

بسم تعالی



دانشگاه صنعتی امیر کبیر
(پلی تکنیک تهران)

آزمایش شماره ۵

کیان پورآذر : ۴۰۱۳۱۴۰۳
علی فرزانه : ۴۰۱۳۱۴۱۵

استاد:
خانم مهندس فرجلو

Half Adder:

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

entity Half_Adder is
    Port ( A : in STD_LOGIC;
          B : in STD_LOGIC;
          Sum : out STD_LOGIC;
          Cout : out STD_LOGIC);
end Half_Adder;

architecture Gate_Level of Half_Adder is
begin
    Sum <= A XOR B;
    Cout <= A AND B;
end Gate_Level;
```

Full Adder:

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

entity Full_Adder is
    Port ( A : in STD_LOGIC;
          B : in STD_LOGIC;
          Cin : in STD_LOGIC;
          Sum : out STD_LOGIC;
          Cout : out STD_LOGIC);
end Full_Adder;

architecture Gate_Level of Full_Adder is
begin
    Sum <= (A XOR B) XOR Cin;
    Cout <= (A AND B) OR (Cin AND (A XOR B));
end Gate_Level;
```

Normal Multiplier:

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

entity multiplier4bit is
  port (
    A: in STD_LOGIC_VECTOR (3 downto 0);
    B: in STD_LOGIC_VECTOR (3 downto 0);
    P: out STD_LOGIC_VECTOR (7 downto 0));
end multiplier4bit;

architecture Behavioral of multiplier4bit is

  component full_adder
    port (
      A,B: in STD_LOGIC;
      Cin: in STD_LOGIC;
      Cout: out STD_LOGIC;
      S: out STD_LOGIC);
  end component;

  component half_adder
    port (
      A,B: in STD_LOGIC;
      Cout: out STD_LOGIC;
      S: out STD_LOGIC);
  end component;

  signal AB: STD_LOGIC_VECTOR (15 downto 0);
  signal C1, C2, C3: STD_LOGIC_VECTOR (3 downto 0);
  signal S1, S2, S3: STD_LOGIC_VECTOR (3 downto 0);

begin

  AB(0) <= A(0) and B(0);

  AB(1) <= A(0) and B(1);
  AB(2) <= A(1) and B(0);

  AB(3) <= A(1) and B(1);
  AB(4) <= A(2) and B(0);

  AB(5) <= A(3) and B(0);
  AB(6) <= A(2) and B(1);

  AB(7) <= A(3) and B(1);
```

```

AB(8) <= A(0) and B(2);
AB(9) <= A(1) and B(2);
AB(10) <= A(2) and B(2);
AB(11) <= A(3) and B(2);

AB(12) <= A(0) and B(3);
AB(13) <= A(1) and B(3);
AB(14) <= A(2) and B(3);
AB(15) <= A(3) and B(3);

-----HALF_ADDERS-----
HA1: half_adder PORT MAP(AB(1),AB(2),C1(0),S1(0));
HA2: half_adder PORT MAP(C1(2),AB(7),C1(3),S1(3));

HA3: half_adder PORT MAP(S1(1),AB(8),C2(0),S2(0));

HA4: half_adder PORT MAP(S2(1),AB(12),C3(0),S3(0));
-----FULL_ADDERS-----
FA1: full_adder PORT MAP(AB(3),AB(4),C1(0),C1(1),S1(1));
FA2: full_adder PORT MAP(AB(5),AB(6),C1(1),C1(2),S1(2));

FA3: full_adder PORT MAP(S1(2),AB(9),C2(0),C2(1),S2(1));
FA4: full_adder PORT MAP(S1(3),AB(10),C2(1),C2(2),S2(2));
FA5: full_adder PORT MAP(C1(3),AB(11),C2(2),C2(3),S2(3));


FA6: full_adder PORT MAP(S2(2),AB(13),C3(0),C3(1),S3(1));
FA7: full_adder PORT MAP(S2(3),AB(14),C3(1),C3(2),S3(2));
FA8: full_adder PORT MAP(C2(3),AB(15),C3(2),C3(3),S3(3));

P(0) <= AB(0);
P(1) <= S1(0);
P(2) <= S2(0);
P(3) <= S3(0);
P(4) <= S3(1);
P(5) <= S3(2);
P(6) <= S3(3);
P(7) <= C3(3);

```

یک ضرب عادی، ضرب را با افزودن مکرر حاصل ضرب جزئی انجام می دهد. عملکرد آن مشابه فرایند ضرب دستی است، که در آن هر رقم از یک عدد با هر رقم دیگری ضرب می شود و نتایج انباشته می شوند. این فرایند معمولاً شامل عملیات جابجایی و اضافه می شود و به صورت متوالی و یک مرحله در یک زمان

Simulation:

		117.260 ns																
Name		Value		0 ns			200 ns			400 ns			600 ns			800 ns		
▼	 a[3:0]	6		14	6	1	12	10	14	6	1	12	10					
▼	 b[3:0]	13		12	13	7	15	14	12	13	7	15	14					
▼	 p[7:0]	78		168	78	7	180	140	168	78	7	180	140					

Array Multiplier:

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

entity Adder_4bit is
    Port ( A : in  STD_LOGIC_VECTOR (3 downto 0);
          B : in  STD_LOGIC_VECTOR (3 downto 0);
          sum : out STD_LOGIC_VECTOR (3 downto 0);
          cout : out STD_LOGIC);
end Adder_4bit;

architecture Behavioral of Adder_4bit is

    signal Carry : std_logic_vector(3 downto 0);

begin
    HA:entity work.Half_Addder port map (A(0), B(0), Sum(0), Carry(0));
    FA1:entity work.Full_Addder port map (A(1), B(1), Carry(0), Sum(1), Carry(1));
    FA2:entity work.Full_Addder port map (A(2), B(2), Carry(1), Sum(2), Carry(2));
    FA3:entity work.Full_Addder port map (A(3), B(3), Carry(2), Sum(3), Carry(3));
    cout <= Carry(3);

end Behavioral;
```

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

entity Array_Multi is
    Port ( A : in  STD_LOGIC_VECTOR (3 downto 0);
          B : in  STD_LOGIC_VECTOR (3 downto 0);
          P : out STD_LOGIC_VECTOR (7 downto 0));
end Array_Multi;

architecture Behavioral of Array_Multi is

    signal B0 : std_logic_vector(3 downto 0);
    signal B1 : std_logic_vector(3 downto 0);
    signal B2 : std_logic_vector(3 downto 0);
    signal B3 : std_logic_vector(3 downto 0);

    signal S1 : std_logic_vector(3 downto 0);
    signal S2 : std_logic_vector(3 downto 0);
    signal S3 : std_logic_vector(3 downto 0);
```

```

signal C : std_logic_vector(2 downto 0);

begin

B0(0)<= A(1) AND B(0);
B0(1)<= A(2) AND B(0);
B0(2)<= A(3) AND B(0);
B0(3)<= '0';

B1(0)<= A(0) AND B(1);
B1(1)<= A(1) AND B(1);
B1(2)<= A(2) AND B(1);
B1(3)<= A(3) AND B(1);

B2(0)<= A(0) AND B(2);
B2(1)<= A(1) AND B(2);
B2(2)<= A(2) AND B(2);
B2(3)<= A(3) AND B(2);

B3(0)<= A(0) AND B(3);
B3(1)<= A(1) AND B(3);
B3(2)<= A(2) AND B(3);
B3(3)<= A(3) AND B(3);

Adder4_1:entity work.Adder_4bit port map (B0 , B1 , S1,C(0));
Adder4_2:entity work.Adder_4bit port map (C(0)& S1(3 downto 1) , B2 ,S2,C(1));
Adder4_3:entity work.Adder_4bit port map (C(1)& S2(3 downto 1) , B3 ,S3,C(2));

p(0) <= A(0) AND B(0);
p(1) <= S1(0);
p(2) <= S2(0);
p(3) <= S3(0);
p(7 downto 4)<= C(2)& S3(3 downto 1);

end Behavioral;

```

ضرب آرایه‌ای یک مدار دیجیتالی است که در سخت افزار کامپیوتر برای ضرب دو عدد باینری استفاده می شود. با تجزیه عملیات ضرب به یک سری عملیات ساده تر عمل می کند که معمولاً از آرایه ای از دروازه های منطقی یا سلول هایی که در ردیف ها و ستون ها مرتب شده اند استفاده می کند. هر سلول یک محصول جزئی را محاسبه می کند و نتیجه نهایی با جمع بندی این محصولات جزئی به دست می آید. این طراحی امکان محاسبه کارآمد و موازی ضرب را فراهم می کند و آن را به یک جزء اساسی در واحدهای حسابی پردازنده ها تبدیل می کند.

Simulation:

