

CSEE 4280: Lab 3:

Designing the Toy Processor Datapath

Group 8: Habilou Sanwidi, Kingsley Sitz

School of Electrical and Computer Engineering

University of Georgia

[hss67466@uga.edu](mailto:hss67466@uga.edu), KingsleySitz@gmail.com

Contributions: Please list contributions (in estimated percentages) of each member in the following categories.

• Pre-lab design and analysis:

Habilou - 50% Kingsley - 50%

• In-lab module and testbench design

Habilou - 50% Kingsley - 50%

• In-lab testbench simulation and analysis

Habilou - 50% Kingsley - 50%

• In-lab FPGA synthesis and analysis

Habilou - 50% Kingsley - 50%

• Lab report writing

Habilou - 50% Kingsley - 50%

# **ABSTRACT**

In this lab, we are aiming to build a Datapath. The first part of this lab will consist of building an 8-Bit register using eight 1-Bit register (FDRE). Secondly, We will build a counter using a half-adder array.

1. **INTRODUCTION  
     
   //PreLab**
2. **IMPLEMENTATION DETAILS**
   1. **8-Bit Register**

We will build an 8-bit register using eight 1-bit registers. The 1-bit register symbol is called FDRE. The FDRE symbol is made up of a D-flip-flop and a 2-1 mux. When CE is high, it loads the current value of D. If CE is low, D is ignored. If RST is high the FDRE is reset.

**Figure 1:** 8-Bit Register Schematic

A screenshot of a cell phone

Description automatically generated

**Figure 2:** 8-Bit Register Testbench

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

* 1. **Half Adder**

In order to build a counter, we first need to build a half-adder. A half-adder consists of two gates (AND & XOR gates), 2 inputs (Xi & Ci) and two outputs (Cout & S).

**A close up of a map

Description automatically generatedFigure 3:** Half-Adder Schematic

* 1. **Half-Adder Array**

This component will utilize eight 1-Bit registers, one 7-Bit input (HA\_IN (7:0)) and two outputs (HA\_OUT (7:0) & Overflow).

**Figure 4:** Half-Adder Array Schematic

A close up of a map

Description automatically generated

* 1. **Counter**

We can now build a counter. The counter utilizes the following components:   
8-Bit Register  
8-Bit Mux  
Half-Adder Array  
In order to build our counter, we will follow the following block diagram**.**

**A close up of text on a white background

Description automatically generatedFigure 5:** Block Diagram

**A close up of text on a white background

Description automatically generatedFigure 6:** Counter Schematic

**A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedFigure 7:** Counter Testbench

* 1. **Datapath**

Finally, we can use all the components in our directory to build a Datapath. Due to the complexity of the Datapath, we will not stimulate this module.

**Figure 8:** Datapath Schematic

A picture containing text

Description automatically generated

1. **Error Analysis**
   1. **8-Bit Register**

As we can see in figure 9, the register only updates REG\_OUT to the current value of REG\_IN when LOAD is high. RST reinitializes all inputs and outputs. Figure 9 confirms the functionality of our design.  
  
 **Figure 9:** 8-Bit Register Waveform

A screenshot of a video game

Description automatically generated

* 1. **Counter**

By verifying the result obtain in Figure 10 and the table given in the Lab instruction, we can confirm that our counter works perfectly.  
  
  
 **Figure 10:** Counter Waveform

A picture containing sky, wall

Description automatically generated

1. **CONCLUSION**

We were able to build a Datapath using an 8-Bit register, an 8-Bit Mux and an array of eight 1-Bit registers. We successfully created testbenches for each component and obtained waveforms which confirms the functionality of our final module.