

# **MDV – отладчик для процессоров серии «MULTICORE»**

## **Руководство оператора**

Версия 6.0

## Содержание

1.	Аннотация.....	4
2.	Назначение программы.....	4
3.	Условия выполнения программы.....	4
3.1	Требования по аппаратной части.....	4
3.2	Требования по программной части.....	5
4.	Выполнение программы.....	5
4.1	Установка.....	5
4.2	Режимы работы.....	5
4.2.1	Режим чтения команд из файла.....	6
4.2.2	Интерактивный режим.....	6
4.2.3	Режим потока ввода-вывода.....	7
4.3	Ключи командной строки при запуске отладчика.....	7
4.4	Команды отладчика.....	8
4.4.1	help.....	8
4.4.2	?.....	8
4.4.3	run.....	8
4.4.4	step.....	8
4.4.5	sleep.....	9
4.4.6	reset.....	9
4.4.7	set.....	9
4.4.8	show.....	9
4.4.9	tlb.....	9
4.4.10	source.....	10
4.4.11	load.....	10
4.4.12	save.....	10
4.4.13	loadbin.....	11
4.4.14	savebin.....	11
4.4.15	loadelf.....	11
4.4.16	dump.....	11
4.4.17	sbp.....	11
4.4.18	hbp.....	12
4.4.19	wp.....	12
4.4.20	conf.....	12
4.4.21	exit.....	13
4.4.22	quit.....	13
4.4.23	listdevs.....	13
4.4.24	opendev.....	13
4.4.25	closedev.....	13
4.4.26	clearflash.....	13
4.4.27	testflash.....	14
4.4.28	testmem.....	14
4.4.29	trace.....	14
4.4.30	core.....	14
4.4.31	connect.....	15
4.4.32	loaddesc.....	15
4.4.33	version.....	15
4.4.34	dumpcache.....	15
4.4.35	invdcache.....	15
4.4.36	jtagsspeed.....	15
5.	Формат регистров для команд set и show.....	16
6.	Примечания.....	16



## 1. Аннотация.

Данный документ является руководством пользователя по применению отладчика MDB. В документе приведено назначение программы, условия её выполнения с точки зрения аппаратной и программной части, установка, режимы работы отладчика, система команд. В документе применяются следующие обозначения:

- шестнадцатеричные значения начинаются с префикса '0x', если префикс '0x' отсутствует, то значение считается десятичным;
- параметры команд отладчика, указанные в квадратных скобках, являются необязательными.

## 2. Назначение программы.

Отладчик **MDB** – программа, обеспечивающая удаленную отладку программ на процессорах семейства «multicore» через эмулятор USB-JTAG.

В это семейство входят:

- NVCom-01;
- NVCom-02T;
- MC-24R2;
- MC-24M;
- LDE-Vega;
- NVCom-02;
- MC-0428;
- MCT-03P;
- MC-12M;
- MC-226M;
- MC-30SF6;
- MCom-02.

## 3. Условия выполнения программы.

MDB распространяется под операционные системы семейства Windows NT и дистрибутива CentOS 7.

### 3.1 Требования по аппаратной части.

Минимальные требования к ПЭВМ для работы:

Процессор:

- Intel Core 2 Duo;
- AMD Phenom.

Оперативная память:

- Требования не предъявляются.

Видео карта:

- Требования не предъявляются.

Жесткий диск:

Количество памяти должно соответствовать требованиям, предъявляемым к операционной системе.

### 3.2 Требования по программной части.

Multicore JTAG Server (mjtagserver).

Для ОС семейства Windows NT:

USB драйвер из libusb-win32;

Для ОС дистрибутива CentOS 7:

libusb.i686;

ftdi.i686;

## 4. Выполнение программы.

### 4.1 Установка.

1. Установить USB драйвер для эмулятор USB-JTAG. Подключить эмулятор USB-JTAG адаптер к ПК.

Для ОС семейства Windows NT:

Из директории "Drivers", с правами администратора, запустить

\_install\_libusb\_amd64.bat - для 64bit систем

или

\_install\_libusb\_x86.bat - для 32bit систем.

Для ОС дистрибутива CentOS 7:

С помощью программы yum, которая является консольным менеджером RPM-пакетов, установить libusb.i686, ftdi.i686 .

2. Установить mjtagserver.

Из директории "mjtagserver\_installer", с правами администратора, запустить

Для ОС семейства Windows NT:

install-win.bat

Для ОС дистрибутива CentOS 7:

install-redhat.sh

### 4.2 Режимы работы.

Отладчик MDB может работать в следующих режимах:

- режим чтения команд из файла;
- интерактивный режим;
- режим ввода-вывода.

### 4.2.1 *Режим чтения команд из файла.*

Отладчик позволяет выполнять команды из файлов, при этом файлы команд могут содержать комментарии, пустые строки и «вызовы» команд из других файлов. Комментарии начинаются с символа '#' и продолжаются до конца строки. В одной строке файла должна находиться только одна команда. Пустые строки в файле отладчик интерпретирует как комментарии. Файл должен быть создан в текстовом формате.

### 4.2.2 *Интерактивный режим.*

В интерактивном режиме отладчик способен выполнять одну команду из командной строки. В интерактивном режиме работы для удобства пользователя реализованы следующие возможности:

#### 1. Редактирование команд.

В интерактивном режиме отладчик использует библиотеку GNU Readline и позволяет редактировать командную строку при помощи комбинаций клавиш.

Например:

Ctrl-w удаляет слово перед курсором.

Ctrl-a перемещает курсор в начало строки.

Ctrl-e перемещает курсор в конец строки.

Ctrl-r производит обратный поиск в истории команд (reverse incremental search).

Esc-f перемещает курсор на слово вперед.

Esc-b перемещает курсор на слово назад.

Более подробно о возможностях редактирования командной строки можно узнать из документации на библиотеку Readline.

#### 1. Command history.

Отладчик сохраняет историю команд и позволяет быстро вернуться к набранной ранее команде с помощью стрелки «вверх».

#### 2. Tab-completion.

При нажатии на клавишу Tab, отладчик завершает набор частично набранной команды или параметра. При двойном нажатии на Tab, отображаются возможные варианты продолжения.

#### 3. Command prediction.

Если команда набрана не полностью, то отладчик пытается дополнить команду до полного имени и потом запустить ее. Эта возможность позволяет выполнять команды, не набирая их полностью, ускоряя работу в интерактивном режиме и повышая её комфортность.

#### 4. Command repeating.

Если осуществляется перевод строки, но команда не задана (пустая линия), то отладчик повторяет выполнение предыдущей команды. Эта особенность очень удобна при исполнении отлаживаемой программы по шагам и для вывода дампа памяти.

## 5. Shell Command Execution.

Отладчик позволяет выполнять внешние программы. Для этого нужно набрать '!' перед именем команды. Например: '!ls -l' распечатает содержимое текущей директории.

### 4.2.3 *Режим потока ввода-вывода.*

Отладчик также способен выполнять команды из стандартного потока ввода-вывода. Такие возможности отладчика позволяют использовать его как универсальный инструмент из других программ и надстроек. Такими программами могут являться интегрированная среда разработки или графический интерфейс для отладчика. Кроме того, отладчик может вызываться из языка-интерпретатора (например perl) и выполнять команды под его управлением, обеспечивая взаимодействие с нижним уровнем скрытое для пользователя. Использование отладчика из других программ позволяет реализовывать циклы, условное исполнение команд и автоматически анализировать полученные результаты. Такое применение очень удобно, например, для реализации наборов тестов.

## 4.3 Ключи командной строки при запуске отладчика.

Отладчик распознает следующие ключи командной строки:

- h Помощь по ключам командной строки.
- n Не начинать работу с первым попавшимся устройством.
- s cmd Выполнить одну команду и выйти из отладчика.

При использовании ключа -s отладчик выполняет заданную команду и завершается. Если команда содержит параметры, то необходимо использовать кавычки.

- f file Выполнять команды из указанного файла.

При использовании ключа -f отладчик переходит в интерактивный режим и выполняет команды из указанного файла, выводя результат на экран. При использовании ключа -q, совместно с -f, на экран кроме ошибок ничего выводиться не будет. После завершения команд, отладчик остается в интерактивном режиме.

- q Quiet mode. В этом режиме отладчик ничего не выводит на экран кроме ошибок. Этот ключ может быть полезен при использовании совместно с -f.

- v Подробный режим ведения лога.
- w Проверка загруженной программы в память из ELF файла.

При использовании ключей -h и -s отладчик не переходит в интерактивный режим.

После запуска отладчик выводит свою версию, дату сборки, список подключённых устройств, затем выводит приглашение к вводу команд "mdb>".

Например:  
Successfully connected to \\.\Pipe\mdb\_pipe.  
mdblib version: 6.0.0.2248<976f6867cc> (Nov 19 2015)  
List of suitable devices:  
0. NVCom-02T on EZ-USB0  
Opening device: EZ-USB0  
mdb>

Далее пользователь вводит команды и отладчик выполняет их. После выполнения команды отладчик снова выводит prompt "mdb> " и ожидает ввода следующей команды.

## 4.4 Команды отладчика.

В этом разделе описываются команды отладчика, их формат и операции, которые они выполняют.

### 4.4.1 *help*

Синтаксис: **help [command]**

Описание:

Выводит краткую справку по заданной команде или по всем командам, если команда не указана.

Пример:

mdb> help tlb

### 4.4.2 *?*

Синтаксис команды: **? [command]**

Описание:

Аналогично команде help.

Пример команды:

mdb> ? tlb

### 4.4.3 *run*

Синтаксис: **run [addr]**

Описание:

Переводит ядро процессора в состояние исполнения кода, после чего ожидает его останова.

Если адрес запуска не задан, то исполнение кода выполняется без изменения PC. Если адрес задан, то в PC заносится адрес старта и процессор запускается на исполнение.

Пример:

mdb> run 0xbfc01000

### 4.4.4 *step*

Синтаксис: **step [N]**

Описание:

По данной команде ядро процессора выполняет одну или N инструкций и возвращается в режим отладки. При большом N, команда может выполняться достаточно долго. Если N не указан, то выполняется одна инструкция.



#### 4.4.5 *sleep*

Синтаксис: **sleep [sec]**

Описание:

Пауза. Эта команда может быть полезна для использования в командных файлах. Пауза может быть прервана по Ctrl-C.

#### 4.4.6 *reset*

Синтаксис: **reset**

Описание:

Сброс процессора в начальное состояние.

#### 4.4.7 *set*

Синтаксис: **set (regname | address | symbol) value**

Описание:

Устанавливает регистр 'regname' или память в заданное значение. Адрес памяти может быть задан как через отладочный глобальный символ symbol, так и явно через виртуальный 32-х битовый адрес. Формат регистров описан в разделе 5.

Пример:

```
mdb> set 0xbfc00010 0x12344321
mdb> set RISC1.pc 0x80000000
mdb> set var1 0x4
```

#### 4.4.8 *show*

Синтаксис: **show (regname1 | address1 | symbol1) (regname2 | address2 | symbol2) ...**

Описание:

Отображает содержимое перечисленных регистров и памяти. Адрес памяти может быть представлен так же, как и в команде 'set'. Формат регистров описан в разделе 5. При некорректно заданном адресе выдается ошибка.

Пример:

```
mdb> show 0xbfc00010
0xbfc00010 : 0xffffffff
mdb> show DSP0.pc
PCU.PC : 0x0000 (B8480120)
mdb> show var1
var0 : 0xf
mdb> show regAndVariableThatDoesNotExist
"regAndVariableThatDoesNotExist" is neither a register nor a symbol, wrong name?
```

#### 4.4.9 *tlb*

Синтаксис: **tlb**

Описание:

Команда позволяет отобразить содержимое TLB для всех процессоров семейства «multicore», кроме MCom-02.

Пример:

```
mdb> tlb
```

No	G	ASID	PM	VPN2	PFN0	C0	D0	V0	PFN1	C1	D1	V1
0	0	0x35	0x30c	0x6d5c8	0x392f0	1	0	0	0x19958	0	0	1

1	1	0x36	0x3cf	0x367d2	0x4bb2b	1	1	0	0xd0b96	1	0	0
2	1	0x2a	0xff3	0x041fd	0x30deb	1	1	1	0x448e2	1	0	0
3	1	0xf1	0xfc3	0x07b69	0xb045e	0	1	0	0x7eb15	0	0	1
4	0	0x11	0xfc0	0x35d23	0x2260a	0	0	1	0x98f06	0	0	0
5	1	0xb2	0xc0f	0x489c1	0x6ef74	0	1	0	0x7bdea	0	1	0
6	1	0xaf	0xcc3	0x35237	0x71636	0	1	0	0x3b9e1	0	0	1
7	0	0x91	0xc33	0x099ad	0xaba7e	1	0	0	0xc8fd9	0	1	1
8	0	0x75	0x0cf	0x6d76b	0xdf62a	0	0	0	0xc6166	1	0	1
9	0	0xd0	0xc00	0x224d2	0x81b2a	0	0	0	0x8da23	1	1	0
10	1	0x21	0x03c	0x3d5d7	0xb0f06	1	1	1	0x6c7fc	0	1	1
11	0	0x58	0x3ff	0x76c25	0x53f66	0	0	0	0xae2a9	0	1	0
12	1	0x67	0xf0c	0x004fd	0xe2caf	0	1	0	0x8fe5a	1	1	1
13	0	0x5a	0xc0f	0x5156c	0x40fb9	0	1	0	0xf85a6	0	0	0
14	0	0x2b	0xf33	0x62c9b	0x43678	0	0	0	0x3a175	1	0	0
15	0	0x96	0x003	0x3462a	0xbe436	1	1	1	0x61d56	0	1	1

#### 4.4.10 *source*

**Синтаксис:** `source filename`

**Описание:**

Исполняет команды из указанного файла (отладчик переходит в режим чтения команд из файла).

#### 4.4.11 *load*

**Синтаксис:** `load filename address`

**Описание:**

Загружает программу в память из текстового файла. Запись строки, в текстовом файле, имеет следующий формат:

`32bit_word 32bit_word 32bit_word 32bit_word //address`

,где

32bit\_word – 32-ух битовое слово по основанию 16.

address – 32-ух битовый адрес.

Пример содержимого файла:

`bfc06538 bfc00074 bfc00080 bfc00ff0 //bfc00060`

`1000ffec 00000000 //bfc00070`

**Пример:**

`mdb> load program.txt`

#### 4.4.12 *save*

**Синтаксис:** `save filename (address | symbol) size`

**Описание:**

Сохраняет содержимое памяти в формате совместимом с командой load. Начало блока памяти может быть установлено как адресом, так и отладочным глобальным символом. **Пример:**

`mdb> save file.txt 0xbfc00000 0x1000`

`mdb> save file.txt main 0x1000`

#### 4.4.13 *loadbin*

Синтаксис: **loadbin filename address**

Описание:

Загружает содержимое файла filename в память по адресу address.

#### 4.4.14 *savebin*

Синтаксис: **savebin filename (address | symbol) size**

Описание:

Сохраняет size байт памяти с адреса address или отладочного глобально символа symbol в файл filename.

#### 4.4.15 *loadelf*

Синтаксис: **loadelf filename**

Описание:

Загружает содержимое ELF файла в память. Записывает значение entry point в PC. После успешного выполнения этой команды отладчик позволяет использовать имена глобальных символов из ELF файла в командах show/set/sbp/hbp/wp/save/savebin/dump.

#### 4.4.16 *dump*

Синтаксис: **dump [(address | symbol) size]**

Описание:

Выводит шестнадцатеричный дамп памяти с адреса address или отладочного глобального символа symbol. Если address или symbol не задан, то дамп выводится с адреса следующего за последним распечатанным словом при предыдущем выполнении этой команды. size - размер в байтах, должен быть кратен 4. Если параметр size не задан, то используется значение size от предыдущей команды. По умолчанию значение size равно 256. Пример:

```
mdb> dump 0xbfc00000 0x200
```

#### 4.4.17 *sbp*

Синтаксис: **sbp (set | unset) (vm\_addr | symbol) [length]**

Описание:

Команда управления программными точками останова (software breakpoints). При вызове без параметров выводит информацию об установленных программных точках останова.

set – устанавливает точку останова по адресу vm\_addr или по отладочному глобальному символу symbol и выводит её номер.

unset – удаляет точку останова по адресу vm\_addr или по отладочному глобальному символу symbol.

length – длина инструкции на которую будет установлена точка останова. Значение длины инструкции, по умолчанию, составляет 4 байта.

Пример:

```
mdb> sbp set 0x80000000
```

```
Software breakpoint number 0 is set.
```

```
mdb> sbt unset 0x80000000
```

```
Software breakpoint number 0 is deleted.
```

#### 4.4.18 *hbp*

**Синтаксис:** **hbp (set | unset) (vm\_addr | symbol) [length]**

**Описание:**

Команда управления аппаратными точками останова (hardware breakpoints). Описание аргументов аналогично команде sbp.

#### 4.4.19 *wp*

**Синтаксис:** **wp (set | unset) (r | w | rw) (vm\_addr1 | symbol1) [ (vm\_addr2 | symbol2) ]**  
**wp clear num N**

**Описание:**

Команда управления точками останова по обращению к памяти (watchpoints).

set – установить watchpoint

unset – удалить watchpoint

r - read (чтение)

w - write (запись)

rw - read/write (чтение и запись)

clear num – удаление watchpoint по номеру N.

Если задан только адрес vm\_addr1 или отладочный глобальный символ symbol1, то точка останова ставится только на это адрес. Если указан vm\_addr2 или symbol2, точка останова ставится на диапазон адресов [ (vm\_addr1 | symbol1); (vm\_addr2 | symbol2) ].

**Пример:**

```
mdb> wp set rw 0x80000000
```

```
Watchpoint breakpoint number 0 is set.
```

```
mdb> wp clear num 0
```

```
Watchpoint at range 0x80000000-0x80000000 is deleted.
```

Команды hbp и wp используют одни и те же аппаратные ресурсы. Нельзя одновременно задать breakpoint и watchpoint с одинаковым номером.

#### 4.4.20 *conf*

**Синтаксис:** **conf [set | unset option]**

**Описание:**

Команда управления настройками mdb. При вызове без параметров выводит список и состояние настроек.

set – установить настройку в значение «истина»

unset – сбросить настройку в значение «ложь»

option – название настройки.

Для пользователя доступны следующие настройки:

blkio – блочный режим чтения/записи (значительно ускоряет работу, если поддерживается эмулятором USB-JTAG)

checkwrites – проверка успешности записи данных из ELF файла.

output – ведение лога.

verbose – расширенный режим ведения лога.

expert – отключение, в MDB, вспомогательных функций обработки отладочных событий.

gdb – включение совместимости с gdb.

profiler – включение профилирования. Вывод профилирования помещается в файл tmpProfile. Он находится в той же директории где и MDB.

#### 4.4.21 *exit*

Синтаксис: **exit**

Описание:

Выход из MDB.

#### 4.4.22 *quit*

Синтаксис: **quit**

Описание:

Аналогично команде exit.

#### 4.4.23 *listdevs*

Синтаксис: **listdevs**

Описание:

Перечислить все доступные для отладки устройства.

Пример:

mdb> listdevs

List of suitable devices:

0. nvcom on EZ-USB0

#### 4.4.24 *opendev*

Синтаксис: **opendev [num]**

Описание:

Выбрать устройство, отладка которого будет производиться.

num – номер устройства в списке выведенным listdevs.

Пример:

mdb> opendev 0

Opening device: EZ-USB0

#### 4.4.25 *closedev*

Синтаксис: **closedev**

Описание:

Освободить текущее отлаживаемое устройство.

**setflash**

Синтаксис: **setflash [address]**

Описание:

Указать адрес, с которого начинается флеш-память. Команда необходима для корректной работы записи во флэш-память.

Адрес по умолчанию для всех поддерживаемых процессоров, на данный момент, кроме Mcom-02 — 0xbc000000.

#### 4.4.26 *clearflash*

Синтаксис: **clearflash base\_address [sector\_address]**

Описание:

Если задан `sector_address`, стереть сектор флеш-памяти, находящийся по адресу `base_address + sector_address`. Иначе стереть всю флеш-память, начинающуюся с адреса `base_address`.

#### 4.4.27 *testflash*

Синтаксис: **testflash [base\_address [data\_size]]**

Описание:

Проверка флеш-памяти, размера `data_size`, с указанием адреса `base_address`, с которого она начинается.

#### 4.4.28 *testmem*

Синтаксис: **testmem vm\_addr size [passno]**

Описание:

Проверка памяти, размера `size`, начиная с виртуального адреса `vm_address`. `passno` – количество итераций проверки.

#### 4.4.29 *trace*

Синтаксис: **trace**

**trace trace\_source filename**

**trace ep endpoint\_index new\_filename**

Описание:

Управление выводом трассировки. Вызов без параметров отображает текущие потоки вывода трассировки. Команда с названием файла `filename` из источника трассы `trace_source` помещает вывод трассы в этот файл. Источник трассы - программный модуль `mdbmon`. `Mdbmon` линкуется при сборке отлаживаемой программы. Вывод осуществляется через функцию `printf`. Формат трассы определяется в первом аргументе `printf`. Команда с индексом потока вывода трассы `endoint_index` и названием файла `new_filename` перенаправляет вывод трассы в этот файл. Индекс потока вывода трассы можно получить при запуске команды без параметров.

Пример:

```
mdb> trace mdbmon traceLog.txt
```

```
mdb> trace
```

```
Stream Id:   File:
```

```
2           traceLog.txt
```

```
mdb> trace ep 2 newTraceLog.txt
```

```
mdb> trace
```

```
Stream Id:   File:
```

```
2           newTraceLog.txt
```

#### 4.4.30 *core*

Синтаксис: **core [coreName]**

Описание:

Выбрать ядро процессора, отладка которого будет осуществляться. Такие команды как `sbr`, `hbr`, `step`, `run` будут применяться к этому ядру. Вызов без параметров выведет список доступных ядер.

`coreName` – имя ядра.

Пример:

```
mdb> core
```

```
RISC0
RISC1
DSP0
DSP1
*ALL
mdb> core RISC0
mdb> core
*RISC0
RISC1
DSP0
DSP1
ALL
```

#### 4.4.31 *connect*

Синтаксис: **connect**  
**connect host:port**  
**connect (pipe | local\_socket )**

Описание:

Подключиться к mjttagserver'у по адресу host:port по протоколу TCP/IP, либо через именованный канал pipe (под Windows), либо через доменный сокет local\_socket (под Linux). Вызов без параметров вызывает подключение к серверу на локальном компьютере.

#### 4.4.32 *loaddesc*

Синтаксис: **loaddesc [file]**

Описание:

Загрузить xml-файл с описанием всех известных процессоров. При вызове без параметров будет использовано описание, вшитое в mdb при сборке.

#### 4.4.33 *version*

Синтаксис: **version**

Описание:

Вывести информацию о номере версии и дате сборке mdb.

#### 4.4.34 *dumpcache*

Синтаксис: **dumpcache [filename]**

Описание:

Сохранить информацию о кеше данных и инструкций в файлы с названиями icache\_filename, itag\_filename, dcache\_filename, dtag\_filename. Эта команда не работает для Mcom-02.

#### 4.4.35 *invdcache*

Синтаксис: **invdcache**

Описание:

Сбросить локальный кэш регистров процессора в MDB.

#### 4.4.36 *jtagsspeed*

Синтаксис: **jtagsspeed hz**

Описание:

Установить частоту работы JTAG в Гц, на эмуляторе USB-JTAG.

Пример:

```
mdb> jtagspeed 1000000
```

## 5. Формат регистров для команд set и show.

Синтаксис: device\_name.reg\_name

Описание:

, где

device\_name – имя устройства к которому относится регистр.

reg\_name – имя регистра.

Пример:

DSP0.PC

dsp0.pc

MFBSP0.CSR

## 6. Примечания.

При доступе к памяти, отладчик производит преобразование адресов из виртуальных в физические. Поэтому пользователь должен использовать виртуальные адреса.

Большинство команд отладчика могут быть прерваны по Ctrl-C. Если команда в это время ожидает останова CPU, то отладчик запрашивает режим отладки.

Если команда была введена с ошибками, то отладчик выведет формат этой команды, которого необходимо придерживаться.

Программный модуль mdbmon является частью библиотеки newlib.



Версия документа	
1.00 (24.05.2016 г)	Начальная ревизия документа