

# Getting Started Tutorial - Processing the Design

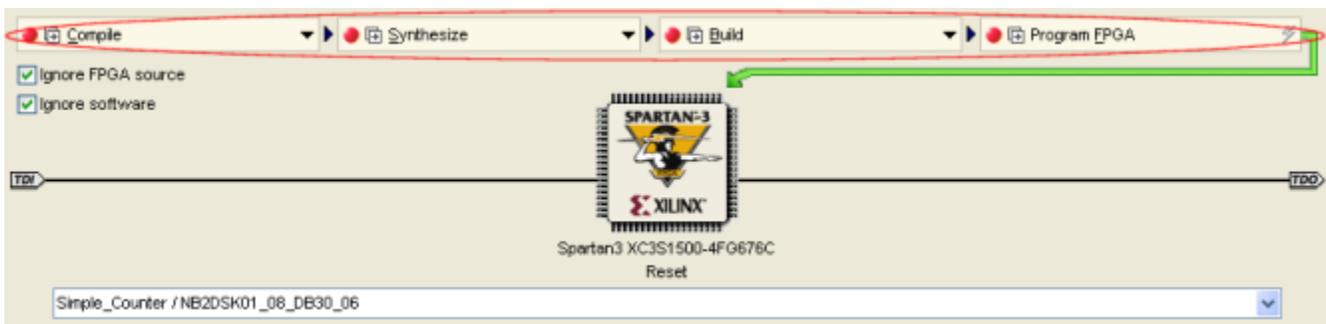
Once the task of capturing an FPGA-based design is complete, the next logical step is to process the source files. This involves compilation and synthesis of the design, to obtain a source netlist file for input to relevant vendor place and route tools.

The process continues, running the place and route tools to ensure that the design will fit within a chosen physical device and to generate an FPGA programming file. This programming file can then be taken to the ultimate step in the processing chain – programming the physical FPGA device with the design.

This entire process flow – from captured source files to programmed physical device – is carried out from the **Devices** view.

- Ensure that the **Devices** view is the active view within Altium Designer (**View » Devices View**).
- Ensure that the **Live** option is still enabled and that the **Connected** indicator is still Green.

Within this view, it is the controls associated with the detected physical FPGA device, in the Hard Devices chain, that we are interested in for this part of the tutorial – collectively referred to as the 'Process Flow' for the physical device (Figure 1).



**Figure 1. Associated Process Flow for the physical FPGA device.**

This Process Flow will only be presented providing the following additional conditions are met:

- The relevant vendor tools, for the FPGA device fitted to the daughter board you are using, are installed.
- The drop-down field below the icon for the device in the chain shows a valid *Project / Configuration* pairing. A valid pairing exists when a project has an associated configuration that contains a constraint file targeting the design to the detected physical device.

Having run the auto-configuration feature, a valid configuration already exists for our example project, containing a constraint file that targets the daughter board device. The *Project / Configuration* entry therefore appears as *Simple\_Counter / ConfigurationName* (e.g. *Simple\_Counter / NB2DSK01\_08\_DB30\_06* in Figure 1).

The Process Flow itself consists of four distinct stages, with the output of each stage required as input to the next. Although the entire Process Flow can be run by clicking directly on the **Program FPGA** button – the last stage in the flow – it is worth considering and using each stage in turn.

You can run all stages of the flow up to and including the current stage, by clicking on the arrow icon located on the left side of the stage button.

- Click on the **Compile** button. This stage is used to perform a compile of the source documents in the associated FPGA project. If our project included processor cores, the associated embedded software projects would also be compiled during this stage. For our simple, non-processor design, this stage will simply use the Design Compiler to verify that the design is free from electrical and drafting errors. As we have previously compiled the project, we will be fine for this stage!

Feedback on the Compile process can be viewed in the **Messages** panel. Once any stage has completed successfully, its associated indicator will turn Green.

- Click on the **Synthesize** button. During synthesis, source documents are translated into intermediate VHDL files which are then synthesized into a top-level EDIF netlist, suitable for vendor Place & Route tools. For the particular FPGA device being targeted, synthesized model files associated with components in the design will be searched for and copied into the relevant synthesis output folder.

If the synthesis is completed successfully, a folder called `Generated [ConfigurationName]` is created which holds the generated EDIF (`Simple_Counter.edf`), VHDL (`Simple_Counter.vhd`) and synthesis log (`Simple_Counter_Synth.log`) files.

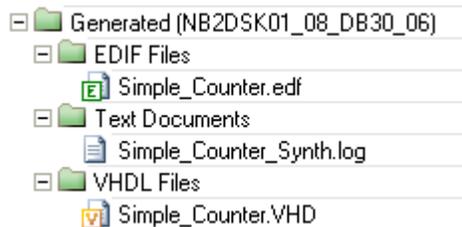


Figure 2. Generated files from the Synthesis stage of the flow.

Feedback on the Synthesis process can be viewed in the **Messages** panel.

- Click on the **Build** button. In this stage, Altium Designer calls and runs the vendor Place & Route tools, to process the design for the target physical FPGA device.

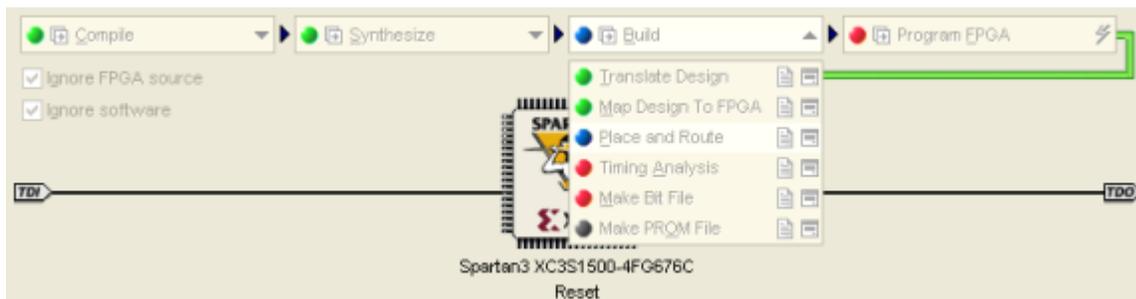


Figure 3. Reaching the Place and Route stage of the overall Build process.

Each of the following five sub-stages will run in turn:

**Translate Design:** uses the top-level EDIF netlist and related synthesized model files, obtained from the Synthesis stage of the Process Flow, to create a file in Native Generic Database (NGD) format.

**Map Design To FPGA:** maps the design to FPGA primitives.

**Place and Route:** takes the low-level description of the design (from the mapping stage) and works out how to place the required logic inside the FPGA. Once arranged, the required interconnections are routed.

**Timing Analysis:** performs a timing analysis of the design, in accordance with any timing constraints that have been defined. If there are no specified constraints – we don't have any for our simple design – default enumeration will be used.

**Make Bit File:** generates the programming file that is required for downloading the design to the physical device.

Feedback on the build progress can be obtained by viewing the **Messages** panel. More detailed vendor-related feedback can be obtained from the **Output** panel (accessed from the menu associated to the **System** button at the bottom of the main design window).

Once the stage has completed successfully, the *Results Summary* dialog will appear. This dialog provides summary information with respect to resource usage within the target device, as well as any timing information. Close this dialog.

The screenshot shows two overlapping windows from an IDE. The top window is the 'Messages' panel, which contains a table of build messages. The bottom window is the 'Results Summary' dialog, which displays resource usage statistics for the target device.

Class	Document	Source	Message	Time	Date
ProcessFlowInfo	Simple_Counter.edi	Build	Running NGDDBuild On C:\Program Files\Altium Designer Summer 08\Examples\Basic FPGA Design Tutorial\ProjectOutputs\Simple_Counter.edi	7:06:16...	20/04/2008
ProcessFlowInfo	Simple_Counter.edi	Build	NGDDBuild - Completed Successfully	7:06:27...	20/04/2008
ProcessFlowInfo	Simple_Counter.ngd	Map	Running MAP On C:\Program Files\Altium Designer Summer 08\Examples\Basic FPGA Design Tutorial\ProjectOutputs\Simple_Counter.ngd	7:06:27...	20/04/2008
ProcessFlowInfo	Simple_Counter.ngd	Map	MAP - Completed Successfully	7:06:31...	20/04/2008
ProcessFlowInfo	Simple_Counter_map...	Place & Route	Running PAR On C:\Program Files\Altium Designer Summer 08\Examples\Basic FPGA Design Tutorial\ProjectOutputs\Simple_Counter_map...	7:06:31...	20/04/2008

Device Resources - Usage Summary		
4-Input LUTs - Logic	13 / 26,624	1%
IO Pins	13 / 487	2%
Slice Flip Flops	10 / 26,624	1%
Slices with only related logic	7 / 7	100%
Slices with unrelated logic	0 / 7	0%
Slices	7 / 13,312	1%
Total 4-Input LUTs - Logic	13 / 26,624	1%

Design Statistics - Timing Summary  
No timing constraints.

Show Results Summary dialog    Note: The Results Summary also appears in the Output panel

Print...    Copy    Report    Close

Figure 4. Accessing build process information and obtaining a final report of the build results.

- Click on the **Program FPGA** button. The programming file obtained from the Build stage is downloaded to the physical FPGA device on the daughter board. Download occurs over the PC's JTAG link to the Desktop NanoBoard, with progress displayed in Altium Designer's Status Bar.

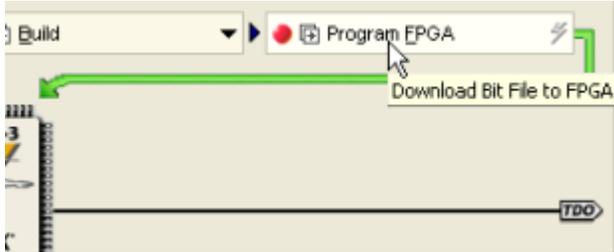


Figure 5. Starting the programming process.

Once the design has been downloaded, the text underneath the physical device's icon in the Devices view will change from *Reset* to *Programmed*. On the hardware side, the 'Program' LED on the daughter board will be lit (Green), confirming that the design has been loaded into the physical device.

- Use switches on the NanoBoard's DIP-switch as follows:
  - *Switch 8*: switch ON to start the counter if previously stopped and/or display the LEDs shifting in from the left
  - *Switch 7*: switch ON to start the counter if previously stopped and/or display the LEDs shifting in from the right
  - *Switch 6*: switch ON to stop the counter. Count will be stopped provided Switches 7 and 8 are OFF.
- Press the 'DAUGHTER BD TEST/RESET' button on the NanoBoard to clear the LEDs.
- Save the project.

You will notice that the User LEDs on the Desktop NanoBoard look to be all on at the same time, which really defeats the purpose of having a twisted-ring counter! This is because the reference clock on the NanoBoard is 20MHz. We need to slow down the clock by a factor of one million to see the LEDs displaying sequentially. In the section [Getting Started Tutorial - Exploring Design Hierarchy](#), we will look at adding some clock division to achieve this and, in doing so, also explore the use of hierarchy in an FPGA design.

## See Also

### [Processing the Captured FPGA Design](#)