การจัดองค์การคอมพิวเตอร์

Project Overview

31110321 Computer Organization
สำหรับนักศึกษาชั้นปีที่ 3 สาขาวิชาวิศวกรรมคอมพิวเตอร์

ทรงฤทธิ์ กิติศรีวรพันธุ์

songrit@npu.ac.th

สาขาวิชาวิศวกรรมคอมพิวเตอร์
มหาวิทยาลัยนครพนม

Revised 202-02-15

Lecture plan

- 1.1 Boolean Logic
- 1.2 Boolean Functions Synthesis
- 1.3 Logic Gates
- 1.4 Hardware Description Language
- 1.5 Hardware Simulation
- 1.6 Multi-Bit Buses
- **1.7 Project Overview**

Project 1

- Given : Nand
- Goal : สร้างเกตต่อไปนี้

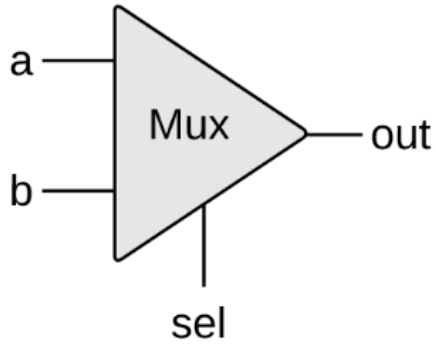
Project 1

- Given : Nand
- Goal : สร้างเกตต่อไปนี้

Elementary logic gates	16-bit variants	Multi-way variants
Not	Not16	Or8Way
And	And16	Mux4Way16
Or	Or16	Mux8Way16
Xor	Mux16	DMux4Way
Mux ←		DMux8Way
DMux ←		

Mux

- Multiplexor




```
if (sel==0)
   out=a
else
   out=b
```

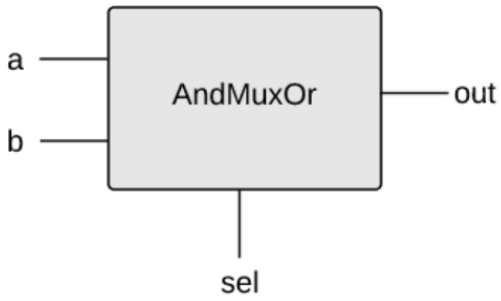
a	b	sel	out
0	0	0	**0**
0	1	0	**0**
1	0	0	**1**
1	1	0	**1**
0	0	1	**0**
0	1	1	**1**
1	0	1	**0**
1	1	1	**1**

sel	out
0	**a**
1	**b**

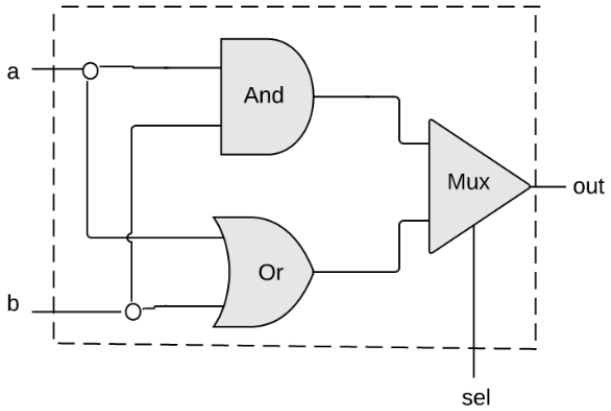
ความหมายเดียวกับ truth table

- 2-way multiplexor ใช้ sel เลือกข้อมูลทางใดทางหนึ่ง
- ใช้แพร่หลาย:
  - วงจรดิจิทัล
  - การสื่อสารเครือข่ายคอมพิวเตอร์

สร้างเกตที่โปรแกรมได้ด้วย mux




```
if (sel==0)
   out = (a And b)
else
   out = (a Or b)
```



a	b	sel	out
0	0	0	**0**
0	1	0	**0**
1	0	0	**0**
1	1	0	**1**
0	0	1	**0**
0	1	1	**1**
1	0	1	**1**
1	1	1	**1**

When `sel==0` the gate acts like an And gate

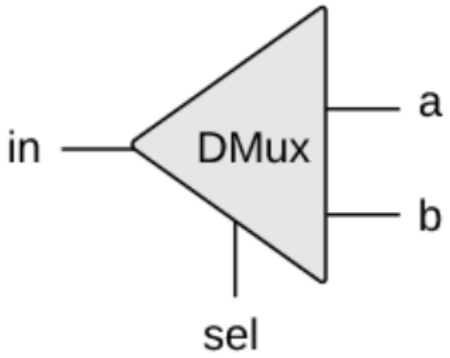
When `sel==1` the gate acts like an Or gate

```
CHIP AndMuxOr {
    IN a, b, sel;
    OUT out;

    PARTS:
    And (a=a, b=b, out=andOut);
    Or  (a=a, b=b, out=orOut);
    Mux (a=andOut, b=orOut, sel=sel, out=out);
}
```

Demux

- Demultiplexor




```
if (sel==0)
   {a,b}={in,0}
else
   {a,b}={0,in}
```

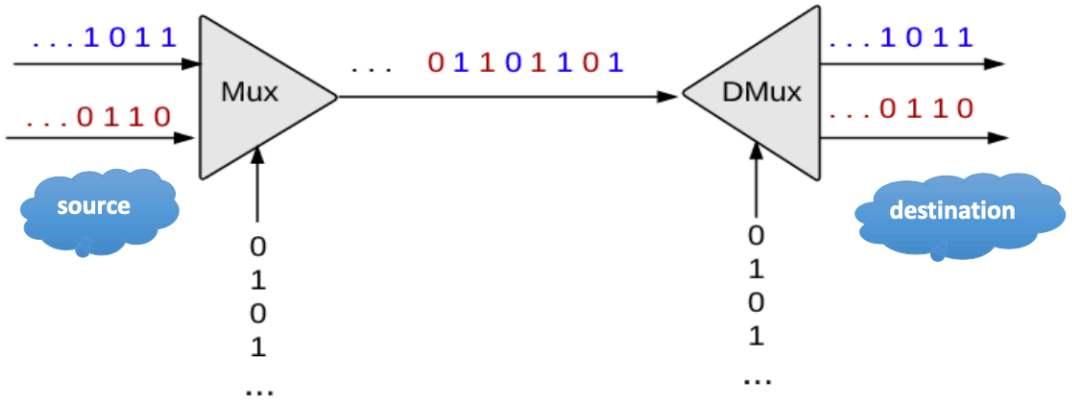
in	sel	a	b
0	0	**0**	**0**
0	1	**0**	**0**
1	0	**1**	**0**
1	1	**0**	**1**

- ทำตรงข้าม Mux
- เข้า 1 ออก 2

DMux.hdl

```
CHIP DMux {
    IN in, sel;
    OUT a, b;

    PARTS:
    // Put your code here:
}
```



- มีข้อจำกัดมีสายเส้นเดียว มีข้อมูล 2 ชุด
- เลือกข้อมูลใดข้อมูลหนึ่ง ในหนึ่งเวลา
- sel ต่อกับสวิจช์ใช้เลือกข้อมูล
- sel = 1 เลือก a , sel = 0 เลือก b

Project 1

Elementary logic gates

- Not
- And
- Or
- Xor
- Mux
- DMux

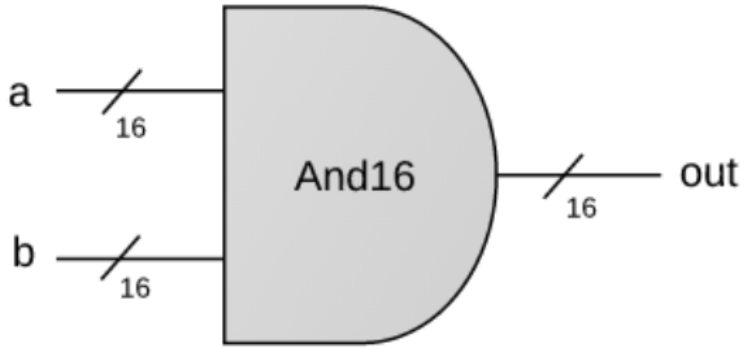
16-bit variants

- Not16
- And16
- Or16
- Mux16

Multi-way variants

- Or8Way
- Mux4Way16
- Mux8Way16
- DMux4Way
- DMux8Way

And16




```
a = 1 0 1 0 1 0 1 1 0 1 0 1 1 1 0 0
b = 0 0 1 0 1 1 0 1 0 0 1 0 1 0 1 0
--------------------------------------
out = 0 0 1 0 1 0 0 1 0 0 0 0 1 0 0 0
```

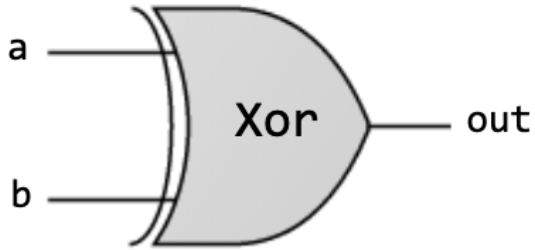
```
CHIP And16 {
    IN a[16], b[16];
    OUT out[16];

    PARTS:
    // Put your code here:
}
```

- ประมวลครั้งละ 16 bit
- (ใช้ multi buses)

Demo : Chip building

- Xor



outputs 1 if a != b

Xor.cmp

```
a b out
0 0  0
0 1  1
1 0  1
1 1  0
```

เมื่อรันโปรแกรม Xor.hdl ในไดเร็กทอรี่ที่มีไฟล์ Xor.tst และ Xor.cmp โปรแกรม Hardware Simulator จะทดสอบการทำงานให้อัตโนมัติ

Xor.hdl

```
CHIP Xor {
     IN  a, b;
     OUT out;

     PARTS:
     // Put your code here.
}
```

Xor.tst

```
load Xor.hdl,
output-file Xor.out,
compare-to Xor.cmp,
output-list a b out;
set a 0, set b 0, eval, output;
set a 0, set b 1, eval, output;
set a 1, set b 0, eval, output;
set a 1, set b 1, eval, output;
```

Not16

```
Chip name: Not16
Inputs:    in[16] // a 16-bit pin
Outputs:   out[16]
Function:  For i=0..15 out[i]=Not(in[i]).
```

And16

```
Chip name: And16
Inputs:    a[16], b[16]
Outputs:   out[16]
Function:  For i=0..15 out[i]=And(a[i],b[i]).
```

Or16

Chip name: Or16
Inputs: a[16], b[16]
Outputs: out[16]
Function: For i=0..15 out[i]=Or(a[i],b[i]).

Mux16

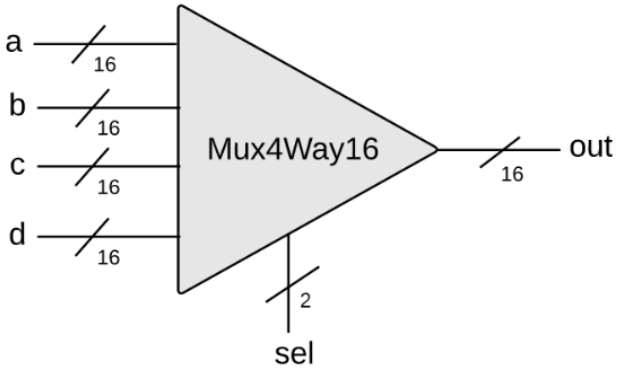
```
Chip name: Mux16
Inputs:    a[16], b[16], sel
Outputs:   out[16]
Function:  If sel=0 then for i=0..15 out[i]=a[i]
           else for i=0..15 out[i]=b[i].
```

Multi-way : Or8Way

```
Chip name: Or8Way
Inputs:    in[8]
Outputs:   out
Function:  out=Or(in[0],in[1],...,in[7]).
```

16-bit, 4-way multiplexor

- Mux4Way16



sel[1]	sel[0]	out
0	0	**a**
0	1	**b**
1	0	**c**
1	1	**d**

Mux4Way16.hdl

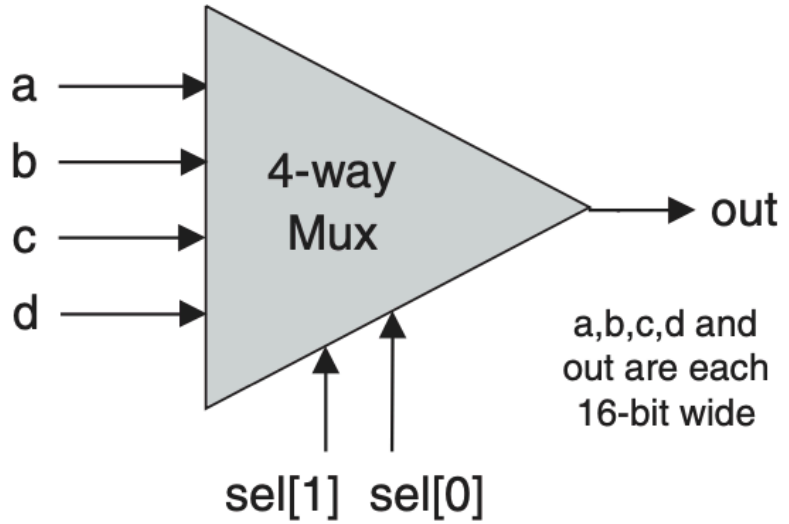
```
CHIP Mux4Way16 {
    IN a[16], b[16], c[16], d[16],
       sel[2];
    OUT out[16];

    PARTS:
    // Put your code here:
}
```

- คำใบ้
  - ใช้ Mux16 หลายตัว

Mux4Way16

sel[1]	sel[0]	out
0	0	a
0	1	b
1	0	c
1	1	d



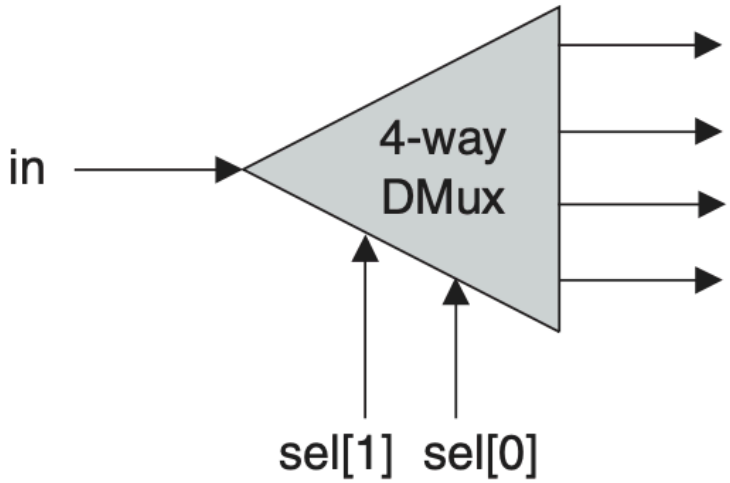
Chip name: Mux4Way16
Inputs: a[16], b[16], c[16], d[16], sel[2]
Outputs: out[16]
Function: If sel=00 then out=a else if sel=01 then out=b else if sel=10 then out=c else if sel=11 then out=d
Comment: The assignment operations mentioned above are all 16-bit. For example, "out=a" means "for i=0..15 out[i]=a[i]".

Mux8Way16

```
Chip name: Mux8Way16
Inputs:    a[16],b[16],c[16],d[16],e[16],f[16],g[16],h[16],
           sel[3]
Outputs:   out[16]
Function:  If sel=000 then out=a else if sel=001 then out=b
           else if sel=010 out=c ... else if sel=111 then out=h
Comment:   The assignment operations mentioned above are all
           16-bit. For example, "out=a" means "for i=0..15
           out[i]=a[i]".
```

DMux4Way

sel[1]	sel[0]	a	b	c	d
0	0	in	0	0	0
0	1	0	in	0	0
1	0	0	0	in	0
1	1	0	0	0	in




```
Chip name: DMux4Way
Inputs:    in, sel[2]
Outputs:   a, b, c, d
Function:  If sel=00 then       {a=in, b=c=d=0}
           else if sel=01 then  {b=in, a=c=d=0}
           else if sel=10 then  {c=in, a=b=d=0}
           else if sel=11 then  {d=in, a=b=c=0}.
```

DMux8Way

```
Chip name: DMux8Way
Inputs:    in, sel[3]
Outputs:   a, b, c, d, e, f, g, h
Function:  If sel=000 then       {a=in, b=c=d=e=f=g=h=0}
           else if sel=001 then {b=in, a=c=d=e=f=g=h=0}
           else if sel=010 ...
           ...
           else if sel=111 then {h=in, a=b=c=d=e=f=g=0}.
```