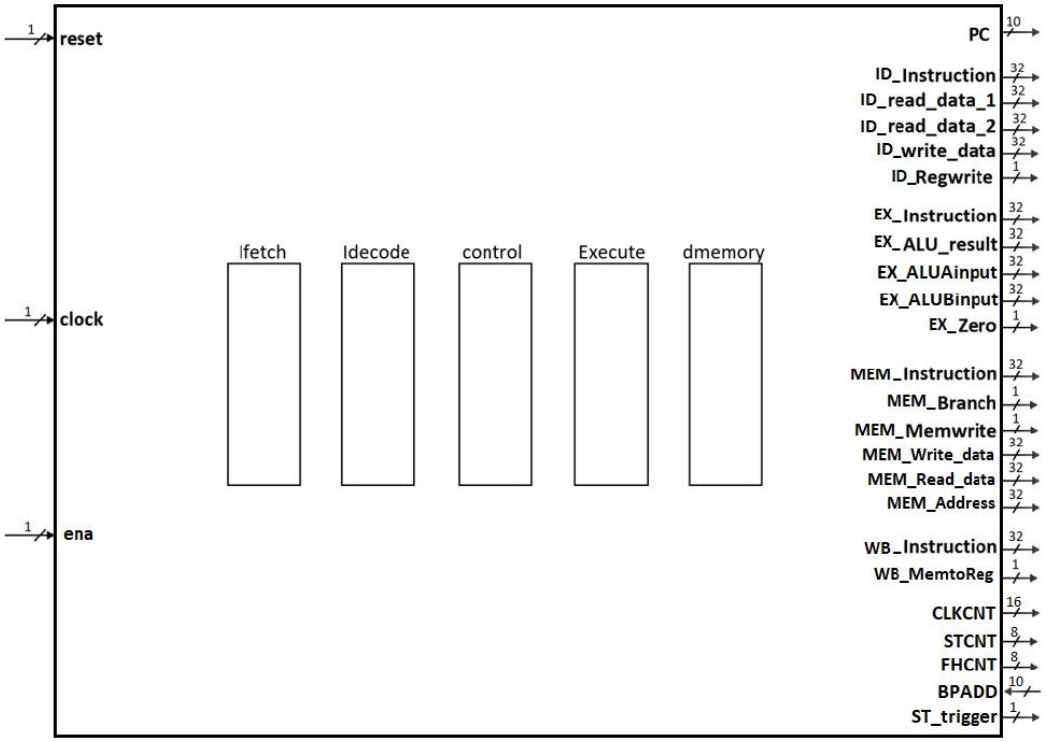
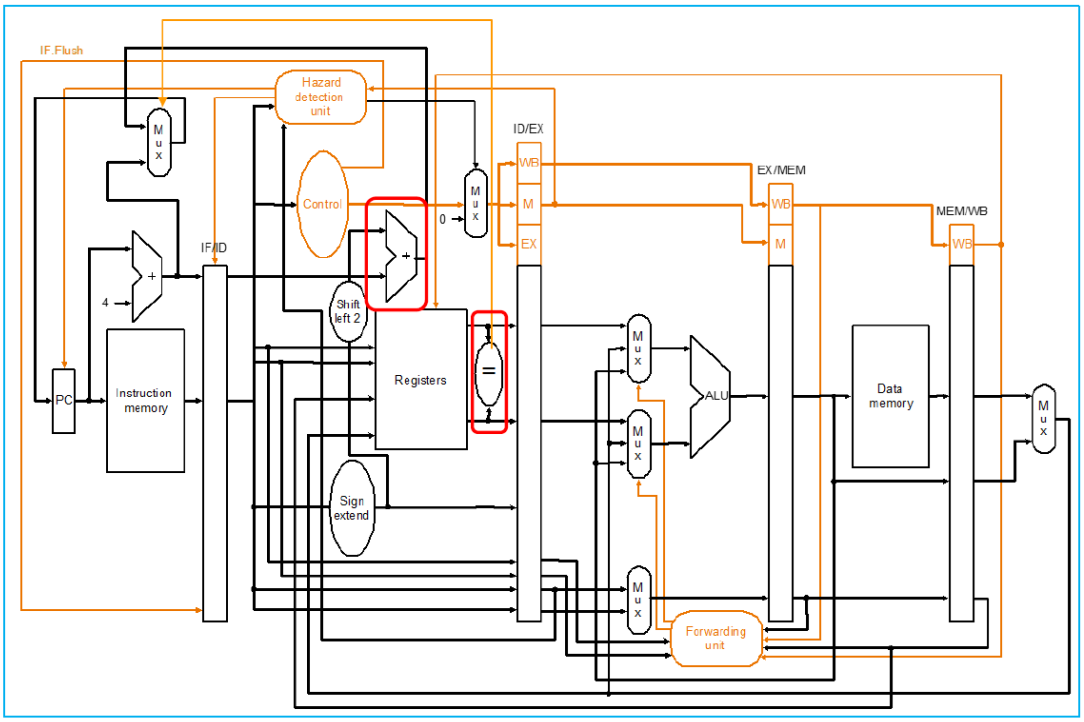
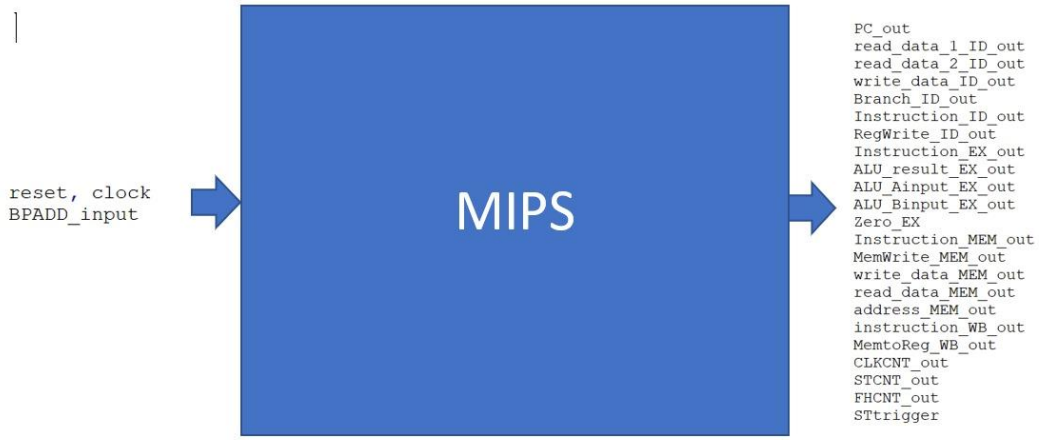
## ניסוי 5:

בניסוי זה מימשנו מעבד piplined MIPS עם יחידת stall ו- forward, וכן יחידת hazard המבצעת flush.

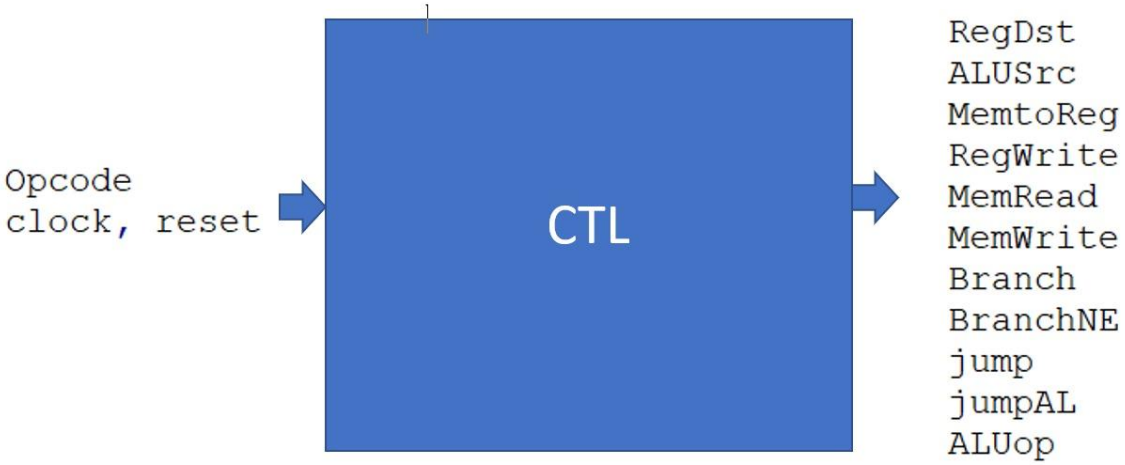


לפנינו פירוט של כל בלוק:

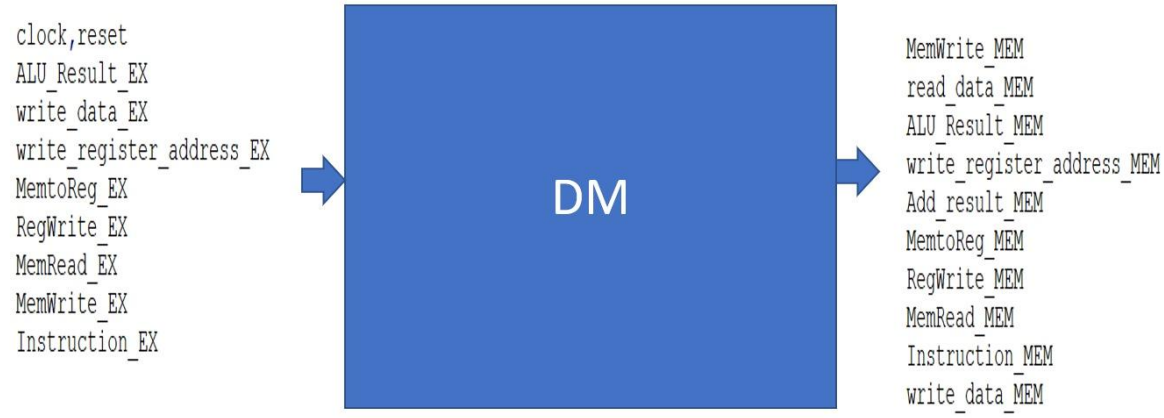







Instruction_IF
pc_plus_4_IF
result_WB
clock,reset
write_register_address_WB
jump_ID
JumpAL_ID
RegWrite_WB
stall
Branch_ID
BranchNE_ID
ALUOp_ID
ID
read_data_1_ID
read_data_2_ID
Instruction_ID
sft16_ID
PCSrc_ID
pc_plus_4_ID
write_register_address_1_ID
write_register_address_0_ID
read_register_1_address_ID
Add_Result_ID
Sign_extend_ID
inst_j_ID
Jr_ID
write_data_ID
flush_ID

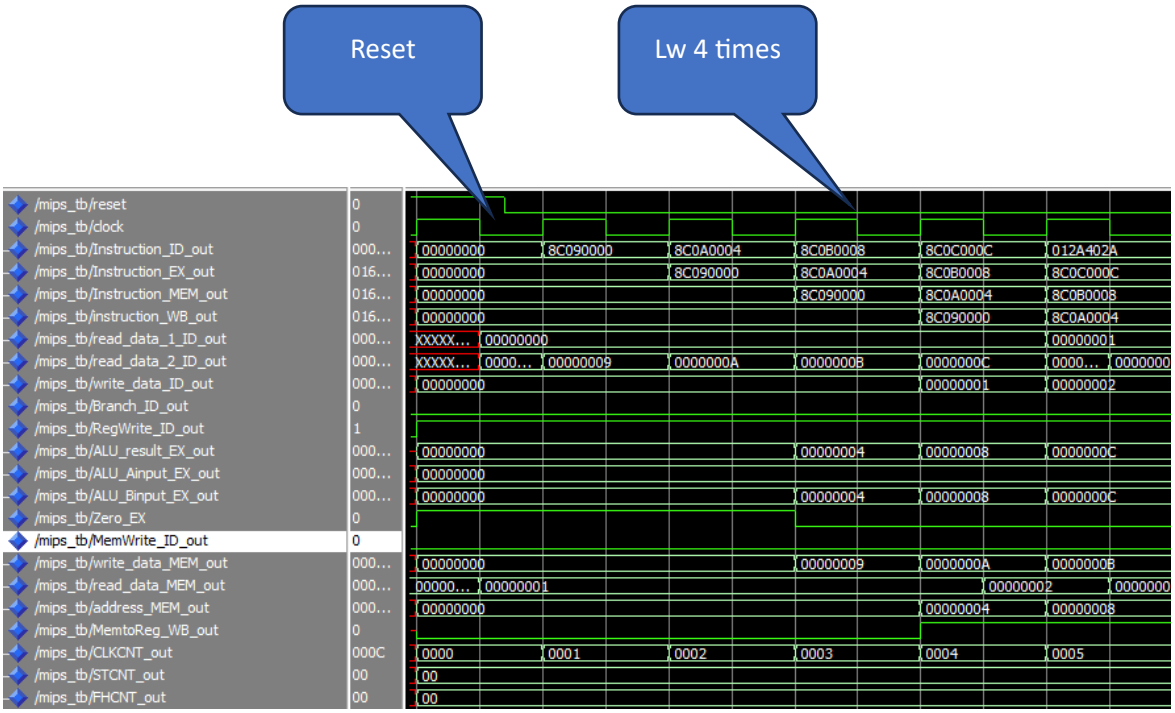

Read_data_1_ID
Read_data_2_ID
write_register_address_1_ID
write_register_address_0_ID
Sign_extend_ID
PC_plus_4_ID
sft16_ID
clock, reset
RegDst_ID
ALUSrc_ID
MemtoReg_ID
RegWrite_ID
MemRead_ID
MemWrite_ID
ALUop_ID
stall
read_register_1_address_ID
forward_A
forward_B
result_WB
ALU_Result_MEM
Instruction_ID
EX
ALU_Result_EX
write_data_EX
write_register_address_EX
write_register_address_0_EX_hazard
MemtoReg_EX
RegWrite_EX
MemRead_EX
MemWrite_EX
Branch_EX
BranchNE_EX
Zero_EX
read_register_1_address_EX
Instruction_EX
Ainput_EX
Binput_EX

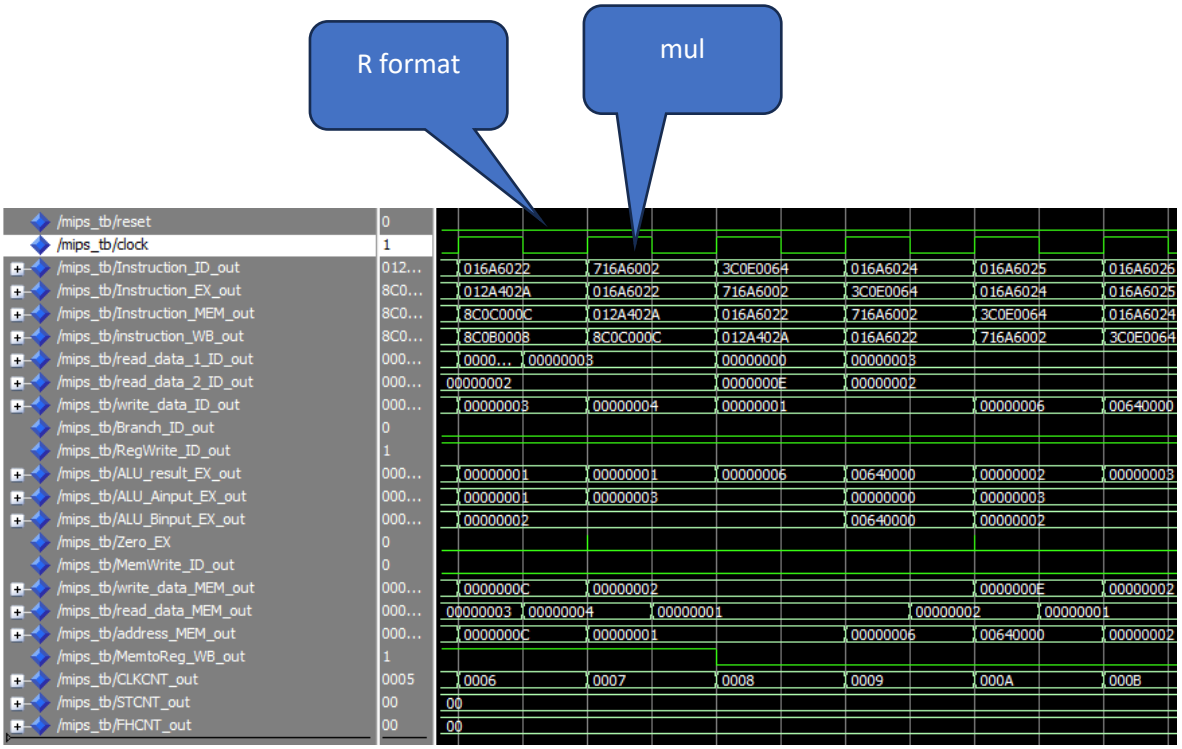

clock,reset
ALU_Result_EX
write_data_EX
write_register_address_EX
MemtoReg_EX
RegWrite_EX
MemRead_EX
MemWrite_EX
Instruction_EX
DM
MemWrite_MEM
read_data_MEM
ALU_Result_MEM
write_register_address_MEM
Add_result_MEM
MemtoReg_MEM
RegWrite_MEM
MemRead_MEM
Instruction_MEM
write_data_MEM

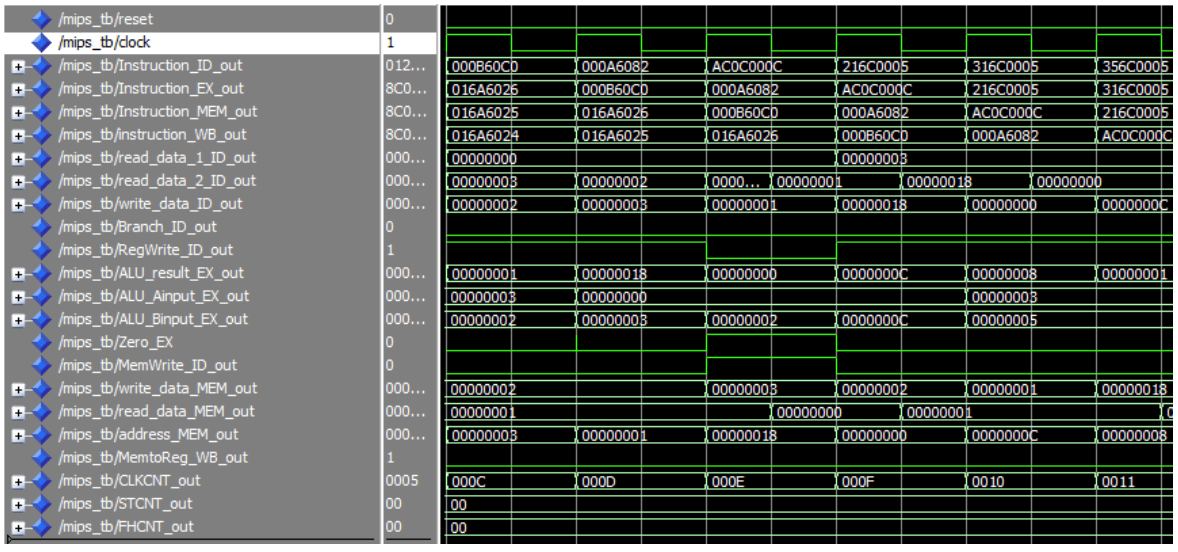

read_data_MEM
ALU_Result_MEM
write_register_address_MEM
clock,reset
MemtoReg_MEM
RegWrite_MEM
Instruction_MEM
WB
result_WB
write_register_address_WB
RegWrite_WB
Instruction_WB
MemtoReg_WB

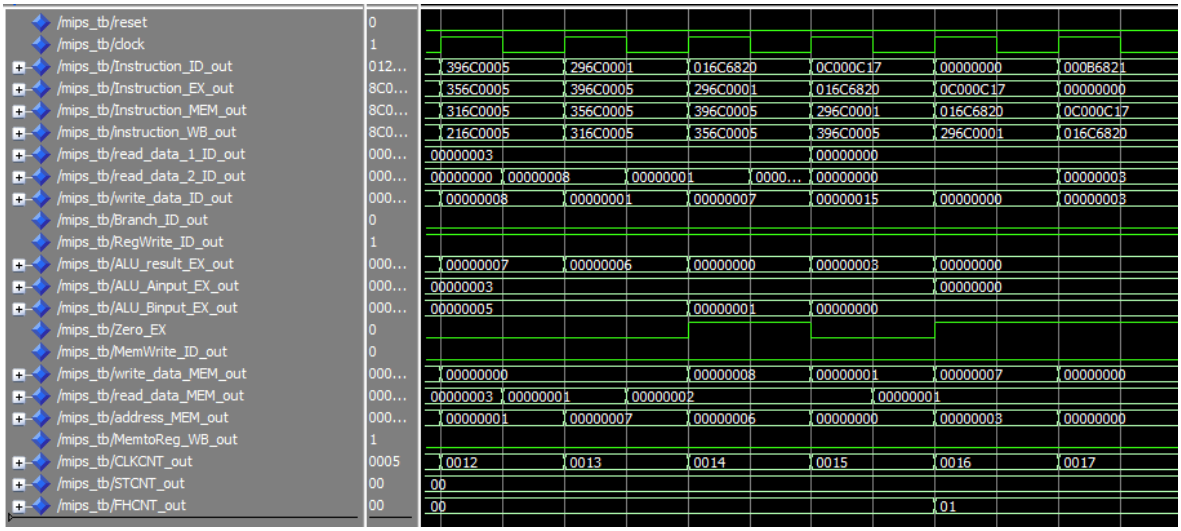
לפנינו דוגמת קוד בדיקה שהרצנו, ואחריה תוצאות הtestbench בmodelsim:

```
.data
        i: .word 1
        j: .word 2
        g: .word 3
        h: .word 4
        f: .word 5
.text
        lw $t1,0
        lw $t2,4
        lw $t3,8
        lw $t4,12
        slt $t0,$t1,$t2
        sub $t4, $t3, $t2
        mul $t4, $t3, $t2
        lui $t6, 100
        and $t4,$t3,$t2
        or $t4,$t3,$t2
        xor $t4,$t3,$t2
        sll $t4,$t3,3
        srl $t4,$t2,2
        sw $t4,12
        addi $t4,$t3,5
        andi $t4,$t3,5
        ori $t4,$t3,5
        xori $t4,$t3,5
        slti $t4,$t3,1
TWO:    add $t5,$t3,$t4
ONE:    jal IF
        j THREE
        add $t5,$t3,$t4
IF:     move $t5,$t3
        jr $ra
THREE:  bne $t0,$t1,END
ELSE:   sub $t5,$t3,$t4
END:    sw $t5,f
```




R format
mul
/mips_tb/reset
/mips_tb/clock
/mips_tb/Instruction_ID_out
/mips_tb/Instruction_EX_out
/mips_tb/Instruction_MEM_out
/mips_tb/instruction_WB_out
/mips_tb/read_data_1_ID_out
/mips_tb/read_data_2_ID_out
/mips_tb/write_data_ID_out
/mips_tb/Branch_ID_out
/mips_tb/RegWrite_ID_out
/mips_tb/ALU_result_EX_out
/mips_tb/ALU_Ainput_EX_out
/mips_tb/ALU_Binput_EX_out
/mips_tb/Zero_EX
/mips_tb/MemWrite_ID_out
/mips_tb/write_data_MEM_out
/mips_tb/read_data_MEM_out
/mips_tb/address_MEM_out
/mips_tb/MemtoReg_WB_out
/mips_tb/CLKCNT_out
/mips_tb/STCNT_out
/mips_tb/FHCNT_out

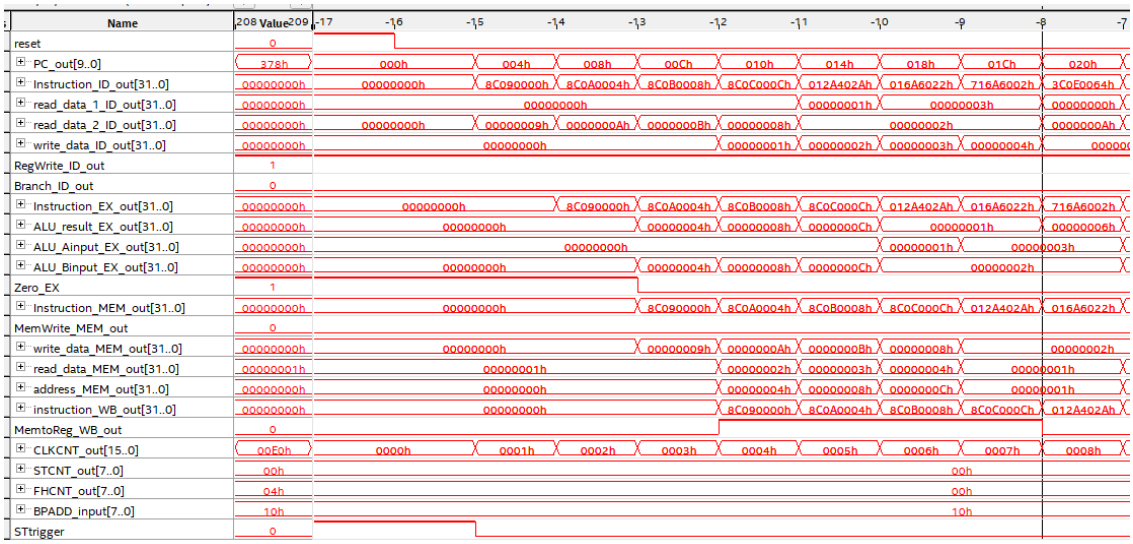

/mips_tb/reset
/mips_tb/clock
/mips_tb/Instruction_ID_out
/mips_tb/Instruction_EX_out
/mips_tb/Instruction_MEM_out
/mips_tb/instruction_WB_out
/mips_tb/read_data_1_ID_out
/mips_tb/read_data_2_ID_out
/mips_tb/write_data_ID_out
/mips_tb/Branch_ID_out
/mips_tb/RegWrite_ID_out
/mips_tb/ALU_result_EX_out
/mips_tb/ALU_Ainput_EX_out
/mips_tb/ALU_Binput_EX_out
/mips_tb/Zero_EX
/mips_tb/MemWrite_ID_out
/mips_tb/write_data_MEM_out
/mips_tb/read_data_MEM_out
/mips_tb/address_MEM_out
/mips_tb/MemtoReg_WB_out
/mips_tb/CLKCNT_out
/mips_tb/STCNT_out
/mips_tb/FHCNT_out

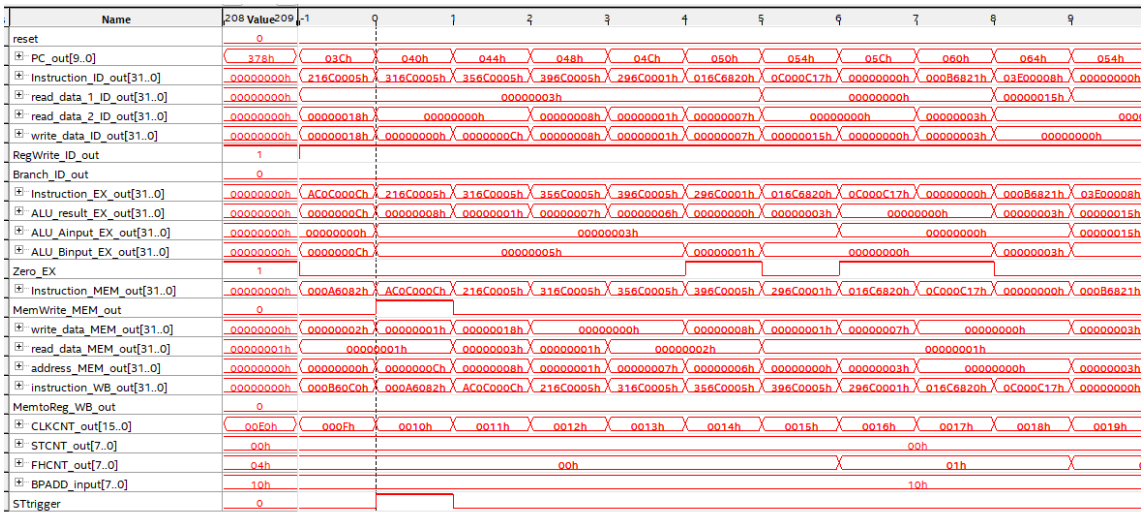



וכן הלאה...

כעת צרבנו את הקוד לבקר באמצעות הquartus.

לפנינו תמונה מהSignal-Tap לאותו קוד אסמבלי שהרצנו בmidelsim:






Fmax של המערכת הוא:

| | Fmax | Restricted Fmax | Clock Name | Note |
|---|---|---|---|---|
| 1 | 37.59 MHz | 37.59 MHz | clock | |
| 2 | 65.17 MHz | 65.17 MHz | altera_reserved_tck | |

וה-logic usage report הוא:

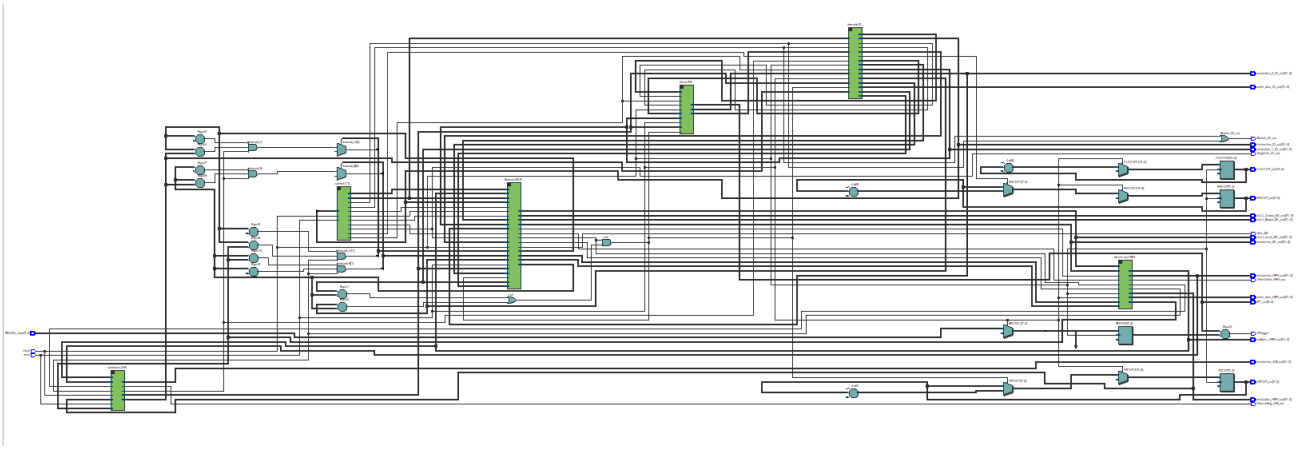
**Analysis & Synthesis Resource Utilization by Entity**

<<Filter>>

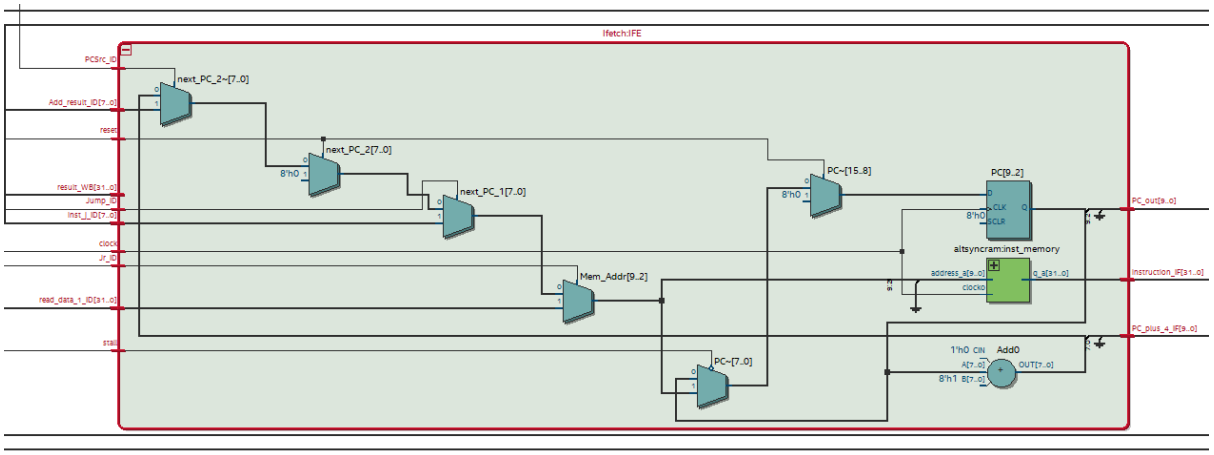
| | Compilation Hierarchy Node | Combinational ALUTs | Dedicated Logic Registers | Pins | Mem | Blc | Virtual Pins |
|---|---|---|---|---|---|---|---|
| 1 | \|MIPS | 2707 (58) | 8005 (40) | 10 | ... | 2 | 464 |
| 1 | \|Execute:EXE\| | 566 (451) | 105 (105) | 0 | 0 | 2 | 0 |
| 2 | \|Idecode:ID\| | 847 (847) | 1032 (1032) | 0 | 0 | 0 | 0 |
| 3 | \|Ifetch:IFE\| | 51 (51) | 8 (8) | 0 | ... | 0 | 0 |
| 4 | \|control:CTL\| | 15 (15) | 0 (0) | 0 | 0 | 0 | 0 |
| 5 | \|dmemory:MEM\| | 5 (5) | 104 (104) | 0 | ... | 0 | 0 |
| 6 | \|sld_hub:auto_hub\| | 91 (1) | 90 (0) | 0 | 0 | 0 | 0 |
| 7 | \|sld_signaltap:auto_signaltap_0\| | 1042 (2) | 6523 (946) | 0 | ... | 0 | 0 |
| 8 | \|writeback:WB\| | 32 (32) | 103 (103) | 0 | 0 | 0 | 0 |

נסתכל על RTL viewer של כל רכיב:

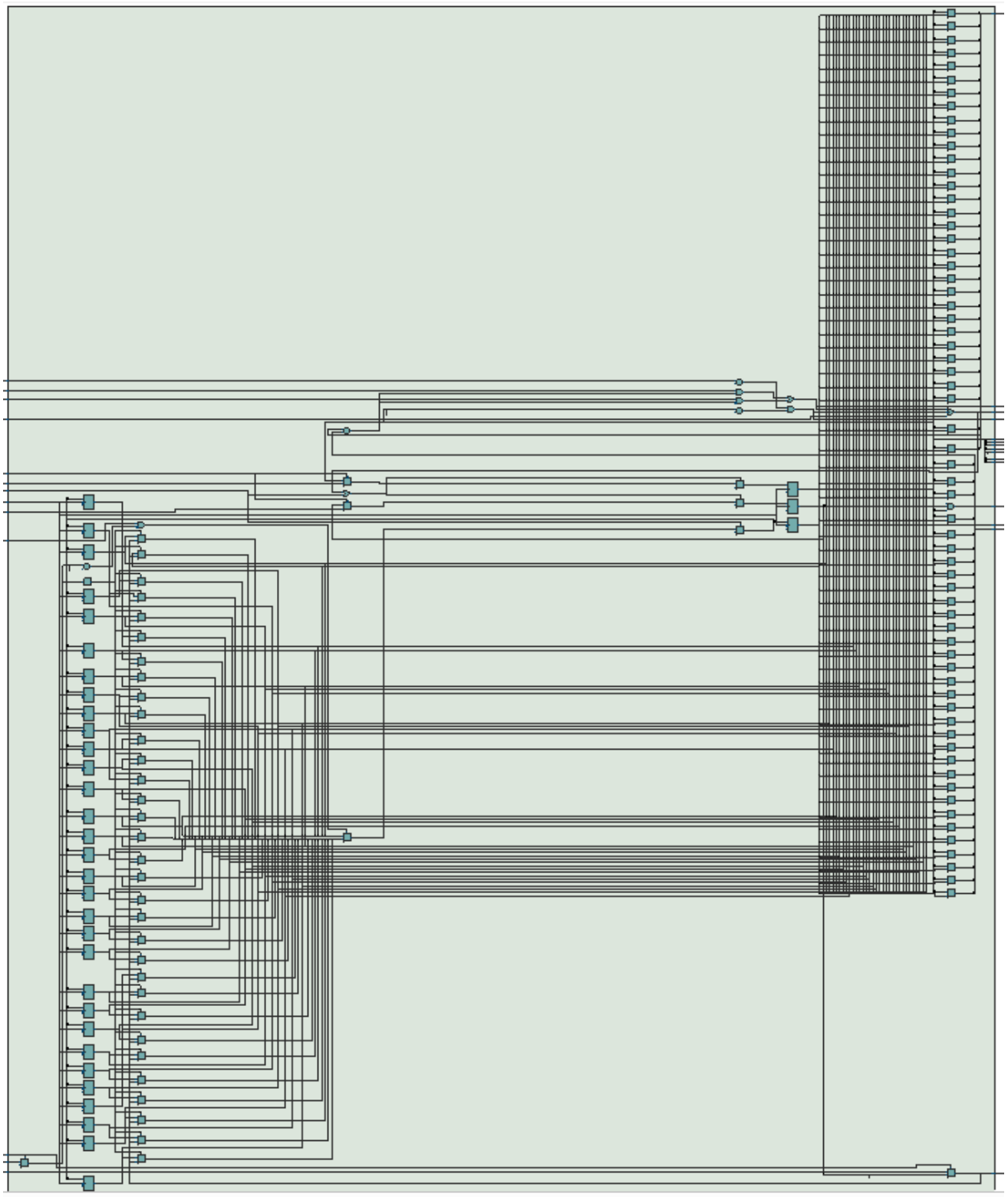
MIPS:



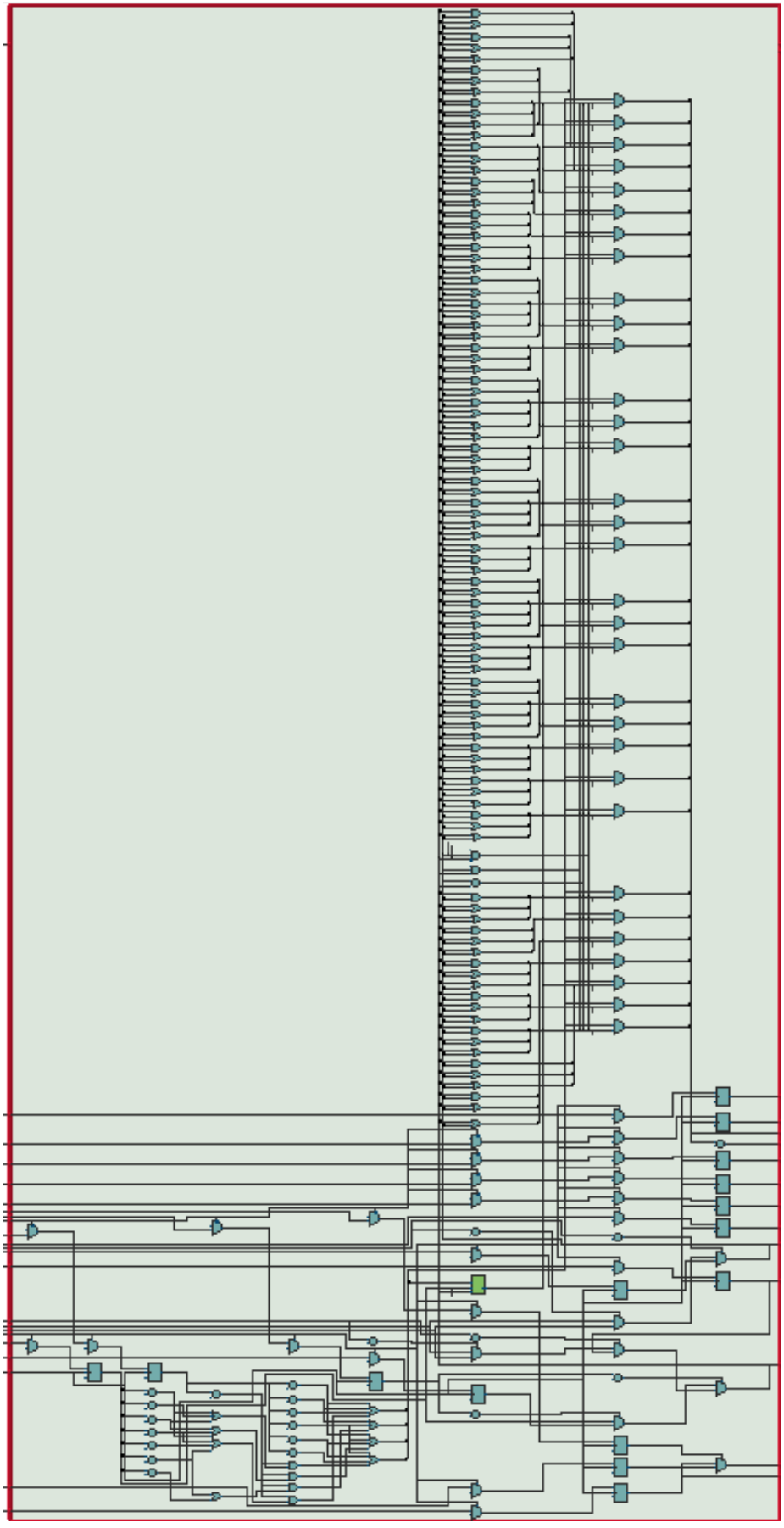
:IFETCH



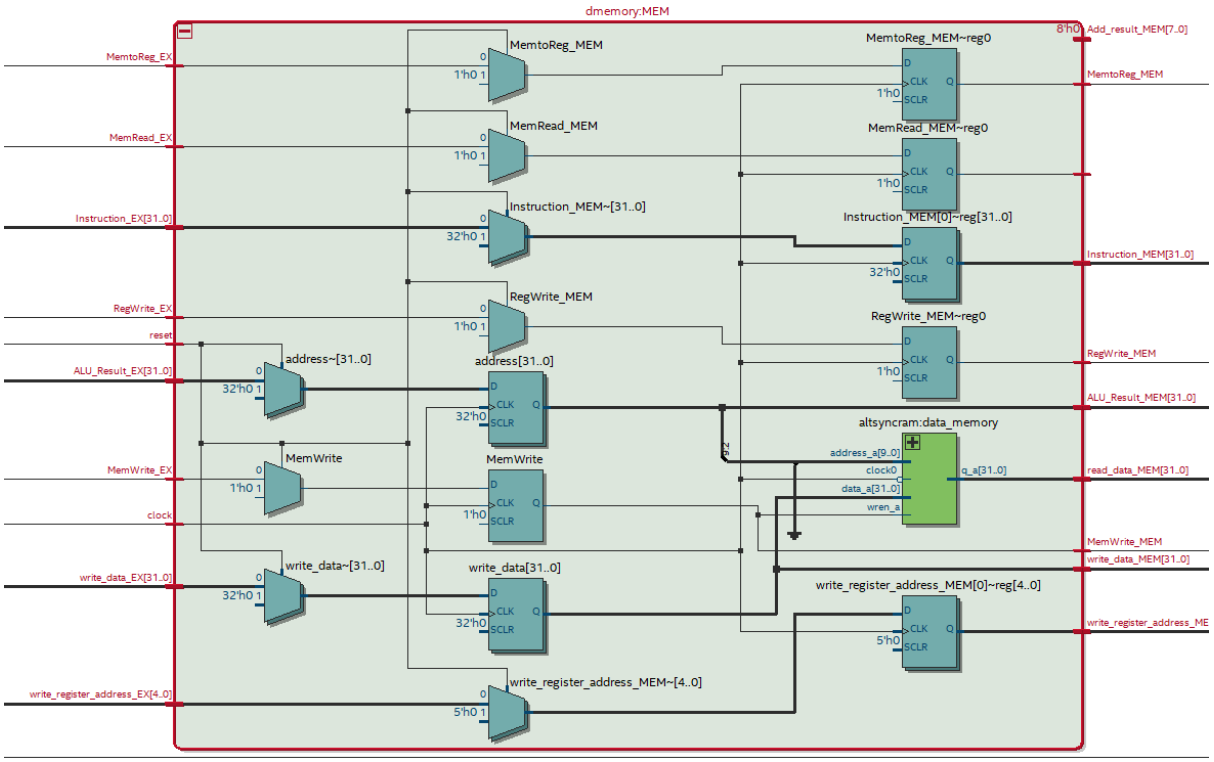

:IDECODE



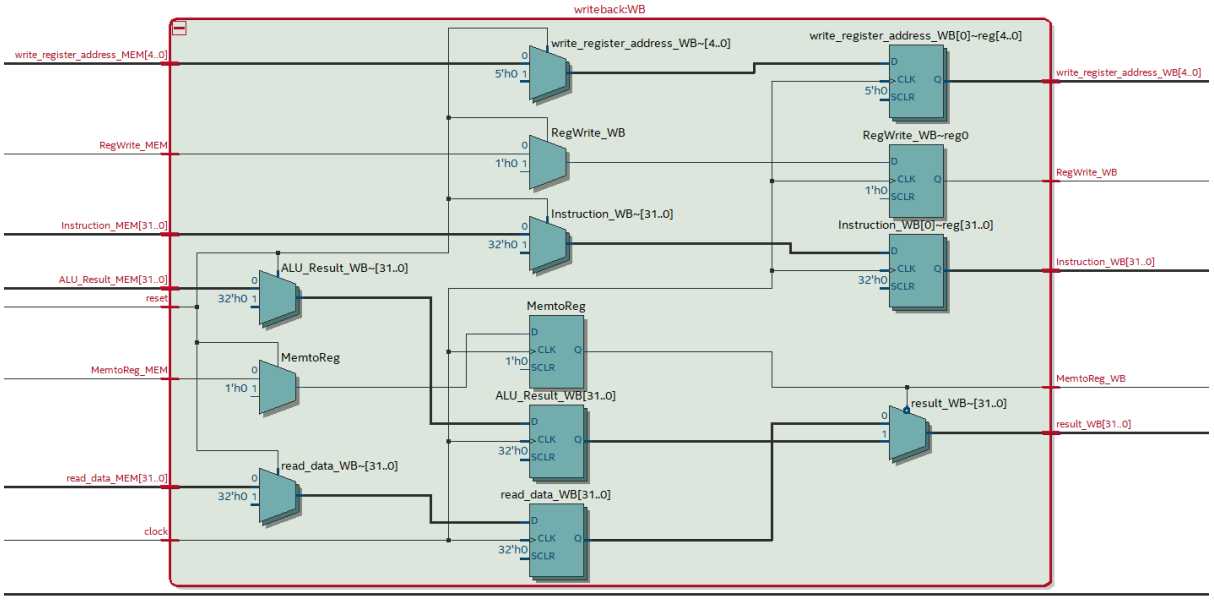
:EXECUTE



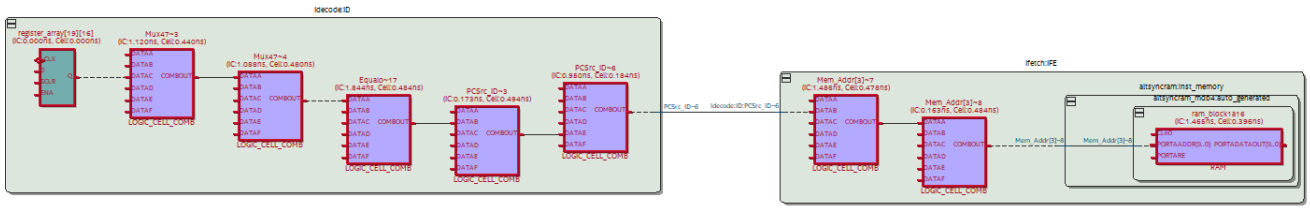
:DMEMORY


dmemory:MEM
MemtoReg_EX
MemRead_EX
Instruction_EX[31..0]
RegWrite_EX
reset
ALU_Result_EX[31..0]
MemWrite_EX
clock
write_data_EX[31..0]
write_register_address_EX[4..0]
MemtoReg_MEM
MemRead_MEM
Instruction_MEM~[31..0]
RegWrite_MEM
address~[31..0]
address[31..0]
MemWrite
write_data~[31..0]
write_data[31..0]
write_register_address_MEM~[4..0]
MemtoReg_MEM~reg0
MemRead_MEM~reg0
Instruction_MEM[0]~reg[31..0]
RegWrite_MEM~reg0
altsyncram:data_memory
write_register_address_MEM[0]~reg[4..0]
8'h0
Add_result_MEM[7..0]
MemtoReg_MEM
Instruction_MEM[31..0]
RegWrite_MEM
ALU_Result_MEM[31..0]
read_data_MEM[31..0]
MemWrite_MEM
write_data_MEM[31..0]
write_register_address_ME

:WRITEBACK

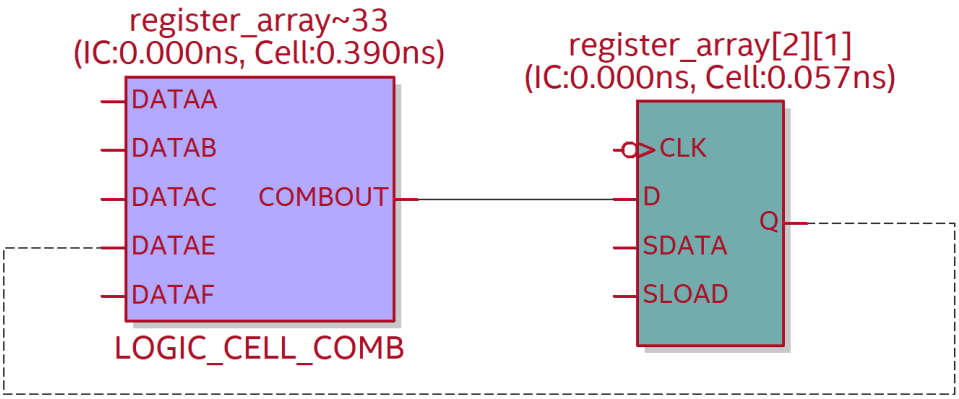

writeback:WB
write_register_address_MEM[4..0]
RegWrite_MEM
Instruction_MEM[31..0]
ALU_Result_MEM[31..0]
reset
MemtoReg_MEM
read_data_MEM[31..0]
clock
write_register_address_WB~[4..0]
RegWrite_WB
Instruction_WB~[31..0]
ALU_Result_WB~[31..0]
MemtoReg
ALU_Result_WB[31..0]
read_data_WB~[31..0]
read_data_WB[31..0]
write_register_address_WB[0]~reg[4..0]
RegWrite_WB~reg0
Instruction_WB[0]~reg[31..0]
result_WB~[31..0]
write_register_address_WB[4..0]
RegWrite_WB
Instruction_WB[31..0]
MemtoReg_WB
result_WB[31..0]

כעת נבחן את המסלול הקריטי:



כפי שניתן לראות, המסלול הקריטי הוא המסלול המבצע פעולה עם הזיכרון (lw או sw). הדבר הגיוני מכיוון שפעולה זו עוברת בכל שכבות המעבד ונכנסת לזיכרון הRAM ולכן היא הארוכה ביותר.

המסלול הקצר ביותר:

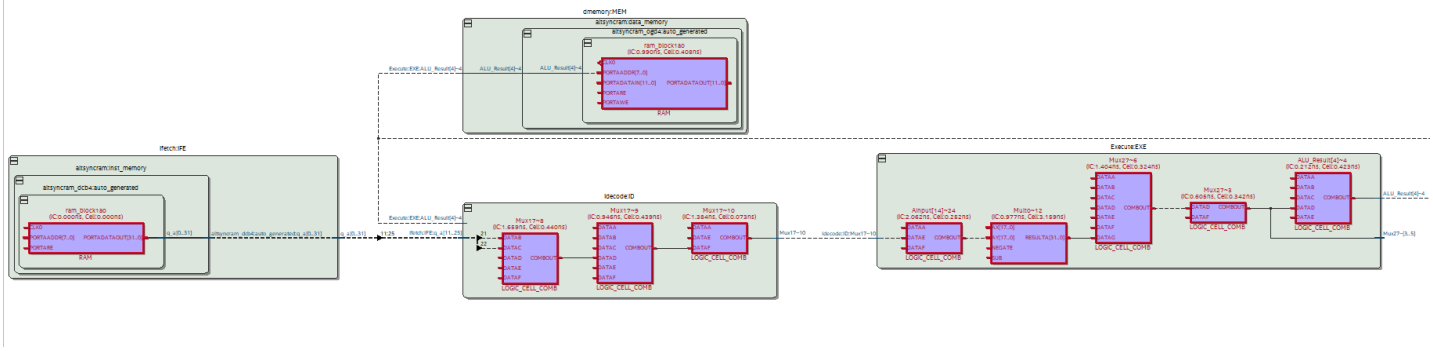



נבחן גם את המסלול הקריטי במעבד הsingle cycle:

ה-fmax הוא:

| | Fmax | Restricted Fmax | Clock Name | Note |
|---|---|---|---|---|
| 1 | 29.19 MHz | 29.19 MHz | clock | |
| 2 | 58.94 MHz | 58.94 MHz | altera...ed_tck | |

והמסלול הקריטי הוא:



גם כאן זהו המסלול הנכנס לזיכרון, וככל הנראה קשור לפקודת sw או לפקודת lw.