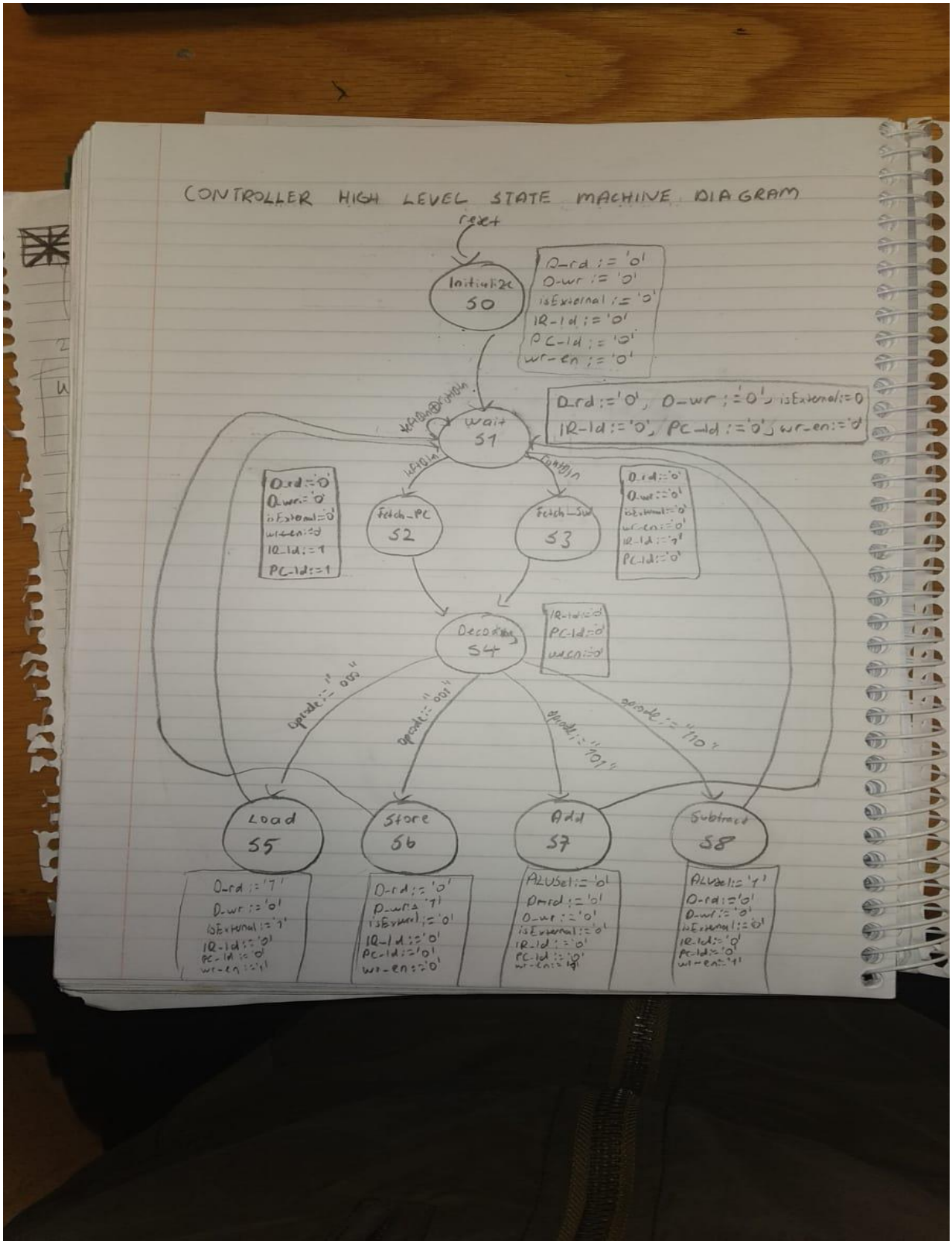
CS223 Digital Design

Kerem Şahin

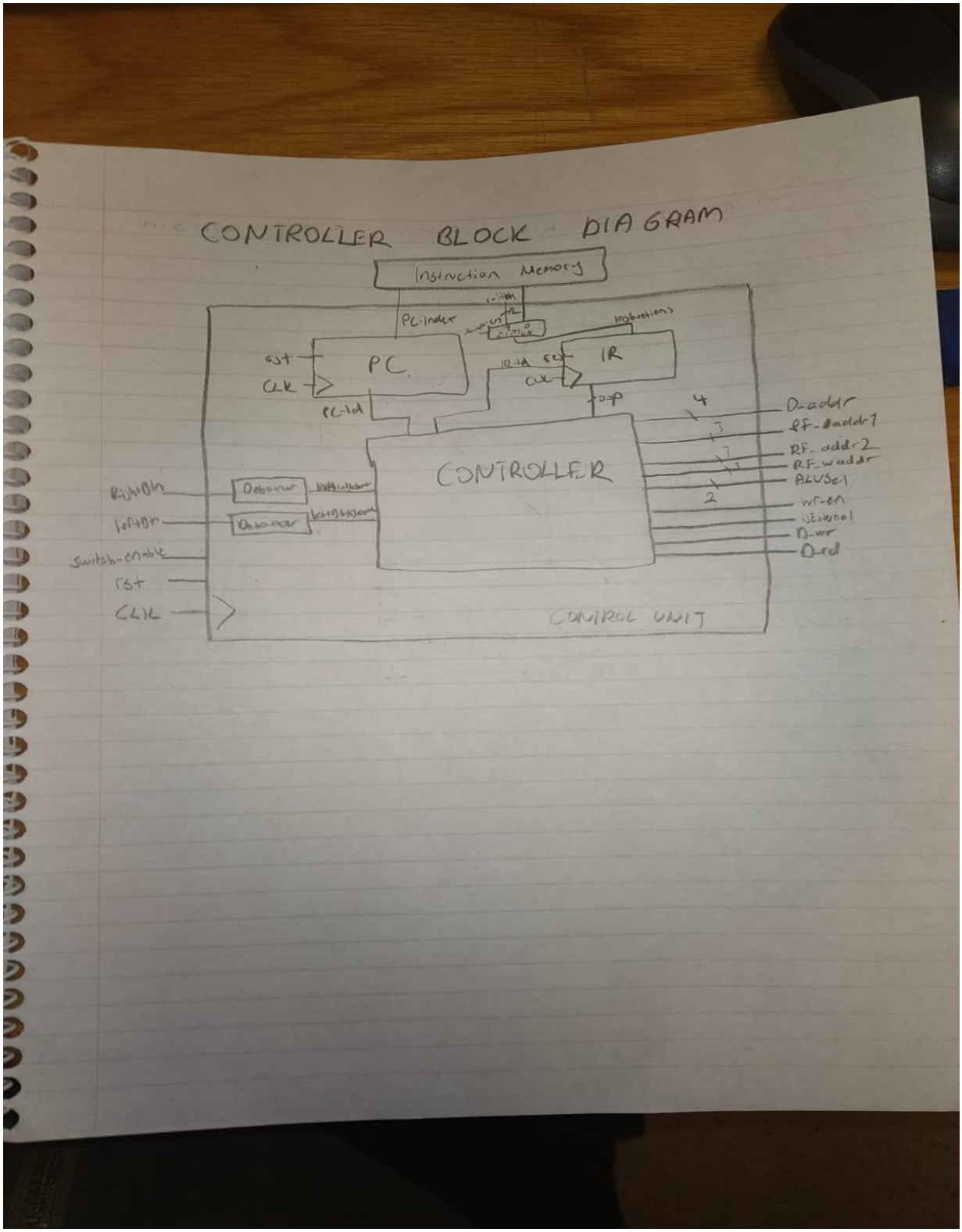
21901724

CS223 Final Project Report

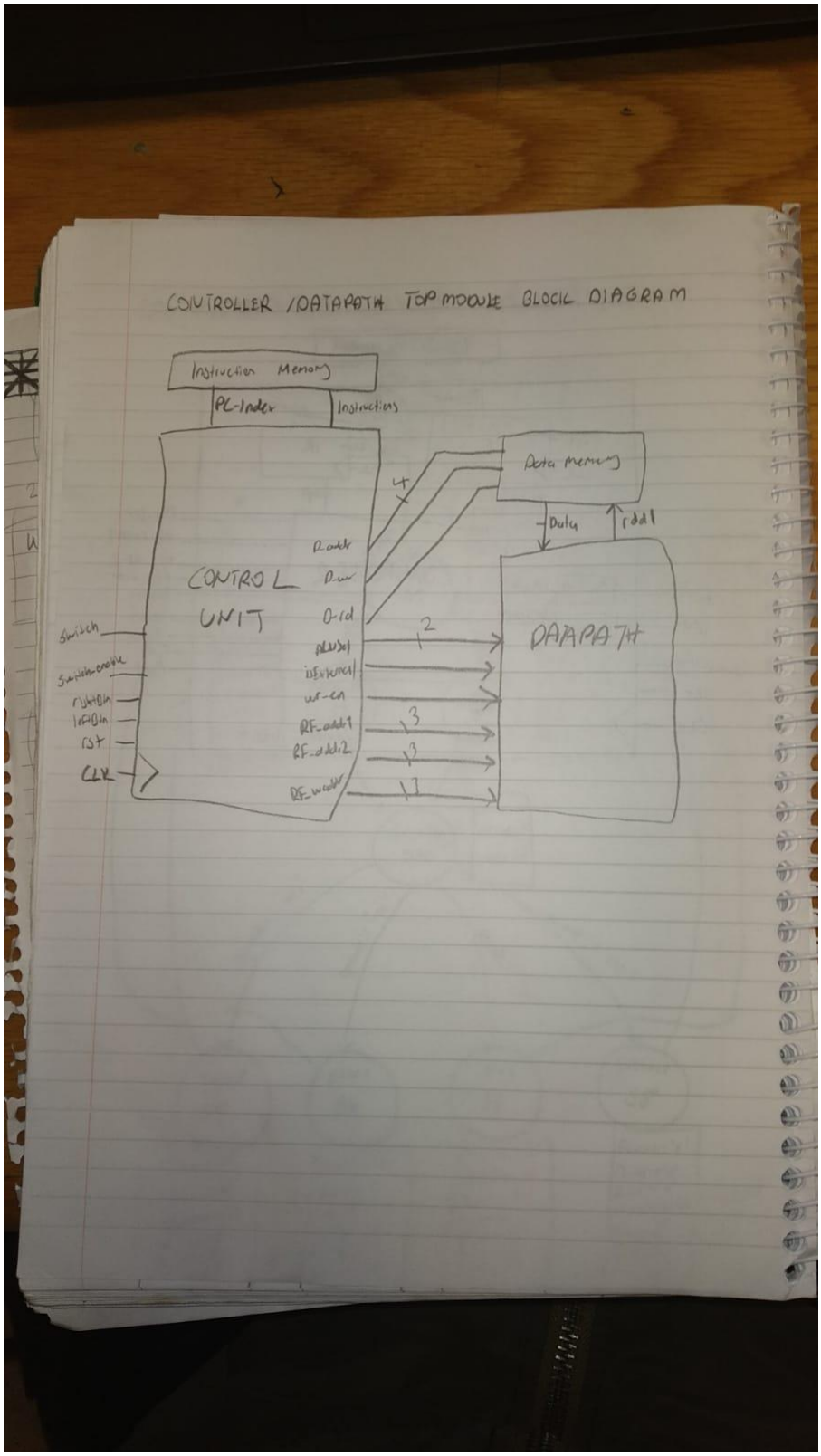
A) Controller High-Level State Machine Diagram



B) Controller Block Diagram



C) Controller/Datapath Top Module Block Diagram



D) Testbench

```
module testbench_Simple_Processor();


    logic[3:0] EXTData;
    logic isExternal, clk, pushButn, reset;
    logic[2:0] AddrSrc1;
    logic[2:0] AddrSrc2;
    logic[2:0] AddrDest;
    logic[1:0] ALUSel;
    logic[3:0] Res;


    Simple_Processor processxd( clk, reset, leftBtn, rightBtn, switchEn, [11:0] switch,
[6:0]seg, dp,[3:0] an


    Datapath pathmodule(EXTData, AddrSrc1, AddrSrc2, AddrDest, isExternal, clk,
pushButn, reset, ALUSel, Res);


    initial
        clk = 1;
    always
        begin
        #1;
        clk = ~clk;
        end


    initial begin
    reset = 1; #10;
    reset = 0; #10;
```

```
EXTData = 1;
isExternal = 1;
AddrSrc1 = 1;
AddrSrc2 = 1;
AddrDest = 0;
pushButn = 0;
ALUSel = 2'b00;
#10;
EXTData = 5;
isExternal = 1;
AddrSrc1 = 1;
AddrSrc2 = 1;
AddrDest = 1;
pushButn = 1;
ALUSel = 2'b01;
#10;
EXTData = 2;
isExternal = 1;
AddrSrc1 = 1;
AddrSrc2 = 1;
AddrDest = 2;
pushButn = 0;
ALUSel = 2'b10;
#10;
EXTData = 0;
isExternal = 1;
AddrSrc1 = 1;
```

```
        AddrSrc2 = 0;
        AddrDest = 3;
        pushButn = 1;
        ALUSel = 2'b11;
        #10;
        end
endmodule

module testbench_Simple_Processor();

        logic clk, reset, leftBtn, rightBtn, switchEn;
        logic [3:0] Res, rdd1;
        logic [11:0] switch;
        logic [6:0] seg;
        logic [3:0] an;
        logic dp;
        assign switchEn = 0;
        assign switch = 0;
        assign rightBtn = 0;
        Simple_Processor maalesefxd( clk, reset, leftBtn, rightBtn, switchEn, switch,seg,
dp, an, Res, rdd1);

        initial
                clk = 1;
        always
                begin
                #10;
                clk = ~clk;
```

```
        end

    initial begin
    reset = 1;
    #10;
    reset = 0;
    #20;
    leftBtn = 0;
    #20;
    leftBtn = 1;
    #20;
    leftBtn = 0;
    #20;
    leftBtn = 1;
    #20;
    leftBtn = 0;
    #20;
    leftBtn = 1;
    #20;
    leftBtn = 0;
    #20;
    leftBtn = 1;
    #20;
    leftBtn = 0;
    end

endmodule

module testbench_ROM();
    logic [2:0] rda;
    logic [11:0] rdd;
```

```
        Instruction_Memory IM_testbench( rda, rdd );

        initial begin
        #5;
        rda = 3'b000;
        #5;
        rda = 3'b001;
        #5;
        rda = 3'b010;
        #5;
        rda = 3'b011;
        #5;
        rda = 3'b100;
        #5;
        rda = 3'b101;
        #5;
        rda = 3'b110;
        #5;
        rda = 3'b111;
        end
endmodule

module testbench_PC();
        logic clk, rst, ld;
        logic [2:0] q;
        PC pc_test( clk, ld, rst, q );
```

```
        initial
            clk = 1;
        always
            begin
            #1;
            clk = ~clk;
            end

        initial begin
        #5; rst = 1;
        #5; rst = 0;
        #1; ld = 0;
        #1; ld = 1;
        #1; ld = 0;
        #1; ld = 1;
        #1; ld = 0;
        #1; ld = 1;
        #1; ld = 0;
        #1; ld = 1;
        end
endmodule

module testbench_IR();
        logic clk, ld, rst;
        logic [11:0] d, q;
```

```
        IR IR_test( clk, ld, rst, d, q );

        initial
                clk = 1;
        always
                begin
                #1;
                clk = ~clk;
                end

        initial begin
        #5; rst = 1;
        #5; rst = 0;
        #1; ld = 0;
        #1;
        ld = 1;
        d = 12'b0000000000000010;
        #1; ld = 0;
        end
endmodule

module testbench_controlUnit();

        logic clk, reset, switchEn;
        logic [11:0] switch;
        logic [3:0] D_addr;
        logic [2:0] RF_addr1, RF_addr2, RF_waddr;
```

```
logic [1:0] ALUSel;
logic IR_ld, PC_ld, isLoad, isStore, isAdd, isSubtract;
logic D_rd, D_wr, isExternal, wr_en, leftBtnDebounce, rightBtnDebounce;

assign switchEn = 0;

Control_Unit controlUnit( clk, reset, leftBtnDebounce, rightBtnDebounce, switchEn, switch, D_addr, D_rd,D_wr,RF_addr1, RF_addr2, RF_waddr, ALUSel, isExternal, wr_en,IR_ld, PC_ld, isLoad, isStore, isAdd, isSubtract );

initial
    clk = 1;
always
    begin
    #10;
    clk = ~clk;
    end

initial begin
reset = 1;
#10; reset = 0;
#30;
leftBtnDebounce = 1;
#20; leftBtnDebounce = 0;
#20;
leftBtnDebounce = 1;
#20; leftBtnDebounce = 0;
#20;
```

```
leftBtnDebounce = 1;
#20; leftBtnDebounce = 0;
#20;
leftBtnDebounce = 1;
#20; leftBtnDebounce = 0;
  leftBtnDebounce = 1;
#20; leftBtnDebounce = 0;
#20;
  leftBtnDebounce = 1;
#20; leftBtnDebounce = 0;
#20;
  leftBtnDebounce = 1;
#20; leftBtnDebounce = 0;
#20;
  leftBtnDebounce = 1;
#20; leftBtnDebounce = 0;
#20;
        leftBtnDebounce = 1;
#20; leftBtnDebounce = 0;
#20;
leftBtnDebounce = 1;
#20; leftBtnDebounce = 0;
#20;
leftBtnDebounce = 1;
#20; leftBtnDebounce = 0;
#20;
leftBtnDebounce = 1;
```

```
        #20; leftBtnDebounce = 0;
          leftBtnDebounce = 1;
        #20; leftBtnDebounce = 0;
        #20;
          leftBtnDebounce = 1;
        #20; leftBtnDebounce = 0;
        #20;
          leftBtnDebounce = 1;
        #20; leftBtnDebounce = 0;
        #20;
          leftBtnDebounce = 1;
        #20; leftBtnDebounce = 0;
        #20;
        end

endmodule

module testbench_controllerUnit( );
        logic clk, reset, leftBtnDebounce, rightBtnDebounce, switchEn;
        logic [11:0] switch;
        logic [3:0] D_addr;
        logic D_rd, D_wr;
        logic [2:0] RF_addr1, RF_addr2, RF_waddr;
        logic [1:0] ALUSel;
        logic isExternal;
        logic wr_en;
        logic [2:0] PC_out;
```

```
    Control_Unit controlUnitTest(
endmodule

module testbench_control();
    logic clk, reset,leftBtnDebounce, rightBtnDebounce, wr_en;
    logic [2:0] opcode;
    logic isLoad, isStore, isAdd, isSubtract, D_rd, D_wr, isExternal, IR_ld, PC_ld;
    logic [1:0] ALUSel;

    Controller controller( clk, reset, leftBtnDebounce, rightBtnDebounce, opcode,
isLoad, isStore, isAdd, isSubtract, D_rd, D_wr, isExternal, IR_ld, PC_ld, ALUSel, wr_en );

    initial
        clk = 1;
    always
        begin
        #10;
        clk = ~clk;
        end

    initial begin
    reset = 1;
    #10; reset = 0;
    #30;
    opcode = 3'b000;
    leftBtnDebounce = 1;
    #10; leftBtnDebounce = 0;
    #10;
```

```
        opcode = 3'b001;
        leftBtnDebounce = 1;
        #10; leftBtnDebounce = 0;
        #10;
        opcode = 3'b101;
        leftBtnDebounce = 1;
        #10; leftBtnDebounce = 0;
        #10;
        opcode = 3'b110;
        leftBtnDebounce = 1;
        #10; leftBtnDebounce = 0;
        end
endmodule

module tb_DataMem();
          logic D_rd, D_wr, clk, reset;
          logic [3:0] D_addr, W_data;
          logic [3:0] R_data;
          Data_memory data_memory_test(D_rd, D_wr, clk, reset, D_addr, W_data,
R_data);

          initial
                clk = 1;
        always
                begin
                #10;
                clk = ~clk;
                end
```

```
initial begin
reset = 1;
D_wr = 0;
D_addr = 0;
W_data = 0;
#10;
reset = 0;
#10;
D_rd = 1;
D_addr = 0;
#10;
D_addr = 1;
#10;
D_addr = 2;
#10;
D_addr = 3;
#10;
D_addr = 4;
#10;
D_addr = 5;
#10;
D_addr = 6;
#10;
D_addr = 7;
#10;
D_wr = 1;
```

```
        D_rd = 0;
        D_addr = 8;
        W_data = 4'hF;
        #10;
        D_wr = 0;
        D_rd = 1;
        end
endmodule
```