Mapping of an Automated Target Recognition Application from a Graphical Software Environment to FPGA-based Reconfigurable Hardware

Benjamin Levine, Senthil Natarajan, Chandra Tan, Danny Newport, Don Bouldin
Microelectronic Systems Research Laboratory
University of Tennessee, Knoxville, TN 37996-2100
{levine, senthil, chandra, newport, bouldin}@microsys0.engr.utk.edu

Abstract

A significant obstacle to the widespread adoption of FPGA-based configurable computing hardware has been the difficulty of mapping applications onto this hardware. We are developing a software development system called CHAMPION to automate the process of mapping applications in the graphical software environment Khoros to multiple FPGA-based architectures. The work described here consists of the development of requirements for the library primitives used by CHAMPION and the manual mapping of an automatic target recognition algorithm onto FPGA hardware.

1. Introduction

FPGA-based reconfigurable computing systems have been shown to have considerable performance advantages over conventional processor-based systems for certain types of applications [1][2]. Despite this, these systems are not in widespread use. One of the main obstacles to their use has been the difficulty inherent in implementing applications on this hardware. Current systems require the user to be familiar with digital hardware design, hardware description languages, and device-specific tools such as compilers and place and route software. We are developing a software development system, called CHAMPION, for the automated mapping of applications in the Khoros Cantata graphical programming environment to FPGA-based reconfigurable computing hardware. Khoros is already widely used for algorithm development and simulation. CHAMPION will be used to map Khoros applications to a reconfigurable coprocessor, speeding execution of the application and greatly increasing designer productivity. Since multiple hardware architectures can be targeted, CHAMPION should also be able to map a completed algorithm to reconfigurable hardware to be used independently of a host computer. Before the automated system can be completed, manual mapping of algorithms from Khoros to hardware is necessary to gain familiarity with the steps required, and also to give benchmarks for the performance and time for completion achievable using manual techniques.

2. Library Primitives

Khoros contains libraries of operators that are represented onscreen by an icon, or glyph. These glyphs are connected in a manner representing the dataflow to create an application. A glyph can represent a single, simple function such as addition, a high-level function such as convolution, or a group of related functions such as statistical calculations. An essential feature of CHAMPION is the development of a comprehensive set of precompiled library primitives corresponding to the various Khoros glyphs. In the case of simple functions such as addition, there will be one corresponding precompiled primitive for each glyph. More complex functions such as convolution can be formed as macros of lower-level functions, so that many primitives will correspond to a single Khoros glyph. In the case of glyphs that represent a number of distinct, but related functions, such as statistics, a primitive (or macro) will exist for each of the separate functions. One primitive will be required for each of the distinct functions selected by the user from those available in the corresponding glyph. In a few cases, library primitives were needed to perform hardware-specific functions for which there are no exact equivalents in Khoros. For example, in some cases data must be stored in RAM while results needed for a subsequent operation are computed. A library primitive is thus needed to perform transfers from the FPGAs to RAM. Instances where these hardware specific primitives are needed are generally easily identifiable and can be automatically inserted by CHAMPION. In addition to the functionality needed to perform the operation of the corresponding Khoros glyph, each library primitive also needs additional hardware to implement data transfer and interconnection between glyphs. A set of control lines and a standard data format were developed to allow the hardware glyphs to be connected together in the same manner that the Khoros glyphs can be connected together. Since the primitives are precompiled for a particular FPGA family, they can be characterized in terms of area and delay. CHAMPION will not need to compile the entire application; instead, it must partition the application spatially across multiple FPGAs; partition temporally, if necessary, across multiple configurations of each FPGA; and combine the precompiled primitives with the necessary interconnects.

3. START Algorithm

The application selected for the initial manual mapping is an automatic target recognition algorithm called START (for Simple, Two-criterion, Automatic Recognition of Targets). This algorithm takes as input infrared images and uses statistical methods to find probable targets such as tanks and armored personnel carriers, if present, and draws a box around any that are found. The algorithm was chosen because it is a non-trivial algorithm using common image processing functions. A representative set of input and output images can be seen in Figure 1.

4. Manual Mapping

The START algorithm was first implemented in Khoros; the resultant workspace can be seen in Figure 2. Note that this figure shows only the topmost level of the application; much of the complexity is in lower levels, represented on the top layer only as single procedure glyphs. Library primitives for all of the glyphs used in the application were created in VHDL and then compiled for the specific FPGAs used, the Xilinx XC4000XL series. A value for size in configurable

logic blocks (CLBs) and delay was obtained for each primitive. The total number of CLBs needed for the application was then determined by adding up the CLBs used by each primitive required. The target architecture for the initial manual mapping was an Annapolis Micro Systems Wildforce-XL board. This board contains four Xilinx XC4013XL FPGAs, called PE1 through PE4, used as general purpose processing elements, as well as other FPGAs used for interfacing and control and to implement programmable interconnects between the FPGAs. The four XC4013XL FPGAs provide 2,880 CLBs [3]. As the application required more CLBs than were available on the Wildforce board, more than one configuration of the board was necessary and so the application needed to be partitioned temporally as well as spatially. Manual partitioning was accomplished by inserting successive primitives in the datapath into successive FPGAs, until the number of CLBs needed exceeded the number available on the FPGA. Then the primitive was put into the next available FPGA and the process continued until the partitioning was complete. The temporal partitioning was accomplished by considering the second configuration to be a second set of identical FPGAs, with the addition of primitives necessary to store intermediate results while the board was being reconfigured. While more efficient packing of the application into the FPGAs could have been done, this would have necessitated more cuts in the dataflow graph and thus required more interconnects between FPGAs, increasing delays and therefore slowing execution of the application. The partitioning of the dataflow graph for the START algorithm onto the target hardware is shown in Figures 3 and 4. The numbers next to each operator show how CLBs it uses and the larger numbers show the number of CLBs used in each of the FPGAs.

5. Conclusions

Based on the known delays for the primitives and the characteristics of the Wildforce board, we estimate that the hardware implementation of the START algorithm should process one image in approximately 750ms. Compared with nearly 40 seconds to process one image in the equivalent Khoros workspace, a speed-up factor of more than 54 appears achievable. CHAMPION will be expected to complete this same mapping automatically, in a few minutes, as opposed to the days necessary to complete the manual mapping. It is expected that the automatic mapping should provide a speedup within 10-20% of that achieved manually, as the same library primitives will be used by both mappings. The speed of CHAMPION will greatly enhance the productivity of the Khoros user and encourage the widespread use of FPGA-based computing machines.

Input FLIR Image Labelled Output Image

Figure 1: Representative Input Image and Resultant Output of the START Algorithm.

Figure 2: Khoros Workspace for START Algorithm

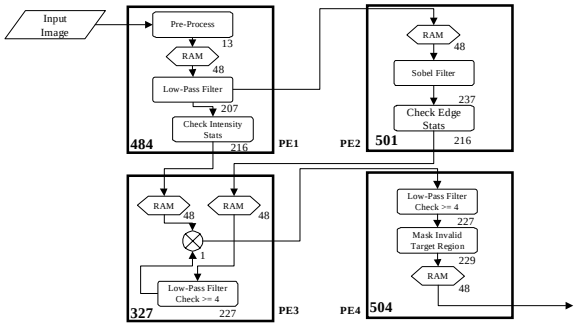

Figure 3: First Board Configuration

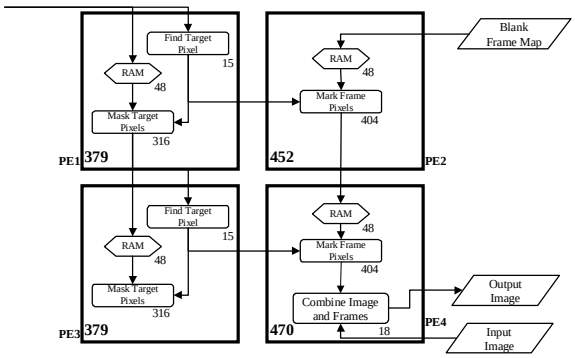

Figure 4: Second Board Configuration

6. Acknowledgements

The authors gratefully acknowledge the support of DARPA grant F33615-97-C-1124.

References:

[1]J. Babb, M. Frank, V. Lee, E. Waingold, R. Barua, M. Taylor, J. Kim, S. Devabhaktuni, and A. Agarwal, "The RAW Benchmark Suite: Computation Structures for General Purpose Computing," *Proc. IEEE Symposium on FPGAs for Custom Computing Machines, 1997.*

[2] N. Ratha, A. Jain, D. Rover, "Convolution on Splash 2," *Proc. IEEE Symposium on FPGAs for Custom Computing Machines, 1995.*

[3] Xilinx, *The Programmable Logic Databook*, 1998.