**Lab 1**

**Counter Tutorial**

**Prepared by:**

**Steven Blair and Daniel Olsen**

**Prepared for:**

**Dr. Wang**

**ECE 428/593B-001**

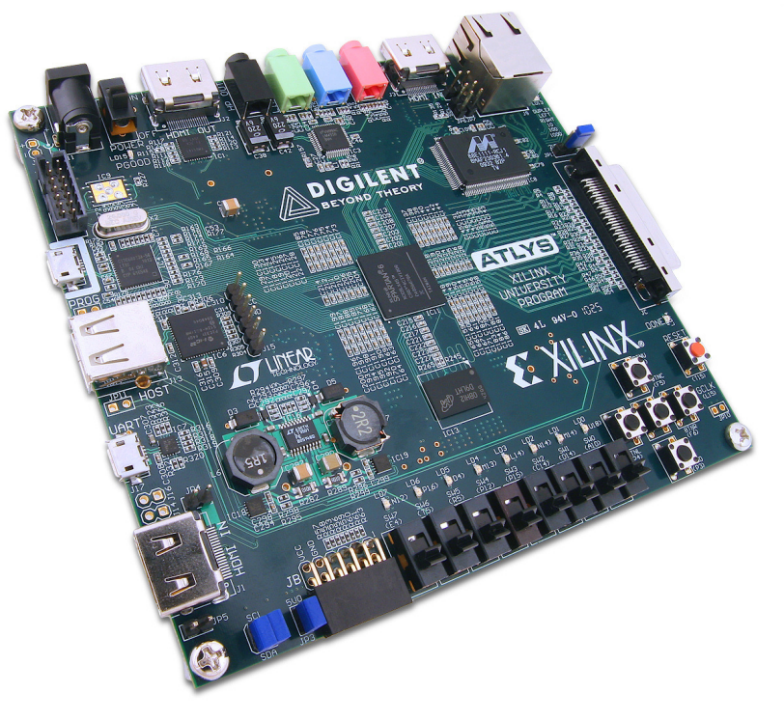
**Programmable ASIC Design**

**February 19, 2015**

# Purpose

The purpose of this lab is to introduce students to the three tools that will be used extensively in this course: the Atlys Development Board, a high-performance development board with FPGA; the Adept FPGA Programmer, a custom interface that programs a bitstream onto the development board; and the Xilinx ISE Design Suite, a toolchain that assists in the capture and implementation of a digital circuit as a bitstream used by the programmer. By implementing, simulating, and testing a simple 3-bit counter with reset using the buttons, lights, and FPGA available on the development board, students are to become comfortable with the general design flow for an FPGA.

Figure 1. Digilent Atlys FPGA board



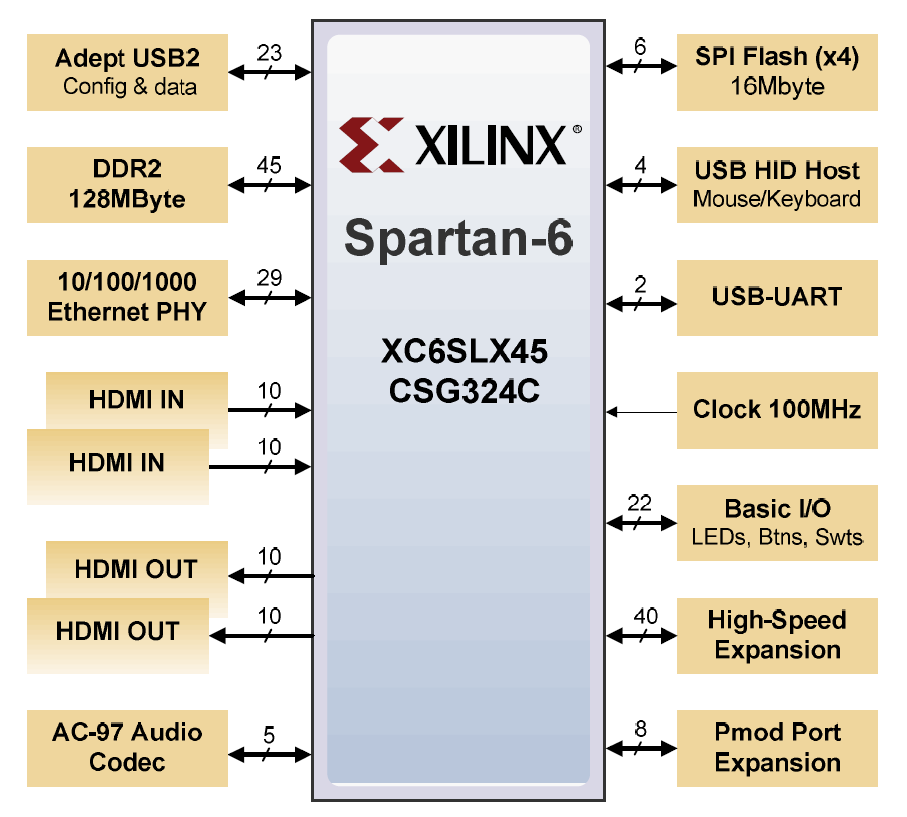
# Background

The Atlys Development Board is a complete digital circuit development platform based on the Xilinx Spartan-6 LX45 FPGA. The FPGA includes 6,822 slices that each contain four 6-input lookup tables and eight D flip-flops. The development board includes the following features:

* 100 MHz CMOS Oscillator
* Four PLLs
* Eight DCMs
* 128 Megabytes of DDR2 RAM
* 16 Megabytes of SPI Flash
* 2.1 Megabits of Fast Block RAM
* 10/100/1000 Ethernet Interface
* USB-UART/USB-HID Interfaces
* Two Input and Two Output HDMI Interfaces
* Line-In, Line-Out, and Microphone Ports
* Eight LEDs, Six Buttons, and Eight Slide Switches

A schematic representation of the resources available to the FPGA is included in Figure 2

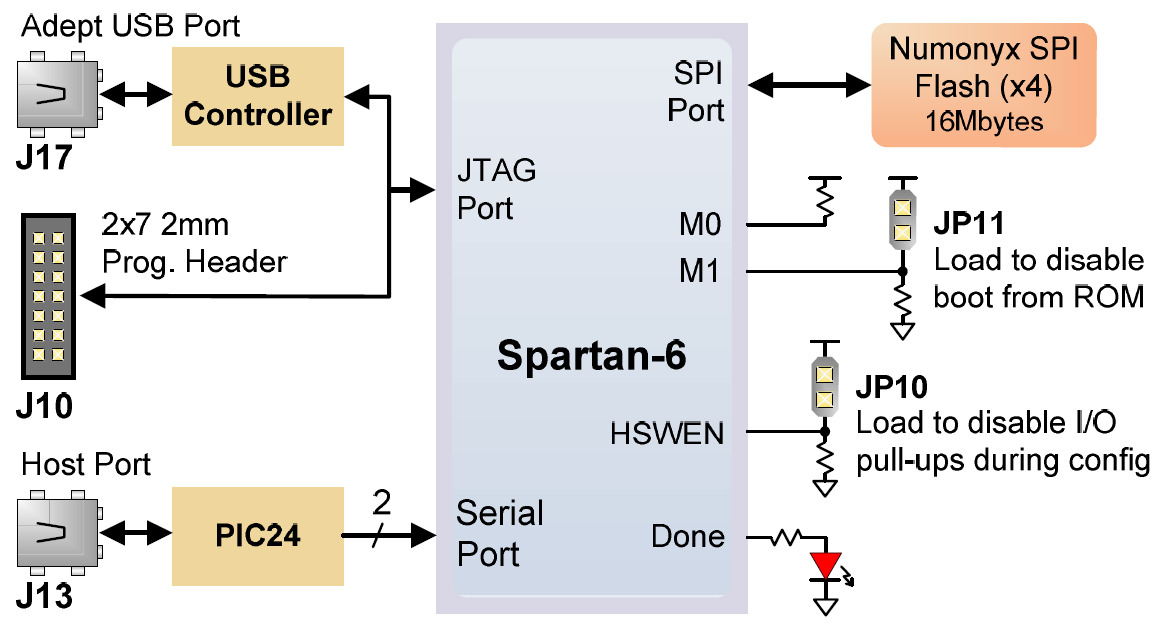
Figure 2. Atlys FPGA board architecture [1]



The development board implements Digilent's Adept USB2 system which offers programming and monitoring through a custom interface and supports the Xilinx ISE Toolchain. This is just one of three ways that the FPGA on the Atlys Development Board can be programmed; the FPGA can also be programmed using the JTAG port at any time power is supplied to the board or using a USB memory stick attached to the USB-HID port at power-up.

A schematic representation of the available programming options is included in Figure 3. The Adept system is supported through port J17; the JTAG system is supported through port J10; and the USB memory stick is supported through port J13. If the jumper JP11 is loaded at boot up, the FPGA will be configurable using any of the three programming methods; if the jumper is not loaded, the FPGA will automatically configure itself from onboard RAM.

Figure 3. Atlys FPGA board configuration circuit [1]



# Procedure

In the main part of this lab, students were asked to follow a standard design flow to implement a VHDL description of a 3-bit counter on the Atlys Development Board. Students followed a tutorial provided with the laboratory manual that included the following steps:

1. In the Xilinx ISE Design Suite, create a new project using VHDL source files that will be implemented on the Spartan 6 family with device XC6SLX45 and package CSG324.
2. Create a new VHDL source file using the VHDL source code provided in the laboratory manual. Run a syntax check to ensure later steps will execute properly.
3. Create a new VHDL testbench that is capable of testing the major functionality of the circuit described in the VHDL source file. Run a syntax check to ensure later simulations will execute properly.
4. Run the ISim tool. Using the VHDL testbench, run a functional simulation of the VHDL source file to verify functional operation of the circuit
5. Run the PlanAhead tool. Use the software to generate a user constraint file (UCF) that defines the relationship between input and output signals from the VHDL source file and the physical interface provided by the development board. Set the Site value for the ‘Light[0]’, ‘Light[1]’, and ‘Light[2]’ signals to U18, M14 and N14, respectively; set the Site values for the ‘reset’ and ‘countup’ signals to N4 and P3, respectively.
6. Synthesize the VHDL source file and the UCF file using the Xilinx Synthesis Tool.
7. Run the ISim tool. Using the VHDL testbench, use the software to run a post-synthesis simulation of the VHDL source file to verify functional operation and ensure timing requirements.
8. Implement the design using the translation, map, and place and route tools available in Xilinx ISE.
9. Run the ISim tool. Using the VHDL testbench, use the software to run a post-place-and-route simulation of the VHDL source file to verify functional operation and ensure timing requirements.
10. Generate the bitstream used to program the FPGA using the tools available in Xilinx ISE.
11. Run the Adept FPGA Programmer. Select the bitstream generated in the previous step and program the FPGA using the defaults automatically detected for the Atlys Development Board.
12. Verify correct operation of the circuit on the Atlys Development Board.

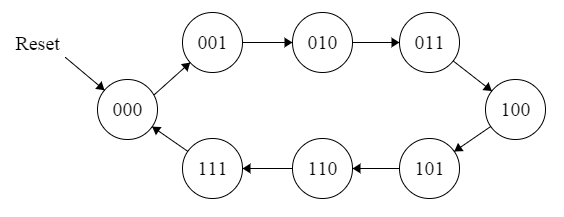
Following the procedure, it is suggested that students examine the schematic and timing facilities provided after the synthesis and implementation steps in the design flow.

# Discussion

## High-Level Design

The VHDL code describes a finite state machine synchronized by the 'countup' signal with the state variables stored in a 3-bit variable which is copied to a 3-bit bus and with the state transitions defined by the state transition diagram in Figure 4; an asynchronous 'reset' signal is also described in order to put the finite state machine into a known state. By connecting the 'countup' signal and the 'reset' signal to a button on the development board and by connecting each wire in the 3-bit bus to an LED on the development board, the circuit implements a simple counter that can be operated and read from the physical interface of the development board.

Figure : Finite State Machine for Counter



## High-Level Verification

The VHDL code was synthesized, implemented, and programmed onto the FPGA on the Atlys Development Board. It was demonstrated that the P3 button properly incremented the counter and the N4 button properly reset the counter; the value of the counter was displayed via the LEDs defined in the UCF.

## Simulation

To test the counter during all phases of design and implementation, a VHDL testbench was composed. The testbench resets the circuit for one clock cycle, operates the circuit for eight clock cycles, and resets the circuit indefinitely which allows for functional and timing verification of the transitions in the finite state machine and for functional verification of the reset.

Pre-synthesis simulation demonstrated a one-to-one correspondence between the simulated response and expected response for the circuit; the circuit remained in the reset state for one cycle before counting out each transition for eight cycles before remaining in the reset state indefinitely. Post-synthesis simulation demonstrated a similar response except that state transitions occurred an almost imperceptible 100 ps after the arrival of the 'countup' signal. Post-place-and-route simulation demonstrated a similar functional response to the pre-synthesis and post-synthesis simulations but demonstrated a different timing response. Although there was not enough delay present in the circuit to cause timing violations, there was a perceptible 13.292 ns delay between the arrival of the 'countup' signal and the state transition.

The presence of more delay in each phase of the implementation procedure is attributable to the more accurate component and routing models available for simulation. Pre-synthesis simulation includes no timing information; post-synthesis simulation includes timing information related to components; and post-place-and-route simulation includes timing information related to both components and routes.

## Synthesized Schematic

After synthesis, two schematics were provided by the Xilinx ISE Design Suite in order to illustrate a high-level, specialized view of the circuit and a low-level, generic view of the circuit.

The register transfer level schematic provides a high-level structural description of the circuit. Since the VHDL source file only describes a single architecture with a single behavioral definition, the RTL schematic only shows a single module that corresponds to the behavioral definition.

The technology map schematic provides a low-level structural description of the circuit as it would be implemented using Xilinx primitives. For this schematic, the VHDL source file is synthesized into implementable logic. A global buffer and input buffer are provided for the ‘countup’ and ‘reset’ signals while output buffers are provided for each signal in the ‘Lights’ bus. A 2-input lookup table, a 3-input lookup table, and an inverter are provided to define the next state and next output logic; three D flip-flops are provided to store the current state.

## Implemented Schematic

After implementation, a schematic was provided by the Xilinx FPGA Editor in order to illustrate at a very low-level the specific implementation of the circuit by the FPGA.

The circuit can be largely divided into four blocks: the input pins, the output pins, the control logic, and the clock route. The input pins represent the entry point for information from the buttons and the output pins represent the exit point for signals to the LEDs on the development board. The control logic includes the sequential elements used to store the finite state machine state as well as the combinational logic that determines the next state and next output. The clock route includes the global buffer and clock routing resources used to drive the 'countup' signal from the input pins to the control logic. All blocks were connected to each other via programmable routing resources.

The control logic was implemented by a single slice on the FPGA and can be further broken down into the components used in the slice. A set of programmable state elements were used as D flip-flops to store the state of the finite state machine, and a set of look-up tables were programmed to provide the necessary logic to govern state transitions.

## Static Timing Analysis

A static analysis executed on the design revealed that the longest path for the circuit requires a setup delay of 1.197 ns along the path from the second state flip-flop (FFd2) to the first state flip-flop (FFd1); the path demonstrates a data delay of 1.162 ns through one logic level, a clock skew of 0 ns, and a clock uncertainty of 0.035 ns.

# Extension: Modifying Pin Assignments

## Purpose

In the extension, students were asked to change the pin assignments so that the operation of the circuit could be controlled and verified from different buttons and lights than those indicated in the laboratory manual.

## Procedure

The procedure is largely the same as for the main part of the lab except that the following change is made:

1. Modify the UCF file to change the assignment between singles from the VHDL source file and the physical interface provided by the development board. Set the Site value for the ‘Light[0]’, ‘Light[1]’, and ‘Light[2]’ signals to L14, M13 and D4, respectively. Set the Site values for the ‘reset’ and ‘countup’ signals to N4 and A10, respectively.

## Results

After synthesizing, implementing, and programming the circuit, a test was run on the Atlys Development Board. Actuation of the A10 slide switch caused a transition in the finite state machine as expected; the state of the finite state machine was displayed on the new L14, M13, and D4 LEDs.

Since there should be no major changes in the synthesized or implemented design aside from some differences in routing, the extra results for this test are not included here.

# References

|  |  |
| --- | --- |
| [1] | Digilent, Inc., "Atlys Board Reference Manual," 2 May 2013. [Online]. Available: http://www.digilentinc.com/Data/Products/ATLYS/Atlys\_rm\_V2.pdf. [Accessed 18 February 2015]. |