محمد مهدی نظری ۹۹۳۱۰۶۱ – آرمین ابراهیمی صبا ۹۹۳۱۰۸۶

گزارش کار این آزمایش به دو بخش تقسیم شده که بخش اول شامل ripple adder و carry select adder هستند و بخش دوم شامل carry lookahead adder است .

قطعه کد این بخش مطابق مدار های ارائه شده در بخش پیش گزارش طراحی شده اند .

۱ – ripple carry adder

این جمع کننده ۴ بیتی شامل ۴ جمع کننده کامل است که متوالیا به هم از طریق کری خروجی وصل شده اند .

```
19  ----------------------------------------------------------------------------------
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22
23  -- Uncomment the following library declaration if using
24  -- arithmetic functions with Signed or Unsigned values
25  --use IEEE.NUMERIC_STD.ALL;
26
27  -- Uncomment the following library declaration if instantiating
28  -- any Xilinx primitives in this code.
29  --library UNISIM;
30  --use UNISIM.VComponents.all;
31
32  entity ripple_carry_adder is
33     port(
34        a, b : in STD_LOGIC_VECTOR(3 downto 0);
35        cin : in STD_LOGIC;
36        sum: out STD_LOGIC_VECTOR(3 downto 0);
37        cout : out STD_LOGIC
38     );
39  end entity ;
40
41  architecture Behavioral of ripple_carry_adder is
42
43     component full_adder is
44       Port ( a : in STD_LOGIC;
45              b : in STD_LOGIC;
46              Cin : in STD_LOGIC;
47              sum : out STD_LOGIC;
48              Cout : out STD_LOGIC);
49     end component ;
50
51     signal c : STD_LOGIC_VECTOR(2 downto 0);
52
53  begin
54
```

```
53  begin
54
55     FullAdder1 : full_adder port map (
56        a => a(0),
57        b => b(0),
58        Cin => cin,
59        sum => sum(0),
60        Cout => c(0)
61     );
62
63     FullAdder2 : full_adder port map (
64        a => a(1),
65        b => b(1),
66        Cin => c(0),
67        sum => sum(1),
68        Cout => c(1)
69     );
70
71     FullAdder3 : full_adder port map (
72        a => a(2),
73        b => b(2),
74        Cin => c(1),
75        sum => sum(2),
76        Cout => c(2)
77     );
78
79     FullAdder4 : full_adder port map (
80        a => a(3),
81        b => b(3),
82        Cin => c(2),
83        sum => sum(3),
84        Cout => cout
85     );
86
87
88  end Behavioral ;
89
```

۲ – carry select adder

این جمع کننده از دو جمع کننده ریپل تشکیل شده که کری یکی ۱ و دیگری ۰ داده شده و بصورت موازی با هم کار میکنند و در نهایت هر ۵ خروجی هر کدام متناظرا به یک ماکس ۲ به ۱ میروند که کلید آن ماکس کری ورودی داده شده است .

```
32  entity carry_select_adder is
33     port(
34        a, b : in STD_LOGIC_VECTOR(3 downto 0);
35        cin : in STD_LOGIC;
36        sum: out STD_LOGIC_VECTOR(3 downto 0);
37        cout : out STD_LOGIC
38     );
39  end carry_select_adder;
40
41  architecture Behavioral of carry_select_adder is
42
43  component ripple_carry_adder is
44     port(
45        a, b : in STD_LOGIC_VECTOR(3 downto 0);
46        cin : in STD_LOGIC;
47        sum: out STD_LOGIC_VECTOR(3 downto 0);
48        cout : out STD_LOGIC
49     );
50  end component ;
51
52  component mux_2to1 is
53  port (
54  a , b : in std_logic ;
55  s : in std_logic ;
56  output : out std_logic );
57  end component ;
58
59  signal sum0,sum1 : std_logic_vector( 3 downto 0 );
60  signal cout0,cout1 : std_logic ;
61
62  begin
63
64  ripple_0 : ripple_carry_adder port map (
65  a => a ,
66  b => b ,
67  cin => '0' ,
68  sum => sum0 ,
```

```
71  ripple_1 : ripple_carry_adder port map
72  a => a ,
73  b => b ,
74  cin => '1' ,
75  sum => sum1 ,
76  cout => cout1 );
77
78  mux_cout : mux_2to1 port map (
79  a => cout0 ,
80  b => cout1 ,
81  s => cin ,
82  output => cout );
83
84  mux_sum0 : mux_2to1 port map (
85  a => sum0(0) ,
86  b => sum1(0) ,
87  s => cin ,
88  output => sum(0) );
89
90  mux_sum1 : mux_2to1 port map (
91  a => sum0(1) ,
92  b => sum1(1) ,
93  s => cin ,
94  output => sum(1) );
95
96  mux_sum2 : mux_2to1 port map (
97  a => sum0(2) ,
98  b => sum1(2) ,
99  s => cin ,
100 output => sum(2) );
101
102 mux_sum3 : mux_2to1 port map (
103 a => sum0(3) ,
104 b => sum1(3) ,
105 s => cin ,
106 output => sum(3) );
107
```

```
32  entity mux_2to1 is
33  port (
34  a , b : in std_logic ;
35  s : in std_logic ;
36  output : out std_logic );
37  end mux_2to1;
38
39  architecture Behavioral of mux_2to1 is
40  component or_gate is
41      Port ( a : in STD_LOGIC;
42             b : in STD_LOGIC;
43             output : out STD_LOGIC);
44  end component;
45
46  component and_gate is
47      Port ( a : in std_logic;
48             b : in std_logic;
49             output : out std_logic);
50  end component;
51
52  signal output0 , output1 : std_logic ;
53  begin
54  and_instance1 : and_gate port map (
55  a => a ,
56  b => not s ,
57  output => output0 );
58
59  and_instance2 : and_gate port map (
60  a => s ,
61  b => b ,
62  output => output1 );
63
64  or_instance : or_gate port map (
65  a => output0 ,
66  b => output1 ,
67  output => output );
```

تست بنچ ها :

تست بنچ هردو ماژول مانند هم با ۵ حالت زیر زده شده . برای تست هم بعنوان مثال حاصل جمع ۱۰۱۰ و ۱۱۱۰ با کری ورودی ۱ باید عدد ۵ بیتی ۱۱۰۰۱ یا عدد ۴ بیتی ۱۰۰۱ با کری خروجی ۱ را بدهد .

14 + 10 + 1 = 25

یا مثلا ۰۰۰۰ و ۱۱۱۱ با کری ورودی ۰ باید ۰۱۱۱۱ یا ۱۱۱۱ با کری خروجی ۰ بدهد.

15 + 0 + 0 = 15

```
59
60  process
61  begin
62  a <= "1010";
63  b <= "1110";
64  cin <= '1' ;
65  wait for 15 ns ;
66
67  a <= "1110";
68  b <= "0010";
69  cin <= '1' ;
70  wait for 15 ns ;
71
72  a <= "0000";
73  b <= "1111";
74  cin <= '0' ;
75  wait for 15 ns ;
76
77  a <= "1000";
78  b <= "0110";
79  cin <= '0' ;
80  wait for 15 ns ;
81
82  a <= "1010";
83  b <= "1000";
84  cin <= '1' ;
85  wait for 15 ns ;
86
87  end process ;
88
89  end test;
90
```