Глава 2. ОСОБЕННОСТИ АППАРАТНОЙ РЕАЛИЗАЦИИ МИКРОКОНТРОЛЛЕРОВ ФИРМЫ CYGNAL

2.1. Микроконтроллерное ядро CIP-51 фирмы Cygnal

Все микроконтроллеры фирмы Cygnal оснащены усовершенствованным микроконтроллерным ядром CIP-51, полностью совместимым по набору инструкций со стандартным MCS-51™ ядром. Это позволяет использовать при разработке программного обеспечения стандартные x51 ориентированные ассемблеры и компиляторы языков высокого уровня (C-51, PL/M-51, FORTRAN-51, PASCAL-51 и т.п.), а также проблемно ориентированные библиотеки подпрограмм, созданные различными коллективами за десятилетия существования x51-совместимых микроконтроллеров. Микроконтроллерное ядро (CIP-51) содержит полный набор периферийных узлов, стандартный для MCS-51. В зависимости от типа семейства, ядро может содержать 3, 4 или 5 таймеров-счетчиков, один или два последовательных порта UART, как минимум 256 байт встроенной оперативной памяти, 128-байтный регистр специальных функций SFR (Special Function Register). Ядро также обеспечивает управление линиями портов ввода/вывода, которых микроконтроллеры различных семейств могут иметь от 1 до 8 (т.е. от 8 до 64 линий ввода/вывода). Несомненным достоинством этого ядра является встроенная аппаратура отладки. Ядро обеспечивает непосредственное управление аналоговыми и цифровыми подсистемами микроконтроллера через соответствующие регистры SFR. Таким образом, ядро CIP-51 с одной стороны обеспечивает полную совместимость со стандартным x51 совместимым ядром, с другой имеет значительно более широкие аппаратные возможности за счет пополнения встроенной цифровой и аналоговой периферии. Структурная схема ядра CIP-51 показана на рис.2.1.

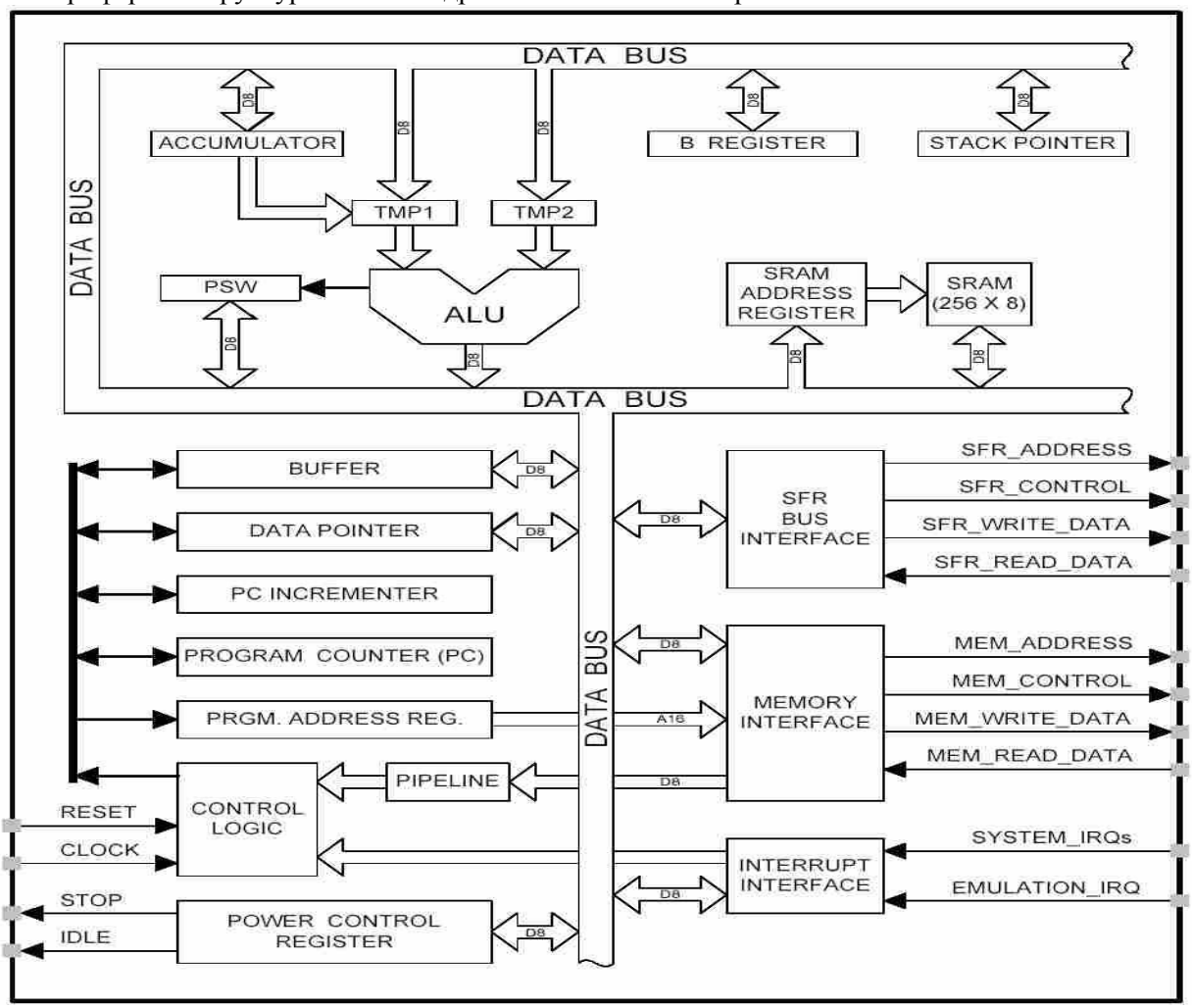

Рис.2.1. Структурная схема ядра CIP-51

Несомненно, важнейшим достоинством ядра CIP-51 является усовершенствованная конвейерная архитектура, которая позволяет значительно увеличить производительность по сравнению со стандартной x51 (8051) архитектурой. В микроконтроллерах со стандартной архитектурой 8051 все

инструкции, за исключением MUL и DIV, выполнялись за 12 или 24 машинных цикла. При этом максимальная тактовая частота для большинства x51 совместимых микроконтроллеров составляла 12-24 МГц, и лишь некоторые x51 совместимые микроконтроллеры могли работать на более высоких частотах. Модернизированное ядро CIP-51 выполняет 70% инструкций ***за один или два*** машинных цикла, и вообще не имеет инструкций, выполняющихся более чем за восемь машинных циклов. При этом, практически все микроконтроллеры фирмы Cygnal (27 типов, 72%) могут работать при частоте тактового генератора 25МГц, шесть микроконтроллеров работают на частотах до 20Мгц (16%), и четыре новых микроконтроллера седьмого семейства (C8051F12x) функционируют на частотах до 100Мгц. При этом, соответственно развивается пиковая (предельная) производительность 25, 20 и 100 миллионов инструкций в секунду - MIPS (Million Instructions Per Second). В качестве примера на рис.2.2. показано соотношение предельной производительности некоторых распространенных микроконтроллеров в сравнении с предельной производительностью микроконтроллеров фирмы Cygnal при тактовой частоте 25МГц.

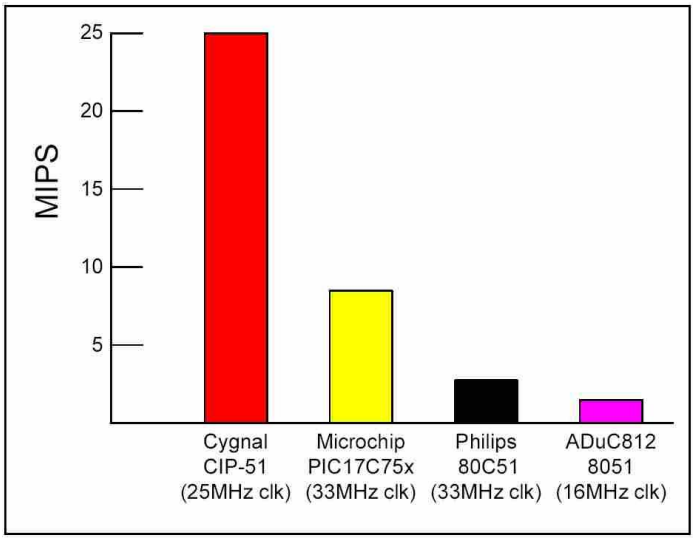

Рис.2.2. Соотношение предельной производительности некоторых микроконтроллеров

Ядро CIP-51 имеет 109 инструкций. Количество инструкций и соответствующее количество машинных циклов, необходимых для их исполнения, приведено в таблице 2.1.

Таблица 2.1.

Инструкция	26	50	5	14	7	3	1	2	1
Количество машинных циклов	1	2	2/3	3	3/4	4	4/5	5	8

Еще одним важным достоинством микропроцессорного ядра CIP-51 является наличие встроенной аппаратной подсистемы отладки программного обеспечения. Связь с подсистемой микроконтроллера осуществляется через последовательный интерфейс JTAG, соответствующий IEEE 1149.1. При этом, обеспечиваются как режим внутрисистемного программирования - ISP (In System Programarable), так и собственно режим отладки. При программировании возможна запись, как всего массива программы, так и модификация отдельных байтов. Естественно, что содержимое памяти программ может также читаться и сверяться с оригиналом. Содержимое любого байта программ может читаться или изменяться с использованием инструкций MOVC или MOVX, что также позволяет осуществлять энергонезависимое хранение данных и оперативно их модифицировать под управлением программы.

Встроенная схема обеспечивает отладку в режиме реального времени выполнения программы. При этом возможны установка точек останова и контроля переменных, запуск, останов и пошаговое выполнение программы (включая подпрограммы прерываний), контроль обращений программы к стеку, контроль и модификация содержимого регистров и оперативной памяти. Используемый метод отладки ненавязчив и неагрессивен, и не требует никаких ресурсов микроконтроллера (RAM, стека, таймеров и т.п.).

Для ядра CIP-51 с подсистемой отладки фирмой Cygnal созданы программно-инструментальные средства отладки. Программное обеспечение выполнено в виде интегрированной среды развития - IDE (Integrated Development Environment), включающей мощный редактор, макроассемблер, отладчик и программатор. Также имеется компилятор языка "C".

Важно отметить, что вся цифровая и аналоговая периферия микроконтроллера функционирует корректно в процессе отладки. В процессе останова микроконтроллера или в пошаговом режиме вся периферия (кроме ADC) также работает в соответствующем режиме.

Фирма Cygnal предлагает эволюционные комплекты развития - "киты" (Developments Kits) для различных семейств и групп своих микроконтроллеров, включающие:

- программное обеспечение для персонального компьютера PC (Personal Computer);
- конвертор RS232C (персонального компьютера) в JTAG (питается от платы развития и потребляет ~20mA при напряжении питания 2.7-3.6V);
- собственно плату развития (с установленным соответствующим микроконтроллером и большим макетным полем);
- соединительные кабели и настенный источник питания.

Программное обеспечение может функционировать в операционных системах Windows 95/98 /NT/2000/XP.

2.2. Набор инструкций микроконтроллеров фирмы Cygnal

Набор инструкций ядра CIP-51 полностью совместим со стандартным MCS-51™ набором инструкций. Соответственно, все программное обеспечение, созданное для MCS-51 может быть использовано и для CIP-51. Все инструкции CIP-51 эквивалентны инструкциям MCS-51 и по мнемонике, и по кодам. Отличие имеется только во времени выполнения инструкций.

Во многих исполнениях x51-совместимых микроконтроллеров существует различие между машинным циклом и циклом (периодом) тактового генератора (или просто "тактом"). Т.е., в них один машинный цикл может быть равен от 2 до 12 тактов. В отличие от них, время выполнения инструкций в CIP-51 измеряется только в тактах. Благодаря конвейерной архитектуре ядра CIP-51, многие инструкции выполняются непосредственно во время чтения самой инструкции, т.е. в том же такте. Инструкции условного ветвления выполняются быстрее, если условие не соблюдается. Полный набор инструкций CIP-51 с указанием мнемоники, количества байт и времени выполнения в тактах приведен в таблице 2.2.

В ядре CIP-51 инструкция MOVX имеет возможность доступа к Flash памяти программ, в отличие от других x51-совместимых микроконтроллеров, в которых она используется для доступа к внешней памяти данных. Это позволяет модифицировать Flash память программ в процессе выполнения программы, использовать часть Flash памяти для энергонезависимого хранения данных, осуществлять отладку программного обеспечения. Управление доступом осуществляется путем модификации соответствующего бита в регистре PSCTL.

Таблица 2.2.

Набор инструкций микроконтроллеров фирмы Cygnal

Мнемоника	Описание	Кол-во байтов	Кол-во тактов
ARITHMETIC OPERATIONS - АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ			
ADD A, Rn	Сложить содержимое регистра с A	1	1
ADD A, direct	Сложить прямо адресуемый байт с A	2	2
ADD A, @Ri	Сложить косвенно адресуемый байт с A	1	2
ADD A, #data	Сложить данные с A	2	2
ADDC A, Rn	Сложить содержимое регистра с A с переносом	1	1
ADDC A, direct	Сложить прямо адресуемый байт с A с переносом	2	2
ADDC A, @Ri	Сложить косвенно адресуемый байт с A с переносом	1	2
ADDC A, #data	Сложить данные с A с переносом	2	2
SUBB A, Rn	Вычесть содержимое регистра из A с заемом	1	1
SUBB A, direct	Вычесть прямо адресуемый байт из A с заемом	2	2
SUBB A, @Ri	Вычесть косвенно адресуемый байт из A с заемом	1	2
SUBB A, #data	Вычесть данные из A с заемом	2	2
INC A	Увеличить A (+1)	1	1
INC Rn	Увеличить содержимое регистра	1	1
INC direct	Увеличить прямо адресуемый байт	2	2
INC @Ri	Увеличить косвенно адресуемый байт	1	2
DEC A	Уменьшить A (-1)	1	1
DEC Rn	Уменьшить содержимое регистра	1	1
DEC direct	Уменьшить прямо адресуемый байт	2	2
DEC @Ri	Уменьшить косвенно адресуемый байт	1	2

INC DPTR	Увеличить указатель данных	1	1
MUL AB	Умножить A на B	1	4
DIV AB	Разделить A на B	1	8
DA A	Десятичная коррекция аккумулятора	1	1
LOGICAL OPERATIONS - ЛОГИЧЕСКИЕ ОПЕРАЦИИ			
ANL A, Rn	Операция "И" регистра и A	1	1
ANL A, direct	Операция "И" прямо адресуемого байта и A	2	2
ANL A, @Ri	Операция "И" косвенно адресуемого байта и A	1	2
ANL A, #data	Операция "И" данных и A	2	2
ANL direct, A	Операция "И" A и прямо адресуемого байта	2	2
ANL direct, #data	Операция "И" прямо адресуемого байта и данных	3	3
ORL A, Rn	Операция "ИЛИ" регистра и A	1	1
ORL A, direct	Операция "ИЛИ" прямо адресуемого байта и A	2	2
ORL A, @Ri	Операция "ИЛИ" косвенно адресуемого байта и A	1	2
ORL A, #data	Операция "ИЛИ" данных и A	2	2
ORL direct, A	Операция "ИЛИ" A и прямо адресуемого байта	2	2
ORL direct, #data	Операция "ИЛИ" прямо адресуемого байта и данных	3	3
XRL A, Rn	Операция "Исключительное ИЛИ" регистра и A	1	1
XRL A, direct	Операция "Исключительное ИЛИ" прямо адресуемого байта и A	2	2
XRL A, @Ri	Операция "Исключительное ИЛИ" косвенно адресуемого байта и A	1	2
XRL A, #data	Операция "Исключительное ИЛИ" данных и A	2	2
XRL direct, A	Операция "Исключительное ИЛИ" A и прямо адресуемого байта	2	2
XRL direct, #data	Операция "Исключительное ИЛИ" прямо адресуемого байта и данных	3	3
CLR A	Очистить A	1	1
CPL A	Дополнение A	1	1
RL A	Сдвиг A влево	1	1
RLC A	Сдвиг A влево с переносом	1	1
RR A	Сдвиг A вправо	1	1
RRC A	Сдвиг A вправо с переносом	1	1
SWAP A	Обмен тетрадами A	1	1
DATA TRANSFER - ОПЕРАЦИИ ПЕРЕДАЧИ ДАННЫХ			
MOV A, Rn	Перенести содержимое регистра в A	1	1
MOV A, direct	Перенести прямо адресуемый байт в A	2	2
MOV A, @Ri	Перенести косвенно адресуемый байт в A	1	2
MOV A, #data	Перенести данные в A	2	2
MOV Rn, A	Перенести содержимое A в регистр	1	1
MOV Rn, direct	Перенести прямо адресуемый байт в регистр	2	2
MOV Rn, #data	Перенести данные в регистр	2	2
MOV direct, A	Перенести содержимое A в прямо адресуемый байт	2	2
MOV direct, Rn	Перенести содержимое регистра в прямо адресуемый байт	2	2
MOV direct, direct	Перенести прямо адресуемый байт в прямо адресуемый байт	3	3
MOV direct, @Ri	Перенести косвенно адресуемый байт в прямо адресуемый байт	2	2
MOV direct, #data	Перенести данные в прямо адресуемый байт	3	3
MOV @Ri, A	Перенести содержимое A в косвенно адресуемый байт	1	2
MOV @Ri, direct	Перенести прямо адресуемый байт в косвенно адресуемый байт	2	2
MOV @Ri, #data	Перенести данные в косвенно адресуемый байт	2	2
MOV DPTR, #datal6	Перенести 16-bit данные в DPTR	3	3
MOVC A, @A+DPTR	Перенести косвенный байт (DPTR) в A	1	3
MOVC A, @A+PC	Перенести косвенный байт (PC) в A	1	3
MOVX A, @Ri	Перенести внешние данные (8-bit адрес) в A	1	3
MOVX @Ri, A	Перенести A во внешние данные (8-bit адрес)	1	3
MOVX A, @DPTR	Перенести внешние данные (16-bit адрес) в A	1	3
MOVX @DPTR, A	Перенести A во внешние данные (16-bit адрес)	1	3
PUSH direct	Записать прямо адресуемый байт в стек	2	2
POP direct	Извлечь прямо адресуемый байт из стека	2	2
XCH A, Rn	Поменять содержимое регистра и A	1	1
XCH A, direct	Поменять содержимое прямо адресуемого байта и A	2	2
XCH A, @Ri	Поменять содержимое косвенно адресуемого байта и A	1	2
XCHD A, @Ri	Поменять младшую тетраду косвенно адресуемого байта и A	1	2

BOOLEAN MANIPULATION - БИТОВЫЕ ОПЕРАЦИИ			
CLR C	Сбросить флаг переноса (=0)	1	1
CLR bit	Сбросить прямо адресуемый бит	2	2
SETB C	Установить флаг переноса (=1)	1	1
SETB bit	Установить прямо адресуемый бит	2	2
CPL C	Инвертировать перенос	1	1
CPL bit	Инвертировать прямо адресуемый бит	2	2
ANL C, bit	Операция "И" прямо адресуемого бита и переноса	2	2
ANL C, /bit	Операция "И" инверсного прямо адресуемого бита и переноса	2	2
ORL C, bit	Операция "ИЛИ" прямо адресуемого бита и переноса	2	2
ORL C, /bit	Операция "ИЛИ" инверсного прямо адресуемого бита и переноса	2	2
MOV C, bit	Перенести значение прямо адресуемого бита в перенос	2	2
MOV bit, C	Перенести значение переноса в прямо адресуемый бит	2	2
PROGRAM BRANCHING - ОПЕРАЦИИ ВЕТВЛЕНИЯ			
JC rel	Переход если перенос установлен	2	2/3
JNC rel	Переход если перенос не установлен	2	2/3
JB bit, rel	Переход если прямо адресуемый бит установлен	3	3/4
JNB bit, rel	Переход если прямо адресуемый бит не установлен	3	3/4
JBC bit, rel	Переход если прямо адресуемый бит установлен и очистка бита	3	3/4
ACALL addr11	Абсолютный короткий вызов подпрограммы	2	3
ACALL addr16	Абсолютный длинный вызов подпрограммы	3	4
RET	Возврат из подпрограммы	1	5
RETI	Возврат из прерывания	1	5
AJMP addrl1	Абсолютный средний переход	2	3
LJMP addi16	Абсолютный длинный переход	3	4
SJMP rel	Короткий переход (относительный адрес)	2	3
JMP @A+DPTR	Косвенный переход (с DPTR)	1	3
JZ rel	Переход если A == 0	2	2/3
JNZ rel	Переход если A ≠ 0	2	2/3
CJNE A, direct, rel	Переход если A ≠ прямо адресуемый байт	3	3/4
CJNE A, #data, rel	Переход если A ≠ данным	3	3/4
CJNE Rn, #data, rel	Переход если содержимое регистра ≠ данным	3	3/4
CJNE @Ri, #data, rel	Переход если косвенные данные ≠ данным	3	4/5
DJNZ Rn, rel	Уменьшить значение регистра и перейти, если ≠ 0	2	2/3
DJNZ direct, rel	Уменьшить значение прямо адресуемого байта и перейти, если ≠0	3	3/4
NOP	Нет операции	1	1

В приведенной таблице использованы следующие обозначения:

Rn - Регистры R0-R7 выбранного банка регистров;

@R1 - Косвенно адресуемые ячейки памяти через регистры R0-R1;

rel - 8-bit смещение к первому байту следующей инструкции. Используется в SJMP и всех остальных условных переходах;

direct - 8-bit прямо адресуемый адрес. Он используется для прямой адресации памяти данных (0x00-0x7F) или SFR регистров (0x80-0xFF);

#data - 8-bit константа;

#data16 - 16-bit константа;

bit - прямо адресуемый бит в памяти данных или SFR;

addr11 - 11-bit адрес, используемый инструкциями ACALL и AJMP. Адрес должен быть в пределах 2K страницы памяти программ;

addr16 - 16-bit адрес, используемый инструкциями LCALL и LJMP. Адрес может быть расположен где угодно, в пределах 64K адресного пространства программ.

Ядро имеет один не используемый код 0xA5, который воспринимается так же, как и инструкция NOP. Все мнемонические обозначения команд защищены - © Intel Corporation 1980.

Поскольку инструкции MCS-51 достаточно хорошо описаны в разнообразной технической литературе, в рамках данной книги они подробно не рассматриваются. Подробное описание инструкций дано, например, в [3].

2.3. Подсистемы ядра CIP-51

Как уже отмечалось выше, все микроконтроллеры имеют типовое ядро CIP-51 фирмы Cygnal с подсистемой отладки и программирования JTAG и набором инструкций, описанные в разделе 2.2. Однако ядро каждого из семейств микроконтроллеров имеет некоторые отличия, заключающиеся в количестве некоторых периферийных устройств, например, таймеров/счетчиков (3, 4 или5), последовательных портов и т.п. Различные микроконтроллеры выпускаются в различных типах корпусов с различным количеством выводов. Поэтому, в различных микроконтроллерах даже одного семейства может быть различное количество однобайтовых портов (от 1 до 5) или линий ввода/вывода (от 8 до 40).

Для обеспечения высокой работоспособности и легкости использования, ядро имеет дополнительные подсистемы обеспечения: развитый контроллер прерываний, мощная подсистема сброса и подсистема тактовых генераторов, подсистема управления питанием.

Подсистема прерываний. Расширенный контроллер прерываний ядра CIP-51 может обслужить достаточно большое количество источников прерывания (среднее - 22 источника прерываний), которое зависит от состава встроенной периферии семейства. Внутренняя периферия может генерировать до 12 прерываний. Еще до 10 внешних источников прерываний могут быть связанными с линиями портов ввода/вывода. Даже самые "слабые" из микроконтроллеров выгодно отличаются от стандартного ядра 8051, обслуживающего всего 7 источников. Наличие большого количества источников прерывания позволяет повысить общую производительность системы за счет обслуживания прерываний от многочисленных аналоговых узлов и освобождения мощности процессора для основной задачи. Очевидно, что это качество очень полезно для систем управления и контроля реального времени. Источники прерываний могут иметь два уровня приоритета. Каждый источник прерываний имеет один или несколько флагов (битов) в регистре специальных функций. При соблюдении сконфигурированных условий во внутренней периферии или на входных линиях внешних входов прерываний, соответствующий источнику прерываний флаг устанавливается в состояние логической единицы, или говорят просто "взводится". Взведение соответствующего флага прерывания вызывает собственно прерывание, если оно разрешено для соответствующего источника. Это, в свою очередь, вызовет генерацию инструкции LCALL на предварительно определенный адрес подпрограммы обработки прерывания соответствующего источника - ISR (Interrupt Service Routine) стазу же после завершения текущей инструкции. Каждая ISR должна завершиться инструкцией RETI, которая возвращает программу на инструкцию, следующую непосредственно за инструкцией, выполняемой в момент генерации прерывания. Если прерывание для соответствующего источника запрещено, соответствующий флаг прерывания игнорируется, а выполнение программы продолжается в нормальном режиме. Очевидно, что каждый источник прерывания может быть разрешен (запрещен) путем установки соответствующего разрешающего бита в регистре специальных функций SFR (IE-EIE2). Все прерывания также могут быть либо разрешены, либо запрещены соответствующей установкой бита EA (IE.7). Некоторые из флагов прерываний устанавливаются в 0 ("снимаются", "стираются") автоматически. Другие флаги необходимо снимать программно.

Внешние источники прерываний. Два внешних источника прерываний (INT0/ и INT1/) могут быть сконфигурированы на восприятие нулевого уровня (потенциала) или перепада с высокого в низкий уровень (задний фронт импульса) путем установки битов IT0 (TCON.0) и IT1 (TCON.2), при этом им соответствуют флаги IEO (TCON.l) и IE1 (TCON.3). Если эти прерывания настроены на восприятие фронта, соответствующие флаги стираются автоматически, как только ядро микроконтроллера сгенерирует переход на соответствующую ISR подпрограмму. Если же прерывания настроены на восприятие уровня, то соответствующий флаг повторяет состояние на соответствующем входе микроконтроллера. Необходимо учитывать, что если входной активный (низкий) потенциал прерывания не будет снят до завершения выполнения процедуры прерывания, эта процедура будет повторена вновь. Еще четыре внешних прерывания (External Interrupts 4-7) настроены на восприятие заднего фронта импульса (отрицательного перепада уровней с высокого в низкий). Соответствующие флаги этих прерываний находятся в регистре прерываний первого порта (Port 1 Interrupt Flag Register).

Каждый из источников прерываний может быть индивидуально запрограммирован на один из двух приоритетных уровней: низкий или высокий. Подпрограммы обработки прерываний низкого приоритета могут быть прерваны источниками прерываний с высоким приоритетом. Очевидно, что подпрограммы обработки прерываний с высоким приоритетом не могут быть прерваны. Каждый источник прерываний имеет индивидуальный бит приоритета в SFR (IP-EIP2), используемый для установки уровня приоритета. После сброса все приоритеты устанавливаются низкими. Если ядро одновременно восприняло два прерывания с разными приоритетами, прерывание с высоким приоритетом обрабатывается первым. Если

одновременно восприняты прерывания с одинаковым приоритетом, вступает в силу фиксированный ордер приоритетов (см. описание соответствующих регистров SFR).

Время реакции на прерывание зависит от состояния процессора в момент возникновения прерывания. Проверка наличия прерываний происходит в каждом такте генератора, а время выполнения инструкции LCALL равно четырем тактам, следовательно, минимально возможное время реакции на прерывание составляет 5 тактов. Если прерывание обнаружено в момент выполнения инструкции RETI, то после ее завершения выполняется следующая команда основной программы, а лишь затем генерируется инструкция LCALL. Следовательно, максимальное время реакции на прерывание будет в том случае, когда прерывание обнаружено при выполнении инструкции RETI, а следом за ней идет самая "длинная" (по времени) инструкция DIV. В этом случае время реакции составит 18 тактов: 1 на обнаружение прерывания, 5 тактов на выполнение инструкции RETI, 8 тактов на выполнение инструкции DIV и 4 такта на выполнение LCALL. Если во время выполнения подпрограммы обработки прерывания пришло следующее прерывание с равным приоритетом, это новое прерывание будет обслужено только после завершения выполнения текущей подпрограммы, включая инструкцию возврата RETI, и следующей инструкции основной программы.

Подсистемы сброса и тактовых генераторов. Подсистемы сброса и тактовых генераторов показаны на рис.2.3.

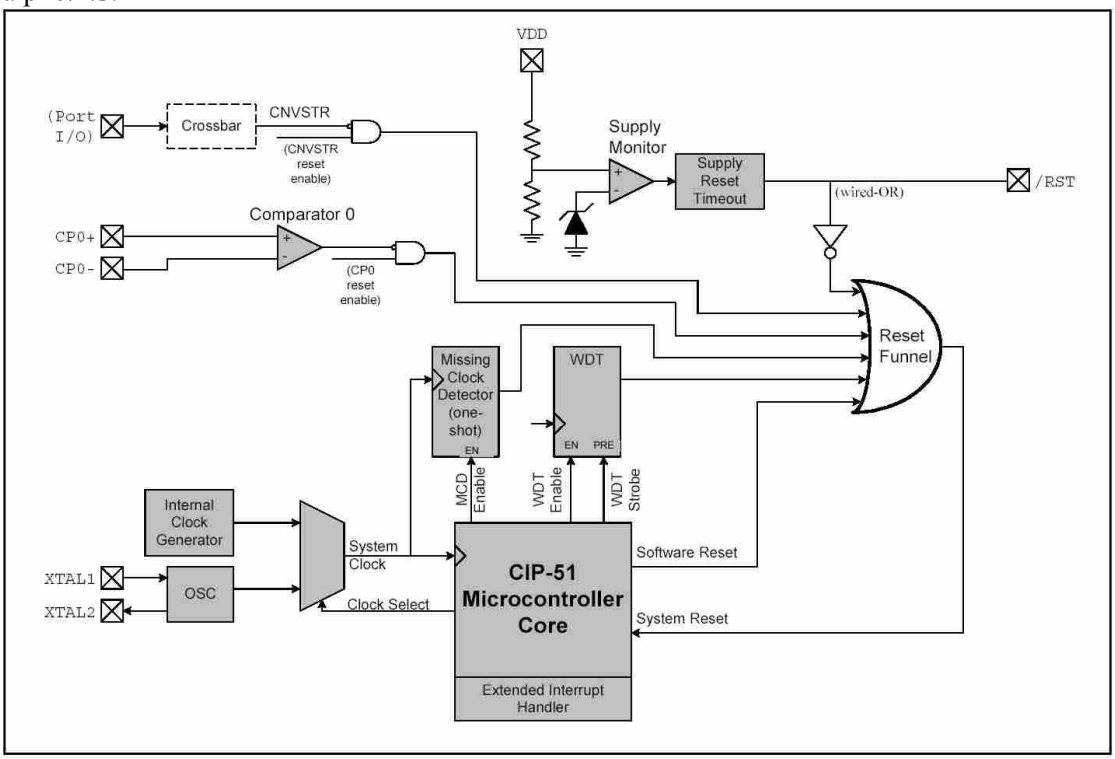

Рис.2.3. Подсистемы сброса и тактовых генераторов.

Подсистема сброса обеспечивает сброс микроконтроллерного ядра в зависимости от семи различных причин: от встроенного монитора питания, который вырабатывает сигнал сброса при значительном снижении уровня питания; от охранного таймера WDT (Watchdog Timer); при отсутствии тактовых импульсов на детекторе тактовых импульсов; по сигналу от компаратора 0; при наличии программного сброса; по сигналу со входов CNVSTR или RST/. Вход сброса RST/ - двунаправленный, что позволяет осуществлять внешний сброс. Каждый из перечисленных причин сброса, кроме входа сброса RST/ и монитора питания, может быть запрещен программно. Охранный таймер WDT при сбросе системы становится активны, что нужно учитывать при разработке программного обеспечения (его нужно либо запретить на стадии инициализации, либо учитывать его время срабатывания).

При возникновении состояния сброса ядро останавливает выполнение программы, переводит все выводы микроконтроллера в начальное состояние, устанавливает в начальное состояние регистры специальных функций, восстанавливает начальное состояние таймеров и прерываний, сбрасывает счетчик адреса в состояние 0x0000 и начинает выполнение программы с этого адреса. Во все порты ввода/вывода записывается код 0xFF, включаются внутренние подтяжки уровней. Если состояние

сброса возникло от монитора питания или записи 1 в PORSF, вывод RST/ переводится в состояние 0 и удерживается в этом состоянии определенное время, необходимое для сброса. После сброса внутренний генератор запускается на частоте 2MHz.. Охранный таймер WDT устанавливается на максимально возможное время.

Как мы уже отмечали выше, существует семь причин возникновения ситуации сброса. Рассмотрим их подробнее:

1. При превышении определенного уровня напряжения питания VRS_T, указанного в электрических характеристиках каждого семейства, вырабатывается сигнал сброса после включения питания (Power-on Reset) RST/. Во время этого режима сброса флаг PORSF (RSTSRC.l) аппаратно устанавливается в логическую единицу. Все другие флаги в регистре RSTSRC не определены. Флаг PORSF обнуляется при сбросе от любых других источников. Поскольку все причины сброса заставляют микроконтроллер начинать выполнять программу с одного нулевого стартового адреса, то только опросом флага PORSF микроконтроллер может определить, что сброс был вызван причиной включения питания. Это необходимо для определения состояния памяти. Понятно, что если был установлен флаг PORSF - память содержит произвольную информацию.
2. Другой причиной сброса может быть программная запись флага PORSF (RSTSRC.l) в логическую единицу.
3. В случае даже кратковременного снижения напряжения питания ниже уровня VRS_T, монитором вырабатывается сигнал сброса от нарушения питания (Power-fail Reset) с установкой флага PORSF.
4. Четвертой причиной может служить внешний сигнал сброса, который подается нулевым уровнем на вывод RST/. Длительность импульса сброса должна быть больше 12 тактов. После выхода из этого режима устанавливается флаг PINRSF (RSTSRC.0). Напомним, что вывод RST/ совместим с пяти вольтовыми уровнями.
5. Еще одной причиной возникновения режима сброса (Missing Clock Detector Reset) является отсутствие в течение более чем 100μs тактовых импульсов. После сброса устанавливается флаг MCDRSF (RSTSRC.2), который позволяет определить причину сброса. Этот источник сброса может быть включен или выключен соответствующей установкой бита MSCLKE регистра OSCICN.
6. Еще одним источником состояния сброса может быть компаратор 0, который может быть запрограммирован на выработку сигнала сброса низкого уровня и запись высокого уровня в флаг CORSEF (RSTSRC.5). Конечно, компаратор 0 должен быть предварительно разрешен битом CPTOCN.7. При этом, если на неинвертирующем входе (CPO+) напряжение ниже, чем на инверсном входе (CPO-), генерируется сигнал сброса (Comparator 0 Reset). Следует помнить, что при этом режиме сброса, импульс на выводе RST/ не генерируется. Следует заметить, что этот сигнал сброса вырабатывается и при активном тактовом генераторе, и в отсутствии тактовых импульсов.
7. Последним источником ситуации сброса может быть внешний сигнал CNVSTR, который может быть запрограммирован, как вход сброса, чувствительный к нулевому потенциалу, путем записи 1 в флаг CNVRSEF (RSTSRC.6). Этот сигнал может быть выведен на один из вводов первых трех портов P0, P1 и P2 через коммутатор ресурсов Crossbar. После этого типа сброса (External CNVSTR Pin Reset) устанавливается флаг CNVRSEF (RSTSRC.6). При этом режиме сброса импульс на выводе RST/ также не генерируется.

Охранный таймер (Watchdog Timer). Охранный таймер WDT также может являться источником сигнала сброса, если происходит его переполнение. Чтобы избежать состояния сброса, WDT должен периодически перезапускаться программным обеспечением. Охранный таймер WDT автоматически запускается с максимальным периодом перезапуска после сброса микроконтроллера. Если в использовании WDT нет необходимости, он должен быть заперт. Если WDT был заперт, он не может быть разрешен до следующего сброса. Сброс от охранного таймера не изменяет состояние вывода RST/.

Охранный таймер WDT состоит из 21 битного таймера, работающего от системного генератора. Таймер измеряет период между специальной записью в управляющий регистр. Если этот период превысит запрограммированный лимит, произойдет сброс. Вообще говоря, WDT включаться и выключаться по необходимости программным путем. Управление производится через специальный регистр (Watchdog Timer Control Register -WDTCN). Перезапуск WDT производится записью кода 0xA5 в регистр WDTCN. Запрет WDT производится последовательной записью байтов 0xDE и 0xAD в WDTCN, как показано в следующем фрагменте:

```
CLR   EA              ; disable all interrupts
MOV   WDTCN, #ODEh    ; disable software
MOV   WDTCN, #OADh    ; watchdog timer
SETB  EA              ; re-enable interrupts
```

Запись байтов 0xDE и 0xAD занимает 4 такта. Во время этих записей недопустимы задержки, поэтому все прерывания должны быть запрещены, как показано выше. Как уже упоминалось выше, WDT может запираться записью кода 0xFF в WDTCN. После запирания, попытки выключения WDT игнорируются до следующего сброса. Прикладное программное обеспечение, предполагающее всегда использовать WDT, должно запирать его записью 0xFF во время инициализации. Полный интервал работы WDT задается соответствующей установкой битов регистра WDTCN.[2:0] и описывается следующим соотношением:

$$4^{3+\,WDTCN.[2:0]} * T_{SYSCLK}, \quad \textit{где } T_{SYSCLK} \textit{ - период тактовых импульсов.}$$

При тактовой частоте 2MHz этот период составляет от 0.032 ms до 524 ms. Следует также помнить, что при установке интервала в биты WDTCN.[2:0], бит WDTCN.7 должен быть равен 0. Чтение этого регистра позволяет прочитать заданный интервал. После сброса значения всех битов равны 1.

Многофункциональный генератор. Встроенный многофункциональный генератор, который начинает функционировать сразу при сбросе микроконтроллера на частоте 2 МГц. При необходимости, можно переключить тактирование микроконтроллера в режим генератора с внешним частотозадающим элементом, в качестве которого может использоваться кварцевый или керамический резонатор, конденсатор, RC цепочка или внешний источник тактовых импульсов. Переключение может осуществляться «на лету», при этом работа микроконтроллера не нарушается. Это позволяет работать на низких частотах (при малом потреблении) и переключать тактирование на высокие частоты только в случае необходимости, например, по сигналу от таймера для обработки сигналов от внешних датчиков. Встроенный тактовый генератор может быть также переключен программно у большинства микроконтроллеров до частоты 16 МГц, а у некоторых из микроконтроллеров до 24 Мгц.

Многофункциональный генератор состоит из внутреннего и внешнего генераторов, каждый из которых может служить частотозадающим. После сброса всегда включается внутренний генератор на частоте 2MHz. Внутренний генератор может быть разрешен или запрещен или его частота может быть изменена путем установки соответствующих значений регистра OSCICN (Internal Oscillator Control Register).

Во время действия состояния сброса оба (внутренний и внешний) генераторы заторможены. Ядро может начать работать на одной из частот внутреннего или внешнего генераторов, определяемой битом CLKSL в регистре OSCICN. Внешний генератор предполагает наличие внешнего резонатора (кварцевого или пьезокерамического), конденсатора или RC цепочки, соединенной между выводами XTAL1 и XTAL2. Внешний генератор должен быть также сконфигурирован с использованием OSCXCN регистра. Кроме того, конечно же может быть использован внешний автономный генератор, выход которого может быть подан на вывод XTAL1. *Оба вывода* XTAL1 и XTAL2 рассчитаны на 3.6V и не могут работать с 5V логикой! Очевидно, также, что прежде чем переключить ядро на работу от внешнего генератора, его необходимо разрешить и запустить.

Некоторые очевидные рекомендации. Цепи подключения кварцевых резонаторов очень чувствительны ко всевозможным наводкам и помехам, поэтому эти цепи должны быть минимальной длины, т.е. резонатор должен располагаться как можно ближе к микроконтроллеру. При использовании наиболее распространенного кварцевого резонатора 11.0592MHz, обычно применяемого с другими x51 совместимыми контроллерами, необходимо оба вывода резонатора зашунтировать на землю конденсаторами с емкостью от 7 pF до 33 pF. Следует особо отметить, что на запуск и стабилизацию режима генератора требуется определенное время, как минимум - 1 ms, как следствие, после установки бита разрешения генерации необходимо сделать соответствующую задержку перед записью бита XTLVLD. Отсутствие задержки может привести к непредсказуемым результатам.

В качестве времязадающей цепочки может быть использована обычная RC цепочка. При этом следует соблюдать определенные рекомендации. Конденсатор должен быть не более 100 pF, но и не менее ~7 pF. Очевидно, что необходимо установить бит разрешения такого режима XFCN (External Oscillator Frequency Control) в регистре OSCXCN. Расчет частоты генератора можно произвести по формуле:

$$f = 1.23*10^3/RC$$

Допустим, мы установили конденсатор C=50pF и резистор R=246K, тогда частота генератора составит:

$$f = 1.23*10^3/RC = 1.23*10^3 / [246 * 50] = 0.1MHz = 100kHz$$

При этом необходимо подсчитать величину XFCN:

XFCN > $\log_2(f/25kHz)= \log_2(100kHz/25kHz) = Iog_2(4)= 2$, т.е. необходимо установить код 010.

Встроенный генератор может работать не только с RC цепочкой, но и с одним только конденсатором, который также должен быть не более 100 pF. Определение частоты производится по формуле:

$$f = KF / (C * VDD)$$

однако коэффициент KF выбирается из специальной таблицы. Механизм подбора коэффициента и определения частоты будет изложен в следующих главах.

Подсистема управления питанием. Ядро CIP-51 имеет два программно-управляемых режима энергосбережения: Idle и Stop. Режим Idle заключается в приостановке ядра, но при этом остаются активными внутренний тактовый генератор и периферия. В режиме Stop приостанавливается не только ядро, но и таймеры (за исключением детектора обнаружения отсутствия тактирования), система прерываний и тактовый генератор. Очевидно, что в режиме Idle потребление больше, чем в режиме Stop. Управление режимами энергосбережения

Осуществляется через регистр PCON (Power Control Register). Еще одним способом понижения энергопотребления является индивидуальное отключение не функционирующей в данный момент периферии. Это позволяет эффективнее осуществлять функции энергосбережения, по сравнению с режимом Stop, для выхода из которого необходимо тратить определенное время на режим сброса.

Режим Idle. Включение режима Idle осуществляется установкой бита PCON.0 (Idle Mode Select bit). Этот режим длится до тех пор, пока инструкция установки бита не завершилась. При этом, содержимое всех внутренних регистров и памяти сохраняется. Любая аналоговая и цифровая периферия может оставаться активной в течение этого режима. Выход их режима Idle возможен либо по одному из разрешенных прерываний, либо по сигналу сброса RST/, либо по истечении времени WDT. При выработке прерывания, первая же команда подпрограммы обработки прерываний должна очистить бит PCON.0 для возобновления работы процессора. Если разрешена работа охранного таймера WDT, то он также может вывести процессор из состояния Idle.

Режим Stop. Установка режима Stop осуществляется установкой бита PCON.l (Stop Mode Select bit). Выход из этого режима возможен только по сигналу сброса (внутреннего или внешнего).

2.4. Встроенная память

Ядро CIP-51 имеет стандартное адресное пространство - 64K и стандартную конфигурацию адресов программ и данных. Ядро оснащено встроенной памятью данных с произвольным доступом - RAM (Random Access Memory) объемом 256 байт (0x00-0xFF).

Младшие 128 байт (0x00-0x7F) доступны инструкциям с прямой и косвенной адресацией, регистры специальных функций SFR (Special Function Register's) доступны только инструкциям с прямой адресацией, а старшие 128 байт (0x80-0xFF) - только инструкциям с косвенной адресацией. Первые 32 байта (0x00-0x1F) адресуются как четыре банка регистров общего назначения, а следующие 16 байт (0x20-0x2F) - имеют битовую адресацию. Некоторые из микроконтроллеров могут иметь дополнительную память во внешнем адресном пространстве памяти данных. Размер этой памяти может быть различен для различных микроконтроллеров. Эта память может быть доступна с помощью инструкции MOVX.

Каждый из четырех банков регистров общего назначения состоит из восьми регистров R0-R7. Только один из банков может быть разрешен в текущий момент времени. Переключение банков осуществляется с помощью битов RS0 (PSW.3) и RS1 (PSW.4).

Программный стек также располагается в основных 256 байтах оперативной памяти. Область стека также задается соответствующим значением регистра SP (Stack Pointer - 0x81 SFR).

Каждое семейство микроконтроллеров имеет индивидуальную карту встроенной памяти, которые будут приведены в соответствующих главах.

Flash память программ различных микроконтроллеров также имеет различный объем и расположение, что также будет показано на соответствующих картах памяти. Основная Flash память программ расположена с адреса 0x0000. Чистая Flash память (после стирания) содержит код 0xFF во всех байтах. При записи каждого байта, его стирание перед записью производится автоматически. Ожидания или определения завершения операции записи/стирания также не требуется. Память рассчитана на как минимум на 20000 циклов записи/стирания.

Вообще говоря, термин "программная память" подразумевает память, доступную только для чтения. Однако, ядро CIP-51 позволяет и писать в эту память при установке разрешающего бита PSCTL.0 (Program Store Write Enable Bit). Запись осуществляется инструкцией MOVX. Этот механизм позволяет обновлять программный код и использовать память программ для записи редко изменяющихся данных и различных настроечных параметров.

Программирование Flash памяти осуществляется через последовательный JTAG интерфейс, управляемый утилитами программирования фирмы Cygnal. Напомним, что операция записи может только очистить какой-либо бит, т.е. записать логический 0. Запись единицы невозможна, поэтому восстановление единиц возможно только операцией стирания. Из этого следует, что прежде чем записать новый байт в память, старый байт необходимо стереть. Flash память, как правило, имеет секторную организацию. Стирание производится посекторно, при этом все байты сектора устанавливаются в 0xFF. Естественно, что при такой операции стирания, записывать также надо целый сектор, что бывает не очень удобно.

Однако есть и другой, более удобный путь. Если установить PSEE (Program Store Erase Enable) бит (PSCTL.l) и PSWE бит (PSCTL.0) в логическую единицу, то становится возможным использование инструкции MOVX для записи только одного байта (с предварительным автоматическим стиранием), а не целого сектора. Приведем типовую последовательность алгоритма программирования.

1. Разрешить операции записи/стирания Flash памяти в регистре FLSCL установкой бита FLASCL.
2. Установить бит PSEE (PSCTL.l) для разрешения стирания сектора Flash памяти.
3. Установить бит PSWE (PSCTL.0) для разрешения операции записи во Flash.
4. Использовать инструкцию MOVX для записи любого байта по произвольному адресу без стирания 512-байтного сектора.
5. Стереть бит PSEE для запрета стирания сектора Flash памяти.
6. Использовать инструкцию MOVX для записи любого байта по выбранному адресу без стирания 512-байтного сектора. Повторять эту операцию для любого количества байт от одного до целого сектора.
7. Стереть бит PSWE для запрета операций Flash записи.

Время записи/стирания автоматически контролируется аппаратно на основе данных, установленных в регистре FLSCL (Flash Memory Timing Prescaler). Четыре бита регистра FLASCL определяют временной интервал операций записи/стирания.

Типовые электрические параметры Flash памяти при номинальном напряжении питания от 2,7 до 3,6V в диапазоне температур от -40°C до +85°C следующие:

1. Количество циклов запись/стирание не менее 20000 (обычно 100000).
2. Время цикла стирания (сектора), не более 10 ms.
3. Время цикла записи байта, не более 40 μs.

Как уже отмечалось выше, Flash память может использоваться не только по прямому назначению, т.е. для хранения кодов программ, но и для энергонезависимого хранения данных. При этом, данные могут быть записаны инструкцией MOVX и прочитаны инструкцией MOVC. Многие семейства микроконтроллеров имеют Flash память, выполненную в виде двух блоков: одного основного большого, начинающегося с адреса 0x0000, и одного дополнительного маленького, обычно размером 128 байт. Следует напомнить, что хотя каждый байт и может быть записан индивидуально, но сектор может быть стерт только полностью. Т.о., чтобы заменить один единственный байт, все данные из сектора должны быть перенесены во временное хранилище в оперативной памяти, затем данные должны быть модифицированы, затем сектор должен быть стерт, а затем, обновленные данные должны быть записаны в этот сектор. Нормальный размер сектора - 512 байт, слишком велик для временного хранения во внутренней оперативной памяти объемом 256 байтов. Поэтому, обычно используется специальный уменьшенный сектор размером 128 байт.

Ядро CIP-51 имеет многоуровневую защиту Flash памяти. Биты PSCTL.0 (Program Store Write Enable) и PSCTL.l (Program Store Erase Enable) защищают Flash память от случайной модифи-

кации программой. Эти биты должны быть предварительно установлены в 1 для того, чтобы разрешить программе модифицировать Flash память.

Имеются также несколько байтов защиты, каждый бит которых защищает блок памяти программ определенного размера от операций чтения, минуя JTAG интерфейс. Таким образом, одни биты защищают память от записи/стирания, другие - от чтения. Блок Flash памяти, содержащий байты защиты, разрешен для записи, но запрещен для стирания программным обеспечением.

Итак, байт FRLB (FLASH Read Lock Byte) содержит биты защиты памяти от чтения через JTAG интерфейс. Логический 0 в соответствующем бите запирает соответствующий блок памяти и делает его недоступным для чтения через JTAG интерфейс. Запись в бит логической единицы разрешает чтение соответствующего блока.

Байт FWELB (FLASH Write/Erase Lock Byte) содержит биты защиты памяти от записи через JTAG интерфейс. Логический 0 блокирует запись, логическая 1 - разрешает запись.

Еще один тип защиты содержится в регистре FLACL (FLASH Access Limit Register). Этот регистр содержит старший байт 16-битного адреса, ниже которого программное чтение запрещено. Все попытки чтения этих адресов возвращают код 0x00. Таким образом, возможна многоступенчатая защита памяти, поскольку блоки, содержащие байты защиты могут быть, в свою очередь заперты. Полное стирание всей защиты возможно только при полном стирании всей программной Flash памяти через JTAG программатор и невозможно из пользовательской программы.

Обратите внимание! Адресация любого защищенного байта при выполнении JTAG операций записи вызывает автоматическое стирание всей памяти программ. Это стирание возможно только через JTAG. Адресация незащищенного байта вызовет стирание только одной страницы или блока, включая и байты безопасности.

Многие из микроконтроллеров фирмы Cygnal имеют дополнительное адресное пространство внешней памяти XRAM (External RAM) различного объема. Это адресное пространство доступно инструкциями MOVX. Если инструкция используется с восьмибитным операндом адреса, например @R1, то старший байт шестнадцатибитного адреса определяется содержимым регистра EMIOCN (External Memory Interface Control Register). При доступе к XRAM должен быть установлен бит PSTCL.0=0. При всех способах доступа к XRAM старшие биты шестнадцатибитного адреса не изменяются. Как следствие, блок внешней памяти XRAM доступен во всей области адресного пространства 64К с начальным смещение, кратным размеру блока. Например, блок с объемом 2048 байт буде доступен по адресам 0x0000 также как и 0x0800, 0x1000, 0x1800, 0x2000, и т.п.

Регистры специальных функций (SFR). Как уже отмечалось выше, управление всеми периферийными узлами и обмен информацией с ними осуществляется ядром CIP-51 через соответствующие SFR регистры. Эти регистры специальных функций (SFRs) расположены в прямо адресуемом адресном пространстве памяти данных в диапазоне адресов 0x80-0xFF.

Регистры специальных функций ядра CIP-51 полностью соответствуют регистрам стандартного 8051, т.е. все стандартные (для 8051) регистры, их назначение, название и адреса сохранены. С другой стороны, регистры специальных функций дополнены возможностями конфигурирования и обмена данными с оригинальными подсистемами микроконтроллеров Cygnal. Очевидно, что каждое из семейств микроконтроллеров, имеющих различный состав периферии, имеет оригинальный набор дополнительных SFR регистров. Для каждого из семейств существует своя уникальная SFR карта. Все SFR карты с их описаниями будут приведены в соответствующих главах книги.

Регистры SFR доступны для инструкций с прямой адресацией в пространстве адресов от 0x80 до 0xFF. Регистры с адресами, оканчивающимися на 0x0 или 0x8 (например, P0, TCON, P1, SCON, IE и т.д.) являются и битадресуемыми и байтадресуемыми. Все остальные регистры SFR являются только байтадресуемыми. Неиспользуемые адреса в адресном пространстве SFR зарезервированы для развития. Использование этих адресов приводит к неопределенному результату и нежелательно.

Подробное описание SFR регистров буде приведено для каждого семейства микроконтроллеров в соответствующих главах.

2.5. Программируемые линии (порты) ввода/вывода и коммутатор ресурсов Crossbar

Микроконтроллеры различных семейств имеют различное количество стандартных байтных портов ввода/вывода (от 1 до 8). Напомним, что стандартный микроконтроллер 8051 имеет четыре порта (Ports 0,1,2 и 3). Порты, не имеющие внешних выводов (ввиду нехватки физических выводов корпусов), могут быть использованы, как регистры общего назначения. Порты ввода/вывода ведут себя так же, как порты стандартного 8051, но имеют дополнительные улучшающие функциональные качества.

Каждая линия порт ввода/вывода может быть настроена как нормальная входная или выходная линия, как линия с "третьим" высокоимпедансным состоянием или как линия с "открытым истоком" через соответствующие регистры (Port Configuration Registers) PRT0CF, PRT1CF, PRT2CF, PRT3CF... Плюс к этому, подтяжки уровней "Pull-Up", обычно используемые в стандартных 8051 микроконтроллерах, могут быть программно запрещены с целью дополнительного энергосбережения.

Несомненным достижением архитектуры микроконтроллеров фирмы Cygnal является наличие функционального узла, называемого "Crossbar", выполняющего роль программно-управляемого коммутационного узла, переключающего периферийные внутренние ресурсы на линии ввода/вывода портов 0,1 и 2. Иными словами, узел "Crossbar" - это коммутатор внутренних ресурсов. Это новшество значительно повышает гибкость разработки микроконтроллерной системы. Внутренние периферийные узлы, такие как таймеры, последовательные интерфейсы, прерывания, входы запуска ADC, выходы компараторов и другие цифровые сигналы, а в некоторых семействах и аналоговые входы ADC, могут быть программно назначены на выводы вышеуказанных портов. Спецификация портов осуществляется путем программирования специальных регистров XBR0, XBR1 и XBR2 и так называемой таблицы приоритетной перекодировки (Priority Decode Table). Рис.2.4 иллюстрирует функциональное назначение *Crossbar (на примере первых двух семейств C8051F0xx и C8051F018-9)* .

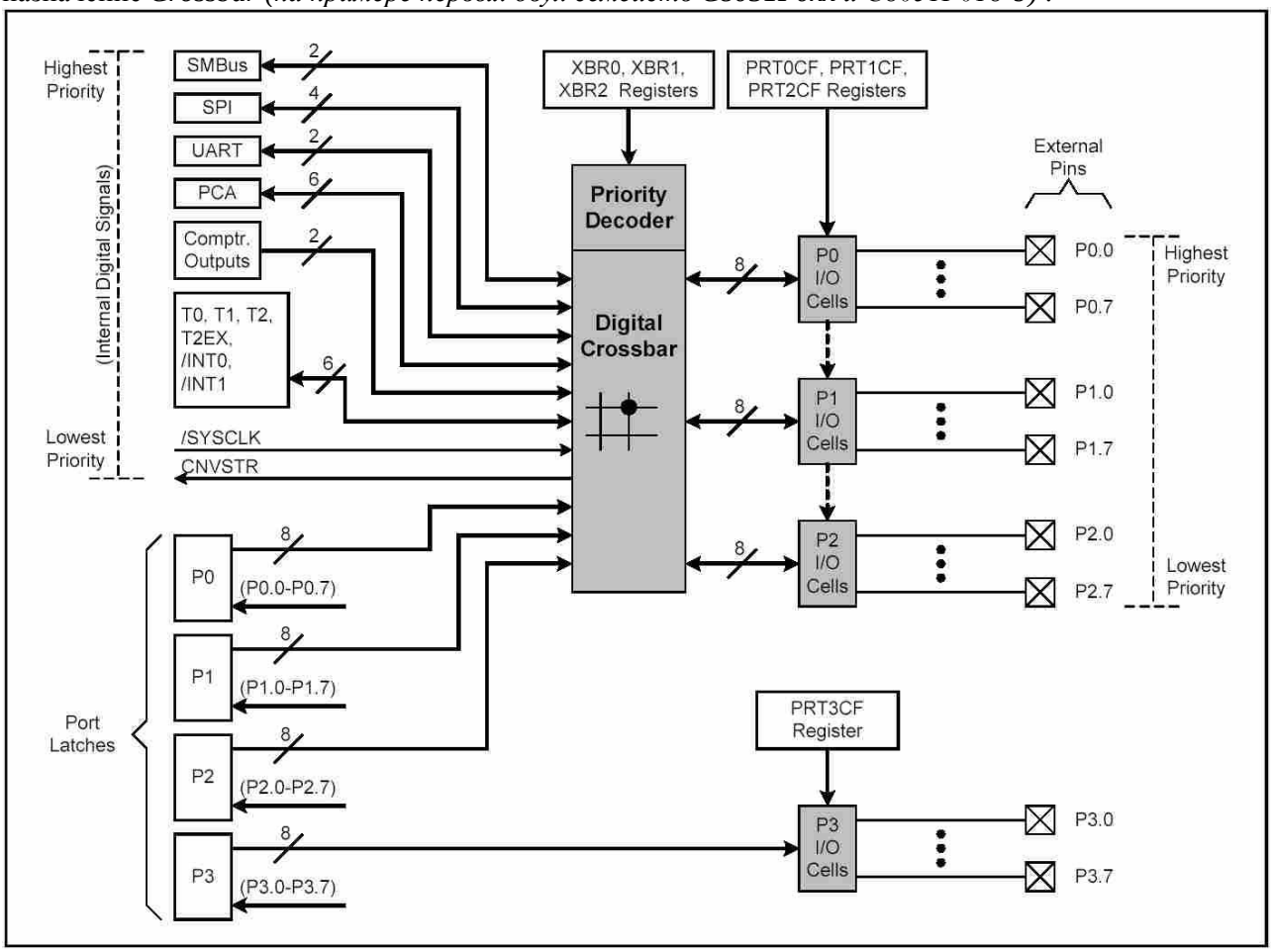

Рис.2.4. CrossBar -коммутатор ресурсов *(на примере семейств C8051F0xx и C8051F018-9)*

Приоритетный дешифратор коммутатора ресурсов (Priority CrossBar Decoder). Главной задачей разработчиков коммутатора ресурсов CrossBar являлось создание гибкой и удобной архитектуры, позволяющей проектировщику контроллеров оперативно выбирать и перераспределять богатые внутренние аналоговые и цифровые ресурсы на ограниченное число внешних выводов корпуса. Таблица приоритетного дешифратора коммутатора ресурсов индивидуальна для каждого семейства микроконтроллеров ввиду различного состава внутренней периферии. Однако, методика выбора периферии одинакова для всех семейств. На рис.2.5. в качестве примера, показана таблица приоритетов дешифратора коммутатора ресурсов C8051F0xx и C8051F018-9 семейств.

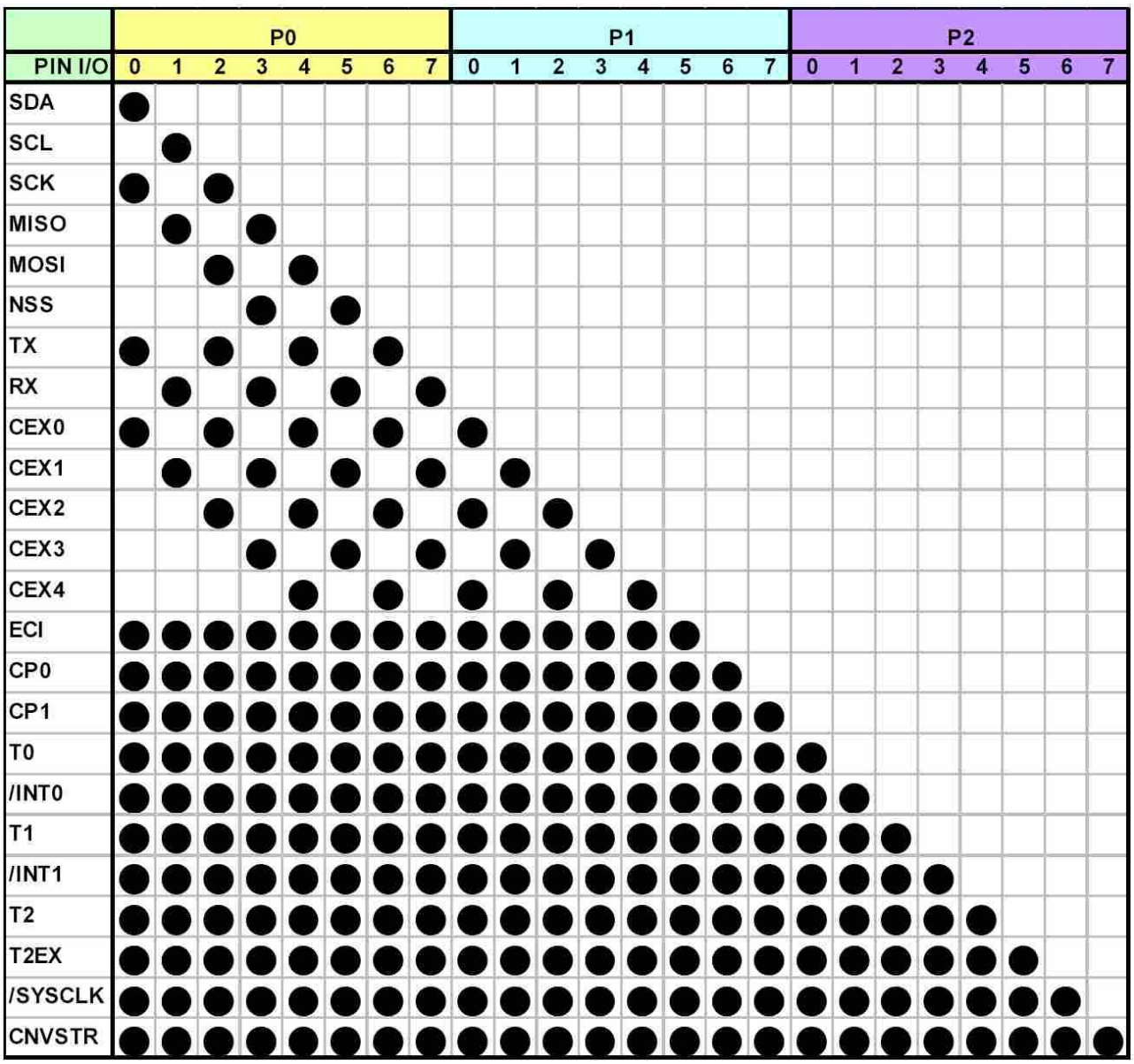

Рис.2.5. Пример приоритетного дешифратора коммутатора ресурсов

В таблице по вертикали перечислены выводы всех периферийных устройств, которые могут быть выведены на вводы/вывода портов P0...P2. Выводы периферийных устройств перечислены в порядке убывания приоритета. Например, наивысший приоритет имеет интерфейс SMBus, имеющий сигналы SDA и SCL, затем следует интерфейс SPI, имеющий сигналы SCK, MISO, MOSI и NSS, затем последовательный порт UART с сигналами TX и RX, затем программируемый массив-счетчик PCA, имеющий выходы CEX0,…,CEX4, и т.д.

Как уже было сказано выше, выбор периферийных устройств осуществляется путем программирования специальных регистров XBR0, XBR1 и XBR2. Эти регистры содержат бит (или биты), определяющие активность каждого из приведенных на рис.2.5. периферийных устройств. Например, включение в состав или выключение из состава периферийных устройств интерфейсов SMBus, SPI и UART осуществляется установкой или стиранием одного бита (для каждого интерфейса). Для программируемого массива - счетчика PCA имеется три бита, определяющих количество включаемых каналов.

Методика определения активной периферии и соотнесение ее к конкретным портам ввода/вывода следующие:

1. Проектировщик в регистрах XBR0, XBR1 и XBR2 устанавливает в логическую единицу биты, отвечающие за активность тех или иных периферийных устройств. Биты неиспользуемых периферийных устройств устанавливаются в состояние логического 0.
2. Назначение выводов активных периферийных устройств всегда начинается с младших битов порта P0. Далее, при полном заполнении порта P0, начинается заполнение порта P1, а если его не хватит, то и P2. Заполнение происходит в соответствии с приоритетом со сдвигом в сторону свободных

младших выводов. Например, если устройство SMBus активно, то его сигналы занимают младшие разряды порта P0: SDA - P0.0; SCL - P0.1. Если же устройство SMBus пассивно (выключено), а следующее активное устройство - SPI, то его сигналы занимают младшие разряды: SCK - P0.0; MISO - P0.1; MOSI- P0.2; NSS - P0.3. Если и устройство SPI пассивно, а активно устройство UART, то его сигналы займут младшие разряды: TX - P0.0; RX - P0.1.

3. Следующие активные устройства всегда занимают более младшие свободные выводы портов P0...P2. Например, если первым активным устройством был интерфейс SPI, и его сигналы заняли младшие разряды: SCK - P0.0; MISO - P0.1; MOSI- P0.2; NSS - P0.3, а следующее активное устройство - PCA с заданным одним каналом, его канал буде назначен на первый младший свободный вывод: CEX0 - P0.4.
4. Выводы, оставшиеся свободными после назначения активных периферийных устройст могут быть использованы, как программируемые вводы/выводы общего назначения.
5. После назначения активных периферийных устройств необходимо разрешить работу (установить активность) CrossBar путем установки бита XBARE в регистре XBR2. *Пока CrossBar не активна, все линии портов остаются в режиме ввода вне зависимости от назначений регистров XBRn.*

Порты ввода/вывода. Выходные характеристики портов ввода/вывода устанавливаются в регистрах конфигурации портов (Port Configuration Registers): PRTOCF, PRT1CF, PRT2CF и PRT3CF. Каждый вывод может быть назначен как вывод "с открытым истоком" или как трехстабильный вывод. Такое назначение требуется даже для линий ввода/вывода интерфейсов, назначенных в регистрах XBRn. Следует помнить, что только линии SDA и SCL интерфейса SMBus и линия RX в режиме 0 интерфейса UART автоматически устанавливаются как линии "с открытым истоком". Бит WEAKPUD в регистре XBR2 отвечает за "слабую подтяжку" выходного уровня, если этот бит установлен в 0, то "слабая подтяжка" выходов осуществляется для всех линий портов, установленных а режим "с открытым истоком". С другой стороны, этот бит не действует на линии ввода/вывода, установленные как трехстабильные. Кроме того, "слабая подтяжка" выключается для выводов, находящихся в состоянии логического нуля, для исключения бесполезных потерь мощности. На рис.2.6. показана функциональная схема одной линии порта.

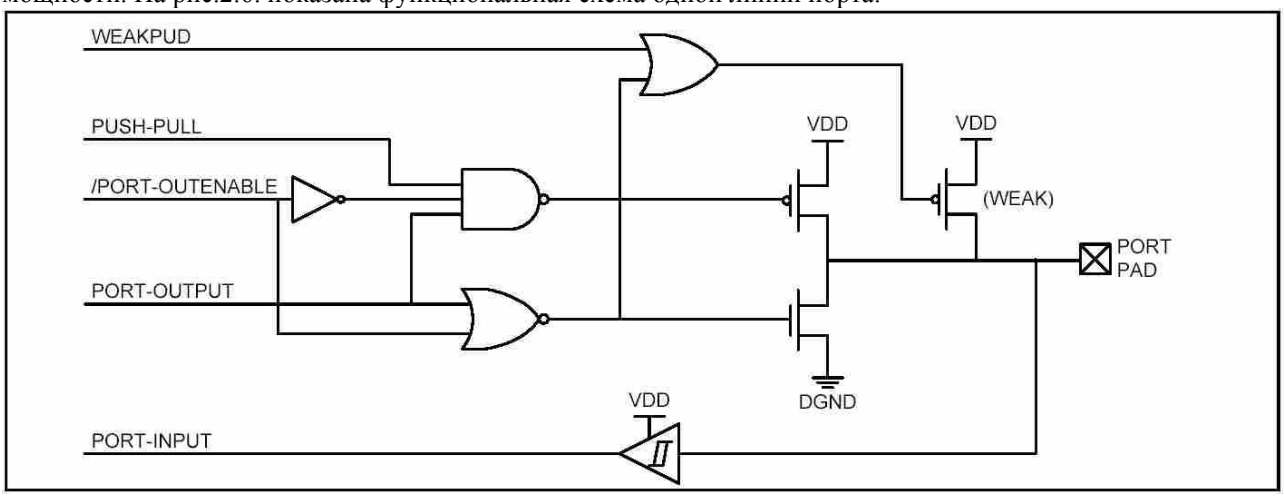

Рис.2.6. Функциональная схема линии порта.

Как видно из рассмотрения этой функциональной схемы, так называемая "слабая подтяжка" (weak) представляет собой слегка приоткрытый полевой транзистор с высоким сопротивлением канала в открытом состоянии.

Микроконтроллеры фирмы Cygnal могут иметь от 1 до 5 однобайтных портов ввода/вывода (в зависимости от типа корпуса), которые могут использоваться для назначения линий ввода/вывода периферийных встроенных устройств или как порты ввода/вывода общего назначения. Обращение к каждому такому порту осуществляется через соответствующий регистр SFR (P0, P1, …), доступный как побайтно, так и побитно. При записи в порт, записанное значение защелкивается и выводится на соответствующие выводы. При чтении, логические уровни порта могут быть прочитаны независимо от назначений, произведенных в регистрах XBRn. Исключением являются действия с инструкциями, выполняющими "чтение - модификацию - запись" (*read-modify-write):* ANL, ORL, XRL, JBC, CPL, INC, DEC, DJNZ и MOV, CLR или SET, при которых адресуется непосредственный бит в порту SFR.

При выполнении этих инструкций состояние регистра порта (а не состояние выводов) читается, модифицируется и записывается обратно в регистр.

В некоторых микроконтроллерах, имеющих корпуса с ограниченным количеством выводов, некоторые из портов могут не иметь своих выводов. Например, порты P2 и P3 не имеют выводов в микроконтроллерах F001/06/11/16, порты PI, P2 и P3 не имеют выводов в микроконтроллерах F002/07/12/17. Однако эти порты остаются доступными для ядра CIP-51. В этом случае, разработчику следует заботиться самому о корректном написании программы.

2.6. Программируемый массив-счетчик PCA

В дополнение к стандартным таймерам/счетчикам, микроконтроллеры Cygnal имеют еще один оригинальный узел - программируемый массив-счетчик - PCA (Programmable Counter/Timer Array). Он состоит из специализированного шестнадцатибитного таймера/счетчика с времязадающим узлом (TimeBase) и пятью модулями захвата / сравнения - CCM (Capture/Compare Module). Времязадающий узел (TimeBase) может подавать на вход PCA одну из четырех частот: системную частоту тактирования, деленную на 12 или 4, выход переполнения таймера 0 или внешний счетный вход - ECI (External Clock Input). Каждый из пяти модулей захвата / сравнения CCM может быть запрограммирован на выполнение одной из четырех функций: переключаемая по фронту защелка, программный счетчик, высокоскоростной выход и широто-импульсный модулятор - PWM (Pulse Width Modulator). Естественно, что внешний счетный вход и выходы модулей захвата / сравнения CCM могут быть настроены с помощью CrossBar на необходимые выводы портов ввода/вывода или заданы, как источник прерывания. Кроме того, в некоторых из микроконтроллеров на базе расширенного PCA может быть реализован таймер реального времени - RTC (Real Time Clock).

Структурная схема обычного программируемого массива-счетчика показана на рис.2.7., а его функциональная схема - на рис.2.8.

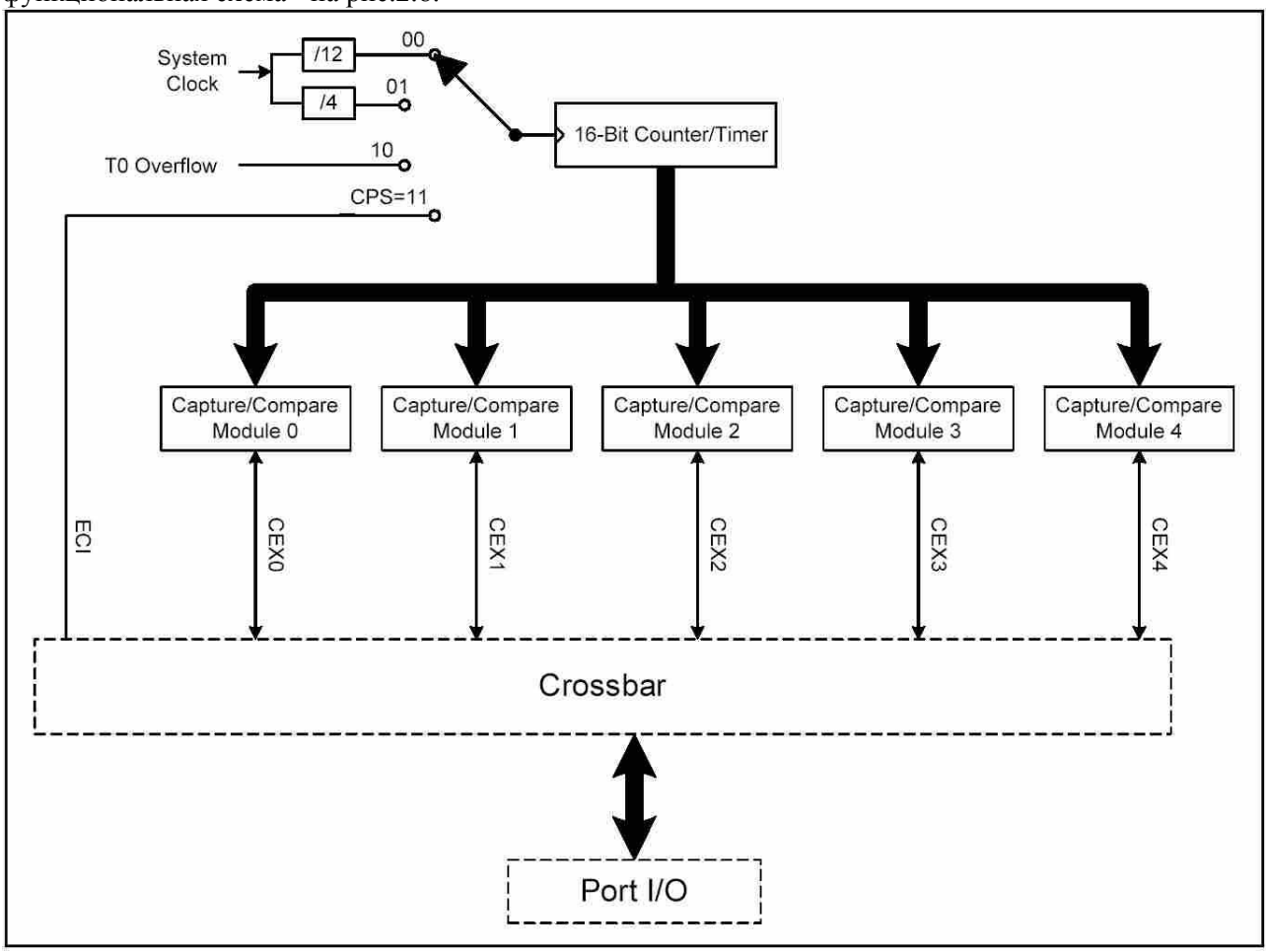

Рис.2.7. Программируемый массив-счетчик

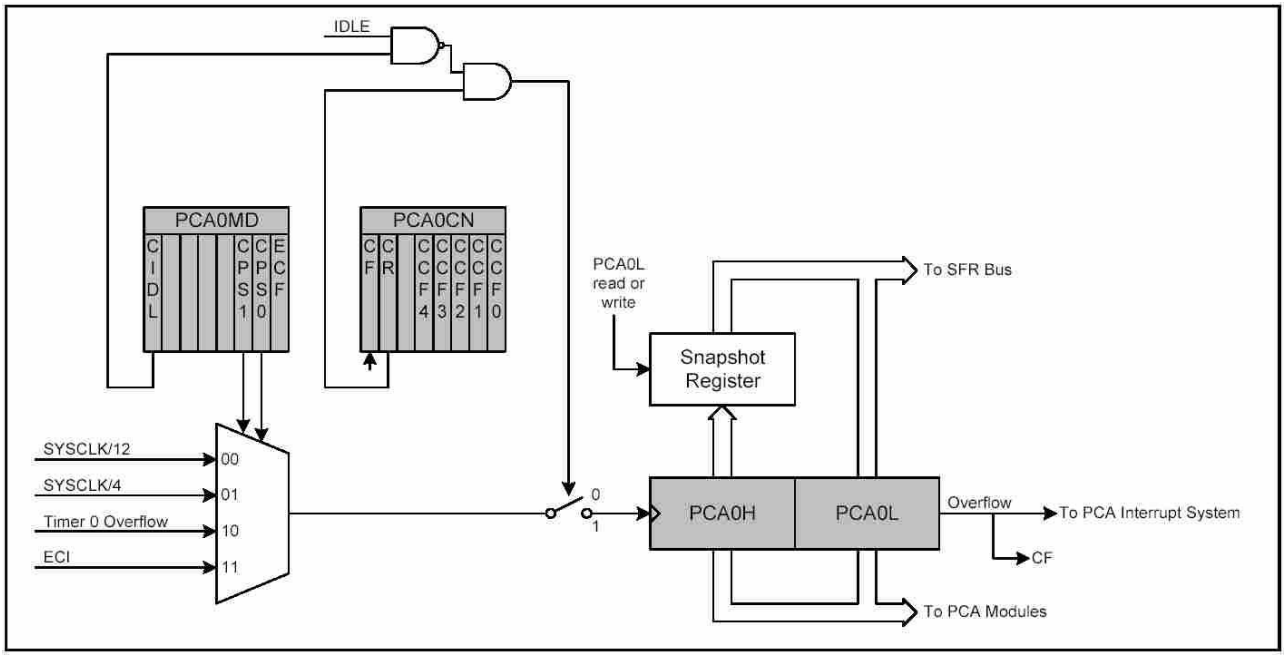

Рис.2.8. Функциональная схема программируемого массива-счетчика

Рассмотрим подробнее основные четыре режима работы модулей CCM:

Режим переключений по фронту (Edge-triggered Capture Mode). На рис.2.9. показана функциональная схема, поясняющая функционирование PCA в этом режиме. В этом случае, вывод CEXn является входом. Он может быть настроен установкой битов CAPPn и CAPNn в регистре PCAOCPMn на восприятие как переднего фронта импульса (перепада из низкого в высокий уровень), так и заднего фронта импульса (перепада из высокого в низкий уровень) или для работы по обоим фронтам. При приходе запрограммированного фронта в регистре PCA0CN устанавливается соответствующий флаг CCFn и вырабатывается соответствующее прерывание, если оно разрешено. Одновременно значение счетчика (регистры PCA0L и PCA0H) перезаписывается в регистры PCA0CPLn и PCA0CPHn. Флаг CCFn автоматически аппаратно не сбрасывается, и должен быть очищен программой.

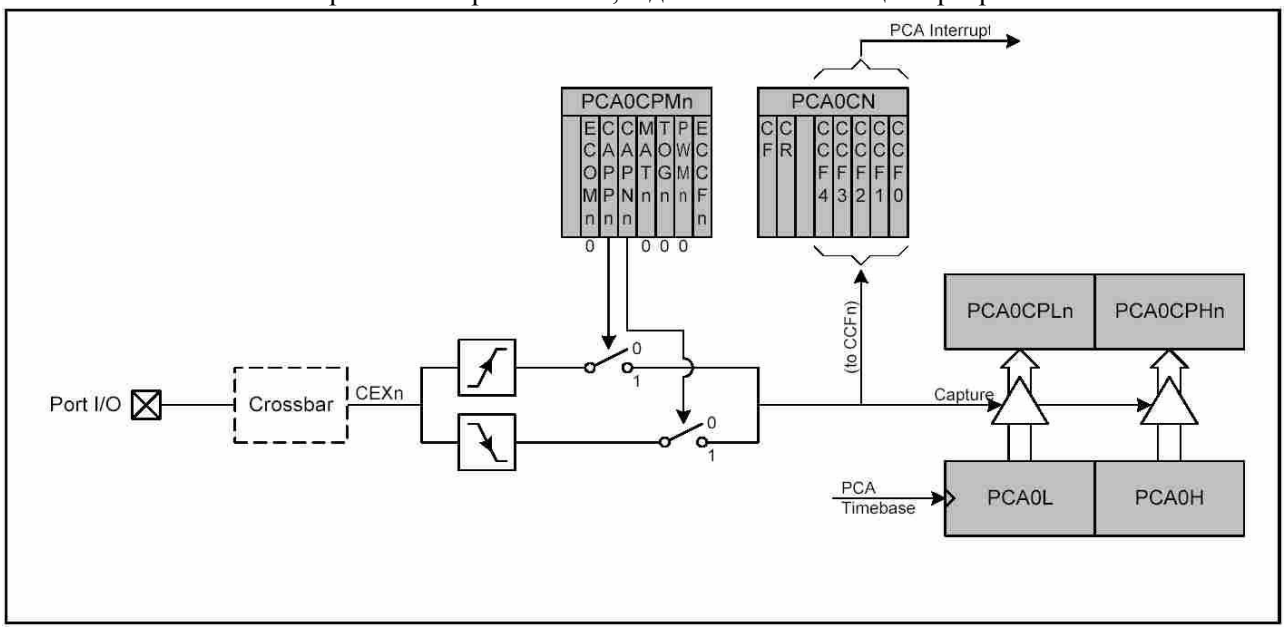

Рис.2.9. Режим переключений PCA по фронту

Режим программного счетчика (Software Timer (Compare) Mode). Этот режим показан на рис.2.10. В этом режиме происходит сравнение шестнадцатибитных величин счетчика PCA (регистры PCA0L и PCA0H) и регистров PCA0CPLn и PCA0CPHn. При совпадении кодов регистре PCA0CN устанавливается соответствующий флаг CCFn и вырабатывается соответствующее прерывание, если оно разрешено. Флаг CCFn автоматически аппаратно не сбрасывается, и должен быть очищен программой. Режим разрешается установкой битов ECOMn и MATn в регистре PCA0CPMn.

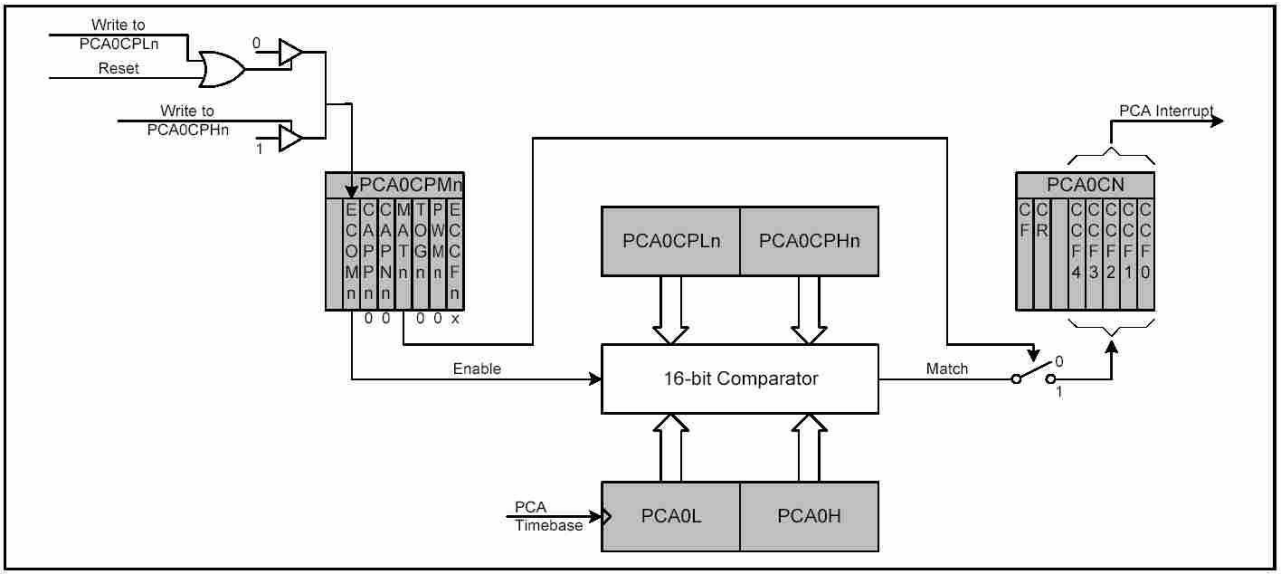

Рис.2.10. Режим программного счетчика PCA

Режим высокоскоростного выхода (High Speed Output Mode). Функциональная схема режима показана на рис.2.11. В этом режиме сравниваются состояния счетчика PCA и регистров PCA0CPHn и PCA0CPLn. В момент совпадения происходит установка встроенного триггера, сброс которого происходит по переполнению младшего байта счетчика PCA. Режим настраивается установкой битов TOGn, MATn и ECOMn в регистре PCAOCPMn.

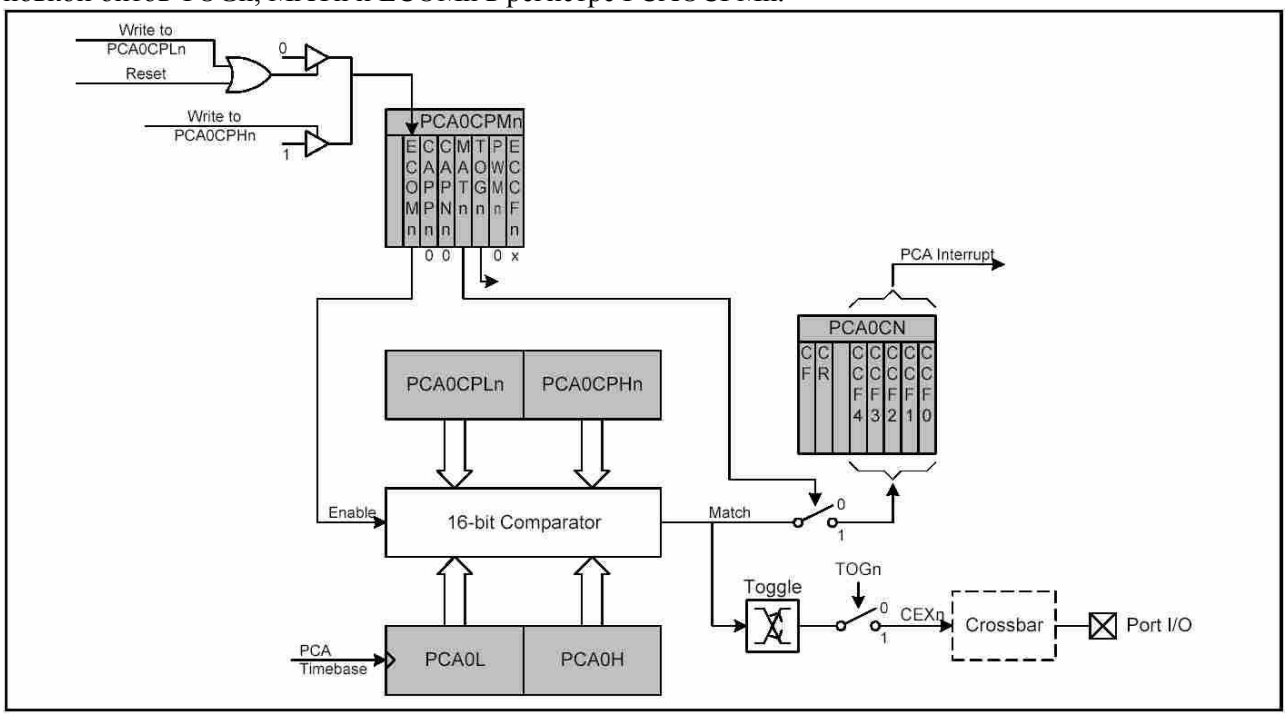

Рис.2.11. Режим высокоскоростного выхода

Режим широтоимпульсного модулятора (PWM- Pulse Width Modulator Mode). Функциональная схема режима показана на рис.2.12. Все модули могут независимо использоваться для формирования PWM сигналов и вывода их на соответствующие вывода CEXn. Очевидно, что выходная частота зависит от заданного источника входных импульсов, выбираемых в узле TimeBase. Параметры выходного сигнала определяются настройкой регистра PCA0CPLn. Если код младшего байта PCA0L счетчика PCA равен величине PCA0CPLn, соответствующий выход CEXn устанавливается в состояние логической единицы. При переполнении байта PCA0L выход CEXn будет сброшен, а его значение (байта PCA0L) изменится с 0xFF на 0x00. Кроме того, значение регистра PCA0CPLn будет аппаратно перезагружено из PCA0CPHn.

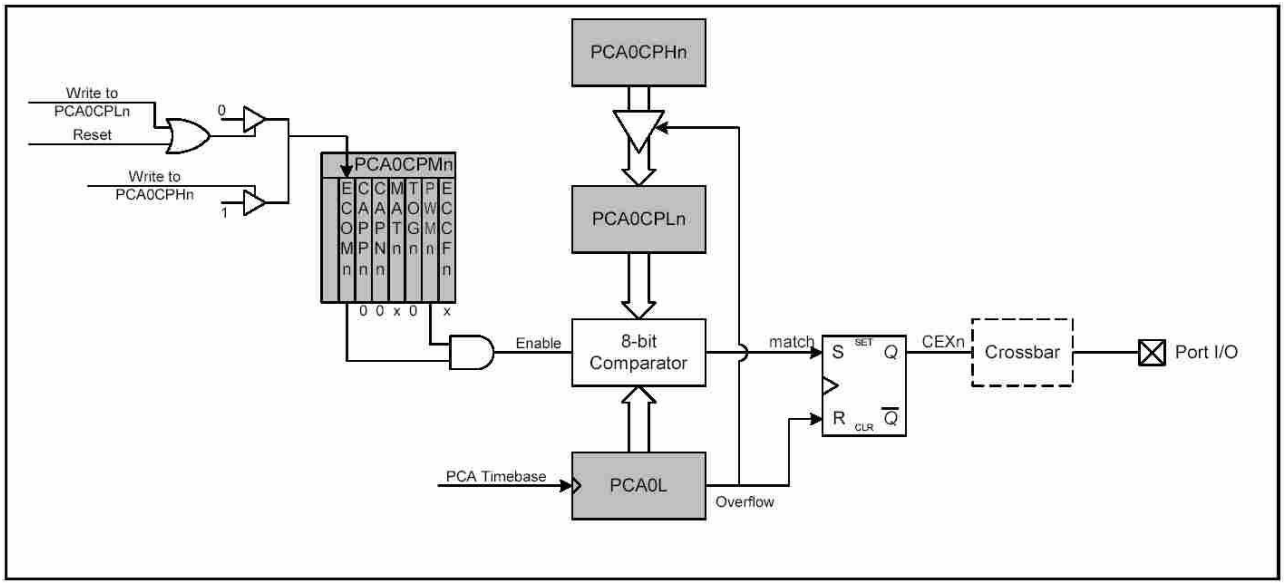

Рис.2.12. Режим широтоимпульсного модулятора

Как уже отмечалось выше, шестнадцатибитный таймер/счетчик PCA состоит из двух восьмибитных регистров SFRs: PCA0L и PCA0H. Регистр PCA0H - старший (MSB), а регистр PCA0L - младший байт (LSB). Чтение регистра PCA0L автоматически защелкивает содержимое регистра PCA0H для последующего чтения. Чтение этих регистров не нарушает ход текущей операции. Биты CPS1 и CPS0 регистра PCA0MD выбирают режим timebase, как показано в таблице 2.3.

Таблица 2.3.

CPS1	CPS0	Источники сигналов узла Timebase
0	0	Системная частота, деленная на 12
0	1	Системная частота, деленная на 4
1	0	Переполнение таймера 0
1	1	Задний фронт импульса (перепад из высокого уровня в низкий) на взоде ECI. При этом максимально возможная скорость определяется, как системная частота, деленная на 4

Когда таймер/счетчик PCA переполняется, т.е. переходит из состояния 0xFFFF в 0x0000, флаг переполнения счетчика CF (Counter Overflow Flag) в регистре PCA0MD устанавливается в логическую 1 и генерируется соответствующее прерывание CF, если оно разрешено. Установка в 1 бита ECF в регистре PCA0MD разрешает генерацию CF прерывания при установке соответствующего флага. Бит CF не стирается аппаратно (автоматически) и должен быть очищен программно. Конечно же, глобальные прерывания PCA0 должны быть разрешены (бит IE.7 в регистре EA и бит EPCA0 в регистре EIE1 должны быть установлены) для генерации прерывания CF. Очистка бита CIDL в регистре PCA0MD разрешает продолжение функционирования PCA при переходе ядра в Idle режим энергосбережения.

2.7. Интерфейс SMBus / I2CBus

Многие микроконтроллеры фирмы Cygnal оснащаются последовательным интерфейсом ввода/вывода SMBus, соответствующим спецификации версии 1.1. (System Management Bus Specification, v.1.1) [5-7]. Этот интерфейс представляет собой двухпроводную двунаправленную последовательную шину, совместимую с последовательным интерфейсом I^2C. Чтение и запись информации производится байтами под автономным управлением встроенного контроллера последовательной передачи данных. Данные могут передаваться до скорости, соответствующей 1/178 от системной частоты. При этом скорость передачи может быть выше, чем предусмотрено спецификацией. Реальная скорость передачи зависит от типов устройств, подключенных к шине (точнее от их предельного быстродействия), и конечно может быть намного меньшей.

В описываемом интерфейсе различают два типа передачи данных: передачи данных от главного (Master) источника к адресуемому подчиненному (Slave) приемнику, и передачи данных от адресуемого подчиненного (Slave) источника к главному (Master) приемнику. Инициатором обоих типов передач данных всегда является Master (главный) интерфейс (узел), он передает в шину тактовую последовательность импульсов. Интерфейс SMBus может работать как в качестве главного, так и подчиненного узла. Описываемый интерфейс поддерживает также работу шины с несколькими главными узлами. При этом, при одновременной попытке двух или более узлов инициировать обмен (в режиме Master), используется схема арбитража. Функциональная схема интерфейса SMBus показана на рис.2.13.

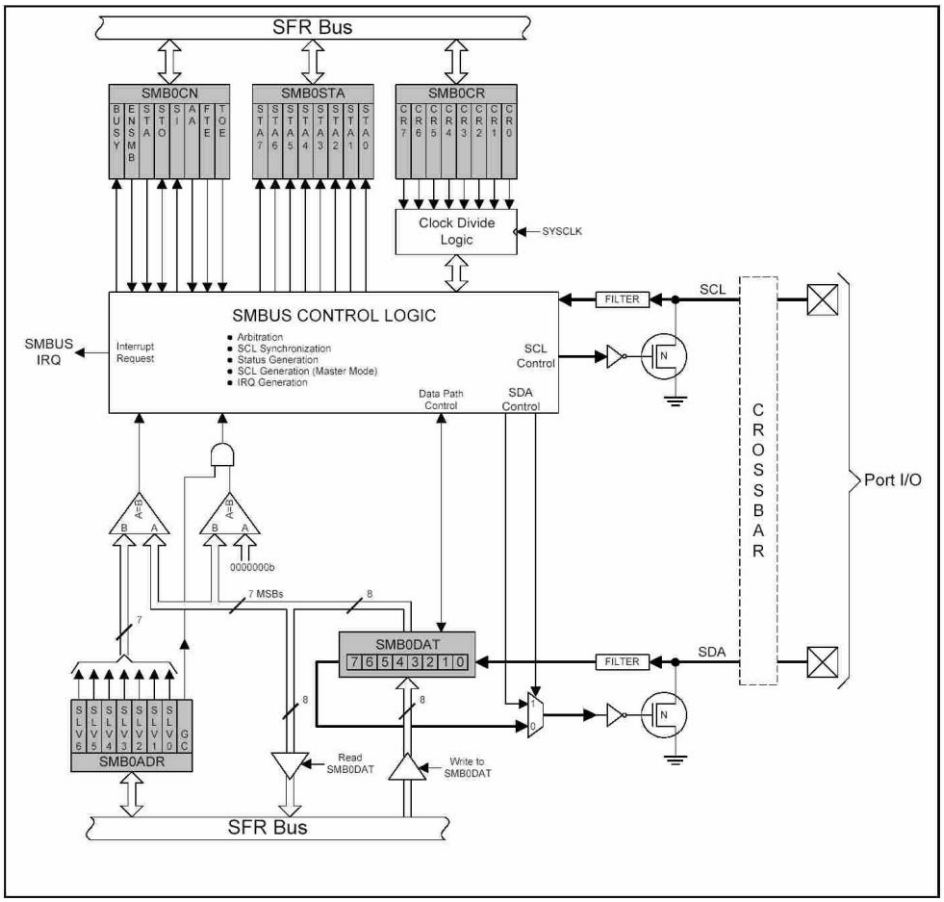

Рис.2.13. Функциональная схема интерфейса SMBus

На рис.2.14. показана типовая конфигурация шины SMBus. На рисунке показано, что интерфейс SMBus может работать с напряжениями от 3В до 5В, и что к одной шине могут быть подключены приборы, работающие с различными напряжениями. Интерфейс включает две линии: SCL - линия тактовых импульсов и SDA линия передачи данных. Обе линии двунаправленные. Обе линии выполнены с выходами типа "открытый коллектор" и предполагают наличие подтягивающих резисторов к положительному источнику питания. Из этого следует, что когда линии пассивны, они имеют высокий логический уровень. Количество узлов на шине SMBus ограничено форматом адреса (127 адресов) и временем переключения фронтов: переднего (перепад от низкого логического уровня к высокому), который должен быть не более 300ns, и заднего (перепад от высокого логического уровня к низкому), который должен быть не более 1000ns.

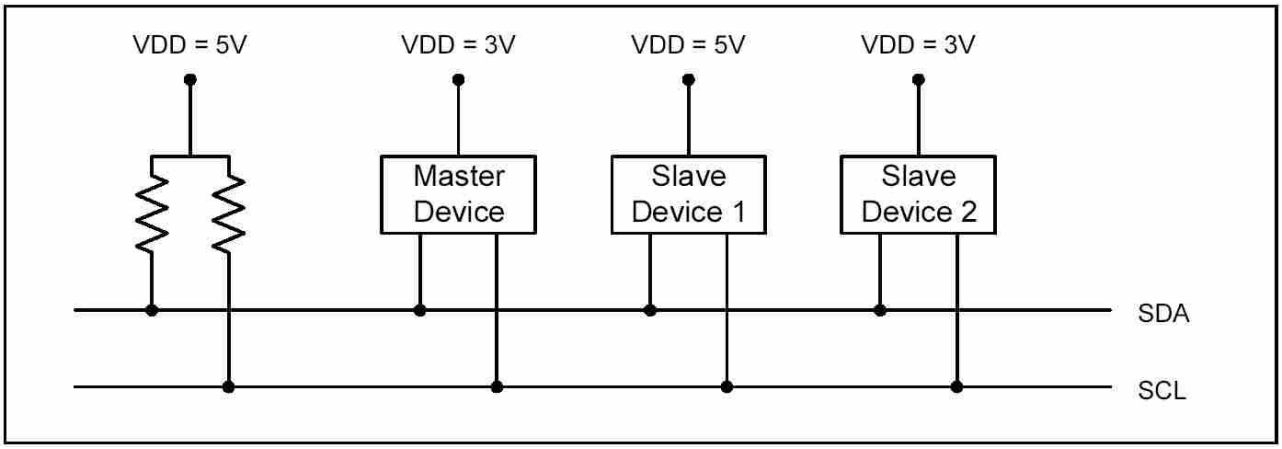

Рис.2.14. Типовая конфигурация шины SMBus

Типовой цикл обмена данных в шине SMBus состоит из: состояния старта (Start), байта адреса, одного или нескольких байтов данных, состояния останова (Stop). Передача байта адреса и каждого из байтов данных сопровождаются подтверждающим (Acknowledge) битом от приемника. Байт адреса состоит из семи битов адреса (т.е. максимально возможное количество устройств - 127) и бита направления R/W (1 = Чтение / 0 = Запись). Операции с адресом 0x00 означают, что Master узел производит передачу всем Slave узлам одновременно. Как уже указывалось выше, все операции производятся под управлением узла - Master. Он генерирует состояние старта, далее передает адрес с битом направления (операции R/W). Если производится операция записи (R/W=0) от главного (Master) узла к подчиненному узлу, то главный узел побайтно передает данные и ожидает от подчиненного узла бита подтверждения после каждого переданного байта данных. Если производится операция чтения (R/W=1), то подчиненный узел передает побайтно данные и ожидает бит подтверждения от главного узла после каждого переданного байта. После завершения передачи данных главный узел формирует состояние останова (Stop) и освобождает шину.

Итак, интерфейс может находиться в одном из четырех режимов:

1. Главный источник (Master Transmitter Mode);
2. Главный приемник (Master Receiver Mode);
3. Ведомый (подчиненный) источник (Slave Transmitter Mode);
4. Ведомый (подчиненный) приемник (Slave Receiver Mode).

Главный узел может начать обмен только если шина свободна. Состояние "свободной линии" определяется по высокому логическому уровню на обеих линиях в течение определенного времени. Два или более главных узла могут попытаться одновременно начать обмен и одновременно сгенерировать состояние Start на шине. Поскольку узел не может знать о намерении других узлов одновременно с ним начать обмен по шине, необходима схема арбитража. Это решается достаточно просто. Состояние Start заключается в переводе линии SDA в состояние логического нуля на определенное время. Несколько узлов беспрепятственно могут сделать это. Одновременно они анализируют состояние этой линии. Первый из узлов, который попытается перевести линию в высокий уровень и захватит магистраль (выиграет арбитраж), заставляя остальные узлы освободить шину.

Шина SMBus использует механизм синхронизации, аналогичный интерфейсу I2C. Главный узел передает тактовые импульсы, а ведомый либо работает на предложенной главным узлом скорости, либо притормаживает скорость, переводя линию SCL в низкий логический уровень через один импульс SCL, снижая, таким образом, скорость тактовых импульсов.

На шине SMBus возможны несколько типов состояния таймаута.

Таймаут тактовой линии в нуле (SCL Low Timeout). Если линия SCL переведена низкоскоростным ведомым узлом в состояние низкого логического уровня, никакая дальнейшая передача невозможна. Очевидно, что главный узел в этой ситуации ничего не может сделать. Для исключения этой ситуации предусмотрено, что каждый узел определяет длительность нахождения линии SCL в состоянии низкого уровня, и если эта длительность превышает 25 ms, состояние считается таймаутом. Каждый узел, определивший состояние таймаута на шине, обязан сбросить свой интерфейс не позднее 10 ms. Интерфейс SMBusне имеет специального узла мониторинга линии SCL, однако любой из таймеров общего назначения в режиме шестнадцатибитного регистра с автозагрузкой может быть использован для этого, а таймер 3 многих микроконтроллеров специально приспособлен для таких целей.

Таймаут тактовой линии в единице (SCL High Timeout). Этот таймаут можно считать функционально нормальным (т.е. не ошибочным), т.к. он свидетельствует о том, что шина свободна. Этот таймаут определяется, если обе линии SCL и SDA, находятся в состоянии логической единицы более чем 50 mks. Если бит FTE в регистре SMB0CN установлен, то определить свободна ли линия можно по состоянию регистра SMB0CR.

Интерфейс SMBus, как и все остальные интерфейсы, доступен через 5 регистров (в SFR): SMB0CN - регистр управления; SMB0CR - регистр задания скорости; SMB0ADR - регистр адреса; SMB0DAT - регистр данных и SMB0STA - регистр состояния. Подробное описание регистров будет приведено для каждого из семейств в соответствующей главе.

2.8. Последовательный периферийный интерфейс SPI

Последовательный периферийный интерфейс SPI (Serial Peripheral Interface) представляет собой полнодуплексный четырехпроводный интерфейс с шинной конфигурацией подключаемых узлов (устройств). SPI интерфейс позволяет подключать к одному ведущему или главному (Master) узлу несколько ведомых (Slave) узлов через общую шину. Отдельный сигнал NSS (Slave-Select signal) выбора ведомого устройства используется для выбора ведомого устройства при осуществлении с ним обмена данными. Кроме того, возможно также построение системы со многими ведущими узлами. Обнаружение конфликтов при одновременной передаче нескольких ведущих узлов также предусмотрено.

Интерфейс SPI может быть запрограммирован для работы в качестве ведущего (Master) или ведомого (Slave). Если интерфейс запрограммирован как ведущий, он может работать на максимальной скорости передачи данных (bits/sec), равной половине тактовой частоты. Если интерфейс запрограммирован для работы в качестве ведомого, его максимальная скорость в полнодуплексном режиме равняется одной десятой тактовой частоты. Подразумевается, что источником синхронизации в обоих случаях является системный генератор тактовой частоты. Если ведущий интерфейс вырабатывает SCK, NSS и последовательные входные данные асинхронно, максимальная скорость передачи должна быть меньше одной десятой тактовой частоты.

Существует еще один особенный режим, когда ведущий должен только передавать данные ведомому (полудуплексный режим), и не должен принимать данные от ведомого. В этом случае максимальная скорость передачи составляет одну четвертую от системной тактовой частоты, при синхронном режиме работы.

На рис.2.15. показана функциональная схема работы SPI интерфейса, а на рис.2.16. показана типовая структурная схема SPI сети.

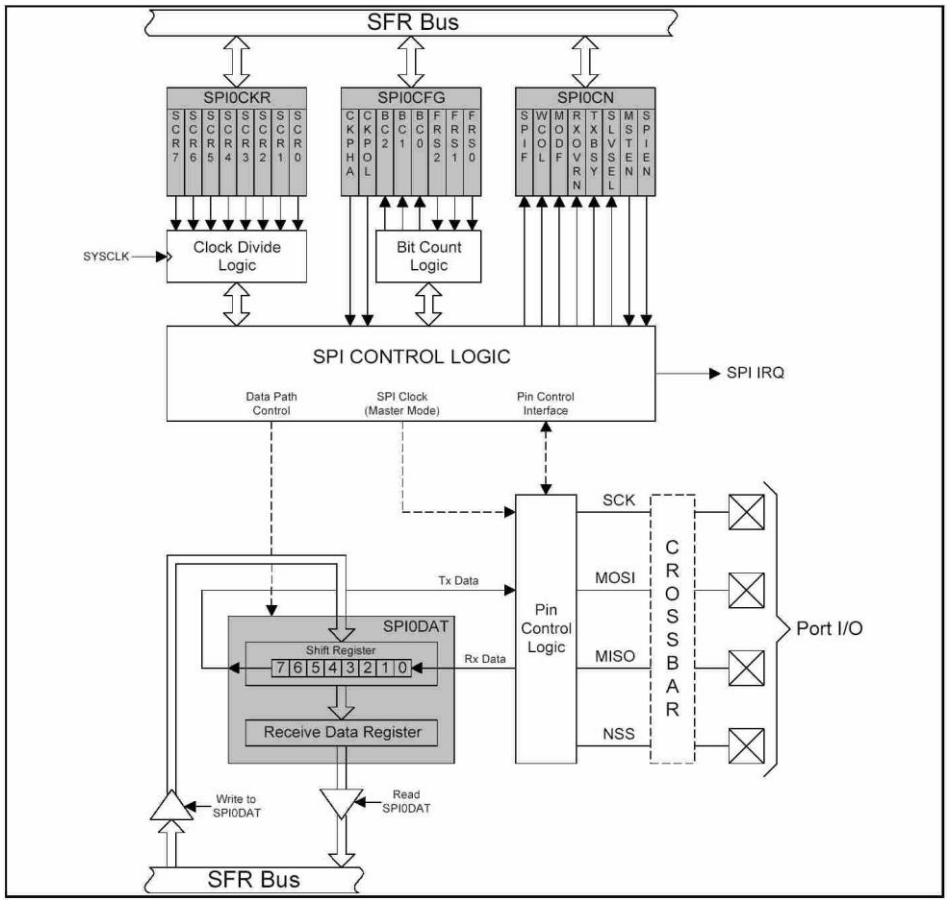

Рис.2.15. Функциональная схема SPI интерфейса

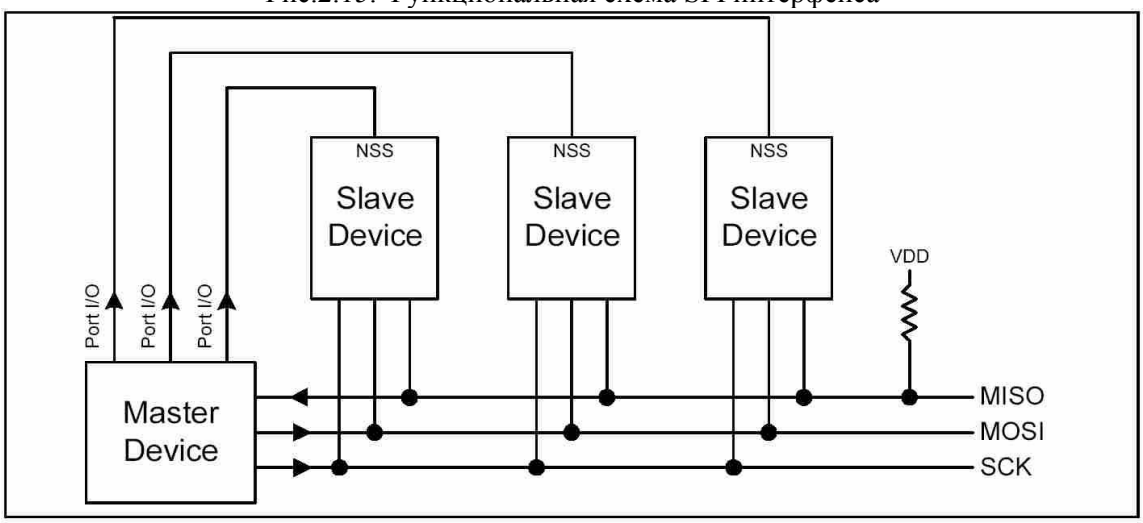

Рис.2.16. Типовая структурная схема SPI сети

Интерфейс SPI имеет четыре сигнальные линии: MOSI, MISO, SCK и NSS.

Линия MOSI (Master-Out, Slave-In) - выходная линия данных ведущего интерфейса и входная линия данных ведомого интерфейса. Из названия следует, что линия предназначена для передачи данных от ведущего (Master) интерфейса (или узла сети) к ведомому (Slave) интерфейсу (или узлу сети).

Линия MISO (Master-In, Slave-Out) - входная линия данных ведущего интерфейса и выходная линия данных ведомого интерфейса. Линия предназначена для передачи данных от ведомого интерфейса к ведущему. Данные передаются байтами, побитно, начиная со старшего бита. Следует помнить, что вывод MISO ведомого интерфейса находится в высокоимпедансном состоянии, если ведомый интерфейс не выбран по линии NSS

Линия NSS (Slave Select) - линия выборки ведомого, предназначена для выборки (низким логическим потенциалом) ведомого интерфейса ведущим.

Линия SCK (Serial Clock) - выходная линия тактовых импульсов ведущего узла и входная линия тактовых импульсов ведомого узла. Линия SCK используется для синхронизации передачи данных между ведущим и ведомым интерфейсами по линиям MOSI и MISO.

В сети на базе SPI интерфейсов только один интерфейс может быть ведущим. Интерфейс устанавливается в режим ведущего установкой флага MSTEN (Master Enable flag) - бита SPI0CN.l. Если интерфейс установлен в режим ведущего, то запись байта данных в регистр данных SPI0DAT приводит к началу передачи. Ведущий интерфейс немедленно побитно сдвигает данные и выдает их на линию MOSI в сопровождении тактовых импульсов на линии SCK. После завершения передачи устанавливается флаг SPIF (SPIOCN.7). Если разрешены прерывания, выдается соответствующее прерывание. Кроме того, интерфейс может быть запрограммирован на выдачу от одного до восьми битов для осуществления связи с SPI приборами, имеющими различную длину слова. Длина передачи (количество передаваемых битов) может быть задана битами SPIFRS в регистре конфигурации SPIOCFG.[2:0] (SPI Configuration Register). Соединение двух интерфейсов SPI (ведущего и ведомого) показано на рис.2.17.

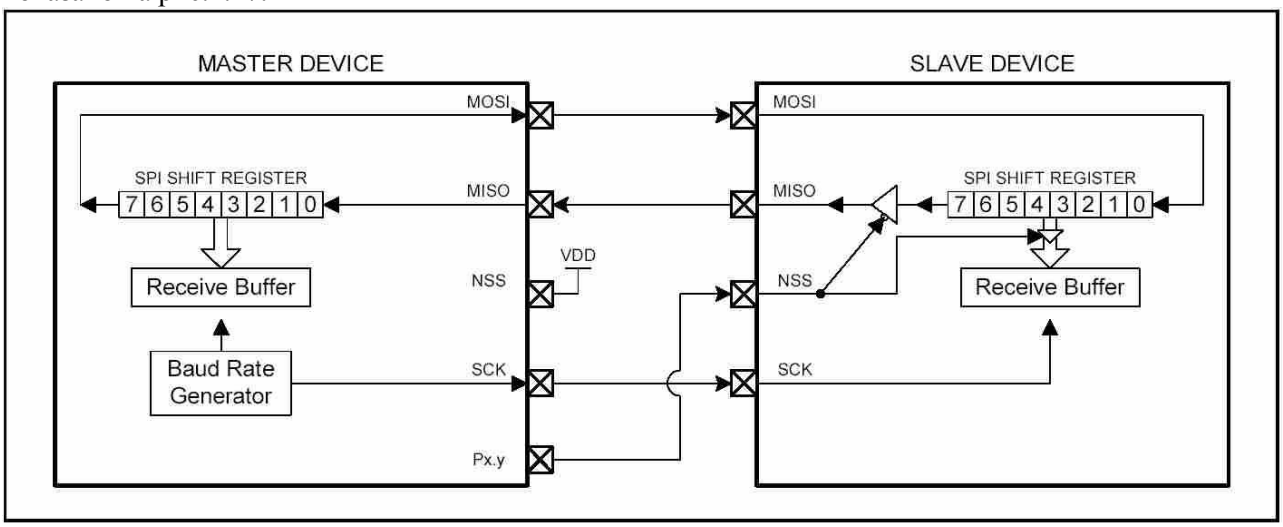

Рис.2.17. Соединение двух интерфейсов SPI

Выше уже отмечалось, что интерфейс может работать в полнодуплексном режиме, это означает, что возможна одновременная передача данных по линиям MOSI от ведущего к ведомому, и MISO от ведомого к ведущему. Данные, полученные от ведомого интерфейса, заменяют данные в регистре данных ведущего интерфейса. Этот регистр дважды буферизирован на ввод, но не на вывод. Т.е. если в регистр данных SPI0DAT производится попытка записи данных во время передачи предыдущего байта, устанавливается флаг WCOL (SPIOCN.6) и попытка записи игнорируется. Таким образом, текущая передача данных продолжается непрерывно. Чтение из регистра данных SPI0DAT приводит к чтению приемного буфера. Если прием не закончен, устанавливается флаг RXOVRN (SPIOCN.4). Новые данные не передаются в регистр чтения, пока предыдущий принятый байт не будет прочитан. Очевидно, что при задержке чтения принятых байтов может произойти потеря данных. Если SPI интерфейс на настроен, как ведущий (Master), он будет работать в режиме ведомого (Slave).

Кроме того, поддерживается режим сети с многими ведущими. Флаг MODF (SPIOCN.5 - Mode Fault flag) устанавливается в логическую единицу, если интерфейс определен как ведущий (MSTEN=1) и
вывод NSS переведен в низкий логический уровень, т.е. SPI интерфейс пытаются использовать в качестве ведомого. Если при этом установлен флаг MODF, биты MSTEN и SPIEN в регистре управления SPI стираются автоматически аппаратно, переводя интерфейс в автономное состояние. Таким образом, в системе с многими ведущими ядро может определить свободна ли шина путем опроса флага SLVSEL (SPI0CN.2) перед тем, как установить MSTEN флаг (т.е. определить интерфейсу режим ведущего) и инициализировать обмен.

На рис.2.18 показаны временные диаграммы работы SPI интерфейса. Возможны четыре комбинации фаз тактовых импульсов и их полярности в зависимости от комбинации управляющих битов в регистре конфигурации SPI0CFG (SPI Configuration Register). Бит CKPHA (SPI0CFG.7) выбирает одну из двух фаз тактовых импульсов, т.е. фронт, по которому осуществляется запись данных. Другой бит CKPOL (SPI0CFG.6) определяет активную полярность (высокий или низкий уровень). Очевидно, что и ведущий, и ведомый узлы должны иметь одинаковые настройки фазы и полярности.

Еще одна важная особенность настройки заключается в том, что интерфейс SPI должен быть запрещен (путем обнуления бита SPIEN, SPI0CN.0) на время настройки фазы и полярности тактовых импульсов.

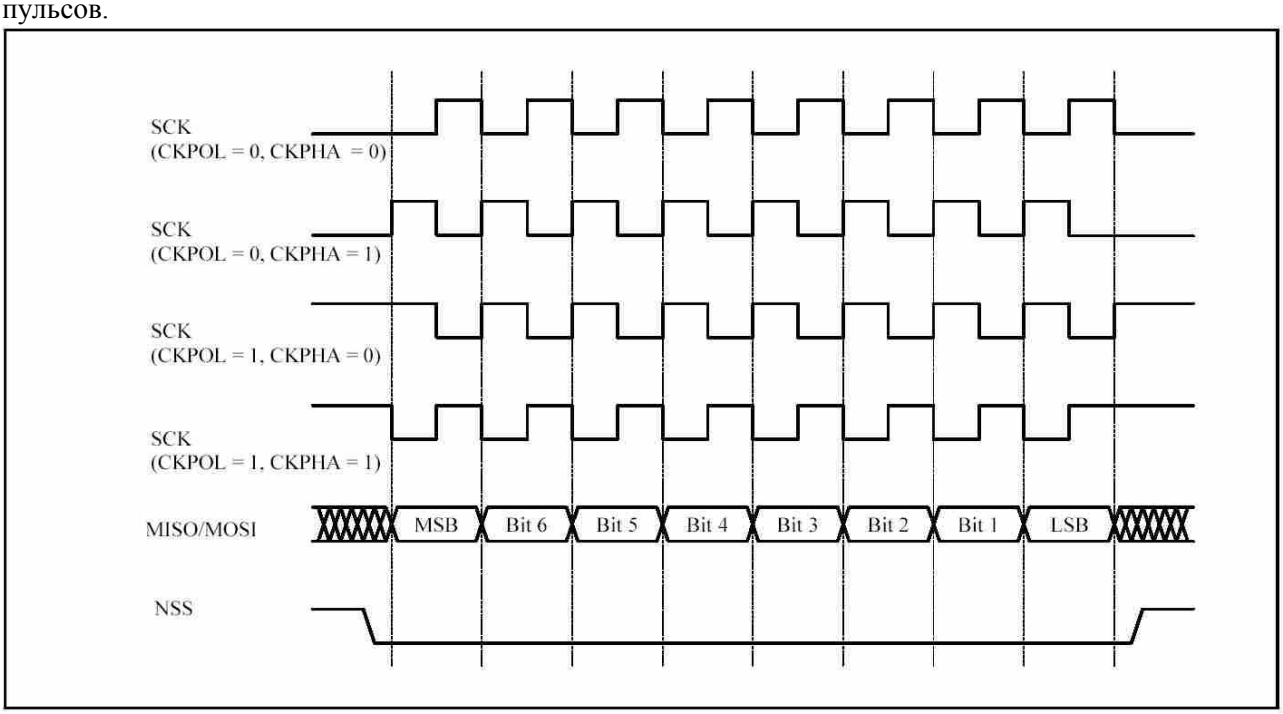

Рис.2.18. Временные диаграммы работы SPI интерфейса

Кроме описанных SFR регистров, при настройке SPI интерфейса используются регистр настройки скорости передачи (SPI0CKR). Подробное описание назначения различных регистров и их конфигурирования будут приведены в соответствующих разделах книги.

2.9. Последовательный порт UART

Последовательный порт UART микроконтроллеров Cygnal предназначен для асинхронной передачи данных. Он может также работать в полнодуплексном режиме. Во всех режимах входные данные буферизируются в специальном регистре. Подразумевается, что данные должны быть считаны из этого регистра до момента завершения приема следующих данных. Управление и обмен данными с последовательным портом UART осуществляется через соответствующие регистры в SFR: последовательный регистр управления SCON (Serial Control Register) и регистр данных SBUF (Serial Data Buffer), причем единственный регистр данных обслуживает и прием, и передачу данных. При чтении автоматически осуществляется доступ к принятым данным, а при записи - к регистру передачи данных. Естественно, что последовательный порт может генерировать прерывания, если они разрешены. Существуют два программно доступных флага: флаг завершения передачи байта TI (Transmit Interrupt flag) (SCON.l) и флаг приема байта RI (Receive Interrupt flag) (SCON.0). Оба они устанавливаются в высокий логический уровень после завершения соответствующей операции. Обнуление этих флагов производится только программно. Функциональная схема интерфейса последовательного порта UART показана на рис.2.19.

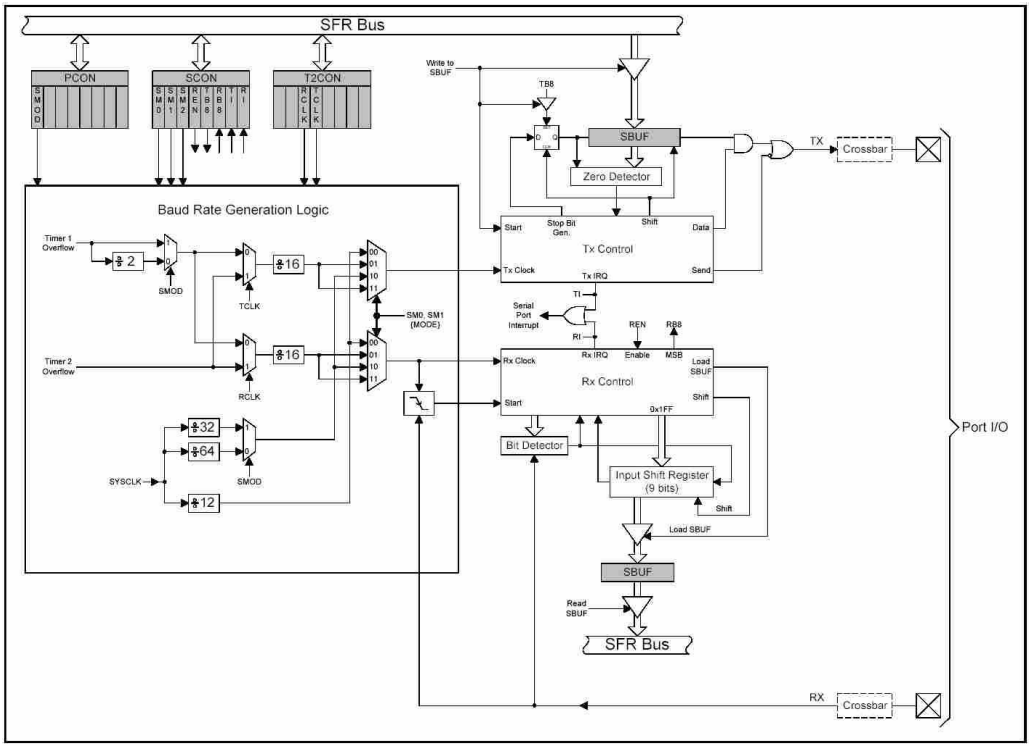

Рис.2.19. Функциональная схема интерфейса последовательного порта UART

Последовательный порт UART может функционировать в одном из четырех режимов, выбираемом установкой управляющих флагов в SCON регистре. Эти режимы отличаются протоколом и скоростью передачи данных (см. табл.2.4.).

Таблица 2.4.

Таблица режимов последовательного порта UART

Режим	Синхронизация	Источник тактовых импульсов	Кол. битов	Start/Stop биты
0	Синхронная	SYSCLK/12	8	-
1	Асинхронная	Переполнение таймеров 1 или 2	8	1 Start, 1 Stop
2	Асинхронная	SYSCLK/32 или SYSCLK/64	9	1 Start, 1 Stop
3	Асинхронная	Переполнение таймеров 1 или 2	9	1 Start, 1 Stop

Режим 0 - Синхронный режим UART. Режим 0 предназначен для синхронного полудуплексного обмена данными. Последовательные данные передаются и принимаются по линии RX, а линия TX предназначена для тактовых импульсов. Последовательный порт должен быть настроен, как ведущий (Master) для генерации тактовых импульсов при передаче данных в обоих направлениях. В этом режиме передаются и принимаются восемь битов (первый младший). Данные начинают передаваться сразу же после их записи в регистр SBUF. При передаче последнего бита байта устанавливается флаг TI (SCON.l). Прием байта начинается, если установлен бит REN (Receive Enable bit) (SCON.4) и обнулен бит RI (SCON.0). После завершения приема последнего бита устанавливается бит RI и прием останавливается до тех пор, пока этот бит не будет обнулен программно. Если разрешены прерывания UART, то после установки битов-флагов TI и RI генерируются соответствующие прерывания. В этом режиме линия RX работает как выход/вход "с открытым истоком", т.е. предполагается наличие внешнего подтягивающего резистора к линии напряжения питания. Рис.2.20. иллюстрирует работу UART в режиме 0.

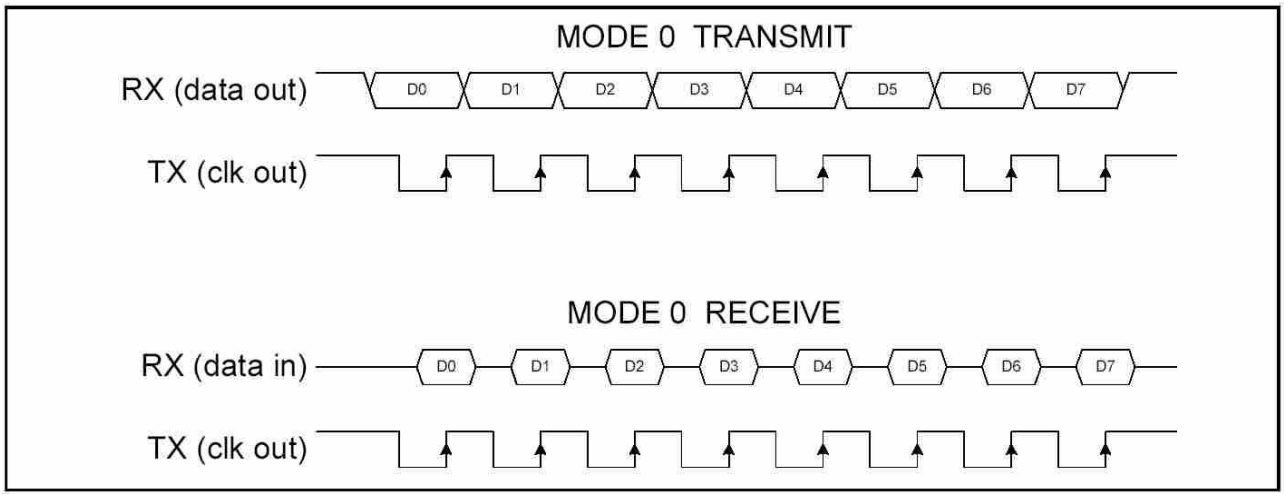

Рис.2.20. Временная диаграмма работы UART в режиме 0

Режим 1 - 8-и битный UART с переменной скоростью. Режим 1 предназначен для стандартного асинхронного полнодуплексного обмена, при котором передается (принимается) 10 бит: один стартовый, восемь битов данных (первым передается младший бит) и один стоповый бит. Данные передаются с вывода TX одного интерфейса на вывод RX другого интерфейса. При приеме, восемь принятых битов данных запоминаются в регистре SBUF, а стоповый бит - в бите RB8 (SCON.2). Передача данных начинается сразу после записи байта в регистр SBUF. Флаг TI (SCON.l) устанавливается после приема восьми битов данных в момент приема стопового бита. Прием начинается после установки бита REN (SCON.4). После приема стопового бита, данные записываются в регистр приема SBUF при соблюдении следующих условий: флаг RI=0, *и если установлен SM2 бит, стоповый бит должен быть равным 1*. В этом случае, как упоминалось выше, данных запоминаются в регистре SBUF, стоповый бит - в бите RB8 (SCON.2) и устанавливается флаг RI. При несоблюдении этих условий, записи в SBUF и RB8 не происходит и флаг RI не устанавливается. Прерывания, если они разрешены, вырабатываются при установке флагов RI и TI.

В режиме 1 для генерации тактовых импульсов используется таймер. Последовательный порт UART может использовать таймеры 1 или 2 в режиме автозагрузки. При переполнении выбранного таймера, он посылает импульс на схему генерации тактирования (формирования скорости), где поступающая частота делится на 16 для формирования временной диаграммы режима 1. Если используется таймер 1, он должен быть запрограммирован в режим восьмиразрядного счетчика с автозагрузкой. Коэффициент деления записывается в регистр TH1. Скорость передачи данных определяется по формуле:

$$Mode\ 1\ Baud\ Rate = (2^{SMOD} / 32) * (SYSCLK) / (12^{(T1M-1)} * (256- TH1))$$

Бит SMOD (PCON.7) делить или нет импульсы переполнения таймера 1 на два. После сброса этот бит устанавливается в состояние логического 0, что означает деление на два или низшую скорость генератора скорости. Очевидно, что изменяя код делителя TH1 и SMOD, можно регулировать скорость последовательного порта, и если используя бит TIM в CKCON установить деление на 1, то можно достигнуть максимальную скорость, равную тактовой частоте, деленной на 12.

При использовании таймера 2 в качестве генератора скорости необходимо устанавливать бита RCLK и/или TCLK в логическую единицу. Установка этих битов автоматически запрещает прерывания от таймера 2 и конфигурирует его на работу с тактовой частотой, деленной на 2. Если необходима другая входная частота, то установкой бита C/T2 в логическую единицу таймер переключается на работу от внешнего входа T2. При этом, комбинация входной частоты и величины автозагрузки таймера определяют скорость последовательного порта по следующей формуле:

$$Mode\ 1\ Baud\ Rate = SYSCLK/[32 * (65536-[RCAP2H:RCAP2L])],$$

где [RCAP2H:RCAP2L] - шестнадцатибитная величина , записанная в регистре захвата. Временная диаграмма режима 1 показана на рис.2.21.

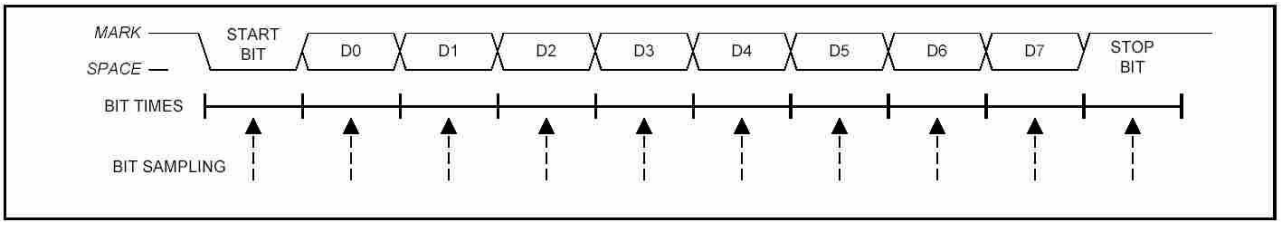

Рис.2.21. Временная диаграмма работы UART в режиме 1

Режим 2 - 9-и битный UART на фиксированной скорости. Режим 2 предназначен для асинхронного полнодуплексного обмена, при котором передается (принимается) 11 бит: один стартовый, восемь битов данных (первым передается младший бит), программируемый девятый бит данных и один стоповый бит. При передаче, девятый бит данных определяются битом TBS (SCON.3). Он может быть назначен, как флаг четности P в PSW ил напрямую формироваться микроконтроллером. При приеме девятый бит данных поступает в RB8 (SCON.2), а стоповый бит игнорируется.

Передача данных начинается после записи байта в регистр данных SBUF. Флаг TI (SCON.l) устанавливается после передачи последнего бита данных в начале формирования стопового бита. Прием данных может начаться в любое время после установки бита REN (Receive Enable bit) (SCON.4). После приема стопового бита байт данных записывается в регистр данных SBUF, при условии, что бит RI обнулен, бита SM2 установлен и девятый бит равен логической единице. Если эти условия соблюдены, восемь битов данных записываются в SBUF, девятый бит записывается в RB8 и устанавливается флаг RI. Если же описанные выше условия не соблюдены, данные не записываются ни в SBUF, ни в RB8 и флаг RI не устанавливается. Конечно же, если разрешены прерывания, они вырабатываются при установке флагов TI и RI.

Скорость передачи в режиме 2 является функцией тактовой частоты:

$$Mode\ 2\ Baud\ Rate = 2^{SMOD} * (SYSCLK/64).$$

Бит SMOD (PCON.7) определяет делитель, на который делится SYSCLK (32 или 64). На рис.2.22. показана временная диаграмма режима 2.

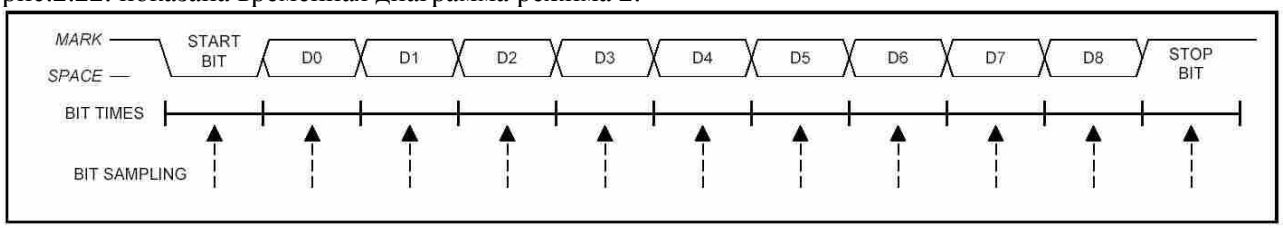

Рис.2.22. Временная диаграмма работы UART в режиме 2

Режим 3 - 9-и битный UART с переменной скоростью. Режим 3 аналогичен режиму 2 во всех отношениях, кроме переменной скорости передачи. Скорость передачи в этом режиме задается также как и в режиме 1. В третьем режиме передаются 11 бит: стартовый бит, восемь бит данных (первым передается младший бит), программируемый девятый бит и стоповый бит. Для задания скорости используются 1 или второй таймеры. Иными словами, режим 3 работы UART, это тот же режим 2, только с заданием скорости, как в режиме 1.

Многопроцессорная связь последовательных портов UART. Режимы 2 и 3 поддерживают многопроцессорную связь между главным ведущим портом (микроконтроллером, узлом) и несколькими ведомыми при специальном использовании девятого бита. Если ведущий узел собирается передать данные одному или нескольким ведомым узлам, он сперва посылает байт адреса для определения приемника. Адресный байт отличается от байтов данных тем, что девятый бит в нем всегда равен логической единице. В байтах данных девятый бит всегда равен нулю.

В ведомых узлах путем установки бита SM2 (SCON.5) последовательный порт UART конфигурируется так, что когда принимается неизвестный байт, он записывается в регистр данных только тогда, когда девятый бит равен 1, т.е. происходит автоматическое выделение адресного байта. В подпрограмме обработки прерывания от приема байта принятый адрес распознается. Если принятый адрес совпадает с адресом узла, ведомый узел обнуляет бит SM2, разрешая тем самым сохранение последующих байтов данных и выработку соответствующих флагов и прерываний. Узлы, которые не

распознали свой адрес, оставляют SM2 бит установленным и ожидают следующего адресного байта. Адресуемый узел принимает все сообщение и опять устанавливает бит SM2 для определения следующего адреса. Часть адресов может быть индивидуальна для ведомых узлов, а некоторые адреса могут относиться ко всем (или группе) ведомых узлов. При этом реализуется одновременная передача многим узлам, называемая "broadcast". Ведущий узел также может быть по разному настроен на прием всех сообщений или нескольких групп ведомых. Рис. 2.23. показывает типовое соединение UART в многопроцессорной сети. В таблице 2.5. приведены значения скоростей передачи данных в зависимости от тактовой частоты.

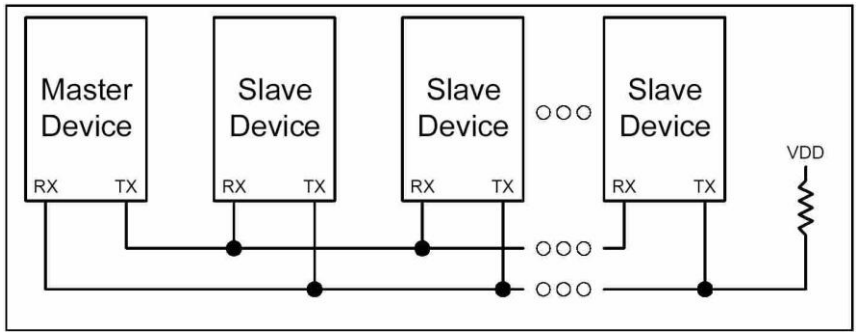

Рис.2.23. Типовое соединение UART в многопроцессорной сети

Таблица.2.5.

Значения скоростей работы UART в зависимости от тактовой частоты

Тактовая частота, MHz	Коэффициент деления	Код для таймера 1 *	Скорость передачи UART **
24.0	208	0xF3	115200(115384)
23.592	205	0xF3	115200(113423)
22.1184	192	0xF4	115200
18.432	160	0xF6	115200
16.5888	144	0xF7	115200
14.7456	128	0xF8	115200
12.9024	112	0xF9	115200
11.0592	96	0xFA	115200
9.216	80	0xFB	115200
7.3728	64	0xFC	115200
5.5296	48	0xFD	115200
3.6864	32	0xFE	115200
1.8432	16	0xFF	115200
24.576	320	0xEC	76800
25.0	434	0xE5	57600 (57870)
25.0	868	0xCA	28800
24.576	848	0xCB	28800 (28921)
24.0	833	0xCC	28800 (28846)
23.592	819	0xCD	28800(28911)
22.1184	768	0xD0	28800
18.432	640	0xD8	28800
16.5888	576	0xDC	28800
14.7456	512	0xE0	28800
12.9024	448	0xE4	28800
11.0592	384	0xE8	28800
9.216	320	0xEC	28800
7.3728	256	0xF0	28800
5.5296	192	0xF4	28800
3.6864	128	0xF8	28800
1.8432	64	0xFC	28800

Примечания:

* Подразумевается SMOD=1 и T1M=1.

** В скобках указана реальная скорость.

2.10. Таймеры

Как уже отмечалось раньше, микроконтроллеры фирмы Cygnal различных семейств могут иметь от трех до пяти таймеров. Первые три шестнадцатибитные таймера/счетчика совместимы с таймерами стандартных 8051 микроконтроллеров. Дополнительные таймеры имеют специальное назначение. Например, четвертый шестнадцатибитный таймер используется совместно с аналого-цифровыми преобразователями ADC, интерфейсом SMBus или может использоваться для общих применений. Он может измерять временной интервал или генерировать периодический запрос прерываний. Таймеры 0 и 1 почти идентичны и могут работать в четырех режимах. Таймер 2 имеет дополнительные возможности, которых нет у таймеров 0 и 1. Таймер 3 почти такой же, как и второй, но без захвата и возможности работы в качестве генератора скорости (например, UART). Основные параметры таймеров приведены в таблице 2.6.

Таблица 2.6.

Основные параметры таймеров

Таймеры 0 и 1	Таймер 2	Таймер 3
13-битный таймер/счетчик	16-битный таймер/счетчик с автозагрузкой	16-битный таймер с автозагрузкой
16-битный таймер/счетчик	16-битный таймер/счетчик с захватом	
8-битный таймер/счетчик с автозагрузкой	Генератор скорости	
Два 8-битных таймера/счетчика (Таймер 0 только)		

При работе в качестве таймеров, их регистры с каждым импульсом увеличиваются на единицу. Счетные импульсы получаются делением тактовой частоты на 1 или 12 в соответствие с битами выбора счетных импульсов (Timer Clock Select) (T2M-T0M) в регистре CKCON. Деление на 12 используется для совместимости ос стандартными 8051 микроконтроллерами.

При работе в режиме счетчика содержимое увеличивается на каждый задний фронт импульса (переход от высокого к низкому уровню) на входах TO, Tl или T2. Предельная частота счета в этом режиме составляет 1/4 частоты тактовых импульсов. Входной сигнал не обязательно должен быть периодическим, но он должен находиться в каждом логическом уровне по крайней мере два периода тактовой частоты для полной достоверности восприятия.

Таймеры 0 и 1. Таймеры 0 и 1 доступны и управляются через регистры SFR Каждый таймер/счетчик представляет собой шестнадцатибитный регистр, доступный как два отдельных байта: младший байт (TLO или TL1) и старший байт (THO или TH1). В регистре управления TCON можно разрешить / запретить работу таймеров и задать их статус. Оба они могут работать в одном из четырех режимов, установленных в битах M1-M0 регистра TMOD. Естественно, что каждый таймер может быть настроен индивидуально.

Режим 0: 13-битный таймер/счетчик (Таймеры 0 и 1). Таймеры 0 и 1 могут функционировать в режиме 13-битного таймера/счетчика в режиме 0. Оба таймера функционируют одинаково, т.е. приводимое описание для таймера 0 может также относится и к таймеру 1.

Регистр TH0 содержит восемь старших битов, а регистр TL0 - пять младших битов (TL0.4+TL0.0) 13-битного таймера/счетчика. Три старших бита регистра TL0 не определены и могут быть либо маскированы, либо игнорироваться при чтении. В режиме 13-битного таймера значение регистров увеличивается и при переходе от значения 0x1FFF к значению 0x0000 (overflows) устанавливается флаг TF0 (TCON.5), а также может вырабатываться соответствующее прерывание, если оно разрешено. Бит C/T0 (TMOD.2) источник счетных импульсов. Обнуление этого флага выбирает для счета тактовые импульсы, а установка - разрешает срабатывание по заднему фронту внешних импульсов. Установка бита TR0 (TCON.4) разрешает таймер, когда GATE0 (TMOD.3) равен нулю или входной сигнал INT0/ находится в состоянии логической единицы. Если GATE0 (TMOD.3) равен единице, таймер управляется внешним сигналом INT0/ произвольной формы. Параметры настройки режима 0 приведены в таблице 2.7.

Таблица 2.7.

Параметры настройки таймеров в режим 0

Бит TR0	Бит GATE0	Вход INT0/	Таймер / Счетчик
0	X	X	Запрещен
1	0	X	Разрешен
1	1	0	Запрещен
1	1	1	Разрешен

Примечание: Символ X означает произвольное состояние

Установка бита TR0 не сбрасывает регистры таймера. Регистры таймера должны быть инициализированы до разрешения таймера.

Таймер 1 функционирует аналогично таймеру 0. На рис.2.24 показана функциональная схема таймеров 0 и 1 в режиме 0.

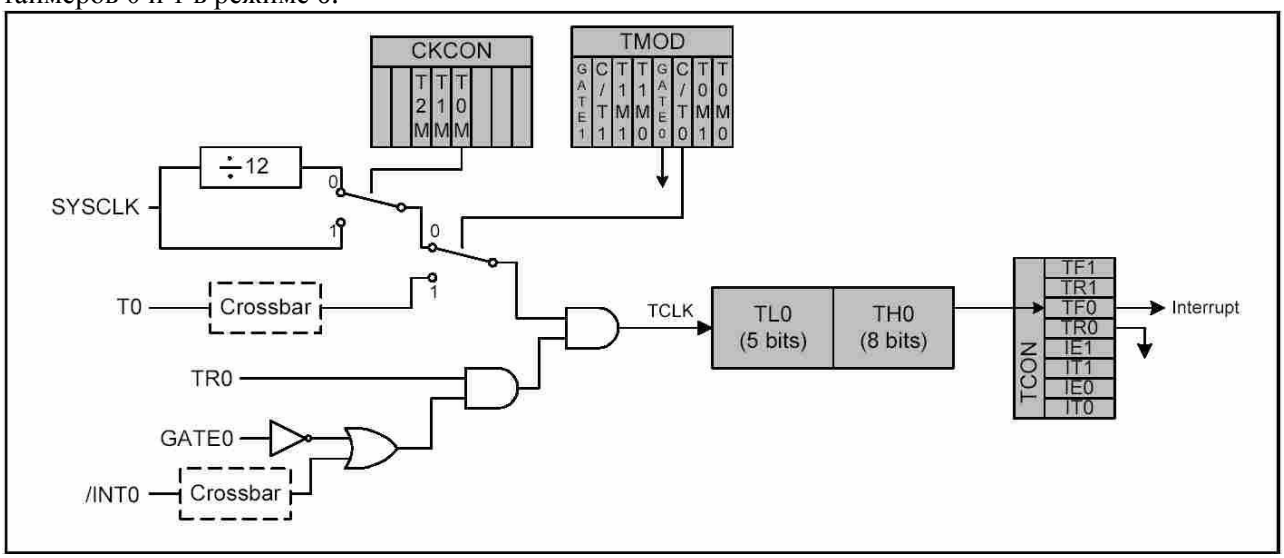

Рис.2.24. Функциональная схема режима 0 (для таймеров 0 и 1)

Режим 1: 16-битный таймер счетчик (Таймеры 0 и 1). Режим 1 функционирует также как и режим 0, за исключением того, что регистры шестнадцатиразрядные.

Режим 2: 8-битный таймер/счетчик с автозагрузкой (Таймеры 0 и 1). Режим 2 таймеров 0 и 1 рассчитан на их работу в качестве 8-битных таймеров/счетчиков с автозагрузкой начальной величины. В этом режиме регистр TL0 выполняет роль счетчика, а регистр TH0 содержит загружаемую величину. Когда регистр TL0 переполняется, устанавливается флаг TF0 (TCON.5) и начальная величина загружается из регистра TH0 в регистр TL0. Естественно, может генерироваться прерывание TF0, если оно разрешено. Перезагружаемая величина, содержащаяся в регистре TH0, не изменяется. Следует помнить, что перед тем, как разрешать работу таймера/счетчика, необходимо инициализировать регистр TL0. Таймер 1 функционирует аналогично таймеру 0. Рис.2.25. иллюстрирует работу таймеров 0 и 1 в режиме 2.

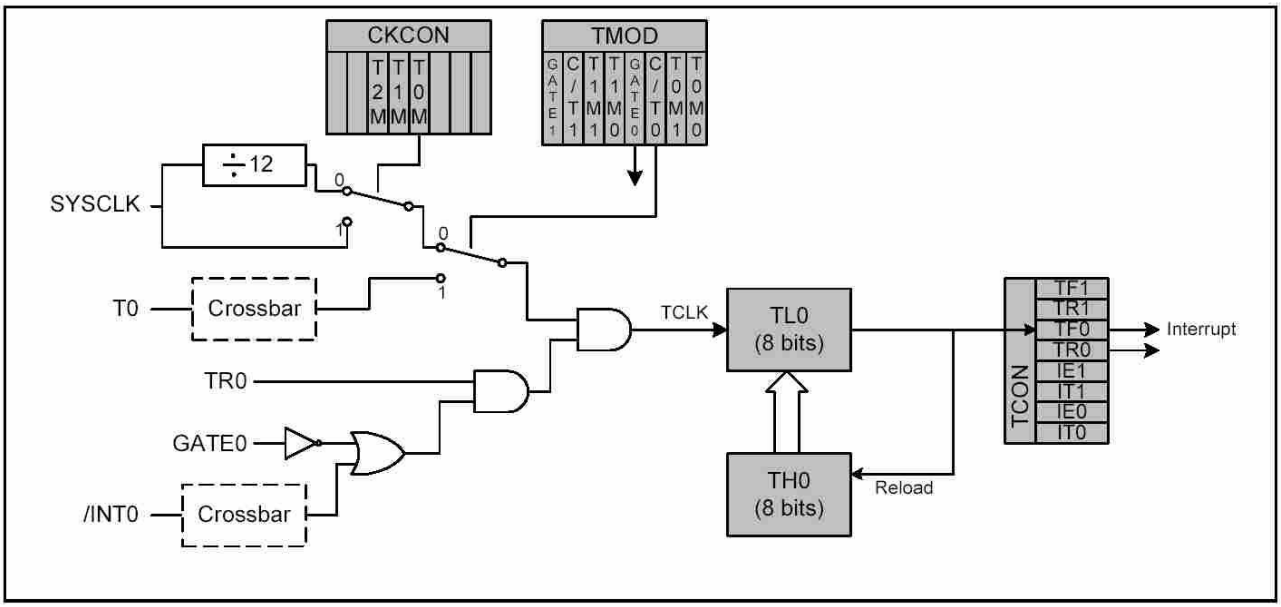

Рис.2.25. Функциональная схема режима 2 (для таймеров 0 и 1)

Режим 3: Два 8-битных таймера/счетчика (только для таймера 0). В режиме 3 таймеры 0 и 1 работают по-разному Таймер 0 работает, как два 8-битных таймера/счетчика. Таймером/счетчиком на регистре TL0 управляют биты: TR0, C/T0, GATE0 и TF0 (регистров TCON и TMOD). Он может использовать для работы системную тактовую частоту или внешний вход. Таймер на регистре TH0 управляется битом TR1. Он может использовать для счета только системную тактовую частоту. Переполнение регистра TH0 вызывает установку флага TF1 и может генерировать прерывание. Таймер 1 в режиме 3 не работает, и когда таймер 0 работает в режиме 3, таймер 1 может работать в режимах 0, 1 или 2, но не может использовать внешний сигнал, устанавливать флаг TF1 и генерировать прерывание. Однако, переполнение таймера 1 может использоваться для генерации последовательной скорости передачи. На рис.2.26. показана функциональная схема режима 3 таймера 0.

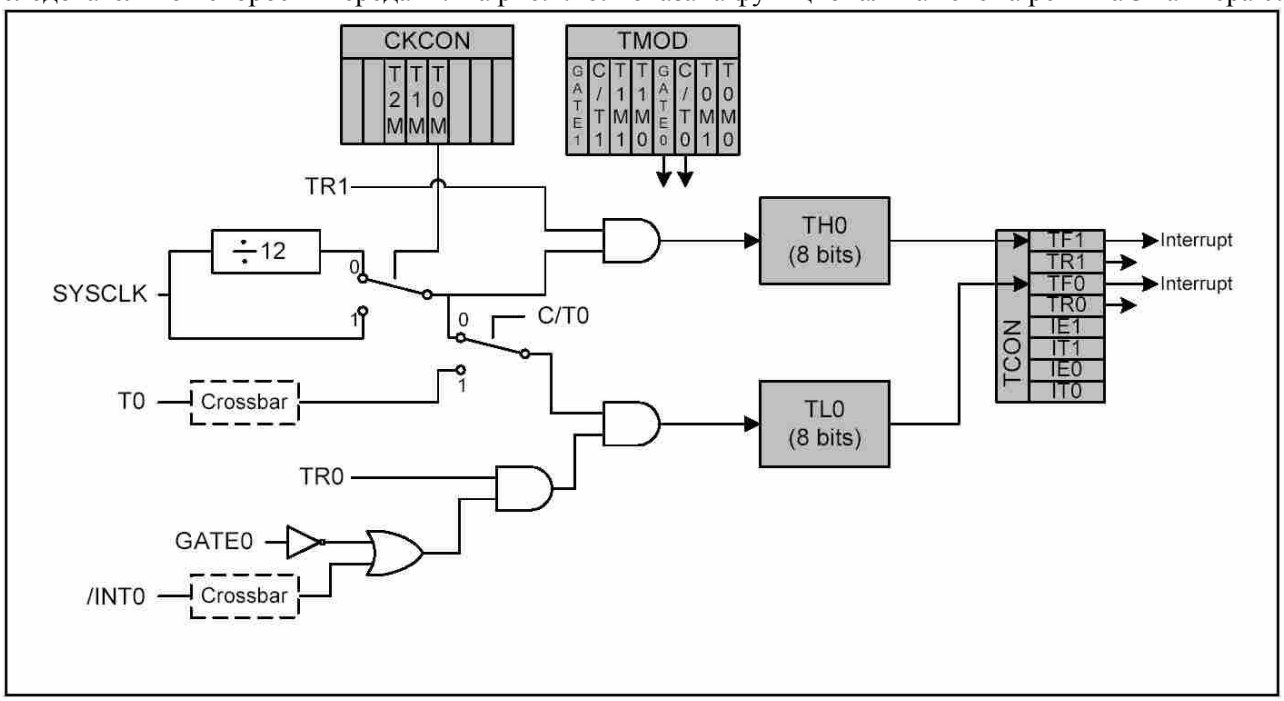

Рис.2.26. Функциональная схема режима 3 таймера 0

Таймер 2. Таймер 2 является 16-битным таймером/счетчиком, представленным в пространстве SFR двумя однобайтными регистрами: TL2 (младший байт) и TH2 (старший байт). Также как и таймеры 0 и 1, таймер 2 может использовать для работы системную тактовую частоту или вход внешней частоты. Источник тактирования определяется состоянием бита C/T2 (T2CON.1). Обнуление этого бита означает использование тактовой частоты, деленной на восемь или двенадцать, в за-

висимости от бита выбора частоты T2M регистра CKCON. Если C/T2 установлен, для счета используется задний фронт импульсов на входе внешней частоты T2. Кроме того, таймер 2 может использоваться для запуска аналого-цифрового преобразователя ADC.

Режимы работы таймера 2 отличаются от режимов работы таймеров 0 и 1. Он может функционировать в трех режимах: режиме 16-битного таймера/счетчика с захватом, режиме 16-битного таймера/счетчика с автозагрузкой или режиме генератора последовательной скорости. Режимы выбираются установкой битов в регистре T2CON, как показано в таблице 2.8.

Таблица 2.8.

Установка режимов таймера 2

RCLK	TCLK	CP/RL2	TR2	Режим
0	0	1	1	16-битный таймер/счетчик с захватом
0	0	0	1	16- битный таймер/счетчик с автозагрузкой
0	1	X	1	Генератор последовательной скорости для TX
1	0	X	1	Генератор последовательной скорости для RX
1	1	X	1	Генератор последовательной скорости для TX и
X	X	X	0	Выключен

Режим 0: 16-битного таймера/счетчика с захватом. При приходе заднего фронта импульса (перепада из верхнего логического уровня в нижний) на внешний вход T2EX, значение из счетных регистров (TH2 и TL2) перезаписывается в регистры захвата (RCAP2H и RCAP2L). Таймер 2 может использовать для счета системную тактовую частоту, тактовую частоту деленную на 12, а также внешние импульсы (по заднему фронту) на входе T2. Обнуление бита C/T2 (T2CON.1) означает использование для счета системных тактовых импульсов, деленных на 1 или на 12, в зависимости от состояния бита T2M регистра CKCON. Если бит C/T2 (T2CON.1) установлен, используются импульсы с внешнего входа T2. При переполнении регистров устанавливается флаг TF2 (T2CON.7) и может быть сгенерировано прерывание, если оно разрешено. Режим 16-битного таймера/счетчика с захватом задается установкой битов CP/RL2 (T2CON.0), TR2 (T2CON.2) и EXEN2 (T2CON.3). Если бит EXEN2 (T2CON.3) не установлен (равен 0), импульсы входа T2EX игнорируются. На рис.2.27. показана функциональная схема таймера 2 в режиме 0.

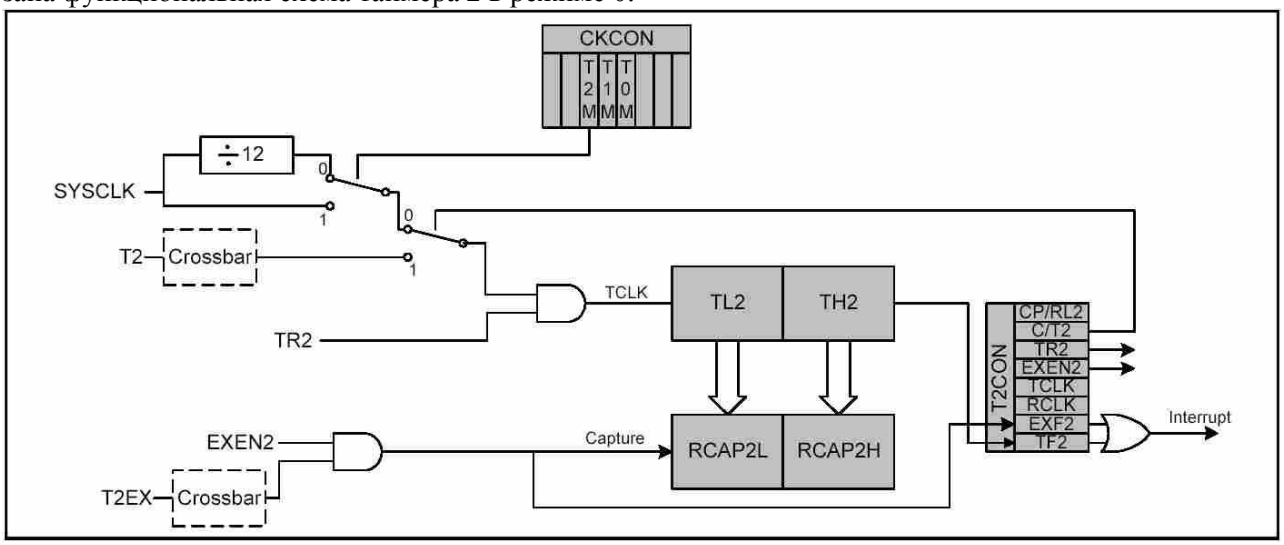

Рис.2.27. Функциональная схема таймера 2 в режиме 0

Режим 1: 16-битного таймера/счетчика с автозагрузкой. В этом режиме при переполнении счетчика устанавливается флаг TF2 и генерируется прерывание (если оно разрешено). Кроме того, шестнадцатеричное значение из регистров RCAP2H и RCAP2L автоматически перезаписывается в счетные регистры. Режим выбирается обнулением бита CP/RL2. Установка бита TR2 разрешает счет. В этом режиме таймер может использовать системные тактовые импульсы или внешний вход в качестве источника счетных импульсов. Источник выбирается битом C/T2. Если установлен EXEN2, задний фронт импульса на входе T2EX вызывает перезагрузку, иначе вход T2EX игнорируется. На рис.2.28. показана функциональная схема таймера 2 в режиме 1.

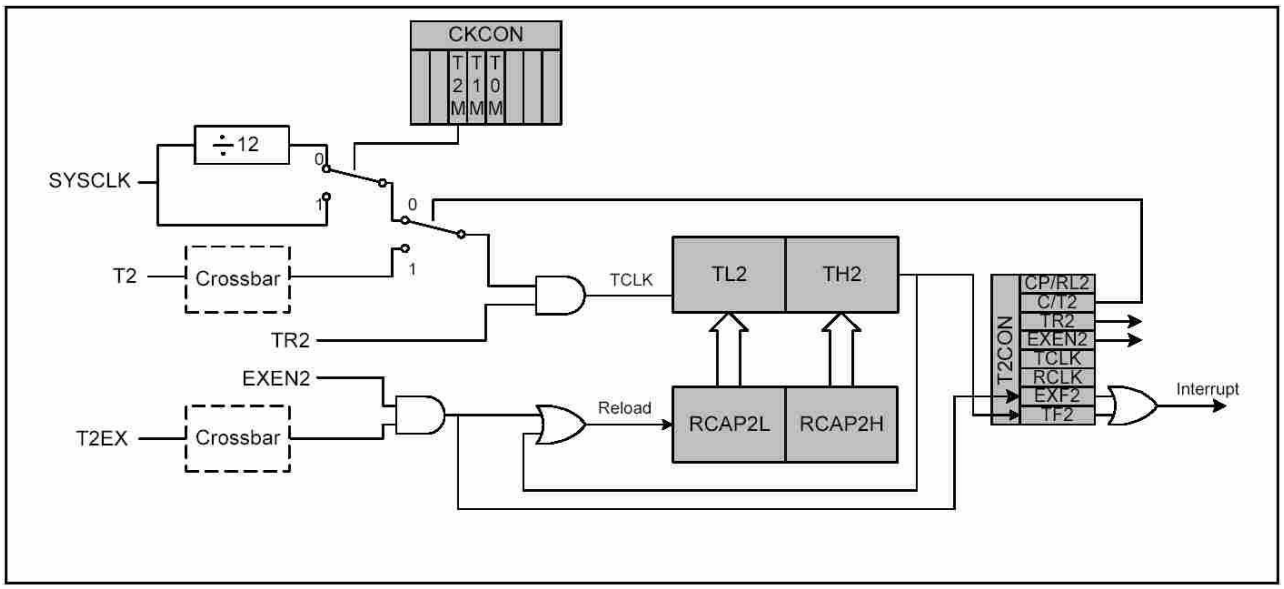

Рис.2.28. Функциональная схема таймера 2 в режиме 1

Режим 2: Генератор двоичной скорости. Таймер 2 может использоваться для генерации скорости последовательного порта UART, когда он работает в режиме 1 или 3. В этом режиме таймер работает с автозагрузкой, перезагружая значения из регистров RCAP2H и RCAP2L. При этом флаг TF2 не устанавливается, и прерывания не генерируются. Событие переполнения используется для тактирования UART, при этом возможно совместное или независимое генерирование скорости последовательного обмена для передачи и приема. Управление режимом осуществляется установкой битов RCLK (T2CON.5) и/или TCLK (T2CON.4). Когда оба бита установлены, таймер 2 работает в режиме автозагрузки независимо от состояния бита CP/RL2. При этом скорость UART определяется в соответствии с выражением:

Baud Rate = Timer 2 Overflow Rate /16.

Следует отметить важное исключение. Во всех остальных режимах, таймеры могут использовать системную тактовую частоту, деленную на один или на двенадцать, в зависимости от состояния бита T2M регистра CKCON. Но в рассматриваемом случае, когда таймер 2 используется в качестве генератора последовательной скорости передачи, тактовая частота делится на два.

Если необходимо использовать в качестве источника внешнюю частоту со входа T2, необходимо установить бит C/T2. В этом случае используется выражение:

*Baud Rate = FCLK / [32 * (65536-[RCAP2H:RCAP2L])]*

В этом выражении входная частота, поступающая на внешний вывод T2, обозначена как FCLK. Величина [RCAP2H:RCAP2L] это 16-битная величина делителя. Как было сказано выше, в этом режиме не устанавливается флаг TF2 и не генерируются соответствующие прерывания, но если будет установлен бит EXEN2, то задний фронт на входе T2EX установит флаг EXF2, а также может быть сформировано прерывание таймера 2. Следовательно, вход T2EX, в этом случае, может использоваться, как дополнительный источник прерываний. На рис.2.29. показана функциональная схема таймера 2 в режиме 2.

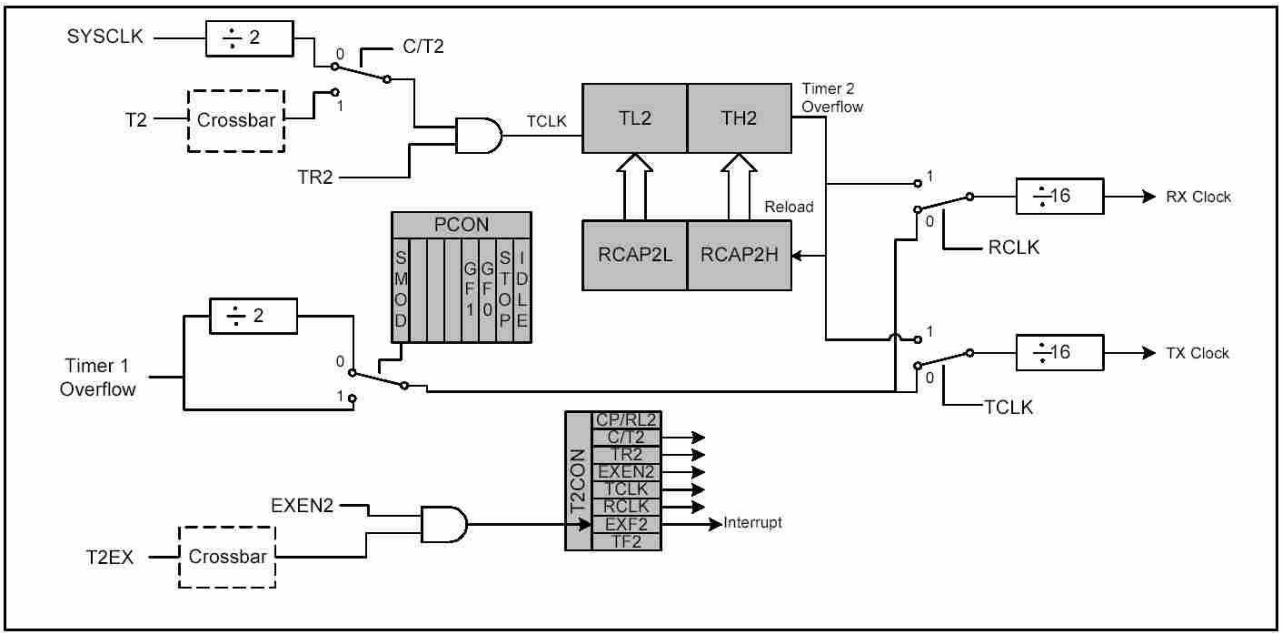

Рис.2.29. Функциональная схема таймера 2 в режиме 2

Таймер 3. Таймер 3 - это 16-битный таймер, который в SFR отображается двумя регистрами: TMR3L (младший байт) и TMR3H (старший байт). Таймер 3 работает от системных тактовых импульсов, деленных на один или на двенадцать, в зависимости от состояния бита T3M в регистре TMR3CN.

Таймер 3 всегда работает в режиме таймера с автозагрузкой (из регистров TMR3RLL и TMR3RLH) и не может работать в качестве счетчика. Он наиболее часто используется для перезапуска ADC или как тактовый генератор для интерфейса SMBus. На рис.2.30. показана функциональная схема таймера 3.

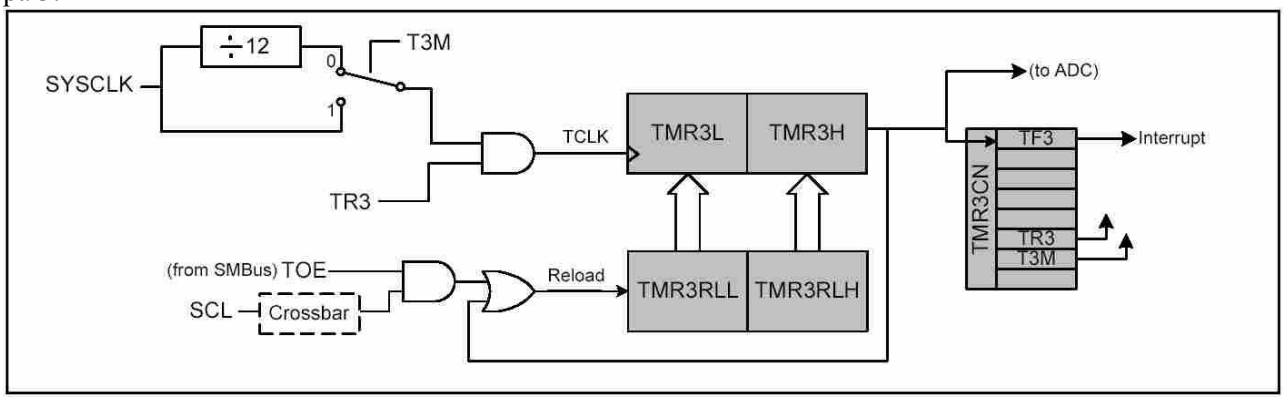

Рис.2.30. Функциональная схема таймера 3

2.11. Интерфейс программирования и отладки JTAG

Каждый из микроконтроллеров фирмы Cygnal имеет встроенный интерфейс JTAG, предназначенный для внутрисистемного программирования, тестирования и отладки программного обеспечения. Описываемый интерфейс полностью совместим с IEEE 1149.1 спецификацией. Для детального ознакомления со спецификацией и протоколом интерфейса следует ознакомится с этим документом. В рамках же данной книги мы рассмотрим только основные положения.

Доступ к интерфейсу JTAG осуществляется через регистры SFR: регистр инструкций IR и регистры данных DR. Интерфейс имеет четыре специальных вывода: TCK, TMS, TDI и TDO. Все выводы совместимы с пяти-вольтовой логикой. Через 16-битный регистр инструкций можно выполнять восемь инструкций, приведенных в таблице 2.9. Кроме того, существует три регистра данных (DR), связанных с выполнением этих инструкций.

Таблица 2.9.

Инструкции JTAG интерфейса

IR код	Инструкция	Описание
0x0000	EXTEST	Выбор регистра для контроля и управления выводами микроконтроллера.
0x0002	SAMPLE / PRELOAD	Выбор регистра для трассировки
0x0004	IDCODE	Выбор идентификационного регистра микроконтроллера
0xFFFF	BYPASS	Выбор промежуточного регистра данных
0x0082	Flash Control	Выбор FLASHCON регистра для управления чтения и записью в FLASHDAT регистр
0x0083	Flash Data	Выбор FLASHDAT регистра для чтения и записи во Flash память
0x0084	Flash Address	Выбор FLASHADR регистра для задания адреса операций чтения, записи и стирания Flash памяти
0x0085	Flash Scale	Выбор FLASHSCL регистра, который контролирует временные параменты всех операций

Для анализа состояния внутренних ресурсов микроконтроллера и его внешних выводов, интерфейс JTAG оснащен так называемым "регистром пограничного осмотра" (Boundary Scan Register), который представляет собой 87-битный сдвиговый регистр. Он используется инструкциями EXTEST и SAMPLE. В таблице 2.10 описаны биты Boundary Scan регистра. Инструкция EXTEST позволяет осуществлять захват (Чтение состояния в текущий момент времени) и обновление, а инструкция SAMPLE - только захват. Следует отметить, что некоторые микроконтроллеры фирмы Cygnal имеют большее количество портов, и соответственно, расширенный Boundary Scan регистр.

Таблица 2.10.

Назначение битов Boundary Scan регистра.

Номера битов	Действие	Назначение
0	Захват	Разрешение сброса от ядра
	Обновление	Разрешение сброса от вывода RST/
1	Захват	Вход сброса с вывода RST/
	Обновление	Выход сброса на вывод RST/
2	Захват	Внешнее тактирование с вывода XTAL1
	Обновление	Не используется
3	Захват	Разрешение подтяжки уровней от ядра
	Обновление	Разрешение подтяжки уровней портов
4-11	Захват	Адресная шина SFR от ядра CIP-5 1 (Bit4=SFRA0, Bit5=SFRA1)
	Обновление	Адресный бит SFR (Bit4=XSFRA0, Bit5=XSFRA1)
12-19	Захват	Бит шины данных SFR чтения из SFR (Bitl2=SFRD0,
	Обновление	Бит шины данных SFR записи в SFR (Bitl2=SFRD0,
20	Захват	Строб записи в SFR от ядра CIP-51
	Обновление	Строб записи на шину SFR
21	Захват	Строб чтения из SFR от ядра CIP-51
	Обновление	Строб чтения на шину SFR
22	Захват	Строб чтения -модификации -записи SFR от CIP-51
	Обновление	Строб чтения -модификации -записи на шину SFR
23,25,27,29, 31,33,35,37	Захват	Разрешение выходов P0.n от ядра (Bit23=P0.0, Bit25=P0.1, и т.д.)
	Обновление	Разрешение выходов P0.n для выводов (Bit23=P0.0, Bit25=P0.1)
24,26,28,30, 32,34,36,38	Захват	Вход P0.n с вывода (Bit24=P0.0, Bit26=P0.1, и т.д.)
	Обновление	Выход P0.n на вывод (Bit24=P0.0, Bit26=P0.1, и т.д.)
39,41,43,45, 47,49,51,53	Захват	Разрешение выходов P1.n от ядра (Bit39=P1.0, Bit41=P1.1, и т.д.)
	Обновление	Разрешение выходов P1.n для выводов (Bit39=P1.0, Bit41=P1.1)
40,42,44,46,	Захват	Вход P1.n с вывода (Bit40=P1.0, Bit42=P1.1, и т.д.)

48,50,52,54	Обновление	Выход P1.n на вывод (Bit40=P1.0, Bit42=P1.1, и т.д.)
55,57,59,61, 63,65,67,69	Захват	Разрешение выходов P2.n от ядра (Bit55=P2.0, Bit57=P2.1, и т.д.)
	Обновление	Разрешение выходов P2.n для выводов (Bit55=P2.0, Bit57=P2.1)
56,58,60,62, 64,66,68,70	Захват	Вход P2.n с вывода (Bit56=P2.0, Bit58=P2.1, и т.д.)
	Обновление	Выход P2.n на выход (Bit56=P2.0, Bit58=P2.1, и т.д.)
71,73,75,77, 79,81,83,85	Захват	Разрешение выходов P3.n от ядра (Bit71=P3.0, Bit73=P3.1, и т.д.)
	Обновление	Разрешение выходов P3.n для выводов (Bit71=P3.0, Bit73=P3.1)
72,74,76,78, 80,82,84,86	Захват	Вход P3.n с вывода (Bit72=P3.0, Bit74=P3.1, и т.д.)
	Обновление	Выход P3.n на вывод (Bit72=P3.0, Bit74=P3.1, и т.д.)

Для общего представления о возможностях JTAG интерфейса коротко рассмотрим его инструкции.

Инструкция EXTEST. Инструкция осуществляет доступ через регистр IR. Она позволяет контролировать состояния всех периферийных встроенных устройств, шины SFR регистров и подтягивающих элементов. Все выводы микроконтроллера устанавливаются в состояние логической единицы.

Инструкция SAMPLE. Инструкция также осуществляет доступ через регистр IR. Инструкция используется при трассировке и позволяет контролировать состояние входов микроконтроллера.

Инструкция BYPASS. Инструкция также осуществляет доступ через регистр IR. Инструкция обеспечивает доступ к однобитному JTAG регистру данных.

Инструкция IDCODE. Инструкция также осуществляет доступ через регистр IR. Она обеспечивает доступ к 32-битному регистру идентификации прибора.

Инструкции операций с Flash памятью. Flash память может программироваться непосредственно через JTAG интерфейс через регистры управления (Flash Control), данных (Flash Data), адреса (Flash Data) и масштаба (Flash Scale). Эти косвенные регистры данных ассоциированы с основным регистром инструкций JTAG. Операции чтения и записи в косвенные регистры могут быть выполнены предварительным заданием DR адреса в IR регистре. Каждая операция чтения или записи инициализируется предварительной записью "косвенного операционного кода" - IndOpCode (Indirect Operation Code) в выбранный регистр данных. Входной формат команды имеет 20 бит, причем старшие 19 и 18 занимает IndOpCode, а остальные - данные. В IndOpCode используются 3 комбинации: 10 - чтения, 11 - запись, остальные - опрос. Операция опроса используется для проверки бита занятости.

Более подробные сведения о работе интерфейса JTAG и его режимах приведены в [8].

2.12. Источник опорного напряжения

Источник опорного напряжения состоит из стабильного (15ppm/°C) генератора опорного напряжения напряжением 1.2В и буферного усилителя с коэффициентом усиления 2. Таким образом, внутренний источник имеет опорное напряжение - 2.4В. Кроме того, при необходимости, опорное напряжение может быть подано на внешний вывод, при этом ток с этого вывода на аналоговую землю AGND не должен превышать 200 мкА. Может также использоваться внешний источник опорного напряжения, которое подается на вывод VREF. При этом внутренний источник должен быть запрещен программно. *Внешний источник не может превышать напряжение аналогового питания AV+ более чем на 0.3В.* Управление буферным усилителем и источниками опорного напряжения осуществляется через регистр управления опорным напряжением REF0CN. Бит BIASE регистра REF0CN разрешает работу опорной цепи для ADC и DAC, а бит REFBE осуществляет выбор источника. Если оба бита установлены, аналоговые узлы используют внутренний источник опорного напряжения. Если используется внешний источник опорного напряжения, бит REFBE должен быть обнулен, а бит BIASE - установлен. Если в системе не будут использоваться ни ADC, ни DAC, оба бита должны быть обнулены. Во многих микроконтроллерах имеется температурный датчик, подключенный к одному из входов мультиплексора. Бит TEMPE регистра REF0CN разрешает или запрещает работу этого датчика. Функциональная схема подсистемы источника опорного напряжения показана на рис.2.31.

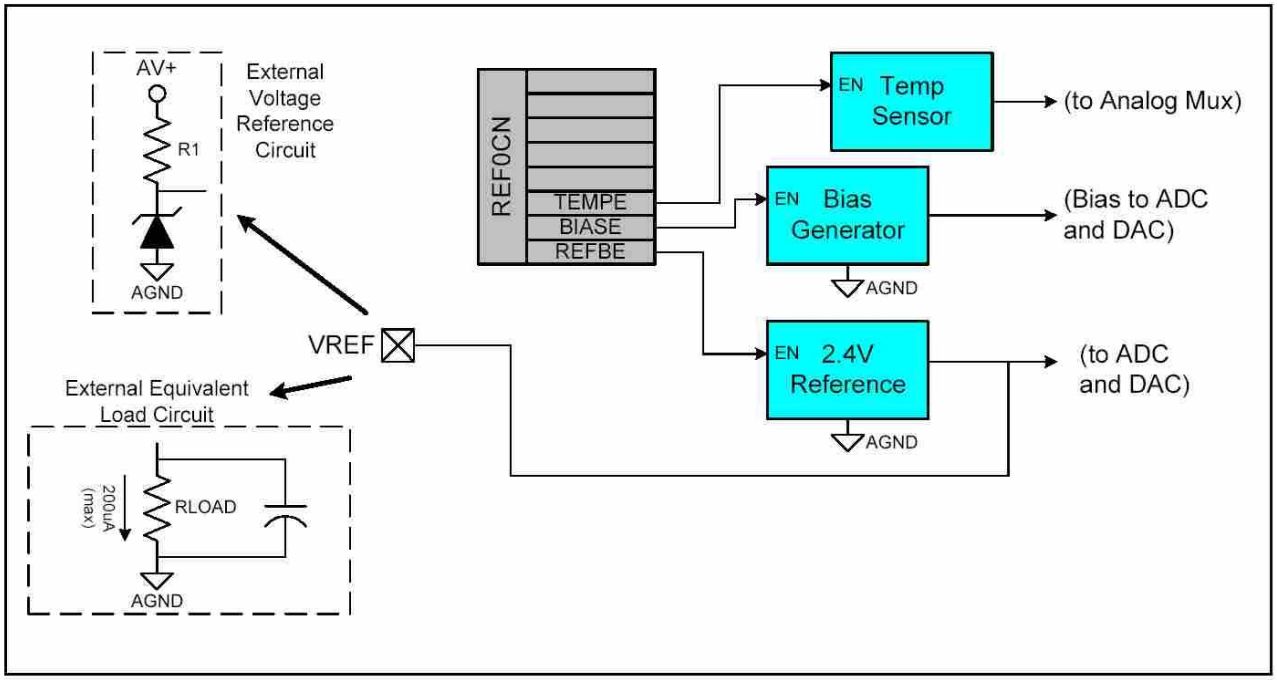

Рис.2.31. Функциональная схема источника опорного напряжения

2.13. Компараторы

Различные семейства микроконтроллеров фирмы Cygnal могут иметь два (или один) аналоговых компаратора напряжения. Выходы каждого аналогового компаратора через коммутатор ресурсов Crossbar могут быть подключены к выводам микроконтроллера. Функциональная схема полной подсистемы компараторов представлена на рис.2.32.

Каждый выход может работать, как обычный выход или как выход с "открытым истоком". Величина гистерезиса каждого компаратора может задаваться программно через соответствующий регистр управления компаратором (Comparator control register) (CPTOCN, CPT1CN). Пользователь может не только программировать величину гистерезиса, но и его симметрию, положительную и отрицательную величину вокруг определенного заданного значения. Выходы компараторов могут опрашиваться программно или генерировать прерывания.

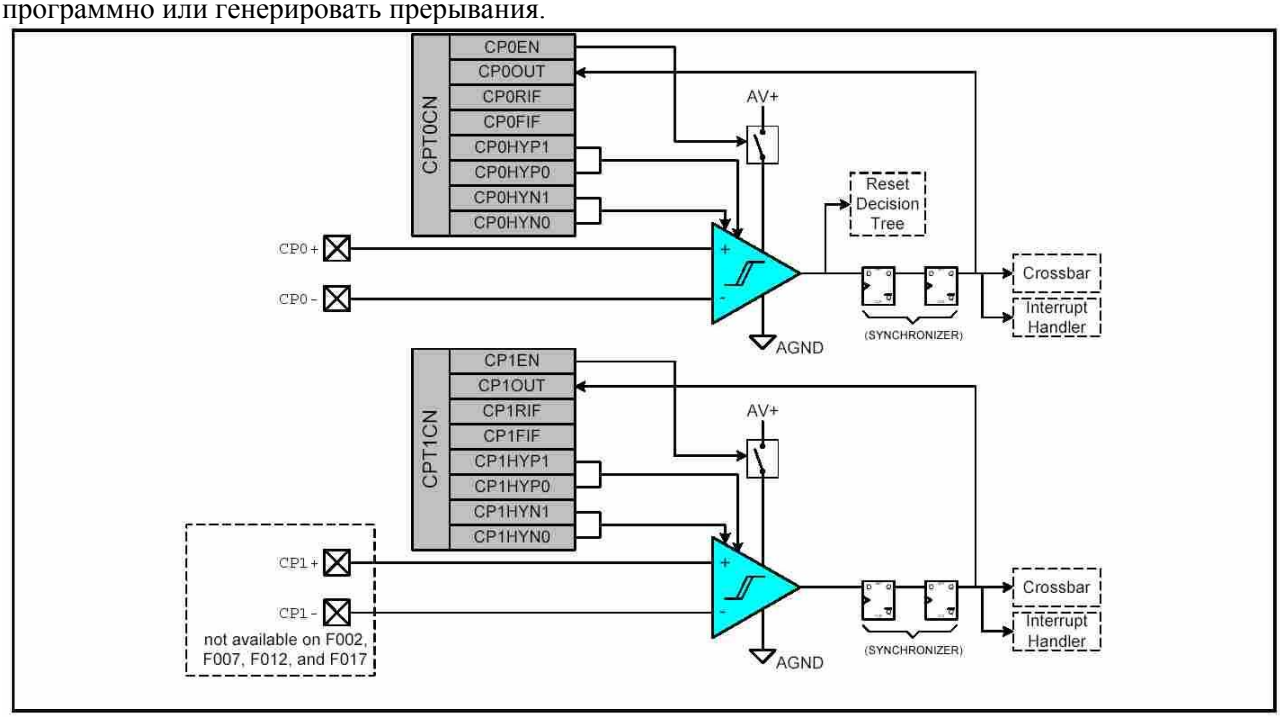

Рис.2.32. Функциональная схема подсистемы компараторов

Каждый из компараторов может быть индивидуально разрешен или запрещен. Если компаратор запрещен, то его выход, назначенный на внешний вывод через коммутатор ресурсов Crossbar, переводится в состояние логического нуля, а для того, чтобы приостановить его прерывания, ток питания соответствующих цепей снижается ниже 1 мкА. Входы компаратора 0 могут функционировать в диапазоне от -0,25В до {напряжения питания (AV+)+0,25В}. Гистерезис компаратора 0 программируется установкой битов 3-0 регистра CPT0CN. Отрицательный гистерезис может быть задан установкой бита CP0HYN. На рис.2.33.показано, как устанавливаются гистерезисы.

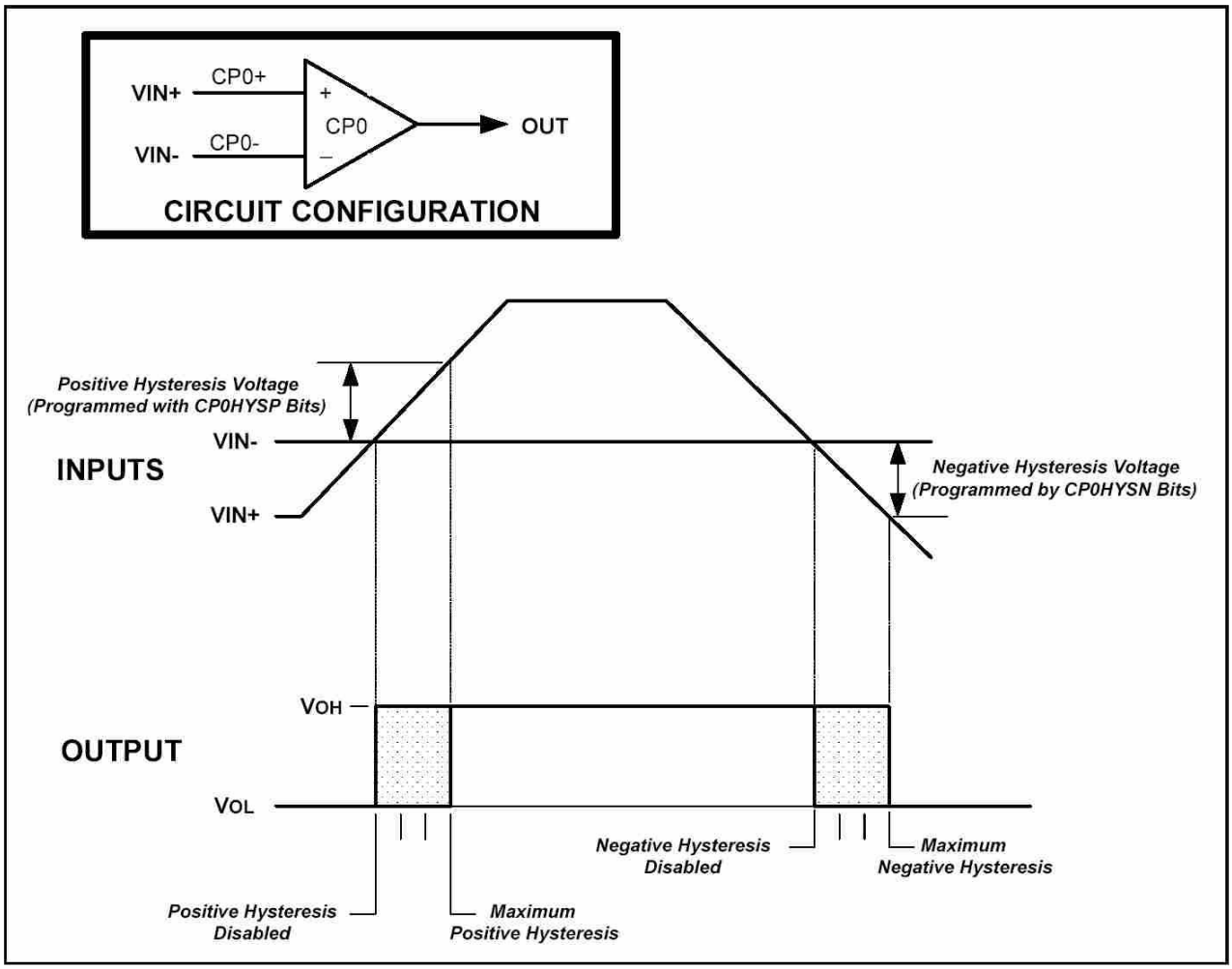

Рис.2.33. Установка гистерезисов компараторов

Прерывания компараторов могут вырабатываться при положительном или отрицательном перепаде уровней на выходе. Имеются также два флага, устанавливаемые по перепадам. Флаг CP0FIF устанавливается при отрицательном перепаде, а флаг CP0RIF - при положительном перепаде. Эит флаги устанавливаются аппаратно, а сбрасываются только ядром. Опрос состояния выхода компаратора 0 может быть произведен в любой момент времени путем чтения бита CP0OUT. Следует помнить, что и выход, и прерывания будут запрещены после включения питания до разрешения работы компаратора путем установки бита CP0EN. Кроме того, следует иметь ввиду, что значения на выходе компаратора стабилизируются в течение 20 mkS после установки бита CP0EN. Компаратор 0 может быть также запрограммирован, как источник ситуации сброса.

Операции с компаратором 1 идентичны операциям с компаратором 0, за исключением того, что компаратор 1 управляется регистром CPT1CN. Кроме того, компаратор 1 не может быть источником ситуации сброса.

2.14. Цифро-аналоговые преобразователи

Некоторые микроконтроллеры фирмы Cygnal оснащены двумя 12-битными цифро-аналоговыми преобразователями (DAC) с выходом по напряжению. Каждый DAC имеет диапазон выходных напряжений от 0 до VREF-1LSB (опорного напряжения минус один квант) для входных кодов от кода 0x000 до кода 0xFFF соответственно. При использовании, например DAC0, 12-битные

данные записываются в два регистра: младший байт DAC0L и старший байт DAC0H. Данные запоминаются после записи старшего байта, поэтому ***всегда следует соблюдать последовательность записи: сперва записывается младший байт DAC0L, а затем старший байт DAC0H!*** Цифро-аналоговые преобразователи DAC могут использоваться в 8-битном режиме путем инициализации регистра DAC0L величиной 0x00 и записью данных только в регистр DAC0H со сдвигом данных влево. Регистр управления DAC0CN позволяет разрешать/запрещать DAC0 с помощью бита DAC0CN.7 (1 разрешает работу DAC) и модифицировать его входные данные. Когда DAC запрещен, его выход переводится в высокоимпедансное состояние и его ток потребления снижается ниже величины 1 мкА. Для нормального функционирования необходимо бит BIASE в регистре REF0CN установить (в логическую единицу). И конечно же, для нормальной работы DAC необходимо сперва настроить источник опорного напряжения. Как уже отмечалось выше, в некоторых случаях входные данные для DAC должны быть предварительно подготовлены. Наиболее часто предварительная подготовка данных заключается в их сдвиге по разрядной сетка. Для облегчения программирования работы DAC, в регистре DAC0DF имеются три бита DAC0CN. [2:0], которые позволяют сделать эту операцию аппаратно (имеется 5 режимов).

Второй цифро-аналоговый преобразователь DAC1 работает аналогично первому. Функциональная схема цифро-аналоговых преобразователей показана на рис.2.34.

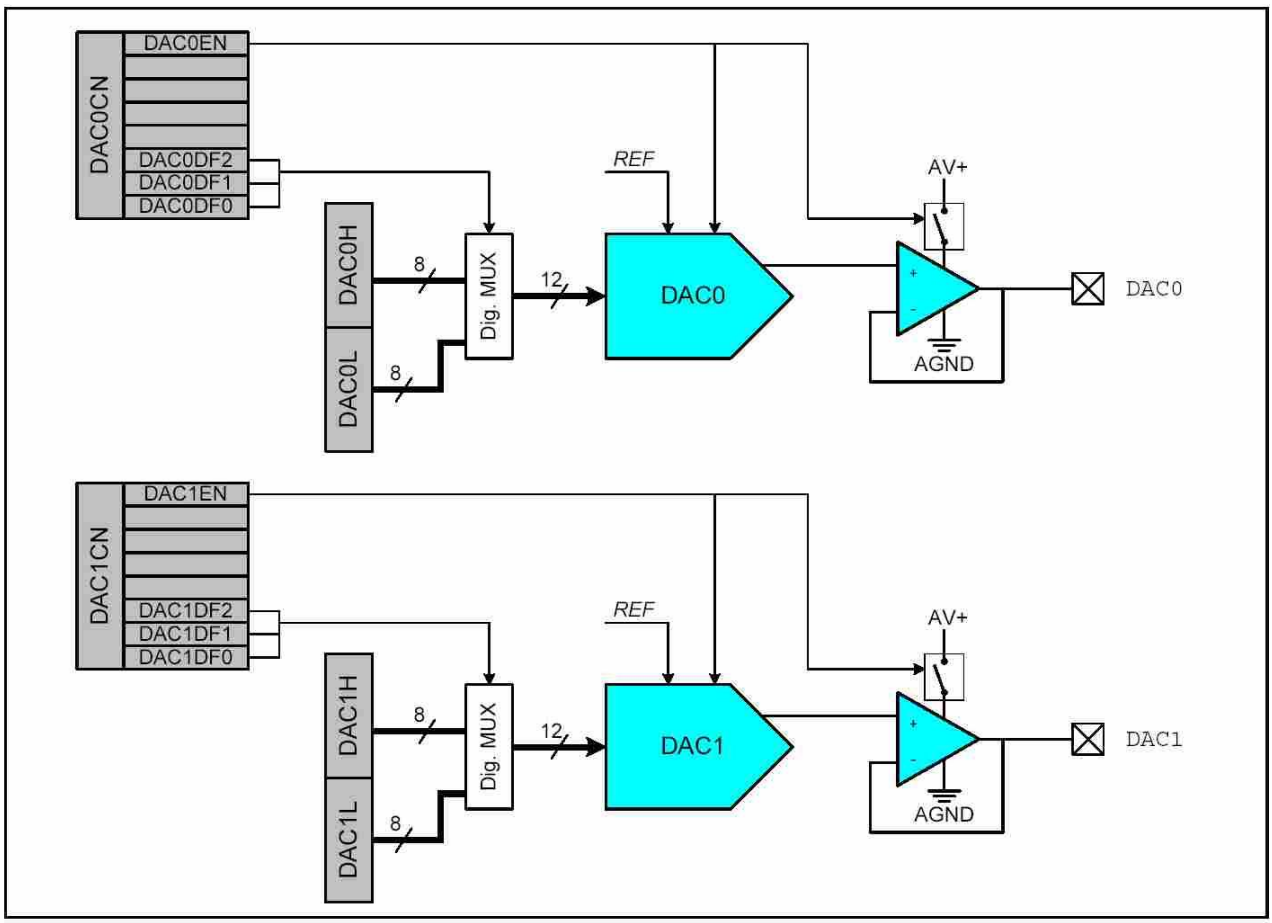

Рис.2.34. Функциональная схема цифро-аналоговых преобразователей

2.15. Аналого-цифровые преобразователи ADC

Различные микроконтроллеры фирмы Cygnal могут иметь один или два аналого-цифровых преобразователя. В данном разделе мы ознакомимся с описанием этих аналого-цифровых узлов в различных исполнениях.

2.15.1. Двенадцатибитные аналого-цифровые преобразователи

Некоторые микроконтроллеры фирмы Cygnal (например, C8051F000/1/2/5/6/7) имеют настраиваемые 9-ти канальные аналоговые мультиплексоры - AMUX (Analog Multiplexer), предвари-

тельные усилители с программируемым коэффициентом усиления PGA (Programmable Gain Amplifier) и 12-битный ADC следящего типа с детектором окна. Естественно, что все управление этими узлами сведено в несколько регистров SFR. Работа подсистемы разрешена только если бит ADCEN в регистре ADC0CN установлен. Если бит обнулен, подсистема переводится в режим энергосбережения. Кроме того, для работы подсистемы необходимо установить бит BIASE в регистре REF0CN. Функциональная схема 12-битного ADC представлена на рис.2.35.

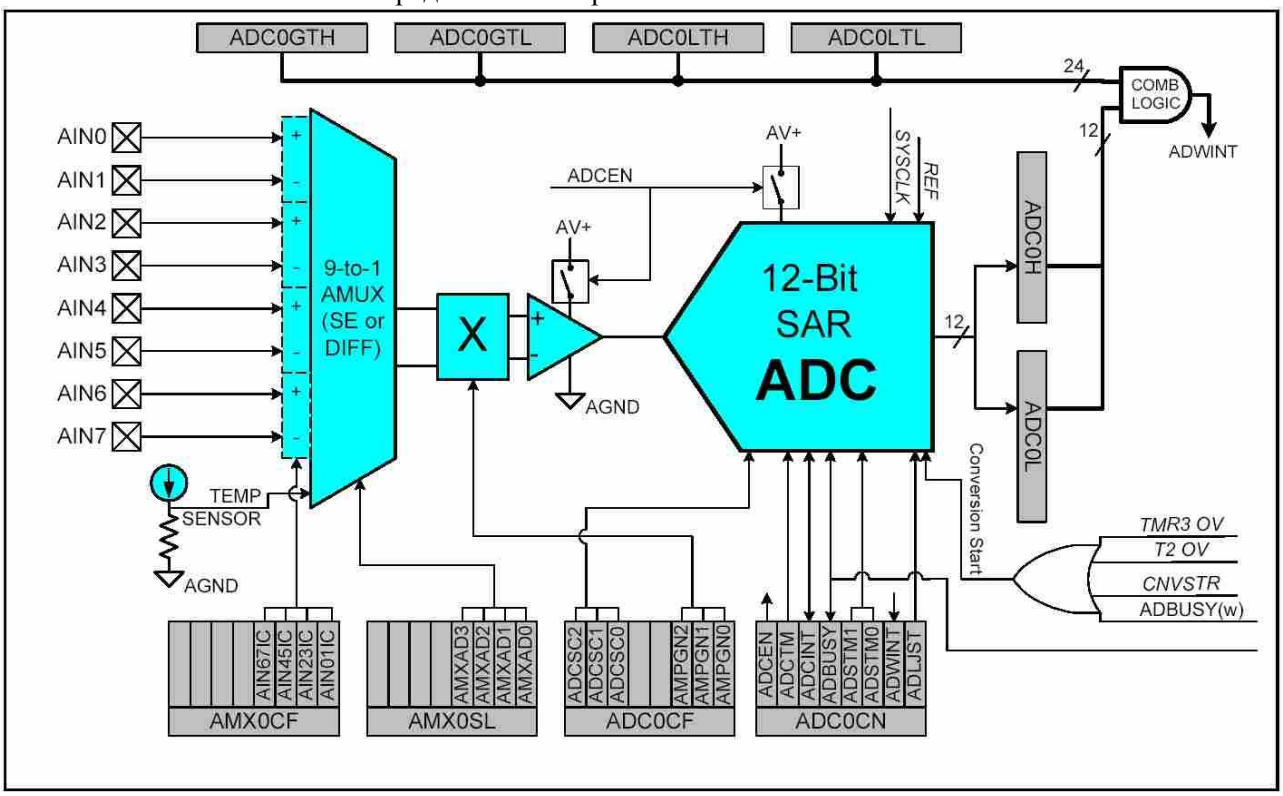

Рис.2.35. Функциональная схема 12-битного ADC

Аналоговый мультиплексор может иметь различное число входов. В большинстве первых микроконтроллеров число входов равно девяти, причем восемь из них могут использоваться для внешних измерений, а девятый используется для измерения температуры. Выходы аналогового мультиплексора соединены с входами программируемого усилителя. Входной мультиплексор можно запрограммировать так, что он будет работать в однополярном или дифференциальном режиме. Перепрограммирование входов возможно "на лету". После сброса мультиплексор работает, как восемь однополярных входов. Имеются два регистра, связанные с работой аналогового мультиплексора: первый регистр выбора канала (Channel Selection register) AMX0SL и второй - регистр конфигурации (Configuration register) AMX0CF. Коэффициент усиления усилителя также определяется битами AMPGN2-0 регистра конфигурации ADC0CF. Коэффициент усиления может быть различным у различных моделей микроконтроллеров, обычно он равен 0.5, 1, 2, 4, 8 или 16. После сброса он устанавливается равным 1. В некоторых моделях коэффициент усиления изменяется от 0.5 до 4, а в других вообще неизменен. Полные сведения об этом читатель найдет в главах, описывающих конкретные семейства.

Аналого-цифровой преобразователь использует опорное напряжение VREF для формирования полной шкалы квантования, поэтому оно должно быть задано до начала работы ADC. В разных моделях микроконтроллеров имеются ADC с двумя типами быстродействия: среднего быстродействия (до 100 ksps) и высокого быстродействия (до 500 ksps). Частота преобразования ADC вырабатывается из системной тактовой частоты. Скорость преобразования может быть замедлена на 2,4,8 или 16 установкой битов ADCSC в регистре ADC0CF. Это используется для выравнивания скорости преобразования при работе на различных тактовых частотах. Запуск преобразования возможен одним из четырех способов в зависимости от состояния битов запуска (Start of Conversion Mode bits) (ADSTM1, ADSTM0) в регистре ADC0CN:

1. Установкой бита ADBUSY регистра ADC0CN;
2. По переполнению таймера 3;

3. По фронту внешнего сигнала CNVSTR;
4. По переполнению таймера 2.

Установка бита ADBUSY удобна для программного управления, т.к. бит остается в 1 до завершения преобразования и сбрасывается в 0. Кроме того, может вырабатываться прерывание, если оно разрешено, устанавливаться флаг, данные в регистрах MSB (старшие биты) и LSB (младший байт). Кроме того, данные могут выравниваться по младшему или старшему разряду. Бит ADCTM в регистре ADC0CN управляет режимом отслеживания. В исходном состоянии вход ADC непрерывно отслеживается, за исключением времени преобразования. Установка бита ADCTM позволяет задать один из четырех режимов слежения в зависимости от состояния битов ADSTM1-0 регистра ADC0CN:

1. Слежение начинается с установки бита ADBUSY и продолжается до 3 тактов;
2. Слежение начинается с момента переполнения таймера 3 и продолжается до 3 тактов;
3. Слежение начинается только когда на вход CNVSTR поступает низкий уровень;
4. Слежение начинается с момента переполнения таймера 2 и продолжается до 3 тактов;

Режимы 1, 2 и 4 используются когда пуск преобразования производится программно или когда ADC работает непрерывно. Режим 3 используется для аппаратного запуска ADC. В этом случае механизм отслеживания находится в режиме низкого энергопотребления, пока на входе CNVSTR высокий уровень. Естественно, что ADC можно также перевести в режим энергосбережения (shutdown). Временные соотношения работы ADC показаны на рис.2.36.

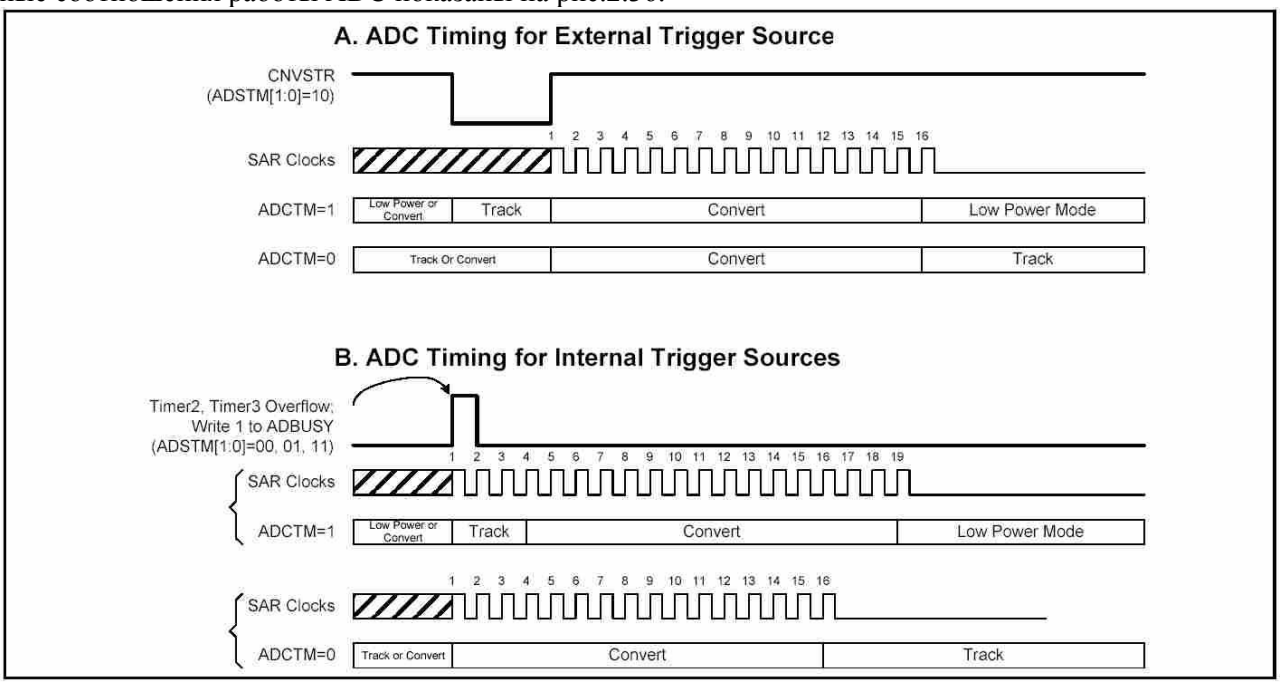

Рис.2.36. Временные диаграммы работы ADC

На рис.2.37. показана передаточная функция температурного датчика, подключенного к одному из входов ADC при установленном единичном коэффициенте усиления.

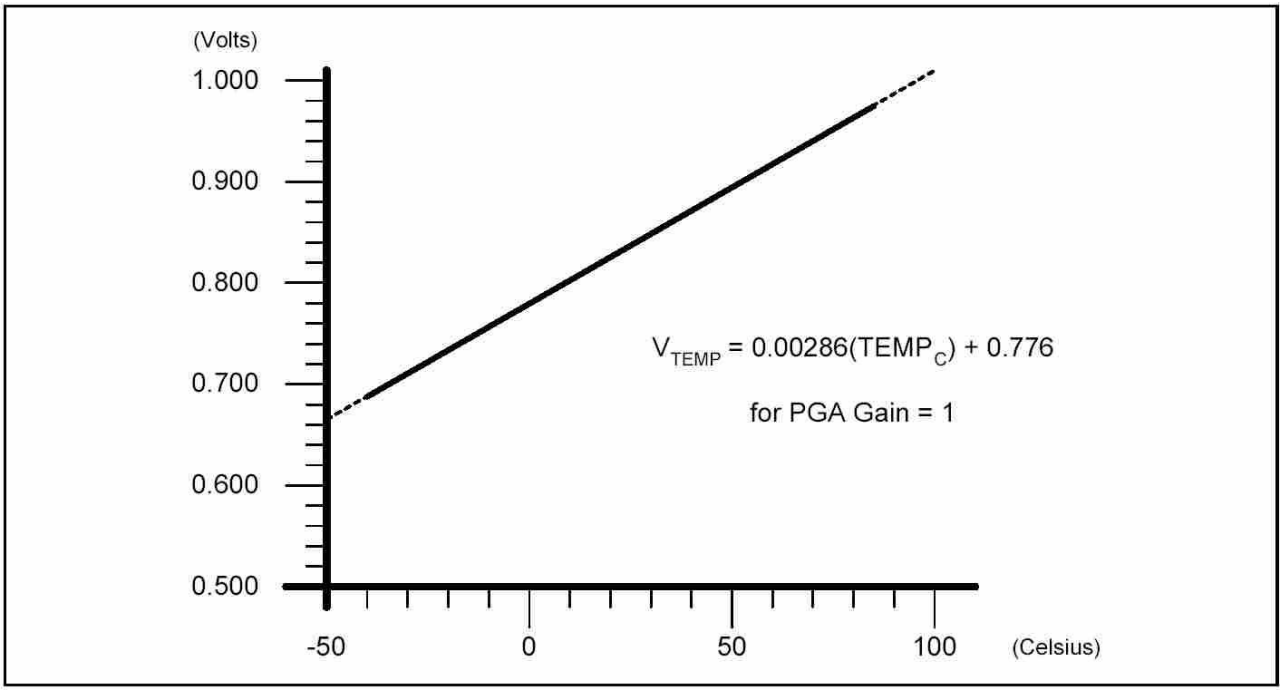

Рис.2.37. Передаточная функция температурного датчика

Важной особенностью подсистемы ADC является наличие программируемого "оконного детектора" ADC. Оконный детектор имеет два регистра, программируемых пользователем, в которые он может записать две 12-битные (в общем случае) величины - два порога. Интервал величин (кодов) между этими порогами определяет некое "окно" выходных кодов. Узел оконного детектора постоянно отслеживает выходной код ADC и сравнивает его с заданными в регистрах величинами порогов, и если выходной код выходит за пределы заданных порогов (иными словами, за пределы "окна"), узел может информировать систему об этом. Информирование может производиться путем подачи соответствующего прерывания или программным опросом флага ADWINT в регистре ADC0CN. Регистры верхнего порога (ADC Greater-Than) - ADC0GTH и ADC0GTL.. Регистры нижнего порога (ADC Less-Than) - ADC0LTH и ADC0LTL. Следует также отметить, что установка флага ADWINT в регистре ADC0CN может быть настроена на срабатывание вне окна или внутри окна путем задания регистров ADC0GTx и ADC0LTx.

2.15.2. Десятибитные аналого-цифровые преобразователи

Некоторые микроконтроллеры фирмы Cygnal (например, C8051F010/11/12/15/16/17 и другие) имеют не 12, а 10-битные ADC. Функциональная схема десятибитного аналого-цифрового преобразователя представлено на рис.2.38, а описание полностью соответствует приведенному в разделе 2.15.1, естественно за исключением того, что ADC десятибитный.

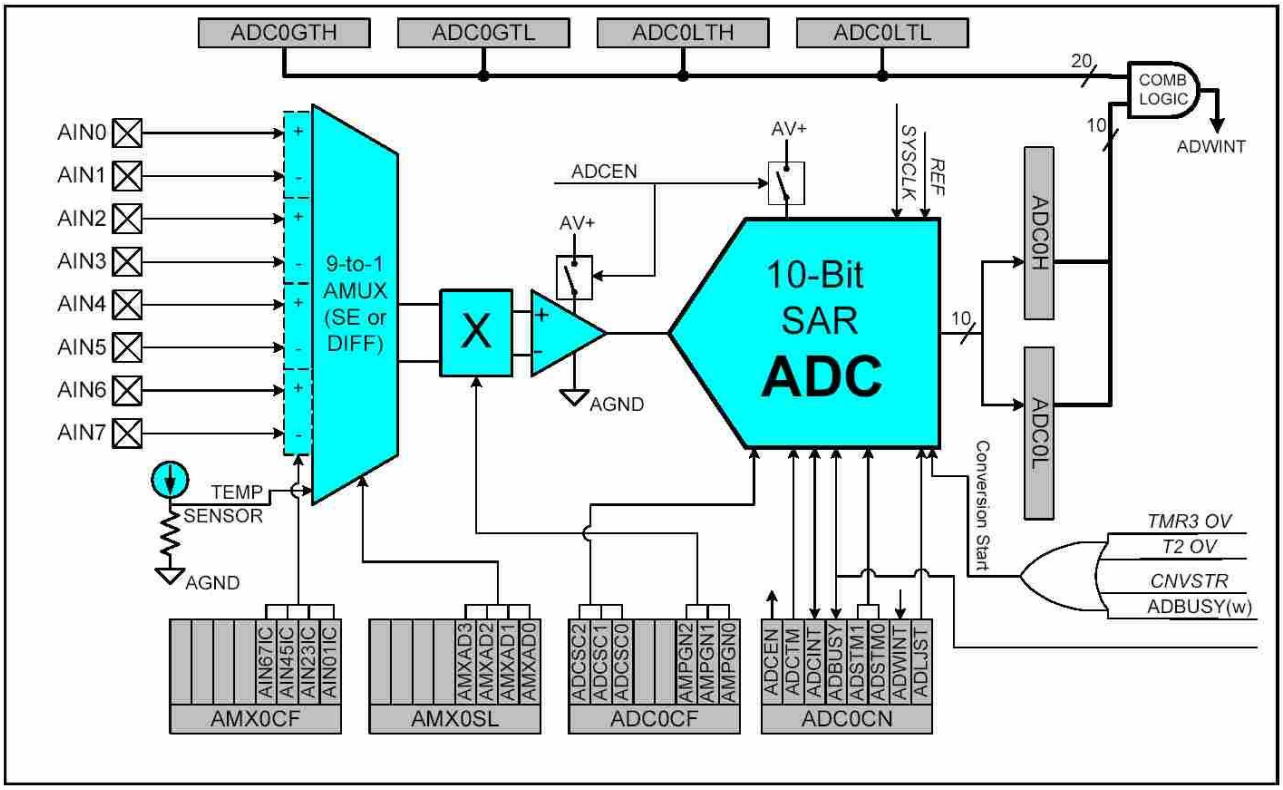

Рис.2.38. Функциональная схема 10-битного ADC

2.15.3. Восьмибитные аналого-цифровые преобразователи

Некоторые микроконтроллеры фирмы Cygnal (например, C8051F020-040) имеют (второй) быстродействующий 8-битный аналого-цифровой преобразователь ADC1, настраиваемый 8-ти канальный аналоговый мультиплексор - AMUX1, предварительный усилитель с программируемым коэффициентом усиления PGA1. Естественно, что все управление этими узлами сведено в несколько регистров SFR. Функциональная схема 8-битного ADC представлена на рис.2.39.

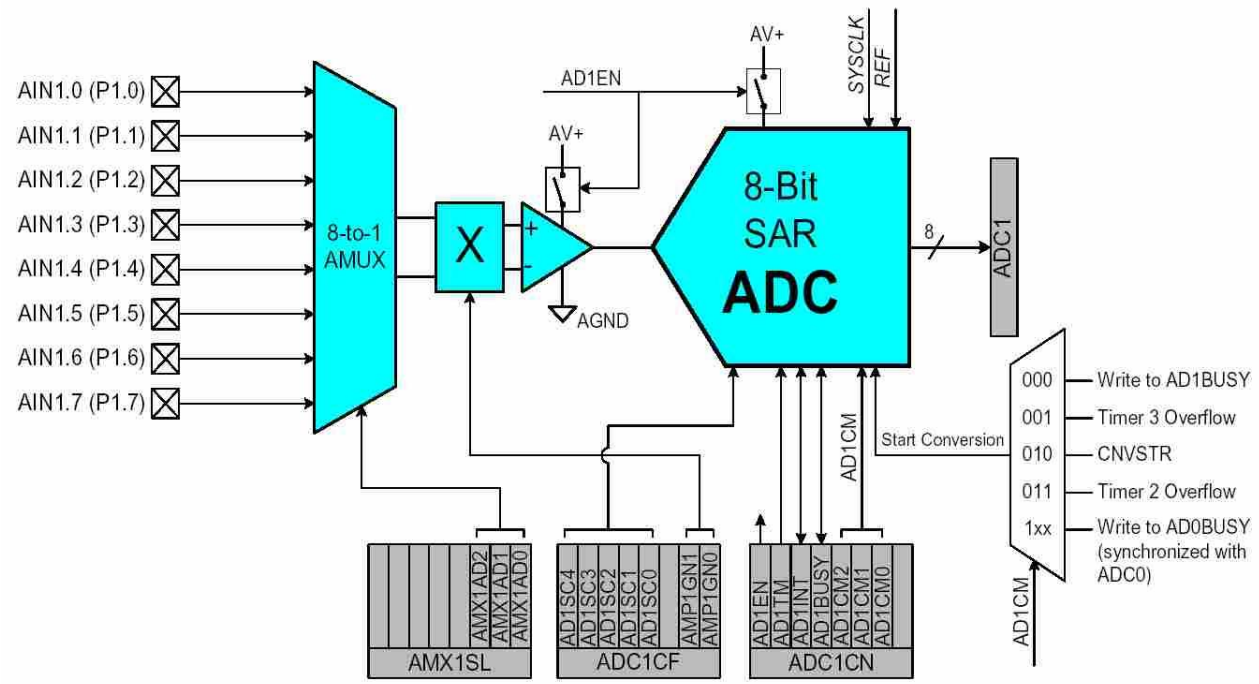

Рис.2.39. Функциональная схема 8-битного ADC

Функционирование этого преобразователя, его функциональная схема и SFR регистры очень похожи на приведенные в разделе 2.15.1.

В заключение следует еще раз подчеркнуть, что в этой главе коротко описаны только базовые варианты реализации основных узлов. Реализация одних и тех же узлов в микроконтроллерах фирмы Cygnal постоянно модернизируется. По видимому, разработчики фирмы Cygnal ищут оптимальные функциональные решения. Незначительные отличия одинаковых функциональных узлов различных семейств микроконтроллеров поясняются при описании соответствующих регистров специального назначения. Таким же образом модернизируются и встроенные интерфейсы, в которые от семейства к семейству появляются все новые и новые возможности.