Application Report
SPRAA54 - September 2004

Using Code Composer Studio with Virtio Platform Simulators

SW Tools Group – TI India

ABSTRACT

The Virtio VPOM2420 has been developed to model the TI Innovator™ kit for the OMAP2420 Platform. This virtual platform works with Code Composer Studio™ to develop and debug ARM and DSP programs. Code Composer Studio can be connected to VPOM2420 to develop and debug digital signal processors (DSPs), ARM11, and ARM7 code individually, or simultaneously.

Contents

List of Figures

1 Introduction

The VPOM2420 Virtual Platform is a fast, full-function software emulation of the TI Innovator Development Kit for the OMAP™ platform. VPOM2420 delivers dramatic gains in developer productivity by providing a complete desktop environment for application, middleware, operating system and device driver development. The VPOM2420 supports a variety of operating systems (OSs) and tool chains for application development.

Integration of Code composer Studio with VPOM2420 allows DSP developers to develop and debug applications for the OMAP2420 platform. Code Composer Studio provides ARM and DSP developers basic debugging capabilities such as: loading DSP programs, execution control, breakpoint control, and register and memory visibility.

Code Composer Studio installation has the necessary drivers for the integration of CCStudio into VPOM2420. Developers will be able to connect CCStudio to debug ARM and DSP programs in full system modeled in VPOM2420.

1.1 Prerequisites

To make use of the C55x, ARM11, ARM7 and OMAP Virtio simulators through Code Composer Studio, VPOM2420 must be installed on your machine. You can purchase the VPOM2420 installation from Virtio. For details, check the following website: http://www.virtio.com

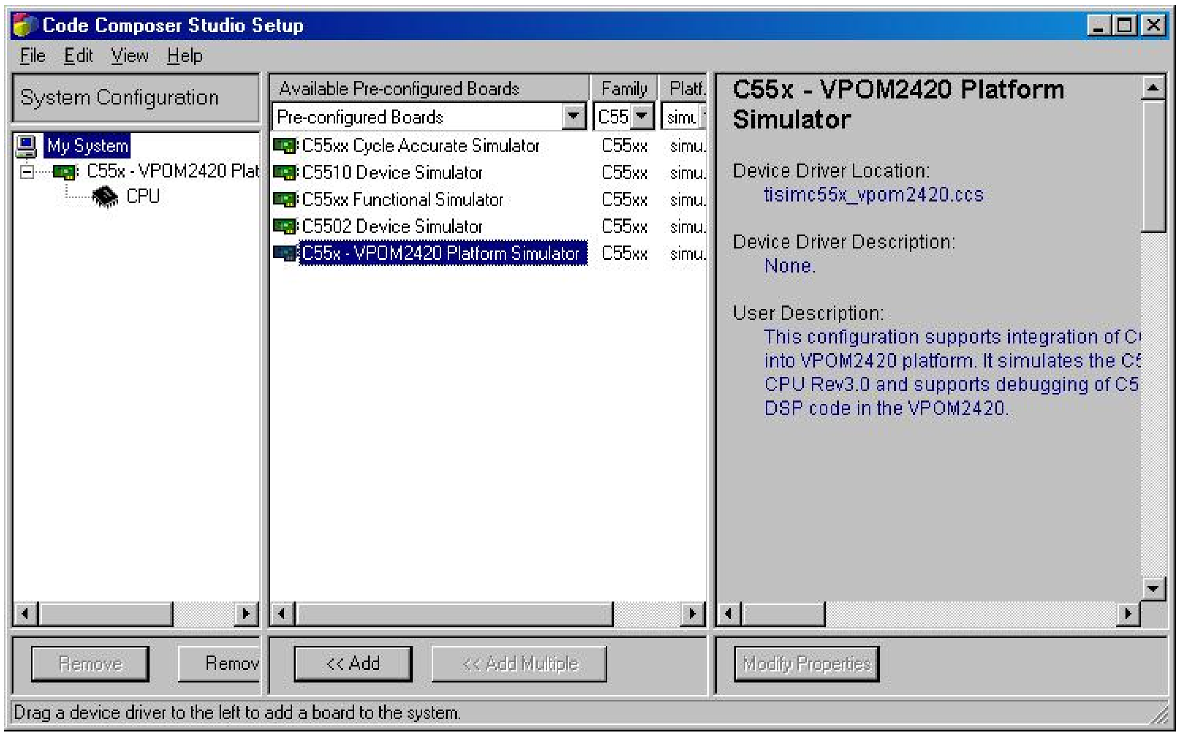
2 Configuring Code Composer Studio

Code Composer Studio must be configured correctly to access the correct simulator configuration. This section will guide you through the setup procedure.

2.1 Configuring CCStudio to Connect Only DSP in VPOM2420

To configure CCStudio Setup for selecting only the C55x DSP configuration for VPOM2420:

1. Start Setup CCStudio from your desktop icon.
2. From the middle pane of CCSetup, select the Pre-configured Board option. Select the C55xx family and simulator platform. Select the configuration, C55x-VPOM2420 Platform simulator from the Available Pre-configured Boards column and add it to the system. Finally, save the selection. Figure 1 shows a screenshot of the setup after importing the configuration.



My System
C55x - VPOM2420 Plat
CPU

Pre-configured Boards | C55 | simu

Available Pre-configured Boards	Family	Platf.
C55xx Cycle Accurate Simulator	C55xx	simu.
C5510 Device Simulator	C55xx	simu.
C55xx Functional Simulator	C55xx	simu.
C5502 Device Simulator	C55xx	simu.
C55x - VPOM2420 Platform Simulator	C55xx	simu.

C55x - VPOM2420 Platform Simulator

Device Driver Location:
tisimc55x_vpom2420.ccs

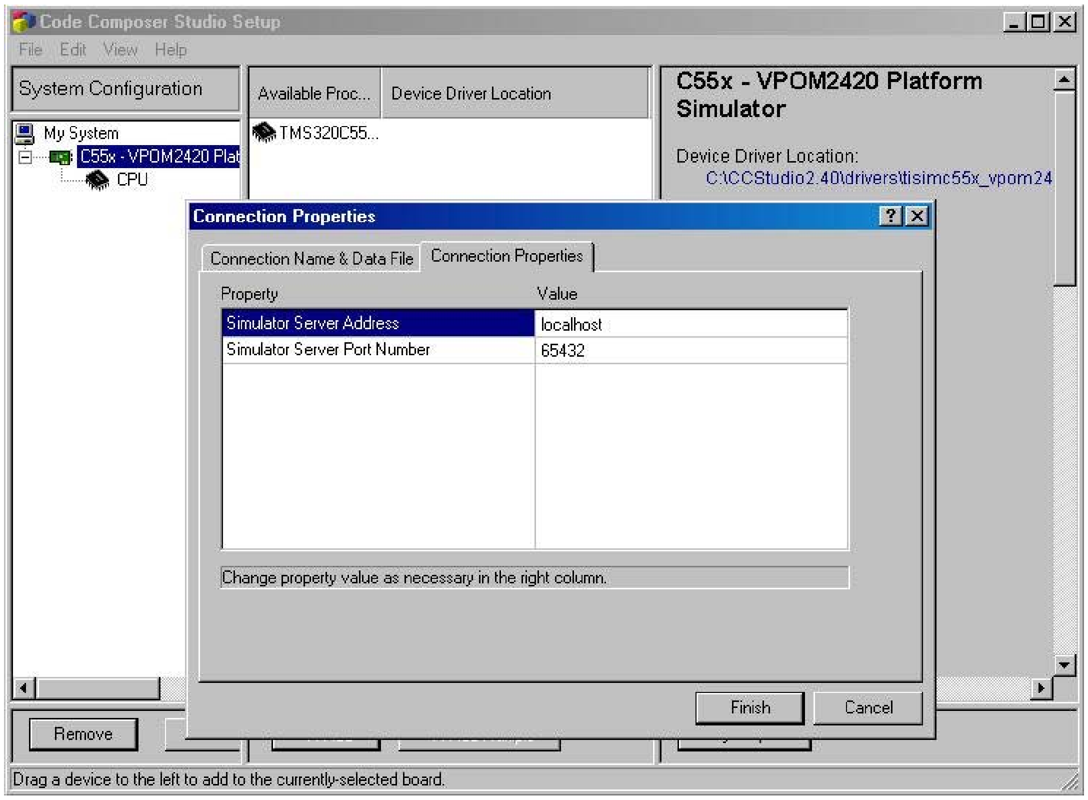
Device Driver Description:
None.

User Description:
This configuration supports integration of C
into VPOM2420 platform. It simulates the C5
CPU Rev3.0 and supports debugging of C5
DSP code in the VPOM2420.

Drag a device driver to the left to add a board to the system.

Figure 1. CCSetup Import Configuration for VPOM2420 C55x Simulator

3. Remove any other boards from the System Configuration pane.
4. To see board properties, right click on the C55x Driver for VPOM2420 in the System Configuration pane and select Properties from the context menu (see Figure 2). Check the Connection Properties tab to see the values of the server address (default value is localhost) and server port number (default value is 65432). Typically, you won’t need to update these fields. However, if you need to change any of these parameters, follow step 5; otherwise, go directly to step 7.



C55x - VPOM2420 Platform Simulator

Device Driver Location:
C:\CCStudio2.40\drivers\tisimc55x_vpom24

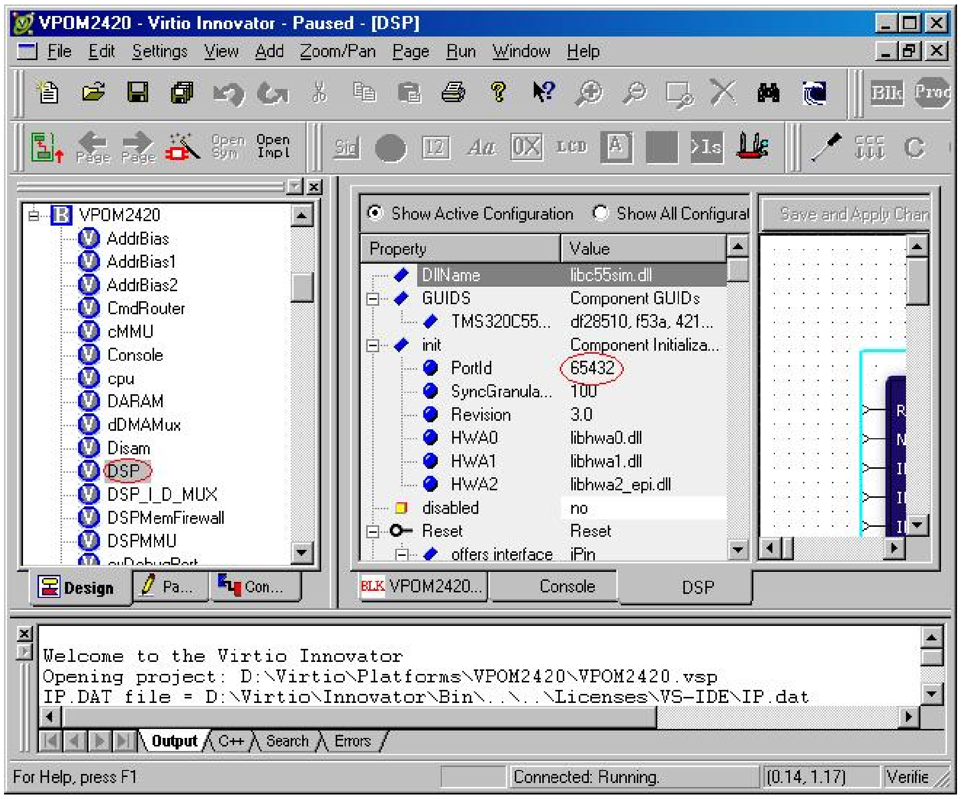
Property	Value
Simulator Server Address	localhost
Simulator Server Port Number	65432

Change property value as necessary in the right column.

Drag a device to the left to add to the currently-selected board.

Figure 2. CCStudio Configuration Settings for VPOM2420 C55x Simulator

5. CCStudio and VPOM communicate with each other using a network port address. If the server port number is already in use, edit this field to another value that is not in use and save the configuration. Click Finish to save the setup parameters.
6. If you change the server port number, you must change the portId in VPOM2420 platform as shown in Figure 3. To do so, follow these steps:
   a. Launch VPOM2420 .
   b. Navigate the left pane to reach the DSP configuration. (VPOM2420 → DSP).
   c. Double−click on the DSP component. This will open a configurations window.
   d. In this configurations window, modify the PortId value (under Init section) to match the server port number set in step 5.



VPOM2420
- AddrBias
- AddrBias1
- AddrBias2
- CmdRouter
- cMMU
- Console
- cpu
- DARAM
- dDMAMux
- Disam
- DSP
- DSP_I_D_MUX
- DSPMemFirewall
- DSPMMU

Property	Value
DllName	libc55sim.dll
GUIDS	Component GUIDs
TMS320C55...	df28510, f53a, 421...
init	Component Initializa...
PortId	65432
SyncGranula...	100
Revision	3.0
HWA0	libhwa0.dll
HWA1	libhwa1.dll
HWA2	libhwa2_epi.dll
disabled	no
Reset	Reset
offers interface	iPin

```
Welcome to the Virtio Innovator
Opening project: D:\Virtio\Platforms\VPOM2420\VPOM2420.vsp
IP.DAT file = D:\Virtio\Innovator\Bin\..\..\Licenses\VS-IDE\IP.dat
```

Figure 3. Component Setting from VPOM2420

7. Save the setup parameters and exit.

2.2 Configuring Code Composer Studio to Connect ARM11

CCStudio Setup must be configured appropriately for selecting the ARM11 configuration for VPOM2420:

1. Start Setup CCStudio from your desktop icon.
2. From the middle pane of CCSetup, select the Pre-configured Boards option. Select the ARM11 family and simulator platform. Then select the configuration, ARM11-VPOM2420 Platform Simulator from the Available Pre-configured Boards column and add it to the system. Finally, save the selection. Figure 4 shows a screenshot of the setup after importing the configuration.

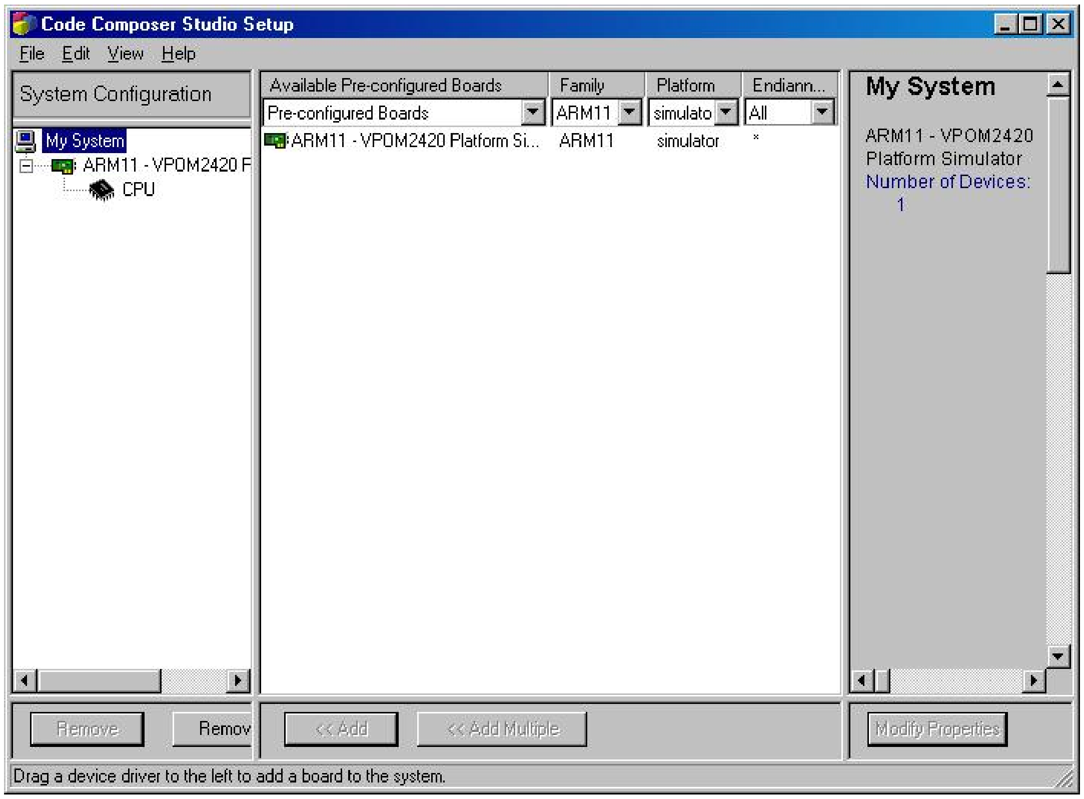


Figure 4. CCStudio Setup Configuration for ARM11 Simulator in VPOM2420

3. Remove any other boards from the System Configuration pane.
   Unlike the CCStudio for C55x, the CCStudio for ARM11 does not have board parameters.
4. Save the setup parameters and exit.

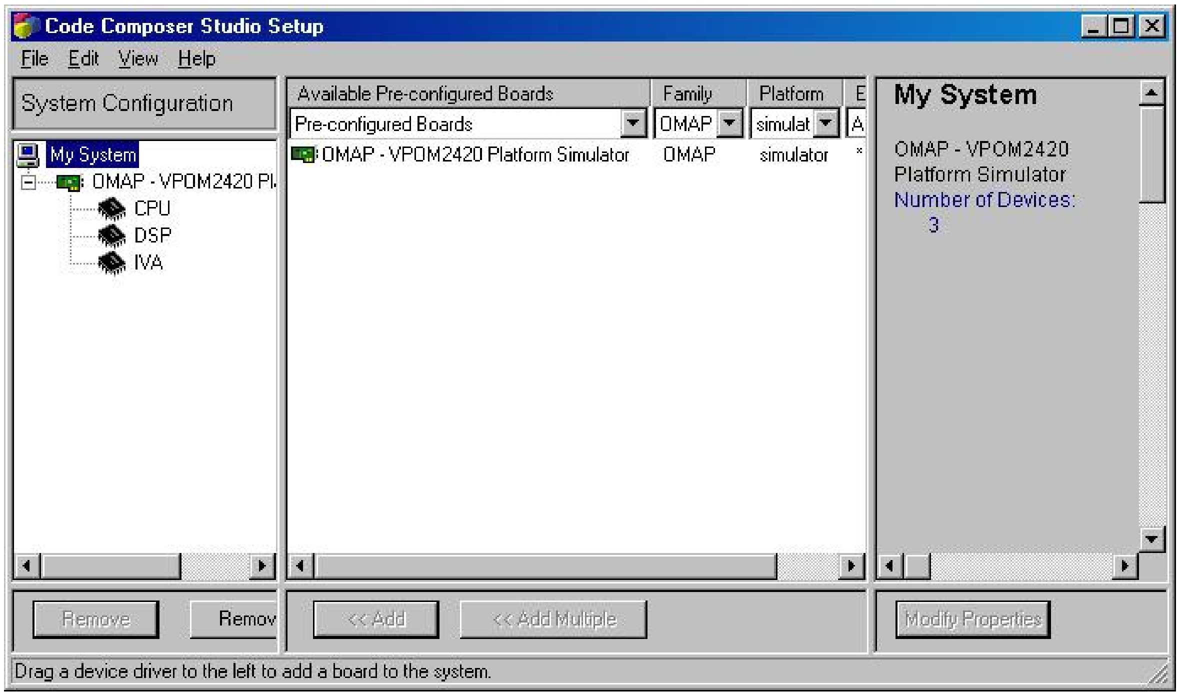
2.3 Configuring Code Composer Studio to Connect only ARM7

The steps for configuring CCStudio to connect ARM7 are identical to that for ARM11. Follow the steps in the previous section, replacing ARM11 with ARM7.

2.4 Configuring CCStudio for DSP, ARM11, and ARM7

CCStudio Setup must be configured appropriately for selecting the OMAP configuration for VPOM2420:

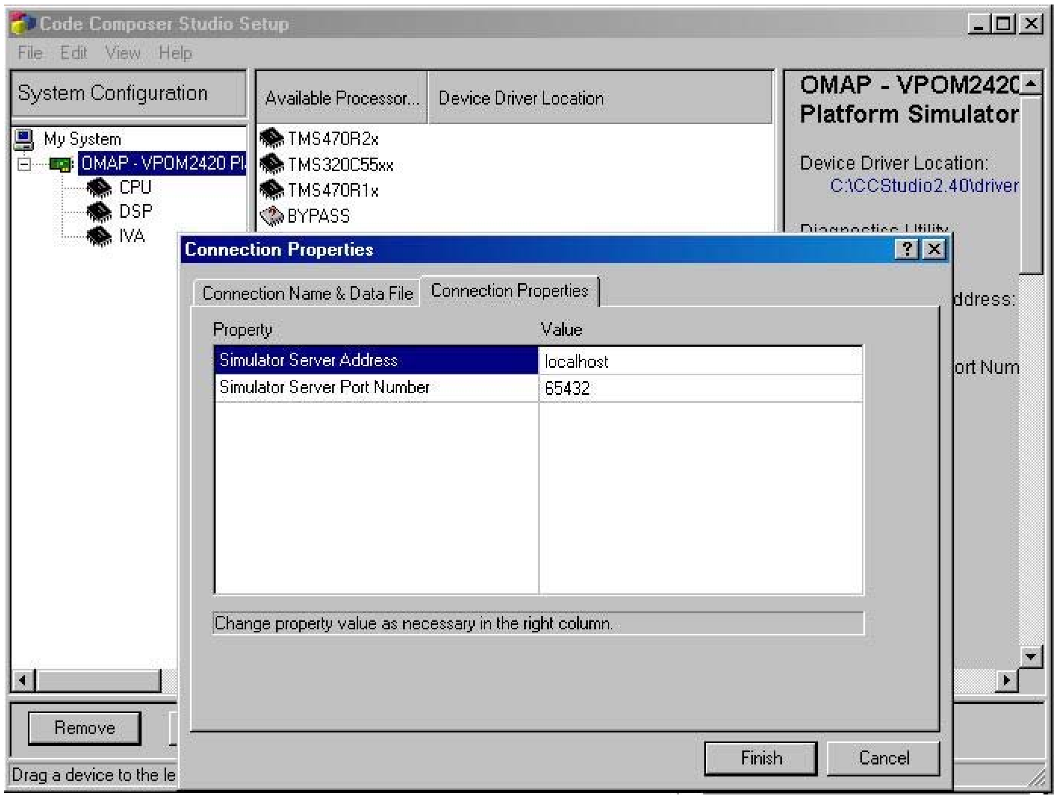
1. Start the Setup CCStudio from the desktop icon.
2. From the middle pane of CCSetup, select the Pre-configured Boards option. Select the OMAP family and simulator platform. Then select the configuration, OMAP-VPOM2420 Platform Simulator from the Available Pre-configured Boards column and add it to the system. Finally, save the selection. Figure 5 shows a screenshot of the setup after importing the configuration.



Drag a device driver to the left to add a board to the system.

Figure 5. CCStudio Setup Import Configuration for ARM11, ARM7, and C55x

3. Remove any other boards from the System Configuration pane.
4. To see board properties, right-click on OMAP Driver for VPOM2420 in the System Configuration pane and select Properties from the context menu. Check the Connection Properties tab to see values of the server address (default value is localhost) and server port number (default value is 65432). Typically, you won't need to update these fields. However, if you need to change any of these parameters, see step 5; otherwise, go to step 7.



Property	Value
Simulator Server Address	localhost
Simulator Server Port Number	65432

Change property value as necessary in the right column.

Figure 6. Simulator Configuration Settings for all of ARM7, ARM11, and C55x

5. CCStudio and C55x simulator in VPOM communicate with each other using a network port address. If the server port number is already in use, then edit this field to another value that is not in use and save the configuration.
6. If you change the server port number in CCStudio Setup as described in step 5, you must change the portId in the VPOM2420 platform as follows:
   a. Launch VPOM2420.
   b. Navigate the left pane to reach the DSP configuration. (VPOM2420 –> DSP).
   c. Double−click on the DSP component. This will open a configurations window.
   d. In this configurations window, modify the PortId value (under Init section) to match the server port number set in step 5.
7. Save the setup and exit.

3 Launching CCStudio

This section includes information about how to use CCStudio with VPOM2420. The following steps should be taken before invoking CCStudio:

- Make sure that simulator configuration is correct.
- Make sure that port number is the same in both CCStudio configuration and VPOM2420 component settings, as described in section 2.1.
- Make sure that VPOM2420 is running before CCStudio is launched.

- If you are going to use only the DSP or ARM11 processor, select Native Apps from the configuration panel. Otherwise (ARM7 only or all processors simultaneously) select Multi−core Native Apps. Select Go on VPOM2420, so that the DSP library gets loaded.
- VPOM2420 must be in halted state before CCStudio can be invoked. VPOM2420 can be halted by issuing command Y from the VPOM2420 console window.
- Now launch CCStudio.

3.1 DSP Application in Standalone Mode

To run a DSP application in standalone mode, follow these steps:

1. Invoke VPOM2420 and run the Native Apps platform.
2. While running DSP in standalone mode, ARM will not be exercised. So, bring DSP out of reset by executing the [VPOM2420::DSP]dr command from the Virtio innovator console window.

4 Debugging Using CCStudio

This section describes different CCStudio capabilities supported by this simulator release. If you are new to CCStudio, then we recommend that you read *Code Composer Studio Getting Started Guide* (SPRU509) before reading this section.

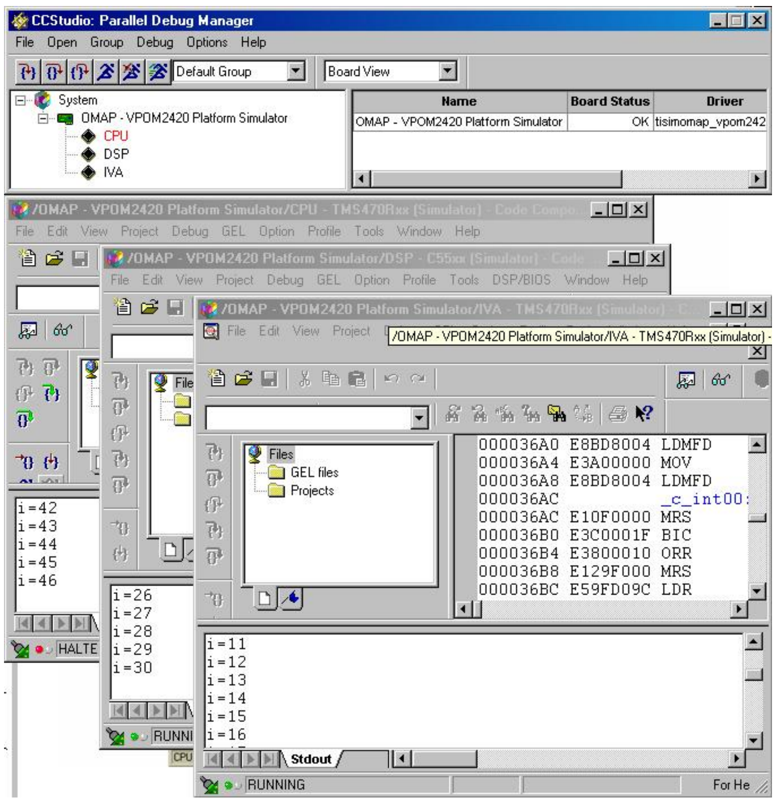
4.1 CCStudio Single Processor Debugging Capabilities

The CCStudio integration provides basic support for debugging, namely:

- Read and Write to IO registers and memory. This also includes program load.
- Execution Control : Run, Step, Halt, Reset
- Set and Clear Breakpoints. CCStudio shows only those breakpoints which are set from CCStudio. Breakpoints can also be set, cleared, and seen from the Virtio command window. The simulator stops at these breakpoints, but as CCStudio does not recognize these breakpoints, it immediately issues Run and the simulation resumes. Hence, the simulation halts only at the user breakpoints that are set using CCStudio.
- Any other features based on simple run and breakpoints, e.g., go−main, source step, step over, etc., are automatically supported.
- CIO on the CCStudio side is supported.
- CCStudio can be connected and disconnected multiple times. Connect and disconnect happen automatically when CCStudio is invoked and exited, respectively.

4.2 CCStudio Multiprocessor Debugging Capabilities

When CCStudio is configured for a multiprocessor target, it comes with the Parallel Debug Manager (PDM) in addition to the usual CCStudio debugger window for each of the processor in the multiprocessor target. Figure 7 shows the PDM (top left) and usual CCStudio debugger windows (ARM11, DSP and ARM7 from top to bottom) when configured for both DSP and ARM11 processors.


CCStudio: Parallel Debug Manager
File Open Group Debug Options Help
Default Group
Board View
System
OMAP - VPOM2420 Platform Simulator
CPU
DSP
IVA
/OMAP - VPOM2420 Platform Simulator/CPU - TMS470Rxx (Simulator) - Code Compo
/OMAP - VPOM2420 Platform Simulator/DSP - C55xx (Simulator) - Code
/OMAP - VPOM2420 Platform Simulator/IVA - TMS470Rxx (Simulator) - C
File Edit View Project Debug GEL Option Profile Tools Window Help
File Edit View Project Debug GEL Option Profile Tools DSP/BIOS Window Help
Files
GEL files
Projects
Stdout
HALTE
RUNNI
RUNNING
For He

Name	Board Status	Driver
OMAP - VPOM2420 Platform Simulator	OK	tisimomap_vpom242

```
000036A0 E8BD8004 LDMFD
000036A4 E3A00000 MOV
000036A8 E8BD8004 LDMFD
000036AC          _c_int00:
000036AC E10F0000 MRS
000036B0 E3C0001F BIC
000036B4 E3800010 ORR
000036B8 E129F000 MRS
000036BC E59FD09C LDR
```

```
i=42
i=43
i=44
i=45
i=46
```

```
i=26
i=27
i=28
i=29
i=30
```

```
i=11
i=12
i=13
i=14
i=15
i=16
```

Figure 7. CCStudio with the Multiprocessor Debugger Window

The current integration provides basic support for single processor debugging from the processor specific debugger windows as described in section 4.1.

In addition to those features, CCStudio supports the following commands from the PDM window. To use the commands, make sure that you group all of the three processors into a single logical board. You can ensure this by selecting Default Group in the first box and Group Setup in the second box, and ensuring that all three processors have the boxes checked under Default Group.

- Synchronized Execution Control: Run, Halt & Animate.
- Synchronized steps do not work properly in this release.
- Global breakpoints are not supported.
- CCStudio can be connected and disconnected multiple times. Connect and disconnect happen automatically when CCStudio is invoked and exited, respectively. Note that the disconnect happens when the PDM window is closed, not when the processor-specific debugger window is closed.

4.3 Execution Control

In VPOM2420 with CCStudio integration, the execution commands can be given from the following places:

- VPOM2420 console window

- Processor-specific windows in CCStudio
- PDM window in CCStudio

Among the above three types of execution windows, the console window works closely with the VPOM2420 system simulation infrastructure. It keeps track of the debuggers that are attached to the simulation, and the state of the corresponding processor. The simulation only runs when all the attached debuggers want it to run.

The listDbg command can be used from the console to list the state of the system-recognized, attached debuggers. Whenever the system receives a run request from one of the attached debuggers, but the simulation is waiting for other attached debuggers to issue their run commands, a message is printed in the console stating which debuggers the simulation is waiting for.

The following six different scenarios depend on how the above three type of execution windows are used.

1. **Debugging from VPOM2420 system console without attaching CCStudio to any one of the processors.**
   This scenario is the simplest one. Simply debug from the console window by issuing execution and breakpoint commands to DSP, ARM7 and ARM11.

2. **Debugging program from CCStudio that is connected to one of ARM11, ARM7 and DSP but not more**.
   In this case, once the debugger has started, you do not have to bother about the console window. You will debug programs for the single processor by issuing commands from the CCStudio. The execution command will automatically start and stop the platform simulation when required.

3. **Debugging program from processor specific debugger window in CCStudio that is connected to all of the three processors.**
   In this case, once CCStudio has started, you do not have to initiate an execution command from the console window. However, you have to give an execution command from both processor-specific windows. The VPOM2420 system keeps track of the state of the debuggers. It does not start simulation until the execution has been initiated from all the processor-specific debugger windows. If, after simulation has started, one processor hits the breakpoint state of the corresponding debugger, it is halted. The other debugger is still kept running, although platform simulation does not proceed at this state. Once the halted debugger issues another run command, the simulation starts as usual.

4. **Debugging program from PDM window in CCStudio that is connected to all of the three processors.**
   The repetitive work of clicking run buttons from all the processor-specific windows can be avoided by using this configuration. In this case, issuing a run command from the PDM window will automatically start the simulation. You do not have to give run commands in the processor-specific debugger windows. Initiating Halt from the PDM will halt both the processors. However, before using the execution from the PDM window, make sure that you have kept both the processor in the same execution group. You can do this by invoking Group→Edit menu.

5. **Debugging from VPOM2420 system console even when CCStudio is connected one or more processors.**
   If you have attached the CCStudio, you may not want to use the VPOM2420 console window for debugging. Still, if you wish to debug the VPOM2420 system, you should initiate a run on the CCStudio windows (either by issuing a run command from all the processor-specific windows or by giving a synch run command from the PDM window) and immediately stop the platform simulation by giving a “y” command from the console. This will keep the debugger windows running. Now, you can use a combination of trace/go/breakpoints commands on VPOM2420 components to debug system behavior. You can find different execution control commands supported by VPOM2420 in the guide that will be available in the VPOM2420 installation.
6. **Debugging simultaneously from both VPOM2420 system console and CCStudio that is connected one or more processors.**
   This is the most complicated usage scenario. In this case, you need to initiate a run or any other execution control command from both sides. Once execution stops on one of the sides, enter a run or go only command from that side. For example, to debug DSP code and DSP−MMU accesses, put some breakpoints on the DSP as well as DSP−MMU. Then, initiate the run/go command from CCStudio and from VPOM2420. If the DSP breakpoint is hit, initiate the run only command from the DSP side. Instead, if the DSP−MMU breakpoint is hit, you need to give the go only command from the VPOM2420 side.

4.4 Loading Programs

CCStudio allows you to load programs in coff or dwarf format. However, you must ensure that program load is happening to valid memory locations. If a particular memory address is unmapped, program load will fail.

4.5 Memory Display

CCStudio allows you to access DSP, ARM7 or ARM11 memory for reading and writing. If memory read fails in VPOM2420, the debugger displays 0xBD for all failed accesses. Memory access can fail if it is for unmapped memory, or if it generates a DSP−MMU page fault.

4.6 CPU Register Display

CCStudio allows you to access C55x, ARM7 and ARM11 registers using the register display window. The hardware accelerator register display is not supported as of now.

4.7 Handling Breakpoints

CCStudio allows you to set and clear breakpoints using its available breakpoint controls.

Appendix A

This chapter lists CCStudio features which are unsupported by this simulator.

A.1 Unsupported Features

- RTDX support
- Pin−connect
- Port−connect
- Pipeline Stall Analyser
- C55x hardware accelerator register display
- Simulator analysis

A.2 Limitations and Known Issues

- Synchronized steps do not work properly.
- When all of ARM11, ARM7, and DSP are connected to the debugger, in some cases when one of the processors is halted, CCStudio may generate the error message, Invalid CIO command (num). Ignore this message.