States

ADD/ADC/ADZ/ADA/NDU/NDC/NDZ

S_0

PC → mem, X, aluA

mem → IR

aluB → +1

aluC → temp

temp → PC

11-9 → A_1

8-6 → A_2

P_1 → T_1

P_2 → T_2

X → aluA

Imm6 → aluB

aluC → T_3

PC → T_4

temp

S_2

T_1 — AluA

T_2 — aluB

aluC — T_3

alu zero — Z

alu carry — C

S_3

5-3 → A_3

T_3 → P_3

States

API

DATE

S_0

|

S_1

|

S_4

$T_1 - aluA$

$5\text{-}0 - Imm\ 6$

$Imm\ 6 - aluB$

$aluC \rightarrow T_3$

$alucarry \rightarrow C$

$aluzero \rightarrow Z$

|

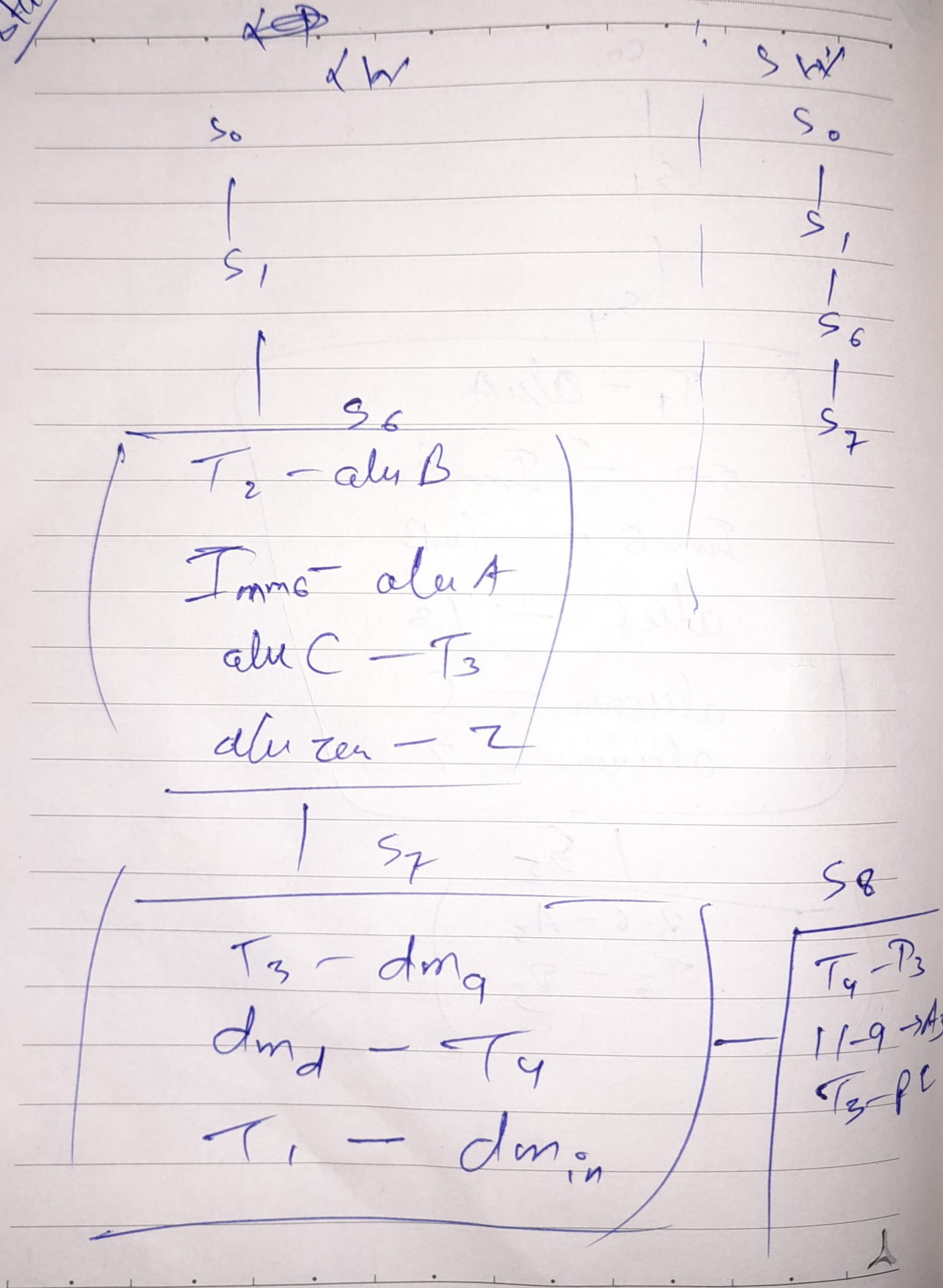
S_5

$8\text{-}6 - A_3$

$T_3 - D_3$

States

DATE


S0
S1
S6
T2 - alu B
Imm6 - alu A
alu C - T3
alu zen - Z
S7
T3 - dm9
dmd - T4
T1 - dmin
S8
T4 - P3
11-9 → A3
T3 - PC

Notes

DATE

LHI

S_0

|

S_9

Imm9 + 7'0' → P_3

11 - 9 → A_3

BEQ

S_0

|

S_1

|

S10

T_1 - alu A

T_2 → alu B

if alu-zero

~~T_3~~ → PC

States

DATE

JAY

S_0

S_1

S_8

$T_4 \leftarrow D_3$

$11-9 \leftarrow A_3$

$T_3 \leftarrow PC$

States

DATE

JAR

S_0

S_1

S_{11}

$T_4 - D_3$

$11 - 9 \rightarrow A_3$

$T_2 \rightarrow PC$

JRI

S_0

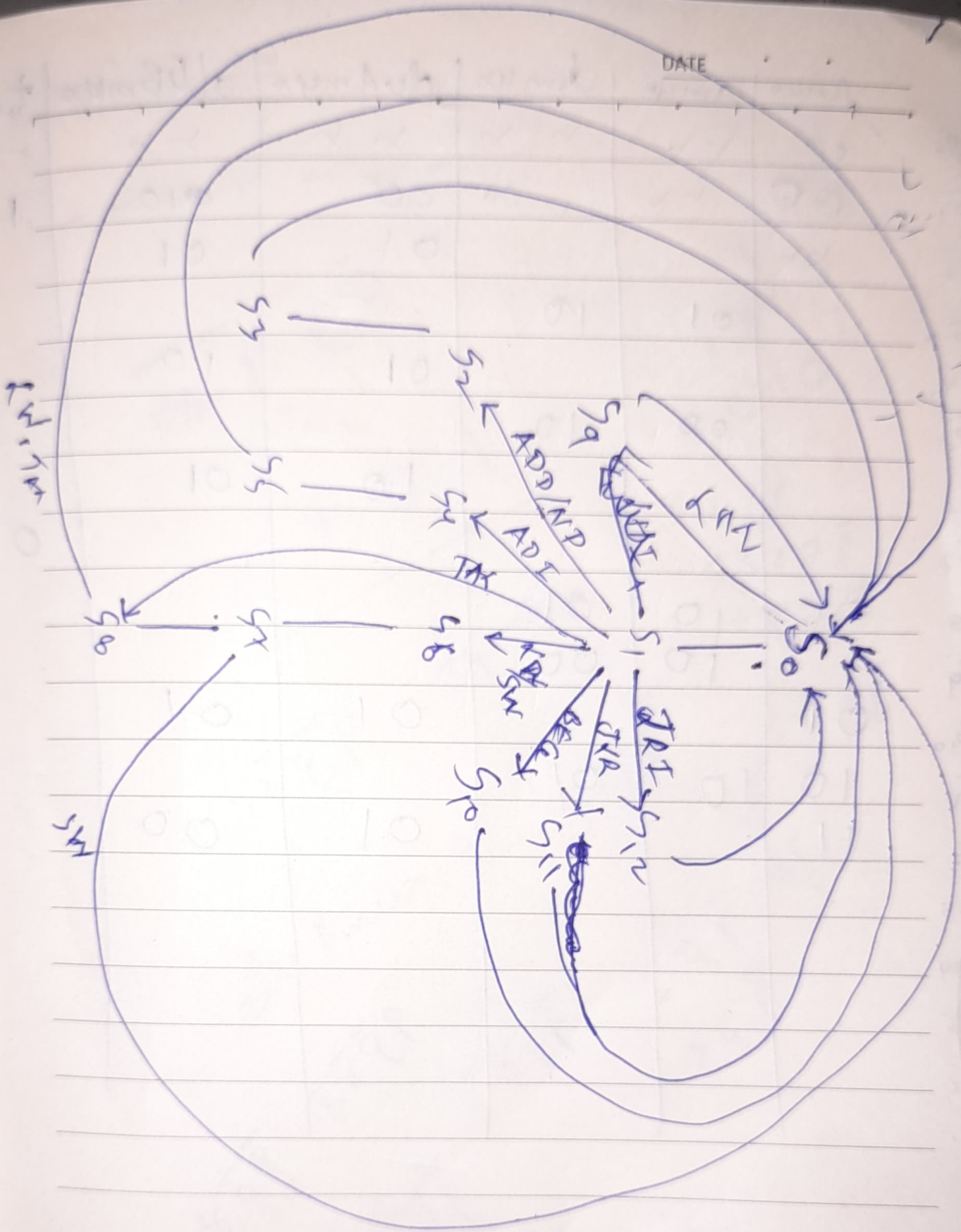
S_1

S_{12}

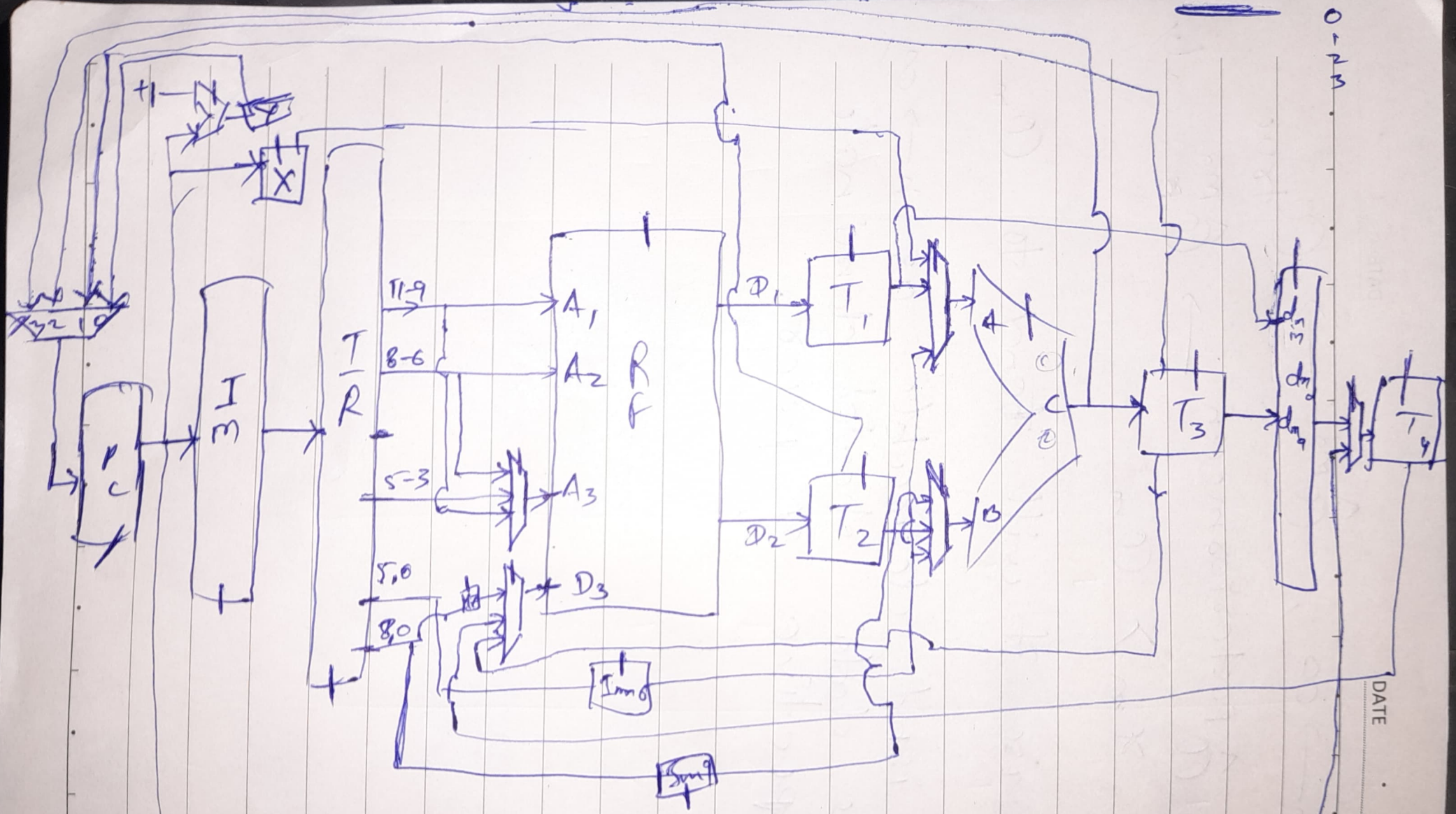
$T_1 - alu A$

$Imm9 - alu B$

$alu C - PC$



State machine



Datapath component

MUX controls

DATE

	pcmux	$a_3$mux	$d_3$mux	aluAmux	aluBmux	$t_4$mux
S_0	~~XX~~	XX	XX	XX	XX	XX
S_1	00	XX	XX	00	10	1
S_2	XX			01	01	
S_3		01	10			
S_4				01	10	
S_5		00	10			
S_6				10	01	
S_7						0
S_8	01	10	01			
S_9		10	00			
S_{10}	01			01	01	
S_{11}	10	10	01			
S_{12}	11			01	00	
S_{13}						
S_{14}						
S_{15}						
S_{16}						
S_{11}						

DATE

f_4 max = $\bar{B}$

d_4 max = $\bar{A}\bar{C}$, C

d_4 max = BC , $\bar{B}\bar{D} + \bar{C}\bar{D}$

d_3 max = $\bar{A}$, $\bar{D} + AC$

a_3 max = A , $\bar{A}.\bar{B}$

pc max = $B + CD$, ~~$A \bar{D}$~~ $\bar{D}$

DATE

For this project, first we designed the components required for it, e.g., alu, 2x1 mux, 4x2 mux, registers, register files, instruction memory, data memory, register file, etc.

Then, we made states corresponding to each operation, finally we got 13 states (0 to 12). So in instruction fetch & S_1 is instruction decode after that we had to use different if statements that worked as decoders determining the next state according to the state machine diagram.

We assembled all the components to make one big component of data path which takes as input clk, rst and control bits and output as C flag, Z flag, zero out of alu, opcode and func.

DATE

Then, in final IITBRISC(2022), we have implemented state machine logic and also set write enable of registers, register file and data memory wherever required.

for setting control bits for muxes, we used K-maps to get formulas for every control bit of muxes.