CSE 241

Lecture 5

University at Buffalo
School of Engineering and Applied Sciences

Overview for this lecture

- Reminders
- Finish Combinational Logig

Reminders

- Including today there are 5 more sessions and then the final exam

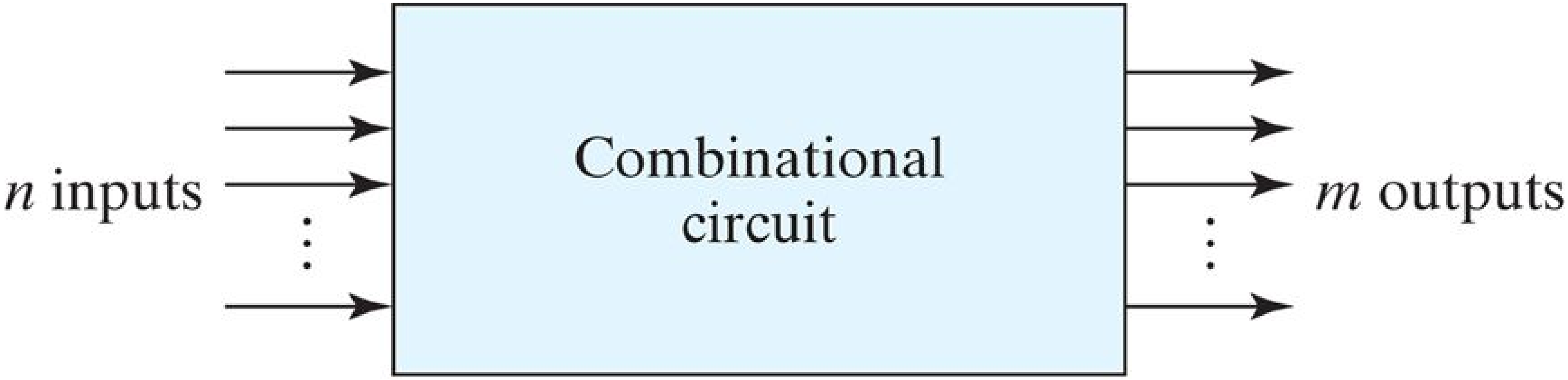
Exam Regrade Rules

- Treat as a take home exam
- Can earn up to half the points lost back
- Do not write on the original exam
  - if you do you will not be eligible for corresponding regrade points
- You must redo the entire problem, not just the part of the problem you got wrong
  - On the multipart problem each part (a, b c, ect.) you do not need to redo the entire problem, just the entire part of the problem (a, b, d, ect)
- Use your resources - open note, open book, ask the instructor
  - DO NOT TALK TO OTHER STUDENTS OR PEOPLE!!!
- Due at start of lecture on Wednesday

Questions?

What is Combinational Logic?

- Circuits fall into one of two categories
  - Combinational
  - Sequential
- These circuits depend on only the current inputs



Copyright ©2013 Pearson Education, publishing as Prentice Hall

MSI

- Medium Scale Integration
- Level of complexity of the combinational logic we will explore
- How Complex Is This?
  - All these circuits can be fabricated to an IC

Anyone recall what IC is short for?

Some key things about Combinational Logic Design

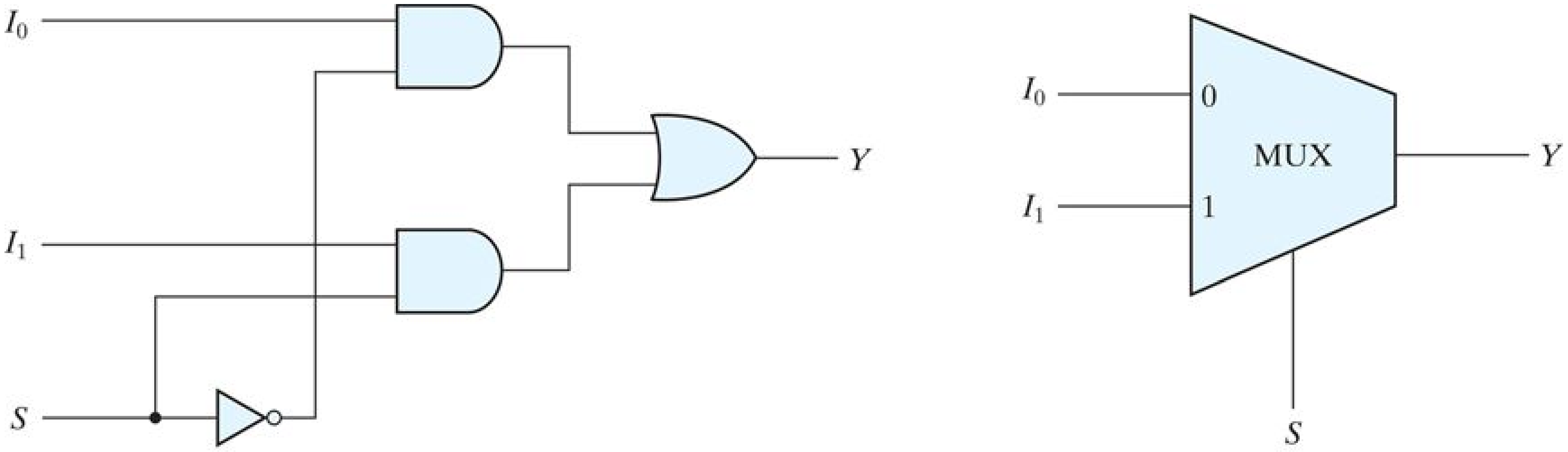
- No Feedback Paths
- No Memory Elements
- Can have multiple inputs *and* outputs

Design Procedure

1. Determine the number of inputs and outputs- Assign a symbol to each
2. Derive the Truth Table
3. Obtain the simplified Boolean functions for each output as a function of the inputs
4. Draw the logic diagram and verify correctness (manually or by simulation)

Multiplexers

- These are a fundamental of combinational logic design
- We often call them a mux or muxes
- There are power of 2 inputs
- Select one output
    - this is why we like them

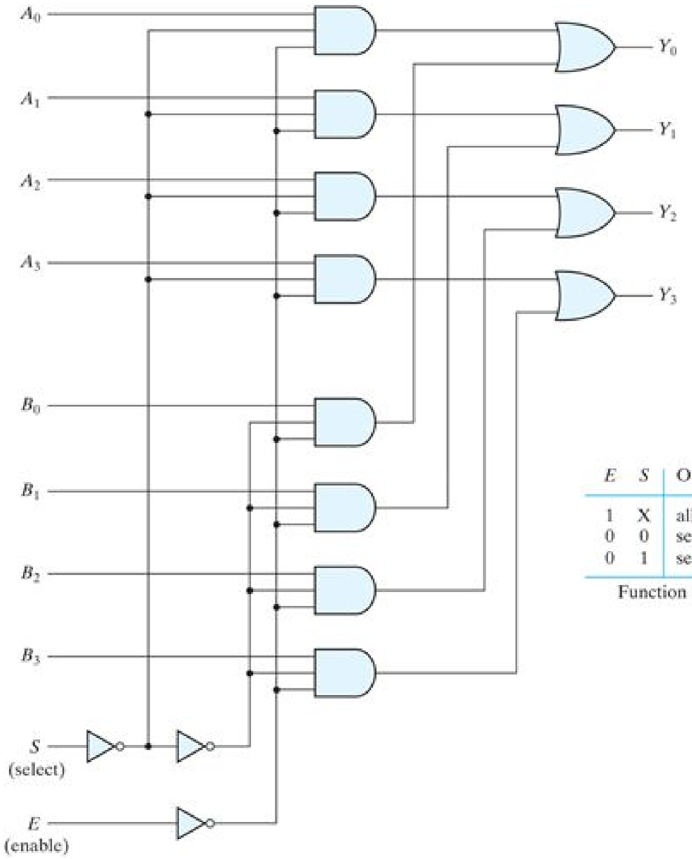


(a) Logic diagram

(b) Block diagram

Copyright ©2013 Pearson Education, publishing as Prentice Hall

We can build bigger

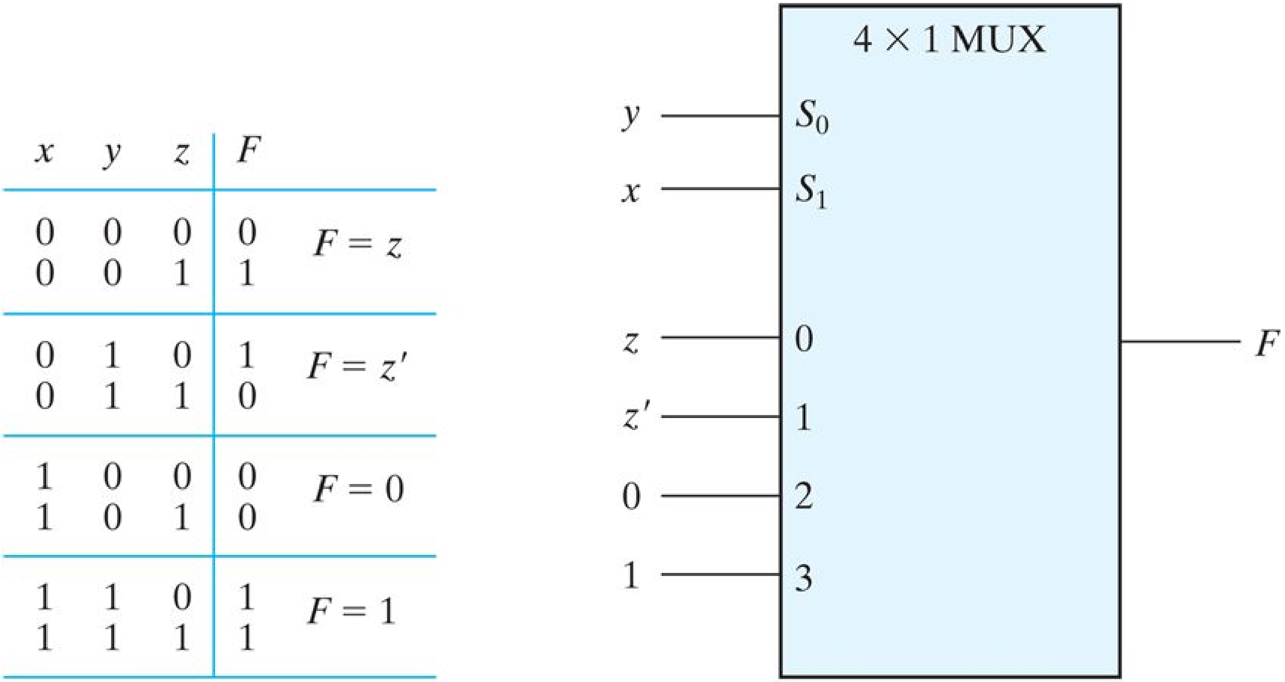


E	S	Output Y
1	X	all 0's
0	0	select A
0	1	select B

Function table

Copyright ©2013 Pearson Education, publishing as Prentice Hall

Or we can make more decisions



x	y	z	F	
0	0	0	0	$F = z$
0	0	1	1	
0	1	0	1	$F = z'$
0	1	1	0	
1	0	0	0	$F = 0$
1	0	1	0	
1	1	0	1	$F = 1$
1	1	1	1	

(a) Truth table

(b) Multiplexer implementation

Copyright ©2013 Pearson Education, publishing as Prentice Hall

But what is going on inside?

It is really just multiple 2 to 1 multiplexers

Let me show you...

Let’s work out a 8 to 1 Mux

How does this play into combinational logic?

- Look at the truth table two rows at a time.
    - Compare f with the least significant variable.
        - If f = 0, that data input is 0.
        - If f = 1, that data input is 1.
        - If f = the least significant variable w, that data input is w.
        - If f = w’, that data input is w’.

Let’s do an example

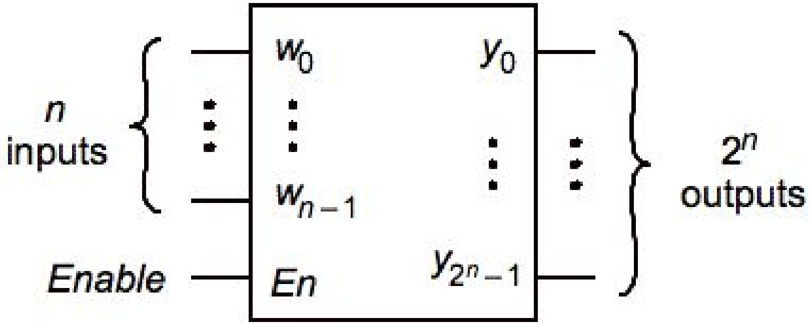
$$f(A,B,C) = \Sigma m(0,1,2,7)$$

Exercise

$$F(A,B,C) = \Sigma m(0,2,3,7)$$

Decoder

- Decoders decode encoded information.
- A decoder has
  - n data inputs,
  - an enable input
  - and typically 2n outputs.
- Only one output is active at a time.
- These are really useful for memory addressing



- A decoder generates all possible minterms of n variables.
- To design any combinational logic circuit simply connect the outputs for the desired minterms to an OR gate.

What does a 3 to 8 Line Decoder do?

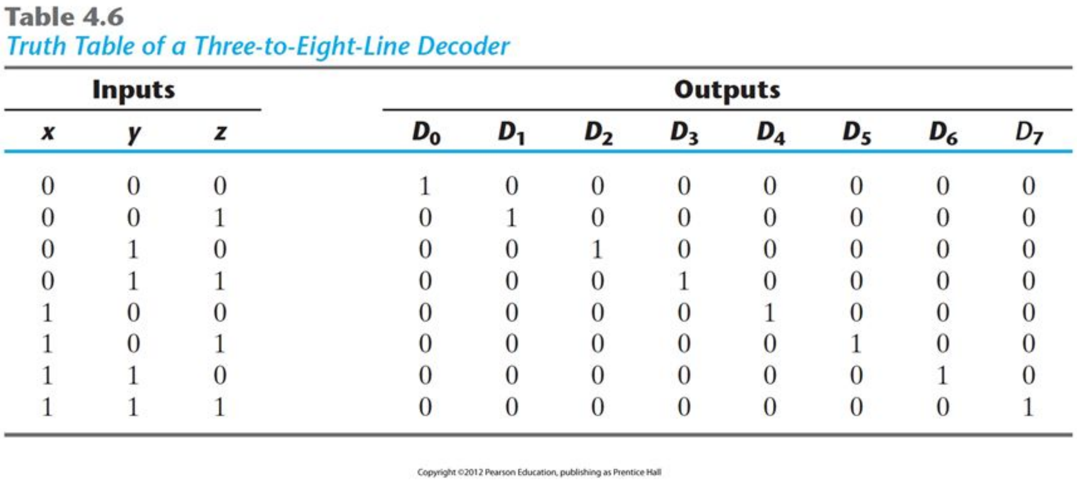
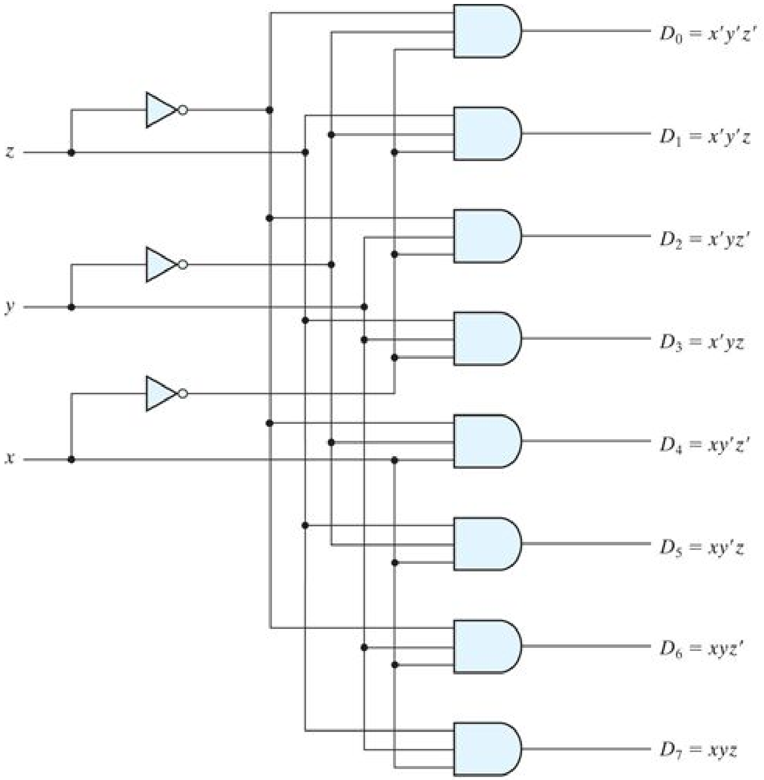


Table 4.6
Truth Table of a Three-to-Eight-Line Decoder

Inputs			Outputs							
x	y	z	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Copyright ©2012 Pearson Education, publishing as Prentice Hall



Copyright ©2013 Pearson Education, publishing as Prentice Hall

Let’s do an example of a decoder

$$F(A,B,C) = \Sigma m(0,1,2,7)$$

Exercise

$$F(A,B,C) = \Sigma m(0,2,3,6,7)$$

Memory Addressing with Decoders

- Decoders are used to implement memory addresses.
- Memory is organized in rows.
- The address is input to the decoder; the select lines at the decoder’s output select the row.

Read Only Memory - ROM

- A read-only-memory (ROM) provides permanent memory storage.
- It is composed of individual cells which are organized into rows of words.
- Variations are PROM and EEPROM which allow rewriting the contents of cells.

Decoding a Memory Address

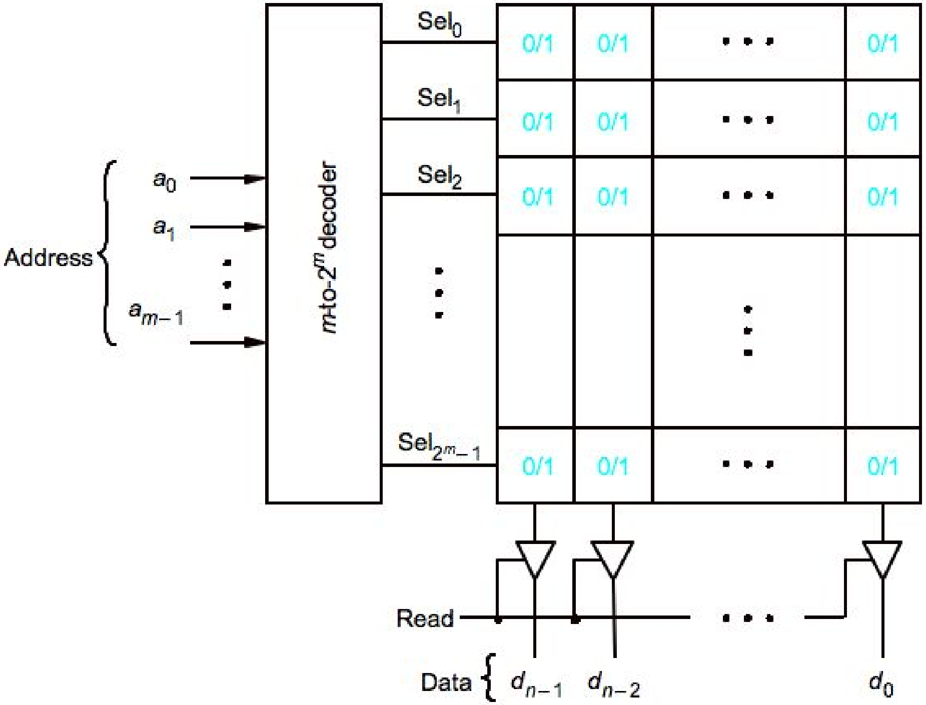


Figure 6.21. A 2^m x n read-only memory (ROM) block.

Demultiplexer

- A demultiplexer sends one data input to one of several outputs.
  - A decoder can serve as a demultiplexer.
  - Data is input on the enable line.
  - The other inputs determine which output line receives the data.
  - An n-to-2^n decoder can serve as a 1-to-n demultiplexer.

What is the opposite of Decoders?

Encoders

- Encoders are the opposite of decoders.
- They encode information into more compact forms.
- A binary encoder encodes information from 2^n inputs into an n-bit code.
- A priority encoder outputs the input with highest priority.

Priority Encoder

The function is if there are two or more inputs then the one with highest priority will take precedence

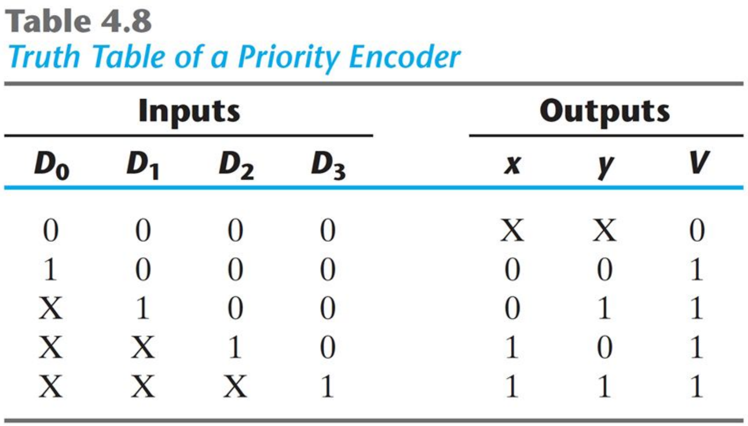
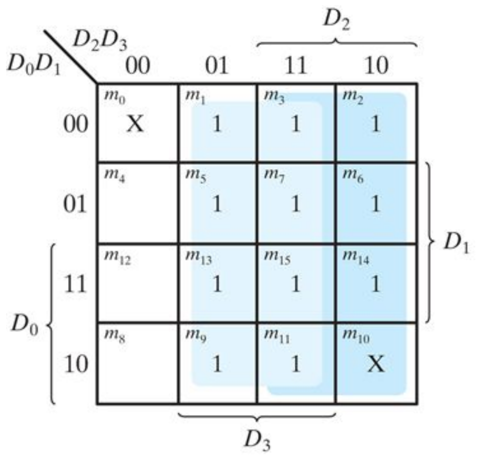


Table 4.8
Truth Table of a Priority Encoder

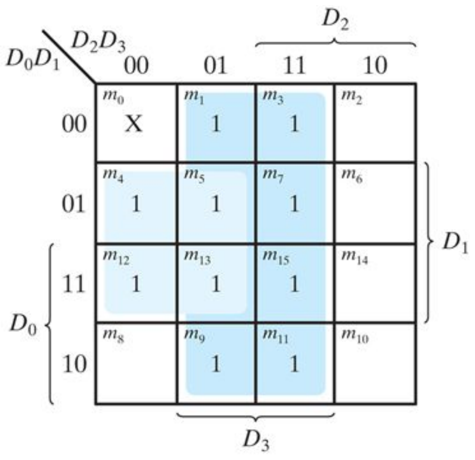
Inputs				Outputs		
D_0	D_1	D_2	D_3	x	y	V
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1

Copyright ©2012 Pearson Education, publishing as Prentice Hall



D_0D_1 \ D_2D_3	00	01	11	10
00	m_0 X	m_1 1	m_3 1	m_2 1
01	m_4	m_5 1	m_7 1	m_6 1
11	m_{12}	m_{13} 1	m_{15} 1	m_{14} 1
10	m_8	m_9 1	m_{11} 1	m_{10} X

$x = D_2 + D_3$



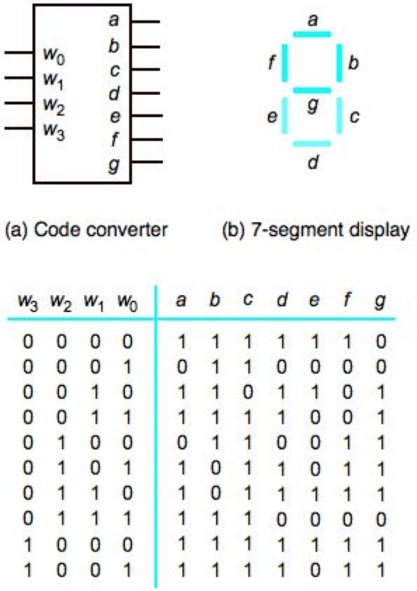
D_0D_1 \ D_2D_3	00	01	11	10
00	m_0 X	m_1 1	m_3 1	m_2
01	m_4 1	m_5 1	m_7 1	m_6
11	m_{12} 1	m_{13} 1	m_{15} 1	m_{14}
10	m_8	m_9 1	m_{11} 1	m_{10}

$y = D_3 + D_1D'_2$

Copyright ©2013 Pearson Education, publishing as Prentice Hall

Code Convertors

BCD to 7 Segment Display



(a) Code converter

(b) 7-segment display

w_3	w_2	w_1	w_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1

(c) Truth table

Figure 6.25. A BCD-to-7-segment display code converter.

Let’s Design a Display Decoder

- What does that mean?
  - Based off an Altera Lab
  - We will be able to control the placement of HELLO across 5- 7 segment Displays
- Where do we start?
  - The decoder for the letters
    - We will design a 3 to 7 decoder to control our 7-segment display
    - We won’t use all the combinations, unused combinations we will just turn the display off