

ЗАПУСК ПРОГРАММЫ НА МИКРОСХЕМЕ 1892ВМ14Я В РЕЖИМЕ ЗАГРУЗКИ ЧЕРЕЗ UART



____ОАО НПЦ «ЭЛВИС»

support@elvees.com, www.multicore.ru



оглавление

1.BBE	ĮЕНИЕ	. 3
2. CBEJ	ЦЕНИЯ О ПРОЦЕДУРАХ СБРОСА И НАЧАЛЬНОЙ ЗАГРУЗКИ	. 4
3. CO3Д	ĮАНИЕ И СБОРКА ПРОЕКТА В MCSTUDIO	. 6
3.1 3.2	Допустимые адреса для сборки проекта Перенос настроек регистров в текст	. 6 . 6
4. ЗАГР	УЗКА ПРОЕКТА ЧЕРЕЗ UART	10
4.1 4.2	Получение файла в формате Intel-HEX Загрузка программы	10 12



1. ВВЕДЕНИЕ

В документе описан процесс сборки программы для микросхемы 1892ВМ14Я, с последующей её загрузкой по интерфейсу UART0. Процесс загрузки рассмотрен на примере отладочного модуля Салют-ЭЛ24ОМ1.

2. СВЕДЕНИЯ О ПРОЦЕДУРАХ СБРОСА И НАЧАЛЬНОЙ ЗАГРУЗКИ

Режим загрузки определяется состоянием внешних выводов BOOT[2:0] микросхемы. Значения этих выводов при получении сигнала сброса заносятся в регистр BOOT, формат которого приведен в Таблица 2.1.

Номер разряда	Обозначение	Назначение	Доступ	Исходное состояние
31:3	-	Не используется	R	0
2:0	BOOT	Значение внешних выводов ВООТ, определяющих источник начальной загрузки микросхемы. 0x0 – загрузка из внешней памяти NOR Flash/SRAM с помощью контроллера NORMPORT; 0x1 – загрузка из внешней памяти NAND Flash с помощью контроллера NANDMPORT; 0x2 – загрузки из накристальной ROM памяти и UART0; 0x3 – загрузки из накристальной ROM памяти и SPI0; 0x4 – загрузки из накристальной ROM памяти и SDMMC0; 0x5-0x7 – зарезервировано.	R	-

Таблица 2.1. Формат регистра ВООТ

Для включения режима загрузки из UART необходимо установить выводы BOOT в состояние «010». На отладочном модуле Салют-ЭЛ24OM1 это можно сделать при помощи переключателя SA1, показанного на Рисунок 2.1.



Рисунок 2.1 Переключатель SA1 на отладочном модуле Салют-ЭЛ24ОМ1

€ЭЭЛBUC

3. СОЗДАНИЕ И СБОРКА ПРОЕКТА В МСЅТИДІО

Процедура создания и сборки проекта, предназначенного для загрузки через UART, в среде MCStudio происходит так же, как для проекта с загрузкой через JTAG.

3.1 Допустимые адреса для сборки проекта

Оптимальным вариантом является сборка проекта в адресах накристальной памяти. Наиболее удобно для этих целей использовать внутреннюю память RAM, так как она не требует никаких предварительных настроек. Также допустима сборка в адресах VRAM и XYRAM, но для этого требуется предварительно включить тактирование этих областей памяти.

Если к NORMPORT подключено внешнее статическое O3У, то можно загрузить в него программу, собранную в адресах этого внешнего O3У. Для этого необходимо будет предварительно проинициализировать внешнюю память командами из консоли UART, записав нужные значения в соответствующие регистры.

Использование DDR-памяти также требует инициализации, однако выполнение этой процедуры через UART является довольно трудоёмкой задачей. Поэтому для программы, собранной в адресах DDR, рекомендуется использовать дополнительную программузагрузчик.

3.2 Перенос настроек регистров в текст

При загрузке микросхемы значения регистров, отвечающих за тактирование и настройку частот, не сохраняются.

Во время загрузки проекта через JTAG программа-отладчик производит настройку этих регистров непосредственно перед записью программы в память. В среде разработки MCStudio 3A эти настройки указаны в окне Options->Device (см. Рисунок 3.1); в среде MCStudio 4 – в окне Run->Debug Configurations->%Название конфигурации%->Gdb Init (Рисунок 3.2). При необходимости они могут быть изменены. Настройки из этих окон в текст программы не попадают.

C))/BUC

Options	
oject Paths Sections Device Debugger Tools Connectio	n]
MCom-02 SimCore.dll	
set 0x38094068 0xf # CMCTR.GATE_DSP_CT	R
set 0x3809410c 0x1 # CMCTR.SEL_SPLL	
set 0x38094100 0x2 # CMCTR.SEL_APLL	
set 0x38094104 0x2 # CMCTR.SEL CPLL	
set 0x38094114 0x2 # CMCTR.SEL_UPLL	
set 0x38094108 0x4 # CMCTR.SEL_DPLL	· · · · ·
set 0x38094048 0x21 # CMCIR.GAIL_CORE_C	IK SVS CTD
Set 0x3809404C 0x11111111 # CMCIR.GATE_	SIS_CIR
# setup registers to allow high frequen	ces
set 0x38094004 0x1 # DIV MPU CTR = 1:	
set 0x38094008 0x3 # DIV ATB CTR = 3;	
set 0x3809400c 0x1 # DIV APB CTR = 1;	
set 0x38094014 0x1	.;
1	T
libModel3MdbBridge.dll	
Port: EZ-USB15	Platform : MCom-02
	OK Cancel
	Cancer

Рисунок 3.1. Настройка регистров в MCStudio 3A



Debug Configurations	The I may be done that you and h	×
Create, manage, and run configuration	ons ebugging in not possible	-
Image: Second secon	Name: New_configuration (1) Main Gdb Init, Debug Options Advanced Use Default	Restore Default
Multicore Debug Configuration New_configuration New_configuration (1)	Gdb init file content: set architecture arm set tcp auto-retry on set tcp connect-timeout 30 set remotetimeout 120 target remote :@@PORT@@ monitor set 0x38094100 0x1 # CMCTR.SEL_SPLL monitor set 0x38094100 0x2 # CMCTR.SEL_SPLL monitor set 0x38094104 0x2 # CMCTR.SEL_OPLL monitor set 0x38094104 0x2 # CMCTR.SEL_OPLL monitor set 0x38094108 0x1 # CMCTR.GATE_CORE_CTR monitor set 0x3809404 0x1 # fDCTR.GATE_CORE_CTR monitor set 0x3809404 0x1 # fDCTR.GATE_SVS_CTR # setup registers to allow high frequences monitor set 0x3809404 0x1 # mDCTR.BATE_SYS_CTR # setup registers to allow high frequences monitor set 0x3809400 0x1 # DTV_MPU_CTR = 1; monitor set 0x3809400 0x1 # DTV_MPU_CTR = 1; monitor set 0x3809400 0x1 # DTV_MPU_CTR = 1; monitor set 0x3809401 0x1 # fDX_MPU_CTR = 1; monitor set 0x380001 0x1 # fDX_MPU_CTR = 1; monitor set 0	E
Filter matched 4 of 14 items		Revert Apply
0		Debug Close

Рисунок 3.2. Настройка регистров в MCStudio 4

Для того, чтобы при загрузке через UART программа работала корректно, нужно производить настройку всех необходимых регистров отдельно.

Один из способов сделать это – прописать настройку регистров непосредственно внутри кода. Например, настройки, приведённые на Рисунок 3.1, в этом случае будут выглядеть следующим образом:

```
GATE DSP CTR = 0xf; // Включение тактирования DSP
                          // т
SET SPLL = 0x1;
                          11
SET APLL = 0x2;
                              SET CPLL = 0x2;
                          // | Настройка частот блоков PLL
SET UPLL = 0x2;
                          // |
                          // ]
SET DPLL = 0 \times 4;
GATE CORE CTR = 0x21; // Включение тактирования VPU
GATE SYS CTR = 0xfffffff; // Включение всех частот в CMCTR SYS
                           // т
DIV MPU CTR = 0 \times 1;
                           // |
DIV ATB CTR = 0x3;
                                Настройка делителей тактовой частоты
                           // ]
DIV APB CTR = 0x1;
GATE MPU CTR = 0 \times 1;
                       // Включение тактовых частот ATCLK, APCLK
```

Данный способ применим только в том случае, когда нет принципиальной разницы, когда производить настройку регистров – до записи в память, или после.

Этот аспект имеет значение, когда речь идёт о тактировании области памяти, в которую будет производиться запись программы. Для этих целей помимо стандартной области RAM

возможно использование памяти VPU-ядра VRAM и памяти DSP-ядра XYRAM (Таблица 3.1).

Базовый	Конечный	Размер	Объём физической	Описание
адрес	адрес	ооласти	памяти	
2000_0000	2000_FFFF	64 Кбайт	64 Кбайт	RAM
3A40_0000	3A87_FFFF	4,5 Мбайт	256 Кбайт	XYRAM (память DSP-ядра)
3B00_0000	3BFF_FFFF	16 Мбайт	1 Мбайт	VRAM (память VPU-ядра)

Таблица 3.1. Накристальная память, доступная для записи программ

Для того чтобы успешно произвести запись, на нужную область памяти должно быть подано тактирование. Однако после подачи питания на микросхему тактирование на ядра VPU и DSP не подаётся. Следовательно, регистр, разрешающий это тактирование, должен быть настроен после сброса, но до начала загрузки программы. Для этого необходимо воспользоваться командами из консоли UART. Более подробно этот процесс описан в разделе 4.2.

4. ЗАГРУЗКА ПРОЕКТА ЧЕРЕЗ UART

4.1 Получение файла в формате Intel-HEX

Монитор UART поддерживает приём файлов в форматах Intel-HEX и SREC. Для того чтобы получить файл прошивки в формате Intel-HEX, необходимо выполнить следующие действия:

- 1. После сборки проекта найти в папке с проектом elf-файл %Project.o, где %Project имя проекта;
- 2. Скопировать этот файл в папку %MCS3A%\arm-eltools\bin, где %MCS3A% путь, куда установлена среда разработки MCStudio 3A;
- 3. Вызвать утилиту ОВЈСОРУ со следующими параметрами (см. Рисунок 4.1):

```
arm-none-eabi-objcopy.exe -O ihex %Project.o %Project.hex
```

Рисунок 4.1. Пример ввода команды для проекта sample gpio

В результате в папке bin должен появиться файл %Project.hex.

MCStudio может выполнять преобразование с помощью утилиты OBJCOPY самостоятельно. В этом случае файл %Project.hex появится в папке проекта и будет обновляться после каждой сборки.

Чтобы настроить MCStudio 3A на выполнение такого действия, нужно в окне Options->Tools записать команду

arm-none-eabi-objcopy -O ihex %Project.o %Project.hex

как показано на Рисунок 4.2.

В MCStudio 4 elf-файл создаётся с расширением elf (%Project.elf). Поэтому для выставления аналогичной настройки нужно в окне Properties->C/C++ Build->Settings в графе Post-build Steps записать команду

arm-none-eabi-objcopy -O ihex %Project.elf %Project.hex

с указанием названия проекта, как показано на Рисунок 4.3.

0015 5601	ni nie.	Parmen	uniy aloo.	21C011021110.111				
Name	Туре	Apply to	Enable	Command	stdOut	ViewOut	Filter	Add.
gcc	0	3	1	arm-none-eabi-gcc -specs=mcom02.specs -c -gdwarf-2 -DLIB	%File.lst		*.c *.C	
gcc	0	3	1	arm-none-eabi-gcc -specs=mcom02.specs -c -gdwarf-2 -mcpu	rm-none-eabi-gcc -specs=mcom02.specs -c -gdwarf-2 -mcpu %File.lst			
gcc	1	3	1	arm-none-eabi-as -c -gdwarf-2 -mcpu=cortex-a9 -Wall -o %F	%File.lst		*.s *.S *.asm *.ASM	
d	2	1	1	arm-none-eabi-ld -r -N -T %Unit.xl %Files.o -o %Unit			*,*	Edit
objdump	3	1	0	arm-none-eabi-objdump -x %Unit	%Unit.he		*,*	
as	1	4	1	elcore-elvis-elf-as -mcx7 -gstabs -al %File.s -o %File.o	%File.lst		*.s *.S *.asm *.ASM	•
ld	2	2	1	elcore-elvis-elf-ld -e 0 -t -N -T %Unit.xl %Files.o -o %Unit.o			*,*	
objdump	2	2	1	elcore-elvis-elf-objdump -D %Unit.o	%Unit.dis		*,*	Load
elcopy	3	2	1	elcopy -L xxx %Unit.o %Unit			*,*	
gcc	2	0	1	arm-none-eabi-gcc -specs=mcom02.specs -nostartfiles -T %P			*,*	
d	1	0	0	ld -N -T m0211nlib.ld %Units -o %Project.o			*.s *.S *.asm *.ASM	
objdump	3	0	1	arm-none-eabi-objdump -s %Project.o	%Project.full		*,*	
objdump	3	0	1	arm-none-eabi-objdump -x %Project.o	%Project.he		*,*	
objdump	3	2	1	elcore-elvis-elf-objdump -x %Unit	%Unit.he		*,*	
objdump	1	0	1	arm-none-eabi-objdumpdisassemble-all %Project.o	%Project.dis		*.s *.S *.asm *.ASM	
архиват	3	0	0	arm-none-eabi-ar rc lib%Project.a %Units	%File.lst	0	*,*	
индекси	3	0	0	arm-none-eabi-ranlib lib%Project.a	%File.lst		*,*	
objcopy	3	0	1	arm-none-eabi-objcopy -O ihex %Project.o %Project.hex				
				E	1			

Рисунок 4.2

pe filter text	Settings 🗘 🕆 🖒 🔻
 Resource C/C++ Build Build Variables Environment Logging Settings Tool Chain Editor C/C++ General MultiCore Project Propertie Project References Refactoring History 	Configuration: MultiCore_Configuration_Debug [Active] Manage Configurations Manage Configurations Tool Settings Build Steps Build Artifact Binary Parsers Error Parsers Pre-build steps Command: Description:
Kun/Debug Settings	Post-build steps Command: arm-none-eabi-objcopy -O ihex sample_gpio.elf sample_gpio.hex Description:

Рисунок 4.3

11

CEEPEREDEC

4.2 Загрузка программы

Файл Intel-HEX загружается в память микросхемы с помощью эмулятора терминала (например, PuTTY).

Для того, чтобы загрузить программу в память микросхемы, необходимо предварительно выполнить следующие действия:

- 1. Перевести переключатель SA1 в положение «010», соответствующее режиму загрузки по UART, при отключённом питании отладочного модуля;
- 2. Подать питание на отладочный модуль;
- 3. Соединить отладочный модуль с компьютером при помощи интерфейса USB-UART. В отладочном модуле Салют-ЭЛ24OM1 USB-порт выведен на mini-USB розетку XS13;
- 4. Определить СОМ-порт компьютера, на котором проинициализирован драйвер UART-USB:
 - 4.1. Открыть приложение "Диспетчер устройств";
 - 4.2. Определить номер СОМ-порта (см. Рисунок 4.4):



Рисунок 4.4. Определение номера СОМ-порта

- 5. Запустить терминал и установить следующие настройки (см. Рисунок 4.5):
 - Connection Type Serial
 - Serial line COMn, где n номер СОМ-порта, определённый на шаге 4.2
 - Speed 115200



Рисунок 4.5. Задание параметров на примере сеанса в РиТТУ

После подтверждения настроек откроется окно терминала. Внутри него и будет производиться обмен данными по UART.

Перед первой загрузкой файла необходимо выполнить команды терминала:

```
cache 1
autorun 0
```

Первая команда необходима для корректного приема данных при загрузке файлов Intel-HEX, вторая – для отключения автоматического исполнения загруженной программы.

Если предполагается загрузка программы в адреса VRAM или XYRAM, то предварительно необходимо разрешить тактирование соответствующих ядер посредством записи значений в соответствующие регистры. Для этих целей в загрузчике BootROM предусмотрена поддержка команды *set <address> <value>*, где *address –* адрес регистра, *value –* записываемое значение. В данном случае требуется:

• Записать в единицы в 0 и 5 разряды регистра GATE_CORE_CTR для разрешения тактирования VPU-ядра при помощи команды¹:

GJJABUC

¹ Если помимо VPU требуется подать частоту на другие части блока CMCTR_CORE, то для регистра GATE_CORE_CTR требуется подобрать другое значение в соответствии с разделом 2.3.3.13 руководства пользователя на микросхему.

set 0x38094048 0x21

• Записать единицы в первые 4 разряда регистра GATE_DSP_CTR для разрешения тактирования DSP-ядра с помощью команды:

set 0x38094068 0xf

Далее нужно отправить в терминал UART Intel-HEX файл, полученный в пункте 4.1. Это можно сделать при помощи соответствующей функции терминала (если таковая имеется), либо написать скрипт, выполняющий данную операцию (например, на языке Python).

При использовании PuTTY (или другого терминала, не имеющего подходящей функции) можно воспользоваться более простым способом: для этого нужно открыть файл в текстовом редакторе Блокнот, скопировать его содержимое полностью и затем вставить его в окне терминала, нажав правую кнопку мыши. В течение всего процесса загрузки на отладочном модуле Салют-ЭЛ24OM1 будет мигать светодиод VD3.

Вставленный текст не отобразится в терминале. Вместо него по окончании загрузки появится запись об объёме полученных данных. Для запуска программы необходимо ввести команду *run*. Команда *run* осуществляет запуск программы по адресу начала внутренней RAM-памяти (0x2000000). Если точка входа в программу не совпадает с этим адресом, то необходимо воспользоваться командой *run* <*address*>, которая выполняет запуск программы по адресу *address*;

Результат выполнения указанных выше действий представлен на Рисунок 4.6.



Рисунок 4.6. Результат загрузки программы через терминал РиТТҮ