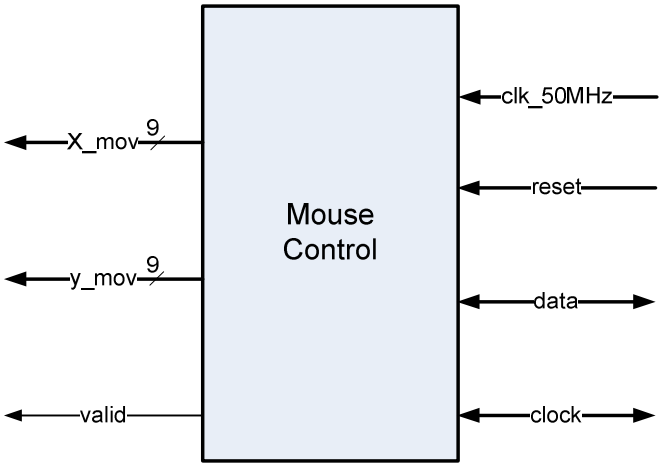
שפת תיאור חומרה VHDL פתרון המבחן

שאלה 1 -60 נקודות

להלן מערכת ספרתית סינכרונית המשמשת כבקר התקן ה-mouse. המערכת כוללת כניסות ויציאות הבאות (איור 1):



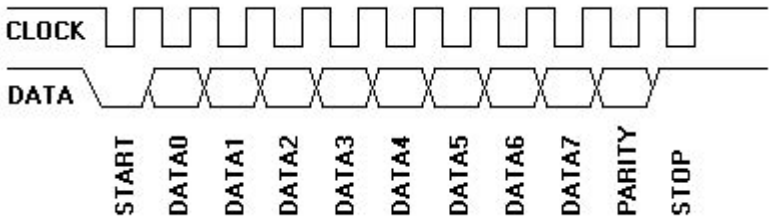
איור 1.

קווי ה-clock וה-data הם קווים דו-כיווניים (פורט מסוג inout) המשמשים את הבקר לאתחול ה-mouse וגם את ה-mouse לשליחת הנתונים לבקר (כמובן שהבקר הוא זה שמנהל את הקווים האלה), x_mov וה- y_mov הם הזזה בצירים אופקי ואנכי בהתאם וה-valid הוא אינדיקציה על קבלת נתון חדש ותקין (כולל בדיקת ה-parity וה-overflow). המערכת היא מערכת סינכרונית הפועלת לפי השעון של 50 MHz.

ה-mouse שולח נתונים ב-3 בייטים הבאים:

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 1	Y overflow	X overflow	Y sign bit	X sign bit	Always 1	Middle Btn	Right Btn	Left Btn
Byte 2	X movement							
Byte 3	Y movement							

כל בייט נשלח לפי פרוטוקול PS2 המתואר להלן (איור 2)



איור 2.

שני הקווים המופיעים באיור 2 הם קווי השעון וה-data אשר מקשרים בן ה-mouse לבן הבקר ופועלים בתזמון הבא: תדר השעון 10-16.7 KHz, זמן מעלית השעון ועד להופעה של ערך חדש בקו data לפחות 5 us , זמן שינוי הערך בקו data עד לירידת השעון לפחות 5 us אך לא יותר מ-25 us. משמעות של כל ביט

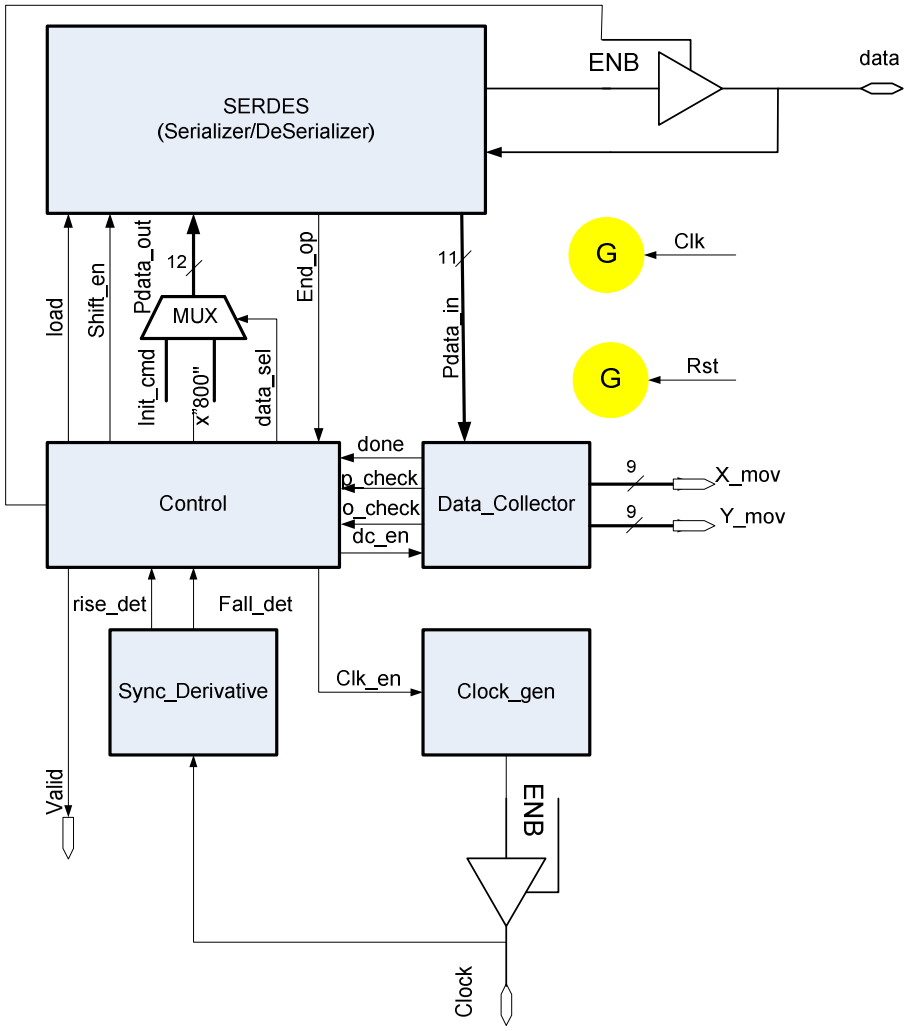
בקו ה-data: Start –תמיד 0, 8 ביט DATA (LSB first), Parity odd (אי זוגי), Stop – תמיד 1. סה"כ אורך החבילה הוא 11 ביט. קווי השעון (clock) וה-data של ה-mouse פעילים אך ורק כאשר יש צורך בשליחת הנתונים, אחרת הם ברמה לוגית 1. הזזה בציר X (X movement) היא ערך של 9 ביט במשלים ל-2 (ביט 6 של בייט הראשון והבייט השני במלואו). הזזה בציר Y (Y movement) היא ערך של 9 ביט במשלים ל-2 (ביט 7 של בייט הראשון והבייט השלישי במלואו). על מנת לאפשר ל-mouse שליחת נתונים, הבקר שולח דרך הקווים clockו-data פקודת אתחול שהיא"x"F6 באופן חד פעמי בתחילת העבודה עם ה-mouse.

ממש יחידת בקרה של מערכת הנ"ל כ-FSM. בפתרון הבעיה יש לבצע את השלבים הבאים:

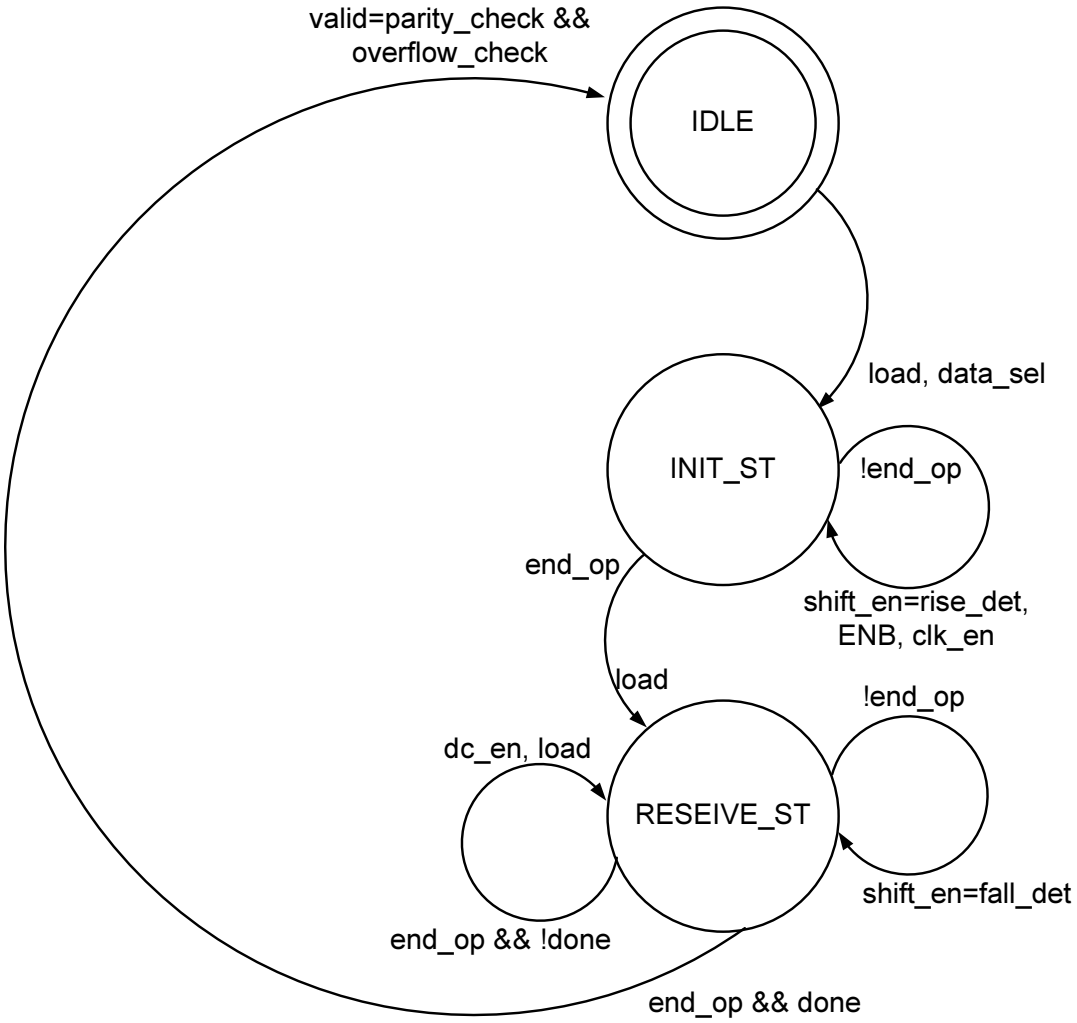
1. בנה ארכיטקטורה של המערכת ע"י חלוקת ליחידת בקרה ויחידות ביצוע ושרטט דיאגראמת בלוקים של המערכת **(10 נקודות)**
2. לשרטט דיאגרמת מצבים של מכונת המצבים בחלק המבקר **(10 נקודות)**
3. לכתוב קוד VHDL מלא (entity, architecture, library) המייצג את יחידת הבקרה **(40 נקודות)**

קוד הנכתב אמור ליהות בר סינתזה!!!

הפתרון:



דיאגראמת מצבים



```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.std_logic_arith.all;
4  use ieee.std_logic_unsigned.all;
5
6  entity mouse_ctrl is
7     port(
8        clk_50    : in  std_logic;
9        rst       : in  std_logic;
10       p_check   : in  std_logic;
11       o_check   : in  std_logic;
12       end_op    : in  std_logic;
13       done      : in  std_logic;
14       rise_det  : in  std_logic;
15       fall_det  : in  std_logic;
16       load      : out std_logic;
17       data_sel  : out std_logic;
18       shift_en  : out std_logic;
19       enb       : out std_logic;
20       clk_en    : out std_logic;
21       valid     : out std_logic;
22       dc_en     : out std_logic
23       );
24 end entity mouse_ctrl;
25
26 architecture arc_mouse_ctrl of mouse_ctrl is
27    type   FSM_ST is (IDLE, INIT_ST, RESEIVE_ST);
28    signal curr_st : FSM_ST;
29 begin
30    fsm : process (clk_50, rst) is
31    begin
32       if (rst = '0') then
33          curr_st  <= IDLE;
34          load     <= '0';
35          data_sel <= '0';
36          shift_en <= '0';
37          enb      <= '0';
38          clk_en   <= '0';
39          valid    <= '0';
40          dc_en    <= '0';
41       elsif rising_edge(clk_50) then
42          load     <= '0';
43          data_sel <= '0';
44          shift_en <= '0';
45          enb      <= '0';
46          clk_en   <= '0';
47          valid    <= '0';
48          dc_en    <= '0';
49          case curr_st is
50             when IDLE => load     <= '1';
51                          data_sel <= '1';
52             when INIT_ST => if (end_op = '1') then
53                                curr_st <= RESEIVE_ST;
54                                load    <= '1';
55                             else
56                                shift_en <= rise_det;
57                                enb      <= '1';
58                                clk_en   <= '1';
59                             end if;
60             when RESEIVE_ST => if (end_op = '0') then
61                                   shift_en <= fall_det;
62                                elsif(done = '0') then
63                                   load  <= '1';
64                                   dc_en <= '1';
65                                else
66                                   valid   <= p_check and o_check;
67                                   curr_st <= IDLE;
68                                end if;
69             when others => curr_st <= IDLE;
70          end case;
71       end if;
72    end process fsm;
73 end architecture arc_mouse_ctrl;
74
```

שאלה 2 -20 נקודות

לכתוב פונקציה רקורסיבית המקבלת כארגומנט bit_vector ובודקת האם ווקטור המתקבל הוא סימטרי ולא. פונקציה מחזירה Boolean, true במידה והווקטור סימטרי אחרת false. לדוגמה ווקטורים "011000110" ו-"10111101" הינם סימטריים והווקטור "110111001" אינו סימטרי. **יש לכתוב פונקציה ברת סינתזה.**

```
function is_symmetric (d : std_logic_vector) return boolean is
   alias t : std_logic_vector(0 to (d'length-1)) is d;
   variable res : boolean;
begin
   if (d'length=1) then
      res:=true;
   elsif (d'length=2) then
      res:=(t(t'left)=t(t'right));
   else
      res:=(t(t'left)=t(t'right)) and is_synmmetric(t((t'left+1) to (t'right-1)));
   end if;
   return res;
end function is_symmetric;
```

שאלה 3 -20 נקודות

נתונה מערכת צירופית המקבלת בשתי כניסות שלה a ו -b ערכים של 8 ביט המוצגים בקוד One-Hot (קוד הכולל ביט אחד בלבד ברמה לוגית '1', שאר הביטים נמצאים ב-'0'). לערכי One-Hot של הכניסות מיוחסים ערכים עשרוניים לפי טבלה הבאה:

ערך בקוד One-Hot	ערך עשרוני
00000001	0
00000010	1
00000100	2
00001000	3
00010000	4
00100000	5
01000000	6
10000000	7

מתבצע חיבור אריתמטי בין שני ערכים העשרוניים המיוחסים לערכי One-Hot של כניסות, ותוצאת החיבור מוצגת במוצא C של 15 ביט בקוד One-Hot לפי דוגמה הבאה: נתונים "A="00001000 ו "B="01000000, A פורט מזוהה עם ערך עשרוני 3, ופורט -B מזוהה עם ערך עשרוני 6. תוצאת החיבור האריתמטי בניהם היא 9. התוצאה תוצג בקוד One-Hot באופן הבא: "C="000001000000000. במידה ולפחות אחד מהכניסות לא מוצגת בקוד One-Hot , אזי במוצא C מופיע "111111111111111".

כתוב תיאור של מערכת הנ"ל בVHDL באופן מלא,הכולל הצהרת ENTITY ו ARCHITECTURE

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.std_logic_arith.all;
4  use ieee.std_logic_unsigned.all;
5  entity one_hot_add is
6     port (a, b : in  std_logic_vector(7 downto 0);
7           c    : out std_logic_vector(14 downto 0));
8  end entity one_hot_add;
9  architecture arc_one_hot_add of one_hot_add is
10 begin
11    process (a, b) is
12       variable ad, bd, absum : std_logic_vector(3 downto 0);
13       variable temp          : std_logic_vector(15 downto 0);
14       variable error         : std_logic;
15    begin
16       case conv_integer(a) is
17          when 1      => ad := x"0";
18          when 2      => ad := x"1";
19          when 4      => ad := x"2";
20          when 8      => ad := x"3";
21          when 16     => ad := x"4";
22          when 32     => ad := x"5";
23          when 64     => ad := x"6";
24          when 128    => ad := x"7";
25          when others => ad := x"f";
26       end case;
27       case conv_integer(b) is
28          when 1      => bd := x"0";
29          when 2      => bd := x"1";
30          when 4      => bd := x"2";
31          when 8      => bd := x"3";
32          when 16     => bd := x"4";
33          when 32     => bd := x"5";
34          when 64     => bd := x"6";
35          when 128    => bd := x"7";
36          when others => bd := x"f";
37       end case;
38       absum:=('0' & ad(2 downto 0)) + ('0' & bd(2 downto 0));
39       error:=((ad(3) or bd(3));
40       temp :=(others => error);
41       temp(conv_integer(absum)):= '1';
42       c <= temp(c'range);
43    end process;
44 end architecture arc_one_hot_add;
45
46
47
```