UM-SJTU JOINT INSTITUTE
INTRO TO COMPUTER ORGANIZATION
(VE 370)

INDIVIDUAL PROJECT REPORT

PROJECT 2

Name: **Weikai Zhou** ID: **518021911039**

Date: 12 Nov 2020

1. Objective

In the individual part of this project, we are going to implement the single cycle processor in Verilog so that it can support memory-reference instructions load word (`lw`) and store word (`sw`), arithmetic-logical instructions `add`, `addi`, `sub`, `and`, `andi`, `or`, and `slt`, jumping instructions branch equal (`beq`), branch not equal (`bne`), and jump (`j`).

2. Architecture of single cycle processor

The figure below shows the architecture of single cycle processor (Figure 1).

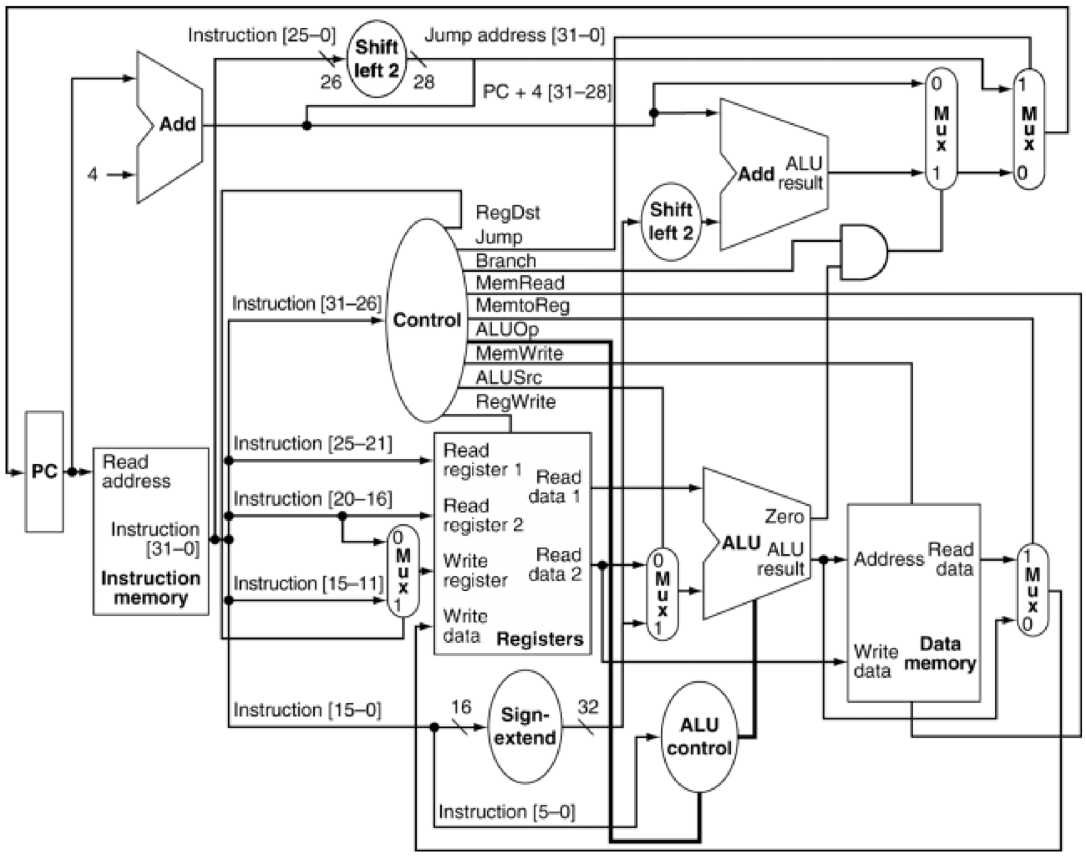


Figure 1. Architecture of single cycle processor

From the figure, we know that we can divide the processor into PC, Instruction Memory, Registers, Control Unit, Sign-Extend, ALU Control, ALU and Data Memory parts and connect them with wires. The detailed sourced code is in appendix.

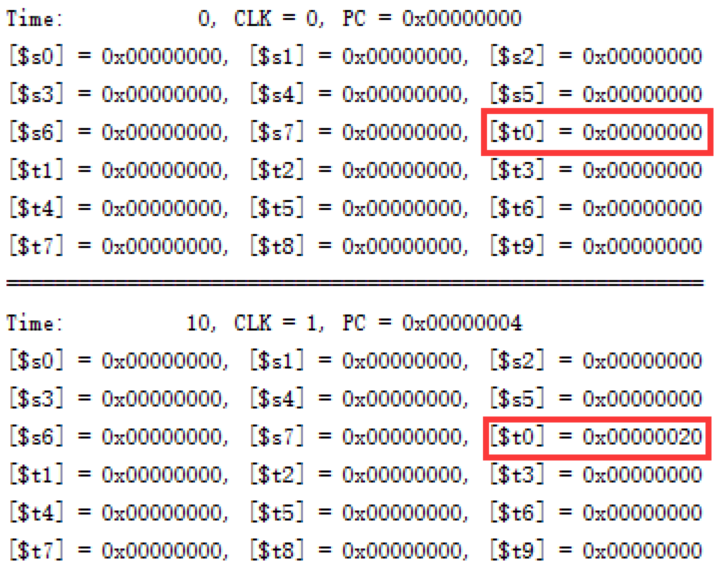
3. Simulation result

Since single cycle processor conducts one instruction in one clock time, there is no hazard we need to solve. Based on the “InstructionMem_for_P2_Demo.txt” provided by TAs, I write my own instructions, which is in appendix.

3.1. addi

The figure below shows the result of "`addi $t0, $zero, 0x20 (LOOP3)`" (Figure 2).

$$[\$t0] = [\$0] + 0x20 = 0x20$$



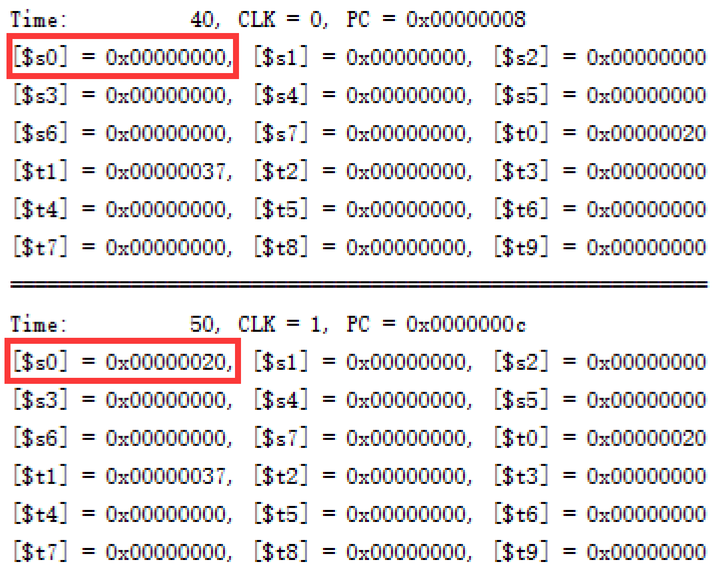
```
Time:          0, CLK = 0, PC = 0x00000000
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000000
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
===========================================================
Time:         10, CLK = 1, PC = 0x00000004
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
```

Figure 2. Result of `addi`

3.2. and

The figure below shows the result of "`and $s0, $t0, $t1`" (Figure 3).

$$[\$s0] = [\$t0] \& [\$t1] = 0x20 \& 0x37 = 0x20$$



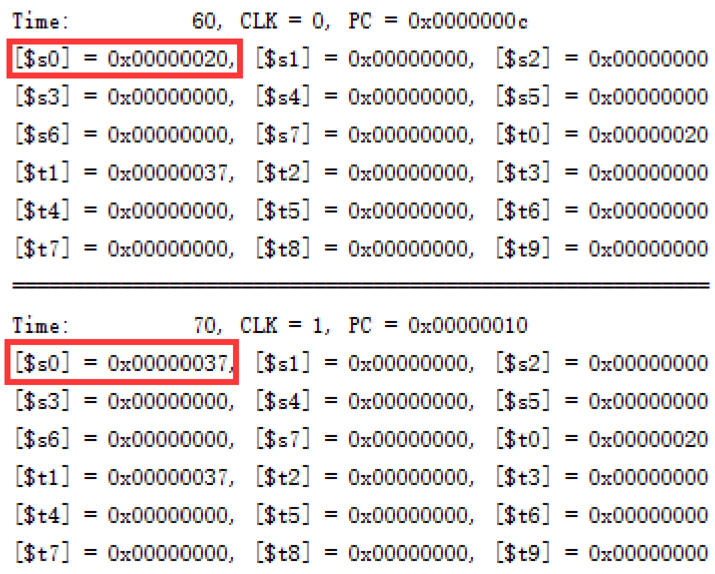
```
Time:         40, CLK = 0, PC = 0x00000008
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
===========================================================
Time:         50, CLK = 1, PC = 0x0000000c
[$s0] = 0x00000020, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
```

Figure 3. Result of `and`

3.3. `or`

The figure below shows the result of "`or $s0, $t0, $t1`" (Figure 4).

$$[\$s0] = [\$t0] \mid [\$t1] = 0x20 \mid 0x37 = 0x37$$



```
Time:          60, CLK = 0, PC = 0x0000000c
[$s0] = 0x00000020, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:          70, CLK = 1, PC = 0x00000010
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
```

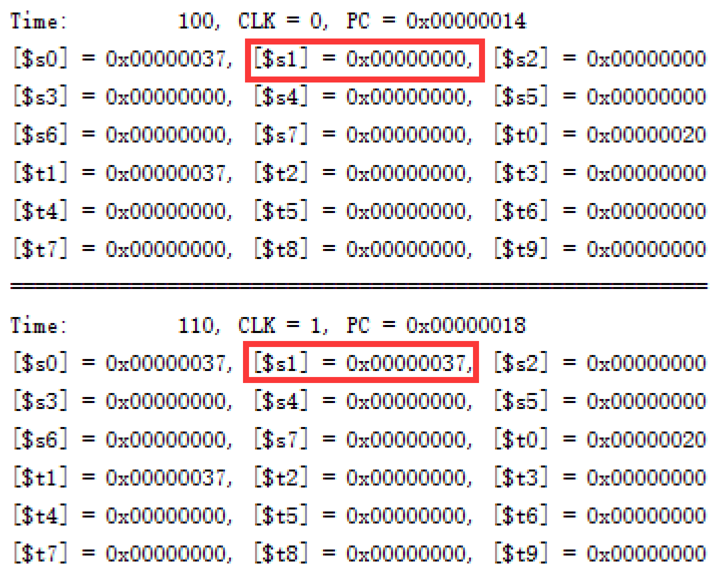
Figure 4. Result of `or`

3.4. `sw` & `lw`

For `sw` and `lw`, we use the following instructions.

```
sw $s0, 4($zero)
lw $s1, 4($zero)
```

Therefore, we can know that after running the code, the value in `$s1` should be the same as the value in `$s0`. The figure below shows the result (Figure 5).



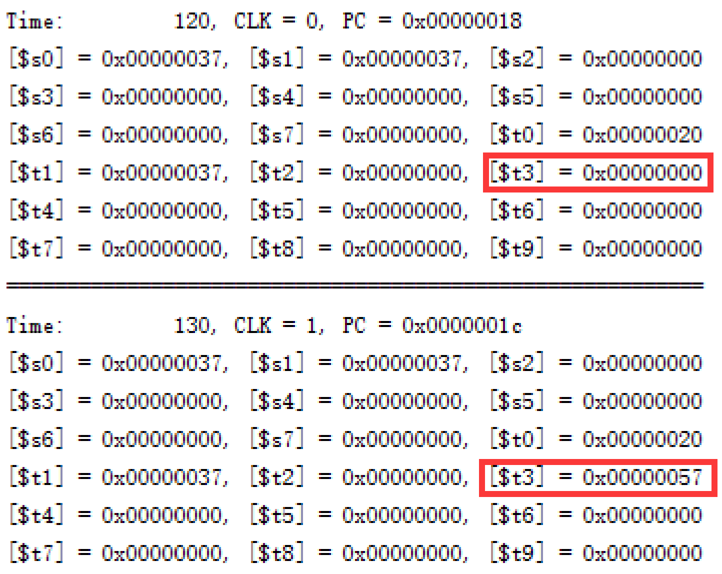
```
Time:         100, CLK = 0, PC = 0x00000014
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:         110, CLK = 1, PC = 0x00000018
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
```

Figure 5. Result of `sw` & `lw`

3.5. add

The figure below shows the result of "`add $t3, $t1, $t0`" (Figure 6).

$$[\$t3] = [\$t1] + [\$t0] = 0x37 + 0x20 = 0x57$$



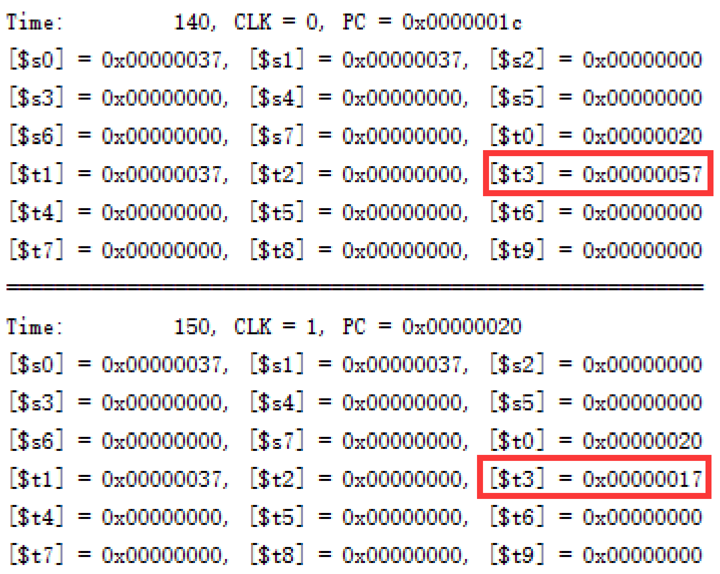
```
Time:         120, CLK = 0, PC = 0x00000018
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
===========================================================
Time:         130, CLK = 1, PC = 0x0000001c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000057
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
```

Figure 6. Result of `add`

3.6. sub

The figure below shows the result of "`sub $t3, $t1, $t0`" (Figure 7).

$$[\$t3] = [\$t1] - [\$t0] = 0x37 - 0x20 = 0x17$$



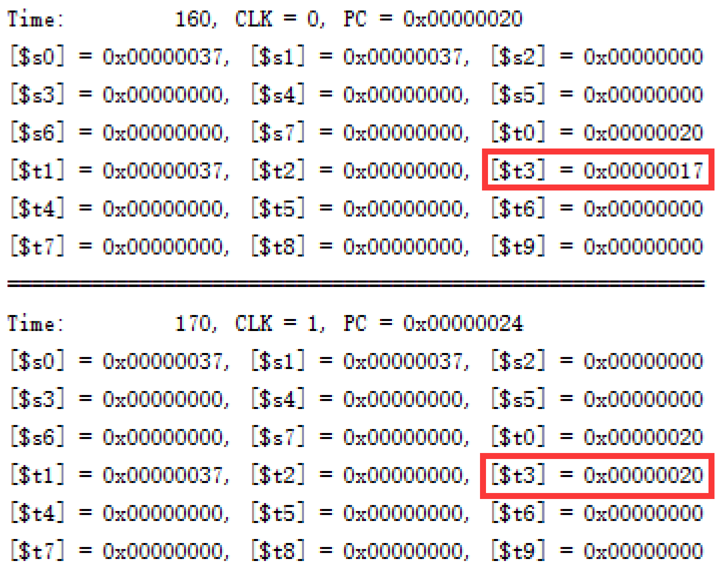
```
Time:         140, CLK = 0, PC = 0x0000001c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000057
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
===========================================================
Time:         150, CLK = 1, PC = 0x00000020
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000017
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
```

Figure 7. Result of `sub`

3.7. `andi`

The figure below shows the result of "`andi $t3, $t0, 0x37`" (Figure 8).

$$[\$t3] = [\$t0] \mathbin{\&} 0x37 = \mathrm{0x20} \mathbin{\&} \mathrm{0x37} = \mathrm{0x20}$$



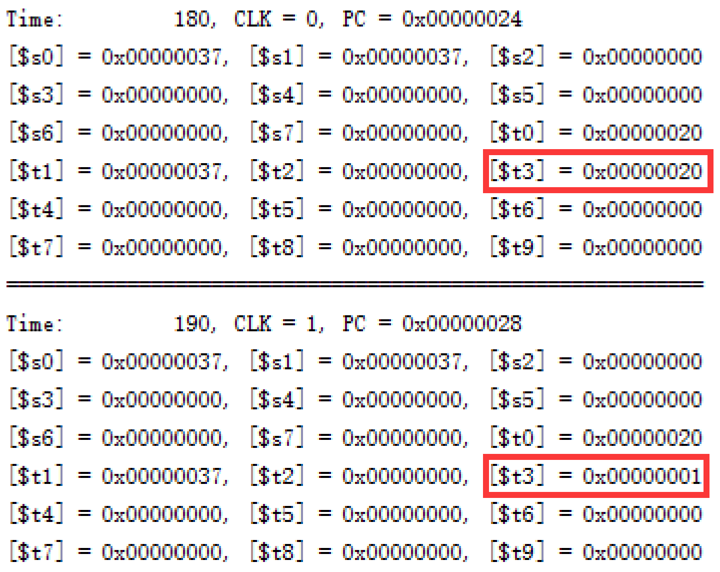
```
Time:          160,  CLK = 0,  PC = 0x00000020
[$s0] = 0x00000037,  [$s1] = 0x00000037,  [$s2] = 0x00000000
[$s3] = 0x00000000,  [$s4] = 0x00000000,  [$s5] = 0x00000000
[$s6] = 0x00000000,  [$s7] = 0x00000000,  [$t0] = 0x00000020
[$t1] = 0x00000037,  [$t2] = 0x00000000,  [$t3] = 0x00000017
[$t4] = 0x00000000,  [$t5] = 0x00000000,  [$t6] = 0x00000000
[$t7] = 0x00000000,  [$t8] = 0x00000000,  [$t9] = 0x00000000
==============================================================
Time:          170,  CLK = 1,  PC = 0x00000024
[$s0] = 0x00000037,  [$s1] = 0x00000037,  [$s2] = 0x00000000
[$s3] = 0x00000000,  [$s4] = 0x00000000,  [$s5] = 0x00000000
[$s6] = 0x00000000,  [$s7] = 0x00000000,  [$t0] = 0x00000020
[$t1] = 0x00000037,  [$t2] = 0x00000000,  [$t3] = 0x00000020
[$t4] = 0x00000000,  [$t5] = 0x00000000,  [$t6] = 0x00000000
[$t7] = 0x00000000,  [$t8] = 0x00000000,  [$t9] = 0x00000000
```

Figure 8. Result of `andi`

3.8. `slt`

The figure below shows the result of "`slt $t3, $t0, $t1`" (Figure 9).

$$[\$t3] = ([\$t0] < [\$t1])\ ?\ 1{:}\,0 = 1$$

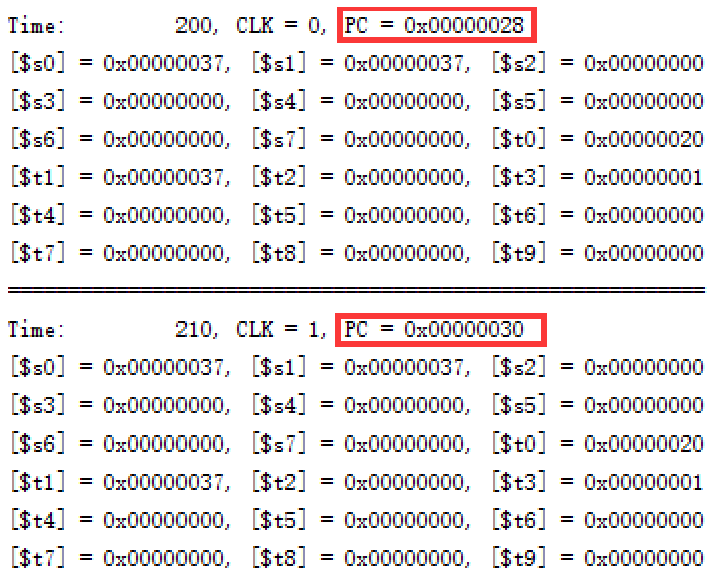


```
Time:          180,  CLK = 0,  PC = 0x00000024
[$s0] = 0x00000037,  [$s1] = 0x00000037,  [$s2] = 0x00000000
[$s3] = 0x00000000,  [$s4] = 0x00000000,  [$s5] = 0x00000000
[$s6] = 0x00000000,  [$s7] = 0x00000000,  [$t0] = 0x00000020
[$t1] = 0x00000037,  [$t2] = 0x00000000,  [$t3] = 0x00000020
[$t4] = 0x00000000,  [$t5] = 0x00000000,  [$t6] = 0x00000000
[$t7] = 0x00000000,  [$t8] = 0x00000000,  [$t9] = 0x00000000
==============================================================
Time:          190,  CLK = 1,  PC = 0x00000028
[$s0] = 0x00000037,  [$s1] = 0x00000037,  [$s2] = 0x00000000
[$s3] = 0x00000000,  [$s4] = 0x00000000,  [$s5] = 0x00000000
[$s6] = 0x00000000,  [$s7] = 0x00000000,  [$t0] = 0x00000020
[$t1] = 0x00000037,  [$t2] = 0x00000000,  [$t3] = 0x00000001
[$t4] = 0x00000000,  [$t5] = 0x00000000,  [$t6] = 0x00000000
[$t7] = 0x00000000,  [$t8] = 0x00000000,  [$t9] = 0x00000000
```

Figure 9. Result of `slt`

3.9. beq

The figure below shows the result of “`beq $t0, $t0, LOOP1`” (Figure 10). The PC jumps to where `LOOP1` locates.

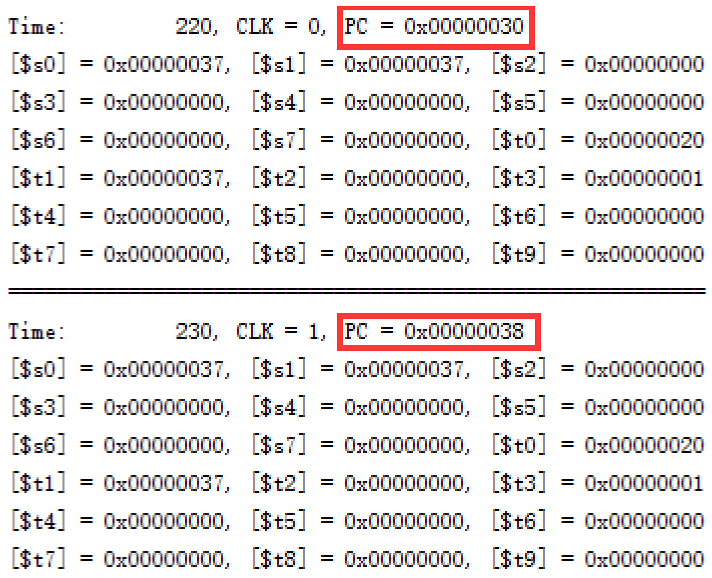


```
Time:          200, CLK = 0, PC = 0x00000028
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==============================================================
Time:          210, CLK = 1, PC = 0x00000030
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
```

Figure 10. Result of `beq`

3.10. bne

The figure below shows the result of “`bne $t0, $t1, LOOP2 (LOOP1)`” (Figure 11). The PC jumps to where `LOOP2` locates.

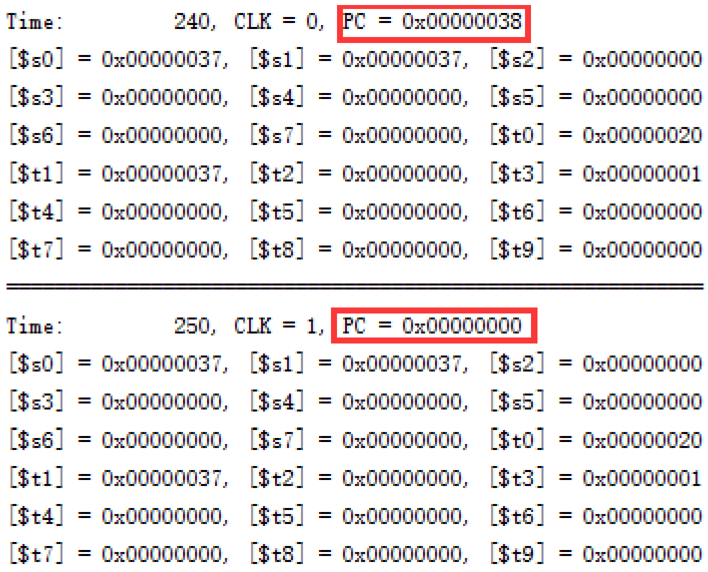


```
Time:          220, CLK = 0, PC = 0x00000030
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==============================================================
Time:          230, CLK = 1, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
```

Figure 11. Result of `bne`

3.11. `j`

The figure below shows the result of "`j LOOP3 (LOOP2)`" (Figure 12). The PC jumps to where `LOOP3` locates.



```
Time:          240, CLK = 0, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
=============================================================
Time:          250, CLK = 1, PC = 0x00000000
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
```

Figure 11. Result of `j`

Therefore, we know that all the instructions required has been achieved successfully.

4. Conclusion

In the individual part of this project, we have implemented the single cycle processor in Verilog so that it can support memory-reference instructions load word (`lw`) and store word (`sw`), arithmetic-logical instructions `add`, `addi`, `sub`, `and`, `andi`, `or`, and `slt`, jumping instructions branch equal (`beq`), branch not equal (`bne`), and jump (`j`).

From this project, we know that designing the processor from a larger perspective is important because the wire used to connect different components can be annoying.

5. Peer Evaluation

Name	Level of contribution	Description of contribution
Weikai Zhou (myself)	5	Writing pipelined processor (excluding forwarding, hazard detection), final debugging, writing report
Shengyuan Xu	5	Writing report
Yiwei Yang	5	Writing forwarding, hazard detection and report
Yuxuan Zhang	5	Writing code that connects pipelined processor with the FPGA, writing report

6. Appendix

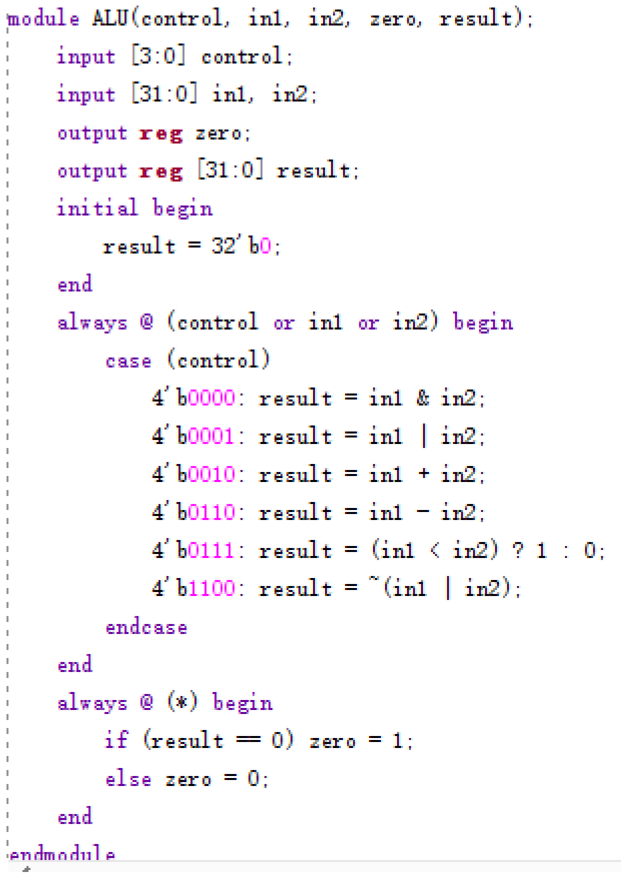
6.1. Test instructions

```
00100000 00001000 00000000 00100000 //addi $t0, $zero, 0x20 (LOOP3)
00100000 00001001 00000000 00110111 //addi $t1, $zero, 0x37
00000001 00001001 10000000 00100100 //and $s0, $t0, $t1
00000001 00001001 10000000 00100101 //or $s0, $t0, $t1
10101100 00010000 00000000 00000100 //sw $s0, 4($zero)
10001100 00010001 00000000 00000100 //lw $s1, 4($zero)
00000001 00101000 01011000 00100000 //add $t3, $t1, $t0
00000001 00101000 01011000 00100010 //sub $t3, $t1, $t0
00110001 00001011 00000000 00110111 //andi $t3, $t0, 0x37
00000001 00001001 01011000 00101010 //slt $t3, $t0, $t1
00010001 00001000 00000000 00000001 //beq $t0, $t0, LOOP1
00100000 00001001 00000000 00110111 //addi $t1, $zero, 0x37
00010101 00001001 00000000 00000001 //bne $t0, $t1, LOOP2 (LOOP1)
00100000 00001001 00000000 00110111 //addi $t1, $zero, 0x37
00001000 00000000 00000000 00000000 //j LOOP3 (LOOP2)
```

6.2. Single cycle

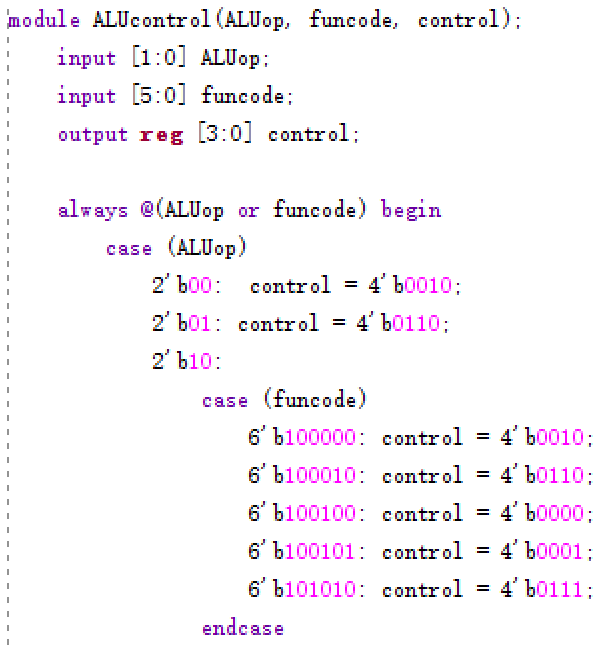
```
module singlecycle(clk);
    input clk;
    wire [31:0] instruction, pcin, pcout, pc4, jumpaddr, branchaddr, tempaddr,
                regreaddata1, regreaddata2, writedata, extend, ALUsec, ALUresult, memreaddata;
    wire [4:0] writereg;
    wire [3:0] ALUcontrol;
    wire [1:0] ALUop;
    wire regwrite, ALUSrc, branch, jump, beq, bne, regdst, memtoreg, memread, memwrite, ALUzero;

    PC pc (
        clk, pcin, pcout
    );

    assign pc4 = pcout + 4;

    InstructionMemory im (
        pcout, instruction
    );

    assign branchaddr = pc4 + (extend << 2);
    assign branch = (ALUzero & beq) | (~ALUzero & bne);
    MUX2_1 temppc(pc4, branchaddr, tempaddr, branch);
    assign jumpaddr = {pc4[31:28], instruction[25:0], 2'b0};
    MUX2_1 finalpc(tempaddr, jumpaddr, pcin, jump);

    Control  control(
        instruction[31:26],
        regdst, jump,
        beq, bne,
        memread, memwrite,
        memtoreg, ALUSrc,
        regwrite, ALUop
    );

    assign writereg = (regdst) ? instruction[15:11] : instruction[20:16];

    Register regs(
        clk, regwrite,
        instruction[25:21], instruction[20:16],
        writereg, writedata,
        regreaddata1, regreaddata2
    );

    SignExtend signextend(
        instruction[15:0], extend
    );

    ALUcontrol alucon(
        ALUop, instruction[5:0], ALUcontrol
    );

    MUX2_1 ALUMUX(regreaddata2, extend, ALUsec, ALUSrc);

    ALU alu(
        ALUcontrol,
        regreaddata1, ALUsec,
        ALUzero, ALUresult
    );

    MUX2_1 daALU(ALUresult, memreaddata, writedata, memtoreg);

    DataMemory dm(
        clk, ALUresult,
        memread, memwrite,
        regreaddata2, memreaddata
    );

endmodule
```

6.3. ALU



```
module ALU(control, in1, in2, zero, result);
    input [3:0] control;
    input [31:0] in1, in2;
    output reg zero;
    output reg [31:0] result;
    initial begin
        result = 32'b0;
    end
    always @ (control or in1 or in2) begin
        case (control)
            4'b0000: result = in1 & in2;
            4'b0001: result = in1 | in2;
            4'b0010: result = in1 + in2;
            4'b0110: result = in1 - in2;
            4'b0111: result = (in1 < in2) ? 1 : 0;
            4'b1100: result = ~(in1 | in2);
        endcase
    end
    always @ (*) begin
        if (result == 0) zero = 1;
        else zero = 0;
    end
endmodule
```

6.4. ALUControl



```
module ALUcontrol(ALUop, funcode, control);
    input [1:0] ALUop;
    input [5:0] funcode;
    output reg [3:0] control;

    always @(ALUop or funcode) begin
        case (ALUop)
            2'b00:  control = 4'b0010;
            2'b01: control = 4'b0110;
            2'b10:
                case (funcode)
                    6'b100000: control = 4'b0010;
                    6'b100010: control = 4'b0110;
                    6'b100100: control = 4'b0000;
                    6'b100101: control = 4'b0001;
                    6'b101010: control = 4'b0111;
                endcase
            2'b11: control = 4'b0000;
        endcase
    end

endmodule
```

6.5. Register

```
module Register(clk, regwrite, readreg1, readreg2, writereg, writedata, readdata1, readdata2);
    input clk, regwrite;
    input [4:0] readreg1, readreg2, writereg;
    input [31:0] writedata;
    output [31:0] readdata1, readdata2;
    reg [31:0] regs [0:31];
    integer i;

    initial begin
        for(i = 0 ; i < 32; i = i + 1) begin
            regs[i] = 32'b0;
        end
    end

    always @(posedge clk) begin
        if (regwrite) begin
             regs[writereg] = writedata;
        end
    end

    assign readdata1 = regs[readreg1];
    assign readdata2 = regs[readreg2];

endmodule
```

6.6. Control

```
module Control(opcode, regdst, jump, beq, bne, memread, memwrite, memtoreg, ALUSrc, regwrite, ALUop);
    input [5:0] opcode;
    output reg regdst, jump, beq, bne, memread, memtoreg, memwrite, ALUSrc, regwrite;
    output reg [1:0] ALUop;

    initial begin
        regdst = 1'b0;
        jump = 1'b0;
        beq = 1'b0;
        bne = 1'b0;
        memread = 1'b0;
        memtoreg = 1'b0;
        memwrite = 1'b0;
        ALUSrc = 1'b0;
        regwrite = 1'b0;
        ALUop = 2'b00;
    end

    always @ (opcode) begin
        case (opcode)
            6'b100011: begin // lw
                regdst = 1'b0;
                jump = 1'b0;
                beq = 1'b0;
                bne = 1'b0;
                memread = 1'b1;
                memtoreg = 1'b1;
                memwrite = 1'b0;
                ALUSrc = 1'b1;
                regwrite = 1'b1;
                ALUop = 2'b00;
            end
            6'b101011: begin // sw
                regdst = 1'b0;
                jump = 1'b0;
                beq = 1'b0;
                bne = 1'b0;
                memread = 1'b0;
                memtoreg = 1'b0;
                memwrite = 1'b1;
                ALUSrc = 1'b1;
                regwrite = 1'b0;
                ALUop = 2'b00;
            end

            6'b000000: begin // add, sub, and, or, slt
                regdst = 1'b1;
                jump = 1'b0;
                beq = 1'b0;
                bne = 1'b0;
                memread = 1'b0;
                memtoreg = 1'b0;
                memwrite = 1'b0;
                ALUSrc = 1'b0;
                regwrite = 1'b1;
                ALUop = 2'b10;
            end
            6'b001000: begin // addi
                regdst = 1'b0;
                jump = 1'b0;
                beq = 1'b0;
                bne = 1'b0;
        memread = 1'b0;
        memtoreg = 1'b0;
        memwrite = 1'b0;
        ALUSrc = 1'b1;
        regwrite = 1'b1;
        ALUop = 2'b00;
    end
    6'b001100: begin // andi
        regdst = 1'b0;
        jump = 1'b0;
        beq = 1'b0;
        bne = 1'b0;
        memread = 1'b0;
        memtoreg = 1'b0;
        memwrite = 1'b0;
        ALUSrc = 1'b1;
        regwrite = 1'b1;
        ALUop = 2'b11;
    end
    6'b000100: begin // beq
        regdst = 1'b1;
        jump = 1'b0;
        beq = 1'b1;
        bne = 1'b0;
        memread = 1'b0;
        memtoreg = 1'b0;
        memwrite = 1'b0;
        ALUSrc = 1'b0;
        regwrite = 1'b0;
        ALUop = 2'b01;
    end

    6'b000101: begin // bne
        regdst = 1'b1;
        jump = 1'b0;
        beq = 1'b0;
        bne = 1'b1;
        memread = 1'b0;
        memtoreg = 1'b0;
        memwrite = 1'b0;
        ALUSrc = 1'b0;
        regwrite = 1'b0;
        ALUop = 2'b01;
    end
    6'b000010: begin // j-type
        regdst = 1'b1;
        jump = 1'b1;
```

```
                    beq = 1'b0;
                    bne = 1'b0;
                    memread = 1'b0;
                    memtoreg = 1'b0;
                    memwrite = 1'b0;
                    ALUSrc = 1'b0;
                    regwrite = 1'b0;
                    ALUop = 2'b00;
                end
            endcase
        end

endmodule
```

6.7. InstructionMemory

```
module InstructionMemory(address, instruction);
    input [31:0] address;
    output [31:0] instruction;
    reg [31:0] memory [0:14];

    initial begin
        memory[0]=32'b00100000000010000000000000100000; //addi $t0, $zero, 0x20 (LOOP3)
        memory[1]=32'b00100000000010010000000000110111; //addi $t1, $zero, 0x37
        memory[2]=32'b00000001000010011000000000100100; //and $s0, $t0, $t1
        memory[3]=32'b00000001000010011000000000100101; //or $s0, $t0, $t1
        memory[4]=32'b10101100000100000000000000000100; //sw $s0, 4($zero)
        memory[5]=32'b10001100000100010000000000000100; //lw $s1, 4($zero)
        memory[6]=32'b00000001001010000101100000100000; //add $t3, $t1, $t0
        memory[7]=32'b00000001001010000101100000100010; //sub $t3, $t1, $t0
        memory[8]=32'b00110001000010110000000000110111; //andi $t3, $t0, 0x37
        memory[9]=32'b00000001000010010101100000101010; //slt $t3, $t0, $t1
        memory[10]=32'b00010001000010000000000000000001; //beq $t0, $t0, LOOP1
        memory[11]=32'b00100000000010010000000000110111; //addi $t1, $zero, 0x37
        memory[12]=32'b00010101000010010000000000000001; //bne $t0, $t1, LOOP2 (LOOP1)
        memory[13]=32'b00100000000010010000000000110111; //addi $t1, $zero, 0x37
        memory[14]=32'b00001000000000000000000000000000; //j LOOP3 (LOOP2)
    end

    assign instruction = memory[address >> 2];

endmodule
```

6.8. PC

```
module PC(clk, in, out);
    input clk;
    input [31:0] in;
    output reg [31:0] out;

    initial begin
        out = 32'b0;
    end

    always @ (posedge clk) begin
        out = in;
    end

endmodule
```

6.9. DataMemory

```
module DataMemory(clk, address, memread, memwrite, writedata, readdata);
    input clk;
    input [31:0] address;
    input memread, memwrite;
    input [31:0] writedata;
    output reg [31:0] readdata;
    reg [31:0] memory [0:31];
    integer i;

    initial begin
        for(i = 0; i < 32; i = i + 1)
            memory[i] = 32'b0;
    end

    always @ (posedge clk) begin
        if (memwrite) begin
             memory[address >> 2] = writedata;
        end
    end
    always @ (*) begin
        if (memread) begin
            readdata = memory[address >> 2];
        end
        else begin
            readdata = 32'b0;
        end
    end
endmodule
```

6.10. SignExtend

```
module SignExtend(in, out);
    input [15:0] in;
    output [31:0] out;

    assign out[15:0] = in[15:0];
    assign out[31:16] = (in[15]) ? 16'b1111111111111111 : 16'b0;

endmodule
```

6.11. MUX2_1

```
module MUX2_1(in1, in2, out, sel);
    input [31:0] in1, in2;
    input sel;
    output reg [31:0] out;

    always @ (*) begin
        if (sel) begin
            out = in2;
        end
        else begin
            out = in1;
        end
    end

endmodule
```

6.12. Testbench

```
module test();
    reg clk;
    integer t;
    singlecycle uut(.clk(clk));

    initial begin
        clk = 0;
        t = 0;
    end

   always #10 begin
      $display("==============================================================");
      $display("Time: %d, CLK = %d, PC = 0x%H", t, clk, uut.pcout);
      $display("[$s0] = 0x%H, [$s1] = 0x%H, [$s2] = 0x%H", uut.regs.regs[16], uut.regs.regs[17], uut.regs.regs[18]);
      $display("[$s3] = 0x%H, [$s4] = 0x%H, [$s5] = 0x%H", uut.regs.regs[19], uut.regs.regs[20], uut.regs.regs[21]);
      $display("[$s6] = 0x%H, [$s7] = 0x%H, [$t0] = 0x%H", uut.regs.regs[22], uut.regs.regs[23], uut.regs.regs[8]);
      $display("[$t1] = 0x%H, [$t2] = 0x%H, [$t3] = 0x%H", uut.regs.regs[9], uut.regs.regs[10], uut.regs.regs[11]);
      $display("[$t4] = 0x%H, [$t5] = 0x%H, [$t6] = 0x%H", uut.regs.regs[12], uut.regs.regs[13], uut.regs.regs[14]);
      $display("[$t7] = 0x%H, [$t8] = 0x%H, [$t9] = 0x%H", uut.regs.regs[15], uut.regs.regs[24], uut.regs.regs[25]);
      clk = ~clk;
      t = t + 10;
   end
   always #280 $stop;
endmodule
```

6.13. Simulation result

```
==========================================================
Time:           0, CLK = 0, PC = 0x00000000
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000000
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==========================================================
Time:          10, CLK = 1, PC = 0x00000004
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==========================================================
Time:          20, CLK = 0, PC = 0x00000004
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
```

```
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:          30, CLK = 1, PC = 0x00000008
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:          40, CLK = 0, PC = 0x00000008
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:          50, CLK = 1, PC = 0x0000000c
[$s0] = 0x00000020, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:          60, CLK = 0, PC = 0x0000000c
[$s0] = 0x00000020, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:          70, CLK = 1, PC = 0x00000010
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:         80, CLK = 0, PC = 0x00000010
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:         90, CLK = 1, PC = 0x00000014
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:        100, CLK = 0, PC = 0x00000014
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:        110, CLK = 1, PC = 0x00000018
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:        120, CLK = 0, PC = 0x00000018
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:        130, CLK = 1, PC = 0x0000001c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
```

```
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000057
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==========================================================
Time:        140, CLK = 0, PC = 0x0000001c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000057
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==========================================================
Time:        150, CLK = 1, PC = 0x00000020
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000017
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==========================================================
Time:        160, CLK = 0, PC = 0x00000020
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000017
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==========================================================
Time:        170, CLK = 1, PC = 0x00000024
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000020
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==========================================================
Time:        180, CLK = 0, PC = 0x00000024
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000020
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==========================================================
Time:        190, CLK = 1, PC = 0x00000028
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==========================================================
Time:        200, CLK = 0, PC = 0x00000028
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==========================================================
Time:        210, CLK = 1, PC = 0x00000030
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==========================================================
Time:        220, CLK = 0, PC = 0x00000030
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==========================================================
Time:        230, CLK = 1, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
==========================================================
Time:        240, CLK = 0, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
```

```
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:        250, CLK = 1, PC = 0x00000000
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
============================================================
Time:        260, CLK = 0, PC = 0x00000000
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000001
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
```