

Successfully Designing FPGA-Based Systems

By Nagesh Gupta, President and CEO, Taray Inc.

Increases in field-programmable gate array (FPGA) capabilities, combined with growing system complexity, have created many FPGA-based system design challenges. One key challenge is choosing the right FPGA for the design needs, and maximizing the use of FPGA resources. Cadence offers recommendations for power-supply connections, pin selections and assignments, and other tips and methodologies to help customers design high-quality FPGA-based systems.

Contents

- Designing High-Quality FPGA-Based Systems2
- Power Supplies/Power Distribution Systems3
- Clocks4
- Source Synchronous Busses5
- High-Speed Serial Busses6
- FPGA Pin Assignment to Ensure Routability6
- Achieving High-Quality FPGA-Based Systems.....6

FPGAs have come a long way since they were introduced in the 80s. The initial FPGAs were primarily used to create “glue-logic” (logic that performs certain essential functions that are not done by other components on the board). Today, FPGAs have the density and functionality necessary to implement a system on chip (SOC) (see figure 1).

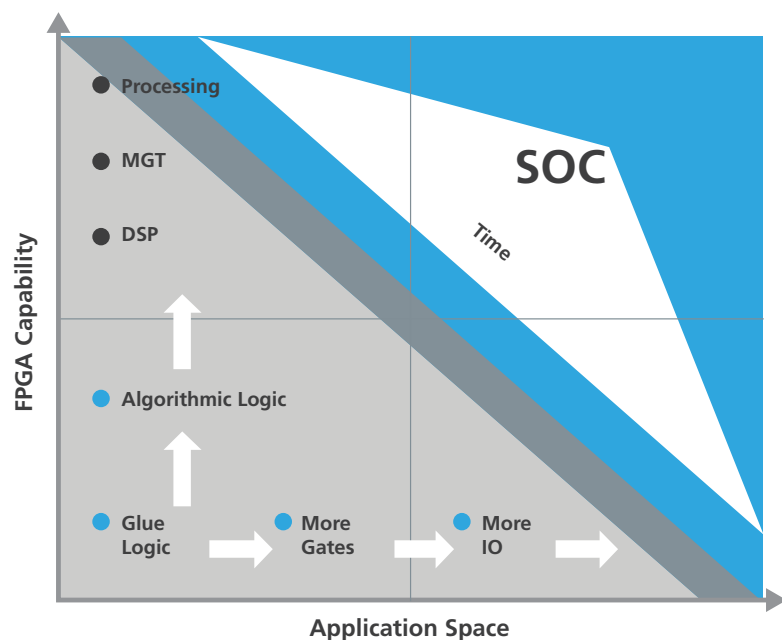


Figure 1: FPGA Growth and Usage Trends

This increase in FPGA capability, combined with the fact that the system has become a lot more complex, has introduced several FPGA-based system design challenges. Greater FPGA capability and enhanced configurability also implies increased pin-counts for such FPGAs. The challenges of designing FPGA-based systems are twofold. The first revolves around the FPGA itself: i.e., choosing the right FPGA for the design needs, and maximizing the use of FPGA resources. The second dimension has to do with FPGA-PCB co-design, which involves ensuring that pin-assignment is optimized for the PCB. This paper focuses on the first challenge—designing high-quality FPGA-based systems. A separate white paper will focus on the FPGA-PCB co-design aspects.

Designing High-Quality FPGA-Based Systems

When the goal is first-time success, high quality, and minimal debug effort, design teams must consider a laundry list of items—a list that is especially long for FPGA-based systems. Today, this means a great deal of busy work and attention to minute details.

The problems inherent in FPGA-based system designs can be classified into the following categories:

1. Functional – issues that cause the design to not work functionally. For example, if a clock to the FPGA is not coming in through a clock pin specified by the FPGA vendor, the clock cannot be distributed with a reasonable skew, and hence the design will fail to function.
2. Electrical – electrical issues that again cause the board to not work. If a 3.3 V LVTTTL signal is connected to an FPGA bank that has a 1.8 V voltage rail, the signal will not function electrically, and so the design will fail.
3. Marginal – issues that allow the board to work most of the time (but cannot be guaranteed to work all of the time). For example, if a clock is connected to a non-clock pin in the FPGA, it cannot be correctly distributed within the FPGA. When this happens, the design may not work at some frequencies, though it may work at certain other frequencies.

Some of the electrical and functional issues require a re-spin of the board. Some of these may be rectified with ‘green wires’ (or wires of your favorite color), but may make the board less robust. Such fixes typically require a re-spin before the board is production worthy. Marginal issues are not only harder to find, but are harder to debug and fix as well. More importantly, the marginal issues are the ones that could show up at a customer site—which makes it critical to put more emphasis on avoiding these.

Power Supplies/Power Distribution Systems

FPGAs have a high number of power supplies. For example, a simple design with a PCI Express interface and a DDR2 DIMM will require the power supplies shown in Table 1.

Voltage rail value	Need
1.2V	Serial I/O power
0.9V	Reference voltage for DDR2
3.3V	LVTTTL control signals for PCIe interface
1.8V	I/O Voltage for FPGA banks connected to DDR2
1.0V	FPGA Internal voltage
GND	Ground

Table 1: Power supplies for a simple design

Every bank of an FPGA can require two or more power supplies. Moreover, the logical connections to the bank will determine the voltage and current requirements for the power rail voltage and current requirement for that bank. This makes it important to pay attention to the voltage connections every time there is even a seemingly “minor” change in the pins connected to that bank.

As an example, if a DDR2 SDRAM is connected to a particular bank, a 1.8V and 0.9V (reference voltage) power supply is required for that bank. On the other hand, if an LVDS bus is connected to the bank, a 2.5V power supply is required to be connected instead of 1.8V. Looking at a DDR2 DIMM. While most pins in the DIMM use 1.8V SSTL, there are some I2C signals that could be of any other standard.

There are various decoupling recommendations for each of these different power supplies. These recommendations are either provided by the FPGA vendors or they may be determined based on a designer’s electrical simulations.

To summarize, the power supply connections require the following considerations:

1. Create a power budget for each power supply voltage in the system. This is to ensure that the power supplies can adequately supply the current required by the FPGA and the interface.
2. Review to ensure the power rails are of the right value. While this may seem an obvious statement, the reason for this is not obvious, because last-minute logic changes and pin optimizations could have changed the power rail requirements.
3. Add decoupling caps recommended by the FPGA vendors.
4. Ensure that power supply sensors are located close to the FPGA power pins. This ensures that the power supply can drive the dynamic current required by the FPGA.
5. Follow the recommendations of the FPGA vendor and the power supply vendors.

Clocks

FPGAs have different types of clock pins. For example, they have “global” clock pins, which are used to distribute the clock throughout the FPGA. Other clock pins are used to distribute clocks within a certain region of the FPGA. In Figure 2, a global clock is shown highlighted in red in the die view and in blue in the package view. In the same figure, the global clocks have been marked in bold font.

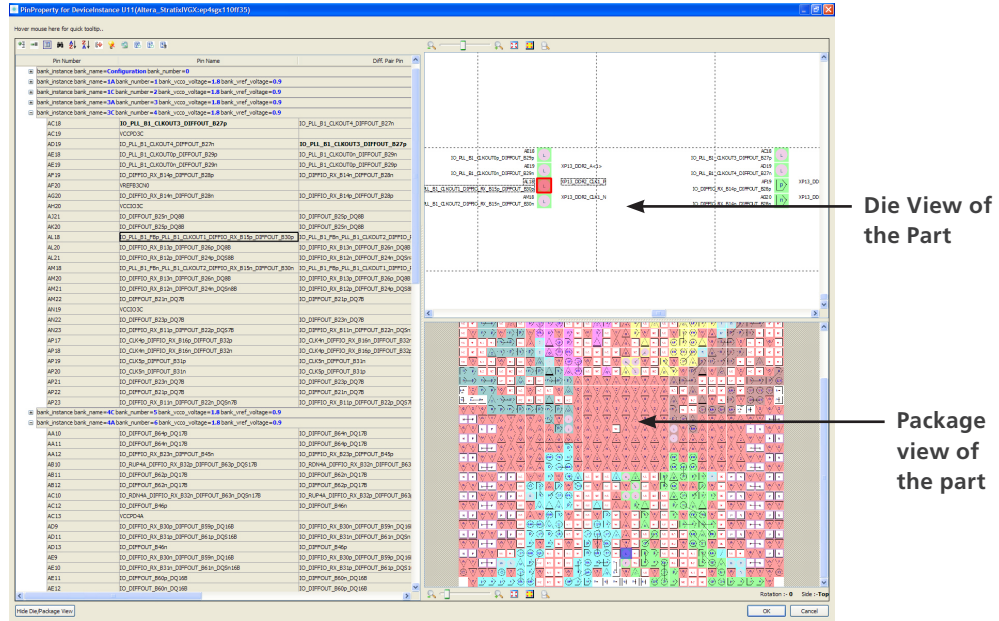


Figure 2: Illustration of global and regional clock pins in a Stratix IV FPGA

Figure 3 illustrates the same concept for a Xilinx Virtex-5 FPGA. Whether using an Altera device or a Xilinx device (or devices from other vendors), it is critical to pay close attention to the use of clock pins. As such, it is important that the tools used to perform the I/O assignment process understand the device-specific architectural requirements related to clocks.

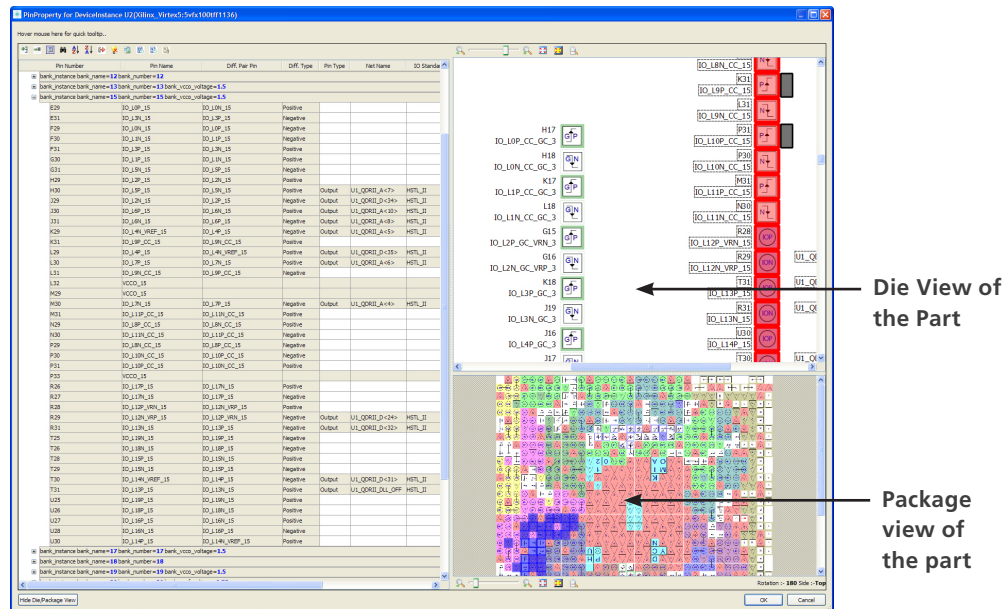


Figure 3: Illustration of global and regional clock pins in a Virtex-5 FPGA

Incorrect selection of clock pins in the FPGA can cause a design to not work functionally. What’s worse is the fact that the tools may not always warn the designer of such problems. Additionally, the design will also simulate correctly, and so the verification test benches may not uncover issues with the pin selections related to clocks.

Selecting an incorrect pin for a clock can create a marginal design. A marginal design will work for some conditions, while it may not work for other conditions.

Selecting the clock pins for a design requires the following considerations:

1. The frequency of the clock – The frequency will determine if the design should use a single-ended clock or differential clock. Also, FPGA vendors will specify clock resources depending on the required minimum and maximum frequency.
2. Local versus global – Determine if the clock is going to be used throughout the FPGA. If so, it will be necessary to use a global clock. If the clock is being used for local data capture, the designer may want to consider using one of the regional or local clock pins.

Source Synchronous Busses

Source synchronous busses are related to clocks (the previous section). A source synchronous bus contains clocks and an associated set of data signals. Most FPGAs require certain groups of pins to be used for source synchronous busses. Again, in most FPGA tools this is an entirely manual process where the designer is responsible for ensuring proper pin selection. Only by selecting the correct FPGA pins is it possible to ensure the clocks will reach the associated data signals. Incorrect pin allocation will cause the design to functionally fail. Figure 4 shows an illustration of different clock regions in a Virtex-5 FPGA with different colors in die and package.

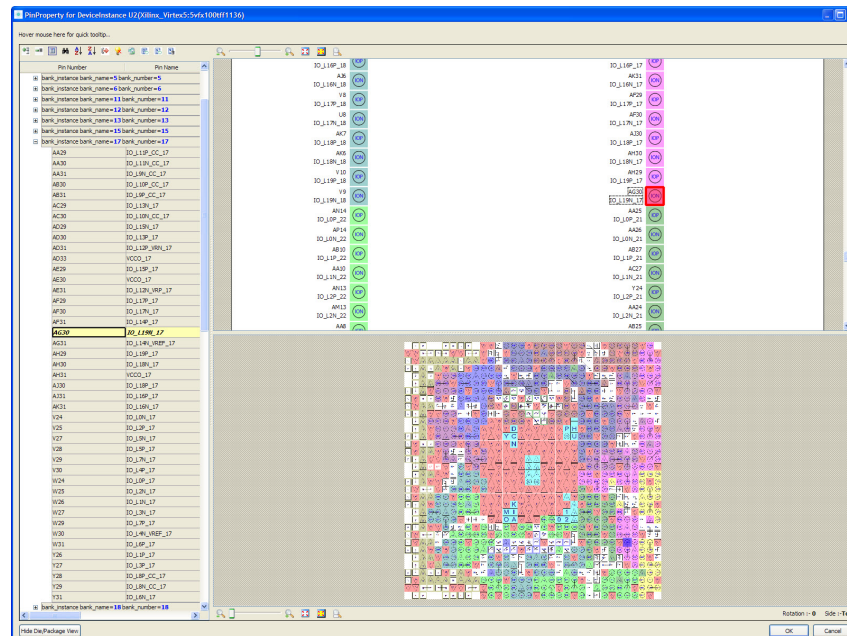


Figure 4: Illustrating different clock regions with different colors

Simulations and verification will not be able to determine a problem with pin selections. An example of a problem that will not be reported is when one of the data pins is outside the source synchronous group, and thus the associated clock may not be used for capturing the data pin. The FPGA tools report problems associated with clock/data groups during place and route of the corresponding RTL.

High-Speed Serial Busses

Today's FPGAs provide pins with high-speed transceivers for designing interfaces with speeds to 10Gbps. Designing with high-speed serial I/O requires carefully following the guidelines related to board layout, power supply designs, terminations, etc. Some of the FPGAs require isolating the transceiver power supply pins from any of the regular or single-ended I/Os. Not following these guidelines may cause marginal designs in most cases, or a design that fails in the lab.

FPGA Pin Assignment to Ensure Routability

With large-pin-count devices, the time required to do initial pin assignment is increasing. Often when the FPGA is integrated with the PCB design flow, PCB designers need FPGA pins to be swapped to reduce the time to route on the board as well as to reduce the number of layers required to fanout FPGA pins. This problem gets worse when pin counts on FPGAs rise above 500 to 750, as well as when multiple FPGAs are on the board.

PCB designers are concerned about improving routability and sometimes propose pin swapping without the knowledge of what is possible and what is not. Several requests from the PCB designer will not be possible to achieve for various reasons. Additionally, such requests typically come too late in the design cycle, putting pressure on both the PCB designer as well as on the FPGA designer.

To reduce design iterations at the tail end of the design cycle, it is important to consider reviewing relative placement of components within the FPGA system while doing pin assignment. This approach will reduce routing challenges at the tail end of the design while reducing iterations to swap FPGA pins.

Achieving High-Quality FPGA-Based Systems

The programmable nature of FPGAs makes them very different than ASICs/ICs. FPGAs also have a very high number of I/O pins—certainly higher than the typical ICs on the board. Higher numbers of I/Os make it more complex to design the PC board, and require special considerations to achieve high quality.

Let us try to define high-quality boards:

1. The board works consistently with a 20% margin in frequency. The margin is required both below and above the operating frequency.
2. The board works when tested with key parts from different process corners.
3. It has a 5% or 10% margin on voltage and temperature.
4. The board-level constraints are developed for high-speed signals using a solution space exploration approach that enables a constraint-driven PCB layout.

Here are some ways to achieve high-quality FPGA-based systems:

1. Terminate the high-speed signals based on FPGA vendor recommendations.
2. Follow the vendor guidelines for high-speed serial I/O designs.
3. Follow FPGA vendor guidelines for SSO budgets.
4. Perform pre-layout simulation-based solution space exploration to develop optimum constraints for high-speed signals. This will enable a constraint-driven PCB layout.
5. Reduce the number of layers. Reducing the PCB layers improves the electrical quality of the signals.
6. Ensure good return current paths for all the signal transmission paths. Good return paths are achieved by ensuring an adjacent power/ground plane for all the signal planes.
7. Reduce the number of vias in the design. Vias increase the inductance on a signal. By reducing the vias the inductance on the signal is reduced, and thus the signal is transmitted and received at a higher quality.
8. Keep the trace length of signals to a minimum. Reducing the trace length reduces transmission line losses, and thus improves signal quality.
9. Budget the power supplies and ensure the power supplies are sufficient to handle system needs.
10. Ensure sufficient decoupling capacitors of the appropriate values are placed on the power supply.

Most importantly, ensure that the project has a highly experienced board designer familiar with the unique nature of FPGAs and review the design thoroughly at every step. Make sure to review the final layout and the schematics, and make sure that they are all consistent with the FPGA design and pin-outs. Also, ensure good communication between the PCB design team and FPGA design team.



Cadence is transforming the global electronics industry through a vision called EDA360. With an application-driven approach to design, our software, hardware, IP, and services help customers realize silicon, SoCs, and complete systems efficiently and profitably. www.cadence.com