

1	Введение	1-1
1.1	<i>Состав 16-разрядного семейства микроконтроллеров</i>	1-2
1.2	<i>Описание основных возможностей</i>	1-4
2	Обзор архитектуры микроконтроллера	2-1
2.1	<i>Основная концепция архитектуры центрального процессора и оптимизации его работы</i>	2-2
2.2	<i>Системные средства микроконтроллера</i>	2-9
2.3	<i>Периферийные модули микроконтроллера</i>	2-15
2.4	<i>Защищенные биты</i>	2-16
3	Организация памяти	3-1
3.1	<i>Внутренняя ROM</i>	3-3
3.2	<i>Внутренняя RAM и область SFR-регистров</i>	3-5
3.3	<i>Внутренняя XRAM</i>	3-10
3.4	<i>Адресное пространство внешней памяти</i>	3-13
3.5	<i>Пересечение границ различных типов памяти</i>	3-14
4	Блок ЦПУ	4-1
4.1	<i>Конвейерная обработка команд</i>	4-3
4.2	<i>Битовое управление и битовая защита</i>	4-10
4.3	<i>Время выполнения команд (Instruction State Time)</i>	4-11
4.4	<i>SFR-регистры</i>	4-11
5	Система прерываний и ловушек	5-1
5.1	<i>Структура системы прерываний</i>	5-2
5.2	<i>Операции с PEC-каналами</i>	5-13
5.3	<i>Приоритетность прерываний и запросов на PEC-обслуживание</i>	5-17
5.4	<i>Сохранение состояния во время обслуживания прерывания</i>	5-18
5.5	<i>Время ответа на прерывание</i>	5-19
5.6	<i>Внешние прерывания</i>	5-23
5.7	<i>Функции ловушек</i>	5-25
6	Параллельные порты	6-1
6.1	<i>Порт 0</i>	6-7
6.2	<i>Порт 1</i>	6-10
6.3	<i>Порт 2</i>	6-13
6.4	<i>Порт 3</i>	6-19
6.5	<i>Порт 4</i>	6-23
6.6	<i>Порт 5</i>	6-27
6.7	<i>Порт 6</i>	6-29
6.8	<i>Порт 7</i>	6-33
6.9	<i>Порт 8</i>	6-39

7	Специальные выводы микроконтроллера	7-1
8	Интерфейс внешней шины	8-1
8.1	Режимы внешней шины	8-2
8.2	Программируемые характеристики шины	8-11
8.3	Цикл работы шины, управляемый сигналом \overline{READY}	8-17
8.4	Управление контроллером внешней шины	8-19
8.5	Режим холостого хода EBC	8-26
8.6	Арбитраж внешней шины	8-27
8.7	Интерфейс XBUS	8-31
9	Блоки таймеров основного назначения	9-1
9.1	Блок GPT1	9-1
9.2	Блок таймеров GPT2	9-17
10	Асинхронный/Синхронный последовательный интерфейс	10-1
10.1	Работа в асинхронном режиме	10-4
10.2	Работа в синхронном режиме	10-8
10.3	Возможности аппаратного обнаружения ошибок	10-10
10.4	Создание baud rate ASC0	10-10
10.5	Управление прерываниями ASC0	10-12
11	Высокоскоростной синхронный последовательный интерфейс	11-1
11.1	Полнодуплексные режимы работы	11-7
11.2	Операции в полудуплексном режиме	11-9
11.3	Создание тактового сигнала	11-11
11.4	Механизм определения ошибок	11-13
11.5	Управление прерываниями SSC	11-14
12	Сторожевой таймер	12-1
13	Аппаратный загрузчик	13-1
14	Блок сравнения и захвата (CAPCOM)	14-1
14.1	CAPCOM таймеры	14-4
14.2	Прерывания таймеров CAPCOM-блока	14-8
14.3	Регистры захвата и сравнения	14-8
14.4	Режим захвата	14-12
14.5	Режимы сравнения	14-13
14.6	Прерывания при захвате и сравнении	14-19

15 Модуль широтно-импульсной модуляции	15-1
15.1 Режимы работы	15-2
15.2 Регистры модуля ШИМ	15-7
15.3 Создание запросов на прерывание	15-11
15.4 Выходные ШИМ-сигналы	15-11
16 Модуль АЦП	16-1
16.1 Работа и выбор режима	16-2
16.2 Управление временем преобразования	16-10
16.3 Управление прерываниями АЦП	16-11
17 Системный RESET	17-1
18 Режимы пониженного энергопотребления	18-1
18.1 Режим покоя	18-1
18.2 Режим отключения питания	18-2
18.3 Значения выводов микроконтроллера во время режима покоя и режима отключения питания	18-3
19 Системное программирование	19-1
19.1 Операции со стеком	19-3

1 Введение

Быстро растущая область применения встроенных микроконтроллерных систем управления представляет собой одну из наиболее критичных к времени исполнения рабочих сред для современных микроконтроллеров. Сложные алгоритмы управления основаны на большом количестве обрабатываемых входных как цифровых, так и аналоговых сигналов, и необходимые выходные сигналы должны создаваться с ограниченным максимальным временем ответа. Применение встроенных устройств управления зачастую предъявляет повышенные требования к размерам устройства, величине потребляемой электроэнергии и максимальной стоимости устройства.

Поэтому для применения встроенных систем управления необходимо использовать микроконтроллеры, которые:

- Предлагают высокий уровень интеграции системы
- Исключают необходимость в дополнительных периферийных устройствах
- Предлагают систему защиты и механизм предотвращения ошибок

С увеличением сложности встроенных систем управления, увеличиваются требования к производительности центрального процессора и функциональности периферии. Для увеличения производительности фирмой Infineon было разработано семейство 16-разрядных CMOS микроконтроллеров. При разработке новой архитектуры не стояла задача обеспечения обратной совместимости с 8-разрядными микроконтроллерами.

Архитектура семейства 16-разрядных микроконтроллеров была создана с использованием успешных аппаратных и программных решений, которые хорошо зарекомендовали себя в популярных 8-разрядных семействах микроконтроллеров фирмы Infineon.

Данное руководство описывает некоторые из числа 16-разрядных микроконтроллеров фирмы Infineon обширного семейства C166, так же называемого семейством C167.

В данном руководстве приведено описание следующих процессоров семейства C167:

- | | |
|---------------|--|
| • C167CR-RM | PLL, 2Кбайт XRAM, CAN модулем |
| • C167CR-4RM | PLL, 2Кбайт XRAM, 32 Кбайт ROM, CAN модулем |
| • C167CR-16RM | PLL, 2Кбайт XRAM, 128 Кбайт ROM, CAN модулем |
| • C167CR-16FM | PLL, 2Кбайт XRAM, 128 Кбайт Flash-память, CAN-модуль |

- C167SR-LM PLL, 2Кбайт XRAM
- C167S-4RM PLL, 32Кбайт XRAM
- C167-LM Базовая версия

1.1 Состав 16-разрядного семейства микроконтроллеров

16-разрядное семейство микроконтроллеров фирмы Infineon было разработано для удовлетворения высоких требований при создании встроенных систем управления реального времени. Архитектура семейства была оптимизирована таким образом, чтобы обеспечить высокое быстродействие операций и минимальное время ответа на внешние воздействия (прерывания). Для уменьшения необходимости во вмешательстве центрального процессора, была интегрирована интеллектуальная периферийная подсистема, что также уменьшило необходимость в частом использовании внешних интерфейсов для передачи данных. Высокая гибкость архитектуры позволяет применять микроконтроллер в различных приложениях, в таких областях как автомобильная промышленность, в промышленных устройствах для управления или передачи данных.

В основу ядра 16-разрядного семейства микроконтроллеров был положен модульный принцип. Все микроконтроллеры семейства выполняют эффективный оптимизированный набор команд (для контроллеров второго поколения добавлены дополнительные команды). Это позволяет легко и просто расширять состав семейства микроконтроллера за счет изменения размера и типа встроенной памяти, за счет изменения набора периферийных модулей микроконтроллера и числа ножек микроконтроллера.

Концепция XBUS открывает прямой путь для интеграции дополнительных периферийных модулей, для удовлетворения специальных запросов при построении систем управления.

Поскольку программы встроенных устройств управления со временем становятся большего объема, то находят все большее применение языки высокого уровня, поскольку программы, написанные на языках высокого уровня (HLL), просты для понимания, отладки и дальнейшего совершенствования.

Микроконтроллеры типа 80C166 являются устройствами первого поколения семейства 16-разрядных микроконтроллеров фирмы Infineon. В этих устройствах была реализована 16-разрядная архитектура.

Ко второму поколению процессоров можно отнести устройства типа C165 и C167. Второе поколение является более эффективным, что обусловлено наличием дополнительных команд для поддержки HLL,

увеличенным адресным пространством, увеличенной встроенной RAM и улучшенным управлением различными ресурсами, расположенных на внешней шине.

Большое количество вариантов микроконтроллеров второго поколения обеспечивает дополнительные возможности, такие как дополнительная встроенная высокоскоростная RAM, встроенный CAN-модуль, PLL-модуль и др.

При разработке эффективных систем для увеличения производительности часто встает задача в интеграции специальной периферии. Это достигается путем поддержки технологии XBUS, разработанной INFINEON для 16-разрядных микроконтроллеров второго поколения. XBUS является внутренним отображением интерфейса внешней шины, что открывает путь к упрощению интеграции периферийных модулей путем стандартизации интерфейсов. В некоторые члены семейства включен интегрированный CAN модуль, использующий преимущества этой технологии.

Семейство C165 является упрощенной версией C167. C165 имеет уменьшенный размер корпуса, пониженное потребление энергии за счет исключения модулей CAPCOM, АЦП и ШИМ-контроллера.

Большое количество различных моделей обеспечивается путем использования различных типов памяти: ROM, Flash и неэнергонезависимой памяти. Использование специальных функциональных модулей в различных моделях вносит большое разнообразие в модельный ряд.

Микроконтроллеры могут быть предложены в различных корпусах, для различных температурных режимов и частотных показателей.

В настоящее время планируется и разрабатывается большое количество стандартных и специальных моделей.

1.2 Описание основных возможностей

C167 является улучшенным представителем 16-разрядного семейства однокристальных микроконтроллеров фирмы Infineon. В нем сочетаются высокая производительность центрального процессора (до 12,5 миллионов операций в секунду) с высокой функциональностью периферийных модулей.

Ниже приведены несколько основных особенностей, содействующих высокой производительности C167:

Высокая производительность конвейера команд с четырьмя ступенями выполнения команд

- Минимальное время исполнения команд - 80нс, причем большинство команд исполняются за 1 такт
- Время умножения двух 16-разрядных чисел - 400нс, время деления двух чисел (32-бит/16-бит) - 1мкс
- Высокопроизводительные внутренние шины данных
- Регистровая организация, основанная на переключаемых множественных банках регистров
- Поддержка одноктактного контекстного переключения
- 16 Мбайт линейного адресного пространства для программного кода и данных (архитектура Фон-Неймана)
- Поддержка системного стека с автоматическим определением верхней и нижней границ стека

Набор высокоэффективных инструкций, оптимизированных для задач управления

- Размерность данных – бит, байт и слово
- Наличие гибких и эффективных режимов адресации памяти, существенно уменьшающих объем программного кода
- Наличие расширенных логических битовых операций с прямой адресацией к области памяти 6 Кбит для управления периферийными модулями и установленными пользователем флагов
- Аппаратные ловушки для определения запрещенных состояний во время исполнения программы
- Поддержка HLL для операций условных переходов и эффективного доступа к данным

Встроенная память

- 2-х Кбайтная внутренняя RAM для переменных, банков регистров, системного стека и программного кода
- 2-х Кбайтная высокоскоростная XRAM для переменных, пользовательского стека и кода (присутствует не во всех моделях)
- Встроенная ROM или Flash - память (присутствует не во всех моделях)

Интерфейс внешней шины

- Мультиплексная или демultipлексная конфигурация шины
- Возможность сегментации памяти и генерации сигналов chip select
- 8-разрядная и 16-разрядная шина данных
- Выбор параметров цикла работы шины для пяти программируемых адресных областей

Система прерываний с 16 уровнями приоритетов

- 56 возможных источников прерываний с отдельными векторами прерываний
- 240/400нсек типичное/максимальное время отклика на прерывание в случае работы с внутренней памятью программы
- быстрые внешние прерывания

8-канальный контроллер периферийных событий (PES)

- Передача данных за один такт при обслуживании прерывания PES-контроллером
- Подсчет количества обслуженных запросов на PES-обслуживание (после запрограммированного числа PES-передач используется стандартное обслуживание прерывания центральным процессором)
- Исключение дополнительных операций для сохранения и восстановления состояния системы при запросе прерывания

Интеллектуальная периферийная подсистема микроконтроллера

- 16-канальный 10-разрядный контроллер АЦП с программируемым временем преобразования (минимальное время - 9.7мкс), режим автоматического сканирования, режим вставки преобразования
- Два 16-канальных блока захвата/сравнения с двумя независимыми временными базами для каждого, гибкий ШИМ модуль/фиксатор событий с 5 различными режимами работы, включающие 4 16-разрядных таймера/счетчика с максимальным разрешением 320нс
- 4-канальный ШИМ-модуль
- 2 многофункциональных модуля таймеров общего назначения

GPT1: три 16-ти разрядных таймера/счетчика с разрешением 320нс

GPT2: два 16-ти разрядных таймера/счетчика с максимальным разрешением 160нс

- Асинхронно-синхронный последовательный канал (USART) с тактовым генератором, с определением ошибок четности, перезаписи и формата передачи
- Высокоскоростной синхронный последовательный канал с программируемой длиной передаваемых данных и программируемым направлением сдвига
- Сторожевой таймер с программируемыми временными интервалами
- Аппаратный (bootstrap) загрузчик для гибкой инициализации системы
- Встроенный в микроконтроллер CAN-модуль (не во всех моделях)

111 каналов ввода-вывода с индивидуальной побитной адресацией

- Состояние высокого сопротивления в режиме ввода
- Двухтактный выходной каскад или с открытым стоком
- Возможность выбора порога входного напряжения (не для всех моделей)

Различные температурные режимы

- От 0° до +70°C, от -40° до +85°C, от -40° до +125°C

Технология CMOS фирмы INFINEON

- CMOS технология с малым потреблением энергии, включающая режим остановки процессора и режим энергосохранения

144-выводной пластиковый квадратный корпус типа PMQFP

- EIAJ стандарт, шаг выводов 0.65 мм, технология поверхностного монтажа

Поддержка разработчиков

Поставщики средств разработки представляют большое количество программных и аппаратных средств разработки для 16-разрядного семейства микроконтроллеров фирмы INFINEON. Высокое качество и надежность этих средств проверена во многих проектах за многие годы. Средства разработки для 16-разрядных микроконтроллеров фирмы INFINEON включают следующие компоненты:

- Компиляторы (C, Modula, Фортран)
- Макроассемблеры, линковщики, компоновщики, менеджеры библиотек, преобразователи форматов
- Симуляторы архитектуры
- Высокоуровневые отладчики
- Операционные системы реального времени
- Внутрисхемные эмуляторы (основанные на эмуляционных или стандартных кристаллах)
- Подключаемые эмуляторы
- Переходные колодки для эмуляторов, колодки для микросхем
- Логические анализаторы
- Оценочные платы с программными мониторами
- Промышленные платы (также для CAN, FUZZY, PROFIBUS, FORTH приложений)
- Программы для управления сетями (CAN, PROFIBUS)

Список терминов

ASC	Асинхронный/синхронный последовательный интерфейс
CAPCOM	Блок захвата и сравнения
EBC	Контроллер внешней шины
ESFR	Расширенные регистры специальных функций
GPR	Регистры основного назначения
PEC	Контроллер внешних событий
SFR	Регистры специальных функций
SSC	Синхронный последовательный интерфейс
XBUS	Внутреннее отображение внешней шины
XBUS	Внутреннее расширение внешней памяти

2 Обзор архитектуры микроконтроллера

Архитектура C167 объединяет в себе преимущества как RISC, так и CISC процессоров, при этом удалось достичь хорошо сбалансированного результата. C167 не только имеет мощное процессорное ядро и набор периферийных модулей, но также имеет высокоэффективную систему взаимодействия между ними. Одна из четырех шин, используемая параллельно в C167, – XBUS, которая представляет собой внутреннее отображение интерфейса внешней шины. Эта шина обеспечивает стандартный способ интеграции в микроконтроллер специальных блоков для различных систем. Таким образом возникает возможность для создания большого числа моделей стандарта C167 без внесения изменений в базовую архитектуру микроконтроллера.

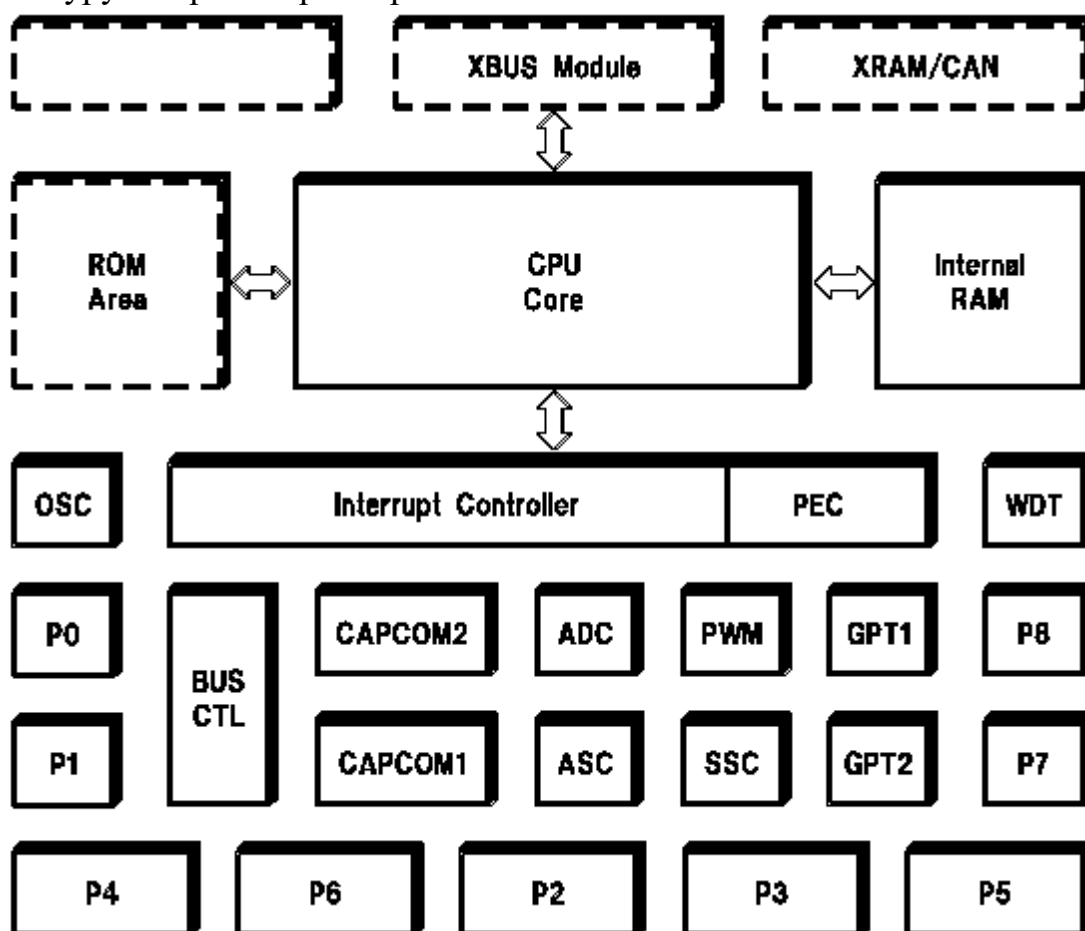


Рисунок 2-1

C167 Функциональная блок-схема

2.1 Основная концепция архитектуры центрального процессора и оптимизации его работы

Ядро процессора содержит 4-ступенчатый конвейер для выполнения команд, 16-разрядное арифметико-логическое устройство (АЛУ) и регистры специального назначения (SFR-регистры). Дополнительно в центральный процессор включены модуль умножения и деления, генератор битовых масок и генератор сдвига.

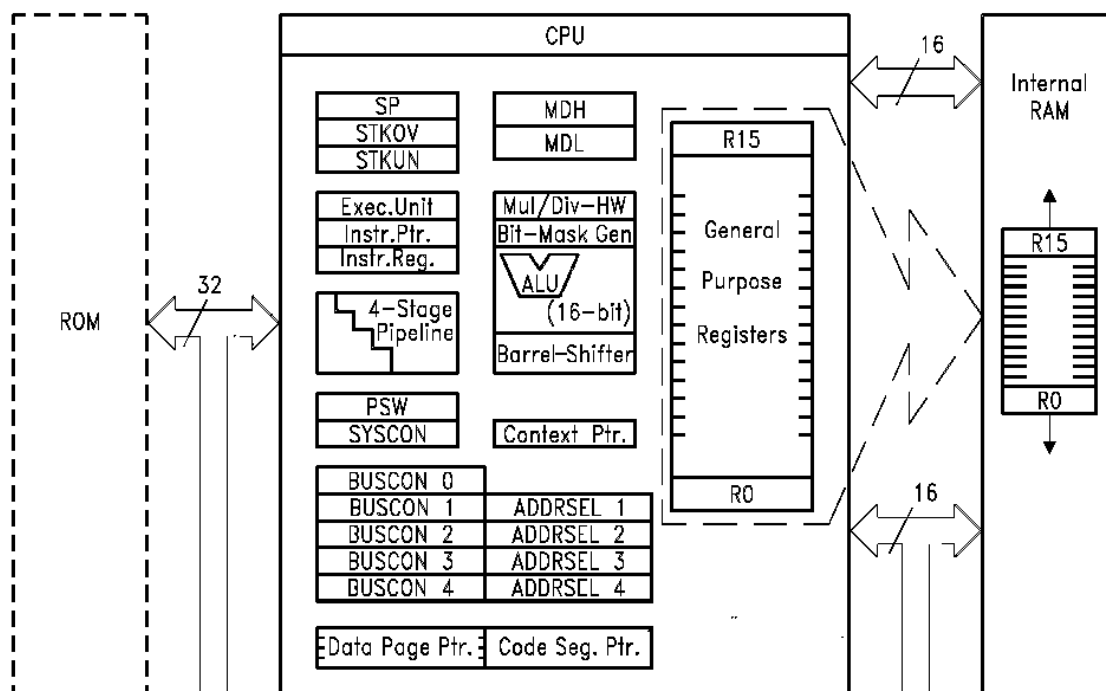


Рисунок 2-2

Блок-схема центрального процессора

Для объединения высокой производительности и высокой гибкости, было оптимизировано количество блоков в ядре. Функциональные блоки ядра ЦПУ управляются логикой, декодирующей инструкции. Ниже произведен обзор следующих разделов:

- 1) Широкий набор инструкций / быстрое выполнение команд
- 2) Высоко-функциональное 8-, 16-разрядное АЛУ
- 3) Расширенная обработка битов и управление периферией
- 4) Разнообразные возможности для выполнения условных переходов, вызовов подпрограмм и циклических обработок
- 5) Содержательный и оптимальный формат команд
- 6) Система прерываний с программируемым приоритетом

Широкий набор инструкций / быстрое выполнение команд

Большая часть команд C167 может выполняться за 1 машинный цикл, занимающий 100нс при частоте процессора 20мГц. Для примера, команды сдвига и команды циклического сдвига всегда производятся за один машинный цикл, независимо от числа сдвигаемых битов.

Команды условных переходов, умножения и деления занимают более одного машинного цикла. Однако, эти команды можно оптимизировать. К примеру, команды условных переходов нуждаются в дополнительном машинном цикле, когда переход совершен, однако последующие переходы при циклической обработке не нуждаются в дополнительном машинном цикле благодаря, так называемому “Jump Cash”.

Деление 32-х разрядного числа на 16-ти разрядное число занимает 1мкс, а умножение двух 16-ти разрядных чисел занимает 0.5мкс при частоте процессора 20 мГц.

Время исполнения команд уменьшается при использовании конвейерной обработки команд. Это позволяет обрабатывать ядру процессора несколько последовательных стадий команд параллельно. Следующие четыре стадии выполнения команд конвейера обеспечивают оптимальный баланс для ядра процессора:

ВЫБОРКА: в этой стадии команды вызываются из внутренней RAM или ROM или из внешней памяти, при этом адрес определяется текущим значением IP.

ДЕКОДИРОВАНИЕ: в этой стадии вызванные перед этим команды декодируются и также вызываются необходимые операнды.

ИСПОЛНЕНИЕ: в этой стадии производятся операции над вызванными ранее операндами.

ОБРАТНАЯ ЗАПИСЬ: в этой стадии результаты операции записываются по заранее известному адресу.

Декодер команд

Декодирование команд первично производится на выходах PLA и основано на исходном коде команды. На каждой стадии выполнения команды микрокод в явном виде не используется, для каждой стадии конвейера контрольные сигналы поступают из управляющих регистров, значения которых определяются на стадии декодирования команды. На скорость работы конвейера в первую очередь влияют циклы ожидания при доступе к внешней памяти, при этом сигналы от управляющих регистров остаются без изменения. Многотактные команды выполняются путем вставки команд и при помощи внутренних машинных циклов, которые изменяют необходимые сигналы управления.

Высокофункциональное 8-ми - 16-ти разрядное АЛУ

Все стандартные арифметические и логические операции производятся в 16-ти разрядном АЛУ. Для операций с байтами, 6-ой и 7-ой бит результата операции влияет на корректную установку флагов состояний. Большая точность вычислений обеспечивается путем установки флага “CARRY-IN” в АЛУ, предыдущей вычисленной частью операции. Внутренние блоки АЛУ, выполняющие операции, оптимизированы для работы с 8-ми разрядными и 16-ти разрядными числами. После заполнения конвейера, одна инструкция будет выполняться в течении одного машинного цикла, за исключением инструкций умножения и деления. Передовой алгоритм Booth обеспечивает перемножение четырех битов и деление двух битов за один машинный такт. В этих операциях используются два спаренных 16-ти разрядных регистра – MDL и MDH, и таким образом эти операции нуждаются в 4 тактах для умножения двух 16-ти разрядных чисел и 9 тактах для деления 32-х разрядного числа на 16-ти разрядное число, плюс дополнительно каждой операции необходим один такт для настройки и регулировки операндов и результатов.

Длинные команды умножения и деления могут быть прерваны во время выполнения, что позволяет добиваться более быстрого ответа на прерывания. Эти команды также позволяют преобразовывать байты данных, при знаковом расширении для слова данных. Структура внутренней шины также позволяет передавать байты или слова из периферии или на периферию с учетом требований переферийного оборудования.

Набор флагов автоматически обновляется в PSW после каждой арифметической, логической операции, операции с битами и операции перемещения данных. Наличие флагов позволяет совершать операции условного перехода по специальному условию. Поддержка как арифметики со знаком, так и беззнаковой арифметики обеспечивается с помощью определяемых пользователем условий перехода. Эти флаги также автоматически сохраняются при входе процессора в прерывание или обслуживание ловушки. Все адреса переходов также рассчитываются в АЛУ.

16-ти разрядный генератор сдвига обеспечивает необходимое количество сдвигов за один такт. Также поддерживаются операции сдвига и циклического сдвига.

Расширенная обработка битов и управление периферией

Для обработки битов предназначено большое число команд. Использование этих команд обеспечивает эффективное управление и тестирование периферийного оборудования. В отличие от аналогичных команд других микроконтроллеров, эти команды обеспечивают постоянный доступ к двум операндам в побитноадресуемом пространстве памяти, без необходимости перемещения данных в промежуточные регистры.

Некоторые логические команды доступны как для совершения преобразований слов и байтов, так и поддерживают возможность преобразования битов. Это позволяет пользователям сравнивать и модифицировать биты регистров управления периферийными модулями с помощью одной команды. В систему команд для исключения длинных потоков команд для побитного изменения значений добавлены команды одновременного изменения значения нескольких битов. Эти операции выполняются за один машинный такт.

Широкие возможности для выполнения условных переходов, вызовов подпрограмм и циклических обработок

При выполнении команд перехода после совершения перехода необходим один дополнительный машинный такт, что обусловлено большим числом операций переходов в микроконтроллерных программах. Оптимизация осуществлена путем предварительного расчета адреса во время декодирования команды. Для уменьшения загрузки при обработке циклов, были предложены три решения:

- Первое решение обеспечивает выполнение перехода за один цикл после первой проверки условия цикла. Таким образом теряется только один машинный цикл при цикловой обработке. При выходе из цикла не требуется дополнительных машинных циклов. Для цикловых обработок не требуется специальных инструкций и циклы определяются автоматически во время исполнения команд условных переходов.
- Второе решение для команд переходов позволяет определять конец таблицы и избегать использования двух команд сравнения, встроенных в цикл. Для этого в конце таблицы помещается наименьшее отрицательное значение и используется условие перехода, если ни это значение ни сравниваемая величина не были найдены. Цикл прерывается если одно из двух условий было выполнено. Состояние выхода затем может быть протестировано.
- Обеспечивается более гибкое решение, чем имеющиеся в других микроконтроллерах команда “decrement and skip on zero”. С помощью использования команд сравнения и инкремента или декремента, пользователь может сравнивать любые значения. Это позволяет

счетчику таблицы охватывать любые пределы. Это является особым преимуществом системы команд C167 при табличном поиске.

Сохранение текущего состояния системы автоматически совершается во внутреннем системном стеке, без дополнительного использования команд сохранения состояния до входа и после выхода из прерываний и обслуживания ловушек. Команды работы с подпрограммами записывают значение IP в системный стек при входе в подпрограмму, при этом на выполнение этих команд требуется больше времени чем для команды безусловного перехода, так как необходимо сохранить состояние системы.

В этих командах также обеспечена поддержка косвенных переходов и вызовов подпрограмм. За счет этого обеспечен множественный режим переходов CASE в ассемблерных макросах и языках высокого уровня.

Сжатый и оптимизированный форматы команд

Для оптимизации производительности конвейера набор команд был разработан таким образом, чтобы включать в себя концепцию RISC. Эта концепция в первую очередь позволяет быстро декодировать команды и операнды при уменьшении загрузки конвейера. Эта концепция однако не исключает возможность использования сложных команд, которые необходимы для разработчиков программ. При разработке набора команд преследовались следующие цели:

- 1) Обеспечение емких команд, которые необходимы для выполнения часто используемых команд и потоков повторяющихся команд. Избежание передачи данных во временные регистры и из временных регистров, типа аккумуляторов. Совершение параллельных задач, таких как сохранение режима работы процессора до начала входа в прерывание.
- 2) Избежание сложных схем декодирования путем расположения операндов в согласованных областях для каждой команды, также исключение редко используемых режимов сложной адресации..
- 3) Обеспечение более частого использования команд длиной в слово. Все команды, использующие другой формат, занимают два слова. Это позволяет расположить все команды в границах слов, что упрощает группировку команд. Это также предоставляет возможность использования большего набора команд относительных переходов.

Высокая производительность, реализованная схемотехническим построением процессора, может эффективно использоваться программистами с помощью высокофункционального набора команд C167, который включает следующие классы:

- Арифметические команды
- Логические команды
- Команды обработки битов
- Команды сравнения и команды контролирующие метки
- Команды сдвига и циклического сдвига
- Команды приоритетности
- Команды перемещения данных
- Команды системного стека
- Команды перехода и управления подпрограммами
- Команды возврата
- Команды управления системой
- Различные команды

Возможными операндами могут быть биты, байты и слова. Специальные команды поддерживают преобразование (расширение) байтов в слова. Для операндов предназначены различные варианты режимов прямой, косвенной и непосредственной адресации.

Система прерываний с программируемым приоритетом

Для возможности обработки большого количества прерываний были включены следующие блоки:

- 1) Контроллер периферийных событий (PES) используется для разгрузки центрального процессора от ответов на запросы на прерывания. Это исключает дополнительные операции при входе и выходе из прерывания или ловушки, путем однократного перемещения байта или слова, вызванного запросом на прерывание, между двумя адресами в нулевом сегменте памяти с возможностью увеличения указателя либо адреса источника либо адреса назначения. Только один такт центрального процессора тратится для обслуживания PES.
- 2) Многоприоритетный контроллер прерываний позволяет расположить все прерывания по приоритетам. Имеется возможность для группировки прерываний, что позволяет предотвращать прерывание друг другом одинаковых по приоритету задач. Для каждого из возможных источников прерываний существует отдельный регистр управления, который включает в себя флаг запроса на прерывание, флаг разрешения прерывания и поле приоритетов прерываний. В случае начала обслуживания прерывания центральным процессором, его можно прервать только посредством запроса более высокого приоритета.

Для стандартной обработки прерываний, каждый из возможных источников прерываний имеет вектор прерывания.

- 3) Переключаемые банки регистров: Эта особенность позволяет, пользователям определять до 16-ти регистров общего назначения, расположенных в любом месте внутренней RAM. Специальная одноктактная команда позволяет переключать банки регистров с одной задачи на другую.
- 4) Прерываемые многотактные команды: Укороченное время начала прерывания доступно вследствие возможности прерывания многотактных команд (умножения и деления).

Время ответа на прерывание находится в пределах 250-500нс (в случае выполнения внутренней программы).

Входы внешних прерываний проверяются каждые 50нс, что позволяет распознавать очень короткие внешние сигналы.

C167 также обеспечивает великолепный механизм распознавания и обработки некорректных или ошибочных состояний, которые возникают в течении работы (так называемые «аппаратные ловушки»). Аппаратные ловушки вызывают немедленную немаскируемую реакцию системы, которая похожа на стандартное обслуживание прерываний (переход к определенной точке таблицы прерываний). Возможность использования аппаратной ловушки дополнительно указывается в специальном бите регистра флагов ловушек (TFR). Аппаратные ловушки прерывают исполнение любой программы, за исключением других ловушек с более высоким приоритетом. В свою очередь обслуживание аппаратных ловушек не может быть обычным путем прервано стандартными прерываниями или PEC-прерываниями.

Программные прерывания поддерживаются с помощью команды TRAP, в которой указывается номер прерывания или ловушки.

2.2 Системные средства микроконтроллера

Микроконтроллер C167 обеспечен большим количеством мощных системных средств, расположенных вокруг ЦПУ. Соединение центрального процессора с этими средствами позволяет получать в результате высокую производительность всего семейства C167.

Контроллер периферийных событий (РЕС) и управление прерываниями

Контроллер периферийных событий позволяет реагировать на запросы на прерывание с помощью однократной передачи данных (байта или слова). При этом используется один командный такт и не требуется сохранять и возобновлять состояние системы. Сравнение приоритетов каждого источника прерываний производится после каждого машинного такта в блоке управления прерываниями. Если РЕС-обслуживание имеет высший приоритет, то начинается РЕС-передача. Если необходимо обслужить прерывание с помощью центрального процессора, то проверяется текущий уровень приоритета центрального процессора, записанный в регистре PSW, и если уровень приоритета прерывания выше, то оно будет обслужено. После разрешения прерывания, текущее состояние системы сохраняется в системном стеке и центральный процессор переходит по специальному системному вектору прерывания.

РЕС содержит набор SFR-регистров, которые содержат значения и контрольные биты для 8 каналов передачи данных. Необходимо отметить, что РЕС использует предназначенную ему область RAM, которая содержит адреса источников данных и адреса назначения. РЕС управляется таким же образом, как и другие периферийные устройства, через SFR-регистры, содержащие полную информацию о каждом канале.

После каждого РЕС-обслуживания автоматически происходит декрементирование значения счетчик РЕС-передач данных, для исключения неограниченного по времени режима передачи данных. Когда значение счетчика достигнет нуля, дальнейшие запросы на РЕС-обслуживание будут обслуживаться по соответствующему вектору как стандартные прерывания. РЕС-обслуживание подходит, например, для перемещения содержимого регистров в таблицу памяти или для перемещения из таблицы памяти. C167 содержит 8 РЕС-каналов, каждый из которых имеет возможность осуществлять быстрые перемещения данных по запросу на прерывание.

Область памяти

Пространство памяти микроконтроллера C167 устроено по принципу архитектуры Фон-Неймана. Это означает, что память программного кода, память данных, регистров и портов ввода-вывода организована в одном и том же линейном адресном пространстве, которое может достигать 16 Мбайт. Полное пространство памяти может быть доступно для данных размером байт либо слово. Отдельная часть внутренней памяти может также иметь прямую битовую адресацию.

Внутренняя 16-битная 2-х Кбайтная RAM обеспечивает быстрый доступ к регистрам общего назначения (GPR-регистры), пользовательским данным (переменным) и системному стеку. Внутренняя RAM также может использоваться для программного кода. Особенная схема декодирования обеспечивает гибкие пользовательские банки регистров во внутренней памяти, в то время как остальная часть памяти оптимизируется для пользовательских данных.

Центральный процессор использует банки регистров, состоящие из 16 GPR-регистров длиной в байт или слово, которые физически расположены во внутренней RAM. Регистр Context Pointer (CP) определяет базовый адрес активного банка регистров, доступного для центрального процессора в настоящее время. Количество банков регистров ограничено доступным объемом RAM. Для простой передачи параметров банки регистров могут перекрываться друг с другом.

Системный стек, длиной до 1024 слов, обеспечивает временное хранение данных. Системный стек также расположен в области RAM микроконтроллера, и доступ центрального процессора к нему определяется через регистр указателя стека (SP). Содержимое двух независимых регистров общего назначения STKOV и STKUN автоматически сравнивается со значением регистра указателя стека во время каждого доступа к стеку, для определения верхней и нижней границ стека.

Аппаратное определение объема выбранной памяти, расположенной во внутренней памяти, позволяет пользователям производить доступ с помощью прямой или косвенной адресации и получать необходимые данные без использования временных регистров или специальных команд.

Внутренняя 16-битная 2-х Кбайтная XRAM обеспечивает быстрый доступ к пользовательским данным (переменным), пользовательским стекам и программному коду. Микроконтроллерная X-RAM организована как X-периферия, и программный доступ к ней организуется как к внешней памяти. Поэтому в XRAM нельзя размещать банки регистров, и она не является побитно адресуемой. XRAM обеспечивает 16-разрядный доступ с максимально возможной скоростью.

Необязательная внутренняя ROM обеспечивает сохранение как данных так и программного кода. Эта область памяти соединяется с процессором через 32-разрядную шину. Таким образом ввод команды, состоящей из двух слов, может быть произведен за один машинный такт. Выполнение программы из внутренней ROM является самым быстрым способом из всех возможных альтернатив.

Для регистров специальных функций зарезервировано 1024 байта адресного пространства. Стандартная область для регистров специальных функций (SFR) занимает 512 байт, в то время как область расширенных регистров специальных функций (ESFR) занимает другие 512 байт. (E)SFR-регистры, используемые для функций управления и контроля различных модулей микроконтроллера, имеют размер в одно слово. Неиспользуемые (E)SFR адреса зарезервированы для будущих моделей семейства C167 с расширенными функциональными возможностями.

Интерфейс внешней шины

Для создания систем, в которых требуется больший объем памяти, чем расположено внутри микроконтроллера, имеется возможность через интерфейс внешней шины подключать до 16 Мбайт RAM и/или ROM. Встроенный контроллер внешней шины (EBC) позволяет получить доступ к внешней памяти и/или к внешним периферийным устройствам с возможностью широкой настройки. Может быть независимо друг от друга выбрано до пяти различных адресных областей с различными режимами работы шины: мультиплексная или демultipлексная, 8- или 16-разрядная шина данных, также имеется возможность для изменения длительности цикла шины (циклы ожидания, длительность сигналов). Это позволяет настроить доступ для различных типов памяти и периферии. Если устройство не работает в режиме single chip, в котором не требуется внешняя память, то EBC может контролировать доступ к внешним устройствам с помощью одного из ниже приведенных режимов внешнего доступа:

- 16-/18-/20-/24-разрядный адрес, 16-разрядные данные, демultipлексная шина
- 16-/18-/20-/24-разрядный адрес, 8-разрядные данные, демultipлексная шина
- 16-/18-/20-/24-разрядный адрес, 16-разрядные данные, мультиплексная шина
- 16-/18-/20-/24-разрядный адрес, 8-разрядные данные, мультиплексная шина

В режим с демultipлексной шиной PORT1 используется для вывода адресов, и PORT0 используется для ввода-вывода данных. Режим с мультиплексной шиной использует PORT0 для обоих адресов и для ввода-вывода данных. Все режимы используют PORT4 для вывода старших

разрядов адреса (A16...), в том случае если применяется шина адреса с более чем 16-ю разрядами.

Для интерфейса внешней шины важными представляются временные характеристики (циклы ожидания, ширина ALE и задержка чтения и записи). Эти характеристики спроектированы легко программируемыми, что позволяет пользователям адаптировать различные типы памяти и/или периферии в широком диапазоне. Доступ к очень медленной памяти или периферии поддерживается с помощью специальной функции “Ready”.

Для приложений, которые нуждаются менее чем в 64Кбайтах адресного пространства, может быть выбрана модель не сегментированной памяти, где все данные могут быть определены с помощью 16-ти разрядной адресации, и при этом PORT4 не используется в качестве выходного канала старших битов адреса (A23...A16). В случае использования сегментированной памяти могут потребоваться старшие биты адреса.

Встроенная в микроконтроллер шина XBUS является внутренним отображением внешней шины и позволяет организовывать доступ к интегрированным, специально предназначенных для приложений, модулям и встроенной периферии как к внешним компонентам. При этом интерфейс для связи периферии с центральным процессором строго определен.

Включенные в микроконтроллер XRAM и CAN-модуль являются представителями X-периферии.

Генератор тактовых сигналов

Тактовый генератор обеспечивает тактовые импульсы для управления оборудованием C167. Генератор может работать либо с внешним кварцем и соответствующей схемой включения (см. рекомендации в разделе «Специальные выводы»), либо может быть запущен посредством сигналов внешнего тактового генератора. Генератор либо выдает (через буфер) тактовые сигналы для оборудования микроконтроллера, производя при этом деление на два частоты тактового генератора, либо сигнал поступает на встроенный в микроконтроллер умножитель (phase locked loop - PLL), который умножает входную частоту на коэффициент F (зависящий от режима работы и/или от типа микроконтроллера). Результирующий внутренний тактовый сигнал обозначается как «CPU clock». Для центрального процессора и внутренней периферии микроконтроллера тактовым генератором генерируются два независимых тактовых сигнала. Во время режима ожидания сигнал «CPU clock» отключен, однако тактовый сигнал для периферийного оборудования продолжает вырабатываться. Оба сигнала перестают вырабатываться, когда процессор переходит в “спящий” режим.

Встроенная в микроконтроллер PLL-схема позволяет работать C167 с низкочастотным тактовым генератором, обеспечивая при этом максимальную производительность. PLL умножает частоту внешнего тактового генератора на управляемый коэффициент F и генерирует сигнал CPU clock со скажностью 50%. PLL также обеспечивает механизм устранения ошибок, который позволяет определять отклонения частоты и выполнять корректирующие действия в случае ошибки внешнего тактового генератора.

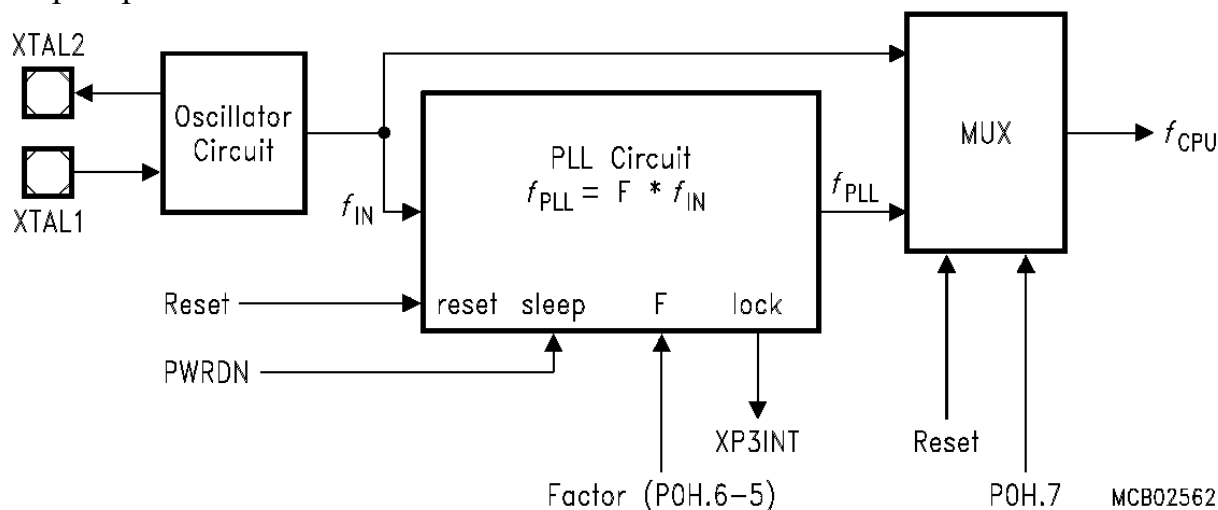


Рисунок 2-3

Блок-схема PLL

Операции с PLL

Для того чтобы тактовый сигнал вырабатывался при помощи PLL, необходимо на время “сброса” микроконтроллера падать высокий уровень напряжения на вход P0H7. После подачи питания, при достижении напряжения $V_{cc} 5V \pm 10\%$, PLL через 1 мс обеспечивает стабильный тактовый сигнал даже в случае отсутствия внешних тактовых сигналов (в этом случае PLL будет работать на базовой частоте 2..5МГц). PLL начнет попытки синхронизации с внешним тактовым генератором, как только на вход будет поступать стабильный тактовый сигнал. В течении 1 мс после стабилизации работы внешнего тактового генератора с определенной частотой, PLL захватит частоту генератора, и начнет выдавать собственные тактовые сигналы с частотой $F \cdot f_{osc}$, т.е. PLL перейдет на работу от внешнего генератора.

Примечание: Если необходима работа на стабильной частоте сразу после сброса, необходимо сохранять активный уровень напряжения на входе \overline{RSTIN} , до пор пока PLL захватит частоту внешнего генератора (в течении 1мс).

В случае определения частоты с помощью PLL, частота CPU clock выбирается с помощью умножения на необходимый коэффициент. Ниже приведена таблица с возможными значениями:

P0.15-13 (P0H.7-5)			Частота CPU $f_{CPU} = f_{XTAL} * F$	Предельные значения внешней частоты ¹⁾	Примечания
1	1	1	$f_{XTAL} * 4$	От 2.5 до 6.25 МГц	По умолчанию
1	1	0	$f_{XTAL} * 3$	От 3.33 до 8.33 МГц	
1	0	1	$f_{XTAL} * 2$	От 5 до 12.5 МГц	
1	0	0	$f_{XTAL} * 5$	От 2 до 5 МГц	
0	X	X	$f_{XTAL} * 1$	От 1 до 25 МГц	Прямая передача ²⁾

¹⁾ Использование внешнего тактового генератора устанавливает границы тактового сигнала CPU clock в пределах от 10 до 25 МГц

²⁾ В этом режиме имеет место критичная зависимость от скважности сигнала генератора. В режиме эмуляции прямая передача выбирается путем подачи «1» на вход P0.15 (P0H.7)

PLL постоянно синхронизируется с внешним тактовым генератором. Из-за того, что частота внешнего генератора составляет 1/F от выходной частоты PLL, выходная частота может быть немного выше или ниже требуемой величины. Такой разброс не вносит погрешности на больших временных интервалах. Для коротких интервалов (1..4 такта центрального процессора) погрешность составляет менее 4%.

В случае определения PLL отсутствия входного внешнего тактового сигнала, генерируется запрос на прерывание. Это предупреждающее прерывание сигнализирует о том, что частота PLL вышла из установившегося режима и не является более стабильной. Такая ошибка может иметь место в том случае, когда частота внешнего тактового генератора нестабильна, что может вызвать сломанный кристалл. В этом случае механизм синхронизации с внешней частотой отключается и PLL переходит на базовую частоту (2..5МГц). В случае потери внешнего генератора, работа от базовой частоты позволяет выполнить CPU операции экстренного сохранения.

Работа без PLL

PLL можно отключить, подав низкий уровень напряжения на время “сброса” на P0H.7. В этом случае при использовании внешних тактовых сигналов от внешнего генератора, частота ЦПУ будет равна частоте внешних тактовых сигналов $f_{OSC} = f_{CPU}$. Максимальное значение входной частоты зависит от скважности сигнала внешнего генератора, потому что необходимо принимать во внимание минимальное значение clock phases (TCLs).

2.3 Периферийные модули микроконтроллера

В семействе C167 произведено четкое разделение периферийных модулей от ядра. Эта структура позволяет производить максимальное количество параллельных операций и позволяет добавлять или удалять периферийные модули из различных моделей без изменения ядра контроллера. Каждый функциональный блок обрабатывает данные независимо, и при этом передача данных и управление осуществляется через общую шину. Периферия управляется путем записи данных в соответствующие специальные функциональные регистры (SFR). SFR расположены либо в стандартной области SFR-регистров ($00'FE00_H \dots 00'FFFF_H$) или в области дополнительных ESFR-регистров ($00'F000_H \dots 00'F1FF_H$).

Основными периферийными модулями семейства C167 являются:

- Два блока таймеров основного назначения (GPT1 и GPT2)
- Два последовательных интерфейса (ASC0 и SSC)
- Сторожевой таймер
- Два 16-канальных модуля захвата и сравнения (CAPCOM1 и CAPCOM2)
- 4-канальный ШИМ модуль
- 10-ти разрядный АЦП
- Девять портов ввода-вывода с 111 линиями ввода вывода

Каждый периферийный модуль содержит набор SFR-регистров, которые управляют работой блоков и временно хранят результаты работы. Каждый периферийный модуль имеет набор флагов состояний. Для каждого периферийного модуля индивидуально генерируется тактовый сигнал на базе сигнала CPU clock.

Интерфейсы периферийных модулей

В основном, представленная в микроконтроллере периферия имеет два типа интерфейсов. Это интерфейс для связи с ЦПУ и интерфейс для внешнего оборудования. Передача данных между ЦПУ и периферией происходит посредством SFR-регистров и прерываний. SFR-регистры управляют и отслеживают состояние периферии а также используются для передачи данных на периферию. Запросы на прерывание, посылаемые периферией, основаны на соответствующих событиях, которые имеют место во время работы периферии (такие как готовность результата, возникновение ошибки и др.).

Для связи с внешним оборудованием используются специальные выводы параллельного интерфейса. Они задействуются в случае использования функций ввода или вывода с внешней периферии. Их также

называют альтернативными функциями ввода-вывода для выводов порта, в противоположность функции ввода-вывода основного назначения.

Синхронизация периферии с ЦПУ

Работа ЦПУ и периферии основана на тактовом сигнале CPU clock (f_{CPU}). Внутренний генератор получает сигнал от кристалла или внешнего тактового генератора. Во время режима покоя сигнал CPU clock перестает вырабатываться, однако периферийное оборудование при этом продолжает свою работу. Периферийные SFR-регистры могут быть доступны один раз за такт. Если программа производит запись в SFR-регистр, и если одновременно с этим производится изменение значения SFR-регистра периферией, то операция программной записи имеет более высокий приоритет. Подробно синхронизация периферии описана в специальных разделах для каждого модуля.

2.4 Защищенные биты

В C167 предусмотрен специальный механизм защиты битов, которые могут быть аппаратно изменены, в том случае если программный доступ к связанным с ними битами может привести к непредумышленным изменениям (также см. раздел «Центральный процессор»).

Защищены следующие биты:

Регистр	Имена битов	Примечания
T2IC, T3IC, T4IC	T2IR, T3IR, T4IR	Флаги запроса на прерывание GPT1
T5IC, T6IC	T5IR, T6IR	Флаги запроса на прерывание GPT2
CRIC	CRIR	Флаг запроса на прерывание GPT2 CAPREL
T3CON, T6CON	T3OTL, T6OTL	GPTx выходные защелки
T0IC, T1IC	T0IR, T1IR	Флаги запроса на прерывание CAPCOM1
T7IR, T8IC	T7IR, T8IR	Флаги запроса на прерывание CAPCOM2
S0TIC, S0TBIC	S0TIR, S0TBIR	Флаги запроса на прерывание ASC0 передачи (буфер)
S0RIC, S0EIC	S0RIR, S0EIR	Флаги запроса на прерывание получения ли ошибки ASC0
S0CON	S0REN	Флаг разрешения ASC0 получения
SSCTIC, SSCRIC	SSCTIR, SSCRIR	Флаги запроса на прерывание SSC передачи или приема
SSCEIC	SSCEIR	Флаги запроса на прерывание SSC ошибки
SSCCON	SSCBSY	Флаг занятости SSC
SSCCON	SSCBE, SSCPE	Флаги ошибки SSC
SSCCON	SSCRE, SSCTE	Флаги ошибки SSC
ADCIC, ADEIC	ADCIR, ADEIR	Флаг запроса на прерывание переполнения или конца преобразования АЦП
ADCON	ADST, ADCRQ	Флаг начала АЦП-преобразования, флаг запроса на вставку преобразования
CC31IC...CC16IC	CC31IR...CC16IR	Флаги на запросы на прерывания CAPCOM2
CC15IC...CC0IC	CC15IR...CC0IR	Флаги на запросы на прерывания CAPCOM1
PWMIC	PWMIR	Флаги на запросы на прерывания ШИМ-модуля
TFR	TFR.15,14,13	Флаги ловушек класса А
TFR	TFR.7,3,2,1,0	Флаги ловушек класса В
P2	P2.15...P2.0	Все биты порта 2
P7	P7.7...P7.0	Все биты порта 7
P8	P8.7...P8.0	Все биты порта 8
XpyIC (y=3...0)	XpyIR (y=3...0)	Запросы на прерывание X-периферии y

Σ = 106 защищенных битов

3 Организация памяти

Пространство памяти C167 организовано по принципу архитектуры Фон-Неймана. Это означает, что программный код и данные содержатся в одном и том же линейном адресном пространстве. Все физически разделенные области памяти, включая ROM, Flash memory, внутреннюю RAM, внутреннюю область SFR- и ESFR-регистров, адресное пространство для интегрированной XBUS периферии (XRAM и CAN-модуль) и внешняя память расположены в одном общем адресном пространстве.

C167 имеет общее адресуемое пространство размером до 16 Мбайт. Оно разбито на 256 сегментов по 64 Кбайт в каждом, и каждый сегмент в свою очередь делится на четыре страницы данных по 16 Кбайт каждая (см. ниже рисунок).

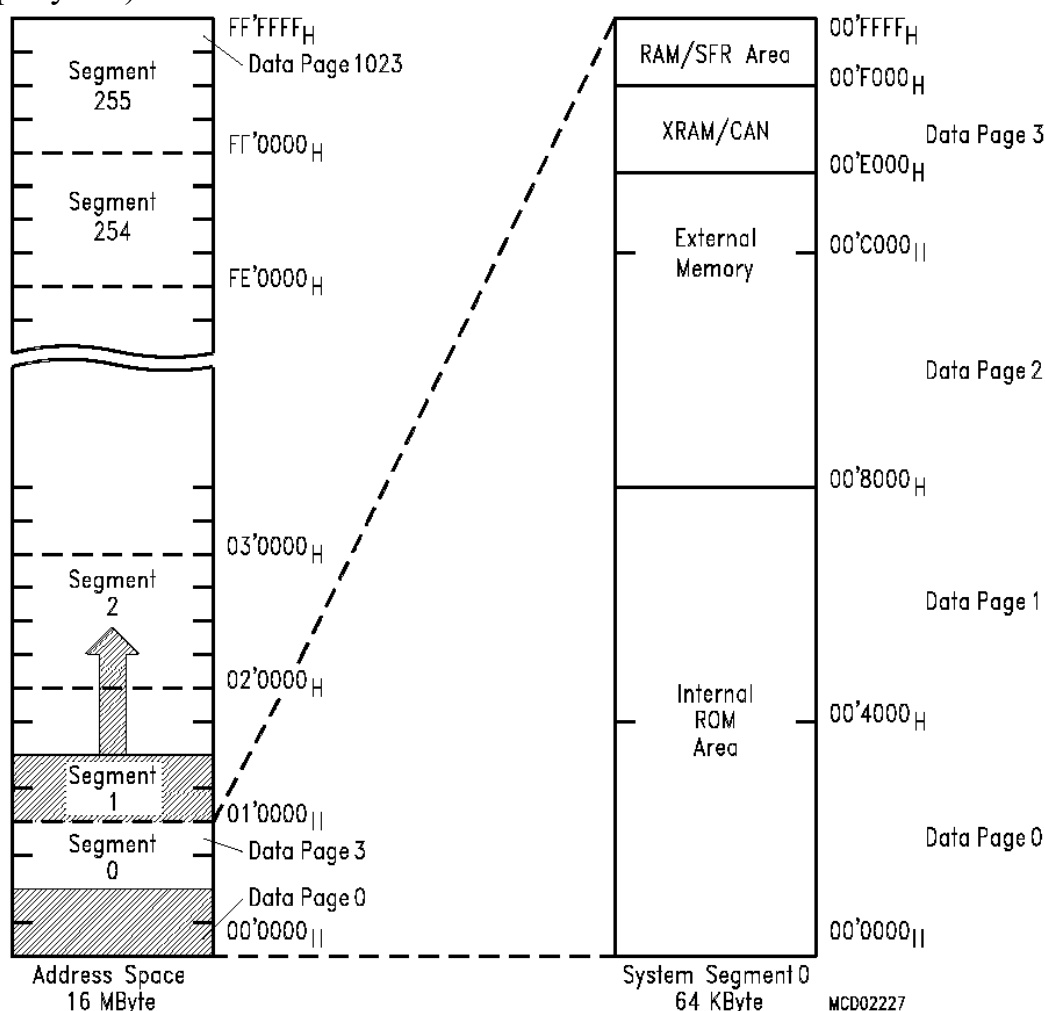


Рисунок 3-1

Область памяти и адресное пространство

Большая часть внутренней памяти находится в сегменте 0, который является системным. Верхние 4 Кбайта сегмента 0 (00`F000_H ... 00`FFFF_H) охватывают внутреннюю RAM и область SFR- и ESFR-регистров. Нижние 32 Кбайта нулевого сегмента (00`0000_H ... 00`7FFF_H) могут быть частично заняты либо внутренней ROM либо flash memory. Внутренняя ROM также может быть перемещена в сегмент 1 (01`0000_H ... 01`7FFF_H). Возможность доступа к внешней памяти в нижней части сегмента 0 или к внутренней ROM может быть отключена пользователем.

Программный код и данные могут храниться в любой части области внутренней памяти, за исключением блока SFR-регистров, которые могут быть использованы для управления и передачи данных, но не для программного кода.

Примечание: При использовании устройств не содержащих ROM, доступ к области внутренней ROM может привести к непредсказуемым результатам.

Байты могут сохраняться в четных или нечетных адресах. Слова сохраняются в восходящем порядке. Младший байт располагается в четном адресе и в следующем нечетном адресе – старший байт. Двойные слова (только программный код) располагаются в восходящем порядке как два последовательных слова. Одиночные биты всегда располагаются на отведенных им позициях для битов в адресах слов. Нулевая позиция бита имеет наименьшее значение в байте с четным адресом, и позиция 15-го бита имеет наибольшее значение в байте со следующим нечетным адресом. Битовая адресация поддерживается для части SFR-регистров, части внутренней RAM и для регистров основного назначения.

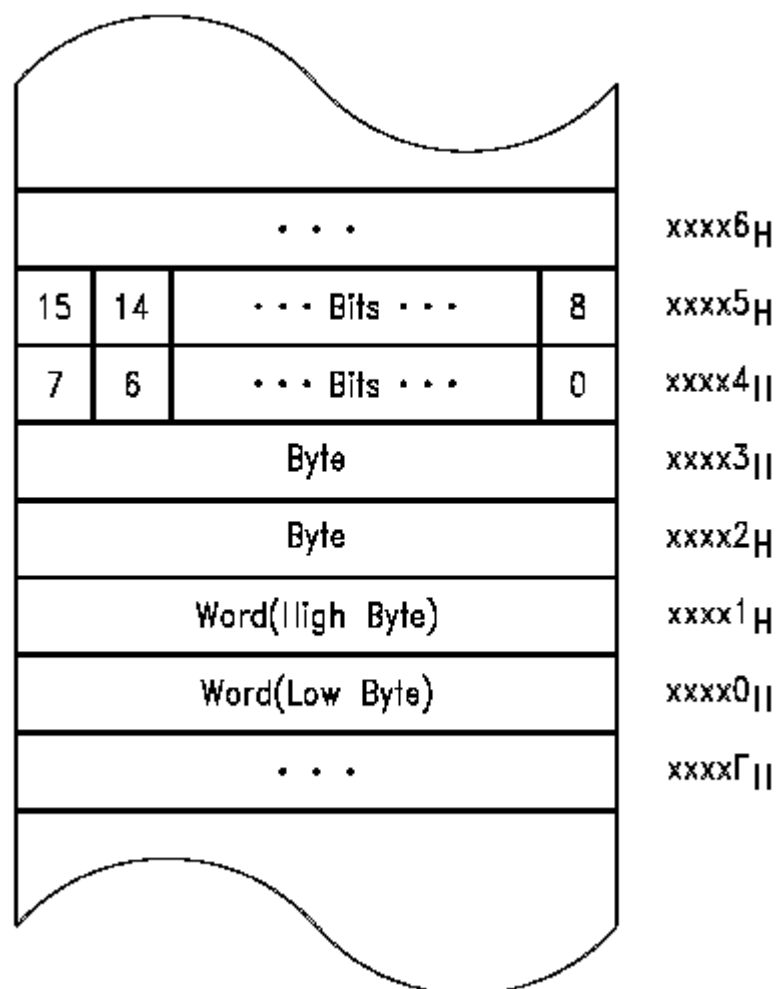


Рисунок 3-2

Хранение слов, байтов и битов в памяти, организованной по байтовому принципу

Примечание: Блоки байтов для слов или двойных слов всегда должны храниться в одинаковой физической области памяти (внутренняя, внешняя, ROM, RAM) и организованной области памяти (страница, сегмент).

3.1 Внутренняя ROM

C167 может резервировать адресное пространство в широких пределах (в зависимости от версии) для программируемой ROM (организованной, как X-32) или flash памяти. Младшие 32 Кбайта внутренней ROM или flash памяти обозначаются как область внутренней памяти. Доступ к ROM можно разрешать или запрещать при помощи бита ROMEN регистра SYSCON. Значение этого бита устанавливается во время “сброса”, в зависимости от значения на выводе \overline{EA} , или может быть программно изменено. Если в этот бит записана «1», то внутренняя ROM занимает

нижние 32 Кбайта в сегменте 0 или в сегменте 1. Размещение ROM управляется изменением значения бита ROMS1 регистра SYSCON.

Примечание: Размер области внутренней ROM не зависит от реальной физической ROM. Микроконтроллеры с объемом ROM менее чем 32 Кбайта или вообще без внутренней ROM будут иметь все ту же 32 Кбайтную область, отведенную под ROM, в случае разрешения работы с внутренней ROM. Микроконтроллеры с большим объемом памяти размещают в этом адресном пространстве только ту часть памяти, которая попадает в эти пределы.

Микроконтроллеры с ROM свыше 32 Кбайт оставшуюся часть памяти размещают в середине сегмента 1, начиная с адреса 01'8000_H.

Внутренняя ROM/Flash может использоваться как для данных, так и для программного кода.

Привязка кода всегда осуществляется к четному адресу байта. Наибольший возможный адрес кода во внутренней памяти является либо xx'xxFE_H для команд из одного слова, либо xx'xxFC_H для двухсловных команд. По этим адресам необходимо размещать команды безусловного перехода, потому что последовательный переход через границу между внутренней ROM и внешней памятью не поддерживается и может привести к ошибочным результатам.

Любой доступ для чтения слов или байтов может осуществляться с помощью косвенного или прямого 16-ти разрядного режимов адресации. Для операндов во внутренней памяти нет режима короткой адресации. Любой доступ к словам производится через четные байты адреса. Наибольший возможный адрес данных для слова во внутренней памяти: xx'xxFE_H. При передачи данных в PEC-контроллере доступ к ячейкам памяти осуществляется с помощью указателей адреса источника и назначения, независимо от содержимого DPR-регистра.

Внутренняя ROM не обеспечивает хранения однобитных данных и поэтому не является побитно адресуемой.

Внутренняя память может быть программно включена, выключена или перемещена в сегмент 0 или 1.

3.2 Внутренняя RAM и область SFR-регистров

Область RAM и SFR-регистров расположена на странице 3 нулевого сегмента. В этой области возможен доступ к 2 Кбайтам внутренней RAM (организованной как 1к*16) и к двум 512-байтным блокам SFR-регистров.

Некоторые применения RAM:

- Системный стек (программируемая часть)
- Блок регистров основного назначения (GPR)
- Указатели адресов источников и назначения для контроллера периферийных событий (PEC)
- Хранение любых данных и переменных
- Хранение программного кода

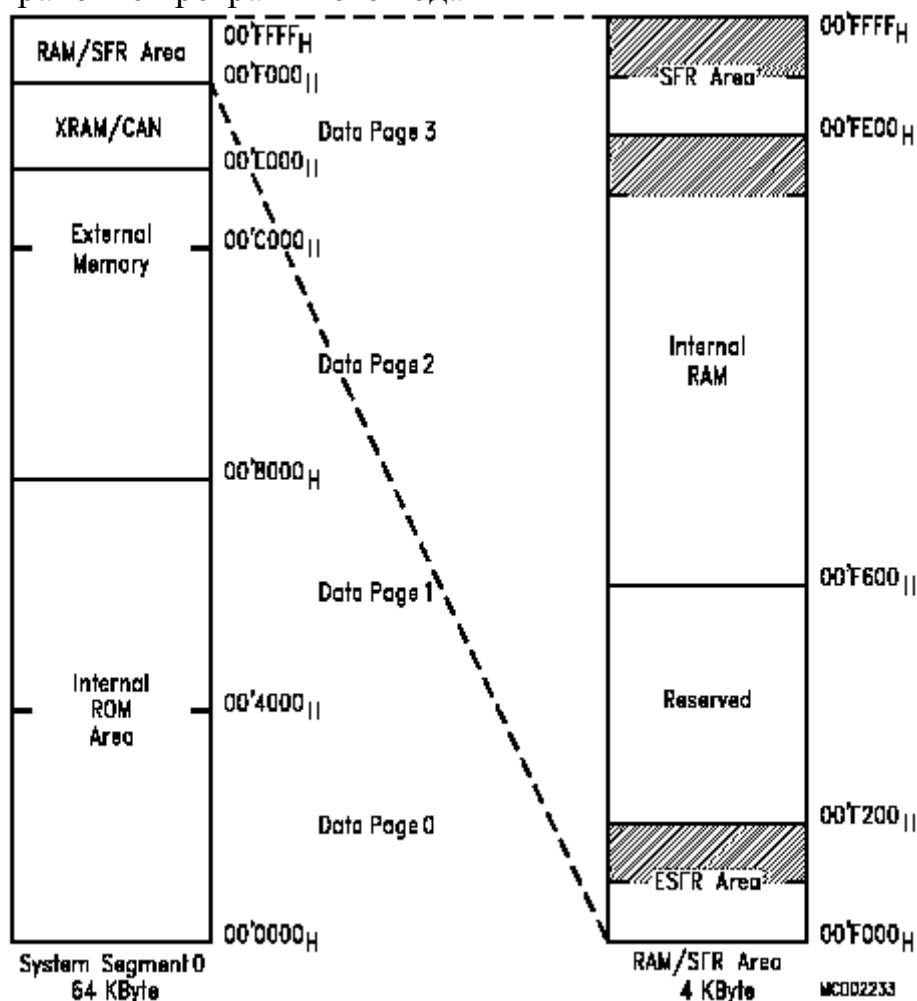


Рисунок 3-3

Область внутренней RAM и SFR-регистров

Примечание: Старшие 256 байт области SFR-регистров, область ESFR-регистров и внутренняя RAM являются побитно адресуемыми (см. затененные блоки на рисунке выше)

При обращении к коду всегда производится доступ к четным адресам байтов. Наибольший возможный адрес во внутренней RAM – либо 00`FDFE_H для команд, состоящих из одного слова, либо 00`FDFC_H для двухсловных команд. В этих адресах необходимо размещать команды безусловного перехода, потому что прямой переход из RAM в область SFR-регистров не поддерживается и может привести к ошибочным результатам.

Любые данные во внутренней памяти, состоящие из слова или байта, могут быть доступны с помощью косвенного или прямого 16-битного режима адресации. 00`FDFE_H –наибольший возможный адрес данных длиной в одно слово во внутренней памяти. Для передачи данных с помощью PEC-контроллера, внутренняя память может быть доступна независимо от содержимого DPR регистров с помощью указателей PEC-источников и точек назначения.

Сташие 256 байт внутренней памяти (от 00`FD00_H до 00`FDFF_H) и текущий банк GPR-регистров доступны для хранения однобитных данных, и таким образом возможно осуществление побитной адресации.

Системный стек

Системный стек может быть определен во внутренней RAM. Размер системного стека контролируется набором битов STKSZ регистра SYSCON (см. таблицу ниже).

<STKSZ>	Размер стека (в словах)	Адреса внутренней памяти (слова)
0 0 0 _B	256	00`FBFE _H ... 00`FA00 _H (предустановлено после сброса)
0 0 1 _B	128	00`FBFE _H ... 00`FB00 _H
0 1 0 _B	64	00`FBFE _H ... 00`FB80 _H
0 1 1 _B	32	00`FBFE _H ... 00`FBC0 _H
1 0 0 _B	512	00`FBFE _H ... 00`F800 _H
1 0 1 _B	-	Зарезервировано, не использовать
1 1 0 _B	-	Зарезервировано, не использовать
1 1 1 _B	1024	00`FDFE _H ... 00`F600 _H (Примечание: нециркулирующий стек)

Для всех операций с системным стеком внутренняя RAM доступна через регистр указателя стека (SP). Заполнение стека производится по направлению от максимального адреса к наименьшему адресу. В системном стеке возможен доступ только к словам данных. Контроль за нижним и верхним выбранным пределом стека осуществляется с помощью регистров переполнения стека (STKOV) и опустошения стека (STKUN). Эти два контрольные регистры стека могут использоваться не только для защиты от уничтожения данных, но также позволяют использовать циркулирующий

стек с аппаратной поддержкой заполнения и очистки стека (за исключением 2-кбайтного размера стека).

Советы по использованию циклического стека приведены в разделе «Системное программирование»

Регистры общего назначения

Регистры общего назначения (GPR) могут быть объединены в блок из 16 последовательных слов во внутренней RAM. Содержимое регистра Context Pointer (CP-регистр) определяет базовый адрес банка регистров, активного в настоящий момент. Банк регистров может содержать до 16 слов GPR-регистров (R0...R15) и/или до 16 байтов GPR-регистров (RL0, RH0...RL7, RH7). 16 байт GPR-регистров расположены в первых восьми словах GPR-регистров (на таблице ниже).

В противоположность системному стеку, нумерация внутри банка регистров возрастает от меньших к большим адресам и занимает в максимальном случае 32 байта. При использовании CP-регистра в качестве базового адреса (независимого от содержимого текущего DPP-регистра), GPR-регистры доступны с помощью 2-, 4- или 8-битного режима адресации. Необходимо отметить, что каждый бит в текущем активном банке доступен индивидуально.

Распределение GPR-регистров в адресах RAM:

Внутренние адреса RAM	Байтовые регистры		Регистры слов
<CP> + 1E _H	—		R15
<CP> + 1C _H	—		R14
<CP> + 1A _H	—		R13
<CP> + 18 _H	—		R12
<CP> + 16 _H	—		R11
<CP> + 14 _H	—		R10
<CP> + 12 _H	—		R9
<CP> + 10 _H	—		R8
<CP> + 0E _H	RH7	RL7	R7
<CP> + 0C _H	RH6	RL6	R6
<CP> + 0A _H	RH5	RL5	R5
<CP> + 08 _H	RH4	RL4	R4
<CP> + 06 _H	RH3	RL3	R3
<CP> + 04 _H	RH2	RL2	R2
<CP> + 02 _H	RH1	RL1	R1
<CP> + 00 _H	RH0	RL0	R0

C167 поддерживает быстрое переключение между банками регистров. Большое количество банков регистров может одновременно физически существовать во внутренней RAM. При этом является активным только выбранный в СР банк регистров. Выбор другого активного банка регистров возможен путем простого изменения значения в СР. Специальная команда переключения контекста (SCXT) обеспечивает переключение банков регистров и автоматическое сохранение предыдущего содержимого СР. Количество используемых банков регистров ограничено только размером доступного объема RAM. Советы по использованию, переключению и совмещению банков регистров описаны в разделе «Системное программирование».

Указатели адресов источников и назначения для PEC

Адреса 16-ти слов во внутренней RAM (от 00'FCE0_H до 00'FCFE_H) зарезервированы под указатели адресов источников и назначения при передачи данных по восьми PEC-каналам. Каждый канал использует пару адресов, которые хранятся в двух последовательных словах. В младшем слове (SRCP_x) хранится адрес источника, а в старшем слове (DSTP_x) – адрес назначения (x = 7...0).

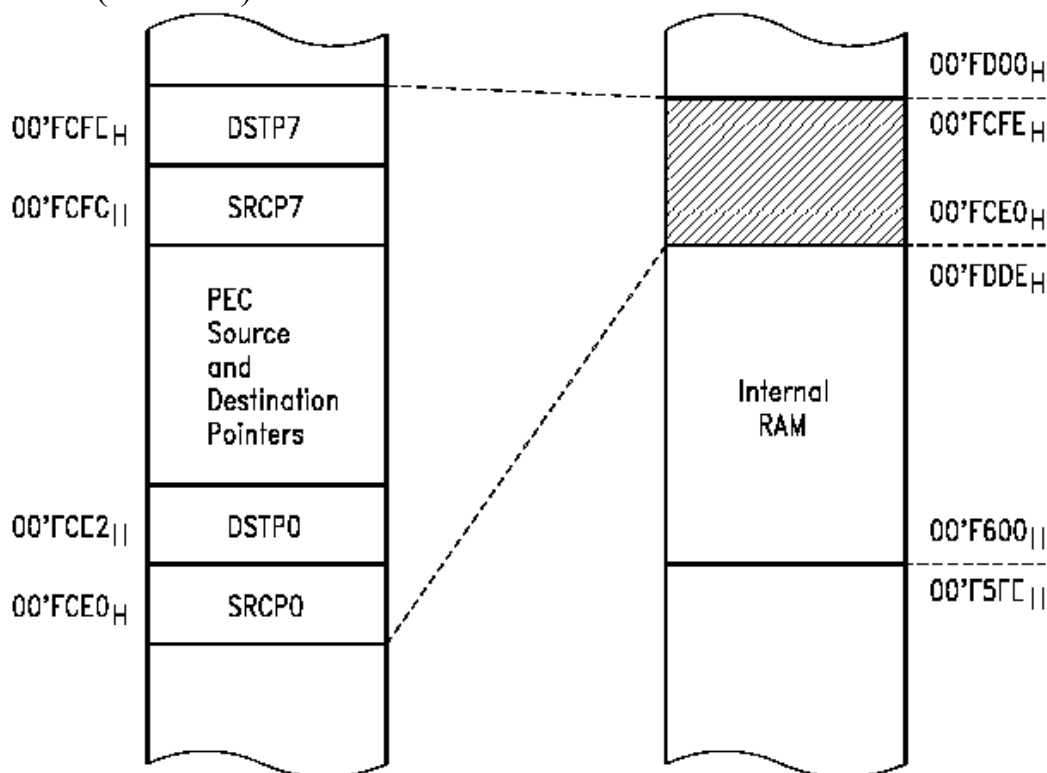


Рисунок 3-4

Расположение PEC-указателей

Всякий раз, как только совершается передача данных, пара указателей точки источника и точки назначения, выбранных по номеру PEC-канала,

получают, независимо от текущего значения DPP-регистра, доступ к этим адресам. Если PЕС-канал не используется, то область, отводящаяся под указатели точек, доступна и может быть использована для хранения байтов или слов данных.

Более детальное описание приведено в разделе «Функции прерываний и ловушек».

Регистры специальных функций (SFR-регистры)

Управление ЦПУ, интерфейсом шины, портами ввода-вывода и внутренней периферией осуществляется через SFR-регистры. SFR-регистры размещены в двух 512-байтных областях. Первый блок SFR-регистров расположен над областью внутренней RAM (00'FFFF_H ... 00'FE00_H), блок ESFR-регистров расположен ниже внутренней RAM (00'F1FF_H... 00'F000_H).

При адресации SFR-регистров может применяться как режим косвенной, так и режим 16-ти разрядной прямой адресации. Использование 8-ми разрядного смещения адреса одновременно с использованием базового адреса позволяет производить адресацию как к словам, так и к их младшим байтам. Однако при этом нельзя адресовать соответствующие старшие байты!!!

Примечание: Запись любого байта SFR-регистра приводит к тому, что значение второго байта этого слова SFR сбрасывается!!!

Верхняя половина каждого блока SFR-регистров является побитно адресуемой, поэтому биты контроля за состоянием и биты управления могут быть изменены или проверены при помощи битовой адресации.

При использовании 8-ми разрядного прямого режима адресации для доступа к регистрам в области ESFR, необходимо использовать команду EXTR, прежде чем переключать режим короткой адресации из области стандартных SFR-регистров в область ESFR-регистров. Необходимость в этом отсутствует для 16-ти разрядной прямой и косвенной адресации. GPR-регистры (R15...R0) копируются и таким образом становятся доступными без переключения из обоих блоков регистров через короткую 2-, 4- или 8-ми разрядную адресацию.

Пример

EXTR	#4	; Переключение в область ESFR ;для следующих 4 команд
MOV	ODP2, #data16	;ODP2 использует 8-ми разрядную адресацию ;регистров
BFLDL	DP6, #mask, #data8	;Битовая адресация для поля битов
BSET	DP1H.7	;Битовая адресация для одиночного бита
MOV	T8REL, R1	;T8REL использует 16-разрядную адресацию, ;R1 скопирован и также доступен через ;ESPR-режим


```

;-----      ;-----      ;(Для доступа не нужен EXTR)
;-----      ;-----      ;Зона действия команды EXTR #4
;-----      ;-----      ;заканчивается здесь
MOV      T8REL, R1      ;T8REL использует 16-разрядные адреса,
;-----      ;-----      ;R1 скопирован и не нуждается в
;-----      ;-----      ;переключении

```

Для уменьшения необходимости в использовании команды EXTR, область ESFR-регистров захватывает в основном те регистры, которые необходимы для инициализации и режима выбора. Регистры, которые часто используются, расположены в области стандартных SFR-регистров.

Примечание: Средства разработки, снабженные мониторным доступом к области ESFR-регистров, будут автоматически вставлять команды EXTR или выдавать предупреждения, в случае пропущенных или лишних команд EXTR.

3.3 Внутренняя XRAM

XRAM расположена на странице 3 и обеспечивает доступ к 2 Кбайтам внутренней памяти (созданной по принципу 1к*16). Так как XRAM подсоединена через внутреннюю XBUS, то доступ к ней организован как и ко внешней памяти, однако при этом не выполняется цикл внешней шины. Доступ к XRAM можно программно разрешить или запретить через бит XPEN регистра SYSCON. После “сброса” в этом бите хранится «0», и для организации доступа к внутренней XRAM, XPEN может быть программно изменен во время инициализации. При отключенной XRAM (предустановленно после RESET), все попытки доступа к XRAM характеризуются как попытки доступа к внешним ресурсам. XRAM может использоваться как для программного кода, так и для хранения данных.

Доступ к программному коду всегда производится через четные адреса байтов. Расположение наибольшего возможного адреса кода в XRAM – либо 00`E7FE_H для однословных команд, либо 00`E7FC_H для двухсловных команд. В этих адресах необходимо вставлять команды безусловного перехода, так как не поддерживается прямой переход из адресного пространства XRAM в адресное пространство внешней памяти. При совершении прямого перехода возможны ошибочные результаты.

Любое чтение слова или байта данных может быть доступно с помощью косвенной или прямой 16-битной адресации. Для XRAM-операндов отсутствует режим короткой адресации. Доступ к словам данных производится через четные адреса байтов. Наибольший возможный адрес слова данных в XRAM – 00`E7FE_H. XRAM может быть доступна для передачи PEC-данных независимо от содержимого DPP регистров через указатели точки источника и точки перехода.

Примечание: Так как XRAM является аналогом внешней памяти, то она не может использоваться для хранения системного стека и банков регистров. XRAM не обеспечивает сохранения одиночных битов и поэтому не является побитно адресуемой.

Внутренняя XRAM доступна без применения циклов ожидания и использует 16-разрядную демультиплексную шину, цикл которой занимает 100 нс(при частоте ЦПУ 20 МГц). Несмотря на то, что XRAM является аналогом внешней памяти, при этом не используются регистры BUSCONx / ADDRSELx, но при этом в какой то мере используются специальные XBCON/XADRS регистры. Эти регистры масочно-запрограммированы и недоступны для пользователей. Для доступа к XRAM зарезервирована область памяти с адресами от 00`E000_H до 00`E7FF_H.

Доступ к XRAM для внешних устройств

В случае установки «1» в бите XREF-SHARE регистра SYSCON, XRAM микроконтроллера C167 может быть доступна для внешних устройств, во время режима захвата через интерфейс шины C167. При этом необходимо запрограммировать режим внешнего доступа аналогично режиму доступа к XRAM: демультиплексная шина, минимальное время цикла доступа 100нс.

Примечание: Конфигурация регистра SYSCON не может быть изменена после выполнения команды EINIT.

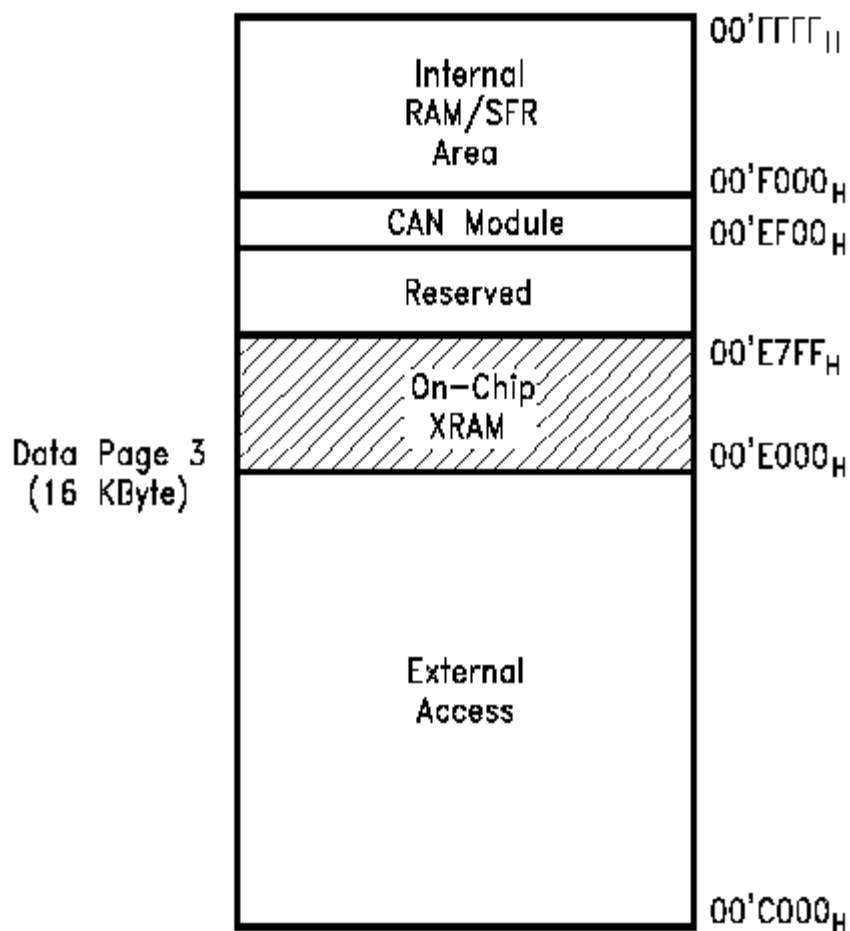


Рисунок 3-5

Область внутренней XRAM

3.4 Адресное пространство внешней памяти

C167 может использовать до 16 Мбайт адресного пространства. Только часть этого пространства используется внутренней памятью. Все адреса, не используемые внутренней памятью (ROM или RAM) и не занятые под регистры, могут быть переопределены под внешнюю память. Внешняя память доступна с помощью интерфейса внешней шины C167.

Поддерживается **четыре различных размера под банки памяти**:

- Несегментированный режим: 64 Кбайта через A15... A0 порта 0 или порта 1
- 2-битный сегментированный режим: 256 Кбайт через A17... A16 порта 4 и A15 ... A0 порта 0 или порта 1
- 4-битный сегментированный режим: 1 Мбайт через A19...A16 порта 4 и A15...A0 порта 0 или порта 1
- 8-битный сегментированный режим: 16 Мбайт через A23...A16 порта 4 и A15...A0 порта 0 или порта 1

Возможна прямая адресация к каждому банку при помощи шины адреса, одновременно с этим используется программируемый сигнал Chip Select для выбора различных банков памяти.

C167 также поддерживает 4 различных типа работы шины:

- Мультиплексная 16-разрядная шина адреса и данных через Port0 (предустанавливается после RESET)
- Мультиплексная 8-разрядная шина адреса и данных через Port0/P0L
- Демультиплексная 16-разрядная шина адреса через Port1 и шина данных через Port0
- Демультиплексная 8-разрядная шина адреса через Port1 и шина данных через P0L

Модель памяти и режим шины выбираются во время «сброса» путем подачи сигналов на вывод \overline{EA} и выводы порта 1. Для более глубокого изучения см. раздел «Интерфейс внешней шины».

Слова и байты внешних данных могут быть доступны только через режим косвенной или прямой 16-разрядной адресации, с помощью использования одного из четырех DPP регистров. Для внешних операндов отсутствует режим короткой адресации памяти. Любой доступ к словам данных производится через четный адрес байта.

Для PEC-передачи данных, может использоваться независимый от содержимого DPP-регистров доступ к внешней памяти в сегменте 0, используя указатели адресов источника и назначения.

Внешняя память не обеспечивает режим хранения битов и поэтому не адресуема побитно.

3.5 Пересечение границ различных типов памяти

Адресное пространство C167 неявно подразделяется на блоки с эквивалентными размерами с различной дискретностью и также на логические области памяти. Пересечение границ между блоками и областями (кода и данных) требует к себе повышенного внимания, при выполнении микроконтроллером необходимых операций.

Области памяти представляют собой разделенные адресные пространства, которые необходимы для обеспечения различных типов памяти. Все возможные типы памяти можно поделить на внутреннюю область RAM и SFR-регистров, внутреннюю ROM (если интегрирована), внутреннюю X-периферию (если интегрирована) и внешнюю память.

При доступе к последовательным данным в различных областях памяти не может возникать никаких проблем. Однако при исполнении программного кода, различные области памяти необходимо переключать посредством команд перехода. Прямое пересечение границ не поддерживается и может привести к ошибочным результатам.

Примечание: Изменение области внешней памяти на область RAM и SFR-регистров имеет место в сегменте 0.

Сегменты – соприкасающиеся блоки по 64 Кбайта. Для получения кода имеется возможность ссылаться на сегменты через указатель сегмента кода CSP. Для получения данных используется прямой номер сегмента без использования стандартной схемы DPP.

Во время выполнения кода, сегмент не изменяется автоматически, и поэтому его необходимо переключать.

В больших программах принято за правило, что наибольший возможный адрес сегмента содержит команду безусловного перехода к следующему фрагменту программы, для предотвращения попытки выхода из текущего сегмента.

Страницы данных – примыкающие блоки по 16 Кбайт. Доступ к ним обеспечивается посредством указателей номера страницы DPP3... 0, либо путем прямого указания номера страницы для доступа к данным. Каждый DPP-регистр может выбирать одну из возможных 1024 страниц данных. Непосредственно используемый DPP-регистр выбирается путем определения двух верхних битов в 16-битном адресе. Последовательные 16-разрядные адреса данных, пересекающих границу 16-Кбайтных страниц данных, будут использовать различные указатели страниц данных, в то время как физические адреса в памяти могут не являться последовательными.

4 Блок ЦПУ

Основным предназначением ЦПУ является выбор и декодирование команд, снабжение операндами АЛУ, выполнение операций над этими операндами в АЛУ и сохранение ранее вычисленных результатов. Так как ЦПУ является основным компонентом микроконтроллера С167, на него также оказывают влияние определенные действия периферийной подсистемы.

С помощью четырехступенчатого конвейера может параллельно обрабатываться до четырех команд. При параллельной обработке команд возможно выполнение большого количества команд С167 за один машинный цикл (100нс при частоте 20МГц). В этом разделе описывается как работа конвейера с обычной последовательностью команд, так и с командами перехода, для которых, в частности, имеется аппаратное обеспечение для ускорения выполнения. Рассматривается как стандартное так и нестандартное время выполнения команд.

Доступ к внутренней памяти осуществляется ЦПУ напрямую, а доступ к внешней памяти выполняется с помощью автоматически вызываемого ЦПУ контроллера внешней шины. Одновременно с осуществлением доступа к внешней памяти, ЦПУ по возможности продолжает выполнять операции. Когда ЦПУ нуждается в новом доступе к внешней памяти, и при этом получение необходимых внешних данных в данный момент невозможно, ЕВС возьмет на себя управление ЦПУ, до тех пор пока запрос на получение данных не будет удовлетворен.

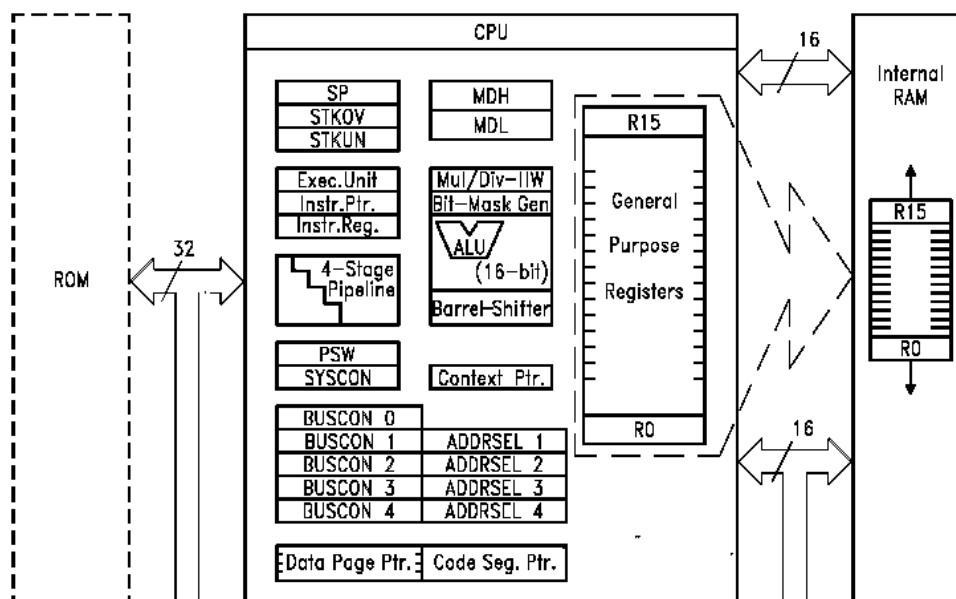


Рисунок 4-1

Блок-схема ЦПУ

Периферийные модули C167 работают независимо от ЦПУ, необходимо отметить, что для каждого модуля имеется независимый тактовый сигнал. Обмен данными и управляющими сигналами между ЦПУ и периферией производится посредством регистров специальных функций (SFR-регистров). Всякий раз, когда периферия нуждается в реакции ЦПУ, внутренний контроллер прерываний сравнивает все поступившие от периферийных устройств запросы друг с другом и наделяет один из них высшим приоритетом. Если приоритеты текущих операций ЦПУ меньше, чем приоритеты выбранных периферийных запросов, будет обслужено прерывание.

Есть два типа реакции на прерывания:

- **Стандартные действия с прерываниями** заключаются в том, что ЦПУ сохраняет состояние текущей программы и возвратный адрес в стеке, прежде чем перейти к адресу кода по таблице векторов прерываний.
- **Работа с прерываниями при помощи PEC-контроллера** отнимает только один машинный такт от работы ЦПУ над текущей программой для совершения передачи данных с помощью PEC-контроллера.

Системные ошибки, определяемые по ходу выполнения программы, (так называемые аппаратные ловушки) или внешние немаскируемые прерывания также обслуживаются как стандартные прерывания с очень высоким приоритетом.

В противоположность другим периферийным устройствам между сторожевым таймером и ЦПУ имеет место закрытое соединение. Если включить сторожевой таймер в рабочий режим, то он будет ожидать сигналы от ЦПУ через программируемые промежутки времени, иначе сторожевой таймер подаст сигнал на “сброс”. Таким образом сторожевой таймер может оберегать ЦПУ от работы в ошибочном режиме при выполнении кода программы с ошибками. После “сброса” системы сторожевой таймер начинает отсчет времени автоматически, но его, при необходимости, можно программно отключить.

По сравнению с нормальным режимом работы можно выделить следующие режимы:

- Режим “сброс”: к нему можно причислить любой из перезапусков (аппаратный, программный либо с использованием сторожевого таймера). Любой перезапуск принуждает ЦПУ вернуться в предустановленный активный режим.
- Режим покоя: Тактовый сигнал ЦПУ отключен, в то время как тактовые сигналы для периферии продолжают подаваться.
- Режим отключения питания: Все тактовые сигналы выключены.

Для функционирования ядра ЦПУ предназначен набор регистров специальных функций:

- Основная конфигурация системы :SYSCON (RP0H)
- Индикация и управление состоянием ЦПУ :PSW
- Управление доступа к данным :IP, CSP
- Управление страницами данных :DPP0-DPP3
- Управление доступа к GPR-регистрам :CP
- Управление доступа к системному стеку :SP, STKUN, STKOV
- Поддержка умножения и деления :MDL, MDH, MDC
- Поддержка констант ЦПУ :ZEROS, ONES

4.1 Конвейерная обработка команд

Конвейер команд C167 разделяет выполнение команд на 4 ступени, каждая из которых имеет индивидуальное название:

1-ая -> Выборка:

На этой стадии команды, выбранные с помощью Указателя команд (IP) и указателя сегмента кода (CSP), вызываются либо из внутренней ROM, либо внутренней RAM, либо из внешней памяти.

2-ая -> Декодирование

На этой стадии команды декодируются и если необходимо вычисляются адреса операндов, и вызываются необходимые операнды. Для всех команд, которым необходим доступ к системному стеку, значение SP-регистра по необходимости либо инкрементируется, либо декрементируется. В случае декодирования команды перехода, указатель команд и указатель сегмента кода изменяют свое значение на необходимое для обеспечения перехода.

3-я -> Выполнение:

На этой стадии осуществляется операции в АЛУ, над предварительно вызванными операндами. При этом изменяются, соответственно команде, флаги состояний в PSW-регистре. Во время этой стадии также осуществляются все прямые записи в память SFR-регистров, а также осуществляется запись автоматически инкрементированных и декрементированных чисел, использующих указатель косвенной адресации.

4-ая -> обратная запись:

На этой стадии все внешние операнды и оставшиеся операнды во внутренней RAM записываются обратно.

Особенностью C167 является наличие инжектированных команд. Такие команды распознаются системой и обеспечивается их выполнение, которое не может быть завершено за один такт. Эти команды автоматически вставляются в стадию декодирования конвейера и проходят оставшиеся

стадии в режиме похожем на стандартный. Инжектированные команды также применяются для программных прерываний.

Последовательное выполнение команд

Каждая команда проходит через все четыре стадии, не обращая внимание на возможное отсутствие необходимости выполнения каждой стадии. Выполнение команд отнимает как минимум четыре такта, так как для выполнения одной стадии необходим, как минимум, один машинный такт. Однако конвейер позволяет параллельно обрабатывать до четырех команд. Поэтому при выполнении многих команд кажется, что они выполняются в течении одного машинного такта. Этот эффект начинает сказываться, сразу после заполнения конвейера командами после системного «сброса» (на рисунке ниже).

Конвейерная обработка команд увеличивает среднюю пропускную способность конвейера.

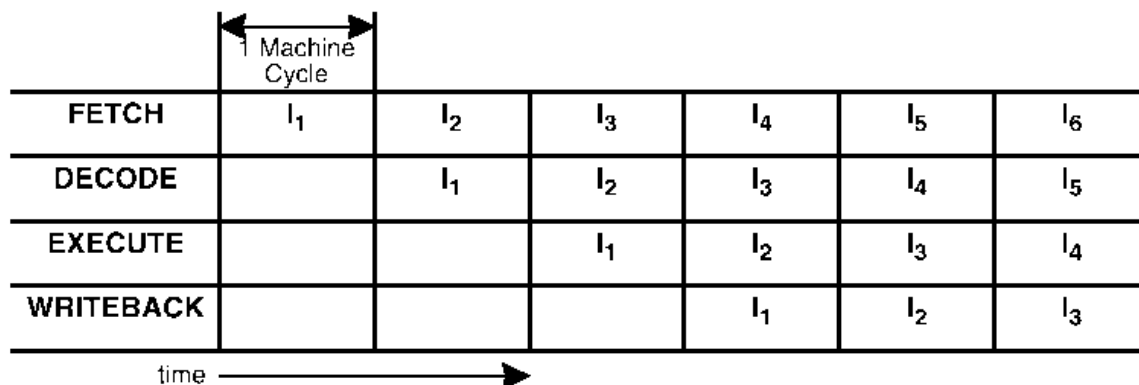


Рисунок 4-2

Последовательная конвейерная обработка команд

Стандартное выполнение команд перехода

Командный конвейер помогает ускорить выполнение программы. В случае совершения перехода, команда, заранее вызванная в конвейер, зачастую не является командой, которая должна быть декодирована следующей. Таким образом, в нормальном режиме необходим как минимум один дополнительный машинный такт для вызова команды, являющейся целью перехода. Дополнительный машинный такт создается с помощью инжекции команды (на рисунке ниже).

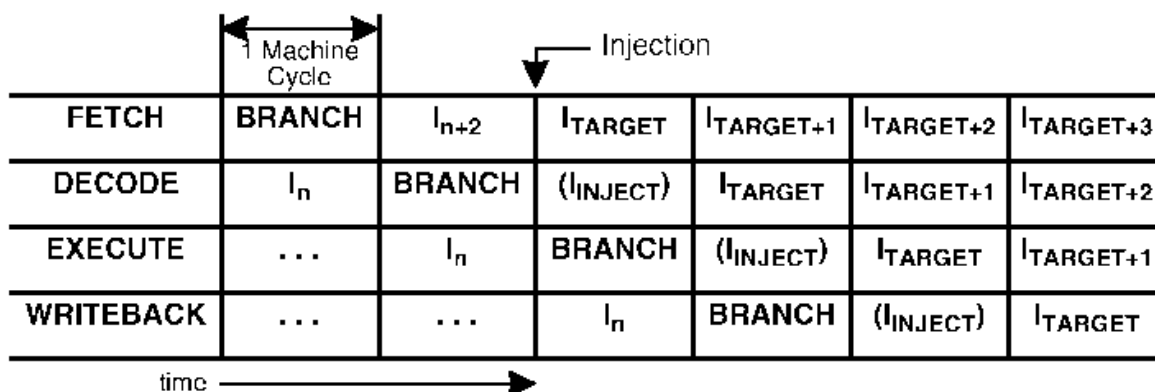


Рисунок 4-3

Стандартная конвейерная обработка команд перехода

В случае не выполнения условия для совершения условного перехода, в ходе выполнения программы отсутствуют изменения, и таким образом не нужен дополнительный такт. В этом случае команда, следующая за командой условного перехода, в начале следующего машинного такта после декодирования команды условного перехода, входит в стадию декодирования.

Выполнение кэшированных команд перехода

В C167 включен кэш переходов для оптимизации условных переходов по одному адресу. При совершении кэшированного перехода, может быть сохранен лишний такт, расходуемый на выборку команды цели перехода, и таким образом команда, соответствующая кэшу перехода, в большинстве случаев использует только один машинный такт.

Для достижения этого используется следующий механизм:

При прохождении кэшируемой команды перехода через стадию выборки в первый раз (при этом выполняется условие перехода), команда цели перехода вызывается как обычная команда, используя временную задержку в один машинный такт. Однако в противоположность стандартным командам перехода, команда цели перехода, после того как произведена выборка команды, дополнительно сохраняется в кэше команд. Кэшируемые команды перехода – JMPA, JMPR, JB, JBC, JNB, JNBS.

После каждого последующего выполнения этой кэшированной команды перехода, команда цели перехода не вызывается из программной памяти, а забирается из кэша и немедленно инжектируется в стадию декодирования конвейера (на рисунке ниже).

Уменьшение времени выполнения кэшированной команды перехода всегда имеет место при втором и последующем использовании этой команды. Команда сохраняется в кэше переходов, до тех пор пока между

двумя одинаковыми кэшируемыми командами перехода не будет выполнена команда, имеющая возможность изменения фундаментальных регистров (JMPS, CALLS, RETS, TRAP, RETI), или пока не будет выполнено стандартное прерывание.

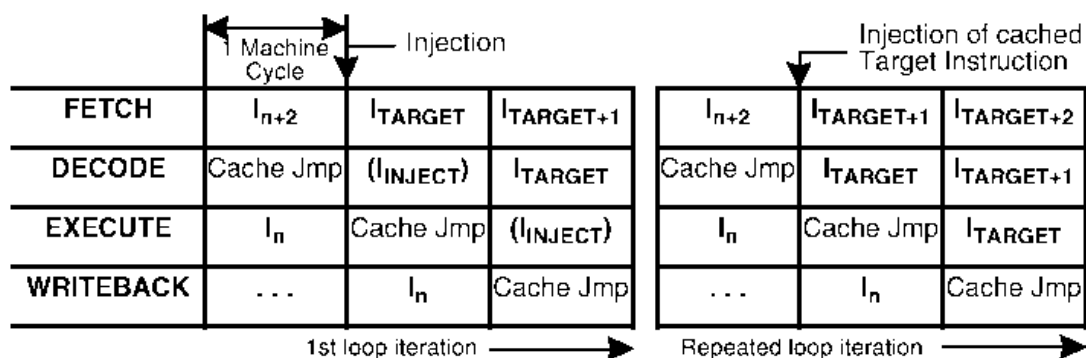


Рисунок 4-4

Конвейерная обработка кэшируемых команд перехода

Особые эффекты конвейера

Параллельно может выполняться до четырех различных команд, и поэтому в C167 было создана дополнительная логика для обнаружения зависимостей команд, которые не могут находиться на различных стадиях конвейера без потери производительности. Эта дополнительная логика (для чтения и записи вычисленных операндов до номинального исполнения этих команд) во время оптимизации последовательности выполнения команд выявляет большинство возможных конфликтов, в том числе конфликты одновременного использования шины несколькими командами, и таким образом в большинстве случаев конвейерная обработка команд прозрачна для пользователей. Однако в некоторых очень редких случаях программистам следует уделять внимание тому, что C167 – конвейерная машина. В этих случаях задержки, вызванные конфликтами конвейерной обработки, могут использоваться для выполнения других команд для повышения производительности.

• Изменение значения Context Pointer

Команды, вычисляющие физические адреса GPR-операндов при помощи СР-регистра, обычно не способны использовать новое значение СР-регистра, измененное в предыдущей команде. Таким образом, для уверенности в использовании нового значения СР, необходимо добавить как минимум одну команду между командами изменения СР-регистра и следующей командой, использующей GPR-регистры, как показано на следующем примере:

I_n	:SCXT	CP, #0FC00h	;запись нового значения CP
I_{n+1}	:....		;команда, не использующая GPR
I_{n+2}	:MOV	R0, #dataX	;запись в GPR0 новых данных

• Изменение значения указателя страницы данных

Команды, вычисляющие физический адрес операндов с помощью DPPn-регистров ($n = 0...3$), не могут использовать новое значение DPPn-регистра, измененное в непосредственно предшествующей команде. Поэтому для уверенности в использовании нового значения DPPn-регистра, необходимо вставить как минимум одну команду между командой записи нового значения DPPn-регистра и следующей командой, использующую режим прямой или косвенной адресации, как показано в следующем примере:

I_n	: MOV	DPP0, #4	;выбор страницы данных 4 ;с помощью DPP0
I_{n+1}	:		;команда, не использующая DPP0
I_{n+2}	: MOV	DPP0:0000H, R1	;перемещение содержимого R1 ;по адресу 01'0000 _H (в странице 4) ;сегментация включена

• Изменение значения указателя стека

Ни одна из команд RET, RETI, RETS, RETP или POP не может корректно использовать новое значение SP-регистра, измененное непосредственно перед выполнением этой команды. Таким образом для доступа к стеку по новому значению SP-регистра, необходимо вставить как минимум одну команду между командой записи в SP-регистр и одной из, использующих SP-регистр, вышеупомянутых команд, как показано на следующем примере:

I_n	:MOV	SP, #0FA40h	;выбор новой вершины стека
I_{n+1}	:....		;команда, не использующая ;возврат операндов из системного стека
I_{n+2}	:POP	R0	;возврат значения слова из вершины ;нового стека в R0

• Последовательность действий при доступе к внешней памяти

Команды, находящиеся на различных стадиях конвейера, могут одновременно выдавать запросы на контроллер внешней шины (ЕВС). Последовательность команд, выполняемых ЦПУ, может отличаться от последовательности доступа к внешней памяти, совершаемого ЕВС. Различные операции доступа к внешней памяти имеют различные приоритеты:

1. Чтение данных
2. Выборка кода
3. Запись данных

• Управление прерываниями

Программные изменения PSW (как прямые, так и неявно выраженные) производятся в стадии выполнения соответствующих команд. При внесении изменений в уровни приоритетов, новые уровни приоритетов будут использоваться только после окончания следующей команды. Иными словами, после команды, отключающей прерывания с помощью IEN или ILVL, в течении следующего такта может быть получен ответ на запрос на прерывание. Таким образом, критичную ко времени исполнения последовательность команд не следует начинать после команды отключающей прерывания, как показано на следующем примере:

```
INT_OFF:      BCLR      IEN      ;полное запрещение прерываний
              IN-1          ;не критичная команда ко времени
CRIT_1ST: IN          ;начало непрерываемой, критичной
              ;ко времени исполнения, зависимости

              ...
CRIT_LAST    IN+X          ;конец зависимости
INT_ON       BSET      IEN      ;общее разрешение прерываний
```

Примечание: Описываемая задержка в один такт, также имеет место в случае разрешения прерываний, т.е. не признается запрос на прерывание до выполнения следующей команды.

• Инициализация выводов портов

Изменение направления выводов портов (ввод или вывод) вступает в силу, только после команды, следующей за командой изменения направления. Так как битовые команды (BSET, BCLR) используют зависимость чтение-изменение-запись, доступную для всего порта, то команда, следующая за командой изменения направления порта, не должна иметь доступа к этому порту.

Не правильно:

```
BSET      DP3.13      ;Определение P3.13 на вывод
BSET      P3.5         ;P3.13 все еще определен на вход
              ;Механизм чтение-изменение-запись
              ;читает P3.13
```

```
Правильно: BSET      DP3.13      ;Определение P3.13 на вывод
              NOP              ;Любая команда не использующая Port3
              BSET      P3.5         ;P3.13 уже определен на вход
              ;Механизм чтение-изменение-запись
              ;читает значение P3.13
```

- **Изменение конфигурации системы**

Команда, следующая за командой, изменяющей конфигурацию системы путем перезаписи содержимого регистра SYSCON (т.е. расположение внутренней ROM, сегментация, размер стека), не может использовать новую конфигурацию. В этом случае необходимо вставить команду, не имеющую доступа к новым ресурсам. Доступ к программному коду в новой области ROM будет возможен только после команды безусловного перехода к этой области.

Примечание: Как правило, команды, изменяющие расположение ROM, должны исполняться из внутренней RAM или внешней памяти.

- **BUSCON/ADDRSEL**

Команда, следующая за командой, изменяющей свойства внешней адресной области, не может использовать операнды из новой области. В этом случае необходимо вставить команду, не использующую доступа к этим адресам. Доступ к коду в области новых адресов будет получен только после команды безусловного перехода в эту область.

Примечание: Как правило команды, выполняющие изменение свойств внешней шины, не располагаются в изменяемой области внешней памяти.

- **Временные зависимости**

Конвейерная обработка команд уменьшает среднее время выполнения команд (в большинстве случаев вместо четырех тактов используется один). Однако в редких случаях, при возникновении особых ситуации в конвейере, время выполнения команды увеличивается либо на половину, либо на целый машинный такт. Хотя дополнительно затраченное время представляет собой только малую долю от времени исполнения всей программы, в некоторых случаях необходимо избегать задержек при исполнении критичных ко времени модулей программы.

Следующий раздел посвящен описанию некоторых способов оптимизации частей программ критичных ко времени исполнения для конвейерной обработки.

4.2 Битовое управление и битовая защита

C167 имеет несколько механизмов для битового управления. Эти механизмы либо манипулируют с битами регистров управления, управляя внутренней периферией, либо управляют функциями ввода-вывода с помощью выводов портов.

Команды BSET, BCLR, BAND, BOR, BXOR, BMOV, BMOVN устанавливают или очищают значения необходимых битов. Команды BFLDL и BFLDH позволяют одновременно изменять до 8 битов байта. Команды JBC и JNB (а также команды условного перехода) оценивают значения битов, для определения возможности совершения перехода.

Примечание: Битовые операции, совершаемые с неопределенными областями памяти, всегда считывают нулевое значение бита, при этом операция записи не дает никакого эффекта в данной битовой области.

Все, совершающие операции с битами или группами битов, команды используют внутренний механизм чтение-изменение-запись, который обеспечивает доступ ко всему слову, которое содержит необходимый бит(ы).

Этот метод имеет несколько особенностей:

- Могут быть изменены только, расположенные во внутренней области адресов биты (т.е. внутренняя RAM и SFR-регистры). Внешние адреса не могут использоваться для битовых команд

Побитно адресуемы только верхние 256 байт SFR-регистров, область ESFR-регистров и внутренняя RAM (см. раздел «Организация памяти»). Для других SFR-регистров необходимо применять команды для работы с байтами и словами.

Примечание: Все GPR-регистры, независимо от расположения банка регистров, побитно адресуемы, с помощью Context Pointer (CP). Даже те GPR-регистры, которые расположены вне области побитно адресуемой памяти, имеют возможность побитной адресации.

- Механизм чтение-изменение-запись может быть опасным при обращении к битам, показывающим аппаратное состояние системы. В этом случае микроконтроллер может аппаратно изменить необходимый бит, во время совершения операции чтение-изменение-запись. При этом во время обратной записи возможна перезапись нового значения бита, сгенерированного микроконтроллером. Для решения этой проблемы, либо обеспечивается внутренняя защита (см. ниже), либо используется специальное программирование (см. «Особые эффекты конвейера»).

Защищенные биты не изменяют своего значения во время выполнения последовательности чтение-изменение-запись. Аппаратная логика защиты гарантирует, что операция обратной записи будет иметь эффект только со значащими битами.

Примечание: В случае попытки аппаратного доступа одновременно с программным изменением значения бита, более высокий приоритет имеет программный доступ к этому биту.

Все защищенные биты, включенные в C167, приведены в конце главы «Обзор архитектуры микроконтроллера».

4.3 Время выполнения команд (Instruction State Time)

Время выполнения команд зависит от месторасположения исходного кода, месторасположения считываемых операндов и месторасположения результатов операции. Самый быстрый режим работы C167 обеспечивается при использовании программного кода, расположенного во внутренней ROM. В этом случае большинство команд может обрабатываться за один машинный такт, при этом достигается минимально возможное время выполнения команд.

Все попытки доступа к внешним ресурсам осуществляются с помощью контроллера внешней шины (EBC), работающего параллельно с ЦПУ.

Детальное описание времени выполнения различных команд можно найти в «Описании набора команд семейства микроконтроллеров C16x».

4.4 SFR-регистры

Ядро ЦПУ нуждается в наборе SFR-регистров для хранения информации о состоянии системы, для хранения необходимых АЛУ констант для адресации регистров, для управления системой и конфигурацией шины, а также для операций умножения и деления в АЛУ, сегментации памяти кода, разбития памяти данных на страницы и доступа к регистрам основного назначения и системному стеку.

Механизм доступа к этим SFR-регистрам в ядре ЦПУ идентичен механизму доступа для других SFR-регистров. Так как все SFR-регистры могут управляться посредством любых команд, действительных для адресного пространства SFR-регистров, то отпадает необходимость в создании набора специальных системных команд.

Однако заметим, что к некоторым SFR-регистрам ядра ЦПУ ограничен доступ, что необходимо для обеспечения выполнения операций процессором. Ни при каком условии нельзя осуществить доступ к указателю команд IP и указателю сегмента кода CSP. Их значения могут быть только косвенно изменены посредством команд перехода.

PSW-, SP- и MDS-регрстр могут быть изменены не только с помощью прямых команд, но и посредством выполнения специальных команд ЦПУ.

Примечание: Любые операции записи одного байта в SFR-регистр очищают значение связанного с ним другого байта слова SFR-регрстра.

Не используемые (зарезервированные) биты SFR-регистров не могут быть изменены, и всегда выдают значение «0» при попытке чтения.

Системная конфигурация регистра SYSCON

Этот побитно адресуемый регистр управляет общей системной конфигурацией и функцией управления. Значение регистра SYSCON после RESET зависит от сигналов на выходах PORT0 во время RESET (см. аппаратно значащие выводы).

SFR
SYSCON (FF12_H/89_H) **Значение после RESET: 0XX0_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STKSZ		ROM S1	SGT DIS	ROM EN	BYT DIS	CLK EN	WR CFG	—	—	—	—	XPEN	VISIBL	XPER-SHARE	
rw		rw	rw	rw	rw	Rw	rw	-	-	-	-	rw	rw	rw	

Бит	Функция
XPER-SHARE	Управление доступом к XBUS периферии 0 – Внешний доступ к XBUS периферии выключен 1 – XBUS периферия доступна через внешнюю шину во время режима захвата
VISIBLE	Управление видимым режимом 0 – Доступ к XBUS периферии производится внутренне 1 – Доступ к XBUS периферии проявляется на внешних выводах
XPEN	Бит управления XBUS периферией 0 – Доступ к внутренней X-периферии и ее функциям отключен 1 – Внутренняя X-периферия включена и доступна Примечание: Этот бит имеет смысл только для моделей с X-периферией
WRCFG	Управление конфигурацией записи (устанавливается согласно P0H.0) 0 – Выводы \overline{WR} и \overline{BHE} сохраняют свою нормальную функцию 1 – Вывод \overline{WR} действует как \overline{WRL} , вывод \overline{BHE} как \overline{WRH}
CLKEN	Включение вывода системного тактового сигнала (CLKOUT) 0 – CLKOUT отключен: вход микросхемы может использоваться для функции ввода/вывода основного назначения 1 – CLKOUT включен: ножка используется для тактового системного сигнала
BYTDIS	Включение/выключение управления для вывода BHE (устанавливается согласно размеру шины данных) 0 – Вывод \overline{BHE} включен 1 – Вывод \overline{BHE} выключен, вывод микросхемы может быть использован для функции ввода/вывода основного назначения

Бит	Функция
ROMEN	Включение внутренней ROM (устанавливается согласно значению на выводе \overline{EA} во время RESET) 0 – Внутренняя память отключена: доступ к области ROM производится через внешнюю шину 1 – Внутренняя ROM включена
SGTDIS	Выключение/Включение управления сегментацией 0 – Сегментация включена (значение CSP сохраняется / восстанавливается во время входа / выхода из прерывания) 1 – Сегментация выключена (только IP сохраняется/восстанавливается)
ROMS1	Размещение внутренней ROM 0 – Внутренняя ROM в сегменте 0 (00'0000 _H ... 00'7FFF _H) 1 – Внутренняя ROM в сегменте 1 (01'0000 _H ... 01'7FFF _H)
STKSZ	Размер системного стека Выбирается размер системного стека (во внутренней RAM) от 32 до 1024 слов

Примечание: Регистр SYSCON не может быть изменен после выполнения команды EINIT.

Функции битов XPER-SHARE, VISIBLE, WVCFG, BYTDIS, ROMEN и ROMS1 описаны более подробно в разделе «Контроллер внешней шины».

Включение вывода системного тактового сигнала (CLKEN)

Функция вывода системного тактового сигнала включена путем установки в «1» бита CLKEN регистра SYSCON. Если эта функция включена, вывод порта P3.15 берет на себя альтернативную функцию вывода сигнала CLKOUT. Выходной тактовый сигнал имеет скважность 50% на всех частотах равных частотам работы ЦПУ ($f_{OUT} = f_{CPU}$).

Примечание: Выходной драйвер вывода порта P3.15 автоматически переключается на тактовый сигнал, когда функция CLKOUT включена. Бит направления порта при этом игнорируется.

После RESET функция выходного тактового сигнала отключена.

Выключение/Включение управления сегментацией (SGTDIS)

Бит SGTDIS позволяет выбирать между сегментированным и несегментированным режимами памяти.

В несегментированном режиме все адресное пространство ограничено 64 Кбайтами (сегмент 0) и поэтому 16 битов достаточно для представления всего адреса. Для операторов, неявно использующих стек

(CALL или RET) значение CSP-регистра не сохраняется, и при этом только значение IP сохраняется/восстанавливается из стека.

В режиме сегментированной памяти для команд доступно все адресное пространство. Для команд, неявно использующих стек (CALL или RET), и CSP и IP охраняются в стеке и восстанавливаются из стека. После RESET, по умолчанию установлен режим сегментированной памяти.

Примечание: При задаче режима посредством бита SGTDIS, автоматически определяется необходимость в сохранении в стеке значения CSP-регистра вместе со значением IP-регистра, до входа в подпрограмму обслуживания прерывания.

Размер системного стека (STKSZ)

Это битовое поле определяет физический размер системного стека, расположенного во внутренней RAM C167. Область, состоящая из 32...512 слов или всего объема внутренней RAM, может быть использована под системный стек. Так называемый механизм «циркулирующего стека» позволяет использовать больший виртуальный стек, чем предустановленный объем RAM.

Поле STKSZ описывается в разделе «Системное программирование».

Регистр слова состояния процессора (PSW)

В этом побитно адресуемом регистре отражается текущее состояние микроконтроллера. Две группы битов отражают текущее состояние АЛУ, и текущее состояние прерываний ЦПУ. Независимый бит (USR0) регистра PSW предназначен для использования пользователем в качестве флага общего назначения.

SFR

PSW (FF10_H/88_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ILVL				IEN	HLD EN	-	-	-	USR0	MUL IP	E	Z	V	C	H
rw				rw	rw	-	-	-	rw	rw	Rw	rw	rw	rw	rw

Бит	Функция
N	Отрицательный результат Устанавливается, когда результат операции в АЛУ отрицателен
C	Флаг переноса Устанавливается, когда результат операции в АЛУ создает бит переноса
V	Переполнение результата вычисления Устанавливается, когда в результате операции в АЛУ имеет место переполнение
Z	Флаг нулевого результата Устанавливается, когда имеет место нулевой результат операции в АЛУ
E	Флаг конца таблицы Устанавливается, когда адрес исходного операнда команды 8000 _H или 80 _H
MULIP	Флаг выполнения операции умножения и деления 0 – отсутствие выполнения операции умножения и деления 1 – умножение и деление были прерваны
USR0	Пользовательский флаг основного назначения
HLDEN, ILVL, IEN	Поле управления прерываниями и EBC Показывает запросы на прерывания и включает управление внешней шиной

Флаги состояния ALU (N, C,V,Z,E, MULIP)

Флаги состояния АЛУ регистра PSW показывают состояние после выполнения последней операции в АЛУ. Эти флаги устанавливаются после большинства команд и зависят от специальных правил, зависящих от операции в АЛУ, или от операции перемещения данных.

После выполнения команды, которая явно изменяет состояние PSW-регистра, флаги состояния не могут быть распознаны, как описано, так как явное чтение состояния PSW-регистра сменит значения флагов состояния, потому что они генерируются ЦПУ. Явное чтение PSW-регистра выдаст результат в виде значения, которое показывает состояние PSW-регистра после выполнения непосредственно предшествовавшей операции.

Примечание: После RESET все биты в регистре, отвечающие за состояние АЛУ, очищаются.

- **N-флаг:** Для большинства операций с АЛУ, N-флаг устанавливается в единицу в том случае, если старший значащий бит в результате операции содержит «1», в противоположном случае флаг устанавливается в «0». В случае целочисленных (integer) операций, N-флаг

может быть интерпретирован как бит знака в результате операции (отрицательный $N=1$, положительный $N=0$). Отрицательные числа всегда представляются в виде дополнительных чисел. Возможный диапазон чисел со знаком: в случае данных из одного слова – от -8000_H до $+7FFF_H$, или для однобайтных данных – от -80_H до $+7F_H$. При использовании Boolean битовых операций с одним операндом, N-флаг показывает предыдущее состояние изменяемого бита. Для Boolean битовых операций с двумя операндами N-флаг показывает результат операции XOR с двумя изменяемыми битами.

- **С-флаг:** После операции сложения он показывает наличие переноса из старшего значащего бита в вычисляемом байте или слове. После вычитания или сравнения С-флаг показывает заимствованный бит, который представляет из себя отрицательную логическую величину.

Это означает, что С-флаг устанавливается в «1», если НЕТ переноса из старшего значащего бита вычисляемого слова или байта данных во время операции вычитания, которая выполняется путем двух дополнительных сложений, и С-флаг устанавливается в «0», когда дополнительное сложение приводит к переносу.

С-флаг всегда устанавливается в «0» для логических операций, операций умножения и деления, потому что эти операции не могут вызвать переноса ни при каких условиях.

Для операций сдвига битов и циклического сдвига битов, в С-флаг подставляется значение бита, который был изменен в слове или байте данных последним. Если итог операции был «0», то С-флаг принимает нулевое значение.

Для Boolean операций с битами с одним операндом С-флаг всегда принимает «0» значение. Для Boolean битовых операций с двумя операндами С-флаг принимает результат операции AND с двумя соответствующими битами.

- **V-флаг:** Для сложения, вычитания, операции дополнения (2's complement) V-флаг всегда принимает значение «1», если результат переходит граничные значения для знаковых чисел, (для слов от -8000_H до $+7FFF_H$, или для байтов от -80_H до $+7F_H$). Заметим, что результат целочисленного сложения, целочисленного вычитания или 2's complement не является корректным, если V-флаг показывает арифметическое переполнение.

Для умножения и деления V-флаг устанавливается в «1», если результат не может быть представлен в словесном типе данных. Заметим, что деление на ноль всегда приводит к переполнению. В противоположность результату деления, результат умножения правилен, несмотря на значение V-флага.

Так как логические операции в АЛУ не могут нести неправильный результат, для этих операций V-флаг устанавливается в «0».

V-флаг также используется как «приклеивающийся бит» для операций правого циклического сдвига и правого сдвига. Только с использованием C-флага, ошибка сдвига вызываемая командой правого сдвига, может быть оценена до величины результата половины LSB. Вместе с V-флагом, C-флаг позволяет оценивать ошибку сдвига с лучшим разложением (см ниже таблицу).

Для Boolean битовых операций с только одним операндом V-флаг всегда установлен в «0». Для Boolean битовых операций с двумя операндами V-флаг выставляет результат операции OR с двумя битами.

Оценка ошибки вращения правого сдвига

С-Флаг	V-флаг	Величина ошибки вращения
0	0	Нет ошибки вращения
0	1	0< Ошибка вращения <1/2 LSB
1	0	Ошибка вращения =1/2 LSB
1	1	Ошибка вращения >1/2 LSB

- **Z-флаг:** Z-флаг установлен в «1», если результатом операции в АЛУ является нулевая величина.

Z-флаг всегда устанавливается в «1» для сложения и вычитания with carry в том случае, когда одновременно Z-флаг уже содержит «1» и результат текущей операции в АЛУ тоже равен нулю. Этот механизм обеспечивает поддержку вычислений с большой точностью.

Для Boolean битовых операций только с одним операндом в Z-флаг помещается логическое отрицание изменяемого бита. Для Boolean битовых операций с двумя операндами в Z-флаг помещается результат логической операции NOR над двумя отмеченными битами.

- **Е-флаг:** Е-флаг может быть изменен командой, совершающей операцию в АЛУ или совершающей перемещение данных. Е-флаг устанавливает «0» для тех операндов, которые не могут быть использованы при табличном поиске. Во всех других случаях Е-флаг устанавливается в зависимости от значения исходного операнда, иными словами по достижении конца поиска в таблице. Если значение исходного операнда команды равно минимальному отрицательному числу (8000_H для слов данных или 80_H для байтов данных), Е-флаг устанавливается в «1».

- **MULIP-флаг:** MULIP-флаг аппаратно устанавливается в «1», при входе в подпрограмму обслуживания прерывания, в случае прерывания операции умножения или деления в АЛУ. В зависимости от состояния этого бита, микроконтроллер решает, продолжать или не продолжать умножение или деление после завершения обслуживания прерывания. Значение бита MULIP перезаписывается из стека, при выполнении команды возврата из прерывания (RETI). Обычно это означает, что MULIP-флаг снова после этого принимает нулевое значение.

Примечание: MULIP-флаг – часть окружения задачи! Когда подпрограмма обслуживания прерывания не осуществляет возврата в прерванную команду умножения и деления (т.е. в случае использования таблицы задач, переключающей между независимыми задачами), MULIP-флаг должен быть сохранен как часть окружения задачи, и таким образом должна осуществляться перезапись для новой задачи.

Флаги состояния прерывания ЦПУ (IEN, ILVL)

Бит разрешения прерываний глобально разрешает (IEN=1) или запрещает (IEN=0) прерывания. Четырехбитное поле уровней прерываний (ILVL) отмечает уровень приоритета текущего состояния ЦПУ. Уровень прерывания ЦПУ изменяется аппаратно при входе в подпрограмму обслуживания прерывания. Также для запрещения прерываний ILVL может быть программно изменен. В случае присвоения ЦПУ максимально возможного 15-го уровня, текущая деятельность ЦПУ не может быть прервана, за исключением аппаратных ловушек и внешних немаскируемых прерываний. Подробности рассматриваются в разделе «Функции прерываний и ловушек».

После RESET все прерывания запрещены, и ЦПУ присвоен самый низкий уровень приоритета (ILVL=0).

Указатель команд (IP)

Этот регистр определяет внутренний 16-битный сегментный адрес текущей команды, при этом сегмент данных выбирается при помощи регистра CSP. IP регистр размещен вне адресного пространства С167, и поэтому не доступен для программиста. Однако IP можно изменить косвенным путем с помощью стека, посредством команды возврата.

Значение IP-регистра изменяется при выполнении центральным процессором команд перехода, а также инкрементируется после вызова очередной команды.

Значение после RESET: 0000_H

Бит	Функция
IP (0 – 15 биты)	Устанавливает внутри сегментное смещение адреса, для вызова текущий команды. IP отправляет к текущему сегменту <SEGNR>

Указатель сегмента кода (CSP)

Этот не адресуемый побитно регистр используется для указания сегмента кода, используемого для доступа к командам. Нижние 8 бит регистра CSP предназначены для выбора одного из 256 сегментов, верхние 8 бит зарезервированы для использования в будущих моделях.

CSP (FE08/04_H)

Значение после RESET 0000_H

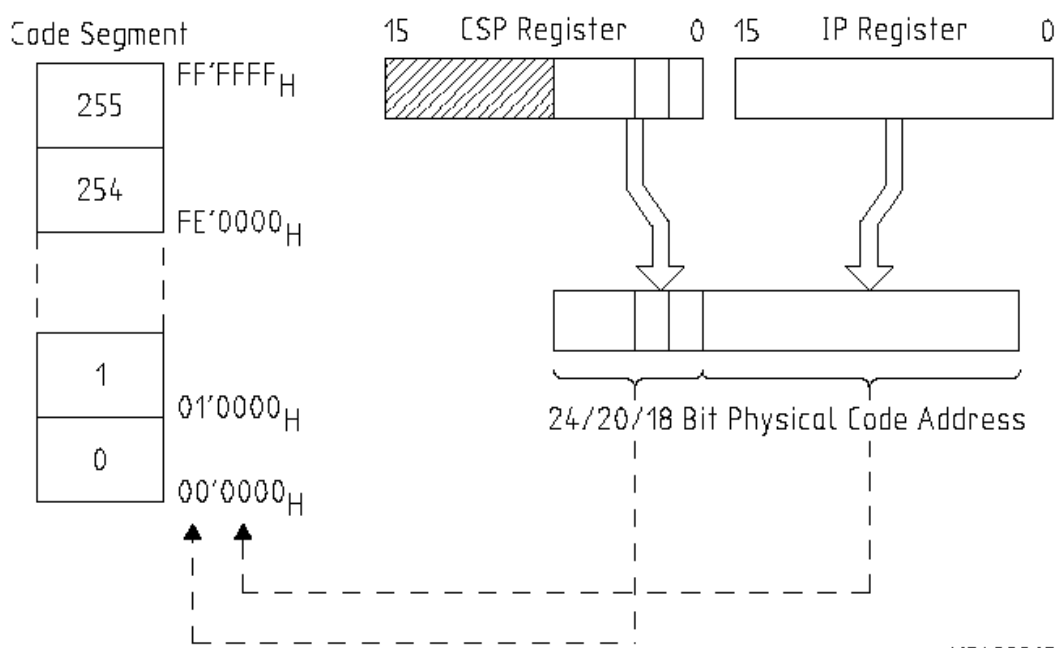
Бит	Функция
SEGNR (0 – 7 биты)	Номер сегмента Устанавливает сегмент кода, из которого вызываются текущие команды. SEGNR игнорируется, при отключенной сегментации.

Адреса ячейки кода создается путем прямого расширения 16-битного содержимого IP-регистра содержимым CSP-регистра, как показано на рисунке ниже.

В случае использования для доступа к внешнему коду режима сегментированной памяти, выбранное число битов адреса сегмента (7... 0, 3... 0 или 1... 0) из регистра CSP поступает на выводы Порты 4 для адреса сегмента A23/A19/A17 ... A16. В режиме несегментированной памяти или в режиме Single Chip, содержимое этого регистра не имеет значения, потому что все возможные обращения к коду автоматически ограничиваются сегментом 0.

Примечание: Значение CSP-регистра может быть только считано, и его значение нельзя изменить посредством прямой записи. Однако его можно изменить либо посредством команд JMPS и CALLS, либо посредством команд RETS и RETI.

В момент разрешения на вход в подпрограмму прерывания или при выполнении команды TRAP, значение CSP-регистра автоматически устанавливается в ноль.



MCA02265

Рисунок 4-5

Адресация с помощью указателя сегмента кода

Примечание: При отключенной сегментации, значение IP используется напрямую, как 16-битный адрес.

Указатели страниц данных DPP0, DPP1, DPP2, DPP3

Эти четыре не адресуемых побитно регистра используются для указания активных страниц. Нижние 10 бит каждого DPP-регистра выбирают одну из 1024 возможных 16 Кбайтных страниц данных, в то время как верхние 6 бит не используются и зарезервированы на будущее. DPP-регистры позволяют получить доступ к полному пространству памяти, разбитому по 16 Кбайтным страницам.

DPP-регистры используются всякий раз при совершении доступа к любому адресу памяти в режиме косвенной или прямой 16-битной адресации (исключая override доступ, доступ EXTended команд и доступ PEC-передачи данных). После RESET, указатель страницы данных инициализируется таким образом, что все результаты косвенной или прямой 16-битной адресации являются идентичными 18-битной адресации. Это позволяет получить доступ к страницам данных 0... 3 в сегменте 0, как показано ниже на рисунке. Если пользователь не хочет использовать какое либо деление на страницы, то не требуется предпринимать никаких других действий.

DPP0 (FE00H/00H)

Значение после RESET 0000H

DPP1 (FE02H/01H)

Значение после RESET 0001H

DPP2 (FE04H/02H)

Значение после RESET 0002H

DPP3 (FE06H/03H)

Значение после RESET 0003H

Бит	Функция
DPPхPN (0 – 9 биты)	Номер страницы данных Устанавливает выбранную страницу данных через DPPх. Только два последних бита DPPх имеют значение при отключенной сегментации.

Использование страниц данных обеспечивается путем сцепления нижних 14 битов косвенного или прямого 16-битного адреса с содержимым DPP-регистра, поставленным на место двух верхних битов. Содержимое выбранного DPP-регистра указывает на одну из 1024 возможных страниц данных. Этот базовый адрес страницы данных вместе с 14-битным сдвигом страницы формирует физический 24/20/18-битный адрес.

В случае использования режима несегментированной памяти, только два самых младших бита DPP-регистра используются для создания физического адреса. Таким образом необходимо соблюдать крайнюю осторожность, при изменении содержимого DPP-регистра, в случае использования несегментированной модели памяти, иначе возможны непредсказуемые результаты. В случае использования режима сегментированной памяти при доступе к внешним данным, выбранное число битов адреса сегмента (9 ... 2, 5... 2 или 3... 2), из необходимого DPP-регистра, выставляется на выводы адреса сегмента A23/A19/A17... A16 Porta 4.

DPP-регистр может быть изменен с помощью любой команды, которая имеет возможность изменять SFR-регистры.

Примечание: Сразу после изменения значения DPP-регистра, новое значение нельзя использовать для вычисления адреса операнда, что обусловлено внутренним конвейером команд (необходимо пропустить один машинный такт).

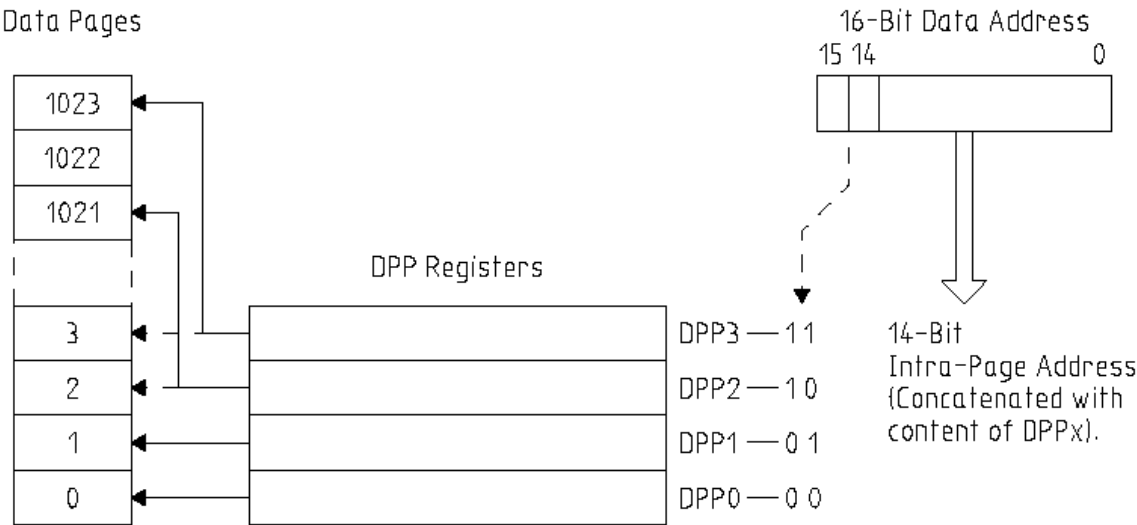


Рисунок 4-6

Адресация с помощью указателя страницы данных DPPx

После RESET или при отключенной сегментации, DPP регистры используют страницы данных 3...0.

Context Pointer (CP)

Этот не адресуемый побитно регистр используется для выбора текущего банка регистров основного назначения (GPR).

SFR

CP (FE10_H/08_H)

Значение после RESET: FC00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	Cp											0
r	r	r	r	Rw											r

Бит	Функция
CP	Изменение группы регистров CP Устанавливает адрес базового слова текущего банка регистров. При чтении значения регистра CP с битами CP.11 ... CP.9 = «000», CP.11 ... CP.10 устанавливаются аппаратно в «11», во всех других случаях все биты битового поля «cp» отдадут считываемые значения.

Примечание: Пользователь сам устанавливает физические адреса GPR-регистров, с помощью записи значений в CP-регистр, при этом адреса должны быть в области внутренней памяти. При невыполнении этих условий возможны ошибочные результаты.

- Не устанавливать значение СР-регистра менее 00`F600_H и более 00`FDFF_H
- Быть внимательным при использовании старших GPR-регистров при значении СР более 00`FDE0_H

Значение СР-регистра может быть изменено с помощью любой команды, которая может изменять значения SFR-регистров.

Примечание: Новое значение СР-регистра не может быть использовано для вычисления адреса GPR-регистра сразу после изменения значения СР-регистра, что связано с внутренним конвейером команд.

Ключевые команды SCXT позволяют сохранить содержимое СР-регистра в стеке и записать в него новое значение за один машинный такт.

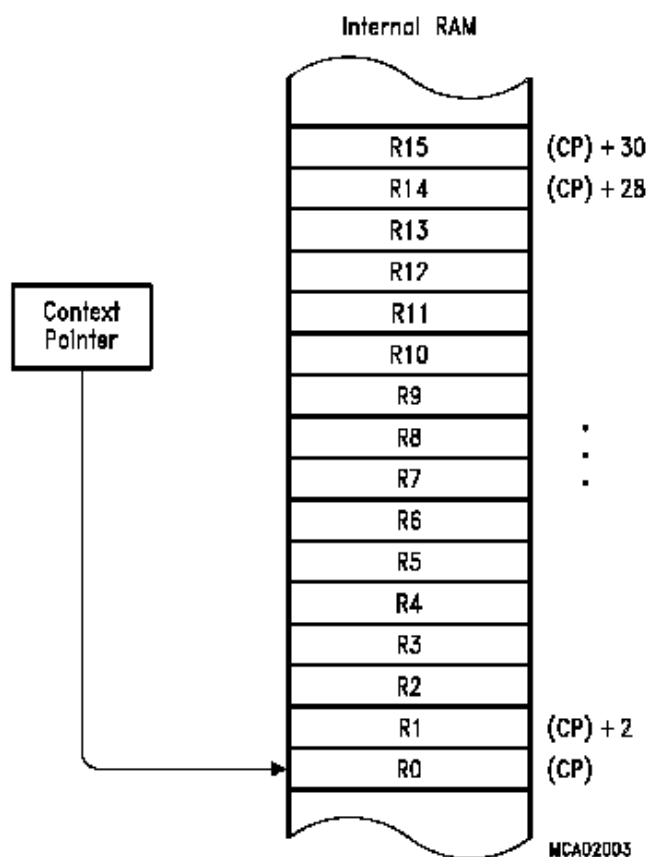


Рисунок 4-7

Выбор банка регистров с помощью СР-регистра

Некоторые режимы адресации используют СР-регистр для вычисления адреса.

Короткая 4-битная адресация GPR-регистров (обозначение: R_w или R_b) позволяет использовать относительные адреса памяти, привязанные к содержимому СР-регистра, т.е. к базовому адресу текущего банка регистров. В зависимости от использования данного режима для относительного доступа к слову (R_w) или байту (R_b), короткий 4-битный

адрес перед сложением с содержимым регистра CP может быть умножен на два для доступа к словам или может быть оставлен без изменений.

GPR-регистры, используемые в качестве указателя косвенного адреса, всегда доступны пословно. Для некоторых команд только первые четыре GPR-регистра могут использоваться в качестве указателя косвенного адреса. Эти GPR-регистры доступны с помощью короткой 2-битной адресации. Вычисление физических адресов идентично короткой 4-битной GPR-адресации.

Короткая 8-битная адресация регистров (обозначение: reg или bitoff) интерпретирует младшие 4 значащих бита в диапазоне от F0_H до FF_H, как короткие 4-битные адреса GPR-регистров, в то время как четыре старших значащих бита игнорируются. Вычисление представленного физического адреса GPR-регистра идентично вычислению короткого 4-битного адреса GPR-регистра. Для доступа к одиночным битам в GPR-регистрах, слово адреса GPR-регистра вычисляется, как описано выше, но позиция бита в слове определяется с помощью независимого дополнительного 4-битного значения.

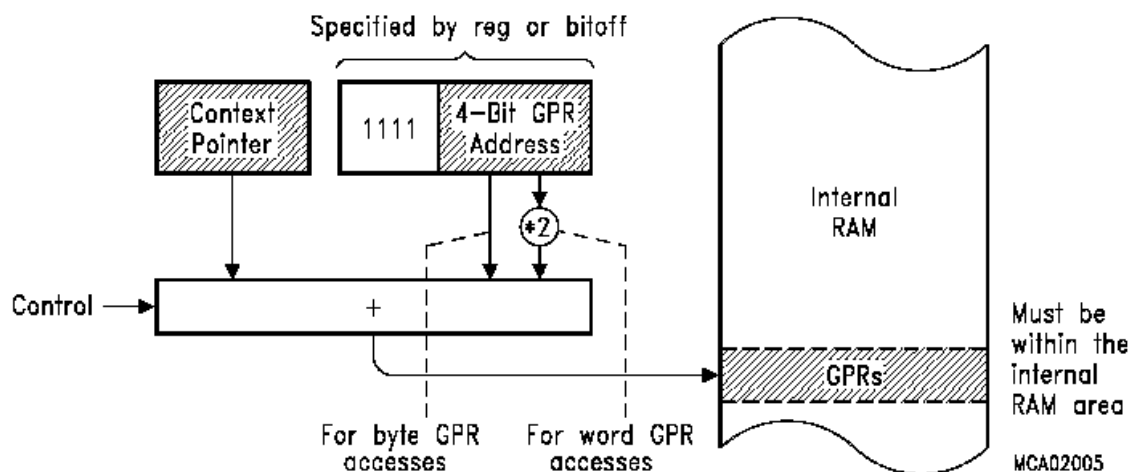


Рисунок 4-8

Неявное использование CP для режима короткой адресации GPR

Указатель стека (SP)

Этот не адресуемый побитно регистр используется для указания адреса вершины внутреннего системного стека (TOS). Значение SP-регистра декрементируется, как только в стек посылаются данные, и инкрементируется после возврата данных из стека. Таким образом системный стек растет от большего к меньшим значениям адресов памяти.

Наименьшему значащему биту в SP-регистре всегда присваивается «0», и битам 15-12 всегда аппаратно присваивается «1», поэтому SP-регистр может принимать значения от F000_H до FFFE_H. Это позволяет получить

доступ физическому стеку в внутренней области RAM C167. Виртуальный стек (обычно больший, чем физический) может быть реализован программным способом. Этот механизм поддерживается регистрами STKOV и STKUN (см. ниже описание).

Значение SP-регистра может быть изменено любой командой, поддерживающей возможность изменения содержимого SFR-регистров.

Примечание: Команды POP и RETURN не должны следовать сразу за командой, изменяющей значение SP-регистра, что обусловлено внутренней структурой конвейера команд.

SFR

SP (FE12_H/09_H)

Значение после RESET: FC00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	sp											0
r	r	r	r	rw											r

Бит	Функция
sp	Модифицируемая часть регистра SP Указывает значение вершины внутреннего системного стека

Указатель переполнения стека STKOV

Значение этого побитно не адресуемого регистра сравнивается со значением SP-регистра после каждой операции, наполняющей данными системный стек (команды PUSH и CALL или прерывания), и после каждого вычитания из значения SP-регистра. Если содержимое SP-регистра менее содержимого STKOV-регистра, то имеет место аппаратная ловушка переполнения стека.

Так как наименьший значащий бит в STKOV-регистре всегда равен нулю а также битам 15-12 всегда аппаратно присваивается «1», то STKOV-регистр может содержать значения от F000_H до FFFE_H.

SFR															
STKOV (FE14 _H /0A _H)								Значение после RESET: FA00 _H							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	stkov											0
r	r	r	r	rw											r
Бит				Функция											
stkov				Модифицируемая часть регистра STKOV Обозначает нижний предел внутреннего системного стека											

Ловушка переполнения стека (вводится в действие, когда $(SP) < (STKOV)$) может быть использована двумя различными способами.

- **Указание на фатальную ошибку** обрабатывает переполнение стека как системную ошибку с помощью, специальной подпрограммы обслуживания ловушки. В этом случае данные на дне стека могут быть перезаписаны посредством сохраненной в стеке информации о статусе, во время обслуживания ловушки переполнения стека.

- **Автоматический flushing системного стека** позволяет использовать системный стек, как «Стековый кэш» для создания большего внешнего пользовательского стека. В этом случае STKOV-регистр должен быть установлен на значение, на 12 ячеек больше, чем необходимо для вершины стека. При этом, наихудшим случаем, который может случиться, является обнаружение состояния переполнения стека в момент входа в обслуживание прерывания. В этом случае требуется шесть дополнительных ячеек стека для сохранения значений IP, PSW, CSP как при обслуживании прерывания, так и при обслуживании аппаратной ловушки.

Более подробно о порядке обслуживания ловушки переполнения стека и об управлении виртуальным стеком описано в разделе: «Системное программирование».

Указатель опустошения стека STKUN

Значение этого не адресуемого побитно регистра сравнивается с SP-регистром после каждой операции, запрашивающей данные из системного стека (при помощи команд POP и RET), и после каждого увеличения значения SP-регистра. Если содержимое SP-регистра больше, чем содержимое STKUN-регистра, то будет иметь место аппаратная ловушка.

Так как наименьший значащий бит в STKUN-регистре всегда равен нулю, а также битам 15-12 всегда аппаратно присваивается «1», то STKUN-регистр может содержать значения от F000_H до FFFE_H.

SFR

STKUN (FE16_H/0B_H)

Значение после RESET: FC00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	stkun											0
r	r	r	r	rw											r

Бит	Функция
stkun	Модифицируемая часть регистра STKUN Определяет верхний предел внутреннего системного стека

Ловушка опустошения стека (задействуется при условии: (SP)>(STKUN)) может быть использована двумя способами.

- **Указание на фатальную ошибку** обрабатывает опустошение стека как системную ошибку с помощью подпрограммы обслуживания ловушки.

- **Автоматический refilling системного стека** позволяет использовать системный стек как «Кеш стека» для получения большего значения объема пользовательского стека. В этом случае, в STKUN-регистр устанавливается значение, которое представляет собой высшее значение адреса основания стека.

Более подробно обслуживание ловушки по опустошению стека и управление виртуальным стеком рассмотрено в разделе «Системное программирование».

Сфера действий управления пределом стека

Управление границами стека реализовано с помощью регистров STKOV и STKUN, определяющих случаи выхода указателя стека SP за пределы очерченной области стека. Переполнение и опустошение стека имеет место при использовании команд ADD или SUB, либо при использовании операций PUSH или POP (явных и неявных, т.е. CALL или RET команды).

Механизм обнаружения ловушек не срабатывает, т.е. ловушки стека не создаются заново когда:

- Значение указателя стека изменяется прямо с помощью команд MOV
- Изменяются пределы области стека (STKOV, STKUN), и поэтому SP находится вне новых границ

Старший регистр умножения/деления MDH

Этот регистр является частью 32-битного регистра умножения/деления, используемого ЦПУ при совершении операций

умножения и деления. После умножения, этот не адресуемый побитно регистр содержит старшие 16 битов 32-битного результата. Для длинного деления, в MDH-регистр должны быть загружены старшие 16 битов 32-битного делимого до начала операции деления. После любого деления в MDH-регистре содержится 16-битный остаток.

SFR															
MDH (FE0C _H /06 _H)								Значение после RESET: 0000 _H							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Mdh															
Rw															

Бит	Функция
mdh	Содержит старшие 16-битов 32-битного регистра умножения и деления MD

Как только программно изменяется значение этого регистра, устанавливается «1» в флаге MDRIU, регистра управления умножением и делением.

В случае прерывания выполнения операции умножения и деления, и в том случае когда в подпрограмме прерывания совершается умножение и деление, значения регистров MDH, MDL и MDC должны быть сохранены во избежание ошибочных результатов.

Подробное описание использования MDH-регистр для программирования алгоритмов умножения и деления дано в разделе «Системное программирование».

Младший регистр умножения/деления MDL.

Этот регистр является частью 32-битного регистра умножения/деления, используемого ЦПУ для совершения умножения или деления. После умножения, этот не адресуемый побитно регистр содержит младшие 16-битов 32-битного результата. Прежде чем начать операцию длинного деления, в MDL-регистр необходимо загрузить младшие 16 битов 32-битного делимого. После любого деления, регистр MDL содержит 16-битное частное.

MDL (FE0E_H/07_H)

SFR

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Mdl															
Rw															

Бит	Функция
Mdl	Содержит младшие 16-битов 32-битного регистра умножения и деления MD

Как только программно изменяется значение этого регистра, в флаге MDRIU устанавливается «1». В случае программного чтения MDL-регистра, флаг MDRIU устанавливается в «0».

В случае прерывания исполнения операции умножения и деления, в том случае когда в подпрограмме прерывания совершается умножение и деление, регистры MDH, MDL и MDC должны быть сохранены во избежание ошибочных результатов.

Подробное описание о том, как использовать MDL-регистр для программирования алгоритмов умножения и деления, дано в разделе «Системное программирование».

Контрольный регистр умножения/деления MDC

Этот адресуемый побитно 16-битный регистр используется ЦПУ во время совершения умножения или деления. Он используется для сохранения требуемой информации для управления операциями умножения и деления. Значение регистра MDC изменяется аппаратно, в течении каждого такта команды умножения и деления.

SFR															
MDL (FF0E _H /87 _H)								Значение после RESET: 0000 _H							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	!!	!!	!!	MDR IU	!!	!!	!!	!!
-	-	-	-	-	-	-	-	r(w)	r(w)	r(w)	r(w)	r(w)	r(w)	r(w)	r(w)
Бит		Функция													
MDRIU		Флаг использования регистра умножения/деления «0»: устанавливается при чтении регистра MDL «1»: устанавливается при записи в регистр MDL или MDH, или когда выполняется операция деления.													
!!		Внутреннее состояние АЛУ Модуль умножения/деления использует эти биты для внутренних управляющих операций Никогда не изменять значений этих битов без сохранения и восстановления значения регистра MDC													

Если в момент выполнения операции умножения или деления было совершено прерывание до окончания вычисления, и если модуль умножения и деления требуется в подпрограмме прерывания, то необходимо сохранить значения регистров MDC, MDH и MDL (для того чтобы была возможность возобновить прерванную операцию) и затем очистить MDC-регистр для того, чтобы подготовить его к новому вычислению.

После завершения нового умножения или деления, необходимо восстановить состояние регистров.

MDRIU-флаг является той частью MDC-регистра, которая представляет интерес для пользователя. Остальная часть MDC-регистра зарезервирована для использования микроконтроллером, и пользователям не следует изменять ее значение, в других случаях, отличных от описанных выше. Иначе, нельзя гарантировать корректного продолжения прерванных операций умножения или деления.

Подробное описание использования MDC-регистра для программирования алгоритмов умножения и деления, дано в разделе «Системное программирование».

Регистр постоянной нуля ZEROS

Все биты этого адресуемого побитно регистра аппаратно зафиксированы на нуле. Этот регистр можно только читать. ZEROS-регистр может быть использован для адресуемой как к регистру константы нуля, т.е. для использования в операциях с битами или создания масок. Регистр может быть доступен с помощью любой команды, которая может адресоваться к регистру SFR.

ZEROS(FF1C/8E_H)

Значение после RESET: 0000_H

Регистр постоянной единицы ONES

Все биты этого адресуемого побитно регистра аппаратно зафиксированы на единице. Этот регистр можно только читать. ONES-регистр может быть использован для адресуемой как к регистру константы единицы, т.е. для использования в операциях с битами или создания масок. Регистр может быть доступен с помощью любой команды, которая может быть адресована к регистру SFR.

ONES(FF1E/8F_H)

Значение после RESET: FFFF_H

5 Система прерываний и ловушек

Архитектура C167 поддерживает несколько способов быстрого и гибкого обслуживания запросов, создаваемых различными как внутренними так и внешними источниками.

Эти механизмы включают:

Нормальная обработка прерываний

ЦПУ временно задерживает выполнение текущей программы и переходит к выполнению подпрограммы, обслуживающей прерывание, для того чтобы обслужить устройство, пославшее запрос на прерывание. Текущее состояние программы (IP, PSW и также CSP в режиме сегментированной памяти) сохраняется во внутреннем системном стеке. !6-уровневая схема приоритетов позволяет пользователю определить, который из текущих запросов на прерываний должен быть обслужен.

Работа с прерываниями при помощи контроллера периферийных событий (PEC)

Обслуживание запросов на прерывание от устройств с помощью интегрированного контроллера периферийных событий (PEC) обеспечивает более быструю альтернативу нормальному программному выполнению прерываний. Переходя на запрос на прерывание, PEC совершает передачу одного слова или байта между двумя точками в сегменте 0 (страницы данных 0... 3) с помощью одного из восьми программируемых PEC-каналов обслуживания. Во время PEC-передачи данных, нормальное выполнение программы приостанавливается на один командный такт. Нет необходимости в сохранении внутреннего состояния программы. Для PEC-обслуживания используется та же самая схема уровней приоритетов, как и для нормального обслуживания прерываний. PEC-передачи разделяют два высших уровня приоритетов.

Функции ловушек

Функции ловушек активируются в ответ на специальные состояния, которые могут возникнуть во время выполнения команд. Ловушка также может быть вызвана внешним воздействием, при помощи не маскируемого вывода прерываний \overline{NMI} . Некоторые функции аппаратных ловушек созданы для выявления ошибочных состояний и исключений, возникающих во время выполнения команд. Аппаратные ловушки имеют высочайший приоритет и вызывают немедленную реакцию системы. Выполнение программных ловушек вызывается командой TRAP, которая вырабатывает программное

прерывание по собственному вектору. Текущее состояние системы для всех типов ловушек сохраняется в системном стеке.

Обработка внешних прерываний

Несмотря на то, что C167 не обеспечен специально предназначенными выводами для прерываний, микроконтроллер может быть соединен с внешними источниками прерываний и может обеспечивать несколько механизмов реакции на внешние события, включая использование стандартного выполнения, не маскируемых прерываний, и быстрых внешних прерываний. Эти функции, за исключением не маскируемых прерываний и входа RESET, представляют собой альтернативные функции внешних портов.

5.1 Структура системы прерываний

C167 предлагает 56 независимых каналов прерываний, для которых можно определить 16 уровней приоритетов. Для того, чтобы обеспечить возможность модульной и совместимой разработки программ, каждый источник прерывания или РЕС-запрос обеспечивается независимым контрольным регистром и вектором прерывания. Контрольный регистр содержит флаг запроса на прерывание, бит разрешения прерывания и уровень приоритета прерывания объединенных источников. Каждый запрос на прерывание вызывается соответствующим событием, зависящим от выбранного режима работы оборудования. Исключениями являются только два последовательных канала микроконтроллера C167, в которых может генерироваться один запрос на прерывание от нескольких типов ошибок. Тем не менее, соответствующие флаги состояния, которые идентифицируют тип ошибки, добавлены в управляющие регистры последовательных каналов.

C167 содержит векторную систему прерываний. В этой системе для векторов зарезервированы соответствующие адреса в пространстве памяти для обслуживания прерываний, ловушек и RESET-сигнала. Как только получен запрос, ЦПУ совершает переход по вектору, связанному с соответствующим источником прерывания. Это позволяет напрямую идентифицировать источники, совершивших запрос. Исключением является только аппаратные ловушки класса В, которые все определены через один вектор прерывания. Флаги состояний в регистре флагов ловушек (TFR) могут быть использованы для определения исключения, которое вызвало ловушку. Для специальной программной команды TRAP, адрес вектора соответствует полю операнда команды. Это поле представляет из себя 7-битный номер ловушки.

Зарезервированная область векторов в самом начале адресного пространства C167 (сегмент 0) определяет таблицу переходов. Согласно

таблице векторов, совершается соответствующая команда перехода, которая передает управление подпрограмме обслуживания прерывания, которая может быть расположена в любом месте адресного пространства. Начало таблицы переходов расположено в наименьшем адресе кодового сегмента 0. Область каждого вектора содержит 2 слова, за исключением RESET-вектора и вектора аппаратных ловушек, занимающих соответственно по 4 или 8 слов.

Таблица, приведенная ниже, содержит все источники, которые могут дать запросы на прерывание или РЕС-обслуживание в микроконтроллере C167. Также в этой таблице приведены все соответствующие вектора, их расположение и номера ловушек.

Примечание: Не используемые в настоящее время, ячейки в таблице (X-периферия) предназначены для внутренней XBUS-периферии. Таким образом, эти строки, могут быть использованы для создания программно управляемых запросов на прерывание посредством установки необходимого бита XPnIR.

Источники прерываний или запросы на обслуживание PEC	Флаг запроса	Флаг разрешения	Вектор прерывания	Адрес вектора	Номер ловушки
CAPCOM-регистр 0	CC0IR	CC0IE	CC0INT	00`0040 _H	10 _H /16 _D
CAPCOM-регистр 1	CC1IR	CC1IE	CC1INT	00`0044 _H	11 _H /17 _D
CAPCOM-регистр 2	CC2IR	CC2IE	CC2INT	00`0048 _H	12 _H /18 _D
CAPCOM-регистр 3	CC3IR	CC3IE	CC3INT	00`004C _H	13 _H /19 _D
CAPCOM-регистр 4	CC4IR	CC4IE	CC4INT	00`0050 _H	14 _H /20 _D
CAPCOM-регистр 5	CC5IR	CC5IE	CC5INT	00`0054 _H	15 _H /21 _D
CAPCOM-регистр 6	CC6IR	CC6IE	CC6INT	00`0058 _H	16 _H /22 _D
CAPCOM-регистр 7	CC7IR	CC7IE	CC7INT	00`005C _H	17 _H /23 _D
CAPCOM-регистр 8	CC8IR	CC8IE	CC8INT	00`0060 _H	18 _H /24 _D
CAPCOM-регистр 9	CC9IR	CC9IE	CC9INT	00`0064 _H	19 _H /25 _D
CAPCOM-регистр 10	CC10IR	CC10IE	CC10INT	00`0068 _H	1A _H /26 _D
CAPCOM-регистр 11	CC11IR	CC11IE	CC11INT	00`006C _H	1B _H /27 _D
CAPCOM-регистр 12	CC12IR	CC12IE	CC12INT	00`0070 _H	1C _H /28 _D
CAPCOM-регистр 13	CC13IR	CC13IE	CC13INT	00`0074 _H	1D _H /29 _D
CAPCOM-регистр 14	CC14IR	CC14IE	CC14INT	00`0078 _H	1E _H /30 _D
CAPCOM-регистр 15	CC15IR	CC15IE	CC15INT	00`007C _H	1F _H /31 _D
CAPCOM-регистр 16	CC16IR	CC16IE	CC16INT	00`00C0 _H	30 _H /48 _D
CAPCOM-регистр 17	CC17IR	CC17IE	CC17INT	00`00C4 _H	31 _H /49 _D
CAPCOM-регистр 18	CC18IR	CC18IE	CC18INT	00`00C8 _H	32 _H /50 _D
CAPCOM-регистр 19	CC19IR	CC19IE	CC19INT	00`00CC _H	33 _H /51 _D
CAPCOM-регистр 20	CC20IR	CC20IE	CC20INT	00`00D0 _H	34 _H /52 _D
CAPCOM-регистр 21	CC21IR	CC21IE	CC21INT	00`00D4 _H	35 _H /53 _D
CAPCOM-регистр 22	CC22IR	CC22IE	CC22INT	00`00D8 _H	36 _H /54 _D
CAPCOM-регистр 23	CC23IR	CC23IE	CC23INT	00`00DC _H	37 _H /55 _D
CAPCOM-регистр 24	CC24IR	CC24IE	CC24INT	00`00E0 _H	38 _H /56 _D
CAPCOM-регистр 25	CC25IR	CC25IE	CC25INT	00`00E4 _H	39 _H /57 _D
CAPCOM-регистр 26	CC26IR	CC26IE	CC26INT	00`00E8 _H	3A _H /58 _D
CAPCOM-регистр 27	CC27IR	CC27IE	CC27INT	00`00EC _H	3B _H /59 _D
CAPCOM-регистр 28	CC28IR	CC28IE	CC28INT	00`00F0 _H	3C _H /60 _D
CAPCOM-регистр 29	CC29IR	CC29IE	CC29INT	00`0110 _H	44 _H /68 _D
CAPCOM-регистр 30	CC30IR	CC30IE	CC30INT	00`0114 _H	45 _H /69 _D
CAPCOM-регистр 31	CC31IR	CC31IE	CC31INT	00`0118 _H	46 _H /70 _D

Источники прерываний или запросы на обслуживание PEC	Флаг запроса	Флаг разрешения	Вектор прерывания	Адрес вектора	Номер ловушки
CAPCOM-таймер 0	T0IR	T0IE	T0INT	00`0080 _H	20 _H /32 _D
CAPCOM-таймер 1	T1IR	T1IE	T1INT	00`0084 _H	21 _H /33 _D
CAPCOM-таймер 7	T7IR	T7IE	T7INT	00`00F4 _H	3D _H /61 _D
CAPCOM-таймер 8	T8IR	T8IE	T8INT	00`00F8 _H	3E _H /62 _D
GPT1-таймер 2	T2IR	T2IE	T2INT	00`0088 _H	22 _H /34 _D
GPT1-таймер 3	T3IR	T3IE	T3INT	00`008C _H	23 _H /35 _D
GPT1-таймер 4	T4IR	T4IE	T4INT	00`0090 _H	24 _H /36 _D
GPT2-таймер 5	T5IR	T5IE	T5INT	00`0094 _H	25 _H /37 _D
GPT2-таймер 6	T6IR	T6IE	T6INT	00`0098 _H	26 _H /38 _D
GPT2 CAPREL регистр	CRIR	CRIE	CRINT	00`009C _H	27 _H /39 _D
Готовность результата АЦП	ADCIR	ADCIE	ADCINT	00`00A0 _H	28 _H /40 _D
Overflow ошибка АЦП	ADEIR	ADEIE	ADEINT	00`00A4 _H	29 _H /41 _D
ASC0 передача	S0TIR	S0TIE	S0TINT	00`00A8 _H	2A _H /42 _D
ASC0 буфер передачи	S0TBIR	S0TBIE	S0TBINT	00`011C _H	47 _H /71 _D
ASC0 получение	S0RIR	S0RIE	S0RINT	00`00AC _H	2B _H /43 _D
ASC0 ошибка	S0EIR	S0EIE	S0TENT	00`00B0 _H	2C _H /44 _D
SSC0 передача	SSCTIR	SSCTIE	SSCTINT	00`00B4 _H	2D _H /45 _D
SSC0 получение	SSCRIR	SSCRIE	SSCRINT	00`00B8 _H	2E _H /46 _D
SSC0 ошибка	SSCEIR	SSCEIE	SSCTEINT	00`00BC _H	2F _H /47 _D
PWM канал 0...3	PWMIR	PWMIE	PWMINT	00`00FC _H	3F _H /63 _D
CAN интерфейс	XP0IR	XP0IE	XP0INT	00`0100 _H	40 _H /64 _D
Вывод X-периферии	XP1IR	XP1IE	XP1INT	00`0104 _H	41 _H /65 _D
Вывод X-периферии	XP2IR	XP2IE	XP2INT	00`0108 _H	42 _H /66 _D
PLL разъединение	XP3IR	XP3IE	XP3INT	00`010C _H	43 _H /67 _D

Примечание: Каждая точка таблицы векторов состоит из области для двух команд из одного слова или одной двухсловной команды.

Для моделей микроконтроллеров, не содержащих CAN-модуль или PLL, соответствующие входы прерываний могут быть использованы для программных прерываний. (см X-периферийные N-выводы).

Таблица, приведенная ниже, содержит адреса векторов для аппаратных ловушек и соответствующих им флагов состояния в регистре TFR. В таблице также перечисляются уровни приоритетов ловушек для тех случаев, когда может быть обнаружено больше одного состояния для ловушки во время выполнения одной команды. После любого RESET (аппаратный RESET, программный RESET при помощи команды SRST, или RESET при переполнении сторожевого таймера), выполнение программы начинается с адреса вектора RESET 00`0000_H. RESET имеет уровень

приоритета выше, чем любое рабочее состояние системы (III уровень приоритетов ловушек).

Аппаратные ловушки могут быть установлены на любой адрес вектора: от 00`0000_Н до 00`01FC_Н. Выполнение подпрограммы, заданной с помощью команды TRAP, всегда выполняется на текущем уровне приоритета ЦПУ, который можно прочитать в ILVL поле битов регистра PSW. Это означает, что подпрограмма, начатая с помощью команды TRAP, может быть прервана всеми аппаратными ловушками или запросами на прерывание более высокого уровня.

Исключенное состояние	Флаг ловушки	Вектор ловушки	Адрес вектора	Номер ловушки	Уровень приоритета ловушки
RESET функции: - аппаратный - программный - переполнение сторожевого таймера		RESET RESET RESET	00`0000 _H 00`0000 _H 00`0000 _H	00 _H 00 _H 00 _H	III III III
Аппаратные ловушки класса А - Не маскируемые прерывания - Переполнения стека - Underflow стека	NMI STKOV STKUF	NMITRAP STOTRAP STUTRAP	00`0008 _H 00`0010 _H 00`0018 _H	02 _H 04 _H 06 _H	II II II
Аппаратные ловушки класса В - Неизвестный код - Защищенная ошибка команды - Нелегальный доступ к словам операндов - Нелегальный командный доступ - Нелегальный доступ к внешней шине	UNDOPC PRTFLT ILLOPA ILLINA ILLBUS	BTRAP BTRAP BTRAP BTRAP BTRAP	00`0028 _H 00`0028 _H 00`0028 _H 00`0028 _H 00`0028 _H	0A _H 0A _H 0A _H 0A _H 0A _H	I I I I I
Зарезервированы			[2C _H -3C _H]	[0B _H -0F _H]	
Программные ловушки - TRAP-команды			Любой [00`0000 _H – 00`01FC _H] в 4 _H	любой [00 _H – 7F _H]	Текущий приоритет ЦПУ

Нормальная обработка прерываний и РЕС-обслуживание

В течении каждого командного такта, из всех возможных источников может быть выбран один запрос с максимальным уровнем приоритета, требующих выполнения прерывания или РЕС-обслуживания,. Каждому источнику запроса может быть назначен свой собственный уровень приоритета. Второй уровень (так же называемый «уровень приоритета группы») позволяет создавать внутренние приоритеты для равнозначных запросов из различных источников с одинаковыми уровнями приоритетов. В конце каждого командного цикла один из запросов с наивысшим текущим приоритетом направляется в систему прерываний. Этот запрос будет обслужен в том случае, если его приоритет выше, чем текущий приоритет ЦПУ в регистре PSW.

Описание регистра системы прерывания

В регистре PSW выполнение прерываний управляется с помощью бита общего разрешения прерываний (IEN) и поля битов уровня приоритета ЦПУ (ILVL). Дополнительно к этому, различные источники прерываний управляются отдельно при помощи специальных регистров управления прерываниями (...IC). Таким образом, разрешение ответа ЦПУ на запрос определяется как индивидуальным регистром управления прерыванием, так и PSW-регистром. РЕС-обслуживание управляется при помощи отдельного для каждого канала РЕССх-регистра и указателей точки источника и точки назначения.

Регистры управления прерываниями

Все регистры управления прерываниями организованы одинаково. Младшие 8 бит регистра управления прерываниями содержат полную информацию о статусе источника прерывания, необходимую для определения запроса с максимальным приоритетом. Старшие 8 бит регистра зарезервированы. Все регистры управления прерываниями являются побитно адресуемыми, и все биты могут прочитаны и записаны программно. Это позволяет программировать и изменять каждый из источников прерываний одной командой. Несмотря на полную доступность регистров управления прерываниями, при использовании команд, оперирующими словами данных, верхние 8 бит при чтении будут возвращать нулевое значение, а при записи будут отбрасываться.

Макет регистров управления прерываниями показан ниже для каждого ххIC-регистра, где хх обозначает номер источника.

<Область SFR>

MDL (уууу_H/zz_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								xxIR	xxIE	ILVL				GLVL	
-								rw	rw	rw				rw	

Бит	Функция
GLVL	Уровень группы Определяет внутренний порядок для запросов с одним уровнем приоритета
ILVL	Уровень приоритета прерывания Определяет уровень приоритета для определения запроса с максимальным приоритетом
XxIE	Управляющий бит разрешения прерывания (индивидуальное разрешение/запрещение данного источника) 0 – Запрос на прерывание запрещен 1 – Запрос на прерывание разрешен
XxIR	Флаг запроса на прерывание 0 – нет висящих запросов 1 – послан запрос на прерывание источником

Флаг запроса на прерывание устанавливается в «1», в тот момент когда будет послан запрос от источника. Этот флаг очищается аппаратно при входе в подпрограмму обслуживания прерывания или при входе в PEC-обслуживание. В случае PEC-обслуживания, этот флаг остается в «1» после передачи данных, если значение поля COUNT регистра PECSx, выбранного PEC-канала, равно нулю. Это позволяет нормальному прерыванию ЦПУ отреагировать на полную передачу данных в PEC-блоке.

Примечание: Программное изменение флага запроса на прерывание приведет к тому же эффекту, как если бы флаг устанавливался или очищался аппаратно.

Уровни приоритета прерываний и групповые уровни

Четыре бита поля ILVL определяют уровень приоритета обслуживания одновременно поданных запросов. Приоритет растет с увеличением числового значения ILVL, и поэтому 0000_B – самый низкий уровень, а 1111_B – самый высокий уровень приоритета.

В том случае, если в один и тот же момент поступает более одного запроса на прерывание с одинаковым уровнем, то для определения более

приоритетного запроса используется значение поля GLVL, задающего приоритет 2-ого уровня. Уровень внутри группы тоже растет с ростом числового значения GLVL, и поэтому 00_B – самый низкий, а 11_B – самый высокий уровень.

Примечание: Все источники запросов на прерывание, которые разрешены и запрограммированы на одинаковый уровень приоритета, должны быть всегда запрограммированы на различные групповые приоритеты. Иначе может быть сгенерирован некорректный вектор прерывания.

При входе в подпрограмму обслуживания прерывания, уровень приоритета победившего источника, в том случае если его уровень выше чем текущий уровень приоритета ЦПУ, копируется в поле ILVL регистра PSW после сохранения старого значения PSW в стеке.

Система прерываний C167 позволяет вставлять друг в друга до 15 подпрограмм, обслуживающих прерывания, с различными уровнями приоритетов (уровень 0 не может выявляться).

Запросы на прерывание, которые запрограммированы на 14 или 15 уровень (т.е. ILVL = 111X_B), будут обслуживаться при помощи PEC, до тех пор пока в поле COUNT регистра PECC не окажется нулевое значение. В этом случае вместо того, чтобы совершить PEC-обслуживание, запрос вызовет нормальное обслуживание прерывания. Запрос на прерывание, запрограммированный на 13 – 1 уровень приоритета, всегда будет обслуживаться с помощью нормального выполнения прерываний.

Примечание: Уровень 0000_B является установленным по умолчанию. Однако запросы по уровню 0 некогда не будут обрабатываться, так как в этом случае невозможно прервать текущую деятельность ЦПУ. Однако разрешенный запрос на прерывание на уровне 0000_B завершит режим покоя и заново активирует процессор.

Для обслуживаемых PEC запросов на прерывание, связанные с ними номера PEC-каналов извлекаются из ILVL (LSB) и GLVL (на рисунке ниже). Поэтому при программировании источников на 15 уровень (ILVL=1111_B) выбирается 7... 4 группа PEC-каналов, при программировании источников на 14 уровень (ILVL=1110_B) выбирается 3... 0 группа PEC каналов. Номер PEC-канала определяется в соответствии со значением в поле уровня группового приоритета GLVL.

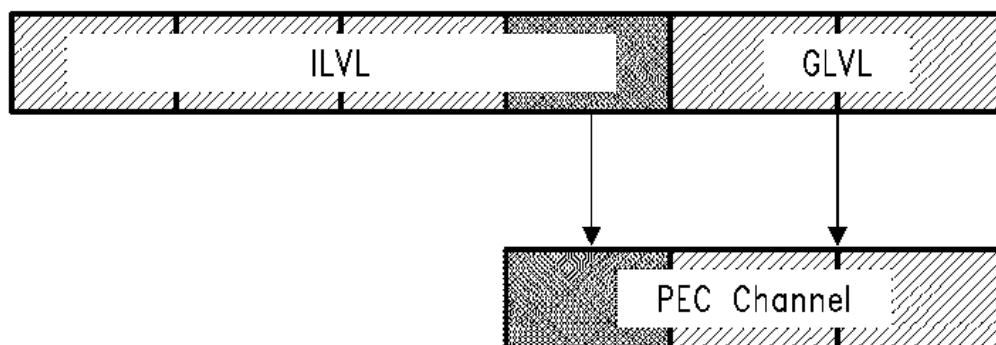


Рисунок 5-1

Уровни приоритетов и PEC-каналы

Одновременные запросы для PEC-каналов расставляются по приоритетам согласно номеру PEC-канала, причем канал 0 имеет наименьший, а канал 8 – максимальный уровень приоритета.

Примечание: Все источники, запрашивающие PEC-обслуживание, должны быть запрограммированы на различные PEC-каналы. В противном случае может быть активирован неправильный PEC-канал.

В таблице, приведенной ниже, показано несколько примеров, в которых указывается, какие действия необходимо предпринимать для программирования того или иного режима в регистре управления прерываниями.

Уровень приоритета		Тип обслуживания	
ILVL	GLVL	COUNT = 00 _H	COUNT ≠ 00 _H
1111	11	прерывание ЦПУ 15 уровень, групповой приоритет 3	РЕС-обслуживание канал 7
1111	10	прерывание ЦПУ 15 уровень, групповой приоритет 2	РЕС-обслуживание канал 6
1110	10	прерывание ЦПУ 14 уровень, групповой приоритет 2	РЕС-обслуживание канал 2
1101	10	прерывание ЦПУ 13 уровень, групповой приоритет 2	прерывание ЦПУ 13 уровень, групповой приоритет 2
0001	11	прерывание ЦПУ 1 уровень, групповой приоритет 3	прерывание ЦПУ 1 уровень, групповой приоритет 3
0001	00	прерывание ЦПУ 1 уровень, групповой приоритет 0	прерывание ЦПУ 1 уровень, групповой приоритет 0
0000	XX	Нет обслуживания!!!	Нет обслуживания!!!

Примечание: Все запросы на уровнях 13... 1 не могут использоваться в качестве запроса на РЕС-передачу. Они всегда обслуживаются при помощи подпрограмм прерываний. При этом не используется регистр РЕСС и не проверяется значение поля COUNT.

Функции управления прерываниями регистра PSW.

Слово состояния процессора (PSW) функционально делится на две части: нижняя часть слова в основном представляет арифметическое состояние ЦПУ, верхняя часть слова PSW управляет системой прерываний C167 и механизмом управления интерфейсом внешней шины.

Примечание: При включении/выключении разрешения на запрос на прерывание с помощью изменения состояния регистра PSW, необходимо учитывать конвейерные эффекты. (смотри раздел «Блок ЦПУ»).

Полностью регистр PSW описан в разделе «Блок ЦПУ».

Рассмотрим поподробнее поле ILVL. Это поле обозначает текущий уровень операций в ЦПУ. Оно отражает текущий уровень приоритета исполняемой программы. При входе в подпрограмму прерывания значение этого поля битов изменяется на значение уровня приоритета обслуживаемого прерывания. Перед этим в стеке сохраняется старое значение. Уровень ЦПУ определяет минимальный уровень прерывания, которое может быть

обслужено. Любое прерывание с таким же уровнем или ниже останется без ответа.

Для того чтобы управлять минимальным уровнем прерывания на который может быть разрешено обслуживание, текущий уровень приоритета ЦПУ может быть программно изменен.

РЕС-передача данных не является в полной мере прерыванием ЦПУ, так как «крадется» только один такт, поэтому РЕС-обслуживание не оказывает влияния на поле ILVL регистра PSW.

Аппаратные ловушки переключают уровень ЦПУ на максимальный уровень (т.е. на 15-ый), поэтому никакие прерывания или РЕС-запросы не будут обслуживаться, пока выполняется подпрограмма обслуживания ловушек.

Примечание: Команда TRAP не изменяет уровень ЦПУ, поэтому программно вызванная подпрограмма обслуживания ловушек может быть прервана запросом с более высоким приоритетом.

5.2 Операции с РЕС-каналами

Периферийный контроллер событий микроконтроллера C167 имеет 8 каналов обслуживания РЕС, которые перемещают одиночный байт или слово между двумя адресами в сегменте 0 (страницы данных 0... 3). Это самый быстрый возможный ответ на прерывание, и во многих случаях достаточный, чтобы обслужить необходимый запрос (такой как параллельные каналы, АЦП и т.д.). Каждый канал управляется с помощью специально предназначенного регистра счетчика/управления РЕС канала (РЕССх) и парой указателей для источника (SRCPх) и точки назначения (DSTPх) передачи данных.

РЕСС регистры управляют действием, совершаемым данным РЕС-каналом.

SFR

РЕССх (FECy_H/6z_H см. таблицу) Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	INC		BWT	COUNT							
-	-	-	-	-	rw		rw	rw							

Бит	Функция
COUNT	Отсчет PEC-передач Отсчитывает PEC-передачи и влияет действия канала
BWT	Выбор передачи байта/слова 0 – передача слова 1 – передача байта
INC	Управление увеличением (изменение SRCPx, DSTPx) 0 0 – указатель не меняется 0 1 – увеличивается DSTPx на 1 или 2 (BWT) 1 0 – увеличивается SRCPx на 1 или 2 (BWT) 1 1 – защищено. Не использовать. (изменено на 1 0 аппаратно)

Адреса регистра управления PEC

Регистр	Адрес	Простр.	Регистр	Адрес	Простр.
РЕСС0	FEC0 _H /60 _H	SFR	РЕСС4	FEC8 _H /64 _H	SFR
РЕСС1	FEC2 _H /61 _H	SFR	РЕСС5	FECA _H /65 _H	SFR
РЕСС2	FEC4 _H /62 _H	SFR	РЕСС6	FECC _H /66 _H	SFR
РЕСС3	FEC6 _H /63 _H	SFR	РЕСС7	FECE _H /67 _H	SFR

Бит передачи слова/байта BWT указывает, байт или слово будут переданы во время цикла PEC-обслуживания. Этот флаг указывает на размер передаваемых слов и на величину, на которую будет изменено значение указателя.

Поле управления увеличением INC указывает, который из PEC-указателей будет увеличен после передачи. Однако невозможно увеличить сразу оба указателя. Если указатели не увеличиваются (INC = 00), то представленный канал всегда передает данные из одного и того же источника в одну и ту же точку назначения.

Примечание: Зарезервированная комбинация «11» аппаратно изменяется на «10». Однако не рекомендуется использовать эту комбинацию.

Поле счетчика PEC-передач управляет работой своего PEC-канала. Содержимое битового поля COUNT определяет совершаемое действие во

время запроса. COUNT может позволять ограниченное число PEC-передач, нелимитированные передачи или полное отсутствие PEC-обслуживания.

Таблица, представленная ниже, суммирует, как от флага запроса на прерывание IR и действия PEC-канала зависит значение поля COUNT.

Предыдущее значение счетчика	Новое значение счетчика	IR после PEC-обслуживания	Действия PEC-канала Комментарии
FF _H	FF _H	0	Перемещение байта/слова Длительный режим передачи т.е. COUNT не меняется
FE _H ... 02 _H	FD _H ... 01 _H	0	Перемещение байта/слова и уменьшение значения COUNT
01 _H	00 _H	1	Перемещение байта/слова Устанавливается флаг запрещающий, PEC - обслуживание
00 _H	00 _H	(1)	Нет действий!!! Активируется подпрограмма обслуживания прерывания вместо PEC-канала

Счетчик количества PEC-передач позволяет обслуживать строго определенное количество запросов PEC-канала, и затем (при достижении счетчиком 00_H) активируется подпрограмма обслуживания прерывания, связанная с этим уровнем приоритета.

Неограниченные во времени передачи данных выбираются путем установки значения FF_H в битовом поле COUNT. В этом случае значение поля COUNT не изменяется, и PEC-канал обслуживает все запросы до тех пор, пока не будет отключен режим PEC-обслуживания.

Когда после передачи данных значение в COUNT доходит до 00_H, флаг запроса остается установленным. Если же COUNT уже содержит 00_H значение, то PEC-канал переходит в режим покоя, и взамен него используется подпрограмма обслуживания прерывания. Этот механизм позволяет выбирать, обслуживать запросы 14 и 15 уровней с помощью PEC или с помощью подпрограмм обслуживания прерываний.

Примечание: PEC-передачи данных выполняются только в том случае, если их уровень приоритета выше, т.е. только PEC-каналы 7... 4 могут обрабатываться, если ЦПУ работает на 14 уровне.

Все источники запросов на прерывание, которые разрешены и запрограммированы для PEC-обслуживания, должны использовать различные каналы. Иначе только одна передача будет осуществляться для всех одновременных запросов. Когда значение COUNT достигает нуля и ЦПУ прерывается, будет создан некорректный вектор прерывания.

Указатели источников и точек назначения выявляют адреса между которыми должна произойти передача данными. Пара указателей (SRCP_x и DSTP_x) связаны с каждым из 8 PEC-каналов. Эти указатели не находятся в области SFR-регистров, но расположены во внутренней RAM ниже побитно адресуемой области (на рисунке ниже).

DSTP7	00'FCFE _H	DSTP3	00'FCEE _H
SRCP7	00'FCFC _H	SRCP3	00'FCEC _H
DSTP6	00'FCFA _H	DSTP2	00'FCEA _H
SRCP6	00'FCF8 _H	SRCP2	00'FCE8 _H
DSTP5	00'FCF6 _H	DSTP1	00'FCE6 _H
SRCP5	00'FCF4 _H	SRCP1	00'FCE4 _H
DSTP4	00'FCF2 _H	DSTP0	00'FCE2 _H
SRCP4	00'FCF0 _H	SRCP0	00'FCE0 _H

Рисунок 5-2

Расположение PEC-указателей во внутренней памяти

PEC-передачи данных не используют указатели страниц данных DPP3... DPP0. Указатели источников и точек назначения используются как 16-битный адрес внутри сегмента 0, поэтому данные могут быть переданы между двумя точками только в первых четырех страницах данных 3... 0.

Адреса указателей для неактивных PEC-каналов могут быть использованы для хранения обычных данных. Место в RAM занимают только необходимые указатели.

Примечание: Если выбрана передача данных слова для конкретного PEC-канала (т.е. BWT = 0), связанные с ним указатели источника и точки назначения должны указывать на адрес четного байта. В ином случае при использовании этого, канала будет активирована ловушка неправильного доступа к слову данных.

5.3 Приоритетность прерываний и запросов на РЕС-обслуживание

Разрешение и запрещение запросов на прерывание возможно с помощью трех способов:

Биты управления позволяют переключать каждый индивидуальный источник. Биты управления (xxIE) расположены в отдельных регистрах управления прерываниями. Все запросы на прерывание могут быть глобально включены или отключены посредством бита IEN регистра PSW.

Уровни приоритетов автоматически выбирают определенную группу запросов на прерывание, которые будут допущены. Уровень приоритета источника, победившего в сравнении с текущим уровнем приоритета ЦПУ и уровнями запросов других источников, будет обслужен только в том случае, если его уровень выше уровня приоритета ЦПУ. Программное изменение уровня ЦПУ до необходимого значения позволяет блокировать все запросы этого же или более низкого уровня. Все источники прерываний, которым назначен нулевой уровень, будут отключенными и никогда не будут обслужены.

Команды АТОМІС и EXT автоматически запрещают все запросы на прерывание во время исполнения следующих четырех команд. После завершения этих команд не требуется заново включать разрешения на прерывания (см. раздел «Системное программирование»).

Руководство классами прерываний

Класс прерываний объединяет в себя набор источников прерываний с одинаковой важностью, т.е. с одинаковой приоритетностью с точки зрения системы. Прерывания одного класса не могут прервать друг друга.

Классы с числом членов меньше 4 могут быть представлены в одном и том же уровне приоритета (ILVL) и разделены между собой с помощью группового уровня (GLVL). Функционирование класса при этом будет поддерживаться автоматически при помощи контроллера прерываний.

Классы с числом членов больше 4 могут быть созданы с помощью использования некоторого количества рядом расположенных уровней приоритетов (ILVL) и их групповых уровней (до 4 на уровень). Каждая подпрограмма прерывания из этого класса устанавливает уровень ЦПУ на максимальный уровень приоритета для этого класса. Все запросы такого же или более низкого уровня будут заблокированы, т.е. класс считается используемым.

5.4 Сохранение состояния во время обслуживания прерывания

Прежде чем запрос на прерывание начнет обслуживаться, состояние текущей задачи автоматически будет сохранено в системном стеке. Состояние ЦПУ (PSW) сохраняется до тех пор, пока выполнение прерванной задачи не будет продолжено после возвращения из подпрограммы прерывания. Точка возврата определяется с помощью IP и, в случае использования сегментированного режима памяти, указателя сегмента кода CSP.

В системном стеке сначала сохраняется значение PSW, затем IP (в несегментированном режиме) или затем CSP и IP (для сегментированного режима). Эта последовательность оптимизирует использование системного стека, при отключенной сегментации.

Значение поля уровня приоритета ЦПУ (ILVL в PSW) изменяется на значение приоритета обслуживаемого запроса на прерывание, поэтому ЦПУ после этого работает с новым уровнем приоритета. Если во время выполнения операции умножения или деления было допущено прерывание, то бит MULIP регистра PSW устанавливается в «1». В этом случае значение, сохраненное в стеке, не адрес следующей команды, а продолжение выполнения команды умножения и деления, так как эта команда была прервана и необходимо завершить ее после возврата из прерывания.

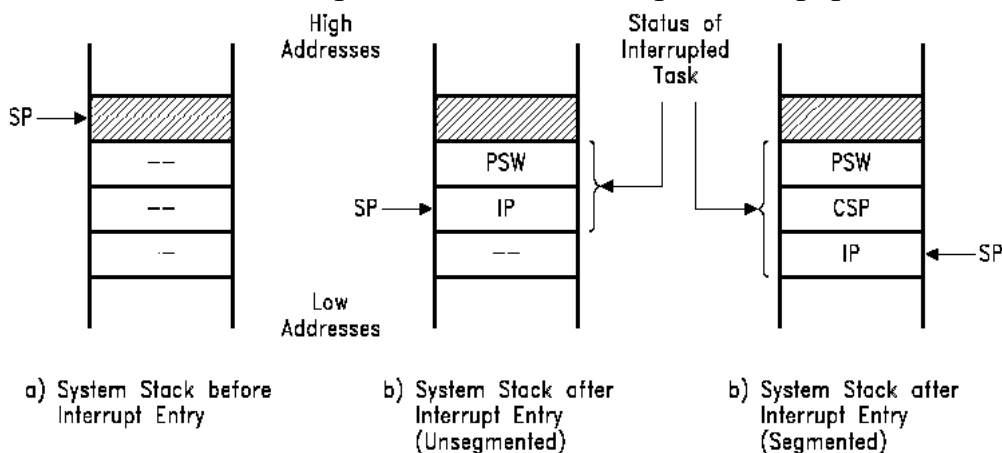


Рисунок 5-3

Состояние задачи, сохраненное в системном стеке

Флаг обслуживаемого запроса на прерывание устанавливается в 0. В IP-регистр загружается вектор, связанный с источником запроса (в случае режима сегментированной памяти значение CSP очищается), после этого первая команда подпрограммы прерывания вызывается из адреса вектора, необходимого для перехода в саму подпрограмму прерывания. Значение указателя страницы данных и Context Pointer при этом не изменяются.

Когда подпрограмма обслуживания прерывания заканчивает свою работу (после выполнения команды RETI), информация о состоянии вызывается из системного стека в обратном порядке.

Переключение context

Подпрограмма обслуживания прерывания обычно сохраняет все используемые регистры в стеке и перед возвращением восстанавливает их значения. Чем больше регистров используется в подпрограмме прерывания, тем больше времени теряется при сохранении и восстановлении. C167 позволяет переключать полный банк ЦПУ-регистров (GPR-регистры) за одну команду, и поэтому подпрограмма обслуживания использует собственный независимый банк GPR.

Команда «SCXT CP, #New_Bank» отсылает содержимое CP в системный стек и загружает в CP значение базового адреса «New_Bank». С этого момента подпрограмма использует собственные GPR-регистры. При завершении выполнения подпрограммы прерывания этот банк регистров сохраняется, и таким образом содержимое банка будет доступно при следующем прерывании.

Перед возвращением из прерывания (команда RETI), предыдущее значение CP просто записывается назад из системного стека.

Примечание: Первая команда после SCXT не должна использовать GPR.

Ресурсы, используемые подпрограммой прерывания, необходимо сохранять при входе в подпрограмму обслуживания и необходимо восстанавливать при возврате в основную программу, в том числе DPP и регистры модуля умножения и деления.

5.5 Время ответа на прерывание

Время ответа на прерывание представляет из себя временной интервал, который проходит с момента установки флага запроса на прерывание разрешенного источника прерывания до того момента, когда первая команда (I1) вызывается из адреса вектора прерывания. Как правило время ответа на прерывание занимает 3 командных такта.

Стадии конвейера	Такт 1	Такт 2	Такт 3	Такт 4
Выборка	N	N+1	N+2	I1
Декодирование	N-1	N	TRAP (1)	TRAP(2)
Исполнение	N-2	N-1	N	TRAP
Запись	N-3	N-2	N-1	N

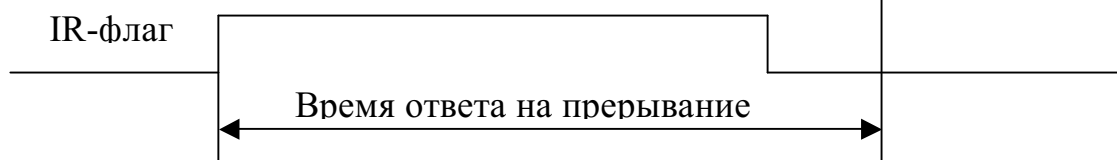


Рисунок 5-4

Временные диаграммы работы конвейера при входе в прерывание

Все команды в конвейере, включая команду N (во время исполнения которой устанавливается флаг запроса на прерывание), завершаются прежде чем начинается выполнение подпрограммы прерывания. Реальное время выполнения этих команд влияет на время ответа на прерывание.

На рисунке, приведенном выше, флаг запроса на прерывание устанавливается в 1-ом такте (вызов команды N). Во время 2-ого такта производится сравнение уровней приоритетов источников прерывания и выбирается источник с максимальным уровнем. В третьем такте TRAP команда инжектируется в стадию декодирования конвейера, заменяя собой команду N+1 и устанавливая в 0 флаг запроса на прерывание. Такт 4 завершает инжектированную TRAP команду (сохраня PSW, IP и CSP в случае сегментированной памяти) и вызывает первую команду из адреса вектора.

Все команды, введенные на конвейер после установки флага запроса на прерывание (N+1, N+2), будут выполнены после возвращения из подпрограммы обслуживания прерывания.

Минимальное время ответа на прерывание составляет 5 тактов (250 нс @ 20 МГц CPU clock). При этом выполняются команды только из внутренней ROM, отсутствуют запросы на чтение внешних операндов, а также нет запросов на прерывание во время последнего состояния такта команды. При установке флага запроса на прерывание во время первой стадии командного такта, минимально возможное время ответа увеличивается до 6 тактов (300 нс @ 20МГц CPU Clock).

Время ответа на прерывание увеличивается за счет задержки в выполнении команд в конвейере, которые обрабатывались до входа в подпрограмму прерывания (включая команду N).

- Когда между командами N-2/N-1 или N-1/N или N имеет место прямая запись значений в PSW или SP, минимальное время ответа может увеличиться на 1 такт.

- Когда команда N совершает чтение операнда из внутренней ROM, или когда команда N – call, return, trap или MOV RN (Rm. #data16) расположена во внутренней ROM, минимальное время может дополнительно увеличиться на 2 такта.

- В случае чтения командой N регистра PSW, и если команда N-1 оказывает влияние на флаги состояния, время ответа на прерывание может дополнительно увеличиться на 2 такта.

В худшем случае, время ответа на прерывание, во время выполнения программы внутренней ROM, может дополнительно вырасти до 12 тактов (600нс @ 20МГц CPU Clock).

Любой обращение ко внешней памяти может дополнительно увеличить время ответа на прерывание. Могут наблюдаться следующие варианты:

- Вызов команды из внешней памяти
- Операнд читается из внешней памяти
- Результат записывается во внешнюю память

В зависимости от того, где находится команда, где находится исходный и конечный операнд, определяется количество комбинаций. Однако заметим, что на задержку влияют только конфликты доступа.

Несколько примеров иллюстрируют эти задержки:

- В наихудшем случае, время ответа на прерывание, при доступе к внешней памяти имеет место в том случае, когда команды N, N+1, N+2 выполняются из внешней памяти, команды N-1 и N требуют чтения внешних операндов, команды N-3 ... N записывают свои операнды назад во внешнюю память, и вектор прерывания указывает на адрес в внешней памяти. В этом случае время ответа на прерывание состоит из времени осуществления 9 циклов доступа к 16-разрядной внешней шине. Это обусловлено тем, что команда I1 не может быть вызвана с помощью внешней шины до тех пор, пока не будут записаны, вызваны и прочтены данные от предыдущих команд.

- Когда в примере, описанном выше, вектор направлен во внутреннюю память, время ответа на прерывание состоит из 7 доступов ко внешней шине и из 2 тактов, потому что вызов команды I1 из внутренней ROM не может начаться раньше чем будет произведен доступ к внешней шине.

Если при выходе из подпрограммы прерывания по команде RETI, уже установлен другой флаг запроса на прерывание, следующее обслуживание прерывания не начнется до тех пор, пока не будут завершены два последних цикла команд, прерванной программы. В большинстве случаев за это время как раз выполняются две команды. Если первая команда после команды RETI

– команда перехода (без попадания кэша), или если читается операнд из внутренней ROM, или если адрес команды не расположен в RAM, то будет выполнена только одна команда.

Примечание: Доступ к шине в контексте также включает задержку, которая имеет место из-за внешнего сигнала \overline{READY} или из-за арбитража внешней шины в режиме HOLD.

Время ответа PEC.

Время ответа PEC представляет из себя временной интервал, который проходит с момента установки «1» в флаге разрешения прерывания, до момента начала PEC-передачи данных. Как правило время PEC-ответа составляет два такта.



Рисунок 5-5

Временные диаграммы работы конвейера при входе в PEC-обслуживание

На рисунке, представленном выше, флаг запроса на прерывание устанавливается в 1-ом такте (вызов команды N). В течении 2 такта производится выявление источника прерывания с самым высоким приоритетом. В 3 такте «команда» PEC-передачи данных инжектируется в стадию декодирования конвейера, вместо команды N+1, при этом флаг запроса на прерывание устанавливается в «0». Такт 4 завершает инжектированную PEC-передачу данных и возвращается к выполнению команды N+1.

Все команды, введенные в конвейер, после установки флага запроса на прерывание (N+1, N+2) будут выполнены после PEC-передачи данных.

Примечание: Если в момент нахождения прерывания с максимальным уровнем, производится чтение регистра PECC7... PECC0, и если побеждает PEC-запрос, то этот такт пропускается, и процесс нахождения запроса с максимальным уровнем приоритета повторяется.

Минимальное время ответа ПЕС составляет 3 такта (150 нс @ 20 МГц CPU Clock). Это происходит при выполнении программы из внутренней памяти, когда не используются внешние операнды, и не устанавливаются флаги запросов на прерывания во время первого состояния командного такта. Если производится установка флага запроса на прерывание во время первого состояния командного такта, минимальное время ответа ПЕС составляет 4 такта (150 нс @ 20 МГц CPU Clock).

Различные случаи, при которых увеличивается время задержки рассмотрены выше.

В наихудшем случае время ответа ПЕС увеличивается на 7 доступов к внешней шине.

5.6 Внешние прерывания

Хотя C167 не имеет специальных входов для внешних прерываний, обеспечивается много возможностей для реагирования на внешние события. Это достигается путем использования части портов ввода-вывода для входов прерываний. Функция прерывания может быть либо соединена с главной функцией порта, либо использоваться вместо основной функции, т.е. в том случае, когда не требуется использовать основную функцию этого вывода порта.

Сигнал прерывания может быть подключен к:

- CC31IO... CC0IO, линии захвата/сравнения модуля CAPCOM
- T4IN, T2IN – входы таймера
- CAPIN – вход захвата GPT2

Для каждого из этих входов, в качестве запроса на прерывание может быть выбран либо положительный, либо отрицательный фронт, либо и положительный и отрицательный фронт запроса на прерывание либо ПЕС-обслуживание одновременно. Выбор фронта совершается в регистре управления периферийного модуля, связанного с данным выводом порта. Периферия должна быть запрограммирована на специальный режим работы, позволяющий использовать вход для входов внешних прерываний. Приоритеты внешних сигналов определяются в регистре управления прерываниями от периферийного модуля, и обслуживания внешнего запроса на прерывание будет использоваться, связанный с ним вектор.

Примечание: В случае использования любого из перечисленных внешних входов прерываний, необходимо обязательно переключать линию порта на вход, изменяя значение бита управления направлением DPx.y регистра управления направлением порта DPx.

Выводы микросхемы, используемые для входов внешних прерываний.

Вывод порта	Оригинальная функция	Регистр управления
P2.0-15/CC0-15IO	Вход захвата 0 – 15 CAPCOM	CC0-CC15
P8.0-7/CC16-23IO	Вход захвата 16 – 23 CAPCOM	CC16-CC23
P1H.4-7/CC24-27IO	Вход захвата 24 – 27 CAPCOM	CC24-CC27
P7.4-7/CC28-31IO	Вход захвата 28 – 31 CAPCOM	CC28-CC31
P1H.4-7/CC24-27IO	Вход захвата 24 – 27 CAPCOM	CC24-CC27
P3.7/T2IN	Вспомогательный вход таймера T2	T2CON
P3.5/T2IN	Вспомогательный вход таймера T4	T4CON
P3.2/CAPIN	Вход захвата GPT2	T5CON

В случае использования выводов порта CCxIO как входов внешних прерываний, необходимо выбирать режим захвата в битовом поле CCMODx регистра управления регистром захвата и сравнения CCx. При записи значения 001_B в CCMODx, при обнаружении положительного фронта внешнего сигнала на входе CCxIO, будет устанавливаться флаг запроса на прерывание CCxIR. При записи значения 010_B в CCMODx, при обнаружении отрицательного фронта внешнего сигнала на входе CCxIO, будет устанавливаться флаг запроса на прерывание CCxIR. Во всех трех случаях, содержимое CAPCOM таймера будет заперто в регистре захвата CCx, при этом не имеет значение работает таймер или нет. Когда бит разрешения прерывания CCxIE установлен в «1», будет подан запрос на PEC-обслуживание или запрос на прерывание по вектору CCxINT, в случае прихода внешнего сигнала.

Выводы T2IN или T4IN могут использоваться для входов внешних прерываний, в том случае когда вспомогательный таймер T2 или T4 в блоке GPT1 настроен на режим захвата. Этот режим выбирается в случае установки значения 101_B в управляющем поле T2M или T4M регистра управления T2CON или T4CON. Тип фронта срабатывания по внешнему сигналу выбирается при задаче значений в битовых полях T2I или T4I. При установке в этих полях значения X01_B, флаги запроса на прерывание T2IR или T4IR в регистрах T2IC и T4IC будут устанавливаться при приходе положительного внешнего фронта сигнала на выводах T2IN или T4IN. При установке в этих полях значения X10_B, флаги запроса на прерывание T2IR или T4IR в регистрах T2IC и T4IC будут устанавливаться при приходе отрицательного внешнего фронта сигнала на выводах T2IN или T4IN. При программировании T2I или T4I X01_B, то отрицательный фронт входных сигналов устанавливает флаги запросов на прерывание. При установке X11_B в битовых полях T2I и T4I, флаг запроса на прерывание устанавливается как при положительном фронте входных сигналов, так и отрицательном. Во всех трех случаях содержимое ядра таймера T3 будет захвачено в дополнительный регистр таймера T2 или T4, при подаче сигнала на вход T2IN или T4IN. При установке «1» в битах разрешения прерывания T2IE или T4IE, будет

сгенерирован PEC-запрос или запрос на прерывание по вектору T2INT или T4INT.

Использование вывода CAPIN незначительно отличается от использования выводов входа таймера для подачи внешних прерываний. При установке в «0» бита разрешения режима захвата T5SC регистра T5CON, фронт сигнала на входе CAPIN устанавливает флаг запроса на прерывание CRIR регистра CRIC, и при этом не задействуется функция захвата регистра CAPREL.

Регистр CAPREL может использоваться для перезагрузки содержимого регистра таймера T5, одновременно с использованием входа CAPIN для подачи внешних прерываний. С помощью битового поля CI регистра T5CON выбирается необходимый фронт для прерывания то внешнего сигнала. При установке CI = 01_B используется положительный фронт. Если CI = 10_B – отрицательный фронт. Если CI = 11_B – используются как положительный так и отрицательный фронт внешнего сигнала запроса на прерывания. При установке в «1» бита разрешения прерывания CRIE, генерируется PEC-запрос или запрос на прерывание по вектору CRINT.

Примечание: Не маскируемый вход прерывания \overline{NMI} и вход RESET \overline{RSTIN} обеспечивают возможность реагирования ЦПУ на внешние входные сигналы. \overline{NMI} и \overline{RSTIN} предназначены для вызывающих аппаратные ловушки входов.

5.7 Функции ловушек

Обычное выполнение прерываний ловушек похоже на выполнение стандартных прерываний. Однако, в тех случаях когда необходима немедленная реакция системы, функции ловушек обеспечивают возможность обхода системы приоритетов. Функции ловушек не маскируются и всегда имеют приоритет выше, чем уровень приоритета запроса на прерывание.

C167 обеспечивает два различных типа механизмов ловушек. **Аппаратные ловушки** реагируют на события, которые случаются по ходу выполнения программы (т.е. неправильный доступ или неопределенный программный код), **программные ловушки** иницируются с помощью команды во время выполнения программы.

Программные ловушки

Команда TRAP используется для того, чтобы произвести программный вызов подпрограммы прерывания. Номер ловушки, обозначенный в поле операнда команды TRAP, определяет адрес вектора перехода в диапазоне адресов от 00`0000_H до 00`01FC.

При выполнении команды TRAP имеет место эффект, аналогичный запросу на прерывание по этому вектору. Значения PSW, CSP (в случае

сегментации памяти) и IP сохраняются во внутреннем системном стеке, и после этого происходит переход по адресу вектора. При включенной сегментации в случае выполнения ловушки, для обслуживания подпрограммы прерывания значение CSP устанавливается для кодового сегмента 0. При выполнении команды TRAP не устанавливаются флаги запроса на прерывание. Вызванная командой TRAP, подпрограмма обслуживания прерывания должна быть завершена командой RETI (возврат из прерывания) для обеспечения корректного выполнения.

Примечание: Уровень ЦПУ в регистре PSW не изменяется после использования команды TRAP, поэтому подпрограмма обслуживания выполняется на том же уровне приоритета, что и основная программа. Таким образом, подпрограмма обслуживания, выполняемая по команде TRAP, может быть прервана другой ловушкой или прерыванием с более высоким уровнем приоритета.

Аппаратные ловушки

Аппаратные ловушки являются следствием ошибок либо особых состояний системы, который могут происходить во время выполнения программы. Аппаратные ловушки можно также совершать преднамеренно, т.е. вводя некорректный код, и при этом генерируется ловушка неправильного программного кода. C167 различает восемь различных функций аппаратных ловушек. При обнаружении состояния аппаратной ловушки, ЦПУ переходит по адресу вектора данной ловушки. В зависимости от состояния обнаруженной ловушки, вызвавшая ее команда может быть либо завершена либо не завершена, прежде чем будет введена подпрограмма обслуживания ловушки.

Аппаратные ловушки – не маскируемы и всегда имеют уровень приоритета выше, чем какое-либо другое состояние ЦПУ. В случае определения в одном и том же командном такте нескольких состояний аппаратных ловушек, будет обслужена аппаратная ловушка с наивысшим уровнем приоритета (см. таблицу в подразделе «Структура системы прерываний»).

Содержимое регистров PSW, CSP (в режиме сегментированной памяти) и IP сохраняется во внутреннем системном стеке, и уровень ЦПУ устанавливается в максимально возможное значение уровня приоритета (т.е. 15 уровень), при этом запрещаются все прерывания. В режиме включенной сегментации CSP выставляет нулевой сегмент кода. Подпрограмма обслуживания ловушки завершает свою работу по команде RETI.

Восемь функций аппаратных ловушек подразделяется на два класса:

Класс А ловушек содержит

- внешнее не маскируемое прерывание \overline{NMI}
- переполнение стека

- underflow стека

Эти ловушки делят между собой один уровень приоритета, при этом каждая имеет индивидуальный вектор адреса

Класс В ловушек содержит

- Неопределенный программный код
- Ошибку защиты
- Неправильный доступ к операнду слов
- Неправильный командный доступ
- Ловушка неправильного доступа к внешней шине

Эти ловушки имеют один и тот же уровень приоритета, и одинаковый адрес вектора.

Адресуемый побитно регистр флагов ловушек (TFR) позволяет подпрограмме прерывания определить тип случившейся ловушки. Каждая функция ловушки указывается с помощью независимого флага. При совершении аппаратной ловушки, устанавливается в «1», связанный с ней флаг в регистре TFR.

SFR

TFR (FFAC_H/D6_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMI	STK OV	STK UF	-	-	-	-	-	UND OPC	-	-	-	PRT FLT	ILL OPA	ILL INA	ILL BUS
rw	rw	rw	-	-	-	-	-	rw	-	-	-	rw	rw	rw	rw

Бит	Функция
ILLBUS	Флаг неправильного доступа к внешней шине Попытка внешнего доступа с неопределенной внешней шиной
ILLINA	Флаг неправильного командного доступа Попытка перехода к нечетному адресу
ILLOPA	Флаг неправильного доступа к слову операнда Попытка доступа (чтения или записи) по нечетному адресу слова операнда
PRTFLT	Флаг ошибки защиты Обнаружение защищенной команды в неправильном формате
UNDOPC	Флаг неопределенного программного кода Текущая декодируемая команда не является командой C167
STKUF	Флаг underflow стека Текущее значение указателя стека превышает содержимое регистра STKUN

Бит	Функция
STKOF	Флаг переполнения стека Текущее значение указателя стека меньше содержимого регистра STKOV
NMI	Не маскируемый флаг прерывания Определение отрицательного фронта на выводе \overline{NMI}

Примечание: Подпрограмма обслуживания ловушки должна очистить флаг, вызвавшей ее, ловушки, в ином случае будет произведен новый запрос на обслуживание ловушки после выхода из подпрограммы обслуживания. Программная установка флага запроса ловушки приводит к эффекту, аналогичному аппаратной установке ловушки.

Функции RESET (аппаратные, программные, от сторожевого таймера) могут быть отнесены к типу ловушек. Функции RESET имеют высочайший системный приоритет (уровень приоритета ловушек III).

Ловушки класса А имеют второй уровень приоритета, в третьей категории находится класс В ловушек, и поэтому класс А ловушек может прервать класс В. Если одновременно случается больше одной ловушки класса А, происходит внутреннее распределение по приоритетам, при этом NMI ловушка имеет высочайший приоритет, а ловушка underflow стека имеет наименьший уровень.

Все ловушки класса В имеют одинаковый уровень приоритета (уровень I). При одновременном активизировании нескольких ловушек класса В, устанавливаются соответствующие флаги в регистре TFR, и начинается обслуживание подпрограммы ловушки. Поскольку все ловушки класса В имеют один и тот же вектор, то одновременных ловушек класса В, определяется программно во время выполнения подпрограммы обслуживания.

Если во время обслуживания ловушек класса В, имеет место ловушка класса А, она будет немедленно обслужена. Однако во время выполнения подпрограммы обслуживания ловушки класса А, ни одна ловушка класса В не будет обслужена до тех пор, пока подпрограмма обслуживания ловушки не завершится командой RETI. В этом случае случившаяся ловушка класса В сохраняется в TFR-регистре, но при этом теряется значение IP команды, при которой имела место ловушка.

В случае обнаружения ловушки неопределенного кода (класс В) одновременно с NMI-ловушкой (класс А), устанавливается и флаг NMI и флаг UNDOCP, при этом сохраняется в системном стеке IP команды, вызвавшей ловушку неопределенного кода, и начинается обслуживание NMI. После возврата из подпрограммы обслуживания NMI, значение IP

возвращается из стека и немедленно посылается обратно в стек, так как необходимо выполнить обслуживание UNDOCP-ловушки.

Внешняя NMI ловушка

При обнаружении перехода отрицательного фронта на входе \overline{NMI} (не маскируемое прерывание), NMI флаг регистра TFR устанавливается в «1» и ЦПУ начинает выполнять подпрограмму обслуживания.

Примечание: NMI-вывод проверяется каждый такт ЦПУ для выявления фронтов.

Ловушка неопределенного кода

Эта ловушка может быть использована для эмуляции неизвестных команд. Подпрограмма обслуживания ловушки может проверить ошибочный код, базируясь на сохраненном в IP значении. Для продолжения работы основной программы, прежде чем исполнить команду RETI, сохраненное в стеке значение IP должно быть увеличено на значение эквивалентное месту, занимаемому неизвестной командой.

Ловушка ошибки защиты

Всякий раз при исполнении одной из защищенных команд, в том случае когда код команды не повторяется дважды во втором слове команды, и байт последующего кода не является дополнительным к первому, устанавливается флаг PRTFLT регистра TFR и ЦПУ начинает исполнять подпрограмму ловушки ошибки защиты. Защищенные команды включают SISWDT, EINIT, IDLE, PWRDN, SRST и SRVWDT. Значение IP посылается в системный стек перед выполнением подпрограммы, и затем записывается назад при выходе из подпрограммы

6 Параллельные порты

Для передачи или приема параллельных данных и одиночных внешних сигналов управления, в микроконтроллере C167 присутствует 111 линий параллельного ввода-вывода. Архитектура портов содержит один 16-разрядный порт IO (Порт 2), восемь 8-разрядных портов IO (Порт 0 P0H – P0L, Порт 1 0 P0H – P0L, Порт 4, Порт 6, Порт 7, Порт 8), один 15-разрядный порт IO (Порт 3) и один 16-разрядный входной порт (Порт 5).

Эти линии портов могут быть использованы для операций ввода вывода основного назначения под программным управлением, или могут быть использованы для интегрированной периферии C167 или для контроллера внешней шины.

Все линии портов являются адресуемыми побитно, и все линии ввода-вывода индивидуально программируются на вход или вывод, с помощью регистров направления (за исключением Порта 5). Порты ввода-вывода являются двунаправленными портами, которые можно переключать в состояние высокого сопротивления при настройке на вход. Выходные драйвера пяти портов ввода-вывода (2, 3, 6, 7, 8) могут быть сконфигурированы для работы в режиме push/pull или для работы в режиме с открытым коллектором с помощью контрольных регистров. Логический уровень на входе измеряется во входном триггере один раз за такт, при этом не имеет значения конфигурация порта (на вход или на выход).

Если произвести операцию записи в порт, настроенного на вход, то данное значение будет записано в выходной триггер порта, однако операция чтения порта перезаписывает значение на входе в триггер порта. Операция чтение-изменение-запись читает значение вывода микросхемы, модифицирует его и записывает назад в выходной триггер.

Запись в вывод, сконфигурированного на выход ($DPx.y = \text{«1»}$) приводит к тому, что выходной триггер и вывод микросхемы соединены, поскольку выходной буфер включен. Чтение этого вывода приводит к возврату значения выходного триггера. Операция чтение-изменение-запись читает значение выходного триггера, изменяет его и записывает назад в выходной триггер, таким образом также изменяя уровень на выводе микросхемы.

Data Input / Output Registers	Direction Control Registers	Threshold / Open Drain Control Registers
P0L	DP0L E	PICON E
P0H	DP0H E	
P1L	DP1L E	
P1H	DP1H E	
P2	DP2	ODP2 E
P3	DP3	ODP3 E
P4	DP4	
P5		
P6	DP6	ODP6 E
P7	DP7	ODP7 E
P8	DP8	ODP8 E

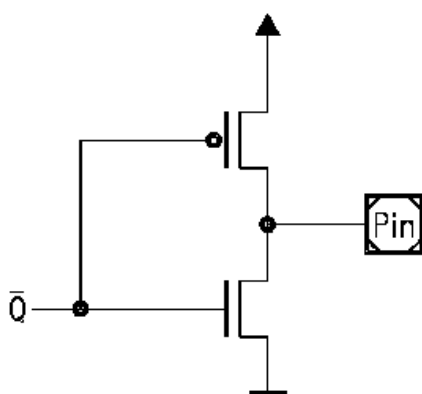
Рисунок 6-1

SFR-регистры и выводы микросхемы для параллельного порта

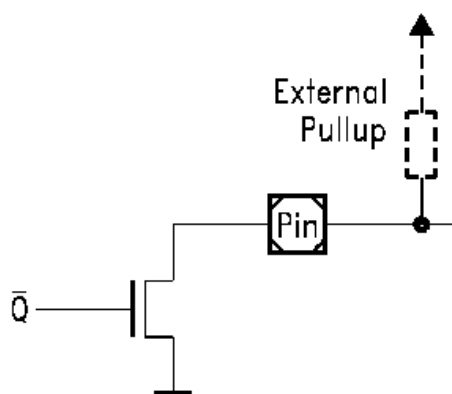
Режим с открытым коллектором

В C167 часть портов имеют возможность работы в режиме с открытым коллектором. В режиме push/pull выходной драйвер имеет транзисторы как на высокой так и на нижней стороне, таким образом линия замыкается либо на высокий либо на низкий уровень напряжения. В режиме с открытым коллектором верхний транзистор всегда выключен и поэтому выходной драйвер может устанавливать только низкий уровень на выходе. При записи «1» в триггер порта, нижний транзистор выключается и выход переходит в состояние высокого сопротивления. Высокий уровень в этом случае должен быть обеспечен внешним устройством. С использованием этой возможности, возможно объединять по несколько портов вместе для обеспечения конфигурации wired-AND, что позволяет без дополнительного программного кода осуществлять операции И/ИЛИ для разрешения или запрещения выходного сигнала.

Эта функция добавлена в порты P2, P3, P6, P7 и P8 (см. соответствующие подразделы) и управляется с помощью специального регистра управления режимом с открытым коллектором ODPx. Эти регистры позволяют переопределять отдельные выводы в этот режим. Если в ODPx.y установлена «1», то в этом случае выбран режим с открытым коллектором. Заметим, что эти регистры находятся в ESFR-области.



Push/Pull Output Driver



Open Drain Output Driver

Рисунок 6-2

Выходные драйвера в режиме Push/Pull и в режиме открытого коллектора

Управление входными порогоми

Стандартные входы C167 определяют состояние входных сигналов, согласно уровням ТТЛ. Для того чтобы выделять сигналы шумов для всех выводов портов 2, 3, 7, 8, можно установить отличные от стандартных уровни ТТЛ, более близкие к CMOS входные уровни. Специальные уровни определяются над ТТЛ-уровнями и устанавливают определенные значения границ гистерезиса для предотвращения срабатываний от шумов, в то время когда сигнал близок к граничным уровням.

Регистр управления входами портов PICON позволяет выбирать уровни для каждого байта порта, иными словами для управления 8-битными портами 7 и 8 отводится по одному биту, в то время как для портов 2 и 3 отводится по два бита.

ESFR
PICON (F1C4_H/E2_H) Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P8LIN	P7LIN	-	-	P3HIN	P3LIN	P2HIN	P2LIN
								rw	rw	-	-	rw	rw	rw	rw

Бит	Функция
PxLIN	Выбор входного уровня порта X меньший байт 0: Выводы Px.7 ... Px.0 переключаются по стандартным TTL-уровням 1: Выводы Px.7 ... Px.0 переключаются по специальным уровням
PxHIN	Выбор входного уровня порта X старший байт 0: Выводы Px.15 ... Px.8 переключаются по стандартным TTL-уровням 1: Выводы Px.15 ... Px.8 переключаются по специальным уровням

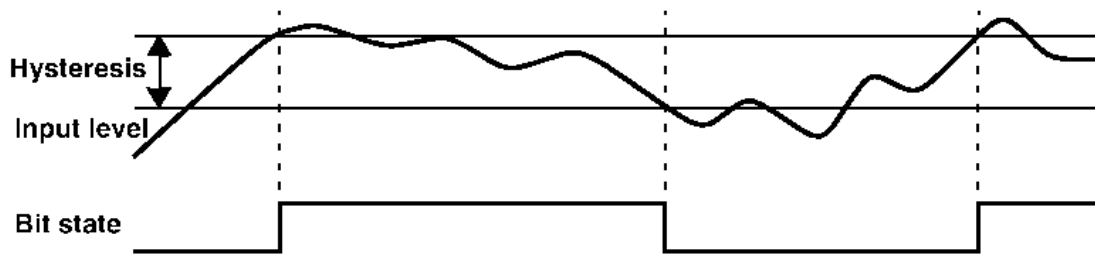


Рисунок 6-3

Гистерезис входных уровней

Альтернативные функции портов.

Каждая линия порта имеет одну альтернативную программируемую входную или выходную функцию.

Порт 0 и Порт 1 могут быть использованы как линии адреса и линии данных для доступа к внешней памяти.

Порт 4 обеспечивает дополнительные биты для сегментированного выхода A23/19/17... A16 в системах, где доступны больше 64 Кбайт памяти.

Порт 6 обеспечивает функции вывода сигнала chip select и линии арбитража шины.

Порт 2, 7, 8 связан с входами захвата или выходами сравнения блока CAPCOM и/или с выходами ШИМ-блока.

Порт 2 также используется для быстрых внешних входов прерываний и для входа таймера 7.

Порт 3 включает в себя альтернативные функции ввода/вывода таймера, последовательного интерфейса, сигнал управления шиной $\overline{BHE}/\overline{WRH}$ и выход тактового генератора (CLKOUT).

Порт 5 используется для аналоговых входных каналов АЦП или сигналов управления таймером.

При использовании альтернативной функции вывода порта для вывода данных, порт должен быть направлен на выход ($DPx.y=1$), за исключением некоторых сигналов, постоянно использующихся после RESET и настраиваемых автоматически. В ином случае выводы остаются в состоянии высокого сопротивления и не используются в качестве альтернативных выходных функций. В триггер порта необходимо записать «1», потому что он объединяется с данными альтернативного вывода по функции AND (исключая выходные ШИМ-сигналы).

Если используется альтернативная функция ввода, то направление вывода порта должно быть запрограммировано на вход ($DPx.y = 0$). После RESET направление порта автоматически устанавливается на вход. Если никаких внешних устройств не подключено к выводу порта, то направление порта не влияет на значение в выходном триггере порта. В этом случае, вход порта отражает состояние выходного триггера порта. Таким образом, альтернативная входная функция читает значение, сохраненное в выходном триггере.

Для большинства линий портов, во время использования альтернативной функции ввода или вывода за установки необходимого направления отвечает пользовательская программа. Это достигается путем установки значения в бите управления направлением вывода порта $DPx.y$, перед началом использования альтернативной функции. Однако имеются линии порта, в которых автоматически устанавливается направление. Например в режиме мультиплексной внешней шины порта 0, направление переключаться несколько за один цикл чтения шины. Очевидно, что это невозможно сделать посредством команд. В этих случаях, направление порта меняется автоматически аппаратно.

Структура всех портов, имеющих только входную альтернативную функцию, одинакова. Линии порта, имеющие только выходную альтернативную функцию, имеют различные структуры. Это обусловлено тем, что направление вывода порта имеет возможность переключаться и зависеть от того, доступен ли порт для пользовательской программы или нет в режиме альтернативной функции.

Все линии портов, не использующиеся альтернативными функциями, могут быть использованы для линий ввода-вывода основного назначения. При использовании выводов портов для вывода основного назначения, для

включения выходных драйверов необходимо записать инициализирующее значение в `prior` замка порта, во избежании нежелательных передач на выход порта.

SINGLE_BIT:	BSET	P4.7	; установленный уровень
			; выхода – «высокий»
	BSET	DP4.7	; переключение на
			; выходной драйвер

BIT_GROUP:	BFLDH	P4, #24H, #24H	; установленный уровень
			; выхода – «высокий»
	BFLDH	DP4, #24H, #24H	; переключение на
			; выходной драйвер

Примечание: При использовании нескольких пар BSET, для управления большим количеством выводов порта, необходимо чтобы эти пары были независимыми с помощью команд, не ссылающихся на порт (см «Эффект конвейера» в разделе «Блок ЦПУ»).

Каждый из этих портов и их альтернативные функции ввода и вывода описывается в последующих подразделах.

6.1 Порт 0

Два восьмибитных порта P0H и P0L представляют из себя две части порта 0. В обе половины порта 0 может производить запись без изменения значения другой половины.

В том случае, когда этот порт используется для функции ввода-вывода основного назначения, направление каждой линии может быть сконфигурировано с помощью регистров направления DP0H и DP0L.

SFR

P0L (FF00_H/80_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P0L.7	P0L.6	P0L.5	P0L.4	P0L.3	P0L.2	P0L.1	P0L.0
-	-	-	-	-	-	-	-	rw	Rw	rw	rw	rw	rw	rw	rw

SFR

P0H (FF02_H/81_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P0H.7	P0H.6	P0H.5	P0H.4	P0H.3	P0H.2	P0H.1	P0H.0
-	-	-	-	-	-	-	-	rw	Rw	rw	rw	rw	rw	rw	rw

Бит	Функция
P0X.y	Регистр данных бита y порта P0H или P0L

ESFR

DP0L (F100_H/80_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								DP0L.7	DP0L.6	DP0L.5	DP0L.4	DP0L.3	DP0L.2	DP0L.1	DP0L.0
-	-	-	-	-	-	-	-	rw	Rw	rw	rw	rw	rw	rw	rw

ESFR

DP0H (F102_H/81_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								DP0H.7	DP0H.6	DP0H.5	DP0H.4	DP0H.3	DP0H.2	DP0H.1	DP0H.0
-	-	-	-	-	-	-	-	rw	Rw	rw	rw	rw	rw	rw	rw

Бит	Функция
DP0X.y	Регистр направления порта DP0H или DP0L бита y DP0X.y = 0: линия порта P0X.y на ввод (высокое сопр.) DP0X.y = 1: линия порта P0X.y на вывод

Альтернативные функции порта 0

При включенной внешней шине порт 0 используется в качестве шины данных или в качестве шины адреса и данных.

Заметим, что внешняя 8-битная демультиплексная шина использует только P0L, в то время как P0H остается свободным для ИО (при условии, если не включен другой режим шины).

Порт 0 также используется для выбора конфигурации системы при старте. Во время RESET, порт 0 настроен на вход, и каждая линия выставлена на высокий уровень через внутреннее устройство pullup. Каждая линия может быть индивидуально подключена к нулевому потенциалу, с помощью внешнего pulldown устройства. Выставляя необходимые линии на низкий уровень напряжения, можно изменять установочные значения.

Внутреннее pullup устройство разработано подобно внешним pulldown резисторам, которые могут использоваться для подачи корректного низкого уровня напряжения. Внешние pulldown резисторы могут оставаться подсоединенными к порту 0 во время нормальной работы микроконтроллера, однако необходимо обращать внимание на то, чтобы их значения не мешали нормальному функционированию порта 0.

С окончанием RESET, выбранная конфигурация шины будет записана в регистр BUSCON0. Конфигурация старшего байта порта 0, копируется в специальном регистре RP0H. Этот регистр, предназначенный только для чтения, запоминает номер Chip select и адрес сегмента.

Когда завершается RESET, внутреннее устройство pullup выключается, и порт 0 переключается в обычный режим работы.

Во время режима внешнего доступа с помощью мультиплексной шиной, в порт 0 сначала выводится 16-битный внутрисегментный адрес. Затем порт 0 переключается в режим высокого входного сопротивления для чтения входных команд или данных. В 8-битном режиме шины данных, необходимо два цикла для доступа к слову, сначала для доступа к младшему байту и потом для доступа к старшему байту слова. Во время циклов записи порт 0 выводит байт данных или слово после вывода адреса.

Во время режима внешнего доступа в режиме демультиплексной шины, порт 0 читает входящую команду или слово данных или выводит байт данных или слово.

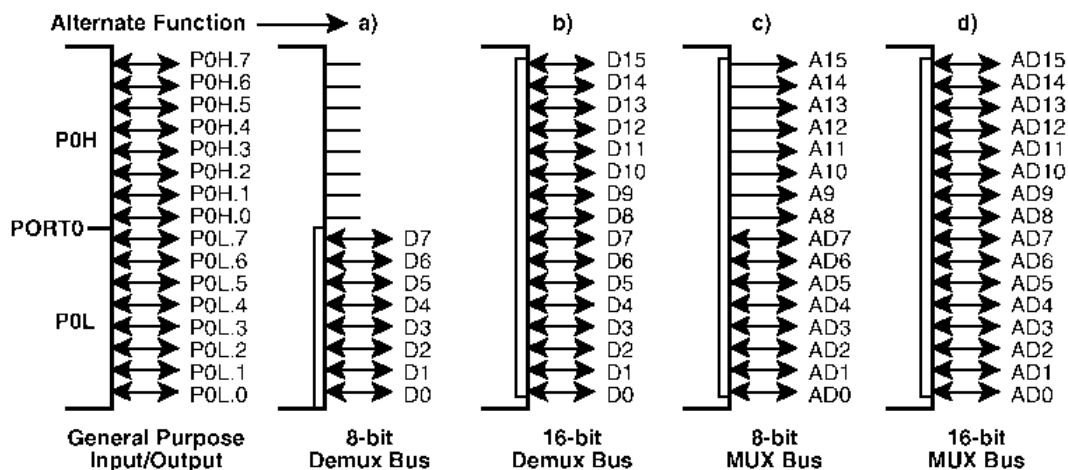


Рисунок 6-4

Порты IO и альтернативные функции

Когда включен режим внешней шины, направление вывода порта и загрузка данных в выходной триггер порта управляется с помощью блока контроллера шины. Вход порта и выходной триггер неподконтрольны внутренней шине и переключаются на линию, названную «альтернативный вывод данных» с помощью мультиплексора. Альтернативные данные могут быть либо 16-битным внутрисегментным адресом или 8/16-битными данными. Входные данные порта 0 читаются на линии «Альтернативный вход данных». В то время когда включен режим внешней шины, не должна производиться запись в выходной триггер порта пользовательской программой, иначе могут иметь место непредсказуемые результаты. Когда внешняя шина отключена вступает в силу последнее, записанное пользователем, содержимое регистра направления.

Рисунок, показанный ниже, показывает структуру выводов порта 0.

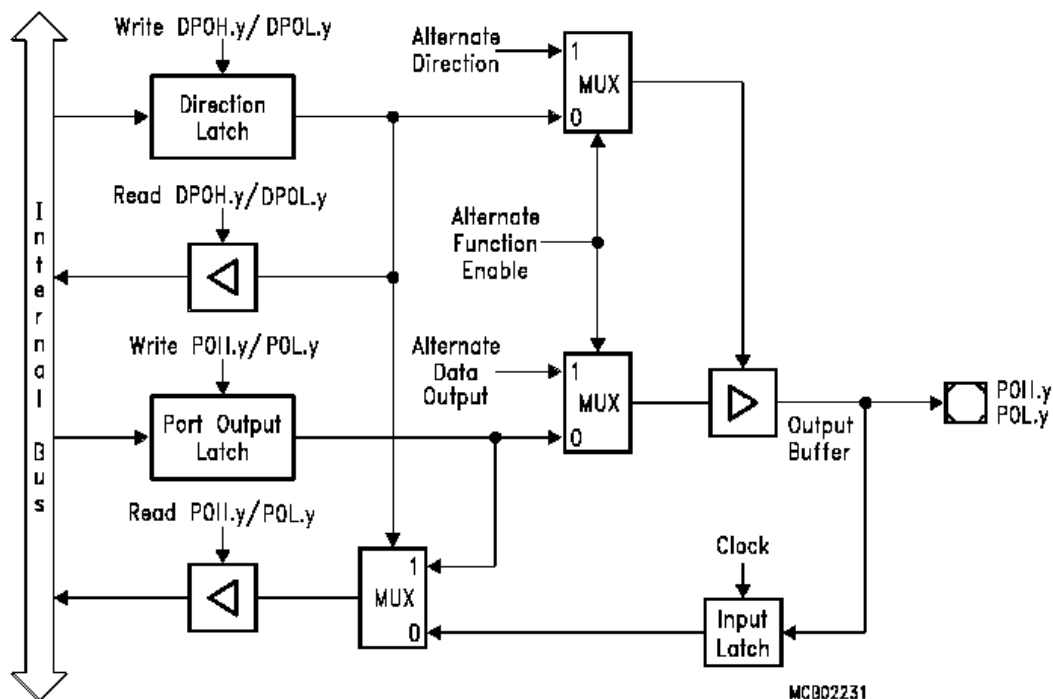


Рисунок 6-5

Блок-схема выводов порта 0

6.2 Порт 1

Два восьмибитных порта P1H и P1L представляют из себя две части порта 0. В обе половины порта 0 может быть произведена запись без изменения значения другой половины.

Если этот порт используется для ввода-вывода основного назначения, направление каждой линии может быть сконфигурировано с помощью регистров направления DP1H и DP1L.

SFR

P1L (FF04_H/82_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P1L.7	P1L.6	P1L.5	P1L.4	P1L.3	P1L.2	P1L.1	P1L.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

SFR

P1H (FF06_H/83_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P1H.7	P1H.6	P1H.5	P1H.4	P1H.3	P1H.2	P1H.1	P1H.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
P1X.y	Регистр данных бита у порта P1H или P1L

ESFR

DP1L (F104_H/82_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								DP1L.7	DP1L.6	DP1L.5	DP1L.4	DP1L.3	DP1L.2	DP1L.1	DP1L.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

ESFR

DP0H (F104_H/82_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								DP1H.7	DP1H.6	DP1H.5	DP1H.4	DP1H.3	DP1H.2	DP1H.1	DP1H.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
DP1X.y	Регистр направления порта DP1H или DP1L бита у DP1X.y = 0: линия порта P1X.y на ввод (высокое сопр.) DP1X.y = 1: линия порта P1X.y на вывод

Альтернативная функция порта 1

При включенном режиме демultipлексной внешней шины, порт 1 используется в качестве шины адреса.

Заметим, что в режиме демultipлексной шины используется порт 1, в качестве 16-битного порта. В ином случае все 16 линий порта используются для ввода-вывода основного назначения.

Старшие четыре вывода порта 1 (P1H.7 ... P1H.4) работают также в качестве входных линий захвата для блока CAPCOM2 (CC27IO ... CC24IO).

Как и все другие входы захвата, входы захвата P1H.7 ... P1H.4 могут быть использованы как входы внешних прерываний (время ответа 400нс 20МГц CPU clock).

Во время внешнего доступа в режиме демультиплексной шины, в качестве альтернативной функции порт 1 выводит 16-битный адрес внутри сегмента.

Во время внешнего доступа в режиме мультиплексной шины, порт 1 не используется и доступен для ввода-вывода основного назначения.

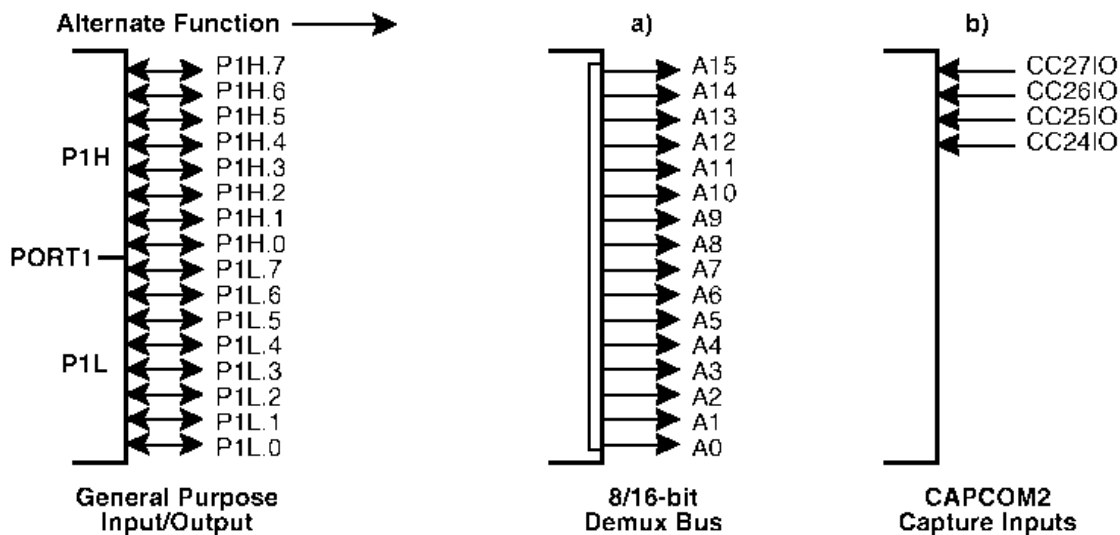


Рисунок 6-6

Ввод-вывод порта 1 и его альтернативные функции

При включенном режиме внешней шины, направление вывода порта и загрузка данных в выходной триггер порта управляется с помощью блока контроллера шины. Вход выходного триггера порта отключен от внутренней шины и переключен на линию под названием «Альтернативный вывод данных» через мультиплексор. Альтернативные данные представляют из себя 16-битный адрес внутри сегмента. Во время работы в режиме внешней шины, пользовательской программе не следует совершать запись в выходной триггер порта, иначе могут быть получены непредсказуемые результаты. Когда режим внешней шины отключается вступает в действие содержимое регистра направления, записанное пользователем.

Рисунок, предоставленный ниже, объясняет структуру порта 1.

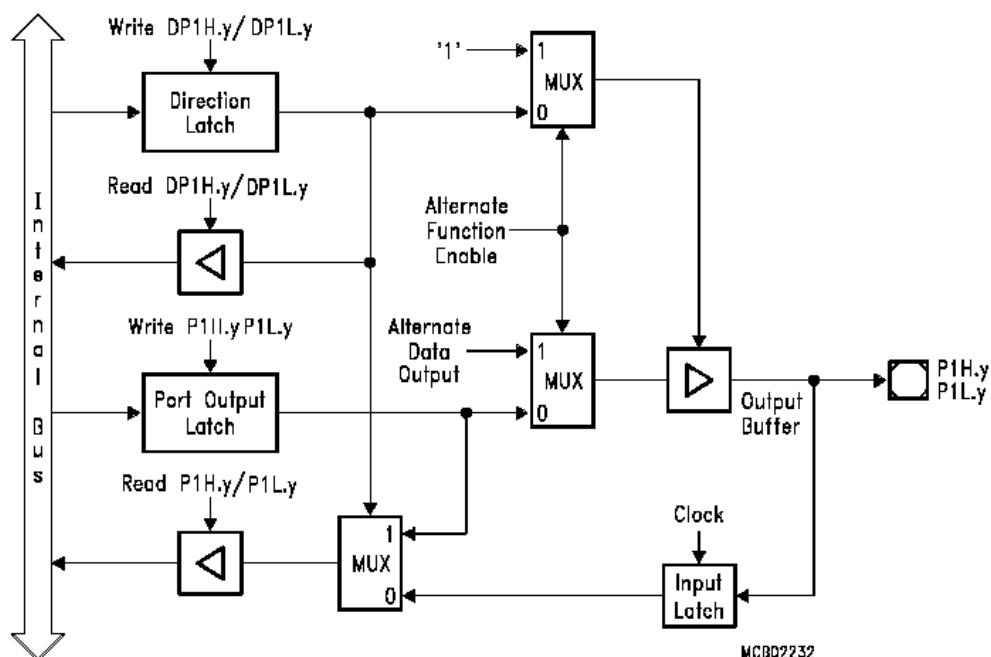


Рисунок 6-7

Блок-схема порта 1

6.3 Порт 2

Если этот 16-битный порт используется для ввода-вывода основного назначения, то направление каждой линии может быть сконфигурировано с помощью регистра направления DP2. Каждая линия порта может быть переключена либо в режим push/pull или в режим с открытым коллектором, с помощью регистра управления открытым коллектором ODP2.

SFR

P2 (FFC0_H/E0_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P2.15	P2.14	P2.13	P2.12	P2.11	P2.10	P2.9	P2.8	P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
P2.y	Бит y регистра порта данных P2

SFR

DP2 (FFC2_H/E1_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DP2 .15	DP2 .14	DP2 .13	DP2 .12	DP2 .11	DP2 .10	DP2 .9	DP2 .8	DP2 .7	DP2 .6	DP2 .5	DP2 .4	DP2 .3	DP2 .2	DP2 .1	DP2 .0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
DP2.y	<p>Бит у регистра направления порта DP2</p> <p>DP2.y = 0: линия порта P2.y на ввод (высокое сопр.)</p> <p>DP2.y = 1: линия порта P2.y на вывод</p>

ESFR

ODP2 (F1C2_H/E1_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODP2 .15	ODP2 .14	ODP2 .13	ODP2 .12	ODP2 .11	ODP2 .10	ODP2 .9	ODP2 .8	ODP2 .7	ODP2 .6	ODP2 .5	ODP2 .4	ODP2 .3	ODP2 .2	ODP2 .1	ODP2 .0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
ODP2.y	<p>Бит у регистра управления открытым коллектором порта DP2</p> <p>ODP2.y = 0: Выходной драйвер линии порта 2 P2.y в режиме push/pull</p> <p>ODP2.y = 1: Выходной драйвер линии порта 2 P2.y в режиме открытого коллектора</p>

Альтернативные функции порта 2

Все линии порта 2 (P2.15... P2.0) обслуживаются как входы захвата или выходы сравнения (CC15IO... CC0IO) блока CAPCOM1.

Когда линии порта 2 используются в качестве входов захвата, состояние входного триггера, показывающего состояние вывода порта, прямо подключено к CAPCOM блоку при помощи линии «Альтернативный вход данных». Если используется сигнал внешнего триггера захвата, направление вывода порта должно быть выставлено на вход. Если порт направлен на выход, то будет читаться состояние выходного триггера порта, поскольку вывод порта представляет состояние выходного триггера. Это может быть использовано для переключения триггера захватываемого события, с помощью программы, устанавливающей значение в выходном триггере порта. Заметим, что при настройке на вывод, не должно быть подключено

никаких внешних устройств, которые могут подать сигнал на вход, иначе может произойти конфликт.

Когда линии порта 2 используются в качестве выходов сравнения (режим сравнения 1 и 3), то сравниваемое событие (или переполнение таймера в режиме сравнения 3) прямо влияет на состояние выходного триггера порта. В режиме сравнения 1, когда происходит правильное соответствующее событие, имеет место чтение состояние выходного триггера порта посредством оборудования управления CAPCOM-блоком через линию «Альтернативный вход данных Latch», и наоборот запись назад в триггер с помощью линии «Альтернативный выход данных». Выходной триггер порта переключается по сигналу «Триггер сравнения», который генерируется CAPCOM-блоком. В режиме сравнения 3, при сравнении значений, в выходной триггер порта записывается «1» через линию «Альтернативный выход данных». Когда происходит переполнение соответствующего таймера, в выходной триггер порта записывается «0». В обоих случаях, выходной триггер is clocked посредством сигнала «триггер сравнения». Направление вывода порта должно быть установлено пользователем на вывод, иначе вывод порта будет в состоянии высокого сопротивления и не будет реагировать состояние выходного триггера.

Как можно увидеть из структуры порта, приведенной ниже, пользовательская программа имеет всегда свободный доступ к выводам порта, в том случае когда они используются в качестве выходов сравнения. Это используется для установки уровня инициализации вывода порта, когда используется режим сравнения 1 или режим двух регистров.

Когда пользователь хочет произвести запись в вывод порта, и одновременно с этим триггер сравнения пытается изменить значение выходного триггера, операция программного чтения имеет более высокий приоритет. Каждый раз, когда имеет место доступ ЦПУ для совершения записи в выходной триггер порта, входной мультиплексор выходного триггера порта переключается на линию, соединенную с внутренней шиной. Выходной триггер порта получает значение с внутренней шины, и при этом аппаратные изменения будут потеряны.

Как и для всех входов захвата, входная функция захвата для выводов P2.15 ... P2.0 может также быть использована для входов внешних прерываний (с временем ответа 400 нс при частоте 20МГц).

Старшие восемь линий порта 2 (P2.15... P2.8) также могут обслуживать входы быстрых внешних прерываний (EX7IN ... EX0IN).

В дополнение к этому P2.15 предназначен для входа таймера 7 CAPCOM2 (T7IN).

Таблица, приведенная ниже, суммирует альтернативные функции порта 2.

Выводы порта 2	Альтернативная функция ^{a)}	Альтернативная функция ^{b)}	Альтернативная функция ^{b)}
P2.0	CC0IO	-	-
P2.1	CC1IO	-	-
P2.2	CC2IO	-	-
P2.3	CC3IO	-	-
P2.4	CC4IO	-	-
P2.5	CC5IO	-	-
P2.6	CC6IO	-	-
P2.7	CC7IO	-	-
P2.8	CC8IO	EX0IN	-
P2.9	CC9IO	EX1IN	-
P2.10	CC10IO	EX2IN	-
P2.11	CC11IO	EX3IN	-
P2.12	CC12IO	EX4IN	-
P2.13	CC13IO	EX5IN	-
P2.14	CC14IO	EX6IN	-
P2.15	CC15IO	EX7IN	T7IN

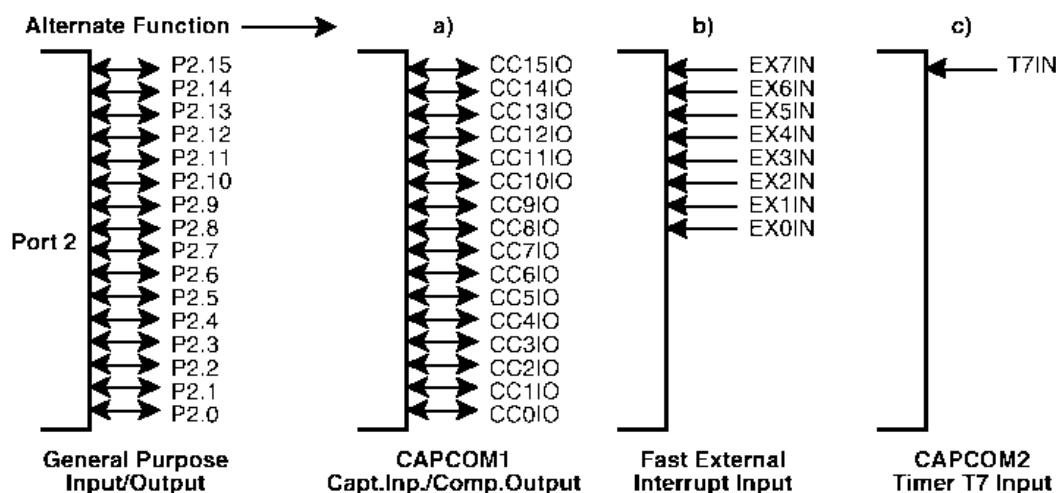


Рисунок 6-8

Функции ввода-вывода порта 2 и альтернативные функции

Выводы порта 2 объединяют данные внутренней шины и альтернативные данные, прежде чем подать их на вход триггера порта.

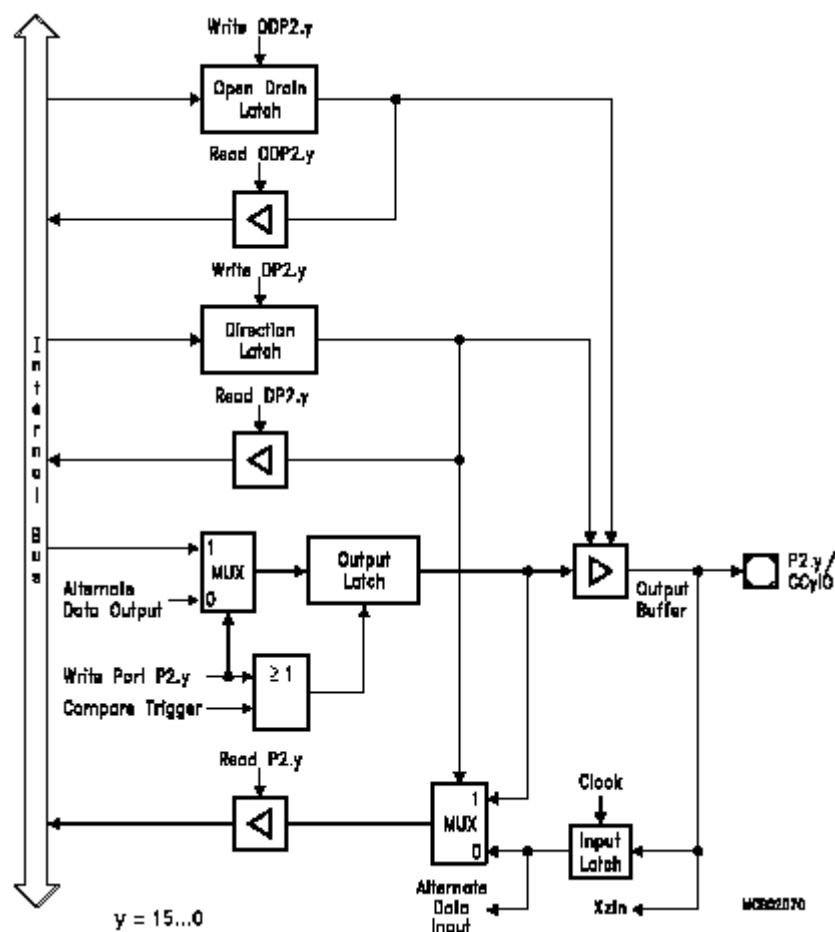


Рисунок 6-9

Блок-схема вывода порта 2

6.4 Порт 3

Если этот 15-битный порт использовать для ввода-вывода основного назначения, то направление каждой линии может быть изменено с помощью регистра направления DP3. Большинство линий порта может быть переключено в режим push/pull или режим с открытым коллектором, с помощью регистра управления открытым коллектором ODP2 (выводы P3.15, P3.14 и P3.12 не поддерживают режим с открытым коллектором!).

Бит P3.14 не подключен к выходу порта, что обусловлено ограниченным количеством выводов микросхемы.

SFR

P3 (FFC4_H/E2_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P3.15	-	P3.13	P3.12	P3.11	P3.10	P3.9	P3.8	P3.7	P3.6	P3.5	P3.4	P3.3	P3.2	P3.1	P3.0
rw	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
Бит		Функция													
P3.y		Бит y регистра порта данных P3													

SFR

DP3 (FFC6_H/E3_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DP3.15	-	DP3.13	DP3.12	DP3.11	DP3.10	DP3.9	DP3.8	DP3.7	DP3.6	DP3.5	DP3.4	DP3.3	DP3.2	DP3.1	DP3.0
rw	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
Бит		Функция													
DP3.y		Бит y регистра направления порта DP3 DP3.y = 0: линия порта P3.y на ввод (высокое сопр.) DP3.y = 1: линия порта P3.y на вывод													

ESFR

ODP3 (F1C6_H/E3_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	ODP3.13	-	ODP3.11	ODP3.10	ODP3.9	ODP3.8	ODP3.7	ODP3.6	ODP3.5	ODP3.4	ODP3.3	ODP3.2	ODP3.1	ODP3.0
-	-	rw	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
ODP3.y	Бит у регистра управления открытым коллектором порта DP3 ODP3.y = 0: Выходной драйвер линии порта 3 P3.y в режиме push/pull ODP3.y = 1: Выходной драйвер линии порта 3 P3.y в режиме открытого коллектора

Альтернативные функции порта 3

Выходы порта 3 предназначены для различных функций, включая линии управления внешним таймером, два последовательных интерфейса и линии управления $\overline{BHE} / \overline{WRH}$ и CLKOUT.

Выходы порта 3	Альтернативная функция	
P3.0	T0IN	Вход счета таймера 0 CAPCOM1
P3.1	T6OUT	Выход триггера таймера 6
P3.2	CAPIN	Вход захвата GPT2
P3.3	T3OUT	Выход триггера таймера 3
P3.4	T3EUD	Внешний up/down вход таймера 3
P3.5	T4IN	Вход счета таймера 4
P3.6	T3IN	Вход счета таймера 3
P3.7	T2IN	Вход счета таймера 2
P3.8	MRST	SSC master получение/slave передача
P3.9	MTSR	SSC master передача/slave получение
P3.10	TxD0	ASC0 выход передачи данных
P3.11	RxD0	ASC0 вход получения данных
P3.12	$\overline{BHE} / \overline{WRH}$	Разр. старшего байта/write high output
P3.13	SCLC	SSC shift clock вход/выход
P3.14	---	Нет вывода микросхемы
P3.15	CLKOUT	Выход системного тактового генератора

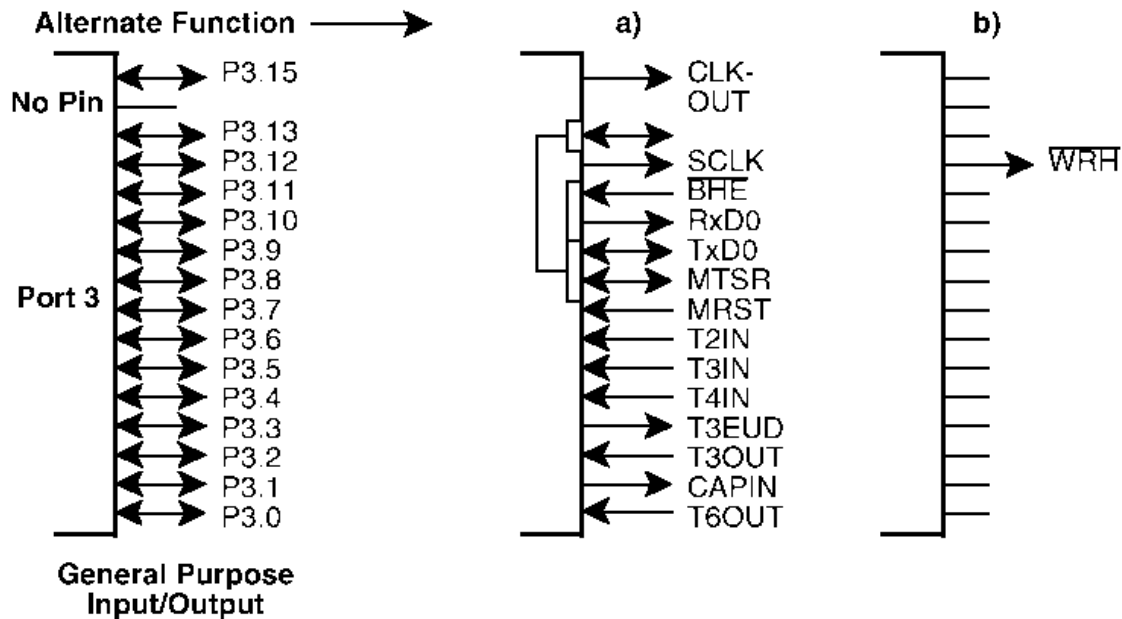


Рисунок 6-10

Функции ввода-вывода порта 3 и альтернативные функции

Структура выводов порта 3 зависит от альтернативных функций (см. рисунок, представленный ниже).

Когда внутренняя периферия, связанная с портом 3, настроена на использование функций альтернативного ввода, имеет место чтение входного триггера, который содержит значение на выводе микросхемы, через линии «Альтернативный ввод данных». Альтернативные функции ввода для порта 3 включают: T0IN, T2IN, T4IN, T3EUD и CAPIN.

Когда внутренняя периферия, связанная с портом 3, настроена на использования порта 3 для вывода, линия «Альтернативный вывод данных» и линия выходного триггера порта соединяются между собой по команде AND. При использовании этих альтернативных функций, пользователь обязан установить направление линии порта на вывод (DP3.y=1), и должен установить «1» в выходном триггере порта (P3.y=1). В ином случае выводы порта окажутся в состоянии высокого сопротивления (в случае конфигурирования на вход) или значение вывода будет постоянно «0» (в случае нулевого значения в выходном триггере порта). Когда альтернативная функция порта для вывода не используется, линия «альтернативный вывод данных» находится в неактивном состоянии, с высоким уровнем напряжения. Выводы порта 3, которые используют функции альтернативного вывода: T6OUT, T3OUT, TxD0 и CLKOUT.

Когда внутренняя периферия, связанная с портом 3, настроена на использование в качестве альтернативного ввода-вывода данных, текущий режим работы такой же, как для каждого из режимов по отдельности.

Выводы порта 3, которые используются для альтернативных функций ввода/вывода: MTSR, MRST, RxD0 и SCLK.

Примечание: Разрешение функции CLKOUT автоматически включает выходной драйвер для вывода P3.15. Не требуется установки единицы в бите DP3.15.

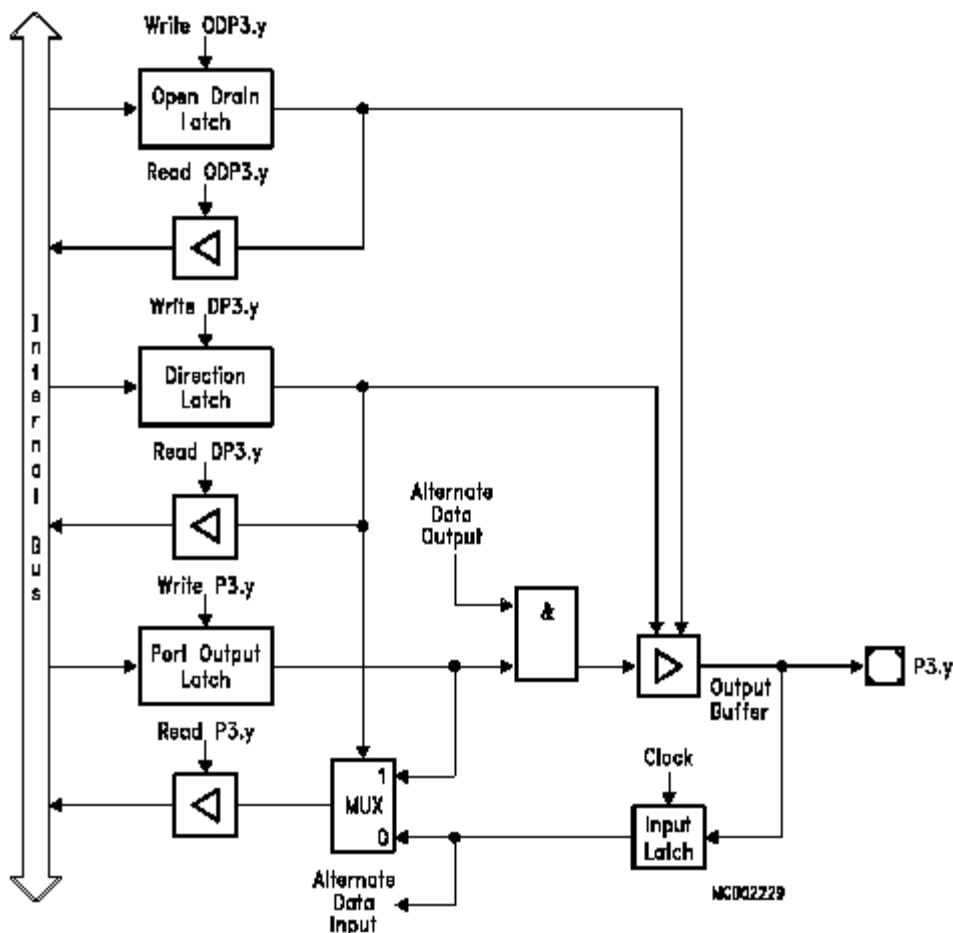


Рисунок 6-11

Блок-схема вывода порта 3 с функциями альтернативного ввода или вывода

Вывод P3.12 ($\overline{BHE}/\overline{WRH}$) является имеет еще одну альтернативную функцию вывода. Однако, его структура незначительно отличается (на рисунке ниже), потому что после RESET, функция \overline{BHE} или \overline{WRH} должны использоваться в зависимости от начальной конфигурации системы. В этих случаях, не представляется возможным, изменение до этого программой значения какого-либо триггера порта. Таким образом альтернативная функция выбирается автоматически. Если $\overline{BHE}/\overline{WRH}$ не используются в системе, то эти выводы могут использоваться в качестве линий портов ввода-вывода основного назначения, при отключенной альтернативной функции (BYTDIS=1/WRCFG=0).

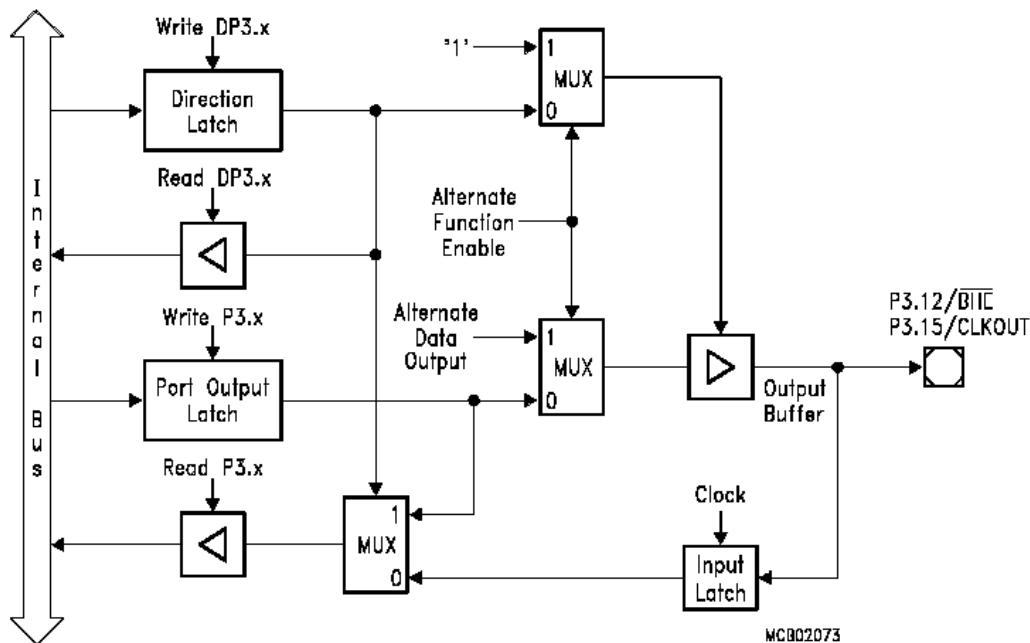


Рисунок 6-12

Блок-схема выводов P3.15 (CLKOUT) и P3.12 (\overline{BHE} / \overline{WRH})

Примечание: При включении функции \overline{BHE} или \overline{WRH} автоматически включается выходной драйвер вывода P3.12. При этом не требуется устанавливать «1» в бите DP3.12.

Во время захвата шины, вывод P3.12 переключается назад к стандартной функции, и затем управляется DP3.12 и P3.12. В этом случае гарантируется работа в режиме захвата.

6.5 Порт 4

При использовании этого порта для ввода-вывода основного назначения, направление каждой линии может быть установлено с помощью соответствующего бита регистра DP4.

SFR
P4 (FFC8_H/E4_H) **Значение после RESET: --00_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P4.7	P4.6	P4.5	P4.4	P4.3	P4.2	P4.1	P4.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
P4.y	Бит y регистра данных порта P4

ESFR
DP4 (FFCA_H/E5_H) **Значение после RESET: --00_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								DP4.7	DP4.6	DP4.5	DP4.4	DP4.3	DP4.2	DP4.1	DP4.0
-	-	-	-	-	-	-	-	ГW	ГW	ГW	ГW	ГW	ГW	ГW	ГW

Бит	Функция
DP4.y	Бит у регистра направления порта DP4 DP4.y = 0: линия порта P4.y на ввод (высокое сопр.) DP4.y = 1: линия порта P4.y на вывод

Альтернативные функции порта 4

При использовании режима сегментированной памяти, во время циклов работы внешней шины, некоторое количество выводов порта 4 может использоваться для вывода адреса сегмента. Количество выводов, использующихся для адреса сегмента, определяется внешним адресным пространством, которое может быть адресовано. Другие выводы порта 4 могут использоваться для ввода-вывода основного назначения. В случае выбора этих линии для сегментации адреса, альтернативная функция порта 4 необходима для получения прямого доступа после RESET. По этой причине порт 4 автоматически переключается на альтернативную функцию.

Количество линий сегментированного адреса выбирается через порт 0 во время RESET. Выбранное значение может быть прочтено из поля битов SALSEL регистра RP0H (только для чтения).

Микроконтроллеры с CAN-интерфейсом используют 2 вывода порта 4, чтобы обеспечить связь CAN-модуля с внешним CAN-передатчиком. В этом случае уменьшается количество возможных линий для сегментной адресации.

Таблица, приведенная ниже, суммирует альтернативные функции порта 4, зависящих от количества выбранных линий сегментной адресации.

Выводы порта 4	Станд. функция SALSEL=01 64Кбайта	Альт. функция SALSEL=11 256Кбайт	Альт. функция SALSEL=00 1Мбайт	Альт. функция SALSEL=10 16Мбайт
P4.0	IO осн. назнач.	A16	A16	A16
P4.1	IO осн. назнач.	A17	A17	A17
P4.2	IO осн. назнач.	IO осн. назнач.	A18	A18
P4.3	IO осн. назнач.	IO осн. назнач.	A19	A19
P4.4	IO осн. назнач.	IO осн. назнач.	IO осн. назнач.	A20
P4.5	IO осн. назнач.	IO осн. назнач.	IO осн. назнач.	A21
P4.6	IO осн. назнач.	IO осн. назнач.	IO осн. назнач.	A22
P4.7	IO осн. назнач.	IO осн. назнач.	IO осн. назнач.	A23

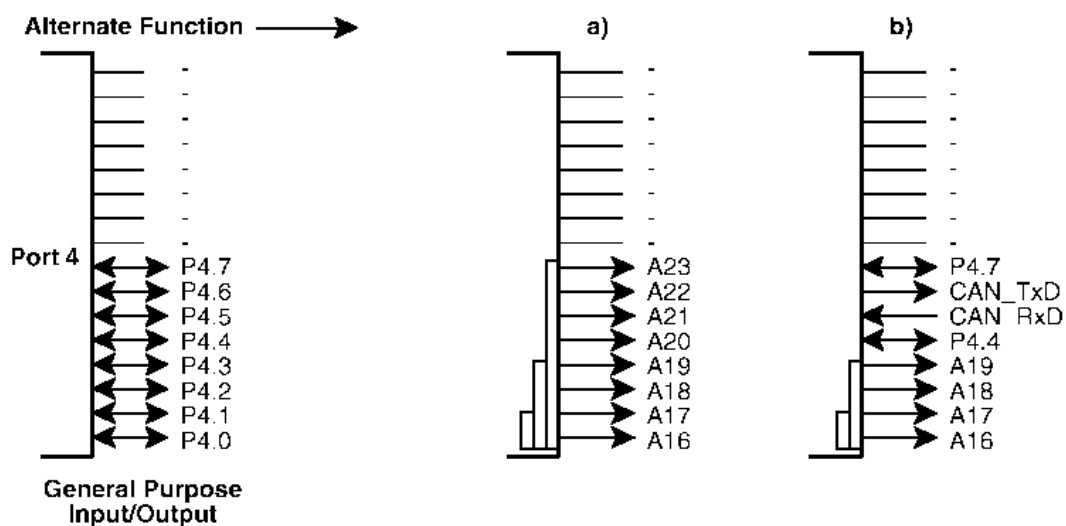


Рисунок 6-13

Функции ввода-вывода и альтернативные функции порта 4

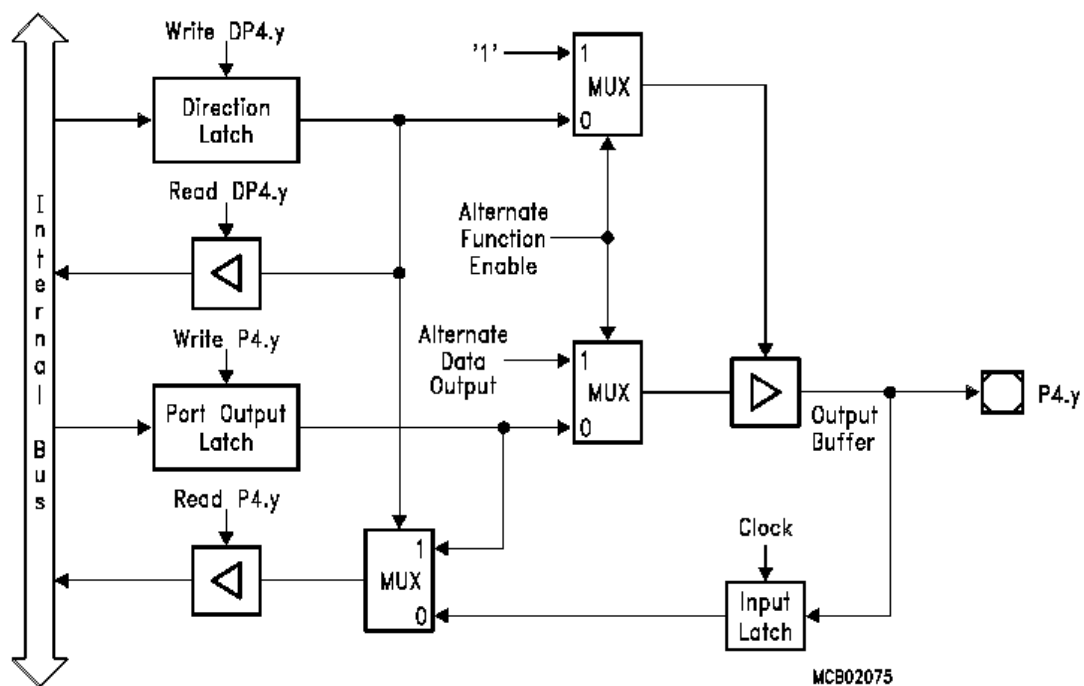


Рисунок 6-14

Блок-схема выводов порта 4

6.6 Порт 5

Этот 16-битный входной порт может только читать данные. Для этого порта нет ни выходного триггера, ни регистра направления. Данные, записанные в P5, будут потеряны.

SFR

P5 (FFA2_H/D1_H)

Значение после RESET: XXXX_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P5.15	P5.14	P5.13	P5.12	P5.11	P5.10	P5.9	P5.8	P5.7	P5.6	P5.5	P5.4	P5.3	P5.2	P5.1	P5.0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Бит	Функция
P5.y	Бит у регистра порта данных P5 (только для чтения)

Альтернативные функции порта 5

Каждая линия порта 5 подсоединена ко входному мультиплексору АЦП. Все линии портов (P5.15... P5.0) могут работать с аналоговыми сигналами (AN15... AN0), для АЦП. Старшие 6 входов порта 5 также могут работать в качестве линий управления внешними таймерами GPT1 и GPT2.

Таблица, представленная ниже, суммирует альтернативные функции порта 5.

Выводы порта 5	Альтернативная функция а)	Альтернативная функция б)	
P5.0	Аналоговый вход AN0	-	
P5.1	Аналоговый вход AN1	-	
P5.2	Аналоговый вход AN2	-	
P5.3	Аналоговый вход AN3	-	
P5.4	Аналоговый вход AN4	-	
P5.5	Аналоговый вход AN5	-	
P5.6	Аналоговый вход AN6	-	
P5.7	Аналоговый вход AN7	-	
P5.8	Аналоговый вход AN8	-	
P5.9	Аналоговый вход AN9	-	
P5.10	Аналоговый вход AN10	T6EUD	Up/down внешн. вход таймера 6
P5.11	Аналоговый вход AN11	T5EUD	Up/down внешн. вход таймера 5
P5.12	Аналоговый вход AN12	T6IN	Вход счета таймера 6
P5.13	Аналоговый вход AN13	T5IN	Вход счета таймера 5
P5.14	Аналоговый вход AN14	T4EUD	Up/down внешн. вход таймера 4
P5.15	Аналоговый вход AN15	T2EUD	Up/down внешн. вход таймера 2

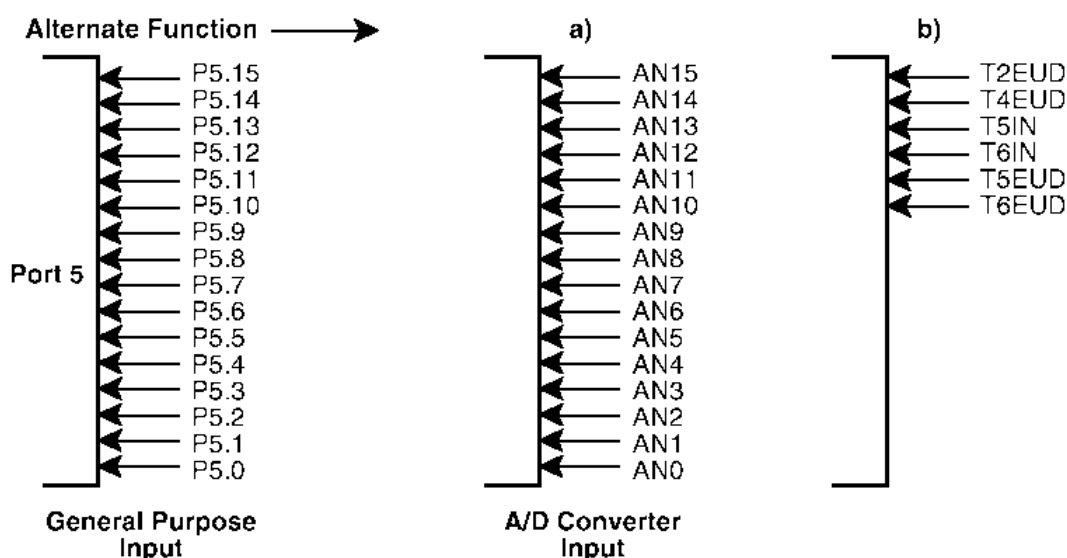


Рисунок 6-15

ИО порта 5 и его альтернативные функции

Выводы порта 5 имеют особую структуру (на рисунке ниже).

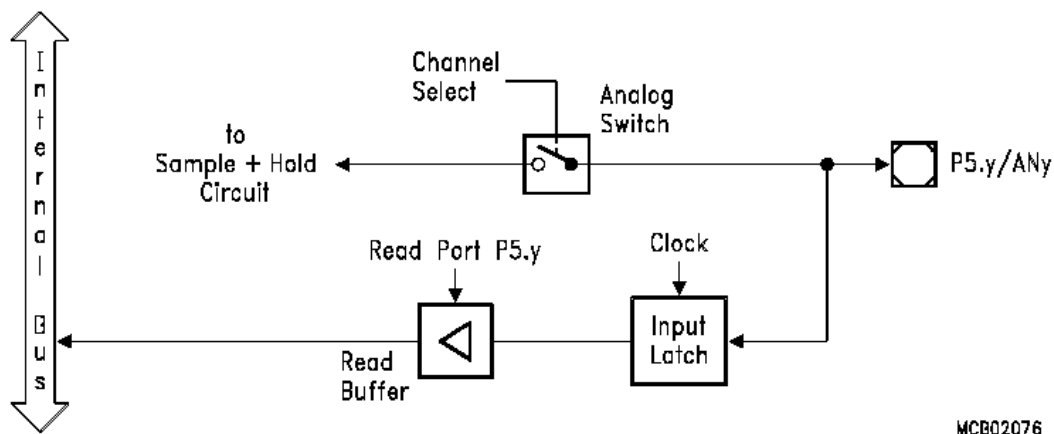


Рисунок 6-16

Блок-схема выводов порта 5

6.7 Порт 6

Если этот 8-битный порт используется для ввода-вывода основного назначения, то направление каждой линии может быть установлено с помощью регистра направления DP6. Каждая линия порта может быть переключена из push/pull режима в режим с открытым коллектором.

SFR

P6 (FFCC_H/E6_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P6.7	P6.6	P6.5	P6.4	P6.3	P6.2	P6.1	P6.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
P6.y	Бит y регистра данных порта P6

SFR

DP6 (FFCE_H/E7_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								DP6.7	DP6.6	DP6.5	DP6.4	DP6.3	DP6.2	DP6.1	DP6.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
DP6.y	Бит у регистра направления порта DP6 DP6.y = 0: линия порта P6.y на ввод (высокое сопр.) DP6.y = 1: линия порта P6.y на вывод

ESFR

ODP6 (F1C6_H/E3_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								ODP6.7	ODP6.6	ODP6.5	ODP6.4	ODP6.3	ODP6.2	ODP6.1	ODP6.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
ODP6.y	Бит у регистра управления открытым коллектором порта DP6 ODP6.y = 0: Выходной драйвер линии порта 6 P6.y в режиме push/pull ODP6.y = 1: Выходной драйвер линии порта 6 P6.y в режиме открытого коллектора

Альтернативные функции порта 6

Сигналы chip select (CS0... CS4), управляемых регистром управления шиной (BUSCON4... BUSCON0), могут быть выведены на 5 выводов порта 6. Другие 3 вывода могут использоваться для арбитража шины для согласования дополнительных masters в системе C167.

Количество сигналов chip select выбирается во время RESET с помощью порта 0. Выбранное значение может быть прочтено из битового поля CSSEL регистра RP0H (только для чтения).

Таблица, приведенная ниже, суммирует альтернативные функции порта 6, зависящие от выбранного числа линий chip select.

Выводы порта 6	Альт. функц. CSSEL = 10	Альт. функц. CSSEL = 01	Альт. функц. CSSEL = 00	Альт. функц. CSSEL = 11
P6.0	IO осн. назн.	CS0	CS0	CS0
P6.1	IO осн. назн.	CS1	CS1	CS1
P6.2	IO осн. назн.	IO осн. назн.	CS2	CS2
P6.3	IO осн. назн.	IO осн. назн.	IO осн. назн.	CS3
P6.4	IO осн. назн.	IO осн. назн.	IO осн. назн.	CS4
P6.5	\overline{HOLD}	Вход внешнего запроса на захват		
P6.6	\overline{HLDA}	Выход подтверждения захвата		
P6.7	\overline{BREQ}	Выход запроса шины		

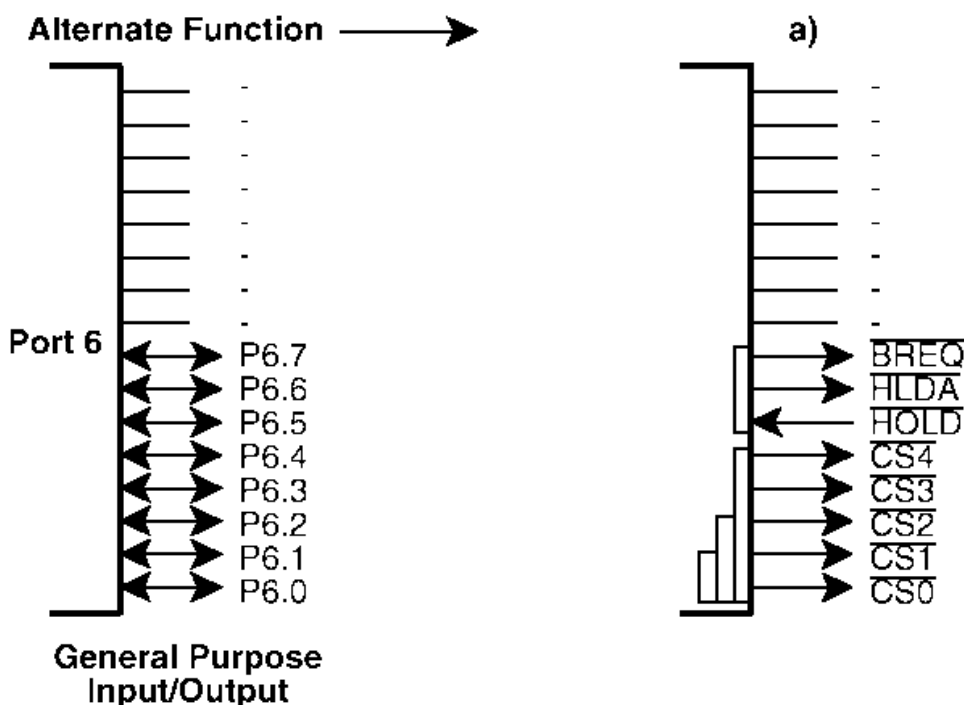


Рисунок 6-17

Функции порта 6

Линии chip select порта 6 имеют дополнительно устройство weak pullup. Это устройство включается в следующих случаях:

- Всегда во время RESET
- Если микроконтроллер находится в режиме захвата (вызванного сигналом \overline{HOLD}), в случае использования порта 6 для вывода сигналов chip select, при этом используемые для этого выводы порта находятся в режиме push/pull ($ODP6.x = \text{«0»}$).

Эта функция добавлена во избежания выбора одновременно нескольких внешних устройств. Она позволяет устанавливать высокий уровень напряжения на выходах chip select во время RESET, и также

позволяет другим masters получить доступ к внешней памяти через эти же линии chip select (wired-AND), во время нахождения C167 в режиме захвата.

В режиме с открытыми коллекторами, внутреннее устройство pullup не является активным в течении режима захвата, в этом случае необходимо использовать внешние устройства pullup.

При введении режима захвата, на CS сигналы на один такт подается высокий уровень, затем выходной уровень контролируется устройством pullup (в том случае, если оно активировано).

Примечание: Режим выходов с открытыми коллекторами может быть выбран во время подпрограммы инициализации, как минимум постоянно после RESET сигнал CS0 должен быть в режиме push/pull.

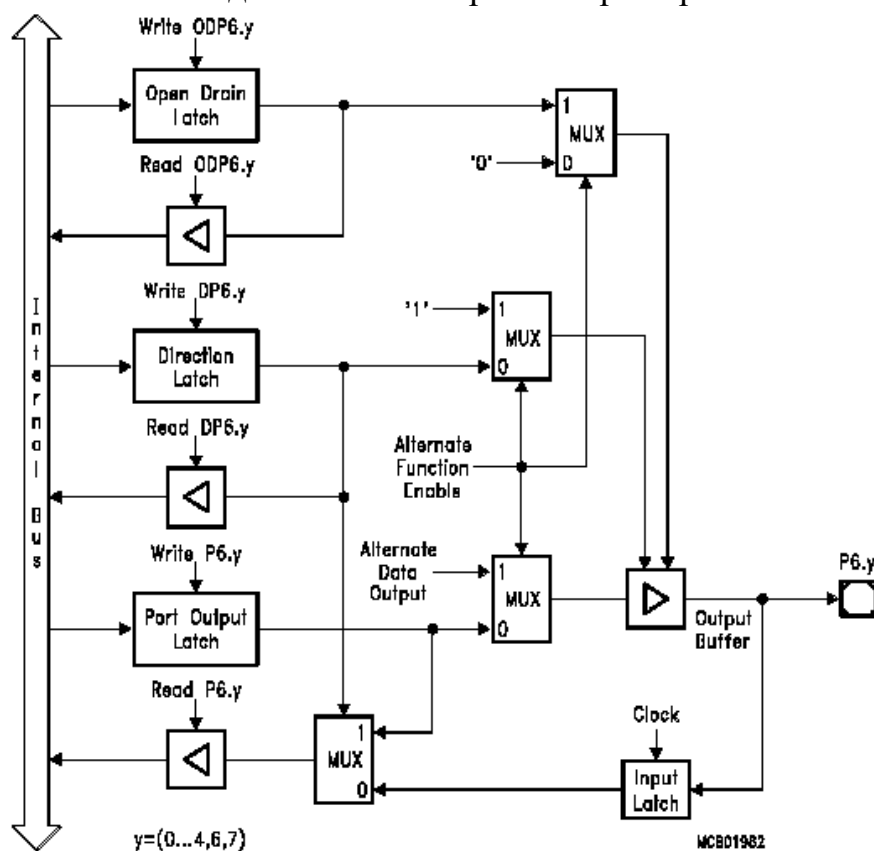


Рисунок 6-18

Блок схема выводов порта 6 с альтернативной функцией вывода

Сигналы арбитража шины \overline{HOLD} , \overline{HLDA} и \overline{BREQ} выбираются в бите HLDEN регистра PSW. При включенных сигналах арбитража шины, необходимое направление этих выводов порта устанавливается автоматически. Заметим что драйвера выводов \overline{HLDA} и \overline{BREQ} автоматически включаются, в то время как драйвер \overline{HOLD} автоматически выключается.

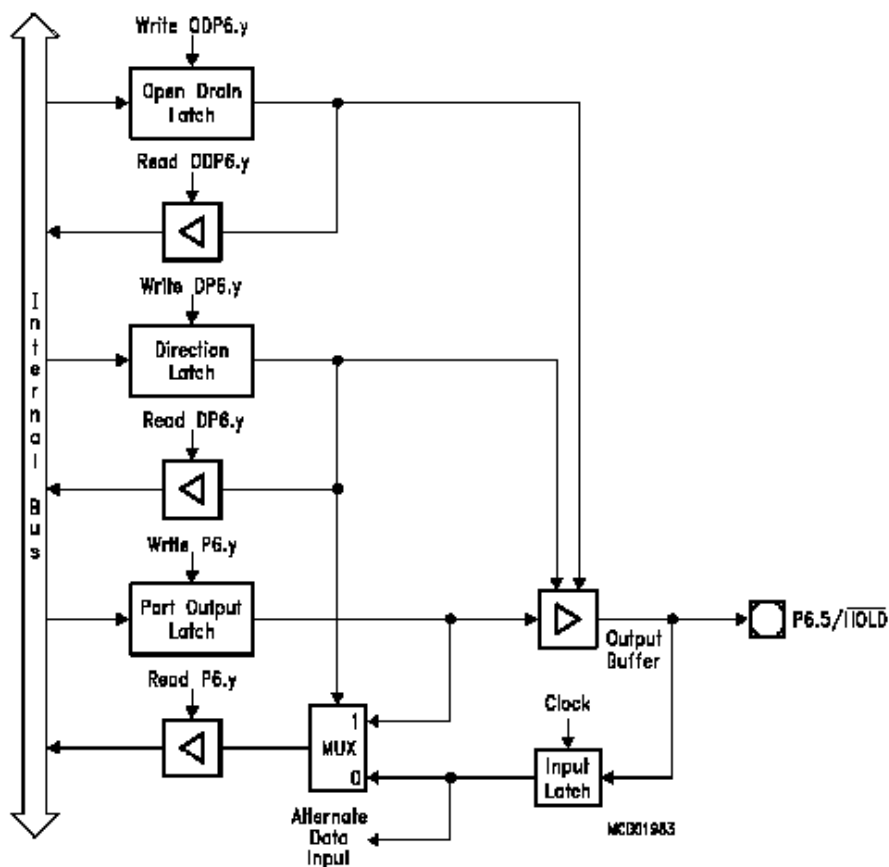


Рисунок 6-19

Блок-схема вывода P6.5 (\overline{HOLD})

6.8 Порт 7

При использовании этого 8-битного порта для ввода-вывода основного назначения, направление каждой линии можно настроить с помощью соответствующего регистра направления DP7. Каждая линия порта может быть переключена в режим push/pull или в режим с открытым коллектором.

SFR

P7 (FFD0_H/E8_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P7.7	P7.6	P7.5	P7.4	P7.3	P7.2	P7.1	P7.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
P7.y	Бит у регистра данных порта P7

SFR

DP7 (FFD2_H/E9_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								DP7.7	DP7.6	DP7.5	DP7.4	DP7.3	DP7.2	DP7.1	DP7.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
DP7.y	Бит у регистра направления порта DP7 DP7.y = 0: линия порта P7.y на ввод (высокое сопр.) DP7.y = 1: линия порта P7.y на вывод

ESFR

ODP7 (F1D2_H/E9_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								ODP7.7	ODP7.6	ODP7.5	ODP7.4	ODP7.3	ODP7.2	ODP7.1	ODP7.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
ODP7.y	Бит у регистра управления открытым коллектором порта DP7 ODP7.y = 0: Выходной драйвер линии порта 7 P7.y в режиме push/pull ODP7.y = 1: Выходной драйвер линии порта 7 P7.y в режиме открытого коллектора

Альтернативные функции порта 7

Старшие 4 линии порта 7 (P7.7... P7.4) могут использоваться в качестве входов захвата или выходов сравнения (CC31IO... CC28IO) блока CAPCOM2.

Процесс обслуживания линий порта для блока CAPCOM, программный доступ у нему и меры предосторожности аналогичны описанным для порта 2.

Как и все другие входы захваты, входы захвата P7.7... P7.0 могут также быть использованы для входа внешних прерываний.

Младшие 4 бита порта 7 (P7.3... P7.0) могут быть предназначены для вывода сигналов ШИМ-модуля (POUT3... POUT0). Значение на выводе порта представляет из себя результат операции XOR между значениями в выходном триггере порта и сигналом выхода ШИМ-модуля. Это позволяет использовать значение альтернативного выхода либо без изменений (т.е. в выходном триггере «0»), либо инвертируя выходное значение (выходной триггер в «1»).

Заметим, что вывод результатов из ШИМ-модуля включается, с помощью битов PENx в PWMCON1.

Таблица суммирует альтернативные функции порта 7.

Выводы порта 7	Альтернативные функции	
P7.0	POUT0	Выход канала 0 ШИМ-модуля
P7.1	POUT1	Выход канала 1 ШИМ-модуля
P7.2	POUT2	Выход канала 2 ШИМ-модуля
P7.3	POUT3	Выход канала 3 ШИМ-модуля
P7.4	CC28IO	CAPCOM канал 28
P7.5	CC29IO	CAPCOM канал 29
P7.6	CC30IO	CAPCOM канал 30
P7.7	CC31IO	CAPCOM канал 31

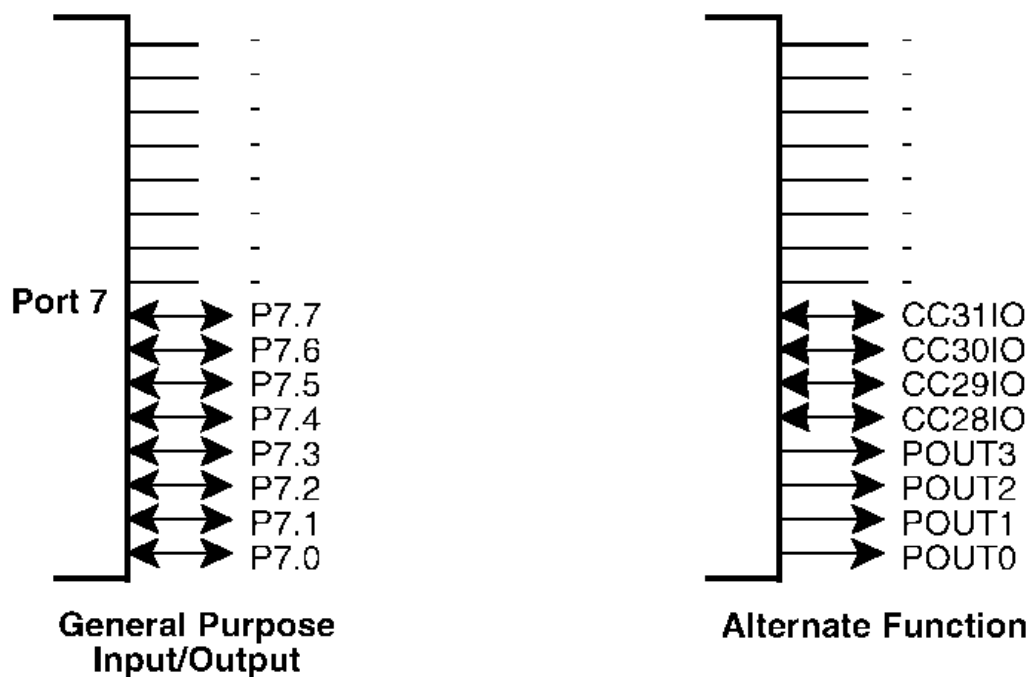


Рисунок 6-20

Функции порта 7

Когда выходной триггер подключен к внешней шине, структура порта отличается от случая, когда выходной триггер подключен к драйверам выхода (смотри на двух рисунках, предоставленных ниже).

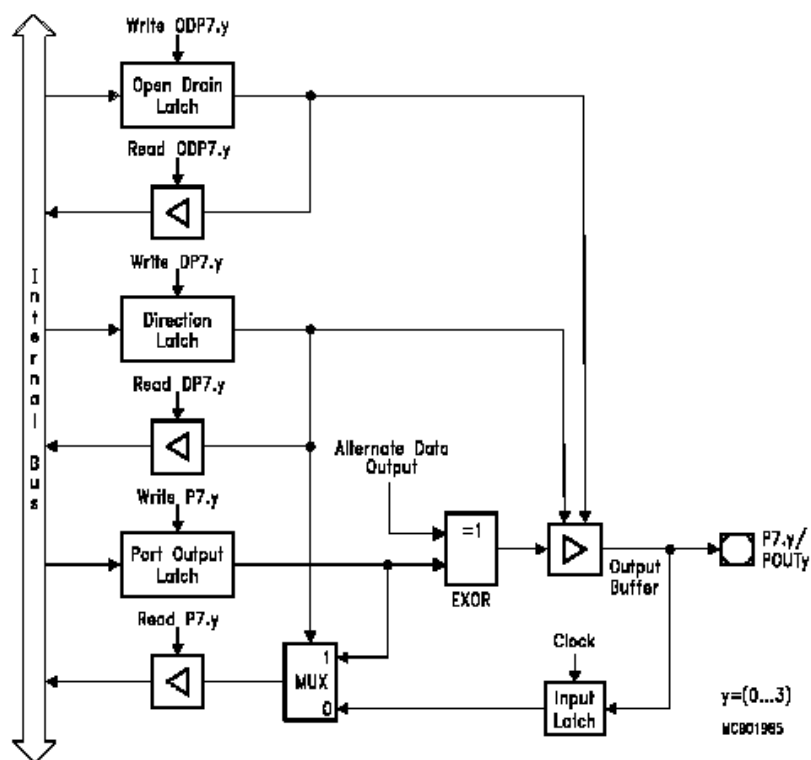


Рисунок 6-21

Блок схема выводов P7.3... P7.0 порта 7

Выводы P7.7... P7.4 (CC31IO... CC28IO) объединяют в себе вывод данных внешней шины и вывод альтернативных данных, перед записью данных в триггер порта, аналогично выводам порта 2.

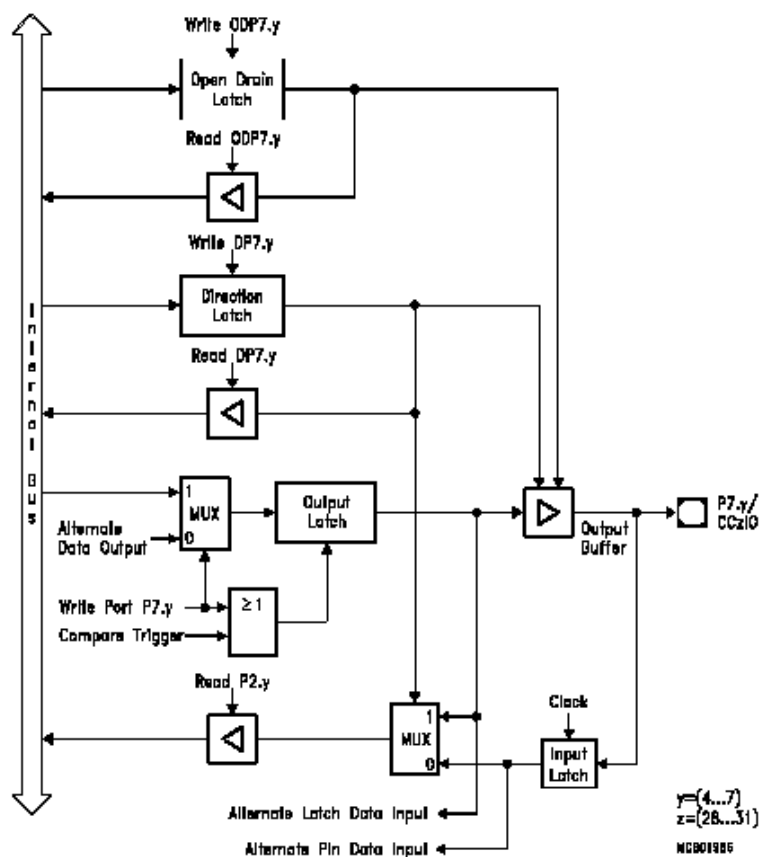


Рисунок 6-22

Блок-схема выводов P7.7... P7.4 порта 7

6.9 Порт 8

При использовании этого порта для ввода-вывода основного назначения, можно настроить направление каждой линии с помощью регистра DP8. Каждая линия порта может быть переключена в режим push/pull или режим с открытым коллектором.

SFR

P8 (FFD4_H/EA_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P8.7	P8.6	P8.5	P8.4	P8.3	P8.2	P8.1	P8.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
P8.y	Бит y регистра данных порта P8

SFR

DP8 (FFD6_H/EB_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								DP8.7	DP8.6	DP8.5	DP8.4	DP8.3	DP8.2	DP8.1	DP8.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
DP8.y	Бит y регистра направления порта DP8 DP8.y = 0: линия порта P8.y на ввод (высокое сопр.) DP8.y = 1: линия порта P8.y на вывод

ESFR

ODP8 (F1D6_H/EB_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								ODP8	ODP8	ODP8	ODP8	ODP8	ODP8	ODP8	ODP8
								.7	.6	.5	.4	.3	.2	.1	.0
-	-	-	-	-	-	-	-	ГW	ГW	ГW	ГW	ГW	ГW	ГW	ГW

Бит	Функция
ODP8.y	<p>Бит у регистра управления открытым коллектором порта DP8</p> <p>ODP8.y = 0: Выходной драйвер линии порта 8 P8.y в режиме push/pull</p> <p>ODP8.y = 1: Выходной драйвер линии порта 8 P8.y в режиме открытого коллектора</p>

Альтернативная функция порта 8

Все линии порта предназначены для выводов (CC23IO... CC16IO) CAPCOM2.

Все функции и блок-схемы аналогичны, представленным для порта 2.

7 Специальные выводы микроконтроллера

Большинство функций, использующих ввод-вывод данных, а также функции управления необходимые для работы C167 реализованы в виде альтернативных функций параллельных портов. Однако для некоторых сигналов предназначены независимые выводы. К ним можно отнести входы осциллятора, входы специальных сигналов управления и входы питания микросхемы.

Вывод	Функция
ALE	Включение latch адреса
\overline{RD}	Стробирующий сигнал чтения внешней периферии
$\overline{WR} / \overline{WRL}$	Стробирующий сигнал записи для внешней периферии / записи младшего байта
\overline{READY}	Вход готовности
\overline{EA}	Включение внешнего доступа
\overline{NMI}	Вход не маскируемого прерывания
XTAL1, XTAL2	Вход/выход осциллятора
\overline{RSTIN}	Вход RESET
\overline{RSTOUT}	Выход RESET
VAREF, VAGND	Питание для АЦП
VPP	Зарезервирован для программирования flash
VCC, VSS	Питание и нулевой потенциал (по 10 выводов для каждого)

ALE предназначен для управления внешними адресными latches, что обеспечивает стабильные значения адреса в режиме мультиплексной шины.

ALE активируется во время каждого цикла внешней шины, независимо от выбранного режима шины. Также он активируется во время каждого цикла внешней шины в режиме демультиплексной шины. При включенной внешней шине (один или больше битов BUSACT установлено в «1»), X-периферия также будет выставлять активный уровень напряжения ALE-сигнала.

ALE не активируется для внутреннего доступа, т.е. доступа к ROM или Flash, к внутренней RAM и к SFR-регистрам. В режиме одного микроконтроллера, то есть при не включенной внешней шине (не установлен не один бит BUSACT), сигнал ALE не активируется при доступе к X-периферии.

Стробирующий сигнал чтения внешней периферии \overline{RD} управляет выходными драйверами внешней памяти или периферии, при чтении данных из внешних устройств. Во время RESET и режима захвата, внутренний pullup гарантирует неактивный уровень напряжения на выходе этого сигнала

(«1»). Во время доступа к внутренней X-периферии этот сигнал также остается на неактивном уровне.

Стробирующий сигнал записи во внешнюю периферию $\overline{WR}/\overline{WRL}$ управляет передачей данных из C167 во внешнюю память или периферийное устройство. Этот вывод может генерировать сигнал \overline{WR} как для доступа к байту, так и для доступа к слову данных. Возможно переопределение функции сигнала, для управления доступом к младшему байту 16-битного устройства (\overline{WRL}). При этом для доступа к старшему байту будет использоваться сигнал \overline{WRH} (альтернативная функция P3.12/ \overline{BHE}). Во время RESET и режима захвата внутренний pullup гарантирует неактивный (высокий) уровень напряжения на выходе $\overline{WR}/\overline{WRL}$. Во время доступа к внутренней X-периферии $\overline{WR}/\overline{WRL}$ также устанавливается на неактивный уровень напряжения.

Вход готовности \overline{READY} обрабатывает сигналы управления от внешней памяти или периферийного внешнего оборудования. Этот сигнал может использоваться для завершения цикла внешней шины. \overline{READY} может быть настроен на работу как в синхронном, так и в асинхронном режиме. Вход \overline{READY} не используется во время waitstates, в тех случаях когда waitstates используются для подконтрольного сигналу \overline{READY} окна адресов.

Вывод разрешения внешнего доступа \overline{EA} определяет режим работы. При $\overline{EA}=1$, C167 после RESET вызывает код из области внутренней ROM, если $\overline{EA}=0$ то C167 начинает использоваться код, вызванный через внешнюю шину. Для микроконтроллеров без внутренней ROM следует уделять особое внимание значению на этом выводе во время RESET.

Вход не маскируемого прерывания \overline{NMI} позволяет вызывать ловушку с самым высоким приоритетом (сигнал ошибки питания). Этот вход также необходим для подтверждения действительности команды PWRDN, переключающей C167 в режим отключения питания. Значение на входе \overline{NMI} проверяется каждый такт ЦПУ.

Вход осциллятора XTAL1 и выход XTAL2 необходимы для подключения внутреннего Pierce осциллятора к внешнему кристаллу. Осциллятор включает в себя инвертор и элемент обратной связи. Стандартная схема внешнего осциллятора (на рисунке ниже) включает в себя кристалл, два прецизионных конденсатора и резистор для ограничения тока через кристалл. Дополнительная LC-схема требуется только для кристалла 3rd overtone, чтобы подавлять осцилляцию в основном режиме. Для измерения допусков частоты в схеме осциллятора может быть временно добавлен тестовый резистор R_Q .

Внешний тактовый сигнал поступает на вход XTAL1, и выходит из выхода XTAL2.

Примечание: Для определения оптимальных параметров для работы осциллятора, рекомендуется измерять допуски частоты для готовых систем.

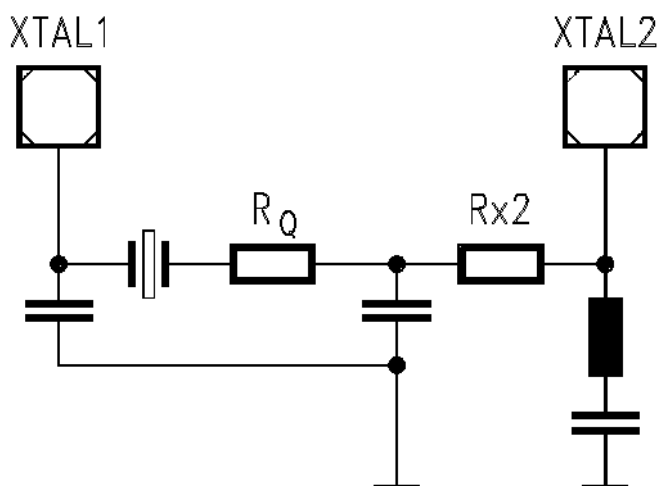


Рисунок 7-1

Схема внешнего осциллятора

Вход RESET \overline{RSTIN} позволяет начать процедуру RESET микроконтроллера. Пороговый уровень обнаружения фронта сигнала для \overline{RSTIN} по сравнению с другими выводами повышен. Это необходимо для минимизации влияния шума на сигнал на входе RESET.

Выход RESET \overline{RSTOUT} подает сигнал RESET для сброса внешней периферии. \overline{RSTOUT} переходит на активный уровень в начале процедуры RESET. \overline{RSTOUT} остается активным до тех пор, пока не будет выполнена команда EINIT. Это позволяет настроить контроллер перед включением внешней периферии.

Выводы питания АЦП VAREF и VAGND предназначены для независимого питания внутреннего АЦП. При этом уменьшается шум, вызываемый аналоговыми входными сигналами, и тем самым улучшается стабильность результата преобразования. Для достижения более точного результата преобразования необходимо наличие собственных фильтров питания для входов VAREF и VAGND от VCC и VSS.

Вход напряжения для программирования flash VPP обеспечивает уровень напряжения необходимый для программирования и очистки внутренней области Flash-памяти. Во время нормальной работы (за исключением очистки и программирования) необходимо этот вывод подключать к VCC. Для микроконтроллеров без Flash-памяти вывод VPP зарезервирован. В этом случае он может быть оставлен не подключенным или быть подключенным к VCC, что может потребоваться для обеспечения совместимости с микроконтроллерами с Flash-памятью.

Выводы питания VCC и VSS обеспечивают питание цифровой логики C167. Напряжение на входах каждой пары VCC/VSS должно быть максимально отфильтровано. Для достижения лучшего результата

рекомендуется использовать двухуровневую фильтрацию, с расположенными как можно ближе, конденсатором 100нФ и с параллельным конденсатором 30... 40 пФ, необходимым для сглаживания пиков тока.

Примечание: Все выводы питания и земли должны быть в обязательном порядке подключены к потенциалу питания и к потенциалу земли.

8 Интерфейс внешней шины

Несмотря на то, что C167 обеспечивает большой объем внутренней RAM и ROM (за исключением моделей без ROM), используемое адресное пространство для внутренних модулей памяти занимает лишь малую часть 16-Мбайтного адресного пространства. Интерфейс внешней шины позволяет получать доступ к внешней периферии и к дополнительной как энергонезависимой, так и не энергонезависимой памяти. Интерфейс внешней шины позволяет конфигурировать шину в широком диапазоне временных характеристик, поэтому возможно использование устройств с практически любыми характеристиками.

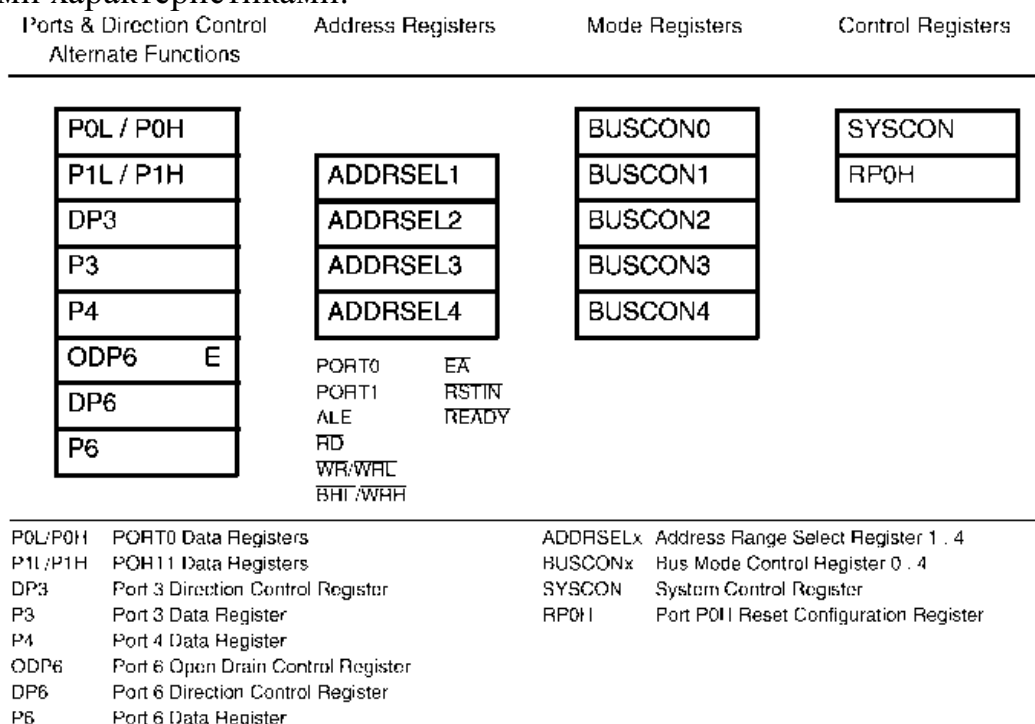


Рисунок 8-1

SFR-регистры и выводы портов интерфейса внешней шины

Доступ к внешней памяти или периферии осуществляется посредством интегрированного контроллера внешней шины (EBC). Управление контроллером EBC осуществляется через регистры SYSCON, BUSCONx и ADDRSELx. В регистрах BUSCONx осуществляется выбор режима цикла работы внешней шины: шина адреса (мультиплексная / демультиплексная), шина данных (16-разр./8-разр.), сигнал chip select и продолжительность цикла (waitstates / управление READY / ALE / RW задержка). Доступ к области адресов, обозначенных при помощи регистра ADDRSELx, осуществляется по параметрам регистров BUSCONx.

Четыре пары регистров BUSCON1/ADDRSEL1... BUSCON4/ADDRSEL4 позволяют предопределять четыре независимые «адресные окна». Доступ к всем адресам вне этих окон осуществляется при помощи регистра BUSCON0.

Режим работы микроконтроллера без внешней периферии (Single Chip Mode)

В данный режим можно войти при подачи на вход \overline{EA} высокого уровня напряжения во время RESET. В этом случае значение регистра BUSCON0 устанавливается на 0000_H, при этом значение бита BUSACT0 сбрасывается, и поэтому внешняя шина не используется.

В режиме без внешней периферии микроконтроллер работает только с внутренними ресурсами. Линии выводов портов не настроены для работы в режиме внешней шины, и поэтому не доступна внешняя память и периферия. При работе в режиме микроконтроллера без внешней периферии, возможно использование внешнего доступа посредством программного управления доступа к внешней шине.

Примечание: Любые попытки доступа в этом режиме к внешнему адресному пространству приводят к аппаратной ловушке ILLBUS.

8.1 Режимы внешней шины

После включения интерфейса внешней шины (BUSACTx=1) и его конфигурирования (в битовом поле BTYP), для построения внешней шины C167 использует линии портов в качестве линий управления.

BTYP	Ширина внешней шины данных	Внешняя шина данных
0 0	8-разрядная	демультиплексный адрес
0 1	8-разрядная	мультиплексный адрес
1 0	16-разрядная	демультиплексный адрес
1 1	16-разрядная	мультиплексный адрес

Конфигурация шины (в поле BTYP) для адресного окна (BUSCON4... BUSCON1) устанавливается программно, обычно во время инициализации системы.

Конфигурация шины (BTYP) для предустановленных адресных границ (BUSCON0) устанавливается в соответствии с сигналами на входах порта 0 во время процедуры RESET, при этом необходимо обеспечить на выводе \overline{EA} низкий уровень напряжения. В ином случае BUSCON0 может быть запрограммирован только программным образом.

16Мбайт адресного пространства C167 делятся на 256 сегментов по 64 Кбайта в каждом. 16-разрядный внутрисегментный адрес выводится через

порт 0 в режиме мультиплексной шины или через порт 1 в режиме демультимплексной шины. При отключенной сегментации может использоваться только один 64-кбайтный сегмент. Дополнительные линии адреса могут быть выведены через порт 4. Для использования различных банков памяти или периферии может быть задействовано несколько сигналов chip select. Работу в этом режиме можно выбрать во время RESET путем установки сигналов на входах битовых полей SALSEL и CSSEL порта RP0H.

Примечание: Значение бита SGTDIS регистра SYSCON определяет необходимость сохранения значения регистра CSP во время прерывания. В случае использования доступа к сегментированной памяти необходимо сохранять значение регистра CSP в стеке.

Режимы с мультиплексной шиной

В режиме с мультиплексной шиной при передаче 16-разрядного адреса и 16-разрядных данных используется порт 0. Адрес вместе с данными мультиплексируется и запирается в выходном latch. Ширина необходимого latch зависит от ширины выбранной шины данных, т.е. для 8-разрядной шины данных необходим однобайтовый latch (значения A15... A8 в регистре P0H остаются без изменения во время мультиплексирования данных и адреса). Для 16-разрядной шины данных необходим latch длиной в одно слово (при доступе к словам данных не используется A0, так как адрес слова всегда четный).

Старшие линии адреса (An ... A16) в случае работы в режиме сегментированного адреса постоянно выводятся через порт 4 и не требуют latch.

ЕВС инициализируется для внешнего доступа посредством создания сигнала включения latch адреса (ALE), после этого адрес выставляется на шину. По отрицательному фронту сигнала ALE происходит сохранение адреса во внешнем устройстве. После некоторого промежутка времени, в течении которого адрес должен быть считан внешним устройством, адрес удаляется с шины. В этот момент ЕВС активирует сигналы управления (\overline{RD} , \overline{WR} , \overline{WRL} или \overline{WRH}). Данные поступают на шину либо с помощью ЕВС (для цикла записи), либо от внешней памяти и периферии (для цикла чтения). После некоторого промежутка времени, которое определяется временем доступа к периферии или памяти, данные на шине становятся корректными.

Цикл чтения: входные данные захватываются и после этого сигнал управления деактивируется. После этого устройство имевшее доступ удаляет данные с шины, и шина переходит в состояние высокого сопротивления.

Цикл записи: Сигнал управления деактивируется. Данные на шине остаются правильными, до тех пор пока не начнется следующий цикл.

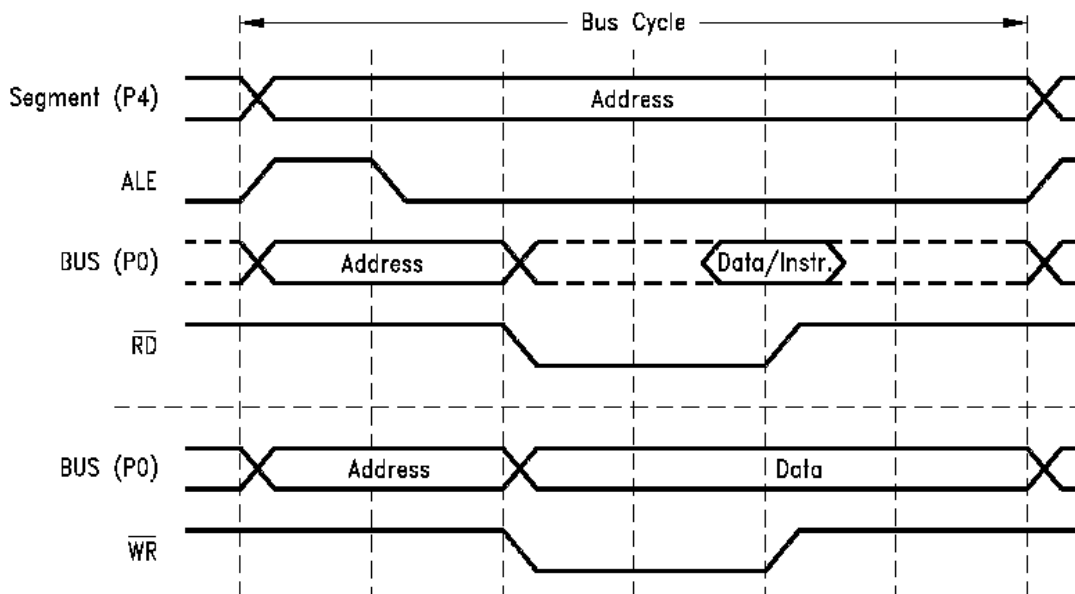


Рисунок 8-2

Цикл мультиплексированной шины

Режимы с демultipлексной шиной

В режиме с демultipлексной шиной 16-разрядный адрес постоянно выводится через порт 1, в то время как для данных используется порт 0 (16-разрядные данные) или P0L (8-разрядные данные).

Линии старших разрядов адреса постоянно выводятся через порт 4 (в тех случаях, когда во время RESET производится установка значений на входах SALSEL). При этом не используются адресные latches.

ЕВС начинает цикл внешнего доступа путем размещения адреса на шине адреса. После запрограммированного интервала времени, ЕВС активизирует необходимый сигнал управления (\overline{RD} , \overline{WR} , \overline{WRL} или \overline{WRH}). Данные передаются в шину данных либо при помощи ЕВС (для циклов записи), либо при помощи внешней памяти или внешней периферии (для циклов чтения). После некоторого периода времени, зависящего от времени доступа к периферии или внешней памяти, данные становятся корректными.

Цикл чтения: Производится захват входных данных. После этого сигнал управления переходит на неактивный уровень напряжения, при этом имевшее доступ устройство удаляет данные с шины и затем шина переходит в состояние высокого сопротивления.

Цикл записи: Сигнал управления переходит на неактивный уровень напряжения. В случае необходимости начала следующего цикла внешней шины ЕВС размещает следующий адрес на шине адреса. Данные остаются на шине данных без изменений до начала следующего цикла внешней шины.

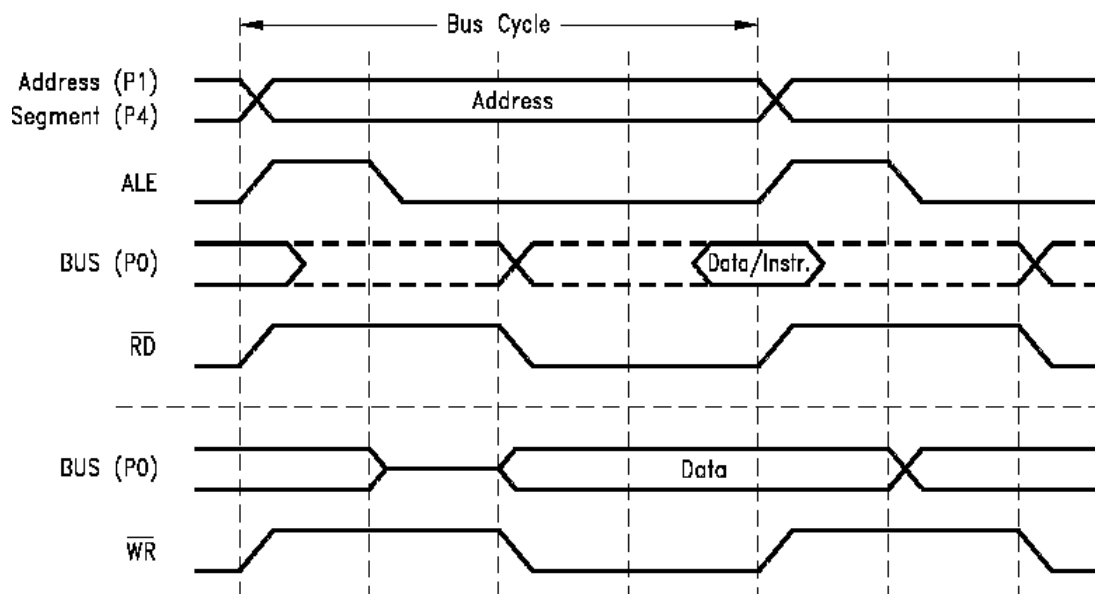


Рисунок 8-3

Цикл демультиплексной шины

Переключение между режимами шины

EBC позволяет динамическое переключение между различными режимами шины, т.е. последующий цикл внешней шины может быть выполнен в другом режиме. Одна и та же адресная область может использовать режим мультиплексной или демультиплексной шины, либо использовать \overline{READY} управление, либо предустановленные waitstates.

Изменение характеристик внешней шины может быть произведено двумя путями:

Перепрограммирование BUSCON и/или ADDRSEL регистров позволяет либо изменить режим шины для данного адресного окна, либо изменить размер адресного окна, используемого в данном режиме шины. Перепрограммирование позволяет использовать большое число различных адресных окон (больше чем доступное количество BUSCON-регистров).

Переключение между предустановленными адресными окнами: при этом автоматически выбирается режим шины, связанный с данным окном. Предустановленные адресные окна позволяют использовать различные режимы работы шины без загрузки ЦПУ, но при этом количество режимов ограничено числом регистров BUSCON. Однако, так как BUSCON0 управляет всей областью памяти, не занятой другими регистрами BUSCON, это позволяет допускать разрывы между окнами, при этом используется режим работы шины BUSCON0.

Порт 1 используется для вывода внутрисегментного адреса в том случае, когда хотя бы в одном из регистров BUSCON выбран режим демультиплексной шины. При этом не имеет значения какой тип шины

используется в текущем цикле шины (мультиплексный или демультиплексный). Это позволяет использовать внешний декодер адреса, подключенный только к порту 1, который используется для всех типов циклов шины.

Примечание: Не следует изменять конфигурацию шины для адресной области, являющейся источником текущего программного кода. Это обусловлено тем, что при конвейерной обработки команд очень трудно определить вызов первой команды, которая использует новую конфигурацию. Допускается изменение конфигурации только тех адресных областей, которые не используются для выборки кода в текущий момент. Это относится как к регистрам BUSCON и так и к регистрам ADDRSEL.

Изменение содержимого регистров BUSCON и ADDRSEL осуществляется через соответствующие SFR-регистры. В начале цикла внешнего доступа (вызов команд или данных) определяется расположение физического адреса данных или кода. В том случае если этот адрес попадает в одно из четырех окон ADDRSEL4...1 используется конфигурация шины в соответствующем регистре BUSON4...1, в ином случае используется предустановленная конфигурация в регистре BUSCON0. После инициализации активных регистров, они автоматически могут быть выбраны и считаны для определения физического адреса. Нет необходимости в дополнительных изменениях содержимого регистров BUSCON/ADDRSEL во время работы, за исключением тех случаев когда используется более четырех адресных окон.

Переключение из режима демультиплексной шины в режим мультиплексной шины представляет собой особый случай. В том случае если в любом из регистров BUSCON указан режим демультиплексной шины, цикл шины как обычно начинается с активизации сигнала ALE и выставления адреса на порт 4 и порт 1. Однако если текущий цикл использует режим мультиплексной шины, для передачи адреса необходим порт 0. В этом случае, выставление адреса на порт 0 задерживается на один такт ЦПУ, при этом задерживается завершение цикла мультиплексной шины и продлевается сигнал ALE (на рисунке ниже).

Дополнительное интервал времени необходим для того, чтобы ранее выбранное устройство (в режиме демультиплексной шины) отключилось от шины данных.

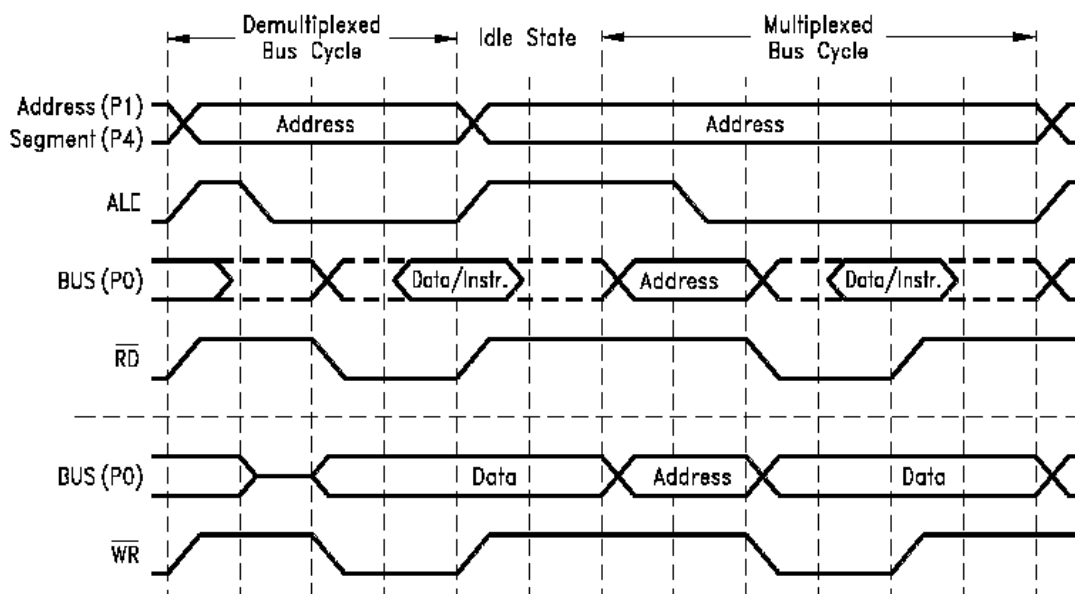


Рисунок 8-4

Переключение из режима демультиплексной в режим мультиплексной шины

Ширина внешней шины данных

ЕВС может работать как с 8-разрядными так и с 16-разрядными внешними устройствами и внешней памятью. Для 16-разрядной шины данных используется порт 0. Для 8-разрядной шины данных используется только P0L, при этом уменьшается необходимость в адресных latches, bus transceivers, bus routing. ЕВС может осуществлять доступ к словам по 8-разрядной шине данных, так же он может осуществлять побайтовый доступ по 16-разрядной шине данных.

Доступ к словам с помощью 8-разрядной шины автоматически разделяется на два последовательных цикла побайтового доступа, при этом сначала доступ осуществляется к младшему байту а затем к старшему. Объединение байтов в слово и разделение слов на байты осуществляется в ЕВС, и результат передается в ЦПУ в явном виде.

Побайтовый доступ к 16-разрядной шине данных необходим в том случае, когда необходимо получить только старший или младший байт слова. В этом случае старший байт вызывается по сигналу \overline{BHE} , а младший байт вызывается по сигналу A0. Таким образом в зависимости от необходимости возможно осуществлять доступ к каждому байту независимо друг от друга, или осуществлять одновременный доступ к слову.

При совершении побайтной записи во внешние 16-разрядные устройства, имеющие только один вход \overline{CS} но два \overline{WR} -входа, ЕВС может корректно выдавать два необходимых сигнала \overline{WR} . В этом случае \overline{WR} обслуживается как \overline{WRL} (для сигнала управления на запись младшего байта),

а вывод \overline{BHE} обслуживается как \overline{WRH} (для сигнала управления на запись старшего байта). Режим работы для выводов \overline{WR} и \overline{BHE} устанавливается в бите WRCFG регистра SYSCON.

При чтении байта из внешнего 16-разрядного устройства, может быть прочтено все слово, и автоматически выбран только необходимый байт. Однако необходимо проявлять повышенное внимание при чтении устройств, меняющих состояние данных в процессе чтения: FIFO, регистры состояния прерывания и др. В этом случае индивидуальные байты необходимо выбирать с помощью \overline{BHE} и A0.

Режим шины	Скорость передачи (скорость для доступа к байту/ слову/ двойному слову)	Системные требования	Свободные линии I/O
8-разрядная мультиплекс	Очень медленная 1.5 / 3 / 6	Низкие (8-разрядный latch, байтовая шина)	P1H, P1L
8-разрядная демульт.	Медленная 1 / 2 / 4	Очень низкие (no latch, байтовая шина)	P0H
16-разрядная мультиплекс	Высокая 1.5 / 1.5 / 3	Высокие (16-разрядный latch, шина слов)	P1H, P1L
16-разрядная демульт.	Очень высокая 1 / 1 / 2	Низкие (no latches, шина слов)	---

Примечание: В тех случаях когда ни в одном из регистров BUSCON не выбран режим демультимплексной шины, порт1 остается доступным для функции ввода-вывода основного назначения.

Отключение/включение управления \overline{BHE} (BYTDIS)

Бит BYTDIS обеспечивает управление состоянием вывода \overline{BHE} . Функция вывода \overline{BHE} включена, в том случае, если в бите BYTDIS содержится «0». В ином случае эта функция отключена и вывод порта может использоваться для стандартной функции ввода-вывода. \overline{BHE} используется контроллером внешней шины для выбора старшего или младшего байта в устройствах с двухбайтной организацией памяти, подключаемых к C167 через 16-разрядную шину внешних данных. После RESET функция \overline{BHE} автоматически включается (BYTDIS = 0), в том случае если во время RESET устанавливается режим 16-разрядной шины данных. Данная функция может быть отключена, если отсутствует необходимость в побайтовом доступе при использовании 16-разрядной шины.

Использование сегментированного адреса

Во время внешнего доступа ЕВС может использовать несколько линий адреса порта 4, при этом достигается расширение 16-разрядного адреса, выводимого через порт 0 или порт 1. Таким образом достигается возможность увеличения доступного адресного пространства. Количество линий адреса сегмента можно выбрать во время RESET путем подачи сигналов на вход битового поля SALSEL регистра RP0H (ниже в таблице).

SALSEL	Линии адреса сегмента	Доступное адресное пространство
1 1	Две: A17... A16	256 Кбайт (предустановленно без pull-down)
1 0	Восемь: A23... A16	16 Мбайт (максимум)
0 1	Нет	64 Кбайт (минимум)
0 0	Четыре: A19... A16	1 Мбайт

Примечание: Общее доступное адресное пространство может быть увеличено посредством использования доступа к нескольким банкам, которые различаются индивидуальными сигналами chip select.

Использование сигналов \overline{CS}

Во время осуществления внешнего доступа, ЕВС может использовать несколько сигналов \overline{CS} , представляющих собой альтернативную функцию порта 6. Эти сигналы позволяют выбирать внешнюю периферию или банк памяти без использования дополнительного внешнего декодера. Количество \overline{CS} линий выбирается во время RESET путем подачи сигналов в битовое поле CSSEL регистра RP0H (ниже в таблице).

CSSEL	Линии chip select	Примечание
1 1	Пять: $\overline{CS4}..\overline{CS0}$	Предустановлено без pull-down
1 0	Нет	Выводы порта 6 свободны для IO
0 1	Две: $\overline{CS1}..\overline{CS0}$	
0 0	Три: $\overline{CS2}..\overline{CS0}$	

Выводы \overline{CSx} привязаны к регистрами BUSCONx и изменяют свое значение при попытке доступа в адресную область, определенную соответствующим BUSCON-регистром. Для любой попытки доступа вне определенной области адресного пространства, сигнал \overline{CSx} переходит в неактивное состояние (высокий уровень напряжения). В начале каждого цикла внешней шины производится определение и выставление необходимого сигнала \overline{CS} . При этом все другие линии \overline{CS} переходят в неактивное состояние (переходят на высокий уровень напряжения на выходе).

Примечание: Сигналы \overline{CS}_x остаются без изменения, в случае доступа к области внутренней памяти (т.е. в том случае, когда не начинается новый цикл внешней шины). Сигналы остаются без изменения даже в том случае, когда эта область попадает в область регистра ADDRSEL_x. Доступ к внутренней X-периферии переводит на неактивный уровень напряжения все сигналы \overline{CS} .

При попытке доступа к адресному окну с неопределенным \overline{CS} -сигналом, все выбранные \overline{CS} -сигналы переходят на неактивный уровень напряжения.

Сигналы chip select могут использоваться в четырех различных режимах, выбираемых с помощью установки битов CSWEN_x и CSREN_x в соответствующем регистре BUSCON_x.

CSWEN _x	CSREN _x	Режим chip select
0	0	Адресный Chip select (предустанавливается после RESET, режим для \overline{CS}_0)
0	1	Chip select для чтения
1	0	Chip select для записи
1	1	Chip select для чтения/записи

Адресные сигналы chip select остаются активными до тех пор, пока не будет произведен доступ в другое адресное окно. Адресный chip select переходит на активный уровень напряжения при поступлении отрицательного фронта ALE и возвращается в неактивное состояние при поступлении отрицательного фронта сигнала ALE для доступа к другой области адресов.

Сигналы Chip select для записи или чтения будут активными столько, сколько остается активным соответствующий управляющий сигнал (\overline{RD} или \overline{WR}). В это время входит программируемая задержка чтения или записи.

Примечание: После RESET вызов первой команды осуществляется в режиме адресного chip select для сигнала \overline{CS}_0 (за исключением режима микроконтроллера без внешней периферии).

Внутренние pullup устройства во время процедуры RESET подключают все выходы сигналов \overline{CS} к высокому уровню напряжения. После окончания RESET, pullup устройства отключаются и драйвера уровня выходного напряжения самостоятельно управляют выбранными \overline{CS} линиями. Не выбранные \overline{CS} линии переводятся в состояние высокого сопротивления на выходе и свободны для ввода-вывода основного назначения.

Когда \overline{HLDA} находится на активном уровне, и соответствующий вывод микросхемы переключается в режим push/pull, pullup устройства активны во время удержания шины выбранным \overline{CS} сигналом. Использование выводов с открытым коллектором приводит к плавающему потенциалу во

время удержания шины. В этом случае необходимо использование внешних pullup устройств или использование новых masters шины, необходимых для передачи соответствующих уровней напряжения на линии \overline{CS} сигналов.

Сегментный адрес и chip select

Интерфейс внешней шины C167 поддерживает большое количество конфигураций внешней памяти. Посредством увеличения числа линий сегментной адресации, C167 может адресовать линейное адресное пространство 256Кбайт, 1Мбайт или 16Мбайт. Это позволяет подключать большие последовательные области памяти, и также позволяет получать доступ к большому числу внешних устройств с использованием внешнего декодера. Посредством увеличения числа линий \overline{CS} -сигналов, C167 может получить доступ к банкам памяти или периферии без внешней дополнительной логики. Эти две возможности можно объединить и оптимизировать для увеличения быстродействия системы. Разрешение одновременно четырех линий адреса сегмента и пяти линий chip select позволяет получить доступ к пяти банкам памяти по 1Мбайту в каждом. Поэтому максимально возможное адресное пространство без дополнительной внешней логики составляет 5 Мбайт.

Примечание: Значение бита SGTDIS регистра SYSCON определяет необходимость сохранения значения регистра CSP в стеке при входе в подпрограмму прерывания (в случае наличия сегментации).

8.2 Программируемые характеристики шины

Наиболее значимые временные характеристики интерфейса внешней шины могут быть запрограммированы пользователем, что позволяет адаптировать их к различным конфигурациям внешней шины с различными типами памяти и/или периферии.

Возможно программное изменение следующих параметров:

- **Управление ALE** определяет длительность сигнала ALE и время удержания адреса после прихода отрицательного фронта
- **Время цикла памяти** (1... 15 waitstates) определяет дозволенное время доступа
- **Время нахождения памяти в переходном состоянии** (1 waitstate) определяет время нахождения напряжения на выходе драйвера данных на неопределенном потенциале
- **Время задержки чтения/записи** определяет задержку начала команды чтения/записи после прихода отрицательного фронта сигнала ALE
- **Управление READY** определяет тип завершения цикла работы шины (внешне или внутренне)

Примечание: Доступ к внутренним ресурсам осуществляется с максимальной скоростью и поэтому не программируется. После RESET для внешнего доступа используется самый медленный из возможных циклов работы шины. Время цикла работы шины может быть оптимизировано посредством настроечной программы.

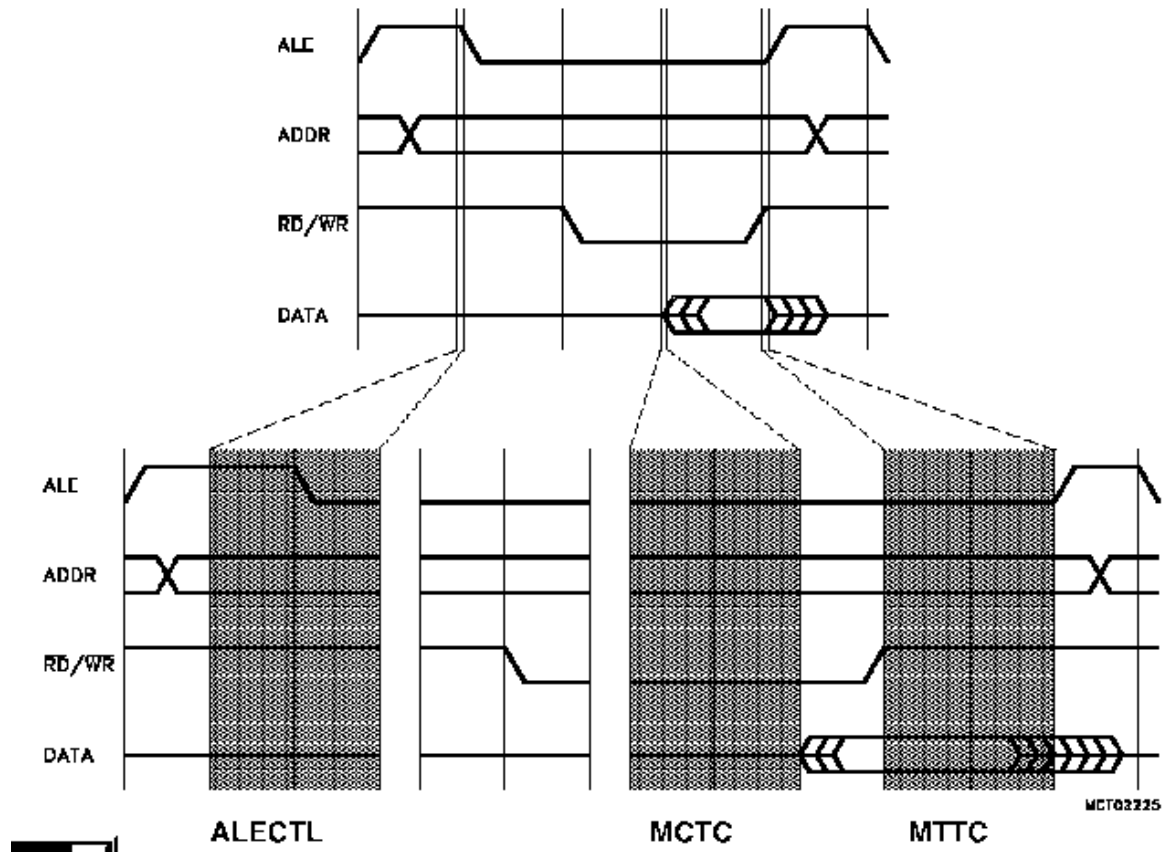


Рисунок 8-5

Программируемый цикл работы внешней шины

Управление ALE

Длительность ALE сигнала и время удержания адреса после отрицательного фронта ALE изменяется при помощи изменения содержимого битов ALECTLx регистра BUSCON. В случае установки «1» в бите ALECTL, доступ к адресному окну во время цикла работы внешней шины будет осуществляться по сигналу ALE с продолжительностью в половину такта ЦПУ (25нс при частоте ЦПУ 20МГц). Время удержания адреса после поступления отрицательного фронта сигнала ALE для мультиплексной шины будет увеличено на половину такта ЦПУ, поэтому длительность передача данных во время цикла работы шины зависит от сигнала CLKOUT (иными словами передача данных имеет задержку в один такт ЦПУ). Это позволяет выделять больше времени на захват адреса.

Примечание: После RESET бит ALECTL0 содержит «1», что необходимо для использования самого медленного режима работы шины, другие ALECTLx также содержат «0».

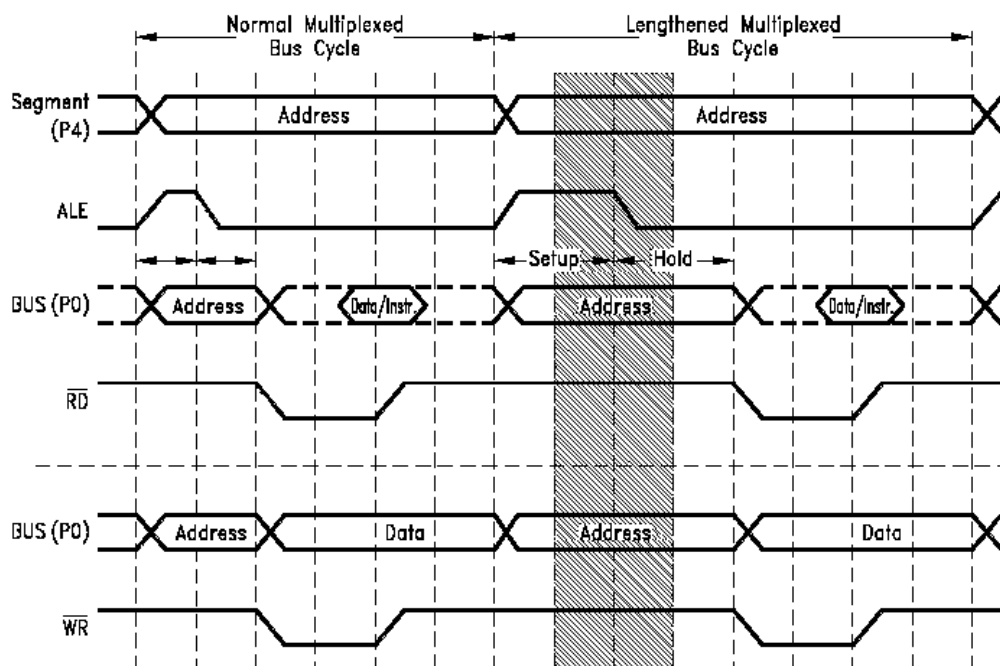


Рисунок 8-6

Управление длительностью ALE

Программируемое время работы цикла памяти

C167 позволяет пользователю настраивать цикл работы внешней шины микроконтроллера для подстройки под внешнюю память или периферию. Время доступа – это общее время, необходимое для перемещения данных в место назначения. Оно представляет из себя интервал времени, в течении которого сигналы микроконтроллера не изменяется.

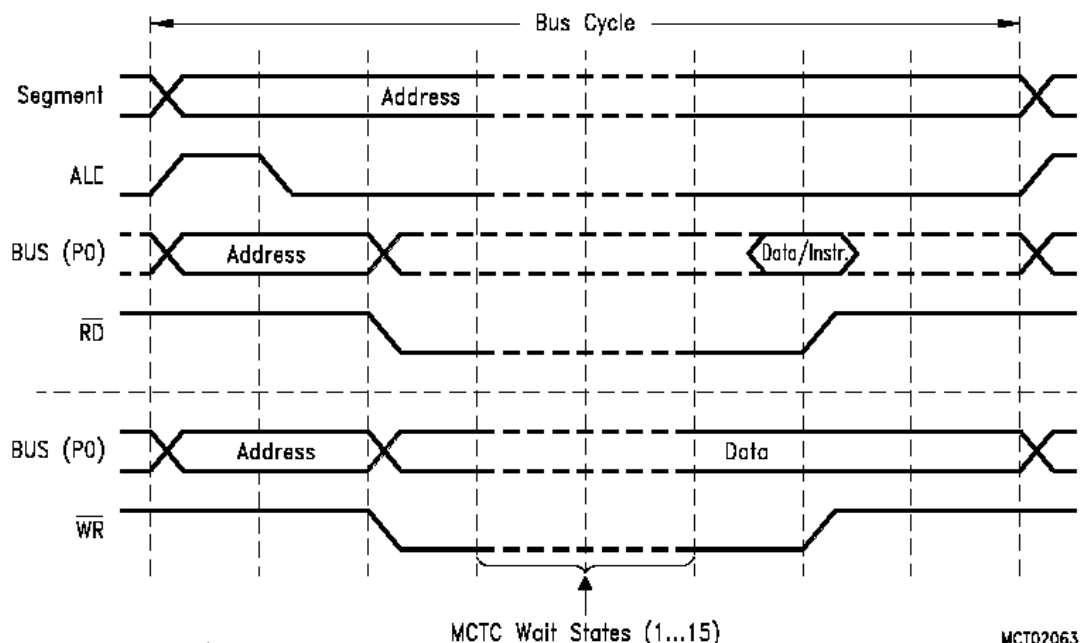


Рисунок 8-7

Время цикла памяти

Цикл работы внешней шины C167 может быть изменен таким образом, чтобы та память или периферия, которая не может работать со скоростью микроконтроллера, могла быть подключена к микроконтроллеру посредством введения wait states во время доступа (на рисунке выше). В том случае если текущий доступ к шине необходим для выполнения команды, ЦПУ находится в холостом режиме во время wait states.

Wait states цикла памяти могут программироваться с шагом в 1 такт ЦПУ (т.е. 50нс при частоте ЦПУ 20МГц) в диапазоне от 0 до 15 тактов (15 предустановленно после RESET) при помощи изменения значения битового поля MCTC регистра BUSCON.

Программируемое время нахождения памяти в переходном состоянии

C167 позволяет пользователям настраивать промежуток времени между двумя последующими внешними доступами, чтобы исключить возможность попадания в промежуток времени, в течении которого внешнее устройство находится в третьем состоянии. Время нахождения в третьем состоянии определяется с того момента, когда внешнее устройство освобождает внешнюю шину после перехода сигнала \overline{RD} на неактивный уровень напряжения.

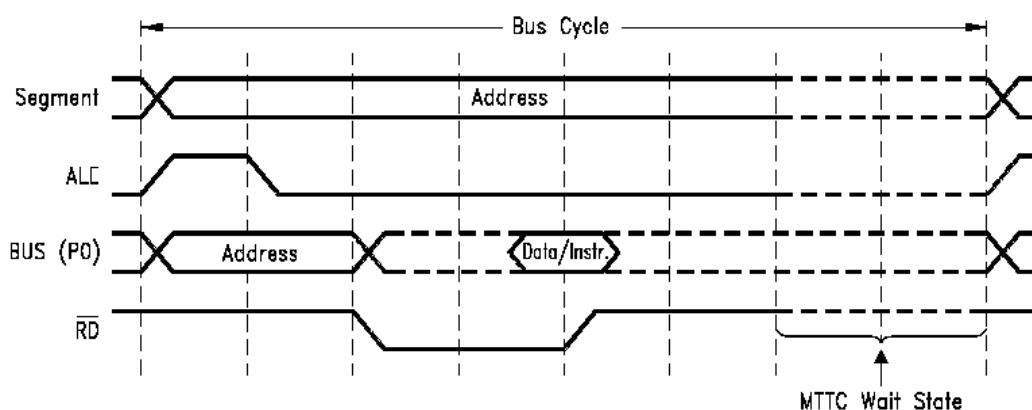


Рисунок 8-8

Время нахождения памяти в третьем состоянии

Вывод следующего адреса на внешнюю шину может быть задержан для той периферии или памяти, которая нуждается в большем временном интервале для выключения драйверов шины. Это достигается путем добавления wait state после предыдущего цикла работы шины. В течение нахождения шины в третьем состоянии, ЦПУ не обязан находиться в состоянии холостого хода, поэтому производительность ЦПУ понизится только в том случае, если требуются получение команды из внешней памяти или вызов данных из внешней памяти в следующем командном такте.

Waitstate памяти в третьем состоянии занимает один такт ЦПУ (50нс при частоте ЦПУ 20МГц) и управляется с помощью изменения значения бита MTTCx регистра BUSCON. Waitstate будет добавлена в том случае, если бит MTTCx установлен в «0» (предустановленно после RESET).

Примечание: При работе цикла внешней шины в режиме мультиплексной шины неявно добавляется один такт нахождения в третьем состоянии waitstate в дополнении к запрограммированной MTTC waitstate.

Задержка сигнала чтения/записи

C167 позволяет пользователям настраивать временные характеристики команд чтения и записи к требованиям внешней периферии. Задержка чтения и записи управляет временным интервалом между отрицательным фронтом сигнала ALE и отрицательным фронтом команды. Без задержки чтения и записи, отрицательный фронт ALE и отрицательный фронт команды будут совпадать (за исключением задержки распространения (propagation delay)). При включенной задержке после прихода отрицательного фронта сигнала ALE сигналы WR/RD переходят на активный уровень напряжения после половины такта ЦПУ (25нс при частоте 20МГц).

Задержка записи и чтения не увеличивает общее время цикла памяти и не уменьшает производительность микроконтроллера в целом. Однако в режиме с мультиплексной шиной драйвера данных внешних устройств могут конфликтовать с адресами, выставленными на шину микроконтроллером C167, в случае использования раннего сигнал \overline{RD} . Поэтому необходимо в режиме мультиплексной шины всегда программировать задержку чтения и записи.

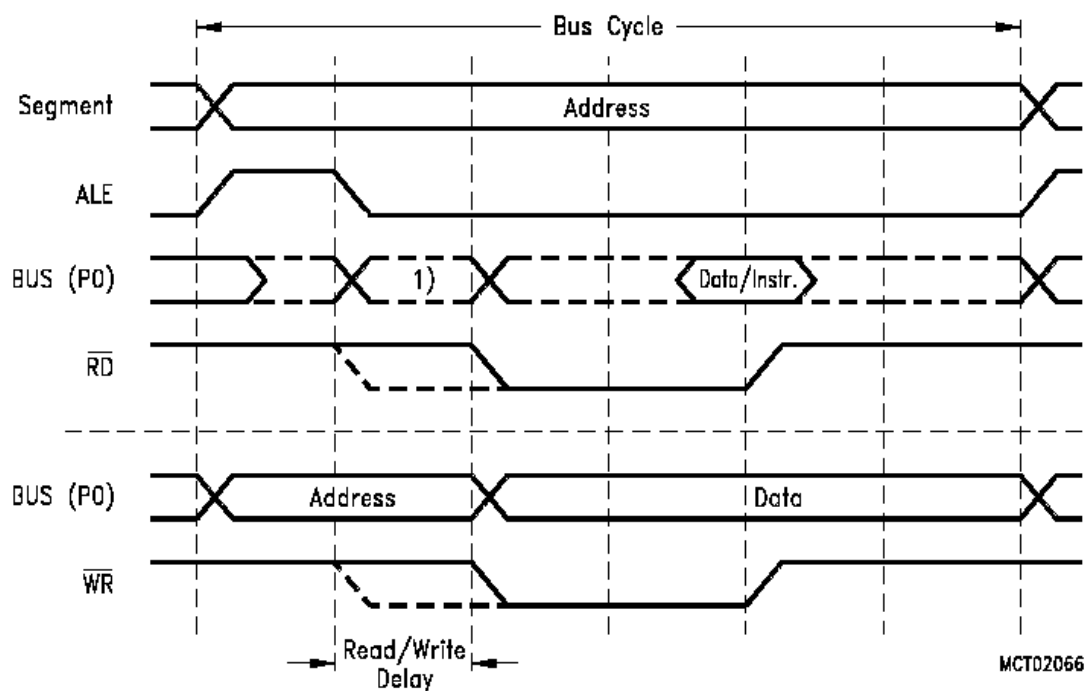


Рисунок 8-9

Задержка записи и чтения

- 1) Драйвера данных от предыдущего цикла работы шины должны быть отключены, прежде чем сигнал \overline{RD} перейдет на активный уровень напряжения.

Задержка записи и чтения управляется в битах RWCDx регистров BUSCON. Задержка команды будет иметь место в том случае, если в бите RWCDx установлен «0» (предусмотрено после RESET).

8.3 Цикл работы шины, управляемый сигналом \overline{READY}

Для ситуаций в которых недостаточно применения программных waitstates, или где время ответа (доступа) к периферии не является постоянным, C167 предлагает цикл работы внешней шины, заканчивающийся по внешнему сигналу \overline{READY} (синхронному или асинхронному). В этом режиме C167 сначала добавляет некоторое количество программируемых waitstates (0... 7), и затем начинает отслеживать \overline{READY} -вход, для определения реального окончания текущего цикла работы шины. Внешние устройства передают низкий уровень напряжения сигнала \overline{READY} в том случае, когда данные захвачены (цикл записи) или доступны (цикл чтения).

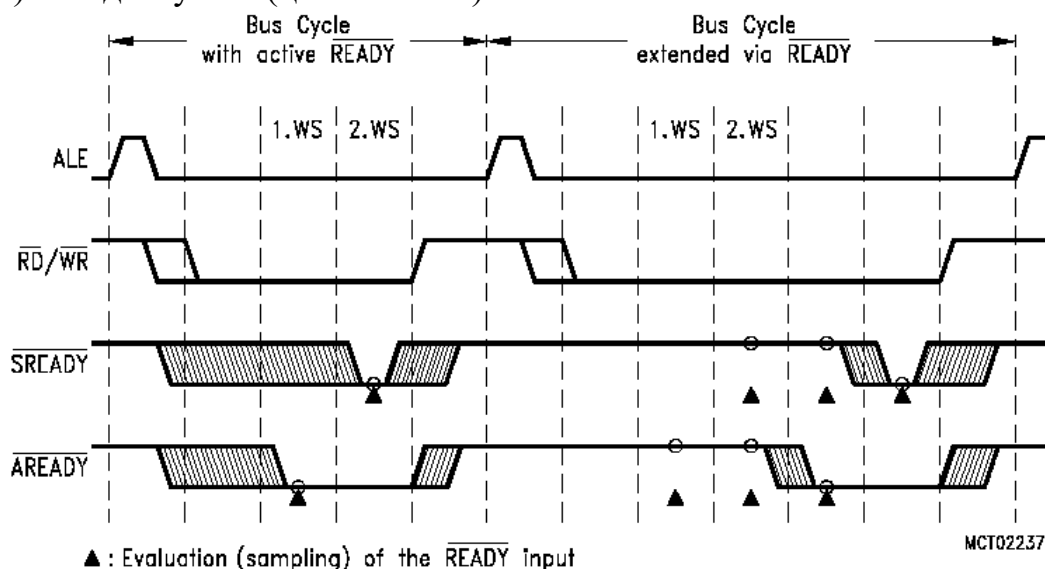


Рисунок 8-10

Цикл работы шины, управляемый сигналом \overline{READY}

Использование сигнала \overline{READY} включается в бите RDYENx регистра BUSCON. В случае использования этой функции (RDYENx = «1»), только младшие 3 бита битового поля MCTC определяют количество вставленных waitstates (0... 7). Бит MSB битового поля MCTC используется для выбора режима работы по сигналу \overline{READY} :

MCTC.3 = 0: Синхронный \overline{READY}

MCTC.3 = 1: Асинхронный \overline{READY} т.е. сигнал \overline{READY} синхронизируется внутренне.

Синхронный \overline{READY} обеспечивает максимально возможный быстрый цикл работы шины, но для него необходима настройка. В этом режиме необходимо, чтобы периферийная логика использовала сигнал CLKOUT для управления временными характеристиками \overline{READY} .

Асинхронный \overline{READY} является менее ограниченным жесткими рамками, но требует дополнительных waitstates для внутренней синхронизации. Так как асинхронный \overline{READY} проще захватить (на рисунке выше), то для обеспечения соответствующего цикла работы шины необходимы программируемые waitstates (см. также ниже примечания о «нормально-готовой» периферии).

Сигнал \overline{READY} (в особенности асинхронный \overline{READY}), переходящий в активное состояние под действием внешних устройств, может в ответ на положительный фронт сигнала \overline{RD} или \overline{WR} вернуться в неактивное состояние.

Примечание: В случае работы адресного окна с функцией \overline{READY} , каждый цикл работы шины с этим окном должен завершаться по сигналу \overline{READY} . В ином случае контроллер оказывается в подвешенном состоянии до тех пор, пока не будет совершен RESET. Функцию слежения за превышением времени исполнения команды, по которой осуществляется доступ к шине, обеспечивает сторожевой таймер.

Объединение функции \overline{READY} и предустановленных waitstates

Эта возможность оказывает благоприятное влияние на работу шины в двух случаях:

Периферия и компоненты памяти с фиксированным временем доступа, имеющие возможность использовать \overline{READY} , может группироваться в одном и том же адресном окне. В этом случае (внешняя) логика управления waitstates переведет сигнал \overline{READY} в активный уровень напряжения, либо с помощью сигналов памяти chip select, либо с помощью периферии. После предустановленного количества waitstates, C167 проверяет состояние линии \overline{READY} , для определения окончания цикла работы шины. При доступе к памяти сигнал уже находится на низком уровне (см. пример А, представленный на рисунке выше). При доступе к периферии сигнал может быть задержан (пример В, представленный на рисунке выше). Так как память имеет тенденцию быть более быстрой чем периферия, не должно ожидать воздействия на быстродействие системы.

При использовании функции \overline{READY} с так называемой «нормально-готовой» периферией, возможно возникновение ошибочных циклов работы шины, если сигнал \overline{READY} срабатывает слишком рано. Эти устройства посылают сигнал \overline{READY} , в то время как они еще находятся в режиме покоя. При получении к ним доступа, они оставляют сигнал \overline{READY} на неактивном уровне напряжения до тех пор, пока не завершится цикл работы шины. После этого сигнал \overline{READY} снова устанавливается в активное состояние. Однако, если периферия устанавливает сигнал \overline{READY} в неактивное состояние после первой точки выборки C167, контроллер выбирает активный уровень сигнала

\overline{READY} и прекращает текущий цикл работы шины. Конечно, это происходит слишком рано. Посредством вставления предустановленных waitstates, первая точка выборки \overline{READY} -сигнала может быть смещена во времени в тот временной интервал, где периферия сохраняет контроль за линией \overline{READY} (т.е. после второго waitstates, как показано на рисунке выше).

8.4 Управление контроллером внешней шины

ЕВС управляется при помощи набор регистров. Основные параметры настраиваются в регистре SYSCON, а именно интерфейсные выходы (\overline{WR} , \overline{BHE}), сегментация памяти и расположение внутренней ROM. Особенности работы цикла памяти, такие как режим chip select, использование \overline{READY} , продолжительность ALE, режим работы внешней шины, задержка чтения и записи и waitstates управляются через регистры BUSCONx. Четыре регистра (BUSCON4... BUSCON1) связаны с регистрами выбора адреса (ADDRSEL4... ADDRSEL1), которые позволяют предустанавливать четыре области адресов и индивидуальные характеристики внутри этих областей. Все попытки доступа по адресам, не покрытым этими областями, управляются с помощью регистра BUSCON0. Это позволяет в одной и той же системе использовать компоненты памяти или периферии с различными интерфейсами, добиваясь при этом оптимизированного доступа к каждому внешнему устройству.

Регистр SYSCON подробно описан в главе посвященной ядру микроконтроллера. Структура пяти регистров BUSCON идентична. Управляющие выбранными адресными окнами регистры BUSCON4... BUSCON1, полностью управляются программно. Регистр BUSCON0, использующийся для выборки первого адреса после RESET, частично управляется аппаратно, т.е. инициализируется через порт 0 во время RESET. Аппаратное управление необходимо для инициализации внешней шины для тех систем, где отсутствует внутренняя программная память.

SFR

BUSCON0 (FF0C_H/86_H)

Значение после RESET: 0XX0_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSW EN0	CSR EN0	-	RDY EN0	-	BUS ACT0	ALE CTL0	-	BTYP		MTT C0	RWD C0	MCTC			
rw	rw	-	rw	-	rw	rw	-	rw		-	-	rw			

Регистры BUSCON1... BUSCON4 имеют точно такую же структуру.

BUSCON1 (FF14H/8AH)

BUSCON2 (FF16H/8BH)

BUSCON3 (FF18H/8CH)

BUSCON4 (FF1AH/8DH)

Примечание:, Если во время RESET подать высокий уровень напряжения на вход \overline{EA} , то в регистре BUSCON0 устанавливается 0000_H. Если во время RESET на вход \overline{EA} подать низкий уровень напряжения, то биты BUSACT0 и ALECTL0 устанавливаются в «1» и значение битового поля ВТУР устанавливается в соответствии со значениями на входе порта 0.

Бит	Функция
MCTC	Управление временем цикла памяти (количеством waitstates) 0 0 0 0 : 15 waitstates 1 1 1 1 : Нет waitstates
RWDCx	Управление задержкой записи и чтения для BUSCONx 0: задержка имеет место, после отрицательного фронта сигнала ALE 1: нет задержки
MTTCx	Управление временем нахождения в переходном состоянии 0: 1 waitstate 1: нет waitstates
BTYP	Конфигурация внешней шины 0 0 : 8-разрядная демультиплексная шина 0 1 : 8-разрядная мультиплексная шина 1 0 : 16-разрядная демультиплексная шина 1 1 : 16-разрядная мультиплексная шина
ALECTLx	Управление длительностью ALE 0: нормальный сигнал ALE 1: удлиненный сигнал ALE
BUSACTx	Управление работой шины 0: внешняя шина отключена 1: внешняя шина включена
RDYENx	Разрешение входа READY 0: цикл работы внешней шины управляется только битами MCTC 1: Цикл работы внешней шины управляется входным сигналом READY
CSRENx	Разрешение чтения с помощью сигнала chip select 0: Сигнал \overline{CS} независим от сигнала чтения \overline{RD} 1: Сигнал \overline{CS} генерируется для выполнения команды чтения
CSWEN	Разрешение записи с помощью сигнала chip select 0: Сигнал \overline{CS} независим от сигнала записи \overline{WR} , \overline{WRH} , \overline{WRL} 1: Сигнал \overline{CS} генерируется для выполнения команды записи

SFR
ADDRSEL1 (FE18_H/0C_H) **Значение после RESET: 0000_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RGSAD												RGSZ			
rw												rw			

Регистры ADDRSEL2... ADDRSEL4 имеют точно такую же структуру.

ADDRSEL2 (FE1A_H/0D_H)

ADDRSEL3 (FE1C_H/0E_H)

ADDRSEL4 (FE1E_H/0F_H)

Бит	Функция
RGSZ	Выбор размера Определяет размер области адресов, управляемой парой регистров BUSCON _x /ADDRSEL _x .
RGSAD	Старшие биты адреса Определяет старшие биты в адресе (A23...), соответствующей области адресов.

Примечание: Отсутствие регистра ADDRSEL0 обусловлено тем, что регистр BUSCON0 управляет доступом ко всем адресам кроме адресных окон, установленных в регистрах BUSCON4... BUSCON1.

Настройка адресных окон

Для установки 4 различных адресных областей внутри адресного пространства C167 предназначены четыре пары регистров BUSCON4/ADDRSEL4... BUSCON1/ADDRSEL1. К каждой из этих адресных областей можно совершать внешний доступ в различных и независимых друг от друга режимов работы шины. В каждом регистре ADDRSEL_x определяется адресное окно с параметрами доступа, установленными в регистре управления внешним доступом BUSCON_x. В представленной ниже таблице обозначены «R» только те биты адреса, которые используются для конкретного размера окна.

Поле RGSZ	Размер окна	Значение (R) биты в адресе (A23 ... A12)
0 0 0 0	4Кбайта	R R R R R R R R R R R R R R
0 0 0 1	8Кбайта	R R R R R R R R R R R R R x
0 0 1 0	16Кбайта	R R R R R R R R R R R R x x
0 0 1 1	32Кбайта	R R R R R R R R R R x x x x
0 1 0 0	64Кбайта	R R R R R R R R x x x x x x
0 1 0 1	128Кбайта	R R R R R R R x x x x x x x
0 1 1 0	256Кбайта	R R R R R R x x x x x x x x
0 1 1 1	512Кбайта	R R R R R x x x x x x x x x
1 0 0 0	1Мбайт	R R R R x x x x x x x x x x
1 0 0 1	2Кбайта	R R R x x x x x x x x x x x
1 0 1 0	4Мбайта	R R x x x x x x x x x x x x
1 0 1 1	8Мбайта	R x x x x x x x x x x x x x
1 1 x x	Зарезервировано	

Приоритетность адресных окон

При каждой попытке доступа, EBC сравнивает текущий адрес с адресными окнами, устанавливаемыми в соответствующих регистрах (программируемых ADDRSELx и аппаратных XADRSx). Сравнение производится по четырем уровням.

Уровень 1: Первыми просматриваются области в аппаратно-устанавливаемых регистрах XADRSx. В том случае, если адрес попадает в зону одного из этих регистров, предоставляется доступ к X-периферии, использующей соответствующие регистры XBCONx, и все другие регистры ADDRSELx игнорируются.

Уровень 2: Зоны регистров ADDRSEL2 и ADDRSEL4 оцениваются прежде чем будут оценены зоны соответственно ADDRSEL1 и ADDRSEL3. В случае попадания адреса в зону одного из этих регистров, предоставляется доступ к соответствующей внешней области, при этом используется соответствующий регистр BUSCONx, и регистры ADDRSEL1/3 игнорируются (ниже на рисунке).

Уровень 3: При попадании адреса в зону влияния регистров ADDRSEL1 или ADDRSEL3, производится внешний доступ с использованием соответствующих BUSCONx.

Уровень 4: В том случае если адрес не попадает в зону влияния ни одного из регистров XADRSx или ADDRSELx, то доступ к внешней шине производится с помощью регистра BUSCON0.

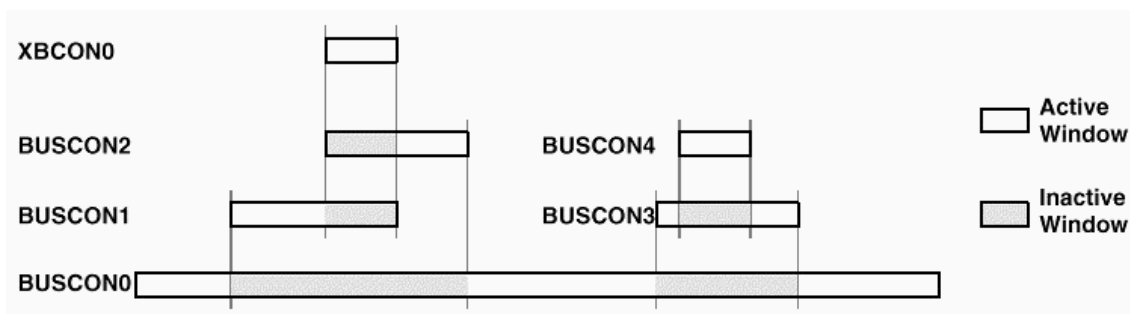


Рисунок 8-11

Приоритеты адресных окон

Примечание: Возможны только показанные выше частичные совпадения. Все другие совпадения приводят к ошибочным циклам работы шины. Т.е. ADDRSEL4 не может перекрывать ADDRSEL2/1. Аппаратные регистры XADRSx не могут перекрываться.

SFR

RP0H (F108_H/84_H)

Значение после RESET: --XX_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								CLKCFG			SALSEL		CSSEL		WRC
-	-	-	-	-	-	-	-	r	r	r	r		r		r

Примечание: RP0H не может быть программно изменен, однако его наличие позволяет проверять текущую конфигурацию.

Бит	Функция
WRC	Настройка записи 0: Выводы \overline{WR} и \overline{BHE} обслуживаются, как сигналы \overline{WRL} и \overline{WRH} 1: Выводы \overline{WR} и \overline{BHE} обслуживаются, как сигналы \overline{WR} и \overline{BHE}
CSSEL	Выбор линий chip select (количества активных выводов \overline{CS}) 0 0: три \overline{CS} линии $\overline{CS2} \dots \overline{CS0}$ 0 1: две \overline{CS} линии $\overline{CS1} \dots \overline{CS0}$ 1 0: нет \overline{CS} линий 1 1: пять \overline{CS} линии $\overline{CS4} \dots \overline{CS0}$
SALSEL	Выбор линий для адресации сегмента (количество активных выходов для адреса сегмента) 0 0: 4-битный адрес сегмента A19... A16 0 1: Нет линий адреса сегмента 1 0: 8-битный адрес сегмента A23... A16 1 1: 2-битный адрес сегмента A17... A16
CLKCFG	Настройка режима тактового генератора Эти выводы определяют режим тактового генератора, т.е. определяют механизм, по которому внутренний тактовый генератор ЦПУ работает от внешнего тактового входа (XTAL1).

Предостережения и советы

- Интерфейс внешней шины не будет задействован до тех пор, пока хотя бы в одном из регистров BUSCON не будет установлена «1» в бите BUSACT.
- Порт 1 не будет выводить внутрисегментный адрес до тех пор, пока как минимум в одном из BUSCON регистров не будет выбран режим демультимплексированной внешней шины.
- Не все области адреса, определенные через регистры ADDRSELx, могут пересекаться друг с другом. Работа с EBC в этом случае может привести к непредсказуемым результатам.
- Области адресов, определенные с помощью регистров ADDRSELx, могут пересекаться с внутренними адресными пространствами. В этом случае будет выполнен доступ к внутренней памяти.
- При любом доступе к внутреннему адресному пространству, EBC переходит в пассивный режим (см. Режим холостого хода EBC).

8.5 Режим холостого хода EBC

При включенном интерфейсе внешней шины, EBC переходит в режим холостого хода, при отсутствии попыток выполнения внешнего доступа. Пока осуществляется доступ к внутренним ресурсам, интерфейс внешней шины не используется.

Доступ к X-периферии также управляется EBC. Однако при использовании X-периферии не создается правильный цикл работы внешней шины.

Это обусловлено тем, что адрес и читаемые данные в цикле работы XBUS отражаются на интерфейсе внешней шины (ниже в таблице). Шина адреса отражается в порте 1, порте 4. При выполнении цикла XBUS изменяют свое значение \overline{BHE} , и ALE . Сигналы \overline{CS} порта 6 переходят на неактивный (высокий) уровень напряжения, так как EBC переключается на внутренний \overline{XCS} сигнал.

Внешние сигналы управления (\overline{RD} и \overline{WR} и др.) остаются на неактивном (высоком) уровне напряжения.

Состояние интерфейса внешней шины во время режима холостого хода EBC:

Выводы	Только внутренний доступ	XBUS доступ
Порт 0	плавающий потенциал	плавающий потенциал для операции чтения XBUS, Записываемые данные для операции записи
Порт 1	Последний использованный внешний адрес (при использовании интерфейса шины)	Последний использованный адрес XBUS (при использовании интерфейса шины)
Порт 4	Последний использованный адрес сегмента (на выбранных выводах)	Последний использованный XBUS адрес сегмента (на выбранных выводах)
Порт 6	Внешний сигнал \overline{CS} остается в активном положении, соответствующий последнему использованному адресу	Не активный уровень \overline{CS} сигналов
\overline{BHE}	Уровень соответствует, последнему внешнему доступу	Уровень соответствует последнему доступу к XBUS
ALE	«0»	Изменяется, как предустановленный для X-периферии
\overline{RD}	«1»	«1»
$\overline{WR}/\overline{WRL}$	«1»	«1»
\overline{WRH}	«1»	«1»

8.6 Арбитраж внешней шины

В системах с высокой производительностью может быть эффективным распределение внешних ресурсов, таких как банки памяти или периферийные устройства, среди нескольких микроконтроллеров. C167 поддерживает эту функцию с возможностью определения приоритетов доступа к внешней шине, т.е. к внешним устройствам.

Определение приоритетов шины позволяет внешним masters запрашивать шину C167 с помощью входа \overline{HOLD} . C167 отвечает на запрос через выход \overline{HLDA} , и выставляет на шину состояние высокого сопротивления. После этого новый master получает доступ к периферийным устройствам или к банкам памяти с помощью тех же самых линий шины, что и C167. Одновременно с этим C167 может продолжать выполнение программы до тех пор, пока не потребуются доступ к внешней шине. Все действия, для которых необходимы только внутренние ресурсы такие как память данных или внутренняя периферия, могут выполняться параллельно.

Когда C167 нуждается в доступе к внешней шине, оккупированной другим master шины, он выставляет сигнал требования на возвращение шины на вывод \overline{BREQ} .

Режим арбитража внешней шины может быть включен посредством установки «1» в битах HLDEN регистра PSW. В этом случае под управление ЕВС попадают три вывода микросхемы, предназначенные для арбитража шины независимо от их основных функций ввода-вывода: \overline{HOLD} , \overline{HLDA} и \overline{BREQ} . Бит HLDEN может быть установлен в «0» во время выполнения программы, в которой внешние ресурсы не могут быть востребованными другими masters шины. В этом случае C167 не реагирует на запрос \overline{HOLD} от других внешних masters. При установке бита HLDEN в «0» во время нахождения C167 в режиме захвата (выполнение кода из внутренней RAM/ROM), выход из этого режима может произойти только после того как \overline{HOLD} снова установится на неактивный уровень напряжения. Иными словами в этом случае текущий режим захвата будет завершен стандартным способом, и только на следующий запрос \overline{HOLD} не будет ответной реакции.

Для объединения двух микроконтроллеров C167 необходимо использовать дополнительные логические элементы, так как необходимо обеспечить соединение выходных сигналов \overline{BREQ} и \overline{HLDA} . Избежать необходимость использования дополнительной логики можно путем переключения одного из контроллеров в режим Slave, при этом \overline{HLDA} будет переключен на ввод. Это позволяет обеспечивать постоянное подключение slave микроконтроллера к master контроллера без использования дополнительной логики. Микроконтроллер можно перевести в режим slave при помощи установки в «1» бита DP6.7. В случае установки «0» в бите DP6.7 (предусматривается по умолчанию после RESET), используется master режим.

Примечание: В случае установки «0» в бите HLDEN выходы \overline{HOLD} , \overline{HLDA} и \overline{BREQ} могут сохранить свои альтернативные функции (арбитраж шины) после выключения режима арбитража.

Подключение masters шины

В тех случаях когда несколько микроконтроллеров C167 или C167 и другой master шины распределяют между собой внешние ресурсы, может появиться необходимость в использовании дополнительной логики. Она может потребоваться, того чтобы обеспечить работу master шины и также для того, чтобы дать возможность отдавшему управление C167 вернуть контроль над шиной. Дополнительная логика требуется в том случае, если другой master шины не может автоматически удалить запрос на захват, после завершения использования распределенных ресурсов.

В случае подключения двух C167 не требуются дополнительные внешние логические элементы. В этом случае один из микроконтроллеров

должен работать в режиме Master (предустанавливается после RESET при $DP6.7 = 0$), а другой в режиме Slave (выбирается при $DP6.7 = 1$).

В режиме Slave C167 меняет направление входа \overline{HLDA} , на ввод, в то время как master's \overline{HLDA} вывод остается выходным. В этом случае отпадает необходимость в использовании дополнительной логики для арбитража шины (ниже на рисунке).

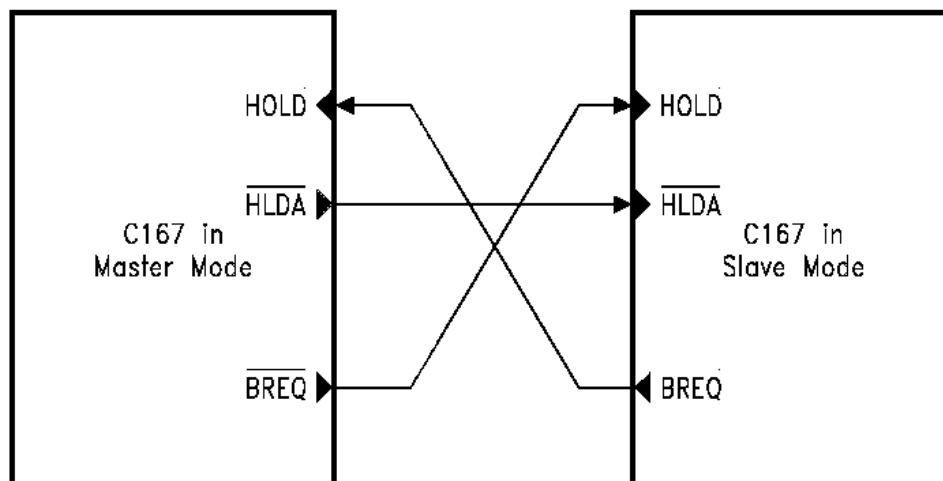


Рисунок 8-12

Распределение внешних ресурсов при использовании режима Slave

При включенном арбитраже шины ($HLDEN = 1$), автоматически контроллером ЕВС управляются три соответствующих вывода. Значения битов регистра направления порта остаются без изменения («0»). Режим MASTER построен таким образом, чтобы было достигнута совместимость с ранними моделями. Режим Slave может быть включен при помощи переключения направления \overline{BREQ} на выход ($DP6.7 = 1$). Установка «1» в этом бите никогда не требовалась в режиме Master для более ранних устройств.

Введение в режим захвата

Доступ к внешней шине C167 осуществляется путем установки нуля на входе \overline{HOLD} . После синхронизации этого сигнала, C167 завершает, в том случае, если имел место, текущий цикл работы внешней шины. После этого микроконтроллер освобождает внешнюю шину и посредством установки нуля на выходе \overline{HLDA} передает доступ к шине. Во время нахождения в режиме захвата C167 работает без интерфейса внешней шины, и как следствие:

- Выходы шины адреса и данных переходят в состояние высокого сопротивления
- посредством внутреннего драйвера ALE устанавливается в «0»
- Линии управления чтением и записью устанавливаются в «1»

- Выходы chip select устанавливаются в «1» (в режиме push/pull) или находятся в состоянии высокого сопротивления (в режиме с открытым коллектором)

В случае необходимости микроконтроллера C167 в режиме захвата совершить доступ к внешней шине, микроконтроллер устанавливает выход запроса шины \overline{BREQ} на активный уровень напряжения. Это необходимо для оповещения о необходимости передачи шины схеме арбитражи. \overline{BREQ} может перейти на активный уровень напряжения только во время режима захвата. Во время нормальной работы он всегда находится на неактивном уровне напряжения.

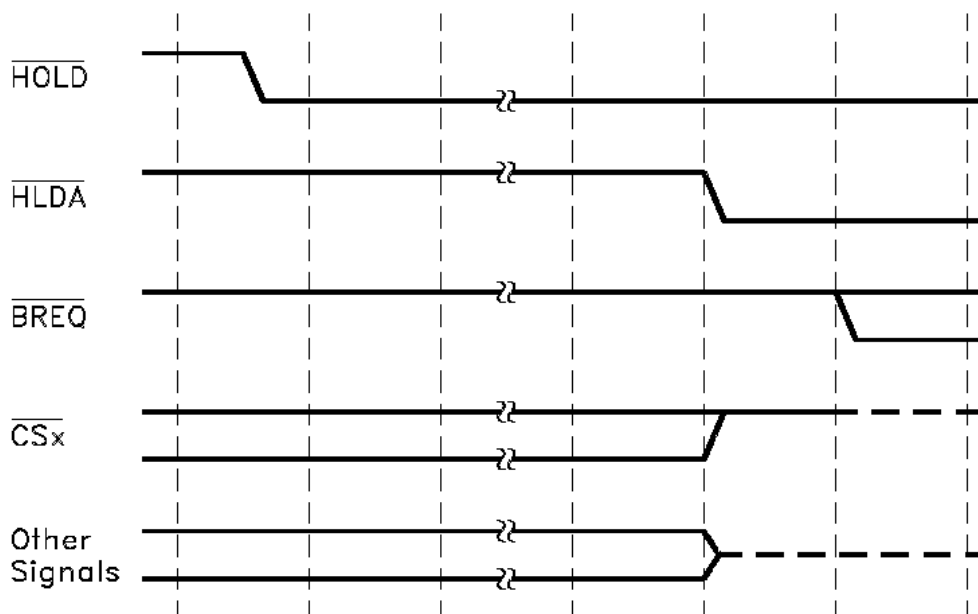


Рисунок 8-13

Арбитраж внешней шины, освобождение шины

Примечание: Прежде чем передать доступ, C167 завершит текущий цикл работы шины. На рисунке выше это обозначено пунктиром. При этом возможна задержка передачи шины.

В случае захвата шины, вывод P3.12 переключается назад на стандартную функцию общего назначения. Сохранение нулевого значения в бите DP3.12 гарантирует в режиме захвата высокое сопротивление на выходе.

Выход из режима захвата

Master шины может вернуть права доступа микроконтроллеру C167 путем установки «1» на входе \overline{HOLD} . После синхронизации этого сигнала C167 переводит вывод \overline{HLDA} в «1». После завершения обмена контрольными сигналами микроконтроллеру C167 возвращаются права доступа к внешней шине.

В зависимости от логики арбитража, контроль над шиной может быть возвращен C167 в двух случаях:

- Внешний master больше не нуждается в доступе к общим ресурсам, поэтому master возвращает права доступа
- В том случае когда C167 нуждается в доступе к общим ресурсам, он выставляет сигнал \overline{BREQ} . После этого внешняя логика арбитража переводит сигналы \overline{HLDA} другого master на неактивный уровень напряжения. Затем внешняя шина в зависимости от уровня приоритета текущего master может освободиться для C167.

Примечание: Невозможно прервать режим захвата установкой «0» в бите HLDEN.

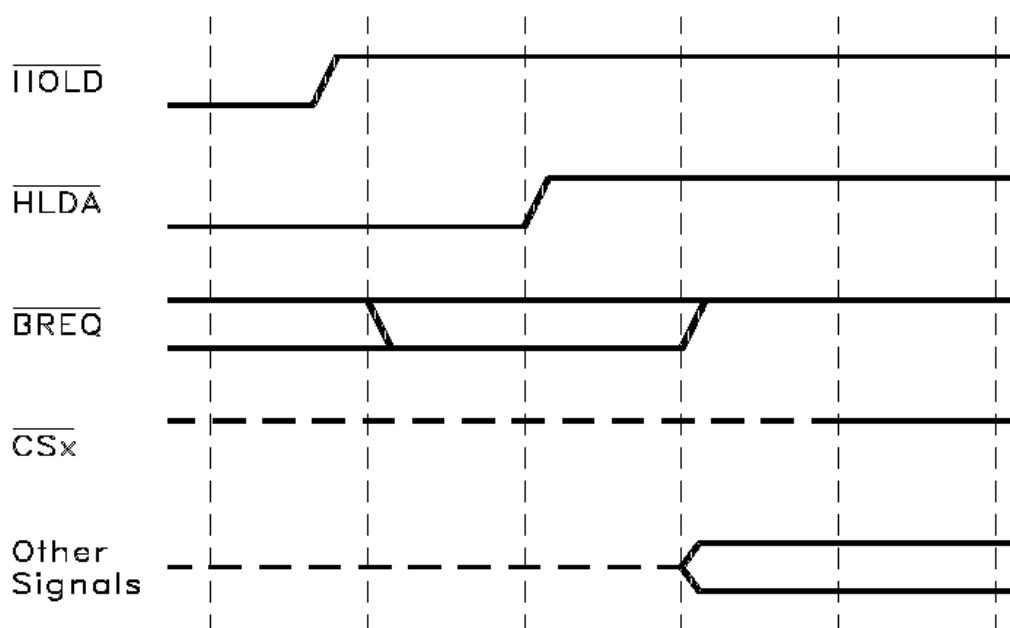


Рисунок 8-14

Арбитраж внешней шины, возвращение шины

Примечание: Отрицательный фронт сигнала \overline{BREQ} указывает на крайнюю точку возврата для переключения на процедуру возврата. Даже если \overline{BREQ} переходит в активный уровень напряжения раньше, то последовательность возврата инициализируется с помощью сигнала \overline{HOLD} . \overline{HOLD} и \overline{BREQ} соединены через внешнюю схему арбитража. Заметим также, что \overline{HOLD} может переходить на неактивный уровень напряжения без запроса шины C167.

8.7 Интерфейс XBUS

В архитектуру C167 включен внутренний XBUS интерфейс, позволяющий ядру осуществлять обмен данными с интегрированной

периферией. XBUS является внутренним отражением внешней шины, и работает по аналогичным законам.

XBUS-интерфейс микроконтроллера C167 поддерживает до трех X-периферийных устройств.

Для каждого периферийного устройства, подключенного к XBUS, имеется независимое адресное окно, управляемое регистрами XBCON и XADRS. Так как во многих случаях XBUS-интерфейс необходим для передачи данных только для небольшого количества регистров, то в регистре XADRS устанавливается наименьший возможный в данном случае размер адресного окна. Так как эта пара регистров управляет внутренней периферией лучше, чем внешней, то маскируемое программирование для этих регистров лучше, чем пользовательское.

Доступ к X-периферии обеспечивается аналогично доступу к внешней периферии, поэтому X-периферия поддерживает как побайтную адресацию, так и адресацию данных длиной в слово. При осуществлении доступа к X-периферии возможно наличие независимой адресной шины. Вектора прерываний и настроечные выводы (порта 0) обеспечивают интеграцию X-периферии в архитектуру микроконтроллера.

9 Блоки таймеров основного назначения

Блоки таймеров основного назначения GPT1 и GPT2 имеют гибкую многофункциональную структуру. Таймеры могут использоваться для измерения времени, подсчета количества событий, измерения длительности импульсов, генерации импульсов, умножения частоты, и других функций. В два блока объединены пять 16-битных таймеров.

Блок GPT1 содержит 3 таймера/счетчика с максимальным разрешением 400 нс (при частоте ЦПУ 20МГц). Блок GPT2 содержит 2 таймера/счетчика с максимальным разрешением 200 нс, а также 16-битный регистр захвата и перезагрузки (CAPREL). Все таймеры обоих блоков могут работать независимо друг от друга в различных режимах: gated таймер или режим счетчика. Также возможно объединение таймеров из одного блока. Вспомогательные таймеры GPT1 также могут быть настроены в режим регистра перезагрузки или захвата таймера ядра. Дополнительный CAPREL-регистр блока GPT2 поддерживает операции захвата и перезагрузки. Таймер T6 может быть объединен с таймерами блока CAPCOM (T0, T1, T7 и T8). Для каждого блока таймеров имеют место альтернативные функции параллельных портов, и каждый блок имеет собственные прерывания.

9.1 Блок GPT1

С точки зрения программиста, блок GPT1 содержит следующий набор SFR-регистров:

Ports & Direction Control Alternate Functions	Data Registers	Control Registers	Interrupt Control
<div> <div>ODP3</div> <div>DP3</div> <div>P3</div> <div>P5</div> </div> <div> <div>T2IN/P3.7</div> <div>T3IN/P3.6</div> <div>T4IN/P3.5</div> <div>T3OUT/P3.3</div> </div> <div> <div>T2EUD/P5.15</div> <div>T3EUD/P3.4</div> <div>T4EUD/P5.14</div> </div>	<div>T2</div> <div>T3</div> <div>T4</div>	<div>T2CON</div> <div>T3CON</div> <div>T4CON</div>	<div>T2IC</div> <div>T3IC</div> <div>T4IC</div>
<div>ODP3</div> <div>DP3</div> <div>P3</div> <div>T2CON</div> <div>T3CON</div> <div>T4CON</div>	<div>Port 3 Open Drain Control Register</div> <div>Port 3 Direction Control Register</div> <div>Port 3 Data Register</div> <div>GPT1 Timer 2 Control Register</div> <div>GPT1 Timer 3 Control Register</div> <div>GPT1 Timer 4 Control Register</div>	<div>T2</div> <div>T3</div> <div>T4</div> <div>T2IC</div> <div>T3IC</div> <div>T4IC</div>	<div>GPT1 Timer 2 Register</div> <div>GPT1 Timer 3 Register</div> <div>GPT1 Timer 4 Register</div> <div>GPT1 Timer 2 Interrupt Control Register</div> <div>GPT1 Timer 3 Interrupt Control Register</div> <div>GPT1 Timer 4 Interrupt Control Register</div>

Рисунок 9-1

SFR-регистры и выводы портов, связанные с GPT1

Все три таймера блока GPT1 (T2, T3 и T4) могут работать в трех основных режимах: таймер, gated таймер и режим счетчика. Все таймеры могут производить отсчет как в положительном, так и в отрицательном направлении. Для каждого таймера имеет место вход, являющийся альтернативной функцией вывода порта 3. Этот вход может использоваться в качестве управления gate в режиме gated таймер или в качестве входа отсчета для режима счетчика. Направление отсчета может быть запрограммировано программно, а также может изменяться по ходу работы при помощи внешнего управляющего сигнала. Каждое переполнение таймера T3 или достижения им нуля может быть выведено через вывод порта в качестве функции альтернативного вывода. Вспомогательные таймеры T3 и T4 могут быть связаны с таймером ядра T2, или могут использоваться в качестве регистров захвата значения таймера ядра.

Текущее содержимое каждого таймера может быть прочтено и изменено с помощью ЦПУ, посредством доступа к соответствующим регистрам таймеров T2, T3 и T4, расположенным в не адресуемом побитно пространстве SFR-регистров. В случае совершения записи значения в регистр таймера при помощи ЦПУ перед инкрементированием или декрементированием значения таймера или перед совершением захвата значения таймера, более высокий приоритет имеет команда ЦПУ.

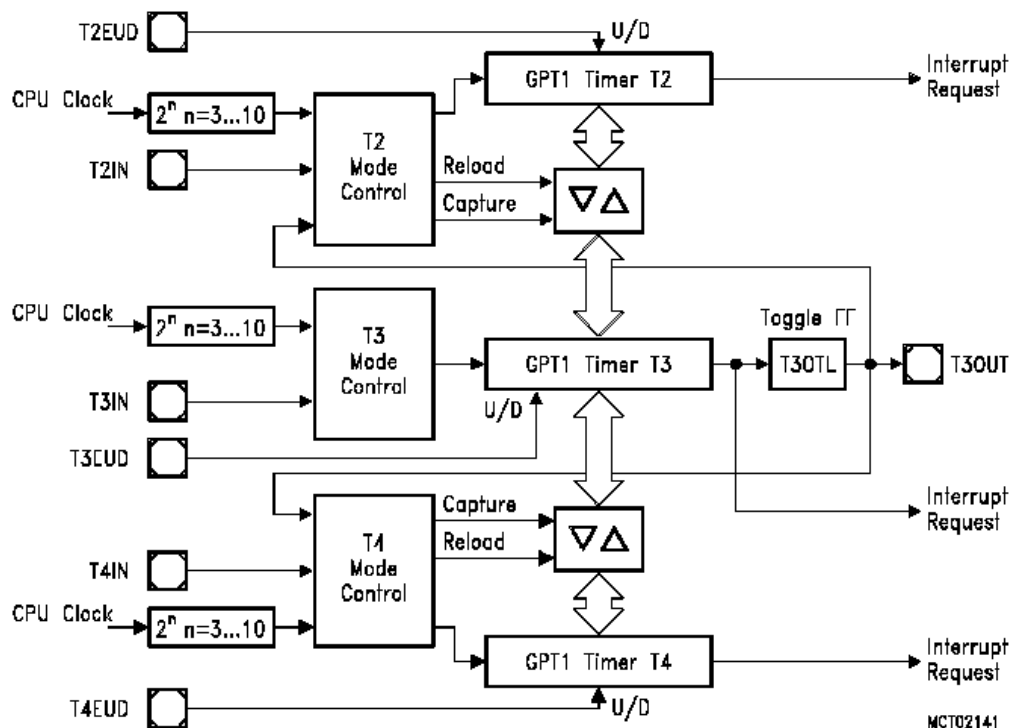


Рисунок 9-2

Блок-схема GPT1

Ядро таймера T3 GPT1

Ядро таймера T3 настраивается и управляется с помощью побитно адресуемого регистра T3CON.

SFR
ТЗCON (FF42_H/A1_H) Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	T3 OTL	T3OE	T3UDE	T3UD	T3R	T3M			T3I		
-	-	-	-	-	rw	rw	rw	rw	rw	rw			rw		

Бит	Функция
T3I	Выбор входа таймера 3 Зависит от режима работы, см. соответствующий раздел
T3M	Управление режимом работы таймера 3 0 0 0: Режим таймера 0 0 1: Режим счетчика 0 1 0: Gated таймер с низким активным уровнем напряжения 0 1 1: Gated таймер с высоким активным уровнем напряжения 1 x x: зарезервировано, не использовать
T3R	Бит работы таймера 3 T3R = 0: T3 стоит T3R = 1: T3 работает
T3UD	Управление направлением счета T3^{*)}
T3UDE	Разрешение внешнего управления направлением счета таймера 3^{*)}
T3OE	Разрешение использования альтернативного вывода порта T3OE = 0: Запрещено T3OE = 1: Разрешено
T3OTL	Выходной триггер таймера 3 Меняет состояние при каждом переполнении и опустошении таймера. Может программно изменять свое значение

^{*)} Для достижения эффекта от установки значений в битах T3UD и T3UDE необходимо следовать таблице направлений, представленной ниже.

Бит работы таймера 3

Таймер может начать работать или остановиться по сигналу программы, после установки значения в бите T3R. В режиме gated таймера помимо этого таймер будет работать только в том случае, если gate находится в активном состоянии (1 или 0, в зависимости от запрограммированного значения).

Управление направлением счета

Направление счета ядра таймера может быть установлено либо программно, либо при помощи внешнего сигнала T3EUD, являющегося альтернативной функцией вывода порта P3.4. Эта возможность можно выбрать с помощью установки «1» в бите T3UD и T3UDE регистра T3CON. Для программного управления направление счета устанавливается в бите T3UD. При T3UDE = 1, вывод T3EUD используется в качестве источника управления направлением. Однако при этом для изменения направления счета можно использовать бит T3UD, как показано ниже в таблице. Направление счета можно менять вне зависимости от того, работает таймер или нет.

Когда вывод T3EUD/P3.4 используется для управления направлением счета, этот вывод порта должен быть сконфигурирован на ввод, т.е. DP3.4 = 0.

Управление направлением счета T3 блока GPT1

Вывод TxEUD	Бит TxDEUD	Бит TxUD	Направление счета
X	0	0	Инкрементирование
X	0	1	Декрементирование
0	1	0	Инкрементирование
1	1	0	Декрементирование
0	1	1	Декрементирование
1	1	1	Инкрементирование

Примечание: Управление направлением можно применять как для таймера T3, так и для вспомогательных таймеров T2 и T4.

Выходной триггер таймера 3

Переполнение и опустошение таймера T3 изменяют значение бита T3OTL регистра T3CON. Дополнительно к этому значение бита T3OTL можно изменять программно. Использование альтернативной функции T3OUT/P3.3 включается в бите T3OE регистра T3CON. Для использования этой функции, необходимо записать «1» в триггер данных порта P3.3, и также необходимо настроить P3.3 на выход, путем установки DP3.3=1. Если T3OE = 1, то T3OUT выводит состояние T3OTL. В том случае когда T3OE = 0, вывод T3OUT может использоваться для ввода-вывода основного назначения.

В дополнение к этому, T3OTL может использоваться при переполнении и опустошении таймера в качестве входа тактового сигнала счетчика, или как источник переключения для функции перезагрузки вспомогательных таймеров T2 и T4. В случае использования этой функции,

состояние T3OTL не может выведено через T3OUT, потому что при этой функции обеспечивается внутреннее соединение.

Таймер 3 в режиме таймера

Режим таймера для T3 выбирается путем установки 000_B в битовом поле T3M регистра T3CON. В этом режиме T3 изменяет свое содержимое, в соответствии с системным тактовым генератором (CPU clock). При этом частота определяется при помощи программируемого делителя. Коэффициенты программируемого делителя выбираются в соответствии со значением битового поля T3I. Входная частота f_{T3} таймера T3 и длительность такта r_{T3} линейно зависят от частоты ЦПУ.

$$f_{T3} = \frac{f_{CPU}}{8 \cdot 2^{\langle T3I \rangle}}$$

$$r_{T3}(\text{мкс}) = \frac{8 \cdot 2^{\langle T3I \rangle}}{f_{CPU}(\text{МГц})}$$

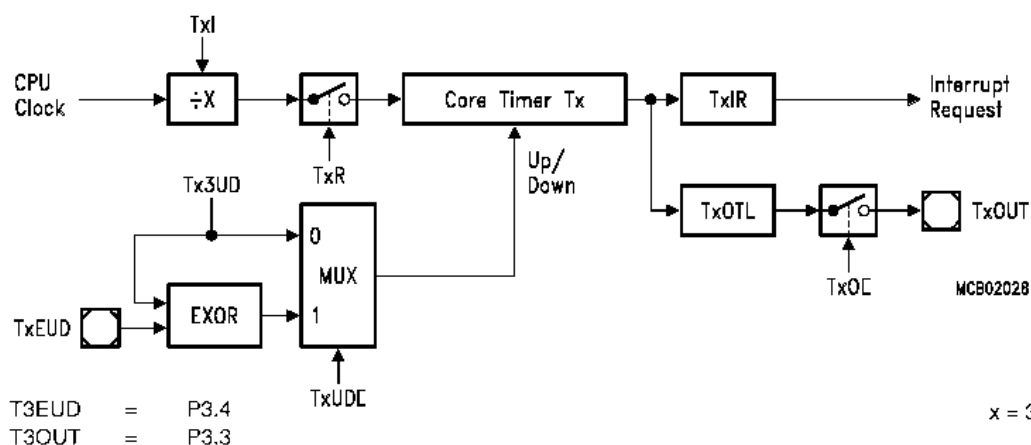


Рисунок 9-3

Блок-схема T3 в режиме таймера

В таблице приведены ниже возможные значения частот, величин квантов и периодов при использовании частоты ЦПУ 20МГц. Эту таблицу также можно использовать в режиме Gated таймера и также можно использовать для вспомогательных таймеров T2 и T4 в режиме таймера и режиме gated таймера. Заметим, что некоторые режимы могут иметь только три значащих цифры.

$f_{CPU} = 20\text{МГц}$	Выбор входа таймера T2I/T3I/T4I							
	000 _B	001 _B	010 _B	011 _B	100 _B	101 _B	110 _B	111 _B
Коэффициент деления	8	16	32	64	128	256	512	1024
Входная частота	2.5 МГц	1.25 МГц	625 кГц	312.5 кГц	156.25 кГц	78.125 кГц	39.06 кГц	19.53 кГц
Квантование	400 нс	800 нс	1.6 мкс	3.2 мкс	6.4 мкс	12.8 мкс	25.6 мкс	51.2 мкс
Период	26 мс	52.5 мс	105 мс	210 мс	420 мс	840 мс	1.68 с	3.36 с

Таймер 3 в режиме gated таймера

Режим gated таймера T3 выбирается путем установки «010» или «011» в поле T3M регистра T3CON. Значение бита T3M.0 (T3CON.3) указывает на активный уровень напряжения на входе **gate**. В режиме gated таймера входная частота устанавливается по правилам аналогичным режиму таймера. Однако сигнал тактового генератора на входе таймера T3 отключается после подачи сигнала на вход T3IN, являющегося альтернативной функцией вывода порта P3.6.

Для включения этого режима, необходимо настроить P3.6 на вход, т.е. DP3.6 = 0.

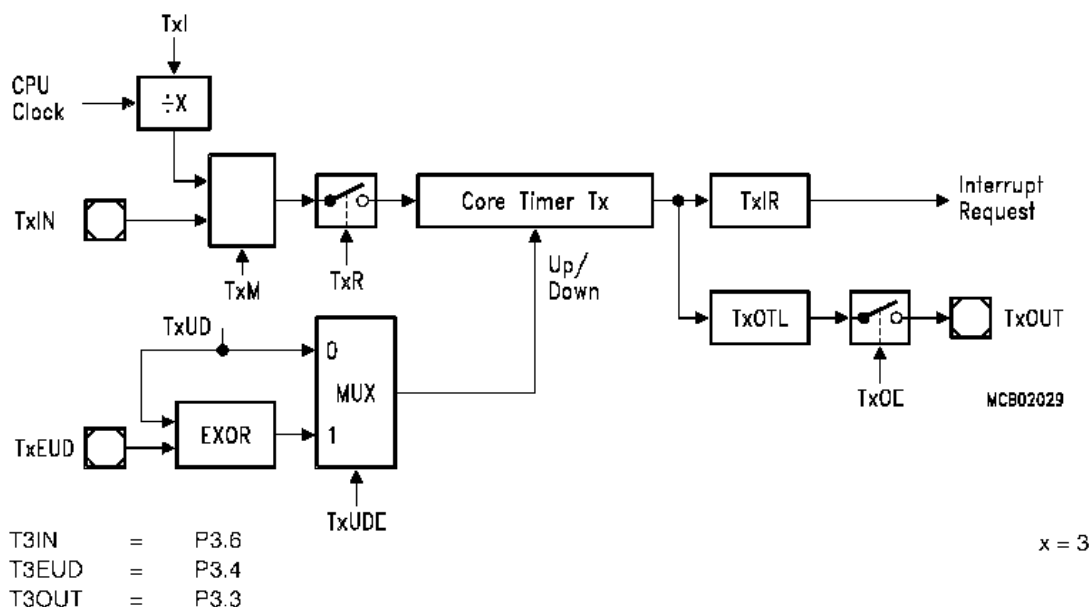


Рисунок 9-4

Блок-схема T3 в режиме gated таймера

В случае установки T3M.0=0, таймер будет работать, когда на T3IN приходит сигнал с низким уровнем напряжения. Высокий уровень напряжения на этом входе останавливает работу таймера. В том случае когда

T3M.0=1, таймер переходит в рабочее состояние при подаче высокого уровня напряжения на вход T3IN. Таймер также можно программно включить или выключить, изменяя значение бита T3R. Таймер будет находиться в рабочем режиме только в том случае, когда выполняются сразу оба вышеперечисленных условия (T3R=1, T3IN на активном уровне напряжения).

Примечание: Подача gate сигнала на вход T3IN не приводит к подаче запроса на прерывание.

Т3 в режиме счетчика

Режим счетчика таймера Т3 выбирается путем установки 001 в битовом поле T3M регистра T3CON. В этом режиме отсчеты таймера производятся по фронту сигнала на входе T3IN, являющегося альтернативной функцией P3.6. Фронт сигнала, приводящего к увеличению или уменьшению значения в счетчике, может быть как положительный, так и отрицательный, также оба фронта могут производить изменения значения счетчика. Битовое поле T3I управляющего регистра T3CON, выбирает фронт (ниже в таблице).

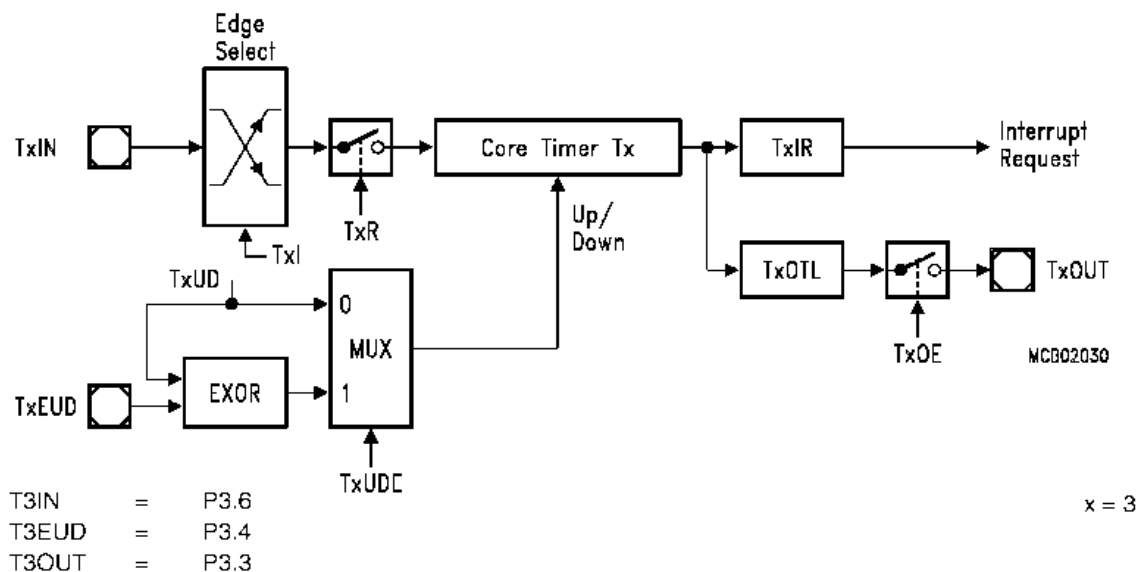


Рисунок 9-5

Блок-схема Т3 в режиме счетчика

Выбор фронта срабатывания ТЗ в режиме счетчика

ТЗІ	Фронт срабатывания для увеличения/уменьшения значения счетчика
0 0 0	Нет, счетчик отключен
0 0 1	Положительный фронт на входе ТЗІN
0 1 0	Отрицательный фронт на входе ТЗІN
0 1 1	Оба фронта на входе ТЗІN
1 x x	Зарезервирован. Не использовать.

Для работы счетчика необходимо настроить на ввод ТЗІN/P3.6, т.е. DP3.6 = 0. Максимальная входная частота, доступная в режиме счетчика, составляет $f_{CPU}/16$ (1.25 МГц при частоте ЦПУ 20МГц). Для уверенности в том что фронт входного сигнала счетчика воспринимается корректно, необходимо как минимум в течении 8 тактов ЦПУ сохранять значение на входе постоянным.

Вспомогательные таймеры Т2 и Т4 блока GPT1

Оба вспомогательных таймера имеют точно такой же набор функций как и ТЗ. Их можно настроить на работу в режиме таймера, gated таймера или в режиме счетчика. При этом возможен такой же способ настройки частоты и сигнала счета, что и для ТЗ. В дополнение к этим трем режимам вспомогательные таймеры имеют дополнительные режимы для работы вместе с ТЗ, также их можно использовать в качестве регистров захвата или перезагрузки таймера ТЗ.

Примечание: У вспомогательных таймеров отсутствуют выходные триггеры, и для них нет альтернативных функций портов.

Индивидуальные настройки таймеров Т2 и Т4 можно совершать в побитно адресуемых регистрах управления Т2CON и Т4CON. Заметим, что идентичные функции для всех трех таймеров, размещаются на одинаковых позициях в регистрах управления.

T2CON (FF40_H/A0_H)

SFR

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	T2 UDE	T2UD	T2R	T2M			T2I		
-	-	-	-	-	-	-	Rw	rw	rw	rw			rw		

Бит	Функция
TxM	Управление режимом работы таймера 3 0 0 0: Режим таймера 0 0 1: Режим счетчика 0 1 0: Gated таймер с низким активным уровнем напряжения 0 1 1: Gated таймер с высоким активным уровнем напряжения 1 0 0: Режим перезагрузки 1 0 1: Режим захвата 1 1 x: Зарезервировано, не использовать.

Регистр T4CON имеет идентичную структуру регистру T2CON.

T4CON: FF44_H/A2

При использовании таймера T2 и T4 в режиме gated таймера необходимо учесть отсутствие выводов портов с альтернативными функциями, необходимыми для работы в этом режиме.

Таймеры T2 и T4 в режиме счетчика

Режим счетчика в вспомогательных таймерах T2 и T4 выбирается с помощью установки 001 в битовом поле TxM регистра TxCON. В режиме счетчика изменение значения таймера производится либо по приходу фронта сигнала на входе TxIN, либо при переключении триггера T3OTL.

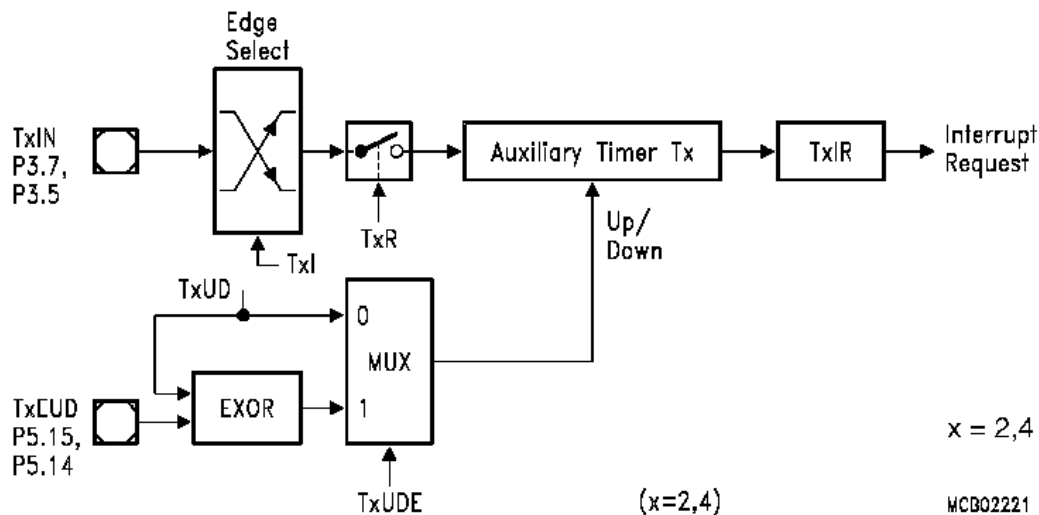


Рисунок 9-6

Блок-схема вспомогательного таймера в режиме счетчика

Переключение таймера можно производить как по положительному, так и по отрицательному фронту, также возможен режим переключения одновременно по обоим фронтам, или по переключению бита T3OTL. Режим можно выбрать путем установки значений в битовом поле TxI регистра TxCON:

T2I/T4I	Фронт срабатывания для инкрементирования/ декрементирования значения счетчика
0 0 0	Нет, счетчик отключен
0 0 1	Положительный фронт на входе TxIN
0 1 0	Отрицательный фронт на входе TxIN
0 1 1	Оба фронта на входе TxIN
1 0 1	Положительное переключение триггера T3OTL
1 1 0	Отрицательное переключение триггера T3OTL
1 1 1	Любое переключение триггера T3OTL

Примечание: Только при переполнении и опустошении таймера T3 будет производиться изменение значения счетчиков T2/T4 в режиме T3OTL. Программное изменение значения T3OTL не будет влиять на изменение значений в таймерах T2/T4.

Объединение таймеров

Благодаря возможности использования бита T3OTL в качестве входа вспомогательного таймера в режиме счетчика, имеется возможность объединения таймера T3 с вспомогательным таймером. В зависимости от выбранного направления переключения бита T3OTL для изменения значения вспомогательного таймера, можно сформировать 32-битный или 33-битный таймер/счетчик.

- **32-битный таймер/счетчик:** При использовании одновременно и положительного и отрицательного изменения значения T3OTL в качестве тактового сигнала вспомогательного таймера, осуществляется режим 32 битного счетчика/таймера. Таким образом два таймера образуют один 32-битный таймер.

- **33-битный таймер/счетчик:** При использовании только либо положительного либо отрицательного изменения значения T3OTL, объединенный таймер будет переключаться только каждое второе переполнение или опустошение T3. Таким образом создается 33-битный таймер (16-битный T3 + T3OTL + 16-битный вспомогательный таймер).

Направление счета в двух объединенных регистрах может быть различным. Это позволяет создавать широкий диапазон различных конфигураций.

В режиме объединенных таймеров T3 может работать в режиме таймера, gated таймера или режиме счетчика.

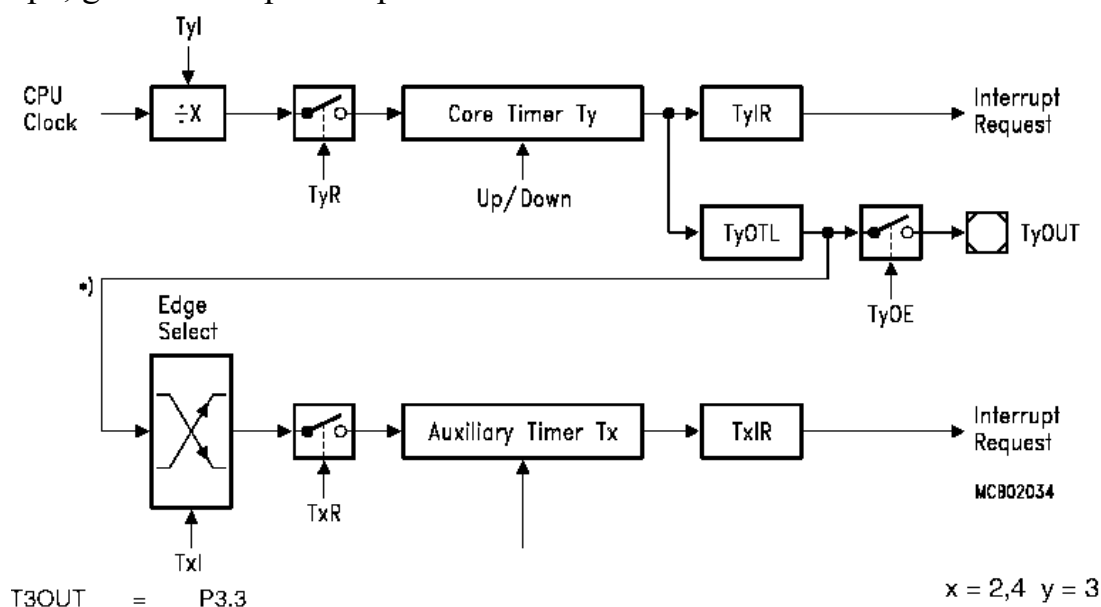


Рисунок 9-7

Объединение T3 и вспомогательного таймера

Вспомогательный таймер в режиме перезагрузки

Режим перезагрузки вспомогательных таймеров T2 и T4 можно установить путем установки 100 в битовом поле TxM регистра TxCON. В режиме перезагрузки в таймер T3 записывается содержимое регистра вспомогательного таймера по одному из двух различных управляющих сигналов. Сигналы управления аналогичны управляющим сигналам для режима счетчика: фронт на входе вспомогательного таймера или изменение значения триггера T3OTL.

Примечание: При программировании режима перезагрузки, счет в вспомогательном таймере T2 или T4 останавливается вне зависимости от содержимого флага TxR.

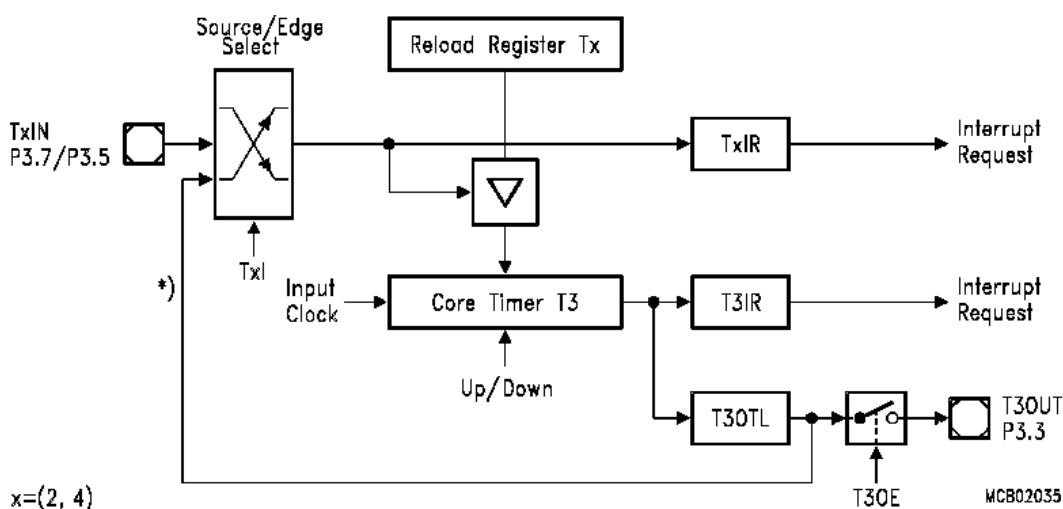


Рисунок 9-8

Вспомогательный таймер GPT1 в режиме перезагрузки

Одновременно с подачей управляющего сигнала для загрузки содержимого T2 или T4 в регистр таймера T2, выставляется флаг запроса на прерывание (T2IR или T4IR).

Примечание: Когда в качестве управляющего сигнала выбирается T3OTL, также выставляется флаг запроса на прерывание T3IR, выявляющий переполнение или опустошение таймера T3.

Программное изменение значения T3OTL не оказывает влияния на этот режим.

В режиме перезагрузки в зависимости от типа управляющего фронта сигнала T3OTL может быть создано несколько различных режимов. В зависимости от выбранного типа перехода могут совершаться следующие функции:

- При выборе обоих типов переключений, в качестве управляющих для перезагрузки, в таймер T3 будет загружаться содержимое

вспомогательного таймера при каждом переполнении и опустошении. Этот режим является стандартным режимом.

- В случае выбора только положительного или отрицательного типа переключения Т3ОТL, в качестве управляющего сигнала для перезагрузки, в таймер Т3 будет записываться содержимое вспомогательного таймера только при каждом втором переполнении или опустошении.

- Использование «однопереходного» режима для обоих таймеров позволяет создавать гибкий режим широтно-импульсной модуляции. Необходимо один из вспомогательных таймеров запрограммировать на перезагрузку Т3 по положительному переходу Т3ОТL, другой необходимо запрограммировать на перезагрузку по отрицательному переходу. В этом случае осуществляется последовательная перезагрузка обоих вспомогательных таймеров.

Рисунок, расположенный ниже, показывает пример создания ШИМ-сигнала, при использовании данного механизма. Значение Т2 устанавливает длительность импульса ШИМ-сигнала (положительный переход Т3ОТL), значение Т4 определяет длительность паузы ШИМ-сигнала. ШИМ-сигнал может быть выведен через вывод Т3ОUТ, в том случае если: Т3ОЕ = 1, Р3.3 = 1, DР3.3 = 1. При использовании этого метода длительность импульса или паузы можно изменять в широком диапазоне значений.

Примечание: Триггер Т3ОТL является доступным программно и может быть изменен. Значение Т3ОТL можно программно изменить для модификации ШИМ-сигнала. Однако при этом не произойдет перезагрузки таймера Т3.

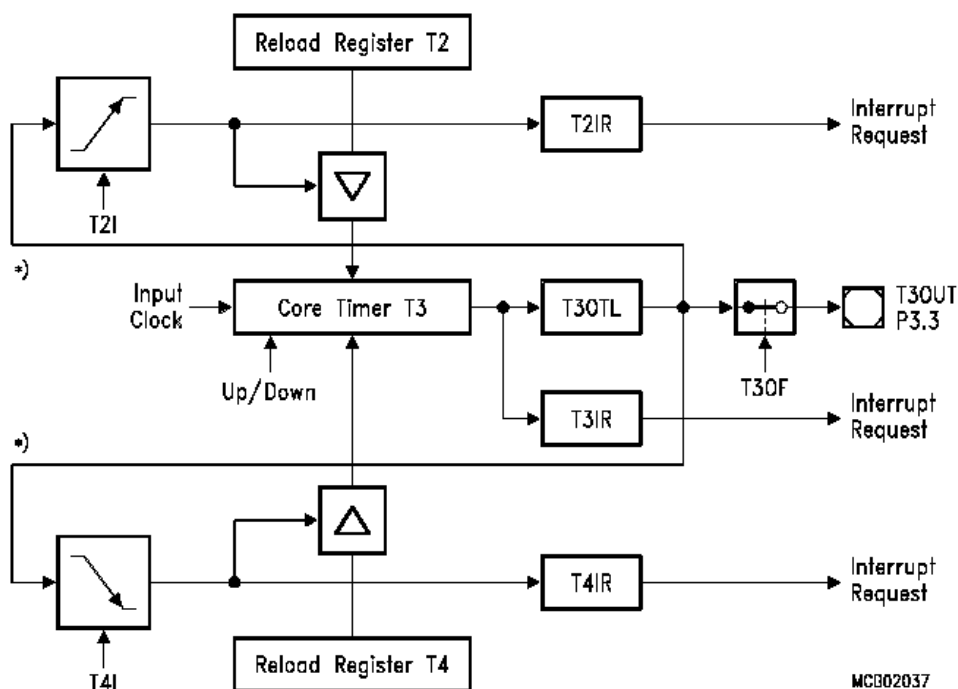


Рисунок 9-9

Конфигурация перезагрузки таймера GPT1 для создания ШИМ-сигнала

Примечание: Хотя и допускается, следует избегать одинаковых режимов перезагрузки обоих таймеров. В этом случае оба перезагружающих регистра попытаются загрузить свое значение в T3. Если выбрана эта комбинация, пренебрегается содержимое T2, и производится перезагрузка T4.

Режим захвата вспомогательного таймера

Режим захвата для вспомогательных таймеров T2 и T4 можно выбрать путем установки 101 в битовом поле TxM регистра TxCON. В режиме захвата содержимое T3 заключается в регистр вспомогательного таймера, в ответ на фронт сигнала с внешнего входа TxIN вспомогательного таймера. Фронт срабатывания сигнала, может быть как положительным, так и отрицательным либо одновременно и положительным и отрицательным.

Два младших значащих бита битового поля TxI используются для выбора необходимого фронта реагирования (см. таблицу в разделе посвященном режиму счетчика), значение старшего бита TxI.2 не имеет значения в этом режиме. Рекомендуется в этом бите устанавливать «0».

Примечание: При программировании этого режима счет в вспомогательном таймере T2 или T4 останавливается вне зависимости от значения флага TxR.

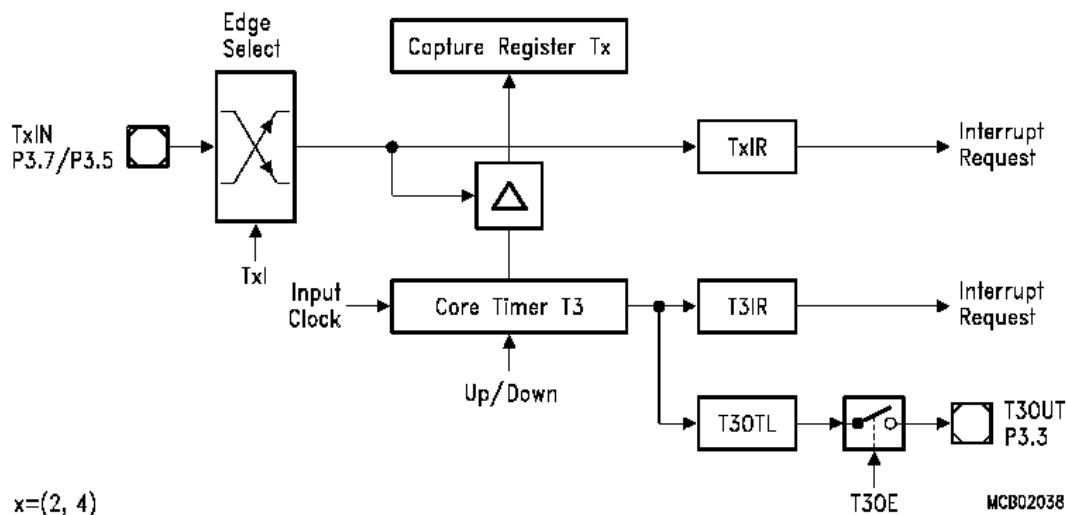


Рисунок 9-10

Вспомогательный таймер GPT1 в режиме захвата

Одновременно с приходом фронта управляющего сигнала, содержимое таймера T3 загружается в регистр вспомогательного таймера, при этом устанавливается флаг запроса на прерывание TxIR.

Примечание: Необходимо установить в 0 значения битов управления направлением DP3.7 для T2IN)и DP3.5 для T4IN. Для того, чтобы определить фронт, необходимо в течении как минимум 8 тактов ЦПУ сохранять уровень напряжения сигнала управления без изменения.

Управление прерываниями таймеров GPT1 блока

При переходе таймера из FFFF_H в 0000_H или при переходе из 0000_H в FFFF_H, устанавливается флаг запроса на прерывание (T2IR, T3IR или T4IR) регистра TxIC. В тех случаях когда устанавливается «1» в битах разрешения прерывания (T2IE, T3IE или T4IE регистра TxIC), возможно совершение прерывания по соответствующему вектору (T2INT, T3INT или T4INT) или возможно совершение РЕС-обслуживания. Для каждого из трех таймеров имеется свой регистр управления прерываниями.

SFR

T2IC (FF60_H/B0_H)

Значение после RESET: --00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								T2IR	T2IE	ILVL				GLVL	
-								rw	rw	rw				rw	

Другие регистры имеют аналогичную структуру и отличаются только адресом:

T3IC (FF62_H/B1_H)

T4IC (FF64_H/B2_H)

В случае необходимости подробности можно увидеть в описании регистров управления прерываниями.

9.2 Блок таймеров GPT2

С точки зрения программиста блок GPT2 состоит из набора SFR-регистров. Также к ним можно отнести регистры портов и регистры направления портов, альтернативные функции которых используются для функций блока GPT2.

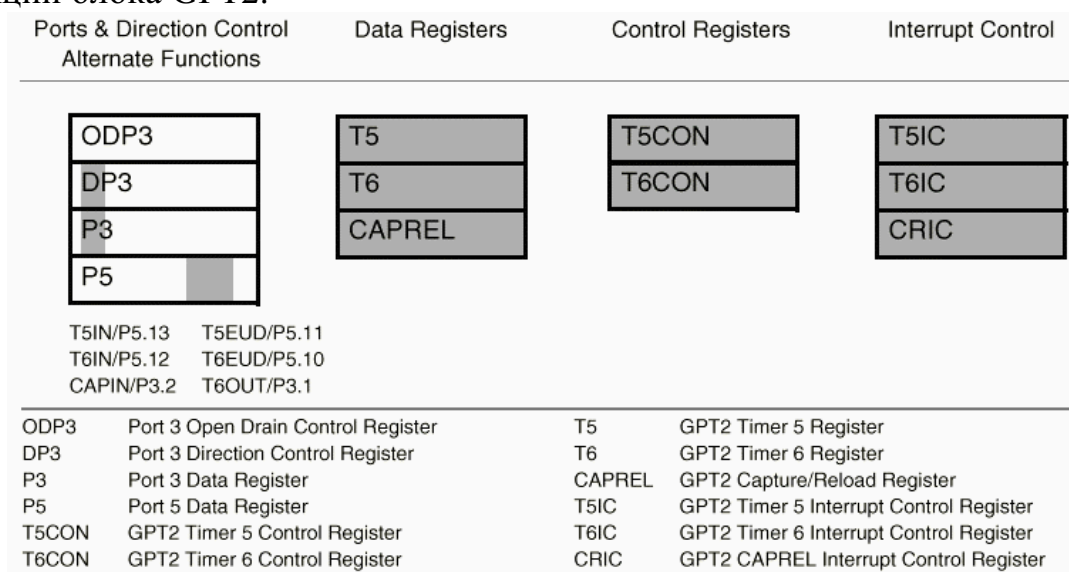


Рисунок 9-11

SFR-регистры и выходы портов, связанные с блоком таймеров GPT2

Блок таймеров GPT2 поддерживает высокую точность измерения времени с максимальным разрешением 200 нс (при частоте ЦПУ 20МГц). В GPT2 входят два таймера T5 и T6, а также 16-битный регистр захвата и перезагрузки. Таймер T6 является основным таймером, а T5 – дополнительным таймером GPT2.

Для каждого таймера имеется вывод порта, с альтернативной функцией, предназначенной для обслуживания режима gated таймера, или в качестве входа сигнала счетчика. Направление счета (положительное / отрицательное) можно запрограммировать, или можно динамически изменять в зависимости от внешнего сигнала на входе. Переполнение и опустошение таймера T6 отслеживаются в бите T6OTL. Значение этого бита можно вывести в качестве альтернативной функции порта. Также возможен режим перезагрузки значения для таймера T6 из регистра CAPREL.

С помощью T6OTL можно объединять таймеры T6 и T5. Объединение T6 с таймерами CAPCOM блока возможно осуществлять напрямую. По

фронту внешнего сигнала возможно сохранять содержимое таймера T5 в регистре CAPREL, при этом возможно осуществлять сброс значения таймера T5. Счет в таймерах T6 и T5 возможен как в положительном, так и в отрицательном направлении, Значение таймера можно изменить или прочитать с помощью не адресуемых побитно регистров.

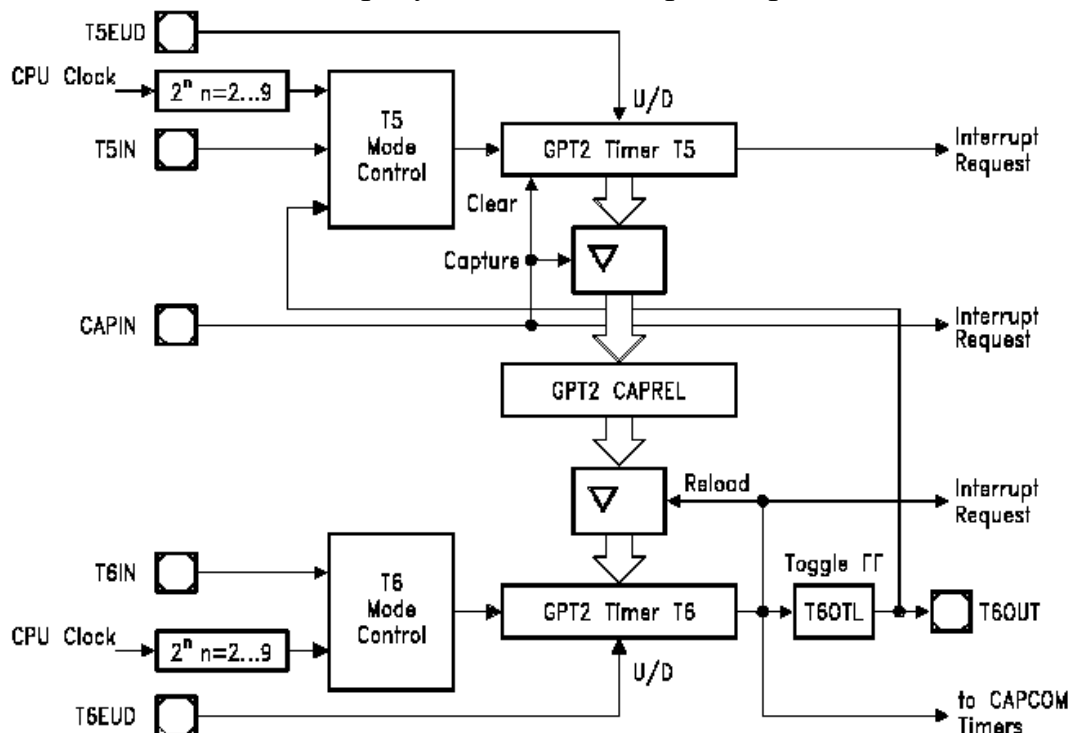


Рисунок 9-12

Блок-схема GPT2

Таймер T6 GPT2

Управление режимом работы таймера T6 осуществляется в побитно адресуемом регистре T6CON.

SFR
T6CON (FF48_H/A4_H) Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T6SR	-	-	-	-	T6OTL	T6OE	T6UDE	T6UD	T6R	T6M			T6I		
-	-	-	-	-	rw	rw	rw	rw	rw	rw			rw		

Бит	Функция
T6I	Выбор входа таймера 6 Зависит от режима работы, см. соответствующий раздел
T6M	Управление режимом работы таймера 6 0 0 0: Режим таймера 0 0 1: Режим счетчика 0 1 0: Gated таймер с низким активным уровнем напряжения 0 1 1: Gated таймер с высоким активным уровнем напряжения 1 x x: зарезервировано, не использовать
T6R	Бит работы таймера 6 T6R = 0: T6 стоит T6R = 1: T6 работает
T6UD	Управление направлением счета T6 ^{*)}
T6UDE	Разрешение внешнего управления направлением счета таймера 6 ^{*)}
T6OE	Разрешение функции использования альтернативного вывода T6OE = 0: Запрещена T6OE = 1: Разрешена
T6OTL	Выходной триггер таймера 6 Меняет состояние при каждом переполнении и опустошении таймера. Может менять свое значение программно
T6SR	Включение режима перезагрузки таймера 6 T6SR = 0: отключен режим T6SR = 1: включен режим

^{*)} Для достижения эффекта от установки значений в битах T6UD и T6UDE необходимо следовать таблице направлений, представленной ниже.

Бит работы таймера 6

Таймер начинает работу или останавливает счет по команде программы, с помощью изменения значения бита T6R. В режиме gated таймера, таймер будет работать только в том случае, если gate находится в активном состоянии (1 или 0, в зависимости от запрограммированного режима), и T6R = 1.

Управление направлением счета

Направление счета ядра таймера может быть изменено либо программно, либо при помощи подачи внешнего сигнала T6EUD, являющегося альтернативной функцией вывода порта P5.10. Эта возможность выбирается в бите T6UD и T6UDE регистра T6CON. При программном управлении, направление счета можно выбрать в бите T6UD. При T6UDE = 1, вывод T6EUD используется в качестве источника управления направлением счета. Однако одновременно с этим возможно использование бита T6UD для изменения направления счета, как показано ниже в таблице. Направление счета можно поменять вне зависимости от того, работает таймер или нет.

Управление направлением счета T6 блока GPT2

Вывод TxEUD	Бит TxUDE	Бит TxUD	Направление счета
X	0	0	Инкрементирование
X	0	1	Декрементирование
0	1	0	Инкрементирование
1	1	0	Декрементирование
0	1	1	Декрементирование
1	1	1	Инкрементирование

Примечание: Управление направлением для таймера T6 соответствует таймеру T5.

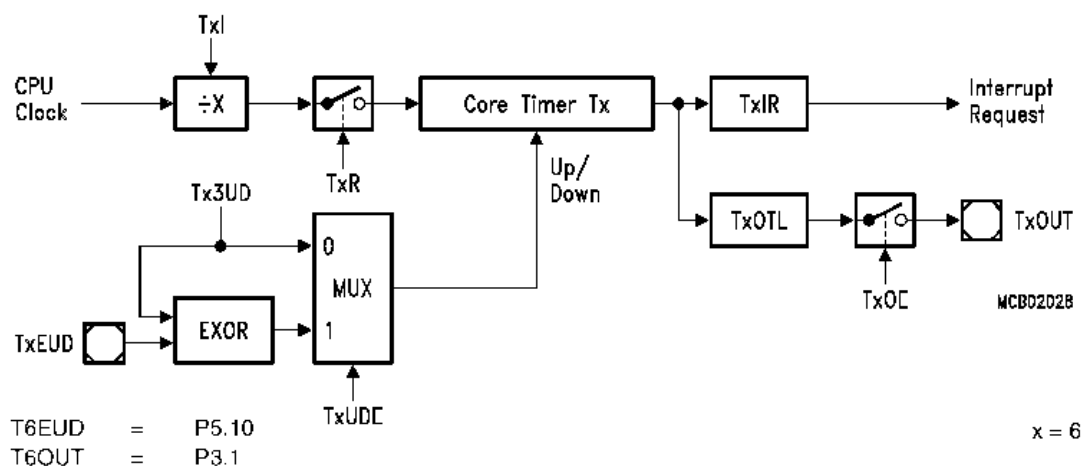
Выходной триггер таймера T6

При переполнении или опустошении таймера T6 производится переключение значения бита T6OTL регистра T6CON. Значение бита T6OTL можно также изменять программно. Бит T6OE регистра T6CON включает использование альтернативной функции T6OUT/P3.1. Для использования этой функции, необходимо записать в «1» в триггер данных порта P3.1, и также необходимо настроить P3.1 на выход в соответствующем бите регистра управления DP3.1. Если T6OE = 1, то T6OUT выводит состояние T6OTL. При T6OE = 0, вывод T6OUT может использоваться для ввода-вывода основного назначения.

Сигналы о переполнении и опустошении таймера Т6 могут быть использованы в качестве тактовых импульсов для таймеров САРСОМ блока. При работе в этом режиме обеспечивается прямое внутреннее соединение между таймером Т6 и САРСОМ таймерами.

Режим таймера для Т6 выбирается путем установки 000_В в битовом поле Т6М регистра Т6CON. В этом режиме Т6 изменяет содержимое, в соответствии с сигналом от системного тактового генератора (CPU clock). При этом частота работы таймера определяется при помощи программируемого делителя. Коэффициенты программируемого делителя выбираются в соответствии с значениями битового поля Т3І. Входная частота f_{T6} таймера Т6 и длительность такта r_{T6} линейно зависят от частоты ЦПУ.

$$r_{T3}(\text{MKC}) = \frac{4 \cdot 2^{<T6I>}}{f_{\text{CPU}}(\text{MGu})}$$



Блок-схема Т6 в режиме таймера

В таблице приведены ниже возможные значения частот, длительности такта и периода при использовании частоты ЦПУ 20МГц. Эту таблицу можно применить к режиму Gated таймера и для вспомогательного таймера T5 в режиме таймера и режиме gated таймера. Заметим, что в некоторых режимах значение имеют только три значащих цифры.

$f_{CPU} = 20\text{МГц}$	Выбор входа таймера T2I/T3I/T4I							
	000 _B	001 _B	010 _B	011 _B	100 _B	101 _B	110 _B	111 _B
Коэффициент деления	4	8	16	32	64	128	256	512
Входная частота	5 МГц	2.5 МГц	1.25 МГц	625 кГц	312.5 кГц	156.25 кГц	78.125 кГц	39.06 кГц
Квантование	200 нс	400 нс	800 нс	1.6 мкс	3.2 мкс	6.4 мкс	12.8 мкс	25.6 мкс
Период	13 мс	26 мс	52.5 мс	105 мс	210 мс	420 мс	840 мс	1.68 с

Таймер 3 в режиме gated таймера

Режим gated таймера для T6 выбирается путем установки «010» или «011» в поле T6M регистра T6CON. Значение бита T6M.0 (T6CON.3) определяет активный уровень напряжения на входе gate. В режиме gated таймера изменение входной частоты можно производить аналогично изменению входной частоты в режиме таймера. В режиме gated таймера отключение тактового входного сигнала производится посредством подачи сигнала на вход T6IN, являющегося альтернативной функцией вывода порта P5.12.

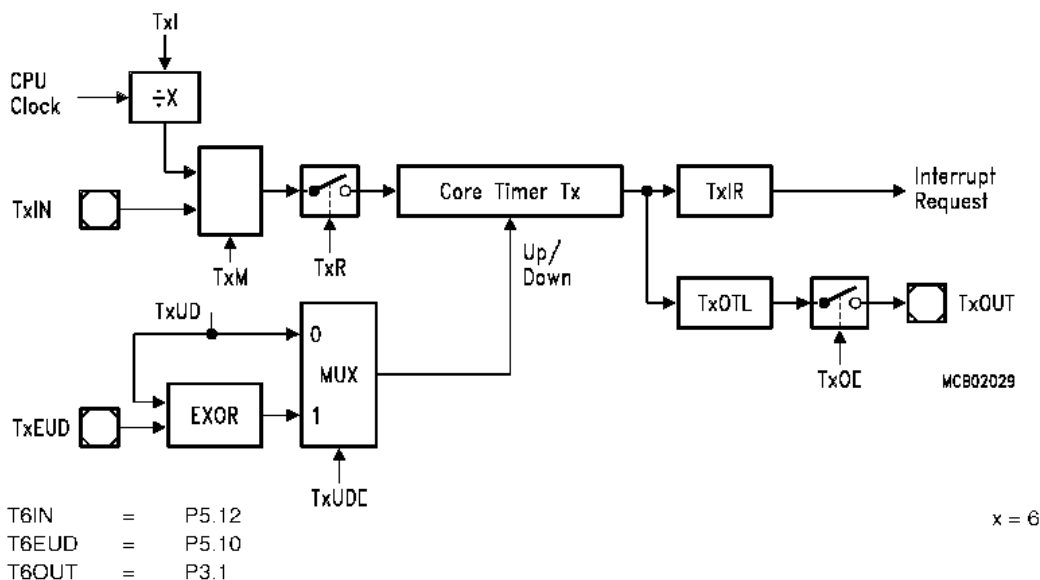


Рисунок 9-14

Блок-схема T3 в режиме gated таймера

Если T6M.0=0, таймер будет работать в случае подачи низкого уровня напряжения на вход T6IN. При подаче высокого уровня напряжения счет будет остановлен. Если T6M.0=1, таймер будет работать при подаче высокого уровня напряжения на вход T6IN. Путем изменения значения бита

Т6R таймер можно включать или отключать. Таймер находится в рабочем режиме только в том случае, если выполнены сразу оба вышеперечисленных условия (Т6R=1, Т6IN на активном уровне напряжения).

Примечание: Подача gate сигнала на вход Т6IN не приводит к подаче запроса на прерывание.

Т6 в режиме счетчика

Режим счетчика для таймера Т6 выбирается путем установки 001 в битовом поле Т6М регистра Т6CON. В этом режиме отсчеты производятся по фронту сигнала на входе Т6IN, являющегося альтернативной функцией P5.12. Фронт сигнала, приводящего к увеличению или уменьшению значения счетчика, может быть как положительный, так и отрицательный, а также оба фронта могут производить изменения значения счетчика. Тип фронта переключения выбирается в битовом поле Т6I управляющего регистра Т6CON (в таблице).

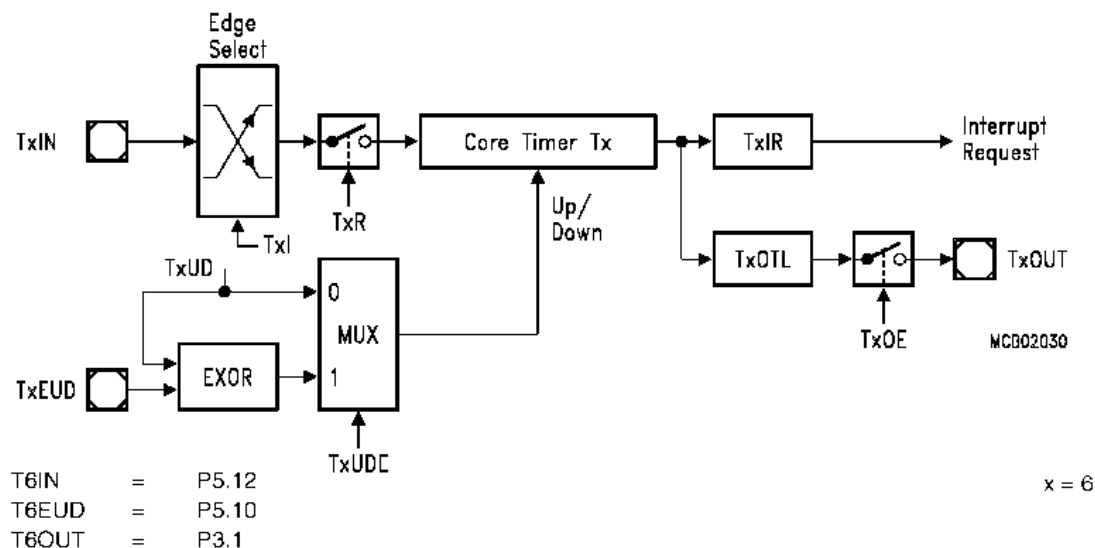


Рисунок 9-15

Блок-схема Т6 в режиме счетчика

Выбор фронта срабатывания Т6 в режиме счетчика

T6I	Фронт срабатывания для увеличения/уменьшения значения счетчика
0 0 0	Нет, счетчик отключен
0 0 1	Положительный фронт на входе Т6IN
0 1 0	Отрицательный фронт на входе Т6IN
0 1 1	Оба фронта на входе Т6IN
1 x x	Зарезервирован. Не использовать.

Максимальная доступная в режиме счетчика входная частота составляет $f_{CPU}/8$ (2.5 МГц при частоте ЦПУ 20МГц). Для корректной отработки фронта входного сигнала счетчика необходимо в течении как минимум 4 тактов ЦПУ сохранять без изменение значение сигнала на входе.

Вспомогательный таймер T5 блока GPT2

T6 может работать в режимах таймера, gated таймера или в режиме счетчика. Также вспомогательный таймер можно объединить с T6.

Примечание: У вспомогательного таймера отсутствуют собственный альтернативный вывод порта и выходной триггер.

Индивидуальные настройки таймера T5 производятся в побитно адресуемом регистре управления T5CON. Одинаковые функции для T5 и T6 управляются битами, размещенными на одинаковых позициях в регистрах управления.

SFR
T5CON (FF46_H/A3_H) Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T5SC	T5 CLR	CI	-	-	-	T5 UDE	T5UD	T5R	-	T5M					
rw	rw	rw	-	-	-	rw	rw	rw	-	rw					

Бит	Функция
T5M	Управление режимом работы таймера 5 0 0: Режим таймера 0 1: Режим счетчика 1 0: Gated таймер с низким активным уровнем напряжения 1 1: Gated таймер с высоким активным уровнем напряжения
CI	Выбор входа регистра CAPREL 0 0: Захват выключен 0 1: По положительному фронту на входе CAPIN 1 0: По отрицательному фронту на входе CAPIN 1 1: По обоим фронтам
T5SC	Разрешение режима захвата значения таймера 5 T5SC = 0: Запрещен T5SC = 1: Разрешен

Описание остальных битов аналогично таймеру T6.

Таймер T5 в режиме счетчика

Режим счетчика в вспомогательном таймере T5 выбирается при помощи установки 001 в битовом поле T5M регистра T5CON. В режиме счетчика изменение значения таймера производится либо по фронту сигнала на входе T5IN, либо при переключении триггера T6OTL.

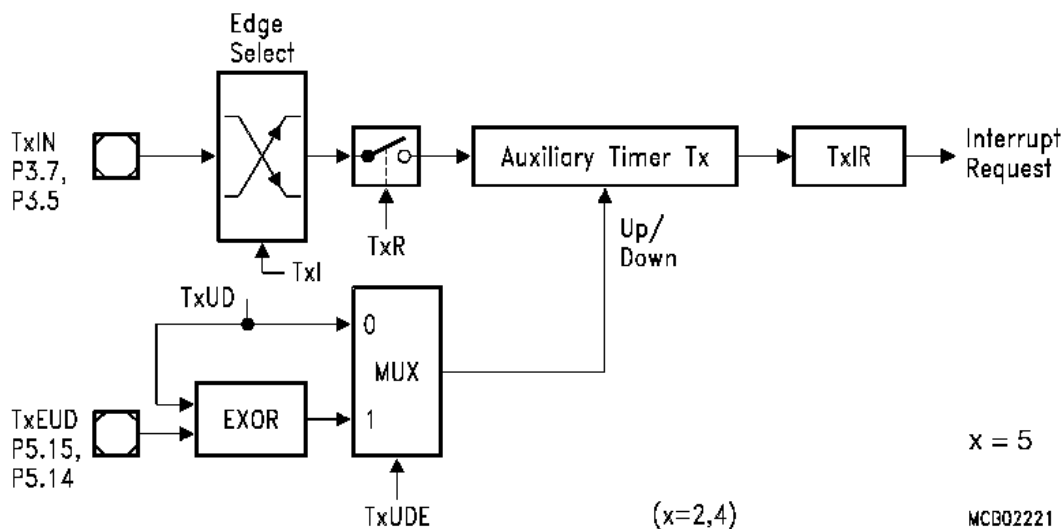


Рисунок 9-16

Блок-схема вспомогательного таймера T5 в режиме счетчика

Изменение значения таймера может производиться как по положительному так и по отрицательному фронту, а также возможен режим переключения одновременно по обоим фронтам, или по изменению значения T6OTL. Необходимый режим можно установить в битовом поле T5I регистра T5CON:

T5I	Фронт срабатывания для увеличения/уменьшения значения счетчика
0 0 0	Нет, счетчик отключен
0 0 1	Положительный фронт на входе T5IN
0 1 0	Отрицательный фронт на входе T5IN
0 1 1	Оба фронта на входе T5IN
1 0 1	Положительное переключение триггера T6OTL
1 1 0	Отрицательное переключение триггера 65OTL
1 1 1	Любое переключение триггера T6OTL

Примечание: Переключение счетчиков T5 будет иметь место только в случае переполнения или опустошения таймера T6. Программное изменение значения T6OTL не будет влиять на изменение значения таймера T5.

Объединение таймеров

Благодаря возможности использования бита T6OTL в качестве входа вспомогательного таймера в режиме счетчика, имеется возможность объединения таймера T6 с вспомогательным таймером. В зависимости от выбранного направления переключения бита T6OTL для изменения значения вспомогательного таймера, можно сформировать 32-разрядный или 33-разрядный таймер/счетчик.

- **32-разрядный таймер/счетчик:** При одновременном использовании как положительного так и отрицательного изменения значения бита T6OTL в качестве тактового сигнала вспомогательного таймера, создается режим 32-разрядного счетчика/таймера. Таким образом два таймера образуют один 32-битный таймер.

- **33-разрядный таймер/счетчик:** При использовании либо положительного, либо отрицательного изменения значения бита T6OTL, переключение объединенного таймера будет происходить только при каждом втором переполнении или опустошении T6. Таким образом образуется 33-разрядный таймер (16-разрядный T6 + T6OTL + 16-разрядный вспомогательный таймер).

Направление счета в двух объединенных таймерах может различаться.

В режиме объединенных таймеров T6 может работать в режиме таймера, gated таймера или режиме счетчика.

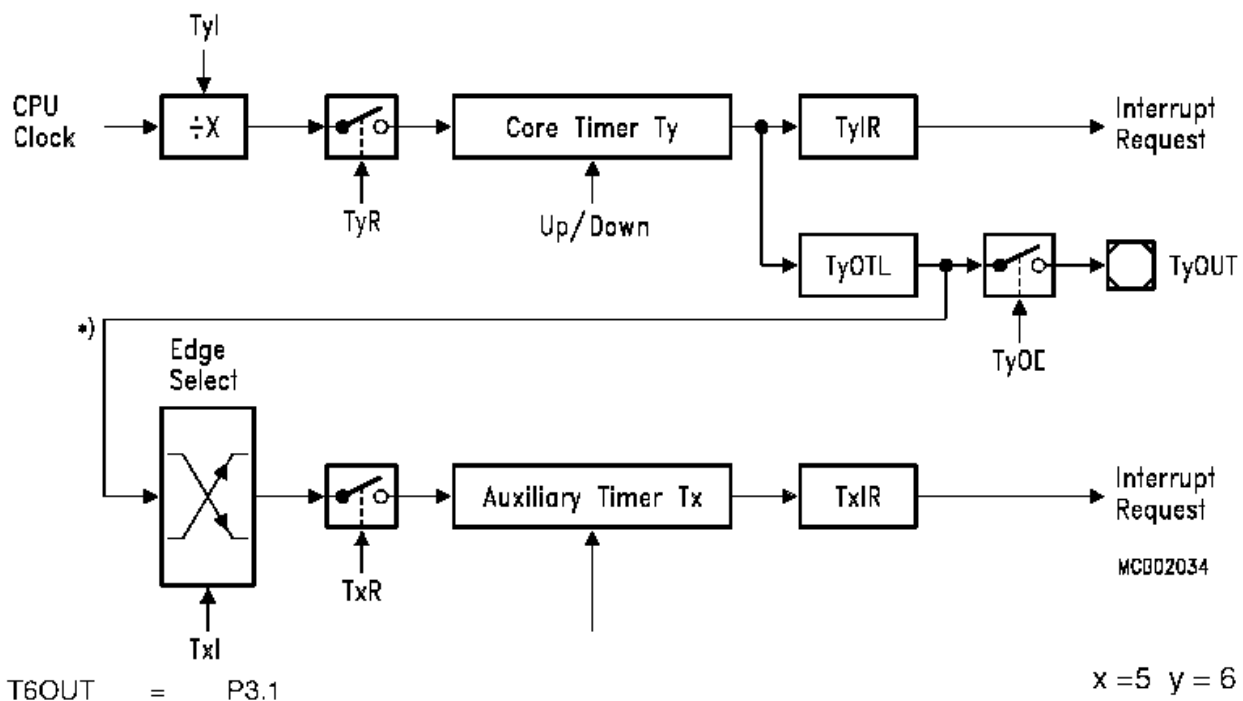


Рисунок 9-17

Объединение T6 и вспомогательного таймера

Регистр захвата и перезагрузки CAPREL в режиме захвата

Этот 16-разрядный регистр может использоваться в качестве регистра захвата значения вспомогательного таймера T5. Работа в этом режиме выбирается путем установки «1» в бите T5SC регистра T5CON. Источником сигнала захвата является сигнал на входе CAPIN. В качестве входа CAPIN используется альтернативная функция порта P3.2. Управляющий фронт может быть как положительный так и отрицательный либо оба фронта одновременно. Для установки типа фронта используются два младших бита поля T5I. Значения выбираются аналогично описанному выше режиму.

По приходу необходимого фронта сигнала на входе CAPIN содержимое вспомогательного таймера T5 сохраняется в регистре CAPREL. При этом устанавливается флаг запроса на прерывание CRIR. После этого возможно сбросить значение в таймере T5. Эта функция управляется битом T5CLR регистра T5CON. В случае T5CLR = 1, после сохранения значения таймера T5 в регистре CAPREL содержимое таймера будет сброшено.

Примечание: Установка «1» в бите T5SC имеет влияние только на необходимость совершения захвата. Если T5SC = 0, то сигнал на входе CAPIN используется для очистки таймера T5 и для подачи запроса на прерывание CRIC.

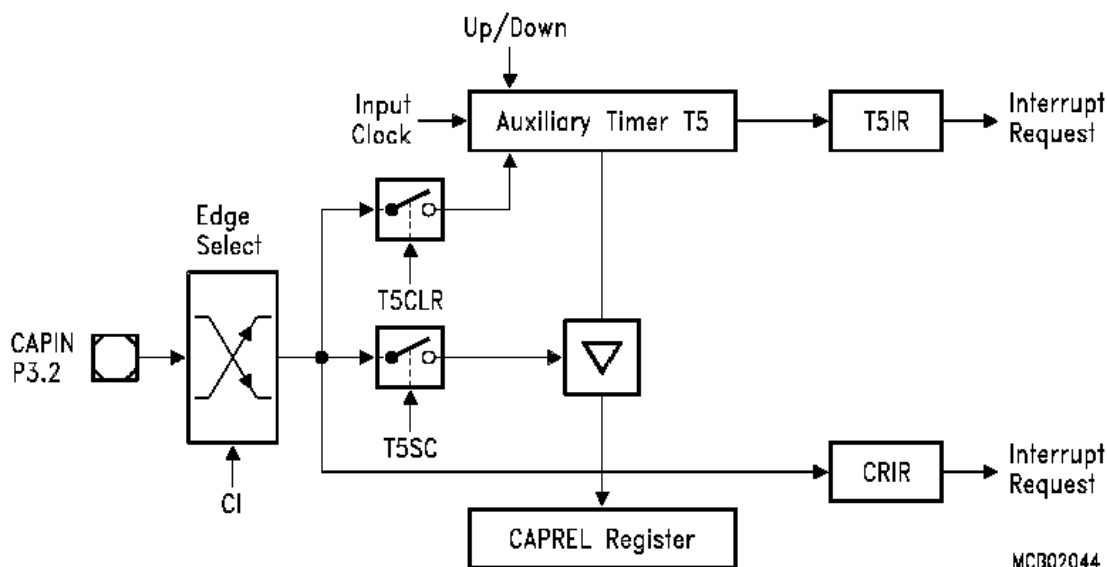


Рисунок 9-18

Регистр CAPREL в режиме захвата

Регистр CAPREL в режиме перезагрузки

Возможно использование этого регистра в режиме перезагрузки содержимого регистра T6. Этот режим можно выбрать путем установки «1» в бите T6SR регистра T6CON. Для перезагрузки значения таймера необходимо совершение переполнения или опустошения таймера T6.

При переходе значения таймера из FFFF в 0000 или при переходе из 0000 в FFFF (при счете в отрицательном направлении) значение регистра CAPREL загружается в таймер T6. При этом не устанавливается флаг запроса на прерывание CRIR. Однако устанавливается флаг запроса на прерывания T6IR, который показывает переполнения и опустошения T6.

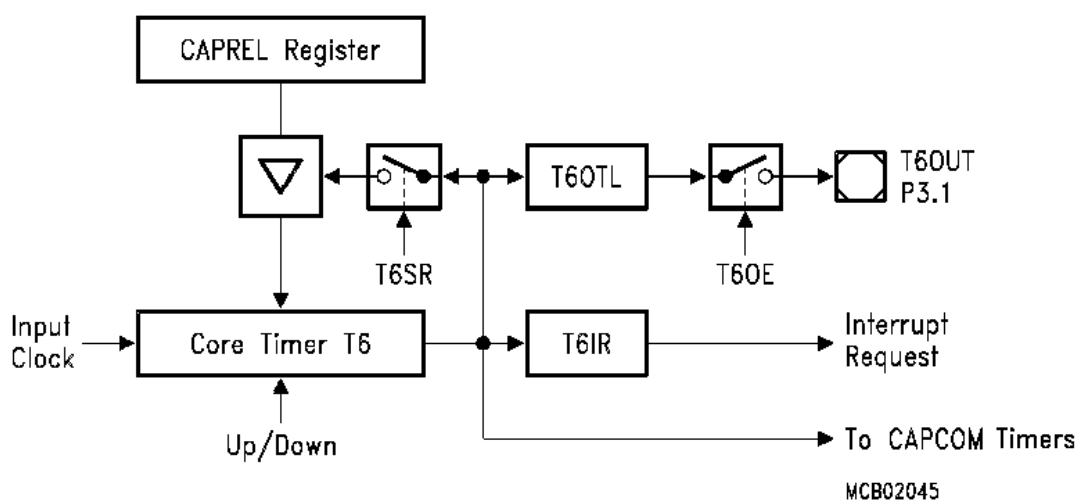


Рисунок 9-19

Регистр CFPREL в режиме перезагрузки

Регистр CAPREL в режиме захвата-перезагрузки

Структура GPT2 устроена таким образом, что имеется возможность одновременно включать функцию перезагрузки и захвата для регистра CAPREL в битах T5SC и T6SR. Данная особенность позволяет производить умножение частоты.

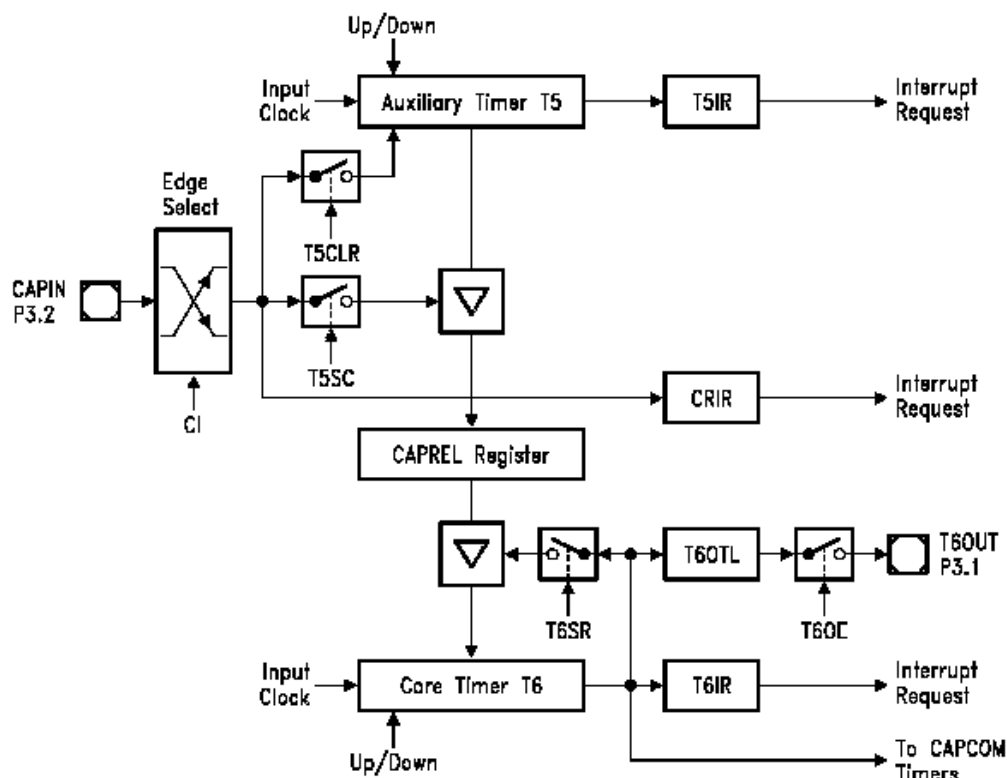


Рисунок 9-20

Регистр CAPREL в режиме захвата-перезагрузки

Работа в этом режиме позволяет умножать частоту сигнала на входе CAPIN.

Пусть частота T5 составляет $1/32$ от частоты ЦПУ. По приходу сигнала на вход CAPIN значение таймера T5 записывается в регистр CAPREL, и после этого производится сброс значения таймера T5. Таким образом в регистре CAPIN всегда содержится значение временного интервала между двумя внешними сигналами. Таймер T6 настроен на работу на частоте в 4 раза меньше частоты ЦПУ, т.е. в 8 раз быстрее T5. T6 работает в режиме отрицательного счета, поэтому при каждом достижении нуля происходит загрузка значения из регистра CAPREL. Таким образом частота T6 в 8 раз больше частоты на входе CAPIN таймера T5. В качестве выходного сигнала может использоваться T6OTL, и его значение может быть выведено на T6OUT.

Управление прерываниями от таймеров GPT2 и CAPREL

При переходе значений таймера из FFFF в 0000 (положительное направление счета) и при переходе из 0000 в FFFF (отрицательное направление счета) устанавливаются флаги запроса на прерывание T5IR или T6IR. При обнаружении фронта сигнала на входе CAPIN (нужной полярности), устанавливается флаг запроса CRIR. Для всех этих запросов будет разрешено прерывание или PEC-обслуживание, если в соответствующих битах разрешения прерывания будет установлена 1.

Регистры имеют общую структуру для всех xxIC регистров.

T5IC (FF66/B3_H)

T6IC (FF68/B4_H)

CRIC(FF6A/B5_H)

10 Асинхронный/Синхронный последовательный интерфейс

Асинхронный/Синхронный последовательный интерфейс (ASC0) обеспечивает передачу данных между микроконтроллером C167 и другими микроконтроллерами, микропроцессорами и периферией.

ASC0 поддерживает полнодуплексную асинхронную связь с пропускной способностью до 625Кбод, и полудуплексную синхронную связь с пропускной способностью до 2.5Мбод (при частоте ЦПУ 20МГц). В синхронном режиме прием и передача данных осуществляется синхронно по генерируемому C167 сигналу. В асинхронном режиме, может осуществляться передача 8- или 9-разрядных данных, создание бита проверки четности и создание различного количества стоповых бит. Возможность использования Parity, framing, и определение ошибок overrun увеличивает достоверность передаваемых данных. При передаче и приеме данных осуществляется двойная буферизация. Для многопроцессорных связанных систем добавлен механизм распознавания адреса в байтах данных. Тестирование осуществляется при помощи опции loop-back. 13-разрядный генератор bad rate создает независимый тактовый сигнал для блока ASC0.

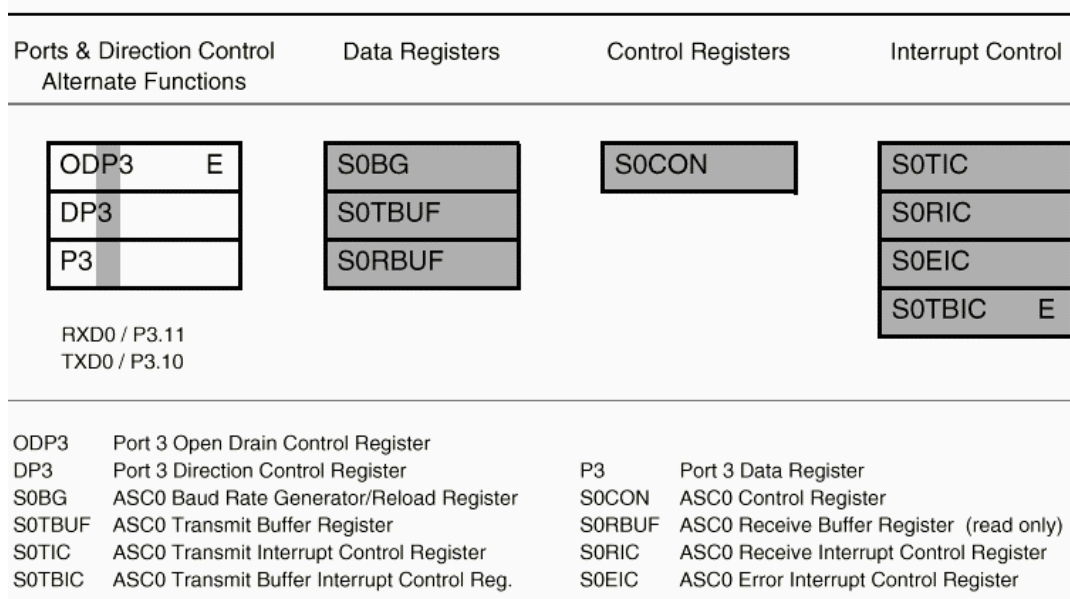


Рисунок 10-1

SFR-регистры и выходы портов, связанные с ASC0

Для управления режимом работы последовательного порта ASC0 предназначен побитно адресуемый регистр S0CON. Этот регистр содержит биты управления режимом и способов проверки ошибок, а также флаги состояния и флаги обнаружения ошибок.

S0CON (FFB0_H/D8_H)

SFR

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S0R	S0LB	S0BRS	S0ODD	-	S0OE	S0FE	S0PE	S0OEN	S0FEN	S0PEN	S0REN	S0STP	S0M		
rw	rw	rw	rw	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

Бит	Функция
S0M	Управление режимом работы ASC0 0 0 0: 8-разрядные данные, синхронный режим 0 0 1: 8-разрядные данные, асинхронный режим 0 1 0: Зарезервировано! Не использовать 0 1 1: 7-разрядные данные + parity, асинхронный режим 1 0 0: 9-разрядные данные, асинхронный режим 1 0 1: 8-разрядные данные + бит пробуждения, асинхронный режим 1 1 0: Зарезервировано! Не использовать 1 1 1: 8-разрядные данные + parity, асинхронный режим
S0STP	Выбор количества стоповых бит 0: Один стоповый бит 1: Два стоповых бита
S0REN	Бит разрешения передачи 0: Передача запрещена 1: Передача разрешена (предустанавливается аппаратно, после выбора синхронного режима)
S0PEN	Бит включения проверки четности 0: Игнорировать parity 1: Проверять parity
S0FEN	Бит разрешения проверки framing 0: Игнорировать ошибки framing 1: Проверять ошибки framing
S0OEN	Бит разрешения проверки overrun 0: Игнорировать ошибку overrun 1: Проверять ошибку overrun
S0PE	Флаг ошибки Parity Устанавливается в случае ошибки четности, в том случае если S0PEN = 1. Очистка бита должна производиться программно
S0FE	Флаг ошибки Framing Устанавливается в случае ошибки parity, если S0FEN = 1. Очистка бита должна производиться программно

Бит	Функция
S0OE	Флаг ошибки Overrun Устанавливается в случае ошибки parity, если S0OEN = 1. Очистка бита должна производиться программно
S0ODD	Бит выбора четности 0: по четным (parity будет установлена на четное количество «1» в данных) 1: по нечетным (parity будет установлена на нечетное количество «1» в данных)
S0BRS	Бит выбора скорости передачи (baudrate) 0: Деление частоты генератора на запрограммированное значение + постоянная величина (зависящая от режима) 1: Дополнительное уменьшение частоты тактового генератора на 2/3 rd
S0LB	Бит разрешения режима LoopBack 0: Режим стандартной передачи/получения данных 1: Включение режима LoopBack
S0R	Бит работы генератора baudrate 0: Генератор baudrate отключен (ASC0 деактивирован) 1: Генератор baudrate включен

Передача данных начинается с записи данных в буферный регистр SOTBUF либо по команде, либо по PEC-передаче. Будет передано только то количество бит данных в буфер, которое определено для данного режима работы. Т.е. биты записанные с 9-ой по 15-ую позицию регистра SOTBUF, никогда не передаются. После завершения передачи, значение буферного регистра передачи будет очищено.

Передаваемые данные проходят через двойную буферизацию, поэтому можно записывать новые данные в буфер до окончания передачи предыдущих данных. Это позволяет создавать бесперебойную передачу данных.

Прием данных начинается с установки «1» в бите S0REN. После завершения приема данных, полученные данные и бит четности (если выбран режим с проверкой четности) можно быть прочтен в буферном регистре получения S0RBUF. Биты старшей половины регистра не имеют значения в данном режиме, и поэтому при чтении этих регистров возвращаются нулевые значения.

При получении данных используется двойная буферизация, поэтому можно начать получение следующих данных, до того как будут прочитаны предыдущие данные из буфера. Во всех режимах с помощью установки «1» в бите S0OEN можно использовать режим обнаружения ошибки overrun

буфера получения. При работе в этом режиме флаг состояния S0OE и флаг запроса на прерывание по ошибке S0EIR будут установлены в «1», в том случае, если буферный регистр получения не был прочтен до завершения получения следующих данных. При этом производится запись новых данных в буфер поверх старых.

Режим Loop-Back используется для параллельного получения передаваемых данных в буфер получения. Данная возможность находит свое применение для тестирования подпрограммы последовательной передачи данных на ранних стадиях разработки без необходимости создания внешней сети. В режиме loop-back необходимо использовать альтернативные функции выводов порта 3.

Примечание: последовательная передача данных возможна только в том случае, если установлена «1» в бите работы генератора Baud Rate (S0R). В ином случае последовательный интерфейс не используется.

10.1 Работа в асинхронном режиме

В асинхронном режиме поддерживается полнодуплексная связь, то есть передатчик и приемник данных используют один и тот же формат frame и одинаковую скорость передачи. Данные передаются через вывод TXD0/P3.10.

RXD0/P3.11 служит для получения данных.

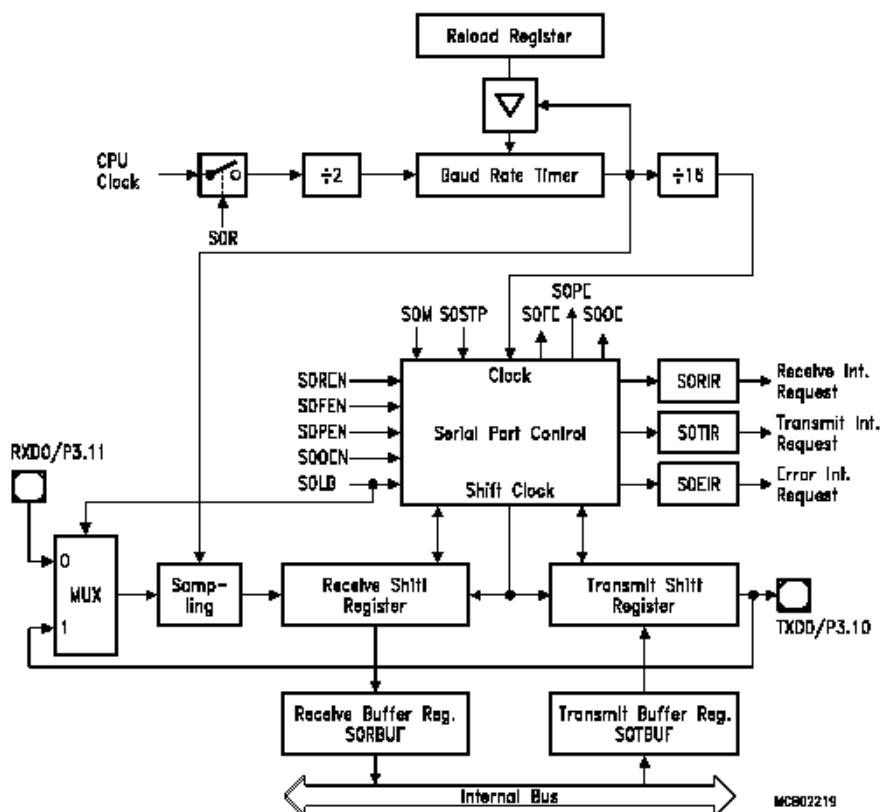


Рисунок 10-2

Последовательный порт ASC0 в асинхронном режиме

Формат данных в асинхронном режиме

Формат 8-разрядных данных состоит либо из 8 бит данных ($S0M = 001$), либо из 7 бит данных и автоматически создаваемого бита контроля четности ($S0M = 011$). Контроль четности может быть как за четным так и за нечетным количеством, в зависимости от значения бита $S0ODD$ регистра $S0CON$. В случае контроля четности за четным количеством, результатом операции modulo-2-sum по семи значащим битам является значение бита «1». Если ведется контроль за нечетным количеством, то в этом случае значение бита будет «0». Режим контроля четности включается при установке бита $S0PEN$ (не используется при 8-разрядном формате данных). В случае получения неправильного контрольного бита устанавливаются флаг ошибки контроля четности $S0PE$ и флаг запроса на прерывание по ошибке. Значение бита контроля четности сохраняется в $S0RBUF.7$.

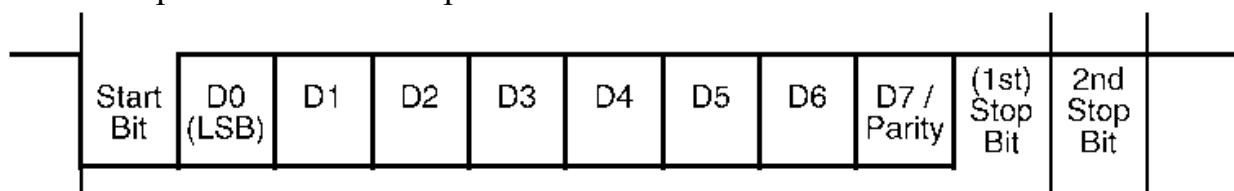


Рисунок 10-3

Формат данных в асинхронном 8-битном режиме

9-разрядный формат данных включает в себя либо 9 битов данных $D8...D0$ ($S0M=100$), либо 8 битов данных $D7...D0$ и автоматически генерируемый бит контроля четности ($S0M=111$), либо 8 битов данных $D7...D0$ и бит пробуждения ($S0M=101$). Контроль четности может быть как за четным так и за нечетным количеством.

В режиме пробуждения, получаемые данные передаются в буфер получения только в том случае, если 9-ый бит (бит пробуждения) содержит «1», в ином случае флаг запроса на прерывание по получению данных не устанавливается, и таким образом не происходит передача данных.

Данный режим может использоваться в многопроцессорных системах:

Когда ведущий микроконтроллер хочет передать блок данных, то первым делом посылается байт адреса, определяющий подчиненный микроконтроллер. Байт адреса отличается от байта данных, тем что дополнительный 9-ый бит для адреса устанавливается в «1», а для данных устанавливается в «0». Поэтому при передаче данных не прерывается работа ни одного из подчиненных микроконтроллеров, работающих в режиме 8-разрядных данных + бит пробуждения. При передаче байта адреса будет прервана работа всех микроконтроллеров, после этого каждый из подчиненных микроконтроллеров проверит значение 8-разрядного адреса. Тот микроконтроллер, для которого предназначается передача, переключится

в 9-битный режим данных (путем очистки бита S0M.0). Подчиненный микроконтроллер после этого получит байт данных, а все остальные останутся в режиме 8 бит данных + бит пробуждения, и таким образом проигнорируют все полученные данные.

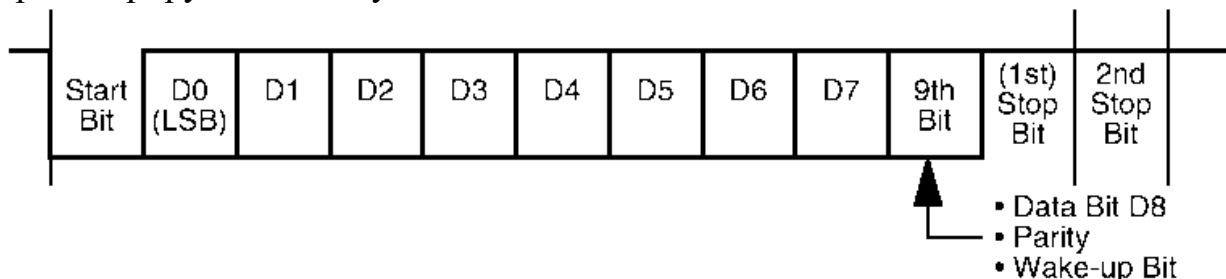


Рисунок 10-4

Формат данных в асинхронном 8-разрядном режиме

Асинхронная передача начинается при переполнении divide-by-16 счетчика (выше на рисунке), при этом устанавливается в «1» бит S0R. После этого данные загружаются в S0TBUF. Формат передаваемых данных содержит следующие элементы:

- Стартовый бит
- Поле данных (8 или 9 бит, а также первый LSB, включая, в случае необходимости, бит контроля четности)
- Ограничитель (1 или 2 стоповых бита)

Передача данных имеет двойную буферизацию. Когда передатчик находится в незанятом режиме, передаваемые данные, загруженные в S0TBUF, немедленно передаются в промежуточный регистр, таким образом освобождая S0TBUF для следующих передаваемых данных. Этот процесс проявляется себя путем установки флага запроса на прерывание S0TBIR по передаче данных в буфер. После этого одновременно с передачей данных может производиться загрузка следующих данных в буфер S0TBUF.

Флаг запроса на прерывание по завершению передачи данных (S0TIR) устанавливается, перед тем как будет передан последний бит пакета, иными словами перед первым или вторым стоповым битом из промежуточного регистра будут стерты данные.

Необходимо настроить вывод передачи данных TXD0/P3.10 на вывод значения альтернативной функции, т.е. P3.10 = 1 и DP3.10 = 1.

Асинхронная передача данных начинается при поступлении отрицательного фронта на вход RXD0, при условии, что $S0R = 1$ и $S0REN = 1$. Измерение значения на входе получения данных RXD0 производится 16 раз за один такт генератора baudrate. Значения битов определяются на 7-ом – 9-ом измерении значения на входе. Такая схема позволяет добиваться безошибочных результатов.

При обнаружении нуля в стартовом бите, схема получения данных сбрасывает все значения и ожидает следующего отрицательного фронта на входе RXD0. При правильном значении стартового бита, схема получения данных продолжает свою работу и записывает данные в промежуточный регистр получения.

После получения последнего стопового бита, содержимое промежуточного регистра передается в буфер данных S0RBUF. Параллельно с 9-ым измерением в последнем стоповом бите вне зависимости от правильности значения последнего стопового бита устанавливается флаг запроса на прерывание по получению данных S0RIR. После этого схема получения данных переходит в режим ожидания следующего стартового бита (отрицательного фронта).

Необходимо настроить вход получения данных на ввод $DP3.11=0$.

Асинхронный прием данных прекращается после очистки значения бита S0REN. В случае необходимости после этого завершается текущий прием данных, включая генерацию флага запроса на прерывание и запроса на прерывание по ошибке.

Примечание: В режиме пробуждения получаемые данные передаются только в буфер получения, если бит пробуждения установлен в «1». Если в этом бите содержится «0», дальнейшие действия не производятся.

При работе в синхронном режиме поддерживается полудуплексная связь, основанная на одном входе при помощи регистров перемещения. Данные передаются и получаются через вывод RXD0/P3.11, при этом тактовый сигнал выводится через TXD0/P3.10. Для работы в синхронном режиме необходимо установить 000 в битовом поле S0M.

The diagram illustrates the internal architecture of the Serial Port I/O Controller (SIOC). At the top, the **Reload Register** feeds into a divider (∇), which in turn controls the **Baud Rate Timer**. The **Baud Rate Timer** is clocked by the **CPU Clock** through a switch controlled by **SQR** and a divider ($\div 2$). The timer's output is divided by 4 ($\div 4$) to provide a **Shift Clock** to the **Serial Port Control** block. The **Serial Port Control** block also receives **SQM=000B** and **SQOC** signals. It manages the **Receive Shift Register** and **Transmit Shift Register**, which are connected to the **Receive Buffer Reg. SORBUF** and **Transmit Buffer Reg. SOTBUF** respectively. These buffers interface with the **Internal Bus**. The **Serial Port Control** also generates interrupt requests: **SQRIR** (Receive Int. Request), **SOTIR** (Transmit Int. Request), and **SQOIR** (Error Int. Request). On the left, the **TXD0/P3.10** pin is connected to the **Serial Port Control**, and the **RXD0/P3.11** pin is connected to a **MUX** (multiplexer) that routes data between the **Receive Shift Register** and the **Transmit Shift Register**.

Синхронный режим последовательного порта ASC0

10-8

После этого параллельно с передачей данных можно загружать новое значение в буфер. Биты данных передаются по тактовым импульсам. После завершения передачи восьмого бита выставляется высокий уровень напряжения на выходах TXD0 и RXD0, также устанавливается флаг запроса на прерывание по завершению передачи S0TIR, и прекращается передача данных.

Для правильной работы в этом режиме необходимо установить P3.10=1 и DP3.10=1. Также на время передачи необходимо настроить RXD0/P3.11 на вывод (P3.11=1 и DP3.11).

Режим синхронного приема данных будет разрешен после установки «1» в бите S0REN. При S0R=1 данные, поступающие на вход RXD0, записываются в промежуточном регистре по тактовым импульсам, поступающим на выход TXD0. После получения восьмого бита содержимое промежуточного регистра получения будет передано в буферный регистр получения S0RBUF, и одновременно с этим будет установлен флаг запроса на прерывание по получению данных S0RIR, после этого значение бита S0REN очищается и прекращается последовательный прием данных.

Также необходимо тактовый генератор настроить на выход, путем установки P3.10=1 и DP3.10=1. Вывод RXD0/P3.11 должен быть настроен на альтернативную функцию ввода (DP3.11=0).

Синхронный прием данных прекращается после очистки бита S0REN. После этого будет завершен текущий прием данных, включая генерацию запросов на прерывание по завершению запроса на прерывание по ошибке передачи (в случае необходимости). Запись нового значения в буфер передачи во время приема данных не приведет к началу передачи.

Если предыдущий данные не были прочитаны из буферного регистра до завершения передачи следующего байта, будет выставлен флаг запроса на прерывание S0EIR и флаг ошибочного состояния overrun S0OE(если разрешена установка флага в бите S0OEN).

10.3 Возможности аппаратного обнаружения ошибок

Для повышения надежности последовательного обмена данными последовательный порт ASC0 может выставлять запрос на прерывание по ошибке, указывающего на наличие ошибок. При этом три флага состояния ошибок регистра S0CON указывают на тип произошедшей ошибки. Одновременно с завершением приема данных будет установлен флаг запроса на прерывание по ошибке S0EIR в том случае, если:

- при установленном бите обнаружения ошибок S0FEN, не обнаружена «1» в любом из ожидаемых стоповых битов. После этого будет установлен флаг S0FE, показывающий ошибки формата передаваемых данных.
- При включенном режиме проверки четности (S0PEN=1), в случае обнаружения неправильного результата, устанавливается флаг S0PE.
- В том случае если не был считан буфер до окончания следующего цикла получения данных, устанавливается флаг S0OE.

10.4 Создание baud rate ASC0

Последовательный порт ASC0 имеет собственный тактовый baud rate генератор с возможностью перезагрузки 13-разрядного значения, позволяющего устанавливать скорость передачи вне зависимости от GPT-таймеров.

Генератор baud rate работает частотой, составляющей половину от частоты ЦПУ (10МГц при частоте ЦПУ 20МГц). Отсчет в таймере ведется в отрицательном направлении и может начаться или остановиться после изменении значения бита S0R регистра S0CON. При каждом переходе через нуль, на последовательный порт посылается один тактовый импульс. После этого в таймер загружается значение, сохраненное в 13-разрядном регистре. Также результирующую частоту можно уменьшить, изменяя значение в бите S0BRS. При S0BRS=1, частота тактового сигнала уменьшается на 1/3. Поэтому скорость передачи ASC0 зависит от частоты ЦПУ, перезагружаемого значения, значения бита S0BRS и от режима работы (синхронный или асинхронный).

Регистр S0BG является двухфункциональным регистром. Чтение S0BG возвращает содержимое таймера (биты 15... 13 возвращают 0), в то время как запись в регистр изменяет значения регистра перезагрузки (при этом биты 15... 13 игнорируются).

Авто-перезагрузка содержимого таймера осуществляется каждый раз после записи нового значения в регистр S0BG. Однако, если запись совершается при S0R=0, значение таймера не будет перезагружено, до тех пор пока не будет разрешена работа тактового генератора baud rate.

Скорость передачи в асинхронном режиме

Для работы в асинхронном режиме, скорость передачи и необходимое значение для перезагрузки можно определить из формул:

$$B_{Asinc} = \frac{f_{CPU}}{16 \cdot (2 + \langle S0BRS \rangle) \cdot (\langle S0BRL \rangle + 1)}$$

$$S0BRS = \left(\frac{f_{CPU}}{16 \cdot (2 + \langle S0BRS \rangle) \cdot B_{Asinc}} \right) - 1$$

$\langle S0BRL \rangle$ - содержимое регистра перезагрузки

$\langle S0BRS \rangle$ - содержимое бита S0BRS

Максимальная скорость передачи данных при использовании частоты ЦПУ 20МГц может достигать в асинхронном режиме 625 Кбод. Таблица, представленная ниже, показывает различные варианты.

Скорость передачи	S0BRS = 0, f_{CPU} = 20МГц		S0BRS = 0, f_{CPU} = 20МГц	
	Разброс	Перезагруз. значение	Разброс	Перезагруз. значение
625 Кбод	0	0000	---	---
19.2 Кбод	+1.7%/-1.4%	001F _H /0020 _H	+1.7%/-1.4%	001F _H /0020 _H
9600 Бод	+0.2%/-1.4%	0040 _H /0041 _H	+1.0%/-1.4%	002A _H /002B _H
4800 Бод	+0.2%/-0.6%	0081 _H /0082 _H	+1.0%/-0.2%	0055 _H /0056 _H
2400 Бод	+0.2%/-0.2%	0103 _H /0104 _H	+0.4%/-0.2%	00AC _H /00AD _H
1200 Бод	+0.2%/-0.4%	0207 _H /0208 _H	+0.1%/-0.2%	015A _H /015B _H
600 Бод	+0.1%/-0.0%	0410 _H /0411 _H	+0.1%/-0.1%	02B5 _H /02B6 _H
75 Бод	+1.7%	1FFF _H	+0.0%/-0.0%	15B2 _H /15B3 _H

Скорость передачи в синхронном режиме

Для работы в синхронном режиме, скорость передачи и необходимое значение для перезагрузки можно определить из формул:

$$B_{Asinc} = \frac{f_{CPU}}{4 \cdot (2 + \langle S0BRS \rangle) \cdot (\langle S0BRL \rangle + 1)}$$

$$S0BRS = \left(\frac{f_{CPU}}{4 \cdot (2 + \langle S0BRS \rangle) \cdot B_{Asinc}} \right) - 1$$

$\langle S0BRL \rangle$ - содержимое регистра перезагрузки

$\langle S0BRS \rangle$ - содержимое бита S0BRS

Максимальная скорость передачи данных, при использовании частоты ЦПУ 20МГц, может достигать в синхронном режиме 2.5 Мбод.

10.5 Управление прерываниями ASC0

Для последовательного порта ASC0 имеется четыре побитно адресуемых регистра. Регистр S0TIC управляет прерыванием передачи, S0TBIC управляет прерыванием буфера передачи, S0RIC управляет прерыванием получения и S0EIC управляет прерыванием по ошибке последовательного порта ASC0. Каждый источник прерывания имеет собственный вектор прерывания.

В случае запроса на прерывание по ошибке, тип ошибки можно определить в регистре управления S0CON.

Примечание: В отличие от флага запроса на прерывание S0EIR, флаги состояния ошибки S0FE/S0PE/S0OE автоматически не сбрасываются при входе в подпрограмму обслуживания прерывания. Поэтому их необходимо программно очищать.

S0TIC (FF6C/B_{6H})

S0RIC (FF6E/B_{7H})

S0EIC (FF70/B_{8H})

S0TIC (F19C/CE_H)

Примечание: Подробности в описании регистров управления прерываниями.

Использование прерываний ASC0

В случае нормальной работы (т.е. без прерываний по ошибкам) ASC0 имеет три запроса на прерывание, предназначенных для управления обменом данными с помощью последовательного интерфейса:

- S0TBIR: устанавливается, когда данные перемещаются из S0TBUF в промежуточный регистр
- S0TIR: устанавливается перед передачей последнего бита в асинхронном режиме, либо после передачи последнего бита в синхронном режиме
- S0RIR: устанавливается, когда полученные данные перемещаются в S0RBUF

Если нет необходимости в получении данных, передатчик обслуживается при помощи двух прерываний.

Для **одиночных передач** достаточно использовать прерывание передатчика (S0TIR), которое показывает окончание передачи данных.

Для **многократных последовательных передач** необходимо загружать следующие данные до того, когда последний бит предыдущих данных будет передан. В асинхронном режиме, при этом теряется только один такт для ответа на запрос на прерывание. В синхронном режиме это невозможно.

Использование S0TBIR предоставляет возможность для загрузки новых данных в буфер до окончания передачи предыдущих данных.

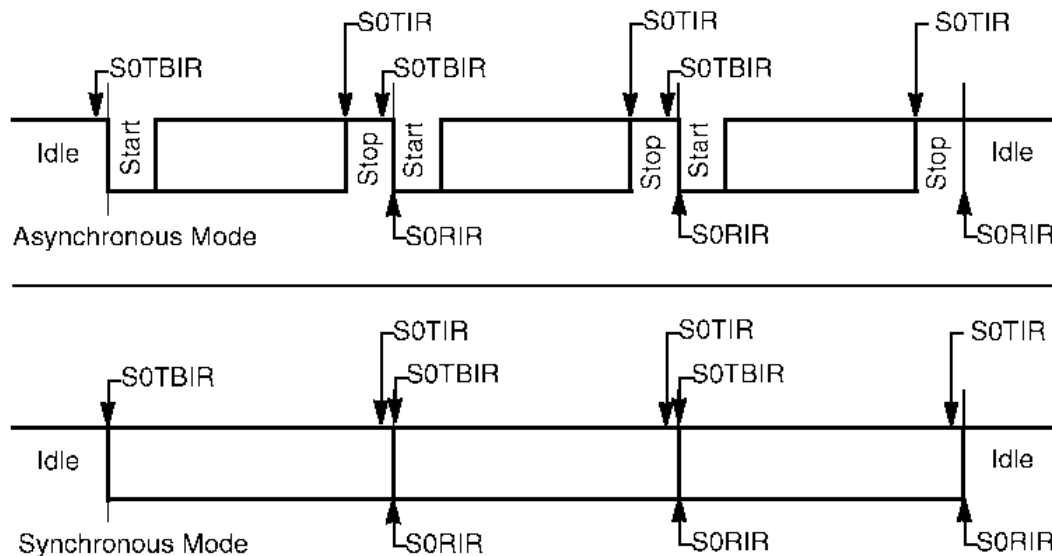


Рисунок 10-6

Создание прерываний для ASC0

11 Высокоскоростной синхронный последовательный интерфейс

Высокоскоростной последовательный интерфейс SSC предназначен для создания высокоскоростных последовательных соединений между C167 и другими микроконтроллерами или внешней периферией.

SSC поддерживает полнодуплексное и полудуплексное соединение с пропускной способностью до 5Мбод (при частоте ЦПУ 20МГц). Тактовый сигнал может создаваться как самим SSC (в режиме master), так и может поступать от внешнего master (в подчиненном режиме). Имеется возможность для программного изменения размера данных, направления передачи, полярности тактового сигнала и фазы. Эти возможности позволяют обеспечивать передачу данных с SPI-совместимыми устройствами. При приеме и передаче данных осуществляется двойная буферизация. 16-разрядный генератор baud rate SSC создает независимый тактовый сигнал.

Высокоскоростной синхронный последовательный интерфейс имеет возможности для гибкой настройки, и поэтому может использоваться передачи данных по другим синхронным последовательным интерфейсам (например ASC0 в синхронном режиме). SSC можно использовать в режимах master/slave или в многопроцессорных сетях, а также для работы с распространенным SPI-интерфейсом. Передача и прием данных производится через выводы MTSR/P3.9 (Master передача / Slave прием) и MRST/P3.8 (Master прием / Slave передача). Для вывода тактового сигнала используется SCLK/P3.13.

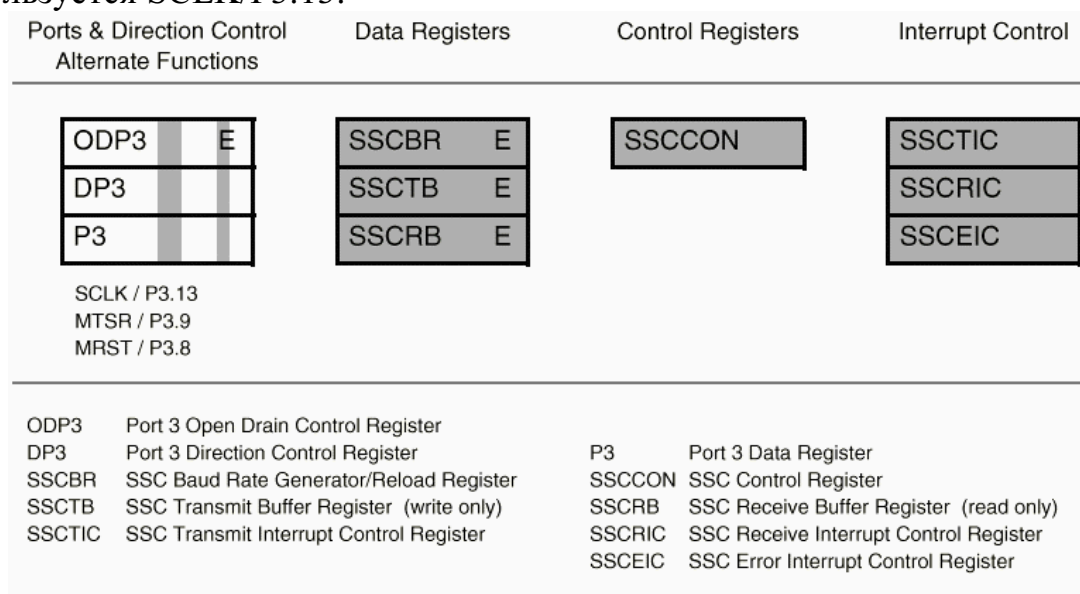


Рисунок 11-1

SFR-регистры и выводы портов для SSC

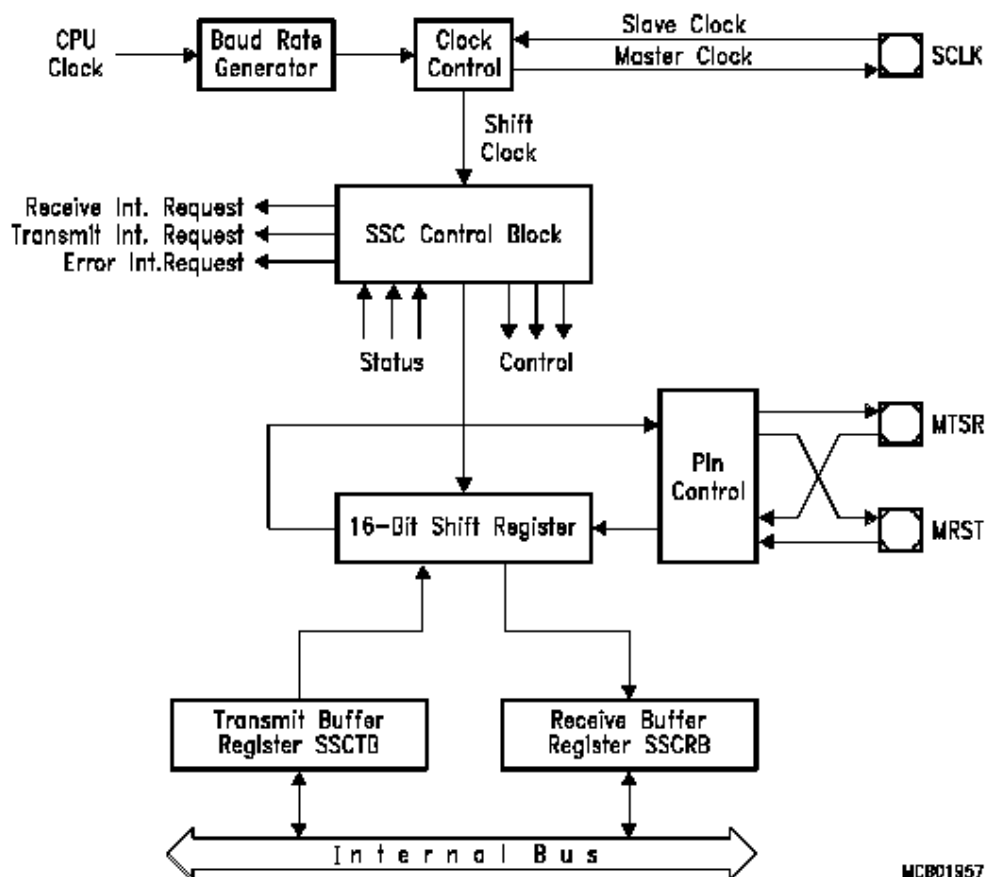


Рисунок 11-2

Блок-схема синхронного последовательного порта SSC

Режим работы SSC управляется с помощью побитно адресуемого регистра SSCCON. Этот регистр имеет две функции:

- во время программирования (SSC отключен: SSCEN = 0) предоставляется доступ к набору битов управления
- Во время работы (SSCEN = 1) предоставляется доступ к набору флагов состояния

Ниже приведено описание регистра SSCCON для обоих состояний:

SFR

SSCCON (FFB2_H/D9_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSC EN=0	SSC MS	-	SSC AREN	SSC BEN	SSC PEN	SSC REN	SSC TEN	-	SSC P0	SSC PH	SSC HB	SSCBM			
rw	rw	-	rw	rw	rw	rw	rw	-	rw	rw	rw	rw			

Бит	Функция
SSCBM	Выбор размера данных SSC 0: Зарезервировано, не использовать. 1...15: Передача данных длиной 2 ... 15 бит (<SSCBM>+1)
SSCHB	Бит управления типом заголовка SSC 0: сначала передача/прием LSB 1: сначала передача/прием MSB
SSCPH	Бит управления фазой тактового сигнала SSC 0: передача по переднему фронту, захват по заднему фронту 1: передача по заднему фронту, захват по переднему фронту
SSCP0	Бит управления полярностью тактового сигнала SSC 0: Состояние ожидания при низком уровне напряжения, передний фронт – переход из 0 в 1 1: Состояние ожидания при высоком уровне напряжения, передний фронт – переход из 1 в 0
SSCTEN	Бит разрешения определения ошибок передачи SSC 0: Игнорировать ошибки при передаче 1: Проверять ошибки при передаче
SSCREN	Бит разрешения определения ошибок при приеме SSC 0: Игнорировать ошибки при приеме 1: Проверять ошибки при приеме
SSCPEN	Бит разрешения определения ошибки фазы SSC 0: Игнорировать ошибки фазы 1: Проверять ошибки фазы
SSCBEN	Бит разрешения определения ошибки в скорости передачи SSC 0: Игнорировать ошибки в скорости передачи 1: Проверять ошибки в скорости передачи
SSCAREN	Бит разрешения автоматического перезапуска SSC 0: Не требуется дополнительных действий при ошибке в скорости передачи 1: SSC автоматически перезапускается при определении ошибки в скорости передачи

Бит	Функция
SSCMS	Бит выбора Master SSC 0: Подчиненный режим, работа с внешним тактовым сигналом на входе SCLK 1: Режим Master, создание собственного тактового сигнала и вывод его через SCLK
SSCEN	Бит разрешения работы SSC = 0 Передача и прием данных отключены. Имеется доступ к битам управления.

SFR

SSCCON (FFB2_H/D9_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSC EN=1	SSC MS	-	SSC BSY	SSC BE	SSC PE	SSC RE	SSC TE	-	-	-	-	SSCBC			
rw	rw	-	rw	rw	rw	rw	rw	-	-	-	-	r			

Бит	Функция
SSCBC	Поле отсчета битов SSC Счетчик передачи изменяет свое значение после каждого переданного бита. Не производить запись!!!
SSCTE	Флаг ошибки передачи SSC 1: Передача началась, при этом не было изменено значение в slave буфере передачи
SSCRE	Флаг ошибки приема SSC 1: Передача закончилась до того, как приемный буфер был прочитан
SSCPE	Флаг ошибки фазы SSC 1: Принимаемые данные изменяются во время считывания уровня напряжения
SSCBE	Флаг ошибки в скорости передачи SSC 1: Получаемые данные изменяются во время фронта чтения тактового сигнала
SSCBSY	Флаг занятости SSC Устанавливается во время передачи. Не производить запись!!!

Бит	Функция
SSCMS	Бит выбора Master SSC 0: Подчиненный режим. Работа с внешним тактовым сигналом SCLK 1: Режим Master. Тактовый сигнал создается в SSC и выводится через SCLK
SSCEN	Бит выбора SSC =1 Передача и прием разрешены. Доступны флаги состояния и управление M/S

При записи в SSCCON, необходимо в зарезервированные биты регистра записывать нулевые значения.

Регистр перемещения SSC с помощью логики управления подключен и к выводу передачи, и к выводу приема (см. блок-схему). Передача и прием последовательных данных может синхронизироваться и осуществляться одновременно. Передаваемые данные записываются в буфер передачи SSCTB. По мере освобождения промежуточного регистра перемещения данные перемещаются в него. SSC-master начинает передачу сразу после заполнения промежуточного регистра, а SSC-slave (SSCMS = 0) будет ожидать тактового сигнала. После начала передачи, устанавливается флаг SSCBSY и флаг запроса на прерывание по передаче (SSCTIR), указывающий на возможность записи нового значения в регистр SSCTB. После передачи запрограммированного количества битов (2... 16), содержимое промежуточного регистра перемещается в буфер приема SSCRB, и устанавливается запрос на прерывание по получению (SSCRIR). При отсутствии дальнейших передач данных (SSCTB пуст), бит SSCBSY сбрасывается. Нельзя программно изменять значение этого бита, так как этот флаг управляется аппаратно.

Настройку последовательной передачи данных можно производить в широком диапазоне:

- Данные могут содержать от 2 до 16 битов
- Передача может начинаться с LSB или MSB
- Тактовый импульс может иметь как низкий так и высокий уровень напряжения ожидания
 - Данные могут передаваться как по переднему так и по заднему фронту тактового сигнала
 - Скорость передачи может быть установлена от 152 Бод до 5Мбод (при частоте ЦПУ 20МГц)
 - Тактовый сигнал может быть как внешним (режим slave) так и внутренним (режим master)

Возможна передача данных любого размера: от 2-разрядных до 16-разрядных данных. Начало передачи с LSB (SSCHB=0) позволяет производить обмен данными с устройствами, поддерживающими ASC0 в синхронном режиме (C166, 8051). Начало передачи с MSB позволяет осуществлять обмен по SPI интерфейсу.

Вне зависимости от того выбран MSB или LSB, передаваемые данные всегда выстраиваются в регистрах SSCTB и SSCRB в правильном порядке, при этом LSB передаваемых данных размещается в бите 0. С помощью логики промежуточного регистра биты данных перераспределяются для передачи. Значения в невыбранных битах SSCTB игнорируются, а значения в невыбранных битах SSCRB не обоснованы и должны программно отбрасываться с помощью подпрограммы обслуживания передатчика.

Управление тактовыми импульсами позволяет адаптировать режим приема и передачи SSC к различным последовательным интерфейсам. Один из фронтов тактового сигнала (передний или задний) используется для передачи данных, в то время как другой используется для захвата данных. Изменяя значение бита SSCPH можно установить тип фронт (передний или задний) используемый для каждой функции. Изменяя значение бита SSCP0 можно установить уровень напряжения на выходе тактового генератора на холостом ходу.

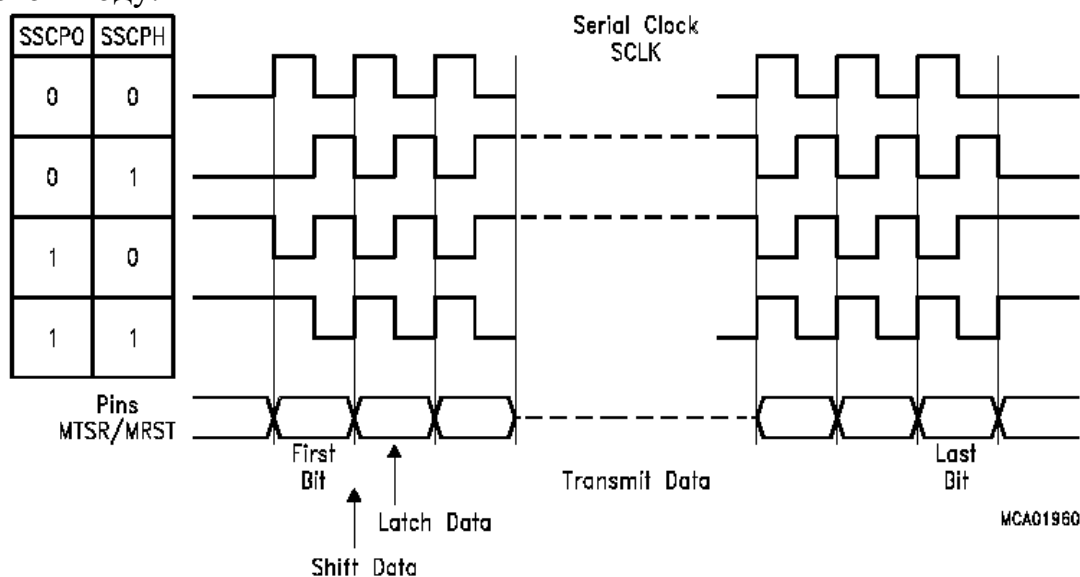


Рисунок 11-3

Полярность и фаза тактового сигнала

11.1 Полнодуплексные режимы работы

Режим работы линии всегда определяет микроконтроллер в режиме master. Линия, подключенная к выходу для вывода данных MTSR, является линией передачи, а линия приема данных подключается к входу MRST, линия тактового сигнала подключается к выводу SCLK. Тактовый сигнал может выдавать только то устройство, которое работает в режиме master. Все устройства в режиме slave используют внешний тактовый сигнал, поэтому для всех slave-устройств необходимо переключить в режим ввода данных SCLK ($DP3.13=0$). В режиме master выходной регистр передачи подключается к линии передачи, которая в свою очередь соединена с регистром приема микроконтроллера в режиме slave. Выход регистра передачи в режиме slave подключен к линии приема для того, чтобы позволить master-контроллеру получать данные. Внешние соединения являются жестко определенными, и поэтому для каждого конкретного устройства функция и направление этих выводов определяются с помощью выбора режима (master или slave).

Примечание: Направление передачи, показанное на рисунке, используется как для MSB старта, так и для LSB старта.

Во время настройке один микроконтроллер всегда устанавливается в режим master, а все остальные в режим slave.

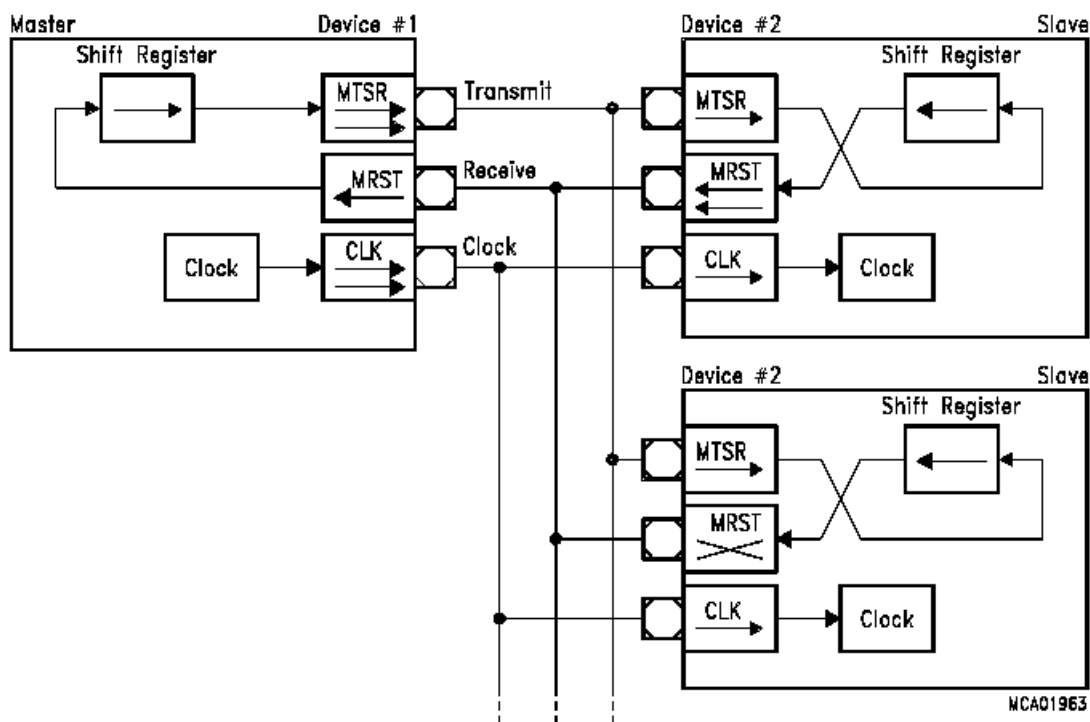


Рисунок 11-4

Конфигурация SSC в полнодуплексном режиме

Все выходы для вывода данных должны быть объединены между собой и подключены к линии приема данных. Во время передачи данных каждый slave-контроллер может передавать данные из своего регистра передачи. Существуют два пути для преодоления конфликтов на линии между двумя slave-устройствами:

Только одно slave-устройство включает драйвер передачи для MRST. Для всех других устройств MRST программируется на вход. После этого только один микроконтроллер может выставлять данные на линию приема. Master выбирает slave-устройство либо с помощью независимых линий выбора, либо с помощью специальных команд, посланных на slave-устройство. После этого выбранное slave-устройство переключает свой MRST на выход и находится в таком состоянии до тех пор, пока не получит сигнала или команды отключения.

Slave-устройства используют выходы с открытыми коллекторами на MRST. В этом случае соединение происходит по формуле Wired-AND. При этом линия получения нуждается во внешнем pullup устройстве. При этом если все не выбранные для передачи slave-устройства выставляют «1» на выходе, исключается искажение посылаемых данных. Master выбирает активный Slave либо с помощью специальной команды, либо с помощью независимых линий выбора.

При включенном последовательном интерфейсе, master-устройство может начать первую передачу данных посредством записи данных в регистр SSCTB. После этого значение из регистра записывается в регистр передачи, и на следующем такте генератора baud rate первый бит передаваемых данных будет размещен на линии MTSR. В зависимости от выбранной тактовой фазы, будет выдан тактовый импульс на линию SCLK. На обратном фронте импульса будет произведен захват данных master'ом на входе MRST. В результате имеет место полнодуплексный обмен данными. Тактовый сигнал одновременно подается на все slave-устройства, при этом master-устройство и выбранное slave-устройство произведут синхронную передачу и прием данных. После запрограммированного числа тактовых импульсов (в зависимости от длины данных) регистры передачи всех slave-устройств содержат данные, переданные master-устройством, а регистр передачи master устройства содержит данные только от выбранного slave-устройства.

После этого данные из регистра передачи копируются в буферный регистр SSCRB и одновременно с этим устанавливается флаг запроса на прерывание SSCRIR.

Как только данные из буферного регистра будут переданы в регистр передачи slave-устройства, первый бит (MSB или LSB) сразу поступает на выход MRST, не дожидаясь первого тактового импульса от master-устройства. Это необходимо по причине того, что первый фронт тактового

сигнала может использоваться для приема данных, так как возможна различная выбранная фаза.

Примечание: Прием и передача данных в SSC всегда происходят одновременно, независимо от правильности передаваемых или принимаемых данных. В этом кроется одно из отличий SSC от интерфейса ASC0.

При инициализации SCLK необходимо остерегаться ненужных тактовых фронтов от master-устройства. Внутреннее состояние альтернативной линии выхода должно оставаться в «1», в то время пока SSC отключен. Тактовый сигнал объединен по функции AND с выходным триггером соответствующей линией порта. Включение SSC в режиме, где тактовый сигнал находится в покое при низком уровне напряжения (SSCPO=0), немедленно переводит вывод альтернативных данных и вывод порта SCLK (по операции AND) на низкий уровень напряжения. Для избежания этого, используется следующая последовательность:

- Выбор уровня напряжения покоя тактового сигнала (SSCPO=x)
- Загрузка в триггер вывода порта этого же значения (P3.13=x)
- Переключение порта на вывод (DP3.13=1)
- Включение SSC (SSCEN=1)
- Если SSCPO=0, включение функции альтернативного вывода (P3.13=1)

Эта же последовательность действий может использоваться для передачи функции мастера другому микроконтроллеру в сети. В этом случае, предыдущий master и будущий master переключают свои режимы (SSCMS) и меняют направление выводов портов.

11.2 Операции в полудуплексном режиме

В полудуплексном режиме, и для приема, и для передачи данных используется только одна линия. Линия обмена данными соединяет и MTSR, и MRST каждого из устройств. Линия тактовых сигналов подключена к SCLK-выводам.

Master-устройство управляет передачей данных при помощи собственного тактового сигнала. Slave-устройство использует внешний тактовый сигнал. Последовательные данные могут передаваться между arbitrary stations, потому что все выводы для приема и передачи объединены и подключены к одной линии.

Во избежания конфликтов во время передачи данных в полудуплексном режиме используется два способа:

- Разрешение на включение передачи дается только одному передающему устройству
- Использование режима с открытым коллектором для всех не передающих устройств

Так как входы и выходы данных соединены вместе, то передаваемые устройством данные, подаются на его собственный вход (MRST в режиме master, MTSR в режиме Slave). Поэтому имеется возможность для определения искажения данных на общей линии передачи.

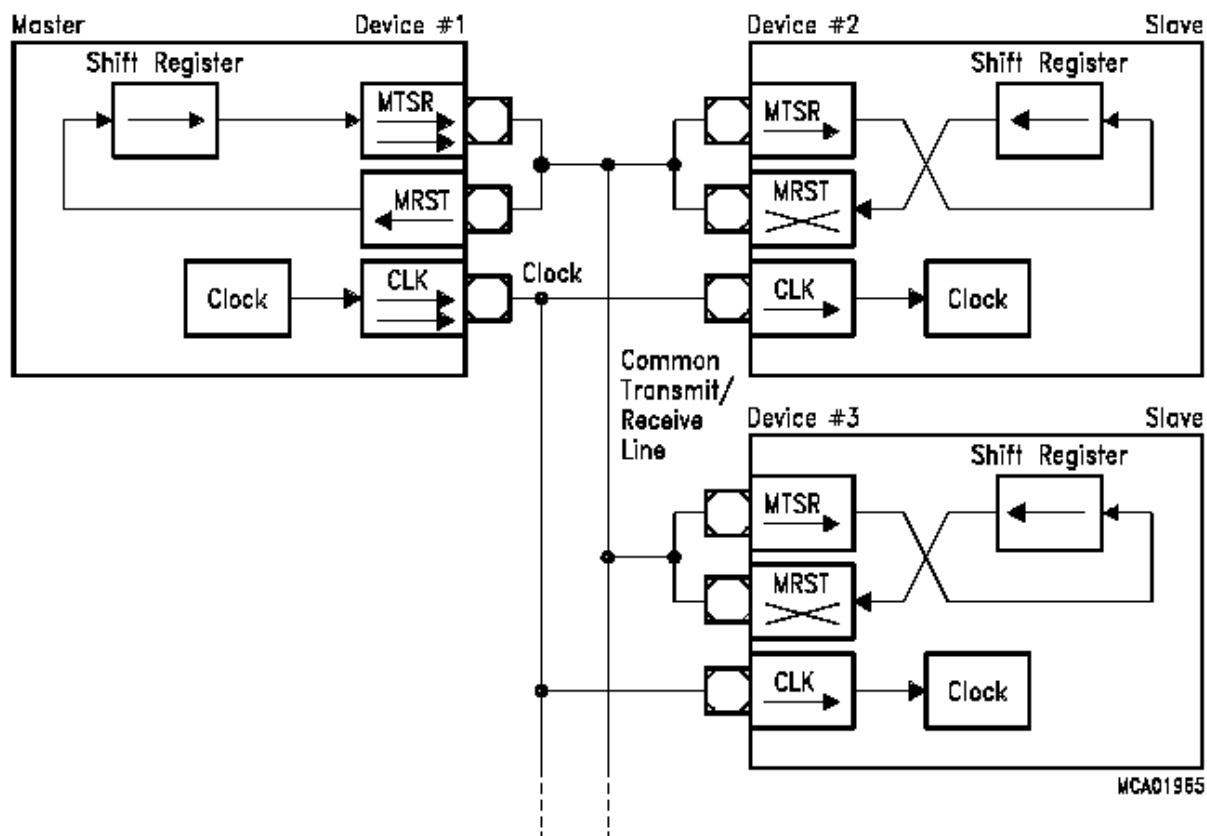


Рисунок 11-5

Блок-схема в полудуплексном режиме

Неограниченные передачи

Установка флага запроса на прерывание по передаче указывает на то, что буфер передачи SSCTB опустел и готов для приема следующих передаваемых данных. В том случае, когда загрузка данных была произведена до окончания текущей передачи, сразу после завершения передачи в промежуточный регистр передачи загружаются новые данные, и после этого начинается следующая передача без дополнительной задержки. В этом случае на линии отсутствует пауза между двумя пакетами данных. Этот режим используется для передачи в том случае, когда устройству требуется более 16 битов в одном пакете. Также он используется для передачи данных между 8-разрядными и 16-разрядными устройствами по последовательной шине.

Примечание: Эта особенность используется только в том случае, когда длина данных, кратна выбранной базовой длине передачи.

Управление портом

Для связи с внешним миром SSC использует три вывода порта 3. P3.13/SCLK – линия тактового сигнала, P3.8/MRST и P3.9/MTSR – линии последовательной передачи/приема данных. Режим работы этих линий зависит от выбранного режима работы (master или slave). Для использования этих выводов для альтернативных функций необходимо установить «1» в соответствующих триггерах порта, так как выход триггера порта и выход линии альтернативной функции связаны по функции AND. В случае отсутствия необходимости в альтернативной функции, вывод альтернативной линии переводится на высокий уровень напряжения, что позволяет осуществлять ввод-вывод через триггер порта. Направление линий при использовании альтернативных функций зависит от режима работы. При переключении режимов SSC автоматически определяет необходимое направление (ввод или вывод). Однако пользователь сам обязан запрограммировать направление выводов порта, как указано в таблице. Использование выводов с открытым коллектором позволяет избегать конфликтов на шине и позволяет уменьшать затраты на аппаратный hand-snaking или на линии выбора slave-приемника. В этом режиме, в некоторых случаях отсутствует необходимость в переключении направления линии порта. Таблица, представленная ниже, суммирует необходимые значения в различных режимах и на различных линиях.

Вывод	Режим Master			Режим Slave		
	Функция	Триггер порта	Направление	Функция	Триггер порта	Направление
P3.13/SCLK	Тактовый выход	P3.13=1	DP3.13=1	Тактовый вход	P3.13=x	DP3.13=0
P3.9/MTSR	Выход данных	P3.9=1	DP3.9=1	Вход данных	P3.9=x	DP3.9=0
P3.8/MTSR	Вход данных	P3.8=1	DP3.8=1	Выход данных	P3.8=x	DP3.8=0

Примечание: «X» означает, что в этом месте может быть любое значение, однако рекомендуется устанавливать в этих битах «1», так как в случае переключения между режимами master и slave эти выводы уже находятся в корректном положении.

11.3 Создание тактового сигнала

SSC имеет собственный 16-разрядный тактовый генератор с функцией перезагрузки 16-разрядного значения, что позволяет устанавливать скорость передачи вне зависимости от характеристик внутренних таймеров GPT1 и GPT2.

Тактовый генератор работает с частотой, составляющей половину от частоты ЦПУ (10МГц при частоте ЦПУ 20МГц). Отсчет в таймере ведется в сторону уменьшения значений. Начало и окончание счета может быть произведено с помощью установки значения в бите SSCEN регистра SSCCON. Регистр SSCBR—двухфункциональный регистр тактового генератора/перезагрузки. При попытке чтения этого регистра при включенном SSC возвращается значение содержимого таймера. Чтение SSCBR при отключенном SSC возвращает значение для перезагрузки таймера. Также при отключенном SSC возможна запись значения для перезагрузки таймера.

Примечание: Запрещается производить запись в SSCBR при включенном SSC.

Скорость передачи и необходимое значение для перезагрузки можно вычислить по формулам:

$$B_{SSC} = \frac{f_{CPU}}{2 \cdot (<SSCBR> + 1)}$$

$$SSCBR = \left(\frac{f_{CPU}}{2 \cdot Baudrate_{SSC}} \right) - 1$$

<SSCBR> - содержимое регистра перезагрузки.

Максимальная скорость передачи при частоте 20МГц достигает 5Мбод. В таблице представлены некоторые возможные значения скорости передачи и необходимые для них значения для перезагрузки.

Скорость передачи	Длительность такта	Значение для перезагрузки
Зарезервировано	--- ---	0000 _H
5 Мбод	200 нс	0001 _H
3.3 Мбод	300 нс	0002 _H
2.5 Мбод	400 нс	0003 _H
2.0 Мбод	500 нс	0004 _H
1.0 Мбод	1 мкс	0009 _H
100 Кбод	10 мкс	0063 _H
10 Кбод	100 мкс	03E7 _H
1.0 Кбод	1 мс	270F _H
152.6 Бод	6.6 мс	FFFF _H

Примечание: Содержимое SSCBR должно быть больше нуля.

11.4 Механизм определения ошибок

Во время работы SSC в состоянии определить четыре типа ошибочных состояний. Ошибка получения и ошибка фазы определяются в любом режиме, ошибка передачи и ошибка скорости передачи может быть определена только в режиме slave. После обнаружения ошибки выставляется соответствующий флаг. Если до этого была установлена «1» в соответствующем бите разрешения ошибки, также генерируется запрос на прерывание по ошибке SSCEIR. После этого подпрограмма обслуживания прерывания по ошибке проверяет флаги ошибок для выявления типа имевшей место ошибки. Флаги ошибок автоматически не сбрасываются, и поэтому их необходимо программно очищать перед выходом из подпрограммы обслуживания прерывания.

Примечание: Подпрограмма обслуживания прерывания по ошибке обязана очистить вызвавший ее флаг, во избежания повторного входа в прерывание.

Ошибка получения: Определяется в режиме Master или Slave, если предыдущие данные не были считаны из буфера получения SSCRB до завершения приема новых данных. В этом случае устанавливается флаг SSCRE. После этого в буфер получения SSCRB будет записано новое значение, и старое значение будет потеряно.

Ошибка фазы: Определяется в режиме Master или Slave, в том случае когда сигнал на входе MRST, проверяемый с частотой ЦПУ, (в режиме Master) или на входе MTSR (в режиме Slave), изменяет свое значение в интервале, начинающемся за один такт ЦПУ до прихода фронта тактового сигнала передачи и заканчивающемся через два такта после прихода фронта. При этом устанавливается флаг SSCPE.

Ошибка скорости передачи: Определяется в Slave режиме, в том случае, когда входной тактовый сигнал отклоняется от запрограммированного более чем на 100%. Иными словами в том случае когда он составляет менее половины или более двойного значения от номинального. При этом устанавливается флаг SSCBE. Использование возможности определения этой ошибки позволяет настраивать частоту тактового генератора передачи slave устройства на частоту master устройства. Также использование этой функции позволяет определять дополнительные ложные или пропущенные такты на линии передачи тактового сигнала.

Примечание: В том случае когда эта ошибка имеет место при установленной «1» в бите SSCAREN, автоматически производится сброс блока SSC. Это необходимо для переинициализации SSC в том случае, когда определяется слишком много или слишком мало тактов.

Ошибка передачи: Определяется в Slave режиме в том случае, когда master инициировал передачу данных (Master послал тактовый сигнал), а

значение в буфере передачи SSCTB slave устройства не было изменено с прошлой передачи. При этом устанавливается флаг SSCTE. В том случае когда передача начинается без записи нового значения в буфер передачи, slave передает «старое» содержимое промежуточного регистра.

В том случае когда данный slave не выбран для передачи, это может приводить к конфликтам данных на линии приема/передачи в полудуплексном режиме (в режиме с открытым коллектором). В режиме с открытым коллектором не выбранный slave должен выставить на линию только единицы для корректной передачи данных выбранным slave, т.е. в буфер должно быть записано FFFF_H.

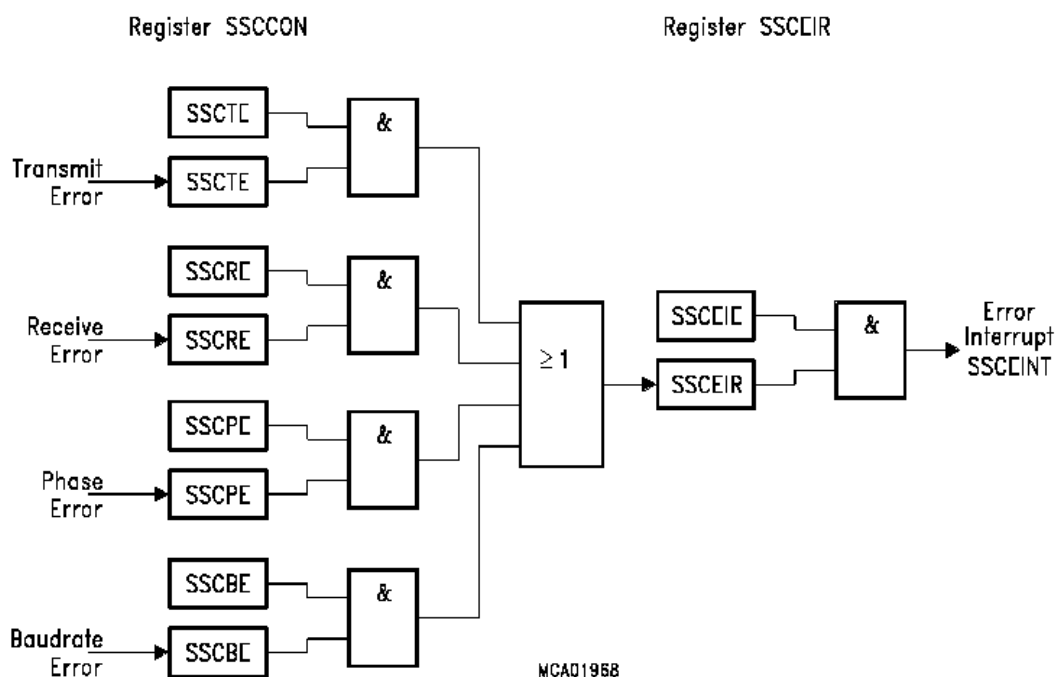


Рисунок 11-6

Управление прерываниями по ошибкам SSC

11.5 Управление прерываниями SSC

Для канала SSC существуют три регистра управления прерываниями. Регистр SSCTIC управляет прерыванием по передаче, SSCRIC – прерыванием по приему, SSCEIC – прерыванием по ошибке.

SSCTIC (FF72_H/B9_H)

SSCRIC (FF74_H/BA_H)

SSCEIC (FF76_H/BB_H)

12 Сторожевой таймер

Для предотвращения аппаратных или программных коллизий в архитектуре C167 предусмотрен сторожевой таймер. В том случае если имела место программная ошибка, и в результате было допущено переполнение таймера, будет начата процедура RESET. В случае совершения внутреннего RESET подается сигнал на выход \overline{RSTOUT} , необходимый для сброса внешней периферии. Сторожевой таймер следит за выполнением программы, при этом правильность выполнения программы определяется не допущением переполнения сторожевого таймера. Сторожевой таймер может сработать в тех случаях, когда программная ошибка вызвана аппаратной неисправностью. Сторожевой таймер предопределяет максимально возможное время работы в неисправном режиме.

Для функционирования сторожевого таймера предусмотрены два регистра: регистр содержимого сторожевого таймера WDT (только для чтения) и управляющий регистр для инициализации WDTCON.

Сторожевого таймера состоит из 16-разрядного счетчика, считающего по тактовому сигналу, составляющему либо $\frac{1}{2}$, либо $\frac{1}{128}$ часть частоты ЦПУ. 16-разрядный таймер реализован в виде двух объединенных 8-разрядных таймеров. Для изменения максимального времени ожидания обслуживания используется программная установка значения для 8 старших битов. Значения младших 8 битов аппаратно сбрасываются после каждого обслуживания.

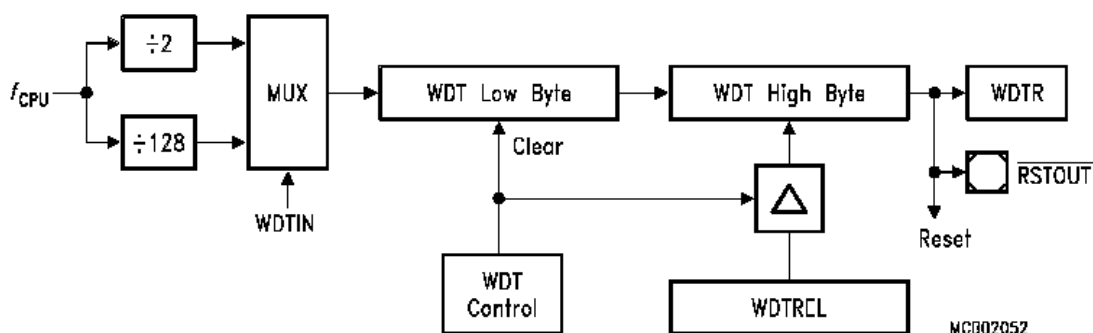


Рисунок 12-1

Блок-схема сторожевого таймера

Работа сторожевого таймера

Текущее значение сторожевого таймера содержится в не адресуемом побитно регистре WDT (только для чтения). Работа сторожевого таймера управляется в побитно адресуемом регистре WDTCON. В этом регистре содержится значение для перезагрузки старшего байта таймера, а также в этом регистре выбирается коэффициент деления частоты ЦПУ. В этом регистре также содержится флаг переполнения сторожевого таймера.

SFR

WDTCON (FFAE_H/D7_H)

Значение после RESET: 000X_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDTREL								-	-	-	-	-	-	WDT R	WDT IN
rw								-	-	-	-	-	-	r	rw

Бит	Функция
WDTIN	Выбор частоты отсчетов сторожевого таймера 0: Частота ЦПУ делится на 2 1: Частота ЦПУ делится на 128
WDTR	Флаг сброса сторожевого таймера Устанавливается в «1», когда происходит переполнение таймера Очищается аппаратно, при выполнении команды SRVDWT
WDTREL	Значения старшего байта сторожевого таймера при перезагрузке

Примечание: Если RESET инициируется сторожевым таймером, после RESET в данном регистре содержится 0002_H, в ином случае – 0000_H.

После любого RESET, сторожевой таймер начинает работать с частотой, составляющей ½ от частоты ЦПУ. Сторожевой таймер можно отключить по команде DISWDT. Команда DISWDT является защищенной 32-разрядной командой, и поэтому может быть выполнена только в интервале между RESET и командами EINIT (конец инициализации) или SRVDWT (сброс значения сторожевого таймера). Любая из этих двух команд запрещает последующее выполнение команды DISWDT.

В том случае когда сторожевой таймер не выключен по команде DISWDT, счет продолжается даже в режиме покоя. Если сторожевой таймер не дожидается команды сброса SRVDWT до достижения таймером значения FFFF_H, будет подан сигнал на внутренний RESET. При этом на выходе \overline{RSTOUT} также будет подан сигнал. Основное различие программного или аппаратного RESET от RESET по сигналу сторожевого таймера содержится в установке «1» в бите WDTR регистра WDTCON. После аппаратного RESET или команды SRVDWT значение этого бита очищается. Программа может

проверять значение этого бита, для определения причины, вызвавшей RESET.

В том случае когда цикл внешней шины не использует сигнал \overline{READY} , или определил активный уровень напряжения на входе \overline{READY} после запрограммированного количества waitstates, сторожевой таймер дожидается окончания цикла внешней шины, прежде чем начать RESET. Во всех иных случаях работа шины будет прервана.

Примечание: В случае работы в режиме аппаратного загрузчика после аппаратного RESET сторожевой таймер будет отключен.

Для недопущения переполнения сторожевого таймера необходимо периодически подавать команду сброса. Сторожевой таймер сбрасывается посредством защищенной 32-разрядной команды SRVWDT. При сбросе сторожевого таймера обнуляются младшие 8 битов, и в старшие биты регистра WDT загружается значение битового поля WDTREL регистра WDTCON. При сбросе сторожевого таймера производится обнуление значения бита WDTR. Команда SRVDWT закодирована таким образом, чтобы вероятность непреднамеренного выполнения команды (т.е. выполнения данной команды из неверного адреса) была сведена до минимума. Если команда SRVDWT не вписывается в формат защищенной команды, то, прежде чем команда будет выполнена, активируется ловушка ошибки защиты.

Период переполнения сторожевого таймера может быть запрограммирован двумя способами:

- **Входная частота** сторожевого таймера выбирается при помощи изменения значения бита WDTIN регистра WDTCON

- **Значение для перезагрузки** WDTREL программируется в регистре WDTCON

$$P_{WDT} = \frac{2^{(1+\langle WDTIN \rangle \cdot 6)} \cdot (2^{16} - \langle WDTREL \rangle \cdot 2^8)}{f_{CPU}}$$

Ниже в таблице приведены возможные временные пределы при частоте ЦПУ 20МГц.

Значение для перезагрузки WDTREL	Делитель частоты ЦПУ	
	2 (WDTIN = 0)	128 (WDTIN = 1)
FF _H	25.6мкс	1.6мс
00 _H	6.55мс	419мс

Примечание: Из соображений безопасности необходимо перед каждым сбросом сторожевого таймера перезаписывать значение в регистре WDTCON.

13 Аппаратный загрузчик

Встроенный аппаратный загрузчик позволяет загрузить начальную программу, которая будет выполняться после RESET, с помощью последовательного интерфейса. При работе в режиме аппаратного загрузчика не требуется наличие внешней или внутренней ROM с адресом 00'0000_H. Аппаратный загрузчик передает данные и код во внутреннюю RAM. При помощи второго уровня программы загрузчика возможна передача данных через последовательный интерфейс во внешнюю RAM. При этом отпадает необходимость во внутренней или внешней ROM.

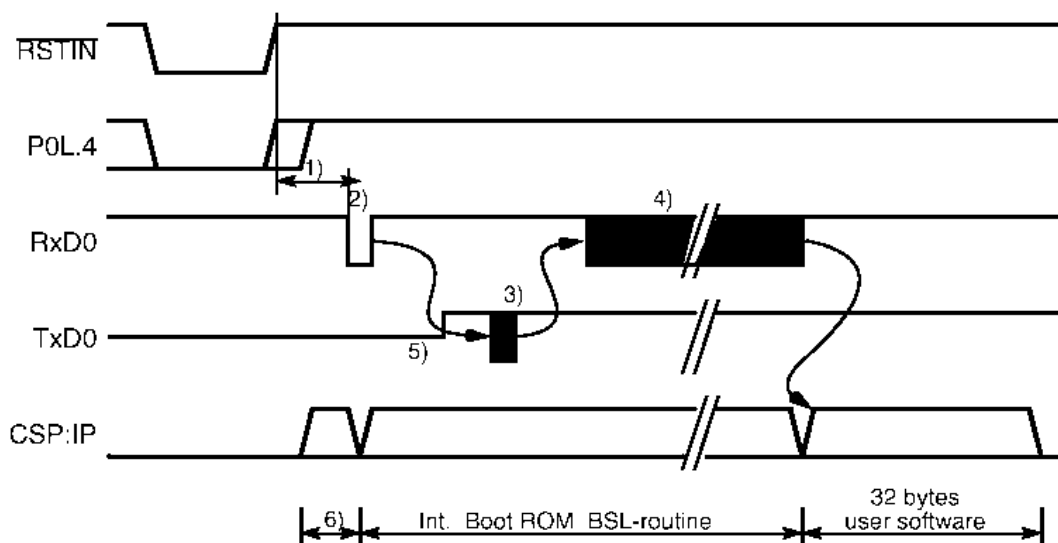


Рисунок 13-1

Временные диаграммы работы аппаратного загрузчика

- 1) BSL время инициализации больше 2мкс при частоте ЦПУ 20МГц
- 2) Посылается нулевой байт (1 стартовый бит + восемь нулевых битов + 1 стоповый бит)
- 3) Посланный C167 идентификационный байт
- 4) 32 байта кода или данных, посланные извне
- 5) Внимание! TxD0 передает только необходимое время после получения нулевого байта (2.5мкс при частоте ЦПУ 20МГц)
- 6) Внутренняя ROM загрузчика

Аппаратный загрузчик может использоваться для загрузки полных приложений для микроконтроллеров без внутренней ROM. Также он используется для загрузки программ в готовые системы для тестирования или калибровки. Еще одним применением является загрузка приложения для программирования Flash памяти (в тех устройствах где она есть).

Механизм BSL можно использовать и для стандартного старта, и для особых случаев (обновление firmware, окончательное программирование и тестирование).

Активизация аппаратного загрузчика

C167 входит в режим BSL, в том случае когда на вход P0L.4 подан низкий уровень напряжения перед окончанием аппаратного RESET. В этом случае вне зависимости от выбранного режима шины, активизируется аппаратный загрузчик. Код аппаратного загрузчика хранится в специальной области Boot-ROM. Для этой области не требуется отведения части стандартной mask ROM или flash памяти.

После входа в режим BSL и после необходимой настройки, C167 начинает следить за линией RXD0, для обнаружения нулевого байта (один стартовый бит, восемь нулевых битов, один стоповый бит). C167 вычисляет скорость передачи данных по продолжительности этого нулевого байта, после этого ASC0 настраивается на эту скорость и TxD0 переключается на вывод данных. TxD0 возвращает идентификационный байт с той же скоростью передачи, после чего начинается загрузка данных.

Идентификационный байт указывает на тип boot-устройства.

8xC166 55_H

C165 B5_H

C167 C5_H (предыдущая версия возвращает значение A5_H)

После входа в режим DSL устанавливается следующая конфигурация (те значения, которые отличаются от нормальных значений после RESET, выделены жирным шрифтом):

Сторожевой таймер	отключен
CP	FA00 _H
SP	FA40 _H
S0CON	8011_H
S0BG	acc. to 00 byte
STKUN	FA40 _H
STKOV	FA0C _H 0<->C
BUSCON0	согласно начальной конфигурации
P3.10/TXD0	«1»
DP3.10	«1»

Поскольку после нормального RESET сторожевой таймер отключен, программа, загруженная аппаратным загрузчиком, не ограничена во времени при работе.

Примечание: При работе аппаратного загрузчика, программный код не будет выполняться из внутренней ROM.

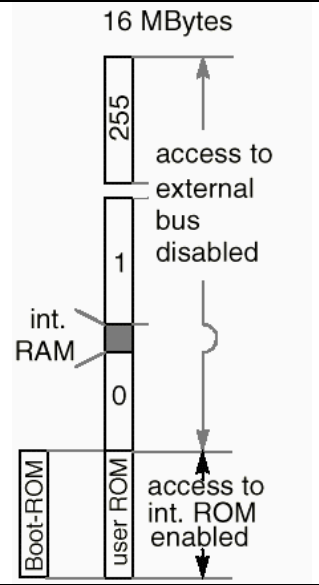
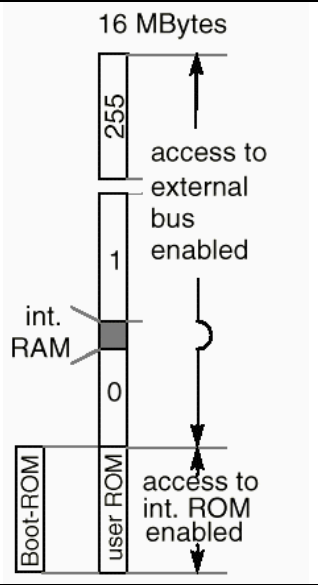
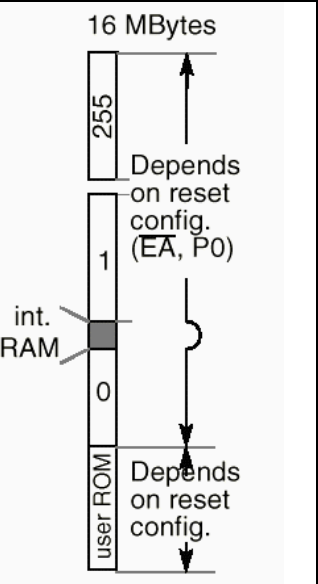
После отсылки идентификационного байта задействован ASC0-приемник, для приема 32 байт. Для работы BSL достаточно полудуплексного режима.

Конфигурация памяти после RESET

Конфигурация памяти (т.е. возможность доступа) после RESET в режиме аппаратного загрузчика отличается от конфигурации в стандартном режиме. Во время работы в режиме BSL, значение на входе \overline{EA} не оказывает влияние, и доступ к внутренней ROM частично перераспределяется. Выборка кода производится из специальной Boot-ROM, в то время как возможно осуществлять чтение данных из внутренней пользовательской ROM.

Примечание: Пользовательская программа не должна осуществлять попыток выполнения кода из внутренней памяти в режиме BSL. Эти попытки доступа будут перенаправлены в Boot-ROM.

В тех случаях когда область внутренней памяти располагается в сегменте «1», Boot-ROM также перемещается в сегмент «1»,

			
Режим BSL активен	Да (P0L.4 = 0)	Да (P0L.4 = 0)	Нет (P0L.4 = 1)
\overline{EA}	«1»	«0»	доступ приложения
Выборка кода из внутренней ROM	Доступ к Boot-ROM	Доступ к Boot-ROM	Доступ к пользовательской ROM
Выборка данных из внутренней ROM	Доступ к пользовательской ROM	Доступ к пользовательской ROM	Доступ к пользовательской ROM

Загрузка начального кода

После отсылки идентификационного байта BSL входит в режим получения 32 байтов через ASC0. Эти байты последовательно сохраняются во внутренней ROM по адресам 00`FA40_H – 00`FA5F. Для выполнения загруженного кода BSL производит переход по адресу первой загруженной команды 00`FA40_H. После этого загрузка данных через ASC0 будет прервана, однако микроконтроллер продолжит работу в режиме BSL. Наиболее вероятно, что этот код содержит подпрограмму загрузки необходимых данных или кода, так как среднее приложение нуждается более чем в 16 командах. Для второго уровня доступа может использоваться предустановленный режим ASC0 для получения данных и сохранения их в определенных пользователем адресах.

Второй уровень полученного кода может быть кодом конечного приложения. Также этот код может содержать более сильную подпрограмму загрузки, которая расширяет протокол передачи. Также возможно, что этот код содержит подпрограмму изменения конфигурации системы и включения интерфейса шины для сохранения полученных данных во внешней памяти.

Этот процесс может иметь несколько приближений. Во всех случаях C167 остается в режиме BSL, т.е. сторожевой таймер отключен и ограничен доступ во внутреннюю ROM. Все попытки выборки кода из области внутренней ROM будут перенаправлены в специальную Boot-ROM.

Выход из режима BSL

Для выполнения программы в нормальном режиме, необходимо сначала прервать режим BSL. C167 выходит из режима BSL посредством программного RESET (игнорируя сигнал на входе P0L.4) или посредством аппаратного RESET (при этом необходимо подать «1» на вход P0L.4). После RESET в соответствии с сигналом на входе \overline{EA} микроконтроллер начнет выполнять код из адреса 00`0000_H либо из внутренней ROM, либо из внешней памяти.

Выбор baud rate для BSL

Расчет baud rate для C167 в зависимости от длительности первого нулевого байта позволяет настраивать аппаратный загрузчик на широкий диапазон baud rate. При этом однако, для обеспечения гарантированной передачи данных, должны быть соблюдены верхний и нижний пределы.

$$B_{C167} = \frac{f_{CPU}}{32 \cdot (S0BRL + 1)}$$

Для измерения длительности нулевого байта используется таймер T6. Так как таймер T6 имеет достаточно большую квантованность значений, то результат его измерения используется для первого приближения к реальной скорости передачи. Следующее приближение заключается в вычислении

значения для перезагрузки S0BRL из содержимого таймера. На формулах, приведенных ниже, показаны эти зависимости:

$$S0BRL = \frac{T6 - 36}{72}, \quad T6 = \frac{9}{4} \cdot \frac{f_{CPU}}{B_{Host}}$$

Для корректной передачи данных извне максимальное отклонение между внутренней скоростью передачи и реальной скоростью передачи внешнего устройства должна составлять не более 2.5%.

$$F_B = \left| \frac{B_{Contr} - B_{Host}}{B_{Contr}} \right| \cdot 100\% \text{ - вычисление погрешности}$$

Отклонение baud rate является нелинейной функцией, зависящей от частоты ЦПУ и скорости передачи внешнего устройства. Максимум F_B увеличивается с ростом скорости передачи внешнего устройства.

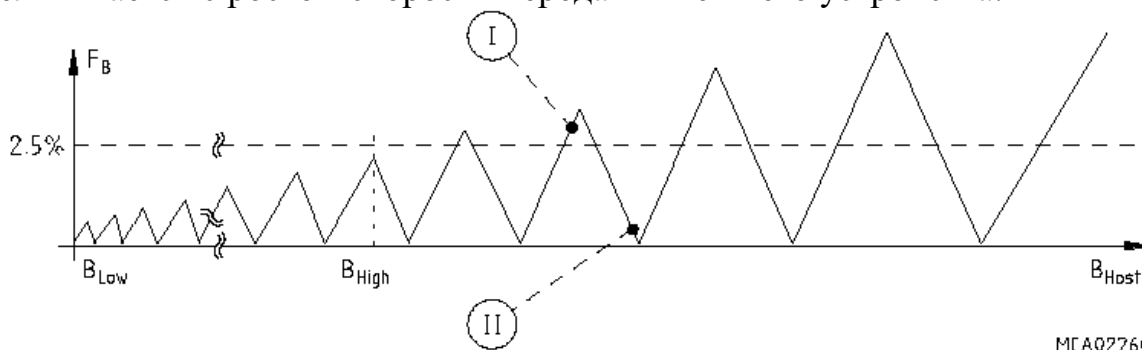


Рисунок 13-2

Погрешность скорости передачи в зависимости от скорости передачи внешнего устройства и C167

Минимальная скорость передачи (на рисунке B_{Low}) определяет максимально возможное значение в таймере T6 при измерении длительности нулевого байта, т.е. в зависимости от частоты ЦПУ. Так как максимальное значение в счетчике T6 равно 2^{16} , то при частоте ЦПУ 20МГц скорость передачи составляет 687 Бод. В этом случае наименьшая стандартная скорость передачи 1200Бод. Скорость передачи менее 687 Бод приведет к переполнению таймера. В этом случае ASC0 не будет корректно инициализирован.

Максимальная скорость передачи (B_{High}) выбирается из условия не превышения предела. Максимальная стандартная скорость передачи, удовлетворяющая этим условиям, - 19200 Бод.

Наивысшая скорость передачи может быть определена для не выходящих за пределы реальных отклонений. Скорость может превышать этот предел (обозначенная I на рисунке выше), в то время как большая скорость (обозначенная II) остается внутри границ. Это зависит от интерфейса внешнего устройства.

14 Блок сравнения и захвата (CAPCOM)

В архитектуру C167 включены два схожих блока CAPCOM. Основное отличие блоков заключается в способе подключения к параллельным портам. В каждый блок включает в себя 32 канала, связанных с 4 таймерами. Захват содержимого таймера CAPCOM блока может осуществляться по внутреннему или по внешнему сигналу. В CAPCOM-блоке возможно осуществлять сравнение содержимого таймера с значением регистра, и согласно результату сравнения выдавать выходные сигналы. Использование функций CAPCOM-блоков позволяет осуществлять с минимальной загрузкой ЦПУ создание и управление временными зависимостями по 16 каналам.

С точки зрения программиста CAPCOM представляет из себя набор SFR-регистров. К регистрам управления CAPCOM также можно отнести регистры управления портами, используемых в качестве альтернативных функций.

Ports & Direction Control Alternate Functions	Data Registers	Control Registers	Interrupt Control
DP1H E	T0	T01CON	T0IC
P1H	T0REL		
ODP2 E	T1		T1IC
DP2	T1REL		
P2	T7 E	T78CON	T7IC E
ODP3 E	T7REL E		
DP3	T8 E		T8IC E
P3	T8REL E		
ODP7 E	CC0-3	CCM0	CC0IC-3IC
DP7	CC4-7	CCM1	CC4IC-7IC
P7	CC8-11	CCM2	CC8IC-11IC
ODP8 E	CC12-15	CCM3	CC12IC-15IC
DP8	CC16-19	CCM4	CC16IC-19IC E
P8	CC20-23	CCM5	CC20IC-23IC E
CC0IO/P2.0...CC15IO/P2.15	CC24-27	CCM6	CC24IC-27IC E
CC16IO/P8.0...CC23IO/P8.7	CC28-31	CCM7	CC28IC-31IC E
CC24IO/P1H.4...CC27IO/P1H.7			
CC28IO/P7.4...CC31IO/P7.7			
ODPx Port x Open Drain Control Register		TxREL CAPCOM Timer x Reload Register	
IOPx Port x Direction Control Register		Tx CAPCOM Timer x Register	
Px Port x Data Register		CC0...15 CAPCOM1 Register 0...15	
		CC16...31 CAPCOM2 Register 16...31	
		CCM0...3 CAPCOM1 Mode Control Register 0...3	
		CCM4...7 CAPCOM2 Mode Control Register 4...7	
T01CON CAPCOM1 Timers T0 and T1 Control Register		CC0...15IC CAPCOM1 Interrupt Control Register 0...15	
T78CON CAPCOM2 Timers T7 and T8 Control Register		CC16...31IC CAPCOM2 Interrupt Control Register 16...31	
T0IC/T1IC CAPCOM1 Timer 0/1 Interrupt Control Register			
T7IC/T8IC CAPCOM2 Timer //8 Interrupt Control Register			

Рисунок 14-1

SFR-регистры и выходы портов связанные с блоком CAPCOM

CAPCOM в основном используется для работы с высокоскоростными задачами ввода-вывода и для создания импульсов, ШИМ-сигналов, или для вычисления времени прихода сигналов. Его можно использовать для создания 16 программных таймеров. Минимальная квантованность сигналов блока CAPCOM – 400нс при частоте ЦПУ 20МГц.

Каждый из CAPCOM-блоков содержит два 16-разрядных таймера (T0/T1 для CAPCOM1 и T7/T8 для CAPCOM2). Для каждого таймера имеет место собственный регистр перезагрузки (TxREL). В каждый CAPCOM-блок включено 16 регистров сравнения/захвата двойного назначения (CC0-CC15 для CAPCOM1 и CC16-CC31 для CAPCOM2).

Входной тактовый сигнал CAPCOM-блока может определяться частотой ЦПУ, либо может определяться по переполнению или опустошению таймера T6 блока GPT2. T0 и T7 могут работать в режиме счетчика (от внешнего сигнала).

Каждый регистр захвата и сравнения индивидуально может быть запрограммирован на функцию захвата или функцию сравнения. Каждый регистр может быть привязан к одному из двух таймеров соответствующего блока. Для каждого регистра захвата и сравнения имеется собственный вывод порта, служащий в качестве входа функции захвата и выхода для функции сравнения (за исключением CC27... CC24 на выводах P1H.7... P1H.4, которые служат только для функции захвата). Захват происходит по приходу на вход фронта сигнала, при этом текущее значение таймера сохраняется в соответствующем регистре захвата и сравнения. При работе в функции сравнения уровень напряжения на выходе порта изменяется, в зависимости от результата сравнения значения регистра захвата и сравнения с текущим значением таймера. По каждому из этих событий или по переполнению таймера генерируется соответствующий запрос на прерывание.

На рисунке, представленном ниже, показана структура блока CAPCOM.

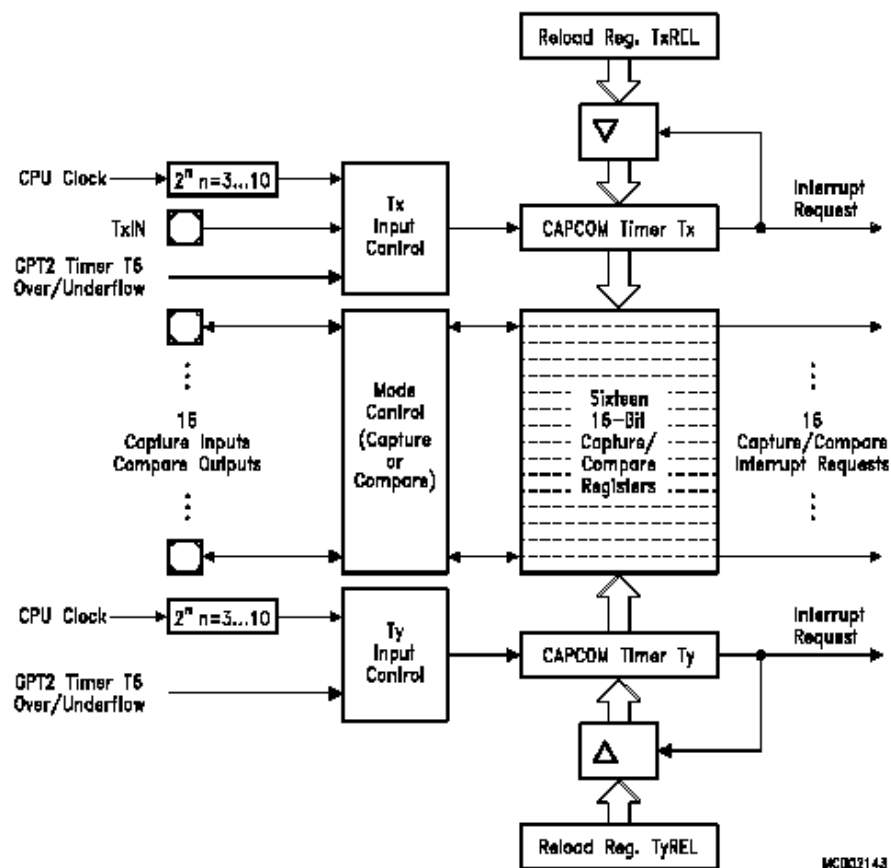


Рисунок 14-2

Блок-схема CAPCOM-блока

Примечание: CAPCOM2 имеет 16 входов захвата, но только 12 выходов сравнения

14.1 CAPCOM таймеры

Регистры захвата и сравнения каждого CAPCOM-блока, базируются независимо друг от друга на таймерах T0/T1 и T7/T8 (с минимальной длительностью такта 400нс при частоте ЦПУ 20МГц). Эти таймеры также могут использоваться независимо от регистров захвата и сравнения.

Структура всех четырех таймеров одинакова, однако выбор входных сигналов таймеров T0/T7 отличается от сигналов таймеров T1/T8.

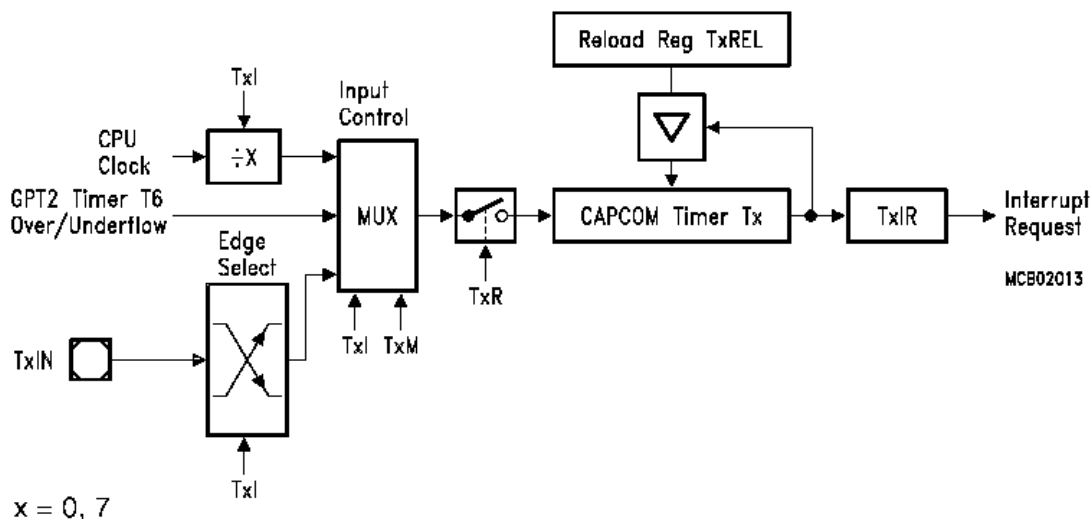


Рисунок 14-3

Блок-схема таймеров T0 и T7

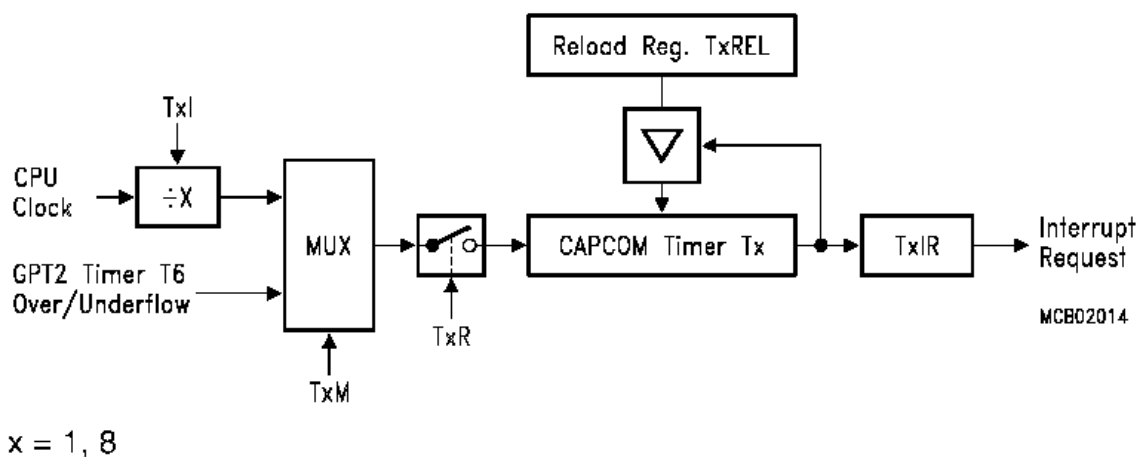


Рисунок 14-4

Блок-схема таймеров T1 и T8

Примечание: В том случае когда внешний входной сигнал подключен к входным линиям таймеров T0 и T7, то возможна синхронная реакция таймеров на входной сигнал. Таким образом можно использовать

два таймера как один, содержимое которого может сравниваться сразу с 32 регистрами захвата.

Функции CAPCOM-таймеров управляются с помощью побитно адресуемых 16-разрядных управляющих регистров T01CON и T78CON. Старший байт T01CON управляет работой T1, а младший байт – T0. В свою очередь старший байт T78CON управляет работой T8, а младший – T7. Для всех четырех таймеров параметры управления идентичны (за исключением внешних входов).

SFR

T01CON (FF50_H/A8_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	T1R	-	-	T1M	T1I	-	T0R	-	-	T0M	T0I	-	-	-	-
-	rw	-	-	rw	rw	-	rw	-	-	rw	rw	-	-	-	-

SFR

T01CON (FF20_H/90_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	T8R	-	-	T8M	T8I	-	T7R	-	-	T7M	T7I	-	-	-	-
-	rw	-	-	rw	rw	-	rw	-	-	rw	rw	-	-	-	-

Бит	Функция
TxI	Выбор входа таймера/счетчика x В режиме таймера (TxM = 0): Входная частота = $f_{CPU} / 2^{(<TxI>+3)}$ В режиме счетчика (TxM = 1): X00 – по переполнению/опустошению T6 GPT2 X01 – по положительному фронту на входе TxIN*) X10 – по отрицательному фронту на входе TxIN*) X11 – по обоим фронтам на входе TxIN*)
TxM	Выбор режима работы таймера/счетчика X 0: Режим таймера (от внутреннего тактового сигнала) 1: Режим счетчика (от внешнего сигнала или T6)
TxR	Управление работой таймера/счетчика X TxR = 0: Tx стоит TxR = 1: Tx работает

*) Этот режим доступен только для T0 и T7. В случае выбора этого режима для T1 или T8, эти таймеры будут остановлены.

Флаги TxR позволяют включать и выключать таймеры. В дальнейшем при описании режимов работы таймера подразумевается, что все таймеры включены.

Во всех режимах счет всегда идет в положительном направлении. Текущее значение таймера доступно ЦПУ через регистр Tx, расположенный в не адресуемой побитно области SFR. В том случае когда запись значения в таймер производится во время изменения значения таймера или во время перезагрузки значения, более высокий приоритет имеет операция записи ЦПУ. При этом для корректной работы таймера функция инкрементирования или изменения значения отключается.

Режим таймера

Режим таймера или счетчика выбирается в битах TxM регистров T01CON и T78CON. В режиме таймера (TxM = 0) в качестве входного тактового сигнала таймера используется тактовый сигнал ЦПУ с входным делителем. Значение делителя выбирается в битовом поле TxI.

Частота тактового сигнала f_{Tx} для таймера Tx определяется в виде функции:

$$f_{Tx} = \frac{f_{CPU}}{2^{(<TxI>+3)}}$$

В момент перехода таймера из FFFF_H в 0000_H, в таймер перезагружается значение из регистра TxREL. Это значение определяет период P_{Tx} между двумя последовательными переполнениями таймера Tx:

$$P_{Tx} = \frac{(2^{16} - <TxREL>) \cdot 2^{(<TxI>+3)}}{f_{CPU}}$$

Входная частота, разрешение и период для различных значений TxI представлены ниже в таблице:

$f_{CPU} = 20\text{МГц}$	Выбор значения TxI							
	000	001	010	011	100	101	110	111
Делитель	8	16	32	64	128	256	512	1024
Входная частота	2.5 МГц	1.25 МГц	625 кГц	312.5 кГц	156.25 кГц	78.125 кГц	39.06 кГц	19.53 кГц
Разрешение	400нс	800нс	1.6 мкс	3.2 мкс	6.4 мкс	12.8 мкс	25.6 мкс	51.2 мкс
Период	26 мс	52.5 мс	105 мс	210 мс	420 мс	840 мс	1.68 с	3.36 с

Если значение обоих таймеров CAPCOM-блока одновременно инкрементируется или перезагружается, то всегда T0 обслуживается на один такт ЦПУ раньше чем T1, и соответственно T7 обслуживается раньше чем T8.

Режим счетчика

В режиме счетчика ($TxM = 1$) в качестве входного тактового сигнала таймера выступает сигнал о переполнении/опустошении таймера T6 блока GPT2. В дополнение к этому T0 и T7 могут работать от внешних тактовых сигналов. В качестве сигнала для инкрементирования содержимого T0/T7 может выступать положительный, отрицательный или одновременно оба фронта сигнала на входах T0IN/P3.0 и T7IN/P2.15.

В том случае когда T1 и T8 запрограммированы для работы в режиме счетчика, значение битового поля TxI используется для определения источника тактового сигнала. В том случае если в TxI таймеров T1 и T8 записано значение отличное от X00, то T1 и T8 останавливаются.

Если T0 и T7 запрограммированы на работу в режиме счетчика, тип фронта тактового сигнала выбирается в битовом поле TxI (в том случае если используется внешний сигнал).

Примечание: Для использования T0IN и T7IN в качестве тактовых входов, необходимо в соответствующих битах регистров направления портов (DP3.0 или DP2.15) установить «0».

В том случае если они настроены на выход, таймер будет переключаться при изменении значения в бите входного триггера порта P3.0 или P2.15. Эта функция может пригодиться при тестировании и отладки.

Максимальная внешняя тактовая частота для таймеров T0 или T7 в режиме счетчика составляет $f_{CPU}/16$ (1.25МГц при частоте ЦПУ 20МГц). Для уверенности в том что фронт сигнала захвачен, необходимо как минимум в течении 8 тактов ЦПУ сохранять значение на входе порта. Инкрементирование значения в SFR-регистре счетчика T0/T7 происходит в течении 8 тактов ЦПУ после прихода фронта сигнала.

Перезагрузка

Загрузка в таймер 16-разрядного значения, сохраненного в соответствующем регистре перезагрузки, происходит в обоих режимах при каждом переполнении таймера (переходе через FFFF к 0000). При этом в таймер загружается значение из регистра перезагрузки TxREL. После этого таймер продолжает инкрементирование загруженного значения.

Регистры перезагрузки TxREL не адресуются побитно.

14.2 Прерывания таймеров CAPCOM-блока

В случае переполнения таймера выставляется флаг запроса на прерывание T_xIR. В тех случаях когда включено разрешение прерываний в бите T_xIE, этот флаг может быть сигналом для входа в стандартное прерывание или для входа в РЕС-обслуживание.

Каждый таймер имеет свой регистр управления прерываниями:

T0IC(FF9C_H/CE_H)

T1IC(FF9E_H/CF_H)

T7IC(F17A_H/BE_H)

T8IC(F17C_H/BF_H)

Примечание: Полное описание этих регистров можно найти в разделе о прерываниях.

14.3 Регистры захвата и сравнения

16-разрядные регистры захвата и сравнения CC0 – CC31 и таймеры T0/T1 и T7/T8 используются в качестве регистров данных в операциях захвата или сравнения. Регистры захвата и сравнения не являются побитно адресуемыми.

Каждый из регистров CC0... CC31 может быть индивидуально запрограммирован на режим захвата или на один из четырех режимов сравнения (за исключением CC24... CC27). Каждый регистр может быть привязан к одному из двух таймеров соответствующего блока CAPCOM. Использование особой комбинации режимов сравнения позволяет ввести режим сравнения «double-register». В тех случаях когда часть регистров сравнения и захвата не используется, их можно использовать для хранения данных основного назначения.

Работа всех 32 регистров сравнения и захвата управляется с помощью 8-ми побитно адресуемых 16-разрядных регистров управления режимом CSM0... CSM7. Структура всех регистров организована идентично.

Регистры управления режимом захвата и сравнения для CAPCOM1-блока (CC0 ... CC15).

SFR

CCM0 (FF52_H/A9_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACC3	CCMOD3			ACC2	CCMOD2			ACC1	CCMOD1			ACC0	CCMOD0		
rw	rw			rw	rw			rw	rw			rw	rw		

SFR

CCM1 (FF54_H/AA_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACC7	CCMOD7			ACC6	CCMOD6			ACC5	CCMOD5			ACC4	CCMOD4		
rw	rw			rw	rw			rw	rw			rw	rw		

SFR

CCM2 (FF56_H/AB_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACC11	CCMOD11			ACC10	CCMOD10			ACC9	CCMOD9			ACC8	CCMOD8		
rw	rw			rw	rw			rw	rw			rw	rw		

SFR

CCM3 (FF58_H/AC_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACC15	CCMOD15			ACC14	CCMOD14			ACC13	CCMOD13			ACC12	CCMOD12		
rw	rw			rw	rw			rw	rw			rw	rw		

Регистры управления режимом захвата и сравнения для CAPCOM2-блока (CC16 ... CC32).

SFR

CCM4 (FF22_H/91_H)
Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACC 19	CCMOD19			ACC 18	CCMOD18			ACC 17	CCMOD17			ACC 16	CCMOD16		
rw	rw			rw	rw			rw	rw			rw	rw		

SFR

CCM1 (FF24_H/92_H)
Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACC 23	CCMOD23			ACC 22	CCMOD22			ACC 21	CCMOD21			ACC 20	CCMOD20		
rw	rw			rw	rw			rw	rw			rw	rw		

SFR

CCM6 (FF26_H/93_H)
Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACC 27	CCMOD27			ACC 26	CCMOD26			ACC 25	CCMOD25			ACC 24	CCMOD24		
rw	rw			rw	rw			rw	rw			rw	rw		

SFR

CCM7 (FF28_H/94_H)
Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACC 31	CCMOD31			ACC 30	CCMOD30			ACC 29	CCMOD29			ACC 28	CCMOD27		
rw	rw			rw	rw			rw	rw			rw	rw		

Бит	Функция
CCMODx	Выбор режима для регистра CCx Возможные режимы сравнения и захвата приведены ниже
ACCx	Бит привязки регистра CCx 0: CCx привязан к T0 (CAPCOM1) / T7(CAPCOM2) 1: CCx привязан к T1 (CAPCOM1) / T8(CAPCOM2)

Выбор режимов захвата и сравнения

CCMODx	Выбранный режим работы
0 0 0	Режим сравнения и захвата отключен
0 0 1	Захват по положительному фронту на входе CsxIO
0 1 0	Захват по отрицательному фронту на входе CsxIO
0 1 1	Захват по обоим фронтам на входе CsxIO
1 0 0	Режим сравнения 0: Только прерывание (Interrupt Only) Допускается несколько прерываний за один период таймера; Разрешен режим сравнения «double register» для регистров CC8... CC15 и CC16... CC31.
1 0 1	Режим сравнения 1: Переключение выходного сигнала в зависимости от результатов сравнения Допускается несколько результативных сравнений за один период таймера; В режиме сравнения «double-register» необходимо установить в этот режим регистры CC0... CC7 и CC16 ... CC23
1 1 0	Режим сравнения 2: Только прерывание (Interrupt Only) Допускается только одно прерывание за период таймера
1 1 1	Режим сравнения 3: Установка значения выходного сигнала в зависимости от результатов сравнения Сброс значений на выходах при каждом переполнении таймера; Допускается только одно прерывание за период таймера

Примечание: Каналы 24... 27 могут создавать запрос на прерывание но не выводов портов для сигнала. Событие захвата или сравнения в канале 31 может использоваться для переключения АЦП на инжектированный канал (в том случае если он включен).

14.4 Режим захвата

В ответ на внешнее событие значение таймера (T0/T1 или T7/T8 в зависимости от используемого блока CAPCOM и от состояния в бите управления ACCx) сохраняется в регистре CCx. Имеется возможность для выбора типа фронта сигнала управления захватом.

Тип фронта переключения выбирается в битах CCMODx регистра управления режима CAPCOM. Вне зависимости от типа события вызвавшего захват, устанавливается флаг запроса на прерывание CCxIR.

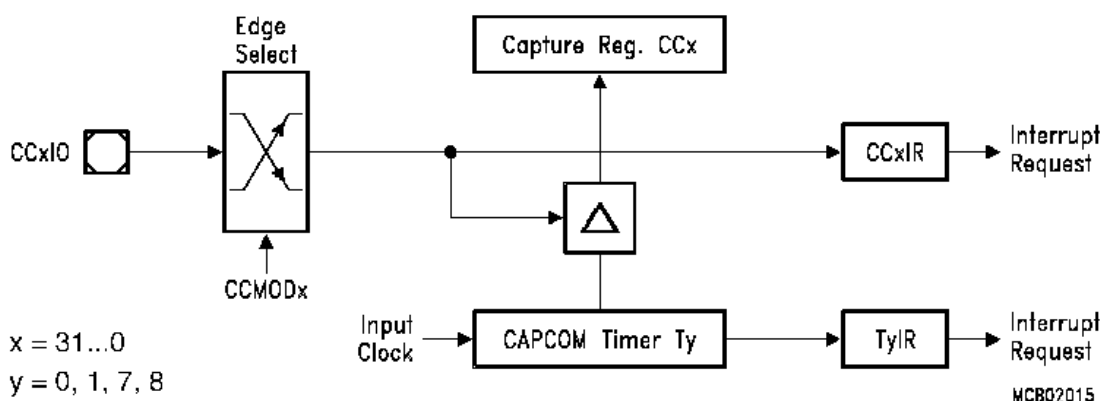


Рисунок 14-5

Блок-схема режима захвата

В случае совершения захвата по внешнему сигналу CCxIO, необходимо установить «0» в соответствующем бите регистра направления порта. Для гарантированного захвата, необходимо как минимум в течении 8 тактов ЦПУ удерживать сигнал на входе захвата без изменения.

Входной сигнал захвата будет сканироваться во время 8 тактов. В том случае если одновременно с этим произошло изменение значения в таймере или инкрементирование значения таймера, будет произведен захват нового значения таймера.

В том случае, когда CCxIO настроен на вывод данных, захват может быть произведен программно, путем изменения значения на выходе порта.

14.5 Режимы сравнения

Режимы сравнения позволяют с минимальной загрузкой процессора переключать события (прерывания и/или выходные сигналы). Во всех режимах сравнения, сохраненное в регистре CCx, значение постоянно сравнивается с содержимым таймера (T0/T1 или T7/T8). В тот момент, когда значение таймера достигает значения регистра CCx, выдается сигнал на выход CCxIO (за исключением CC24IO... CC27IO) и устанавливается флаг запроса на прерывание CCxIR.

В том случае если два регистра запрограммированы на одинаковое значение сравнения, будут выставлены флаги запросов на прерывание и также изменены значения на выходах в течении 8 тактов ЦПУ после достижения таймером сравниваемого значения. После этого другие результативные сравнения по этому значению будут запрещены, до тех пор пока таймер снова не достигнет этого значения, или значение таймера не будет программно изменено. После RESET работа в режиме сравнения будет начата, если значение в таймере инкрементируется или программно изменяется и выбирается один из режимов сравнения.

Различные режимы сравнения можно установить в поле CCMODx регистра управления режимом захвата и сравнения.

Режим сравнения 0

Режим Interrupt Only используется для создания программных временных зависимостей. Режим сравнения 0 выбирается путем установки 100 в битовом поле CCMODx.

В этом режиме флаг запроса на прерывание устанавливается при достижении таймера значения регистра CCx. Если значение регистра CCx изменяется несколько раз в течении одного периода таймера, возможно несколько результативных сравнений за один период таймера. При работе в этом режиме вывод порта CCxIO не изменяет своего значения.

В том случае когда для регистров CC8... CC15 или CC24... CC31 выбран режим 0, работа в режиме double-register возможна, если банк регистров 1 запрограммирован в режим сравнения 1.

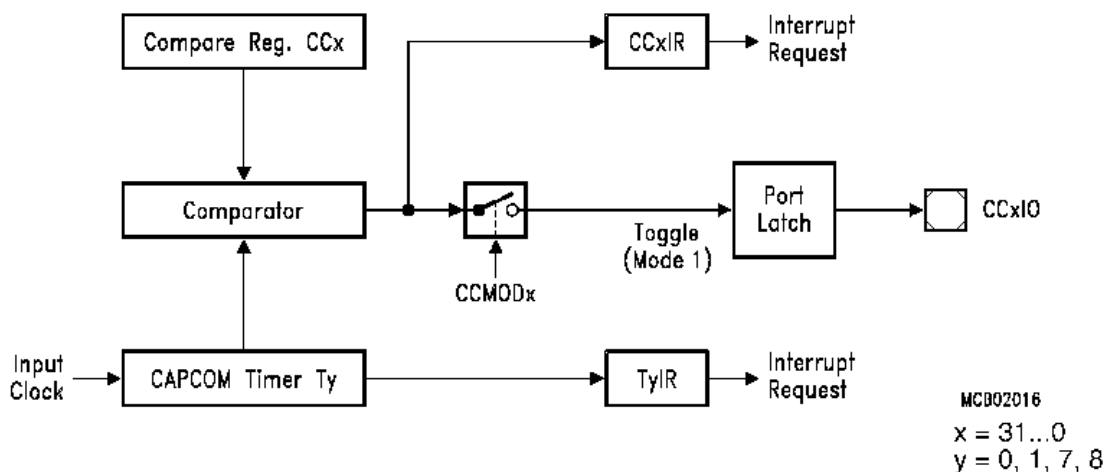


Рисунок 14-6

Блок-схема режима 0 и 1

В примере приведенном ниже, новое значение cv2 записывается в регистр CCx после событий сравнения #1 и #3, а cv1 записывается после событий #2 и #4. Данная зависимость имеет место в результате обработки запросов на прерывание от таймера T1, и запросов на прерывание от регистров CCx. Моменты совершения запросов CCxIR определяется значениями cv1 и cv2.

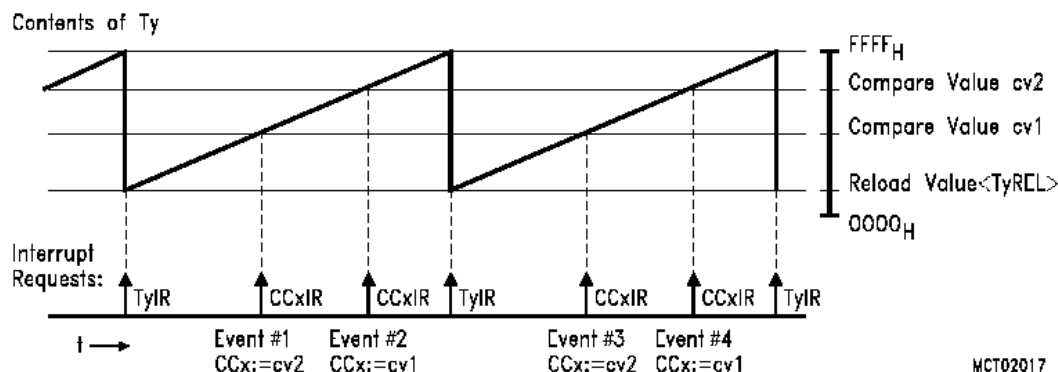


Рисунок 14-7

Пример для режимов 0 и 1

Режим сравнения 1

Режим сравнения 1 выбирается путем установки 101 в битовом поле CCMODx.

При работе в этом режиме, в случае достижения таймером значения в регистре CCx, устанавливается флаг запроса на прерывание CCxIR и инвертируется значение на выходах CCxIO. При этом последовательность действий для инвертирования значения в выходном триггере такова, что сначала значение в выходном триггере порта читается, затем инвертируется и записывается назад.

При работе в режиме сравнения 1 возможна обработка нескольких событий за один период таймера. В случае переполнения таймера не происходит инвертирования значения на выходе и не запрещаются или не разрешаются дальнейшие сравнения.

Для использования выводов порта в качестве выводов сигналов CCxIO необходимо устанавливать направление порта на вывод данных в соответствующем регистре направления порта.

В режиме сравнения 1 значение на выходе порта инвертируется после каждого результативного сравнения.

Примечание: Если одновременно с программной записью в порт имеет место инвертирование значения на выходе вследствие результативного сравнения, то программная запись имеет более высокий уровень приоритета. При этом не будет произведено инвертирование выходного значения по сигналу сравнения.

В случае программирования регистров CC0... CC7 или CC16... CC23 в режим 1, эти регистры могут работать в режиме сравнения double-compare в том случае, если регистр соответствующего банка 1 запрограммирован в режим 0.

Режим сравнения 2

Режим сравнения 2 аналогичен режиму сравнения 0, главное отличие состоит в том, что имеется возможность создания только одного запроса прерывания за один период таймера. Режим сравнения 2 выбирается путем установки 110 в битовом поле CCMODx.

При первом обнаружении достижения таймером значения регистра CCx устанавливается флаг запроса на прерывание CCxIR. При этом результат сравнения не выводится на вывод порта. Однако после первого результативного сравнения, до тех пор пока не произойдет переполнение таймера, на все остальные результативные сравнения в этот период таймера не будет устанавливаться запрос на прерывание.

На рисунке приведенном ниже, после события #1 в регистр CCx записывается значение cv2. Однако даже в том случае если новое значение

регистра превышает старое, следующее результативное сравнение #2 будет иметь место только в следующем периоде таймера.

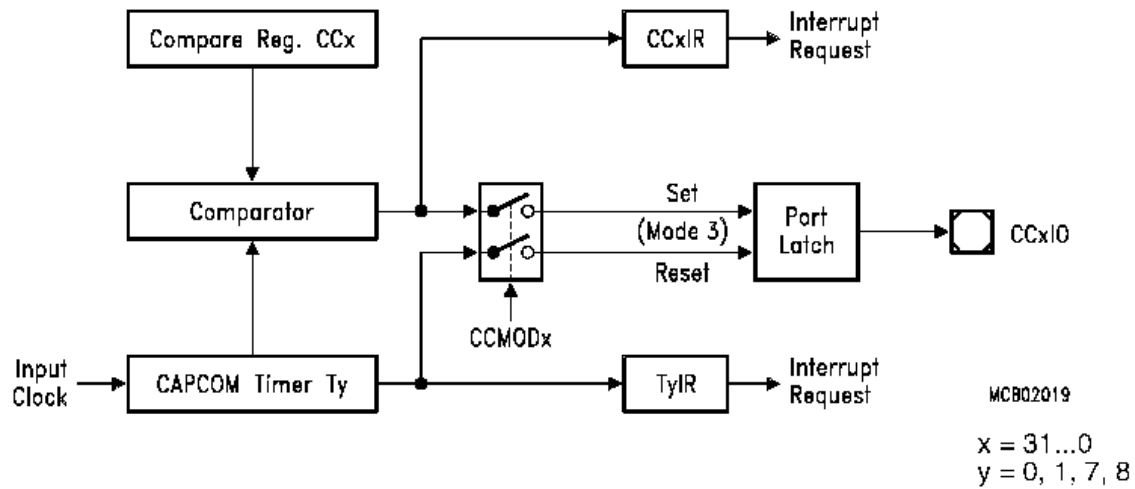


Рисунок 14-8

Блок-схема режима сравнения 2 и 3

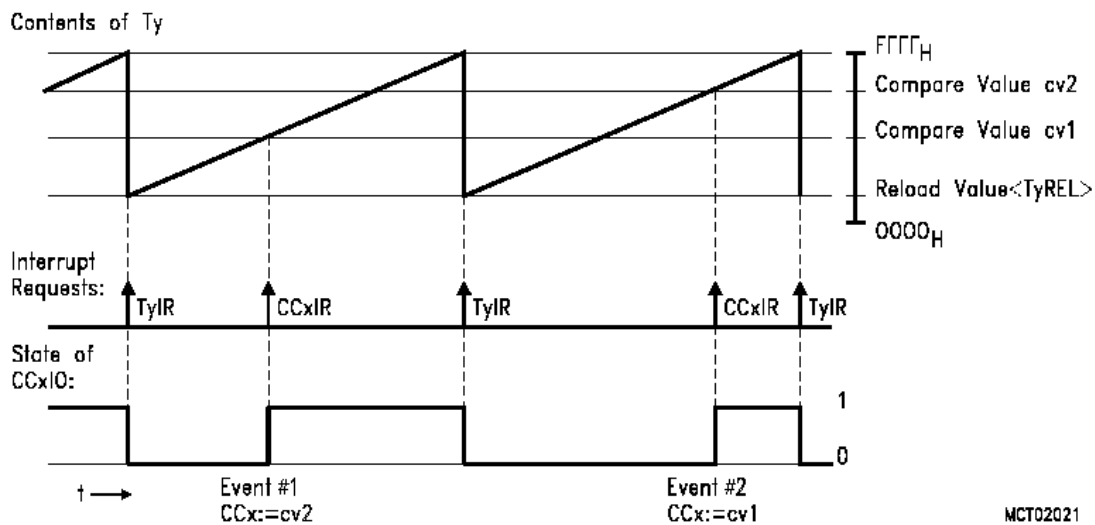


Рисунок 14-9

Пример для режимов 2 и 3

Режим сравнения 3

Режим сравнения 3 выбирается путем установки 111 в битовом поле CCMODx. В режиме 3 за один период таймера может быть обработано только одно результативное сравнение.

После первого результативного сравнения устанавливается флаг запроса на прерывание и выставляется «1» на выходе CCxIO. При переполнении таймера на выходе устанавливается низкий уровень напряжения.

Необходимо в регистре направления порта установить порт на вывод данных. В этом режиме возможно изменять значение в триггере порта в любой момент.

Примечание: В случае совершения одновременной программной записи в вывод порта и установки значения в выводе порта по результативному сравнению, более высокий уровень приоритета имеет программная запись. При этом не будет совершено изменение выходного значения по сигналу сравнения.

Для каналов 24... 27 в режиме сравнения 3 имеется возможность для создания запроса на прерывание, но отсутствует возможность для вывода сигнала.

Double-register режим сравнения

В этом режиме два регистра сравнения работают вместе, управляя одним выходом. Этот режим вызывается специальной комбинацией режимов в двух регистрах.

Для работы в этом режиме 16 регистров каждого блока CAPCOM разделены на два банка по 8 регистров в каждом. Регистры CC0... CC7 и CC16... CC23 образуют банк 1. Регистры CC8... CC15 и CC24... CC31 образуют банк 2. В этом режиме один регистр из банка 1 и один регистр из банка 2 образуют регистровую пару. Оба регистра пары привязаны к выводу регистра банка 1.

Ниже приведены возможные пары регистров:

CAPCOM1			CAPCOM2		
Пара регистров		Выводы порта	Пара регистров		Выводы порта
Банк 1	Банк 2		Банк 1	Банк 2	
CC0	CC8	CC0IO	CC16	CC24	CC16IO
CC1	CC9	CC1IO	CC17	CC25	CC17IO
CC2	CC10	CC2IO	CC18	CC26	CC18IO
CC3	CC11	CC3IO	CC19	CC27	CC19IO
CC4	CC12	CC4IO	CC20	CC28	CC20IO
CC5	CC13	CC5IO	CC21	CC29	CC21IO
CC6	CC14	CC6IO	CC22	CC30	CC22IO
CC7	CC15	CC7IO	CC23	CC31	CC23IO

Для каждой пары регистров режим сравнения double-register может быть запрограммирован отдельно. Для включения этого режима необходимо регистр из первого банка запрограммировать в режим сравнения 1, а регистр из второго банка должен быть запрограммирован в режим сравнения 0.

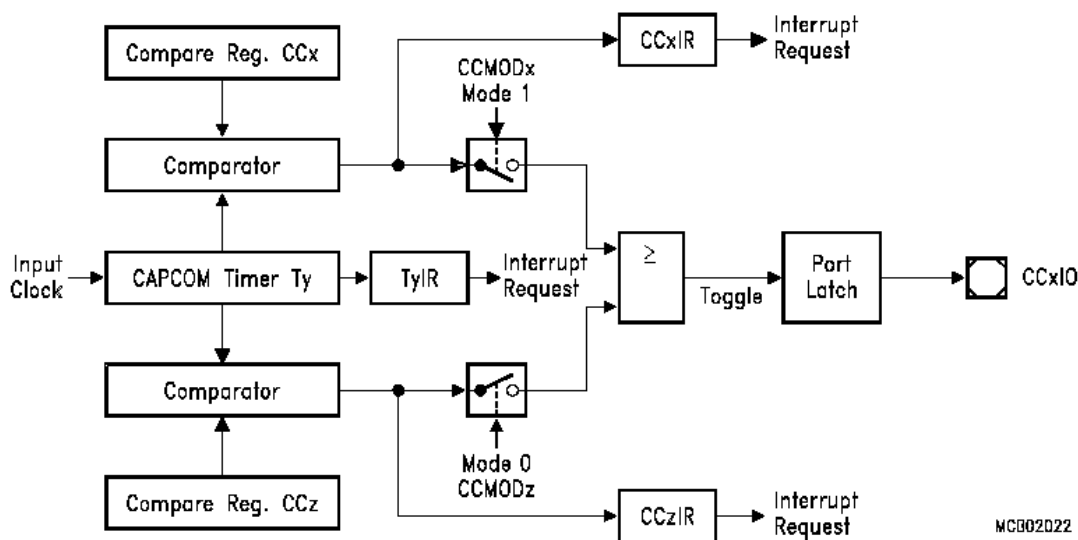
Если регистр из банка 1 отключен или запрограммирован в другой режим, регистр из другого банка будет работать в режиме сравнения 0.

Ниже под обозначением CCz подразумевается регистр из второго банка, а под CCx – регистр из первого банка.

В том случае если значение таймера достигает значения регистра из пары (CCx или CCz), выставляется флаг запроса на прерывание (CCxIO или CCzIO), а также переключается напряжение на выходе CCx связанного с регистром банка 1.

Примечание: Если одновременно было выявлено результативное сравнение в регистрах CCx и CCz одной регистровой пары, то уровень напряжения на выходе будет изменен только один раз, однако будут выставлены два запроса на прерывание, один по вектору CCxINT, другой по CCzINT.

Для того чтобы использовать выводы портов в качестве выводов CCxIO в режиме double-register, необходимо установить «1» в бите регистра управления направлением порта. В этой конфигурации выход порта имеет точно такие же характеристики как и в режиме 1.



MCB02022

x = 23...16, 7...0
y = 0, 1, 7, 8
z = 31...24, 15...8

Рисунок 14-10

Блок-схема режима сравнения double-register

В примере, приведенном ниже, оба регистра были привязаны к одному таймеру, однако имеется возможность для отдельной привязки каждого регистра пары к различным таймерам из соответствующего блока CAPCOM. В этом примере во время работы значения регистров CCx и CCz не изменялись.

Примечание: Вывод CCzIO, не используемый в режиме double-register, может быть переопределен для ввода-вывода основного назначения.

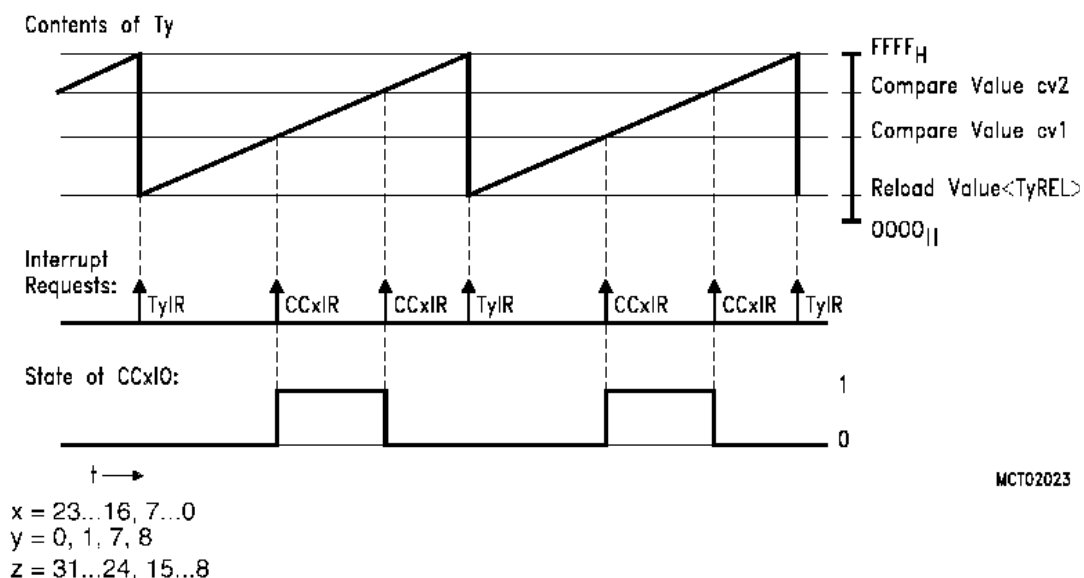


Рисунок 14-11

14.6 Прерывания при захвате и сравнении

При совершении захвата или сравнения, выставляется флаг запроса на прерывание CCxIR. Этот флаг может использоваться для входа в обычную подпрограмму прерывания, или для начала процедуры РЕС-обслуживания.

Прерывания по захвату могут использоваться в качестве запросов прерываний с дополнительной функцией сохранения времени установки запроса на прерывание.

Каждый из 32 регистров CC0... CC31 имеет регистр управления прерываниями CC0IO... CC31IO.

CAPCOM1			CAPCOM2		
Регистр	Адрес	Простр.	Регистр	Адрес	Простр.
CC0IC	FF78 _H /BC _H	SFR	CC16IC	F160 _H /B0 _H	ESFR
CC1IC	FF7A _H /BD _H	SFR	CC17IC	F162 _H /B1 _H	ESFR
CC2IC	FF7C _H /BE _H	SFR	CC18IC	F164 _H /B2 _H	ESFR
CC3IC	FF7E _H /BF _H	SFR	CC19IC	F166 _H /B3 _H	ESFR
CC4IC	FF80 _H /C0 _H	SFR	CC20IC	F168 _H /B4 _H	ESFR
CC5IC	FF82 _H /C1 _H	SFR	CC21IC	F16A _H /B5 _H	ESFR
CC6IC	FF84 _H /C2 _H	SFR	CC22IC	F16C _H /B6 _H	ESFR
CC7IC	FF86 _H /C3 _H	SFR	CC23IC	F16E _H /B7 _H	ESFR
CC8IC	FF88 _H /C4 _H	SFR	CC24IC	F170 _H /B8 _H	ESFR
CC9IC	FF8A _H /C5 _H	SFR	CC25IC	F172 _H /B9 _H	ESFR
CC10IC	FF8C _H /C6 _H	SFR	CC26IC	F174 _H /BA _H	ESFR
CC11IC	FF8E _H /C7 _H	SFR	CC27IC	F176 _H /BB _H	ESFR
CC12IC	FF90 _H /C8 _H	SFR	CC28IC	F178 _H /BC _H	ESFR
CC13IC	FF92 _H /C9 _H	SFR	CC29IC	F184 _H /C2 _H	ESFR
CC14IC	FF94 _H /CA _H	SFR	CC30IC	F18C _H /C6 _H	ESFR
CC15IC	FF96 _H /CB _H	SFR	CC31IC	F194 _H /CA _H	ESFR

15 Модуль широтно-импульсной модуляции

ШИМ-модуль С167 позволяет создавать до четырех независимых ШИМ-сигналов. Выходная частота ШИМ-сигналов может меняться в диапазоне от 4.8 Гц до 10МГц для одностороннего ШИМ. Для вустороннего ШИМ частота может меняться в диапозоне от 2.4Гц до 5МГц. Минимальное значение частоты определяется разрядностью таймера (16-разрядный) и значением делителя тактового сигнала таймера (CLK/1 или CLK/64). Максимальная частота достигается при условии, что PWM сигнал изменяется после каждого цикла таймера. В реальных приложениях максимальная частота зависит от необходимой точности выходного ШИМ-сигнала.

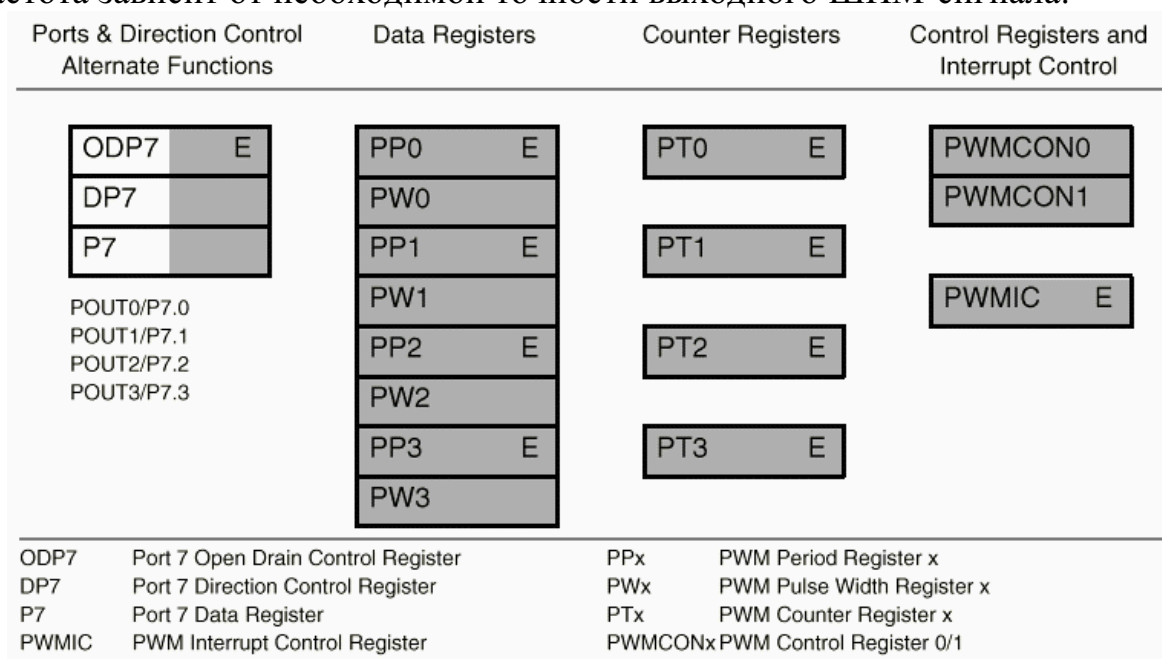


Рисунок 15-1

SFR-регистры и выводы портов ШИМ-модуля

ШИМ-модуль содержит четыре независимых ШИМ-канала. Каждый канал имеет 16-разрядный счетчик с возможностью смены направления счета, 16-разрядный регистр периода PPx с shadow latch, 16-разрядный регистр PWx с shadow latch, два компаратора и логику управления.

Работа всех четырех регистров управляется в двух регистрах управления, PWMCON0 и PWMCON1, управление прерываниями от всех каналов осуществляется в регистре PWMIC.

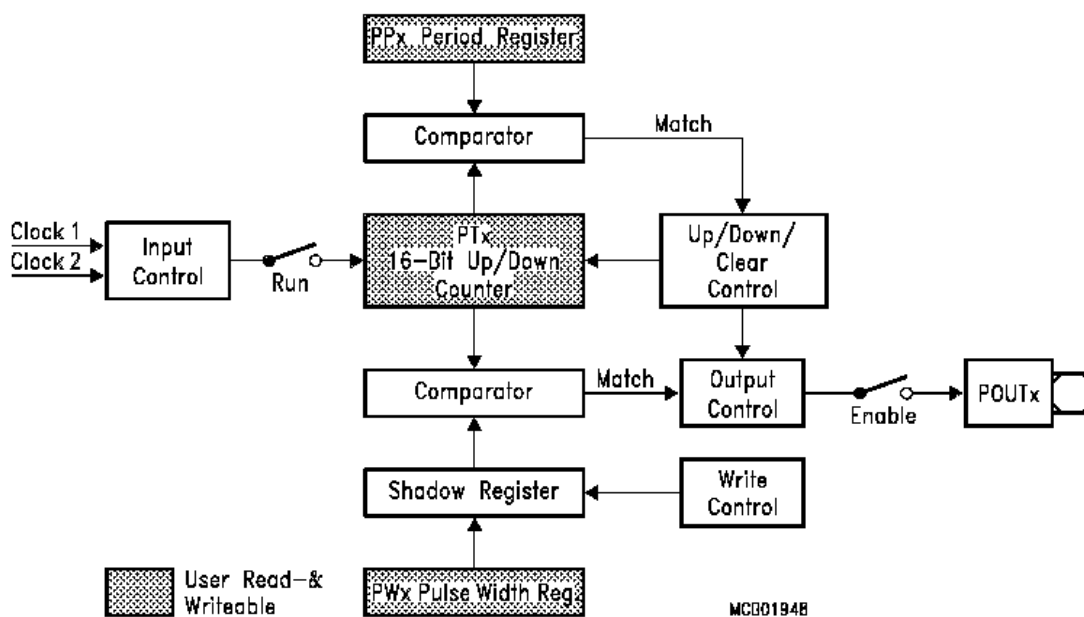


Рисунок 15-2

Блок-схема ШИМ-канала

15.1 Режимы работы

ШИМ-модуль может работать в четырех различных режимах

- Стандартный ШИМ (edge aligned ШИМ) доступен для всех четырех каналов
- Симметричный ШИМ (center aligned ШИМ) доступен для всех четырех каналов
- Режим Burst объединяет каналы 0 и 1
- Режим однократного импульса доступен для каналов 2 и 3

Примечание: Выходные сигналы ШИМ-канала и значения, записанными в выходной триггер порта, объединяются по функции XOR. После RESET в выходных триггерах порта устанавливается 0, поэтому ШИМ сигнал выводится без изменения. В случае установки в триггере порта «1» ШИМ-сигнал при выводе будет инвертироваться.

Режим 0: Стандартный ШИМ (Edge aligned ШИМ)

Режим 0 выбирается путем установки «0» в бите РМх регистра PWMCON1. В этом режиме значение таймера РТх инкрементируется, до тех пор не достигнет значения регистра периода. После этого на следующем такте счета значение таймера сбрасывается в 0000_H, и отсчет продолжается. Значение выходного сигнала будет равно «1» в том случае, если значение таймера больше или равно значения регистра ширины импульса. В момент сброса таймера выходной ШИМ-сигнал будет также установлен в «0». Период ШИМ-сигнала зависит от значения в регистре РРх+1.

Скважность выходного ШИМ-сигнала управляется посредством изменения значения в регистре ширины импульса. Это позволяет устанавливать скважность в диапазоне от 0% до 100%. В случае записи 0000_H в регистр ширины импульса, выходной сигнал всегда остается на высоком уровне напряжения, и поэтому скважность равна 100%. В случае записи в регистр ширины импульса значения более длительности периода, выходное значение всегда остается равным «0», и поэтому скважность сигнала равна 0%.

Рисунок, представленный ниже, иллюстрирует работу и форму выходного сигнала канала 0. Этот режим называется Edge Aligned ШИМ (выровненный по фронту), так как значение в регистре ширины импульса влияет только на положение положительного фронта сигнала. Отрицательный фронт всегда зафиксирован и привязан к моменту сброса таймера.

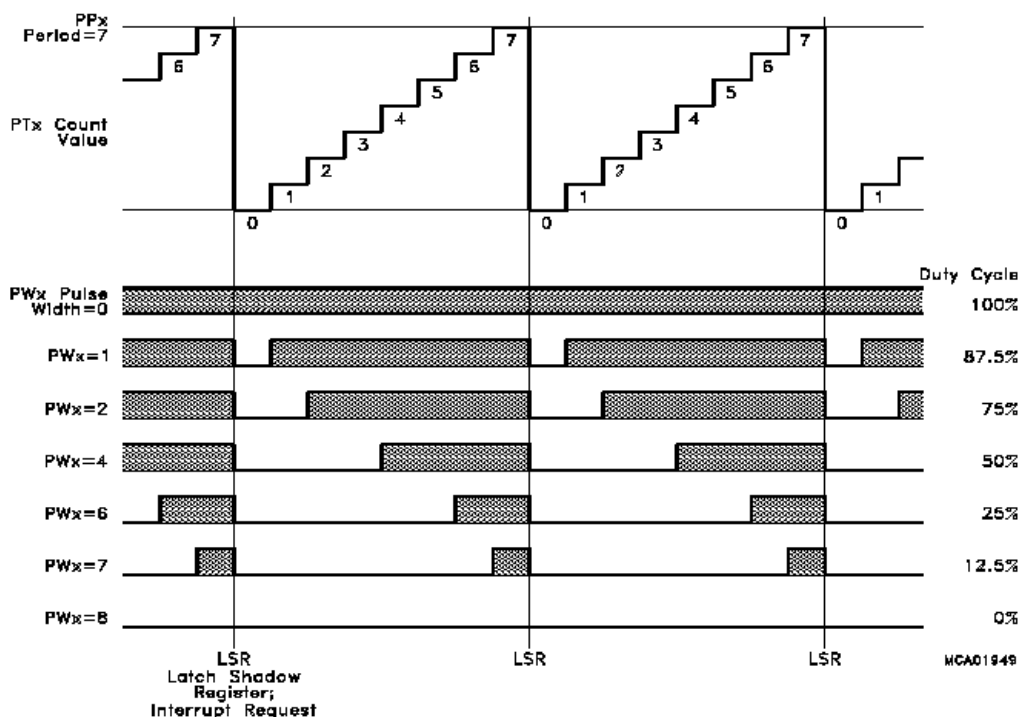


Рисунок 15-3

Работа в режиме 0

Режим 1: Симметричный ШИМ (Center Aligned ШИМ)

Режим 1 можно выбрать путем установки «1» в бите РМх регистра PWMCON1. В этом режиме таймер РТх производит отсчет в положительном направлении, до тех пор пока не достигнет значения регистра периода. На следующем такте таймера направление счета изменяется и начинается отсчет вниз, до тех пор пока не будет достигнут 0000_н. Затем цикл повторяется.

Выходной ШИМ-сигнал содержит высокий уровень напряжения, когда значение таймера больше или равно значению регистра ширины импульса. Отсюда следует, что в режиме «1» можно влиять как на расположение положительного так и на расположение отрицательного фронта сигнала.

Заметим, что в режиме «1» период ШИМ сигнала – это удвоенный период таймера:

$$PWM_PERIOD_{\text{Режим1}} = 2 \cdot ([PPx] + 1)$$

Рисунок ниже иллюстрирует форму выходного сигнала ШИМ-канала в режиме 1 при различных значениях в регистре ширины импульса. Этот режим называется Center Aligned ШИМ, так как значение в регистре ширины импульса оказывает симметричное влияние на фронты выходного сигнала.

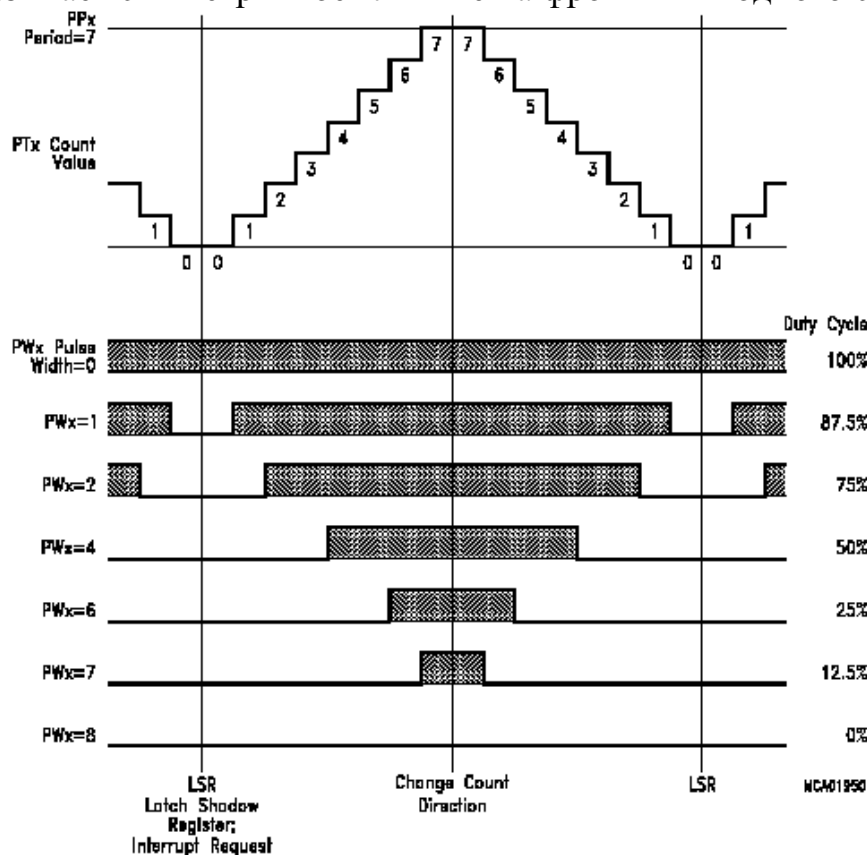


Рисунок 15-4

Работа в режиме 0

Режим burst

Этот режим можно выбрать путем установки «1» в бите PB01 регистра PWMCON1. В этом режиме сигналы каналов 0 и 1 объединяются на выводе канала 0. Выход канала 0 представляет из себя объединенные по функции AND канал 0 и канал 1. При этом может дополнительно использоваться выход канала 1.

Каждый из двух каналов может работать в режиме 0 или в режиме 1.

Функции XOR накладывается на выходной сигнал и на значение в триггере порта после сложения по функции AND сигналов каналов 0 и 1.

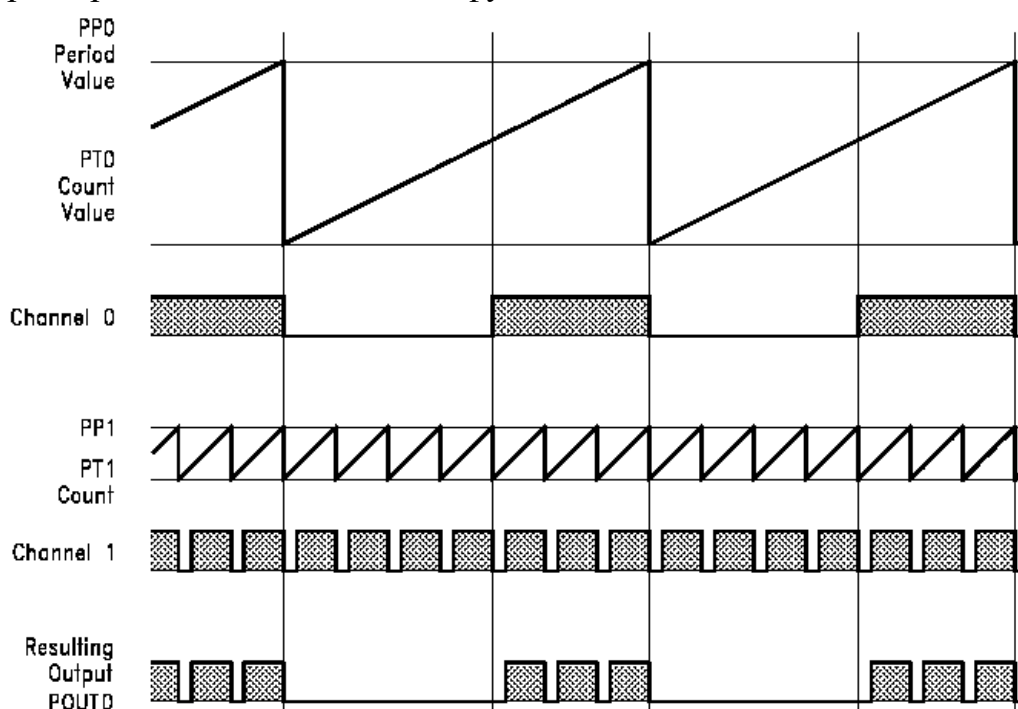


Рисунок 15-5

Работа в режиме Burst

Режим однократного импульса

Этот режим можно выбрать путем установки «1» в бите PSx регистра PWMCON1. Этот режим доступен для ШИМ-каналов 2 и 3.

В этом режиме таймер PTx начинает работать по сигналу программы, и после этого начинает отсчет в положительном направлении, до тех пор пока не будет достигнуто значение регистра периода. На следующий такт значение таймера будет сброшено, и таймер будет аппаратно остановлен, т.е. значение бита PTRx будет сброшено. Выходной сигнал ШИМ-канала содержит находится на высоком уровне напряжения, когда содержимое таймера больше или равно содержимого регистра ширины импульса. Сигнал возвращается на низкий уровень напряжения после сброса значения таймера. Для создания нескольких импульсов необходимо каждый раз устанавливать «1» в бите PTRx.

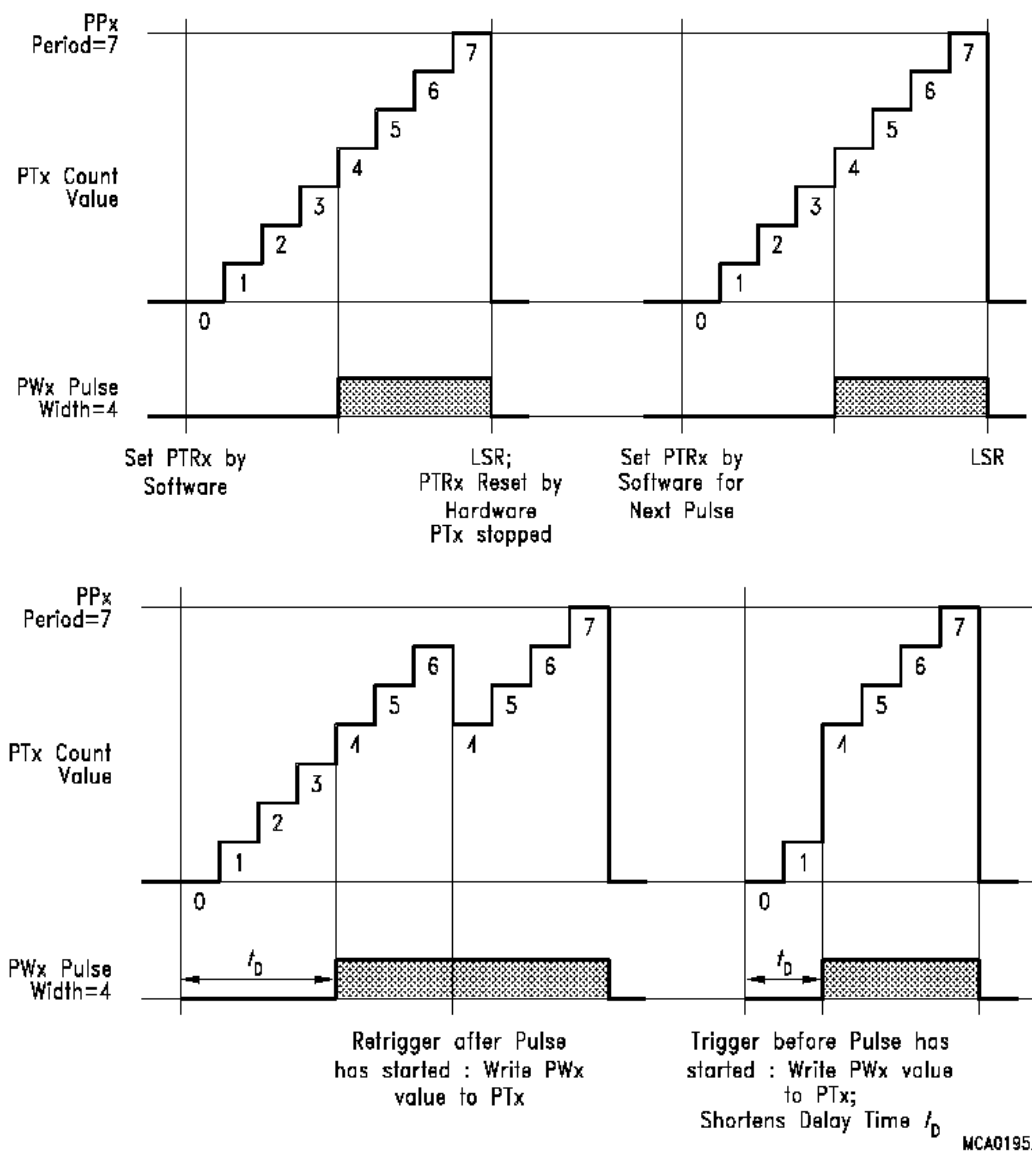


Рисунок 15-6

Работа в режиме однократного импульса

После старта счета таймера (т.е. PTRx = 1) можно программно изменять длину выходного импульса. Запись нового значения в таймер может изменить месторасположение положительного и отрицательного фронта выходного сигнала. Многократная запись значения в таймер возможна только в том случае, когда таймер работает, т.е. после старта импульса и до остановки таймера.

В случае загрузки в регистр таймера значения shadow регистра PRx, данный ШИМ-импульс будет оборван при следующем такте (при этом значение счетчика будет сброшено и сам счетчик остановлен).

Изменяя значения периода (PRx), начальное значение таймера (PTx) и значение ширины импульса (PWx), можно в широком диапазоне варьировать ширину импульса и задержку импульса.

15.2 Регистры модуля ШИМ

Модуль ШИМ управляется с помощью двух наборов регистров. Форма импульса выбирается в регистрах PTx (таймер), PRx (период) и PWx (ширина импульса). Два общих регистра управляют режимами работы и основными функциями модуля ШИМ (PWMCON0 и PWMCON1).

Счетчик PTx с управляемым направлением счета

Счетчик каждого ШИМ-канала работает от тактового сигнала ЦПУ. При этом частота переключения счетчика либо равна частоте ЦПУ, либо составляет 1/64 от частоты ЦПУ. Делитель частоты выбирается в бите PTIx регистра PWMCON0. В зависимости от значения бита PTRx счетчик считает либо в положительном направлении, либо в отрицательном направлении. Включение и выключение счета управляется в бите PTRx. Управляя битом PTRx, можно до некоторой степени управлять выходным ШИМ-сигналом.

Входная частота и режим	8-разрядный ШИМ	10-разрядный ШИМ	12-разрядный ШИМ	14-разрядный ШИМ	16-разрядный ШИМ
f _{CPU} Режим 0	78.13 кГц	19.53 кГц	4.88 кГц	1.22 кГц	305 Гц
f _{CPU} /64 Режим 0	1.22 кГц	305 Гц	76.3 Гц	19.1 Гц	4.77 Гц
f _{CPU} Режим 1	39.1 кГц	9.77 кГц	2.44 кГц	610 Гц	152.6 Гц
f _{CPU} /64 Режим 1	610 Гц	156.2 Гц	38.15 Гц	9.54 Гц	2.4 Гц

Регистры периода PPx

Значение 16-разрядного регистра периода PPx определяет длительность ШИМ-периода, т.е. частоту ШИМ-сигнала. Этот регистр имеет буфер в виде shadow-регистра. В начале каждого нового ШИМ-цикла или после записи нового значения в PPx (когда счетчик остановлен), в shadow-регистр загружается значение регистра PPx. ЦПУ получает доступ к регистру PPx, в то время когда производится сравнение содержимого shadow-регистра с содержимым счетчика PTx. В тот момент, когда значение в счетчике сравнивается с значением PPx shadow-регистра, счетчик, в зависимости от режима, либо сбрасывается в нуль, либо начинает считать в отрицательном направлении.

Регистры ширины импульса PWx

Значение этого регистра предопределяет ширину ШИМ-импульса. Этот регистр имеет буфер в виде shadow-регистра. ЦПУ получает доступ к содержимому регистра PWx в то время, когда производится аппаратное сравнение содержимого shadow-регистра с содержимым счетчика PTx. В начале каждого нового ШИМ-цикла или после записи нового значения в PWx, в shadow-регистр загружается значение регистра PWx.

Когда значение счетчика становится больше или сравнивается со значением shadow-регистра, ШИМ-сигнал устанавливается в «1».

Регистр	Адрес	Рег. прост.	Регистр	Адрес	Рег. прост.
PW0	FE30 _H /18 _H	SFR	PT0	F030 _H /18 _H	ESFR
PW1	FE32 _H /19 _H	SFR	PT1	F032 _H /19 _H	ESFR
PW2	FE34 _H /1A _H	SFR	PT2	F034 _H /1A _H	ESFR
PW3	FE36 _H /1B _H	SFR	PT3	F036 _H /1B _H	ESFR
Примечание: Эти регистры не адресуемы побитно			PP0	F038 _H /1C _H	ESFR
			PP1	F03A _H /1D _H	ESFR
			PP2	F03C _H /1E _H	ESFR
			PP3	F03E _H /1F _H	ESFR

Регистр управления ШИМ PWMCON0

Регистр PWMCON0 управляет работой таймера каждого ШИМ-канала. Также в регистре PWMCON0 осуществляется управление прерываниями. Биты управления сгруппированы вместе, что позволяет запускать или останавливать все четыре канала одновременно с помощью одной команды для поля битов.

SFR

PWMCON0 (FF30_H/98_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIR3	PIR2	PIR1	PIR0	PIE3	PIE2	PIE1	PIE0	PTI3	PTI2	PTI1	PTI0	PTR3	PTR2	PTR1	PTR0
rw	Rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
PTRx	Бит управления работой ШИМ-счетчика x 0: Счетчик РТх отключен от тактового сигнала 1: Счетчик РТх работает
PTIx	Выбор делителя частоты входного тактового сигнала 0: Счетчик работает с частотой ЦПУ 1: Счетчик работает с 1/64 от частоты ЦПУ
PIEx	Флаг разрешения прерываний от ШИМ-канала 0: Не разрешены запросы от канала x 1: Канал x выставляет запросы (сбрасывается программными способами)
PIRx	Флаг запроса на прерывание ШИМ-канала x 0: отсутствует запрос от канала x 1: Канал x выставил запрос на прерывание (сбрасывается программными способами)

Регистр управления ШИМ PWMCON1

Этот регистр управляет режимами работы и выходами четырех ШИМ-каналов. Базовый режим работы каждого канала (режим 0 или режим 1) выбирается при помощи установки значения в бите режима PMx. Режим burst (канал 0 и 1) и режим однократного импульса (канал 2 или 3) выбирается при помощи отдельных битов управления. Разрешение на вывод сигнала ШИМ-канала может быть дано для каждого канала при помощи установки «1» в бите PENx. В том случае когда вывод сигнала ШИМ-канала запрещен, данный вывод порта может использоваться для ввода-вывода основного назначения. При этом ШИМ-канал продолжает выставлять запросы на прерывание.

SFR

PWMCON1 (FF32_H/99_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS3	PS2	-	PB0 1	-	-	-	-	PM3	PM2	PM1	PM0	PEN3	PEN2	PEN1	PEN0
rw	rw	-	rw	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Бит	Функция
PENx	Бит разрешения вывода сигнала канала x 0: Вывод сигнала канала x запрещен 1: Вывод сигнала канала x разрешен
PMx	Бит управления режимом ШИМ-канала x 0: Канал x работает в режиме 0 1: Канал x работает в режиме 1
PB01	Бит управления режимом Burst для ШИМ-каналов 0/1 0: Каналы 0 и 1 работают независимо в обычных режимах 1: Выходы каналов 0 и 1 объединены по функции AND, и результат выводится через вывод канала 0
PSx	Бит управления режимом однократного импульса канала x 0: Канал x работает в обычном режиме 1: Канал x работает в режиме однократного импульса

15.3 Создание запросов на прерывание

Каждый из четырех каналов ШИМ-модуля может генерировать собственный локальный запрос на прерывание. Каждый из этих локальных запросов может активировать глобальное прерывание от ШИМ-контроллера. Глобальное прерывание модуля воспринимается ЦПУ как обычный запрос на прерывание. Управление прерыванием от ШИМ-модуля осуществляется в регистре PWMIC. Подпрограмма обслуживания прерываний может самостоятельно определить канал, вызвавшее прерывание, при помощи чтения флагов локальных запросов на прерывание канала PIRx регистра PWMCON0. Флаг запроса на прерывание PIRx устанавливается в начале нового периода ШИМ-сигнала. Это указывает на то, что регистры PRx и PWx готовы к получению новых значений. Также в случае установки «1» в бите PIEx, при выставлении локального флага будет выставлен глобальный флаг запроса на прерывание PWMIR регистра PWMIC.

Примечание: Флаги запросов на прерывание от каналов PIRx после входа в подпрограмму обслуживания прерывания не сбрасываются автоматически, поэтому их необходимо сбрасывать программными методами. Глобальный флаг запроса на прерывание PWMIR сбрасывается автоматически после начала выполнения подпрограммы обслуживания, несмотря на то сколько было выставлено флагов локальных запросов от каналов. Однако в процессе выполнения подпрограммы обслуживания может быть выставлен новый флаг запроса на прерывание.

PWMIC (F17E/BF_H) – полное описание можно найти в разделе посвященном прерываниям.

15.4 Выходные ШИМ-сигналы

Выходные сигналы четырех ШИМ-каналов (POUT0... POUT3) являются альтернативной функцией вывода порта 7 (P7.0 ... P7.3). Вывод сигнала каждого ШИМ-канала можно разрешать или запрещать с помощью бита PENx регистра PWMCONx.

ШИМ-сигналы и значения в выходных триггерах порта объединены по функции XOR. В том случае, если P7.x = 1 выходной ШИМ-сигнал будет проинвертирован.

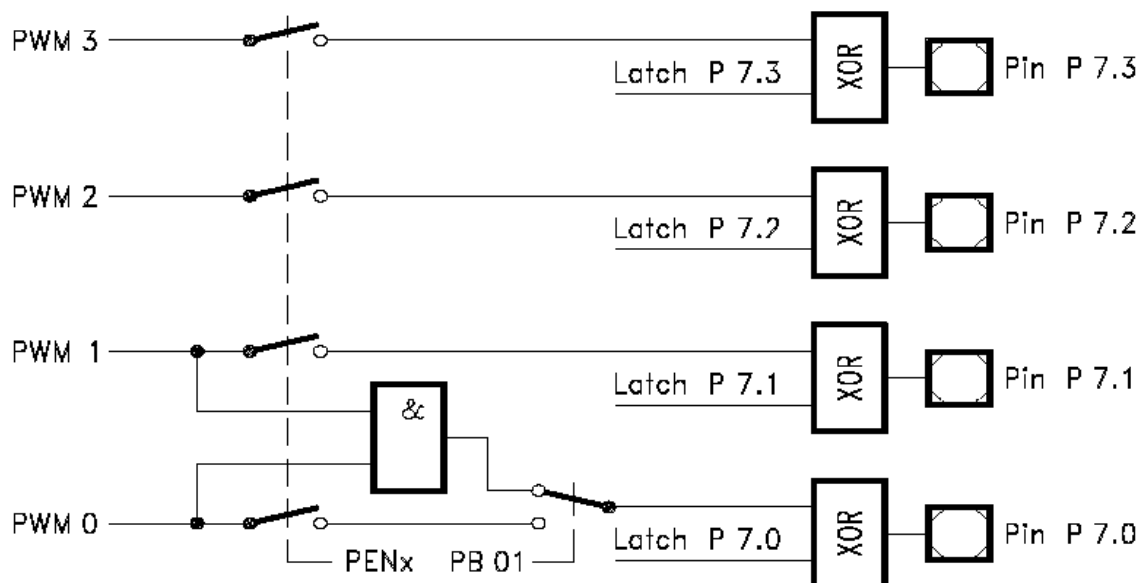


Рисунок 15-7

Создание выходных ШИМ-сигналов

Примечание: Использование режима с открытым коллектором для вывода ШИМ-сигналов позволяет объединять несколько ШИМ-каналов по функции wired-AND. При этом для любого числа ШИМ-каналов обеспечивается работа в режиме burst.

Программное управление сигналами ШИМ-каналов

В некоторых режимах (системная ошибка или процедура экстренного сохранения) необходимо программно изменять значения напряжений на выходах ШИМ-каналов.

При сбросе значения бита PTRx таймер канала x останавливается, и выходной сигнал остается на текущем уровне напряжения. Изменяя значения в регистрах RWx или PTx можно менять значение на выходе ШИМ-канала.

Также путем установки значений в триггере порта 7 можно инвертировать сигнал на выходе ШИМ-канала.

Установка «0» в бите разрешения канала PENx позволяет переключать вывод порта на его выходной триггер.

Примечание: Прежде чем вносить программные изменения в уровень напряжения на выходе, необходимо остановить таймер.

16 Модуль АЦП

Архитектура С167 имеет внутренний 10-разрядный АЦП и схему захвата и считывания. Мультиплексор обеспечивает выбор одного из 16 входных аналоговых каналов (альтернативная функция порта 5) либо под программным управлением (режим фиксированных каналов), либо автоматически (режим автоматического сканирования). Автоматическая самостоятельная калибровка позволяет модулю АЦП приспосабливаться к изменениям температуры или к различным вариантам работы.

Для удовлетворения высоким требованиям встроенных систем управления АЦП поддерживает следующие режимы преобразования:

- **Однократное преобразование для фиксированного канала** обеспечивает только один результат в выбранном канале
- **Многократное преобразование для фиксированного канала**
- **Однократное преобразование с автоматическим сканированием** обеспечивает один результат от каждого канала из выбранной группы
- **Ожидание режима чтения ADDAT** – автоматическое начало преобразования, после прочтения предыдущего результата
- **Режим инъекции канала** – вставка канала в группу каналов (автоматическое сканирование)

Набор SFR-регистров и входов каналов обеспечивает управление и результат АЦП.

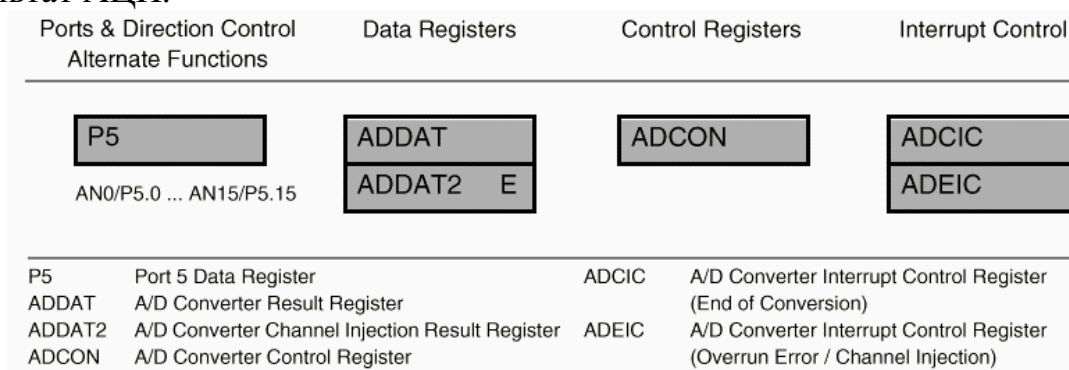


Рисунок 16-1

SFR-регистры и выходы портов для модуля АЦП

Внешние аналоговые опорные сигналы V_{AREF} и V_{AGND} зафиксированы. Независимое питание АЦП уменьшает помехи от других цифровых сигналов.

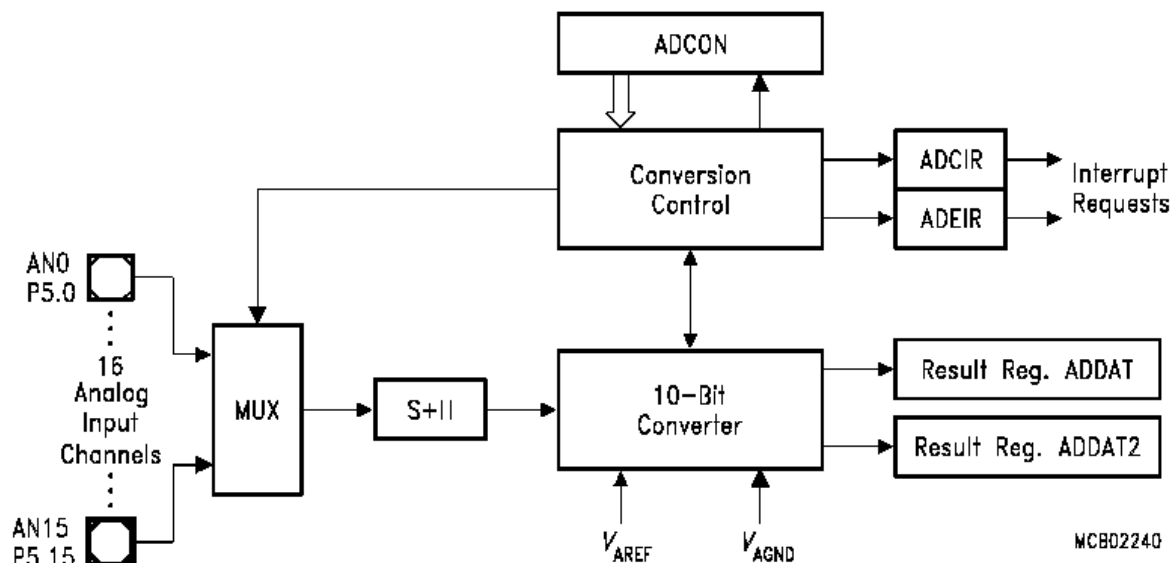


Рисунок 16-2

Блок-схема АЦП

16.1 Работа и выбор режима

Аналоговые входные каналы AN0... AN15 являются альтернативной функцией порта 5. Порт 5 является 16-разрядным **только** входным портом. Линии порта могут использоваться для ввода как аналоговых, так и цифровых сигналов. Не требуется никаких специальных действий для настройки порта 5 на ввод аналоговых сигналов.

Функции АЦП управляются в побитно адресуемом регистре управления АЦП ADCON.

SFR
ADCON (FFA0_H/D0_H) **Значение после RESET: 0000_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCTC		ADSTC		AD CRQ	AD CIN	AD WR	AD BSY	ADST	-	ADM		ADCH			
rw		rw		rw	rw	rw	r	rw	-	rw		rw			

Бит	Функция
ADCH	Выбор аналогового канала
ADM	Выбор режима АЦП 0 0: Однократное преобразование фиксированного канала 0 1: Многократное преобразование фиксированного канала 1 0: Однократное преобразование с автоматическим сканированием 1 1: Многократное преобразование с автоматическим сканированием
ADST	Стартовый бит АЦП
ADBSY	Флаг занятости АЦП 1: Преобразование в действии
ADWR	Управление ожидания чтения АЦП
ADCIN	Разрешение инъекции канала АЦП
ADCRQ	Флаг запроса на инъекцию канала АЦП
ADSTC	Управление временем сэмплирования АЦП ^{*)}
ADCTC	Управление временем преобразования АЦП ^{*)}

^{*)}ADSTC и ADCTC управляют временем преобразования. Подробнее в разделе «Управление временем преобразования».

В битовом поле ADCH указывается номер входного аналогового канал, значение которого будет преобразовано сразу после включения преобразователя (первый канал в последовательном преобразовании в режиме автосканирования). В битовом поле ADM выбирается режим работы АЦП. Преобразование (или последовательность преобразований) начинается после установки «1» в бите ADST. Установка «0» в этом бите позволяет остановить АЦП.

Флаг занятости ADBSY (только для чтения) устанавливается во время совершения преобразования.

Результат преобразования записывается в регистре ADDAT, а в случае инъектированного преобразования – в регистре ADDAT2.

Примечание: Данные в битовом поле CHNR регистра ADC указывают на номер канала текущего преобразования. Битовое поле CHNR

регистра ADDAT2 используется ЦПУ для указания инжектированного канала.

SFR

ADDAT (FEA0_H/50_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNR				-	-	ADRES									
rw				-	-	rw									

SFR

ADDAT2 (F0A0_H/50_H)

Значение после RESET: 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHNR				-	-	ADRES									
rw				-	-	rw									

Бит	Функция
ADRES	10-разрядный результат преобразования
CHNR	Номер канала (4 бита, идентифицирующих преобразуемый канал)

Преобразование начинается с установки «1» в бите ADST. После этого выставляется флаг занятости ADBSY, выбирается канал и начинается процесс преобразования. Напряжение на входе канала будет захвачено внутренними средствами во время преобразования. После завершения преобразования результат вместе с номером канала сохраняется в регистре ADDAT. Одновременно с этим выставляется флаг запроса на прерывание.

Если во время работы АЦП программно сбросить значение бита ADST, то работа преобразователя прекратится после завершения текущего преобразования (в режиме фиксированного канала) или после завершения последовательности преобразований (в режиме автосканирования).

Установка «1» в бите ADST во время совершения преобразования прервет текущее преобразование и начнет новое с параметрами predeterminedенными в регистре ADCON.

Во время совершения преобразования возможно изменить значения поля выбора режима ADM и значения поля выбора канала ADCH. Изменения вступят в силу после завершения текущего преобразования или текущей последовательности преобразований.

Режимы преобразования фиксированных каналов

Этот режим можно выбрать путем установки 00 (однократное преобразование) или 01 (многократное преобразование) в битовом поле ADM регистра ADCON. Преобразование начинается после установки «1» в бите ADST. После этого выставляется флаг занятости ADBSY и начинается преобразование напряжения канала, указанного в битовом поле ADCH. После завершения преобразования выставляется флаг запроса на прерывание.

В режиме однократного преобразования преобразователь автоматически останавливается после завершения преобразования, при этом сбрасываются биты ADBSY и ADST.

В режиме многократных преобразований каждое новое преобразование начинается автоматически для канала, указанного в битовом поле ADCH. После каждого преобразования выставляется флаг запроса на прерывание ADCIR.

В случае программной установки «0» в бите ADST во время совершения преобразования, преобразователь закончит текущее преобразование, затем преобразователь будет остановлен и будет сброшен бит ADBSY.

Режимы преобразования с автосканированием

Эти режимы можно выбрать путем установки 10 (однократное преобразование) или 11 (многократное преобразование) в битовом поле ADM регистра ADCON. В режиме автосканирования преобразователь без каких-либо программных изменений номера канала автоматически последовательно преобразует напряжения на входах нескольких каналов, начиная с канала, указанного в битовом поле ADCH и заканчивая каналом 0.

Преобразование начинается после установки «1» в бите ADST. После этого выставляется флаг занятости ADBSY и производится преобразование напряжения канала, указанного в битовом поле ADCH. После завершения преобразования выставляется флаг запроса на прерывание, и автоматически начинается следующее преобразование для следующего канала (номер которого на единицу меньше). После завершения каждого преобразования выставляется флаг запроса на прерывание ADCIR. После завершения преобразования в канале 0 текущая последовательность считается завершенной.

В режиме однократного преобразования преобразователь автоматически останавливается и сбрасывает значения битов ADBSY и ADST.

В режиме многократного преобразования преобразователь автоматически начинает новую последовательность с канала, указанного в битовом поле ADCH.

В случае программного сброса значения бита ADST во время совершения преобразования, преобразователь будет остановлен и будет сброшено значение бита ADBSY после завершения текущей последовательности преобразований.

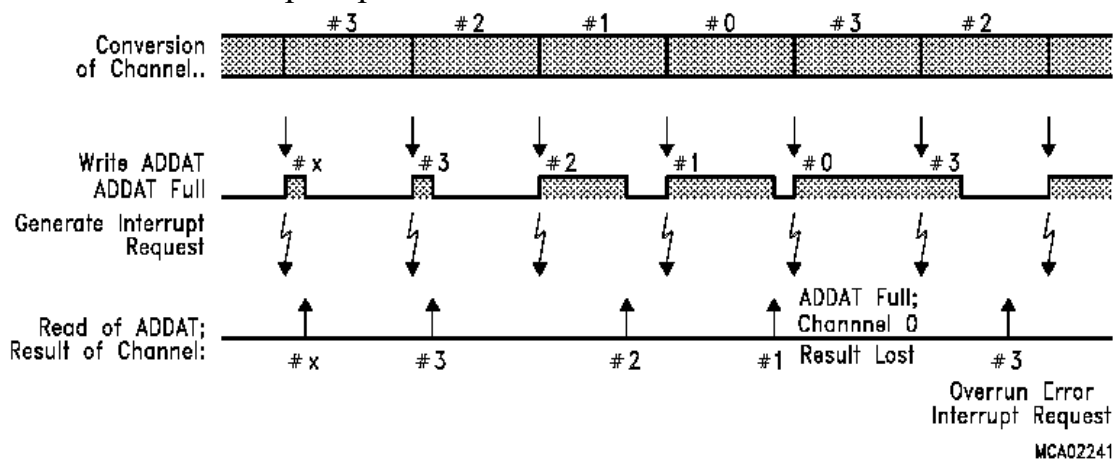


Рисунок 16-3

Пример работы в режиме автосканирования

Режим ожидания чтения ADDAT

Если значение предыдущего результата АЦП не было прочитано из регистра ADDAT до завершения нового преобразования, то оно будет потеряно. При этом будет выставлен флаг запроса на прерывание overrun ADEIR.

Для избежания прерывания по ошибке и потерь результатов преобразования при работе в режиме многократных преобразований, необходимо переключить АЦП в режим ожидания чтения ADDAT. Этот режим можно вызвать с помощью установки «1» в бите ADWR регистра ADCON.

Когда значение ADDAT не прочтено до окончания следующего преобразования, результат нового преобразования будет сохранен во временном буфере, и дальнейшие преобразования не будут начаты, при этом значения ADST и ADBSY остаются неизменными, и запрос на прерывание по окончанию преобразования не генерируется. После чтения старого значения из ADDAT содержимое временного буфера переносится в ADDAT, при этом генерируется запрос на прерывание по окончанию преобразования ADCIR, и начинается следующее преобразование. Этот механизм работает как в режиме однократного преобразования, так и в режиме многократного преобразования.

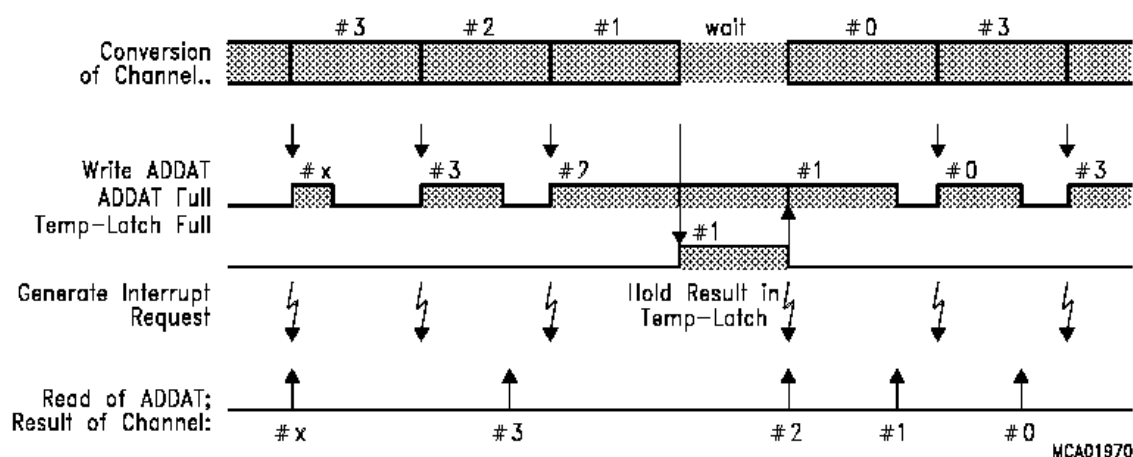


Рисунок 16-4

Пример работы в режиме ожидания чтения

Режим инъекции канала

Режим инъекции канала позволяет преобразовывать напряжение необходимого канала (когда АЦП работает в многократном режиме или в режиме автосканирования) без изменения текущего режима работы. После завершения преобразования напряжения этого канала АЦП продолжает работу в обычном режиме.

Режим инъекции канала включается при установке «1» в бите ADCIN регистра ADCON. При работе в этом режиме необходимо установить режим ожидания чтения ADDAT (ADWR=1). Преобразуемый канал выбирается в битовом поле CHNR регистра ADDAT2.

Примечание: При совершении преобразования напряжения инжектированного канала АЦП не изменяет значения битов CHNR регистра ADDAT2. Результат записывается в поле ADRES регистра ADDAT2. Поскольку номер канала CHNR регистра ADDAT2 не записывается в буфер, никогда не следует изменять его значение во время преобразования напряжения инжектированного канала. В ином случае входной мультиплексор переключится на новый канал.

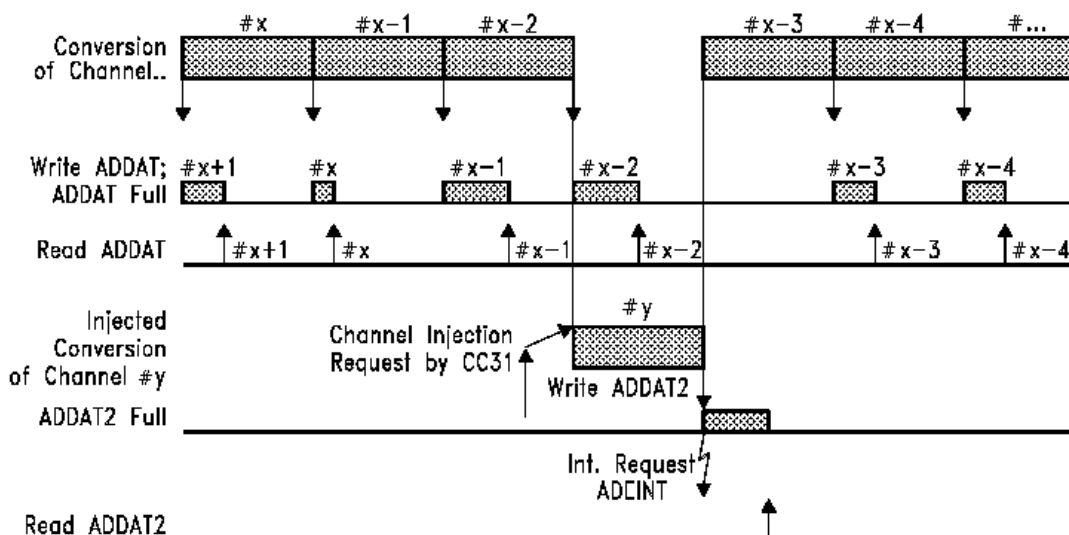


Рисунок 16-5

Пример работы в режиме инъекции канала

Инжекция канала может быть проведена двумя способами:

- Программная установка «1» в бите запроса инъекции канала ADCRQ

- По событию сравнения или захвата в регистре захвата и сравнения CC31 блока CAPCOM2. При этом также устанавливается «1» в бите ADCRQ.

Второй способ позволяет производить АЦП в определенный момент времени, при достижении счетчиком блока CAPCOM2 определенного значения или при захвате значения регистра CC31. Эта функция также позволяет записывать момент времени, при котором имело место АЦП.

Примечание: В бите запроса на инъекцию канала ADCRQ будет устанавливаться «1» при любом запросе на прерывание от канала CC31 блока CAPCOM2. При этом не имеет значения включен режим инъекции или нет. Рекомендуется всегда очищать ADCRQ прежде чем включать режим инъекции.

Примечание: Во время инжектированного преобразования запрос на инъекцию другого канала не может быть воспринят. Флаг запроса ADCRQ остается выставленным до тех пор, пока результат инжектированного преобразования не будет записан в регистр ADDAT2.

Примечание: В том случае, когда перед инъекцией канала преобразователь находился в режиме покоя, и одновременно с преобразованием напряжения инжектированного канала поступает программный сигнал начала нормального преобразования, инжектированное преобразование будет прервано и преобразователь начнет работу в выбранном режиме. С помощью проверки значения бита ожидания ADBSY перед началом нового преобразования можно предотвратить обрыв процесса.

После завершения текущего преобразования будет начато инжектированное преобразование выбранного канала. После завершения преобразования результат записывается в регистр ADDAT2, и выставляется флаг запроса на прерывание по завершению преобразования в инжектированном канале ADEIR.

Примечание: Если при этом используется режим ожидания чтения ADDAT, и временный буфер заполнен, то следующее преобразование (в том числе и из инжектированного канала) будет поставлено в режим ожидания. Временный буфер может хранить данные как для ADDAT, так и для ADDAT2.

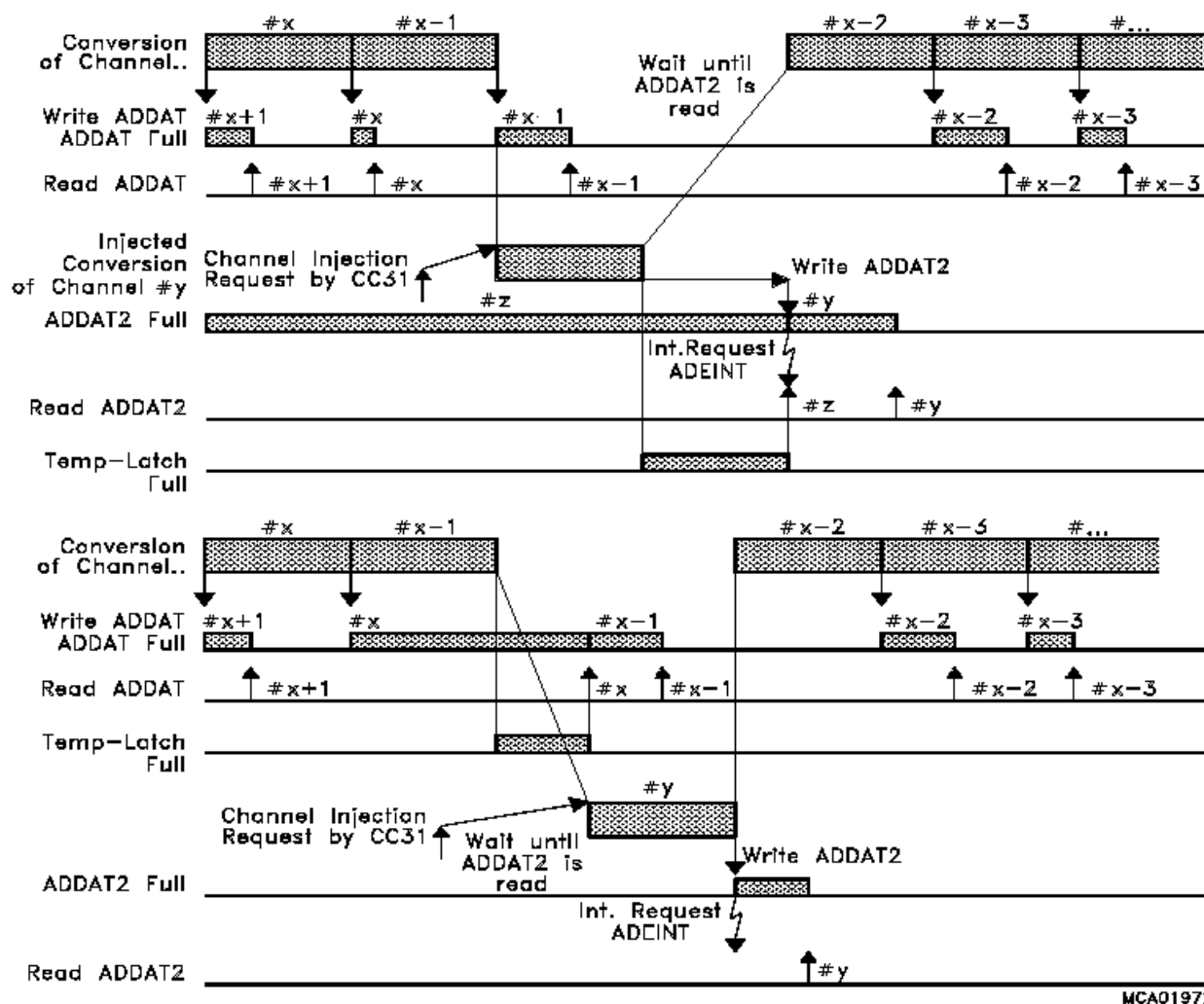


Рисунок 16-6

Пример работы в режиме с инжектированными каналами в ожидании чтения

16.2 Управление временем преобразования

В начале преобразования напряжение на аналоговом входе загружается в преобразователь. Это процесс называется сэмплированием. Затем это напряжение в течении 10 шагов преобразуется в цифровое напряжение, отсюда следует что АЦП имеет 10-разрядный результат преобразования. Следующие четыре шага используются для внутренней калибровки модуля АЦП. Во время этих 14 шагов внутренние конденсаторы постоянно заряжаются и разряжаются через вывод V_{AREF} .

Величина тока, вытекающего из источника преобразуемого напряжения для сэмплирования и перезарядки, зависит от времени, которое занимает каждый шаг, так как напряжение на конденсаторах должно достигать необходимого значения за ограниченный отрезок времени. Однако максимальный ток источника преобразуемого сигнала зависит от внутреннего сопротивления источника преобразуемого сигнала.

Величина времени, занимаемое двумя различными действиями при преобразовании (сэмплирование или оцифровка) может быть программно изменено относительно частоты ЦПУ. Это позволяет настраивать АЦП к особенностям системы:

Быстрое преобразование: достигается при программировании таймеров АЦП на максимальную скорость. При этом необходимо, чтобы внутреннее сопротивление источника сигнала преобразования было минимальным.

При высоком внутреннем сопротивлении: необходимо запрограммировать таймеры АЦП на минимальную скорость. При этом можно использовать источники сигнала с высоким внутренним сопротивлением. В этом случае время преобразования велико.

Величина времени преобразования программируется в старших четырех битах регистра ADCON. Базовая частота преобразования выбирается в битовом поле ADCTC (управление временем преобразования). Эта величина используется для тактирования 14 шагов преобразования. Величина времени сэмплирования выбирается в битовом поле ADSTC. Ниже в таблице представлены возможные комбинации. Временные характеристики зависят от работы блока TCL: $f_{CPU} = \frac{1}{2} TCL$.

ADCTC	Величина тактового сигнала оцифровки t_{CC}	ADSTC	Величина тактового сигнала сэмплирования t_{SC}
0 0	$TCL * 24$	0 0	t_{CC}
0 1	Зарезервировано	0 1	$t_{CC} * 2$
1 0	$TCL * 96$	1 0	$t_{CC} * 4$
1 1	$TCL * 48$	1 1	$t_{CC} * 8$

Полное время преобразования занимает $14 \cdot t_{CC} + 2 \cdot t_{SC} + 4TCL$ (9.7мкс при частоте ЦПУ 20МГц). В эту величину входит время сэмплирования, время оцифровки и время передачи результата в регистр.

16.3 Управление прерываниями АЦП

В конце каждого преобразования выставляется флаг запроса на прерывание ADCIR регистра ADCIC. Этот флаг может использоваться для входа в стандартное прерывание по вектору ADCINT, или может использоваться для входа в PEC-обслуживание. Эти действия необходимы для записи результата преобразования во внутренней RAM.

Флаг запроса на прерывание ADEIR регистра ADEIC будет выставлен в том случае, если результат предыдущего преобразования не будет сохранен в памяти до окончания нового преобразования (прерывание по ошибке в стандартном режиме), либо если результат преобразования был сохранен в регистре ADDAT2 (прерывание по окончанию инжектированного преобразования).

ADCIC (FF98_H/CC_H)

ADEIC (FF9A_H/CD_H)

Подробное описание этих регистров в разделе, посвященном прерываниям.

17 Системный RESET

Функция внутреннего системного RESET обеспечивает инициализацию C167 в предустановленном режиме, и вызывается либо подачей сигнала аппаратного RESET на вход \overline{RSTIN} , либо при выполнении команды SRST, либо при переполнении сторожевого таймера.

Если происходит хотя бы одно из этих событий, микроконтроллер сбрасывается и устанавливается в предопределенное состояние. Когда RESET инициируется, незаконченные процессы прерываются и завершается текущий цикл внутреннего доступа. В начале RESET будет прерван цикл внешней шины, за исключением RESET по переполнению сторожевого таймера. После этого драйвера выводов шины и портов ввода-вывода будут переведены в состояние высокого сопротивления. В зависимости от типа RESET может иметь место активирование сигнала \overline{RSTOUT} .

Процедура внутреннего RESET занимает 516 тактов ЦПУ (25.8 мкс при частоте ЦПУ 20МГц). Внутреннее состояние RESET остается активным во время всей процедуры RESET до тех пор, пока сигнал на входе \overline{RSTIN} остается активным. После окончания процедуры RESET, начальная конфигурация читается с входов порта 0. После этого выходы ALE, \overline{RD} и \overline{WR} переходят на неактивный уровень напряжения.

Примечание: Значение бита ADP захватывается одновременно с положительным фронтом сигнала \overline{RSTIN} .

После завершения RESET микроконтроллер начинает выполнение программы из адреса 00'0000_H. В этом адресе как правило размещается команда перехода для начала подпрограммы инициализации приложений и периферии и SFR-регистров.

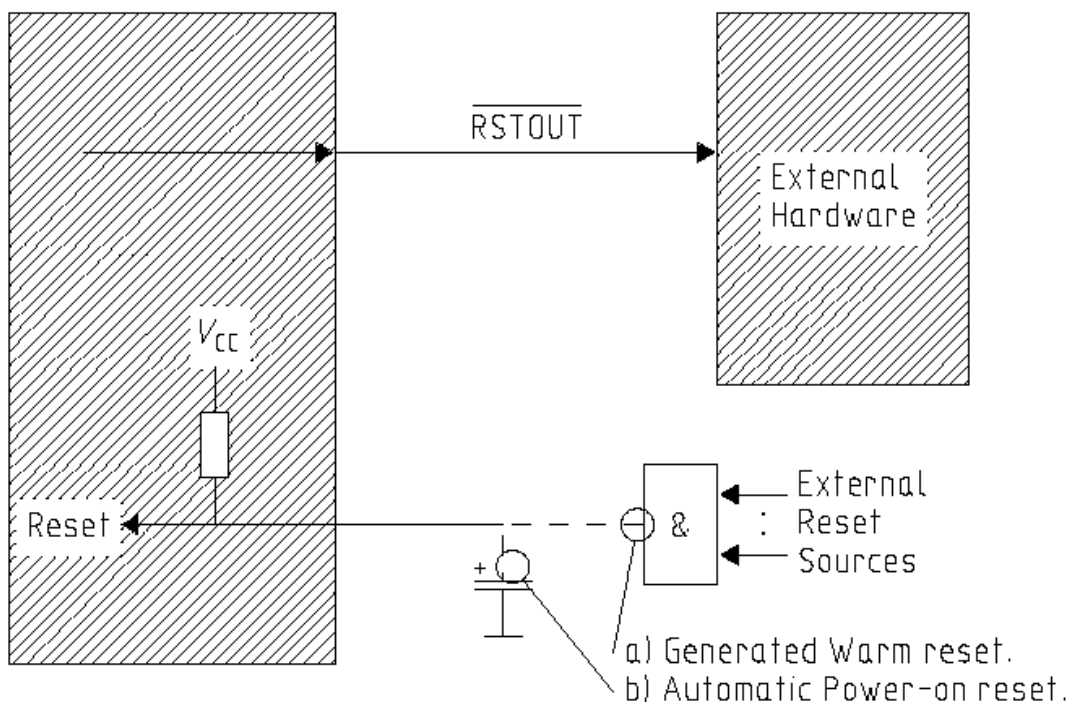


Рисунок 17-1

Схема подключения источников внешнего RESET

Аппаратный RESET

Аппаратный RESET имеет место при поступлении низкого уровня напряжения на вход \overline{RSTIN} . Для гарантированного сигнала захвата необходимо сохранять этот сигнал на входе без изменений как минимум в течении 2 тактов ЦПУ. Более короткие сигналы также могут вызывать RESET, но при этом необходимо их подавать точно в момент захвата сигнала. Тем не менее рекомендуется удерживать сигнал \overline{RSTIN} без изменений в течении 1мс. После завершения процедуры RESET снова проверяется значение сигнала на входе \overline{RSTIN} . Если уровень сигнала все еще остается активным, то состояние RESET будет продлено до тех пор, пока не будет снят сигнал.

Во время RESET для конфигурирования системы необходимо установить на необходимые уровни сигналы входах порта 0. Это имеет большое значение для RESET, прервавшего цикл чтения внешней шины. Во время установки необходимых уровней напряжений в течении короткого промежутка времени уровни сигналов могут не соответствовать необходимым. В частности может быть неправильной выбор частоты PLL. Поэтому рекомендуется в течении 1мс удерживать сигнал на входе \overline{RSTIN} без изменений. В этом случае будут установлены необходимые уровни напряжения, и выбрана необходимая частота PLL.

Внутренний резистор, подключенный к шине питания, имеет номинальное значение 50-150кОм. Простое подключение конденсатора на вход \overline{RSTIN} позволяет осуществлять процедуру RESET при подаче питания.

Примечание: После того как тактовый сигнал стабилизируется (для стабилизации внутреннего осциллятора необходимо около 10... 50 мс), RESET при подаче питания нуждается во времени, занимаемом двумя обычными процедурами RESET (1036 тактов ЦПУ).

Программный RESET

Программный RESET может быть совершен в любой момент выполнения программы по команде SRST. Эта команда применяется в тех случаях, когда необходимо выйти из режима аппаратного загрузчика, или в случае аппаратной ловушки, приведшей к ошибке системы.

Примечание: При программном RESET на конфигурацию системы не влияют сигналы на входах P0L.5... P0L.0.

RESET по переполнению сторожевого таймера

В случае переполнения сторожевого таймера, будет начата процедура RESET. В отличие от аппаратного и программного RESET, цикл работы внешней шины завершается перед началом процедуры RESET, в том случае когда либо не используется сигнал \overline{READY} , либо уровень напряжения сигнала \overline{READY} является активным. В том случае когда напряжение на входе \overline{READY} находится на неактивном уровне после запрограммированного количества waitstates, цикл внешней шины прерывается. После этого начинается процедура RESET.

Примечание: RESET по переполнению сторожевого таймера не может произойти в режиме аппаратного загрузчика.

Выводы микроконтроллера C167 после RESET

В зависимости от функций после завершения RESET различные группы выводов активируются различными способами. Сигналы управления и сигналы шины активируются сразу после завершения процедуры RESET согласно уровням напряжений на входа порта 0. Поэтому после процедуры RESET либо может иметь место внешний доступ, либо сигналы управления останутся на не активном уровне напряжения. Порты ввода-вывода основного назначения остаются в состоянии высокого сопротивления на выходе, до тех пор пока их режим работы не будет программно изменен. Напряжение на выводе \overline{RSTOUT} остается на активном уровне до окончания процедуры инициализации.

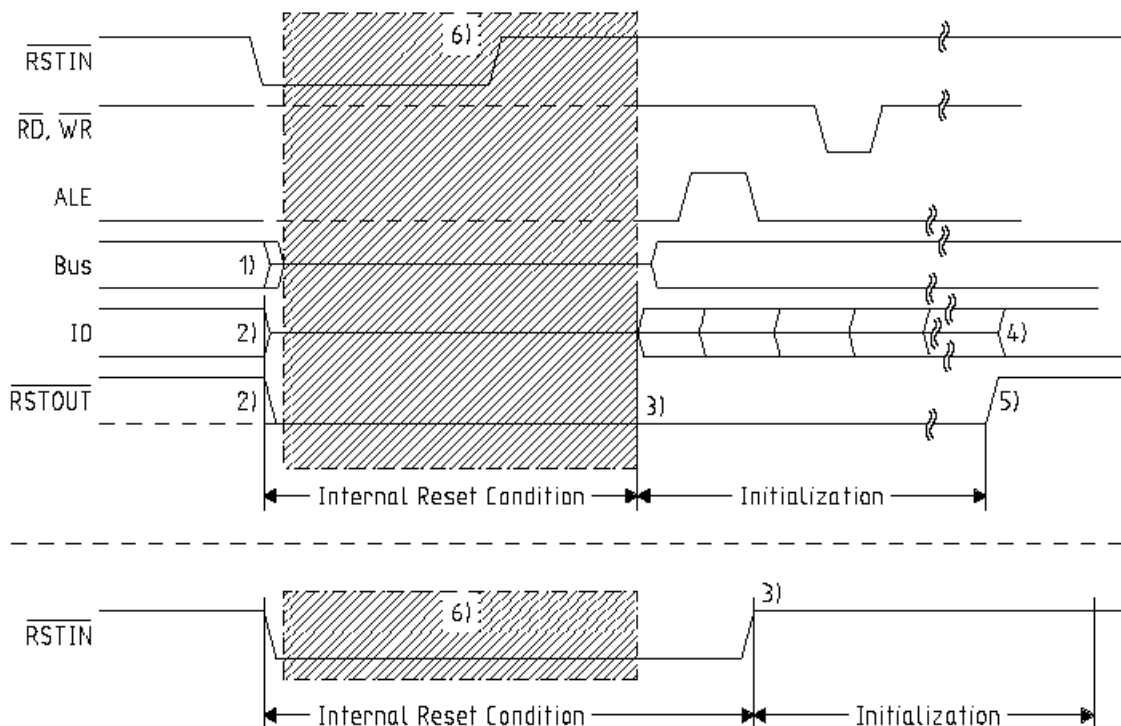


Рисунок 17-2

Сигналы ввода и вывода при RESET

В случае длительного удержания активного уровня напряжения на входе \overline{RSTIN} активация выходных сигналов будет задержана до момента снятия активного уровня напряжения на входе \overline{RSTIN} .

- 1) Текущий цикл работы шины завершается или прерывается
- 2) Переключение асинхронно в случае подачи сигнала \overline{RSTIN} , или синхронно в случае программного RESET, или RESET по переполнению сторожевого таймера
- 3) Точка завершения RESET. C167 начинает выполнять программу
- 4) Активация выводов портов управляется программно
- 5) Выполнение команды EINIT
- 6) Скрытая область процедуры RESET

Вывод \overline{RSTOUT}

Этот вывод предназначен для передачи сигнала RESET к внешним компонентам системы. Сигнал \overline{RSTOUT} переходит на активный уровень в начале процедуры RESET. Он остается активным после окончания процедуры RESET, до тех пор пока не будет выполнена команда EINIT. Это позволяет полностью настроить внутреннюю периферию контроллера, прежде чем разрешить работу внешней периферии.

Примечание: Напряжение сигнала \overline{RSTOUT} , P0L.0 и P0L.1 будет плавающим в случае использования режима эмуляции или режима adapt.

Работа сторожевого таймера после RESET

Сторожевой таймер начинает работу после завершения процедуры RESET. Частота сторожевого таймера составляет $\frac{1}{2}$ от частоты ЦПУ. Поэтому переполнение таймера после завершения процедуры RESET происходит через 131072 такта (6.55мс при частоте ЦПУ 20МГц). В случае переполнения сторожевого таймера выставляется флаг WDTR регистра WDTCON. Этот флаг сбрасывается при внешнем RESET или при обслуживании сторожевого таймера. После завершения процедуры RESET можно отключить работу сторожевого таймера посредством команды DISWDT. Эта команда имеет защищенный формат. Для достижения большей безопасности выполнение этой команды разрешено только в тот период времени, когда не были выполнены команды SRVWDT или EINIT. После этих команд выполнение команды DISWDT не будет оказывать влияние на работу сторожевого таймера.

Значения после RESET регистров C167

Во время процедуры RESET устанавливаются начальные значения в регистрах C167. Большинство SFR-регистров при этом устанавливаются нулевые значения. Поэтому после RESET вся периферия и система прерываний находятся в выключенном состоянии. Однако имеется несколько исключений из этого правила, необходимых для инициализации системы. Значения этих регистров либо фиксированы либо управляются входами порта 0.

DPP1	0001 _H
DPP2	0002 _H
DPP3	0003 _H
CP	FC00 _H
STKUN	FC00 _H
STKOV	FA00 _H
SP	FC00 _H
WDTCON	0002 _H (в случае RESET по переполнению таймера, иначе 0000 _H)
S0RBUF	XX _H
SSCRB	XXXX _H
SYSCON	0XX0 _H (устанавливается согласно конфигурации после RESET)
BUSCON0	0XX0 _H (устанавливается согласно конфигурации после RESET)
RP0H	XX _H
ONES	FFFF _H

Внутренняя память после RESET

Системный RESET не влияет на содержимое внутренней памяти. Однако после RESET, вызванным включением питания, содержимое внутренней памяти неопределенно. Это означает, что значения GPR-регистров и указателей точек источников и точек назначения PEC-передачи также остаются неизменными после обычного RESET, но являются неопределенными после RESET, вызванным подаче питания.

Конфигурация портов и внешней шины во время RESET

После RESET все порты C167 настроены на вход и их драйвера находятся в режиме высокого входного сопротивления. Вывод \overline{ALE} устанавливается на низкий уровень, и выводы \overline{RD} и \overline{WR} устанавливаются на высокий уровень напряжения.

Регистры SYSCON и BUSCON0 настраиваются согласно сигналам на входах порта 0.

В случае внешнего старта ($\overline{EA}=0$):

- Поле типа шины (BTYP) регистра BUSCON0 настраивается согласно сигналам на входе P0L.7 и P0L.6
- BUSACT0 = 1 регистра BUSCON0
- ALECTL0 = 1 регистра BUSCON0
- ROMEN = 0 регистра SYSCON
- BYTDIS регистра SYSCON настраивается согласно ширине шины данных

В случае внутреннего старта ($\overline{EA}=1$)

- Регистр BUSCON0 = 0000_H
- ROMEN = 1 регистра SYSCON
- BYTDIS = 0 регистра SYSCON, т.е. \overline{BHE} отключен

Другие биты регистра BUSCON0 а также другие регистры BUSCON сбрасываются. При этих значениях выбирается самый медленный режим внешнего доступа. Функция Ready отключена до окончания RESET.

Когда RESET завершен, конфигурация порта 0, порта 4 и порта 6 и конфигурация сигнала \overline{BHE} зависит от выбранного типа шины, установленного во время RESET. В тех случаях когда во время RESET выбран один из режимов работы внешней шины, порт 0 (и порт 1) работает в выбранном режиме шины. Порт 4 выводит выбранное количество линий сегмента адреса (после RESET все установлены на нулевые уровни). Порт 6 передает выбранное количество линий \overline{CS} ($\overline{CS0}=0$, в то время как остальные активные линии \overline{CS} выставлены в «1»). В тех случаях, когда используется менее 64 Кбайт памяти, сегментация отключена.

В случае использования аппаратного загрузчика, вывод TxD0/P3.13 переключается в режим вывода после приема нулевого байта.

Все другие выводы остаются в состоянии высокого сопротивления до тех пор, пока их свойства не будут программно изменены, или до тех пор пока они не потребуются для работы внутренней периферии.

Подпрограмма инициализации программы

После выхода из состояния RESET выборка первой команды производится из адреса 00'0000_H. Этот адрес представляет из себя первый адрес таблицы векторов прерываний – вектор RESET. В таблице прерываний под этот вектор отводится четыре слова. Как правило в этих адресах размещается команда перехода непосредственно к программе инициализации.

Примечание: В случае работы в режиме аппаратного загрузчика не происходит выборка кода из адреса 00'0000_H.

В режиме single chip первая команда вызывается из внутренней ROM. В ином случае будет вызвана команда из внешней памяти. В случае включенной внутренней памяти (ROMEN = 1 регистра SYSCON) подпрограмма инициализации может самостоятельно включить и настроить интерфейс внешней шины перед выполнением команды EINIT. Если внешний доступ разрешен после RESET, может потребоваться перенастроить характеристики внешней шины, так как после RESET в регистре SYSCON записаны характеристики самого медленного режима работы шины.

Перед тем как начинать работу с банками GPR-регистров и системным стеком, необходимо записать необходимые значения в CP и SP. Дополнительно к этому необходимо записать значение в регистр переполнения (STKOV) и опустошения стека (STKUN). После RESET регистры CP, SP и STKUN содержат 00'FC00_H, а регистр STKOV содержит 00'FA00_H. В этом случае в стек может быть записано до 256 слов.

В конце инициализации системы можно разрешить работу системы прерываний путем установки «1» в бите IEN регистра PSW. Перед этим необходимо полностью завершить процедуру инициализации.

Подпрограмма инициализации может быть закончена посредством команды EINIT. Эта команда имеет защищенный формат. При выполнении этой команды отключается действие команды DISWDT, отключается возможность записи в регистр SYSCON, и на выводе \overline{RSTOUT} выставляется неактивный уровень напряжения. Этот сигнал может использоваться для сигнализирования внешней периферии об окончании инициализации подпрограммы прерывания.

Начальная конфигурация системы

Большинство функций C167 можно выбрать либо во время фазы инициализации, либо повторно во время выполнения программы. Однако некоторые функции должны быть выбраны на более ранней стадии, поскольку они могут потребоваться для организации доступа к первому адресу программы (в том числе вывод \overline{EA} для указания внутреннего или внешнего старта).

Этот выбор можно сделать во время RESET путем подачи необходимых уровней напряжений на входы порта 0. Во время RESET внутренними средствами на входы порта 0 выставляется высокий уровень напряжения. В случае подключения через внешний резистор к нулевому потенциалу можно добиться установки нулевого значения на входе порта во время RESET.

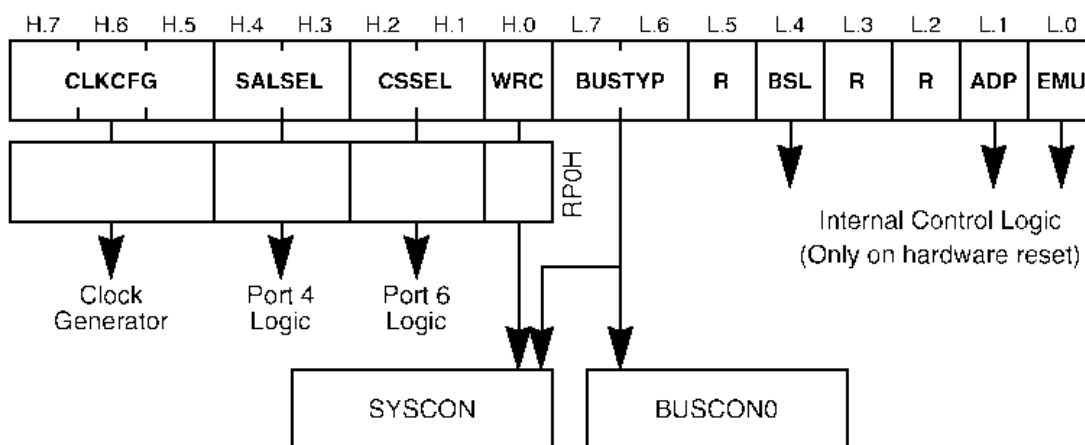


Рисунок 17-3

Конфигурация порта 0 во время RESET

Выводы для управления работой внутренней логики управления и зарезервированные выводы имеют значение только во время аппаратного RESET. Выводы, предназначенные для настройки C167, имеют значение во время любого RESET.

Конфигурация на выводах порта сохраняется для последующего чтения программой в регистре RP0H.

Примечание: Зарезервированные выводы (маркированные «R») должны оставаться на высоком уровне напряжения. Внешняя нагрузка на этих выводах должна иметь такие значения, чтобы не мешать внутренним резисторам поддерживать на входе высокий уровень напряжения во время RESET.

Ниже описаны различные варианты предустановки значений на входе порта 0.

Режим эмуляции

Установка низкого уровня напряжения на входе P0L.0 во время RESET обеспечивает работу в режиме эмуляции. Этот режим позволяет получить доступ к внутренней XBUS-периферии через выводы интерфейса внешней шины. При этом \overline{RSTOUT} находится в состоянии высокого сопротивления. Также автоматически включается вход внешнего тактового сигнала CLKOUT.

Этот режим используется для специальных целей и не используется в базовых устройствах C167. Поэтому следует оставлять высокий уровень на входе P0L.0.

По умолчанию: Режим эмуляции отключен.

Режим Adapt

Установка низкого уровня напряжения на входе P0L.1 во время RESET обеспечивает работу в режиме adapt, похожем на работу в состоянии RESET. В этом режиме C167 переходит в пассивное состояние. При этом выводы C167 находятся в состоянии высокого сопротивления. Также \overline{RSTOUT} находится в состоянии высокого сопротивления. Прекращает работу внутренний осциллятор.

В этом режиме можно смоделировать виртуальное исключение C167 из системы. Поэтому внешний эмулятор может управлять работой системы, даже в тех случаях, когда микроконтроллер остается на месте. Обычный C167 может продолжать управление системой после обычного RESET с $P0L.1 = 1$.

По умолчанию: Режим adapt отключен

Режим аппаратного загрузчика

Установка низкого уровня напряжения на входе P0L.4 во время RESET активирует внутренний аппаратный загрузчик. C167 остается в режиме автозагрузчика, до тех пор пока не будет RESET, при котором $P0L.4=1$.

По умолчанию: Режим аппаратного загрузчика отключен.

Тип внешней шины

Значения на входах P0L.7 и P0L.6 (BUSTYP) позволяют выбрать режим работы внешней шины, в том случае если выбран внешний старт посредством установки значения на входе \overline{EA} . Использование этого режима позволяет производить выборку первого кода после RESET посредством интерфейса внешней шины. Значения этих двух битов копируются в регистр BUSCON0.

Значение бита P0L.7 управляет шириной внешней шины, а значение на выводе P0L.6 управляет режимом вывода адреса (мультиплексный или

демультиплексный). Если необходимо, возможно изменить значения этих битов программно после RESET.

Значения поля ВТУР	Ширина внешней шины	Режим внешней шины адреса
0 0	8-разрядные данные	Демультиплексный адрес
0 1	8-разрядные данные	Мультиплексный адрес
1 0	16-разрядные данные	Демультиплексный адрес
1 1	16-разрядные данные	Мультиплексный адрес

Порт 0 и порт 1 автоматически переключаются в выбранный режим. В режиме мультиплексной шины 16-разрядный адрес сегмента и выходные данные передаются через порт 0, при этом порт 1 не используется. В демультиплексном режиме 16-разрядный адрес передается через порт 1, при этом для данных используется порт 0 или P0L (для 8-разрядной шины адреса).

При использовании 16-разрядной шины данных автоматически включается \overline{BHE} . Для 8-разрядной шины \overline{BHE} может быть отключен при помощи установки значения в бите BYTDIS регистра SYSCON.

По умолчанию: 16-разрядная шина данных мультиплексная шина адреса.

Примечание: Если на входе \overline{EA} выбран внутренний старт, значения на этих входах перестают влиять на режим шины, и битовое поле ВТУР регистра BUSCON0 очищается.

Конфигурация записи

Установка значение на входе P0H.0 (WRC) влияет на режим работы выводов \overline{WR} и \overline{BHE} . Когда используется значение по умолчанию, эти выходы выполняют свою стандартную функцию. Однако при подаче низкого уровня напряжения на этот вход используется альтернативная конфигурация, т.е. \overline{WRH} и \overline{WRL} . Значение бита WRC сохраняется в триггере порта, а инвертированное значение копируется в бит WRCFG регистра SYSCON.

Линии Chip Select

Значения на входах P0H.2 и P0H.1 (CSSEL) во время RESET определяют количество активных сигналов Chip Select.

CSSEL	Линии Chip Select	Примечание
1 1	Пять: CS4... CS0	По умолчанию
1 0	Нет	Порт 6 свободен для IO
0 1	Две: CS1... CS0	
0 0	Три: CS2... CS0	

Примечание: Выбранное количество сигналов CS не может быть программно изменено после RESET.

Линии сегмента адреса

Установка значений на входах P0H.4 и P0H.3 (SALSEL) влияет на количество активных линий сегмента адреса.

SALSEL	Линии сегмента адреса	Доступное адресное пространство
1 1	Две: A17... A16	256 Кбайт (по умолчанию)
1 0	Восемь: A23... A16	16 Мбайт (максимум)
0 1	Нет	64 Кбайта (минимум)
0 0	Четыре: A19... A16	1 Мбайт

Примечание: Выбранное количество линий адреса сегмента не может быть программно изменено после RESET.

Управление генерацией тактового сигнала

Установка во время RESET значений на входах P0H.7 – P0H.5 (CLKCFG) используется для выбора режима генерации тактового сигнала (внутренняя PLL). Тактовый сигнал осциллятора либо напрямую используется ЦПУ, либо сначала попадает на вход внутреннего PLL, с выходной сигнал которого используется ЦПУ. При этом PLL может изменять тактовый сигнал с необходимым значением коэффициента. Значения этих битов сохраняются в регистре RP0H.

P0.15-13 (P0H.7-5)	Частота ЦПУ $f_{CPU} = f_{XTAL} * F$	Пределы частоты внешнего тактового сигнала ¹⁾	Примечания
1 1 1	$f_{XTAL} * 4$	2.5 – 6.25 МГц	По умолчанию
1 1 0	$f_{XTAL} * 3$	3.33 – 8.33 МГц	
1 0 1	$f_{XTAL} * 2$	5 – 12.5 МГц	
1 0 0	$f_{XTAL} * 5$	2 – 5 МГц	
0 x x	$f_{XTAL} * 1$	1 – 25 МГц	Прямая передача ²⁾

¹⁾ Пределы частоты внешнего тактового сигнала установлены таким образом, чтобы выходная частота PLL лежала в рамках 10... 25 МГц.

²⁾ Максимальная зависимость от скважности внешнего тактового сигнала

18 Режимы пониженного энергопотребления

В архитектуру C167 включены два программно управляемых режима пониженного энергопотребления.

В режиме покоя ЦПУ прекращает свою работу, однако периферия продолжает работать. Режим покоя может быть прерван посредством любого RESET или любого запроса на прерывание.

В режиме отключения питания прекращает работу как ЦПУ, так и периферия. Режим отключения питания можно прервать только аппаратным RESET.

Примечание: Перед входом в режим покоя или в режим отключения питания завершаются любые циклы внешней шины. Однако в том случае, когда случае сигнал \overline{READY} включен но не активирован во время предыдущего цикла доступа к шине, микроконтроллер не будет переведен в режим покоя или режим отключения питания.

18.1 Режим покоя

Потребление электроэнергии микроконтроллером может быть уменьшено путем входа в режим покоя. В этом режиме вся периферия, **включая сторожевой таймер**, продолжает нормально функционировать, приостанавливается только работа ЦПУ.

Режим покоя может быть прерван запросом на прерывание от разрешенного источника прерываний, т.е. от того источника, чей индивидуальный флаг разрешения прерываний установлен в «1». При этом режим покоя будет прерываться вне зависимости от значения бита IEN.

В том случае, если эти запросы обслуживаются с помощью подпрограмм обслуживания прерывания, обслуживание запроса начнется только в том случае, если его уровень выше текущего уровня приоритета ЦПУ, а также в том случае когда система обслуживания прерываний включена. После возврата из подпрограммы обслуживания прерываний (т.е. после подпрограммы RETI), ЦПУ начинает выполнять команду, следующую сразу за командой IDLE. В том случае, когда система обслуживания прерываний не включена, или уровень приоритета данного прерывания менее уровня ЦПУ, запрос на прерывание не обслуживается и продолжается выполнение команды, следующей за командой IDLE.

Для запроса на прерывание, запрограммированного на PEC-обслуживание, будет совершена PEC-передача данных, в том случае, если уровень приоритета передачи выше текущего уровня ЦПУ, и система обслуживания прерываний включена. После завершения PEC-передачи данных ЦПУ возвратится в режим покоя. Если текущий уровень приоритета

ЦПУ выше уровня запроса на PEC-обслуживание, или система обслуживания прерываний отключена, PEC-обслуживание не будет произведено, и ЦПУ не вернется в режим покоя а продолжит выполнение программы с команды, следующей после команды IDLE.

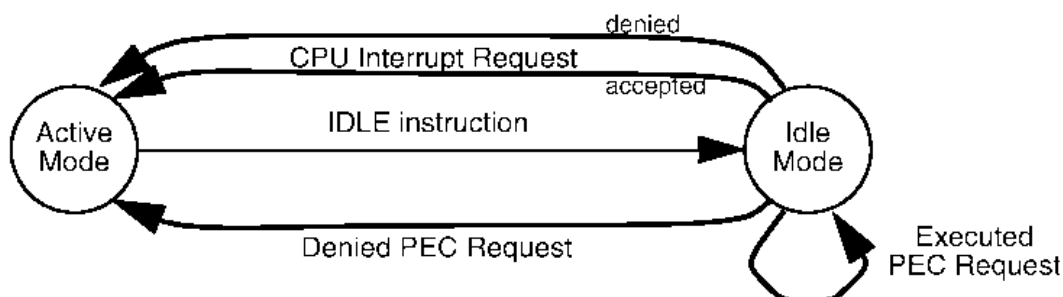


Рисунок 18-1

Граф переходов из режима покоя и активного режима

Режим покоя может быть также прекращен с помощью не маскируемого прерывания на входе \overline{NMI} .

Сторожевой таймер может использоваться для отслеживания в режиме покоя. Внутренний RESET будет совершен в том случае, если не будет запросов на прерывание или сигналов на входе NMI, до переполнения сторожевого таймера. Во избежания переполнения сторожевого таймера во время режима покоя, необходимо программировать сторожевой таймер на необходимую длительность периода перед входом в режим покоя.

18.2 Режим отключения питания

Для большего уменьшения энергопотребления, необходимо переводить микроконтроллер в режим отключения питания. При этом будет отключена подача тактового сигнала на внутренние блоки. Однако содержимое внутренней RAM будет сохранено, так как питание продолжает поступать на вход Vcc. В режиме отключения питания останавливается сторожевой таймер. Этот режим может быть отключен только после аппаратного RESET, т.е. после подачи низкого уровня напряжения на вход \overline{RSTIN} . После RESET порты и значения SFR-регистров будут приведены к начальным значениям, однако содержимое RAM останется без изменений.

Существует два уровня защиты от непреднамеренного входа в режим выключения питания. Во-первых команда PWRDN (команда входа в режим отключения питания) имеет 32-разрядный защищенный формат. Во-вторых эта команда оказывает влияние на состояние микроконтроллера только в том случае, когда на вход \overline{NMI} (не маскируемое прерывание) подается низкий

уровень напряжения. В этом случае после завершения команды PWRDN микроконтроллер входит в режим отключения питания.

Описанная выше последовательность для входа в режим отключения питания может использоваться в соединении с внешним сигналом неисправности питания, подаваемом на вход \overline{NMI} . После подачи этого сигнала микроконтроллер начинает обрабатывать подпрограмму обслуживания NMI ловушки. Эта подпрограмма сохраняет внутреннее состояние микроконтроллера в RAM. После этого выставляются необходимые флаги в RAM, и выполняется команда PWRDN. Если в этот момент сигнал еще не убран со входа \overline{NMI} , то микроконтроллер вводится в режим отключения питания, в ином случае продолжается выполнение программы. В режиме отключения питания напряжение на входах Vcc может быть понижено до 2.5В. При этом содержимое внутренней RAM будет сохранено.

Подпрограмма стартовой инициализации может проверять идентификационные флаги в RAM на предмет определения состояния микроконтроллера перед RESET.

18.3 Значения выводов микроконтроллера во время режима покоя и режима отключения питания

Во время нахождения в режиме покоя все линии портов, используемые для вывода данных основного назначения, выводят данные записанные в порт последними. В том случае когда выводы порта переопределены под альтернативную функцию, состояние на выходе портов определяется соответствующим периферийным модулем.

Те выводы портов, которые используются для функций управления, переходят на неактивный уровень (например \overline{WR}) или переходят в состояние, основанное на последнем цикле доступа к шине (например \overline{BHE}). Выводы портов предназначенные для вывода данных/адреса содержат адрес/данные, которые были выведены в последнем цикле работы внешней шины, перед входом в режим покоя.

P0H в режиме мультиплексной шины с 8-разрядной шиной данных выводит последний адрес. P0L в режиме покоя всегда находится в режиме высокого сопротивления.

Порт 1 выводит младшие 16-разрядов последнего адреса в режиме демультиплексной шины. В ином случае выводы порта 1 содержат значения выходного триггера порта.

Порт 4 выводит значение адреса сегмента последнего доступа к шине. В ином случае порт 4 выводит данные выходного триггера порта.

В режиме отключения питания все выводы портов, настроенных на вывод данных основного назначения, выдают значения данных записанных последними в выходной триггер порта.

Для выводов портов, используемых для альтернативной функции, значения на выходе определяются внутренней периферией.

В ниже приведенной таблице суммируются значения на выводах в режимах пониженного потребления.

Выводы	Режим покоя		Режим отключения питания	
	Нет внешней шины	Внешняя шина включена	Нет внешней шины	Внешняя шина включена
ALE	0	0	0	0
$\overline{RD}, \overline{WR}$	1	1	1	1
CLKOUT	Активный	Активный	1	1
\overline{RSTOUT}	¹⁾	¹⁾	¹⁾	¹⁾
P0L	Данные триггера порта	Высокое сопротивление	Данные триггера порта	Высокое сопротивление
P0H	Данные триггера порта	A15...A8 ²⁾ / Высокое сопротивление	Данные триггера порта	A15...A8 ²⁾ / Высокое сопротивление
Порт 1	Данные триггера порта	Последний ³⁾ адрес/ Данные триггера порта	Данные триггера порта	Последний ³⁾ адрес/ Данные триггера порта
Порт 4	Данные триггера порта	Последний сегмент/ Данные триггера порта	Данные триггера порта	Последний сегмент/ Данные триггера порта
\overline{BHE}	Данные триггера порта	Последнее значение	Данные триггера порта	Последнее значение
\overline{HLDA}	Данные триггера порта	Последнее значение	Данные триггера порта	Последнее значение
\overline{BREQ}	Данные триггера порта	1	Данные триггера порта	1
\overline{CSx}	Данные триггера порта	Последнее значение ⁴⁾	Данные триггера порта	Последнее значение ⁴⁾
Другие порты Другие выводы	Данные триггера порта/ Альт. функция	Данные триггера порта/ Альт. функция	Данные триггера порта/ Альт. функция	Данные триггера порта/ Альт. функция

Примечание:

¹⁾: «1», если перед входом в режим отключения питания была выполнена команда EINIT. В ином случае – «0».

²⁾: Для режима мультиплексной шины с 8-разрядной шиной данных

³⁾: Для режима демultipлексной шины

⁴⁾: Последний активный сигнал CS остается в активном состоянии («0»). Все другие сигналы остаются в неактивном состоянии («1»).

19 Системное программирование

Для облегчения разработки программного обеспечения, в набор команд включено большое количество возможностей, в том числе конструирование модулей, использование меток и использование контекстного переключения. Ниже приведенные советы позволяют полностью использовать набор команд.

Команды, включающие в себя подмножество команд

Во многих случаях, наборы команд других микроконтроллеров реализованы в архитектуре C167 в виде одной команды. Это позволяет повысить функциональность системы и понизить сложность декодирования. Для облегчения ассемблерного программирования, команды, аналогичные командам других микроконтроллеров, могут быть выполнены в виде макросов. Таким образом можно достичь совместимости на уровне названия команд.

Напрямую заменяемые команды – это команды других микроконтроллеров, замененные следующими командами архитектуры C167:

Подменяемые команды	Команды C167	Функция
CLR Rn	AND Rn, #0 _H	Очистка регистра
CPLB Bit	BMOVN Bit, Bit	Функция дополнения
DEC Rn	SUB Rn, #1 _H	Декрементирование
INC Rn	ADD Rn, #1 _H	Инкрементирование
SWAPB Rn	ROR Rn, #8 _H	Обмен байтами в слове

Умножение и деление

Каждая команда неявным образом использует 32-разрядный регистр MD (MDL – младшие 16 разрядов, MDH – старшие 16 разрядов). При чтении этого регистра или в начале операции умножения или деления устанавливается флаг MDRIU регистра MDC. После чтения регистра MDL флаг сбрасывается. Так как во время выполнения команд умножения или деления возможно обслуживание прерывания, не дожидаясь окончания команды, этот флаг необходим тем подпрограммам, которые используют собственные команды умножения или деления. В случае прерывания выполнения команд умножения или деления, такими подпрограммами, подпрограмма должна сначала сохранить значения регистра MD в стеке.

Флаг переполнения устанавливается в том случае, если результат умножения или деления содержит больше 16 разрядов. Этот флаг используется для определения необходимости копирования обоих слов результата в регистре MD.

Для выполнения операции умножения двух беззнаковых 16-битных чисел необходимо совершить следующую последовательность команд:

```
SAVE:   JNB      MDRIU, START ;Проверка использования MD
        SCXT     MDC, #0010H ;Сохранение и очистка регистра
                                   ;управления, снятие флага MDRIU
                                   ;(требуется только для
                                   ;прерываемых команд)
        BSET     SAVED      ;Указание сохранения
        PUSH     MDH        ;Сохранение предыдущего
        PUSH     MDL        ;значения MD в системном стеке
START:   MULU     R1, R2     ;Умножение 16x16,
                                   ;устанавливается флаг MDRIU
        JMPR     cc_NV, COPYL ;Тестирование переполнения резул.
        MOV      R3, MDH     ;Сохранение MDH в R3
COPYL:   MOV      R4, MDL     ;Сохранение MDL в R4
                                   ;очистка MDRIU
RESTORE: JNB      SAVED, DONE ;Проверка сохранения старого
                                   ;значения MD в стеке
        POP      MDL
        POP      MDH
        POP      MDC
        BCLR     MDC         ;Умножение завершено
DONE:    ...
```

В этом примере необходимость в сохранении и восстановлении значений в стеке, имеется в том случае, если эта последовательность действий является частью подпрограммы прерывания, которая прервала выполнение команд умножения или деления.

Для совершения деления пользователь должен сначала записать делимое в регистр MD. В случае деления 16/16, необходимо произвести запись только в MDL. Результат сохраняется в регистре MD. При этом в MDL содержится целый результат деления, а в MDH – остаток от деления.

Ниже приведена последовательность для совершения деления 32/16.

```
MOV      MDH, R1 ;Запись делимого в регистр MD
                                   ;При этом устанавливается флаг MDRIU
MOV      MDL, R2 ;Запись младшей половины делимого
DIV       R3      ;Деление 32-битного MD на 16-битный R3
JMPR     cc_V, ERROR ;Проверка переполнения
MOV      R3, MDH  ;Сохранения остатка
MOV      R4, MDL  ;Сохранения целого результата в R4, сброс MDRIU
```

В том случае, когда команда умножения или деления прерывается во время исполнения, адрес прерванной команды сохраняется в стеке, и в

регистре PSW устанавливается флаг MULIP для подпрограммы прерывания. В момент выхода из подпрограммы прерывания (по команде RETI), перед записью из стека старого значения PSW проверяется бит MULIP. Если этот бит установлен, команда умножения/деления читается из адреса сохраненного в стеке и завершается после выполнения команды RETI.

Примечание: Флаг MULIP – часть окружения прерываемой задачи. В том случае когда после прерывания нет возврата в прерванную задачу (т.е. scheduler переключает на другую задачу), необходимо изменить флаг MULIP на значение, соответствующее новой задаче.

Вычисления в BCD-формате

C167 не осуществляет напрямую вычисления для BCD-данных. При осуществлении вычисления с BCD-данными необходимо произвести преобразования в двоичные данные, и затем произвести обратное преобразования в BCD-формат. Для обеспечения повышенной производительности команд деления двоичные данные можно быстро преобразовать в BCD-данные посредством деления на 10_в. Преобразование из BCD-формата в двоичный формат можно произвести посредством многократных команд побитового сдвига. При этом время занимаемое на выполнение данной последовательности команд совпадает с временем выполнения команд напрямую поддерживающих BCD-формат данных.

19.1 Операции со стеком

C167 поддерживает два типа стека. Системный стек используется микроконтроллером и расположен во внутренней RAM. Пользовательский стек используется для хранения пользовательских данных в стеке во внутренней либо внешней RAM. Для обоих типов стека характерным является рост стека от больших к меньшим адресам.

Внутренний системный стек

Системный стек предназначен для хранения векторов возврата, указателей сегмента, состояния процессора при работе в подпрограммах прерываний. Текущее значение вершины стека хранится в регистре указателя стека (SP). При записи новых данных в стек значение указателя декрементируется. При чтении данных из стека значение указателя инкрементируется.

Также возможно использование системного стека для временного хранения данных или для пересылки данных между задачами или подпрограммами. Однако в большинстве случаев система хранения данных в банках регистров обеспечивает лучшую производительность при пересылки данных между задачами.

Примечание: В системном стеке возможно сохранение только слов данных. Для хранения байтов данных необходимо либо произвести перевод их в формат слов, либо не обращать внимание на другой байт слова.

Для обнаружения переполнения или опустошения стека предназначены два регистра: STKOV и STKUN. При обнаружении переполнения или опустошении срабатывают соответствующие ловушки.

Примечание: В случае программного изменения значения указателя стека, не происходит сравнения с STKOV или STKUN.

Во многих случаях по адресам векторов ловушек переполнения или опустошения стека располагают команду программного RESET (SRST).

Также возможно приспособить ловушки по переполнению или опустошению стека для создания кэша стека, т.е. для создания большего значения стека. При этом в системном стеке будет располагаться только часть содержимого стека, остальная часть стека будет в другой части памяти. Таким образом можно высвободить большую часть внутренней RAM для кода программы, данных или банков регистров данных.

Циркулирующий (виртуальный) стек

Этот способ использования стека позволяет сохранять значение стека при достижении им максимального значения. В этот момент часть данных стека должна быть сохранена во внешней памяти, для того чтобы освободить пространство для последующего заполнения стека. Эта процедура называется *flushing* стека. После выполнения некоторого количества возвратов из подпрограмм или после выполнения нескольких команд *pop* стек опустошается. При этом имеет место ловушка по опустошению стека, и необходимо осуществить возврат ранее сохраненных данных в стек из внешней памяти. Эта процедура называется *filling* стека. Так как большое количество вложенных подпрограмм имеет место достаточно редко, то *flushing* и *filling* случаются достаточно редко. В тех программах, где данные события происходят довольно часто, не следует употреблять эту структуру, так как при этом резко замедляется выполнение основной программы.

Основной способ изменения области виртуального стека базируется на изменении значений регистров SP, STKOV и STKUN. Изменение значений этих регистров необходимо для определения физической области стека во внутренней RAM для аппаратного обеспечения. Для области виртуального стека можно указать все возможные области адреса 00`F000_H – 00`FFFE_H. При этом значения регистров STKOV и STKUN должны лежать в том же 4Кбайтном пределе.

Размер физической области стека во внутренней RAM определяется в битовом поле STKSZ регистра SYSCON.

<STKSZ>	Размер стека	Адреса во внутренней RAM	Значащие биты указателя стека
0 0 0	256	00`FBFE _H ... 00`FA00 _H (по умолчанию)	SP.8... SP.0
0 0 1	128	00`FBFE _H ... 00`FB00 _H	SP.7... SP.0
0 1 0	64	00`FBFE _H ... 00`FB80 _H	SP.6... SP.0
0 1 1	32	00`FBFE _H ... 00`FBC0 _H	SP.5... SP.0
1 0 0	512	00`FBFE _H ... 00`F800 _H	SP.9... SP.0
1 0 1	---	Зарезервировано. Не использ.	---
1 1 0	---	Зарезервировано. Не использ.	---
1 1 1	1024	00`FDFE _H ... 00`F600 _H (не циркулирующий стек)	SP.11... SP.0

Адреса виртуального стека преобразуются в физические адреса посредством совмещения значащих битов указателя стека SP с значащими битами верхнего предела области физического стека (00`FBFE_H). Это совмещение производится аппаратно.

После RESET значения всех трех регистров направлены в область физического стека и позволяют использовать без изменений область стека на 256 слов (STKOV = FC00_H, STKUN = FC00_H, SP = FC00_H, STKSZ = 000_B).

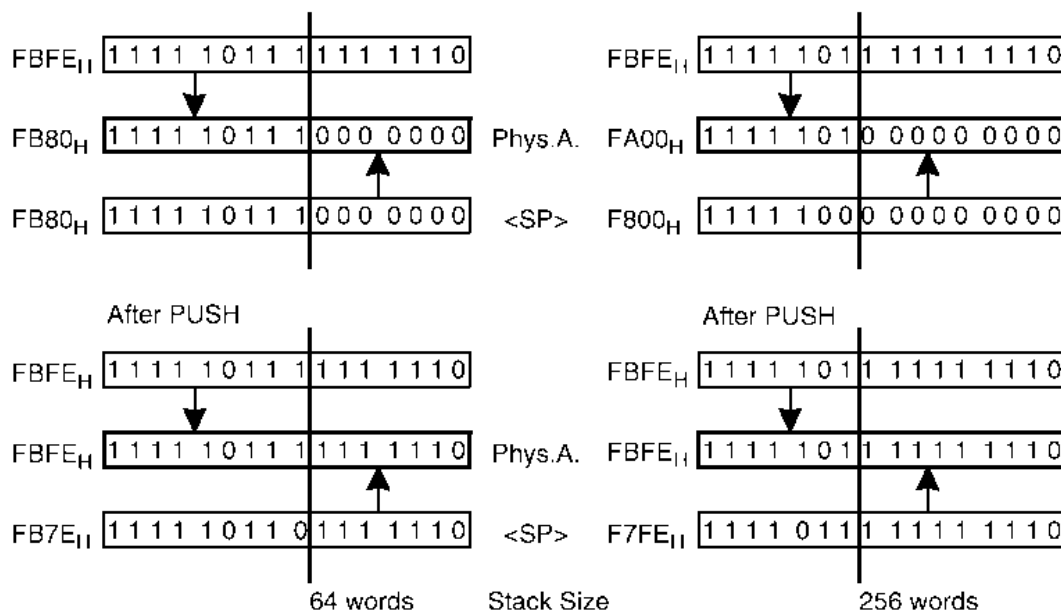


Рисунок 19-1

Создание физического адреса стека

В следующем примере показывается работа механизма циркулирующего стека, оказывающего эффект на размещение виртуального стека. Сначала регистр R1 сохраняется в наименьшем физическом адресе

стека, согласно выбранному размеру стека. По следующей команде регистр R2 будет сохранен по наибольшему адресу стека, при этом значение SP будет уменьшено на 2.

```
MOV      SP, #0F802H      ;Установка значения SP после ввода в
                           ;физический стек 256 слов
                           ;(SP) = F802H: Физический адрес = FA02H
PUSH     R1                ;(SP) = F800H: Физический адрес = FA00H
PUSH     R2                ;(SP) = F7FEH: Физический адрес = FBFEH
```

Эффект преобразования адресов заключается в том, что физический адрес стека возвращается из конца стека в его начало. При проведении операций filling и flushing внутреннего стека, механизм циркулярного стека необходим только для перемещения той части данных стека, которая будет использоваться в данный момент (т.е. верхняя часть стека). Данные стека, которые остаются в нижней части внутреннего стека, не нуждаются в перемещении при операциях filling или flushing, так как указатель стека автоматически вернется к началу свободной части области стека.

Примечание: Режим циркулярного стека применим для стека размером 32 или 512 слов (STKSZ = 000 или 100). Этот режим не применим для STKSZ = 111, так как при этом под внутренний стек используется вся область внутреннего RAM.

При достижении одной из границ внутреннего стека, вызывается ловушка по переполнению или опустошению стека. В подпрограмме обслуживания ловушки осуществляется перемещение заранее установленной части внутреннего стека из внешнего стека или во внешний стек. Общее количество переданных данных определяется средним необходимым значением стека, а также зависит от частоты совершения входов в подпрограммы, от частоты обслуживания ловушек и прерываний. В большинстве случаев эта величина составляет от 1/4 до 1/10 от размера внутреннего стека. После завершения передачи, значение указателей границ стека изменяется и отражает заново определенную область внутреннего стека. Таким образом можно писать программы не обращая внимание на ограничения размера стека. При использовании этого механизма увеличивается время выполнения программы, за счет выполнения подпрограмм обслуживания ловушек.

Для инициализации микроконтроллера в режим циркулярного стека необходимо произвести следующие операции:

- Определить размер области физического системного стека во внутренней RAM (битовое поле STKSZ регистра SYSCON)
- Определить два регистра границ внешнего стека. Эти значения будут позже проверяться при передачи данных во внешний стек во время выполнения подпрограмм обслуживания ловушек

- Установить указатель переполнения стека (STKOV) на предел области внутреннего стека, причем к величине предела необходимо добавить 6 слов (необходимо зарезервировать пространство для сохранения двух входов в прерывание).

Внутренний стек будет заполняться до тех пор, пока не достигнет значения указателя переполнения стека. После входа в подпрограмму обслуживания ловушки вершина стека будет сохранена во внешней памяти. Внутренние указатели будут изменены таким образом, чтобы указывать на измененную область памяти. После выхода из подпрограммы обслуживания ловушки, значение указателя стека будет возвращено к вершине внутреннего стека, и таким образом стек продолжит возрастание до тех пор, пока не будет заново достигнуто значение указателя переполнения стека.

При достижении опустошения стека, дно стека перезагружается из области внешней памяти и после этого изменяются значения внутренних указателей.

Линейный стек

Архитектура C167 позволяет использовать режим линейного стека (STKSZ = 111). В этом режиме системный стек занимает всю область внутренней RAM. При этом создается большой объем стека, и отпадает необходимость в подпрограммах для передачи данных в режиме циркулирующего стека. Однако в режиме линейного стека не остается места в RAM для переменных или программного кода. В этом режиме обнаружение переполнения или опустошения стека обслуживается как фатальная ошибка.

Несмотря на то, что указатель стека может покрывать область адресов от 00'F000_H до 00'FFFE_H, физически системный стек должен располагаться во внутренней RAM. Таким образом использоваться может только область 00'F600_H – 00'FDFF_H.

Примечание: Следует избегать осуществления доступа к адресам 00'F000_H – 00'F5FE_H, поскольку в них содержатся ESFR-регистры, а также зарезервированная область.

Пользовательский стек

Пользовательский стек предназначен для разгрузки системного стека путем создания стеков данных отдельного для каждой задачи. При использовании пользовательского стека отсутствует аппаратное определение переполнения и опустошения стека. Для осуществления пользовательского стека подходят следующие режимы адресации:

[-Rw], Rb или [-Rw], Rw: косвенная адресация с предварительным декрементированием.

Использование этого режима позволяет осуществлять пересылку одного байта или слова в пользовательский стек. В этом режиме можно

использовать только команды MOV. В качестве указателя пользовательского стека можно определять любой GPR-регистр.

Rb, [Rw+] или Rw, [Rw+] Косвенная адресация с последующим инкрементированием значения индексного регистра.

Используется для изъятия одного байта или слова из пользовательского стека. Этот режим доступен для большинства команд. Однако в качестве указателя пользовательского стека могут выступать только регистры R0 – R3 блока GPR-регистров.

Rb, [Rw+] или Rw, [Rw+] Косвенная адресация с последующим инкрементированием.

Используется для изъятия одного байта или слова из пользовательского стека. В этом режиме можно использовать только команды MOV. В качестве указателя пользовательского стека можно определять любой GPR-регистр.