



SVP-405. Загрузчик IBL

Руководство пользователя

Версия 1.0



Код документа: UG-SVP-405-IBL
Дата сборки: 27 мая 2015 г.
Листов в документе: 46

© 2015, ООО «Скан Инжиниринг Телеком - СПб»
<http://www.setdsp.ru>

Содержание

Список рисунков	3
Список таблиц	3
Список листингов	3
Список процедур	4
Перечень сокращений и условных обозначений	5
1 Общие сведения	6
2 Сборка образа IBL из исходных кодов	7
2.1 Установка MinGW в Windows системе	7
2.2 Конфигурация окружения сборки	10
2.3 Сборка загрузчика	10
3 Скрипт для записи EEPROM и NOR флеш памяти модуля SVP-405	12
3.1 Структура каталога «Program»	13
3.2 Работа со скриптом записи NOR флеш и EEPROM памяти	15
3.3 Запись образов в NOR флеш память	17
3.4 Запись образов в EEPROM память	22
4 Конфигурация IBL	24
5 Импорт и запуск целевой конфигурации модуля	31
6 Подготовка образов приложений для загрузки на нескольких ядрах процессора	34
6.1 Компоненты MAD Utils	34
6.2 Режимы работы MAD Utils	35
6.3 Работа с MAD Utils в режиме «Prelinker bypass mode»	35
6.4 Конфигурация MAP Tool	36
6.5 Конфигурационный файл развертывания	36
6.6 Запуск MAP Tool	38
6.7 Загрузка образа	38
Приложение А: Примеры работы скрипта записи EEPROM и NOR флеш памяти	39
Приложение Б: Разделение вывода сообщений (CIO) ядер процессоров	42
Приложение В: Выбор режима загрузки IBL	45
Список литературы	46

Список рисунков

2-1	Установка MinGW. Выбор каталога репозитариев	7
2-2	Установка MinGW. Окно согласия с лицензией	8
2-3	Установка MinGW. Выбор пути установки	8
2-4	Установка MinGW. Выбор устанавливаемых компонентов	9
2-5	Приглашение командной строки MinGW Shell	9
2-6	Редактирование файла «setupenvMsys.sh» в редакторе vim	10
3-1	Состояния светодиодов на передней панели модуля SVP-405 во время записи	18
3-2	Организация EEPROM памяти на модуле SVP-405	22
4-1	Подключение к процессору «DSP1_C6678»	24
4-2	Окно загрузки кода на ядро процессора	24
4-3	Внешний вид окна «Debug» после загрузки кода на ядра процессоров	25
4-4	Внешний вид окна «Console» после запуска кода на процессоре	25
4-5	Пункт главного меню для вызова окна управления GEL файлами	26
4-6	Окно управления GEL файлами	26
4-7	Окно управления GEL файлами с загруженным файлом «i2cConfig.gel»	26
4-8	Пункт главного меню «Scripts > SET SVP-405 IBL»	27
4-9	Внешний вид окна «Console» после выполнения записи конфигурации в EEPROM память	27
5-1	Пункт меню для отображения окна целевых конфигураций	31
5-2	Меню импорта целевой конфигурации	31
5-3	Окно выбора файла для импорта целевой конфигурации	32
5-4	Окно выбора способа импорта файла целевой конфигурации	32
5-5	Запуск целевой конфигурации	32
5-6	Список ядер процессоров модуля SVP-405	33
6-1	Схема работы MAD в режиме «Prelinker bypass mode»	35
Б-1	Контекстное меню целевой конфигурации	42
Б-2	Окно настроек целевой конфигурации	42
Б-3	Открытие второго окна «Console»	43
Б-4	Два окна «Console»	43
Б-5	Выбор ядра для отображения вывода в окне «Console»	43
Б-6	Вывод сообщений с двух ядер в два окна «Console»	44

Список таблиц

2-1	Файлы, создаваемые при сборке загрузчика	10
3-1	Параметры командной строки скрипта «svr405_program.js»	15
4-1	Основные конфигурационные параметры файла «i2cConfig.gel»	30
6-1	Параметры конфигурационного файла MAP Tool	36
6-2	Параметры файла конфигурации развертывания	37
В-1	Положение переключателей модуля SVP-405 для установки режимов загрузки IBL	45

Список листингов

3-1	Скрипт «svr405_program.bat»	12
3-2	Скрипт «svr405_program.sh»	12
3-3	Структура каталога «Program»	13
3-4	Вывод скрипта «svr405_program.bat», запущенного без параметров	15
3-5	Вывод в UART загрузки демонстрационного приложения веб-сервера с NOR флеш памяти	19
3-6	Вывод в UART при запуске теста платформы	20
4-1	Фрагмент GEL файла «i2cConfig.gel» с конфигурационными параметрами загрузчика IBL для модуля SVP-405	27
6-1	Пример конфигурационного файла для MAP Tool	36
6-2	Пример файла конфигурации развертывания	37

A-1	Вывод в терминал при запуске команды «svp405_program.bat NOR all»	39
A-2	Вывод в терминал при запуске команды «svp405_program.bat EEPROM all»	41

Список процедур

3-1	Запись образов в NOR флеш память	17
3-2	Запись образа демонстрационного приложения веб-сервера в NOR флеш память и его загрузка	17
3-3	Запись образов в EEPROM память	22
3-4	Запись образа загрузчика IBL в EEPROM память и его загрузка.....	22

Перечень сокращений и условных обозначений

BOOTP	Bootstrap Protocol	30, 38
CCS	Code Composer Studio	6, 7, 10–12, 14, 17, 32, 33, 42, 43
CGT	Code Generation Tools	36
CIO	Console Input/Output	2, 14, 16, 33, 42
COFF	Common Object File Format	30
DDR	Double Data Rate	13, 17, 34, 36
DSO	Dynamic Shared Object	34
DSS	Debug Server Scripting	11, 12
EEPROM	Electrically Erasable Programmable Read-Only Memory	2–4, 6, 7, 10–17, 19, 22–25, 27, 35, 39, 41
ELF	Executable and Linkable Format	30, 34
GEL	General Extension Language	3, 10, 25–27
I²C	Inter-Integrated Circuit	7, 15, 16, 19, 22, 35
IBL	Intermediate Boot Loader	2–4, 6, 7, 9–11, 13, 17, 19, 22–25, 27, 30, 34–36, 38, 45
IP	Internet Protocol	30
JSON	Java Script Object Notation	36, 37
MAC	Media Access Control	30
MAD	Multicore Application Deployment	2, 3, 34–36, 38
MAP	Multiple Application Pre-linker	2, 34–38
MCSDK	MultiCore Software Development Kit	34, 36, 38
NML	No Man's Land	36
NOR	Not OR	2–4, 6, 12–15, 17–19, 23, 36, 38, 39
OFD	Object File Dump	36
ROMFS	Read-Only Memory File System	34–37
ROM	Read-Only Memory	35
TFTP	Trivial File Transfer Protocol	6, 13, 36, 38
TI	Texas Instruments	7
UART	Universal Asynchronous Receiver-Transmitter	3, 13, 17, 19, 20
OC	Операционная Система	6

1 Общие сведения

Загрузчик IBL (Intermediate Boot Loader) позволяет выполнять загрузку приложений на процессоры модуля SVP-405 с NOR флеш памяти или по Ethernet с TFTP сервера в локальной сети. Конкретный режим загрузки выбирается при помощи переключателей на плате модуля SVP-405 отдельно для каждого процессора (см. приложение В).

В данном документе дано описание основных возможностей загрузчика IBL, описан процесс сборки образов IBL из исходных кодов для записи в EEPROM память процессоров модуля SVP-405 (раздел 2), описана процедура записи загрузчика IBL в EEPROM модуля SVP-405 (раздел 3.4), описан процесс конфигурирования уже записанного в EEPROM память загрузчика.

Также, в документе описан процесс записи образов приложений в NOR флеш память модуля SVP-405 для последующий загрузки этих образов загрузчиком IBL.

В данном документе описана работа со средой разработки CCS (Code Composer Studio) версии 5.4.0.00091. Установочный файл среды разработки CCS можно скачать в сети интернет с официального сайта¹, где доступны версии CCS для Windows и Linux систем. На сопроводительном диске к модулю SVP-405 в папке «Install» имеется установочный файл для среды разработки CCS версии 5.4.0.00091 для Windows системы (файл «ccs_setup_5.4.0.00091.exe»).

В данном руководстве предполагается, что среда разработки CCS установлена в папку «C:\ti» и используется компьютер с установленной 64-х разрядной версией ОС Windows 7.

Внимание



Для установки некоторых программ при помощи установочных дистрибутивов, содержащихся на сопроводительном диске к модулю SVP-405, может потребоваться подключение к сети интернет.

¹ http://processors.wiki.ti.com/index.php/Download_CCS

2 Сборка образа IBL из исходных кодов

Для сборки IBL потребуется установленный компилятор для процессоров Texas Instruments серии C6000. Данный компилятор входит в состав среды разработки CCS компании TI.

Результатом сборки IBL из исходных кодов является образ загрузчика готовый к записи в I²C EEPROM модуля SVP-405.

Внимание



Перед выполнением сборки загрузчика, описанной в данном разделе, перепишите с сопроводительного диска к модулю SVP-405 папку «ibl» на жесткий диск компьютера. Далее, предполагается, что все содержимое папки «ibl» с сопроводительного диска переписано в папку «D:/Dev/Modules/SVP-405/IBL».

Для сборки загрузчика IBL в Windows системе, кроме компилятора C6000 процессоров, необходима GNU система сборки MinGW. Скачать последнюю версию MinGW можно на официальном сайте¹.

В данном документе описана работа с MinGW версии 20120426. Установочный дистрибутив MinGW версии 20120426 можно найти на сопроводительном диске к модулю SVP-405 в папке «Install» (файл «mingw-get-inst-20120426.exe»).

2.1 Установка MinGW в Windows системе

В данном разделе описан процесс установки MinGW версии 20120426 с установочного дистрибутива, содержащегося на сопроводительном диске к модулю SVP-405 (файл «Install/mingw-get-inst-20120426.exe»).

При установке MinGW в окне выбора каталога репозитория (рисунок 2-1) необходимо выбрать пункт «Use pre-packaged repository catalogues» (использовать каталог репозитория с заранее собранными пакетами).

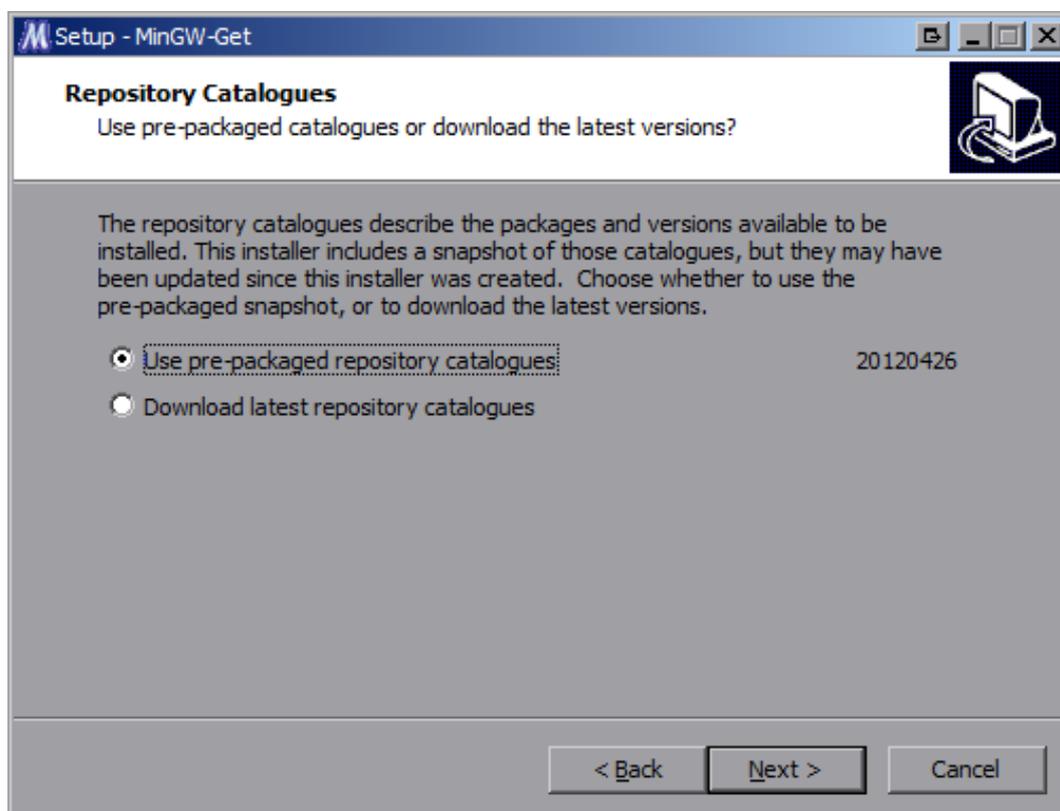


Рисунок 2-1: Установка MinGW. Выбор каталога репозитория

¹ <http://www.mingw.org>

В окне согласия с лицензией (рисунок 2-2), прочитайте текст лицензии, и если вы согласны со всем, что там написано, выберите пункт «I accept the agreement» и нажмите кнопку «Next».

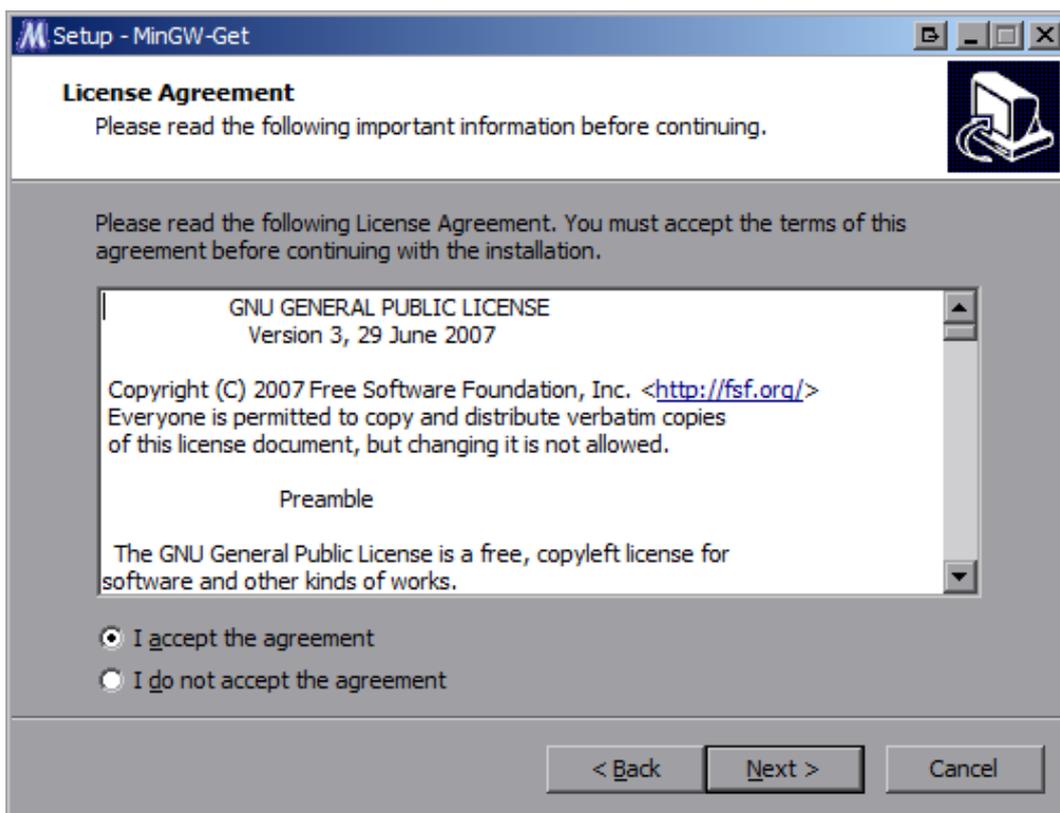


Рисунок 2-2: Установка MinGW. Окно согласия с лицензией

Путь установки MinGW (рисунок 2-3) рекомендуется оставить по умолчанию («C:/MinGW»).

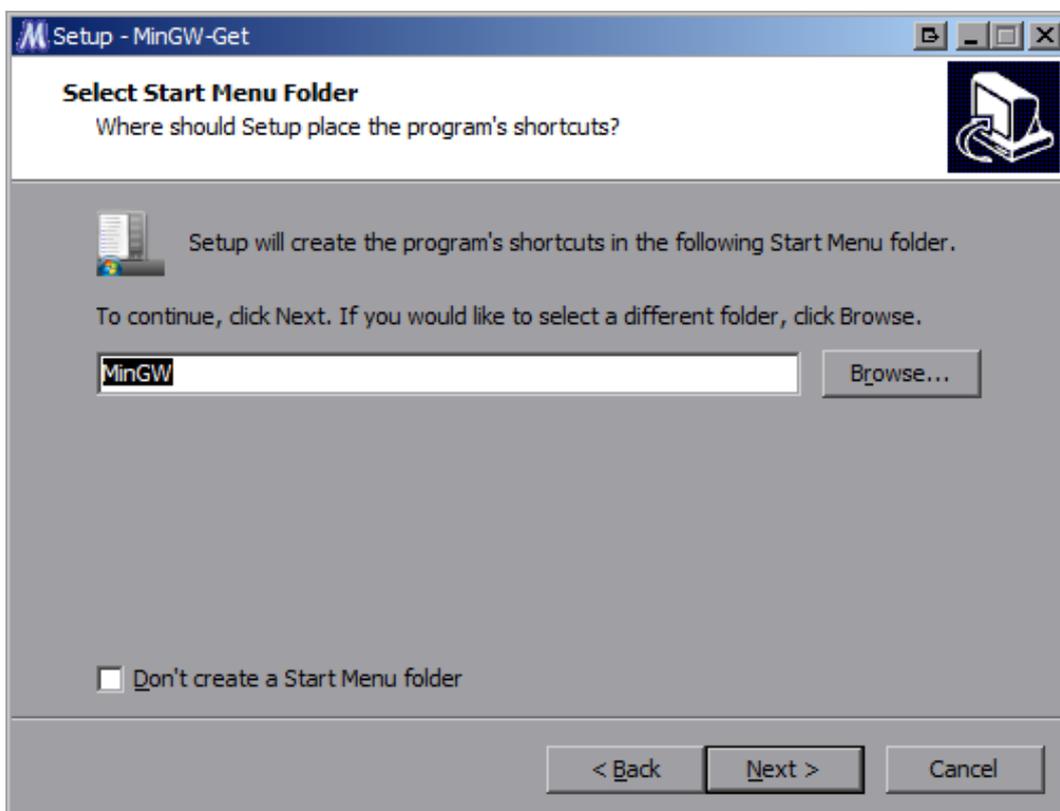


Рисунок 2-3: Установка MinGW. Выбор пути установки

В окне выбора устанавливаемых компонентов (рисунок 2-4) необходимо обязательно отметить следующие компоненты:

- «C Compiler»;
- «MSYS Basic System»;
- «MinGW Developer ToolKit».

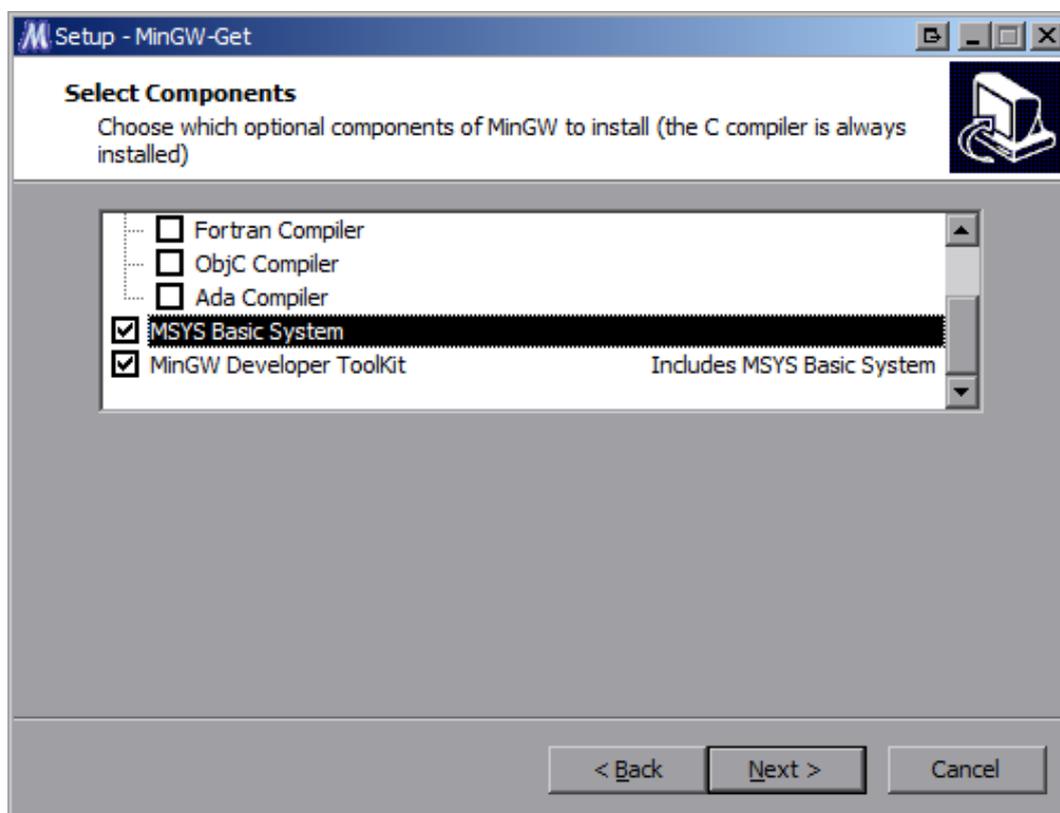


Рисунок 2-4: Установка MinGW. Выбор устанавливаемых компонентов

Остальные компоненты можно отметить на собственное усмотрение. На дальнейший процесс сборки загрузчика IBL их наличие или отсутствие никак не повлияет.

Остальные параметры установки, которые не описаны в данном руководстве, можно оставить в виде, предлагаемом установщиком по умолчанию.

После установки MinGW, через меню «Пуск», запустите «MinGW Shell» (рисунок 2-5). Все последующие действия по сборке IBL будут производиться путем ввода команд в MinGW Shell.

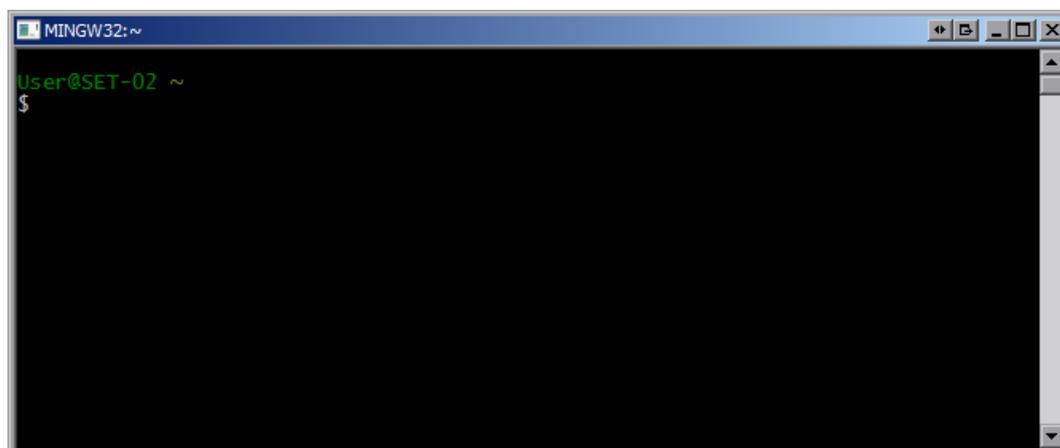
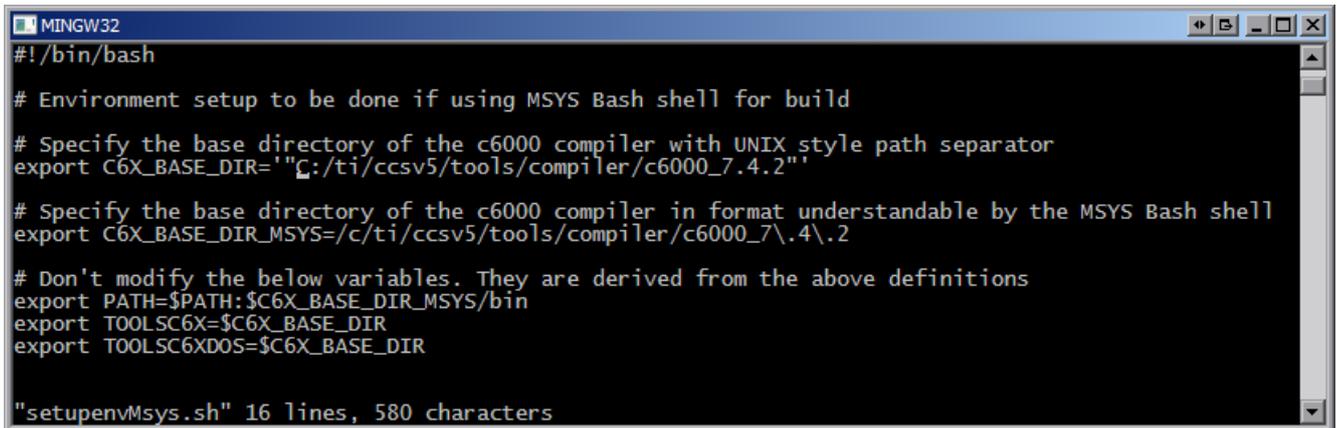


Рисунок 2-5: Приглашение командной строки MinGW Shell

2.2 Конфигурация окружения сборки

В данном разделе предполагается, что исходные коды загрузчика IBL переписаны с сопроводительного диска к модулю SVP-405 в папку «D:/Dev/Modules/SVP-405/IBL», а файлы компилятора для процессоров C6000 серии расположен в папке «C:/ti/ccsv5/tools/compiler/c6000_7.4.2». В том случае, если путь к расположению файлов компилятора для процессоров C6000 отличается, необходимо выполнить соответствующие изменения в скрипте конфигурации окружения сборки «D:/Dev/Modules/SVP-405/IBL/src/make/setupenvMsys.sh». В данном файле необходимо правильно указать путь к файлам компилятора для процессоров C6000 серии.

На рисунке 2-6 приведен снимок экрана MinGW Shell с открытым файлом «setupenvMsys.sh» в редакторе vim. На данном рисунке пути установлены в соответствии с путями указанными выше.



```

MINGW32
#!/bin/bash

# Environment setup to be done if using MSYS Bash shell for build

# Specify the base directory of the c6000 compiler with UNIX style path separator
export C6X_BASE_DIR="C:/ti/ccsv5/tools/compiler/c6000_7.4.2"

# Specify the base directory of the c6000 compiler in format understandable by the MSYS Bash shell
export C6X_BASE_DIR_MSYS=/c/ti/ccsv5/tools/compiler/c6000_7\4\2

# Don't modify the below variables. They are derived from the above definitions
export PATH=$PATH:$C6X_BASE_DIR_MSYS/bin
export TOOLSC6X=$C6X_BASE_DIR
export TOOLSC6XD0S=$C6X_BASE_DIR

"setupenvMsys.sh" 16 lines, 580 characters

```

Рисунок 2-6: Редактирование файла «setupenvMsys.sh» в редакторе vim

Для редактирования файла «setupenvMsys.sh» в редакторе vim выполните в MinGW Shell команду:

```
vim /d/Dev/Modules/SVP-405/IBL/src/make/setupenvMsys.sh
```

В файле «D:/Dev/Modules/SVP-405/IBL/src/make/setupenvMsys.sh» путь к файлам компилятора для процессоров C6000 серии необходимо указать в качестве значений двух переменных — «C6X_BASE_DIR» и «C6X_BASE_DIR_MSYS». При этом, в значении переменной «C6X_BASE_DIR_MSYS» такие символы как пробел, точка, открывающая и закрывающая круглые скобки должны обязательно предваряться (экранироваться) символом обратной косой черты «\».

2.3 Сборка загрузчика

Для запуска процесса сборки загрузчика IBL для модуля SVP-405, выполните в MinGW Shell последовательно следующие команды:

```
cd /d/Dev/Modules/SVP-405/IBL/src/make/
source setupenvMsys.sh
make svp405
```

После выполнения этих команд, будет запущена сборка загрузчика IBL для модуля SVP-405, которая может занять до 30 минут (в зависимости от производительности системы).

В таблице 2-1 перечислены файлы, которые создаются в папке «D:/Dev/Modules/SVP-405/IBL/src/make/bin» после выполнения сборки загрузчика IBL для модуля SVP-405.

Таблица 2-1: Файлы, создаваемые при сборке загрузчика

Файл	Описание
«i2cConfig.gel»	GEL файл конфигурации IBL для CCS (см. раздел 4)
«i2cparam_0x50_svp405_le_0x500.out»	Бинарный образ программы конфигурации IBL для процессора TMS320C6678 модуля SVP-405 (см. раздел 4)
«i2crom_0x50_svp405_le.bin»	Бинарный образ IBL для процессора TMS320C6678 для загрузки в EEPROM память модуля SVP-405 (см. раздел 3.4)

Продолжение таблицы на следующей странице

Продолжение таблицы 2-1

Файл	Описание
«i2crom_0x50_svp405_le.dat»	Бинарный образ IBL для процессора TMS320C6678 модуля SVP-405 в формате CCS.

Для записи в EEPROM память модуля SVP-405 предназначен файл «i2crom_0x50_svp405_le.bin».

Запись образа загрузчика в EEPROM память производится при помощи специального DSS скрипта для среды разработки CCS, работа с которым рассмотрена в разделе 3.

3 Скрипт для записи EEPROM и NOR флеш памяти модуля SVP-405

Для записи образов в EEPROM или NOR флеш память процессоров модуля SVP-405 предназначен специальный DSS скрипт «svp405_program.js», который расположен в папке «Program» на сопроводительном диске к модулю SVP-405.

Перед использованием скрипта «svp405_program.js» для записи EEPROM или NOR флеш памяти, необходимо скопировать с сопроводительного диска к модулю SVP-405 папки «Program», «TargetConfigurations» и «GEL» со всем содержимым в папку «D:/Dev/Modules/SVP-405»¹.

Скрипт «svp405_program.js» требует для своей работы установленную систему разработки CCS. Дистрибутив сетевой установки CCS имеется на сопроводительном диске к модулю SVP-405 (см. раздел 1).

Для запуска скрипта «svp405_program.js» в папке «Program» имеются вспомогательные скрипты запуска для Windows и Linux систем:

- «svp405_program.bat»: скрипт для запуска в Windows системе (см. листинг 3-1);
- «svp405_program.sh»: скрипт для запуска в Linux системе (см. листинг 3-2).

Листинг 3-1: Скрипт «svp405_program.bat»

```
1 @echo off
2 set PROGRAM_SVP_405_TARGET_CONFIG_FILE=../TargetConfigurations/SVP-405-LAN560v2-SD.ccxml
3 set DSS_SCRIPT_DIR="C:\ti\ccsv5\ccs_base\scripting\bin"
4 call %DSS_SCRIPT_DIR%\dss.bat svp405_program.js %*
```

Листинг 3-2: Скрипт «svp405_program.sh»

```
1 #!/bin/sh
2 export PROGRAM_SVP_405_TARGET_CONFIG_FILE=../TargetConfigurations/SVP-405-LAN560v2-SD.ccxml
3 export DSS_SCRIPT_DIR=~/.ti/ccsv5/ccs_base/scripting/bin
4 $DSS_SCRIPT_DIR/dss.sh svp405_program.js $@
```

Как видно из листингов 3-1 и 3-2, в скриптах задаются значения для двух переменных окружения:

- «PROGRAM_SVP_405_TARGET_CONFIG_FILE»: путь к файлу описания целевой конфигурации модуля SVP-405;
- «DSS_SCRIPT_DIR»: путь к бинарным файлам DSS среды разработки CCS.

Важная информация



Только для Windows: В скрипте «svp405_program.bat» значение переменной «DSS_SCRIPT_DIR» обязательно должно быть записано в кавычках, а значение переменной «PROGRAM_SVP_405_TARGET_CONFIG_FILE», наоборот, не должно содержать кавычек.

После определения значений для переменных окружения, в скрипте происходит вызов скрипта кроссплатформенного DSS скрипта «svp405_program.js», с передачей ему аргументов командной строки.

Примечание

В данном документе рассматривается работа со скриптом «svp405_program.bat» для Windows системы. Для выполнения аналогичных действий в Linux системе, достаточно заменить вызовы скрипта «svp405_program.bat» на вызовы скрипта «svp405_program.sh».

В разделе 3.1 описывается структура каталога «Program», в котором работает скрипт «svp405_program.js». Далее, в разделе 3.2, описывается непосредственная работа со скриптом.

¹ Конечная папка может отличаться. В данном документе предполагается, что копирование выполнено именно в эту папку

3.1 Структура каталога «Program»

В листинге 3-3 представлена структура каталога «Program» с сопроводительного диска модуля SVP-405, в котором располагаются все необходимые файлы для выполнения записи в EEPROM или NOR флеш память процессоров модуля SVP-405.

Листинг 3-3: Структура каталога «Program»

```
Program
+- bin
| +- ibl
| | +- i2crom_0x50_svp405_le.bin
| | +- i2cConfig.ge1
| | +- i2cparam_0x50_svp405_le_0x500.out
| |
| +- platform_test
| | +- platform_test_svp405.bin
| |
| +- webserver
| | +- webserv_svp405.bin
| |
| +- eeprom_0x50_c6678.bin
| +- nor_c6678.bin
|
+- logs
|
+- writers
| +- eepromwriter_c6678.out
| +- norwriter_c6678.out
|
+- svp405_program.bat
+- svp405_program.sh
+- svp405_program.js
```

Кроме файлов скриптов «svp405_program.bat», «svp405_program.sh» и «svp405_program.js», в каталоге «Program» имеется два вложенных каталога:

В каталоге «bin» расположены файлы образов готовые для записи в EEPROM и NOR флеш память процессоров модуля SVP-405. Пользовательские файлы, подготовленные для записи в EEPROM или NOR флеш память, должны быть помещены в каталог «bin».

Непосредственно в каталоге «bin» расположены два файла, которые используются скриптом «svp405_program.js» для записи по умолчанию:

- «eeprom_0x50_c6678.bin»: образ для записи в EEPROM память процессора TMS320C6678. По умолчанию, данный файлы представляет собой собранный образ загрузчика IBL, который можно получить путем выполнения шагов, описанных в разделе 2 данного документа.
- «nor_c6678.bin»: образ для записи в NOR флеш память процессора TMS320C6678. По умолчанию, данный файлы представляет собой собранный образ демонстрационного приложения веб-сервера. Подробное описание по сборке и запуску демонстрационного приложения веб-сервера дано в документе [1].

В каталогах «bin/webserver» и «bin/platform_test» расположены образы приложений, готовые для записи в NOR флеш память процессоров. Данные образы, также могут применяться для загрузки с TFTP сервера.

В каталоге «bin/platform_test» размещены образы приложения теста платформы (файл «platform_test_svp405.bin»), которые выполняют следующие тесты: UART, NOR флеш память, EEPROM память, DDR память и тест светодиодов на передней панели.

Важная информация



При запуске приложения теста платформы следует иметь в виду, что если во время выполнения тестов EEPROM памяти или NOR флеш памяти отключить питание модуля, содержимое тестируемого типа памяти может быть повреждено.

В каталоге «bin/webserver» находится образ собранного приложения демонстрационного веб-сервера (файл «webserv_svp405.bin»). Подробное описание данного приложения дано в документе [1].

Исходные коды приложения теста платформы находятся в проекте «PlatformTest_C6678» рабочего пространства CCS. Папку с рабочим пространством можно найти на сопроводительном диске к модулю SVP-405 в папке «CCS_Workspace».

В каталоге «writers» расположены файлы программ, для осуществления записи в EEPROM и NOR флеш память процессоров. Данные файлы получены путем сборки проектов «EEPROM_Writer» (файл «eepromwriter_c6678.out») и NOR_Writer» (файл «norwriter_c6678.out») рабочего пространства CCS для модуля SVP-405. Папку с рабочим пространством со всеми проектами можно найти на сопроводительном диске в папке «CCS_Workspace».

В каталоге «logs» сохраняются логи вывода CIO с процессоров в случае запуска скрипта с параметром log (см. таблицу 3-1).

3.2 Работа со скриптом записи NOR флеш и EEPROM памяти

Работа со скриптом записи EEPROM и NOR флеш памяти модуля SVP-405 осуществляется из командной строки. Выбор того или иного режима работы скрипта осуществляется при помощи параметров командной строки, передаваемых скрипту при запуске.

Если запустить скрипт «svp405_program.bat» без параметров, будет выдана краткая справка по возможным параметрами командной строки (см. листинг 3-4).

Листинг 3-4: Вывод скрипта «svp405_program.bat», запущенного без параметров

```

1 D:\Dev\Modules\SVP-405\Program>svp405_program.bat
2
3 Syntax: svp405_program.js <NOR|EEPROM> <options>
4
5 Mode:
6   NOR   : program NOR flash memory
7   EEPROM : program I2C EEPROM memory
8
9 Options:
10  dsp<n>      : program DSP<n>
11  all         : program all DSP's (DSP1 and DSP2)
12  image=<image> : image file for all DSP's
13  image_dsp<n>=<image> : individual image for DSP<n>
14                  (image must be in 'bin' folder)
15  bus_dsp<n>=<address> : I2C bus EEPROM address for DSP<n>
16                  Address must be a decimal number
17                  Only for EEPROM mode. Default is 80 (0x50)
18  bus=<address> : I2C bus EEPROM address for all DSP's
19                  Address must be a decimal number
20                  Only for EEPROM mode. Default is 80 (0x50)
21  log         : Enable CIO logging
22
23 D:\Dev\Modules\SVP-465\Program>
```

Таким образом, первый параметр командной строки задает режим работы скрипта и может принимать одно из значений:

- NOR: режим записи NOR флеш памяти;
- EEPROM: режим записи EEPROM памяти.

Количество остальных параметров командной строки может варьироваться. Доступные параметры приведены в таблице 3-1.

Таблица 3-1: Параметры командной строки скрипта «svp405_program.js»

Параметр	Описание
dsp<n>	Включает запись образа в память для процессора с номером <n>. Значение <n> может быть от 1 до 2: <ul style="list-style-type: none"> • 1: DSP1_TMS320C6678; • 2: DSP2_TMS320C6678.
all	Включает запись образа в память сразу для 2-х процессоров модуля SVP-405. Эквивалентно заданию параметров командной строки dsp<n> для всех <n>.
image=<image>	Параметр задает имя файла (<image>) образа для записи на обоих процессорах. Может использоваться в том случае, когда на все процессоры необходимо записать один и тот же образ.
image_dsp<n>=<image>	Параметр позволяет указать имя файла (<image>) образа для конкретного процессора <n>.
bus_dsp<n>=<address>	Параметр позволяет указать адрес (<address>) I ² C шины EEPROM памяти для записи образа для конкретного процессора <n>. Значение данного параметра учитывается только в режиме EEPROM. По умолчанию, если параметр не задан, используется адрес шины 80 (0x50).

Продолжение таблицы на следующей странице

Продолжение таблицы 3-1

Параметр	Описание
bus=<address>	Параметр позволяет указать адрес (<address>) I ² C шины EEPROM памяти для записи образа сразу для всех 2-х процессоров. Значение данного параметра учитывается только в режиме EEPROM. По умолчанию, если параметр не задан, используется адрес шины 80 (0x50).
log	Параметр включает запись вывода CIO со всех процессоров модуля SVP-405 в файлы, которые будут помещены в каталог «logs». Создаваемые файлы лога имеют формат «SVP-405-DSP<n>-<time>-cio.txt», где <n> — номер процессора модуля SVP-405 (от 1 до 4), <time> — время запуска скрипта «svp405-program.js». По умолчанию, запись вывода CIO в файлы отключена.

Важная информация

Значения адреса шины для параметров bus_dsp<n> и bus обязательно должны указываться в десятичном виде. Соответственно, для адреса шины 0x50 (значение по умолчанию) необходимо указывать значение 80, а для адреса шины 0x51 — значение 81.

3.3 Запись образов в NOR флеш память

Для записи образа в NOR флеш память модуля SVP-405 выполните шаги процедуры 3-1.

Внимание



Запись в NOR флеш память рекомендуется выполнять при включенном режиме «NOBOOT» на соответствующем процессоре (см. приложение В).

Процедура 3-1. Запись образов в NOR флеш память

1. Выполните подготовку образа(ов).

Внимание



Следует помнить, что объем NOR флеш памяти, установленной на модуле SVP-405, составляет 16 Мбайт на каждый процессор. Таким образом, ограничение на размер образов, записываемых в NOR флеш память, составляет 16 Мбайт.

2. Скопируйте подготовленный(е) для записи образ(ы) в каталог «D:/Dev/Modules/SVP-405/Program/bin».
3. Перейдите в каталог «D:/Dev/Modules/SVP-405/Program».
4. Запустите скрипт «svp405_program.bat» в режиме NOR с требуемыми параметрами командной строки (см. таблицу 3-1 раздела 3.2).

В процедуре 3-1 представлены общие шаги, необходимые для записи образа в NOR флеш память процессоров модуля SVP-405.

В качестве примера, в процедуре 3-2 подробно описаны шаги, необходимые для выполнения записи в NOR флеш память всех четырех процессоров модуля SVP-405 образа приложения демонстрационного веб-сервера [1], начиная от сборки приложения, и заканчивая его загрузкой с NOR флеш памяти при помощи загрузчика IBL.

Важная информация



Записываемые в NOR флеш память файлы образов обязательно должны иметь расширение «.bin».

При сборке приложений в CCS, по умолчанию, файлы имеют расширение «.out». Перед выполнением записи, необходимо переименовать файлы таким образом, чтобы они имели расширение «.bin».

Процедура 3-2. Запись образа демонстрационного приложения веб-сервера в NOR флеш память и его загрузка

1. Выполните сборку приложения веб-сервера для конфигураций сборки «Debug» (процесс сборки приложения веб-сервера подробно описан в документе [1]).

В результате сборки приложений веб-сервера должен быть получен файл «D:/Dev/Modules/SVP-405/CCS_Workspace/WebServer_C6678/Debug/websrv_svp405.out»:

Примечание

Модуль SVP-405 поставляется с записанным приложением теста платформы, в котором выполняются последовательно тест светодиодов на передней панели модуля, частичный тест EEPROM памяти, частичный тест NOR флеш памяти и полный тест внешней DDR памяти. Кроме того, записанный в NOR флеш память тест выводит различную информацию о модуле. Пример вывода в UART теста платформы приведен в листинге 3-6.

2. Скопируйте файл, полученный при выполнении шага 1 в папку «D:/Dev/Modules/SVP-405/Program/bin». Для этого выполните в терминале последовательно

следующие команды (при копировании, также выполняется изменение расширения «.out» на «.bin», что необходимо для правильной записи):

```
d:
cd D:\Dev\Modules\SVP-405
copy CCS_Workspace\WebServer_C6678\Debug\websrv_svp405.out Program\bin\websrv_svp405.bin
```

- Установите для всех процессоров модуля SVP-405 режим «NOBOOT» в соответствии с данными таблицы В-1 приложения В.
- Подключите отладчик к модулю SVP-405 и к компьютеру (предполагается использование отладчика Spectrum Digital XDS560v2 STM LAN).
- Включите модуль SVP-405.
- Перейдите в каталог «D:/Dev/Modules/SVP-405/Program». Для этого необходимо выполнить в терминале команды:

```
d:
cd D:\Dev\Modules\SVP-405\Program
```

- Запустите скрипт «svp405_program.bat» выполнив в терминале команду:

```
svp405_program.bat NOR all image=websrv_svp405.bin
```

Будет запущен процесс записи образов в NOR флеш память процессоров модуля SVP-405. При этом, вывод в терминал должен быть аналогичен тому, что приведен в листинге А-1 приложения А. Весь процесс записи должен занимать не более 5 минут.

После вывода в терминал сообщения «Executing writers on DSP(s)...» на процессорах модуля SVP-405 начинается процесс записи данных образов в NOR флеш память. При этом, светодиоды в соответствующей группе («DSP1» и/или «DSP2») на передней панели модуля SVP-405 будут иметь определенное состояние в зависимости от хода процесса записи (см. рисунок 3-1).

Идет процесс записи или верификации данных:



При возникновении ошибки записи или верификации:



Запись успешно завершена:



Рисунок 3-1: Состояния светодиодов на передней панели модуля SVP-405 во время записи

При нормальном процессе записи и верификации записанных данных, должен гореть только зеленый светодиод «LD1». Желтый светодиод «LD2» гореть не должен.

Если во время записи или верификации происходит ошибка, то загорается желтый светодиод «LD2», зеленый светодиод «LD1» гаснет (см. рисунок 3-1). При этом, работа программы записи прекращается.

В случае успешного завершения записи и верификации записанных данных, загорятся оба светодиода «LD1» и «LD2».

8. Выключите модуль SVP-405.
9. Если в EEPROM память процессоров модуля SVP-405 еще не записан образ загрузчика IBL, выполните шаги процедуры 3-4.

Примечание

Модуль SVP-405 поставляется с уже записанным образом загрузчика IBL в EEPROM память (адрес I²C шины 0x50) обоих процессоров модуля. Поэтому, если в EEPROM память (адрес I²C шины 0x50) не производилась запись каких либо других образов, выполнять запись загрузчика не требуется.

10. Установите для всех процессоров модуля SVP-405 режим «NOR» в соответствии с данными таблицы В-1 приложения В.
11. Включите модуль SVP-405.

После выполнения всех шагов процедуры 3-2, вывод в UART с процессоров модуля SVP-405 при его включении должен выглядеть подобно приведенному в листинге 3-5. В листинге 3-5 приведен вывод в UART загрузки демонстрационного приложения веб-сервера на процессоре «DSP1_C6678». Вывод в UART с других процессоров должен выглядеть аналогично.

Листинг 3-5: Вывод в UART загрузки демонстрационного приложения веб-сервера с NOR флеш памяти

```

1 IBL INIT
2 I2C boot
3 EDC enabled
4 -- Device is C6678
5 -- Power down PCIE...
6 -- Power up PCIE...
7 -- Setting PCIESEN = 0x01, PCIESSMODE = 0x00 (EP)...
8 -- Program PLL settings and enable PLL
9 -- Setting multiplier to x10 (REFCLK = 250 MHz)...
10 -- Wait for PCIE PLL lock...
11 -- PLL locked
12 -- Disable link training...
13 -- Configuring PCIE registers...
14 * Register SERDES_STS = 0x00000005
15   - PLL locked
16   - Lane0 is ok
17   - Lane1 is ok
18 * Register PCS_STATUS = 0x00001300
19 * Register DEBUG0     = 0x00003900
20 * Register DEBUG1     = 0x08200000
21 * Register LINK_CTRL2 = 0x00010002
22 * Register PL_GEN2    = 0x0000010F
23 -- Enable link training...
24 -- Waiting for link up...
25 -- Link is up. Booting...
26 * Register SERDES_STS = 0x00000001
27   - PLL locked
28   - Lane0 is ok
29   - Lane1 is ok
30 * Register PCS_STATUS = 0x00001333
31 * Register DEBUG0     = 0x03112011
32 * Register DEBUG1     = 0x08000010
33 * Register LINK_CTRL2 = 0x00000002
34 * Register PL_GEN2    = 0x0000010F
35 Boot table processor executed
36
37 IBL version: 1.0.0.16-4
38 Device is SVP-405
39 c4:ed:ba:98:04:1e
40 IBL: PLL and DDR Initialization Complete
41 IBL Result code 00
42 Multiboot enabled
43 SPI initialized
44 IBL: Booting from NOR
45 Booting from NOR flash from address
46   = 0x8157c8e0
47 Platform initialized
48 Start BIOS 6
49 QMSS successfully initialized

```

```

50 CPPI successfully initialized
51 PA successfully initialized
52 HUA version 2.00.00.04
53 Setting hostname to setdemo
54 MAC Address: C4-ED-BA-98-04-7A
55 EMAC ports count: 1
56 Configuring DHCP client
57 PASS successfully initialized
58 Ethernet subsystem successfully initialized
59 Ethernet eventId : 48 and vectId (Interrupt) : 7
60 Registration of the EMAC Successful, waiting for link up ..
61 Service Status: DHCPC      : Enabled      :           : 000
62 Service Status: THHTTP     : Enabled      :           : 000
63 Service Status: DHCPC      : Enabled      : Running   : 000
64 Network Added: If-1:192.168.2.85
65 Service Status: DHCPC      : Enabled      : Running   : 017

```

В листинге 3-6 приведен пример вывода в UART запуска приложения теста платформы модуля SVP-405 (файл «Program/bin/platform_test_svp405.bin»).

Листинг 3-6: Вывод в UART при запуске теста платформы

```

1 Platform test for module
2
3 / _ _ _ \ \ / / _ _ _ \ | | | | / _ _ \ | _ _ |
4 | ( _ _ \ \ / / | | ) | _ _ | | | | | | | | |
5 \ _ _ \ \ / / | | _ _ / _ _ | | | | | | | \
6 _ _ ) | \ / | | | | | | | | | | | | |
7 | _ _ / \ / | | | | | | | | | | | | |
8
9 p_info->version = 2.00.00.14
10 p_info->board_name = SVP-405
11 p_info->board_id = SVP405
12 p_info->serial_nbr = 4151001
13 p_info->slot_n = 2
14 p_info->dsp_n = DSP2
15 p_info->cpu.core_count = 8
16 p_info->cpu.name = TMS320C6678
17 p_info->cpu.id = 21
18 p_info->cpu.revision_id = 0
19 p_info->cpu.endian = 1
20 p_info->frequency = 999 MHz
21 p_info->led[PLATFORM_USER_LED_CLASS].count = 2
22 p_info->emac.port_count = 1
23 -> EMAC port 0
24 MAC address = c4:ed:ba:98:04:1e
25
26 NOR device info:
27 p_device->device_id = 0xbb18
28 p_device->manufacturer_id = 0x20
29 p_device->width = 8
30 p_device->block_count = 256
31 p_device->page_count = 256
32 p_device->page_size = 256
33 p_device->spare_size = 0
34 p_device->handle = 0xbb18
35 p_device->flags = 0
36 p_device->bboffset = 0
37
38 EEPROM device (I2C address 0x50) info:
39 p_device->device_id = 0x50
40 p_device->manufacturer_id = 0x1
41 p_device->width = 8
42 p_device->block_count = 1
43 p_device->page_count = 1
44 p_device->page_size = 65536
45 p_device->spare_size = 0
46 p_device->handle = 0x50
47 p_device->flags = 0
48 p_device->bboffset = 0
49
50 EEPROM device (I2C address 0x51) info:
51 p_device->device_id = 0x51
52 p_device->manufacturer_id = 0x1
53 p_device->width = 8
54 p_device->block_count = 1
55 p_device->page_count = 1

```

```
56 p_device->page_size          = 65536
57 p_device->spare_size         = 0
58 p_device->handle              = 0x51
59 p_device->flags               = 0
60 p_device->bboffset           = 0
61
62 Current core id is 0
63
64 -----
65 LED test
66 -----
67 Testing USER class leds (2 leds)...
68 - LED 0 ON
69 - LED 0 OFF
70 - LED 0 ON
71 - LED 0 OFF
72 - LED 1 ON
73 - LED 1 OFF
74 - LED 1 ON
75 - LED 1 OFF
76 LED test complete
77
78 -----
79 EEPROM test
80 -----
81 test_eeprom: Partial EEPROM test mode
82 test_eeprom: 0x50: Saving EEPROM contents...
83 test_eeprom: 0x50: Writing 1024 bytes...
84 test_eeprom: 0x50: Read and check data...
85 test_eeprom: 0x50: Restoring EEPROM contents...
86 test_eeprom: 0x50: Test passed
87 test_eeprom: 0x51: Saving EEPROM contents...
88 test_eeprom: 0x51: Writing 1024 bytes...
89 test_eeprom: 0x51: Read and check data...
90 test_eeprom: 0x51: Restoring EEPROM contents...
91 test_eeprom: 0x51: Test passed
92 EEPROM test complete
93
94 -----
95 NOR test
96 -----
97 test_nor: Partial NOR test mode
98 test_nor: NOR flash sector size is 65536 bytes
99 test_nor: Test block size is 65536 bytes
100 test_nor: Saving original data from NOR...
101 test_nor: Writing pattern data to NOR...
102 test_nor: Reading pattern data from NOR...
103 test_nor: Comparing data...
104 test_nor: Data is equal. Test passed.
105 test_nor: Write back original data to NOR...
106 test_nor: Test passed
107 NOR test complete
108
109 -----
110 External memory test
111 -----
112 Memory test: start = 0x8000000, end = 0xffffffff
113 Memory test: Write a pattern...
114 Memory test: Read and check pattern...
115 Memory test: Write a pattern for complementary values...
116 Memory test: Read and check pattern...
117 External memory test passed
118 External memory test complete
119
120 Test completed
```

3.4 Запись образов в EEPROM память

Для записи образа в EEPROM память модуля SVP-405 выполните шаги процедуры 3-3.

Внимание



Запись в EEPROM память рекомендуется выполнять при включенном режиме «NOBOOT» на соответствующем процессоре (см. приложение В).

Процедура 3-3. Запись образов в EEPROM память

1. Выполните подготовку образа(ов).

Внимание



Следует помнить, что общий объем EEPROM памяти, установленной на модуле SVP-405, составляет 128 Кбайт на каждый процессор. EEPROM память, установленная на модуле SVP-405, разделена на две части (каждая объемом по 64 Кбайта): адресу 0x50 шины I²C соответствует первая часть, а адресу 0x51 шины I²C — вторая часть (см. рисунок 3-2). При этом, загрузка модуля SVP-405 возможна только из первой части EEPROM с адресом шины I²C 0x50. Таким образом, ограничение на размер образов, записываемых в EEPROM память, составляет 64 Кбайт.

2. Скопируйте подготовленный(е) для записи образ(ы) в каталог «D:/Dev/Modules/SVP-405/Program/bin».
3. Перейдите в каталог «D:/Dev/Modules/SVP-405/Program».
4. Запустите скрипт «svp405_program.bat» в режиме EEPROM с требуемыми параметрами командной строки (см. таблицу 3-1 раздела 3.2).

EEPROM (128 Кбайт)



BOOT — область загрузчика IBL; USER — область пользовательских данных

Рисунок 3-2: Организация EEPROM памяти на модуле SVP-405

В процедуре 3-3 представлены общие шаги, необходимые для записи образа в EEPROM память процессоров модуля SVP-405.

Важная информация



Записываемые в EEPROM память файлы образов обязательно должны иметь расширение «.bin».

В качестве примера, в процедуре 3-4 подробно описаны шаги, необходимые для выполнения записи в EEPROM память всех четырех процессоров модуля SVP-405 образа загрузчика IBL.

Процедура 3-4. Запись образа загрузчика IBL в EEPROM память и его загрузка

1. Выполните сборку загрузчика IBL (процесс сборки IBL подробно описан в разделе 2).
В результате сборки загрузчика IBL должен быть получен файл «D:/Dev/Modules/SVP-405/ibl/src/make/bin/i2crom_0x50_svp405_le.bin».

Примечание

Модуль SVP-405 поставляется с уже записанным образом загрузчика IBL в EEPROM память обоих процессоров. Образ, записанный в EEPROM память при изготовлении модуля SVP-405 содержится в файле «i2crom_0x50_svp405_le.bin». Данный файл расположен в каталоге «D:/Dev/Modules/SVP-405/Program/bin/ibl».

2. Скопируйте файл, полученный при выполнении шага 1 в папку «D:/Dev/Modules/SVP-405/Program/bin». Для этого выполните в терминале последовательно следующие команды:

```
d:  
cd D:\Dev\Modules\SVP-405  
copy ibl\src\make\bin\i2crom_0x50_svp405_le.bin Program\bin
```

3. Установите для всех процессоров модуля SVP-405 режим «NOBOOT» в соответствии с данными таблицы B-1 приложения B.
4. Подключите отладчик к модулю SVP-405 и к компьютеру (предполагается использование отладчика Spectrum Digital XDS560v2 STM LAN).
5. Включите модуль SVP-405.
6. Перейдите в каталог «D:/Dev/Modules/SVP-405/Program». Для этого необходимо выполнить в терминале команды:

```
d:  
cd D:\Dev\Modules\SVP-405\Program
```

7. Запустите скрипт «svp405_program.bat» выполнив в терминале команду:

```
svp405_program.bat EEPROM all image=i2crom_0x50_svp405_le.bin
```

Будет запущен процесс записи образа в EEPROM память процессоров модуля SVP-405. При этом, вывод в терминал должен быть аналогичен тому, что приведен в листинге A-2 приложения A. Весь процесс записи должен занимать не более 5 минут.

Состояния светодиодов «DSP» на передней панели модуля SVP-405 во время записи EEPROM памяти соответствуют состояниям светодиодов во время записи NOR флеш памяти, которые описаны в шаге 7 процедуры 3-2.

8. Выключите модуль SVP-405.
9. Установите для всех процессоров модуля SVP-405 требуемый режим загрузки («NOR», «TFTP» или «PCI-E») в соответствии с данными таблицы B-1 приложения B.
10. Включите модуль SVP-405.

4 Конфигурация IBL

Записанный в EEPROM память загрузчик IBL хранит блок с конфигурационными параметрами в этой же EEPROM памяти с определенным смещением. Конфигурация IBL заключается в изменении данных блока с конфигурационными параметрами в EEPROM памяти.

Для облегчения процесса изменения данных в блоке конфигурационных данных загрузчика, при сборке IBL (см. раздел 2), создается образ специальной программы «i2cparam_0x50_svp405_le_0x500.out» (см. таблицу 2-1), которая предназначена для загрузки на процессорах модуля SVP-405 для конфигурации загрузчика.

Примечание

Значение «0x500» в имени файлов «i2cparam_0x50_svp405_le_0x500.out» определяет величину смещения расположения блока конфигурационных параметров загрузчика IBL относительно начала EEPROM памяти.

Для запуска программы конфигурации загрузчика IBL необходимо, в первую очередь, запустить целевую конфигурацию модуля SVP-405, как показано в разделе 5.

После запуска целевой конфигурации, как показано в разделе 5, выполните подключение к требуемому процессору (в данном примере рассмотрена конфигурация загрузчика на процессоре «DSP1_C6678»). Для этого, нажмите правой кнопкой мыши на названии ядра «DSP1_C6678_0» и выберите пункт меню «Connect Target» (см. рисунок 4-1).

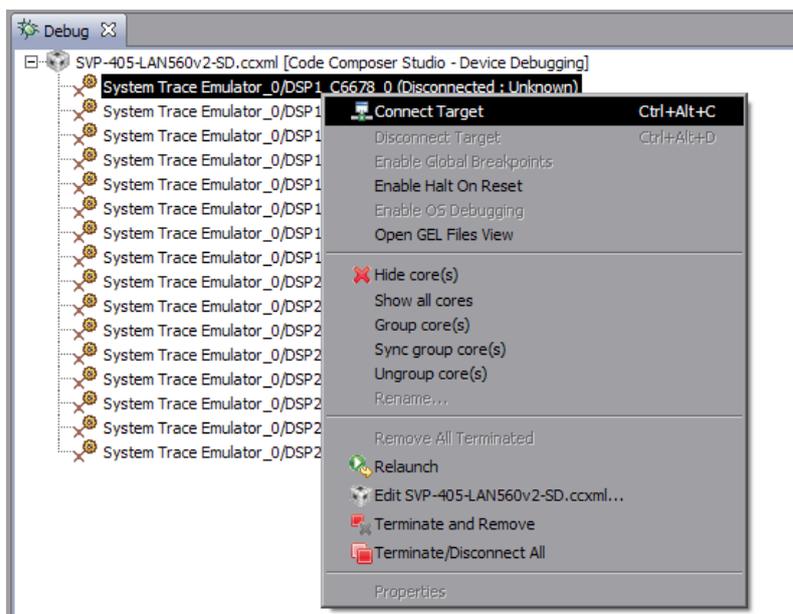


Рисунок 4-1: Подключение к процессору «DSP1_C6678»

Выберите ядро «DSP1_C6678_0» в окне «Debug» (щелкните левой кнопкой мыши по названию ядра). Выберите пункт главного меню «Run > Load > Load Program...». В открывшемся окне (рисунок 4-2) нажмите на кнопку «Browse».

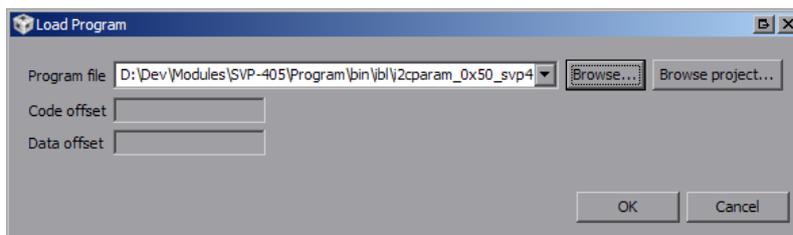


Рисунок 4-2: Окно загрузки кода на ядро процессора

В открывшемся окне выбора файлов необходимо выбрать файл «i2cparam_0x50_svp405_le_0x500.out» (см. таблицу 2-1) и нажать на кнопку «ОК». Данный файл также можно найти в папке «D:/Dev/Modules/SVP-405/Program/bin/ibl».

После загрузки кода на ядро процессора, окно «Debug» должно выглядеть, как показано на рисунке 4-3.

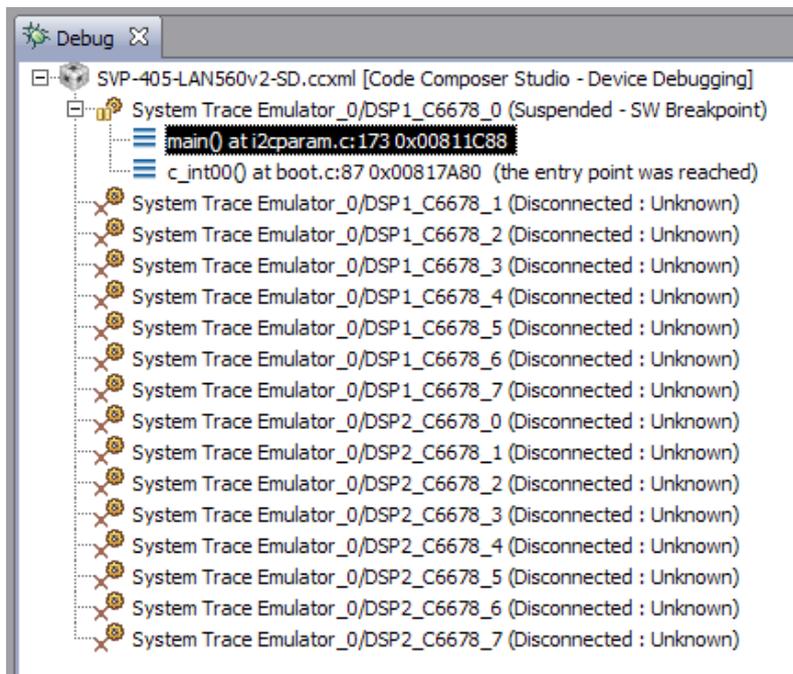


Рисунок 4-3: Внешний вид окна «Debug» после загрузки кода на ядра процессоров

Выделите в окне «Debug» ядро «DSP1_C6678_0» щелкнув по нему левой кнопкой мыши. и запустите выполнение кода, нажав на кнопку  («Resume»). Кнопка  («Resume») находится в верхней части окна «Debug» (см. рисунок 4-3).

После запуска приложения конфигурации загрузчика, в окно «Console» будет выведено сообщение «Run the GEL for the device to be configured, press return to programm the I2C» (см. рисунок 4-4).

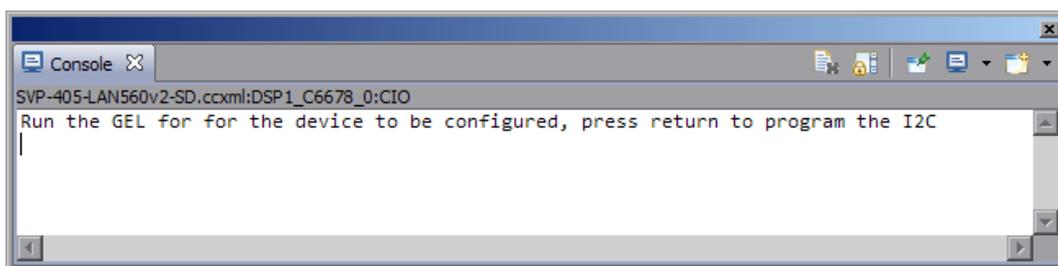


Рисунок 4-4: Внешний вид окна «Console» после запуска кода на процессоре

Данное сообщение означает, что программа ожидает пока будет произведена конфигурация параметров загрузчика IBL и предлагает нажать на клавишу «Enter» для записи выполненной конфигурации в EEPROM память.

Конфигурация параметров загрузчика осуществляется при помощи специального GEL файла «i2cConfig.gel», который можно найти в папке «D:/Dev/Modules/SVP-405/Program/bin/ibl».

Для вызова окна управления GEL файлами, выберите пункт главного меню «Tools > GEL Files», как показано на рисунке 4-5.

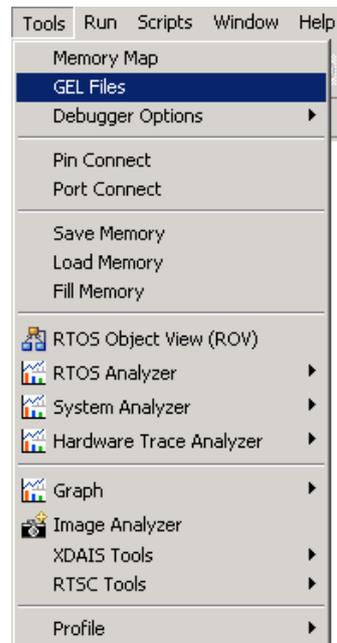


Рисунок 4-5: Пункт главного меню для вызова окна управления GEL файлами

В открывшемся окне «GEL Files» (рисунок 4-6) нажмите правой кнопкой мыши в области со списком загруженных GEL файлов (изначально там должен быть только один файл «svr405.gel») и выберите пункт меню «Load GEL...».

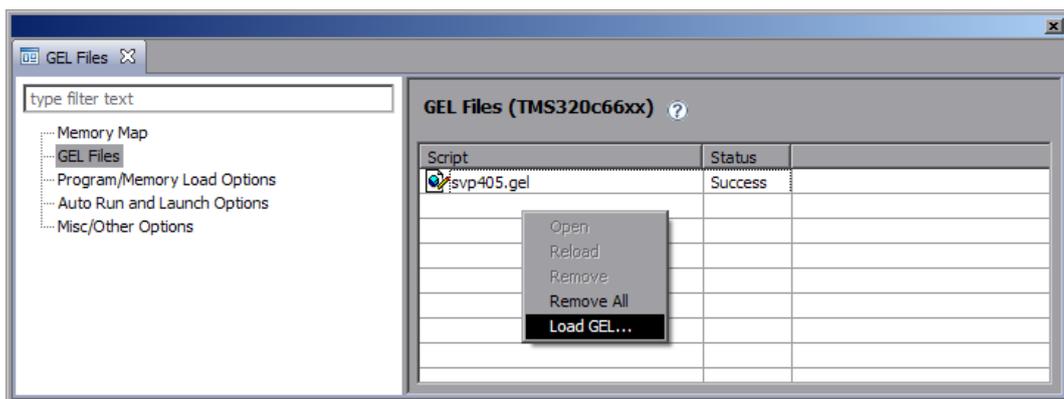


Рисунок 4-6: Окно управления GEL файлами

В открывшемся окне необходимо выбрать файл «D:/Dev/Modules/SVP-405/Program/bin/ibl/i2cConfig.gel». После чего, в списке загруженных файлов окна «GEL Files» должен появиться файл «i2cConfig.gel», как показано на рисунке 4-7.

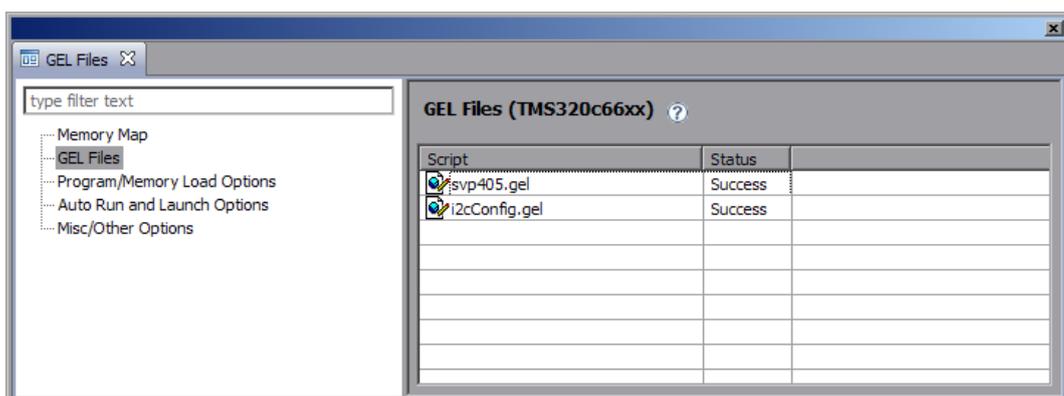


Рисунок 4-7: Окно управления GEL файлами с загруженным файлом «i2cConfig.gel»

После выполнения данных действий, появится пункт меню «Scripts > SET SVP-405 IBL > setConfig_svp405_main» при выборе которого будет применена конфигурация загрузчика (см. рисунок 4-8).

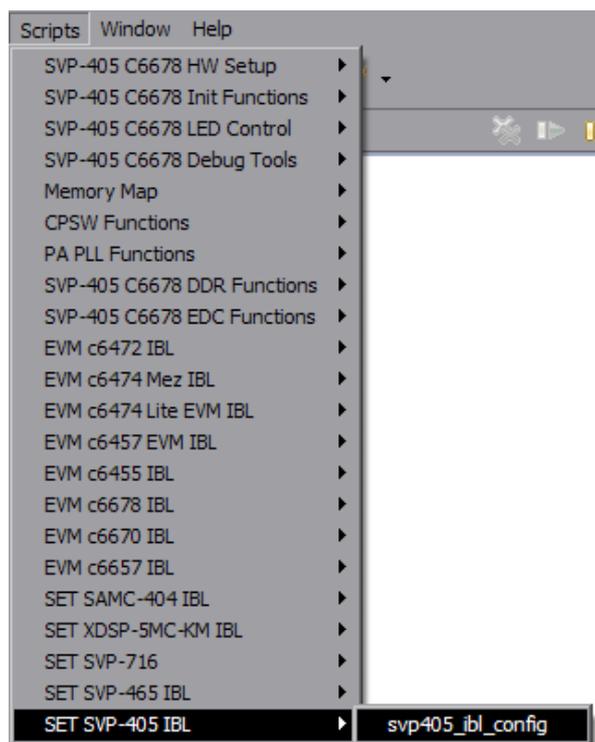


Рисунок 4-8: Пункт главного меню «Scripts > SET SVP-405 IBL»

Далее, необходимо щелкнуть мышкой в окне «Console» и нажать клавишу «Enter» для выполнения записи конфигурационных параметров загрузчика в EEPROM память. После чего, в окне «Console» должно появиться сообщение «I2c table write complete» (см. рисунок 4-9).

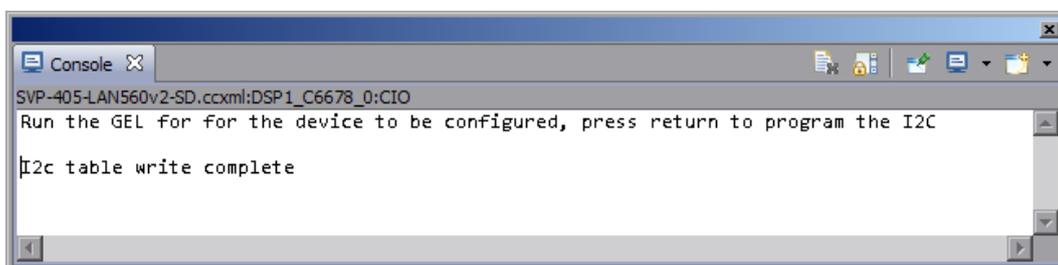


Рисунок 4-9: Внешний вид окна «Console» после выполнения записи конфигурации в EEPROM память

Конфигурационные параметры загрузчика IBL задаются в загружаемом файле «i2cConfig.gel» путем его редактирования. Редактирование данного файла должно производиться до его загрузки в окно «GEL Files». В том случае, если файл редактируется после того, как он загружен в окне «GEL Files», после редактирования, необходимо щелкнуть правой кнопкой мыши по названию файла в окне «GEL Files» и выбрать пункт меню «Reload» (см. рисунок 4-6).

В листинге 4-1 приведен фрагмент файла «i2cConfig.gel» с конфигурационными параметрами загрузчика для модуля SVP-405.

Листинг 4-1: Фрагмент GEL файла «i2cConfig.gel» с конфигурационными параметрами загрузчика IBL для модуля SVP-405

```

1691 menuitem "SET SVP-405 IBL";
1692
1693 hotmenu svp405_ibl_config()
1694 {
1695     ibl.iblMagic = ibl_MAGIC_VALUE;
1696
1697     ibl.iblEvmType = ibl_SVP_405;
1698

```

```

1699 /* Main PLL: 122.88 MHz reference, 1000MHz output */
1700 ibl.pllConfig[ibl_MAIN_PLL].doEnable = 1;
1701 ibl.pllConfig[ibl_MAIN_PLL].prediv = 29;
1702 ibl.pllConfig[ibl_MAIN_PLL].mult = 472;
1703 ibl.pllConfig[ibl_MAIN_PLL].postdiv = 2;
1704 ibl.pllConfig[ibl_MAIN_PLL].pllOutFreqMhz = 1000;
1705
1706 /* DDR PLL: 66.67 MHz reference, 1333 MHz output */
1707 /* outFreqMhz = 2 * inFreq * mult / postdiv = 66.667 * 2 * 20 / 2 = 1333.3 */
1708
1709 /* Net PLL: 122.88 MHz reference, 1044 MHz output (followed by a built in divide by 3 to give 348 MHz to PA) */
1710 /* outFreqMhz = (inFreq * mult / postdiv) / 3 = (122.88 * 17 / 2) / 3 = 1044.48 / 3 = 348.16 */
1711
1712 /* DDR PLL: */
1713 ibl.pllConfig[ibl_DDR_PLL].doEnable = 1;
1714 ibl.pllConfig[ibl_DDR_PLL].prediv = 1;
1715 ibl.pllConfig[ibl_DDR_PLL].mult = 20;
1716 ibl.pllConfig[ibl_DDR_PLL].postdiv = 2;
1717 ibl.pllConfig[ibl_DDR_PLL].pllOutFreqMhz = 1333;
1718
1719 /* Net PLL: 122.88 MHz reference, 1044 MHz output (followed by a built in divide by 3 to give 350 MHz to PA) */
1720 ibl.pllConfig[ibl_NET_PLL].doEnable = 1;
1721 ibl.pllConfig[ibl_NET_PLL].prediv = 1;
1722 ibl.pllConfig[ibl_NET_PLL].mult = 17;
1723 ibl.pllConfig[ibl_NET_PLL].postdiv = 2;
1724 ibl.pllConfig[ibl_NET_PLL].pllOutFreqMhz = 1044;
1725
1726
1727 ibl.ddrConfig.configDdr = 1;
1728 ibl.ddrConfig.uEmif.emif4p0.registerMask = ibl_EMIF4_ENABLE_sdRamConfig | ibl_EMIF4_ENABLE_sdRamRefreshCtl |
  ← ibl_EMIF4_ENABLE_sdRamTiming1 | ibl_EMIF4_ENABLE_sdRamTiming2 | ibl_EMIF4_ENABLE_sdRamTiming3 |
  ← ibl_EMIF4_ENABLE_ddrPhyCtl1;
1729
1730 // NOTE: this values is not used for configuration on SVP-465
1731 // Configuration for EMIF is hardcoded in 'hw/ddrs/emif4/emif4.c'
1732 ibl.ddrConfig.uEmif.emif4p0.sdRamConfig = 0x63062A32;
1733 ibl.ddrConfig.uEmif.emif4p0.sdRamConfig2 = 0;
1734 ibl.ddrConfig.uEmif.emif4p0.sdRamRefreshCtl = 0x00005161;
1735 ibl.ddrConfig.uEmif.emif4p0.sdRamTiming1 = 0x1113783C;
1736 ibl.ddrConfig.uEmif.emif4p0.sdRamTiming2 = 0x30B37FE3;
1737 ibl.ddrConfig.uEmif.emif4p0.sdRamTiming3 = 0x559F8ADF;
1738 ibl.ddrConfig.uEmif.emif4p0.lpDdrNvmTiming = 0;
1739 ibl.ddrConfig.uEmif.emif4p0.powerManageCtl = 0;
1740 ibl.ddrConfig.uEmif.emif4p0.iODFTTestLogic = 0;
1741 ibl.ddrConfig.uEmif.emif4p0.performCountCfg = 0;
1742 ibl.ddrConfig.uEmif.emif4p0.performCountMstRegSel = 0;
1743 ibl.ddrConfig.uEmif.emif4p0.readIdleCtl = 0;
1744 ibl.ddrConfig.uEmif.emif4p0.sysVbusmIntEnSet = 0;
1745 ibl.ddrConfig.uEmif.emif4p0.sdRamOutImpdedCalCfg = 0;
1746 ibl.ddrConfig.uEmif.emif4p0.tempAlterCfg = 0;
1747 ibl.ddrConfig.uEmif.emif4p0.ddrPhyCtl1 = 0x0010010f;
1748 ibl.ddrConfig.uEmif.emif4p0.ddrPhyCtl2 = 0;
1749 ibl.ddrConfig.uEmif.emif4p0.priClassSvceMap = 0;
1750 ibl.ddrConfig.uEmif.emif4p0.mstId2ClsSvce1Map = 0;
1751 ibl.ddrConfig.uEmif.emif4p0.mstId2ClsSvce2Map = 0;
1752 ibl.ddrConfig.uEmif.emif4p0.eccCtl = 0;
1753 ibl.ddrConfig.uEmif.emif4p0.eccRange1 = 0;
1754 ibl.ddrConfig.uEmif.emif4p0.eccRange2 = 0;
1755 ibl.ddrConfig.uEmif.emif4p0.rdWrtExcThresh = 0;
1756
1757
1758 ibl.sgmiiconfig[0].configure = 1;
1759 ibl.sgmiiconfig[0].adviseAbility = 0x9801; // 1
1760 ibl.sgmiiconfig[0].control = 32; // 1
1761 ibl.sgmiiconfig[0].txConfig = 0x106a1; // 0x108a1
1762 ibl.sgmiiconfig[0].rxConfig = 0x700621;
1763 ibl.sgmiiconfig[0].auxConfig = 0x51;
1764
1765 ibl.sgmiiconfig[1].configure = 0;
1766 ibl.sgmiiconfig[1].adviseAbility = 1;
1767 ibl.sgmiiconfig[1].control = 1;
1768 ibl.sgmiiconfig[1].txConfig = 0x106a1; // 0x108a1;
1769 ibl.sgmiiconfig[1].rxConfig = 0x700621;
1770 ibl.sgmiiconfig[1].auxConfig = 0x51;
1771
1772 ibl.mdioConfig.nMdioOps = 0;
1773
1774 ibl.spiConfig.addrWidth = 24;
1775 ibl.spiConfig.nPins = 5;
1776 ibl.spiConfig.mode = 1;
1777 ibl.spiConfig.csel = 2;
1778 ibl.spiConfig.c2tdelay = 8;
1779 ibl.spiConfig.busFreqMHz = 20;
1780
1781 ibl.emifConfig[0].csSpace = 2;
1782 ibl.emifConfig[0].busWidth = 8;
1783 ibl.emifConfig[0].waitEnable = 0;

```

```

1784
1785     ibl.emifConfig[1].csSpace    = 0;
1786     ibl.emifConfig[1].busWidth  = 0;
1787     ibl.emifConfig[1].waitEnable = 0;
1788
1789     ibl.bootModes[0].bootMode = ibl_BOOT_MODE_NOR;
1790     ibl.bootModes[0].priority = ibl_HIGHEST_PRIORITY;
1791     ibl.bootModes[0].port     = 0;
1792
1793     ibl.bootModes[0].u.norBoot.bootFormat = ibl_BOOT_FORMAT_ELF;
1794     ibl.bootModes[0].u.norBoot.bootAddress[0][0] = 0; /* Image 0 NOR offset byte address in LE mode */
1795     ibl.bootModes[0].u.norBoot.bootAddress[0][1] = 0x800000; /* Image 1 NOR offset byte address in LE mode */
1796     ibl.bootModes[0].u.norBoot.bootAddress[1][0] = 0; /* Image 0 NOR offset byte address in BE mode */
1797     ibl.bootModes[0].u.norBoot.bootAddress[1][1] = 0x800000; /* Image 1 NOR offset byte address in BE mode */
1798     ibl.bootModes[0].u.norBoot.interface = ibl_PMEM_IF_SPI;
1799     ibl.bootModes[0].u.norBoot.blob[0][0].startAddress = 0x80000000; /* Image 0 load start address in LE mode */
1800     ibl.bootModes[0].u.norBoot.blob[0][0].sizeBytes = 0x800000; /* Image 0 size (10 MB) in LE mode */
1801     ibl.bootModes[0].u.norBoot.blob[0][0].branchAddress = 0x80000000; /* Image 0 branch address after loading in LE mode
1802     ← */
1803     ibl.bootModes[0].u.norBoot.blob[0][1].startAddress = 0x80000000; /* Image 1 load start address in LE mode */
1804     ibl.bootModes[0].u.norBoot.blob[0][1].sizeBytes = 0x800000; /* Image 1 size (10 MB) in LE mode */
1805     ibl.bootModes[0].u.norBoot.blob[0][1].branchAddress = 0x80000000; /* Image 1 branch address after loading in LE mode
1806     ← */
1807     ibl.bootModes[0].u.norBoot.blob[1][0].startAddress = 0x80000000; /* Image 0 load start address in BE mode */
1808     ibl.bootModes[0].u.norBoot.blob[1][0].sizeBytes = 0x800000; /* Image 0 size (10 MB) in BE mode */
1809     ibl.bootModes[0].u.norBoot.blob[1][0].branchAddress = 0x80000000; /* Image 0 branch address after loading in BE mode
1810     ← */
1811     ibl.bootModes[0].u.norBoot.blob[1][1].startAddress = 0x80000000; /* Image 1 load start address in BE mode */
1812     ibl.bootModes[0].u.norBoot.blob[1][1].sizeBytes = 0x800000; /* Image 1 size (10 MB) in BE mode */
1813     ibl.bootModes[0].u.norBoot.blob[1][1].branchAddress = 0x80000000; /* Image 1 branch address after loading in BE mode
1814     ← */
1815
1816     ibl.bootModes[1].bootMode = ibl_BOOT_MODE_NONE;
1817
1818     ibl.bootModes[2].bootMode = ibl_BOOT_MODE_TFTP;
1819     ibl.bootModes[2].priority = ibl_HIGHEST_PRIORITY+1;
1820     ibl.bootModes[2].port = 0; // ibl_PORT_SWITCH_ALL;
1821
1822     ibl.bootModes[2].u.ethBoot.doBootp = TRUE;
1823     ibl.bootModes[2].u.ethBoot.useBootpServerIp = TRUE;
1824     ibl.bootModes[2].u.ethBoot.useBootpFileName = TRUE;
1825     ibl.bootModes[2].u.ethBoot.bootFormat = ibl_BOOT_FORMAT_ELF;
1826
1827     SETIP(ibl.bootModes[2].u.ethBoot.ethInfo.ipAddr, 192,168,1,3);
1828     SETIP(ibl.bootModes[2].u.ethBoot.ethInfo.serverIp, 192,168,1,2);
1829     SETIP(ibl.bootModes[2].u.ethBoot.ethInfo.gatewayIp, 192,168,1,1);
1830     SETIP(ibl.bootModes[2].u.ethBoot.ethInfo.netmask, 255,255,255,0);
1831
1832     /* Use the e-fuse value */
1833     ibl.bootModes[2].u.ethBoot.ethInfo.hwAddress[0] = 0;
1834     ibl.bootModes[2].u.ethBoot.ethInfo.hwAddress[1] = 0;
1835     ibl.bootModes[2].u.ethBoot.ethInfo.hwAddress[2] = 0;
1836     ibl.bootModes[2].u.ethBoot.ethInfo.hwAddress[3] = 0;
1837     ibl.bootModes[2].u.ethBoot.ethInfo.hwAddress[4] = 0;
1838     ibl.bootModes[2].u.ethBoot.ethInfo.hwAddress[5] = 0;
1839
1840     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[0] = 's';
1841     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[1] = 'v';
1842     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[2] = 'p';
1843     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[3] = '4';
1844     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[4] = '0';
1845     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[5] = '5';
1846     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[6] = '.';
1847     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[7] = 'b';
1848     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[8] = 'i';
1849     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[9] = 'n';
1850     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[10] = '\0';
1851     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[11] = '\0';
1852     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[12] = '\0';
1853     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[13] = '\0';
1854     ibl.bootModes[2].u.ethBoot.ethInfo.fileName[14] = '\0';
1855
1856     ibl.bootModes[2].u.ethBoot.blob.startAddress = 0x80000000; /* Load start address */
1857     ibl.bootModes[2].u.ethBoot.blob.sizeBytes = 0x20000000;
1858     ibl.bootModes[2].u.ethBoot.blob.branchAddress = 0x80000000; /* Branch address after loading */
1859
1860     ibl.chkSum = 0;
1861 }

```

В таблице 4-1 дано краткое описание основных конфигурационных параметров, из файла «i2cConfig.gel», их назначение и возможные значения.

Параметры, которые не описаны в таблице 4-1, связаны с аппаратной конфигурацией оборудования модуля SVP-405 и их изменение не рекомендуется.

Таблица 4-1: Основные конфигурационные параметры файла «i2cConfig.gel»

Параметр	Описание
<code>ibl.bootModes[2].u.ethBoot.doBootp</code>	Для режима загрузки TFTP. Если равен TRUE, IBL будет пытаться получить сетевую конфигурацию по протоколу BOOTP. Если равно FALSE, то будут использованы параметры конфигурации сети описанные ниже.
<code>ibl.bootModes[2].u.ethBoot.bootFormat</code>	Задаёт формат загружаемого образа. Может принимать одно из следующих значений: <ul style="list-style-type: none"> <code>ibl_BOOT_FORMAT_COFF</code> — объектный формат COFF. Загружается через встроенный в IBL загрузчик COFF файлов; <code>ibl_BOOT_FORMAT_ELF</code> — объектный формат ELF. Загружается через встроенный в IBL загрузчик ELF файлов; <code>ibl_BOOT_FORMAT_BBLOB</code> — бинарный формат готовый к загрузке на модуле (не требующий соответствующего загрузчика); <code>ibl_BOOT_FORMAT_AUTO</code> — автоматическое определение формата по сигнатуре файла; <code>ibl_BOOT_FORMAT_NAME</code> — автоматическое определение формата по расширению загружаемого файла («.out» — COFF, «.elf» — ELF, «.bin» — BBLOB).
<code>ibl.bootModes[2].u.ethBoot.ethInfo.hwAddress[0...5]</code>	Задаёт значение аппаратного MAC адреса сетевого интерфейса. Если все значения равны 0, используется встроенный производителем в процессор MAC-адрес.
<code>ibl.bootModes[2].u.ethBoot.ethInfo.fileName[0...63]</code>	Задаёт имя файла для загрузки. Максимальная длина имени файла составляет 64 символа. Последним символом имени файла загрузки обязательно должен быть символ «\0».
<code>ibl.bootModes[2].u.ethBoot.ethInfo.ipAddr</code>	Определяет значение фиксированного IP адреса. Океты IP адреса в файле «i2cConfig.gel» записываются через запятую. Значение параметр используется в случае, если <code>ibl.bootModes[2].u.ethBoot.doBootp = FALSE</code> .
<code>ibl.bootModes[2].u.ethBoot.ethInfo.serverIp</code>	Задаёт IP адрес сервера загрузки в случае, если <code>ibl.bootModes[2].u.ethBoot.doBootp = FALSE</code> .
<code>ibl.bootModes[2].u.ethBoot.ethInfo.gatewayIp</code>	Задаёт IP адрес основного шлюза в случае, если <code>ibl.bootModes[2].u.ethBoot.doBootp = FALSE</code> .
<code>ibl.bootModes[2].u.ethBoot.ethInfo.netmask</code>	Задаёт маску подсети в том случае, если <code>ibl.bootModes[2].u.ethBoot.doBootp = FALSE</code> .
<code>ibl.bootModes[2].u.ethBoot.useBootpServerIp</code>	Если равен FALSE, то в качестве IP адреса сервера загрузки будет использовано значение параметра <code>ibl.bootModes[2].u.ethBoot.ethInfo.serverIp</code> . Если равен TRUE, то будет использован IP адрес сервера загрузки, указанный в BOOTP ответе от BOOTP сервера. Данный параметр имеет значение только в том случае, когда <code>ibl.bootModes[2].u.ethBoot.doBootp = TRUE</code> .
<code>ibl.bootModes[2].u.ethBoot.useBootpFileName</code>	Если равен FALSE, то в качестве имени файла для загрузки будет использовано значение параметра <code>ibl.bootModes[2].u.ethBoot.ethInfo.fileName[0...63]</code> . Если равен TRUE, то будет использовано имя файла загрузки, указанное в BOOTP ответе от BOOTP сервера. Данный параметр имеет значение только в том случае, когда <code>ibl.bootModes[2].u.ethBoot.doBootp = TRUE</code> .
<code>ibl.bootModes[2].u.ethBoot.blob.startAddress</code>	Адрес области памяти куда будет осуществляться загрузка образа.
<code>ibl.bootModes[2].u.ethBoot.blob.sizeBytes</code>	Максимальный допустимый объем образа, который допускается загрузить.
<code>ibl.bootModes[2].u.ethBoot.blob.branchAddress</code>	Адрес точки входа, куда будет осуществлен переход после завершения загрузки образа приложения.

5 Импорт и запуск целевой конфигурации модуля

Для загрузки кода приложений на модуль SVP-405, в первую очередь, необходимо запустить целевую конфигурацию модуля SVP-405. В папке «TargetConfigurations» сопроводительного диска к модулю SVP-405 находятся файлы целевых конфигураций для различных отладчиков. В данном документе рассматривается загрузка кода через отладчик Spectrum Digital XDS560v2 STM LAN. Данному отладчику соответствует файл целевой конфигурации «SVP-405-LAN560v2-SD.ccxml», который необходимо импортировать в рабочее пространство.

Выберите пункт главного меню «View > Target Configurations» (рисунок 5-1)

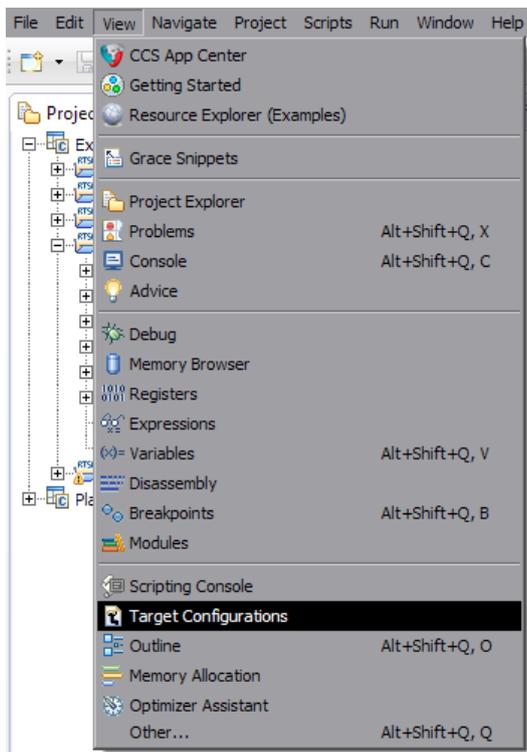


Рисунок 5-1: Пункт меню для отображения окна целевых конфигураций

В окне целевых конфигураций («Target Configurations»), нажмите правой кнопкой мыши на свободной области для вызова контекстного меню и выберите пункт «Import Target Configuration» (см. рисунок 5-2).

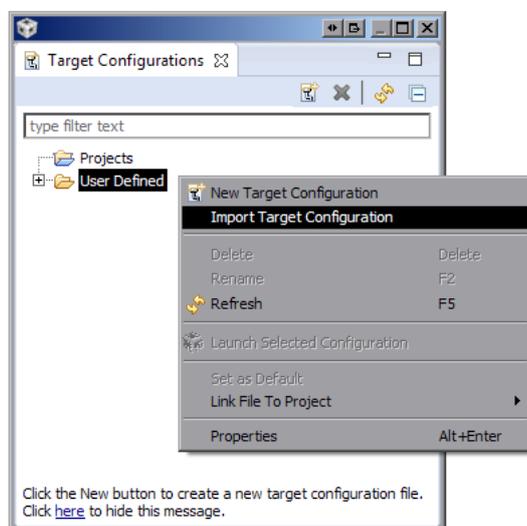


Рисунок 5-2: Меню импорта целевой конфигурации

В появившемся окне выбора файла (см. рисунок 5-3) необходимо выбрать файл «SVP-405-LAN560v2-SD.ccxml» и нажать на кнопку «Открыть». В данном документе предполагается, что папка «TargetConfigurations» с сопроводительного диска к модулю SVP-405, где расположен файл «SVP-405-LAN560v2-SD.ccxml», скопирована в папку «D:/Dev/Modules/SVP-405/TargetConfigurations».

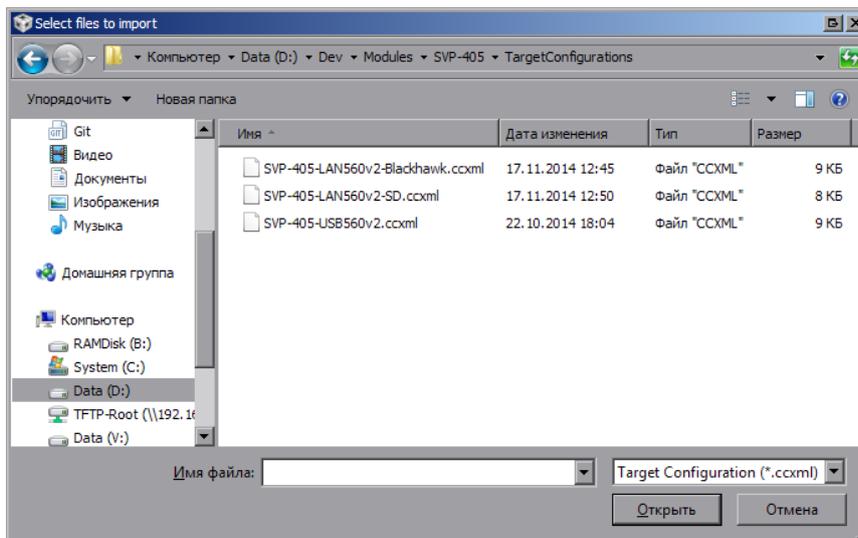


Рисунок 5-3: Окно выбора файла для импорта целевой конфигурации

После нажатия на кнопку «Открыть» появится окно выбора способа импорта файла целевой конфигурации (рисунок 5-4). Необходимо выбрать способ «Link to files» и нажать на кнопку «ОК».

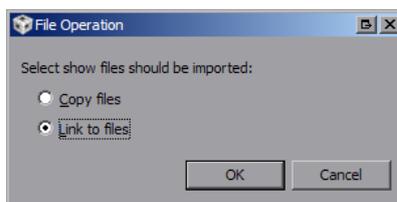


Рисунок 5-4: Окно выбора способа импорта файла целевой конфигурации

Для запуска целевой конфигурации, в окне целевых конфигураций («Target Configurations»), необходимо нажать правой кнопкой мыши на целевой конфигурации и выбрать пункт меню «Launch Selected Configuration» (см. рисунок 5-5).

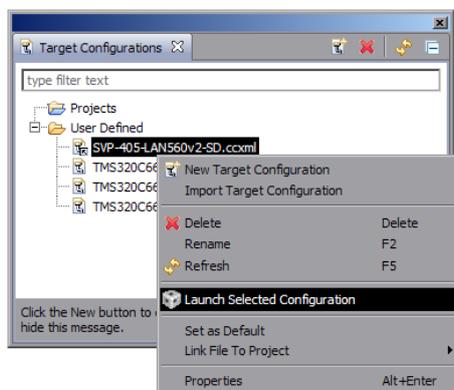


Рисунок 5-5: Запуск целевой конфигурации

После запуска целевой конфигурации модуля SVP-405, среда разработки CCS перейдет в режим отладки и в окне «Debug» будет выведен список ядер обоих процессоров модуля SVP-405, как показано на рисунке 5-6.

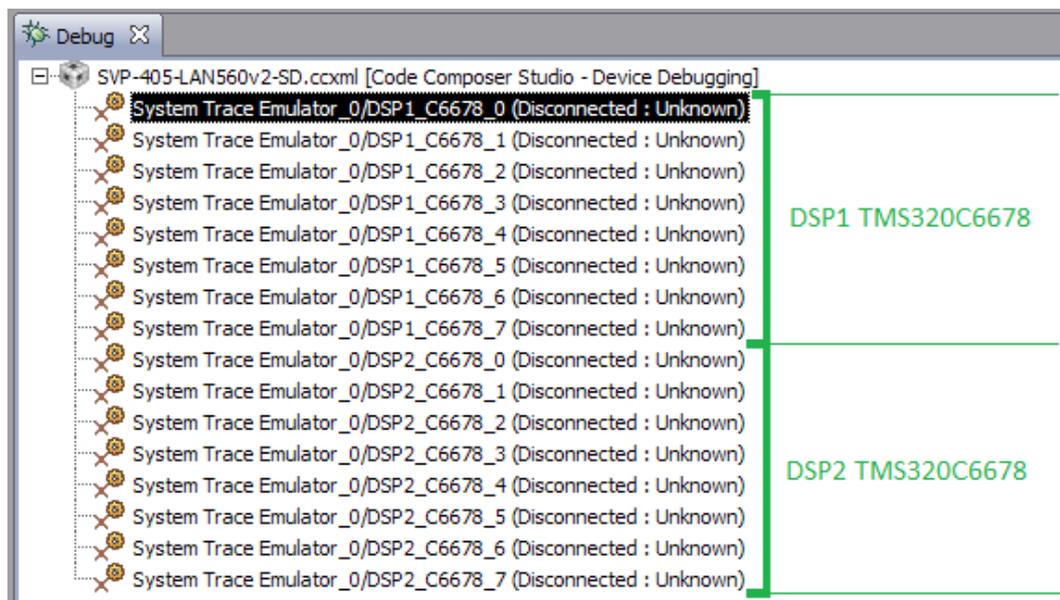


Рисунок 5-6: Список ядер процессоров модуля SVP-405

Примечание

По умолчанию, среда CCS настроена таким образом, что при запуске кода сразу на нескольких процессорах или нескольких ядрах одного процессора, вывод (C/O) со всех ядер процессоров будет выводиться в одно и то же окно «Console». В приложении Б даны инструкции по настройке отдельного вывода (C/O) каждого ядра процессора в отдельное окно «Console».

6 Подготовка образов приложений для загрузки на нескольких ядрах процессора

Для подготовки образов приложений, которые необходимо загрузить сразу на несколько ядер процессора, используется набор специальных утилит [MAD](#) разработанных компанией Texas Instruments.

Набор утилит [MAD](#) входит в состав [MCSDK](#) (MultiCore Software Development Kit). Более подробную информацию по возможностям и приемам использования набора утилит [MAD](#) можно получить ознакомившись с документом [2].

6.1 Компоненты [MAD Utils](#)

[MAD Utils](#) предоставляет набор утилит для достижения следующих целей:

- необходимость выполнения загрузки множества приложений на несколько ядер;
- необходимость сохранения памяти, путем выделения общего кода многоядерных приложений.

[MAD Utils](#) состоит из пяти основных утилит, которые можно разделить на две категории: «Утилиты времени сборки» и «Утилиты времени исполнения».

Утилиты времени сборки (build time utilities) включают в себя:

- [Static linker](#) — статический линковщик, предназначенный для линковки приложений и зависимых динамических общих объектов [DSO](#) (Dynamic Shared Object).
- [Prelink Tool](#) — используется для назначения сегментам [ELF](#) файла виртуальных адресов.
- [MAP Tool](#) — используется для назначения виртуальных адресов сегментам многоядерных приложений. Пользователь определяет нужные разделы памяти для устройства и высокоуровневые инструкции сегмента размещения в [MAP Tool](#). Основываясь на данной информации, [MAP Tool](#) определяет виртуальные и физические адреса времени исполнения для каждого сегмента [ELF](#) файла для каждого приложения. Затем вызывается [Prelink Tool](#) для выделения области для хранения всех приложений и их [DSO](#). [MAP Tool](#) также генерирует набор активационных записей для загрузки приложения на определенное ядро. Активационные записи — это инструкции загрузчика времени исполнения, которые выполняют следующие действия:
 - настройка карты виртуальной памяти и атрибутов доступа и защиты областей памяти;
 - копирование и инициализация загружаемых сегментов памяти на их адреса времени исполнения.

Полученный образ приложения и активационные записи запаковываются в образ [ROMFS](#), который предназначен для загрузки на целевом устройстве.

Утилиты времени исполнения (run time utilities) включают в себя:

- Утилита начальной загрузки. В качестве данной утилиты выступает загрузчик [IBL](#), который предоставляет функциональность загрузки образа [ROMFS](#) в общую внешнюю память устройства ([DDR](#)).
- [MAD Loader](#) — утилита загрузки времени исполнения, которая обеспечивает функционал запуска приложений на заданном ядре. Данная утилита выполняет следующие действия для обеспечения запуска приложения на ядре:
 - конфигурация карты виртуальной памяти для ядра;
 - конфигурация атрибутов и режимов доступа для каждого раздела памяти;
 - копирование сегментов памяти с локального адреса в адрес времени исполнения;
 - выполнение прединициализационных функций приложения;
 - выполнение инициализационных функций зависимых библиотек и приложений;
 - запуск приложения (переход по адресу точки входа).

6.2 Режимы работы MAD Utils

MAD Utils позволяет работать в двух режимах:

- **Prelinker bypass mode.** В данном режиме утилита MAP Tool не выполняет назначения адресов сегментам приложения и Prelink Tool не вызывается. Данный режим подходит в тех случаях, когда просто требуется выполнить загрузку приложения или нескольких приложений на конкретном ядре или ядрах.
- **Prelinker mode.** В данном режиме утилита MAP Tool выполняет назначение адресов сегментам приложения и вызывает Prelink Tool. Данный режим подходит в тех случаях, когда разработчику требуется, чтобы MAP Tool выполнил присвоение адресов для общего кода между несколькими ядрами, на которых должно работать приложение.

Внимание



В данном документе рассматривается только работа MAD Utils в режиме работы «Prelinker bypass mode». Информацию по работе с MAD Utils в режиме «Prelinker mode» можно получить обратившись к документу [2].

6.3 Работа с MAD Utils в режиме «Prelinker bypass mode»

На рисунке 6-1 изображена схема работы MAD Utils в режиме «Prelinker bypass mode».

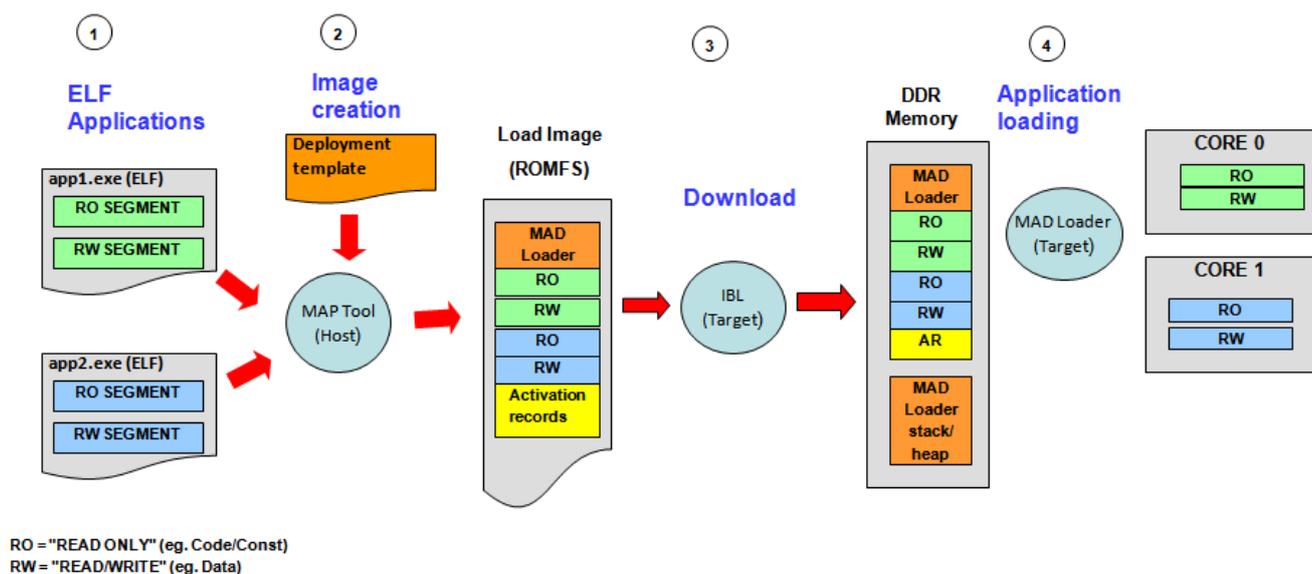


Рисунок 6-1: Схема работы MAD в режиме «Prelinker bypass mode»

Работа с MAD Utils может быть разделена на два этапа — этап подготовки образа для загрузки и этап загрузки образа на целевом устройстве.

Этап подготовки образа состоит из следующих шагов:

- Сборка приложения (статическая линковка по адресам исполнения);
- Создание файла конфигурации развертывания для MAP Tool с определением приложения для каждого ядра.
- Запуск MAP Tool с файлом конфигурации развертывания в качестве входных данных.
- MAP Tool создает образ для загрузки (в формате ROMFS), содержащий в себе активационные записи для каждого приложения.

Этап загрузки образа на целевом устройстве выглядит следующим образом:

- При загрузке, в первую очередь, на устройстве выполняется код загрузчика из ROM памяти. Этот загрузчик выполняет загрузку IBL, который должен быть записан в I²C EEPROM память устройства.

- IBL выполняет загрузку MAD образа с TFTP сервера или NOR флеш памяти в DDR память. При этом, IBL должен быть сконфигурирован таким образом, что бы переход осуществлялся по адресу точки входа MAD Loader.
- MAD Loader выполняет разбор образа ROMFS и выполняет загрузку сегментов приложения по их адресам времени исполнения для каждого ядра.
- MAD Loader выполняет переход по адресу точки входа для каждого ядра, запуская таким образом выполнения приложения на ядрах целевого устройства.

6.4 Конфигурация MAP Tool

Входными данными для MAP Tool является конфигурационный файл в формате JSON. Данный конфигурационный файл должен содержать параметры, приведенные в таблице 6-1.

Таблица 6-1: Параметры конфигурационного файла MAP Tool

Параметр	Описание
deploymentCfgFile	Путь к конфигурационному файлу развертывания.
LoadImageName	Имя файла образа, который будет сгенерирован. Данный файл образа (в формате ROMFS) будет помещен в папку «images».
prelinkExe	Путь к запускаемому файлу Prelink Tool. Данная утилита является частью CGT (Code Generation Tools).
stripExe	Путь к запускаемому файлу Strip Tool. Данная утилита является частью CGT.
ofdTool	Путь к запускаемому файлу OFD Tool. Данная утилита является частью CGT.
malApp	Путь к файлу приложения MAD Loader.
nmlLoader	Путь к файлу приложения NML Loader. NML Loader является составной частью MAD Loader.

В листинге 6-1 приведен пример конфигурационного файла для MAP Tool.

Листинг 6-1: Пример конфигурационного файла для MAP Tool

```

1 {
2   "deploymentCfgFile" : ".\\deploy.json",
3   "LoadImageName"    : "mad_test.bin",
4   "prelinkExe"       : "C:\\ti\\ccsv5\\tools\\compiler\\c6000_7.4.2\\bin\\prelink6x",
5   "stripExe"         : "C:\\ti\\ccsv5\\tools\\compiler\\c6000_7.4.2\\bin\\strip6x",
6   "ofdTool"          : "C:\\ti\\ccsv5\\tools\\compiler\\c6000_7.4.2\\bin\\ofd6x.exe",
7   "malApp"           : "C:\\ti\\mcsdk_2_01_02_06\\tools\\boot_loader\\mad-utils\\mad-loader\\bin\\C6670\\le\\mal_app.exe",
8   "nmlLoader"        : "C:\\ti\\mcsdk_2_01_02_06\\tools\\boot_loader\\mad-utils\\mad-loader\\bin\\C6670\\le\\nml.exe"
9 }

```

В листинге 6-1 указаны пути с учетом следующих предположений:

- используется MCSDK версии 2.01.02.06 установленная в папку «C:/ti/mcsdk_2_01_02_06»;
- используется компилятор версии 7.4.2 установленный в папку «C:/ti/ccsv5/tools/compiler/c6000_7.4.2»;
- файл конфигурации развертывания «deploy.json» находится в той же папке.

6.5 Конфигурационный файл развертывания

В режиме «Prelinker bypass mode» конфигурационный файл развертывания определяет следующую информацию:

- области памяти для загрузки;
- данные приложений для развертывания.

Файл конфигурации развертывания является файлом в формате JSON. Для режима «Prelinker bypass mode» файл должен содержать параметры, приведенные в таблице 6-2.

Таблица 6-2: Параметры файла конфигурации развертывания

Параметр	Описание
deviceName	JSON объект, определяющий целевое устройство. Может принимать следующие значения: <ul style="list-style-type: none"> C6657 — 2-х ядерный процессор TMS320C6657; C6670 — 4-х ядерный процессор TMS320C6670; C6678 — 8-и ядерный процессор TMS320C6678;
partitions	Данный параметр идентифицирует разделы памяти, которые будут загружены из ROMFS образа в память устройства при загрузке. Определение каждого из разделов должно содержать следующие параметры: <ul style="list-style-type: none"> name — имя раздела. Используется для идентификации раздела памяти в отладочном выводе MAP Tool; vaddr — виртуальный адрес раздела памяти; size — размер области памяти в байтах; loadPartition — определяет будет ли данный раздел загружен или нет.
applications	Данный параметр идентифицирует приложения для развертывания. Определение каждого приложения должно содержать следующие параметры: <ul style="list-style-type: none"> name — имя приложения. Используется для идентификации в отладочном выводе; fileName — имя файла образа приложения полученного в результате сборки приложения; allowedCores — номера ядер, на которых разрешен запуск данного приложения.
appDeployment	Данный параметр определяет приложения для каждого из ядер целевого устройства. Представляет из себя массив (размер массива должен соответствовать количеству ядер целевого устройства), каждый элемент которого должен содержать имя приложения, которое необходимо загрузить на соответствующем ядре целевого устройства. В том случае, если на каком либо ядре загружать приложение не требуется, соответствующий элемент данного массива задается в виде пустой строки ("").

Листинг 6-2: Пример файла конфигурации развертывания

```

1  {
2      "deviceName" : "C6670",
3      "partitions" : [
4          {
5              "name" : "load-partition",
6              "vaddr" : "0x9e000000",
7              "size" : "0x2000000",
8              "loadPartition" : true
9          }
10     ],
11     "applications" : [
12         {
13             "name" : "app1",
14             "fileName" : "../build/app_1.out",
15             "allowedCores" : [0,1,2,3]
16         },
17         {
18             "name" : "app2",
19             "fileName" : "../build/app_2.out",
20             "allowedCores" : [0,1,2,3]
21         }
22     ],
23     "appDeployment" : [
24         "app1",
25         "app1",
26         "",
27         "app2"
28     ]
29 }

```

6.6 Запуск MAP Tool

Для запуска MAP Tool потребуется установленный Python интерпретатор, который можно бесплатно скачать с официального сайта¹.

Запуск MAP Tool в режиме «Prelinker bypass mode» осуществляется путем выполнения команды:

```
python maptool.py <файл_конфигурации> bypass-prelink
```

Внимание



Следует помнить, что образ, предназначенный для загрузки на целевом устройстве, будет создан в папке «images». Имя файла образа задается в конфигурационном файле MAP Tool (см. раздел 6.4).

В качестве параметра <файл_конфигурации> необходимо указать имя файла конфигурации MAP Tool (см. раздел 6.4). Python скрипт «maptool.py» располагается в папке «tools/boot_loader/mad-utils/map-tool» относительно папки установки MCSDK («C:\ti\mcSDK_2_01_02_06»). Для запуска MAP Tool непосредственно из папки установки MCSDK необходимо выполнять команду указав полный путь к файлу «maptool.py»:

```
python "C:\ti\mcSDK_2_01_02_06\tools\boot_loader\mad-utils\map-tool\maptool.py" <файл_конфигурации>
← bypass-prelink
```

Примечание

Инструкции по записи полученного образа в NOR флеш память приведены в разделе 3.3 данного документа.

Инструкции по установке и настройке служб BOOTP и TFTP, необходимых для осуществления загрузки образа по Ethernet с TFTP сервера, можно найти в документе [3].

6.7 Загрузка образа

Для того, чтобы загрузчик IBL мог корректно загружать образы, подготовленные при помощи MAD Utils, необходимо выполнить его конфигурацию. Подробное описание процедуры конфигурации загрузчика IBL приведено в разделе 4 данного документа.

В том случае, если подготовленный образ планируется загружать по Ethernet с TFTP сервера, то необходимо установить следующие параметры:

```
ib1.bootModes[2].u.ethBoot.bootFormat      = ib1_BOOT_FORMAT_BLOB;
ib1.bootModes[2].u.ethBoot.blob.startAddress = 0x9e000000;
ib1.bootModes[2].u.ethBoot.blob.sizeBytes   = 0x20000000;
ib1.bootModes[2].u.ethBoot.blob.branchAddress = 0x9e001040;
```

Если же загрузка подготовленного образа будет происходить с NOR флеш памяти, то необходимо установить параметры:

```
ib1.bootModes[0].u.norBoot.bootFormat      = ib1_BOOT_FORMAT_BLOB;
ib1.bootModes[0].u.norBoot.blob[0][0].startAddress = 0x9e000000;
ib1.bootModes[0].u.norBoot.blob[0][0].sizeBytes   = 0x800000;
ib1.bootModes[0].u.norBoot.blob[0][0].branchAddress = 0x9e001040;
```

Дополнительное описание конфигурационных параметров приведено в таблице 4-1 раздела 4 данного документа.

¹ <https://www.python.org/download/>

Приложение А Примеры работы скрипта записи EEPROM и NOR флеш памяти

В листинге А-1 приведен пример вывода в терминал запуска скрипта «svp405_program.bat» для записи в NOR флеш память образов по умолчанию на все четыре процессора модуля SVP-405.

Примечание

В целях сокращения объема листингов, приведенных в данном приложении, некоторые их части вырезаны. В местах вырезанных данных листингов, помещен текст в виде «ВЫРЕЗАНО: Описание вырезанного текста».

Листинг А-1: Вывод в терминал при запуске команды «svp405_program.bat NOR all»

```
1 D:\Dev\Modules\SVP-405\Program>svp405_program.bat NOR all
2
3 SVP-405 program script started
4 Copyright (c) 2014, Scan Engineering Telecom SPb
5
6 Writing to DSP1 ENABLED
7 Writing to DSP2 ENABLED
8
9 Target configuration file:
10  "../TargetConfigurations/SVP-405-LAN560v2-SD.ccxml"
11
12 Images for DSP's:
13   DSP1: "bin/nor_c6678.bin" (564107 bytes)
14   DSP2: "bin/nor_c6678.bin" (564107 bytes)
15
16 Configuring debug server for SVP-405 board...
17 Done!
18 Opening a debug session for 2 DSP(s)...
19 Loaded FPGA Image: C:\ti\ccsv5\ccs_base\common\uscif\dtc_top.jbc
20 Done!
21 Connecting to DSP(s)...
22
23 ВЫРЕЗАНО: Вырезана часть с GEL инициализацией ядер процессоров
24
25 Done!
26 Loading binary images to DSP(s) memory...
27   Loading file "bin/nor_c6678.bin" (564107 bytes)...
28   Loading file "bin/nor_c6678.bin" (564107 bytes)...
29 Done!
30 Configuring writers on DSP(s)...
31 Done!
32 Executing writers on DSP(s)...
33 NOR Writer Utility Version 02.00.00.00
34 Copyright (c) 2014, Scan Engineering Telecom SPb
35
36 Write size:   564107 bytes
37 Start address: 0x0
38
39 Flashing sector 0 (0 bytes of 564107)
40 NOR Writer Utility Version 02.00.00.00
41 Copyright (c) 2014, Scan Engineering Telecom SPb
42
43 Write size:   564107 bytes
44 Start address: 0x0
45
46 Flashing sector 0 (0 bytes of 564107)
47 Flashing sector 1 (65536 bytes of 564107)
48 Flashing sector 1 (65536 bytes of 564107)
49 Flashing sector 2 (131072 bytes of 564107)
50 Flashing sector 2 (131072 bytes of 564107)
51 Flashing sector 3 (196608 bytes of 564107)
52 Flashing sector 3 (196608 bytes of 564107)
53 Flashing sector 4 (262144 bytes of 564107)
54 Flashing sector 4 (262144 bytes of 564107)
55 Flashing sector 5 (327680 bytes of 564107)
56 Flashing sector 5 (327680 bytes of 564107)
57 Flashing sector 6 (393216 bytes of 564107)
58 Flashing sector 6 (393216 bytes of 564107)
59 Flashing sector 7 (458752 bytes of 564107)
60 Flashing sector 7 (458752 bytes of 564107)
```

```
61 Flashing sector 8 (524288 bytes of 564107)
62 Flashing sector 8 (524288 bytes of 564107)
63 Reading and verifying sector 0 (0 bytes of 564107)
64 Reading and verifying sector 1 (65536 bytes of 564107)
65 Reading and verifying sector 2 (131072 bytes of 564107)
66 Reading and verifying sector 3 (196608 bytes of 564107)
67 Reading and verifying sector 0 (0 bytes of 564107)
68 Reading and verifying sector 4 (262144 bytes of 564107)
69 Reading and verifying sector 1 (65536 bytes of 564107)
70 Reading and verifying sector 5 (327680 bytes of 564107)
71 Reading and verifying sector 2 (131072 bytes of 564107)
72 Reading and verifying sector 6 (393216 bytes of 564107)
73 Reading and verifying sector 3 (196608 bytes of 564107)
74 Reading and verifying sector 7 (458752 bytes of 564107)
75 Reading and verifying sector 4 (262144 bytes of 564107)
76 Reading and verifying sector 8 (524288 bytes of 564107)
77 Reading and verifying sector 5 (327680 bytes of 564107)
78 NOR programming completed successfully
79 Reading and verifying sector 6 (393216 bytes of 564107)
80 Reading and verifying sector 7 (458752 bytes of 564107)
81 Reading and verifying sector 8 (524288 bytes of 564107)
82 NOR programming completed successfully
83 Done!
84 Terminating debug sessions on DSP(s)...
85 Done!
86 Stopping debug server...
87 Done!
88
89
90 D:\Dev\Modules\SVP-405\Program>
```

В листинге A-2 приведен пример вывода в терминал запуска скрипта «svp405_program.bat» для записи в EEPROM память образов по умолчанию на все четыре процессора модуля SVP-405.

Листинг A-2: Вывод в терминал при запуске команды «svp405_program.bat EEPROM all»

```
1
2 D:\Dev\Modules\SVP-405\Program>svp405_program.bat EEPROM all
3
4 SVP-405 program script started
5 Copyright (c) 2014, Scan Engineering Telecom SPb
6
7 Writing to DSP1 ENABLED
8 Writing to DSP2 ENABLED
9
10 Target configuration file:
11   "../TargetConfigurations/SVP-405-LAN560v2-SD.ccxml"
12
13 Images for DSP's:
14   DSP1: "bin/eeeprom_0x50_c6678.bin" (57468 bytes)
15   DSP2: "bin/eeeprom_0x50_c6678.bin" (57468 bytes)
16
17 Configuring debug server for SVP-405 board...
18 Done!
19 Opening a debug session for 2 DSP(s)...
20 Loaded FPGA Image: C:\ti\ccsv5\ccs_base\common\uscif\dtc_top.jbc
21 Done!
22 Connecting to DSP(s)...
23
24 ВЫРЕЗАНО: Вырезана часть с GEL инициализацией ядер процессоров
25
26 Loading binary images to DSP(s) memory...
27   Loading file "bin/eeeprom_0x50_c6678.bin" (57468 bytes)...
28   Loading file "bin/eeeprom_0x50_c6678.bin" (57468 bytes)...
29 Done!
30 Configuring writers on DSP(s)...
31 Done!
32 Executing writers on DSP(s)...
33 EEPROM Writer Utility Version 02.00.00.00
34 Copyright (c) 2014, Scan Engineering Telecom SPb
35
36 Write size:      57468 bytes
37 I2C bus address: 0x50
38 I2C start address: 0x0
39 Swap data:      no
40
41 Writing 57468 bytes from DSP memory address 0x0c000100 to EEPROM (0x0050)...
42 Returned platform error number is 4
43 EEPROM write failed
44 EEPROM Writer Utility Version 02.00.00.00
45 Copyright (c) 2014, Scan Engineering Telecom SPb
46
47 Write size:      57468 bytes
48 I2C bus address: 0x50
49 I2C start address: 0x0
50 Swap data:      no
51
52 Writing 57468 bytes from DSP memory address 0x0c000100 to EEPROM (0x0050)...
53 Reading 57468 bytes from EEPROM (0x0050) to DSP memory address 0x0c010100...
54 Verifying data read...
55 EEPROM programming completed successfully
56 Done!
57 Terminating debug sessions on DSP(s)...
58 Done!
59 Stopping debug server...
60 Done!
61
62
63 D:\Dev\Modules\SVP-405\Program>
```

Приложение Б Разделение вывода сообщений (CIO) ядер процессоров

По умолчанию, среда CCS настроена таким образом, что при запуске кода сразу на нескольких процессорах или нескольких ядрах одного процессора, вывод (CIO) со всех ядер процессоров будет выводиться в одно и то же окно «Console». В данном приложении даны инструкции по настройке отдельного вывода (CIO) каждого ядра процессора в отдельное окно «Console».

После запуска целевой конфигурации, в окне «Debug», нажмите правой кнопкой мыши на названии файла целевой конфигурации и выберите пункт меню «Edit X...», где X — имя файла целевой конфигурации (см. рисунок Б-1).

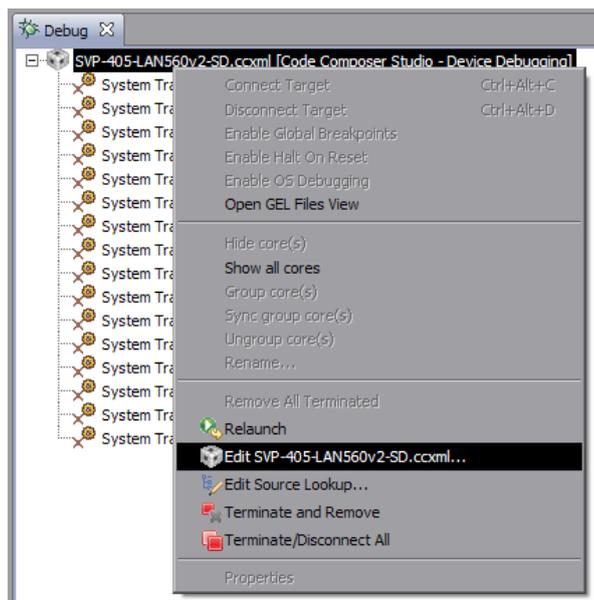


Рисунок Б-1: Контекстное меню целевой конфигурации

В открывшемся окне, снимите галочку с опции «Use the same console for the CIO of all CPUs» (см. рисунок Б-2) и нажмите на кнопки «Apply» и «Continue».

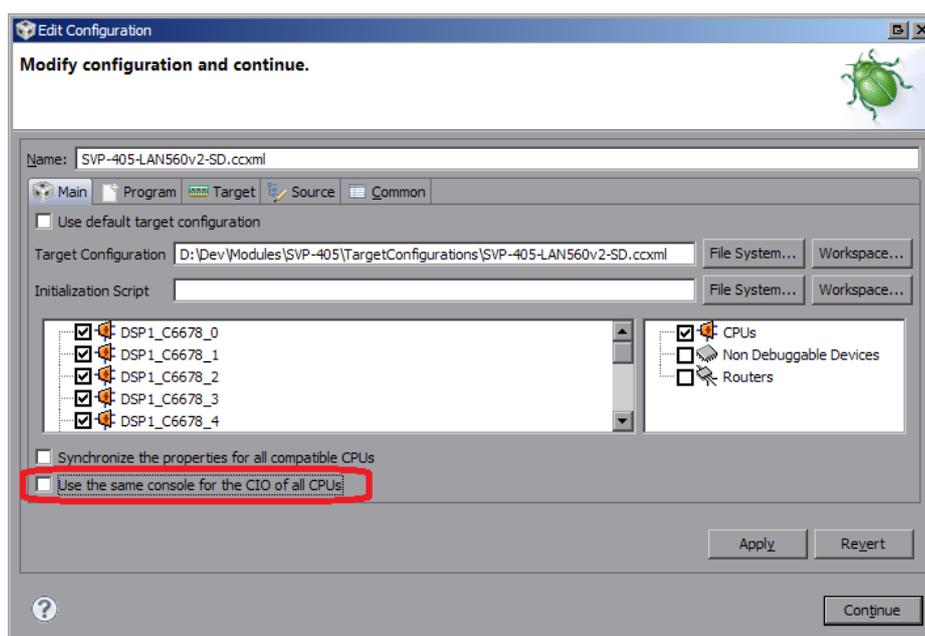


Рисунок Б-2: Окно настроек целевой конфигурации

В окне «Console» нажмите на кнопку  («Open Console») и выберите пункт меню «New Console View» (см. рисунок Б-3).

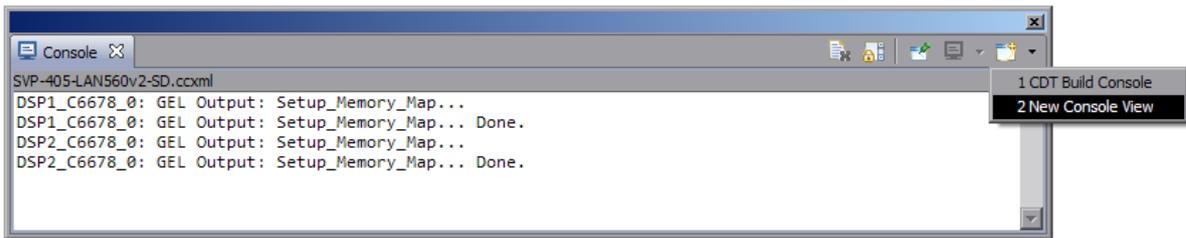


Рисунок Б-3: Открытие второго окна «Console»

После этого, будет открыто еще одно окно «Console», которое можно переместить в любое удобное место окна CCS, например, как показано на рисунке Б-4.

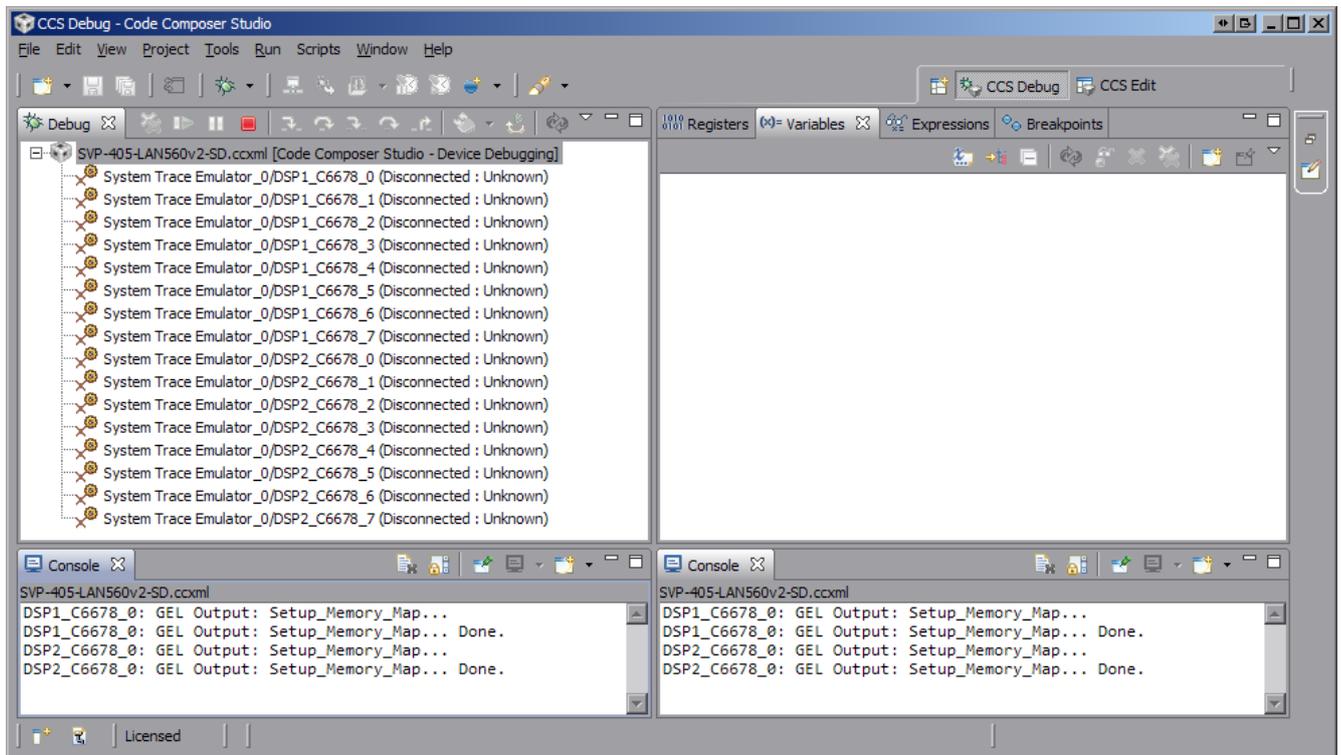


Рисунок Б-4: Два окна «Console»

Теперь, если запустить приложения на различных ядрах процессоров, для вывода с каждого отдельного ядра будет происходить в отдельное окно. Для того, чтобы выбрать вывод какого ядра необходимо отображать в конкретном окне «Console», необходимо нажать на кнопку  («Display Selected Console») этого окна и выбрать пункт меню соответствующий нужному ядру (см. рисунок Б-5).



Рисунок Б-5: Выбор ядра для отображения вывода в окне «Console»

Следует иметь ввиду, что в данном списке (рисунок Б-5) можно выбрать только те ядра, с которых уже был произведен какой либо вывод.

Например, если на ядре «DSP1_C6678_0» запустить простое приложение, выводящее сообщение «Message from application 1», а на ядре «DSP2_C6678_0» запустить второе приложение, выводящее сообщение «Message from application 2», то список доступных консолей будет соответствовать рисунку Б-5, а вывод в два окна будет выглядеть, как показано на рисунке Б-6.

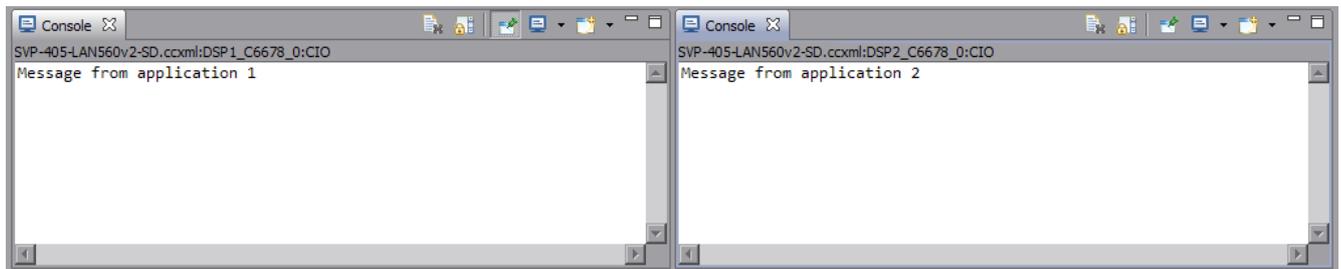


Рисунок Б-6: Вывод сообщений с двух ядер в два окна «Console»

Для закрепления окна «Console» за конкретным ядром используется кнопка  («Pin Console»).

Приложение В Выбор режима загрузки IBL

Выбор режима загрузки IBL осуществляется при помощи переключателей SW1–SW4 на плате модуля SVP-405.

В таблице В-1 представлены положения переключателей SW1–SW4 для всех возможных режимов загрузки IBL для каждого из двух процессоров модуля SVP-405.

Таблица В-1: Положение переключателей модуля SVP-405 для установки режимов загрузки IBL

Режим	DSP1_C6678		DSP2_C6678	
	SW1	SW2	SW3	SW4
NOBOOT (DEBUG)	Разомкнут	Разомкнут	Разомкнут	Разомкнут
NOR	Разомкнут	Замкнут	Разомкнут	Замкнут
TFTP (BOOTP)	Замкнут	Разомкнут	Замкнут	Разомкнут
PCI-E	Замкнут	Замкнут	Замкнут	Замкнут

Для режима NOBOOT устанавливаются значения $BOOTMODE[7:0] = 0000000b$ для соответствующего процессора. Для режима NOR устанавливаются значения $BOOTMODE[7:0] = 00000101b$ для соответствующего процессора. Для режима TFTP устанавливаются значения $BOOTMODE[7:0] = 00100101b$ для соответствующего процессора. Для режима PCI-E устанавливаются значения $BOOTMODE[7:0] = 00000100b$ и $PCIESSMODE[1:0] = 00b$ для соответствующего процессора.

Примечание

$BOOTMODE[7:0]$ является частью регистра $DEVSTAT[8:1]$, а $PCIESSMODE[1:0]$ является частью регистра $DEVSTAT[15:14]$. В процессоре TMS320C6678 32-х битный регистр $DEVSTAT$ доступен по адресу 0x02620020.

Список литературы

1. SVP-405. Сборка и запуск приложения веб-сервера. Руководство пользователя. [UG-SVP-405-WEB](#) (цит. на с. 13, 17).
2. Multicore Application Deployment (MAD) Utilities. User's Guide.
URL: http://processors.wiki.ti.com/index.php/MAD_Utils_User_Guide (цит. на с. 34, 35).
3. Установка и настройка сервера сетевой загрузки (BOOTP и TFTP). Руководство пользователя.
[UG-CMN-BOOTP-TFTP](#) (цит. на с. 38).