VE370 Project Report

Project 2 - Fall 2020

Name: Qinhang Wu
Id: 518370910041
Email: william_wu@sjtu.edu.cn
Last modified: Nov.12, 2020

Table of Contents

- Single-cycle Processer Design
- Verilog Implementation
- Verilog Source Code
- Peer Evaluation

VE370 Project 2 Individual Report

VE370 Project 2 Individual Report
- Overview
- Verilog Simulation
- Verilog Source Code
  - main module
  - program counter
  - instruction memory
  - next pc
  - control
  - register
  - alu control
  - alu
  - data memory
  - testbench
- Peer Evaluation

Overview

VE370 Project 2 is designed for students to get a better understanding of **Single-cycle and Pipelined Processor Design**.

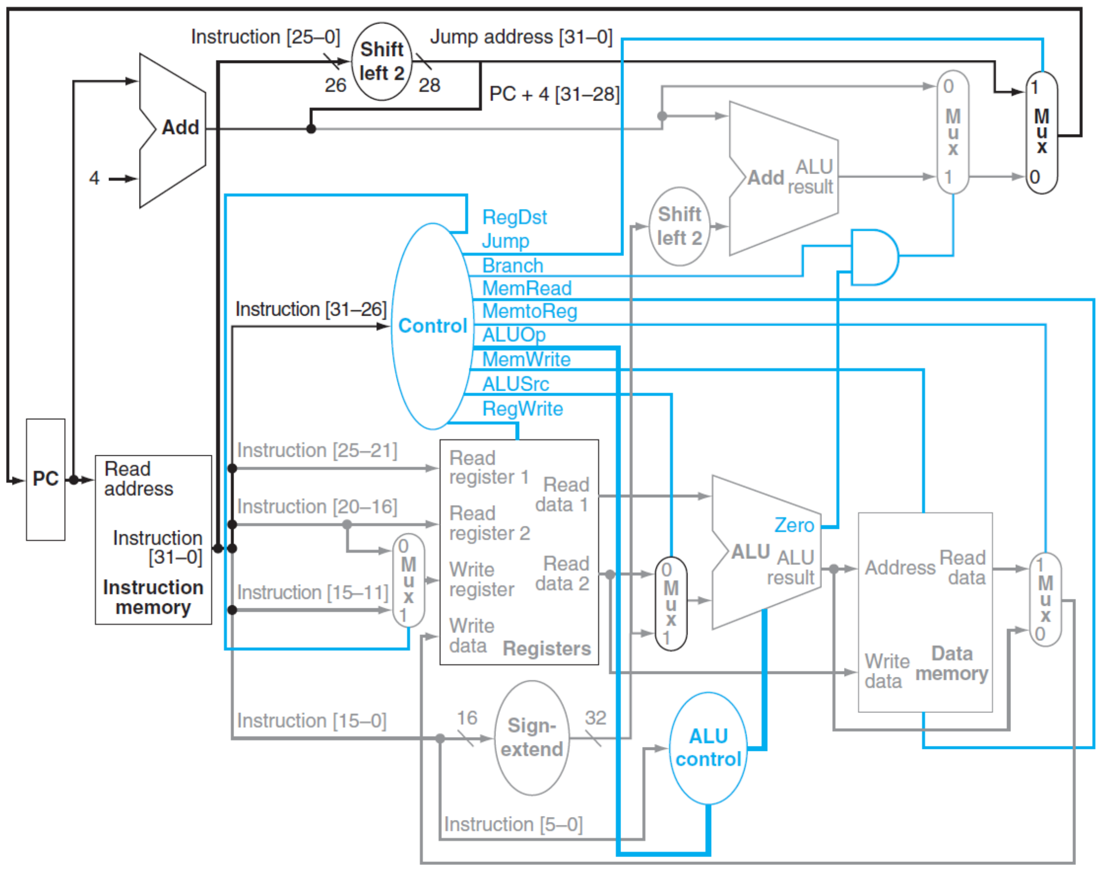

Figure 1. Single Cycle Diagram (MIPS)

Verilog Simulation

A set of instructions are used to test the single-cycle implementation.

Testcase:

```
00100000000010000000000000100000 //addi $t0, $zero, 0x20
00100000000010010000000000110111 //addi $t1, $zero, 0x37
00000001000010011000000000100100 //and $s0, $t0, $t1
00000001000010011000000000100101 //or $s0, $t0, $t1
10101100000100000000000000000100 //sw $s0, 4($zero)
10101100000010000000000000001000 //sw $t0, 8($zero)
00000001000010011000100000100000 //add $s1, $t0, $t1
00000001000010011001000000100010 //sub $s2, $t0, $t1
00100000000010000000000000100000 //addi $t0, $zero, 0x20
00100000000010000000000000100000 //addi $t0, $zero, 0x20
00100000000010000000000000100000 //addi $t0, $zero, 0x20
00010010001100100000000000010010 //beq $s1, $s2, error0
10001100000100010000000000000100 //lw $s1, 4($zero)
00110010001100100000000001001000 //andi $s2, $s1, 0x48
00100000000010000000000000100000 //addi $t0, $zero, 0x20
00100000000010000000000000100000 //addi $t0, $zero, 0x20
00100000000010000000000000100000 //addi $t0, $zero, 0x20
00010010001100100000000000001111 //beq $s1, $s2, error1
10001100000100110000000000001000 //lw $s3, 8($zero)
00100000000010000000000000100000 //addi $t0, $zero, 0x20
00100000000010000000000000100000 //addi $t0, $zero, 0x20
```

```
00100000000010000000000000100000 //addi $t0, $zero, 0x20
00010010000100110000000000001101 //beq $s0, $s3, error2
00000010010100011010000000101010 //slt $s4, $s2, $s1 (Last)
00100000000010000000000000100000 //addi $t0, $zero, 0x20
00100000000010000000000000100000 //addi $t0, $zero, 0x20
00100000000010000000000000100000 //addi $t0, $zero, 0x20
00010010100000000000000000001111 //beq $s4, $0, EXIT
00000010001000001001000000100000 //add $s2, $s1, $0
00001000000000000000000000010111 //j Last
00100000000010000000000000000000 //addi $t0, $0, 0(error0)
00100000000010010000000000000000 //addi $t1, $0, 0
00001000000000000000000000111111 //j EXIT
00100000000010000000000000000001 //addi $t0, $0, 1(error1)
00100000000010010000000000000001 //addi $t1, $0, 1
00001000000000000000000000111111 //j EXIT
00100000000010000000000000000010 //addi $t0, $0, 2(error2)
00100000000010010000000000000010 //addi $t1, $0, 2
00001000000000000000000000111111 //j EXIT
00100000000010000000000000000011 //addi $t0, $0, 3(error3)
00100000000010010000000000000011 //addi $t1, $0, 3
00001000000000000000000000111111 //j EXIT
```

Simulation result:

```
==========================================================
----------------------------------------------------------
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:         20, 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:         40, 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:         60, 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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          80, CLK = 1, PC = 0x00000010
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         100, CLK = 1, PC = 0x00000014
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         120, CLK = 1, PC = 0x00000018
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         140, CLK = 1, PC = 0x0000001c
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         160, CLK = 1, PC = 0x00000020
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xffffffe9
[$s3] = 0x00000000, [$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         180, CLK = 1, PC = 0x00000024
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xffffffe9
[$s3] = 0x00000000, [$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          200, CLK = 1, PC = 0x00000028
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xffffffe9
[$s3] = 0x00000000, [$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          220, CLK = 1, PC = 0x0000002c
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xffffffe9
[$s3] = 0x00000000, [$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          240, CLK = 1, PC = 0x00000030
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xffffffe9
[$s3] = 0x00000000, [$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          260, CLK = 1, PC = 0x00000034
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0xffffffe9
[$s3] = 0x00000000, [$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          280, CLK = 1, PC = 0x00000038
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          300, CLK = 1, PC = 0x0000003c
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          320, CLK = 1, PC = 0x00000040
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          340, CLK = 1, PC = 0x00000044
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          360, CLK = 1, PC = 0x00000048
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          380, CLK = 1, PC = 0x0000004c
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          400, CLK = 1, PC = 0x00000050
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          420, CLK = 1, PC = 0x00000054
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          440, CLK = 1, PC = 0x00000058
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         460, CLK = 1, PC = 0x0000005c
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         480, CLK = 1, PC = 0x00000060
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         500, CLK = 1, PC = 0x00000064
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         520, CLK = 1, PC = 0x00000068
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         540, CLK = 1, PC = 0x0000006c
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         560, CLK = 1, PC = 0x00000070
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         580, CLK = 1, PC = 0x00000074
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          600, CLK = 1, PC = 0x0000005c
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          620, CLK = 1, PC = 0x00000060
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          640, CLK = 1, PC = 0x00000064
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          660, CLK = 1, PC = 0x00000068
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          680, CLK = 1, PC = 0x0000006c
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:          700, CLK = 1, PC = 0x000000ac
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         720, CLK = 1, PC = 0x000000b0
[$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, [$t9] = 0x00000000
----------------------------------------------------------
Time:         740, CLK = 1, PC = 0x000000b4
[$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
----------------------------------------------------------
==========================================================
```

Note: since `syscall` is not implemented here, a series of NOP instructions will be run by default after PC reaches EXIT. Several repeated NOP results are omitted here.

Verilog Source Code

main module

On top of the program, `main.v` interconnects different modules:

```verilog
// main driver program for single-cycle processor

`timescale 1ns / 1ps
`include "alu_control.v"
`include "alu.v"
`include "control.v"
`include "data_memory.v"
`include "instru_memory.v"
`include "next_pc.v"
`include "program_counter.v"
`include "register.v"

module main(
  input clk // clock signal for PC and RD
);

  wire [31:0] pc_in,
              pc_out;

  wire [5:0] im_ctr;
  wire [5:0] im_funcode;
  wire [31:0] im_instru;

  wire [31:0] r_wbdata, // dm_out
              r_read1,
              r_read2;

  wire  c_RegDst,
        c_Jump,
        c_Branch,
        c_Bne,
        c_MemRead,
        c_MemtoReg,
        c_MemWrite,
        c_ALUSrc,
        c_RegWrite;
  wire [1:0] c_ALUOp;

  wire [3:0] c_ALUcontrol;

  wire c_zero;
  wire [31:0] alu_result;

  // wire [31:0] dm_out;

program_counter asset_pc(
  .clk (clk),
  .next (pc_in),
  .out (pc_out)
);

instru_memory asset_im(
  .addr (pc_out),
  .ctr (im_ctr),
  .funcode (im_funcode),
  .instru (im_instru)
);

register asset_reg(
  .clk (clk),
  .instru (im_instru),
  .RegWrite (c_RegWrite),
  .RegDst (c_RegDst),
  .WriteData (r_wbdata),
  .ReadData1 (r_read1),
  .ReadData2 (r_read2)
);

alu asset_alu(
  .data1 (r_read1),
  .read2 (r_read2),
  .instru (im_instru),
  .ALUSrc (c_ALUSrc),
  .ALUcontrol (c_ALUcontrol),
  .zero (c_zero),
  .ALUresult (alu_result)
);

alu_control asset_aluControl(
  .ALUOp (c_ALUOp),
  .instru (im_funcode),
  .ALUcontrol (c_ALUcontrol)
);

control asset_control(
  .instru (im_instru),
  .RegDst (c_RegDst),
  .Jump (c_Jump),
  .Branch (c_Branch),
  .Bne (c_Bne),
  .MemRead (c_MemRead),
  .MemtoReg (c_MemtoReg),
  .ALUOp (c_ALUOp),
  .MemWrite (c_MemWrite),
  .ALUSrc (c_ALUSrc),
  .RegWrite (c_RegWrite)
);

data_memory asset_dm(
  .clk (clk),
  .addr (alu_result), // im_instru
  .wData (r_read2),
  .ALUresult (alu_result),
  .MemWrite (c_MemWrite),
  .MemRead (c_MemRead),
  .MemtoReg (c_MemtoReg),
  .rData (r_wbdata)
);

next_pc asset_nextPc(
  .old (pc_out),
  .instru (im_instru),
  .Jump (c_Jump),
  .Branch (c_Branch),
  .Bne (c_Bne),
  .zero (c_zero),
  .next (pc_in)
);

endmodule
```

program counter

`program_counter.v` describes the functionality of the program counter:

```
`timescale 1ns / 1ps

module program_counter(
  input clk,
  input [31:0] next, // the input address
  output reg [31:0] out // the output address
);

  initial begin
    out = -4; // NEVER REACHED ADDRESS
  end

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

endmodule
```

instruction memory

`instru_memory.v` describes the functionality of the instruction memory:

```
`timescale 1ns / 1ps

module instru_memory(
  // input clk,
  input [31:0] addr,
  output reg [5:0] ctr, // [31-26]
  output reg [5:0] funcode, // [5-0]
  // output reg [4:0] read1, // [25-21]
  // output reg [4:0] read2, // [20-16]
  // output reg [4:0] write, // [15-11]
  output reg [31:0] instru // [31-0]
  // output [15:0] num // [15-0]
);

  parameter SIZE_IM = 128; // size of this memory, by default 128*32
  reg [31:0] mem [SIZE_IM-1:0]; // instruction memory

  integer n;
  initial begin
    for(n=0;n<SIZE_IM;n=n+1) begin
      mem[n] = 32'b11111100000000000000000000000000;
    end
    $readmemb("C:\\Users\\William Wu\\Documents\\Mainframe Files\\UMJI-SJTU\\1 Academy\\20 Fall\\VE370\\Project\\p2\\single_cycle\\testcases\\testcase.txt",mem);
    // NOTE: the absolute path is used here.
    instru = 32'b11111100000000000000000000000000;
  end

  always @(addr) begin
    if (addr == -4) begin // init
      instru = 32'b11111100000000000000000000000000;
    end else begin
      instru = mem[addr >> 2];
    end
    ctr = instru[31:26];
    funcode = instru[5:0];
  end

endmodule
```

next pc

`next_pc.v` describes the functionality of calculating the next address:

```
`timescale 1ns / 1ps

module next_pc(
  input [31:0] old, // the original program addr.
  input [31:0] instru, // the original instruction
    // [15-0] used for sign-extention
    // [25-0] used for shift-left-2
  input Jump,
  input Branch,
  input Bne,
  input zero,
  output reg [31:0] next
);

  reg [31:0] sign_ext;
  reg [31:0] old_alter; // pc+4
  reg [31:0] jump; // jump addr.
  reg zero_alter;

  initial begin
    next = 32'b0;
  end

  always @(old) begin
    old_alter = old + 4;
  end

  always @(zero,Bne) begin
    zero_alter = zero;
    if (Bne == 1) begin
      zero_alter = ! zero_alter;
    end
  end

  always @(instru) begin
    // jump-shift-left
    jump = {4'b0,instru[25:0],2'b0};

    // sign-extension
    if (instru[15] == 1'b0) begin
      sign_ext = {16'b0,instru[15:0]};
    end else begin
      sign_ext = {{16{1'b1}},instru[15:0]};
    end
    sign_ext = {sign_ext[29:0],2'b0}; // shift left
  end

  always @(instru or old_alter or jump) begin
    jump = {old_alter[31:28],jump[27:0]};
  end

  always @(old_alter,sign_ext,jump,Branch,zero_alter,Jump) begin
    // assign next program counter value
    if (Branch == 1 & zero_alter == 1) begin
      // $display("Taking branch");
      next = old_alter + sign_ext;
    end else begin
      // $display("Normal proceeding");
      next = old_alter;
    end
    if (Jump == 1) begin
      // $display("Taking jump");
      next = jump;
    end
  end

endmodule
```

control

`control.v` describes the functionality of generating different control signals:

```
`timescale 1ns / 1ps

module control(
  input [31:0] instru,
  output reg RegDst,
  output reg Jump,
  output reg Branch,
  output reg Bne, // 1 indicates bne
  output reg MemRead,
  output reg MemtoReg,
  output reg [1:0] ALUOp,
  output reg MemWrite,
  output reg ALUSrc,
```

```verilog
  output reg RegWrite
);

  initial begin
    RegDst = 0;
    Jump = 0;
    Branch = 0;
    MemRead = 0;
    MemtoReg = 0;
    ALUOp = 2'b00;
    MemWrite = 0;
    ALUSrc = 0;
    RegWrite = 0;
  end

  always @(instru) begin
    case (instru[31:26])
      6'b000000: begin// ARITHMETIC
        RegDst = 1;
        ALUSrc = 0;
        MemtoReg = 0;
        RegWrite = 1;
        MemRead = 0;
        MemWrite = 0;
        Branch = 0;
        Bne = 0;
        ALUOp = 2'b10;
        Jump = 0;
      end
      6'b001000: begin// addi
        RegDst = 0;
        ALUSrc = 1;
        MemtoReg = 0;
        RegWrite = 1;
        MemRead = 0;
        MemWrite = 0;
        Branch = 0;
        Bne = 0;
        ALUOp = 2'b00;
        Jump = 0;
      end
      6'b001100: begin// andi
        RegDst = 0;
        ALUSrc = 1;
        MemtoReg = 0;
        RegWrite = 1;
        MemRead = 0;
        MemWrite = 0;
        Branch = 0;
        Bne = 0;
        ALUOp = 2'b11;
        Jump = 0;
      end
      6'b100011: begin // lw
        RegDst = 0;
        ALUSrc = 1;
        MemtoReg = 1;
        RegWrite = 1;
        MemRead = 1;
        MemWrite = 0;
        Branch = 0;
        Bne = 0;
        ALUOp = 2'b00;
        Jump = 0;
      end
      6'b101011: begin // sw
        RegDst = 0; // x
        ALUSrc = 1;
        MemtoReg = 0; // x
        RegWrite = 0;
        MemRead = 0;
        MemWrite = 1;
        Branch = 0;
        Bne = 0;
        ALUOp = 2'b00;
        Jump = 0;
      end
      6'b000100: begin // beq
        RegDst = 0; // x
        ALUSrc = 0;
        MemtoReg = 0; // x
        RegWrite = 0;
        MemRead = 0;
        MemWrite = 0;
        Branch = 1;
        Bne = 0;
        ALUOp = 2'b01;
        Jump = 0;
      end
      6'b000101: begin // bne
        RegDst = 0; // x
        ALUSrc = 0;
        MemtoReg = 0; // x
        RegWrite = 0;
        MemRead = 0;
        MemWrite = 0;
        Branch = 1;
        Bne = 1;
        ALUOp = 2'b01;
        Jump = 0;
      end
      6'b000010: begin // j
        RegDst = 0; // x
        ALUSrc = 0;
        MemtoReg = 0; // X
        RegWrite = 0;
        MemRead = 0;
        MemWrite = 0;
        Branch = 0;
        Bne = 0;
        ALUOp = 2'b01;
        Jump = 1;
      end
      default: begin
        RegDst = 0; // X
        ALUSrc = 0;
        MemtoReg = 0; // X
        RegWrite = 0;
        MemRead = 0;
        MemWrite = 0;
        Branch = 0;
        Bne = 0;
        ALUOp = 2'b00;
        Jump = 0;
      end
    endcase
  end

endmodule
```

register

`register.v` describes the functionality of register management:

```
`timescale 1ns / 1ps

module register(
  input clk,
  input [31:0] instru, // the raw 32-bit instruction
  input RegWrite,
  input RegDst,
  input [31:0] WriteData, // from WB stage
  // input [4:0] WriteReg,
  output [31:0] ReadData1,
  output [31:0] ReadData2
);

  reg [31:0] RegData [31:0]; // register data

  // initialize the regester data
  integer i;
  initial begin
    for(i=0;i<32;i=i+1) begin
      RegData[i] = 32'b0;
    end
  end

  assign ReadData1 = RegData[instru[25:21]];
  assign ReadData2 = RegData[instru[20:16]];

  always @(posedge clk) begin // RegWrite, RegDst, WriteData, instru)
    if (RegWrite == 1'b1) begin
      if (RegDst == 1'b0) begin
        RegData[instru[20:16]] = WriteData;
      end else begin
        RegData[instru[15:11]] = WriteData;
      end
    end
  end

endmodule
```

alu control

`alu_control` describes the functionality of generating alu control signals:

```
`timescale 1ns / 1ps

module alu_control(
  input [1:0] ALUOp,
  input [5:0] instru,
  output reg [3:0] ALUcontrol
);

  always @(ALUOp, instru) begin
    case (ALUOp)
      2'b00:
        ALUcontrol = 4'b0010;
      2'b01:
        ALUcontrol = 4'b0110;
      2'b10: begin
        case (instru)
          6'b100000: // add
            ALUcontrol = 4'b0010;
          6'b100010: // sub
            ALUcontrol = 4'b0110;
          6'b100100: // and
            ALUcontrol = 4'b0000;
          6'b100101: // or
            ALUcontrol = 4'b0001;
          6'b101010: // slt
            ALUcontrol = 4'b0111;
          default:
            ;
        endcase
      end
      2'b11:
        ALUcontrol = 4'b0000;
      default:
        ;
    endcase
  end

endmodule
```

alu

`alu.v` describes the functionality of the alu unit:

```
`timescale 1ns / 1ps

module alu(
  input [31:0] data1,
  input [31:0] read2,
  input [31:0] instru, // used for sign-extension
  input ALUSrc,
  input [3:0] ALUcontrol,
  output reg zero,
  output reg [31:0] ALUresult
);

  reg [31:0] data2;

  always @(ALUSrc, read2, instru) begin
    if (ALUSrc == 0) begin
      data2 = read2;
    end else begin
      // SignExt[Instru[15:0]]
      if (instru[15] == 1'b0) begin
        data2 = {16'b0,instru[15:0]};
      end else begin
        data2 = {{16{1'b1}},instru[15:0]};
      end
    end
  end

  always @(data1, data2, ALUcontrol) begin
    case (ALUcontrol)
      4'b0000: // AND
        ALUresult = data1 & data2;
      4'b0001: // OR
        ALUresult = data1 | data2;
      4'b0010: // ADD
        ALUresult = data1 + data2;
      4'b0110: // SUB
        ALUresult = data1 - data2;
      4'b0111: // SLT
        ALUresult = (data1 < data2) ? 1 : 0;
      4'b1100: // NOR
        ALUresult = data1 |~ data2;
      default:
        ;
    endcase
    if (ALUresult == 0) begin
      zero = 1;
    end else begin
      zero = 0;
    end
  end

endmodule
```

data memory

`data_memory.v` describes the functionality of the data memory:

```
`timescale 1ns / 1ps

module data_memory(
  input clk,
  input [31:0] addr,
  input [31:0] wData,
  input [31:0] ALUresult,
  input MemWrite,
  input MemRead,
  input MemtoReg,
  output reg [31:0] rData
);

  parameter SIZE_DM = 128; // size of this memory, by default 128*32
  reg [31:0] mem [SIZE_DM-1:0]; // instruction memory

  // initially set default data to 0
  integer i;
  initial begin
    for(i=0; i<SIZE_DM-1; i=i+1) begin
      mem[i] = 32'b0;
    end
  end

  always @(addr or MemRead or MemtoReg or ALUresult) begin
    if (MemRead == 1) begin
      if (MemtoReg == 1) begin
        rData = mem[addr];
      end else begin
        rData = ALUresult; // X ?
      end
    end else begin
      rData = ALUresult;
    end
  end

  always @(posedge clk) begin // MemWrite, wData, addr
    if (MemWrite == 1) begin
      mem[addr] = wData;
    end
  end

endmodule
```

testbench

Finally, a testbench is included in `testbench.v` to test and simulate the program:

```
// testbench for simulation of the single-cycle processor

`timescale 1ns / 1ps
`include "main.v"

module testbench;
  integer currTime;
  reg clk;

  main uut(
    .clk (clk)
  );

  initial begin
    #0
    clk = 0;
    currTime = -10;
    uut.asset_pc.out = -4;
    $display("=========================================================");

    #988 $display("=========================================================");
    #989 $stop;
  end

  always @(posedge clk) begin
    // indicating a posedge clk triggered
    $display("---------------------------------------------------------");
    #1; // wait for writing back
    $display("Time: %d, CLK = %d, PC = 0x%H",currTime, clk, uut.asset_pc.out);
    $display("[$s0] = 0x%H, [$s1] = 0x%H, [$s2] = 0x%H",uut.asset_reg.RegData[16],uut.asset_reg.RegData[17],uut.asset_reg.RegData[18]);
    $display("[$s3] = 0x%H, [$s4] = 0x%H, [$s5] = 0x%H",uut.asset_reg.RegData[19],uut.asset_reg.RegData[20],uut.asset_reg.RegData[21]);
    $display("[$s6] = 0x%H, [$s7] = 0x%H, [$t0] = 0x%H",uut.asset_reg.RegData[22],uut.asset_reg.RegData[23],uut.asset_reg.RegData[8]);
    $display("[$t1] = 0x%H, [$t2] = 0x%H, [$t3] = 0x%H",uut.asset_reg.RegData[9],uut.asset_reg.RegData[10],uut.asset_reg.RegData[11]);
    $display("[$t4] = 0x%H, [$t5] = 0x%H, [$t6] = 0x%H",uut.asset_reg.RegData[12],uut.asset_reg.RegData[13],uut.asset_reg.RegData[14]);
    $display("[$t7] = 0x%H, [$t8] = 0x%H, [$t9] = 0x%H",uut.asset_reg.RegData[15],uut.asset_reg.RegData[24],uut.asset_reg.RegData[25]);
  end

  always #10 begin
    clk = ~clk;
    currTime = currTime + 10;
  end

endmodule
```

Peer Evaluation

Name	Level of contribution (0~5)	Description of contribution
Wu Qinhang (me)	5	system design and debug module integration synthesis and FPGA implementation
Xu Jiaying	5	patch for pipelined system RTL schematic debug and report refinement
Liu Yuru	5	hazard detection unit branch bonus unit RTL schematic and report refinement
Yang Gengchen	5	forwarding unit FPGA implementation debug and report refinement

Each of the group member contributes to the pipelined processor equally.