



CprE 281: Digital Logic

Instructor: Alexander Stoytchev

<http://www.ece.iastate.edu/~alexs/classes/>

Design Examples

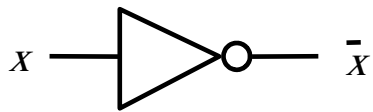
CprE 281: Digital Logic
Iowa State University, Ames, IA
Copyright © Alexander Stoytchev

Administrative Stuff

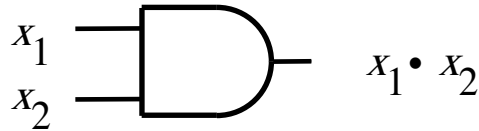
- **HW3 is out**
- **It is due on Monday Sep 10 @ 4pm.**
- **Please write clearly on the first page (in BLOCK CAPITAL letters) the following three things:**
 - **Your First and Last Name**
 - **Your Student ID Number**
 - **Your Lab Section Letter**
- **Also, please**
 - **Staple your pages**

Quick Review

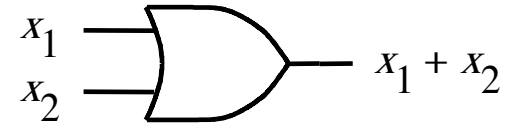
The Three Basic Logic Gates



NOT gate

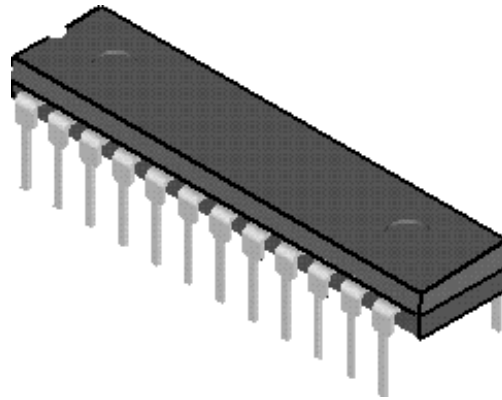


AND gate

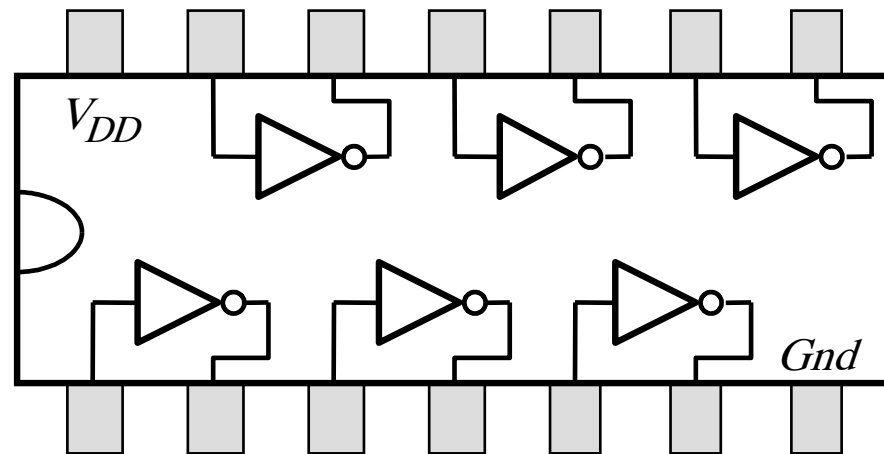


OR gate

You can build any circuit using only these three gates



(a) Dual-inline package



(b) Structure of 7404 chip

Figure B.21. A 7400-series chip.

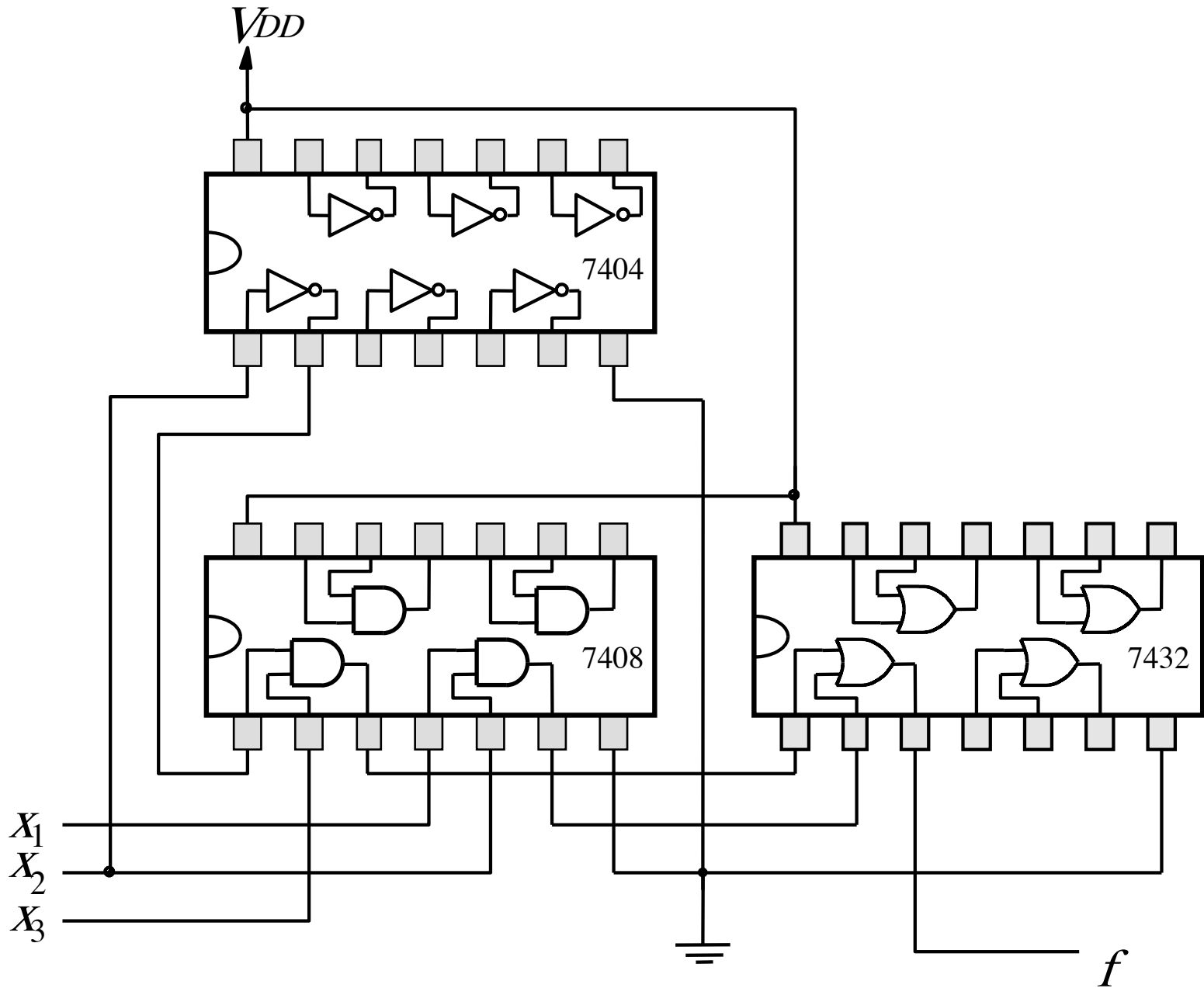
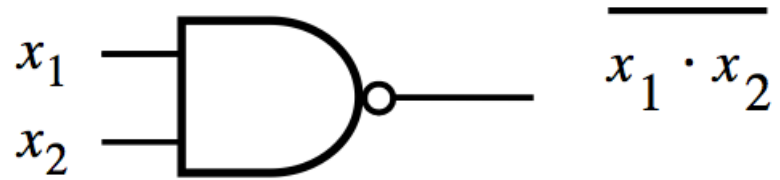


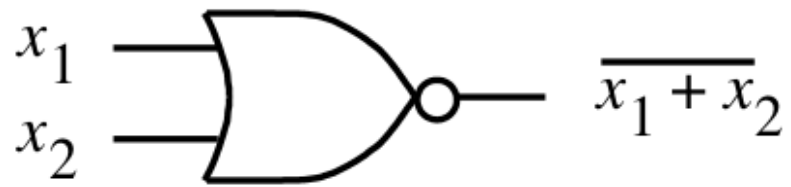
Figure B.22. An implementation of $f = x_1x_2 + \bar{x}_2x_3$.

NAND Gate



x_1	x_2	f
0	0	1
0	1	1
1	0	1
1	1	0

NOR Gate



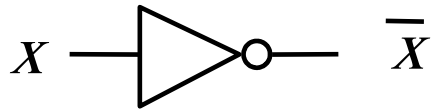
x_1	x_2	f
0	0	1
0	1	0
1	0	0
1	1	0

Why do we need two more gates?

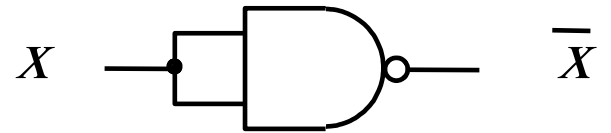
They can be implemented with fewer transistors.

(more about this later)

Building a NOT Gate with NAND



x	\bar{x}
0	1
1	0



x	x	f
0	0	1
1	1	0

impossible combinations

Thus, the two truth tables are equal!

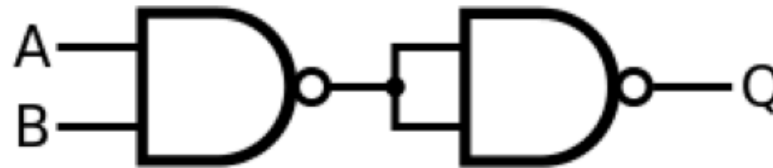
Building an AND gate with NAND gates

Desired AND Gate



$$Q = A \text{ AND } B$$

NAND Construction



$$= \text{NOT}(\text{NOT}(A \text{ AND } B))$$

Truth Table

Input A	Input B	Output Q
0	0	0
0	1	0
1	0	0
1	1	1

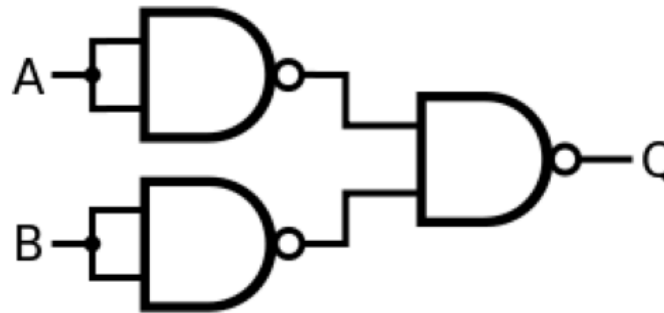
Building an OR gate with NAND gates

Desired OR Gate



$$Q = A \text{ OR } B$$

NAND Construction



$$= \text{NOT} [\text{NOT}(A \text{ AND } A) \text{ AND } \text{NOT}(B \text{ AND } B)]$$

Truth Table

Input A	Input B	Output Q
0	0	0
0	1	1
1	0	1
1	1	1

Implications

**Any Boolean function can be implemented
with only NAND gates!**

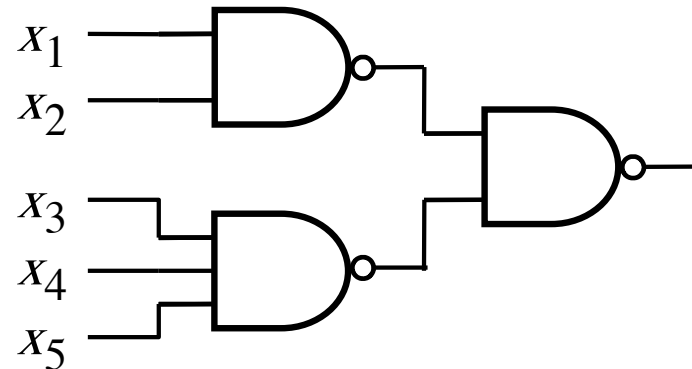
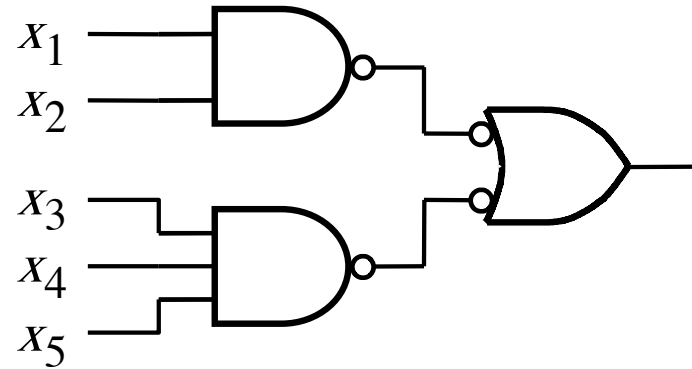
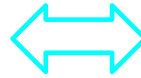
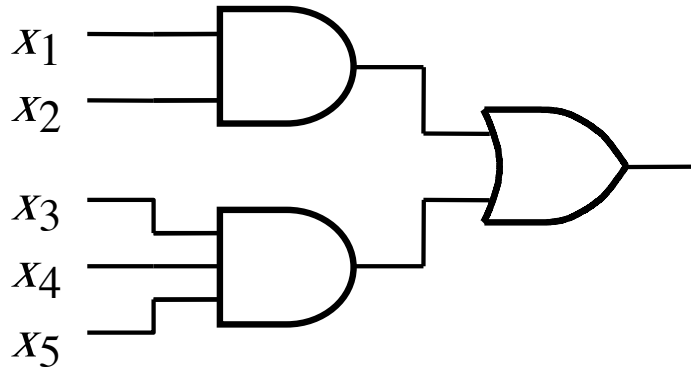
Implications

**Any Boolean function can be implemented
with only NAND gates!**

The same is also true for NOR gates!

NAND-NAND Implementation of Sum-of-Products Expressions

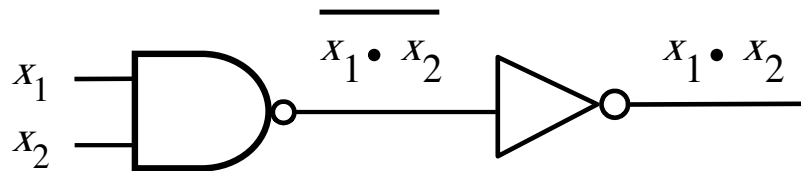
Sum-Of-Products



This circuit uses ANDs & OR

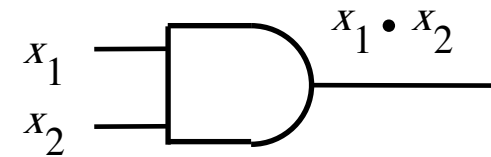
This circuit uses only NANDs

NAND followed by NOT = AND



x_1	x_2	f
0	0	1
0	1	1
1	0	1
1	1	0

f
0
0
0
1



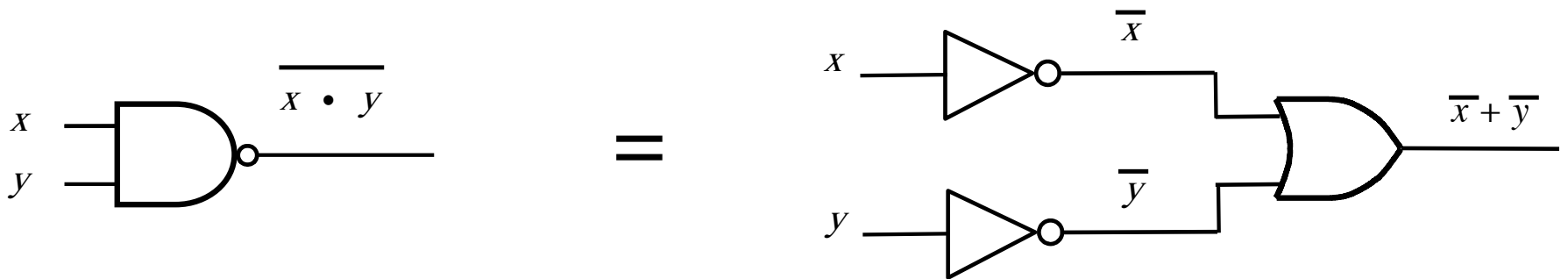
x_1	x_2	f
0	0	0
0	1	0
1	0	0
1	1	1

DeMorgan's Theorem

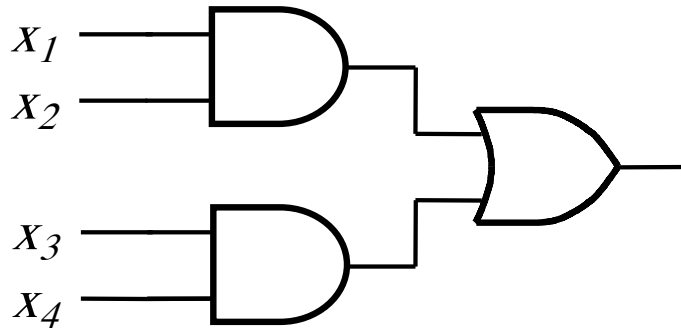
15a. $\overline{\mathbf{x} \cdot \mathbf{y}} = \overline{\mathbf{x}} + \overline{\mathbf{y}}$

DeMorgan's Theorem

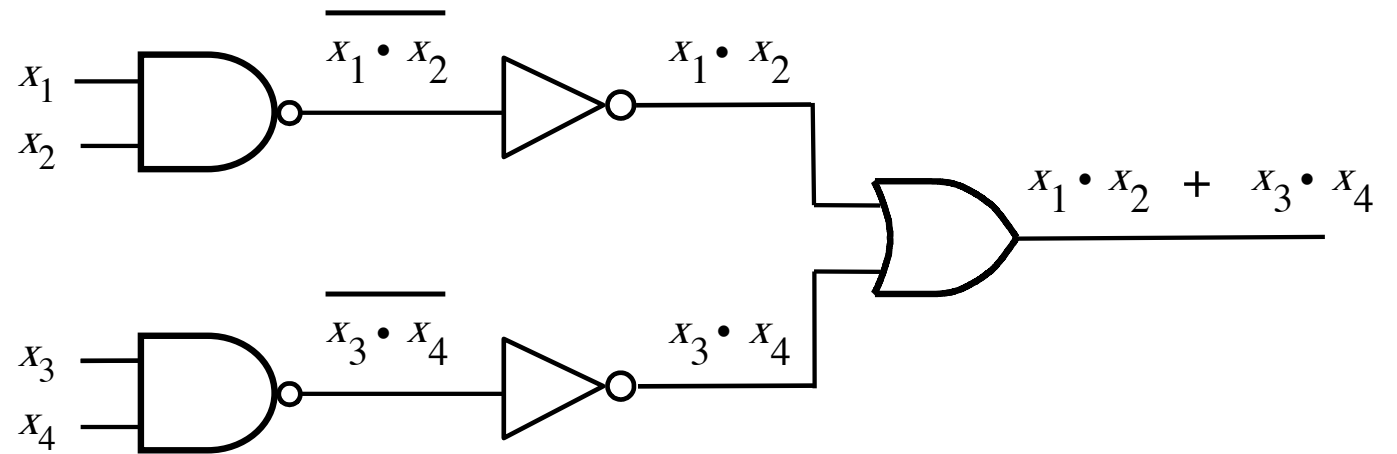
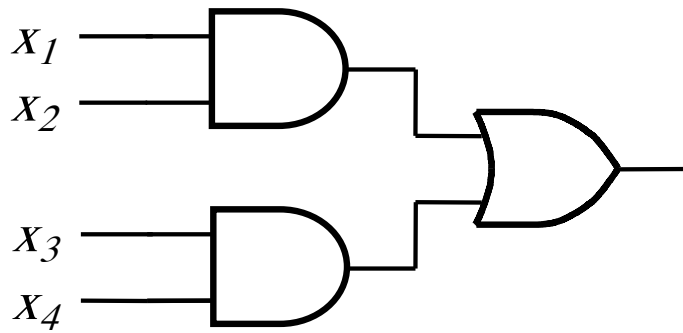
15a. $\overline{x \cdot y} = \bar{x} + \bar{y}$



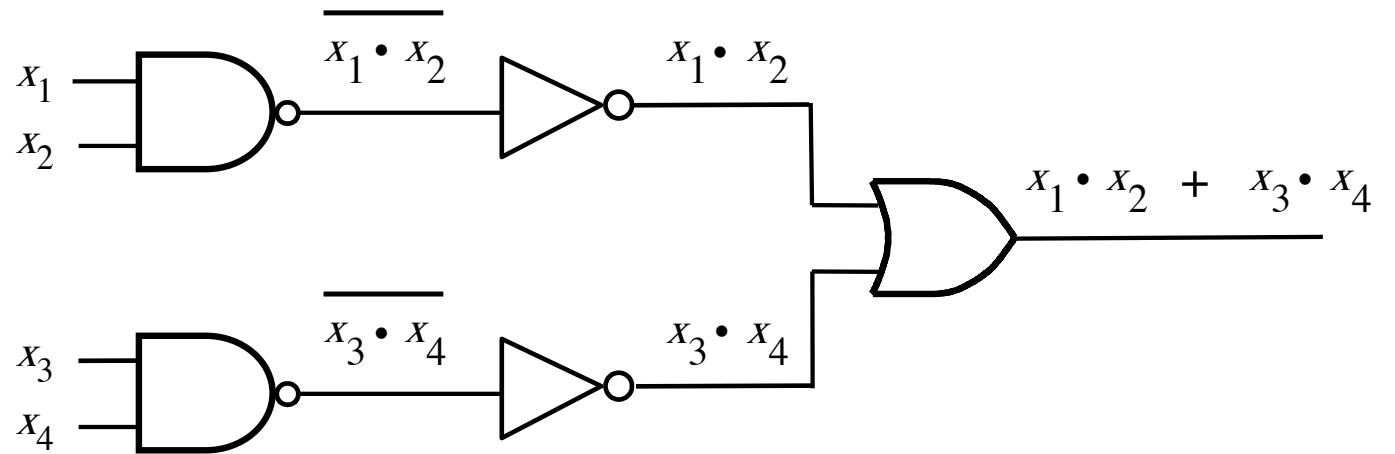
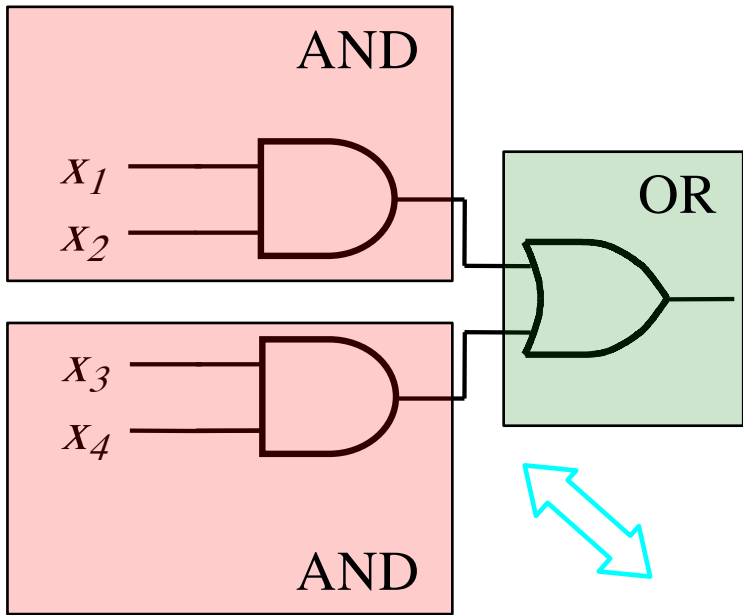
Sum-Of-Products



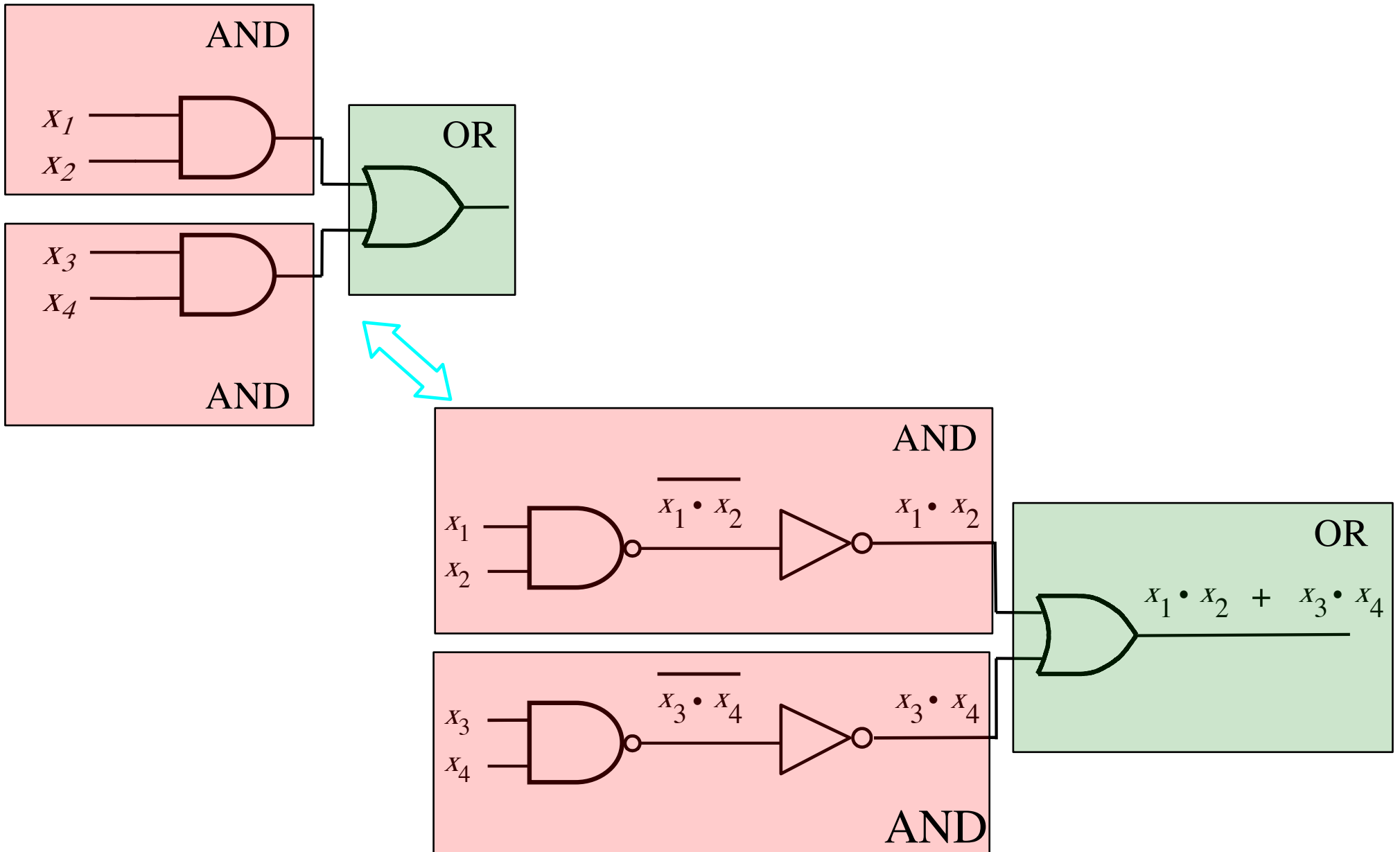
Sum-Of-Products



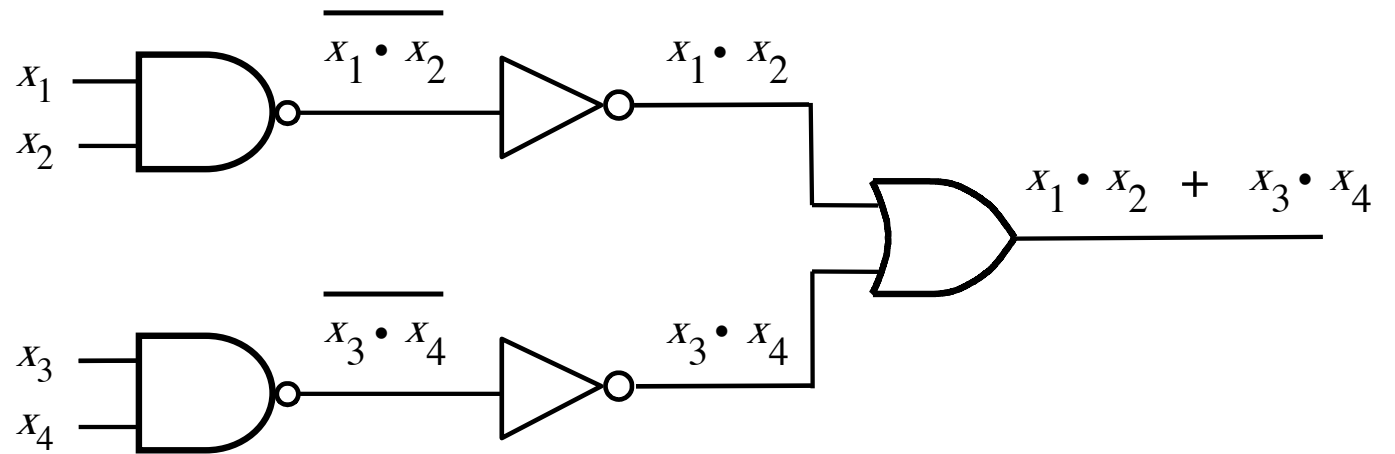
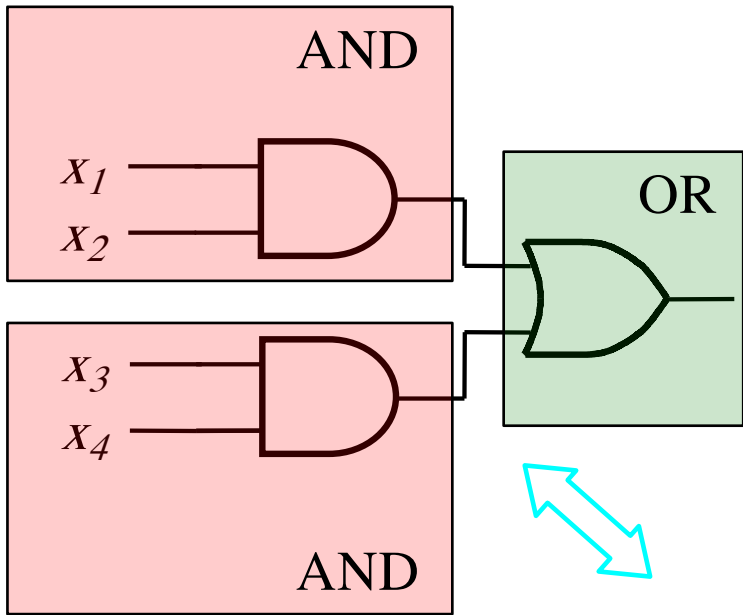
Sum-Of-Products



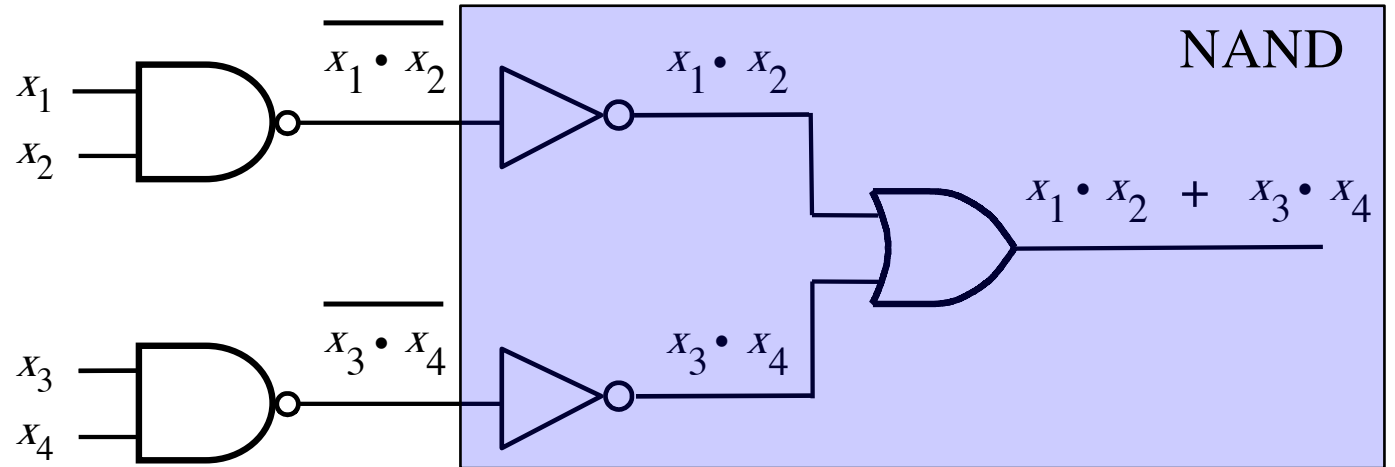
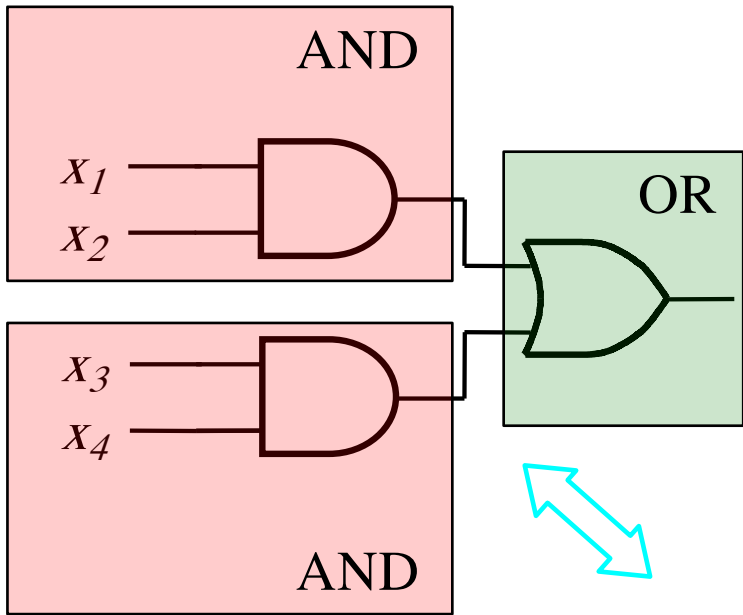
Sum-Of-Products



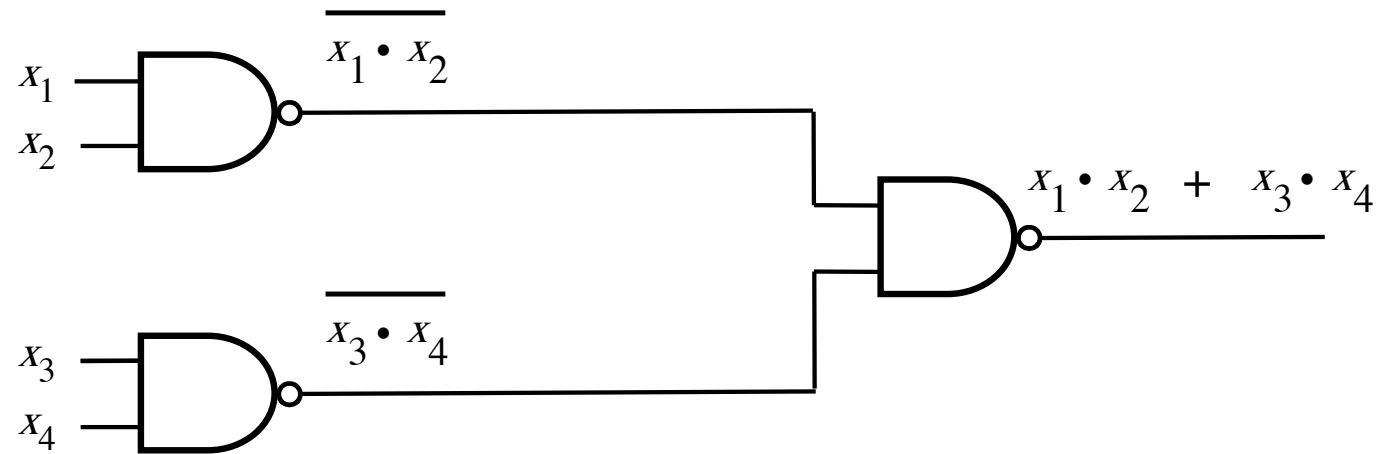
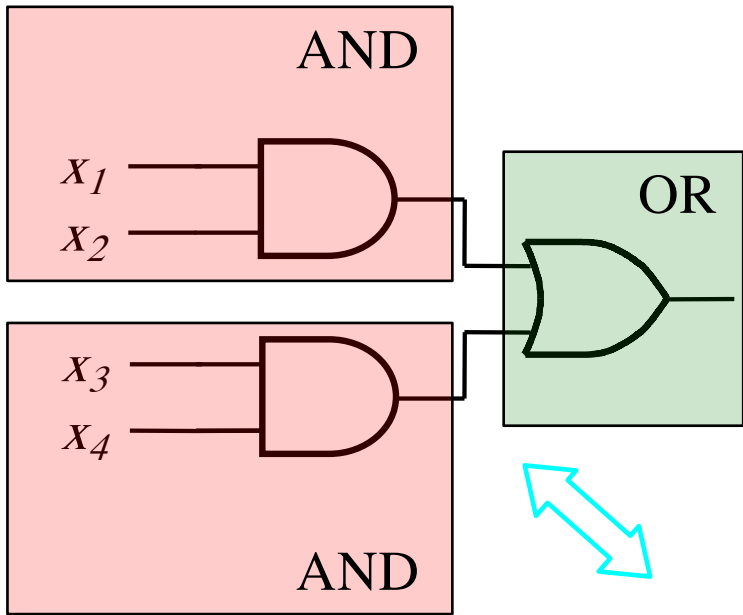
Sum-Of-Products



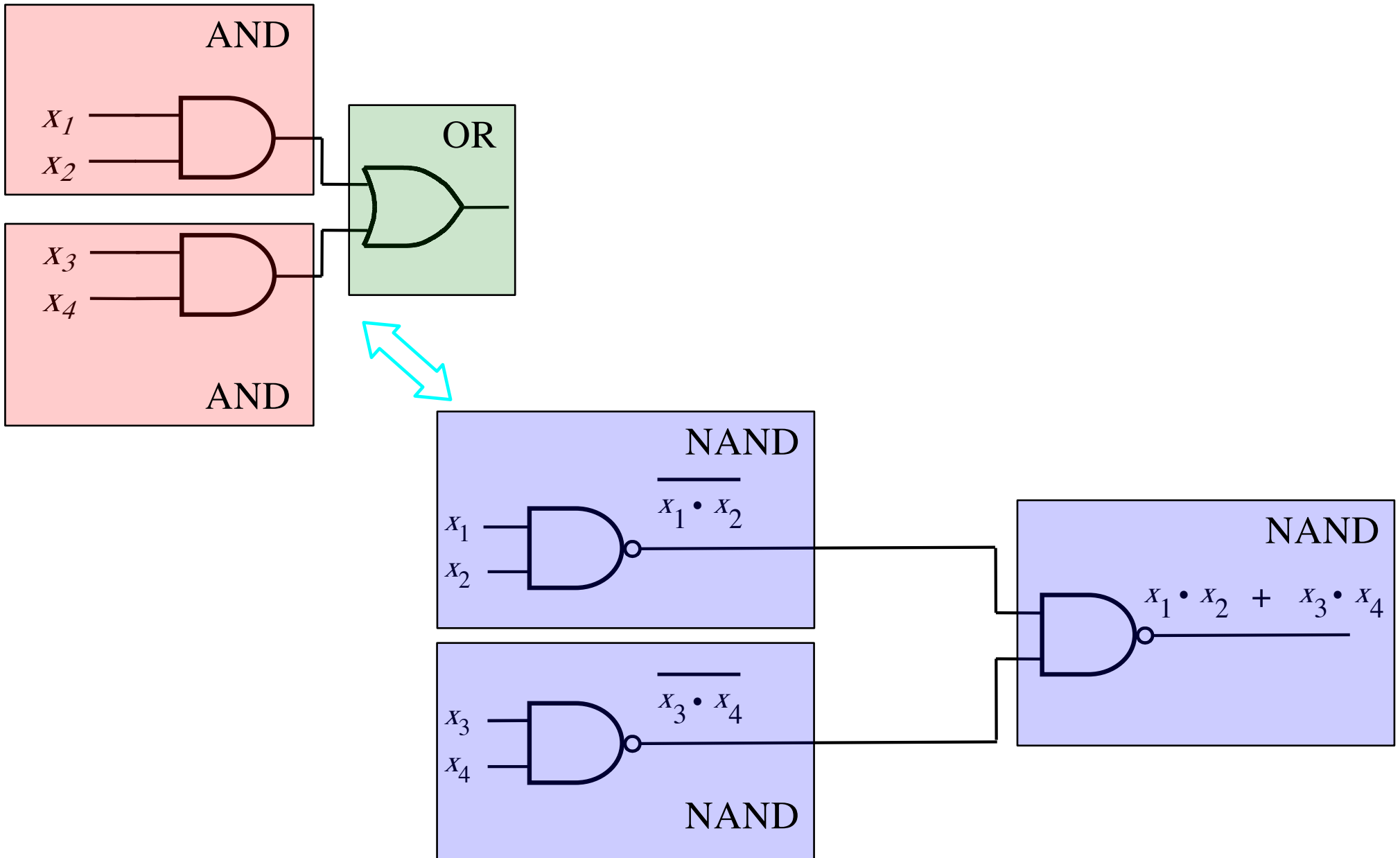
Sum-Of-Products



Sum-Of-Products

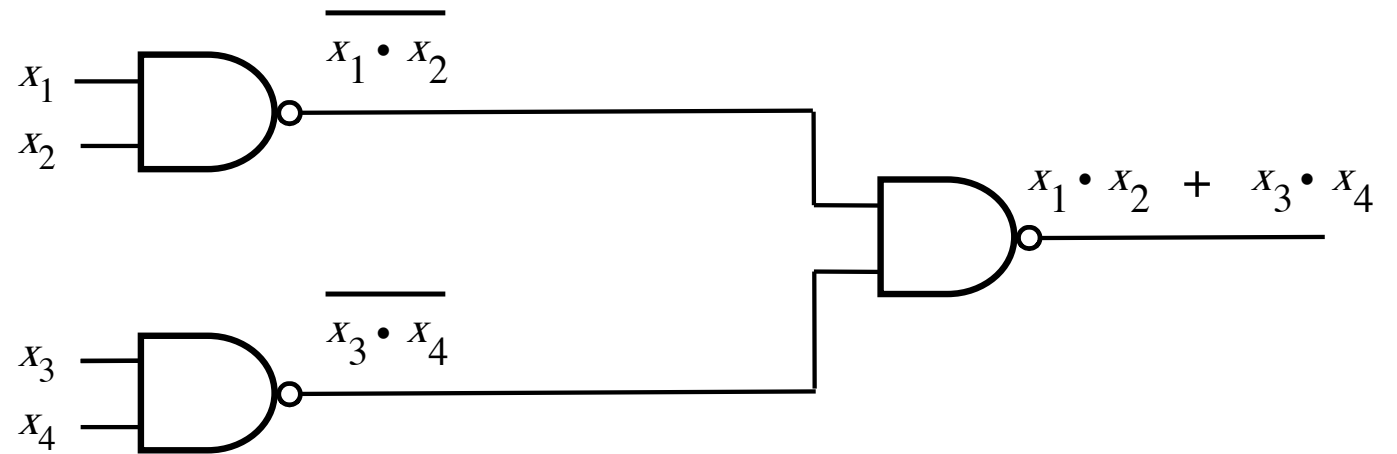
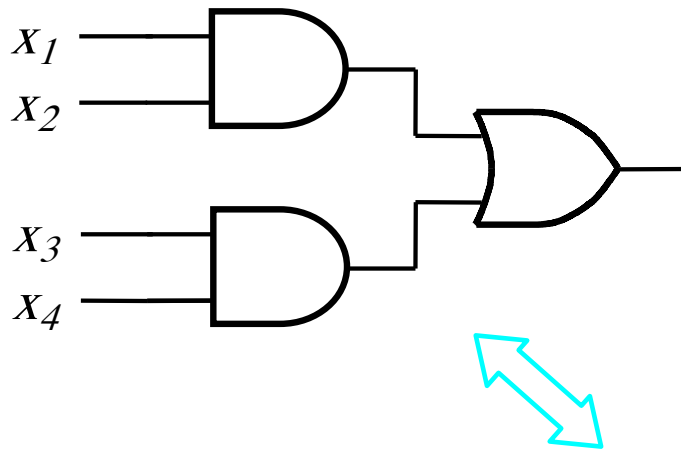


Sum-Of-Products



This circuit uses only NANDs

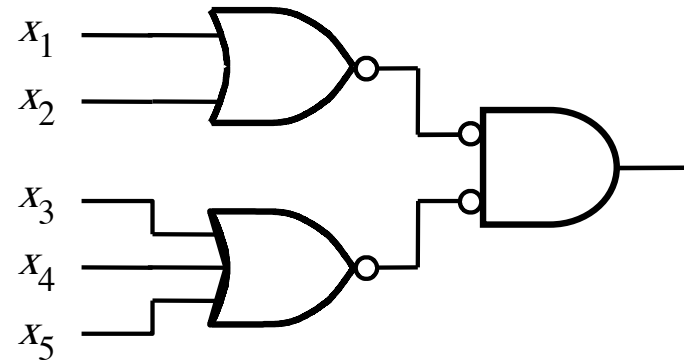
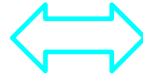
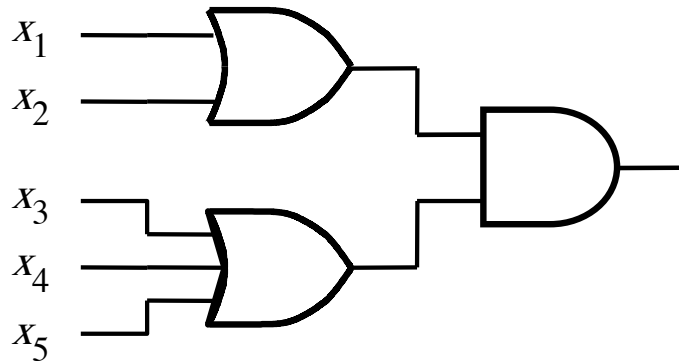
Sum-Of-Products



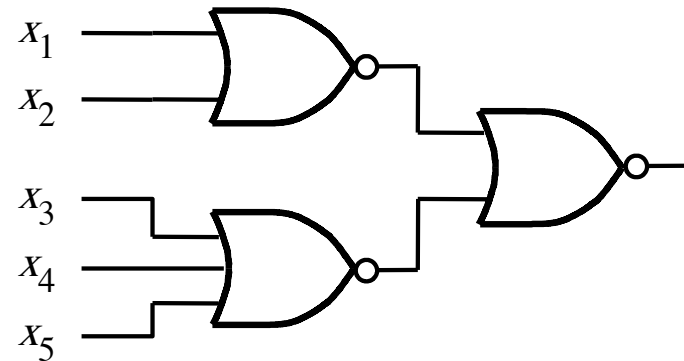
This circuit uses only NANDs

NOR-NOR Implementation of Product-of-Sums Expressions

Product-Of-Sums

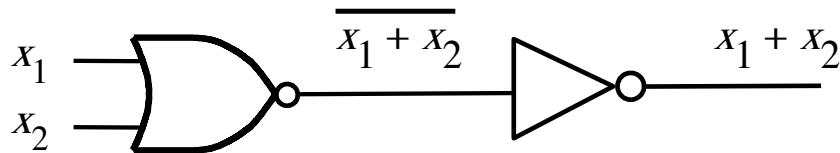


This circuit uses ORs & AND



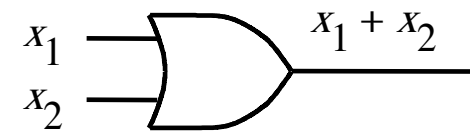
This circuit uses only NORs

NOR followed by NOT = OR



x_1	x_2	f
0	0	1
0	1	0
1	0	0
1	1	0

f
0
1
1
1



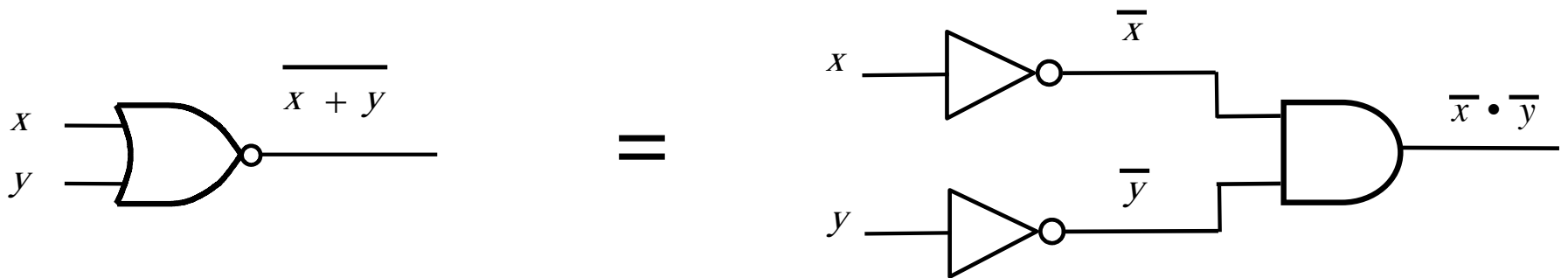
x_1	x_2	f
0	0	0
0	1	1
1	0	1
1	1	1

DeMorgan's Theorem

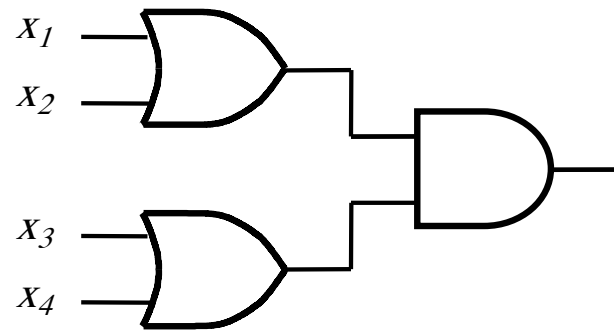
15b. $\overline{x + y} = \bar{x} \cdot \bar{y}$

DeMorgan's Theorem

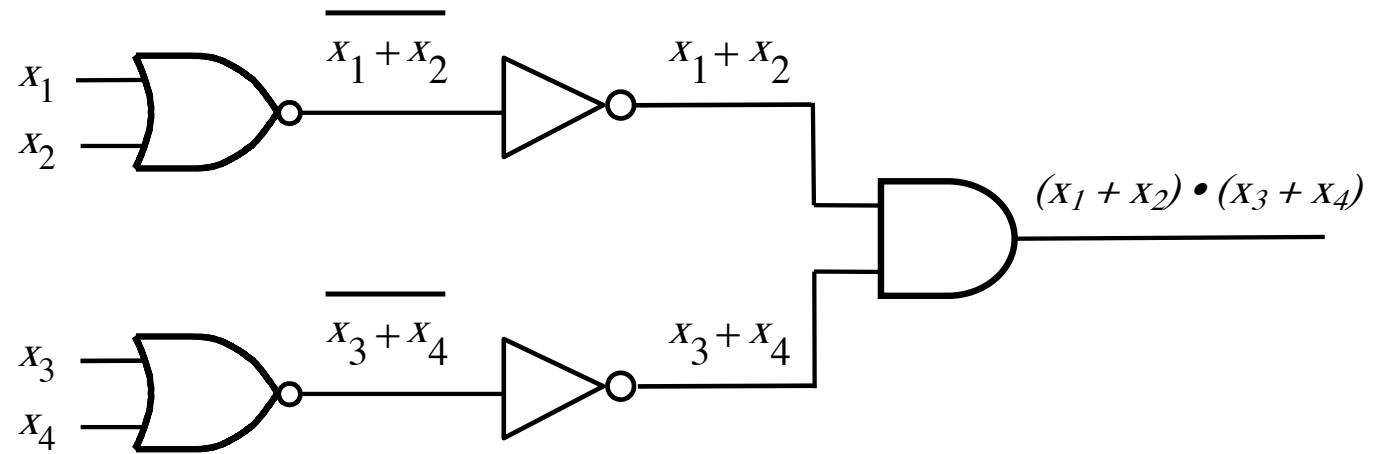
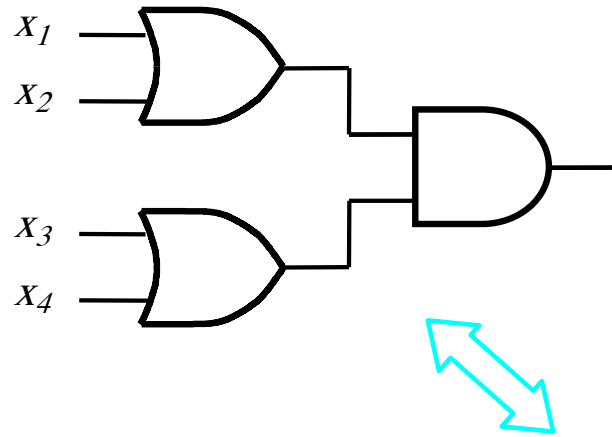
15b. $\overline{x + y} = \bar{x} \cdot \bar{y}$



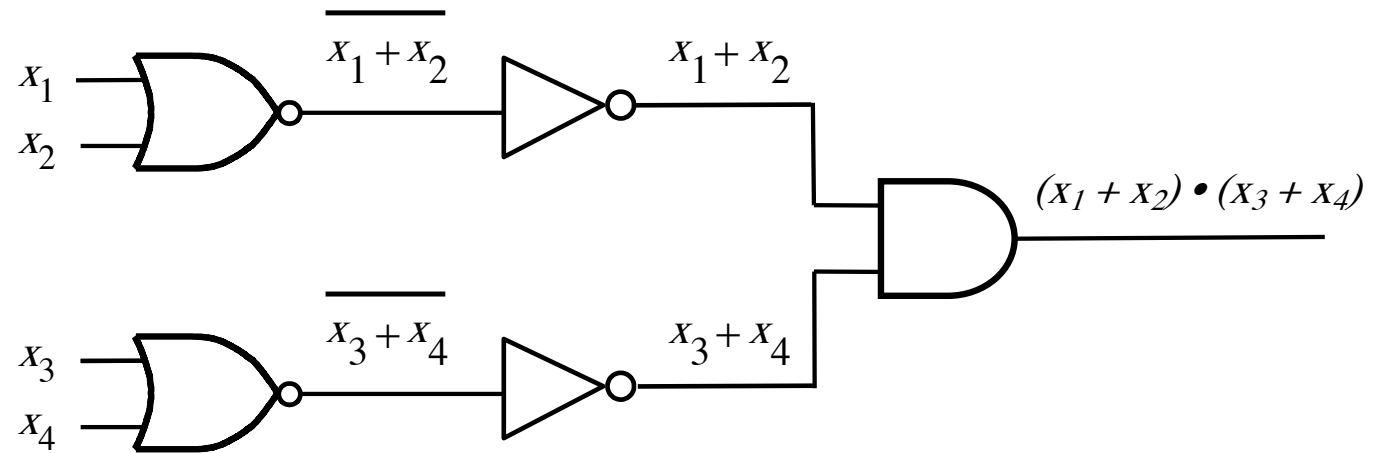
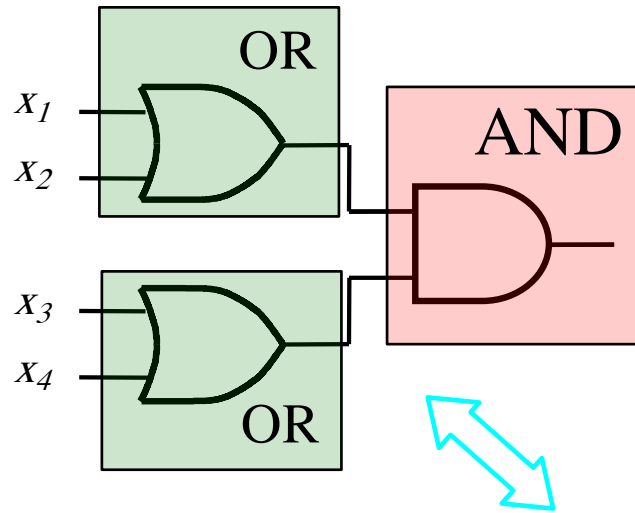
Product-Of-Sums



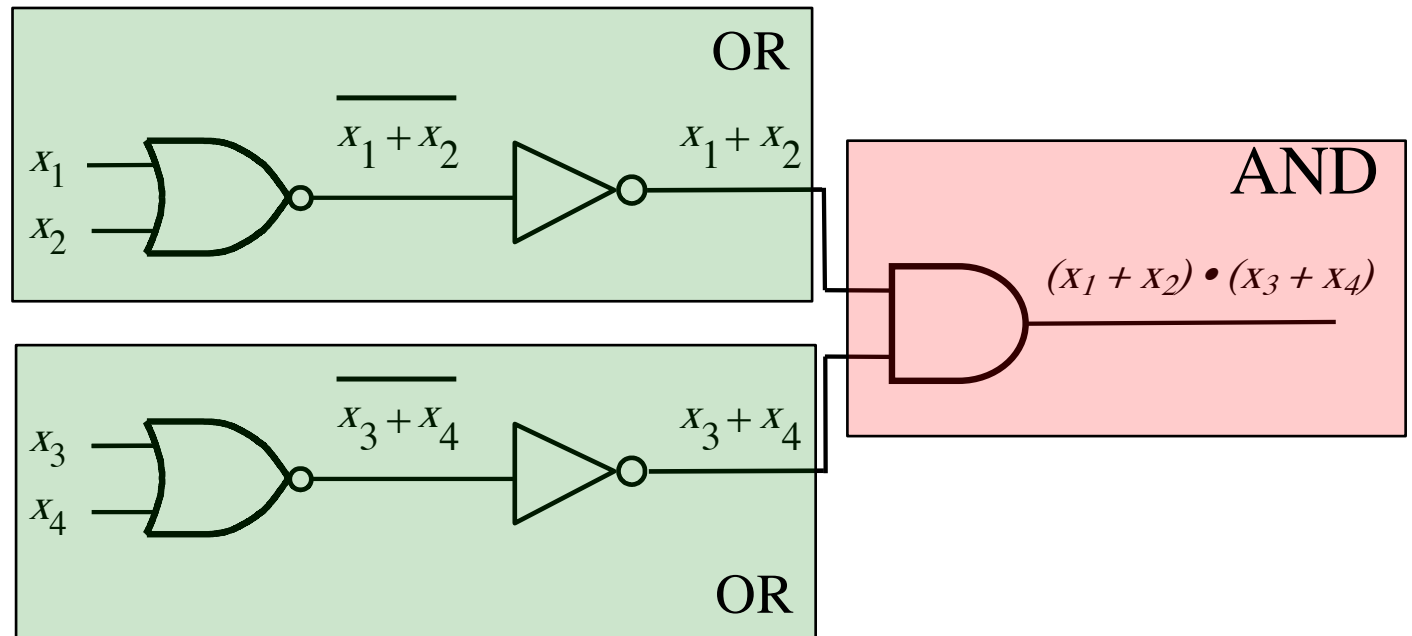
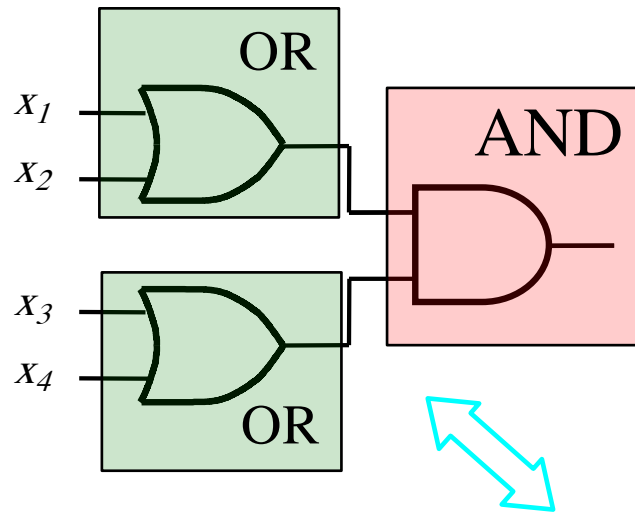
Product-Of-Sums



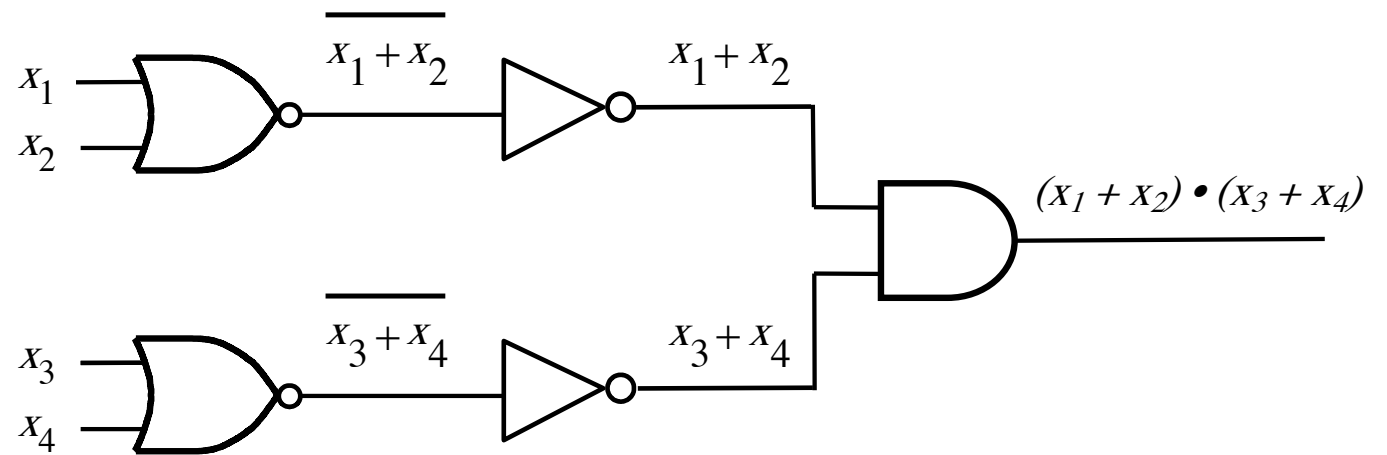
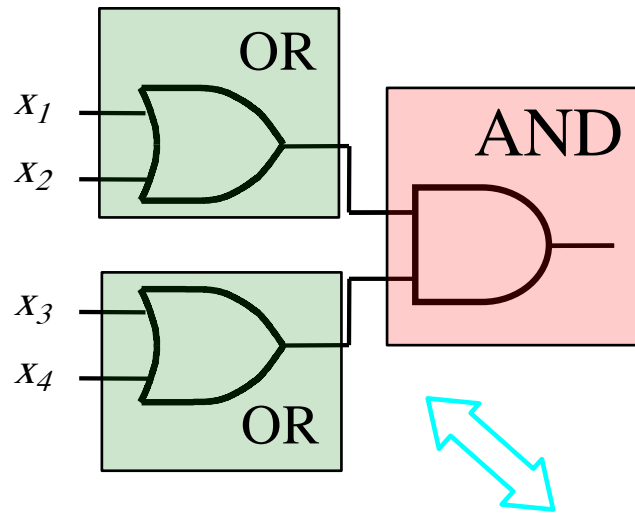
Product-Of-Sums



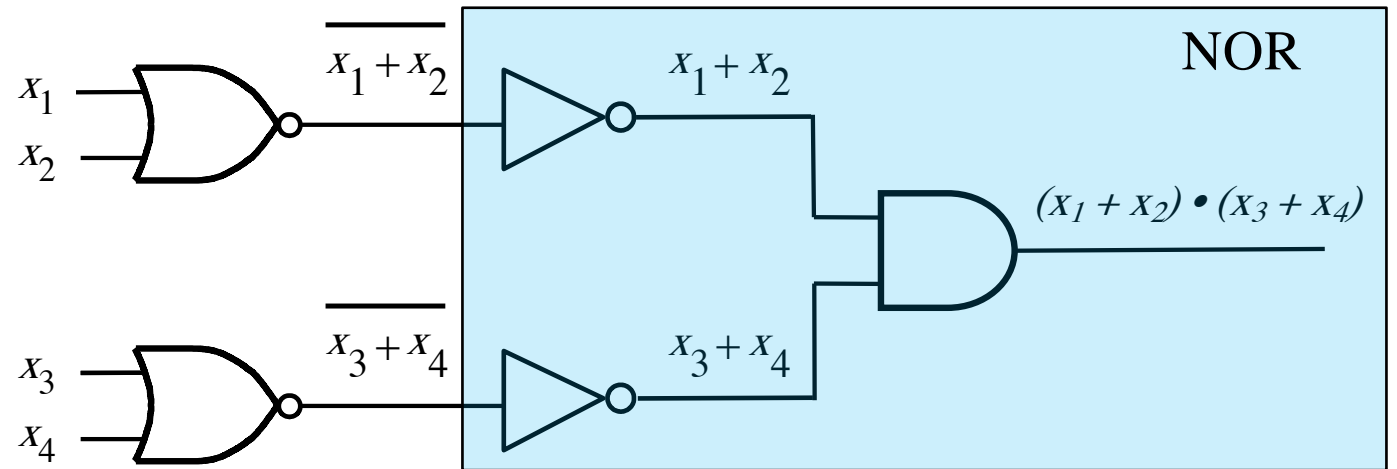
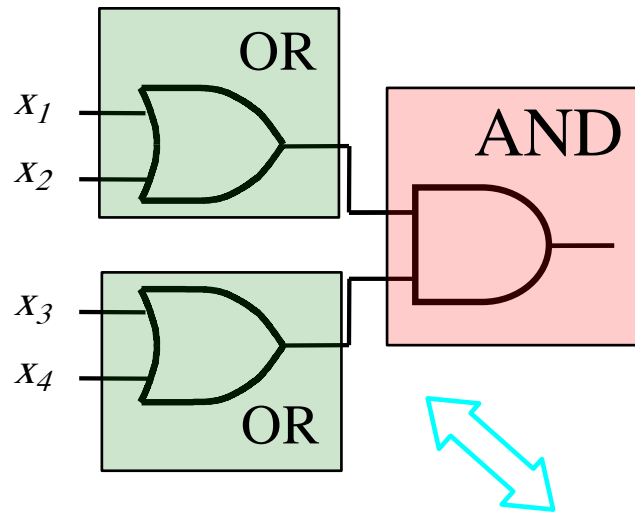
Product-Of-Sums



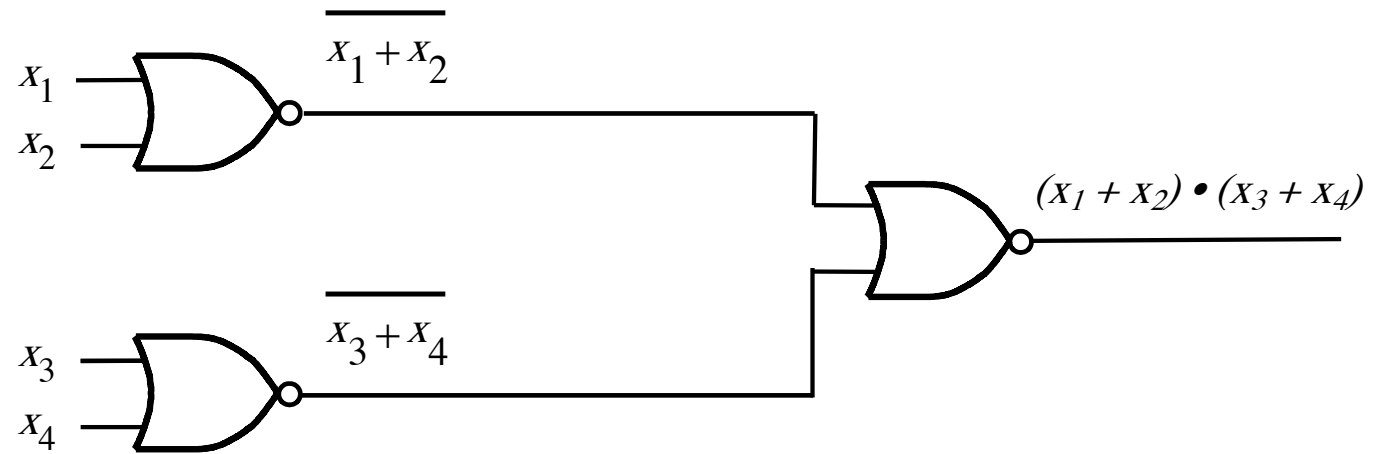
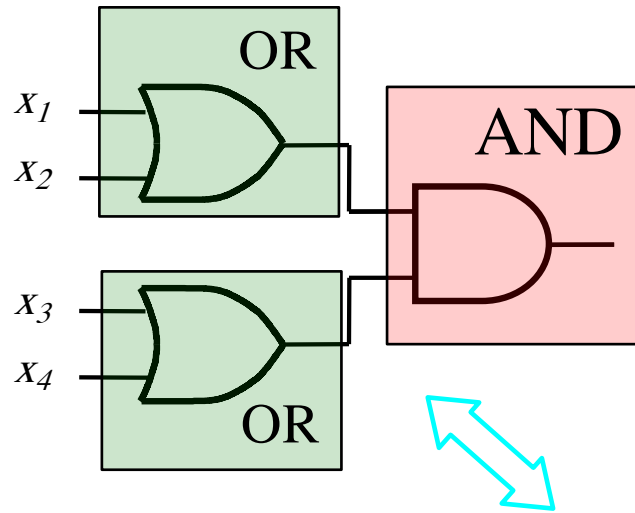
Product-Of-Sums



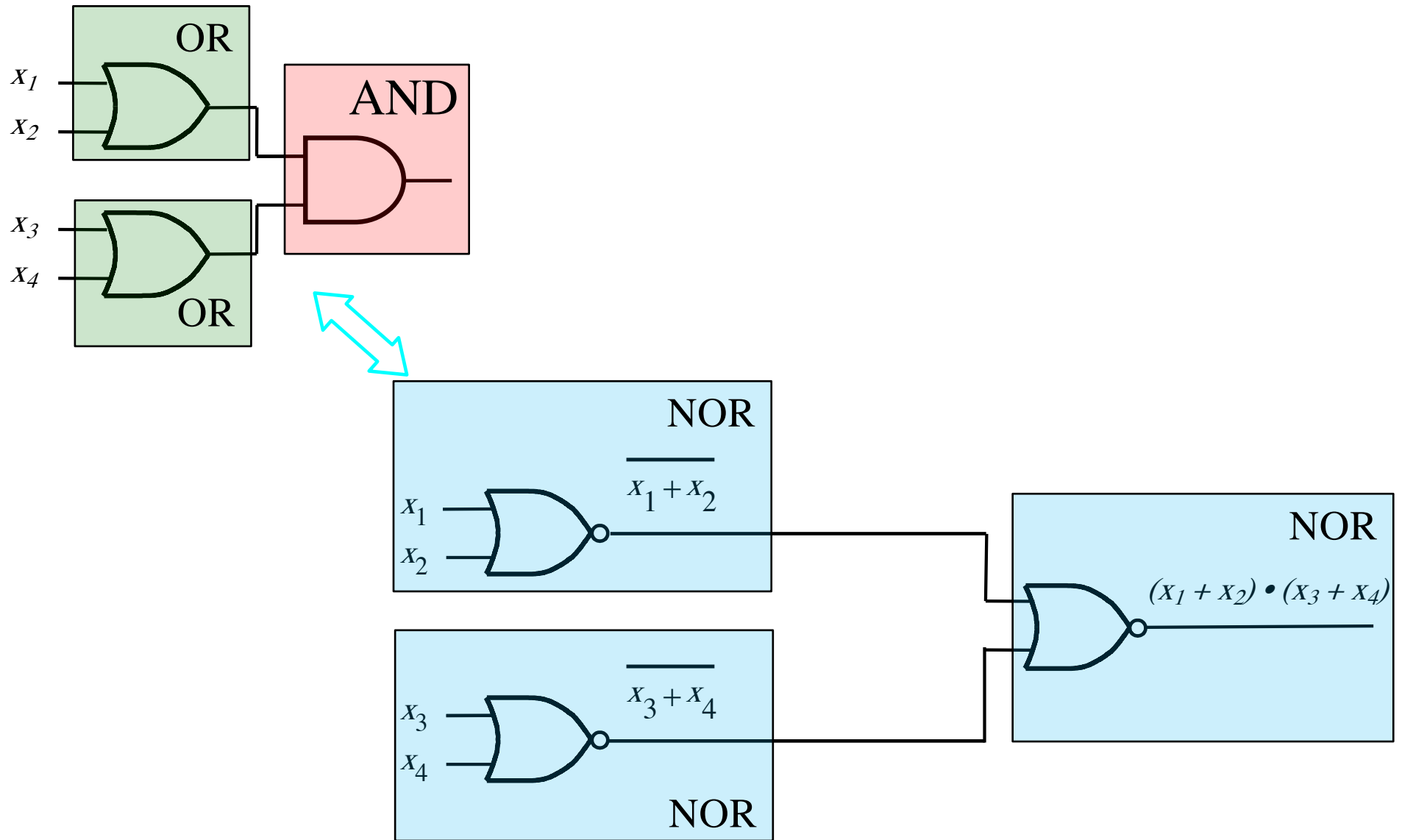
Product-Of-Sums



Product-Of-Sums

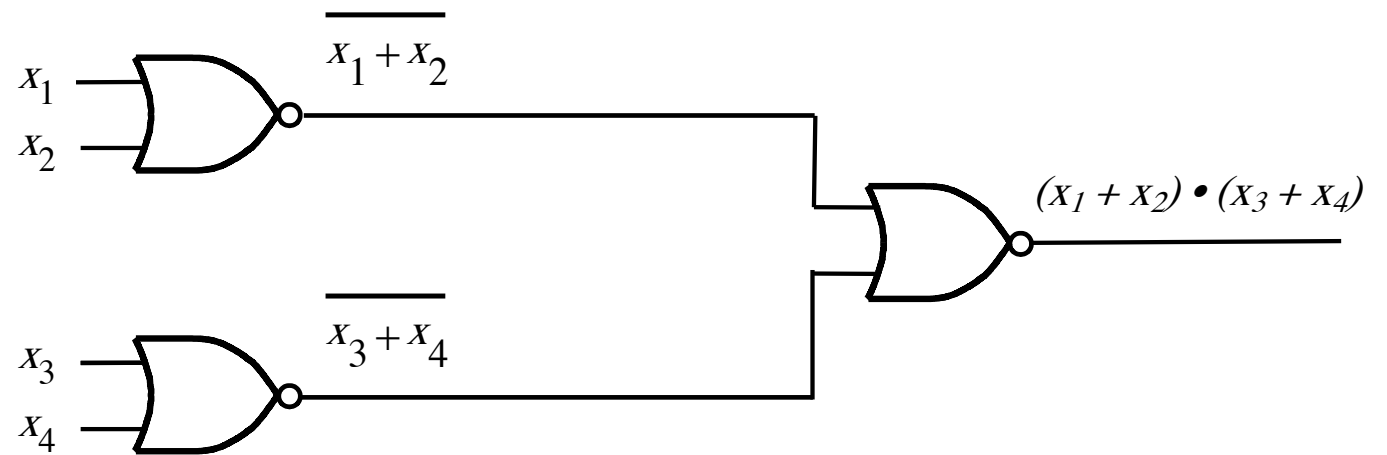
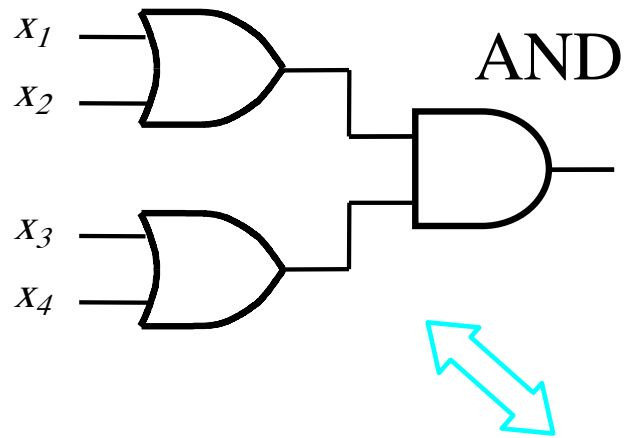


Product-Of-Sums



This circuit uses only NORs

Product-Of-Sums



This circuit uses only NORs

Another Synthesis Example

Truth table for a three-way light control

x_1	x_2	x_3	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Minterms and Maxterms (with three variables)

Row number	x_1	x_2	x_3	Minterm	Maxterm
0	0	0	0	$m_0 = \bar{x}_1\bar{x}_2\bar{x}_3$	$M_0 = x_1 + x_2 + x_3$
1	0	0	1	$m_1 = \bar{x}_1\bar{x}_2x_3$	$M_1 = x_1 + x_2 + \bar{x}_3$
2	0	1	0	$m_2 = \bar{x}_1x_2\bar{x}_3$	$M_2 = x_1 + \bar{x}_2 + x_3$
3	0	1	1	$m_3 = \bar{x}_1x_2x_3$	$M_3 = x_1 + \bar{x}_2 + \bar{x}_3$
4	1	0	0	$m_4 = x_1\bar{x}_2\bar{x}_3$	$M_4 = \bar{x}_1 + x_2 + x_3$
5	1	0	1	$m_5 = x_1\bar{x}_2x_3$	$M_5 = \bar{x}_1 + x_2 + \bar{x}_3$
6	1	1	0	$m_6 = x_1x_2\bar{x}_3$	$M_6 = \bar{x}_1 + \bar{x}_2 + x_3$
7	1	1	1	$m_7 = x_1x_2x_3$	$M_7 = \bar{x}_1 + \bar{x}_2 + \bar{x}_3$

Let's Derive the SOP form

x_1	x_2	x_3	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

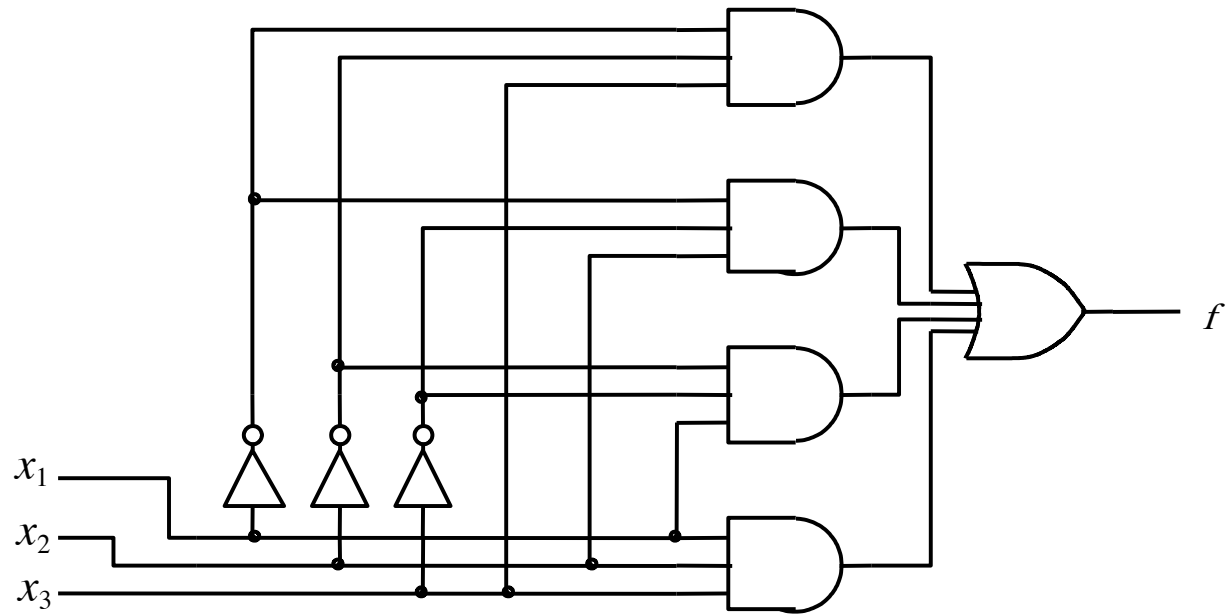
Let's Derive the SOP form

x_1	x_2	x_3	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$$f = m_1 + m_2 + m_4 + m_7$$

$$= \bar{x}_1\bar{x}_2x_3 + \bar{x}_1x_2\bar{x}_3 + x_1\bar{x}_2\bar{x}_3 + x_1x_2x_3$$

Sum-of-products realization



Let's Derive the POS form

x_1	x_2	x_3	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

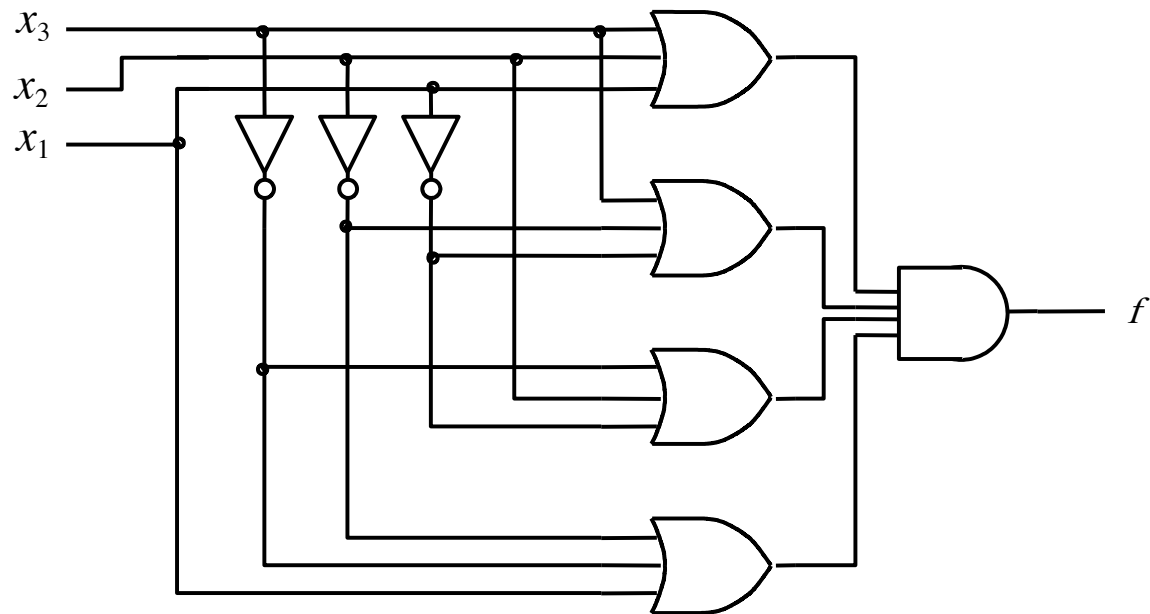
Let's Derive the POS form

x_1	x_2	x_3	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$$f = M_0 \cdot M_3 \cdot M_5 \cdot M_6$$

$$= (x_1 + x_2 + x_3)(x_1 + \bar{x}_2 + \bar{x}_3)(\bar{x}_1 + x_2 + \bar{x}_3)(\bar{x}_1 + \bar{x}_2 + x_3)$$

Product-of-sums realization

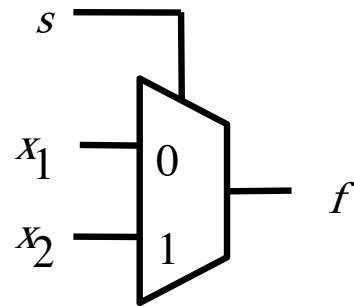


Multiplexers

2-1 Multiplexer (Definition)

- **Has two inputs: x_1 and x_2**
- **Also has another input line s**
- **If $s=0$, then the output is equal to x_1**
- **If $s=1$, then the output is equal to x_2**

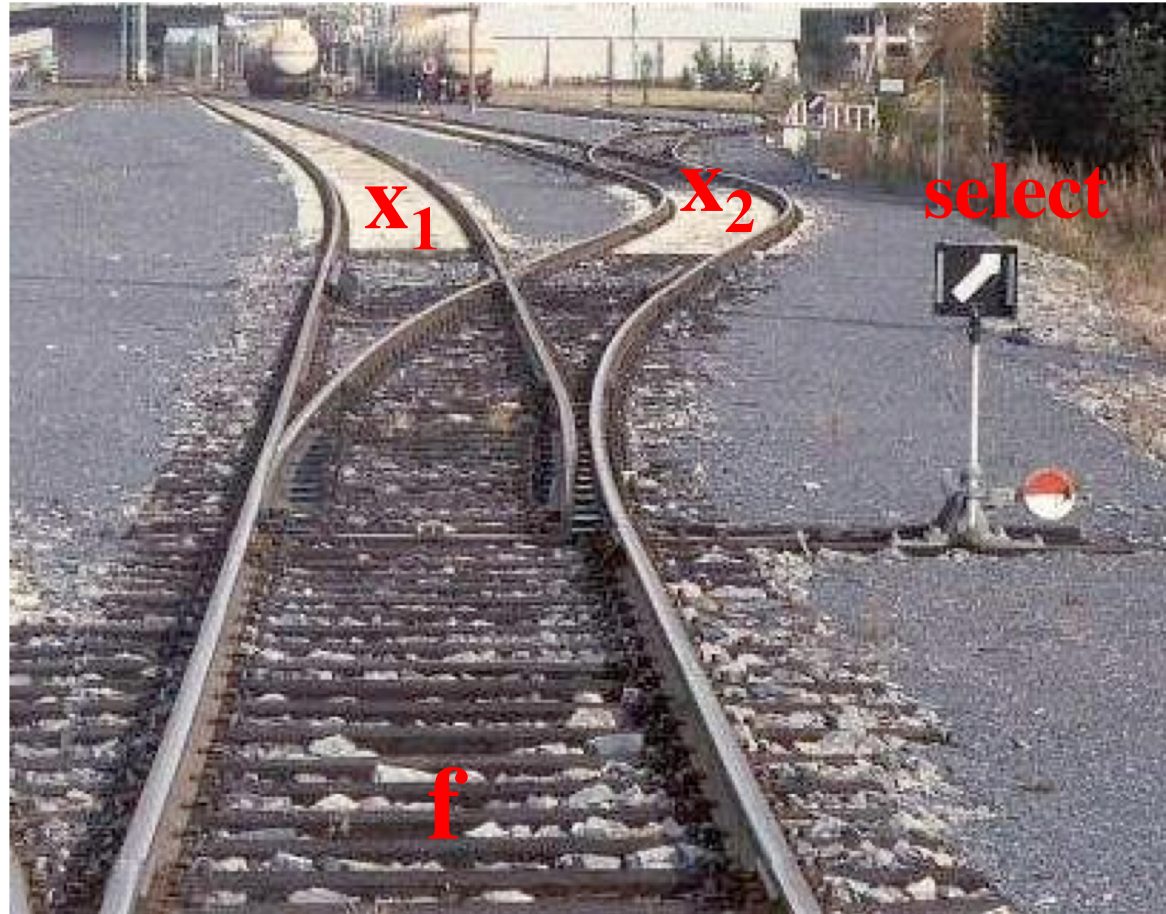
Graphical Symbol for a 2-1 Multiplexer



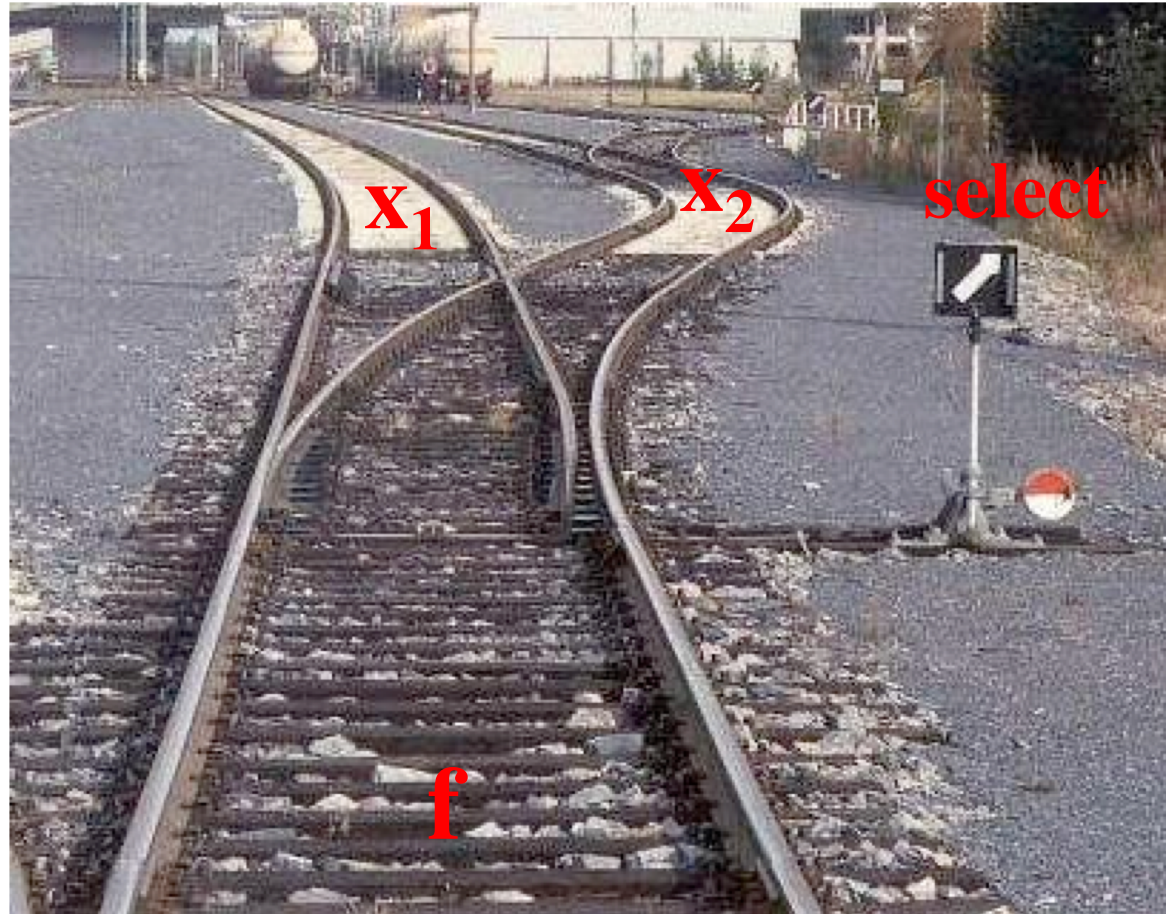
Analogy: Railroad Switch



Analogy: Railroad Switch



Analogy: Railroad Switch



This is not a perfect analogy because the trains can go in either direction, while the multiplexer would only allow them to go from top to bottom.

Truth Table for a 2-1 Multiplexer

s x_1 x_2	$f(s, x_1, x_2)$
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	1

Let's Derive the SOP form

$s \ x_1 \ x_2$	$f(s, x_1, x_2)$
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	1

Let's Derive the SOP form

$s \ x_1 \ x_2$	$f(s, x_1, x_2)$
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	1

Let's Derive the SOP form

$s \ x_1 \ x_2$	$f(s, x_1, x_2)$
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	1

Where should we
put the negation signs?

$s \ x_1 \ x_2$

$s \ x_1 \ x_2$

$s \ x_1 \ x_2$

$s \ x_1 \ x_2$

Let's Derive the SOP form

$s \ x_1 \ x_2$	$f(s, x_1, x_2)$	
0 0 0	0	
0 0 1	0	
0 1 0	1	$\bar{s} \ x_1 \ \bar{x}_2$
0 1 1	1	$\bar{s} \ x_1 \ x_2$
1 0 0	0	
1 0 1	1	$s \ \bar{x}_1 \ x_2$
1 1 0	0	
1 1 1	1	$s \ x_1 \ x_2$

Let's Derive the SOP form

$s \ x_1 \ x_2$	$f(s, x_1, x_2)$	
0 0 0	0	
0 0 1	0	
0 1 0	1	$\bar{s} x_1 \bar{x}_2$
0 1 1	1	$\bar{s} x_1 x_2$
1 0 0	0	
1 0 1	1	$s \bar{x}_1 x_2$
1 1 0	0	
1 1 1	1	$s x_1 x_2$

$$f(s, x_1, x_2) = \bar{s} x_1 \bar{x}_2 + \bar{s} x_1 x_2 + s \bar{x}_1 x_2 + s x_1 x_2$$

Let's simplify this expression

$$f(s, x_1, x_2) = \bar{s} x_1 \bar{x}_2 + \bar{s} x_1 x_2 + s \bar{x}_1 x_2 + s x_1 x_2$$

Let's simplify this expression

$$f(s, x_1, x_2) = \bar{s} x_1 \bar{x}_2 + \bar{s} x_1 x_2 + s \bar{x}_1 x_2 + s x_1 x_2$$

$$f(s, x_1, x_2) = \bar{s} x_1 (\bar{x}_2 + x_2) + s (\bar{x}_1 + x_1) x_2$$

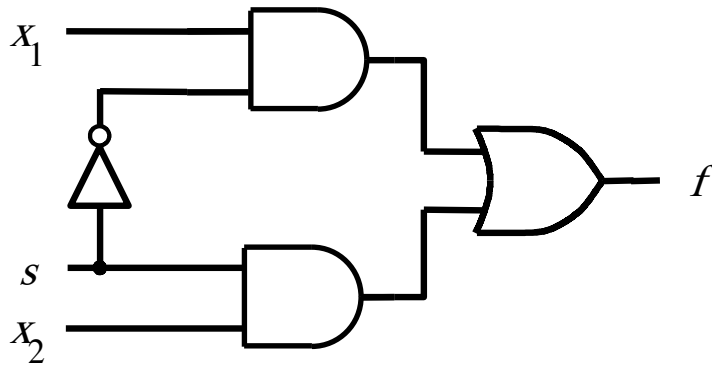
Let's simplify this expression

$$f(s, x_1, x_2) = \bar{s} x_1 \bar{x}_2 + \bar{s} x_1 x_2 + s \bar{x}_1 x_2 + s x_1 x_2$$

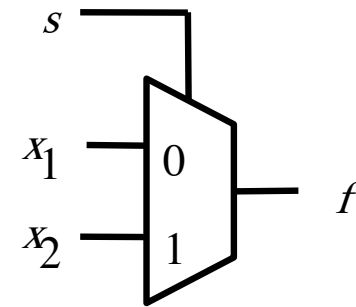
$$f(s, x_1, x_2) = \bar{s} x_1 (\bar{x}_2 + x_2) + s (\bar{x}_1 + x_1) x_2$$

$$f(s, x_1, x_2) = \bar{s} x_1 + s x_2$$

Circuit for 2-1 Multiplexer



(b) Circuit



(c) Graphical symbol

$$f(s, x_1, x_2) = \bar{s} x_1 + s x_2$$

More Compact Truth-Table Representation

s x_1 x_2	$f(s, x_1, x_2)$
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	1

(a) Truth table

s	$f(s, x_1, x_2)$
0	x_1
1	x_2

4-1 Multiplexer (Definition)

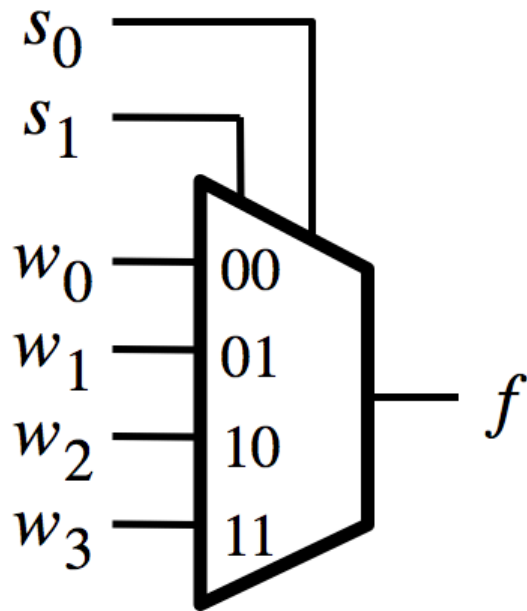
- Has four inputs: w_0 , w_1 , w_2 , w_3
- Also has two select lines: s_1 and s_0
- If $s_1=0$ and $s_0=0$, then the output f is equal to w_0
- If $s_1=0$ and $s_0=1$, then the output f is equal to w_1
- If $s_1=1$ and $s_0=0$, then the output f is equal to w_2
- If $s_1=1$ and $s_0=1$, then the output f is equal to w_3

4-1 Multiplexer (Definition)

- **Has four inputs: w_0 , w_1 , w_2 , w_3**
- **Also has two select lines: s_1 and s_0**
- **If $s_1=0$ and $s_0=0$, then the output f is equal to w_0**
- **If $s_1=0$ and $s_0=1$, then the output f is equal to w_1**
- **If $s_1=1$ and $s_0=0$, then the output f is equal to w_2**
- **If $s_1=1$ and $s_0=1$, then the output f is equal to w_3**

We'll talk more about this when we get to chapter 4, but here is a quick preview.

Graphical Symbol and Truth Table



(a) Graphic symbol

s_1	s_0	f
0	0	w_0
0	1	w_1
1	0	w_2
1	1	w_3

(b) Truth table

The long-form truth table

The long-form truth table

$S_1 S_0$	I_3	I_2	I_1	I_0	F	$S_1 S_0$	I_3	I_2	I_1	I_0	F	$S_1 S_0$	I_3	I_2	I_1	I_0	F	$S_1 S_0$	I_3	I_2	I_1	I_0	F
0 0	0	0	0	0	0	0 1	0	0	0	0	0	1 0	0	0	0	0	0	1 1	0	0	0	0	0
	0	0	0	1	1		0	0	0	1	0		0	0	0	1	0		0	0	0	1	0
	0	0	1	0	0		0	0	1	0	1		0	0	1	0	0		0	0	1	0	0
	0	0	1	1	1		0	0	1	1	1		0	0	1	1	0		0	0	1	1	0
	0	1	0	0	0		0	1	0	0	0		0	1	0	0	1		0	1	0	0	0
	0	1	0	1	1		0	1	0	1	0		0	1	0	1	1		0	1	0	1	0
	0	1	1	0	0		0	1	1	0	1		0	1	1	0	1		0	1	1	0	0
	0	1	1	1	1		0	1	1	1	1		0	1	1	1	1		0	1	1	1	0
	1	0	0	0	0		1	0	0	0	0		1	0	0	0	0		1	0	0	0	1
	1	0	0	1	1		1	0	0	1	0		1	0	0	1	0		1	0	0	1	1
	1	0	1	0	0		1	0	1	0	1		1	0	1	0	0		1	0	1	0	1
	1	0	1	1	1		1	0	1	1	1		1	0	1	1	0		1	0	1	1	1
	1	1	0	0	0		1	1	0	0	0		1	1	0	0	1		1	1	0	0	1
	1	1	0	1	1		1	1	0	1	0		1	1	0	1	1		1	1	0	1	1
	1	1	1	0	0		1	1	1	0	1		1	1	1	0	1		1	1	1	0	1
	1	1	1	1	1		1	1	1	1	1		1	1	1	1	1		1	1	1	1	1

The long-form truth table

$S_1 S_0$	I_3	I_2	I_1	I_0	F	$S_1 S_0$	I_3	I_2	I_1	I_0	F	$S_1 S_0$	I_3	I_2	I_1	I_0	F	$S_1 S_0$	I_3	I_2	I_1	I_0	F
0 0	0	0	0	0	0	0 1	0	0	0	0	0	1 0	0	0	0	0	0	1 1	0	0	0	0	0
	0	0	0	1	1		0	0	0	1	0		0	0	0	1	0		0	0	0	1	0
	0	0	1	0	0		0	0	1	0	1		0	0	1	0	0		0	0	1	0	0
	0	0	1	1	1		0	0	1	1	1		0	0	1	1	0		0	0	1	1	0
	0	1	0	0	0		0	1	0	0	0		0	1	0	0	1		0	1	0	0	0
	0	1	0	1	1		0	1	0	1	0		0	1	0	1	1		0	1	0	1	0
	0	1	1	0	0		0	1	1	0	1		0	1	1	0	1		0	1	1	0	0
	0	1	1	1	1		0	1	1	1	1		0	1	1	1	1		0	1	1	1	0
	1	0	0	0	0		1	0	0	0	0		1	0	0	0	0		1	0	0	0	1
	1	0	0	1	1		1	0	0	1	0		1	0	0	1	0		1	0	0	1	1
	1	0	1	0	0		1	0	1	0	1		1	0	1	0	0		1	0	1	0	1
	1	0	1	1	1		1	0	1	1	1		1	0	1	1	0		1	0	1	1	1
	1	1	0	0	0		1	1	0	0	0		1	1	0	0	1		1	1	0	0	1
	1	1	0	1	1		1	1	0	1	0		1	1	0	1	1		1	1	0	1	1
	1	1	1	0	0		1	1	1	0	1		1	1	1	0	1		1	1	1	0	1
	1	1	1	1	1		1	1	1	1	1		1	1	1	1	1		1	1	1	1	1

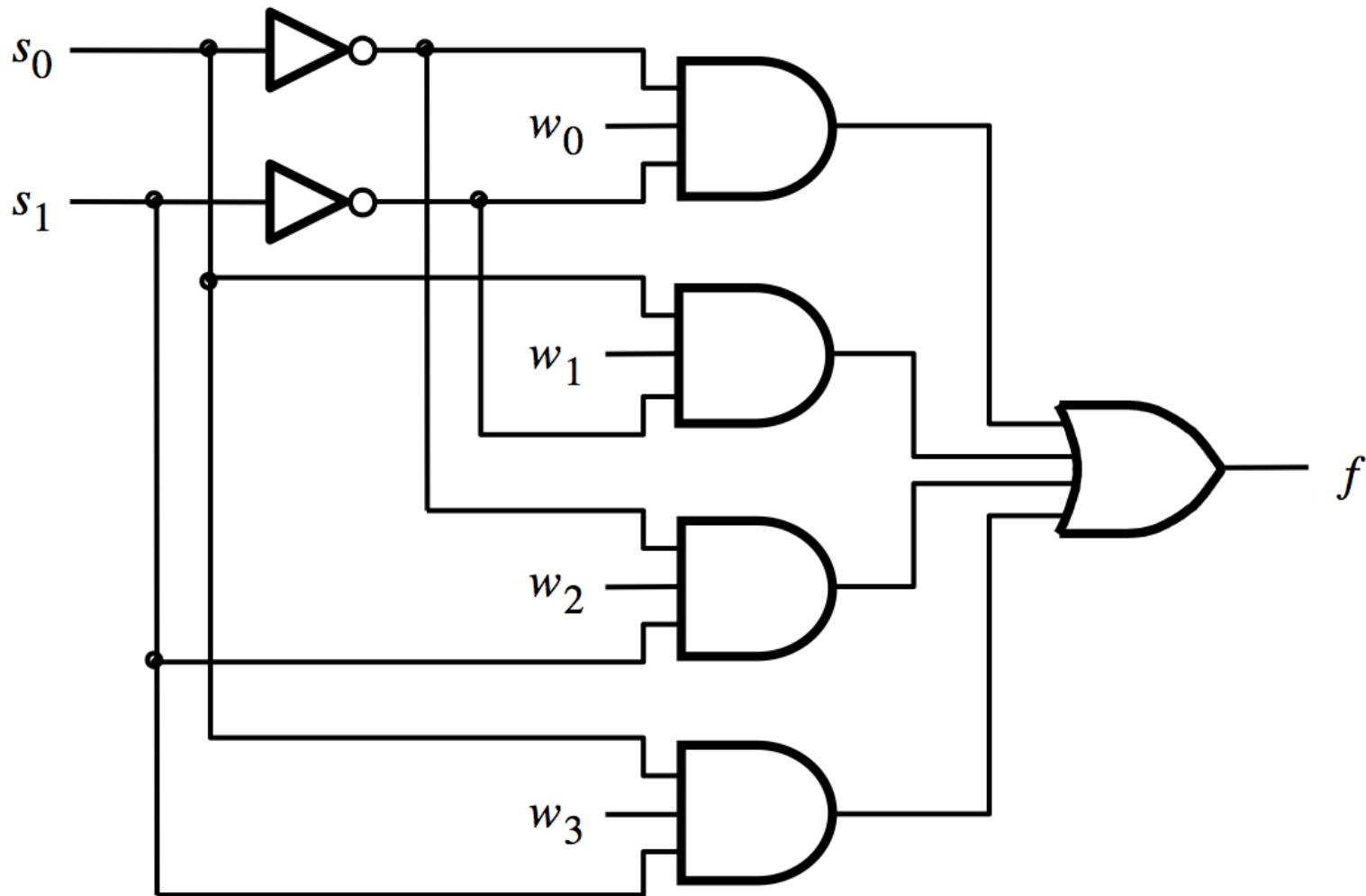
The long-form truth table

$S_1 S_0$	I_3	I_2	I_1	I_0	F	$S_1 S_0$	I_3	I_2	I_1	I_0	F	$S_1 S_0$	I_3	I_2	I_1	I_0	F	$S_1 S_0$	I_3	I_2	I_1	I_0	F
0 0	0	0	0	0	0	0 1	0	0	0	0	0	1 0	0	0	0	0	0	1 1	0	0	0	0	0
	0	0	0	1	1		0	0	0	1	0		0	0	0	1	0		0	0	0	1	0
	0	0	1	0	0		0	0	1	0	1		0	0	1	0	0		0	0	1	0	0
	0	0	1	1	1		0	0	1	1	1		0	0	1	1	0		0	0	1	1	0
	0	1	0	0	0		0	1	0	0	0		0	1	0	0	1		0	1	0	0	0
	0	1	0	1	1		0	1	0	1	0		0	1	0	1	1		0	1	0	1	0
	0	1	1	0	0		0	1	1	0	1		0	1	1	0	1		0	1	1	0	0
	0	1	1	1	1		0	1	1	1	1		0	1	1	1	1		0	1	1	1	0
	1	0	0	0	0		1	0	0	0	0		1	0	0	0	0		1	0	0	0	1
	1	0	0	1	1		1	0	0	1	0		1	0	0	1	0		1	0	0	1	1
	1	0	1	0	0		1	0	1	0	1		1	0	1	0	0		1	0	1	0	1
	1	0	1	1	1		1	0	1	1	1		1	0	1	1	0		1	0	1	1	1
	1	1	0	0	0		1	1	0	0	0		1	1	0	0	1		1	1	0	0	1
	1	1	0	1	1		1	1	0	1	0		1	1	0	1	1		1	1	0	1	1
	1	1	1	0	0		1	1	1	0	1		1	1	1	0	1		1	1	1	0	1
	1	1	1	1	1		1	1	1	1	1		1	1	1	1	1		1	1	1	1	1

The long-form truth table

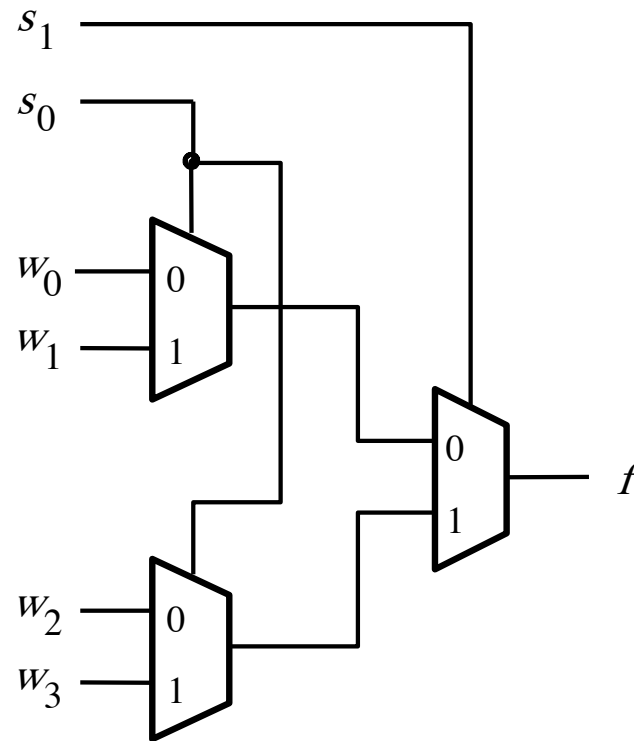
$S_1 S_0$	I_3	I_2	I_1	I_0	F	$S_1 S_0$	I_3	I_2	I_1	I_0	F	$S_1 S_0$	I_3	I_2	I_1	I_0	F	$S_1 S_0$	I_3	I_2	I_1	I_0	F
0 0	0	0	0	0	0	0 1	0	0	0	0	0	1 0	0	0	0	0	0	1 1	0	0	0	0	0
	0	0	0	1	1		0	0	0	1	0		0	0	0	1	0		0	0	0	1	0
	0	0	1	0	0		0	0	1	0	1		0	0	1	0	0		0	0	1	0	0
	0	0	1	1	1		0	0	1	1	1		0	0	1	1	0		0	0	1	1	0
	0	1	0	0	0		0	1	0	0	0		0	1	0	0	1		0	1	0	0	0
	0	1	0	1	1		0	1	0	1	0		0	1	0	1	1		0	1	0	1	0
	0	1	1	0	0		0	1	1	0	1		0	1	1	0	1		0	1	1	0	0
	0	1	1	1	1		0	1	1	1	1		0	1	1	1	1		0	1	1	1	0
	1	0	0	0	0		1	0	0	0	0		1	0	0	0	0		1	0	0	0	1
	1	0	0	1	1		1	0	0	1	0		1	0	0	1	0		1	0	0	1	1
	1	0	1	0	0		1	0	1	0	1		1	0	1	0	0		1	0	1	0	1
	1	0	1	1	1		1	0	1	1	1		1	0	1	1	0		1	0	1	1	1
	1	1	0	0	0		1	1	0	0	0		1	1	0	0	1		1	1	0	0	1
	1	1	0	1	1		1	1	0	1	0		1	1	0	1	1		1	1	0	1	1
	1	1	1	0	0		1	1	1	0	1		1	1	1	0	1		1	1	1	0	1
	1	1	1	1	1		1	1	1	1	1		1	1	1	1	1		1	1	1	1	1

4-1 Multiplexer (SOP circuit)



[Figure 4.2c from the textbook]

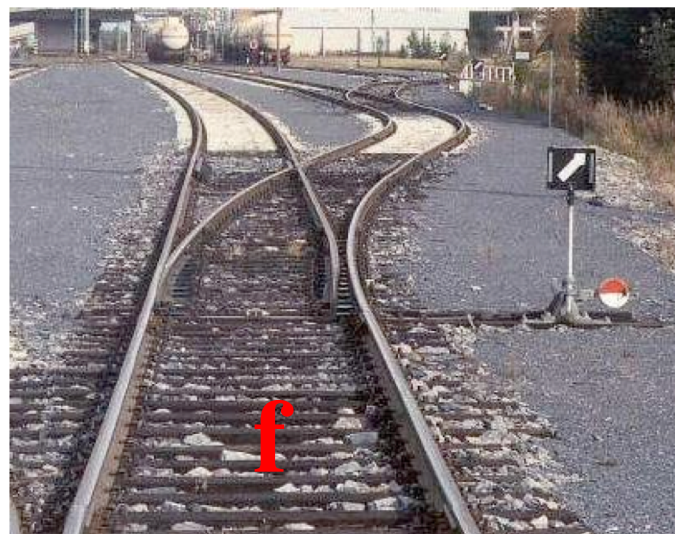
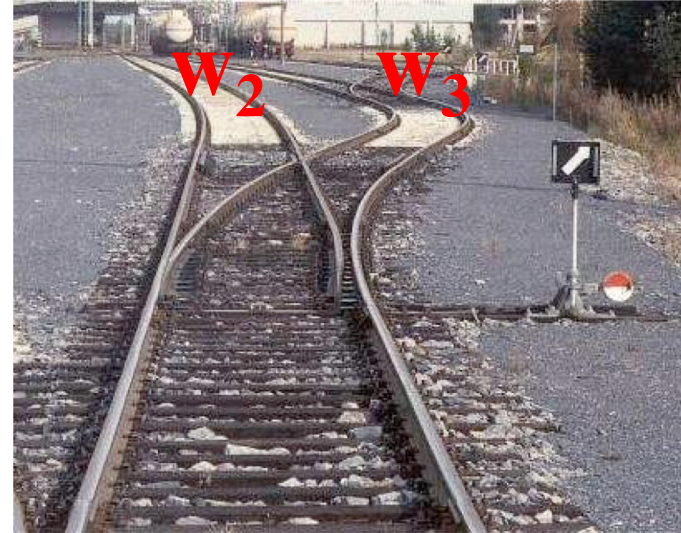
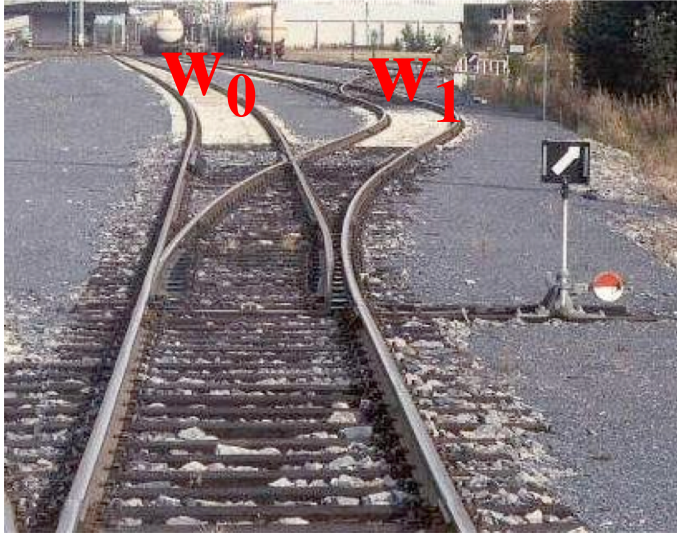
Using three 2-to-1 multiplexers to build one 4-to-1 multiplexer



Analogy: Railroad Switches

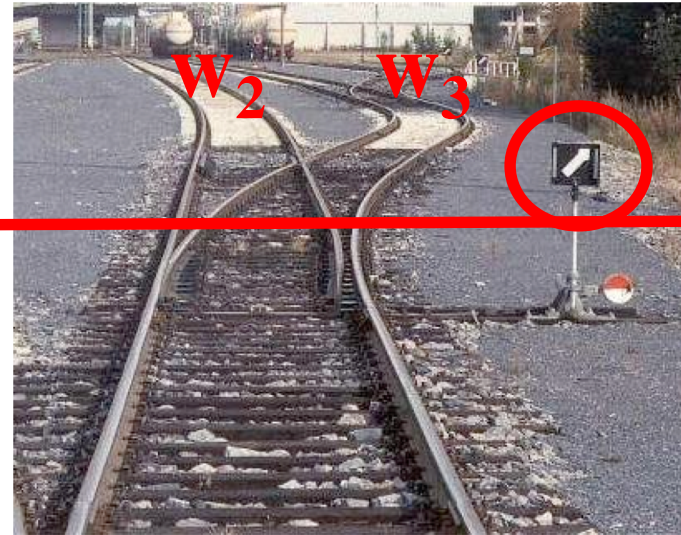
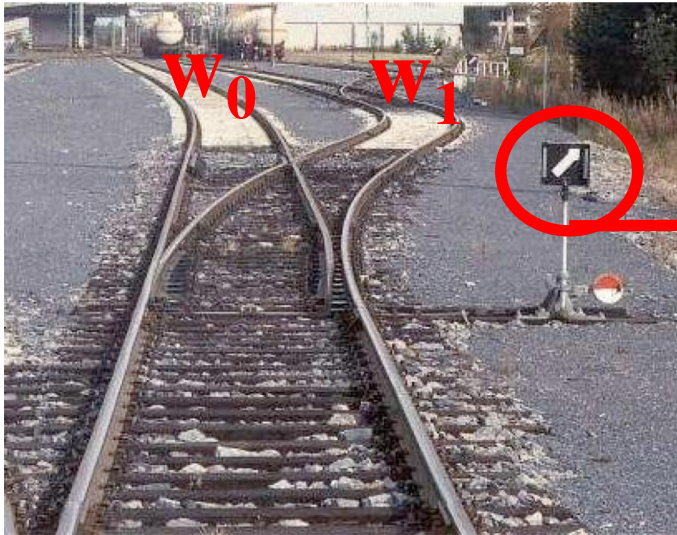


Analogy: Railroad Switches



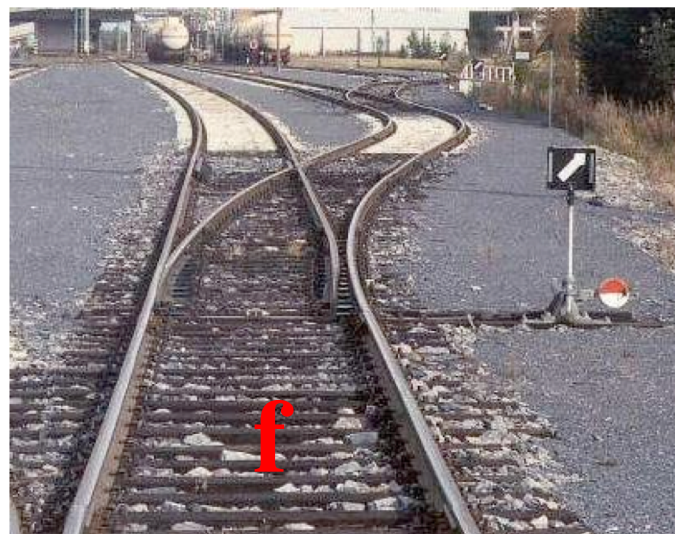
S_1

Analogy: Railroad Switches



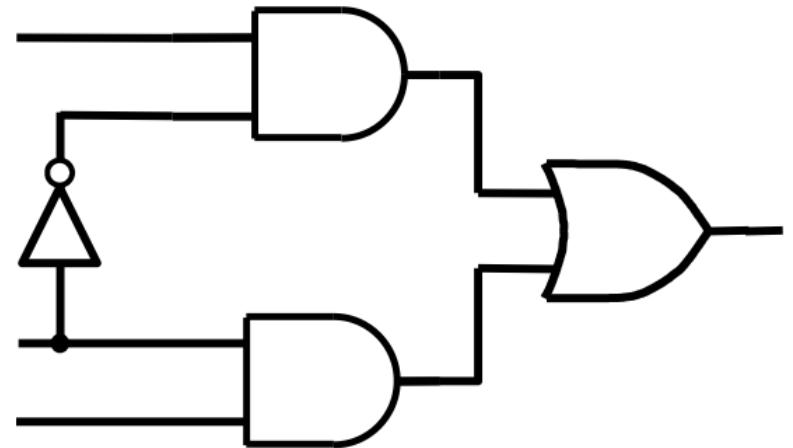
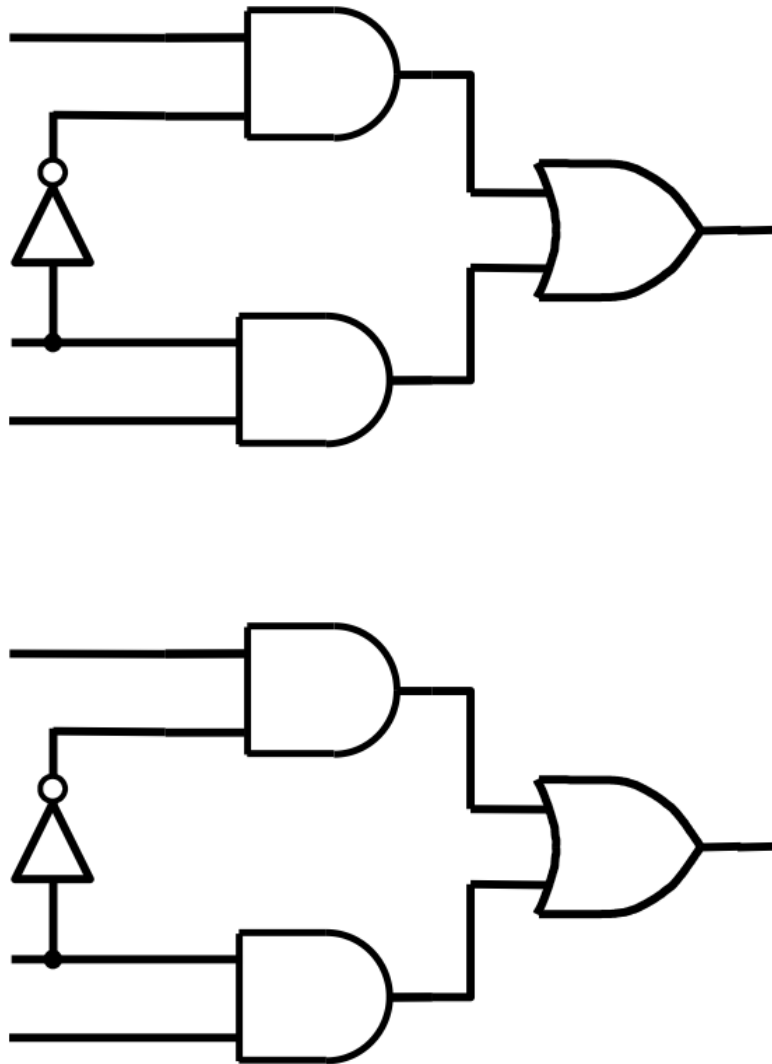
S_0

these two
switches are
controlled
together

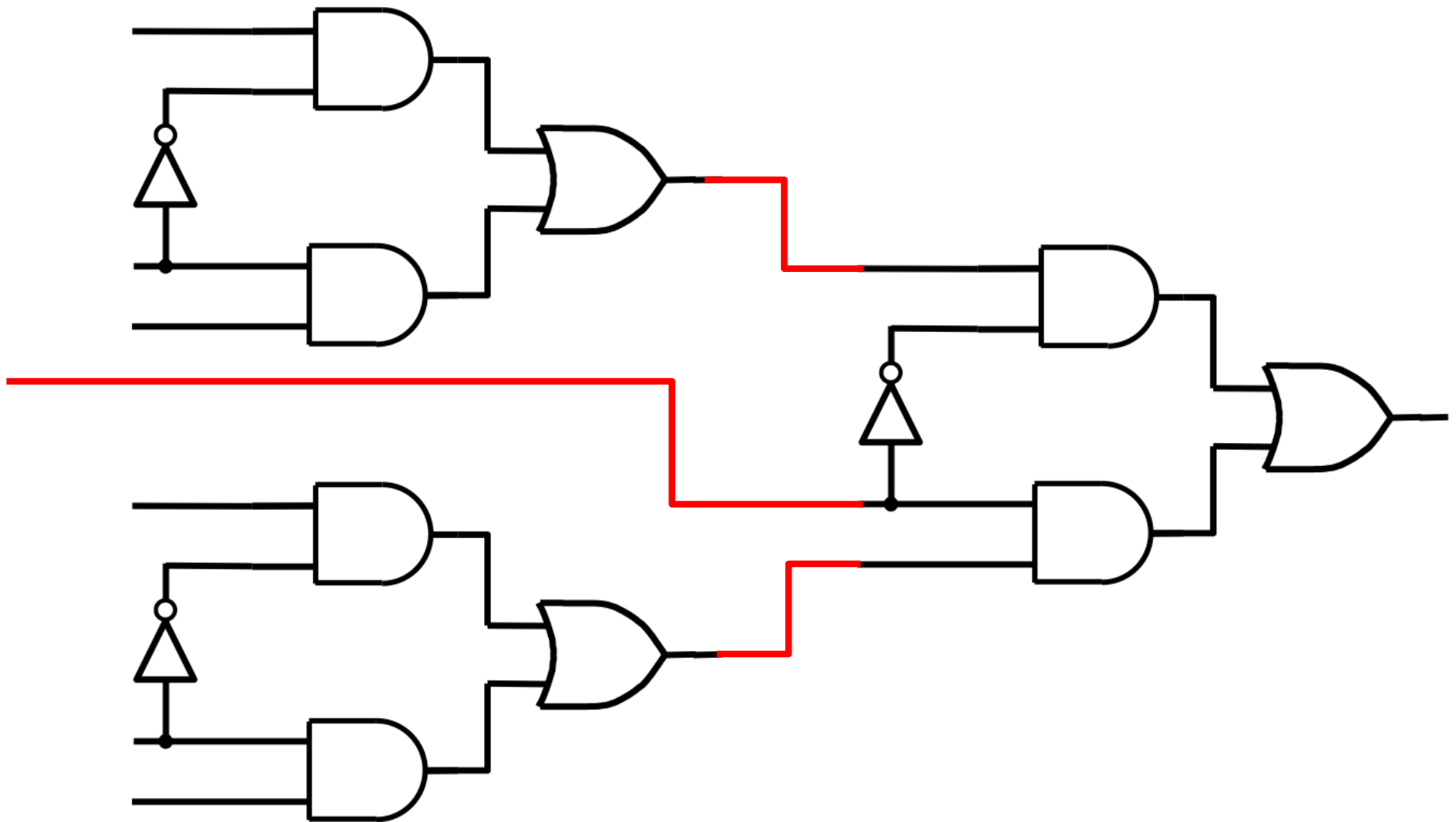


S_1

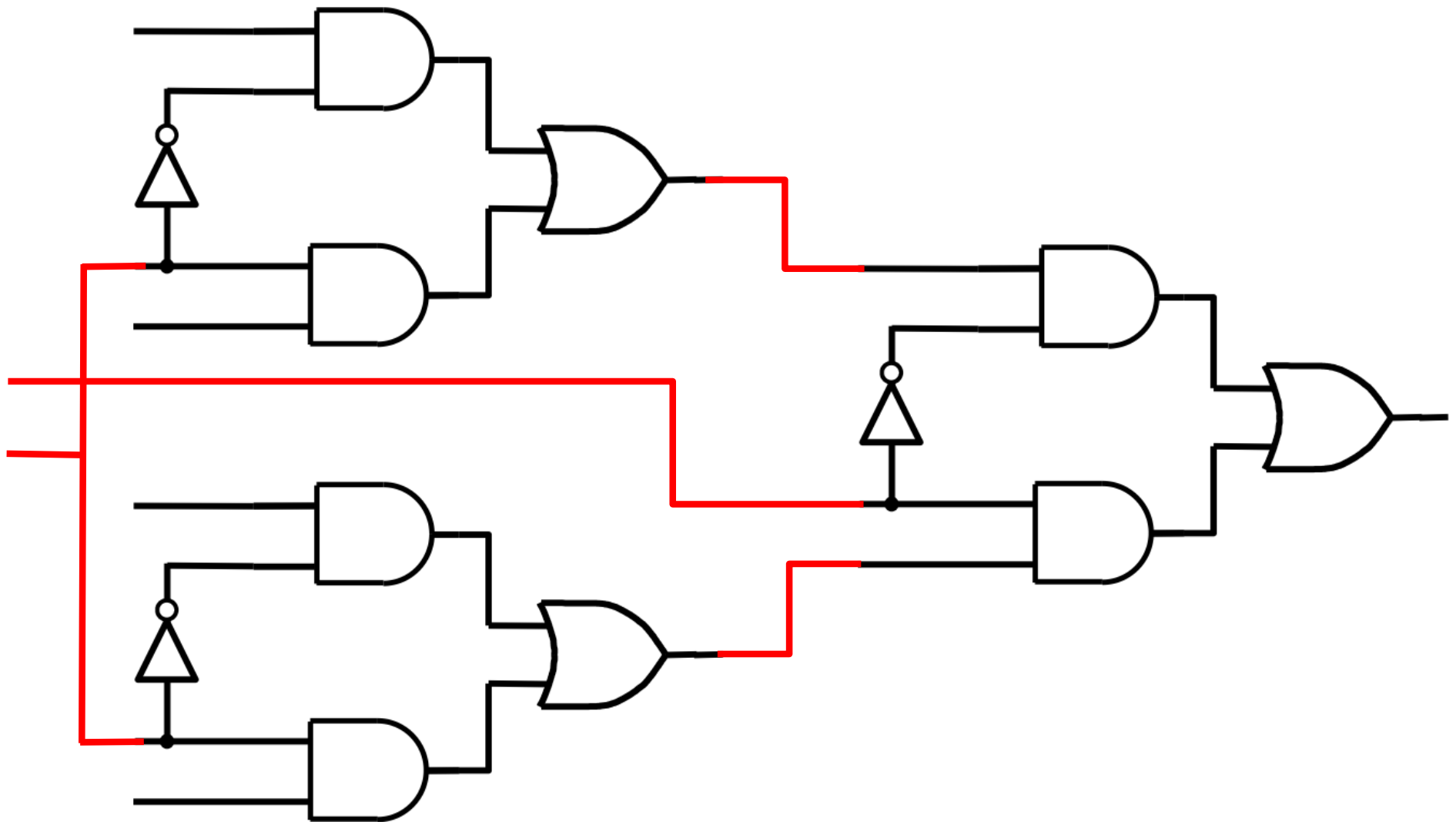
Using three 2-to-1 multiplexers to build one 4-to-1 multiplexer



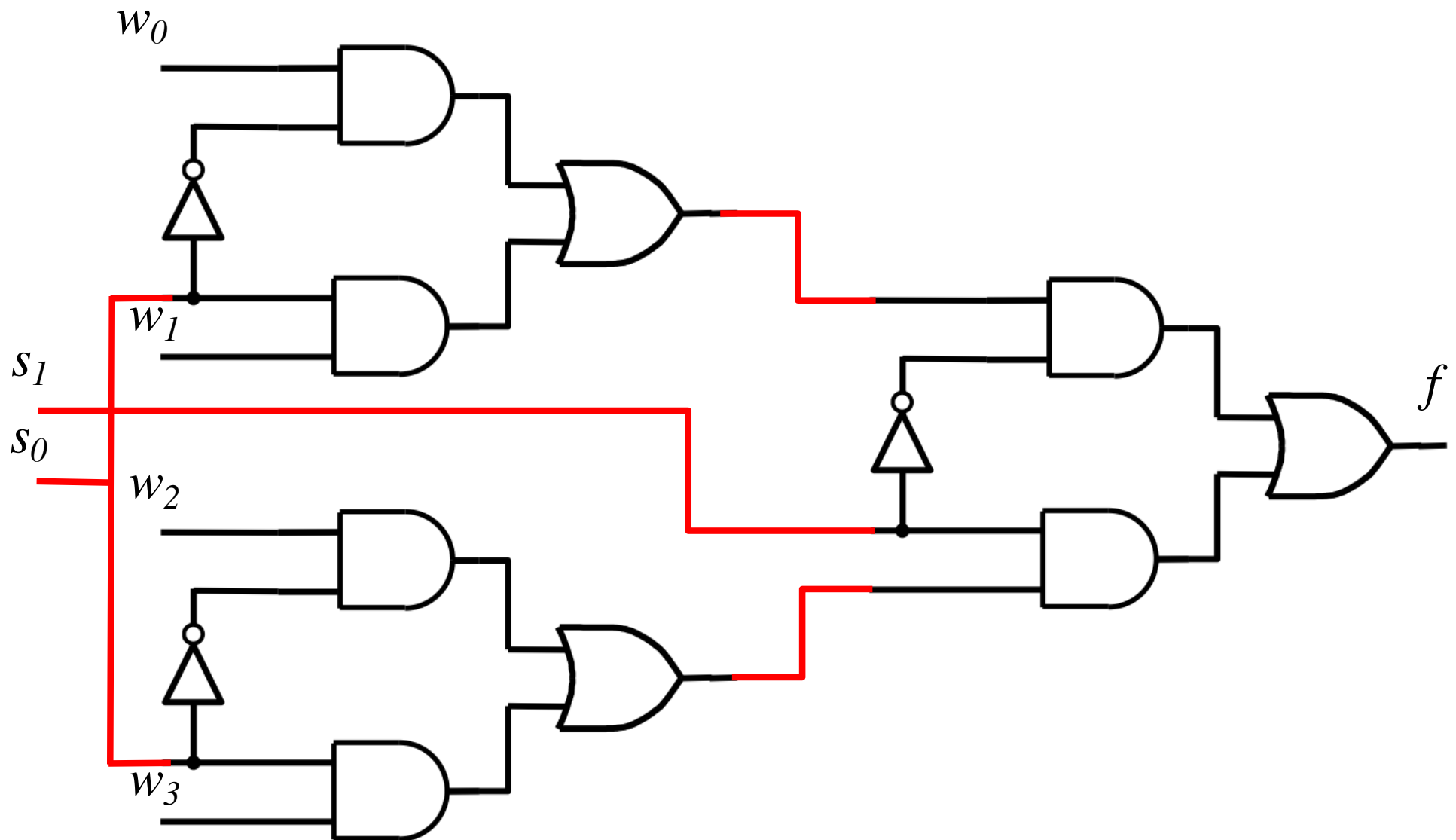
Using three 2-to-1 multiplexers to build one 4-to-1 multiplexer



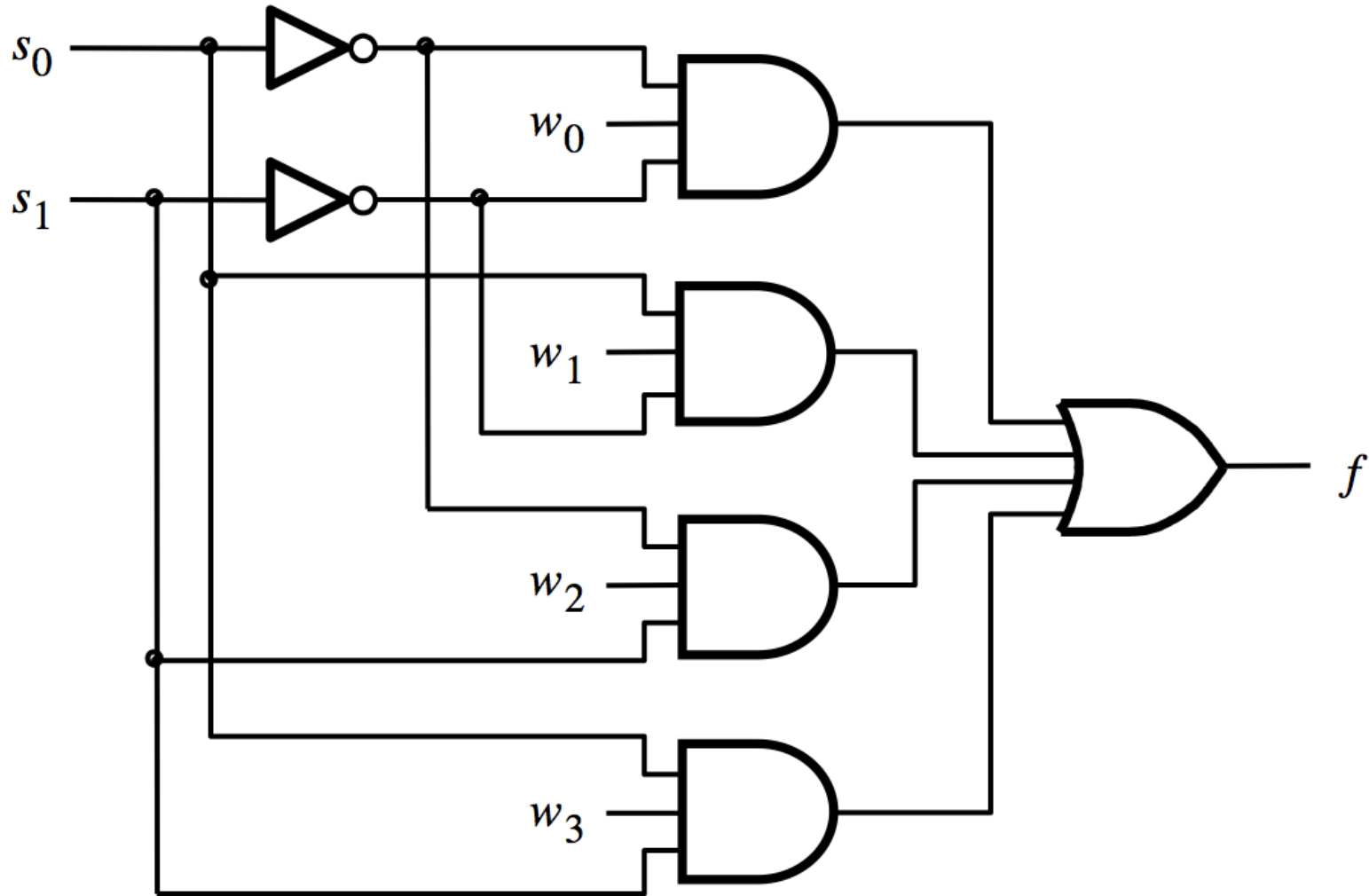
Using three 2-to-1 multiplexers to build one 4-to-1 multiplexer



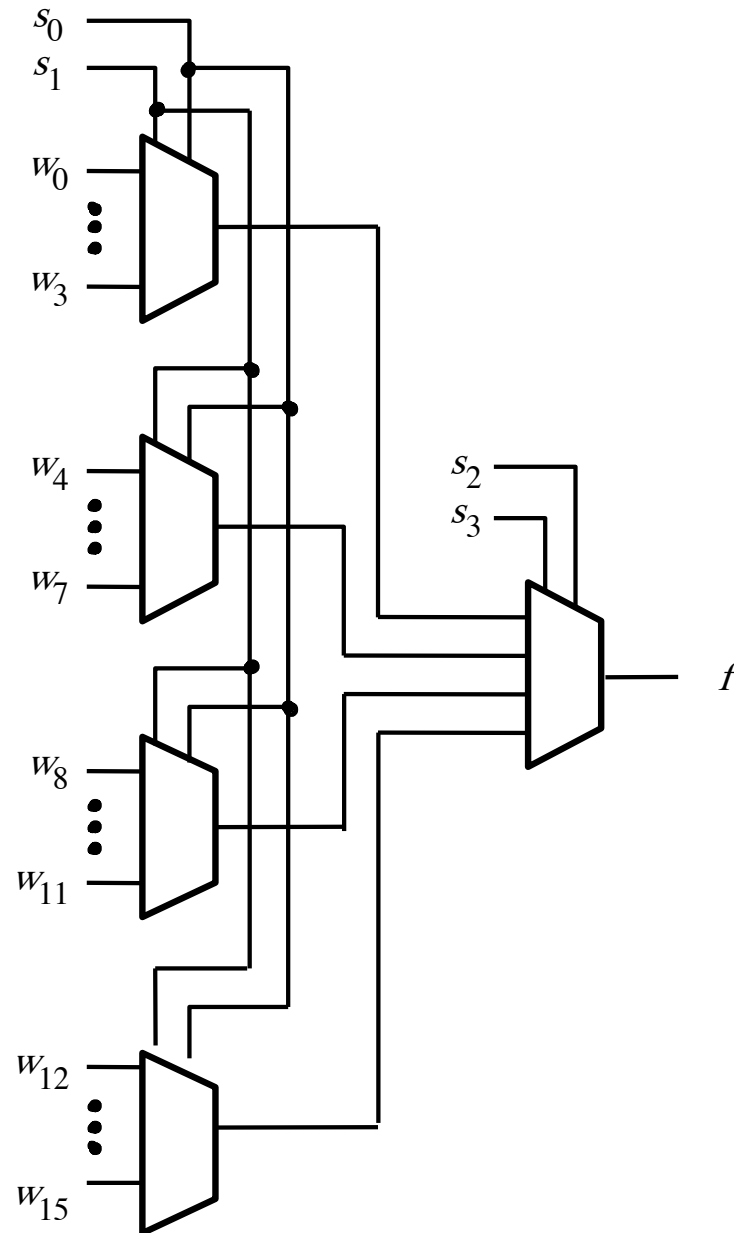
Using three 2-to-1 multiplexers to build one 4-to-1 multiplexer



That is different from the SOP form of the 4-1 multiplexer shown below, which uses fewer gates



16-1 Multiplexer



[Figure 4.4 from the textbook]



[<http://upload.wikimedia.org/wikipedia/commons/2/26/SunsetTracksCrop.JPG>]

Questions?

THE END