FPGA and Programmable Logic JOURNAL
www.fpgajournal.com
TRENDS
TECHNOLOGY
TOOLS
TECHNIQUES

It's Not All About the FPGA Anymore

by Bruce Riggins – Mentor Graphics Corporation

Introduction

FPGA logic design no longer rules the project schedule like it once did. Twenty years ago, implementing the logic in 20 and 24-pin programmable logic devices, such as 22V10s or PAL16R8s, was tricky, given the state of programmable logic tools at the time, but doing so certainly didn't consume the lion's share of the overall design effort. Back then, the deciding factor in project schedules was typically the design of the PCB onto which those devices were placed. As the sophistication of programmable devices has increased – evolving into the FPGAs commonly used today – the design-cycle 'long pole' has changed hands several times, with each change of ownership being dictated by a combination of device complexity and the overall state of EDA software available at the time to address various design challenges.

For the last eight or ten years, the design of FPGA logic has held the distinction of being the most intricate piece of the larger, board-level, project. As such, team leaders have usually deferred to the FPGA designers when it comes to setting schedules. However, that is beginning to change. Companies utilizing very large FPGAs, and even those using multiple, smaller devices, are starting to realize that the majority of the work is being borne by the PCB designer, who is tasked with meeting routing, timing, signal integrity, cost, manufacturability, reliability and a host of other constraints – all after being handed a database where the PCB itself has been given little, if any, up-front consideration. Device design complexity vs. board design complexity has come full circle and is settling somewhere in the middle. A recent EE Times survey suggests that while roughly 33% of companies have some mechanism to address the FPGA/PCB co-design issue today (usually with internally-developed tools and scripts), within two years that is expected to grow to 46%.

This article discusses some of the difficulties posed by integrating a large number of FPGA pins into a system. Note that it is not the size of the FPGAs

used nor how many, but the total number of FPGA pins that most directly influence the amount of effort required to optimize the complete design. For the purpose of this article, the term 'PCB' has been applied to loosely represent everything that isn't inside the FPGA, generally the schematic and the PCB.

FPGA Evolution

In order to understand why the design of the PCB has once again assumed the somewhat dubious honor of being a major factor in the design cycle, it's helpful to quickly review even the recent history of FPGA packages. In 1998, Xilinx introduced their Virtex device, which contained 680 pins, 512 of which were usable, all in a 1mm-pitch BGA. By late 2000, the Virtex II was announced, which contained 1108 usable pins, also in a 1mm-pitch 1517-pin BGA. By early 2002, packages had grown to 1704 pins, and in the middle of last year 1760 pin packages were announced with the Virtex 4. Altera has followed a similar path.

FPGAs Increased Design Influence

Given the industry's system-on-chip trend, where increased functionality is being driven into fewer, more highly-integrated components, the FPGA designer has been granted the authority to influence a larger and larger percentage of the overall system and, as a result, the total number of pins in the design. Unfortunately, in many cases, that authority has not come with the associated responsibility to insure that the usage of those pins has been optimized for the system and not just for the FPGA. This is not necessarily purely the fault of the FPGA designer. The fact is, until recently, EDA tools have not provided a viable means of enabling the FPGA designer to adequately consider the consequences of the I/O pin assignments on the rest of the system.

To illustrate this, consider a simple example with a relatively small FPGA and a connector. Figure 1a shows a 'rat's nest' view of a PCB where the I/O pin assignments for a 32-bit data bus were chosen by the FPGA place and route tools. Having no view into the PCB, the pin locations were chosen to provide the best results for the FPGA. Figure 1b shows the actual routing on the PCB. Notice the severe amount of 'tromboning' that the PCB routing tools had to insert into some of the traces in order to equalize trace lengths for the bus.

FPGA and Programmable Logic JOURNAL

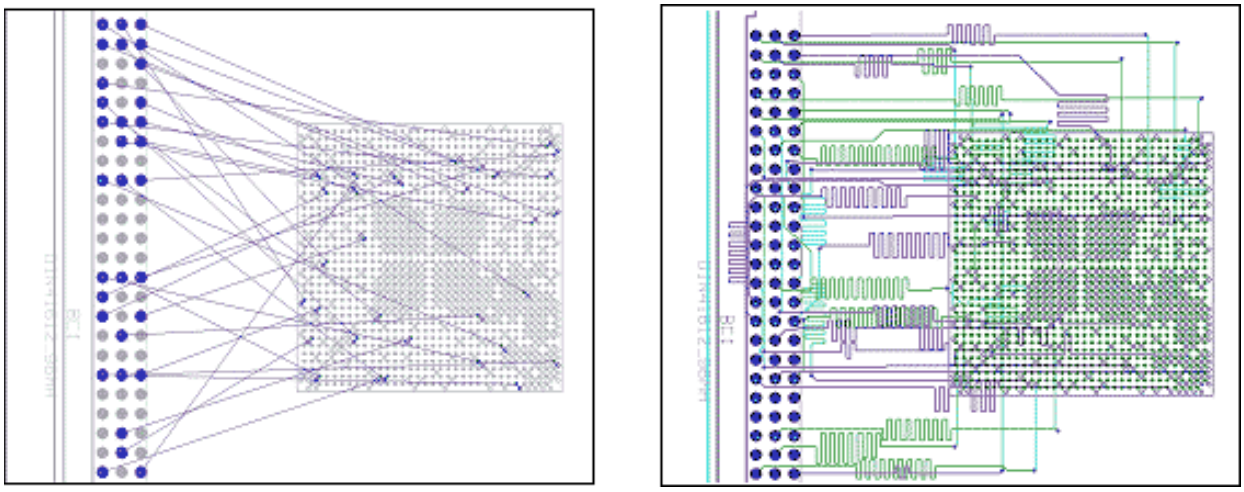

Figure 1 a & b : - Inferior System-Level FPGA I/O Pin Assignment. Figure 1a (left) shows a PCB 'rat's nest' view. Figure 1b shows the actual PCB routing, exhibiting significant 'tromboning'.

In Figure 2, the data bus pins were moved to improve their proximity to the connector. Understandably, the PCB routing is significantly improved: the longest trace has been reduced from 3.6 to 1.8 inches and is 320ps faster.

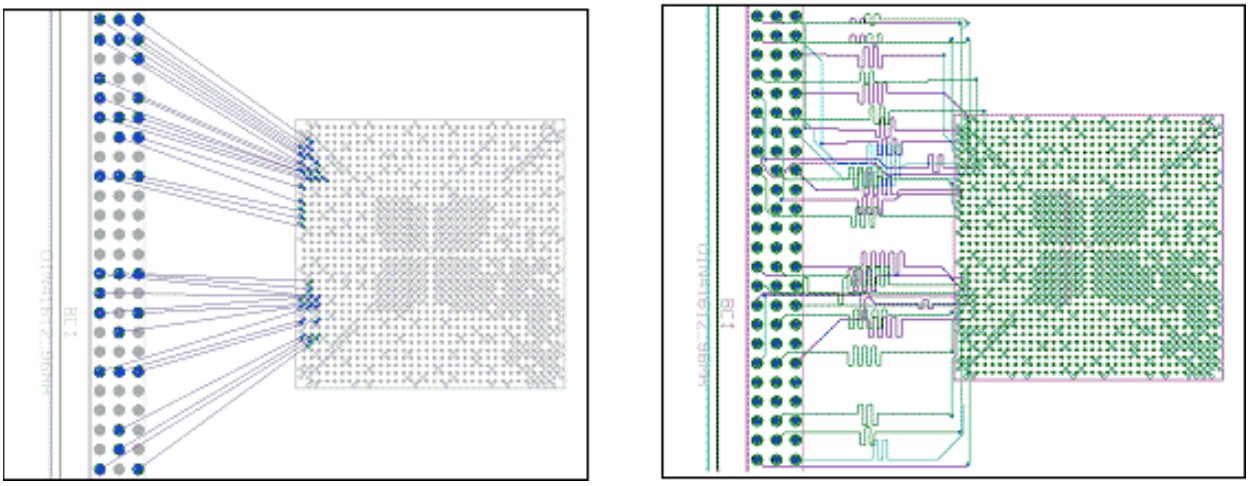

Figure 2 a & b: - Improved System-Level FPGA I/O Pin Assignment. Figure 2a shows the PCB 'rat's nest' view of improved FPGA pin assignment. Figure 2b shows that while 'tromboning' remains, it has been reduced considerably.

This improvement in PCB routing does not come without potential costs to the FPGA. Optimizing the design for PCB routing while ignoring the FPGA is as shortsighted as ignoring the PCB while deferring to the FPGA. In other words, both the PCB and FPGA designers need to come together to optimize the pin assignments for both the FPGA and the PCB. To close the loop on this example, the FPGA designer would have to re-synthesize and re-run place and route on the FPGA, with the new pin assignments, to insure that it still meets its timing requirements.

System Timing and Signal Integrity

The example in Figures 1 and 2 hint at a third discipline that can be affected by FPGA I/O assignments: system timing. In this case we were able to reduce the overall signal length by 320ps. In a high-speed design this can mean the difference between meeting and failing timing. Extrapolating this example to a more complex design, it should be easy to see that optimizing the FPGAs I/O pin assignments can help to reduce PCB layers, as well as the number of vias. For signals operating in the multi-gigahertz frequency range, vias operate like miniature antennas, degrading signal quality with every added via. This is not a "just do your best" option for successful operation of the system. A case in point: the industry-standard PCI Express bus specification explicitly suggests the use of less than two vias per trace, and trace length matching to within .005 inches. So now a fourth design discipline has entered into the equation – signal integrity.

In addition to the inter-twined effects described above, one of the major challenges posed by large FPGAs is the conceptually straightforward process of creating the symbols, placing them on a schematic, and wiring them to the rest of the design. The prospect of creating a 1200 or 1500-pin symbol is not only daunting but realistically unfeasible. Symbols representing such large devices simply cannot be placed on a single schematic sheet, and even if they could, there would be no room left for the nets. The symbols need to be fractured, but the schematic design tools need to be able to recognize those fractures as parts of a larger, homogenous component. Once the fractures are created and placed, they must still be connected to the rest of the system and making net connections to 1500 pins is no less overwhelming than creating the symbols in the first place.

FPGA and Programmable Logic JOURNAL

Complicating the situation even further, most FPGAs go through a minimum of three to five iterations before converging on a final I/O pin assignment. Somebody – or something – needs to maintain the symbols and schematics with each design change and communicate and synchronize those changes between every engineer, tool and database. Considering that 30% of designs contain two or more FPGAs, it is impractical to attempt to manage all of this manually.

Given the scarcity of tools to help manage this effort, many companies have simply resorted to locking the pin assignments made by the FPGA designer. The effect of this is to tie the PCB designer's hands behind his back in his efforts to optimize the routing and in essence turning the FPGA into a programmable ASIC.

New FPGA/PCB Integration Tools

EDA vendors are starting to get the message, and are offering new tools that enable FPGA I/O design in the context of the PCB. These tools begin to bring FPGA and PCB designers together in a common environment so that each can weigh the tradeoffs associated with a given choice of I/O pin assignments. They automatically create the FPGA symbols (fractures) and schematics and maintain those objects as the design matures. They also create and update the necessary FPGA synthesis and place & route files based on new pin assignments to close the loop between the FPGA and the schematic/PCB. Finally, they reopen the flexibility of the FPGA to the PCB designers, enabling them to swap pins, but only those that have been defined as having equivalent and legal swapping characteristics. One tool currently available to address these issues is I/O Designer from Mentor Graphics, introduced in early 2004.

FPGA and Programmable Logic JOURNAL

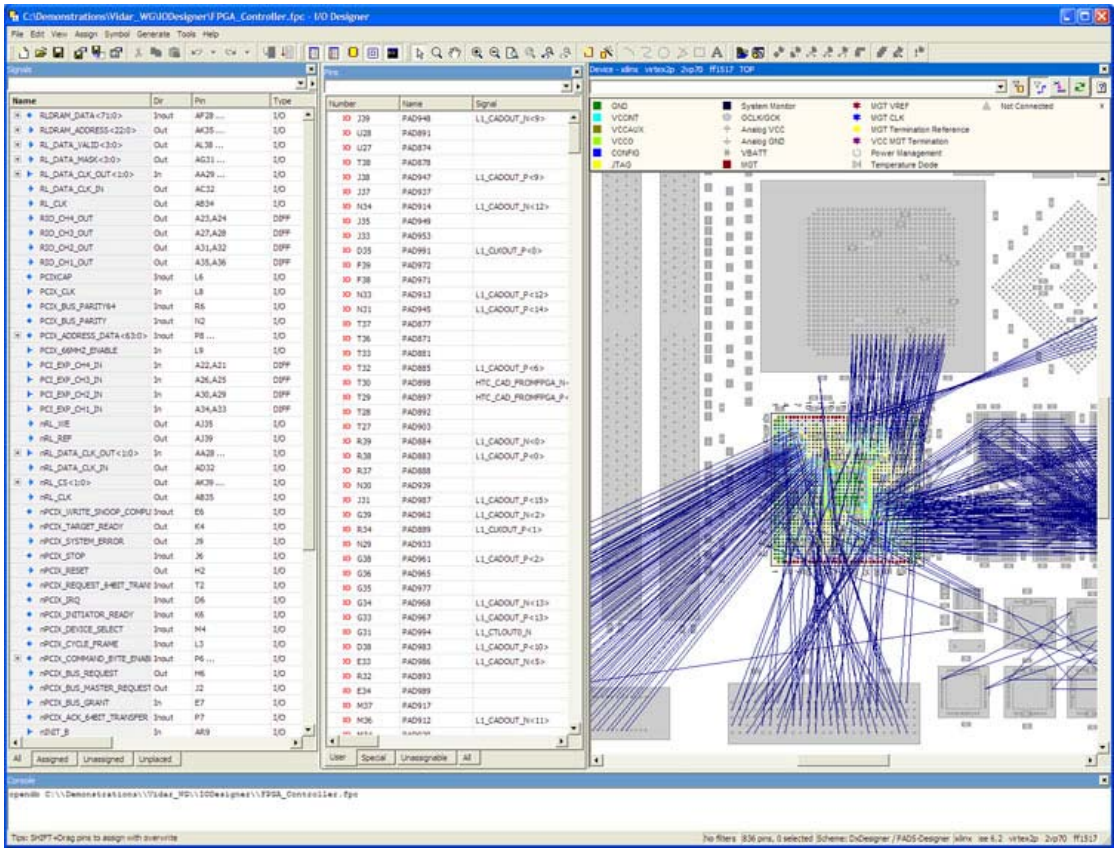

Figure 3 - In this screenshot of I/O Designer, engineers can clearly see the FPGAs signals and pins, as well as the connections that the FPGA has to other components on the board.

Conclusion

Companies can no longer focus solely on FPGA design goals while disregarding the issues that FPGA pin assignments create for PCB routing and performance. Conversely, they cannot focus solely on PCB layout either, as layouts that over-constrain the FPGA will create problems for the FPGA place & route tools, making it difficult to achieve timing closure. What is needed is a process and tool set that enables engineers from multiple disciplines to collaborate in a common environment to define FPGA I/O pin assignments that are optimized for the entire system.

by Bruce Riggins – Mentor Graphics Corporation

September 15, 2005