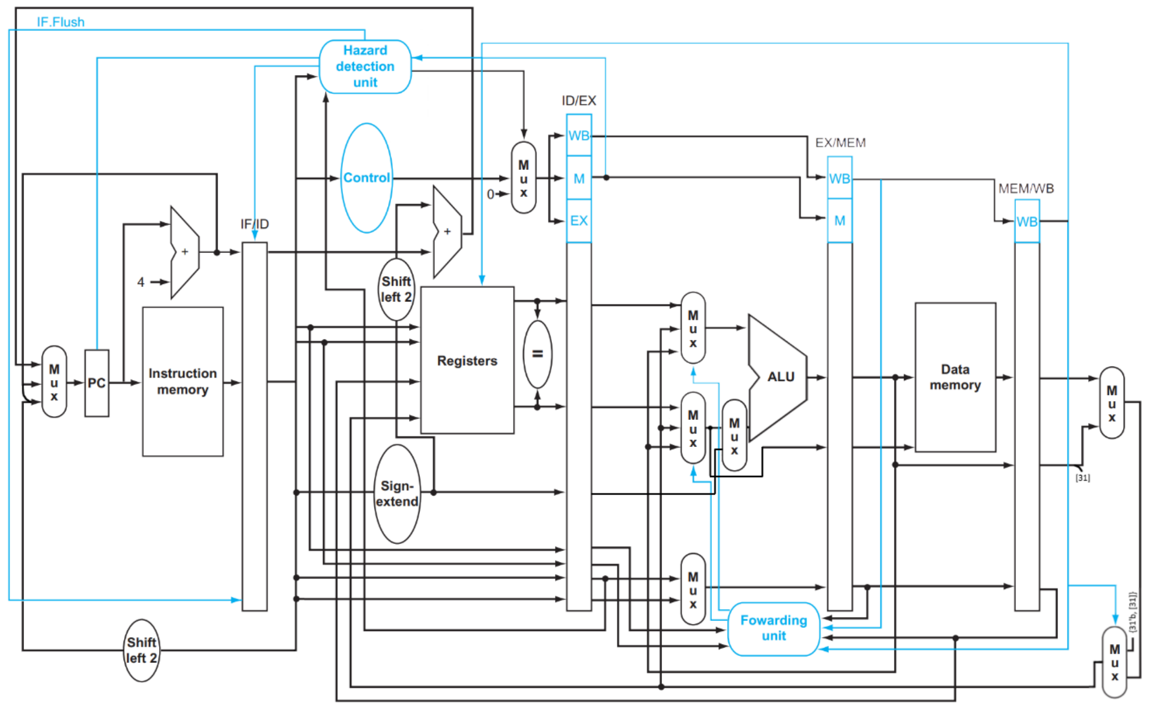
به نام خدا

تمرین کامپیوتری ۳ – طراحی Pipeline پردازنده میپس

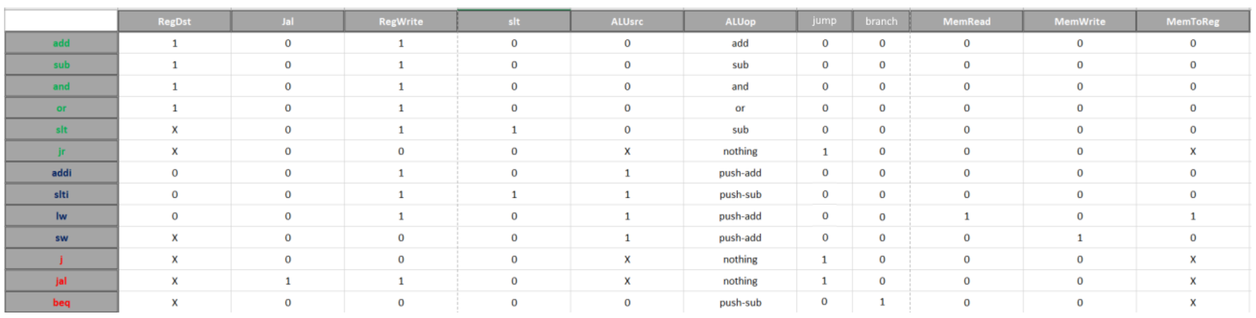
مهدی چراغی – ۸۱۰۱۹۹۳۹۹

پویا صادقی – ۸۱۰۱۹۹۴۴۷

دیتاپث:

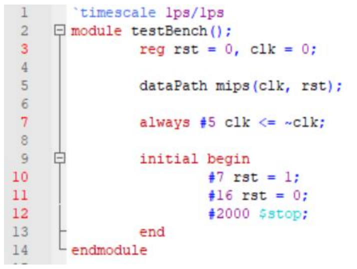


کنترلر:

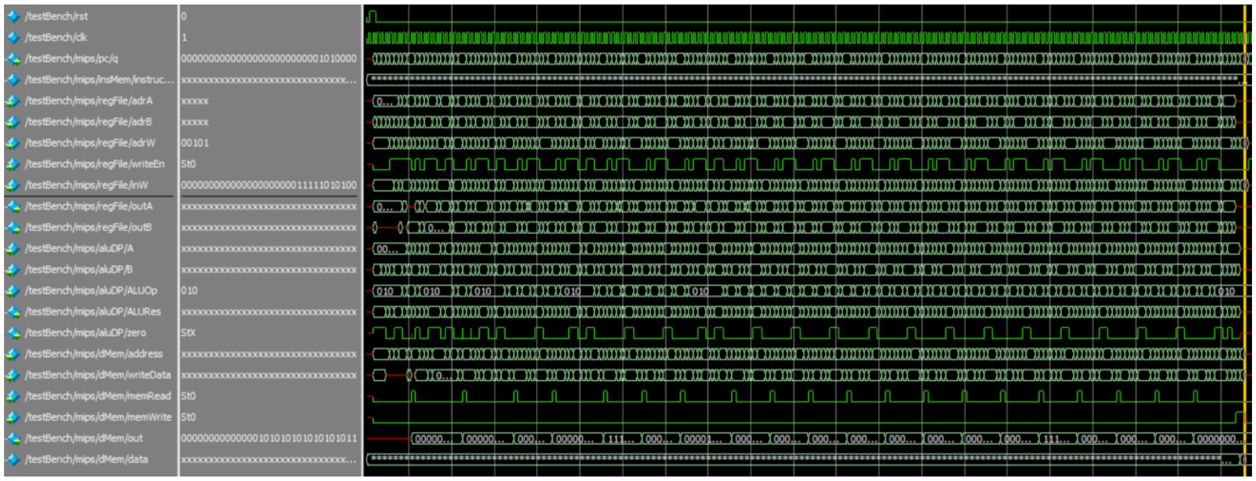


	RegDst	Jal	RegWrite	slt	ALUsrc	ALUop	jump	branch	MemRead	MemWrite	MemToReg
add	1	0	1	0	0	add	0	0	0	0	0
sub	1	0	1	0	0	sub	0	0	0	0	0
and	1	0	1	0	0	and	0	0	0	0	0
or	1	0	1	0	0	or	0	0	0	0	0
slt	X	0	1	1	0	sub	0	0	0	0	0
jr	X	0	0	0	X	nothing	1	0	0	0	X
addi	0	0	1	0	1	push-add	0	0	0	0	0
slti	0	0	1	1	1	push-sub	0	0	0	0	0
lw	0	0	1	0	1	push-add	0	0	1	0	1
sw	X	0	0	0	1	push-add	0	0	0	1	0
j	X	0	0	0	X	nothing	1	0	0	0	X
jal	X	1	1	0	X	nothing	1	0	0	0	X
beq	X	0	0	0	0	push-sub	0	1	0	0	X

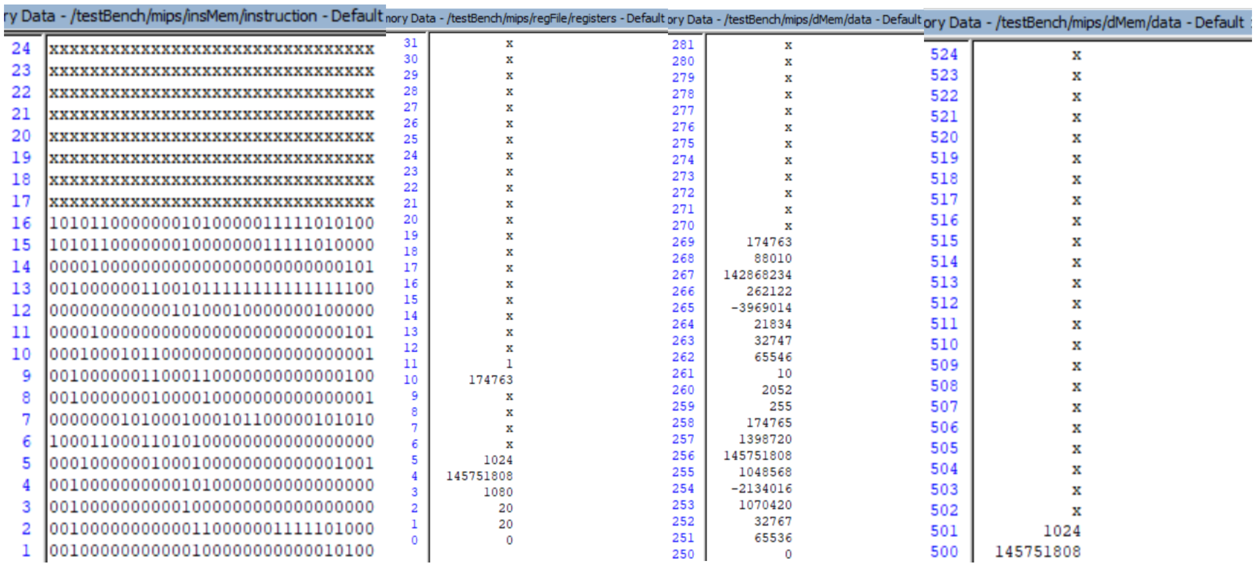
تست بنچ:



```verilog
`timescale 1ps/1ps
module testBench();
        reg rst = 0, clk = 0;

        dataPath mips(clk, rst);

        always #5 clk <= ~clk;

        initial begin
                #7 rst = 1;
                #16 rst = 0;
                #2000 $stop;
        end
endmodule
```



```
/testBench/rst                      0
/testBench/clk                      1
/testBench/mips/pc/q                00000000000000000000000001010000
/testBench/mips/insMem/instruc...   xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx...
/testBench/mips/regFile/adrA        xxxxx
/testBench/mips/regFile/adrB        xxxxx
/testBench/mips/regFile/adrW        00101
/testBench/mips/regFile/writeEn     St0
/testBench/mips/regFile/inW         00000000000000000000111110100100
/testBench/mips/regFile/outA        xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
/testBench/mips/regFile/outB        xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
/testBench/mips/aluDP/A             xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
/testBench/mips/aluDP/B             xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
/testBench/mips/aluDP/ALUOp         010
/testBench/mips/aluDP/ALURes        xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
/testBench/mips/aluDP/zero          StX
/testBench/mips/dMem/address        xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
/testBench/mips/dMem/writeData      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
/testBench/mips/dMem/memRead        St0
/testBench/mips/dMem/memWrite       St0
/testBench/mips/dMem/out            00000000000000101010101010101011
/testBench/mips/dMem/data           xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx...
```



ry Data - /testBench/mips/insMem/instruction - Default

```
24 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
23 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
22 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
21 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
20 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
19 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
18 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
17 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
16 10101100000001010000011111010100
15 10101100000001000000011111010000
14 00001000000000000000000000000101
13 00100000011001011111111111111100
12 00000000000010100010000000100000
11 00001000000000000000000000000101
10 00010001011000000000000000000001
 9 00100000011000110000000000000100
 8 00100000001000010000000000000001
 7 00000001010001000101100000101010
 6 10001100011010100000000000000000
 5 00010000001000100000000000001001
 4 00100000000001010000000000000000
 3 00100000000000100000000000000000
 2 00100000000000110000001111101000
 1 00100000000000100000000000010100
```

nory Data - /testBench/mips/regFile/registers - Default

```
31          x
30          x
29          x
28          x
27          x
26          x
25          x
24          x
23          x
22          x
21          x
20          x
19          x
18          x
17          x
16          x
15          x
14          x
13          x
12          x
11          1
10     174763
 9          x
 8          x
 7          x
 6          x
 5       1024
 4  145751808
 3       1080
 2         20
 1         20
 0          0
```

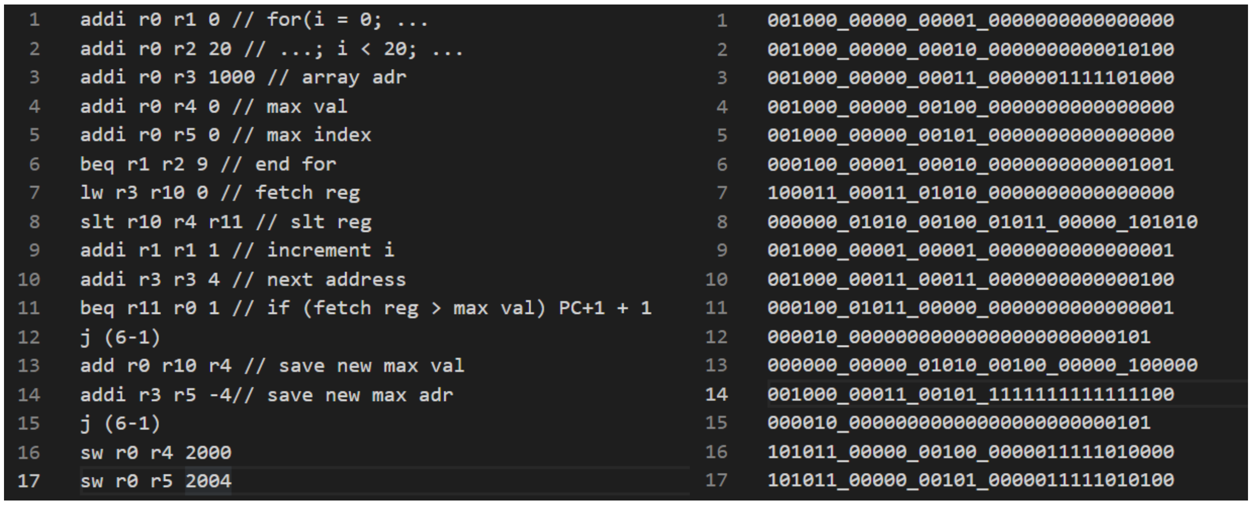
ory Data - /testBench/mips/dMem/data - Default

```
281          x
280          x
279          x
278          x
277          x
276          x
275          x
274          x
273          x
272          x
271          x
270          x
269     174763
268      88010
267  142868234
266     262122
265   -3969014
264      21834
263      32747
262      65546
261         10
260       2052
259        255
258     174765
257    1398720
256  145751808
255    1048568
254   -2134016
253    1070420
252      32767
251      65536
250          0
```

ory Data - /testBench/mips/dMem/data - Default

```
524          x
523          x
522          x
521          x
520          x
519          x
518          x
517          x
516          x
515          x
514          x
513          x
512          x
511          x
510          x
509          x
508          x
507          x
506          x
505          x
504          x
503          x
502          x
501       1024
500  145751808
```

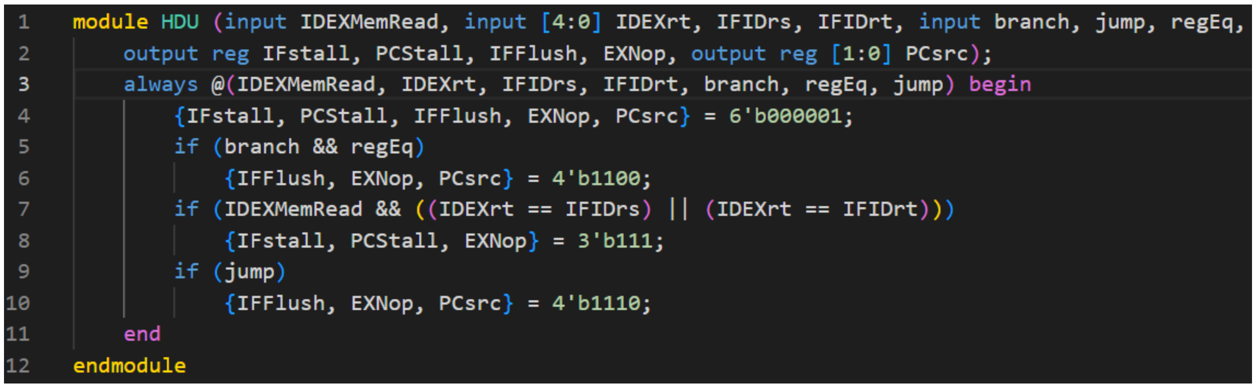
دستورات:



```
addi r0 r1 0 // for(i = 0; ...
addi r0 r2 20 // ...; i < 20; ...
addi r0 r3 1000 // array adr
addi r0 r4 0 // max val
addi r0 r5 0 // max index
beq r1 r2 9 // end for
lw r3 r10 0 // fetch reg
slt r10 r4 r11 // slt reg
addi r1 r1 1 // increment i
addi r3 r3 4 // next address
beq r11 r0 1 // if (fetch reg > max val) PC+1 + 1
j (6-1)
add r0 r10 r4 // save new max val
addi r3 r5 -4// save new max adr
j (6-1)
sw r0 r4 2000
sw r0 r5 2004
```

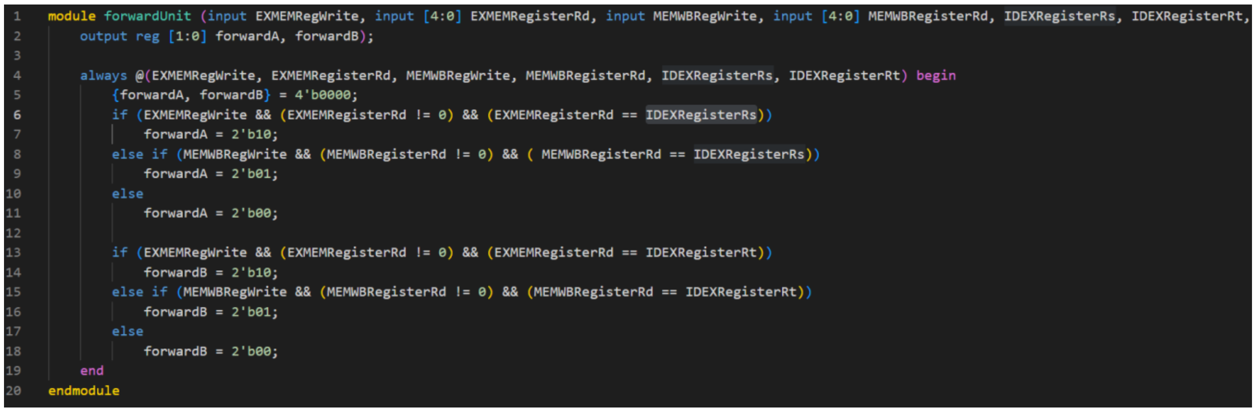
```
001000_00000_00001_0000000000000000
001000_00000_00010_0000000000010100
001000_00000_00011_0000001111101000
001000_00000_00100_0000000000000000
001000_00000_00101_0000000000000000
000100_00001_00010_0000000000001001
100011_00011_01010_0000000000000000
000000_01010_00100_01011_00000_101010
001000_00001_00001_0000000000000001
001000_00011_00011_0000000000000100
000100_01011_00000_0000000000000001
000010_00000000000000000000000101
000000_00000_01010_00100_00000_100000
001000_00011_00101_1111111111111100
000010_00000000000000000000000101
101011_00000_00100_0000011111010000
101011_00000_00101_0000011111010100
```

هازارد دیتکتور:



```
module HDU (input IDEXMemRead, input [4:0] IDEXrt, IFIDrs, IFIDrt, input branch, jump, regEq,
    output reg IFstall, PCStall, IFFlush, EXNop, output reg [1:0] PCsrc);
    always @(IDEXMemRead, IDEXrt, IFIDrs, IFIDrt, branch, regEq, jump) begin
        {IFstall, PCStall, IFFlush, EXNop, PCsrc} = 6'b000001;
        if (branch && regEq)
            {IFFlush, EXNop, PCsrc} = 4'b1100;
        if (IDEXMemRead && ((IDEXrt == IFIDrs) || (IDEXrt == IFIDrt)))
            {IFstall, PCStall, EXNop} = 3'b111;
        if (jump)
            {IFFlush, EXNop, PCsrc} = 4'b1110;
    end
endmodule
```

فوروارد یونیت:



```
module forwardUnit (input EXMEMRegWrite, input [4:0] EXMEMRegisterRd, input MEMWBRegWrite, input [4:0] MEMWBRegisterRd, IDEXRegisterRs, IDEXRegisterRt,
    output reg [1:0] forwardA, forwardB);

    always @(EXMEMRegWrite, EXMEMRegisterRd, MEMWBRegWrite, MEMWBRegisterRd, IDEXRegisterRs, IDEXRegisterRt) begin
        {forwardA, forwardB} = 4'b0000;
        if (EXMEMRegWrite && (EXMEMRegisterRd != 0) && (EXMEMRegisterRd == IDEXRegisterRs))
            forwardA = 2'b10;
        else if (MEMWBRegWrite && (MEMWBRegisterRd != 0) && ( MEMWBRegisterRd == IDEXRegisterRs))
            forwardA = 2'b01;
        else
            forwardA = 2'b00;

        if (EXMEMRegWrite && (EXMEMRegisterRd != 0) && (EXMEMRegisterRd == IDEXRegisterRt))
            forwardB = 2'b10;
        else if (MEMWBRegWrite && (MEMWBRegisterRd != 0) && (MEMWBRegisterRd == IDEXRegisterRt))
            forwardB = 2'b01;
        else
            forwardB = 2'b00;
    end
endmodule
```