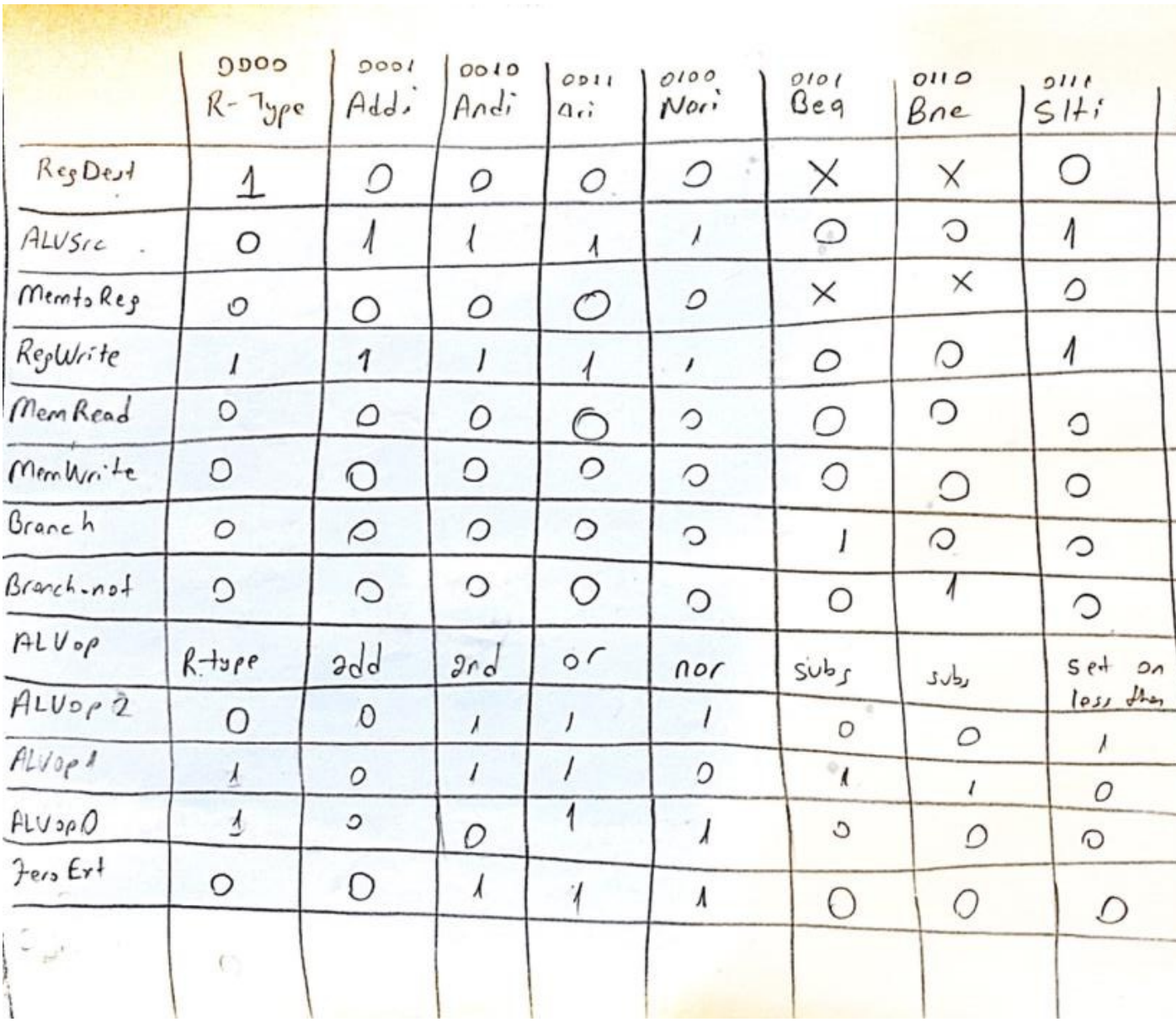
CSE 331/503
Computer Organization
Final Project – MiniMIPS Design
REPORT

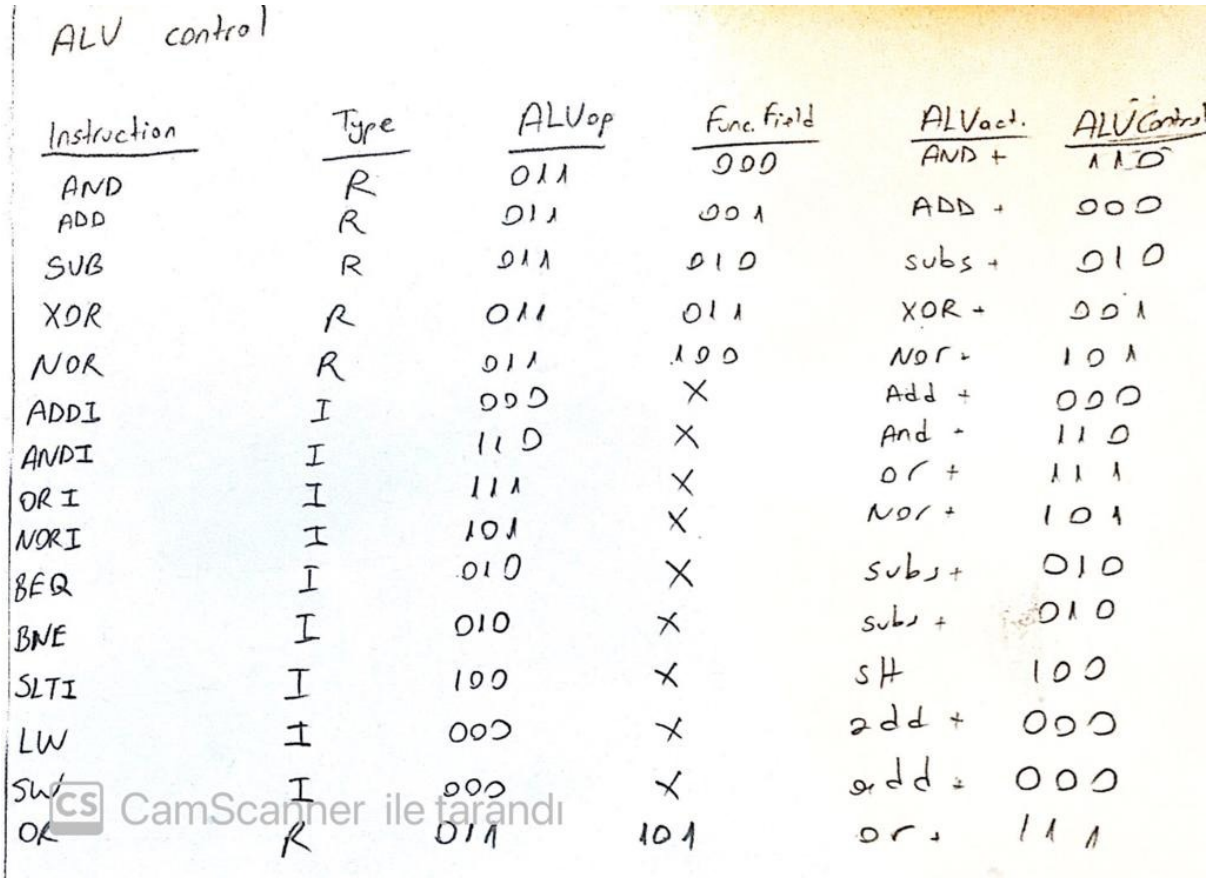
Burak Çiçek
1901042260

MAIN CONTROL TRUTH TABLE



	0000 R-Type	0001 Addi	0010 Andi	0011 Ori	0100 Nori	0101 Beq	0110 Bne	0111 Slti
RegDest	1	0	0	0	0	X	X	0
ALUSrc	0	1	1	1	1	0	0	1
MemtoReg	0	0	0	0	0	X	X	0
RegWrite	1	1	1	1	1	0	0	1
MemRead	0	0	0	0	0	0	0	0
MemWrite	0	0	0	0	0	0	0	0
Branch	0	0	0	0	0	1	0	0
Branch_not	0	0	0	0	0	0	1	0
ALUop	R-type	add	and	or	nor	subs	subs	set on less than
ALUop 2	0	0	1	1	1	0	0	1
ALUop 1	1	0	1	1	0	1	1	0
ALUop 0	1	0	0	1	1	0	0	0
Zero Ext	0	0	1	1	1	0	0	0

ALU CONTROL TRUTH TABLE



ALU control

Instruction	Type	ALUop	Func. Field	ALUoct.	ALUControl
AND	R	011	000	AND →	110
ADD	R	011	001	ADD →	000
SUB	R	011	010	subs →	010
XOR	R	011	011	XOR →	001
NOR	R	011	100	Nor →	101
ADDI	I	000	X	Add →	000
ANDI	I	110	X	And →	110
ORI	I	111	X	or →	111
NORI	I	101	X	Nor →	101
BEQ	I	010	X	subs →	010
BNE	I	010	X	subs →	010
SLTI	I	100	X	sH	100
LW	I	000	X	add →	000
SW	I	000	X	add →	000
OR	R	011	101	or →	111

-My MINIMips design is support I and R Type Instructions which is 16 bit.
-Also $0 is always zero, it is not changable.
-Registers are 32 Bit.
-Instruction memory can support 41 instructions for this assignment but it can increment easily.
-There is 8 Registers in this design.

Testbenches:

zero_extend6to32:

```
# time=  0, imm6= 100000, result= 00000000000000000000000000100000
# time= 20, imm6= 010101, result= 00000000000000000000000000010101
# time= 40, imm6= 101010, result= 00000000000000000000000000101010
```

xor32bit:

```
# time =  0, a =01111011111111111111111111111110, b=01111111111111111100001111111111, , out=00000100000000000011110000000001
# time = 20, a =11100000000000000000000000000000, b=00000000000000000000000000000111, , out=11100000000000000000000000000111
```

subt32bit:

```
# time =  0, a =00000000000000000000000000000111, b=00000000000000000000000000000110, , out=00000000000000000000000000000001
# time = 20, a =00000000000000000000000000000111, b=00000000000000000000000001110000, , out=11111111111111111111111110010111
```

slt32bit:

```
# time =  0, a =01111011111111111111111111111110, b=01111111111111111100001111111111, , out=00000000000000000000000000000001
# time = 20, a =01100000000000000000000000000000, b=00000000000000000000000000000111, , out=00000000000000000000000000000000
```

sign_extend6to32:

```
# time=  0, imm16= 101010, result= 11111111111111111111111111101010
# time= 20, imm16= 010101, result= 00000000000000000000000000010101
# time= 40, imm16= 110010, result= 11111111111111111111111111110010
```

shiftr_64bit:

```
# time =  0, a =0000000000000000000000000000001000000000000000000000000000000000, out=0000000000000000000000000000001000000000000000000000000000000000
# time = 20, a =0000011100000000000000000000001000000000000000000000000000000000, out=0000011100000000000000000000001000000000000000000000000000000000
```

shiftr_32bit:

```
# time =  0, a =00000000000000000000000000000110, out=00000000000000000000000000000011
# time = 20, a =10000000000000000000000000010110, out=01000000000000000000000000001011
```

shift_left32bit_2:

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

or_32bit:

```
# time =  0, a =01111011111111111111111111111110, b=01111111111111111100001111111111, , out=01111111111111111111111111111111
# time = 20, a =11100000000000000000000000000000, b=00000000000000000000000000000111, , out=11100000000000000000000000000111
```

not_32bit:

```
# time =  0, a =11111111111111111111111111111111, out=00000000000000000000000000000000
# time = 20, a =00000000000000000000000000000000, out=11111111111111111111111111111111
```

nor_32bit:

```
# time =  0, a =01111011111111111111111111111110, b=01111111111111111100001111111111, , out=10000000000000000000000000000000
# time = 20, a =11100000000000000000000000000000, b=00000000000000000000000000000111, , out=00011111111111111111111111111000
```

mux8_32bit:

```
# time =  0 , Out=11100000000000000000000000000110
# time = 20 , Out=11111111111111111000000000000110
```

mux4_32bit:

```
# time =  0 , Out=00000000000000000000000000000111
# time = 20 , Out=00000000000000000000000000000110
# time = 40 , Out=00000001110000000000000000000110
# time = 60 , Out=11100000000000000000000000000110
```

mux2to1_3bit:

```
# time =  0, input0 = 001, input1 = 100, selectionBit = 0, result = 001
# time = 20, input0 = 000, input1 = 111, selectionBit = 1, result = 111
```

mux2to1_gate_32:

```
# time =  0 , Out=00000000000000000000000000000111
# time = 20 , Out=00000000000000000000000000000110
```

mips_registers:

```
# time =  0, clock = 1 read_register_1 = 001, read_register_2 = 010, read_data_1 = 00000000000000000000000000000001, read_data_2 = 00000000000000000000000000000010
# time = 20, clock = 0 read_register_1 = 001, read_register_2 = 010, read_data_1 = 00000000000000000000000000000001, read_data_2 = 00000000000000000000000000000010
# time = 40, clock = 1 read_register_1 = 101, read_register_2 = 100, read_data_1 = 01010101010101010101010101010101, read_data_2 = 00000000000000000000000000000100
# time = 60, clock = 0 read_register_1 = 101, read_register_2 = 100, read_data_1 = 01010101010101010101010101010101, read_data_2 = 00000000000000000000000000000100
# time = 80, clock = 1 read_register_1 = 111, read_register_2 = 110, read_data_1 = 00000000000000000000000000000111, read_data_2 = 00000000000000000000000000000110
# time = 100, clock = 0 read_register_1 = 101, read_register_2 = 100, read_data_1 = 01010101010101010101010101010101, read_data_2 = 00000000000000000000000000000100
# time = 120, clock = 1 read_register_1 = 101, read_register_2 = 100, read_data_1 = 11111111111111111111111111111111, read_data_2 = 00000000000000000000000000000100
```

mips_data_memory:

```
# time =  0, address = 00000000000000000000000000000001, read_data = 00000000000000000000000000010100, write_data = 00000000000000000000000000000111, read_signal = 1, writ
e_signal = 0, clock = 1
# time = 20, address = 00000000000000000000000000000001, read_data = 00000000000000000000000000010100, write_data = 00000000000000000000000000000111, read_signal = 1, writ
e_signal = 0, clock = 0
# time = 40, address = 00000000000000000000000000000001, read_data = 11110000111100001111000011110000, write_data = 11110000111100001111000011110000, read_signal = 0, writ
e_signal = 1, clock = 1
# time = 60, address = 00000000000000000000000000000001, read_data = 11110000111100001111000011110000, write_data = 11110000111100001111000011110000, read_signal = 0, writ
e_signal = 1, clock = 0
# time = 80, address = 00000000000000000000000000000001, read_data = 11110000111100001111000011110000, write_data = 00000000000000000000000000000111, read_signal = 1, writ
e_signal = 0, clock = 1
# time = 100, address = 00000000000000000000000000000001, read_data = 11110000111100001111000011110000, write_data = 00000000000000000000000000000111, read_signal = 1, wri
te_signal = 0, clock = 0
```

main_control:

```
# time=  0,
# opcode= 0000,
# RegDst= 1,
# ALUsrc= 0,
# MemtoReg= 0,
# RegWrite= 1,
# MemRead= 0,
# MemWrite= 0,
# Branch= 0,
# Branch_not= 0,
# ALUop= 011,
# zeroExt= 0
```

```
# time= 20,
# opcode= 0001,
# RegDst= 0,
# ALUsrc= 1,
# MemtoReg= 0,
# RegWrite= 1,
# MemRead= 0,
# MemWrite= 0,
# Branch= 0,
# Branch_not= 0,
# ALUop= 000,
# zeroExt= 0
```

```
# time= 40,
# opcode= 0010,
# RegDst= 0,
# ALUsrc= 1,
# MemtoReg= 0,
# RegWrite= 1,
# MemRead= 0,
# MemWrite= 0,
# Branch= 0,
# Branch_not= 0,
# ALUop= 110,
# zeroExt= 1
```

```
# time= 60,
# opcode= 0011,
# RegDst= 0,
# ALUsrc= 1,
# MemtoReg= 0,
# RegWrite= 1,
# MemRead= 0,
# MemWrite= 0,
# Branch= 0,
# Branch_not= 0,
# ALUop= 111,
# zeroExt= 1
```

```
# time= 80,
# opcode= 0100,
# RegDst= 0,
# ALUsrc= 1,
# MemtoReg= 0,
# RegWrite= 1,
# MemRead= 0,
# MemWrite= 0,
# Branch= 0,
# Branch_not= 0,
# ALUop= 101,
# zeroExt= 1
```

```
# time= 100,
# opcode= 0101,
# RegDst= 0,
# ALUsrc= 0,
# MemtoReg= 0,
# RegWrite= 0,
# MemRead= 0,
# MemWrite= 0,
# Branch= 1,
# Branch_not= 0,
# ALUop= 010,
# zeroExt= 0
```

```
# time= 120,
# opcode= 0110,
# RegDst= 0,
# ALUsrc= 0,
# MemtoReg= 0,
# RegWrite= 0,
# MemRead= 0,
# MemWrite= 0,
# Branch= 0,
# Branch_not= 1,
# ALUop= 010,
# zeroExt= 0
```

```
# time= 140,
# opcode= 0111,
# RegDst= 0,
# ALUsrc= 1,
# MemtoReg= 0,
# RegWrite= 1,
# MemRead= 0,
# MemWrite= 0,
# Branch= 0,
# Branch_not= 0,
# ALUop= 100,
# zeroExt= 0
```

```
# time= 160,
# opcode= 1000,
# RegDst= 0,
# ALUsrc= 1,
# MemtoReg= 1,
# RegWrite= 1,
# MemRead= 1,
# MemWrite= 0,
# Branch= 0,
# Branch_not= 0,
# ALUop= 000,
# zeroExt= 0
```

```
# time= 180,
# opcode= 1001,
# RegDst= 0,
# ALUsrc= 1,
# MemtoReg= 0,
# RegWrite= 0,
# MemRead= 0,
# MemWrite= 1,
# Branch= 0,
# Branch_not= 0,
# ALUop= 000,
# zeroExt= 0
```

intruction_memory:

```
# time=  0, read_address= 00000000000000000000000000000000, instruction= 0000001010011000, clock= 1
# time= 20, read_address= 00000000000000000000000000000000, instruction= 0000001010011000, clock= 0
# time= 40, read_address= 00000000000000000000000000000001, instruction= 0000000100101000, clock= 1
# time= 60, read_address= 00000000000000000000000000000001, instruction= 0000000100101000, clock= 0
# time= 80, read_address= 00000000000000000000000000000010, instruction= 0000001010100001, clock= 1
```

half_adder:

```
# time =  0, a =0, b=0, sum=0, carry_out=0
# time = 20, a =1, b=0, sum=1, carry_out=0
# time = 40, a =0, b=1, sum=1, carry_out=0
# time = 60, a =1, b=1, sum=0, carry_out=1
```

full_adder:

```
# time =  0, a =0, b=0, carry_in=0, sum=0, carry_out=0
# time = 20, a =0, b=0, carry_in=1, sum=1, carry_out=0
# time = 40, a =0, b=1, carry_in=0, sum=1, carry_out=0
# time = 60, a =0, b=1, carry_in=1, sum=0, carry_out=1
# time = 80, a =1, b=0, carry_in=0, sum=1, carry_out=0
# time = 100, a =1, b=0, carry_in=1, sum=0, carry_out=1
# time = 120, a =1, b=1, carry_in=0, sum=0, carry_out=1
# time = 140, a =1, b=1, carry_in=1, sum=1, carry_out=1
```

full_adder_32bit:

```
# time =  0, a =00000000000000000000000000000110, b=00000000000000000000000000000111, , out=00000000000000000000000000001101
# time = 20, a =00000000000100000100000000000001, b=00000000000000000100000000000010, , out=00000000000100001000000000000011
```

and_32bit:

```
# time =  0, a =01111011111111111111111111111110, b=01111111111111111100001111111111, , out=01111011111111111100001111111110
# time = 20, a =11100000000000000000000000000000, b=00000000000000000000000000000111, , out=00000000000000000000000000000000
```

alu_control:

```
# time=  0, ALUop= 011, funct= 000, ALUControl= 110
# time= 20, ALUop= 011, funct= 001, ALUControl= 000
# time= 40, ALUop= 011, funct= 010, ALUControl= 010
# time= 60, ALUop= 011, funct= 011, ALUControl= 001
# time= 80, ALUop= 011, funct= 100, ALUControl= 101
# time= 100, ALUop= 000, funct= 100, ALUControl= 000
# time= 120, ALUop= 110, funct= 100, ALUControl= 110
# time= 140, ALUop= 111, funct= 000, ALUControl= 111
# time= 160, ALUop= 101, funct= 000, ALUControl= 101
# time= 180, ALUop= 010, funct= 000, ALUControl= 010
# time= 200, ALUop= 100, funct= 110, ALUControl= 100
# time= 220, ALUop= 000, funct= 000, ALUControl= 000
# time= 240, ALUop= 011, funct= 101, ALUControl= 111
```

alu_32:

```
# time =  0, A = 00000000000000000000000000000001, B=00000000000000000000000000000010, R=00000000000000000000000000000011, S=000
# time = 20, A = 00000000000000000000000000110001, B=00000000000000000000000000110010, R=00000000000000000000000001100011, S=000
# time = 40, A = 11111111111111111111111111111111, B=00000000000000000000000000000000, R=11111111111111111111111111111111, S=001
# time = 60, A = 01110000000011111000000000000000, B=01110001111100000000000000000000, R=00000001111111111000000000000000, S=001
# time = 80, A = 00000000000000000000000000011000, B=00000000000000000000000000000100, R=00000000000000000000000000010100, S=010
# time = 100, A = 00000000000000000000000000011000, B=00000000000000000000001000000100, R=11111111111111111111111000010100, S=010
# time = 120, A = 00000000000000000000000000000000, B=00000000000000000000000000000001, R=00000000000000000000000000000001, S=100
# time = 140, A = 00000000000000000000000000000001, B=00000000000000000000000000000000, R=00000000000000000000000000000000, S=100
# time = 160, A = 00000000000000000000000000000001, B=00000000000000000000000000000001, R=11111111111111111111111111111110, S=101
# time = 180, A = 00000000000000000000000000000001, B=00000000000000000000000000000000, R=11111111111111111111111111111110, S=101
# time = 200, A = 00000000000000000000000000010101, B=00000000000000000000000000010101, R=00000000000000000000000000010101, S=110
# time = 220, A = 00000000000000000000000000000001, B=00000000000000000000000000000000, R=00000000000000000000000000000000, S=110
# time = 240, A = 00000000000000000000000000000011, B=00000000000000000000000000000101, R=00000000000000000000000000000111, S=111
# time = 260, A = 00000000000000000000000000011000, B=00000000000000000000000000000100, R=00000000000000000000000000011100, S=111
```

MODELSIM I/Os

Data Memory

```
00000000000000000000000000000000
00000000000000000000000000010100
00000000000000000000000000000010
00000000000000000000000000000011
00000000000000000000000000000100
00000000000000000000000000000101
00000000000000000000000000000110
00000000000000000000000000000111
00000000000000000000000000001000
00000000000000000000000000001001
00000000000000000000000000001010
00000000000000000000000000001011
00000000000000000000000000001100
00000000000000000000000000001101
00000000000000000000000000001110
00000000000000000000000000001111
00000000000000000000000000010000
00000000000000000000000000010001
00000000000000000000000000010010
00000000000000000000000000010011
00000000000000000000000000010100
00000000000000000000000000010101
00000000000000000000000000010110
00000000000000000000000000010111
00000000000000000000000000011000
00000000000000000000000000011001
00000000000000000000000000011010
00000000000000000000000000011011
00000000000000000000000000011100
00000000000000000000000000011101
00000000000000000000000000011110
00000000000000000000000000011111
00000000000000000000000000000000
00000000000000000000000000000001
00000000000000000000000000000010
00000000000000000000000000000011
00000000000000000000000000000100
00000000000000000000000000000101
00000000000000000000000000000110
00000000000000000000000000000111
```

Register Input

```
00000000000000000000000000000000
00000000000000000000000000000001
00000000000000000000000000000010
00000000000000000000000000000011
00000000000000000000000000000100
00000000000000000000000000000101
00000000000000000000000000000110
00000000000000000000000000000111
```

INSTRUCTIONS

```
0000001010011000 //and $3 = $2 and $1
0000000100101000 //and $5 = $4 and $0
0000001010100001 //add $4 = $2 + $1
0000001011000001 //add $0 = $3 + $1
0000100011101010 //sub $5 = $4 - $3
0000001000101010 //sub $5 = $1 - $0
0000101100110011 //xor $6 = $5 xor $4
0000000010001000 //xor $1 = $0 xor $2
0000000110111100 // $7 = $0 nor $6
0000001111010000 // $2 = $1 nor $7
0000111110001101 // $1 = $7 or $6
0000001000010000 // $2 = $1 or $0
0001001010000101 // addi $2 = $1 + 5
0001000011001101 // addi $3 = $0 + 13
0010010011000110 // andi $3 = $2 && 6
0010000011000000 // andi $3 = $0 && 0
0011000111000001 // ori $7 = $0 || 1
0011111100111111 // ori $4 = $7 || 63
0100000111000001 // nori $7 = $0 ~|| 1
0100111100111111 // nori $4 = $7 || 63
0101100101000010 // beq $4 $5 2
0101011100000011 // beq $3 $4 3
xxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxx
0111111001000000 // slti $7 = $1 < 0
0111111000000001 // slti $7 = $0 < 1
0110100011000010 // bne $3 $4 2
0110110101000100 // bne $6 $5 4
xxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxx
1000000001000001 // lw $1 = M[$0 + 1]
1000111010000001 // lw $2 = M[$7 + 1]
1001000010000000 // sw Mem[$0+0] = $2
1001010101000111 // sw Mem[$2+7] = $5
```

Register_out

```
00000000000000000000000000000000
00000000000000000000000000010100
00000000000000000000000000000011
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000001
00000000000000000000000000000010
11111111111111111111111111111110
```

AND Test 1: PASS

```
# time=  0, clock= 1, PC= 00000000000000000000000000000000, instruction= 0000001010011000,
# opcode= 0000, rs= 001, rt= 010, rd= 011, funct= 000, imm6= 011000
# read_data_1= 00000000000000000000000000000001, read_data_2= 00000000000000000000000000000010,
# write_data= 00000000000000000000000000000000,
# ALUop= 011, ALUcontrol= 110, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000011000, mux_result= 00000000000000000000000000000010
# RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

AND Test 2: PASS

```
time= 40, clock= 1, PC= 00000000000000000000000000000001, instruction= 0000000100101000,
opcode= 0000, rs= 000, rt= 100, rd= 101, funct= 000, imm6= 101000
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000100,
write_data= 00000000000000000000000000000000,
ALUop= 011, ALUcontrol= 110, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 11111111111111111111111111101000, mux_result= 00000000000000000000000000000100
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

ADD Test 1: PASS

```
time= 80, clock= 1, PC= 00000000000000000000000000000010, instruction= 0000001010100001,
opcode= 0000, rs= 001, rt= 010, rd= 100, funct= 001, imm6= 100001
read_data_1= 00000000000000000000000000000001, read_data_2= 00000000000000000000000000000010,
write_data= 00000000000000000000000000000011,
ALUop= 011, ALUcontrol= 000, ALUresult= 00000000000000000000000000000011, ALUzero= 0, extended= 11111111111111111111111111100001, mux_result= 00000000000000000000000000000010
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

ADD Test 2 + $0 Protection Test: PASS

```
time= 120, clock= 1, PC= 00000000000000000000000000000011, instruction= 0000001011000001,
opcode= 0000, rs= 001, rt= 011, rd= 000, funct= 001, imm6= 000001
read_data_1= 00000000000000000000000000000001, read_data_2= 00000000000000000000000000000000,
write_data= 00000000000000000000000000000001,
ALUop= 011, ALUcontrol= 000, ALUresult= 00000000000000000000000000000001, ALUzero= 0, extended= 00000000000000000000000000000001, mux_result= 00000000000000000000000000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

SUB Test 1: PASS

```
time= 160, clock= 1, PC= 00000000000000000000000000000100, instruction= 0000100011101010,
opcode= 0000, rs= 100, rt= 011, rd= 101, funct= 010, imm6= 101010
read_data_1= 00000000000000000000000000000011, read_data_2= 00000000000000000000000000000000,
write_data= 00000000000000000000000000000011,
ALUop= 011, ALUcontrol= 010, ALUresult= 00000000000000000000000000000011, ALUzero= 0, extended= 11111111111111111111111111101010, mux_result= 00000000000000000000000000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

SUB Test 2: PASS

```
time= 200, clock= 1, PC= 00000000000000000000000000000101, instruction= 0000001000101010,
opcode= 0000, rs= 001, rt= 000, rd= 101, funct= 010, imm6= 101010
read_data_1= 00000000000000000000000000000001, read_data_2= 00000000000000000000000000000000,
write_data= 00000000000000000000000000000001,
ALUop= 011, ALUcontrol= 010, ALUresult= 00000000000000000000000000000001, ALUzero= 0, extended= 11111111111111111111111111101010, mux_result= 00000000000000000000000000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

XOR Test 1: PASS

```
time= 240, clock= 1, PC= 00000000000000000000000000000110, instruction= 0000101100110011,
opcode= 0000, rs= 101, rt= 100, rd= 110, funct= 011, imm6= 110011
read_data_1= 00000000000000000000000000000001, read_data_2= 00000000000000000000000000000011,
write_data= 00000000000000000000000000000010,
ALUop= 011, ALUcontrol= 001, ALUresult= 00000000000000000000000000000010, ALUzero= 0, extended= 11111111111111111111111111110011, mux_result= 00000000000000000000000000000011
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

XOR Test 2: PASS

```
time= 280, clock= 1, PC= 00000000000000000000000000000111, instruction= 0000000010001000,
opcode= 0000, rs= 000, rt= 010, rd= 001, funct= 000, imm6= 001000
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000010,
write_data= 00000000000000000000000000000000,
ALUop= 011, ALUcontrol= 110, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000001000, mux_result= 00000000000000000000000000000010
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

NOR Test 1: PASS

```
time= 320, clock= 1, PC= 00000000000000000000000000001000, instruction= 0000000110111100,
opcode= 0000, rs= 000, rt= 110, rd= 111, funct= 100, imm6= 111100
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000010,
write_data= 11111111111111111111111111111101,
ALUop= 011, ALUcontrol= 101, ALUresult= 11111111111111111111111111111101, ALUzero= 0, extended= 11111111111111111111111111111100, mux_result= 00000000000000000000000000000010
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

NOR Test 2: PASS

```
time= 360, clock= 1, PC= 00000000000000000000000000001001, instruction= 0000001111010000,
opcode= 0000, rs= 001, rt= 111, rd= 010, funct= 000, imm6= 010000
read_data_1= 00000000000000000000000000000000, read_data_2= 11111111111111111111111111111101,
write_data= 00000000000000000000000000000000,
ALUop= 011, ALUcontrol= 110, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000010000, mux_result= 11111111111111111111111111111101
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

OR Test 1: PASS

```
time= 400, clock= 1, PC= 00000000000000000000000000001010, instruction= 0000111110001101,
opcode= 0000, rs= 111, rt= 110, rd= 001, funct= 101, imm6= 001101
read_data_1= 11111111111111111111111111111101, read_data_2= 00000000000000000000000000000010,
write_data= 11111111111111111111111111111111,
ALUop= 011, ALUcontrol= 111, ALUresult= 11111111111111111111111111111111, ALUzero= 0, extended= 00000000000000000000000000001101, mux_result= 00000000000000000000000000000010
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

OR Test 2: PASS

```
time= 440, clock= 1, PC= 00000000000000000000000000001011, instruction= 0000001000010000,
opcode= 0000, rs= 001, rt= 000, rd= 010, funct= 000, imm6= 010000
read_data_1= 11111111111111111111111111111111, read_data_2= 00000000000000000000000000000000,
write_data= 00000000000000000000000000000000,
ALUop= 011, ALUcontrol= 110, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000010000, mux_result= 00000000000000000000000000000000
RegDst= 1, ALUsrc= 0, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

ADDI Test 1: PASS

```
time= 480, clock= 1, PC= 00000000000000000000000000001100, instruction= 0001001010000101,
opcode= 0001, rs= 001, rt= 010, rd= 000, funct= 101, imm6= 000101
read_data_1= 11111111111111111111111111111111, read_data_2= 00000000000000000000000000000000,
write_data= 00000000000000000000000000000100,
ALUop= 000, ALUcontrol= 000, ALUresult= 00000000000000000000000000000100, ALUzero= 0, extended= 00000000000000000000000000000101, mux_result= 00000000000000000000000000000101
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

ADDI Test 2: PASS

```
time= 520, clock= 1, PC= 00000000000000000000000000001101, instruction= 0001000011001101,
opcode= 0001, rs= 000, rt= 011, rd= 001, funct= 101, imm6= 001101
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000000,
write_data= 00000000000000000000000000001101,
ALUop= 000, ALUcontrol= 000, ALUresult= 00000000000000000000000000001101, ALUzero= 0, extended= 00000000000000000000000000001101, mux_result= 00000000000000000000000000001101
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

ANDI Test 1: PASS

```
time= 560, clock= 1, PC= 00000000000000000000000000001110, instruction= 0010010011000110,
opcode= 0010, rs= 010, rt= 011, rd= 000, funct= 110, imm6= 000110
read_data_1= 00000000000000000000000000000100, read_data_2= 00000000000000000000000000001101,
write_data= 00000000000000000000000000000100,
ALUop= 110, ALUcontrol= 110, ALUresult= 00000000000000000000000000000100, ALUzero= 0, extended= 00000000000000000000000000000110, mux_result= 00000000000000000000000000000110
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 1
```

ANDI Test 2: PASS

```
time= 600, clock= 1, PC= 00000000000000000000000000001111, instruction= 0010000011000000,
opcode= 0010, rs= 000, rt= 011, rd= 000, funct= 000, imm6= 000000
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000100,
write_data= 00000000000000000000000000000000,
ALUop= 110, ALUcontrol= 110, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000000000, mux_result= 00000000000000000000000000000000
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 1
```

ORI Test 1: PASS

```
time= 640, clock= 1, PC= 00000000000000000000000000010000, instruction= 0011000111000001,
opcode= 0011, rs= 000, rt= 111, rd= 000, funct= 001, imm6= 000001
read_data_1= 00000000000000000000000000000000, read_data_2= 11111111111111111111111111111101,
write_data= 00000000000000000000000000000001,
ALUop= 111, ALUcontrol= 111, ALUresult= 00000000000000000000000000000001, ALUzero= 0, extended= 00000000000000000000000000000001, mux_result= 00000000000000000000000000000001
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 1
```

ORI Test 2: PASS

```
time= 680, clock= 1, PC= 00000000000000000000000000010001, instruction= 0011111100111111,
opcode= 0011, rs= 111, rt= 100, rd= 111, funct= 111, imm6= 111111
read_data_1= 00000000000000000000000000000001, read_data_2= 00000000000000000000000000000011,
write_data= 00000000000000000000000000111111,
ALUop= 111, ALUcontrol= 111, ALUresult= 00000000000000000000000000111111, ALUzero= 0, extended= 00000000000000000000000000111111, mux_result= 00000000000000000000000000111111
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 1
```

NORI Test 1: PASS

```
time= 720, clock= 1, PC= 00000000000000000000000000010010, instruction= 0100000111000001,
opcode= 0100, rs= 000, rt= 111, rd= 000, funct= 001, imm6= 000001
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000001,
write_data= 11111111111111111111111111111110,
ALUop= 101, ALUcontrol= 101, ALUresult= 11111111111111111111111111111110, ALUzero= 0, extended= 00000000000000000000000000000001, mux_result= 00000000000000000000000000000001
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 1
```

NORI Test 2: PASS

```
time= 760, clock= 1, PC= 00000000000000000000000000010011, instruction= 0100111100111111,
opcode= 0100, rs= 111, rt= 100, rd= 111, funct= 111, imm6= 111111
read_data_1= 11111111111111111111111111111110, read_data_2= 00000000000000000000000000111111,
write_data= 00000000000000000000000000000000,
ALUop= 101, ALUcontrol= 101, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000111111, mux_result= 00000000000000000000000000111111
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 1
```

BEQ Test 1: PASS

```
time= 800, clock= 1, PC= 00000000000000000000000000010100, instruction= 0101100101000010,
opcode= 0101, rs= 100, rt= 101, rd= 000, funct= 010, imm6= 000010
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000001,
write_data= 11111111111111111111111111111111,
ALUop= 010, ALUcontrol= 010, ALUresult= 11111111111111111111111111111111, ALUzero= 0, extended= 00000000000000000000000000000010, mux_result= 00000000000000000000000000000001
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite= 0, MemRead= 0, MemWrite= 0, Branch= 1, Branch_not= 0, extend_type= 0
```

BEQ Test 2: PASS

(You can see the PC changes other Instruction)

```
time= 840, clock= 1, PC= 00000000000000000000000000010101, instruction= 0101011100000011,
opcode= 0101, rs= 011, rt= 100, rd= 000, funct= 011, imm6= 000011
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000000,
write_data= 00000000000000000000000000000000,
ALUop= 010, ALUcontrol= 010, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000000011, mux_result= 00000000000000000000000000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite= 0, MemRead= 0, MemWrite= 0, Branch= 1, Branch_not= 0, extend_type= 0
```

SLTI Test 1: PASS

```
time= 880, clock= 1, PC= 00000000000000000000000000011000, instruction= 0111111001000000,
opcode= 0111, rs= 111, rt= 001, rd= 000, funct= 000, imm6= 000000
read_data_1= 11111111111111111111111111111110, read_data_2= 11111111111111111111111111111111,
write_data= 00000000000000000000000000000001,
ALUop= 100, ALUcontrol= 100, ALUresult= 00000000000000000000000000000001, ALUzero= 0, extended= 00000000000000000000000000000000, mux_result= 00000000000000000000000000000000
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

SLTI Test 2: PASS

```
time= 920, clock= 1, PC= 00000000000000000000000000011001, instruction= 0111111000000001,
opcode= 0111, rs= 111, rt= 000, rd= 000, funct= 001, imm6= 000001
read_data_1= 11111111111111111111111111111110, read_data_2= 00000000000000000000000000000000,
write_data= 00000000000000000000000000000001,
ALUop= 100, ALUcontrol= 100, ALUresult= 00000000000000000000000000000001, ALUzero= 0, extended= 00000000000000000000000000000001, mux_result= 00000000000000000000000000000001
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite= 1, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

BNE Test 1: PASS

```
time= 960, clock= 1, PC= 00000000000000000000000000011010, instruction= 0110100011000010,
opcode= 0110, rs= 100, rt= 011, rd= 000, funct= 010, imm6= 000010
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000000,
write_data= 00000000000000000000000000000000,
ALUop= 010, ALUcontrol= 010, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000000010, mux_result= 00000000000000000000000000000000
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 1, extend_type= 0
```

BNE Test 2: PASS

(You can see the PC changes other Instruction)

```
time= 1000, clock= 1, PC= 00000000000000000000000000011011, instruction= 0110110101000100,
opcode= 0110, rs= 110, rt= 101, rd= 000, funct= 100, imm6= 000100
read_data_1= 00000000000000000000000000000010, read_data_2= 00000000000000000000000000000001,
write_data= 00000000000000000000000000000001,
ALUop= 010, ALUcontrol= 010, ALUresult= 00000000000000000000000000000001, ALUzero= 0, extended= 00000000000000000000000000000100, mux_result= 00000000000000000000000000000001
RegDst= 0, ALUsrc= 0, MemtoReg= 0, RegWrite= 0, MemRead= 0, MemWrite= 0, Branch= 0, Branch_not= 1, extend_type= 0
```

LW Test 1: PASS

```
time= 1040, clock= 1, PC= 00000000000000000000000000011111, instruction= 1000000001000001,
opcode= 1000, rs= 000, rt= 001, rd= 000, funct= 001, imm6= 000001
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000001,
write_data= 00000000000000000000000000010100,
ALUop= 000, ALUcontrol= 000, ALUresult= 00000000000000000000000000000001, ALUzero= 0, extended= 00000000000000000000000000000001, mux_result= 00000000000000000000000000000001
RegDst= 0, ALUsrc= 1, MemtoReg= 1, RegWrite= 1, MemRead= 1, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

LW Test 2: PASS

```
time= 1080, clock= 1, PC= 00000000000000000000000000100000, instruction= 1000110010000001,
opcode= 1000, rs= 110, rt= 010, rd= 000, funct= 001, imm6= 000001
read_data_1= 00000000000000000000000000000010, read_data_2= 00000000000000000000000000000100,
write_data= 00000000000000000000000000000011,
ALUop= 000, ALUcontrol= 000, ALUresult= 00000000000000000000000000000011, ALUzero= 0, extended= 00000000000000000000000000000001, mux_result= 00000000000000000000000000000001
RegDst= 0, ALUsrc= 1, MemtoReg= 1, RegWrite= 1, MemRead= 1, MemWrite= 0, Branch= 0, Branch_not= 0, extend_type= 0
```

SW Test 1: PASS

```
time= 1120, clock= 1, PC= 00000000000000000000000000100001, instruction= 1001000010000000,
opcode= 1001, rs= 000, rt= 010, rd= 000, funct= 000, imm6= 000000
read_data_1= 00000000000000000000000000000000, read_data_2= 00000000000000000000000000000011,
write_data= 00000000000000000000000000000000,
ALUop= 000, ALUcontrol= 000, ALUresult= 00000000000000000000000000000000, ALUzero= 1, extended= 00000000000000000000000000000000, mux_result= 00000000000000000000000000000000
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite= 0, MemRead= 0, MemWrite= 1, Branch= 0, Branch_not= 0, extend_type= 0
```

SW Test 2: PASS

```
time= 1160, clock= 1, PC= 00000000000000000000000000100010, instruction= 1001010101000111,
opcode= 1001, rs= 010, rt= 101, rd= 000, funct= 111, imm6= 000111
read_data_1= 00000000000000000000000000000011, read_data_2= 00000000000000000000000000000001,
write_data= 00000000000000000000000000001010,
ALUop= 000, ALUcontrol= 000, ALUresult= 00000000000000000000000000001010, ALUzero= 0, extended= 00000000000000000000000000000111, mux_result= 00000000000000000000000000000111
RegDst= 0, ALUsrc= 1, MemtoReg= 0, RegWrite= 0, MemRead= 0, MemWrite= 1, Branch= 0, Branch_not= 0, extend_type= 0
```

Data Memory OUTPUT

```
00000000000000000000000000000011
00000000000000000000000000010100
00000000000000000000000000000010
00000000000000000000000000000011
00000000000000000000000000000100
00000000000000000000000000000101
00000000000000000000000000000110
00000000000000000000000000000111
00000000000000000000000000001000
00000000000000000000000000001001
00000000000000000000000000000001
```