Computer Organization and Architecture Laboratory

Final Assignment

RISC Processor Design

Group 2

Akash Das (20CS10006)

Suhas A M (20CS10066)

Instruction Format:

5 types:

(i) R-type:

opcode	rs	rt	Immediate
6 bits	5 bits	5 bits	16 bits

(ii) Immediate:

opcode	rs	Immediate
6 bits	5 bits	21 bits

(iii) Branching/ Conditional:

opcode	rs	offset
6 bits	5 bits	21 bits

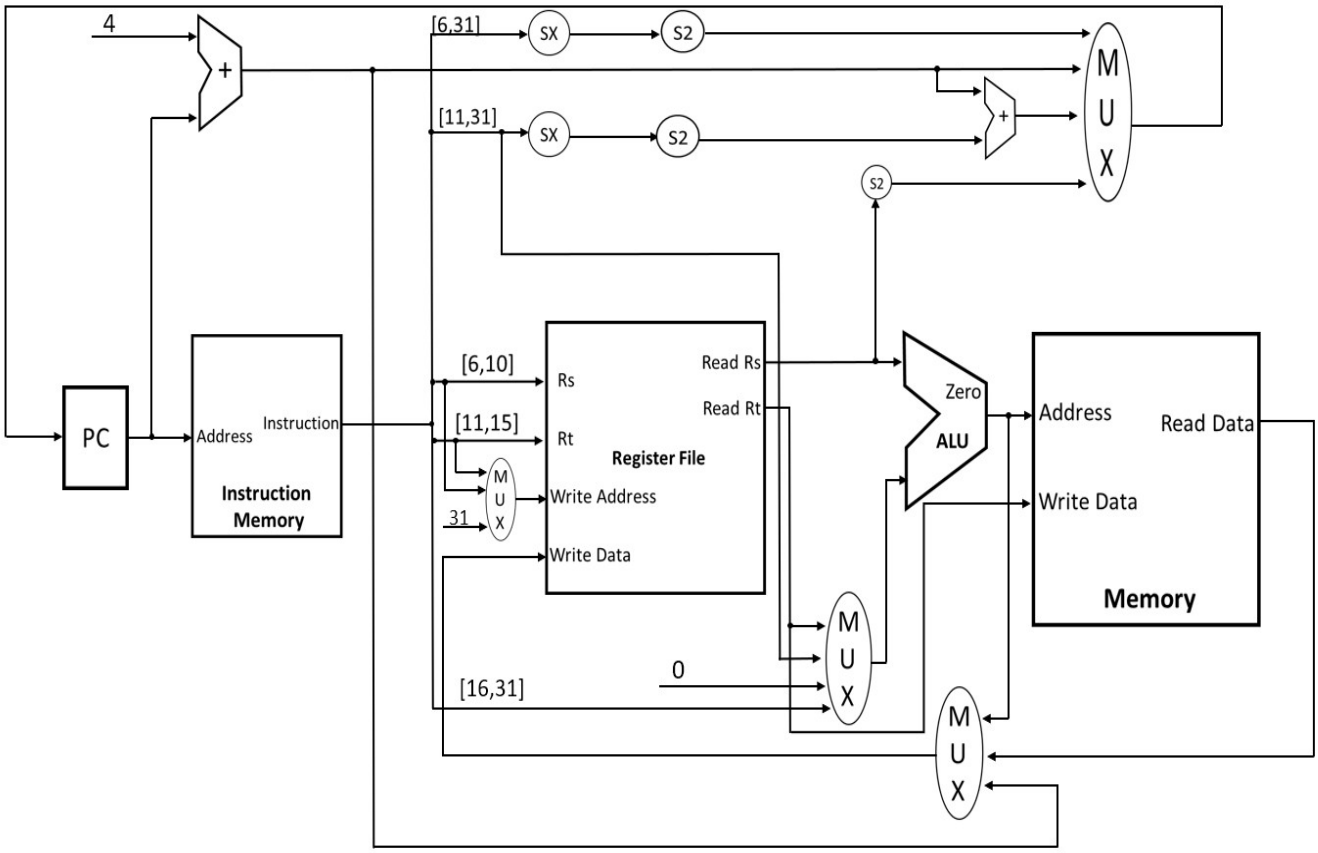
(iv) Unconditional jump:

opcode	address
6 bits	26 bits

(v) Memory read/write:

opcode	rs	rt	Offset
6 bits	5 bits	5 bits	16 bits

Design:



Control Signal:

** dnm: doesn’t matter

ADD:

000000

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b00; // rs

data_select = 2'b00; // alu_output

alu_src = 2'b00; // rt_data

instr_select = 2'b00; // pc+1

operation = 4'd0; // add

COMP:
000001

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b00; // rs

data_select = 2'b00; // alu_output

alu_src = 2'b00; // rt_data

instr_select = 2'b00; // pc+1

operation = 4'd1; // comp

AND:

000010

// and

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b00; // rs

data_select = 2'b00; // alu_output

alu_src = 2'b00; // rt_data

instr_select = 2'b00; // pc+1

operation = 4'd2; // and

XOR:

000011

// xor

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b00; // rs

data_select = 2'b00; // alu_output

alu_src = 2'b00; // rt_data

instr_select = 2'b00; // pc+1

operation = 4'd3; // xor

ADDI:

010000

//addi

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b00; // rs

data_select = 2'b00; // alu_output

alu_src = 2'b11; // imm2

instr_select = 2'b00; // pc+1

operation = 4'd0; // add

COMI:

010001

//compi

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b00; // rs

data_select = 2'b00; // alu_output

alu_src = 2'b11; // imm2

instr_select = 2'b00; //

operation = 4'd1; //

SHLL:

000100

// shll

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b00; // rs

data_select = 2'b00; // alu_output

alu_src = 2'b10; // imm1

instr_select = 2'b00; // pc+1

operation = 4'd4; // shll

SHRL:

000101

// shrl

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b00; // rs

data_select = 2'b00; // alu_output

alu_src = 2'b10; // imm1

instr_select = 2'b00; // pc+1

operation = 4'd5; // shrl

SHLLV:

001000

// shllv

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b00; // rs

data_select = 2'b00; // alu_output

alu_src = 2'b00; // rt_data

instr_select = 2'b00; // pc+1

operation = 4'd4; // shll

SHRLV:

001001

// shrlv

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b00; // rs

data_select = 2'b00; // alu_output

alu_src = 2'b00; // rt_data

instr_select = 2'b00; // pc+1

operation = 4'd5; // shrl

SHRA:

000110

// shra

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b00; // rs

data_select = 2'b00; // alu_output

alu_src = 2'b10; // imm1

instr_select = 2'b00; // pc+1

operation = 4'd6; // shrl

SHRAV:

001010

// shrav

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b00; // rs

data_select = 2'b00; // alu_output

alu_src = 2'b00; // rt_data

instr_select = 2'b00; // pc+1

operation = 4'd6; // shrl

LW:

001110

// lw

mem_write = 1'b0; //

mem_read = 1'b1; //

write_reg = 1'b1; //

reg_dst = 2'b01; // rt

data_select = 2'b01; // mem_output

alu_src = 2'b10; // imm1

instr_select = 2'b00; // pc+1

operation = 4'd0; // add

SW:

001111

// sw

mem_write = 1'b1; //

mem_read = 1'b1; //

write_reg = 1'b0; //

reg_dst = 2'b00; // doesn't matter

data_select = 2'b00; // doesn't matter

alu_src = 2'b10; // imm1

instr_select = 2'b00; // pc+1

operation = 4'd0; // add

B:

100000

// b

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b0; //

reg_dst = 2'b00; // dnm

data_select = 2'b00; // dnm

alu_src = 2'b00; // dnm

instr_select = 2'b10; // label

operation = 4'd8; //

BR:

010010

//br

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b0; //

reg_dst = 2'b00; // dnm

data_select = 2'b00; // dnm

alu_src = 2'b00; // dnm

instr_select = 2'b11; // (rs)

operation = 4'd8; //

BLTZ:

010011

//bltz

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b0; //

reg_dst = 2'b00; // dnm

data_select = 2'b00; // dnm

alu_src = 2'b00; // dnm

instr_select = 2'b01; // imm2

operation = 4'd8; //

BZ:

010100

//bz

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b0; //

reg_dst = 2'b00; // dnm

data_select = 2'b00; // dnm

alu_src = 2'b00; // dnm

instr_select = 2'b01; // imm2

operation = 4'd8; //

BNZ:

010101

//bnz

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b0; //

reg_dst = 2'b00; // dnm

data_select = 2'b00; // dnm

alu_src = 2'b00; // dnm

instr_select = 2'b01; // imm2

operation = 4'd8; //

BL:

100001

// bl

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b10; // 31

data_select = 2'b10; // pc + 1

alu_src = 2'b00; // dnm

instr_select = 2'b10; // label

operation = 4'd8; //

BCY:

100010

// bcy

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b0; //

reg_dst = 2'b00; // dnm

data_select = 2'b00; // dnm

alu_src = 2'b00; // dnm

instr_select = 2'b10; // label

operation = 4'd8; //

BNCY:

100011

// bncy

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b0; //

reg_dst = 2'b00; // dnm

data_select = 2'b00; // dnm

alu_src = 2'b00; // dnm

instr_select = 2'b10; // label

operation = 4'd8; //

DIFF:

000111

// diff

mem_write = 1'b0; //

mem_read = 1'b0; //

write_reg = 1'b1; //

reg_dst = 2'b00; // rs

data_select = 2'b00; // alu_output

alu_src = 2'b00; // rt_data

instr_select = 2'b00; // pc+1

operation = 4'd7; // diff