EE 309 - MICROPROCESSORS

PIPELINED IITB-RISC

December 1, 2018

Kumar Ashutosh
Beni Madhav Agrawal
Mustafa Ali
Navnit Kumar

Contents

Introduction

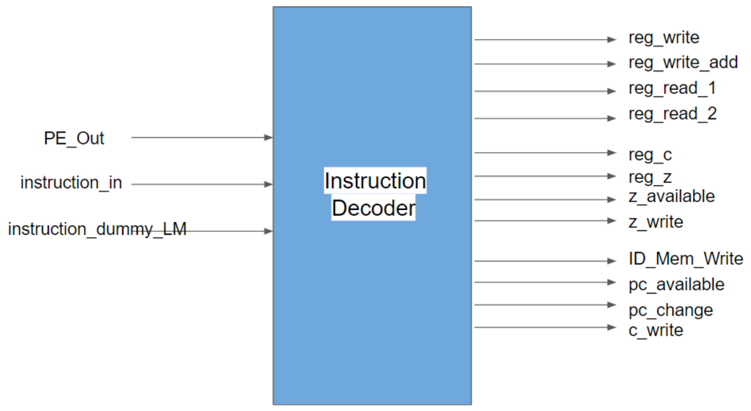
IITB-RISC is a simple yet powerful microprocessor. It has **8 registers**, each of **16 bits**. The pipelined version of IITB-RISC follows standard six Piprline stages - **Instruction Fetch**, **Instruction Decode**, **Register Read**, **Execution**, **Memory Access** and **Write Back**.

Data Path Components

The Data Path components in each stage is listed below along with the purpose and representative diagram.

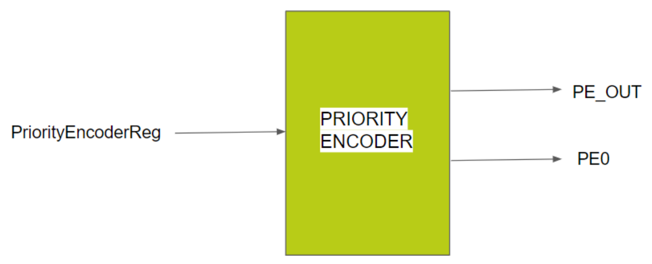
Instruction Decode Stage

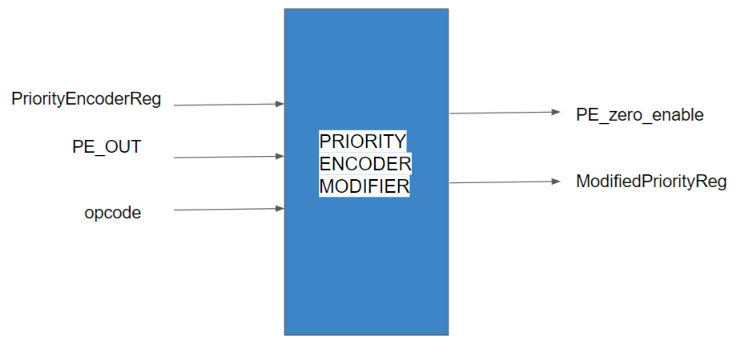
1. Instruction Decoder: Instruction Decoder takes into input the instruction and produces various control signals. This generated signal is propagated down the pipeline. The schematic diagram of the Instruction Decoder is given below.



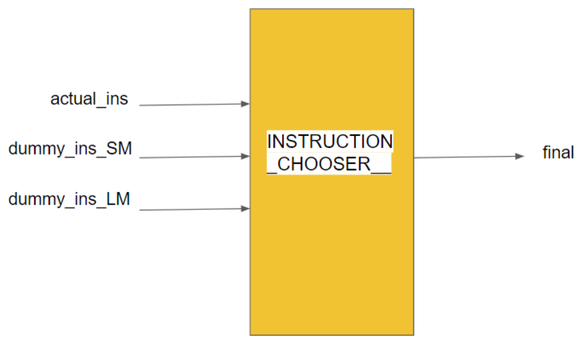
2. Priority Encoder and Priority Encoder Modifier: This component is useful for **LM** and **SM** instructions. Priority Encoder takes into input 8 bits (the last eight bits of LM/SM instruction) and gives the 3-bit index of the first non-zero bit. This is helpful in choosing which Register to Store/Load from.
   The Priority Encoder Modifier takes into input the output of Priority Encoder. It also takes the same 8 bits as input. It returns eight bit with the bit corresponding to Priority Encoder Output set to zero.
   Thus using the combination of Priority Encoder and Priority Encoder Modifier, we can

iteratively tackle the LM and SM instructions.





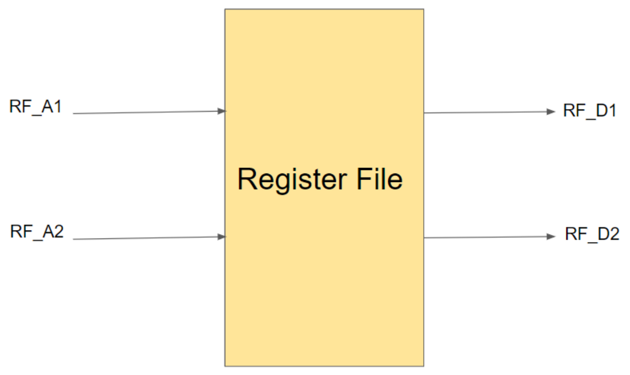
3. Instruction Chooser: For LM and SM, we first converted the LM instruction into multiple LW instructions and stalled the instruction fetch stage to obtain the desired behaviour. Hence we have an instruction chooser as a MUX to choose between the original instruction and the dummy LW and SW instructions.



Register Read Stage

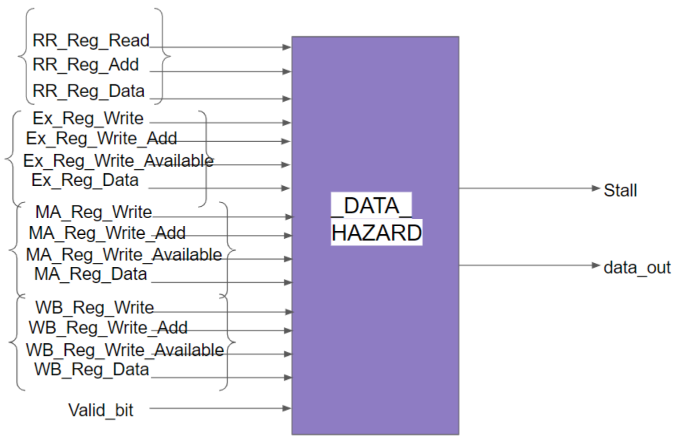
1. Register File: Register File is an essential component of the IITB-RISC Microprocessor. It has 8 registers which can be modified and accessed. The Register File is the interface

for this functionality. In the Register Read stage, we are only required to read the values (and not write). We have two input ports to the Register File and correspondingly two output ports. Each of the input-output combination is used to access the content of the registers. The schematic diagram of the register file is given below.

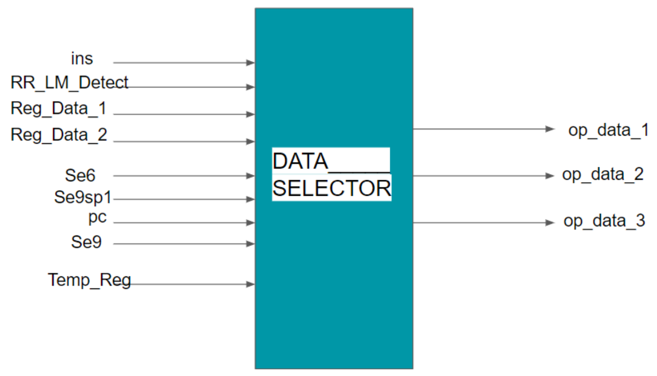


2. Sign Extenders: Many of the instruction requires addition of numbers of different bit length. Hence, we need a sign extender for the correctness of the expression. Hence we have **Sign Extended 6** and **Sign Extended 9** which extends a 6 bit and 9 bit inputs to 16 bits respectively. In addition to these two, we have another special sign extender in which the higher 9 bits are assigned the values and the lower 7 are assigned to 0.

3. Data Hazards: Due to the Pipelined Structure of the machine, we are having a lot of dependecies while performing various operations. Hence in the Register Read we have Hazard Mitigator which forwards the just calculated data to the RR stage for the current instruction to use the value already calculated but not written back to the Registers.



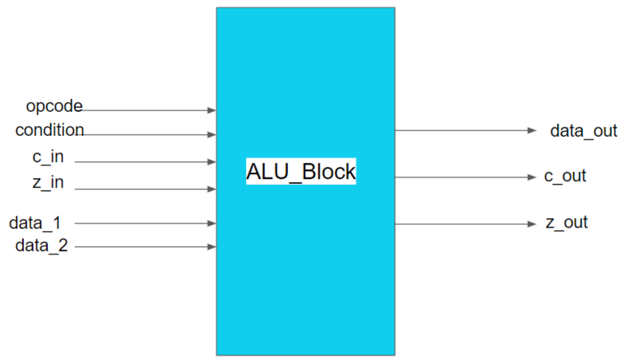
4. Data Selector: The Execution stage has ALU, so we use a MUX to channel the various data into 3 output ports. The first two would directly go to the input of the ALU and the third is used simply to move forward in the pipeline. It is used in some of the instructions including SW, SM.



5. BEQ Check: Since BEQ checks the equality of the two bits being reaf by the registers (after mitigating hazards), we have a single bit which checks the equality of the two data being provided as an input to the Stage.

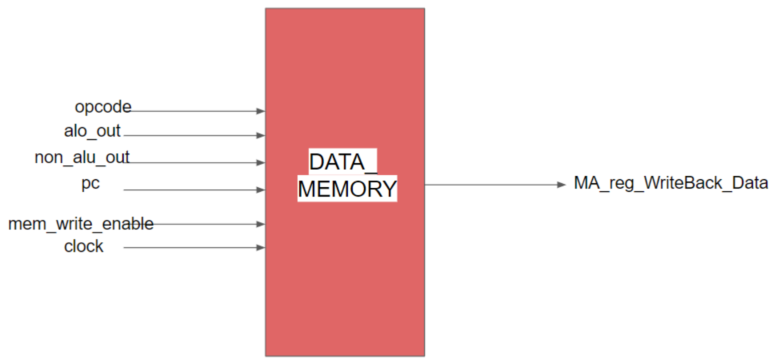
Execution Stage

1. ALU Block: ALU Block simples takes into input the two data values, the carry bit and the zero flag bit and also the instruction. The output is simply the desired ALU operation as specified in the instruction.



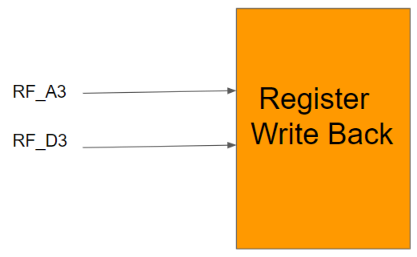
Memory Access Stage

1. Data Memory: This stage is meant for memory interactions. Data from memory is read or stored into the memory based on the instruction. There are two data inputs to this stage, one data and the other address. The write enables decides what operation to do, either write to memory or read from it. The writing operation is done only when the valid bit is set.



Write Back Stage

1. Register Write Back: This is the last write back stage which writes a data back to the Register File. In addition to this, it also stores the PC or PC+1 depending on whether the PC changed in the pipeline or not.



List of Pipeline Registers

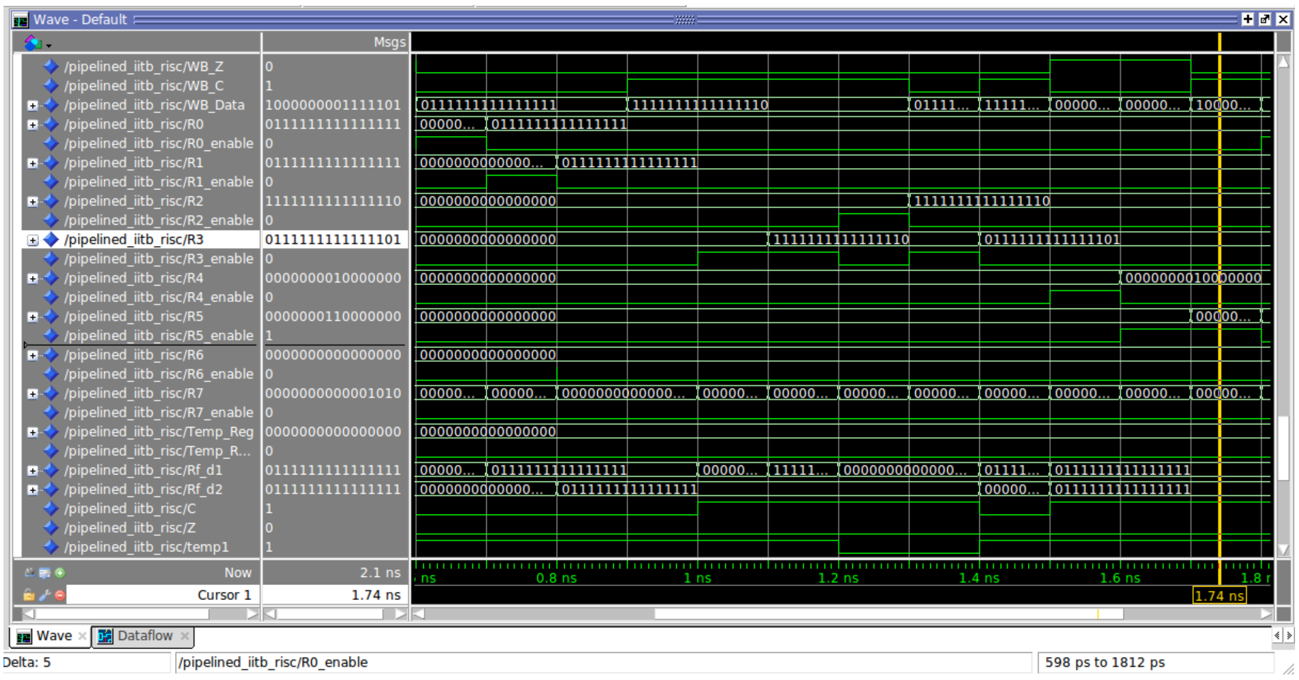
Several Control Signals were generated in the ID Stage of the pipeline and the same needs to be propagated down the pipeline. Some of the control signals and the values which gets modified in different stages of the pipeline are

- Instruction
  IF → ID → RR → EX → MA → WB
- Register Read : This bit tells if the instruction is reading from a register or not.
  ID → RR → EX → MA → WB
- Register Write : This bit tells if the instruction is writing to a register or not.
  ID → RR → EX → MA → WB
- PC Modify : This bit tells if the instruction is modifying PC or not.
  ID → RR → EX → MA → WB
- Z Modify : This bit tells if the instruction is modifying PC or not.
  ID → RR → EX → MA → WB
- Z Read : This bit tells if the instruction reads the value of Z.
  ID → RR → EX → MA → WB

SIMULATIONS

1. **LW, ADC, ADZ with dependency**

```
0   =>   "0100000110001011", --r0 = 7fff
1   =>   "0100001110001011", --r1 = 7fff
2   =>   "0000000001110001", --r6 = r1+r0 if z
3   =>   "0000000001011010", --r3 = r0+r1 if c
4   =>   "0000000001011010", --r3 = r0+r1 if c
5   =>   "0000000001010000", --r2 = r0+r1
6   =>   "0000010001011010", --r3 = r2+r1 if c
7   =>   "0000011001011010", --r3 = r3+r1 if c
8   =>   "0011100000000001",--r4 = 10000000
9   =>   "0011101000000011",--r5 = 110000000
10   =>   "0000011100101010", --r5 = r3 + r4 if c
```



	Msgs
/pipelined_iitb_risc/WB_Z	0
/pipelined_iitb_risc/WB_C	1
/pipelined_iitb_risc/WB_Data	1000000001111101
/pipelined_iitb_risc/R0	0111111111111111
/pipelined_iitb_risc/R0_enable	0
/pipelined_iitb_risc/R1	0111111111111111
/pipelined_iitb_risc/R1_enable	0
/pipelined_iitb_risc/R2	1111111111111110
/pipelined_iitb_risc/R2_enable	0
/pipelined_iitb_risc/R3	0111111111111101
/pipelined_iitb_risc/R3_enable	0
/pipelined_iitb_risc/R4	0000000010000000
/pipelined_iitb_risc/R4_enable	0
/pipelined_iitb_risc/R5	0000000110000000
/pipelined_iitb_risc/R5_enable	1
/pipelined_iitb_risc/R6	0000000000000000
/pipelined_iitb_risc/R6_enable	0
/pipelined_iitb_risc/R7	0000000000001010
/pipelined_iitb_risc/R7_enable	0
/pipelined_iitb_risc/Temp_Reg	0000000000000000
/pipelined_iitb_risc/Temp_R...	0
/pipelined_iitb_risc/Rf_d1	0111111111111111
/pipelined_iitb_risc/Rf_d2	0111111111111111
/pipelined_iitb_risc/C	1
/pipelined_iitb_risc/Z	0
/pipelined_iitb_risc/temp1	1
Now	2.1 ns
Cursor 1	1.74 ns

Delta: 5 /pipelined_iitb_risc/R0_enable 598 ps to 1812 ps

Figure 1: Simulation 1

2. **ALU followed by LOAD**
   LOAD followed by ALU
   LOAD followed by LOAD

```
0  =>  "0001000011000001",  --r3= 000001
1  =>  "0001010100000010", -- r4= 000010
2  =>  "0000011100010000", -- r2 = r3 +r4 = 11
3  =>  "0100001010000001", -- r1 = [r2 + 1] = [4] = 11
4  =>  "0000001010101000", -- r5 = r1+r2 = 110
5  =>  "0100011101000001", -- r3 = [r5+1] =[7] =1001
6  =>  "0100100101000001", -- r4 = [r5+1] =[7] =1001
7  =>  "0100100101000001", -- r4 = [r5+1] =[7] =1001
8  =>  "0100110100000001", -- r6 = [r4+1] =[10]= 11111111
```

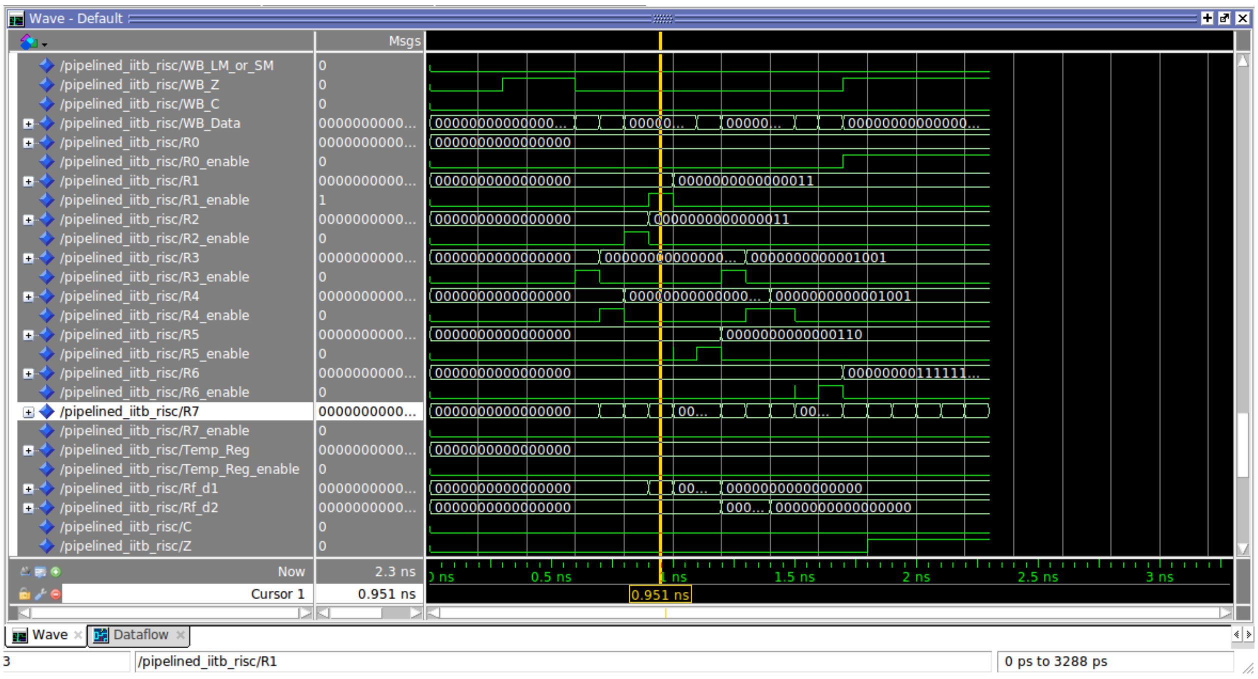
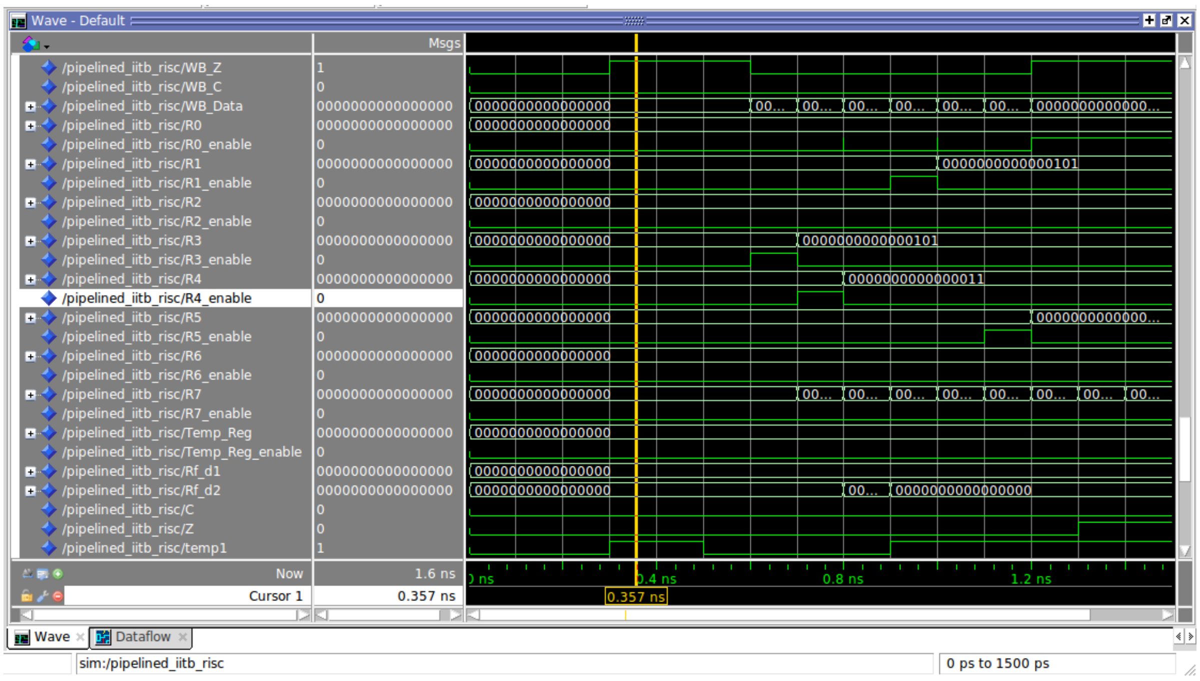

Wave - Default
Msgs
/pipelined_iitb_risc/WB_LM_or_SM
/pipelined_iitb_risc/WB_Z
/pipelined_iitb_risc/WB_C
/pipelined_iitb_risc/WB_Data
/pipelined_iitb_risc/R0
/pipelined_iitb_risc/R0_enable
/pipelined_iitb_risc/R1
/pipelined_iitb_risc/R1_enable
/pipelined_iitb_risc/R2
/pipelined_iitb_risc/R2_enable
/pipelined_iitb_risc/R3
/pipelined_iitb_risc/R3_enable
/pipelined_iitb_risc/R4
/pipelined_iitb_risc/R4_enable
/pipelined_iitb_risc/R5
/pipelined_iitb_risc/R5_enable
/pipelined_iitb_risc/R6
/pipelined_iitb_risc/R6_enable
/pipelined_iitb_risc/R7
/pipelined_iitb_risc/R7_enable
/pipelined_iitb_risc/Temp_Reg
/pipelined_iitb_risc/Temp_Reg_enable
/pipelined_iitb_risc/Rf_d1
/pipelined_iitb_risc/Rf_d2
/pipelined_iitb_risc/C
/pipelined_iitb_risc/Z
0000000000000011
0000000000001001
0000000000000110
000000000111111...
Now 2.3 ns
Cursor 1 0.951 ns
0 ns
0.5 ns
1 ns
1.5 ns
2 ns
2.5 ns
3 ns
Wave
Dataflow
/pipelined_iitb_risc/R1
0 ps to 3288 ps

Figure 2: Simulation 2

3. **STORE followed by LOAD followed by STORE**

```
0  =>  "0001000011000101",  --r3= 000101
1  =>  "0001010100000011", -- r4= 000011
2  =>  "0101011100000001",  -- [r4+1] = r3 =101
3  =>  "0100001000000100", -- r1 = [100] = 101
4  =>  "0101001100000000", -- [r4] = r1
5  =>  "0100101100000000", -- r5 = [r4]
```

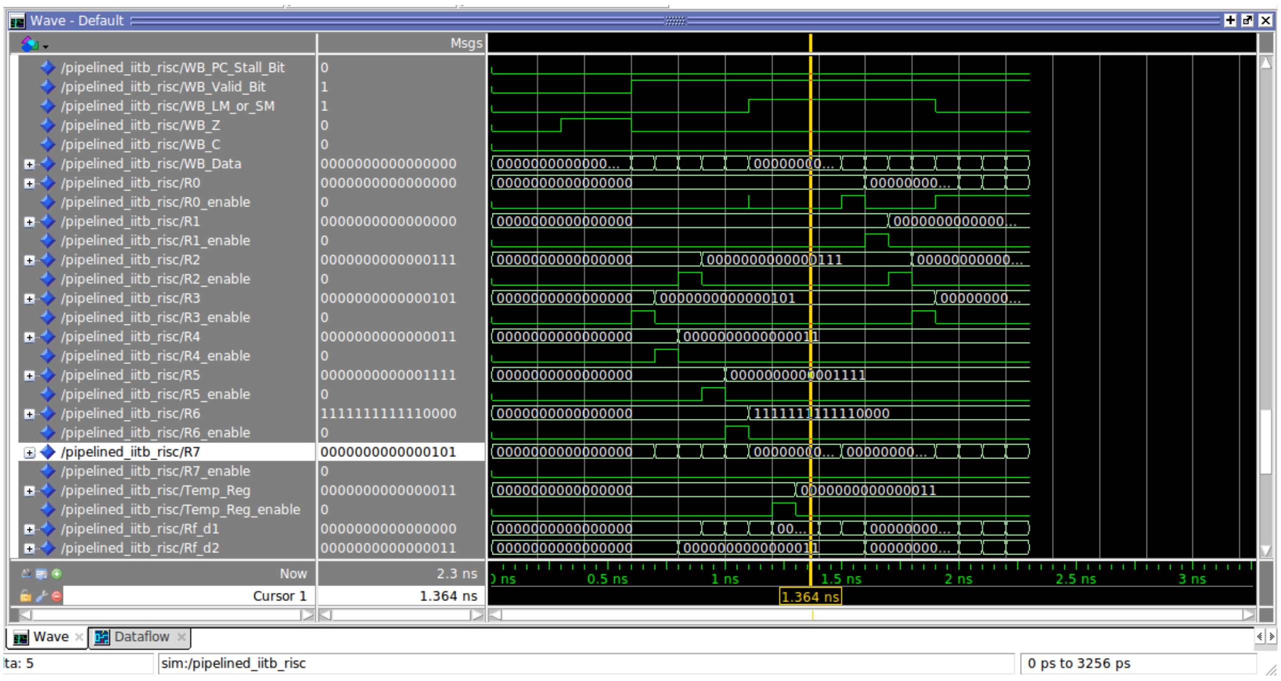


Wave - Default
Msgs
/pipelined_iitb_risc/WB_Z 1
/pipelined_iitb_risc/WB_C 0
/pipelined_iitb_risc/WB_Data 0000000000000000
/pipelined_iitb_risc/R0 0000000000000000
/pipelined_iitb_risc/R0_enable 0
/pipelined_iitb_risc/R1 0000000000000000
/pipelined_iitb_risc/R1_enable 0
/pipelined_iitb_risc/R2 0000000000000000
/pipelined_iitb_risc/R2_enable 0
/pipelined_iitb_risc/R3 0000000000000000
/pipelined_iitb_risc/R3_enable 0
/pipelined_iitb_risc/R4 0000000000000000
/pipelined_iitb_risc/R4_enable 0
/pipelined_iitb_risc/R5 0000000000000000
/pipelined_iitb_risc/R5_enable 0
/pipelined_iitb_risc/R6 0000000000000000
/pipelined_iitb_risc/R6_enable 0
/pipelined_iitb_risc/R7 0000000000000000
/pipelined_iitb_risc/R7_enable 0
/pipelined_iitb_risc/Temp_Reg 0000000000000000
/pipelined_iitb_risc/Temp_Reg_enable 0
/pipelined_iitb_risc/Rf_d1 0000000000000000
/pipelined_iitb_risc/Rf_d2 0000000000000000
/pipelined_iitb_risc/C 0
/pipelined_iitb_risc/Z 0
/pipelined_iitb_risc/temp1 1
0000000000000101
0000000000000011
Now 1.6 ns
Cursor 1 0.357 ns
0 ns
0.4 ns
0.8 ns
1.2 ns
0.357 ns
Wave
Dataflow
sim:/pipelined_iitb_risc
0 ps to 1500 ps

Figure 3: Simulation 3

4. **SM followed by SM**

```
0  =>  "0001000011000101",  --r3= 000101
1  =>  "0001000100000011", -- r4= 000011
2  =>  "0001000010000111", -- r2= 000111
3  =>  "0001000101001111", -- r5= 001111
4  =>  "0001000110110000", -- r6= 110000
5  =>  "0111100000111100",  --sm
6  =>  "0110100000001111",  --lm
```

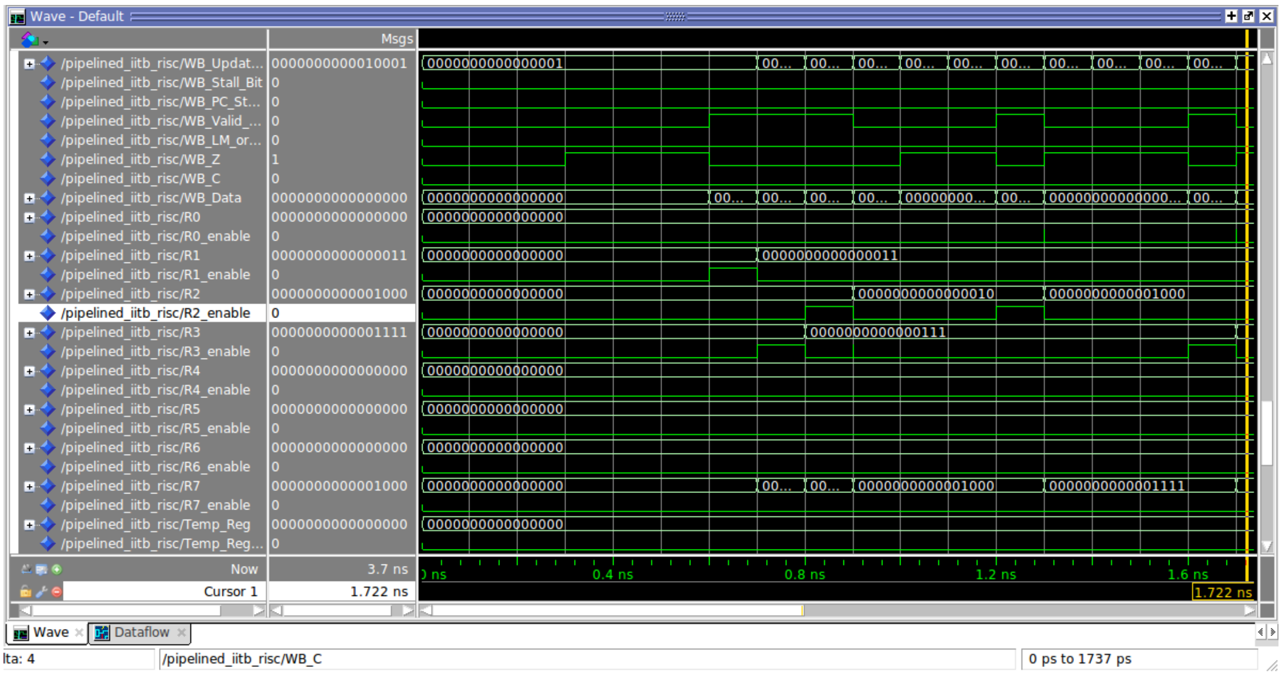


```
Wave - Default
                                          Msgs
/pipelined_iitb_risc/WB_PC_Stall_Bit      0
/pipelined_iitb_risc/WB_Valid_Bit         1
/pipelined_iitb_risc/WB_LM_or_SM          1
/pipelined_iitb_risc/WB_Z                 0
/pipelined_iitb_risc/WB_C                 0
/pipelined_iitb_risc/WB_Data              0000000000000000
/pipelined_iitb_risc/R0                   0000000000000000
/pipelined_iitb_risc/R0_enable            0
/pipelined_iitb_risc/R1                   0000000000000000
/pipelined_iitb_risc/R1_enable            0
/pipelined_iitb_risc/R2                   0000000000000111
/pipelined_iitb_risc/R2_enable            0
/pipelined_iitb_risc/R3                   0000000000000101
/pipelined_iitb_risc/R3_enable            0
/pipelined_iitb_risc/R4                   0000000000000011
/pipelined_iitb_risc/R4_enable            0
/pipelined_iitb_risc/R5                   0000000000001111
/pipelined_iitb_risc/R5_enable            0
/pipelined_iitb_risc/R6                   1111111111110000
/pipelined_iitb_risc/R6_enable            0
/pipelined_iitb_risc/R7                   0000000000000101
/pipelined_iitb_risc/R7_enable            0
/pipelined_iitb_risc/Temp_Reg             0000000000000011
/pipelined_iitb_risc/Temp_Reg_enable      0
/pipelined_iitb_risc/Rf_d1                0000000000000000
/pipelined_iitb_risc/Rf_d2                0000000000000011
                                  Now     2.3 ns
                             Cursor 1     1.364 ns
ta: 5     sim:/pipelined_iitb_risc     0 ps to 3256 ps
```

Figure 4: Simulation 4

5. **JAL**

```
0  =>   "0001000001000011",  --r1= 000011
1  =>   "0001000011000111",  --r3= 000111
2  =>   "1000010000000110", -- r2 = 10
3  =>   "0001000011000001",  --r3= 000001
7  =>   "0000010011100000",  --r4= r2+r3
8  =>   "1000010000000111", -- r2 = 8
15 =>   "1001011010000000", -- r3 = 1111
```

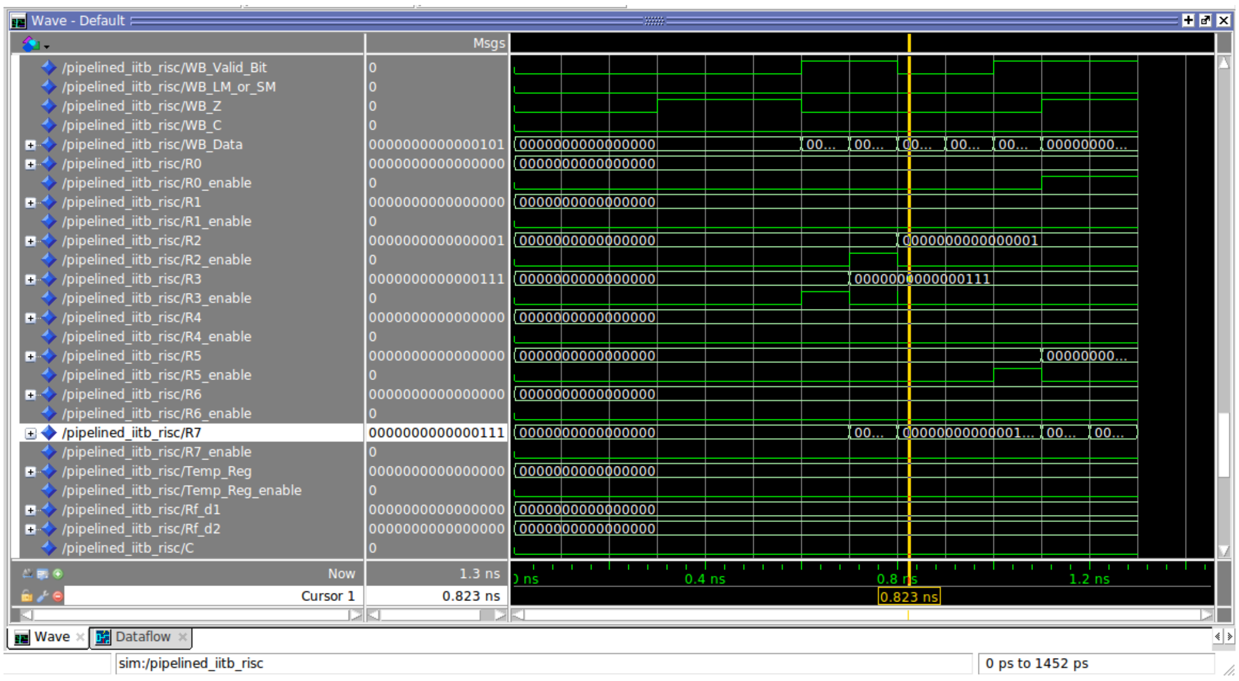


	Msgs
/pipelined_iitb_risc/WB_Updat...	0000000000010001
/pipelined_iitb_risc/WB_Stall_Bit	0
/pipelined_iitb_risc/WB_PC_St...	0
/pipelined_iitb_risc/WB_Valid_...	0
/pipelined_iitb_risc/WB_LM_or...	0
/pipelined_iitb_risc/WB_Z	1
/pipelined_iitb_risc/WB_C	0
/pipelined_iitb_risc/WB_Data	0000000000000000
/pipelined_iitb_risc/R0	0000000000000000
/pipelined_iitb_risc/R0_enable	0
/pipelined_iitb_risc/R1	0000000000000011
/pipelined_iitb_risc/R1_enable	0
/pipelined_iitb_risc/R2	0000000000001000
/pipelined_iitb_risc/R2_enable	0
/pipelined_iitb_risc/R3	0000000000001111
/pipelined_iitb_risc/R3_enable	0
/pipelined_iitb_risc/R4	0000000000000000
/pipelined_iitb_risc/R4_enable	0
/pipelined_iitb_risc/R5	0000000000000000
/pipelined_iitb_risc/R5_enable	0
/pipelined_iitb_risc/R6	0000000000000000
/pipelined_iitb_risc/R6_enable	0
/pipelined_iitb_risc/R7	0000000000001000
/pipelined_iitb_risc/R7_enable	0
/pipelined_iitb_risc/Temp_Reg	0000000000000000
/pipelined_iitb_risc/Temp_Reg...	0
Now	3.7 ns
Cursor 1	1.722 ns

Figure 5: Simulation 5

6. **JLR**

```
0  =>   "0001000011000111",  --r3= 000111
1  =>   "1001010011000000", -- r2 = 1
2  =>   "0001000011000101",  --r3= 000101
3  =>   "0001000110000111",  --r6= 000111
6  =>   "0001000110001111",  --r6= 001111
7  =>   "0001000101011111",  --r5= 011111
```

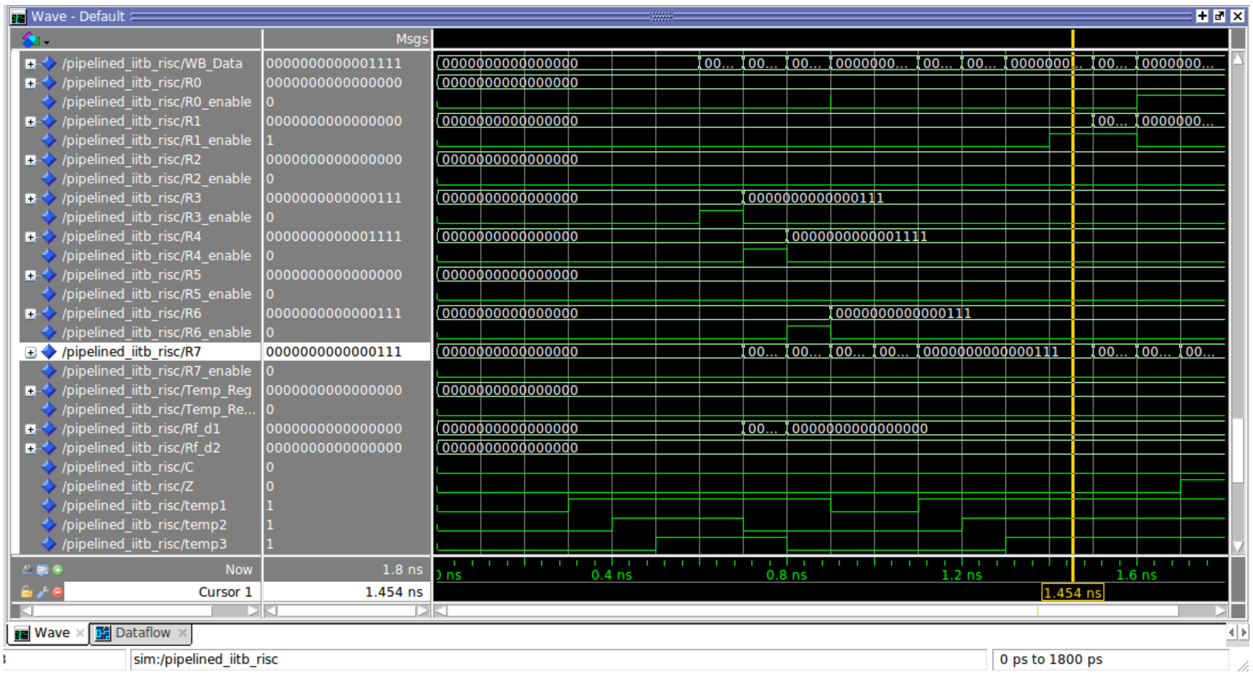


```
/pipelined_iitb_risc/WB_Valid_Bit       0
/pipelined_iitb_risc/WB_LM_or_SM        0
/pipelined_iitb_risc/WB_Z               0
/pipelined_iitb_risc/WB_C               0
/pipelined_iitb_risc/WB_Data            0000000000000101
/pipelined_iitb_risc/R0                 0000000000000000
/pipelined_iitb_risc/R0_enable          0
/pipelined_iitb_risc/R1                 0000000000000000
/pipelined_iitb_risc/R1_enable          0
/pipelined_iitb_risc/R2                 0000000000000001
/pipelined_iitb_risc/R2_enable          0
/pipelined_iitb_risc/R3                 0000000000000111
/pipelined_iitb_risc/R3_enable          0
/pipelined_iitb_risc/R4                 0000000000000000
/pipelined_iitb_risc/R4_enable          0
/pipelined_iitb_risc/R5                 0000000000000000
/pipelined_iitb_risc/R5_enable          0
/pipelined_iitb_risc/R6                 0000000000000000
/pipelined_iitb_risc/R6_enable          0
/pipelined_iitb_risc/R7                 0000000000000111
/pipelined_iitb_risc/R7_enable          0
/pipelined_iitb_risc/Temp_Reg           0000000000000000
/pipelined_iitb_risc/Temp_Reg_enable    0
/pipelined_iitb_risc/Rf_d1              0000000000000000
/pipelined_iitb_risc/Rf_d2              0000000000000000
/pipelined_iitb_risc/C                  0
```

Figure 6: Simulation 6

7. **BEQ with dependencies**

```
0  =>  "0001000011000111",  --r3= 000111
1  =>  "0001010100001111", -- r4= 001111
2  =>  "0001010110000111", --r6=000111
3  =>  "1100100110000011",-- beq r4,r6,3 if true jump to 6
4  =>  "1100011110000011",-- beq r3,r6,3 if true jump to 7
5  =>  "0001000001000011",  --r1= 000011
6  =>  "0001000001000111",  --r1= 000111
7  =>  "0001000001001111",  --r1= 001111
8  =>  "0001000001011111",  --r1= 011111
```

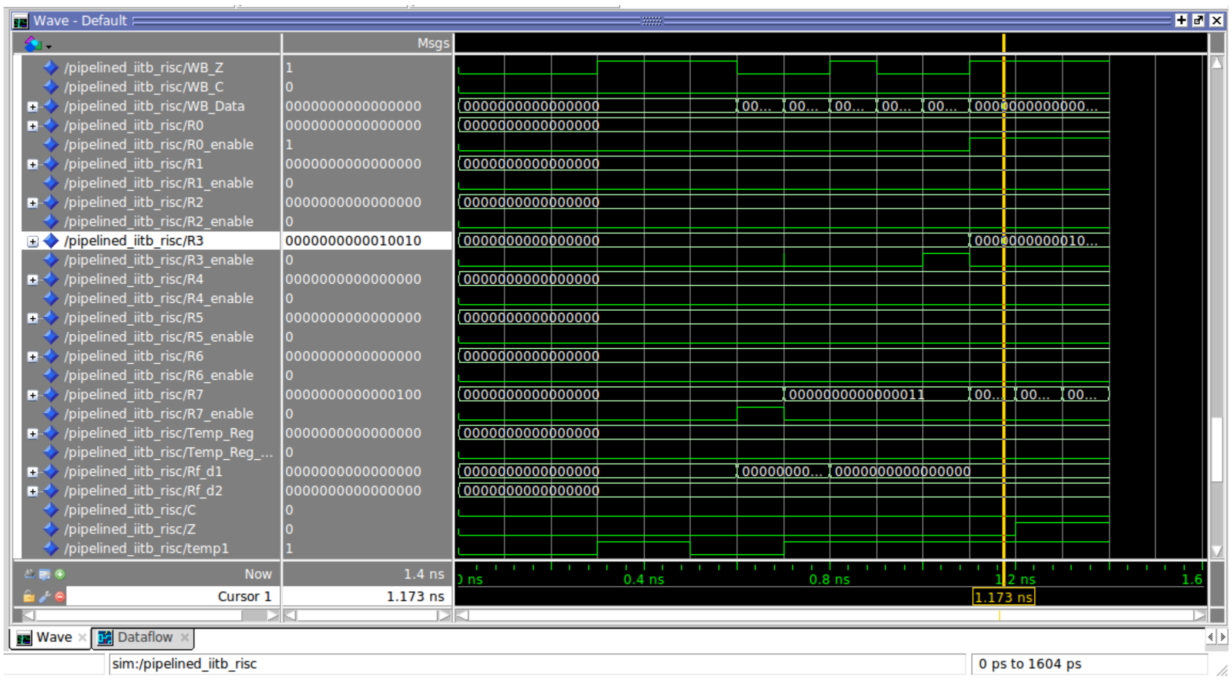


	Msgs
/pipelined_iitb_risc/WB_Data	0000000000001111
/pipelined_iitb_risc/R0	0000000000000000
/pipelined_iitb_risc/R0_enable	0
/pipelined_iitb_risc/R1	0000000000000000
/pipelined_iitb_risc/R1_enable	1
/pipelined_iitb_risc/R2	0000000000000000
/pipelined_iitb_risc/R2_enable	0
/pipelined_iitb_risc/R3	0000000000000111
/pipelined_iitb_risc/R3_enable	0
/pipelined_iitb_risc/R4	0000000000001111
/pipelined_iitb_risc/R4_enable	0
/pipelined_iitb_risc/R5	0000000000000000
/pipelined_iitb_risc/R5_enable	0
/pipelined_iitb_risc/R6	0000000000000111
/pipelined_iitb_risc/R6_enable	0
/pipelined_iitb_risc/R7	0000000000000111
/pipelined_iitb_risc/R7_enable	0
/pipelined_iitb_risc/Temp_Reg	0000000000000000
/pipelined_iitb_risc/Temp_Re...	0
/pipelined_iitb_risc/Rf_d1	0000000000000000
/pipelined_iitb_risc/Rf_d2	0000000000000000
/pipelined_iitb_risc/C	0
/pipelined_iitb_risc/Z	0
/pipelined_iitb_risc/temp1	1
/pipelined_iitb_risc/temp2	1
/pipelined_iitb_risc/temp3	1
Now	1.8 ns
Cursor 1	1.454 ns

Figure 7: Simulation 7

8. **R7 updation using ADI**

```
0  =>  "0001000111000011",  --r7= 000011
1  =>  "0001000011000111",  --r3= 000111
3  =>  "0001111011001111",  --r3= 001111 + r7 = 10010
```



```
/pipelined_iitb_risc/WB_Z            1
/pipelined_iitb_risc/WB_C            0
/pipelined_iitb_risc/WB_Data         0000000000000000
/pipelined_iitb_risc/R0              0000000000000000
/pipelined_iitb_risc/R0_enable       1
/pipelined_iitb_risc/R1              0000000000000000
/pipelined_iitb_risc/R1_enable       0
/pipelined_iitb_risc/R2              0000000000000000
/pipelined_iitb_risc/R2_enable       0
/pipelined_iitb_risc/R3              0000000000010010
/pipelined_iitb_risc/R3_enable       0
/pipelined_iitb_risc/R4              0000000000000000
/pipelined_iitb_risc/R4_enable       0
/pipelined_iitb_risc/R5              0000000000000000
/pipelined_iitb_risc/R5_enable       0
/pipelined_iitb_risc/R6              0000000000000000
/pipelined_iitb_risc/R6_enable       0
/pipelined_iitb_risc/R7              0000000000000100
/pipelined_iitb_risc/R7_enable       0
/pipelined_iitb_risc/Temp_Reg        0000000000000000
/pipelined_iitb_risc/Temp_Reg_...    0
/pipelined_iitb_risc/Rf_d1           0000000000000000
/pipelined_iitb_risc/Rf_d2           0000000000000000
/pipelined_iitb_risc/C               0
/pipelined_iitb_risc/Z               0
/pipelined_iitb_risc/temp1           1
Now        1.4 ns
Cursor 1   1.173 ns
```

Figure 8: Simulation 8