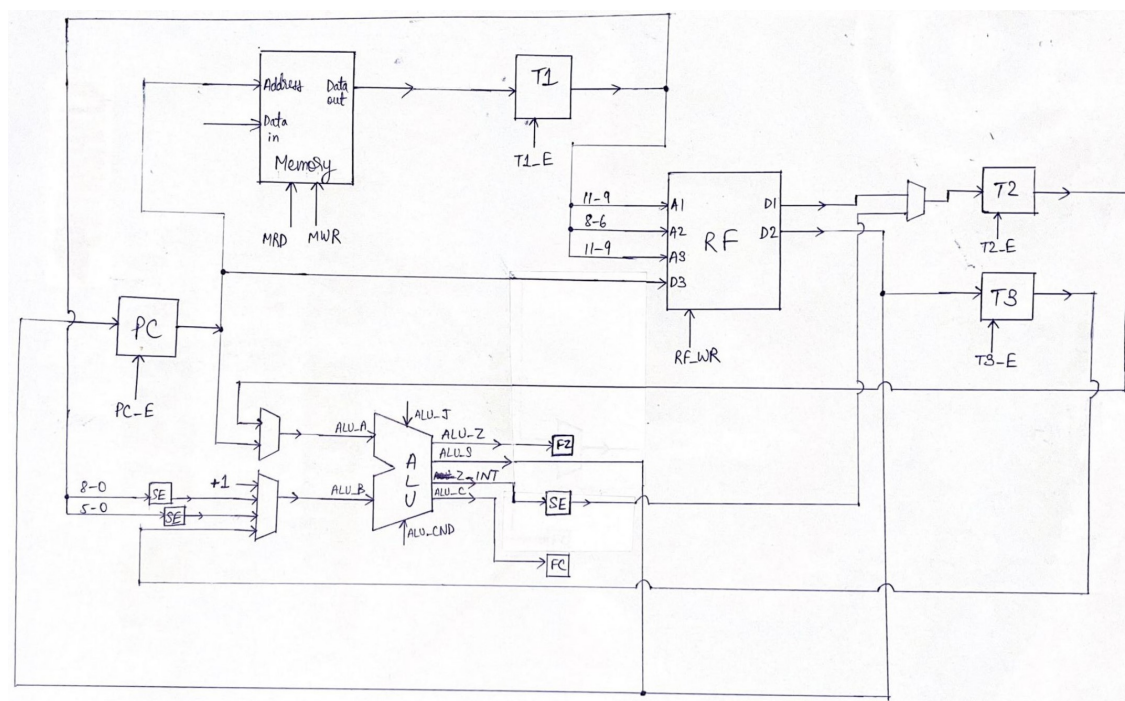


EE 224 Course Project : CPU

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

9 Nov 2022

Data-flow



1

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	T4.E
T3 \rightarrow ALU.B	ALU.J \leftarrow $T1_{14-13}$
$T1_{1-0} \rightarrow$ ALU.CND	
ALU.S \rightarrow T4	
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 \rightarrow ALU_A T3 \rightarrow ALU_B Z_int \rightarrow PZ_15 \rightarrow T4 ALU_CND \leftarrow 00	ALU_J \leftarrow 11 T4_E

(PZ_15 here means pad 15 zeros to the left)

S_7 (Updating PC in BEQ)

Data Transfer	Commands
PC \rightarrow ALU_A if($T_{4_0} == 0$) then $+1 \rightarrow$ ALU_B else $T_{15-0} \rightarrow$ SE_6 \rightarrow ALU_B ALU_CND \leftarrow 11 ALU_S \rightarrow PC	ALU_J \leftarrow 00 PC_E

S_8 (Storing PC into REG_A)

Data Transfer	Commands
$T_{11-9} \rightarrow$ RF_A3 PC \rightarrow RF_D3	RF_WR

S_9 (Branching PC to the address PC + immediate)

Data Transfer	Commands
PC \rightarrow ALU_A $T_{18-0} \rightarrow$ SE_9 \rightarrow ALU_B ALU_CND \leftarrow 11 ALU_S \rightarrow PC	ALU_J \leftarrow 00 PC_E

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

Data Transfer	Commands
$T_{18-6} \rightarrow$ RF_A1 RF_D1 \rightarrow PC	PC_E

S_{11} (Executing Load Higher Immediate)

Data Transfer	Commands
$T_{11-9} \rightarrow$ RF_A3 $T_{18-0} \rightarrow$ PZ_7 \rightarrow 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 \rightarrow ALU_A $T_{15-0} \rightarrow$ SE_6 \rightarrow ALU_B ALU_S \rightarrow T4	ALU_J \leftarrow 00 ALU_CND \leftarrow 11 T4_E

S_{13} (Writing to the memory)

Data Transfer	Commands
T4 \rightarrow M_add	MWR
T2 \rightarrow M_data	

S_{14} (Reading from memory)

Data Transfer	Commands
T4 \rightarrow M_add	MDR
M_data \rightarrow T2	T2_E

S_{15} (Writing to the register)

Data Transfer	Commands
T1 ₁₁₋₉ \rightarrow RF_A3	RF_WR
T2 \rightarrow RF_D3	

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

Data Transfer	Commands
(0000000000000000) \rightarrow T2	T2_E
T1 ₁₁₋₉ \rightarrow RF_A2	T3_E
RF_D2 \rightarrow T3	
counter := int(T2 ₂₋₀)	

S_{17} (Looping step 1 of SM)

Data Transfer	Commands
<pre> if(mod2(counter) == 0) then{ T3 \rightarrow ALU_A +1 \rightarrow ALU_B if(T1_{counter}==1) then{ T3 \rightarrow M_add counter \rightarrow 3 bit unsigned \rightarrow RF_A1 RF_D1 \rightarrow M_data ALU_S \rightarrow T4 }} else{ T4 \rightarrow ALU_A +1 \rightarrow ALU_B if(T1_{counter}==1) then{ T4 \rightarrow M_add counter \rightarrow 3 bit unsigned \rightarrow RF_A1 RF_D1 \rightarrow M_data ALU_S \rightarrow T3 }} </pre>	<pre> if(mod2(counter) == 0) then T3_E\leftarrow0 else T3_E\leftarrow1 if(mod2(counter) == 1) then T4_E\leftarrow1 else T3_E\leftarrow0 ALU_J \leftarrow 00 ALU_CND \leftarrow 11 MWR </pre>

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

Data Transfer	Commands
counter (converted to 16 bit) \rightarrow ALU_A	ALU_J \leftarrow 00
1 bit \rightarrow ALU_B	ALU_CND \leftarrow 11
ALU_S \rightarrow counter	

S_{19} (Looping step 1 of LM)

Data Transfer	Commands
<pre> if(mod2(counter) == 0) then{ counter → 3 bit unsigned → RF_A3 T3 → M_add M_data → RF_D3 T3 → ALU_A +1 → ALU_B if($T1_{counter} == 1$) then{ ALU_S → T4 }} else{ counter → 3 bit unsigned → RF_A3 T4 → M_add M_data → RF_D3 if($T1_{counter} == 1$) then{ ALU_S → T3 }}</pre>	<pre> if(mod2(counter) == 0) then T3_E ← 0 else T3_E ← 1 if(mod2(counter) == 1) then T4_E ← 1 else T3_E ← 0 ALU_J ← 00 ALU_CND ← 11 MDR</pre>

Instructions with their State Diagrams and Control Signals

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

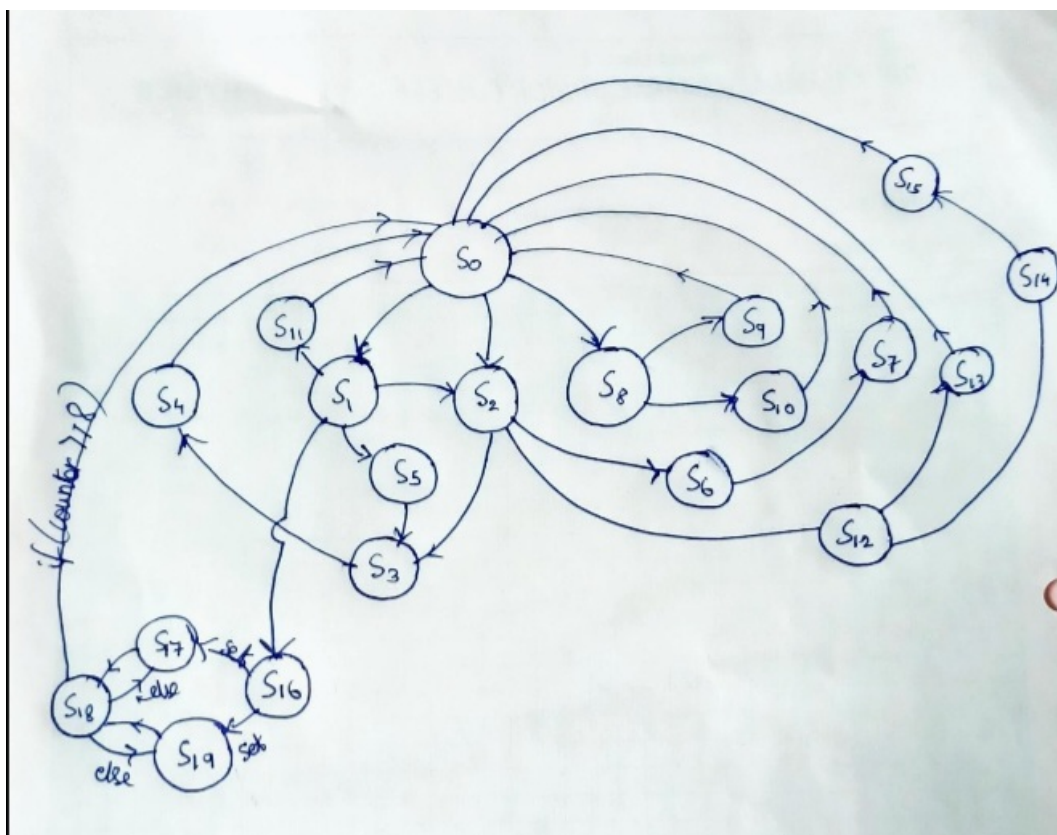


Figure 1: State Transition Diagram