BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI K. K. BIRLA Goa Campus
First Semester 2014-2015, CS F342 Computer Architecture, Lab – 4, 9th October 2014

Design and implement Pipeline Implementation with following specifications.

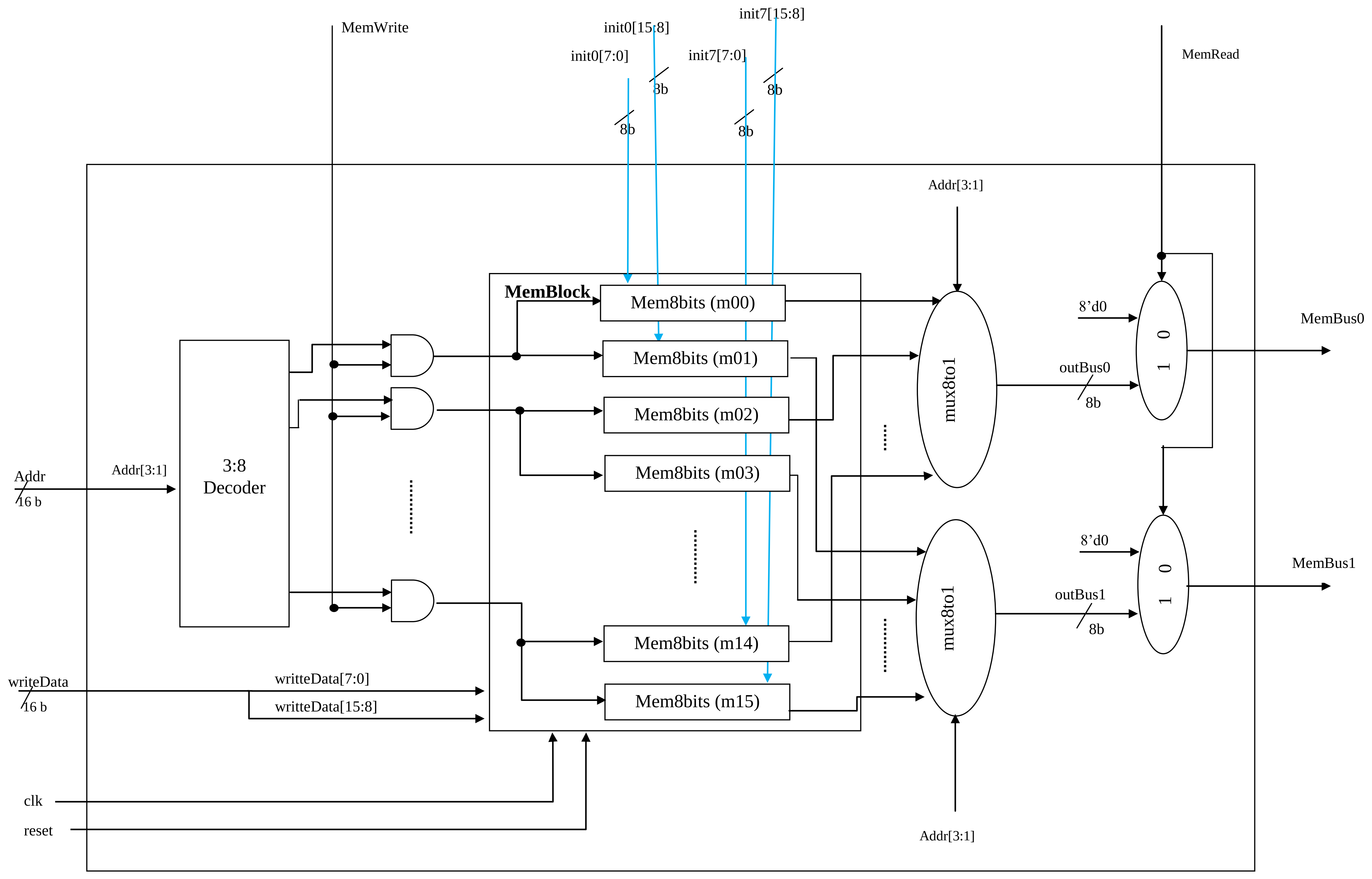
Instruction Format

Instruction	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Opcode															
lw	0	0	rs			rt			Imm							
sw	0	1	rs			rt			Imm							
addi	1	0	rs			rt			Imm							
add	1	1	rs			rt			rd			X	X	X	X	0
sub	1	1	rs			rt			rd			X	X	X	X	1

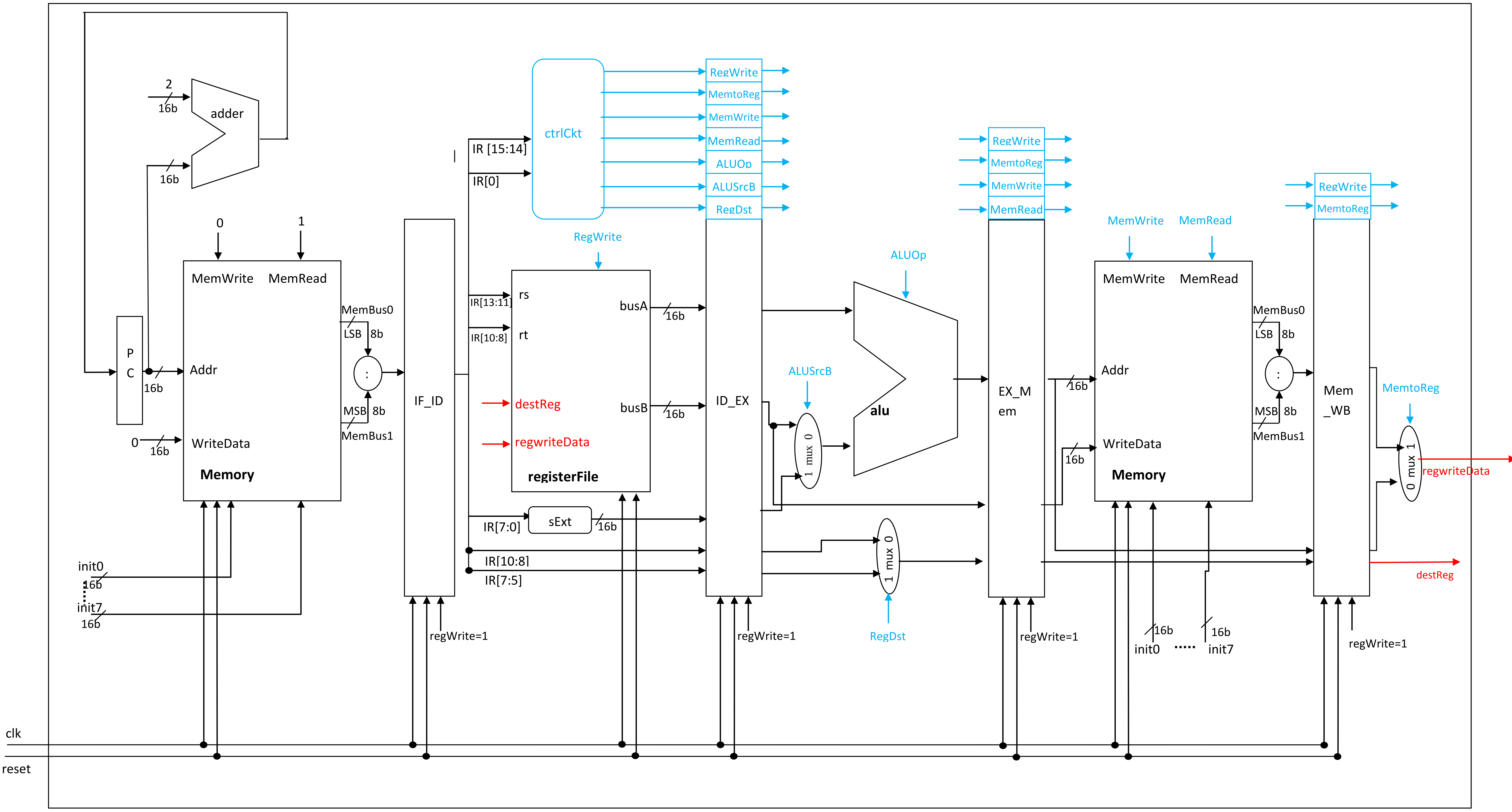
Control circuit

Instruction	**Opcode[15:14]**	**Opcode[0]**	**RegDst**	**ALUSrcB**	**ALUOp**	**MemRead**	**MemWrite**	**MemtoReg**	**RegWrite**
lw	00	X	0	1	0	1	0	1	1
sw	01	X	X	1	0	0	1	X	0
addi	10	X	0	1	0	0	0	0	1
add	11	0	1	0	0	0	0	0	1
sub	11	1	1	0	1	0	0	0	1

1. **module Memory(input clk, input reset, input [15:0] init0, init1, init2, init3, init4, init5, init6, init7, input MemWrite, input MemRead, input [15:0] Addr, input [15:0] writeData, output [7:0] MemBus0, output [7:0] MemBus1);**
2. **module Register16bit(input clk, input reset, input writeEn, input [15:0] writeData, output [15:0] outR);**
3. **module Register3bit(input clk, input reset, input writeEn, input [2:0] writeData, output [2:0] outR);**
4. **module Register1bit(input clk, input reset, input writeEn, input writeData, output outR);**
5. **module RegisterFile(input clk, input reset, input regWrite, input [2:0] srcRegA, input [2:0] srcRegB, input [2:0] destReg, input [15:0] writeData, output [15:0]outBusA, output [15:0] outBusB);**
6. **module IF_ID_PipelineRegister(input clk, input reset, input regWrite, input [15:0] instr, output [15:0] IR);**
7. **module ID_EX_PipelineRegister(input clk, input reset, input regWrite, input [15:0] BusA, input [15:0] BusB, input [15:0] SignExt8to16Imm, input [2:0] rd, input [2:0] rt, input ALUSrcB, input ALUOp, input RegDst, input MemRead, input MemWrite, input MemToReg, input RegWrite, output [15:0] P2BusA, output [15:0] P2BusB, output [15:0] P2SignExt8to16Imm, output [2:0] P2rd, output [2:0] P2rt, output P2ALUSrcB, output P2ALUOp, output P2RegDst, output P2MemRead, output P2MemWrite, output P2MemToReg, output P2RegWrite);**
8. **module EX_MEM_PipelineRegister(input clk, input reset, input regWrite, input [15:0] ALUOut, input [15:0] BusB, input [2:0] DstReg, input MemRead, input MemWrite, input MemToReg, input RegWrite,output [15:0] P3ALUOut, output [15:0] P3BusB, output [2:0] P3DstReg, output P3MemRead, output P3MemWrite, output P3MemToReg, output P3RegWrite);**
9. **module MEM_WB_PipelineRegister(input clk, input reset, input regWrite, input [15:0] ALUOut, input [15:0] DMOut, input [2:0] DstReg, input RegWrite, input MemToReg, output [15:0] P4ALUOut, output [15:0] P4DMOut, output [2:0] P4DstReg, output P4RegWrite, output P4MemToReg);**
10. **module ALU(input [15:0] in1, input [15:0] in2, input ALUOp, output reg [15:0] ALUOut);**
11. **module SignExt8to16(input [7:0] in, output reg [15:0] SignExtOut);**
12. **module Mux2to1_16bits(input [15:0] in1, input [15:0] in2, input Sel, output reg [15:0] MuxOut);**
13. **module Mux2to1_3bits(input [2:0] in1, input [2:0] in2, input Sel, output reg [2:0] MuxOut);**
14. **module CtrlCkt(input [1:0] opcode, input funcCode, output reg ALUSrcB, output reg ALUOp, output reg RegDst, output reg MemRead, output reg MemWrite, output reg MemToReg, output reg RegWrite);**
15. **module PCadder(input [15:0] in, output reg [15:0] out);**
16. **module TopModule(input clk, input reset, output [15:0] RegWriteData);**

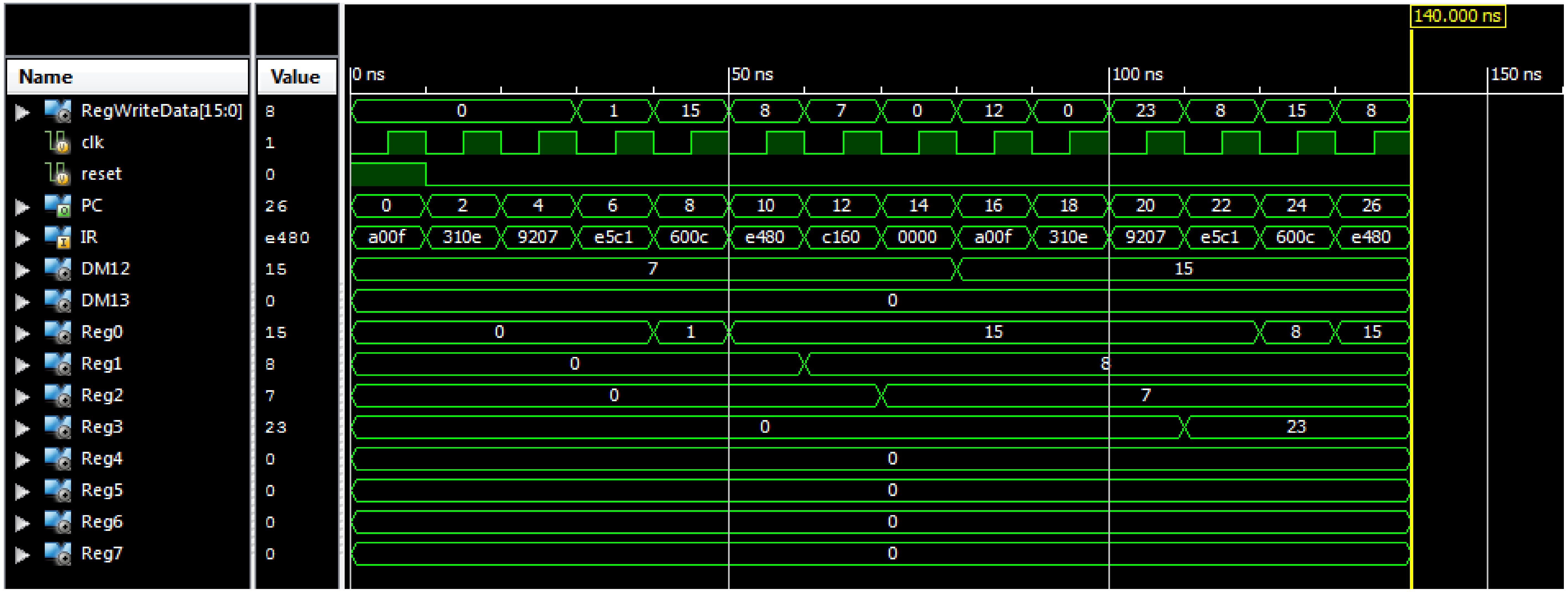


MemWrite
init0[15:8]
init0[7:0]
8b
8b
init7[15:8]
init7[7:0]
8b
8b
MemRead
Addr[3:1]
MemBlock
Mem8bits (m00)
Mem8bits (m01)
Mem8bits (m02)
Mem8bits (m03)
Mem8bits (m14)
Mem8bits (m15)
3:8
Decoder
Addr
16 b
Addr[3:1]
mux8to1
mux8to1
8'd0
outBus0
8b
0
1
MemBus0
8'd0
outBus1
8b
0
1
MemBus1
writeData
16 b
writteData[7:0]
writteData[15:8]
clk
reset
Addr[3:1]



adder
2
16b
16b
P
C
16b
0
1
MemWrite
MemRead
Addr
WriteData
Memory
0
16b
MemBus0
LSB
8b
MSB
8b
MemBus1
init0
16b
init7
16b
IF_ID
regWrite=1
IR [15:14]
IR[0]
ctrlCkt
RegWrite
MemtoReg
MemWrite
MemRead
ALUOp
ALUSrcB
RegDst
IR[13:11]
IR[10:8]
rs
rt
busA
busB
16b
16b
destReg
regwriteData
registerFile
sExt
IR[7:0]
16b
IR[10:8]
IR[7:5]
ID_EX
regWrite=1
ALUOp
alu
ALUSrcB
mux 0 1
mux 0 1
RegDst
EX_Mem
regWrite=1
MemWrite
MemRead
Addr
WriteData
Memory
16b
16b
MemBus0
LSB
8b
MSB
8b
MemBus1
16b
init0
16b
init7
Mem_WB
MemtoReg
mux 1 0
regwriteData
destReg
regWrite=1
clk
reset

```
module testBench;
        reg clk, reset;
        wire [15:0] RegWriteData;
        TopModule uut (.clk(clk), .reset(reset), .RegWriteData(RegWriteData));
        always
                #5 clk=~clk;
        initial
        begin
                clk = 0; reset = 1;
                #10 reset=0;
                #130 $finish;
        end
endmodule
```



140.000 ns
Name
Value
0 ns
50 ns
100 ns
150 ns
RegWriteData[15:0] 8
0 1 15 8 7 0 12 0 23 8 15 8
clk 1
reset 0
PC 26
0 2 4 6 8 10 12 14 16 18 20 22 24 26
IR e480
a00f 310e 9207 e5c1 600c e480 c160 0000 a00f 310e 9207 e5c1 600c e480
DM12 15
7 15
DM13 0
0
Reg0 15
0 1 15 8 15
Reg1 8
0 8
Reg2 7
0 7
Reg3 23
0 23
Reg4 0
0
Reg5 0
0
Reg6 0
0
Reg7 0
0