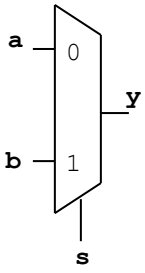
Notes - Chapter 3

MULTIPLEXERS (MUXS)

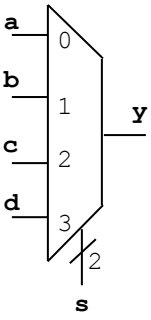
- This logic circuit selects one of many input signals and forwards the selected input to the output line.
- Boolean equations for MUX2-to-1, MUX4-to-1, MUX8-to-1:



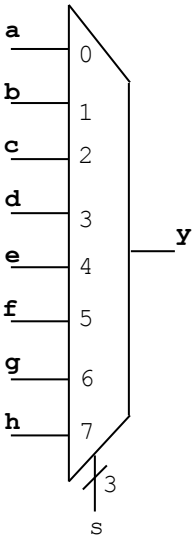
s	a	b	y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

s	y
0	a
1	b

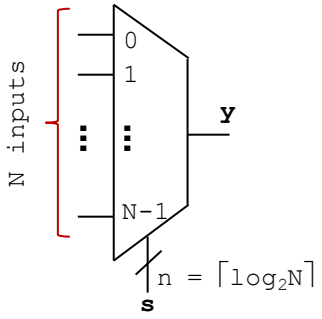
$$y = \bar{s}a + sb$$



$$y = \bar{s}_1\bar{s}_0a\ \bar{s}_1s_0b + s_1\bar{s}_0c + s_1s_0d$$



$$y = \bar{s}_2\bar{s}_1\bar{s}_0a + \bar{s}_2\bar{s}_1s_0b + \bar{s}_2s_1\bar{s}_0c + \bar{s}_2s_1s_0d + s_2\bar{s}_1\bar{s}_0e + s_2\bar{s}_1s_0f + s_2s_1\bar{s}_0g + s_2s_1s_0h$$



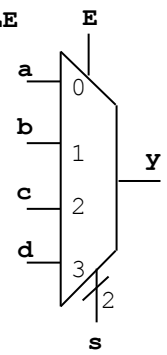
- Normally, a multiplexer has $N = 2^n$ inputs, one output, and a selector with n bits.
- But, if a multiplexer has N inputs, where N is not a power of 2, the number of bits of the selector is given by: $\lceil log_2 N \rceil$.

Multiplexers with Enable

- An enable input provides us with an extra level of control. If the multiplexer is enabled, the circuit just works. If the multiplexer is not enabled, no input is allowed into the output, and the multiplexer output becomes '0' (if the output is active-high) or '1' (if the output if active-low).
- The enable input can be either active-high or active-low:

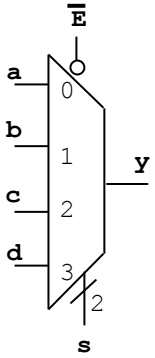
ACTIVE HIGH ENABLE

E	s_1	s_0	y
1	0	0	a
1	0	1	b
1	1	0	c
1	1	1	d
0	X	X	0



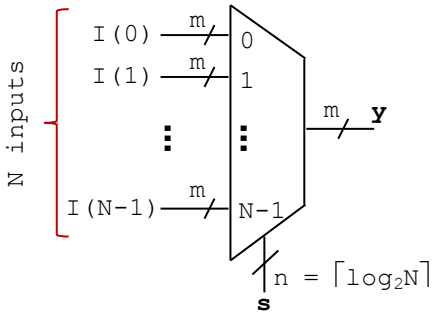
ACTIVE LOW ENABLE

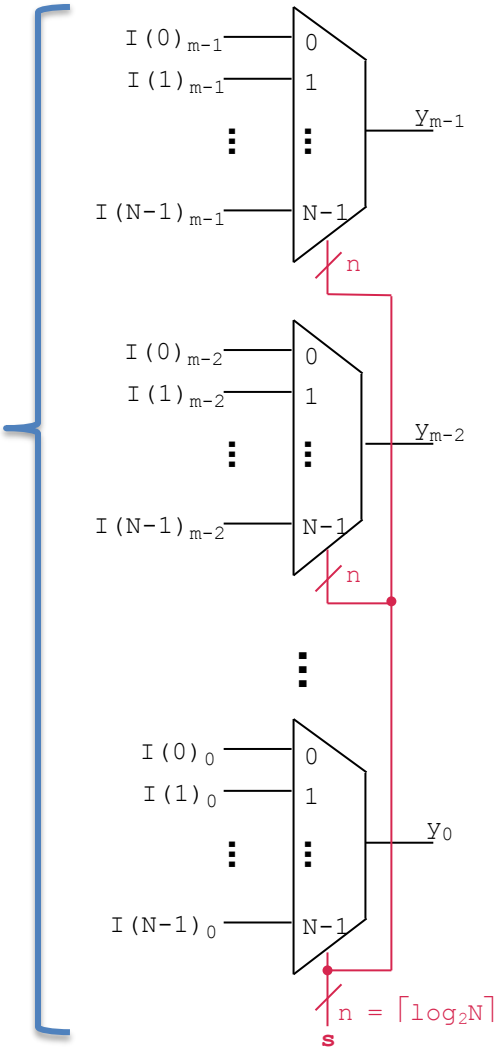
$\bar{E}$	s_1	s_0	y
0	0	0	a
0	0	1	b
0	1	0	c
0	1	1	d
1	X	X	0



Bus Multiplexers

- Usually we want input signals to contain more than one bit.
- In the figure, each input signal contains 'm' bits.
- This 'bus multiplexer' can be built by 'm' multiplexers, each taking care of only one bit for all the inputs.

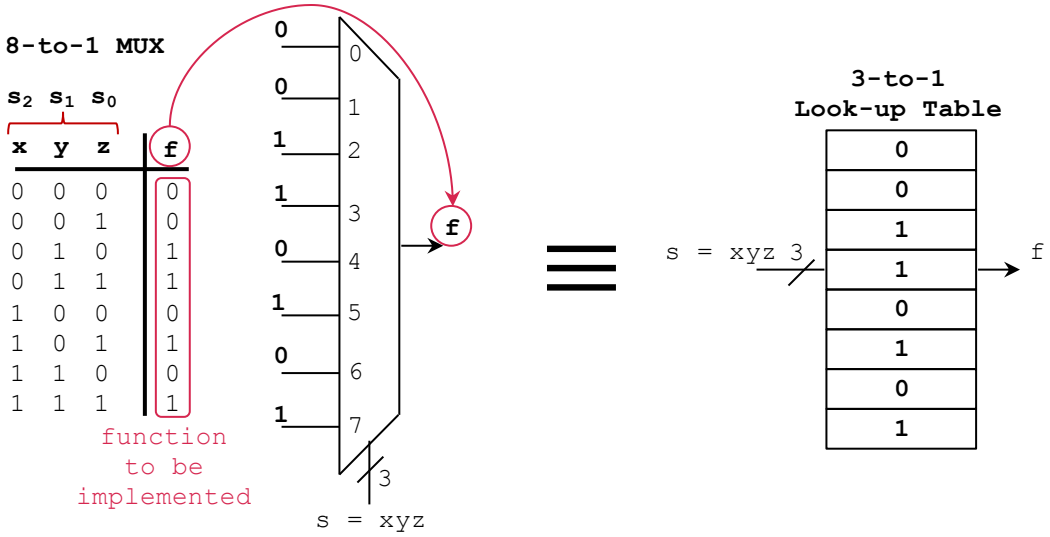




- We have 'N' inputs and therefore the selector has $n = \lceil log_2 N \rceil$ bits.
- Note that the selector is the same for all the multiplexers.

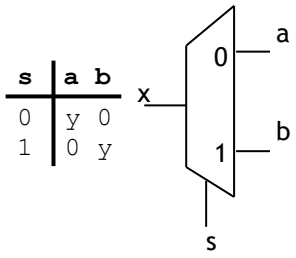
Logic Circuits with MUXs

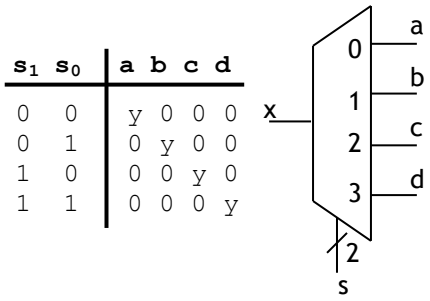
- Multiplexers can be used to implement Boolean Functions. The selector can be thought as the input variables, the input bits are fixed values that are passed onto the output according to the selector.
- This multiplexor with fixed inputs implements a logic function. The functionality of this circuit is similar to that of a Look-Up Table (LUT), which is a ROM-like circuit whose values are obtained by addressing them. FPGAs implement Boolean functions using LUTs. In the example, a 3-to-1 LUT is an LUT with 3 inputs, i.e., it contains $2^3 = 8$ addresses.



DEMULTIPLEXERS

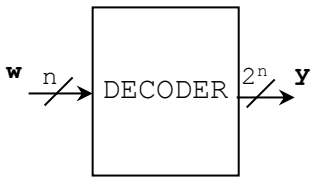
- A demultiplexer performs the opposite operation of the multiplexers.

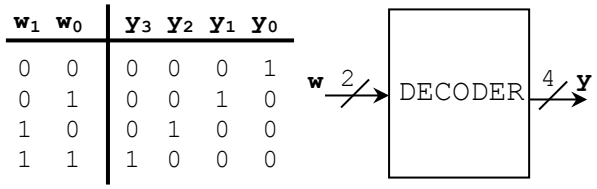


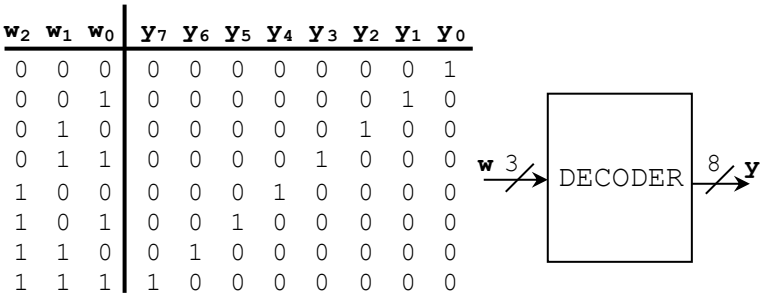


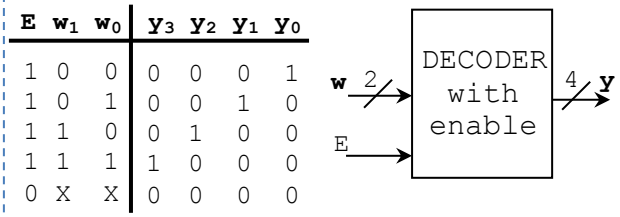
DECODERS

- Generally speaking, decoders are circuits that transform the inputs into outputs following a certain rule, provided that the number of outputs is greater than or equal to the number of inputs.
- Here, we discuss standard decoders for which a specific input/output rule exists. These decoders have n inputs and 2^n outputs. We show examples of: a 2-to-4 decoder, 3-to-8 decoder, and a 2-to-4 decoder with enable. The output y_i is activated when the decimal value of the input w is equal to i.



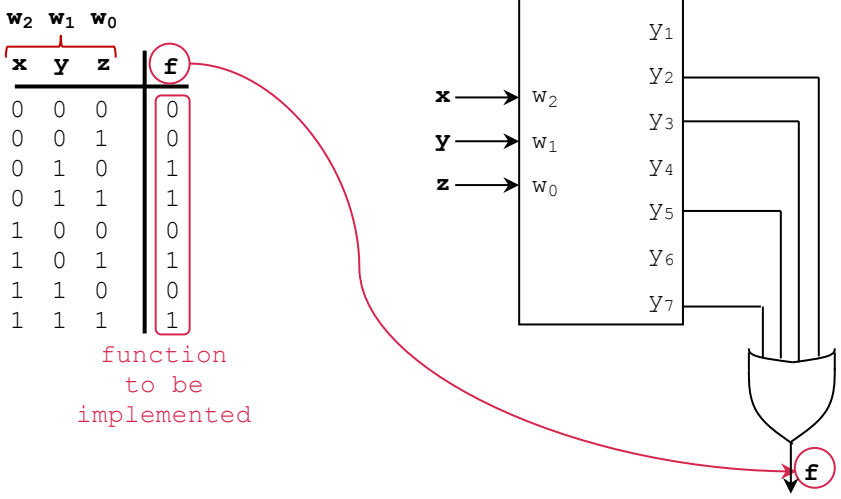






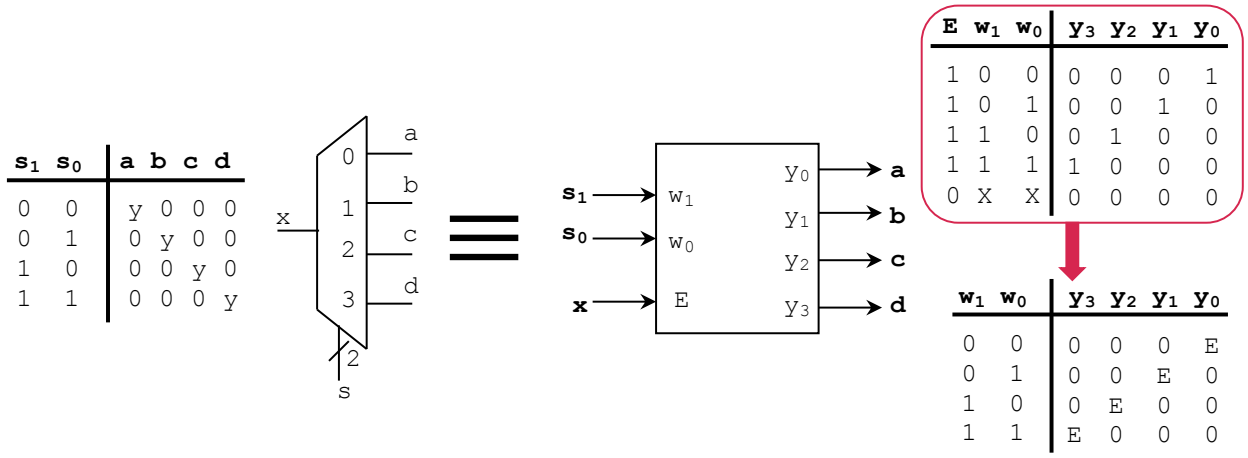
Logic Circuits with Decoders

- Decoders can be used to implement Boolean functions. Note that each output is actually a minterm:



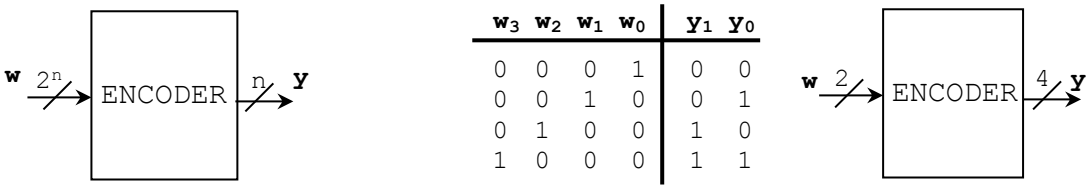
Implementing Demultiplexors with Decoders

- By utilizing the enable input of a decoder as our input signal, we can effectively implement a demultiplexor using a decoder:



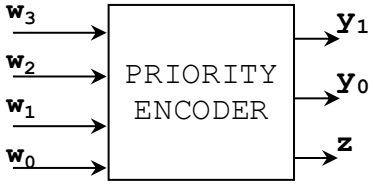
ENCODERS

- Generally speaking, encoders are circuits that transform the inputs into outputs following a certain rule, provided that the number of outputs is lower than the number of inputs.
- Here, we discuss standard encoders for which a specific input/output rule exists. These encoders have 2^n inputs and n outputs. The operation is exactly the opposite as in the case of the decoder: whenever and input w_i is activated, then the index i appears at the output y (in binary form).



Priority Encoders:

- Standard encoder: we check whether an specific input is activated for the output to have a value.
- What happens when more than one input is activated? A solution is to create an extra output that is activated to indicate than an unexpected condition has occurred.
- Another more interesting solution is to create a ***priority encoder***, that is if more than one input is activated, then we only pay attention to the input bit of the highest order. For example if $w = 1101$, then we only pay attention to $w(3) = 1$, if $w = 0111$, we only pay attention to $w(2) = 1$. This results in the following truth table for a 4-to-2 priority encoder:



w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	0	0	0
1	x	x	x	1	1	1
0	1	x	x	1	0	1
0	0	1	x	0	1	1
0	0	0	1	0	0	1

- What happens when no input is activated? Here we run out of output bits in y to represent this case. Thus, we include an extra output z that it is '0' when no input activated, and '1' otherwise.

PRACTICE EXERCISES

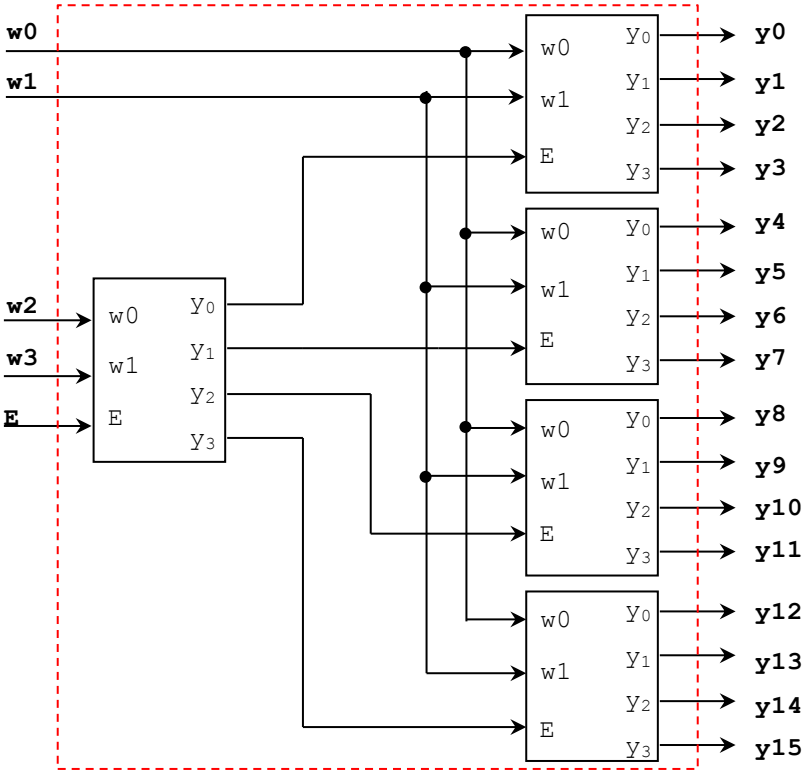
1. Implement the following functions using i) decoders and ii) multiplexers:

✓ $F = \overline{X + Y} + ZY$	✓ $F = (X + Y + Z)(X + Y + \bar{Z})$
✓ $F(X, Y, Z) = \sum(m_0, m_2, m_6)$.	✓ $F = XY + YZ + XZ$
✓ $F(X, Y, Z) = \prod(M_2, M_4, M_7)$	✓ $F = X \oplus Y \oplus Z$

2. Using ONLY 4-to-1 MUXs, implement an 8-to-1 MUX.

3. Implement a 6-to-1 MUX using i) only NAND gates, and ii) only NOR gates.

4. Verify that the following circuit made of out of five 2-to-4 decoders with enable represents a 4-to-16 decoder with enable. Tip: Create the truth table.



5. Using only 2-to-1 MUXs, implement the XOR and XNOR gates.

6. Using only a 4-to-1 MUX, implement the following functions.
   - $F(X, Y, Z) = \sum(m_1, m_3, m_5, m_7)$.
   - $F(X, Y, Z) = \sum(m_1, m_3, m_5)$
   - $F(X, Y, Z) = \sum(m_3, m_5, m_7)$.
   - $F(X, Y, Z) = \sum(m_5, m_7)$.

7. Complete the following timing diagram: