EEL 4712 – Digital Design
Test 2 – Spring Semester 2008 Name _____

IMPORTANT:
- Please be neat and write (or draw) carefully. If we cannot read it with a reasonable effort, it is assumed wrong.
- As always, the best answer gets the most points.

COVER SHEET:

Problem:	Points:
1 (12 pts)	
2 (20 pts)	
3 (10 pts)	
4 (15 pts)	
5 (18 pts)	
6. (25 pts)	

Total ☐

Re-Grade Information:

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

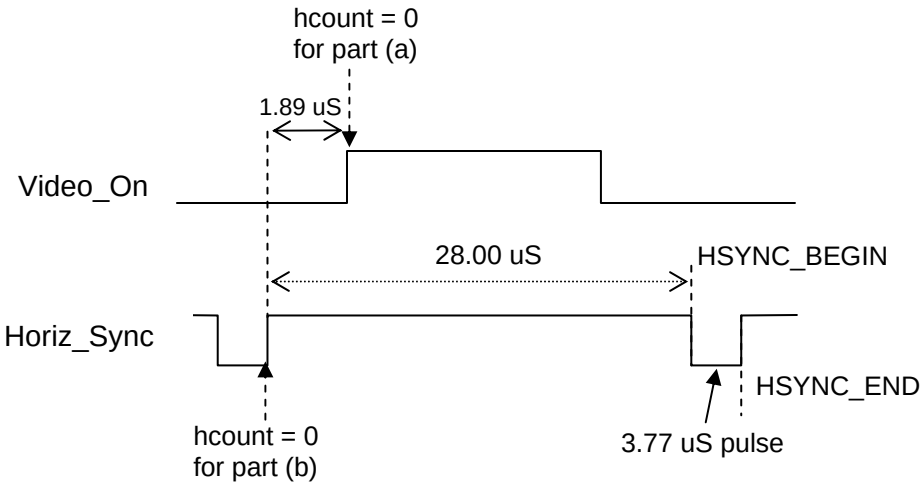
_____

_____

12 pts.

1. <u>Miscellaneous.</u> For all three parts of Problem 1, assume a 25.175 MHz board clock rate.

VGA display calculation:



(a) Assume we set the "reference point" for hcount = 0 as shown, what would be the constant values for hcount for HSYNC_BEGIN and HSYNC_END? (4 pts)

HSYNC_BEGIN = _____ HSYNC_END = _____

For credit, please show work:

(b) Assume we set the "reference point" for hcount = 0 as shown, what would be the constant values for hcount for HSYNC_BEGIN and HSYNC_END? (4 pts)

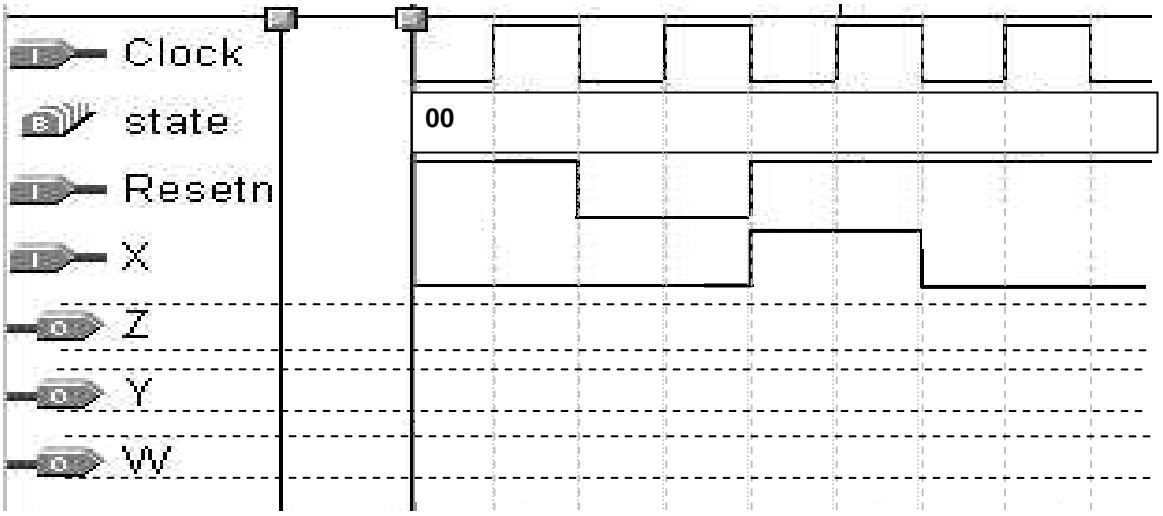
HSYNC_BEGIN = _____ HSYNC_END = _____

For credit, please show work:

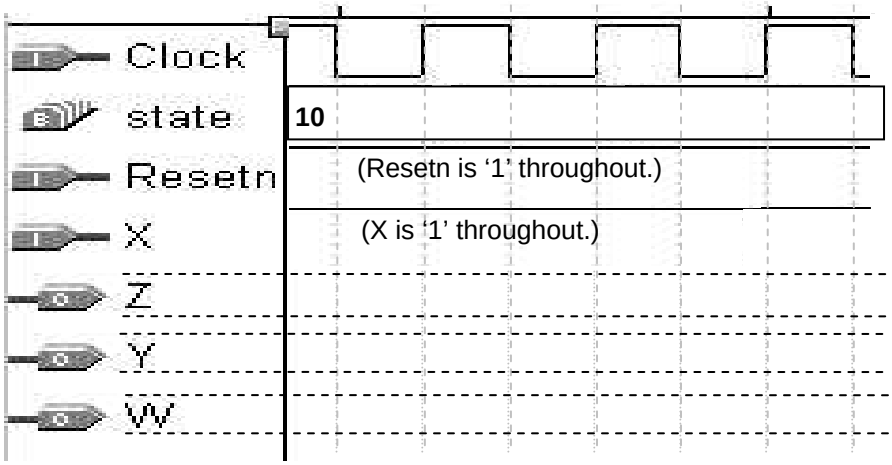
(c) Using a N-bit counter to implement a clock divider, how many flipflops are need to obtain a divided clock with a period of approximately 40 us. For credit, please show work. (4 pts)

20 pts.

2. Based on the VHDL code at the bottom of this page and on the next page, complete the following two timing diagrams. Specify the values for state (in binary), Z, Y, and W. Note that the two timing diagrams are **independent** of each other, with different initial values. Also note that the initial value of all flip-flops are '0'.



Be sure to show timing delays. Also, complete the timing diagrams to the end.



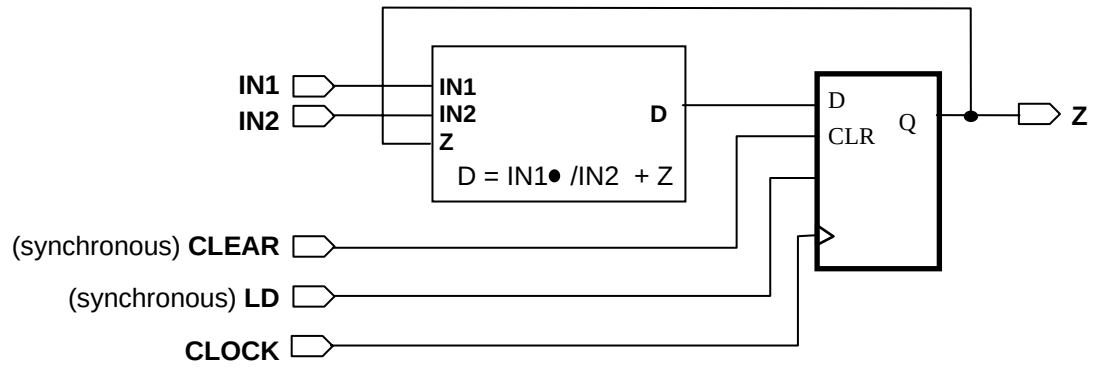

```
ENTITY T2Prob1 IS
PORT (Clock, Resetn, X :  IN STD_LOGIC;
        Y,Z,W   : OUT    STD_LOGIC ) ;
END T2Prob1 ;
```

(The architecture section is on the next page.)

```
ARCHITECTURE Behavior OF T2Prob1 IS
    SIGNAL state : STD_LOGIC_Vector (1 DOWNTO 0);
    SIGNAL StateA, StateB, StateC, StateD : STD_LOGIC;
BEGIN
    StateA <= '1' WHEN state = "11" ELSE '0';
    StateB <= '1' WHEN state = "10" ELSE '0';
    StateC <= '1' WHEN state = "01" ELSE '0';
    Z <= StateA OR StateB OR (StateC AND NOT X);

    PROCESS (Clock )
    BEGIN
        IF (Clock'EVENT AND Clock = '1')THEN
          IF Resetn = '0' THEN state(1) <= '0' ;
          ELSE
                CASE state IS
                   WHEN "11" =>
                         IF X = '0' THEN state(1) <= '0' ;
                         ELSE state(1) <= '1' ;
                                W <= '0';
                         END IF ;
                   WHEN "10" =>
                         state(1) <= '0' ;
                   WHEN "01" =>
                         IF X = '0' THEN state(1) <= '1';
                         ELSE state(1) <= '0';
                                W <= '1';
                         END IF ;
                   WHEN "00" =>
                         state(1) <= '1';
                   WHEN OTHERS =>
                         state(1) <= '1';
                END CASE ;
       END IF ;
      END IF ;
    END PROCESS ;

    PROCESS
    BEGIN
          WAIT UNTIL (Clock'EVENT AND Clock = '1');
           IF Resetn = '0' THEN state(0) <= '1' ;
           ELSE
               CASE state IS
                   WHEN "11" =>
                         IF X = '0' THEN state(0) <= '0' ;
                         ELSE state(0) <= '0' ;
                         END IF ;
                   WHEN "10" =>
                         state(0) <= '1' ;
                         Y <= '1';
                   WHEN "01" =>
                         IF X = '0' THEN state(0) <= '1';
                         ELSE state(0) <= '1';
                         END IF ;
                   WHEN "00" =>
                         state(0) <= '1';
                   WHEN OTHERS =>
                         state(0) <= '1';
                         Y <= '0';
              END CASE ;
           END IF ;
    END PROCESS ;
END Behavior ;
```

3. Given the following circuit:

10 pts.

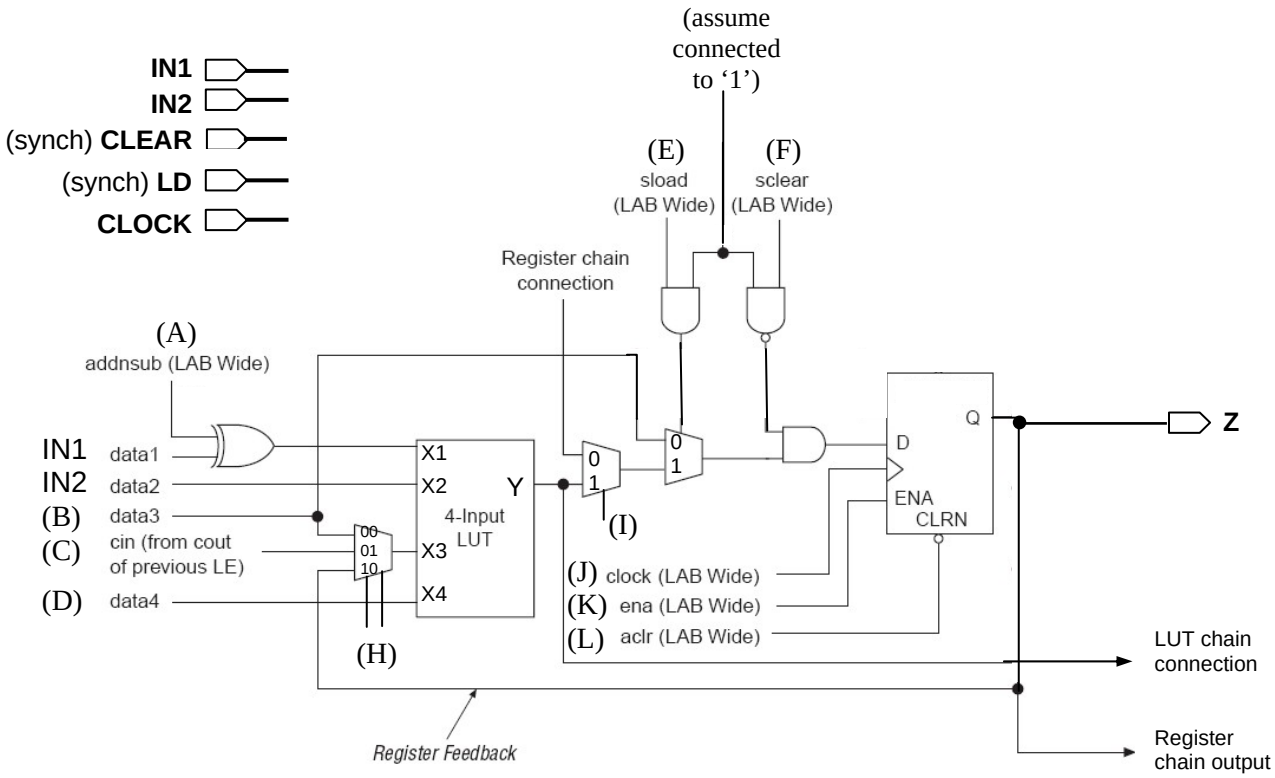


(a) Complete the following VHDL specification for it.

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;

ENTITY Test2LEprob IS
    PORT ( IN1, IN2, CLEAR, LD , CLOCK: IN  STD_LOGIC ; -- synchronous CLEAR, LD
                                    Z   : OUT STD_LOGIC ) ;
END Test2LEprob ;

ARCHITECTURE LEArch OF Test2LEprob IS

SIGNAL

BEGIN
    PROCESS
    BEGIN -- for maximum credit, all your code should be inside this PROCESS statement.




    END PROCESS;
END LEArch ;
```

15 pts.

4. You are to "program" the logic element (LE) of an Altera Cyclone II device to implement the circuit from the previous page (from Problem 3). In other words, for each "location" (A through L), specify what should be connected to it. It can be 0, 1, X ("don't care"), or a signal name (like CLEAR). If it is "don't care", you must put X (not 0 or 1). Also specify the contents of the LUT (look up table).



Put your answers here. Each signal should be connected to 0, 1, X ("don't care"), NC (for not connected), or a signal name. If it is "don't care", you must put X (not 0 or 1).

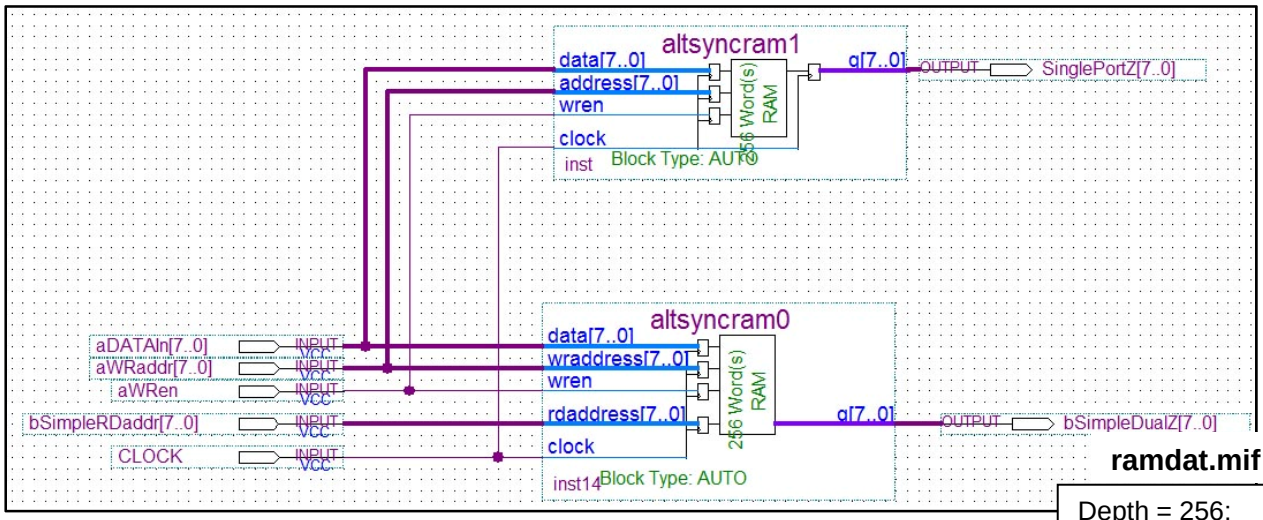
(A)	(E)	(J)
(B)	(F)	(K)
(C)	(H)	(L)
(D)	(I)	

Contents of LUT:

X1	X2	X3	X4	Y
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

18 pts.

5. AltSyncRAM



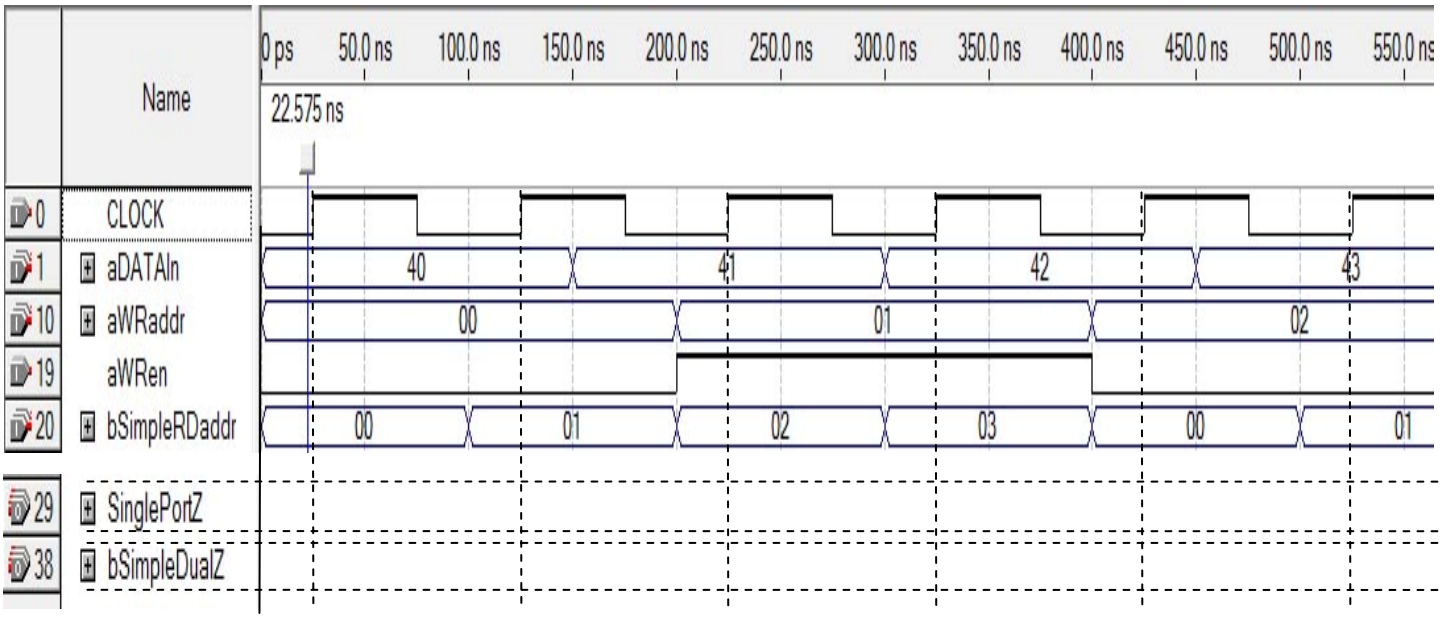
ramdat.mif

```
Depth = 256;
Width = 8;
Address_radix = hex;
Data_radix = hex;
Content
Begin
00 : 70;    07 : 77;
01 : 71;    08 : 78;
02 : 72;    09 : 79;
03 : 73;    0A : 7A;
04 : 74;    0B : 7B;
05 : 75;    0C : 7C;
06 : 76;    etc.
```

Complete the following timing diagram.

Assume all flip-flops are initialized to '0'.

Both RAM's has the same data (ramdat.mif).



Please show delays and put answers for SinglePortZ and bSimpleDualZ and Ydata in hex.

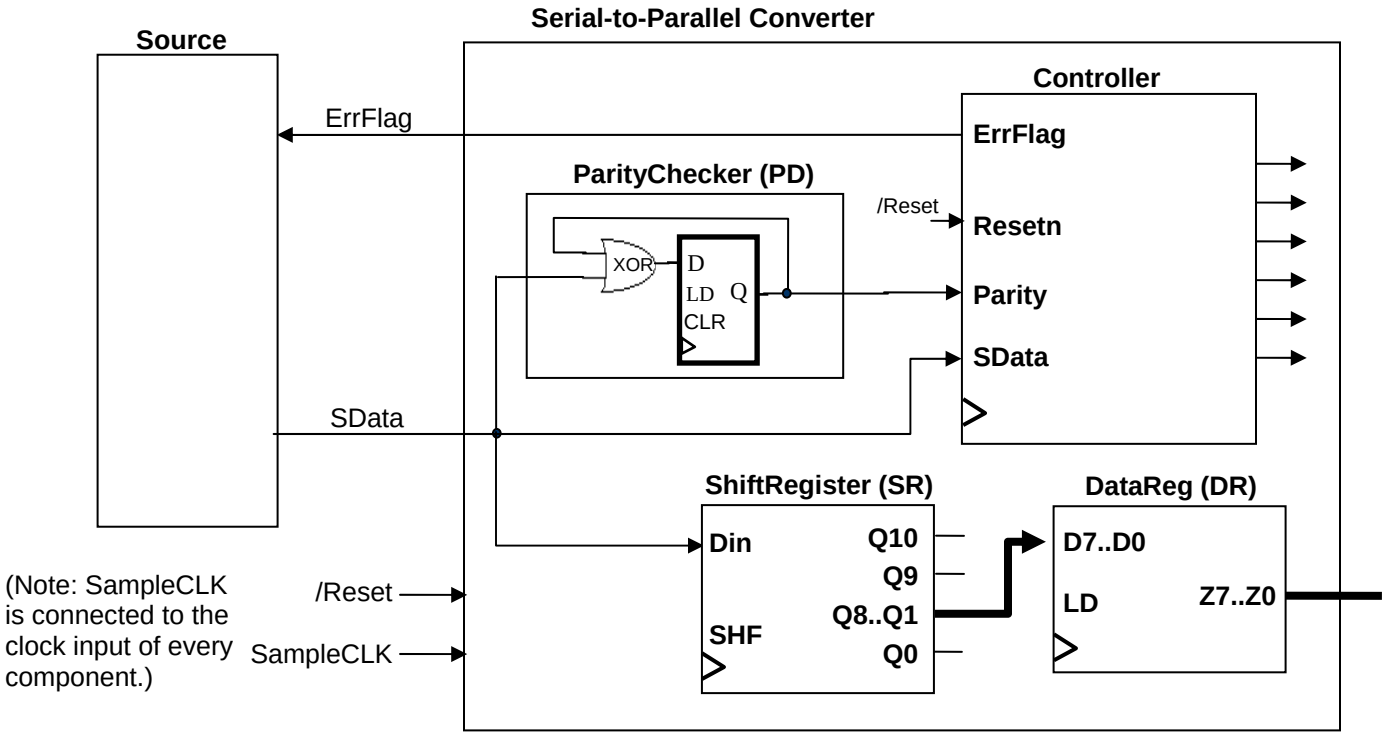
25 pts.

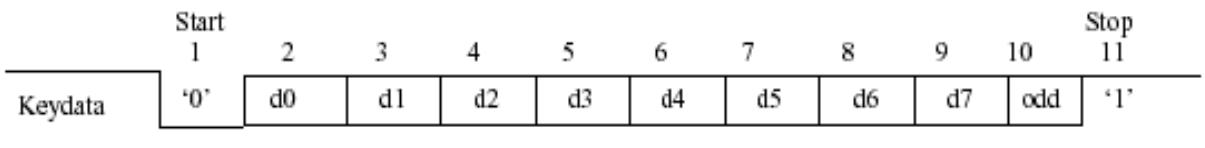
6. Digital design using ASM: serial-to-parallel converter (asynchronous mode)

A serial-to-parallel converter receives a data frame serially, extract the 8-bit data byte, and load the data byte into a data register. In the asynchronous mode, only the data bits (SData in the following figure) are sent. There is no synchronizing clock signal (like keyclk in Lab 6).

So, this problem is very much like the "post-lab" design of the asynchronous mode of serial communication, with the following differences:

- There is only one type of data byte (vs. 3 types in Lab 6: i.e., there are no make code, $F0, break code).
- A ParityChecker (PD) is used to make sure the data frame has "odd" parity (i.e., the number of 1's in the data frame is an odd number).
- For each data byte, if it is correct (i.e., the parity is odd), it will be loaded in DataReg (DR). If it is not correct, the data byte is not loaded and ErrFlag is set to '1' until a start bit has been detected again.
- The SampleCLK frequency is 16 times the data bit rate.
- /Reset, when '0', will reset the Controller.





Note: SampleCLK has the frequency to sample each data bit 16 times.

6. (continued)

(a) You will need some counters. Draw the block diagrams of the counters that you need. For each, specify its inputs and outputs. You don't have to design the "inside" (5 pts)

(b) Give the ASM chart for the controller. The best answer gets the most points. In other words, using less states and conditional outputs to improve performance will result in more points. However, it is better to get a correct answer with more than minimum states than to get an erroneous answer with fewer states. (20 pts)

```
ENTITY __entity_name IS
        PORT(__input_name, __input_name        : IN  STD_LOGIC;
             __input_vector_name               : IN  STD_LOGIC_VECTOR(__high downto __low);
             __bidir_name, __bidir_name        : INOUT     STD_LOGIC;
             __output_name, __output_name      : OUT       STD_LOGIC);
END __entity_name;

ARCHITECTURE a OF __entity_name IS
        SIGNAL __signal_name : STD_LOGIC;
        SIGNAL __signal_name : STD_LOGIC;
BEGIN
        -- Process Statement
        -- Concurrent Signal Assignment
        -- Conditional Signal Assignment
        -- Selected Signal Assignment
        -- Component Instantiation Statement
END a;

__instance_name: __component_name  PORT MAP  (__component_port => __connect_port,
                                              __component_port => __connect_port);

WITH __expression SELECT
        __signal <= __expression WHEN __constant_value,
                    __expression WHEN __constant_value,
                    __expression WHEN __constant_value,
                    __expression WHEN __constant_value;

__signal <= __expression WHEN __boolean_expression ELSE
            __expression WHEN __boolean_expression ELSE
            __expression;

IF __expression THEN
   __statement;
   __statement;
ELSIF __expression THEN
   __statement;
   __statement;
ELSE
   __statement;
   __statement;
END IF;

CASE __expression IS
        WHEN __constant_value =>
           __statement;
           __statement;
        WHEN __constant_value =>
           __statement;
           __statement;
        WHEN OTHERS =>
           __statement;
           __statement;
END CASE;

WAIT UNTIL __expression;
```