

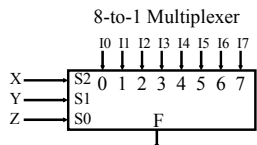
Implementing a Truth Table using a Multiplexer

From: http://class.ece.iastate.edu/arun/CprE281_F05/lectures/f05_week05.pdf

Start Reading Here:

Multiplexing and Multiplexer

- Multiplexers are circuits which select one of many inputs
- In here, we assume that we have one-bit inputs (in general, each input may have more than one bit)
- Suppose we have eight inputs: I0, I1, I2, I3, I4, I5, I6, I7
- We want one of them to be output based on selection signals
- 3 bits of selection signals to decide which input goes to output
- Note the order of selection signals
 - X is MSB and Z is LSB



X	Y	Z	F
0	0	0	I0
0	0	1	I1
0	1	0	I2
0	1	1	I3
1	0	0	I4
1	0	1	I5
1	1	0	I6
1	1	1	I7

CprE 210

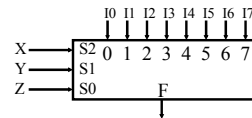
Lec 15

1

Multiplexer Implementation

- We can write a logic expression for output F as follows

$$F = X'Y'Z'I_0 + X'Y'Z'I_1 + X'Y'Z'I_2 + X'Y'Z'I_3 + X'Y'Z'I_4 + X'Y'Z'I_5 + X'Y'Z'I_6 + X'Y'Z'I_7$$
- This circuit can be implemented using
 - eight 4-input AND gates and one 8-input OR gates



X	Y	Z	F
0	0	0	I0
0	0	1	I1
0	1	0	I2
0	1	1	I3
1	0	0	I4
1	0	1	I5
1	1	0	I6
1	1	1	I7

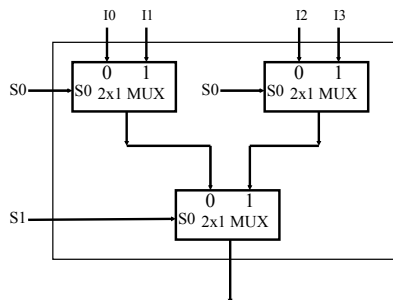
CprE 210

Lec 15

2

Don't Worry About

Implementing 4-to-1 MUX using 2-to-1 MUXs



CprE 210

Lec 15

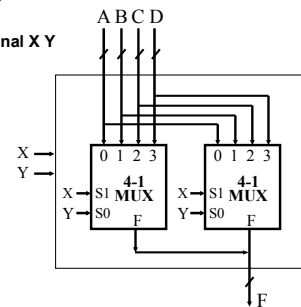
3

Don't Worry About

Making a 2-bit 4-to-1 Multiplexer

- Four 2-bit inputs A, B, C, D
- One 2-bit output F
- Two bits of selection signal X Y

X	Y	F
0	0	A
0	1	B
1	0	C
1	1	D



CprE 210

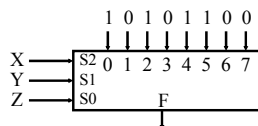
Lec 15

4

Important

Synthesis of Logic Functions using Multiplexers

- Multiplexers can be directly used to implement a function
- Easiest way is to use function inputs as selection signals
- Input to multiplexer is a set of 1s and 0s depending on the function to be implemented
- We use a 8-to-1 multiplexer to implement function F
- Three select signals are X, Y, and Z, and output is F
- Eight inputs to multiplexer are 1 0 1 0 1 1 0 0
- Depending on the input signals
 - multiplexer will select proper output



X	Y	Z	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

CprE 210

Lec 15

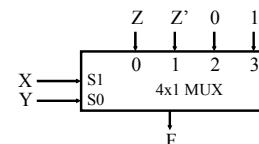
5

Important

Implementing 3-variable functions with 4x1 MUX

- Divide the outputs into 4 groups based on X and Y.
- Write the outputs as a function of Z
- There are only 4 possibilities: $F=Z$, $F=Z'$, $F=0$, $F=1$

X	Y	Z	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1



CprE 210

Lec 15

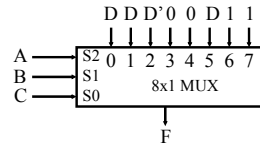
6

Important:

Don't Worry about

Implementing 4-variable functions with 8x1 MUX

A	B	C	D	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1



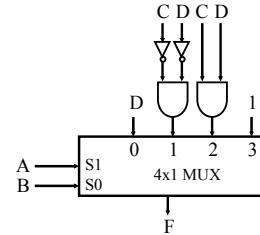
CprE 210

Lec 15

7

Implementing 4-variable functions with 4x1 MUX

A	B	C	D	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1



CprE 210

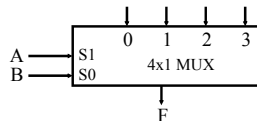
Lec 15

8

Don't worthy about the rest.

Implementing 4-variable functions with 4x1 MUX

A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1



CprE 210

Lec 15

9

Definition of Decoder

- Suppose we have n input bits (which can represent up to 2^n distinct elements of coded information).
- We need a device that allows us to select which of the 2^n elements, devices, memory locations, etc. is being selected.
- In general:
 - A decoder has n input bits
 - A decoder has 2^n (or less) output bits
 - As a rule, all but one of the outputs is zero (deselected) at any time (called *one-hot encoded*)

CprE 210

Lec 15

10

2-to-4 Decoder

- The 2-to-4 decoder is a block which decodes the 2-bit binary inputs and produces four outputs
- One output corresponding to the input combination is a one
- Two inputs and four outputs are shown in the figure
- The equations are
 - $y_0 = x_1' \cdot x_0'$
 - $y_1 = x_1' \cdot x_0$
 - $y_2 = x_1 \cdot x_0'$
 - $y_3 = x_1 \cdot x_0$
- The truth table:

x1	x0	y3	y2	y1	y0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

CprE 210

Lec 15

11

Definition of Encoder

- Encoders perform the inverse function of Decoders.
- An encoder has 2^n (or less) input bits and n output bits
- The output bits generate the binary code corresponding to the input value
- Assuming only one input has a value of 1 at any given time
- Example: An 8-to-3 Encoder

Inputs								Outputs		
D7	D6	D5	D4	D3	D2	D1	D0	A2	A1	A0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

$$\begin{aligned} A_2 &= D_4 + D_5 + D_6 + D_7 \\ A_1 &= D_2 + D_3 + D_6 + D_7 \\ A_0 &= D_1 + D_3 + D_5 + D_7 \end{aligned}$$

CprE 210

Lec 15

12