

Xilinx/ Mentor Graphics PCB Guide



Xilinx is disclosing this Document and Intellectual Property (hereinafter “the Design”) to you for use in the development of designs to operate on, or interface with Xilinx FPGAs. Except as stated herein, none of the Design may be copied, reproduced, distributed, republished, downloaded, displayed, posted, or transmitted in any form or by any means including, but not limited to, electronic, mechanical, photocopying, recording, or otherwise, without the prior written consent of Xilinx. Any unauthorized use of the Design may violate copyright laws, trademark laws, the laws of privacy and publicity, and communications regulations and statutes.

Xilinx does not assume any liability arising out of the application or use of the Design; nor does Xilinx convey any license under its patents, copyrights, or any rights of others. You are responsible for obtaining any rights you may require for your use or implementation of the Design. Xilinx reserves the right to make changes, at any time, to the Design as deemed desirable in the sole discretion of Xilinx. Xilinx assumes no obligation to correct any errors contained herein or to advise you of any correction if such be made. Xilinx will not assume any liability for the accuracy or correctness of any engineering or technical support or assistance provided to you in connection with the Design.

THE DESIGN IS PROVIDED “AS IS” WITH ALL FAULTS, AND THE ENTIRE RISK AS TO ITS FUNCTION AND IMPLEMENTATION IS WITH YOU. YOU ACKNOWLEDGE AND AGREE THAT YOU HAVE NOT RELIED ON ANY ORAL OR WRITTEN INFORMATION OR ADVICE, WHETHER GIVEN BY XILINX, OR ITS AGENTS OR EMPLOYEES. XILINX MAKES NO OTHER WARRANTIES, WHETHER EXPRESS, IMPLIED, OR STATUTORY, REGARDING THE DESIGN, INCLUDING ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND NONINFRINGEMENT OF THIRD-PARTY RIGHTS.

IN NO EVENT WILL XILINX BE LIABLE FOR ANY CONSEQUENTIAL, INDIRECT, EXEMPLARY, SPECIAL, OR INCIDENTAL DAMAGES, INCLUDING ANY LOST DATA AND LOST PROFITS, ARISING FROM OR RELATING TO YOUR USE OF THE DESIGN, EVEN IF YOU HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THE TOTAL CUMULATIVE LIABILITY OF XILINX IN CONNECTION WITH YOUR USE OF THE DESIGN, WHETHER IN CONTRACT OR TORT OR OTHERWISE, WILL IN NO EVENT EXCEED THE AMOUNT OF FEES PAID BY YOU TO XILINX HEREUNDER FOR USE OF THE DESIGN. YOU ACKNOWLEDGE THAT THE FEES, IF ANY, REFLECT THE ALLOCATION OF RISK SET FORTH IN THIS AGREEMENT AND THAT XILINX WOULD NOT MAKE AVAILABLE THE DESIGN TO YOU WITHOUT THESE LIMITATIONS OF LIABILITY.

The Design is not designed or intended for use in the development of on-line control equipment in hazardous environments requiring fail-safe controls, such as in the operation of nuclear facilities, aircraft navigation or communications systems, air traffic control, life support, or weapons systems (“High-Risk Applications”). Xilinx specifically disclaims any express or implied warranties of fitness for such High-Risk Applications. You represent that use of the Design in such High-Risk Applications is fully at your risk.

Copyright © 1995-2007 Xilinx, Inc. All rights reserved. XILINX, the Xilinx logo, and other designated brands included herein are trademarks of Xilinx, Inc. PowerPC is a trademark of IBM, Inc. All other trademarks are the property of their respective owners.

About This Guide

This guide contains information for FPGA designers and Printed Circuit Board (PCB) engineers about processes and mechanisms available within ISE and various Mentor Graphics tools to efficiently implement an FPGA on a PCB. The first section of the guide covers the PCB and FPGA design flows highlighting steps in these flows where data is exchanged between these two software environments. Then for each identified step the guide details processes, files, and options available to perform the identified task.

This guide covers features from Mentor Graphics I/O Designer and DxDesigner with some reference to layout tools such as Expedition. For full details regarding these tools, please check Mentor Graphics documentation available at:
www.mentor.com/supportnet/documentation.

If you use multiple vendor software tools for your PCB design flow, such as Cadence OrCAD for schematic capture with Mentor Graphics PADs for PCB layout, refer to the specific documentation for those tools. For Cadence tools, please refer to the *"Xilinx/Cadence PCB Guide"* for additional information.

Additional Resources

To find additional documentation, see the Xilinx website at:

<http://www.xilinx.com/literature>.

To search the Answer Database of silicon, software, and IP questions and answers, or to create a technical support WebCase, see the Xilinx website at:

<http://www.xilinx.com/support>.

To find additional documentation regarding Mentor Graphics PCB design tools, see:

<http://www.mentor.com/supportnet/documentation>.

Table of Contents

Preface: About This Guide

Additional Resources	3
----------------------------	---

Xilinx/Mentor Graphics PCB Guide

Implementing a Xilinx FPGA on a Printed Circuit Board	7
Design Flow	7
Schematic Capture Tool in the FPGA Design Flow	9
PCB Layout Tool in the FPGA Design Flow	9
Mentor Graphics PCB Design Tools	9
Mentor Graphics PADS Series	9
Mentor Graphics Expedition Series	10
Mentor Graphics I/O Designer Flow	10
Multi-Vendor Flow	10
Common Tasks	10
Create an Initial FPGA Pinout	10
Necessary Information	11
Process	12
Generate an Initial FPGA I/O Constraint File (UCF)	13
UCF Creation with a Text Editor	13
UCF Creation with PACE or Floorplan Editor	14
UCF Creation with I/O Designer	14
UCF Creation with the PIN2UCF Utility	14
Create a Schematic Symbol (Schematic Symbol Shape and Content)	14
Necessary Information	15
Process	15
Create a Layout Symbol	16
Map Schematic Symbols to the Layout Symbol	16
Update ISE from Pinout Changes Done in the Schematic Tool	17
PADs or Expedition Series	17
I/O Designer	17
Update ISE with Pinout Changes Done in the Layout Tool	17
PADs or Expedition Series	17
I/O Designer	17
Update the PCB Database with Pinout Changes Done in ISE	18
PADs or Expedition Series	18
I/O Designer	18

Xilinx/Mentor Graphics PCB Guide

This guide contains the following sections:

- [“Implementing a Xilinx FPGA on a Printed Circuit Board”](#)
- [“Common Tasks”](#)

Implementing a Xilinx FPGA on a Printed Circuit Board

In recent years, the design of FPGAs and printed circuit boards (PCBs) have become increasingly parallelized as opposed to the traditional sequential model. This is mostly due to market pressure that demands a fast design cycle and rapid adaptability to specification changes. In the past the FPGA was typically designed before the board or was added to an already designed board to perform some glue logic function, voltage or protocol conversion. Often the same PCB engineers were doing both the FPGA and PCB designs. Today, with their increasing internal and I/O capabilities, FPGAs can take on more core features of an application which require longer development time and greater expertise and manpower. On the board side, tight form factor, signal integrity, and electromagnetic regulations require sharp skills and dedicated personnel. Therefore, FPGA and PCB are now two separate design teams working in different environments and often physically distant. Paradoxically, pressures in terms of time and adaptability to market requires many more interactions between these design environments so that functionality, performance and cost objective are delivered on time. In practice, this translates into back and forth data exchanges throughout the design process between design teams to update the board symbols and FPGA constraints.

Design Flow

[Figure 1, page 8](#), System Development Cycle, shows a typical flow in the PCB and FPGA development cycle (white boxes). It also highlights the steps that require communication between FPGA and PCB software tools (grey boxes). [“Common Tasks,” page 10](#) details the mechanisms and processes available to perform each of these data exchanges.

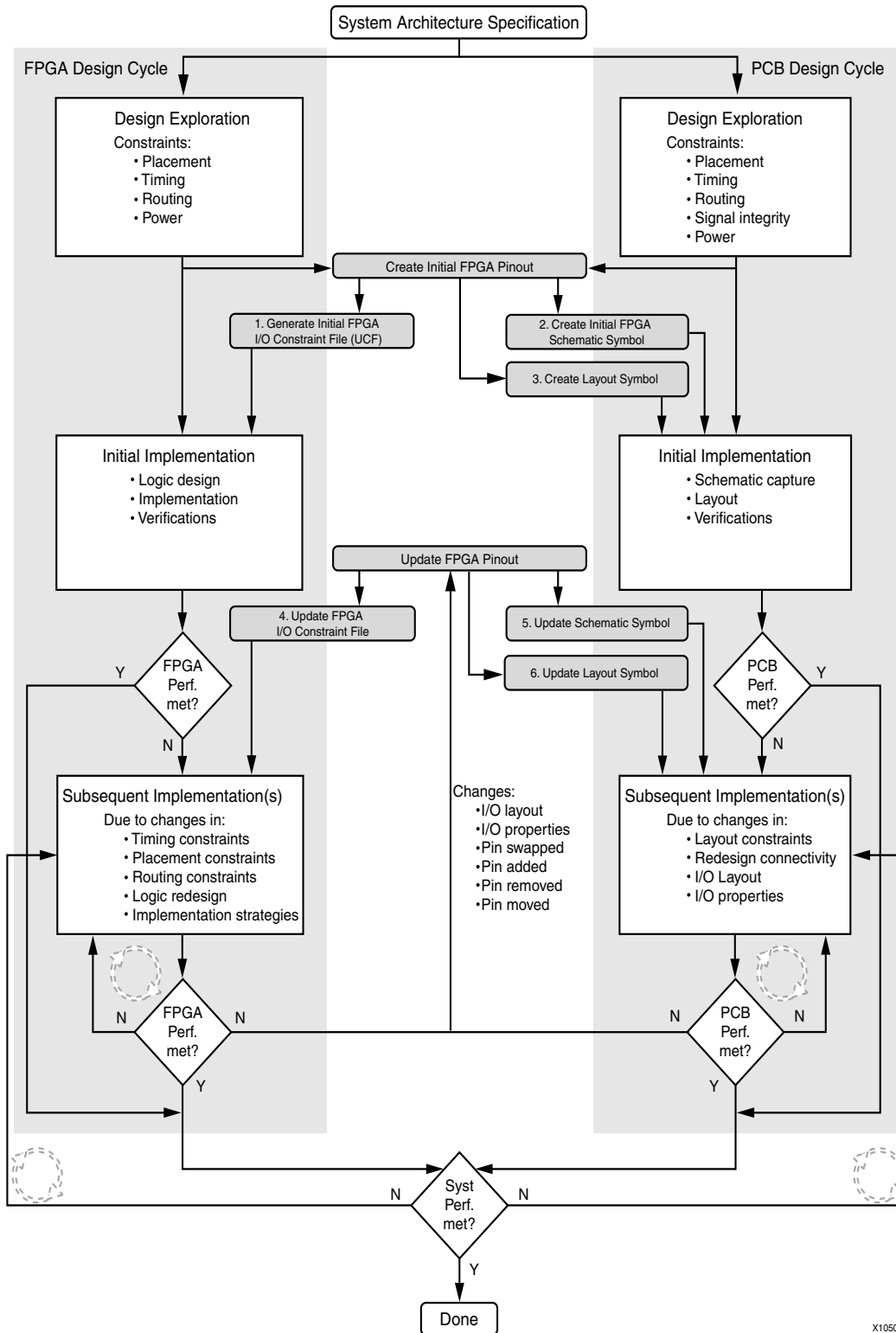


Figure 1: System Design Cycle with FPGA and PCB Databases Synchronization Steps Highlighted

PCB design requires two main tools; namely the schematic capture tool and the PCB layout tool. These tools are described in the following sections.

Schematic Capture Tool in the FPGA Design Flow

The schematic capture tool enables designers to create a graphical representation of connections between components on the PCB. This data helps anyone involved in the project to understand how components on this board are connected between themselves and with the outside world. The layout designer also uses this information to physically place and route all signals on the PCB.

Tips: Since an FPGA is a programmable component, its requirements on the PCB are unique to your application. Xilinx recommends that you add within the schematic all the specific components necessary for both the programming and the behavior of this device in your particular application.

- Add decoupling capacitors. Since FPGAs can be programmed to perform in a wide range of applications which translate into a wide range of decoupling needs, it is not practical for Xilinx to embed decoupling networks inside the device. The schematic engineer often adds all the decoupling network details on the schematic so as to let the PCB designer place these components in the vicinity of the FPGA package.
- Add other external components necessary to enable specific FPGA features. For instance the schematic designer needs to attach digitally controlled impedance (DCI) calibration resistors to VRN and VRP pins when I/Os on the device have the on-chip termination option enabled.
- Add debug, probe, and test points.
- Add pin swapping information. It is often useful at this point to define which pins can be swapped without violating FPGA banking, clocking, or SSO rules. This is crucial information for the PCB designer when trying to minimize wire crossover and congestion in placing and routing the FPGA on the board.

PCB Layout Tool in the FPGA Design Flow

The PCB layout tool reads the component and connectivity description in the schematic capture tool and physically places and route these components on the PCB. The output is a set of masks and geometries that allow manufacturing of the PCB.

Tips: In order to efficiently place and route a programmable device, the PCB designer needs the following information:

- Board physical dimensions. Dimension of the board, mandatory position of connectors, etc.
- Stackup dimensions. Number and orientation of signal layers, number and location of power and ground planes, board material, traces properties, etc.
- Components footprint. Exact dimensions of each component package.
- Components landing pattern. Shape of the junction area between the component and the board including manufacturing tolerances.
- Board environment properties. Available space around the PCB (air flow, obstacle, vibrations, cooling system, access to power and connectors, etc.)

Mentor Graphics PCB Design Tools

Below is a brief description of Mentor Graphics' tool chains and capabilities available for designing printed circuit boards. Please refer to the tool's documentation for further details.

Mentor Graphics PADS Series

This tool set is typically appropriate for low to medium complexity boards and single site design teams. This document refers to DxDesigner or Dx-PADS as the schematic capture

tool and PADS Layout as the layout tool. Depending on your exact software configuration, your Mentor documentation might refer to slightly different names, however all features and methodologies presented here are available to you.

Mentor Graphics Expedition Series

This tool set is typically appropriate for medium to high complexity designs and for single to many site design teams. This document refers to DxDesigner or Dx-Expedition as the schematic capture tool and Expedition PCB as the layout tool. Depending on your exact software configuration, your Mentor documentation might refer to slightly different names, however all features and methodologies presented here are available to you.

Mentor Graphics I/O Designer Flow

This flow is meant to complement either of the previous flows. I/O Designer adds many features that assist with the creation and maintenance of FPGA symbols. On the FPGA side, I/O Designer understands all Xilinx device families and their I/O properties and restrictions. It can therefore guide you through the pin assignment process along with the definition of pin swappability. On the PCB side, I/O Designer offers wizards and automation to generate and fracture symbols and automatically connect them to the hierarchical schematics while maintaining those connections when pin assignments change. I/O Designer integrates with Xilinx ISE and keeps the constraint file (UCF) up to date with board schematic symbols. Therefore, consistency can be maintained when changes are made either in the FPGA or PCB domain in order to meet overall system performance.

Multi-Vendor Flow

You can use multiple vendor software tools for your PCB design. For instance Cadence Concept-HDL or OrCAD can be used for schematic capture with Mentor Graphics PADs or Expedition series for PCB layout. Cadence users, please refer to the *“Xilinx/Cadence PCB Guide”* for information about Cadence tools. When using other PCB software packages please refer to your vendor's documentation.

Common Tasks

The following section covers the process, available software features, and file manipulations needed to accomplish each task in the PCB or FPGA design flow related to the FPGA pinout. Each of these tasks is illustrated in the flow chart [Figure 1, page 8](#). Tasks associated exclusively with either PCB or FPGA design flow are represented as white boxes. Tasks common to both design flows are represented as grey shaded boxes

Create an Initial FPGA Pinout

After taking into account the system specification describing the different parts and connectors on the board along with communication channels linking them together, the next step for the designer is to infer a device and package that can accommodate these communication channels then expand, classify, and assign each signal to a particular pin on the chosen FPGA package. This task requires FPGA architecture knowledge allowing PCB designers to find an optimal pinout for the PCB design. Therefore, this task is typically done by an FPGA engineer. With the help of tools such as I/O Designer this may now also be driven from the PCB software environment allowing PCB designers to find an optimal pinout for the PCB design.

Necessary Information

I/O placement requirements may come from a variety of sources. To save time Xilinx recommends that you draw the list of I/Os and learn about the FPGA architecture before starting I/O placement.

Tip: FPGA requirements can be found in the device data sheet and user guides.

- System requirements:
 - ◆ Identify properties and number of each I/O standard required (including direction, input and output voltages, drive strength, slew rate, data rate, etc.).
 - ◆ Identify differential pairs.
 - ◆ Identify global/regional clock signal with their associated data signals.
 - ◆ Identify Multi Gigabit Serial Transceivers.
 - ◆ List I/O location constraints imposed by predefined IP Blocks (third party IP, IP generated by CORE Generator, Architecture Wizard, the Memory Interface Generator, or the Embedded Development Kit (EDK)).
- FPGA Requirements (refer to the device user guides for more information):
 - ◆ Acquire device knowledge including:
 - I/O compatibility or IO banking requirements.
 - Device package properties such as I/O count for the entire device and per bank and clock region, I/O data rate and signaling capabilities such as single ended/differential or single/dual data rate support, etc.
 - I/O access to internal resources. Resources such as clock buffer, RAM, serializer/deserialiser, etc.
 - Device clocking capabilities such as internal clock management resources, I/O with direct access to clock networks, etc.
 - Device Simultaneous Switching Output specifications.
 - Package trace delays.
 - ◆ Reserve and Prohibit usage of special purpose pins:
 - Prohibit package pins. Because of die or package migration or future design growth constraints.
 - Reserve configuration pins.
 - Reserve JTAG pins.
 - Reserve DCI pins.
- PCB Requirements:
 - ◆ Formulate package escape strategies. Determine board number and direction of layers, pin spacing, etc.
 - ◆ Signal integrity. Estimate amplitude and timing margins for each signal type.
 - ◆ Air flow. Ensure work area has sufficient airflow.
 - ◆ Placement and orientation of neighboring parts which could constrain the FPGA placement or access of PCB signals to the FPGA.
 - ◆ Connectivity to other devices. Other devices may impact the optimal FPGA I/O design.

Process

Depending on the your preferences and company policies, there are different mechanisms you can use to assign pins on an FPGA. The following three methods are the most common.

Create a Pinout in a Spreadsheet Environment

To create a pinout in a spreadsheet environment, create two spreadsheets, the first with your design I/O requirements (signal name, I/O standards, direction, etc.), and a second with properties for each pin in the package (pin number, I/O bank number, pin name, etc.). Then going down the list of your design signals, filter out and sort package pins of the second spreadsheet to determine compatible device I/Os. Finally, go back to your original design I/O spreadsheet and assign pin numbers (or I/O bank numbers) to your signal names. Once this is done, you will export this pinout to the schematic and FPGA tools as detailed in the next paragraphs.

This method is most often used by advanced users with extensive knowledge of the FPGA's capabilities. Since there is no design rules checking (DRC) done by any tool during this process, the resulting I/O assignment could fail during FPGA or board implementation.

Tip: You can easily create the package property spreadsheet with one of the following:

- Partgen utility. For example:

```
partgen -v xc5v1x30ff676
```

- Download the package file for your target:
 - ◆ Virtex families: Refer to Answer Record 20578 at: www.xilinx.com/xlnx/xil_ans_display.jsp?iLanguageID=1&iCountryID=1&getPagePath=20578
 - ◆ Spartan-3 family: Refer to Answer Record 21035 at: www.xilinx.com/xlnx/xil_ans_display.jsp?iLanguageID=1&iCountryID=1&getPagePath=21035

Create a Pinout in ISE (PACE or Floorplan Editor)

PACE and Floorplan Editor are similar graphical tools that assist with the floorplanning and pin assignment processes. The Package View tab and Design Objects tab of these tools allow you to edit signal properties and to drag and drop your signals to pin number or pin number to signal name. You can also use these tools to assign signals to sets of pins such as I/O banks and leave the task of assigning individual signals to individual pin numbers within a specified range to the implementation tool. The Design Objects tab also presents signals properties such as I/O standards, direction, and voltages. The Package View tab uses color schemes to highlight I/Os belonging to the same I/O banks, and also displays the location of differential pairs, global clock pins, and regional and local clocks with their associated data pins. You may also prohibit multiple pins in a single action such as DCI, configuration pins, or package pins unavailable in another die/package combinations. The Design Objects and Package View tab combination makes it easier to identify package pins that support your signal properties. To make an assignment, you can type in your assignments in the Design Objects tab or drag and drop a signal to a package pin in the Package View (or vice-versa).

Finally, the tool performs two types of checks. First it performs a set of checks while you are making assignments preventing the most common violations. You can also invoke a more complete set of verifications by clicking **Run Design Rule Checks** from the **Floorplan** drop-down menu. This generates a report with further details on violations.

Tip: In PACE or Floorplan Editor you do not need to completely define the I/O assignment within the tool itself. You may read in partial assignments in the form of spreadsheets (CSV) or Xilinx UCF files then assign pins that have not yet been assigned or modify any assignment. You may also use the import function simply to run DRC and verify an assignment done outside of Xilinx software.

Refer to the PACE or Floorplan Editor online help for more information on working with these tools.

Create a Pinout in I/O Designer

Using I/O Designer to create an FPGA pinout is very similar to using PACE or the Floorplan Editor. In addition to the features described previously, I/O Designer adds a table view of the package along with advanced filtering options which help in isolating signals or pins of interest. In the physical device view, I/O Designer also shows the PCB floorplan and nets between devices allowing optimization of the connectivity during the I/O assignment process. With regards to the legality of pin assignment done within I/O Designer, Xilinx and Mentor Graphics work closely together to continuously ensure ISE and I/O Designer have identical sets of DRCs so that an I/O assignment done within I/O Designer will be valid in ISE. I/O Designer shows the PCB floorplan and guides or ratsnests in the physical device view allowing optimization of the connectivity during the I/O Design process.

Here too, users do not need to completely define the I/O assignment within I/O Designer. You may read in partial assignments in the form of spreadsheets (CSV or FPGAXchange) or Xilinx files (PAD, GYD or UCF) and then make adjustments or perform checks within I/O Designer.

Tip: In I/O Designer remember to experiment with **Ctrl**, **Alt**, and **Shift** keys to assign multiple signals to multiple package pins at once.

Generate an Initial FPGA I/O Constraint File (UCF)

In ISE, I/O constraints can be entered in a unique user constraint file (UCF) attached to the design project. They can also be attached to the HDL (Verilog or VHDL) source code, the synthesis constraint file (XCF) or embedded in the logic netlist (NGC, EDF, or EDN files, and associated NCF files). The problem with entering I/O properties and location constraints in multiple files is that maintenance, portability, and updates to the design become much more complex. Xilinx recommends that you specify the maximum number of I/O related constraint within a single UCF file.

UCF Creation with a Text Editor

You can create a UCF file by simply typing the constraints into a text editor. When creating a UCF file in this way, please refer to the *“Constraints Guide”* for the syntax of all I/O related constraints. This method is most often used by companies that have developed their own scripts that read in a spreadsheet and convert the data into UCF syntax.

If you already have your I/O constraints defined in a spreadsheet format, you can use the import function in PACE or Floorplan Editor. In this case, the tool parses your spreadsheet and converts recognized data into UCF syntax. Data is recognized for all column headers that match the following:

- Pin Number
- Pin Name
- Signal Name
- Direction

- IO Standard
- IO Bank Number
- Drive (mA)
- Slew Rate
- Termination
- IOB Delay
- Diff Type
- Diff Pair
- Swap Group

Refer to the PACE and Floorplan Editor online help for more information.

UCF Creation with PACE or Floorplan Editor

When using PACE or Floorplan Editor, the assignments performed are saved by default in the UCF file defined during the initial project creation.

UCF Creation with I/O Designer

Within I/O Designer, once you have created a design database, imported or created a pin assignment, and the results are satisfactory, you can export this assignment as a Xilinx user constraint file (UCF) to run the FPGA place and route tool with this I/O placement. To do so, select **PR Constraints File** from the **Generate** drop-down menu. The file is saved at the location specified during the initial project creation.

UCF Creation with the PIN2UCF Utility

With the other methods for creating a pinout described in this section you can either create a complete pin assignment or create a partial one (assign a signal names to sets of pin numbers) and let the back-end place and route tool perform the actual assignment within this specified set. In the second case, you can assign a signal or a set of signals to a pin, a set of pins, a bank, or a set of banks and thus give the implementation tool the task of assigning an exact package pin number to each individual user I/O.

If after the implementation you are satisfied with this pinout and want to preserve it for future implementation runs then you can do one of the following:

- In Project Navigator go to the **Process** window and expand the **Implement Design** process. Next, expand the **Place & Route** process, and double-click **Back-Annotate Pin Locations**. A UCF file is created and has all your I/O signals locked to a specific package pin number.
- Use the PIN2UCF utility to lock a particular pinout for the next implementation iteration. To use the PIN2UCF utility, type the following at the command line.

```
pin2ucf ncd_file_name | gyd_file_name -o ucf_file_name
```

Create a Schematic Symbol (Schematic Symbol Shape and Content)

An FPGA schematic symbol is used to describe the electrical connectivity between each device and its environment (other parts, connectors, etc.). Unlike most other components, FPGA symbols are not likely to be available in a predefined library. One purpose of a library is to allow reuse of its elements across different applications. FPGAs, by definition are programmable and application specific so no two designs will have the same

connectivity (signal names and pinout) with the outside world. Therefore, few symbol properties can be reused from one project to another.

Necessary Information

A schematic symbol is a graphic to which global properties are attached. Each symbol lists and locates I/Os on the graphic and may also tag I/Os with additional properties. Typically FPGAs have more I/Os than can be represented on a single schematic sheet, therefore, FPGA symbols are often split into a multiple fractures and hierarchy levels to simplify readability. Each company and sometimes each engineer has their own process and opinion as to what an FPGA schematic symbol should look like or contain. Below is the minimum set of data required for an FPGA schematic signal plus some additional information which could make the FPGA schematic symbols more useful.

- Graphical symbol — Allows the component can be placed on the schematic. Choose a shape that allows placement of a fair number of I/Os. Some engineers use a different shape depending on the type of interface.
- Device name — Component ASCII name that makes it easy for someone reading the schematic to know what this component is.
- Reference designator — Unique and short identifier for each component on the schematic.
- I/O name (or pin name) — A separate name for each I/O on the symbol. Xilinx recommends using the same name as in the top level HDL description. Since this is the name FPGA designers will be familiar with, this makes for easier communications between schematic and FPGA designers. In addition each I/O can be tagged with additional visible or non-visible properties. Therefore, Xilinx recommends adding data sheet name, pin name, and I/O direction.
- Non-user I/Os — Ensure that all pins available on the package have an entry in the schematic symbol. Some of them may be visible such as unused or logistical I/O (DCI reference). Others may be hidden as they are not of interest for describing the board functionality, such as power, ground, no connect, or reserved I/Os. Having all I/Os present on the schematic symbol will be appreciated because it helps with mapping the schematic symbol to the layout symbol. It will also facilitate the maintenance of symbols when pin swaps occur.
- Pin Number — Locates the I/O on the package ball array.

Process

Depending on the engineers' preferences and company policies, there are several mechanisms for creating schematic symbols for FPGAs. Most engineers use one of the three methods described below. The easiest way to create and maintain schematic symbols is to use I/O Designer.

Schematic Symbol Creation in a Text Editor

To create a schematic symbol in a text editor please refer to Dx-PADS or DX-Expedition documentation on how to set the location, shape, and all other properties of the symbol.

This method is most often used by companies that have developed their own scripts that read in a spreadsheet, HDL, or Xilinx PAD files and automatically convert the data onto a symbol.

Create a Schematic Symbol with Dx-PADS or Dx-Expedition

Within DxDesigner use the Symbol Wizard guided process or use the DxBoardLink utility. Using DxBoardLink you can import signal names, pin number, and other properties directly from the Xilinx PAD file for FGPA designs or GYD files for CPLDs. In ISE, the PAD file is generated at the end of the implementation step. On a placed database, you can also generate the PAD file using the ReportGen command line utility.

Tip: To create a PAD file with ReportGen, type the following:

```
reportgen ncd_file_name -pad
```

Create a Schematic Symbol with I/O Designer

Use I/O Designer to create an FPGA symbol from a multitude of sources. You can also create the symbol starting from an empty database. Since entering details for each I/O on a large package can be quite cumbersome and error prone, I/O Designer lets you import your signal names and already defined I/O properties from the following:

- ISE files (UCF, PAD, GYD)
- Spreadsheet files (FPGAXchange, CSV)
- HDL files (Verilog, VHDL).

Not all these files are available at any given time in the design flow. Use the Database Wizard and then Symbol Wizard to guide you through the process of reading in available I/O data, then add and edit symbol properties to finally configure then fracture the symbol.

In short, the Symbol Wizard guides you through the creation process by successively prompting you to enter the following:

- Basic information — Symbol, device and package names, etc.
- Symbol fracturing information — Fracturing scheme, max number of pins per page, etc.
- How to represent special pins — Config, JTAG, DCI, Power and Ground, etc.
- Symbol appearance — Shape, pin position, port dimension and spacing, symbol port label and position.

Create a Layout Symbol

The layout symbol contains the device physical dimensions such that copper traces can be accurately routed to and from the FPGA pins or balls. PADS and Expedition flows have dialog boxes that allow you to enter package physical dimensions such as the balls location and dimension. The layout symbols are not design specific and can therefore be stored in a predefined library and shared among many FPGA designs with the same package.

Tips:

- Use the Mechanical Drawing section in the Packaging and Pinout Specification of the specific device user guide to capture this information.
- Go to www.xilinx.com and click **Documentation** to find the user guide for your device.

Map Schematic Symbols to the Layout Symbol

This is the process of mapping the pin numbers on the schematic symbols to the pin numbers on the layout symbol. Whenever possible, also enter pin swapability information during this step.

In the Dx-PADS flow, enter this data via a set of layout symbol attributes. In Dx-Expedition enter it via the parts database and editor utility. Please refer to the tool's documentation for details on how to establish this mapping.

Note: When using I/O Designer, the mapping data including the necessary swapping information that you define within I/O Designer is automatically generated and maintained.

Update ISE from Pinout Changes Done in the Schematic Tool

There are several occasions throughout the design process where pinout changes may happen in the schematic tool and must then be propagated to the FPGA user constraint file (UCF). For instance, the board design may have started before the FPGA internal logic. Therefore, pins may have been added, removed, renamed, or relocated. The schematic engineer may also discover improperly assigned pins or that the system specifications have changed requiring more, fewer, or different I/O properties.

The ISE user constraint file (UCF) must be kept in sync with the board I/O to avoid system malfunction.

PADs or Expedition Series

In either PADS or Expedition, there is no direct link to update the ISE user constraint file. Typically, the schematic engineer compiles a list of pinout changes in the form of a spreadsheet or meets with the FPGA engineer to ensure those changes are propagated and possible within the FPGA environment.

I/O Designer

Whenever I/O Designer detects a pinout modification such that the schematic and the FPGA user constraint file (UCF or CSV) are out of sync, a red flashing light appears in the bottom right-hand corner of the application. Click on it and select which change to propagate in which database to re-synchronize the design environments. This simple process removes many manual edits and email or oral discussions between member of separate teams and avoids cases where information gets lost or distorted.

Update ISE with Pinout Changes Done in the Layout Tool

At this stage in the design flow there are many ways to take advantage of FPGA I/O programmability by modifying its pinout to optimize the PCB. For instance, there may be a need to reduce wire cross over to be able to complete the PCB routing without requiring an additional routing layer. Another common practice is to move or swap pins in order to match or reduce trace length or reduce the number of vias or layer changes due to signal integrity or board timing concerns. This results in a PCB database that is out of sync with both the schematic and FPGA databases. To synchronize these environments, first propagate the pinout changes to the schematic database then to the FPGA database as previously described in ["Update ISE from Pinout Changes Done in the Schematic Tool."](#)

PADs or Expedition Series

This is done in two steps. First propagate pin swaps from the layout tool to the schematics and symbols. Refer to Mentor Graphics documentation for instructions. The second step is to update ISE by synchronizing the UCF file with the schematic symbol data base as previously described in ["Update ISE from Pinout Changes Done in the Schematic Tool."](#)

I/O Designer

When I/O Designer detects a modification in the layout tool database that affects an FPGA pinout, a red flashing light appears in the bottom right-hand corner of the application. The process to synchronize the databases is similar to the previous section in that you simply click on the flashing light and select which change to propagate to which database. Click **Finish** to launch the synchronization of the different design environments. This automation removes many manual edits, verification, and email or oral discussions between members of separate teams and prevents cases where information gets lost or distorted, saving time in developing and debugging the system.

Update the PCB Database with Pinout Changes Done in ISE

Whether the I/O layout change is due to a timing constraint change, a new piece of logic being added, or a change to existing logic, pinout changes at the FPGA level happen throughout the design cycle. Whenever a pinout change occurs, it is important to propagate this change to the PCB schematic and layout environments to ensure they are not designing with an obsolete I/O assignment and that the board constraints have not been violated. First, synchronize the schematic database with the new FPGA I/O layout. Next, synchronize the layout database to the schematic database.

PADS or Expedition Series

Within either the PADS or Expedition design environment, there are no direct processes to update an FPGA schematic symbol. Typically the FPGA engineer compiles a list of pinout changes made in the form of a spreadsheet or meets with the schematic engineer. Then the schematic engineer regenerates the symbol (using DxBoardLink) or manually enters the changes.

Tips: Whenever a schematic symbol is regenerated there is a risk of this symbol losing connectivity with other parts on the schematic. Here are several tips to minimize this possibility.

- Ensure that the pin locations stay the same on the symbol by using the same settings as much as possible.
- Use find and replace features to reconnect the symbols properly.
- Avoid direct connections between schematic nets and FPGA symbols.

To propagate pin swaps or more generally I/O layout or property changes from the schematic editor to the layout tool, please refer to the Mentor Graphics documentation for your layout tool.

I/O Designer

Whenever I/O Designer detects a modification of the FPGA constraint file, a flashing red light appears at the bottom right-hand corner of the application. The process to synchronize the databases becomes similar to the methodology detailed in previous paragraphs in that you simply click on the flashing light and select which change to propagate to which database and then click **Finish** to launch the database synchronization. This automation removes many manual edits, verification, and email or oral discussions between members of separate teams. Another great benefit is that the maximum of connectivity in the schematic is preserved. Time consuming, manual, and error prone steps are saved with this flow.