LAB 8
MIPS Instruction Decode Stage

In the single cycle MIPS processor, there are 5 stages of execution. The Instruction Decode stage is used to generate the value of control signals and perform register access:

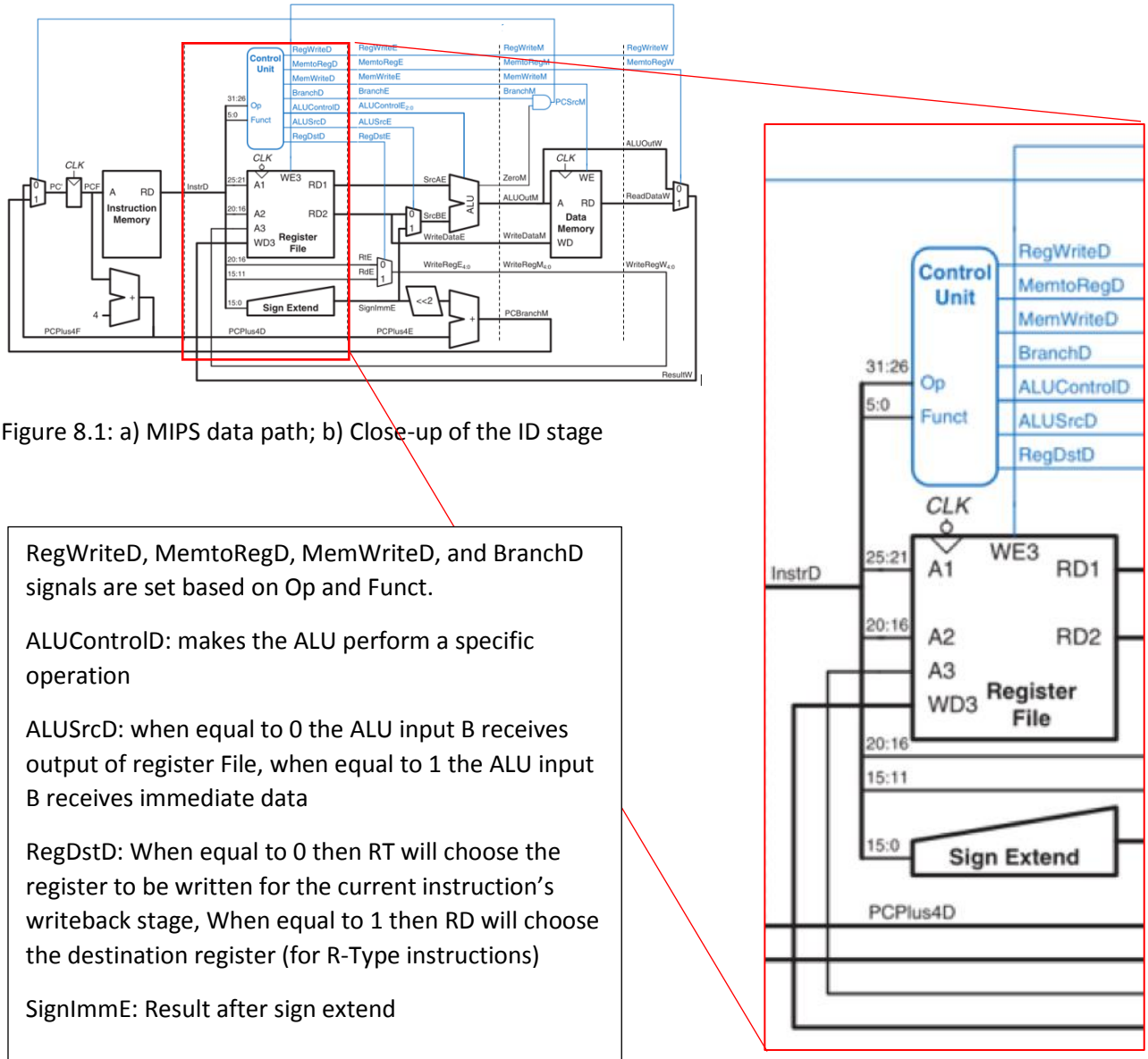

Figure 8.1: a) MIPS data path; b) Close-up of the ID stage

RegWriteD, MemtoRegD, MemWriteD, and BranchD signals are set based on Op and Funct.

ALUControlD: makes the ALU perform a specific operation

ALUSrcD: when equal to 0 the ALU input B receives output of register File, when equal to 1 the ALU input B receives immediate data

RegDstD: When equal to 0 then RT will choose the register to be written for the current instruction's writeback stage, When equal to 1 then RD will choose the destination register (for R-Type instructions)

SignImmE: Result after sign extend

Get the regfile.v file from lab 2. **Change width to 32.**

A sign extender is needed: create a file **se16.v** and implement Verilog module given in the Figure 8.2.

```
module se16(a, y);
  input [15:0] a;
  output [31:0] y;

  assign y= (a[15]==1) ? {16'hffff,a}:
  {16'h0000,a};

endmodule
```

Figure 8.2: Sign-extend module

Create Control Unit in a file named **controlUnit.v** and name the module **controlUnit**, as shown in the Figure 8.3.:

```
`timescale 1ns/100ps
module controlUnit(op, funct, regwrited, memtoregd, memwrited, branchd,
alucontrold, alusrcd, regdstd);
  input [5:0] op, funct;
  output [2:0] alucontrold;
  output regwrited, memtoregd, memwrited, branchd, alusrcd, regdstd;

  assign alucontrold = (op==0 & funct==36) ? 0:
    (op==0 & funct==37) ? 1:
    ((op==0 & funct==32) | op==8 | op==35 | op==43) ? 2:
    ((op==0 & funct==34)| op==4) ? 6:
    (op==0 & funct==42) ? 7:
    3'hx, //default alucontrold value
    regwrited = (op==0 | op==8 | op==35) ? 1:0,
    memtoregd = (op==35) ? 1:
    (op==0 | op==8) ? 0: 1'bx, //default memtoregd value
    memwrited = (op==43) ? 1:0,
    branchd = (op==4) ? 1:0,
    alusrcd = (op==8 | op==35 | op==43) ? 1:0,
    regdstd = (op==0) ? 1: (op==8 | op==35) ? 0 :
    1'bx; //default regdstd value
endmodule
```

Figure 8.3: Control Unit module

Use the block diagram in Figure 8.1 and complete the module skeleton below in order to create the ID stage:

```
1  `timescale 1ns/100ps
2  `include "regfile.v"
3  `include "se16.v"
4  `include "controlUnit.v"
5
6  `define datasize 32
7  module ID_Stage(clk, instr, writeregw, resultw, regwritew, memtoregd,
   memwrited, branchd, alucontrold, alusrcd, regdstd, regwrited, rd1, rd2,
   signimme);
8
9    input regwritew, clk;
10   input [`datasize-1:0] instr, resultw;
11   input [4:0] writeregw;
12   output [2:0] alucontrold;
13   output regwrited, memtoregd, memwrited, branchd, alusrcd, regdstd;
14   output [`datasize-1:0] rd1, rd2, signimme;
15
16   assign regwritew = regwrited;
17
18  //Fill out
19   controlUnit controlUnit();
20   regfile rf();
21   se16 signExtend();
22
23 endmodule
```

Figure 8.4: Skeleton of ID_Stage module

The skeleton of the testbench file is on the BeachBoard. **You must update the testbench!** Fill out the Table 8.1, and use the values you filled in to update the Verilog test fixture lab8_testbench.

Table 8.1

Test Case	Operation	Instr
0	add $5, $3, $4	32'h00642820
1	sub $8, $4, $5	
2	slt $9, $4, $5	
3	beq $10, $4, 2	
4	lw $11, 4($0)	
5	sw $12, 4($0)	
6	and $13, $4, $5	
7	or $14, $4, $5	

Test your design to ensure it works correctly: if your design is correct, you should obtain the expected values as shown in the Figure 8.5.

WHAT TO SUBMIT

Once you have verified proper functionality of your project:

1. Copy the contents of **ID_Stage module (design.sv)** to a text file named **ID_Stage.txt** and upload the BeachBoard Dropbox for Lab 8.
2. Copy the contents of your **lab8_testbench.sv** into **lab8_testbench.txt** and upload the BeachBoard Dropbox for Lab 8.
3. Copy the contents of *all* other files used in this lab: **regrile.v, se16.v, controlUnit.v** into the file with the same name but with **.txt** extension and upload the BeachBoard Dropbox for Lab 8.
4. Additionally, upload your Lab report (see the LabReportTemplete in the Documents folder on BeachBoard), include populated Table 1 in your lab repot.

NOTE: **keep** these lab files as they will be needed for Extra Credit lab!

Created by Josh Hayter, updated by Jelena Trajkovic

```
Control Signals for test case 0: 1000010011
Control Signals for test case 1: 1000110011
Control Signals for test case 2: 1000111011
Control Signals for test case 3: 0x011100x0
Control Signals for test case 4: 1100010101
Control Signals for test case 5: 0x100101x0
Control Signals for test case 6: 1000000011
Control Signals for test case 7: 1000001011
Done
```

Figure 8.5: Expected output for the ID_Stage module