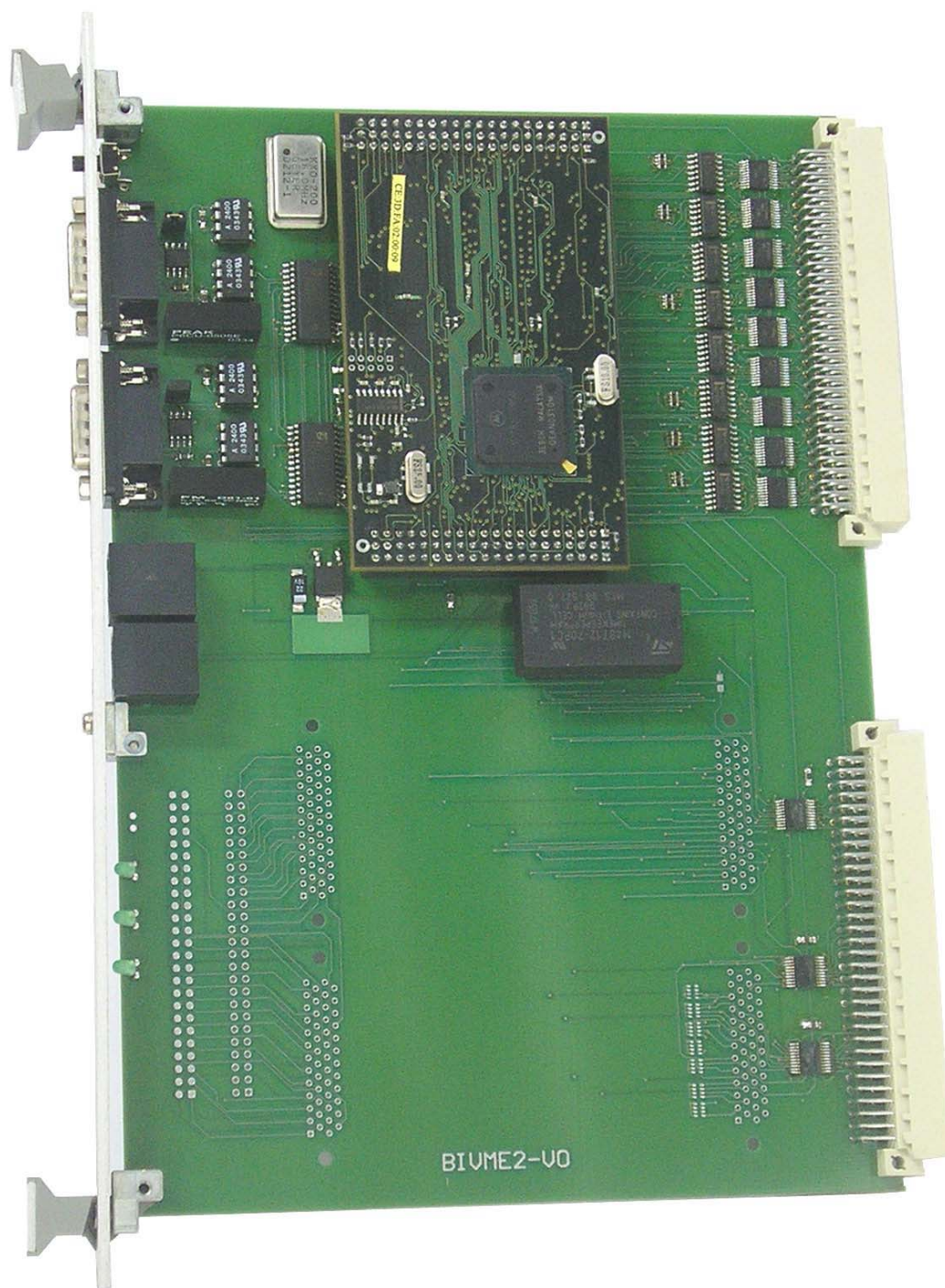


VME – контроллер BIVME-2

Техническое описание и инструкция по эксплуатации



Новосибирск, 2005 г.

Оглавление

1. НАЗНАЧЕНИЕ	4
2. ОСНОВНЫЕ ХАРАКТЕРИСТИКИ	4
3. ПОДКЛЮЧЕНИЕ КОНТРОЛЛЕРА	6
4. ЗАГРУЗОЧНЫЙ МОНИТОР	9
4.1. КОНСОЛЬ МОНИТОРА	9
4.2. РЕДАКТИРОВАНИЕ ПАРАМЕТРОВ МОНИТОРА	9
4.3. СЕРВИСНЫЕ КОМАНДЫ	10
4.4. ОСОБЕННОСТИ ЗАГРУЗКИ ОС LINUX	11
4.5. ОБНОВЛЕНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ КОНТРОЛЛЕРА	11
5. КАРТА ПАМЯТИ.....	13
6.VME ИНТЕРФЕЙС.....	14
6.1. ОБЩЕЕ ОПИСАНИЕ.....	14
6.2. УПРАВЛЯЮЩИЕ РЕГИСТРЫ VME ИНТЕРФЕЙСА.....	14
6.3. АДРЕСНЫЙ ДЕКОДЕР.....	18
6.4. VME – ЗАПРОСЧИК.....	18
6.5. VME – АРБИТР.....	19
6.6. ОБРАБОТЧИК ПРЕРЫВАНИЙ	19
7. ПРОГРАММНАЯ ПОДДЕРЖКА VME ИНТЕРФЕЙСА.....	20
7.1. ВВЕДЕНИЕ.....	20
7.2. БИБЛИОТЕКА LIBVMEDIRECT	21
7.2.1. <i>Функции инициализации/закрытия</i>	21
7.2.2. <i>Установка параметров VME интерфейса</i>	21
7.2.3. <i>Функции ввода-вывода</i>	22
7.3. МОДУЛЬ VMEIO.....	23
7. RTC ТАЙМЕР И БАТАРЕЙНАЯ ПАМЯТЬ	24
8. CAN ИНТЕРФЕЙСЫ	25

Список рисунков

Рисунок 1. Структурная схема контроллера.....	5
Рисунок 2. Расположение разъемов и индикации	8
Рисунок 3. Структура программного обеспечения VME интерфейса.....	20

Список таблиц

Таблица 1. Технические характеристики контроллера BIVME2	4
Таблица 2. Назначение контактов разъема Ethernet X3	7
Таблица 3. Назначение контактов консольного порта X10.....	7
Таблица 4. Назначение контактов разъемов CAN (вилка DB9).....	7
Таблица 5. Карта памяти периферийных устройств	13
Таблица 6. Адреса регистров VME.....	14
Таблица 7. Наименования битов регистра Reg0.....	15
Таблица 8. Наименования битов регистра Reg1	15
Таблица 9. Наименования битов регистра Reg2.....	15
Таблица 10. Назначение битов регистра Reg0.....	16
Таблица 11. Назначение битов регистра Reg1	17
Таблица 12. Назначение битов регистра Reg2.....	17
Таблица 13. Параметры функции libvmecl и их описание.....	21

1. Назначение

Контроллер BIVME-2 разработан для эксплуатации в составе магистрально-модульных систем стандарта VME. Контроллер выполняет функции системного контроллера и VME-мастера. Наличие встроенных интерфейсов Ethernet и CAN позволяет легко интегрировать VME систему в состав современного компьютерного комплекса. Программная поддержка включает открытые операционные системы RTEMS и Linux.

Возможные области применений:

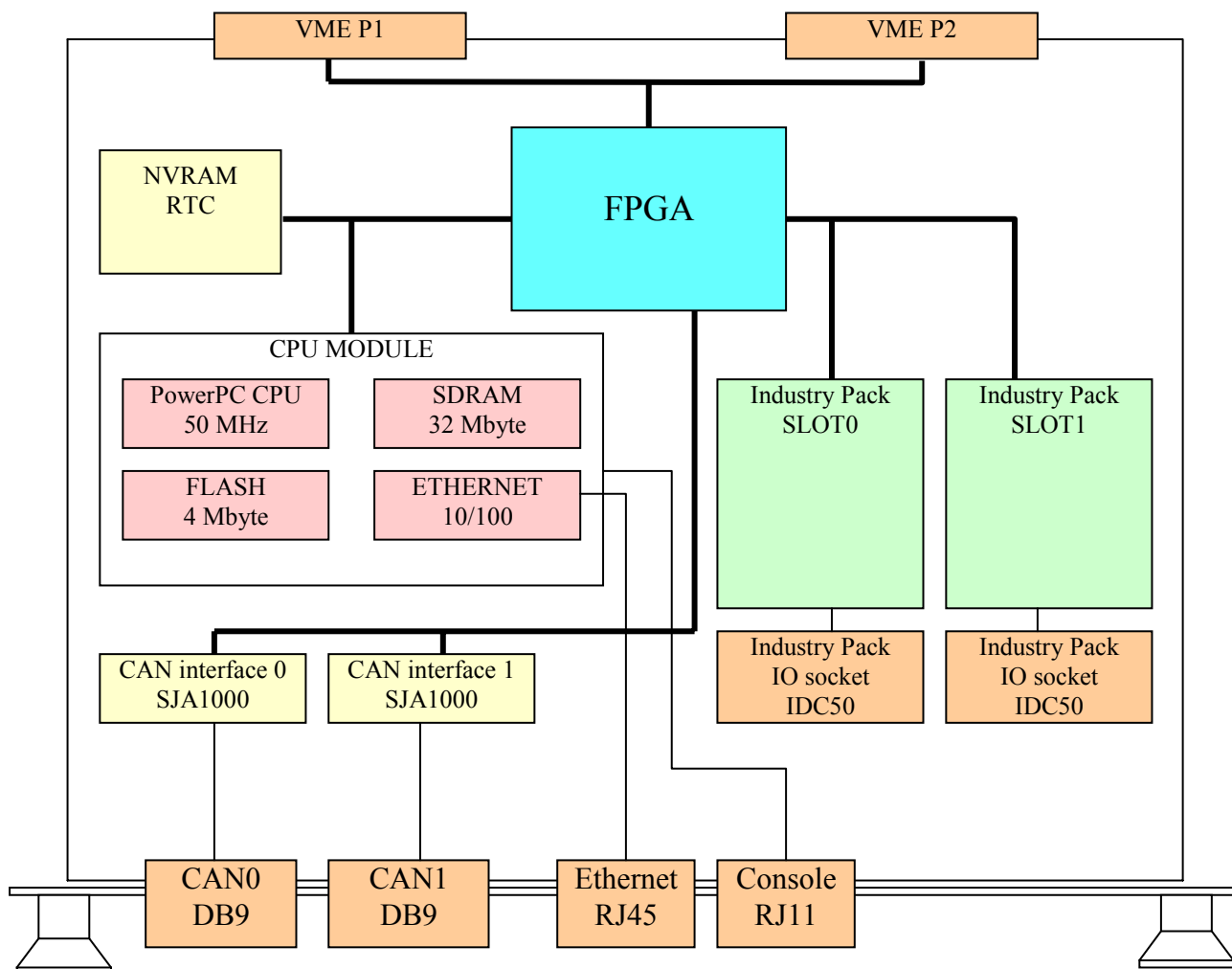
- Управление промышленными установками
- Системы сбора данных
- Контроллер ввода-вывода

2. Основные характеристики

Таблица 1. Технические характеристики контроллера BIVME2

Процессор	Freescape PowerPC MPC852
Тактовая частота	50 Mhz
ОЗУ	32 Мбайт SDRAM
Flash	4 Мбайт
Интерфейсы	Ethernet 10/100
	Консоль RS232
	2 интерфейса CAN 2.0b
	2 слота Industry Pack
	VME
Прочее	Микросхема RTC
VME – шина	
Типы адресации	A16, A24, A32 Программируемый адресный модификатор
Форматы данных	D8, D16, D32
VME запросчик	Программируемая линия запроса, освобождение по запросу
VME арбитр	Фиксированный и круговой приоритеты
Прерывания	IRQ1..IRQ7 на VME шине Вызов прерывания по AC_FAIL

Рисунок 1. Структурная схема контроллера



3. Подключение контроллера

На плате контроллера расположены (см. Рисунок 2):

1. **Кнопка сброса (SW1).** При нажатии кнопки происходит аппаратный сброс контроллера. Если установлен режим системного контроллера, сигнал сброса подается на всю VME систему. Если BIVME2 не является системным, сбрасывается только локальный процессор.
2. **Индикатор локальной сети (HL1).** Индицирует наличие сигнала линка, при этом светодиод светится. Прием и передача пакетов контроллером отображается путем кратковременного выключения светодиода.
3. **Индикатор активности VME шины (HL2).** Загорается при обращении процессора к VME шине.
4. **Индикаторы активности шины CAN (HL3, HL4).** Индикаторы загораются при приеме или передаче пакетов через CAN линию. HL3 соответствует линии CAN0, HL4 соответствует линии CAN1.
5. **Разъемы шины CAN (X5, X6).** X6 соответствует CAN0, X5 соответствует CAN1.
6. **Разъем Ethernet 10/100 для подключения локальной сети (X3).** Через этот разъем контроллер подключается стандартным кабелем к сетевому концентратору.
7. **Разъем для подключения консоли (X10).** Через кабель-переходник подключается к порту RS232 ПК или терминала.
8. **Штыревые разъемы под плоский кабель для вывода сигналов IP мезонинов (X8, X9).** Каждый разъем имеет 50 контактов и подсоединен одноименными контактами к разъему ввода-вывода мезонина. X8 соответствует IP мезонину 0, X9 соответствует IP мезонину 1.
9. **Разъемы VME шины P1, P2.** Разъем P2 используется при 32-х разрядном доступе к данным и 32-х разрядном адресе. На некоторых кросс-платах VME этот режим может не поддерживаться, и разъем P2 может быть не задействован.

Таблица 2. Назначение контактов разъема Ethernet X3

Контакт	Цепь
1	TX+
2	TX-
3	RX+
4	
5	
6	RX-
7	
8	

Таблица 3. Назначение контактов консольного порта X10

Контакт	Цепь	направление
1	RXD	ВХОД
2	TXD	ВЫХОД
3	GND	
4	GND	
5		
6		

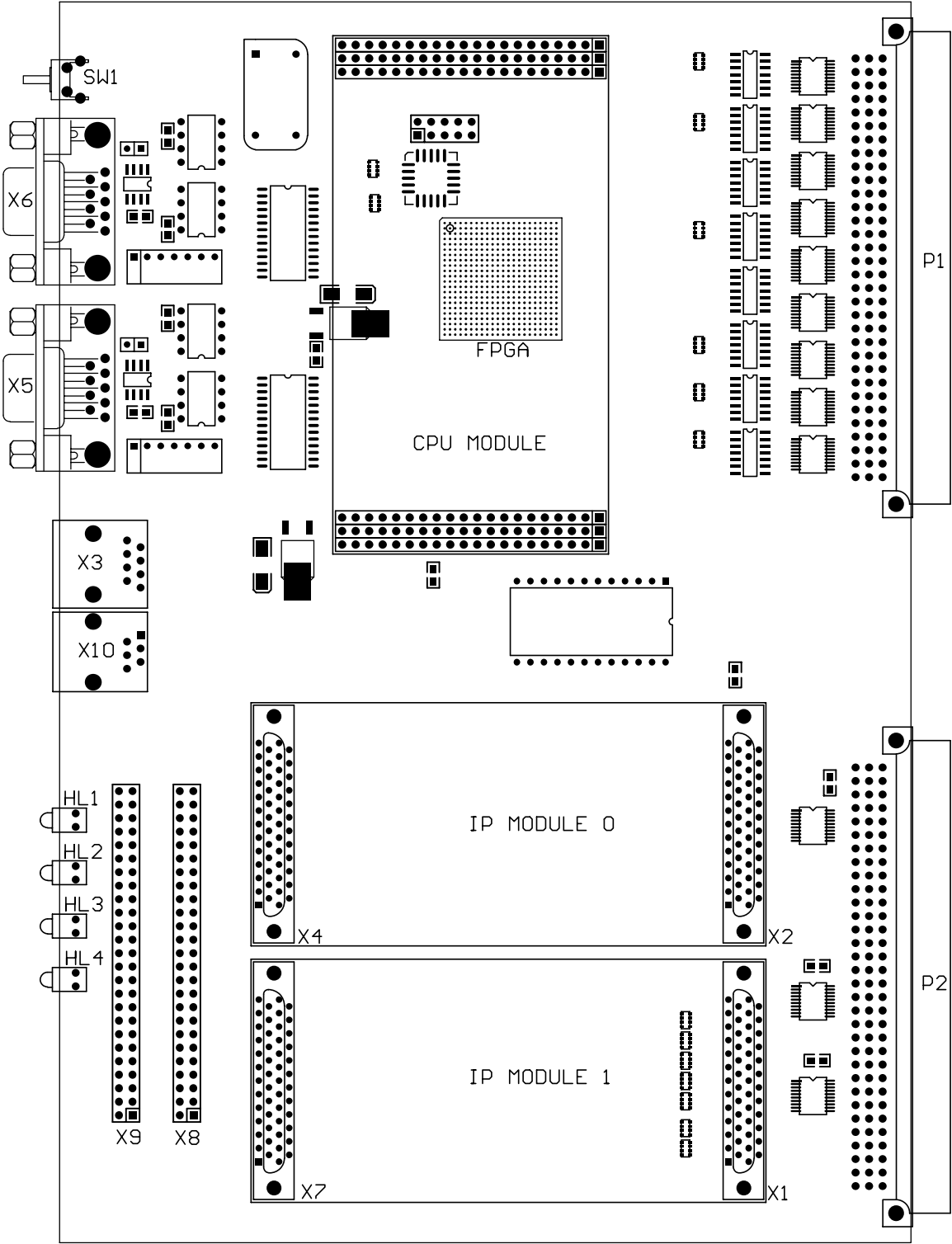
Таблица 4. Назначение контактов разъемов CAN (вилка DB9)

контакт	сигнал
1	Согласующий резистор 1
2	CanL
3	Оплетка кабеля CAN
4	Согласующий резистор 2
5	Земля устройства
6	Выход питания 0 в
7	CanH
8	NC
9	Выход питания 5 в

Примечания:

- i. Внутренний согласующий резистор (120 ом) может быть включен джампером, установленным на плате контроллера рядом с разъемом CAN, или путем замыкания выводов 1,4.
- ii. Выводы 6, 9 могут быть использованы как изолированное питание внешнего устройства с потреблением не более 100 ма.

Рисунок 2. Расположение разъемов и индикации



4. Загрузочный монитор

После включения питания или перезапуска контроллера управление всегда получает загрузочный монитор. В зависимости от установленных параметров, монитор сразу после запуска может начать загрузку операционной системы или перейти в командный режим. Командный режим используется для конфигурации монитора и выполнения определенных сервисных функций. В этом режиме монитор выполняет вводимые пользователем через консоль команды и выводит на консоль результат их выполнения. Набор команд зависит от текущего меню. Содержимое меню можно посмотреть нажатием в терминальной программе клавиш <h>, <Enter>.

Монитор выполняет следующие функции:

- Загрузка и старт операционной системы
- Программирование flash памяти
- Выполнение сервисных функций по команде пользователя (ARP запрос, ICMP echo)
- Редактирование параметров и настроек сети, относящихся к работе монитора
- Копирование, заполнение, тестирование памяти

Параметры монитора сохраняются во flash памяти и не зависят от параметров операционной системы.

4.1. Консоль монитора

Для подключения консоли необходимо использовать специальный кабель и терминальную программу с параметрами: 38400, 8b, np, flow control = off

4.2. Редактирование параметров монитора

Переход в меню редактирования параметров происходит по команде **opt** монитора. Доступны следующие параметры:

myip - ввод IP адреса монитора

servip - ввод IP адреса сервера, с которого будет загружаться файл по TFTP протоколу

gwip - ввод IP адреса сетевого шлюза

mask - ввод сетевой маски

file - имя файла, загружаемого из TFTP сервера

loadptr - адрес области памяти, куда будет загружаться файл (hex)

jump ptr - адрес передачи управления по команде go или при автозапуске (hex)

bootstr - содержимое командной строки, не имеет значения для монитора, используется только операционной системой

bootptr - адрес, куда копируется командная строка, если установлен флаг «Copy bootstring to RAM»

bootsrc – источник загрузки файла (0 – TFTP сервер, 1 – флэш).

bootlen – длина файла. Параметр используется только при загрузке из флэш памяти.

list - выводит значения всех параметров

flags - переход в меню установки флагов

Доступны следующие флаги:

- "verbose mode" - при установке в "on" монитор выводит больше текстовых сообщений при работе;
- "TFTP standalone" – должен быть установлен в "on".
- "Auto bootp" - выполнять BOOTP запрос при перезапуске
- "Auto load" - выполнять загрузку файла по протоколу TFTP при перезапуске (аналогично команде load)
- "Auto jump" - выполнять передачу управления при перезапуске (аналогично команде go)
- "Copy bootstring to RAM" - копировать командную строку в оперативную память при перезапуске
- "Watchdog timer" – включить сторожевой таймер (автоматический перезапуск при зависании контроллера)

update - сохранить все параметры во flash

4.3. Сервисные команды

pings - монитор переходит в режим ICMP эхо сервера и отвечает на ping запросы от других станций в сети. Режим служит для проверки соединения локальной сети. Перед выполнением команды должны быть установлены сетевые настройки (IP адрес, маска, шлюз)

bootp - монитор запрашивает сетевые параметры у BOOTP сервера

arp - монитор делает ARP запрос и получает Ethernet адрес сервера по его IP адресу. Предварительно должен быть введен IP адрес сервера (параметр servip).

load - монитор загружает файл в оперативную память контроллера по TFTP потоку. Адрес памяти задается параметром loadptr, сетевой адрес - параметром servip, имя файла – параметром file. Используется для тестовых целей.

go - монитор передает управление по адресу

mdump - распечатка содержимого памяти

mfill - заполнение области памяти кодом

mtest - тестирование области памяти

fflash – программирование flash памяти. Команда аналогична команде load, только данные загружаются не в оперативную память, а программируются во flash. Длина запрограммированного участка во flash памяти определяется длиной файла.

4.4. Особенности загрузки ОС Linux

Для загрузки адаптированного для PowerPC ядра ОС Linux необходимо учитывать, что в память контроллера загружается не исполняемый файл, а пакет, подготовленный программой `lpack`. В теле пакета в сжатом виде хранится ядро и образ файловой системы Linux. Пакет также содержит заголовок с указанием размеров ядра, образа файловой системы и их смещениями. Образ файловой системы может быть опциональным, если используется загрузка через NFS. Параметры монитора для загрузки Linux должны быть следующими:

`jump_ptr = 200000`

`load_ptr = 200000`

`boot_ptr` – не имеет значения

Образ с пакетом Linux загружается из флэш памяти или с удаленного TFTP сервера, в зависимости от параметра `bootsrc`.

После загрузки пакета Linux монитор передаст управление ядру, которое должно знать, где находится корневая файловая система Linux. Возможны два варианта:

1. Образ файловой системы может находиться в том же пакете, что и ядро. В этом случае ядру через параметр монитора `bootstr` передается командная строка следующего вида:

`root=/dev/ram rw ramdisk_size=6000 init=/usr/sbin/flash`

2. Ядро Linux использует сетевую файловую систему NFS. Командная строка должна иметь вид (пример):

`root=/dev/nfs rw nfsroot=192.168.1.1:/home/sdk/ppc860/linux/initrd
ip=192.168.1.2:192.168.1.1::255.255.255.0::eth0:off`

где

192.168.1.1 – адрес NFS сервера

192.168.1.2 – адрес контроллера

/home/sdk/ppc860/linux/initrd – директория NFS сервера, в которой располагается корневая файловая система Linux.

4.5. Обновление программного обеспечения контроллера

Если flash память контроллера используется для хранения пакета ОС Linux, обновление программного обеспечения происходит по следующей процедуре:

- Запустить программу TFTP сервера на компьютере и разрешить серверу доступ на чтение к какой-либо директории
- Скопировать в эту директорию файл `image.bin` с образом программного обеспечения контроллера
- Подключить к контроллеру кабель консоли и кабель локальной сети
- Запустить на компьютере программу терминала и установить параметры последовательного порта: скорость 34800, 8 бит, 1 стоп бит, без контроля четности.
- Перезапустить контроллер (команда **reboot**).

- Остановить процесс загрузки, нажав любую клавишу. После этого контроллер будет находиться в режиме монитора.
- Если необходимо, изменить IP адрес монитора и сервера, а также сетевую маску (меню **opt** монитора).
- Программирование происходит по команде fflash монитора:
boot> fflash<CR>

5. Карта памяти

Большинство ресурсов контроллера отображается в 32-битное физическое адресное пространство. Обращение к устройствам возможно либо через драйверы операционной системы (в контексте ядра), либо путем отображения окон в прикладной программе.

Таблица 5. Карта памяти периферийных устройств

Адреса	Размер	Устройство
00000000..01FFFFFF	32 Мбайт	SDRAM (*)
80000000..83FFFFFF	64 Мбайт	VME шина, адресация A32, страница (**)
F0000000..F0FFFFFF	16 Мбайт	VME шина, адресация A24 (**)
F8000000..F83FFFFF	4 Мбайт	Flash (*)
FFF00000..FFF0FFFF	64 Кбайт	Блок регистров процессора MPC852
FFF40000..FFF43FFF	16 Кбайт	Блок регистров VME интерфейса (**)
FFF44000..FFF400FF	256 байт	Область памяти RTC (**)
FFF48000..FFF4801F	32 байт	Блок регистров CAN интерфейса 0
FFF48020..FFF4803F	32 байт	Блок регистров CAN интерфейса 1
FFFF0000...FFFFFFFF	64 Кбайт	VME шина, адресация A16 (**)

Примечания:

* расположение области памяти программируется загрузочным монитором

** расположение области памяти программируется VME библиотекой

6. VME интерфейс

6.1. Общее описание

VME интерфейс выполняет следующие функции :

- Трансляция циклов от локальной шины процессора на VME шину в соответствии с картой адресов и форматом данных
- Арбитраж VME : запрос и освобождение шины, системный арбитр (в режиме системного контроллера)
- Обработка прерываний: передача прерывания от VME на локальный процессор в соответствии с приоритетами и маской прерывания, выдача циклов подтверждения прерывания

BIVME2 может работать как в режиме системного контроллера, так и совместно с другим системным контроллером. Режим системного контроллера отличается наличием следующих функций:

- Включен арбитр шины
- Включена задержка на дейза-цепочке подтверждения прерываний
- Контроллер является источником тактовых импульсов по линии SYSCLK
- Локальный сброс контроллера приводит к системному сбросу на VME магистрали

VME интерфейс состоит из следующих функциональных модулей:

- Адресный декодер
- VME запросчик
- VME арбитр
- Обработчик прерываний

6.2. Управляющие регистры VME интерфейса

Таблица 6. Адреса регистров VME

Адрес	Название	Тип
FFF40000	Reg0	RW
FFF40004	Reg1	RW
FFF40008	Reg2	RW
FFF40104	Inth1	RO
FFF40108	Inth2	RO

FFF4010C	Inth3	RO
FFF40110	Inth4	RO
FFF40114	Inth5	RO
FFF40118	Inth6	RO
FFF4011C	Inth7	RO

Операции со всеми регистрами должны осуществляться по 32-битному доступу. После инициализации системы значения битов в записываемых регистрах =0. Наименования битов в регистрах Reg0 .. Reg2 и их назначение приведены в таблицах ниже по тексту.

Таблица 7. Наименования битов регистра Reg0

Бит	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Название	SF	OVL5..OVL0						ART	SYS	BRL1	BRL0	PAM	AM3..AM0			

Таблица 8. Наименования битов регистра Reg1

Бит	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Название	xxx									IM7..IM1						ACM

Таблица 9. Наименования битов регистра Reg2

Бит	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Название	xxx								CI0	CI1	CER	AF	SF	IPL2..IPL0		

Таблица 10. Назначение битов регистра Reg0

Биты	Назначение
AM3..AM0	Биты адресного модификатора шины VME. Значения битов регистра соответствуют одноименным линиям на шине.
PAM	<p>Если PAM=0, линии AM5, AM4 VME шины устанавливаются в соответствии с типом адресного пространства в данном цикле шины:</p> <p>[AM5,AM4] = [0,0] для A32 [AM5,AM4] = [1,0] для A16 [AM5,AM4] = [1,1] для A24</p> <p>Если PAM=1, значения на линиях не зависят от типа адреса и вместе с полем AM3..AM0 регистра задают пользовательский адресный модификатор.</p> <p>[AM5,AM4] = [0,1]</p>
BRL1..BRL0	<p>Биты кодируют уровень запроса шины у арбитра VME.</p> <p>[BRL1, BRL0] = [0, 0] соответствуют линии запроса BR0 [BRL1, BRL0] = [1, 1] соответствуют линии запроса BR3</p>
SYS	<p>0 - режим системного контроллера выключен 1 – режим системного контроллера включен</p>
ART	<p>Тип арбитра 0 PRI (приоритетный арбитр) 1 RRS (круговой приоритет)</p>
OVL5..OVL0	Номер адресной страницы при доступе A32. Биты соответствуют адресным линиям A31..A26 на шине VME.
SF	<p>Установка линии SYS FAIL на шине VME 0 SYS FAIL не активна (высокий уровень) 1 SYS FAIL активна (низкий уровень)</p>

Таблица 11. Назначение битов регистра Reg1

Биты	Назначение
АСМ	Маска сигнала AC FAIL на шине VME 0 запретить прерывание 1 разрешить прерывание
IM1..IM7	Маска соответствующих линий прерывания VME шины 0 запретить прерывание 1 разрешить прерывание
xxx	Не используются

Таблица 12. Назначение битов регистра Reg2

Биты	Назначение
IPL2..IPL0	Текущий наиболее приоритетный уровень прерывания. Поле кодируется на основании состояния линий IRQ1..IRQ7 и доступно только на чтение.
SF	Чтение линии SYS FAIL на шине VME 0 SYS FAIL не активна (высокий уровень) 1 SYS FAIL активна (низкий уровень)
AF	Чтение линии AC FAIL на шине VME 0 AC FAIL не активна (высокий уровень) 1 AC FAIL активна (низкий уровень)
CER	Чтение: Наличие ошибки шины в последнем обращении к интерфейсу VME. 0 – нет ошибки 1 – нет ответа от VME модуля или наличие сигнала BERR Запись: 0 – игнорируется 1 – сброс ошибки
CI0, CI1	Состояние линии прерывания соответствующего CAN контроллера

Регистры Inth1..Inth7

Чтение этих регистров приводит к появлению на VME шине цикла подтверждения прерывания для соответствующего уровня. При этом биты 7..0 прочитанных данных будут содержать вектор прерывания, выставленный на шину периферийным модулем. Чтение регистров InthX происходит в обработчике прерывания.

6.3. Адресный декодер

После захвата VME шины посредством процедуры арбитража, VME контроллер может читать и писать данные на VME шину. Для этого локальный процессор должен обратиться в одно из адресных пространств на локальной шине, декодируемых VME-интерфейсом (см. карту памяти контроллера). При этом адреса на локальной шине будут однозначно соответствовать адресам на VME шине:

Локальный адрес	VME адрес	Тип адресации
FFFF0000..FFFFFFFF	0000..FFFF	A16
F0000000..F0FFFFFF	00000000..FFFFFF	A24
80000000..83FFFFFF	00000000..3FFFFFFF	A32

Для типа адресации A32 VME адрес определяется также номером страницы, см. Reg0, биты OVL5..OVL0.

Длительность цикла чтения или записи данных в периферийный модуль VME зависит от времени отклика конкретного модуля, но ограничивается 64 мкс. При длительности VME цикла более 64 мкс, контроллер выставит на шине сигнал BERR и локальный процессор получит прерывание “ошибка шины”.

6.4. VME – запросчик

VME запросчик проводит процедуры захвата и освобождения шины. Захват шины инициируется после обращения локального процессора к VME шине, если контроллер не является мастером. На время процедуры арбитража локальный процессор будет находиться в состоянии ожидания. Время ожидания ограничено и составляет 64 мкс. При превышении этого времени процессор выходит из состояния ожидания и получает прерывание “ошибка шины”. Сигнал BERR на VME шине при этом не выставляется. Если в течение 64 мкс шину захватить не удалось, запросчик продолжает удерживать запрос шины до ее получения, хотя локальный процессор вышел из состояния ожидания.

Линия запроса программируется битами Reg0[6,5]. Например, Reg0[6,5] = [1,0] соответствует запросу по линии 2. Наиболее приоритетная линия запроса шины имеет номер 3.

Стратегия освобождения шины – “освобождение по запросу”, т.е. запросчик освобождает VME шину, если обнаруживает запрос от другого контроллера. Шина освобождается также, если обнаруживается сигнал BLCK.

6.5. VME – арбитр

Функция арбитра включена, когда BIVME-1 является системным контроллером, т.е. установлен в слот 0 и бит Reg0.SYS = 1. В этом случае арбитр начинает предоставлять VME шину в соответствии со стратегией предоставления шины, установленной битом Reg0.ART.

1. Reg0.ART = 0. PRI арбитр. В случае обнаружения запроса одновременно по нескольким линиям, шина предоставляется контроллеру, который выставил запрос по наиболее приоритетной линии. Наивысший приоритет имеет линия 3, самый низкий приоритет у линии 0.
2. Reg0.ART = 1. RRS арбитр. Шина предоставляется случайным образом любому контроллеру, который выставил запрос по линии, отличной от уровня запроса текущего мастера. Опрос линий при этом арбитр производит по круговой схеме.

В любом случае, если арбитру требуется передать шину другому контроллеру, он выставляет сигнал BCLR и ожидает освобождение шины текущим мастером. Ожидание освобождения не ограничено по времени.

6.6. Обработчик прерываний

Для обработки прерывания от VME шины программное обеспечение должно установить бит маски прерывания в регистре Reg1. Из всех разрешенных и активных прерываний VME интерфейсом выбирается наиболее приоритетное и устанавливается запрос прерывания локальному процессору. Если разрешена обработка данного уровня прерывания, контроллер переходит в подпрограмму обработки, где проводится цикл подтверждения прерывания путем чтения регистра inthX. Наибольший приоритет имеет прерывание по сигналу AC_FAIL, наименьший – прерывание по линии IRQ1. Внутри контроллера VME интерфейс подключен к линии IRQ1 процессора.

Более подробно, последовательность событий выглядит так:

1. Установка нужной маски прерывания;
2. Вызов подпрограммы обработки прерывания (при получении соответствующего запроса по VME шине);

В подпрограмме обработки:

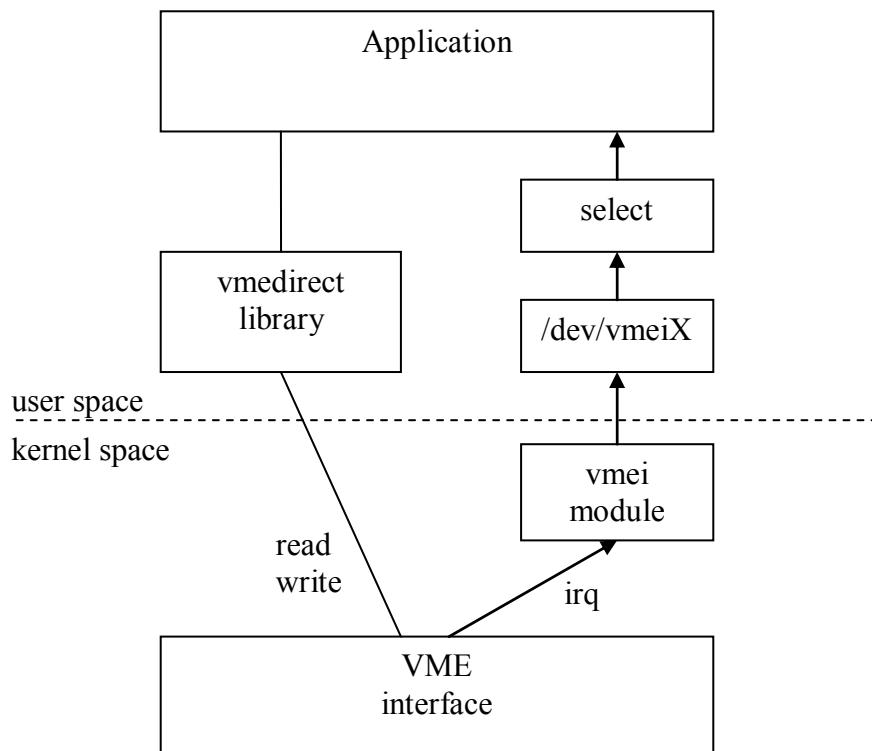
3. Читается уровень вызвавшего прерывания int_level в регистра Reg2[2..0].
4. Если int_level = 0, программа выполняет аварийные действия, связанные со сбоем питания, иначе переходит к следующему пункту.
5. Читается регистр inthX, где X=int_level. Это приводит к появлению на VME шине цикла подтверждения прерывания для уровня int_level. После проведения цикла подтверждения прерывания периферийный модуль сбрасывает свой запрос. Прочитанные данные будут содержать вектор прерывания, выставленный периферийным модулем.

7. Программная поддержка VME интерфейса

7.1. Введение

Программное обеспечение VME интерфейса состоит из двух независимых компонентов – библиотеки `libvmedirect` и модуля `vmei.o`. Библиотека используется для реализации функций ввода-вывода через VME интерфейс и содержит набор подпрограмм для чтения и записи данных нескольких форматов через различные адресные пространства VME. Библиотека собирается в единый исполняемый модуль вместе с приложением и обращается к аппаратуре VME интерфейса из пользовательского адресного пространства, путем прямой адресации к окнам в физической памяти контроллера. Модуль `vmei.o` представляет собой загружаемый Linux драйвер, предназначенный для приема прерываний VME интерфейса и передачи их пользовательскому приложению для дальнейшей обработки.

Рисунок 3. Структура программного обеспечения VME интерфейса



7.2. Библиотека libvmedirect

Библиотека состоит из функций инициализации, управления настройками VME интерфейса, ввода-вывода и закрытия. Код библиотеки содержится в файле libvmedirect.a. Часть функций библиотеки носят inline характер и находятся в заголовочном файле libvmedirect.h.

7.2.1. Функции инициализации/закрытия

int libvme_init(void)

В начале работы с библиотекой приложение должно вызвать данную функцию. При успешном выполнении возвращаемое значение 0, в противном случае отрицательное число.

void libvme_close(void)

При завершении работы с библиотекой приложение должно вызвать функцию закрытия.

7.2.2. Установка параметров VME интерфейса

int libvme_ctl(int mode, int *value)

Функция предназначена для установки режимов VME интерфейса. Режимы, устанавливаемые данной функцией, не защищены от множественного доступа. Решение этой проблемы остается на усмотрение разработчика прикладного ПО. Возвращаемое значение функции – 0 в случае успешного выполнения, отрицательное число при ошибке.

Таблица 13. Параметры функции libvme_ctl и их описание

mode	комментарий
VME_AM_W	Младшие 5 бит параметра задают значение адресного модификатора, в соответствии с описанием Reg0.
VME_AM_R	Чтение адресного модификатора, значение находится в 5 младших битах параметра.
VME_BUSL_W	Младшие 2 бита параметра задают уровень запроса шины в соответствии с описанием Reg0.
VME_BUSL_R	Чтение уровня запроса шины, значение находится в младших 2-х битах параметра.
VME_SYSTEM_W	Если параметр =1, устанавливается режим системного контроллера. Если параметр =0, режим системного контроллера сбрасывается.
VME_SYSTEM_R	Чтение режима системного контроллера. Полученное значение помещается в параметр (см. предыдущий пункт).

VME_ARB_W	Если параметр =1, устанавливается режим кругового арбитра. Если параметр =0, устанавливается режим приоритетного арбитра. Опция имеет значение только для режима системного контроллера.
VME_ARB_R	Чтение режима арбитра. Полученное значение помещается в параметр (см. предыдущий пункт).
VME_SYSFAIL_W	Если параметр =1, активируется сигнал SYSFAIL. Если параметр =0, сигнал SYSFAIL сброшен.
VME_SYSFAIL_R	Чтение сигнала SYSFAIL на шине VME (1 – не активен, 0 – активен).
VME_ACFAIL_R	Чтение сигнала ACFAIL на шине VME (1 – не активен, 0 – активен).

7.2.3. Функции ввода-вывода

Функции ввода-вывода являются inline функциями, что существенно уменьшает накладные расходы при циклическом обращении к ним. Соответственно, при компиляции приложения необходимо установить флаги компилятора, разрешающие inline функции.

Возвращаемое значение всех функций:

0 – цикл выполнен успешно

-1 – нет ответа от VME модуля

Описанные ниже функции обращаются к одному из трех адресных пространств VME, в имени функции обозначенному как XXX. В соответствии с адресным пространством должны быть заданы диапазоны допустимого физического адреса, передаваемого параметром bus_addr.

XXX	bus_addr	комментарий
a16	00000000..0000FFFF	адресное пространство a16, 64 Кбайт
a24	00000000..00FFFFFF	адресное пространство a24, 16 Мбайт
a32	00000000..FFFFFFFF	адресное пространство a32, 4 Гбайт

static __inline__ int

libvme_read_XXX_byte(unsigned int bus_addr, unsigned char *byte)

Чтение байта из адресного пространства XXX.

static __inline__ int

libvme_read_XXX_word(unsigned int bus_addr, unsigned short *word)

Чтение слова из адресного пространства XXX.

static __inline__ int

libvme_read_XXX_dword(unsigned int bus_addr, unsigned int *dword)

Чтение двойного слова из адресного пространства XXX.

static __inline__ int

libvme_write_XXX_byte(unsigned int bus_addr, unsigned char byte)

Запись байта в адресное пространство XXX.

static __inline__ int

libvme_write_XXX_word(unsigned int bus_addr, unsigned short word)

Запись слова в адресное пространство XXX.

static __inline__ int

libvme_write_XXX_dword(unsigned int bus_addr, unsigned int dword)

Запись двойного слова в адресное пространство XXX.

7.3. Модуль vmei.o

Модуль реализует программный интерфейс прерываний от VME шины и является драйвером символьного устройства Linux. Обращение к драйверу осуществляется через специальные файлы устройств /dev/vmei0../dev/vmei7. Файл устройства /dev/vmei0 обслуживает прерывание от сигнала AC FAIL, файлы /dev/vmei1../dev/vmei7 обслуживают линии прерываний IRQ1..IRQ7 соответственно. Драйвер vmei реализует для прикладной задачи следующие вызовы: open/close, select, read.

open/close

Для обслуживания прерывания приложение должно сначала открыть соответствующий специальный файл. Одна линия прерывания обслуживается не более чем одним приложением.

select

При вызове этой системной функции происходит разрешение соответствующей файловому дескриптору линии прерывания и приложение переходит в режим ожидания. Выход из ожидания осуществляется по истечении таймута, заданного в параметре select, или при получении прерывания по линии IRQx. Прерывание маскируется (запрещается) снова в процедуре обработки.

read

Вызов read завершает обработку прерывания. Приложение через этот системный вызов может прочитать вектор, который был выставлен на шину устройством, запросившим прерывание. Чтение вектора должно осуществляться только в 4-х байтовый буфер.

Сам вектор представляет собой целое число:

Положительные значения 0..255 говорят об успешно выполненном цикле подтверждения прерывания.

Значение -1 соответствует ошибке шины при выполнении цикла подтверждения (отсутствие сигнала DTACK или наличие сигнала BERR).

Значение -2 говорит о завершении select по таймауту до получения прерывания.

Вектора прерываний не устанавливаются в очередь, каждое последующее прерывание переписывает старое значение вектора.

7. RTC таймер и батарейная память

Часы реального времени и встроенная в них статическая память продолжают функционировать при выключении питания контроллера. Питание этих узлов осуществляется от литиевой батареи. Сохранение работоспособности часов и памяти гарантируется производителем на срок до 10 лет. Область статической памяти отображается в адресное пространство контроллера и занимает 256 байт, из которых 8 старших ячеек отведено для управления часами реального времени. Назначение регистров управления часами приведено в документации на микросхему RTC M48T02. Остальные 248 ячеек могут быть использованы как прямо адресуемая память с батарейным резервированием. Прерываний от RTC не предусмотрено.

Для доступа к RTC могут быть использованы следующие функции, входящие в состав libvmedirect:

void libvme_rtc_start(void)

void libvme_rtc_stop(void)

Запустить или остановить RTC. Запуск может понадобиться при активации нового контроллера. Останов RTC переводит часы в режим экономии батареи.

void libvme_rtc_settime(int seconds, int minutes, int hours)

Установить время, заданное в параметрах функции.

void libvme_rtc_gettime(int *seconds, int *minutes, int *hours)

Возвращает текущее время

void libvme_rtc_setdate(int date, int month, int year)

Устанавливает дату, заданную в параметрах функции. Год 00..99.

void libvme_rtc_getdate(int *date, int *month, int *year)

Возвращает текущую дату.

Все функции работоспособны только после вызова libvme_init.

8. CAN интерфейсы

Контроллер BIVME-2 имеет два установленных CAN интерфейса, выполненных на базе микросхем SJA1000 фирмы Philips. Управление интерфейсами осуществляется через регистры, отображаемые в память контроллера. Каждый интерфейс имеет 32 байтовых регистра, назначение которых приведено в описании SJA1000.

Прерывания от обоих интерфейсов объединены по «ИЛИ» и подключены к линии IRQ6 процессора. Таким образом, программного обеспечения, управляющего CAN интерфейсами, должно работать в режиме разделяемых прерываний.

Доступ к CAN интерфейсам может осуществляться через драйвер can4linux. При этом конфигурация каналов 0 и 1 в файле can.conf должна быть следующая:

```
Chipset_0=sja1000  
Base_0=0xFFF48000  
Irq_0=12
```

```
Chipset_1=sja1000  
Base_1=0xFFF48020  
Irq_1=12
```

Методика работы через драйвер описана в документации can4linux.

