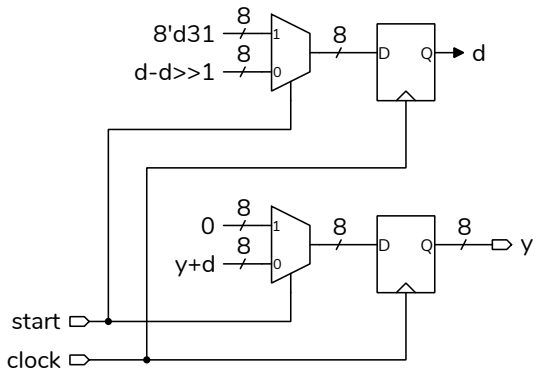
Solutions to Final Exam

There were two versions of each question. The answers for both versions are given below.

Question 1 — 5 marks

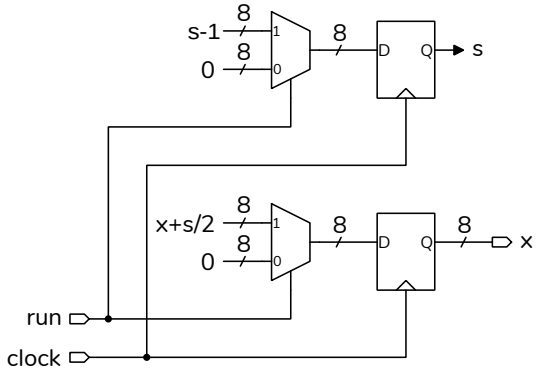
First version

Write a Verilog module named **vpos** corresponding to the diagram shown below. The module has a **1**-bit input **start**, a clock input **clock**, and an **8**-bit registered output **y**. Bits should be numbered in decreasing order. Follow the course coding conventions. You may omit comments.



Second Version

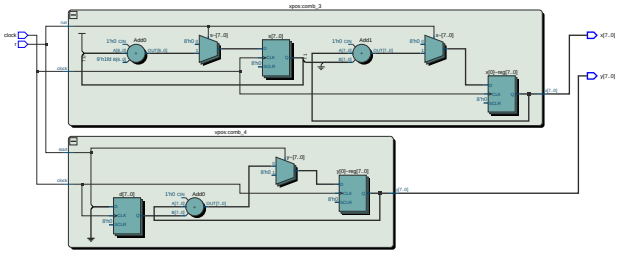
Write a Verilog module named **xpos** corresponding to the diagram shown below. The module has a **1**-bit input **run**, a clock input **clock**, and an **8**-bit registered output **x**. Bits should be numbered in decreasing order. Follow the course coding conventions. You may omit comments.



Answers

```
// Question 1 First version

module vpos ( input logic start, clock,
              output logic [7:0] y ) ;

   logic [7:0] d ;

   always_ff @(posedge clock) d
     <= start ? 8'd31 : d - d>>1 ;

   always_ff @(posedge clock) y
     <= start ? 0 : y + d ;

endmodule

// Question 1 Second version

module xpos ( input logic run, clock,
              output logic [7:0] x ) ;

   logic [7:0] s ;

   always_ff @(posedge clock) s
     <= run ? s-1 : 0 ;

   always_ff @(posedge clock) x
     <= run ? x+s/2 : 0 ;

endmodule

module exam1 ( input logic r, clock,
               output logic [7:0] x, y ) ;

   xpos ( r, clock, x ) ;
   vpos ( r, clock, y ) ;

endmodule
```

The Quartus synthesis results are shown below:



Note that the **vpos** schematic generated by Quartus has optimized away the multiplexer because the expression **d-d>>1** simplifies to 0[1] and thus the multiplexer output is either 0 when **start** is 0 or **8'h1f** when **start** is 1. This can be achieved simply by connecting bits 4 through 0 of the register input to **start**.

[1] This was unintentional.

Question 2 — 8 marks

Fill in the table below with the value of each expression. Give your answers as Verilog numeric literals including the length and using a hexadecimal base. Assume the following declarations:

Answers

For:

```
logic [7:0] x = 8'hca ;
logic [3:0] y = 4'b1001 ;
```

expression	value
{x[3:0],y[3:0],x[7:4]}	12'ha9c
!x && x	1'h0
~x | x	8'hff
x << 2 + 2'd3	8'h40
y/2	32'h4
x[3:0] > y	1'h1
x^y	8'hc3
x[0] ? x : y	8'h9

For:

```
logic [7:0] x = 8'hac ;
logic [3:0] y = 4'b1001 ;
```

expression	value
{x[3:0],y[3:0],x[7:4]}	12'hc9a
!x && x	1'h0
~x | x	8'hff
x << 2 + 2'd3	8'h80
y/2	32'h4
x[3:0] > y	1'h1
x^y	8'ha5
x[0] ? x : y	8'h9

Question 3 — 3 marks

Fill in the blank boxes in the table below so that all values in each row are consistent (agree with each other). The first row is an example.

Answers

signal name	truth value (T/F)	truth value in an expression (0/1)	logic level (H/L)	Verilog value when input (0/1)
$\overline{\texttt{busy}}$	T	1	L	0
dark	T	1	H	1
even*	T	1	L	0
$\overline{\texttt{short}}$	T	1	L	0

signal name	truth value (T/F)	truth value in an expression (0/1)	logic level (H/L)	Verilog value when input (0/1)
$\overline{\texttt{busy}}$	T	1	L	0
dark	F	0	L	0
even*	F	0	H	1
$\overline{\texttt{short}}$	T	1	L	0

Question 4 — 4 marks

First version:

A module with inputs **f** and **b**, and a clock signal, **clock**, is declared as:

```
module statemachine
  ( input logic f, b, clock,
    output logic [1:0] state ) ;

   // your code here

endmodule
```

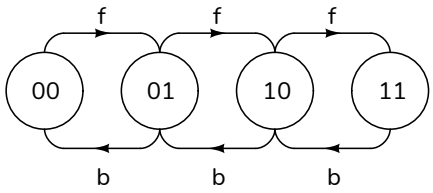
Second version:

A module with inputs **p** and **n**, and a clock signal, **clock**, is declared as:

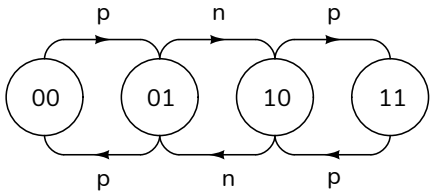
```
module statemachine
  ( input logic p, n, clock,
    output logic [1:0] state ) ;

   // your code here

endmodule
```

The module's behaviour can be described as a state machine with four states that have values `2'd0` through `2'd3`. The state transitions are as described in the state transition diagram below:

First version:



Second version:



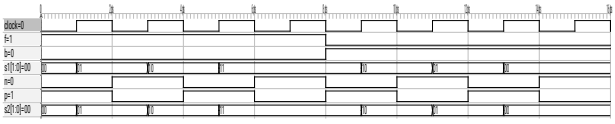
Write (a) Verilog statement(s) below that, if placed after the comment, would result in the behaviour described in the state transition diagram above. Do not repeat code already given. Follow the course coding conventions. You may omit comments.

Answers

A listing of the two state machines, a test bench and the simulation waveforms is shown below.

```
module statemachine
  ( input logic f, b, clock,
    output logic [1:0] state ) ;

   always_ff @(posedge clock)
     state <= state == 2'b00 && f ? 2'b01 :
              state == 2'b01 && f ? 2'b10 :
              state == 2'b10 && f ? 2'b11 :
              state == 2'b11 && b ? 2'b10 :
              state == 2'b10 && b ? 2'b01 :
              state == 2'b01 && b ? 2'b00 :
              state ;

endmodule

module statemachine2
  ( input logic p, n, clock,
    output logic [1:0] state ) ;

   always_ff @(posedge clock)
     state <= state == 2'd0 && p ? 2'd1 :
              state == 2'd1 && n ? 2'd2 :
              state == 2'd2 && p ? 2'd3 :
              state == 2'd1 && p ? 2'd0 :
              state == 2'd2 && n ? 2'd1 :
              state == 2'd3 && p ? 2'd2 :
              state ;

endmodule
```

```
module sm_tb ;

   logic f, b, p, n, clock ;
   logic [1:0] s1, s2 ;
   logic fb [] = {1,1,1,1,0,0,0,0} ;
   logic pn [] = {1,0,1,0,1,0,1,0} ;

   statemachine  sm0 (.state(s1),.*) ;
   statemachine2 sm1 (.state(s2),.*) ;

   initial begin
      $dumpfile("sm.vcd") ;
      $dumpvars ;
      { sm0.state, sm1.state } = '0 ;
      clock = '0 ;
      for ( int i=0 ; i<8 ; i++ ) begin
         {f,b,p,n} = {fb[i],~fb[i],pn[i],~pn[i]} ;
         #1 clock = 1 ;
         #1 clock = 0 ;
      end
      $finish ;
   end

endmodule
```



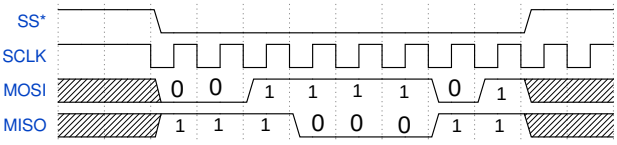
Question 5 — 4 marks

The following waveform shows signals on an SPI interface that transfers bits most-significant-bit first. How many bits were transferred in each direction? What value was transmitted from the master to the slave? What value was transmitted from the slave to the master? Give you answers as a hexadecimal number. Show your work.

Answers

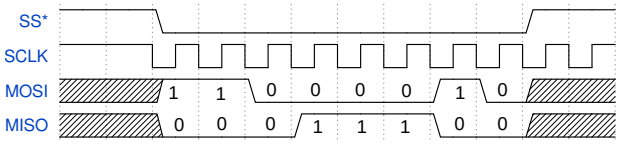
The bits transferred are marked on the diagrams below.

First version:



In this case 8 bits were transferred in each direction. From the master to the slave (MOSI) the bits were 7'b0011_1101 or 3d in hexadecimal and from the slave to the master (MISO) the bits were 7'b1110_0011 or e3 in hexadecimal.

Second version:



In this case 8 bits were also transferred in each direction. From the master to the slave (MOSI) the bits were 7'b1100_0010 or c2 in hexadecimal and from the slave to the master (MISO) the bits were 7'b0001_1100 or 1c in hexadecimal.

Question 6 — 2 marks

The maximum propagation delay through any combinational logic path in a CPLD is 7 (or 17) ns, the minimum setup time of its registers is 2 ns and the maximum clock-to-output delay is 1 ns. What is the fastest clock frequency at which this CPLD can operate?

Answers

Setting available setup time equal to the setup time requirement (no slack) and solving for the clock period:

$$\begin{aligned} t_{clock} &= t_{CO} + t_{SU} + t_{PD} \\ &= 1 + 2 + 7 \text{ (or 17)} \\ &= 10 \text{ ns (or 20 ns)} \end{aligned}$$

And the corresponding maximum clock frequency is $1/t_{clock}$ = 100 MHz (or 50 MHz).

Question 7 — 2 marks

A digital power quality analyzer needs to measure the level of harmonics up to the 40 (or 50)'th harmonic of the 60 Hz power line frequency (i.e. up to 2400 (or 3000) Hz) with a quantization SNR of over 60 (or 80) dB.

What minimum ADC sampling rate is required? What number of bits of resolution is required?

Answers

The minimum sampling rate must be more than twice the bandwidth (highest frequency). In this case $> 2 \times 2400 =$ > 4800 Hz (or > 6000 Hz).

The quantization SNR in dB is (roughly) given by $6B$ where B is the number of bits. Solving $60 = 6B$ for B gives $B =$ 10 bits (or $B = 80/6 = 13.3$ so 14 bits are required.

Question 8 — 2 marks

You are choosing the logic levels for a new design and need to maintain a noise margin of 1 (or 1.5) V for both high and low logic levels. $V_{IL(max)}$ is 2 V and $V_{IH(min)}$ is 3 V. What are $V_{OL(max)}$ and $V_{OH(min)}$?

Answers

Using the equations:

- noise margin(low) = $V_{IL(max)}$ - $V_{OL(max)}$
- noise margin(high) = $V_{OH(min)}$ - $V_{IH(min)}$

and solving for $V_{OL(max)}$ and $V_{OH(min)}$:

- $V_{OL(max)}$ = $V_{IL(max)}$ - noise margin(low) = $2 - 1 =$ 1 V (or $2 - 1.5 =$ 0.5 V)
- $V_{OH(min)}$ = noise margin(high) + $V_{IH(min)}$ = $1 + 3 =$ 4 V (or $1.5 + 3=$ 4.5 V)

Question 9 — 2 marks

A CMOS digital logic circuit operates from a 1.5 V battery for 20 hours before the battery is exhausted. How long would the same battery last if the clock frequency was reduced by a factor of 10 (or 20)? *Hint: Battery life is inversely proportional to current consumption.*

Answers

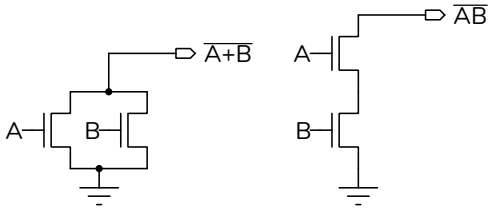
The power consumption is proportional to the clock frequency. Since the voltage is unchanged, the current consumption will also be proportional to the clock frequency. Reducing the clock rate by a factor of 10 (or 20) will increase the battery life by a factor of 10 (or 20) to 200 (or 400) hours.

Question 10 — 1 marks

Draw the schematic of a two-input NAND (or NOR) gate with an *open-drain* output. Use MOSFET transistors.

Answers

The *open-drain* output NOR and NAND gate schematics are shown below. Note that an open-drain output does not source current and any pull-up resistors are connected at the output of the gate.



Question 11 — 5 marks

For each term in the left column write the number of the most appropriate match in the right column. There is only one best match for each term. No marks will be deducted for wrong answers.

Answers

fabless	manufacturer	4	1
die	chip	2	2
TQFP	package	3	4
wafer	300 mm	1	5
CPLD	programmable	5	3