Cairo University
Faculty of Engineering

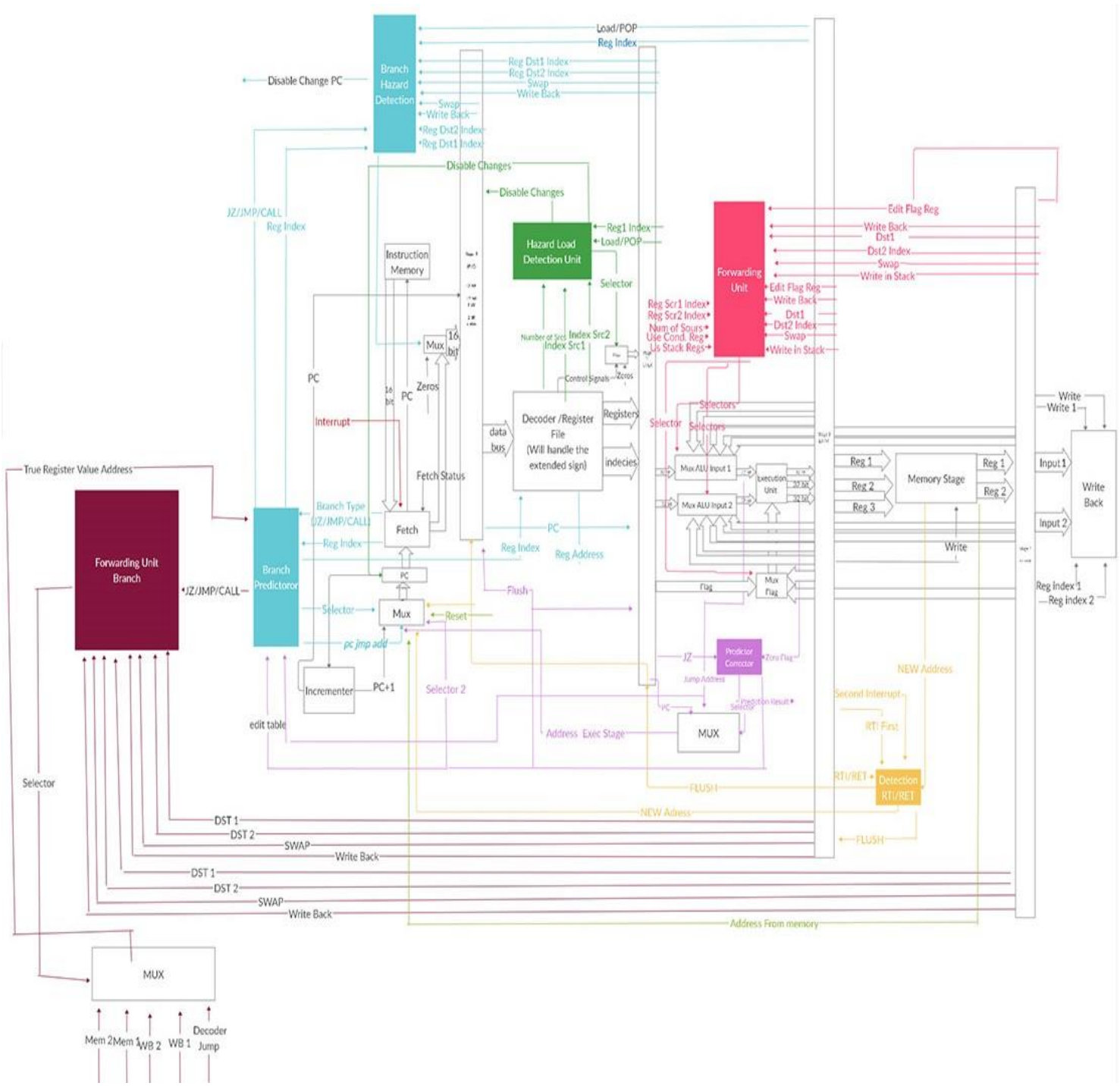
Project Arch

Final delivery

team 1

Name	Section	B.N
Aly Ramzy Hassan	1	36
Mouhamed Khier	2	12
Nour Ahmed	2	31
Hager Ahmed	2	34

Full design:


Load/POP
Reg Index
Branch Hazard Detection
Disable Change PC
Reg Dst1 Index
Reg Dst2 Index
Swap
Write Back
Disable Changes
JZ/JMP/CALL
Reg Index
Hazard Load Detection Unit
Reg1 Index
Load/POP
Selector
Instruction Memory
Forwarding Unit
Edit Flag Reg
Write Back
Dst1
Dst2 Index
Swap
Write in Stack
Reg Scr1 Index
Reg Scr2 Index
Num of Sours
Use Cond. Reg
Is Stack Regs
Mux
16 bit
Index Src2
Index Src1
Zeros
PC
Interrupt
Control Signals
Decoder /Register File (Will handle the extended sign)
Registers
indecies
data bus
Selectors
Write
Write 1
Input 1
Input 2
Write Back
Reg index 1
Reg index 2
True Register Value Address
Forwarding Unit Branch
JZ/JMP/CALL
Branch Predictoror
Branch Type (JZ/JMP/CALL)
Fetch
Fetch Status
Reg Address
Flush
Reset
pc jmp add
Incrementer
PC+1
Selector 2
edit table
Mux ALU Input 1
Mux ALU Input 2
Execution Unit
Reg 1
Reg 2
Reg 3
Memory Stage
Mux Flag
Flag
JZ
Predictor Corrector
Jump Address
Prediction result
NEW Address
Second Interrupt
RTI First
RTI/RET
Detection RTI/RET
FLUSH
NEW Adress
Address Exec Stage
MUX
Address From memory
DST 1
DST 2
SWAP
Write Back
Mem 2
Mem 1
WB 2
WB 1
Decoder
Jump

The unfinished parts of our processor:

1- Cash memory
2- Branch prediction

The Modified Test case:

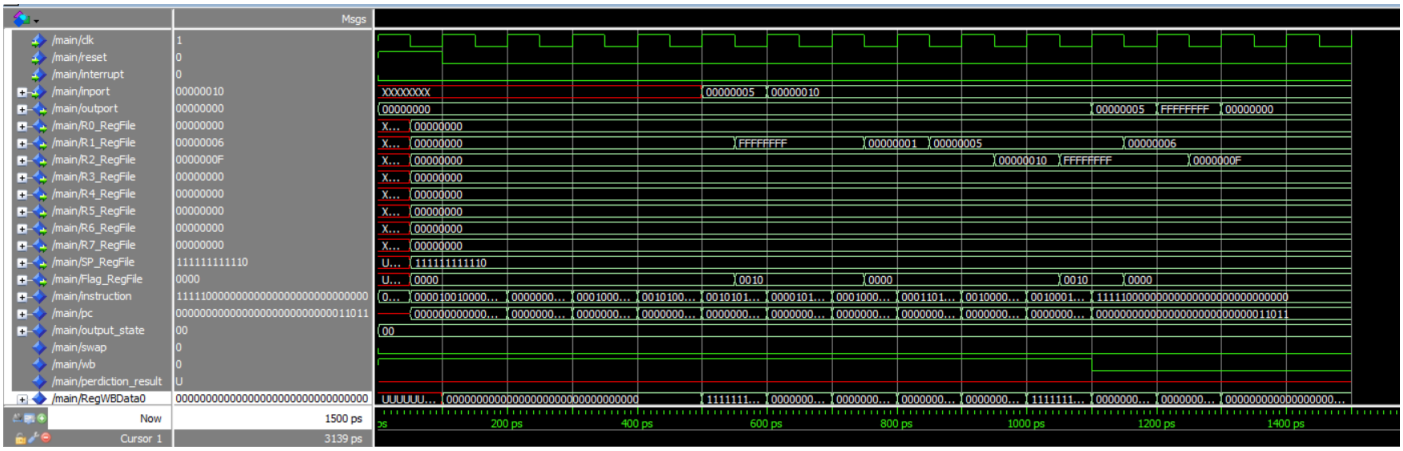
- We modified the branch test case by adding one pop instructions after each interrupt

  Because we have some issue in interrupt and the new test case in the final

  assessment folder

One operand file:

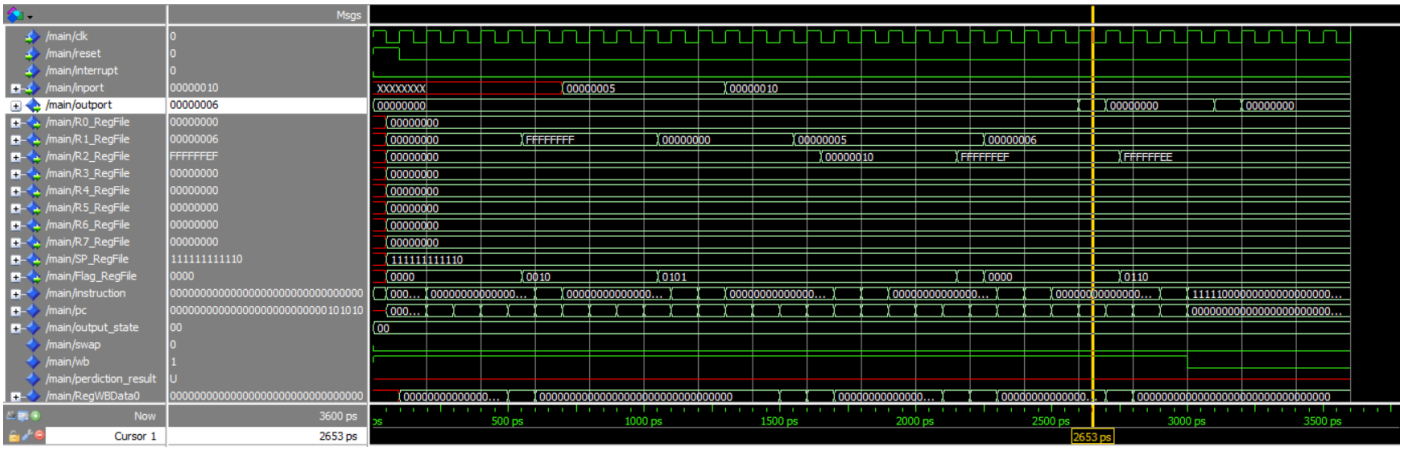
Before forwarding without no operation

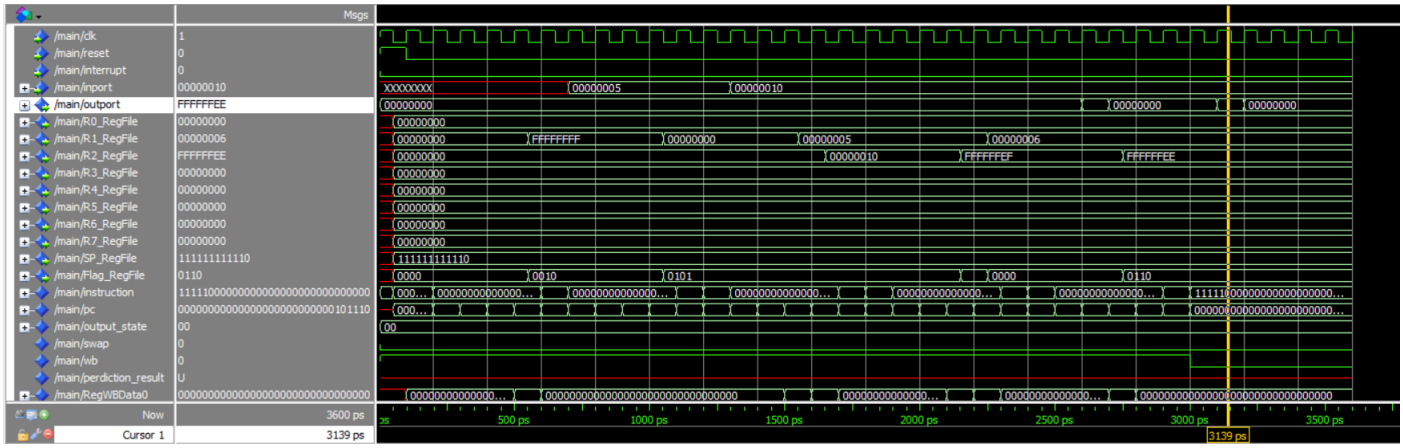


Hazards faced without forwarding:

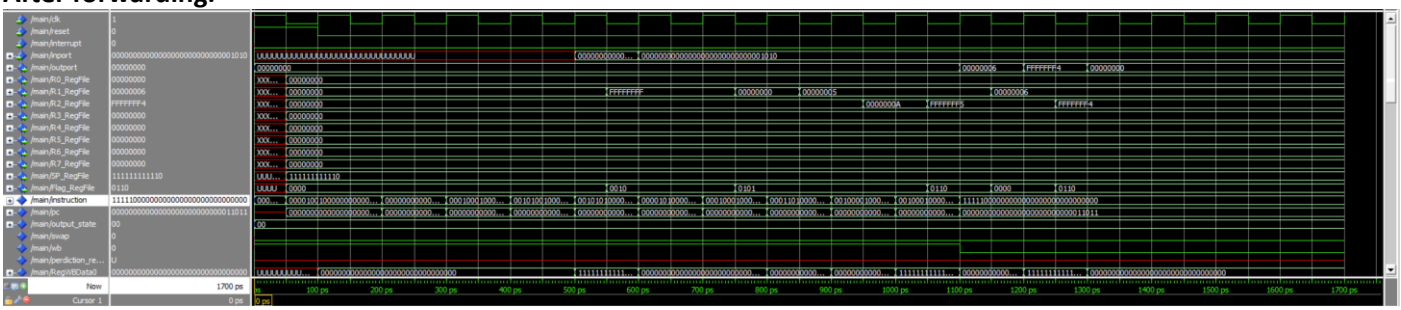
1- Inc R1 hazard is read after write.
2- Not R2 hazard is read after write.
3- Dec R2 hazard is read after write.
4- Out R1 hazard is read after write.
5- Out R2 hazard is read after write.

Before forwarding with no operation(solving the hazards with nop)



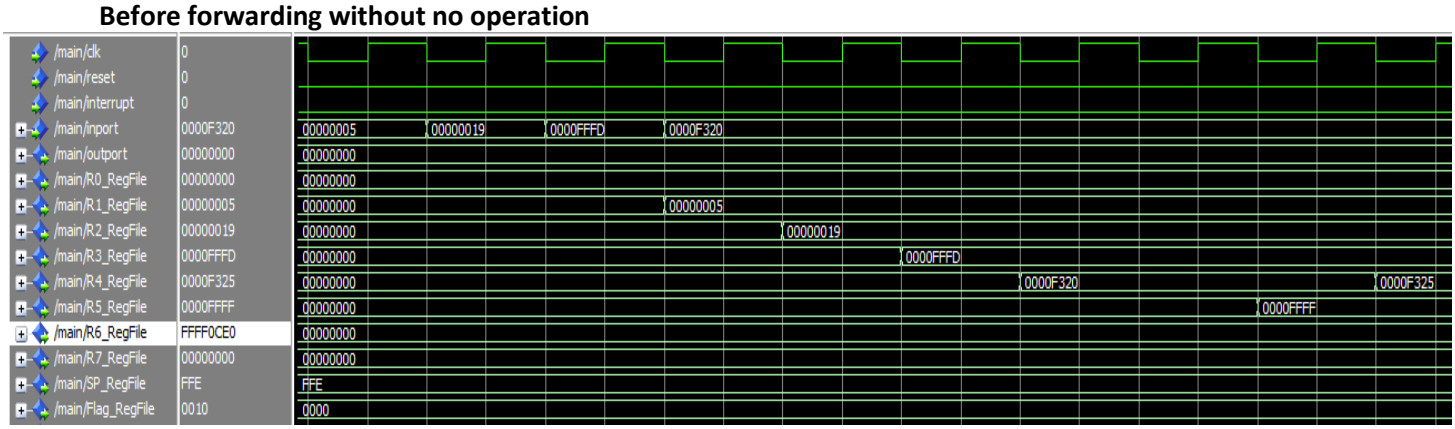


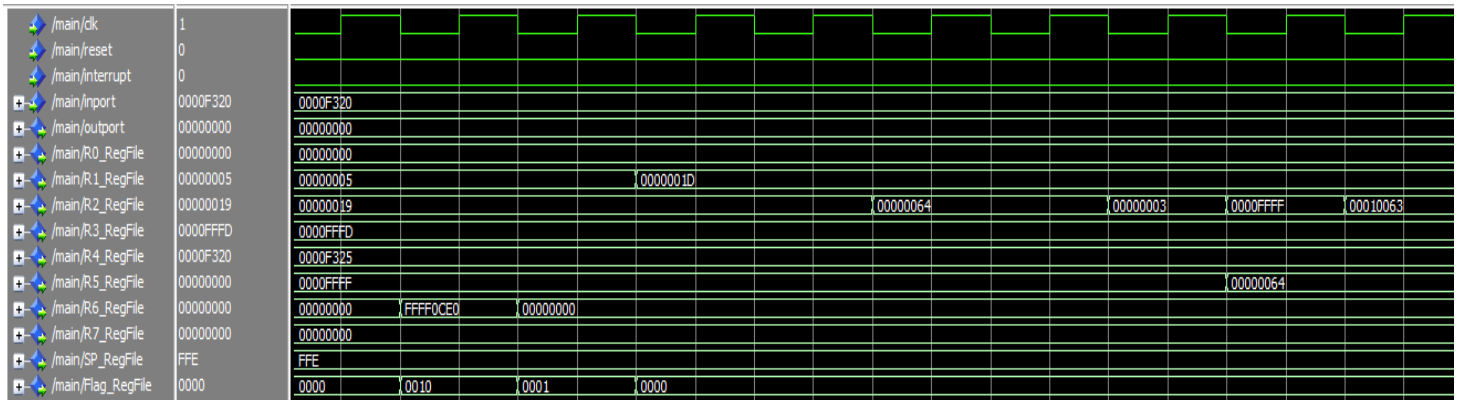
After forwarding:



Two operand file:

Before forwarding without no operation

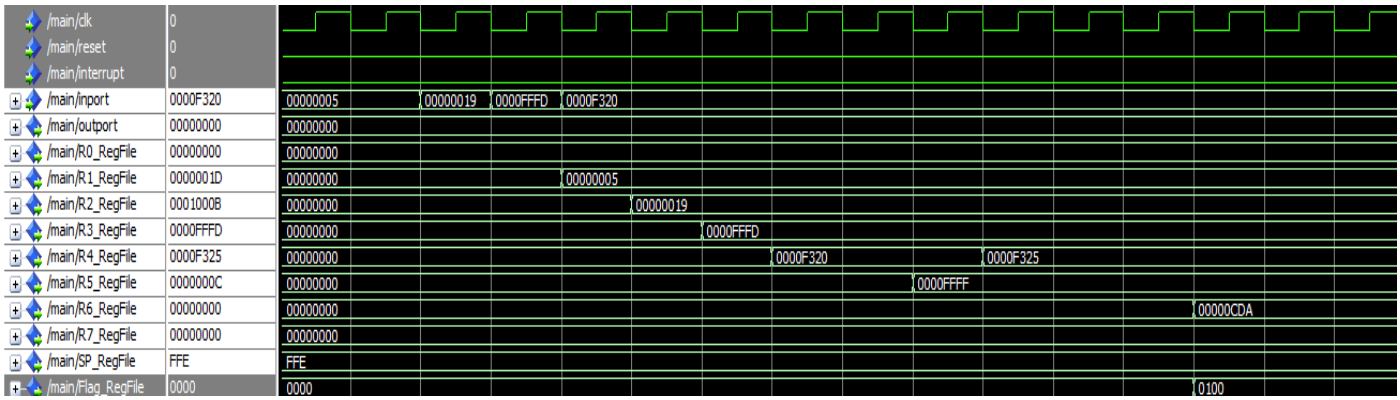


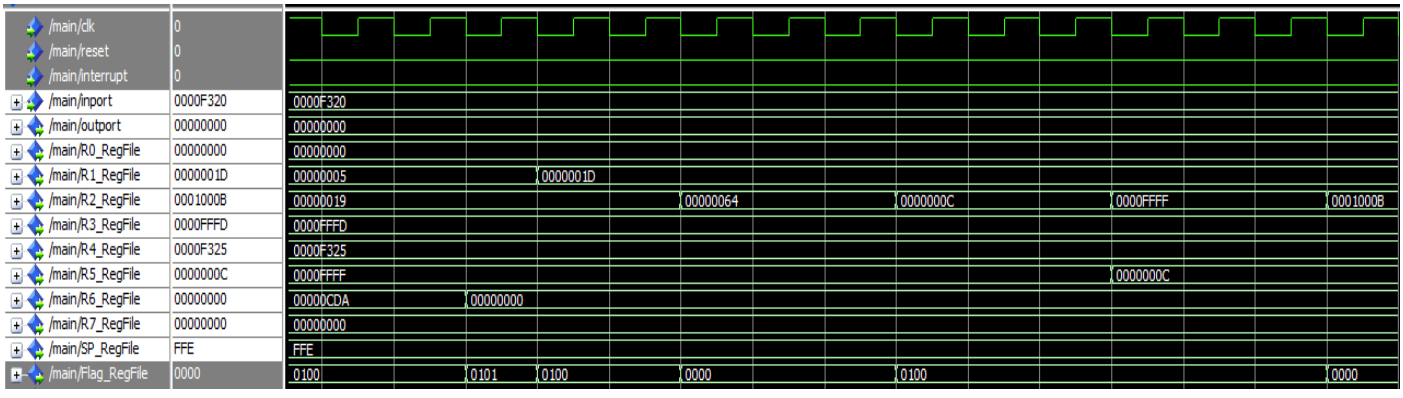


Hazards faced without forwarding:

1. SUB R5,R4,R6 hazard is read after write R4 & R5
2. AND R7,R6,R6 hazard is read after write R6
3. SHR R2,3 hazard is read after write R2
4. SWAP R2,R5 hazard is read after write R2
5. ADD R5,R2,R2 hazard is read after write R2 & R5

Before forwarding with no operation(solving the hazards with nop)

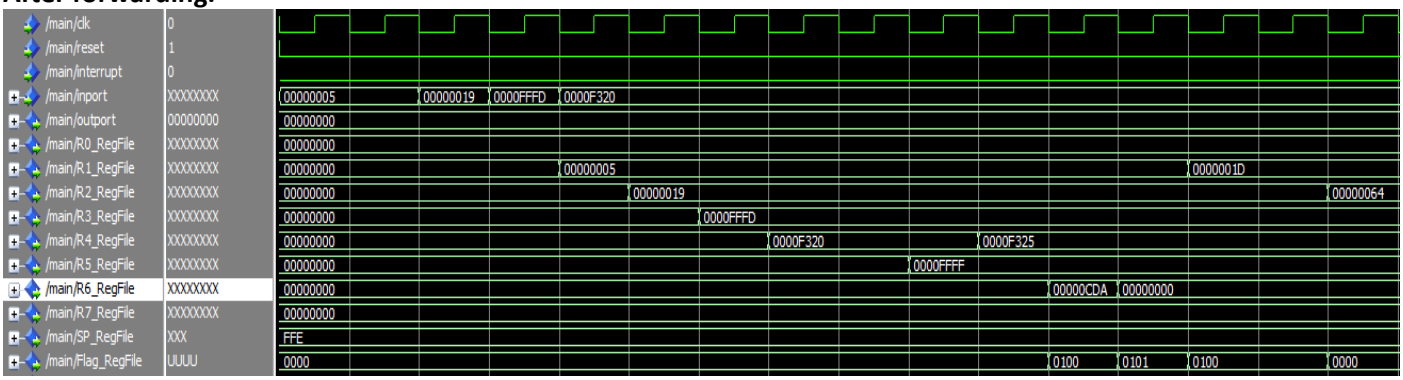


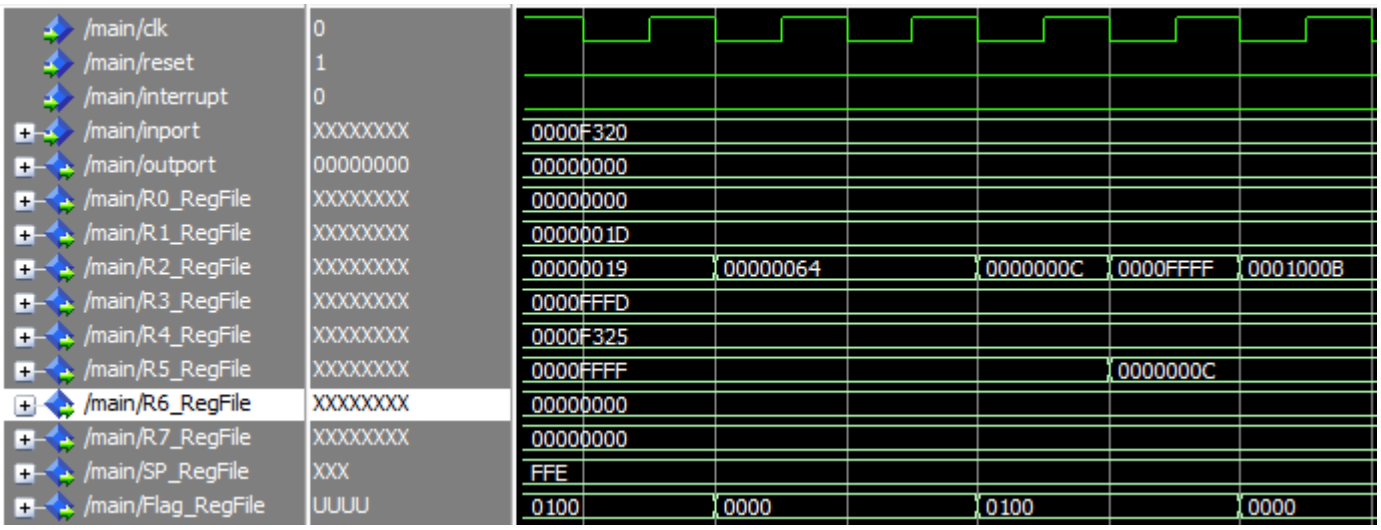


```
.ORG 2  #this is the interrupt address
100

.ORG 10
in   R1       #add 5 in R1
in   R2       #add 19 in R2
in   R3       #FFFD
in   R4       #F320
IADD R3,R5,2  #R5 = FFFF , flags no change
ADD  R1,R4,R4    #R4= F325 , C-->0, N-->0, Z-->0
NOP
NOP
SUB  R5,R4,R6    #R6= 0CDA , C-->1, N-->0,Z-->0
NOP
NOP
AND  R7,R6,R6    #R6= 00000000 , C-->no change, N-->0, Z-->1
OR   R2,R1,R1    #R1=1D  , C--> no change, N-->0, Z--> 0
SHL  R2,2     #R2=64  , C--> 0, N -->0 , Z -->0
NOP
SHR  R2,3     #R2=0C  , C -->1, N-->0 , Z-->0
NOP
NOP
SWAP R2,R5    #R5=0C ,R2=FFFF  ,no change for flags
NOP
NOP
ADD  R5,R2,R2    #R2= 1000B (C,N,Z= 0)
```

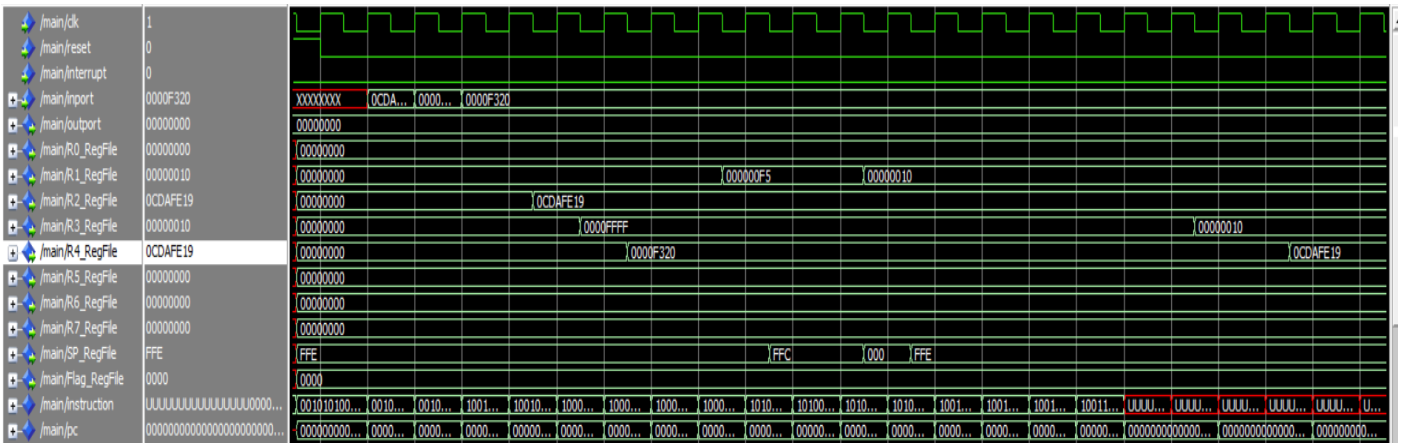
After forwarding:



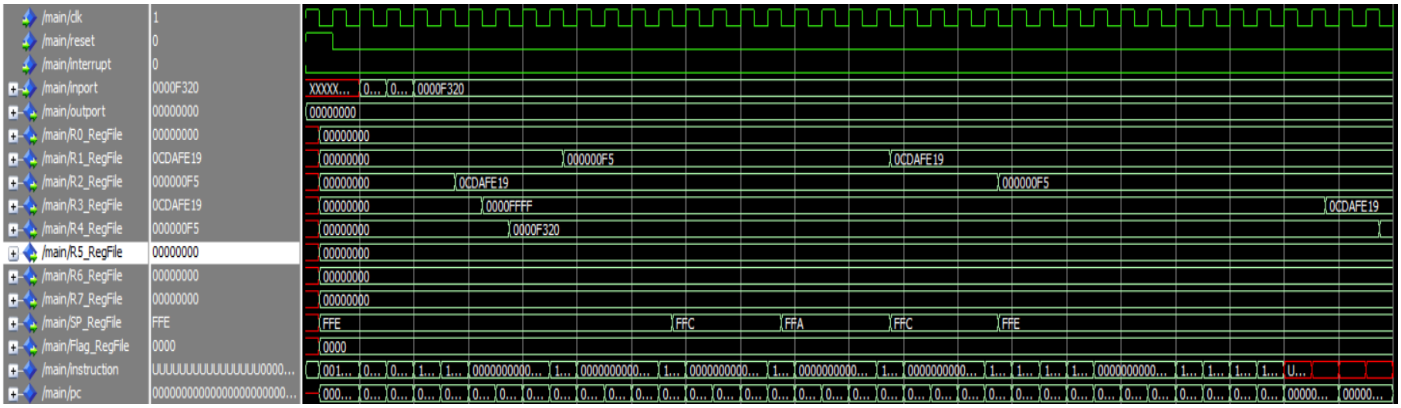


Memory file:

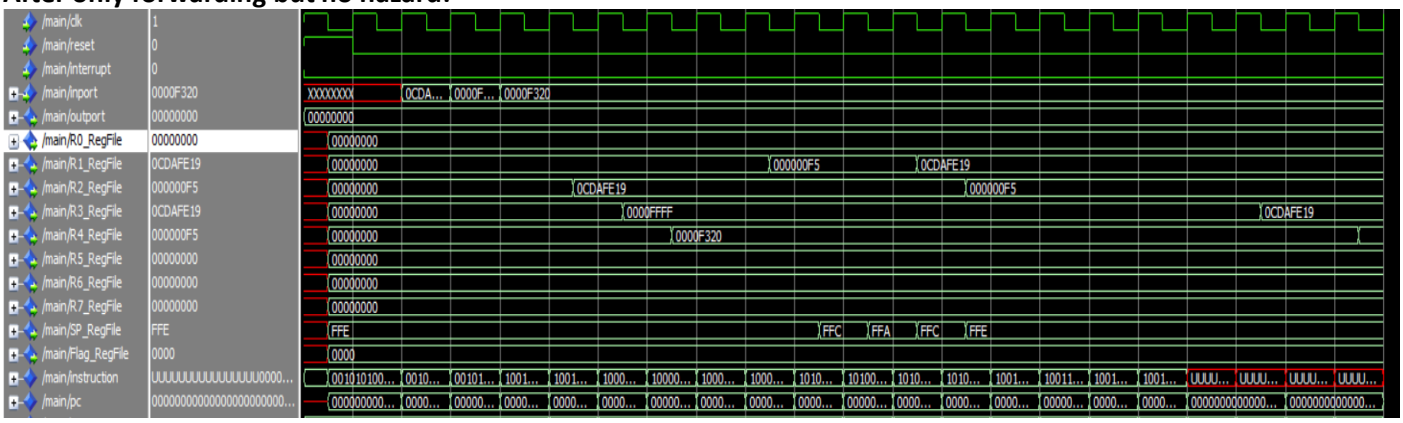
Before forwarding and hazard without no operation



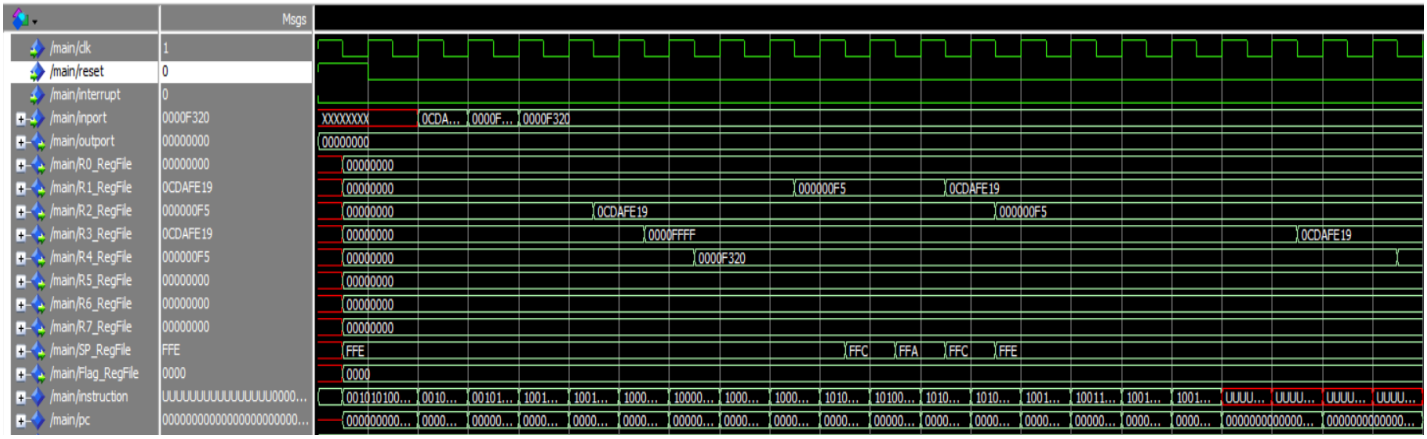
Before forwarding and hazard with no operation(solving the hazards with nop)



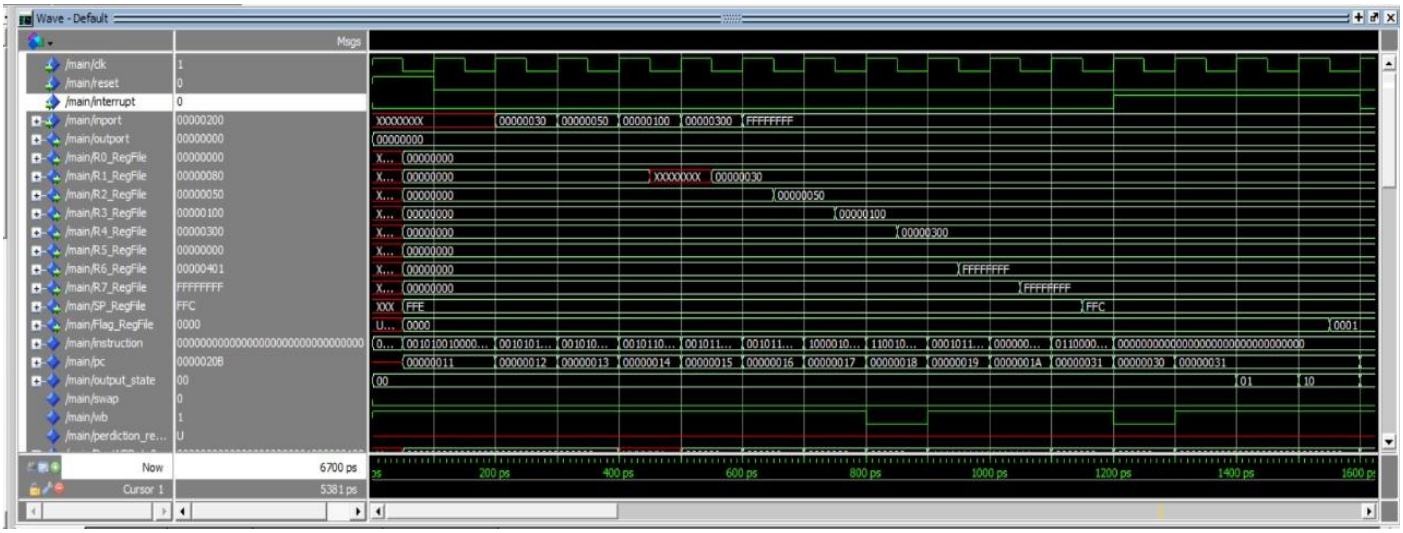
After only forwarding but no hazard:

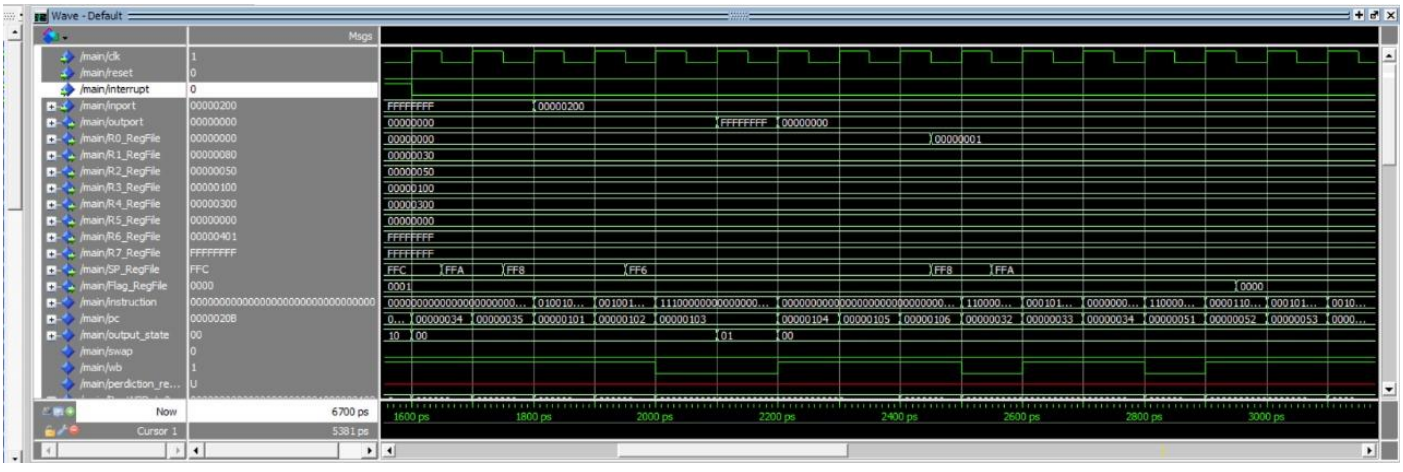


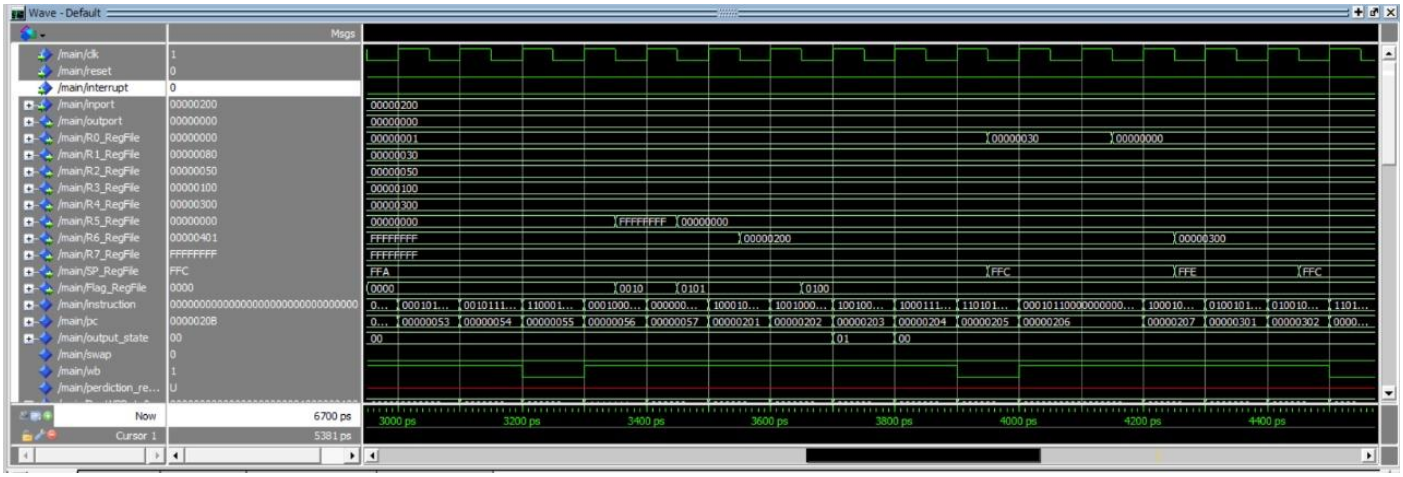
After forwarding and hazard:

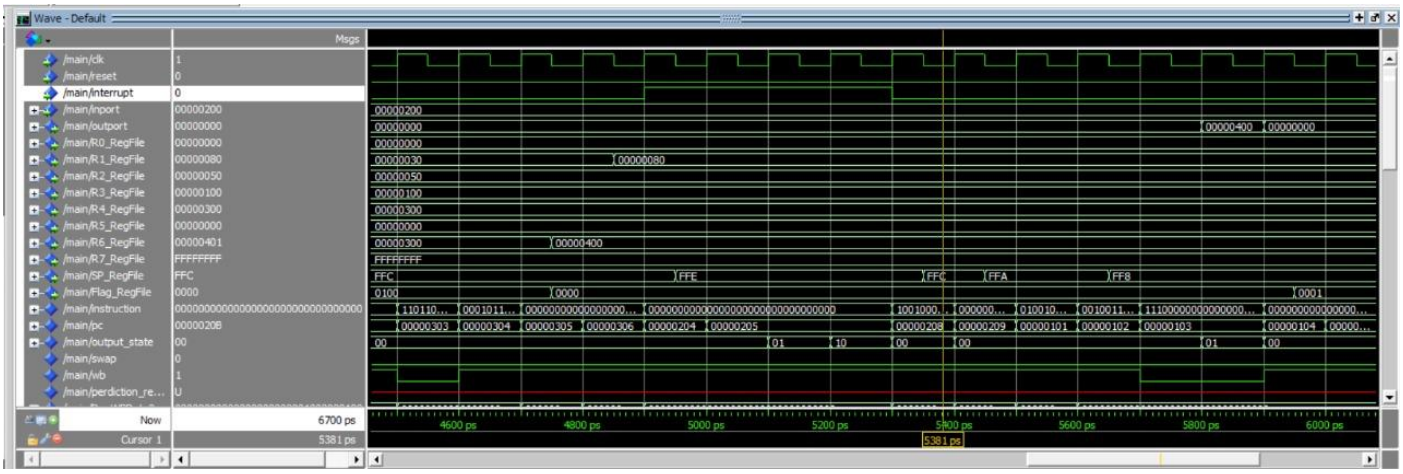

Msgs
/main/clk
/main/reset
/main/interrupt
/main/inport
/main/outport
/main/R0_RegFile
/main/R1_RegFile
/main/R2_RegFile
/main/R3_RegFile
/main/R4_RegFile
/main/R5_RegFile
/main/R6_RegFile
/main/R7_RegFile
/main/SP_RegFile
/main/Flag_RegFile
/main/instruction
/main/pc

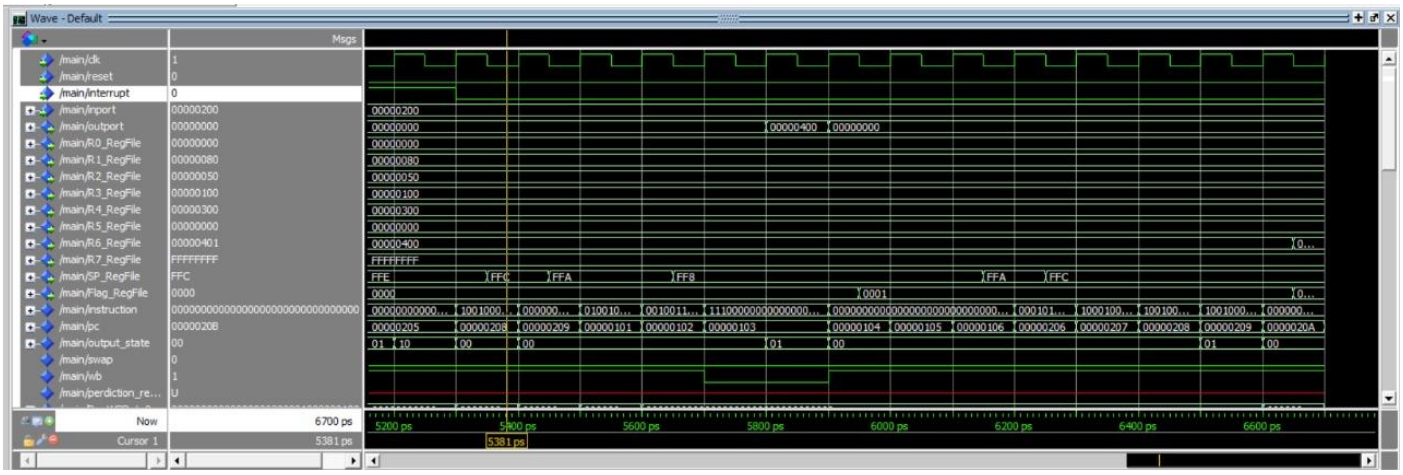
Branch file:


Wave - Default
Msgs
/main/clk
/main/reset
/main/interrupt
/main/inport
/main/outport
/main/R0_RegFile
/main/R1_RegFile
/main/R2_RegFile
/main/R3_RegFile
/main/R4_RegFile
/main/R5_RegFile
/main/R6_RegFile
/main/R7_RegFile
/main/SP_RegFile
/main/Flag_RegFile
/main/instruction
/main/pc
/main/output_state
/main/swap
/main/wb
/main/perdiction_re...
Now
6700 ps
Cursor 1
5381 ps


Wave - Default
Msgs
/main/clk
/main/reset
/main/interrupt
/main/inport
/main/outport
/main/R0_RegFile
/main/R1_RegFile
/main/R2_RegFile
/main/R3_RegFile
/main/R4_RegFile
/main/R5_RegFile
/main/R6_RegFile
/main/R7_RegFile
/main/SP_RegFile
/main/Flag_RegFile
/main/instruction
/main/pc
/main/output_state
/main/swap
/main/wb
/main/perdiction_re...
Now
6700 ps
Cursor 1
5381 ps


Wave - Default
Msgs
/main/clk
/main/reset
/main/interrupt
/main/inport
/main/outport
/main/R0_RegFile
/main/R1_RegFile
/main/R2_RegFile
/main/R3_RegFile
/main/R4_RegFile
/main/R5_RegFile
/main/R6_RegFile
/main/R7_RegFile
/main/SP_RegFile
/main/Flag_RegFile
/main/Instruction
/main/pc
/main/output_state
/main/swap
/main/wb
/main/perdiction_re...
Now
6700 ps
Cursor 1
5381 ps
3000 ps
3200 ps
3400 ps
3600 ps
3800 ps
4000 ps
4200 ps
4400 ps


Wave - Default
Msgs
/main/clk
/main/reset
/main/interrupt
/main/inport
/main/outport
/main/R0_RegFile
/main/R1_RegFile
/main/R2_RegFile
/main/R3_RegFile
/main/R4_RegFile
/main/R5_RegFile
/main/R6_RegFile
/main/R7_RegFile
/main/SP_RegFile
/main/Flag_RegFile
/main/Instruction
/main/pc
/main/output_state
/main/swap
/main/wb
/main/perdiction_re...
Now
6700 ps
Cursor 1
5381 ps
4600 ps
4800 ps
5000 ps
5200 ps
5381 ps
5600 ps
5800 ps
6000 ps


Wave - Default
Msgs
/main/clk
/main/reset
/main/interrupt
/main/inport
/main/outport
/main/R0_RegFile
/main/R1_RegFile
/main/R2_RegFile
/main/R3_RegFile
/main/R4_RegFile
/main/R5_RegFile
/main/R6_RegFile
/main/R7_RegFile
/main/SP_RegFile
/main/Flag_RegFile
/main/Instruction
/main/pc
/main/output_state
/main/swap
/main/wb
/main/perdiction_re...
Now
6700 ps
Cursor 1
5381 ps
5200 ps
5381 ps
5600 ps
5800 ps
6000 ps
6200 ps
6400 ps
6600 ps