Processing the Captured FPGA Design

Summary

This application note describes how to use the Devices view to compile, synthesize, build and ultimately download a design to the physical FPGA device.

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.

The Devices View – an Overview

The **Devices** view can be thought of as 'command central' with respect to processing and debugging a captured FPGA design.

The view can be configured in one of two modes – Live or Not Live. In order to program a physical device with the design and interactively debug it, you must be able to communicate with the device and therefore work 'Live'. When you wish to simply synthesize and run the place and route tools, or investigate the viability of the design with multiple (and different) physical devices, without actually programming a device, then host-to-device communications is not required and you can work in the Not Live mode.

The following sections detail the environment and functionality provided when working in either mode.

Working in Live Mode

Working with the **Devices** view in Live mode enables you to interact with your design in real time. In this mode you are able to not only compile, synthesize and build the design, but use the programming file obtained from running Vendor place and route tools to actually download the design into a physical FPGA device – either on a NanoBoard or a User Board (i.e. a third party development board or a production board). Once the device is programmed, you can make use of the various controls available to debug the design.

The **Devices** view is configured to operate in Live mode by enabling the **Live** option. The mode can also be defined on the **FPGA – Devices View** page of the *Preferences* dialog, accessed by choosing the **Preferences** command from the **Tools** menu. From this page you can also specify whether or not you want to enter Live mode upon starting the **Devices** view. Figure 1 illustrates the **Devices** view when configured in Live mode.

The view is essentially divided into the following four key areas:

- NanoBoard Controllers chain

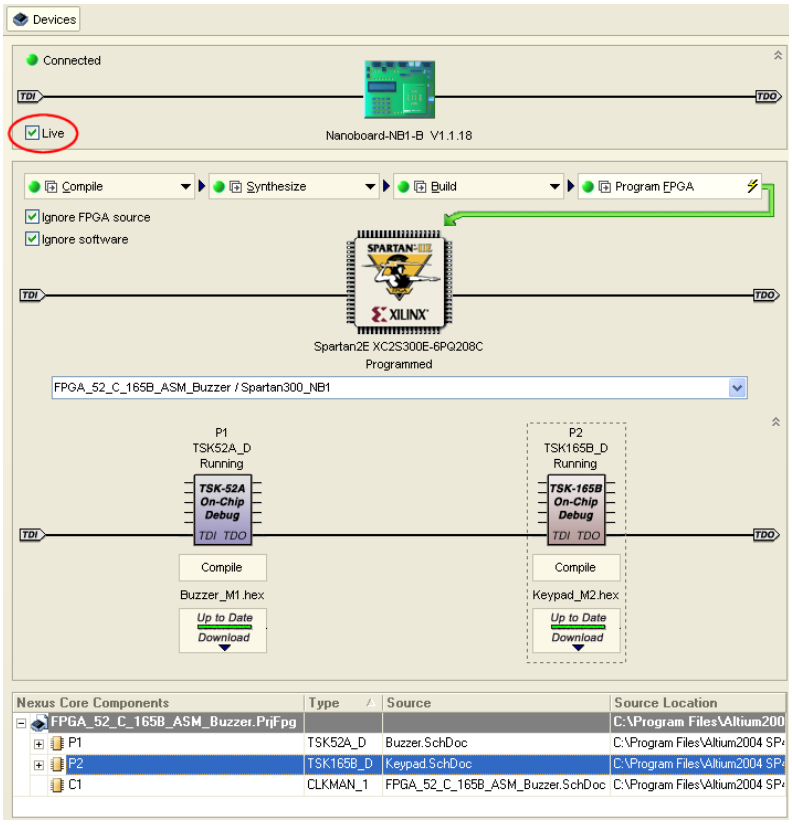


Figure 1. Devices view configured in Live mode.

- Hard Devices chain
- Soft Devices chain
- Nexus Core Component listing.

NanoBoard Controllers Chain

This area of the **Devices** view contains a graphical listing of the NanoBoard Controllers for each NanoBoard detected in the chain. As each NanoBoard has a single Controller, you are effectively seeing the number of powered-up NanoBoards that have been daisy-chained together.

Figure 2 shows the display for a single connected and powered-up NanoBoard.

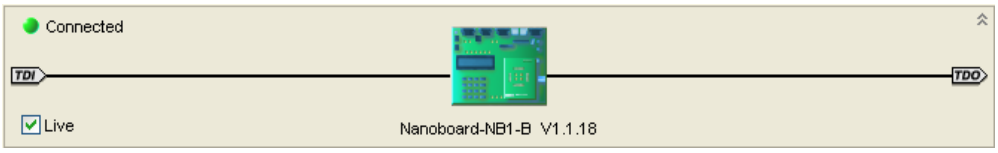


Figure 2. NanoBoard Controllers chain.

The text underneath each NanoBoard Controller icon reflects the name of the NanoBoard upon which the Controller is located, as well as the version of the Firmware that is running in the Controller (e.g. V1.1.18).

Hard Devices Chain

This area of the **Devices** view contains a graphical listing of all physical FPGA devices detected on all powered-up and connected NanoBoards and/or User Boards.

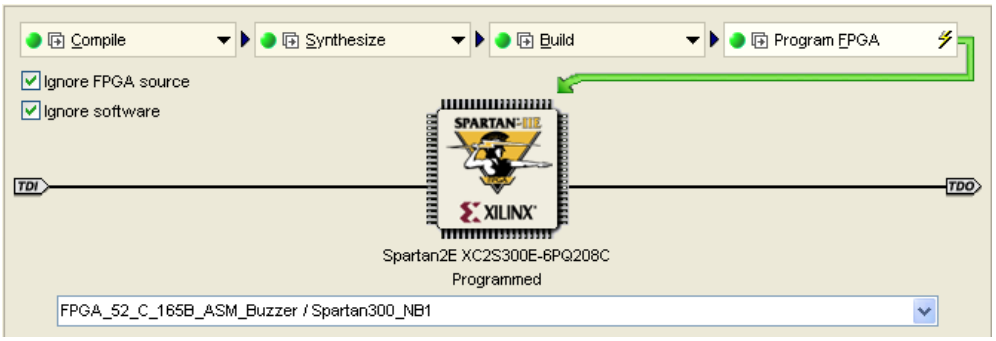


Figure 3. Hard Devices chain.

Each detected NanoBoard will have a single physical FPGA device associated with it (located on a Daughter Board plug-in). A user's Production Board might contain multiple FPGA devices.

Each detected device will appear as an icon in the chain. An FPGA design is targeted to a particular device and board by means of a constraint file. The drop-down field that appears below the icon for a physical device allows you to choose the FPGA project and associated configuration, which in turn contains the constraint file that targets the design to the device.

The text underneath the icon shows the physical device that has been found on the targeted board and the current state of that device (e.g. Reset, Programmed).

Each physical device will also have its own Process Flow associated with it. It is this Process Flow, as described later in the section *Processing a Design*, that allows you to compile, synthesize, build and program the physical device.

Soft Devices Chain

This area of the **Devices** view contains a graphical listing of all Nexus-enabled core devices detected within the FPGA designs that are targeted to the physical devices shown in the Hard Devices chain.

Note: The Soft Devices chain only becomes populated with the Nexus-enabled devices for a design once that design has been compiled – either from the **Project** menu or by running the Compile stage of the Process Flow associated with the physical device in question.

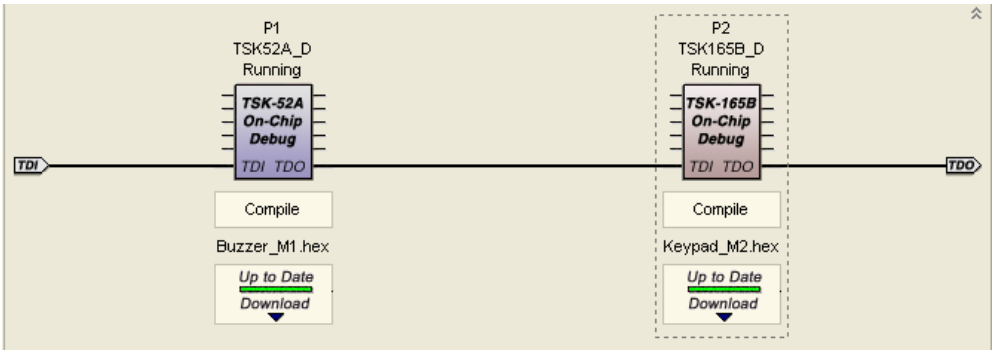


Figure 4. Software Devices chain.

The Nexus 5001 standard is used as the protocol for communications between the host and all devices that are debug-enabled with respect to this protocol. Nexus-enabled devices include all debug-enabled (OCD version) processors and the various virtual instruments that can be used in a design (frequency generators, frequency counters, digital I/O modules and logic analyzers).

With respect to processor devices in the chain, the compiled Hex file for the embedded software is also shown, with controls for compiling and downloading directly to the processor – in order to keep the design up to date.

Nexus Core Component Listing

This area of the **Devices** view lists each FPGA project that is open in the **Projects** panel and the Nexus core components contained therein. These components include processors, instruments and memory blocks.

Double-clicking on an entry will cross probe to the component on the relevant schematic sheet of the design.

Nexus Core Components	Type	Source	Source Location
FPGA_52_C_165B_ASM_Buzzer.PrjFpg			C:\Program Files\Altium2004 SP4\Examples\FPGA Processors\Buzzer
P1	TSK52A_D	Buzzer.SchDoc	C:\Program Files\Altium2004 SP4\Examples\FPGA Processors\Buzzer - TSK165B TS
Buzzer.PrjEmb			C:\Program Files\Altium2004 SP4\Examples\FPGA Processors\Buzzer - TSK165B TS
M1	RAMD_8x1K	Buzzer.SchDoc	C:\Program Files\Altium2004 SP4\Examples\FPGA Processors\Buzzer - TSK165B TS
P2	TSK165B_D	Keypad.SchDoc	C:\Program Files\Altium2004 SP4\Examples\FPGA Processors\Buzzer - TSK165B TS
Keypad.PrjEmb			C:\Program Files\Altium2004 SP4\Examples\FPGA Processors\Buzzer - TSK165B TS
M2	RAMS_12x512	Keypad.SchDoc	C:\Program Files\Altium2004 SP4\Examples\FPGA Processors\Buzzer - TSK165B TS
C1	CLKMAN_1	FPGA_52_C_165B_ASM_Buzzer.SchDoc	C:\Program Files\Altium2004 SP4\Examples\FPGA Processors\Buzzer - TSK165B TS

Figure 5. Nexus components contained within open FPGA projects.

Scanning the Chains – JTAG Communication

In the **Devices** view, in Live mode, there are three distinct chains – for NanoBoard Controllers, Hard Devices and Soft Devices respectively. However, the communications between the host computer and the NanoBoard/User Board consists of a single JTAG chain. So how do we obtain the information for three separate chains in the **Devices** view from this single communications link?

The answer to that question is multiplexing. The three individual chains are multiplexed within the NanoBoard Controller. The information for the currently selected chain is sent via the JTAG communication link and then a software multiplexer is used to channel the information to the relevant chain in the **Devices** view.

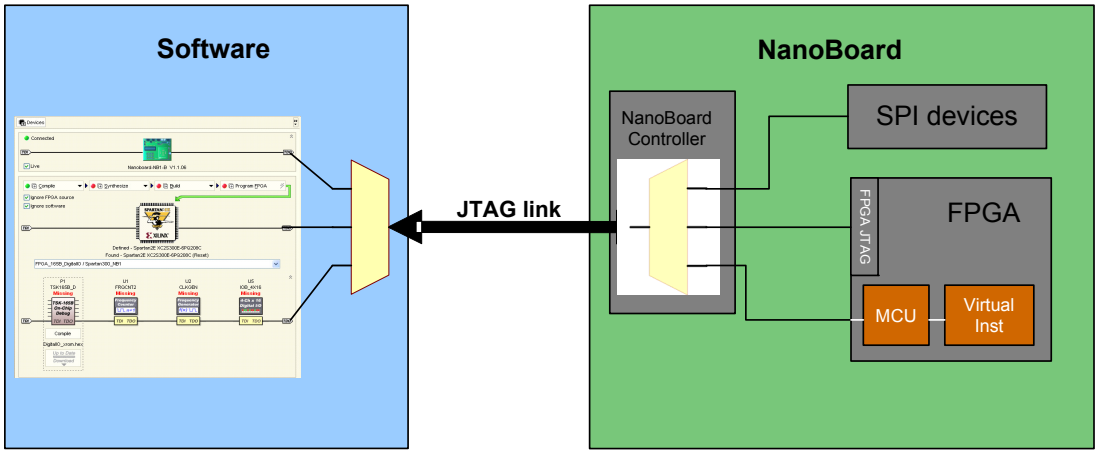


Figure 6. Accessing information for the three chains over a single JTAG link.

When connected to one or more NanoBoards in Live mode, the JTAG chain is refreshed (polled) at a defined interval. This interval, by default, is 10 seconds and can be defined on the **FPGA – Devices View** page of the *Preferences* dialog (**Tools » Preferences**). The scanning involves selection at the respective multiplexers of each separate chain in turn – refreshing the chains from the top down (NanoBoard Controllers, Hard Devices, then Soft Devices).

Each of the three chain regions in the **Devices** view is characterized as consisting of information scanned 'Live' over the JTAG link by adding the familiar JTAG TDI and TDO symbols at the ends of the chain.

Working in Not Live Mode

When the **Devices** view is configured in Not Live mode, by disabling the **Live** option in the view, you are effectively detached from any NanoBoard/User Board that may be connected. This is shown by the effective removal of the NanoBoard Controllers chain from the view, as illustrated in Figure 7.

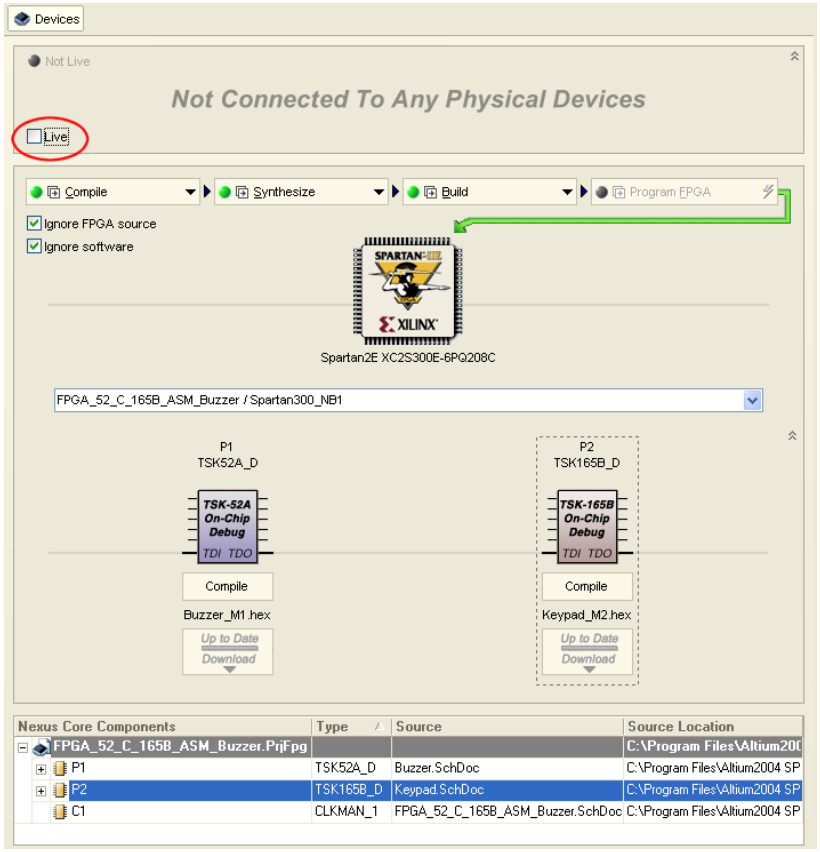


Figure 7. Devices view configured in Not Live mode.

As the Hard and Soft Devices regions of the view are no longer based on information scanned from the physical device(s) on the board(s), they are not depicted as chains in this mode. The tell-tale JTAG TDI and TDO symbols are therefore removed and the chain links grayed-out.

While in this mode, disconnected from any board, you will not be able to program a physical FPGA device. In the Process Flow for each device appearing in the Hard Devices region, the Program FPGA stage will be unavailable.

Testing Physical Device Viability

In Not Live mode, the Hard Devices region of the **Devices** view allows you to compile, synthesize and run the place and route tools for any number of different FPGA devices. This allows you to target different designs (or even the same design) to a number of different physical devices, in order to compare which device best suits your design requirements. This can be especially useful if the NanoBoard does not have a plug-in for the device you wish to use and you want to test it out before using it on a Production Board.

Note: In order to target an FPGA design project to a new physical device, a configuration must be created/selected and an appropriate constraint file added to the configuration, which targets that device.

In Live mode, the Hard Devices are those detected on connected boards. In Not Live mode, the devices are those that you purposefully add to the region. You can add a new device or change/remove an existing one directly from the Hard Devices region. To add, simply right-click anywhere in the region away from an icon and click **Add » Browse** on the floating menu that appears. Alternatively, right-click on an existing device icon and choose **Add » Browse** from the subsequent pop-up menu. If you want to change an existing device, simply right-click on that device and choose the **Change » Browse** command. The *Add Physical Device* or *Change Physical Device* dialog will appear respectively, as shown in Figure 8.

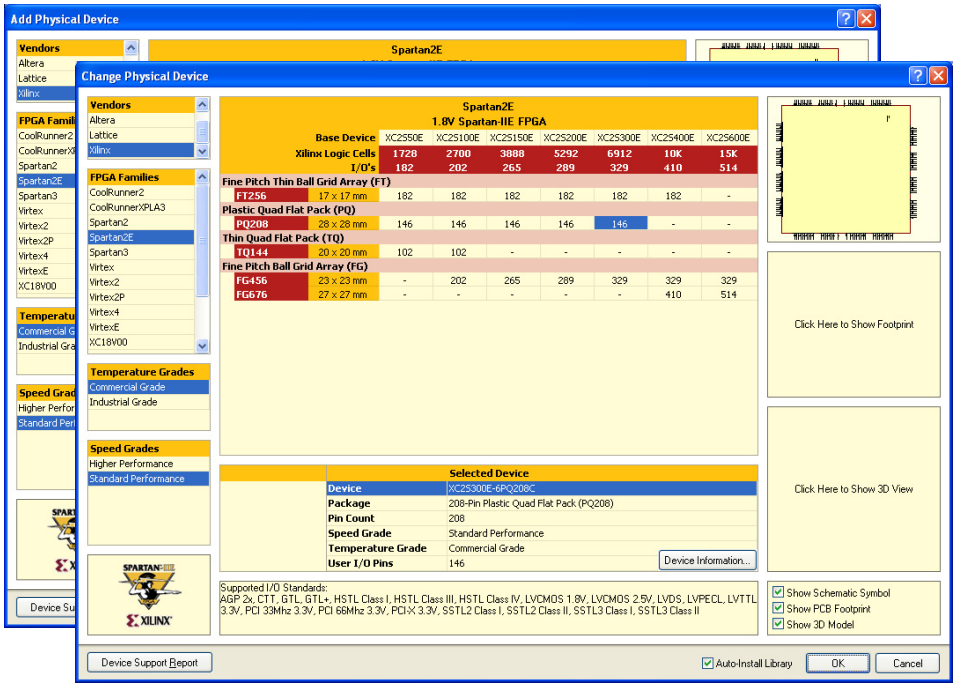
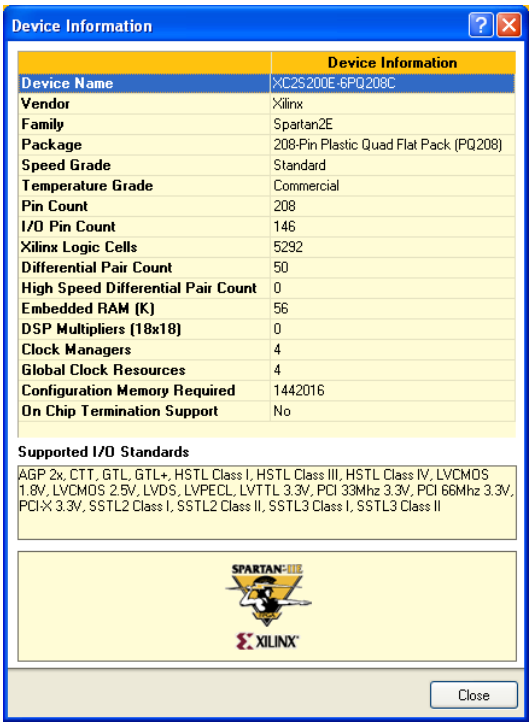


Figure 8. Browsing for the required FPGA Device.

These dialogs allow you to browse the range of physical FPGA devices[1] supported by Altium Designer, available across different device families and from different Vendors. Available (and supported) devices will have a pin number value entered in the main device availability grid. Devices that do not exist are represented by a hyphen character '-'.



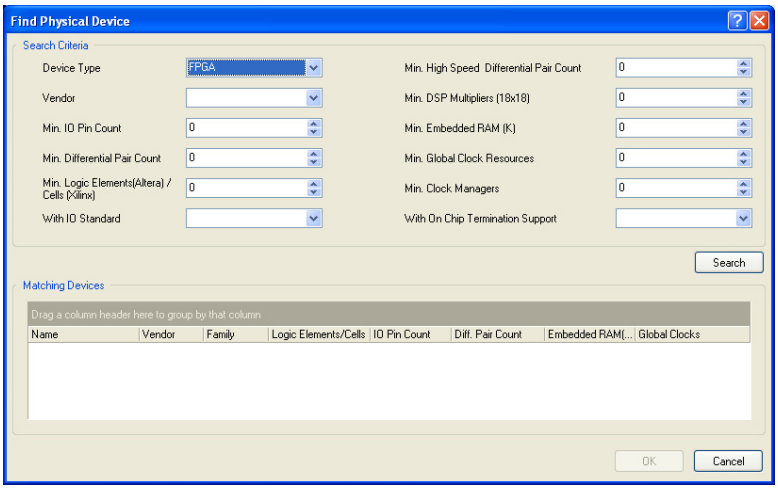
Clicking on the **Device Support Report** button, at the bottom-left of either dialog, will generate a full report (`DeviceSupport.Txt`) listing all physical devices supported by Altium Designer. Devices are listed by vendor and device family. You will be given the option to include or exclude device details (package, pin count, user I/O pins, etc) for the report as required. Once generated, the report will open as the active document in the main design window.

In each device family, key information for considering a device is made available in the **Selected Device** region of either dialog. These factors include:

- Package Type (e.g. Thin Quad Flat Pack, Fine Pitch Ball Grid Array)
- Pin Count
- Speed Grade (e.g. Standard or High Performance)
- Temperature Grade (Commercial or Industrial)
- Number of User I/O Pins.

[1] PLD and configuration devices can also be found in this dialog.

Click on the **Device Information** button associated to this region to access the *Device Information* dialog, which lists detailed information with respect to the currently selected device in the dialog, including: Number of Logic Cells, Embedded RAM and Global Clock Resources.



Searching for a Physical Device

When adding a new device or changing an existing one, you can search for a particular device using the *Find Physical Device* dialog. Access to this dialog is made by right-clicking over an existing device in the Hard Devices region of the view and choosing either **Add » Search** or **Change » Search** from the subsequent pop-up menu that appears.

The **Search Criteria** region of the dialog allows you to specify the scope of the search. Use the **Device Type** field to specify what kind of device to search for – FPGA, CPLD or PROM.

Use the **Vendor** field to constrain the search to devices from a particular vendor. Leave this field blank to search across all supported vendors.

Use the remaining fields to narrow/fine tune the search as required, based on specific device attributes. For those attributes that carry a numerical value, such as IO Pin Count and Global Clock Resources, specify the minimum quantities required. By default, all such attributes are set to zero.

The **With IO Standard** and **With On Chip Termination Support** fields are also, by default, left blank. This means a device will be returned by the search, irrespective of the IO Standards it supports or whether it provides support for on chip termination. Use these fields to restrict the search to a particular IO Standard and/or definite inclusion of on chip termination support respectively.

Note: When searching for a PROM device, only two search criteria will be available – Vendor and Min. Memory Size. All other fields will be grayed-out.

After defining the search criteria as required click the **Search** button – all devices found that fall under the scope determined by the criteria will be listed in the **Matching Devices** region of the dialog, as illustrated by example in Figure 9. Should you wish to search again with different criteria, simply modify the search scope as required and click **Search** again.

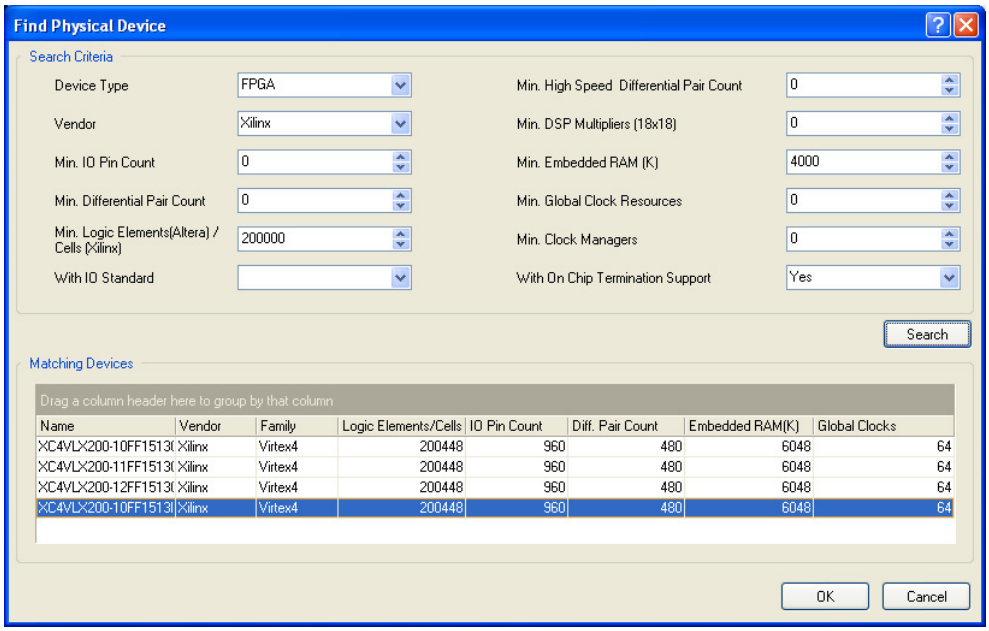


Figure 9. Matching devices returned for an example search.

The information displayed for each device in the list is essentially the same as can be found using the **Device Information** button for that same selected device in the *Add Physical Device* or *Change Physical Device* dialogs. In fact, right-clicking on a selected device in the list and choosing the **Full Device Information** command will present the same *Device Information* dialog discussed previously.

You can control which information gets displayed for the search results – and in what order – using the *Hide/Show Columns* dialog, accessed by choosing the **Show/Hide Columns** command from the right-click menu for the region.

Re-ordering of data columns can be carried out directly within the Matching Devices region by clicking on a column header and dragging it horizontally to the location required. Should you wish to group by one or more specific information columns, simply drag the required column header to the space above the headers. Figure 10 illustrates an example of grouping the search results by Vendor and Family.

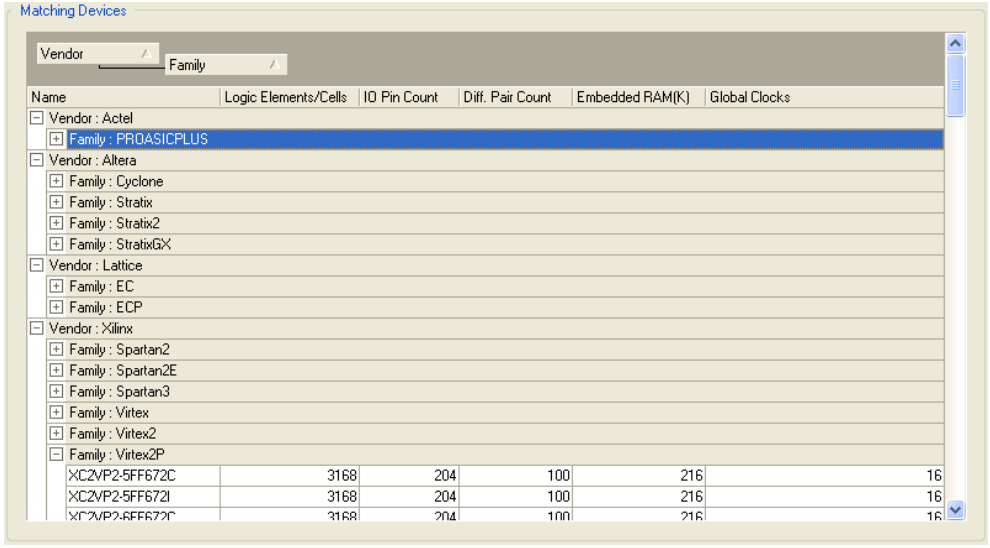
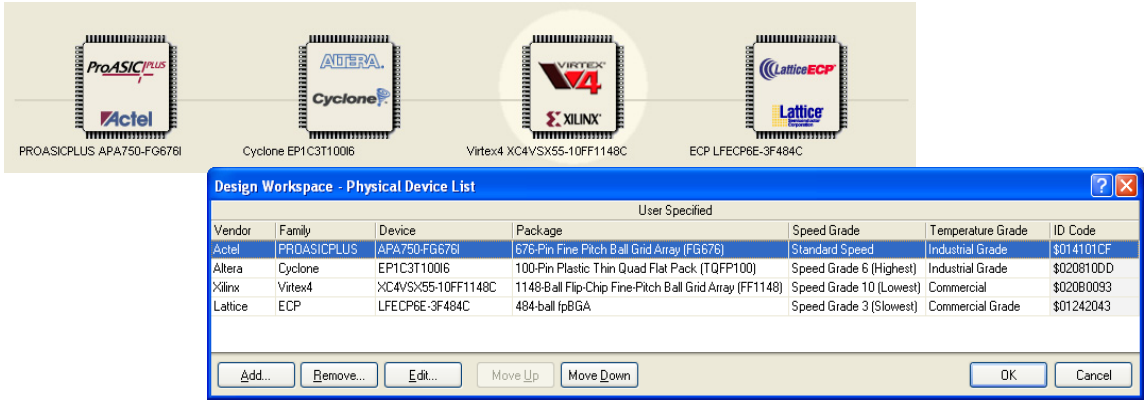


Figure 10. Search results grouped by specific attributes.

By default the last device in the returned list of devices will be selected. Either double-click on the device required or select it and click **OK** – the device will either be added to the Hard Devices region or will replace the currently selected device in the region, depending on whether the dialog was accessed using the **Add** or **Change** command.

Accessing the Physical Device List

The changes you make with respect to the physical devices in the **Devices** view are reflected in the *Design Workspace – Physical Device List* dialog. This dialog is accessed from the **Tools** menu, with the **Devices** view configured in Not Live mode (**Tools » Device List**). As its name suggests, this dialog lists all of the physical devices currently in the Hard Devices region of the view and provides Vendor, family and specific device-related information, as well as the ID of the device – read from a specific ID register, internal to the device.



The dialog also allows you to add, edit or remove devices as well as specifying the order in which devices appear in the Hard Devices region. Top to bottom in the dialog list maps as left to right in the **Devices** view. This last feature can be useful if you have several physical devices that you want to compile, synthesize and build using the **Compile All Bit Files** command, and in a particular order.

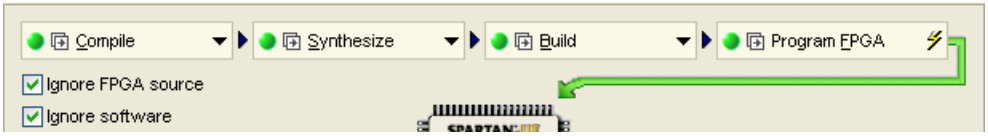
Note: If you have previously worked Live, the dialog will contain an entry for each physical device found on any NanoBoards and/or User Boards connected in the JTAG chain. Conversely, if you have added physical devices to the list whilst working in Not Live mode and then change to Live mode and back again, all physical device entries in the list will be removed and replaced by only those detected in the scanned JTAG chain (i.e. those on connected NanoBoards and/or User Boards).

Processing a Design

When it comes to processing a design, the way in which you use the **Devices** view depends on the particular objective you have in mind and might include:

- verification that the design synthesizes
- use of the Vendor place and route tools to ensure the design will fit inside the chosen physical device
- use of the Vendor place and route tools to generate pin assignments for use in a constraint file
- full processing, with download to a physical device on a NanoBoard or User Board.

Whatever the objective, the stages involved are all carried out using a Process Flow. Each physical device in the Hard Devices region of the view has a Process Flow associated with it consisting of four distinct stages – Compile, Synthesize, Build and Program FPGA.



Pre-processing Preparation

In order to access the Process Flow for a physical device, a suitable project-configuration combination must be available. For a combination to be suitable, a configuration must have been defined in the FPGA project and that configuration should contain a constraint file that targets that particular device.

Suitable project-configuration combinations are automatically detected, where they exist, and are made available in a drop-down below the physical device. Figure 11 illustrates the case where a physical device in the view does not have a suitable project-configuration combination and hence the Process Flow for that device is not available.

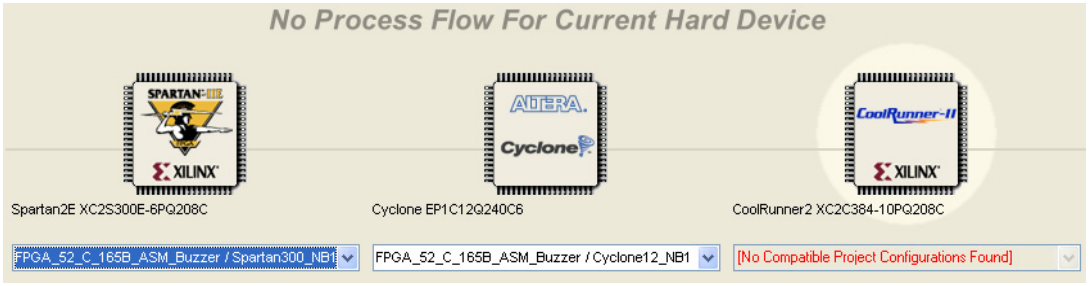


Figure 12. A suitable Project-Configuration combination is required to access the Process Flow.

Constraint Files	Configurations		
Constraint Filename	Cyclone12_NB1	Spartan300_NB1	CoolRunner384_NB1
CoolRunner.Constraint	☐	☐	☑
FPGA_52_C_165B_ASM_Buzzer.Constraint	☑	☑	☑
NB1_6_EP1C12Q240.Constraint	☑	☐	☐
NB1_6_XC2S300E-6PQ208.Constraint	☐	☑	☐

Figure 11. Adding the relevant constraint file to the required configuration.

In this case, a constraint file targeting the CoolRunner-II device would have to be created and then assigned to a configuration using the *Configuration Manager* dialog (**Project » Configuration Manager**).

The suitable project-configuration combination will then be automatically detected and added into the **Devices** view, giving access to the Process Flow for the device.

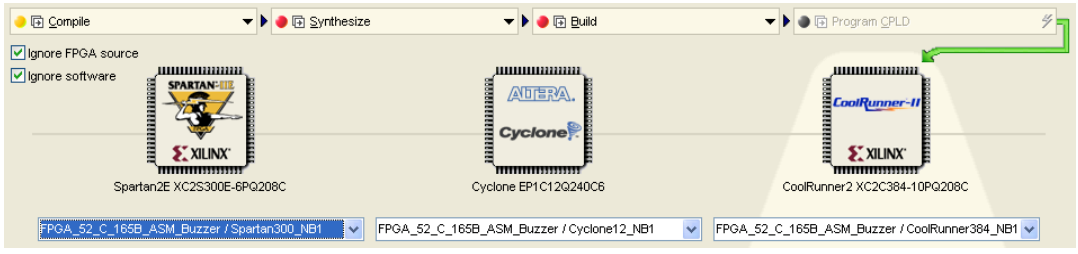


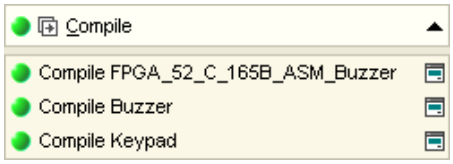
Figure 13. Detecting the required project-configuration combination.

For a detailed description of configurations and constraint files, see the *Design Portability, Configurations and Constraints* article.

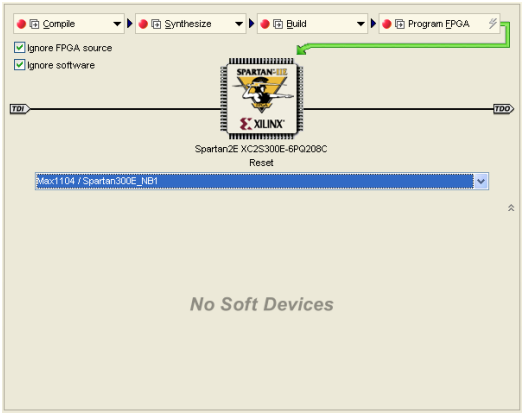
Process Flow Stages

As briefly mentioned earlier, the Process Flow consists of four distinct stages – Compile, Synthesize, Build and Program FPGA. The output of each stage is required as the input to the next, with the final stage performing the download of the design into the physical FPGA device. The following sections briefly discuss these various stages.

Compile



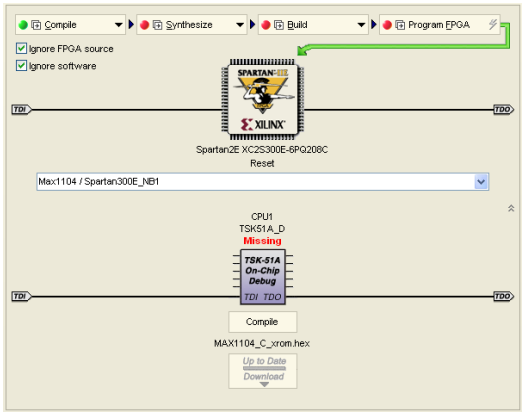
This stage of the Process Flow is used to perform a compile of the source documents in the associated FPGA project. If the design includes any processor cores, the associated embedded projects are also compiled – producing a Hex file in each case.



This stage can be run with the **Devices** view configured in either Live or Not Live mode.

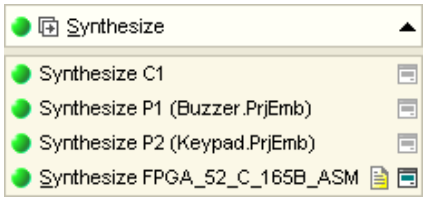
The compile process is identical to that performed from the associated **Project** menu. Running this stage can verify that the captured source is free of electrical, drafting and coding errors.

The source FPGA project(s) must be compiled in order to see Nexus-enabled device entries in the Soft Devices region of the view. When a project has not been compiled at all, the Soft Devices region will display the text: `No Soft Devices`.



When a project is compiled for the first time, information regarding the structure of the project is gathered and stored in a generated Project Structure file (`ProjectName.PrjFpgStructure`). This file is primarily used to present the project's document structure in the **Projects** panel – including any linked sub-projects – but also includes information about the Nexus components used in the project as well as defined configurations for the project. Once this file is generated, the Soft Devices region will become populated with the Nexus devices used. If the Project Structure file is deleted, the region will no longer show soft devices for that project. In this case, simply recompile to regenerate the structure file and subsequently display the soft devices.

Synthesize



This stage of the Process Flow is used to synthesize the compiled FPGA project, as well as any other components that need to be generated and synthesized to a specific device architecture. The synthesis files generated are subsequently used by the Vendor place and route tools during the Build stage of the Flow. Running this stage will determine whether the design is synthesizable or not.

This stage can be run with the **Devices** view configured in either Live or Not Live mode.

The actual steps involved in providing a top-level EDIF netlist and satellite synthesis model files for use by the Build stage can be summarized as follows:

- The cores for any design/device specific blocks used in the FPGA design will be auto-generated and synthesized (e.g. a block of RAM wired to an OCD-version microcontroller for use as external Program memory space). These synthesized models will contain compiled information from the embedded project (Hex file).
- The main FPGA design is then synthesized. An intermediate VHDL file for each schematic sheet in the design will be generated and a top-level EDIF netlist created using these and any additional VHDL source files.
- For the particular physical device chosen, synthesized model files associated with components in the design will be searched for and copied to the relevant output[2] folder. Both System and User presynthesized models are supported.

  The top-level folder for System presynthesized models is the `/Program Files/Altium Designer 6/Library/Edif` folder, sub-divided by Vendor and device family.

  The top-level folder for User presynthesized models is defined on the **FPGA – Synthesis** page of the *Preferences* dialog (**Tools » FPGA Preferences** from any schematic in the FPGA project).

  The following list summarizes the order (top to bottom = first to last) in which folders are searched when looking for a synthesized model associated with a component in the design:

  – FPGA project folder

  – User models top folder\Vendor folder\Family folder

  – User models top folder\Vendor folder

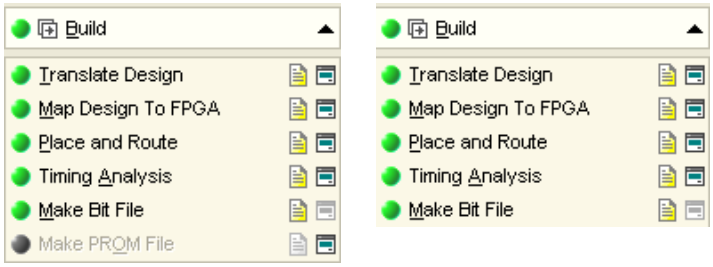
  – User models top folder

  – System models top folder (Edif)\Vendor Folder\Family folder

  – System models top folder (Edif)\Vendor folder

  – System models top folder (Edif).

Build



Build stage of the Process Flow for Xilinx (left) and Actel/Altera/Lattice (right) devices.

This stage of the Process Flow is used to run the Vendor place and route tools. This stage can be run with the **Devices** view configured in either Live or Not Live mode.

Running the tools at this stage can verify if a design will indeed fit inside the chosen physical device. You may also wish to run the Vendor tools if you want to obtain pin assignments for importing back into the relevant constraint file.

The end result of running this stage is the generation of an FPGA programming file that will ultimately be used to program the physical device with the design. There are essentially five main stages to the build process:

- **Translate Design** – uses the top-level EDIF netlist and 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, default enumeration will be used.
- **Make Bit File** – generates the programming file that is required for downloading the design to the physical device.

[2] Generated output files are stored in a folder with the same name as the configuration used for the project, located along the output path defined in the **Options** tab of the *Options for Project* dialog (**Project » Project Options**).

When targeting a Xilinx device an additional stage is available – **Make PROM File**. This stage is used when you want to generate a configuration file for subsequent download to a Xilinx configuration device on a User board.

After the Build stage has completed, the *Results Summary* dialog will appear (Figure 14). This dialog provides summary information with respect to resource usage within the target device, as well as timing information. Information can be copied and printed from the dialog or, alternatively, detailed mapping (`*.Mrp`) and timing (`*.Twr`) reports can be generated. The dialog can be disabled from opening, should you wish, as the information is readily available in the **Output** panel.

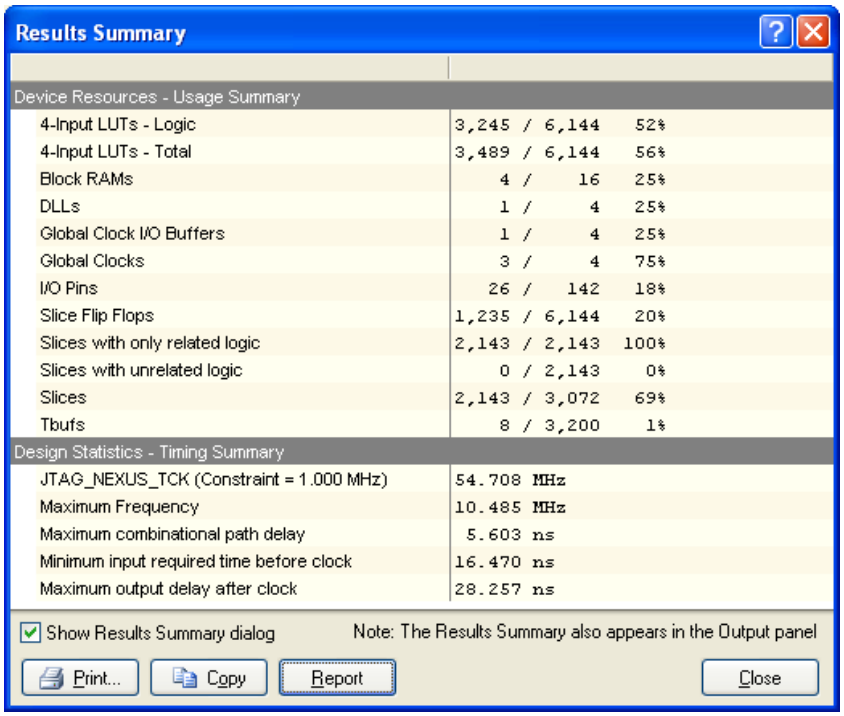
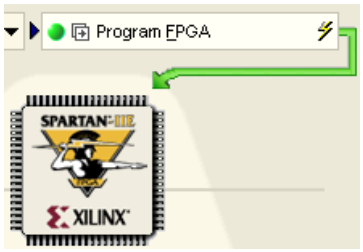


Figure 14. Summarizing device resource usage and design timing statistics

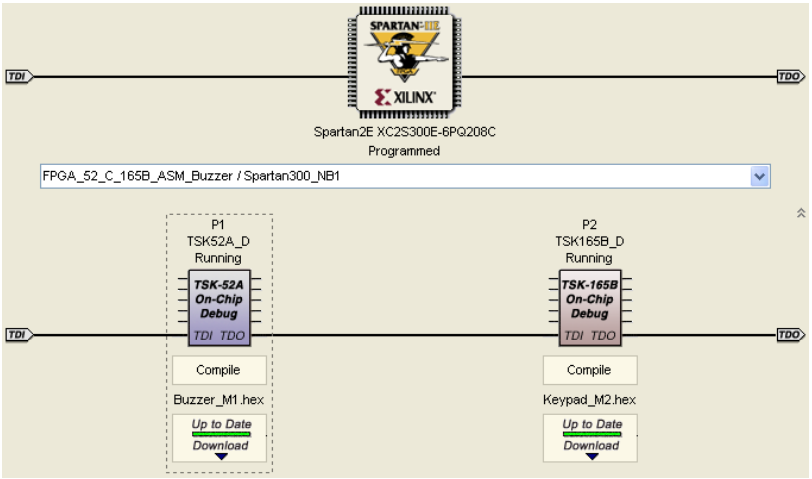
Program FPGA



This stage of the Process Flow is used to download the design into the physical FPGA device on a NanoBoard or User Board. This stage is only available when the **Devices** view is configured in Live mode.

This stage of the Flow can only be used once the previous three stages have been run successfully and an FPGA programming file has been generated.

As the programming file is downloaded to the device via the JTAG link, the progress will be shown in the Status bar. Once successfully downloaded, the text underneath the device will change from `Reset` to `Programmed` and any Nexus-enabled devices on the Soft Devices chain will be displayed as `Running`.



Design Processing – Useful Tips

The following sections highlight useful things to know when processing an FPGA design from within the **Devices** view.

Process Flow States

As each stage in the Process Flow for a physical device is run, the steps in that stage can undergo a transition to various possible states. Each state is color-coded. The full range of possible states is summarized below:

- Grey - **Not Available**. The step or stage cannot be run.
- Red - **Missing**. The step or stage has not been previously run.
- Yellow - **Out of Date**. A source file has changed and the step or stage must be run again in order to obtain up to date file(s).
- Blue - **Running**. The step or stage is currently being executed.
- Orange - **Cancelled**. The step or stage has been halted by user intervention.
- Magenta - **Failed**. An error has occurred while running the current step of the stage.
- Green - **Up to Date**. The step or stage has been run and the generated file(s) are up to date.

The state of the parent stage will reflect the state reached by the step that was last running within that stage.

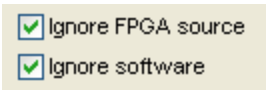
Process Flow Time-Stamp Options

The Process Flow is, by default, automatically refreshed with respect to the state of all files involved (based on their time-stamps). This refresh occurs as part of the automatic polling for the **Devices** view, which can be enabled/disabled on the **FPGA – Devices View** page of the *Preferences* dialog (**Tools » Preferences**). The polling interval can also be defined (default is 10 seconds).

If you want to perform an instant manual refresh of the **Devices** view – and therefore the Process Flow – this can be achieved simply by hitting the **F5** key.

When calculating whether the stages of the Process Flow are up to date or not, the time-stamps of the source files in the FPGA project (and any embedded projects) are used. Comparison is made between the original source files and any modified version of those files.

Two options are provided in the **Devices** view that allow you to control whether FPGA source file (schematics, VHDL files, constraint files) and/or embedded software file (C, ASM) time-stamps are ignored or not, when considering the status of the Process Flow for the physical device.



If ignored, the source files/software files will not be considered when determining whether or not a stage in the Process Flow is up to date.

For example, consider the case where you wish to rebuild the programming file for download to the physical device. The design has previously been compiled and synthesized (status is up-to-date/green in the **Devices** view). A change is made to the top-level source schematic of the FPGA project, maybe a new port has been added. If the option to **Ignore FPGA source** file time-stamps is enabled, the modified schematic will not be considered. Instead, the Build stage of the Process Flow will proceed, using the existing (last generated) synthesis files.

If the option had been disabled, both the Compile and Synthesize stages of the Process Flow would be shown as out-of-date/yellow. In this case, running the Build stage would actually re-run the previous two stages of the flow – enabling the change in the design to be included in the newly built FPGA programming file.

Running a Stage

Each stage of the Process Flow is run by clicking on its corresponding button in the Flow. These buttons are not, however, singular in their functionality. The actual feature accessed depends on where along the button you click. Hovering the cursor along the length of a stage button will reveal the different functionality it provides.

Running the Current Stage Only

Clicking on the textual entry of a stage button will run that stage. If one or more previous stages in the Flow have not been run or are shown as not up to date those stage will also be run, automatically, in order to obtain the input file(s) needed for the current stage.

If previous stages in the Flow are already up to date (Green), only the current stage will be run, using the last generated output from the previous stage.

The current stage will be run in accordance with the state of the **Ignore FPGA source** and **Ignore software** options. If these options are enabled and a change has been made to either a source FPGA project or embedded project document, the previous stage(s) in the Flow will only be re-run if they are not currently in the up-to-date state (Green).

Running all Previous Stages and the Current Stage

Clicking on the icon on a stage button will cause all previous stages to be re-run, in sequence, followed by the current stage itself. This feature equates to a total rebuild up to and including the current stage.

This feature does not take into consideration the state of the two Ignore options – all source files in FPGA and embedded projects are recompiled, in order to obtain the most current, up to date set of input files for use when running the current stage.

Program FPGA Stage – Additional Functionality

The button in the Process Flow for the Program FPGA stage offers the same features as the buttons for the other stages and one addition – the ability to use an existing programming file to program the physical device.

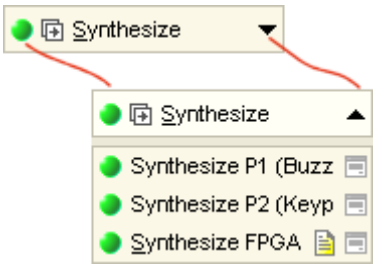
This feature becomes available only after an initial FPGA programming file has been generated (by successful execution of the Compile, Synthesize and Build stages). This allows you to program the device, even if the Process Flow is registering as out of date.

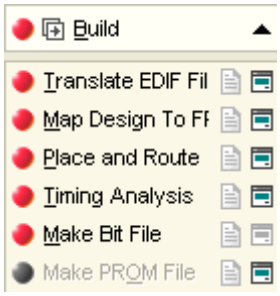
To download an existing programming file to the FPGA, simply click on the lightning bolt icon - .

Accessing and Defini Process Options

Clicking on the status icon (e.g.)or the expand/collapse arrow ( ) will expand a stage, showing its constituent steps.

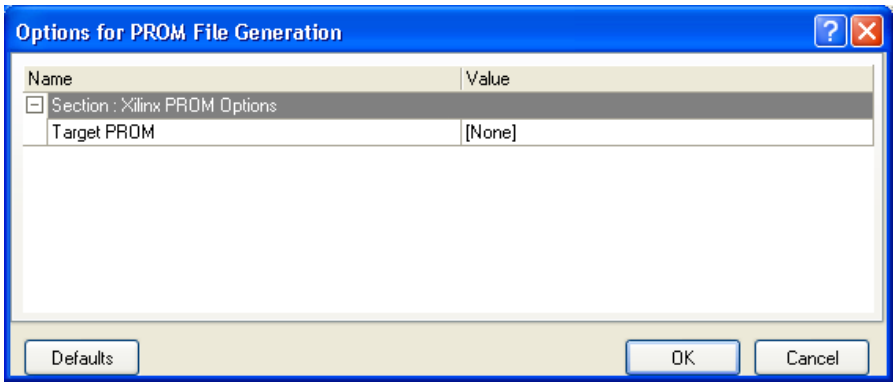
From this expanded view, you are able to view the progress of the stage, when it is running, at a finer level.





Access to the relevant options/setup dialog for a sub-stage (where available) is provided by clicking on the associated icon. For example, consider the case where you want to generate a PROM file for subsequent download to a Xilinx configuration device on a User Board. In the Process Flow associated to the targeted FPGA device, expand the Build section. The last entry in the Build stage menu is **Make PROM File**.

Clicking on the icon, to the far right of this menu entry, will open the *Options for PROM File Generation* dialog, from where you can choose the target configuration device, which resides on the User Board and to which the generated PROM file will be downloaded.



Where a report is available upon running a stage step, access to this report is made by clicking on the associated icon.

Halting the Process Flow

The Process Flow can be halted during the Build stage by pressing the **ESC** key. For the Compile and Synthesize stages, they have their own progress dialogs that allow you to abort and cancel respectively. Once the Program FPGA stage has been entered, you cannot halt the Process Flow.

Viewing Real-Time Progress Information

When running the various stages of the Process Flow, progress information can be viewed using the **Messages** panel. For detailed progress information with respect to the Vendor Place and Route tools, use the **Output** panel (see Figure 15).

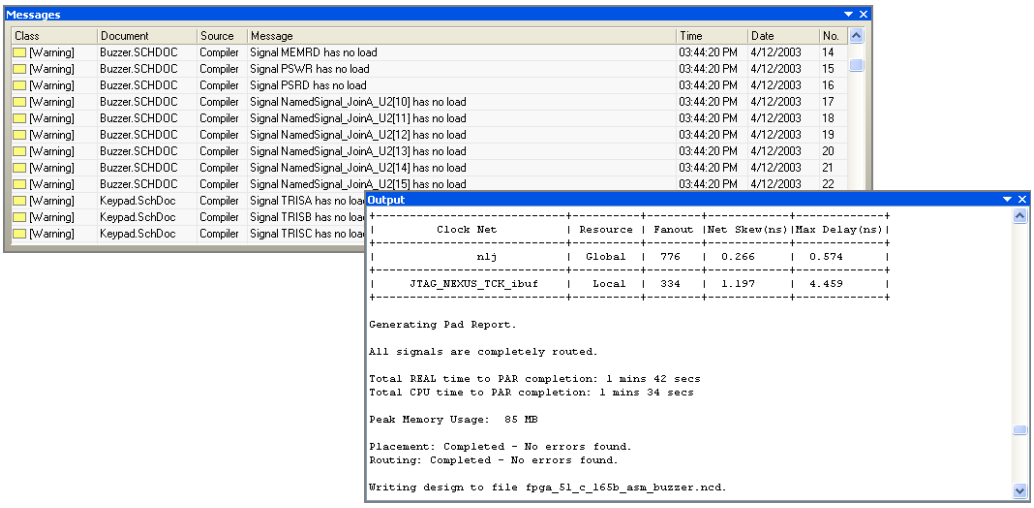
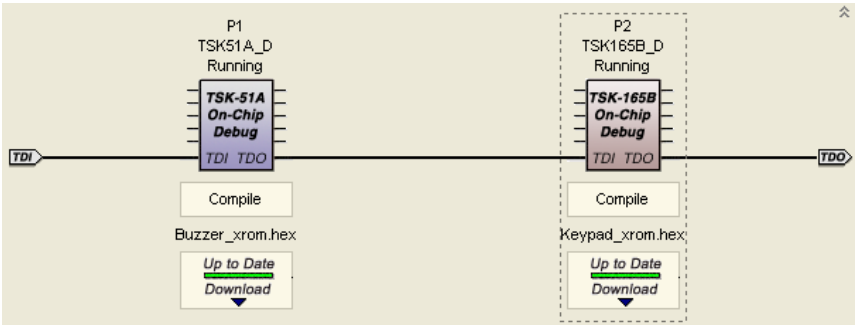


Figure 15. Monitoring progress using the Messages and Output panels.

Updating Embedded Software Independently of the Process Flow

Once the physical device has been programmed with the design, real-time debugging may well highlight the need to change the embedded code running in a processor. To have to recompile and rebuild the design each time a code change is made – essentially running the entire Process Flow again – would be tedious and time-consuming to say the least. To this end, functionality is provided to recompile the embedded project, aside from the Process Flow, in order to obtain an updated Hex file, which can then be downloaded directly to the processor running in the FPGA.

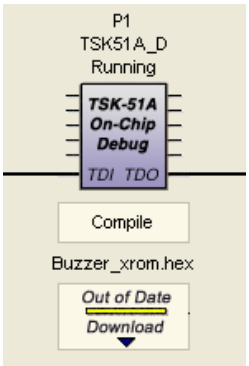
This functionality is provided in the Soft Devices region of the **Devices** view.



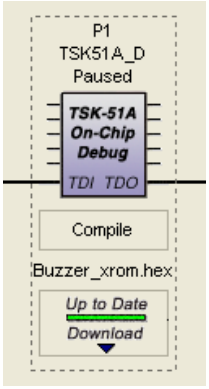
For each processor core detected in the design, controls are available to compile the source embedded project for the code running on the processor and to download it directly – updating the code without having to reprogram the physical device. The name of the compiled Hex file is displayed for reference.

After the design has been downloaded to the physical device, if a change is made to a source file in the embedded project and that file is saved, the indicator bar on the **Up to Date/Download** button will change from green to yellow and the button text will change to `Out of Date/Download`.

To obtain an updated Hex file, simply click on the **Compile** button (above the name of the Hex file). This will recompile the embedded project and generate the latest Hex file. The indicator bar on the



Out of Date/Download button will change back to green and the button text will return to `Up to Date/Download`.



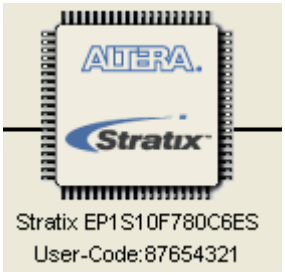
At this point, the Hex file is up to date but is not yet the version of code running in the processor. To download the latest code, simply click the **Up to Date/Download** button. The processor is reset, placed in the Paused state and then the updated code is downloaded to the FPGA.

If the processor was in the Paused state prior to clicking the **Up to Date/Download** button, it will remain in the Paused state after download has completed. If it was running prior to the download, it will be set running after download has completed.

To set the processor back to the 'Running' state, simply right-click on the icon for the processor and choose **Continue** from the subsequent pop-up menu that appears.

Defining a User ID Code for an FPGA Device

Once the design is successfully downloaded, the text underneath the icon for the physical device (in the Hard Devices region) will change from `Reset` to `Programmed`. You can, if desired, define a specific user ID code for the device. Simply enter a dedicated hex code in the **User ID Code for FPGA** field, in the **Options** tab of the *Options for FPGA Project* dialog. By default, the entry in this field is `oxFFFFFFFF`, which will yield `Programmed` in the **Devices** view when the device is successfully programmed. User ID codes can be especially useful when your board has multiple FPGA devices of the same type and you want to easily distinguish which device is which from within the **Devices** view.



When a specific ID code is defined, upon successful programming of a device the text underneath the device's icon will change from `Reset` to `User-Code: n`, where `n` is the corresponding ID defined for that device.

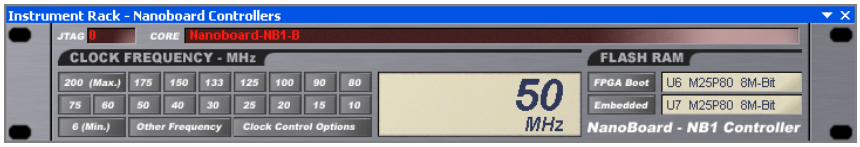
Accessing Device Controls and Debugging

After downloading a design into a physical FPGA device, you can begin to interactively debug the design.

Working in the **Devices** view, configured in Live mode, you can access controls to the devices that appear on the three separate chains. These controls enable you to interact with your design in real-time and, when using a NanoBoard, affords you the luxury of polishing a design before proceeding to the production phase.

Working with the NanoBoard Controller Chain

To access the controls for a NanoBoard Controller, double-click on the icon for that Controller in the NanoBoard Controllers chain. The **Instrument Rack – NanoBoard Controllers** panel will appear, with the chosen Controller added to the rack.



If the physical device is on a User Board that 'hangs' off a NanoBoard, rather than on a Daughter Board plugged-in to the NanoBoard, the controls will still be accessible but will have no effect.

If there are more than one NanoBoards chained together, the NanoBoard Controller chain will reflect each detected (powered-up) board. Each NanoBoard Controller in the chain will only appear in the Instrument Rack after its corresponding icon has been specifically double-clicked.

The Instrument Panel for a Controller provides access to a number of controls, enabling you to:

- Define the frequency of the board clock
- Program Flash Memory for bootstrapping the FPGA device
- Program Flash Memory for embedded use within a design

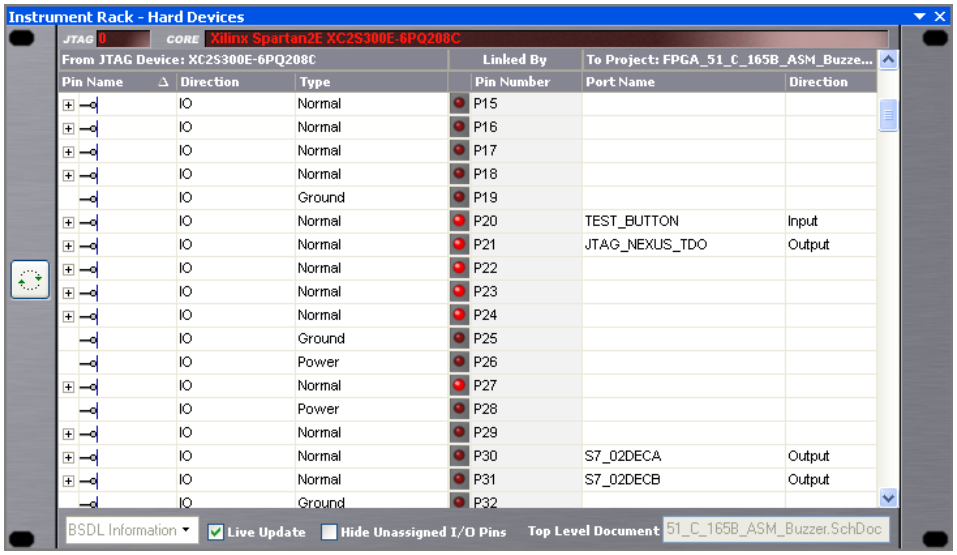
For detailed information on the Instrument Rack – NanoBoard Controllers panel, press **F1** while the cursor is over the (focused) panel.

For detailed information on bootstrapping an FPGA device, refer to the *Bootstrapping the Daughter Board FPGA* application note.

For information on using embedded Flash RAM within a design, refer to the *Utilizing the NanoBoard Flash Memory* application note.

Working with the Hard Devices Chain

To access controls for a physical device, double-click on the icon for that device in the Hard Devices chain. The **Instrument Rack – Hard Devices** panel will appear, with the chosen device added to the rack.



Each physical device in the chain will only appear in the Instrument Rack after its corresponding icon has been specifically double-clicked.

The Instrument Panel provides information concerning the mapping of ports in the FPGA project to physical pins of the device. Information is also given about the polarity and type of each physical pin.

Controls allow you to filter only assigned I/O pins of the device and to also enable live updating. With this feature enabled, a column of LEDs is displayed – one LED per physical pin. As you interact with the design, affected pins will toggle state, which is reflected in the state of the corresponding LEDs on the Instrument Panel.

Enabling the **Live Update** option also allows you to display the state of the nets directly on the project schematic. To do this, simply place a Probe on the net or bus whose state you wish to interrogate.

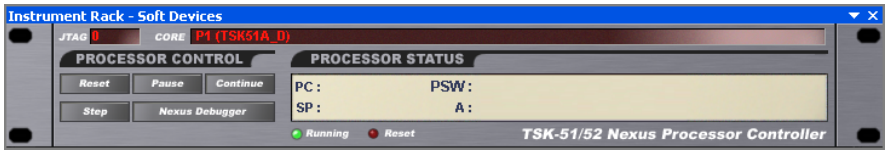
For detailed information on the Instrument Rack – Hard Devices panel, press **F1** while the cursor is over the (focused) panel.

Working with the Soft Devices Chain

The following sections describe briefly the various soft devices and how their corresponding controls are accessed.

Processor Cores

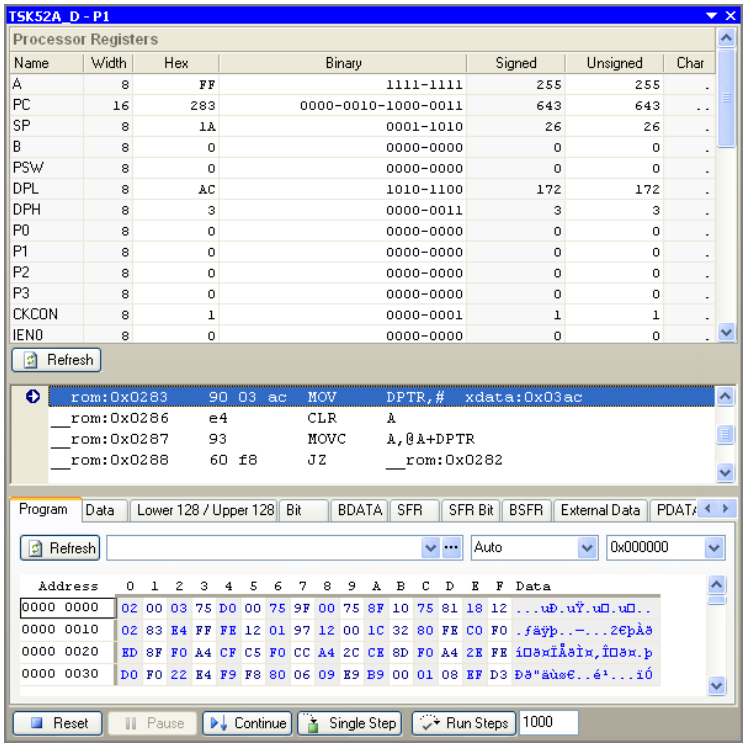
To access the controls for a Nexus-enabled (OCD-version) processor, simply double-click on the icon for that device in the Soft Devices chain. The **Instrument Rack – Soft Devices** panel will appear, with the chosen device added to the rack.



For processors, the Nexus protocol enables you to debug the core through communication with an OCDS Unit. This unit provides debug controls that allow you to:

- interrogate the processor's registers
- define/control hardware breakpoints (if supported)
- read/write Program and/or Data memory spaces.

The debug controls for a processor used in a design can be accessed from its Instrument Panel by clicking the **Nexus Debugger** button. A dedicated debug panel for the processor will appear, which in turn allows you to interrogate and to a lighter extent control, debugging of the processor and its embedded code, notably with respect to the registers and memory.



For more information on the content and use of processor debug panels, press **F1** when the cursor is over one of these (focused) panels.

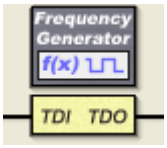
Figure 16. Processor debugging using the associated processor debug panel (TSK52A_D shown)

Virtual Instruments

Virtual instruments are Nexus-enabled devices that you can place in your design to enable you to generate/monitor signals within the design, rather than being limited to signals that only appear at the physical pins of the device. The following sections overview the various instruments available.

Frequency Generator

To access the controls for a Frequency Generator, simply double-click on the icon for that device in the Soft Devices chain. The **Instrument Rack – Soft Devices** panel will appear, with the chosen device added to the rack.

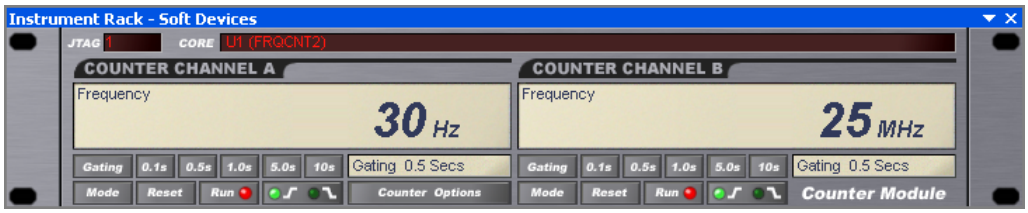


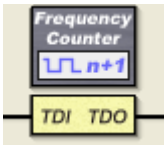
The Frequency Generator instrument takes a reference clock as its input (time-base) and produces output frequencies that are integer divisors of this time base frequency.

For detailed information on the Frequency Generator, see the *CLKGEN Frequency Generator* core reference.

Frequency Counter

To access the controls for a Frequency Counter, simply double-click on the icon for that device in the Soft Devices chain. The Instrument Rack – Soft Devices panel will appear, with the chosen device added to the rack.



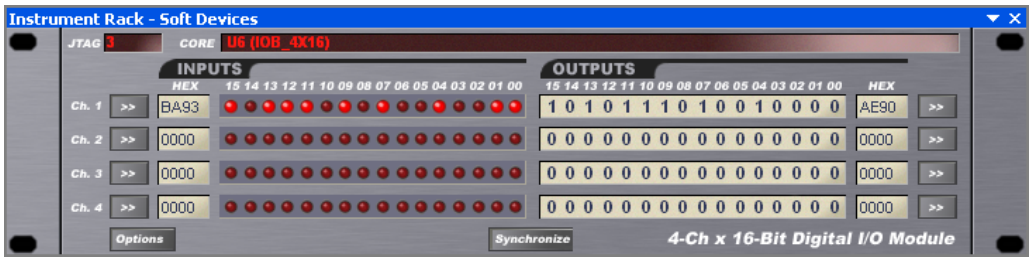


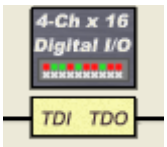
The instrument provides a two-channel, three-mode frequency counter. For each input signal, the device counts the number of edges – rising or falling – detected within a specified gating period. Depending on the mode of operation selected, each channel can display the frequency of the signal, its period, or the total number of edges counted.

For detailed information on the Frequency Counter, see the *FRQCNT2 Frequency Counter* core reference.

Digital I/O Module

To access the controls for a Digital I/O Module, simply double-click on the icon for that device in the Soft Devices chain. The Instrument Rack – Soft Devices panel will appear, with the chosen device added to the rack.

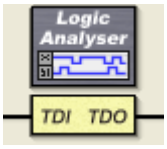




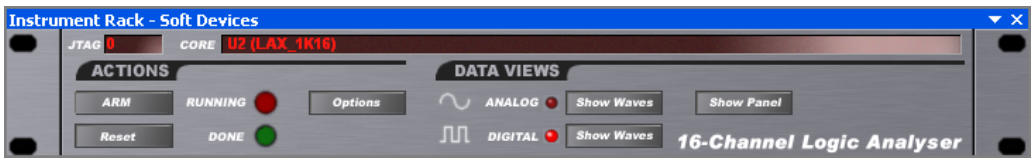
A range of 1, 2 and 4-channel instruments are available, catering for 8- and 16-bit digital I/O. Each device features separated input and outputs, with the ability to synchronize the two.

For detailed information on the Digital I/O Module, see the *IOB_x Digital I/O Module* core reference.

Logic Analyzer



To access the controls for a Logic Analyzer, simply double-click on the icon for that device in the Soft Devices chain. The **Instrument Rack – Soft Devices** panel will appear, with the chosen device added to the rack.



A range of 8- or 16-channel logic analyzer instruments are available. The device family includes analyzers with pre-defined storage memories for captured data, as well as analyzers that provide a memory interface – allowing you the freedom to connect a block of RAM, the size of which is determined by an address bus of up to 20 bits.

For detailed information on the Logic Analyzer, see the *LAX_x Logic Analyzer* core reference.

Revision History

Date	Version No.	Revision
20-Jan-2004	1.0	New product release
01-Jul-2005	1.1	Updated for Altium Designer SP4
12-Dec-2005	1.2	Path references updated for Altium Designer 6
28-Feb-2008	2.0	Updated for Altium Designer Summer 08
17-Aug-2011	-	Updated template.