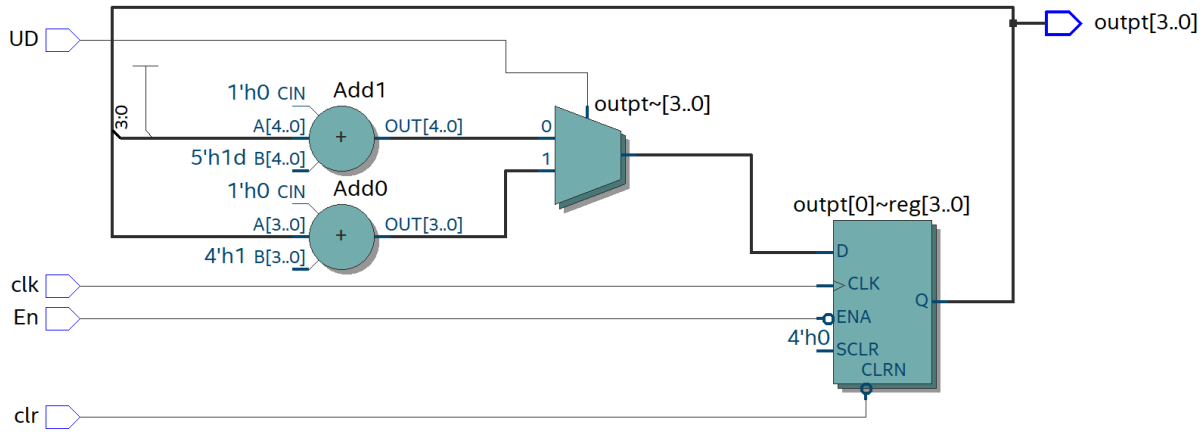
מעבדה לVHDL – מונים

חלק 1 – מונה בינארי/BCD

כתבנו תוכנית עבור המונה בינארי:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity BinaryCounter is
port(En,clk,UD,clr: in std_logic;
outpt: buffer std_logic_vector(3 downto 0):="0000");
end;

architecture one of BinaryCounter is
begin
process(En,clk)
begin
if (clr = '0') then outpt <= (others => '0');
elsif (En = '0') then
   if (clk 'event and clk = '1') then
      if (ud = '1') then -- UP
         outpt <= outpt + 1;
      else -- Down
         outpt <= outpt - 1;
      end if;
end if;
end if;
end process;
end;
```

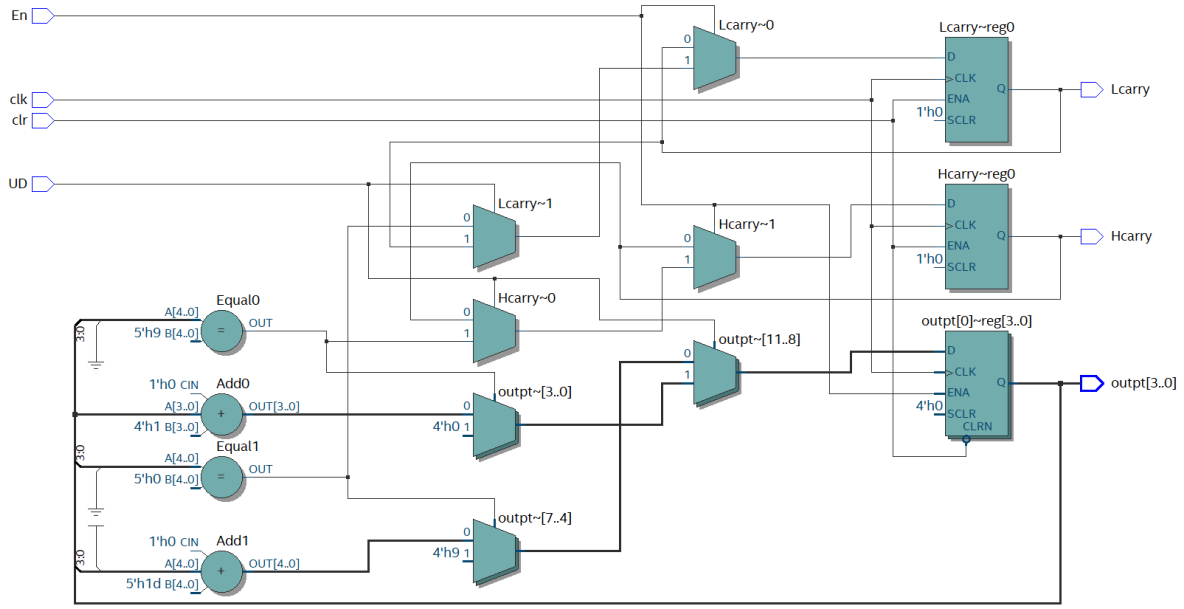
:RTL



לאחר מכן כתבנו תוכנית עבור מונה BCD:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity BCDCounter is
port(
clr,En,clk,UD: in std_logic;
Lcarry,Hcarry: out std_logic:='0';
outpt: buffer std_logic_vector(3 downto 0):="0000");
end;

architecture one of BCDCounter is

begin
process(En,clk)
begin
if (clr = '0') then outpt <= (others => '0');
elsif (En = '1') then
   if (clk 'event and clk = '1') then
      if (ud = '1') then -- UP
         if (outpt = 9) then
            Hcarry <= '1';
            outpt <= (others=>'0');
         else
            Hcarry <= '0';
            outpt <= outpt + 1;
         end if;
      else -- Down
         if (outpt = 0) then
            outpt <= "1001";
            Lcarry <= '1';
         else
            Lcarry <= '0';
            outpt <= outpt - 1;
         end if;
      end if;
   end if;
end if;
end process;
end;
```

:RTL



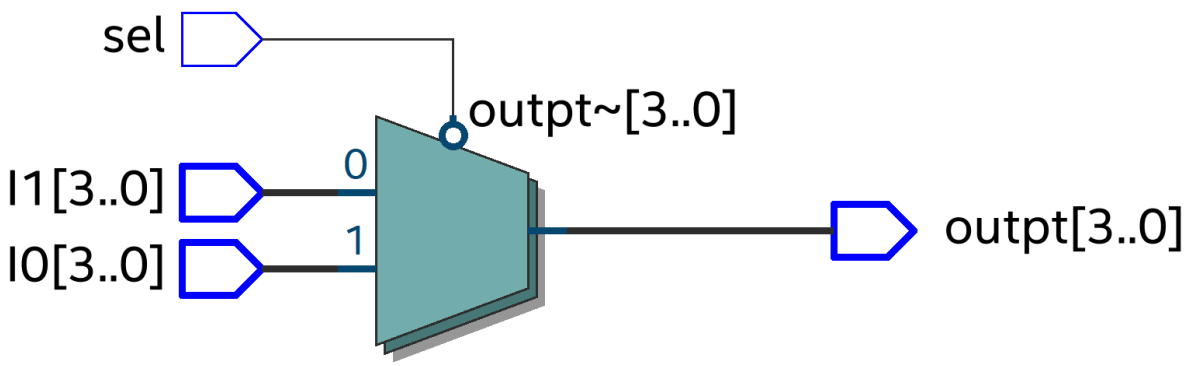
תכננו את המונה BCD בצורה כזו כדי שנוכל לשרשר את מונה הזה אחד אחרי השני בחלק השני של הניסוי.
לשם כך הגדרנו למונה את היציאות Lcarry, Hcarry שכאשר המונה מסיים לספור מעלה מתקבל פולס בHcarry וכשהמונה מסיים לספור מטה מתקבל פולס בLcarry.

לאחר כתיבת שני התוכניות עבור המונים כתבנו תוכנית עבור mux אשר בוחר עם איזה מונה לעבוד (התפקיד של המרבב הזה הוא פשוט כדי להעביר את המונה שאנו בוחרים לעבוד איתו למקודד של התצוגה).

התוכנית של המרבב:

```
library ieee;
use ieee.std_logic_1164.all;
entity mux is port
(I1,I0:std_logic_vector (3 downto 0);
sel:std_logic;
outpt: out std_logic_vector (3 downto 0));
end;
architecture one of mux is
begin
with sel select
outpt <= I0 when '0',
         I1 when '1';
end;
```

RTL:

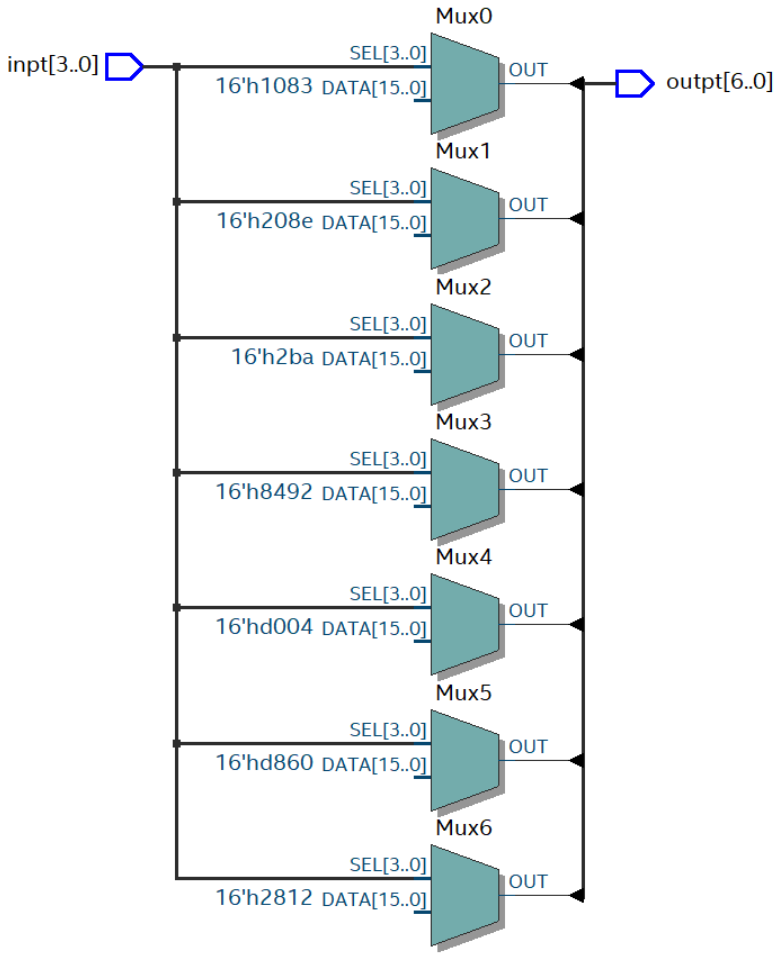


לאחר שכתבנו את כל הרכיבים הגענו לכתיבת התוכנית למפענח של התצוגת 7 המקטעים.

תוכנית:

```
library ieee;
use ieee.std_logic_1164.all;

entity Decoder7Segment is
port(
inpt:  in  std_logic_vector(3 downto 0);
outpt: out std_logic_vector(6 downto 0));
end;

architecture one of Decoder7Segment is
begin

with inpt select
outpt <= "1000000" when "0000", -- '0'
         "1111001" when "0001", -- '1'
         "0100100" when "0010", -- '2'
         "0110000" when "0011", -- '3'
         "0011001" when "0100", -- '4'
         "0010010" when "0101", -- '5'
         "0000010" when "0110", -- '6'
         "1111000" when "0111", -- '7'
         "0000000" when "1000", -- '8'
         "0010000" when "1001", -- '9'
         "0001000" when "1010", -- 'A'
         "0000011" when "1011", -- 'B'
         "1000110" when "1100", -- 'C'
         "0100001" when "1101", -- 'D'
         "0000110" when "1110", -- 'E'
         "0001110" when "1111"; -- 'F' (OTHERS)

end;
```

RTL:

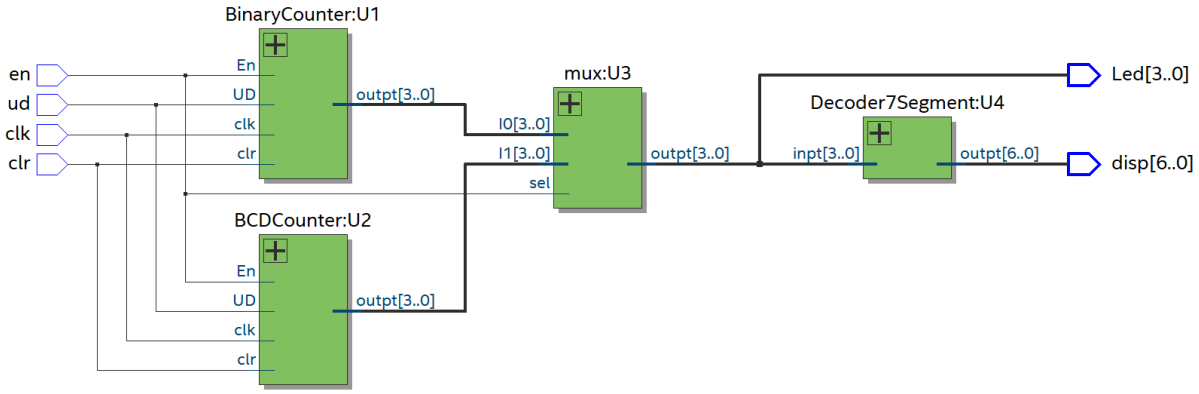


כעת הגענו לשלב הסופי כתיבת תוכנית שמשלבת את כל היחידות הנ"ל (כ-component) בכדי ליצור מונה שיכול לספור מעלה/מטה בקידוד בינארי/BCD.

התוכנית:

```
library ieee;
use ieee.std_logic_1164.all;
entity countBCDBin is
port(
clr,ud,clk,en: in std_logic;
Led: out std_logic_vector(3 downto 0);
disp: out std_logic_vector(6 downto 0));
end;

architecture one of countBCDBin is

-- / Components \ --
component BinaryCounter
port(En,clk,UD,clr: in std_logic;
outpt: buffer std_logic_vector(3 downto 0):="0000");
end component;

component BCDCounter
port(clr,En,clk,UD: in std_logic;
Lcarry,Hcarry: out std_logic:='0';
outpt: buffer std_logic_vector(3 downto 0):="0000");
end component;

component mux
port(I1,I0:std_logic_vector (3 downto 0);
sel:std_logic;
outpt:out std_logic_vector (3 downto 0));
end component;

component Decoder7Segment
port(inpt:  in  std_logic_vector(3 downto 0);
outpt: out std_logic_vector(6 downto 0));
end component;
-- / Components \ --

signal s1,s2,s3: std_logic_vector(3 downto 0);
begin

U1: BinaryCounter   port map (en,clk,ud,clr,s1);
U2: BCDCounter      port map (clr,en,clk,ud,open,open,s2);
U3: mux             port map (s2,s1,en,s3);
U4: Decoder7Segment port map (s3,disp);
Led <= s3;
end;
```

RTL:



חלק ב'

כעת היה עלינו לכתוב מונה שסופר מאפס עד 999.
נעזרנו בקוד של המונה BCD כאשר שרשרנו 3 מונים מסוג זה אחד אחרי השני כאשר הclk של המונה הראשון הוא הclk של המערכת, הclk של המונה השני הוא נבחר על ידי ההדק UD להיות Hcarry או Lcarry כך שכאשר המונה הראשון מסיים לספור המונה השני מקבל פולס אחד בCLK וכך הלאה גם למונה השלישי.

התוכנית:

```
library ieee;
use ieee.std_logic_1164.all;

entity DecCounter is
port(
clk,UD,clr: in std_logic;
outpt: buffer std_logic_vector(11 downto 0);
disp1,disp2,disp3: out std_logic_vector(6 downto 0));
end;

architecture one of DecCounter is

-- / Components \ --
component BCDCounter
port(
En,clk,UD: in std_logic;
Lcarry,Hcarry: out std_logic:='0';
outpt: buffer std_logic_vector(3 downto 0):="0000";
clr: in std_logic);
end component;

component Decoder7Segment
port(inpt:  in  std_logic_vector(3 downto 0);
outpt: out std_logic_vector(6 downto 0));
end component;
-- / Components \ --

signal internClk0,internclk1,sHcarry0,sLcarry0,sHcarry1,sLcarry1: std_logic;

begin

internClk0 <= sHcarry0 when (UD = '1') else sLcarry0;
internClk1 <= sHcarry1 when (UD = '1') else sLcarry1;

Count1: BCDCounter port map ('1', clk,        UD, sLcarry0, sHcarry0, outpt(3  downto 0), clr);
Count2: BCDCounter port map ('1', internclk0, UD, sLcarry1, sHcarry1, outpt(7  downto 4), clr);
Count3: BCDCounter port map ('1', internclk1, UD, open,     open,     outpt(11 downto 8), clr);

Dispx: Decoder7Segment port map (outpt(3  downto 0), disp1); -- Yehidot
Dispy: Decoder7Segment port map (outpt(7  downto 4), disp2); -- asarot
Dispz: Decoder7Segment port map (outpt(11 downto 8), disp3); -- meot

end;
```

מגיש: אריאל אוחיון 206912354

:RTL