ECE 27000 S22 P 9

Chapter 7. More Combinational Building Blocks

7.2. Priority Encoding

When multiple devices are connected to a bus, only one can write into the bus at any time.

In general, in a computer system, it is possible for multiple devices to request service from the same device at the same time.
Only one of them should be serviced at any time.

To decide which device to service, the devices are given priorities.

A priority encoder accepts requests for service from multiple devices.

Each device is connected to an input of the priority encoder.
The device asserts the input associated with it to request service.

The priority encoder produces the number corresponding to the device that should be serviced.
This is the device with the highest priority that is requesting service.

Truth table of a 4-input priority encoder:
The priorities (from high to low): I3, I2, I1, I0.

I3	I2	I1	I0	IDLE	Y1	Y0
0	0	0	0	1	x	x
0	0	0	1	0	0	0
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	0	1	0
0	1	0	1	0	1	0
0	1	1	0	0	1	0
0	1	1	1	0	1	0
1	0	0	0	0	1	1
1	0	0	1	0	1	1
1	0	1	0	0	1	1
1	0	1	1	0	1	1
1	1	0	0	0	1	1
1	1	0	1	0	1	1
1	1	1	0	0	1	1
1	1	1	1	0	1	1

To write the functions for Y1 and Y0 it is convenient to partition the truth table as follows.

	I3	I2	I1	I0	IDLE	Y1	Y0
	0	0	0	0	1	x	x
H0	0	0	0	1	0	0	0
H1	0	0	1	0	0	0	1
	0	0	1	1	0	0	1
H2	0	1	0	0	0	1	0
	0	1	0	1	0	1	0
	0	1	1	0	0	1	0
	0	1	1	1	0	1	0
H3	1	0	0	0	0	1	1
	1	0	0	1	0	1	1
	1	0	1	0	0	1	1
	1	0	1	1	0	1	1
	1	1	0	0	0	1	1
	1	1	0	1	0	1	1
	1	1	1	0	0	1	1
	1	1	1	1	0	1	1

$Hi = 1$ when Ii is the highest priority input equal to 1.

$H3 = I3$
$H2 = I3' \cdot I2$
$H1 = I3' \cdot I2' \cdot I1$
$H0 = I3' \cdot I2' \cdot I1' \cdot I0$

$Y1 = H3 + H2 = I3 + I3' \cdot I2$
$Y0 = H3 + H1 = I3 + I3' \cdot I2' \cdot I1$
$IDLE = I3' \cdot I2' \cdot I1' \cdot I0'$

An 8-input priority encoder:

$H7 = I7$ (outputs 111)

$H6 = I7' \cdot I6$ (outputs 110)

$H5 = I7' \cdot I6' \cdot I5$ (outputs 101)

$H4 = I7' \cdot I6' \cdot I5' \cdot I4$ (outputs 100)

$H3 = I7' \cdot I6' \cdot I5' \cdot I4' \cdot I3$ (outputs 011)

$H2 = I7' \cdot I6' \cdot I5' \cdot I4' \cdot I3' \cdot I2$ (outputs 010)

$H1 = I7' \cdot I6' \cdot I5' \cdot I4' \cdot I3' \cdot I2' \cdot I1$
(outputs 001)

$H0 = I7' \cdot I6' \cdot I5' \cdot I4' \cdot I3' \cdot I2' \cdot I1' \cdot I0$
(outputs 000)

$Y2 = H7 + H6 + H5 + H4$

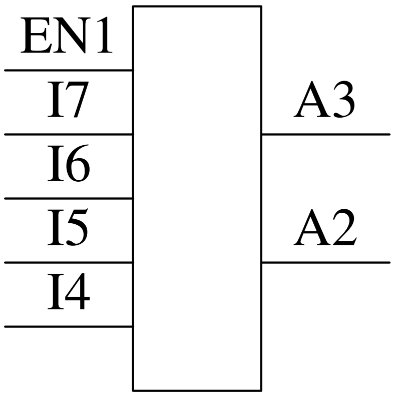
$Y1 = H7 + H6 + H3 + H2$

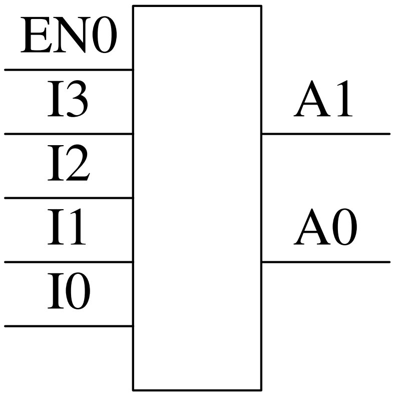
$Y0 = H7 + H5 + H3 + H1$

$IDLE = I7' \cdot I6' \cdot I5' \cdot I4' I3' \cdot I2' \cdot I1' \cdot I0'$

Cascading Priority Encoders

Suppose that two groups of four devices issue requests for the same resource.
Each group is handled by one priority encoder.
The outputs A3A2 and A1A0 need to be combined into outputs R2R1R0 of an 8-input priority encoder.





The upper priority encoder needs to produce an output only if
$I7 = 1$ or $I6 = 1$ or $I5 = 1$ or $I4 = 1$.
The lower priority encoder needs to produce an output only if
$I7 = 0$ and $I6 = 0$ and $I5 = 0$ and $I4 = 0$.
$EN1 = I7 + I6 + I5 + I4$
$EN0 = I7' \cdot I6' \cdot I5' \cdot I4'$

If $EN1 = 1$:
$R2R1R0 = 1A3A2 = EN1A3A2$
$A1 = 0$ and $A0 = 0$

If $EN1 = 0$:
$R2R1R0 = 0A1A0 = EN1A1A0$
$A3 = 0$ and $A2 = 0$

$R2 = EN1$
$R1 = A3 + A1$
$R0 = A2 + A0$

7.3. Exclusive-OR (XOR) Gates and Parity Functions

XOR gates are important for many applications, among them:

• Comparison.
An XOR gate can compare two bits for equality.
The outputs of several XOR gates can be combined to check equality between multi-bit numbers.

• Partity generation and checking.
An XOR function calculates the sum module 2, or parity, of the inputs.
Error detecting and correcting codes are based on parity checking.

• Addition.
XOR functions are used to form the sum bits in addition.

• Counting.
Sequential circuits called binary counters use XOR gates to form the next value of each bit.

• Random number generation.
Linear sequential circuits called linear-feedback shift registers (*LFSR*s) consist of flip-flops and XOR gates.
LFSRs can be used as random number generators.

An $Exclusive - OR$ (XOR) gate is a two-input gate whose output is 1 if exactly one of its inputs is 1.
An XOR gate produces a 1 output if its inputs are different.
$X \oplus Y = X' \cdot Y + X \cdot Y'$.

An $Exclusive - NOR$ $(XNOR)$ gate is a two-input gate whose output is 1 if its inputs have the same value.
$(X \oplus Y)' = X \cdot Y + X' \cdot Y'$.

Truth tables:

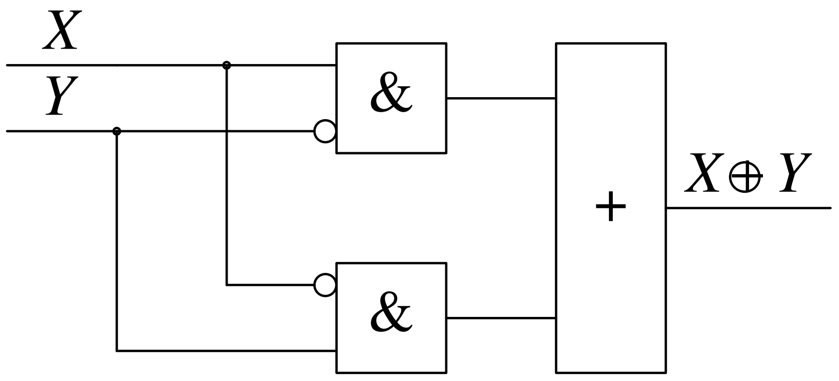
X	Y	$X \oplus Y$	$(X \oplus Y)'$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

K-map of $X \oplus Y$:

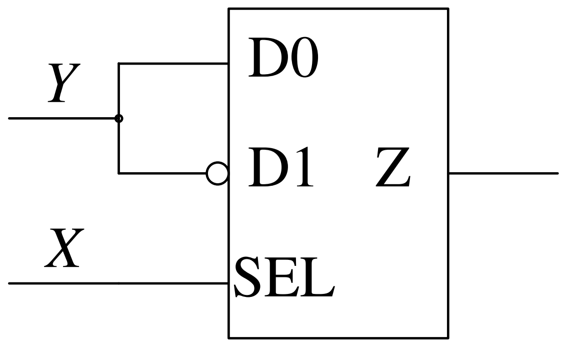
		X	
		0	1
Y	1	1	0
	0	0	1

Exclusive OR is not a basic function of switching algebra.
Most technologies cannot perform an XOR function directly.
They implement XOR gates using multiple basic gates.

Implementation using basic gates:



Implementation using a MUX:



$Z = X \cdot Y' + X' \cdot Y$

Properties of XOR gates:

$X \oplus 0 = X$

$X \oplus 1 = X'$

$(X \oplus Y)' = X' \oplus Y = X \oplus Y'$

$X \oplus Y = X' \oplus Y'$

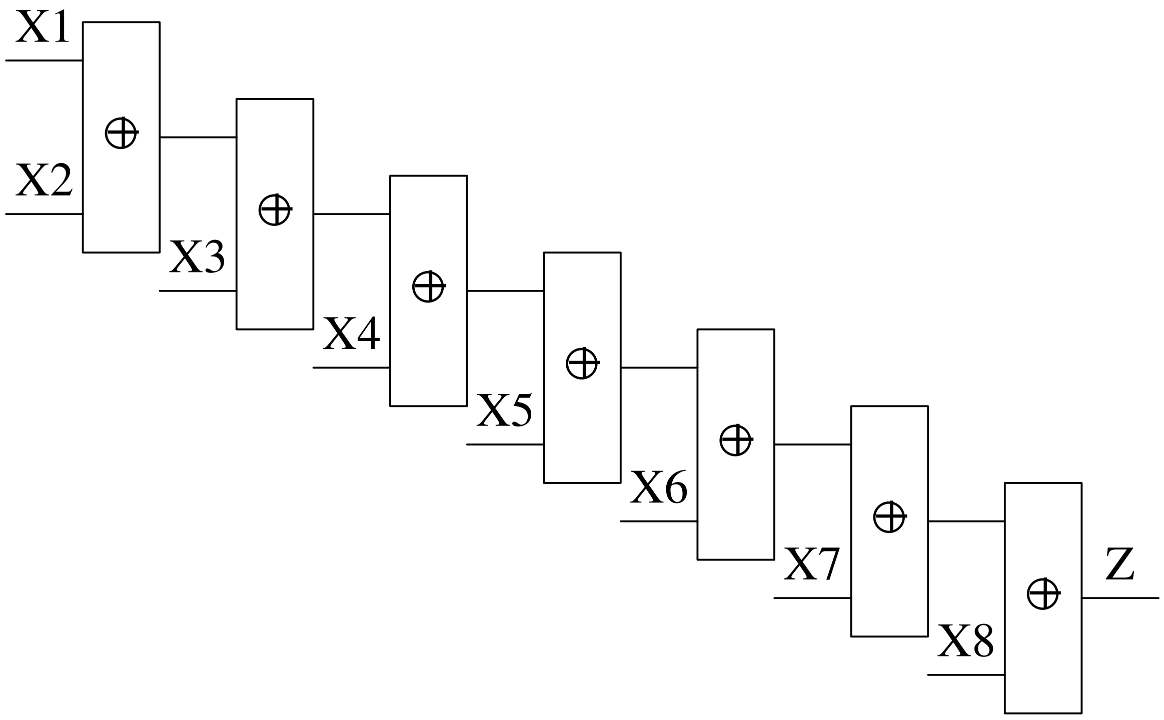
$X \oplus Y = Y \oplus X$

$(X \oplus Y) \oplus Z = X \oplus (Y \oplus Z) = X \oplus Y \oplus Z$

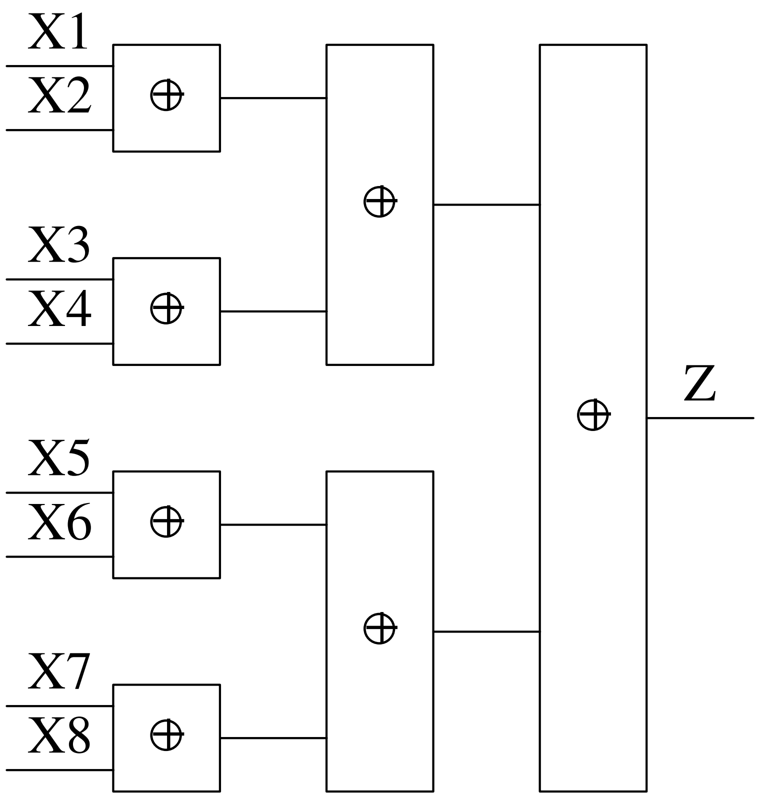
An *n*-variable XOR function can be implemented as a cascade or as a tree.

$$Z = X1 \oplus X2 \oplus X3 \oplus X4 \oplus X5 \oplus X6 \oplus X7 \oplus X8$$

$$Z = (((((((X1 \oplus X2) \oplus X3) \oplus X4) \oplus X5) \oplus X6) \oplus$$
$$\oplus X7) \oplus X8) =$$



$$Z = (X1 \oplus X2) \oplus (X3 \oplus X4) \oplus (X5 \oplus X6) \oplus$$
$$\oplus (X7 \oplus X8)$$



The output of an n-input XOR function is 1 iff an odd number of variables are 1.
Expalantion:
Arrange the variables equal to 1 in pairs, leaving one 1 variable alone.
For the pairs use $1 \oplus 1 = 0$.
Use $0 \oplus 0 = 0$ to cancel all the 0s.
A 1 is left.

K-map of $W \oplus X \oplus Y \oplus Z$:

		WX			
		00	01	11	10
YZ	00		1		1
	01	1		1	
	11		1		1
	10	1		1	