XILINX®

XAPP352 (v1.2) December 19, 2001

Utilizing a User Constraint File for CoolRunner XPLA3 CPLDs

Summary

This application note provides an introduction to the capabilities and functionality of the User Constraint File (UCF) for CoolRunner™ XPLA3 CPLD designs in WebPACK™ Project Navigator.

Introduction

For design entry in ABEL, VHDL, or Verilog, only a few constraint options can be embedded in the source code. In this case, designers have the option to use a UCF. For designs in ABEL, VHDL, Verilog, external netlist (EDIF), schematic, or another design entry, a UCF provides an external means for entering fitter constraint parameters in one design file.

User constraint files allow designers the flexibility to manually control some of the following attributes in a CoolRunner XPLA3 CPLD design:

- Pin and node assignments
- Slew rate control
- Fit options
- Input pin characteristics
- Initial register state
- Global clock use

Many design constraints from the list above are software selectable during implementation. The WebPACK Project Navigator GUI allows some constraints to be generally set for the device. A UCF allows designers to set characteristics for an individual signal in a design. This application note provides an overview of the options available for setting design constraints in WebPACK Project Navigator.

Please note that several syntax and help guides for a UCF exist within the WebPACK Project Navigator help index. These help files are updated with each release of software and provide a reliable source of information. The WebPACK help menu can be located on a PC from the **Start | Programs | Xilinx CPLD WebPACK | Help and Technical Support** menu.

UCF Specifications

Table 1 illustrates some of the CoolRunner XPLA3 attributes that can be specified in the UCF. More information on these attributes are discussed in the sections below.

Table 1: **CoolRunner Attributes**

Type of Constraint	Description
Pin Locking	Assign a specific signal to a particular pin or function block/macrocell location
Node Locking	Assign a specific signal to a function block/macrocell location
Pin Prohibiting	Prohibit any signal from being assigned to this pin
Slew Rate	Select the slew rate for a specific output signal
Initial State	Control the state of a specific output register on power up

Table 1: **CoolRunner Attributes *(Continued)***

Type of Constraint	Description
Input Pullup	Enable the internal pullup on input pins in the device
Signal Optimization	Enable or disable the reduction of logic for a signal
Global Clock	Assign a specific clock signal to the global clock network

Constraint Priority

Table 2 describes the constraint options that are available in the UCF, WebPACK Project Navigator GUI, or design source code. When multiple design constraints are specified, Table 2 summarizes the priority implementation.

Table 2: **Constraint Option Support and Priority**

Type of Constraint	UCF	WebPACK Project Navigator GUI	Design Source Code[1]
Pin Locking	Supported	N/A	Supported (Highest Priority)
Node Locking	Supported	N/A	N/A
Pin Prohibiting	Supported	N/A	N/A
Slew Rate	Supported	Supported (Global Setting) (Lowest Priority)	Supported (Highest Priority)
Initial State	Supported	Supported (Global Setting) (Lowest Priority)	Supported (Highest Priority)
Input Pullup	Supported	N/A	N/A
Signal Optimization	Supported	N/A	Supported (Highest Priority)
Global Clock	Supported	N/A	Supported

Notes:

1. For more information on setting design constraints in the VHDL, Verilog, or ABEL design source code, please refer to the WebPACK Project Navigator **Help | Online Documentation | CPLD Attributes**.

Specify UCF

The UCF can be edited directly from within WebPACK Project Navigator as shown in Figure 1 under the Design Entry Utilities.

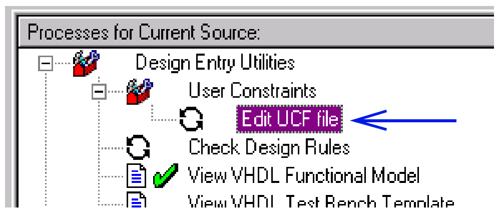
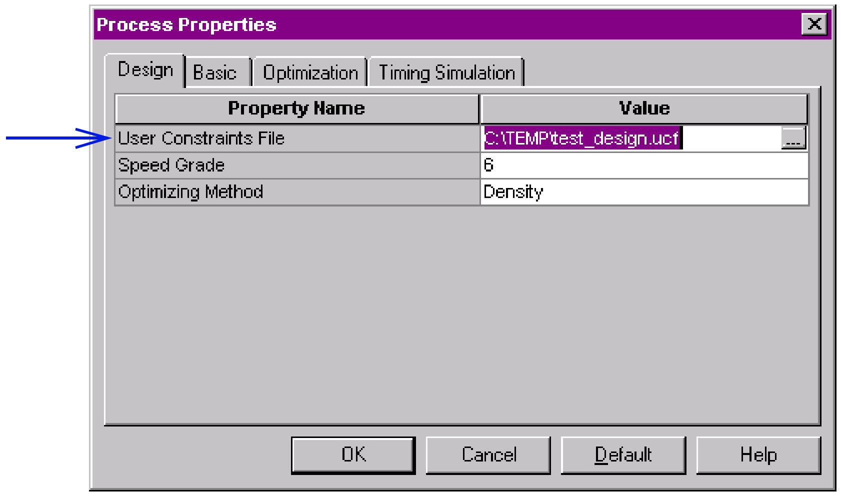

Processes for Current Source:
Design Entry Utilities
User Constraints
Edit UCF file
Check Design Rules
View VHDL Functional Model
View VHDL Test Bench Template

Figure 1: **Link to Edit UCF**

The UCF that is opened when this option is invoked is specified from the **Implement Design Process Properties** window as shown in Figure 2. If no file is specified, editing a UCF file as shown in Figure 1 will create and open a UCF associated with the selected design.



Property Name	Value
User Constraints File	C:\TEMP\test_design.ucf
Speed Grade	6
Optimizing Method	Density

Figure 2: **Specification of Constraint File**

Pin Locking

Pin locking is performed in a number of ways during design implementation. Pins can be assigned in one of three methods:

- Specified in the source code for ABEL, Verilog, or VHDL designs
- Automatic assignment in WebPACK Project Navigator
- Individual pin specification in the UCF

Pin locking is directly affected by the options available for XPLA3™ I/O and macrocell configurations. Please refer to **WP105: CoolRunner XPLA3 Architecture Overview** for more information on the XPLA3 architecture. In XPLA3 devices, each function block has 16 macrocells. The number of macrocells available as I/O is package dependent. The number of available function blocks will depend on the size of CoolRunner device being used. The available configurations for each macrocell in the device can be specified in the UCF, the design source code, or done automatically in WebPACK Project Navigator, in one of five ways:

- Fast input register
- Combinational output signal
- Buried combinational signal
- Buried registered signal
- Registered output signal
- Fast input register and buried combinational signal

Figure 3 illustrates a macrocell register that can function as a fast input register, with the preceding combinational logic preserved and fedback to the interconnect array as a buried combinational node.

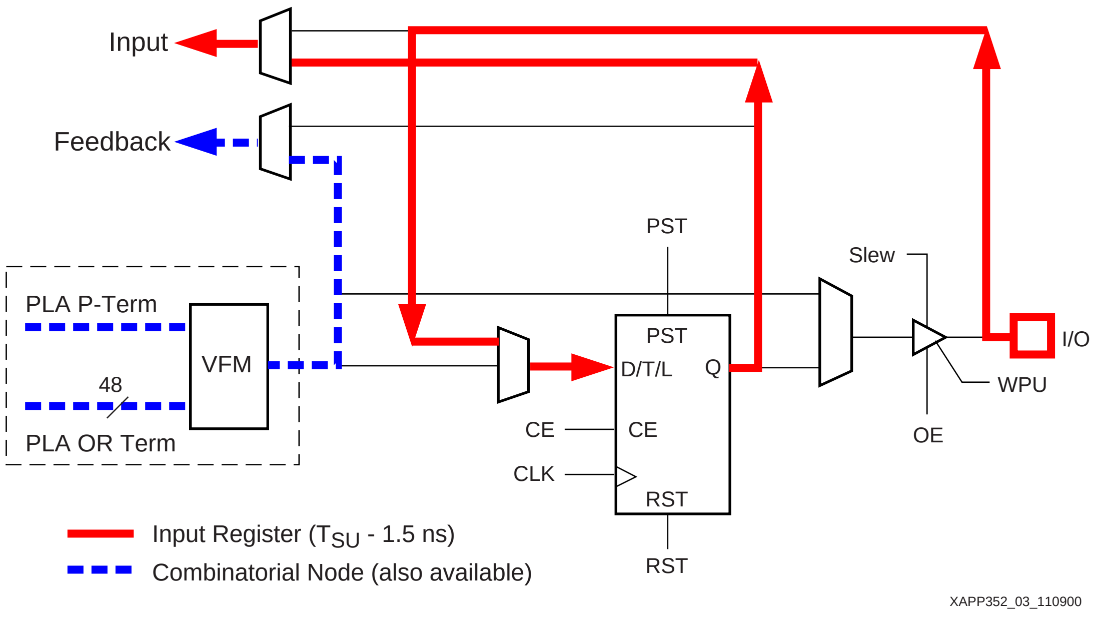


Figure 3: **XPLA3 Macrocell Input Register and Node Configuration**

Automatic Pin Locking

WebPACK Project Navigator provides the option for automatic pin locking after a design has been compiled and synthesized. Figure 4 shows how to invoke this option if so desired.

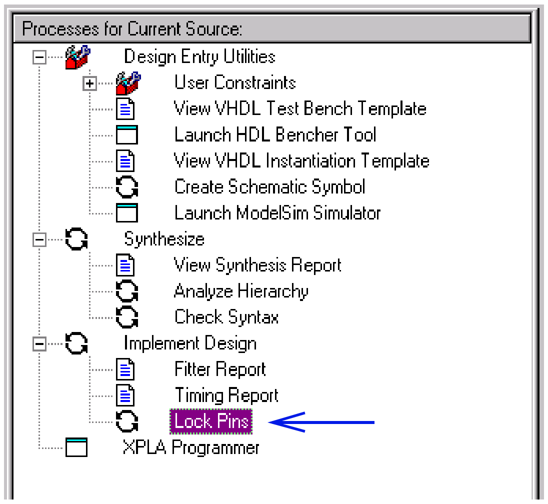


Figure 4: **Locking Pins Automatically**

When starting a design, it is recommended to run the automatic lock pins function after all design and fitter changes have been made. The automatic lock pins option creates a UCF for the selected design file. This file needs to be specified in WebPACK Project Navigator as shown in Figure 2. Refer to section "Pin and Node Locking Options" on page 6 for keeping pin assignments.

If a different pinout is desired, it is then easier to go back and manually edit the UCF. If pin locations have been changed, go back and recompile the design.

UCF Pin Locking

Input and output signals can be assigned to a specific pin in either the UCF or design source code. Table 3 illustrates the syntax and an example for locking a signal to a specific location in an XPLA3 UCF. To assign a signal to a particular pin, the pin number must be specified. The pin numbers that are available are package dependent and more information is available from the XPLA3 product data sheet.

Table 3: **Pin Locking**

Syntax	Examples

NET <signal_name> LOC=Pnn;
Note: nn is the pin number as described in the data sheet for PC, PQ, TQ and similar packages
	NET clk LOC = P10;
	NET qout<0> LOC = P20;

NET <signal_name> LOC=RC;
Note: RC is the row and column pin number specification as described in the data sheet for CS, BG, and similar packages
	NET clk LOC = C8;
	NET qout<1> LOC = G2;

In the WebPACK Project Navigator UCF, signals can also be assigned to a specific function block. The syntax for this assignment is shown in Table 4.

Table 4: **Pin Assignment to a Function Block**

Syntax	Examples

NET <signal_name> LOC=FBm;
Note: m is the function block number from 1 to x depending on device size
	NET clk LOC = FB1;
	NET clk LOC = FB01;
	NET qout<0> LOC = FB12;

For more information on the architectural features of the XPLA3 CPLD family that enable flexible pin locking capabilities, please refer to **XAPP332: Pin Locking in CoolRunner XPLA3 CPLDs**.

Node Locking

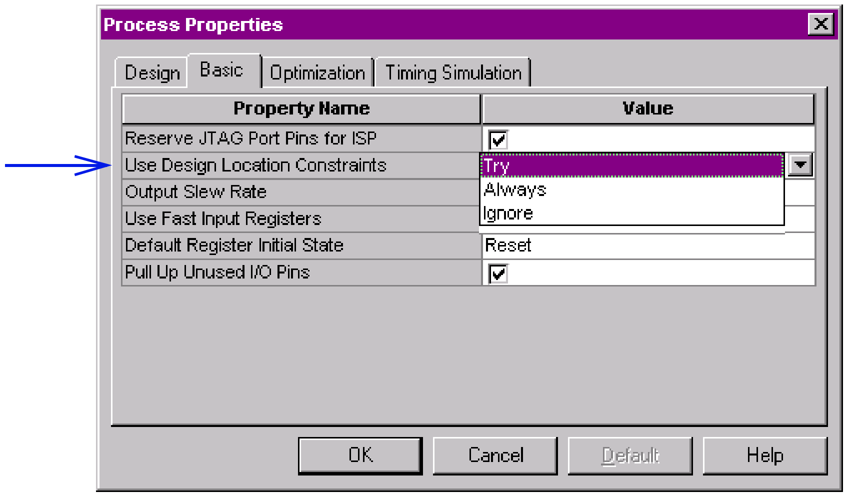
When fitting a design, it may also be desirable to assign a signal to a specific macrocell. For designs that need to lock signals to macrocells, the node locking functionality in the UCF is available. To lock a node, the signal must be assigned to a specific function block and macrocell number. Depending on the size of the part being targeted, function block numbers range from 1 to x and macrocells are numbered from 1 to 16. The function block and macrocell numbers and any corresponding pin numbers are available in the CoolRunner data sheets. Table 5 shows the syntax and an example for using node locking in a design.

Table 5: **Node Locking**

Syntax	Examples
NET <signal_name> LOC=FBm_n; Note: m is the function block number from 1 to x depending on device size and n is the macrocell number ranging from 1 to 16.	NET clk LOC = FB1_4; NET qout<0> LOC = FB12_9; NET qout<1> LOC = FB12_10;

Pin and Node Locking Options

In WebPACK Project Navigator, the fitting process interpolates the UCF. The **Implement Design Process Properties** window allows selecting the option to ALWAYS, TRY, or IGNORE pin and node settings in the UCF or source code as shown in Figure 5. The ALWAYS attribute forces the fitter to keep all pin and node assignments when targeting the design to a particular device. If it is unable to fit the design it will notify the user in an error message. The TRY attribute uses the settings specified in the design and/or UCF when first fitting the design; if the fitter is unable to fit the design it warns the user, then makes modifications as necessary. The IGNORE option discards any settings from the design and/or UCF. Make sure the appropriate setting is selected as shown in Figure 5.



Property Name	Value
Reserve JTAG Port Pins for ISP	✓
Use Design Location Constraints	Try (Always, Ignore)
Output Slew Rate	
Use Fast Input Registers	
Default Register Initial State	Reset
Pull Up Unused I/O Pins	✓

Figure 5: **Using Design Location Constraints**

Pin Prohibiting

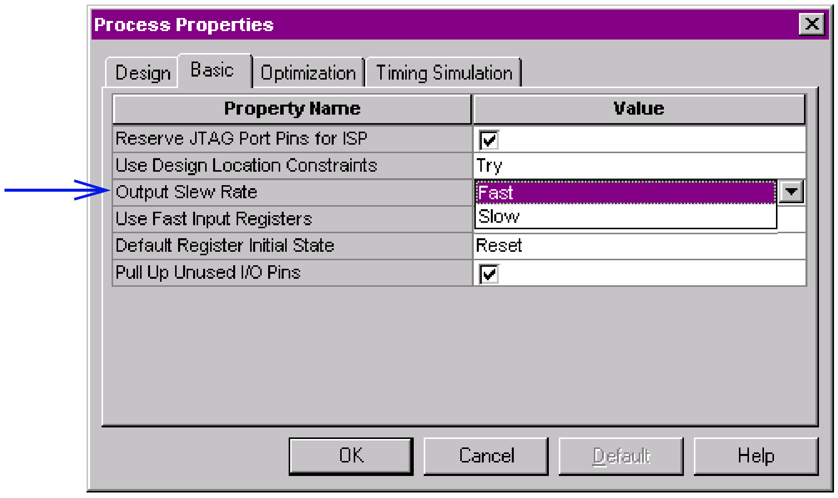
This feature of the UCF allows the designer to prohibit specific pins from being assigned to signals in the design. This attribute differs from package to package as shown in Table 6 below. Note the PROHIBIT attribute can also accept a comma separated list of pins. This command can not prohibit a range of locations. Also note, the PROHIBIT command has higher priority over any signal pin or node assignments in the design.

Table 6: **Pin Prohibit Specification**

Package	Syntax	Example
PC, PQ, TQ and similar packages	**config PROHIBIT = Pnn;** (nn in the pin #)	config PROHIBIT = P9, P12, P15;
CS, BG, and similar packages	**config PROHIBIT = RC;** (Row, Column)	config PROHIBIT = G2;

Slew Rate

The output slew rate for all signals in a design can be specified in the **Implement Design Process Properties** window as shown in Figure 6.



Property Name	Value
Reserve JTAG Port Pins for ISP	✓
Use Design Location Constraints	Try
Output Slew Rate	Fast (Fast / Slow)
Use Fast Input Registers	
Default Register Initial State	Reset
Pull Up Unused I/O Pins	✓

Figure 6: **Controlling Output Slew Rate**

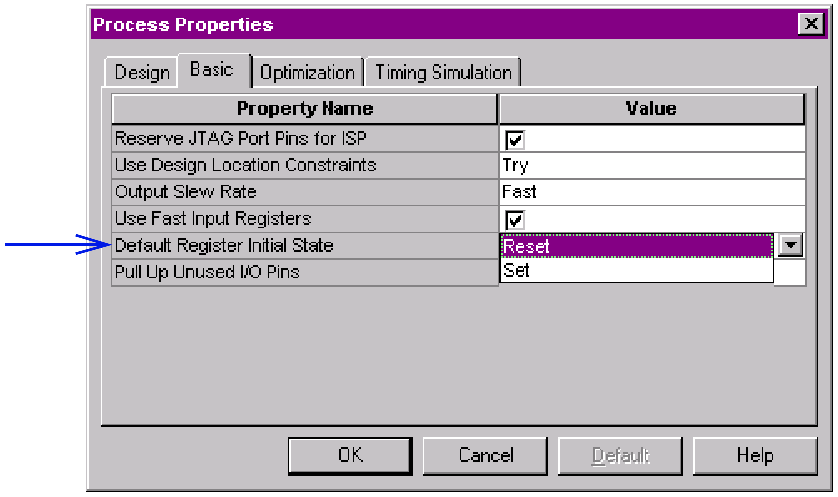
The UCF allows specification for an individual output slew rate and has priority over default settings in the Implement Design options. The slew rate can be selected as SLOW or FAST as the syntax and an example for the UCF is shown in Table 7. If a SLOW slew rate is specified for a specific signal, an additional delay of 4.0 ns is incurred.

Table 7: **Output Slew Rate**

Syntax	Examples
NET <signal_name> <slewrate>; Note: <slewrate> can be set to either FAST or SLOW.	NET clk SLOW; NET qout<0> FAST;

Initial State

The initial state of all registers is selectable for XPLA3 and can be specified in the **Implement Design Process Properties** window. XPLA3 can implement all registers in either a reset or preset initial state as shown in Figure 7.



Property Name	Value
Reserve JTAG Port Pins for ISP	✓
Use Design Location Constraints	Try
Output Slew Rate	Fast
Use Fast Input Registers	✓
Default Register Initial State	Reset
Pull Up Unused I/O Pins	Set

Figure 7: **Setting Register Initial State**

Selecting the initial state of an individual register is done in the UCF. The UCF allows the power-up initial state to be either SET or RESET as shown in Table 8.

Table 8: **Register Initial State**

Syntax	Examples

NET <signal_name> INIT = x;
Note: x can be set to either R (reset) or S (set).
	NET clk INIT = R;
	NET qout<0> INIT = S;

Input Pullup

XPLA3 pins used as inputs have internal pullup resistors available. The input pullup resistor value in XPLA3 is not guaranteed nor tested and can range from 20k to 200k Ohms. Applying an internal pullup resistor to an input pin can be done through the UCF. The syntax and an example are shown in Table 9.

Table 9: **Input Pullup**

Syntax	Examples

NET <signal_name> PULLUP;	NET clk PULLUP;
	NET qout<0> PULLUP;

Signal Optimization

In the design fitting process, the implementation of signals can be specified in the UCF. The fitter interprets signals and reduces logic to create an optimum fit in the device. The attributes that are available for optimizing signal equations in XPLA3 are described in Table 10.

Table 10: **Signal Optimization Options**

Command	Functionality
KEEP	Preserves internal nodes during design implementation. The fitter will keep this signal as specified by its equation. This option may increase the number of product terms and/or macrocells needed in the design.
COLLAPSE	Forces an internal node to collapse forward into the logic of another signal. This option may reduce the number of macrocells needed in the design.
NOREDUCE	Preserves redundant logic terms during design implementation. This option may create more internal nodes.

Table 11 shows the syntax and an example for signal optimization options in the UCF.

Table 11: **Signal Optimization**

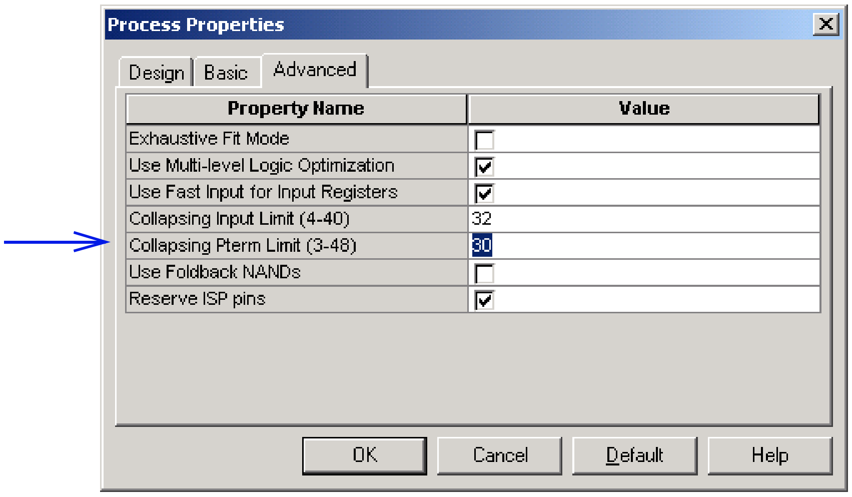
Syntax	Examples

NET <signal_name> <attribute>;
Note: <attribute> can be either KEEP, COLLAPSE, or NOREDUCE
	NET q_int KEEP;
	NET state_com COLLAPSE;
	NET q_com NOREDUCE;

Product Terms

The use of product terms can be adjusted for implementation of logic equations. The maximum number of product term for a logic equation can be specified for design implementation. There are a total of 48 available product terms in each XPLA3 function block. Some logic equations in a design may require a large number of product terms and could consume a large percentage of available product terms in a single function block. If it is desired to fit additional logic within the same function block, requiring unique product terms, the number of product terms available to a specific signal must be reduced.

The WebPACK **Implement Design Process Properties** shown in Figure 8 illustrates how to specify the number of product terms available to all logic equations in a design.



Property Name	Value
Exhaustive Fit Mode	☐
Use Multi-level Logic Optimization	✓
Use Fast Input for Input Registers	✓
Collapsing Input Limit (4-40)	32
Collapsing Pterm Limit (3-48)	30
Use Foldback NANDs	☐
Reserve ISP pins	✓

Figure 8: **GUI Setting for Maximum Product Terms**

When implementing a design, it may be desired to specify the maximum number of product terms available to a specific logic equation. Reducing the number of product terms available to one logic equation increases the amount of logic available to other logic terms in the function block.

Table 12 illustrates the UCF syntax and an example for specifing the number of product terms accessible to a specific logic equation.

Table 12: **Max Product Term Specification**

Syntax	Examples

NET <signal_name> MAXPT=<integer>;
Note: <integer> must be a positive integer less than or equal to 48 (product terms available in XPLA3 function block)
	NET qout MAXPT=8;
	NET cnt_out MAXPT=20;

Global Clock

Please refer to WP105: "CoolRunner XPLA3 Architecture Overview" for more information on the CoolRunner XPLA3 architecture found here: http://www.xilinx.com/products/coolpld/wp105.pdf. XPLA3 CPLDs have four global clock inputs. Each function block in the device has two of the four global clocks available at each macrocell. When a design utilizes a clock input in multiple function blocks, the clock is automatically implemented as a global clock and must be assigned to a global clock input pin. When it is desired to specify a clock as a global clock, the UCF syntax shown in Table 13 illustrates this constraint.

Table 13: **Global Clock Syntax**

Syntax	Examples

NET <signal_name> BUFG=CLK;	NET sys_clk BUFG=CLK;
	NET clk1 BUFG=CLK;

Additional Notes

The asterisk (*) can be used as a wildcard character in the NET or PIN signal name in the UCF. The asterisk character will apply the same attribute to more than one element in the design.

Listing all attributes	Using an asterisk (*)
NET qout<0> FAST; NET qout<1> FAST; NET qout<2> FAST; NET qout<3> FAST;	NET qout* FAST

Conclusion

All attributes specified in the WebPACK Project Navigator GUI, the UCF, or the design source code can be verified by reviewing the fitter and timing reports of the design. Utilizing a UCF allows designers to maintain design attributes over design iterations and throughout the design life cycle. A UCF is easy to use and controls signal properties for an entire design. For further assistance with a UCF or WebPACK Project Navigator software, please search for solutions at **Xilinx Online Technical Support**.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
11/09/00	1.0	Initial Xilinx release.
06/20/01	1.1	Updated for support of WebPACK release 4.1. This software release will not support the PIN specification. Instead, the NET specification must be implemented.
12/19/01	1.2	Updated for support with WebPACK 4.1i release. New XPLA3 attribute support includes BUFG and MAXPT. Updated to be CoolRunner XPLA3 specific.