16-разрядное ядро ЦПУ F²MC-16LX

В статьях, опубликованных ранее в нашем журнале (№ 5 и 7'2001), посвященных микроконтроллерам Fujitsu серии F²MC-16LX (12), были представлены основные особенности процессоров данного семейства и интегрированная система разработки программного обеспечения Softune Workbench. В данной статье будут кратко рассмотрены особенности ядра этих процессоров.

Дмитрий Киселев

design@cec-mc.ru

Контроллеры семейства F²MC-16LX имеют 16-разрядное ядро, разработанное для приложений, которые требуют оперативной обработки с высоким быстродействием, как, например, бытовое радиоэлектронное и электрическое оборудование. Система команд F²MC-16LX позволяет быстро и эффективно осуществлять обработку данных. В дополнение к 16-разрядным данным, ядро ЦПУ F²MC-16LX может обрабатывать 32-разрядные данные, используя внутренний 32-разрядный аккумулятор (32-разрядные данные могут быть обработаны некоторыми командами). В системе может быть использовано до 16 Мбайт памяти, обращение к которой возможно с помощью линейного указателя или с использованием банков. Структура инструкций, основанная на архитектуре F²MC-8A-T, была улучшена: добавлены команды, совместимые с языками высокого уровня, расширены способы адресации, улучшены команды умножения и деления, расширена побитовая обработка. Свойства ЦПУ F²MC-16LX перечислены ниже.

- Минимальное время выполнения инструкции — 62,5 нс (при 4 МГц).
- Максимальный объем памяти — 16 Мбайт (обращение в линейном режиме или с использованием банков).
- Система команд оптимизирована для использования контроллером.
- Различные типы данных: бит, байт, слово, длинное слово.
- Расширенные способы адресации — 23 типа.
- Работа с высокой точностью (длина 32 бита), основанная на 32-разрядном аккумуляторе.
- Восьмиуровневая программируемая система прерываний.
- Независимая от ЦПУ автоматическая передача данных между устройствами в стиле ПДП. До 16 каналов расширенного обслуживания ввода-вывода.
- Система команд, совместимая с языками высокого уровня, например, Си.
- Усовершенствованная скорость выполнения — 4-байтовая очередь.

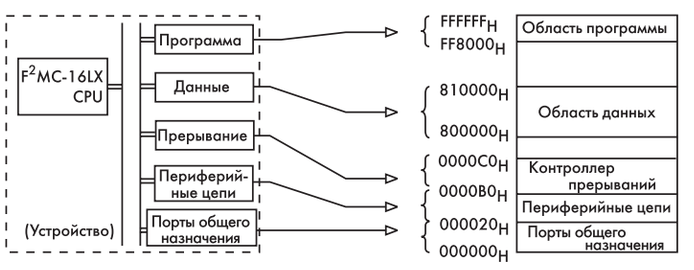

Рис. 1. Взаимоотношение между системой F²MC-16LX и картой памяти

Пространство памяти

ЦПУ F²MC-16LX имеет адресуемое пространство памяти емкостью 16 Мбайт. Все входные и выходные данные программы размещены в этом пространстве. ЦПУ обращается к ресурсам, указывая их адреса с использованием 24-разрядной адресной шины.

На рис. 1 показано взаимоотношение между системой F²MC-16LX и картой памяти.

F²MC-16LX имеет два типа адресации:

- *Линейная адресация:*
  24-разрядный адрес определяется командой.
- *Банковая адресация:*
  восемь старших битов адреса определяются соответствующим банковым регистром, а оставшиеся 16 битов младшего разряда определяются командой.

На рис. 2 показано типичное разбиение адресного пространства на банки, при этом показано значение соответствующих регистров банков.

Рассмотрим подробнее режимы линейной и банковой адресации.

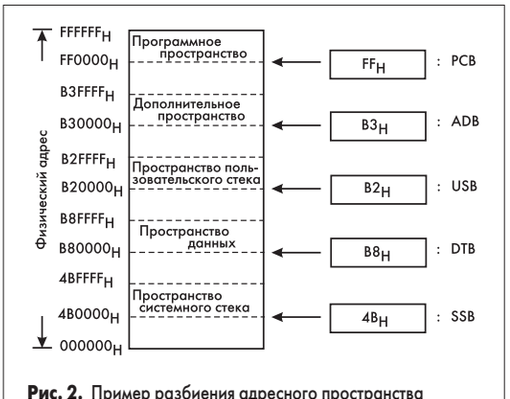

Рис. 2. Пример разбиения адресного пространства с использованием банков

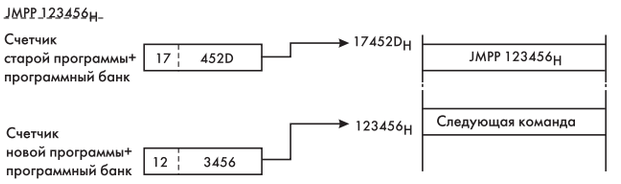

Рис. 3. Пример линейного метода (24-разрядная спецификация операнда)

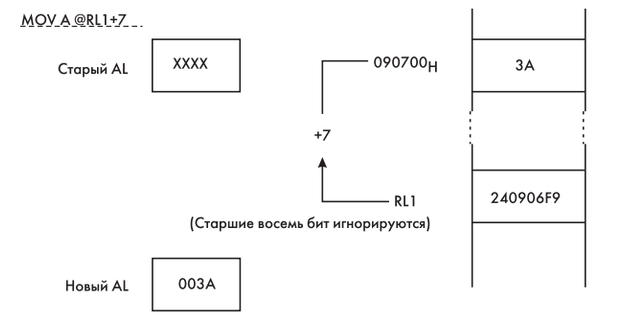

Рис. 4. Пример линейного метода (32-разрядная косвенно-регистровая спецификация)

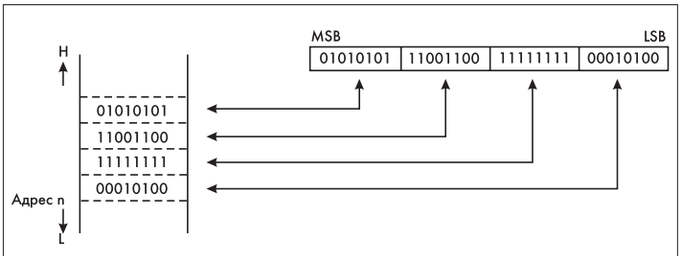

Рис. 5. Пример распределения многобайтных данных в памяти

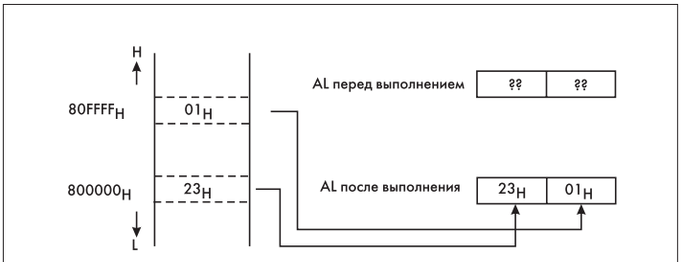

Рис. 6. Выполнение команды MOVW A, 080FFFFH

Линейная адресация

Ядро поддерживает два типа линейной адресации:

- 24-разрядная спецификация операнда: непосредственно определяет 24-разрядный адрес, используя операнды (рис. 3).
- 32-разрядная косвенно-регистровая спецификация: косвенно определяет 24 младших бита 32-разрядной величины универсального регистра как адрес (рис. 4).

Режим адресации с использованием банков

При банковом методе адресации пространство памяти 16 Мбайт разделено на 256 банков по 64 кбай каждый, которые определяются следующими пятью банковыми регистрами:

- регистр банка программы (PCB);
- регистр банка данных (DTB);
- регистр банка стека пользователя (USB);
- регистр банка системного стека (SSB);
- дополнительный банковый регистр (ADB).

Регистр банка программы (PCB)

Банк памяти размером 64 кбай, определенный регистром PCB, называется программным пространством (PC). Пространство PC содержит команды, таблицы векторов и непосредственные значения данных.

Регистр банка данных (DTB)

Банк памяти, определенный регистром DTB, называется пространством данных (DT). Пространство DT содержит читаемые или перезаписываемые данные и регистры управления или данных для внутренних и внешних ресурсов.

Регистр банка стека пользователя (USB) и регистр банка системного стека (SSB)

Банк памяти, определенный регистром USP или SSP, определяет пространство стека (SP). К пространству SP производится обращение во время выполнения команд push/pop или при работе с прерываниями. При этом флаг S в регистре состояния определяет пространство стека, к которому нужно обратиться.

Дополнительный банковый регистр (ADB)

Банк памяти, определенный регистром ADB, называется дополнительным пространством (AD). Пространство AD может содержать данные, которые, например, не поместились в пространство DT.

После сброса регистры DTB, USB, SSB и ADB инициализированы значением 00h. PCB инициализирован значением, установленным вектором сброса. Пример разделения адресного пространства на банки показан на рис. 2.

Многобайтные данные в пространстве памяти

Многобайтные данные хранятся в памяти таким образом, что младшие байты расположены по младшим адресам. Отметим, что если сигнал сброса приходит сразу после того, как младшие биты записаны, то старшие биты могут не быть записаны. На рис. 5 показан пример расположения в памяти 32-битного слова. Младшие 8 бит элемента данных сохраняются по адресу *n*, затем по адресу *n*+1, адресу *n*+2 и т. д.

Доступ к многобайтным данным

Доступ к данным осуществляется в пределах банка. Поэтому, например, для команды, обращающейся к многобайтному элементу данных, адрес 0000h, следующий за адресом FFFFh, определяется в том же самом банке (рис. 6).

Регистры

Регистры F²MC-16LX можно разделить на два типа: специальные регистры ЦПУ и регистры общего назначения в памяти. Специальные регистры — это выделенные внутренние аппаратные средства ЦПУ, которые ориентированы на специфическое использование, определенное архитектурой ЦПУ. Регистры общего назначения используют адресное пространство совместно с ОЗУ. Работа с регистрами общего назначения может быть идентична работе с обычным пространством памяти.

Специальные регистры

Ядро ЦПУ F²MC-16LX содержит 13 специальных регистров:

- Аккумулятор (A=AH:AL): может использоваться как одиночный 32-битный аккумулятор или как два 16-битных аккумулятора.
- Пользовательский указатель стека (USP): 16-битный указатель, указывающий область пользовательского стека.
- Системный указатель стека (SSP): 16-битный указатель, определяющий системную область стека.
- Слово состояние процессора (PS): 16-битный регистр, отражающий состояние системы.
- Программный счетчик (PC): 16-битный регистр, содержащий адрес инструкции.
- Регистр банка программы (PCB): 8-битный регистр, указывающий пространство PC.
- Регистр банка данных (DTB): 8-битный регистр, указывающий пространство DT.
- Регистр банка стека пользователя (USB): 8-битный регистр, указывающий пространство пользовательского стека.
- Регистр банка системного стека (SSB): 8-битный регистр, указывающий пространство системного стека.
- Дополнительный банковый регистр (ADB): 8-битный регистр, указывающий пространство AD.

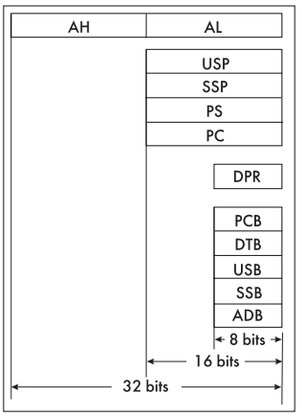

Рис. 7. Регистры специального назначения

- Регистр страницы с прямой адресацией (DPR): 8-битный регистр, указывающий страницу с прямой адресацией.

Регистры общего назначения

Регистры общего назначения расположены в оперативной памяти по адресам 000180h — 00037Fh. Указатель банкового регистра (RP) показывает, какие из вышеупомянутых адресов в настоящее время используются как банковый регистр. Каждый банк имеет три следующих типа регистров. Эти регистры взаимосвязаны, как показано на рис. 8.

- R0… R7: 8-битный регистр общего назначения;
- RW0… RW7: 16-битный регистр общего назначения;
- RL0… RL3: 32-битный регистр общего назначения.

Соотношения между старшими и младшими байтами регистра байта или слова определяются следующим образом:

$$RW(i+4) = R(i\times2+1)\times256+R(i\times2)\ [i=0\ldots3]$$

Соотношения между старшими и младшими байтами RLi и RW определяются следующим выражением:

$$RL(i) = RW(i\times2+1)\times65536+RW(i\times2)\ [i=0\ldots3]$$

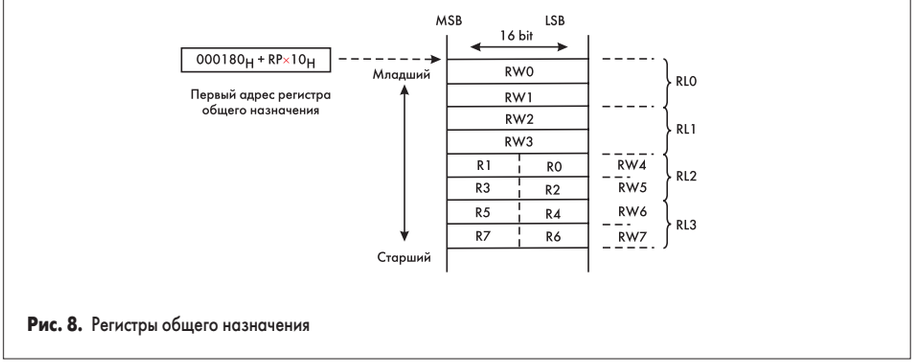

Рис. 8. Регистры общего назначения

Регистровый банк

Банк регистров состоит из 8 слов. Его можно использовать как регистры общего назначения для арифметических операций (байты — регистры R0… R7, слова — регистры RW0… RW7, и длинные слова RL0… RL3), а также как указатели команд. В таблице 1 приведены функции регистров. Значения банка регистров не инициализируются сбросом. Состояние, которое было до сброса, сохраняется. После включения питания банк регистров будет иметь неопределенное значение.

Таблица 1

R0...R7	Используются как операнды команд. Примечание: R0 используется также как счетчик для циклического сдвига.
RW0...RW7	Используются в качестве указателей и операндов команд. Примечание: RW0 используется также как счетчик для команд, работающих со строкой.
RL0...RL3	Используются как длинные указатели и операнды команд.

Аккумулятор (A)

Регистр — аккумулятор (A) состоит из двух 16-битных регистров AH и AL и используется как временное хранилище результатов операций и передаваемых данных. Во время обработки 32-битных данных регистры AH и AL используются совместно. При работе со словами в режиме обработки 16-битных данных или для работы с байтами в режиме обработки 8-битных данных используется только AL (рис. 9 и 10). Различные типы арифметических и логических операций могут выполняться между данными, сохраненными в аккумуляторе и данными в памяти или регистрах общего назначения Ri (RWi, RLi). Когда слово или более короткий элемент данных перенесены в AL, предшествующий элемент данных в AL автоматически посылается в AH (функция сохранения данных). Операции между AL и AH предназначены для повышения эффективности обработки. Когда байт или более короткий элемент данных перенесен в AL, данные, расширенные знаком или нулем, сохраняются как 16-битный элемент данных в AL. При этом данные в AL могут быть обработаны как слово или как длинный байт. Регистр A не инициализируется сбросом и сразу после сброса имеет неопределенное значение.

Указатель стека пользователя (USP) и системный указатель стека (SSP)

USP и SSP — 16-разрядные регистры, которые указывают адрес памяти для сохранения и восстановления данных, при выполнении подпрограмм или команд pop/push. Регистр USP разрешен, когда флаг S в процессорном регистре состояния равен '0', а регистр SSP допускается, когда флаг S='1' (рис. 11). Регистр SSP используется для обработки стека в подпрограмме прерывания, в то время как USP используется для обработки стека вне подпрограмм прерывания. Если пространство стека не разделено, то используется только SSP. При обработке стека старшие восемь битов адреса определены регистром SSB (для SSP) или USB (для USP). Регистры USP и SSP не инициализируются сбросом.

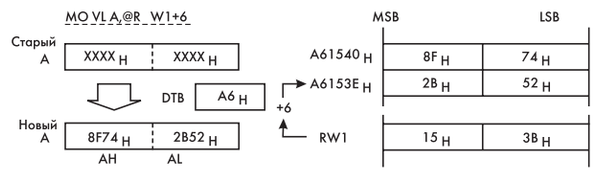

Рис. 9. Передача 32-разрядных данных

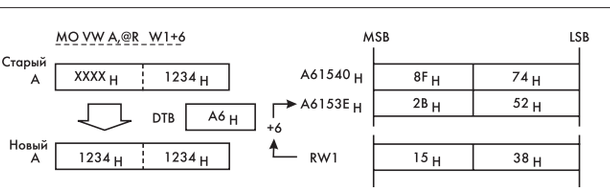

Рис. 10. Передача AL-AH

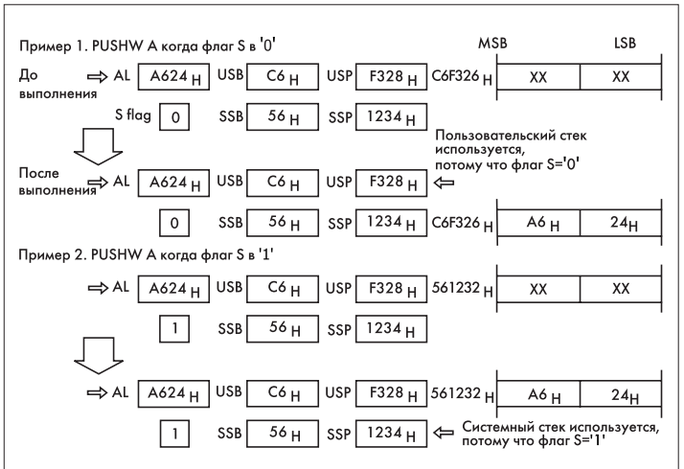

Рис. 11. Команда обработки стека и указатель стека

Регистр PS (Processor Status) состоит из трех полей (рис. 12): регистра состояния (CCR), регистра — указателя банка (RP) и регистра масок уровней прерываний (ILM).

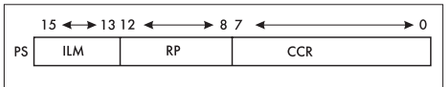

Рис. 12. Структура слова статуса процессора (PS)

Рассмотрим подробнее каждое из этих полей. Регистр CCR (рис. 13) представляет собой регистр флагов.

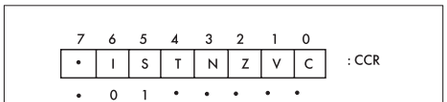

Рис. 13. Регистр флагов CCR

Назначение флагов регистра следующее.

I: флаг разрешения прерывания

Непрограммные прерывания разрешены, когда флаг I равен '1', и замаскированы, когда I равен '0'. Флаг I очищается сбросом.

S: флаг стека

Когда S = '0', USP разрешен как указатель обработки стека. Когда S = '1', SSP разрешен как указатель обработки стека. Флаг S устанавливается сбросом или при прерывании.

T: флаг «sticky» (липучка)

Флаг T устанавливается в '1', когда имеется, по крайней мере, одна '1' в данных, сдвинутых из переноса после выполнения инструкции логического или арифметического сдвига вправо. Иначе во флаге T устанавливается '0'. Кроме того, '0' устанавливается во флаге T, когда количество сдвигов нулевое.

N: флаг отрицания

Флаг N устанавливается, если наиболее значащий бит (MSB) результата операции равен '1'.

Z: флаг нуля

Флаг Z устанавливается, когда результат операции равен '0', в противном случае он сброшен.

V: флаг переполнения

Флаг V устанавливается, когда происходит переполнение в результате выполнения операции.

C: флаг переноса

Флаг C устанавливается, когда в результате выполнения инструкции происходит перенос из MSB, в противном случае сброшен.

Указатель банкового регистра (RP)

Регистр RP устанавливает зависимость между регистрами общего назначения F²MC-16LX и внутренними адресами ОЗУ. Регистр RP указывает первый адрес памяти используемого банкового регистра. Эффективный адрес можно вычислить так: [00180h+(RP)×10h]. Регистр RP состоит из пяти битов (рис. 14), и может принимать значение между 00h и 1Fh. Регистровые банки могут быть распределены в памяти по адресам 000180h — 00037h.

Однако даже в пределах этого диапазона регистровые банки не могут использоваться как регистры общего назначения, если банки не находятся во внутреннем ОЗУ. Регистр RP инициализируется в нулевое состояние сбросом. При записи байта в регистр RP только пять младших битов данных будут использованы.

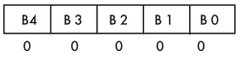

Рис. 14. Указатель банкового регистра (RP)

Регистр масок уровней прерываний (ILM)

Регистр ILM состоит из трех битов, указывая уровень маскируемого прерывания ЦПУ. Запрос на прерывание принимается только тогда, когда уровень прерывания выше, чем уровень, определенный этими тремя битами. Таким образом, уровень 0 является самым высокоприоритетным уровнем прерывания, а уровень 7 — самым низкоприоритетным (табл. 2). Поэтому для прерывания, которое будет принято, значение уровня должно быть меньше, чем текущее значение ILM. Когда прерывание принято, значение его уровня устанавливается в ILM. Таким способом предотвращается принятие прерываний таких же или низших уровней. ILM инициализируется в состояние всех нулей при сбросе. Команда может передавать регистру ILM восьмиразрядное значение, но только младшие три бита данных будут использованы.

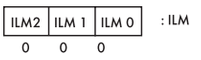

Рис. 15. Регистр масок уровней прерываний (ILM)

Таблица 2

ILM2	ILM1	ILM0	Уровень величины	Доступный уровень прерывания
0	0	0	0	Прерывание отключено
0	0	1	1	Только 0
0	1	0	2	Меньше чем 2
0	1	1	3	Меньше чем 3
1	0	0	4	Меньше чем 4
1	0	1	5	Меньше чем 5
1	1	0	6	Меньше чем 6
1	1	1	7	Меньше чем 7

Программный счетчик (PC)

Регистр PC — 16-разрядный счетчик, указывающий младшие 16 бит адреса инструкции, которая будет выполнена ЦПУ. Старшие восемь бит адреса определены регистром PCB. Регистр PC может быть изменен командой условного перехода, командой вызова подпрограммы, прерыванием или сбросом. Регистр PC также может быть использован как базовый указатель для доступа к операнду.

Прямой страничный регистр (DPR)

Регистр DPR определяет с 8-го по 15-й адресные биты в режиме прямой адресации, как показано на рис. 17. Длина регистра DPR — 8 бит, начальное значение после сброса 01H. DPR может быть прочитан или записан программно.

Префиксные коды

Префиксные коды размещаются перед командой и частично изменяют ее действие. Могут быть использованы три основных типа префиксных кодов: префикс выбора банка, префикс общего банкового регистра и префикс запрещения изменения флага.

Префикс выбора банка

Когда перед командой размещен префикс выбора банка, пространство памяти, используемое командой для доступа к данным, может быть выбрано независимо от режима адресации. Таблица 3 показывает префиксы выбора банка и соответствующие пространства памяти.

Здесь необходимо отметить, что существует группа команд, которые игнорируют уста-

Таблица 3

Префикс выбора банка	Выбранное пространство
PCB	Пространство PC
DTB	Пространство данных
ADB	Пространство AD
SPB	Пространство SSP или USP используется согласно значению флага стека

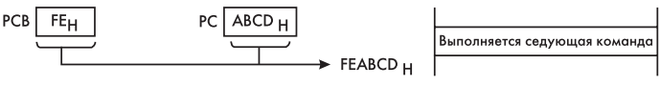

Рис. 16. Вычисление адреса инструкции

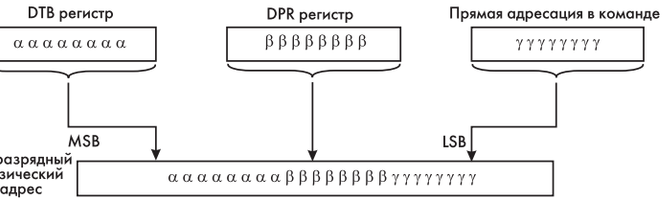

Рис. 17. Физический адрес в режиме прямой адресации

Рис. 18. Команда отключения прерываний

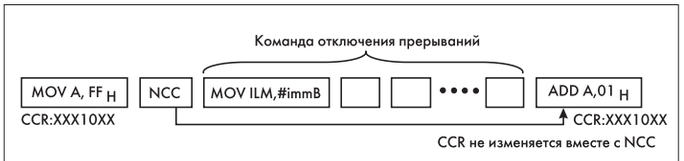

Рис. 19. Команды отключения прерываний и префиксные коды

новленные перед ними префиксы. При работе с такими командами необходимо соблюдать осторожность. К таким командам относятся следующие:

Команды, работающие со строками: MOVS, MOVSW, SCEQ, SCWEQ, FILS, FILSW.

Банковый регистр, определенный операндом, используется независимо от префикса.

Команды, работающие со стеком: PUSHW, POPW.

SSB или USB используются согласно флагу S независимо от префикса.

Команда POPW PS.

SSB или USB используются совместно с флагом S независимо от префикса. Префикс действует на следующую команду.

Команды доступа к I/O: MOV A, io / MOV io, A /MOVX A, io / MOVW A, io /MOVW io, A / MOV io, #imm8/ MOV io, #imm16 / MOVB A, io:bp / MOB io:bp, A /SETB io:bp / CLRB io:bp/ BBC io:bp, rel / BBS io:bp, rel WBTC, WBTS .

Пространство ввода-вывода банка используется независимо от префикса.

Команды изменения флага (AND CCR, #imm8, OR CCR,#imm8, MOV ILM,#imm8)

Команды выполняются нормально, но префикс действует на следующую команду.

Команда RETI:

SSB используется независимо от префикса.

Префикс общего банкового регистра (CMR)

Для упрощения обмена данными между многими задачами может понадобиться, чтобы доступ к одному и тому же банковому регистру был достаточно простым независимо от значения RP. Когда префикс CMR размещен перед командой, которая обращается к банковому регистру, эта команда обращается к общему банку (как будто RP=0), который расположен по адресам 000180h — 0018Fh, независимо от текущего значения RP.

При этом рекомендуется осторожно использовать следующие команды:

Команды, работающие со строками (MOVS, MOVSW, SCEQ, SCWEQ, FILS, FILSW).

В случае прихода запроса прерывания во время выполнения такой команды, работающей со строкой и имеющей префиксный код, после отработки обработчика прерывания префиксный код становится неправильным, что влечет неверное выполнение команды. Поэтому не рекомендуется ставить префикс перед такой командой.

Команды изменения флага (AND CCR, #imm8, OR CCR,#imm8, POPW PS, MOV ILM,#imm8). Команды выполняются нормально, но префикс действует на следующую команду.

Префикс отключения изменения флагов (NCC)

Для отключения изменения флагов можно использовать префиксный код отключения изменения флагов NCC. Размещение NCC перед командой отключает изменения флагов, связанные с этой командой.

Рекомендуется осторожно использовать следующие команды.

Команды, работающие со строками (MOVS, MOVSW, SCEQ, SCWEQ, FILS, FILSW):

Не ставьте префикс перед командой, работающей со строкой, использующей NCC. Причины те же, что и в предыдущем случае.

Команды изменения флага (AND CCR, #imm8, OR CCR,#imm8, POPW PS):

Команды выполняются нормально, но префикс действует на следующую команду.

Команды прерываний (INT vct8, INT9, INT addr16, INTP addr24, RETI):

CCR изменяется согласно описанию команды независимо от префикса.

Команда JCTX @A:

CCR изменяется согласно описанию команды независимо от префикса.

Команда MOV ILM,#imm8

Команда выполняется нормально, но префикс действует на следующую команду.

Команды блокирования прерываний

Запросы прерываний не принимаются в следующих командах:

```
-MOV ILM,#imm8 -PCB
-SPB
-OR CCR,#imm8
-NCC
-AND CCR,#imm8
-ADB
-CMR
-POPW PS
-DTB
```

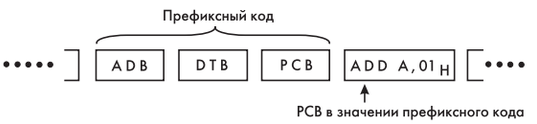

Рис. 20. Последовательные префиксные коды

Если прерывание приходит во время выполнения любой из этих команд, то прерывание может быть обработано только, когда выполняется какая-нибудь другая команда. Этот процесс показан на рис. 18.

Ограничения на команды отключения прерываний и префиксные коды

Когда префиксный код размещается перед командой отключения прерываний, то он действует на первую команду, следующую после команды отключения прерываний (рис. 19).

Последовательные префиксные коды

Когда несколько префиксных кодов размещено последовательно, действует последний (рис. 20).

Прерывания

Прерывание временно откладывает выполнение текущей программы и передает управление написанной пользователем подпрограмме, которая и обрабатывает событие, вызвавшее прерывание. Микроконтроллеры семейства F²MC-16LX выполняют четыре вида прерываний.

- Аппаратное прерывание

В ответ на запрос прерывания, сформированный периферийным ресурсом, передает управление заданной пользователем подпрограмме обработки прерывания.

- Программное прерывание

В ответ на запрос прерывания, сформированный ориентированной на программное прерывание командой, такой, как команда **INT**, передает управление заданной пользователем подпрограмме обработки прерывания.

- Прерывание по расширенному интеллектуальному сервису I/O (EI²OS)

Сервис EI²OS обеспечивает автоматическую пересылку данных между ресурсами и памятью, выполняя пересылки в стиле прямого обращения к памяти (DMA). При выполнении определенного количества пересылок, сервис EI²OS автоматически выполняет обработку программы прерывания. Сервис EI²OS представляет собой одно из аппаратных прерываний.

- Обработка исключений

В основном то же, что и прерывание. Выполняется путем временного прерывания выполнения текущей программы при обнаружении в последовательности команд неопределенной команды (отсутствующей в карте команд). Обработка исключения эквивалент-

на выполнению команды программного прерывания **INT10**.

Микроконтроллеры семейства F²MC-16LX способны выполнять прерывания от 256 источников запросов прерываний, которым соответствуют 256 векторов прерываний (от INT0 до INT256), размещенных по старшим адресам пространства памяти (с FFFC00h по FFFFFFh).

Программное прерывание может использовать все прерывания (от INT0 до INT256), но ряд векторов прерываний используется аппаратными прерываниями, прерываниями сервиса EI²OS и прерыванием обработки исключений. За каждым периферийным ресурсом закрепляется свой вектор прерывания и свой регистр управления прерыванием (ICR).

В таблице 4 показаны номера прерываний и размещение векторов прерываний.

В таблице 5 показана связь между источниками прерываний, за исключением программного прерывания, векторами прерываний и регистрами управления прерываниями. В качестве примера приведена таблица связей микроконтроллера MB90F428. Таблицы микроконтроллеров других семейств, имеющие другой набор модулей периферийных ресурсов, будут иметь другое наполнение.

Если в регистре ICR объединены два ресурса, то только один может использовать EI²OS. Если в регистре ICR объединены два ресурса и один из них определяет EI²OS, то остающийся ресурс не может использовать прерывание.

Аппаратное прерывание

Аппаратное прерывание отвечает за прием сигнала запроса прерывания от периферийного ресурса, приостановку выполнения процессором текущей программы и передачу управления заданной пользователем подпрограмме обработки прерывания. Сервис EI²OS и внешнее прерывание также обрабатываются по типу аппаратного прерывания.

Задачи аппаратного прерывания. Аппаратное прерывание аппаратно сравнивает уровень приоритета прерывания запроса прерывания, формируемого ресурсом, с регистром маски уровня приоритета прерывания (ILM) и проверяет состояние флага I в статусе процессора (PS), решая, принимать прерывание или нет. С принятием аппаратного прерывания содержимое регистров CPU автоматически сохраняется в системном стеке, уровень приоритета текущего прерывания сохраняется в регистре маски уровня приоритета (ILM), и затем выполняется переход по соответствующему вектору прерывания.

Множественное прерывание. Может быть запущено и множественное аппаратное прерывание — прерывание при одновременном поступлении нескольких запросов прерывания.

Сервис EI²OS. Сервис EI²OS выполняет автоматические пересылки между памятью и I/O и запускает аппаратное прерывание при завершении пересылки. Сервис EI²OS не может быть запущен во множественном режиме. Во время обработки какого-либо EI²OS все другие запросы и аппаратных прерываний и сервиса EI²OS откладываются.

Таблица 4

Команда программного прерывания	Младший адрес вектора (L)	Средний адрес вектора (M)	Старший адрес вектора (H)	Режимный регистр	Номер прерывания	Аппаратное прерывание
INT0	FFFFFCh	FFFFFDh	FFFFFEh	Не используется	#0	Нет
:	:	:	:	:	:	:
INT7	FFFFE0h	FFFFE1h	FFFFE2h	Не используется	#7	Нет
INT8	FFFFDCh	FFFFDDh	FFFFDEh	FFFFDFh	#8	Вектор RESET
INT9	FFFFD8h	FFFFD9h	FFFFDAh	Не используется	#9	Нет
INT10	FFFFD4h	FFFFD5h	FFFFD6h	Не используется	#10	Обработка исключений
INT11	FFFFD0h	FFFFD1h	FFFFD2h	Не используется	#11	Аппаратное прерывание #0
INT12	FFFFCCh	FFFFCDh	FFFFCEh	Не используется	#12	Аппаратное прерывание #1
INT13	FFFFC8h	FFFFC9h	FFFFCAh	Не используется	#13	Аппаратное прерывание #2
INT14	FFFFC4h	FFFFC5h	FFFFC6h	Не используется	#14	Аппаратное прерывание #3
:	:	:	:	:	:	:
INT254	FFFC04h	FFFC05h	FFFC06h	Не используется	#254	Нет
INT255	FFFC00h	FFFC01h	FFFC02h	Не используется	#255	Нет

Примечание:

1) Аппаратные прерывания #NN закрепляются за конкретными ресурсами — источниками запросов прерываний

2) Не использованные векторы прерываний должны быть закреплены за адресами обработки исключений

Таблица 5

Источник прерывания	Взаимосвязь с EI²OS	Вектор прерывания			Регистр управления прерыванием		Приоритет
		Номер		Адрес	ICR	Адрес	
Сброс (RESET)	x	#08	08h	FFFFDCh	–	–	Высший
Команда **INT9**	x	#09	09h	FFFFD8h	–	–	
Обработка исключений	x	#10	0Ah	FFFFD4h	–	–	
CAN0 RX	x	#11	0Bh	FFFFD0h	ICR00	0000B0h*	
CAN0 TX/NS	x	#12	0Ch	FFFFCCh			
CAN1 RX	x	#13	0Dh	FFFFC8h	ICR01	0000B1h*	
CAN1 TX/NS	x	#14	0Eh	FFFFC4h			
Захват входа 0	xx	#15	0Fh	FFFFC0h	ICR02	0000B2h*	
DTP/внешнее прерывание; при обнаружении на канале 0	xx	#16	10h	FFFFBCh			
Перезагрузка таймера 0	xx	#17	11h	FFFFB8h	ICR03	0000B3h*	
DTP/внешнее прерывание; при обнаружении на канале 1	xx	#18	12h	FFFFB4h			
Захват входа 1	xx	#19	13h	FFFFB0h	ICR04	0000B4h*	
DTP/внешнее прерывание; при обнаружении на канале 2	xx	#20	14h	FFFFACh			
Захват входа 2	xx	#21	15h	FFFFA8h	ICR05	0000B5h*	
DTP/внешнее прерывание; при обнаружении на канале 3	xx	#22	16h	FFFFA4h			
Захват входа 3	xx	#23	17h	FFFFA0h	ICR06	0000B6h*	
DTP/внешнее прерывание; при обнаружении на каналах 4/5	xx	#24	18h	FFFF9Ch			
PPG timer 0	xx	#25	19h	FFFF98h	ICR07	0000B7h*	
DTP/внешнее прерывание; при обнаружении на каналах 6/7	xx	#26	1Ah	FFFF94h			
PPG таймер 1	xx	#27	1Bh	FFFF90h	ICR08	0000B8h*	
Перезагрузка 1	xx	#28	1Ch	FFFF8Ch			
PPG таймер 2	xxx	#29	1Dh	FFFF88h	ICR09	0000B9h*	
Таймер-счетчик интервалов (основной тактовый сигнал)	x	#30	1Eh	FFFF84h			
Переполнение автономного таймера	x	#31	1Fh	FFFF80h	ICR10	0000BAh*	
Конец преобразования A/D преобразователем	xxx	#32	20h	FFFF7Ch			
Очистка автономного таймера	x	#33	21h	FFFF78h	ICR11	0000BBh*	
Генератор аудиосигнала	x	#34	22h	FFFF74h			
Таймер временной базы	x	#35	23h	FFFF70h	ICR12	0000BCh*	
Таймер интервалов (сигнал субтактирования)	x	#36	24h	FFFF6Ch			
Прием UART1	xxxx	#37	25h	FFFF68h	ICR13	0000BDh*	
Передача UART1	xx	#38	26h	FFFF64h			
Прием UART0	xxxx	#39	27h	FFFF60h	ICR14	0000BEh*	
Передача UART0	xx	#40	28h	FFFF5Ch			
Состояние Flash-памяти	x	#41	29h	FFFF58h	ICR15	0000BFh*	
Модуль генерации задержанного прерывания	x	#42	2Ah	FFFF54h			Низший

xxx — источник прерывания может быть использован

x — источник прерывания не может быть использован

xxxx — источник прерывания может быть использован и располагает функцией останова сервиса EI²OS

xx — источник прерывания может быть использован когда не используется источник прерывания связанный с регистром ICR

* – Уровень приоритета ресурсов, общих для регистра ICR, одинаков.

Если в регистре ICR объединены два ресурса, то только один может использовать EI²OS

Таблица 6. Средства аппаратного прерывания

Компоненты системы прерываний	Средства, связанные с аппаратными прерываниями	Функции
Ресурс	Бит разрешения прерывания, бит запроса прерывания	Управление запросом прерывания от ресурса
Контроллер прерываний	Регистр управления прерываниями (ICR)	Устанавливает уровень приоритета и управляет сервисом EI²OS
CPU	Флаг разрешения прерывания (I)	Разрешает/запрещает прерывания
	Регистр масок уровня приоритета (ILM)	Сравнивает уровень приоритета запрашиваемого прерывания с уровнем приоритета текущего прерывания
	Микрокод	Исполняет подпрограмму обработки прерывания
Ячейки памяти с адресами от FFFC00h по FFFFFFh	Таблица векторов прерываний	Хранит адреса перехода для обработки прерывания

Внешнее прерывание. Внешнее прерывание, включая прерывание по активации из режима Standby, воспринимается как аппаратное прерывание встроенным периферийным ресурсом (детектором запроса прерывания).

Средства аппаратного прерывания. Связанный с аппаратными прерываниями механизм содержит четыре блока средств, показанных в таблице 6. При использовании аппаратного прерывания эти четыре блока должны быть установлены программными средствами.

Для более подробного знакомства с работой ядра микроконтроллеров семейства Fujitsu F²MC-16LX рекомендуем изучить руководство «F²MC-16LX 16-BIT MICROCONTROLLER MB90500 Series. PROGRAMMING MANUAL», которое можно найти на диске, распространяемом фирмой «КТЦ-МК».

Литература

1. КрыловЕ.И. Гаврилюк С.В. 16-разрядные Flash-микроконтроллеры семейства F2MC-16LX фирмы Fujitsu // Компоненты и технологии. 2001. №5. С.38–42.
2. КрыловЕ.И. Гаврилюк С.В. Интегрированная среда разработки программ микроконтроллеров фирмы Fujitsu // Компоненты и технологии. 2001. №7. С.104–107.