

Instruction: 1000

Explanation: $RD = RS1 + RS2$

IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;

ID : ALUSrcA=0; ALUSrcB=5;

EX : ALUSrcA=1; ALUSrcB=1; ALUOp=Add

MEMR : MemtoReg=0; RegDst=0; RegWrite=1;

WB : _____

Instruction: 1001

Explanation: $RD = RD + \text{Sign-Extended-Immediate Data}$

IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;

ID : ALUSrcA=0; ALUSrcB=5;

EX : ALUSrcA=2; ALUSrcB=2; ALUOp=Add

MEMR : MemtoReg=0; RegDst=0; RegWrite=1;

WB : _____

Instruction: 1010

Explanation: $RD = RD + \text{Immediate Data (upper byte filled by zeros)}$

IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;

ID : ALUSrcA=0; ALUSrcB=5;

EX : ALUSrcA=2; ALUSrcB=3; ALUOp=Add

MEMR : MemtoReg=0; RegDst=0; RegWrite=1;

WB : _____

Instruction: 1100

Explanation: $RD = RS1 - RS2$

IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;

ID : ALUSrcA=0; ALUSrcB=5;

EX : ALUSrcA=1; ALUSrcB=1; ALUOp=Sub

MEMR : MemtoReg=0; RegDst=0; RegWrite=1;

WB : _____

Instruction: 1101

Explanation: $RD = RD - \text{Sign-Extended-Immediate Data}$

IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;

ID : ALUSrcA=0; ALUSrcB=5;

EX : ALUSrcA=2; ALUSrcB=2; ALUOp=Sub

MEMR : MemtoReg=0; RegDst=0; RegWrite=1;

WB : _____

Instruction: 1110

Explanation: $RD = RD - \text{Immediate Data (upper byte filled by zeros)}$

IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;

ID : ALUSrcA=0; ALUSrcB=5;

EX : ALUSrcA=2; ALUSrcB=3; ALUOp=Sub
MEMR : MemtoReg=0; RegDst=0; RegWrite=1;
WB : _____

Instruction: 0000 Func Field: 0001

Explanation: RD = shift logical left (fill zeros) RD by the immediate number specified
IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;
ID : ALUSrcA=0; ALUSrcB=5;
EX : ALUSrcA=2; ALUSrcB=4; ALUOp=SLL
MEMR : MemtoReg=0; RegDst=0; RegWrite=1;
WB : _____

Instruction: 0000 Func Field: 0010

Explanation: RD = shift logical right (fill zeros) RD by the immediate number specified
IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;
ID : ALUSrcA=0; ALUSrcB=5;
EX : ALUSrcA=2; ALUSrcB=4; ALUOp=SRL
MEMR : MemtoReg=0; RegDst=0; RegWrite=1;
WB : _____

Instruction: 0000 Func Field: 0011

Explanation: RD = shift right (copy MSB) RD by the immediate number specified
IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;
ID : ALUSrcA=0; ALUSrcB=5;
EX : ALUSrcA=2; ALUSrcB=4; ALUOp=SAR
MEMR : MemtoReg=0; RegDst=0; RegWrite=1;
WB : _____

Instruction: 1011

Explanation: RD = RS1 nand RS2
IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;
ID : ALUSrcA=0; ALUSrcB=5;
EX : ALUSrcA=1; ALUSrcB=1; ALUOp=Nand
MEMR : MemtoReg=0; RegDst=0; RegWrite=1;
WB : _____

Instruction: 1111

Explanation: RD = RS1 or RS2
IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;
ID : ALUSrcA=0; ALUSrcB=5;
EX : ALUSrcA=2; ALUSrcB=2; ALUOp=Or
MEMR : MemtoReg=0; RegDst=0; RegWrite=1;
WB : _____

Instruction: 0111

Explanation: RD = RD nand (sign extended immediate data)

IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;

ID : ALUSrcA=0; ALUSrcB=5;

EX : ALUSrcA=2; ALUSrcB=2; ALUOp=Nand

MEMR : MemtoReg=0; RegDst=0; RegWrite=1;

WB : _____

Instruction: 0110

Explanation: RD = RD or (sign extended immediate data)

IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;

ID : ALUSrcA=0; ALUSrcB=5;

EX : ALUSrcA=2; ALUSrcB=2; ALUOp=Or

MEMR : MemtoReg=0; RegDst=0; RegWrite=1;

WB : _____

Instruction: 0100

Explanation: branch to the address in register RT when RA and RB are equal

IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;

ID : ALUSrcA=0; ALUSrcB=5;

EX : ALUSrcA=1; ALUSrcB=1; ALUOp=Sub; {PCsrc=2; PCWriteCond1=1; --logic for PC Write}

MEMR : _____

WB : _____

Instruction: 0101

Explanation: branch to the address in register RT when RA and RB are unequal

IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;

ID : ALUSrcA=0; ALUSrcB=5;

EX : ALUSrcA=1; ALUSrcB=1; ALUOp=Sub; {PCsrc=2; PCWriteCond2=1; --logic for PC Write}

MEMR : _____

WB : _____

Instruction: 0011

Explanation: Jump to PC + sign extended immediate data

IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;

ID : ALUSrcA=0; ALUSrcB=5;

EX : PCsrc=0; PCwrite=1;

MEMR : _____

WB : _____

Instruction: 0001

Explanation: load RD from data memory, whose location is pointed out by RP + sign- extended immediate data left shifted by one bit. RD can be one among R12, R13, R14 and R15 and

hence the upper 2-bits (11) of the register id are not specified. RP (pointer) can be one among R8, R9, R10 and R11 only, hence upper two bits (10) are not specified.

IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;

ID : ALUSrcA=0; ALUSrcB=5;

EX : ALUSrcA=3; ALUSrcB=6; ALUOp=Add

MEMR : IorD=1; MemWrite=1; IRWrite=0;

WB : MemtoReg=1; RegDst=1; RegWrite=1;

Instruction: 0010

Explanation: Store contents of RD to data memory location pointed out by RP + sign- extended immediate data padded with a LSB 0. RD can be one among R12, R13, R14 and R15 and hence the upper 2-bits (11) of the register id are not specified. RP (pointer) can be one among R8, R9, R10 and R11 only, hence upper two bits (10) are not specified.

IF : MemRead=1; IRWrite=1; PCSrc=1; PCWrite=1; IorD=0; ALUSrcA=0; ALUSrcB=0;

ID : ALUSrcA=0; ALUSrcB=5;

EX : ALUSrcA=3; ALUSrcB=6; ALUOp=Add;

MEMR : IorD=1; MemWrite=1

WB : _____

ALUOp signals(3 bits) -

Add - 000

Shift Arithmetic Right - 110

Sub - 001

Nand - 010

Shift Logical Left - 100

Or - 011

Shift Right Logical - 101

Control Signals(Bits) -

ImRead(1) - Instruction memory read, 1-active, 0-not active

PCWrite(1) - Write to PC reg, for PC=PC+2, 1-active, 0-not active

PCWriteCond1(1) - 1 implies Branch if equal opcode, 0 implies other opcode

PCWriteCond2(1) - 1 implies Branch if not equal opcode, 0 implies other opcode

MemRead(1) - Data memory read, 1-active, 0-not active

MemWrite(1) - Data memory write, 1-active, 0-not active

IRWrite(1) - Instruction register write, 1-active, 0-not active

MemtoReg(1) - Mux to decide write data for registers, 1-MDR, 0-ALUout

PCSource(2) - Mux for selecting data to PC, 0-ALUout, 1-PC+2, 2-Branch register value(RT)

ALUOp_MC(3) - Output from main control to ALUcontrol depending only on opcode

ALUSrcB(3) - value of 2nd input to ALU, 0-2, 1-reg[3:0], 2-sign_extended[7:0],

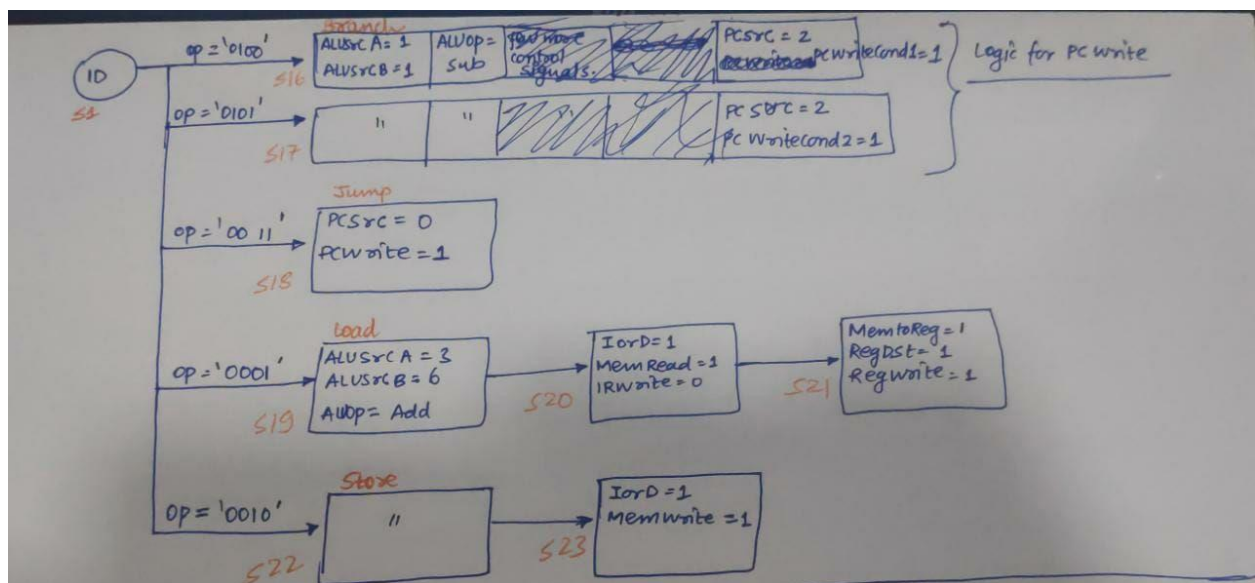
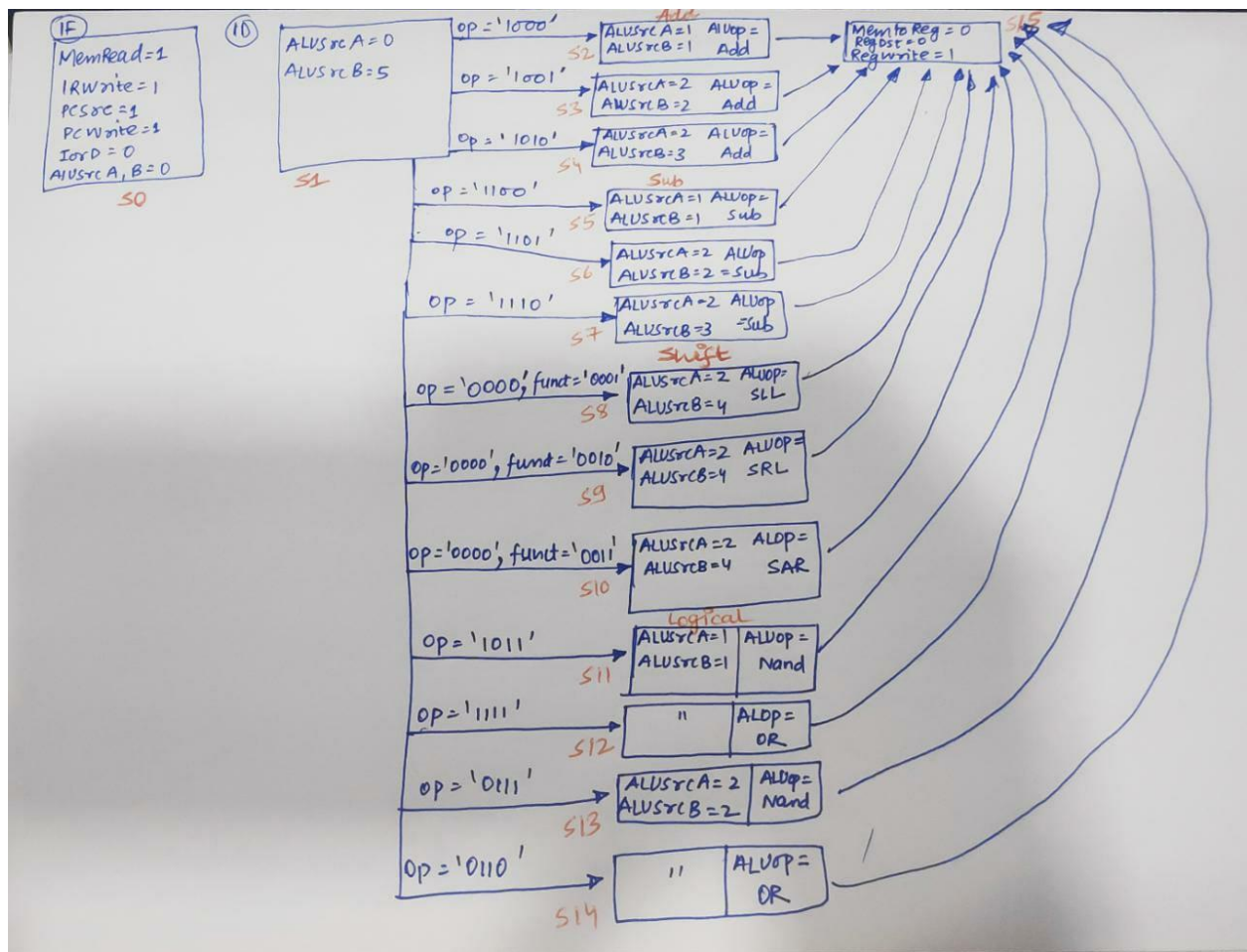
3-zero_extension[7:0], 4-zero_extension[7:4], 5-sign_extension[11:0],

6-sign_extended_1bit_shift_left[7:0]

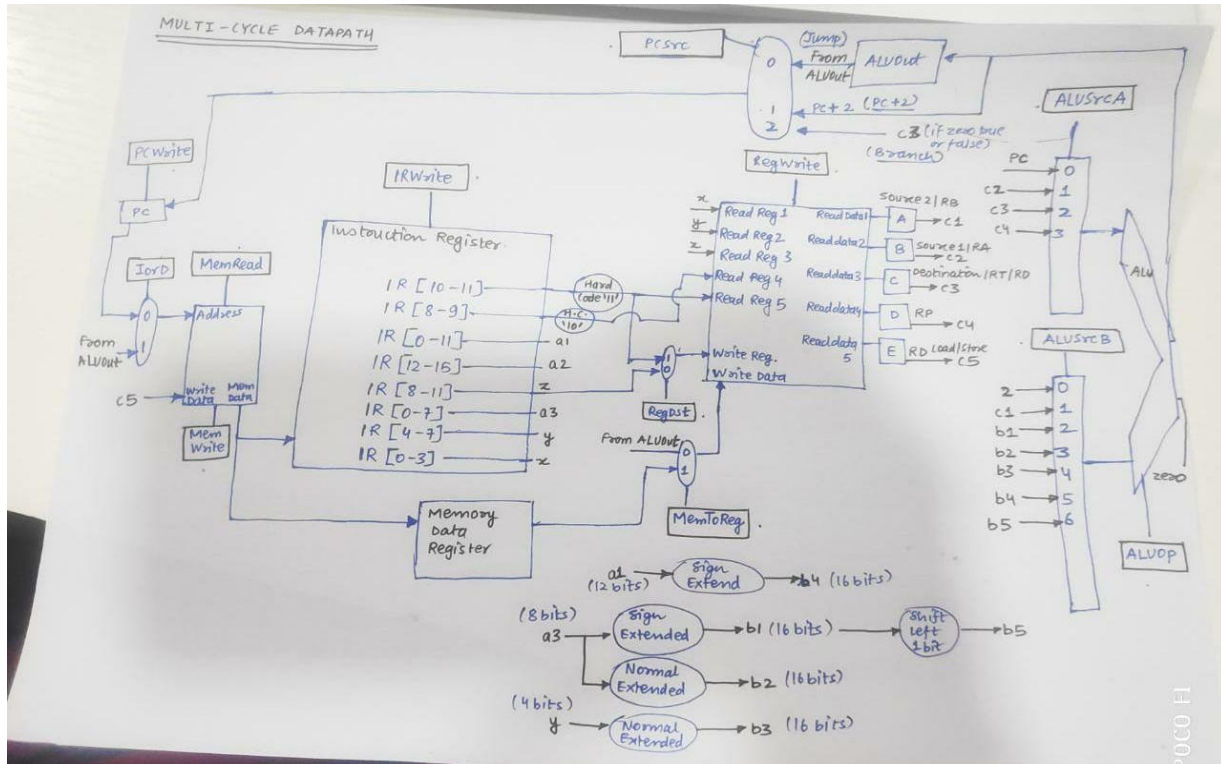
ALUSrcA(2) - value of 1st input to ALU, 0-PC, 1-reg[7:4], 2-reg[11:8], 3-reg['10'+9:8]

RegWrite(1) - Write to registers, 1-active, 0-not active

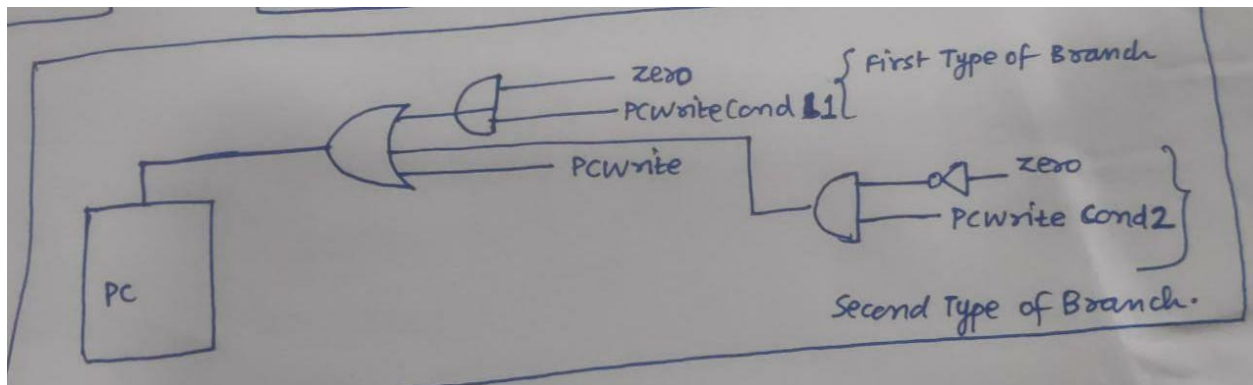
RegDst(1) - chooses input for write reg, 0-Instr[11:8], 1-['11'+Instr[11:10]]



FSM for the processor, each block is 1 cycle.



Datapath



Control signal for writing to PC reg