

Abstract: Configuration Relocation and Defragmentation for Reconfigurable Computing¹

Katherine Compton, James Cooley, Stephen Knol

Department of Electrical and Computer Engineering
Northwestern University
Evanston, IL USA
kati@ece.nwu.edu

Scott Hauck

Department of Electrical Engineering
University of Washington
Seattle, WA USA
hauck@ee.washington.edu

Abstract

Custom computing systems exhibit significant speedups over traditional microprocessors by mapping compute-intensive sections of a program to reconfigurable logic [Hauck98]. However, the high overhead of reconfiguration can limit the execution times achievable with these systems. Research has shown that the ability to relocate and defragment configurations on an FPGA dramatically decreases the overall configuration overhead [Li00]. We therefore explore the adaptation of the Xilinx 6200 series FPGA for relocation and defragmentation. Due to some of the complexities involved with this structure, we also present a novel architecture designed from the ground up to provide relocation and defragmentation support with a negligible area increase over a generic partially reconfigurable FPGA.

Relocation and Defragmentation

The basic partially reconfigurable FPGA design is powerful, but it faces limitations imposed by configuration locations determined at compile time. If two different configurations were mapped at compile time to overlapping locations in the FPGA, they can not be used simultaneously. The ability to relocate one or both of these configurations to a new location might allow for concurrent use. Defragmentation extends this idea to move the configurations already present on the FPGA to new locations, consolidating unused area. Therefore, an incoming configuration may be able to be programmed onto the FPGA without removing any of the configurations already present.

Xilinx 6200 for Relocation

We chose the Xilinx 6200 FPGA [Xilinx96] to consider for use with configuration relocation because it is a commercial partially reconfigurable FPGA with a regular cell layout and local routing structure. This facilitates the relocation of a configuration from one area of the FPGA to another.

When relocating a mapping in a 2D array, there are a few requirements that we need to meet in order for its functionality to be preserved. First, the routing programmed into each cell must be changed to reflect the

overall rotation or flip of the configuration. Second, a cell must also be shifted by the same horizontal and vertical offsets as the entire configuration being relocated. Third, the relative routing between cells within a configuration must remain intact. Finally, the relative position of a cell within a configuration must be maintained.

The actual changes of the routing can be performed through the modification of the programming information for the affected cell. Position information is controlled by the address associated with the programming data, and so modifications to this address change the location of the cell.

Limitations of the 6200

There are a number of issues that reduce the appeal of the 6200 for relocation. First, the I/O of a configuration must be re-routed with each move, as the pin locations are fixed at fabrication of the circuit. Second, in reality the 6200 is not completely homogenous. A hierarchical long-distance routing structure is present within the architecture. This prevents configurations that use this routing from being arbitrarily relocated, as they must be placed in a location with identical connections to the distance routing.

Additional concerns arise when considering the possibility of defragmentation on the 6200. We can read back the configuration, perform the necessary modifications to the programming and address information, and write it back to the array in its new location. This process would be slow, as it entails reading and writing between the array and external circuitry. Alternately, by adding additional routing that would allow for a more limited (shift-only) defragmentation this process can be performed potentially more quickly, but we lose flexibility in the defragmentation and incur a large area penalty. Finally, defragmenting a 2D array is a complex operation. Essentially, the FPGA must go through a floorplanning stage each time it is defragmented, which is a time-consuming process.

New Architecture

We propose a new architecture designed specifically to exploit the benefits of relocation and defragmentation. We will refer to this architecture as the R/D (Relocation / Defragmentation) FPGA. Using a few simple concepts in

¹ The complete paper is available at <http://www.ece.nwu.edu/~kati/publications.html>

the design phase of the FPGA, we can ensure that the architecture is suitable for relocation and defragmentation.

The ability to selectively program portions of the FPGA is critical to the philosophy of relocation and defragmentation, and therefore the architecture must be partially reconfigurable. We also require homogeneity to ensure there are no functional obstacles to moving a configuration from one location to any other location within the boundaries of the array. Additionally, by using a bus-based input/output structure, we have a location-independent method to provide external signals to the individual configurations. Finally, we use a one-dimensional FPGA design. This simplifies the calculations for the configuration movements as well as the hardware required.

Architecture Specifics

The architecture of the R/D FPGA is shown in Figure 1. Similar to the partially reconfigurable FPGA, the memory array of the R/D FPGA is composed of an array of SRAM bits. However, the column decoder, multiplexer, and input tri-state drivers have been replaced with a structure we term the "staging area". This staging area is a small SRAM buffer, which is essentially a set of memory cells equal in number to one row of programming bits in the FPGA memory array, where a row of logic cells contains a number of rows of configuration bits. Each row, and therefore the staging area, contains several words of data.

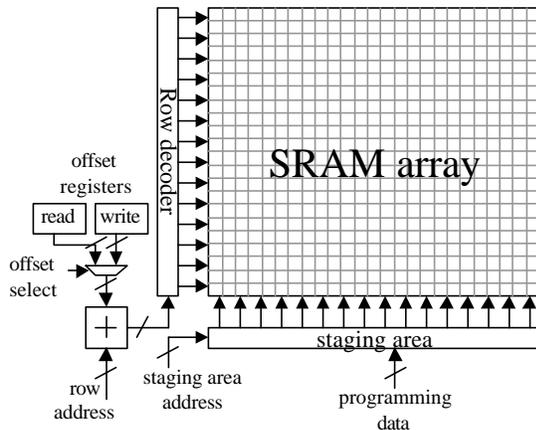


Figure 1: The R/D FPGA architecture

The staging area is filled in an addressable fashion one word at a time. Once the information for the row is complete in the staging area, the entire staging area is written in a single operation to the FPGA's programming memory at the row location indicated by the row address. Apart from the addressability of the staging area, it is similar in function to a structure present in the Xilinx Virtex FPGA [Xilinx99].

R/D Operation

Relocation of a configuration is accomplished by altering the row address provided to the row decoder. This allows

for a simple way to dynamically locate individual configurations to fit available free space. If we consider a number of full row width configurations that would have been programmed onto a basic partially reconfigurable FPGA, we are only adding $\langle \# \text{ rows} \rangle + 1$ cycles to the configuration time in order to allow relocation.

Defragmentation of the R/D FPGA requires that rows be moved from existing locations on the FPGA to new locations without overwriting any necessary data. Using the R/D architecture, this can be done in $\langle \# \text{ rows} \rangle * 2 + 2$ read/write cycles.

Estimated Size Comparison

We modeled the sizes of the basic partially reconfigurable FPGA and the R/D FPGA using the same structures used in [Li00]. The sizes are estimated using the areas of tileable components. For this size evaluation, we modeled each with a megabit (2^{20} bits) of configuration data in a square layout ($\# \text{ rows} = \# \text{ columns}$). The area of the partially reconfigurable array was calculated to be $8.547 \times 10^9 \lambda^2$, while the area of the R/D FPGA was calculated to be $8.549 \times 10^9 \lambda^2$, a difference of .02%. According to this comparison, the R/D FPGA has only a negligible size increase over a basic partially reconfigurable FPGA.

Conclusions

Although a stylized version of the Xilinx 6200 FPGA can be converted to handle relocation and even defragmentation, the realities of the architecture pose significant drawbacks to both relocation and defragmentation. We therefore presented a new architecture to more fully support these ideas. The R/D architecture is designed to require little additional run-time effort on the part of the CPU, and requires only a negligible area increase (.02%) over a basic partially reconfigurable FPGA.

References

- [Hauck98] S. Hauck, "The Roles of FPGAs in Reprogrammable Systems", *Proceedings of the IEEE*, Vol. 86, No. 4, pp. 615-638, April 1998.
- [Li 00] Z. Li, K. Compton, S. Hauck, "Configuration Caching for FPGAs", in preparation for *IEEE Symposium on Field-Programmable Custom Computing Machines*, 2000.
- [Xilinx96] *XC6200: Advance Product Specification*, Xilinx, Inc., San Jose, CA: 1996.
- [Xilinx99] *Virtex™ Configuration Architecture Advanced Users' Guide*, Xilinx, Inc., San Jose, CA: 1999.