Architectural Synthesis from Behavioral C Code to Implementation in a Xilinx FPGA

The Frontier Design A\|RT Designer product efficiently maps a software design into a hardware description language implementation suitable for FPGA synthesis.

by Doug Johnson, Business Development Manager, Frontier Design Inc., doug_johnson@frontierd.com

As FPGA densities soar past the million-gate level, there are enough transistors to put entire systems on a single device. To help manage these complex designs and reduce simulation time, large systems are typically prototyped, tested and debugged in software that is usually written using the C programming language. C code can execute 10 to 100 times faster than code written in either VHDL or Verilog HDL, and for complex communications systems C simulation is the often the only alternative.

For hardware implementation, C-language prototypes are usually migrated to an FPGA to enable optimum flexibility and reconfigurability. The Frontier Design A\|RT Designer efficiently and accurately maps a large software design (such as a DSP algorithm written in C code) onto an optimized hardware architecture (FPGA). A\|RT Designer is ideal for creating a processor-like architecture for DSP algorithms such as speech compression, image processing, and for the major blocks in communications systems such as GSM, W-CDMA and IS-136.

Implementing C Code in a Hardware Architecture

Algorithms and systems written in C code are sets of sequential operations that have no regard for resources, parallelism, or timing. The sequential nature of software means that clocking and parallelism are not considerations. To get a piece of software into dedicated hardware, specific hardware resources, such as ALUs, multipliers, adders, RAM, ROM, and registers, must be created and the various operations assigned to them. The operations must then be scheduled so that data dependencies are accounted for. The only way to arrive at the optimum architecture is to look at how the operations are executed on a variety of architectures until the best solution is found.

Until now, there have been no EDA tools that support the exploration of alternative hardware architectures. There was no way to engage in an intermediate architectural exploration between software and silicon. Writing the design in a hardware description language (HDL) requires an explicit architecture because an HDL is a

description of the actual register transfers in a specific architecture. Writing a complex design in a hardware description language is too difficult and time consuming to do more than once.

Design Flow Using the A|RT Tools

The EDA software tools from Frontier Design bridge the gap between the complex system as described in C code and it's final implementation in optimum hardware architecture. Frontier Design has applied its 15 years of experience in transforming DSP algorithms (written in C code) into working silicon, to create a methodology that is called "Algorithm to Register Transfer," or A|RT.

The focal point of the design flow shown in Figure 1 is the A|RT Designer tool that takes the C code and transforms it into synthesizable VHDL and Verilog HDL that is suitable for driving FPGA synthesis tools. A|RT Designer automatically synthesizes hardware architectures directly from behavioral-level C-language programs and

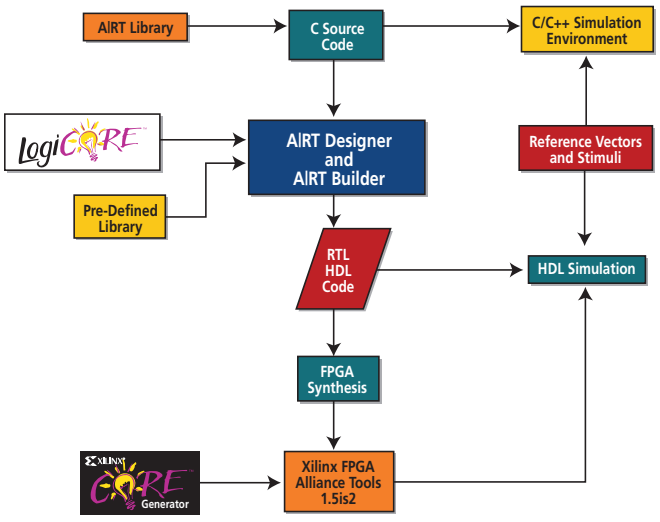


Figure 1 - A|RT design flow.

then generates a register-transfer-level HDL description.

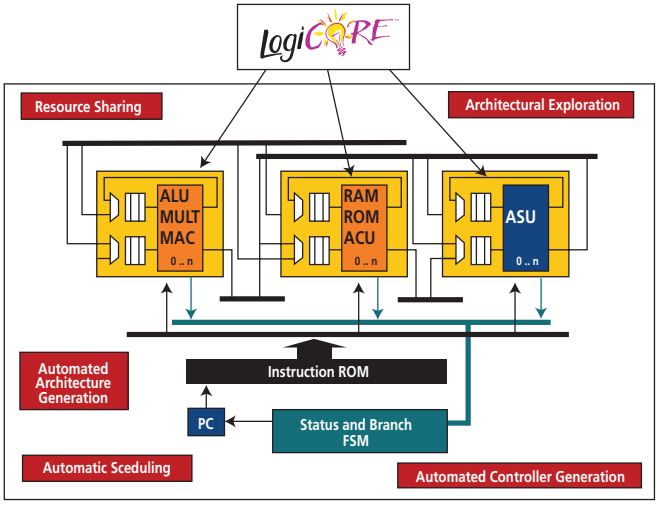
With A|RT Designer, you can interactively add or remove resources, re-assign operations or alter their scheduling, and then analyze the resulting performance characteristics. A|RT Designer is not constrained by parameters such as clock rate or silicon area, so multiple architectures can be explored very quickly. A|RT Designer automatically generates a reconfigurable datapath architecture, plus a VLIW (Very Large Instruction Word) processor and microcode to direct and schedule operations.

Interactive Architectural Synthesis

Although A|RT Designer can be used to automatically synthesize a push-button architecture with minimal intervention by the user, it has been designed with an interactive paradigm from the beginning. A wide variety of reports are available to help you analyze the characteristics of the design. Based on the design constraints, you have a variety of optimization options, plus a variety of directives that you can use to refine the architecture. Design iterations can take as little as a few minutes, so you have the freedom to explore multiple processor architectures very quickly.

Using A|RT to Target Optimum FPGA Implementation

A|RT Designer converts the C behavioral model of an algorithm into an RTL netlist. The resulting hardware architecture consists of a set of data path blocks that are controlled by a microcode-based controller. The datapath blocks are funda-



ASU=> Application Specific Unit
An ASU is a user defined processing block that can perform an operation such as an FFT butterfly

Figure 2 - A|RT designer processor architecture.

mental DSP building blocks such as multipliers, ALUs, register files, RAM, and ROM. Figure 2 is a simplified illustration of the processor architecture used by A|RT Designer.

One of the most powerful capabilities of the A|RT methodology is the ability to target predefined, optimized components in the C code, such as those created by the Xilinx Core Generator and those in the Xilinx LogiCORE library. By exploiting these predefined building blocks, a highly optimum, silicon-efficient implementation of the architecture is possible. For each of the LogiCORE blocks, it suffices to create a simple model file for the resource, containing information on the I/O pins, the cycle-level timing, the instructions available on the resource, and the C functions that can be mapped to those instructions. In addition, A|RT Designer performs resource sharing. This means that the tool can reuse the same resource within several clock cycles. For implementation in Xilinx FPGAs, the LogiCORE blocks can be resources A|RT Designer uses to build the datapath. The datapath will optimize silicon area and achieve highest performance since they are already optimized for the FPGA architecture.

Conclusion

The benefits to the communications system engineer and DSP algorithm designer and hardware engineer are enormous because the LogiCORE and Core Generator blocks are now usable by both the system engineer and the hardware engineer. The system engineer can now incorporate pre-defined hardware functionality directly in C code. The hardware engineer gets a VHDL or Verilog HDL model from A|RT Designer that directly instantiates area and performance-optimized LogiCORE and Core Generator DSP building blocks.

Also, a DSP algorithm typically requires extensive memory such as register files or block RAM and the tools can be directed to target the block RAM capabilities of the Spartan and Virtex devices extending the efficiency of the target FGPA.

A|RT Designer is supported for PC Windows/NT 4.0, for HP-UX 10.20, and for Sun/Solaris 2.7. For more information, please see the Frontier Design website at www.frontierd.com.