CS 230 Project

Team Members
Gowri Sriya Mannepalli - 200050043
Gundabathula Sasank - 200050045
Mattapally Varun - 200050073
R Rakesh Kumar - 200050120

Design

The design consists of a 16-bit architecture with 8 registers. It has 8 general purpose registers, labeled from R0 to R7. PC points to the next instruction. All addresses are short word addresses (i.e. address 0 corresponds to the first two bytes of main memory, address 1 corresponds to the second two bytes of main memory, etc.).This architecture uses a condition code register that has two flags Carry flag (c) and Zero flag (z). The system takes clock and reset as inputs.

Basic overview of a process

As we have stated earlier, PC points to the next instruction in the memory. When a process has to be executed, the instruction is fetched from the instructions register. The operation code is extracted and the control is directed to perform that particular operation. For example, say the instruction is to ADD A and B. We direct the control/ flow to the ALU unit to perform the operation.

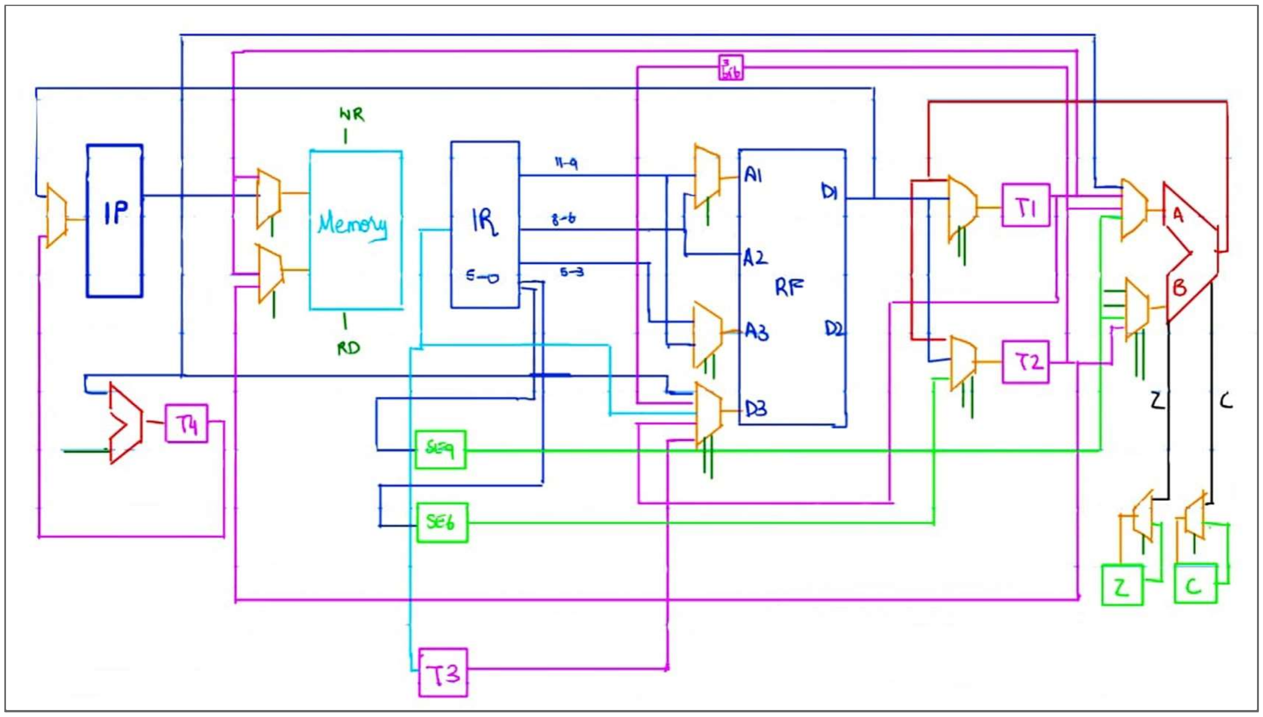
Execution of an Instruction

Every instruction set is implemented as follows:

- Instruction Fetch (IF): We fetch the current instruction from the memory and update PC to PC+1.
- Instruction Decode (ID): We parse the instruction in this step. We load the registers that are mentioned in the instruction and calculate the required information to perform the operation. We also double-check the value of the control units when reading/writing to the register to make sure that the operation of reading/writing the data was performed.
- Execution (EX): The ALU operates on the operands prepared in the prior cycle. For example, suppose a JUMP operation has to be performed and the final position is given relative to another position, then we calculate the absolute value of the final position before proceeding. A similar calculation will have to be done for BEQ operation. The following 3 operations are performed based on the instruction type:

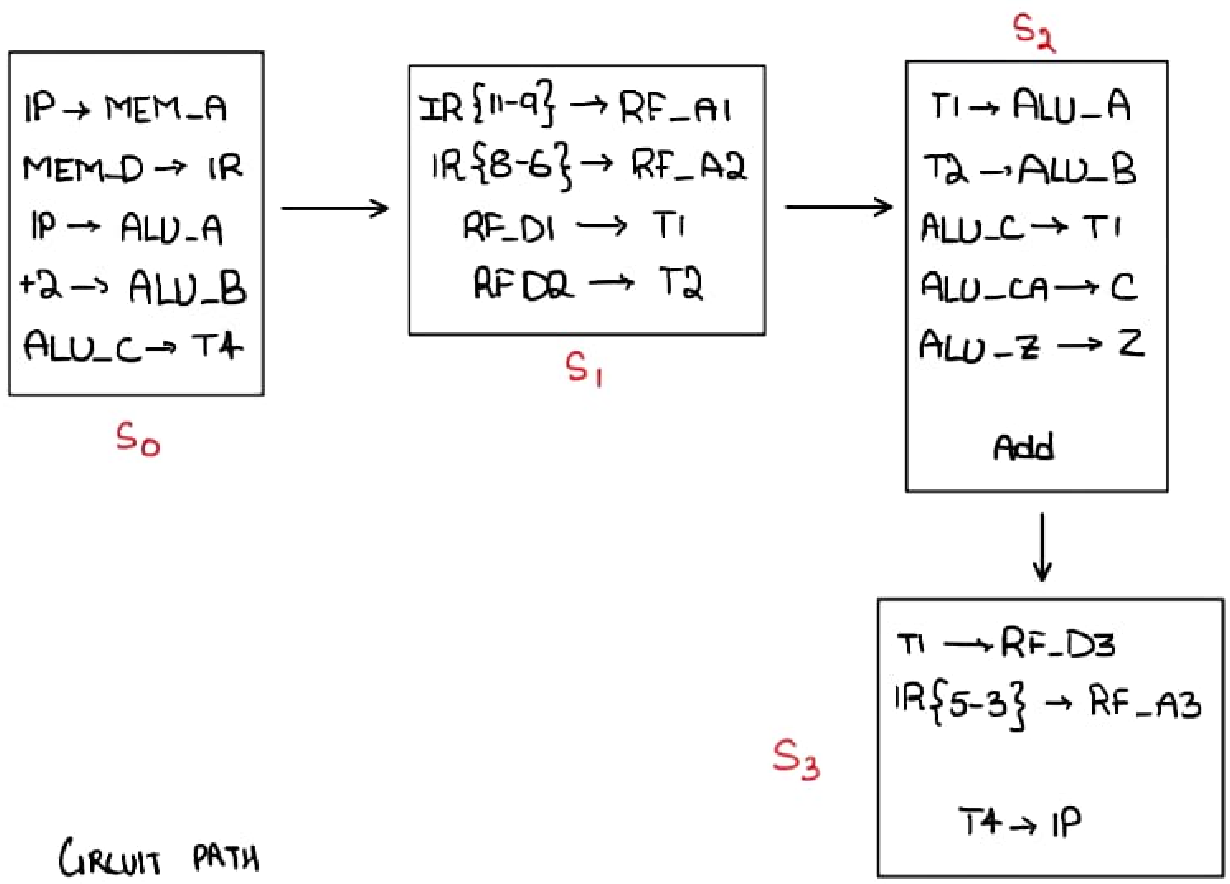
    - ❖ Memory reference: The ALU adds the base register and the offset to form the effective address.
    - ❖ Register-Register ALU instruction: The ALU performs the operation specified by the ALU opcode on the values read from the register file.
    - ❖ Register-Immediate ALU instruction: The ALU performs the operation specified by the ALU opcode on the first value read from the register file and the sign-extended immediate.
- Memory Access (MEM): If the instruction is a load, memory does a read using the effective address computed in the previous cycle. If it is a store, then the memory writes the data from the second register read from the register file using the effective address.
- Write-back cycle (WB): Write the result into the register file, whether it comes from the memory system (for a load) or from the ALU (for an ALU instruction).

General Data Path

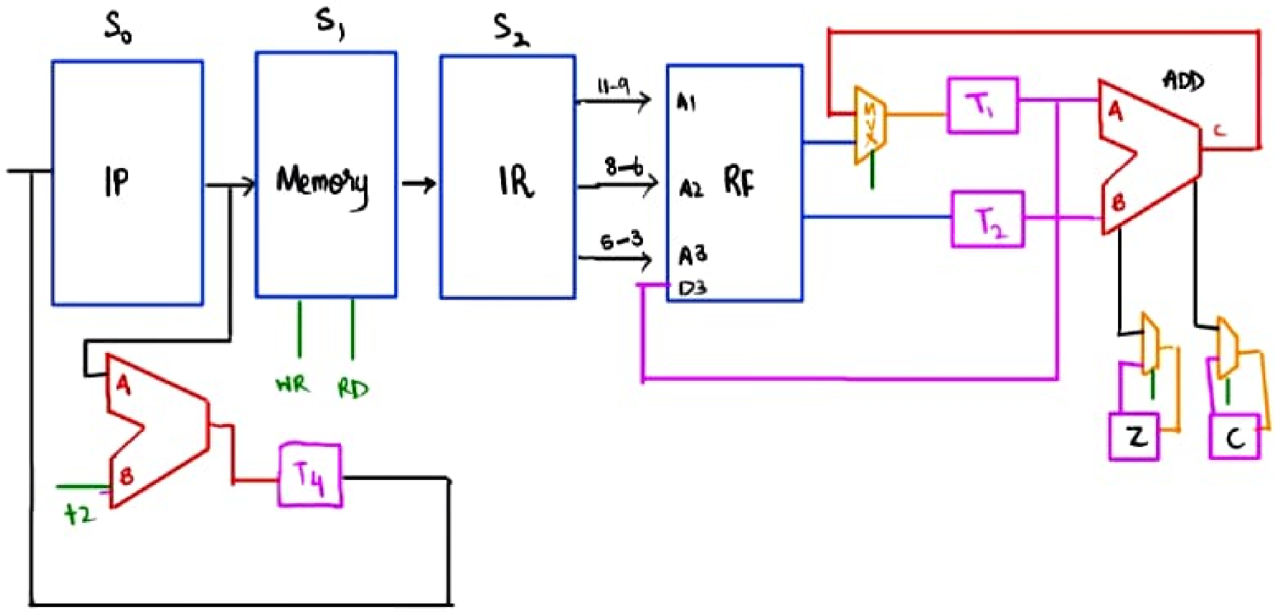


The above diagram that summarizes the data path and is color coded. The ALU and its outputs are red. The temporary registers and their outputs are pink. MUXes are in orange color.

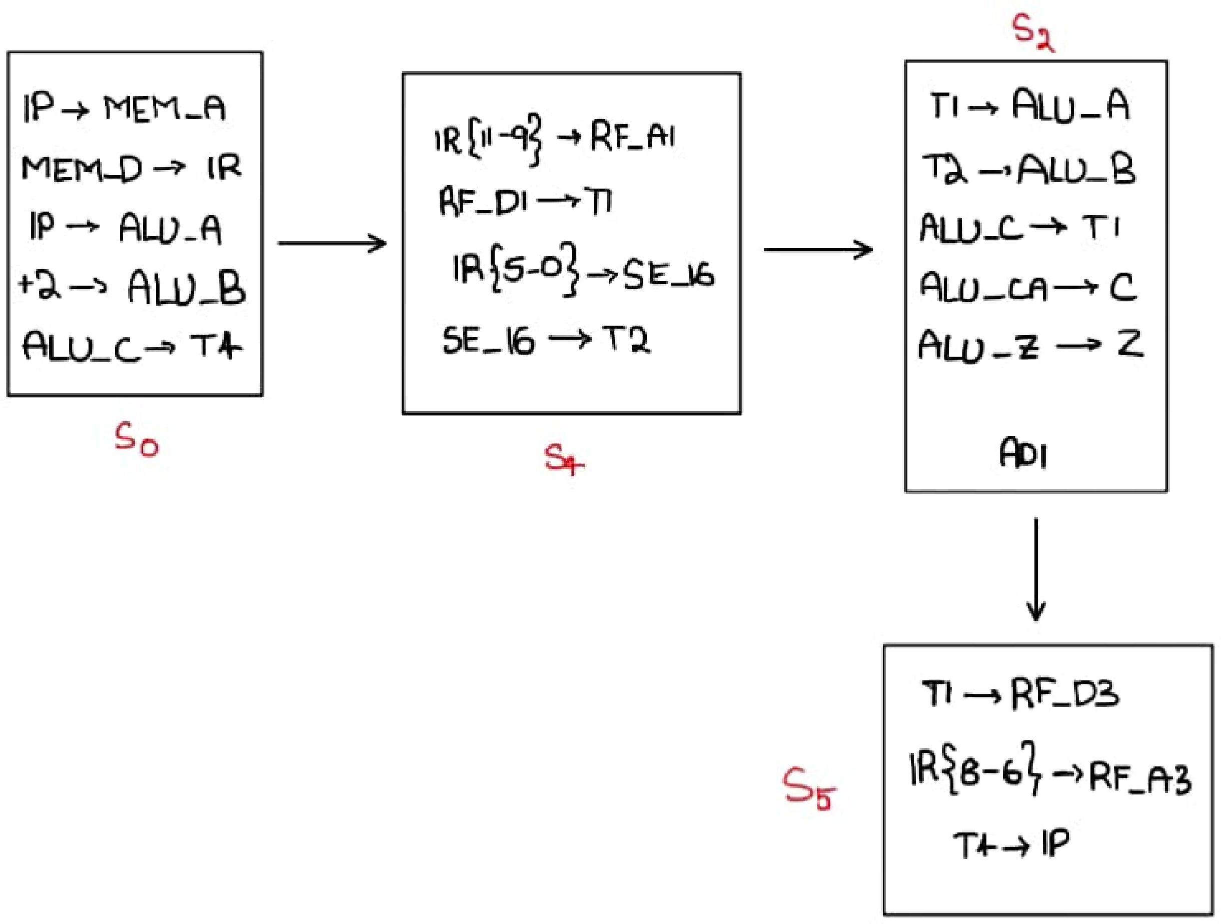
ADD, ADC, ADZ



Circuit Path



ADI



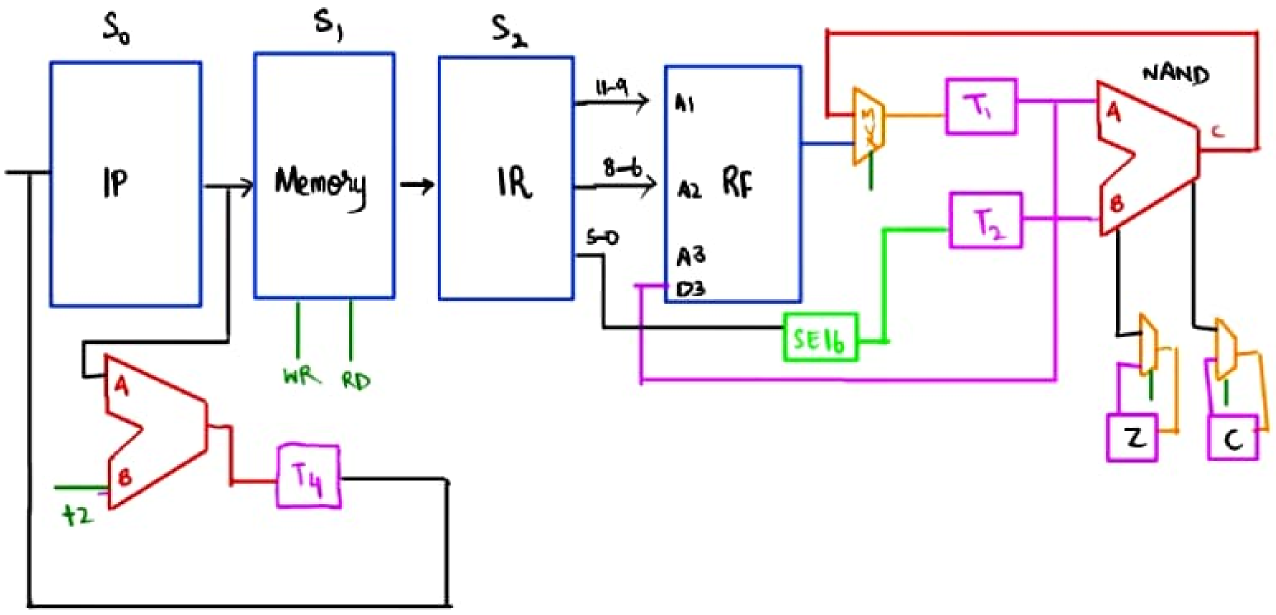
S_0
IP → MEM_A
MEM_D → IR
IP → ALU_A
+2 → ALU_B
ALU_C → T4

S_4
IR{11-9} → RF_A1
RF_D1 → T1
IR{5-0} → SE_16
SE_16 → T2

S_2
T1 → ALU_A
T2 → ALU_B
ALU_C → T1
ALU_CA → C
ALU_Z → Z
ADI

S_5
T1 → RF_D3
IR{8-6} → RF_A3
T4 → IP

CIRCUIT PATH



NDU, NDC, NDZ

S_0

IP → MEM_A
MEM_D → IR
IP → ALU_A
+2 → ALU_B
ALU_C → T4

→

S_1

IR{11-9} → RF_A1
IR{8-6} → RF_A2
RF_D1 → T1
RF D2 → T2

→

S_2'

T1 → ALU_A
T2 → ALU_B
ALU_C → T1
ALU_CA → C
ALU_Z → Z

NAND

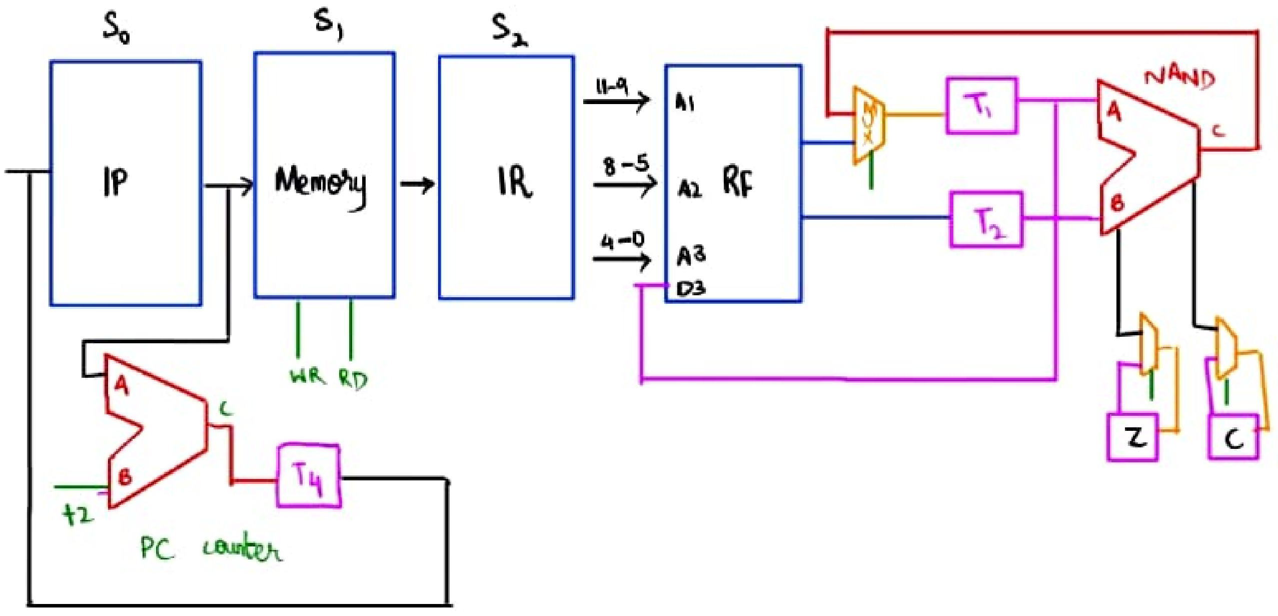
↓

S_3

T1 → RF_D3
IR{5-3} → RF_A3

T4 → IP

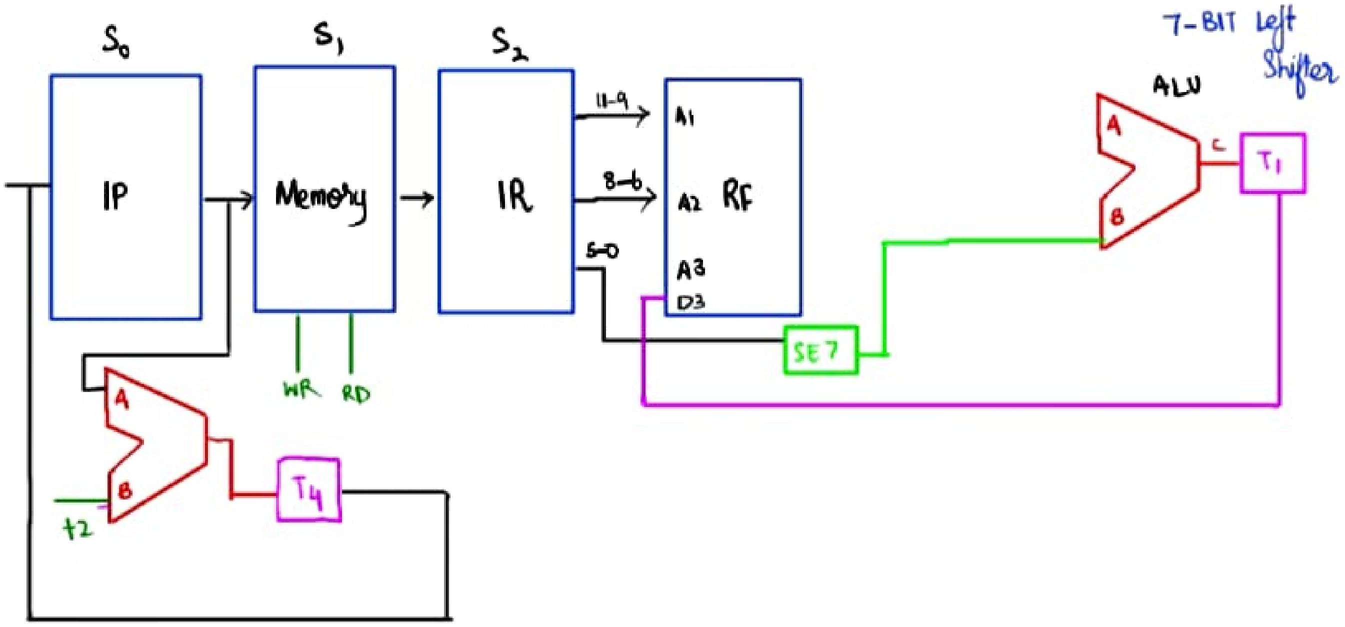
CIRCUIT PATH



LHI

$$\begin{tikzcd}
\boxed{\begin{array}{l} IP \to MEM\_A \\ MEM\_D \to IR \\ IP \to ALU\_A \\ +2 \to ALU\_B \\ ALU\_C \to T4 \end{array}} \arrow[r] & \boxed{\begin{array}{l} IR\{8\text{-}0\} \to SE\_16 \\ SE\_16 \to ALU\_A \\ ALU\_C \to T1 \end{array}} \arrow[r] & \boxed{\begin{array}{l} T1 \to RF\_D3 \\ IR\{11\text{-}9\} \to RF\_A3 \\ T4 \to IP \end{array}} \\
S_0 & S_6 & S_7
\end{tikzcd}$$

CIRCUIT PATH



LW

S_0

IP → MEM_A
MEM_D → IR
IP → ALU_A
+2 → ALU_B
ALU_C → T4

→

S_8

IR{8-6} → RF_A2
RF_D2 → T2
IR{5-0} → SE_16
SE16 → T1

→

S_2

T1 → ALU_A
T2 → ALU_B
ALU_C → T1

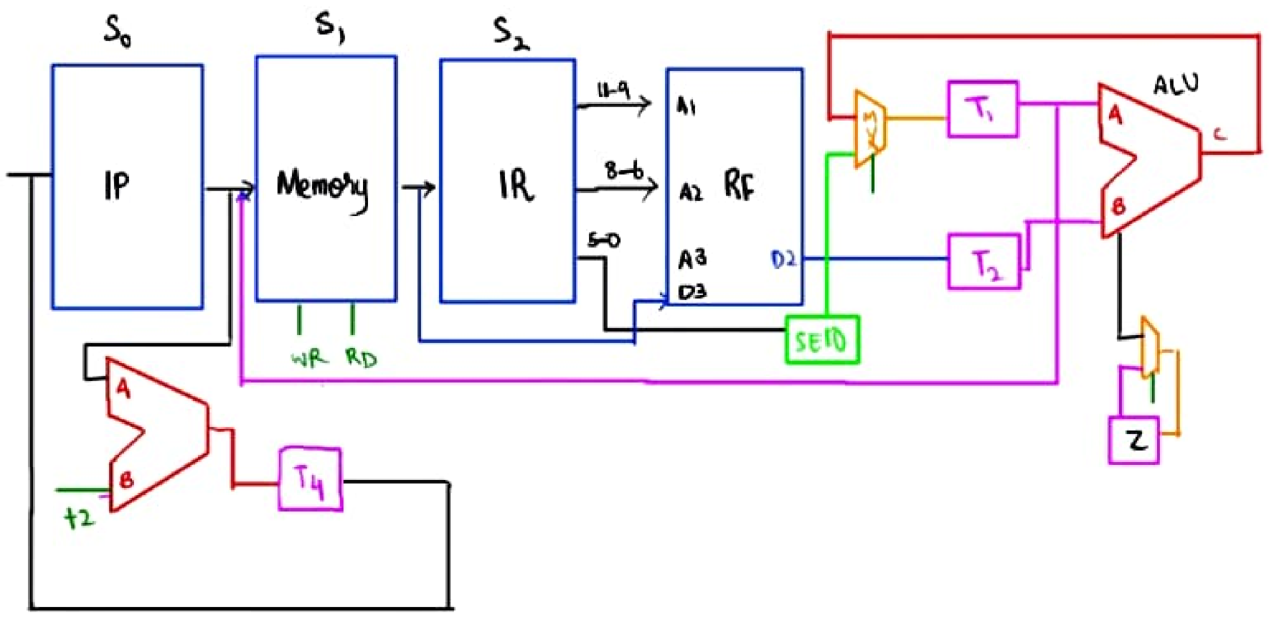
ALU_Z → Z

↓

S_9

T1 → MEM_A
MEM_D → RF_D3
IR{11-9} → RF_A3
T4 → IP

CIRCUIT PATH



SW

IP $\rightarrow$ MEM_A
MEM_D $\rightarrow$ IR
IP $\rightarrow$ ALU_A
+2 $\rightarrow$ ALU_B
ALU_C $\rightarrow$ T4

S_0

$\longrightarrow$

IR{8-6} $\rightarrow$ RF_A2
RF_D2 $\rightarrow$ T2
IR{5-0} $\rightarrow$ SE_16
SE_16 $\rightarrow$ T1

S_8

$\longrightarrow$

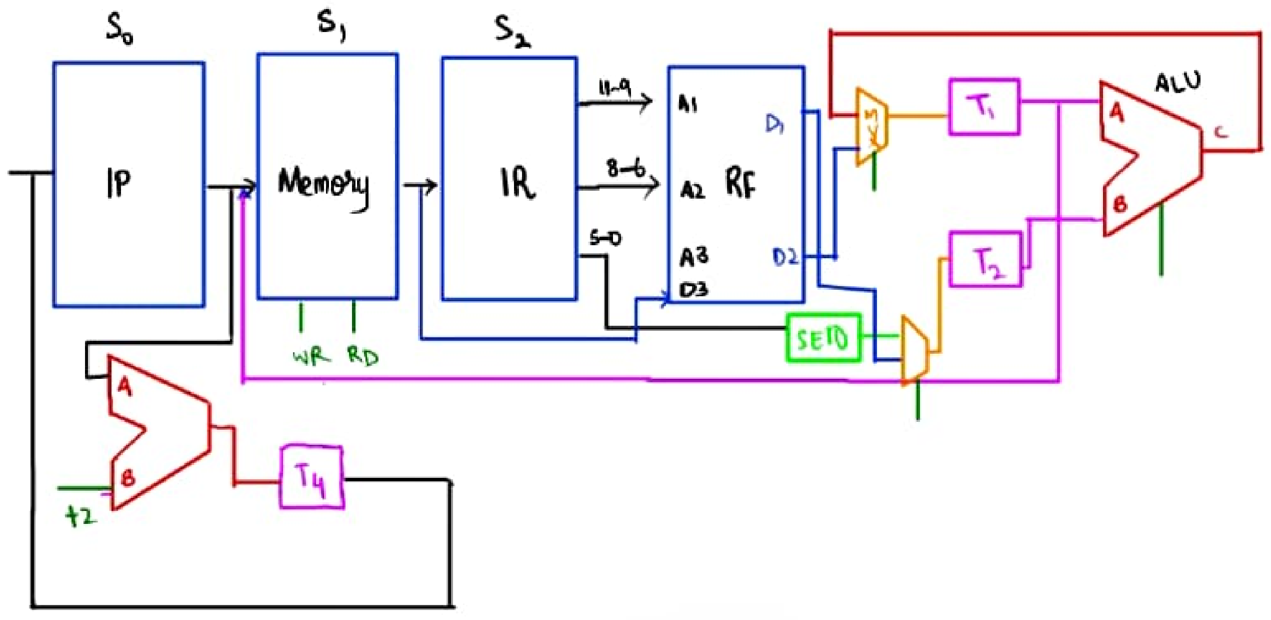
S_2

T2 $\rightarrow$ ALU_B
T1 $\rightarrow$ ALU_A
ALU_C $\rightarrow$ T1

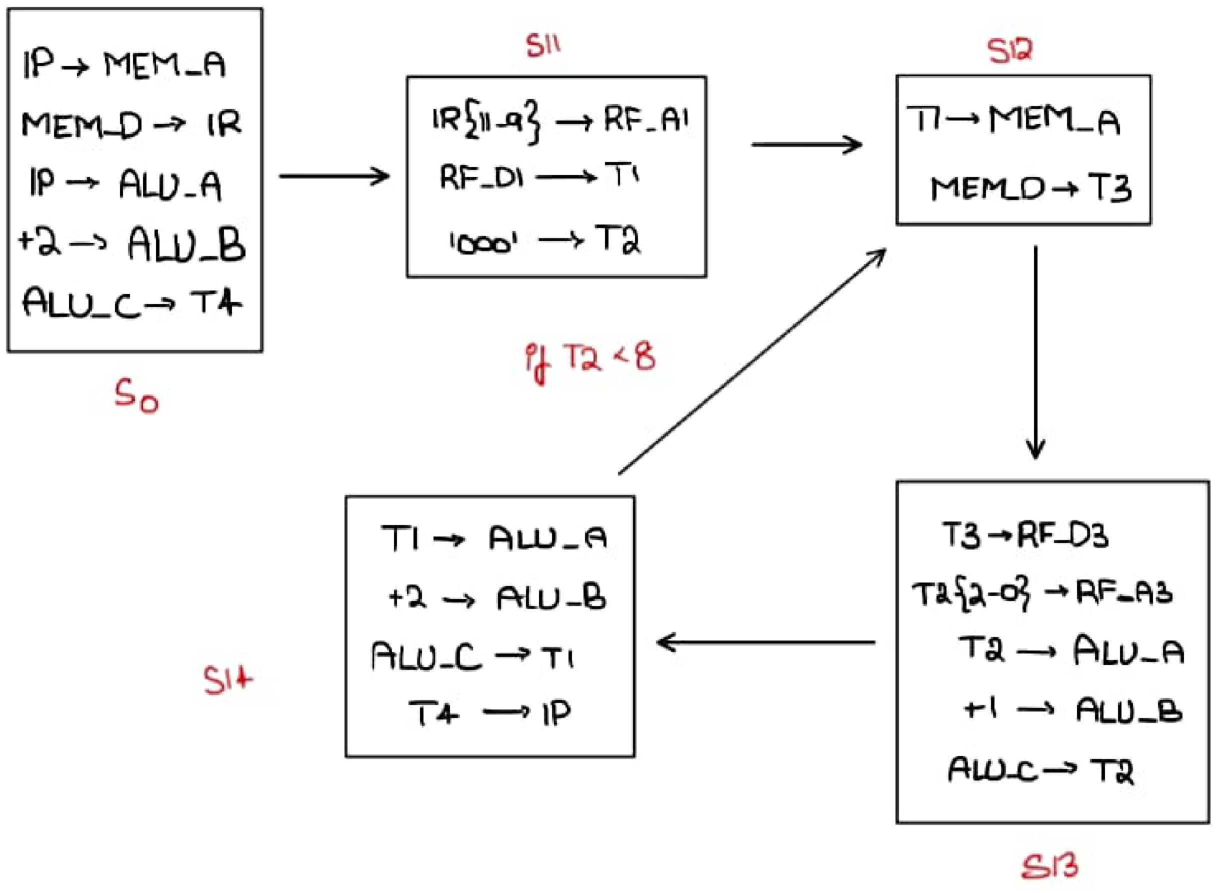
$\downarrow$

IR{11-9} $\rightarrow$ RF_A1
RF_D1 $\rightarrow$ T2
T1 $\rightarrow$ MEM_A
T2 $\rightarrow$ MEM_B
T4 $\rightarrow$ IP

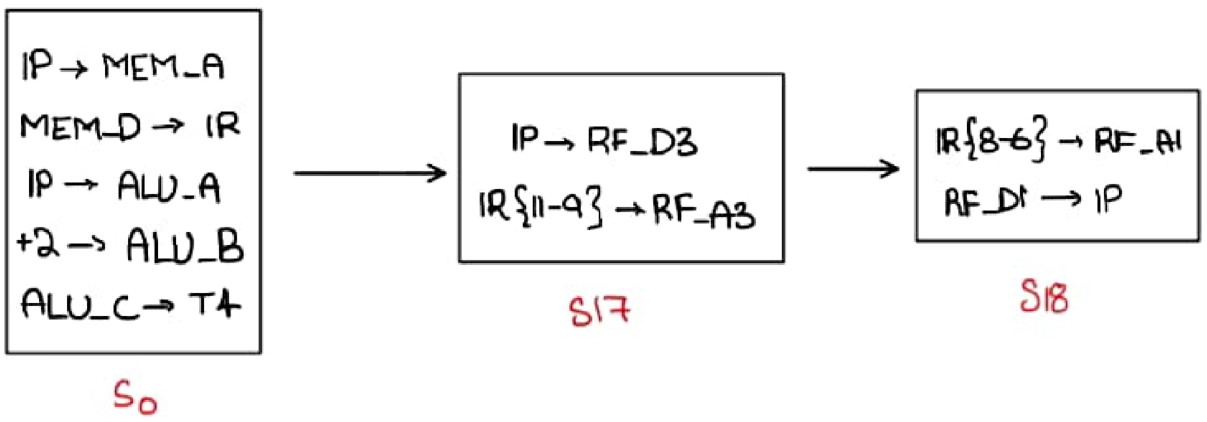
CIRCUIT PATH



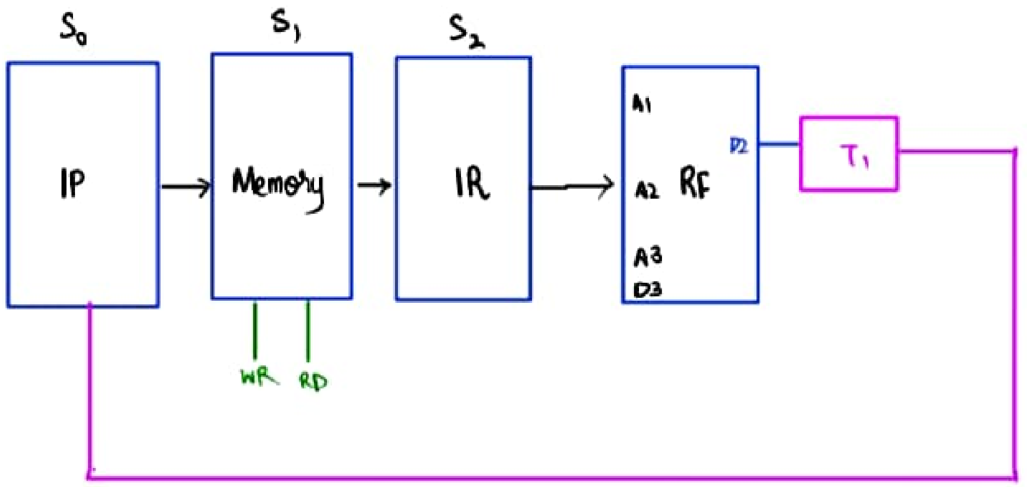
LA


IP → MEM_A
MEM_D → IR
IP → ALU_A
+2 → ALU_B
ALU_C → T4
S0
S11
IR{11-9} → RF_A1
RF_D1 → T1
'000' → T2
S12
T1 → MEM_A
MEM_D → T3
if T2 < 8
T3 → RF_D3
T2{2-0} → RF_A3
T2 → ALU_A
+1 → ALU_B
ALU_C → T2
S13
S14
T1 → ALU_A
+2 → ALU_B
ALU_C → T1
T4 → IP

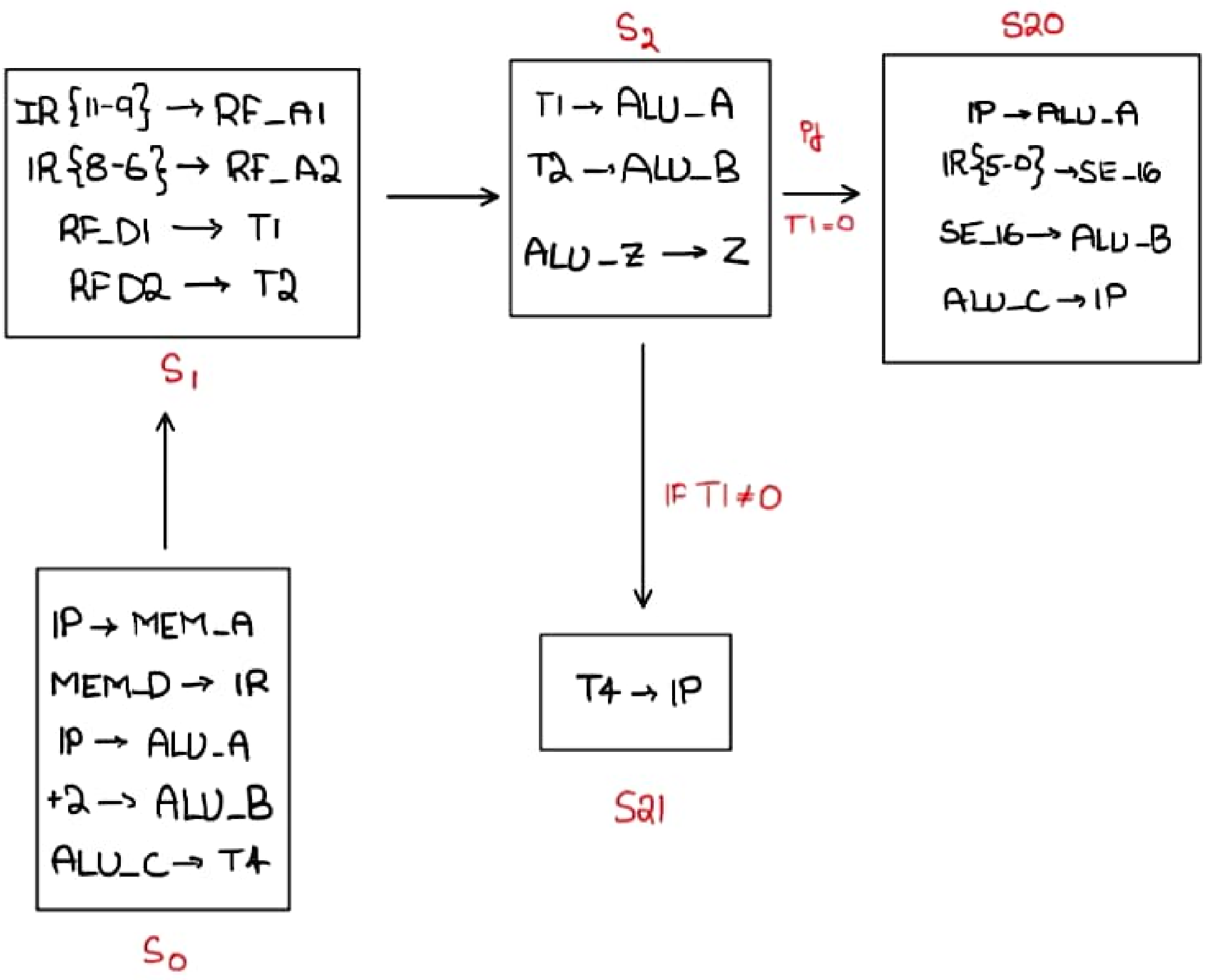
JLR



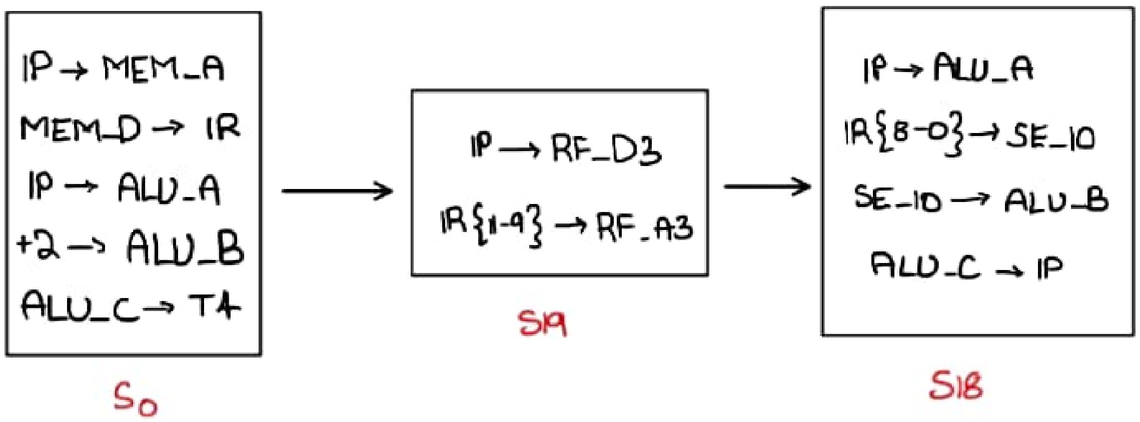
Circuit path

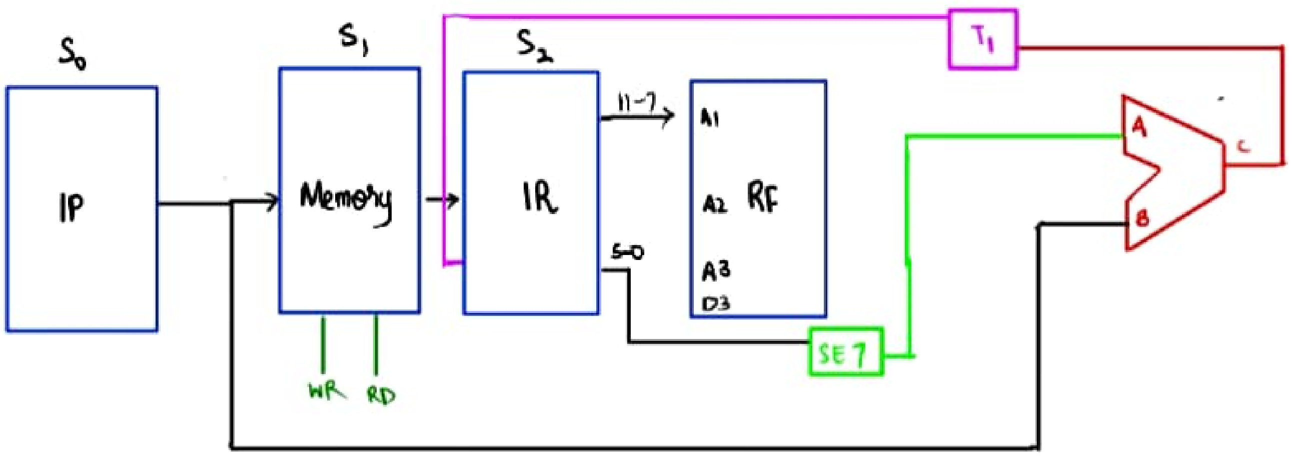


BEQ

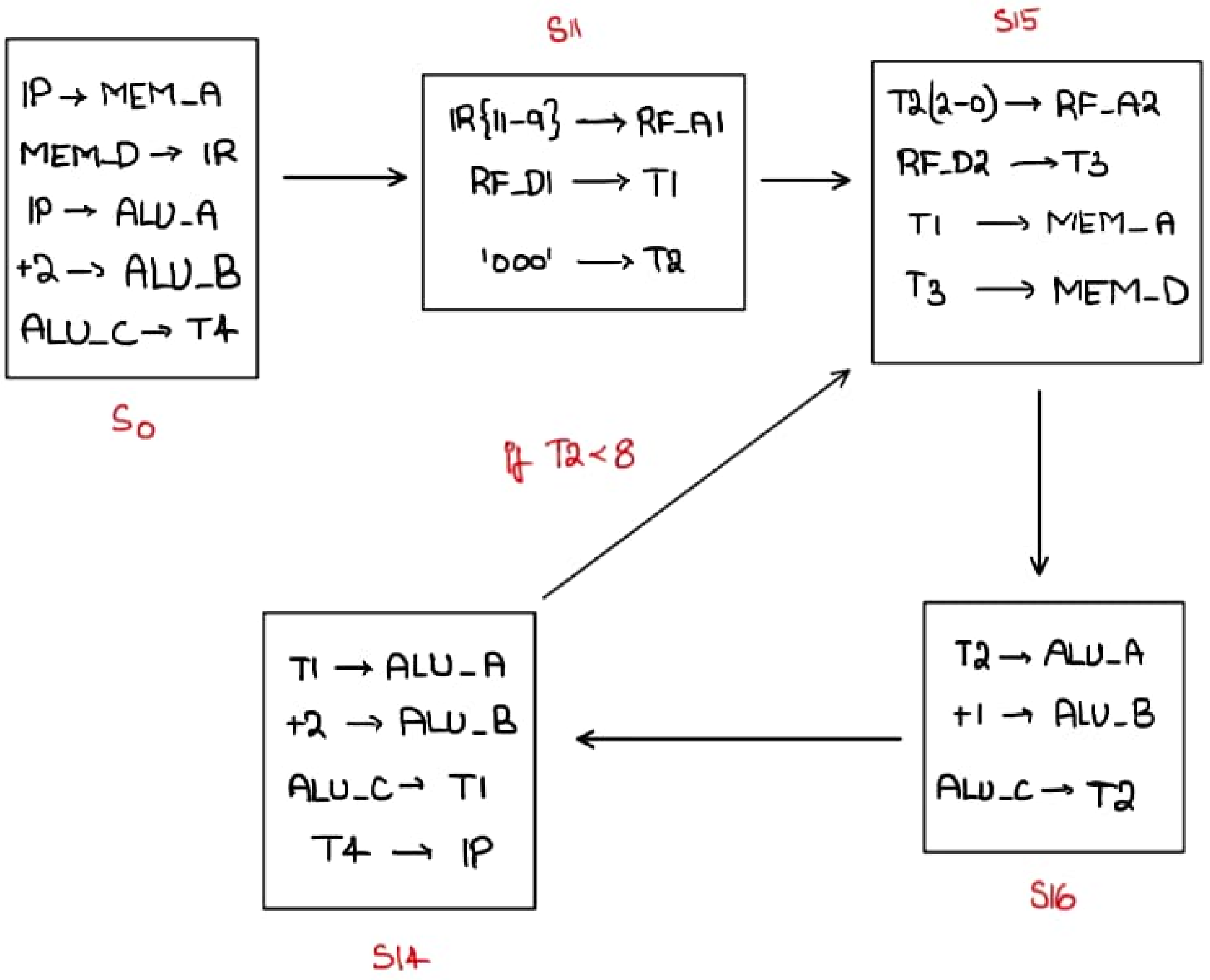

S2
S20
IR{11-9} → RF_A1
IR{8-6} → RF_A2
RF_D1 → T1
RF D2 → T2
T1 → ALU_A
T2 → ALU_B
ALU_Z → Z
if
T1=0
IP → ALU_A
IR{5-0} → SE_16
SE_16 → ALU_B
ALU_C → IP
S1
IF T1≠0
IP → MEM_A
MEM_D → IR
IP → ALU_A
+2 → ALU_B
ALU_C → T4
T4 → IP
S21
S0

JAL

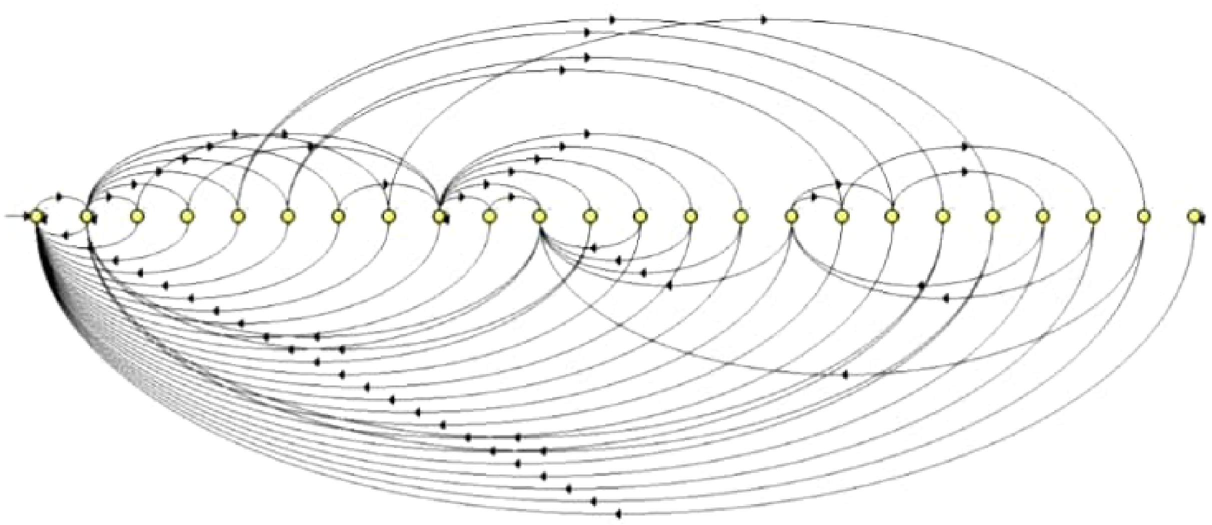

IP → MEM_A
MEM_D → IR
IP → ALU_A
+2 → ALU_B
ALU_C → T1
S0
IP → RF_D3
IR{11-9} → RF_A3
S19
IP → ALU_A
IR{8-0} → SE_10
SE_10 → ALU_B
ALU_C → IP
S18


S0
IP
S1
Memory
WR
RD
S2
IR
11-7
5-0
A1
A2
RF
A3
D3
SE7
T1
A
B
C

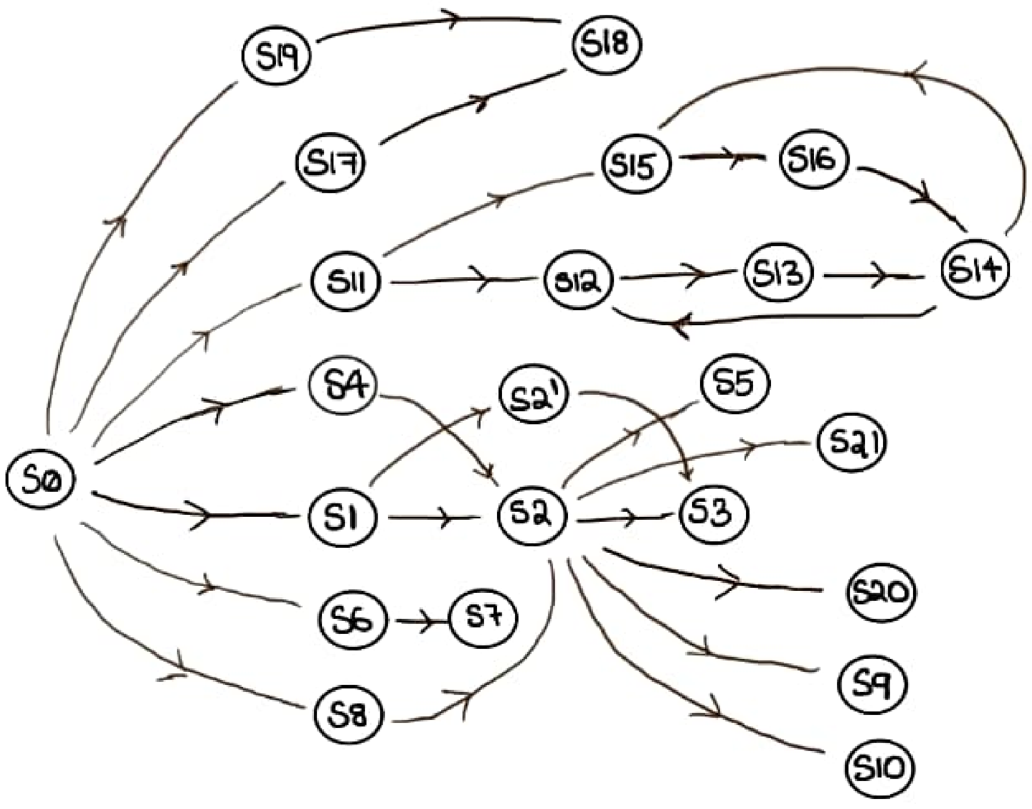
SA


IP → MEM_A
MEM_D → IR
IP → ALU_A
+2 → ALU_B
ALU_C → T4
S0
S11
IR{11-9} → RF_A1
RF_D1 → T1
'000' → T2
S15
T2(2-0) → RF_A2
RF_D2 → T3
T1 → MEM_A
T3 → MEM_D
If T2<8
T1 → ALU_A
+2 → ALU_B
ALU_C → T1
T4 → IP
S14
T2 → ALU_A
+1 → ALU_B
ALU_C → T2
S16

Netlist Viewer



State diagram


S0
S1
S2
S2'
S3
S4
S5
S6
S7
S8
S9
S10
S11
S12
S13
S14
S15
S16
S17
S18
S19
S20
S21