Lab 06

Adding Memory and Bootstrapping Circuit

Category	**Ryan Cruz** ryan.cruz25@uga.edu	**Zachary Davis** zachdav@uga.edu
Pre-lab	50	50
In-lab Module & Testbench Design	50	50
In-lab Testbench Sim. & Analysis	50	50
In-lab FPGA Synthesis & Analysis	50	50
Lab Report Writing	50	50

March 1, 2018

Contents

1 Lab Purpose **3**

2 Implementation Details **3**

2.1 Prelab 3

2.2 Circuit Implementation in Schematic Editor 4

2.3 Simulation 5

3 Experimental Results **10**

4 Significance **11**

5 Comments/Suggestions **11**

1 Lab Purpose

Entering this lab, we now have all of the components to complete the design of the Toy Processor. The two main modules that have resulted from our previous labs are the datapath and the controller. In this lab, we will connect those two components and then test its behavior with a simple program by encoding certain instructions and executing them in a test fixture to view data inputs and their respective outputs.

2 Implementation Details

2.1 Prelab

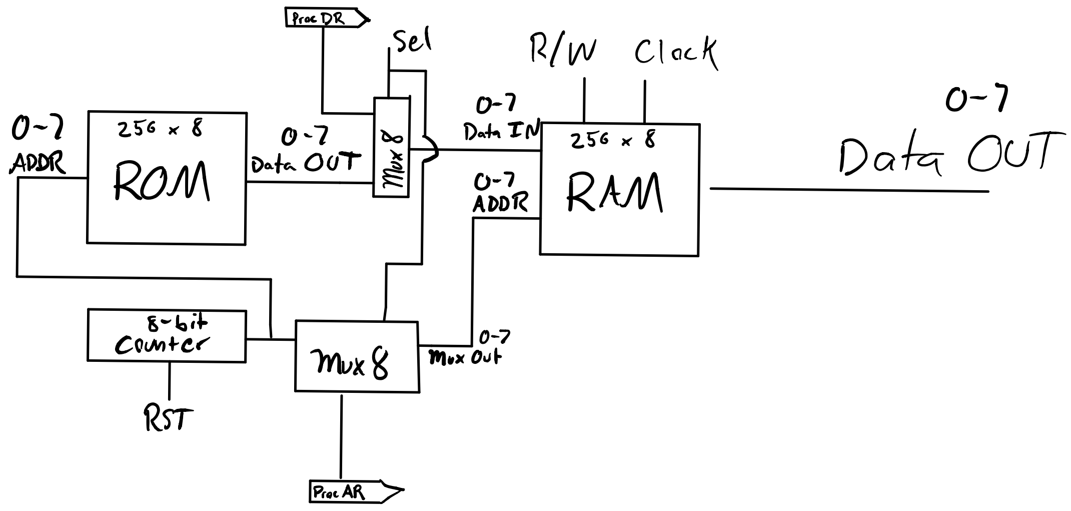

Figure 1: Memory Bootstrapping circuit to copy the entire contents of the ROM array into the RAM array.

2.2 Circuit Implementation in Schematic Editor

Figure 2: Controller and Datapath components from previous labs connected here in toy_sch.

2.3 Simulation

```
//toy_tb.v
‘timescale 1ns/1ps

module toy_tbw_tb_0;

        reg CLK = 1’b0;
        reg [7:0] D_IN = 8’b00000000;
        reg RESET = 1’b1;

        wire [7:0] ADDR;
        wire [7:0] D_OUT;
        wire MEM_EN;
        wire RORW;
        wire S0;
        wire S1;
        wire S2;
        wire S3;
        wire S4;
        wire S5;

        initial // Clock process for CLK
                begin
                        forever
                                begin
                                        CLK = 1’b0;
                                        #50;
                                        CLK = 1’b1;
                                        #50;
                                end
                end

        toy_sch UUT (
        .CLK(CLK),
        .D_IN(D_IN),
        .RESET(RESET),
        .ADDR(ADDR),
        .D_OUT(D_OUT),
        .MEM_EN(MEM_EN),
        .RORW(RORW),
        .S0(S0),
        .S1(S1),
        .S2(S2),
        .S3(S3),
        .S4(S4),
        .S5(S5));
```

```
initial
        begin
                // ------------- Current Time: 135ns
                #135;
                RESET = 1'b0;
                // -------------------------------------

                // ------------- Current Time: 235ns
                #100;
                D_IN = 8'b00000001;
                // -------------------------------------

                // ------------- Current Time: 335ns
                #100;
                D_IN = 8'b00000000;
                // -------------------------------------

                // ------------- Current Time: 435ns
                #100;
                D_IN = 8'b10101010;
                // -------------------------------------

                // ------------- Current Time: 535ns
                #100;
                D_IN = 8'b00000000;
                // -------------------------------------

                // ------------- Current Time: 735ns
                #200;
                D_IN = 8'b00000100;
                // -------------------------------------

                // ------------- Current Time: 835ns
                #100;
                D_IN = 8'b00000000;
                // -------------------------------------

                // ------------- Current Time: 1035ns
                #200;
                D_IN = 8'b00000001;
                // -------------------------------------

                // ------------- Current Time: 1135ns
                #100;
                D_IN = 8'b00000000;
                // -------------------------------------
// ------------ Current Time: 1235ns
#100;
D_IN = 8'b11111110;
// ------------------------------------

// ------------ Current Time: 1335ns
#100;
D_IN = 8'b00000000;
// ------------------------------------

// ------------ Current Time: 1535ns
#200;
D_IN = 8'b00000010;
// ------------------------------------

// ------------ Current Time: 1635ns
#100;
D_IN = 8'b00000000;
// ------------------------------------

// ------------ Current Time: 1735ns
#100;
D_IN = 8'b00000011;/////*********
// ------------------------------------

// ------------ Current Time: 1835ns
#100;
D_IN = 8'b00000000;
// ------------------------------------

// ------------ Current Time: 2035ns
#200;
D_IN = 8'b00010000;
// ------------------------------------

// ------------ Current Time: 2135ns
#100;
D_IN = 8'b00000000;
// ------------------------------------

// ------------ Current Time: 2235ns
#100;
D_IN = 8'b11111111;
// ------------------------------------

// ------------ Current Time: 2335ns
#100;
```

```
D_IN = 8'b00000000;
// ------------------------------------

// ------------- Current Time: 2535ns
#200;
D_IN = 8'b00000100;
// ------------------------------------

// ------------- Current Time: 2635ns
#100;
D_IN = 8'b00000000;
// ------------------------------------

// ------------- Current Time: 2835ns
#200;
D_IN = 8'b00001000;
// ------------------------------------

// ------------- Current Time: 2935ns
#100;
D_IN = 8'b00000000;
// ------------------------------------

// ------------- Current Time: 3035ns
#100;
D_IN = 8'b11001100;
// ------------------------------------

// ------------- Current Time: 3135ns
#100;
D_IN = 8'b00000000;
// ------------------------------------

// ------------- Current Time: 3335ns
#200;
D_IN = 8'b00000001;
// ------------------------------------

// ------------- Current Time: 3435ns
#100;
D_IN = 8'b00000000;
// ------------------------------------

// ------------- Current Time: 3535ns
#100;
D_IN = 8'b00001111;
// ------------------------------------
```

```
                        // ------------ Current Time: 3635ns
                        #100;
                        D_IN = 8'b00000000;
                        // ------------------------------------

                        // ------------ Current Time: 3835ns
                        #200;
                        D_IN = 8'b00001000;
                        // ------------------------------------

                        // ------------ Current Time: 3935ns
                        #100;
                        D_IN = 8'b00000000;
                        // ------------------------------------

                        // ------------ Current Time: 4035ns
                        #100;
                        D_IN = 8'b11111110;
                        // ------------------------------------

                        // ------------ Current Time: 4135ns
                        #100;
                        D_IN = 8'b00000000;
                        // ------------------------------------
                end
endmodule
```

Test bench runs through every given instruction by just changing values of D_IN in binary and adding waits for visibility convenience. Refer to the Expiremental Results section to view the wavefrom results of this testbench.

3 Experimental Results

4 Significance

With the toy processor now built we can now execute very simple programs including tasks like add, subtract, clear, branch if not zero, and store. To improve its use, we can implement more features like ROM and RAM paths in order to execute more complex programs, and bring the whole implementation to the board.

5 Comments/Suggestions

N.A.