UM-SJTU JOINT INSTITUTE
Introduction to Computer Organization
(VE370)

PROJECT REPORT
PROJECT 2

Name: Wu Yuhao ID:517370910119 Group: 17
Name: Zhong Xiaoxue ID:517370910076
Name: Jin Zhejian ID:517370910167

Date: 7 November 2019

1 Project Description

Model both single cycle and pipelined implementation of MIPS computer in Verilog that support a subset of MIPS instruction set including:

· The memory-reference instructions load word (lw) and store word (sw)

· The arithmetic-logical instructions add, addi, sub, and, andi, or, and slt

· The jumping instructions branch equal (beq), branch not equal (bne), and jump (j)

Use Figure 1 as a top-level block diagram of your single cycle implementation and Figure 2 for the pipelined structure. Note: there may be some components and control signals omitted from the figures that you will have to add to support all instructions listed above. Forwarding should be implemented in the pipelined structure to handle data and control hazards. (source: Computer Organization and Design, by Patterson and Hennessy, Morgan Kaufmann Publishers).

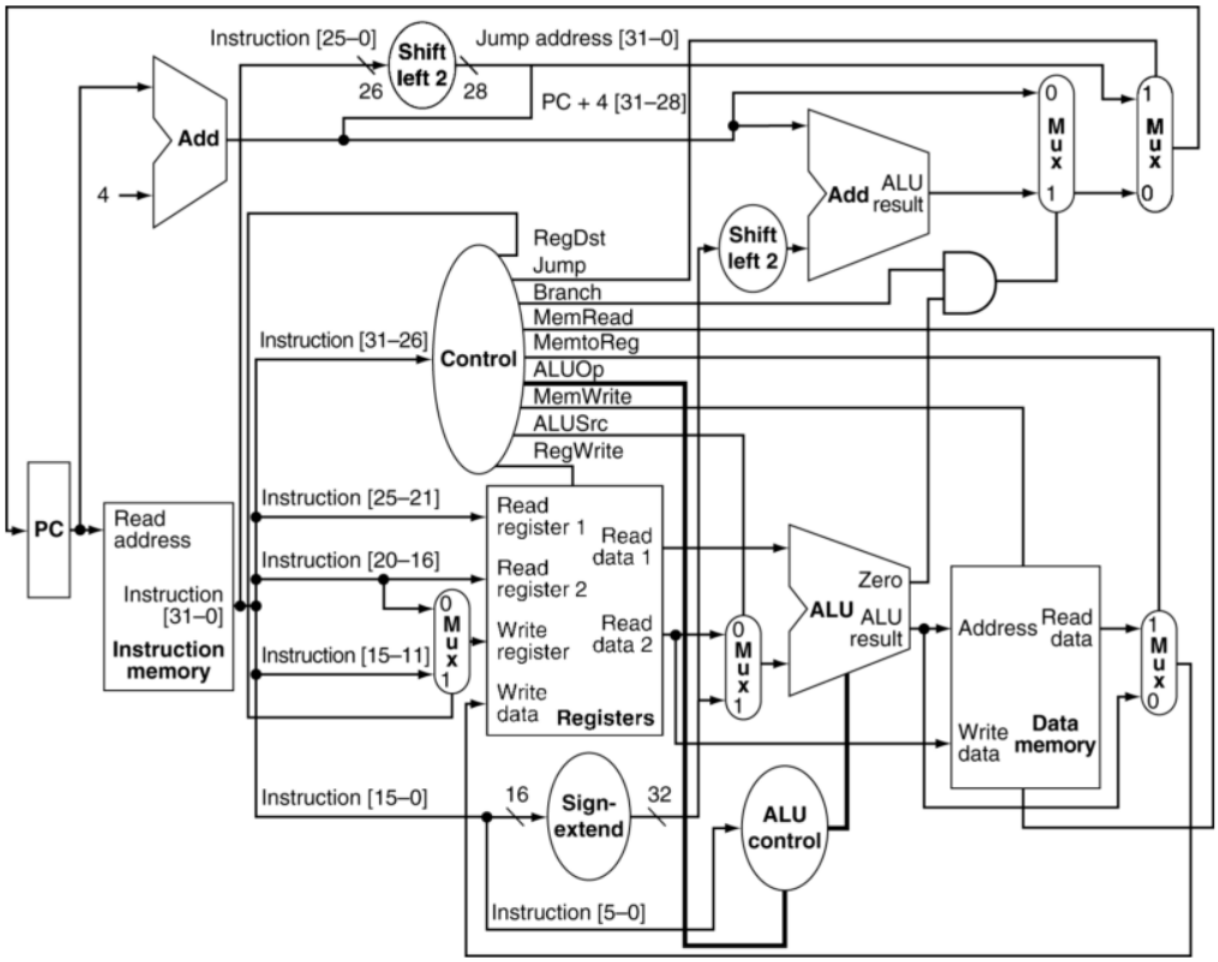

Figure 1: Single cycle implementation of MIPS architecture

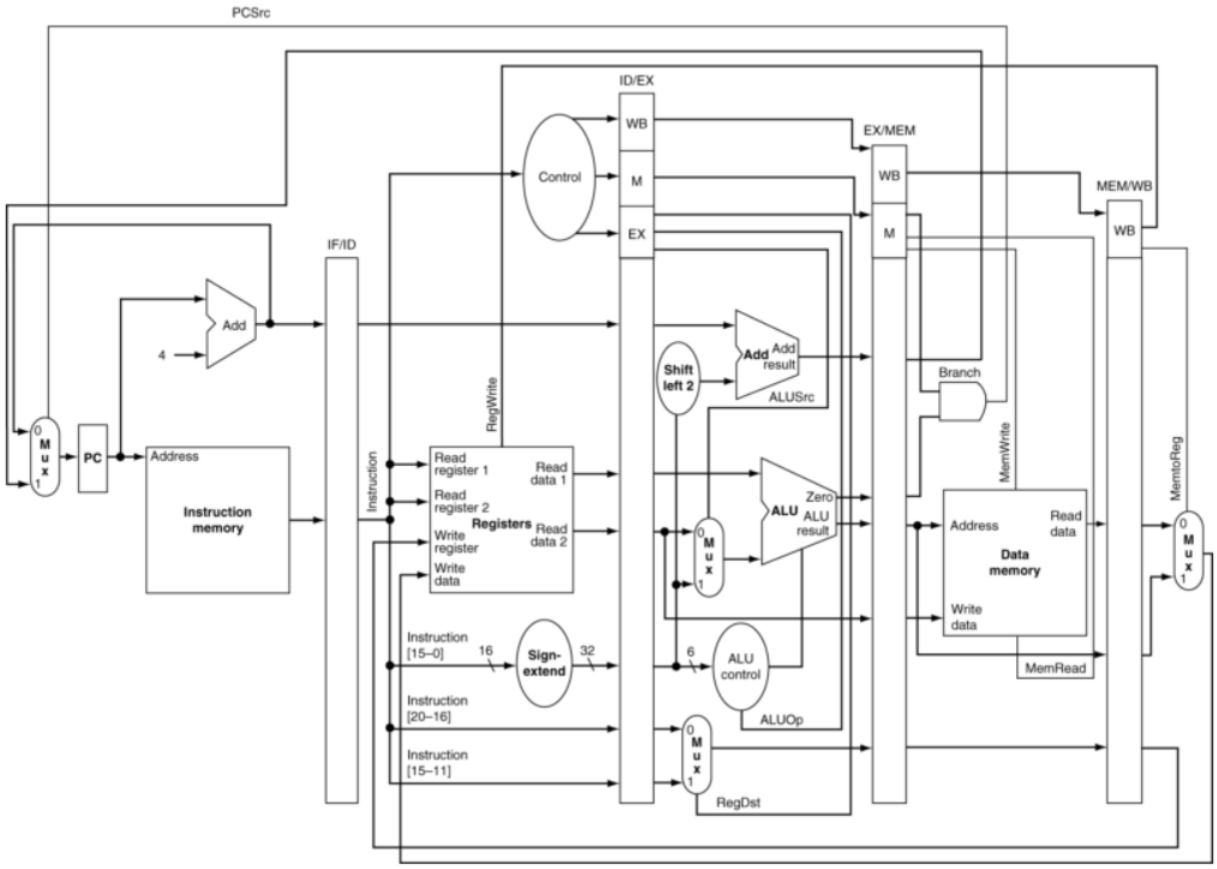

Figure 2: Pipelined implementation of MIPS architecture

2 Single-cycle

2.1 Verilog Code Realisation

The Verilog code with comments is given in the appendix. We divided the CPU into components of:

1. **Adder:** to add two numbers A and B. And we use adder in single-cycle processor to get next address (PC+4).

2. **Sign extension:** to extend a 16-bit input to a 32-bit output so that we can calculate the target address or use it as another input for ALU.

3. **Register:** to have an access to Register File so that content can be read from registers or written into registers.

4. **ALU control:** to output the 4-bit ALU control signal so that ALU can do the corresponding operation. With given ALUOp and function code, ALU control can give the 4-bit ALU control signal.

5. **ALU:** to do the corresponding operation with two 32-bit inputs and a 4-bit ALU control signal.

6. **Data Memory:** to have an access to Data Memory so that the processor can store data into data memory or read data from data memory.

7. **Instruction Memory:** to store instructions and with an input PC address, it can output the corresponding instruction.

8. **Control:** with the input instruction code, it can give the corresponding values of control signals that are needed in future stages (like MemRead, MemWrite, etc).

9. **Program Counter:** to have an access to PC register so that it can output the correct PC address in IF stage.

10. **N bit MUX:** to help choose the correct data depending on the control signal (used for branch instructions or choose rt/rd).

The main function “single_cycle” wired all components according to Figure 1 and realized the whole single cycle process.

2.2 Simulation Result and Discussion

For simulation, we use Vivado to write a simulation file ”test_single” which is shown below:

```
`timescale 1ns / 1ps

`include "single_cycle.v"

module single_cycle_tb;

    integer i = 0;

  // Inputs
  reg clk;

  // Instantiate the Unit Under Test (UUT)
  single_cycle uut (
    .clk(clk)
  );

  initial begin
    // Initialize Inputs
    clk = 0;
        $dumpfile("single_cycle.vcd");
        $dumpvars(1, uut);
        $display("Texual result of single cycle:");
        $display("================================================");
        #750;
        $stop;
  end

    always #10 begin
        $display("Time: %d, CLK = %d, PC = 0x%H", i, clk, uut.PC_out);
        $display("[$s0] = 0x%H, [$s1] = 0x%H, [$s2] = 0x%H",uut.get_reg.
    register_memory[16],
        uut.get_reg.register_memory[17],
        uut.get_reg.register_memory[18]);
        $display("[$s3] = 0x%H, [$s4] = 0x%H, [$s5] = 0x%H", uut.get_reg.
    register_memory[19], uut.get_reg.register_memory[20], uut.get_reg.
    register_memory[21]);
```

```
        $display("[$s6] = 0x%H, [$s7] = 0x%H, [$t0] = 0x%H", uut.get_reg.register_memory[22], uut.get_reg.register_memory[23], uut.get_reg.register_memory[8]);
        $display("[$t1] = 0x%H, [$t2] = 0x%H, [$t3] = 0x%H", uut.get_reg.register_memory[9], uut.get_reg.register_memory[10], uut.get_reg.register_memory[11]);
        $display("[$t4] = 0x%H, [$t5] = 0x%H, [$t6] = 0x%H", uut.get_reg.register_memory[12], uut.get_reg.register_memory[13], uut.get_reg.register_memory[14]);
        $display("[$t7] = 0x%H, [$t8] = 0x%H, [$t9] = 0x%H", uut.get_reg.register_memory[15], uut.get_reg.register_memory[24], uut.get_reg.register_memory[25]);
        clk = ~clk;
        if (~clk) begin
            i = i + 1;
            $display("================================================");
        end
    end
endmodule
```

The following is our simulation result for the single-cycle processor in Vivado. And from simulation, we can see that the single cycle processor successfully completed the instructions.

Simulation Result:

```
Texual result of single cycle:
================================================
Time:              0, CLK = 0, PC = 0xfffffffc
[$s0] = 0x00000000, [$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:              0, CLK = 1, 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:              1, CLK = 0, PC = 0x00000000
[$s0] = 0x00000000, [$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:              1, 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:              2, CLK = 0, PC = 0x00000004
[$s0] = 0x00000000, [$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:              2, 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:              3, CLK = 0, PC = 0x00000008
[$s0] = 0x00000020, [$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:              3, 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:              4, CLK = 0, PC = 0x0000000c
[$s0] = 0x00000037, [$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:              4, 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:              5, 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:              5, 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:              6, 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:              6, CLK = 1, PC = 0x00000018
[$s0] = 0x00000037, [$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:              7, CLK = 0, PC = 0x00000018
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0x00000000
[$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:              7, CLK = 1, PC = 0x0000001c
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0x00000000
[$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:              8, CLK = 0, PC = 0x0000001c
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$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:              8, CLK = 1, PC = 0x00000020
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$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:              9, CLK = 0, PC = 0x00000020
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$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:              9, CLK = 1, PC = 0x00000024
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$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
```

```
152 ================================================
153 Time:            10, CLK = 0, PC = 0x00000024
154 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
155 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
156 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
157 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
158 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
159 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
160 Time:            10, CLK = 1, PC = 0x00000028
161 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
162 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
163 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
164 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
165 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
166 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
167 ================================================
168 Time:            11, CLK = 0, PC = 0x00000028
169 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
170 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
171 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
172 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
173 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
174 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
175 Time:            11, CLK = 1, PC = 0x0000002c
176 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
177 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
178 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
179 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
180 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
181 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
182 ================================================
183 Time:            12, CLK = 0, PC = 0x0000002c
184 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
185 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
186 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
187 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
188 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
189 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
190 Time:            12, CLK = 1, PC = 0x00000030
191 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
192 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
193 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
194 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
195 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
196 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
197 ================================================
198 Time:            13, CLK = 0, PC = 0x00000030
199 [$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0xfffffff9
200 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
201 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
202 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
203 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
204 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
205 Time:            13, CLK = 1, PC = 0x00000034
206 [$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0xfffffff9
207 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
208 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
209 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
```

```
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:              14, CLK = 0, PC = 0x00000034
[$s0] = 0x00000037, [$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:              14, 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] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:              15, 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] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:              15, CLK = 1, PC = 0x0000003c
[$s0] = 0x00000037, [$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:              16, CLK = 0, PC = 0x0000003c
[$s0] = 0x00000037, [$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:              16, CLK = 1, PC = 0x00000040
[$s0] = 0x00000037, [$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:              17, CLK = 0, PC = 0x00000040
[$s0] = 0x00000037, [$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:              17, CLK = 1, PC = 0x00000044
[$s0] = 0x00000037, [$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:           18, CLK = 0, PC = 0x00000044
[$s0] = 0x00000037, [$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:           18, CLK = 1, PC = 0x00000048
[$s0] = 0x00000037, [$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:           19, CLK = 0, PC = 0x00000048
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:           19, CLK = 1, PC = 0x0000004c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:           20, CLK = 0, PC = 0x0000004c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:           20, CLK = 1, PC = 0x00000050
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:           21, CLK = 0, PC = 0x00000050
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:           21, CLK = 1, PC = 0x00000054
```

```
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:              22, CLK = 0, PC = 0x00000054
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:              22, CLK = 1, PC = 0x00000058
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:              23, CLK = 0, PC = 0x00000058
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:              23, CLK = 1, PC = 0x0000005c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:              24, CLK = 0, PC = 0x0000005c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$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:              24, CLK = 1, PC = 0x00000060
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$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:              25, CLK = 0, PC = 0x00000060
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
```

```
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:              25, CLK = 1, PC = 0x00000064
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:              26, CLK = 0, PC = 0x00000064
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:              26, CLK = 1, PC = 0x00000068
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:              27, CLK = 0, PC = 0x00000068
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:              27, CLK = 1, PC = 0x0000006c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:              28, CLK = 0, PC = 0x0000006c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:              28, CLK = 1, PC = 0x00000070
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:              29, CLK = 0, PC = 0x00000070
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
```

```
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:            29, CLK = 1, PC = 0x00000074
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:            30, CLK = 0, PC = 0x00000074
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:            30, CLK = 1, PC = 0x0000005c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:            31, CLK = 0, PC = 0x0000005c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:            31, CLK = 1, PC = 0x00000060
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
================================================
Time:            32, CLK = 0, PC = 0x00000060
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:            32, CLK = 1, PC = 0x00000064
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
================================================
Time:            33, CLK = 0, PC = 0x00000064
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
```

```
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:              33, CLK = 1, PC = 0x00000068
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
================================================
Time:              34, CLK = 0, PC = 0x00000068
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:              34, CLK = 1, PC = 0x0000006c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
================================================
Time:              35, CLK = 0, PC = 0x0000006c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:              35, CLK = 1, PC = 0x000000ac
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
================================================
Time:              36, CLK = 0, PC = 0x000000ac
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:              36, CLK = 1, PC = 0x000000b0
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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
================================================
```

3 Pipelined

3.1 Verilog Code Realisation

The Verilog code with comments is given in the appendix. For Pipelined Processor, the basic components are the same as the components for single-cycle process:

1. **Adder:** to add two numbers A and B. And we use adder in single-cycle processor to get next address (PC+4).

2. **Sign extension:** to extend a 16-bit input to a 32-bit output so that we can calculate the target address or use it as another input for ALU.

3. **Register:** to have an access to Register File so that content can be read from registers or written into registers.
   There is a change for pipeline processor: since we need to display the content in SSD, thus we add an extra input and output to obtain the content in corresponding register represented by the switched on FPGA board.

4. **ALU control:** to output the 4-bit ALU control signal so that ALU can do the corresponding operation. With given ALUOp and function code, ALU control can give the 4-bit ALU control signal.

5. **ALU:** to do the corresponding operation with two 32-bit inputs and a 4-bit ALU control signal.

6. **Data Memory:** to have an access to Data Memory so that the processor can store data into data memory or read data from data memory.

7. **Instruction Memory:** to store instructions and with an input PC address, it can output the corresponding instruction.

8. **Control:** with the input instruction code, it can give the corresponding values of control signals that are needed in future stages (like MemRead, MemWrite, etc).

9. **Program Counter:** to have an access to PC register so that it can output the correct PC address in IF stage.

10. **N bit MUX:** to help choose the correct data depending on the control signal (used for branch instructions or choose rt/rd).

And we add some additional components to deal with some data/control hazard:

1. **Forwarding:** to forward the correct data for operation in case of WAR (write after read)so that some cases of data hazard will be solved without adding bubbles. By comparison of rs, rt and rd in (ID for beq/bne) EX, MEM, WB stages, it will output a control signal for forward mux to choose correct data.

2. **Forward Mux:** to choose the correct data among three data with the control signal output from forwarding unit.

3. **Hazard Detection:** to detect control/data hazard in ID stage. And if hazard appears, a bubble will be inserted into ID stage.

4. **Pipe write:** to push the data/ control signals in one stage to the next future stage. (And for IF/ID register, a control signal IF_write or IF_Flush will control the component, if one of the two signal is zero, then the data pushed into ID stage will be all zero.)

5. **ID_EX Pipeline:** similar function as pipe_write, just to push all needed data/signal into next EX stage.

And to realize SSD drive, there are also some extra components added in the pipeline processor:

1. **Clock Divider:** to divide the input clock (from FPGA board) into a lower frequency clock. And the output clock will be used for SSD display.

2. **Ring Counter:** to change the digit on SSD periodically.

The main function "pipeline" wired all components according to Figure 2 and realized the whole process with given instructions. The RTL Implementation is shown as below:

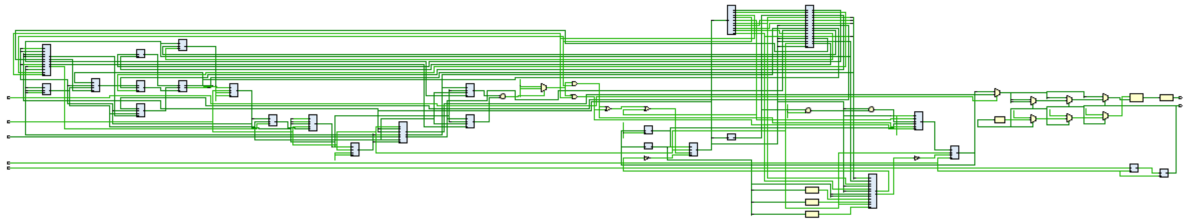

Figure 3: Pipelined RTL Implementation

3.2 Control and Data Hazard Detection and Solution

As data/control hazard is a critical problem for pipeline processor, in this part, we will explain our code for control/data hazard in detail.

In pipeline processors, there exist some hazard due to the adjacent instructions which may cause wrong outputs, such as (lw $t0, 0($t1), and $s0, $t0, $t2). To avoid such cases, the pipelined process needs a hazard detection in **ID** stage.

Our group write a hazard detection unit including both data and control hazard detection. (Note: the rule for hazard detection in our program is a little different from the method shown in the slide: if a hazard is detected, then we will insert bubble in ID stage instead of in EX stage.)

3.2.1 Hazard Detection

Pipeline_HazardDetection.v

```
module HazardDtection(
   //input IF_instruction,
    input clk,
    input bne,
    input branch,
    input jump,
```

```
    input ID_MemRead,
    input ID_RegWrite,
    input EX_MemRead,
    input [4:0] EX_Rt,
    input [4:0] ID_Rt,
    input [4:0] ID_Rd,
    input [4:0] IF_Rs,
    input [4:0] IF_Rt,
    output IF_Write,
    output PC_Write
    );

reg PCtem;
        reg IFIDtem;

        initial begin
            PCtem<=1;
            IFIDtem<=1;
        end
```

Output **PC_Write** and **IF_Write** will control the PC Register and IF/IF pipeline register. If **PC_Write=0**, the PC address will be hold, or it will load next PC address.

And if **IF_Write=1**, the IF/ID pipeline register will push data in IF stage into ID stage, otherwise, the instruction pushed into ID stage is 0x00000000, which indicates a bubble inserted into ID stage. (PC_Write and IF_Write always happen at the same time.)

1. Case 1: lw in ID stage, and destination register of lw is also the source register for instruction in IF stage. (e.g.:lw $t0, 0($t1), add $s0, $t0, $t2)

```
        always@(posedge clk)
        begin
        if (ID_MemRead && ((ID_Rt==IF_Rs) | (ID_Rt==IF_Rt))&&(branch
    ==0)&&(jump==0)&&(bne==0))
        begin
            PCtem<=0;
            IFIDtem<=0;
        end
```

2. Case 2: beq or bne in IF stage, and the destination register of lw instruction in EX stage is also a source register for beq/bne in ID stage. (e.g.:add $t0, $t1; $t2, addi $t3, $t3, 0; beq $t0, $t1, EXIT)

   Since beq/bne should be solved in ID stage, and the needed register (MEM stage now)is still not be overwritten.

```
        else if ((bne||branch) && EX_MemRead &&((EX_Rt==IF_Rs) | (
    EX_Rt==IF_Rt))&&(jump==0))
        begin
            PCtem<=0;
            IFIDtem<=0;
        end
```

3. Case 3: beq or bne in IF stage, and the destination register of instruction in ID stage is also a source register for beq/bne in ID stage. (e.g.:add $t0, $t1; beq $t0, $t1, EXIT)

   And for I-type instruction, destination register is ID_rt and for R-type, destination register is ID_rd.

```
        //I/lw and adjacent beq
        else if((bne||branch) && ID_RegWrite&&  ((ID_Rt==IF_Rs&& IF_Rs
    !=0)|(ID_Rt==IF_Rt&& IF_Rt!=0))&&(jump==0))
        begin
            PCtem<=0;
            IFIDtem<=0;
        end
        // r-type and adjacent beq
        else if((bne||branch) && ID_RegWrite && ((ID_Rd==IF_Rs&&IF_Rs
    !=0)|(ID_Rd==IF_Rt&& IF_Rt!=0))&&(jump==0))
           begin
               PCtem<=0;
               IFIDtem<=0;
           end
        else
```

4. Case 4: Without any hazard, then PC_Write = 1 and IF_Write = 1.

```
        begin
            PCtem<=1;
            IFIDtem<=1;
        end
        end
        assign IF_Write=IFIDtem;
        assign PC_Write=PCtem;
        endmodule
```

3.2.2 Hazard Solution: Forwarding

To solve some cases of data hazard, we can use forwarding unit. Thus, these cases of data hazard are not included in hazard detection part. Pipeline_forwarding.v

```
      module forwarding(
    input beq,
    input bne,
    input [4:0] ID_rs,
    input [4:0] ID_rt,
    input EM_RW,
    input MW_RW,
    input MEM_MemWrite,
    input [4:0] EM_Rd,
    input [4:0] IE_Rs,
    input [4:0] IE_Rt,
    input [4:0] MW_Rd,
    output [1:0] beqA,
    output [1:0] beqB,
    output  [1:0] ForwardA,
    output  [1:0] ForwardB,
    output sw_src

    );

reg [1:0] tmpA,tmp_beqA;
reg [1:0] tmpB, tmp_beqB;
reg temp_sw_src;
initial begin
    tmp_beqA<=0;
    tmp_beqB<=0;
```

```
  tmpA<=0;
  tmpB<=0;
  temp_sw_src<=0;
end
```

ForwardA and ForwardB are the control signals for the muxes to choose inputs of ALU in EX stage. beqA and beqB are the control signals of the muxes to choose data for comparison. And sw_src is the control signal to choose the store data in Data Memory for instruction sw.

1. Case 1: Content in destination register for instruction in EX/MEM stage is a source register for beq/bne. (e.g: add $t0, $t1, $t2; beq $s0, $t0, EXIT).

```
always@(*)
begin
if(EM_RW && (EM_Rd!=0) && (EM_Rd==ID_rs)&& (bne||beq))
  begin
    tmp_beqA<=2'b10;
  end
else if((bne||beq)&&(MW_RW && (MW_Rd!=0) && (MW_Rd==ID_rs))&& ~(EM_RW
   && (EM_Rd!=0) && (EM_Rd==IE_Rs)))
  begin
    tmp_beqA<=2'b01;
  end
else
  begin
    tmp_beqA<=2'b00;
  end
if(EM_RW && (EM_Rd!=0) && (EM_Rd==ID_rt)&& (bne||beq))
        begin
              tmp_beqB<=2'b10;
        end
    else if((bne||beq)&&(MW_RW && (MW_Rd!=0) && (MW_Rd==ID_rt))&& ~(
   EM_RW && (EM_Rd!=0) && (EM_Rd==IE_Rs)))
        begin
              tmp_beqB<=2'b01;
        end
    else
        begin
              tmp_beqB<=2'b00;
        end
```

2. Case 2: Due to the different condition for readReg and writeReg, it may be possible that WAR (write after read) will happen, which will cause an incorrect data to be stored in Data Memory. And to avoid such case, we need to use a signal to select the correct data for sw.

```
if(MEM_MemWrite && MW_RW &&(EM_Rd==MW_Rd))
  begin
  temp_sw_src<=1'b1;
  end
else
begin
 temp_sw_src<=1'b0;
end
```

3. Case 3: Content in destination register for instruction in MEM/WB stage is input for ALU (EX stage). (e.g: add $t0, $t1, $t2; sub $s0, $t0, $t1).

```
if(EM_RW && (EM_Rd!=0) && (EM_Rd==IE_Rs))
  begin
    tmpA<=2'b10;
  end
else if((MW_RW && (MW_Rd!=0) && (MW_Rd==IE_Rs))&& ~(EM_RW && (EM_Rd
    !=0) && (EM_Rd==IE_Rs)))
  begin
    tmpA<=2'b01;
  end
else
  begin
    tmpA<=2'b00;
  end

if(EM_RW && (EM_Rd!=0) && (EM_Rd==IE_Rt))
  begin
    tmpB<=2'b10;
  end
else if((MW_RW && (MW_Rd!=0) && (MW_Rd==IE_Rt))&& ~(EM_RW && (EM_Rd
    !=0) && (EM_Rd==IE_Rt)))
  begin
    tmpB<=2'b01;
  end
else
  begin
    tmpB<=2'b00;
  end
end

assign beqA=tmp_beqA;
assign beqB=tmp_beqB;
assign ForwardA=tmpA;
assign ForwardB=tmpB;
assign sw_src=temp_sw_src;
endmodule
```

3.3 Simulation Result and Discussion

We used Vivado to write a simulation file which is shown below:

```
`timescale 1ns / 1ps
module test_pipe;
parameter half_period = 1;
  reg  clk,reset=0;
  pipeline pipeline1(clk,reset);
  initial begin
  clk<=1'b0;
  end

  always
  begin
   #half_period clk = ~clk;
      $display("Time: %d,  ClK = %b,  PC = 0x%h,", $time, clk, test_pipe.
   pipeline1.PC_out);
      $display("[$s0] = 0x%h, [$s1] = 0x%h, [$s2] = 0x%h",
                        test_pipe.pipeline1.get_reg.register_memory[16],
                       test_pipe.pipeline1.get_reg.register_memory[17],
                       test_pipe.pipeline1.get_reg.register_memory[18]);
```

```
18      $display("[$s3] = 0x%h, [$s4] = 0x%h, [$s5] = 0x%h",  test_pipe.
    pipeline1.get_reg.register_memory[19],
19                                          test_pipe.pipeline1.get_reg.
    register_memory[20],
20                                          test_pipe.pipeline1.get_reg.
    register_memory[21]);
21     $display("[$s6] = 0x%h, [$s7] = 0x%h, [$t0] = 0x%h",  test_pipe.
    pipeline1.get_reg.register_memory[22],
22                                                          test_pipe.
    pipeline1.get_reg.register_memory[23],
23                                                          test_pipe.
    pipeline1.get_reg.register_memory[8]);
24     $display("[$t1] = 0x%h, [$t2] = 0x%h, [$t3] = 0x%h", test_pipe.
    pipeline1.get_reg.register_memory[9],
25
    test_pipe.pipeline1.get_reg.register_memory[10],
26
    test_pipe.pipeline1.get_reg.register_memory[11]);
27      $display("[$t4] = 0x%h, [$t5] = 0x%h, [$t6] = 0x%h", test_pipe.
    pipeline1.get_reg.register_memory[12],
28                      test_pipe.pipeline1.get_reg.register_memory[13],
29                    test_pipe.pipeline1.get_reg.register_memory[14]);
30     $display("[$t7] = 0x%h, [$t8] = 0x%h, [$t9] = 0x%h",  test_pipe.
    pipeline1.get_reg.register_memory[15],
31
                  test_pipe.pipeline1.get_reg.register_memory[24],
32
                  test_pipe.pipeline1.get_reg.register_memory[25]);
33  if (~clk) $display("
    ================================================================");
34    end
35    initial begin
36    $display("================================================================"
      );
37    #41 $stop;
38    end
39  endmodule
```

The following is the simulation result of the pipelined processor. We can see that the simulation result is the same as single-cycle processor, but takes much shorter time.

4 SSD Realization

For demonstration on FPGA board, we also write the code for SSD driver in the pipeline processor as a part in pipeline.v:

```
1  module pipeline(
2  clk,reset,C,AN,clock,reg_switch,PC_switch
3      );
4      input clk,reset,clock,PC_switch;
5      input [4:0] reg_switch;
6      output reg [6:0] C;
7      output [3:0] AN;
8      wire [31:0] Q2;
9      initial begin
10             C=7'b1111111;
11     end
```

```
 // code for pipeline processor is omitted.

 Registers get_reg(
    .read_register_1(ID_instruction[25:21]),
    .read_register_2(ID_instruction[20:16]),
    .switch(reg_switch),
    .regWrite(WB_RegWrite),
    .write_register(WB_rd),
    .write_data(write_data),
    .read_data_1(ID_read1),
    .read_data_2(ID_read2),
    .clk(clk),
    .Q2(Q2)
);
//SSD
   wire clock_d;
   reg [3:0] Q;

   clock_divider clock1(clock_d, clock, reset);

   ring_counter ring(AN, clock_d, reset);

 wire [31:0] ssd_out;
   // here only needs to use 4 hexo numbers because first 4 numbers must be
    0 for the small instruction value
assign ssd_out = (PC_switch)? PC_out:Q2;

   always @ (*) begin
       if (AN == 4'b1110) Q <= ssd_out[3:0];
       else if (AN == 4'b1101) Q <= ssd_out[7:4];
       else if (AN == 4'b1011) Q <= ssd_out[11:8];
       else if (AN == 4'b0111) Q <= ssd_out[15:12];
   end
   // ssd numbers
  always @ (*) begin
       case (Q)
           4'h0: C = 7'b1000000;
           4'h1: C = 7'b1111001;
           4'h2: C = 7'b0100100;
           4'h3: C = 7'b0110000;
           4'h4: C = 7'b0011001;
           4'h5: C = 7'b0010010;
           4'h6: C = 7'b0000010;
           4'h7: C = 7'b1111000;
           4'h8: C = 7'b0000000;
           4'h9: C = 7'b0010000;
           4'ha: C = 7'b0001000;
           4'hb: C = 7'b0000011;
           4'hc: C = 7'b1000110;
           4'hd: C = 7'b0100001;
           4'he: C = 7'b0000110;
           4'hf: C = 7'b0001110;
           default: C = 7'b1111111;
       endcase
   end

endmodule
```

And the following lists the usage of every variable:

- **clk:** a clock control for the pipeline processor;
- **reset:** if reset = 1, then PC will be reset to zero, and pipeline processor begins at the first instruction;
- **C:** a 7-bit output to determine the number shown on SSD;
- **AN:** a 4-bit output to control whether digit to shown for SSD on FPGA board;
- **clock:** a clock from FPGA board for SSD display (and will be divided into a lower frequency);
- **reg_switch:** a 5-bit input to determine which register to be displayed by SSD;
- **PC_switch:** if PC_switch = 1, then SSD will show the PC address, otherwise, it will display contents of registers;
- **Q2:** a 31-bit wire to save the content in the corresponding register represented by reg_switch;
- **ssd_out:** a 31-bit wire to save the content to be displayed by SSD (whether it saves the PC address or the content in register depending on the value of PC_switch).

And in order to get the content in the register represented by reg_switch, we change a bit in Registers.v and add an extra input (reg_switch) and output (Q2) so that we can have an access to Register File.

5 Appendix

5.1 Single-cycle

single_cycle.v

```
`ifndef MODULE_SINGLE_CYCLE
`define MODULE_SINGLE_CYCLE
`include "adder.v"
`include "ALU_control.v"
`include "ALU.v"
`include "control.v"
`include "data_memory.v"
`include "instruction_memory.v"
`include "program_counter.v"
`include "register.v"
`include "sign_extension.v"
`include "Mux_N_bit.v"

module single_cycle(clk,reset);
    input clk;
    input reset;
    // Input PC Signals
    wire [31:0] PC_in, PC_out, PC_plus_four;
    // Instructions memory signals
    wire [31:0] Instructions;
    // Regsiter signals
    wire [31:0] write_data,reg_read_data_1,reg_read_data_2;
    wire [4:0] write_address;
```

```
    // ALU
    wire [31:0] ALU_a,ALU_b,ALU_out;
    wire [3:0] ALU_control;
    wire output_zero;
    assign ALU_a = reg_read_data_1;
    // D-mem
    wire [31:0] D_MEM_read_data;
    // Control
  wire RegDst, Jump, Branch, MemRead, MemtoReg, MemWrite, ALUSrc, RegWrite;
  wire [1:0] ALUOp;
  wire [3:0] ALU_control_out;
    // Branch
    wire [31:0] extended_imm,shifted_imm,branch_result,branch_out;
    wire branch_mux;
    assign branch_mux = output_zero & Branch;
    // Jump
    wire [31:0] jump_addr;
    // PC + 4[31:28]
    assign jump_addr = {PC_plus_four[31:28],Instructions[25:0],2'b0};
    assign branch_out = PC_plus_four + (extended_imm<<2);
    // Connect PC
    program_counter PC(
        .PC_in(PC_in),
        .PC_out(PC_out),
        .reset(reset),
        .clk(clk)
    );
    // Calculate PC + 4
    adder add_PC_by_four(
        .a(PC_out),
        .b(32'b0100),
        .sum(PC_plus_four)
    );
    // Get instruction
    instruction_memory get_ins(
        .address(PC_out),
        .instruction(Instructions)
    );
    // Read Register
    Registers get_reg(
        .read_register_1(Instructions[25:21]),
        .read_register_2(Instructions[20:16]),
        .regWrite(RegWrite),
        .write_register(write_address),
        .write_data(write_data),
        .read_data_1(reg_read_data_1),
        .read_data_2(reg_read_data_2),
        .clk(clk)
    );
    // ALU
    ALU alu(
        .ALUCtrl(ALU_control_out),
        .a(ALU_a),
        .b(ALU_b),
        .ALU_result(ALU_out),
        .zero(output_zero)
    );
    // Control
```

```
    Control con(
        .op_code(Instructions[31:26]),
        .RegDst(RegDst),
        .Branch(Branch),
        .Jump(Jump),
        .MemRead(MemRead),
        .MemtoReg(MemtoReg),
        .MemWrite(MemWrite),
        .ALUSrc(ALUSrc),
        .RegWrite(RegWrite),
        .ALUOp(ALUOp)
    );
    // ALU control
    ALU_control alu_ctrl(
        .funct(Instructions[5:0]),
        .ALUOp(ALUOp),
        .ALUCtrl(ALU_control_out)
    );
    // Sign Extension
    sign_extension sign_ext(
        .shortInput(Instructions[15:0]),
        .longOutput(extended_imm)
    );
    // Data memory
    data_memory dm(
        .clk(clk),
        .MemRead(MemRead),
        .MemWrite(MemWrite),
        .address(ALU_out),
        .write_data(reg_read_data_2),
        .read_data(D_MEM_read_data)
    );

    // Mux
    Mux_N_bit#(5) mux_select_write_reg(
        .in1(Instructions[20:16]),
        .in2(Instructions[15:11]),
        .select(RegDst),
        .out(write_address)
    );
    Mux_N_bit#(32) mux_select_ALU_in(
        .in1(reg_read_data_2),
        .in2(extended_imm),
        .select(ALUSrc),
        .out(ALU_b)
    );
    assign write_data = (MemtoReg == 1'b0) ? ALU_out:D_MEM_read_data;
    Mux_N_bit#(32) mux_select_PC_one(
        .in1(PC_plus_four),
        .in2(branch_out),
        .select(branch_mux),
        .out(branch_result)
    );
    assign PC_in = (Jump == 1'b0) ? branch_result:jump_addr;
endmodule
`endif
```

adder.v

```
`ifndef MODULE_ADDER
`define MODULE_ADDER
`timescale 1ns / 1ps
module adder(
    input [31:0] a,
    input [31:0] b,
    output [31:0] sum
    );
    reg [31:0] sum;
    always @(a or b)
        begin
            sum = a + b;
        end
endmodule
`endif
```

ALU_control.v

```
`ifndef MODULE_ALU_CONTROL
`define MODULE_ALU_CONTROL
`timescale 1ns / 1ps
module ALU_control(
    funct,ALUOp,ALUCtrl
    );
    input [5:0] funct;
    input [1:0] ALUOp;
    output [3:0] ALUCtrl;
    reg [3:0] ALUCtrl;
    always @ (funct or ALUOp)
    begin
        case(ALUOp)
            2'b00: assign ALUCtrl = 4'b0010;
            2'b01: assign ALUCtrl = 4'b0110;
            2'b10:
                begin
                    if (funct == 6'b100000)      assign ALUCtrl = 4'b0010;
                    else if (funct == 6'b100010) assign ALUCtrl = 4'b0110;
                    else if (funct == 6'b100100) assign ALUCtrl = 4'b0000;
                    else if (funct == 6'b100101) assign ALUCtrl = 4'b0001;
                    else if (funct == 6'b101010) assign ALUCtrl = 4'b0111;
                end
            2'b11:
                assign ALUCtrl = 4'b0000;
        endcase
    end
endmodule
`endif
```

ALU.v

```
`ifndef MODULE_ALU
`define MODULE_ALU
`timescale 1ns / 1ps
module ALU(
    ALUCtrl,a,b,zero,ALU_result
    );
    input [3:0] ALUCtrl;
    input [31:0] a, b;
    output zero;
    output [31:0] ALU_result;
```

```
    reg zero;
    reg [31:0] ALU_result;
    always @ (a or b or ALUCtrl)
    begin
        case (ALUCtrl)
            4'b0000:
            begin
                assign ALU_result = a & b;
                assign zero = (a & b == 0) ? 1:0;
            end
            4'b0001:
            begin
                assign ALU_result = a | b;
                assign zero = (a | b == 0) ? 1:0;
            end
            4'b0010:
            begin
                assign ALU_result = a + b;
                assign zero = (a + b == 0) ? 1:0;
            end
            4'b0110:
            begin
                assign ALU_result = a - b;
                assign zero = ( a == b) ? 1:0;
            end
            4'b0111:
            begin
                assign ALU_result = (a < b) ? 1:0;
                assign zero = (a < b) ? 0:1;
            end
            default:
            begin
                assign ALU_result = a;
                assign zero = (a == 0) ? 1:0;
            end
        endcase
    end
endmodule
`endif
```

control.v

```
`ifndef MODULE_CONTROL
`define MODULE_CONTROL
`timescale 1ns / 1ps
module Control (
    input       [5:0]   op_code,
    output reg  [1:0]   ALUOp,
    output reg  RegDst,
                Jump,
                Branch,
                MemRead,
                MemtoReg,
                MemWrite,
                ALUSrc,
                RegWrite
);

    initial begin
```

```
18          RegDst      = 1'b0;
19          Jump        = 1'b0;
20          Branch      = 1'b0;
21          MemRead     = 1'b0;
22          MemtoReg    = 1'b0;
23          MemWrite    = 1'b0;
24          ALUSrc      = 1'b0;
25          RegWrite    = 1'b0;
26          ALUOp       = 2'b00;
27      end
28
29      always @ ( op_code ) begin
30          case (op_code)
31              6'b000000: begin // R-type
32                  RegDst      <= 1'b1;
33                  Jump        <= 1'b0;
34                  Branch      <= 1'b0;
35                  MemRead     <= 1'b0;
36                  MemtoReg    <= 1'b0;
37                  MemWrite    <= 1'b0;
38                  ALUSrc      <= 1'b0;
39                  RegWrite    <= 1'b1;
40                  ALUOp       <= 2'b10;
41              end
42              6'b000010: begin // j
43                  RegDst      <= 1'b1;
44                  Jump        <= 1'b1;
45                  Branch      <= 1'b0;
46                  MemRead     <= 1'b0;
47                  MemtoReg    <= 1'b0;
48                  MemWrite    <= 1'b0;
49                  ALUSrc      <= 1'b0;
50                  RegWrite    <= 1'b0;
51                  ALUOp       <= 2'b10;
52        end
53              6'b000100: begin // beq
54                  RegDst      <= 1'b1;
55                  Jump        <= 1'b0;
56                  Branch      <= 1'b1;
57                  MemRead     <= 1'b0;
58                  MemtoReg    <= 1'b0;
59                  MemWrite    <= 1'b0;
60                  ALUSrc      <= 1'b0;
61                  RegWrite    <= 1'b0;
62                  ALUOp       <= 2'b01;
63              end
64              6'b000100: begin // bne
65                  RegDst      <= 1'b1;
66                  Jump        <= 1'b0;
67                  Branch      <= 1'b0;
68                  MemRead     <= 1'b0;
69                  MemtoReg    <= 1'b0;
70                  MemWrite    <= 1'b0;
71                  ALUSrc      <= 1'b0;
72                  RegWrite    <= 1'b0;
73                  ALUOp       <= 2'b01;
74              end
75              6'b001000: begin // addi
```

```
                RegDst      <= 1'b0;
                Jump        <= 1'b0;
                Branch      <= 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;
                Branch      <= 1'b0;
                MemRead     <= 1'b0;
                MemtoReg    <= 1'b0;
                MemWrite    <= 1'b0;
                ALUSrc      <= 1'b1;
                RegWrite    <= 1'b1;
                ALUOp       <= 2'b11;
            end
            6'b100011: begin // lw
                RegDst      <= 1'b0;
                Jump        <= 1'b0;
                Branch      <= 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;
                Branch      <= 1'b0;
                MemRead     <= 1'b0;
                MemtoReg    <= 1'b0;
                MemWrite    <= 1'b1;
                ALUSrc      <= 1'b1;
                RegWrite    <= 1'b0;
                ALUOp       <= 2'b00;
        end

            default: ;
          endcase
    end

endmodule
`endif
```

data_memory.v

```
`ifndef MODULE_DATA_MEMORY
`define MODULE_DATA_MEMORY
`timescale 1ns / 1ps
module data_memory (
    input               clk,
    input               MemRead,
```

```
                    MemWrite,
    input     [31:0]  address,
                      write_data,
    output    [31:0]  read_data
);
    parameter         size = 64; // size of data register_memory
    integer           i;
    wire      [31:0]  index;
    reg       [31:0]  register_memory [0:size-1];

    assign index = address >> 2; // address/4
    initial begin
        for (i = 0; i < size; i = i + 1)
            register_memory[i] = 32'b0;
        // read_data = 32'b0;
    end
    always @ ( posedge clk ) begin
        if (MemWrite == 1'b1) begin
            register_memory[index] = write_data;
        end
    end
endmodule
`endif
```

instruction_memory.v

```
`ifndef MODULE_INSTRUCTION_MEMORY
`define MODULE_INSTRUCTION_MEMORY
module instruction_memory (
    input       [31:0]  address,
    output      [31:0]  instruction
);
    parameter size = 128;
    // initialize memory
    reg [31:0] memory [0:size-1];
    // clear all memory to nop
    initial begin
        for (i = 0; i < size; i = i + 1)
            memory[i] = 32'b0;
        // include the instruction_memory
        `include "InstructionMem_for_P2_Demo_updated.txt"
    end
    // Output the menmory at address
    assign instruction = memory[address >> 2];
endmodule
`endif
```

program_program_counter.v

```
`ifndef MODULE_PC
`define MODULE_PC
`timescale 1ns / 1ps
module program_counter(
    PC_in,PC_out,reset,clk
    );
    input [31:0] PC_in;
    output [31:0] PC_out;
    input reset, clk;
    reg [31:0] PC_out;
    initial begin
```

```
        PC_out = -4;
    end
    always @(posedge clk)
    begin
        if(reset)
            begin
                PC_out <= 32'b0;
            end
        else PC_out <= PC_in[31:0];
    end
endmodule
`endif
```

register.v

```
`ifndef MODULE_REGISTERS
`define MODULE_REGISTERS
`timescale 1ns / 1ps

module Registers (
    input       clk, regWrite,
    input        [4:0]   read_register_1,  read_register_2,
    input        [4:0]   write_register,
    input        [31:0]  write_data,
    output       [31:0]  read_data_1,read_data_2
);
  parameter size = 32;           // 32-bit CPU, $0 - $31
    reg [31:0] register_memory [0:size-1];
    integer i;

    initial begin
        for (i = 0; i < size; i = i + 1)
            register_memory[i] = 32'b0;
    end
    assign read_data_1 = register_memory[read_register_1];
    assign read_data_2 = register_memory[read_register_2];
    always @(negedge clk) begin
        if (regWrite == 1)
            register_memory[write_register] <= write_data;
    end
endmodule // registers
`endif
```

sign_extension.v

```
`ifndef MODULE_SIGN_EXT
`define MODULE_SIGN_EXT
`timescale 1ns / 1ps
module sign_extension(
    shortInput, longOutput
    );
    input [15:0] shortInput;
    output [31:0] longOutput;
    assign longOutput[15:0] = shortInput[15:0];
    assign longOutput[31:16] = shortInput[15]?16'b1111_1111_1111_1111:16'b0
    ;
endmodule
`endif
```

Mux_N_bit.v

```
`ifndef MODULE_Mux
`define MODULE_Mux
module Mux_N_bit(in1,in2,out,select);
    parameter N = 32;
    input[N-1:0] in1,in2;
    input select;
    output[N-1:0] out;
    assign out = select?in2:in1;
endmodule
`endif
```

Simulation Result:

```
Texual result of single cycle:
=============================================
Time:              0, CLK = 0, PC = 0xfffffffc
[$s0] = 0x00000000, [$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:              0, CLK = 1, 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:              1, CLK = 0, PC = 0x00000000
[$s0] = 0x00000000, [$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:              1, 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:              2, CLK = 0, PC = 0x00000004
[$s0] = 0x00000000, [$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:              2, 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:              3, CLK = 0, PC = 0x00000008
[$s0] = 0x00000020, [$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:              3, 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:              4, CLK = 0, PC = 0x0000000c
[$s0] = 0x00000037, [$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:              4, 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:              5, 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:              5, 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:              6, 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:              6, CLK = 1, PC = 0x00000018
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
```

```
105 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
106 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
107 ================================================
108 Time:              7, CLK = 0, PC = 0x00000018
109 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0x00000000
110 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
111 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
112 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
113 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
114 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
115 Time:              7, CLK = 1, PC = 0x0000001c
116 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0x00000000
117 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
118 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
119 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
120 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
121 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
122 ================================================
123 Time:              8, CLK = 0, PC = 0x0000001c
124 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
125 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
126 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
127 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
128 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
129 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
130 Time:              8, CLK = 1, PC = 0x00000020
131 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
132 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
133 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
134 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
135 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
136 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
137 ================================================
138 Time:              9, CLK = 0, PC = 0x00000020
139 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
140 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
141 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
142 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
143 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
144 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
145 Time:              9, CLK = 1, PC = 0x00000024
146 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
147 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
148 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
149 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
150 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
151 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
152 ================================================
153 Time:             10, CLK = 0, PC = 0x00000024
154 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
155 [$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
156 [$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
157 [$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
158 [$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
159 [$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
160 Time:             10, CLK = 1, PC = 0x00000028
161 [$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
162 [$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:           11, CLK = 0, PC = 0x00000028
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$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:           11, CLK = 1, PC = 0x0000002c
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$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:           12, CLK = 0, PC = 0x0000002c
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$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:           12, CLK = 1, PC = 0x00000030
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$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:           13, CLK = 0, PC = 0x00000030
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0xfffffff9
[$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:           13, CLK = 1, PC = 0x00000034
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0xfffffff9
[$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:           14, CLK = 0, PC = 0x00000034
[$s0] = 0x00000037, [$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:           14, 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] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:              15, 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] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:              15, CLK = 1, PC = 0x0000003c
[$s0] = 0x00000037, [$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:              16, CLK = 0, PC = 0x0000003c
[$s0] = 0x00000037, [$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:              16, CLK = 1, PC = 0x00000040
[$s0] = 0x00000037, [$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:              17, CLK = 0, PC = 0x00000040
[$s0] = 0x00000037, [$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:              17, CLK = 1, PC = 0x00000044
[$s0] = 0x00000037, [$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:              18, CLK = 0, PC = 0x00000044
[$s0] = 0x00000037, [$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, [$t9] = 0x00000000
Time:              18, CLK = 1, PC = 0x00000048
[$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, [$t9] = 0x00000000
================================================
Time:              19, CLK = 0, PC = 0x00000048
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:              19, CLK = 1, PC = 0x0000004c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
================================================
Time:              20, CLK = 0, PC = 0x0000004c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:              20, CLK = 1, PC = 0x00000050
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
================================================
Time:              21, CLK = 0, PC = 0x00000050
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:              21, CLK = 1, PC = 0x00000054
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
================================================
Time:              22, CLK = 0, PC = 0x00000054
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:              22, CLK = 1, PC = 0x00000058
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
================================================
Time:              23, CLK = 0, PC = 0x00000058
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:              23, CLK = 1, PC = 0x0000005c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
================================================
Time:              24, CLK = 0, PC = 0x0000005c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:              24, CLK = 1, PC = 0x00000060
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:              25, CLK = 0, PC = 0x00000060
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:              25, CLK = 1, PC = 0x00000064
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:              26, CLK = 0, PC = 0x00000064
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
```

```
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:              26, CLK = 1, PC = 0x00000068
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:              27, CLK = 0, PC = 0x00000068
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:              27, CLK = 1, PC = 0x0000006c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:              28, CLK = 0, PC = 0x0000006c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:              28, CLK = 1, PC = 0x00000070
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:              29, CLK = 0, PC = 0x00000070
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
Time:              29, CLK = 1, PC = 0x00000074
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$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:              30, CLK = 0, PC = 0x00000074
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$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:              30, CLK = 1, PC = 0x0000005c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$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:              31, CLK = 0, PC = 0x0000005c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:              31, CLK = 1, PC = 0x00000060
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:              32, CLK = 0, PC = 0x00000060
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:              32, CLK = 1, PC = 0x00000064
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:              33, CLK = 0, PC = 0x00000064
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$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:              33, CLK = 1, PC = 0x00000068
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
```

```
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:              34, CLK = 0, PC = 0x00000068
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:              34, CLK = 1, PC = 0x0000006c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
================================================
Time:              35, CLK = 0, PC = 0x0000006c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:              35, CLK = 1, PC = 0x000000ac
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
================================================
Time:              36, CLK = 0, PC = 0x000000ac
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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, [$t9] = 0x00000000
Time:              36, CLK = 1, PC = 0x000000b0
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$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
================================================
```

5.2 Pipielined

Many files of pipeline processor are the same as single-cycle processor. The different files are shown below. clock_clock_divider.v

```
`timescale 1ns / 1ps

module clock_divider(divided_clock, clock, reset);

```

```
    input clock, reset;
    output divided_clock;

    reg divided_clock1;

    initial begin
    divided_clock1 <= 0;
    end

    parameter r = 100;
    integer now = 1;

    //notice that reset is not used here
    always @ (posedge clock)
        begin
        if (now <= r/2) divided_clock1 = 1;
        else divided_clock1 =0;
        if (now == r) now =1;
        else now = now + 1;
        end

    assign divided_clock = divided_clock1;

endmodule
```

control.v

```
`ifndef MODULE_CONTROL
`define MODULE_CONTROL
`timescale 1ns / 1ps

module Control (
    input       [5:0]   op_code,
    output reg  [1:0]   ALUOp,
    output reg  RegDst,
                Jump,
                Branch,
                Bne,
                MemRead,
                MemtoReg,
                MemWrite,
                ALUSrc,
                RegWrite
);

    initial begin
        RegDst      = 1'b0;
        Jump        = 1'b0;
        Branch      = 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 @ ( *) begin
```

```
33          case (op_code)
34              6'b000000: begin // R-type
35                  RegDst      <= 1'b1;
36                  Jump        <= 1'b0;
37                  Branch      <= 1'b0;
38                  Bne          <=1'b0;
39                  MemRead     <= 1'b0;
40                  MemtoReg    <= 1'b0;
41                  MemWrite    <= 1'b0;
42                  ALUSrc      <= 1'b0;
43                  RegWrite    <= 1'b1;
44                  ALUOp       <= 2'b10;
45              end
46              6'b000010: begin // j
47                  RegDst      <= 1'b1;
48                  Jump        <= 1'b1;
49                  Branch      <= 1'b0;
50                   Bne          <=1'b0;
51                  MemRead     <= 1'b0;
52                  MemtoReg    <= 1'b0;
53                  MemWrite    <= 1'b0;
54                  ALUSrc      <= 1'b0;
55                  RegWrite    <= 1'b0;
56                  ALUOp       <= 2'b10;
57      end
58              6'b000100: begin // beq
59                  RegDst      <= 1'b1;
60                  Jump        <= 1'b0;
61                  Branch      <= 1'b1;
62                   Bne          <=1'b0;
63                  MemRead     <= 1'b0;
64                  MemtoReg    <= 1'b0;
65                  MemWrite    <= 1'b0;
66                  ALUSrc      <= 1'b0;
67                  RegWrite    <= 1'b0;
68                  ALUOp       <= 2'b01;
69              end
70              6'b000100: begin // bne
71                  RegDst      <= 1'b1;
72                  Jump        <= 1'b0;
73                  Branch      <= 1'b0;
74                   Bne          <=1'b1;
75                  MemRead     <= 1'b0;
76                  MemtoReg    <= 1'b0;
77                  MemWrite    <= 1'b0;
78                  ALUSrc      <= 1'b0;
79                  RegWrite    <= 1'b0;
80                  ALUOp       <= 2'b01;
81              end
82              6'b001000: begin // addi
83                  RegDst      <= 1'b0;
84                  Jump        <= 1'b0;
85                  Branch      <= 1'b0;
86                   Bne          <=1'b0;
87                  MemRead     <= 1'b0;
88                  MemtoReg    <= 1'b0;
89                  MemWrite    <= 1'b0;
90                  ALUSrc      <= 1'b1;
```

```
                RegWrite    <= 1'b1;
                ALUOp       <= 2'b00;
            end
            6'b001100: begin // andi
                RegDst      <= 1'b0;
                Jump        <= 1'b0;
                Branch      <= 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'b100011: begin // lw
                RegDst      <= 1'b0;
                Jump        <= 1'b0;
                Branch      <= 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;
                Branch      <= 1'b0;
                 Bne         <=1'b0;
                MemRead     <= 1'b0;
                MemtoReg    <= 1'b0;
                MemWrite    <= 1'b1;
                ALUSrc      <= 1'b1;
                RegWrite    <= 1'b0;
                ALUOp       <= 2'b00;
      end

            default: ;
        endcase
    end

endmodule
`endif
```

forward_mux.v

```
`timescale 1ns / 1ps
`ifndef MODULE_FORWARD_MUX
`define MODULE_FORWARD_MUX
module forward_mux(
EX_read,
WB_result,
MEM_result,
out,
select
    );
```

```
        parameter N = 32;
        input[N-1:0] EX_read,MEM_result,WB_result;
        input [1:0] select;
        output[N-1:0] out;
        reg [N-1:0] out;
        always@(*)
        begin
        if(select==2'b00) out<=EX_read;
        else if(select==2'b01) out<=WB_result;
        else if(select==2'b10) out<=MEM_result;
        end
    endmodule
`endif
```

forwarding.v

```
`timescale 1ns / 1ps
module forwarding(
    input beq,
    input bne,
    input [4:0] ID_rs,
    input [4:0] ID_rt,
    input EM_RW,
    input MW_RW,
    input MEM_MemWrite,
    input [4:0] EM_Rd,
    input [4:0] IE_Rs,
    input [4:0] IE_Rt,
    input [4:0] MW_Rd,
    output [1:0] beqA,
    output [1:0] beqB,
    output  [1:0] ForwardA,
    output  [1:0] ForwardB,
    output sw_src

    );

reg [1:0] tmpA,tmp_beqA;
reg [1:0] tmpB, tmp_beqB;
reg temp_sw_src;
initial begin
    tmp_beqA<=0;
    tmp_beqB<=0;
  tmpA<=0;
  tmpB<=0;
  temp_sw_src<=0;
end

always@(*)
begin
if(EM_RW && (EM_Rd!=0) && (EM_Rd==ID_rs)&& (bne||beq))
  begin
    tmp_beqA<=2'b10;
  end
else if((bne||beq)&&(MW_RW && (MW_Rd!=0) && (MW_Rd==ID_rs))&& ~(EM_RW && (
    EM_Rd!=0) && (EM_Rd==IE_Rs)))
  begin
    tmp_beqA<=2'b01;
  end
```

```
else
  begin
    tmp_beqA<=2'b00;
  end
if(EM_RW && (EM_Rd!=0) && (EM_Rd==ID_rt)&& (bne||beq))
        begin
              tmp_beqB<=2'b10;
        end
    else if((bne||beq)&&(MW_RW && (MW_Rd!=0) && (MW_Rd==ID_rt))&& ~(EM_RW
   && (EM_Rd!=0) && (EM_Rd==IE_Rs)))
        begin
              tmp_beqB<=2'b01;
        end
    else
        begin
              tmp_beqB<=2'b00;
        end



if(MEM_MemWrite && MW_RW &&(EM_Rd==MW_Rd))
  begin
  temp_sw_src<=1'b1;
  end
else
begin
 temp_sw_src<=1'b0;
end
if(EM_RW && (EM_Rd!=0) && (EM_Rd==IE_Rs))
  begin
    tmpA<=2'b10;
  end
else if((MW_RW && (MW_Rd!=0) && (MW_Rd==IE_Rs))&& ~(EM_RW && (EM_Rd!=0) &&
   (EM_Rd==IE_Rs)))
  begin
    tmpA<=2'b01;
  end
else
  begin
    tmpA<=2'b00;
  end

if(EM_RW && (EM_Rd!=0) && (EM_Rd==IE_Rt))
  begin
    tmpB<=2'b10;
  end
else if((MW_RW && (MW_Rd!=0) && (MW_Rd==IE_Rt))&& ~(EM_RW && (EM_Rd!=0) &&
   (EM_Rd==IE_Rt)))
  begin
    tmpB<=2'b01;
  end
else
  begin
    tmpB<=2'b00;
  end
end

assign beqA=tmp_beqA;
```

```
assign beqB=tmp_beqB;
assign ForwardA=tmpA;
assign ForwardB=tmpB;
assign sw_src=temp_sw_src;
endmodule
/*module forwarding(
EX_rs,EX_rt,
MEM_rd,MEM_RegWrite,
WB_rd, WB_RegWrite,
selectA, selectB
);
input EX_rs,EX_rt,MEM_rd,MEM_RegWrite,WB_rd, WB_RegWrite;
output [1:0] selectA, selectB;
reg [1:0] selectA, selectB;
initial begin
selectA<=2'b00;
selectB<=2'b00;
end
always@(*)
begin
selectA<=2'b00;
selectB<=2'b00;
if((MEM_RegWrite==1'b1)&&(MEM_rd!=0)) //EX hazard
begin
if(EX_rs==MEM_rd) selectA<=2'b10;
if(EX_rt==MEM_rd) selectB<=2'b10;
end
if((WB_RegWrite==1'b1)&&(WB_rd!=0)) // MEM hazard
begin
if(selectA==2'b00)
begin
if(EX_rs==MEM_rd) selectA<=2'b01;
end
if(selectA==2'b00)
begin
if(EX_rt==MEM_rd) selectB<=2'b01;
end
end
end
endmodule*/
```

HazardDtection.v

```
`timescale 1ns / 1ps


module HazardDtection(
   //input IF_instruction,
    input clk,
    input bne,
    input branch,
    input jump,
    input ID_MemRead,
    input ID_RegWrite,
    input EX_MemRead,
    input [4:0] EX_Rt,
    input [4:0] ID_Rt,
    input [4:0] ID_Rd,
    input [4:0] IF_Rs,
```

```
    input [4:0] IF_Rt,
    output IF_Write,
    output PC_Write
    //output Controlsrc
    );

reg PCtem;
       // reg Controltem;
        reg IFIDtem;

        initial begin
            PCtem<=1;
            //Controltem<=0;
            IFIDtem<=1;
        end


        always@(posedge clk)
        begin
        //load use hazard:lw $2,0($1);and $4,$2,$5
        if (ID_MemRead && ((ID_Rt==IF_Rs) | (ID_Rt==IF_Rt))&&(branch==0)&&(jump==0)&&(bne==0))
        begin
            PCtem<=0;
          //  Controltem<=1;
            IFIDtem<=0;
        end
        //lw and not ajacent beq
        else if ((bne||branch) && EX_MemRead &&((EX_Rt==IF_Rs) | (EX_Rt==IF_Rt))&&(jump==0))
        begin
            PCtem<=0;
           // Controltem<=1;
            IFIDtem<=0;
        end
        //I/lw and adjacent beq
        else if((bne||branch) && ID_RegWrite&&  ((ID_Rt==IF_Rs&& IF_Rs!=0)|(ID_Rt==IF_Rt&& IF_Rt!=0))&&(jump==0))
        begin
            PCtem<=0;
           //Controltem=1;
            IFIDtem<=0;
        end
        // r-type and adjacent beq
        else if((bne||branch) && ID_RegWrite && ((ID_Rd==IF_Rs&&IF_Rs!=0)|(ID_Rd==IF_Rt&& IF_Rt!=0))&&(jump==0))
            begin
                PCtem<=0;
                //Controltem=1;
                IFIDtem<=0;
            end
        /*else if (jump)
        begin
            PCtem=1;
            Controltem=0;
            IFIDtem=1;
        end*/

```

```
        else
        begin
             PCtem<=1;
             IFIDtem<=1;
            //Controltem<=0;
        end



        end


        assign IF_Write=IFIDtem;
        assign PC_Write=PCtem;
        //assign Controlsrc=Controltem;
endmodule
```

ID_EXpipeline.v

```
`timescale 1ns / 1ps


module ID_EXpipeline(
    input RegWrite,
    input MemtoReg,
    input MemRead,
    input MemWrite,
    input [1:0] ALUop,
    input ALUsrc,
    input RegDst,
    input [31:0] Readdata1,
    input [31:0] Readdata2,
    input [31:0] ExtendResult,
    input [4:0] Rd,
    input [4:0] Rt,
    input [4:0] Rs,
    input [5:0] func,

    output reg RegWrite1,
    output reg MemtoReg1,
    output reg MemRead1,
    output reg MemWrite1,
    output reg [1:0] ALUop1,
    output reg ALUsrc1,
    output reg RegDst1,
    output reg [31:0] RData1,
    output reg [31:0] RData2,
    output reg [31:0] ER,
    output reg [4:0]Rd1,
    output reg [4:0]Rt1,
    output reg [4:0]Rs1,
    output reg [5:0] func1,
    input clk
    );

    always@(negedge clk)
   begin
    Rs1<=Rs;
    RegWrite1<=RegWrite;
```

```
    MemtoReg1<=MemtoReg;
    MemRead1<=MemRead;
    MemWrite1<=MemWrite;
    ALUop1<=ALUop;
    ALUsrc1<=ALUsrc;
    RData1<=Readdata1;
    RData2<=Readdata2;
    ER<=ExtendResult;
    Rd1<=Rd;
    Rt1<=Rt;
    RegDst1<=RegDst;
    func1<=func;
   end

endmodule
```

PC_MUX.v

```
`timescale 1ns / 1ps
module PC_MUX(
IF_PC_plus4,
bne_mux,
beq_mux,
ID_Jump,
ID_Jaddress,
ID_target,
PC_in
    );
input [31:0] IF_PC_plus4, ID_Jaddress, ID_target;
input beq_mux,bne_mux, ID_Jump;
output [31:0] PC_in;
reg [31:0] PC_in;
always@(*)
begin
if (beq_mux==1||bne_mux ==1) PC_in<=ID_target;
else if (ID_Jump==1) PC_in<=ID_Jaddress;
else PC_in<=IF_PC_plus4;
end
endmodule
```

pipe_write.v

```
`timescale 1ns / 1ps
`ifndef MODULE_PIPE_WRITE
`define MODULE_PIPE_WRITE
module pipe_write(
Reg_in,
Reg_out,
flush,
clk,
hold
);
//after finish the stage, push the items into next pipelined register(every
    stage execute at the same time)
//IFtoID: instruction, PC+4
//IDtoEX:Jump, Branch, MemRead, MemtoReg,ALUop,MemWrite,ALUSrc,RegWrite,
   ID_rs, ID_rt, ID_read1,ID_read2,ID_rd,PC+4
//EXtoMEM: MemRead, MemtoReg,,MemWrite,RegWrite,EX_out, EX_rd
//MEMtoWB: MemtoReg,,RegWrite,WB_rd
   parameter size =  64; //size of the pipelined registers
```

```
    input [size-1:0] Reg_in;
    input flush,clk,hold;
    output [size-1:0] Reg_out;
    reg [size-1:0] Reg_out;
    always@(negedge clk) // second half clock for write
    begin
    if(flush==1 || hold ==1) Reg_out<=0;
    else    Reg_out<=Reg_in;
    end
endmodule
`endif
```

pipeline.v

```
`timescale 1ns / 1ps
module pipeline(
clk,reset,C,AN,clock,reg_switch,PC_switch
    );
    input clk,reset,clock,PC_switch;
    input [4:0] reg_switch;
    output reg [6:0] C;
    output [3:0] AN;
    initial begin
            C=7'b1111111;
    end

// ALU
wire [31:0] ALU_a,ALU_b,ALU_out,EX_result,MEM_result,WB_result,temp_ALU_b;
wire [3:0] ALU_control;
wire [5:0] EX_func;
wire [1:0] ID_ALUOp, EX_ALUOp,selectA,selectB, beqA, beqB;
//Control Signals
wire ID_RegDst,ID_Jump,ID_Branch,ID_ALUSrc,ID_MemRead,ID_MemWrite,
    ID_MemtoReg,ID_RegWrite,ID_Bne,
EX_RegDst,EX_Jump,EX_Branch,EX_ALUSrc,EX_MemRead,EX_MemWrite,EX_MemtoReg,
    EX_RegWrite,EX_Bne,
MEM_Jump,MEM_Branch,MEM_MemRead,MEM_MemWrite,MEM_MemtoReg,MEM_RegWrite,
    MEM_Bne,
WB_MemtoReg,WB_RegWrite,EX_Zero,MEM_Zero;
//address
wire [31:0] ID_extend, EX_extend,ID_Jaddress,ID_target;
//PC Signals
wire [31:0] PC_in, PC_out, IF_PC_plus4, ID_PC_plus4;
// Instructions memory signals
wire [31:0] IF_instruction, ID_instruction, sw_data, beq1, beq2;
//Register data
wire [31:0] ID_read1, ID_read2, EX_read1,EX_read2,write_data,MEM_read2,
    MEM_data,WB_data;
//Registers
wire [4:0] EX_rs,EX_rt,EX_rd,temp_rd, MEM_rd, WB_rd;
//bne,neq
wire beq_mux,bne_mux,ID_Zero,sw_src;
assign ID_Zero= (beq1==beq2)? 1:0;
assign beq_mux = ID_Zero & ID_Branch;
assign bne_mux = (~ID_Zero) & ID_Bne;
//hazard
wire PC_write, IF_Flush, IF_Write,beq,bne,jump;
assign beq = (IF_instruction[31:26]==6'b000100)?1:0;
assign bne = (IF_instruction[31:26]==6'b000101)?1:0;
```

```
assign jump = (IF_instruction[31:26]==6'b000010)?1:0;
assign IF_Flush = beq_mux | bne_mux |ID_Jump;
// Connect PC
 program_counter PC(PC_in,PC_out,reset,clk,~PC_write);
//IF
//instruction memory
instruction_memory get_ins(
      .address(PC_out),
      .instruction(IF_instruction)
 );
// Cal PC + 4
adder add_PC_plus4(
    .a(PC_out),
    .b(32'b0100),
    .sum(IF_PC_plus4)
);

//Jump target address
assign ID_Jaddress={ID_PC_plus4[31:28],ID_instruction[25:0],2'b00};
//Hazard Detection Unit
HazardDtection HD(
  //IF_instruction,
  clk,
  bne,
  beq,
  jump,
  ID_MemRead,
  ID_RegWrite,
  EX_MemRead,
  EX_rt,
  ID_instruction[20:16],
  ID_instruction[15:11],
  IF_instruction[26:21],
  IF_instruction[20:16],
  IF_Write,
  PC_write
    );
// Register File
wire [31:0] Q2;
assign ID_target=ID_PC_plus4 + ID_extend * 4;
Registers get_reg(
    .read_register_1(ID_instruction[25:21]),
    .read_register_2(ID_instruction[20:16]),
    .switch(reg_switch),
    .regWrite(WB_RegWrite),
    .write_register(WB_rd),
    .write_data(write_data),
    .read_data_1(ID_read1),
    .read_data_2(ID_read2),
    .clk(clk),
    .Q2(Q2)
);
Control controller(ID_instruction[31:26], ID_ALUOp,ID_RegDst,ID_Jump,
    ID_Branch,ID_Bne,ID_MemRead, ID_MemtoReg,ID_MemWrite,ID_ALUSrc,
    ID_RegWrite);
sign_extension sign_ext(
    .shortInput(ID_instruction[15:0]),
    .longOutput(ID_extend)
```

```
);
//EX
// ALU
 ALU alu(
      .ALUCtrl(ALU_control),
      .a(ALU_a),
      .b(ALU_b),
      .ALU_result(EX_result),
      .zero(EX_Zero)
 );
// ALU control
 ALU_control alu_ctrl(
      .funct(EX_func),
      .ALUOp(EX_ALUOp),
      .ALUCtrl(ALU_control)
 );
//forwarding

forwarding forwarding_unit(
ID_Branch,
ID_Bne,
ID_instruction[25:21],
ID_instruction[20:16],
MEM_RegWrite,
WB_RegWrite,
MEM_MemWrite,
MEM_rd,
EX_rs,
EX_rt,
WB_rd,
beqA,
beqB,
selectA,
selectB,
sw_src
);
//MEM
data_memory dm(
     .clk(clk),
     .MemRead(MEM_MemRead),
     .MemWrite(MEM_MemWrite),
     .address(MEM_result),
     .write_data(sw_data),
     .read_data(MEM_data)
);
// Mux
Mux_N_bit#(5) mux_select_write_reg(
     .in1(EX_rt),
     .in2(EX_rd),
     .select(EX_RegDst),
     .out(temp_rd)
);
Mux_N_bit#(32) mux_select_ALU_in(
     .in1(temp_ALU_b),
     .in2(EX_extend),
     .select(EX_ALUSrc),
     .out(ALU_b)
);
```

```
156 forward_mux BEQ_MUX1(ID_read1,write_data,MEM_result,beq1,beqA);
157 forward_mux BEQ_MUX2(ID_read2,write_data,MEM_result,beq2,beqB);
158 forward_mux ALU_MUX1(EX_read1,write_data,MEM_result,ALU_a,selectA);
159 forward_mux ALU_MUX2(EX_read2,write_data,MEM_result,temp_ALU_b,selectB);
160 Mux_N_bit#(32) mux_write(
161     .in1(WB_result),
162     .in2(WB_data),
163     .select(WB_MemtoReg),
164     .out(write_data)
165 );
166 Mux_N_bit#(32) mux_sw(
167     .in1(MEM_read2),
168     .in2(write_data),
169     .select(sw_src),
170     .out(sw_data)
171 );
172 PC_MUX next_PC(IF_PC_plus4,bne_mux,beq_mux,ID_Jump,ID_Jaddress,ID_target,
    PC_in);
173 //overwrite register
174 pipe_write#(64) IF_ID ({IF_instruction,IF_PC_plus4},{ID_instruction,
    ID_PC_plus4},IF_Flush,clk,~IF_Write);
175 ID_EXpipeline ID_EX(
176     ID_RegWrite,
177     ID_MemtoReg,
178     ID_MemRead,
179     ID_MemWrite,
180     ID_ALUOp,
181     ID_ALUSrc,
182     ID_RegDst,
183     ID_read1,
184     ID_read2,
185     ID_extend,
186     ID_instruction[15:11],
187     ID_instruction[20:16],
188     ID_instruction[25:21],
189     ID_instruction[5:0],
190     EX_RegWrite,
191     EX_MemtoReg,
192     EX_MemRead,
193     EX_MemWrite,
194     EX_ALUOp,
195     EX_ALUSrc,
196     EX_RegDst,
197     EX_read1,
198     EX_read2,
199     EX_extend,
200     EX_rd,
201     EX_rt,
202     EX_rs,
203     EX_func,
204     clk
205     );
206
207 pipe_write#(85) EX_MEM({temp_rd,EX_result,EX_read2,EX_Zero,EX_MemRead,
    EX_MemWrite,EX_MemtoReg,EX_RegWrite},
208 {MEM_rd,MEM_result,MEM_read2,MEM_Zero,MEM_MemRead,MEM_MemWrite,MEM_MemtoReg
    ,MEM_RegWrite},
209 0,clk,0);
```

```
pipe_write#(71) MEM_WB({MEM_result,MEM_data,MEM_rd,MEM_MemtoReg,
    MEM_RegWrite},
{WB_result,WB_data,WB_rd,WB_MemtoReg,WB_RegWrite},
0,clk,0);
//SSD
   wire clock_d;
   reg [3:0] Q;

   clock_divider clock1(clock_d, clock, reset);

   ring_counter ring(AN, clock_d, reset);

 wire [31:0] ssd_out;
   // here only needs to use 4 hexo numbers because first 4 numbers must be
    0 for the small instruction value
assign ssd_out = (PC_switch)? PC_out:Q2;

   always @ (*) begin
       if (AN == 4'b1110) Q <= ssd_out[3:0];
       else if (AN == 4'b1101) Q <= ssd_out[7:4];
       else if (AN == 4'b1011) Q <= ssd_out[11:8];
       else if (AN == 4'b0111) Q <= ssd_out[15:12];
   end
   // ssd numbers
  always @ (*) begin
       case (Q)
           4'h0: C = 7'b1000000;
           4'h1: C = 7'b1111001;
           4'h2: C = 7'b0100100;
           4'h3: C = 7'b0110000;
           4'h4: C = 7'b0011001;
           4'h5: C = 7'b0010010;
           4'h6: C = 7'b0000010;
           4'h7: C = 7'b1111000;
           4'h8: C = 7'b0000000;
           4'h9: C = 7'b0010000;
           4'ha: C = 7'b0001000;
           4'hb: C = 7'b0000011;
           4'hc: C = 7'b1000110;
           4'hd: C = 7'b0100001;
           4'he: C = 7'b0000110;
           4'hf: C = 7'b0001110;
           default: C = 7'b1111111;
       endcase
   end

endmodule
```

program_counter.v

```
`ifndef MODULE_PC
`define MODULE_PC
`timescale 1ns / 1ps
module program_counter(
    PC_in,PC_out,reset,clk,hold
    );
    input [31:0] PC_in;
    output [31:0] PC_out;
    input reset, clk, hold;
```

```
    reg [31:0] PC_out;
    initial begin
        PC_out <= 0;
    end
    always @(negedge clk)
    begin
        if(reset)
            begin
                PC_out <= 32'b0;
            end
      else if (!hold)
       begin
       PC_out<=PC_in[31:0];
       end
    end
endmodule
`endif
```

register.v

```
`ifndef MODULE_REGISTERS
`define MODULE_REGISTERS
`timescale 1ns / 1ps

module Registers (
    input       clk,
          regWrite,
    input       [4:0]   read_register_1,  read_register_2, switch,
    input       [4:0]   write_register,
    input       [31:0]  write_data,
    output      [31:0]  read_data_1,read_data_2,Q2
);
  parameter size = 32;          // 32-bit CPU, $0 - $31
    reg [31:0] register_memory [0:size-1];
    integer i;

    initial begin
        for (i = 0; i < size; i = i + 1)
            register_memory[i] = 32'b0;
    end
    assign read_data_1 = register_memory[read_register_1];
    assign read_data_2 = register_memory[read_register_2];
    assign Q2 = register_memory[switch];
    always @(posedge clk) begin
        if (regWrite == 1)
            register_memory[write_register] <= write_data;
    end
    // falling edge for reading
    // always @(negedge clk) begin
    //  assign read_data_1 = register_memory[read_register_1];
    //  assign read_data_2 = register_memory[read_register_2];
    // end

    // Waiting for fall edge may cause some problem.
endmodule // registers
`endif
```

ring_counter.v

```
`timescale 1ns / 1ps
```

```
2
3  module ring_counter(Q, clock, reset);
4      input clock, reset;
5      output [3:0]Q;
6
7      reg [3:0]Q;
8
9      initial begin
10     Q <= 4'b1110;
11     end
12
13     always @ (posedge reset or posedge clock)
14     begin
15       if (reset == 1'b1) Q <= 4'b1110;
16       else if (Q == 4'b1110) Q <= 4'b0111;
17       else if (Q == 4'b1101) Q <= 4'b1110;
18       else if (Q == 4'b1011) Q <= 4'b1101;
19       else if (Q == 4'b0111) Q <= 4'b1011;
20     end
21
22 endmodule
```