICE-196 – эмулятор нового поколения, созданный с применением новых технологий разработки аппаратуры и программного обеспечения. Применение программируемых матриц большой емкости позволило значительно сократить размеры эмулятора без какого-либо ущерба его функциональным возможностям, свести к минимуму отличия в электрических и частотных характеристиках эмулятора от характеристик эмулируемого процессора и, тем самым, добиться максимальной точности эмуляции на частотах до 20 МГц при напряжениях питания от 4,5 до 5,5 В.

Программная поддержка PICE-196 работает в среде Windows-95/98/ME/NT/2000/XP и предоставляет пользователю обширный сервис как по разработке программ, так и по их отладке.

Эмулятор состоит из основной платы размером (85×82) мм, сменного пода под определенную группу процессоров и сменного адаптера под конкретный тип корпуса. На основной плате реализованы: трассировщик, процессор точек останова. Плата сменного пода содержит эмулирующий процессор под конкретный тип микроконтроллера. Сменные адаптеры обеспечивают установку эмулятора в колодки PLCC на плате пользователя. Питание эмулятора осуществляется от блока питания 5,0 В, 0,5 А или непосредственно от отлаживаемого устройства. Связь с компьютером – по гальванически развязанному каналу RS-232C на скорости 115К бод.

Характеристики аппаратуры

Точная эмуляция – отсутствие каких-либо ограничений на использование программой пользователя ресурсов микроконтроллера.

До 1 Мб при использовании основной платы М-196/Х (до 128 Кб при использовании основной платы М-196) эмулируемой памяти программ и данных. Поддержка банкированной модели памяти. Распределение памяти между эмулятором и устройством пользователя с точностью до одного байта.

Аппаратная поддержка для отладки программ на языках высокого уровня.

Трассировка восьми произвольных внешних сигналов.

Выходы синхронизации аппаратуры пользователя.

Трассировщик реального времени с буфером объемом 16 Кб фреймов по 64 бита с доступом «на лету». Трассировка адреса, данных, сигналов управления, таймера реального времени и восьми внешних сигналов пользователя.

Программируемый фильтр трассировки.

Аппаратный процессор точек останова с возможностью задания сложного условия останова эмуляции по комбинации сигналов адреса, данных, управления, восьми внешних сигналов, таймера реального времени, счетчиков событий и таймера задержки.

Четыре комплексных точки останова, которые могут быть использованы независимо или в комбинациях по условиям AND/OR/IF-THEN.

48-разрядный таймер реального времени.

Прозрачная эмуляция – доступ «на лету» к эмулируемой памяти, точкам останова, процессору точек останова, буферу трассировки, таймеру реального времени.

Управляемый генератор тактовой частоты для эмулируемого процессора. Возможность плавного изменения тактовой частоты от 500 кГц до 20 МГц.

Гальванически развязанный от компьютера канал связи RS-232C со скоростью обмена 115К бод.

Встроенная система самодиагностики аппаратуры эмулятора.

Характеристики программного обеспечения

Программное обеспечение работает в среде Windows-95/98/ME/NT/2000/XP.

Поддерживается разработка программ на уровне ведения проектов для макроассемблера MCA-96 и Си-компилятора MCC-96 «Фирмы Фитон», а также для пакетов кросс-средств языка «Си» и ассемблера фирм Intel и Tasking Software. Помимо указанных пакетов поддерживается полнофункциональная символьная отладка программ, созданных с помощью компилятора фирмы IAR Systems.

Автоматическое сохранение и загрузка файлов конфигурации аппаратуры, интерфейса и опций отладки. Обеспечивается совместимость файлов конфигурации с симулятором PDS-96. Обеспечена переносимость проектов между эмулятором PICE-196 и симулятором PDS-96.

Возможность настройки цветов, шрифтов и других параметров для всех окон одновременно и для каждого окна в отдельности.

Обновление версий PICE-196 осуществляется обновлением его программного обеспечения.

Наименования компонентов эмулятора PICE-196

Для эмулятора PICE-196 существует два варианта основной платы, различающихся по скорости, объему памяти и, соответственно, по цене. Каждый вариант имеет свое наименование: M-196 или M-196-X. Минимальные параметры и цену обеспечивает базовая для PICE-196 плата M-196.

Название ПОДа состоит из следующих символов (слева направо): «POD» указывает, что это ПОД; «196» – обозначает семейство микроконтроллеров.

Название адаптера состоит из следующих символов (слева направо): «ADP» указывает, что это адаптер; «196» – обозначает семейство микроконтроллеров; «LCC» характеризует тип корпуса эмулируемого микроконтроллера («LCC» обозначает PLCC); «52», «68» и «84» – указывают число выводов корпуса.

Комплект поставки эмулятора PICE-196

В комплект поставки эмулятора PICE-196 входят:

Руководство пользователя и паспорт (гарантийный талон).

Компакт-диск с программным обеспечением и документацией.

Аппаратура эмулятора.

Кабель связи с компьютером (RS-232C).

Трассировочный кабель.

Блок питания.

Упаковочная коробка.

16.4 Отладчик-симулятор микроконтроллеров семейства MCS-196 PDS-96

PDS-96 – это интегрированный комплекс профессиональных средств разработки систем на базе семейства микроконтроллеров MCS-196 фирмы Intel, включающий среду разработки, макроассемблер, отладчик-симулятор, примеры программ и проектов, мощную систему контекстной помощи, электронные гипертекстные руководства по всем компонентам пакета, а также краткое руководство пользователя в печатном виде. PDS-96 работает в среде Windows-95/98/ME/NT/2000/XP.

С помощью PDS-96 можно эффективно разрабатывать и отлаживать программы, используя не только входящий в комплект макроассемблер MCA-96, но и Си-компилятор MCC-96 «Фирмы Фитон», а также кросс-средства фирм Intel и Tasking Software, для которых также предоставляется возможность разработки программ на уровне ведения проектов. Помимо указанных пакетов, PDS-96 обеспечивает полнофункциональную символьную отладку программ, созданных с помощью пакета кросс-средств фирмы IAR Systems. Пользователю предоставляется обширный сервис по выполнению отлаживаемой программы в различных режимах, манипуляции различными типами точек останова, просмотру и модификации состояния ресурсов микроконтроллера. Поддерживается

отладка программ по исходному тексту, а также просмотр и изменение значений сложных объектов языка высокого уровня – массивов, структур, указателей.

Среда разработки программ PDS-96 интегрирует в себе средства, используемые при разработке программ для микроконтроллеров MCS-196. Обеспечивается интерактивная поддержка всех этапов разработки от написания исходного текста до зашивки готовой программы в ПЗУ микроконтроллера, а именно:

- написание исходных текстов программ с помощью встроенного многооконного редактора;

- настройка опций кросс-средств, используемых для компиляции программы (ассемблера, компилятора Си, линкера, библиотеки). Настройка производится с помощью диалогов, снабженных контекстной справочной информацией;

- компиляция и линковка программы. Если компилятор обнаруживает ошибки в исходном тексте программы, то строка с ошибкой в окне редактора подсвечивается, и ошибки можно сразу же исправить;

- отладка программы;

- «зашивка» программы в ПЗУ микроконтроллера.

«Интегрированность» среды PDS-96 проявляется в том, что перечисленные этапы разработки связываются в одно целое. Самые трудоемкие этапы, а именно компиляция/линковка с диагностикой и исправлением ошибок, максимально упрощены. PDS-96 самостоятельно следит за изменениями, которые вносятся в исходные тексты программ. Например, исправив ошибку в исходном тексте, можно нажатием одной кнопки выполнить программу «до курсора», заставить PDS-96 перетранслировать изменившиеся модули, загрузить полученную программу в память отладчика и запустить ее до указанной строки. Переход от отладки к редактированию происходит так же прозрачно и быстро.

Отладочные возможности PDS-96

Симулятор PDS-96 представляет собой программно-логическую модель микроконтроллера, имитирующую (симулирующую) работу всех его узлов – памяти, АЛУ, системы команд, регистров и т. д.

Возможности PDS-96:

- отслеживание выполнения программы по ее исходному тексту;

- просмотр и изменение значений любых переменных;

- встроенный анализатор эффективности программного кода;

- точки останова по сложному условию;

- неограниченное количество точек останова по доступу к ячейкам памяти;

- просмотр стека вызовов подпрограмм и функций;

- встроенный строчный ассемблер;

- возможность выполнения программы «назад» на большое количество шагов, а также в непрерывном режиме. При этом состояние модели микроконтроллера полностью восстанавливается;

- точный подсчет интервалов времени и многое другое.

Основные достоинства программно-логической модели микроконтроллера, реализованной в PDS-96, – точная симуляция узлов микроконтроллера и возможность моделировать устройства, подключенные к микроконтроллеру «снаружи» (т. н. моделирование внешней среды), например, внешнюю логику, датчики, клавиатуру, исполнительные устройства (дисплеи), задавать периодические и непериодические воздействия и т. п.

16.5 Кросс-макроассемблер МСА-96

Кросс-макроассемблер МСА-96 предназначен для трансляции исходных текстов программ для процессоров семейства MCS-196 фирмы Intel:

- поддерживает все микроконтроллеры MCS-196 фирмы Intel;
- генерирует HEX-файл и подробный листинг;
- поддерживает широкий набор директив условной трансляции;
- предоставляет удобные средства работы с макросами;
- генерирует подробную символьную информацию для отладчиков;
- допускает использование русских букв в именах;
- поддерживает 32-битные арифметические и логические выражения;
- выполняет проверку перекрытия кода;
- выполняет проверку размещения данных в запрещенных областях;
- включает полный набор include-файлов;
- поставляется как в составе пакета Project-96, так и отдельно.

Макроассемблер МСА-96 поддерживает процессоры семейства Intel MCS-196, включая кристаллы с 24-битной адресацией. Имеется возможность расширять номенклатуру поддерживаемых процессоров без обновления версии ассемблера. В комплект входит набор включаемых файлов, содержащий определения регистров спецназначения для всех ОЭВМ MCS-196 фирмы Intel.

Использование русских букв в именах позволяет создавать исходные тексты программ, обладающие превосходной читаемостью. Генерируется подробный листинг, включающий не только текст программы и адреса инструкций, но также и таблицы символов, макросов, констант и т. п. с указанием имен, к которым не было ссылок в программе.

16.6 Кросс-компилятор языка Си МСС-96

Кросс-компилятор языка «Си» МСС-96 предназначен для трансляции исходных текстов программ для процессоров семейства MCS-196:

- поддерживаются все кристаллы MCS-19;
- соответствует стандарту ANSI/ISO 9899-1990;
- генерирует быстрый код;
- три модели памяти;
- встроенный ассемблер;
- поддержка всей специфики MCS-196 из «Си»;
- быстрая библиотека функций с плавающей точкой;
- большая библиотека стандартных функций (более 120);
- функции для работы с потоками ввода-вывода;
- поддержка для динамически распределяемой памяти;
- поставляется в составе Project-96.

«Си» – исключительно гибкий язык, реализующий концепцию структурного программирования и обладающий богатым набором инструкций. В «Си» удачно совмещены как высокоуровневые абстракции – модульность, процедурность, читабельность исходного текста, так и низкоуровневые средства – работа с абсолютными адресами, встроенный ассемблер, работа с битами. Кроме этого, «Си» позволяет получить эффективно работающий код. Именно эти особенности делают «Си» идеальным для встроенных приложений, где требуется доступ ко всем ресурсам процессора при наличии высокоуровневого синтаксиса. Выполнен в соответствии со стандартом ANSI, поэтому можно в полной мере пользоваться свойством переносимости «Си»-программ, используя уже готовые и отлаженные алгоритмы. МСС-96 поддерживает все ОЭВМ Intel MCS-196, включая кристаллы с 24-битной адресацией. Для полного использования всех возможностей MCS-196 в язык введены необходимые расширения. Встроенный ассемблер

дает возможность написания макросов с параметрами на ассемблере и их использования в качестве inline-функций. Все особенности архитектуры MCS-196 поддерживаются непосредственно из «Си». Например, подпрограммы обслуживания прерываний можно писать на «Си». Вся поддержка такого рода реализована при помощи стандартных директив `#pragma`, поэтому получающийся исходный текст хорошо переносим на другие типы процессоров.

Библиотека компилятора оптимизирована для исполнения на ОЭВМ MCS-196 и содержит более 120 функций, включая инструкции с потоками, форматированный ввод-вывод и поддержку для динамически распределяемой памяти (HEAP). Благодаря оптимизированным алгоритмам, инструкции над числами с плавающей точкой производятся в 3 – 4 раза быстрее, чем у компиляторов фирм Intel и Tasking, причем без потери точности. В комплект компилятора входит набор включаемых файлов, содержащий определения регистров специальных функций для всех ОЭВМ семейства MCS-196, в том числе и для ИС 1874BE66T.

17 Заключение

В настоящем руководстве КФДЛ.431295.033 приведено подробное описание архитектуры, функционального построения, системы команд и особенностей применения ИС 1874BE66Т, которые представляют собой СБИС однокристалльного 16-разрядного микроконтроллера с функцией управления электродвигателями, с внутренней программной памятью объемом 16 Кбайт типа EPROM (однократно программируемая память типа EPROM).

Микроконтроллер предназначен для применения в системах управления и спецприменениях, для управления робототехническими комплексами, в системах автоматизации технологических процессов, в системах автоматизированного управления электроприводом, оргтехнике, вычислительной технике, телекоммуникационной технике и т. д. Его применение позволит обеспечить требуемые тактико-технические данные систем В и ВТ и выполнить требования ТЗ на аппаратуру (комплексы) по назначению, энергопотреблению и массогабаритным показателям. Это даст возможность обеспечить комплектование аппаратуры специального назначения отечественными ИС взамен аналогичных импортных изделий.

КФДЛ.431295.033 может служить практическим пособием по применению микроконтроллеров для разработчиков систем на основе ИС 1874BE66Т и программистов.

Приложение А (обязательное)

Система команд

В приложении дана информация о системе команд микроконтроллера 1874BE66T, описана каждая команда, показана связь между командами и флагами PSW, приведены шестнадцатеричные коды команд, длины команд и время их выполнения.

В таблице А.1 даны переменные, используемые в таблице А.2 для замены операндов команд.

В таблице А.2 приведён список команд в алфавитном порядке и описание каждой из команд.

В таблицах А.3 и А.4 определены аббревиатура и символы, используемые в таблицах А.5 и А.6.

В таблице А.5 показывается влияние каждой команды на флаги PSW.

В таблице А.6 показано влияние флагов PSW или соответствующих битов регистра на команды условного перехода.

В таблице А.7 приводится список шестнадцатеричных кодов команд вместе с соответствующей мнемоникой.

В таблице А.8 приводится карта кодов команд.

В таблице А.9 приводятся длины и коды команд для каждого используемого режима адресации.

В таблице А.10 приводится время выполнения команд, измеряемое в машинных циклах.

Т а б л и ц а А.1 – Переменные, используемые в операндах

Пере- менные	Описание
1	2
aa	Двухбитное поле внутри кода команды, определяющее основной используемый режим адресации. Это поле имеется только в кодах, в которых проводится выбор режима адресации. Поле декодируется следующим образом: 0 0 Прямая регистровая адресация 0 1 Непосредственная адресация 1 0 Косвенная адресация 1 1 Индексная адресация
baop	Однобайтный операнд, адресуются к которому любым способом
bbb	Трёхбитное поле внутри кода команды, определяющее выбор определённого бита внутри регистра
bitno	Трёхбитное поле внутри кода команды, определяющее выбор одного из восьми битов внутри байта
breg	Однобайтный регистр во внутреннем регистровом файле. Когда не ясно, ссылается ли эта переменная на источник или приёмник, переменная помечается соответственно S или D
cadd	Адрес в программном коде
Dbreg *	Однобайтный регистр во внутреннем регистровом файле, использующийся как операнд-приемник в команде
disp	Смещение между концом команды и целевой меткой в программе
Dwreg *	Регистр слова (двухбайтный) во внутреннем регистровом файле, использующийся как операнд-приемник в команде. Должен выравниваться по адресу, делящемуся нацело на два

Окончание таблицы А.1

1	2
Ireg	32-битный регистр во внутреннем регистровом файле. Должен выравниваться по адресу, делящемуся нацело на четыре
Sbreg *	Однobaйтный регистр во внутреннем регистровом файле, обслуживающий операнд-источник в команде
Swreg *	Регистр слова (двухбайтовый) во внутреннем регистровом файле, обслуживающий операнд-источник в команде. Должен выравниваться по адресу, кратному двум
waop	Двухбайтный операнд, к которому адресуются любым способом
wreg	Двухбайтный регистр во внутреннем регистровом файле. Когда не ясно, ссылается ли эта переменная на источник или приёмник, переменная помечается соответственно S или D. Должен выравниваться по адресу, кратному двум
xxx	Три старших бита смещения
* Когда не ясно, ссылается ли эта переменная на источник или приёмник, переменная помечается соответственно S или D.	

Т а б л и ц а А.2 – Система команд

Обозначение команды	Операция	Формат команды
1	2	3
ADD (2 операнда)	«Сложение слов». Сложение слов источника и приёмника и сохранение суммы в приёмнике. $(DEST) \leftarrow (DEST) + (SRC)$	DEST, SRC ADD wreg, waop (011001aa)(waop)(wreg)
ADD (3 операнда)	«Сложение слов». Сложение слов двух источников и сохранение суммы в приёмнике. $(DEST) \leftarrow (SRC1) + (SRC2)$	DEST, SRC1, SRC2 ADD Dwreg, swreg, waop (010001aa)(waop)(Swreg)(Dwreg)
ADDB (2 операнда)	«Сложение байтов». Сложение байтов источника и приёмника и сохранение суммы в приёмнике (левый операнд). $(DEST) \leftarrow (DEST) + (SRC)$	DEST, SRC ADDB breg, baop (011101aa)(baop)(breg)
ADDB (3 операнда)	«Сложение байтов». Сложение байтов двух источников и сохранение суммы в приёмнике. $(DEST) \leftarrow (SRC1) + (SRC2)$	DEST, SRC1, SRC2 ADDB Dbreg, Sbreg, baop (010101aa)(baop)(breg)
ADDC	«Сложение слов с переносом». Сложение слов источника, приёмника, флага переноса (0 или 1) и сохранение суммы в приёмнике. $(DEST) \leftarrow (DEST) + (SRC) + C$	DEST, SRC ADDC wreg, waop (101001aa)(waop)(wreg)
ADDCB	«Сложение байтов с переносом». Сложение байтов источника, приёмника, флага переноса (0 или 1) и сохранение суммы в приёмнике. $(DEST) \leftarrow (DEST) + (SRC) + C$	DEST, SRC ADDCB breg, baop (101101aa)(baop)(breg)

Продолжение таблицы А.2

1	2	3
<p>AND (2 операнда)</p>	<p>«Логическое И слов». Операция И над словами источника и приёмника и сохранение результата в приёмнике. Результат равен единице в соответствующем бите, если оба операнда в этом бите имеют «1»; «0» – в остальных случаях. (DEST) ← (DEST)AND(SRC)</p>	<p>DEST, SRC AND wreg, waop (011000aa)(waop)(wreg)</p>
<p>ANDB (3 операнда)</p>	<p>«Логическое И байтов». Операция И над байтами 2-х источников и сохранение результата в приёмнике. Результат – «1» в соответствующем бите, если оба операнда в этом бите имеют «1»; «0» – в остальных случаях. (DEST) ← (SRC1)AND(SRC2)</p>	<p>DEST, SRC1, SRC2 AND Dbreg, Sbreg, baop (010100aa)(baop)(Sbreg)(Dbreg)</p>
<p>BMOV</p>	<p>«Перемещение блоков». Перемещение блоков данных типа word из одной зоны памяти в другую. Адреса источника и приёмника вычисляются при помощи режима косвенной адресации с автоинкрементом. Длинный регистр (Ireg) содержит указатели источника и приёмника, которые хранятся в соседних регистрах слов. Регистр слова Wreg (CNTREG) определяет количество перемещений. Блоки данных могут находиться в любом месте регистрового ОЗУ, но не должны перекрываться. COUNT ← (CNTREG) LOOP: SRCPTR ← (PTRS) DSTPTR ← (PTRS+2) (DSTPTR) ← (SRCPTR) (PTRS) ← SRCPTR+2 (PTRS+2) ← DSTPTR+2 COUNT ← COUNT- 1 if COUNT ≠ 0 then go to LOOP end if</p>	<p>DEST, SRC BMOV Ireg, wreg (11000001)(wreg)(Ireg)</p> <p>Примечание – CNTREG (Wreg) не декрементируется в процессе операции. Легко неумышленно создать длительную непрерываемую операцию с командой BMOV. Для создания прерываемых операций используется команда BMOVI</p>

Продолжение таблицы А.2

1	2	3
BMOVI	<p>«Прерываемое перемещение блоков».</p> <p>Перемещение блоков данных типа word из одной зоны памяти в другую. Команда идентична BMOV, исключая то, что BMOVI – прерываема. Адреса источника и приёмника вычисляются при помощи косвенного режима адресации с автоинкрементом. Длинный регистр (Ireg) адресуется к указателям источника и приёмника, которые хранятся в соседних регистрах слов. Регистр слов Wreg (CNTREG) определяет количество перемещений. Блоки данных могут находиться в любом месте регистрового ОЗУ, но не должны перекрываться.</p> <p>COUNT ← (CNTREG) LOOP: SRCPTR ← (PTRS) DSTPTR ← (PTRS+2) (DSTPTR) ← (SRCPTR) (PTRS) ← SRCPTR+2 (PTRS+2) ← DSTPTR+2 COUNT ← COUNT- 1 if COUNT ≠ 0 then go to LOOP end if</p>	<p>DEST, SRC BMOVI Ireg,wreg (11001101)(wreg)(Ireg)</p> <p>Примечание – CNTREG (Wreg) не декрементируется, если выполнение команды не было прервано. Если BMOVI прервана, то в CNTREG сохраняется значение, бывшее в нём во время прерывания. По этой причине необходимо перезагружать CNTREG перед началом выполнения операции BMOVI</p>
BR	<p>«Косвенное разветвление».</p> <p>Продолжает выполнение с адреса, определяемого операндом регистра слова.</p> <p>PC ← (DEST)</p>	<p>DEST BR [wreg] (11100011)(wreg)</p>
CLR	<p>«Очистка слова».</p> <p>Обнуляет значение операнда.</p> <p>(DEST) ← 0</p>	<p>DEST CLR wreg (00000001)(wreg)</p>
CLRB	<p>«Очистка байта».</p> <p>Обнуляет значение операнда.</p> <p>(DEST) ← 0</p>	<p>DEST CLRB breg (00010001)(breg)</p>
CLRC	<p>«Очистка флага переноса».</p> <p>Обнуляет флаг переноса.</p> <p>C ← 0</p>	<p>CLRC (11111000)</p>
CLRVT	<p>«Очистка дополнительного флага переполнения».</p> <p>Обнуляет флаг VT</p> <p>VT ← 0</p>	<p>CLRVT (11111100)</p>
CMP	<p>«Сравнение слов».</p> <p>Вычитает слово источника из слова приёмника. Флаги устанавливаются, но операнды остаются прежними. Если имеется заём, флаг переноса равен «0», иначе – «1».</p> <p>(DEST) - (SRC)</p>	<p>DEST, SRC CMP wreg, waop (100010aa)(waop)(wreg)</p>

Продолжение таблицы А.2

1	2	3
СМРВ	«Сравнение байтов». Вычитает байт источника из байта приёмника. Флаги устанавливаются, но операнды остаются прежними. Если имеется заём, флаг переноса равен «0», иначе – «1». (DEST) - (SRC)	DEST, SRC СМРВ breg, baop (100110aa)(baop)(breg)
СМРЛ	«Сравнение чисел типа LONG». Сравнение величин двух операндов типа double-word (long). Операнды определяются с использованием режима прямой адресации. Флаги устанавливаются, но операнды остаются прежними. Если имеется заём, флаг переноса равен «0», иначе – «1». (DEST) - (SRC)	DEST, SRC СМРЛ Ireg, Ireg (11000101)(src Ireg)(destIreg)
ДЕС	«Декремент слова». Декрементирует величину операнда на единицу. (DEST) <- (DEST)-1	DEST ДЕС wreg (00000101)(wreg)
ДЕСВ	«Декремент байта». Декрементирует величину операнда на единицу. (DEST) <- (DEST)-1	DEST ДЕСВ breg (00010101)(breg)
ДИ	«Запрещение прерываний». Запрещает прерывания. Запросы на прерывания не удовлетворяются после этой команды. Interrupt Enable (PSW.1) <- 0	ДИ (11111010)
ДИВ	«Деление чисел типа INTEGER». Делит содержимое приёмника - операнд типа LONG-INTEGЕR на содержимое источника - операнд типа INTEGER, используя знаковую арифметику. Частное сохраняет в младшем слове (т. е. в слове с меньшим адресом) приёмника, а остаток – в старшем слове. (low word DEST) <- (DEST)/(SRC) (high word DEST) <- (DEST) MOD (SRC)	DEST, SRC ДИВ Ireg, waop (11111110)(100011aa)(waop)(Ireg)
ДИВВ	«Деление чисел типа SHORT INTEGER». Делит содержимое приёмника – операнд типа INTEGER на содержимое источника – операнд типа SHORTINTEGЕR, используя знаковую арифметику. Частное сохраняет в младшем байте (т. е. в байте с меньшим адресом) приёмника, а остаток – в старшем байте. (low word DEST) <- (DEST)/(SRC) (high word DEST) <- (DEST)MOD(SRC)	DEST, SRC ДИВВ wreg, baop (11111110)(100111aa)(baop)(wreg)

Продолжение таблицы А.2

1	2	3
DIVU	<p>«Деление чисел типа WORD, незнаковых».</p> <p>Делит содержимое приёмника - операнд типа DOUBLEWORD на содержимое источника операнд слова типа WORD, используя беззнаковую арифметику. Частное сохраняет в младшем слове (т. е. в слове с меньшим адресом) приёмника, а остаток – в старшем слове. Следующие две операции проводятся одновременно.</p> <p>(low word DEST) <- (DEST)/(SRC) (high word DEST) <- (DEST)MOD(SRC)</p>	<p>DEST, SRC DIVU Ireg, waop (100011aa)(waop)(Ireg)</p>
DIVUB	<p>«Деление чисел типа WORD, незнаковых».</p> <p>Делит содержимое приёмника – операнд типа WORD на содержимое источника – операнд типа BYTE, используя беззнаковую арифметику. Частное сохраняет в младшем байте (т. е. в байте с меньшим адресом) приёмника, а остаток – в старшем байте. Следующие две операции проводятся одновременно.</p> <p>(low byte DEST) <- (DEST)/(SRC) (high byte DEST) <- (DEST)MOD(SRC)</p>	<p>DEST, SRC DIVUB wreg, baop (100111aa)(baop)(wreg)</p>
DJNZ	<p>«Декремент и переход, если не равно нулю».</p> <p>Декрементирует величину операнда типа BYTE на единицу. Если результат равен нулю, управление передаётся следующей по порядку команде. Если результат не равен нулю, команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>(COUNT) <- (COUNT) - 1 if (COUNT) ≠ 0 then PC <- PC + disp¹⁾</p>	<p>DJNZ breg, cadd (11100000)(breg)(disp)</p>
DJNZW	<p>«Декремент и переход, если не равно нулю».</p> <p>Декрементирует величину операнда типа WORD на единицу. Если результат равен нулю, управление передаётся следующей по порядку команде. Если результат не равен нулю, команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>(COUNT) <- (COUNT) - 1 if (COUNT) ≠ 0 then PC <- PC + disp¹⁾</p>	<p>DJNZW wreg, cadd (11100001)(wreg)(disp)</p>

Продолжение таблицы А.2

1	2	3
DPTS	«Блокирование периферийного сервера – PTS». Блокирует PTS. PTS Disable (PSW.2) <– 0	DPTS (11101100)
EI	«Разрешение прерываний». Разрешает прерывания, запрашиваемые после выполнения следующего оператора. Запросы на прерывания не могут удовлетворяться немедленно после выполнения этой команды. Interrupt Enable (PSW.1) <– 1	EI (11111011)
EPTS	«Разблокировка периферийного сервера – PTS». Разблокирует PTS. PTS Enable (PSW.2) <– 1	EPTS (11101101)
EXT	«Знаковое расширение INTEGER в LONG-INTEGER». Расширяет знаком младшее слово операнда до двойного слова. if (low word DEST) < 8000 _H then (high word DEST) <– 0 else (high word DEST) <– 0FFFF _H end if	EXT Ireg (00000110)(Ireg)
EXTB	«Знаковое расширение SHORTINTEGER в INTEGER». Расширяет со знаком младший байт операнда до слова. if (low byte DEST) < 80 _H then (high byte DEST) <– 0 else (high byte DEST) <– 0FF _H end if	EXTB wreg (00010110)(wreg)
IDLPD	«Холостой ход/Пониженное потребление». В зависимости от 8-битной величины операнда KEY эта команда выбирает: - вход в режим холостого хода (IDLE) (KEY=1); - вход в режим пониженного потребления (POWERDOWN) (KEY=2); - выполнить последовательность сброса (любое другое значение KEY, не равное единице или двум). Контроллер шины завершает цикл упреждающей выборки перед остановкой CPU или сбросом. if KEY=1 – режим IDLE, if KEY=2 – режим POWERDOWN, другие значения – сброс	IDLPD #key (11110110)(key)
INC	«Инкремент слова». Увеличение значение слова операнда на «1». (DEST) <– (DEST)+1	INC wreg (00000111)(wreg)
INCB	«Инкремент байта». Инкрементирует байт операнда на «1». (DEST) <– (DEST)+1	INCB breg (00010111)(breg)

Продолжение таблицы А.2

1	2	3
JBC	<p>«Переход, если бит очищен».</p> <p>Тестирует определённый бит. Если бит установлен, управление передаётся следующей по порядку команде. Если бит обнулен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>if (specified bit) = 0 then PC ← PC + disp ¹⁾</p>	JBC breg, bitno, cadd (00110bbb)(breg)(disp)
JBS	<p>«Переход, если бит установлен».</p> <p>Тестирует определённый бит. Если бит обнулен, управление передаётся следующей по порядку команде. Если бит установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>if (specified bit) = 1 then PC ← PC + disp ¹⁾</p>	JBS breg, bitno, cadd (00111bbb)(breg)(disp)
JC	<p>«Переход при установленном флаге переноса».</p> <p>Тестирует С-флаг переноса. Если флаг переноса обнулен, управление передаётся следующей по порядку команде. Если флаг переноса установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>if C=1 then PC ← PC + disp ¹⁾</p>	JC cadd (11011011)(disp)
JE	<p>«Переход при равенстве нулю флага нулевого результата».</p> <p>Тестирует Z-флаг нулевого результата. Если флаг обнулен, управление передаётся следующей по порядку команде. Если флаг установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>if Z=1 then PC ← PC + disp ¹⁾</p>	JE cadd (11011111)(disp)

Продолжение таблицы А.2

1	2	3
JGE	<p>«Переход, если знак больше или равен». Тестирует N-флаг отрицательного результата. Если флаг установлен, управление передаётся следующей по порядку команде. Если флаг N обнулен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>if N=0 then PC ← PC + disp¹⁾</p>	JGE cadd (11010110)(disp)
JGT	<p>«Переход, если знак больше». Тестирует Z-флаг нуля и N-флаг отрицательного результата. Если один из флагов установлен, управление передаётся следующей по порядку команде. Если оба флага очищены, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>if N=0 AND Z=0 then PC ← PC + disp¹⁾</p>	JGT cadd (11010010)(disp)
JH	<p>«Переход, если больше (беззнаковый)». Тестируются флаги нуля и переноса. Если флаг переноса обнулен или флаг нуля установлен, управление передаётся следующей по порядку команде. Если флаг переноса установлен, и флаг нуля очищен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>if C=1 AND Z=0 then PC ← PC + disp¹⁾</p>	JH cadd (11011001)(disp)
JLE	<p>«Переход, если знак меньше или равен». Тестирует N-флаг отрицательного результата и C-флаг переноса. Если оба флага очищены, управление передаётся следующей по порядку команде. Если один флаг установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>if N=1 OR Z=1 then PC ← PC + disp¹⁾</p>	JLE cadd (11011010)(disp)

Продолжение таблицы А.2

1	2	3
JLT	<p>«Переход, если знак меньше».</p> <p>Тестирует N-флаг отрицательного результата. Если флаг очищен, управление передаётся следующей по порядку команде. Если флаг установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>if N=1 then PC ← PC + disp¹⁾</p>	JLT cadd (11011110)(disp)
JNC	<p>«Переход, если флаг переноса очищен».</p> <p>Тестирует C-флаг переноса. Если флаг установлен, управление передаётся следующей по порядку команде. Если флаг очищен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>if C=0 then PC ← PC + disp¹⁾</p>	JNC cadd (11010011)(disp)
JNE	<p>«Переход, если не равно».</p> <p>Тестирует Z-флаг нулевого результата. Если флаг установлен, управление передаётся следующей по порядку команде. Если флаг очищен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>if =0 then PC ← PC + disp¹⁾</p>	JNE cadd (11010111)(disp)
JNH	<p>«Переход, если не больше (беззнаковый)».</p> <p>Тестирует флаг нулевого результата и флаг переноса. Если флаг переноса установлен, и флаг нулевого результата очищен, управление передаётся следующей по порядку команде. Если флаг переноса установлен или флаг нуля установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127.</p> <p>if C=0 OR Z=1 then PC ← PC + disp¹⁾</p>	JNE cadd (11010001)(disp)

Продолжение таблицы А.2

1	2	3
JNST	<p>«Переход, если флаг ST очищен». Тестирует флаг ST (sticky bit). Если флаг установлен, управление передаётся следующей по порядку команде. Если флаг очищен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if ST=0 then PC ← PC + disp¹⁾</p>	JNST cadd (11010000)(disp)
JNV	<p>«Переход, если флаг переполнения очищен». Тестирует V-флаг переполнения. Если флаг установлен, управление передаётся следующей по порядку команде. Если флаг очищен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if V=0 then PC ← PC + disp¹⁾</p>	JNV cadd (11010101)(disp)
JNVT	<p>«Переход, если дополнительный флаг переполнения очищен». Тестирует флаг VT. Если флаг установлен, эта команда очищает флаг, и управление передаётся следующей по порядку команде. Если флаг очищен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if VT=0 then PC ← PC + disp¹⁾</p>	JNVT cadd (11010100)(disp)
JST	<p>«Переход, если флаг ST установлен». Тестируется флаг ST. Если флаг очищен, управление передаётся следующей по порядку команде. Если флаг ST установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if ST=1 then PC ← PC + disp¹⁾</p>	JST cadd (11011000)(disp)

Продолжение таблицы А.2

1	2	3
JV	«Переход, если установлен флаг переполнения». Тестируется V-флаг переполнения. Если флаг переполнения очищен, управление передаётся следующей по порядку команде. Если флаг переполнения установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if V=1 then PC ← PC + disp ¹⁾	JV cadd (11011101)(disp)
JVT	«Переход, если дополнительный флаг переполнения установлен». Тестируется флаг VT. Если флаг очищен, управление передаётся следующей по порядку команде. Если флаг установлен, то команда добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Смещение может быть в диапазоне от минус 128 до плюс 127. if VT=1 then PC ← PC + disp ¹⁾	JST cadd (11011100)(disp)
LCALL	«Длинный вызов». Загружает содержимое программного счётчика (адрес возврата) в стек, затем добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Операнд может иметь любой адрес во всём адресном пространстве. SP ← SP - 2 (SP) ← PC PC ← PC + disp (16 бит)	LCALL cadd (11101111)(disp-low)(disp-high)
LD	«Загрузка слова». Загружает значение слова источника в приёмник. (DEST) ← (SRC)	DEST, SRC LD wreg, waop (101000aa)(waop)(wreg)
LDB	«Загрузка байта». Загружает значение байта источника в приёмник. (DEST) ← (SRC)	DEST, SRC LDB breg, baop (101100aa)(baop)(breg)
LDBSE	«Загрузка байта со знаковым расширением». Расширяет знаком операнд источника типа SHORT- INTEGER и загружает его в приёмник типа INTEGER. (low byte DEST) ← (SRC) if (SRC) < 80 _H then (high byte DEST) ← 0 else (high byte DEST) ← 0FF _H	DEST, SRC LDBSE wreg, baop (101111aa)(baop)(wreg)

Продолжение таблицы А.2

1	2	3
LDBZE	«Загрузка байта, расширенного нулём». Значение операнда источника типа BYTE расширяется нулём и загружается в приёмник типа WORD. (low byte DEST) ← (SRC) (high byte DEST) ← 0	DEST, SRC LDBZE wreg, baop (101011aa)(baop)(wreg)
LJMP	«Длинный переход». Добавляет к программному счётчику смещение между концом этой команды и меткой перехода. Операнд может иметь любой адрес во всём адресном пространстве. PC ← PC + disp	LJMP cadd (11100111)(disp-low)(disp-high)
MUL (2 операнда)	«Умножение чисел типа INTEGER». Перемножает операнды источника и приёмника типа INTEGER, используя знаковую арифметику, и сохраняет 32-битный результат в приёмнике типа LONG INTEGER. После выполнения этой команды флаг ST не определен. (DEST) ← (DEST) × (SRC)	DEST, SRC MUL Ireg, waop (11111110)(010111aa)(waop)(Ireg)
MUL (3 операнда)	«Умножение чисел типа INTEGER». Перемножает операнды двух источников типа INTEGER, используя знаковую арифметику, и сохраняет 32-битный результат в приёмнике типа LONG INTEGER. После выполнения этой команды флаг ST не определен. (DEST) ← (SRC1) × (SRC2)	DEST, SRC1, SRC2 MUL Ireg, wreg, waop (11111110)(010011aa)(waop)(wreg)(Ireg)
MULB (2 операнда)	«Умножение чисел типа SHORT INTEGER». Перемножает операнды источника и приёмника типа SHORT INTEGER, используя знаковую арифметику, и сохраняет 16-битный результат в приёмнике типа INTEGER. После выполнения этой команды флаг ST не определен. (DEST) ← (DEST) × (SRC)	DEST, SRC MULB wreg, baop (11111110)(011111aa)(baop)(wreg)
MULB (3 операнда)	«Умножение чисел типа SHORT INTEGER». Перемножает операнды двух источников типа SHORT INTEGER, используя знаковую арифметику, и сохраняет 16-битный результат в приёмнике типа INTEGER. После выполнения этой команды флаг ST не определен. (DEST) ← (SRC1) × (SRC2)	DEST, SRC1, SRC2 MULB wreg, breg, baop (11111110)(010111aa)(baop)(breg)(wreg)

Продолжение таблицы А.2

1	2	3
MULU (2 операнда)	«Умножение чисел типа WORD, беззнаковое». Перемножает операнды источника и приёмника типа WORD, используя беззнаковую арифметику, и сохраняет 32-битный результат в приёмнике типа DOUBLE-WORD. После выполнения этой команды флаг ST не определен. (DEST) ← (DEST) × (SRC)	DEST, SRC MULU Ireg, waop (011011aa)(waop)(Ireg)
MULU (3 операнда)	«Умножение чисел типа WORD, беззнаковое». Перемножает операнды двух источников типа WORD, используя беззнаковую арифметику, и сохраняет 32-битный результат в приёмнике типа DOUBLE-WORD. После выполнения этой команды флаг ST не определен. (DEST) ← (SRC1) × (SRC2)	DEST, SRC1, SRC2 MULU Ireg, wreg, waop (010011aa)(waop)(wreg) (Ireg)
MULUB (2 операнда)	«Умножение чисел типа BYTE, беззнаковое». Перемножает операнды источника и приёмника типа BYTE, используя беззнаковую арифметику, и сохраняет результат типа WORD в приёмнике. После выполнения этой команды флаг ST не определен. (DEST) ← (DEST) × (SRC)	DEST, SRC MULUB wreg, baop (011111aa)(baop)(wreg)
MULUB (3 операнда)	«Умножение чисел типа BYTE, беззнаковое». Перемножает операнды двух источников типа BYTE, используя беззнаковую арифметику, и сохраняет результат типа WORD в приёмнике. После выполнения этой команды флаг ST не определен. (DEST) ← (SRC1) × (SRC2)	DEST, SRC1, SRC2 MULUB wreg, breg, baop (010111aa)(baop)(breg)(wreg)
NEG	«Изменение знака числа типа INTEGER». Изменяет знак операнда типа INTEGER. (DEST) ← (DEST)	NEG wreg (00000011)(wreg)
NEGB	«Изменение знака числа типа SHORT INTEGER». Изменяет знак операнда типа SHORT INTEGER. (DEST) ← (DEST)	NEGB breg (00010011)(breg)
NOP	«Нет операции». Не выполняет никаких действий. Управление переходит к следующей по порядку команде	NOP (11111101)

Продолжение таблицы А.2

1	2	3
NORML	<p>«Нормализация числа типа LONG INTEGER».</p> <p>Нормализует операнд источника типа LONG INTEGER (левый операнд). Эта команда сдвигает операнд влево до тех пор, пока его старший значащий бит – единица или пока не будет совершен 31 сдвиг. Если в старшем значащем бите остался «0» после 31 сдвига, команда останавливает процесс и устанавливает флаг Z. Команда сохраняет число совершённых сдвигов в приёмнике (правый операнд)</p> <p>(COUNT) ← 0 do while (MSB(DEST) = 0) AND (COUNT) < 31 (DEST) ← (DEST) × 2 (COUNT) ← (COUNT) + 1</p>	<p>SRC, DEST NORML Ireg, breg (00001111)(breg)(Ireg)</p>
NOT	<p>«Инверсия числа типа WORD».</p> <p>Инвертирует значение операнда типа word (заменяет каждую «1» на «0» и каждый «0» на «1»)</p> <p>(DEST) ← NOT(DEST)</p>	<p>NOT wreg (00000010)(wreg)</p>
NOTB	<p>«Инверсия числа типа BYTE».</p> <p>Инвертирует значение операнда типа byte (заменяет каждую «1» на «0» и каждый «0» на «1»)</p> <p>(DEST) ← NOT(DEST)</p>	<p>NOTB breg (00010010)(breg)</p>
OR	<p>«Логическое ИЛИ слов».</p> <p>Проводит операцию ИЛИ над операндом источника типа word и операндом приёмника типа word, и меняет первоначальное значение приёмника на результат. Результат равен единице в каждом разряде, в котором хотя бы один из операндов имеет 1.</p> <p>(DEST) ← (DEST)OR(SRC)</p>	<p>DEST, SRC OR breg, waop (100000aa)(waop)(breg)</p>
ORB	<p>«Логическое ИЛИ байтов».</p> <p>Проводит операцию ИЛИ над операндом источника типа byte и операндом приёмника типа byte, и меняет первоначальное значение приёмника на результат. Результат равен единице в каждом разряде, в котором хотя бы один из операндов имеет «1».</p> <p>(DEST) ← (DEST)OR(SRC)</p>	<p>DEST, SRC ORB breg, baop (100100aa)(baop)(breg)</p>
POP	<p>«Чтение слова из стека».</p> <p>Читает слово из вершины стека и помещает его в приёмник.</p> <p>(DEST) ← (SP) SP ← SP + 2</p>	<p>POP waop (110011aa)(waop)</p>

Продолжение таблицы А.2

1	2	3
<p>POPA</p>	<p>«Чтение стека». Эта команда используется взамен POPF для поддержки восьми добавочных прерываний. Она читает два слова из стека и помещает первое слово в пару регистров INT_MASK1/WSR и второе слово – в PSW/INT_MASK. Эта команда инкрементирует указатель стека SP на четыре. Вызовы прерываний не могут осуществляться непосредственно после этой команды. INT_MASK1/WSR ← (SP) SP ← SP + 2 PSW/INT_MASK ← (SP) SP ← SP + 2</p>	<p>POPA (11110101)</p>
<p>POPF</p>	<p>«Чтение флагов из стека». Читает слово из вершины стека и помещает его в PSW. Вызовы прерываний не могут осуществляться непосредственно после этой команды. PSW/INT_MASK ← (SP) SP ← SP + 2</p>	<p>POPF (11110011)</p>
<p>PUSH</p>	<p>«Загрузка слова в стек». Загружает операнд типа word в стек. SP ← SP – 2 (SP) ← (DEST)</p>	<p>PUSH waop (110010aa)(waop)</p>
<p>PUSHA</p>	<p>«Загрузка в стек». Эта команда используется взамен PUSHF для поддержки восьми добавочных прерываний. Она загружает два слова из регистровых пар PSW/INT_MASK и INT_MASK1/WSR в стек. Команда очищает вышеуказанные регистры (кроме WSR) и уменьшает указатель стека на четыре. Вызовы прерываний не могут осуществляться сразу после этой команды. SP ← SP-2 (SP) ← PSW/INT_MASK PSW/INT_MASK ← 0 SP ← SP-2 (SP) ← INT_MASK1/WSR INT_MASK1 ← 0</p>	<p>PUSHA (11110100)</p>
<p>PUSHF</p>	<p>«Загрузка флагов в стек». Загружает PSW в вершину стека, затем очищает его. Очистка PSW запрещает обслуживание прерываний. Вызовы прерываний не могут осуществляться сразу после этой команды. SP ← SP – 2 (SP) ← PSW/INT_MASK PSW/INT_MASK ← 0</p>	<p>PUSHF (11110010)</p>

Продолжение таблицы А.2

1	2	3
RET	«Возврат из подпрограммы». Загружает программный счётчик (PC) из вершины стека. PC ← (SP) SP ← SP + 2	RET (11110000)
RST	«Сброс системы». Инициализирует PSW в 0, PC в 2080h, а SFR – в их значения по сбросу. Выполнение этой команды ведёт к тому, что вывод RESET# находится в низком состоянии 16 машинных циклов. SFR ← Reset Status Pin ← Reset Status PSW ← 0 PC ← 2080 _H	RST (11111111)
SCALL	«Короткий вызов». Загружает содержимое PC (адрес возврата) в стек, затем добавляет к PC смещение между концом этой команды и меткой. Смещение должно быть в пределах от минус 1024 до плюс 1023 включительно. SP ← SP – 2 (SP) ← PC PC ← PC + disp (11 бит) ¹⁾	SCALL cadd (00101xxx)(disp-low)
SETC	«Установка флага переноса». Устанавливает флаг переноса. C ← 1	SETC (11111001)
SHL	«Сдвиг слова влево». Сдвигает операнд типа word приёмника влево столько раз, сколько установлено операндом счёта. Значение операнда может быть задано непосредственно в пределах от нуля до 15 (0F _H) включительно или как содержимое какого-либо регистра (10 _H –0FF _H) со значением в пределах от нуля до 31 (1F _H) включительно. Правые биты результата заполняются нулями. Последний сдвинутый бит хранится во флаге переноса. Temp ← (COUNT) do while Temp ≠ 0 C ← High order bit of (DEST) (DEST) ← (DEST) × 2 Temp ← Temp – 1 end_while	SHL wreg, #count (00001001)(count)(wreg) или SHL wreg, breg (00001001)(breg)(wreg)

Продолжение таблицы А.2

1	2	3
SHLB	<p>«Сдвиг байта влево».</p> <p>Сдвигает операнд типа byte приёмника влево столько раз, сколько установлено операндом счёта. Значение операнда может быть задано непосредственно в пределах от нуля до 15 (0F_H) включительно или как содержимое регистра (10_H – 0FF_H) со значением в пределах от 0 до 31 (1F_H) включительно. Правые биты результата заполняются нулями. Последний сдвинутый бит хранится во флаге переноса.</p> <p>Temp ← (COUNT) do while Temp ≠ 0 C ← High order bit of (DEST) (DEST) ← (DEST) × 2 Temp ← Temp – 1 end_while</p>	<p>SHLB breg, #count (00011001)(count)(breg) или SHLB breg, breg (00011001)(breg)(breg)</p>
SHLL	<p>«Сдвиг числа типа DOUBLE-WORD влево».</p> <p>Сдвигает операнд типа double-word приёмника влево столько раз, сколько установлено специальным операндом счётчиком. Значение операнда может быть задано непосредственно в пределах от нуля до 15 (0F_H) включительно или как содержимое регистра (10_H – 0FF_H) со значением в пределах от нуля до 31 (1F_H) включительно. Правые биты результата заполняются нулями. Последний сдвинутый бит хранится во флаге переноса.</p> <p>Temp ← (COUNT) do while Temp ≠ 0 C ← High order bit of (DEST) (DEST) ← (DEST) × 2 Temp ← Temp – 1 end_while</p>	<p>SHLL Ireg, #count (00001101)(count)(breg) или SHLL Ireg, breg (00001101)(breg)(Ireg)</p>

Продолжение таблицы А.2

1	2	3
SHR	<p>«Логический сдвиг слова вправо».</p> <p>Сдвигает операнд типа word приёмника вправо столько раз, сколько установлено операндом счётчика. Значение операнда может быть задано непосредственно в пределах от нуля до 15 (0F_H) включительно или как содержимое какого-либо регистра (10_H – 0FF_H) со значением в пределах от нуля до 31 (1F_H) включительно. Левые биты результата заполняются нулями. Последний сдвинутый бит хранится во флаге переноса. Команда сначала очищает флаг ST. Если в какой-то момент времени в течение сдвига «1» сдвигается во флаг переноса и осуществляется другой цикл сдвига, команда устанавливает флаг ST.</p> <pre>Temp ← (COUNT) do while Temp ≠ 0 C ← Low order bit of (DEST) (DEST) ← (DEST)/ 2² Temp ← Temp – 1 end while</pre>	<p>SHR wreg, #count (00001000)(count)(wreg) или SHR wreg, breg (00001000)(breg)(wreg)</p>
SHRA	<p>«Арифметический сдвиг слова вправо».</p> <p>Сдвигает операнд типа word приёмника вправо столько раз, сколько установлено специальным операндом счёта. Значение операнда может быть задано непосредственно в пределах от нуля до 15 (0F_H) включительно или как содержимое какого-либо регистра (10_H – 0FF_H) со значением в пределах от нуля до 31 (1F_H) включительно. Если значение старшего бита было «0», сдвигаются нули. Если значение было «1», сдвигаются единицы. Последний сдвинутый бит хранится во флаге переноса. Команда сначала очищает флаг ST. Если в какой-то момент времени в течение сдвига «1» сдвигается во флаг переноса и осуществляется другой цикл сдвига, команда устанавливает флаг ST.</p> <pre>Temp ← (COUNT) do while Temp ≠ 0 C ← Low order bit of (DEST) (DEST) ← (DEST)/ 2³ Temp ← Temp – 1 end while</pre>	<p>SHRA wreg, #count (00001010)(count)(wreg) или SHRA wreg, breg (00001010)(breg)(wreg)</p>

Продолжение таблицы А.2

1	2	3
SHRAB	<p>«Арифметический сдвиг байта вправо». Сдвигает операнд типа byte приёмника вправо столько раз, сколько установлено операндом счёта. Значение операнда может быть задано непосредственно в пределах от нуля до 15 (0F_H) включительно или как содержимое какого-либо регистра (10_H – 0FF_H) со значением в пределах от нуля до 31 (1F_H) включительно. Если величина старшего бита была «0», сдвигаются нули. Если величина была «1», сдвигаются единицы. Последний сдвинутый бит хранится во флаге переноса. Команда сначала очищает флаг ST. Если в какой-то момент времени в течение сдвига «1» сдвигается во флаг переноса и осуществляется другой цикл сдвига, команда устанавливает флаг ST.</p> <p>Temp ← (COUNT) do while Temp ≠ 0 C ← Low order bit of (DEST) (DEST) ← (DEST)/ 2³ Temp ← Temp - 1</p>	<p>SHRAB breg, #count (00011010)(count)(breg) или SHRAB breg, breg (00011010)(breg)(breg)</p>
SHRAL	<p>«Арифметический сдвиг вправо числа типа DOUBLE-WORD». Сдвигает операнд типа double word приёмника вправо столько раз, сколько установлено операндом счёта (правый операнд). Значение операнда может быть задано непосредственно в пределах от нуля до 15 (0F_H) включительно или как содержимое какого-либо регистра (10_H – 0FF_H) со значением в пределах от нуля до 31 (1F_H) включительно. Если величина старшего бита была «0», сдвигаются нули. Если величина была «1», сдвигаются единицы. Команда сначала очищает флаг ST. Если в какой-то момент времени в течение сдвига «1» сдвигается во флаг переноса и осуществляется другой цикл сдвига, команда устанавливает флаг ST.</p> <p>Temp ← (COUNT) do while Temp ≠ 0 C ← Low order bit of (DEST) (DEST) ← (DEST)/ 2³ Temp ← Temp - 1</p>	<p>SHRAL Ireg, #count (00001110)(count)(Ireg) или SHRAL Ireg, breg (00001110)(breg)(Ireg)</p>

Продолжение таблицы А.2

1	2	3
SHRB	<p>«Логический сдвиг байта вправо».</p> <p>Сдвигает операнд типа byte приёмника вправо столько раз, сколько установлено операндом счёта. Значение операнда может быть задано непосредственно в пределах от 0 до 15 (0F_H) включительно или как содержимое какого-либо регистра (10_H – 0FF_H) со значением в пределах от нуля до 31 (1F_H) включительно. Левые биты результата заполняются нулями. Последний сдвинутый бит хранится во флаге переноса. Команда сначала очищает флаг ST. Если в какой-то момент времени в течение сдвига «1» сдвигается во флаг переноса и осуществляется другой цикл сдвига, команда устанавливает флаг ST.</p> <p>Temp ← (COUNT) do while Temp ≠ 0 C ← Low order bit of (DEST) (DEST) ← (DEST)/ 2²) Temp ← Temp – 1</p>	<p>SHRB breg, #count (00011000)(count)(breg) или SHRB breg, breg (00011000)(breg)(breg)</p>
SHRL	<p>«Логический сдвиг вправо числа типа DOUBLE-WORD». Сдвигает операнд типа double-word приёмника вправо столько раз, сколько установлено специальным операндом счёта. Значение операнда может быть задано непосредственно в пределах от нуля до 15 (0F_H) включительно или как содержимое какого-либо регистра регистра (10_H – 0FF_H) со значением в пределах от нуля до 31 (1F_H) включительно. Левые биты результата заполняются нулями. Последний сдвинутый бит хранится во флаге переноса. Команда сначала очищает флаг ST. Если в какой-то момент времени в течение сдвига «1» сдвигается во флаг переноса и осуществляется другой цикл сдвига, команда устанавливает флаг ST.</p> <p>Temp ← (COUNT) do while Temp ≠ 0 C ← Low order bit of (DEST) (DEST) ← (DEST)/ 2²) Temp ← Temp - 1</p>	<p>SHRL Ireg, #count (00001100)(count)(Ireg) или SHRL Ireg, breg (00001100)(breg)(Ireg)</p>
SJMP	<p>«Короткий переход».</p> <p>Добавляет к PC смещение между концом этой команды и меткой перехода. Смещение может быть в пределах от минус 1024 до плюс 1023 включительно.</p> <p>PC ← PC + disp (11 бит)¹⁾</p>	<p>SJMP cadd (00100xxx)(disp-low)</p>

Продолжение таблицы А.2

1	2	3
SKIP	«Нет операции, двухбайтная». Нет действий. Управление передаётся следующей по порядку команде. Это двухбайтная NOP, где второй байт – любая величина, которая просто игнорируется	SKIP breg (00000000)(breg)
ST	«Сохранение слова». Загружает величину операнда источника типа word (левый операнд) в приёмник (правый операнд). (DEST) ← (SRC)	SRC, DEST ST wreg, waop (110000aa)(waop)(wreg)
STB	«Сохранение байта». Сохраняет величину операнда источника типа byte (левый операнд) в приёмнике (правый операнд). (DEST) ← (SRC)	SRC, DEST STB breg, baop (110001aa)(baop)(breg)
SUB (2 операнда)	«Вычитание слов». Вычитает операнд источника типа word из операнда приёмника типа word, сохраняет результат в приёмнике и устанавливает флаг переноса как дополнение заёма. (DEST) ← (DEST) – (SRC)	DEST, SRC SUB wreg, waop (011010aa)(waop)(wreg)
SUB (3 операнда)	«Вычитание слов». Вычитает операнд первого источника типа word из операнда второго источника, сохраняет результат в приёмнике и устанавливает флаг переноса как дополнение заёма. (DEST) ← (SRC1) – (SRC2)	DEST, SRC1, SRC2 SUB Dwreg, Swreg, waop (010010aa)(waop)(Swreg)(Dwreg)
SUBB (2 операнда)	«Вычитание байтов». Вычитает операнд источника типа byte из операнда приёмника типа byte, сохраняет результат в приёмнике и устанавливает флаг переноса как дополнение заёма. (DEST) ← (DEST) – (SRC)	DEST, SRC SUBB breg, baop (010110aa)(baop)(breg)
SUBB (3 операнда)	«Вычитание байтов». Вычитает операнд первого источника типа byte из операнда второго источника, сохраняет результат в приёмнике и устанавливает флаг переноса как дополнение заёма. (DEST) ← (SRC1) – (SRC2)	DEST, SRC1, SRC2 SUBB Dbreg, Sbreg, baop (010110aa)(baop)(Sbreg)(Dbreg)
SUBC	«Вычитание слов с заёмом». Вычитает операнд источника типа word из операнда приёмника типа word. Если флаг переноса очищен, SUBC вычитает 1 из результата. Сохраняет результат в приёмнике и устанавливает флаг переноса как дополнение заёма. (DEST) ← (DEST) – (SRC) – (1 – C)	DEST, SRC SUBC wreg, waop (101010aa)(waop)(wreg)

Продолжение таблицы А.2

1	2	3
SUBCB	<p>«Вычитание байтов с заёмом».</p> <p>Вычитает операнд источника типа byte из операнда приёмника типа byte. Если флаг переноса очищен, SUBC вычитает единицу из результата. Сохраняет результат в приёмнике и устанавливает флаг переноса как дополнение заёма.</p> $(DEST) \leftarrow (DEST) - (SRC) - (1 - C)$	<p>DEST, SRC</p> <p>SUBCB breg, baop (101110aa)(baop)(breg)</p>
TIJMP	<p>«Табличный косвенный переход».</p> <p>Вызывает продолжение выполнения программы по адресу, выбранному из таблицы адресов. Первый регистр типа word, TBASE, содержит 16-битный адрес начала таблицы. TBASE может быть размещён в регистровом ОЗУ до адреса FEH (нет режима окон) или выше FFH (режим окон). Таблица переходов может размещаться в любых незарезервированных ячейках памяти, выровненных по границе слова. Второй регистр типа word, INDEX, содержит 16-битный адрес регистра, который содержит 7-битную величину. Эта величина используется для вычисления смещения в таблице переходов. INDEX размещается в регистровом ОЗУ аналогично TBASE. 16-разрядный адрес в INDEX абсолютен, не изменяется при выполнении команды с использованием режима окон. Байтовый операнд #MASK – 7-битная непосредственная величина для маскирования регистра INDEX. Логическое AND над #MASK и INDEX определяет смещение (OFFSET). OFFSET умножается на 2, выравнивается по границе слова, затем добавляется к базовому адресу (TBASE) для определения адреса приёмника (DEST X).</p> $[INDEX] \text{ AND } \#MASK = \text{OFFSET}$ $(2 \times \text{OFFSET}) + \text{TBASE} = \text{DEST X}$ $PC \leftarrow (\text{DEST X})$	<p>TIJMP wreg, [wreg], #byte (11100010)(wreg)(#byte)(wreg)</p>

Окончание таблицы А.2

1	2	3
TRAP	<p>«Программное прерывание».</p> <p>Эта команда осуществляет вызов прерывания, вектор которого расположен в 2010_H. Эта операция не влияет на состояние флага разрешения прерываний (I) в PSW. Вызовы прерываний не могут следовать сразу после этой команды.</p> <p>SP ← SP – 2 (SP) ← PC PC ← (2010_H)</p> <p>Примечание – Эта команда не поддерживается ассемблером. Команда TRAP предназначена для использования инструментальными средствами разработки. Эти средства могут не поддерживать применение пользователем этой команды</p>	TRAP (11110111)
XCH	<p>«Обмен слов».</p> <p>Меняет значение операнда источника типа word со значением операнда приёмника типа word.</p> <p>(DEST) ← (SRC); (SRC) ← (DEST)</p>	DEST, SRC XCH wreg, waop (00000100)(waop)(wreg) (00001011)(waop)(wreg)
XCHB	<p>«Обмен байтов».</p> <p>Меняет значение операнда источника типа byte со значением операнда приёмника типа byte.</p> <p>(DEST) ← (SRC); (SRC) ← (DEST)</p>	DEST, SRC XCHB breg, baop (00010100)(baop)(breg) (00011011)(baop)(breg)
XOR	<p>«Логическое “исключающее ИЛИ” слов».</p> <p>Выполняет операцию «исключающее ИЛИ» над операндами типа word источника и приёмника и сохраняет результат в приёмнике. Результат имеет «1» в битовой позиции, если один из операндов (но не оба) имеет «1» в этой позиции и нули при всех других значениях битовых позиций.</p> <p>(DEST) ← (DEST)XOR(SRC)</p>	DEST, SRC XOR wreg, waop (100001aa)(waop)(wreg)
XORB	<p>LOGICAL EXCLUSIVE-OR BYTES (Логическое “исключающее ИЛИ” байтов).</p> <p>Выполняет операцию «исключающее ИЛИ» над операндами типа byte источника и приёмника и сохраняет результат в приёмнике. Результат имеет «1» в битовой позиции, если один из операндов (но не оба) имеет «1» в этой позиции и нули при всех других значениях битовых позиций.</p> <p>(DEST) ← (DEST)XOR(SRC)</p>	DEST, SRC XORB breg, baop (100101aa)(baop)(breg)
<p>¹⁾ Смещение (disp) расширяется до 16 бит знаком.</p> <p>²⁾ В этой операции DEST/2 – результат деления без знака.</p> <p>³⁾ В этой операции DEST/2 – результат деления со знаком.</p>		

В таблицах А.3 и А.4 приведены аббревиатуры и символы, используемые в таблицах А.5 и А.6. В таблице А.5 показано влияние каждой команды на флаги слова состояния программы (PSW), а в таблице А.6 показывается влияние флагов PSW или специального регистрового бита на команды условного перехода.

Т а б л и ц а А.3 – Обозначение флагов PSW

Обозначение	Название флага PSW
C	Carry Flag – Флаг Переноса
N	Negative Flag – Флаг Отрицательного Результата
ST	Sticky Bit Flag – Флаг “Дополнительного Бита”
V	Overflow Flag – Флаг Переполнения
VT	Overflow-Trap Flag – Дополнительный Флаг Переполнения
Z	Zero Flag – Флаг Нуля

Т а б л и ц а А.4 – Символы установки флагов PSW

Символ	Описание
+	Команда устанавливает или сбрасывает флаг по результату операции
–	Команда не модифицирует флаг
↓	Команда сбрасывает флаг как определено, но не устанавливает
↑	Команда устанавливает флаг как определено, но не сбрасывает
1	Команда устанавливает флаг
0	Команда сбрасывает флаг
?	Команда оставляет флаг в неопределённом состоянии

Т а б л и ц а А.5 – Влияние команд на установку флагов PSW

Обозначение	Установка флага PSW					
	Z	N	C	V	VT	ST
1	2	3	4	5	6	7
ADD, ADDB	+	+	+	+	↑	–
ADDC, ADDCB	↓	+	+	+	↑	–
AND, ANDB	+	+	0	0	–	–
BMOV, BMOVI	–	–	–	–	–	–
BR (косвенный)	–	–	–	–	–	–
CLR, CLRB	1	0	0	0	–	–
CLRC	–	–	0	–	–	–
CLRVT	–	–	–	–	0	–
CMP, CMPB	+	+	+	+	↑	–
CMPL	+	+	+	+	+	–
DEC, DECB	+	+	+	+	↑	–
DI	–	–	–	–	–	–
DIV, DIVB, DIVU, DIVUB	–	–	–	+	↑	–
DJNZ, DJNZW	–	–	–	–	–	–
DPTS	–	–	–	–	–	–
EI	–	–	–	–	–	–
EPTS	–	–	–	–	–	–
EXT, EXTB	+	+	0	0	–	–
IDLPD	–	–	–	–	–	–
правильный параметр	0	0	0	0	0	0
неверный параметр	0	0	0	0	0	0
INC	+	+	+	+	↑	0
INCB	+	+	+	+	↑	–

Окончание таблицы А.5

1	2	3	4	5	6	7
JBC, JBS, JC, JE, JGE, JGT, JH, JLE, JLT, JNC, JNE, JNH, JNST, JNV	–	–	–	–	–	–
JNVT	–	–	–	–	0	–
JST, JV	–	–	–	–	–	–
JVT	–	–	–	–	0	–
LCALL, LD, LDB, LDBSE, LDBZE	–	–	–	–	–	–
LJMP	–	–	–	–	–	?
MUL, MULB, MULU, MULUB	–	–	–	–	–	?
NEG, NEGB	+	+	+	+	↑	–
NOP	–	–	–	–	–	–
NORML	+	?	0	–	–	–
NOT, NOTB	+	+	0	0	–	–
OR, ORB	+	+	0	0	–	–
POP	–	–	–	–	–	–
POPA, POPF	+	+	+	+	+	+
PUSH	–	–	–	–	–	–
PUSHA, PUSHF	0	0	0	0	0	0
RET	–	–	–	–	–	–
RST	0	0	0	0	0	0
SCALL	–	–	–	–	–	–
SETC	–	–	1	–	–	–
SHL, SHLB, SHLL	+	?	+	+	↑	–
SHR	+	0	+	0	–	+
SHRA, SHRAB, SHRAL	+	+	+	0	–	+
SHRB, SHRL	+	0	+	0	–	+
SJMP	–	–	–	–	–	–
SKIP	–	–	–	–	–	–
ST, STB	–	–	–	–	–	–
SUB, SUBB	+	+	+	+	↑	–
SUBC, SUBCB	↓	+	+	+	↑	–
TIJMP	–	–	–	–	–	–
TRAP	–	–	–	–	–	–
XCH, XCHB	–	–	–	–	–	–
XOR, XORB	+	+	0	0	–	–

Т а б л и ц а А.6 – Влияние флагов PSW или тестируемых битов на команды условного перехода

Команда	Переход осуществляется, если	Продолжается выполнение, если
1	2	3
DJNZ	декрементированный байт $\neq 0$	декрементированный байт = 0
DJNZW	декрементированное слово $\neq 0$	декрементированное слово = 0
JBC	выбранный регистровый бит = 0	выбранный регистровый бит = 1
JBS	выбранный регистровый бит = 1	выбранный регистровый бит = 0
JNC	C = 0	C = 1
JNH	C = 0 OR Z = 1	C = 1 AND Z = 0
JC	C = 1	C = 0
JH	C = 1 AND Z = 0	C = 0 OR Z = 1
JGE	N = 0	N = 1

Окончание таблицы А.6

1	2	3
JGT	$N = 0 \text{ AND } Z = 0$	$N = 1 \text{ OR } Z = 1$
JLT	$N = 1$	$N = 0$
JLE	$N = 1 \text{ OR } Z = 1$	$N = 0 \text{ AND } Z = 0$
JNST	$ST = 0$	$ST = 1$
JST	$ST = 1$	$ST = 0$
JNV	$V = 0$	$V = 1$
JV	$V = 1$	$V = 0$
JNVT	$VT = 0$	$VT = 1$ (очищает VT)
JVT	$VT = 1$ (очищает VT)	$VT = 0$
JNE	$Z = 0$	$Z = 1$
JE	$Z = 1$	$Z = 0$

В таблице А.7 приводится список кодов команд по возрастанию, с соответствующими обозначениями команд.

Т а б л и ц а А.7 – Коды команд

Шестнадцатеричный код	Обозначение команды
1	2
00	SKIP
01	CLR
02	NOT
03	NEG
04	XCH
05	DEC
06	EXT
07	INC
08	SHR
09	SHL
0A	SHRA
0B	XCH
0C	SHRL
0D	SHLL
0E	SHRAL
0F	NORML
10	Зарезервирован
11	CLRB
12	NOTB
13	NEGB
14	XCHB
15	DECB
16	EXTB
17	INCB
18	SHRB
19	SHLB
1A	SHRAB
1B	XCHB
1C-1F	Зарезервирован
20-27	SJMP

Продолжение таблицы А.7

1	2
28-2F	SCALL
30-37	JBC
38-3F	JBS
40	AND с прямой адресацией (3 операнда)
41	AND с непосредственной адресацией (3 операнда)
42	AND с косвенной адресацией (3 операнда)
43	AND с индексной адресацией (3 операнда)
44	ADD с прямой адресацией (3 операнда)
45	ADD с непосредственной адресацией (3 операнда)
46	ADD с косвенной адресацией (3 операнда)
47	ADD с индексной адресацией (3 операнда)
48	SUB с прямой адресацией (3 операнда)
49	SUB с непосредственной адресацией (3 операнда)
4A	SUB с косвенной адресацией (3 операнда)
4B	SUB с индексной адресацией (3 операнда)
4C	MULU с прямой адресацией (3 операнда)
4D	MULU с непосредственной адресацией (3 операнда)
4E	MULU с косвенной адресацией (3 операнда)
4F	MULU с индексной адресацией (3 операнда)
50	ANDB с прямой адресацией (3 операнда)
51	ANDB с непосредственной адресацией (3 операнда)
52	ANDB с косвенной адресацией (3 операнда)
53	ANDB с индексной адресацией (3 операнда)
54	ADDB с прямой адресацией (3 операнда)
55	ADDB с непосредственной адресацией (3 операнда)
56	ADDB с косвенной адресацией (3 операнда)
57	ADDB с индексной адресацией (3 операнда)
58	SUBB с прямой адресацией (3 операнда)
59	SUBB с непосредственной адресацией (3 операнда)
5A	SUBB с косвенной адресацией (3 операнда)
5B	SUBB с индексной адресацией (3 операнда)
5C	MULUB с прямой адресацией (3 операнда)
5D	MULUB с непосредственной адресацией (3 операнда)
5E	MULUB с косвенной адресацией (3 операнда)
5F	MULUB с индексной адресацией (3 операнда)
60	AND с прямой адресацией (3 операнда)
61	AND с непосредственной адресацией (3 операнда)
62	AND с косвенной адресацией (3 операнда)
63	AND с индексной адресацией (3 операнда)
64	ADD с прямой адресацией (3 операнда)
65	ADD с непосредственной адресацией (3 операнда)
66	ADD с косвенной адресацией (3 операнда)
67	ADD с индексной адресацией (3 операнда)
68	SUB с прямой адресацией (3 операнда)
69	SUB с непосредственной адресацией (3 операнда)
6A	SUB с косвенной адресацией (3 операнда)
6B	SUB с индексной адресацией (3 операнда)
6C	MULU с прямой адресацией (3 операнда)
6D	MULU с непосредственной адресацией (3 операнда)
6E	MULU с косвенной адресацией (3 операнда)

Продолжение таблицы А.7

1	2
6F	MULU с индексной адресацией (3 операнда)
70	ANDB с прямой адресацией (3 операнда)
71	ANDB с непосредственной адресацией (3 операнда)
72	ANDB с косвенной адресацией (3 операнда)
73	ANDB с индексной адресацией (3 операнда)
74	ADDB с прямой адресацией (3 операнда)
75	ADDB с непосредственной адресацией (3 операнда)
76	ADDB с косвенной адресацией (3 операнда)
77	ADDB с индексной адресацией (3 операнда)
78	SUBB с прямой адресацией (3 операнда)
79	SUBB с непосредственной адресацией (3 операнда)
7A	SUBB с косвенной адресацией (3 операнда)
7B	SUBB с индексной адресацией (3 операнда)
7C	MULUB с прямой адресацией (3 операнда)
7D	MULUB с непосредственной адресацией (3 операнда)
7E	MULUB с косвенной адресацией (3 операнда)
7F	MULUB с индексной адресацией (3 операнда)
80	OR с прямой адресацией
81	OR с непосредственной адресацией
82	OR с косвенной адресацией
83	OR с индексной адресацией
84	XOR с прямой адресацией
85	XOR с непосредственной адресацией
86	XOR с косвенной адресацией
87	XOR с индексной адресацией
88	СМР с прямой адресацией
89	СМР с непосредственной адресацией
8A	СМР с косвенной адресацией
8B	СМР с индексной адресацией
8C	DIVU с прямой адресацией
8D	DIVU с непосредственной адресацией
8E	DIVU с косвенной адресацией
8F	DIVU с индексной адресацией
90	ORB с прямой адресацией
91	ORB с непосредственной адресацией
92	ORB с косвенной адресацией
93	ORB с индексной адресацией
94	XORB с прямой адресацией
95	XORB с непосредственной адресацией
96	XORB с косвенной адресацией
97	XORB с индексной адресацией
98	СМРВ с прямой адресацией
99	СМРВ с непосредственной адресацией
9A	СМРВ с косвенной адресацией
9B	СМРВ с индексной адресацией
9C	DIVUB с прямой адресацией
9D	DIVUB с непосредственной адресацией
9E	DIVUB с косвенной адресацией
9F	DIVUB с индексной адресацией
A0	LD с прямой адресацией
A1	LD с непосредственной адресацией

Продолжение таблицы А.7

1	2
A2	LD с косвенной адресацией
A3	LD с индексной адресацией
A4	ADDC с прямой адресацией
A5	ADDC с непосредственной адресацией
A6	ADDC с косвенной адресацией
A7	ADDC с индексной адресацией
A8	SUBC с прямой адресацией
A9	SUBC с непосредственной адресацией
AA	SUBC с косвенной адресацией
AB	SUBC с индексной адресацией
AC	LDBZE с прямой адресацией
AD	LDBZE с непосредственной адресацией
AE	LDBZE с косвенной адресацией
AF	LDBZE с индексной адресацией
B0	LDB с прямой адресацией
B1	LDB с непосредственной адресацией
B2	LDB с косвенной адресацией
B3	LDB с индексной адресацией
B4	ADDCB с прямой адресацией
B5	ADDCB с непосредственной адресацией
B6	ADDCB с косвенной адресацией
B7	ADDCB с индексной адресацией
B8	SUBCB с прямой адресацией
B9	SUBCB с непосредственной адресацией
BA	SUBCB с косвенной адресацией
BB	SUBCB с индексной адресацией
BC	LDBSE с прямой адресацией
BD	LDBSE с непосредственной адресацией
BE	LDBSE с косвенной адресацией
BF	LDBSE с индексной адресацией
C0	ST с прямой адресацией
C1	BMOV
C2	ST с косвенной адресацией
C3	ST с индексной адресацией
C4	STB с прямой адресацией
C5	CMPL
C6	STB с косвенной адресацией
C7	STB с индексной адресацией
C8	PUSH с прямой адресацией
C9	PUSH с непосредственной адресацией
CA	PUSH с косвенной адресацией
CB	PUSH с индексной адресацией
CC	POP с прямой адресацией
CD	BMOVI
CE	POP с косвенной адресацией
CF	POP с индексной адресацией
D0	JNST
D1	JNH
D2	JGT

Окончание таблицы А.7

1	2
D3	JNC
D4	JNVT
D5	JNV
D6	JGE
D7	JNE
D8	JST
D9	JH
DA	JLE
DB	JC
DC	JVT
DD	JV
DE	JLT
DF	JE
E0	DJNZ
E1	FJNZW
E2	TIJMP
E3	BR (Indirect)
E4-E6	Зарезервирован
E7	LJMP
E8-EB	Зарезервирован
EC	DPTS
ED	EPTS
EE	Зарезервирован *
EF	LCALL
F0	RET
F1	Зарезервирован
F2	PUSHF
F3	POPF
F4	PUSHA
F5	POPA
F6	IDLPD
F7	TRAP
F8	CLRC
F9	SETC
FA	DI
FB	EI
FC	CLRVT
FD	NOP
FE	DIV/DIVB/MUL/MULB **
FF	RST

* Код EE зарезервирован; однако он не генерирует прерывания невыполнимого кода.

** Знаковое умножение и деление, 2-байтные команды. Для любой знаковой команды первый байт – «FE_H», второй – код соответствующей беззнаковой команды. Например: код MULU (3 операнда) с прямой индексацией «4C_H», поэтому код MUL (3 операнда) с прямой индексацией – «FE4C_H».

В таблице А.8 дана карта кодов команд микроконтроллера. Первая цифра кода – команды по вертикали, вторая – по горизонтали. Соответствующее обозначение команды показано на пересечении двух чисел. Метод адресации: di – прямая, im – непосредственная, in – косвенная, ix – индексная.

Т а б л и ц а А.8 – Карта кодов

Код	x0	x1	x2	x3	x4	x5	x6	x7
0x	SKIP	CLR	NOT	NEG	XCH di	DEC	EXT	INC
1x	*	CLRB	NOTB	NEGB	XCHB di	DECB	EXTB	INCB
2x	SJMP							
3x	JBC							
	бит 0	бит 1	бит 2	бит 3	бит 4	бит 5	бит 6	бит 7
4x	AND (3 операнда)				ADD (3 операнда)			
	di	im	in	ix	di	im	in	ix
5x	ANDB (3 операнда)				ADDB (3 операнда)			
	di	im	in	ix	di	im	in	ix
6x	AND (2 операнда)				ADD (2 операнда)			
	di	im	in	ix	di	im	in	ix
7x	ANDB (2 операнда)				ADDB (2 операнда)			
	di	im	in	ix	di	im	in	ix
8x	OR				XOR			
	di	im	in	ix	di	im	in	ix
9x	ORB				XORB			
	di	im	in	ix	di	im	in	ix
Ax	LD				ADDC			
	di	im	in	ix	di	im	in	ix
Bx	LDB				ADDCB			
	di	im	in	ix	di	im	in	ix
Cx	ST	BMOV	ST		STB	CMPL	STB	
	di	–	in	ix	di		in	ix
Dx	JNST	JNH	JGT	JNC	JNVT	JNV	JGE	JNE
Ex	DJNZ	DJNZW	TIJMP	BR in	*	*	*	LJMP
Fx	RET	*	PUSHF	POPF	PUSHA	POPA	IDLPD	TRAP

Продолжение таблицы А.8

Код	x8	x9	xA	xB	xC	xD	xE	xF
0x	SHR	SHL	SHRA	XCH in	SHRL	SHLL	SHRAL	NORML
1x	SHRB	SHLB	SHRAB	XCHB in	*	*	*	*
2x	SCALL							
3x	JBS							
	бит 0	бит 1	бит 2	бит 3	бит 4	бит 5	бит 6	бит 7
4x	SUB (3 операнда)				MULU (3 операнда)**			
	di	im	in	ix	di	im	in	ix
5x	SUBB (3 операнда)				MULUB (3 операнда)**			
	di	im	in	ix	di	im	in	ix
6x	SUB (2 операнда)				MULU (2 операнда)**			
	di	im	in	ix	di	im	in	ix

Окончание таблицы А.8

Код	x8	x9	xA	xB	xC	xD	xE	xF
7x	SUBB (2 операнда)				MULUB (2 операнда) **			
	di	im	in	ix	di	im	in	ix
8x	CMP				DIVU **			
	di	im	in	ix	di	im	in	ix
9x	CMPB				DIVUB *			
	di	im	in	ix	di	im	in	ix
Ax	SUBC				LDBZE			
	di	im	in	ix	di	im	in	ix
Bx	SUBCB				LDBSE			
	di	im	in	ix	di	im	in	ix
Cx	PUSH				POP	BMOVI	POP	
	di	im	in	ix	di	–	in	ix
Dx	JST	JH	JLE	JC	JVT	JV	JLT	JE
Ex	*	*	*	*	DPTS	EPTS	***	LCALL
Fx	CLRC	SETC	DI	EI	CLRVT	NOP	**	RST
<p>* Резервный код команды.</p> <p>** Знаковое умножение и деление – 2-байтные команды. Для любой знаковой команды первый байт – «FE», второй – код соответствующей незнаковой команды.</p> <p>*** Код EE зарезервирован; однако он не генерирует прерывания невыполнимого кода.</p>								

В таблице А.9 приведён список команд с их длинами и кодами для каждого режима адресации. Каждый режим адресации занимает две графы. В первой графе приведены длины команд, а во второй – шестнадцатеричные коды. Для индексных команд в первой графе длины команд приведены в виде S/L, где S – длина короткоиндексной команды и L – длина длинноиндексной команды. Прочерк «–» в графе показывает, что данный способ адресации для данной команды не применим.

Т а б л и ц а А.9 – Длина команды и шестнадцатеричный код

Арифметические команды (группа 1)								
Обозначение	Прямая		Непосредственная		Косвенная ¹⁾		Индексная S/L ¹⁾	
ADD (2 операнда)	3	64	4	65	3	66	4/5	67
ADD (3 операнда)	4	44	5	45	4	46	5/6	47
ADDB (2 операнда)	3	74	3	75	3	76	4/5	77
ADDB (3 операнда)	4	54	4	55	4	56	5/6	57
ADDC	3	A4	4	A5	3	A6	4/5	A7
ADDCB	3	B4	3	B5	3	B6	4/5	B7
CMP	3	88	4	89	3	8A	4/5	8B
CMPB	3	98	3	99	3	9A	4/5	9B
SUB (2 операнда)	3	68	4	69	3	6A	4/5	6B
SUB (3 операнда)	4	48	5	49	4	4A	5/6	4B
SUBB (2 операнда)	3	78	3	79	3	7A	4/5	7B
SUBB (3 операнда)	4	58	4	59	4	5A	5/6	5B
SUBC	3	A8	4	A9	3	AA	4/5	AB
SUBCB	4	B8	3	B9	3	BA	4/5	BB

Продолжение таблицы А.9

Арифметические команды (группа 2)								
Обозначение	Прямая		Непосредственная		Косвенная ¹⁾		Индексная S/L ¹⁾	
DIV	4	FE 8C	5	FE 8D	4	FE 8E	5/6	FE 8F
DIVB	4	FE 9C	4	FE 9D	4	FE 9E	5/6	FE 9F
DIVU	3	8C	4	8D	3	8E	4/5	8F
DIVUB	3	9C	3	9D	3	9E	4/5	9F
MUL (2 операнда)	4	FE 6C	5	FE 6D	4	FE 6E	5/6	FE 6F
MUL (3 операнда)	5	FE 4C	6	FE 4D	5	FE 4E	6/7	FE 4F
MULB (2 операнда)	4	FE 7C	4	FE 7D	4	FE 7E	5/6	FE 7F
MULB (3 операнда)	5	FE 5C	5	FE 5D	5	FE 5E	6/7	FE 5F
MULU (2 операнда)	3	6C	4	6D	3	6E	4/5	6F
MULU (3 операнда)	4	4C	5	4D	4	4E	5/6	4F
MULUB(2 операнда)	3	7C	3	7D	3	7E	4/5	7F
MULUB(3 операнда)	4	5C	4	5D	4	5E	5/6	5F
Логические команды								
Обозначение	Прямая		Непосредственная		Косвенная ¹⁾		Индексная S/L ¹⁾	
AND (2 операнда)	3	60	4	61	3	62	4/5	63
AND (3 операнда)	4	40	5	41	4	42	5/6	43
ANDB (2 операнда)	3	70	3	71	3	72	4/4	73
ANDB (3 операнда)	4	50	4	51	4	52	5/5	53
OR	3	80	4	81	3	82	4/5	83
ORB	3	90	3	91	3	92	4/5	93
XOR	3	84	4	85	3	86	4/5	87
XORB	3	94	3	95	3	96	4/5	97
Стековые команды								
Обозначение	Прямая		Непосредственная		Косвенная ¹⁾		Индексная S/L ¹⁾	
POP	2	CC			2	CE	3/4	CF
POPA	1	F5						
POPF	1	F3						
PUSH	2	C8	3	C9	2	CA	3/4	CB
PUSHA	1	F4						
PUSHF	1	F2						
Команды работы с данными								
Обозначение	Прямая		Непосредственная		Косвенная ¹⁾		Индексная S/L ¹⁾	
LD	3	A0	4	A1	3	A2	4/5	A3
LDB	3	B0	3	B1	3	B2	4/5	B3
LDBSE	3	BC	3	BD	3	BE	4/5	BF
LDBZE	3	AC	3	AD	3	AE	4/5	AF
ST	3	C0			3	C2	4/5	C3
STB	3	C4			3	C6	4/5	C7
XCH	3	04					4/5	0B
XCHB	3	14					4/5	1B
Команды перехода								
Обозначение	Прямая		Непосредственная		Косвенная ¹⁾		Индексная S/L ¹⁾	
BR (Indirect)					2	E3		
LJMP							-/3	E7
SJMP							2/-	20-27 ²⁾
TIJMP	4	E2	4	E2			-/4	E2

Продолжение таблицы А.9

Команды вызова								
Обозначение	Прямая		Непосредственная		Косвенная ¹⁾		Индексная S/L ¹⁾	
LCALL							3	EF
SCALL							2	28-2F ²⁾
RET					1	F0		
TRAP	1	F7						
Команды условного перехода								
Обозначение	Прямая		Непосредственная		Косвенная ¹⁾		Индексная S/L ¹⁾	
DJNZ							3	E0
DJNZW							3	E1
JBC							3	30-37
JBS							3	38-3F
JC							2	DB
JE							2	DF
JGE							2	D6
JGT							2	D2
JH							2	D9
JLE							2	DA
JLT							2	DE
JNC							2	D3
JNE							2	D7
JNH							2	D1
JNST							2	D0
JNV							2	D5
JNVT							2	D4
JST							2	D8
JV							2	DD
JVT							2	DC
Команды сдвига								
Обозначение	Прямая		Непосредственная		Косвенная ¹⁾		Индексная S/L ¹⁾	
NORML	3	0F						
SHL	3	09						
SHLB	3	19						
SHLL	3	0D						
SHR	3	08						
SHRA	3	0A						
SHRAB	3	1A						
SHRAL	3	0E						
SHRB	3	18						
SHRL	3	0C						
Блочные команды								
Обозначение	Прямая		Непосредственная		Косвенная ¹⁾		Индексная S/L ¹⁾	
BMOV					3	C1		
BMOVI					3	CD		

Окончание таблицы А.9

Специальные команды							
Обозначение	Прямая		Непосредственная		Косвенная ¹⁾		Индексная S/L ¹⁾
CLRC	1	F8					
CLRVT	1	FC					
DI	1	FA					
EI	1	FB					
IDLPD			1	F6			
NOP	1	FD					
RST	1	FE					
SETC	1	F9					
SKIP	2	00					
Команды управления PTS							
Обозначение	Прямая		Непосредственная		Косвенная ¹⁾		Индексная ¹⁾
DPTS	1	EC					
EPTS	1	ED					
Остальные команды							
Обозначение	Прямая		Непосредственная		Косвенная ¹⁾		Индексная S/L ¹⁾
CLR	2	01					
CLRB	2	11					
CMPL	3	C5					
DEC	2	05					
DECB	2	15					
EXT	2	06					
EXTB	2	16					
INC	2	07					
INCB	2	17					
NEG	2	03					
NEGB	2	13					
NOT	2	02					
NOTB	2	12					
<p>¹⁾ Косвенная обычная и косвенная автоинкрементная адресация имеют одинаковые коды команд, также как и коротко- и длинно-индексная адресация. Для использования косвенного обычного или коротко-индексного режима второй байт команды должен быть чётным. Для использования косвенного автоинкрементного или длинно-индексного режима второй байт должен быть нечётным.</p> <p>²⁾ Для этих команд (SCALL, SJMP) три младших значащих бита кода команды соединяются с 8 битами для формирования 11-битного дополнительного кода смещения.</p>							

В таблице А.10 приведён список команд в алфавитном порядке по группам, с их временем выполнения, измеренным в машинных циклах.

Т а б л и ц а А.10 – Время выполнения команд (в машинных тактах)

Арифметические команды (группа 1)						
Обозначение	Прямая	Непосред- ственная	Косвенная*		Индексная*	
			обычная	автоинкре- ментная	корот- кая	длин- ная
ADD (2 операнда)	4	5	6/8	7/9	6/8	7/9
ADD (3 операнда)	5	6	7/10	8/11	7/10	8/11
ADDB (2 операнда)	4	4	6/8	7/9	6/8	7/9
ADDB (3 операнда)	5	5	7/10	8/11	7/10	8/11
ADDC	4	5	6/8	7/9	6/8	7/9
ADDCB	4	4	6/8	7/9	6/8	7/9
CMP	4	5	6/8	7/9	6/8	7/9
CMPB	4	4	6/8	7/9	6/8	7/9
SUB (2 операнда)	4	5	6/8	7/9	6/8	7/9
SUB (3 операнда)	5	6	7/10	8/11	7/10	8/11
SUBB (2 операнда)	4	4	6/8	7/9	6/8	7/9
SUBB (3 операнда)	5	5	7/10	8/11	7/10	8/11
SUBC	4	5	6/8	7/9	6/8	7/9
SUBCB	4	4	6/8	7/9	6/8	7/9
Арифметические команды (группа 2)						
Обозначение	Прямая	Непосред- ственная	Косвенная*		Индексная*	
			обычная	автоинкре- ментная	корот- кая	длин- ная
DIV	26	27	28/31	29/32	29/32	30/33
DIVB	18	18	20/23	21/24	21/24	22/25
DIVU	24	25	26/29	27/30	27/30	28/31
DIVUB	16	16	18/21	19/22	19/22	20/23
MUL (2 операнда)	16	17	18/21	19/22	19/22	20/23
MUL (3 операнда)	16	17	18/21	19/22	19/22	20/23
MULB (2 операнда)	12	12	14/17	15/18	15/18	16/19
MULB (3 операнда)	12	12	14/17	15/18	15/18	16/19
MULU (2 операнда)	14	15	16/19	17/19	17/20	18/21
MULU (3 операнда)	14	15	16/19	17/19	17/20	18/21
MULUB (2 операнда)	10	10	12/15	13/15	12/16	14/17
MULUB (3 операнда)	10	10	12/15	13/15	12/16	14/17
Логические команды						
Обозначение	Прямая	Непосред- ственная	Косвенная*		Индексная*	
			обычная	автоинкре- ментная	корот- кая	длин- ная
AND (2 операнда)	4	5	6/8	7/9	6/8	7/9
AND (3 операнда)	5	6	7/10	8/11	7/10	8/11
ANDB (2 операнда)	4	4	6/8	7/9	6/8	7/9
ANDB (3 операнда)	5	5	7/10	8/11	7/10	8/11
OR	4	5	6/8	7/9	6/8	7/9
ORB	4	4	6/8	7/9	6/8	7/9
XOR	4	5	6/8	7/9	6/8	7/9
XORB	4	4	6/8	7/9	6/8	7/9

Продолжение таблицы А.10

Стековые команды (внутренний стек)						
Обозначение	Прямая	Непосредственная	Косвенная*		Индексная*	
			обычная	автоинкрементная	короткая	длинная
POP	8		10/12	11/13	11/13	12/14
POPA	12					
POPF	7					
PUSH	6	7	9/12	10/13	10/13	11/14
PUSHA	12					
PUSHF	6					
Стековые команды (внешний стек)						
Обозначение	Прямая	Непосредственная	Косвенная*		Индексная*	
			обычная	автоинкрементная	короткая	длинная
POP	11		13/15	14/16	14/16	15/17
POPA	18					
POPF	10					
PUSH	8	9	11/14	12/15	12/15	13/16
PUSHA	18					
PUSHF	8					
Команды работы с данными						
Обозначение	Прямая	Непосредственная	Косвенная*		Индексная*	
			обычная	автоинкрементная	короткая	длинная
LD	4	5	5/8	6/8	6/9	7/10
LDB	4	4	5/8	6/8	6/9	7/10
LDBSE	4	4	5/8	6/8	6/9	7/10
LDBZE	4	4	5/8	6/8	6/9	7/10
ST	4		5/8	6/9	6/9	7/10
STB	4		5/8	6/9	6/9	7/10
XCH	5				8/13	9/14
XCHB	5				8/13	9/14
Команды перехода						
Обозначение	Прямая	Непосредственная	Косвенная*		Индексная*	
			обычная	автоинкрементная	короткая	длинная
BR (Indirect)			7	7		
LJMP						7
SJMP					7	
TIJMP internal/internal	15	15				
external/internal	18	18				
external/external	21	21				
Команды вызова (внутренний стек)						
Обозначение	Прямая	Непосредственная	Косвенная*		Индексная*	
			обычная	автоинкрементная	короткая	длинная
LCALL						11
RET			11			
SCALL					11	
TRAP	16					

Продолжение таблицы А.10

Команды вызова (внешний стек)						
Обозначение	Прямая	Непосредственная	Косвенная*		Индексная*	
			обычная	автоинкрементная	короткая	длинная
LCALL						13
RET			14			
SCALL					13	
TRAP	18					
Команды условного перехода						
Обозначение	Коротко-индексная адресация					
DJNZ	5 (перехода нет), 9 (переход есть)					
DJNZW	6 (перехода нет), 10 (переход есть)					
JBC	5 (перехода нет), 9 (переход есть)					
JBS	5 (перехода нет), 9 (переход есть)					
JC	4 (перехода нет), 8 (переход есть)					
JE	4 (перехода нет), 8 (переход есть)					
JGE	4 (перехода нет), 8 (переход есть)					
JGT	4 (перехода нет), 8 (переход есть)					
JH	4 (перехода нет), 8 (переход есть)					
JLE	4 (перехода нет), 8 (переход есть)					
JLT	4 (перехода нет), 8 (переход есть)					
JNC	4 (перехода нет), 8 (переход есть)					
JNE	4 (перехода нет), 8 (переход есть)					
JNH	4 (перехода нет), 8 (переход есть)					
JNST	4 (перехода нет), 8 (переход есть)					
JNV	4 (перехода нет), 8 (переход есть)					
JNVT	4 (перехода нет), 8 (переход есть)					
JST	4 (перехода нет), 8 (переход есть)					
JV	4 (перехода нет), 8 (переход есть)					
JVT	4 (перехода нет), 8 (переход есть)					
Команды сдвига						
Обозначение	Прямая адресация					
NORML	8 + 1 на сдвиг (9 – сдвига нет)					
SHL	6 + 1 на сдвиг (7 – сдвига нет)					
SHLB	6 + 1 на сдвиг (7 – сдвига нет)					
SHR	6 + 1 на сдвиг (7 – сдвига нет)					
SHRA	6 + 1 на сдвиг (7 – сдвига нет)					
SHRAB	6 + 1 на сдвиг (7 – сдвига нет)					
SHRAL	7 + 1 на сдвиг (8 – сдвига нет)					
SHRB	6 + 1 на сдвиг (7 – сдвига нет)					
SHRL	7 + 1 на сдвиг (8 – сдвига нет)					
SHLL	7 + 1 на сдвиг (8 – сдвига нет)					
Блочные команды						
Обозначение	Косвенная адресация					
BMOV	internal/internal 6 + 8 на слово external/internal 6 + 11 на слово external/external 6 + 14 на слово					
BMOVI	internal/internal 7 + 8 на слово + 14 на прерывание external/internal 7 + 11 на слово + 14 на прерывание external/external 7 + 14 на слово + 14 на прерывание					

Окончание таблицы А.10

Специальные команды						
Обозначение	Прямая	Непосредственная	Косвенная*		Индексная*	
			обычная	автоинкрементная	короткая	длинная
CLRC	2					
CLRVT	2					
DI	2					
EI	2					
IDLPD неверный параметр правильный параметр	–	28 12	–	–	–	–
NOP	2					
RST	4					
SETC	2					
SKIP	3					
Команды управления PTS						
Обозначение	Прямая	Непосредственная	Косвенная*		Индексная*	
			обычная	автоинкрементная	короткая	длинная
DPTS	2					
EPTS	2					
Остальные команды						
Обозначение	Прямая	Непосредственная	Косвенная*		Индексная*	
			обычная	автоинкрементная	короткая	длинная
CLR	3					
CLRB	3					
CMPL	7					
DEC	3					
DECB	3					
EXT	4					
EXTB	4					
INC	3					
INCB	3					
NEG	3					
NEGB	3					
NOT	3					
NOTB	3					
<p>* В таблице время выполнения для косвенного и индексного режимов адресации обозначено в формате R/M, где R – время выполнения с использованием SFR и внутреннего RAM (0_H – 1FF_H), а M – время выполнения с использованием контроллера памяти (200_H – 0FFFF_H).</p>						

Приложение Б
(обязательное)

Описание сигналов

В приложении содержится информация о функциях выводов микроконтроллера 1874BE66T.

Б.1 Функциональные группы сигналов

В таблице Б.1 приведен список сигналов, сгруппированных по функциональному назначению.

Т а б л и ц а Б.1 – Сигналы, сгруппированные по функциональному назначению

Адрес и данные	Управление программирования	Вход/выход	Вход/выход
AD15 – 0	AINC #	P0.7–0/ACH7–0	P7.1,0/ CAPCOM5,4
	CPVER	P1.0/ACH8	P7.3,2/COMP5,4
Управление шины и состояние	PACT#	P1.1/ACH9	P7.6–4
ALE/ADV#	PALE#	P1.2/ACH10/T1CLK	P7.7/FREQOUT
BHE #/WRH #	PBUS.15–0	P1.3/ACH11/T1DIR	
BW	PMODE.3–0	P1.4/ACH12	
		P1.5/ACH13	
INST	PROG#	P2.3–0/CAPCOM3–0	
READY	PVER	P2.7–4/COMP3–0	
RD#		P3.7–0	
WR #/WRL #	Управление процессора	P4.7–0	
		P5.7–0	
	CLKOUT	P6.0/WG1#	
Питание и земля	EA#	P6.1/WG1	
⌋0V	EXTINT	P6.2/WG2#	
#VCC	NMI	P6.3/WG2	
VPR	ONCE#	P6.4/WG3#	
⌋VCC2	RESET#	P6.5/WG3	
#0V	BQ1	P6.6/PWM0	
	BQ2	P6.7/PWM1	

Б.2 Описание сигналов

В таблице Б.2 дано описание граф, используемых в таблице Б.3.

В таблице Б.3 приводится описание сигналов.

Т а б л и ц а Б.2 – Описание граф таблицы Б.3

Заголовок графы	Описание
Обозначение	Список обозначений сигналов, расположенных в алфавитном порядке. Многие выводы имеют несколько функций, поэтому в этой графе сигналов больше, чем выводов
Тип	Идентифицирует функцию вывода, внесенную в список графы. Обозначения: I – вход, O – выход, I/O – двунаправленный вывод, PWR – питание, GND – земля. Все выводы, кроме RESET#, являются фиксирующими входами. RESET# является чувствительным к уровню входом. В режиме POWERDOWN схема использует EXTINT как чувствительный к уровню вход
Наименование и описание	Кратко описывает функцию определенного сигнала, внесенного в список в графе «Обозначение». Также перечисляет другие дополнительные функции, мультиплексированные с сигналом

Т а б л и ц а Б.3 – Описание сигналов

Обозначение	Тип	Наименование и описание
1	2	3
АСН13 – АСН0	I	Аналоговые каналы. Эти выводы – аналоговые входы на АЦП. Эти выводы могут использоваться как аналоговые входы (АСНх) или цифровые входы (P0.y). Хотя выводы могут функционировать одновременно как аналоговые и цифровые входы, это не рекомендуется, так как чтение порта 0 во время преобразования может привести к искажению результатов преобразования. $\cap 0V$ и выводы $\cap VCC$ должны быть подключены для функционирования АЦП и порта 0. АСН13 – АСН0 мультиплексированы следующим образом: АСН0/P0.0, АСН1/P0.1, АСН2/P0.2, АСН3/P0.3, АСН4/P0.4/PMODE.0, АСН5/P0.5/PMODE.1, АСН6/P0.6/PMODE.2, АСН7/P0.7/PMODE.3, АСН8/P1.0, АСН9/P1.1, АСН10/P1.2/T1CLK, АСН11/P1.3/T1DIR и АСН12/P1.4, АСН13/P1.5.
AD15 – AD0	I/O	Системная шина адрес/данных. Эти выводы поддерживают мультиплексированную шину адреса/данных. В течение адресной фазы шинного цикла биты адреса 0 – 15 выводятся на шину и могут быть зафиксированы с помощью сигналов ALE или ADV#. В течение фазы данных на шину выводятся 8- или 16-битные данные. AD7– AD0 мультиплексированы с P3.7–0 и PBUS.7–0. AD15–8 мультиплексированы с P4.7–0 и PBUS.15–8
ADV#	O	Адрес действителен. Этот выходной сигнал с активным низким уровнем устанавливается только во время доступа к внешней памяти. ADV# показывает, что адресная информация действительно находится на шине адреса/данные. Сигнал остается низким пока идет шинный цикл и возвращается в высокий уровень, как только шинный цикл заканчивается. Внешняя защелка может использовать этот сигнал для демultipлексирования адреса из шины адреса/данные. Декодер может также использовать этот сигнал для генерации сигналов «выбор кристалла» внешней памяти. ADV# мультиплексирован с P5.0 и ALE

Продолжение таблицы Б.3

1	2	3																				
AINC#	I	Автоинкремент. Во время программирования этот вход с активным низким уровнем сигнала позволяет использовать автоинкрементирование. (Автоинкремент позволяет читать или писать из последовательных ячеек EPROM, не требуя адреса и через PBUS для каждого чтения или записи.) AINC# устанавливается после того, как каждая ячейка запрограммирована или разгружена. Если AINC# установлен, адрес увеличивается и следующее слово данных программируется или разгружается. AINC# мультиплексирован с P2.4 и COMP0																				
ALE	O	Разрешение записи адреса. Выходной сигнал с активным высоким уровнем устанавливается только во время доступа к внешней памяти. ALE сигнализирует о начале шинного цикла и показывает, что значение адресной информации действительно находится на шине адреса/данных. ALE отличается от ADV# тем, что не остается активным в течение всего шинного цикла. Внешняя защелка может использовать этот сигнал для демultipлексирования адреса из шины адреса/данных. ALE мультиплексировано с P5.0 и ADV#																				
∅0V	GND	Аналоговая земля ∅0V должна быть подключена для работы АЦП, порта 0 и порта 1. Потенциалы ∅0V и #0V должны быть равны																				
BHE #	O	Разрешение старшего байта. Во время 16-битного шинного цикла этот выходной сигнал с активным низким уровнем установлен для записи и чтения слова и старшего байта во внешнюю память. BHE # показывает, что данные действительно передаются по старшим разрядам шины данных. Необходимо использовать BHE# в сочетании с AD0 для определения, какой байт запоминающего устройства передается по системной шине: <table border="0"> <tr> <td>BHE #</td> <td>AD0</td> <td>Доступны</td> </tr> <tr> <td>0</td> <td>0</td> <td>оба байта</td> </tr> <tr> <td>0</td> <td>1</td> <td>только старший байт</td> </tr> <tr> <td>1</td> <td>0</td> <td>только младший байт</td> </tr> </table> BHE# мультиплексирован с P5.5 и WRN#. Примечание – Регистр конфигурации кристалла (CCR0) определяет, функционирует ли этот вывод как BHE# или WRN#. CCR0.2 = 1 – выбран BHE#; CCR0.2 = 0 – выбран WRN#	BHE #	AD0	Доступны	0	0	оба байта	0	1	только старший байт	1	0	только младший байт								
BHE #	AD0	Доступны																				
0	0	оба байта																				
0	1	только старший байт																				
1	0	только младший байт																				
BW	I	Ширина шины. Два бита регистра конфигурации контроллера, CCR0.1 и CCR1.2, наряду с выводом BW управляют шириной шины данных. Когда оба бита CCR установлены, сигнал BW выбирает ширину шины данных. Если установлен только один бит CCR, ширина шины установлена 8 или 16 бит, и сигнал BW игнорируется. <table border="0"> <tr> <td>CCR0.1</td> <td>CCR1.2</td> <td>BW</td> <td></td> </tr> <tr> <td>0</td> <td>1</td> <td>X</td> <td>установлена 8-битная шина данных</td> </tr> <tr> <td>1</td> <td>0</td> <td>X</td> <td>установлена 16-битная шина данных</td> </tr> <tr> <td>1</td> <td>1</td> <td>высокий</td> <td>16-битная шина данных</td> </tr> <tr> <td>1</td> <td>1</td> <td>низкий</td> <td>8-битная шина данных</td> </tr> </table> BW мультиплексирован с P5.7	CCR0.1	CCR1.2	BW		0	1	X	установлена 8-битная шина данных	1	0	X	установлена 16-битная шина данных	1	1	высокий	16-битная шина данных	1	1	низкий	8-битная шина данных
CCR0.1	CCR1.2	BW																				
0	1	X	установлена 8-битная шина данных																			
1	0	X	установлена 16-битная шина данных																			
1	1	высокий	16-битная шина данных																			
1	1	низкий	8-битная шина данных																			

Продолжение таблицы Б.3

1	2	3
CLKOUT	O	Системный тактовый сигнал. Выход внутреннего тактового генератора. Частота CLKOUT равна 1/2 частоты на входе генератора (F_{BQ1}). Скважность сигнала CLKOUT равна 50%
COMP5 – COMP0	O	Выходы каналов сравнения процессора событий EРА. Эти выходы мультиплексированы с другими сигналами и могут иметь конфигурацию стандартного I/O. COMP3 – 0 мультиплексированы следующим образом: COMP0/P2.4/AINC#, COMP1/P2.5/РАСТ#, COMP2/P2.6/CPVER, COMP3/P2.7, COMP4/P7.2, COMP5/P7.3
CPVER	O	Общая верификация программы. Во время программирования (режим SLAVE) высокий уровень этого сигнала показывает, что все ячейки памяти запрограммированы правильно, низкий сигнал указывает, что произошла ошибка во время хотя бы одного из этапов программирования. CPVER мультиплексирован с P2.6 и COMP2
EA#	I	<p>Доступ к внешней памяти. Этот сигнал с низким уровнем разрешает доступ к памяти вне кристалла. Если уровень высокий, разрешен доступ к внутренней памяти EPROM.</p> <p>EA# также контролирует вхождение микроконтроллера в режим программирования. Если $U_{EA\#}$ равно U_{VPR} (обычно + 12,5 В) при нарастающем фронте сигнала RESET#, микроконтроллер входит в режим программирования.</p> <p>Примечание – В промежутке между циклами обращения к внешней памяти шина адреса/данных при неактивном EA# находится в режиме IDLE, порты 3 и 4 можно использовать для стандартного ввода-вывода. При активном EA# эти порты выполняют только функцию шины адрес/данные.</p> <p>EA# фиксируется только по нарастающему фронту сигнала RESET#. Изменение уровня EA# после сброса игнорируется. Необходимо всегда соединять EA# с #0V при использовании микроконтроллера без внутреннего EPROM</p>
CAPCOMP5 – CAPCOMP0	I/O	Каналы захвата/сравнения процессора событий EРА. Высокоскоростные сигналы ввода-вывода для каналов сравнения/захвата EРА. CAPCOMP5–0 мультиплексированы следующим образом: CAPCOMP0/P2.0/PVER, CAPCOMP1/P2.1/PALE#, CAPCOMP2/P2.2/PROG#, CAPCOMP3/P2.3, CAPCOMP4/P7.0, CAPCOMP5/P7.1
EXTINT	I	Внешнее прерывание. Программируемое прерывание управляется регистром WG_PROTECT. Этот регистр определяет: фиксируется ли фронт сигнала прерывания или уровень, и нарастающим фронтом/высоким уровнем или спадающим фронтом/низким уровнем активируется прерывание. В режиме POWERDOWN сигнал EXTINT должен удерживаться больше 50 нс, чтобы восстановить нормальное выполнение операций. Необходимо при этом не разрешать прерывание. Если выполняется прерывание EXTINT, центральный процессор выполняет программу обслуживания прерывания. В противном случае центральный процессор выполняет инструкцию, которая немедленно следует за командой, вызвавшей энергосберегающий режим. В режиме IDLE установление любого разрешенного прерывания позволяет возобновить нормальное функционирование

Продолжение таблицы Б.3

1	2	3
FREQOUT	O	Выход генератора программируемого тактового сигнала. Скважность сигнала 50 %, частота может меняться от 4 кГц до 1 МГц (при $F_{BQ1} = 16$ МГц). FREQOUT мультиплексирован с P7.7
INST	O	Выбор команды. Этот сигнал с активным высоким уровнем действует только в цикле обращения к внешней памяти. Если уровень сигнала высокий, INST показывает, что выбирается команда из внешней памяти. Если уровень сигнала низкий, то происходит считывание данных из внутренней памяти. INST мультиплексирован с P5.1
NMI	I	Немаскируемое прерывание. Нарастающий фронт сигнала NMI производит немаскируемое прерывание. NMI имеет самый высокий приоритет над остальными приоритетами прерываний. Необходимо устанавливать длительность NMI больше одного машинного цикла, чтобы гарантировать его фиксацию
ONCE#	I	Режим внутрисхемной эмуляции. Удержание ONCE# в низком уровне во время нарастающего фронта RESET# переводит устройство в режим эмуляции (ONCE). Этот режим устанавливает все выводы, кроме BQ1 и BQ2, в высокоимпедансное состояние, изолируя, таким образом, устройство от других компонентов системы. Значение ONCE# фиксируют, когда вывод RESET# неактивен. В то время как устройство находится в режиме ONCE, имеется возможность произвести отладку системы, используя схемный эмулятор. Для выхода из режима ONCE, необходимо произвести сброс устройства путем установки низкого уровня сигнала RESET#. Чтобы предотвратить случайный вход в режим ONCE, необходимо сконфигурировать этот вывод как выход или поддерживать его в высоком состоянии в течение сброса, предполагая, что система соответствует U_{IH} спецификации. ONCE# мультиплексирован с P5.4
P0.7 – P0.0	I	Порт 0. Входной высокоимпедансный порт. Выводы порта 0 нельзя оставить неподключенными. Эти выводы могут использоваться как аналоговые входы (ACHx) или цифровые входы (P0.y). Хотя возможно функционирование выводов одновременно как аналоговые и цифровые входы, это не рекомендуется, так как чтение порта 0 во время преобразования может привести к ненадежным результатам преобразования. $\cap 0V$ и $\cap VCC$ должны быть подключены для работы порта 0. Порт 0 мультиплексирован следующим образом: P0.0/ACH0, P0.1/ACH1, P0.2/ACH2, P0.3/ACH3, P0.4/ACH4/PMODE.0, P0.5/ACH5/PMODE.1, P0.6/ACH6/PMODE.2, P0.7/ACH7/ PMODE.3

Продолжение таблицы Б.3

1	2	3
P1.7 – P1.0	I	<p>Порт 1. Это входной высокоимпедансный порт. Выводы порта 1 могут использоваться как аналоговые входы (ACHx) или цифровые входы (P1.y). Хотя возможно функционирование выводов одновременно как аналоговые и цифровые входы, это не рекомендуется, так как чтение порта 1 во время преобразования может привести к ненадежным результатам преобразования. $\cap 0V$ и $\cap VCC$ должны быть подключены для работы порта 1. Порт 1 мультиплексирован следующим образом: P1.0/ACH8, P1.1/ACH9, P1.2/ACH10/T1CLK, P1.3/ACH11/T1DIR, P1.4/ACH12, P1.5/ACH13. P1.6 и P1.7 не мультиплексированы с другими сигналами</p>
P2.7 – P2.0	I/O	<p>Порт 2. Это стандартный 8-битный двунаправленный порт, который мультиплексирован с индивидуально выбираемыми сигналами специальной функции. P2.6 мультиплексирован со специальной функцией тестового режима. Если этот вывод удерживается в низком уровне во время сброса, устройство войдет в резервный тестовый режим, поэтому соблюдается осторожность, если этот вывод используется как вход. Если необходима конфигурация этого вывода как входа, надо удерживать его в высоком состоянии в течение сброса и гарантировать, что система соответствует U_{IH} спецификации для предотвращения случайного входа в режим ONCE. Порт 2 мультиплексирован следующим образом: P2.0/EPA0/PVER, P2.1/EPA1/PALE#, P2.2/EPA2/PROG#, P2.3/EPA3, P2.4/COMP0/AINC#, P2.5/COMP1/PACT#, P2.6/COMP2/CPVER и P2.7/COMP3</p>
P3.7 – P3.0	I/O	<p>Порт 3. Это 8-битный двунаправленный порт с выходами. Выводы объединены с мультиплексной шиной адреса/данных, которая имеет комплементарные выходы. P3.7 – 0 мультиплексированы с AD7 – AD0 и PBUS.7 – PBUS.0</p>
P4.7 – P4.0	I/O	<p>Порт 4. Это 8-битный двунаправленный порт с выходами, программируемыми с открытым стоком или комплементарными. Выводы объединены с мультиплексной шиной адреса/данных, которая имеет комплементарные выходы. P4.7–0 мультиплексированы с AD15–8 и PBUS.15–8</p>
P5.7 – P5.0	I/O	<p>Порт 5. Это 8-битный двунаправленный порт, который мультиплексирован с индивидуально выбираемыми сигналами управления. P5.4 мультиплексирован с функцией ONCE#. Если этот вывод удерживать в низком уровне во время сброса, устройство войдет в режим ONCE. Если необходима конфигурация этого вывода как входа, надо удерживать его в высоком состоянии в течение сброса и гарантировать, что система соответствует U_{IH} спецификации для предотвращения случайного входа в режим ONCE. Порт 5 мультиплексирован следующим образом: P5.0/ALE/ADV#, P5.1/INST, P5.2/WR#/WRL#, P5.3/RD#, P5.4/ONCE#, P5.5/BHE#/WRH#, P5.6/READY и P5.7/BW</p>

Продолжение таблицы Б.3

1	2	3
P6.7 – P6.0	O	Порт 6. Это стандартный 8-битный порт только для вывода, который мультиплексирован со специальными функциями генератора формы сигнала и периферией PWM. Регистр WG_OUT конфигурирует выводы, устанавливает полярность выхода и управляет, будут ли синхронизированы изменения на выходах с событием или происходят немедленно. Порт 6 мультиплексирован следующим образом: P6.0/WG1#, P6.1/WG1, P6.2/WG2#, P6.3/WG2, P6.4/WG3#, P6.5/WG3, P6.6/PWM0 и P6.7/PWM1
P7.7 – P7.0	I/O	Порт 7. Это 8-битный двунаправленный порт с входами, организованными по схеме триггера Шмита. Он мультиплексирован следующим образом: P7.0/CAPCOMP4, P7.1/CAPCOMP5, P7.2/COMP4, P7.3/COMP5, P7.7/FREQOUT, и P7.6, P7.5, P7.4, которые не мультиплексированы
РАСТ#	O	Активное программирование. В режиме автоматического программирования или вывода содержимого памяти (ROM-dump) низкий уровень этого сигнала показывает, что происходит программирование или процесс вывода данных; высокий уровень сигнала указывает, что операция завершена. РАСТ# мультиплексирован с P2.5 и COMP1
PALE#	I	Программный ALE. В ходе программирования спадающий фронт этого сигнала заставляет устройство читать команду и адрес из PBUS. PALE# мультиплексирован с P2.1 и EPA1
PBUS.15 – PBUS.0	I/O	Шина адрес/команда/данные. Во время программирования (режим SLAVE) порты 3 и 4 служат двунаправленными портами с выводами с открытым стоком для передачи команд, адресов и данных к микроконтроллеру или от него. Программирование требует внешних резисторов для поддержки высокого уровня (режим SLAVE). В течение автоматического программирования и ROM-dump, порты 3 и 4 служат системной шиной для доступа к внешней памяти. P4.6 и P4.7 остаются неподключенными; P1.1 и P1.2 являются выводами старших адресов. Программирование (режим SLAVE): PBUS.7–0 мультиплексированы с AD7 – 0 и P3.7 – 0. PBUS.15–8 мультиплексированы с AD15–8 и P4.7–0. Автоматическое программирование: PBUS.7–0 мультиплексированы с AD7–0 и P3.7–0. PBUS.13–8 мультиплексированы с AD13–8 и P4.5–0; PBUS15–14 мультиплексированы с P1.2–1
PMODE.3 – PMODE.0	I	Выбор режима программирования. Определяют режим программирования. PMODE.x фиксируются после сброса устройства и должен быть стабильным во время работы микроконтроллера. PMODE.3–0 мультиплексированы с P0.7–4 и ACH7–4

Продолжение таблицы Б.3

1	2	3
PROG#	I	Начало программирования. Во время программирования спадающий фронт этого сигнала фиксирует данные на PBUS и начинает программирование, нарастающий фронт заканчивает программирование. Текущая ячейка памяти программируется теми же самыми данными, пока PROG# остается установленным, поэтому данные на PBUS должны остаться устойчивыми, пока PROG# является активным. При выводе слова спадающий фронт сигнала выводит содержимое ячеек EPROM на PBUS, в то время как нарастающий фронт заканчивает передачу данных. PROG# мультиплексирован с P2.2 и EPA2
PVER	O	Верификация программы. В процессе программирования (SLAVE) или автопрограммирования PVER обновляется после каждого импульса программирования. Высокий сигнал вывода указывает на успешное программирование ячейки памяти, низкий сигнал указывает на обнаружение ошибки. PVER мультиплексирован с P2.0 и EPA0
PWM1, PWM0	O	Выводы широтно-импульсного модулятора. Это выходы PWM с повышенной нагрузочной способностью. PWM1, 0 мультиплексированы с P6.7,6
RD#	O	Внешнее чтение. Это выходной сигнал чтения. RD# установлен только в процессе чтения внешней памяти. RD# мультиплексирован с P5.3
READY	I	Вход готовности. Это вход с активным высоким уровнем сигнала, который наряду с регистрами конфигурации кристалла определяет число циклов ожидания, включенных в шинный цикл. Регистры конфигурации кристалла выбирают максимальное число тактов ожидания (0, 1, 2, 3 или бесконечно большое), которое может быть включено в шинный цикл. Во время низкого уровня READY циклы ожидания включаются в шинный цикл, пока запрограммированное число тактов ожидания не будет достигнуто. Если READY становится высоким уровнем прежде, чем запрограммированное число тактов ожидания достигнуто, никаких дополнительных тактов ожидания не будет включено в шинный цикл. READY мультиплексирован с P5.6
RESET#	I/O	Вход «сброс» и выход с открытым стоком. Спадающий фронт сигнала RESET# или внутренний сброс инициирует включение транзистора с открытым стоком на выводе RESET# в течение 16 циклов. В режимах POWERDOWN и IDLE установка RESET# заставляет кристалл осуществлять сброс и возвращаться к нормальному рабочему режиму. После сброса устройства первая команда выбирается по адресу FF2080H
T1CLK	I	Вход внешнего тактирования таймера 1 и вход генератора, задающего скорость передачи для программного последовательного обмена. Таймер 1 увеличивается на единицу (уменьшается на единицу) и на нарастающем фронте T1CLK и на спадающем. Используется вместе с T1DIR для квадратурного режима счета. T1CLK мультиплексирован с P1.2 и ACH10

Продолжение таблицы Б.3

1	2	3
T1DIR	I	Вход внешнего управления направлением счета таймера 1 (прямое/обратное). Таймер 1 увеличивается на единицу при высоком уровне T1DIR и уменьшается на единицу – при низком. Используется вместе с T1CLK для квадратурного режима счета. T1DIR мультиплексирован с P1.3 и ACH11
#VCC	PWR	Напряжение питания цифровой части устройства
VPR	PWR	Напряжение программирования. Во время программирования напряжение на выводе VPR обычно +12,5 В. Превышение максимального VPR, приведенного в ТУ, может повредить устройство. VPR также управляет выходом устройства из режима POWERDOWN, когда удерживается в низком состоянии больше 50 нс. Использовать этот метод для выхода из POWERDOWN можно только при использовании внешнего источника синхроимпульсов. На устройствах без внутренней EPROM необходимо соединить VPR с #VCC
\cap VCC	PWR	Опорное напряжение для АЦП. Также является напряжением питания для аналоговой части АЦП и логики, используемой для чтения порта 0 и порта 1
#0V	GND	Цифровая схемная земля. Имеется несколько выводов #0V, все они должны быть соединены
WG3 – WG1	O	Выходы положительных фаз генератора формы сигнала 1–3. трехфазные выходные сигналы используются в устройствах управления двигателями. WG1 мультиплексирован с P6.1, WG2 мультиплексирован с P6.3 и WG3 мультиплексирован с P6.5
WG3# – WG1#	O	Выходы отрицательных фаз генератора формы сигнала 1–3. Комплементарные трехфазные выходные сигналы используются в устройствах управления двигателями.. WG1# мультиплексирован с P6.0, WG2# мультиплексирован с P6.2 и WG3# мультиплексирован с P6.4
WR#	O	Внешняя запись. Это выходной сигнал с активным низким уровнем, устанавливающийся при записи во внешнюю память.. WR# мультиплексирован с P5.2 и WRL#. Регистр конфигурации кристалла (CCR0) определяет, функционирует ли этот вывод как WR# или WRL#. При CCR0.2 = 1 выбран WR#; при CCR0.2 = 0 выбран WRL#
WRH#	O	Запись старшего байта. В 16-битном шинном режиме сигнал WRH# с активным низким уровнем устанавливается при записи старшего байта и слова, в 8-битном шинном режиме - при записи старшего и младшего байтов, а также слова. WRH# мультиплексирован с P5.5 и VHE#. Регистр конфигурации кристалла (CCR0) определяет, функционирует ли этот вывод как VHE# или WRH#. При CCR0.2 = 1 выбран VHE#; при CCR0.2 = 0 выбран WRH#

Окончание таблицы Б.3

1	2	3
WRL#	О	Запись младшего байта. В 16-битном шинном режиме сигнал WRL# с активным низким уровнем устанавливается при записи младшего байта или слова, в 8-битном шинном режиме – при записи старшего и младшего байтов, а также слова. WRL# мультиплексирован с P5.2 и WR#. Регистр конфигурации кристалла (CCR0) определяет, функционирует ли этот вывод как WR# или WRL#. При CCR0.2 = 1 выбран WR #; при CCR0.2 = 0 выбран WRL #
BQ1	I	Вход внешнего тактового сигнала или вывод для подключения кварцевого резонатора. Внутренние генераторы синхроимпульсов формируют периферийные синхроимпульсы, синхроимпульсы центрального процессора и сигнал CLKOUT. Если используется внешний генератор вместо внутрикристалльного, BQ1 служит входом тактов. Внешний сигнал синхроимпульсов должен удовлетворять U_{IH} спецификации для BQ1
BQ2	–	Вывод для подключения кварцевого резонатора. Необходимо оставить BQ2 неподключенным, если используется внешний источник синхроимпульсов

Б.3 Заданные по умолчанию состояния

В таблице Б.4 определяются символы, которыми обозначено состояние вывода. Для определения норм U_{OL} , U_{IL} , U_{OH} и U_{IH} необходимо обратиться к техническим условиям на микросхемы АЕЯР.431280.713ТУ.

В таблице Б.5 приведены значения сигналов в различных режимах.

Т а б л и ц а Б.4 – Определение символов состояния

Символ	Определение
0	Напряжение меньше или равное U_{OL} , U_{IL}
1	Напряжение больше или равное U_{OH} , U_{IH}
HiZ	Высокоимпедансное состояние
LoZ0	Низкий импеданс, формирователь «0» с повышенной нагрузочной способностью
LoZ1	Низкий импеданс, формирователь «1» с повышенной нагрузочной способностью
MD0	Средняя поддержка «0»
MD1	Средняя поддержка «1»
WK0	Слабая поддержка «0»
WK1	Слабая поддержка «1»
ODIO	Вход/выход с открытым стоком

Т а б л и ц а Б.5 – Заданные по умолчанию состояния сигналов

Сигналы порта	Дополни- тельные функции	Состояние сигналов		IDLE	POWER DOWN
		во время сброса	после сброса ¹⁾		
1	2	3	4	5	6
P0.7 – P0.0	ACH7 – 0	HiZ	–	HiZ	HiZ
P1.1, P1.0	ACH9, 8	HiZ	–	HiZ	HiZ
P1.2	ACH10/ T1CLK	HiZ	–	HiZ	HiZ
P1.3	ACH11/ T1DIR	HiZ	–	HiZ	HiZ
P1.4	ACH12	HiZ	–	HiZ	HiZ
P1.5	ACH13	HiZ	–	HiZ	HiZ
P1.7, P1.6			–	HiZ	HiZ
P2.0	CAPCOMP0	WK1 ²⁾	WK1	³⁾	³⁾
P2.1	CAPCOMP1	WK1	WK1	³⁾	³⁾
P2.3 – P2.2	CAPCOMP3,2	WK1	WK1	³⁾	³⁾
P2.6 – P2.4	COMP2 – 0	WK1	WK1, кроме P2.5 = LZ ²⁾	³⁾	³⁾
P2.7	COMP3	WK1	WK1	³⁾	³⁾
P3.7 – P3.0	AD7 – 0	WK1	HiZ	⁴⁾	⁴⁾
P4.7 – P4.0	AD15 – 8	WK1	HiZ	⁴⁾	⁴⁾
P5.0	ADV#/ALE	WK1 ²⁾	WK1 ⁵⁾	⁶⁾	⁶⁾
P5.1	INST	WK1	WK1	⁷⁾	⁷⁾
P5.2	WR#/WRL#	WK1 ²⁾	WK1	⁸⁾	⁸⁾
P5.3	RD#	WK1 ²⁾	WK1 ⁵⁾	⁸⁾	⁸⁾
P5.4	ONCE#	WK1	MD1 ²⁾	⁸⁾	⁸⁾
P5.5	BHE#/WRH#	WK1	WK1 ⁵⁾	⁸⁾	⁸⁾
P5.6	READY	WK1	⁹⁾	¹⁰⁾	¹⁰⁾
P5.7	BW	WK1	¹¹⁾	¹⁰⁾	¹⁰⁾
P6.0	WG1#	WK1	WK1	¹²⁾	¹²⁾
P6.1	WG1	WK1	WK1	¹²⁾	¹²⁾
P6.2	WG2#	WK1	WK1	¹²⁾	¹²⁾
P6.3	WG2	WK1	WK1	¹²⁾	¹²⁾
P6.4	WG3#	WK1	WK1	¹²⁾	¹²⁾
P6.5	WG3	WK1	WK1	¹²⁾	¹²⁾
P6.6	PWM0	WK0	–	¹²⁾	¹²⁾
P6.7	PWM1	WK0	–	¹²⁾	¹²⁾
P7.1, P7.0	CAPCOMP5,4	WK1 ²⁾	–	³⁾	³⁾
P7.3, P7.2	COMP5,4	WK1	–	³⁾	³⁾
P7.6 – P7.4	–	WK1	–	³⁾	³⁾
P7.7	FREQOUT	WK1	–	³⁾	³⁾

Окончание таблицы Б.5

1	2	3	4	5	6
–	CLKOUT	При действующем CLKOUT, LoZ0/1	–	CLKOUT активный, LoZ0/1	LoZ0
–	EA#	HiZ	–	HiZ	HiZ
–	EXTINT	HiZ	–	HiZ	HiZ
–	NMI	WK0	–	WK0	WK0
–	RESET#	LoZ0/HiZ ¹³⁾	–	HiZ	HiZ
–	VPR	HiZ	–	LoZ1	LoZ1
–	BQ1	вход осциллятора, HiZ	вход осциллятора, HiZ	вход осциллятора, HiZ	вход осциллятора, HiZ
–	BQ2	Osc выход, LoZ0/1	Osc выход, LoZ0/1	Osc выход, LoZ0/1	¹⁴⁾

¹⁾ Состояния сигналов в этой колонке действительны, пока программное обеспечение не запишет в регистр P_x_MODE.

²⁾ Эти выводы также управляют тестовым режимом.

³⁾ Если P_x_MODE.y = 0, порт работает как запрограммирован. Если P_x_MODE.y = 1, то работа вывода управляется периферийным устройством.

⁴⁾ Если EA# = 0, Порт 3 и Порт 4 = HiZ. Если EA# = 1, Порт 3 и Порт 4 = ODIO.

⁵⁾ Если EA# = 1, на выводе WK1. Если EA# = 0, P5.0, P5.3 и P5.5 формируются как выходы, а их функции: ADV#, RD# или BHE # соответственно.

⁶⁾ Если P5_MODE.0 = 0, порт работает как запрограммирован. Если P5_MODE.0 = 1 и CCR.3 = 1 (режим ALE), вывод - LoZ0. Если P5_MODE.0 = 1 и CCR.3 = 0 (ADV# режим), вывод - LoZ1.

⁷⁾ Если P5_MODE.1 = 0, порт работает как запрограммирован. Если P5_MODE.1 = 1, вывод - LoZ0.

⁸⁾ Если P5_MODE.y = 0, порт работает как запрограммирован. Если P5_MODE.y = 1, вывод - LoZ1.

⁹⁾ Функция READY не выбрана, необходимо три машинных цикла, чтобы осуществить фиксацию CCB.

¹⁰⁾ Если P5_MODE.y = 0, порт работает как запрограммирован. Если P5_MODE.y = 1, вывод - HiZ.

¹¹⁾ Функция BW выбрана.

¹²⁾ Выводы порта в режиме выхода работают, как запрограммированы. В режиме специальной функции работа вывода управляется периферийным устройством.

¹³⁾ Если выход RESET# = 0, то значение на выводе LoZ0. Если выход RESET# = 1, то значение на выводе HiZ.

¹⁴⁾ Если BQ1 = 1, вывод - LoZ0. Если BQ1 = 0, вывод - LoZ1.

Приложение В
(обязательное)

Регистры микроконтроллера

В приложении дана информация о регистрах микроконтроллера 1874BE66T.

В таблице В.1 определяется принадлежность регистров функциональным блокам.

В таблице В.2 представлены регистры с мнемоникой, названием и состоянием после сброса.

В таблицах В.3 –В56 описывается индивидуально каждый регистр.

Т а б л и ц а В.1 – Функциональные блоки микроконтроллера и соответствующие регистры

АЦП	Конфигурация кристалла	ЦПУ	ЕРА (x = 0–5)
AD_COMMAND AD_RESULT AD_TEST AD_TIME	CCR0 CCR1 PPW USFR	ONES_REG PSW SP ZERO_REG	COMPx_CON CMPx_TIME EPAx_CON EPAx_TIME
Генератор программируемого тактового сигнала	Порты (x = 0–5, 7)	Прерывания и PTS	Управление памятью
FREQ_CNT FREQ_GEN	Px_DIR Px_MODE Px_PIN Px_REG	INT_MASK INT_MASK1 INT_PEND INT_PEND1 PI_MASK PI_PEND PTSSEL PTSSRV	WSR
ШИМ (x = 0, 1)	Таймеры (x = 1, 2)	Генератор формы сигнала (x = 1, 2, 3)	
PWM_COUNT PWM_PERIOD PWMx_CONTROL	TxCONTROL T1RELOAD TIMERx WATCHDOG	WG_COMPx WG_CONTROL WG_COUNTER WG_OUTPUT WG_PROTECT WG_RELOAD	

Т а б л и ц а В.2 – Регистры микроконтроллера с мнемоникой, адресами и состоянием после сброса

Мнемоника регистра	Название регистра	Адрес регистра	Значение после сброса микроконтроллера	
			Старший байт	Младший байт
1	2	3	4	5
AD_COMMAND	Регистр команд АЦП	1FAC _H		10000000 _B
AD_RESULT	Регистр результата АЦП	1FAA _H	11111111 _B	11000000 _B
AD_TEST	Регистр проверки АЦП	1FAE _H		11000000 _B
AD_TIME	Регистр преобразования и выборки АЦП	1FAF _H	–	11111111 _B
CCR0	Регистр 0 конфигурации	1)	–	XXXXXXXX _B
CCR1	Регистр 1 конфигурации	1)	–	XXXXXXXX _B
COMP0_CON	Регистр ЕРА управление канала сравнения 0	1F58 _H	–	00000000 _B
COMP1_CON	Регистр ЕРА управление канала сравнения 1	1F5C _H	–	00000000 _B
COMP2_CON	Регистр ЕРА управление канала сравнения 2	1F60 _H	–	00000000 _B
COMP3_CON	Регистр ЕРА управление канала сравнения 3	1F64 _H	–	00000000 _B
COMP4_CON	Регистр ЕРА управление канала сравнения 4	1F68 _H	–	00000000 _B
COMP5_CON	Регистр ЕРА управление канала сравнения 5	1F6C _H	–	00000000 _B
COMP0_TIME	Регистр ЕРА времени канала сравнения 0	1F5A _H	XXXXXXXX _B	XXXXXXXX _B
COMP1_TIME	Регистр ЕРА времени канала сравнения 1	1F5E _H	XXXXXXXX _B	XXXXXXXX _B
COMP2_TIME	Регистр ЕРА времени канала сравнения 2	1F62 _H	XXXXXXXX _B	XXXXXXXX _B
COMP3_TIME	Регистр ЕРА времени канала сравнения 3	1F66 _H	XXXXXXXX _B	XXXXXXXX _B
COMP4_TIME	Регистр ЕРА времени канала сравнения 4	1F6A _H	XXXXXXXX _B	XXXXXXXX _B
COMP5_TIME	Регистр ЕРА времени канала сравнения 5	1F6E _H	XXXXXXXX _B	XXXXXXXX _B
ЕРА0_CON	Регистр управления каналом захвата/сравнения 0	1F40 _H	–	00000000 _B

Продолжение таблицы В2

1	2	3	4	5
EPA1_CON	Регистр управления каналом захвата/сравнения 1	1F44 _H	–	00000000 _B
EPA2_CON	Регистр управления каналом захвата/сравнения 2	1F48 _H	–	00000000 _B
EPA3_CON	Регистр управления каналом захвата/сравнения 3	1F4C _H	–	00000000 _B
EPA4_CON	Регистр управления каналом захвата/сравнения 4	1F50 _H	–	00000000 _B
EPA5_CON	Регистр управления каналом захвата/сравнения 5	1F54 _H	–	00000000 _B
EPA0_TIME	Регистр времени захвата канала захвата/сравнения 0	1F42 _H	XXXXXXXX _B	XXXXXXXX _B
EPA1_TIME	Регистр времени захвата канала захвата/сравнения 1	1F46 _H	XXXXXXXX _B	XXXXXXXX _B
EPA2_TIME	Регистр времени захвата канала захвата/сравнения 2	1F4A _H	XXXXXXXX _B	XXXXXXXX _B
EPA3_TIME	Регистр времени захвата канала захвата/сравнения 3	1F4E _H	XXXXXXXX _B	XXXXXXXX _B
EPA4_TIME	Регистр времени захвата канала захвата/сравнения 4	1F52 _H	XXXXXXXX _B	XXXXXXXX _B
EPA5_TIME	Регистр времени захвата канала захвата/сравнения 5	1F56 _H	XXXXXXXX _B	00000000 _B
FREQ_CNT	Регистр счета	1FBA _H	–	00000000 _B
FREQ_GEN	Регистр частоты	1FB8 _H	–	00000000 _B
INT_MASK	Регистр маски прерывания	0008 _H	–	00000000 _B
INT_MASK1	Регистр 1 маски прерывания	0013 _H	–	00000000 _B
INT_PEND	Регистр задержки прерывания	0009 _H	–	00000000 _B
INT_PEND1	Регистр 1 задержки прерывания	0012 _H	–	00000000 _B
ONES_REG	Единичный регистр	0002 _H	1111111 _B	1111111 _B
P2_DIR	Регистр направления ввода-вывода порта 2	1FD2 _H	–	1111111 _B
P5_DIR	Регистр направления ввода-вывода порта 5	1FF3 _H	–	1111111 _B

Продолжение таблицы В2

1	2	3	4	5
P7_DIR	Регистр направления ввода-вывода порта 7	1FD3 _H	–	11111111 _B
P2_MODE	Регистр режима порта 2	1FD0 _H	–	00000000 _B
P5_MODE	Регистр режима порта 5	1FF1 _H	–	2)
P7_MODE	Регистр режима порта 7	1FD1 _H	–	00000000 _B
P0_PIN	Регистр входа порта 0	1FA8 _H	–	3)
P1_PIN	Регистр входа порта 1	1FA9 _H	–	3)
P2_PIN	Регистр входа порта 2	1FD6 _H	–	3)
P3_PIN	Регистр входа порта 3	1FFE _H	–	3)
P4_PIN	Регистр входа порта 4	1FFF _H	–	3)
P5_PIN	Регистр входа порта 5	1FF7 _H	–	11111111 _B
P7_PIN	Регистр входа порта 7	1FD7 _H	–	XXXXXXXX _B
P2_REG	Регистр выхода порта 2	1FD4 _H	–	11111111 _B
P3_REG	Регистр выхода порта 3	1FFC _H	–	11111111 _B
P4_REG	Регистр выхода порта 4	1FFD _H	–	11111111 _B
P5_REG	Регистр выхода порта 5	1FF5 _H	–	3)
P7_REG	Регистр выхода порта 7	1FD5 _H	–	11111111 _B
PI_MASK	Регистр маскирования периферийного прерывания	1FBC _H		10101010 _B
PI_PEND	Регистр задержки периферийного прерывания	1FBE _H	–	10101010 _B
PPW	Регистр ширины программируемого импульса	нет прямого доступа	–	–
PSW	Регистр слова состояния процессора	нет прямого доступа	–	–
PTSEL	Регистр выбора PTS	0004 _H	00000000 _B	00000000 _B
PTSSRV	Регистр обслуживания PTS	0006 _H	00000000 _B	00000000 _B
PWM_COUNT	Регистр счета ШИМ	1FB6 _H	–	00000000 _B
PWM_PERIOD	Регистр периода ШИМ	1FB4 _H	–	00000000 _B
PWM0_CONTROL	Регистр управления ШИМ 0	1FB0 _H	–	00000000 _B
PWM1_CONTROL	Регистр управления ШИМ 1	1FB2 _H	–	00000000 _B
SP	Регистр указателя стека	0018 _H	XXXXXXXX _B	XXXXXXXX _B
T1CONTROL	Регистр управления таймера 1	1F78 _H	–	00000000 _B
T2CONTROL	Регистр управления таймера 2	1F7C _H	–	00000000 _B
T1RELOAD	Регистр перезагрузки таймера 1	1F72 _H	XXXXXXXX _B	XXXXXXXX _B
TIMER1	Регистр значения таймера 1	1F7A _H	00000000 _B	00000000 _B
TIMER2	Регистр значения таймера 2	1F7E _H	00000000 _B	00000000 _B

Окончание таблицы В2

1	2	3	4	5
USFR	Регистр специальных функций UPROM	1FF6 _H	–	00000010 _B
WATCHDOG	Регистр сторожевого таймера	000A _H	–	XXXXXXXX _B
WG_COMP1	Регистр буфера сравнения 1 ГФС	1FC2 _H	00000000 _B	00000000 _B
WG_COMP2	Регистр буфера сравнения 2 ГФС	1FC4 _H	00000000 _B	00000000 _B
WG_COMP3	Регистр буфера сравнения 3 ГФС	1FC6 _H	00000000 _B	00000000 _B
WG_CONTROL	Регистр управления ГФС	1FCC _H	00000000 _B	11000000 _B
WG_COUNTER	Регистр значения счетчика ГФС	1FCA _H	XXXXXXXX _B	XXXXXXXX _B
WG_OUTPUT	Регистр управления выходами ГФС	1FC0 _H	00000000 _B	00000000 _B
WG_PROTECT	Регистр защиты ГФС	1FCE _H	11110000 _B	00000000 _B
WG_RELOAD	Регистр перезагрузки ГФС	1FC8 _H	00000000 _B	00000000 _B
WSR	Регистр выбора окна	0014 _H	–	00000000 _B
ZERO_REG	Регистр нуля	0000 _H	00000000 _B	00000000 _B

¹⁾ Регистры CCR0, CCR1 после сброса микроконтроллера загружаются байтами конфигурации CCB0, CCB1, за исключением случая входа в режимы программирования. В этом случае загружаются байты конфигурации программирования (PCCB). Байты CCB0 и CCB1 располагаются во внутренней памяти программ по адресам 2018_H (CCB0) и 201A_H (CCB1).

²⁾ Значение после сброса равно 80_H, если сигнал EA# высокого уровня, и A9_H, если сигнал EA# низкого уровня.

³⁾ Значение после сброса равно FF_H, если вывод не задействован.

Регистр AD_COMMAND

Адрес: 1FAC_H

Состояние сброса: 80_H

Регистр команд АЦП (AD_COMMAND) выбирает номер АЦ канала, который будет преобразован, определяет, начинается ли АЦ преобразование немедленно или вызывается командой ЕРА, и выбирает режим преобразования.

Т а б л и ц а В.3 – Разряды регистра команд АЦП (AD_COMMAND)

Номер разряда	Мнемоника	Функция																																																																											
7	–	Зарезервирован, для совместимости с будущими устройствами записывают в этот бит ноль																																																																											
6, 5	M1 – M0	Режим АЦ преобразования. ¹⁾ Эти биты определяют режим АЦП. <table border="1"> <thead> <tr> <th>M1</th> <th>M0</th> <th>Режим</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>10-битное преобразование</td> </tr> <tr> <td>0</td> <td>1</td> <td>8-битное преобразование</td> </tr> <tr> <td>1</td> <td>0</td> <td>обнаружение напряжения выше порогового</td> </tr> <tr> <td>1</td> <td>1</td> <td>обнаружение напряжения ниже порогового</td> </tr> </tbody> </table>	M1	M0	Режим	0	0	10-битное преобразование	0	1	8-битное преобразование	1	0	обнаружение напряжения выше порогового	1	1	обнаружение напряжения ниже порогового																																																												
M1	M0	Режим																																																																											
0	0	10-битное преобразование																																																																											
0	1	8-битное преобразование																																																																											
1	0	обнаружение напряжения выше порогового																																																																											
1	1	обнаружение напряжения ниже порогового																																																																											
4	GO	Запуск АЦ преобразования. ²⁾ Запись этого бита запускает АЦП. Значение, записанное в него, определяет когда должно начаться преобразование. 0 – ЕРА запускает преобразование, 1 – начинается немедленно																																																																											
3 – 0	ACH3 – ACH0	Выбор канала АЦП. В эти биты записывается номер канала АЦП. <table border="1"> <thead> <tr> <th>ACH3</th> <th>ACH2</th> <th>ACH1</th> <th>ACH0</th> <th>канал АЦП</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>ACH0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>ACH1</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>ACH2</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>ACH3</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>ACH4</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>ACH5</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>ACH6</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>ACH7</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>ACH8</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>ACH9</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>ACH10</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>ACH11</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>ACH12</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>ACH13</td></tr> </tbody> </table>	ACH3	ACH2	ACH1	ACH0	канал АЦП	0	0	0	0	ACH0	0	0	0	1	ACH1	0	0	1	0	ACH2	0	0	1	1	ACH3	0	1	0	0	ACH4	0	1	0	1	ACH5	0	1	1	0	ACH6	0	1	1	1	ACH7	1	0	0	0	ACH8	1	0	0	1	ACH9	1	0	1	0	ACH10	1	0	1	1	ACH11	1	1	0	0	ACH12	1	1	1	0	ACH13
ACH3	ACH2	ACH1	ACH0	канал АЦП																																																																									
0	0	0	0	ACH0																																																																									
0	0	0	1	ACH1																																																																									
0	0	1	0	ACH2																																																																									
0	0	1	1	ACH3																																																																									
0	1	0	0	ACH4																																																																									
0	1	0	1	ACH5																																																																									
0	1	1	0	ACH6																																																																									
0	1	1	1	ACH7																																																																									
1	0	0	0	ACH8																																																																									
1	0	0	1	ACH9																																																																									
1	0	1	0	ACH10																																																																									
1	0	1	1	ACH11																																																																									
1	1	0	0	ACH12																																																																									
1	1	1	0	ACH13																																																																									

¹⁾ Когда выбран режим детектора порога для аналогового входа, никакое другое преобразование не может быть начато. Если другое значение загружено в AD_COMMAND, режим детектора порога запрещен и выполняется новая команда.

²⁾ Преобразование запускается записью в бит GO, а не его значением. Даже если бит GO имеет нужное значение, его необходимо установить снова, чтобы начать преобразование немедленно или же повторно очистить его для запуска преобразования модулем ЕРА.

Регистр AD_RESULT

Адрес: 1FAA_H

Состояние сброса: FFC0_H

В старший байт регистра результата АЦП (AD_RESULT) может быть записано значение для установки опорного напряжения в режиме детектирования порога. При чтении результата АЦ преобразования читаются старший и младший байты. В младшем байте содержатся два младших разряда 10-битного преобразования, номер и состояние АЦ преобразования канала.

Т а б л и ц а В.4 – Разряды регистра результата АЦП (AD_RESULT) при записи

Номер разряда	Мнемоника	Функция
15 – 8	REFV7 – REFV0	Опорное напряжение. Эти биты определяют пороговое значение. Они выбирают опорное напряжение для сравнения с аналоговым входным напряжением. Когда напряжение на аналоговом входе становится выше или ниже порогового значения, бит отложенного прерывания по завершению АЦ преобразования устанавливается. Для определения значения, записываемого в регистр, необходимо воспользоваться следующей формулой $\text{Опорное напряжение} = \frac{(\text{желаемое пороговое напряжение}) \times 256}{(U_{\text{DVCC}} - U_{\text{D0V}})}$
7 – 0	–	Зарезервированы; для совместимости с будущими устройствами в эти биты записывают нули

Т а б л и ц а В.5 – Разряды регистра результата АЦП (AD_RESULT) при чтении

Номер разряда	Мнемоника	Функция
15 – 6	ADRLT9 – ADRLT0	Результат АЦП. Эти биты содержат результат АЦ преобразования
5	–	Зарезервирован. Этот бит не определен
4	STATUS	Состояние АЦП. Показывает состояние АЦП. До 8 тактов требуется, чтобы установить этот бит после начала команды. При проверке этого бита необходимо ждать, по крайней мере, 8 тактов. 0 АЦ преобразование не выполняется. 1 АЦ преобразование выполняется
3 – 0	ACH3 – ACH0	Номер АЦП канала. Эти биты указывают номер АЦП канала, который использовался для преобразования (таблица В.3)

Регистр AD_TESTАдрес: 1FAE_HСостояние сброса: C0_H

Регистр проверки АЦ преобразования (AD_TEST) определяет установки корректировки ошибок смещения по постоянному току.

Т а б л и ц а В.6 – Разряды регистра проверки АЦ преобразования (AD_TEST)

Номер разряда	Мнемоника	Функция
7 – 5	–	Зарезервированы; для совместимости с будущими устройствами в эти биты записывают нули
4	OFF1	Смещение. Этот бит, наряду с OFF0 (бит 2), позволяет устанавливать точку смещения нуля. OFF1 OFF0 0 0 нет коррекции 0 1 прибавление 2,5 мВ 1 0 вычитание 2,5 мВ 1 1 вычитание 5,0 мВ
3	–	Зарезервирован; для совместимости с будущими устройствами в этот бит записывают ноль
2	OFF0	См. бит 4 (OFF1)
1, 0	–	Зарезервированы; для совместимости с будущими устройствами в эти биты записывают нули

Регистр AD_TIME

Адрес: 1FAFH

Состояние сброса: FFH

Регистр времени АЦП (AD_TIME) программирует время выборки и время преобразования для каждого бита. Этот регистр программирует скорость, с которой может идти АЦ преобразование, но не скорость, с которой это преобразование будет проходить корректно. Необходимо инициализировать регистр AD_TIME перед инициализацией регистра AD_COMMAND. Нельзя записывать значения в этот регистр во время преобразования – результаты непредсказуемы.

Т а б л и ц а В.7 – Разряды регистра времени АЦП (AD_TIME)

Номер разряда	Мнемоника бита	Функция
7–5	SAM2–0	Время выборки АЦП. Эти разряды определяют время выборки. Для вычисления времени выборки используется формула $SAM = \frac{T_{SAM} \times F_{BQ1} - 2}{8},$ где SAM – от 1 до 7; T _{SAM} – время выборки, мкс; F _{BQ1} – входная частота на выводе BQ1, МГц
4–0	CONV4–0	Время АЦ преобразования. Эти биты определяют время преобразования для каждого бита. Для вычисления времени преобразования используется формула $CONV = \frac{T_{CONV} \times F_{BQ1} - 3}{2 \times B} - 1,$ где CONV – от 2 до 31; T _{CONV} – время преобразования, мкс; F _{BQ1} – входная частота на выводе BQ1, МГц; B – число битов, которые будут преобразованы (8 или 10)

Регистр CCR0

Нет прямого доступа.

Регистр конфигурации 0 (CCR0) управляет режимом пониженного потребления мощности, сигналами управления шины, защитой встроенной памяти. Три разряда в комбинации с двумя разрядами регистра CCR1 управляют «затягиванием» шинного цикла и разрядностью шины данных.

Т а б л и ц а В.8 – Разряды регистра конфигурации кристалла 0 (CCR0)

Номер бита	Мнемоника	Функция
1	2	3
7, 6	LOC1, LOC0	Биты защиты. Управляют доступом чтения или записи EPROM в режиме нормальных операций LOC1 LOC0 Доступ EPROM 0 0 запрет чтения и записи 0 1 запрет чтения 1 0 запрет записи 1 1 нет защиты
5, 4	IRC1, IRC0	Управление «затягиванием» шинного цикла. Два бита совместно с IRC1.1 (CCR1) и READY управляют «затягиванием» шинного цикла. Если READY удерживается в низком уровне, цикл шины «затягивается» в соответствии с запрограммированным числом тактов (машинных). Если READY переключается в высокий уровень раньше, чем достигнуто запрограммированное число тактов, «затягивание» цикла прекращается. При выборе варианта неопределённого времени «затягивания» P5.6 конфигурируется как сигнал READY. Рекомендуется дополнить аппаратное обеспечение счётчиком необходимого числа тактов «затягивания» и установки READY высоким уровнем, т. к. дефектная ИС памяти может занимать шину неопределённое время. IRC2 IRC1 IRC0 «Затягивание» времени цикла шины 0 0 0 нет 0 X 1 запрещённые 0 1 X комбинации 1 0 0 1 такт 1 0 1 2 такта 1 1 0 3 такта 1 1 1 определяется сигналом READY

Окончание таблицы В.8

1	2	3
3	ALE	Разрешение записи адреса и запись данных. Определяют комбинацию сигналов управления шиной при внешних циклах чтения и записи. ALE WR режимы 1 0 стробирование действительного адреса и записи (ADV#, RD#, WRL#, WRH#)
2	WR	0 1 стробирование действительного адреса (ADV#, RD#, WR#, BHE#) 1 0 стробирование записи (ADV#, RD#, WRL#, WRH#) 1 1 стандартный (ALE, RD#, WR#, BHE#)
1	BW0	Управление разрядностью шины. Осуществляется совместно с BW1 (CCR1.2) BW1 BW0 разрядность 0 0 запрещённая комбинация 0 1 только 16-разрядная 1 0 только 8-разрядная 1 1 управление по BW
0	PD	Выбор режима пониженного потребления мощности. Разрешает команде IDLPD #2 перевести МК в режим пониженного потребления мощности. PD = 0 – запрет пониженного потребления мощности; PD = 1 – разрешение пониженного потребления мощности

Регистр CCR1

Нет прямого доступа.

Регистр конфигурации 1 (CCR1) разрешает сторожевой таймер, выбирает временной режим шины, совместно с CCR1 управляет «затягиванием» шинного цикла и разрядностью шины данных.

Т а б л и ц а В.9 – Разряды регистра конфигурации кристалла 1 (CCR1)

Номер бита	Мнемоника	Функция
1	2	3
7, 6	1	Записываются «1»
5	0	Записывается «0»
4	1	Записывается «1»
3	WDE	Разрешение сторожевого таймера. 0 – разрешён всегда, 1 – запрещён до очистки регистра WATCHDOG
2	BW1	Управление разрядностью шины. Осуществляется совместно с BW0 (CCR0.1) BW1 BW0 разрядность 0 0 запрещённая комбинация 0 1 только 16-разрядная 1 0 только 8-разрядная 1 1 управление по BW
1	IRC2	Управление затягиванием шинного цикла. Бит совместно с IRC0, IRC1 (CCR0.4, CCR0.5) и READY управляют затягиванием шинного цикла. Если READY удерживается в низком уровне, цикл шины затягивается в соответствии с запрограммированным числом тактов (машинных). Если READY переключается в высокий уровень раньше, чем достигнуто запрограммированное число тактов, затягивание цикла прекращается. При выборе варианта неопределённого времени затягивания P5.6 конфигурируется как сигнал READY. Рекомендуется дополнить аппаратное обеспечение счётчиком необходимого числа тактов «затягивания» и установки READY высоким уровнем, т. к. дефектная ИС памяти может занимать шину неопределённое время. IRC2 IRC1 IRC0 «Затягивание» времени цикла шины 0 0 0 нет 0 X 1 запрещённые 0 1 X комбинации 1 0 0 1 такт 1 0 1 2 такта 1 1 0 3 такта 1 1 1 определяется сигналом READY
0	0	Записывается «0»

Регистр COMP_x_CON (x = 0–5)Адреса: 1F58_H, 1F5C_H, 1F60_H, 1F64_H, 1F68_H, 1F6C_HСостояние после сброса: 00_H

Регистры управления ЕРА сравнением (COMP_x_CON) определяют функции сравнения каналов ЕРА. Для регистров COMP0_CON, COMP2_CON, COMP4_CON разряд 2 имеет функциональность WRG, для регистров COMP1_CON, COMP3_CON, COMP5_CON разряд 2 имеет функциональность AD.

Т а б л и ц а В.10 – Разряды регистра управления сравнением ЕРА (COMP_x_CON)

Номер разряда	Название	Функция
1	2	3
7	ТВ	Выбор временной базы. Определяет текущий таймер. 0 – таймер 1, текущий таймер, а таймер 2 – противоположный таймер, 1 – таймер 2, текущий таймер, а таймер 1 – противоположный таймер. Событие сравнения (старт АЦ преобразования, очистка, установка или переключение выхода, и/или перезагрузка любого таймера) происходит, когда значение в текущем таймере соответствует времени, запрограммированному в регистре времени события
6	СЕ	Разрешение сравнения. Разрешает работать каналу ЕРА в режиме сравнения. 0 – режим сравнения запрещен, 1 – режим сравнения разрешен
5, 4	M1, M0	Выбор режима ЕРА. Определяет тип сравниваемого события. M1 M0 Действие в режиме сравнения 0 0 нет изменений 0 1 переключение в низкий уровень 1 0 переключение в высокий уровень 1 1 переключение уровня
3	RE	Повторное разрешение. Повторное разрешение применимо только к режиму сравнения. Это позволяет продолжать выполнять сравнение события каждый раз, когда регистр (ЕРА _x _TIME) соответствует текущему таймеру, а не только до первого сравнения. 0 – сравнение, функция запрещена после первого события, 1 – сравнение, функция всегда разрешена

Окончание таблицы В.10

1	2	3
2	WGR AD	<p>Перезагрузка генератора формы сигнала, АЦ преобразование. Функция этого бита зависит от канала ЕРА. Для каналов захвата/сравнения CAPCOMP0, CAPCOMP2, CAPCOMP4: Разряд WGR позволяет использовать действия ЕРА, чтобы вызвать перезагрузку новых значений в генератор формы сигнала. 0 – нет действия, 1 – разрешена перезагрузка генератора формы сигнала. Для каналов захвата/сравнения CAPCOMP1, CAPCOMP3, CAPCOMP5: Разряд AD разрешения АЦ преобразования позволяет использовать действия ЕРА, чтобы начать АЦ преобразование, которое было предварительно подготовлено в регистрах управления АЦП. 0 – не начинать АЦ преобразование, 1 – старт аналого-цифрового преобразования при сравнении</p>
1	ROT	<p>Сброс противоположного таймера. Выбирает таймер, который должен быть сброшен, если RT бит установлен. 0 – выбирает текущий таймер для возможного сброса 1 – выбирает противоположный таймер для возможного сброса. Состояние ТВ бита определяет, какой таймер является текущим таймером и какой таймер является противоположным таймером</p>
0	RT	<p>Сброс таймера. Этот бит управляет, будет ли таймер, выбранный битом ROT, сброшен. 1 – сбрасывает таймер, выбранный битом ROT 0 – запрещает функцию сброса</p>

Регистр COMP_x_TIME (x = 0–5)Адреса: 1F5A_H, 1F5E_H, 1F62_H, 1F66_H, 1F6A_H, 1F6E_HСостояние после сброса: XXXX_H

Регистры времени сравнения COMP_x_TIME блока ЕРА для каналов только сравнения (COMP0 – COMP5). Канал ЕРА включает событие сравнения на выводе, когда значение соответствующего таймера сравнивается со значением в COMP_x_TIME.

Т а б л и ц а В.11 – Разряды регистра времени сравнения (COMP_x_TIME) для каналов только сравнения

Номер разряда	Описание
15 – 0	Значение времени события блока ЕРА. Запись в регистр определяет время события сравнения для каналов сравнения

Регистр ЕРА_x_TIME (x = 0–5)Адреса: 1F42_H, 1F46_H, 1F4A_H, 1F4E_H, 1F52_H, 1F56_HТ а б л и ц а В.12 – Адреса и значения после сброса для регистров ЕРА_x_TIME

Регистр	Адрес	Значение после сброса
ЕРА0_TIME	1F42 _H	XXXX _H
ЕРА1_TIME	1F46 _H	XXXX _H
ЕРА2_TIME	1F4A _H	XXXX _H
ЕРА3_TIME	1F4E _H	XXXX _H
ЕРА4_TIME	1F52 _H	XXXX _H
ЕРА5_TIME	1F56 _H	XX00 _H

Регистры времени сравнения ЕРА_x_TIME блока ЕРА для каналов захвата-сравнения (CAPCOMP0 – CAPCOMP5). В режиме захвата ЕРА производит захват значения соответствующего таймера в ЕРА_x_TIME при событии на выводе. В режиме сравнения ЕРА включает событие сравнения на выводе, когда значение соответствующего таймера сравнивается со значением в ЕРА_x_TIME. Регистры не буферизированы в режиме сравнения.

Т а б л и ц а В.13 – Разряды регистра времени сравнения (ЕРА_x_TIME) для каналов захвата- сравнения

Номер разряда	Описание
15 – 0	Значение таймера ЕРА. Когда канал ЕРА конфигурирован в режиме захвата, этот регистр содержит значение, захваченное у соответствующего таймера, если на выводе имеет место событие. Когда канал ЕРА конфигурирован в режим сравнения, в этот регистр надо записать время события, которое будет сравниваться со значением в соответствующем таймере

Регистр EPA_x_CON (x = 0–5)Адреса: 1F40_H, 1F44_H, 1F48_H, 1F4C_H, 1F50_H, 1F54_HСостояние сброса: 00_H

Регистры управление EPA (EPA_x_CON) управляют функциями соответствующих каналов захвата/сравнения. Для регистров EPA0_CON, EPA2_CON, EPA4_CON разряд 2 имеет функциональность WRG, для регистров EPA1_CON, EPA3_CON, EPA5_CON разряд 2 имеет функциональность AD.

Т а б л и ц а В.14 – Разряды регистра управления EPA (EPA_x_CON)

Номер разряда	Название	Функция
1	2	3
7	ТВ	Выбор временной базы. Определяет текущий таймер. 0 – таймер 1, текущий таймер, а таймер 2 – противоположный таймер, 1 – таймер 2, текущий таймер, а таймер 1 – противоположный таймер. Событие сравнения (старт АЦ преобразования, очистка, установка или переключение выхода, и/или перезагрузка любого таймера) происходит, когда значение в текущем таймере соответствует времени, запрограммированному в регистре времени события. Когда происходит захват (по заднему, по переднему фронту или по любому изменению фронта на выводе блока EPA), соответствующее значение таймера сохраняется в регистре EPA _x _TIME
6	СЕ	Разрешение сравнения. Разрешает работать каналу EPA в режиме захвата или сравнения. 0 – режим захвата, 1 – режим сравнения
5, 4	M1, M0	Выбор режима EPA. В режиме захвата определяет тип события, которое будет захватываться на входе. В режиме сравнения определяет действие, выполняемое блоком EPA в момент сравнения значения таймера и времени события. M1 M0 Действие в режиме захвата 0 0 нет изменений 0 1 захват по заднему (отрицательному) фронту 1 0 захват по переднему (положительному) фронту 1 1 захват по любому фронту M1 M0 Действие в режиме сравнения 0 0 нет изменений 0 1 переключение в низкий уровень 1 0 переключение в высокий уровень 1 1 переключение уровня
3	RE	Повторное разрешение. Повторное разрешение применимо только к режиму сравнения. Это позволяет продолжать выполнять сравнение события каждый раз, когда регистр (EPA _x _TIME) соответствует текущему таймеру, а не только до первого сравнения. 0 – сравнение, функция запрещена после первого события, 1 – сравнение, функция всегда разрешена

Окончание таблицы В.14

1	2	3
2	WGR AD	<p>Перезагрузка генератора формы сигнала, АЦ преобразование. Функция этого бита зависит от канала ЕРА. Для канала захвата/сравнения CAPCOMP0, CAPCOMP2, CAPCOMP4: бит WGR позволяет использовать действия ЕРА, чтобы вызвать перезагрузку новых значений в генератор формы сигнала. 0 – нет действия, 1 – разрешена перезагрузка генератора формы сигнала. Для каналов захвата/сравнения CAPCOMP1, CAPCOMP3, CAPCOMP5: бит AD разрешения АЦ преобразования позволяет использовать действия ЕРА, чтобы начать АЦ преобразование, которое было предварительно подготовлено в регистрах управления АЦП. 0 – не начинать АЦ преобразование 1 – старт АЦ преобразования при сравнении</p>
1	ROT	<p>Сброс противоположного таймера. Управление разными функциями для режимов захвата и сравнения. Выбирает таймер, который должен быть сброшен, если RT бит установлен. В режиме захвата: 0 – нет действия, 1 – сбрасывает противоположный таймер. В режиме сравнения: 0 – выбирает текущий таймер для возможного сброса, 1 – выбирает противоположный таймер для возможного сброса. Состояние бита ТВ определяет, какой таймер является текущим таймером и какой таймер является противоположным таймером</p>
0	ON/RT	<p>Перезапись/сброс таймера. Этот бит перезаписывает новое значение в режиме захвата и сбрасывает таймер в режиме сравнения. В режиме захвата (ON) ошибка переполнения происходит во время входного захвата, в то время как регистр (ЕРАх_TIME) и его буфер переполнены. Когда происходит переполнение, бит ON определяет: будут ли перезаписаны старые данные или новые данные игнорируются: 0 – новые данные игнорируются, 1 – старые данные в буфере перезаписываются. В режиме сравнения (RT): 0 – запрещает функцию сброса, 1 – сброс таймера, выбираемого битом ROT</p>

Регистр `FREQ_CNT`Адрес: `1FBAH`Состояние сброса: `00H`

Регистр только для чтения `FREQ_CNT` содержит текущее значение счетчика обратного счета

Т а б л и ц а В.15 – Разряды регистра счета `FREQ_CNT` генератора программируемого тактового сигнала

Номер разряда	Функция
7 – 0	Текущее значение счетчика обратного счета

Регистр `FREQ_GEN`Адрес: `1FB8H`Состояние сброса: `00H`

Регистр `FREQ_GEN` содержит программируемое значение для выходной частоты. Значение (`FREQ`) из регистра перегружается в счетчик обратного счета при достижении этим счетчиком нулевого значения.

Т а б л и ц а В.16 – Разряды регистра частоты генератора программируемого тактового сигнала `FREQ_GEN`

Номер разряда	Функция
7 – 0	<p>Выходная частота. Необходимо использовать следующую формулу для расчета значения (<code>FREQ</code>) регистра <code>FREQ_GEN</code>:</p> $FREQ = \frac{F_{BQ1}}{16 \times FREQ_OUT} - 1,$ <p>где <code>FREQ</code> – 8-разрядное значение регистра <code>FREQ_GEN</code>; F_{BQ1} – частота тактового сигнала на входе <code>BQ1</code>, МГц; $FREQ_OUT$ – частота выходного сигнала на выводе <code>FREQOUT</code>, МГц</p>

Регистр INT_MASK

Адрес: 0008_H

Состояние сброса: 00_H

Регистр маски прерываний INT_MASK разрешает или запрещает индивидуальные запросы прерываний. Команды EI и DI разрешают и запрещают обслуживание всех маскируемых прерываний. INT_MASK – младший байт слова состояния процессора (PSW). PUSHF или PUSHA сохранит содержимое этого регистра в стеке и затем очистит этот регистр. Вызов прерывания не может произойти немедленно после этих инструкций. POPF или POPA восстанавливают содержимое регистра.

Т а б л и ц а В.17 – Разряды регистра маски прерываний (INT_MASK)

Номер бита	Мнемоника бита	Прерывание	Адреса стандартных векторов
7	COMP2	ЕРА сравнение канал 2	200E _H
6	CAPCOMP2	ЕРА захват/сравнение канал 2	200C _H
5	COMP1	ЕРА сравнение канал 1	200A _H
4	CAPCOMP1	ЕРА захват/сравнение канал 1	2008 _H
3	COMP0	ЕРА сравнение канал 0	2006 _H
2	CAPCOMP0	ЕРА захват/сравнение канал 0	2004 _H
1	AD	АЦ преобразование завершено	2002 _H
0	OVRTM *	Переполнение/антипереполнение таймера	2000 _H

П р и м е ч а н и е – Установка бита разрешает соответствующее прерывание.

* Таймер 1 и таймер 2 могут производить мультиплексное прерывание по переполнению/антипереполнению таймера. Записывается PI_MASK, чтобы разрешить источники прерываний. Считается PI_PEND, чтобы определить, который источник вызвал прерывание.

Регистр INT_MASK1

Адрес: 0013_H

Состояние сброса: 00_H

Регистр маски прерывания (INT_MASK1) позволяет или запрещает индивидуальные запросы прерываний. (Инструкции EI и DI позволяют и запрещают обслуживание всех маскируемых прерываний.) INT_MASK1 может считываться или записываться как байтовый регистр. PUSHA сохраняет содержимое регистра в стеке, а POPA восстанавливает его содержимое.

Т а б л и ц а В.18 – Разряды регистра маски прерываний (INT_MASK)

Номер бита	Мнемоника бита	Прерывание	Адреса стандартных векторов
7	NMI	Немаскируемое прерывание	203E _H
6	EXTINT	Вывод EXTINT	203C _H
5	PI *	Мультиплексное периферийное прерывание	203A _H
4	CAPCOMP5	ЕРА захват/сравнение канал 5	2038 _H
3	COMP4	ЕРА сравнение канал 4	2036 _H
2	CAPCOMP4	ЕРА захват/сравнение канал 4	2034 _H
1	COMP3	ЕРА сравнение канал 3	2032 _H
0	CAPCOMP3	ЕРА захват/сравнение канал 3	2030 _H

П р и м е ч а н и е – Установка бита разрешает соответствующее прерывание.

* ГФС и канал 5 блока ЕРА в режиме сравнения могут вызвать это прерывание. Запись в PI_MASK разрешает источники прерываний, чтение PI_PEND определяет вызывающий прерывание источник.

Регистр INT_PEND

Адрес: 0009_H

Состояние сброса: 00_H

Когда аппаратные средства обнаруживают запрос прерывания, устанавливается соответствующий бит в регистрах INT_PEND или INT_PEND1. Когда вектор выбран, то аппаратные средства очищают бит ожидания. Программное обеспечение может генерировать прерывание установкой соответствующего бита ожидания прерывания.

Т а б л и ц а В.19 – Разряды регистра ожидания прерывания (INT_PEND)

Номер бита	Мнемоника бита	Прерывание	Адреса стандартных векторов
7	COMP2	ЕРА сравнение канал 2	200E _H
6	CAPCOMP2	ЕРА захват/сравнение канал 2	200C _H
5	COMP1	ЕРА сравнение канал 1	200A _H
4	CAPCOMP1	ЕРА захват/сравнение канал 1	2008 _H
3	COMP0	ЕРА сравнение канал 0	2006 _H
2	CAPCOMP0	ЕРА захват/сравнение канал 0	2004 _H
1	AD	АЦ преобразование завершено	2002 _H
0	OVRTM *	Переполнение/антипереполнение таймера	2000 _H

П р и м е ч а н и е – Любой установленный бит указывает, что ожидается обслуживание прерывания. Бит прерывания очищается, когда процесс передается по соответствующему вектору прерывания.

* Таймер 1 и таймер 2 могут произвести мультиплексное прерывание по переполнению/антипереполнению. Записав в PI_MASK разрешение источника прерывания, считайте PI_PEND, чтобы определить источник, который вызвал прерывание.

Регистр INT_PEND1

Адрес: 0012_H

Состояние сброса: 00_H

Когда аппаратные средства обнаруживают запрос прерывания, они устанавливают соответствующий бит в регистрах INT_PEND или INT_PEND1. Когда вектор выбран, аппаратные средства очищают соответствующий бит. Программное обеспечение может произвести прерывание установкой бита ожидания прерывания.

Т а б л и ц а В.20 – Разряды регистра ожидания прерывания 1 (INT_PEND1)

Номер бита	Мнемоника бита	Прерывание	Адреса стандартных векторов
7	NMI	Немаскируемое прерывание	203E _H
6	EXTINT	Вывод EXTINT	203C _H
5	PI *	Мультиплексное периферийное прерывание	203A _H
4	CAPCOMP5	ЕРА захват/сравнение канал 5	2038 _H
3	COMP4	ЕРА сравнение канал 4	2036 _H
2	CAPCOMP4	ЕРА захват/сравнение канал 4	2034 _H
1	COMP3	ЕРА сравнение канал 3	2032 _H
0	CAPCOMP3	ЕРА захват/сравнение канал 3	2030 _H

Примечание – Любой установленный бит указывает, что ожидается обслуживание прерывания. Бит прерывания очищается, когда процесс передается по соответствующему вектору прерывания.

* ГФС и канал 5 блока ЕРА в режиме сравнения могут вызвать это прерывание. Запись в PI_MASK разрешает источники прерываний, чтение PI_PEND определяет вызывающий прерывание источник.

Регистр ONES_REG

Адрес: 02_H

Состояние сброса: FFFF_H

Двухбайтный регистр ONES_REG всегда равен FFFF_H. Он используется как источник константы с единичными значениями битов для проведения операций сравнения (компарации).

Т а б л и ц а В.21 – Разряды единичного регистра ONES_REG

Номер разряда	Функция
15 – 0	Эти биты всегда содержат единицы и значение в регистре равно FFFF _H

Регистр P_x_DIR

(x = 2, 5, 7)

Адрес: таблица В.23

Состояние сброса: таблица В.23

Каждый вывод портов 2, 5, 7 может работать в любом стандартном режиме ввода-вывода: комплементарный выход, выход с открытым стоком или высокоимпедансный вход. Регистры ввода-вывода P2_DIR, P5_DIR, P7_DIR определяет направление ввода-вывода для каждого вывода портов 2, 5, 7. Установки для выхода с открытым стоком или для высокоимпедансного входа идентичны. При конфигурации выхода с открытым стоком требуется внешняя поддержка высокого уровня (pull-up).

Т а б л и ц а В.22 – Разряды регистров направления ввода-вывода P2_DIR, P5_DIR, P7_DIR

Номер бита	Мнемоника бита	Функция
7	P _x .7	Направление порта x. Каждый бит P _x _DIR управляет направлением соответствующего вывода. 0 – комплементарный выход (только выход), 1 – вход или выход с открытым стоком (вход, выход или вход-выход). Выходы с открытым стоком требуют внешний pull-up
6	P _x .6	
5	P _x .5	
4	P _x .4	
3	P _x .3	
2	P _x .2	
1	P _x .1	
0	P _x .0	

Т а б л и ц а В.23 – Адреса и значения сброса регистров P2_DIR, P5_DIR, P7_DIR

Имя регистра	Адрес	Значение после сброса
P2_DIR	1FD2 _H	FF _H
P5_DIR	1FF3 _H	FF _H
P7_DIR	1FD3 _H	FF _H

Регистр P_x_MODE**(x = 2, 5, 7)**

Адрес: таблица В.25

Состояние сброса: таблица В.25

Каждый разряд регистра режима P_x_MODE управляет соответствующей функцией вывода, как стандартного ввода-вывода порта 2, 5 или 7 или сигнала специальной функции.

Т а б л и ц а В.24 – Разряды регистров режима портов P2_MODE, P5_MODE, P7_MODE

Номер бита	Мнемоника бита	Функция
7	P _x .7	Режим порта x. Каждый бит P _x _MODE определяет, функционирует ли соответствующий вывод как стандартный вывод порта ввода-вывода или как сигнал специальной функции. 0 – стандартный вывод порта ввода – вывода, 1 – сигнал специальной функции
6	P _x .6	
5	P _x .5	
4	P _x .4	
3	P _x .3	
2	P _x .2	
1	P _x .1	
0	P _x .0	

Т а б л и ц а В.25 – Адреса и значения сброса регистров P2_MODE, P5_MODE, P7_MODE

Имя регистра	Адрес	Значение после сброса
P2_MODE	1FD0 _H	00 _H
P5_MODE	1FF1 _H	FF _H , если вывод не управляется; 80 _H , если на выводе EA# высокий уровень; A9 _H , если на выводе EA# низкий уровень
P7_MODE	1FD1 _H	00 _H

Т а б л и ц а В.26 – Сигналы специальных функций портов 2, 5, 7

Порт 2		Порт 5		Порт 7	
Вывод	Сигнал специальной функции	Вывод	Сигнал специальной функции	Вывод	Сигнал специальной функции
P2.7	COMP3	P5.7	BW	P7.7	FREQOUT
P2.6	COMP2/CPVER	P5.6	READY	P7.6	–
P2.5	COMP1/PACT#	P5.5	BHE#/WRH#	P7.5	–
P2.4	COMP0/AINC#	P5.4	ONCE#	P7.4	–
P2.3	CAPCOMP3	P5.3	RD#	P7.3	COMP5
P2.2	CAPCOMP2/PROG#	P5.2	WR#/WRL#	P7.2	COMP4
P2.1	CAPCOMP1/PALE#	P5.1	INST	P7.1	CAPCOMP5
P2.0	CAPCOMP0/PVER	P5.0	ALE/ADV#	P7.0	CAPCOMP4

Регистр P_x_PIN**(x = 0 – 5, 7)**

Адрес: таблица В.28

Состояние сброса: таблица В.28

Каждый разряд регистра входа P_x_PIN отображает текущее состояние соответствующего вывода независимо от его конфигурации.

Т а б л и ц а В.27 – Разряды регистров входа P_x_PIN (x = 0 – 5, 7) портов 2, 5, 7

Номер бита	Мнемоника бита	Функция
7	P _x .7	Вход порта x. Каждый бит P _x _PIN отражает текущее состояние соответствующего вывода независимо от конфигурации вывода
6	P _x .6	
5	P _x .5	
4	P _x .4	
3	P _x .3	
2	P _x .2	
1	P _x .1	
0	P _x .0	

Т а б л и ц а В.28 – Адреса и значения после сброса регистров P_x_PIN (x = 0 – 5, 7)

Имя регистра	Адрес	Значение после сброса
P0_PIN	1FA8 _H	FF _H , если вывод не используется
P1_PIN	1FA9 _H	FF _H , если вывод не используется
P2_PIN	1FD6 _H	FF _H , если вывод не используется
P3_PIN	1FFE _H	FF _H , если вывод не используется
P4_PIN	1FFF _H	FF _H , если вывод не используется
P5_PIN	1FF7 _H	FF _H , если вывод не используется
P7_PIN	1FD7 _H	XX _H

Регистр P_x_REG

(x = 2 – 5, 7)

Адрес: таблица В.30

Состояние сброса: таблица В.30

Для управления направлением движения данных (ввода или вывода) используются регистры P_x_REG.

Т а б л и ц а В.29 – Разряды регистров выхода данных портов P_x_REG (x = 0 – 5, 7)

Номер бита	Мнемоника бита	Функция
7	P _x .7	Выход данных порта x. Для входа необходимо установить соответствующий P _x _REG бит.
6	P _x .6	Для выхода написать данные, которые будут выведены каждым выводом в соответствующий бит P _x _REG. Когда вывод конфигурируется как стандартный вход-выход (P _x _MODE.y = 0), результат записи центрального процессора в P _x _REG немедленно видим на выводе. Когда вывод формируется как сигнал специальной функции (P _x _MODE.y = 1), соответствующая периферия на МК или компонент вне МК управляет выводом. Центральный процессор может все еще писать P _x _REG, но состояние вывода не меняется, пока он не переключен к его стандартной функции входа-выхода.
5	P _x .5	Эта особенность позволяет программному обеспечению формировать вывод как стандартный вход-выход (очистить P _x _MODE.y), инициализировать или переписывать значение вывода, затем формировать вывод как сигнал специальной функции (необходимо установить P _x _MODE.y). Этим способом инициализация, восстановление ошибки, обработка исключения и т. д. могут быть сделаны без изменения работы связанного периферийного устройства
4	P _x .4	
3	P _x .3	
2	P _x .2	
1	P _x .1	
0	P _x .0	

Т а б л и ц а В.30 – Адреса и значения после сброса регистров P_x_REG (x = 2 – 5, 7)

Имя регистра	Адрес	Значение после сброса
P2_REG	1FA8 _H	FF _H
P3_REG	1FD4 _H	FF _H
P4_REG	1FFC _H	FF _H
P5_REG	1FF5 _H	FF _H , если вывод не используется
P7_REG	1FD5 _H	FF _H

Регистр PI_MASK

Адрес: 1FBC_H

Состояние сброса: AA_H

Регистр маски периферийных прерываний (PI_MASK) разрешает или запрещает запросы прерываний, связанные с периферийными прерываниями (PI), и прерываниями по переполнению/антипереполнению таймеров (OVRTM).

Т а б л и ц а В.31 – Разряды регистра маски прерываний периферийных устройств (PI_MASK)

Номер бита	Мнемоника бита	Функция
7, 5, 3, 1	–	Зарезервированы для совместимости с будущими устройствами, записываются нули
6	COMP5	ЕРА сравнение, канал 5. Установка этого бита разрешает прерывание для канала 5 блока ЕРА в режиме сравнения. Прерывания блока ГФС и ЕРА сравнение Канал 5 ассоциируются с периферийными прерываниями (PI). Установка INT_MASK.5 разрешает PI
4	WG	Генератор формы сигнала. Установка бита разрешает прерывание генератора формы сигнала. Прерывания блока ГФС и ЕРА сравнение Канал 5 ассоциируются с периферийными прерываниями (PI). Установка INT_MASK.5 разрешает PI
2	OVRTM2	Переполнение/антипереполнение таймера 2. Установка этого бита разрешает прерывание переполнения/антипереполнения таймера 2. Прерывания переполнения/антипереполнения таймера 2 и таймера 1 связаны с прерыванием переполнения/антипереполнения таймера (OVRTM). Установка INT_MASK.0 разрешает OVRTM
0	OVRTM1	Переполнение/антипереполнение таймера 1. Установка этого бита разрешает прерывание переполнения/антипереполнения таймера 1. Прерывания переполнения/антипереполнения таймера 2 и таймера 1 связаны с прерыванием переполнения/антипереполнения таймера (OVRTM). Установка INT_MASK.0 разрешает OVRTM

Регистр PI_PEND

Адрес: 1FBЕ_H

Состояние сброса: АА_H

Когда аппаратные средства обнаруживают ожидаемое периферийное прерывание или прерывание таймера, они устанавливают соответствующий бит в регистрах ожидания прерывания (INT_PEND или INT_PEND1) и в регистре ожидания периферийного прерывания (PI_PEND). Когда вектор выбран, аппаратные средства очищают соответствующий бит INT_PEND/INT_PEND1. Чтение этого регистра очищает все биты PI_PEND. Программное обеспечение может произвести прерывание установкой бита PI_PEND.

Т а б л и ц а В.32 – Разряды регистра ожидания прерывания периферийных устройств PI_PEND

Номер бита	Мнемоника бита	Функция
7, 6, 5, 3, 1	–	Резервные. Эти биты не используются
6	COMP5	ЕРА сравнение, канал 5. Установка этого бита показывает ожидание прерывание для канала 5 блока ЕРА в режиме сравнения. Прерывания блока ГФС и ЕРА сравнение Канал 5 ассоциируются с периферийными прерываниями (PI). Установка INT_MASK.5 разрешает PI. Установка PI_MASK.6 позволяет COMP5
4	WG	Генератор формы сигнала. Установка бита указывает, что ожидается прерывание
2	OVRTM2	Переполнение/антипереполнение таймера 2. Установленный бит указывает на ожидаемое прерывание по переполнению/антипереполнению таймера 2. Прерывания по переполнению/антипереполнению таймера 2 и таймера 1 связаны с прерыванием по переполнению/антипереполнению таймера (OVRTM). Установка INT_MASK.0 позволяет OVRTM. Установка PI_MASK.2 позволяет OVRTM2
0	OVRTM1	Переполнение/антипереполнение таймера 1. Установленный бит указывает на ожидаемое прерывание по переполнению/ антипереполнению таймера 1 и таймера 2. Они связаны с прерыванием по переполнению/антипереполнению таймера (OVRTM). Установка INT_MASK.0 позволяет OVRTM. Установка PI_MASK.0 позволяет OVRTM1

Регистр PPW

Нет прямого доступа.

Чтобы правильно определить PPW_VALUE для нужной частоты устройства, используется формула приведенная в таблице В.33.

PPW не имеет прямого доступа.

Результат округляют до следующего старшего целого числа.

Регистр ширины программирующего импульса загружается из внешней памяти EPROM (адреса 14_H и 15_H) в режиме автопрограммирования. PPW_VALUE определяет ширину программирующего импульса.

Т а б л и ц а В.33 – Разряды регистра PPW

Номер бита	Обозначение	Функция
15 – 0	PPW15 – 0	PPW_VALUE. Это значение устанавливает ширину программирующего импульса для режима автопрограммирования. Чтобы вычислить PPW_VALUE, используют формулу $PPW_VALUE = 62,5 \times 2 \times F_{BQ1} ,$ где PPW_VALUE – 16-битное слово; F _{BQ1} – входная частота в МГц

Регистр PSW

Нет прямого доступа.

Слово состояния процессора состоит из двух байтов. Старший байт является словом состояния или статуса, а в качестве младшего байта выступает регистр INT_MASK. Слово состояния содержит бит PSW.1, который является битом разрешения или запрета обслуживания всех маскируемых прерываний, также содержит бит PSW.2, который является битом разрешения или запрета сервера периферийных транзакций (PTS), а также содержит шесть логических (Булевых) флагов, отображающих состояние пользовательской программы.

Слово состояния, как часть регистра PSW, не имеет прямого доступа. Для доступа к слову состояния необходимо поместить его содержание в стек командой PUSHF, а затем извлечь значение в любой регистр. Команды PUSHF и PUSHA сохраняют PSW в системный стек и затем очищают PSW. Команды POPF и POPA восстанавливают его.

Т а б л и ц а В.34 – Разряды регистра PSW

Номер бита	Мнемоника	Функция
1	2	3
15	Z	Zero Flag – «Флаг Нуля». Этот флаг устанавливается для индикации нулевого результата операции. Для очень точных вычислений флаг нуля не может быть установлен, если команды используют бит переноса из предыдущей операции (например, ADDC, SUBC). Эти команды даже могут очистить этот бит. Это происходит потому, что флаг нуля должен отражать результат всей операции, а не только последнего вычисления. Например, если результат сложения младших слов двух двойных слов равен нулю, флаг нуля будет установлен. Когда старшие слова складываются с использованием команды ADDC, флаг вновь устанавливается, если результат нулевой, и очищается, если иначе
14	N	Negative Flag – «Флаг Отрицательного Результата». Этот флаг сигнализирует, что результат операции отрицательный. Состояние флага корректно в случае переполнения. Для всех сдвиговых операций и команды NORML флаг становится равным старшему значащему биту результата, даже если сдвиг нулевой
13	V	Overflow Flag – «Флаг Переполнения». Этот флаг сигнализирует, что результат слишком большой, чтобы его представить корректно в доступном пространстве. Для операций сдвига (SHL, SHLB, SHLL) флаг устанавливается, если старший значащий бит операнда изменяется в течение сдвига. Для операций деления частное запоминается в младшую половину операнда назначения, и в старшую половину записывается остаток. Флаг переполнения устанавливается, если частное выходит за границы младшей части операнда назначения

Окончание таблицы В.34

1	2	3
12	VT	<p>Overflow-Trap Flag – «Дополнительный Флаг Переполнения». Этот флаг устанавливается, когда устанавливается флаг переполнения, однако он очищается только командами CLRVT, JVT, JNVT. Это позволяет проверять на возможное переполнение в конце последовательности связанных арифметических операций, что более эффективно, чем проверка флага переполнения в конце каждой операции</p>
11	C	<p>Carry Flag – «Флаг Переноса». Установка этого флага показывает арифметический перенос или последний бит выдвинутого операнда. Он очищается, если операция вычитания завершается заемом. Обычно результат округлен в большую сторону, если этот бит установлен. Этот бит позволяет выполнять более точное разрешение при округлениях</p>
10	PSE	<p>Разрешение PTS. Бит глобального разрешения или запрета блока PTS. Команда EPTS устанавливает этот бит, а команда DPTS обнуляет его 0 – запрет PTS, 1 – разрешение PTS</p>
9	I	<p>Запрет прерываний (глобальный). Это бит глобального разрешения или запрета применения всех маскируемых прерываний. Биты в регистрах INT_MASK и INT_MASK1 индивидуально разрешают или запрещают прерывания. Команда EI устанавливает этот бит, а команда DI обнуляет его</p>
8	ST	<p>Sticky Bit Flag – «Флаг Дополнительного Бита». Этот флаг устанавливается для индикации того, что во время сдвига вправо единичный бит сдвигается во флаг переноса (C) и затем выдвигается из него. Он может быть использован вместе с флагом переноса для более высокой точности результата при округлении</p>

Регистр PTSSEL

Адрес: 0004_H

Состояние сброса: 0000_H

Регистр выбора PTS (PTSSEL) выбирает или PTS микропрограмму, или процедуру обслуживания стандартного прерывания для каждого запроса прерывания. Установка бита выбирает PTS микропрограмму; очистка бита выбирает процедуру обслуживания стандартного прерывания. Когда PTSCOUNT становится равным нулю, аппаратные средства очищают соответственно PTSSEL бит. PTSSEL бит должен быть установлен в соответствии с техническим описанием для перевыбора канала PTS.

Т а б л и ц а В.35 – Разряды регистра выбора PTS (PTSSEL)

Номер бита	Мнемоника бита	Прерывание, обслуживаемое микропрограммой PTS	Адреса векторов прерываний PTS
15	–	Зарезервированный; для совместимости с будущими устройствами записать ноль в этот бит	–
14	EXTINT	Вывод EXTINT	205C _H
13	PI*	Мультиплексное периферийное прерывание	205A _H
12	CAPCOMP5	ЕРА захват/сравнение канал 5	2058 _H
11	COMP4	ЕРА сравнение канал 4	2056 _H
10	CAPCOMP4	ЕРА захват/сравнение канал 4	2054 _H
9	COMP3	ЕРА сравнение канал 3	2052 _H
8	CAPCOMP3	ЕРА захват/сравнение канал 3	2050 _H
7	COMP2	ЕРА сравнение канал 2	204E _H
6	CAPCOMP2	ЕРА захват/сравнение канал 2	204C _H
5	COMP1	ЕРА сравнение канал 1	204A _H
4	CAPCOMP1	ЕРА захват/сравнение канал 1	2048 _H
3	COMP0	ЕРА сравнение канал 0	2046 _H
2	CAPCOMP0	ЕРА захват/сравнение канал 0	2044 _H
1	AD	АЦ преобразование завершено	2042 _H
0	OVRTM *	Переполнение/антипереполнение таймера	2040 _H

Примечание – Установка бита разрешает обслуживание соответствующего прерывания микропрограммой PTS.

* Обслуживание PTS некорректно для мультиплексных прерываний, так как PTS не может определить источник прерываний.

Регистр PTSSRV

Адрес: 0006_H

Состояние сброса: 0000_H

Регистр обслуживания PTS (PTSSRV) используется аппаратными средствами, чтобы указать, что последнее PTS прерывание обслужено программой PTS. Когда PTSCOUNT равен нулю, аппаратные средства очищают соответствующий бит PTSSSEL и устанавливают бит PTSSRV, который запрашивает end-of-PTS прерывание. Когда прерывание end-of-PTS вызвано, аппаратные средства очищают PTSSRV бит. Бит в регистре PTSSSEL надо установить для повторного разрешения канала PTS.

Т а б л и ц а В.36 – Разряды регистра PTSSRV обслуживания PTS

Номер бита	Мнемоника бита	Прерывание, обслуживаемое микропрограммой PTS	Адреса векторов прерываний PTS
15	–	Зарезервированный; для совместимости с будущими устройствами записать ноль в этот бит	–
14	EXTINT	Вывод EXTINT	205C _H
13	PI*	Мультиплексное периферийное прерывание	205A _H
12	CAPCOMP5	ЕРА захват/сравнение канал 5	2058 _H
11	COMP4	ЕРА сравнение канал 4	2056 _H
10	CAPCOMP4	ЕРА захват/сравнение канал 4	2054 _H
9	COMP3	ЕРА сравнение канал 3	2052 _H
8	CAPCOMP3	ЕРА захват/сравнение канал 3	2050 _H
7	COMP2	ЕРА сравнение канал 2	204E _H
6	CAPCOMP2	ЕРА захват/сравнение канал 2	204C _H
5	COMP1	ЕРА сравнение канал 1	204A _H
4	CAPCOMP1	ЕРА захват/сравнение канал 1	2048 _H
3	COMP0	ЕРА сравнение канал 0	2046 _H
2	CAPCOMP0	ЕРА захват/сравнение канал 0	2044 _H
1	AD	А/D преобразование завершено	2042 _H
0	OVRTM *	Переполнение/антипереполнение таймера	2040 _H

Примечание – Бит для соответствующего прерывания устанавливается аппаратно через стандартный вектор запроса прерывания end-of-PTS.

* Обслуживание PTS не применяется для мультиплексных прерываний, потому что PTS не может определить источники этих прерываний.

Регистр PWM_COUNT (только чтение)Адрес: 1FB6_HСостояние сброса: 00_H

Регистр периода счета ШИМ (PWM_COUNT) обеспечивает текущее значение периода счетчика обратного счета.

Т а б л и ц а В.37 – Разряды регистра периода счета ШИМ (PWM_COUNT)

Номер разряда	Функция
7 – 0	Значение периода счета PWM. Регистр содержит текущее значение периода счетчика обратного счета

Регистр PWM_PERIODАдрес: 1FB4_HСостояние сброса: 00_H

Регистр периода ШИМ (PWM_PERIOD) управляет периодом выходных ШИМ сигналов. Он содержит значение, которое определяет число состояний счета, необходимое для инкрементирования счетчика ШИМ. Значение PWM_PERIOD загружается в регистр счетчика периода ШИМ всякий раз, когда значение в счетчике равно нулю.

Т а б л и ц а В.38 – Разряды регистра периода ШИМ PWM_PERIOD

Номер разряда	Функция
7 – 0	Период сигнала ШИМ. Этот регистр управляет периодом сигнала на выводах ШИМ. Значение PWM_PERIOD загружается в регистр счетчика периода ШИМ всякий раз, когда значение в счетчике равно нулю

Регистр PWMx_CONTROL, (x = 0, 1)Адреса: 1FB0_H, 1FB2_HСостояние сброса: 00_H

Регистр управления ШИМ (PWMx_CONTROL) определяет скважность сигнала ШИМ для канала x. Ноль, загруженный в этот регистр ШИМ, устанавливает состояние низкого уровня сигнала (постоянное, скважность равна 0 %). FF_H в этом регистре устанавливает для ШИМ максимальную скважность (99,6 %) и состояние постоянного высокого уровня сигнала.

Т а б л и ц а В.39 – Разряды регистра управления ШИМ (PWMx_CONTROL)

Номер разряда	Функция
7 – 0	Скважность ШИМ. Этот регистр управляет скважностью ШИМ. Ноль, загруженный в этот регистр ШИМ, устанавливает состояние постоянного низкого уровня сигнала (скважность равна 0 %). FF _H в этом регистре устанавливает для ШИМ максимальную скважность (99,6 %) и состояние постоянного высокого уровня сигнала

Регистр SPАдрес: 0018_HСостояние сброса: XXXX_H

Системный указатель стека может указывать на любой адрес во внутренней или внешней памяти. Он должен быть выровненным словом и должен быть задан перед применением. Указатель стека декрементируется перед выполнением команды PUSH и инкрементируется после команды POP. Указатель стека должен инициализироваться в два байта (память 64 Кбайт) или в четыре байта (память 1 Мбайт) выше самой старшей ячейки стека. Если стековые операции не выполняются, ячейки с адресами 18_H и 19_H могут быть использованы, как стандартные регистры.

Т а б л и ц а В.40 – Разряды регистра указателя стека системы SP

Номер разряда	Функция
15 – 0	Указатель стека. Регистр представляет собой указатель стека системы

Регистр T1CONTROL

Адрес: 1F78_H

Состояние после сброса: 00_H

Регистр управления таймера 1 (T1CONTROL) определяет источник синхроимпульсов, направление счета и период счета для таймера 1.

Таблица В.41 – Разряды регистра управления таймера 1 (T1CONTROL)

Номер разряда	Мнемоника	Функция
1	2	3
7	CE	Разрешение счета. Этот бит разрешает или запрещает таймеры. После сброса таймеры запрещены и не разрешен их запуск. 0 – запрещает таймеры, 1 – разрешает таймеры
6	UD	Прямой/обратный счет. Этот бит определяет направление счета таймера в выбранных режимах (см. биты режима, M2:0). 0 – счет в обратном направлении, 1 – счет в прямом направлении
5 – 3	M2 – M0	Биты управления синхронизацией и направлением счета ЕРА. Эти биты определяют источник синхроимпульсов и источник сигнала, управляющего направлением счета. M2 M1 M0 Источник синхроимпульсов Источник направления 0 0 0 F _{BQ1} /4 UD бит (T1CONTROL.6) X 0 1 T1CLK вход ¹⁾ UD бит (T1CONTROL.6) 0 1 0 F _{BQ1} /4 T1DIR вход 0 1 1 T1CLK вход ¹⁾ T1DIR вход 1 1 1 квадратурная синхронизация, использующая T1CLK и T1DIR
2 – 0	P2 – P0	Биты коэффициента деления частоты синхроимпульсов ЕРА (Prescaler) Эти биты определяют prescaler_divisor для синхроимпульсов. P2 P1 P0 Деление Разрешение ²⁾ 0 0 0 на 1 (запрещено) 250 нс 0 0 1 на 2 500 нс 0 1 0 на 4 1 мкс 0 1 1 на 8 2 мкс 1 0 0 на 16 4 мкс 1 0 1 на 32 8 мкс 1 1 0 на 64 16 мкс 1 1 1 разрешение T1RELOAD –

¹⁾ Если выбрана внешняя синхронизация, таймер считает и на положительный и отрицательный фронт синхроимпульсов.

²⁾ Для 16 МГц. Для других частот используют формулу

$$\text{разрешение} = \frac{4 \times \text{prescaler_divisor}}{F_{BQ1}}$$

Регистр T1RELOAD

Адрес: 1F72_H

Состояние после сброса: XXXX_H

Этот регистр содержит значение реинициализации для таймера 1. При переполнении/антипереполнении таймера 1 загружается значение T1RELOAD в регистр TIMER1, если и квадратурная синхронизация, и перезагрузка разрешены (T1CONTROL.5 – 0 = 1).

Т а б л и ц а В.42 – Разряды регистра перезагрузки таймера 1 (T1RELOAD)

Номер разряда	Функция
15 – 0	Значение перезагрузки для TIMER1. Для реинициализации TIMER1 необходимое значение загружается в этот регистр

Регистр TIMERx (x = 1, 2)

Адреса: 1F72_H, 1F7E_H

Состояние после сброса: 0000_H

Эти регистры содержат текущие значения таймера 1 и 2. Эти регистры могут быть записаны, позволяя этим инициализировать таймеры значениями отличными от нуля.

Т а б л и ц а В.43 – Разряды регистров значения таймера 1 и таймера 2 (TIMERx)

Номер разряда	Функция
15 – 0	Значение TIMERx (x = 1, 2). Можно читать текущее значение таймеров 1 и 2 или записывать новые значения в эти регистры

Регистр T2CONTROL

Адрес: 1F7C_H

Состояние после сброса: 00_H

Регистр управления таймера 2 (T2CONTROL) определяет источник синхроимпульсов, направление и диапазон счёта для таймера 2.

Т а б л и ц а В.44 – Разряды регистра управления таймера 2 (T2CONTROL)

Номер бита	Мнемоника	Функция																																													
7	CE	Разрешение счёта. Этот бит разрешает или запрещает таймеры. После сброса таймеры запрещены и не разрешен запуск. 0 запрещает таймер 1 разрешает таймер																																													
6	UD	Прямой/обратный счёт. Этот бит определяет направление счёта таймера в выбранных режимах (биты режима M2:0). 0 счёт в обратном направлении 1 счёт в прямом направлении																																													
5 – 3	M2 – M0	Биты управления синхронизацией и направлением счёта ЕРА. Эти биты определяют источник синхроимпульсов и источник направления счёта. <table border="0"> <tr> <td>M2</td> <td>M1</td> <td>M0</td> <td>Источник синхроимпульсов</td> <td>Источник направления</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>F_{BQ1}/4</td> <td>UD бит (T2CONTROL.6)</td> </tr> <tr> <td>X</td> <td>0</td> <td>1</td> <td>зарезервирован</td> <td>–</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>зарезервирован</td> <td>–</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>зарезервирован</td> <td>–</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>переполнение таймера 1</td> <td>UD бит (T2CONTROL.6)</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>переполнение таймера 1</td> <td>как у таймера 1</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>зарезервирован</td> <td>–</td> </tr> </table>	M2	M1	M0	Источник синхроимпульсов	Источник направления	0	0	0	F _{BQ1} /4	UD бит (T2CONTROL.6)	X	0	1	зарезервирован	–	0	1	0	зарезервирован	–	0	1	1	зарезервирован	–	1	0	0	переполнение таймера 1	UD бит (T2CONTROL.6)	1	1	0	переполнение таймера 1	как у таймера 1	1	1	1	зарезервирован	–					
M2	M1	M0	Источник синхроимпульсов	Источник направления																																											
0	0	0	F _{BQ1} /4	UD бит (T2CONTROL.6)																																											
X	0	1	зарезервирован	–																																											
0	1	0	зарезервирован	–																																											
0	1	1	зарезервирован	–																																											
1	0	0	переполнение таймера 1	UD бит (T2CONTROL.6)																																											
1	1	0	переполнение таймера 1	как у таймера 1																																											
1	1	1	зарезервирован	–																																											
2 – 0	P2 – P0	Биты коэффициента деления частоты синхроимпульсов ЕРА (Prescaler_divisor). Эти биты определяют prescaler_divisor для синхроимпульсов. <table border="0"> <tr> <td>P2</td> <td>P1</td> <td>P0</td> <td>Деление</td> <td>Разрешение ¹⁾</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>на 1 (запрещение)</td> <td>250 нс</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>на 2</td> <td>500 нс</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>на 4</td> <td>1 мкс</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>на 8</td> <td>2 мкс</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>на 16</td> <td>4 мкс</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>на 32</td> <td>8 мкс</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>на 64</td> <td>16 мкс</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>зарезервировано</td> <td>–</td> </tr> </table> <p>¹⁾ Для 16 МГц. Для других частот используют формулу</p> $\text{Разрешение} = \frac{4 \times \text{prescaler_divisor}}{F_{BQ1}}$	P2	P1	P0	Деление	Разрешение ¹⁾	0	0	0	на 1 (запрещение)	250 нс	0	0	1	на 2	500 нс	0	1	0	на 4	1 мкс	0	1	1	на 8	2 мкс	1	0	0	на 16	4 мкс	1	0	1	на 32	8 мкс	1	1	0	на 64	16 мкс	1	1	1	зарезервировано	–
P2	P1	P0	Деление	Разрешение ¹⁾																																											
0	0	0	на 1 (запрещение)	250 нс																																											
0	0	1	на 2	500 нс																																											
0	1	0	на 4	1 мкс																																											
0	1	1	на 8	2 мкс																																											
1	0	0	на 16	4 мкс																																											
1	0	1	на 32	8 мкс																																											
1	1	0	на 64	16 мкс																																											
1	1	1	зарезервировано	–																																											

Регистр USFR

Адрес: 1FF6_H

Состояние после сброса: 02_H

Нестираемый регистр USFR содержит два бита, которые запрещают поток данных или поток команд из внешней памяти и другое, что определяет неисправный осциллятор. Эти биты можно запрограммировать, но никогда невозможно стереть.

Т а б л и ц а В.45 – Разряды регистра USFR

Номер бита	Обозначение	Функция
7 – 4	–	Зарезервированы. Всегда записывать нули
3	DEI	Установка этого бита запрещает выборку команд из внешней памяти. Любая попытка загрузить внешний адрес приводит к сбросу устройства
2	DED	Установка этого бита запрещает обращение к внешним данным. Любая попытка записи или чтения из внешней памяти приведет к сбросу
1, 0	–	Зарезервированы. Всегда записывать нули

Регистр WACHDOG

Адрес: 000A_H

Состояние после сброса: XX_H

Если регистр WACHDOG не очищать каждые 64К циклов, то сторожевой таймер выполняет рестарт микроконтроллера. Для очистки регистра сторожевого таймера по адресу 000A_H необходимо записать последовательно два байта, сначала 1E_H и затем E1_H. Очистка этого регистра в начале работы позволяет сторожевому таймеру начинать свою работу со значения 0000_H с последующим инкрементом в каждом цикле микроконтроллера.

Бит WDE регистра CCR1 управляет разрешением или запретом работы сторожевого таймера пока регистр WACHDOG очищен. Очистка регистра WDE активирует сторожевой таймер. Установка бита WDE делает сторожевой таймер неактивным, однако его можно активировать очисткой регистра WACHDOG.

После разрешения работы сторожевой таймер может быть запрещен только сбросом микроконтроллера.

Т а б л и ц а В.46 – Разряды регистра сторожевого таймера (WACHDOG)

Номер разряда	Функция
7 – 0	Значение WACHDOG. Регистр содержит восемь старших значащих бит текущего значения сторожевого таймера

Регистр WG_COMP_x (x = 1, 2, 3)

Адреса: FC2_H, 1FC4_H, 1FC6_H

Состояние сброса: 0000_H

Регистр сравнения фаз (WG_COMP_x) управляет скважностью сигнала каждой фазы. Записывается значение в каждый регистр сравнения фаз, чтобы определить отрезок времени, в течение которого соответствующие выходы останутся установленными.

Изменение значения WG_RELOAD изменяет и период несущей и скважность, потому что выходы остаются установленными в течение постоянного отрезка времени, в то время как счетчик требует большего времени, чтобы закончить цикл. Чтобы изменить период опорного сигнала, не изменяя скважность, необходимо пропорционально изменить WG_RELOAD и WG_COMP_x в одно и то же время, немедленно после прерывания.

Т а б л и ц а В.47 – Разряды регистра сравнения фаз (WG_COMP_x)

Номер бита	Функция
15 – 0	<p>Сравнение. Эти биты определяют отрезок времени, в течение которого соответствующие выходы установлены. Используются следующие формулы, чтобы вычислить время установленного выхода и скважность:</p> $T_{\text{OUTPUT}} = \frac{\text{множитель} \times \text{WG_RELOAD}}{F_{\text{BQ1}}},$ $\text{Скважность} = \frac{\text{WG_COMP}_x}{\text{WG_RELOAD}} \times 100\%,$ <p>где T_{OUTPUT} – полное время, когда выход установлен, мкс; F_{BQ1} – частота входного сигнала на входе BQ1, МГц; «множитель» равен четырем для «центрированных» режимов, «множитель» равен двум для «фронтových» режимов; WG_RELOAD – 16-битное значение $\text{WG_RELOAD} \geq \text{WG_COMP}_x$; WG_COMP_x – 16-битное значение $\text{WG_COMP}_x \leq \text{WG_RELOAD}$</p>

Регистр WG_CONTROL

Адрес: 1FCC_H

Состояние сброса: 00C0_H

Регистр WG_CONTROL управления генератора формы сигнала ГФС управляет режимом операции, временем простоя и направлением счета, а также разрешает и запрещает счетчик.

Т а б л и ц а В.48 – Разряды регистра управления ГФС (WG_CONTROL)

Номер бита	Мнемоника	Функция
15	–	Зарезервирован. Для совместимости с последующими устройствами записывается ноль в этот бит
14 – 12	M2 – M0	Режим работы. Это поле управляет режимом работы ГФС. M2 M1 M0 Режим 0 0 0 0 – «центрированный»; регистры обновляются однажды 0 0 1 1 – «центрированный»; регистры обновляются дважды 0 1 0 2 – «фронтной»; регистры обновляются однажды 0 1 1 3 – «фронтной»; регистры обновляются дважды
11	CS	Состояние счётчика. Этот бит только для чтения указывает, считает ли счетчик вверх или в обратном порядке. 0 счет вниз, 1 счет вверх
10	EC	Разрешение счетчика. Этот бит запускает или останавливает счетчик. 0 запретить (остановить) счетчик, 1 разрешить (запустить) счетчик
9 – 0	DT9 – DT0	«Время простоя» (Dead Time). Это поле определяет «время простоя» для всех трех фаз. Используется следующая формула, чтобы вычислить соответствующее значение DT_VALUE: $DT_VALUE = T_{DEAD} \times F_{BQ1} / 2,$ где T_{DEAD} – «время простоя», мкс; F_{BQ1} – частота входного сигнала на входе BQ1, МГц

Регистр WG_COUNTER

Адрес: 1FCA_H

Состояние сброса: XXXX_H

Читается регистр-счетчик генератора формы сигнала (WG_COUNTER), чтобы определить текущее значение счётчика.

Т а б л и ц а В.49 – Разряды регистра-счётчика WG_COUNTER генератора формы сигнала

Номер бита	Функция
15 – 0	Значение счётчика. Этот регистр отражает текущее значение счётчика

WG_OUTPUT (Порт 6)

Адрес: 1FC0_H

Состояние сброса: 0000_H

Регистр WG_OUTPUT управляет функциями порта 6. Если используется порт 6 для выходов общего назначения, то необходимо записать C0_H (для активно-высоких выводов) или 00_H (для активно-низких выводов) в старший байт WG_OUTPUT и записать желаемые значения выводов в младший байт. Более подробно регистр WG_OUTPUT описан в разделе 8 «Генератор формы сигнала» или в разделе 9 «Широтно-импульсный модулятор».

15								8
OP1	OP0	–	M7	M6	M5,M4	M3,M2	M1,M0	
7								0
D7	D6	D5	D4	D3	D2	D1	D0	

Рисунок В.1 – Регистр управления выходного порта WG_OUTPUT

Т а б л и ц а В.50 – Поля регистра управления выходного порта WG_OUTPUT

Разряд	Мнемоника	Функция
15, 14	OP1, OP0	Полярность вывода. Эти биты выбирают полярность вывода. «0» – активный низкий уровень, «1» – активный высокий уровень
13	–	Зарезервирован. Для совместимости с будущими устройствами необходимо записать ноль в этот бит
12 – 8	M7 – M0	Режим. Эти биты выбирают или периферийную функцию, или функцию выхода общего назначения. Необходимо очистить эти биты для выхода общего назначения
7 – 0	D7 – D0	Данные. В режиме выхода общего назначения эти биты выставляются на выводах. Желаемые значения записывают в эти биты (биты 7 – 0 соответствуют выводам P6.7 – P6.0)

Регистр WG_OUTPUT (Генератор формы сигнала)

Адрес: 1FC0_H

Состояние сброса: 0000_H

Регистр порта 6 управления функцией выхода генератора формы сигнала (WG_OUTPUT) управляет функционированием выходов генератора формы сигнала (ГФС) и блока ШИМ. И генератор формы сигнала, и блок ШИМ совмещают выводы с портом 6. Наличие битов управления в отдельном регистре позволяет формировать все выводы порта записью единственного регистра WG_OUTPUT.

Т а б л и ц а В.51 – Регистр генератора формы сигнала (WG_OUTPUT) – конфигурация выводов порта 6

Номер разряда	Мнемоника	Функция
1	2	3
15	OP1	Полярность выхода. Выбор полярности для выходов отрицательных фаз WG1#, WG2#, WG3#. 0 – выходы с активным низким уровнем, 1 – выходы с активным высоким уровнем
14	OP0	Полярность выхода. Выбор полярности для выходов положительных фаз WG1, WG2, WG3. 0 – выходы с активным низким уровнем, 1 – выходы с активным высоким уровнем
13	SYNC	Синхронизация. Выбор обновления регистра WG_OUTPUT. Обновление синхронизировано с событием или выполняется немедленно. 0 – обновление WG_OUTPUT немедленно, 1 – синхронизация обновления WG_OUTPUT с событием. Для гарантии того, чтобы выходы находились в определенном состоянии при старте ГФС, необходимо очистить этот бит и затем установить его позже, если необходимо синхронизировать WG_OUTPUT по событию
12	PE7	P6.7/PWM1 функция. Выбирает или функцию порта, или ШИМ функцию вывода P6.7/PWM1. 1 – PWM1 0 – P6.7
11	PE6	P6.6/PWM0 функция. Выбирает или функцию порта, или ШИМ функцию вывода P6.6/PWM0. 1 – PWM0 0 – P6.6

Окончание таблицы В.51

1	2	3
10	PH3.2	Функция фазы 3. Выбирает или функцию порта, или функцию вывода ГФС для P6.4/WG3# и P6.5/WG3. 0 P6.4, P6.5 1 WG3#, WG3
9	PH2.2	Функция фазы 2. Выбирает или функцию порта, или функцию вывода ГФС для P6.2/WG2# и P6.3/WG2. 0 P6.2, P6.3 1 WG2#, WG2
8	PH1.2	Функция фазы 1. Выбирает или функцию порта, или функцию вывода ГФС для P6.0/WG1# и P6.1/WG1. 0 P6.0, P6.1 1 WG1#, WG1
7	P7	P6.7/PWM1 значение. В этот бит записывают желаемое значение P6.7/PWM1
6	P6	P6.6/PWM0 значение. В этот бит записывают желаемое значение P6.6/PWM0
5, 4	PH3.1, PH3.0	P6.4/WG3#, P6.5/WG3 значения. В этот бит записывают желаемое значение
3, 2	PH2.1, PH2.0	P6.2/WG2 #, P6.3/WG2 значения. В этот бит записывают желаемое значение
1, 0	PH1.1, PH1.0	P6.0/WG1 #, P6.1/WG1 значения. В этот бит записывают желаемое значение

Регистр WG_PROTECT

Адрес: 1FCE_H

Состояние после сброса: F0_H

Регистр защиты ГФС (WG_PROTECT) разрешает и запрещает выходы и схему защиты. Он также выбирает прерывания EXTINT или по уровню, или по фронту и выбирает, который уровень или фронт произведет запрос на прерывание EXTINT.

Т а б л и ц а В.52 – Разряды регистра защиты ГФС (WG_PROTECT)

Номер бита	Мнемоника	Функция
7 – 5	–	Зарезервирован. Для совместимости с будущими устройствами записывать нули в эти биты
3, 2	ES, IT	Разрешение выборки и тип прерывания. Бит ES выбирает, что схема защиты производит выборку по уровню сигнала EXTINT или детектирует перепад сигнала (фронт), в то время как бит IT задаёт, какое значение фронта или уровня вызывает запрос на прерывание. Возможные комбинации следующие: ES IT Событие 0 0 задний фронт 0 1 передний фронт 1 0 низкий уровень 1 1 высокий уровень
1	DP	Запрещение защиты. Этот бит разрешает и запрещает схему защиты. 0 – разрешение защиты, 1 – запрещение защиты
0	EO	Разрешение выходов. Этот бит разрешает и запрещает выходы. 0 – запрещение выходов, 1 – разрешение выходов

Регистр WG_RELOAD

Адрес: 1FC8_H

Состояние сброса: 0000_H

Регистр перезагрузки ГФС (WG_RELOAD) и регистры сравнения фаз (WG_COMPx) управляют периодом опорного сигнала (несущей) и скважностью. Запись значения в регистр перезагрузки устанавливает период опорного сигнала.

При изменении значения WG_RELOAD изменяются и период опорного сигнала, и скважность, потому что выводы остаются установленными в течение постоянного отрезка времени, в то время как счетчик требует большего времени, чтобы закончить цикл. Чтобы изменить период несущей, не изменяя скважность, необходимо пропорционально изменить и WG_RELOAD, и WG_COMPx в одно и то же время, немедленно после прерывания.

Т а б л и ц а В.53 – Разряды регистра WG_RELOAD перезагрузки ГФС

Номер бита	Функция
15 – 0	<p>Перезагрузка. Этот регистр определяет период опорного сигнала (несущей). Используются следующие формулы, чтобы вычислить период несущей и скважность:</p> $T_{\text{CARRIER}} = \frac{\text{множитель} \times \text{WG_RELOAD}}{F_{\text{BQ1}}},$ $\text{Скважность} = \frac{\text{WG_COMPx}}{\text{WG_RELOAD}} \times 100\%,$ <p>где T_{CARRIER} – период несущей, мкс; F_{BQ1} – частота входного сигнала на входе BQ1, МГц; «множитель» равен четырем для «центрированных» режимов; «множитель» равен двум для «фронтowych» режимов; WG_RELOAD – 16-битное значение $\text{WG_RELOAD} \geq \text{WG_COMPx}$; WG_COMPx – 16-битное значение $\text{WG_COMPx} \leq \text{WG_RELOAD}$</p>

Регистр WSR

Адрес: 0014_H

Состояние сброса: 00_H

Регистр выбора окна (WSR) помещает секции оперативной памяти в вершину младшего регистрового файла с 32-, 64- или 128-байтовым приращением. PUSHA сохраняет этот регистр в стеке, а POPA восстанавливает его.

Т а б л и ц а В.54 – Разряды регистра выбора окна (WSR)

Номер бита	Мнемоника бита	Функция
7	–	Зарезервирован; для совместимости с будущими устройствами записывается «0»
6 – 0	W6 – W0	Выбор окна. Эти биты определяют размер окна и номер. В таблице В.55 приводятся значения WSR и прямые адреса для оконной адресации регистров SFR

Т а б л и ц а В.55 – Задание значения WSR и прямые адреса для оконной адресации регистров SFR

Мнемоника регистра	Адрес регистра	32-байтные окна (00E0 _H – 00FF _H)		64-байтные окна (00C0 _H – 00FF _H)		128-байтные окна (0080 _H – 00FF _H)	
		WSR	Прямой адрес	WSR	Прямой адрес	WSR	Прямой адрес
1	2	3	4	5	6	7	8
AD_COMMAND	1FAC _H	7D _H	00EC _H	3E _H	00EC _H	1F _H	00AC _H
AD_RESULT	1FAA _H	7D _H	00EA _H	3E _H	00EA _H	1F _H	00AA _H
AD_TEST	1FAE _H	7D _H	00EE _H	3E _H	00EE _H	1F _H	00AE _H
AD_TIME	1FAF _H	7D _H	00EF _H	3E _H	00EF _H	1F _H	00AF _H
COMP0_CON	1F58 _H	7A _H	00F8 _H	3D _H	00D8 _H	1E _H	00D8 _H
COMP1_CON	1F5C _H	7A _H	00FC _H	3D _H	00DC _H	1E _H	00DC _H
COMP2_CON	1F60 _H	7B _H	00E0 _H	3D _H	00E0 _H	1E _H	00E0 _H
COMP3_CON	1F64 _H	7B _H	00E4 _H	3D _H	00E4 _H	1E _H	00E4 _H
COMP4_CON	1F68 _H	7B _H	00E8 _H	3D _H	00E8 _H	1E _H	00E8 _H
COMP5_CON	1F6C _H	7B _H	00EC _H	3D _H	00EC _H	1E _H	00EC _H
COMP0_TIME*	1F5A _H	7A _H	00FA _H	3D _H	00DA _H	1E _H	00DA _H
COMP1_TIME*	1F5E _H	7A _H	00FE _H	3D _H	00DE _H	1E _H	00DE _H
COMP2_TIME*	1F62 _H	7B _H	00E2 _H	3D _H	00E2 _H	1E _H	00E2 _H
COMP3_TIME*	1F66 _H	7B _H	00E6 _H	3D _H	00E6 _H	1E _H	00E6 _H
COMP4_TIME*	1F6A _H	7B _H	00EA _H	3D _H	00EA _H	1E _H	00EA _H
COMP5_TIME*	1F6E _H	7B _H	00EE _H	3D _H	00EE _H	1E _H	00EE _H
EPA0_CON	1F40 _H	7A _H	00E0 _H	3D _H	00C0 _H	1E _H	00C0 _H
EPA1_CON*	1F44 _H	7A _H	00E4 _H	3D _H	00C4 _H	1E _H	00C4 _H
EPA2_CON	1F48 _H	7A _H	00E8 _H	3D _H	00C8 _H	1E _H	00C8 _H
EPA3_CON*	1F4C _H	7A _H	00EC _H	3D _H	00CC _H	1E _H	00CC _H
EPA4_CON	1F50 _H	7A _H	00F0 _H	3D _H	00D0 _H	1E _H	00D0 _H
EPA5_CON	1F54 _H	7A _H	00F4 _H	3D _H	00D4 _H	1E _H	00D4 _H
EPA0_TIME*	1F42 _H	7A _H	00E2 _H	3D _H	00C2 _H	1E _H	00C2 _H
EPA1_TIME*	1F46 _H	7A _H	00E6 _H	3D _H	00C6 _H	1E _H	00C6 _H
EPA2_TIME*	1F4A _H	7A _H	00EA _H	3D _H	00CA _H	1E _H	00CA _H
EPA3_TIME*	1F4E _H	7A _H	00EE _H	3D _H	00CE _H	1E _H	00CE _H

Окончание таблицы В.55

1	2	3	4	5	6	7	8
EPA4 TIME*	1F52 _H	7A _H	00F2 _H	3D _H	00D2 _H	1E _H	00D2 _H
EPA5 TIME*	1F56 _H	7A _H	00F6 _H	3D _H	00D6 _H	1E _H	00D6 _H
FREQ_CNT	1FBA _H	7D _H	00FA _H	3E _H	00FA _H	1F _H	00BA _H
FREQ_GEN	1FB8 _H	7D _H	00FB _H	3E _H	00F8 _H	1F _H	00B8 _H
P2_DIR	1FD2 _H	7E _H	00F2 _H	3F _H	00D2 _H	1F _H	00D2 _H
P7_DIR	1FD3 _H	7E _H	00F3 _H	3F _H	00D3 _H	1F _H	00D3 _H
P2_MODE	1FD0 _H	7E _H	00F0 _H	3F _H	00D0 _H	1F _H	00D0 _H
P7_MODE	1FD1 _H	7E _H	00F1 _H	3F _H	00D1 _H	1F _H	00D1 _H
P0_PIN	1FA8 _H	7D _H	00E8 _H	3E _H	00E8 _H	1F _H	00A8 _H
P2_PIN	1FD6 _H	7E _H	00F6 _H	3F _H	00D6 _H	1F _H	00D6 _H
P7_PIN	1FD7 _H	7E _H	00F7 _H	3F _H	00D7 _H	1F _H	00D7 _H
P2_REG	1FD4 _H	7E _H	00F4 _H	3F _H	00D4 _H	1F _H	00D4 _H
P7_REG	1FD5 _H	7E _H	00F5 _H	3F _H	00D5 _H	1F _H	00D5 _H
PI_MASK	1FBC _H	7D _H	00FC _H	3E _H	00FC _H	1F _H	00BC _H
PI PEND	1FBE _H	7D _H	00FE _H	3E _H	00FE _H	1F _H	00BE _H
PWM_COUNT	1FB6 _H	7D _H	00F6 _H	3E _H	00F6 _H	1F _H	00B6 _H
PWM_PERIOD	1FB4 _H	7D _H	00F4 _H	3E _H	00F4 _H	1F _H	00B4 _H
PWM0_CONTROL	1FB0 _H	7D _H	00F0 _H	3E _H	00F0 _H	1F _H	00B0 _H
PWM1_CONTROL	1FB2 _H	7D _H	00F2 _H	3E _H	00F2 _H	1F _H	00B2 _H
T1CONTROL	1F78 _H	7B _H	00F8 _H	3D _H	00F8 _H	1E _H	00F8 _H
T1RELOAD	1F72 _H	7B _H	00F2 _H	3D _H	00F2 _H	1E _H	00F2 _H
T2CONTROL	1F7C _H	7B _H	00FC _H	3D _H	00FC _H	1E _H	00FC _H
TIMER1*	1F7A _H	7B _H	00FA _H	3D _H	00FA _H	1E _H	00FA _H
TIMER2*	1F7E _H	7B _H	00FE _H	3D _H	00FE _H	1E _H	00FE _H
WG_COMP1	1FC2 _H	7E _H	00E2 _H	3F _H	00C2 _H	1F _H	00C2 _H
WG_COMP2	1FC4 _H	7E _H	00E4 _H	3F _H	00C4 _H	1F _H	00C4 _H
WG_COMP3	1FC6 _H	7E _H	00E6 _H	3F _H	00C6 _H	1F _H	00C6 _H
WG_CONTROL	1FCC _H	7E _H	00EC _H	3F _H	00CC _H	1F _H	00CC _H
WG_COUNTER	1FCA _H	7E _H	00EA _H	3F _H	00CA _H	1F _H	00CA _H
WG_OUTPUT	1FC0 _H	7E _H	00E0 _H	3F _H	00C0 _H	1F _H	00C0 _H
WG_PROTECT	1FCE _H	7E _H	00EE _H	3F _H	00CE _H	1F _H	00CE _H
WG_RELOAD	1FC8 _H	7E _H	00E8 _H	3F _H	00C8 _H	1F _H	00C8 _H

* Должны адресоваться как слово.

Регистр ZERO_REG

Адрес: 00_H

Состояние сброса: 0000_H

Двухбайтный регистр ZERO_REG всегда равен 0000_H. Он используется как источник константы с нулевыми значениями битов для проведения операций сравнения (компарации) и вычислений.

Т а б л и ц а В.56 – Разряды нулевого регистра ZERO_REG

Номер разряда	Функция
15 – 0	Эти биты всегда содержат нули и значение в регистре равно 0000 _H

