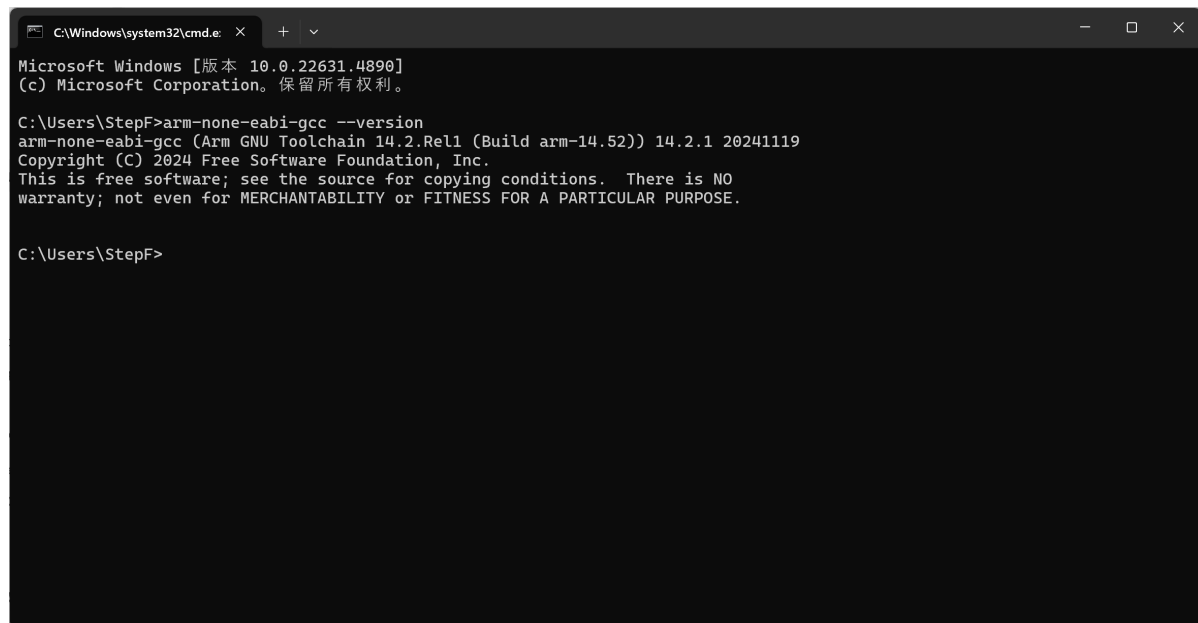
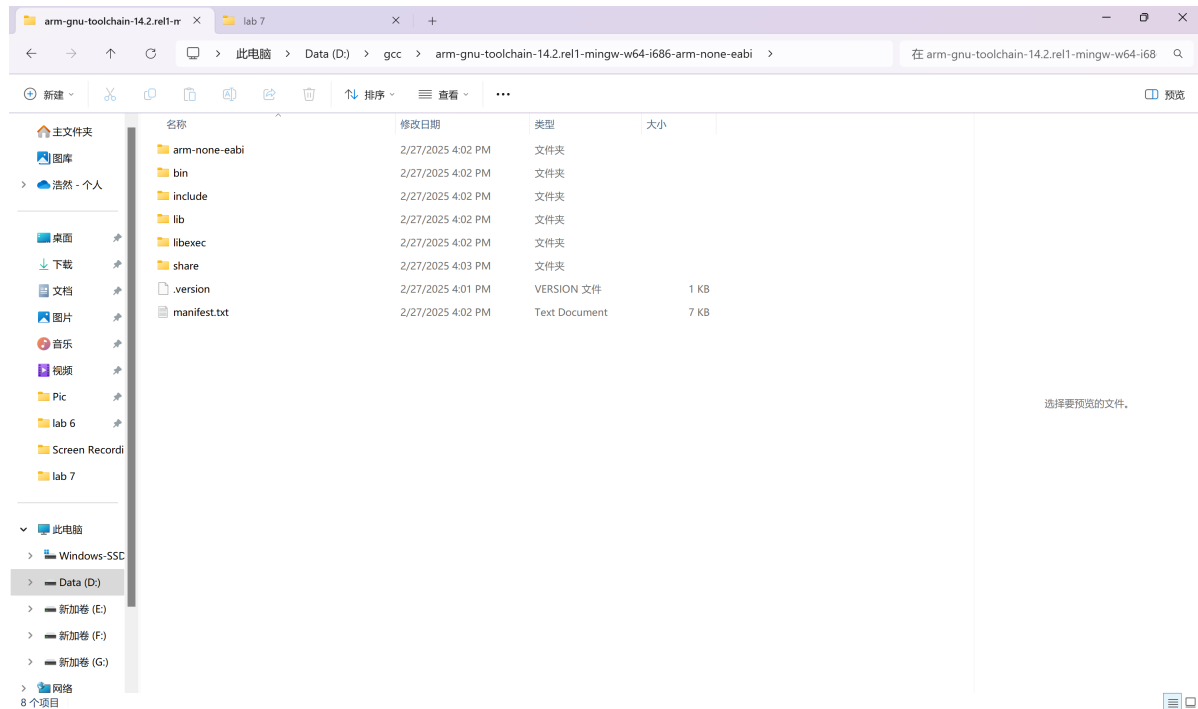


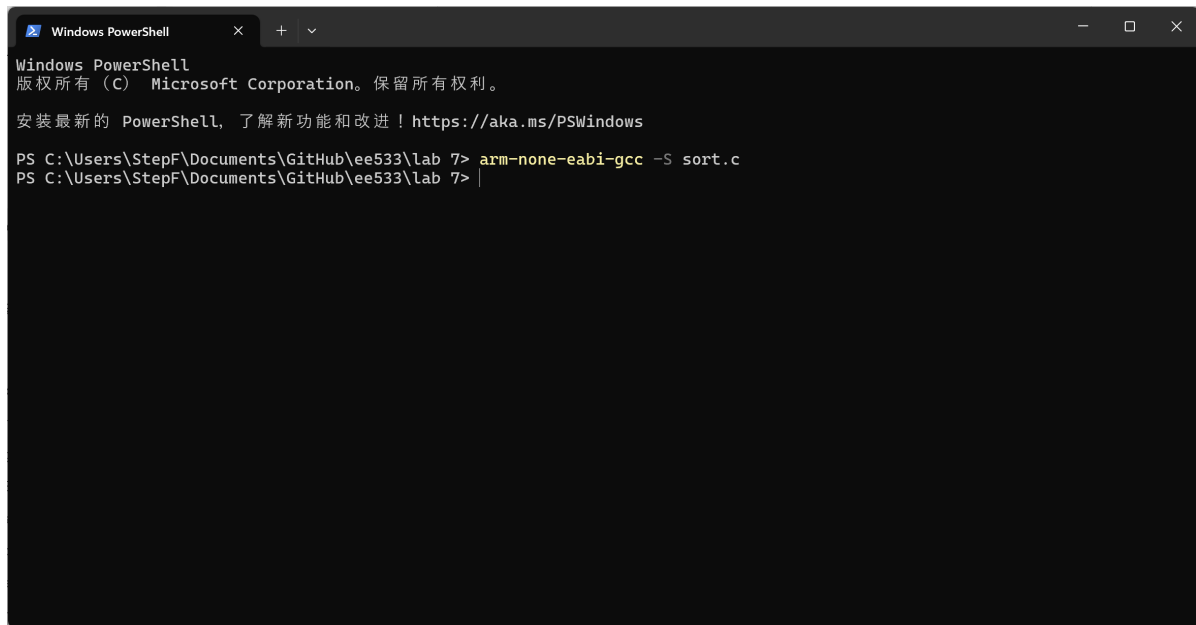
EE533_Lab7_Report

1. Find what Instruction We Need to Implement

1.1 ARM C/C++ Cross-compiler Installation



1.2 Assembly Code for Bubble Sort



```
Windows PowerShell
版权所有 (C) Microsoft Corporation。保留所有权利。

安装最新的 PowerShell，了解新功能和改进！https://aka.ms/PSWindows

PS C:\Users\StepF\Documents\GitHub\ee533\lab 7> arm-none-eabi-gcc -S sort.c
PS C:\Users\StepF\Documents\GitHub\ee533\lab 7> |
```

- sort.c

```
int main() {
    int array[10] = {323, 123, -455, 2, 98, 125, 10, 65, -56, 0};
    int i, j, swap;
    for (i = 0 ; i < 10; i++) {
        for (j = i+1 ; j < 10 ; j++) {
            if (array[j] < array[i]) {
                swap = array[j];
                array[j] = array[i];
                array[i] = swap;
            }
        }
    }
}
```

- sort.s

```
.cpu arm7tdmi
.arch armv4t
.fpu softvfp
.eabi_attribute 20, 1
.eabi_attribute 21, 1
.eabi_attribute 23, 3
.eabi_attribute 24, 1
.eabi_attribute 25, 1
.eabi_attribute 26, 1
.eabi_attribute 30, 6
.eabi_attribute 34, 0
.eabi_attribute 18, 4
.file "sort.c"
.text
.section .rodata
.align 2
.LC0:
```

```

.word    323
.word    123
.word    -455
.word    2
.word    98
.word    125
.word    10
.word    65
.word    -56
.word    0
.text
.align   2
.global  main
.syntax  unified
.arm
.type    main, %function
main:
    @ Function supports interworking.
    @ args = 0, pretend = 0, frame = 56
    @ frame_needed = 1, uses_anonymous_args = 0
    push    {fp, lr}
    add fp, sp, #4
    sub sp, sp, #56
    ldr r3, .L8
    sub ip, fp, #56
    mov lr, r3
    ldmia   lr!, {r0, r1, r2, r3}
    stmia   ip!, {r0, r1, r2, r3}
    ldmia   lr!, {r0, r1, r2, r3}
    stmia   ip!, {r0, r1, r2, r3}
    ldm lr, {r0, r1}
    stm ip, {r0, r1}
    mov r3, #0
    str r3, [fp, #-8]
    b      .L2
.L6:
    ldr r3, [fp, #-8]
    add r3, r3, #1
    str r3, [fp, #-12]
    b      .L3
.L5:
    ldr r3, [fp, #-12]
    lsl r3, r3, #2
    sub r3, r3, #4
    add r3, r3, fp
    ldr r2, [r3, #-52]
    ldr r3, [fp, #-8]
    lsl r3, r3, #2
    sub r3, r3, #4
    add r3, r3, fp
    ldr r3, [r3, #-52]
    cmp r2, r3
    bge .L4
    ldr r3, [fp, #-12]
    lsl r3, r3, #2
    sub r3, r3, #4

```

```

    add r3, r3, fp
    ldr r3, [r3, #-52]
    str r3, [fp, #-16]
    ldr r3, [fp, #-8]
    lsl r3, r3, #2
    sub r3, r3, #4
    add r3, r3, fp
    ldr r2, [r3, #-52]
    ldr r3, [fp, #-12]
    lsl r3, r3, #2
    sub r3, r3, #4
    add r3, r3, fp
    str r2, [r3, #-52]
    ldr r3, [fp, #-8]
    lsl r3, r3, #2
    sub r3, r3, #4
    add r3, r3, fp
    ldr r2, [fp, #-16]
    str r2, [r3, #-52]
.L4:
    ldr r3, [fp, #-12]
    add r3, r3, #1
    str r3, [fp, #-12]
.L3:
    ldr r3, [fp, #-12]
    cmp r3, #9
    ble .L5
    ldr r3, [fp, #-8]
    add r3, r3, #1
    str r3, [fp, #-8]
.L2:
    ldr r3, [fp, #-8]
    cmp r3, #9
    ble .L6
    mov r3, #0
    mov r0, r3
    sub sp, fp, #4
    @ sp needed
    pop {fp, lr}
    bx lr
.L9:
    .align 2
.L8:
    .word .LC0
    .size main, .-main
    .ident "GCC: (Arm GNU Toolchain 14.2.Rel1 (Build arm-14.52)) 14.2.1
20241119"

```

- Included Instructions' Description and Reference

Instruction	Explanation	Reference
add	Add	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/general-data-processing-instructions/add--adc--sub--sbc--and-rsb
sub	Subtract	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/general-data-processing-instructions/add--adc--sub--sbc--and-rsb
mov	Move	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/general-data-processing-instructions/mov-and-mvn
ldm	Load Multiple registers, increment after	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/memory-access-instructions/ldm-and-stm
ldmia	LDMIA is a synonym for LDM	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/memory-access-instructions/ldm-and-stm
stm	Store Multiple registers, increment after	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/memory-access-instructions/ldm-and-stm
stmia	STMIA is a synonym for STM	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/memory-access-instructions/ldm-and-stm
ldr	Load Register with word	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/memory-access-instructions/ldr-and-str--register-offset
str	Store Register word	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/memory-access-instructions/ldr-and-str--register-offset
lsl	Logical Shift Left	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/general-data-processing-instructions/asr--lsl--lsr--ror--and-rrx?lang=en
cmp	Compare	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/general-data-processing-instructions/cmp-and-cmn
push	Push registers onto stack	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/memory-access-instructions/push-and-pop
pop	Pop registers from stack	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/memory-access-instructions/push-and-pop

Instruction	Explanation	Reference
b	Branch	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/branch-and-control-instructions/b--bl--bx--and-blx
bge	Branch if greater than or equal	https://developer.arm.com/documentation/100076/0100/A64-Instruction-Set-Reference/A64-General-Instructions/B-cond/Condition-code-suffixes-and-related-flags
ble	Branch if less or equal	https://developer.arm.com/documentation/100076/0100/A64-Instruction-Set-Reference/A64-General-Instructions/B-cond/Condition-code-suffixes-and-related-flags
bx	Indirect branch	https://developer.arm.com/documentation/dui0552/a/the-cortex-m3-instruction-set/branch-and-control-instructions/b--bl--bx--and-blx

1.3 List of instruction needs to be implemented

1.3.1 Instruction Format Definition

- Instruction Format definition

OP_CODE	Rs	Rt	Rd	Shift	Funct
I_MEM[31:26]	I_MEM[25:21]	I_MEM[20:16]	I_MEM[15:11]	I_MEM[10:6]	I_MEM[5:0]

- OP_Code look up table

Instruction	OP_Code	Description
add	000000	Add
sub	000001	Subtract
mov	000010	Move
ldr	000011	Load Register with word
str	000100	Store Register word
ldm	000101	Load Multiple registers, increment after
stm	000110	Store Multiple registers, increment after
ldmia	000111	LDMIA is a synonym for LDM
stmia	001000	STMIA is a synonym for STM
lsl	001001	Logical Shift Left
cmp	001010	Compare

Instruction	OP_Code	Description
push	001011	Push registers onto stack
pop	001100	Pop registers from stack
b	001101	Branch
bge	001110	Branch if greater than or equal
ble	001111	Branch if less or equal
bx	010000	Indirect branches

1.3.2 Special Registers in ARM ISA

Register Name	Actual Register Location	Description
FP	R11	Frame Pointer
IP	R12	Intra Procedural Call
SP	R13	Stack Pointer
LR	R14	Link Register

2. Pipelined Processor on NetFPGA

2.1 ARM Instruction Generated

```
.data
array: .dword 323, 123, -455, 2, 98, 125, 10, 65, -56, 0
N:     .dword 10

.text
.global _start
_start:
    ldr r4, =array      @ r4 = base address of array
    ldr r5, =N           @ r5 = address of N
    ldr r5, [r5]         @ r5 = N (size of array)
    sub r5, r5, #1       @ r5 = N-1 (outer loop limit)

outer_loop:
    mov r6, #0           @ i = 0

inner_loop:
    sub r7, r6, r5       @ if i >= N-1, exit inner loop
    bge outer_continue

    @ Load array[i] and array[i+1]
    mov r8, r6, LSL #2   @ r8 = i * 4 (word offset)
    add r9, r4, r8       @ r9 = address of array[i]
    ldr r10, [r9]        @ r10 = array[i]
    ldr r11, [r9, #4]    @ r11 = array[i+1]
```

```

    sub r12, r10, r11    @ r12 = array[i] - array[i+1]
    ble no_swap          @ If array[i] <= array[i+1], no swap

    str r11, [r9]         @ array[i] = array[i+1]
    str r10, [r9, #4]     @ array[i+1] = array[i]

no_swap:
    add r6, r6, #1        @ i++
    b inner_loop

outer_continue:
    sub r5, r5, #1        @ Reduce loop limit (N-1, N-2, ...)
    bgt outer_loop        @ If still positive, loop again

end:
    b end                @ Infinite loop (halt)

```

- Assembly Code

```

#0  ldr r4, =array
#1  ldr r5, =N
#2  ldr r5, [r5]
#3  sub r5, r5, #1        @ outer_continue
#4  mov r6, #0            @ outer_loop
#5  mov r8, r6
#6  lsl r8, r8, #2
#7  add r9, r4, r8
#8  ldr r10, [r9]
#9  ldr r11, [r9, #8]
#10 sub r12, r10, r11
#11 ble no_swap
#12 str r11, [r9]
#13 str r10, [r9, #8]
#14 add r6, r6, #1        @ no_swap
#15 sub r7, r6, r5        @ inner_loop
#16 bge outer_continue
#17 b inner_loop
#18 b end                @ end

```

2.2 MIPS Instruction Format

2.2.1 Pseudo Instruction

```

lw r0, array_addr
lw r1, array_size
subi r1, r1, #1
mov r2, #0
outer_loop:
    beq r2, r1, end
    addi r3, r2, #1
inner_loop:
    bgt r3, r1, next_out
    lw r4, r0(r2)
    lw r5, r0(r3)

```



```

blt r4, r5, no_swap
sw r4, r0(r3)
sw r5, r0(r2)
no_swap:
addi r3, r3, #1
j inner_loop
next_out:
addi r2, r2, #1
j outer_loop
end:
j end

```

2.2.2 Real MIPS Instruction

- With manually introduced NOOP, we can avoid data dependency problem and early branch flush problem.

```

movi r1, #9
outer_loop:
noop
noop
beq r2, r1, end
noop
addi r3, r2, #1
inner_loop:
noop
noop
bgt r3, r1, next_out
noop
lw r4, r2(#0)
lw r5, r3(#0)
noop
noop
blt r4, r5, no_swap
noop
sw r4, r3(#0)
sw r5, r2(#0)
no_swap:
addi r3, r3, #1
j inner_loop
noop
next_out:
addi r2, r2, #1
j outer_loop
noop
end:
j end

```

- In

```

#0 movi r1, #9
#1 movi r2, #0
#2 movi r3, #1
#3 noop

```

```

#4  noop
#3  beq r1, r3, end          @outer_loop
#4  noop
#5  beq r2, r1, next_out     @inner_loop
#6  noop
#7  lw r4, r2(#0)
#8  lw r5, r3(#0)
#9  noop
#10 noop
#11 blt r4, r5, no_swap
#12 noop
#13 sw r4, r3(#0)
#14 sw r5, r2(#0)
#15 addi r2, r2, #1          @no_swap
#16 addi r3, r3, #1
#17 j inner_loop
#18 noop
#19 subi r1, r1, #1          @next_out
#20 movi r2, #0
#21 movi r3, #1
#22 j outer_loop
#23 noop
#24 j end                    @end

```

- Instruction OP Code

Instr	OP Code [31:26]
noop	000000
addi	000001
movi	000010
lw	000011
sw	000100
beq	000101
bgt	000110
blt	000111
j	001000

- updated Instr opcode lookup table

Instr	OP Code [31:26]
noop	000000
addi	000001
movi	000010

Instr	OP Code [31:26]
lw	000011
sw	000100
beq	000101
bgt	000110
blt	000111
j	001000
subi	001001

- Instruction Table

Addr	Label	Instr	OP Code [31:26]	Rs [25:21]	Rt [20:16]	Offset [15:0]
0		movi r1, #9	000010	5'd0	5'd1	16'd9
1	outer_loop	noop	000000	5'd0	5'd0	16'd0
2		noop	000000	5'd0	5'd0	16'd0
3		beq r2, r1, end	000101	5'd1	5'd2	16'd24
4		noop	000000	5'd0	5'd0	16'd0
5		addi r3, r2, #1	000001	5'd2	5'd3	16'd1
6	inner_loop	noop	000000	5'd0	5'd0	16'd0
7		noop	000000	5'd0	5'd0	16'd0
8		bgt r3, r1, next_out	000110	5'd1	5'd3	16'd21
9		noop	000000	5'd0	5'd0	16'd0
10		lw r4, r2(#0)	000011	5'd2	5'd4	16'd0
11		lw r5, r3(#0)	000011	5'd3	5'd5	16'd0
12		noop	000000	5'd0	5'd0	16'd0
13		noop	000000	5'd0	5'd0	16'd0
14		blt r4, r5, no_swap	000111	5'd5	5'd4	16'd18
15		noop	000000	5'd0	5'd0	16'd0
16		sw r4, r3(#0)	000100	5'd3	5'd4	16'd0

Addr	Label	Instr	OP Code [31:26]	Rs [25:21]	Rt [20:16]	Offset [15:0]
17		sw r5, r2(#0)	000100	5'd2	5'd5	16'd0
18	no_swap	addi r3, r3, #1	000001	5'd3	5'd3	16'd1
19		j inner_loop	001000	5'd0	5'd0	16'd6
20		noop	000000	5'd0	5'd0	16'd0
21	next_out	addi r2, r2, #1	000001	5'd2	5'd2	16'd1
22		j outer_loop	001000	5'd0	5'd0	16'd1
23		noop	000000	5'd0	5'd0	16'd0
24	end	j end	001000	5'd0	5'd0	16'd24

- New

Addr	Label	Instr	OP Code [31:26]	Rs [25:21]	Rt [20:16]	Offset [15:0]
0		movi r1, #9	000010	5'd0	5'd1	16'd9
1		movi r2, #0	000010	5'd0	5'd2	16'd0
2		movi r3, #1	000010	5'd0	5'd3	16'd1
3		noop	000000	5'd0	5'd0	16'd0
4		noop	000000	5'd0	5'd0	16'd0
5	outer_loop	beq r1, r3, end	000101	5'd1	5'd3	16'd27
6		noop	000000	5'd0	5'd0	16'd0
7	inner_loop	beq r3, r1, next_out	000101	5'd1	5'd3	16'd21
8		noop	000000	5'd0	5'd0	16'd0
9		lw r4, r2(#0)	000011	5'd2	5'd4	16'd0
10		lw r5, r3(#0)	000011	5'd3	5'd5	16'd0
11		noop	000000	5'd0	5'd0	16'd0
12		noop	000000	5'd0	5'd0	16'd0
13		noop	000000	5'd0	5'd0	16'd0
14		blt r4, r5, no_swap	000111	5'd5	5'd4	16'd18
15		noop	000000	5'd0	5'd0	16'd0

Addr	Label	Instr	OP Code [31:26]	Rs [25:21]	Rt [20:16]	Offset [15:0]
16		sw r4, r3(#0)	000100	5'd3	5'd4	16'd0
17		sw r5, r2(#0)	000100	5'd2	5'd5	16'd0
18	no_swap	addi r2, r2, #1	000001	5'd2	5'd2	16'd1
19		addi r3, r3, #1	000001	5'd3	5'd3	16'd1
20		j inner_loop	001000	5'd0	5'd0	16'd7
21		noop	000000	5'd0	5'd0	16'd0
22	next_out	subi r1, r1, #1	001001	5'd1	5'd1	16'd1
23		movi r2, #0	000010	5'd0	5'd2	16'd0
24		movi r3, #1	000010	5'd0	5'd3	16'd1
25		j outer_loop	001000	5'd0	5'd0	16'd5
26		noop	000000	5'd0	5'd0	16'd0
27	end	j end	001000	5'd0	5'd0	16'd27

- Bubble sort expected process

0	1	2	3	4	5	6	7	8	9
323	123	-455	2	98	125	10	65	-56	0
123	323	-455	2	98	125	10	65	-56	0
123	-455	323	2	98	125	10	65	-56	0
123	-455	2	323	98	125	10	65	-56	0
123	-455	2	98	323	125	10	65	-56	0
123	-455	2	98	125	323	10	65	-56	0
123	-455	2	98	125	10	323	65	-56	0
123	-455	2	98	125	10	65	323	-56	0
123	-455	2	98	125	10	65	-56	323	0
123	-455	2	98	125	10	65	-56	0	323
-455	123	2	98	125	10	65	-56	0	323
-455	2	123	98	125	10	65	-56	0	323
-455	2	98	123	125	10	65	-56	0	323
-455	2	98	123	10	125	65	-56	0	323

0	1	2	3	4	5	6	7	8	9
-455	2	98	123	10	65	125	-56	0	323
-455	2	98	123	10	65	-56	125	0	323
-455	2	98	123	10	65	-56	0	125	323
-455	2	98	10	123	65	-56	0	125	323
-455	2	98	10	65	123	-56	0	125	323

3. 5-Stage Pipeline Elements

3.1 IF Stage

3.1.1 PC

- Verilog

```
`timescale 1ns / 1ps

module PC
(
    input clk,
    input rst,
    input [63:0] PC_next,

    output reg [63:0] PC
);

    always @(posedge clk) begin
        if (rst)
            PC <= 64'b0;
        else
            PC <= PC_next;
        end
    endmodule
```

- Testbench

```
`timescale 1ns / 1ps

/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    15:17:32 03/01/2025
// Design Name:    PC
// Module Name:    E:/Documents and Settings/student/EE533_Lab7/PC_tb.v
// Project Name:    EE533_Lab7
// Target Device:
```

```

// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: PC
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////

module PC_tb;

    // Inputs
    reg clk;
    reg rst;
    reg [63:0] PC_next;

    // Outputs
    wire [63:0] PC;

    // Instantiate the Unit Under Test (UUT)
    PC uut (
        .clk(clk),
        .rst(rst),
        .PC_next(PC_next),
        .PC(PC)
    );

    always #50 clk = ~clk;

    initial begin
        // Initialize Inputs
        clk = 1;
        rst = 1;
        PC_next = 0;

        // wait 100 ns for global reset to finish
        #100;
        rst = 0;

        // Add stimulus here
        @(posedge clk);
        PC_next = 64'd1;

        @(posedge clk);
        PC_next = 64'd2;

        @(posedge clk);
        PC_next = 64'd3;

        @(posedge clk);
        PC_next = 64'd4;
    end

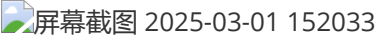
```

```
    @(posedge clk);
    $stop;

end

endmodule
```

- Waveform



3.1.2 PC+1

- Verilog

```
`timescale 1ns / 1ps

module PC_plus_1
(
    input [63:0] PC,
    input [63:0] ONE,

    output [63:0] PC_next
);

    assign PC_next = PC + ONE;

endmodule
```

- Testbench

[illegible]


```

module PC_plus_1_tb;

    // Inputs
    reg [63:0] PC;
    reg [63:0] ONE;

    // Outputs
    wire [63:0] PC_next;

    // Instantiate the Unit Under Test (UUT)
    PC_plus_1 uut (
        .PC(PC),
        .ONE(ONE),
        .PC_next(PC_next)
    );

    initial begin
        // Initialize Inputs
        PC = 0;
        ONE = 1;

        // wait 100 ns for global reset to finish
        #100;

        // Add stimulus here
        #100;
        PC = 1;

        #100;
        PC = 2;

        #100;
        PC = 3;

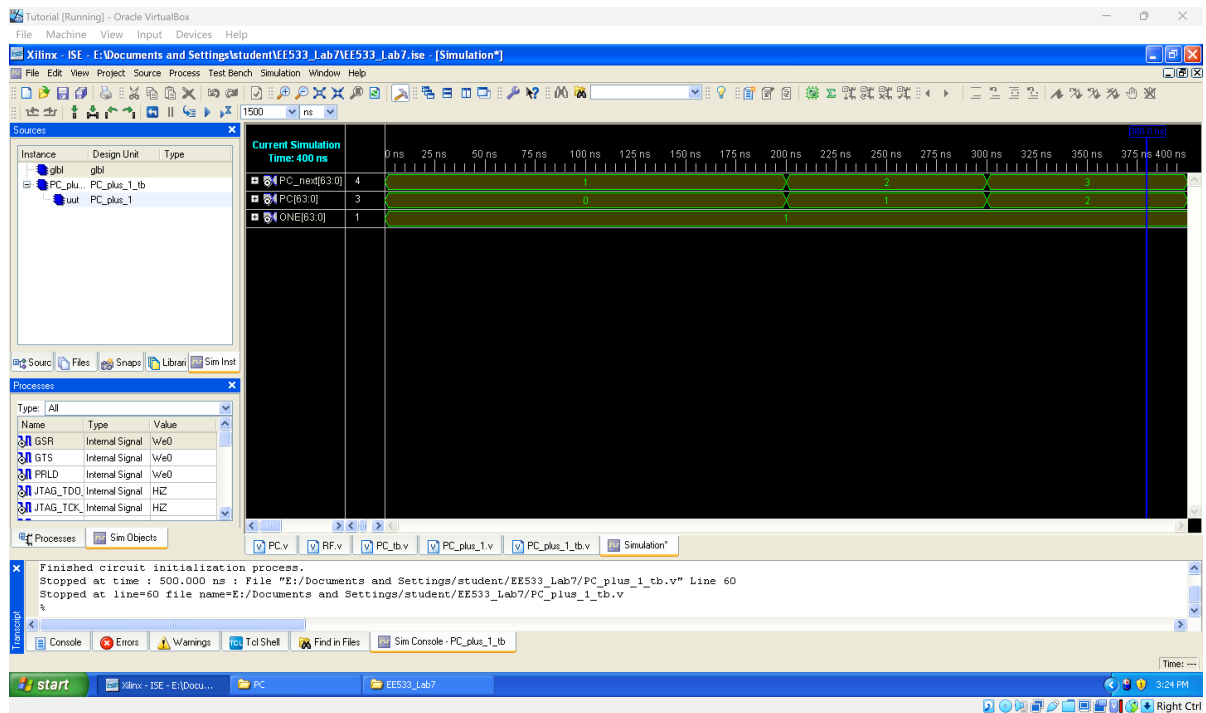
        #100;
        $stop;

    end

endmodule

```

- Waveform



3.1.3 PC_MUX

- Verilog

```
`timescale 1ns / 1ps

module PC_MUX
(
    input [63:0] PC_next_in,
    input [63:0] BTA,
    input PC_ctrl,

    output [63:0] PC_next_out
);

    assign PC_next_out = PC_ctrl == 1? BTA : PC_next_in;

endmodule
```

- Testbench

```
`timescale 1ns / 1ps

/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    15:26:36 03/01/2025
// Design Name:    PC_MUX
// Module Name:    E:/Documents and Settings/student/EE533_Lab7/PC_MUX_tb.v
// Project Name:   EE533_Lab7
// Target Device:
// Tool versions:
// Description:
//
```

```

// Verilog Test Fixture created by ISE for module: PC_MUX
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////

module PC_MUX_tb;

    // Inputs
    reg [63:0] PC_next_in;
    reg [63:0] BTA;
    reg PC_ctrl;

    // Outputs
    wire [63:0] PC_next_out;

    // Instantiate the Unit Under Test (UUT)
    PC_MUX uut (
        .PC_next_in(PC_next_in),
        .BTA(BTA),
        .PC_ctrl(PC_ctrl),
        .PC_next_out(PC_next_out)
    );

    initial begin
        // Initialize Inputs
        PC_next_in = 0;
        BTA = 0;
        PC_ctrl = 0;

        // wait 100 ns for global reset to finish
        #100;

        // Add stimulus here
        #100;
        PC_next_in = 1;
        BTA = 3;
        PC_ctrl = 0;

        #100;
        PC_next_in = 1;
        BTA = 3;
        PC_ctrl = 1;

        #100;
        PC_next_in = 6;
        BTA = 9;
        PC_ctrl = 0;

        #100;
        PC_next_in = 2;
        BTA = 3;
    end

```

```

PC_ctrl = 1;

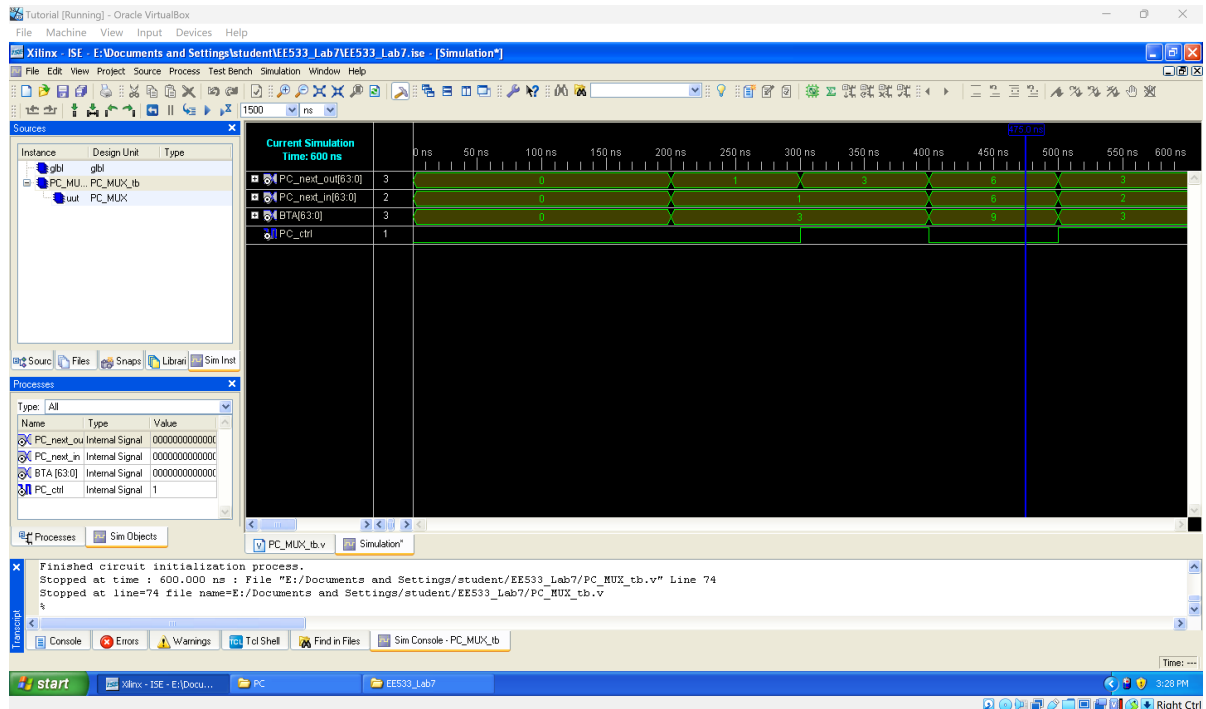
#100;
$stop;

end

endmodule

```

- Waveform



3.1.4 I_MEM

- Verilog
- Vector File

```

memory_initialization_radix=16;
memory_initialization_vector=
08010009,
00000000,
00000000,
14220018,
00000000,
04430001,
00000000,
00000000,
18230015,
00000000,
0C440000,
0C650000,
00000000,
00000000,
1CA40012,
00000000,

```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

- Memory Initialization File

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

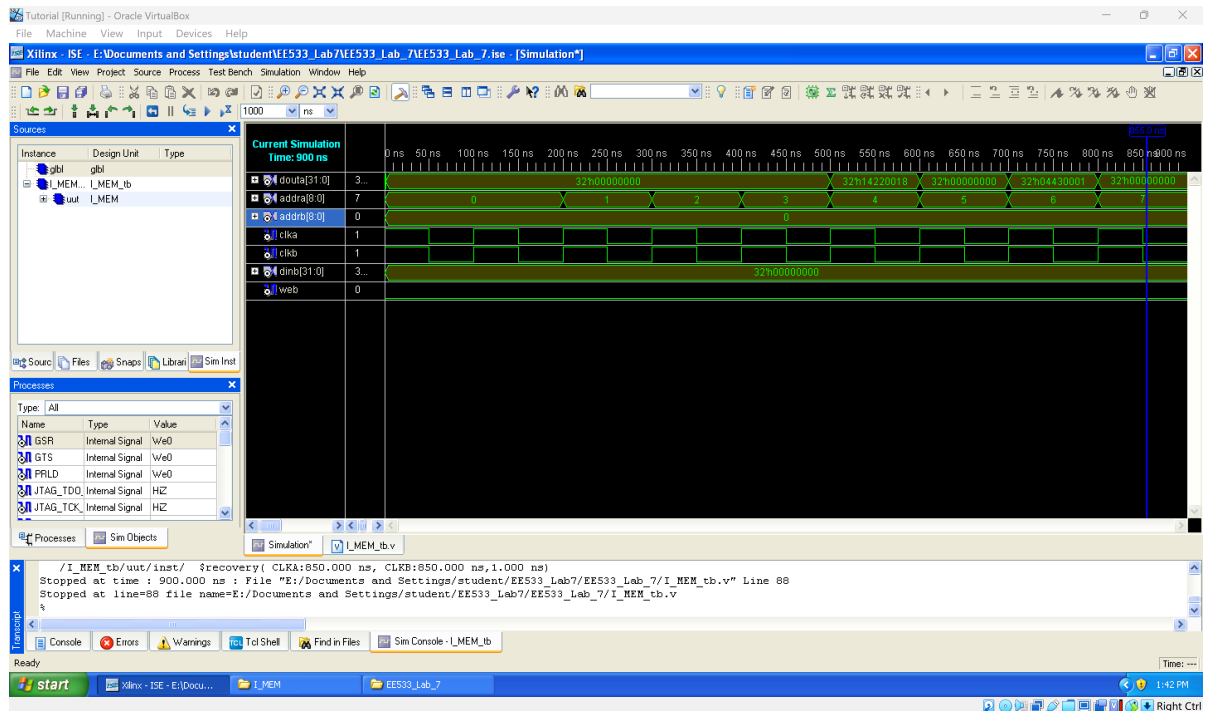
[illegible]

```

00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000

```

- Waveform



3.2 ID Stage

3.2.1 RegFile

- Verilog

```

`timescale 1ns / 1ps

module RF
(
    input clk,
    input rst,
    input wena,
    input [63:0] wdata,
    input [2:0] waddr,
    input [2:0] r0addr,
    input [2:0] r1addr,

    output reg [63:0] r0data,
    output reg [63:0] r1data
);

    reg [63:0] RF [7:0];

    integer i;

    always @(posedge clk) begin
        if (rst == 1)
            begin

```

```

        for (i = 0; i < 8; i = i + 1) begin
            RF[i] <= 64'b0;
        end
    end
    else if (wena == 1)
    begin
        RF[waddr] <= wdata;
    end
end

always @(*) begin
    r0data = ((waddr == r0addr) && wena) ? RF[waddr] : RF[r0addr];
    r1data = ((waddr == r1addr) && wena) ? RF[waddr] : RF[r1addr];
end

endmodule

```

- Testbench

```

`timescale 1ns / 1ps

/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    15:53:18 03/01/2025
// Design Name:    RF
// Module Name:    E:/Documents and Settings/student/EE533_Lab7/RF_tb.v
// Project Name:   EE533_Lab7
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: RF
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
/////////////////////////////////////////////////////////////////

module RF_tb;

    // Inputs
    reg clk;
    reg rst;
    reg wena;
    reg [63:0] wdata;
    reg [2:0] waddr;
    reg [2:0] r0addr;
    reg [2:0] r1addr;

```



```

// Outputs
wire [63:0] r0data;
wire [63:0] r1data;

// Instantiate the Unit Under Test (UUT)
RF uut (
    .clk(clk),
    .rst(rst),
    .wena(wena),
    .wdata(wdata),
    .waddr(waddr),
    .r0addr(r0addr),
    .r1addr(r1addr),
    .r0data(r0data),
    .r1data(r1data)
);

always #50 clk = ~clk;

initial begin
    // Initialize Inputs
    clk = 1;
    rst = 1;
    wena = 0;
    wdata = 0;
    waddr = 0;
    r0addr = 0;
    r1addr = 0;

    // Wait 100 ns for global reset to finish
    #100;
    rst = 0;

    // Add stimulus here
    wena = 1;
    waddr = 3'd1;
    wdata = 64'd17;
    r0addr = 3'b000;
    r1addr = 3'b001;
    #100;

    wena = 1;
    waddr = 3'd2;
    wdata = 64'd85;
    r0addr = 3'd1;
    r1addr = 3'd1;
    #100;

    wena = 0;
    waddr = 3'd2;
    wdata = 64'd17;
    r0addr = 3'd2;
    r1addr = 3'd3;
    #100;

    wena = 1;

```

```

waddr = 3'd3;
wdata = 64'd17;
r0addr = 3'd0;
r1addr = 3'd2;
#100;

wena = 1;
waddr = 3'd4;
wdata = 64'd7;
r0addr = 3'd2;
r1addr = 3'd3;
#100;

wena = 1;
waddr = 3'd5;
wdata = 64'd14;
r0addr = 3'd4;
r1addr = 3'd3;
#100;

wena = 1;
waddr = 3'd6;
wdata = 64'd9;
r0addr = 3'd1;
r1addr = 3'd4;
#100;

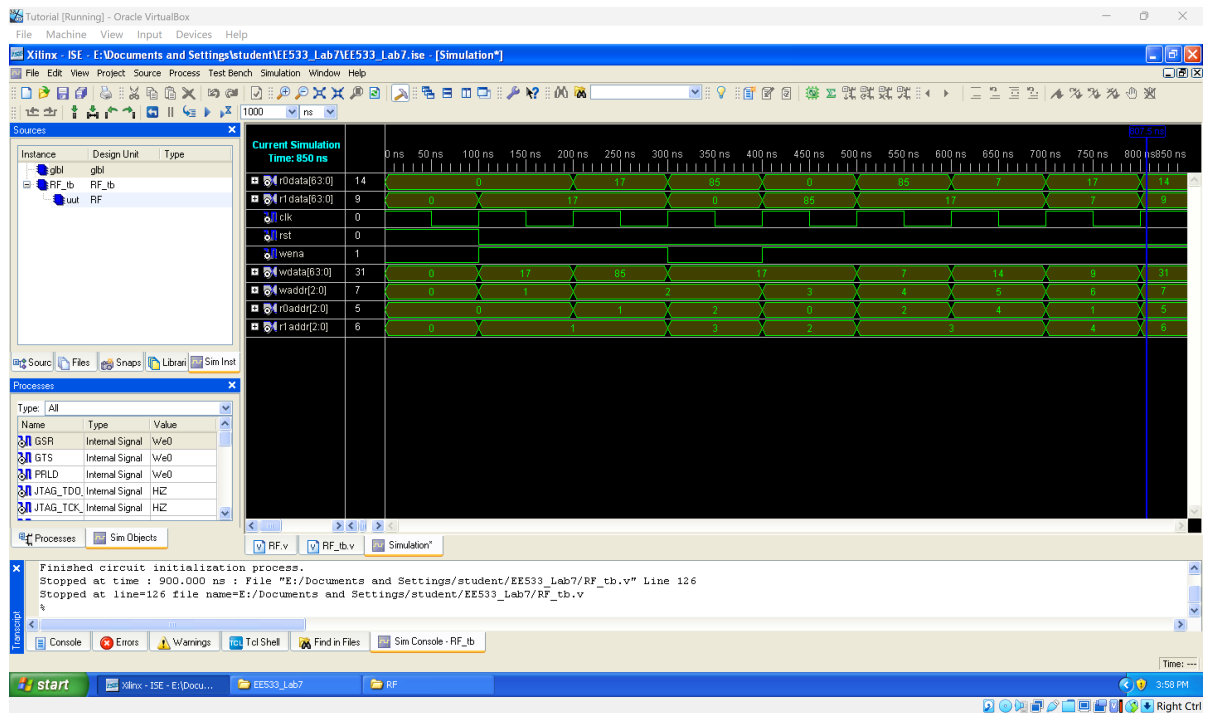
wena = 1;
waddr = 3'd7;
wdata = 64'd31;
r0addr = 3'd5;
r1addr = 3'd6;
#100;

$stop;
end

endmodule

```

- Waveform



3.2.2 Control_Unit

- Verilog

```
`timescale 1ns / 1ps

module Control_Unit
(
    input [5:0] OP_CODE,

    output NOOP_ID,
    output ADDI_ID,
    output MOVI_ID,
    output LW_ID,
    output SW_ID,
    output BEQ_ID,
    output BGT_ID,
    output BLT_ID,
    output J_ID,

    output [3:0] ALU_OP_ID,
    output WME_ID,
    output WRE_ID
);

    assign NOOP_ID = (OP_CODE == 6'd0) ? 1 : 0;
    assign ADDI_ID = (OP_CODE == 6'd1) ? 1 : 0;
    assign MOVI_ID = (OP_CODE == 6'd2) ? 1 : 0;
    assign LW_ID = (OP_CODE == 6'd3) ? 1 : 0;
    assign SW_ID = (OP_CODE == 6'd4) ? 1 : 0;
    assign BEQ_ID = (OP_CODE == 6'd5) ? 1 : 0;
    assign BGT_ID = (OP_CODE == 6'd6) ? 1 : 0;
    assign BLT_ID = (OP_CODE == 6'd7) ? 1 : 0;
    assign J_ID = (OP_CODE == 6'd8) ? 1 : 0;
```

```

    assign ALU_OP_ID = ((OP_CODE == 6'd0) || (OP_CODE == 6'd1) || (OP_CODE ==
6'd3) || (OP_CODE == 6'd4)) ? 4'd0 : 4'd1;
    assign WME_ID = (OP_CODE == 6'd4) ? 1 : 0;
    assign WRE_ID = ((OP_CODE == 6'd1) || (OP_CODE == 6'd2) || (OP_CODE == 6'd3))
? 1 : 0;

endmodule

```

- Testbench

```

`timescale 1ns / 1ps

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    16:16:45 03/01/2025
// Design Name:    Control_Unit
// Module Name:    E:/Documents and Settings/student/EE533_Lab7/Control_Unit_tb.v
// Project Name:   EE533_Lab7
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: Control_Unit
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

module Control_Unit_tb;

    // Inputs
    reg [5:0] OP_CODE;

    // Outputs
    wire NOOP_ID;
    wire ADDI_ID;
    wire MOVI_ID;
    wire LW_ID;
    wire SW_ID;
    wire BEQ_ID;
    wire BGT_ID;
    wire BLT_ID;
    wire J_ID;
    wire [3:0] ALU_OP_ID;
    wire WME_ID;
    wire WRE_ID;

    // Instantiate the Unit Under Test (UUT)
    Control_Unit uut (

```

```

        .OP_CODE(OP_CODE),
        .NOOP_ID(NOOP_ID),
        .ADDI_ID(ADDI_ID),
        .MOVI_ID(MOVI_ID),
        .LW_ID(LW_ID),
        .SW_ID(SW_ID),
        .BEQ_ID(BEQ_ID),
        .BGT_ID(BGT_ID),
        .BLT_ID(BLT_ID),
        .J_ID(J_ID),
        .ALU_OP_ID(ALU_OP_ID),
        .WME_ID(WME_ID),
        .WRE_ID(WRE_ID)
    );

```

```

initial begin
    // Initialize Inputs
    OP_CODE = 0;

    // wait 100 ns for global reset to finish
    #100;

    // Add stimulus here
    #100;
    OP_CODE = 0;

    #100;
    OP_CODE = 1;

    #100;
    OP_CODE = 2;

    #100;
    OP_CODE = 3;

    #100;
    OP_CODE = 4;

    #100;
    OP_CODE = 5;

    #100;
    OP_CODE = 6;

    #100;
    OP_CODE = 7;

    #100;
    OP_CODE = 8;

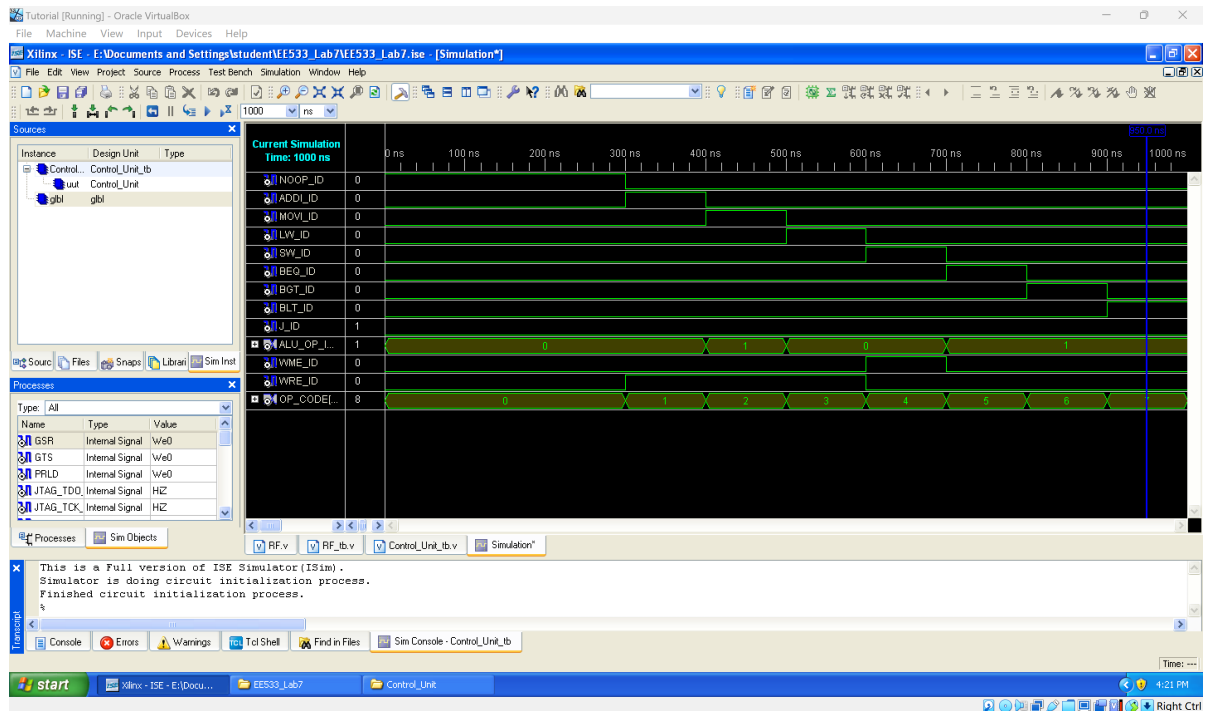
    #100;
    $stop;

end

endmodule

```

- Waveform



3.2.3 Branch_Detection_Unit

- Verilog

```
`timescale 1ns / 1ps

module Branch_Detection_Unit
(
    input [63:0] rs_data,
    input [63:0] rt_data,

    input BEQ_ID,
    input BGT_ID,
    input BLT_ID,
    input J_ID,

    output PC_ctrl
);

    assign PC_ctrl =
        ((BEQ_ID == 1) && (rt_data == rs_data)) ||
        ((BGT_ID == 1) && (rt_data > rs_data)) ||
        ((BLT_ID == 1) && (rt_data < rs_data)) ||
        (J_ID == 1)
        ? 1 : 0;

endmodule
```

- Testbench

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

```

/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    16:29:58 03/01/2025
// Design Name:    Branch_Detection_Unit
// Module Name:    E:/Documents and
Settings/student/EE533_Lab7/Branch_Detection_Unit_tb.v
// Project Name:   EE533_Lab7
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: Branch_Detection_Unit
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
/////////////////////////////////////////////////////////////////

module Branch_Detection_Unit_tb;

    // Inputs
    reg [63:0] rs_data;
    reg [63:0] rt_data;
    reg BEQ_ID;
    reg BGT_ID;
    reg BLT_ID;
    reg J_ID;

    // Outputs
    wire PC_ctrl;

    // Instantiate the Unit Under Test (UUT)
    Branch_Detection_Unit uut (
        .rs_data(rs_data),
        .rt_data(rt_data),
        .BEQ_ID(BEQ_ID),
        .BGT_ID(BGT_ID),
        .BLT_ID(BLT_ID),
        .J_ID(J_ID),
        .PC_ctrl(PC_ctrl)
    );

    initial begin
        // Initialize Inputs
        rt_data = 64'd1;
        rs_data = 64'd0;
        BEQ_ID = 0;
        BGT_ID = 0;
        BLT_ID = 0;
        J_ID = 0;
    end
endmodule

```

```
// wait 100 ns for global reset to finish
#100;

// Add stimulus here
#100;
rt_data = 64'd1;
rs_data = 64'd0;
BEQ_ID = 1;
BGT_ID = 0;
BLT_ID = 0;
J_ID = 0;

#100;
rt_data = 64'd1;
rs_data = 64'd1;
BEQ_ID = 1;
BGT_ID = 0;
BLT_ID = 0;
J_ID = 0;

#100;
rt_data = 64'd1;
rs_data = 64'd2;
BEQ_ID = 1;
BGT_ID = 0;
BLT_ID = 0;
J_ID = 0;

#100;
rt_data = 64'd1;
rs_data = 64'd0;
BEQ_ID = 0;
BGT_ID = 1;
BLT_ID = 0;
J_ID = 0;

#100;
rt_data = 64'd1;
rs_data = 64'd1;
BEQ_ID = 0;
BGT_ID = 1;
BLT_ID = 0;
J_ID = 0;

#100;
rt_data = 64'd1;
rs_data = 64'd2;
BEQ_ID = 0;
BGT_ID = 1;
BLT_ID = 0;
J_ID = 0;

#100;
rt_data = 64'd1;
rs_data = 64'd0;
BEQ_ID = 0;
```



```
BGT_ID = 0;  
BLT_ID = 1;  
J_ID = 0;  
  
#100;  
rt_data = 64'd1;  
rs_data = 64'd1;  
BEQ_ID = 0;  
BGT_ID = 0;  
BLT_ID = 1;  
J_ID = 0;
```

```
#100;  
rt_data = 64'd1;  
rs_data = 64'd2;  
BEQ_ID = 0;  
BGT_ID = 0;  
BLT_ID = 1;  
J_ID = 0;
```

```
#100;  
rt_data = 64'd1;  
rs_data = 64'd0;  
BEQ_ID = 0;  
BGT_ID = 0;  
BLT_ID = 0;  
J_ID = 1;
```

```
#100;  
rt_data = 64'd1;  
rs_data = 64'd1;  
BEQ_ID = 0;  
BGT_ID = 0;  
BLT_ID = 0;  
J_ID = 1;
```

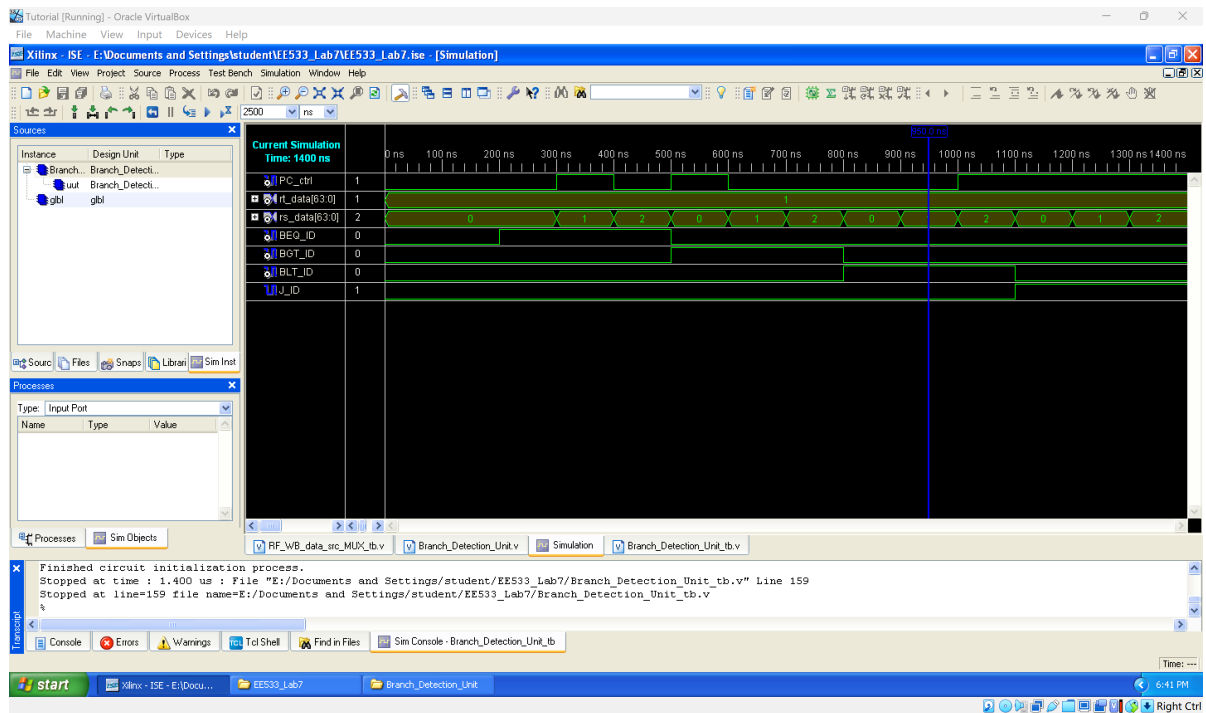
```
#100;  
rt_data = 64'd1;  
rs_data = 64'd2;  
BEQ_ID = 0;  
BGT_ID = 0;  
BLT_ID = 0;  
J_ID = 1;
```

```
#100;  
$stop;
```

```
end
```

```
endmodule
```

- Waveform



3.2.4 Offset_Extend

- Verilog

```
`timescale 1ns / 1ps

module Offset_Extend
(
    input [15:0] offset,
    output [63:0] Offset_ID
);

    assign Offset_ID[15:0] = offset;
    assign Offset_ID[63:16] = 48'b0;

endmodule
```

- Testbench

```
`timescale 1ns / 1ps

/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    16:40:50 03/01/2025
// Design Name:    Offset_Extend
// Module Name:    E:/Documents and Settings/student/EE533_Lab7/Offset_Extend_tb.v
// Project Name:   EE533_Lab7
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: Offset_Extend
```

```

//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////

module Offset_Extend_tb;

    // Inputs
    reg [15:0] Offset;

    // Outputs
    wire [63:0] Offset_ID;

    // Instantiate the Unit Under Test (UUT)
    Offset_Extend uut (
        .Offset(Offset),
        .Offset_ID(Offset_ID)
    );

    initial begin
        // Initialize Inputs
        offset = 0;

        // wait 100 ns for global reset to finish
        #100;

        // Add stimulus here
        #100;
        offset = 16'd0;

        #100;
        offset = 16'd1;

        #100;
        offset = 16'd2;

        #100;
        offset = 16'd3;

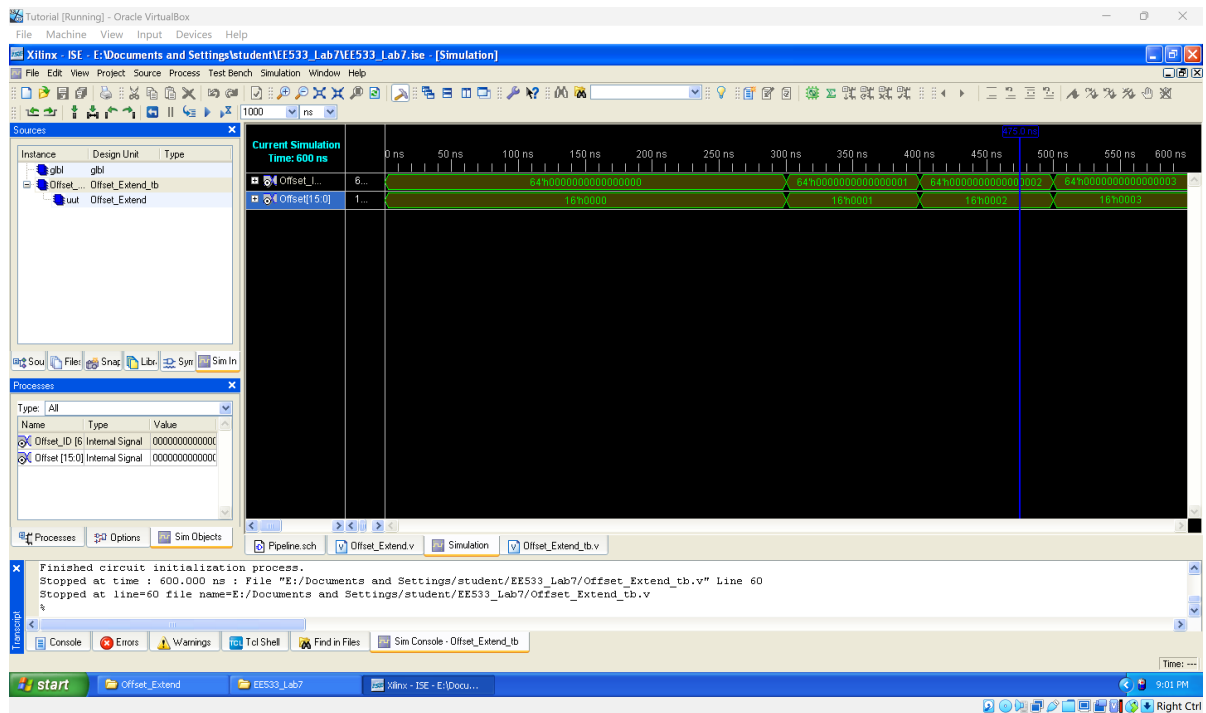
        #100;
        $stop;

    end

endmodule

```

- Waveform



3.3 EX Stage

3.3.1 ALU

- OP_CODE lookup table

aluctrl	Operation	A	B	Expected Output
4'b0000	ADD	1	3	4
4'b0001	SUB	4	2	2
4'b0010	AND	5	7	5
4'b0011	OR	8	3	11
4'b0100	XNOR	13	3	-15
4'b0101	Compare	5	5	1
4'b0110	Left Shift	4	2	16
4'b0111	Right Shift	256	4	16
4'b1000	Substring Compare	15	3	1
4'b1001	Shift-then-Compare	15	3	0

- Verilog

```

`timescale 1ns / 1ps

module ALU (
    input  [63:0] A,
    input  [63:0] B,
    input  [3:0]  ALU_OP,
    output reg [63:0] ALU_Out,

```

```

output reg Zero_Flag,
output reg overflow
);

always @(*) begin
    case (ALU_OP)
        4'b0000: begin // Addition
            {overflow, ALU_Out} = A + B;
        end
        4'b0001: begin // Subtraction
            {overflow, ALU_Out} = A - B;
        end
        4'b0010: ALU_Out = A & B; // Bitwise AND
        4'b0011: ALU_Out = A | B; // Bitwise OR
        4'b0100: ALU_Out = A ^~ B; // Bitwise XNOR
        4'b0101: ALU_Out = (A == B) ? 64'b1 : 64'b0; // Compare (Equality)
        4'b0110: ALU_Out = A << B[5:0]; // Logical Left Shift
        4'b0111: ALU_Out = A >> B[5:0]; // Logical Right Shift
        4'b1000: ALU_Out = substring_match(A, B); // Substring Compare
        4'b1001: ALU_Out = shift_then_compare(A, B); // Shift-then-Compare
        default: ALU_Out = 64'b0;
    endcase

    // Zero Flag
    Zero_Flag = (ALU_Out == 64'b0) ? 1'b1 : 1'b0;

end

// Function to check if B is a substring of A
function [63:0] substring_match;
input [63:0] A, B;
integer i;
begin
    substring_match = 64'b0;
    for (i = 0; i < 64; i = i + 1) begin
        if ((A >> i) & B == B) begin
            substring_match = 64'b1;
        end
    end
end

endfunction

// Function to shift A and then compare with B
function [63:0] shift_then_compare;
input [63:0] A, B;
integer i;
begin
    shift_then_compare = 64'b0;
    for (i = 0; i < 64; i = i + 1) begin
        if ((A >> i) == B) begin
            shift_then_compare = 64'b1;
        end
    end
end

endfunction

endmodule

```

- Testbench

```
`timescale 1ns / 1ps

/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    16:44:30 03/01/2025
// Design Name:    ALU
// Module Name:    E:/Documents and Settings/student/EE533_Lab7/ALU_tb.v
// Project Name:   EE533_Lab7
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: ALU
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
/////////////////////////////////////////////////////////////////

module ALU_tb;

    // Inputs
    reg [63:0] A;
    reg [63:0] B;
    reg [3:0] ALU_OP;

    // Outputs
    wire [63:0] ALU_Out;
    wire Zero_Flag;
    wire Overflow;

    // Instantiate the Unit Under Test (UUT)
    ALU uut (
        .A(A),
        .B(B),
        .ALU_OP(ALU_OP),
        .ALU_Out(ALU_Out),
        .Zero_Flag(Zero_Flag),
        .Overflow(Overflow)
    );

    initial begin
        // Initialize Inputs
        A = 0;
        B = 0;
        ALU_OP = 0;
    end
endmodule
```

```
// wait 100 ns for global reset to finish
#100;

// Add stimulus here
A = 64'd1;
B = 64'd3;
ALU_OP = 4'b0000;
#100;

A = 64'd4;
B = 64'd2;
ALU_OP = 4'b0001;
#100;

A = 64'd5;
B = 64'd7;
ALU_OP = 4'b0010;
#100;

A = 64'd8;
B = 64'd3;
ALU_OP = 4'b0011;
#100;

A = 64'd13;
B = 64'd3;
ALU_OP = 4'b0100;
#100;

A = 64'd5;
B = 64'd5;
ALU_OP = 4'b0101;
#100;

A = 64'd4;
B = 64'd2;
ALU_OP = 4'b0110;
#100;

A = 64'd256;
B = 64'd4;
ALU_OP = 4'b0111;
#100;

A = 64'd15;
B = 64'd3;
ALU_OP = 4'b1000;
#100;

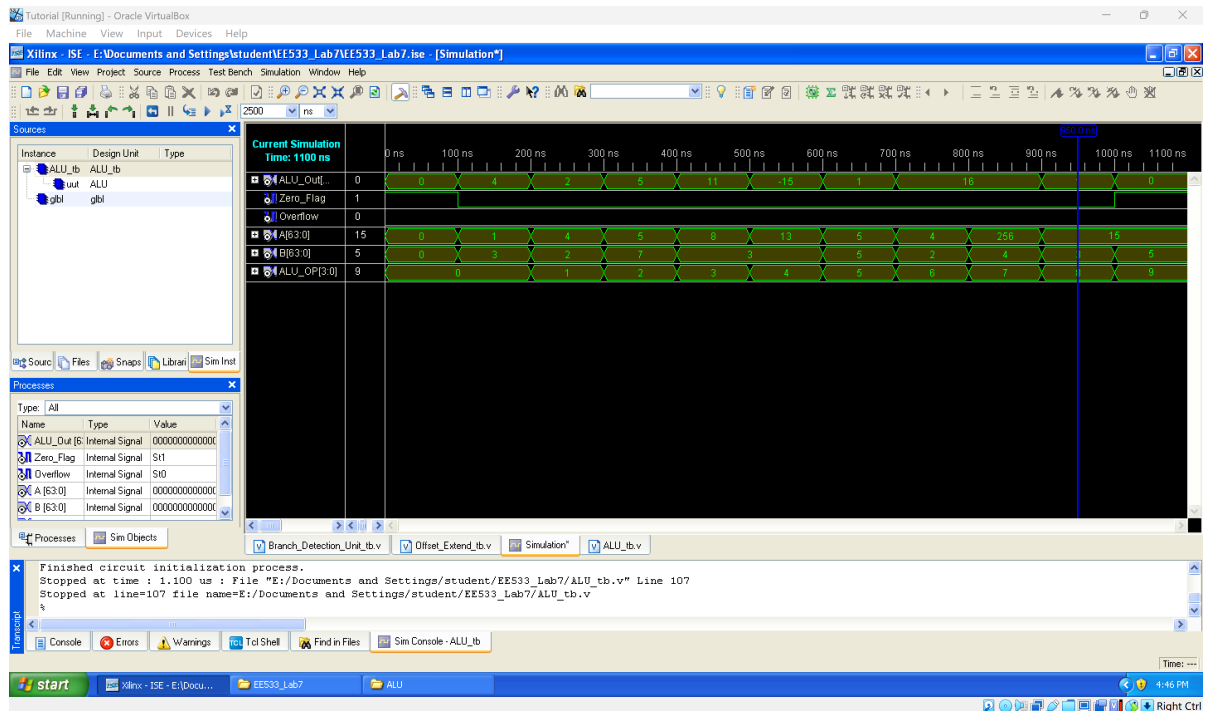
A = 64'd15;
B = 64'd5;
ALU_OP = 4'b1001;
#100;

$stop;
```

end

endmodule

- Waveform



3.3.2 ALU_src_MUX

- Verilog

```
`timescale 1ns / 1ps

module ALU_src_MUX
(
    input [63:0] rt_data,
    input [63:0] offset_EX,
    input ADDI_EX,
    input LW_EX,
    input SW_EX,

    output [63:0] ALU_B
);

    wire ALU_src_ctrl;

    assign ALU_src_ctrl = ADDI_EX || LW_EX || SW_EX;

    assign ALU_B = (ALU_src_ctrl == 1) ? offset_EX : rt_data;

endmodule
```

- Testbench

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



```

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    16:58:15 03/01/2025
// Design Name:    ALU_src_MUX
// Module Name:    E:/Documents and Settings/student/EE533_Lab7/ALU_src_MUX_tb.v
// Project Name:    EE533_Lab7
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: ALU_src_MUX
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

module ALU_src_MUX_tb;

    // Inputs
    reg [63:0] rt_data;
    reg [63:0] offset_EX;
    reg ADDI_EX;
    reg LW_EX;
    reg SW_EX;

    // Outputs
    wire [63:0] ALU_B;

    // Instantiate the Unit Under Test (UUT)
    ALU_src_MUX uut (
        .rt_data(rt_data),
        .offset_EX(offset_EX),
        .ADDI_EX(ADDI_EX),
        .LW_EX(LW_EX),
        .SW_EX(SW_EX),
        .ALU_B(ALU_B)
    );

    initial begin
        // Initialize Inputs
        rt_data = 0;
        offset_EX = 0;
        ADDI_EX = 0;
        LW_EX = 0;
        SW_EX = 0;

        // wait 100 ns for global reset to finish
        #100;

        // Add stimulus here
    end
endmodule

```

```

#100;
rt_data = 64'd1;
offset_EX = 64'd2;
ADDI_EX = 0;
LW_EX = 0;
SW_EX = 0;

#100;
rt_data = 64'd1;
offset_EX = 64'd2;
ADDI_EX = 1;
LW_EX = 0;
SW_EX = 0;

#100;
rt_data = 64'd1;
offset_EX = 64'd2;
ADDI_EX = 0;
LW_EX = 1;
SW_EX = 0;

#100;
rt_data = 64'd1;
offset_EX = 64'd2;
ADDI_EX = 0;
LW_EX = 0;
SW_EX = 1;

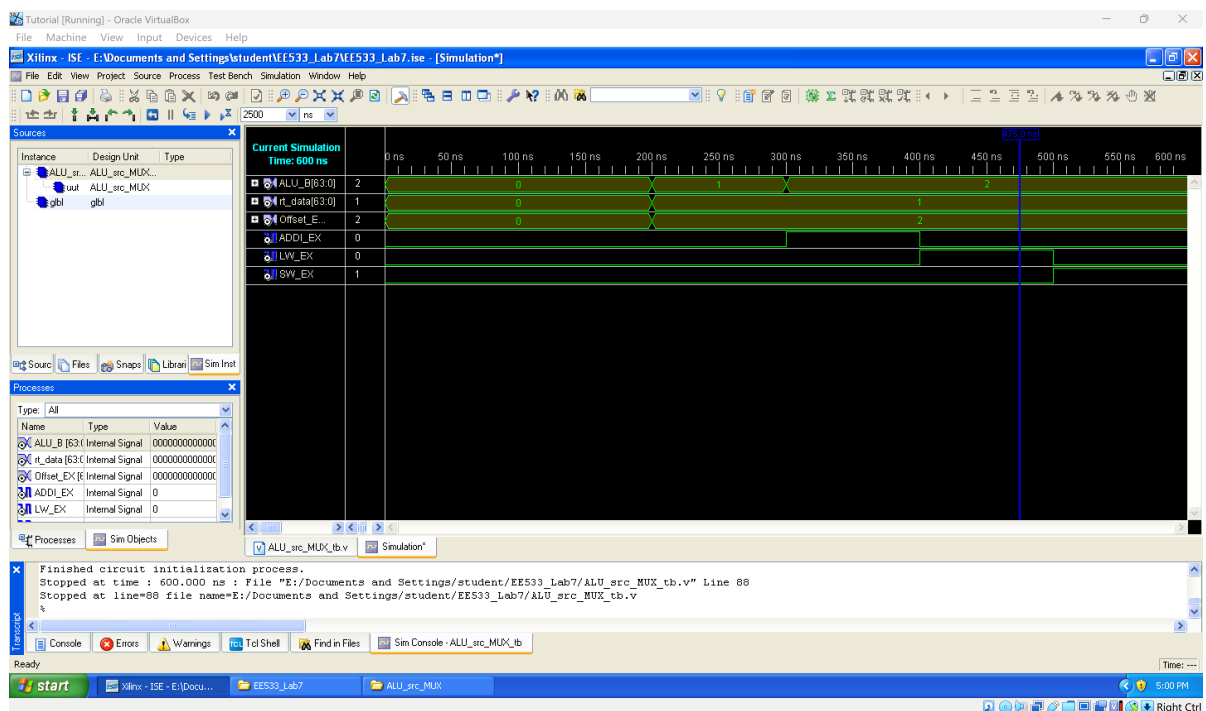
#100;
$stop;

```

end

endmodule

- Waveform



3.4 MEM Stage

3.4.1 D_MEM

- Verilog

```
/******  
*      This file is owned and controlled by Xilinx and must be used      *  
*      solely for design, simulation, implementation and creation of      *  
*      design files limited to Xilinx devices or technologies. Use        *  
*      with non-Xilinx devices or technologies is expressly prohibited    *  
*      and immediately terminates your license.                          *  
*                                                                           *  
*      XILINX IS PROVIDING THIS DESIGN, CODE, OR INFORMATION "AS IS"      *  
*      SOLELY FOR USE IN DEVELOPING PROGRAMS AND SOLUTIONS FOR            *  
*      XILINX DEVICES.  BY PROVIDING THIS DESIGN, CODE, OR INFORMATION    *  
*      AS ONE POSSIBLE IMPLEMENTATION OF THIS FEATURE, APPLICATION        *  
*      OR STANDARD, XILINX IS MAKING NO REPRESENTATION THAT THIS          *  
*      IMPLEMENTATION IS FREE FROM ANY CLAIMS OF INFRINGEMENT,            *  
*      AND YOU ARE RESPONSIBLE FOR OBTAINING ANY RIGHTS YOU MAY REQUIRE     *  
*      FOR YOUR IMPLEMENTATION.  XILINX EXPRESSLY DISCLAIMS ANY           *  
*      WARRANTY WHATSOEVER WITH RESPECT TO THE ADEQUACY OF THE            *  
*      IMPLEMENTATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OR      *  
*      REPRESENTATIONS THAT THIS IMPLEMENTATION IS FREE FROM CLAIMS OF    *  
*      INFRINGEMENT, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS    *  
*      FOR A PARTICULAR PURPOSE.                                          *  
*                                                                           *  
*      Xilinx products are not intended for use in life support           *  
*      appliances, devices, or systems. Use in such applications are      *  
*      expressly prohibited.                                               *  
*                                                                           *  
*      (c) Copyright 1995-2007 Xilinx, Inc.                               *  
*      All rights reserved.                                                *  
*****/  
  
// The synthesis directives "translate_off/translate_on" specified below are  
// supported by Xilinx, Mentor Graphics and Synplify synthesis  
// tools. Ensure they are correct for your synthesis tool(s).  
  
// You must compile the wrapper file D_MEM.v when simulating  
// the core, D_MEM. When compiling the wrapper file, be sure to  
// reference the xilinxCoreLib Verilog simulation library. For detailed  
// instructions, please refer to the "CORE Generator Help".  
  
`timescale 1ns/1ps  
  
module D_MEM(  
    addra,  
    addrb,  
    clka,  
    clkb,  
    dina,  
    doutb,  
    wea);
```

```
input [7 : 0] addra;  
input [7 : 0] addrb;  
input clka;  
input clkb;  
input [63 : 0] dina;  
output [63 : 0] doutb;  
input wea;
```

```
// synthesis translate_off
```

```
BLKMEMDP_V6_3 #(  
    .c_addra_width(8),  
    .c_addrb_width(8),  
    .c_default_data("0"),  
    .c_depth_a(256),  
    .c_depth_b(256),  
    .c_enable_rlocs(0),  
    .c_has_default_data(0),  
    .c_has_dina(1),  
    .c_has_dinb(0),  
    .c_has_douta(0),  
    .c_has_doutb(1),  
    .c_has_ena(0),  
    .c_has_enb(0),  
    .c_has_limit_data_pitch(0),  
    .c_has_nda(0),  
    .c_has_ndb(0),  
    .c_has_rdyb(0),  
    .c_has_rdyb(0),  
    .c_has_rfda(0),  
    .c_has_rfdb(0),  
    .c_has_sinita(0),  
    .c_has_sinitb(0),  
    .c_has_wea(1),  
    .c_has_web(0),  
    .c_limit_data_pitch(18),  
    .c_mem_init_file("D_MEM.mif"),  
    .c_pipe_stages_a(0),  
    .c_pipe_stages_b(0),  
    .c_reg_inputsa(0),  
    .c_reg_inputsb(0),  
    .c_sim_collision_check("NONE"),  
    .c_sinita_value("0"),  
    .c_sinitb_value("0"),  
    .c_width_a(64),  
    .c_width_b(64),  
    .c_write_modea(0),  
    .c_write_modeb(0),  
    .c_ybottom_addr("0"),  
    .cyclka_is_rising(1),  
    .cyclkb_is_rising(1),  
    .c_yena_is_high(1),  
    .c_yenb_is_high(1),  
    .c_yhierarchy("hierarchy1"),  
    .c_ymake_bmm(0),  
    .c_yprimitive_type("16kx1"),
```

```

        .c_ysinita_is_high(1),
        .c_ysinitb_is_high(1),
        .c_ytop_addr("1024"),
        .c_yuse_single_primitive(0),
        .c_ywea_is_high(1),
        .c_yweb_is_high(1),
        .c_yydisable_warnings(1))
inst (
    .ADDRA(addr_a),
    .ADDRB(addr_b),
    .CLKA(clk_a),
    .CLKB(clk_b),
    .DINA(din_a),
    .DOUTB(dout_b),
    .WEA(wea),
    .DINB(),
    .DOUTA(),
    .ENA(),
    .ENB(),
    .NDA(),
    .NDB(),
    .RFDA(),
    .RFDB(),
    .RDYA(),
    .RDYB(),
    .SINITA(),
    .SINITB(),
    .WEB());

// synthesis translate_on

// XST black box declaration
// box_type "black_box"
// synthesis attribute box_type of D_MEM is "black_box"

endmodule

```

- Vector File

```

memory_initialization_radix=16;
memory_initialization_vector=
00000000000000143,
0000000000000007B,
FFFFFFFFFFFFFFE39,
00000000000000002,
00000000000000062,
0000000000000007D,
0000000000000000A,
00000000000000041,
FFFFFFFFFFFFFFFC8,
00000000000000000,
00000000000000000,
00000000000000000,
00000000000000000,

```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

- Memory Initialization File

[illegible]

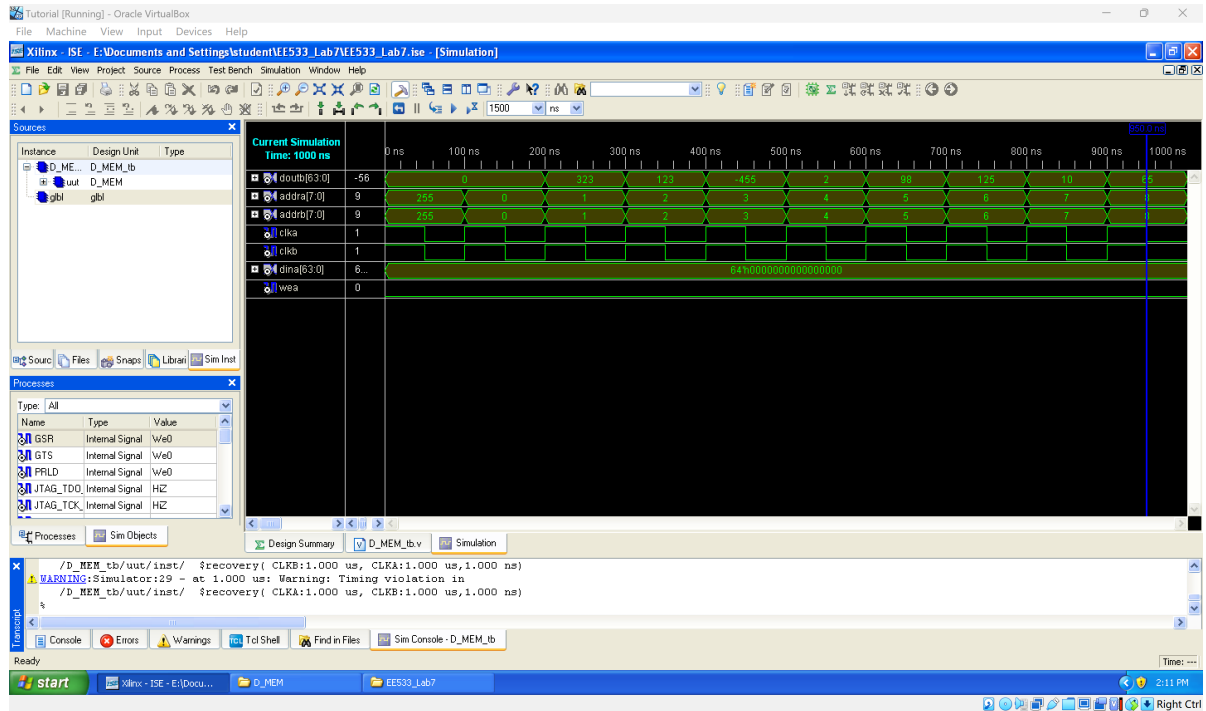
[illegible]

[illegible]

A large rectangular area filled with a dense, repeating pattern of small black squares or dots, resembling a textured background or a low-resolution image.

A large rectangular area filled with many rows of random alphanumeric strings.

- Waveform



3.4.2 D_addr_src_MUX

- Verilog

```

`timescale 1ns / 1ps

module D_addr_src_MUX
(
    input [63:0] ALU_result_M,
    input [4:0] rt_M,
    input SW_M,

    output [7:0] D_addr
);

    assign D_addr[4:0] = (SW_M == 1) ? ALU_result_M[4:0] : rt_M[4:0];
    assign D_addr[7:5] = 0;

endmodule
  
```

- Testbench

```

`timescale 1ns / 1ps

////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 17:09:35 03/01/2025
// Design Name: D_addr_src_MUX
// Module Name: E:/Documents and
Settings/student/EE533_Lab7/D_addr_src_MUX_tb.v
// Project Name: EE533_Lab7
  
```

```

// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: D_addr_src_MUX
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////

module D_addr_src_MUX_tb;

    // Inputs
    reg [63:0] ALU_result_M;
    reg [4:0] rt_M;
    reg SW_M;

    // Outputs
    wire [7:0] D_addr;

    // Instantiate the Unit Under Test (UUT)
    D_addr_src_MUX uut (
        .ALU_result_M(ALU_result_M),
        .rt_M(rt_M),
        .SW_M(SW_M),
        .D_addr(D_addr)
    );

    initial begin
        // Initialize Inputs
        ALU_result_M = 0;
        rt_M = 0;
        SW_M = 0;

        // wait 100 ns for global reset to finish
        #100;

        // Add stimulus here
        #100;
        ALU_result_M = 64'd1;
        rt_M = 5'd2;
        SW_M = 0;

        #100;
        ALU_result_M = 64'd3;
        rt_M = 5'd4;
        SW_M = 1;

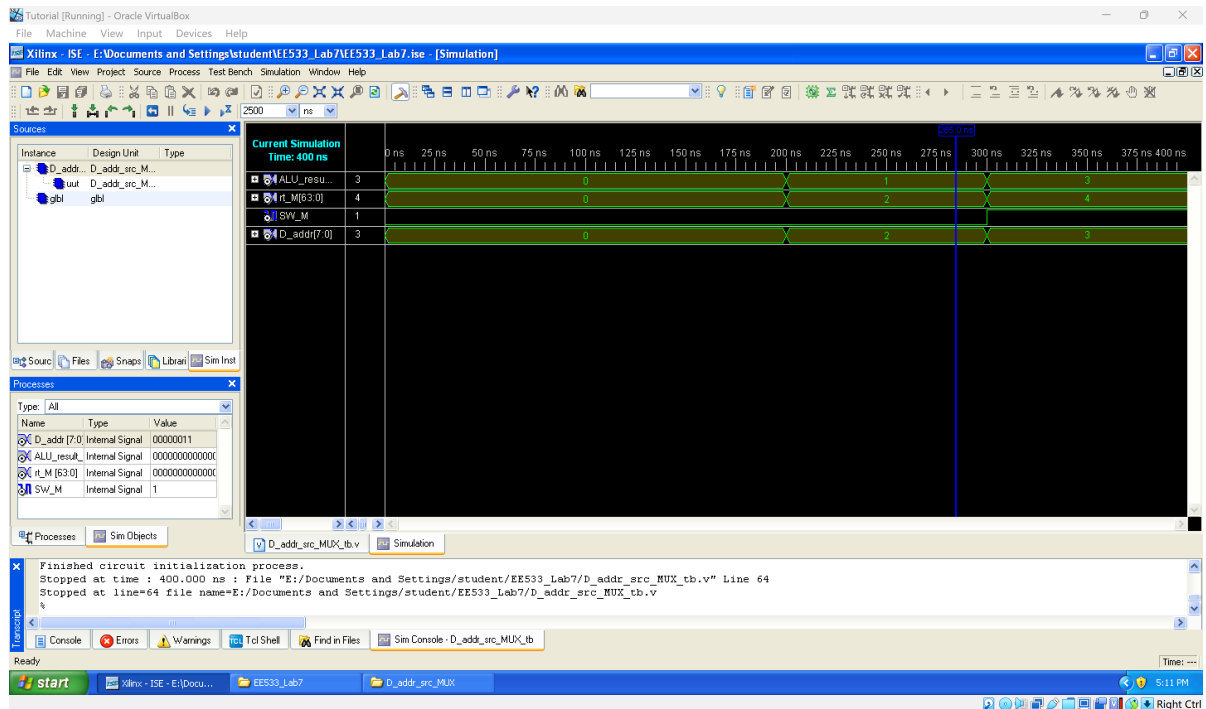
        #100;
        $stop;

    end

```


endmodule

- Waveform



3.5 WB Stage

3.5.1 RF_WB_data_src_MUX

- Verilog

```
`timescale 1ns / 1ps

module RF_WB_data_src_MUX
(
    input [63:0] D_out_WB,
    input [63:0] ALU_out_WB,
    input [63:0] Offset_WB,

    input LW_WB,
    input ADDI_WB,
    input MOVI_WB,

    output [63:0] RF_WB_Din
);

    wire [63:0] temp;

    assign temp = (~LW_WB && ADDI_WB) ? ALU_out_WB : D_out_WB;
    assign RF_WB_Din = (MOVI_WB && ~LW_WB && ~ADDI_WB) ? Offset_WB : temp;

endmodule
```

- Testbench

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

```
/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 17:24:15 03/01/2025
// Design Name: RF_WB_data_src_MUX
// Module Name: E:/Documents and
Settings/student/EE533_Lab7/RF_WB_data_src_MUX_tb.v
// Project Name: EE533_Lab7
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: RF_WB_data_src_MUX
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
/////////////////////////////////////////////////////////////////
```

```
module RF_WB_data_src_MUX_tb;
```

```
    // Inputs
```

```
    reg [63:0] D_out_WB;
    reg [63:0] ALU_out_WB;
    reg [63:0] Offset_WB;
    reg LW_WB;
    reg ADDI_WB;
    reg MOVI_WB;
```

```
    // Outputs
```

```
    wire [63:0] RF_WB_Din;
```

```
    // Instantiate the Unit Under Test (UUT)
```

```
    RF_WB_data_src_MUX uut (
        .D_out_WB(D_out_WB),
        .ALU_out_WB(ALU_out_WB),
        .Offset_WB(Offset_WB),
        .LW_WB(LW_WB),
        .ADDI_WB(ADDI_WB),
        .MOVI_WB(MOVI_WB),
        .RF_WB_Din(RF_WB_Din)
    );
```

```
    initial begin
```

```
        // Initialize Inputs
```

```
        D_out_WB = 0;
        ALU_out_WB = 0;
        Offset_WB = 0;
        LW_WB = 0;
        ADDI_WB = 0;
```

```

    MOVI_WB = 0;

    // wait 100 ns for global reset to finish
    #100;

    // Add stimulus here
    #100;
    D_out_WB = 64'd1;
    ALU_out_WB = 64'd2;
    Offset_WB = 64'd3;
    LW_WB = 0;
    ADDI_WB = 0;
    MOVI_WB = 0;

    #100;
    D_out_WB = 64'd1;
    ALU_out_WB = 64'd2;
    Offset_WB = 64'd3;
    LW_WB = 1;
    ADDI_WB = 0;
    MOVI_WB = 0;

    #100;
    D_out_WB = 64'd1;
    ALU_out_WB = 64'd2;
    Offset_WB = 64'd3;
    LW_WB = 0;
    ADDI_WB = 1;
    MOVI_WB = 0;

    #100;
    D_out_WB = 64'd1;
    ALU_out_WB = 64'd2;
    Offset_WB = 64'd3;
    LW_WB = 0;
    ADDI_WB = 0;
    MOVI_WB = 1;

    #100;
    D_out_WB = 64'd1;
    ALU_out_WB = 64'd2;
    Offset_WB = 64'd3;
    LW_WB = 0;
    ADDI_WB = 0;
    MOVI_WB = 0;

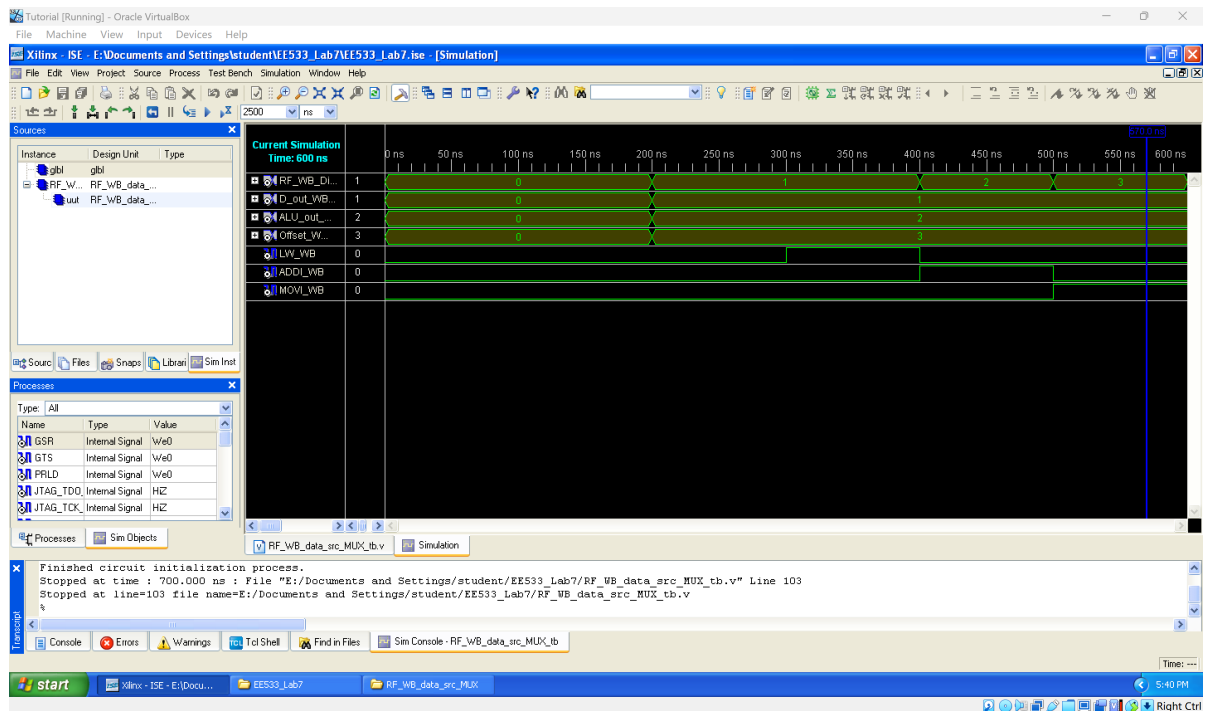
    #100;
    $stop;

end

endmodule

```

- Waveform



3.6 Stage Reg

3.6.1 IF_ID_Reg

- Verilog

```
`timescale 1ns / 1ps

module IF_ID_Reg
(
    input [31:0] Instruction,

    output [5:0] OP_CODE_ID,
    output [4:0] rs_ID,
    output [4:0] rt_ID,
    output [15:0] Offset_ID
);

    assign OP_CODE_ID = Instruction[31:26];
    assign rs_ID = Instruction[25:21];
    assign rt_ID = Instruction[20:16];
    assign Offset_ID = Instruction[15:0];

endmodule
```

- Testbench

```
`timescale 1ns / 1ps

////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    18:48:10 03/01/2025
// Design Name:    IF_ID_Reg
```

```

// Module Name:    E:/Documents and Settings/student/EE533_Lab7/IF_ID_Reg_tb.v
// Project Name:   EE533_Lab7
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: IF_ID_Reg
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////

module IF_ID_Reg_tb;

    // Inputs
    reg [31:0] Instruction;

    // Outputs
    wire [5:0] OP_CODE_ID;
    wire [4:0] rs_ID;
    wire [4:0] rt_ID;
    wire [15:0] Offset_ID;

    // Instantiate the Unit Under Test (UUT)
    IF_ID_Reg uut (
        .Instruction(Instruction),
        .OP_CODE_ID(OP_CODE_ID),
        .rs_ID(rs_ID),
        .rt_ID(rt_ID),
        .Offset_ID(Offset_ID)
    );

    initial begin
        // Initialize Inputs
        Instruction = 0;

        // wait 100 ns for global reset to finish
        #100;

        // Add stimulus here
        #100;
        Instruction = 32'b000010000000000010000000000001001;

        #100;
        Instruction = 32'b0;

        #100;
        Instruction = 32'b0;

        #100;
        Instruction = 32'b000101000010001000000000000011000;
    end

```

```

#100;
Instruction = 32'b0;

#100;
Instruction = 32'b00000100010000110000000000000001;

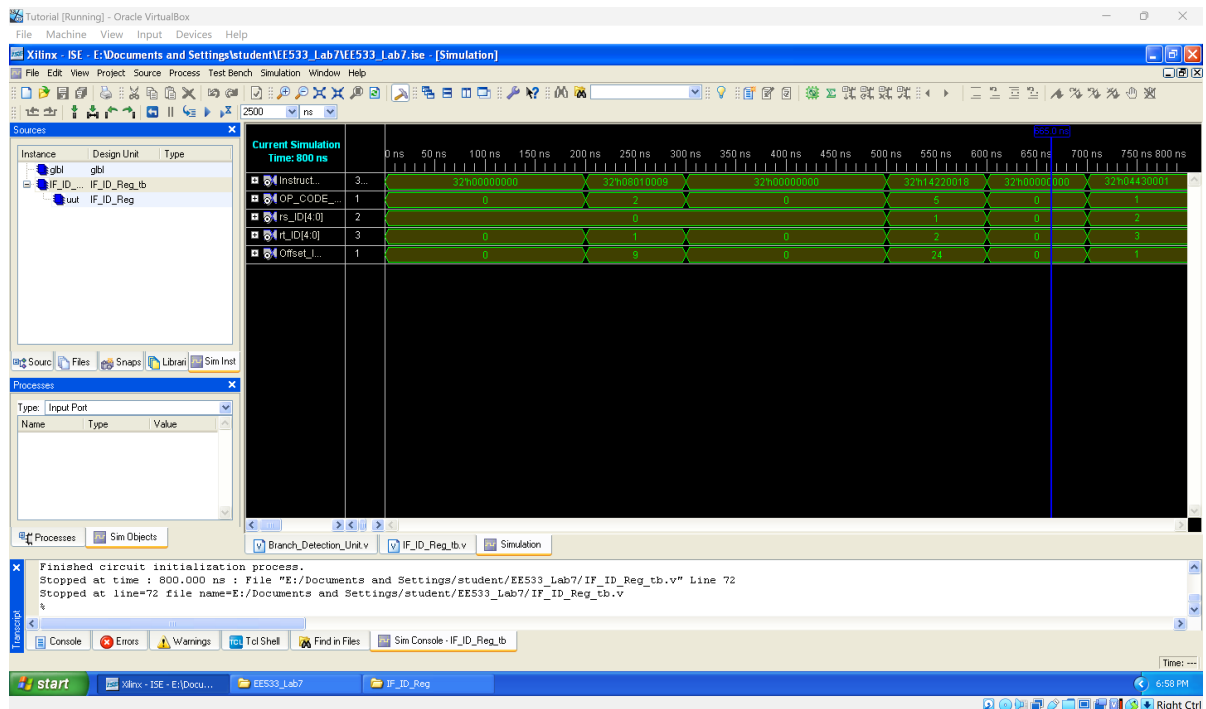
#100;
$stop;

end

endmodule

```

- Waveform



3.6.2 ID_EX_Reg

- Verilog

```

`timescale 1ns / 1ps

module ID_EX_Reg
(
    input clk,
    input rst,

    input [3:0] ALU_OP_ID,
    input NOOP_ID,
    input ADDI_ID,
    input MOVI_ID,
    input LW_ID,
    input SW_ID,

    input WME_ID,
    input WRE_ID,
    input [63:0] rs_data_ID,

```

```

input [63:0] rt_data_ID,
input [4:0] rt_ID,
input [63:0] offset_ID,

output reg [3:0] ALU_OP_EX,
output reg NOOP_EX,
output reg ADDI_EX,
output reg MOVI_EX,
output reg LW_EX,
output reg SW_EX,

output reg WME_EX,
output reg WRE_EX,
output reg [63:0] rs_data_EX,
output reg [63:0] rt_data_EX,
output reg [4:0] rt_EX,
output reg [63:0] Offset_EX
);

always @(posedge clk) begin
    if (rst) begin
        ALU_OP_EX <= 0;
        NOOP_EX <= 0;
        ADDI_EX <= 0;
        MOVI_EX <= 0;
        LW_EX <= 0;
        SW_EX <= 0;
        WME_EX <= 0;
        WRE_EX <= 0;
        rs_data_EX <= 0;
        rt_data_EX <= 0;
        rt_EX <= 0;
        Offset_EX <= 0;
    end
    else begin
        ALU_OP_EX <= ALU_OP_ID;
        NOOP_EX <= NOOP_ID;
        ADDI_EX <= ADDI_ID;
        MOVI_EX <= MOVI_ID;
        LW_EX <= LW_ID;
        SW_EX <= SW_ID;
        WME_EX <= WME_ID;
        WRE_EX <= WRE_ID;
        rs_data_EX <= rs_data_ID;
        rt_data_EX <= rt_data_ID;
        rt_EX <= rt_ID;
        Offset_EX <= Offset_ID;
    end
end

endmodule

```

- Testbench

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

```

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    19:13:54 03/01/2025
// Design Name:    ID_EX_Reg
// Module Name:    E:/Documents and Settings/student/EE533_Lab7/ID_EX_Reg_tb.v
// Project Name:   EE533_Lab7
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: ID_EX_Reg
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

module ID_EX_Reg_tb;

    // Inputs
    reg clk;
    reg rst;
    reg [3:0] ALU_OP_ID;
    reg NOOP_ID;
    reg ADDI_ID;
    reg MOVI_ID;
    reg LW_ID;
    reg SW_ID;
    reg WME_ID;
    reg WRE_ID;
    reg [63:0] rs_data_ID;
    reg [63:0] rt_data_ID;
    reg [4:0] rt_ID;
    reg [63:0] Offset_ID;

    // Outputs
    wire [3:0] ALU_OP_EX;
    wire NOOP_EX;
    wire ADDI_EX;
    wire MOVI_EX;
    wire LW_EX;
    wire SW_EX;
    wire WME_EX;
    wire WRE_EX;
    wire [63:0] rs_data_EX;
    wire [63:0] rt_data_EX;
    wire [4:0] rt_EX;
    wire [63:0] Offset_EX;

    // Instantiate the Unit Under Test (UUT)

```



```

ID_EX_Reg uut (
    .clk(clk),
    .rst(rst),
    .ALU_OP_ID(ALU_OP_ID),
    .NOOP_ID(NOOP_ID),
    .ADDI_ID(ADDI_ID),
    .MOVI_ID(MOVI_ID),
    .LW_ID(LW_ID),
    .SW_ID(SW_ID),
    .WME_ID(WME_ID),
    .WRE_ID(WRE_ID),
    .rs_data_ID(rs_data_ID),
    .rt_data_ID(rt_data_ID),
    .rt_ID(rt_ID),
    .Offset_ID(Offset_ID),
    .ALU_OP_EX(ALU_OP_EX),
    .NOOP_EX(NOOP_EX),
    .ADDI_EX(ADDI_EX),
    .MOVI_EX(MOVI_EX),
    .LW_EX(LW_EX),
    .SW_EX(SW_EX),
    .WME_EX(WME_EX),
    .WRE_EX(WRE_EX),
    .rs_data_EX(rs_data_EX),
    .rt_data_EX(rt_data_EX),
    .rt_EX(rt_EX),
    .Offset_EX(Offset_EX)
);

```

```

always #50 clk = ~clk;

```

```

initial begin
    // Initialize Inputs
    clk = 1;
    rst = 1;
    ALU_OP_ID = 0;
    NOOP_ID = 0;
    ADDI_ID = 0;
    MOVI_ID = 0;
    LW_ID = 0;
    SW_ID = 0;
    WME_ID = 0;
    WRE_ID = 0;
    rs_data_ID = 0;
    rt_data_ID = 0;
    rt_ID = 0;
    Offset_ID = 0;

    // wait 100 ns for global reset to finish
    #100;
    rst = 0;

    // Add stimulus here
    @(posedge clk);
    ALU_OP_ID = 4'd0;
    NOOP_ID = 0;

```

```
ADDI_ID = 0;
MOVI_ID = 1;
LW_ID = 0;
SW_ID = 0;
WME_ID = 0;
WRE_ID = 1;
rs_data_ID = 0;
rt_data_ID = 0;
rt_ID = 5'd1;
Offset_ID = 16'd9;
```

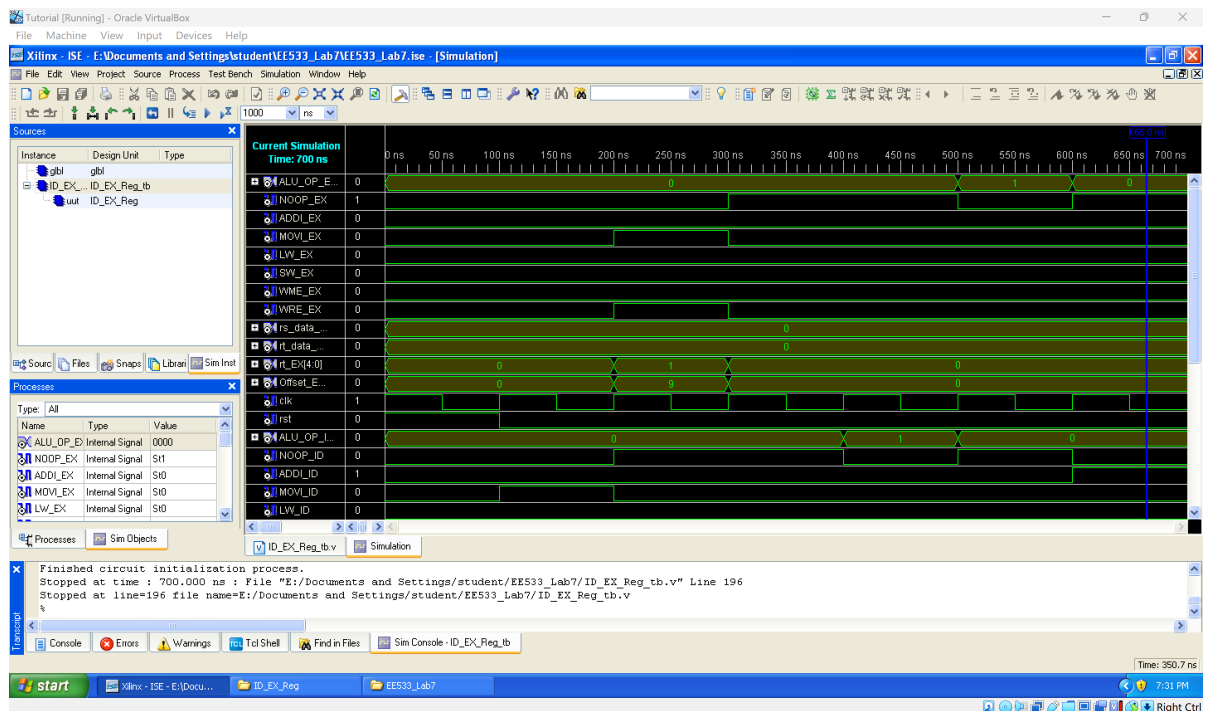
```
@(posedge clk);
ALU_OP_ID = 4'd0;
NOOP_ID = 1;
ADDI_ID = 0;
MOVI_ID = 0;
LW_ID = 0;
SW_ID = 0;
WME_ID = 0;
WRE_ID = 0;
rs_data_ID = 0;
rt_data_ID = 0;
rt_ID = 5'd0;
Offset_ID = 16'd0;
```

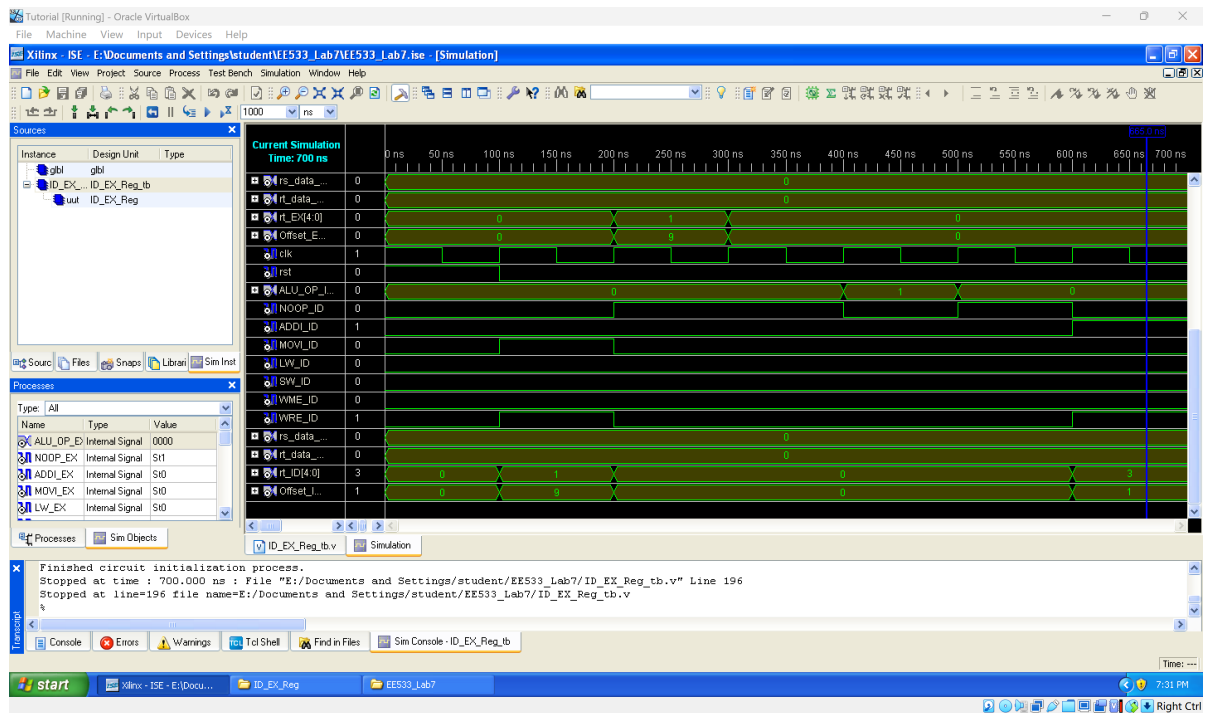
```
@(posedge clk);
ALU_OP_ID = 4'd0;
NOOP_ID = 1;
ADDI_ID = 0;
MOVI_ID = 0;
LW_ID = 0;
SW_ID = 0;
WME_ID = 0;
WRE_ID = 0;
rs_data_ID = 0;
rt_data_ID = 0;
rt_ID = 5'd0;
Offset_ID = 16'd0;
```

```
@(posedge clk);
ALU_OP_ID = 4'd1;
NOOP_ID = 0;
ADDI_ID = 0;
MOVI_ID = 0;
LW_ID = 0;
SW_ID = 0;
WME_ID = 0;
WRE_ID = 0;
rs_data_ID = 0;
rt_data_ID = 0;
rt_ID = 5'd0;
Offset_ID = 16'd0;
```

```
@(posedge clk);
ALU_OP_ID = 4'd0;
NOOP_ID = 1;
```

endmodule





3.6.3 EX_M_Reg

- Verilog

```
`timescale 1ns / 1ps

module EX_M_Reg
(
    input clk,
    input rst,

    input NOOP_EX,
    input ADDI_EX,
    input MOVI_EX,
    input LW_EX,
    input SW_EX,

    input WME_EX,
    input WRE_EX,
    input [63:0] ALU_result_EX,
    input [63:0] rs_data_EX,
    input [63:0] rt_data_EX,
    input [4:0] rt_EX,
    input [63:0] Offset_EX,

    output reg NOOP_M,
    output reg ADDI_M,
    output reg MOVI_M,
    output reg LW_M,
    output reg SW_M,

    output reg WME_M,
    output reg WRE_M,
    output reg [63:0] ALU_result_M,
    output reg [63:0] rs_data_M,
    output reg [63:0] rt_data_M,
```

```

output reg [4:0] rt_M,
output reg [63:0] Offset_M
);

always @(posedge clk) begin
    if (rst) begin
        NOOP_M <= 0;
        ADDI_M <= 0;
        MOVI_M <= 0;
        LW_M <= 0;
        SW_M <= 0;

        WME_M <= 0;
        WRE_M <= 0;
        ALU_result_M <= 0;
        rs_data_M <= 0;
        rt_data_M <= 0;
        rt_M <= 0;
        Offset_M <= 0;
    end
    else begin
        NOOP_M <= NOOP_EX;
        ADDI_M <= ADDI_EX;
        MOVI_M <= MOVI_EX;
        LW_M <= LW_EX;
        SW_M <= SW_EX;

        WME_M <= WME_EX;
        WRE_M <= WRE_EX;
        ALU_result_M <= ALU_result_EX;
        rs_data_M <= rs_data_EX;
        rt_data_M <= rt_data_EX;
        rt_M <= rt_EX;
        Offset_M <= Offset_EX;
    end
end

endmodule

```

- Testbench

```

`timescale 1ns / 1ps

/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    19:34:10 03/01/2025
// Design Name:    EX_M_Reg
// Module Name:    E:/Documents and Settings/student/EE533_Lab7/EX_M_Reg_tb.v
// Project Name:   EE533_Lab7
// Target Device:
// Tool versions:
// Description:
//

```

```

// Verilog Test Fixture created by ISE for module: EX_M_Reg
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////

module EX_M_Reg_tb;

    // Inputs
    reg clk;
    reg rst;
    reg NOOP_EX;
    reg ADDI_EX;
    reg MOVI_EX;
    reg LW_EX;
    reg SW_EX;
    reg WME_EX;
    reg WRE_EX;
    reg [63:0] ALU_result_EX;
    reg [63:0] rs_data_EX;
    reg [63:0] rt_data_EX;
    reg [4:0] rt_EX;
    reg [63:0] offset_EX;

    // Outputs
    wire NOOP_M;
    wire ADDI_M;
    wire MOVI_M;
    wire LW_M;
    wire SW_M;
    wire WME_M;
    wire WRE_M;
    wire [63:0] ALU_result_M;
    wire [63:0] rs_data_M;
    wire [63:0] rt_data_M;
    wire [4:0] rt_M;
    wire [63:0] offset_M;

    // Instantiate the Unit Under Test (UUT)
    EX_M_Reg uut (
        .clk(clk),
        .rst(rst),
        .NOOP_EX(NOOP_EX),
        .ADDI_EX(ADDI_EX),
        .MOVI_EX(MOVI_EX),
        .LW_EX(LW_EX),
        .SW_EX(SW_EX),
        .WME_EX(WME_EX),
        .WRE_EX(WRE_EX),
        .ALU_result_EX(ALU_result_EX),
        .rs_data_EX(rs_data_EX),
        .rt_data_EX(rt_data_EX),

```

```

        .rt_EX(rt_EX),
        .Offset_EX(Offset_EX),
        .NOOP_M(NOOP_M),
        .ADDI_M(ADDI_M),
        .MOVI_M(MOVI_M),
        .LW_M(LW_M),
        .SW_M(SW_M),
        .WME_M(WME_M),
        .WRE_M(WRE_M),
        .ALU_result_M(ALU_result_M),
        .rs_data_M(rs_data_M),
        .rt_data_M(rt_data_M),
        .rt_M(rt_M),
        .Offset_M(Offset_M)
    );

```

```

always #50 clk = ~clk;

```

```

initial begin

```

```

    // Initialize Inputs

```

```

    clk = 1;
    rst = 1;
    NOOP_EX = 0;
    ADDI_EX = 0;
    MOVI_EX = 0;
    LW_EX = 0;
    SW_EX = 0;
    WME_EX = 0;
    WRE_EX = 0;
    ALU_result_EX = 0;
    rs_data_EX = 0;
    rt_data_EX = 0;
    rt_EX = 0;
    Offset_EX = 0;

```

```

    // wait 100 ns for global reset to finish

```

```

    @(posedge clk);
    rst = 0;

```

```

    // Add stimulus here

```

```

    @(posedge clk);
    NOOP_EX = 0;
    ADDI_EX = 0;
    MOVI_EX = 1;
    LW_EX = 0;
    SW_EX = 0;
    WME_EX = 0;
    WRE_EX = 1;
    ALU_result_EX = 0;
    rs_data_EX = 0;
    rt_data_EX = 0;
    rt_EX = 5'd1;
    Offset_EX = 16'd9;

```

```

    @(posedge clk);
    NOOP_EX = 1;

```

```
ADDI_EX = 0;
MOVI_EX = 0;
LW_EX = 0;
SW_EX = 0;
WME_EX = 0;
WRE_EX = 0;
ALU_result_EX = 0;
rs_data_EX = 0;
rt_data_EX = 0;
rt_EX = 5'd0;
offset_EX = 16'd0;
```

```
@(posedge clk);
NOOP_EX = 1;
ADDI_EX = 0;
MOVI_EX = 0;
LW_EX = 0;
SW_EX = 0;
WME_EX = 0;
WRE_EX = 0;
ALU_result_EX = 0;
rs_data_EX = 0;
rt_data_EX = 0;
rt_EX = 5'd0;
Offset_EX = 16'd0;
```

```
@(posedge clk);
NOOP_EX = 0;
ADDI_EX = 0;
MOVI_EX = 0;
LW_EX = 0;
SW_EX = 0;
WME_EX = 0;
WRE_EX = 0;
ALU_result_EX = 0;
rs_data_EX = 0;
rt_data_EX = 0;
rt_EX = 5'd2;
offset_EX = 16'd24;
```

```
@(posedge clk);
NOOP_EX = 1;
ADDI_EX = 0;
MOVI_EX = 0;
LW_EX = 0;
SW_EX = 0;
WME_EX = 0;
WRE_EX = 0;
ALU_result_EX = 0;
rs_data_EX = 0;
rt_data_EX = 0;
rt_EX = 5'd0;
offset_EX = 16'd0;
```

```
@(posedge clk);
NOOP_EX = 0;
```



```

ADDI_EX = 1;
MOVI_EX = 0;
LW_EX = 0;
SW_EX = 0;
WME_EX = 0;
WRE_EX = 0;
ALU_result_EX = 64'd1;
rs_data_EX = 64'd9;
rt_data_EX = 0;
rt_EX = 5'd3;
offset_EX = 16'd0;

```

```
@(posedge clk);
```

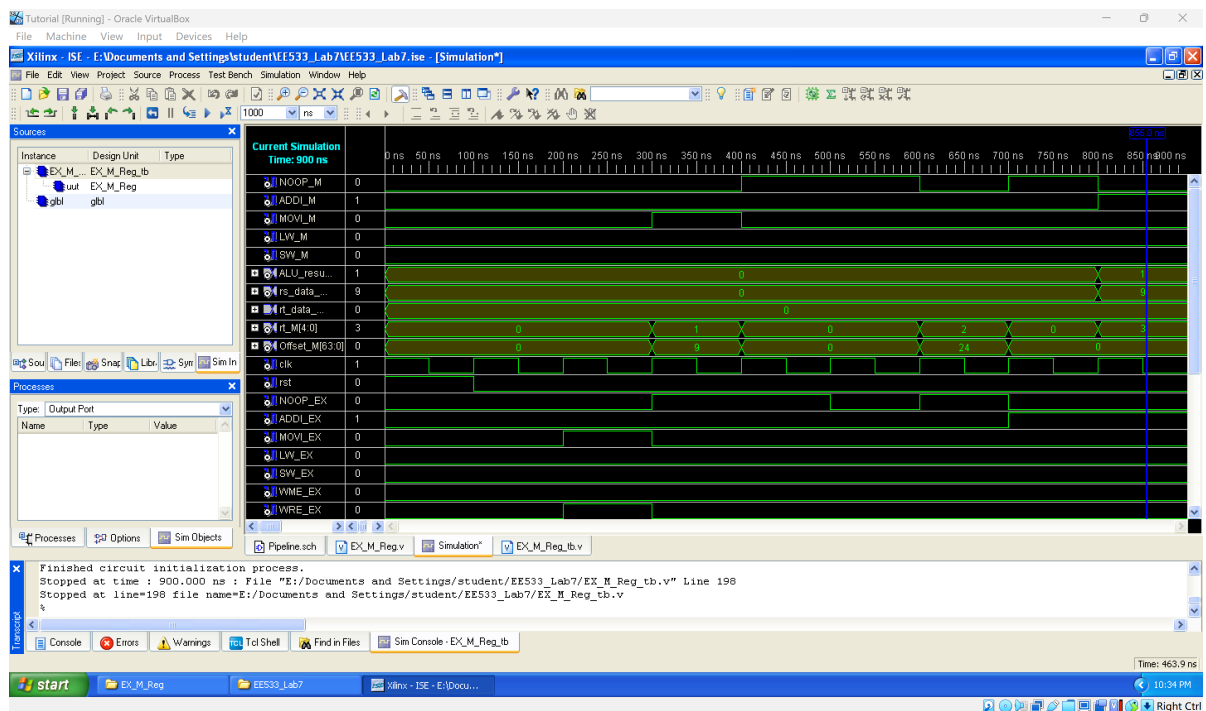
```
@(posedge clk);
```

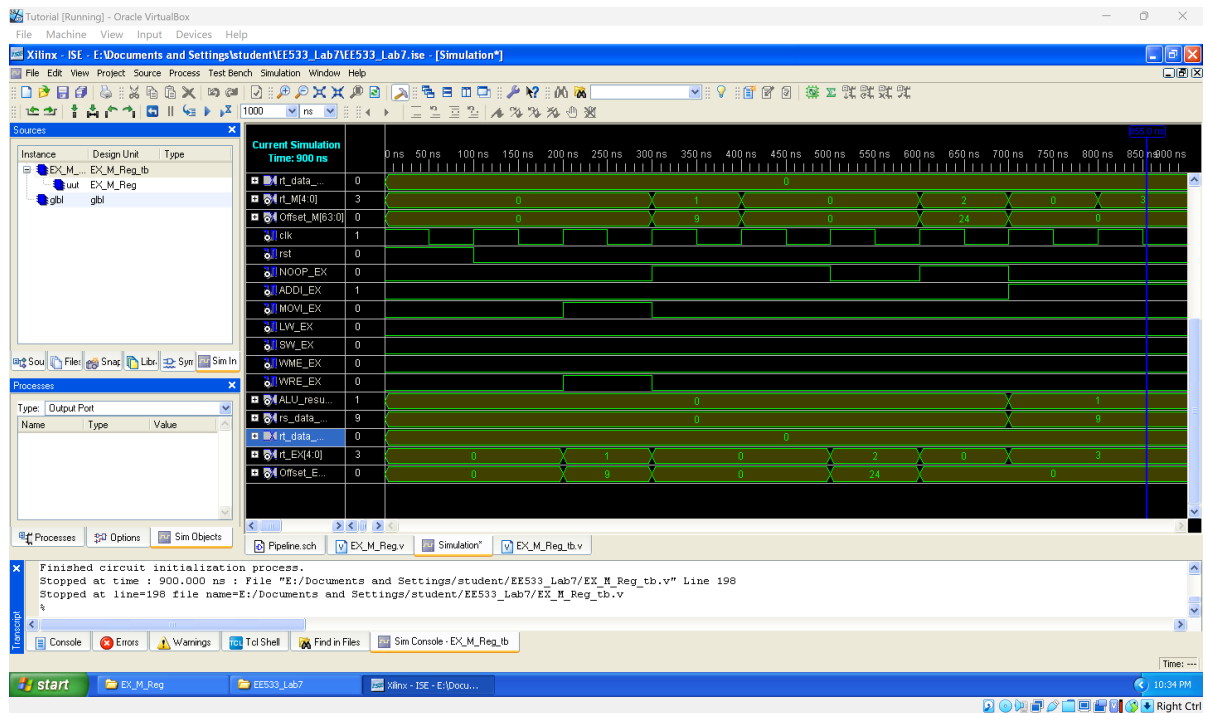
```
$stop;
```

```
end
```

```
endmodule
```

- Waveform





3.6.4 M_WB_Reg

- Verilog

```
`timescale 1ns / 1ps

module M_WB_Reg
(
    input clk,
    input rst,

    input NOOP_M,
    input ADDI_M,
    input MOVI_M,
    input LW_M,
    input SW_M,

    input WRE_M,
    input [63:0] D_out_M,
    input [63:0] rs_data_M,
    input [63:0] ALU_result_M,
    input [63:0] offset_M,
    input [4:0] rt_M,

    output reg NOOP_WB,
    output reg ADDI_WB,
    output reg MOVI_WB,
    output reg LW_WB,
    output reg SW_WB,

    output reg WRE_WB,
    output reg [63:0] D_out_WB,
    output reg [63:0] rs_data_WB,
    output reg [63:0] ALU_result_WB,
    output reg [63:0] offset_WB,
    output reg [2:0] rt_WB
)
```

```

);

always @(posedge clk) begin
    if (rst) begin
        NOOP_WB <= 0;
        ADDI_WB <= 0;
        MOVI_WB <= 0;
        LW_WB <= 0;
        SW_WB <= 0;
        WRE_WB <= 0;
        D_out_WB <= 0;
        rs_data_WB <= 0;
        ALU_result_WB <= 0;
        Offset_WB <= 0;
        rt_WB <= 0;
    end
    else begin
        NOOP_WB <= NOOP_M;
        ADDI_WB <= ADDI_M;
        MOVI_WB <= MOVI_M;
        LW_WB <= LW_M;
        SW_WB <= SW_M;
        WRE_WB <= WRE_M;
        D_out_WB <= D_out_M;
        rs_data_WB <= rs_data_M;
        ALU_result_WB <= ALU_result_M;
        Offset_WB <= Offset_M;
        rt_WB <= rt_M[2:0];
    end
end

endmodule

```

- Testbench

```

`timescale 1ns / 1ps

/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    19:46:24 03/01/2025
// Design Name:    M_WB_Reg
// Module Name:    E:/Documents and Settings/student/EE533_Lab7/M_WB_Reg_tb.v
// Project Name:   EE533_Lab7
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: M_WB_Reg
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created

```

```

// Additional Comments:
//
////////////////////////////////////

module M_WB_Reg_tb;

    // Inputs
    reg clk;
    reg rst;
    reg NOOP_M;
    reg ADDI_M;
    reg MOVI_M;
    reg LW_M;
    reg SW_M;
    reg WRE_M;
    reg [63:0] D_out_M;
    reg [63:0] rs_data_M;
    reg [63:0] ALU_result_M;
    reg [63:0] Offset_M;
    reg [4:0] rt_M;

    // Outputs
    wire NOOP_WB;
    wire ADDI_WB;
    wire MOVI_WB;
    wire LW_WB;
    wire SW_WB;
    wire WRE_WB;
    wire [63:0] D_out_WB;
    wire [63:0] rs_data_WB;
    wire [63:0] ALU_result_WB;
    wire [63:0] Offset_WB;
    wire [2:0] rt_WB;

    // Instantiate the Unit Under Test (UUT)
    M_WB_Reg uut (
        .clk(clk),
        .rst(rst),
        .NOOP_M(NOOP_M),
        .ADDI_M(ADDI_M),
        .MOVI_M(MOVI_M),
        .LW_M(LW_M),
        .SW_M(SW_M),
        .WRE_M(WRE_M),
        .D_out_M(D_out_M),
        .rs_data_M(rs_data_M),
        .ALU_result_M(ALU_result_M),
        .Offset_M(Offset_M),
        .rt_M(rt_M),
        .NOOP_WB(NOOP_WB),
        .ADDI_WB(ADDI_WB),
        .MOVI_WB(MOVI_WB),
        .LW_WB(LW_WB),
        .SW_WB(SW_WB),
        .WRE_WB(WRE_WB),
        .D_out_WB(D_out_WB),

```

```

        .rs_data_WB(rs_data_WB),
        .ALU_result_WB(ALU_result_WB),
        .Offset_WB(Offset_WB),
        .rt_WB(rt_WB)
    );

always #50 clk = ~clk;

initial begin
    // Initialize Inputs
    clk = 1;
    rst = 1;
    NOOP_M = 0;
    ADDI_M = 0;
    MOVI_M = 0;
    LW_M = 0;
    SW_M = 0;
    WRE_M = 0;
    D_out_M = 0;
    rs_data_M = 0;
    ALU_result_M = 0;
    Offset_M = 0;
    rt_M = 0;

    // wait 100 ns for global reset to finish
    @(posedge clk);
    rst = 0;

    // Add stimulus here
    @(posedge clk);
    NOOP_M = 0;
    ADDI_M = 0;
    MOVI_M = 1;
    LW_M = 0;
    SW_M = 0;
    WRE_M = 1;
    D_out_M = 64'd123;
    rs_data_M = 64'd0;
    ALU_result_M = 0;
    Offset_M = 64'd9;
    rt_M = 5'd1;

    @(posedge clk);
    NOOP_M = 1;
    ADDI_M = 0;
    MOVI_M = 0;
    LW_M = 0;
    SW_M = 0;
    WRE_M = 0;
    D_out_M = 64'd323;
    rs_data_M = 64'd0;
    ALU_result_M = 0;
    Offset_M = 64'd0;
    rt_M = 5'd0;

    @(posedge clk);

```

```
NOOP_M = 1;
ADDI_M = 0;
MOVI_M = 0;
LW_M = 0;
SW_M = 0;
WRE_M = 0;
D_out_M = 64'd323;
rs_data_M = 64'd0;
ALU_result_M = 0;
offset_M = 64'd0;
rt_M = 5'd0;
```

```
@(posedge clk);
NOOP_M = 0;
ADDI_M = 0;
MOVI_M = 0;
LW_M = 0;
SW_M = 0;
WRE_M = 0;
D_out_M = 64'd323;
rs_data_M = 64'd0;
ALU_result_M = 0;
offset_M = 64'd0;
rt_M = 5'd0;
```

```
@(posedge clk);
NOOP_M = 0;
ADDI_M = 0;
MOVI_M = 0;
LW_M = 0;
SW_M = 0;
WRE_M = 0;
D_out_M = 64'd323;
rs_data_M = 64'd0;
ALU_result_M = 0;
offset_M = 64'd0;
rt_M = 5'd0;
```

```
@(posedge clk);
NOOP_M = 0;
ADDI_M = 1;
MOVI_M = 0;
LW_M = 0;
SW_M = 0;
WRE_M = 1;
D_out_M = 64'd2;
rs_data_M = 64'd0;
ALU_result_M = 64'd1;
offset_M = 64'd1;
rt_M = 5'd3;
```

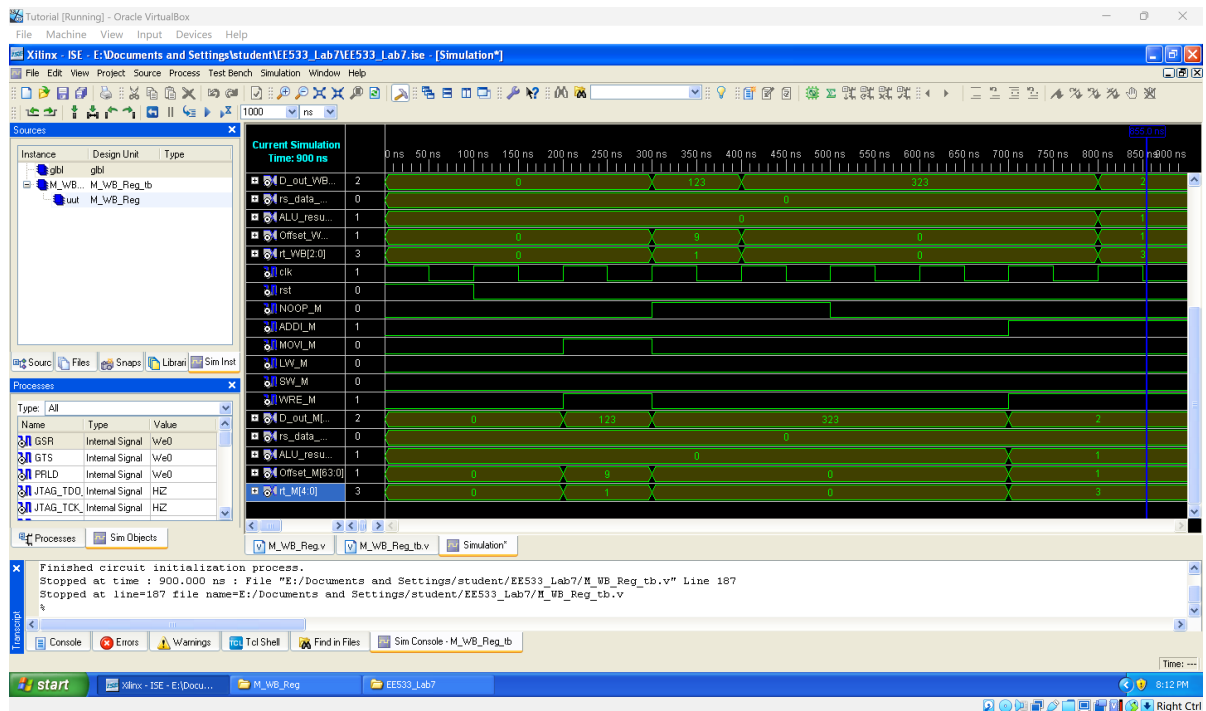
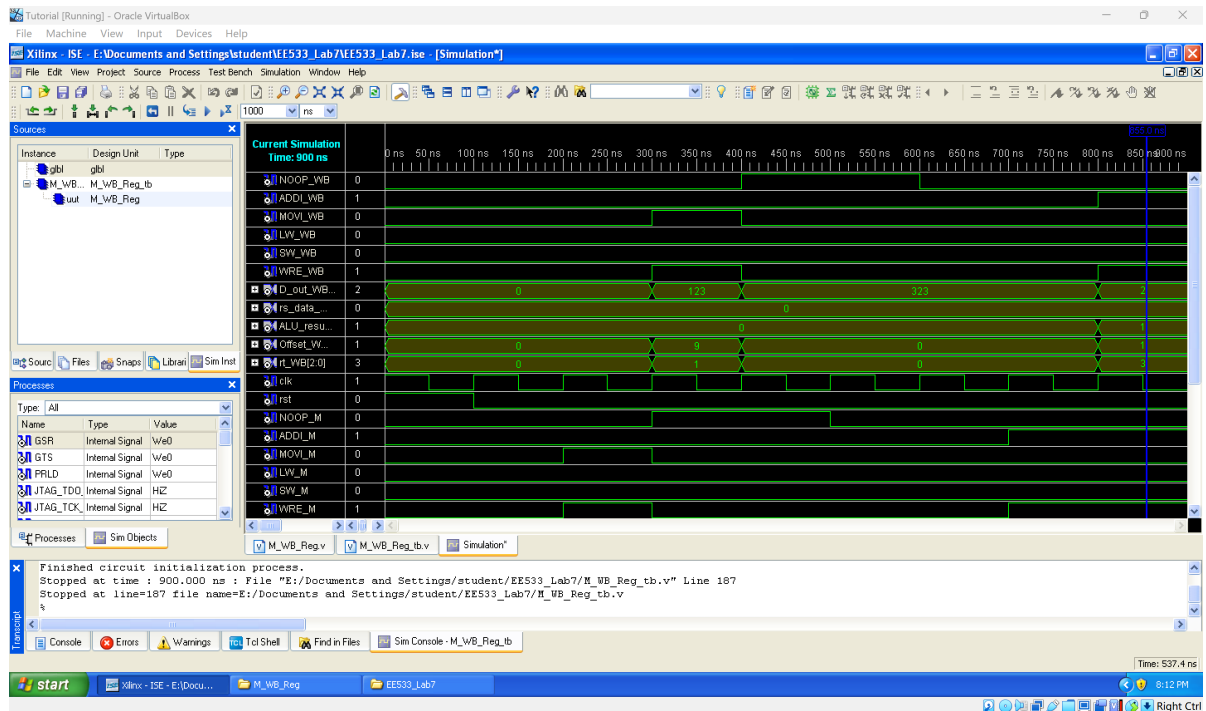
```
@(posedge clk);
```

```
@(posedge clk);
$stop;
```

end

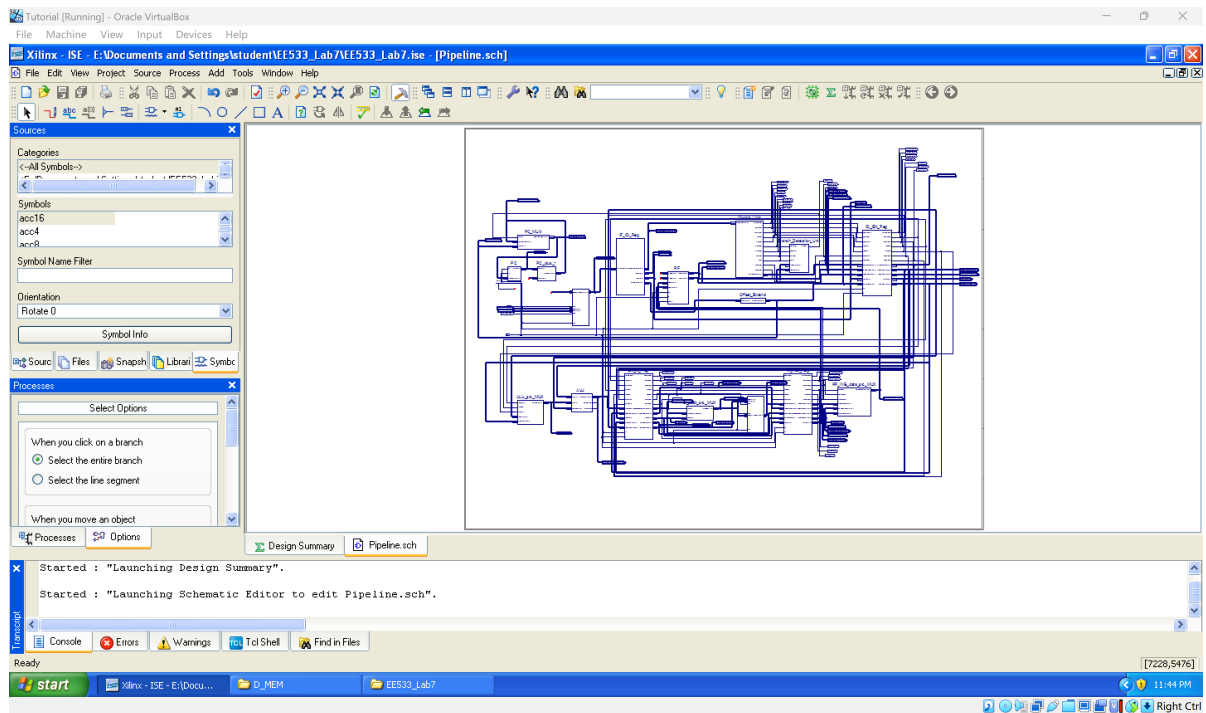
endmodule

- Waveform



4. 5-Stage Pipeline Generated

4.1 Schematic



4.2 Verilog

```

////////////////////////////////////
// Copyright (c) 1995-2008 xilinx, Inc. All rights reserved.
////////////////////////////////////
//
//      ____
//     /  \  /
//    /____\ \   vendor: xilinx
//     \    \  \   version : 10.1
//      \    \   Application : sch2verilog
//       /    /   Filename  : Pipeline.vf
//      /____\ /\   Timestamp : 03/01/2025 22:50:30
//       \    \  \
//      \___\ \___\
//
//Command: C:\Xilinx\10.1\ISE\bin\nt\unwrapped\sch2verilog.exe -intstyle ise -
family virtex2p -w "E:/Documents and Settings/student/EE533_Lab7/Pipeline.sch"
Pipeline.vf
//Design Name: Pipeline
//Device: virtex2p
//Purpose:
//   This verilog netlist is translated from an ECS schematic.It can be
//   synthesized and simulated, but it should not be modified.
//
`timescale 1ns / 1ps

module Pipeline(clk,
                Instr_IN,
                Instr_W_en,
                I_W_Addr,
                rst);

    input clk;
    input [31:0] Instr_IN;
    input Instr_W_en;

```



```

input [8:0] I_W_Addr;
input rst;

wire ADDI_EX;
wire ADDI_ID;
wire ADDI_M;
wire ADDI_WB;
wire [63:0] ALU_B;
wire [3:0] ALU_OP_EX;
wire [3:0] ALU_OP_ID;
wire [63:0] ALU_result_EX;
wire [63:0] ALU_result_M;
wire [63:0] ALU_result_WB;
wire BEQ_ID;
wire BGT_ID;
wire BLT_ID;
wire [63:0] D_out_WB;
wire [31:0] Instruction;
wire J_ID;
wire LW_EX;
wire LW_ID;
wire LW_M;
wire LW_WB;
wire MOVI_EX;
wire MOVI_ID;
wire MOVI_M;
wire MOVI_WB;
wire NOOP_EX;
wire NOOP_ID;
wire NOOP_M;
wire [15:0] Offset;
wire [63:0] Offset_EX;
wire [63:0] Offset_ID;
wire [63:0] Offset_M;
wire [63:0] Offset_WB;
wire [63:0] ONE;
wire [5:0] OP_CODE_ID;
wire [63:0] PC;
wire [63:0] PC_next;
wire [63:0] PC_plus_one;
wire [63:0] RF_WB_Din;
wire [63:0] rs_data_EX;
wire [63:0] rs_data_ID;
wire [63:0] rs_data_M;
wire [4:0] rs_ID;
wire [63:0] rt_data_EX;
wire [63:0] rt_data_ID;
wire [63:0] rt_data_M;
wire [4:0] rt_EX;
wire [4:0] rt_ID;
wire SW_EX;
wire SW_ID;
wire SW_M;
wire WME_EX;
wire WME_ID;
wire WME_M;

```

```

wire WRE_EX;
wire WRE_ID;
wire WRE_M;
wire WRE_WB;
wire XLXN_22;
wire [4:0] XLXN_93;
wire [7:0] XLXN_96;
wire [63:0] XLXN_100;
wire [2:0] XLXN_114;

PC_XLXI_1 (.clk(clk),
           .PC_next(PC_next[63:0]),
           .rst(rst),
           .PC(PC[63:0]));
PC_MUX_XLXI_3 (.BTA(Offset_ID[63:0]),
              .PC_ctrl(XLXN_22),
              .PC_next_in(PC_plus_one[63:0]),
              .PC_next_out(PC_next[63:0]));
I_MEM_XLXI_4 (.addra(PC[8:0]),
             .addrb(I_W_Addr[8:0]),
             .clka(clk),
             .clkb(clk),
             .dinb(Instr_IN[31:0]),
             .web(Instr_W_en),
             .douta(Instruction[31:0]));
IF_ID_Reg_XLXI_5 (.Instruction(Instruction[31:0]),
                 .Offset_ID(Offset[15:0]),
                 .OP_CODE_ID(OP_CODE_ID[5:0]),
                 .rs_ID(rs_ID[4:0]),
                 .rt_ID(rt_ID[4:0]));
RF_XLXI_6 (.clk(clk),
          .rst(rst),
          .r0addr(rs_ID[2:0]),
          .r1addr(rt_ID[2:0]),
          .waddr(XLXN_114[2:0]),
          .wdata(RF_WB_Din[63:0]),
          .wena(WRE_WB),
          .r0data(rs_data_ID[63:0]),
          .r1data(rt_data_ID[63:0]));
Control_Unit_XLXI_7 (.OP_CODE(OP_CODE_ID[5:0]),
                    .ADDI_ID(ADDI_ID),
                    .ALU_OP_ID(ALU_OP_ID[3:0]),
                    .BEQ_ID(BEQ_ID),
                    .BGT_ID(BGT_ID),
                    .BLT_ID(BLT_ID),
                    .J_ID(J_ID),
                    .LW_ID(LW_ID),
                    .MOVI_ID(MOVI_ID),
                    .NOOP_ID(NOOP_ID),
                    .SW_ID(SW_ID),
                    .WME_ID(WME_ID),
                    .WRE_ID(WRE_ID));
Offset_Extend_XLXI_9 (.Offset(Offset[15:0]),
                     .Offset_ID(Offset_ID[63:0]));
ID_EX_Reg_XLXI_10 (.ADDI_ID(ADDI_ID),
                  .ALU_OP_ID(ALU_OP_ID[3:0]),

```

```

        .clk(clk),
        .LW_ID(LW_ID),
        .MOVI_ID(MOVI_ID),
        .NOOP_ID(NOOP_ID),
        .Offset_ID(Offset_ID[63:0]),
        .rst(rst),
        .rs_data_ID(rs_data_ID[63:0]),
        .rt_data_ID(rt_data_ID[63:0]),
        .rt_ID(rt_ID[4:0]),
        .SW_ID(SW_ID),
        .WME_ID(WME_ID),
        .WRE_ID(WRE_ID),
        .ADDI_EX(ADDI_EX),
        .ALU_OP_EX(ALU_OP_EX[3:0]),
        .LW_EX(LW_EX),
        .MOVI_EX(MOVI_EX),
        .NOOP_EX(NOOP_EX),
        .Offset_EX(Offset_EX[63:0]),
        .rs_data_EX(rs_data_EX[63:0]),
        .rt_data_EX(rt_data_EX[63:0]),
        .rt_EX(rt_EX[4:0]),
        .SW_EX(SW_EX),
        .WME_EX(WME_EX),
        .WRE_EX(WRE_EX));

ALU_XLXI_11 (.A(rs_data_EX[63:0]),
            .ALU_OP(ALU_OP_EX[3:0]),
            .B(ALU_B[63:0]),
            .ALU_Out(ALU_result_EX[63:0]),
            .overflow(),
            .Zero_Flag());

ALU_src_MUX_XLXI_12 (.ADDI_EX(ADDI_EX),
                    .LW_EX(LW_EX),
                    .Offset_EX(Offset_EX[63:0]),
                    .rt_data(rt_data_EX[63:0]),
                    .SW_EX(SW_EX),
                    .ALU_B(ALU_B[63:0]));

EX_M_Reg_XLXI_13 (.ADDI_EX(ADDI_EX),
                 .ALU_result_EX(ALU_result_EX[63:0]),
                 .clk(clk),
                 .LW_EX(LW_EX),
                 .MOVI_EX(MOVI_EX),
                 .NOOP_EX(NOOP_EX),
                 .Offset_EX(Offset_EX[63:0]),
                 .rst(rst),
                 .rs_data_EX(rs_data_EX[63:0]),
                 .rt_data_EX(rt_data_EX[63:0]),
                 .rt_EX(rt_EX[4:0]),
                 .SW_EX(SW_EX),
                 .WME_EX(WME_EX),
                 .WRE_EX(WRE_EX),
                 .ADDI_M(ADDI_M),
                 .ALU_result_M(ALU_result_M[63:0]),
                 .LW_M(LW_M),
                 .MOVI_M(MOVI_M),
                 .NOOP_M(NOOP_M),
                 .Offset_M(Offset_M[63:0]),

```

```

        .rs_data_M(rs_data_M[63:0]),
        .rt_data_M(rt_data_M[63:0]),
        .rt_M(XLXN_93[4:0]),
        .SW_M(SW_M),
        .WME_M(WME_M),
        .WRE_M(WRE_M));
D_addr_src_MUX XLXI_14 (.ALU_result_M(ALU_result_M[63:0]),
        .rt_M(XLXN_93[4:0]),
        .SW_M(SW_M),
        .D_addr(XLXN_96[7:0]));
D_MEM XLXI_15 (.addra(XLXN_96[7:0]),
        .addrb(XLXN_96[7:0]),
        .clka(clk),
        .clkb(clk),
        .dina(rt_data_M[63:0]),
        .wea(WME_M),
        .doutb(XLXN_100[63:0]));
M_WB_Reg XLXI_16 (.ADDI_M(ADDI_M),
        .ALU_result_M(ALU_result_M[63:0]),
        .clk(clk),
        .D_out_M(XLXN_100[63:0]),
        .LW_M(LW_M),
        .MOVI_M(MOVI_M),
        .NOOP_M(NOOP_M),
        .Offset_M(Offset_M[63:0]),
        .rst(rst),
        .rs_data_M(rs_data_M[63:0]),
        .rt_M(XLXN_93[4:0]),
        .SW_M(SW_M),
        .WRE_M(WRE_M),
        .ADDI_WB(ADDI_WB),
        .ALU_result_WB(ALU_result_WB[63:0]),
        .D_out_WB(D_out_WB[63:0]),
        .LW_WB(LW_WB),
        .MOVI_WB(MOVI_WB),
        .NOOP_WB(),
        .Offset_WB(Offset_WB[63:0]),
        .rs_data_WB(),
        .rt_WB(XLXN_114[2:0]),
        .SW_WB(),
        .WRE_WB(WRE_WB));
RF_WB_data_src_MUX XLXI_17 (.ADDI_WB(ADDI_WB),
        .ALU_out_WB(ALU_result_WB[63:0]),
        .D_out_WB(D_out_WB[63:0]),
        .LW_WB(LW_WB),
        .MOVI_WB(MOVI_WB),
        .Offset_WB(Offset_WB[63:0]),
        .RF_WB_Din(RF_WB_Din[63:0]));
PC_plus_1 XLXI_18 (.ONE(ONE[63:0]),
        .PC(PC[63:0]),
        .PC_next(PC_plus_one[63:0]));
VCC XLXI_19 (.P(ONE[0]));
Branch_Detection_Unit XLXI_21 (.BEQ_ID(BEQ_ID),
        .BGT_ID(BGT_ID),
        .BLT_ID(BLT_ID),
        .J_ID(J_ID),

```

```

        .rs_data(rs_data_ID[63:0]),
        .rt_data(rt_data_ID[63:0]),
        .PC_ctrl(XLXN_22));

endmodule

```

4.3 Testbench

```

`timescale 1ns / 1ps

/////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    23:28:48 03/01/2025
// Design Name:    Pipeline
// Module Name:    E:/Documents and
Settings/student/EE533_Lab7/EE533_Lab_7/Pipeline_tb.v
// Project Name:   EE533_Lab_7
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: Pipeline
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
/////////////////////////////////////////////////////////////////

module Pipeline_tb;

    // Inputs
    reg clk;
    reg [31:0] Instr_IN;
    reg Instr_W_en;
    reg [8:0] I_W_Addr;
    reg rst;

    // Outputs
    wire ADDI_EX;
    wire ADDI_ID;
    wire ADDI_M;
    wire ADDI_WB;
    wire [63:0] ALU_B;
    wire [3:0] ALU_OP_EX;
    wire [3:0] ALU_OP_ID;
    wire [63:0] ALU_result_EX;
    wire [63:0] ALU_result_M;
    wire [63:0] ALU_result_WB;
    wire BEQ_ID;
    wire BGT_ID;
    wire BLT_ID;

```

```

wire [7:0] D_MEM_addr;
wire [63:0] D_out_M;
wire [63:0] D_out_WB;
wire [31:0] Instruction;
wire J_ID;
wire LW_EX;
wire LW_ID;
wire LW_M;
wire LW_WB;
wire MOVI_EX;
wire MOVI_ID;
wire MOVI_M;
wire MOVI_WB;
wire NOOP_EX;
wire NOOP_ID;
wire NOOP_M;
wire NOOP_WB;
wire [63:0] Offset_EX;
wire [63:0] Offset_ID;
wire [63:0] Offset_M;
wire [63:0] Offset_WB;
wire [5:0] OP_CODE_ID;
wire [63:0] PC;
wire PC_ctrl;
wire [63:0] PC_next;
wire [63:0] PC_plus_one;
wire [63:0] RF_WB_Din;
wire [63:0] rs_data_EX;
wire [63:0] rs_data_ID;
wire [63:0] rs_data_M;
wire [63:0] rs_data_WB;
wire [4:0] rs_ID;
wire [63:0] rt_data_EX;
wire [63:0] rt_data_ID;
wire [63:0] rt_data_M;
wire [4:0] rt_EX;
wire [4:0] rt_ID;
wire [4:0] rt_M;
wire [2:0] rt_WB;
wire SW_EX;
wire SW_ID;
wire SW_M;
wire SW_WB;
wire WME_EX;
wire WME_ID;
wire WME_M;
wire WRE_EX;
wire WRE_ID;
wire WRE_M;
wire WRE_WB;

// Instantiate the Unit Under Test (UUT)
Pipeline uut (
    .clk(clk),
    .Instr_IN(Instr_IN),
    .Instr_W_en(Instr_W_en),

```

```
.I_W_Addr(I_W_Addr),
.rst(rst),
.ADDI_EX(ADDI_EX),
.ADDI_ID(ADDI_ID),
.ADDI_M(ADDI_M),
.ADDI_WB(ADDI_WB),
.ALU_B(ALU_B),
.ALU_OP_EX(ALU_OP_EX),
.ALU_OP_ID(ALU_OP_ID),
.ALU_result_EX(ALU_result_EX),
.ALU_result_M(ALU_result_M),
.ALU_result_WB(ALU_result_WB),
.BEQ_ID(BEQ_ID),
.BGT_ID(BGT_ID),
.BLT_ID(BLT_ID),
.D_MEM_addr(D_MEM_addr),
.D_out_M(D_out_M),
.D_out_WB(D_out_WB),
.Instruction(Instruction),
.J_ID(J_ID),
.LW_EX(LW_EX),
.LW_ID(LW_ID),
.LW_M(LW_M),
.LW_WB(LW_WB),
.MOVI_EX(MOVI_EX),
.MOVI_ID(MOVI_ID),
.MOVI_M(MOVI_M),
.MOVI_WB(MOVI_WB),
.NOOP_EX(NOOP_EX),
.NOOP_ID(NOOP_ID),
.NOOP_M(NOOP_M),
.NOOP_WB(NOOP_WB),
.Offset_EX(Offset_EX),
.Offset_ID(Offset_ID),
.Offset_M(Offset_M),
.Offset_WB(Offset_WB),
.OP_CODE_ID(OP_CODE_ID),
.PC(PC),
.PC_ctrl(PC_ctrl),
.PC_next(PC_next),
.PC_plus_one(PC_plus_one),
.RF_WB_Din(RF_WB_Din),
.rs_data_EX(rs_data_EX),
.rs_data_ID(rs_data_ID),
.rs_data_M(rs_data_M),
.rs_data_WB(rs_data_WB),
.rs_ID(rs_ID),
.rt_data_EX(rt_data_EX),
.rt_data_ID(rt_data_ID),
.rt_data_M(rt_data_M),
.rt_EX(rt_EX),
.rt_ID(rt_ID),
.rt_M(rt_M),
.rt_WB(rt_WB),
.SW_EX(SW_EX),
.SW_ID(SW_ID),
```

[illegible]

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

```
@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@(posedge clk);

@stop;

end

endmodule
```

4. GitHub Link

- https://github.com/yuezhenglingluan/USC_EE533_lab7.git

