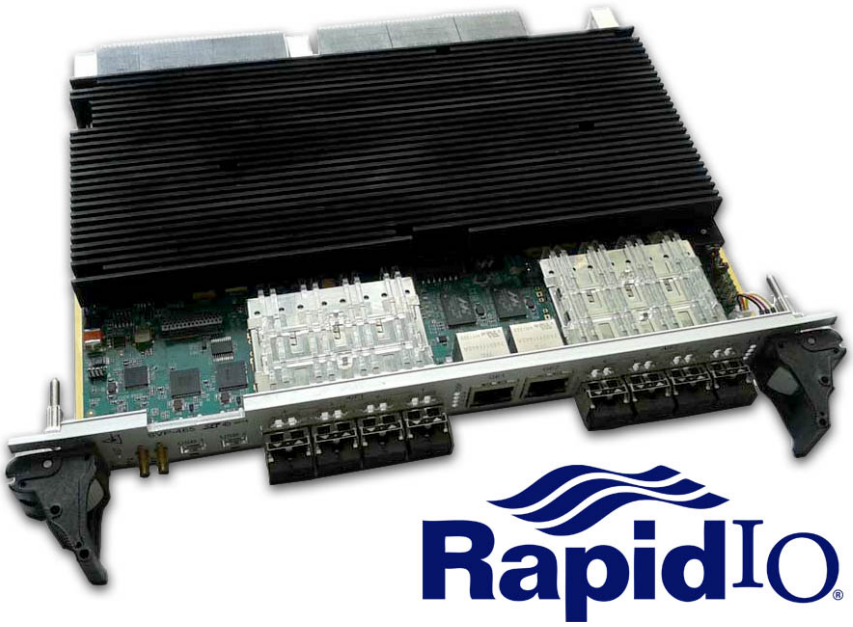
Scan Engineering Telecom SPb

SVP-465. Сборка и запуск теста производительности Serial RapidIO

Руководство пользователя

Версия 1.0



Код документа: UG-SVP-465-SRIO
Дата сборки: 27 мая 2015 г.
Листов в документе: 23

http://www.setdsp.ru

Содержание

Список рисунков

Список таблиц

Список листингов

Список процедур

Перечень сокращений и условных обозначений

CCS	Code Composer Studio	5–7, 15, 21, 22
CIO	Console Input/Output	2, 15, 21
EDMA	Enhanced Direct Memory Access	6
IPC	Inter Process Communication	6
LLD	Low Level Driver	6
PDK	Platform Development Kit	6
SRIO	Serial RapidIO	3, 5–7, 9, 10, 18, 19
TI	Texas Instruments	5
UART	Universal Asynchronous Receiver-Transmitter	19
ОС	Операционная Система	6

1 Общие сведения

В данном документе описан процесс сборки и запуска программы теста производительности передачи данных по шине SRIO (Serial RapidIO) между двумя любыми процессорами модуля SVP-465 через внешний SRIO коммутатор (см. рисунок 1-1).

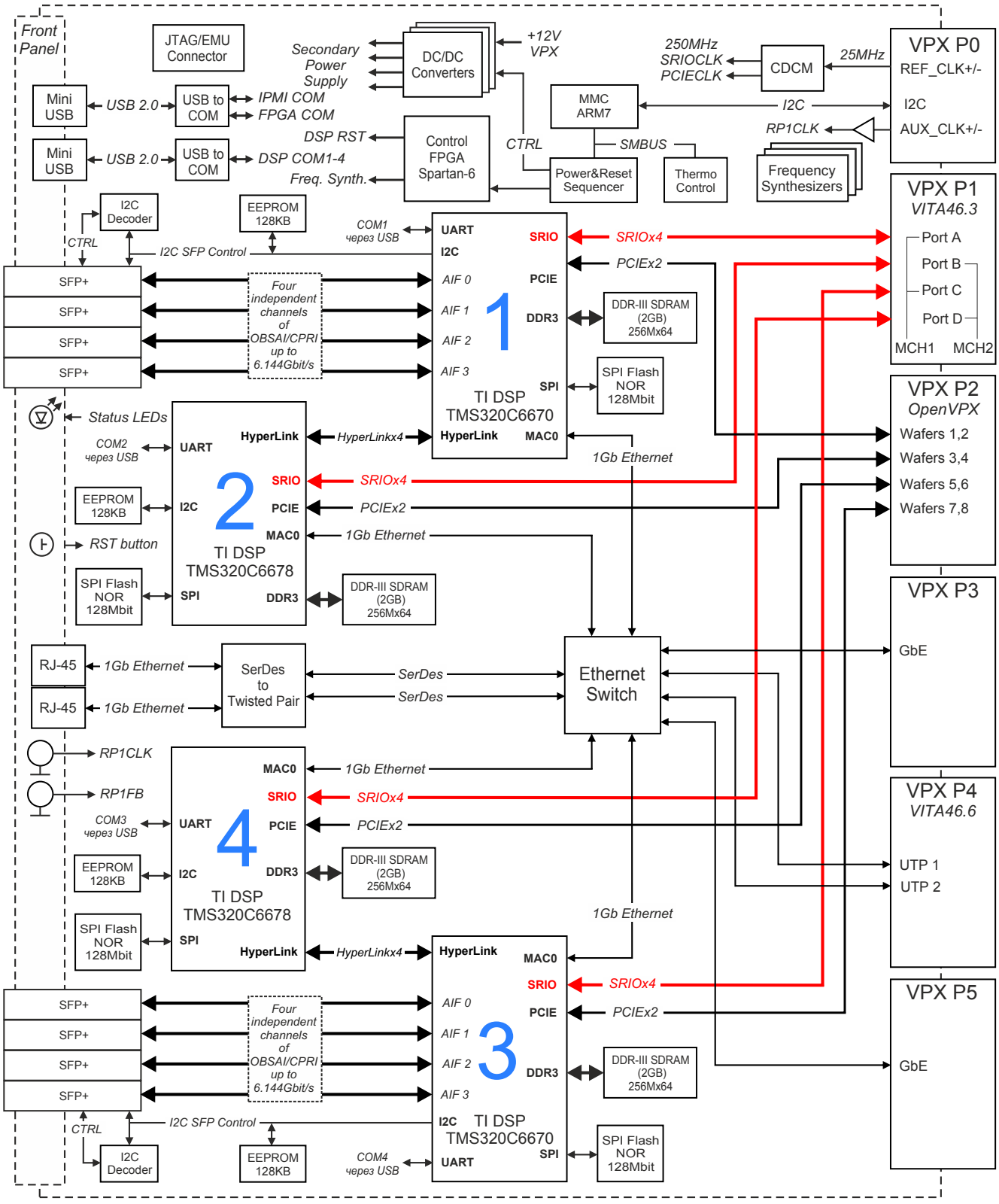


Рисунок 1-1: Структурная схема модуля SVP-465

Для сборки и запуска программы теста производительности SRIO требуется установленная система разработки CCS (Code Composer Studio) компании TI. Сборка и запуск теста производительности SRIO были

проверены на CCS версии 5.4.0.00091 установленной в ОС Windows 7 (64-bit). Среду разработки CCS можно бесплатно скачать с сайта производителя[1].

Тест производительности SRIO написан на основе кода теста, входящего в состав библиотеки TI PDK для процессоров TMS320C6678 и PDK для процессоров TMS320C6670.

Для сборки проекта теста производительности SRIO необходимы следующие установленные пакеты:

- PDK C6670 версии 1.1.2.6 (только для сборки проекта под TMS320C6670);
- PDK C6678 версии 1.1.2.6 (только для сборки проекта под TMS320C6678);
- IPC (Inter Process Communication) версии 1.24.3.32;
- EDMA3 LLD версии 2.11.5 (входит в состав PDK);
- SYS/BIOS версии 6.33.6.50.
- XDCtools версии 3.25.5.94 (установочный дистрибутив для Windows систем имеется на сопроводительном диске в папке «Install»).

В данном списке указаны версии пакетов, на которых производилось тестирование. Сборка проекта теста производительности SRIO возможна с использованием пакетов более ранних версий, однако, в этом случае, не гарантируется правильная работа теста проивзодительности SRIO.

Схема соединения двух процессоров модуля SVP-465 через коммутатор SVPS-205, рассматриваемая в данном документе, показана на рисунке 1-2. Программа теста производительности запускается на двух процессорах модуля SVP-465. На процессора «DSP1_C6670» запускается программа приемника данных (consumer), на процессора «DSP3_C6670» запускается программа генератора данных (producer). После запуска программ на обоих процессорах модуля SVP-465 программа генератор данных передает данные по шине SRIO программе приемнику данных. Во время передачи происходит замер скорости передачи данных и вывод результатов.

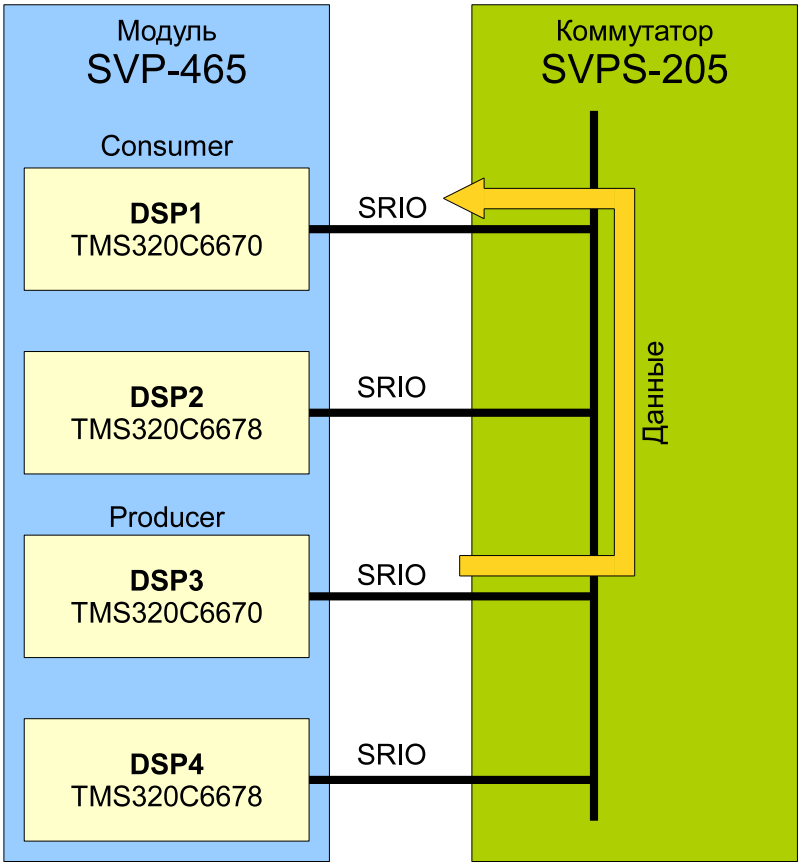


Рисунок 1-2: Схема соединения процессоров SVP-465 через коммутатор SVPS-205

[1] http://www.ti.com/tool/ccstudio

2 Конфигурация рабочего пространства

Перед началом сборки и запуска теста производительности SRIO необходимо с сопроводительного диска к модулю SVP-465 скопировать на жесткий диск компьютера папку рабочего пространства CCS (папка «CCS_Workspace» на диске). В данном документе предполагается, что содержимое папки «CCS_Workspace» было скопировано с сопроводительного диска на жесткий диск компьютера в папку «D:/Dev/Modules/SVP-465/CCS_Workspace».

Кроме папки рабочего пространства, необходимо скопировать с сопроводительного диска папки «TargetConfigurations», «RTSC» и «GEL» со всем содержимым. При этом, важно, чтобы обе данные папки находились на одном уровне. В данном документе предполагается, что содержимое папки «TargetConfigurations» скопировано в папку «D:/Dev/Modules/SVP-465/TargetConfigurations», содержимое папки «GEL» скопировано в папку «D:/Dev/Modules/SVP-465/GEL», а содержимое папки «RTSC» скопировано в папку «D:/Dev/Modules/SVP-465/RTSC».

После запуска среды разработки CCS необходимо указать путь к папке рабочего пространства как показано на рисунке 2-1 («D:/Dev/Modules/SVP-465/CCS_Workspace»).

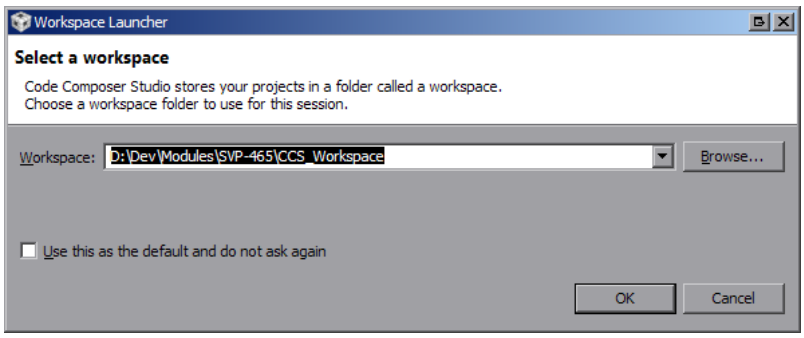
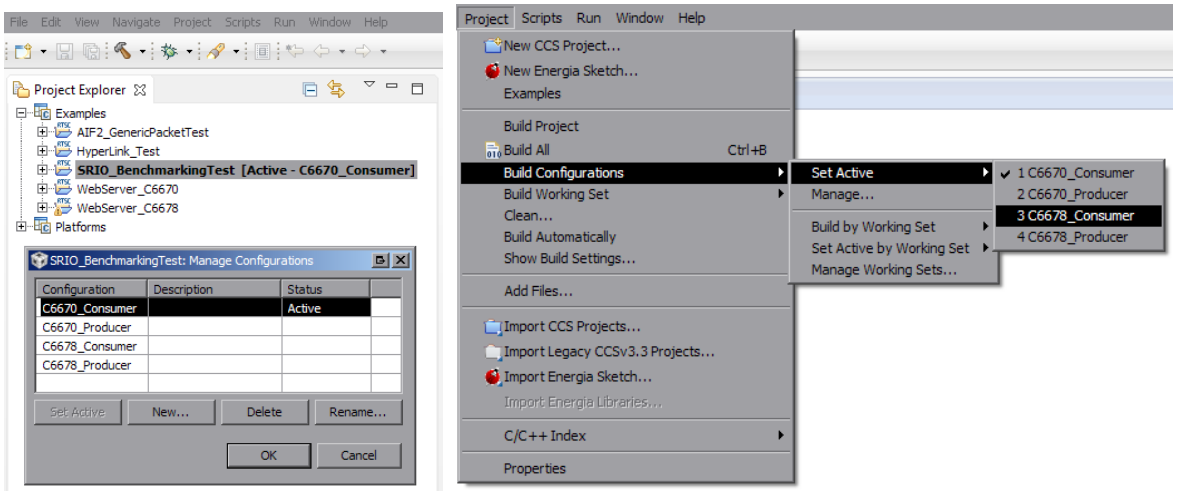


Рисунок 2-1: Выбор пути рабочего пространства (workspace) в CCS

Код теста производительности SRIO содержится в проекте «SRIO_BenchmarkingTest» (рисунок 2-2a). Данный проект имеет четрые конфигурации сборки (в нижней части рисунка 2-2a показано окно управления конфигурациями проекта):

- конфигурация «C6670_Consumer» — проект приемника данных для процессора TMS320C6670;
- конфигурация «C6670_Producer» — проект генератора данных для процессора TMS320C6670;
- конфигурация «C6678_Consumer» — проект приемника данных для процессора TMS320C6678;
- конфигурация «C6678_Producer» — проект генератора данных для процессора TMS320C6678.



а) Обозреватель проектов и окно конфигураций сборки проекта

б) Меню выбора активной конфигурации сборки проекта

Рисунок 2-2: Проект «SRIO_BenchmarkingTest» и его конфигурации сборки

Для выбора активной конфигурации сборки необходимо выбрать пункт главного меню «Project > Build Configurations > Set Active > *Имя_нужной_конфигурации_сборки*» (см. рисунок 2-2б).

3 Конфигурация проекта

Для применения выполненных изменений в конфигурации, необходимо выполнить сборку проекта в соответствии с процедурой, описанной в разделе 4.

3.1 Конфигурация скорости линий SRIO

Скорость линий SRIO устанавливается макроопределением «SRIO_LANE_SPEED» в файле «benchmarking.h» (строка 96) проекта теста производительности:

Листинг 3-1: Файл «benchmarking.h» (макроопределение «SRIO_LANE_SPEED»)

```
92 /* SRIO Lane Rate.(Please see
93  * srioLaneRateGbps_e enum above
94  * for valid values).
95  */
96 #define SRIO_LANE_SPEED                 srio_lane_rate_5p000Gbps
```

Возможные значения для макроопределнеия «SRIO_LANE_SPEED» приведены в таблице 3-1.

Таблица 3-1: Возможные значения макроопределения «SRIO_LANE_SPEED»

Значение	Скорость
srio_lane_rate_1p250Gbps	1.25 Гбит/с
srio_lane_rate_2p500Gbps	2.5 Гбит/с
srio_lane_rate_3p125Gbps	3.125 Гбит/с
srio_lane_rate_5p000Gbps	5.0 Гбит/с (значение по умолчанию)

3.2 Конфигурация режима линий SRIO

Режим линий SRIO устанавливается макроопределением «SRIO_PORT_WIDTH» в файле «benchmarking.h» (строка 102) проекта теста производительности:

Листинг 3-2: Файл «benchmarking.h» (макроопределение «SRIO_PORT_WIDTH»)

```
98  /* SRIO Port Width.(Please see
99   * srioLanesMode_e enum above
100  * for valid values).
101  */
102 #define SRIO_PORT_WIDTH                 srio_lanes_form_one_4x_port
```

Возможные значения для макроопределнеия «SRIO_PORT_WIDTH» приведены в таблице 3-2.

Таблица 3-2: Возможные значения макроопределения «SRIO_PORT_WIDTH»

Значение	Режим
srio_lanes_form_four_1x_ports	Четыре 1x порта
srio_lanes_form_2x_port_and_two_1x_ports	Один 2x порт и два 1x порта
srio_lanes_form_1x_port_and_one_2x_ports	Два 1x порта и один 2x порт
srio_lanes_form_2x_ports	Два 2x порта
srio_lanes_form_4x_port	Один 4x порт (значение по умолчанию)

3.3 Конфигурация SRIO идентификаторов устройств

Так как маршрутизация SRIO осуществляется основываясь на уникальных идентификаторах устройств, то для правильной работы теста производительности необходимо их правильно сконфигурировать.

В случае использования коммутатора SVPS-205, идентификатор устройства будет зависеть от того, в какой слот шасси это устройство будет установлено. Соответствие идентификаторов SRIO с номерами слота крейта приведено в таблице 3-3.

Таблица 3-3: Соответствие SRIO идентификаторов и слотов крейта при использовании коммутатора SVPS-205

Номер слота	**Идентификаторы SRIO для процессоров SVP-465**			
	DSP1_C6670	**DSP2_C6678**	**DSP3_C6670**	**DSP4_C6678**
1	0x10	0x11	0x12	0x13
2	0x20	0x21	0x22	0x23
3	0x30	0x31	0x32	0x33
4	0x40	0x41	0x42	0x43
5	*Коммутатор SVPS-205*		—	
6	—		*Коммутатор SVPS-205*	
7	0x70	0x71	0x72	0x73
8	0x80	0x81	0x82	0x83
9	0x90	0x91	0x92	0x93
10	0xA0	0xA1	0xA2	0xA3

Идентификаторы для генератора и приемника данных теста производительности SRIO устанавливается макроопределениями «PRODUCER_8BIT_DEVICE_ID» и «CONSUMER_8BIT_DEVICE_ID». Данные макроопределния описаны в файле «benchmarking.h» (строки 175 и 180) проекта теста производительности:

Листинг 3-3: Файл «benchmarking.h» (макроопределения «PRODUCER_8BIT_DEVICE_ID» и «CONSUMER_8BIT_DEVICE_ID»)

```
172 /* Producer Device Identifiers.
173  */
174 #define PRODUCER_16BIT_DEVICE_ID          0xFFFF
175 #define PRODUCER_8BIT_DEVICE_ID           0x32
176
177 /* Consumer Device Identifiers.
178  */
179 #define CONSUMER_16BIT_DEVICE_ID          0xFFFF
180 #define CONSUMER_8BIT_DEVICE_ID           0x30
```

Макроопределение «PRODUCER_8BIT_DEVICE_ID» задает идентификатор для устройства (процессора) на котором будет запускаться код генератора данных. Макроопределение «CONSUMER_8BIT_DEVICE_ID» задает идентификатор для устройства (процессора) на котором будет запускаться код приемника данных.

4 Сборка проекта теста производительности

Для сборки проекта нажмите правой кнопой мыши по названию проекта «SRIO_BenchmarkingTest» в окне обозревателя проектов («Project Explorer») для вызова контекстного меню и выберите пункт меню «Build Configurations > Build Selected...» (рисунок 4-1).

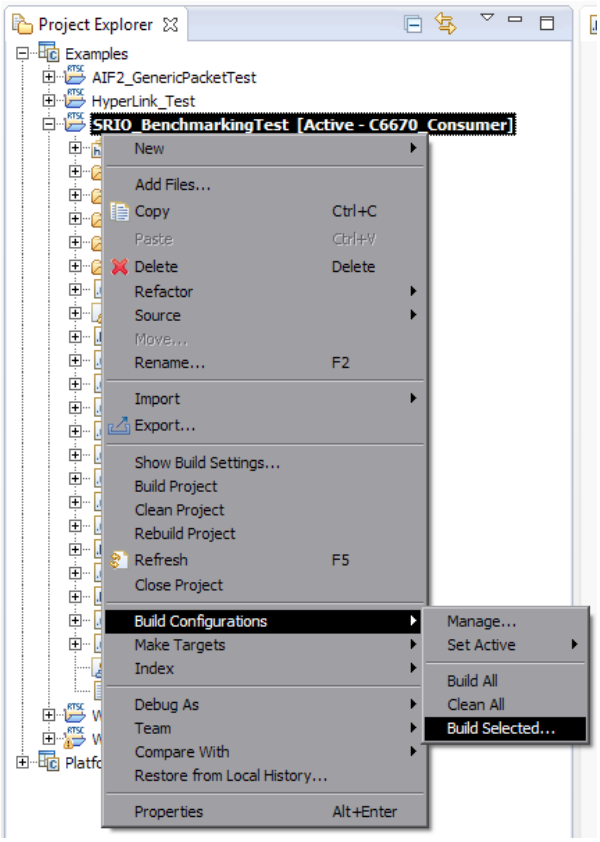


Рисунок 4-1: Пункт меню для сборки проекта

В открывшемся окне (рисунок 4-2) нужно отметить конфигурации сборки, для которых необходимо выполнить сборку и нажать на кнопку «OK».

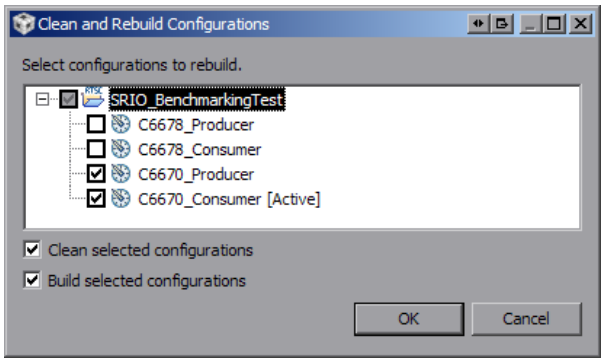


Рисунок 4-2: Окно выбора конфигураций для сборки

Проект «SRIO_BenchmarkingTest» имеет четыре конфигурации сборки:

- конфигурация «C6670_Consumer» — проект приемника данных для процессора TMS320C6670;
- конфигурация «C6670_Producer» — проект генератора данных для процессора TMS320C6670;
- конфигурация «C6678_Consumer» — проект приемника данных для процессора TMS320C6678;
- конфигурация «C6678_Producer» — проект генератора данных для процессора TMS320C6678.

Например, если генератор данных будет запускаться на процессоре TMS320C6678, а приемник данных на процессоре TMS320C6670, то необходимо выполнить сборку конфигураций «C6678_Producer» и «C6670_-Consumer».

После нажатия на кнопку «OK» будет запущен процесс сборки выбранных конфигураций сборки. В случае успешной сборки, в окне консоли (Console) должны быть выведены сообщения, идентичные показанным на рисунке 4-3.

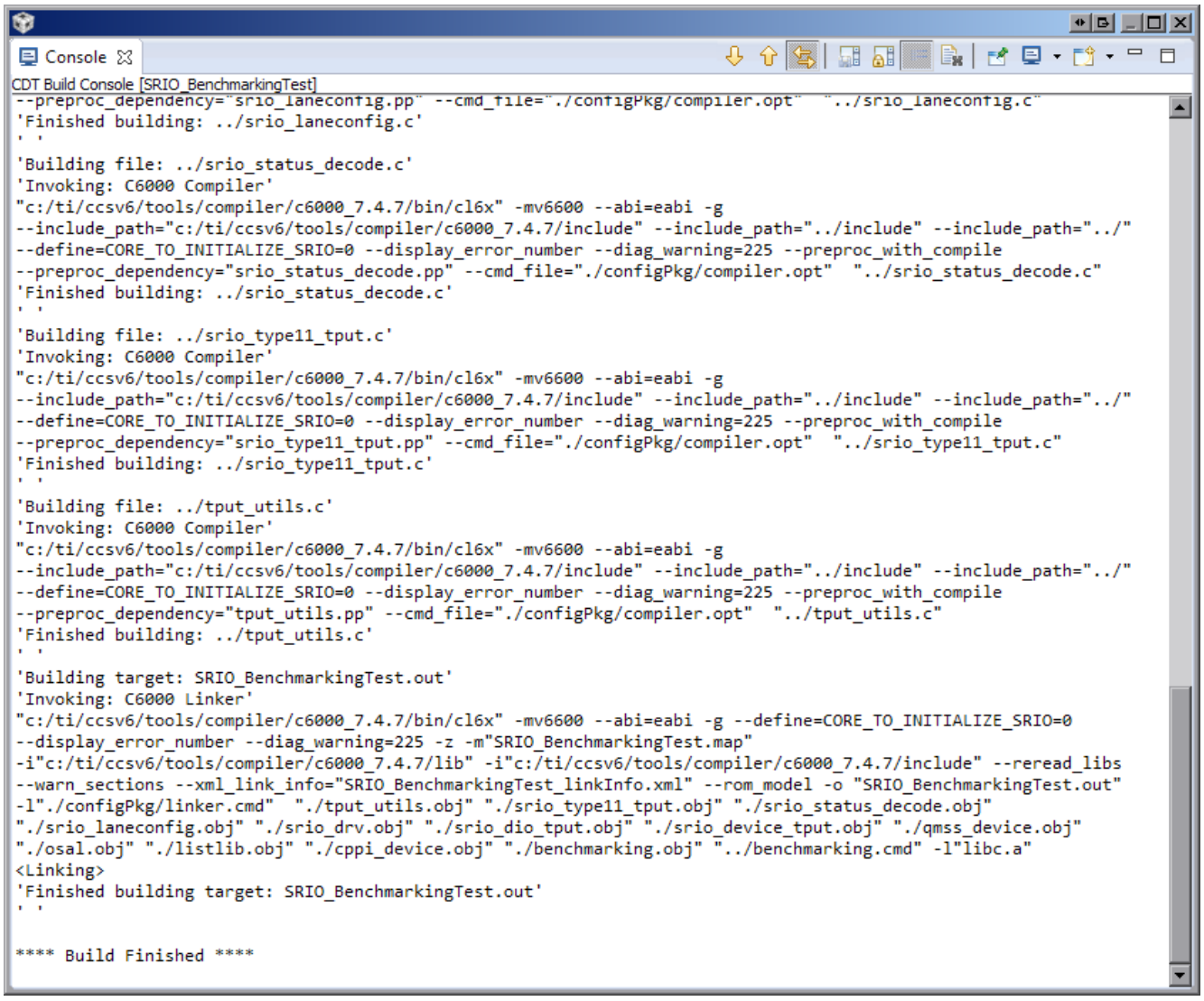


Рисунок 4-3: Вывод процесса сборки в окно «Console»

5 Импорт и запуск целевой конфигурации модуля

Для загрузки кода приложениий на модуль SVP-465, в первую очередь, необходимо запустить целевую конфигурацию модуля SVP-465. В папке «TargetConfigurations» сопроводительного диска к модулю SVP-465 находятся файлы целевых конфигураций для различных отладчиков. В данном документе рассматривается загрузка кода через отладчик Blackhawk USB560 v2 System Trace. Данному отладчику соответствует файл целевой конфигурации «SVP-465-USB560v2.ccxml», который необходимо импортировать в рабочее пространство.

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

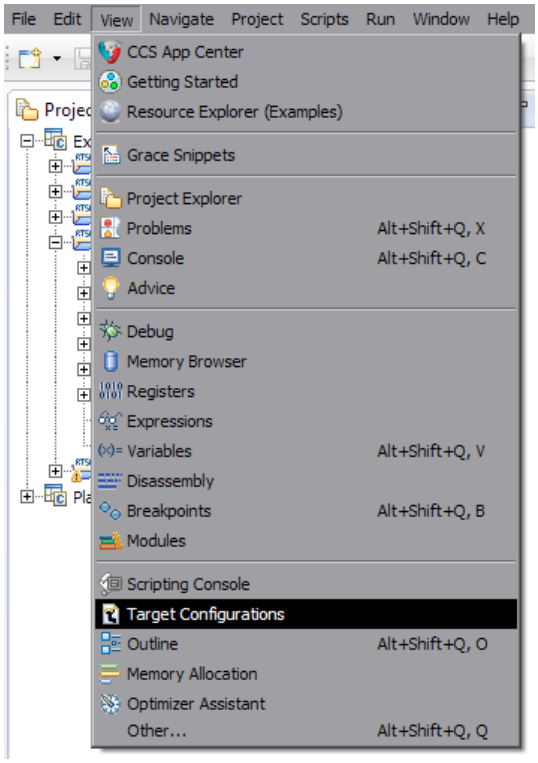


Рисунок 5-1: Пункт меню для отображения окна целевых конфигураций

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

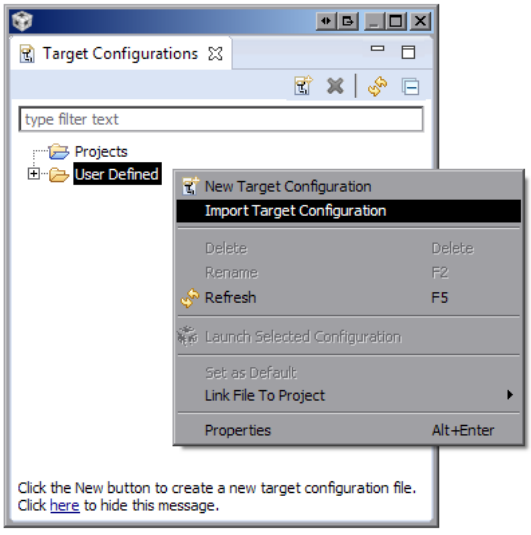


Рисунок 5-2: Меню импорта целевой конфигурации

В появившемся окне выбора файла (см. рисунок 5-3) необходимо выбрать файл «SVP-465-USB560v2.ccxml» и нажать на кнопку «Открыть». В данном документе предполагается, что папка «TargetConfigurations» с сопроводительного диска к модулю SVP-465, где расположен файл «SVP-465-USB560v2.ccxml», скопирована в папку «D:/Dev/Modules/SVP-465/TargetConfigurations».

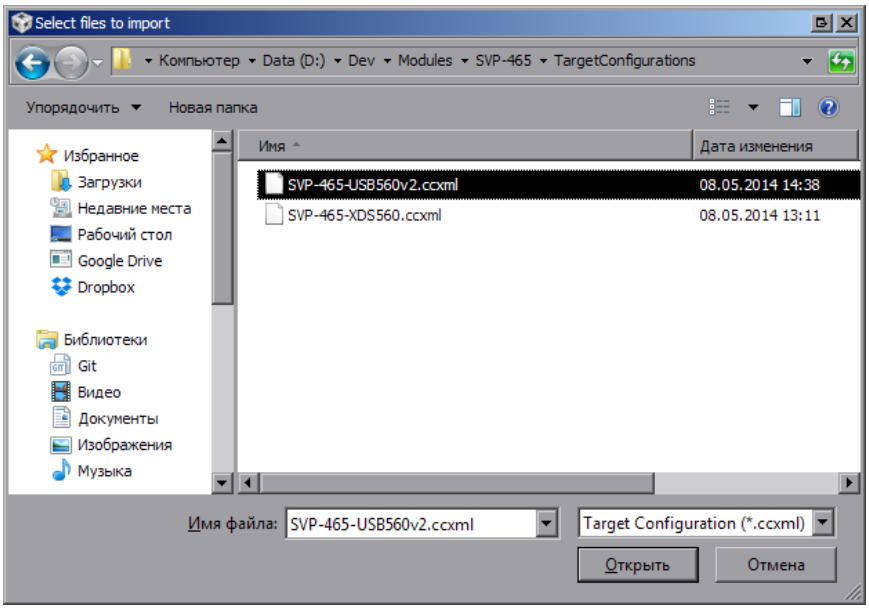


Рисунок 5-3: Окно выбора файла для импорта целевой конфигурации

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 «OK».

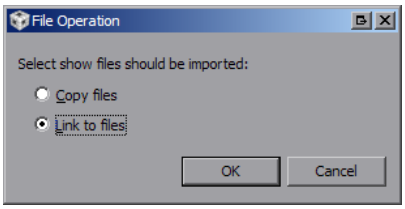


Рисунок 5-4: Окно выбора способа импорта файла целевой конфигурации

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

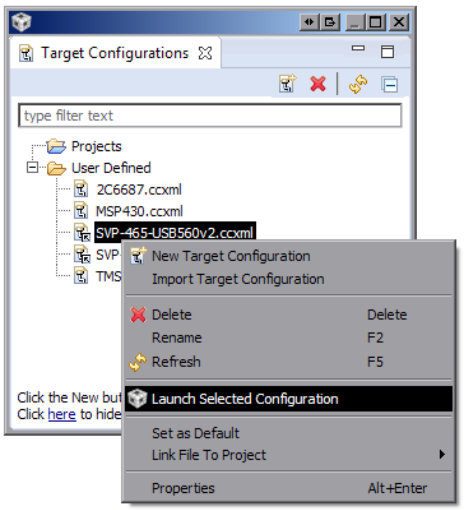


Рисунок 5-5: Запуск целевой конфигурации

После запуска целевой конфигурации модуля SVP-465, среда разработки CCS перейдет в режим отладки и в окне «Debug» будет выведен список ядер всех четырех процессоров модуля SVP-465, как показано на рисунке 5-6.

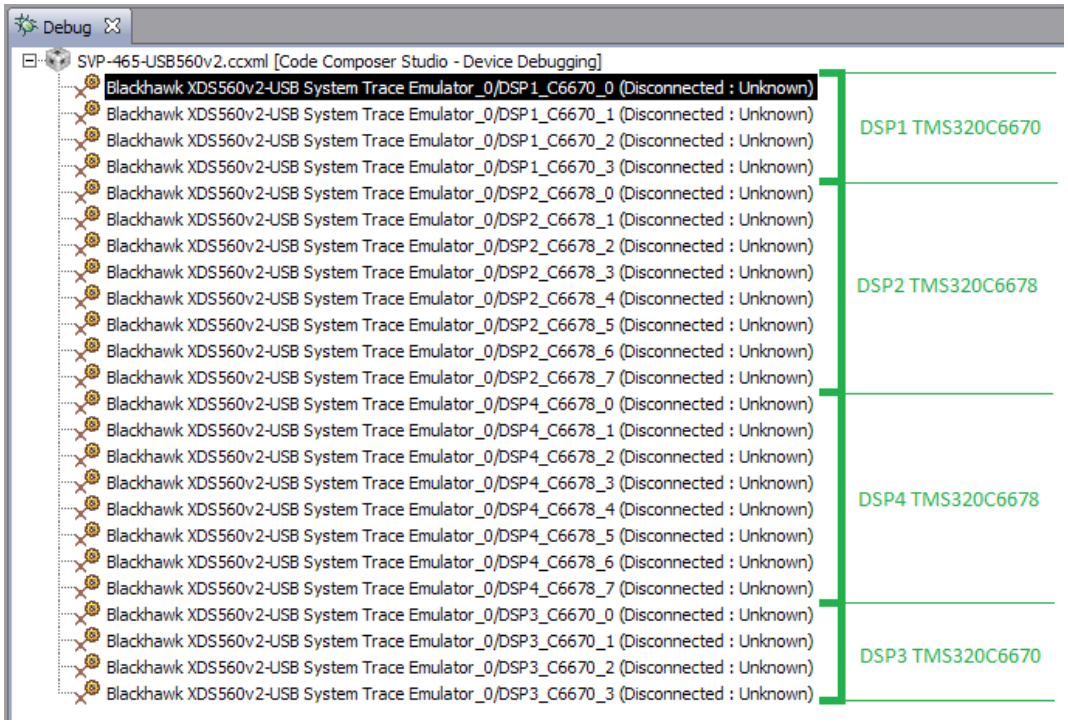


Рисунок 5-6: Список ядер процессоров модуля SVP-465

Примечание

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 приложении A даны инструкции по настройке раздельного вывода (CIO) каждого ядра процессора в отдельное окно «Console».

6 Загрузка кода

Запустите целевую конфигурацию модуля SVP-465, как описано в разделе 5.

В данном документе предполагается, что процессор «DSP1_C6670» является приемником (consumer) данных, а процессор «DSP3_C6670» является генератором (producer) данных.

6.1 Подключение к ядрам процессоров

Щелкните правой кнопкой мыши на названии ядра «DSP1_C6670_0» и выберите пункт меню «Connect Target» для подключения к ядру «DSP1_C6670_0». После выполнения подключения к ядру «DSP1_C6670_0», окно «Debug» должно выглядеть, как показано на рисунке 6-1.

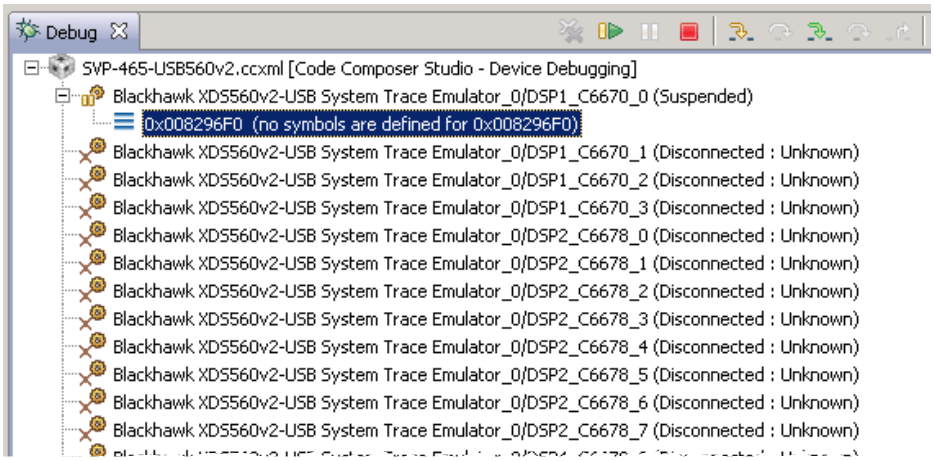


Рисунок 6-1: Окно «Debug» после выполнения подключения к ядру «DSP1_C6670_0»

Выберите ядра «DSP3_C6670_0» и «DSP3_C6670_1» в окне «Debug» (щелкните левой кнопкой мыши на ядре «DSP3_C6670_0», затем, зажав на клавиатуре клавишу Ctrl, щелкните на ядре «DSP3_C6670_1»). Щелкните правой кнопкой мыши на последнем выбранном ядре, и выберите пункт меню «Group core(s)».

Щелкните правой кнопкой мыши на названии группы «Group 1» и выберите пункт меню «Connect Target» для подключения к адрам «DSP3_C6670_0» и «DSP3_C6670_1». После выполнения подключения к ядрам процессоров, окно «Debug» должно выглядеть, как показано на рисунке 6-2.

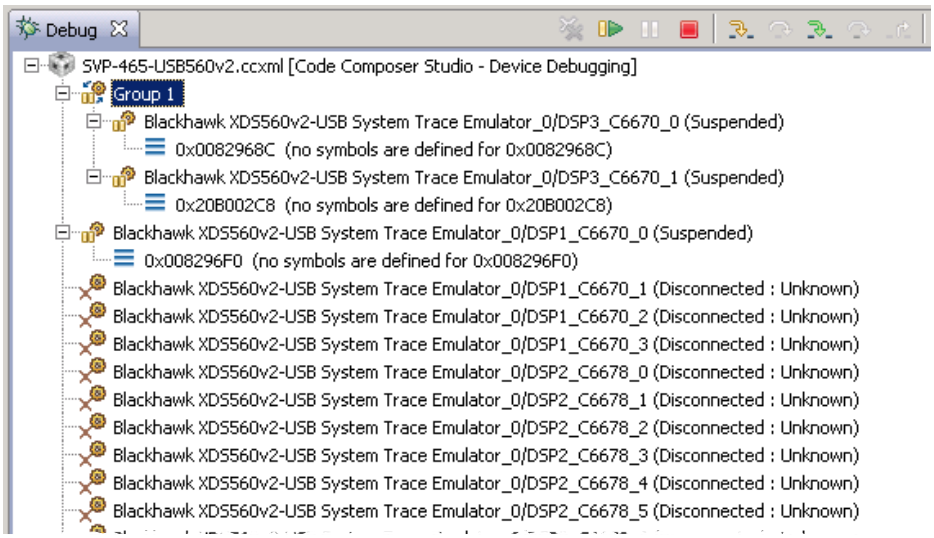


Рисунок 6-2: Окно «Debug» после выполнения подключения к ядрам «DSP3_C6670_0» и «DSP3_C6670_1»

6.2 Загрузка кода на ядра процессоров

Выберите ядро «DSP1_C6670_0» в окне «Debug» (щелкните левой кнопкой мыши по названию ядра). Выберите пункт главного меню «Run > Load > Load Program...» (рисунок 6-3).

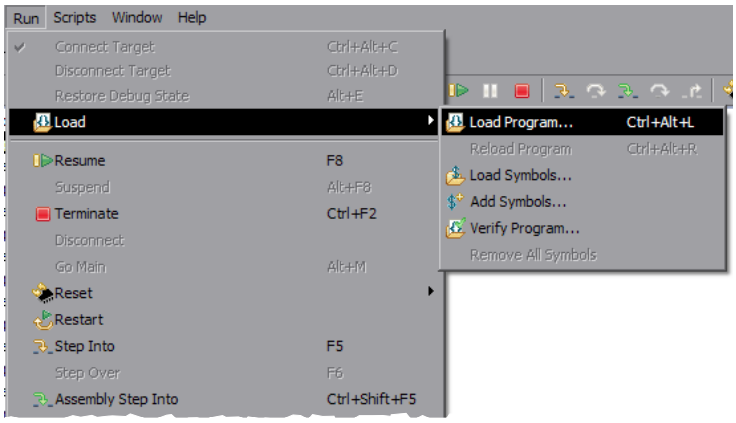
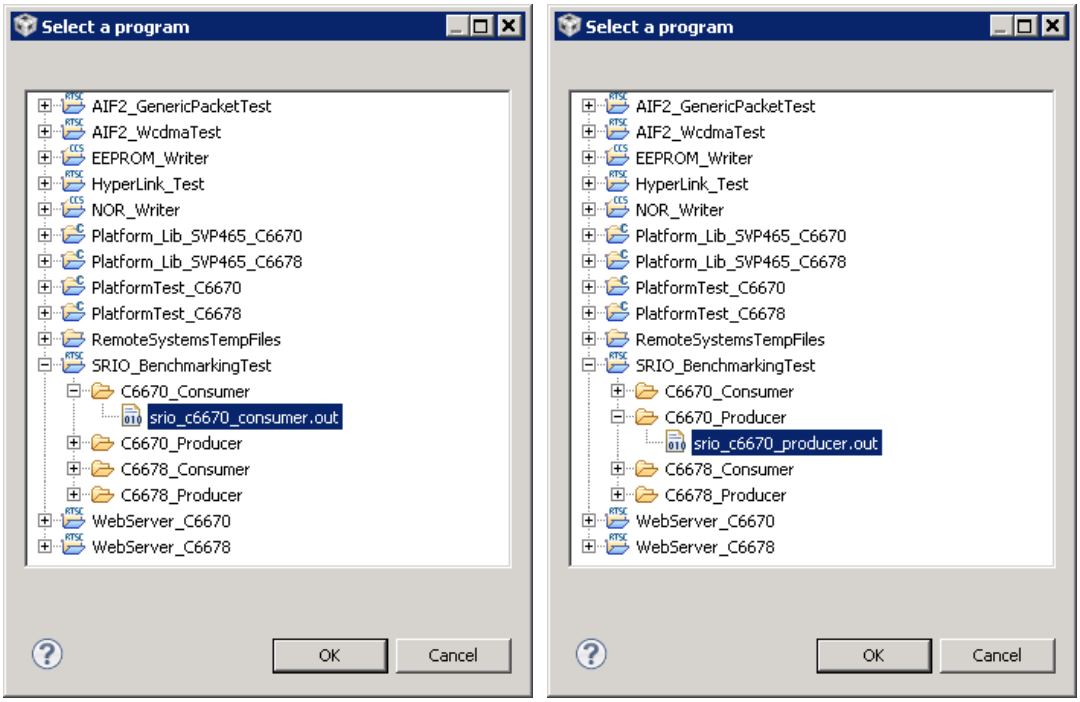


Рисунок 6-3: Меню загрузки кода на ядро процессора

В открывшемся окне нажмите на кнопку «Browse project...» и выберите файл «srio_c6670_consumer.out» из конфигурации сборки «C6670_Consumer» проекта «SRIO_BenchmarkingTest», как показано на рисунке 6-4а, и нажмите на кнопку «OK».



а) Приемник данных (consumer) б) Генератор данных (producer)

Рисунок 6-4: Окно выбора файла для загрузки на ядро процессора

Выберите ядро «DSP3_C6670_1» в окне «Debug» и аналогичным образом выполните загрузку файла «srio_c6670_producer.out», но из конфигурации сборки «C6670_Producer» (см. рисунок 6-4б).

Внимание

Загрузка кода приемника (consumer) данных выполняется на первое ядро процессора («DSP1_C6670_0»), а загрузка кода генератора (producer) данных должна выполняться обязательно на второе ядро процессора («DSP3_C6670_1»).

После загрузки кода теста SRIO на ядра процессоров, окно «Debug» должно выглядеть, как показано на рисунке 6-5.

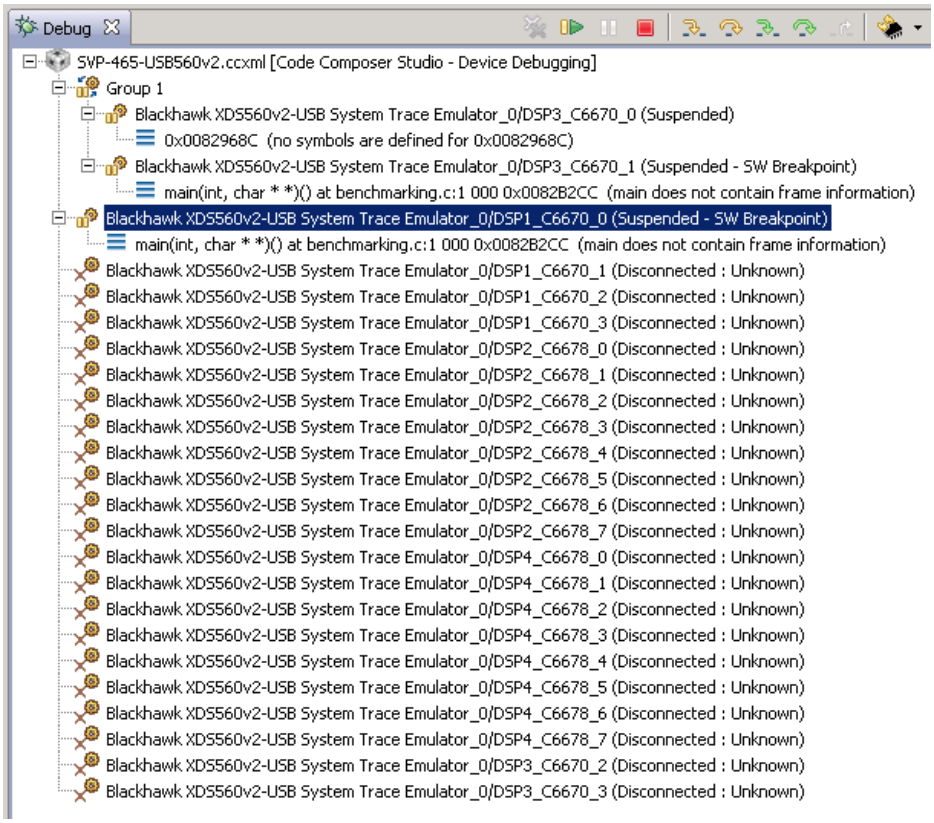


Рисунок 6-5: Внешний вид окна «Debug» после загрузки кода на ядра процессоров

7 Запуск теста

Для запуска теста SRIO выполните шаги, описанные в процедуре 7-1.

Процедура 7-1. Запуск теста SRIO

1. Выполнить загрузку кода теста SRIO на ядра процессоров, как описано в разделе 6.
2. Выбрать в окне «Debug» ядро приемника (consumer) данных («DSP1_C6670_0») щелкнув по нему левой кнопкой мыши.
3. Запустить выполнение кода приемника данных, нажав на кнопку  («Resume»). На рисунке 6-5 кнопка «Resume» расположена в верхнем правом углу.
4. Выбрать в окне «Debug» ядро генератора (producer) данных («DSP3_C6670_1») щелкнув по нему левой кнопкой мыши.
5. Запустить выполнение кода генератора данных, нажав на кнопку  («Resume»).

Важная информация

Запускать код приемника данных необходимо обязательно до того, как будет запущен код передатчика данных. Запускать код передатчика данных можно только после того, как в консоль приемника данных будет выведена строка: «`Debug: SRIO Driver handle 0x861d00`» (значение адреса может отличаться).

В случае успешного запуска теста SRIO, вывод приемника данных (ядро «DSP1_C6670_0») в окно «Console» должен выглядеть аналогично приведенному в листинге 7-1, а вывод с генератора данных (ядро «DSP3_C6670_1») должен выглядеть аналогично приведенному в листинге 7-2.

Примечание

В приложении A приведена информация по настройке раздельного вывода с ядер процессоров. В том случае, если не будет выполнена настройка раздельного вывода с ядер процессоров, сообщения с обоих ядер будут выводиться в одном общем окне «Console».

Весь вывод теста SRIO дублируется в UART для каждого процессора.

Листинг 7-1: Вывод приемника данных с ядра «DSP1_C6670_0»

```
Device MAC address: 00:17:ea:ed:d0:d3
Build: Jun 23 2014, 11:48:42
SRIO lanes: 1p4x
SRIO lane speed: 5.0 Gbps
SRIO lane reversed version (mode = 0x02)
SRIO 8-bit ID's: consumer = 0x30, producer = 0x32
*******************************
*********** CONSUMER ***********
*******************************
WARNING: Please ensure that the CONSUMER is executing before running the PRODUCER!!
Debug: Waiting for module reset...
Debug: Waiting for module local reset...
Debug: Waiting for SRIO ports to be operational...
Debug: SRIO port 0 is operational.
Debug:   Lanes status shows lanes formed as one 4x port
Debug: AppConfig Tx Queue: 0x2a0 Flow Id: 0
Debug: SRIO Driver Instance 0x@00841900 has been created
Debug: Running test in polled mode.
Debug: SRIO Driver handle 0x841900.


Throughput: (RX side, DIO_NW, 5.000GBaud, 4X, tab delimited)
Core    Lanes   Speed   Conn    MsgType OHBytes PktSize Pacing  Thruput     PktsSec.    NumPkts     PktLoss AgPCycs AgLCycs AgICycs AgOCycs Seconds
0       4       5.000   B-S-B   DIO_NW  16      4       0       85.48       2671195.75  12600000    No      368     21      347     0       4.73
0       4       5.000   B-S-B   DIO_NW  16      8       0       170.96      2671195.75  12600000    No      368     21      347     0       4.73
0       4       5.000   B-S-B   DIO_NW  16      16      0       341.91      2671195.75  12600000    No      368     21      347     0       4.73
0       4       5.000   B-S-B   DIO_NW  16      32      0       683.83      2671195.75  12600000    No      368     21      347     0       4.73
0       4       5.000   B-S-B   DIO_NW  16      64      0       1367.65     2671195.75  12600000    No      368     21      347     0       4.73
0       4       5.000   B-S-B   DIO_NW  16      128     0       2735.30     2671195.75  12600000    No      368     21      347     0       4.73
0       4       5.000   B-S-B   DIO_NW  16      256     0       5470.61     2671195.75  12600000    No      368     21      347     0       4.73
0       4       5.000   B-S-B   DIO_NW  16      512     0       10941.22    2671195.75  12600000    No      368     21      347     0       4.73
0       4       5.000   B-S-B   DIO_NW  16      1024    0       13466.11    1643812.75  7800000     No      598     21      577     0       4.75
0       4       5.000   B-S-B   DIO_NW  16      2048    0       13387.76    817123.88   4000000     No      1203    21      1182    0       4.90
0       4       5.000   B-S-B   DIO_NW  16      4096    0       13421.23    409583.34   2200000     No      2400    21      2379    0       5.37
0       4       5.000   B-S-B   DIO_NW  16      8192    0       13435.22    205005.22   1200000     No      4795    21      4774    0       5.85
```

Листинг 7-2: Вывод генератора данных с ядра «DSP3_C6670_1»

```
Device MAC address: 00:17:ea:d5:e2:d4
Build: Jun 23 2014, 11:49:53
SRIO lanes: 1p4x
SRIO lane speed: 5.0 Gbps
SRIO lane reversed version (mode = 0x02)
SRIO 8-bit ID's: consumer = 0x30, producer = 0x32
*********************************
*********** PRODUCER ************
*********************************
WARNING: Please ensure that the CONSUMER is executing before running the PRODUCER!!
Debug: Waiting for module reset...
Debug: Waiting for module local reset...
Debug: Waiting for SRIO ports to be operational...
Debug: SRIO port 0 is operational.
Debug:   Lanes status shows lanes formed as one 4x port
Debug: AppConfig Tx Queue: 0x2a0 Flow Id: 0
Debug: SRIO Driver Instance 0x@00841810 has been created
Debug: Running test in polled mode.
Debug: SRIO Driver handle 0x841810.


Latency: (DIO_NW, 5.000GBaud, 4X, tab delimited)
Core    Lanes   Speed   Conn    MsgType PktSize NumPkts MnLCycs AgLCycs MxLCycs
1       4       5.000   B-S-B   DIO_NW  4       100     863     842     843
1       4       5.000   B-S-B   DIO_NW  8       100     863     865     881
1       4       5.000   B-S-B   DIO_NW  16      100     883     884     903
1       4       5.000   B-S-B   DIO_NW  32      100     917     928     935
1       4       5.000   B-S-B   DIO_NW  64      100     989     1005    1020
1       4       5.000   B-S-B   DIO_NW  128     100     1151    1156    1169
1       4       5.000   B-S-B   DIO_NW  256     100     1457    1460    1475
1       4       5.000   B-S-B   DIO_NW  512     100     1601    1616    1636
1       4       5.000   B-S-B   DIO_NW  1024    100     1907    1919    1942
1       4       5.000   B-S-B   DIO_NW  2048    100     2519    2525    2538
1       4       5.000   B-S-B   DIO_NW  4096    100     3743    3743    3760
1       4       5.000   B-S-B   DIO_NW  8192    100     6172    6173    6191

Throughput: (TX side, DIO_NW, 5.000GBaud, 4X, tab delimited)
Core    Lanes   Speed   Conn    MsgType OHBytes PktSize Pacing  Thruput     PktsSec.    NumPkts     PktLoss AgPCycs AgLCycs AgICycs AgOCycs Seconds
1       4       5.000   B-S-B   DIO_NW  16      4       0       85.33       2666666.75  12600000    No      375     319     41      15      4.73
1       4       5.000   B-S-B   DIO_NW  16      8       0       170.67      2666666.75  12600000    No      375     319     41      15      4.73
1       4       5.000   B-S-B   DIO_NW  16      16      0       341.33      2666666.75  12600000    No      375     319     41      15      4.73
1       4       5.000   B-S-B   DIO_NW  16      32      0       682.67      2666666.75  12600000    No      375     319     41      15      4.73
1       4       5.000   B-S-B   DIO_NW  16      64      0       1365.33     2666666.75  12600000    No      375     319     41      15      4.73
1       4       5.000   B-S-B   DIO_NW  16      128     0       2730.67     2666666.75  12600000    No      375     319     41      15      4.73
1       4       5.000   B-S-B   DIO_NW  16      256     0       5461.33     2666666.75  12600000    No      375     319     41      15      4.73
1       4       5.000   B-S-B   DIO_NW  16      512     0       10922.67    2666666.75  12600000    No      375     319     41      15      4.73
1       4       5.000   B-S-B   DIO_NW  16      1024    200     13451.56    1642036.13  7800000     No      609     319     275     15      4.75
1       4       5.000   B-S-B   DIO_NW  16      2048    816     13385.62    816993.44   4000000     No      1224    318     891     15      4.90
1       4       5.000   B-S-B   DIO_NW  16      4096    2033    13418.51    409500.41   2200000     No      2442    318     2109    15      5.37
1       4       5.000   B-S-B   DIO_NW  16      8192    4469    13435.01    205002.05   1200000     No      4878    318     4545    15      5.85

Latency: (DIO_NR, 5.000GBaud, 4X, tab delimited)
Core    Lanes   Speed   Conn    MsgType PktSize NumPkts MnLCycs AgLCycs MxLCycs
1       4       5.000   B-S-B   DIO_NR  4       100     1238    1271    1338
1       4       5.000   B-S-B   DIO_NR  8       100     1238    1272    1338
1       4       5.000   B-S-B   DIO_NR  16      100     1238    1274    1338
1       4       5.000   B-S-B   DIO_NR  32      100     1238    1325    1343
1       4       5.000   B-S-B   DIO_NR  64      100     1238    1397    1414
1       4       5.000   B-S-B   DIO_NR  128     100     1238    1553    1588
1       4       5.000   B-S-B   DIO_NR  256     100     1238    1868    1901
1       4       5.000   B-S-B   DIO_NR  512     100     1238    2015    2044
1       4       5.000   B-S-B   DIO_NR  1024    100     1238    2324    2358
1       4       5.000   B-S-B   DIO_NR  2048    100     1238    2930    2954
1       4       5.000   B-S-B   DIO_NR  4096    100     1238    4143    4177
1       4       5.000   B-S-B   DIO_NR  8192    100     1238    6587    6593

Throughput: (TX side, DIO_NR, 5.000GBaud, 4X, tab delimited)
Core    Lanes   Speed   Conn    MsgType OHBytes PktSize Pacing  Thruput     PktsSec.    NumPkts PktLoss AgPCycs AgLCycs AgICycs AgOCycs Seconds
1       4       5.000   B-S-B   DIO_NR  28      4       0       30.74       960614.81   4800000 No      1041    317     709     15      5.00
1       4       5.000   B-S-B   DIO_NR  28      8       0       61.48       960614.81   4800000 No      1041    317     709     15      5.00
1       4       5.000   B-S-B   DIO_NR  28      16      0       122.14      954198.50   4800000 No      1048    317     716     15      5.03
1       4       5.000   B-S-B   DIO_NR  28      32      0       233.15      910746.81   4600000 No      1098    318     765     15      5.05
1       4       5.000   B-S-B   DIO_NR  28      64      0       437.23      853970.94   4200000 No      1171    317     839     15      4.92
1       4       5.000   B-S-B   DIO_NR  28      128     0       768.19      750187.56   3800000 No      1333    317     1001    15      5.07
1       4       5.000   B-S-B   DIO_NR  28      256     0       1249.54     610128.13   3200000 No      1639    318     1306    15      5.25
1       4       5.000   B-S-B   DIO_NR  28      512     0       2284.44     557724.50   2800000 No      1793    318     1460    15      5.02
1       4       5.000   B-S-B   DIO_NR  28      1024    0       3910.26     477326.97   2400000 No      2095    318     1762    15      5.03
1       4       5.000   B-S-B   DIO_NR  28      2048    0       6061.41     369959.31   2000000 No      2703    318     2370    15      5.41
1       4       5.000   B-S-B   DIO_NR  28      4096    0       8348.54     254777.06   1400000 No      3925    317     3593    15      5.50
1       4       5.000   B-S-B   DIO_NR  28      8192    0       10319.00    157455.52   800000  No      6351    318     6018    15      5.08
```

Приложение A 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 раздельного 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 A-1).

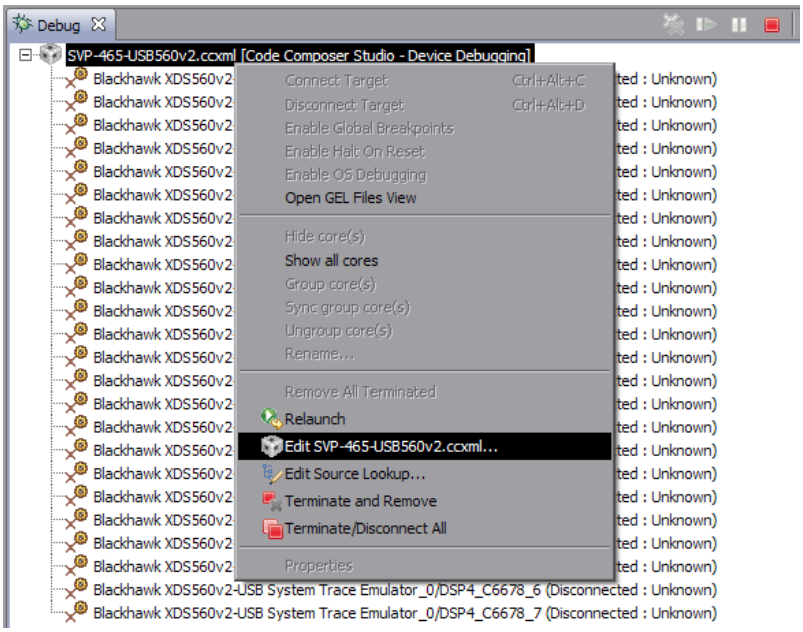


Рисунок A-1: Контекстное меню целевой конфигурации

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

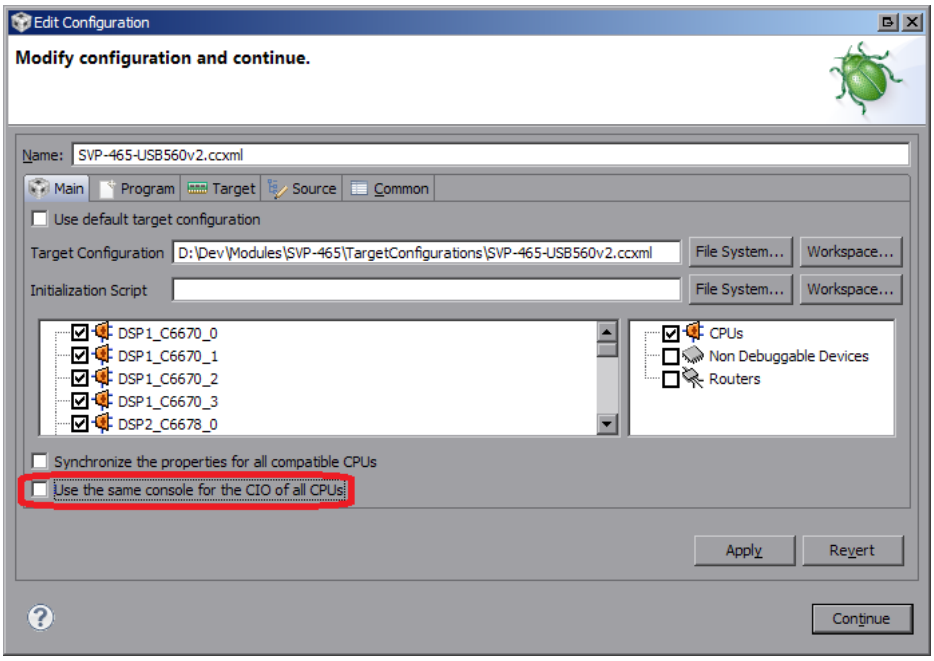


Рисунок A-2: Окно настроек целевой конфигурации

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

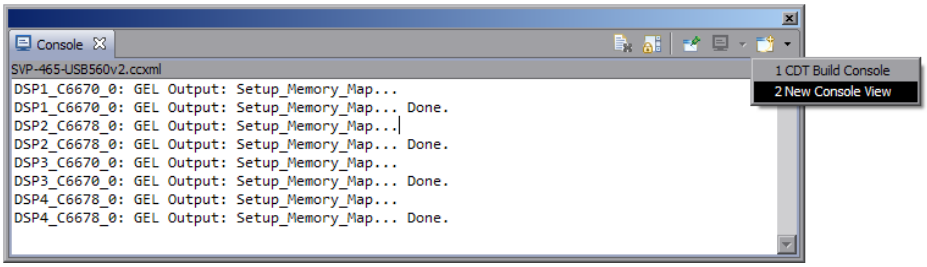


Рисунок A-3: Открытие второго окна «Console»

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

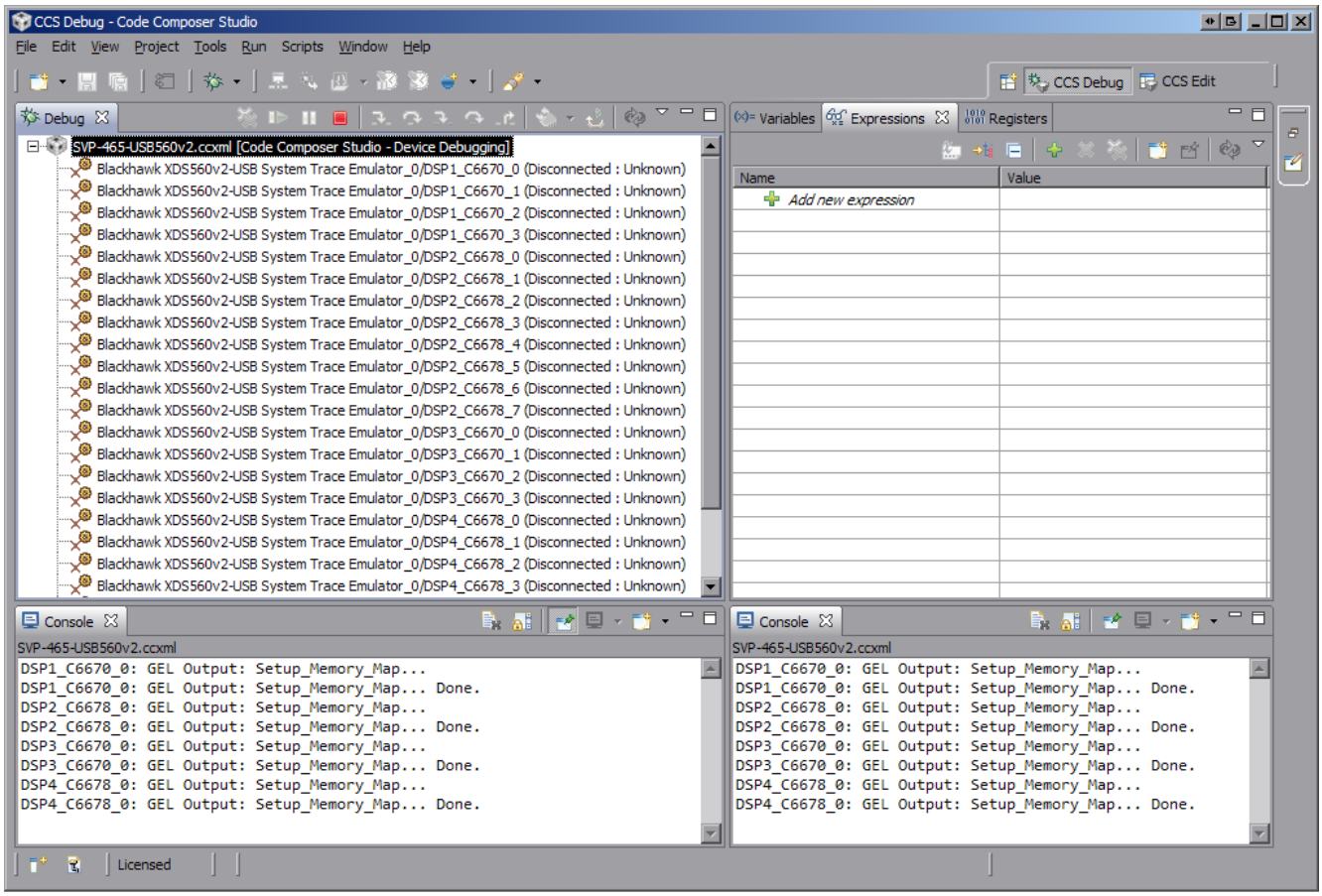


Рисунок A-4: Два окна «Console»

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 A-5).

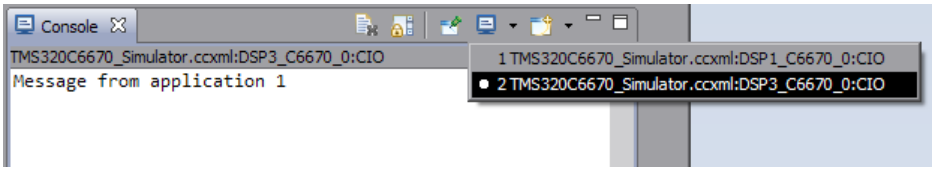


Рисунок A-5: Выбор ядра для отображения вывода в окне «Console»

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

Например, если на ядре «DSP1_C6670_0» запустить простое приложение, выводящее сообщение «Message from application 1», а на ядре «DSP3_C6670_0» запустить второе приложение, выводящее сообщение «Message from application 2», то список доступных консолей будет соответствовать рисунку A-5, а вывод в два окна будет выглядеть, как показано на рисунке A-6.

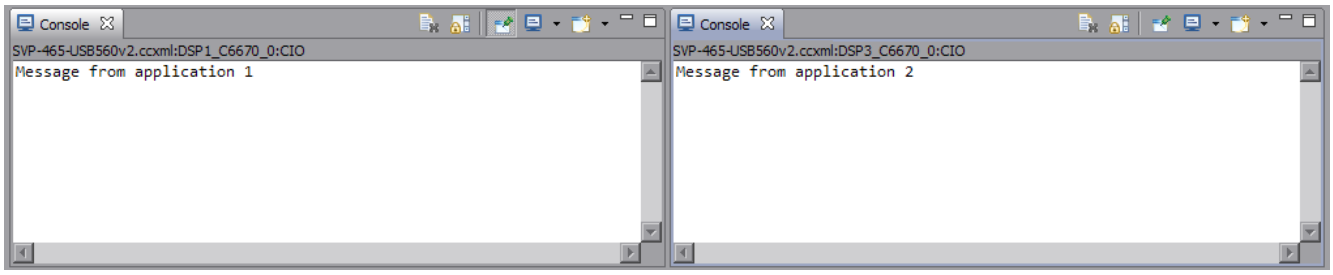


Рисунок A-6: Вывод сообщений с двух ядер в два окна «Console»

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