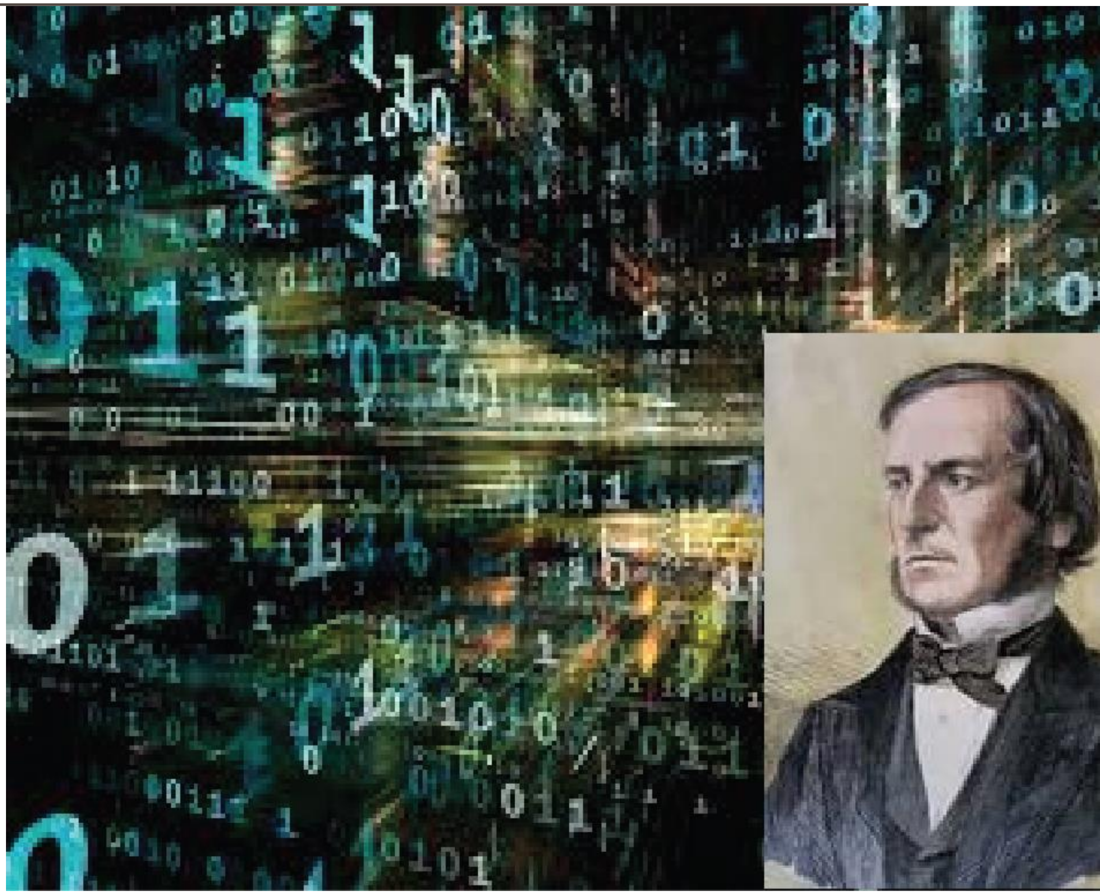
DIGITAL CIRCUITS

Week-7, Lecture-4
Multiplexers



Sneh Saurabh
14th September, 2018

Digital Circuits: Announcements/Revision

Combinational Circuit Design

Using Multiplexers

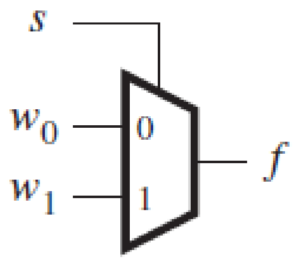
Multiplexer: Basics

- A multiplexer circuit has a number of data inputs, one or more select inputs, and one output.
- It passes the signal value on one of the data inputs to the output.
- The data input is selected by the values of the select inputs

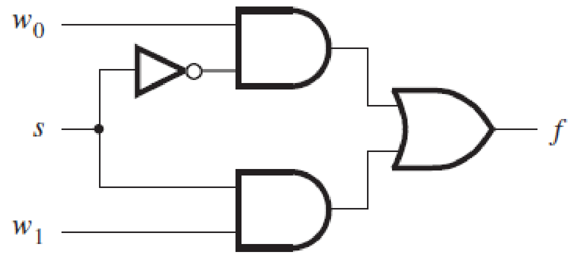
2-to-1 multiplexer:

- Two data inputs, 1 select input and one output
- Output same as the input w_0 if $s = 0$
- Output same as the input w_1 if $s = 1$

$$f(s, w_0, w_1) = s'w_0 + sw_1$$



s	f
0	w_0
1	w_1

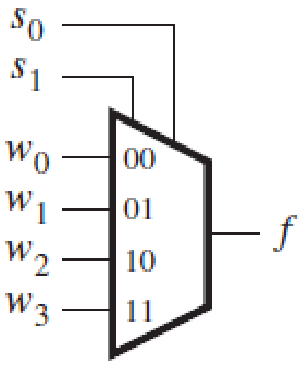


4-to-1 Multiplexer

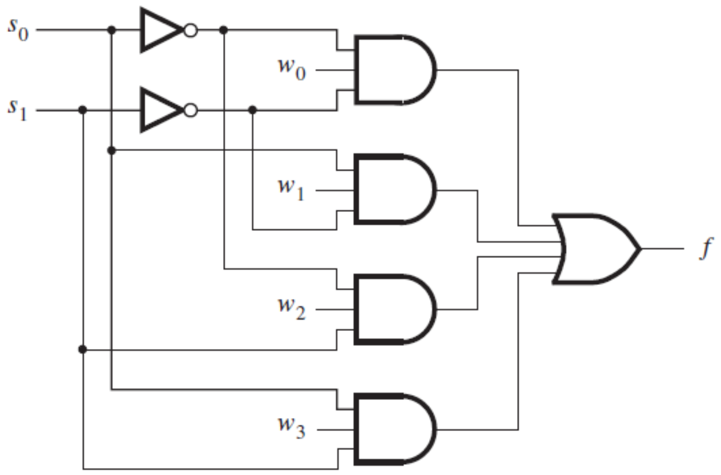
4-to-1 multiplexer:

- Four data inputs, 2 select inputs and one output

$$f = s_0{}'s_1{}'w_0 + s_0s_1{}'w_1 + s_0{}'s_1w_2 + s_0s_1w_3$$



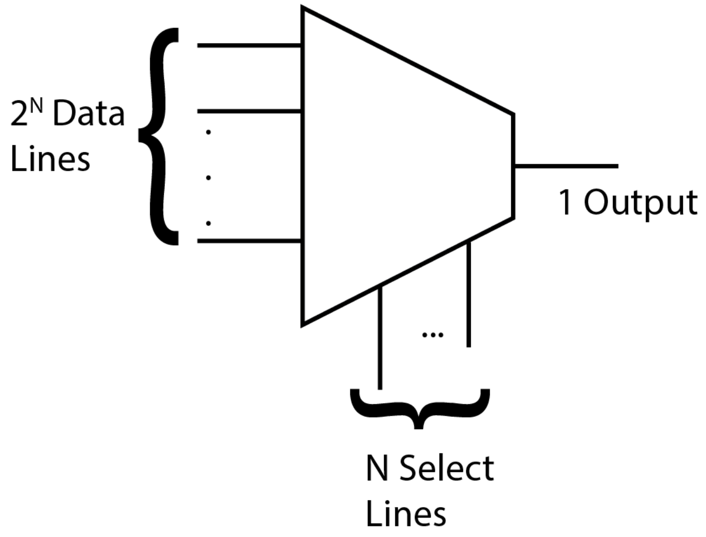
s_1	s_0	f
0	0	w_0
0	1	w_1
1	0	w_2
1	1	w_3



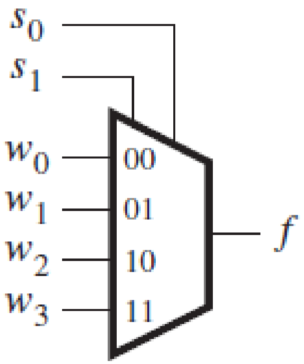
Large Multiplexer

- Similarly larger multiplexers can be built
- Usually data inputs is 2^N, select inputs are N and one output

- Larger multiplexers can be built using smaller multiplexors

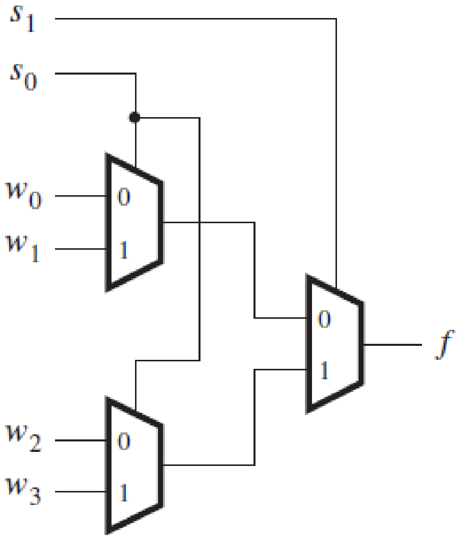


4-to-1 MUX using 2-to-1 MUX

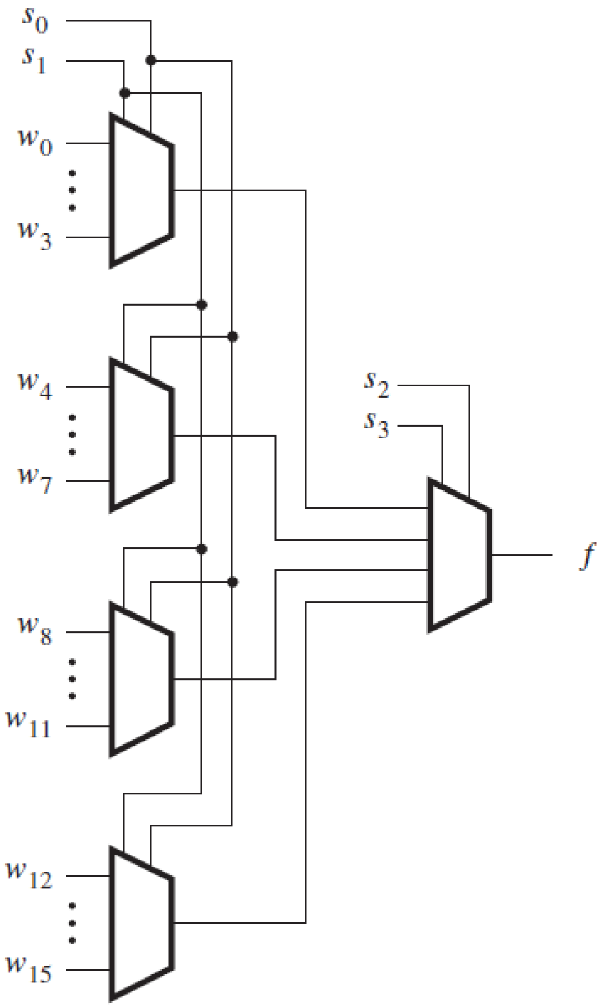


$$f = s_0's_1'w_0 + s_0s_1'w_1 + s_0's_1w_2 + s_0s_1w_3$$

$$= s_1'(s_0'w_0 + s_0w_1) + s_1(s_0'w_2 + s_0w_3)$$



16-to-1 MUX using 4-to-1 MUX



Practice Problem:

- Verify that the given circuit works as 16-to-1 MUX
- Implement 16-to-1 MUX using only 2-to-1 MUXes
- Implement 16-to-1 MUX using 8-to-1 MUXes and other components as required

MUX-based implementation using truth table (1)

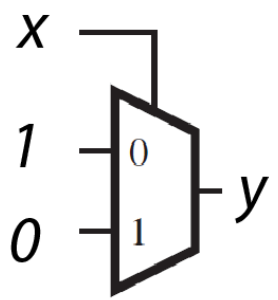
Problem: Given a function (or its truth table) of N variables, find a MUX-based implementation (using 2^N-to-1 MUX).

Problem 1: Given a function $y = x'$, find a MUX-based implementation.

Methodology:

- The input to a logic function can be used in the select line
- The value of the function corresponding to each row of the truth-table can be specified as data-inputs

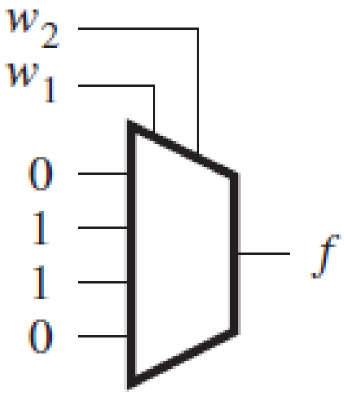
x	y
0	1
1	0



MUX-based implementation using truth table (2)

Problem 2: Given a function $f = w_1 \oplus w_2$, find a MUX-based implementation.

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



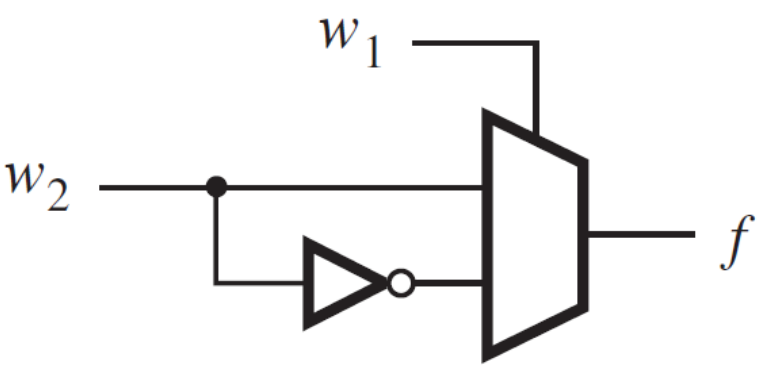
This is not an efficient implementation

MUX-based implementation using truth table (3)

Problem 3: Given a function $f = w_1 \oplus w_2$, find a ***2-to-1 MUX-based*** implementation.

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

w_1	f
0	w_2
1	$\overline{w}_2$



We will look at a formal way to obtain this implementation

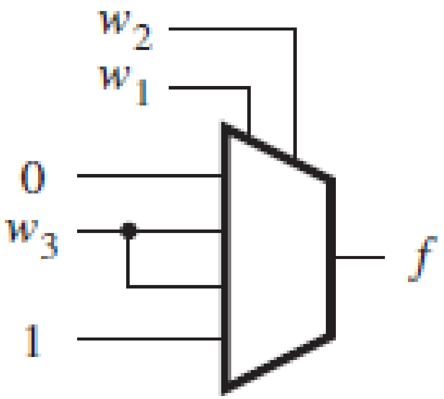
MUX-based implementation (4)

Problem 4: Given a majority function (truth table shown below), obtain a 4-to-1 MUX-based implementation.

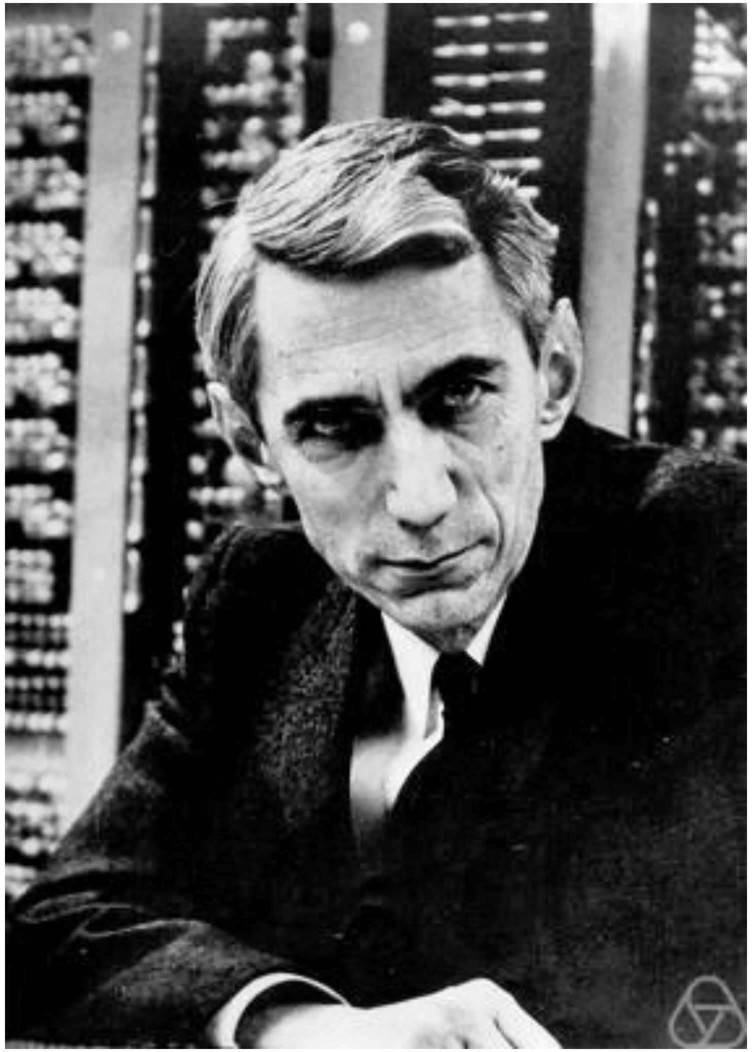
w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

w_1	w_2	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



MUX-synthesis: systematic way



Claude Shannon (1916 – 2001)

- American mathematician, electrical engineer, and cryptographer
- Known as "the father of information theory"

Shannon's Expansion Theorem

Shannon's Expansion Theorem:

Any Boolean function $f(w_1, w_2, \ldots, w_n)$ can be written in the form:

$$f(w_1, w_2, \ldots, w_n) = w_1'.f(0, w_2, \ldots, w_n) + w_1.f(1, w_2, \ldots, w_n)$$

- The term $\boldsymbol{f(0, w_2, \ldots, w_n)}$ is known as the cofactor of f with respect to $w_1{}'$
- It is obtained by replacing all occurrences of w_1 in f with 0
- It is denoted as $f_{w_1{}'}$

- The term $\boldsymbol{f(1, w_2, \ldots, w_n)}$ is known as the cofactor of f with respect to w_1
- It is obtained by replacing all occurrences of w_1 in f with 1
- It is denoted as f_{w_1}

$$f(w_1, w_2, \ldots, w_n) = w_1'.f_{w_1{}'} + w_1.f_{w_1}$$

Shannon's Expansion Theorem: Illustration

Problem:

Do Shannon's expansion of: $f = w_1 \oplus w_2$ and implement in terms of 2-to-1 MUX.

Solution:

$f = w_1 \oplus w_2 = w_1'.w_2 + w_1 w_2'$

$f_{w_1'} = w_2$

$f_{w_1} = w_2{}'$

$f = w_1'.f_{w_1'} + w_1.f_{w_1}$