IITB-CPU

EE224: Digital Systems

By

Anshu Arora

(22B1207)

Chinmay Moorjani

(22B1212)

Sachi Deshmukh

(22B1213)

Sravan K Suresh

(22B3936)

Under the guidance of

Prof. Virendra Singh

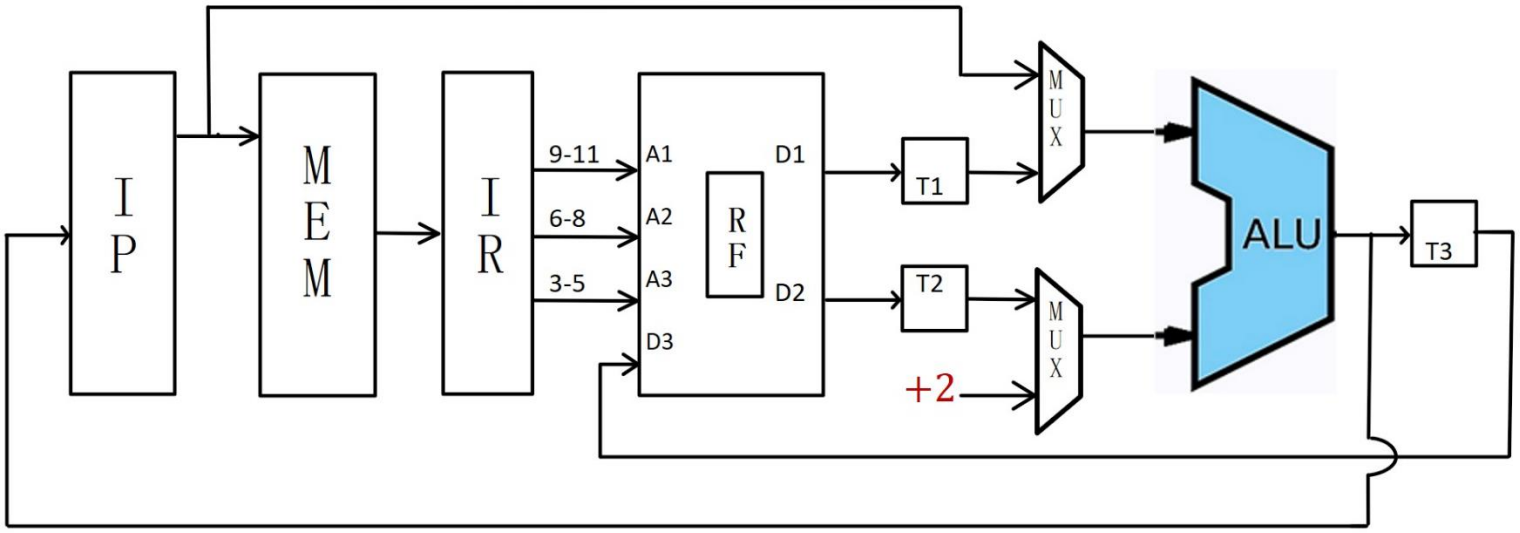
Department of Electrical Engineering,

Indian Institute of Technology, Bombay

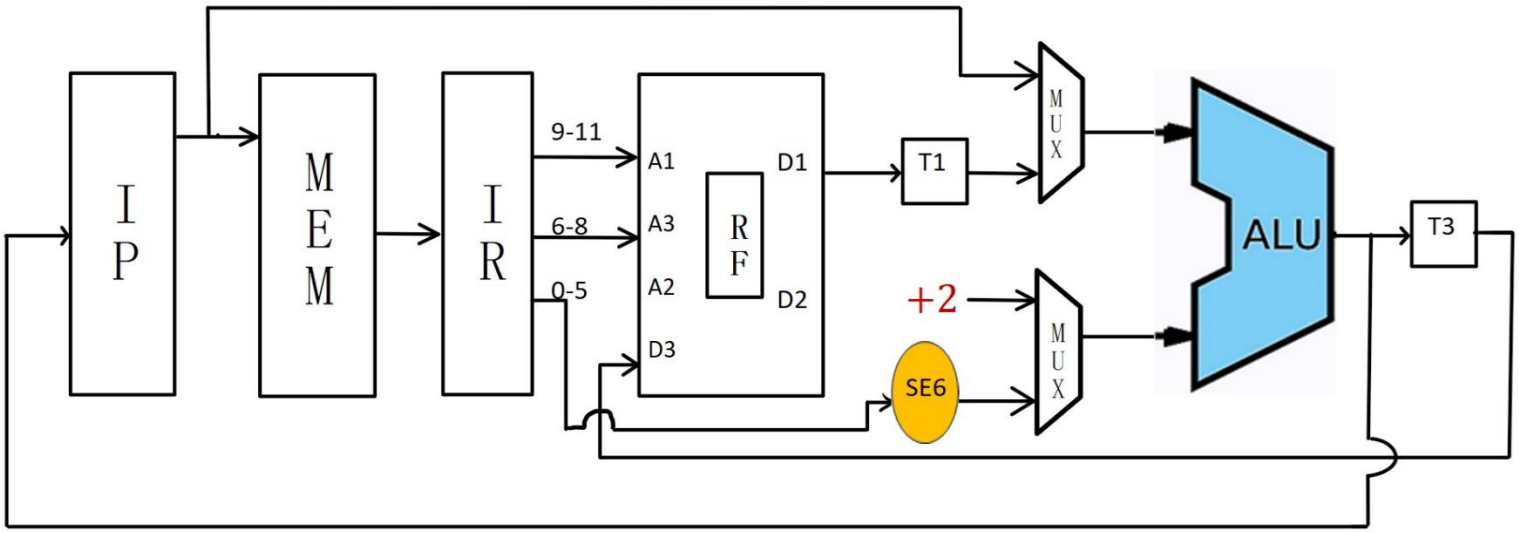
November, 2023

Components of IITB-CPU Design

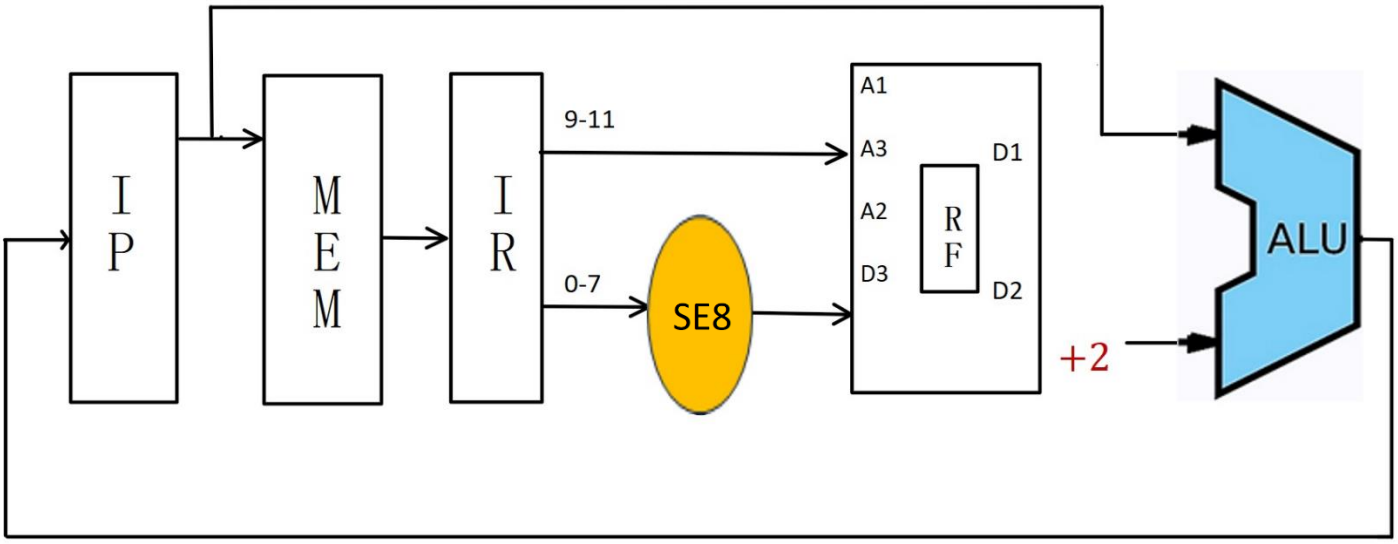
ADD/SUB/MUL/AND/ORA/IMP



ADI

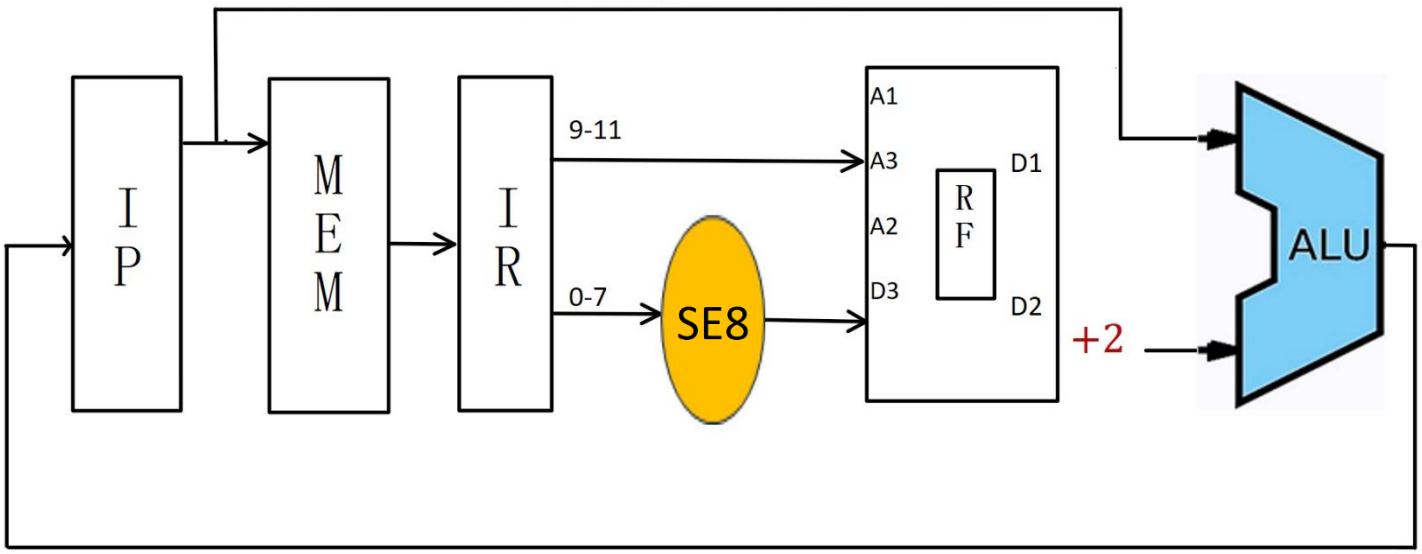


LHI

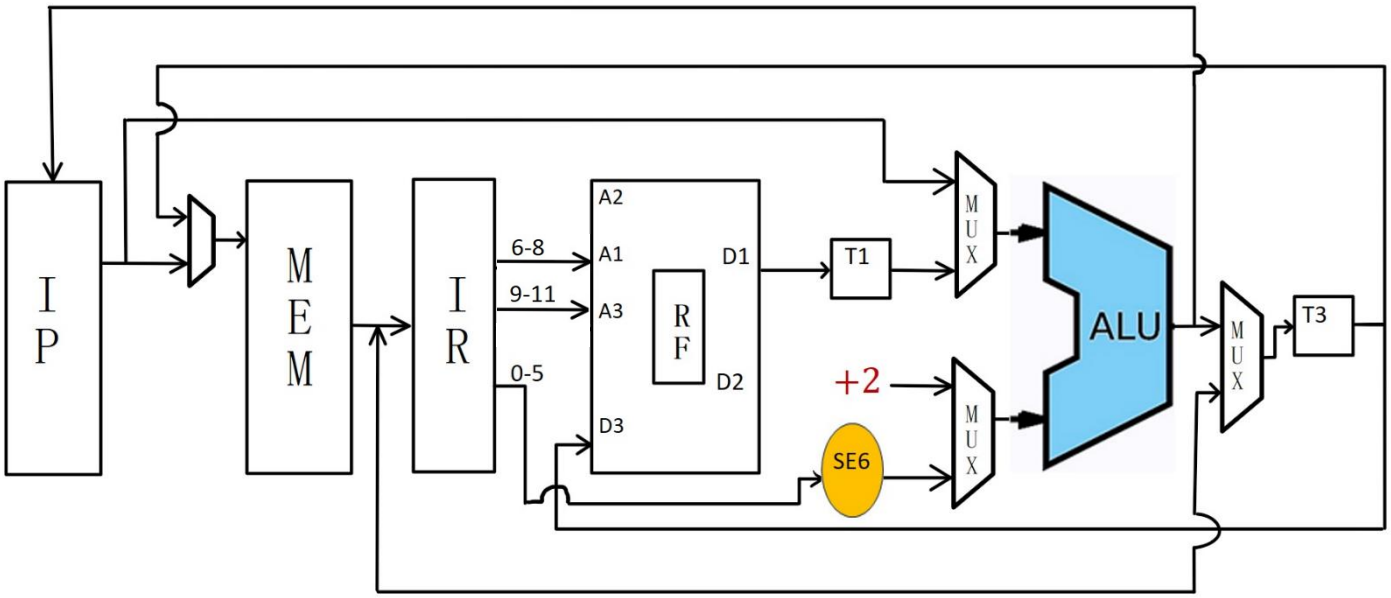


Components of IITB-CPU Design

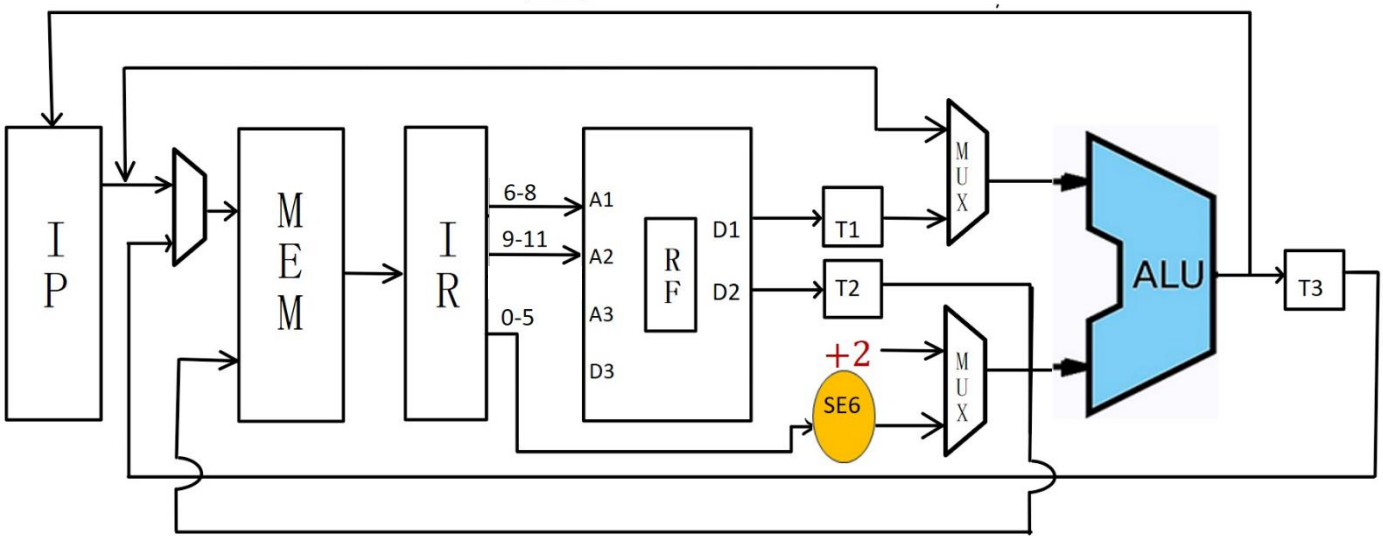
LLI



LW

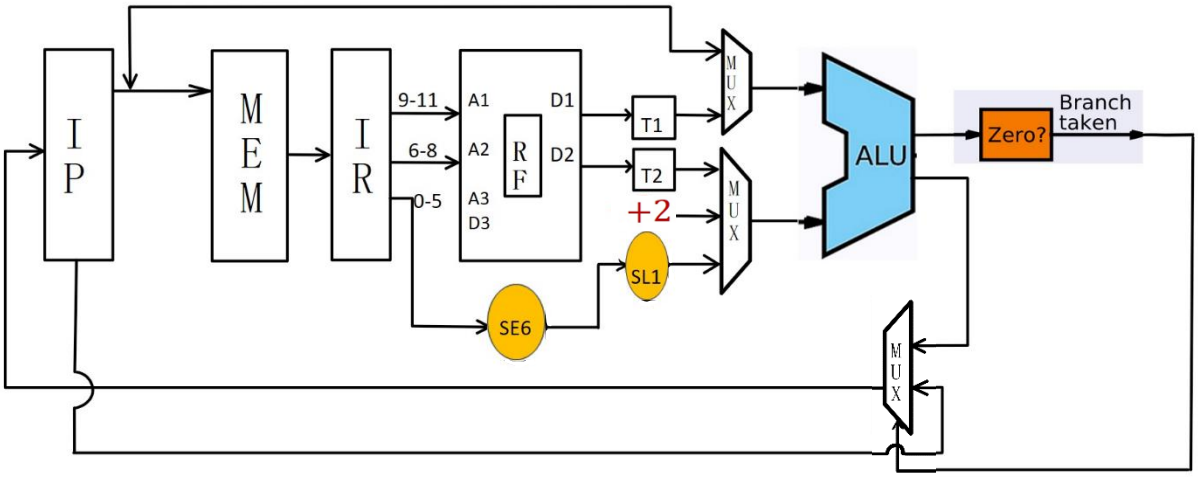


SW

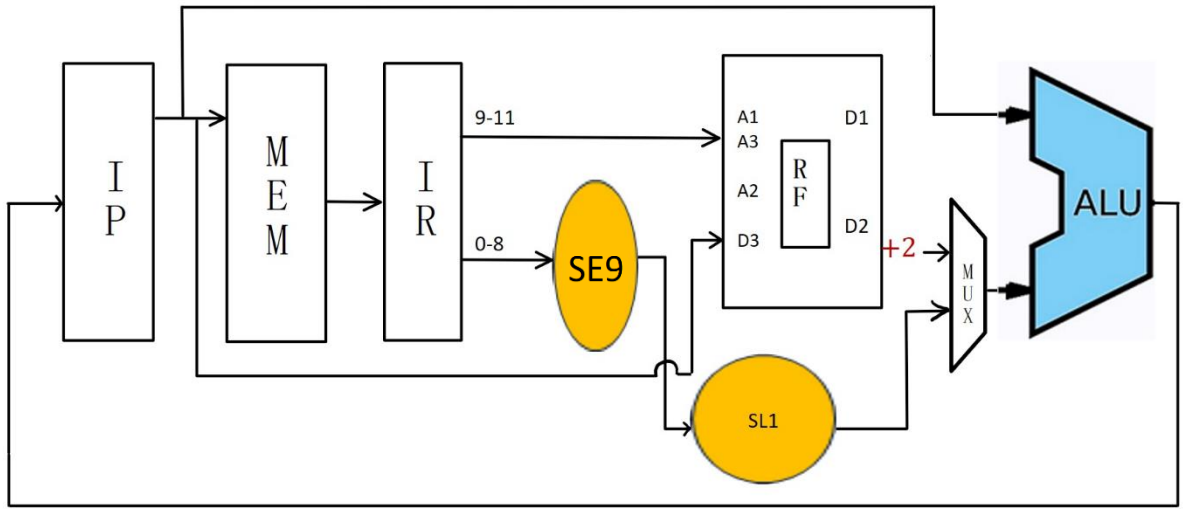


Components of IITB-CPU Design

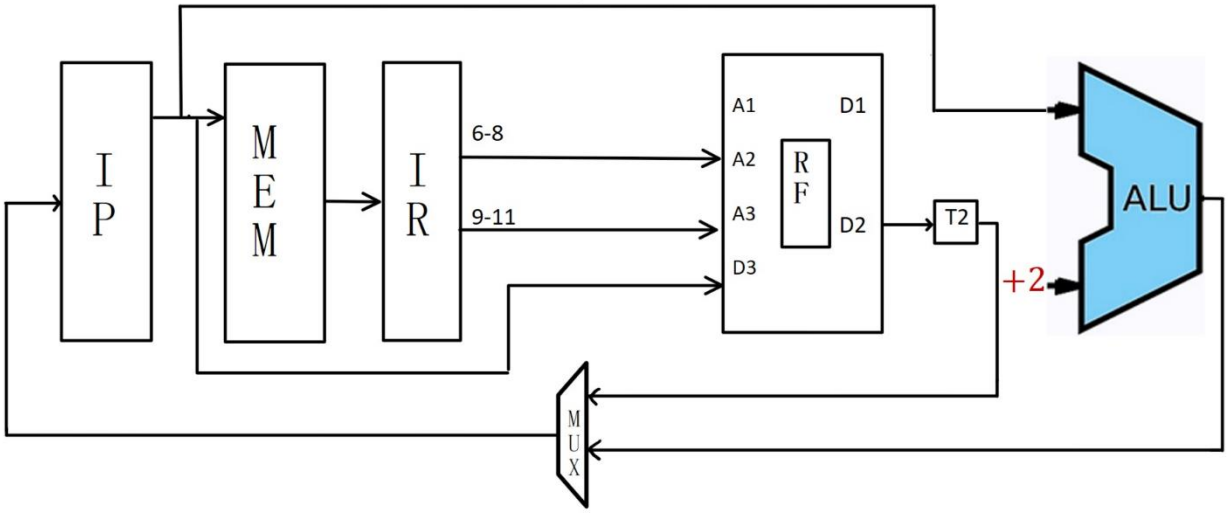
BEQ



JAL



JLR



Finite State Machines for Instructions of IIT-B CPU

Instruction-1: ADDITION (ADD)

S1: Fetch information and update instruction pointer.

S2: Understand and fetch operand.

S3: Execute operation.

S4: Update result.

S1- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

+2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S2- Understand and fetch operand.

IR_9-11 -> RF_A1

IR_6-8 -> RF_A2

RF_D1-> T1

RF_D2-> T2

T1_WRITE

T2_WRITE

S3- Execute operation.

T1 -> ALU_A

T2 -> ALU_B

ALU_C-> T3

ADD

T3_WRITE

S4- Update result.

T3 -> RF_D3

IR_3-5 -> RF_A3

RF_WRITE

Instruction-2: SUBTRACT (SUB)

S5: Fetch information and update instruction pointer.

S6: Understand and fetch operand.

S7: Execute operation.

S8: Update result.

S5- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

+2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S6- Understand and fetch operand.

IR_9-11 -> RF_A1

IR_6-8 -> RF_A2

RF_D1-> T1

RF_D2-> T2

T1_WRITE

T2_WRITE

S7- Execute operation.

T1 -> ALU_A

T2 -> ALU_B

ALU_C-> T3

SUB

T3_WRITE

S8- Update result.

T3 -> RF_D3

IR_3-5 -> RF_A3

RF_WRITE

Instruction-3: MULTIPLY (MUL)

S9: Fetch information and update instruction pointer.

S10: Understand and fetch operand.

S11: Execute operation.

S12: Update result.

S9- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

Instruction Pointer +2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S10- Understand and fetch operand.

IR_9-11 -> RF_A1

IR_6-8 -> RF_A2

RF_D1-> T1

RF_D2-> T2

T1_WRITE

T2_WRITE

S11- Execute operation.

T1 -> ALU_A

T2 -> ALU_B

ALU_C-> T3

MUL

T3_WRITE

S12- Update result.

T3 -> RF_D3

IR_3-5 -> RF_A3

RF_WRITE

Instruction-4: ADD IMMIDIATE (ADI)

S13: Fetch information and update instruction pointer.

S14: Understand and fetch operand.

S15: Execute Addition Operation.

S16: Update Result.

S13- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

+2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S14- Understand and fetch operand.

IR_9-11 -> RF_A1

RF_D1-> T1

T1_WRITE

S15- Execute Addition Operation.

T1 -> ALU_A

IR_0-5 -> SE6 -> ALU_B

ALU_C-> T3

ADD

T3_WRITE

S16-Update Result.

T3 -> RF_D3

IR_6-8 -> RF_A3

RF_WRITE

Instruction-5: LOGICAL AND (AND)

S17: Fetch information and update instruction pointer.

S18: Understand and fetch operand.

S19: Execute operation.

S20: Update result.

S17- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

+2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S18- Understand and fetch operand.

IR_9-11 -> RF_A1

IR_6-8 -> RF_A2

RF_D1-> T1

RF_D2-> T2

T1_WRITE

T2_WRITE

S19- Execute operation.

T1 -> ALU_A

T2 -> ALU_B

ALU_C-> T3

AND

T3_WRITE

S20- Update result.

T3 -> RF_D3

IR_3-5 -> RF_A3

RF_WRITE

Instruction-6: LOGICAL OR (ORA)

S21: Fetch information and update instruction pointer.

S22: Understand and fetch operand.

S23: Execute operation.

S24: Update result.

S21- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

+2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S22- Understand and fetch operand.

IR_9-11 -> RF_A1

IR_6-8 -> RF_A2

RF_D1-> T1

RF_D2-> T2

T1_WRITE

T2_WRITE

S23- Execute operation.

T1 -> ALU_A

T2 -> ALU_B

ALU_C-> T3

OR

T3_WRITE

S24- Update result.

T3 -> RF_D3

IR_3-5 -> RF_A3

RF_WRITE

Instruction-7: LOGICAL IMPLICATION (IMP)

S25: Fetch information and update instruction pointer.

S26: Understand and fetch operand.

S27: Execute operation.

S28: Update result.

S25- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

+2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S26- Understand and fetch operand.

IR_9-11 -> RF_A1

IR_6-8 -> RF_A2

RF_D1-> T1

RF_D2-> T2

T1_WRITE

T2_WRITE

S27- Execute operation.

T1 -> ALU_A

T2 -> ALU_B

ALU_C-> T3

IMP

T3_WRITE

S28- Update result.

T3 -> RF_D3

IR_3-5 -> RF_A3

RF_WRITE

Instruction-8: LOAD HIGHER IMMIDIATE (LHI)

S29: Fetch information and update instruction pointer.

S30: Understand and fetch operand.

S31: Store Require value in Specified Location

S29- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

+2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S30- Understand and fetch operand.

EMPTY STATE

S31- Store Required Value in Specified Location.

IR_0-7 -> SE8Right -> RF_D3

IR_9-11 -> RF_A3

RF_WRITE

Instruction-9: LOAD LOWER IMMIDIATE (LLI)

S32: Fetch information and update instruction pointer.

S33: Understand and fetch operand.

S34: Store Require value in Specified Location

S32- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

+2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S33- Understand and fetch operand.

EMPTY STATE

S34- Store Required Value in Specified Location.

IR_0-7 -> SE8Left -> RF_D3

IR_9-11 -> RF_A3

RF_WRITE

Instruction-10: LOAD (LW)

S35: Fetch information and update instruction pointer.

S36: Understand and fetch operand.

S37: Compute address.

S38: Read Memory.

S39: Update Register.

S35- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

+2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S36- Understand and fetch operand.

IR_6-8 -> RF_A1

RF_D1-> T1

T1_WRITE

S37- Compute Address.

T1 -> ALU_A

IR_0-5 -> SE6 -> ALU_B

ALU_C-> T3

ADD

T3_WRITE

S38- Read Memory.

T3 -> Memory.Address

Memory.Data -> T3

MEM_READ

T3_WRITE

S39-Update Register.

T3 -> RF_D3

IR_9-11 -> RF_A3

RF_WRITE

Instruction-11: STORE (SW)

S40: Fetch information and update instruction pointer.

S41: Understand instruction and read operand.

S42: Compute address.

S43: Write Memory.

S40- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

+2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S41- Understand instruction and read operand.

IR_6-8 -> RF_A1

RF_D1-> T1

IR_9-11 -> RF_A2

RF_D2 -> T2

T1_WRITE

T2_WRITE

S42- Compute Address.

T1 -> ALU_A

IR_0-5 -> SE6 -> ALU_B

ALU_C-> T3

ADD

T3_WRITE

S43- Read Memory.

T3 -> Memory.Address

T2 -> Memory.Data

MEM_WRITE

Instruction-12: BRANCH IF EQUIVALENT (BEQ)

S44: Fetch information and update instruction pointer.

S45: Understand and fetch operand.

S46: Compute Z (Compare R1 and R2, when R1-R2=0 then Z=1).

S47: If Z==1, then IP=IP+2+(IMM6)×2.

S44- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

+2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S45- Understand instruction and read operand.

IR_9-11 -> RF_A1

IR_6-8 -> RF_A2

RF_D1-> T1

RF_D2-> T2

T1_WRITE

T2_WRITE

S46- Compute Z (Compare R1 and R2, when R1-R2=0 then Z=1).

T1 -> ALU_A

T2 -> ALU_B

ALU_Z-> Z

SUB

Z_WRITE

S47- If Z==1, then IP=IP+2+(IMM6)×2.

IP -> ALU_A

IMM6 -> SE6 -> Left Shift by 1 -> ALU_B

If Z==1, ALU_C -> IP

Else IP -> IP

IP_WRITE

Instruction-13: JUMP AND LINK (JAL)

S48: Fetch information and update instruction pointer.

S49: Understand and fetch operand.

S50: Store Current Instruction Pointer.

S51: Compute IP and Update IP

S48- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

+2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S49- Understand instruction and read operand.

EMPTY STATE

S50- Store current instruction pointer.

IR_9-11 -> RF_A3

Instruction Pointer -> RF_D3

RF_WRITE

S51- Compute IP and Update IP

IP -> ALU_A

IR_0-8 -> SE9 -> Left shift by 1 -> ALU_B

ALU_C-> IP

IP_READ

IP_WRITE

Instruction-14: JUMP AND LINK TO REGISTER (JLR)

S52: Fetch information and update instruction pointer.

S53: Understand and fetch operand.

S54: Store Current Instruction Pointer.

S55: Update IP

S52- Fetch information and update instruction pointer.

Instruction Pointer -> Memory.Address

Memory.Data -> Instruction Register

Instruction Pointer -> ALU_A

+2 -> ALU_B

ALU_C -> Instruction Pointer

IP_READ

MEM_READ

ADD

IP_WRITE

S53- Understand instruction and read operand

IR_6-8-> RF_A2

RF_D1-> T2

T2_WRITE

S54- Store current instruction pointer.

IR_9-11 -> RF_A3

Instruction Pointer -> RF_D3

RF_WRITE

S55- Update IP

T2 -> IP

IP_WRITE

MUX Tables

MUX 1

B1	B0	OUTPUT SELECTED
0	0	*Do Nothing*
0	1	*Addition*
1	0	*Subtraction*
1	1	*IR 12_15*

MUX 2

B3	B2	OUTPUT SELECTED
0	0	*Do Nothing*
0	1	*IR 3_5*
1	0	*IR 6_8*
1	1	*IR 9_11*

MUX 3

B6	B5	B4	OUTPUT SELECTED
0	0	0	*Do Nothing*
0	0	1	*T3*
0	1	0	*IR 0_7 → SE8Right*
0	1	1	*IR 0_7 → SE8Left*
1	0	0	*IP*

MUX 4

B8	B7	OUTPUT SELECTED
0	0	*Do Nothing*
0	1	*IP*
1	0	*T1*

MUX 5

B11	B10	B9	OUTPUT SELECTED
0	0	0	*Do Nothing*
0	0	1	*+2*
0	1	0	*T2*

0	1	1	*SE6*
1	0	0	*SE6 → LS1*
1	0	1	*SE9*

MUX 6

B13	B12	OUTPUT SELECTED
0	0	*Do Nothing*
0	1	*IP*
1	0	*T3*

MUX 7

B15	B14	OUTPUT SELECTED
0	0	*Do Nothing*
0	1	*ALU_C*
1	0	*Memory Data*

MUX 8

B17	B16	OUTPUT SELECTED
0	0	*Do Nothing*
0	1	*ALU_C*
1	0	*BEQ*
1	1	*T2*

MUX 9

B15	OUTPUT SELECTED
0	*IR 9_11*
1	*IR 6_8*

MUX 10

B15	OUTPUT SELECTED
0	*IR 6_8*
1	*IR 9_11*

Merging the entire CPU

We have 14 instructions → 55 states.

- S1, S5, S9, S13, S17, S21, S25, S29, S32, S35, S40, S44, S48 and S52 are potentially equivalent states. Let's replace these 14 states by M1.
- S2, S6, S10, S14, S18, S22, S26, S30, S33, S45, S49 and S53 are potentially equivalent states. Let's replace these 14 states by M2.
- S3, S7, S11, S19, S23 and S27 are potentially equivalent states. Let's replace these 6 states by M3.
- S4, S8, S12, S20, S24 and S28 are potentially equivalent states. Let's replace these 6 states by M4.
- S15, S37 and S42 are potentially equivalent states. Let's replace these 3 states by M5.
- Let's replace S38 by M6.
- Let's replace S39 by M7.
- S50 and S54 are potentially equivalent states. Let's replace these 2 states by M8.
- Let's replace S31 by M9.
- Let's replace S34 by M10.
- Let's replace S43 by M11.
- Let's replace S46 by M12.
- Let's replace S47 by M13.
- Let's replace S51 by M14.
- Let's replace S55 by M15.
- Let's replace S16 by M16.
- Let's replace S41 and S36 by M17.

X to	B19	B18	B17	B16	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
M1	0	0	0	1	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0	1
M2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M3	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	1	1
M4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
M5	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0
M6	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
M7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
M8	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0
M9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0
M10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0
M11	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
M12	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0
M13	0	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1
M14	0	0	0	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	1
M15	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
M17	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

M1- Fetch information and update instruction pointer (S1, S5, S9, S13, S17, S21, S25, S29, S32, S35, S40, S44, S48, S52)

M2- Understand and fetch operand (S2, S6, S10, S14, S18, S22, S26, S30, S33, S45, S49, S53)

M3- Execute operation (S3, S7, S11, S19, S23, S27)

M4- Update result (S4, S8, S12, S20, S24, S28)

M5- Execute addition operation (S15, S37, S42)

M6- Read memory (S38)

M7- Update register (S39)

M8- Store current instruction pointer (S50, S54)

M9- Store required value in specified location (S31)

M10- Store required value in specified location (S34)

M11- read memory (S43)

M12- Compute Z (S46)

M13- If z == 1, then ip = ip+2+(imm6)×2 (S47)

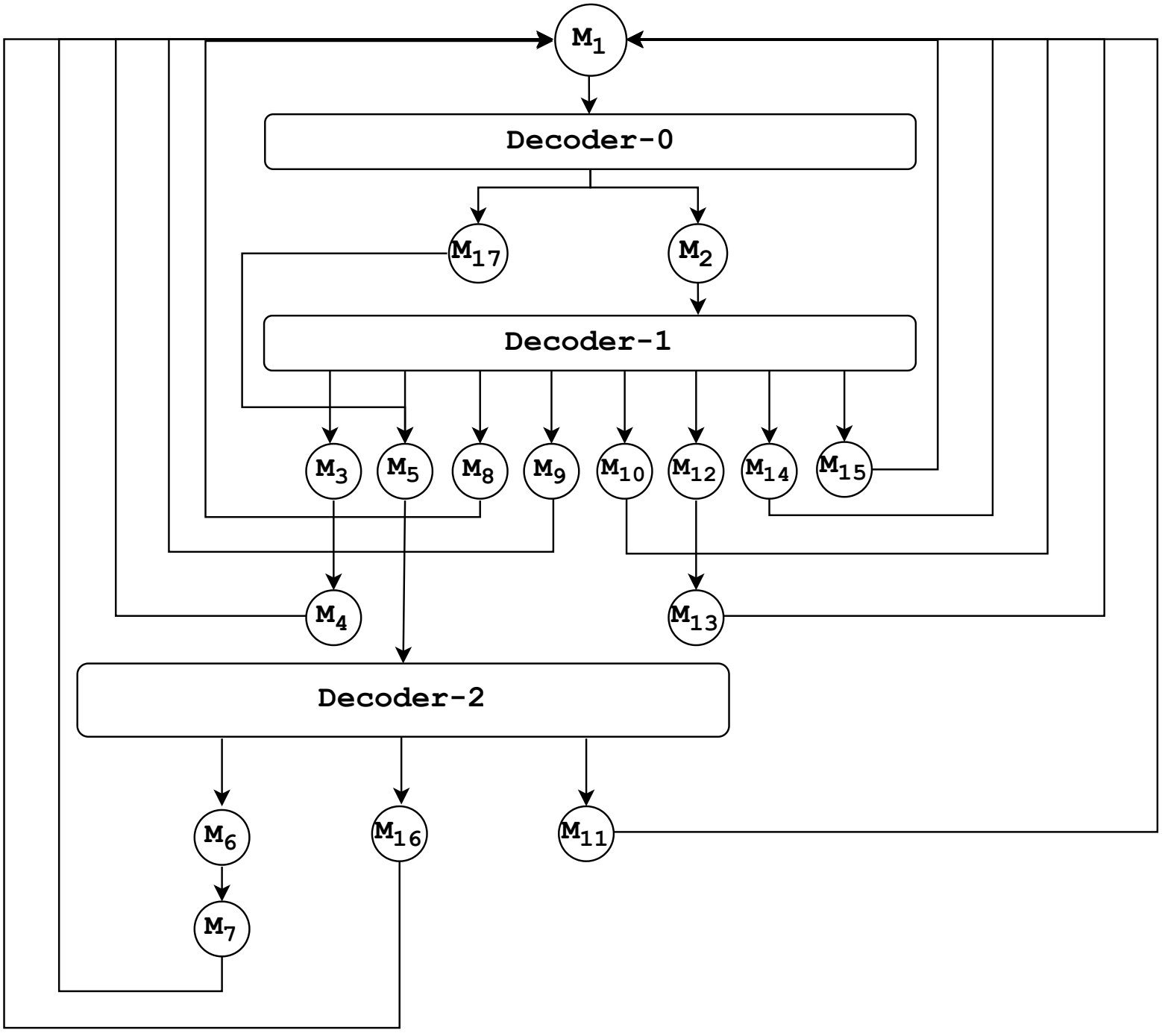
M14- Compute and update IP (S51)

M15- Update IP (S55)

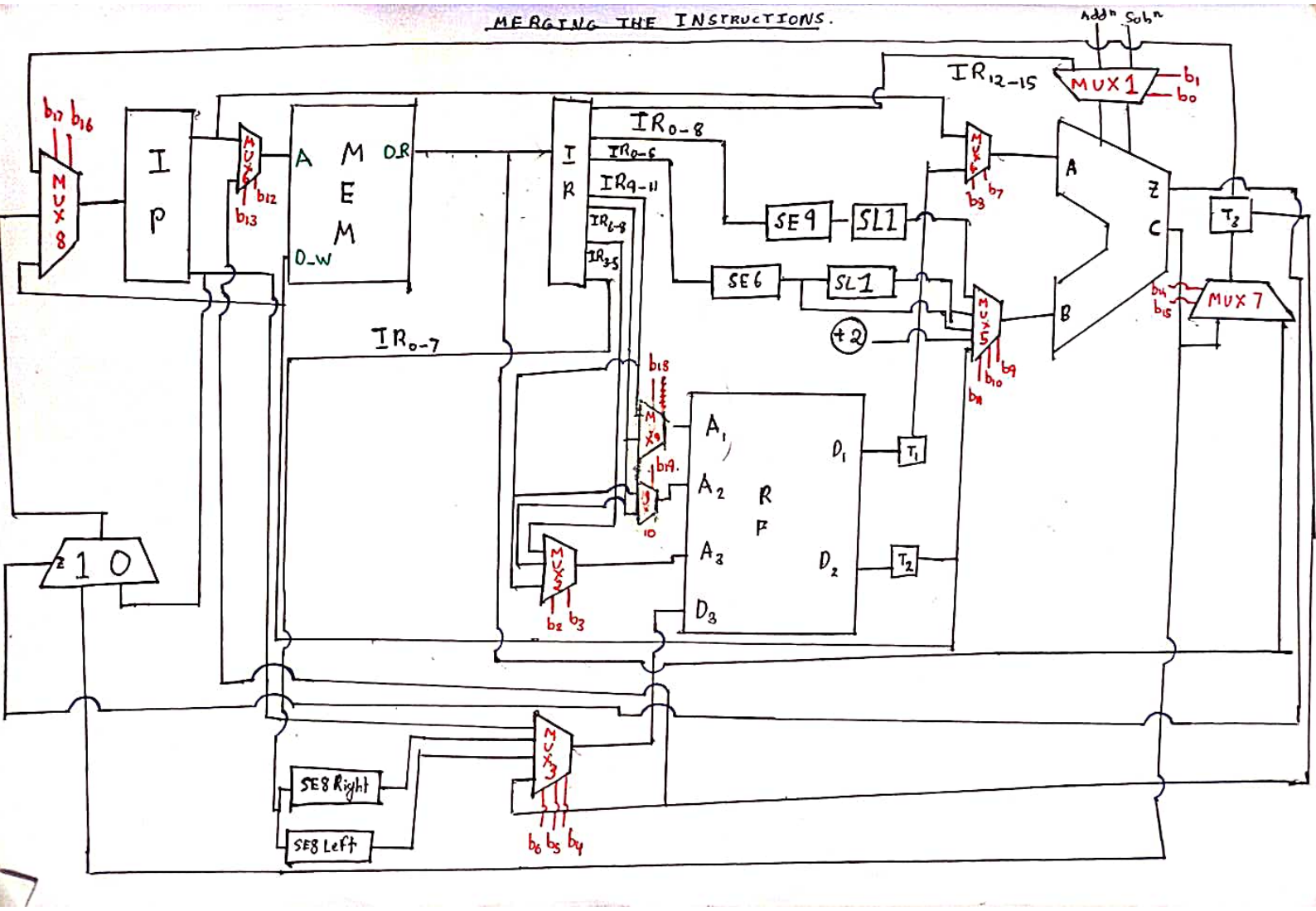
M16- Update result (S16)

M17- Understand and fetch operand (S36, S41)

State Diagram



Datapath


MERGING THE INSTRUCTIONS.
Addⁿ Subⁿ
IR12-15
MUX1
b1 b0
b17 b16
MUX 8
I P
MUX
b12 b13
A MEM DR
D_W
IR
IR0-8
IR0-5
IR9-11
IR6-8
IR3-5
IR0-7
SE9
SL1
SE6
SL1
+2
MUX
b3 b7
MUX
b10 b9 b8
A
B
Z
C
T2
MUX 7
b4 b15
b18
MUX
b19
10
A1 A2 A3 D3
R F
D1 D2
T1 T2
MUX
b2 b3
2 1 0
MUX 3
b6 b5 b4
SE8 Right
SE8 Left