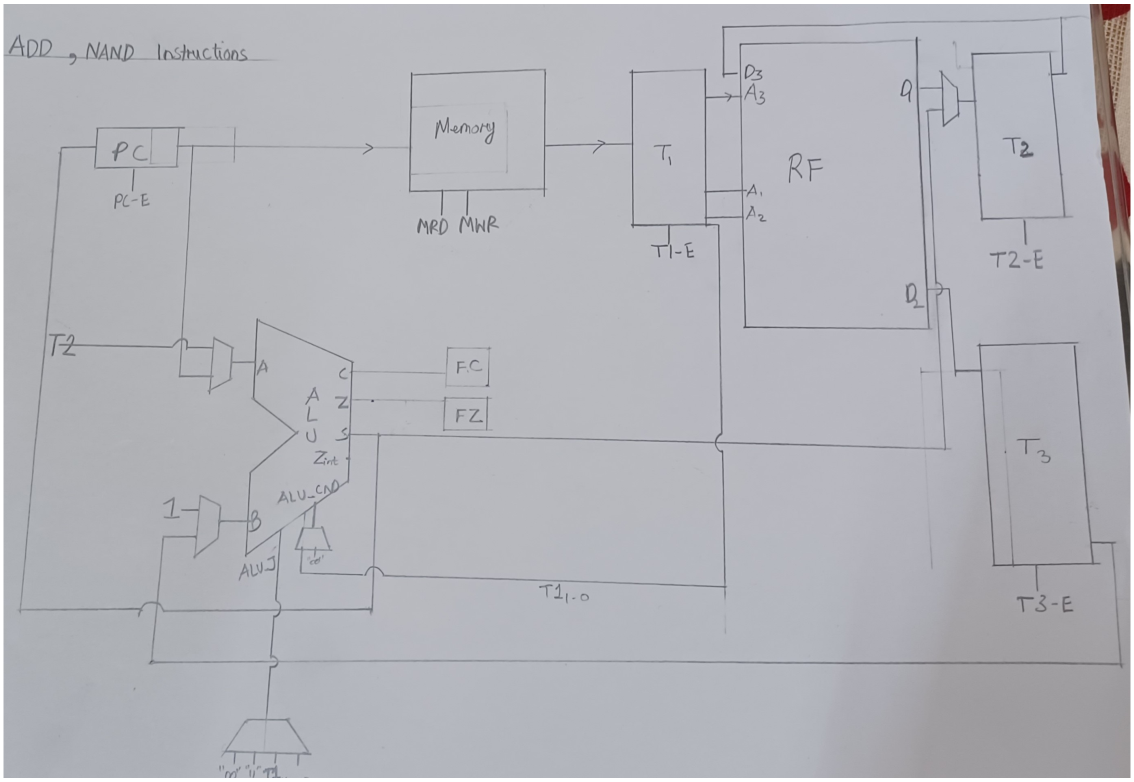
EE 224 Course Project : CPU

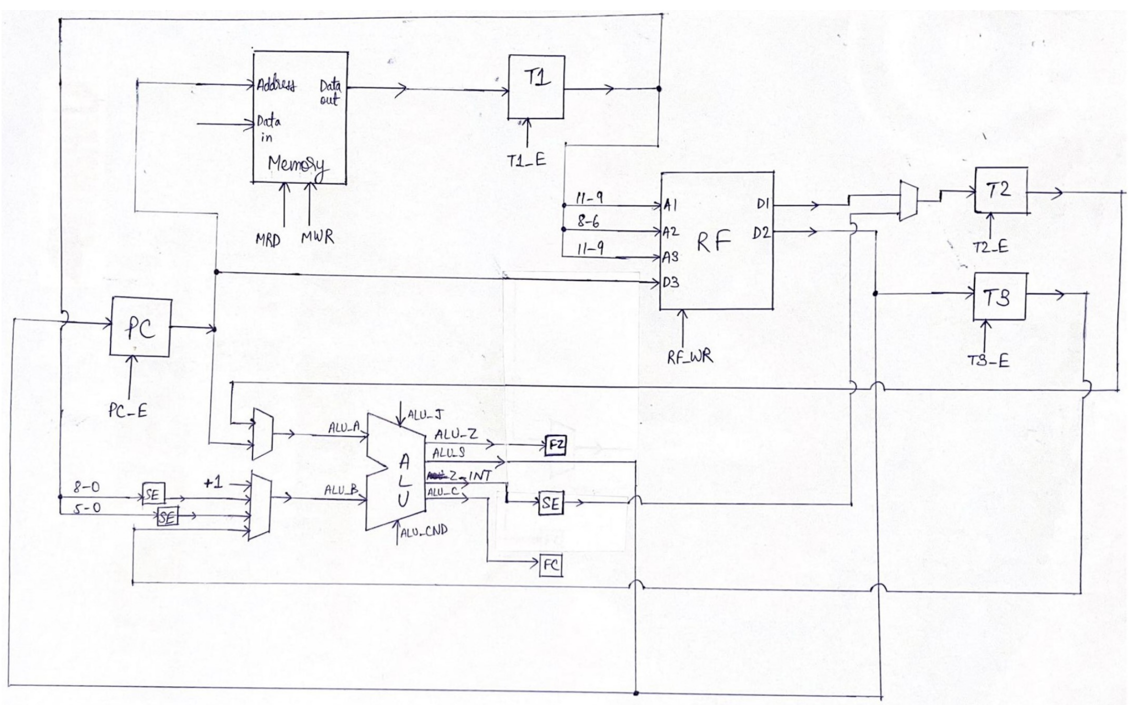
Atharva Kulkarni(210070047)
Harshit Raj(20d070033)
Shreyas Grampurohit(21d070029)
Varad Deshpande(21d070024)

9 Nov 2022

ALU:

Data-flow





We have separated data-flow diagrams for ADD, NAND instructions from other instructions for simplicity.

Inputs:

ALU_A: Takes 16-bit input
ALU_B: Takes 16-bit input

Registers storing the flags:

FC and FZ
These are connected to ALU_C and ALU_Z respectively.

Control Signals:

ALU_J: Takes 2-bit input. This specifies whether to perform addition, subtraction or NAND.
ALU_CND: Takes 2-bit input. This is used to find the new values to be updated to FZ and FC(may or may not be the same as the previous values in FZ and FC) based upon ALU_J(see table).

Outputs:

ALU_C: Outputs the carry flag to be put in the FZ register at the end of the clock cycle.
ALU_Z: Outputs the zero flag to be put in the FZ register at the end of the clock cycle. This may or may not be the same as Z_int.
ALU_S: Outputs sum, NAND, or the difference based on the bits provided in ALU_J.
Z_int: Evaluates to 1 when ALU_S is zero. Else, it evaluates to 0.

ALU_J	Function
00	Addition
01	NAND
11	Subtraction

ALU_J	ALU_CND	Output from ALU_C	Output from ALU_Z
00 (Add)	00	Modified value of carry flag.	Modified value of zero flag.
00 (Add)	10	Modified value of carry flag if input FC is 1. Same as the previous value in FC if FC is 0.	Modified value of zero flag if input FC is 1. Same as the previous value in FZ if FC is 0.
00 (Add)	01	Modified value of carry flag if input FZ is 1. Same as the previous value in FC if FZ is 0.	Modified value of zero flag if input FZ is 1. Same as the previous value in FZ if FZ is 0
00 (Add)	11 (Used for updating PC)	Same as the previous value in FC.	Same as the previous value in FZ.
01 (NAND)	00	Same as the previous value in FC.	Modified value of zero flag.
01 (NAND)	10	Same as the previous value in FC.	Modified value of zero flag if input FC is 1. Same as the previous value in FZ if FC is 0.
01 (NAND)	01	Same as the previous value in FC.	Modified value of zero flag if input FZ is 1. Same as the previous value in FZ if FZ is 0.
11 (Subtract)	xx	Same as the previous value in FC.	Same as the previous value in FZ.

State Descriptions

(PC $\equiv$ R7)

S_0 (Fetching instruction from memory)

Data Transfer	Commands
PC $\rightarrow$ M_add	MDR
M_data $\rightarrow$ T1	T1_E

S_1 (Updating PC)

Data Transfer	Commands
PC $\rightarrow$ ALU_A	PC_E
+1 $\rightarrow$ ALU_B	ALU_J $\leftarrow$ 00
ALU_CND $\leftarrow$ 11	
ALU_S $\rightarrow$ PC	

S_2 (Reading operands)

Data Transfer	Commands
$T1_{11-9}$ $\rightarrow$ RF_A1	T2_E
$T1_{8-6}$ $\rightarrow$ RF_A2	T3_E
RF_D1 $\rightarrow$ T2	
RF_D2 $\rightarrow$ T3	

S_3 (Execution)

Data Transfer	Commands
T2 $\rightarrow$ ALU_A	T2_E
T3 $\rightarrow$ ALU_B	ALU_J $\leftarrow$ $T1_{14-13}$
$T1_{1-0}$ $\rightarrow$ ALU_CND	
ALU_S $\rightarrow$ T2	
ALU_C $\rightarrow$ FC	
ALU_Z $\rightarrow$ FZ	

S_4 (Storing the output)

Data Transfer	Commands
T2 $\rightarrow$ RF_D3	RF_WR
if($T1_{15-12}$ = 0001) then $T1_{8-6}$ $\rightarrow$ RF_A3	
else $T1_{5-3}$ $\rightarrow$ RF_A3	

S_5 (Reading operands (for ADI))

Data Transfer	Commands
$T1_{11-9}$ $\rightarrow$ RF_A1	T2_E
RF_D1 $\rightarrow$ T2	T3_E
$T1_{5-0}$ $\rightarrow$ SE_6 $\rightarrow$ T3	

S_6 (Evaluating condition for BEQ)

Data Transfer	Commands
T2 → ALU_A T3 → ALU_B Z_int → PZ_15 → T2 ALU_CND ← 00	ALU_J ← 11 T2_E

(PZ_15 here means pad 15 zeros to the left)

S_7 (Updating PC in BEQ)

Data Transfer	Commands
PC → ALU_A if($T2_0$ == 0) then +1 → ALU_B else $T1_{5-0}$ → SE_6 → ALU_B ALU_CND ← 11 ALU_S → PC	ALU_J ← 00 PC_E

S_8 (Storing PC into REG_A)

Data Transfer	Commands
$T1_{11-9}$ → RF_A3 PC → RF_D3	RF_WR

S_9 (Branching PC to the address PC + immediate)

Data Transfer	Commands
PC → ALU_A $T1_{8-0}$ → SE_9 → ALU_B ALU_CND ← 11 ALU_S → PC	ALU_J ← 00 PC_E

S_{10} (Branching PC to the address in REG_B)

Data Transfer	Commands
$T1_{8-6}$ → RF_A1 RF_D1 → PC	PC_E

S_{11} (Executing Load Higher Immediate)

Data Transfer	Commands
$T1_{11-9}$ → RF_A3 $T1_{8-0}$ → PZ_7 → RF_D3	RF_WR

(PZ_7 here means pad 15 zeros to the right)

S_{12} (Computing address of the memory destination)

Data Transfer	Commands
T3 → ALU_A $T1_{5-0}$ → SE_6 → ALU_B ALU_S → T3	ALU_J ← 00 ALU_CND ← 11 T3_E

S_{13} (Writing to the memory)

Data Transfer	Commands
T3 → M_add T2 → M_data	MWR

S_{14} (Reading from memory)

Data Transfer	Commands
T3 → M_add M_data → T2	MDR T2_E

S_{15} (Writing to the register)

Data Transfer	Commands
$T1_{11-9}$ → RF_A3 T2 → RF_D3	RF_WR

S_{16} (Initial step of SM and LM)

Data Transfer	Commands
(0000000000000000) → T2 $T1_{11-9}$ → RF_A2 RF_D2 → T3 counter := int($T2_{2-0}$)	T2_E T3_E

S_{17} (Looping step 1 of SM)

Data Transfer	Commands
T3 → ALU_A +1 → ALU_B if($T1_{counter}$==1) then {T3 → M_add $T2_{2-0}$ → RF_A1 RF_D1 → M_data ALU_S → T3 }	T3_E ALU_J ← 00 ALU_CND ← 11 MWR

S_{18} (Updating counter variable (Looping step 2 of SM and LM))

Data Transfer	Commands
counter (converted to 16 bit) → ALU_A 1 bit → ALU_B ALU_S → counter	ALU_J ← 00 ALU_CND ← 11

S_{19} (Looping step 1 of LM)

Data Transfer	Commands
$T1_{counter}$ → RF_WR T3 → M_add M_data → RF_D3 $T2_{2-0}$ → RF_A3 T3 → ALU_A +1 → ALU_B if($T1_{counter}$==1) then ALU_S → T3	T3_E ALU_J ← 00 ALU_CND ← 11 MDR

Instructions with their State Diagrams and Control Signals

Instruction	State flow
ADD	$S_0 \to S_1 \to S_2 \to S_3 \to S_4$
ADC	$S_0 \to S_1 \to S_2 \to S_3 \to S_4$
ADZ	$S_0 \to S_1 \to S_2 \to S_3 \to S_4$
ADI	$S_0 \to S_1 \to S_5 \to S_3 \to S_4$
NDU	$S_0 \to S_1 \to S_2 \to S_3 \to S_4$
NDC	$S_0 \to S_1 \to S_2 \to S_3 \to S_4$
NDZ	$S_0 \to S_1 \to S_2 \to S_3 \to S_4$
LHI	$S_0 \to S_1 \to S_{11}$
LW	$S_0 \to S_1 \to S_2 \to S_{12} \to S_{14} \to S_{15}$
SW	$S_0 \to S_1 \to S_2 \to S_{12} \to S_{13}$
SM	$S_0 \to S_1 \to S_{16} \to S_{17}\ S_{18}$
LM	$S_0 \to S_1 \to S_{16} \to S_{19}\ S_{18}$
BEQ	$S_0 \to S_2 \to S_6 \to S_7$
JAL	$S_0 \to S_8 \to S_9$
JLR	$S_0 \to S_8 \to S_{10}$

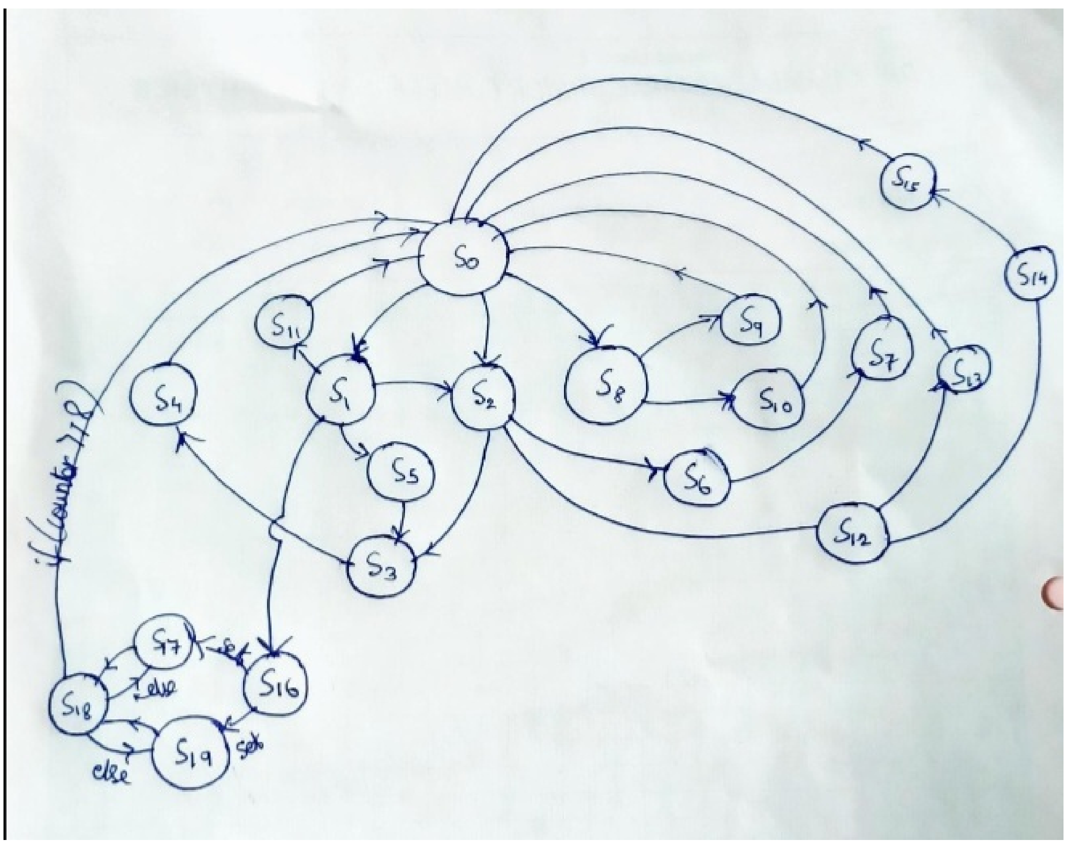


Figure 1: State Transition Diagram