SUB	000000	000111	1	1	1	0	0	1	0	0	1	0	0			
JR	000000	001000	x	x	x	x	x	x	x	x	0	0	0			
ORI	010000	-	0	0	0	1	0	0	0	0	1	0	0	0	0	0
ANDI	010001	-	0	0	1	1	0	0	0	0	1	0	0	0	0	0
XORI	010010	-	0	1	0	1	0	0	0	0	1	0	0	0	0	0
ADDI	010011	-	0	1	1	1	0	0	0	0	1	0	0	0	0	0
NORI	010100	-	1	0	0	1	0	0	0	0	1	0	0	0	0	0
NANDI	010101	-	1	0	1	1	0	0	0	0	1	0	0	0	0	0
SLTI	010110	-	1	1	0	1	0	0	0	0	1	0	0	0	0	0
SUBI	010111	-	1	1	1	1	0	0	0	0	1	0	0	0	0	0
LW	100011	-	0	1	1	1	0	0	0	1	1	1	0	0	0	0
SW	101011	-	0	1	1	1	x	x	x	x	0	0	1	0	0	0
BEQ	110000	-	x	x	x	x	x	x	x	x	0	0	0	1	0	0
J	110001	-	x	x	x	x	x	x	x	x	0	0	0	0	1	1
JAL	110011	-	x	x	x	x	1	0	1	0	1	0	0	0	1	1

Time left 0:05:20

What is the machine code (in hexadecimal) for the instruction

JAL 32 ?

○ AC070010

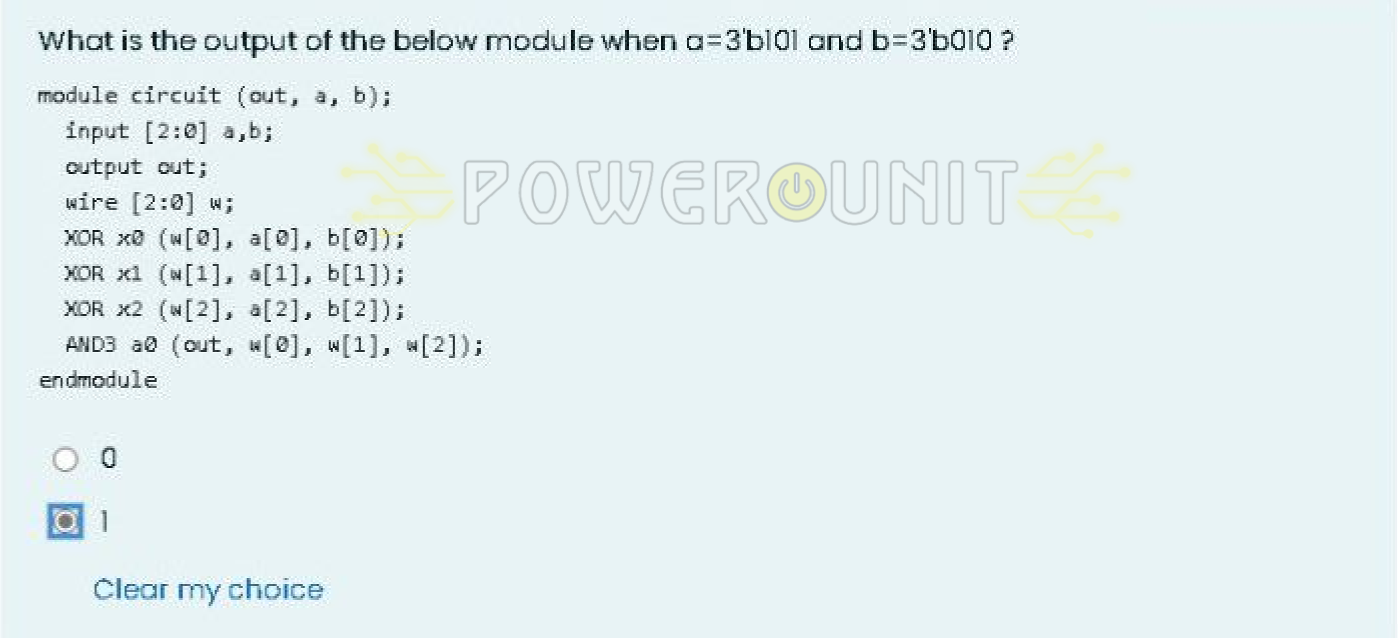
○ AC070008

○ C4000018

● CC000020

○ C4000028

○ CC000010



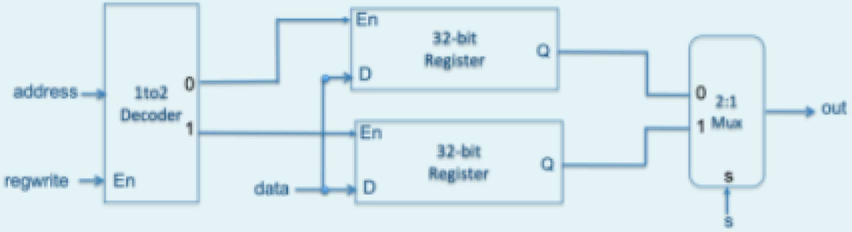
What is the output of the below module when a=3'b101 and b=3'b010 ?

```
module circuit (out, a, b);
  input [2:0] a,b;
  output out;
  wire [2:0] w;
  XOR x0 (w[0], a[0], b[0]);
  XOR x1 (w[1], a[1], b[1]);
  XOR x2 (w[2], a[2], b[2]);
  AND3 a0 (out, w[0], w[1], w[2]);
endmodule
```

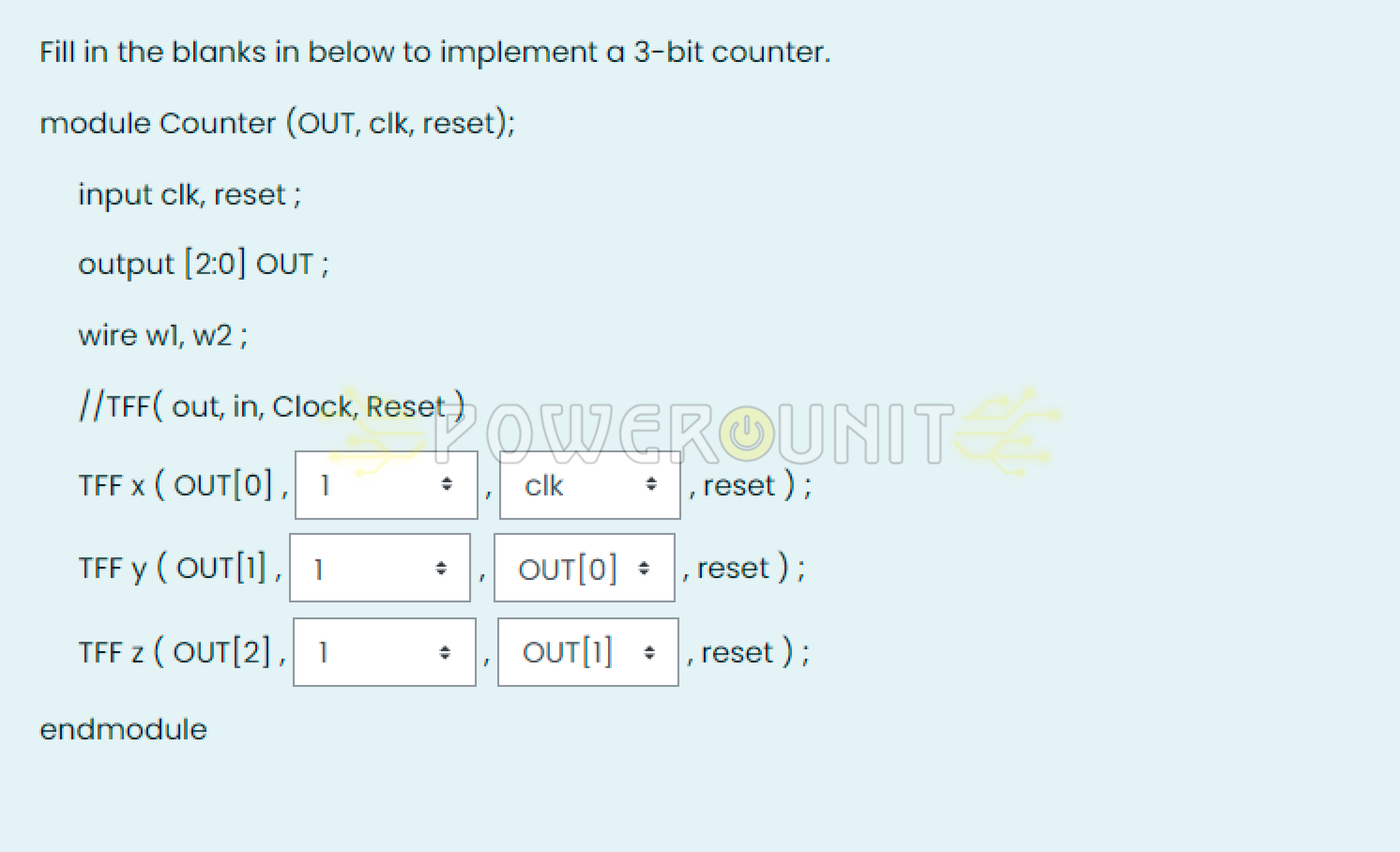
○ 0

● 1

Fill in the blanks in the below code.



```
module registerfile (out, address, regwrite, data, s, clk, reset, enable);

input address, regwrite, s, clk, reset, enable;

input [31:0] data;

output [31:0] out ;

wire  w0, w1;

wire [31:0] Q1, Q0;

// Dec1to2 ( out, in, En)

Dec1to2 ( { w0 , w1 } , address, regwrite);

// Register (Q, D, clk, reset, enable)

Register r0 ( Q0, data, clk, reset, w0 );

Register r1 ( Q1, data, clk, reset, w1 );

//Mux2to1 ( out, in, s)

Mux2to1 m0 (out, { Q0 , Q1 } , s);


endmodule
```

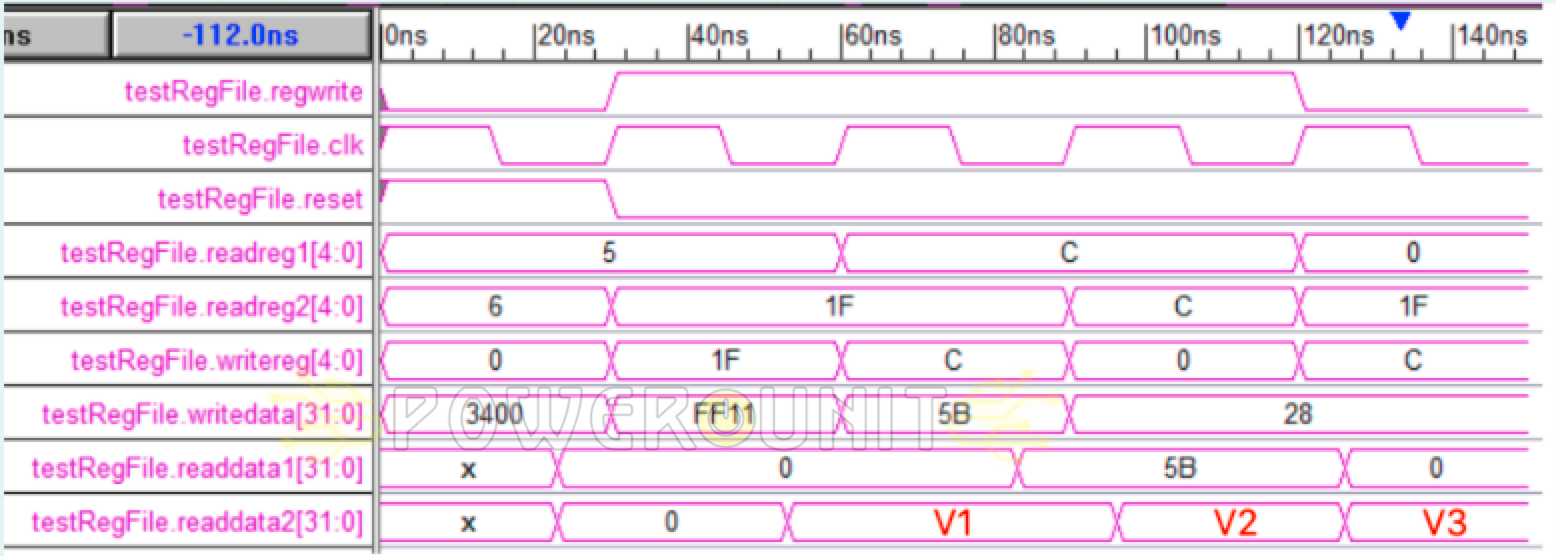


Fill in the blanks in below to implement a 3-bit counter.

```
module Counter (OUT, clk, reset);

   input clk, reset ;

   output [2:0] OUT ;

   wire w1, w2 ;

   //TFF( out, in, Clock, Reset )

   TFF x ( OUT[0] , 1 , clk , reset ) ;

   TFF y ( OUT[1] , 1 , OUT[0] , reset ) ;

   TFF z ( OUT[2] , 1 , OUT[1] , reset ) ;

endmodule
```

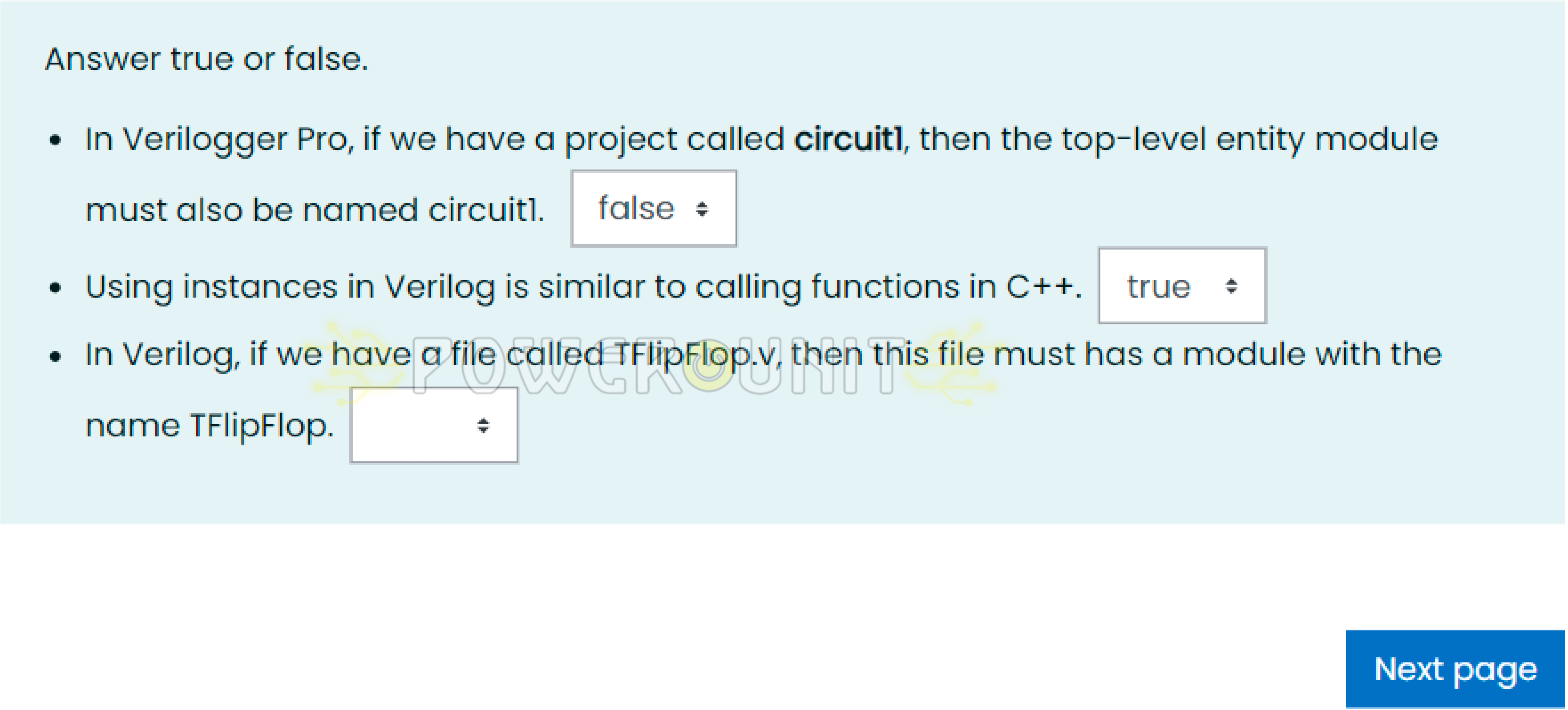
Time left 0:49:54

What are the values of V1, V2, and V3 in the below timing diagram of the register file implemented in experiment 3?



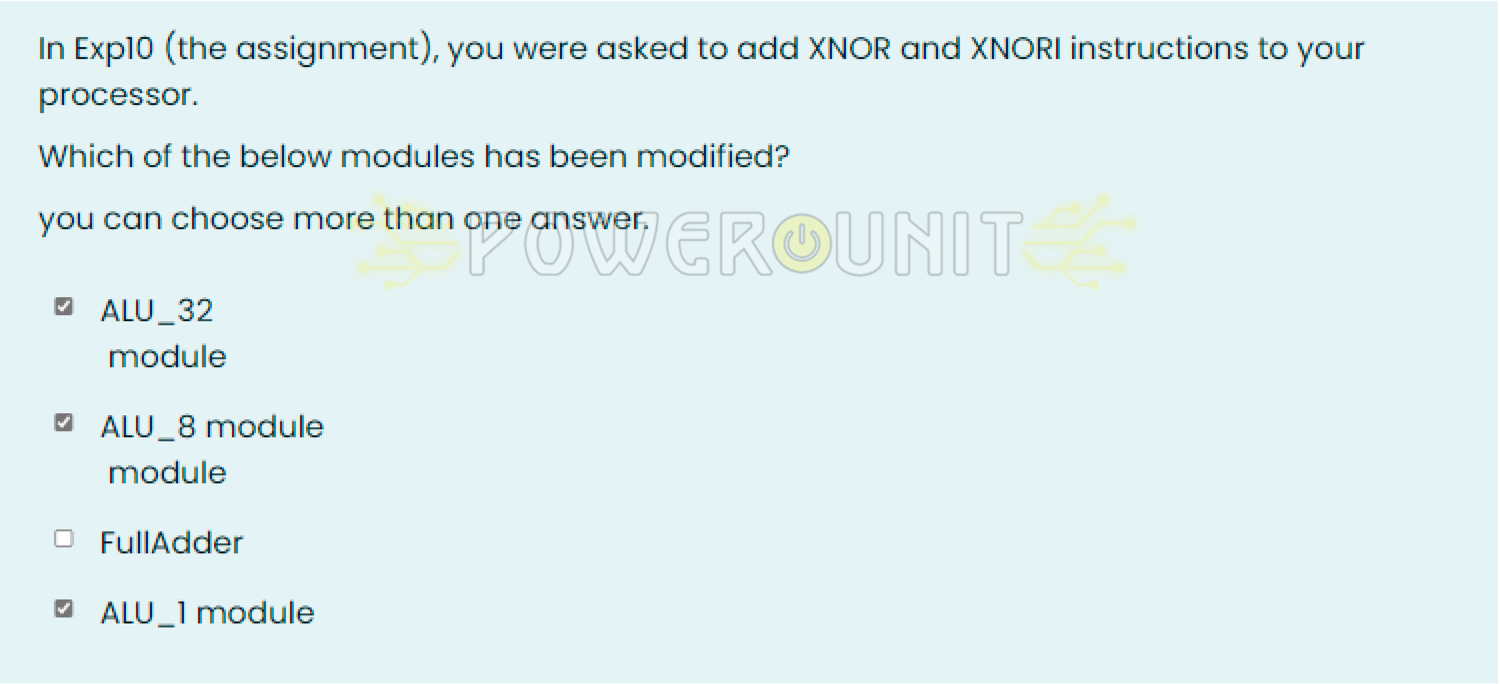
V1 =

V3 =



Answer true or false.

- In Verilogger Pro, if we have a project called **circuit1**, then the top-level entity module must also be named circuit1. [false ⬍]
- Using instances in Verilog is similar to calling functions in C++. [true ⬍]
- In Verilog, if we have a file called TFlipFlop.v, then this file must has a module with the name TFlipFlop. [⬍]

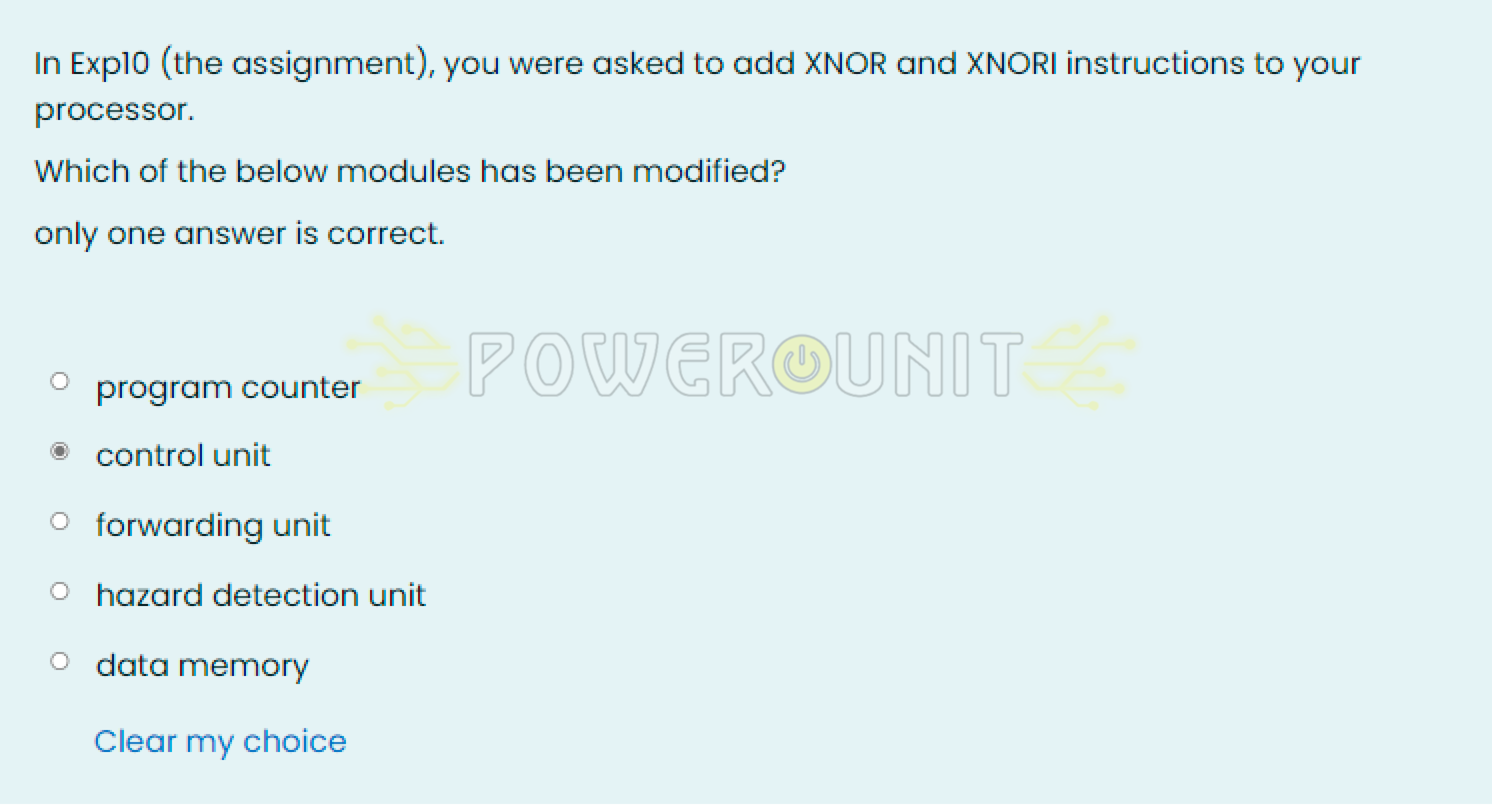


In Exp10 (the assignment), you were asked to add XNOR and XNORI instructions to your processor.

Which of the below modules has been modified?

you can choose more than one answer.

- ☑ ALU_32 module
- ☑ ALU_8 module module
- □ FullAdder
- ☑ ALU_1 module



In Exp10 (the assignment), you were asked to add XNOR and XNORI instructions to your processor.

Which of the below modules has been modified?

only one answer is correct.

○ program counter

● control unit

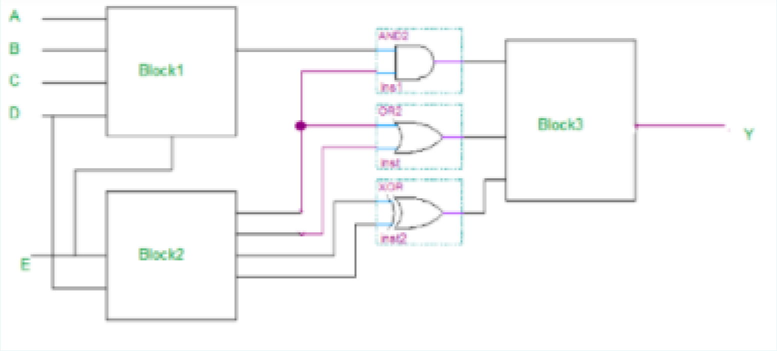
○ forwarding unit

○ hazard detection unit

○ data memory

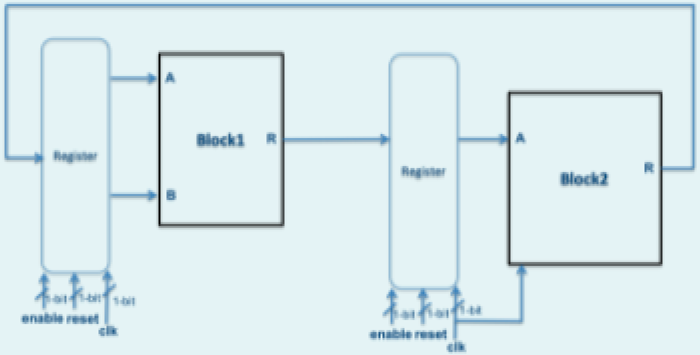
Clear my choice

Time left 0:30:35

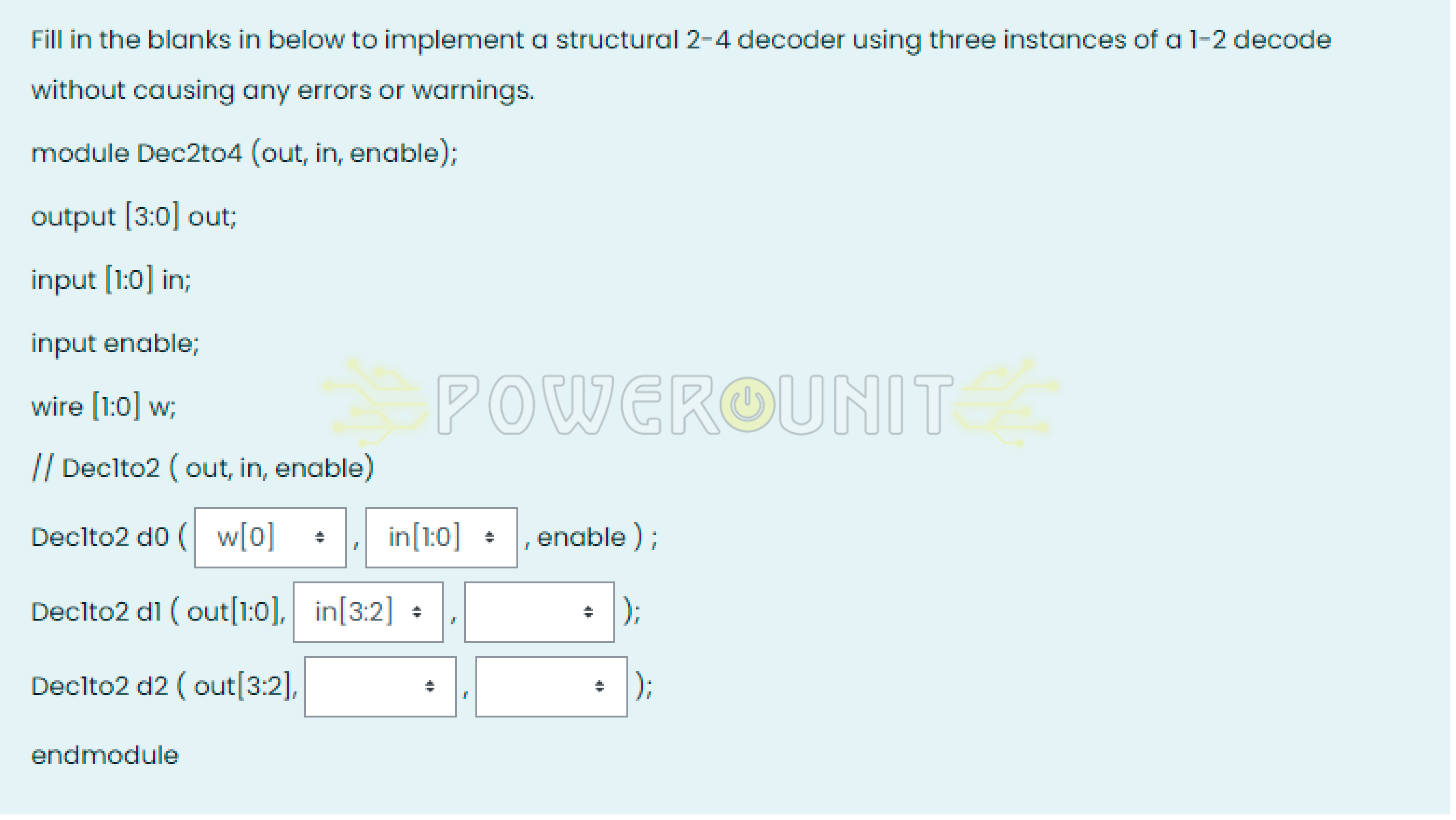


```
module circuit ( A, B, C, D, E, Y);

input A, B, C, D, E;

output Y;

wire x1, x2, x3, x4, x5, x6, x7, x8 ;

// module Block1( IN1, IN2, IN3, IN4, IN5, OUT);

Block1( A, B, C, D, E, x5);

//module Block2 ( IN1, IN2, OUT1, OUT2, OUT3, OUT4);

Block2 ( E, D, [ x1, x2, x3, x4 ÷ ] );

// module AndGate2( IN1, IN2, OUT);

AndGate2( x1, [ x5, x6 ÷ ] );

// module ORGate2( IN1, IN2, OUT);

ORGate2( x1, x2, x7);

// module XorGate2( IN1, IN2, OUT);

XorGate2( [ x3, x4 ÷ ] , x8);

// module Block3 (IN1, IN2, IN3, OUT);

Block3 ( [ x6, x7, x8, Y ÷ ] );
```

Fill in the blanks in the below code.



```
module circuit (clk, reset, enable);

   input, clk, reset, enable;

   wire [31:0] w1, w2, w3, w4, w5;

   //Register ( Q, D, clk, reset, enable )

   Register r0 ( w2 , w1 , clk, reset, enable );

   //Block1( R, A, B )

   Block1 b1 ( w5, w2[31:16] , w2[15:0] );

   //Register ( Q, D, clk, reset, enable )

   Register r1 ( w4 , w5 , clk, reset, enable );

   //Block2 ( R, A, clk)

   Block2 b2 ( w1, w4, clk);

endmodule
```

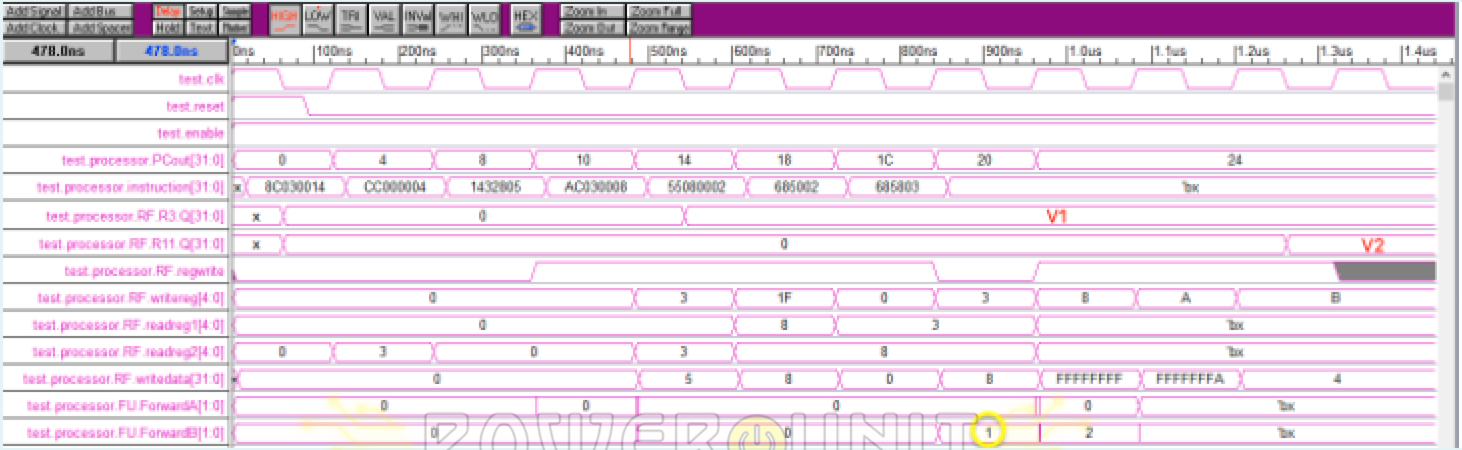


Fill in the blanks in below to implement a structural 2-4 decoder using three instances of a 1-2 decode without causing any errors or warnings.

```
module Dec2to4 (out, in, enable);

output [3:0] out;

input [1:0] in;

input enable;

wire [1:0] w;

// Dec1to2 ( out, in, enable)

Dec1to2 d0 ( [ w[0] ⬍ ] , [ in[1:0] ⬍ ] , enable ) ;

Dec1to2 d1 ( out[1:0], [ in[3:2] ⬍ ] , [ ⬍ ] );

Dec1to2 d2 ( out[3:2], [ ⬍ ] , [ ⬍ ] );

endmodule
```

Consider the below timing diagram for the pipelined processor implemented in experiment 9 to answer the below questions.

Note: if needed, **save** the image to your PC to see better.



What is the machine code of the flushed instruction (in hexadecimal)?

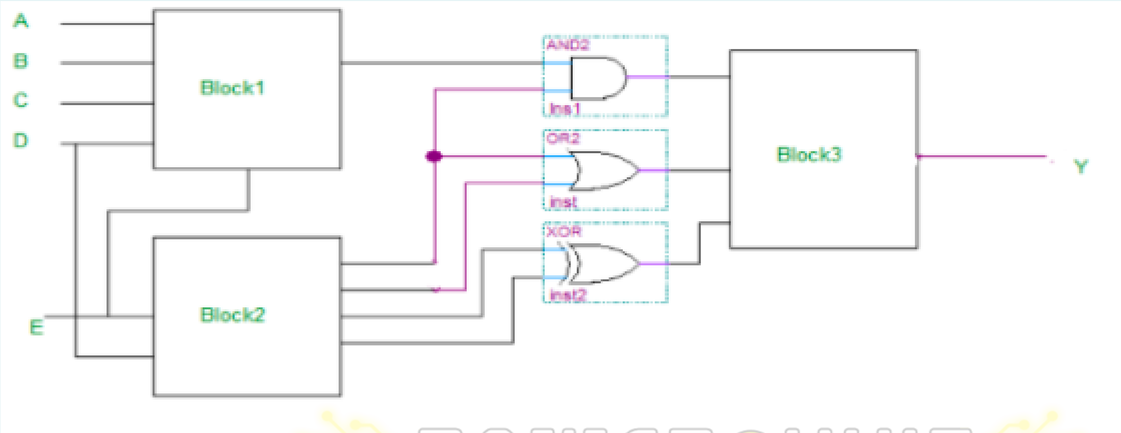
What is the value of V1 ?

How many instructions are stored in the instruction memory?

How many times execute-to-execute forwarding occurred?

What is value of the destination register (rd) when ForwardB=1 (shown inside a yellow circle) ?

In the below circuit, assume Block1 delay is 2ns, Block2 delay is 3ns, Block3 delay is 3ns, and AND, OR, XOR delays are 1ns. What is the delay of the entire circuit?



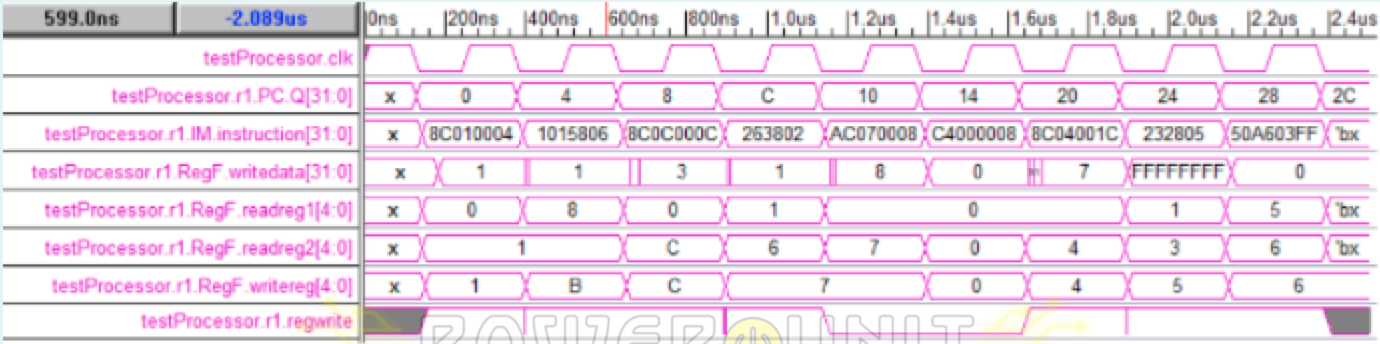
○ 6

○ 7

○ 10

○ 11

○ 13



Given that the above timing diagram is for the single cycle processor in experiment 6, answer the below true or false questions.

The instruction with the machine code AC070008 is a load instruction. False ⬍

The instruction with the machine code C4000008 is a jump instruction. True ⬍