Automatic Rapid Prototyping of Semi-Custom VLSI Circuits Using Actel FPGAs

Jae-tack Yoo Erik Brunvand Kent F. Smith

Department of Computer Science, University of Utah, Salt Lake City, UT 84112

Abstract : *We describe a technique for translating semi-custom VLSI circuits automatically into field programmable gate arrays (FPGAs) for rapid prototyping to develop a system. Using an array multiplier as an example of this translation, the VLSI circuits are designed using a cell-matrix based environment. The multiplier is implemented in CMOS in both synchronous and asynchronous pipelined versions, and translated into Actel FPGAs. All test chips were found to be fully functional, and the translation efficiency in terms of chip speed and area is shown.*

1 Introduction

Recent developments in circuit technology have resulted in a wide variety of choices for building application specific digital hardware systems. The design space ranges from custom and semi-custom integrated circuit design, to mask-programmed gate arrays, to field programmable devices that can be programmed in minutes.

We describe a technique that allows custom designed VLSI circuits to be moved quickly and inexpensively to FPGAs. Ideally, the designer would like to try out a design using inexpensive and quickly programmed devices like FPGAs, and when the design is working correctly, recast that design into a faster technology such as custom CMOS. Unfortunately, this recasting and remapping to a custom technology is typically not easy and it can take a great deal of time. Our method involves designing the custom circuit first in one step, including physical layout and schematic capture [1, 2], and then translating that custom design automatically into an FPGA for testing. If the FPGA circuit is correct, no additional work is needed. Otherwise, changes are made to the custom circuit, and a new FPGA circuit will be automatically generated for testing. If there is an urgent need for the circuit, the FPGA version may be used while the CMOS chip is being fabricated (This takes around 8 weeks through the MOSIS facility.) In particular, we use the Physical Placement of Logic (PPL) design system [1, 2] to design the custom CMOS circuits, and map those designs automatically into Actel FPGAs [3] for prototyping and testing.

Our goal was to develop an automatic circuit translator for translating a PPL circuit into an Actel FPGA circuit. In this paper we describe that automatic translator based on gate-by-gate/cell-by-cell translation.

2 Design Environments

2.1 PPL Design Environment

PPL is a cell-based design environment which does simultaneous schematic capture and physical placement of the cells. It has proven to be an effective design tool for complex VLSI circuits and can produce circuits whose density and speed are comparable to that of full custom design [1, 2].

The PPL design environment includes a wide variety of tools for the design and evaluation of custom integrated circuits. Circuits are designed using a graphical editor called *ACME* [2]. Simulation is accomplished using a switch-level simulator, *simppl*. The PPL environment also contains a circuit extractor and electrical rules checker, *simpplex*, and a netlist generator, *spplice*, that operate on the circuit database [4], which can produce *hspice* netlists to facilitate the simulation of pieces of the circuit at the transistor level.

2.2 Actel Design Environment

In contrast to the custom circuits designed using PPL, Actel FPGAs can be "fabricated" in minutes. The Actel product is a field programmable gate array (FPGA) implemented in 2.0μ or 1.2μ CMOS. The chip is arranged with rows of logic cells interspersed with routing channels. Logic modules are connected to perform the desired function by programming selected anti-fuses.

The Actel ACT-1 architecture was chosen as the target for our translation. The basic ACT-1 logic module is shown in Figure 1. This module is the basis for all the macros in the Actel cell library as well as the special macros we have defined for this translation process. Actel Chips contain these logic modules.

Schematic capture design and simulation tools from Viewlogic [5] produce netlists which will be used by the ALS tools [6]. This ALS program performs pin assignment,

placement and routing of the circuit onto the Actel chip. It also performs static timing analysis and provides delay back-annotation to the *viewsim* simulation program.

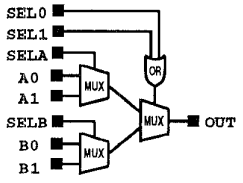

Figure 1: Actel Basic Logic Module

2.3 Resulting Environment

An overview of the translator environment is shown in Figure 2. In contrast with other schematic capture translation systems, our procedure starts with schematic capture and physical design in one step. Circuits are designed as custom circuits using *ACME* and simulated using *simppl*. These circuits are converted, using our circuit translation rules, into Viewlogic netlist format, and from there an Actel FPGA is generated. Simulation of the Actel chip is done from the Viewlogic netlist using the *viewsim* simulator.

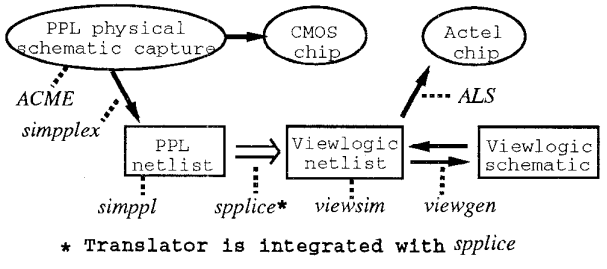

Figure 2: Overview of the translator environment

3 Example: 4-bit Multipliers

To demonstrate the translation process we choose an array multiplier built using a series of carry-save adders (CSA) with a carry-propagate adder in the final stage to resolve the carry-save result as shown in Figure 3. The circuit was designed to fit on a MOSIS Tiny Chip. Two Tiny Chip versions were designed to serve as the source for the translation. These are:

Synchronous-MOSIS (SMO) A version of the pipelined multiplier controlled in a standard synchronous fashion.

Asynchronous MOSIS (AMO) Another version of the multiplier which is controlled asynchronously in the style of micropipelines [7].

4 The Translator

Because PPL circuits are built using a set of fine-grained library cells, it is a somewhat straightforward matter to define cell-by-cell translation rules since the PPL cells have a direct equivalent in the Actel cell library in many cases. For those cells that do not have a direct mapping, we have generated new Actel cells using the basic Actel macro shown in Figure 1. Conversion is achieved by using syntax-directed translation from one cell set to the other.

The process of translation begins with the PPL circuit extractor *simpplex*. PPL circuits may use fine-grained cells to build distributed gate structures that can look like PLA rows and columns, although in a much less constrained form than in a PLA. These structures are collected into standard gates by *simpplex* and entered into a design database representing the circuit. Once in this form, translations of standard INVERTER, AND, NAND, OR, and NOR gates, for example, are straightforward.

Our translator was plugged into the netlist translator *spplice*. This translator generates a *Viewlogic netlist output* from the database description. If desired, this netlist can be converted into a circuit schematic using the *viewgen* program from Viewlogic. This provides flexibility for a designer to modify the netlist prior to programming the Actel part.

Followings show example cell translations.

Full Adder Full adders are represented in PPL as single cells. Because a cell with corresponding logic does not exist in the Actel library, a new cell was designed for the translator as shown in Figure 4. Two different versions of the full adder cell are considered: (1) the Actel FA1B cell with an extra inverter (3 module-delay for sum-out) and (2) an inverter and two Actel basic macros (2 module-delay for sum-out). Either adder cell may be used in the translation, although future versions of the translator may prefer one over the other depending on the structure of the surrounding circuit.

Clock Driver PPL clock driver cells generate two-phase non-overlapping clock signals. These clocks are used by a variety of cells that require this clock. In contrast, Actel cells that re-

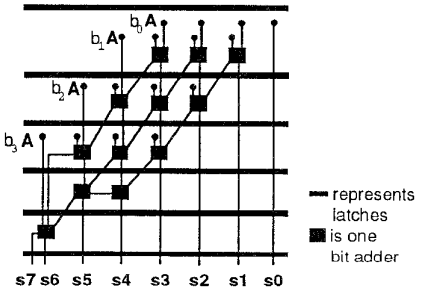

Figure 3: A four-bit pipelined array multiplier

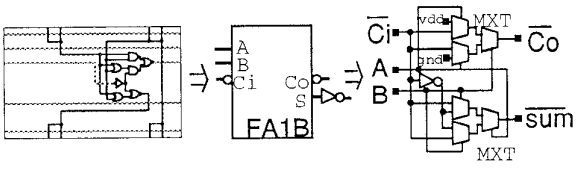

Figure 4: Translation of a full adder

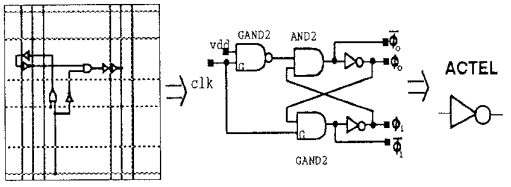

Figure 5: Translation of a clock cell

quire a clock use only a single phase clock. Actel ACT-1 FPGAs provide a single dedicated low-skew clock line across the chip. In an Actel design, this would normally be used to distribute the system clock. However, for the translation, we cannot be sure that the PPL two-phase clock is used only for flip flops. Instead, we must assume that the different phases might be used in different ways. So, the translation implements a clock generator similar to the one in the PPL cell set as shown in Figure 5.

Latch cells Two different types of latch cells are used in the SMO and AMO PPL chips. A static inverting transparent latch is used in the synchronous SMO chip. This is translated directly to an Actel latch with equivalent characteristics. The asynchronous AMO chip uses a dual-edge-triggered latch that allows data to be captured on both edges of a clocking signal. This latch is implemented in Actel using three basic macros as shown in Figure 6.

C-elements [8] C-elements are gates used frequently in asynchronous designs like the AMO chip. This cell from the PPL cell set is implemented using three Actel macros as shown in Figure 7. Because the PPL cell offers both inverted and non-inverted outputs from the C-element, the Actel version also provides both output senses. However, if the inverted output is not used in the circuit, the optimization phase of the ALS

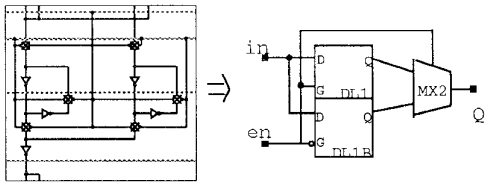

Figure 6: Asynchronous static latch translation

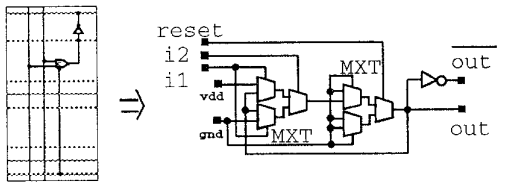

Figure 7: C-elment translation

place and route software will delete the dangling and extra inverter from the FPGA circuit.

5 High-Level Translation Issues

The translations could be improved greatly by applying some higher level translation rules. These rules could improve the resulting FPGA circuit by eliminating several of the inefficiencies mentioned previously. The following are examples of the type of higher level rules.

Inverter elimination Using the PPL cell library generally results in using standard NAND and NOR gates for random logic in the circuit. The Actel library, in contrast, offers many alternate gates with inversions on selected inputs or the output of the gate. These alternate gates could be used to provide equivalent logic functions rather than exact one-for-one mapping. For example, inverters in the PPL chip can, in many cases, be subsumed into the following gate by using a version of that gate with an inverted input.

Flip-flops The dual-edge-triggered latches used by the asynchronous AMO chip are currently translated into a three-module circuit shown in Figure 6. With only a slight modification to the circuit, these asynchronous latches can be implemented in a form that uses only a single Actel macro. A synchronous flip-flop can be decomposed into AND/OR gates and two static latches, which can be individually translated.

Clocking mechanisms The clock generator circuit used in PPL can, as long as it is used only as the clock input to sequential elements, be translated to use the dedicated clock line on the Actel part. Since the Actel latches use the single phase clock, this could make the FPGA circuit much more efficient.

6 Example: Translated Actel Versions

Using the two PPL circuits SMO and AMO as a starting point, four equivalent Actel circuits were generated. Two were done automatically using the syntax-directed cell-by-cell replacement rules, and two translations were done by hand using more efficient high-level translations to check the effect of a more sophisticated translation on the resulting FPGA circuits.

The Actel chips are:

- Synchronous Rule Translation (**SRT**)

Table 1: Multiplier Performance Measurements

	sync. chips		
	SMO	SRT	SHT
Max Clock	24nS	56nS	32nS
Min Latency	120nS	280nS	160nS
	async. chips		
	AMO	ART	AHT
Slowest Stage	24nS	88nS	60nS
Latency Time	72nS	348nS	216nS

- Asynchronous Rule Translation (**ART**)
- Synchronous Hand Translation (**SHT**)
- Asynchronous Hand Translation (**AHT**)

7 Test Results

The two MOSIS Tiny Chips, SMO and AMO, were fabricated in the MOSIS 2-μ CMOS technology. The Actel chips were programmed into A1010A or A1020A Actel parts depending on their size. All chips were functional and were tested for speed on a Tektronics LV500 tester.

Table 1 shows the results. The SRT Actel design using the simplified syntax-directed rules was a factor of 2.3 slower than the PPL chip, and the SHT hand translated version was only a factor of 1.3 slower. For the asynchronous design, the slowest stage reflects the delay encountered in the logic between any two adjacent pipeline stages. The slowdown for the latency is caused in large part by inefficient translation of request/acknowledge control circuitry.

Table 2 shows the area measurements for the example chips. The ratio of the rule-translated Actel chips to the hand-translated chips shows that the high-level translation rules improve the size of the resulting chips by a significant factor as well as the speed. Although it is difficult to compare chip capacities that use different technologies, this also indicates that a PPL tiny chip provides approximately the same functionality as an Actel 1010A or 1020A chip. This will, of course, depend heavily on the structure of the circuit being implemented.

8 Conclusions

We have described a procedure to allow custom VLSI designs to be prototyped quickly and inexpensively using FPGAs. Specifically, we have shown a technique for translating custom PPL chips into Actel FPGAs. By designing the custom chip first as the source of the translation, and mapping that custom chip automatically to an FPGA, we have avoided the problem of technology mapping that is often encountered when trying to design the prototype first and map to a custom technology only after the prototype is found satisfactory.

Table 2: Area Measurements for Multiplier Example

	SMO	AMO
Chip Type	MOSIS tiny chip	
Max Cells	2618 unit cells	
Max Area	in 2.2mm x 2.2 mm	
Used Area	493 cells(logic) 949 cells(wiring)	700 cells(logic) 1189 cells(wiring)

	SRT	SHT	ART	AHT
Actel Chip	1010A	1010A	1020A	1010A
Modules/Chip	295	295	547	295
Modules Used	250	187	385	200
Area Ratio	SRT/SHT = 1.34 , ART/AHT = 1.93			

Using PPL synchronous and asynchronous pipeline multiplier chips as examples, we have demonstrated both an automatic syntax-directed translation and a more sophisticated translation that uses some higher level translation rules. Six chips were produced and all were found to be fully functional. Furthermore, the FPGA circuits were found to be faithful copies of the original PPL circuits. Testing results for these examples show that the FPGA prototype performs at between 43% and 75% of the speed of the synchronous CMOS custom chip and from 21% to 40% of the asynchronous custom chip's speed.

References

[1] Kent F. Smith and Jun Gu. "A Structured Approach for VLSI Circuit Design," *IEEE Computer*, Vol 22, No. 11, November, 1989

[2] Tony M. Carter, Kent F. Smith, Steven R. Jacobs, and Richard M. Neff, "Cell Matrix Methodologies for Integrated Circuit Design," *Integration, The VLSI Journal*, July, 1989

[3] Actel Corporation. *ACT Family Field Programmable Gate Array Databook*, April 1992

[4] Steven R. Jacobs. *PPL Database Reference Manual*, January, 1993

[5] Viewlogic Systems Inc. *Schematic Design User's Guide*, May, 1991

[6] Actel Corporation. *Action Logic System(ALS) Release 2.1 User's Guide*, August 1991

[7] Ivan E. Sutherland. "Micropipelines," *Communications of the ACM*, June 1989

[8] Erik Brunvand. "A cell set for self-timed design using Actel FPGAs," *Tech. report UUCS-91-013*, The University of Utah, 1991