



CprE 281: Digital Logic

Instructor: Alexander Stoytchev

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

Multiplexers

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

Administrative Stuff

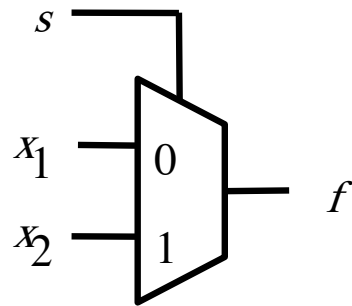
- **HW 6 is due on Monday Oct 10 @ 10pm**
- **Next week: Lab 6**
- **Midterm progress report grades are due next week**

2-to-1 Multiplexer

2-to-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

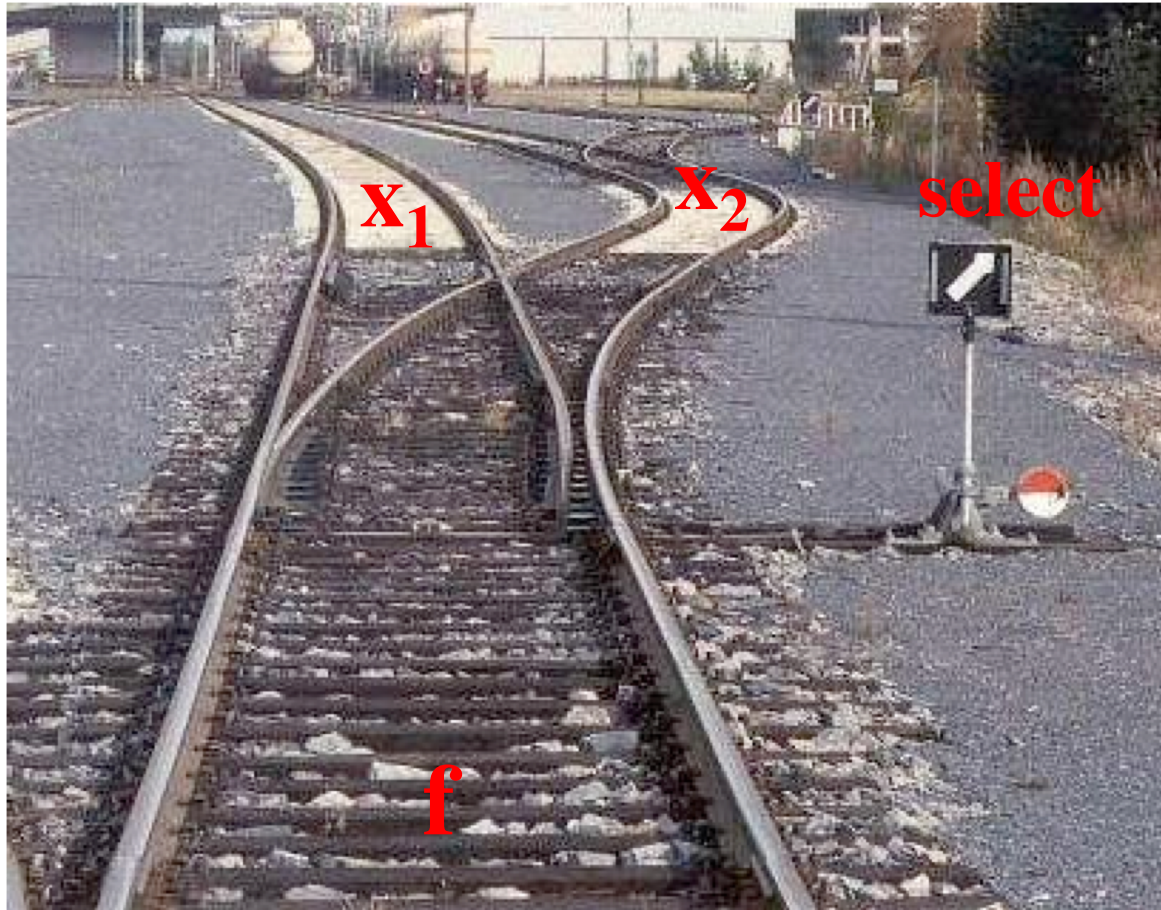
Graphical Symbol for a 2-to-1 Multiplexer



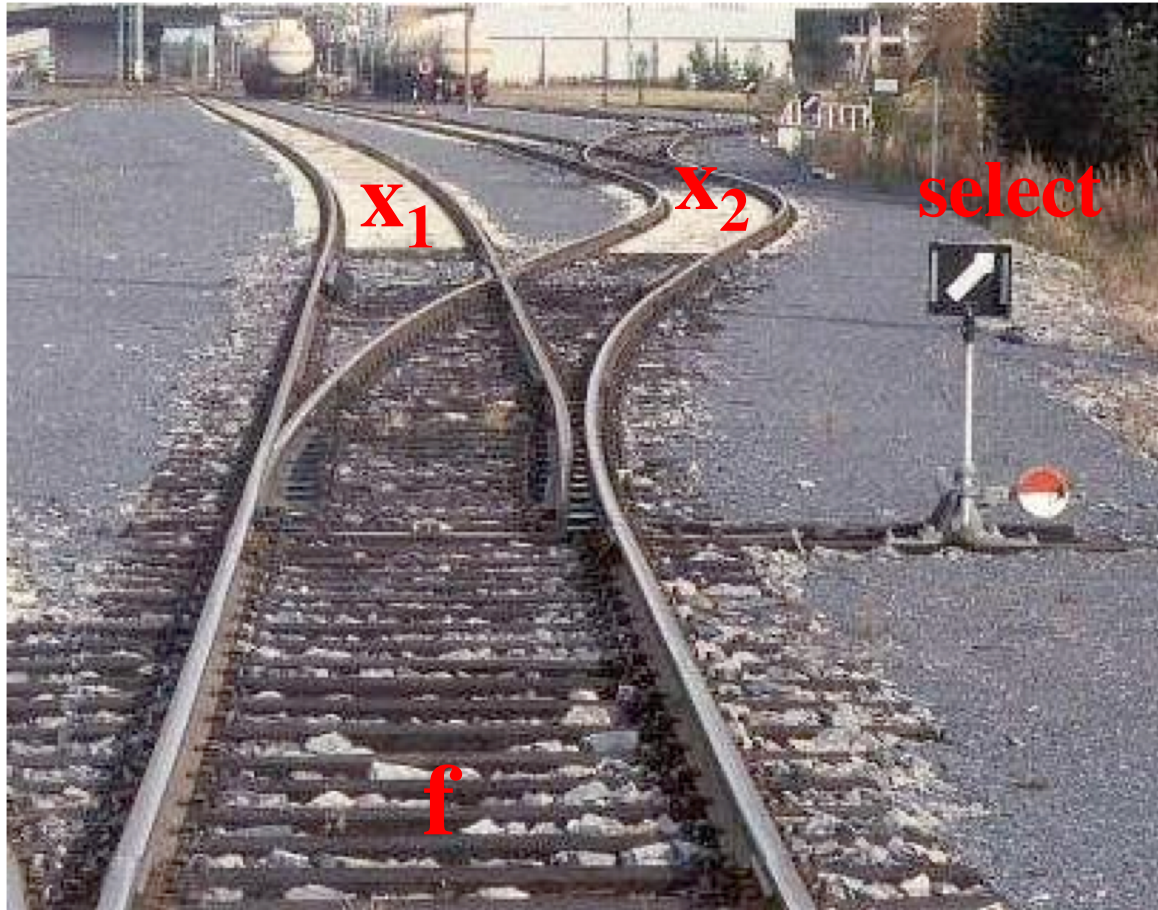
Analogy: Railroad Switch



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-to-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

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

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

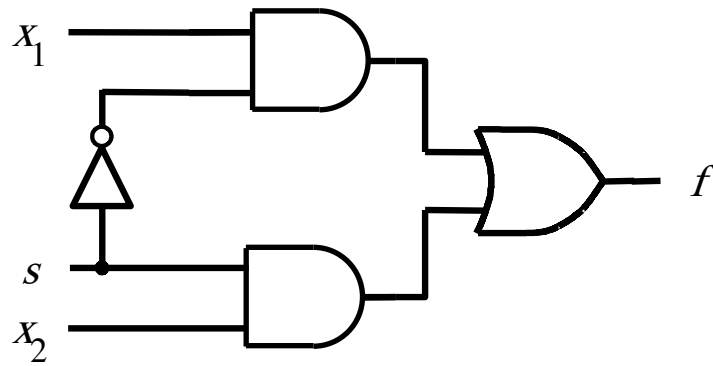
Let's simplify this expression

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

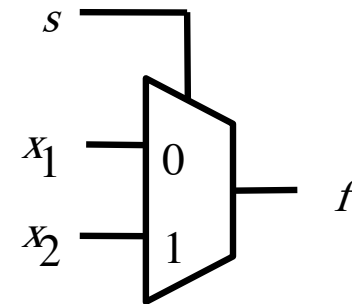
$$f(s, x_1, x_2) = \bar{s} x_1 (\bar{x}_2 + x_2) + s (\bar{x}_1 + x_1) x_2$$

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

Circuit for 2-1 Multiplexer



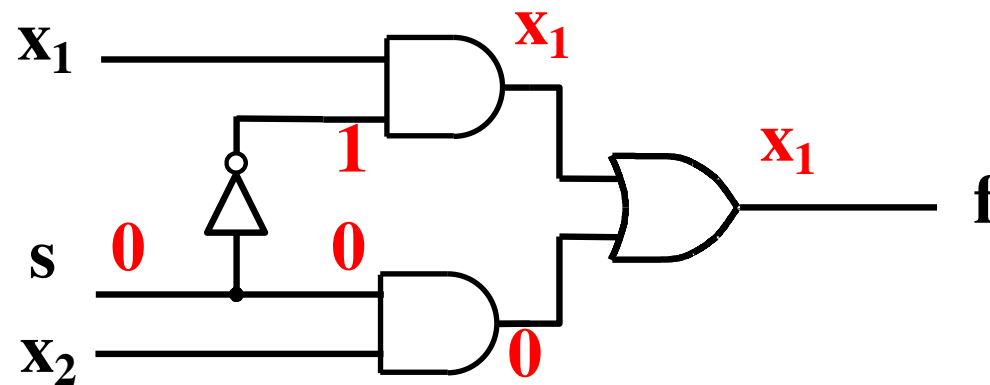
(b) Circuit



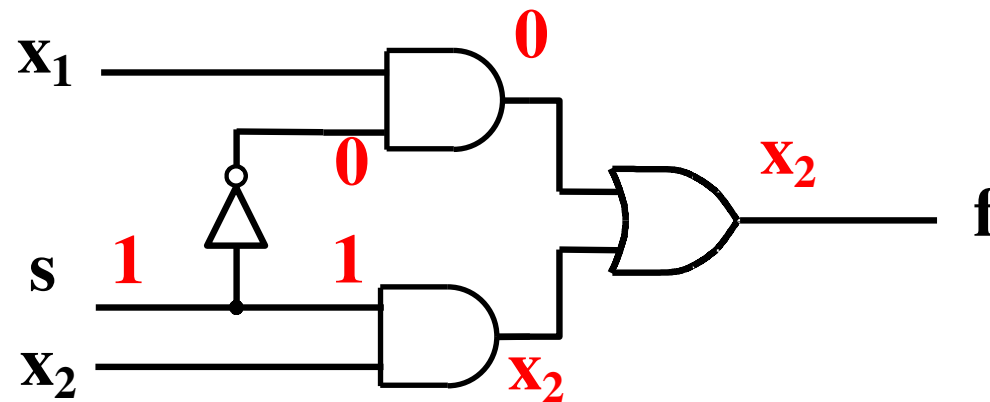
(c) Graphical symbol

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

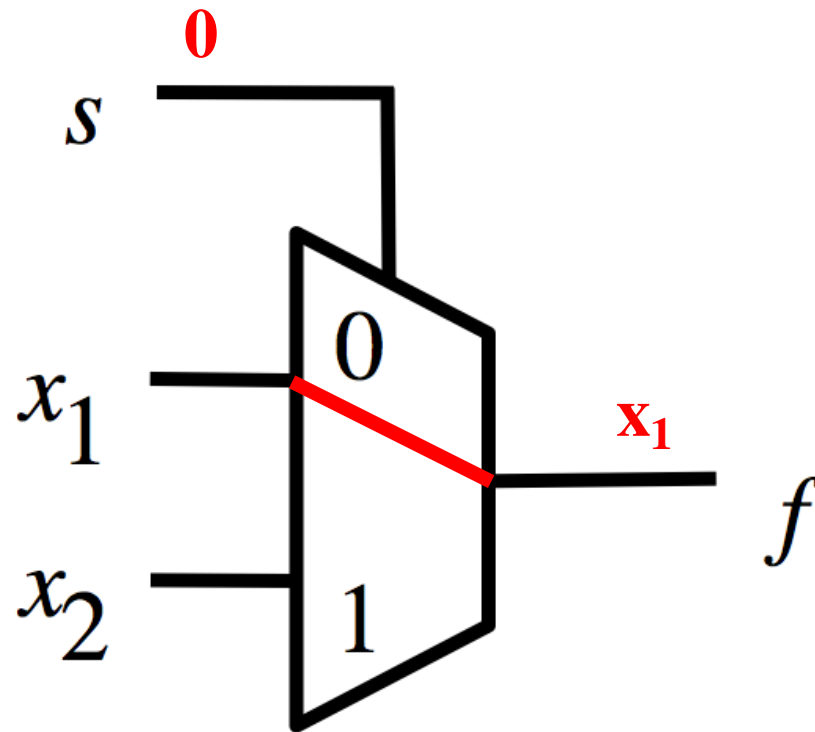
Analysis of the 2-to-1 Multiplexer (when the input $s=0$)



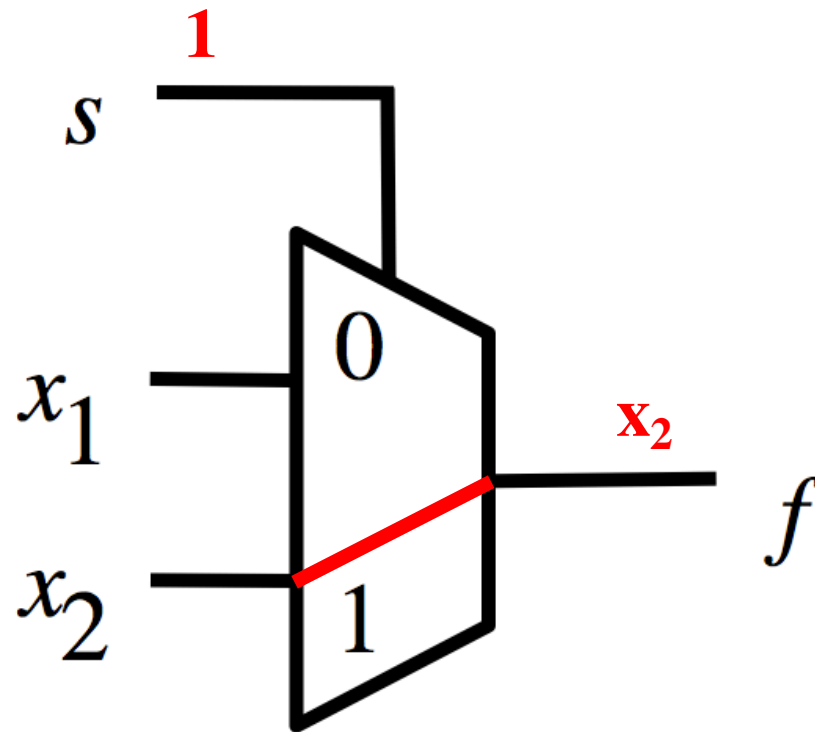
Analysis of the 2-to-1 Multiplexer (when the input $s=1$)



Analysis of the 2-to-1 Multiplexer (when the input $s=0$)



Analysis of the 2-to-1 Multiplexer (when the input $s=1$)



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

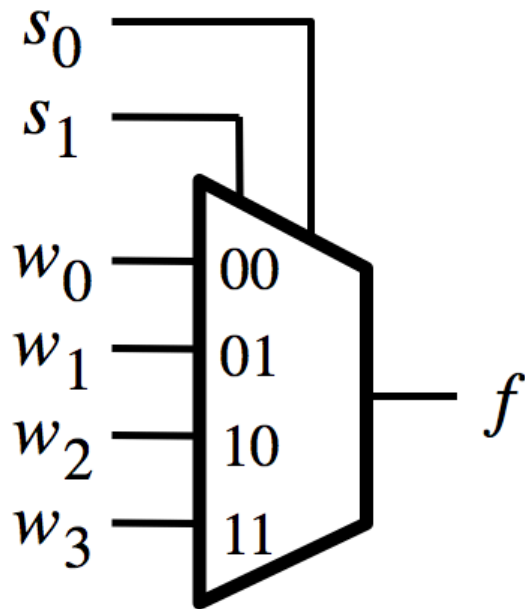
s	$f(s, x_1, x_2)$
0	x_1
1	x_2

4-to-1 Multiplexer

4-to-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

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	0	0	1	1	0 1	0	0	0	1	0	1 0	0	0	0	1	0	1 1	0	0	0	1	0
0 0	0	0	1	0	0	0 1	0	0	1	0	1	1 0	0	0	1	0	0	1 1	0	0	1	0	0
0 0	0	0	1	1	1	0 1	0	0	1	1	1	1 0	0	0	1	1	0	1 1	0	0	1	1	0
0 0	0	1	0	0	0	0 1	0	1	0	0	0	1 0	0	1	0	0	1	1 1	0	0	1	0	0
0 0	0	1	0	1	1	0 1	0	1	0	1	0	1 0	0	1	0	1	1	1 1	0	0	1	0	0
0 0	0	1	1	0	0	0 1	0	1	1	0	1	1 0	0	1	1	0	1	1 1	0	0	1	0	0
0 0	0	1	1	1	1	0 1	0	1	1	1	1	1 0	0	1	1	1	1	1 1	0	0	1	1	0
0 1	0	0	0	0	0	0 1	1	0	0	0	0	1 0	1	0	0	0	0	1 1	0	0	0	0	1
0 1	0	0	0	1	1	0 1	1	0	0	1	0	1 0	1	0	0	1	0	1 1	0	0	1	1	1
0 1	0	0	1	0	0	0 1	1	0	1	0	1	1 0	1	0	1	0	0	1 1	0	0	1	0	1
0 1	0	0	1	1	1	0 1	1	0	1	1	1	1 0	1	0	1	1	0	1 1	0	0	1	1	1
0 1	0	1	0	0	0	0 1	1	1	0	0	0	1 0	1	1	0	0	1	1 1	0	0	1	0	1
0 1	0	1	0	1	1	0 1	1	1	0	1	0	1 0	1	1	0	1	1	1 1	0	0	1	1	1
0 1	0	1	1	0	0	0 1	1	1	0	0	0	1 0	1	1	0	0	1	1 1	0	0	1	0	1
0 1	0	1	1	1	1	0 1	1	1	0	1	0	1 0	1	1	0	1	1	1 1	0	0	1	1	1
0 1	0	1	1	1	1	0 1	1	1	1	1	1	1 0	1	1	1	0	1	1 1	0	0	1	1	1
0 1	1	0	0	0	0	0 1	1	1	1	1	1	1 0	1	1	1	1	1	1 1	0	0	1	1	1
0 1	1	0	0	1	1	0 1	1	1	1	1	1	1 0	1	1	1	1	1	1 1	0	0	1	1	1
0 1	1	0	1	0	0	0 1	1	1	1	1	1	1 0	1	1	1	1	1	1 1	0	0	1	1	1
0 1	1	0	1	1	1	0 1	1	1	1	1	1	1 0	1	1	1	1	1	1 1	0	0	1	1	1
0 1	1	1	0	0	0	0 1	1	1	1	1	1	1 0	1	1	1	1	1	1 1	0	0	1	1	1
0 1	1	1	0	1	1	0 1	1	1	1	1	1	1 0	1	1	1	1	1	1 1	0	0	1	1	1
0 1	1	1	1	0	0	0 1	1	1	1	1	1	1 0	1	1	1	1	1	1 1	0	0	1	1	1
0 1	1	1	1	1	1	0 1	1	1	1	1	1	1 0	1	1	1	1	1	1 1	0	0	1	1	1

identical

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

identical

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

identical

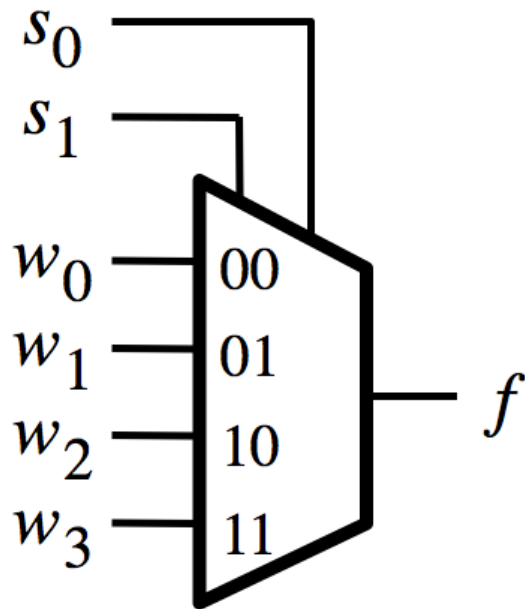
[<http://www.absoluteastronomy.com/topics/Multiplexer>]

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

identical

Graphical Symbol and Truth Table

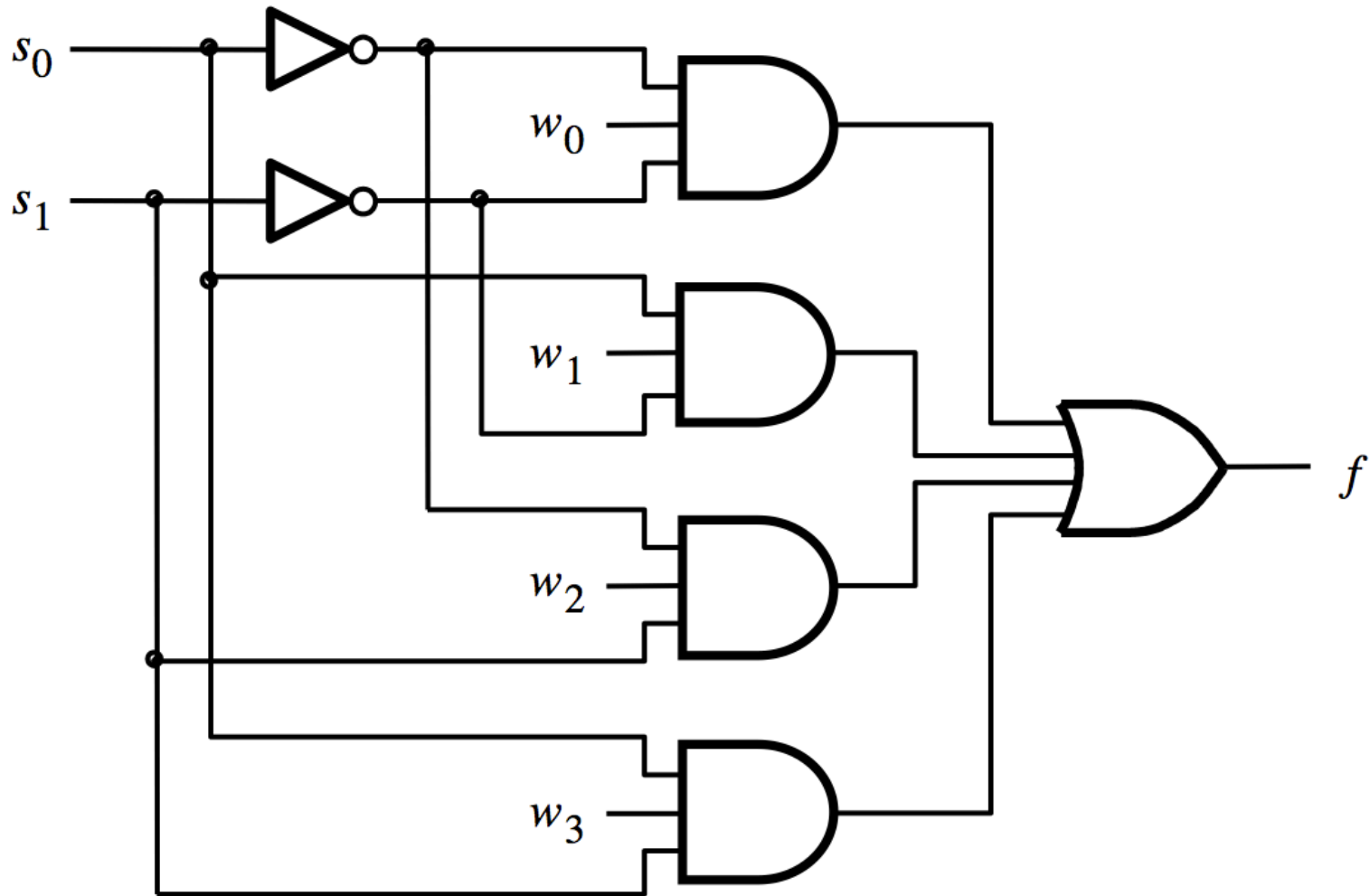


(a) Graphic symbol

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

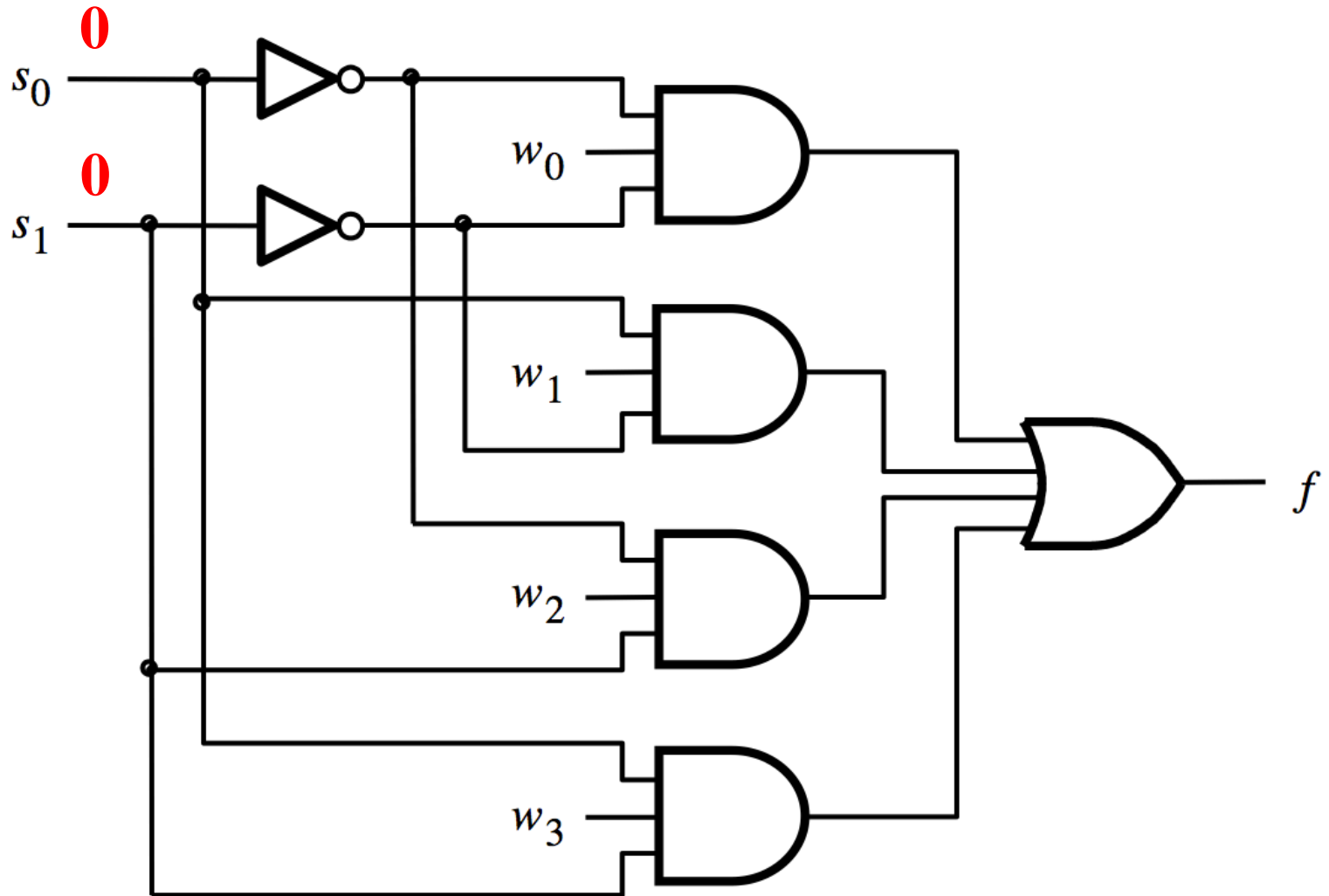
(b) Truth table

4-to-1 Multiplexer (SOP circuit)

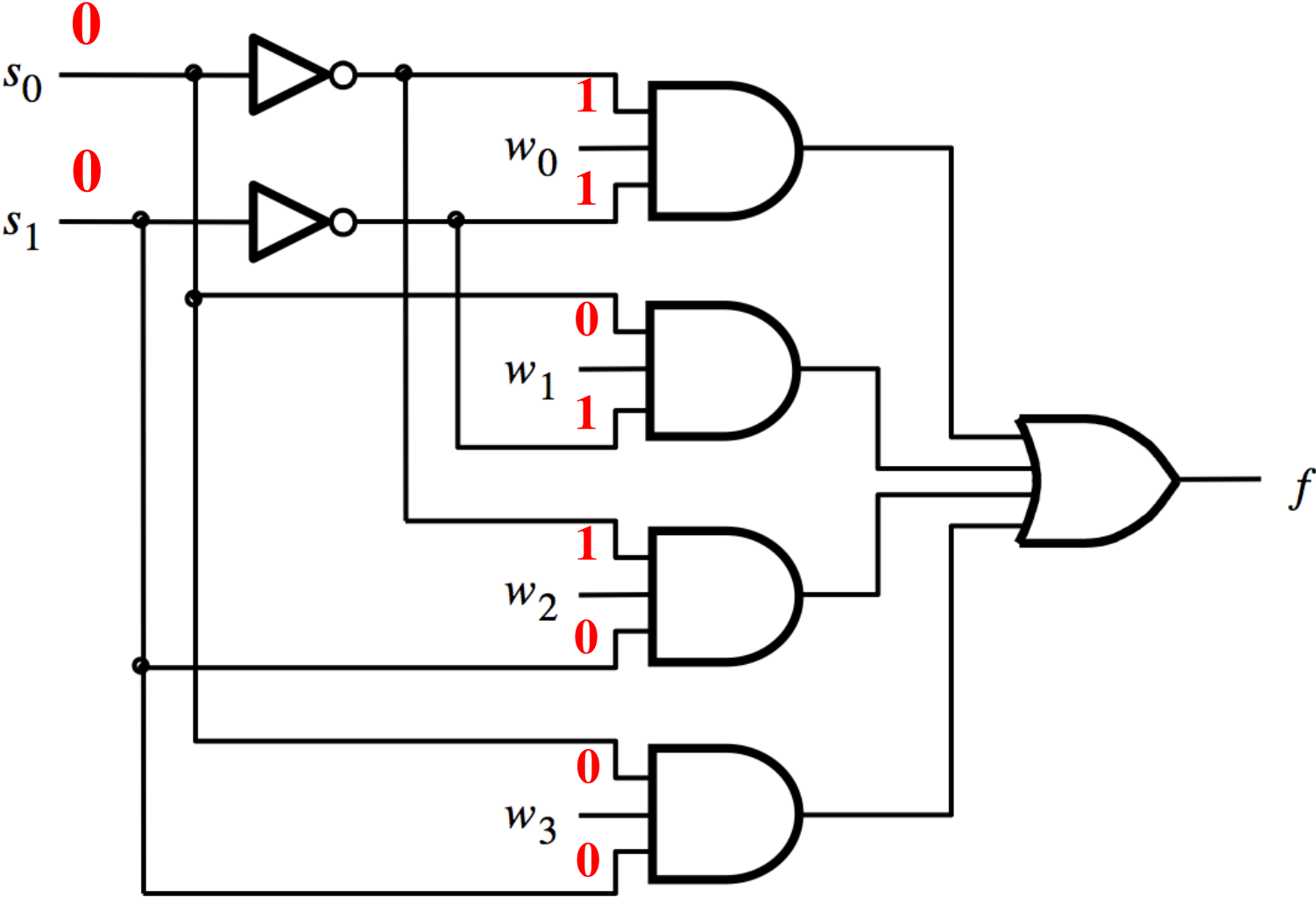


$$f = \overline{s_1} \overline{s_0} w_0 + \overline{s_1} s_0 w_1 + s_1 \overline{s_0} w_2 + s_1 s_0 w_3$$

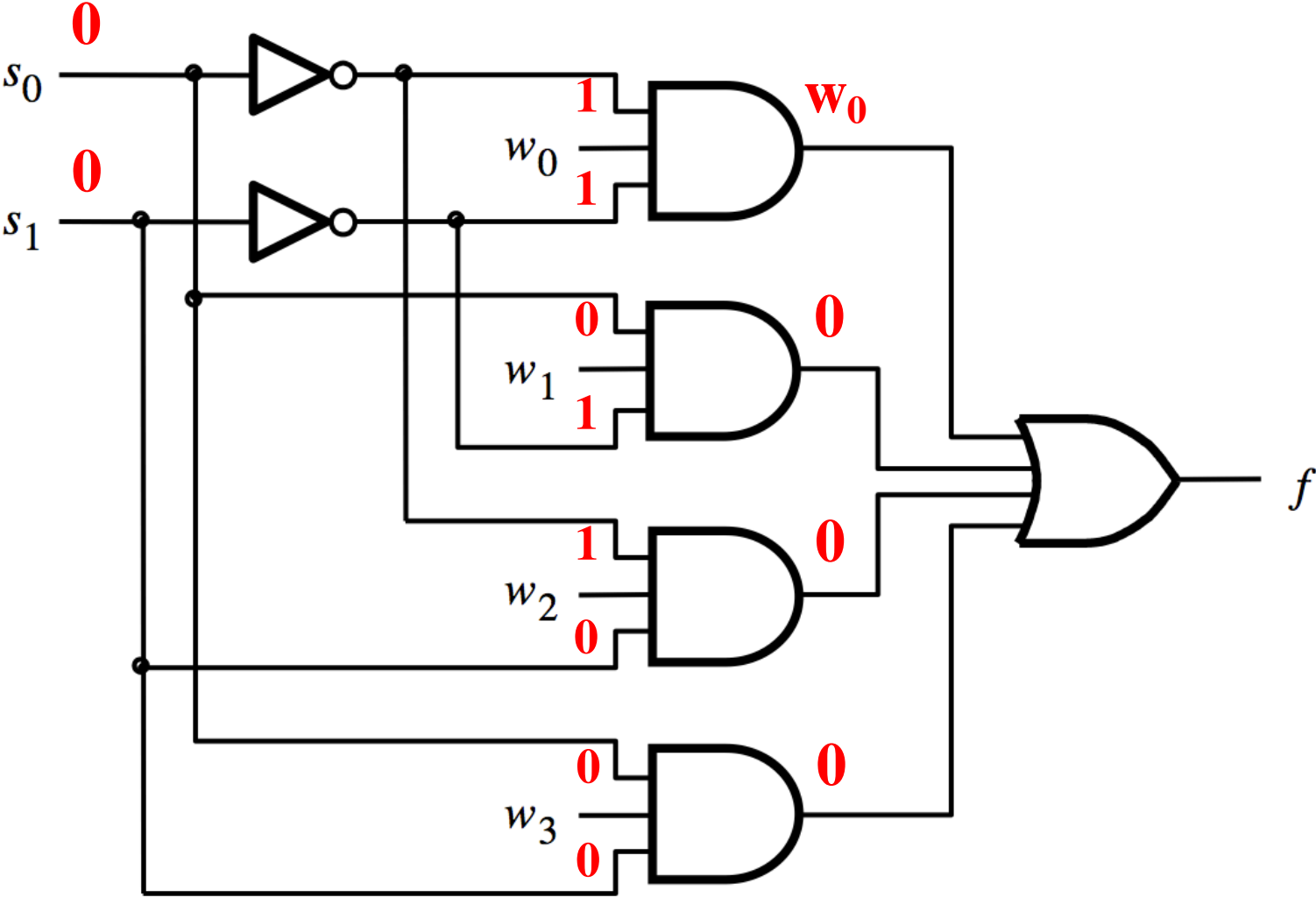
Analysis of the 4-to-1 Multiplexer ($s_1=0$ and $s_0=0$)



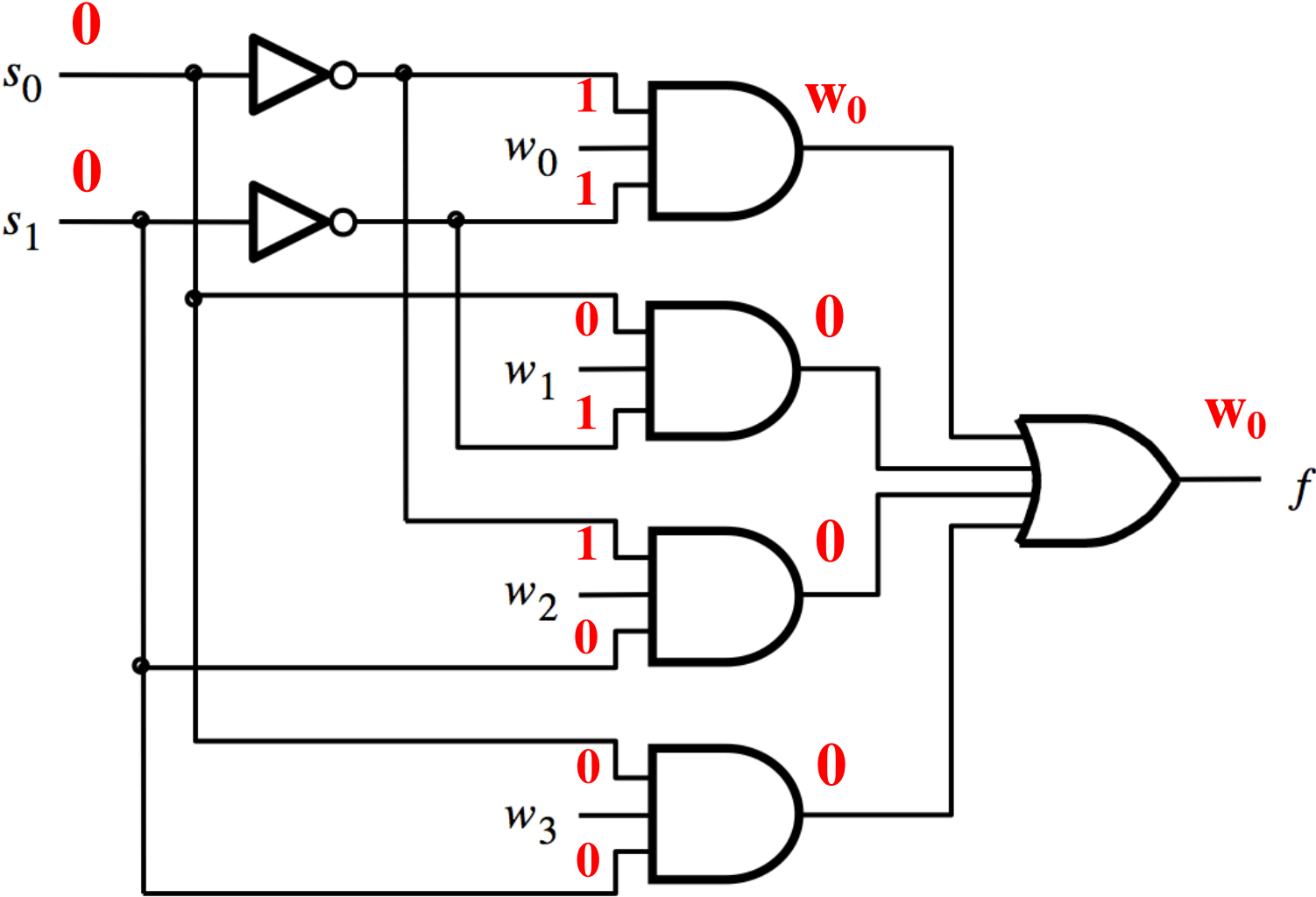
Analysis of the 4-to-1 Multiplexer ($s_1=0$ and $s_0=0$)



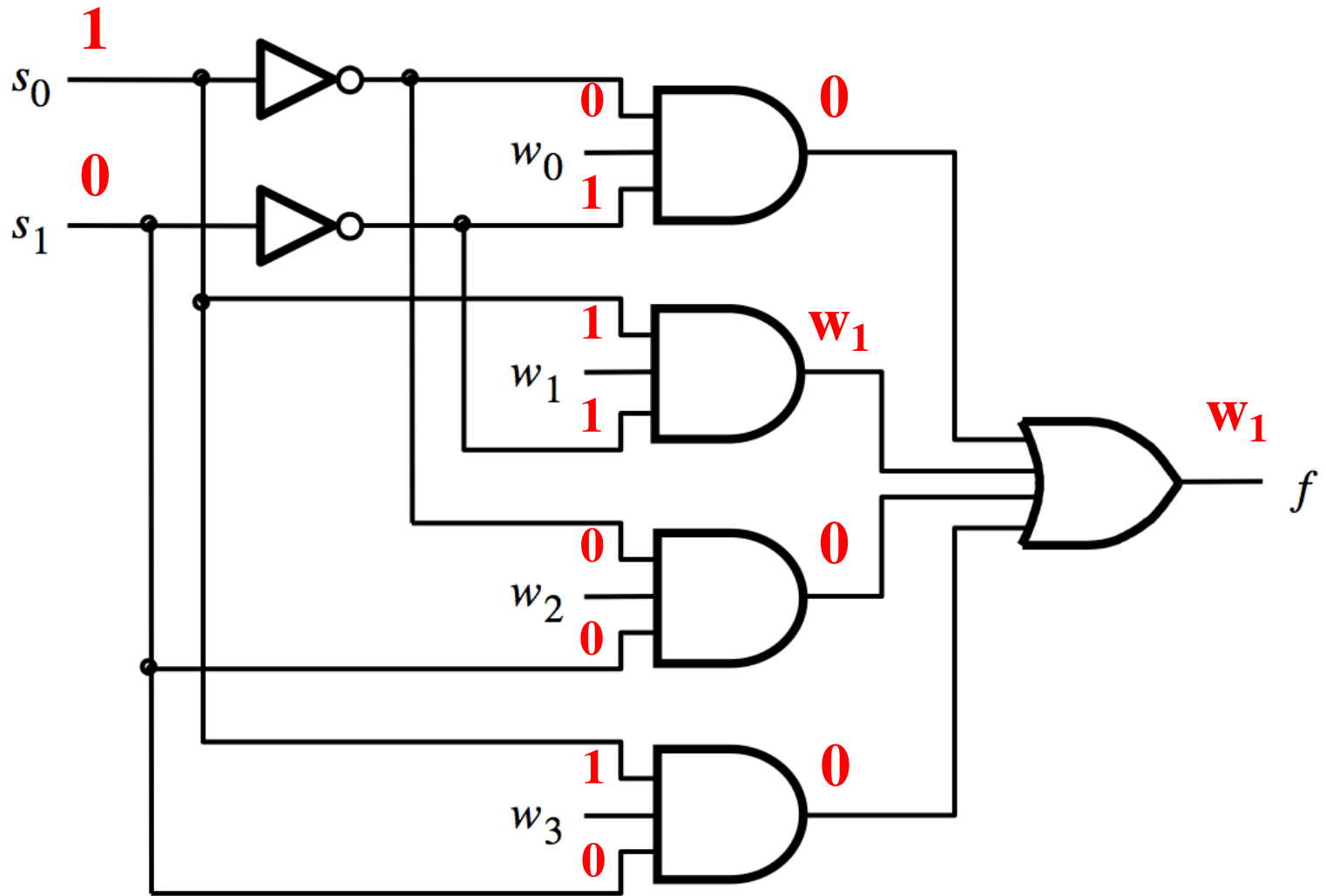
Analysis of the 4-to-1 Multiplexer ($s_1=0$ and $s_0=0$)



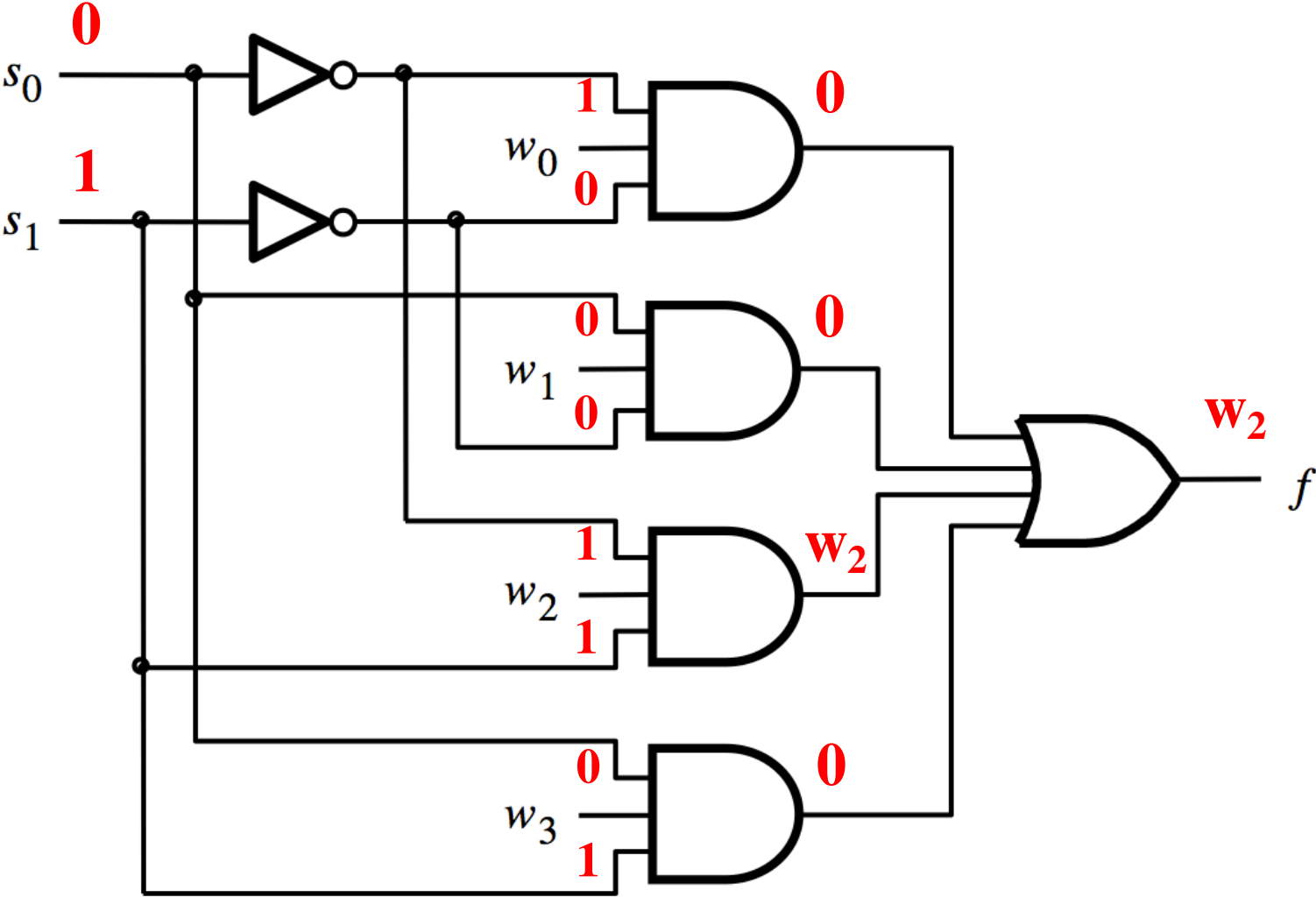
Analysis of the 4-to-1 Multiplexer ($s_1=0$ and $s_0=0$)



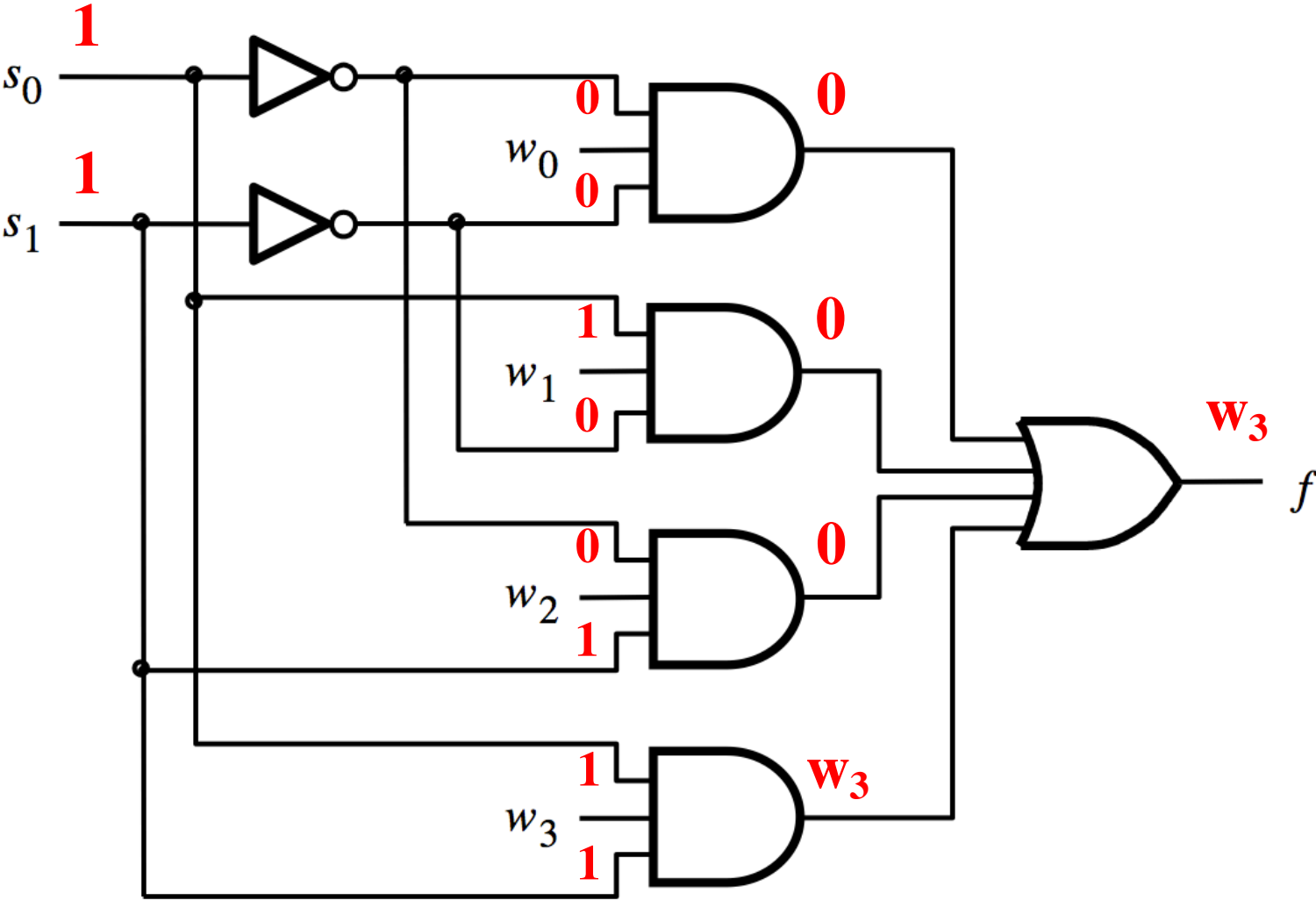
Analysis of the 4-to-1 Multiplexer ($s_1=0$ and $s_0=1$)



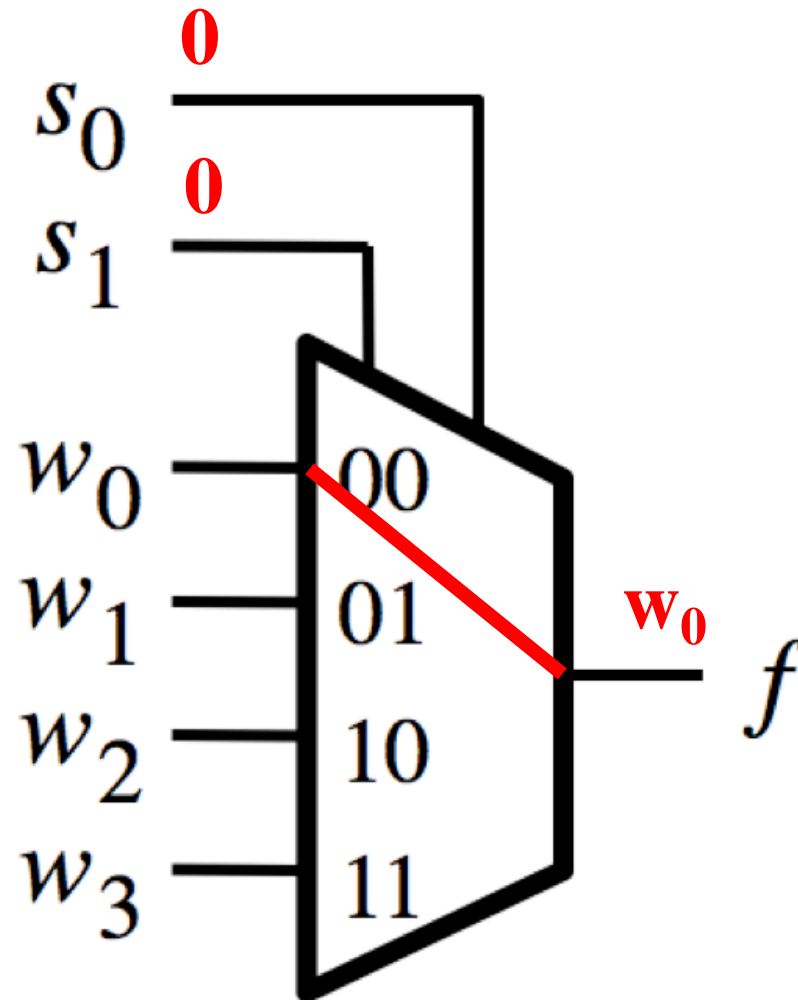
Analysis of the 4-to-1 Multiplexer ($s_1=1$ and $s_0=0$)



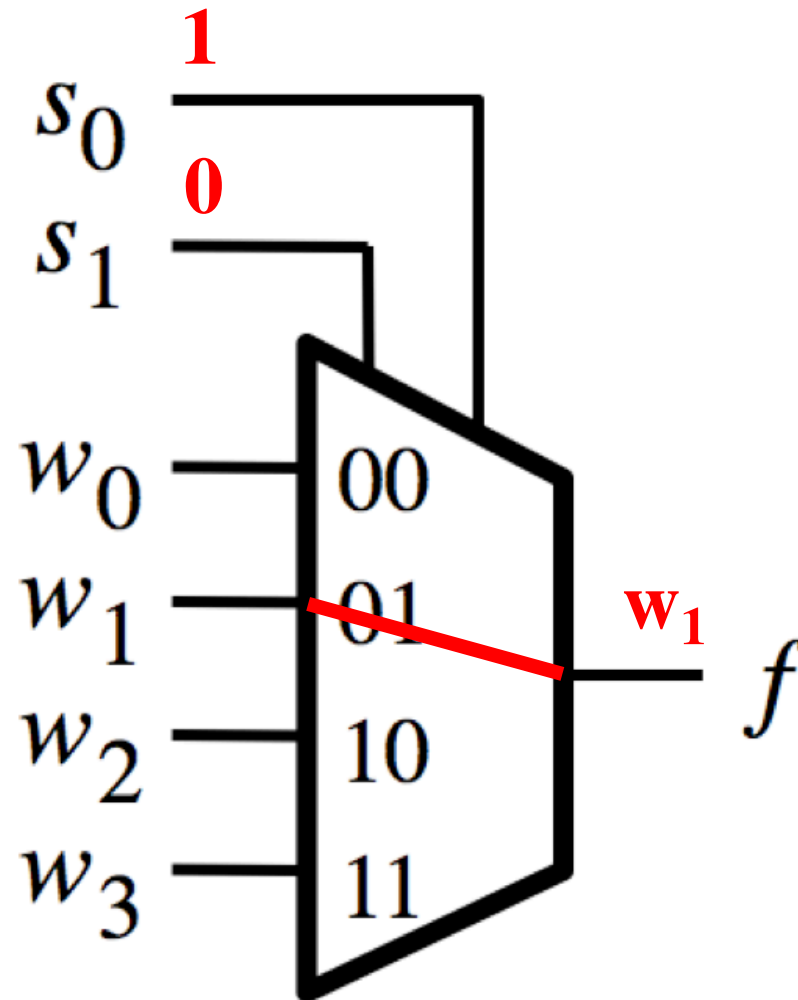
Analysis of the 4-to-1 Multiplexer ($s_1=1$ and $s_0=1$)



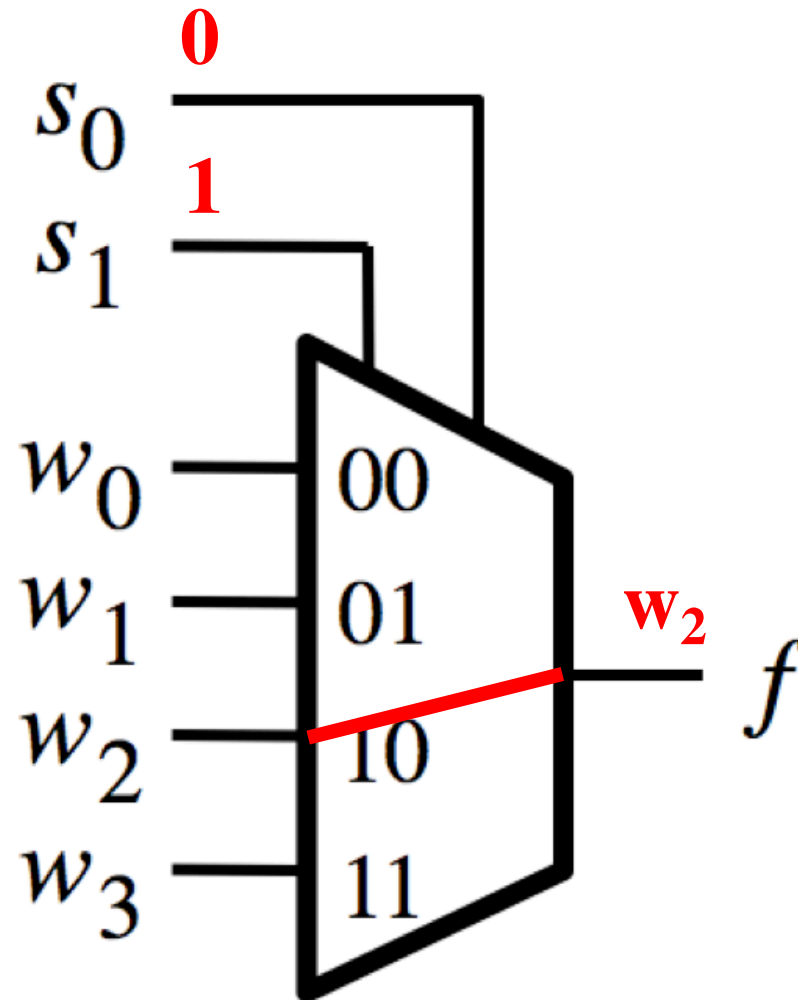
Analysis of the 4-to-1 Multiplexer ($s_1=0$ and $s_0=0$)



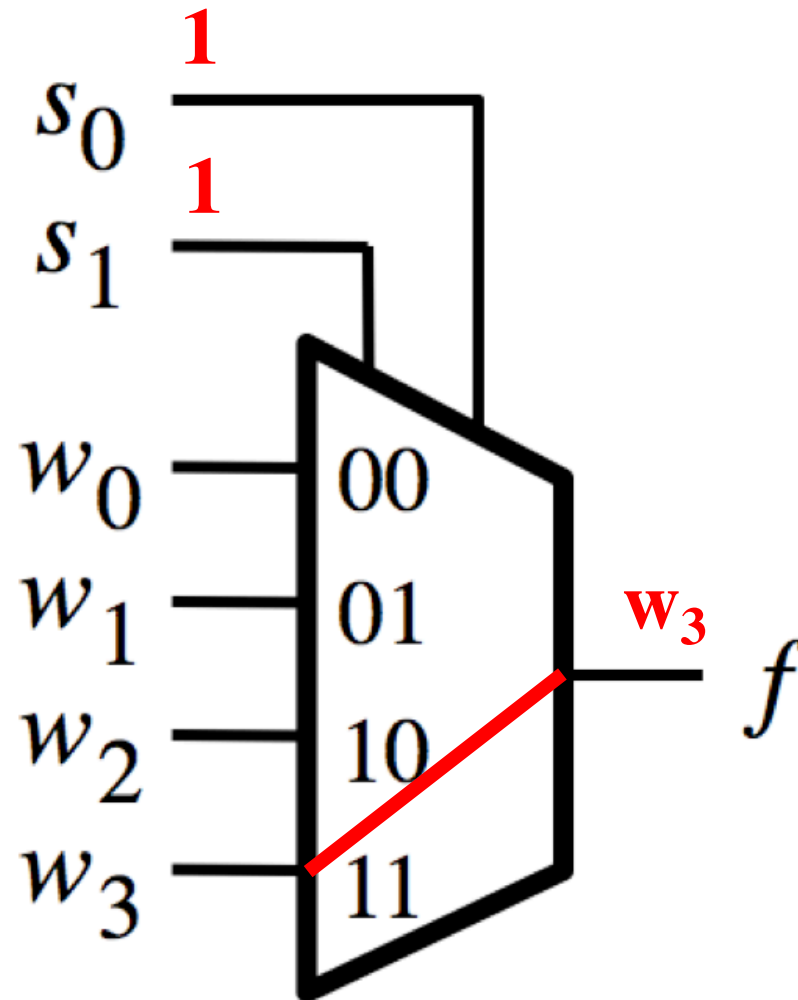
Analysis of the 4-to-1 Multiplexer ($s_1=0$ and $s_0=1$)



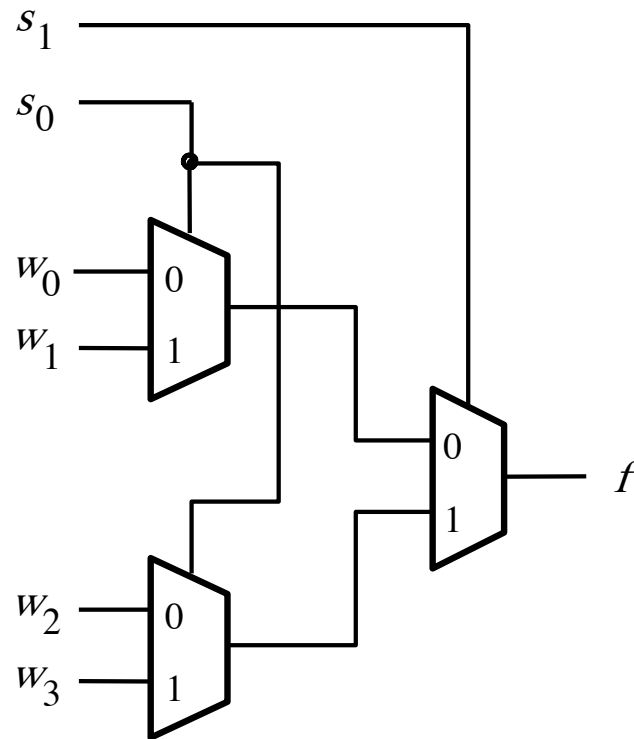
Analysis of the 4-to-1 Multiplexer ($s_1=1$ and $s_0=0$)



Analysis of the 4-to-1 Multiplexer ($s_1=1$ and $s_0=1$)



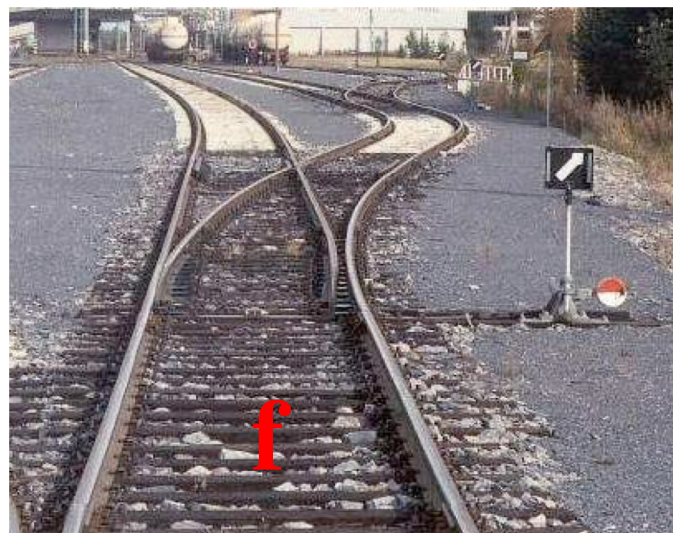
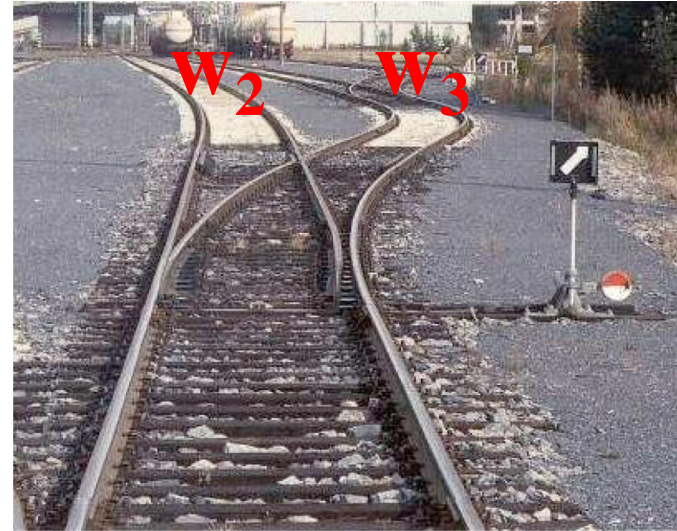
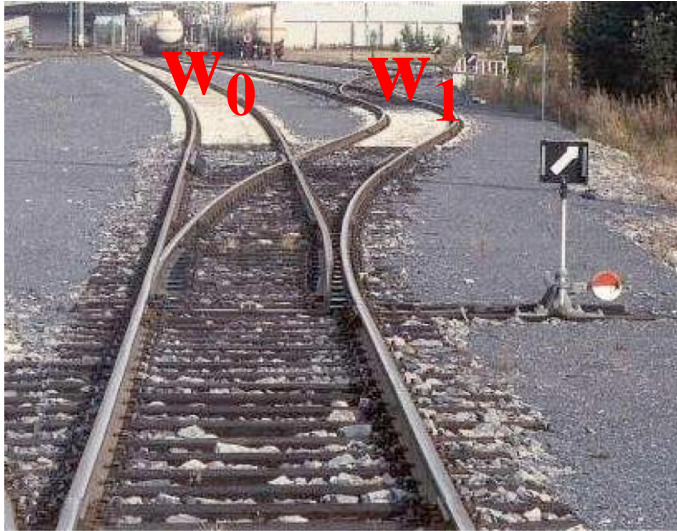
Using three 2-to-1 multiplexers to build one 4-to-1 multiplexer



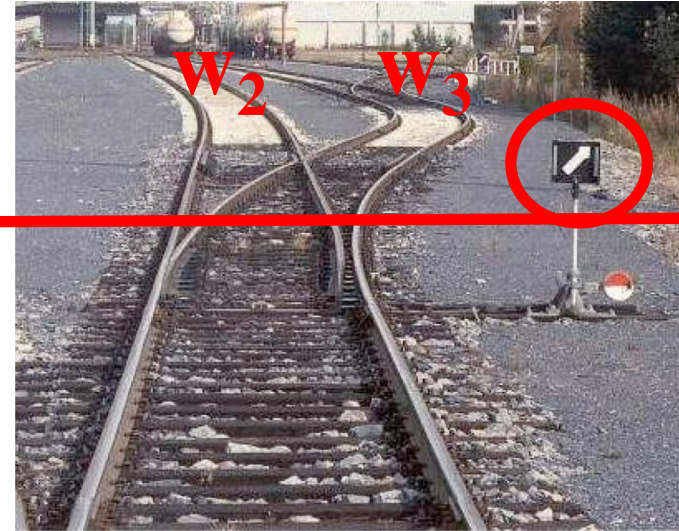
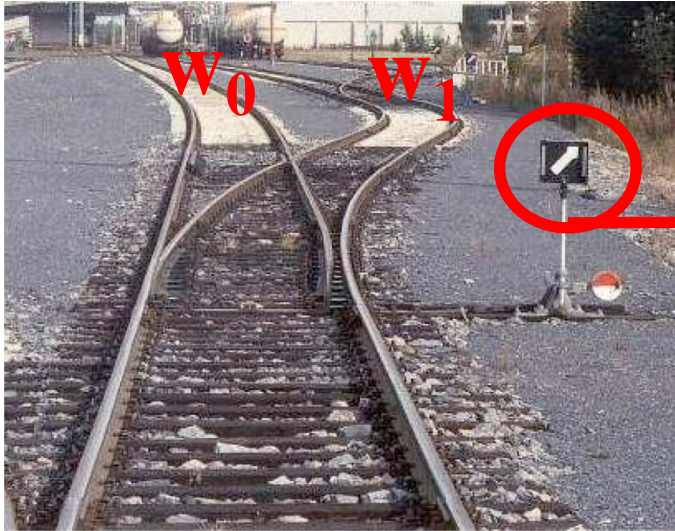
Analogy: Railroad Switches



Analogy: Railroad Switches

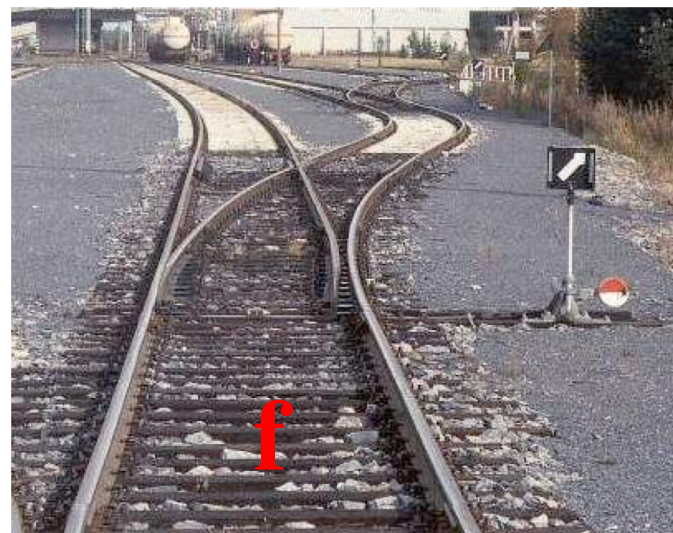


Analogy: Railroad Switches



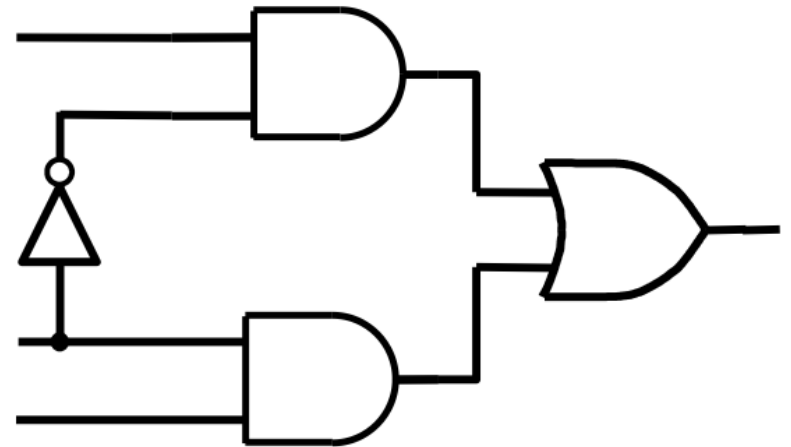
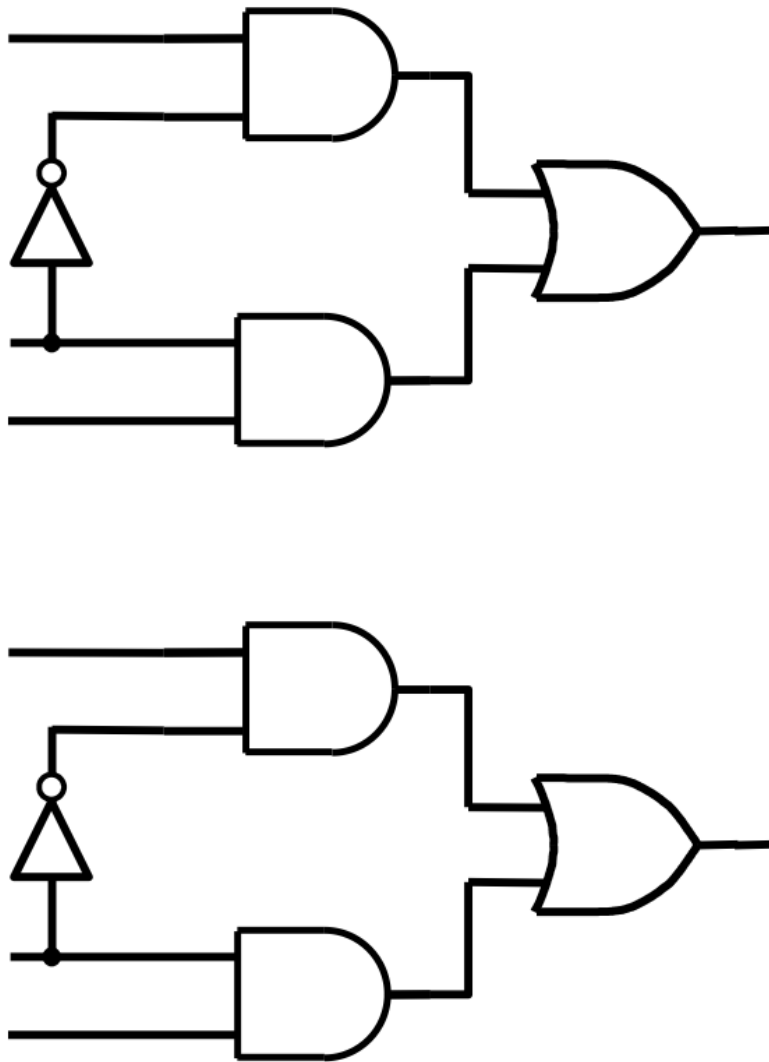
S_0

these two
switches are
controlled
together

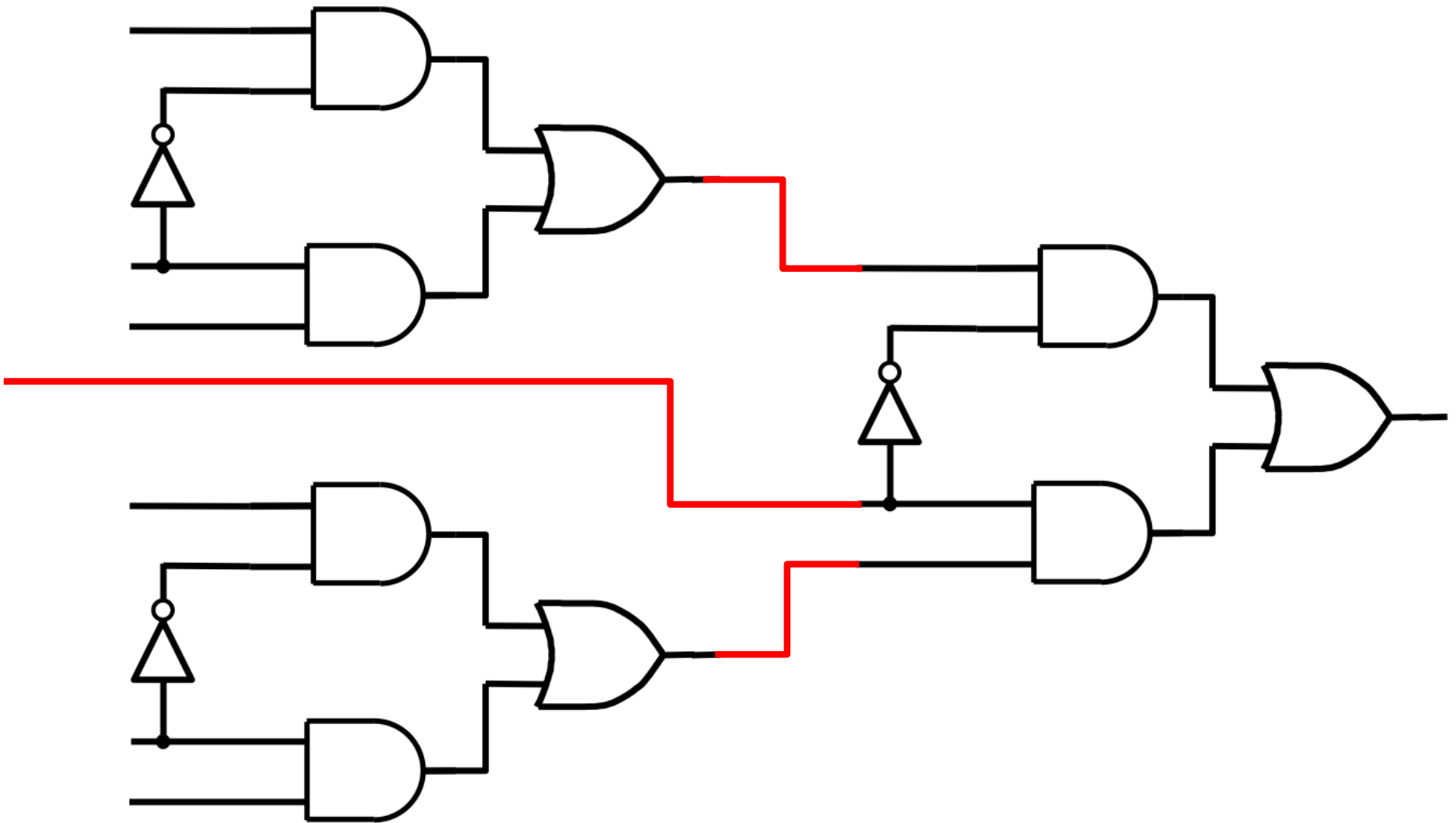


S_1

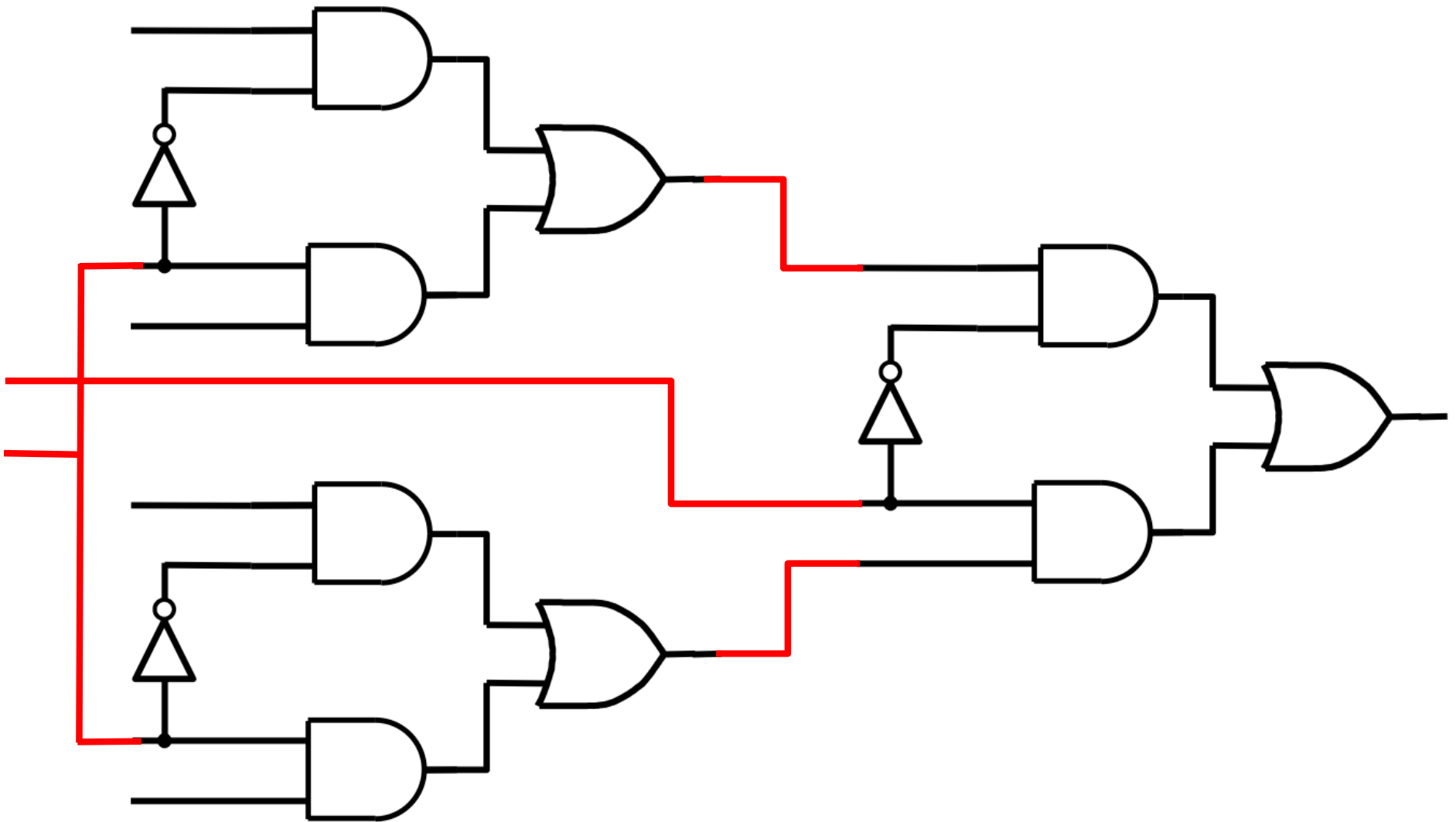
Using three 2-to-1 multiplexers to build one 4-to-1 multiplexer



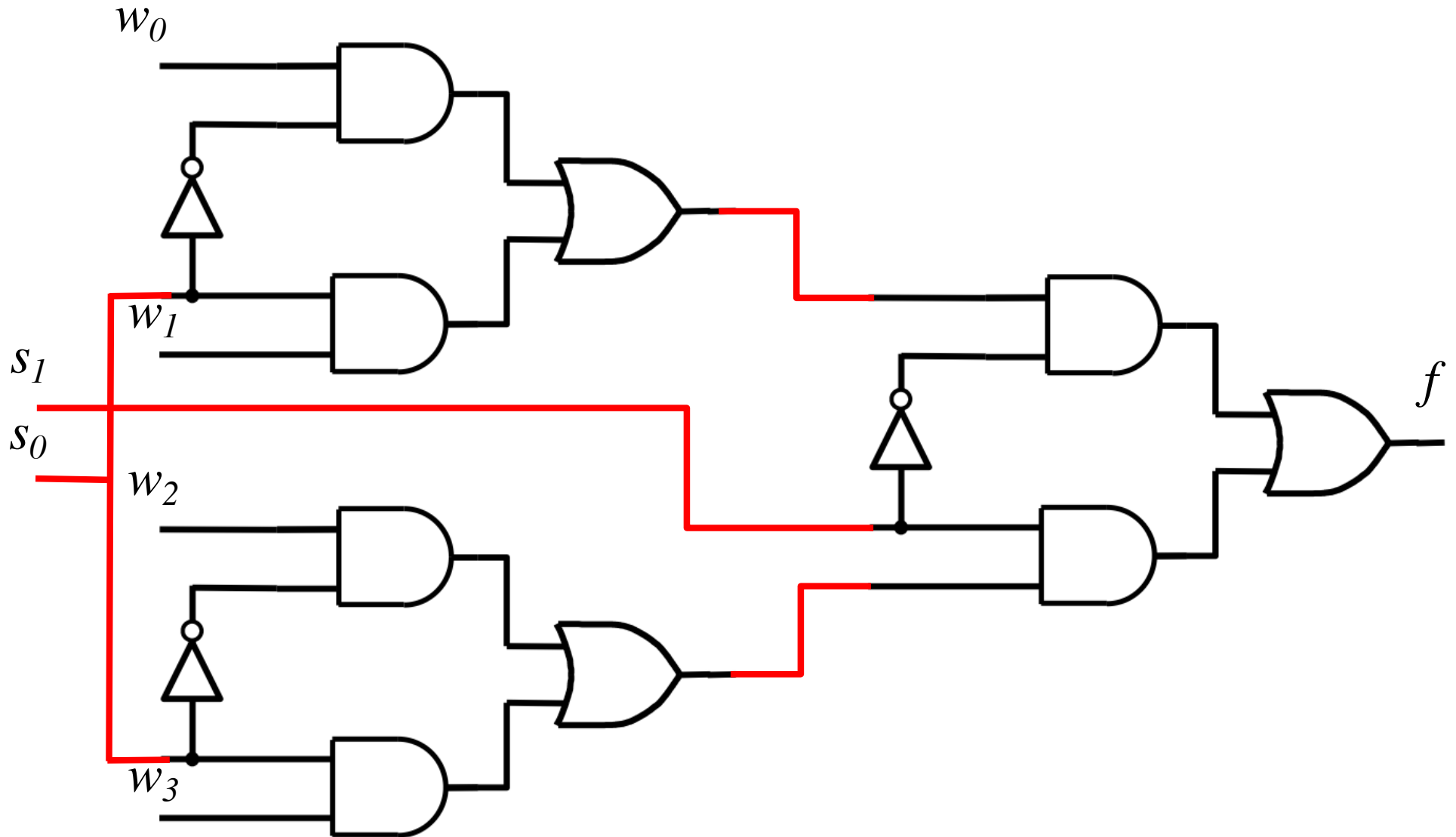
Using three 2-to-1 multiplexers to build one 4-to-1 multiplexer



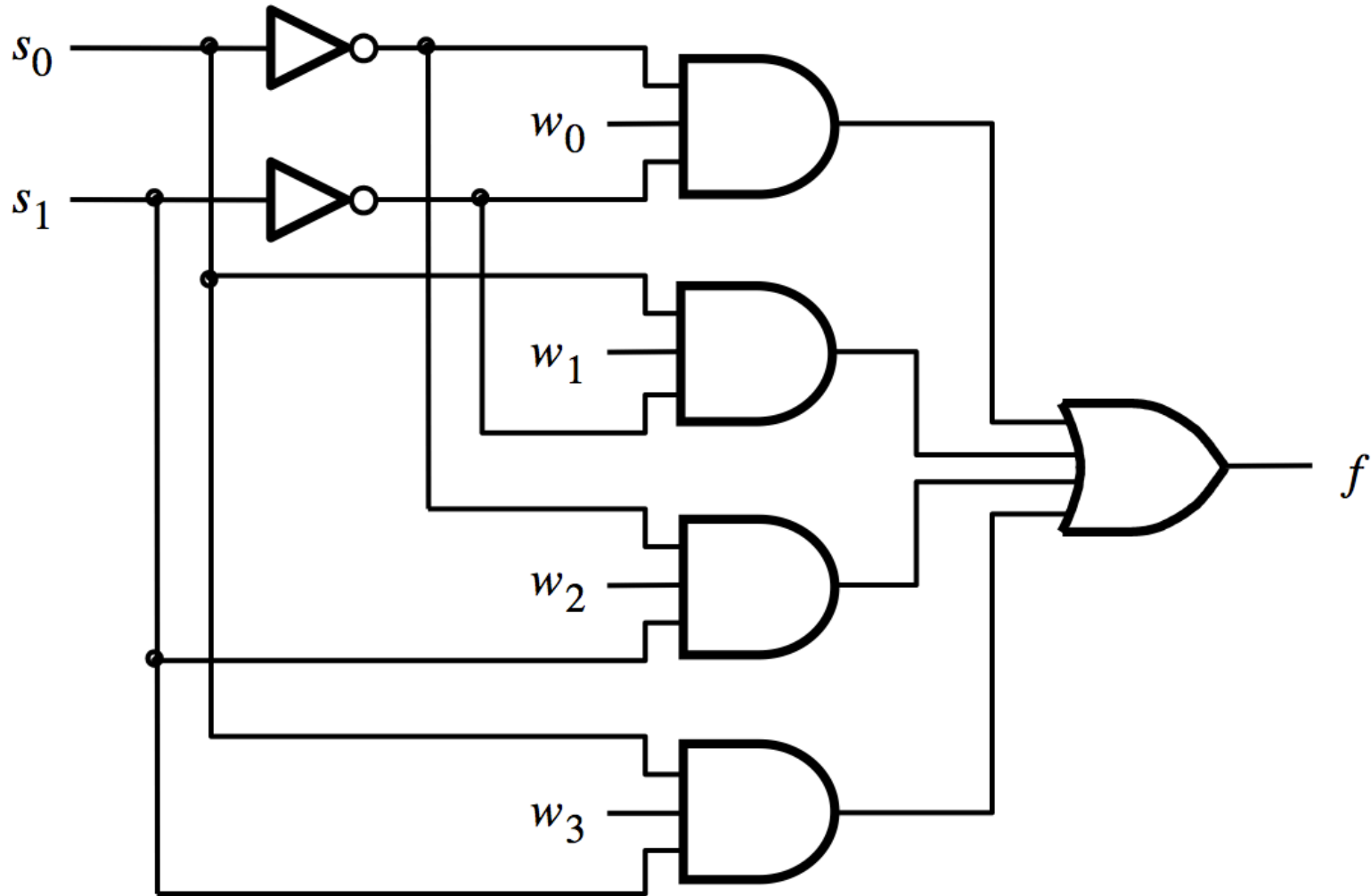
Using three 2-to-1 multiplexers to build one 4-to-1 multiplexer



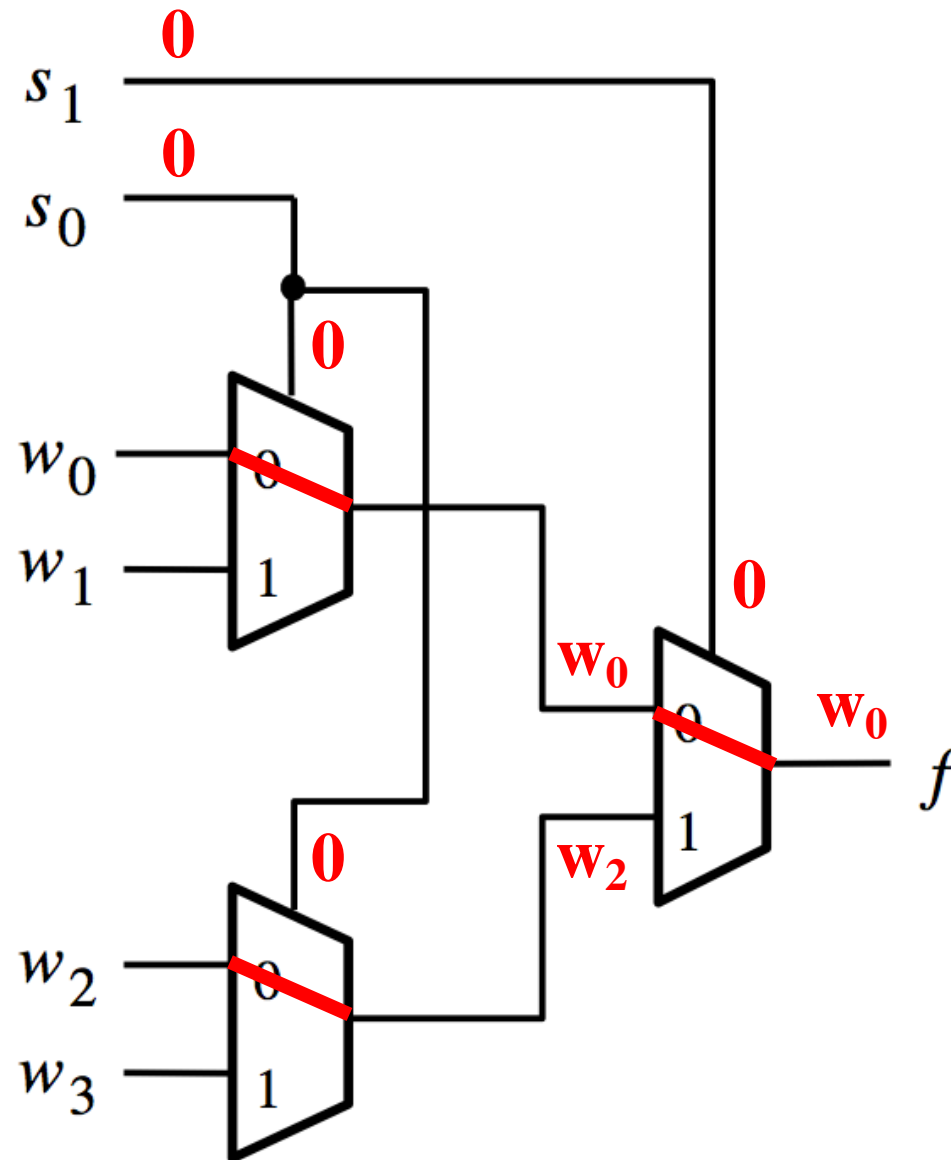
Using three 2-to-1 multiplexers to build one 4-to-1 multiplexer



That is different from the SOP form of the 4-to-1 multiplexer shown below

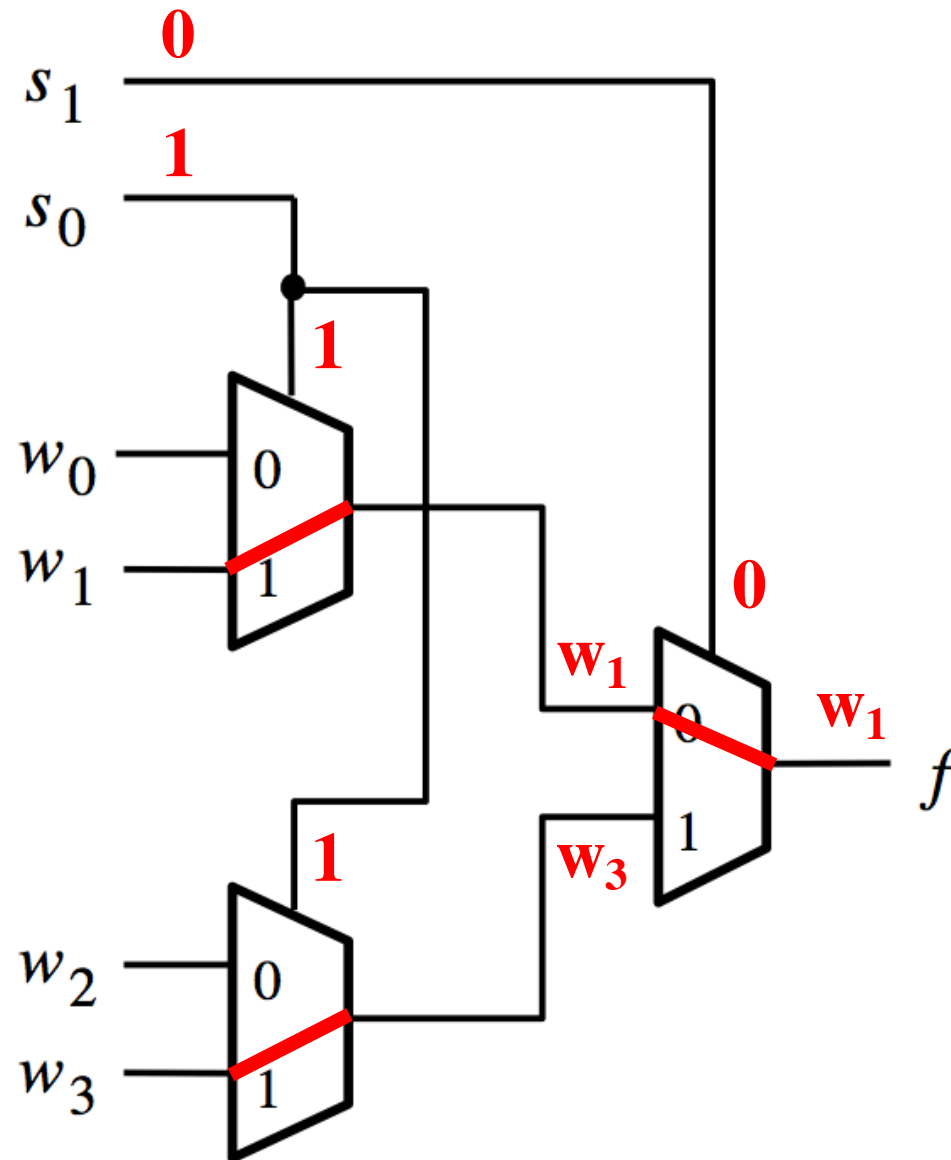


Analysis of the Hierarchical Implementation ($s_1=0$ and $s_0=0$)



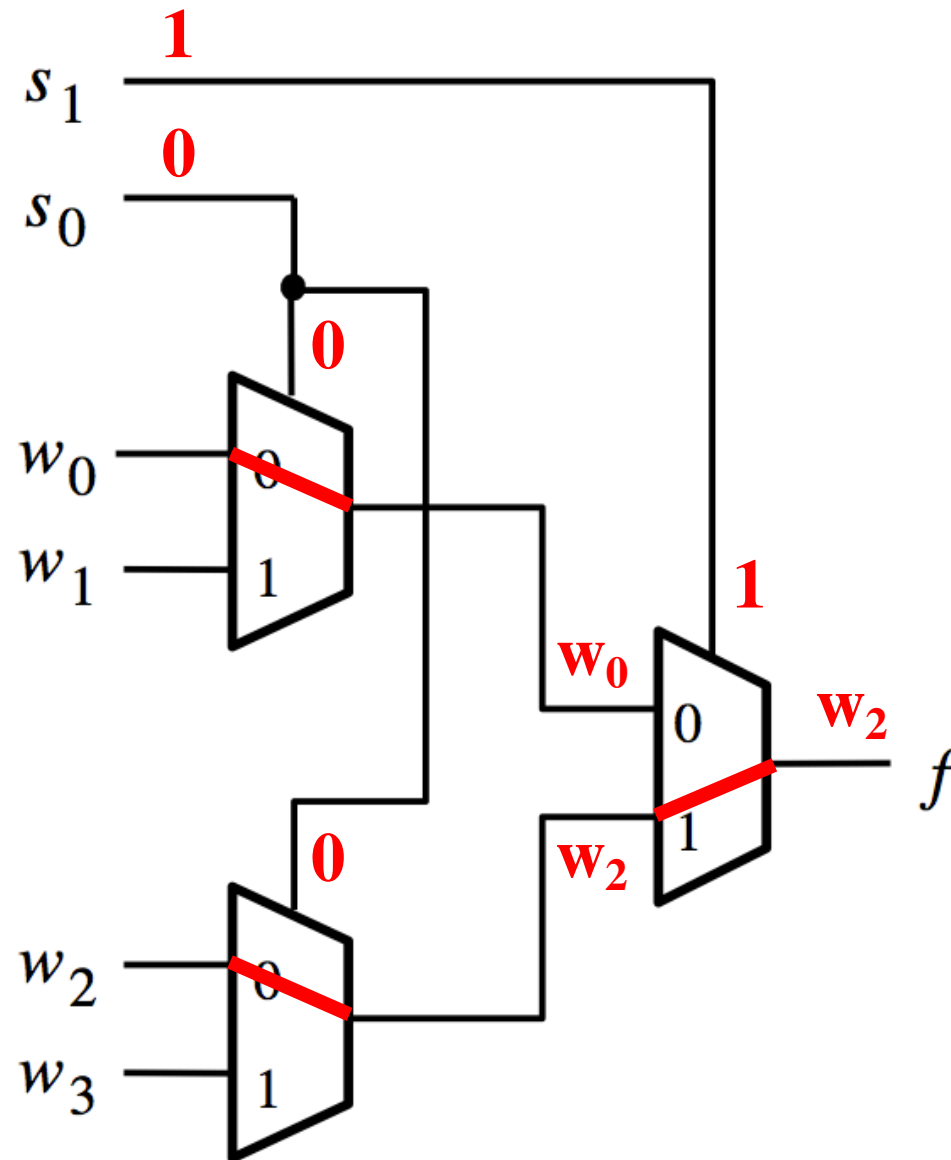
[Figure 4.3 from the textbook]

Analysis of the Hierarchical Implementation ($s_1=0$ and $s_0=1$)



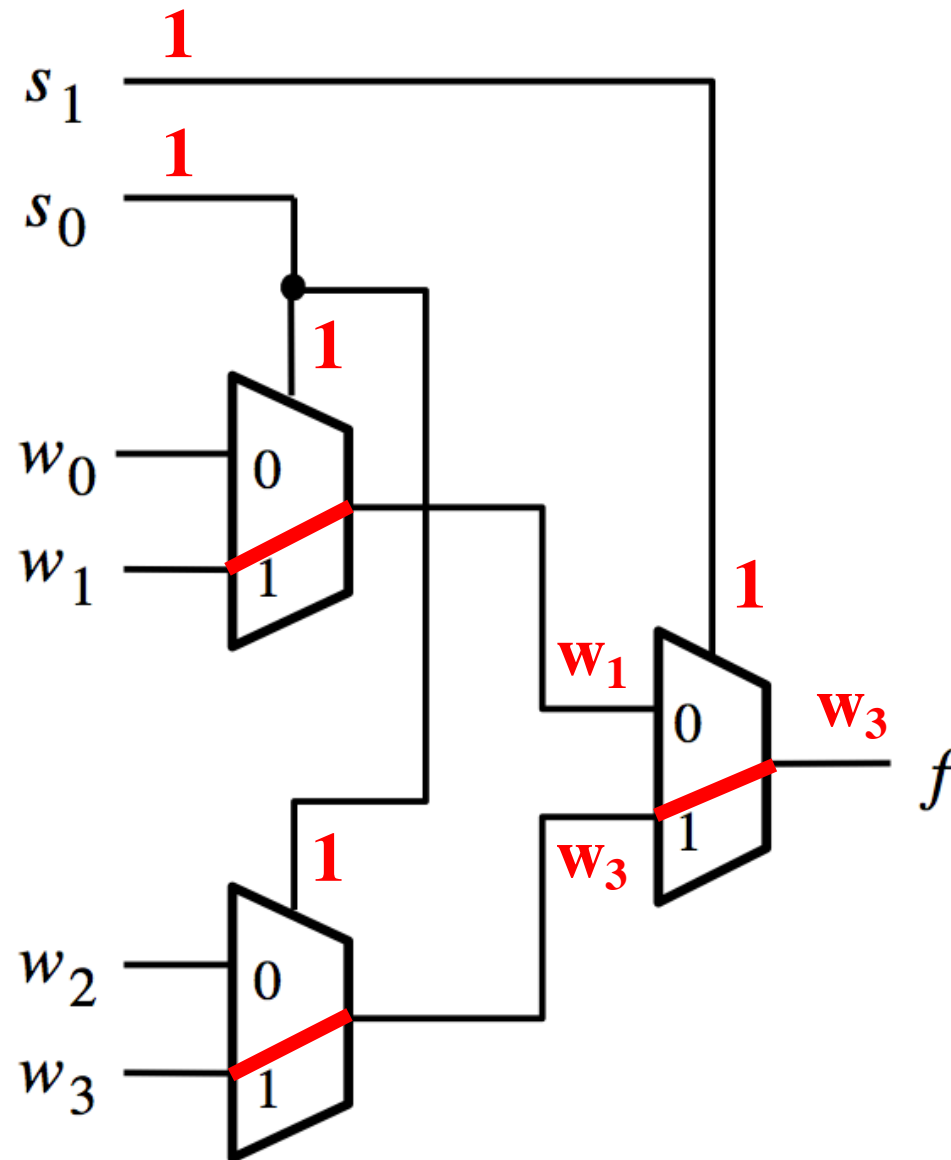
[Figure 4.3 from the textbook]

Analysis of the Hierarchical Implementation ($s_1=1$ and $s_0=0$)



[Figure 4.3 from the textbook]

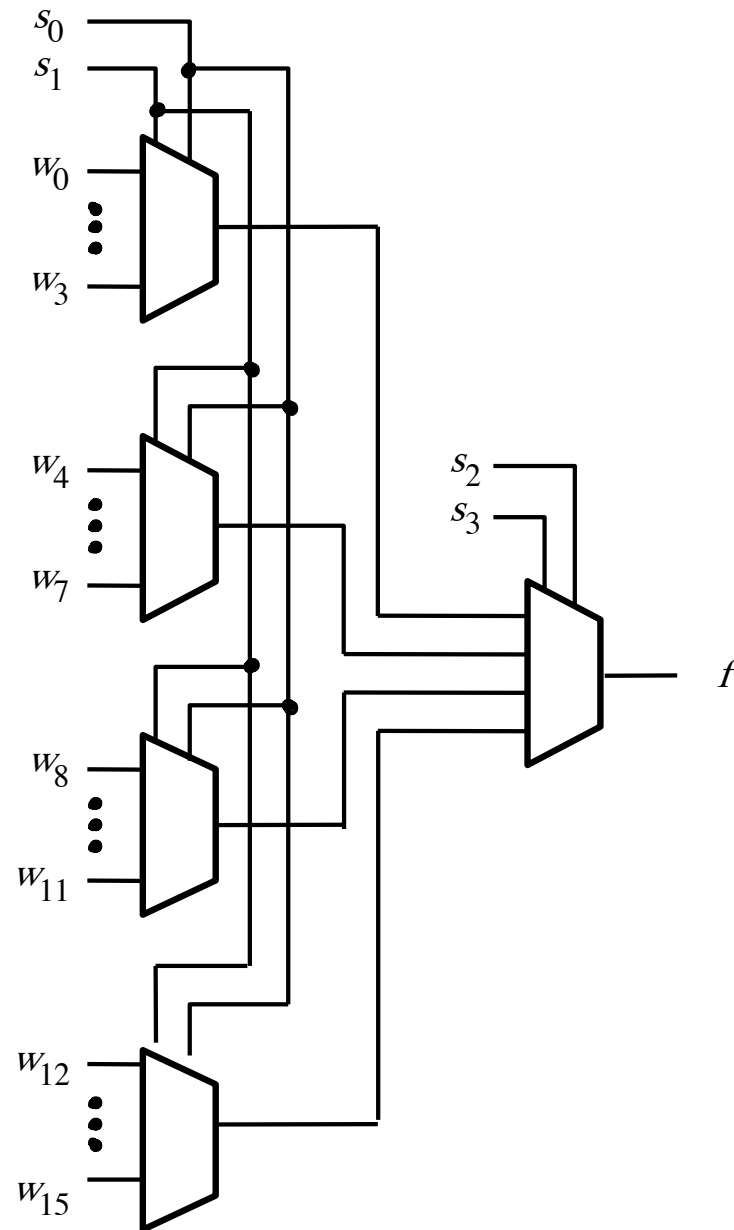
Analysis of the Hierarchical Implementation ($s_1=1$ and $s_0=1$)



[Figure 4.3 from the textbook]

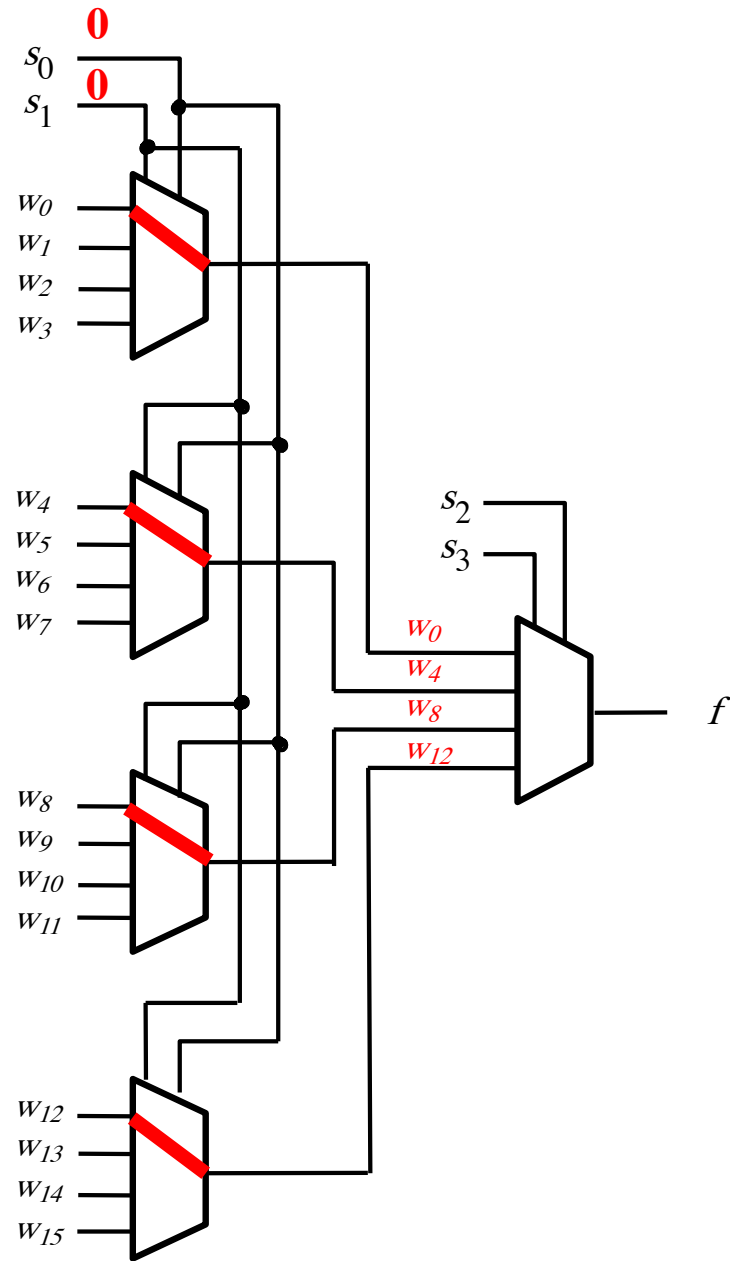
16-to-1 Multiplexer

16-1 Multiplexer

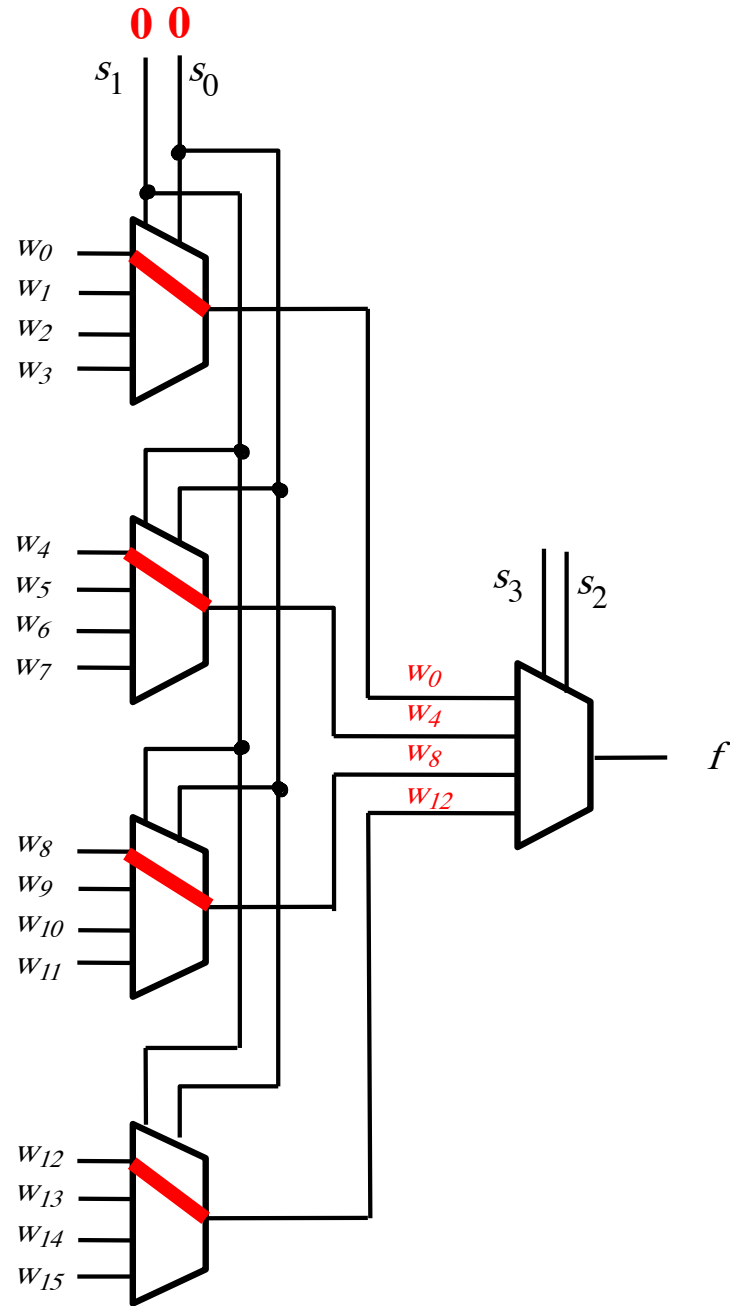


[Figure 4.4 from the textbook]

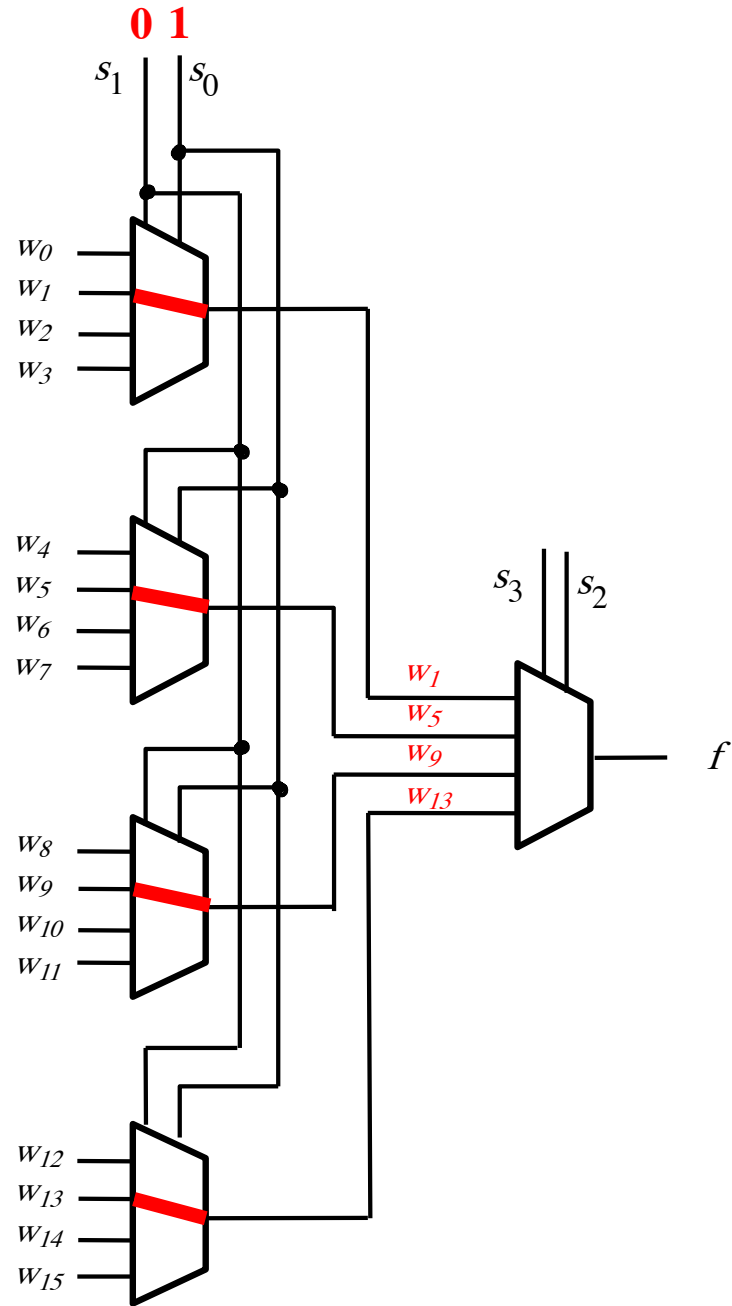
16-1 Multiplexer



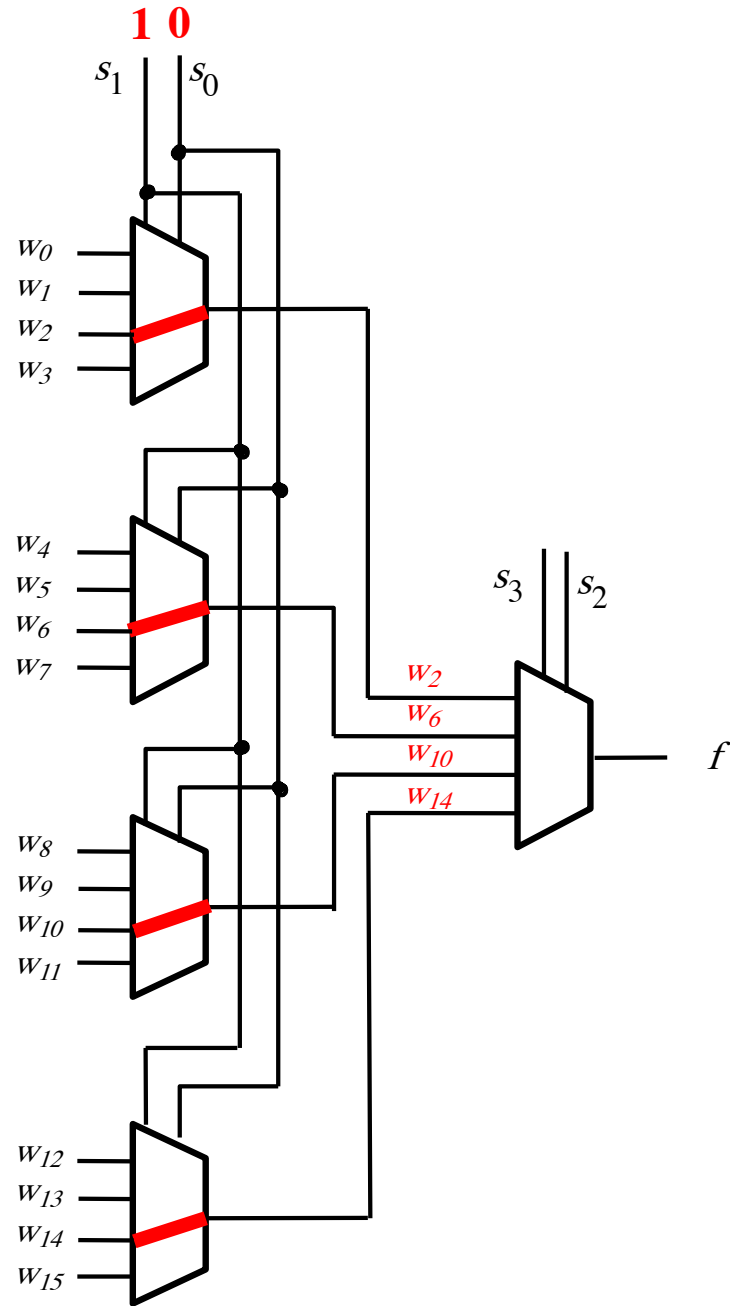
16-1 Multiplexer



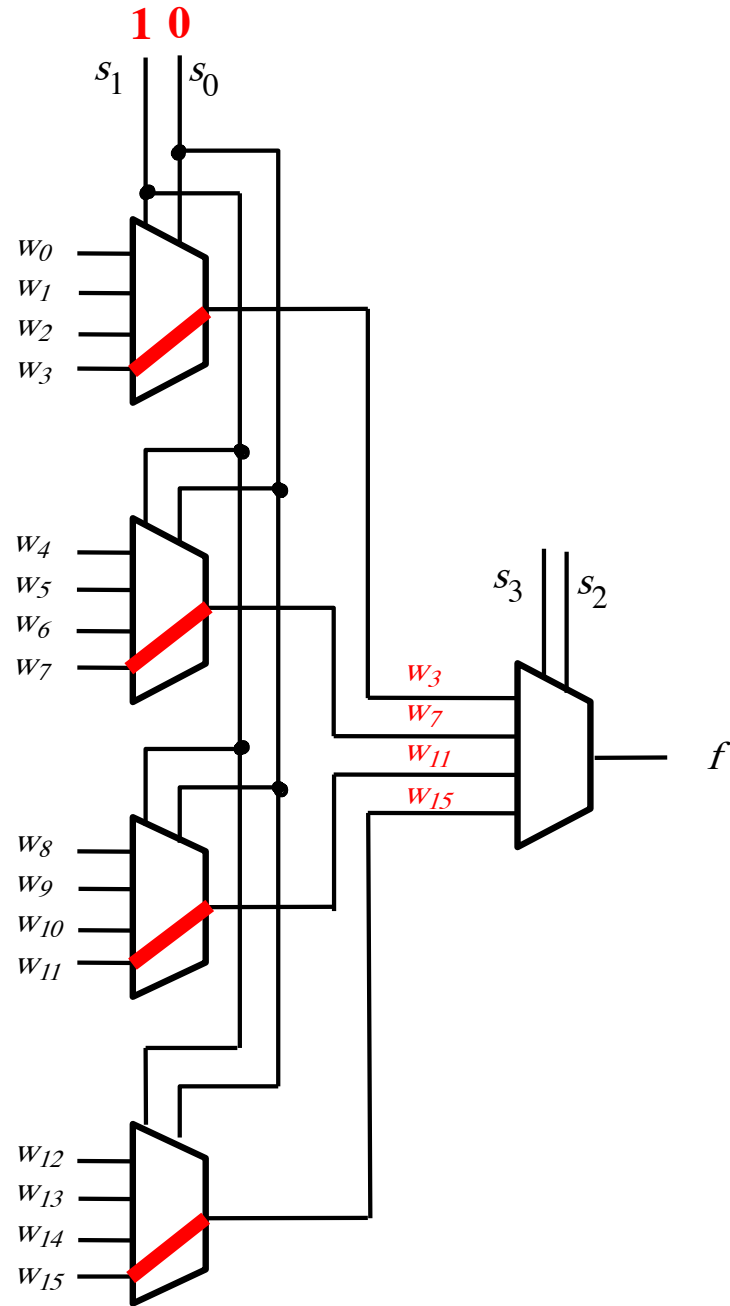
16-1 Multiplexer



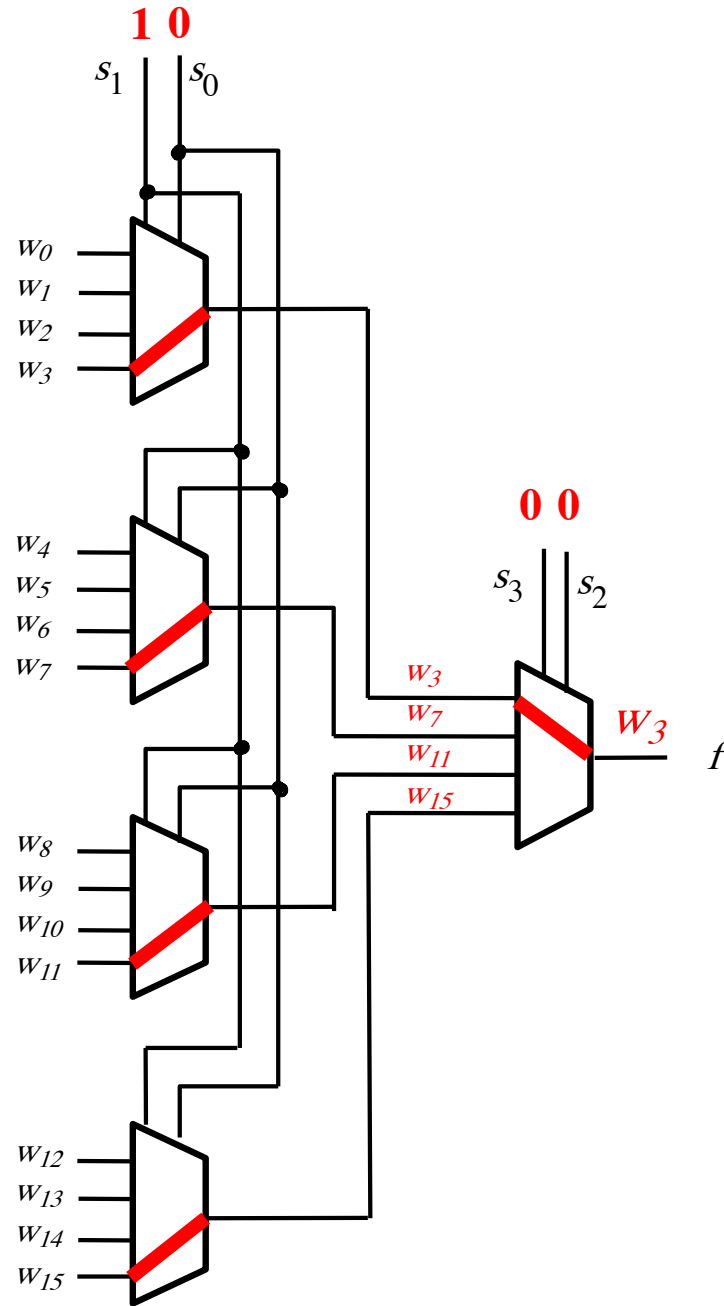
16-1 Multiplexer



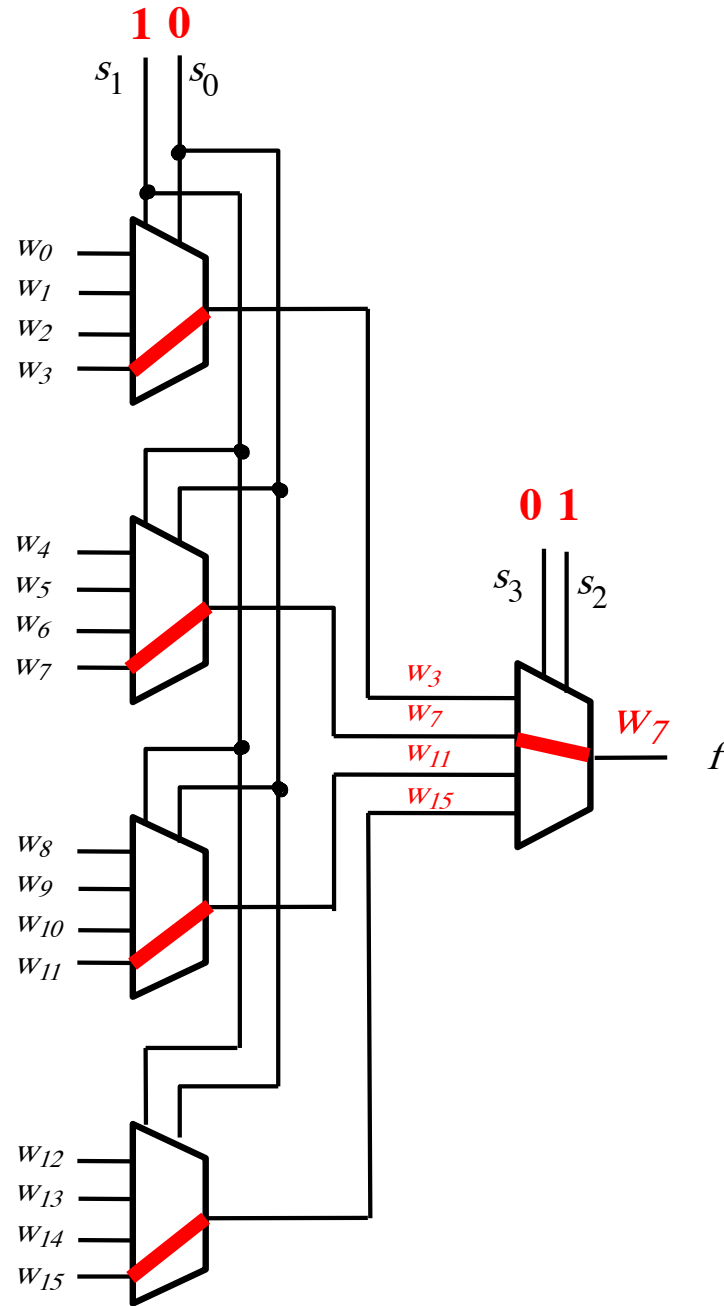
16-1 Multiplexer



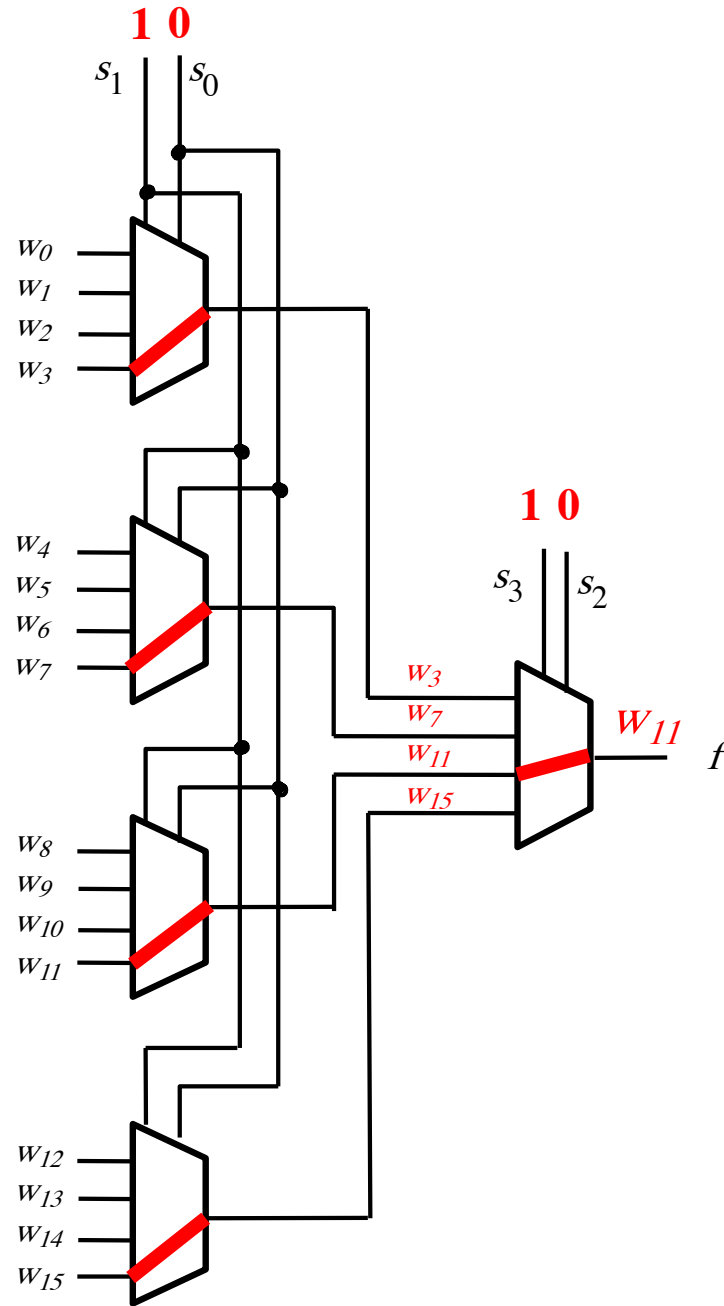
16-1 Multiplexer



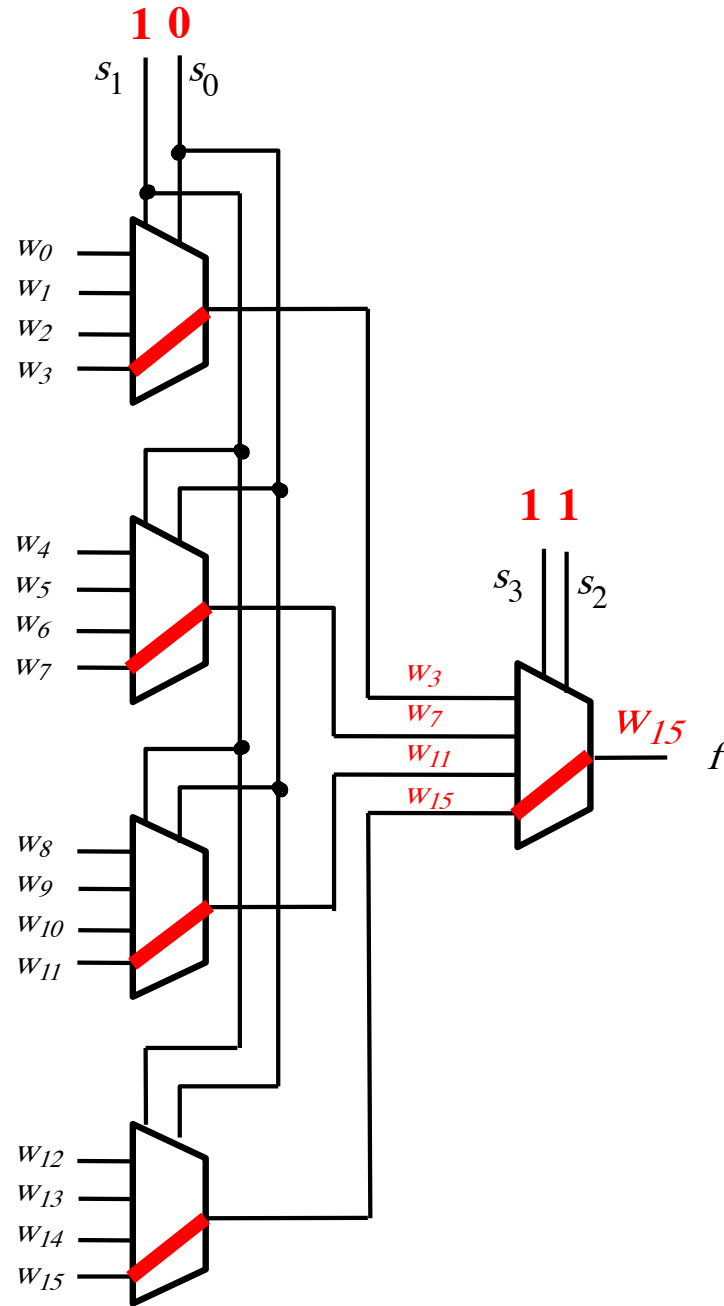
16-1 Multiplexer



16-1 Multiplexer



16-1 Multiplexer

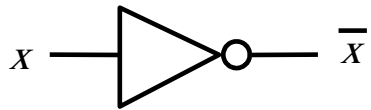




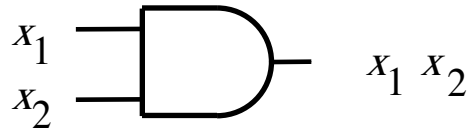
[<http://upload.wikimedia.org/wikipedia/commons/2/26/SunsetTracksCrop.JPG>]

Multiplexers Are Special

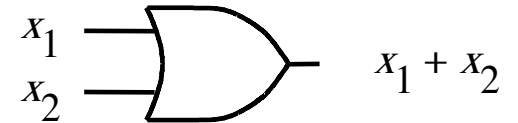
The Three Basic Logic Gates



NOT gate

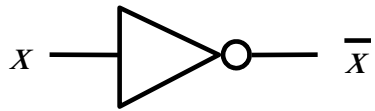


AND gate



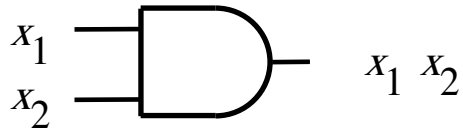
OR gate

Truth Table for NOT



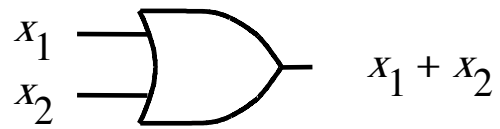
x	\bar{x}
0	1
1	0

Truth Table for AND



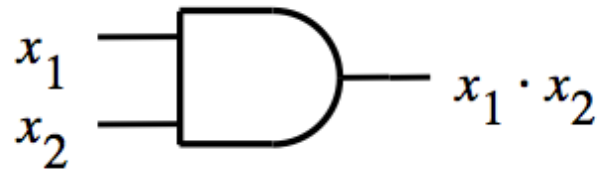
x_1	x_2	$x_1 \cdot x_2$
0	0	0
0	1	0
1	0	0
1	1	1

Truth Table for OR

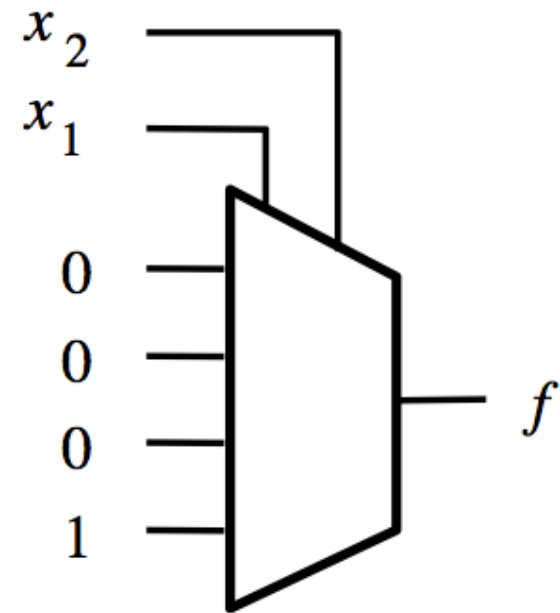


x_1	x_2	$x_1 + x_2$
0	0	0
0	1	1
1	0	1
1	1	1

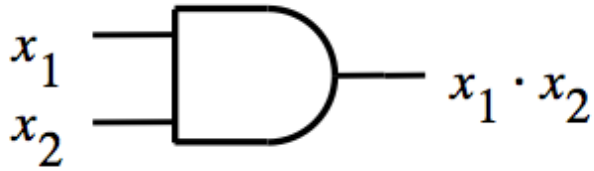
Building an AND Gate with 4-to-1 Mux



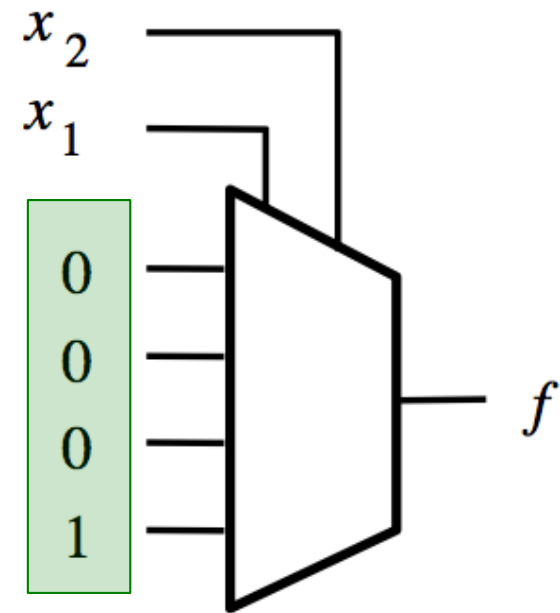
x_1	x_2	$x_1 \cdot x_2$
0	0	0
0	1	0
1	0	0
1	1	1



Building an AND Gate with 4-to-1 Mux

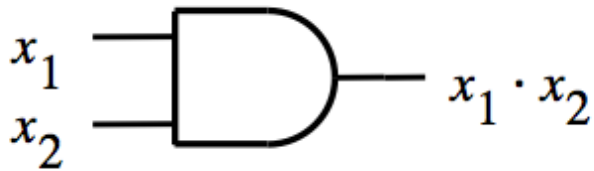


x_1	x_2	$x_1 \cdot x_2$
0	0	0
0	1	0
1	0	0
1	1	1

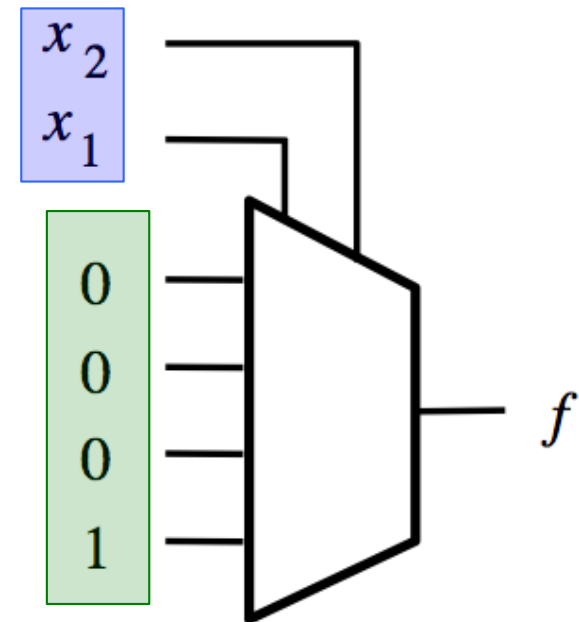


These two are the same.

Building an AND Gate with 4-to-1 Mux



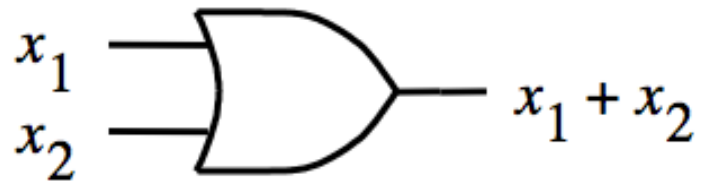
x_1	x_2	$x_1 \cdot x_2$
0	0	0
0	1	0
1	0	0
1	1	1



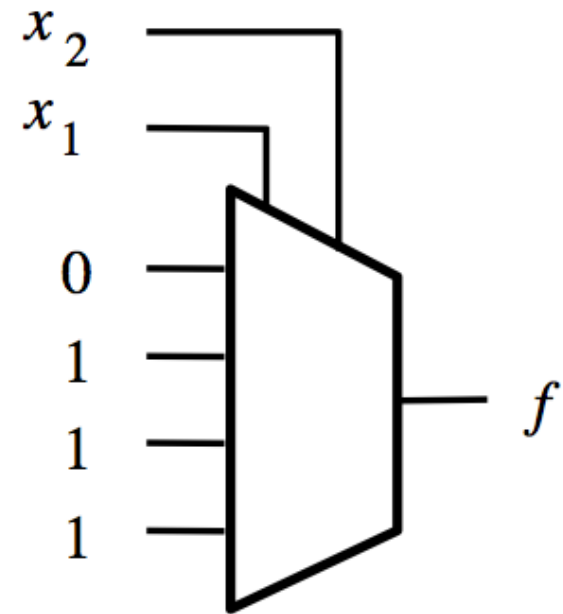
These two are the same.

And so are these two.

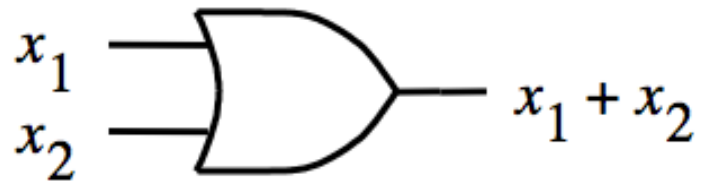
Building an OR Gate with 4-to-1 Mux



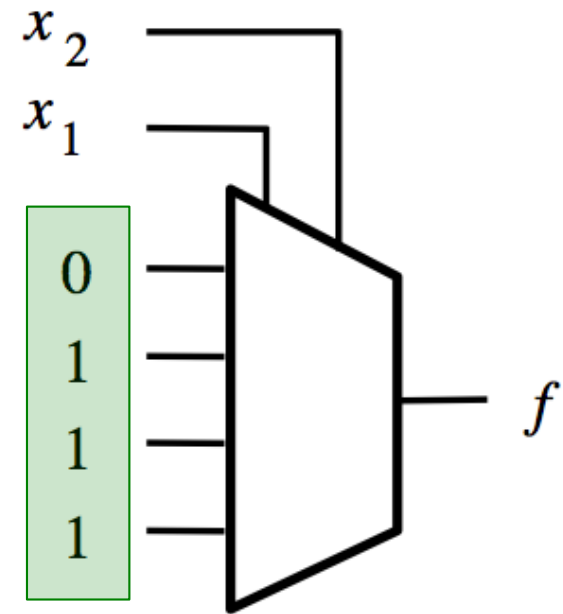
x_1	x_2	$x_1 + x_2$
0	0	0
0	1	1
1	0	1
1	1	1



Building an OR Gate with 4-to-1 Mux

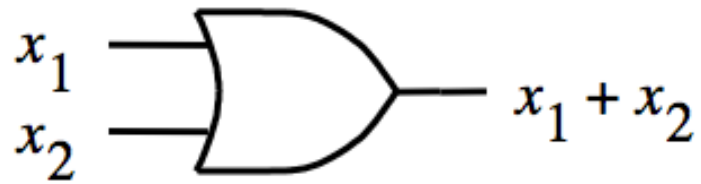


x_1	x_2	$x_1 + x_2$
0	0	0
0	1	1
1	0	1
1	1	1

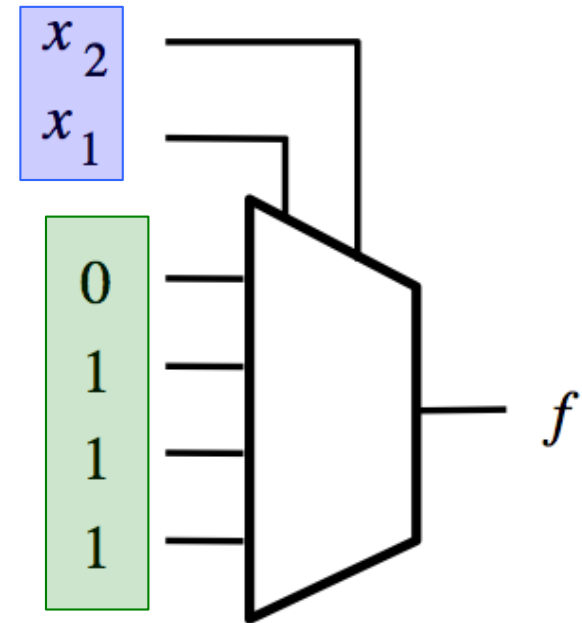


These two are the same.

Building an OR Gate with 4-to-1 Mux



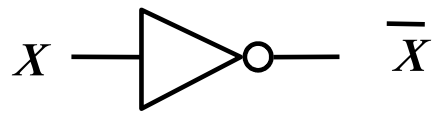
x_1	x_2	$x_1 + x_2$
0	0	0
0	1	1
1	0	1
1	1	1



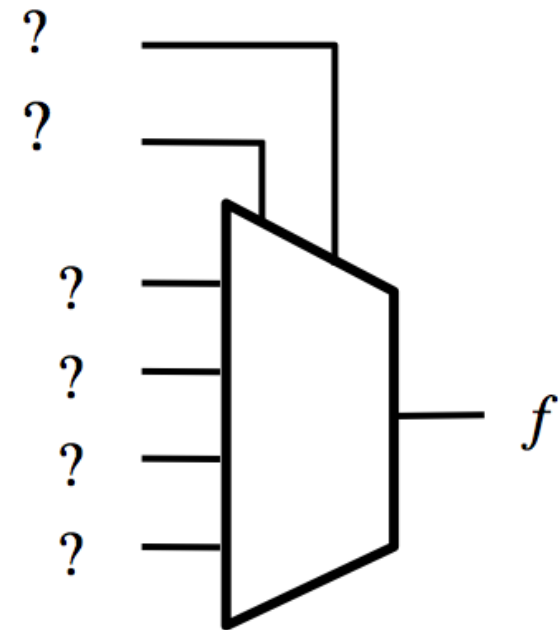
These two are the same.

And so are these two.

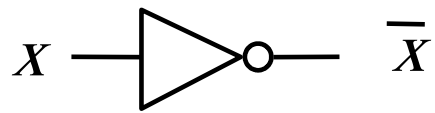
Building a NOT Gate with 4-to-1 Mux



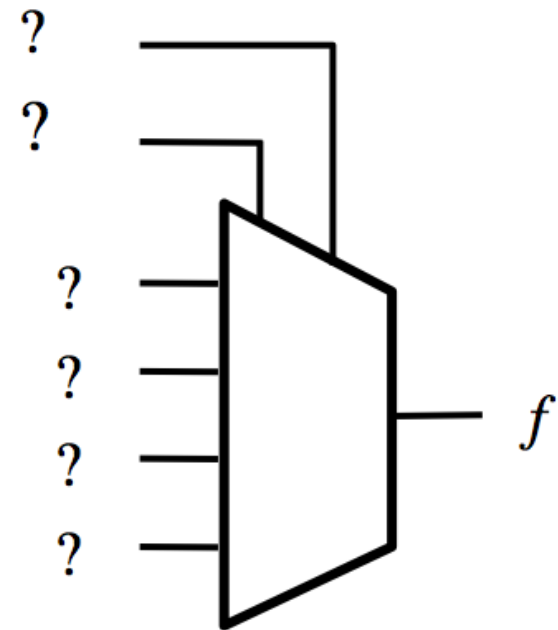
x	\bar{x}
0	1
1	0



Building a NOT Gate with 4-to-1 Mux

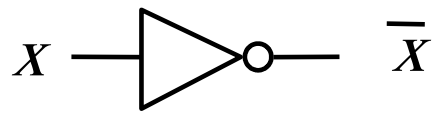


x	y	f
0	0	1
0	1	1
1	0	0
1	1	0

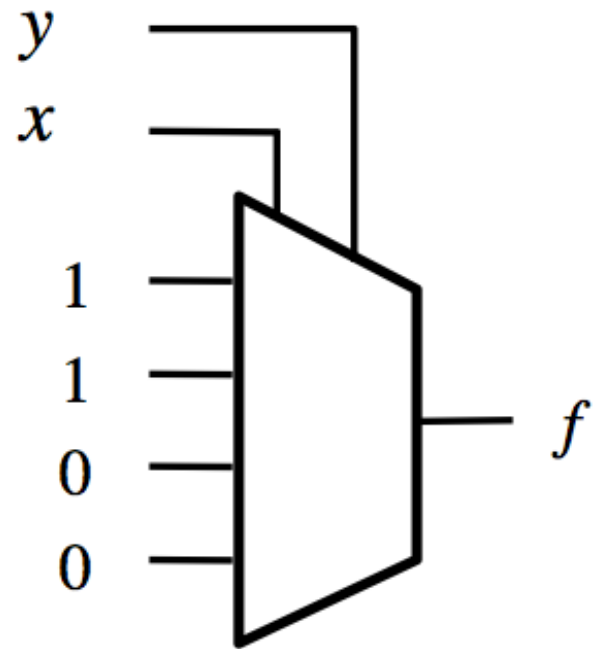


Introduce a dummy variable y .

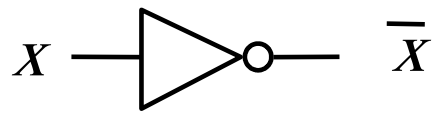
Building a NOT Gate with 4-to-1 Mux



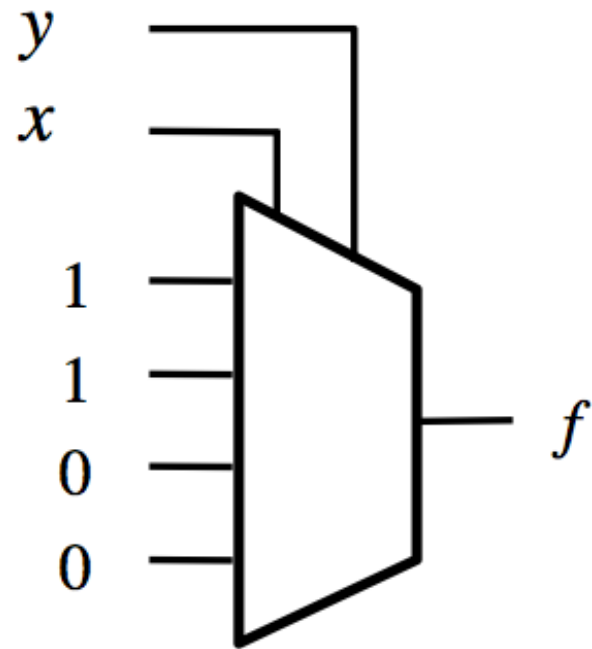
x	y	f
0	0	1
0	1	1
1	0	0
1	1	0



Building a NOT Gate with 4-to-1 Mux

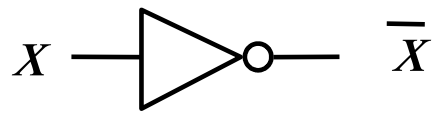


x	y	f
0	0	1
0	1	1
1	0	0
1	1	0

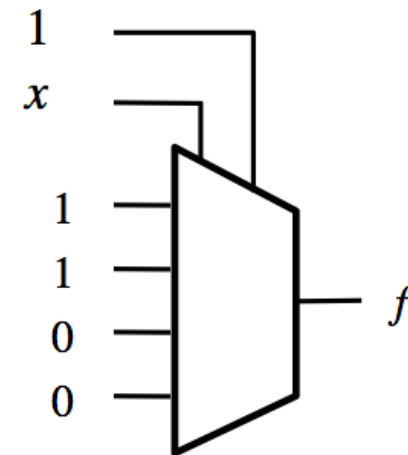
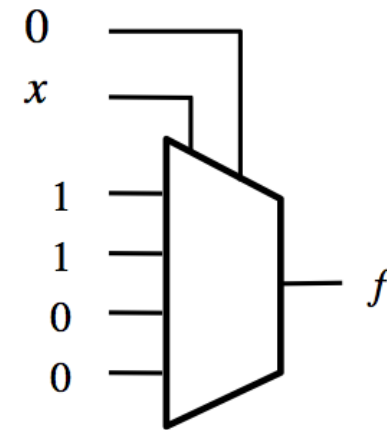


Now set y to either 0 or 1 (both will work). Why?

Building a NOT Gate with 4-to-1 Mux



x	\bar{x}
0	1
1	0

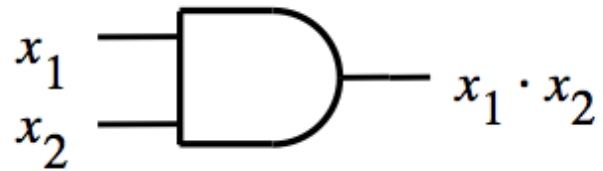


Two alternative solutions.

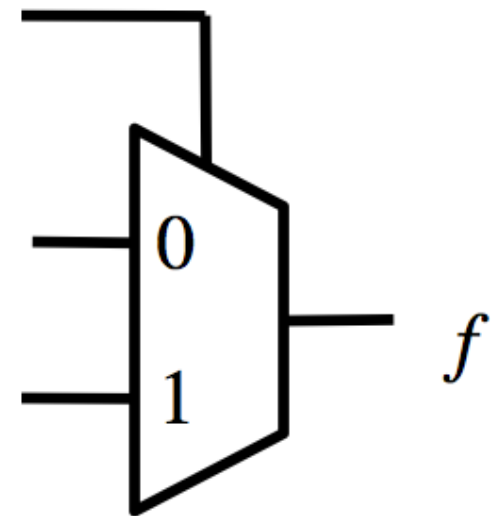
Implications

**Any Boolean function can be implemented
using only 4-to-1 multiplexers!**

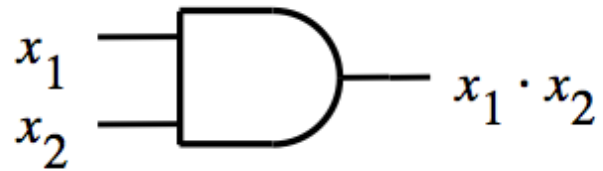
Building an AND Gate with 2-to-1 Mux



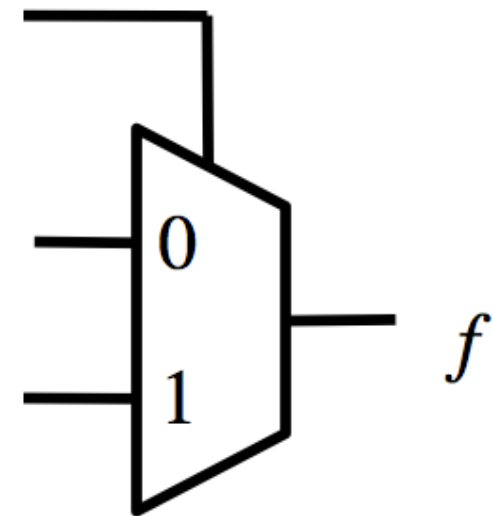
x_1	x_2	$x_1 \cdot x_2$
0	0	0
0	1	0
1	0	0
1	1	1



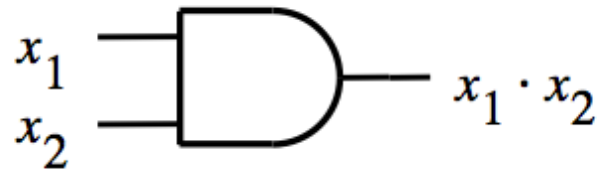
Building an AND Gate with 2-to-1 Mux



x_1	x_2	$x_1 \cdot x_2$
0	0	0
0	1	0
1	0	0
1	1	1

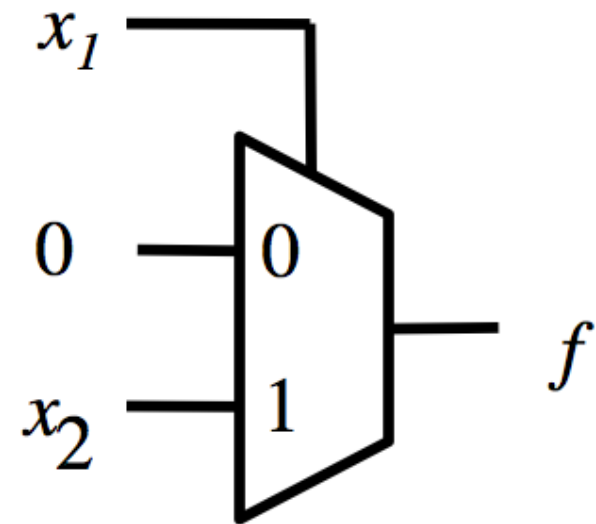


Building an AND Gate with 2-to-1 Mux

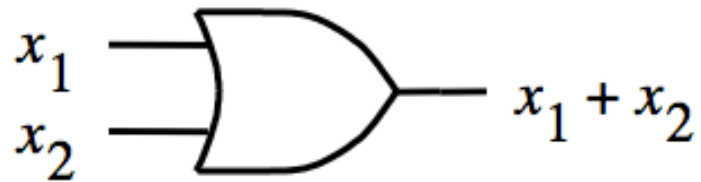


x_1	x_2	$x_1 \cdot x_2$
0	0	0
0	1	0
1	0	0
1	1	1

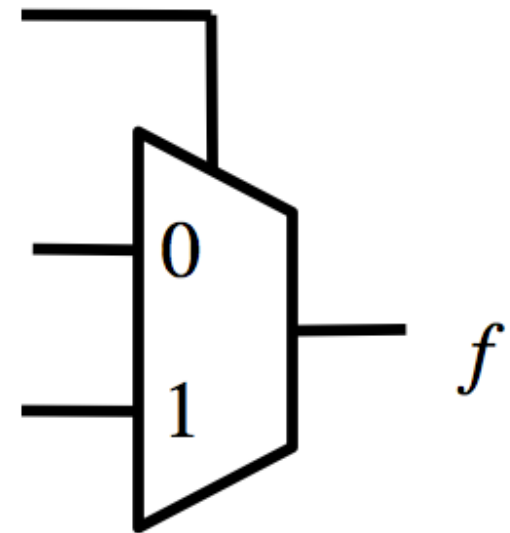
Red annotations: A vertical red line is drawn between the x_1 and x_2 columns. A horizontal red line is drawn between the second and third rows. Red curly braces group the output values: the first two rows (0, 0) are grouped and labeled 0 ; the last two rows (0, 1) are grouped and labeled x_2 .



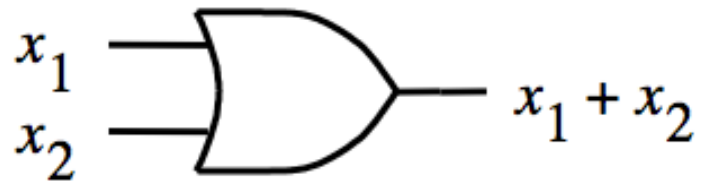
Building an OR Gate with 2-to-1 Mux



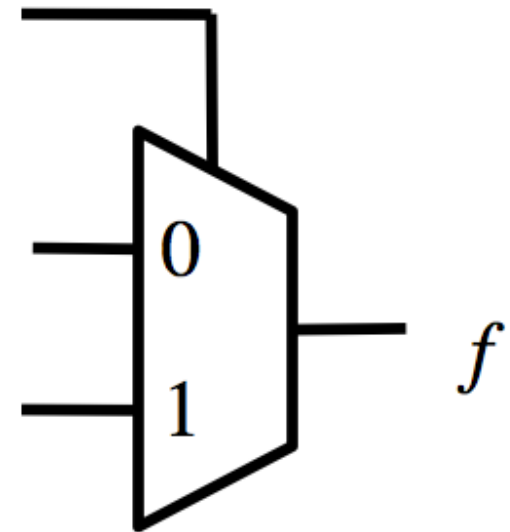
x_1	x_2	$x_1 + x_2$
0	0	0
0	1	1
1	0	1
1	1	1



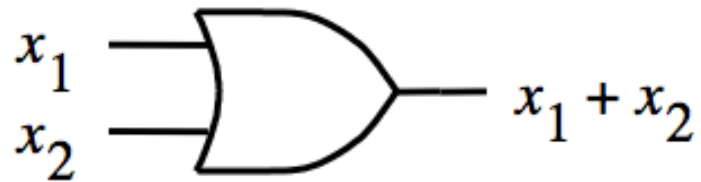
Building an OR Gate with 2-to-1 Mux



x_1	x_2	$x_1 + x_2$
0	0	0
0	1	1
1	0	1
1	1	1

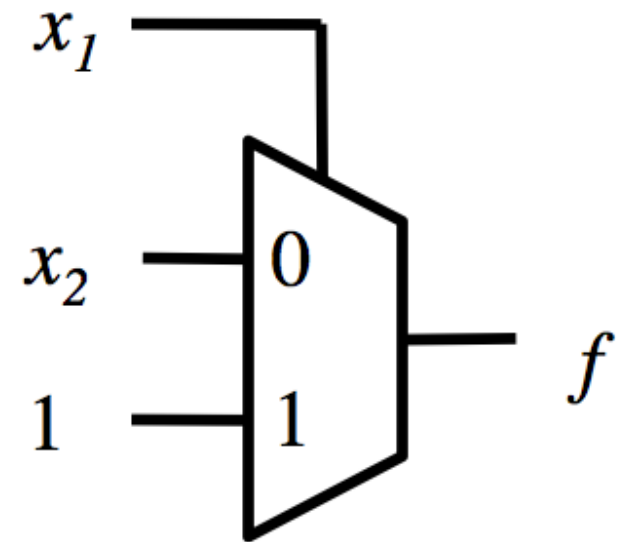


Building an OR Gate with 2-to-1 Mux

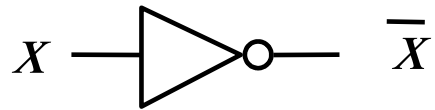


x_1	x_2	$x_1 + x_2$
0	0	0
0	1	1
1	0	1
1	1	1

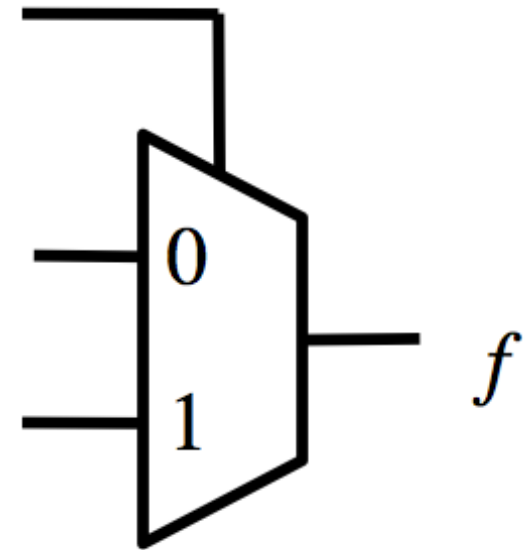
Red annotations: A vertical red line is between the x_1 and x_2 columns. A horizontal red line is between the second and third rows. Red curly braces on the right group the output values: the first two rows are grouped and labeled x_2 , and the last two rows are grouped and labeled 1 .



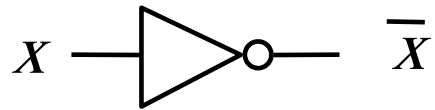
Building a NOT Gate with 2-to-1 Mux



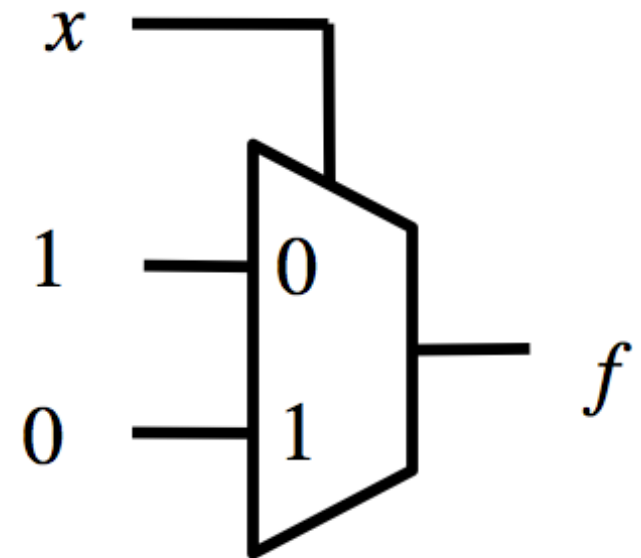
x	\bar{x}
0	1
1	0



Building a NOT Gate with 2-to-1 Mux



x	\bar{x}
0	1
1	0

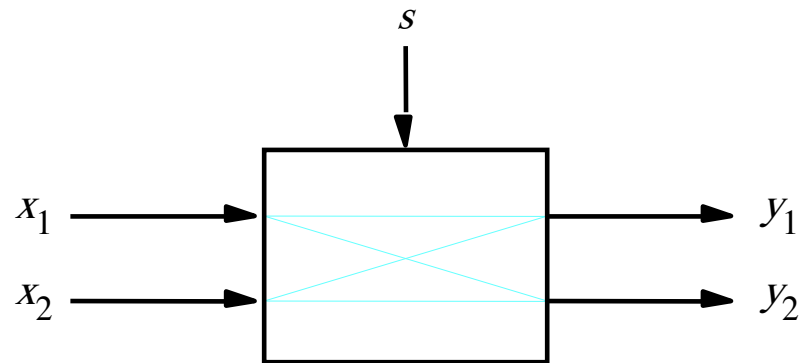


Implications

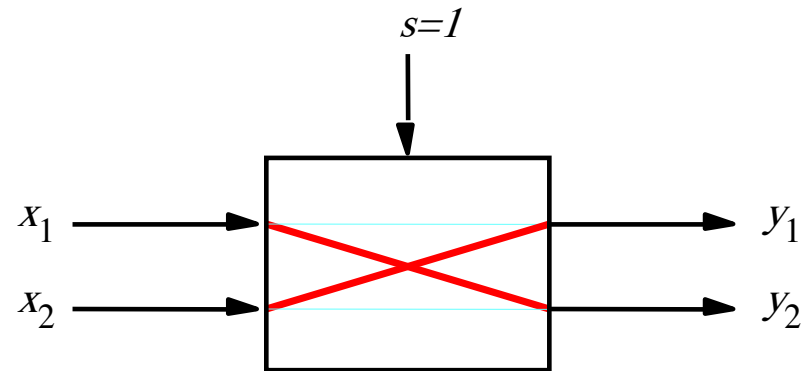
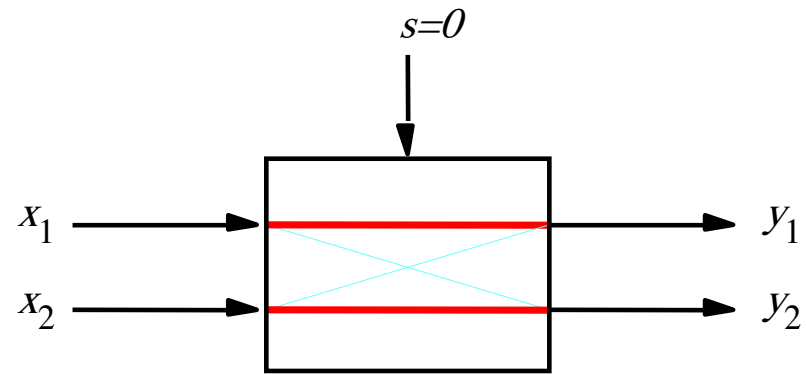
**Any Boolean function can be implemented
using only 2-to-1 multiplexers!**

Switch Circuit

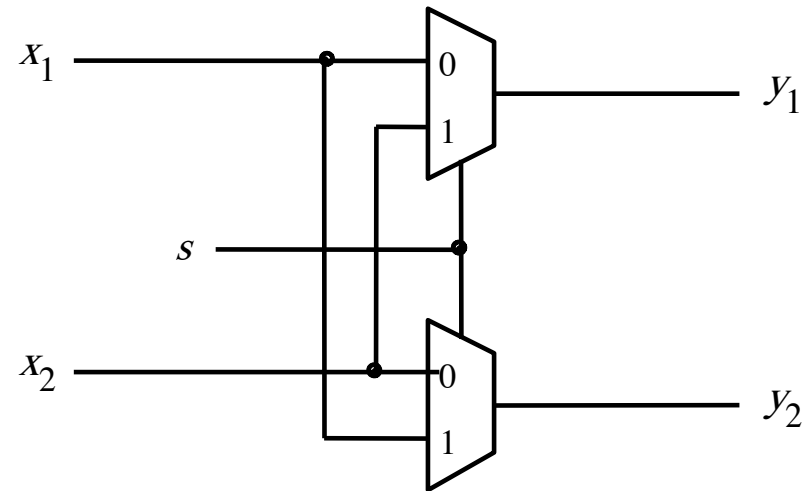
2 x 2 Crossbar switch



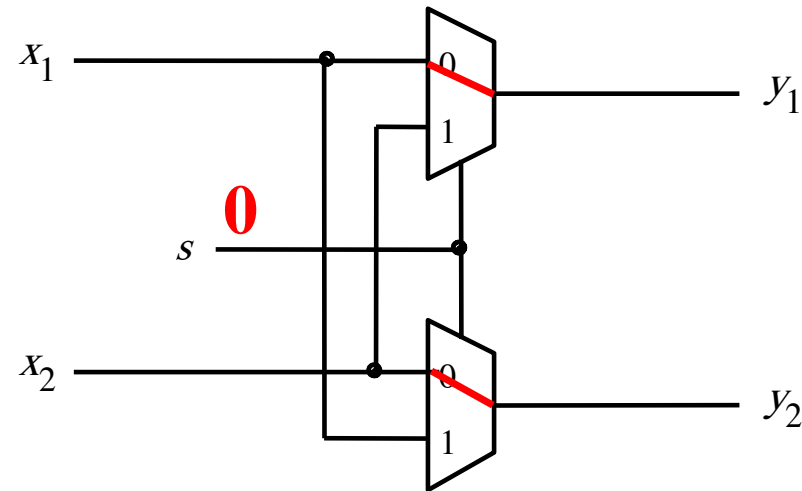
2 x 2 Crossbar switch



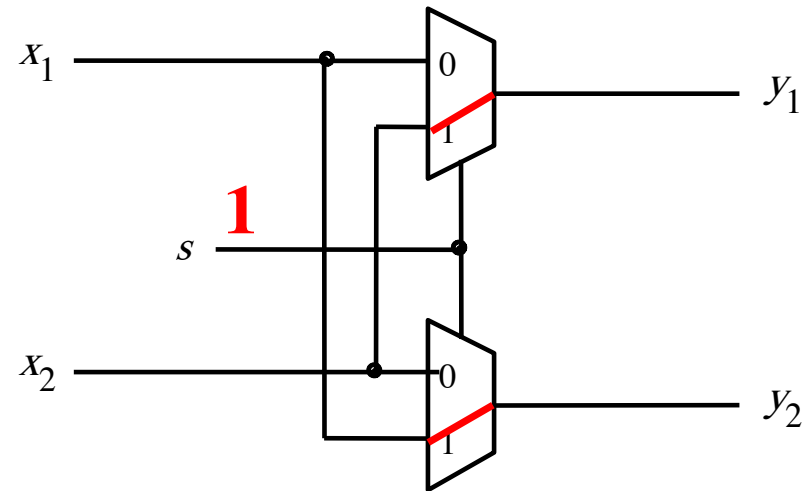
Implementation of a 2 x 2 crossbar switch with multiplexers



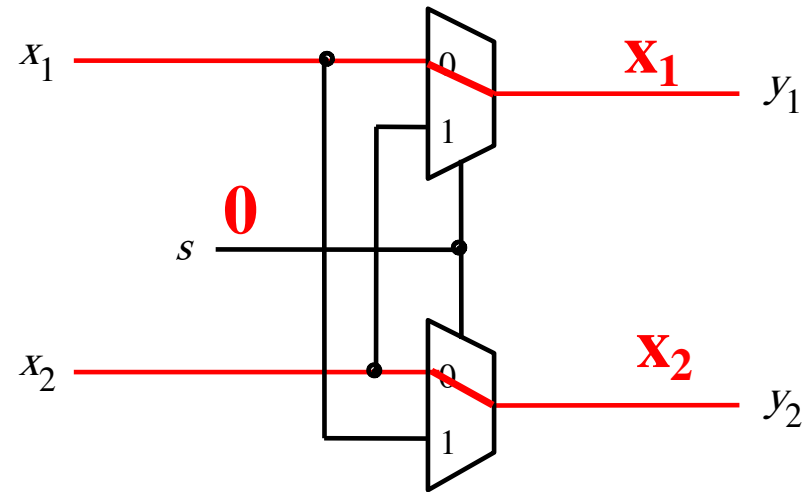
Implementation of a 2 x 2 crossbar switch with multiplexers



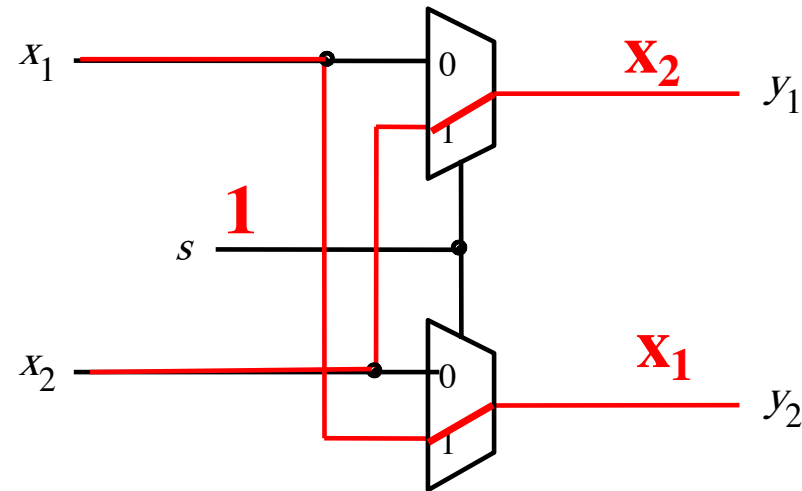
Implementation of a 2 x 2 crossbar switch with multiplexers



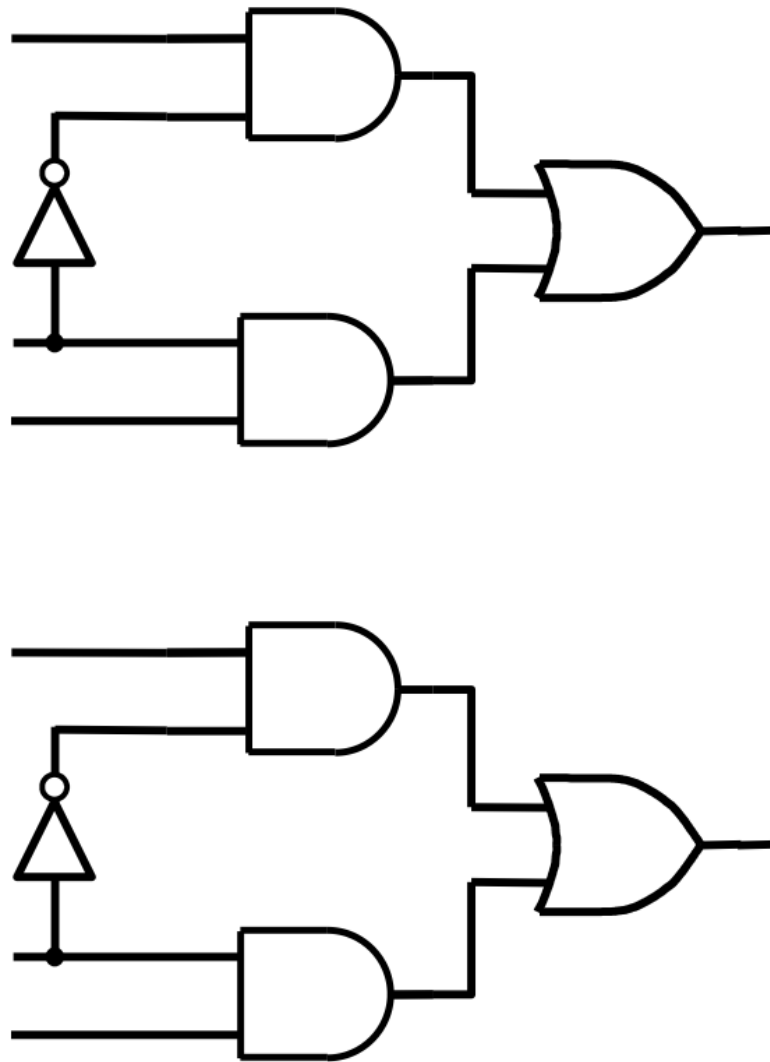
Implementation of a 2 x 2 crossbar switch with multiplexers



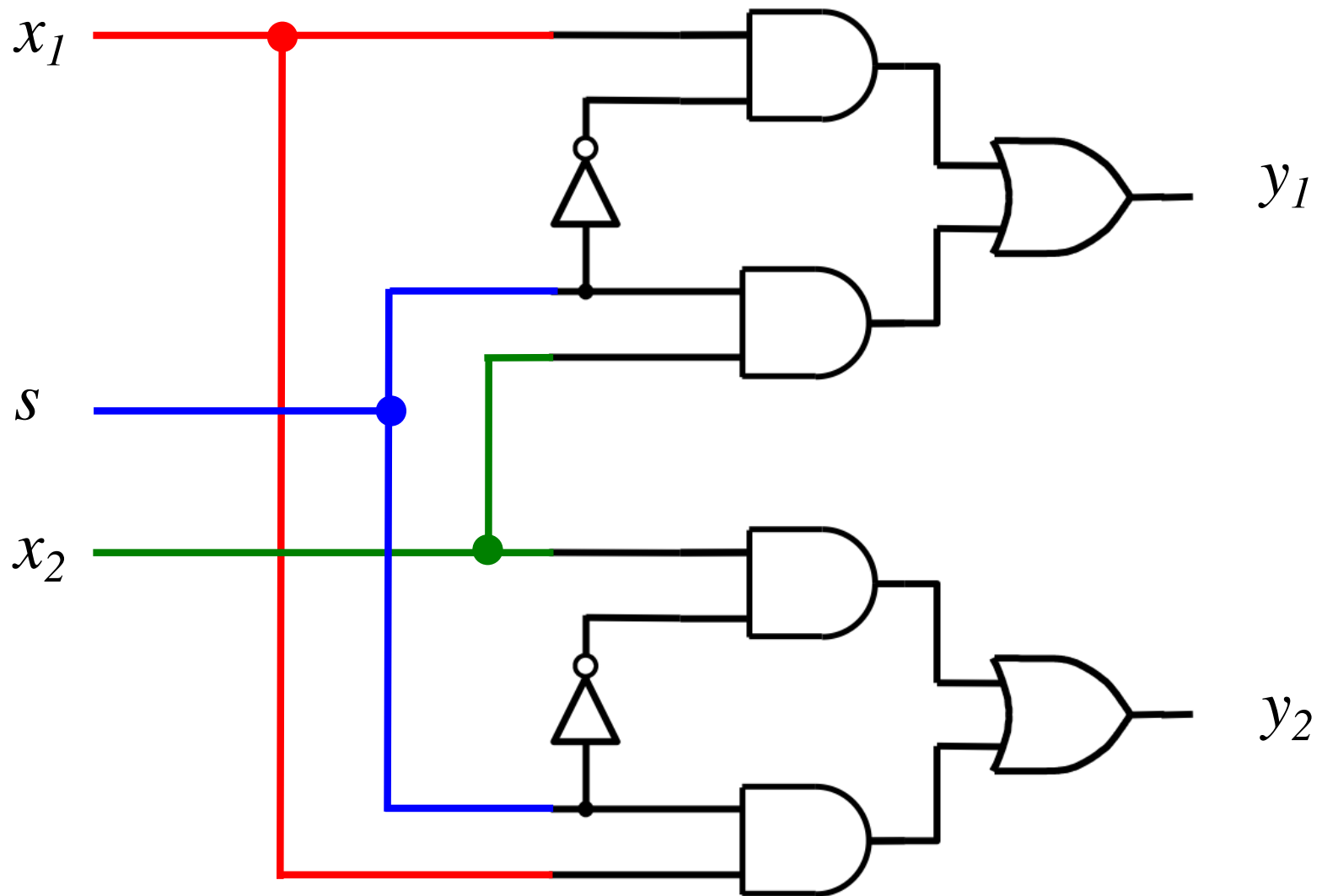
Implementation of a 2 x 2 crossbar switch with multiplexers



Implementation of a 2 x 2 crossbar switch with multiplexers



Implementation of a 2 x 2 crossbar switch with multiplexers



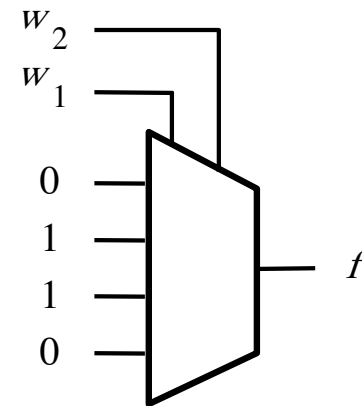
Synthesis of Logic Circuits Using Multiplexers

Synthesis of Logic Circuits Using Multiplexers

**Note: This method is NOT the same as simply replacing each logic gate with a multiplexer!
It is a lot more efficient.**

Implementation of a logic function with a 4-to-1 multiplexer

w_1	w_2	f
0	0	0
0	1	1
1	0	1
1	1	0

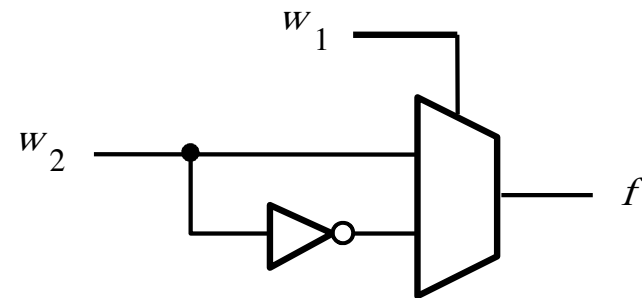


Implementation of the same logic function with a 2-to-1 multiplexer

w_1	w_2	f
0	0	0
0	1	1
1	0	1
1	1	0

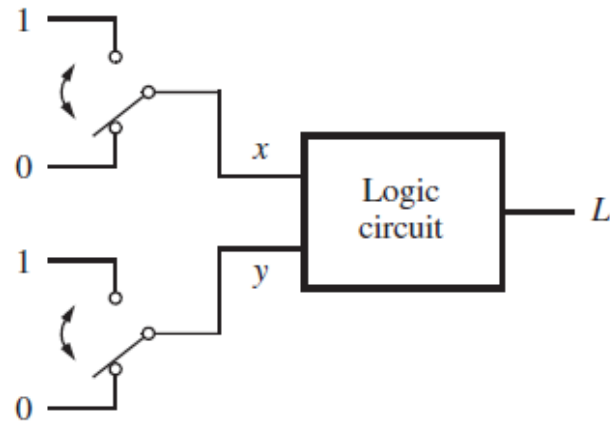
w_1	f
0	w_2
1	$\overline{w_2}$

(b) Modified truth table



(c) Circuit

The XOR Logic Gate

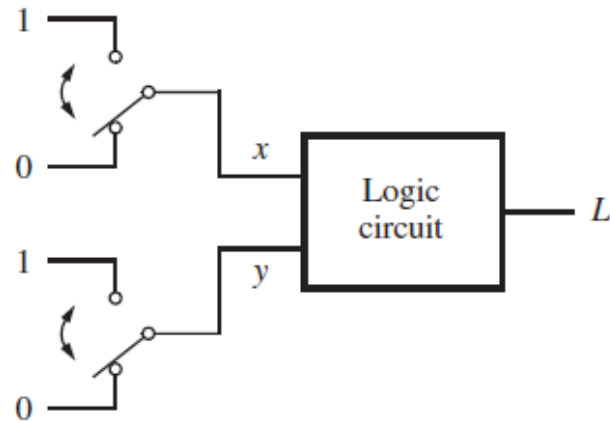


(a) Two switches that control a light

x	y	L
0	0	0
0	1	1
1	0	1
1	1	0

(b) Truth table

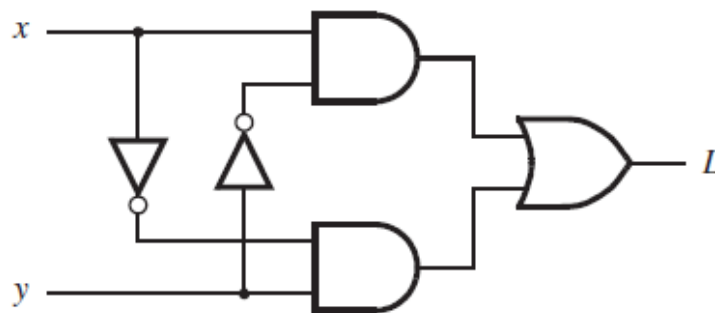
The XOR Logic Gate



(a) Two switches that control a light

x	y	L
0	0	0
0	1	1
1	0	1
1	1	0

(b) Truth table

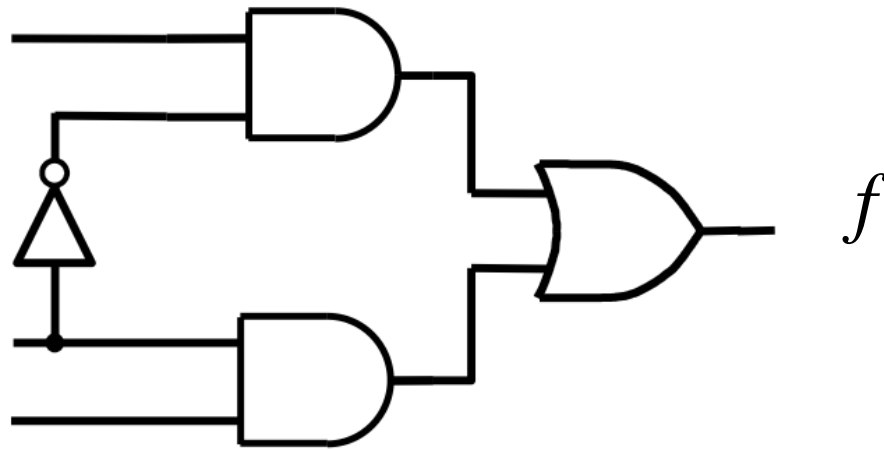


(c) Logic network

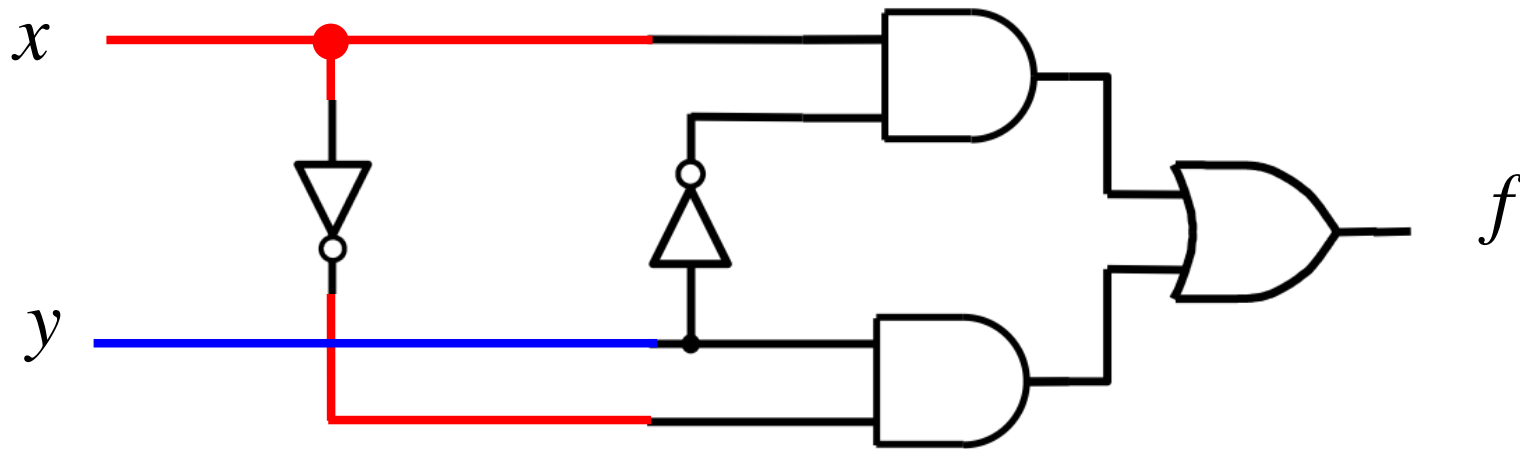


(d) XOR gate symbol

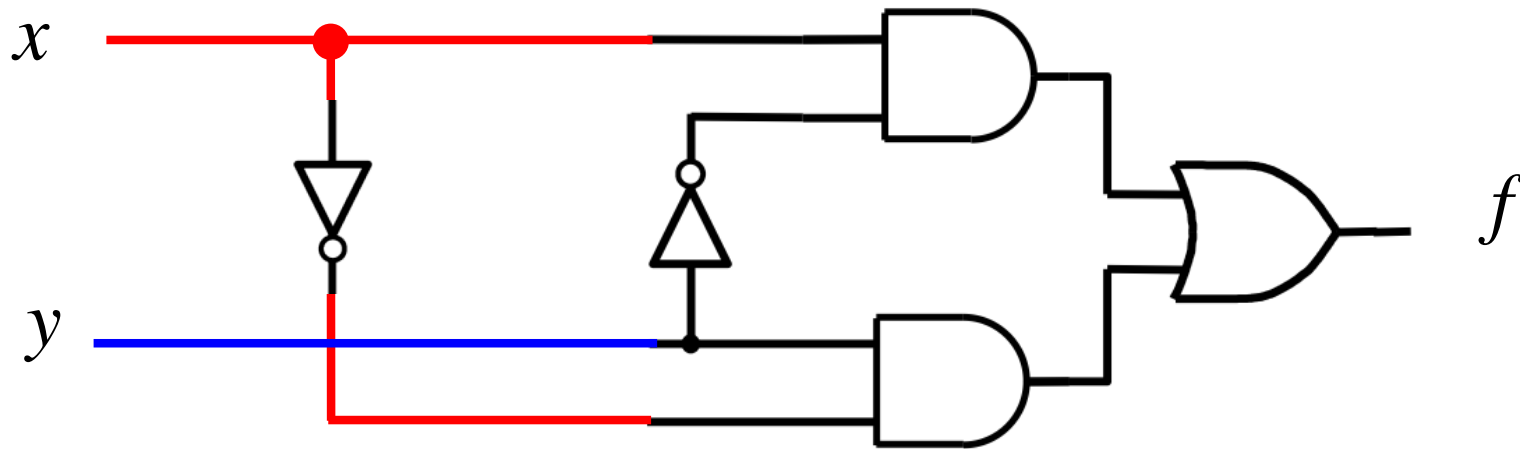
Implementation of the XOR Logic Gate with a 2-to-1 multiplexer and one NOT



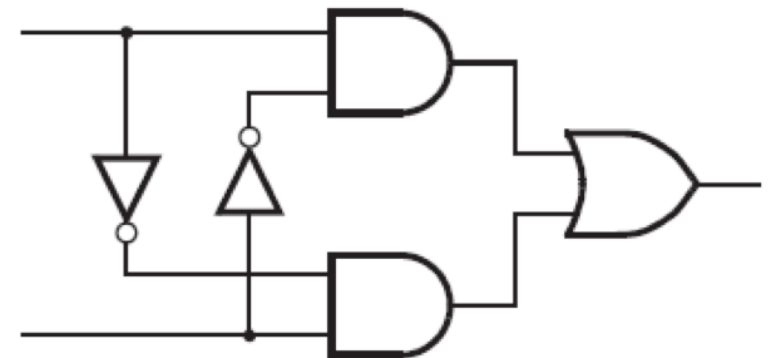
Implementation of the XOR Logic Gate with a 2-to-1 multiplexer and one NOT



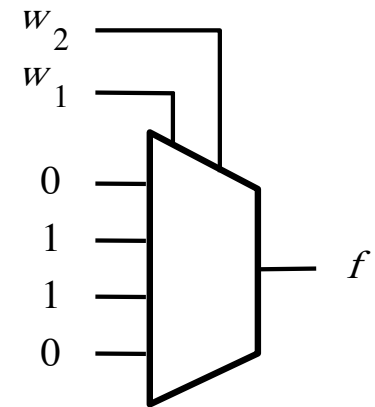
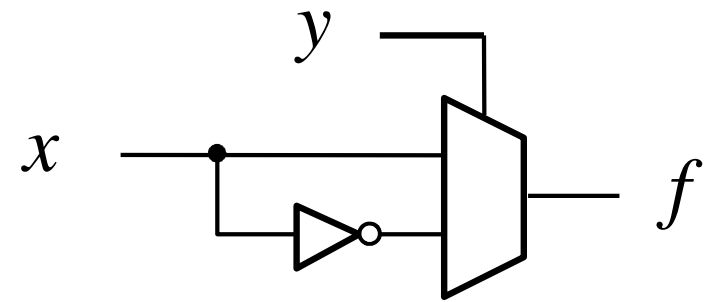
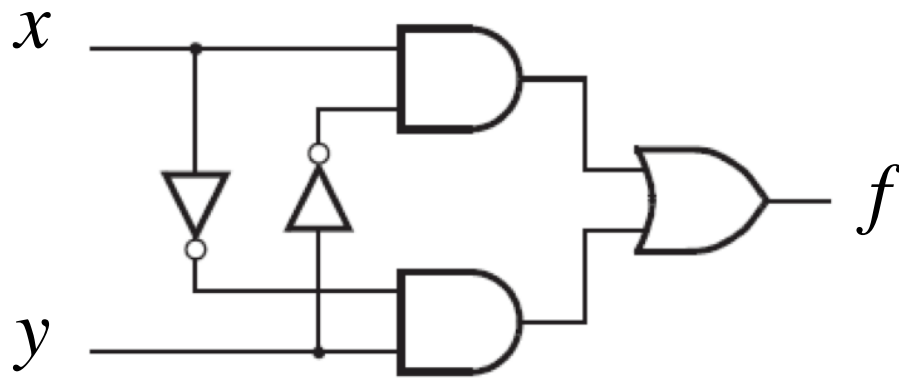
Implementation of the XOR Logic Gate with a 2-to-1 multiplexer and one NOT



These two circuits are equivalent
(the wires of the bottom AND gate are flipped)



**In other words,
all four of these are equivalent!**



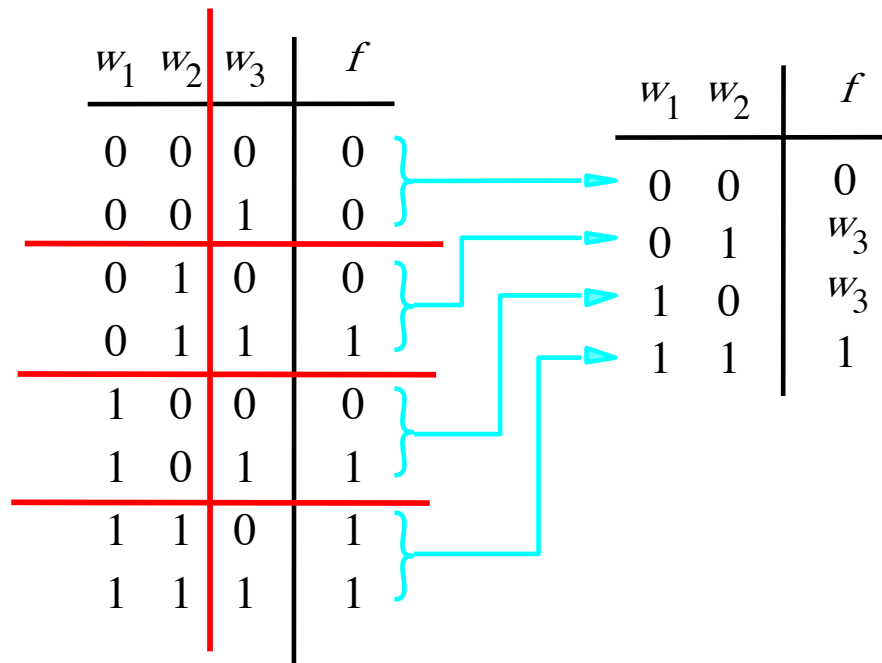
Implementation of another logic function

w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Implementation of another logic function

w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

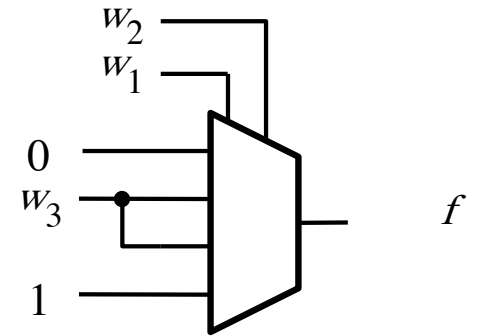
Implementation of another logic function



Implementation of another logic function

w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

w_1	w_2	f
0	0	0
0	1	w_3
1	0	w_3
1	1	1



Another Example (3-input XOR)

Implementation of 3-input XOR with 2-to-1 Multiplexers

w_1	w_2	w_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

Implementation of 3-input XOR with 2-to-1 Multiplexers

w_1	w_2	w_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

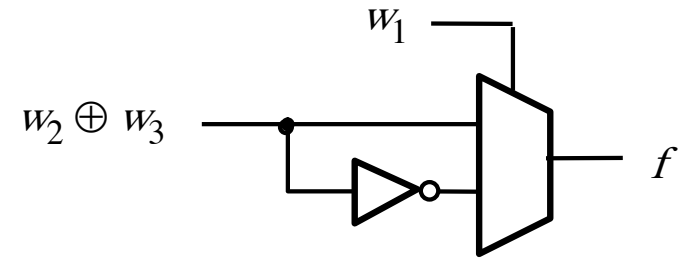
Red annotations in the table:

- A vertical red line is drawn between the w_1 and w_2 columns.
- A horizontal red line is drawn between the w_3 and f columns.
- Red curly braces group the f values for $w_1 = 0$ and $w_1 = 1$.
- Red text $w_2 \oplus w_3$ is placed to the right of the first brace.
- Red text $\overline{w_2 \oplus w_3}$ is placed to the right of the second brace.

Implementation of 3-input XOR with 2-to-1 Multiplexers

w_1	w_2	w_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

(a) Truth table



(b) Circuit

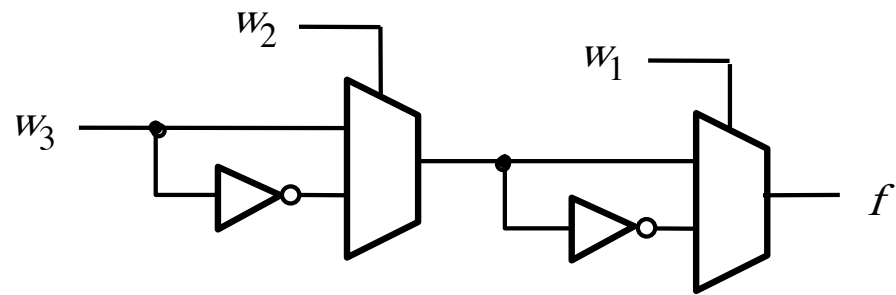
Implementation of 3-input XOR with 2-to-1 Multiplexers

w_1	w_2	w_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

Blue annotations in the table:

- A blue bracket groups the rows where $w_1=0$ and $w_2=0$, labeled w_3 .
- A blue bracket groups the rows where $w_1=0$ and $w_2=1$, labeled $\overline{w_3}$.

(a) Truth table



(b) Circuit

Implementation of 3-input XOR with a 4-to-1 Multiplexer

w_1	w_2	w_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

Implementation of 3-input XOR with a 4-to-1 Multiplexer

w_1	w_2	w_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

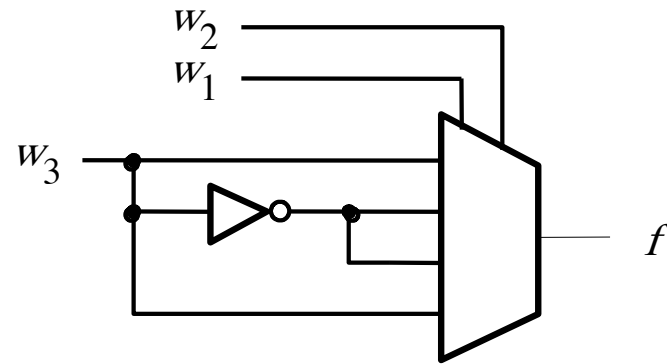
Implementation of 3-input XOR with a 4-to-1 Multiplexer

w_1	w_2	w_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

Implementation of 3-input XOR with a 4-to-1 Multiplexer

w_1	w_2	w_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

(a) Truth table



(b) Circuit

Multiplexor Synthesis Using Shannon's Expansion

Three-input majority function

w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Three-input majority function

w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

w_1	f
0	0
1	1

Three-input majority function

w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

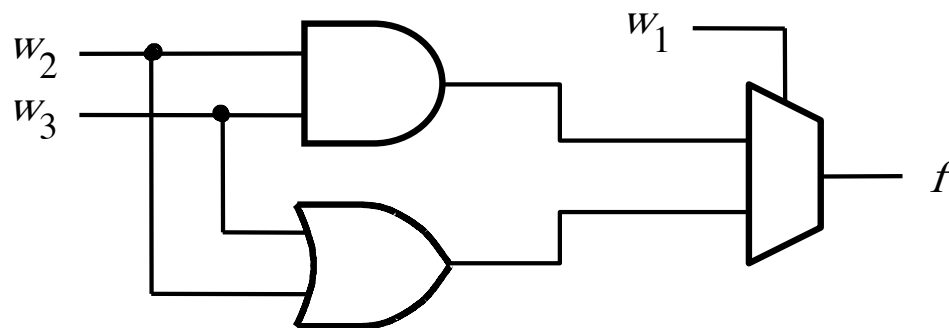
w_1	f
0	$w_2 w_3$
1	$w_2 + w_3$

Three-input majority function

w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

w_1	f
0	$w_2 w_3$
1	$w_2 + w_3$

(b) Truth table



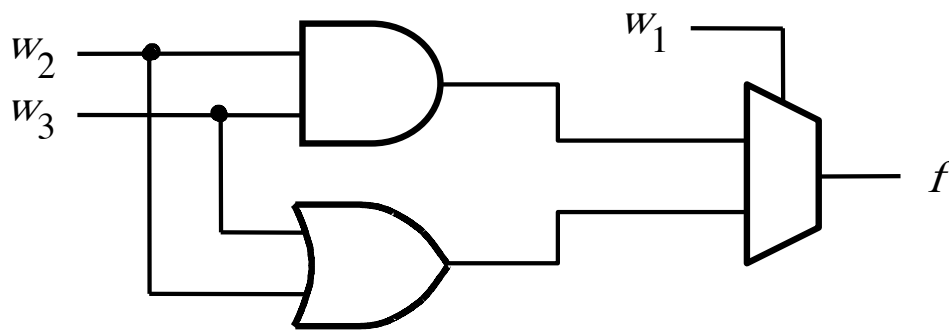
(b) Circuit

[Figure 4.10a from the textbook]

Three-input majority function

$$f = \bar{w}_1 w_2 w_3 + w_1 \bar{w}_2 w_3 + w_1 w_2 \bar{w}_3 + w_1 w_2 w_3$$

$$\begin{aligned} f &= \bar{w}_1 (w_2 w_3) + w_1 (\bar{w}_2 w_3 + w_2 \bar{w}_3 + w_2 w_3) \\ &= \bar{w}_1 (w_2 w_3) + w_1 (w_2 + w_3) \end{aligned}$$



Shannon's Expansion Theorem

Any Boolean function $f(w_1, \dots, w_n)$ can be rewritten in the form:

$$f(w_1, w_2, \dots, w_n) = \bar{w}_1 \cdot f(0, w_2, \dots, w_n) + w_1 \cdot f(1, w_2, \dots, w_n)$$

Shannon's Expansion Theorem

Any Boolean function $f(w_1, \dots, w_n)$ can be rewritten in the form:

$$f(w_1, w_2, \dots, w_n) = \bar{w}_1 \cdot f(0, w_2, \dots, w_n) + w_1 \cdot f(1, w_2, \dots, w_n)$$

$$f = \bar{w}_1 f_{\bar{w}_1} + w_1 f_{w_1}$$

Shannon's Expansion Theorem

Any Boolean function $f(w_1, \dots, w_n)$ can be rewritten in the form:

$$f(w_1, w_2, \dots, w_n) = \bar{w}_1 \cdot f(0, w_2, \dots, w_n) + w_1 \cdot f(1, w_2, \dots, w_n)$$

$$f = \bar{w}_1 f_{\bar{w}_1} + w_1 f_{w_1}$$

cofactor

cofactor

Shannon's Expansion Theorem (Example)

$$f(w_1, w_2, w_3) = w_1w_2 + w_1w_3 + w_2w_3$$

Shannon's Expansion Theorem (Example)

$$f(w_1, w_2, w_3) = w_1w_2 + w_1w_3 + w_2w_3$$

$$f(w_1, w_2, w_3) = w_1w_2 + w_1w_3 + w_2w_3 (\overline{w_1} + w_1)$$

Shannon's Expansion Theorem (Example)

$$f(w_1, w_2, w_3) = w_1w_2 + w_1w_3 + w_2w_3$$

$$f(w_1, w_2, w_3) = w_1w_2 + w_1w_3 + w_2w_3 (\bar{w}_1 + w_1)$$

$$\begin{aligned} f &= \bar{w}_1(0 \cdot w_2 + 0 \cdot w_3 + w_2w_3) + w_1(1 \cdot w_2 + 1 \cdot w_3 + w_2w_3) \\ &= \bar{w}_1(w_2w_3) + w_1(w_2 + w_3) \end{aligned}$$

Shannon's Expansion Theorem (In terms of more than one variable)

$$f(w_1, \dots, w_n) = \bar{w}_1 \bar{w}_2 \cdot f(0, 0, w_3, \dots, w_n) + \bar{w}_1 w_2 \cdot f(0, 1, w_3, \dots, w_n) \\ + w_1 \bar{w}_2 \cdot f(1, 0, w_3, \dots, w_n) + w_1 w_2 \cdot f(1, 1, w_3, \dots, w_n)$$

This form is suitable for implementation with a 4x1 multiplexer.

Another Example

Factor and implement the following function with a 2-to-1 multiplexer

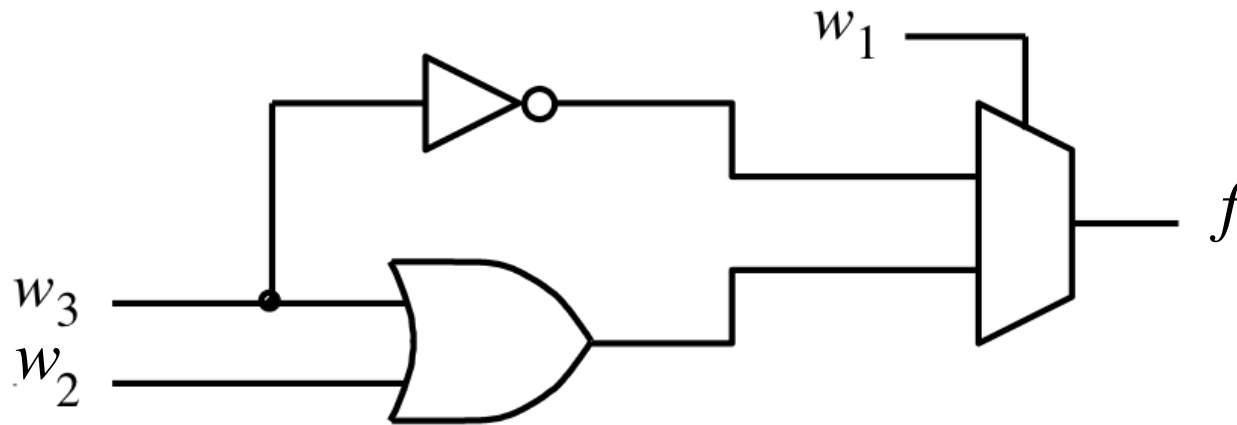
$$f = \bar{w}_1\bar{w}_3 + w_1w_2 + w_1w_3$$

Factor and implement the following function with a 2-to-1 multiplexer

$$f = \bar{w}_1 \bar{w}_3 + w_1 w_2 + w_1 w_3$$

$$\begin{aligned} f &= \bar{w}_1 f_{\bar{w}_1} + w_1 f_{w_1} \\ &= \bar{w}_1 (\bar{w}_3) + w_1 (w_2 + w_3) \end{aligned}$$

Factor and implement the following function with a 2-to-1 multiplexer



$$\begin{aligned} f &= \overline{w_1}f_{\overline{w_1}} + w_1f_{w_1} \\ &= \overline{w_1}(\overline{w_3}) + w_1(w_2 + w_3) \end{aligned}$$

Factor and implement the following function with a 4-to-1 multiplexer

$$f = \bar{w}_1\bar{w}_3 + w_1w_2 + w_1w_3$$

Factor and implement the following function with a 4-to-1 multiplexer

$$f = \bar{w}_1 \bar{w}_3 + w_1 w_2 + w_1 w_3$$
$$= \bar{w}_1 (\bar{w}_2 + w_2) \bar{w}_3 + w_1 w_2 + w_1 (\bar{w}_2 + w_2) w_3$$

Factor and implement the following function with a 4-to-1 multiplexer

$$f = \bar{w}_1 \bar{w}_3 + w_1 w_2 + w_1 w_3$$

$$= \bar{w}_1 (\bar{w}_2 + w_2) \bar{w}_3 + w_1 w_2 + w_1 (\bar{w}_2 + w_2) w_3$$

$$= \bar{w}_1 \bar{w}_2 \bar{w}_3 + \bar{w}_1 w_2 \bar{w}_3 + w_1 w_2 + w_1 \bar{w}_2 w_3 + w_1 w_2 w_3$$

$$= \bar{w}_1 \bar{w}_2 \bar{w}_3 + \bar{w}_1 w_2 \bar{w}_3 + w_1 \bar{w}_2 w_3 + w_1 w_2 (1 + w_3)$$

$$= \bar{w}_1 \bar{w}_2 (\bar{w}_3) + \bar{w}_1 w_2 (\bar{w}_3) + w_1 \bar{w}_2 (w_3) + w_1 w_2 (1)$$

Factor and implement the following function with a 4-to-1 multiplexer

$$f = \bar{w}_1 \bar{w}_3 + w_1 w_2 + w_1 w_3$$

$$= \bar{w}_1 (\bar{w}_2 + w_2) \bar{w}_3 + w_1 w_2 + w_1 (\bar{w}_2 + w_2) w_3$$

$$= \bar{w}_1 \bar{w}_2 \bar{w}_3 + \bar{w}_1 w_2 \bar{w}_3 + w_1 w_2 + w_1 \bar{w}_2 w_3 + w_1 w_2 w_3$$

$$= \bar{w}_1 \bar{w}_2 \bar{w}_3 + \bar{w}_1 w_2 \bar{w}_3 + w_1 \bar{w}_2 w_3 + w_1 w_2 (1 + w_3)$$

$$= \bar{w}_1 \bar{w}_2 (\bar{w}_3) + \bar{w}_1 w_2 (\bar{w}_3) + w_1 \bar{w}_2 (w_3) + w_1 w_2 (1)$$

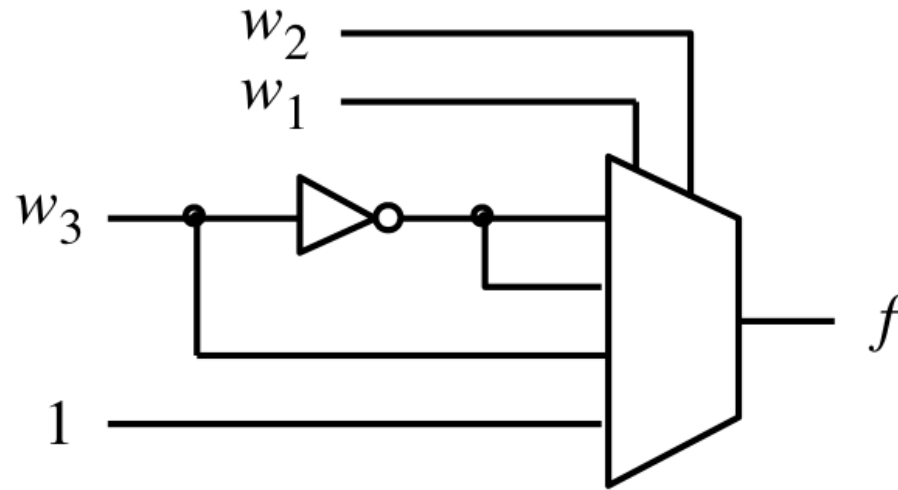
these are the 4 cofactors

Factor and implement the following function with a 4-to-1 multiplexer

$$f = \bar{w}_1\bar{w}_3 + w_1w_2 + w_1w_3$$

$$\begin{aligned} f &= \bar{w}_1\bar{w}_2f_{\bar{w}_1\bar{w}_2} + \bar{w}_1w_2f_{\bar{w}_1w_2} + w_1\bar{w}_2f_{w_1\bar{w}_2} + w_1w_2f_{w_1w_2} \\ &= \bar{w}_1\bar{w}_2(\bar{w}_3) + \bar{w}_1w_2(\bar{w}_3) + w_1\bar{w}_2(w_3) + w_1w_2(1) \end{aligned}$$

Factor and implement the following function with a 4-to-1 multiplexer



$$\begin{aligned} f &= \overline{w_1}\overline{w_2}f_{\overline{w_1}\overline{w_2}} + \overline{w_1}w_2f_{\overline{w_1}w_2} + w_1\overline{w_2}f_{w_1\overline{w_2}} + w_1w_2f_{w_1w_2} \\ &= \overline{w_1}\overline{w_2}(\overline{w_3}) + \overline{w_1}w_2(\overline{w_3}) + w_1\overline{w_2}(w_3) + w_1w_2(1) \end{aligned}$$

Yet Another Example

Factor and implement the following function using only 2-to-1 multiplexers

$$f = w_1w_2 + w_1w_3 + w_2w_3$$

Factor and implement the following function using only 2-to-1 multiplexers

$$f = w_1w_2 + w_1w_3 + w_2w_3$$

$$\begin{aligned} f &= \bar{w}_1(w_2w_3) + w_1(w_2 + w_3 + w_2w_3) \\ &= \bar{w}_1(w_2w_3) + w_1(w_2 + w_3) \end{aligned}$$

Factor and implement the following function using only 2-to-1 multiplexers

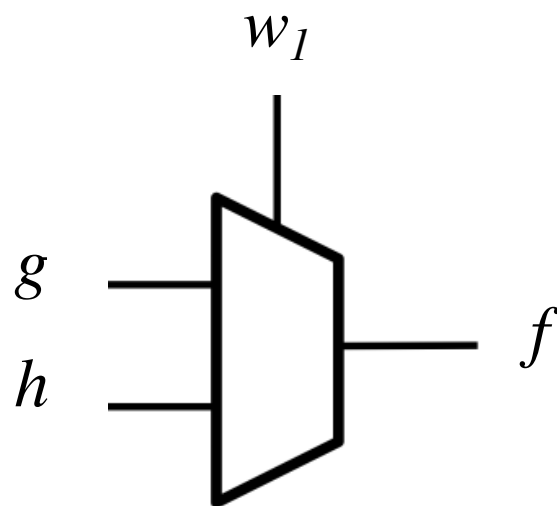
$$f = w_1w_2 + w_1w_3 + w_2w_3$$

$$f = \bar{w}_1(w_2w_3) + w_1(w_2 + w_3 + w_2w_3)$$

$$= \bar{w}_1(\underbrace{w_2w_3}) + w_1(\underbrace{w_2 + w_3})$$

$$g = w_2w_3 \quad h = w_2 + w_3$$

Factor and implement the following function using only 2-to-1 multiplexers



$$f = \bar{w}_1 (w_2 w_3) + w_1 (w_2 + w_3 + w_2 w_3)$$

$$= \bar{w}_1 \underbrace{(w_2 w_3)}_g + w_1 \underbrace{(w_2 + w_3)}_h$$

$$g = w_2 w_3 \quad h = w_2 + w_3$$

Factor and implement the following function using only 2-to-1 multiplexers

$$g = w_2 w_3$$

$$h = w_2 + w_3$$

Factor and implement the following function using only 2-to-1 multiplexers

$$g = w_2 w_3$$



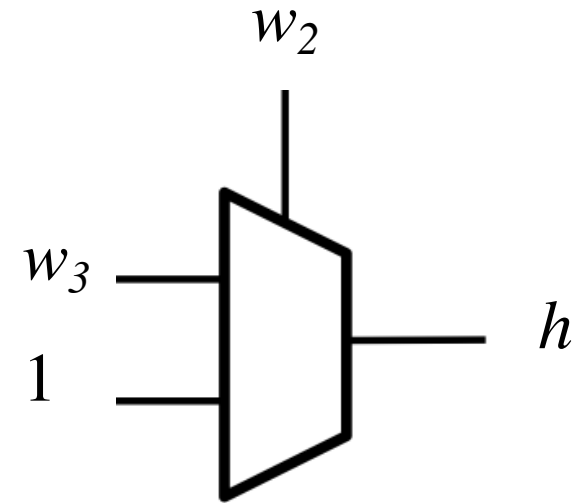
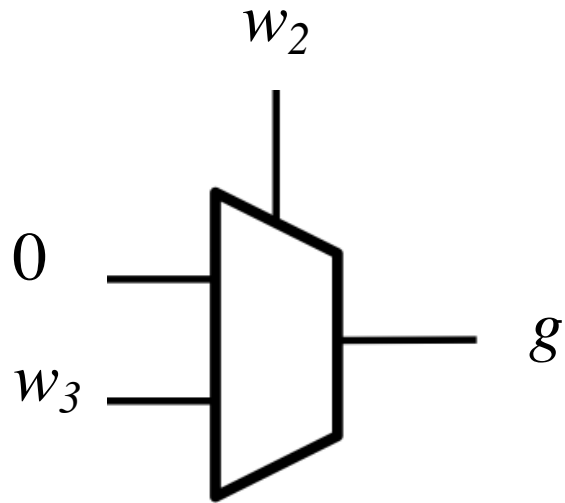
$$g = \bar{w}_2(0) + w_2(w_3)$$

$$h = w_2 + w_3$$



$$h = \bar{w}_2(w_3) + w_2(1)$$

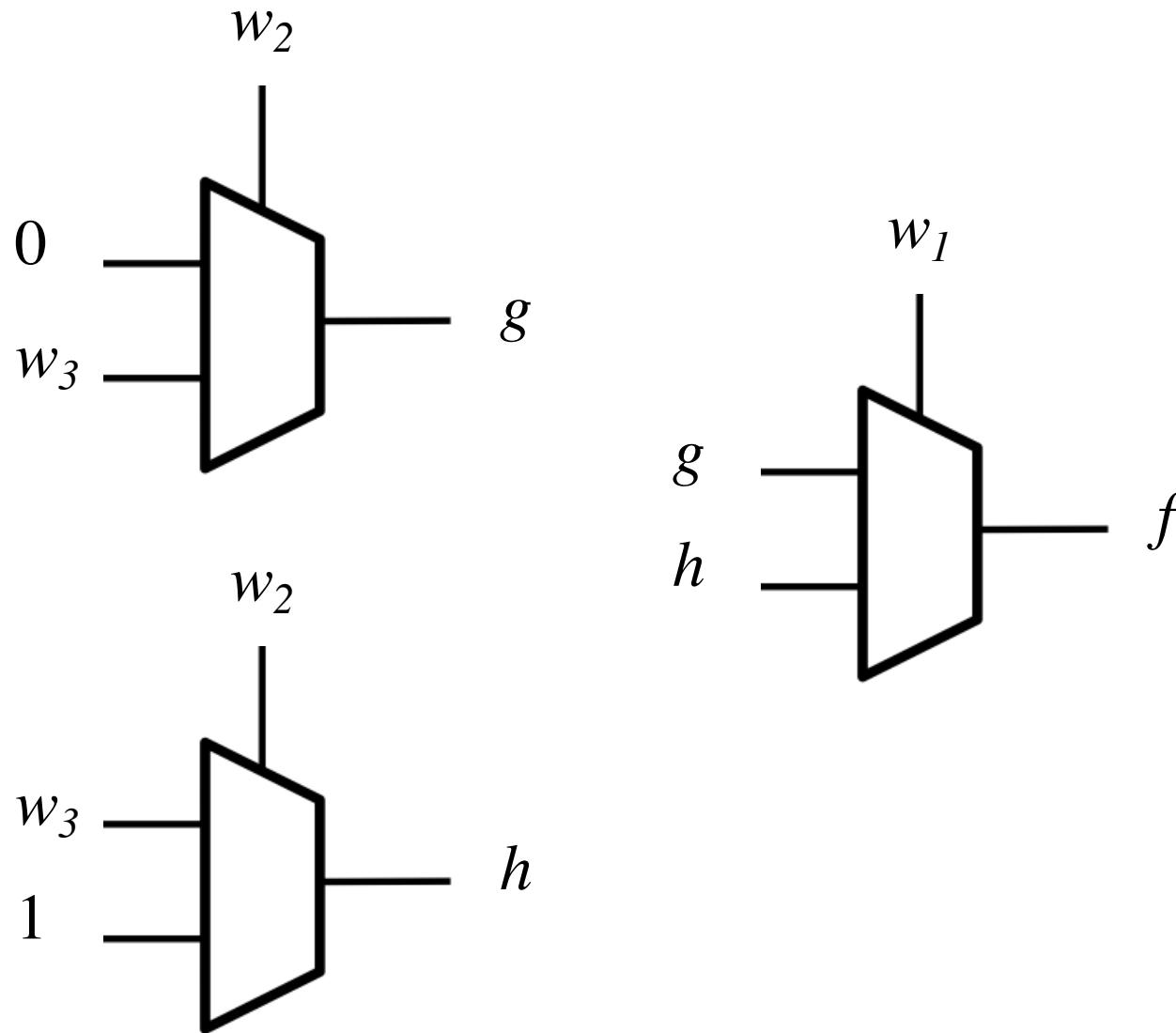
Factor and implement the following function using only 2-to-1 multiplexers



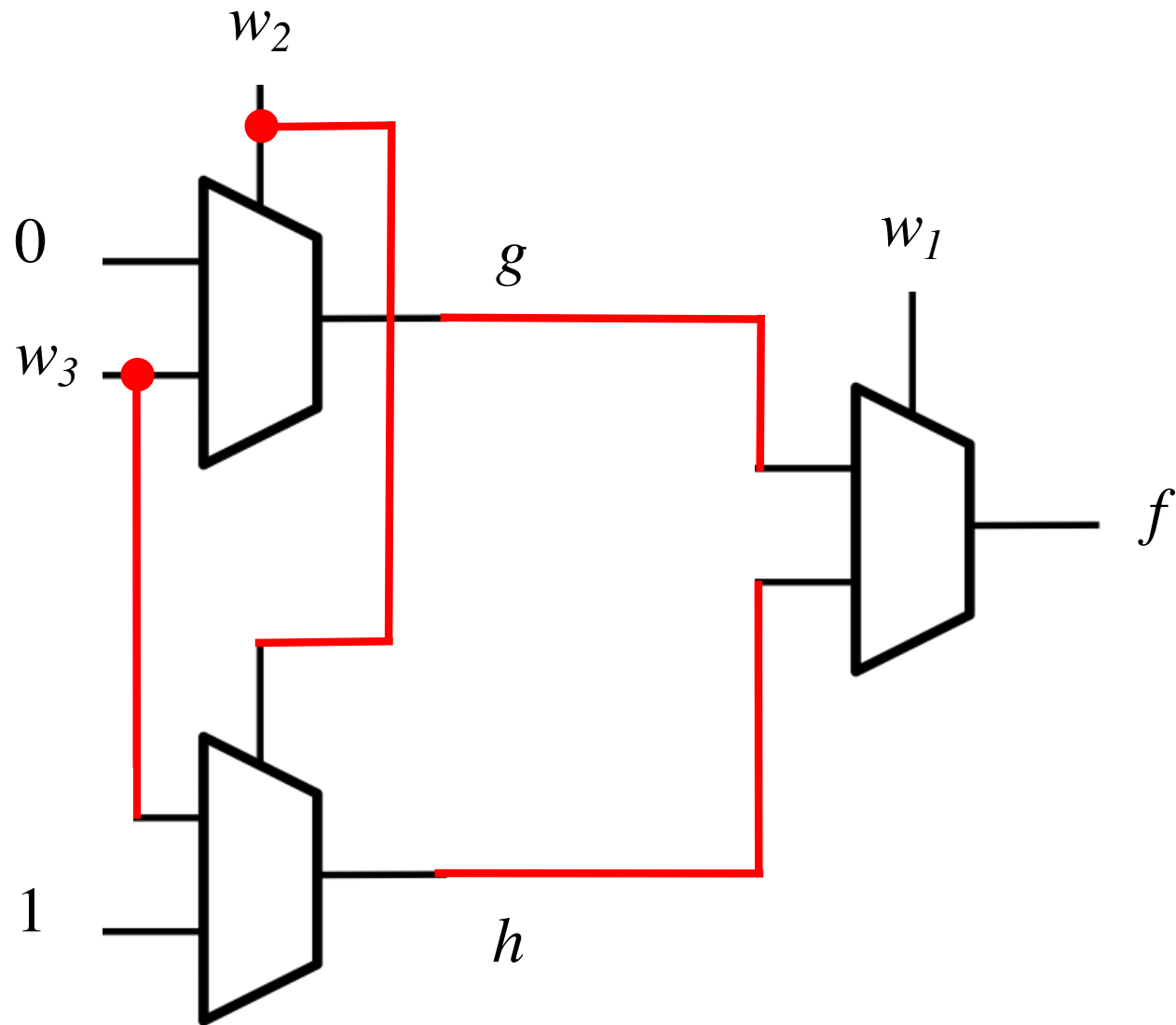
$$g = \bar{w}_2(0) + w_2(w_3)$$

$$h = \bar{w}_2(w_3) + w_2(1)$$

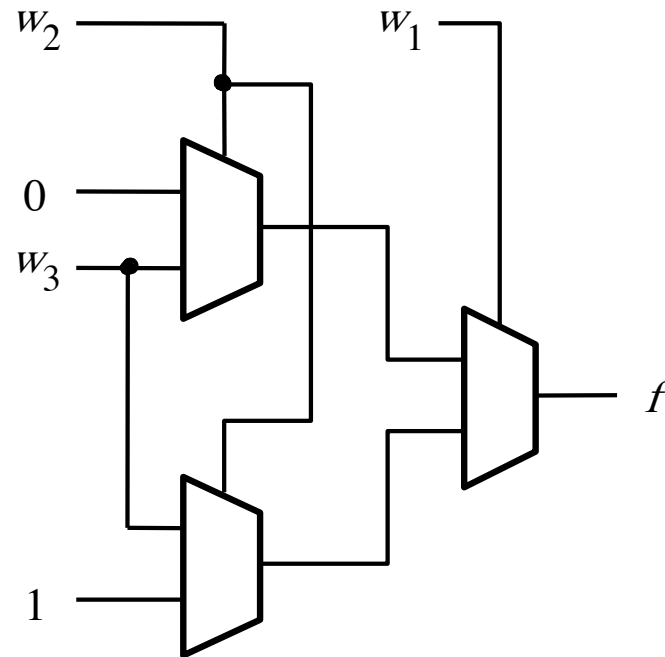
Finally, we are ready to draw the circuit



Finally, we are ready to draw the circuit



Finally, we are ready to draw the circuit



Questions?

THE END