**GEBZE TECHNICAL UNIVERSITY**

**COMPUTER ENGINEERING**

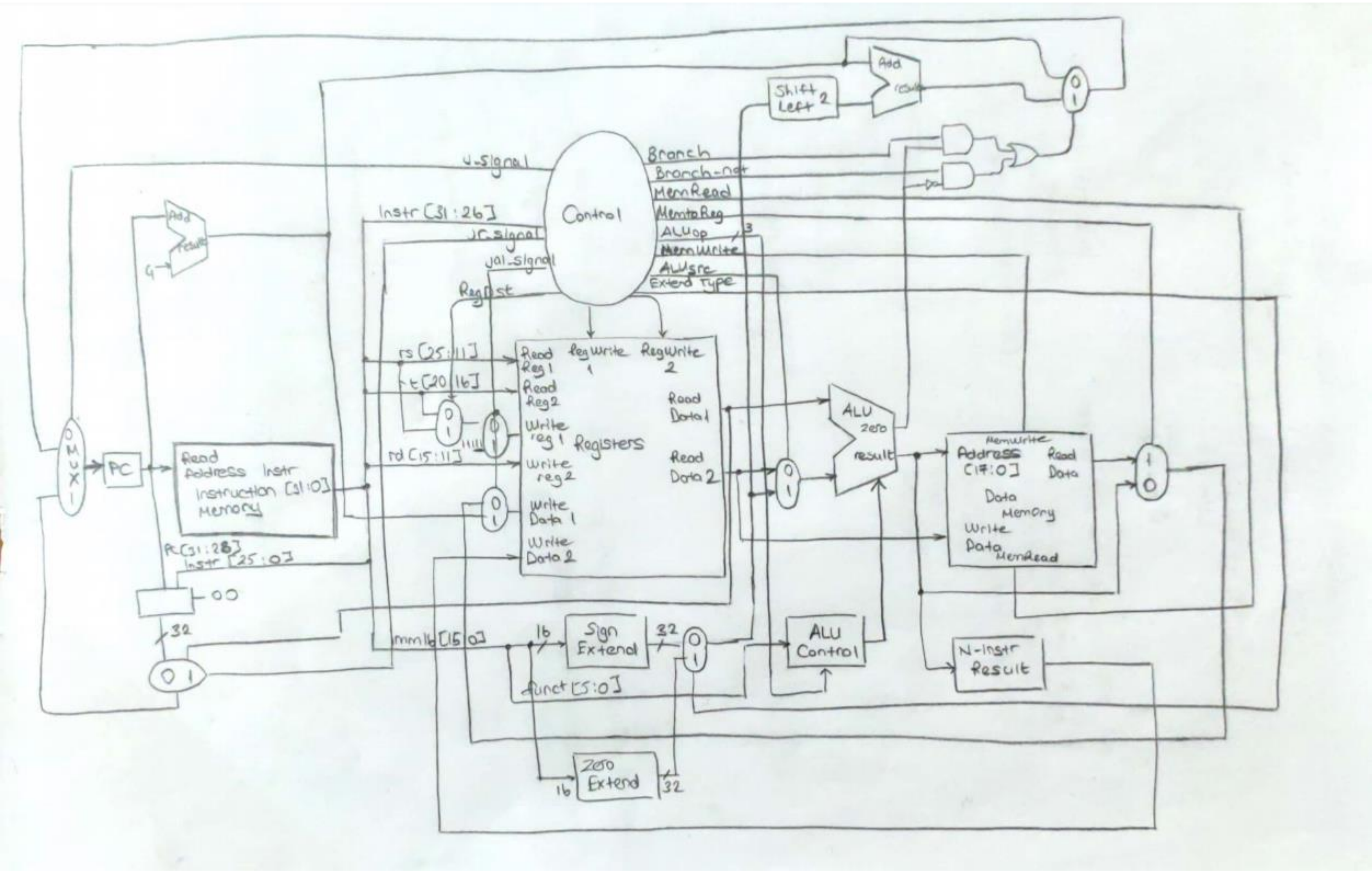
**CSE 331 – 2020 FALL**

# ASSSIGNMENT 4 REPORT

Türker Tercan
171044032

171044032 CSE 331 – 2020 FALL Assignment 4
Türker Tercan

## My Single Cycle MIPS like processor:

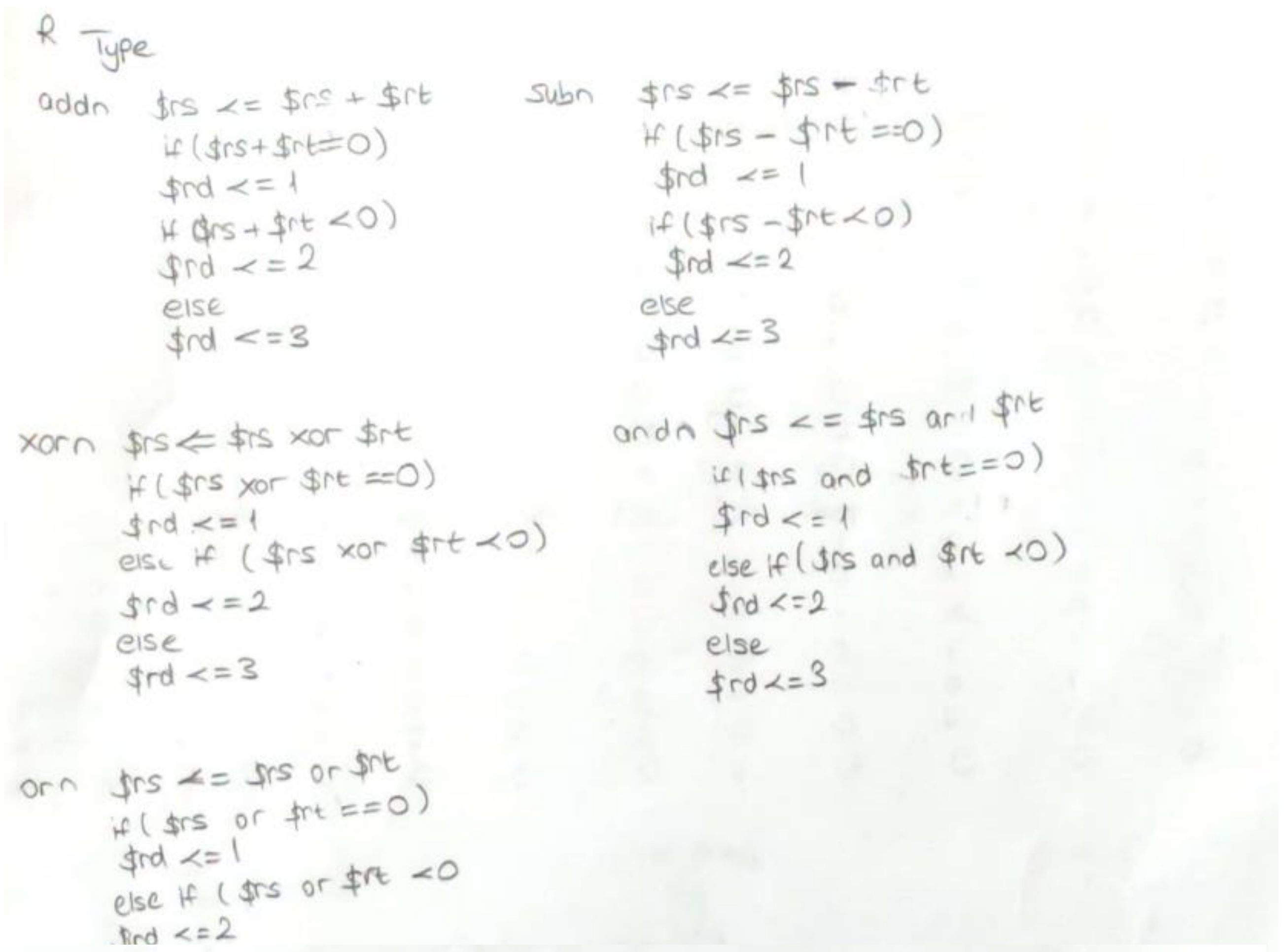


- I implemented my register to store two elements in one cycle
- N instr module takes ALU result and outputs the result what you wanted to

- All needed instructions is implemented.

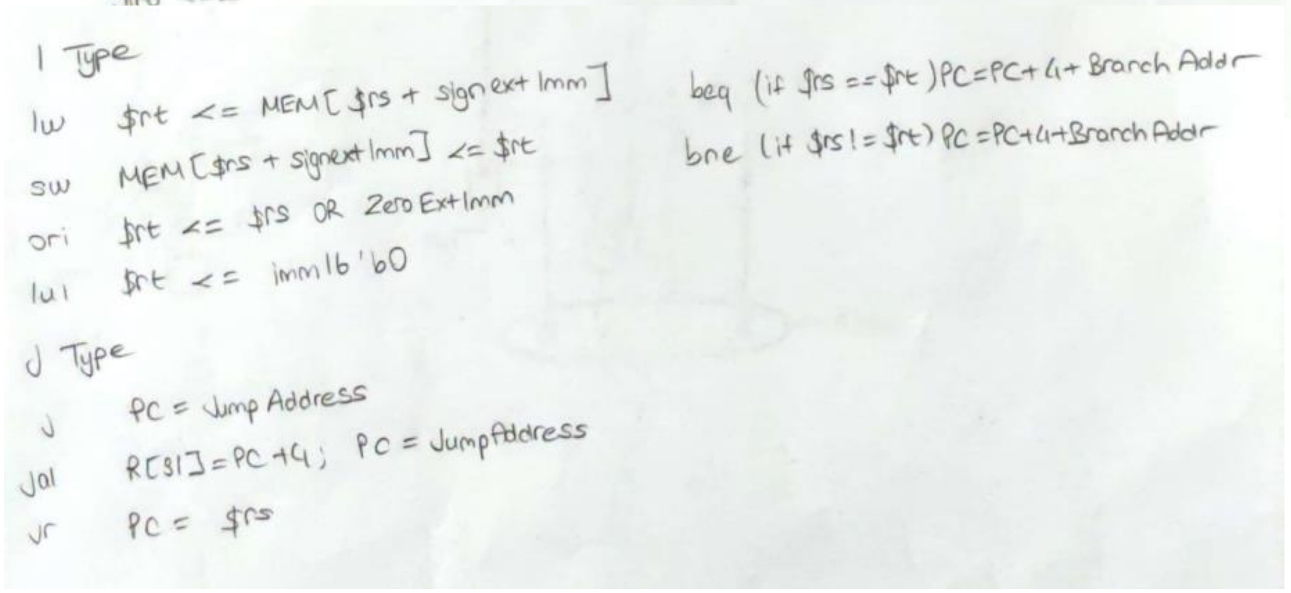
171044032 CSE 331 – 2020 FALL Assignment 4
Türker Tercan

## Instructions:



R – Type

```
addn  $rs <= $rs + $rt
      if ($rs+$rt==0)
      $rd <= 1
      if ($rs + $rt <0)
      $rd <= 2
      else
      $rd <=3

subn  $rs <= $rs - $rt
      if ($rs - $rt ==0)
       $rd <= 1
      if ($rs -$rt<0)
       $rd <=2
      else
      $rd <= 3

xorn  $rs <= $rs xor $rt
      if ($rs xor $rt ==0)
      $rd <=1
      else if ($rs xor $rt<0)
      $rd <=2
      else
      $rd <=3

andn  $rs <= $rs and $rt
      if ($rs and $rt==0)
      $rd <= 1
      else if($rs and $rt <0)
      $rd <=2
      else
      $rd <=3

orn   $rs <= $rs or $rt
      if ( $rs or $rt ==0)
      $rd <=1
      else if ( $rs or $rt <0
      $rd <=2
```



I Type

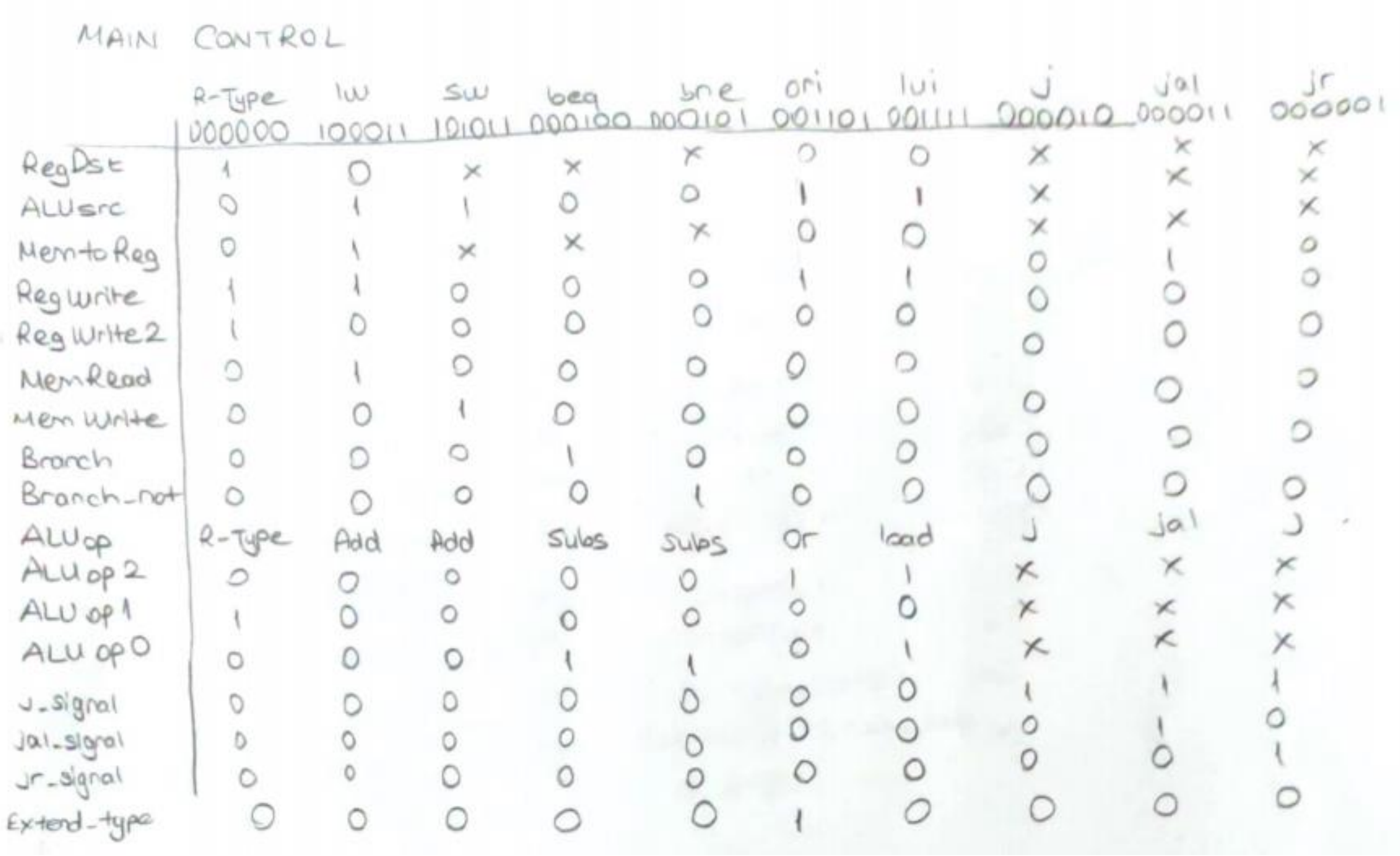
```
lw   $rt <= MEM[ $rs + sign ext Imm ]
sw   MEM [$rs + signext Imm] <= $rt
ori  $rt <= $rs OR Zero Ext Imm
lui  $rt <= imm16'b0

beq  (if $rs ==$rt )PC=PC+4+ Branch Addr
bne  (if $rs != $rt) PC =PC+4+Branch Addr
```

J Type

```
j    PC = Jump Address
jal  R[31]=PC +4; PC = JumpAddress
jr   PC = $rs
```

Türker Tercan

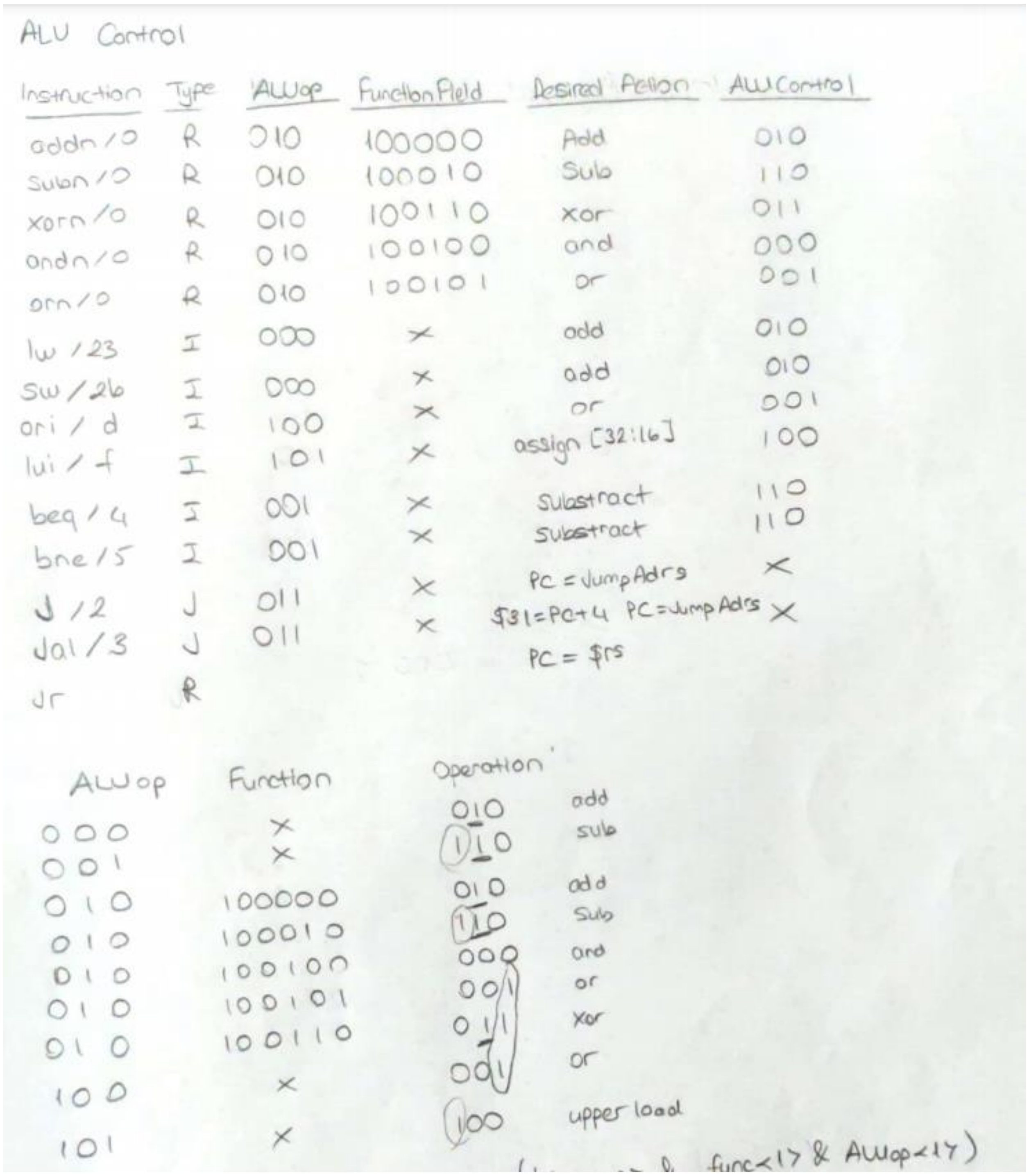
## Truth Table for Main Control:



MAIN CONTROL

| | R-Type 000000 | lw 100011 | sw 101011 | beq 000100 | bne 000101 | ori 001101 | lui 001111 | j 000010 | jal 000011 | jr 000001 |
|---|---|---|---|---|---|---|---|---|---|---|
| RegDst | 1 | 0 | x | x | x | 0 | 0 | x | x | x |
| ALUsrc | 0 | 1 | 1 | 0 | 0 | 1 | 1 | x | x | x |
| MemtoReg | 0 | 1 | x | x | x | 0 | 0 | x | x | x |
| Reg write | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| Reg Write2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| MemRead | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Mem Write | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Branch | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Branch_not | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| ALUop | R-Type | Add | Add | Subs | Subs | Or | load | J | jal | J |
| ALUop 2 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | x | x | x |
| ALU op 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | x | x | x |
| ALU op 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | x | x | x |
| j_signal | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| jal_signal | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| jr_signal | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Extend_type | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

## ALU Control Signals:



ALU Control

| Instruction | Type | ALUop | Function Field | Desired Action | ALU Control |
|---|---|---|---|---|---|
| addn/0 | R | 010 | 100000 | Add | 010 |
| subn/0 | R | 010 | 100010 | Sub | 110 |
| xorn/0 | R | 010 | 100110 | xor | 011 |
| andn/0 | R | 010 | 100100 | and | 000 |
| orn/0 | R | 010 | 100101 | or | 001 |
| lw/23 | I | 000 | x | add | 010 |
| sw/2b | I | 000 | x | add | 010 |
| ori/d | I | 100 | x | or | 001 |
| lui/f | I | 101 | x | assign [32:16] | 100 |
| beq/4 | I | 001 | x | Substract | 110 |
| bne/5 | I | 001 | x | Substract | 110 |
| J/2 | J | 011 | x | PC = JumpAdrs | x |
| Jal/3 | J | 011 | x | $31=PC+4 PC=JumpAdrs | x |
| Jr | R | | | PC = $rs | |

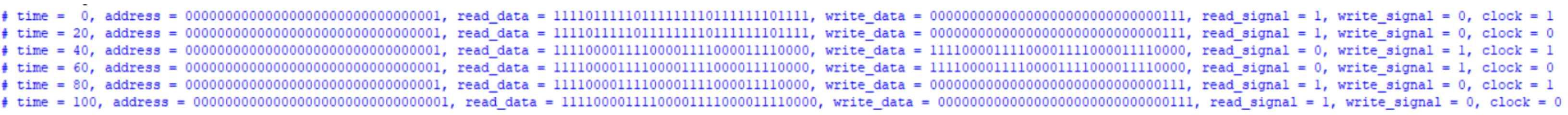
| ALUop | Function | Operation | |
|---|---|---|---|
| 000 | x | 010 | add |
| 001 | x | 110 | sub |
| 010 | 100000 | 010 | add |
| 010 | 100010 | 110 | sub |
| 010 | 100100 | 000 | and |
| 010 | 100101 | 001 | or |
| 010 | 100110 | 011 | xor |
| 100 | x | 001 | or |
| 101 | x | 100 | upper load |

( func<1> & ALUop<1>)

171044032 CSE 331 – 2020 FALL Assignment 4
Türker Tercan

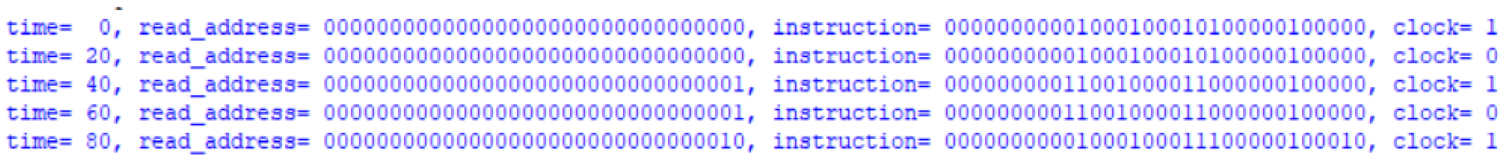
## Testbenches:

### 1. Data Memory Testbench:



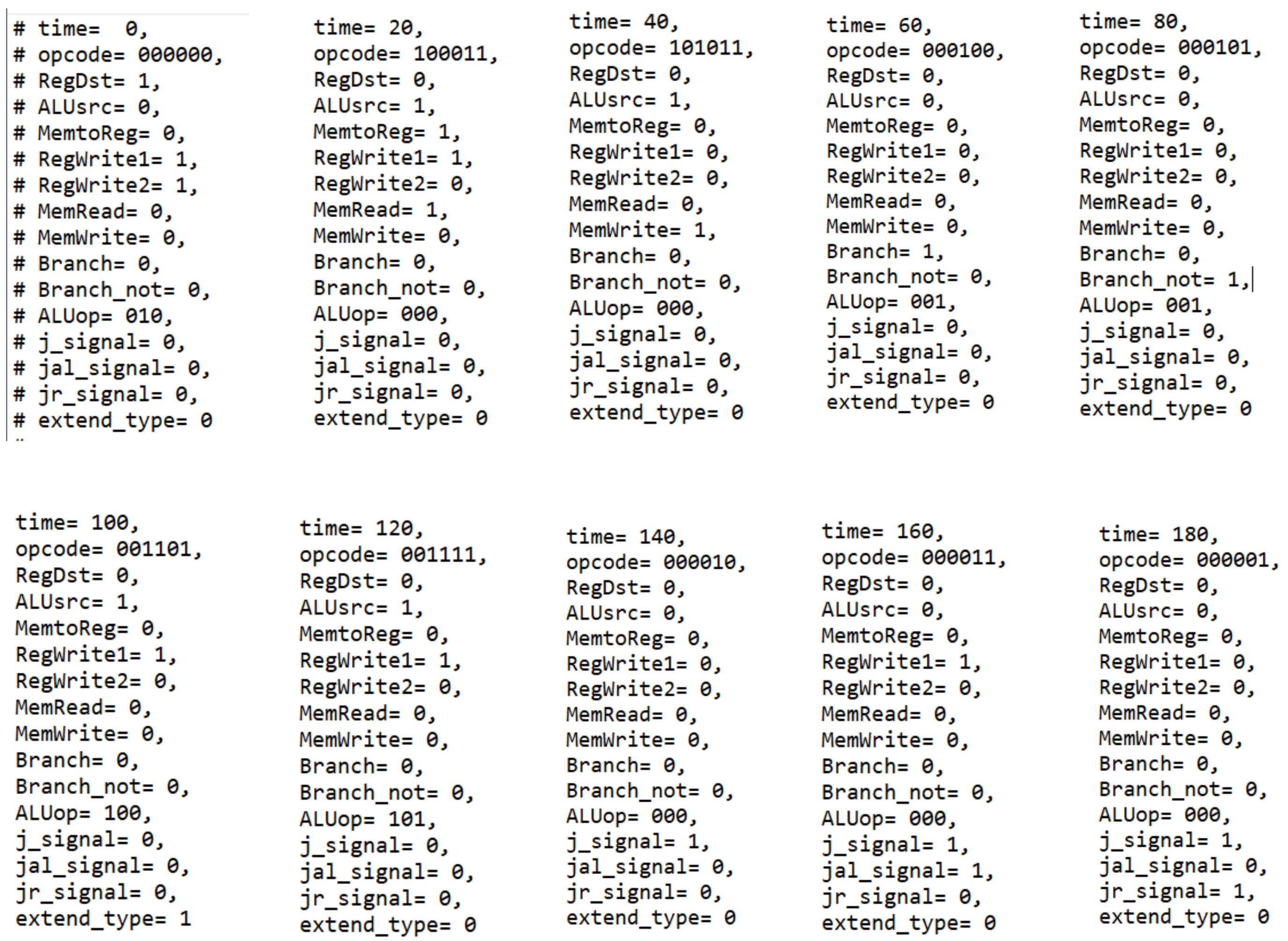
```
# time =  0, address = 00000000000000000000000000000001, read_data = 11110111110111111110111111101111, write_data = 00000000000000000000000000000111, read_signal = 1, write_signal = 0, clock = 1
# time = 20, address = 00000000000000000000000000000001, read_data = 11110111110111111110111111101111, write_data = 00000000000000000000000000000111, read_signal = 1, write_signal = 0, clock = 0
# time = 40, address = 00000000000000000000000000000001, read_data = 11110000111100001111000011110000, write_data = 11110000111100001111000011110000, read_signal = 0, write_signal = 1, clock = 1
# time = 60, address = 00000000000000000000000000000001, read_data = 11110000111100001111000011110000, write_data = 11110000111100001111000011110000, read_signal = 0, write_signal = 1, clock = 0
# time = 80, address = 00000000000000000000000000000001, read_data = 11110000111100001111000011110000, write_data = 00000000000000000000000000000111, read_signal = 1, write_signal = 0, clock = 1
# time = 100, address = 00000000000000000000000000000001, read_data = 11110000111100001111000011110000, write_data = 00000000000000000000000000000111, read_signal = 1, write_signal = 0, clock = 0
```

### 2. Instruction Memory Testbench:



```
time=  0, read_address= 00000000000000000000000000000000, instruction= 00000000010001000101000000100000, clock= 1
time= 20, read_address= 00000000000000000000000000000000, instruction= 00000000010001000101000000100000, clock= 0
time= 40, read_address= 00000000000000000000000000000001, instruction= 00000000110010000110000000100000, clock= 1
time= 60, read_address= 00000000000000000000000000000001, instruction= 00000000110010000110000000100000, clock= 0
time= 80, read_address= 00000000000000000000000000000010, instruction= 00000000010001000111000000100010, clock= 1
```

### 3. Main Control TestBench:



```
# time=  0,
# opcode= 000000,
# RegDst= 1,
# ALUsrc= 0,
# MemtoReg= 0,
# RegWrite1= 1,
# RegWrite2= 1,
# MemRead= 0,
# MemWrite= 0,
# Branch= 0,
# Branch_not= 0,
# ALUop= 010,
# j_signal= 0,
# jal_signal= 0,
# jr_signal= 0,
# extend_type= 0
```

```
time= 20,
opcode= 100011,
RegDst= 0,
ALUsrc= 1,
MemtoReg= 1,
RegWrite1= 1,
RegWrite2= 0,
MemRead= 1,
MemWrite= 0,
Branch= 0,
Branch_not= 0,
ALUop= 000,
j_signal= 0,
jal_signal= 0,
jr_signal= 0,
extend_type= 0
```

```
time= 40,
opcode= 101011,
RegDst= 0,
ALUsrc= 1,
MemtoReg= 0,
RegWrite1= 0,
RegWrite2= 0,
MemRead= 0,
MemWrite= 1,
Branch= 0,
Branch_not= 0,
ALUop= 000,
j_signal= 0,
jal_signal= 0,
jr_signal= 0,
extend_type= 0
```

```
time= 60,
opcode= 000100,
RegDst= 0,
ALUsrc= 0,
MemtoReg= 0,
RegWrite1= 0,
RegWrite2= 0,
MemRead= 0,
MemWrite= 0,
Branch= 1,
Branch_not= 0,
ALUop= 001,
j_signal= 0,
jal_signal= 0,
jr_signal= 0,
extend_type= 0
```

```
time= 80,
opcode= 000101,
RegDst= 0,
ALUsrc= 0,
MemtoReg= 0,
RegWrite1= 0,
RegWrite2= 0,
MemRead= 0,
MemWrite= 0,
Branch= 0,
Branch_not= 1,
ALUop= 001,
j_signal= 0,
jal_signal= 0,
jr_signal= 0,
extend_type= 0
```

```
time= 100,
opcode= 001101,
RegDst= 0,
ALUsrc= 1,
MemtoReg= 0,
RegWrite1= 1,
RegWrite2= 0,
MemRead= 0,
MemWrite= 0,
Branch= 0,
Branch_not= 0,
ALUop= 100,
j_signal= 0,
jal_signal= 0,
jr_signal= 0,
extend_type= 1
```

```
time= 120,
opcode= 001111,
RegDst= 0,
ALUsrc= 1,
MemtoReg= 0,
RegWrite1= 1,
RegWrite2= 0,
MemRead= 0,
MemWrite= 0,
Branch= 0,
Branch_not= 0,
ALUop= 101,
j_signal= 0,
jal_signal= 0,
jr_signal= 0,
extend_type= 0
```

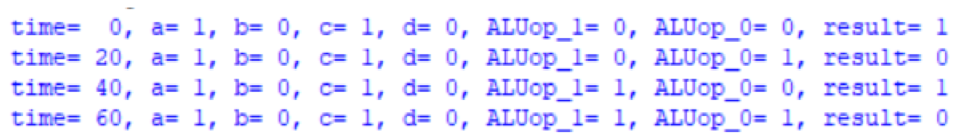
```
time= 140,
opcode= 000010,
RegDst= 0,
ALUsrc= 0,
MemtoReg= 0,
RegWrite1= 0,
RegWrite2= 0,
MemRead= 0,
MemWrite= 0,
Branch= 0,
Branch_not= 0,
ALUop= 000,
j_signal= 1,
jal_signal= 0,
jr_signal= 0,
extend_type= 0
```

```
time= 160,
opcode= 000011,
RegDst= 0,
ALUsrc= 0,
MemtoReg= 0,
RegWrite1= 1,
RegWrite2= 0,
MemRead= 0,
MemWrite= 0,
Branch= 0,
Branch_not= 0,
ALUop= 000,
j_signal= 1,
jal_signal= 1,
jr_signal= 0,
extend_type= 0
```

```
time= 180,
opcode= 000001,
RegDst= 0,
ALUsrc= 0,
MemtoReg= 0,
RegWrite1= 0,
RegWrite2= 0,
MemRead= 0,
MemWrite= 0,
Branch= 0,
Branch_not= 0,
ALUop= 000,
j_signal= 1,
jal_signal= 0,
jr_signal= 1,
extend_type= 0
```

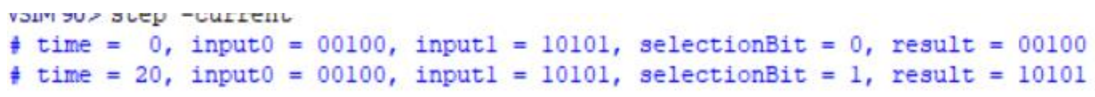
171044032 CSE 331 – 2020 FALL Assignment 4
Türker Tercan

4. Mux 4to1 1bit Testbench:



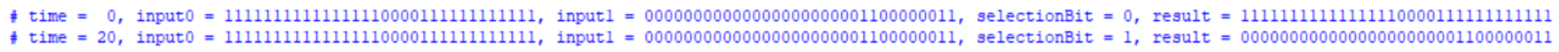
```
time=  0, a= 1, b= 0, c= 1, d= 0, ALUop_1= 0, ALUop_0= 0, result= 1
time= 20, a= 1, b= 0, c= 1, d= 0, ALUop_1= 0, ALUop_0= 1, result= 0
time= 40, a= 1, b= 0, c= 1, d= 0, ALUop_1= 1, ALUop_0= 0, result= 1
time= 60, a= 1, b= 0, c= 1, d= 0, ALUop_1= 1, ALUop_0= 1, result= 0
```

5. Mux 2to1 5bit Testbench:



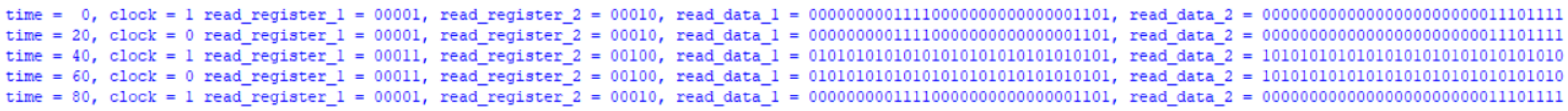
```
VSIM 90> step -current
# time =  0, input0 = 00100, input1 = 10101, selectionBit = 0, result = 00100
# time = 20, input0 = 00100, input1 = 10101, selectionBit = 1, result = 10101
```

6. Mux 2to1 32bit Testbench:



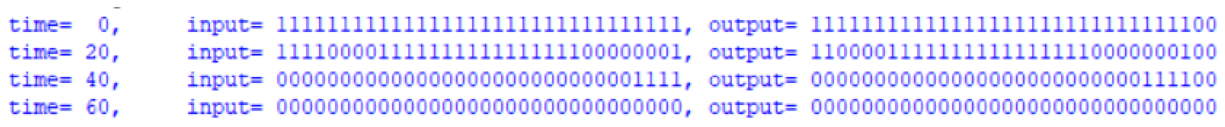
```
# time =  0, input0 = 11111111111111110000111111111111, input1 = 00000000000000000000001100000011, selectionBit = 0, result = 11111111111111110000111111111111
# time = 20, input0 = 11111111111111110000111111111111, input1 = 00000000000000000000001100000011, selectionBit = 1, result = 00000000000000000000001100000011
```

7. Register Testbench:



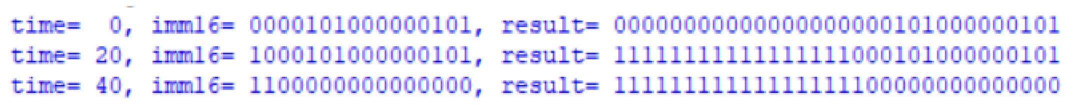
```
time =  0, clock = 1 read_register_1 = 00001, read_register_2 = 00010, read_data_1 = 00000000011110000000000000001101, read_data_2 = 00000000000000000000000011101111
time = 20, clock = 0 read_register_1 = 00001, read_register_2 = 00010, read_data_1 = 00000000011110000000000000001101, read_data_2 = 00000000000000000000000011101111
time = 40, clock = 1 read_register_1 = 00011, read_register_2 = 00100, read_data_1 = 01010101010101010101010101010101, read_data_2 = 10101010101010101010101010101010
time = 60, clock = 0 read_register_1 = 00011, read_register_2 = 00100, read_data_1 = 01010101010101010101010101010101, read_data_2 = 10101010101010101010101010101010
time = 80, clock = 1 read_register_1 = 00001, read_register_2 = 00010, read_data_1 = 00000000011110000000000000001101, read_data_2 = 00000000000000000000000011101111
```

8. Shift Left by 2 32bit Testbench:



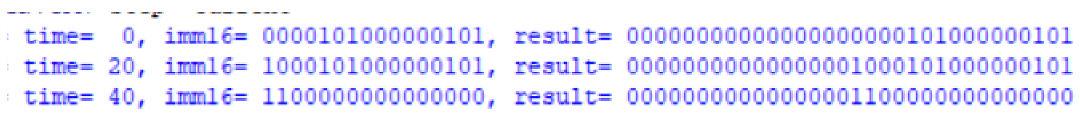
```
time=  0,     input= 11111111111111111111111111111111, output= 11111111111111111111111111111100
time= 20,     input= 11110000111111111111111100000001, output= 11000011111111111111110000000100
time= 40,     input= 00000000000000000000000000001111, output= 00000000000000000000000000111100
time= 60,     input= 00000000000000000000000000000000, output= 00000000000000000000000000000000
```

9. Sign Extend 32bit Testbench:



```
time=  0, imm16= 0000101000000101, result= 00000000000000000000101000000101
time= 20, imm16= 1000101000000101, result= 11111111111111111000101000000101
time= 40, imm16= 1100000000000000, result= 11111111111111111100000000000000
```

10. Zero Extend 32bit Testbench:

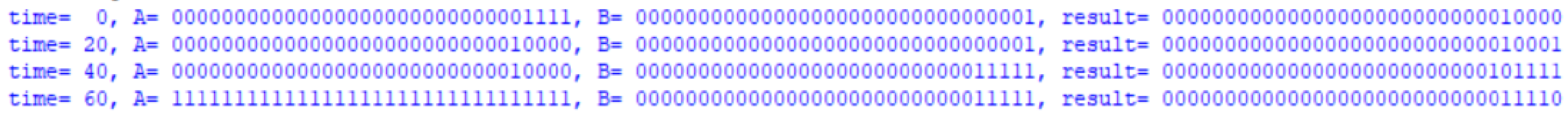


```
time=  0, imm16= 0000101000000101, result= 00000000000000000000101000000101
time= 20, imm16= 1000101000000101, result= 00000000000000001000101000000101
time= 40, imm16= 1100000000000000, result= 00000000000000001100000000000000
```

171044032 CSE 331 – 2020 FALL Assignment 4

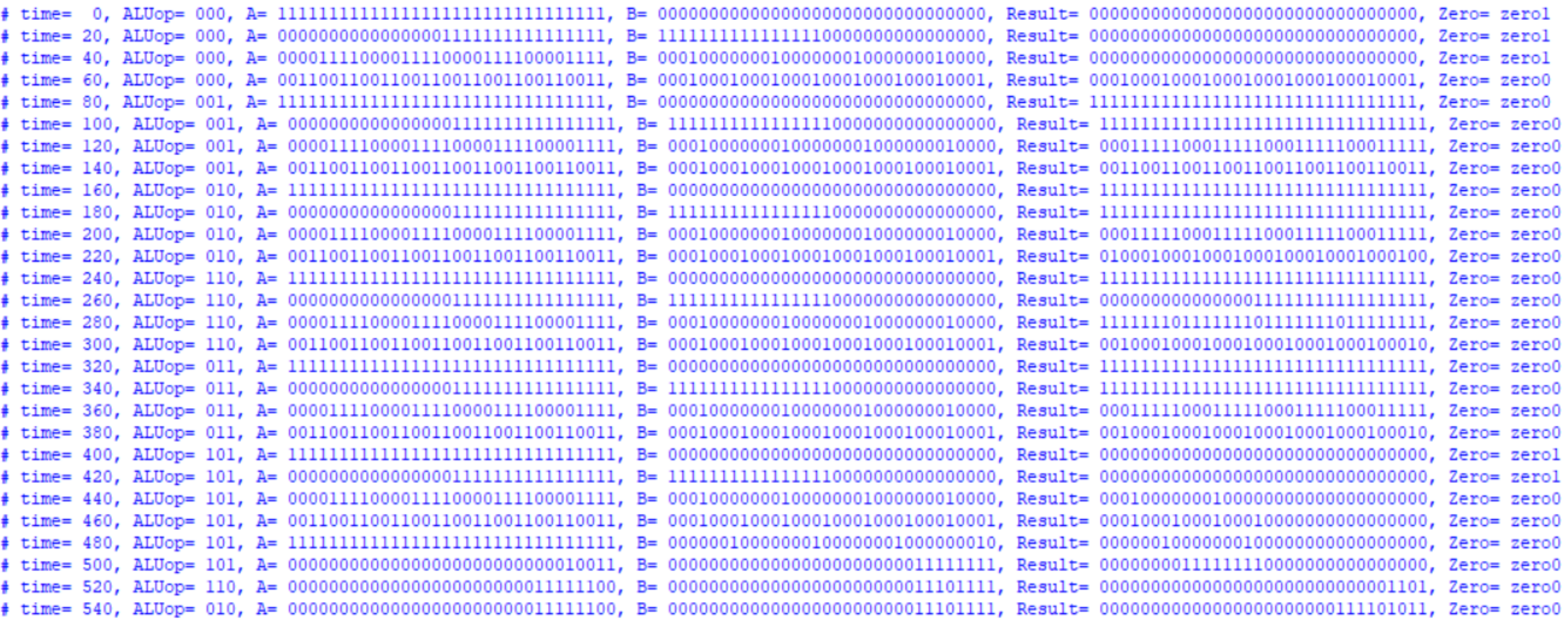
Türker Tercan

11. Adder 32bit Testbench:



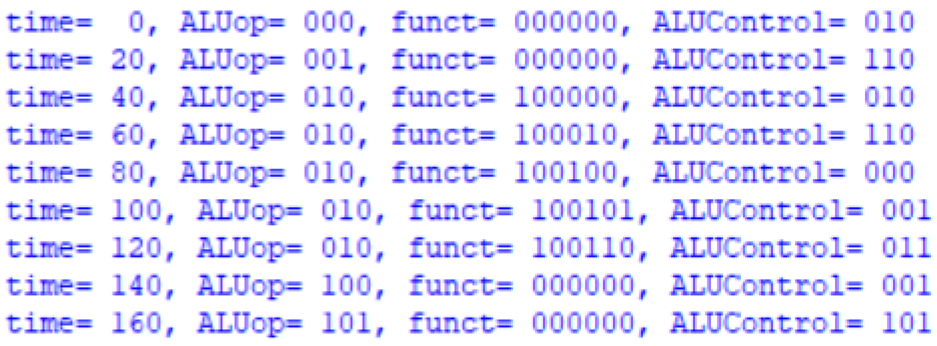
```
time=  0, A= 00000000000000000000000000001111, B= 00000000000000000000000000000001, result= 00000000000000000000000000010000
time= 20, A= 00000000000000000000000000010000, B= 00000000000000000000000000000001, result= 00000000000000000000000000010001
time= 40, A= 00000000000000000000000000010000, B= 00000000000000000000000000011111, result= 00000000000000000000000000101111
time= 60, A= 11111111111111111111111111111111, B= 00000000000000000000000000011111, result= 00000000000000000000000000011110
```

12. ALU 32bit Tesbench:



```
# time=  0, ALUop= 000, A= 11111111111111111111111111111111, B= 00000000000000000000000000000000, Result= 00000000000000000000000000000000, Zero= zero1
# time= 20, ALUop= 000, A= 00000000000000001111111111111111, B= 11111111111111110000000000000000, Result= 00000000000000000000000000000000, Zero= zero1
# time= 40, ALUop= 000, A= 00001111000011110000111100001111, B= 00010000000100000001000000010000, Result= 00000000000000000000000000000000, Zero= zero1
# time= 60, ALUop= 000, A= 00110011001100110011001100110011, B= 00010001000100010001000100010001, Result= 00010001000100010001000100010001, Zero= zero0
# time= 80, ALUop= 001, A= 11111111111111111111111111111111, B= 00000000000000000000000000000000, Result= 11111111111111111111111111111111, Zero= zero0
# time= 100, ALUop= 001, A= 00000000000000001111111111111111, B= 11111111111111110000000000000000, Result= 11111111111111111111111111111111, Zero= zero0
# time= 120, ALUop= 001, A= 00001111000011110000111100001111, B= 00010000000100000001000000010000, Result= 00011111000111110001111100011111, Zero= zero0
# time= 140, ALUop= 001, A= 00110011001100110011001100110011, B= 00010001000100010001000100010001, Result= 00110011001100110011001100110011, Zero= zero0
# time= 160, ALUop= 010, A= 11111111111111111111111111111111, B= 00000000000000000000000000000000, Result= 11111111111111111111111111111111, Zero= zero0
# time= 180, ALUop= 010, A= 00000000000000001111111111111111, B= 11111111111111110000000000000000, Result= 11111111111111111111111111111111, Zero= zero0
# time= 200, ALUop= 010, A= 00001111000011110000111100001111, B= 00010000000100000001000000010000, Result= 00011111000111110001111100011111, Zero= zero0
# time= 220, ALUop= 010, A= 00110011001100110011001100110011, B= 00010001000100010001000100010001, Result= 01000100010001000100010001000100, Zero= zero0
# time= 240, ALUop= 110, A= 11111111111111111111111111111111, B= 00000000000000000000000000000000, Result= 11111111111111111111111111111111, Zero= zero0
# time= 260, ALUop= 110, A= 00000000000000001111111111111111, B= 11111111111111110000000000000000, Result= 00000000000000001111111111111111, Zero= zero0
# time= 280, ALUop= 110, A= 00001111000011110000111100001111, B= 00010000000100000001000000010000, Result= 11111101111111011111110111111111, Zero= zero0
# time= 300, ALUop= 110, A= 00110011001100110011001100110011, B= 00010001000100010001000100010001, Result= 00100010001000100010001000100010, Zero= zero0
# time= 320, ALUop= 011, A= 11111111111111111111111111111111, B= 00000000000000000000000000000000, Result= 11111111111111111111111111111111, Zero= zero0
# time= 340, ALUop= 011, A= 00000000000000001111111111111111, B= 11111111111111110000000000000000, Result= 11111111111111111111111111111111, Zero= zero0
# time= 360, ALUop= 011, A= 00001111000011110000111100001111, B= 00010000000100000001000000010000, Result= 00011111000111110001111100011111, Zero= zero0
# time= 380, ALUop= 011, A= 00110011001100110011001100110011, B= 00010001000100010001000100010001, Result= 00100010001000100010001000100010, Zero= zero0
# time= 400, ALUop= 101, A= 11111111111111111111111111111111, B= 00000000000000000000000000000000, Result= 00000000000000000000000000000000, Zero= zero1
# time= 420, ALUop= 101, A= 00000000000000001111111111111111, B= 11111111111111110000000000000000, Result= 00000000000000000000000000000000, Zero= zero1
# time= 440, ALUop= 101, A= 00001111000011110000111100001111, B= 00010000000100000001000000010000, Result= 00010000000100000000000000000000, Zero= zero0
# time= 460, ALUop= 101, A= 00110011001100110011001100110011, B= 00010001000100010001000100010001, Result= 00010001000100010000000000000000, Zero= zero0
# time= 480, ALUop= 101, A= 11111111111111111111111111111111, B= 00000010000000100000001000000010, Result= 00000010000000100000000000000000, Zero= zero0
# time= 500, ALUop= 101, A= 00000000000000000000000000010011, B= 00000000000000000000000011111111, Result= 00000000111111110000000000000000, Zero= zero0
# time= 520, ALUop= 110, A= 00000000000000000000000011111100, B= 00000000000000000000000011101111, Result= 00000000000000000000000000001101, Zero= zero0
# time= 540, ALUop= 010, A= 00000000000000000000000011111100, B= 00000000000000000000000011101111, Result= 00000000000000000000000111101011, Zero= zero0
```

13. ALU Control



```
time=  0, ALUop= 000, funct= 000000, ALUControl= 010
time= 20, ALUop= 001, funct= 000000, ALUControl= 110
time= 40, ALUop= 010, funct= 100000, ALUControl= 010
time= 60, ALUop= 010, funct= 100010, ALUControl= 110
time= 80, ALUop= 010, funct= 100100, ALUControl= 000
time= 100, ALUop= 010, funct= 100101, ALUControl= 001
time= 120, ALUop= 010, funct= 100110, ALUControl= 011
time= 140, ALUop= 100, funct= 000000, ALUControl= 001
time= 160, ALUop= 101, funct= 000000, ALUControl= 101
```

## ModelSim Simulation:

### Register Input

```
00000000000000000000000000000000
00000000011110000000000000001101
00000000000000000000000011101111
11000000000111100000000000000011
00000000000000000000000101000100
00000000000000000000000000000101
00000000000000000000000000000110
00000000000000000000000000000111
00000000000000000000000000001111
00000000000000000000000011110000
00000000000000000000111100000000
00000000000000000000100000000000
00000000000000000000000000100000
00000000000000000000000000001001
00000000000000000000000000001100
00000000000000000000000001100000
00000000000000000000000000010000
00000000000000000000000000010001
00000000000000000000000000010010
00000000000000000000000000010011
00000000000000000000000000010100
00000000000000000000000000010101
00000000000000000000000000010110
00000000000000000000000000010111
00000000000000000000000000011000
00000000000000000000000000011001
00000000000000000000000000011010
00000000000000000000000000011011
00000000000000000000000000011100
00000000000000000000000000011101
00000000000000000000000000011110
00000000000000000000000000011111
```

### Data Memory

```
00000000000000000000000000000000
11110000111100001111000011110000
00000000000000000000000000000000
00000000000000000000000000000000
01010101010101010101001010101010
00000000000000000000000000000000
00000000000000000000000000000110
00000000000000000000000000000111
00000000000000000000000000001000
00000000000000000000000000001001
00000000000000000000000000001010
00000000000000000000000000001011
00000000000000000000000000001100
00000000000000000000000000001101
00000000000000000000000000001110
00000000000000000000000000001111
00000000000000000000000000010000
00000000000000000000000000010001
00000000000000000000000000010010
00000000000000000000000000010011
00000000000000000000000000010100
00000000000000000000000000010101
00000000000000000000000000010110
00000000000000000000000000010111
```

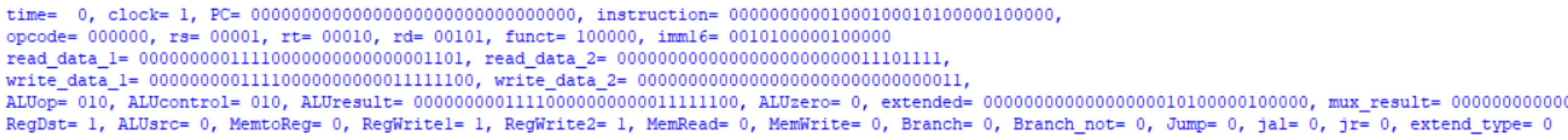
171044032 CSE 331 – 2020 FALL Assignment 4
Türker Tercan

## Instructions:

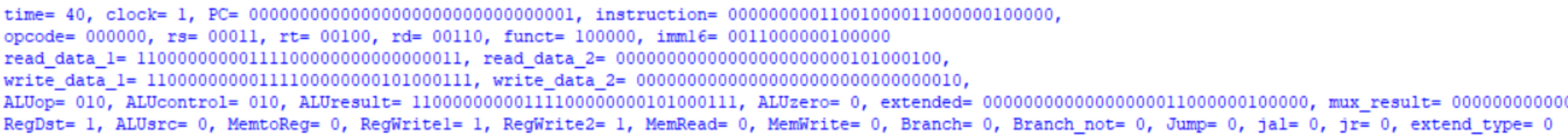
```
00000000001000100010100000100000        //000000        addn
00000000011001000011000000100000        //000001        addn
00000000001000100011100000100010        //000010        subn
00000000011001000100000000100010        //000011        subn
00000000001000100100100000100100        //000100        andn
00000000011001000101000000100100        //000101        andn
00000000001000100101100000100101        //000110        orn
00000000011001000110000000100101        //000111        orn
00000000001000100110100000100110        //001000        xor
00000000101001000111000000100110        //001001        xor
10001100000011110000000000000100        //001010        lw
10001100000010000000000000000001        //001011        lw
10101100000011110000000000000101        //001100        sw
10101100000010000000000000000010        //001101        sw
00110100000100010000111100001111        //001110        ori
00110100000100100000111111110000        //001111        ori
00111110010100111111000011110000        //010000        lui
00111110011101000000000011111111        //010001        lui
00010010101101000000000000000001        //010010        beq
00010000000000010000000000000001        //010011        beq
00110100000100000000111100001111        //010100
00110100000100010000111111110000        //010101
00111100000100101111000011110000        //010110
00111100000100110000000011111111        //010111
00010100000000010000000000000001        //011000        bne
00010110101100110000000000000001        //011001        bne
00110100000100000000111100001111        //011010
00110100000100010000111111110000        //011011
00111100000100101111000011110000        //011100
00111100000100110001000000000001        //011101
00001000000000000000000000001000        //011110        j
00000000101001000111000000100110        //011111
00001000000000000000000000001001        //100000        j
00000000101001000111000000100110        //100001
00000000101001000111000000100110        //100010
00000000001000100111000000100110        //100011
00001100000000000000000000001010        //100100        jal
00001100000000000000000000001011        //100101        jal
00001000000000000000000000001100        //100110        j and program ends
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx        //100111
00000111111000000000000000000000        //101000        jr
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx        //101001
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx        //101010
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx        //101011
00000111111000000000000000000000        //101100        jr
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

171044032 CSE 331 – 2020 FALL Assignment 4
Türker Tercan

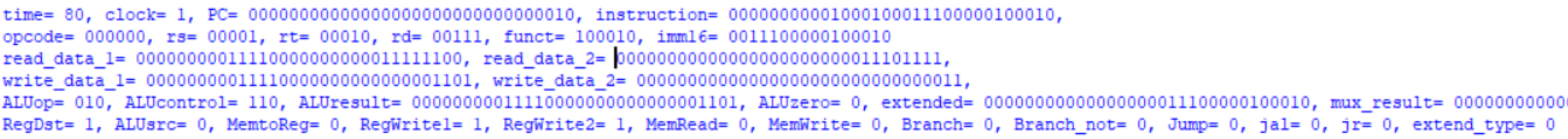
addn Test 1: Passed


```
time=  0, clock= 1, PC= 00000000000000000000000000000000, instruction= 00000000001000100010100000100000,
opcode= 000000, rs= 00001, rt= 00010, rd= 00101, funct= 100000, imm16= 0010100000100000
read_data_1= 00000000111100000000000000001101, read_data_2= 00000000000000000000000011101111,
write_data_1= 00000000111100000000000011111100, write_data_2= 00000000000000000000000000000011,
ALUop= 010, ALUcontrol= 010, ALUresult= 00000000111100000000000011111100, ALUzero= 0, extended= 00000000000000000010100000100000, mux_result= 000000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite1= 1, RegWrite2= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

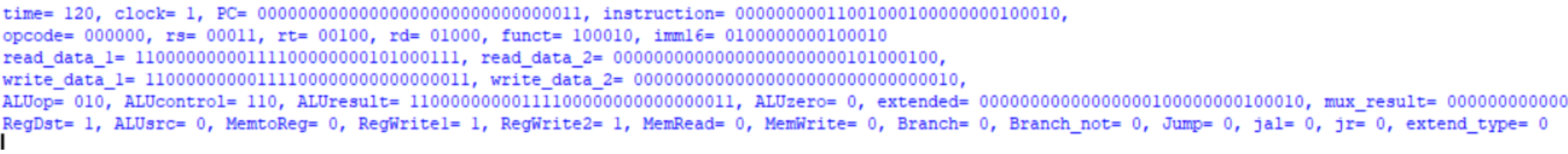
addn Test 2: Passed


```
time= 40, clock= 1, PC= 00000000000000000000000000000001, instruction= 00000000011001000011000000100000,
opcode= 000000, rs= 00011, rt= 00100, rd= 00110, funct= 100000, imm16= 0011000000100000
read_data_1= 11000000000111100000000000000011, read_data_2= 00000000000000000000000101000100,
write_data_1= 11000000000111100000000101000111, write_data_2= 00000000000000000000000000000010,
ALUop= 010, ALUcontrol= 010, ALUresult= 11000000000111100000000101000111, ALUzero= 0, extended= 00000000000000000011000000100000, mux_result= 000000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite1= 1, RegWrite2= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

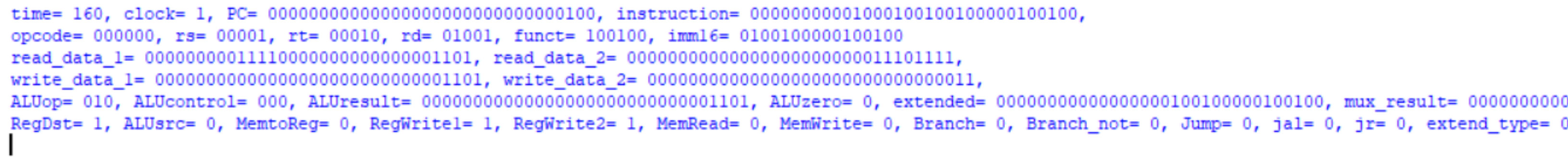
subn Test 1: Passed


```
time= 80, clock= 1, PC= 00000000000000000000000000000010, instruction= 00000000001000100011100000100010,
opcode= 000000, rs= 00001, rt= 00010, rd= 00111, funct= 100010, imm16= 0011100000100010
read_data_1= 00000000111100000000000011111100, read_data_2= 00000000000000000000000011101111,
write_data_1= 00000000111100000000000000001101, write_data_2= 00000000000000000000000000000011,
ALUop= 010, ALUcontrol= 110, ALUresult= 00000000111100000000000000001101, ALUzero= 0, extended= 00000000000000000011100000100010, mux_result= 000000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite1= 1, RegWrite2= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

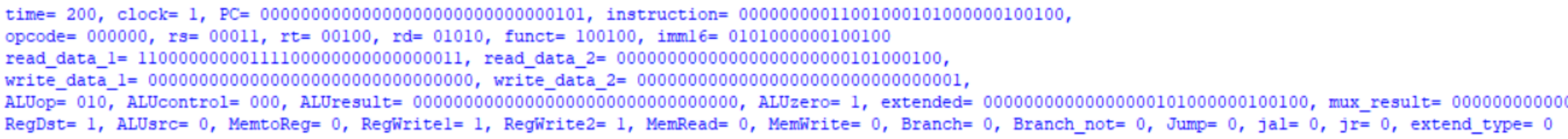
subn Test 2: Passed


```
time= 120, clock= 1, PC= 00000000000000000000000000000011, instruction= 00000000011001000100000000100010,
opcode= 000000, rs= 00011, rt= 00100, rd= 01000, funct= 100010, imm16= 0100000000100010
read_data_1= 11000000000111100000000101000111, read_data_2= 00000000000000000000000101000100,
write_data_1= 11000000000111100000000000000011, write_data_2= 00000000000000000000000000000010,
ALUop= 010, ALUcontrol= 110, ALUresult= 11000000000111100000000000000011, ALUzero= 0, extended= 00000000000000000100000000100010, mux_result= 000000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite1= 1, RegWrite2= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

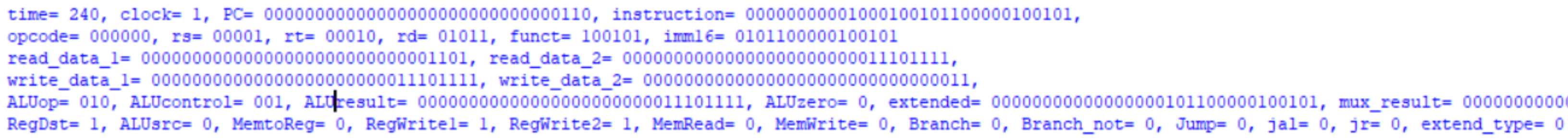
andn Test 1: Passed


```
time= 160, clock= 1, PC= 00000000000000000000000000000100, instruction= 00000000001000100100100000100100,
opcode= 000000, rs= 00001, rt= 00010, rd= 01001, funct= 100100, imm16= 0100100000100100
read_data_1= 00000000111100000000000000001101, read_data_2= 00000000000000000000000011101111,
write_data_1= 00000000000000000000000000001101, write_data_2= 00000000000000000000000000000011,
ALUop= 010, ALUcontrol= 000, ALUresult= 00000000000000000000000000001101, ALUzero= 0, extended= 00000000000000000100100000100100, mux_result= 0000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite1= 1, RegWrite2= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

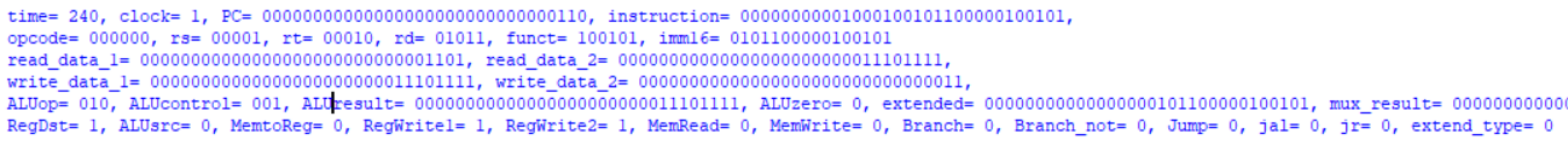
andn Test 2: Passed


```
time= 200, clock= 1, PC= 00000000000000000000000000000101, instruction= 00000000011001000101000000100100,
opcode= 000000, rs= 00011, rt= 00100, rd= 01010, funct= 100100, imm16= 0101000000100100
read_data_1= 11000000000111100000000000000011, read_data_2= 00000000000000000000000101000100,
write_data_1= 00000000000000000000000000000000, write_data_2= 00000000000000000000000000000001,
ALUop= 010, ALUcontrol= 000, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000101000000100100, mux_result= 000000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite1= 1, RegWrite2= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

orn Test 1: Passed

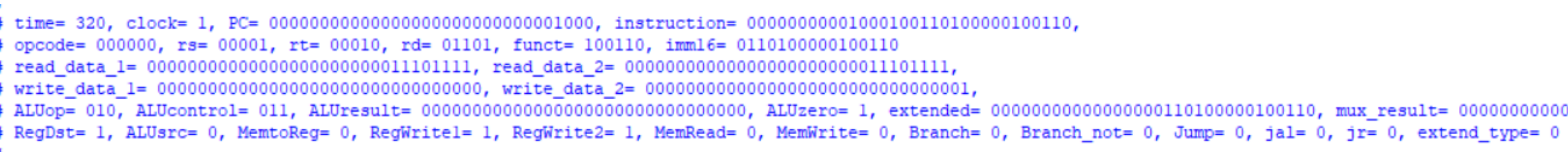

```
time= 240, clock= 1, PC= 00000000000000000000000000000110, instruction= 00000000001000100101100000100101,
opcode= 000000, rs= 00001, rt= 00010, rd= 01011, funct= 100101, imm16= 0101100000100101
read_data_1= 00000000000000000000000000001101, read_data_2= 00000000000000000000000011101111,
write_data_1= 00000000000000000000000011101111, write_data_2= 00000000000000000000000000000011,
ALUop= 010, ALUcontrol= 001, ALUresult= 00000000000000000000000011101111, ALUzero= 0, extended= 00000000000000000101100000100101, mux_result= 0000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite1= 1, RegWrite2= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

orn Test 2: Passed



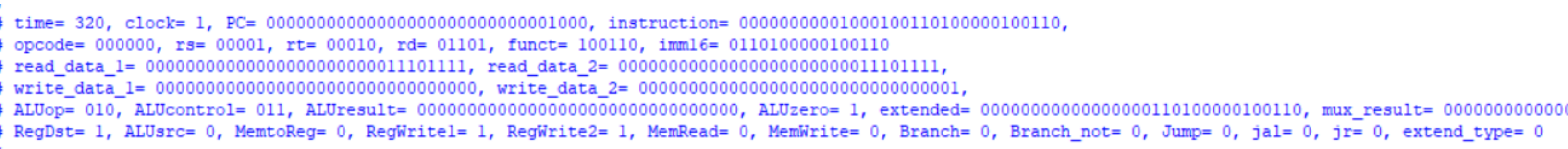
```
time= 240, clock= 1, PC= 00000000000000000000000000000110, instruction= 00000000001000100101100000100101,
opcode= 000000, rs= 00001, rt= 00010, rd= 01011, funct= 100101, imm16= 0101100000100101
read_data_1= 00000000000000000000000000001101, read_data_2= 00000000000000000000000011101111,
write_data_1= 00000000000000000000000011101111, write_data_2= 00000000000000000000000000000011,
ALUop= 010, ALUcontrol= 001, ALUresult= 00000000000000000000000011101111, ALUzero= 0, extended= 00000000000000000101100000100101, mux_result= 00000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite1= 1, RegWrite2= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

xorn Test 1: Passed



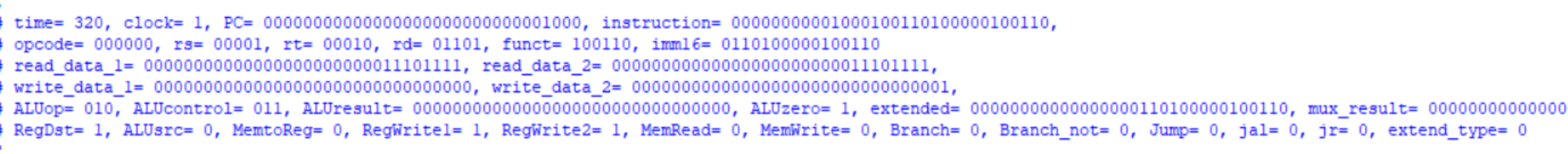
```
time= 320, clock= 1, PC= 00000000000000000000000000001000, instruction= 00000000001000100110100000100110,
opcode= 000000, rs= 00001, rt= 00010, rd= 01101, funct= 100110, imm16= 0110100000100110
read_data_1= 00000000000000000000000011101111, read_data_2= 00000000000000000000000011101111,
write_data_1= 00000000000000000000000000000000, write_data_2= 00000000000000000000000000000001,
ALUop= 010, ALUcontrol= 011, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000110100000100110, mux_result= 00000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite1= 1, RegWrite2= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

xorn Test 2: Passed



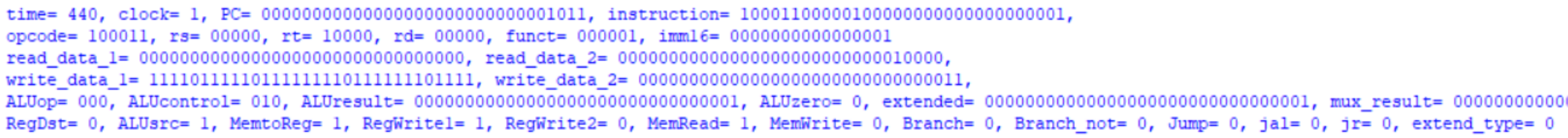
```
time= 320, clock= 1, PC= 00000000000000000000000000001000, instruction= 00000000001000100110100000100110,
opcode= 000000, rs= 00001, rt= 00010, rd= 01101, funct= 100110, imm16= 0110100000100110
read_data_1= 00000000000000000000000011101111, read_data_2= 00000000000000000000000011101111,
write_data_1= 00000000000000000000000000000000, write_data_2= 00000000000000000000000000000001,
ALUop= 010, ALUcontrol= 011, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000110100000100110, mux_result= 00000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite1= 1, RegWrite2= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

lw Test 1: Passed



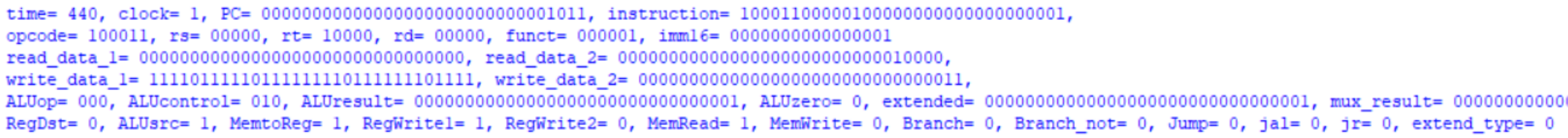
```
time= 320, clock= 1, PC= 00000000000000000000000000001000, instruction= 00000000001000100110100000100110,
opcode= 000000, rs= 00001, rt= 00010, rd= 01101, funct= 100110, imm16= 0110100000100110
read_data_1= 00000000000000000000000011101111, read_data_2= 00000000000000000000000011101111,
write_data_1= 00000000000000000000000000000000, write_data_2= 00000000000000000000000000000001,
ALUop= 010, ALUcontrol= 011, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000110100000100110, mux_result= 0000000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite1= 1, RegWrite2= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

lw Test 2: Passed



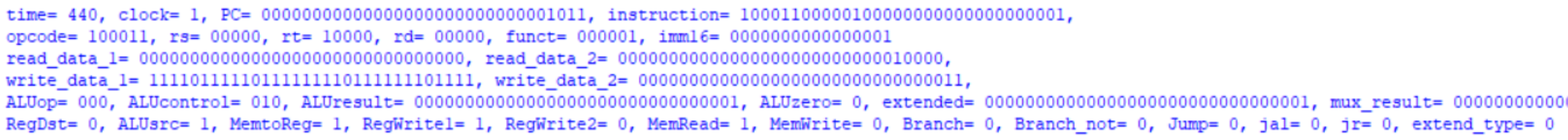
```
time= 440, clock= 1, PC= 00000000000000000000000000001011, instruction= 10001100000100000000000000000001,
opcode= 100011, rs= 00000, rt= 10000, rd= 00000, funct= 000001, imm16= 0000000000000001
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000010000,
write_data_1= 11110111110111111110111111101111, write_data_2= 00000000000000000000000000000011,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000000001, ALUzero= 0, extended= 00000000000000000000000000000001, mux_result= 00000000000
RegDst= 0, ALUsrc= 1, MemtoReg= 1, RegWrite1= 1, RegWrite2= 0, MemRead= 1, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

sw Test 1: Passed



```
time= 440, clock= 1, PC= 00000000000000000000000000001011, instruction= 10001100000100000000000000000001,
opcode= 100011, rs= 00000, rt= 10000, rd= 00000, funct= 000001, imm16= 0000000000000001
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000010000,
write_data_1= 11110111110111111110111111101111, write_data_2= 00000000000000000000000000000011,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000000001, ALUzero= 0, extended= 00000000000000000000000000000001, mux_result= 00000000000
RegDst= 0, ALUsrc= 1, MemtoReg= 1, RegWrite1= 1, RegWrite2= 0, MemRead= 1, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

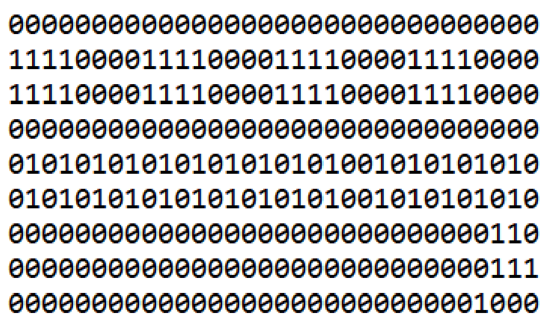
sw Test 2: Passed



```
time= 440, clock= 1, PC= 00000000000000000000000000001011, instruction= 10001100000100000000000000000001,
opcode= 100011, rs= 00000, rt= 10000, rd= 00000, funct= 000001, imm16= 0000000000000001
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000010000,
write_data_1= 11110111110111111110111111101111, write_data_2= 00000000000000000000000000000011,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000000001, ALUzero= 0, extended= 00000000000000000000000000000001, mux_result= 00000000000
RegDst= 0, ALUsrc= 1, MemtoReg= 1, RegWrite1= 1, RegWrite2= 0, MemRead= 1, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

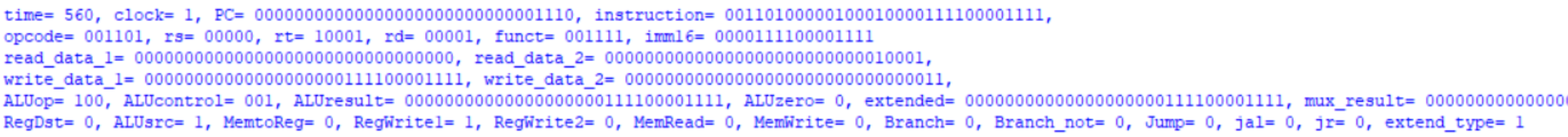
171044032 CSE 331 – 2020 FALL Assignment 4
Türker Tercan

Data Memory After lw and sw test:



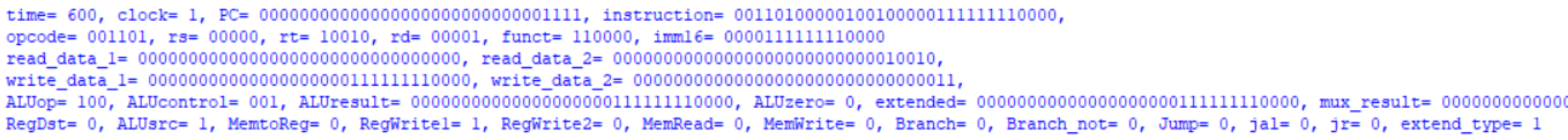
```
00000000000000000000000000000000
11110000111100001111000011110000
11110000111100001111000011110000
00000000000000000000000000000000
01010101010101010101001010101010
01010101010101010101001010101010
00000000000000000000000000000110
00000000000000000000000000000111
00000000000000000000000000001000
```

ori Test 1: Passed



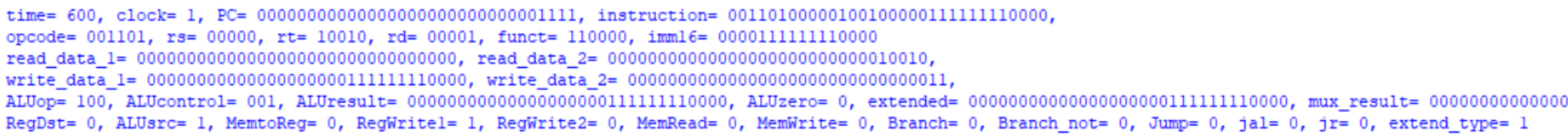
```
time= 560, clock= 1, PC= 00000000000000000000000000001110, instruction= 00110100000100010000111100001111,
opcode= 001101, rs= 00000, rt= 10001, rd= 00001, funct= 001111, imm16= 0000111100001111
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000010001,
write_data_1= 00000000000000000000111100001111, write_data_2= 00000000000000000000000000000011,
ALUop= 100, ALUcontrol= 001, ALUresult= 00000000000000000000111100001111, ALUzero= 0, extended= 00000000000000000000111100001111, mux_result= 00000000000000
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite1= 1, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 1
```

ori Test 2: Passed



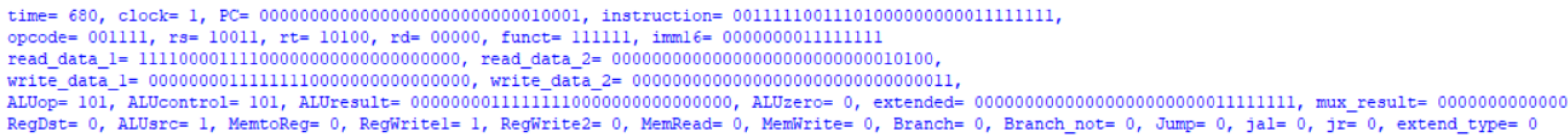
```
time= 600, clock= 1, PC= 00000000000000000000000000001111, instruction= 00110100000100100000111111110000,
opcode= 001101, rs= 00000, rt= 10010, rd= 00001, funct= 110000, imm16= 0000111111110000
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000010010,
write_data_1= 00000000000000000000111111110000, write_data_2= 00000000000000000000000000000011,
ALUop= 100, ALUcontrol= 001, ALUresult= 00000000000000000000111111110000, ALUzero= 0, extended= 00000000000000000000111111110000, mux_result= 000000000000
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite1= 1, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 1
```

lui Test 1: Passed



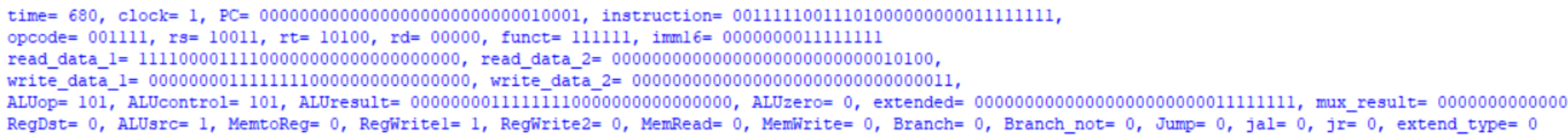
```
time= 600, clock= 1, PC= 00000000000000000000000000001111, instruction= 00110100000100100000111111110000,
opcode= 001101, rs= 00000, rt= 10010, rd= 00001, funct= 110000, imm16= 0000111111110000
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000010010,
write_data_1= 00000000000000000000111111110000, write_data_2= 00000000000000000000000000000011,
ALUop= 100, ALUcontrol= 001, ALUresult= 00000000000000000000111111110000, ALUzero= 0, extended= 00000000000000000000111111110000, mux_result= 0000000000000
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite1= 1, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 1
```

lui Test 2: Passed



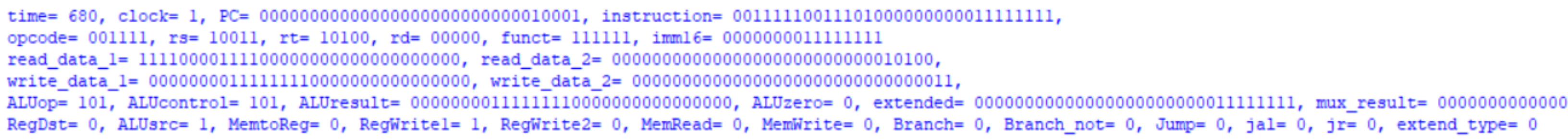
```
time= 680, clock= 1, PC= 00000000000000000000000000010001, instruction= 00111100111010000000000011111111,
opcode= 001111, rs= 10011, rt= 10100, rd= 00000, funct= 111111, imm16= 0000000011111111
read_data_1= 11110000111100000000000000000000, read_data_2= 00000000000000000000000000010100,
write_data_1= 00000000111111110000000000000000, write_data_2= 00000000000000000000000000000011,
ALUop= 101, ALUcontrol= 101, ALUresult= 00000000111111110000000000000000, ALUzero= 0, extended= 00000000000000000000000011111111, mux_result= 000000000000
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite1= 1, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

beq Test 1: Passed



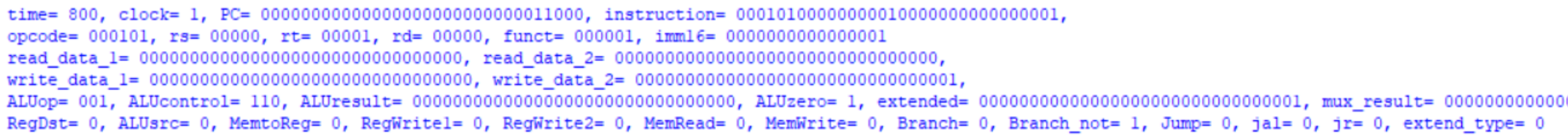
```
time= 680, clock= 1, PC= 00000000000000000000000000010001, instruction= 00111100111010000000000011111111,
opcode= 001111, rs= 10011, rt= 10100, rd= 00000, funct= 111111, imm16= 0000000011111111
read_data_1= 11110000111100000000000000000000, read_data_2= 00000000000000000000000000010100,
write_data_1= 00000000111111110000000000000000, write_data_2= 00000000000000000000000000000011,
ALUop= 101, ALUcontrol= 101, ALUresult= 00000000111111110000000000000000, ALUzero= 0, extended= 00000000000000000000000011111111, mux_result= 000000000000
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite1= 1, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

beq Test 2: Passed



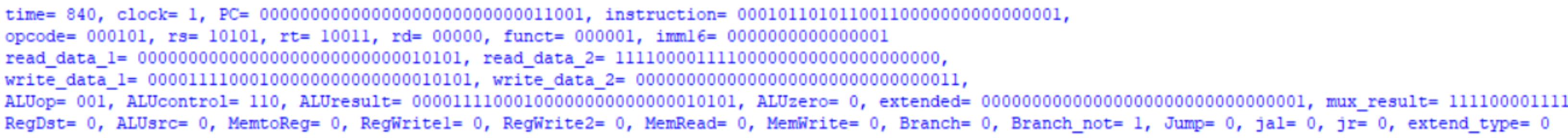
```
time= 680, clock= 1, PC= 00000000000000000000000000010001, instruction= 00111110011101000000000011111111,
opcode= 001111, rs= 10011, rt= 10100, rd= 00000, funct= 111111, imm16= 0000000011111111
read_data_1= 11110000111100000000000000000000, read_data_2= 00000000000000000000000000010100,
write_data_1= 00000000111111110000000000000000, write_data_2= 00000000000000000000000000000011,
ALUop= 101, ALUcontrol= 101, ALUresult= 00000000111111110000000000000000, ALUzero= 0, extended= 00000000000000000000000011111111, mux_result= 0000000000000
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite1= 1, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

bne Test 1: Passed

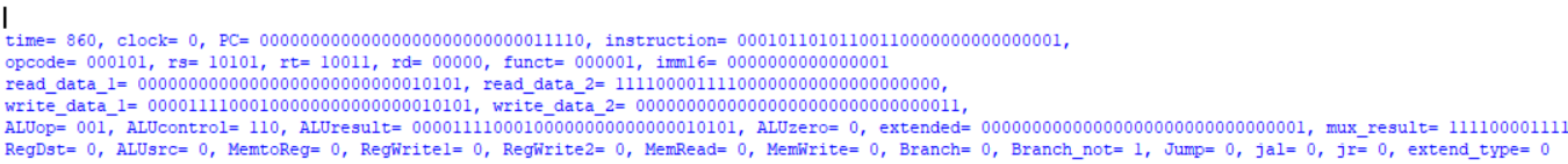


```
time= 800, clock= 1, PC= 00000000000000000000000000011000, instruction= 00010100000000010000000000000001,
opcode= 000101, rs= 00000, rt= 00001, rd= 00000, funct= 000001, imm16= 0000000000000001
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000000,
write_data_1= 00000000000000000000000000000000, write_data_2= 00000000000000000000000000000001,
ALUop= 001, ALUcontrol= 110, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000000001, mux_result= 000000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 0, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 1, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

bne Test 2: Passed – PC <= PC + 4 + imm16

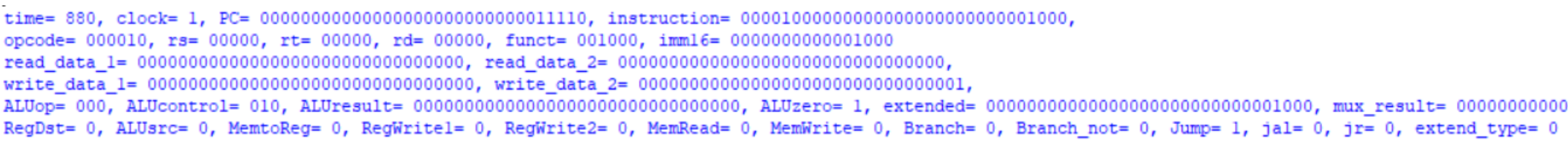


```
time= 840, clock= 1, PC= 00000000000000000000000000011001, instruction= 00010110101100110000000000000001,
opcode= 000101, rs= 10101, rt= 10011, rd= 00000, funct= 000001, imm16= 0000000000000001
read_data_1= 00000000000000000000000000010101, read_data_2= 11110000111100000000000000000000,
write_data_1= 00001111000100000000000000010101, write_data_2= 00000000000000000000000000000011,
ALUop= 001, ALUcontrol= 110, ALUresult= 00001111000100000000000000010101, ALUzero= 0, extended= 00000000000000000000000000000001, mux_result= 111100001111
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 0, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 1, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

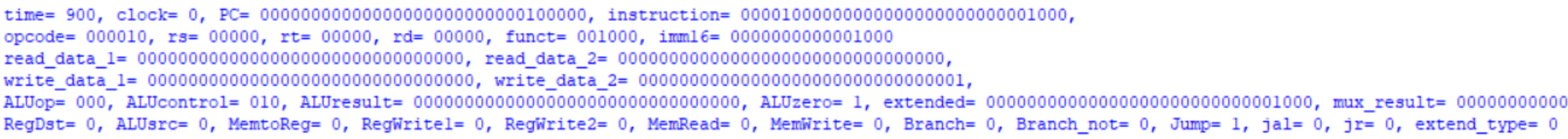


```
time= 860, clock= 0, PC= 00000000000000000000000000011110, instruction= 00010110101100110000000000000001,
opcode= 000101, rs= 10101, rt= 10011, rd= 00000, funct= 000001, imm16= 0000000000000001
read_data_1= 00000000000000000000000000010101, read_data_2= 11110000111100000000000000000000,
write_data_1= 00001111000100000000000000010101, write_data_2= 00000000000000000000000000000011,
ALUop= 001, ALUcontrol= 110, ALUresult= 00001111000100000000000000010101, ALUzero= 0, extended= 00000000000000000000000000000001, mux_result= 111100001111
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 0, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 1, Jump= 0, jal= 0, jr= 0, extend_type= 0
```

j Test 1: Passed – you can see the PC changes

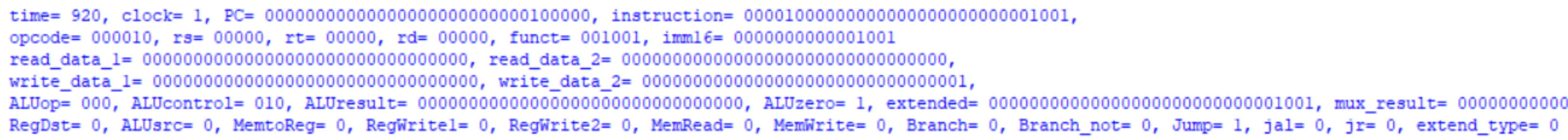


```
time= 880, clock= 1, PC= 00000000000000000000000000011110, instruction= 00001000000000000000000000001000,
opcode= 000010, rs= 00000, rt= 00000, rd= 00000, funct= 001000, imm16= 0000000000001000
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000000,
write_data_1= 00000000000000000000000000000000, write_data_2= 00000000000000000000000000000001,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000001000, mux_result= 00000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 0, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 1, jal= 0, jr= 0, extend_type= 0
```

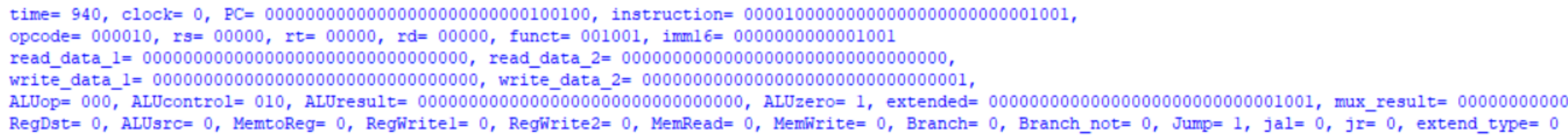


```
time= 900, clock= 0, PC= 00000000000000000000000000100000, instruction= 00001000000000000000000000001000,
opcode= 000010, rs= 00000, rt= 00000, rd= 00000, funct= 001000, imm16= 0000000000001000
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000000,
write_data_1= 00000000000000000000000000000000, write_data_2= 00000000000000000000000000000001,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000001000, mux_result= 00000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 0, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 1, jal= 0, jr= 0, extend_type= 0
```

j Test 2: Passed



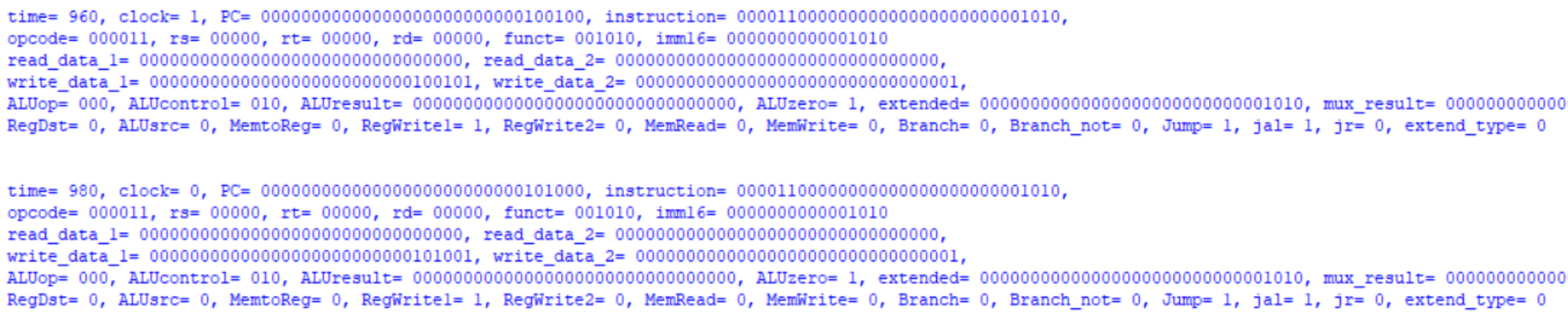
```
time= 920, clock= 1, PC= 00000000000000000000000000100000, instruction= 00001000000000000000000000001001,
opcode= 000010, rs= 00000, rt= 00000, rd= 00000, funct= 001001, imm16= 0000000000001001
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000000,
write_data_1= 00000000000000000000000000000000, write_data_2= 00000000000000000000000000000001,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000001001, mux_result= 00000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 0, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 1, jal= 0, jr= 0, extend_type= 0
```



```
time= 940, clock= 0, PC= 00000000000000000000000000100100, instruction= 00001000000000000000000000001001,
opcode= 000010, rs= 00000, rt= 00000, rd= 00000, funct= 001001, imm16= 0000000000001001
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000000,
write_data_1= 00000000000000000000000000000000, write_data_2= 00000000000000000000000000000001,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000001001, mux_result= 00000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 0, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 1, jal= 0, jr= 0, extend_type= 0
```

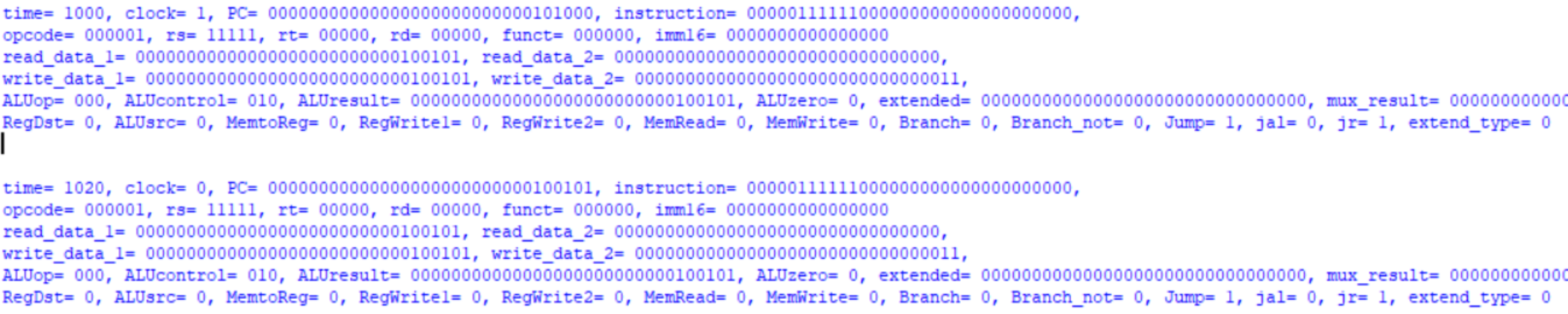
171044032 CSE 331 – 2020 FALL Assignment 4
Türker Tercan

jal Test 1: Passed



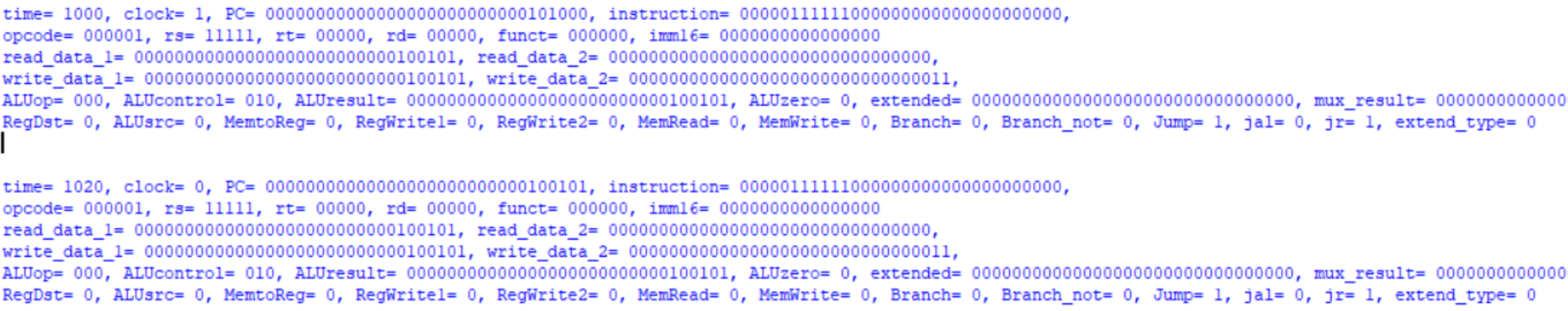
```
time= 960, clock= 1, PC= 00000000000000000000000000100100, instruction= 00001100000000000000000000001010,
opcode= 000011, rs= 00000, rt= 00000, rd= 00000, funct= 001010, imm16= 0000000000001010
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000000,
write_data_1= 00000000000000000000000000100101, write_data_2= 00000000000000000000000000000001,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000001010, mux_result= 000000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 1, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 1, jal= 1, jr= 0, extend_type= 0

time= 980, clock= 0, PC= 00000000000000000000000000101000, instruction= 00001100000000000000000000001010,
opcode= 000011, rs= 00000, rt= 00000, rd= 00000, funct= 001010, imm16= 0000000000001010
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000000,
write_data_1= 00000000000000000000000000101001, write_data_2= 00000000000000000000000000000001,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000001010, mux_result= 000000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 1, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 1, jal= 1, jr= 0, extend_type= 0
```

jr Test 1: Passed



```
time= 1000, clock= 1, PC= 00000000000000000000000000101000, instruction= 00000111111000000000000000000000,
opcode= 000001, rs= 11111, rt= 00000, rd= 00000, funct= 000000, imm16= 0000000000000000
read_data_1= 00000000000000000000000000100101, read_data_2= 00000000000000000000000000000000,
write_data_1= 00000000000000000000000000100101, write_data_2= 00000000000000000000000000000011,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000100101, ALUzero= 0, extended= 00000000000000000000000000000000, mux_result= 000000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 0, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 1, jal= 0, jr= 1, extend_type= 0

time= 1020, clock= 0, PC= 00000000000000000000000000100101, instruction= 00000111111000000000000000000000,
opcode= 000001, rs= 11111, rt= 00000, rd= 00000, funct= 000000, imm16= 0000000000000000
read_data_1= 00000000000000000000000000100101, read_data_2= 00000000000000000000000000000000,
write_data_1= 00000000000000000000000000100101, write_data_2= 00000000000000000000000000000011,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000100101, ALUzero= 0, extended= 00000000000000000000000000000000, mux_result= 000000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 0, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 1, jal= 0, jr= 1, extend_type= 0
```

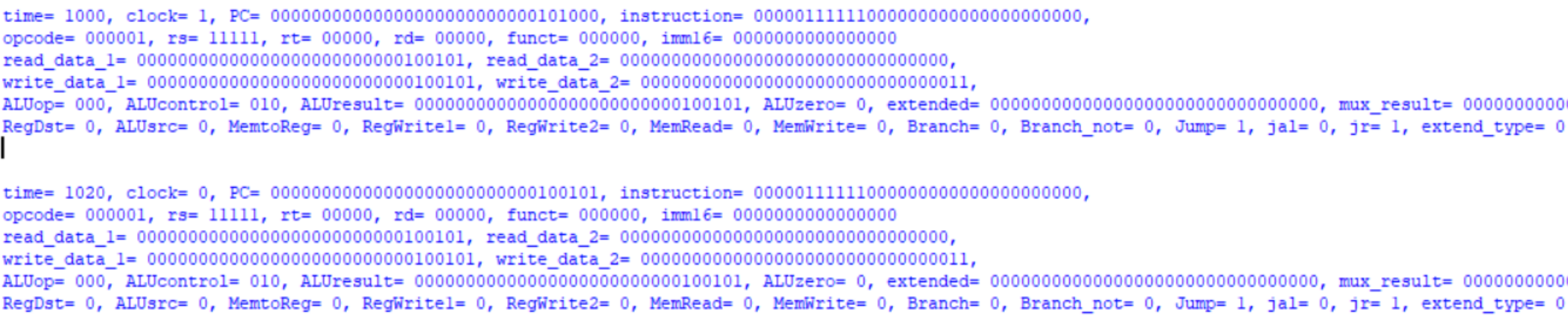
jal Test 2: Passed



```
time= 1000, clock= 1, PC= 00000000000000000000000000101000, instruction= 00000111111000000000000000000000,
opcode= 000001, rs= 11111, rt= 00000, rd= 00000, funct= 000000, imm16= 0000000000000000
read_data_1= 00000000000000000000000000100101, read_data_2= 00000000000000000000000000000000,
write_data_1= 00000000000000000000000000100101, write_data_2= 00000000000000000000000000000011,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000100101, ALUzero= 0, extended= 00000000000000000000000000000000, mux_result= 000000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 0, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 1, jal= 0, jr= 1, extend_type= 0

time= 1020, clock= 0, PC= 00000000000000000000000000100101, instruction= 00000111111000000000000000000000,
opcode= 000001, rs= 11111, rt= 00000, rd= 00000, funct= 000000, imm16= 0000000000000000
read_data_1= 00000000000000000000000000100101, read_data_2= 00000000000000000000000000000000,
write_data_1= 00000000000000000000000000100101, write_data_2= 00000000000000000000000000000011,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000100101, ALUzero= 0, extended= 00000000000000000000000000000000, mux_result= 000000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 0, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 1, jal= 0, jr= 1, extend_type= 0
```

Türker Tercan

jr Test 2: Passed



```
time= 1000, clock= 1, PC= 00000000000000000000000000101000, instruction= 00000111111000000000000000000000,
opcode= 000001, rs= 11111, rt= 00000, rd= 00000, funct= 000000, imm16= 0000000000000000
read_data_1= 00000000000000000000000000100101, read_data_2= 00000000000000000000000000000000,
write_data_1= 00000000000000000000000000100101, write_data_2= 00000000000000000000000000000011,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000100101, ALUzero= 0, extended= 00000000000000000000000000000000, mux_result= 0000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 0, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 1, jal= 0, jr= 1, extend_type= 0

time= 1020, clock= 0, PC= 00000000000000000000000000100101, instruction= 00000111111000000000000000000000,
opcode= 000001, rs= 11111, rt= 00000, rd= 00000, funct= 000000, imm16= 0000000000000000
read_data_1= 00000000000000000000000000100101, read_data_2= 00000000000000000000000000000000,
write_data_1= 00000000000000000000000000100101, write_data_2= 00000000000000000000000000000011,
ALUop= 000, ALUcontrol= 010, ALUresult= 00000000000000000000000000100101, ALUzero= 0, extended= 00000000000000000000000000000000, mux_result= 0000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite1= 0, RegWrite2= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, Jump= 1, jal= 0, jr= 1, extend_type= 0
```