Birla Institute of Technology and Science – Pilani, Hyderabad Campus
Second Semester 2015-16
CS F342: Computer Architecture Assignment (15 Marks)

1. (a) Implement 3-stage pipelined processor in verilog. This processor supports load immediate (li) and shift right arithmetic (sra) instructions only. The processor has Reset, CLK as inputs and no outputs. The processor has instruction fetch unit, register file and Execution/Writeback unit. The processor also contains two pipelined registers IF/ID and ID/EXWB. When reset is activated the PC, IF/ID, ID/EXWB registers are initialized to 0, the instruction memory and registerfile get loaded by predefined values. When the instruction unit starts fetching the first instruction the pipeline registers contain unknown values. When the second instruction is being fetched in IF unit, the IF/ID registers will hold the instruction code for first instruction. When the third instruction is being fetched by IF unit, the IF/ID register contains the instruction code of second instruction and ID/EXWB register contains information related to first instruction.

The instruction and the 8-bit instruction format are shown below:

sra DestinationReg, shiftamount (Shifts data in register specified by register number in RDst field by shift amount and moves back result to same register. Opcode for sra is 00)

Opcode

00	RDst	Shamt
7:6	5:3	2:0

Example usage: sra R0, 4 Arithmetic shift value in R0 by 4 times and store result back in R0.

li DestinationReg, ImmediateData (Signextends data specified in instruction field (2:0) to 8-bits and stores it in register specified by register number in RDst field. Opcode for sra is 11)

Opcode

11	RDst	Immediate Data
7:6	5:3	2:0

Example usage: li R3, 4 (4 = 100, signextension will result in 1111100. This data moves in to R3.

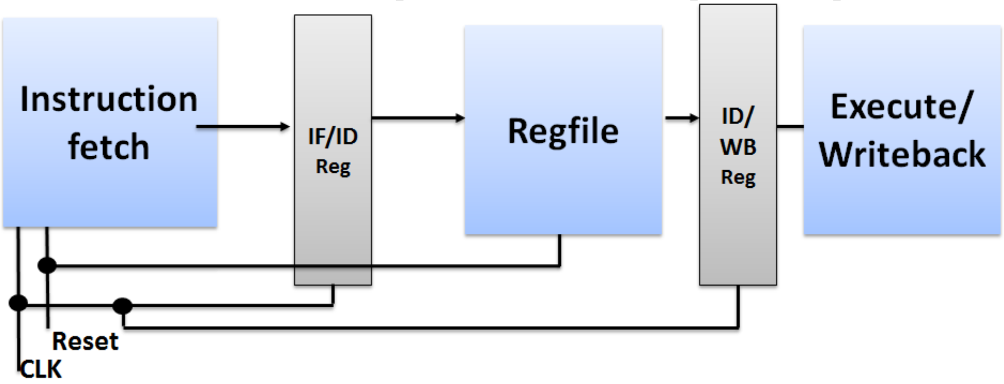
Assume the register file contains 8 registers (R0-R7) each register can hold 8-bit data. On reset assume that the instruction memory gets initialized with four instructions.

li Rx, 4
sra Rx, 2
sra Ry, 4
li Rz, 23

Where x , y, z are related to last 3 digits of your ID No.

If ID number: 20XXXXXXABCH, then x =A mod 8 (A%8),
y= (B+2) mod 8 ((B+2)%8),
y= (C+3) mod 8 ((C+3)%8),

A block level representation of 3-stage pipelined processor is shown below. Please note that for registerfile implementation, both read and write are independent of clk. Write operation depends on control signal.



As part of the assignment three files should be submitted in zipped folder.

1. PDF version of this Document with all the Questions below answered with file name as **IDNO_NAME.pdf**.

2. Design Verilog Files for instruction fetch block, Register file and Pipeline registers.

3. Design Verilog file for the main processor.

The name of the zipped folder should be in the format IDNO_NAME.zip

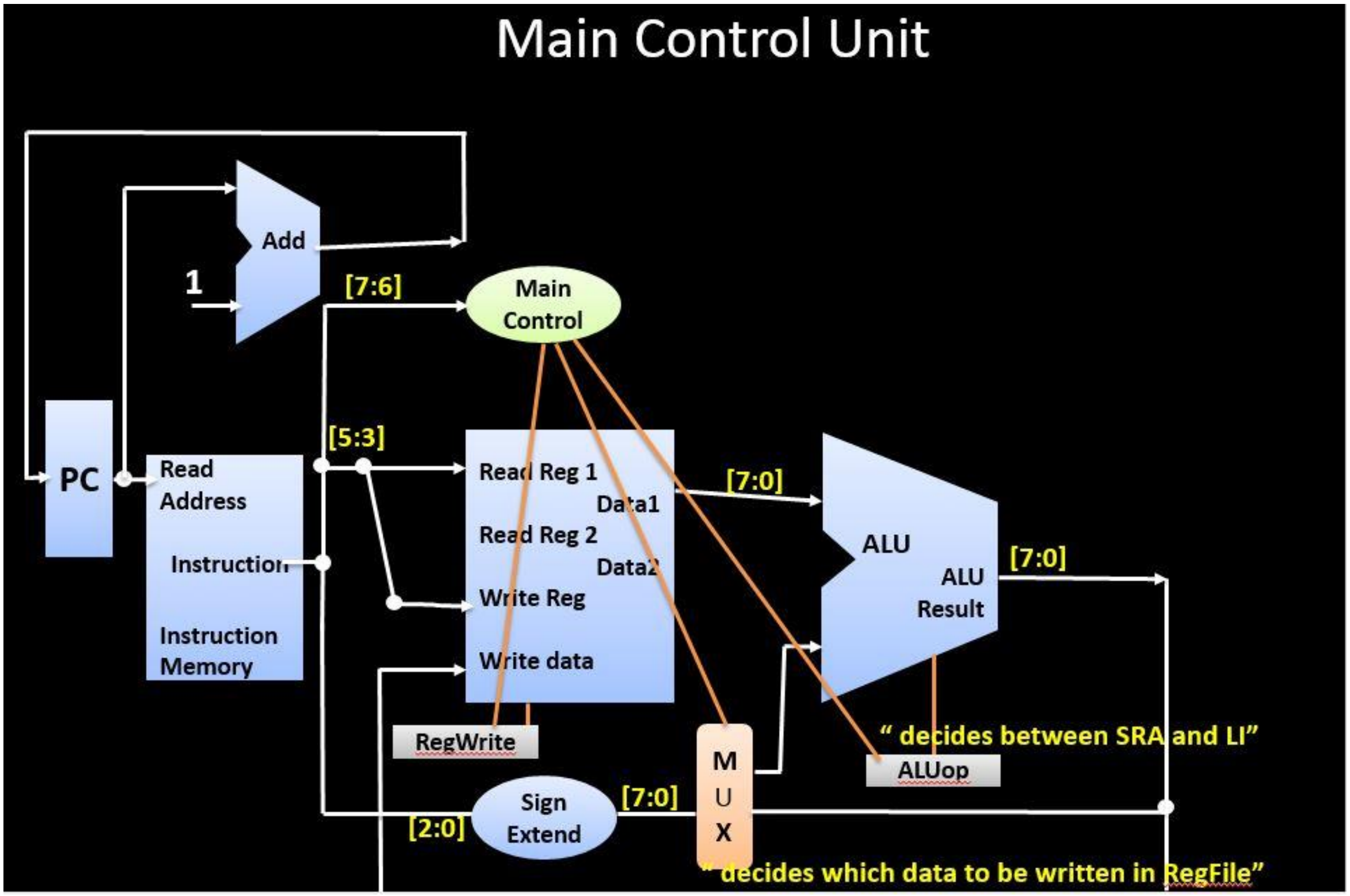
The due date for submission is 16-March-2016, 5:00 PM.

Name: AMAN NIDHI **ID No:** 2013A3PS400H

Questions Related to Assignment

1. **Draw the complete datapath and control signals of the 3-stage pipelined processor. A sample datapath for 5-stage pipelined MIPS processor has been discussed in class. A ppt named Assignmenthelp.ppt contains this 5-stage processor is uploaded in CMS. You can modify this according to your specification.**

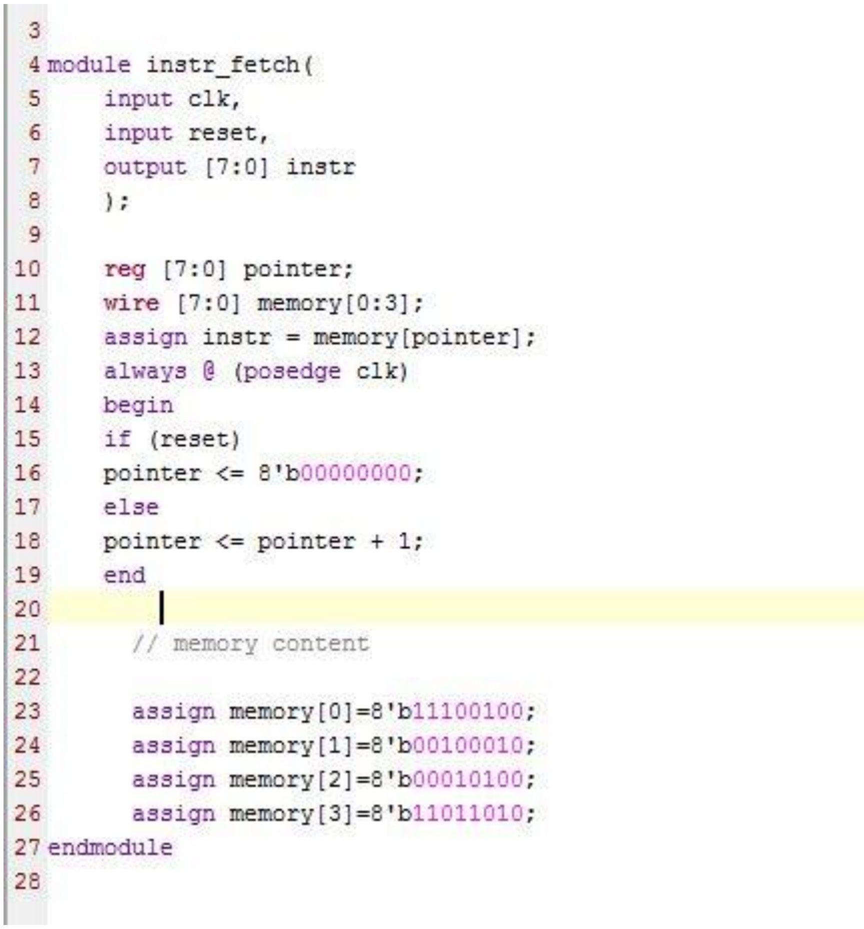
Answer:


Main Control Unit

" decides between SRA and LI"

" decides which data to be written in RegFile"

2. **Implement the Instruction Fetch block. Copy the <u>image</u> of Verilog code of the Instruction fetch block here**

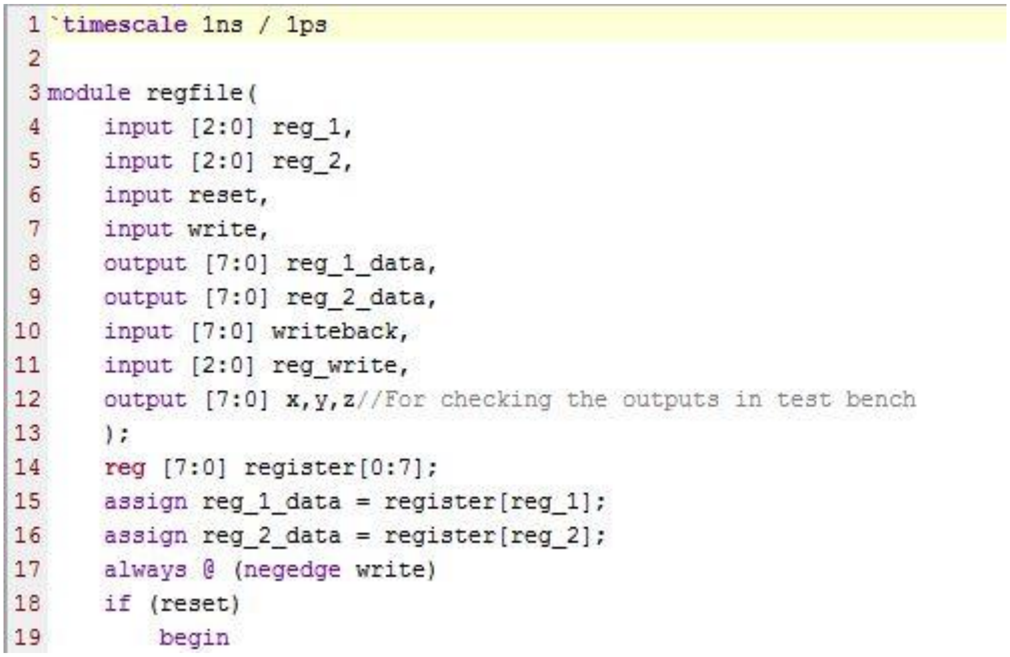


```
module instr_fetch(
    input clk,
    input reset,
    output [7:0] instr
    );

    reg [7:0] pointer;
    wire [7:0] memory[0:3];
    assign instr = memory[pointer];
    always @ (posedge clk)
    begin
    if (reset)
    pointer <= 8'b00000000;
    else
    pointer <= pointer + 1;
    end

      // memory content

      assign memory[0]=8'b11100100;
      assign memory[1]=8'b00100010;
      assign memory[2]=8'b00010100;
      assign memory[3]=8'b11011010;
endmodule

```

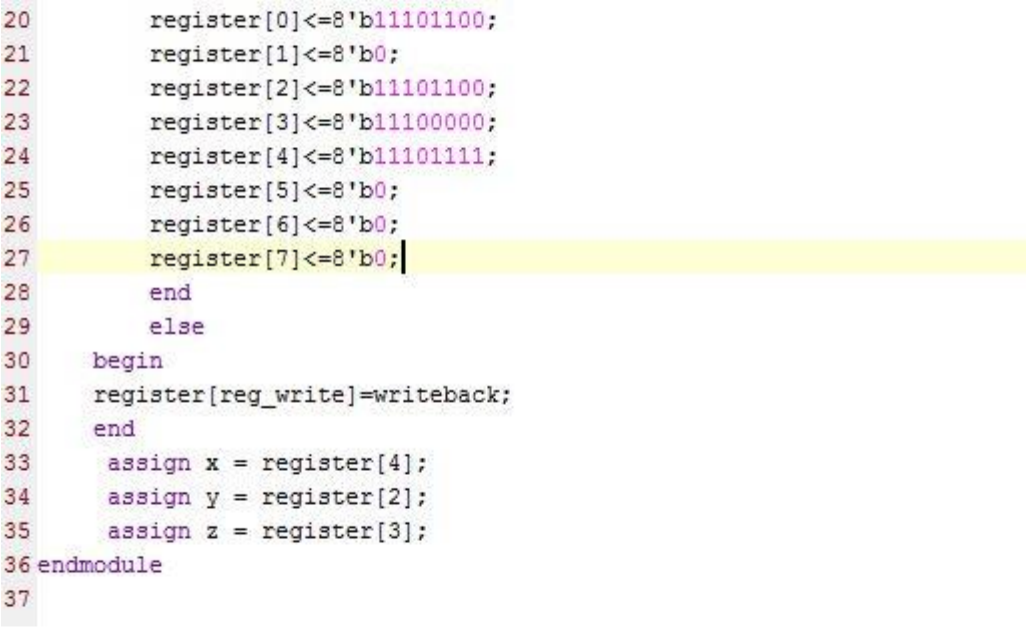
Answer:

3. Implement the Register File and copy the image of Verilog code of Register file unit here.

Answer:



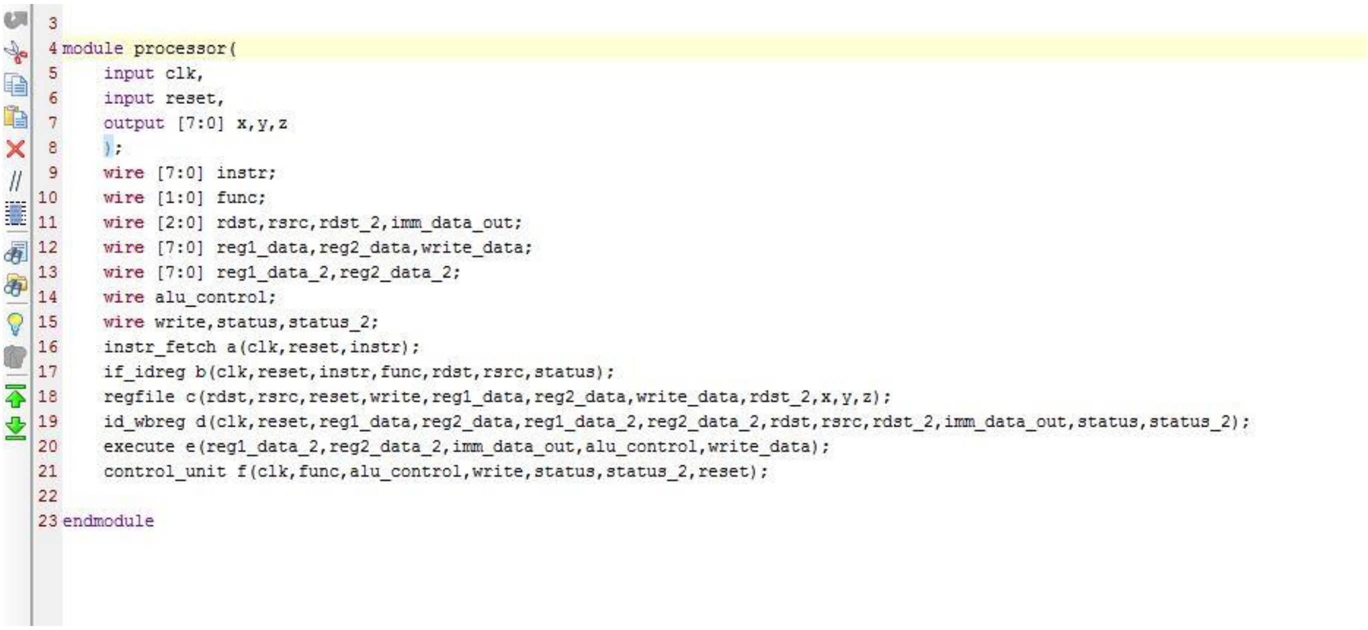
```
`timescale 1ns / 1ps

module regfile(
    input [2:0] reg_1,
    input [2:0] reg_2,
    input reset,
    input write,
    output [7:0] reg_1_data,
    output [7:0] reg_2_data,
    input [7:0] writeback,
    input [2:0] reg_write,
    output [7:0] x,y,z//For checking the outputs in test bench
    );
    reg [7:0] register[0:7];
    assign reg_1_data = register[reg_1];
    assign reg_2_data = register[reg_2];
    always @ (negedge write)
    if (reset)
        begin
```



```
        register[0]<=8'b11101100;
        register[1]<=8'b0;
        register[2]<=8'b11101100;
        register[3]<=8'b11100000;
        register[4]<=8'b11101111;
        register[5]<=8'b0;
        register[6]<=8'b0;
        register[7]<=8'b0;
        end
        else
    begin
    register[reg_write]=writeback;
    end
     assign x = register[4];
     assign y = register[2];
     assign z = register[3];
endmodule

```

4. **Implement complete processor in Verilog (using all the datapath blocks). Copy the <u>image</u> of Verilog code of the processor here.**

Answer:



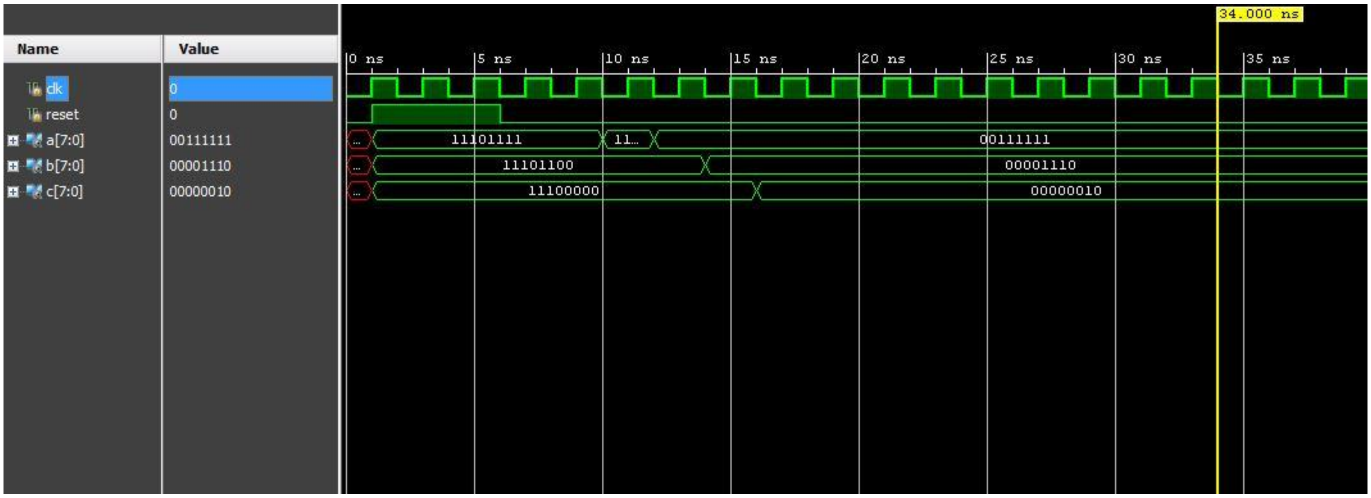
```
module processor(
    input clk,
    input reset,
    output [7:0] x,y,z
    );
    wire [7:0] instr;
    wire [1:0] func;
    wire [2:0] rdst,rsrc,rdst_2,imm_data_out;
    wire [7:0] reg1_data,reg2_data,write_data;
    wire [7:0] reg1_data_2,reg2_data_2;
    wire alu_control;
    wire write,status,status_2;
    instr_fetch a(clk,reset,instr);
    if_idreg b(clk,reset,instr,func,rdst,rsrc,status);
    regfile c(rdst,rsrc,reset,write,reg1_data,reg2_data,write_data,rdst_2,x,y,z);
    id_wbreg d(clk,reset,reg1_data,reg2_data,reg1_data_2,reg2_data_2,rdst,rsrc,rdst_2,imm_data_out,status,status_2);
    execute e(reg1_data_2,reg2_data_2,imm_data_out,alu_control,write_data);
    control_unit f(clk,func,alu_control,write,status,status_2,reset);

endmodule
```

5. **Test the processor design by initializing the instruction memory with a set of instructions (mentioned earlier) and register file with a set of values. List below the data you have filled your registers with.**

Answer: R0: 8'b11101100 R1: **8'b0** R2: 8'b11101100 R3: **8'b11100000** R4: 8'b11101111 R5: 8'b0 R6: 8'b0 R7: 8'b0

Verify if the register file is getting updated according to the set of instructions (mentioned earlier).

Copy verified **<u>Register file</u>** waveform here (show only the Registers that get updated, CLK, RESET):



Name	Value
clk	0
reset	0
a[7:0]	00111111
b[7:0]	00001110
c[7:0]	00000010

6. **Are there any data hazards for processor**
   Answer Yes or No: NO

Give detailed reasoning for your answer here:

Unrelated Questions

What were the problems you faced during the implementation of the processor?

Answer:

Did you implement the processor on your own? If you took help from someone whose help did you take? Which part of the design did you take help for?

Answer: nope, I took help from manoj. we had to discuss on how the data flow will occour and also looked some code on internet

Honor Code Declaration by student:

- My answers to the above questions are my own work.
- I have not shared the codes/answers written by me with any other students. (I might have helped clear doubts of other students).
- I have not copied other's code/answers to improve my results. (I might have got some doubts cleared from other students).

Name: AMAN NIDHI **Date:** 17/3/16
ID No.: 2013A3PS400H