2005

Use of Advanced Synthesis Tools in Teaching VLSI Design

Kevin Berwick
Technological University Dublin, kevin.berwick@tudublin.ie

Dominic Nardone
Technological University Dublin

Use of Advanced Synthesis Tools in Teaching VLSI Design

Kevin Berwick & Domnick Nardone
Dept. of Electronics and Communications Engineering
Dublin Institute of Technology

Field Programmable Gate Array (FPGA) prototyping systems are used extensively in Digital Design Courses, allowing students to experience the complete design process from initial circuit specification to physical implementation on a development board. Using FPGAs significantly reduces the time and cost of prototyping even large designs.

Here in the Department of Electronics and Communications Engineering at the Dublin Institute of Technology, we teach a variety of courses in Digital Design at Diploma and Degree level.

This discussion is based on our experiences on the second Year High Level Design course at Diploma level. The Course begins with an introduction to High Level Design and implementation methodologies. The basic lexical elements of combinational and sequential VHDL are then presented, together with explanations of simulation, synthesis, pipelining and hierarchy. The Course concludes with Finite State Machine design methods. A variety of examples of circuit descriptions and testbenches are presented as examples throughout the Course in order to reinforce the design concepts and to provide code templates to act as a starting point for the students own designs.

Along with the lecture Course there is an associated laboratory program. Students write circuit descriptions and testbenches for simple structures such as Adders and LED decoders. The designs are simulated and synthesized. Finally, the design is implemented using Xilinx Integrated Software Environment (ISE) tools and the bitstream is used to program a Digilab D2E prototyping board based on a Xilinx Spartan 2E FPGA.A Digilab DI01 I/O board connected to the prototyping board is used to apply signals to the FPGA and the outputs can be observed on LEDs and 7 segment displays.

In the past, we used the Xilinx ISE tools for developing our FPGA designs with ModelSim used as the Simulation tool. Recently, we investigated the use of a variety of 3rd party synthesis tools, leading us to choose the Synplify Pro tool from Synplicity as our preferred tool for FPGA synthesis.

Although Quality of Results (QOR) and the short run time of this tool are attractive, they are not critical factors in the small designs typically encountered in our Digital Design Courses. However, hardware visualization is very important in Digital Design education in order to reinforce the differences between writing synthesizable HDL code and writing software in a high level language such as C. The Synplify Pro software offers a number of unique advantages in this area and these were critical in informing our choice to adopt the tool for teaching purposes.

In order to illustrate these advantages, consider a typical introductory design that we ask our students to do. This consists of a simple Finite State Machine designed to recognize the sequence of bits 111.When this sequence occurs, an LED on the Digilab board is to light to indicate that the sequence has been recognized.

After the students write the circuit description in VHDL, they construct a suitable testbench. When the circuit simulates correctly in ModelSim, the student synthesizes the design in the Synplify Pro software. Once synthesis is complete, the HDL Analyst feature in the Synplify Pro tool can be used to generate an RTL block diagram from the HDL code, see Figure 1. The ability to cross probe from the HDL code to the RTL view and from the RTL view back to the code reinforces the coupling between code and hardware. It also allows us to emphasize the fact that VHDL coding is NOT the same as writing software.

Note, from Figure 1, that the VHDL code describing the Finite State Machine is shown as a high level block. 'Pushing' into the block using the tool invokes the FSM viewer, as shown in Figure 2.

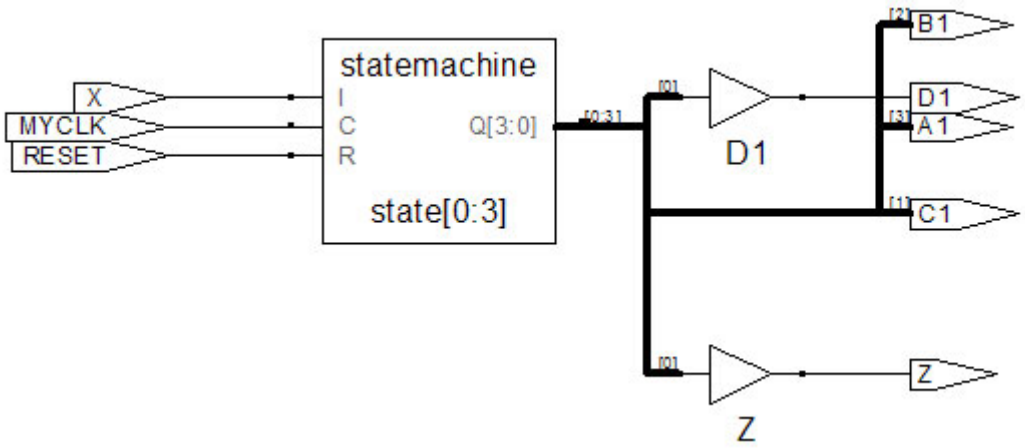

Figure 1. RTL view of Sequence recognition Circuit from Synplify Pro –HDL Analyst

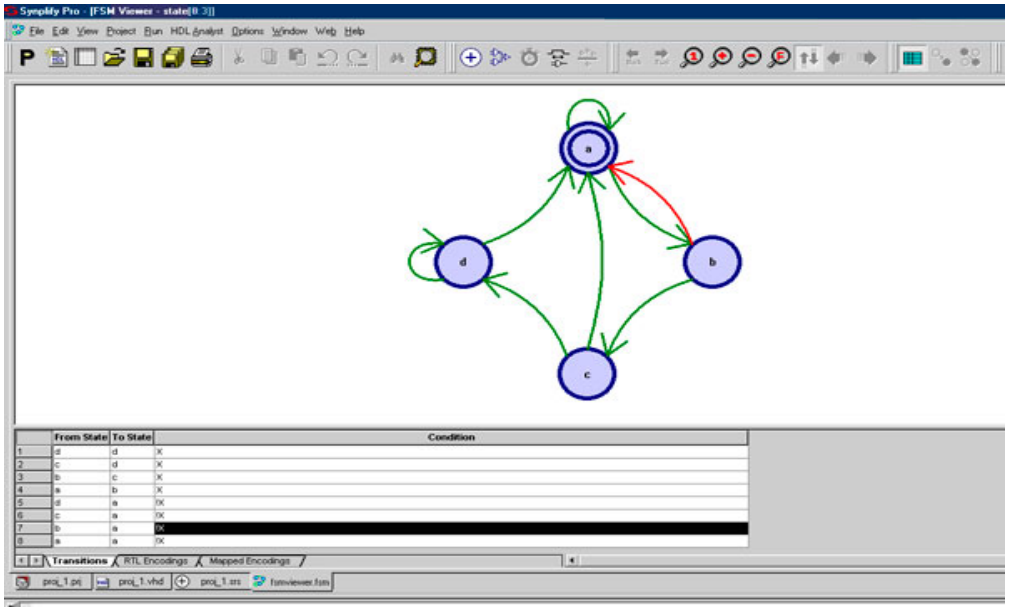

Figure 2. FSM viewer and VHDL code. Note that the highlighted red arrow is associated with the transition from state b to a as indicated in the transition table.

The FSM viewer presents a transition bubble diagram and a table for the encodings and transitions between the states. Here, it is possible to view the 'from' and 'to' states, and

the conditions for each transition from state to state. It is also possible to see the correspondence between the states and the FSM registers in the RTL view. After mapping, the correspondence between the states and the registers within the FPGA are viewable in the Technology View. Each view allows the user to click a bubble and the corresponding state information for that state will be highlighted. In addition, clicking on, for example, State C in this view of the circuit will highlight the VHDL code corresponding to that state, illustrating the crossprobing capabilities of the Synplify Pro tool.

The FSM Viewer is used by the student in order to check that the required numbers of states are present and that all transitions between states are listed correctly. The FSM explorer can be invoked during the synthesis run allowing all the encoding options to be explored. The FSM Explorer is a tool that automatically explores different encoding styles for state machines, and picks the style best suited to the design. According to the synthesis log, in this case a sequential design was chosen by the tool to implement this 4 state machine with only 2 Flip Flops. This is confirmed by looking at the Technology View of the circuit using the HDL Analyst in the Synplify Pro tool, see Figure 3.

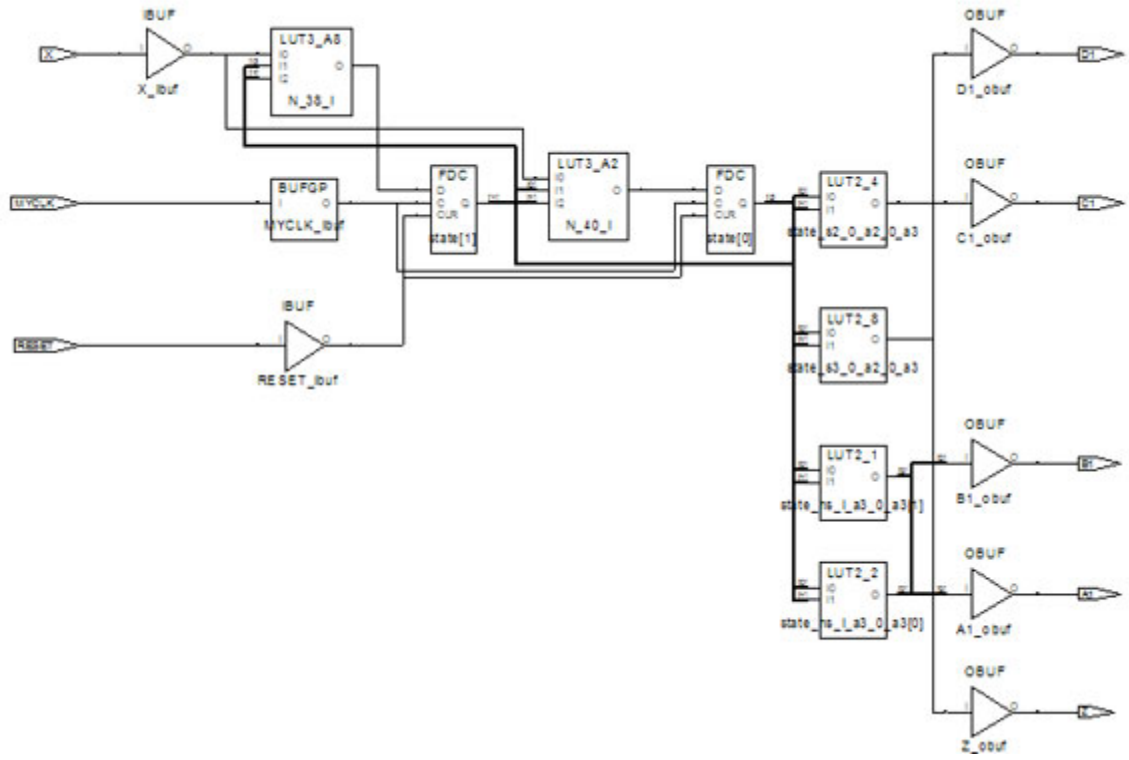

Figure 3 Technology view of Sequence recognition Circuit from the Synplify Pro HDL Analyst.

Finally, the design is implemented using the Xilinx ISE tools and the bitstream is used to program a Digilab D2E Spartan 2E based prototyping board. A DI01 I/O board connected to the prototyping board is used to apply signals to the chip and the output is observed on LEDs. In our experience, the Synplify Pro tool is extremely easy to use and offers an uncluttered ‘front end’. Our students are typically using the tool minutes after a short introduction. When we adopted the Synplify Pro tool, we were already using the Xilinx ISE environment for our designs. It was very easy to set the Synplify Pro

software as the preferred Synthesis tool within the Xilinx ISE Project Manager, meaning that the transition to the Synplify Pro tool was almost effortless. Finally, Synplicity offer very attractive pricing to Educational Institutions on all their tools via their University program. We intend to explore the capabilities of the tool further in the near future applying it to some larger designs as part of our research activities here in the Dublin Institute of Technology. The QOR and runtime advantages of the Tool will be more important in this case. In summary, we find that the Synplify Pro tool offers a number of compelling advantages over competing FPGA synthesis tools and has enhanced our teaching of Digital Design significantly.