Accelerating FPGA Board Design & Ensuring Correctness

Taray, Inc

Nagesh Gupta & Abhijit Athavale

Session # 135

Taray

Presented at

cadence designer network

cdn™ LIVE

Silicon Valley 2007

For manufacturers of telecom, networking, computing and electronic gear, design of complex PCBs that use high-density FPGAs is a process fraught with errors, delays and budget overruns. The design team must overcome a multitude of challenges to complete the board design accurately and on time.

FPGA Board Design Challenge

Parallel FPGA and PCB Design Flows

Today, FPGA design and PCB design flows are along parallel lines. The FPGA designer will generate the top-level design and constraints and then run it through synthesis, place & route and simulation. Once the logic design meets all timing and performance criteria, it is handed-off to the PCB designer for symbol generation schematic entry and layout. The logic and PCB design teams are not coordinated and working towards individual goals of completing the logic design and PCB design as opposed to an optimal system design goal. Obviously, both designers are working to get their designs completed on time, but are now in effect working against each other, thus reducing productivity and ultimately, the entire project suffers from delays and increased costs.

System Architecture

This is the key phase in system design. If done correctly, rest of the design phases can proceed smoothly. Due to the disconnected nature of the existing design flows, a feature/cost trade-off is very difficult to do up front. The architect always faces questions such as - is it possible to increase the performance of the system by increasing the width of the interface? If an 1152-pin package supports 2.4GB throughput, and costs $1x does an upgrade to a 1508 costing $1.2x be able to support a 10GB throughput? The trade-off analysis process usually takes several weeks of effort before any results are visible, and as such avoided because of the big investment in time.

FPGA pin assignment

The logic designer must first research the FPGA I/O architecture and pin requirements for each interface that he is planning to use. Then he must place the interfacing components, such as memories and other processor components according to the FPGA design rules, allocate pins, and create a constraints file and finally create top level RTL. Finally, he must verify pin DRC using FPGA vendor's pin-out verification tool.

Typically, this is a time consuming, and an error prone manual process. The task of pin selection is a multi-dimensional problem, making it extremely difficult when done manually. The following are some of the aspects to consider during the FPGA pin-out process:

1. **Logical constraints**: The pin-out should satisfy the requirements for the protocol underlying the interface. For example, successful data capture for a source-synchronous bus requires that the data and the corresponding clocks are pinned out correctly.
2. **Electrical constraints**: Electrical constraints are related to FPGA DRCs, among other things. FPGAs have complex banking structures and a detailed set of rules associated with this. The Electrical signaling standards of the interface will determine if an interface can use a particular bank.

3. **Physical constraints**: Physical constraints are related to the placement of different devices on the board. Pins should be selected to minimize the wire crossings when the connections are viewed as a rats nest. Pin selection without consideration to the physical constraints can increase the number of layers required to route the board.

Most designers use the following approach for pin assignment:

1. Assign pins to meet the logical constraints. Several front-end point tools help the user for correct logical pin assignment.
2. Ensure that the pin assignment will meet the FPGA DRCs. FPGA banking rules can be verified using the FPGA vendor place-and-route tools.
3. The physical constraints are only observed to the extent of selecting pins on the same side of the FPGA as the location of the corresponding interface device.

There are two key problems with this approach:

1. Correctness is not ensured.
2. The pin selection may not be optimal.

Once the pin-out is complete, the board design can start. First version of the pin-out does not work many times forcing the PCB designer to make changes such as pin-swaps. Pin-swaps make the logic designer go back and possibly change RTL, placement, and/or timing constraints necessitating even more changes to the pin-out. The PCB designer will now start designing with the new pin-out, which may not work, requiring more logic design iterations. The logic designer might discover an error in the design or may need to add a feature requiring a new pin-out. This process can go on for months or weeks before an acceptable pin-out emerges. This pin-out, while acceptable may not be the most optimal. Typically, un-optimized boards have more layers, more vias, and signal and power integrity problems than optimized ones and hence are more expensive to manufacture, test and deploy.

Symbol and Schematic Generation

PCB schematic generation is a drawn out manual process that requires predefined pin-out identification, symbol creation, and schematic entry. The PCB designer must draw schematics to connect all components; mark critical nets for layout, connect-up power supplies, debug interfaces & configuration pins. Due to very nature of this process, it takes multiple man-weeks or months to complete such schematics. Several high-priced PCB designers working for multiple weeks or months are a sizeable cost item. A pin-out change will cause multiple iterations in the schematic generation process delaying the project even further. Every iteration can take up to two weeks.

Solving the Challenge with a Unified FPGA/PCB Design Flow

Solving this challenge requires a new approach that unifies system design without changing any of the existing tool flows. This avoids steep learning curves and does not require un-learning tools. Both logic and PCB design teams work on team goals and not individual goals.

The new 7Circuits tool from Taray exemplifies this approach. Key features of 7Circuits are explained below:

- **Automated pin assignment and connectivity generation**
  - Meet FPGA IO DRC requirements
  - Power pins voltage mapping based on bank IO standards
  - Terminations topology selection based on net IO standard
- **Layout cross-over optimization**
  - Optimization to reduce overall net length and length matching for PCB layout
  - Improves signal integrity
  - Reduces cost by reducing number of PCB layers
- **Generating required symbols and the entire schematic design database**
  - Easily integrates with the existing design flows
  - Reuses existing library symbols in the flow

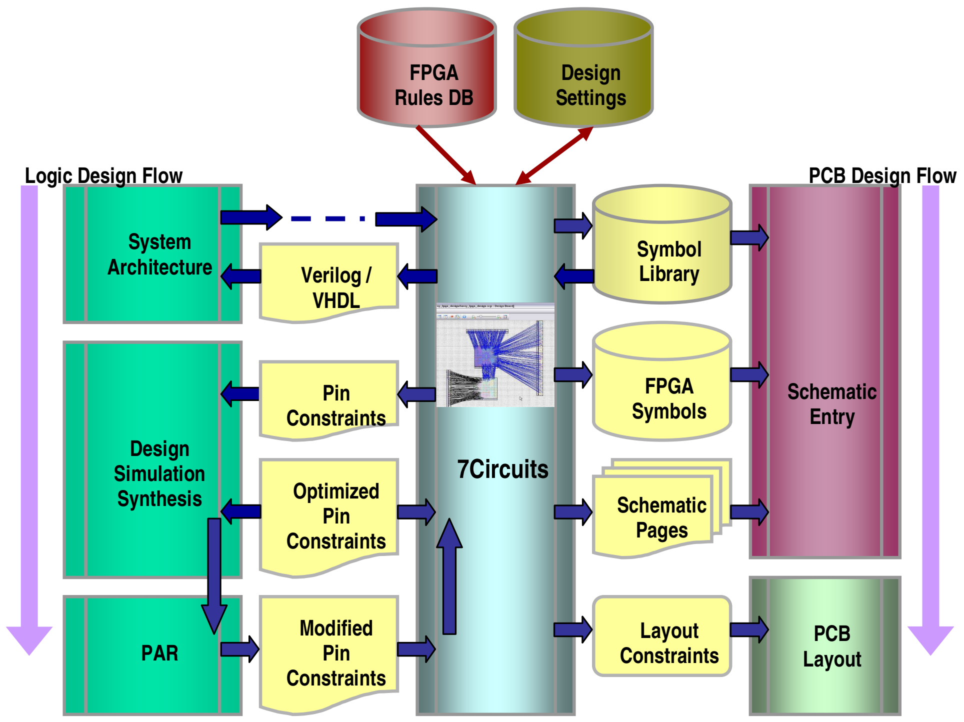


Figure 1. Design flow using the 7Circuits' unified approach .

System Architecture

The architect starts with the design floor plan in this step. Using parts from the library, he can quickly design interface part models. He has complete flexibility to try different topologies (rotation, side switching, address and data busses in serial and parallel configuration etc.), for interfacing parts to find optimal locations. FPGA logic resources needed for each interface are automatically calculated. The architect then defines higher-level design rules required by logic and board designers. 7Circuits provides a very simple interface to enable such tasks. Figure 2 shows a screen shot of the 7Circuits board canvas. Users can easily bring up different FPGAs/memories/other interface components and try out different configurations in minutes. With 7Circuits, a task that takes several days and iterations is completed within minutes.

Formatted: Font: Book Antiqua, 11 pt, Complex Script Font: 11 pt

Deleted: Figure 2

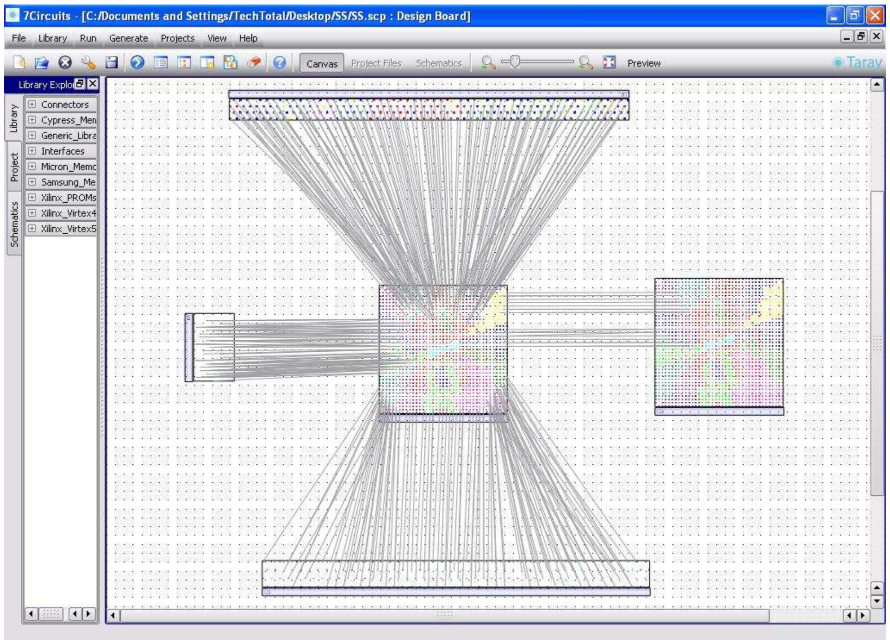
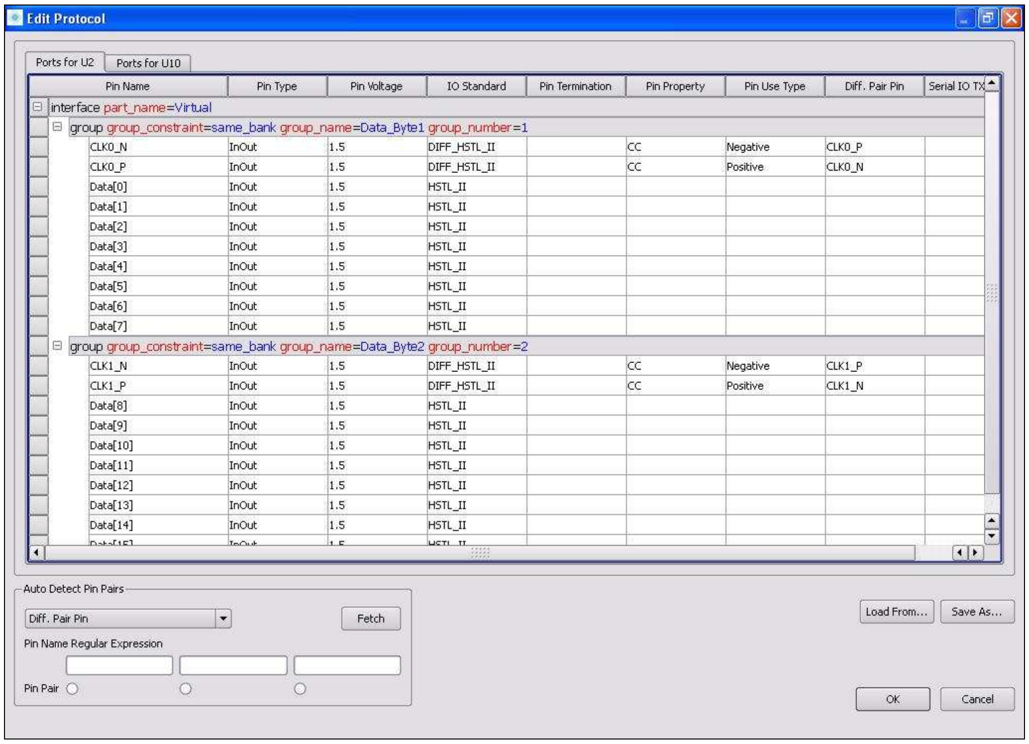


Figure 2. Screen shot showing the board canvas in 7Circuits



Pin Name	Pin Type	Pin Voltage	IO Standard	Pin Termination	Pin Property	Pin Use Type	Diff. Pair Pin	Serial IO TX
interface part_name=Virtual								
group group_constraint=same_bank group_name=Data_Byte1 group_number=1								
CLK0_N	InOut	1.5	DIFF_HSTL_II		CC	Negative	CLK0_P	
CLK0_P	InOut	1.5	DIFF_HSTL_II		CC	Positive	CLK0_N	
Data[0]	InOut	1.5	HSTL_II					
Data[1]	InOut	1.5	HSTL_II					
Data[2]	InOut	1.5	HSTL_II					
Data[3]	InOut	1.5	HSTL_II					
Data[4]	InOut	1.5	HSTL_II					
Data[5]	InOut	1.5	HSTL_II					
Data[6]	InOut	1.5	HSTL_II					
Data[7]	InOut	1.5	HSTL_II					
group group_constraint=same_bank group_name=Data_Byte2 group_number=2								
CLK1_N	InOut	1.5	DIFF_HSTL_II		CC	Negative	CLK1_P	
CLK1_P	InOut	1.5	DIFF_HSTL_II		CC	Positive	CLK1_N	
Data[8]	InOut	1.5	HSTL_II					
Data[9]	InOut	1.5	HSTL_II					
Data[10]	InOut	1.5	HSTL_II					
Data[11]	InOut	1.5	HSTL_II					
Data[12]	InOut	1.5	HSTL_II					
Data[13]	InOut	1.5	HSTL_II					
Data[14]	InOut	1.5	HSTL_II					

Auto Detect Pin Pairs: Diff. Pair Pin

Figure 3. Easy-to-Use Logical Constraint Editor within 7Circuits.

Pin Assignments

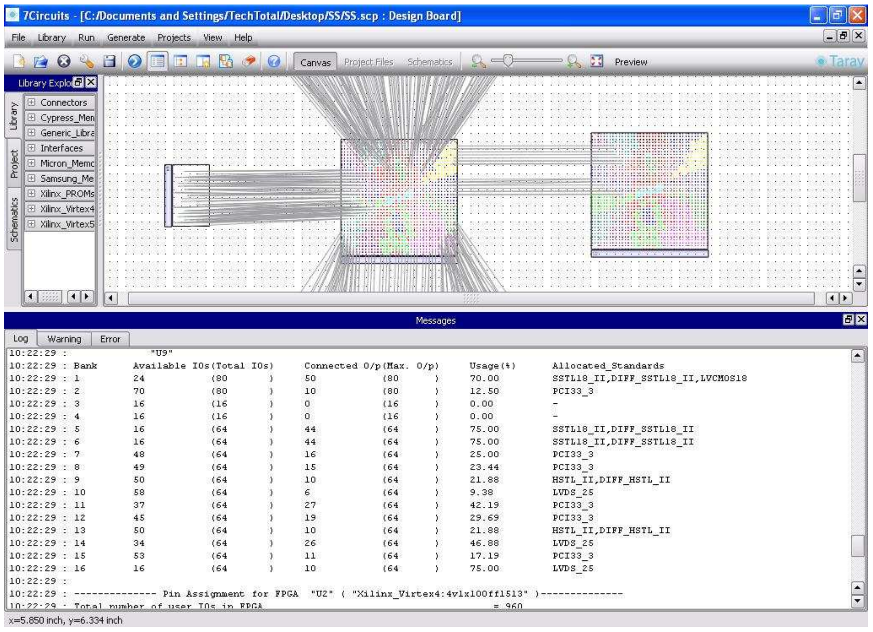
7Circuits has an internal rules engine that uses logical, physical, and electrical constraint information to generate pin assignments. Pin-outs generated in this unified manner are optimized from all respects. At the same time, top-level Verilog/VHDL module, schematic symbols, and pages are generated as inputs for the board design tools.

In addition to solving the pin assignment problem in a unified manner, 7Circuits also ensures correctness of the board design in several areas:

For example:

1. **Correct voltage connections**: FPGAs have multiple voltage rails that must be connected. Moreover, the voltage level of the connection depends upon the logic connections to a particular bank. For example, if an IO standard such as SSTL18 is used in a bank, a V*cc* voltage level of 1.8 volts, and a V*ref* voltage level of 0.9 volts must be connected. Any mistakes here result in a board re-spin. 7Circuits enforces all of the FPGA IO DRCs, thus preventing re-spins.

2. **Dual-purpose pins**: In Xilinx FPGAs, if the on-die termination feature is used in a bank, certain other pins automatically become special purpose pins. This rule is also enforced by 7Circuits.


7Circuits - [C:/Documents and Settings/TechTotal/Desktop/SS/SS.scp : Design Board]
File Library Run Generate Projects View Help
Canvas Project Files Schematics Preview
Taray
Library Explo
Connectors
Cypress_Men
Generic_Libra
Interfaces
Micron_Memc
Samsung_Me
Xilinx_PROMs
Xilinx_Virtex4
Xilinx_Virtex5
Library
Project
Schematics
Messages
Log
Warning
Error
x=5.850 inch, y=6.334 inch

```
10:22:29 :                "U9"
10:22:29 : Bank       Available IOs(Total IOs)     Connected O/p(Max. O/p)     Usage(%)      Allocated_Standards
10:22:29 : 1          24            (80       )    50            (80      )    70.00         SSTL18_II,DIFF_SSTL18_II,LVCMOS18
10:22:29 : 2          70            (80       )    10            (80      )    12.50         PCI33_3
10:22:29 : 3          16            (16       )    0             (16      )    0.00          -
10:22:29 : 4          16            (16       )    0             (16      )    0.00          -
10:22:29 : 5          16            (64       )    44            (64      )    75.00         SSTL18_II,DIFF_SSTL18_II
10:22:29 : 6          16            (64       )    44            (64      )    75.00         SSTL18_II,DIFF_SSTL18_II
10:22:29 : 7          48            (64       )    16            (64      )    25.00         PCI33_3
10:22:29 : 8          49            (64       )    15            (64      )    23.44         PCI33_3
10:22:29 : 9          50            (64       )    10            (64      )    21.88         HSTL_II,DIFF_HSTL_II
10:22:29 : 10         58            (64       )    6             (64      )    9.38          LVDS_25
10:22:29 : 11         37            (64       )    27            (64      )    42.19         PCI33_3
10:22:29 : 12         45            (64       )    19            (64      )    29.69         PCI33_3
10:22:29 : 13         50            (64       )    10            (64      )    21.88         HSTL_II,DIFF_HSTL_II
10:22:29 : 14         34            (64       )    26            (64      )    46.88         LVDS_25
10:22:29 : 15         53            (64       )    11            (64      )    17.19         PCI33_3
10:22:29 : 16         16            (64       )    10            (64      )    75.00         LVDS_25
10:22:29 :
10:22:29 : -------------- Pin Assignment for FPGA  "U2" ( "Xilinx_Virtex4:4vlx100ff1513" )--------------
10:22:29 : Total number of user IOs in FPGA                                          = 960
```

Figure 4. 7Circuits enforces all FPGA design rules during pin assignment.

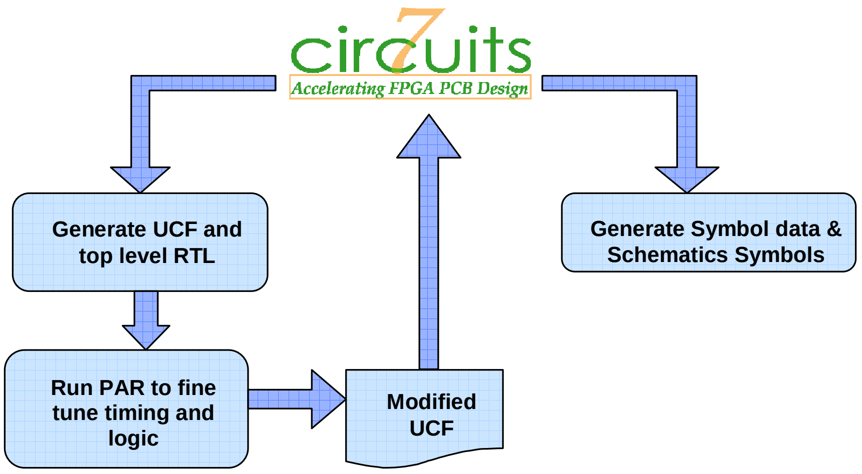
Synchronized FPGA/PCB Design

The logic designer can complete the FPGA design, simulation, and synthesis while the PCB designer starts connecting the FPGA symbols and schematic pages to the rest of the schematic using the inputs providedabove. At this point, both teams are working from the same information, and hence the entire design is fully synchronized. As described next, 7Circuits also supports incremental changes, a must in real life designs, from here on.

FPGA Design Iterations

The FPGA design starts with the pin constraints created by 7Circuits. However, the logic designer may need to change the FPGA pin constraints. One reason for making minor changes to the pin constraints generated by 7Circuits is to meet logic timing. In a real design, there will be several iterations before the pin-out is finally frozen.

7Circuits reads the modified pin constraints file and updates board design files based on the modified pin constraints file. This iterative process becomes very simple with 7Circuits, as shown below:



circuits
Accelerating FPGA PCB Design
Generate UCF and top level RTL
Generate Symbol data & Schematics Symbols
Run PAR to fine tune timing and logic
Modified UCF

Figure 5. Iterating for logic design changes and final schematic/symbol generation.

Integration with the existing flows

For any new tool to be useful, it is imperative that there is a seamless integration with existing design and tool flows. Any new information that is required should be minimal. All the information should build up from the existing information in design libraries. 7Circuits readily integrates with the existing Allegro DE HDL symbol library databases.

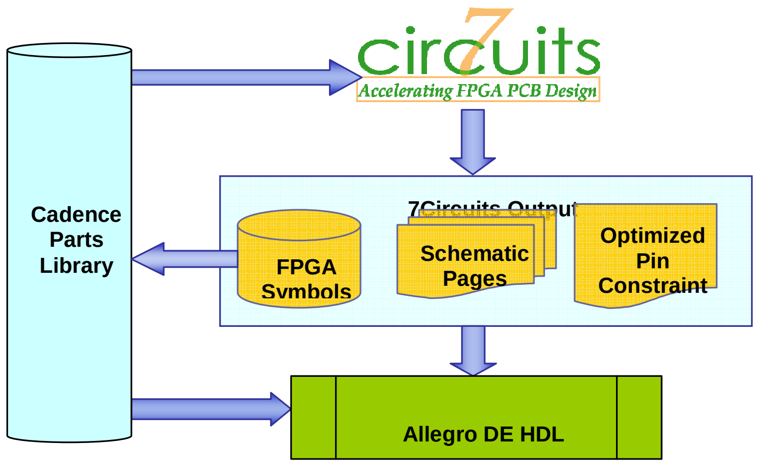


Figure 6. 7Circuits integrates well with Allegro DE HDL.

As illustrated in the above diagram, 7Circuits uses most of the data already present in the schematic symbol database and generates the schematic pages using existing symbols. The logical constraints required for optimization can be entered in the 7Circuits model editor. This information is saved as 7Circuts view of the part in the Allegro DE HDL symbol database. Moreover, 7Circuits databases are stored in open format – scripts can be used to create and save 7Circuits libraries.

Conclusion

PC Board designs based on multiple, dense FPGAs is a reality in all markets. As FPGAs are fully reconfigurable, logic designs, and hence pin-outs, keep changing till the last minute. This introduces a new challenge that requires new solutions. The 7Circuits tool from Taray solves this problem very effectively by synchronizing the PCB and FPGA design flows. 7Circutis automates FPGA pin assignments, generates schematic symbol and pages using the information from the frozen logic design. As such, PC board design projects are accelerated while ensuring correctness.