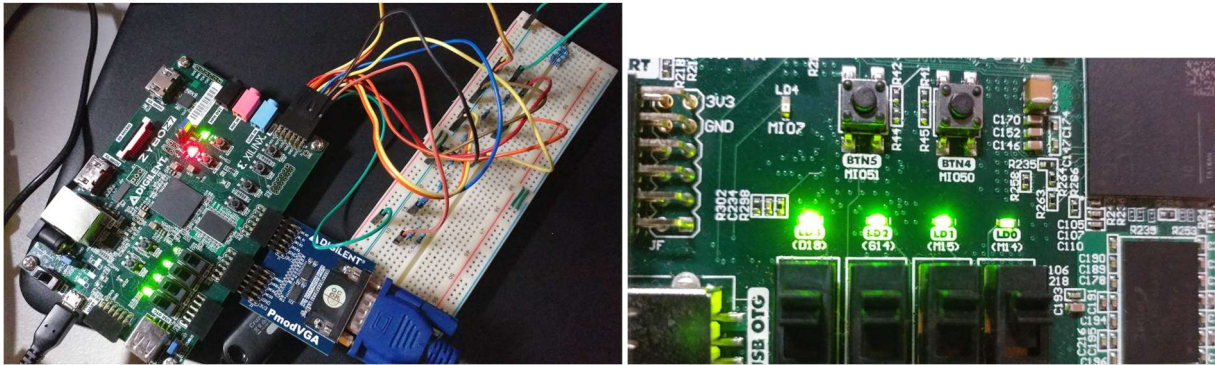


Julie Kim (Engineering Portfolio)

Long Beach Ca 90813 | 562-607-1465 | chealy_ljh@yahoo.co.uk



- RTL Analog to Digital Converter Custom IP (Spring 2021 – FPGA Based SW&HW Co-Desgn)
https://www.youtube.com/watch?v=VVtr0a4FmV0&list=PLWB9IGreGxgr_xbJwedF16atCr6rWCUh&index=2



Input Ports and Output Ports

Two Important signals
to object signal on the
screen.

Generate 25MHz pin
clock

```
16 set_property -dict { PACKAGE_PIN K18 IOSTANDARD LVCMOS33 } [get_ports { btn[0] }]; #IO_L12N_T1_MRCC_35 Sch=btn[0]
17 set_property -dict { PACKAGE_PIN P16 IOSTANDARD LVCMOS33 } [get_ports { btn[1] }]; #IO_L24N_T3_34 Sch=btn[1]
18 #set_property -dict { PACKAGE_PIN K19 IOSTANDARD LVCMOS33 } [get_ports { btn[2] }]; #IO_L10P_T1_AD11P_35 Sch=btn[2]
19 #set_property -dict { PACKAGE_PIN Y16 IOSTANDARD LVCMOS33 } [get_ports { btn[3] }]; #IO_L7P_T1_34 Sch=btn[3]
20 ##LEDs
21 set_property -dict { PACKAGE_PIN M14 IOSTANDARD LVCMOS33 } [get_ports { led[0] }]; #IO_L23P_T3_35 Sch=led[0]
22 set_property -dict { PACKAGE_PIN M15 IOSTANDARD LVCMOS33 } [get_ports { led[1] }]; #IO_L23N_T3_35 Sch=led[1]
23 set_property -dict { PACKAGE_PIN G14 IOSTANDARD LVCMOS33 } [get_ports { led[2] }]; #IO_0_35 Sch=led[2]
24 set_property -dict { PACKAGE_PIN D18 IOSTANDARD LVCMOS33 } [get_ports { led[3] }]; #IO_L3N_T0_DQS_AD1N_35 Sch=led[3]
25 #set_property -dict { PACKAGE_PIN JA (READY) IOSTANDARD LVCMOS33 } [get_ports { ja[0] }]; #IO_L21P_T3_DQS_AD14P_35 Sch=JA1_R_P
26 set_property -dict { PACKAGE_PIN M15 IOSTANDARD LVCMOS33 } [get_ports { ja[1] }]; #IO_L22P_T3_AD7P_35 Sch=JA2_R_P
27 set_property -dict { PACKAGE_PIN K16 IOSTANDARD LVCMOS33 } [get_ports { ja[2] }]; #IO_L24P_T3_AD15P_35 Sch=JA3_R_P
28 set_property -dict { PACKAGE_PIN K14 IOSTANDARD LVCMOS33 } [get_ports { ja[3] }]; #IO_L20P_T3_AD6P_35 Sch=JA4_R_P
29 set_property -dict { PACKAGE_PIN N16 IOSTANDARD LVCMOS33 } [get_ports { ja[4] }]; #IO_L21N_T3_DQS_AD14N_35 Sch=JA1_R_N
30 set_property -dict { PACKAGE_PIN L15 IOSTANDARD LVCMOS33 } [get_ports { ja[5] }]; #IO_L22N_T3_AD7N_35 Sch=JA2_R_N
31 set_property -dict { PACKAGE_PIN J16 IOSTANDARD LVCMOS33 } [get_ports { ja[6] }]; #IO_L24N_T3_AD15N_35 Sch=JA3_R_N
32 set_property -dict { PACKAGE_PIN J14 IOSTANDARD LVCMOS33 } [get_ports { ja[7] }]; #IO_L20N_T3_AD6N_35 Sch=JA4_R_N
33 #I2mod Header
34 #I2mod Header
35 set_property -dict { PACKAGE_PIN V15 IOSTANDARD LVCMOS33 } [get_ports { VGA_R[0] }]; #IO_L10P_T1_34 Sch=jc_p[1]
36 set_property -dict { PACKAGE_PIN W15 IOSTANDARD LVCMOS33 } [get_ports { VGA_R[1] }]; #IO_L10N_T1_34 Sch=jc_n[1]
37 set_property -dict { PACKAGE_PIN T11 IOSTANDARD LVCMOS33 } [get_ports { VGA_R[2] }]; #IO_L1P_T0_34 Sch=jc_p[2]
38 set_property -dict { PACKAGE_PIN T10 IOSTANDARD LVCMOS33 } [get_ports { VGA_R[3] }]; #IO_L1N_T0_34 Sch=jc_n[2]
39 set_property -dict { PACKAGE_PIN W14 IOSTANDARD LVCMOS33 } [get_ports { VGA_B[0] }]; #IO_L8P_T1_34 Sch=jc_p[3]
40 set_property -dict { PACKAGE_PIN Y14 IOSTANDARD LVCMOS33 } [get_ports { VGA_B[1] }]; #IO_L8N_T1_34 Sch=jc_n[3]
41 set_property -dict { PACKAGE_PIN T12 IOSTANDARD LVCMOS33 } [get_ports { VGA_B[2] }]; #IO_L2P_T0_34 Sch=jc_p[4]
42 set_property -dict { PACKAGE_PIN U12 IOSTANDARD LVCMOS33 } [get_ports { VGA_B[3] }]; #IO_L2N_T0_34 Sch=jc_n[4]
43 #I2mod Header JD
44 set_property -dict { PACKAGE_PIN T14 IOSTANDARD LVCMOS33 } [get_ports { VGA_G[0] }]; #IO_L5P_T0_34 Sch=jd_p[1]
45 set_property -dict { PACKAGE_PIN T15 IOSTANDARD LVCMOS33 } [get_ports { VGA_G[1] }]; #IO_L5N_T0_34 Sch=jd_n[1]
46 set_property -dict { PACKAGE_PIN P14 IOSTANDARD LVCMOS33 } [get_ports { VGA_G[2] }]; #IO_L6P_T0_34 Sch=jd_p[2]
47 set_property -dict { PACKAGE_PIN R14 IOSTANDARD LVCMOS33 } [get_ports { VGA_G[3] }]; #IO_L6N_T0_VREF_34 Sch=jd_n[2]
48 set_property -dict { PACKAGE_PIN U14 IOSTANDARD LVCMOS33 } [get_ports { haync }]; #IO_L11P_T1_SRCC_34 Sch=jd_p[3]
49 set_property -dict { PACKAGE_PIN U15 IOSTANDARD LVCMOS33 } [get_ports { vsync }]; #IO_L11N_T1_SRCC_34 Sch=jd_n[3]
```

XADC Control Speed

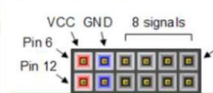


Figure 16. Pinmod diagram.



```
wire [7:0] speed;
assign speed = {4'b00000, dout[15:13]};
assign data_out = (speed < 8'h01)? 8'h1 : speed;

always@(posedge clk)
    _drdy <= {_drdy[0], drdy};

always@(*)
    case (ledidx)
    0: daddr = 7'h1E;
    1: daddr = 7'h17;
    2: daddr = 7'h1F;
    3: daddr = 7'h16;
    default: daddr = 7'h1E;
    endcase
```

Data out
is 16 bits

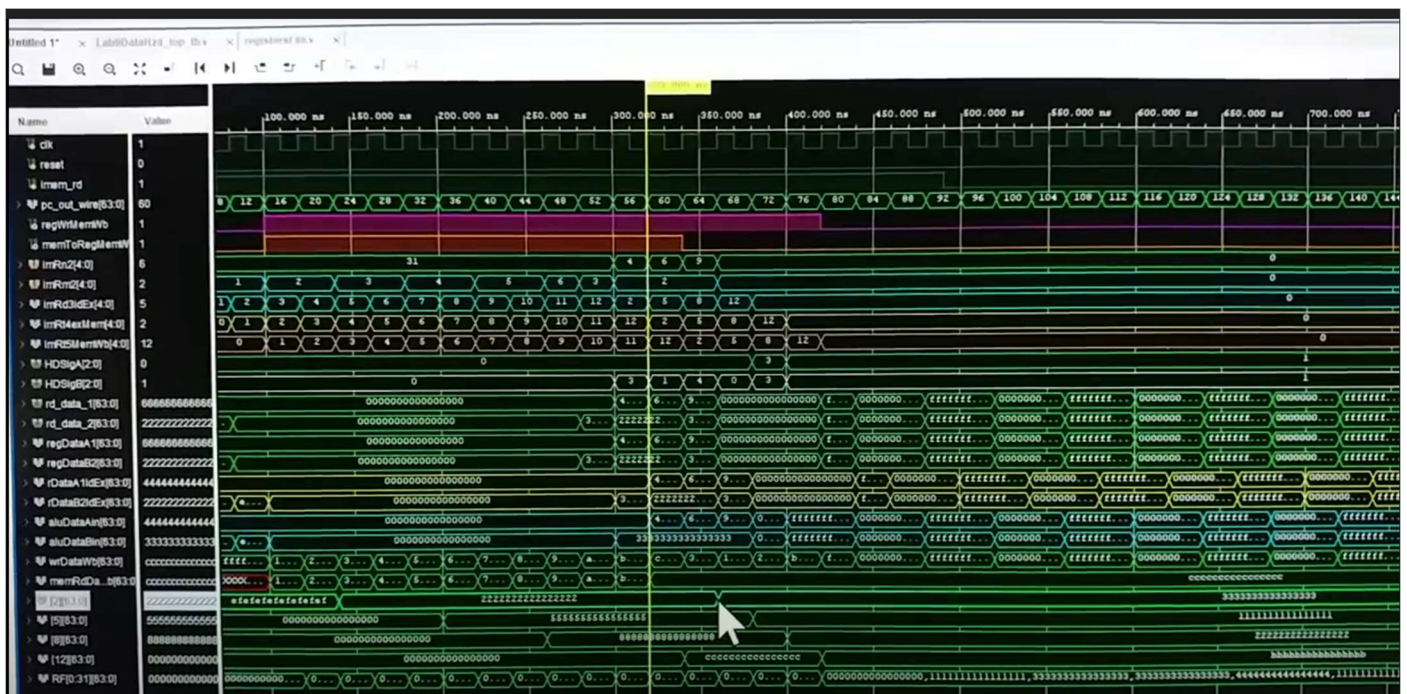
- RISC 32-bit 5 stages Pipelined with Forwarding Unit and Hazard Detector (Fall 2020 – Adv Computer Architecture I)
https://drive.google.com/file/d/1UGjc_aACZLHmnbP7u7_3vsWSSOd89l/view?usp=sharing

```
module registersFile (clk, reset, wr_en, wr_addr, wr_data,
                    rd_addr_1, rd_addr_2, rd_data_1,
                    rd_data_2);
    input clk, reset, wr_en;
    input [4:0] rd_addr_1, rd_addr_2, wr_addr;
    input [63:0] wr_data;
    output wire [63:0] rd_data_1, rd_data_2;

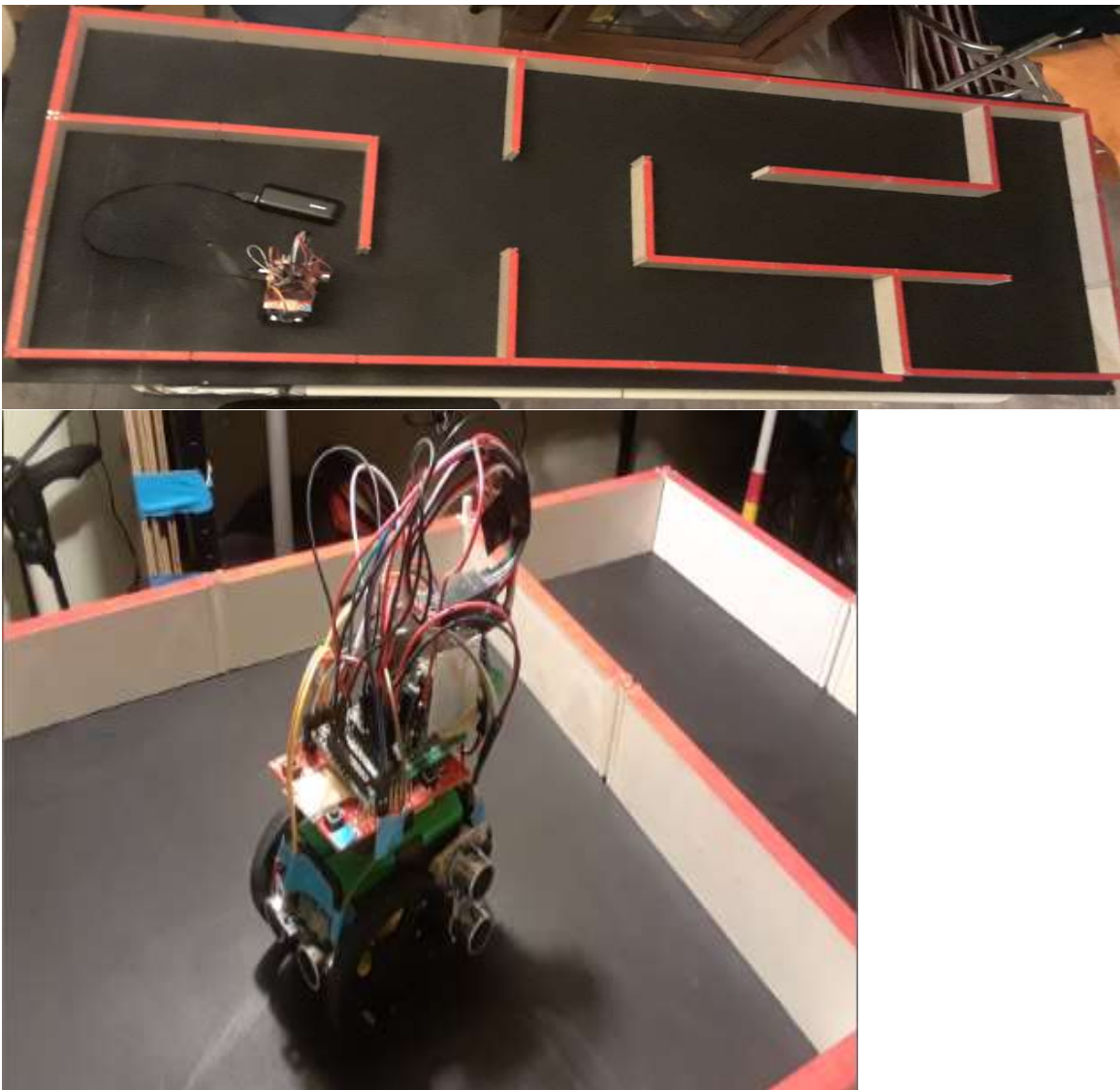
    reg [63:0] RF[0:31];
    //reg [63:0] dtA1, dtB2;
    integer i;

    always @(posedge clk, posedge reset)
```

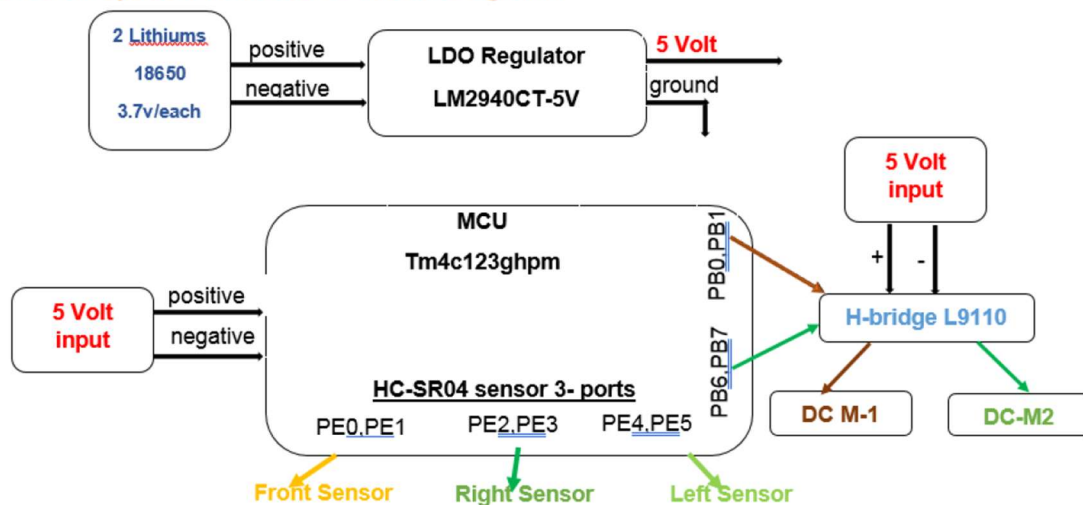
```
157         InstructionMem1.RF[56 + 3] = 8'b110_01000;
158
159         //x2
160         //OR X12, X9, X2 ; Rn, Rm ; data1, data2
161         //32'b10101010000_00010_000000_01001_01100;
162         //32'b10101010_000_00010_000000_01_001_01100;
163         InstructionMem1.RF[60 + 0] = 8'b10101010;
164         InstructionMem1.RF[60 + 1] = 8'b000_00010; //Rm==2
165         // InstructionMem1.RF[60 + 1] = 8'b000_01010; //Rm==10
166         InstructionMem1.RF[60 + 2] = 8'b000000_01;
167         InstructionMem1.RF[60 + 3] = 8'b001_01100;
168
169         @(negedge clk)
170         reset = 1'b1;
171         @(negedge clk)
172         reset = 0;
173         imem_rd = 1'b1;
```



- Real-Time Autonomous Robot (*Fall 2018 - Computer Engr Snr Proj II*)
https://www.youtube.com/watch?v=9_2CTwICjk&list=PLWB9IGreRgxzoJ4VC523rkH4-hUHCCX6L&index=1



Overall System Functional Block Diagram:



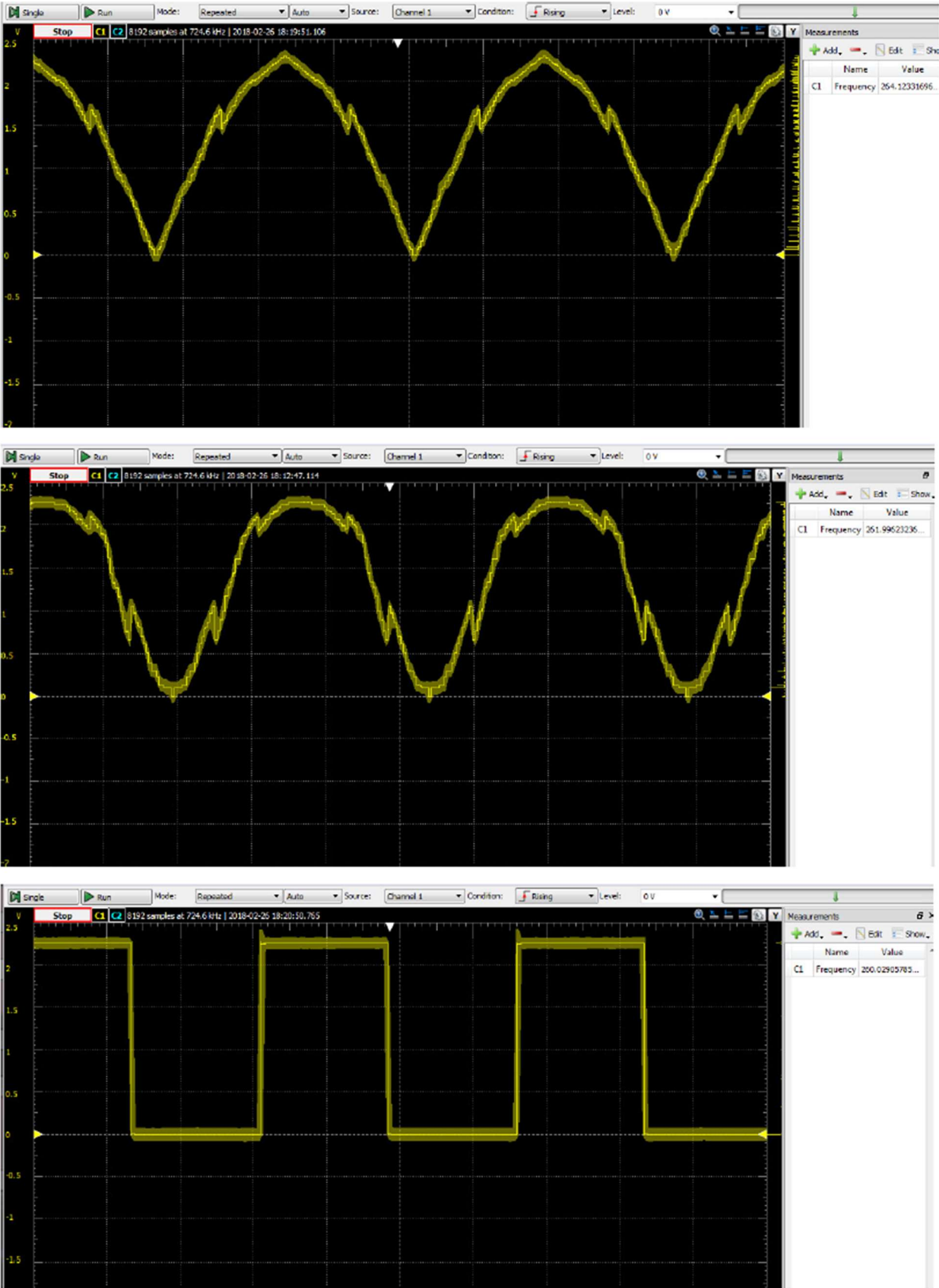
-
- The diagram illustrates a DAC-based audio amplifier circuit. It features a 3.3V supply connected to a DAC module (DAC_out) and an LM386 op-amp. The DAC module is connected to the op-amp's non-inverting input (pin 3). The op-amp's output (pin 5) is connected to a speaker through a series of capacitors. The circuit is powered by a 3.3V supply and a 10V supply.

Calibrating Voltage Value to Binary value

Page 4

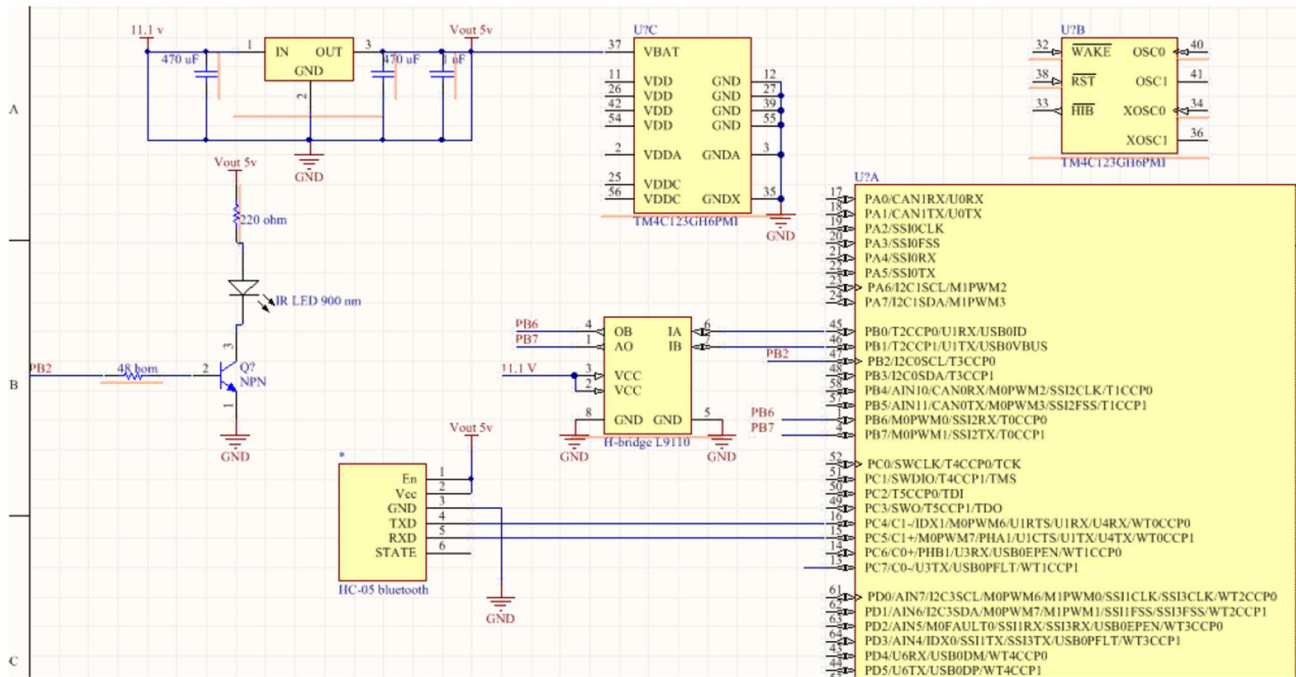
Logic Analyzer Simulation of R2R Digital to Analog Convertor circuit:

Figures:

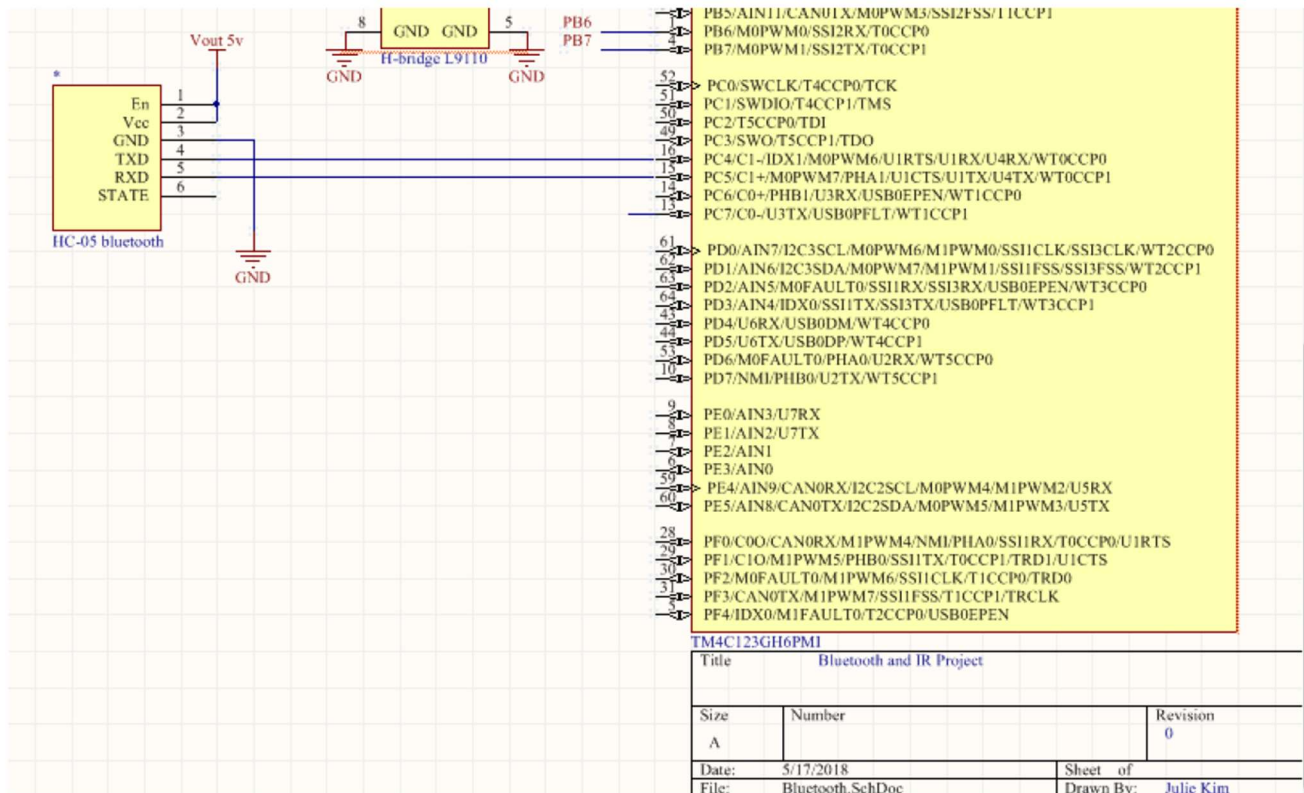


- Bluetooth Controlled Robot (Spring 2018 – Microprocessor+Ctr III)

Schematics:



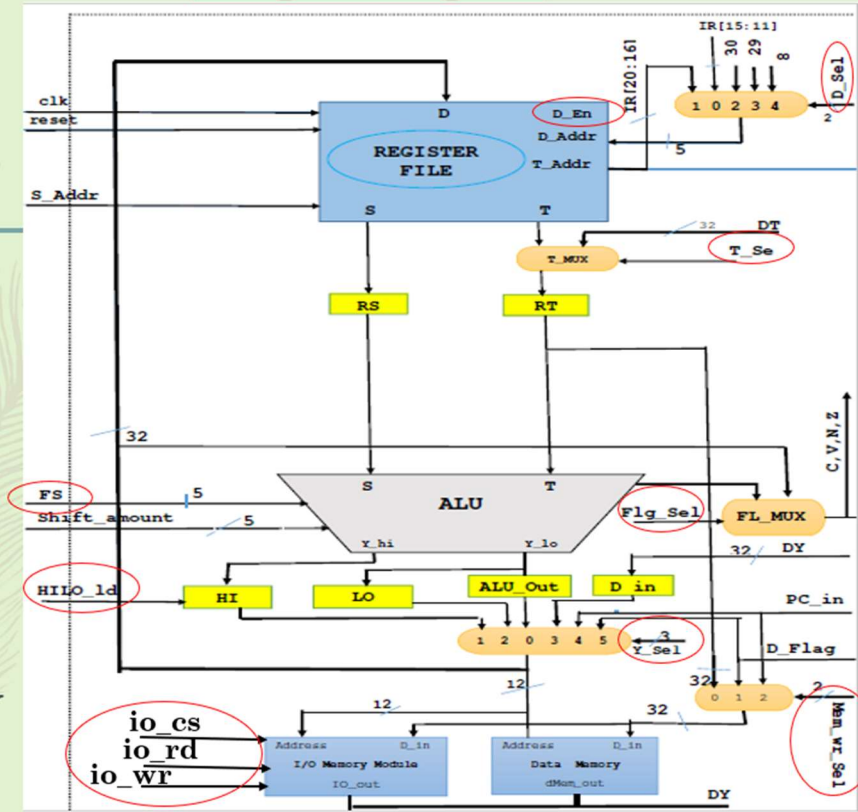
CECS447 | PROJECT3 BLUETOOTH AND



- MIPS Non-Pipelined 32-bit CPU with Barrel Shifter (*Computer Architecture – Fall 2017*)
<https://www.youtube.com/watch?app=desktop&v=e0dxdGOG4c&t=483s>

Integer Datapath:

- Memory Address, 12bit
- Memory Data in, 32bit
- D_in: IDP, DY, PC, D_Flag
- Memory Output: DY



Barrel Shift Left Result Verification:

```

/*****
*BARREL SHIFT LEFT LOGICAL 32
*****/

{5'h0C, 5'd1}: {c, y_lo} = {TData[31], {TData[30:0], 1'b0}}; //SLL_1
{5'h0C, 5'd2}: {c, y_lo} = {TData[30], {TData[29:0], 2'b0}}; //SLL_2
{5'h0C, 5'd3}: {c, y_lo} = {TData[29], {TData[28:0], 3'b0}}; //SLL_3
{5'h0C, 5'd4}: {c, y_lo} = {TData[28], {TData[27:0], 4'b0}}; //SLL_4
{5'h0C, 5'd5}: {c, y_lo} = {TData[27], {TData[26:0], 5'b0}}; //SLL_5
{5'h0C, 5'd6}: {c, y_lo} = {TData[26], {TData[25:0], 6'b0}}; //SLL_6
{5'h0C, 5'd7}: {c, y_lo} = {TData[25], {TData[24:0], 7'b0}}; //SLL_7
{5'h0C, 5'd8}: {c, y_lo} = {TData[24], {TData[23:0], 8'b0}}; //SLL_8
{5'h0C, 5'd9}: {c, y_lo} = {TData[23], {TData[22:0], 9'b0}}; //SLL_9

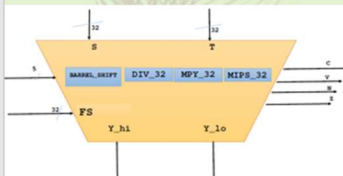
```

Opcode

Function

0x00 \$rs \$rt \$rd shamt 0x00

ALU Internal Design



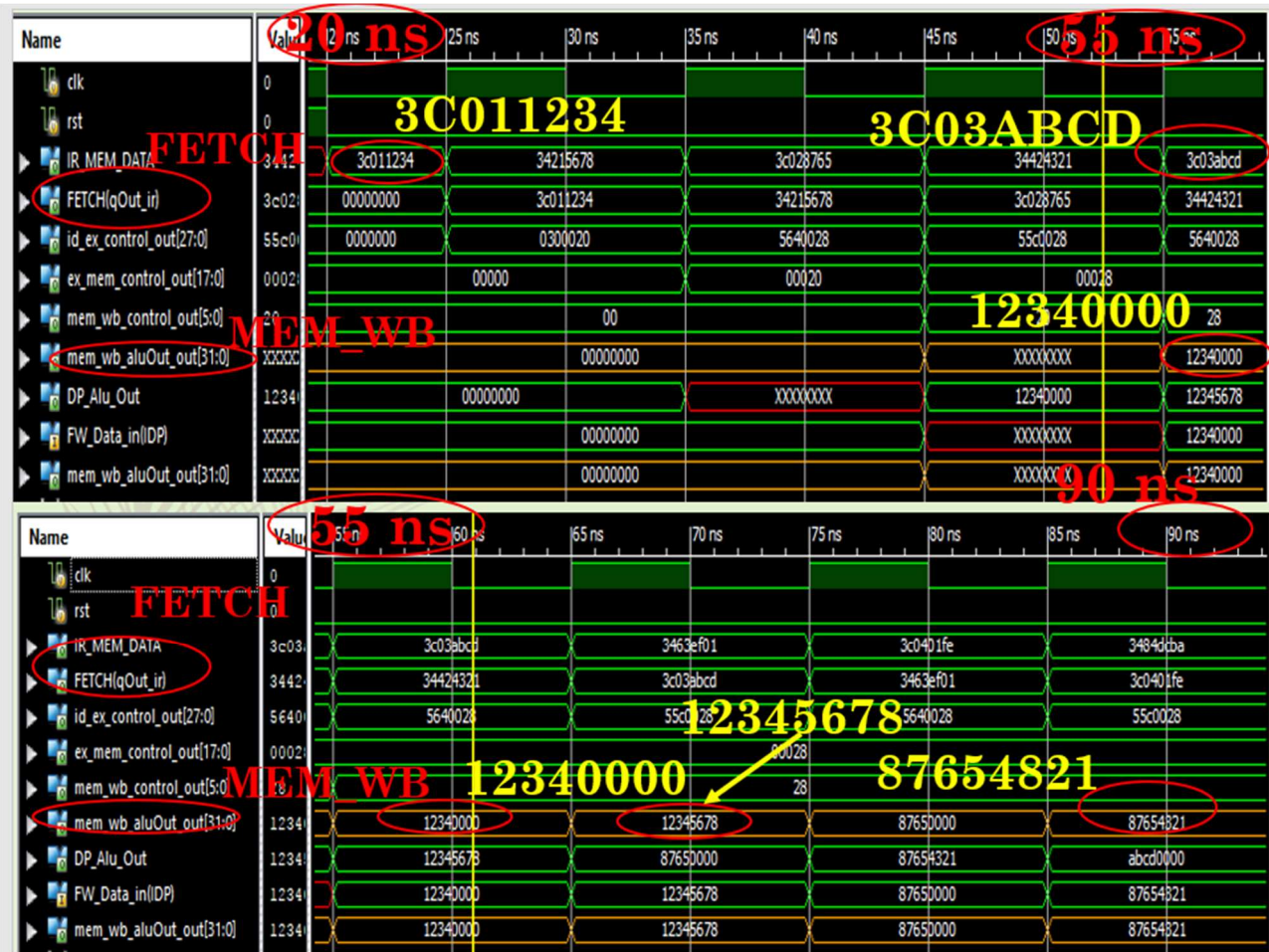
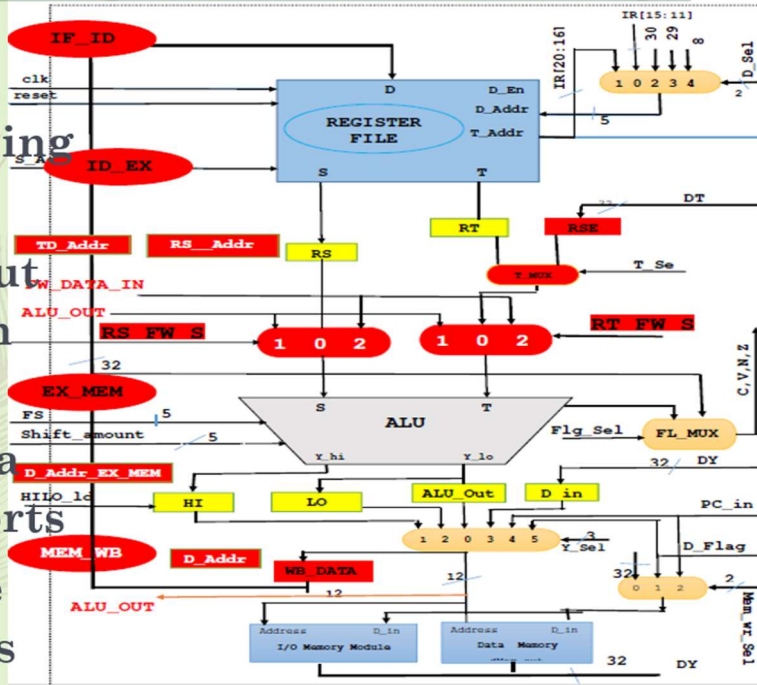
```

*****
CECB440ALUTestbenchResults
*****
SHIFT LEFT LOGICAL:
Finished circuit initialization process.
SLL 04: T=800ffff0 Ylo=00ffff00 c=0 v=0 n=0 z=0
SLL 08: T=800ffff0 Ylo=0ffff000 c=0 v=0 n=0 z=0
SLL 16: T=800ffff0 Ylo=fff00000 c=1 v=0 n=1 z=0
SLL 20: T=800ffff0 Ylo=ff000000 c=1 v=0 n=1 z=0

```

FIVE Stages Pipeline Hardware Design:

- Forwarding Unit
- Two Input Selection ports
- Two Data Input Ports
- Six more Registers



The screenshot displays a Verilog simulation environment. A large red "1231.0 ns" is overlaid on the console window, indicating the simulation has stopped at 1231.0 ps. The console window shows the following text:

```
time=1231.0 ps, S_addr[0f]=ffffef8 || T_addr[1f]=xxxxxxx
FINISH READING REGISTER FILE AFTER BREAK!!!
Stopped at time : 1231 ns : File C:/Users/chealykim/Desktop/440_Final_Project/MIPS_Processor/MIPS_Processor/MCU.v* Line 1391
ISim>
```

The background shows the Verilog code for the MCU.v file. The code includes a break statement at line 1371, which is highlighted by a yellow arrow. The code also includes a display statement for the register file after the break.

End of Portfolio. Thank you for taking the time to review.

Sincerely,

Julie Kim
Julie Kim