Important Note: you can submit your solution only once so be careful.

6

Considering Verilog programming, which of the following statements is TRUE? *
(-/2 Points)

- Verilog is NOT case sensitive
- Each module must have at least two inputs
- You can NOT define more than one module in the same ".v" file
- Wires can NOT be used without definition in the module
- None of the above

Given the pin assignment in the following table, choose the correct number of input and output ports. *
(-/2 Points)

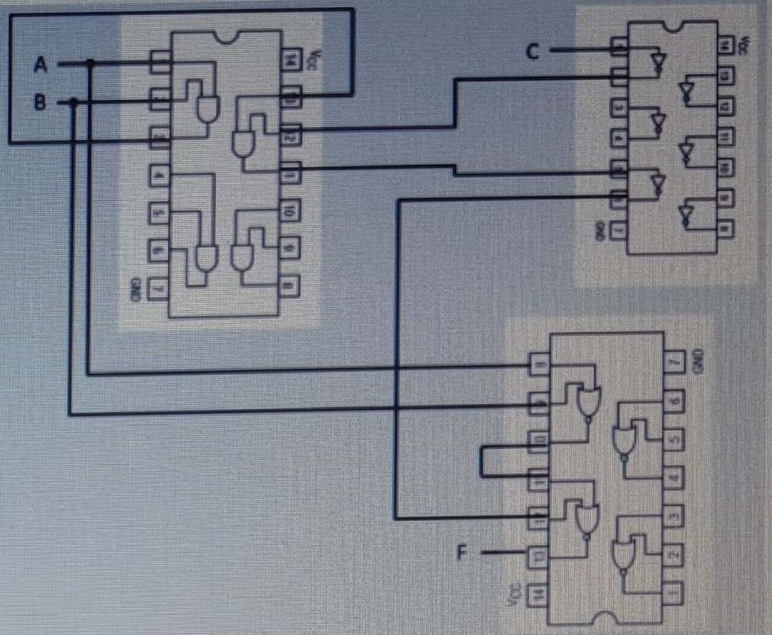
Port	Pin
A	HEX0_D[0]
B	HEX0_D[1]
C	HEX0_D[2]
D	HEX0_D[3]
E	HEX0_D[4]
F	HEX0_D[5]
G	HEX0_D[6]
H	SW[2]
I	SW[3]
J	LEDR [0]

Five input ports and five output ports.

Three input ports and seven output ports.

Two input ports and eight output ports.

8



The figure above shows function "F" implemented using IC's on breadboard (assume that VCC and GND pins are connected). Choose the correct assign statement for function F. *
(-/2 Points)

assign F=(~A&~B&~C)|~A|B

assign F=~((A&B&~C)|A|B)

assign F=~((~A&~B&~C)|~A|B)

assign F=~(~(A&B&~C)|~(A|B))

assign F=~((~A&B&~C)|A|~B)

9

Which of the following statements represents the main difference between functional simulation and timing simulation? *
(-/2 Points)

- Timing simulation requires more input combinations
- Functional simulation requires more input combinations
- Timing simulation is only used with registers and counters
- Functional simulation takes into account the delay of the gates
- Timing simulation takes into account the delay of the gates

10

Given the following Verilog modules and assuming that the top-level entity module "function"

10

Given the following Verilog modules and assuming that the top-level entity module "function" is simulated with X = 1, Y = 1, and Z =1. What are the values of w1, w3 and F? *
(-/2 Points)

```
module function (X, Y, Z, F);
  input X, Y, Z;
  output F;
  wire w1, w2, w3;
  Nandgate n1(X, Y, w1);
  Nandgate n2(w1, w1, w2);
  Nandgate n3(Z, Z, w3);
  Nandgate n4(w2, w3, F);
endmodule
```

```
module Nandgate (in0,in1,out);
  input in0,in1;
  output out;
  assign out = ~(in0 & in1);
endmodule
```

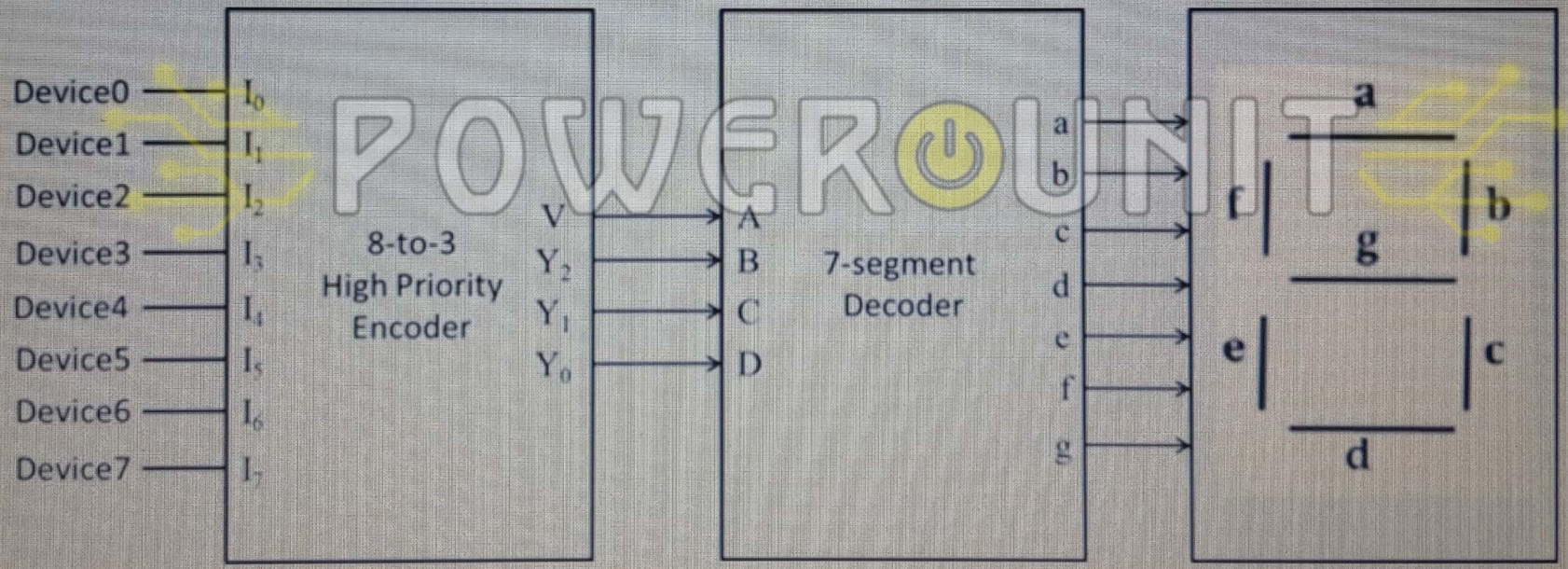

- w1=0,w3=0,F=1
- w1=0,w3=1,F=0
- w1=0,w3=0,F=0
- w1=1,w3=0,F=1
- None of the above

11

Consider the following monitoring system similar to the one discussed in experiment 4. The system receives 1-bit signal from 8 devices (i.e. Device0, Device 1, ...etc). Each device sends logic '0' if it is working properly; otherwise it sends logic '1'. The system works as follows:

- When all devices are working properly, the valid bit output (V) will be '0' and accordingly the 7-segment display is OFF.
- When there is at least one faulty device, the 7-segment displays the number of the faulty device with the highest priority.

If the 7-segment is displaying number 3, which of the following device status is possible? * (-/2 Points)



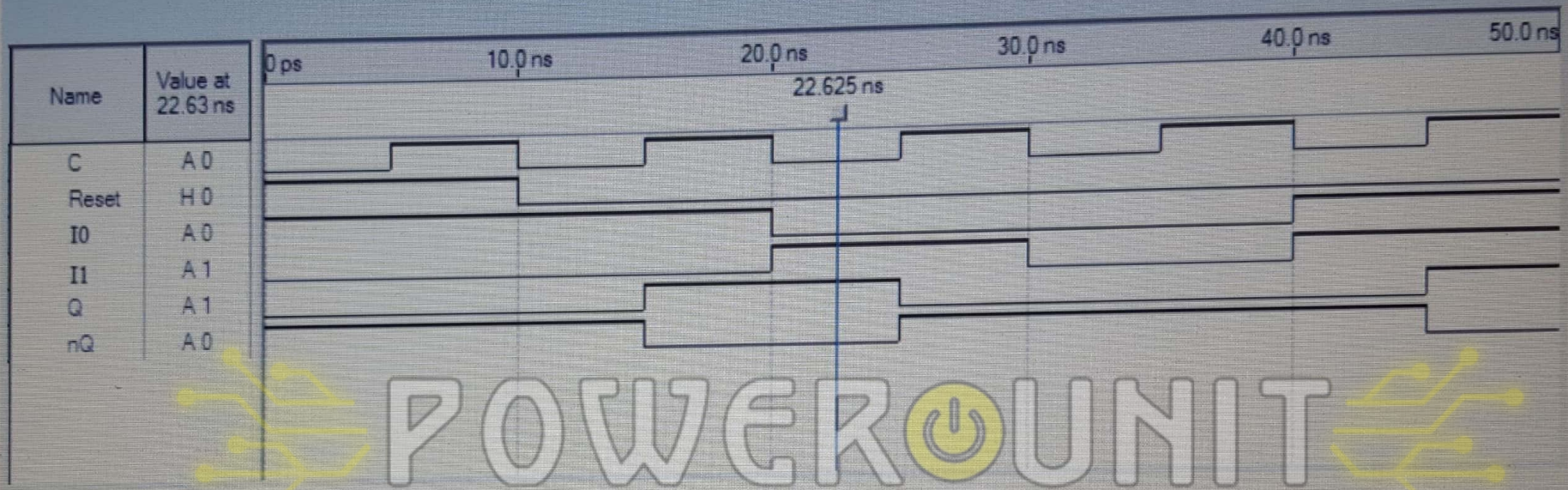
Device0 is faulty, all other devices are working properly

Devices 0 to 3 are faulty, Devices 4 to 7 are working properly

12

What is the type of the positive-edge flip flop whose simulation output is shown in the following figure? *

(-/2 Points)

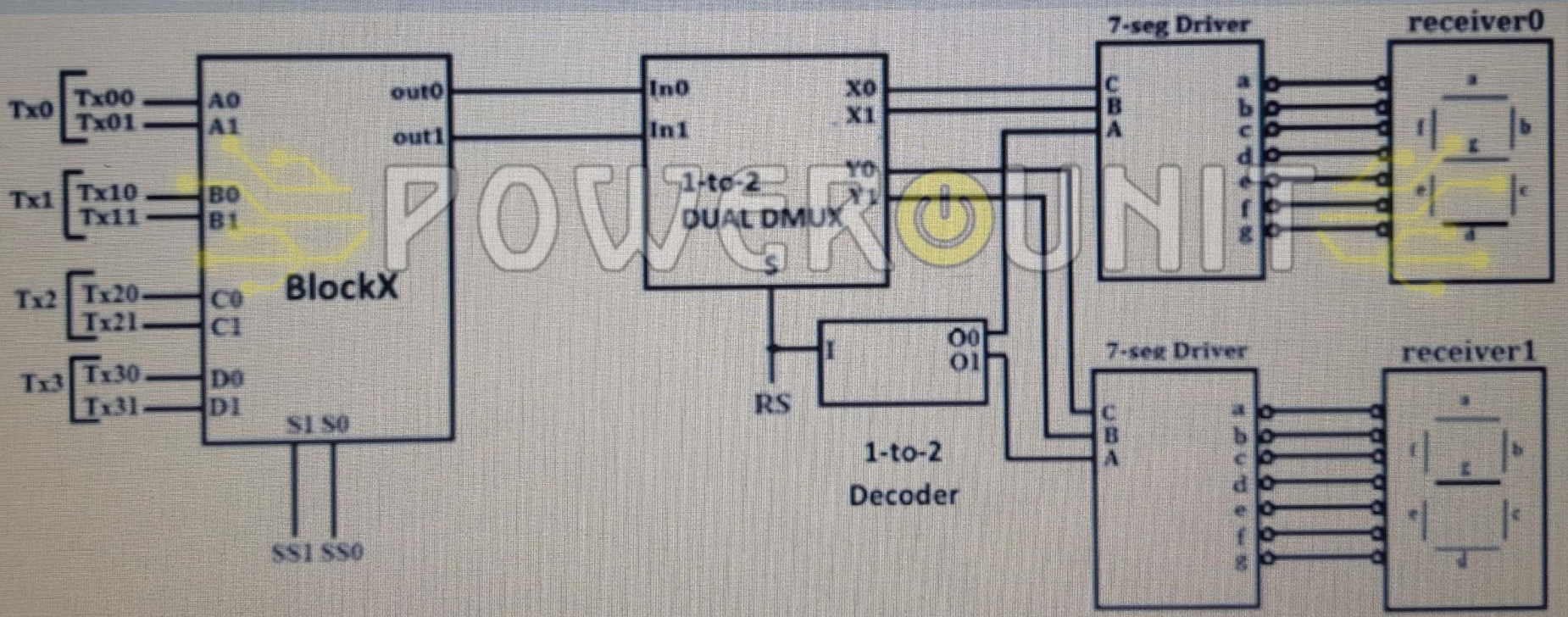


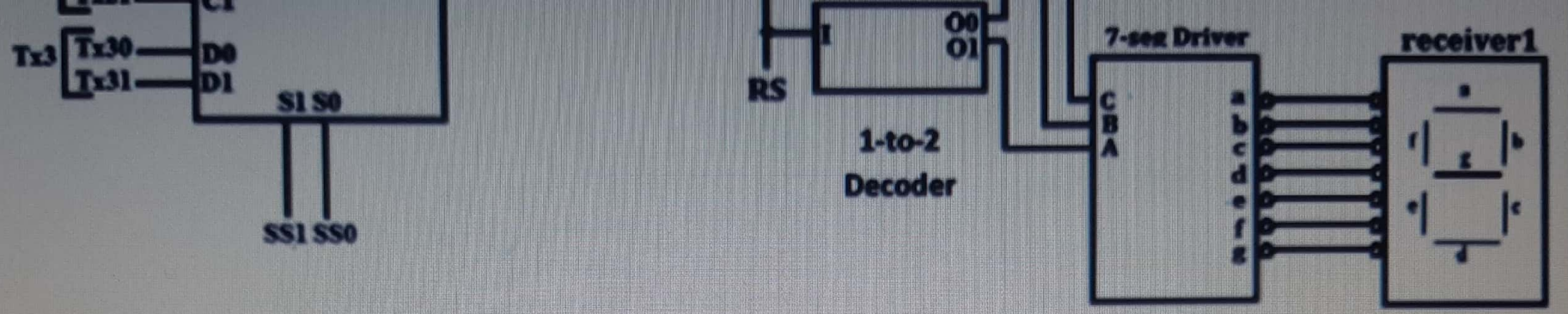
- D-flip flop.
- SR-flip flop
- T-flip flop
- JK-flip flop
- None of the above

Consider the following Time Division Multiplexing (TDM) circuit with four transmitters (TX0, TX1, TX2, and TX3) and two receivers (receiver0 and receiver1). Note that only the seven-segment display of the selected receiver should be turned ON to show the received value while the other seven-segment display should be turned OFF. Accordingly, answer questions 13 to 16 below.

Note:

- When input "A" of the 7-segment Driver is 0, then all segments (a -to- g) will be OFF.
- When input "A" of the7-segment driver is 1, then the 7-segment will display the number on inputs "B" and "C". For example if B=1 and C=0, then it will display 2 and so on.





13

Which of the following module instances can be used to implement BlockX in order to achieve the required functionality? *

(-/2 Points)

- Two instances of 4-to-1 MUX.
- Two instances of Tri-State buffer.
- Two instances of 4-to-2 Encoder.
- Four instances of 2-to-1 MUX.
- One instance of 1-to-4 Dual DMUX.

Four instances of 2-to-1 MUX.

One instance of 1-to-4 Dual DMUX.

14

Which of the following assign statements can be used for outputs X0 and X1 when writing the Verilog code of the 1-to-2 Dual DMUX behaviorally? *
(-/2 Points)

assign X0 = In1&~S , assign X1 = In0&S

assign X0 = In0&~S , assign X1 = In1&~S

assign X0 = In1&S , assign X1 = In0&~S

assign X0 = In0&~S , assign X1 = In1&S

assign X0 = In0&S , assign X1 = In1&~S

15

Assume that (Tx01=1, Tx00=0), (Tx11=0, Tx10=0), (Tx21=1, Tx20=1), (Tx31=0, and Tx30=1), (SS1 = 0, SS0 = 1), and RS = 1. What are the values that will be displayed on receiver0 and

- assign X0 = In0&~S , assign X1 = In1&S
- assign X0 = In0&S , assign X1 = In1&~S

15

Assume that (Tx01=1, Tx00=0), (Tx11=0, Tx10=0), (Tx21=1, Tx20=1), (Tx31=0, and Tx30=1), (SS1 = 0, SS0 = 1), and RS = 1. What are the values that will be displayed on receiver0 and receiver1? *
(-/2 Points)

- receiver0 = 0, receiver1 = all segments are OFF
- receiver0 = 3, receiver1 = all segments are OFF
- receiver0 = 2, receiver1 = all segments are OFF
- receiver0 = all segments are OFF, receiver1 = 1
- receiver0 = all segments are OFF, receiver1 = 0

16

When implementing the top level circuit structurally, determine the number of inputs, the

receiver0 = 2, receiver1 = all segments are OFF

- [x] receiver0 = all segments are OFF, receiver1 = 1
- [ ] receiver0 = all segments are OFF, receiver1 = 0

16

When implementing the top level circuit structurally, determine the number of inputs, the number of outputs, and the MINIMUM number of wires. *
(-/2 Points)

- [ ] Number of inputs = 8, Number of outputs = 14, Number of wires = 2
- [ ] Number of inputs = 8, Number of outputs = 6, Number of wires = 2
- [ ] Number of inputs = 11, Number of outputs = 14, Number of wires = 8
- [ ] Number of inputs = 11, Number of outputs = 6, Number of wires = 2
- [ ] Number of inputs = 10, Number of outputs = 14, Number of wires = 8

We want to implement a 2-bit ALU. The ALU has two 2-bit input numbers A {A1,A0} and B {B1,B0} and a 4-bit output number R {R3,R2,R1,R0}. The ALU circuit also has 2-bit control signal M1 and M0 that are used to choose the ALU operation as shown in the table below.

You are required to fill the blanks in the Verilog code modules in questions 17 to 29 in order to implement this ALU.
Use only the given wires, DO NOT declare any NEW wires.

M_1	M_0	Arithmetic or Logical Operation
0	X	{R3, R2, R1, R0} = A x B
1	0	R3 = 0, {R2, R1, R0} = A - B
1	1	R3 = 0, {R2, R1, R0} = A + B

17

Given the following module for a 2-input XOR gate, choose the correct answer to fill in the space.

```
module XOR2 (x, y, z);
    input x, y;
    output z;
    assign z =          ;
```

17

Given the following module for a 2-input XOR gate, choose the correct answer to fill in the space.

```
module XOR2 (x, y, z);
    input x, y;
    output z;
    assign z = _____;
endmodule *
```

(-/2 Points)

- x & y
- ~x | y
- ~ (x ^ y)
- ~(x | y)
- x ^ y

18

18

Given the module definition of the FullAdder cell below, choose the correct answer to fill in the space.

Note: You can use any of the following gates. Only the headers of these gates are provided; the implementations of the gates are omitted for brevity.

```
module XOR2 (x, y, z); // 2-input XOR gate: x and y are inputs, z is an output
module AND2 (x, y, z); // 2-input AND gate: x and y are inputs, z is an output
module OR2 (x, y, z); // 2-input OR gate: x and y are inputs, z is an output
module INV (x, z); // INVERTER gate: x is an input, z is an output

module FullAdder (in0, in1,Cin, sum, Cout);
    input in0, in1, Cin;
    output sum, Cout;
    wire w0;
    XOR2 gate0 (in0, in1, w0);
    _____

    assign Cout = in0&Cin | in1&Cin | in0&in1;
endmodule *
```

(-/2 Points)

```
AND2 gate1 (in0, in1, sum);

OR2 gate1 (in0, in1, sum);
```

the implementations of the gates are omitted for brevity

```
module XOR2 (x, y, z); // 2-input XOR gate: x and y are inputs, z is an output
module AND2 (x, y, z); // 2-input AND gate: x and y are inputs, z is an output
module OR2 (x, y, z); // 2-input OR gate: x and y are inputs, z is an output
module INV (x, z); // INVERTER gate: x is an input, z is an output

module FullAdder (in0, in1,Cin, sum, Cout);
      input in0, in1, Cin;
      output sum, Cout;
      wire w0;
      XOR2 gate0 (in0, in1, w0);
      ____________________________

      assign Cout = in0&Cin | in1&Cin | in0&in1;
endmodule *
```

(-/2 Points)

- AND2 gate1 (in0, in1, sum);
- OR2 gate1 (in0, in1, sum);
- XOR2 gate1 (in0, in1, sum);
- XOR2 gate1 (w0, Cin, sum);
- AND2 gate1 (Cin, Cout, sum);

19

Given the module definition of the 2-bit Adder/Subtractor cell below, choose the correct answers to fill in the spaces. The cell takes 2-bit input "a" {a1,a0} and 2-bit input "b" {b1,b0}. The output of the cell is 3-bit output {Cout,sum1,sum0}.

Note: Only the headers of the 2-input XOR gate and the Full Adder cell are provided; the implementation is omitted for brevity.

```
module XOR2 (x, y, z); // 2-input XOR gate: x and y are inputs, z is an output
module FullAdder (in0, in1,Cin, sum, Cout); //(in0, in1, and Cin) are inputs, (sum and Cout) are outputs

module TwoBitAddSub (a0, a1, b0, b1, m, sum0, sum1, Cout);
    input a0, a1, b0, b1, m;
    output sum0, sum1, Cout;
    wire w0, w1, w2;
    XOR2 gate0 (m, b0, w0);
    XOR2 gate1 (m, b1, w1);
    FullAdder FA0 (_____,_____,_____, sum0, _____);
    FullAdder FA1 (_____,_____,_____, sum1, _____);
endmodule
```


20

The FA0 line is: *

FullAdder FA1 (_____, _____, _____, sum1, _____);
endmodule

20

The FA0 line is: *
(-/2 Points)

- FullAdder FA0 (a0, w0, 0, sum0, Cout);
- FullAdder FA0 (a0, w0, m, sum0, w2);
- FullAdder FA0 (a0, b0, m, sum0, w2);
- FullAdder FA0 (a0, b0, 1, sum0, Cout);
- None of the above

21

The FA1 line is: *
(-/2 Points)

- FullAdder FA0 (a0, b0, 1, sum0, Cout);
- None of the above

21

The FA1 line is: *
(-/2 Points)

- FullAdder FA1 (a1, b1, 0, sum1, w2);
- FullAdder FA1 (a1, w1, w0, sum1, Cout);
- FullAdder FA1 (a1, w1, w2, sum1, Cout);
- FullAdder FA1 (a1, b1, w1, sum1, w2);
- None of the above

22

Given the module definition of the 2-bit Multiplier cell below, choose the correct answers to fill in the spaces. The Multiplier takes 2-bit input "a" {a1,a0} (i.e. Multiplicand) and 2-bit input "b"

22

Given the module definition of the 2-bit Multiplier cell below, choose the correct answers to fill in the spaces. The Multiplier takes 2-bit input "a" {a1,a0} (i.e. Multiplicand) and 2-bit input "b" {b1,b0} (i.e. Multiplier). The output of the Multiplier is 4-bit output "n" {n3,n2,n1,n0}.

Note: Only the headers of the 2-input AND gate and the 2-bit Adder/Subtractor are provided; the implementations are omitted for brevity.

```
module AND2 (x, y, z); // 2-input AND gate: x and y are inputs, z is an output
module TwoBitAddSub (a0, a1, b0, b1, m, sum0, sum1, Cout) // (a0, a1, b0, b1, and m) are
inputs, (sum0, sum1, and Cout) are outputs

module Multiplier (a0, a1, b0, b1, n0, n1, n2, n3);
      input a0, a1, b0, b1;
      output n0, n1, n2, n3;
      wire w0, w1, w2;
      AND2 gate0 (a0, b0,_____);
      AND2 gate1 (a1, b0, w0);
      AND2 gate1 (a0, b1, w1);
      AND2 gate1 (a1, b1, w2);
      TwoBitAddSub (____,_____,_____,_____,_____,_____,_____,_____);
endmodule
```

endmodule

23

The line of the gate0 is: *
(-/2 Points)

- AND2 gate0 (a0, b0, n0);
- AND2 gate0 (a0, b0, n1);
- AND2 gate0 (a0, b0, n2);
- AND2 gate0 (a0, b0, n3);
- None of the above

24

The line of the TwoBitAddSub is: *
(-/2 Points)

- AND2 gate0 (a0, b0, n1);
- AND2 gate0 (a0, b0, n2);
- AND2 gate0 (a0, b0, n3);
- None of the above

24

The line of the TwoBitAddSub is: *
(-/2 Points)

- TwoBitAddSub (a0, a1, b0, b1, n0, n1, n2, n3);
- TwoBitAddSub (a0, a1, b0, b1, 0, n1, n2, n3);
- TwoBitAddSub (w0, 0, w1, w2, 0, n1, n2, n3);
- TwoBitAddSub (w0, 0, w1, w2, 0, n0, n1, n2);
- TwoBitAddSub (w0, a1, w1, w2, 0, n0, n1, n2);

25

25

Given the following module for a 2-to-1 MUX, choose the correct answer to fill in the space.

```
module MUX2to1 (i0, i1, s, out);
    input i0, i1, s;
    output out;
    assign out = _____;
endmodule *
```

(-/2 Points)

- s & (i0 | i1)
- s | (i0 & i1)
- s & (~i0) | s & i1
- s & i0 | s & (~i1)
- (~s) & i0 | s & i1

26

Given the TOP LEVEL module definition of the 2-bit ALU below, choose the correct answers to

26

Given the TOP LEVEL module definition of the 2-bit ALU below, choose the correct answers to fill in the spaces.

Note: Only the headers of the Inverter gate, the 2-bit Adder/Subtractor, the 2-bit Multiplier, and the 2-to-1 MUX are provided; the implementations are omitted for brevity.

```
module INV (x, z); // INVERTER gate: x is an input, z is an output
module TwoBitAddSub (a0, a1, b0, b1, m, sum0, sum1, Cout); // (a0, a1, b0, b1, and m) are inputs, (sum0, sum1, and Cout) are outputs
module Multiplier (a0, a1, b0, b1, n0, n1, n2, n3); // (a0, a1, b0, b1) are inputs, (n0, n1, n2, n3) are outputs
module MUX2to1 (i0, i1, s, out); // (i0, i1, s) are inputs, out is an output


module TwoBitALU (A0, A1, B0, B1, M0, M1, R0, R1, R2, R3)
    input A0, A1, B0, B1, M0, M1;
    output R0, R1, R2, R3;
    wire w0, w1, w2, w3, w4, w5, w6, w7;
    INV inv (M0, w7);
    TwoBitAddSub alu (A0, A1, B0, B1,_____, w0, w1, w2);
    Multiplier mul (A0, A1, B0, B1, w3, w4, w5, w6);
    MUX2to1 mux0 (w3, w0, _____, R0);
    MUX2to1 mux1 (w4, w1, _____, R1);
    MUX2to1 mux2 (w5, w2, _____, R2);
    MUX2to1 mux3 (_____,_____,_____, R3);
endmodule
```

MUX2to1 mux3 (_____,_____,_____, R3);
endmodule

27

The value of the space in the line "TwoBitAddSub alu (A0, A1, B0, B1,_____, w0, w1, w2);" is: * (-/2 Points)

- M0
- M1
- w3
- w4
- w7

28

Given that the values of the spaces in mux0, mux1, and mux2 lines are the same, choose the correct value of these spaces. *

w4

w7

28

Given that the values of the spaces in mux0, mux1, and mux2 lines are the same, choose the correct value of these spaces. *
(-/2 Points)

M0

M1

w3

w4

w7

29

The values of the spaces in the line "MUX2to1 mux3 (_____,_____,_____, R3);" are: *

w4

w7

29

The values of the spaces in the line "MUX2to1 mux3 (_____,_____,_____, R3);" are: *
(-/2 Points)

w2, w6, M0

w6, w2, M1

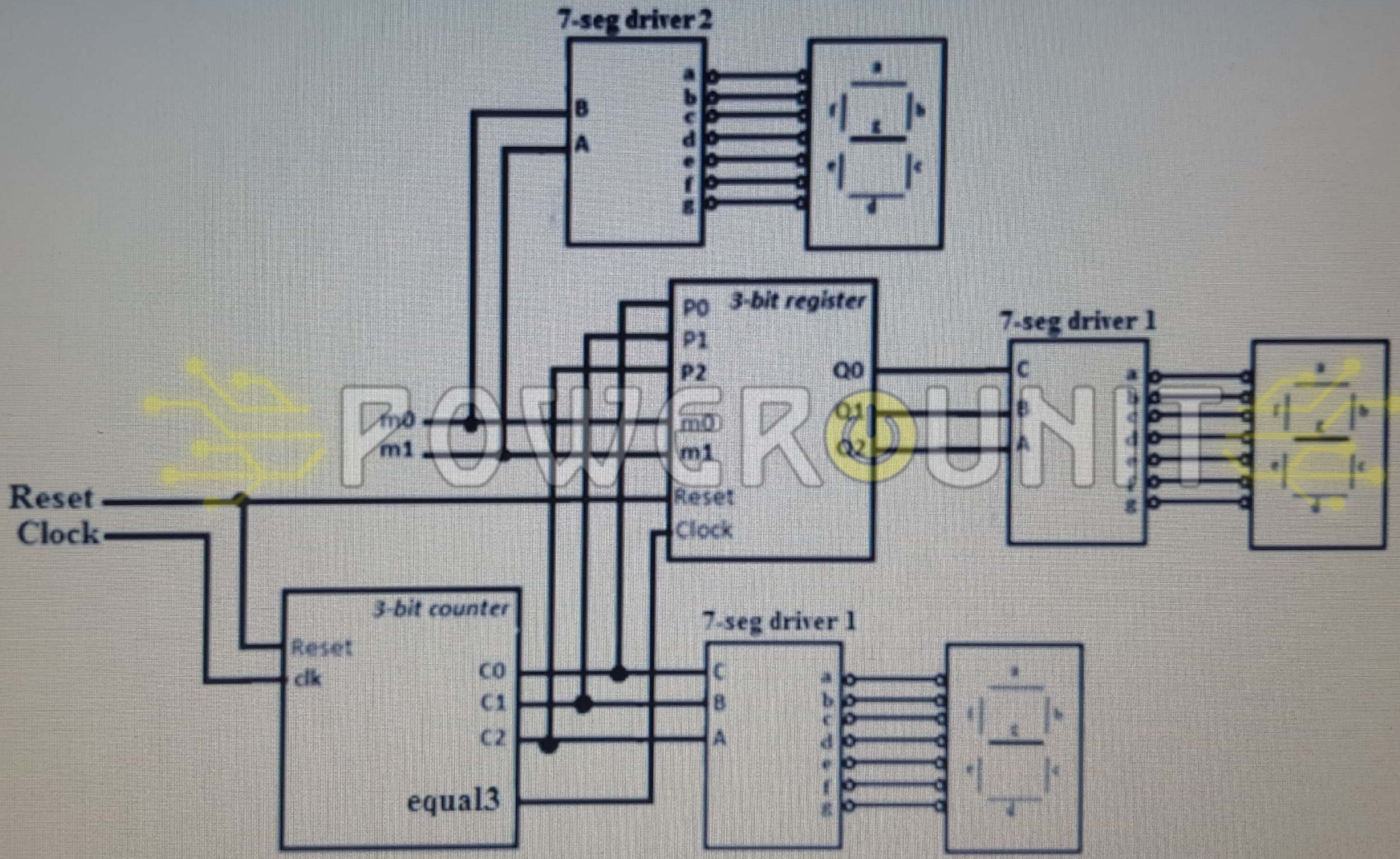
w6, 0, M1

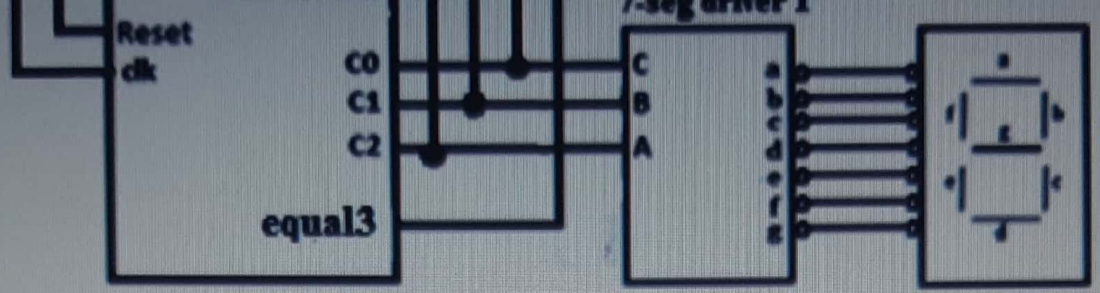
w6, w2, w7

w6, 1, M1

Given the following sequential circuit similar to the one described in experiment 8, answer the questions 30 to 33 below.

Given the following sequential circuit similar to the one described in experiment 8, answer the questions 30 to 33 below.





30

The "equal3" output of the 3-bit counter shown in the figure generates 1 when the value of the counter is "3" otherwise it generates 0. Which of the following module instances can be used in order to implement the "equal3" structurally? *
(-/2 Points)

- One INVERTER gate and two 2-input AND gates.
- One INVERTER gate and two 2-input OR gates.
- Two INVERTER gates and two 2-input AND gates.
- Two INVERTER gates and two 2-input OR gates.
- Two 2-input NOR gates.

31

31

Fill-in the blanks to complete the required instantiation of the 4-to-1 MUX in the 3-bit register module below so that it operates according to the following table:

Only the headers of the 4-to-1 MUX and the DFF are provided; the implementations are omitted for brevity.

```
module mux1 (i0,i1,i2,i3,s0,s1,out); //4-to-1 MUX: i0-to-i3 are the data inputs, s0 and s1 are the selection lines.
module dff1(D,Reset,Clock,Q,Qnot); //D-flip flop with Q and Qnot as outputs

//3-bit register
module reg3b(P0, P1, P2, m0, m1, Reset, Clock, Q0, Q1, Q2);
input P0, P1, P2, m0, m1, Reset, Clock;
output Q0, Q1, Q2;

mux1 mx0(_____,_____,_____,_____,m0,m1,d0); //MUX at the input of the least significant bit of register
//omitted instances of muxes

dff1 ff0(d0, Reset, Clock,Q0,Q0not);
dff1 ff1(d1, Reset, Clock,Q1,Q1not);
dff1 ff2(d2, Reset, Clock,Q2,Q2not);

endmodule *
```

(-/2 Points)

m1	m0	Operation

```
dff1 ff0(d0, Reset, Clock,Q0,Q0not);
dff1 ff1(d1, Reset, Clock,Q1,Q1not);
dff1 ff2(d2, Reset, Clock,Q2,Q2not);

endmodule *
```

(-/2 Points)

m1	m0	Operation
0	0	Rotate left
0	1	Parallel Load
1	0	Complement (toggle all bits)
1	1	Hold

- 0, P0, Q0not, Q0
- Q2, P0, Q0, Q0not
- 0, P0, Q0, Q0not
- Q2, P0, Q0not, Q0
- Q0, Q0not, P0, Q2

Q0, Q0not, P0, Q2

32

Assume that:
1) current counter value is C2, C1, C0 = 1,0,0
2) current register value is Q2, Q1, Q0 = 1, 1, 0
3) current m1, m0 = 0, 1

Given that at the next positive edge of the Clock, the counter value will change to C2, C1, C0 = 0, 1, 1. What would be the new value stored in the 3-bit register? *
(-/2 Points)

Q2, Q1, Q0 = 1, 1, 0

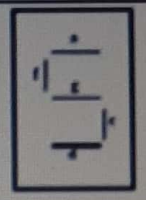
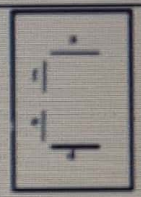
Q2, Q1, Q0 = 0, 1, 1

Q2, Q1, Q0 = 0, 0, 1

Q2, Q1, Q0 = 1, 0, 1

None of above

33

A	B	Letter
0	0	
0	1	
1	0	
1	1	

Fill-in the required behavioral implementation of output "a" in the 7-segdriver2 module below such that it shows a letter that represents the selected register operation according to the table above (Assume we are using common anode 7-segment):

```
// segdriver2
module segdriver2 (A,B,a,b,c,d,e,f,g);
input A,B;
output a,b,c,d,e,f,g;

assign a = _____;
```

```
// segdriver2
module segdriver2 (A,B,a,b,c,d,e,f,g);
input A,B;
output a,b,c,d,e,f,g;

assign a = _____;

//omitted implementations of b-g

endmodule *
```

(-/2 Points)

- ~A & B
- ~(A & B)
- A & B
- A | B
- ~A & ~B