Implementing an 8-bit Processor-based Design in an FPGA

Summary

Tutorial
TU0118 (v3.0) March 04, 2008

This tutorial describes how to implement an 8-bit processor-based design in an FPGA. It describes the creation of FPGA and Embedded projects, creating a C file, setting up processor and compiler options and then configuring and programming the design to an FPGA device.

This tutorial will create an FPGA project containing a single schematic sheet, and an embedded project containing a single C source file. The design and associated software will be downloaded to the Altera Cyclone EP1C12Q240C6 device on the daughter board supplied with the NanoBoard-NB1. The resulting bit counter will be displayed on the NanoBoard's LEDs.

Connect the NanoBoard-NB1 to the parallel port of the PC and power up the board by flicking the ON switch. Make sure you have installed the Quartus tools (web edition) from Altera. These tools can be downloaded from the Altera website (*www.altera.com*).

Note: Although this tutorial targets the Altera Cyclone EP1C12Q240C6 device on a 2-connector daughter board attached to a NanoBoard-NB1, this daughter board can be plugged in to the Desktop NanoBoard NB2DSK01 as well. In fact, the design can be targeted to any daughter board FPGA device, attached to either of these NanoBoards, provided the relevant constraint file(s) are defined to target that device and the relevant vendor tools are installed on your PC.

For a tutorial that looks at implementation of a 32-bit processor-based design, and targeted to a daughter board FPGA device attached to the Desktop NanoBoard NB2DSK01, refer to the document *TU0128 Implementing a 32-bit Processor-based Design in an FPGA*.

Creating the FPGA Project

First we will create an FPGA project.

1. Create a new FPGA project by selecting **File » New » Project » FPGA Project** (or click on **Workspace** button and select **Add New Project » FPGA project**).

2. Save this project as `FPGA_Project1.PrjFpg` by selecting **File » Save Project** (or right-click on the new project name in the **Projects** panel and select **Save Project**) in a new directory called `8-bit FPGA Processor`.

> **Warning:** Do not use spaces or dashes (-) in file names or project names. Use underscores (_) if necessary.

3. Add a new schematic document to the FPGA project by selecting **File » New » Schematic** (or click on the **Project** button and select **Add New to Project » Schematic**). Save the schematic as `Sheet1.SchDoc`, in the same directory as the parent project.

4. Place the following components (listed below in Table 1) on the schematic sheet as shown in Figure 1. Placement can be made direct from the **Libraries** panel. Simply select a component and either click the **Place** button at the top-right of the panel, or click and drag the component directly onto the sheet. Each of these components are available from the default generic FPGA integrated libraries, so there is no need to add any libraries for this tutorial.

Table 1. Components used in the bit counter schematic

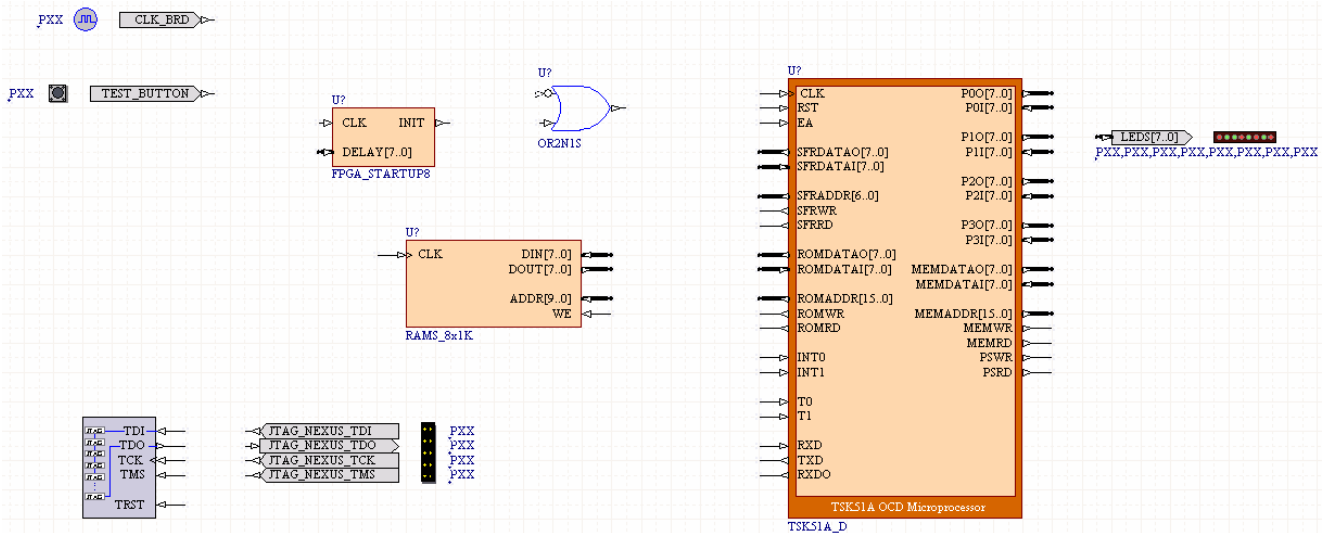| Component name...	Available from FPGA integrated library...
TSK51A_D (the processor)	FPGA Processors.IntLib
RAMS_8x1K	FPGA Memories.IntLib
CLOCK_BOARD, TEST_BUTTON, LED, NEXUS_JTAG_CONNECTOR	FPGA NB1 Port-Plugin.IntLib
NEXUS_JTAG_PORT, OR2N1S	FPGA Generic.IntLib
FPGA_STARTUP8	FPGA Peripherals.IntLib

Figure 1. Initial placement of components on the schematic sheet, Sheet1.SchDoc.

5. Wire the design initially, as shown in Figure 2, by using the **Place » Wire** and **Place » Bus** commands.
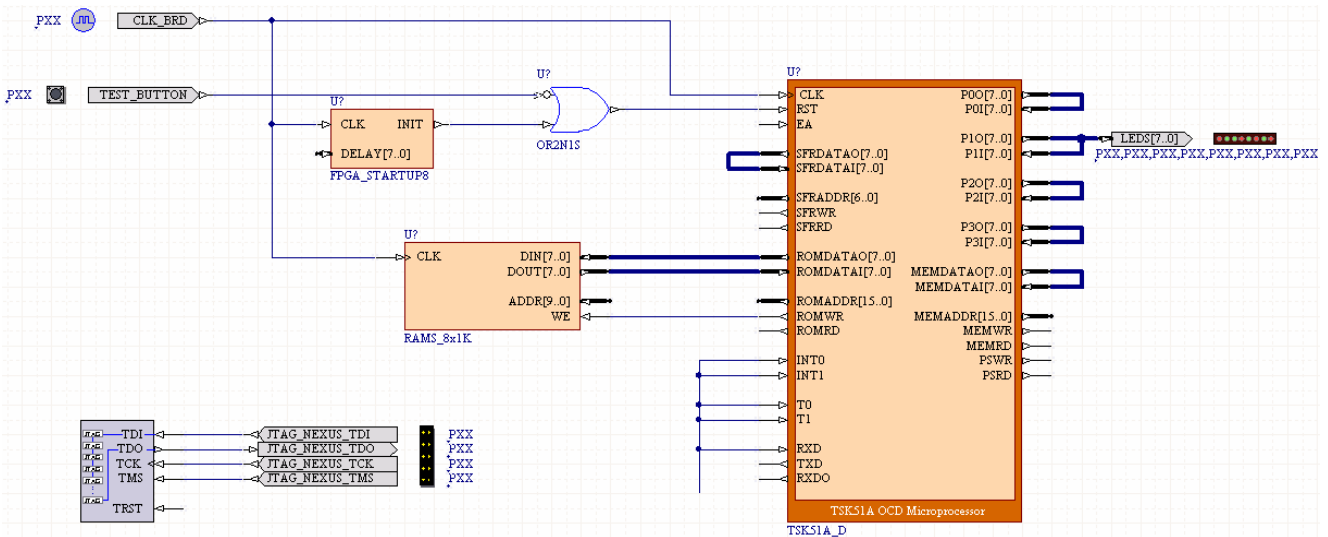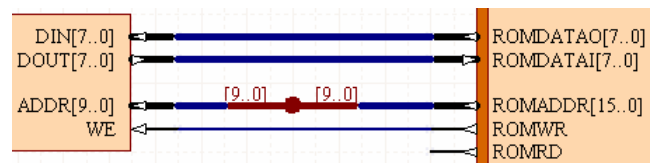

Figure 2. Initial wiring – placement of wires and buses.

6. Connect the RAMS_8x1K's ADDR[9..0] pin to the ROMADDR[15..0] pin on the TSK51A_D symbol using a bus joiner. Place the component JB from the FPGA Generic.IntLib to join the buses. Modify the parameter text to read [9..0] on both sides of the joiner by using inline text editing mode.

7. Place a GND power port (from the **Wiring** toolbar) on the EA pin of the TSK51A_D processor. Change the Style to Arrow using the associated *Power Port* dialog.

8. Connect pins INT0, INT1, T0, T1 and RXD to a GND power port as well.

9. Place a VCC power port (from the **Wiring** toolbar) on the TRST pin of the NEXUS_JTAG_PORT component, using the Bar style.

10. Place a VCC Bus power port (from the **Wiring** toolbar) on the DELAY[7..0] pin of the FPGA_STATRTUP8 component (its **Net** property should read VCCBUS[..] in the associated *Power Port* dialog).

11. Add No ERC markers to each of the output pins in the design that are not being used.

12. Annotate the design using the **Tools » Annotate Schematics Quietly** command. The *Confirm Designator Changes* dialog displays the following message: "There are 4 designators requiring update. Proceed with changes?". Click **Yes**. The designators will be automatically changed from U? to U1, for example, and numerically augment. Your FPGA design schematic should now appear as shown in Figure 3.

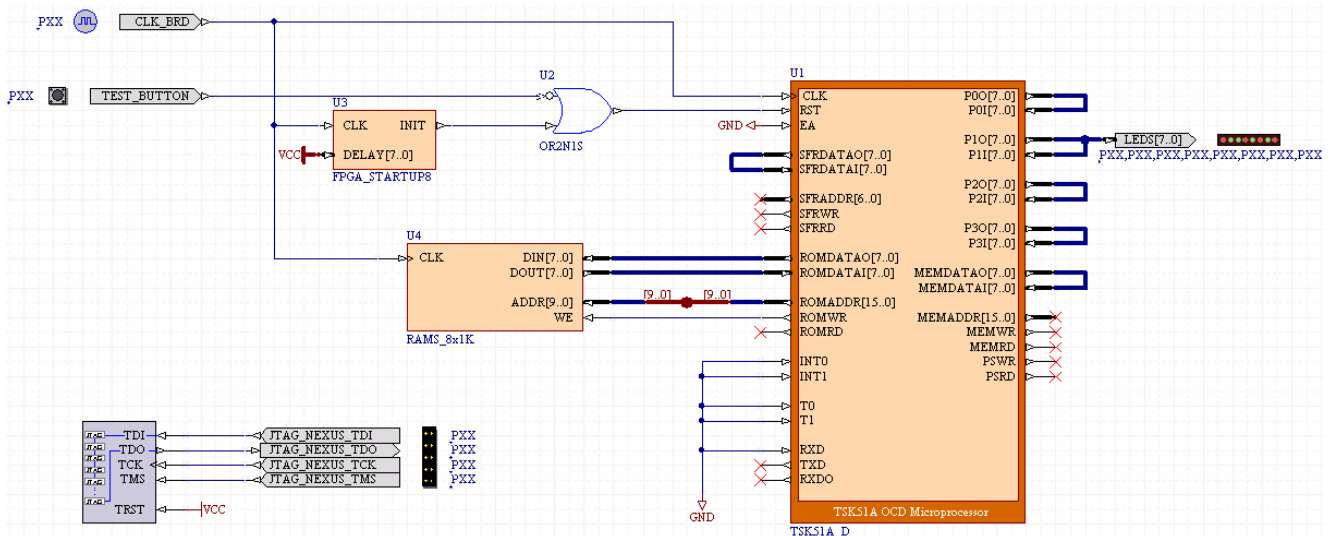> Use the **Spacebar** to rotate a power port while placing it.

Figure 3. Final schematic sheet, fully wired.

13. Save `Sheet1.SchDoc` [shortcut **Ctrl+S**].

Creating the Embedded Project

Now we can create an embedded project that will hold the source file for the software we want to download to the FPGA device.

1. Create a new Embedded project by selecting **File » New » Project » Embedded Project** (or click on the **Workspace** button and select **Add New Project » Embedded Project**).

2. Save this project by selecting **File » Save Project**, or right-click on the new project name in the **Projects** panel and select **Save Project**. Use a name without any spaces (e.g. `Embedded_Project1.PrjEmb`) and save it in the same directory as the FPGA project and schematic files.

3. Create a new C file by right-clicking on the embedded project name in the **Projects** panel and selecting **Add New to Project » C File**, or click on the **Project** button and select **Add New to Project » C File**.

4. Type the C code (as shown) into the new C file.

5. Save the code in the same directory as the other project files. Using the default name (`Source1.C`) is fine for this tutorial.

## Setting the Embedded Software Project Options

Next we will set the processor startup code and set some C compiler options, such as the memory model and code generation options.

1. Right-click on the embedded project name (`Embedded_Project1.PrjEmb`) in the **Projects** panel and select **Project Options**. The *Options for Embedded Project* dialog opens.

2. Add the startup code to your project by clicking on **Processor** in the Build Options list and click on **Startup Code**. Make sure the option **Generate and use startup code <project>_cstart.c** is enabled.
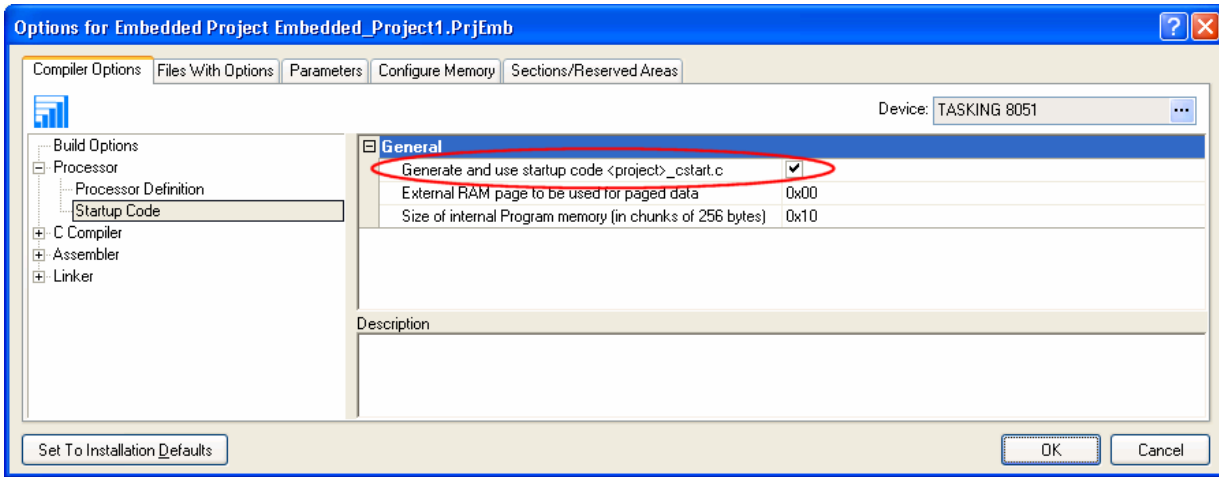
3. Set a memory model by clicking on **C Compiler** in the Build Options list, selecting **Memory Model** and choosing **Small: variables in DATA** from the drop-down list for the compiler memory model. Make sure **Allow reentrant functions** is not selected.
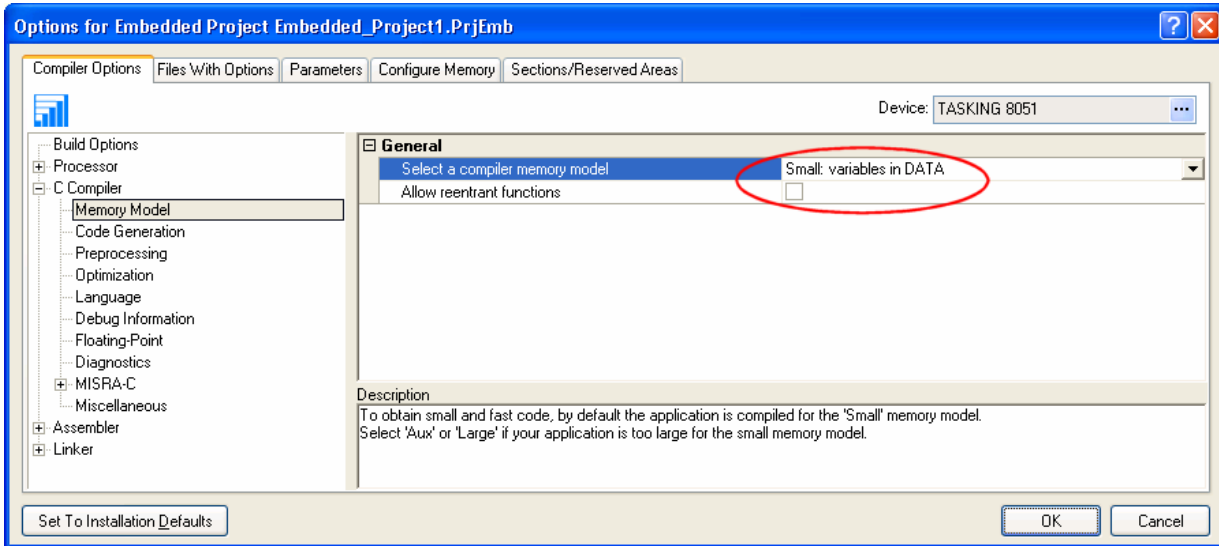
4. While still in the C Compiler options, click on **Code Generation** and enable the Strings options to **Put strings in ROM only**. Make sure the **Interrupts** and **Registers** options are also enabled.
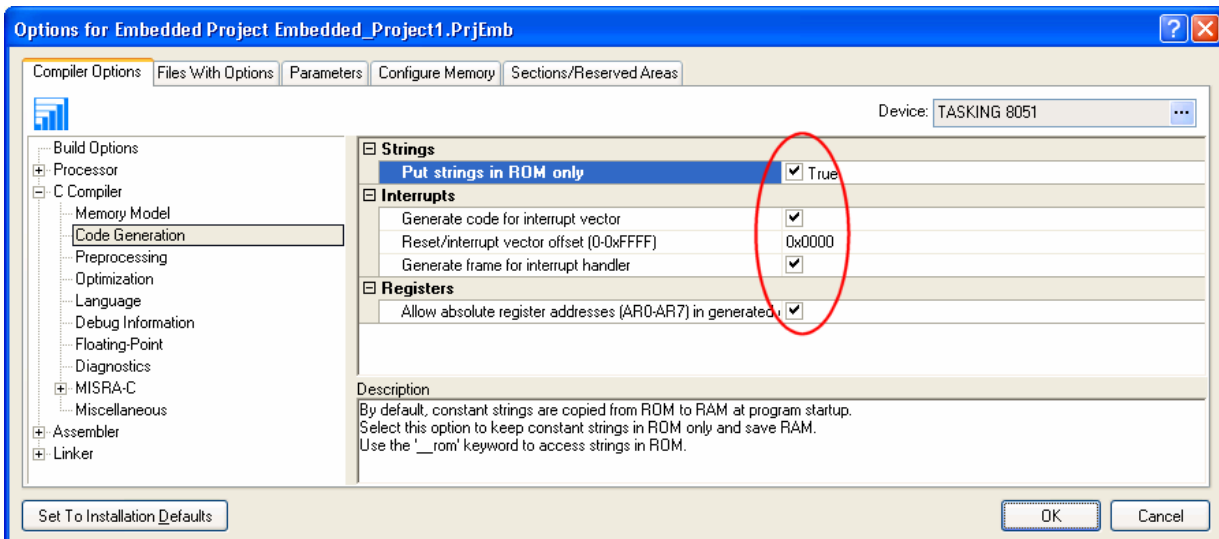

5. Click **OK** to save all options and close the dialog.

TU0118 (v3.0) March 04, 2008

Setting the Processor Properties

Now we need to tell the processor on the schematic where to find its RAM and which sub-design (embedded project) to use when the software is to be downloaded to the FPGA device on the daughter board.

1. Go back to `Sheet1.SchDoc` in the Schematic Editor by clicking on its document tab.

2. Double-click on the TSK51A_D processor (`U1`) to open its *Component Properties* dialog.

3. In the Parameters section, enter the designator of the RAM (most likely `U4` — refer to your schematic) in the **Value** field of the **ChildCore1** parameter. Click **OK**.
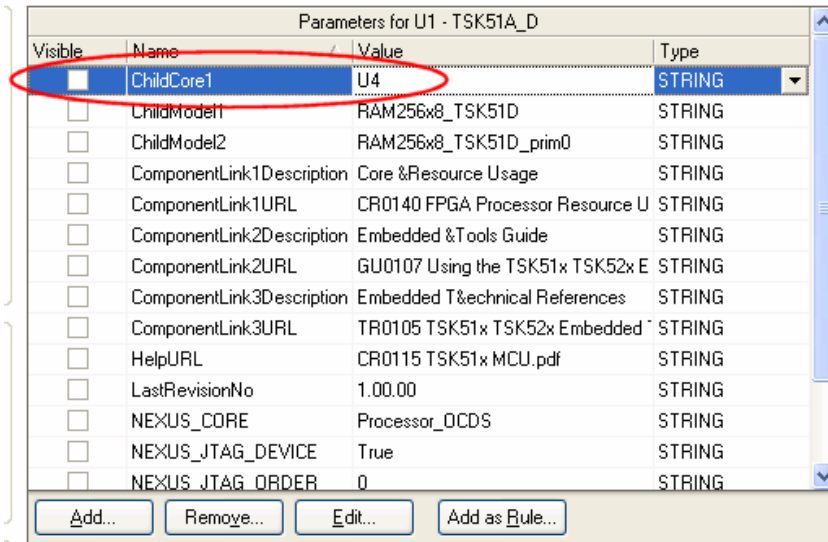
4. To link the Embedded project to the processor component, go to the **Projects** panel and click on **Structure Editor** button to display the project structure view. Make sure that your hardware project is compiled so that it appears in the structure tree (**Project » Compile FPGA Project**).You will see the Valid Sub Projects and Configurations list in the bottom section of the **Projects** panel.
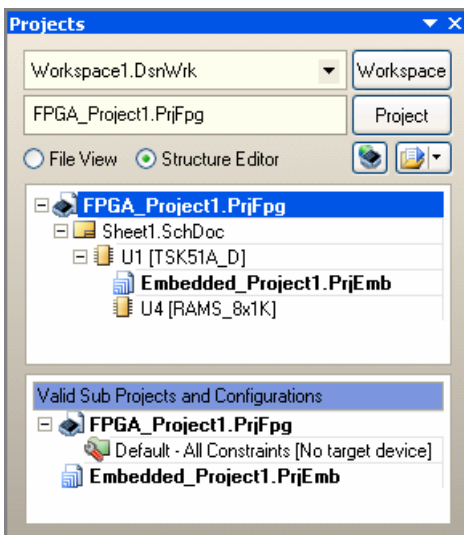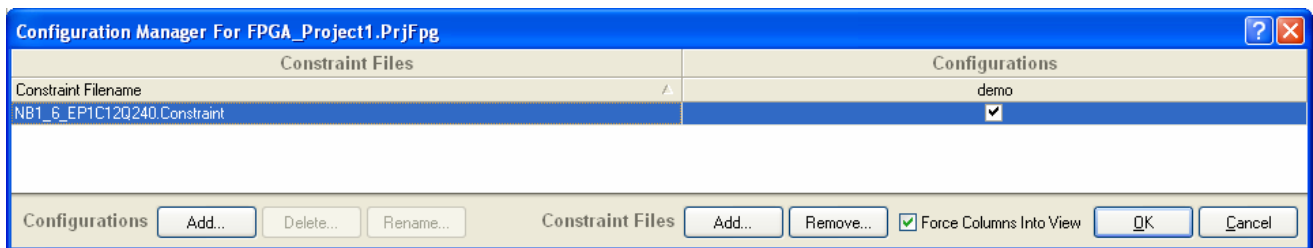

Select the software project which we just created from the Valid Sub Projects and Configurations list, e.g. `Embedded_Project1.PrjEmb`, by clicking on its icon and drag-and-drop it onto the processor component (TSK51A_D) icon in the top section of the **Projects** panel. Note that valid targets will be highlighted. The link will be established and the structure will recompile to re-establish the integrity.

5. Save all files (**File » Save All**).

Configuring the Design to an FPGA Device

Now we need to specify which FPGA device we want to use in our design, e.g. the Altera Cyclone EP1C12Q240C6 device on the 2-connector daughter board attached to the NanoBoard-NB1. We will add a configuration and constraint file to do this. The constraint file will specify the device and determine the pin numbering.

1. To create a new Configuration, right-click on the FPGA project name in the **Projects** panel (changing back to File View if still in Structure Editor mode) and select **Configuration Manager**, or select **Project » Configuration Manager** from the menus. The *Configuration Manager for FPGA_Project1.PrjFpg* dialog appears.

2. Click on the **Add** button in the Configurations section of the dialog and type a configuration name in the *New Configuration* dialog that appears, e.g. `demo`. Click **OK**.

3. Add a constraint file to your configuration by clicking on the **Add** button in the Constraints section of the dialog. In the *Choose Constraint files to add to Project* dialog that appears, drill down to the `\Library\Fpga\NB1 Constraint Files\Altera FPGA` folder and select the `NB1_6_EP1C12Q240.Constraint` file. Click **Open**.

4. Back in the *Configuration Manager* dialog, assign the constraint to the configuration and click **OK**.

Configuration Manager For FPGA_Project1.PrjFpg	
Constraint Files	**Configurations**
Constraint Filename	demo
NB1_6_EP1C12Q240.Constraint	☑

Configurations [Add...] [Delete...] [Rename...]   **Constraint Files** [Add...] [Remove...] ☑ Force Columns Into View [OK] [Cancel]

5. The constraint file appears in the **Projects** panel under the `Settings\Constraints Files` sub-folder.

6. Save all files (**File » Save All**).

Using the Devices View to Program the FPGA

The **Devices** view (**View » Devices View**) allows you to follow through the workflow (from left to right) required to send your program to the FPGA. In this view, you can:

*   Compile the project (and check for errors)

*   Synthesize (create an EDIF netlist)

*   Build (translate the EDIF files, map the design to the FPGA, Place and Route the FPGA, run a Timing Analysis and then Make the Bit File)

*   Program FPGA (download the bit file to the daughter board's FPGA device, e.g. the Altera Cyclone).

When this workflow is completed, you will be able to run the program by flicking on the DIP switches on the NanoBoard. To download your design to the FPGA:

1. Open the **Devices** view by selecting **View » Devices View**.

2. Make sure your NanoBoard-NB1 is properly connected and switched on. In the **Devices** view, click on the **Live** button and check that the **Connected** indicator is green.

3. In the **Devices** view, click on **Compile**. The red indicator will turn green when a successful compilation takes place. If any error messages display in the **Messages** panel, go back to your schematic and embedded source code, correct any errors, save the files and recompile.

4. Click on **Synthesize**. If the synthesis is completed successfully, a folder called `Generated [config_name]` is created which holds the generated EDIF, VHDL and synthesis log file. If the synthesis does not complete, check the **Messages** panel for errors. The configuration which is used in this example, which we named `demo`, will display in the **Devices** view underneath the Altera Cyclone icon.

5. Click on **Build**. This will step through several processes to ultimately make the Bit file that can be downloaded to the FPGA. The Build stage actually invokes the Vendor tools. It is therefore essential that these tools be installed on your PC (in the case of this tutorial, Altera Quartus tools). You will see the buttons next to the various processes turn green as they are successfully completed. The Build button will turn green when all necessary processes are completed and the *Results Summary* dialog appears. Click on **Close** to continue.

> You can run all stages of the workflow up to and including the current stage by clicking on the arrow icon ⊞ located on the left side of the stage button, e.g. clicking on this icon on the **Program FPGA** button will run all previous stages first.

6. Click on **Program FPGA** to download the bit file to the daughter board's Cyclone FPGA device. Watch the process flow and finally the programming of the FPGA through the JTAG bus.

7. When the Program FPGA process is completed, the LEDs on the NanoBoard-NB1 will be flashing as a bit counter.
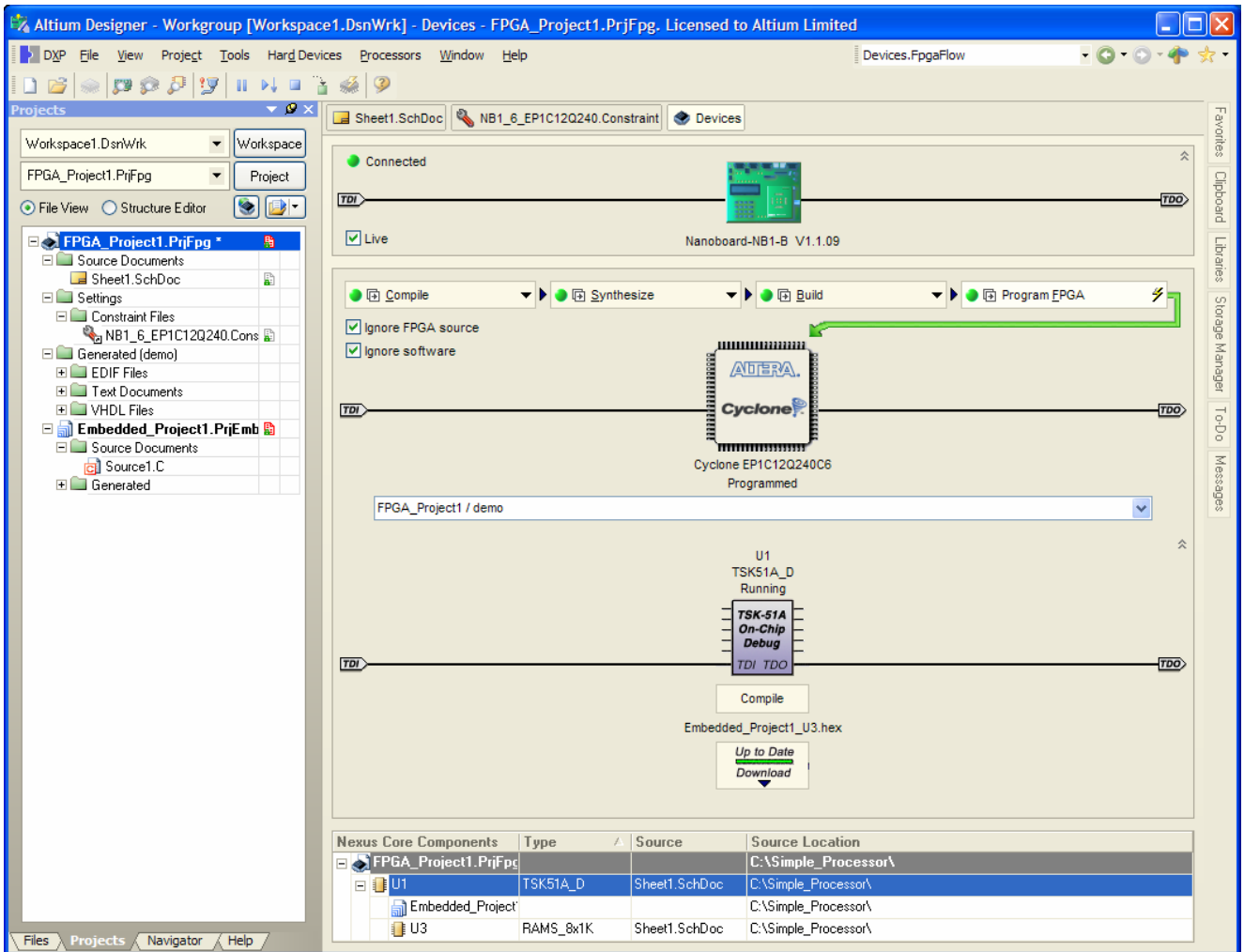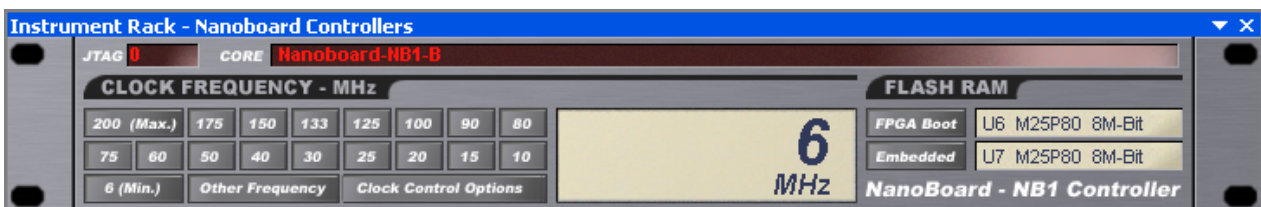

Setting Clock Frequencies

The speed (clock frequency) can be tuned on the NanoBoard-NB1 to slow down or quicken the counter. Let's slow down the counter display.

1. Double-click on the NanoBoard-NB1 icon in the top chain of the **Devices** view.

   The **Instrument Rack – NanoBoard Controllers** panel appears. The default clock frequency is 50MHz.

2. Select another clock frequency, e.g. the slowest (6 MHz), by clicking on the **6(Min)** button.

3. The shifting of the LED display will be slowed down accordingly.

Revision History

Date	Version No.	Revision
29-Jan-2004	1.0	New product release
19-Jan-2005	1.1	Sheet1.SchDoc components updated and SP2 dialog and menu changes.
07-Jul-2005	1.2	Updated for Altium Designer SP4.
12-Dec-2005	1.3	Path references updated for Altium Designer 6
02-Nov-2007	2.0	Document renamed and updated for Altium Designer 6.8.
04-Mar-2008	3.0	Updated for Altium Designer Summer 08