جامعة حلب في المناطق المحررة
كلية الهندسة المعلوماتية
السنة الرابعة

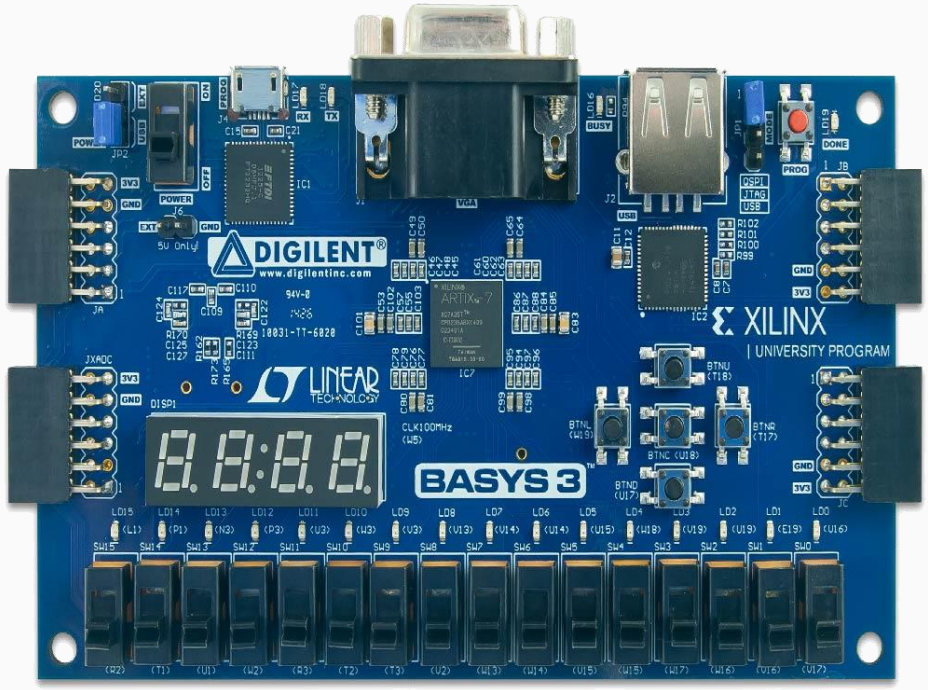
مقرر عملي

بنية وتنظيم الحواسيب2

تصميم معالج 4bit

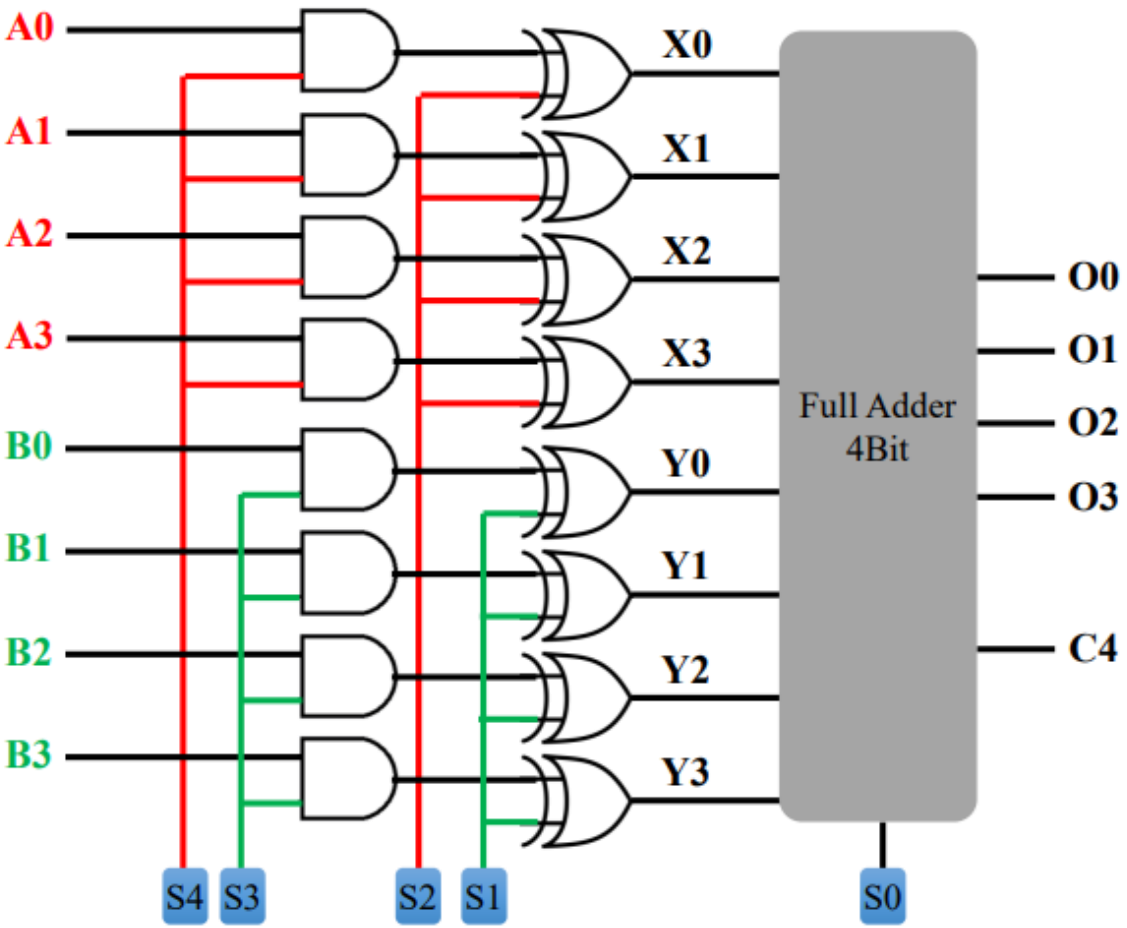
د.م. عبد القادر غزال
م. محمد نور بدوي

العام الدراسي: 2023 - 2024



المحاضرة العملية الخامسة

بناء معالج بسيط



مراجعة:

وصلنا في المحاضرة الماضية عند هذا الحد

حيث قمنا بعملية إنشاء معالج بسيط يقوم

بعمليتي الجمع والطرح.

بناء معالج بسيط

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

entity processor is
    Port ( a : in STD_LOGIC_VECTOR (3 downto 0);
           b : in STD_LOGIC_VECTOR (3 downto 0);
           o : out STD_LOGIC_VECTOR (3 downto 0);
           s: in STD_LOGIC_VECTOR (5 downto 0);
           c0 : in STD_LOGIC;
           c4 : out STD_LOGIC);
end processor;
```

بناء معالج بسيط

```
architecture Behavioral of processor is
        signal c3, c2, c1: std_logic;
        signal x: STD_LOGIC_VECTOR (3 downto 0);
        signal y: STD_LOGIC_VECTOR (3 downto 0);

    begin

        x(0) <= (a(0) and s(4)) xor s(2);
        x(1) <= (a(1) and s(4)) xor s(2);
        x(2) <= (a(2) and s(4)) xor s(2);
        x(3) <= (a(3) and s(4)) xor s(2);

        y(0) <= (b(0) and s(3)) xor s(1);
        y(1) <= (b(1) and s(3)) xor s(1);
        y(2) <= (b(2) and s(3)) xor s(1);
        y(3) <= (b(3) and s(3)) xor s(1);
```

بناء معالج بسيط

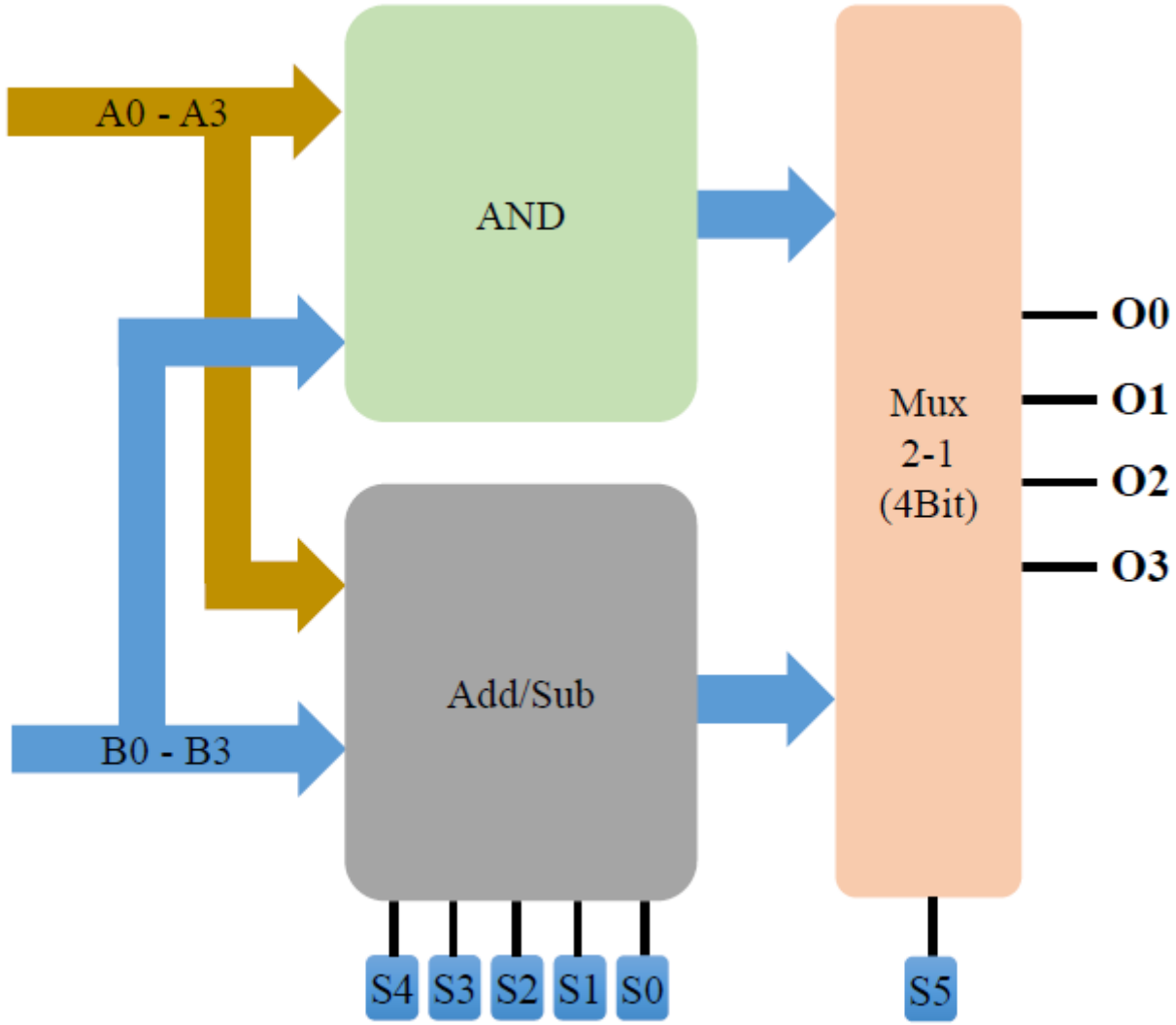
```
        -- The first full adder
        o(0) <= x(0) xor y(0) xor c0;
        c1 <= ( (x(0) xor y(0)) and c0 ) or (x(0) and y(0));

        -- The second full adder
        o(1) <= x(1) xor y(1) xor c1;
        c2 <= ( (x(1) xor y(1)) and c1 ) or (x(1) and y(1));

        -- The third full adder
        o(2) <= x(2) xor y(2) xor c2;
        c3 <= ( (x(2) xor y(2)) and c2 ) or (x(2) and y(2));

        -- The fourth full adder
        o(3) <= x(3) xor y(3) xor c3;
        c4 <= ( (x(3) xor y(3)) and c3 ) or (x(3) and y(3));
end Behavioral;
```

بناء معالج بسيط

ربط الأقطاب مع شريحة FPGA

Name	Dir...	...	Package Pin	Fixed	Bank	I/O Std	Vcco
All ports (20)							
a (4)	IN			☑	14	LVCMOS33*	3.300
a[3]	IN		W17	☑	14	LVCMOS33*	3.300
a[2]	IN		W16	☑	14	LVCMOS33*	3.300
a[1]	IN		V16	☑	14	LVCMOS33*	3.300
a[0]	IN		V17	☑	14	LVCMOS33*	3.300
b (4)	IN			☑	14	LVCMOS33*	3.300
b[3]	IN		W13	☑	14	LVCMOS33*	3.300
b[2]	IN		W14	☑	14	LVCMOS33*	3.300
b[1]	IN		V15	☑	14	LVCMOS33*	3.300
b[0]	IN		W15	☑	14	LVCMOS33*	3.300
o (4)	OI			☑	14	LVCMOS33*	3.300
o[3]	OI		V19	☑	14	LVCMOS33*	3.300
o[2]	OI		U19	☑	14	LVCMOS33*	3.300
o[1]	OI		E19	☑	14	LVCMOS33*	3.300
o[0]	OI		U16	☑	14	LVCMOS33*	3.300
s (6)	IN			☑	34	LVCMOS33*	3.300
s[5]	IN		T2	☑	34	LVCMOS33*	3.300
s[4]	IN		R3	☑	34	LVCMOS33*	3.300
s[3]	IN		W2	☑	34	LVCMOS33*	3.300
s[2]	IN		U1	☑	34	LVCMOS33*	3.300
s[1]	IN		T1	☑	34	LVCMOS33*	3.300
s[0]	IN		R2	☑	34	LVCMOS33*	3.300
Scalar ports (2)							
c0	IN		T3	☑	34	LVCMOS33*	3.300
c4	OI		L1	☑	35	LVCMOS33*	3.300

بناء معالج بسيط



يوضح الشكل الجانبي كيفية تعديل بنية المعالج

بعد إضافة العملية المنطقية And

يتطلب هذا الأمر إجراء عملية And للمداخل وبعد

ذلك نخب الخرج (إما عملية And أو جمع/ طرح)

بناء معالج بسيط

بعد التعديل يصبح لدينا

الكود التالي:

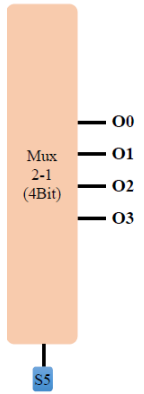
```
architecture Behavioral of processor is

    signal c3, c2, c1: std_logic;
    signal x: STD_LOGIC_VECTOR (3 downto 0);
    signal y: STD_LOGIC_VECTOR (3 downto 0);
    signal and_op: STD_LOGIC_VECTOR (3 downto 0);
    signal add_op: STD_LOGIC_VECTOR (3 downto 0);

begin

    and_op(0) <= a(0) and b(0);

    and_op(1) <= a(1) and b(1);

    and_op(2) <= a(2) and b(2);

    and_op(3) <= a(3) and b(3);
```

بناء معالج بسيط

بعد التعديل يصبح لدينا

الكود التالي:

```
add_op(0) <= x(0) xor y(0) xor c0;
c1 <= ( (x(0) xor y(0)) and c0 ) or (x(0) and y(0));

-- The second full adder
add_op(1) <= x(1) xor y(1) xor c1;
c2 <= ( (x(1) xor y(1)) and c1 ) or (x(1) and y(1));

-- The third full adder
add_op(2) <= x(2) xor y(2) xor c2;
c3 <= ( (x(2) xor y(2)) and c2 ) or (x(2) and y(2));

-- The fourth full adder
add_op(3) <= x(3) xor y(3) xor c3;
c4 <= ( (x(3) xor y(3)) and c3 ) or (x(3) and y(3));

-- Mux 2 -1 (4 bit)
o(0) <= add_op(0) when (s(5) = '1') else and_op(0);
o(1) <= add_op(1) when (s(5) = '1') else and_op(1);
o(2) <= add_op(2) when (s(5) = '1') else and_op(2);
o(3) <= add_op(3) when (s(5) = '1') else and_op(3);
```



انتهت المحاضرة