IITB - CPU

EE - 224 PROJECT

Course Instructor : Professor Virendra Singh

TEAM ID : 58

Group Members:-

1. Sarvadnya Purkar(22B4232)
2. Prajwal Nayak(22B4246)
3. Utkarsh Maurya(22B3910)
4. Samarth Sirsat(22B3988)

COMPONENTS USED

1. Memory

2. Register File, including the PC Register

3.ALU (add, subtract , multiply , and , or , imp)

4.SE7 (To extend 9 bits to 16 bits)

6. SE8 (To extend 8 bits to 16 bits)

7. SE10(To extend 6 bits to 16 bits)

8. Temporary Register

9. MUX 8x1

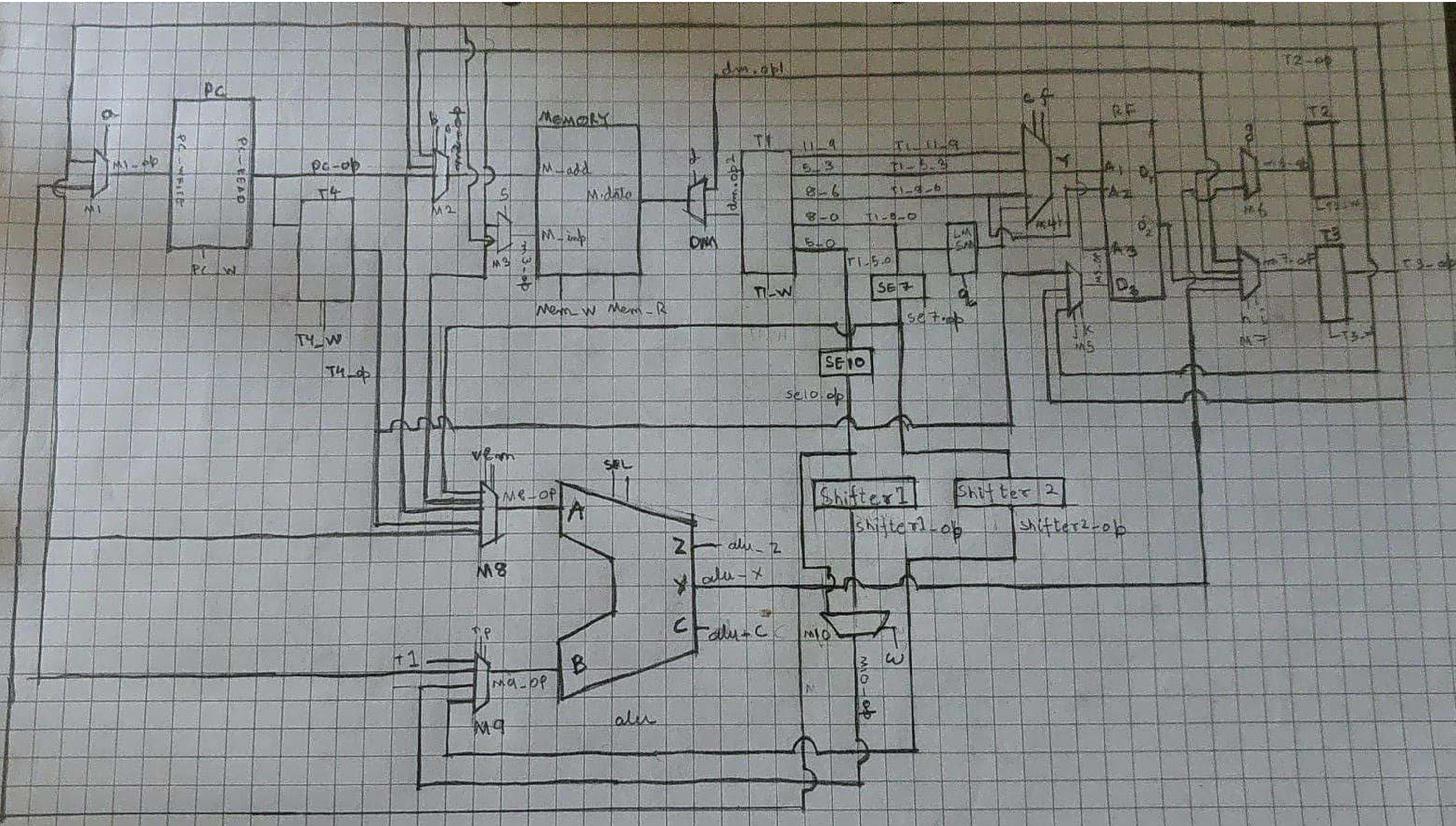
10. MUX 1x 4

11. MUX 1x2

12. DEMUX 1x2 16-bit

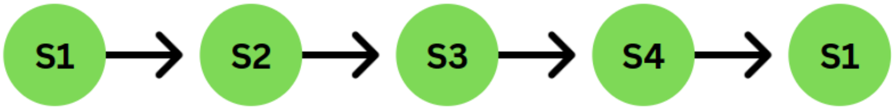
13. DEMUX 1x2 4-bit

14. Shifter 2 bit

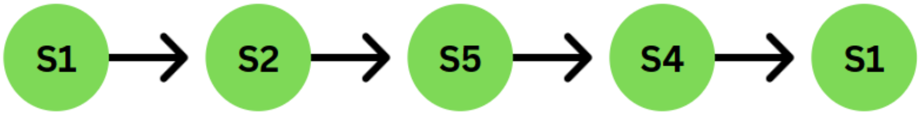

PC
PC_WRITE
PC_READ
PC_W
a
M1_op
M1
pc-op
T4
T4_W
T4_op
b
c
M2
M2-op
S
M3
M3-op
MEMORY
M_add
M_data
M_inp
Mem_W Mem_R
d
DM
dm.op1
dm.op2
T1
T1_W
11_9
5_3
8_6
8_0
5_0
T1_11_9
T1_5_3
T1_8_6
T1_8_0
T1_5_0
LM SM
g
SE7
se7.op
SE10
se10.op
e f
M4
RF
A1
A2
A3
D3
O1
O2
j k
M5
g
M6
T2
T2_op
LT2_w
h i
M7
m7.op
T3
LT3_w
T3_op
Shifter1
shifter1_op
Shifter 2
shifter2_op
vem
M8
M8_op
SEL
A
B
Z
X
C
alu_Z
alu_X
alu_C
alu
+1
T1
M9
M9_op
M10
M10_op
W

STATE FLOW DIAGRAM

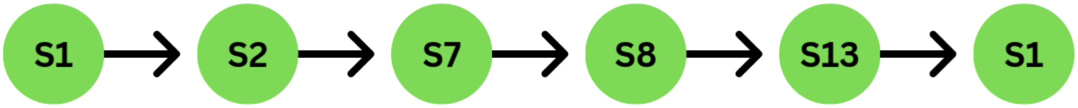
ADD , SUB , MUL , AND , ORA , IMP :



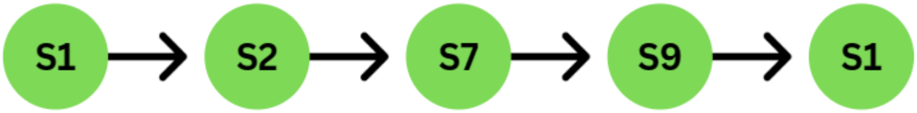
ADI :



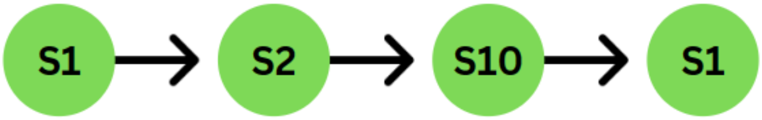
LW :



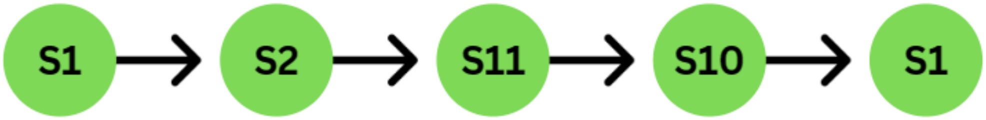
SW :



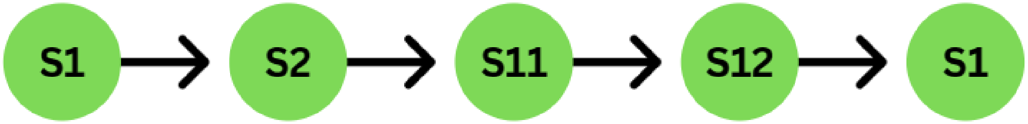
BEQ :



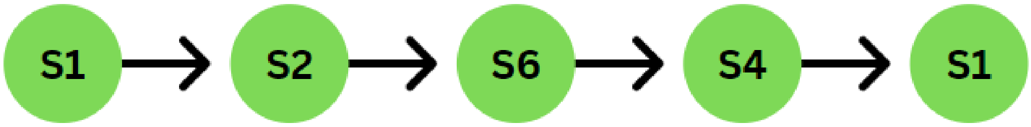
JAL :



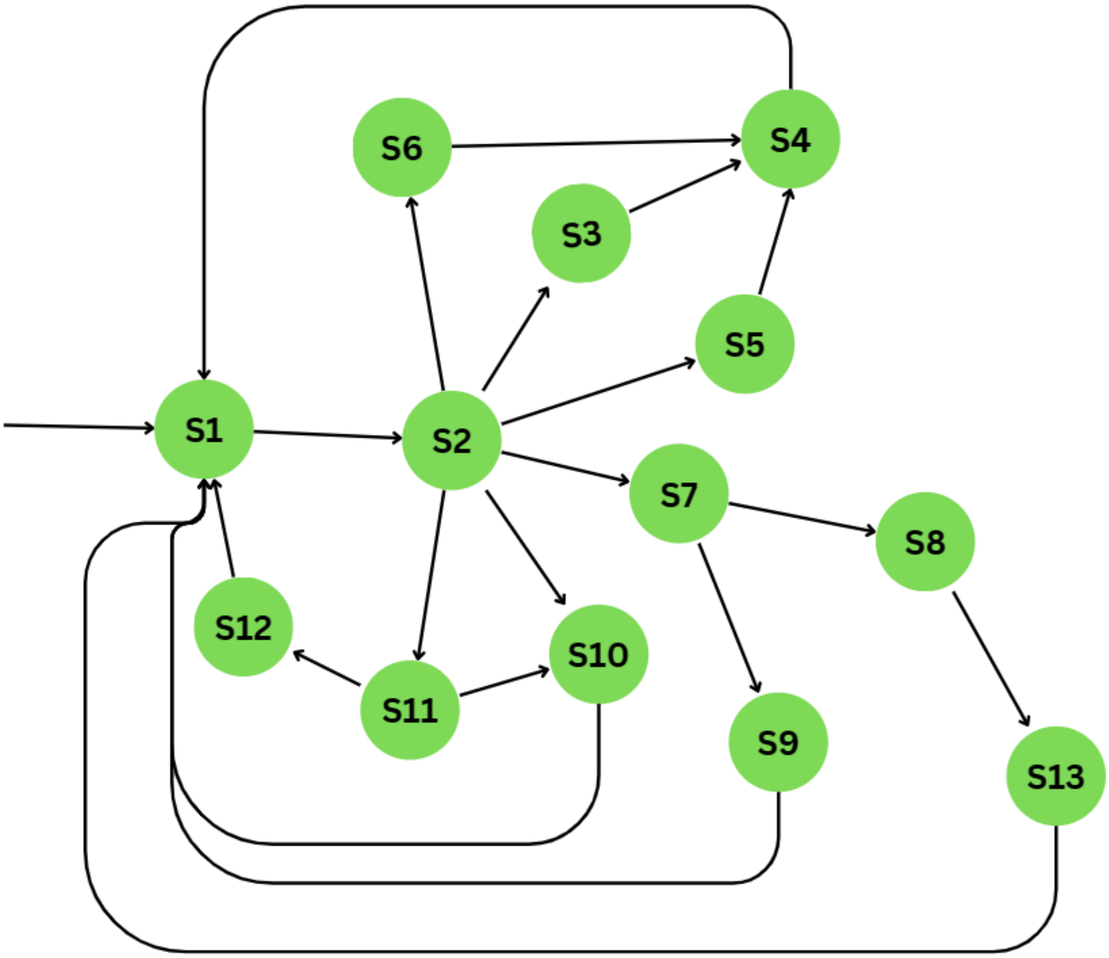
JLR :



LHI , LLI .



STATE DIAGRAM NET



HARDWARE FLOWCHARTS

S1	PC → M_ADD, T4, ALU_A M_DATA → T1 +1 → ALU_B Memory is a two-dimensional integer indexed array. To move to the next memory address, integer 1 is added. ALU_X → PC	ADD MEM_R T1_W PC_W T4_W
S2	T1_11-9 → RF_A1 RF_D1 → T2 T1_8-6 → RF_A2 RF_D2 → T3 T1_5-0 → SE10	T2_W T3_W

S3	T2 → ALU_A
	T3 → ALU_B ALU_X → T2
	
	if (t1_op (14 downto 12) = "000") then --add
	sel<="0000";
	elsif (t1_op (14 downto 12) = "010") then --sub
	sel<="0001";
	elsif (t1_op (14 downto 12) = "011") then --mul
	sel<="0010";
	elsif (t1_op (14 downto 12) = "100") then --and
	sel<="0100";
	elsif (t1_op (14 downto 12) = "101") then --ora
	sel<="0101";
	elsif (t1_op (14 downto 12) = "110") then --imp
	sel<="0110";
	else null;
	end if;
		R1 instruction
		T2_W

S4	Y → RF_A3 T2 → RF_D3	RF_W
S5	SE10 → ALU_B T2 → ALU_A ALU_X → T2	ADD T2_W
S6	T1_8-0 → SE7 SE7 → ALU_A	SHIFT7 T2_W

	ALU_X → T2	
S7	T3→ ALU_A SE10 → ALU_B ALU_X → T3	T3_W ADD
S8	T3 → M_ADD M_DATA → T3	T3_W MEM_R
S9	T3 → M_ADD T2 → M_INP	MEM_W

S10	T4 → ALU_A
	if (t1_op(13)='1') then
	if (T2=T3) then
	SE10 → ALU_B
	sel<="0000";
	ALU_X → PC
	else --nothing must be done
	sel<="1111";
	end if;
	else
	SE7 → ALU_B
	sel<="0000";
	ALU_X → PC
	end if;
		ADD
		PC_W

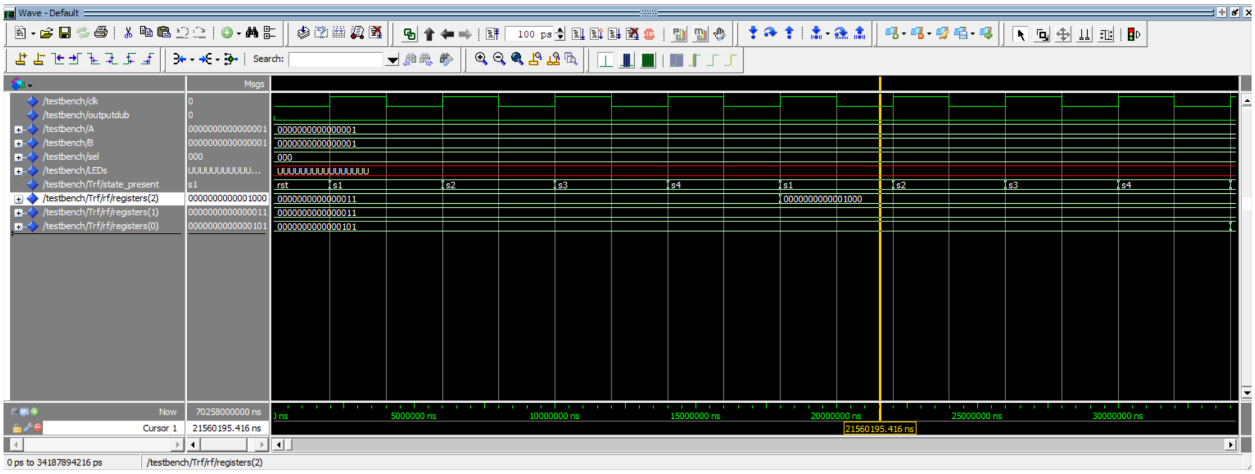
S11	T4 → RF_D3 T1_11_9 → RF_A3	RF_W
S12	T3 → PC	PC_W
S13	T1_11_9 → RF_A3 T3 → RF_D3	RF_W

TESTING

regA is R0, regB is R1, regC is R2

1)ADD{0000000001010000}

The value of r0 is added to r1 and stored in r2



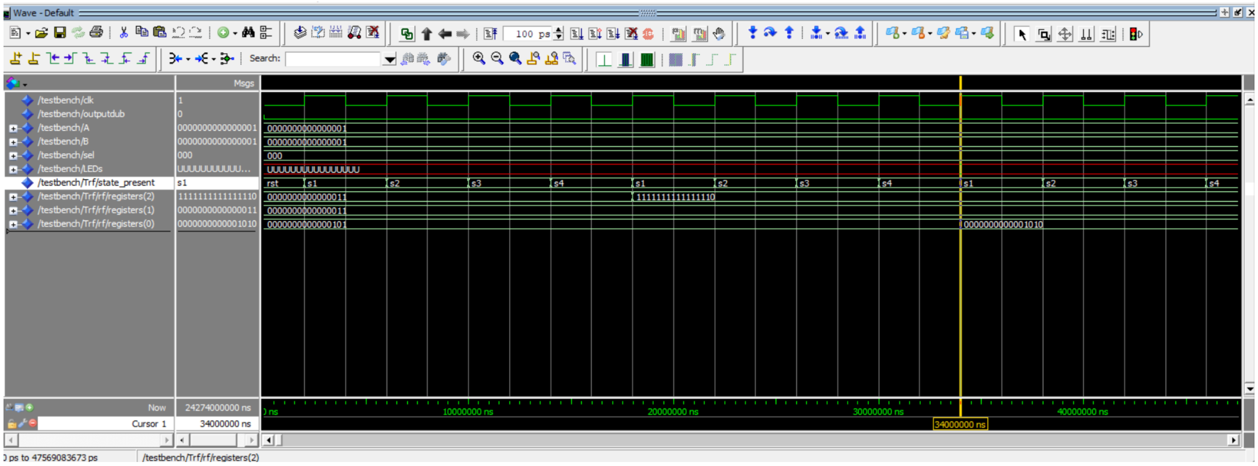
Wave - Default

	Msgs	
/testbench/clk	0	
/testbench/outputdub	0	
/testbench/A	0000000000000001	0000000000000001
/testbench/B	0000000000000001	0000000000000001
/testbench/sel	000	000
/testbench/LEDs	UUUUUUUUUUU...	UUUUUUUUUUUUUUUU
/testbench/Trf/state_present	s1	rst, s1, s2, s3, s4, s1, s2, s3, s4
/testbench/Trf/rf/registers(2)	0000000000001000	0000000000000011, 0000000000001000
/testbench/Trf/rf/registers(1)	0000000000000011	0000000000000011
/testbench/Trf/rf/registers(0)	0000000000000101	0000000000000101
Now	70258000000 ns	
Cursor 1	21560195.416 ns	21560195.416 ns

0 ps to 34187894216 ps /testbench/Trf/rf/registers(2)

2)SUB{0010000001010000}

The value of r0 is subtracted from that of r1 and the result is stored in r2



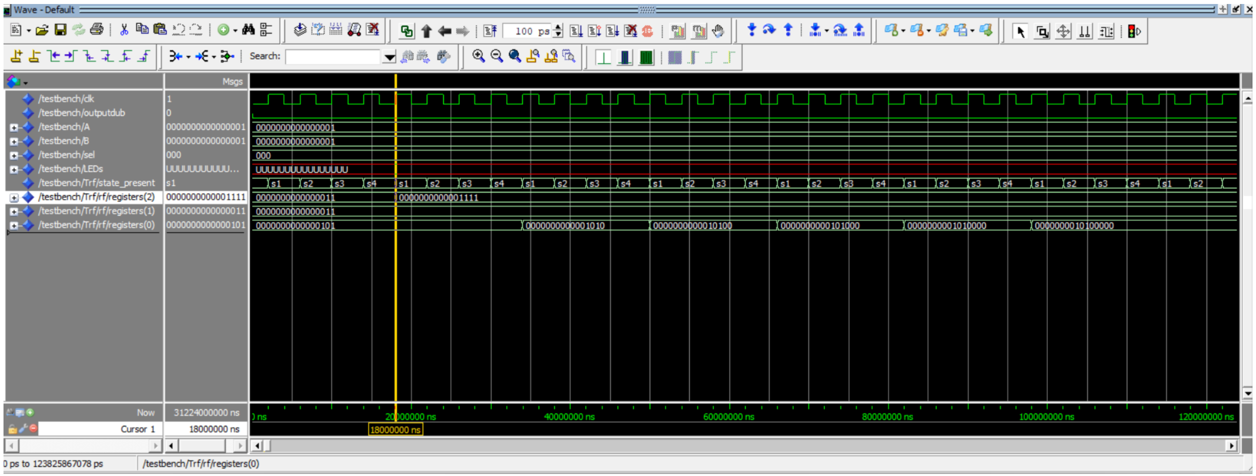
Wave - Default

	Msgs
/testbench/clk	1
/testbench/outputdub	0
/testbench/A	0000000000000001
/testbench/B	0000000000000001
/testbench/sel	000
/testbench/LEDs	UUUUUUUUUUU...
/testbench/Trf/state_present	s1
/testbench/Trf/rf/registers(2)	1111111111111110
/testbench/Trf/rf/registers(1)	0000000000000011
/testbench/Trf/rf/registers(0)	0000000000001010
Now	24274000000 ns
Cursor 1	34000000 ns

0 ps to 47569083673 ps   /testbench/Trf/rf/registers(2)

3)MUL{0011000001010000}

First 4 bits of r0 are multiplied with first 4 bits of r1 and the value is stored in r2



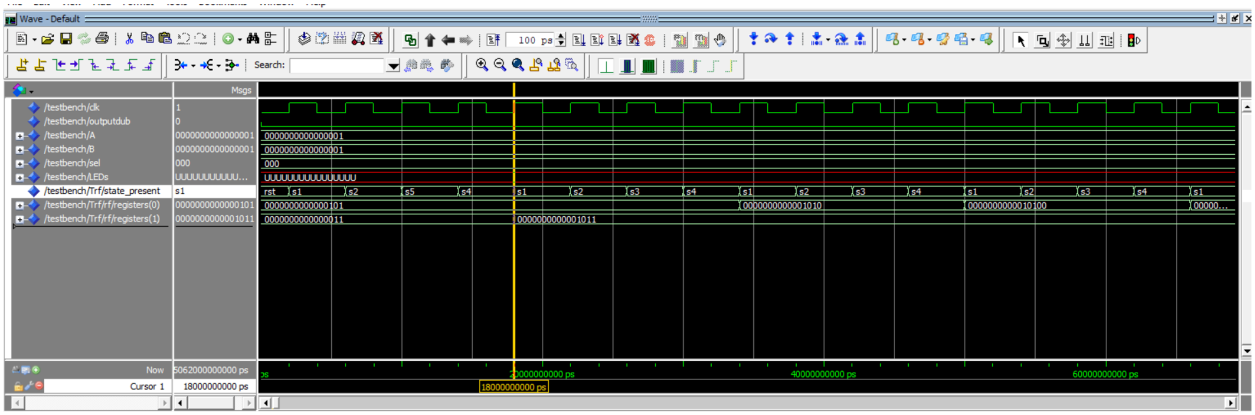
Wave - Default

	Msgs
/testbench/clk	1
/testbench/outputdub	0
/testbench/A	0000000000000001
/testbench/B	0000000000000001
/testbench/sel	000
/testbench/LEDs	UUUUUUUUUUU...
/testbench/Trf/state_present	s1
/testbench/Trf/rf/registers(2)	0000000000001111
/testbench/Trf/rf/registers(1)	0000000000000011
/testbench/Trf/rf/registers(0)	0000000000000101
Now	31224000000 ns
Cursor 1	18000000 ns

0 ps to 123825867078 ps   /testbench/Trf/rf/registers(0)

4)ADI{immediate=6}{0001000001000110}

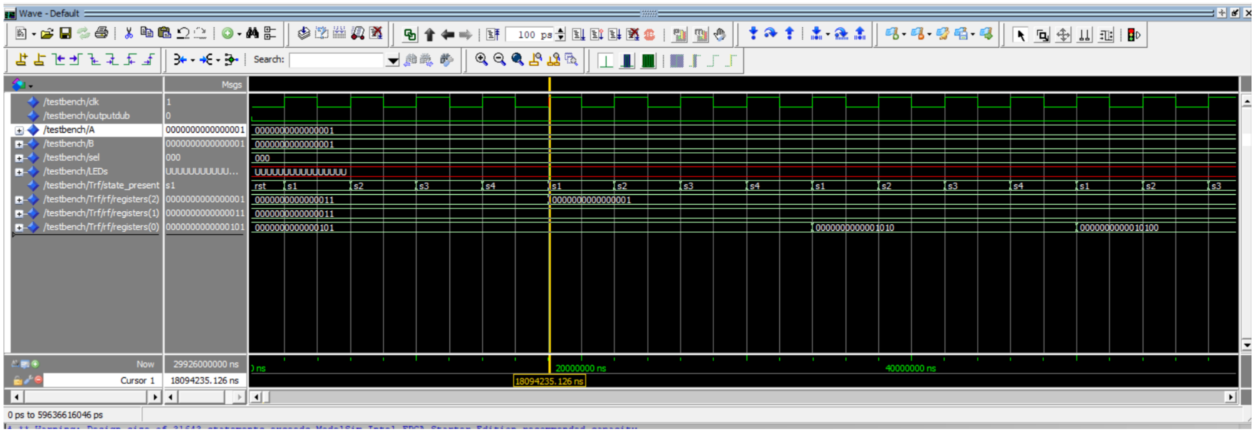
Add content of r0 with Imm (sign extended 000110) and store result in r1



```
Wave - Default
Search:
                                  Msgs
/testbench/clk                    1
/testbench/outputdub              0
/testbench/A                      0000000000000001
/testbench/B                      0000000000000001
/testbench/sel                    000
/testbench/LEDs                   UUUUUUUUUUU...
/testbench/Trf/state_present      s1
/testbench/Trf/rf/registers(0)    0000000000000101
/testbench/Trf/rf/registers(1)    0000000000001011
Now       5062000000000 ps
Cursor 1  18000000000 ps
```

5)AND{0100000001010000}

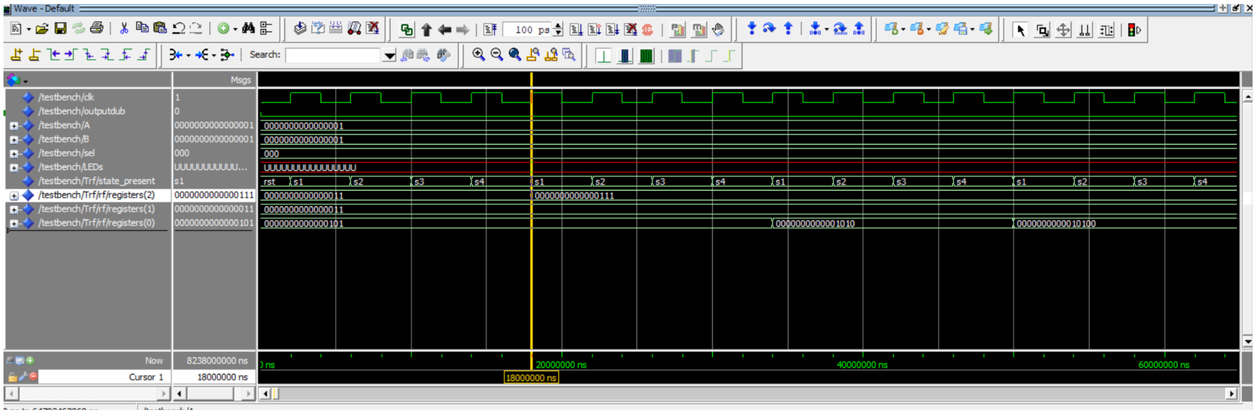
Digital And of the values in r0 and r1 is done which is stored in r2



```
Wave - Default
Search:
                                  Msgs
/testbench/clk                    1
/testbench/outputdub              0
/testbench/A                      0000000000000001
/testbench/B                      0000000000000001
/testbench/sel                    000
/testbench/LEDs                   UUUUUUUUUUU...
/testbench/Trf/state_present      s1
/testbench/Trf/rf/registers(2)    0000000000000001
/testbench/Trf/rf/registers(1)    0000000000000011
/testbench/Trf/rf/registers(0)    0000000000000101
Now       29926000000 ns
Cursor 1  18094235.126 ns
0 ps to 596366160046 ps
```

6)ORA{ 0101000001010000}

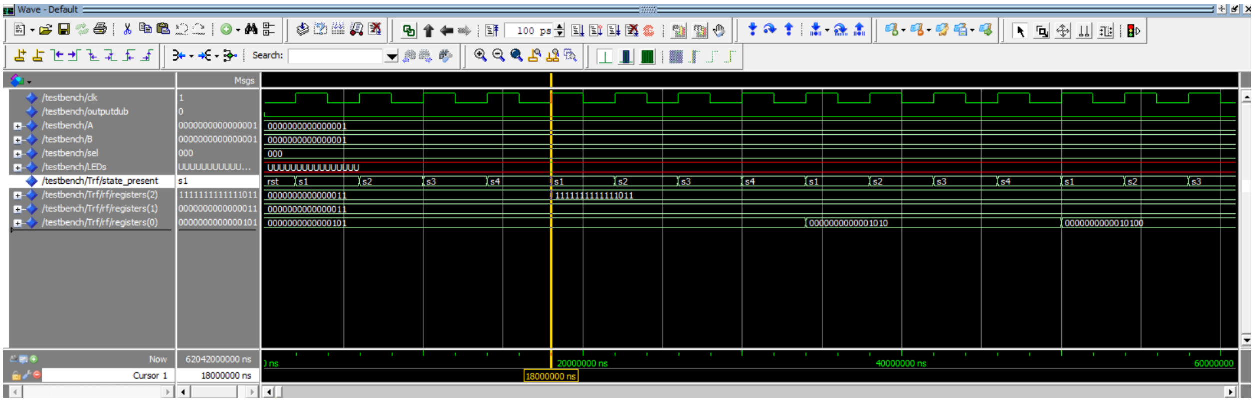
Digital OR of the values in r0 and r1 is done which is stored in r2



```
Wave - Default
                                   Msgs
/testbench/clk                     1
/testbench/outputdub               0
/testbench/A                       0000000000000001
/testbench/B                       0000000000000001
/testbench/sel                     000
/testbench/LEDs                    UUUUUUUUUUU...
/testbench/Trf/state_present       s1
/testbench/Trf/rf/registers(2)     0000000000000111
/testbench/Trf/rf/registers(1)     0000000000000011
/testbench/Trf/rf/registers(0)     0000000000000101
Now                                8238000000 ns
Cursor 1                           18000000 ns
```

7)IMP{ 0110000001010000}

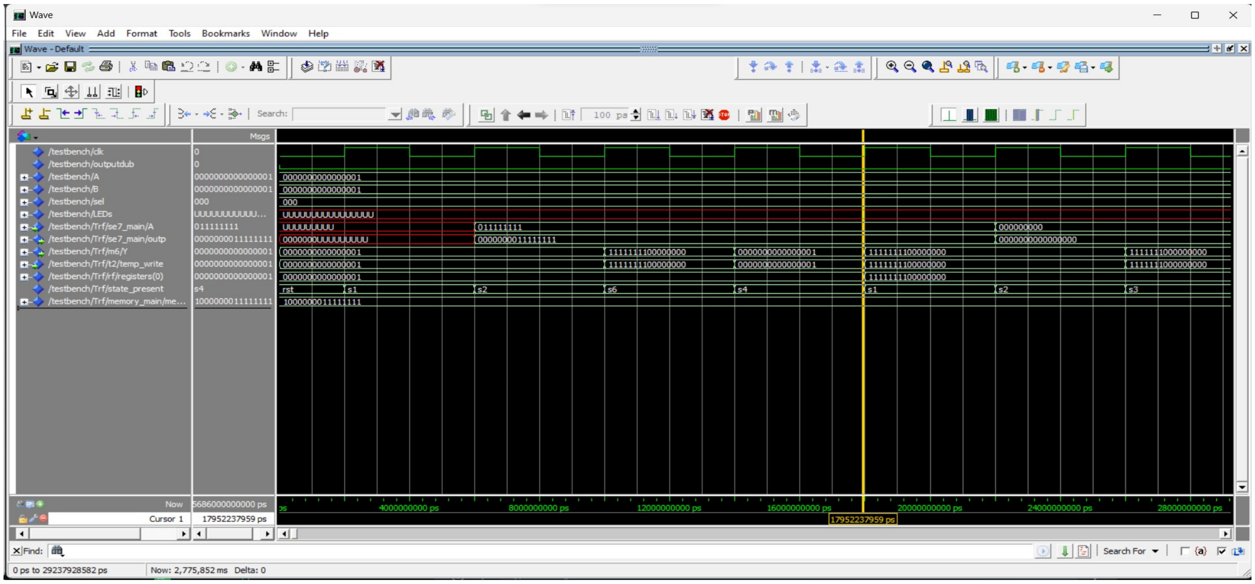
Digital Implication of the values in r0 and r1 is done which is stored in r2



```
Wave - Default
                                   Msgs
/testbench/clk                     1
/testbench/outputdub               0
/testbench/A                       0000000000000001
/testbench/B                       0000000000000001
/testbench/sel                     000
/testbench/LEDs                    UUUUUUUUUUU...
/testbench/Trf/state_present       s1
/testbench/Trf/rf/registers(2)     1111111111111011
/testbench/Trf/rf/registers(1)     0000000000000011
/testbench/Trf/rf/registers(0)     0000000000000101
Now                                62042000000 ns
Cursor 1                           18000000 ns
```

8)LHI{ 1000000011111111}

Here the first 0+8 immediate bits are 0+11111111, the 8 bit immediate i.e 11111111 is put into most significant 9 bits of r0 and the rest of the values are put to 0.



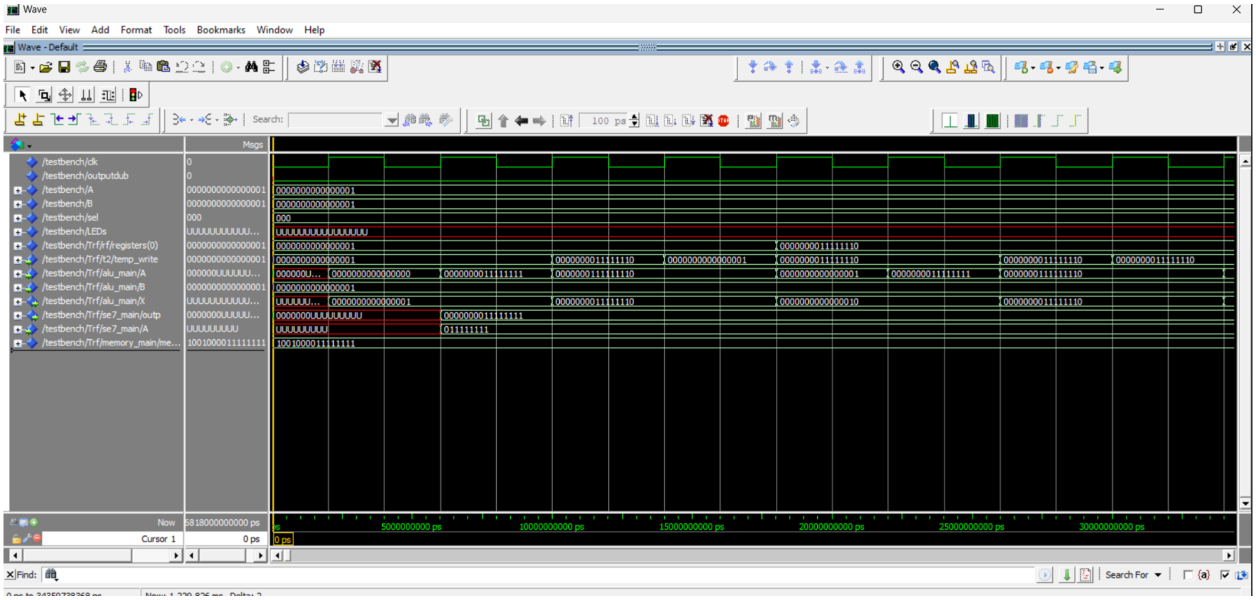
```
/testbench/clk                      0
/testbench/outputdub                0
/testbench/A                        0000000000000001
/testbench/B                        0000000000000001
/testbench/sel                      000
/testbench/LEDs                     UUUUUUUUUUU...
/testbench/Trf/se7_main/A           011111111
/testbench/Trf/se7_main/outp        0000000011111111
/testbench/Trf/m6/Y                 0000000000000001
/testbench/Trf/t2/temp_write        0000000000000001
/testbench/Trf/rf/registers(0)      0000000000000001
/testbench/Trf/state_present        s4
/testbench/Trf/memory_main/me...    1000000011111111
```

```
0000000000000001
0000000000000001
000
UUUUUUUUUUUUUUUU
UUUUUUUUU          011111111                                                                       000000000
0000000UUUUUUUUU   0000000011111111                                                                0000000000000000
0000000000000001                    1111111100000000  0000000000000001  1111111100000000                              1111111000000000
0000000000000001                    1111111100000000  0000000000000001  1111111100000000                              1111111000000000
0000000000000001                                                        1111111100000000
rst         s1     s2               s6                s4                s1                         s2                 s3
1000000011111111
```

```
Now       5686000000000 ps   ps   4000000000 ps   8000000000 ps   12000000000 ps   16000000000 ps   20000000000 ps   24000000000 ps   28000000000 ps
Cursor 1  17952237959 ps                                                          17952237959 ps
0 ps to 29237928582 ps      Now: 2,775,852 ms  Delta: 0
```

9)LLI{ 1001000110010100}

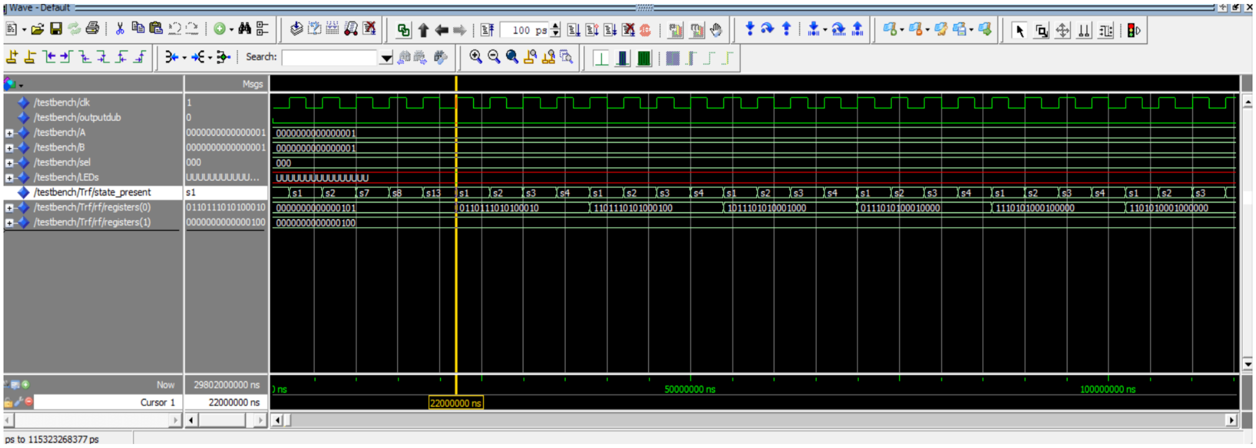
Here the first 0+8 immediate bits are 0+11111111, the 8 bit immediate i.e 11111111 is put into least significant 9 bits of r0 and the rest of the values are put to 0.



```
/testbench/clk                       0
/testbench/outputdub                 0
/testbench/A                         0000000000000001
/testbench/B                         0000000000000001
/testbench/sel                       000
/testbench/LEDs                      UUUUUUUUUUU...
/testbench/Trf/rf/registers(0)       0000000000000001
/testbench/Trf/t2/temp_write         0000000000000001
/testbench/Trf/alu_main/A            000000UUUUUU...
/testbench/Trf/alu_main/B            0000000000000001
/testbench/Trf/alu_main/X            UUUUUUUUUUU...
/testbench/Trf/se7_main/outp         0000000UUUUU...
/testbench/Trf/se7_main/A            UUUUUUUUU
/testbench/Trf/memory_main/me...     1001000011111111
```

10)LW {1010000001000010}

Here the 0th instructions is 1010000001000010 and the 6th instruction is 0110111010100010. R0 is taken as regA and R1 is taken as regB. Reg B has value 4 (i.e. 0000000000000100)stored in it which is added to the immediate (2 in this case) thus there sum comes out to be 6 which is the new pointer and the value at 6th position i.e. 0110111010100010 is written in regA which in this case is R0.



Wave - Default

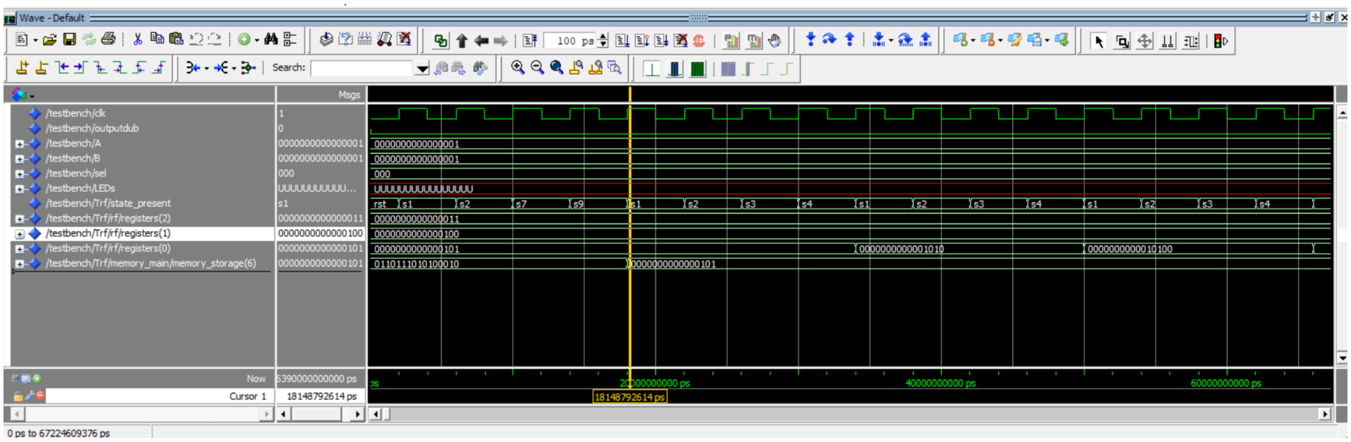
```
/testbench/clk                     1
/testbench/outputdub               0
/testbench/A                       0000000000000001
/testbench/B                       0000000000000001
/testbench/sel                     000
/testbench/LEDs                    UUUUUUUUUUU...
/testbench/Trf/state_present       s1
/testbench/Trf/rf/registers(0)     0110111010100010
/testbench/Trf/rf/registers(1)     0000000000000100

Now        29802000000 ns
Cursor 1   22000000 ns
```

ps to 115323268377 ps

11)SW{ 1011000001000010}

Here Imm (sign extended 000010) is added to the value in r1 i.e 0000000000000100(4 in decimal) it comes out to be 6. As seen from the waveform the value of memory with PC 6 is changed to the value present in R0 i.e 0000000000000101



Wave - Default

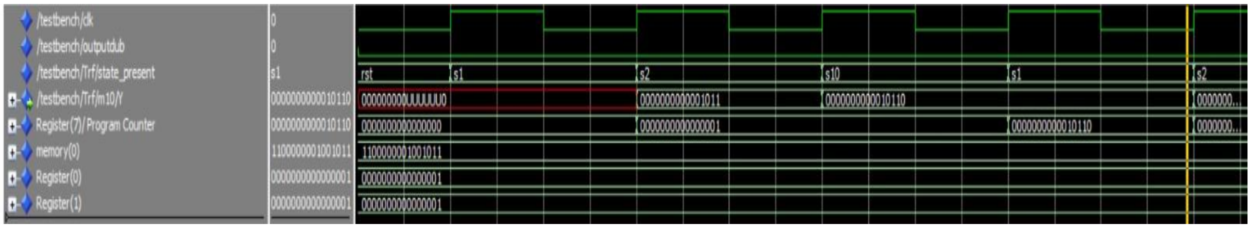
```
/testbench/clk                                  1
/testbench/outputdub                            0
/testbench/A                                    0000000000000001
/testbench/B                                    0000000000000001
/testbench/sel                                  000
/testbench/LEDs                                 UUUUUUUUUUU...
/testbench/Trf/state_present                    s1
/testbench/Trf/rf/registers(2)                  0000000000000011
/testbench/Trf/rf/registers(1)                  0000000000000100
/testbench/Trf/rf/registers(0)                  0000000000000101
/testbench/Trf/memory_main/memory_storage(6)    0000000000000101

Now        6390000000000 ps
Cursor 1   18148792614 ps
```

0 ps to 67224609376 ps

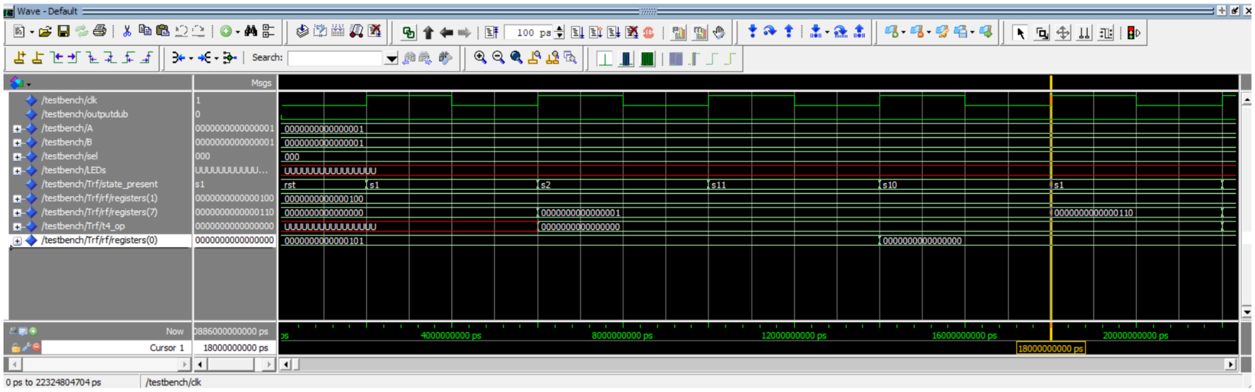
12)BEQ{1100000001001011}

Here the original value of PC i.e the initial value of R7 which is 0000000000000000 is stored in R0. Here Imm (sign extended 001011) multiplied by 2 which comes out to be 010110. This is added to R7 since R1=R2 and then it is stored in R7 ie pc is updated to 0000000000010110.



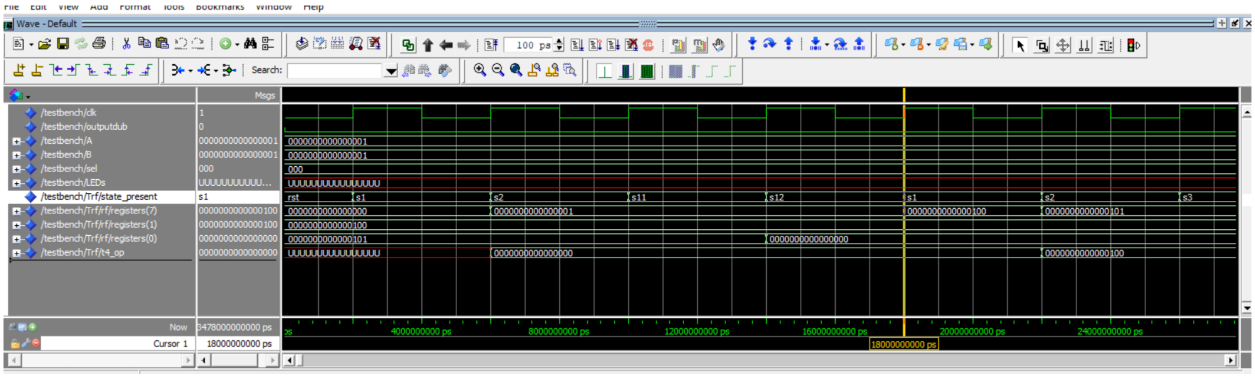
13)JAL{1101000000000011}

Here the original value of PC i.e the initial value of R7 which is 0000000000000000 is stored in R0. Here Imm (sign extended 000011) multiplied by 2 which comes out to be 6(in decimal) is added to r0 and then it is stored in r7 ie pc is updated to 0000000000000110.



14)JLR{1111000001000000}

Here the original value of PC i.e. value in R7(0000000000000000) is stored in r0 and then the value of r1 (i.e 0000000000000100) is stored in pc ie r7 which can also be framed as branching to the address in r1.



	Msgs
/testbench/clk	1
/testbench/outputdub	0
/testbench/A	0000000000000001
/testbench/B	0000000000000001
/testbench/sel	000
/testbench/LEDs	UUUUUUUUUUU...
/testbench/Trf/state_present	s1
/testbench/Trf/rf/registers(7)	0000000000000100
/testbench/Trf/rf/registers(1)	0000000000000100
/testbench/Trf/rf/registers(0)	0000000000000000
/testbench/Trf/t4_op	0000000000000000
Now	3478000000000 ps
Cursor 1	18000000000 ps

CPU CIRCUIT DIAGRAM

CONTRIBUTION

Sarvadnya Purkar(22B4232) and Prajwal Nayak(22B4246) : ➔

1. Circuit Designing
2. Memory Elements
3. Main CPU file
4. Testing
5. Report

Utkarsh Maurya(22B3910) and Samarth Sirsat(22B3988) : ➔

1. Circuit Designing
2. Optimization of the circuit to reduce states
3. All components except Main, Testing and Memory Elements
4. Report