Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

2η Σειρά Ασκήσεων

Μάθημα: Ψηφιακά Συστήματα VLSI

Εξάμηνο: 8ο

Ονοματεπώνυμα: Αλεξοπούλου Γεωργία, Γκενάκου Ζωή

Θέμα 1: Half Adder

Στο πρώτο ζητούμενο της άσκησης πρέπει να υλοποιήσουμε έναν Ημιαθροιστή (Half Adder - HA) σε περιγραφή ροής δεδομένων (Dataflow).

Ο πίνακας αληθείας του Ημιαθροιστή είναι ο παρακάτω:

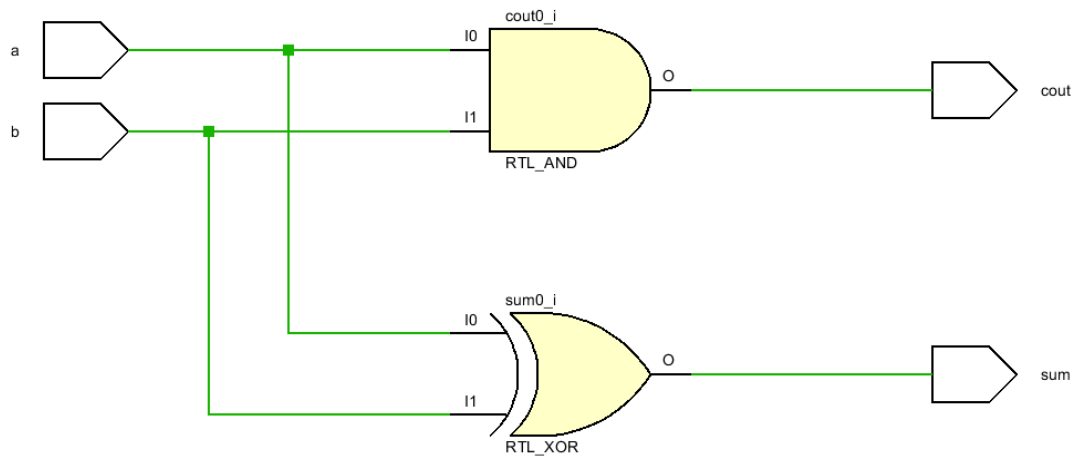
Inputs		**Outputs**	
A	**B**	**Sum**	**Carry Out**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Παρατηρούμε ότι ο Ημιαθροιστής μπορεί να υλοποιηθεί με μία πύλη XOR και μία πύλη AND.

Με αυτή την λογική δημιουργούμε τον ακόλουθο κώδικα στο Vivado.

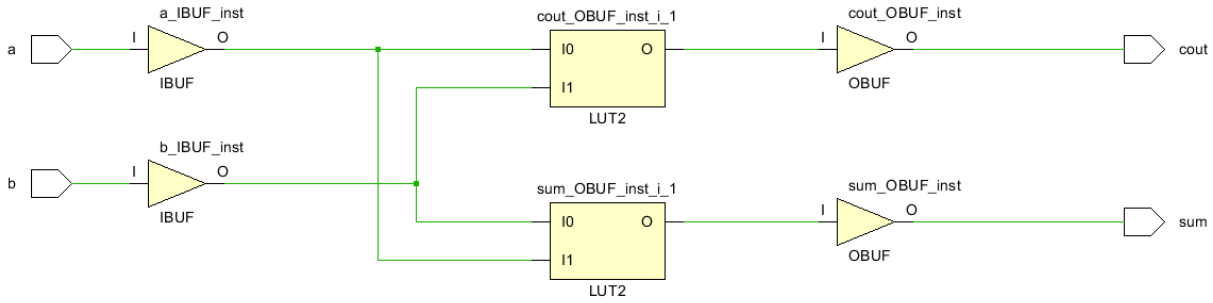
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity half_adder_df is
    Port ( a : in STD_LOGIC;
           b : in STD_LOGIC;
           cout : out STD_LOGIC;
           sum : out STD_LOGIC);
end half_adder_df;

architecture Dataflow of half_adder_df is

begin

    sum <= (a xor b);
    cout <= (a and b);

end Dataflow;
```

Το δομικό διάγραμμα (RTL Schematic) της αρχιτεκτονικής κυκλώματος φαίνεται παρακάτω:



Παρατηρούμε όντως ότι ο ημιαθροιστής υλοποιείται με μία πύλη AND και μία πύλη XOR.

Από τον Synthesizer προκύπτει το παρακάτω schematic:

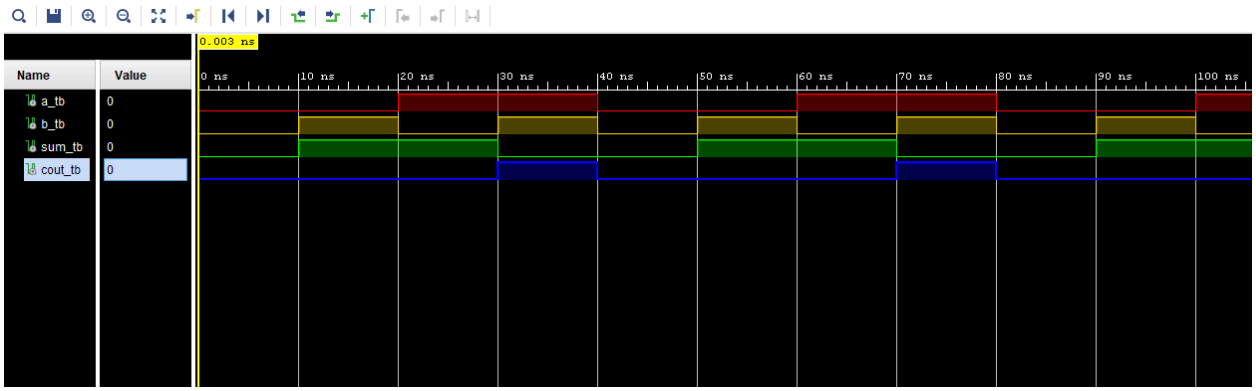


Δημιουργούμε TestBench για να ελέγξουμε την ορθή λειτουργία του κυκλώματος:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity half_adder_tb is
--  Port ( );
end half_adder_tb;

architecture Dataflow of half_adder_tb is

    component half_adder_df is
        Port ( a : in STD_LOGIC;
               b : in STD_LOGIC;
               cout : out STD_LOGIC;
               sum : out STD_LOGIC);
    end component;

    signal a_tb, b_tb, cout_tb, sum_tb : std_logic;

begin
    dut : half_adder_df
        port map(
```

```
        a => a_tb,
        b => b_tb,
        cout => cout_tb,
        sum => sum_tb
    );

    simulation : process
    begin
        a_tb <= '0';
        b_tb <= '0';
        wait for 10 ns;

        a_tb <= '0';
        b_tb <= '1';
        wait for 10 ns;

        a_tb <= '1';
        b_tb <= '0';
        wait for 10 ns;

        a_tb <= '1';
        b_tb <= '1';
        wait for 10 ns;

    end process simulation;

end Dataflow;
```

Η προσομοίωση επιβεβαιώνει την ορθή λειτουργία του κυκλώματος:



Για να βρούμε το κρίσιμο μονοπάτι (critical path) του κυκλώματος, τρέχουμε το Synthesis → Report Timing Summary → Unconstrained Paths → NONE to NONE → Setup

Name	Slack ^1	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Exception	Clock Uncertainty
Path 1	∞	3	4	2	b	sum	5.377	3.778	1.599	∞	input port clock			0.000
Path 2	∞	3	4	2	b	cout	5.351	3.752	1.599	∞	input port clock			0.000

Από τα παραπάνω προκύπτει ότι το critical path είναι εκείνο από την είσοδο B στο άθροισμα Sum με συνολική καθυστέρηση 5.377ns.

Αναλυτικότερα τρέχοντας την εντολή “report_timing_summary -report_unconstrained” στο τερματικό και λαμβάνουμε:

```
Max Delay Paths
--------------------------------------------------------------------------------------
Slack:                    inf
  Source:                 b
                            (input port)
  Destination:            sum
                            (output port)
  Path Group:             (none)
  Path Type:              Max at Slow Process Corner
  Data Path Delay:        5.377ns  (logic 3.778ns (70.255%)  route 1.599ns (29.745%))
  Logic Levels:           3  (IBUF=1 LUT2=1 OBUF=1)

    Location             Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                                                      0.000     0.000 r  b (IN)
                       net (fo=0)                     0.000     0.000    b
                       IBUF (Prop_ibuf_I_O)           0.982     0.982 r  b_IBUF_inst/O
                       net (fo=2, unplaced)           0.800     1.782    b_IBUF
                       LUT2 (Prop_lut2_I0_O)          0.150     1.932 r  sum_OBUF_inst_i_1/O
                       net (fo=1, unplaced)           0.800     2.732    sum_OBUF
                       OBUF (Prop_obuf_I_O)           2.645     5.377 r  sum_OBUF_inst/O
                       net (fo=0)                     0.000     5.377    sum
                                                                      r  sum (OUT)

Min Delay Paths
--------------------------------------------------------------------------------------
Slack:                    inf
  Source:                 b
                            (input port)
  Destination:            cout
                            (output port)
  Path Group:             (none)
  Path Type:              Min at Fast Process Corner
  Data Path Delay:        2.092ns  (logic 1.418ns (67.775%)  route 0.674ns (32.225%))
  Logic Levels:           3  (IBUF=1 LUT2=1 OBUF=1)

    Location             Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                                                      0.000     0.000 r  b (IN)
                       net (fo=0)                     0.000     0.000    b
                       IBUF (Prop_ibuf_I_O)           0.211     0.211 r  b_IBUF_inst/O
                       net (fo=2, unplaced)           0.337     0.548    b_IBUF
                       LUT2 (Prop_lut2_I1_O)          0.045     0.593 r  cout_OBUF_inst_i_1/O
                       net (fo=1, unplaced)           0.337     0.930    cout_OBUF
                       OBUF (Prop_obuf_I_O)           1.162     2.092 r  cout_OBUF_inst/O
                       net (fo=0)                     0.000     2.092    cout
                                                                      r  cout (OUT)
  -------------------------------------------------------------------    -------------------
```

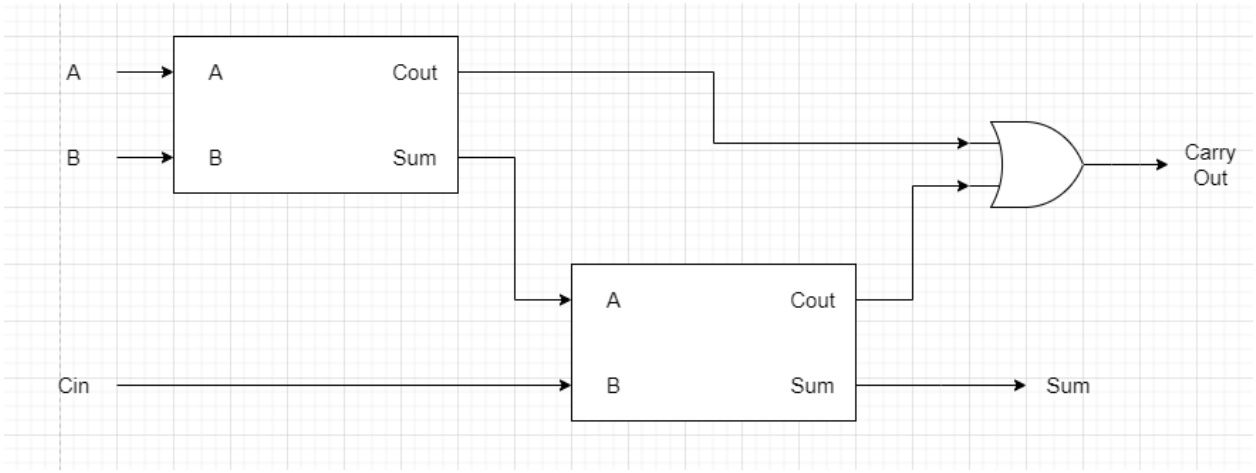
Θέμα 2: Full Adder

Στο δεύτερο ζήτημα μας ζητείται να υλοποιήσουμε έναν πλήρη αθροιστή (Full Adder - FA) με περιγραφή δομής (Structural), βασιζόμενοι στην δομική μονάδα του προηγούμενου ερωτήματος.

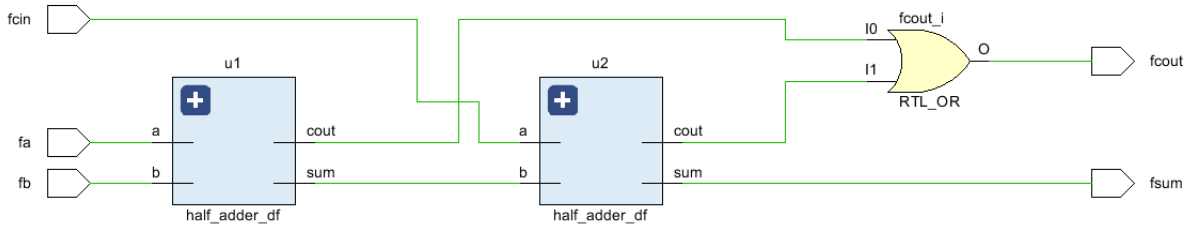
Ο πίνακας αληθείας του Full Adder φαίνεται παρακάτω:

Inputs			**Outputs**	
A	**B**	**Carry In**	**Sum**	**Carry Out**
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

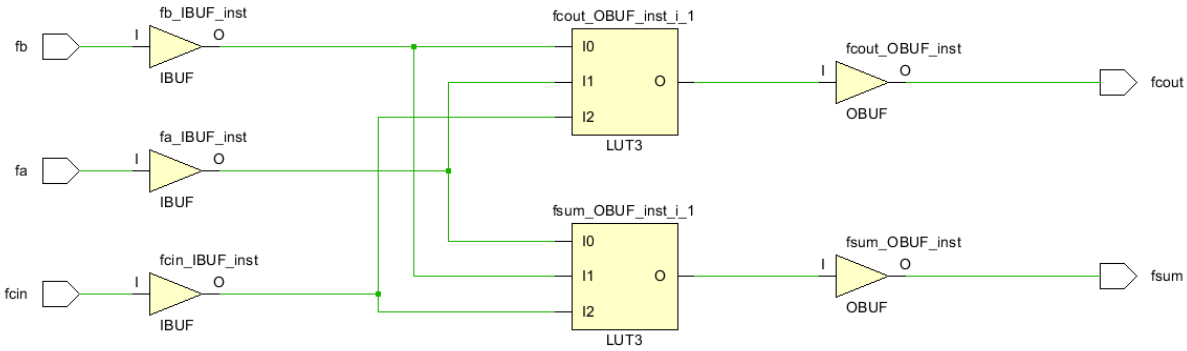
Μπορούμε το κύκλωμα του Full Adder να το φτιάξουμε χρησιμοποιώντας 2 Half Adders και μια πύλη OR ως εξής:



Το παραπάνω περιμένουμε να δούμε και στο RTL:



Από τον Synthesizer προκύπτει το παρακάτω schematic:



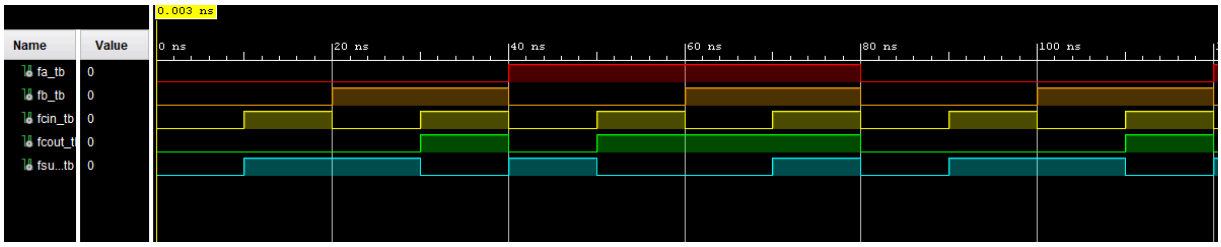
Δημιουργούμε TestBench για να ελέγξουμε την ορθή λειτουργία του κυκλώματος:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity full_adder_tb_str is
--  Port ( );
end full_adder_tb_str;

architecture Structural of full_adder_tb_str is
    component full_adder_str is
        Port ( fa : in STD_LOGIC;
```

```
            fb : in STD_LOGIC;
            fcin : in STD_LOGIC;
            fcout : out STD_LOGIC;
            fsum : out STD_LOGIC);
    end component;


    signal fa_tb, fb_tb, fcin_tb, fcout_tb, fsum_tb : std_logic;

begin
    dut : full_adder_str
        port map (
            fa => fa_tb,
            fb => fb_tb,
            fcin => fcin_tb,
            fcout => fcout_tb,
            fsum => fsum_tb
        );

        simulation : process
        begin
            fa_tb <= '0';
            fb_tb <= '0';
            fcin_tb <= '0';
            wait for 10 ns;

            fa_tb <= '0';
            fb_tb <= '0';
            fcin_tb <= '1';
            wait for 10 ns;

            fa_tb <= '0';
            fb_tb <= '1';
            fcin_tb <= '0';
            wait for 10 ns;

            fa_tb <= '0';
            fb_tb <= '1';
            fcin_tb <= '1';
```

```
        wait for 10 ns;

        fa_tb <= '1';
        fb_tb <= '0';
        fcin_tb <= '0';
        wait for 10 ns;

        fa_tb <= '1';
        fb_tb <= '0';
        fcin_tb <= '1';
        wait for 10 ns;

        fa_tb <= '1';
        fb_tb <= '1';
        fcin_tb <= '0';
        wait for 10 ns;

        fa_tb <= '1';
        fb_tb <= '1';
        fcin_tb <= '1';
        wait for 10 ns;

    end process simulation;

end Structural;
```

Η προσομοίωση επιβεβαιώνει την ορθή λειτουργία του κυκλώματος:



Για το critical path:

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Exception	Clock Uncertainty
Path 1	∞	3	4	2	fb	fsum	5.377	3.778	1.599	∞	input port clock			0.000
Path 2	∞	3	4	2	fcin	fcout	5.351	3.752	1.599	∞	input port clock			0.000

Από τα παραπάνω προκύπτει ότι το critical path είναι εκείνο από την είσοδο B στο άθροισμα Sum με συνολική καθυστέρηση 5.377ns.

```
Max Delay Paths
--------------------------------------------------------------------------------
Slack:                  inf
  Source:               fb
                          (input port)
  Destination:          fsum
                          (output port)
  Path Group:           (none)
  Path Type:            Max at Slow Process Corner
  Data Path Delay:      5.377ns  (logic 3.778ns (70.255%)  route 1.599ns (29.745%))
  Logic Levels:         3  (IBUF=1 LUT3=1 OBUF=1)

    Location             Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                                                       0.000     0.000 r  fb (IN)
                       net (fo=0)                      0.000     0.000    fb
                       IBUF (Prop_ibuf_I_O)            0.982     0.982 r  fb_IBUF_inst/O
                       net (fo=2, unplaced)            0.800     1.782    fb_IBUF
                       LUT3 (Prop_lut3_I1_O)           0.150     1.932 r  fsum_OBUF_inst_i_1/O
                       net (fo=1, unplaced)            0.800     2.732    fsum_OBUF
                       OBUF (Prop_obuf_I_O)            2.645     5.377 r  fsum_OBUF_inst/O
                       net (fo=0)                      0.000     5.377    fsum
                                                                       r  fsum (OUT)
  -------------------------------------------------------------------    -------------------


Min Delay Paths
--------------------------------------------------------------------------------
Slack:                  inf
  Source:               fa
                          (input port)
  Destination:          fsum
                          (output port)
  Path Group:           (none)
  Path Type:            Min at Fast Process Corner
  Data Path Delay:      2.089ns  (logic 1.415ns (67.729%)  route 0.674ns (32.271%))
  Logic Levels:         3  (IBUF=1 LUT3=1 OBUF=1)

    Location             Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                                                       0.000     0.000 r  fa (IN)
                       net (fo=0)                      0.000     0.000    fa
                       IBUF (Prop_ibuf_I_O)            0.211     0.211 r  fa_IBUF_inst/O
                       net (fo=2, unplaced)            0.337     0.548    fa_IBUF
                       LUT3 (Prop_lut3_I0_O)           0.042     0.590 r  fsum_OBUF_inst_i_1/O
                       net (fo=1, unplaced)            0.337     0.927    fsum_OBUF
                       OBUF (Prop_obuf_I_O)            1.162     2.089 r  fsum_OBUF_inst/O
                       net (fo=0)                      0.000     2.089    fsum
                                                                       r  fsum (OUT)
  -------------------------------------------------------------------    -------------------
```

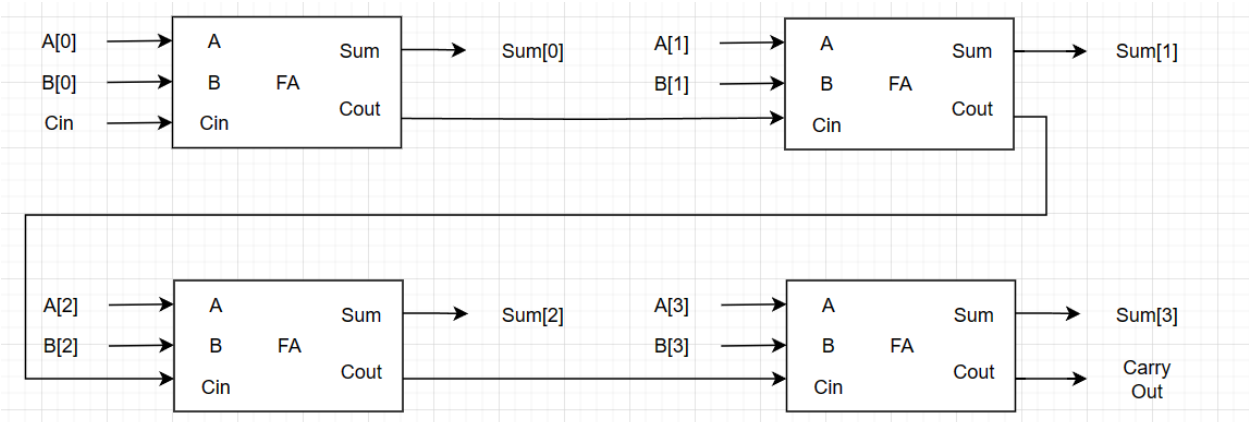
Θέμα 3: 4 bit Parallel Adder

Στο τρίτο ζητούμενο της άσκησης καλούμαστε να υλοποιήσουμε έναν Παράλληλο Αθροιστή των 4 bits (4-bit Parallel Adder - 4-bit PA).

Ο πίνακας αληθείας του 4-bit PA είναι ο εξής:

Inputs			**Outputs**	
A[3:0]	**B[3:0]**	**Carry In**	**Sum[3:0]**	**Carry Out**
0000	0000	0	0000	0
0000	0000	1	0001	0
0000	0001	0	0001	0
0000	0001	1	0010	0
0000	0010	0	0010	0
0000	0010	1	0011	0
0000	0011	0	0011	0
0000	0011	1	0100	0
0000	0100	0	0100	0
0000	0100	1	0101	0
0000	0101	0	0101	0
0000	0101	1	0110	0
κ.ο.κ. …				

Μπορούμε το κύκλωμα του 4-bit Parallel Adder να το υλοποιήσουμε κάνοντας χρήση 4 Full Adders, όπως ορίστηκαν στο *Θέμα 2*:

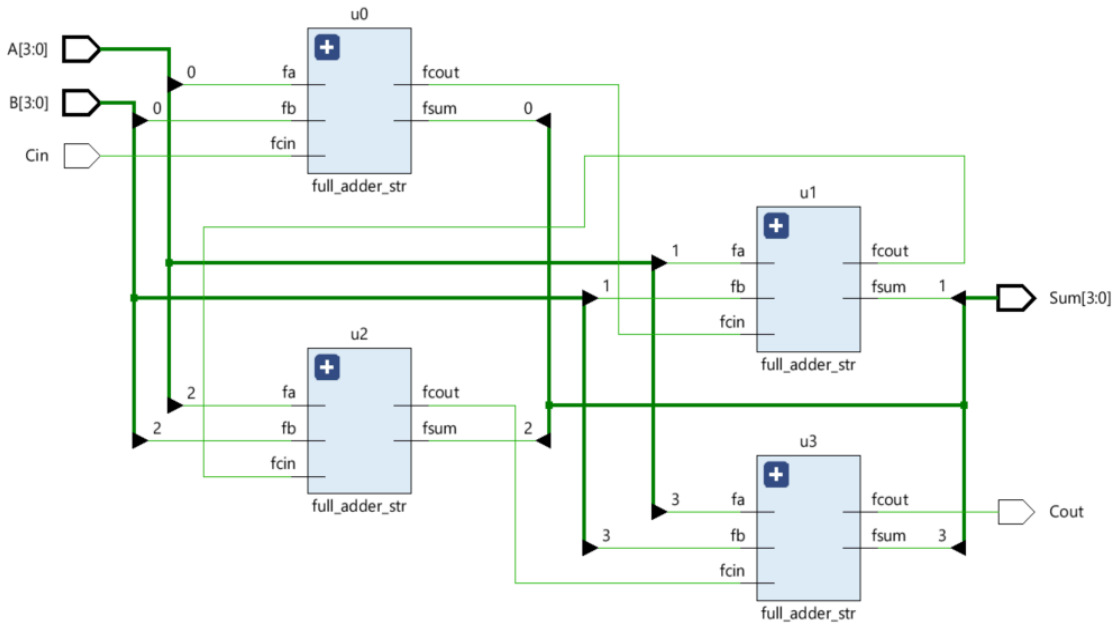


Με αυτή την λογική δημιουργούμε τον ακόλουθο κώδικα στο Vivado.

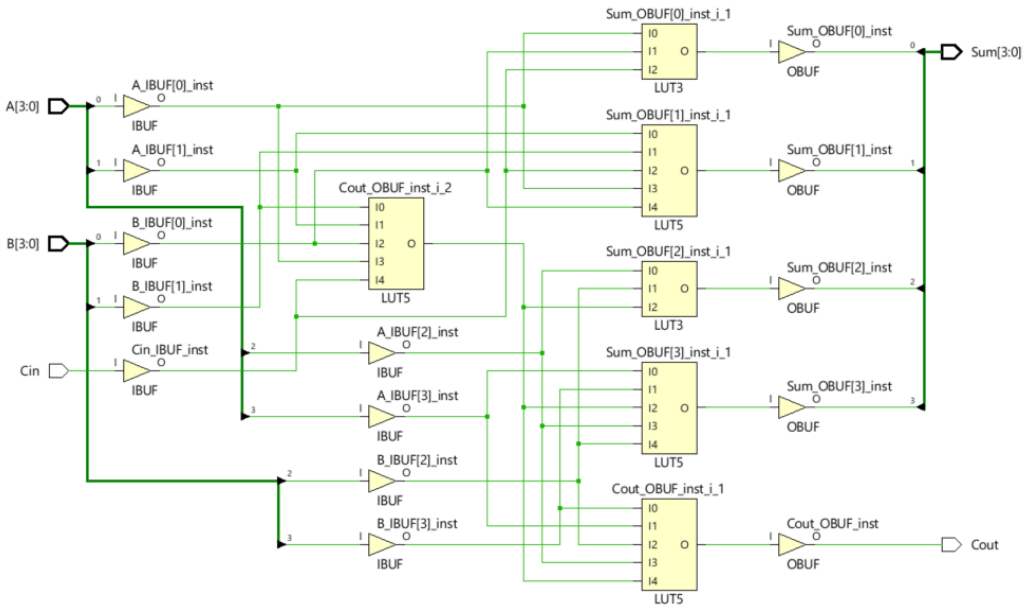
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity full_adder_4_bit is
    Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
           B : in STD_LOGIC_VECTOR (3 downto 0);
           Cin : in STD_LOGIC;
           Cout : out STD_LOGIC;
           Sum : out STD_LOGIC_VECTOR (3 downto 0));
end full_adder_4_bit;

architecture Structural of full_adder_4_bit is
    component full_adder_str is
        Port ( fa : in STD_LOGIC;
               fb : in STD_LOGIC;
               fcin : in STD_LOGIC;
               fcout : out STD_LOGIC;
               fsum : out STD_LOGIC);
    end component;

    signal c0, c1, c2 : std_logic;

begin
u0: full_adder_str port map (fa => A(0), fb => B(0), fcin => Cin,
fcout => c0, fsum => Sum(0));
```

```
u1: full_adder_str port map (fa => A(1), fb => B(1), fcin => c0,
fcout => c1, fsum => Sum(1));
u2: full_adder_str port map (fa => A(2), fb => B(2), fcin => c1,
fcout => c2, fsum => Sum(2));
u3: full_adder_str port map (fa => A(3), fb => B(3), fcin => c2,
fcout => Cout, fsum => Sum(3));


end Structural;
```

Αναμένουμε, λοιπόν, να έχουμε την ίδια εικόνα με παραπάνω και από το RTL Analysis Design:



Από τον Synthesizer προκύπτει το παρακάτω schematic:

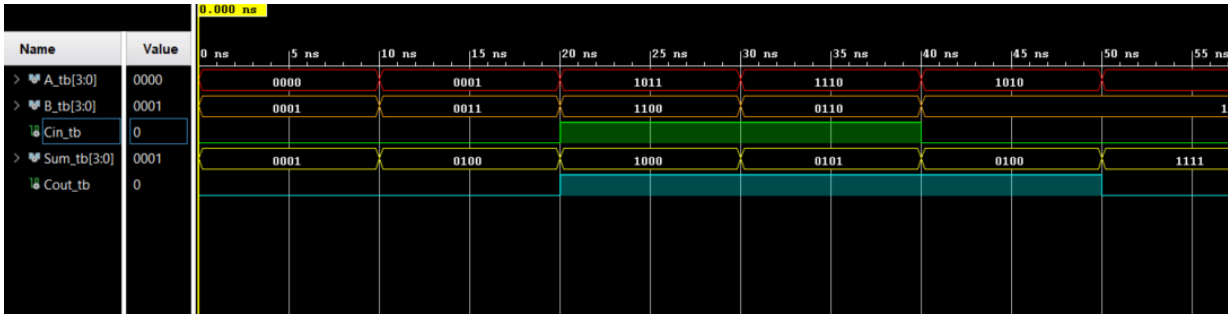


Δημιουργούμε TestBench για να ελέγξουμε την ορθή λειτουργία του κυκλώματος:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;


entity full_adder_4_bit_tb is
--  Port ( );
end full_adder_4_bit_tb;

architecture Structural of full_adder_4_bit_tb is
    component full_adder_4_bit is
        Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
               B : in STD_LOGIC_VECTOR (3 downto 0);
               Cin : in STD_LOGIC;
               Cout : out STD_LOGIC;
               Sum : out STD_LOGIC_VECTOR (3 downto 0));
    end component;


    signal A_tb, B_tb, Sum_tb : std_logic_vector(3 downto 0);
    signal Cin_tb, Cout_tb : std_logic;

begin
    dut : full_adder_4_bit
        port map (
            A => A_tb,
            B => B_tb,
            Cin => Cin_tb,
            Cout => Cout_tb,
            Sum => Sum_tb
        );

        simulation : process
        begin
            A_tb <= "0000";
            B_tb <= "0001";
            Cin_tb <= '0';
```

```
        wait for 10 ns;

        A_tb <= "0001";
        B_tb <= "0011";
        Cin_tb <= '0';
        wait for 10 ns;

        A_tb <= "1011";
        B_tb <= "1100";
        Cin_tb <= '1';
        wait for 10 ns;

        A_tb <= "1110";
        B_tb <= "0110";
        Cin_tb <= '1';
        wait for 10 ns;

        A_tb <= "1010";
        B_tb <= "1010";
        Cin_tb <= '0';
        wait for 10 ns;

        A_tb <= "0101";
        B_tb <= "1010";
        Cin_tb <= '0';
        wait for 10 ns;

        A_tb <= "0101";
        B_tb <= "1010";
        Cin_tb <= '1';
        wait for 10 ns;

        wait;

    end process simulation;

end Structural;
```

Η προσομοίωση επιβεβαιώνει την ορθή λειτουργία του κυκλώματος:



Για το critical path:

Name	Slack ^1	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Exception	Clock Uncertainty
Path 1	∞	4	5	3	Cin	Cout	5.942	3.876	2.066	∞	input port clock			0.000
Path 2	∞	4	5	3	Cin	Sum[2]	5.942	3.876	2.066	∞	input port clock			0.000
Path 3	∞	4	5	3	Cin	Sum[3]	5.936	3.870	2.066	∞	input port clock			0.000
Path 4	∞	3	4	2	B[1]	Sum[1]	5.379	3.780	1.599	∞	input port clock			0.000
Path 5	∞	3	4	3	Cin	Sum[0]	5.351	3.752	1.599	∞	input port clock			0.000

Από τα παραπάνω προκύπτει ότι το critical path είναι 2: αυτό από την είσοδο Cin προς την έξοδο Cout και αυτό από την είσοδο Cin προς την έξοδο Sum[2], με συνολική καθυστέρηση 5.942ns.

```
Max Delay Paths
--------------------------------------------------------------------------------------
Slack:                   inf
  Source:                Cin
                           (input port)
  Destination:           Cout
                           (output port)
  Path Group:            (none)
  Path Type:             Max at Slow Process Corner
  Data Path Delay:       5.942ns  (logic 3.876ns (65.224%)  route 2.066ns (34.776%))
  Logic Levels:          4  (IBUF=1 LUT5=2 OBUF=1)

    Location             Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                                                      0.000     0.000 r  Cin (IN)
                       net (fo=0)                     0.000     0.000    Cin
                       IBUF (Prop_ibuf_I_O)           0.982     0.982 r  Cin_IBUF_inst/O
                       net (fo=3, unplaced)           0.800     1.782    Cin_IBUF
                       LUT5 (Prop_lut5_I4_O)          0.124     1.906 r  Cout_OBUF_inst_i_2/O
                       net (fo=3, unplaced)           0.467     2.373    c1
                       LUT5 (Prop_lut5_I4_O)          0.124     2.497 r  Cout_OBUF_inst_i_1/O
                       net (fo=1, unplaced)           0.800     3.297    Cout_OBUF
                       OBUF (Prop_obuf_I_O)           2.645     5.942 r  Cout_OBUF_inst/O
                       net (fo=0)                     0.000     5.942    Cout
                                                                      r  Cout (OUT)
  -------------------------------------------------------------------    -------------------
```

```
Min Delay Paths
--------------------------------------------------------------------------------
Slack:                    inf
  Source:                 Cin
                            (input port)
  Destination:            Sum[1]
                            (output port)
  Path Group:             (none)
  Path Type:              Min at Fast Process Corner
  Data Path Delay:        2.089ns  (logic 1.415ns (67.729%)  route 0.674ns (32.271%))
  Logic Levels:           3  (IBUF=1 LUT5=1 OBUF=1)

    Location             Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                                                       0.000     0.000 r  Cin (IN)
                         net (fo=0)                    0.000     0.000    Cin
                         IBUF (Prop_ibuf_I_O)          0.211     0.211 r  Cin_IBUF_inst/O
                         net (fo=3, unplaced)          0.337     0.548    Cin_IBUF
                         LUT5 (Prop_lut5_I2_O)         0.042     0.590 r  Sum_OBUF[1]_inst_i_1/O
                         net (fo=1, unplaced)          0.337     0.927    Sum_OBUF[1]
                         OBUF (Prop_obuf_I_O)          1.162     2.089 r  Sum_OBUF[1]_inst/O
                         net (fo=0)                    0.000     2.089    Sum[1]
                                                                       r  Sum[1] (OUT)
  -------------------------------------------------------------------    -------------------
```

Θέμα 4: BCD Full Adder

Στο 4ο ζήτημα, πρέπει να υλοποιήσουμε έναν BCD Πλήρη Αθροιστή (BCD Full Adder - BCD FA) με περιγραφή δομής (Structural). Μας ζητείται να χρησιμοποιήσουμε τη δομική μονάδα που υλοποιήθηκε στο προηγούμενο ερώτημα.

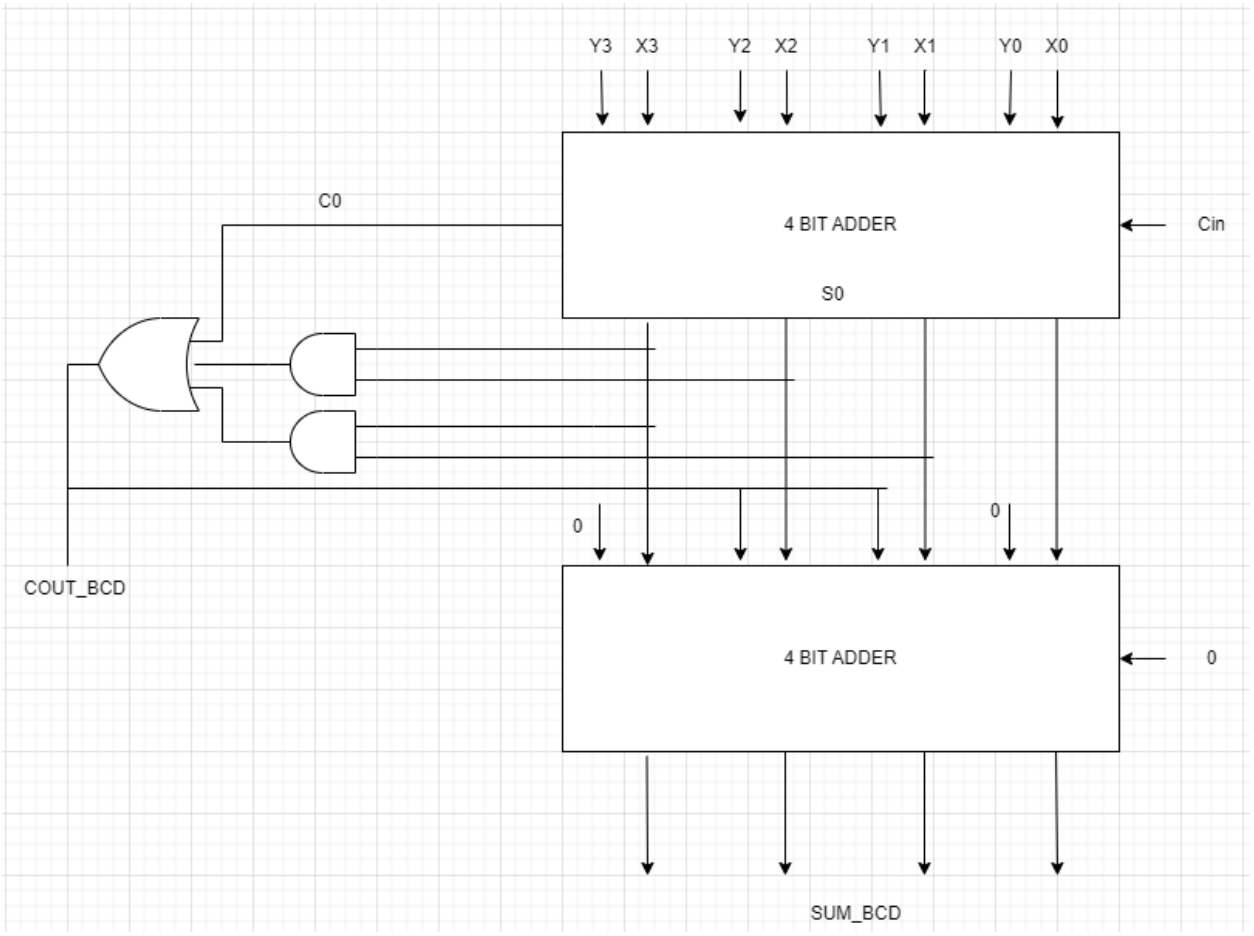
Ένας πλήρης προσθέτης δυαδικού κώδικα δεκαδικού (BCD) είναι ένα ψηφιακό κύκλωμα απαραίτητο για την πρόσθεση Binary Coded Decimal αριθμούς. Αντίθετα με την τυπική δυαδική πρόσθεση, η πρόσθεση BCD απαιτεί συγκεκριμένη χειρισμό για να διασφαλίσει ότι τα αποτελέσματα παραμένουν εντός του έγκυρου εύρους BCD (0 έως 9). Ο πλήρης αθροιστής BCD διενεργεί δυαδική πρόσθεση δεκαδικών ψηφίων, με επιπλέον λογική για την ανίχνευση και διόρθωση οποιουδήποτε άθροισμα υπερβαίνει το 9 προσθέτοντας 6 (0110). Αυτή η διόρθωση διατηρεί την εγκυρότητα του αποτελέσματος ως δεκαδικός αριθμός BCD. Επιπλέον, το κύκλωμα εκτελεί μεταφορές μεταξύ των δεκαδικών ψηφίων για να χειριστεί ακριβείς προσθήκες πολλαπλών ψηφίων.

Έχουμε τον εξής πίνακα:

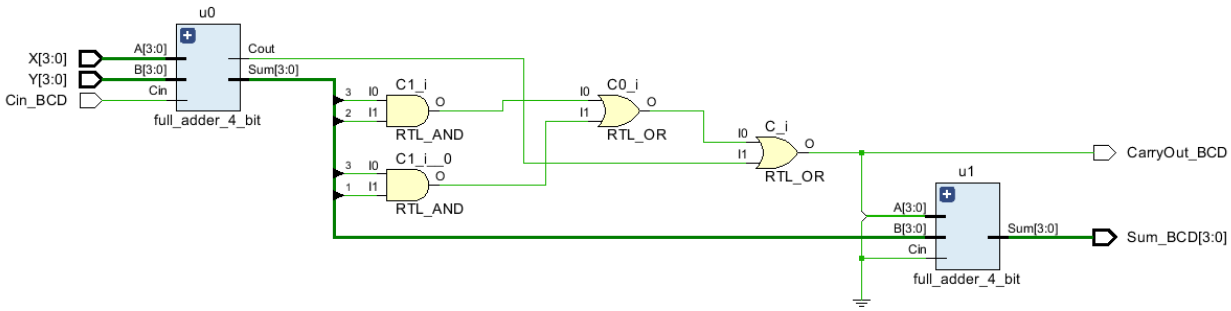
Decimal	**Binary Sum**					**BCD Sum**				
	C'	**S3'**	**S2'**	**S1'**	**S0'**	**C**	**S3**	**S2**	**S1**	**S0**
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	1
2	0	0	0	1	0	0	0	0	1	0
3	0	0	0	1	1	0	0	0	1	1
4	0	0	1	0	0	0	0	1	0	0
5	0	0	1	0	1	0	0	1	0	1
6	0	0	1	1	0	0	0	1	1	0
7	0	0	1	1	1	0	0	1	1	1
8	0	1	0	0	0	0	1	0	0	0
9	0	1	0	0	1	0	1	0	0	1
10	0	1	0	1	0	1	0	0	0	0

11	0	1	0	1	1	1	0	0	0	1
12	0	1	1	0	0	1	0	0	1	0
13	0	1	1	0	1	1	0	0	1	1
14	0	1	1	1	0	1	0	1	0	0
15	0	1	1	1	1	1	0	1	0	1
16	1	0	0	0	0	1	0	1	1	0
17	1	0	0	0	1	1	0	1	1	1
18	1	0	0	1	0	1	1	0	0	0
19	1	0	0	1	1	1	1	0	0	1

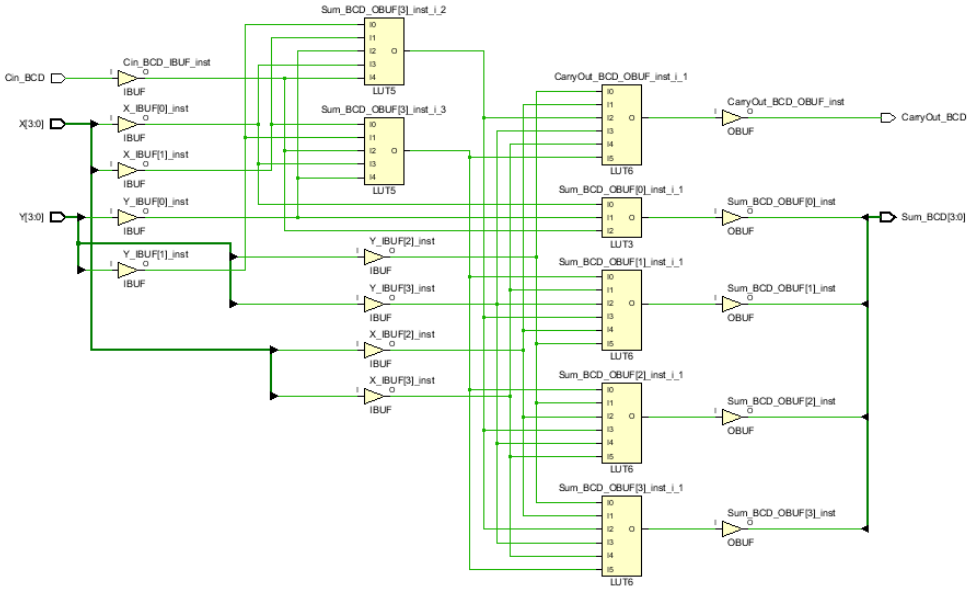
Μπορούμε το κύκλωμα του BCD Full Adder να το φτιάξουμε χρησιμοποιώντας δύο 4bit Adders και μερικές πύλες ως εξής:



Το παραπάνω περιμένουμε να δούμε και στο RTL:



Από τον Synthesizer προκύπτει το παρακάτω schematic:



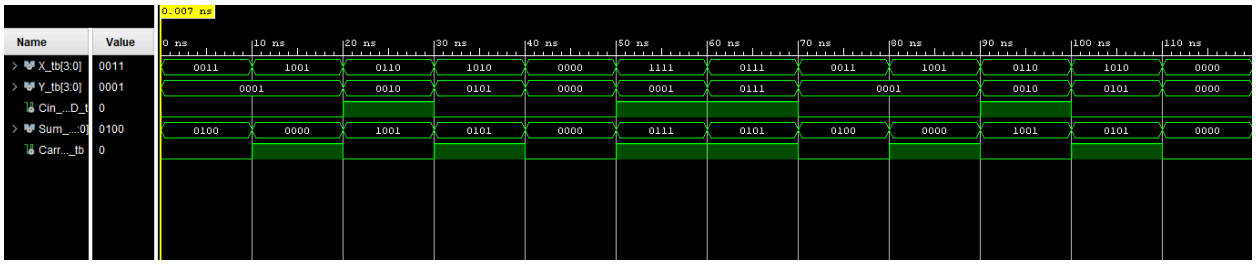
Δημιουργούμε TestBench για να ελέγξουμε την ορθή λειτουργία του κυκλώματος:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity bcd_full_adder_tb is
end bcd_full_adder_tb;

architecture tb_arch of bcd_full_adder_tb is
```

```
  component bcd_full_adder is
    Port (
      X : in std_logic_vector(3 downto 0);
      Y : in std_logic_vector(3 downto 0);
      Cin_BCD : in std_logic;
      Sum_BCD: out std_logic_vector(3 downto 0);
      CarryOut_BCD : out std_logic
    );
  end component;

  signal X_tb, Y_tb : std_logic_vector(3 downto 0);
  signal Cin_BCD_tb : std_logic;

  signal Sum_BCD_tb : std_logic_vector(3 downto 0);
  signal CarryOut_BCD_tb : std_logic;

begin

  DUT : bcd_full_adder
    port map(
      X => X_tb,
      Y => Y_tb,
      Cin_BCD => Cin_BCD_tb,
      Sum_BCD => Sum_BCD_tb,
      CarryOut_BCD => CarryOut_BCD_tb
    );

  stimulus : process
  begin
    X_tb <= "0011";
    Y_tb <= "0001";
    Cin_BCD_tb <= '0';
    wait for 10 ns;

    X_tb <= "1001";
    Y_tb <= "0001";
    Cin_BCD_tb <= '0';
    wait for 10 ns;
```

```
    X_tb <= "0110";
    Y_tb <= "0010";
    Cin_BCD_tb <= '1';
    wait for 10 ns;

    X_tb <= "1010";
    Y_tb <= "0101";
    Cin_BCD_tb <= '0';
    wait for 10 ns;

    X_tb <= "0000";
    Y_tb <= "0000";
    Cin_BCD_tb <= '0';
    wait for 10 ns;

    X_tb <= "1111";
    Y_tb <= "0001";
    Cin_BCD_tb <= '1';
    wait for 10 ns;

    X_tb <= "0111";
    Y_tb <= "0111";
    Cin_BCD_tb <= '1';
    wait for 10 ns;


  end process stimulus;

end tb_arch;
```

Η προσομοίωση επιβεβαιώνει την ορθή λειτουργία του κυκλώματος:



Για το critical path:

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Exception	Clock Uncertainty
Path 1	∞	4	5	4	Y[1]	CarryOut_BCD	5.976	3.904	2.072	∞	input port clock			0.000
Path 2	∞	4	5	4	Y[1]	Sum_BCD[3]	5.976	3.904	2.072	∞	input port clock			0.000
Path 3	∞	4	5	4	Cin_BCD	Sum_BCD[1]	5.948	3.876	2.072	∞	input port clock			0.000
Path 4	∞	4	5	4	Cin_BCD	Sum_BCD[2]	5.948	3.876	2.072	∞	input port clock			0.000
Path 5	∞	3	4	3	Cin_BCD	Sum_BCD[0]	5.351	3.752	1.599	∞	input port clock			0.000

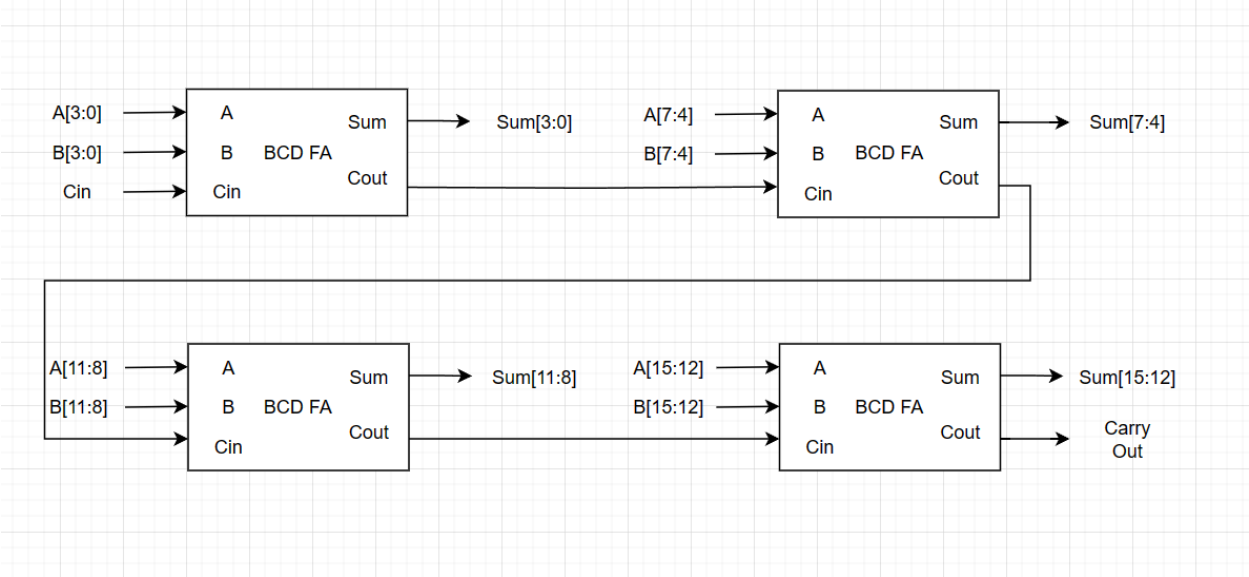
Από τα παραπάνω προκύπτει ότι το critical path είναι εκείνο από την είσοδο Cin στο άθροισμα Sum_BCD[1] και Sum_BCD[1] (MSB και μεσαίο Bit) με συνολική καθυστέρηση 5.948ns.

```
Max Delay Paths
--------------------------------------------------------------------------------------
Slack:                    inf
  Source:                 Y[1]
                            (input port)
  Destination:            CarryOut_BCD
                            (output port)
  Path Group:             (none)
  Path Type:              Max at Slow Process Corner
  Data Path Delay:        5.976ns  (logic 3.904ns (65.321%)  route 2.072ns (34.679%))
  Logic Levels:           4  (IBUF=1 LUT5=1 LUT6=1 OBUF=1)

    Location             Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                                                       0.000     0.000 r  Y[1] (IN)
                         net (fo=0)                    0.000     0.000    Y[1]
                         IBUF (Prop_ibuf_I_O)          0.982     0.982 r  Y_IBUF[1]_inst/O
                         net (fo=2, unplaced)          0.800     1.782    Y_IBUF[1]
                         LUT5 (Prop_lut5_I1_O)         0.152     1.934 r  Sum_BCD_OBUF[3]_inst_i_3/O
                         net (fo=4, unplaced)          0.473     2.407    S0[1]
                         LUT6 (Prop_lut6_I5_O)         0.124     2.531 r  CarryOut_BCD_OBUF_inst_i_1/O
                         net (fo=1, unplaced)          0.800     3.331    CarryOut_BCD_OBUF
                         OBUF (Prop_obuf_I_O)          2.645     5.976 r  CarryOut_BCD_OBUF_inst/O
                         net (fo=0)                    0.000     5.976    CarryOut_BCD
                                                                       r  CarryOut_BCD (OUT)
  -------------------------------------------------------------------    -------------------

Min Delay Paths
--------------------------------------------------------------------------------------
Slack:                    inf
  Source:                 X[3]
                            (input port)
  Destination:            CarryOut_BCD
                            (output port)
  Path Group:             (none)
  Path Type:              Min at Fast Process Corner
  Data Path Delay:        2.092ns  (logic 1.418ns (67.775%)  route 0.674ns (32.225%))
  Logic Levels:           3  (IBUF=1 LUT6=1 OBUF=1)

    Location             Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                                                       0.000     0.000 r  X[3] (IN)
                         net (fo=0)                    0.000     0.000    X[3]
                         IBUF (Prop_ibuf_I_O)          0.211     0.211 r  X_IBUF[3]_inst/O
                         net (fo=4, unplaced)          0.337     0.548    X_IBUF[3]
                         LUT6 (Prop_lut6_I4_O)         0.045     0.593 r  CarryOut_BCD_OBUF_inst_i_1/O
                         net (fo=1, unplaced)          0.337     0.930    CarryOut_BCD_OBUF
                         OBUF (Prop_obuf_I_O)          1.162     2.092 r  CarryOut_BCD_OBUF_inst/O
                         net (fo=0)                    0.000     2.092    CarryOut_BCD
                                                                       r  CarryOut_BCD (OUT)
  -------------------------------------------------------------------    -------------------
```

Θέμα 5: 4-BCD Parallel Adder

Στο τελευταίο ζητούμενο της άσκησης, ζητείται η υλοποίηση ενός Παράλληλου BCD Αθροιστή 4 ψηφίων (4-BCD Parallel Adder - 4-BCD PA). Ο 4-BCD PA είναι μια δομή με 2 εισόδους, Α και Β, η καθεμία εκ των οποίων αποτελείται από 16 bits. Για τους σκοπούς της άσκησης, είναι θεμιτό να αξιοποιήσουμε τέσσερις επιμέρους BCD Αθροιστές, όπως υλοποιήθηκαν στο προηγούμενο Θέμα.

Το διάγραμμα που προκύπτει, με βάση αυτή την υλοποίηση, είναι το παρακάτω:

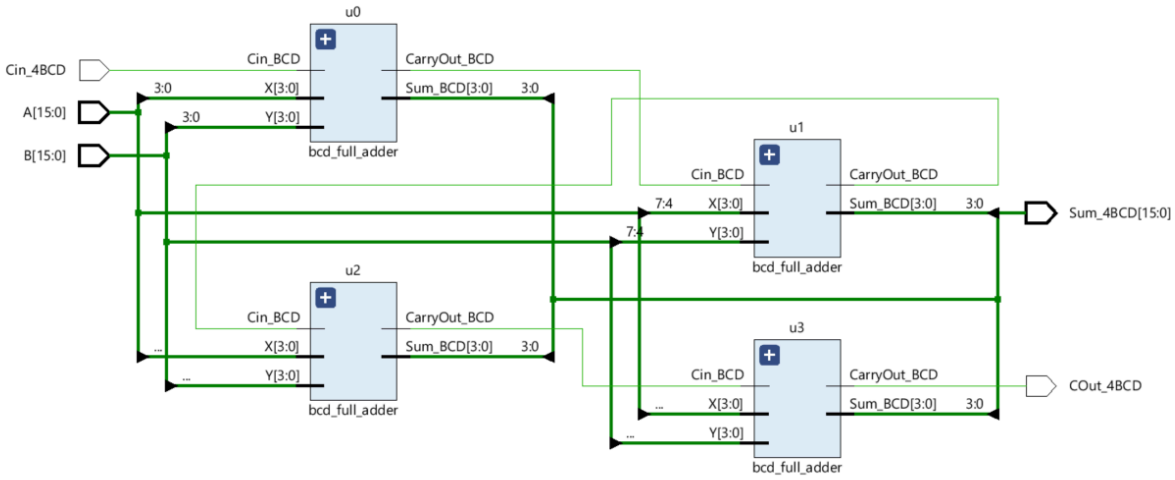


Με αυτή την λογική δημιουργούμε τον ακόλουθο κώδικα στο Vivado.

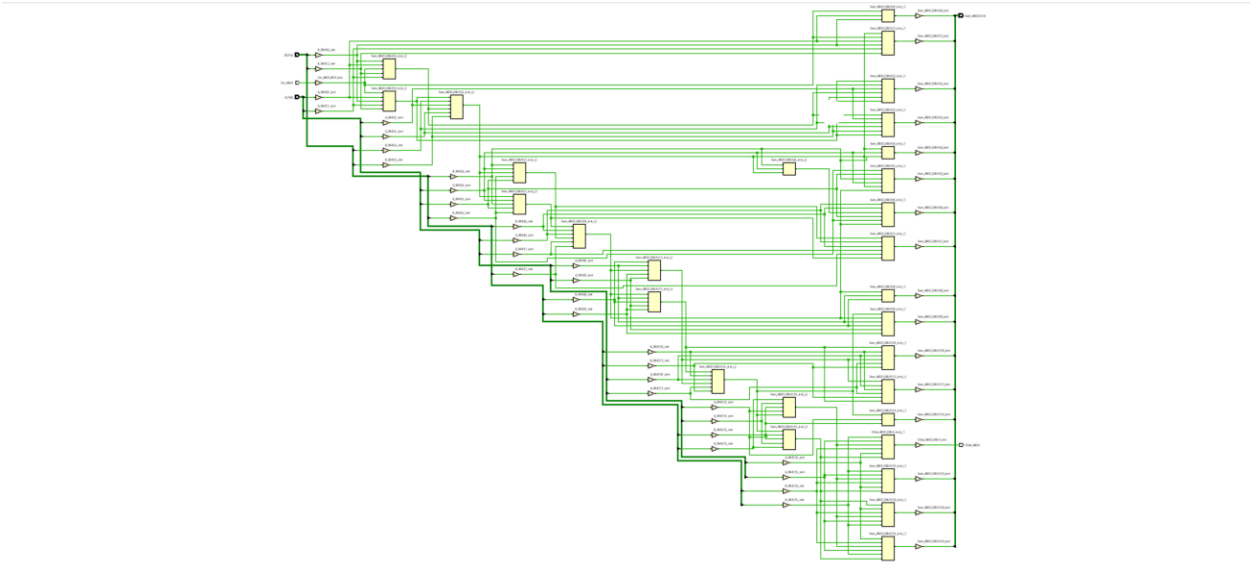
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity bcd_parallel_adder is
    Port (
        A: in std_logic_vector(15 downto 0);
        B: in std_logic_vector(15 downto 0);
        Cin_4BCD: in std_logic;
        Sum_4BCD: out std_logic_vector(15 downto 0);
```

```
        COut_4BCD: out std_logic
    );
end bcd_parallel_adder;

architecture Structural of bcd_parallel_adder is

    component bcd_full_adder is
      Port (
        X : in std_logic_vector(3 downto 0);
        Y : in std_logic_vector(3 downto 0);
        Cin_BCD : in std_logic;
        Sum_BCD: out std_logic_vector(3 downto 0);
        CarryOut_BCD : out std_logic
       );
    end component;

    signal c0, c1, c2: std_logic;


begin

    u0: bcd_full_adder port map (X => A(3 downto 0), Y => B(3 downto
0), Cin_BCD => Cin_4BCD, Sum_BCD => Sum_4BCD(3 downto 0),
CarryOut_BCD => c0);
    u1: bcd_full_adder port map (X => A(7 downto 4), Y => B(7 downto
4), Cin_BCD => c0, Sum_BCD => Sum_4BCD(7 downto 4), CarryOut_BCD =>
c1);
    u2: bcd_full_adder port map (X => A(11 downto 8), Y => B(11
downto 8), Cin_BCD => c1, Sum_BCD => Sum_4BCD(11 downto 8),
CarryOut_BCD => c2);
    u3: bcd_full_adder port map (X => A(15 downto 12), Y => B(15
downto 12), Cin_BCD => c2, Sum_BCD => Sum_4BCD(15 downto 12),
CarryOut_BCD => COut_4BCD);

end Structural;
```

Αναμένουμε, λοιπόν, να έχουμε την ίδια εικόνα με παραπάνω και από το RTL Analysis Design:



Από τον Synthesizer προκύπτει το παρακάτω schematic:



Δημιουργούμε TestBench για να ελέγξουμε την ορθή λειτουργία του κυκλώματος:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity bcd_parallel_adder_tb is
end bcd_parallel_adder_tb;

architecture tb_arch of bcd_parallel_adder_tb is
```

```
    component bcd_parallel_adder is
        Port (
            A: in std_logic_vector(15 downto 0);
            B: in std_logic_vector(15 downto 0);
            Cin_4BCD: in std_logic;
            Sum_4BCD: out std_logic_vector(15 downto 0);
            COut_4BCD: out std_logic
        );
    end component;

    signal A_tb, B_tb, Sum_4BCD_tb : std_logic_vector(15 downto 0);
    signal Cin_4BCD_tb, COut_4BCD_tb : std_logic;

begin

    dut: bcd_parallel_adder
        port map (
            A => A_tb,
            B => B_tb,
            Cin_4BCD => Cin_4BCD_tb,
            Sum_4BCD => Sum_4BCD_tb,
            COut_4BCD => COut_4BCD_tb
        );

    simulation : process
    begin
        A_tb <= "0011001111001010";
        B_tb <= "0011110011010010";
        Cin_4BCD_tb <= '0';
        wait for 10 ns;
        -- Expected output: Sum_4BCD_tb = "0000010011111110" and
COut_4BCD_tb = '0'

        A_tb <= "0000001100100100";
        B_tb <= "0000000111011010";
        Cin_4BCD_tb <= '0';
        wait for 10 ns;
        -- Expected output: Sum_4BCD_tb = "0000010101100100" and
COut_4BCD_tb = '0'
```

```
        A_tb <= "0010110000111010";
        B_tb <= "0011101001101001";
        Cin_4BCD_tb <= '0';
        wait for 10 ns;
        -- Expected output: Sum_4BCD_tb = "0110110100001001" and
COut_4BCD_tb = '0'

        A_tb <= "0000100110010110";
        B_tb <= "0001001010111001";
        Cin_4BCD_tb <= '1';
        wait for 10 ns;
        -- Expected output: Sum_4BCD_tb = "0010001010110110" and
COut_4BCD_tb = '0'
    end process;

end tb_arch;
```

Η προσομοίωση επιβεβαιώνει την ορθή λειτουργία του κυκλώματος:

Για το critical path:

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Exception	Clock Uncertainty
Path 1	∞	10	11	6	A[1]	Sum_4BCD[14]	9.521	4.620	4.901	∞	input port clock			0.000
Path 2	∞	10	11	6	A[1]	COut_4BCD	9.515	4.614	4.901	∞	input port clock			0.000
Path 3	∞	10	11	6	A[1]	Sum_4BCD[13]	9.515	4.614	4.901	∞	input port clock			0.000
Path 4	∞	10	11	6	A[1]	Sum_4BCD[15]	9.515	4.614	4.901	∞	input port clock			0.000
Path 5	∞	9	10	6	A[1]	Sum_4BCD[12]	8.924	4.496	4.428	∞	input port clock			0.000
Path 6	∞	9	10	6	A[1]	Sum_4BCD[9]	8.924	4.496	4.428	∞	input port clock			0.000
Path 7	∞	8	9	6	A[1]	Sum_4BCD[10]	8.327	4.372	3.955	∞	input port clock			0.000
Path 8	∞	8	9	6	A[1]	Sum_4BCD[11]	8.321	4.366	3.955	∞	input port clock			0.000
Path 9	∞	7	8	6	A[1]	Sum_4BCD[5]	7.736	4.248	3.488	∞	input port clock			0.000
Path 10	∞	7	8	6	A[1]	Sum_4BCD[6]	7.736	4.248	3.488	∞	input port clock			0.000

Από τα παραπάνω προκύπτει ότι το critical path είναι αυτό από την είσοδο A[1] προς την έξοδο Sum_4BCD[14], με συνολική καθυστέρηση 9.521ns.

```
Max Delay Paths
--------------------------------------------------------------------------------
Slack:                  inf
  Source:               A[1]
                          (input port)
  Destination:          Sum_4BCD[14]
                          (output port)
  Path Group:           (none)
  Path Type:            Max at Slow Process Corner
  Data Path Delay:      9.521ns  (logic 4.620ns (48.520%)  route 4.901ns (51.480%))
  Logic Levels:         10  (IBUF=1 LUT5=4 LUT6=4 OBUF=1)

    Location             Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                                                       0.000     0.000 f  A[1] (IN)
                       net (fo=0)                      0.000     0.000    A[1]
                       IBUF (Prop_ibuf_I_O)            0.982     0.982 f  A_IBUF[1]_inst/O
                       net (fo=3, unplaced)            0.800     1.782    A_IBUF[1]
                       LUT5 (Prop_lut5_I4_O)           0.124     1.906 f  Sum_4BCD_OBUF[3]_inst_i_2/O
                       net (fo=3, unplaced)            0.467     2.373    Sum_4BCD_OBUF[3]_inst_i_2_n_0
                       LUT6 (Prop_lut6_I3_O)           0.124     2.497 f  Sum_4BCD_OBUF[5]_inst_i_2/O
                       net (fo=6, unplaced)            0.481     2.978    Sum_4BCD_OBUF[5]_inst_i_2_n_0
                       LUT5 (Prop_lut5_I2_O)           0.124     3.102 f  Sum_4BCD_OBUF[7]_inst_i_2/O
                       net (fo=2, unplaced)            0.460     3.562    Sum_4BCD_OBUF[7]_inst_i_2_n_0
                       LUT6 (Prop_lut6_I3_O)           0.124     3.686 f  Sum_4BCD_OBUF[9]_inst_i_2/O
                       net (fo=6, unplaced)            0.481     4.167    Sum_4BCD_OBUF[9]_inst_i_2_n_0
                       LUT5 (Prop_lut5_I2_O)           0.124     4.291 f  Sum_4BCD_OBUF[11]_inst_i_2/O
                       net (fo=3, unplaced)            0.467     4.758    Sum_4BCD_OBUF[11]_inst_i_2_n_0
                       LUT6 (Prop_lut6_I3_O)           0.124     4.882 f  Sum_4BCD_OBUF[12]_inst_i_2/O
                       net (fo=4, unplaced)            0.473     5.355    Sum_4BCD_OBUF[12]_inst_i_2_n_0
                       LUT5 (Prop_lut5_I4_O)           0.124     5.479 r  Sum_4BCD_OBUF[15]_inst_i_2/O
                       net (fo=4, unplaced)            0.473     5.952    Sum_4BCD_OBUF[15]_inst_i_2_n_0
                       LUT6 (Prop_lut6_I3_O)           0.124     6.076 r  Sum_4BCD_OBUF[14]_inst_i_1/O
                       net (fo=1, unplaced)            0.800     6.876    Sum_4BCD_OBUF[14]
                       OBUF (Prop_obuf_I_O)            2.645     9.521 r  Sum_4BCD_OBUF[14]_inst/O
                       net (fo=0)                      0.000     9.521    Sum_4BCD[14]
                                                                       r  Sum_4BCD[14] (OUT)
  -------------------------------------------------------------------    -------------------


Min Delay Paths
--------------------------------------------------------------------------------
Slack:                  inf
  Source:               A[14]
                          (input port)
  Destination:          COut_4BCD
                          (output port)
  Path Group:           (none)
  Path Type:            Min at Fast Process Corner
  Data Path Delay:      2.092ns  (logic 1.418ns (67.775%)  route 0.674ns (32.225%))
  Logic Levels:         3  (IBUF=1 LUT6=1 OBUF=1)

    Location             Delay type                Incr(ns)  Path(ns)    Netlist Resource(s)
  -------------------------------------------------------------------    -------------------
                                                       0.000     0.000 r  A[14] (IN)
                       net (fo=0)                      0.000     0.000    A[14]
                       IBUF (Prop_ibuf_I_O)            0.211     0.211 r  A_IBUF[14]_inst/O
                       net (fo=4, unplaced)            0.337     0.548    A_IBUF[14]
                       LUT6 (Prop_lut6_I4_O)           0.045     0.593 r  COut_4BCD_OBUF_inst_i_1/O
                       net (fo=1, unplaced)            0.337     0.930    COut_4BCD_OBUF
                       OBUF (Prop_obuf_I_O)            1.162     2.092 r  COut_4BCD_OBUF_inst/O
                       net (fo=0)                      0.000     2.092    COut_4BCD
                                                                       r  COut_4BCD (OUT)
  -------------------------------------------------------------------    -------------------
```