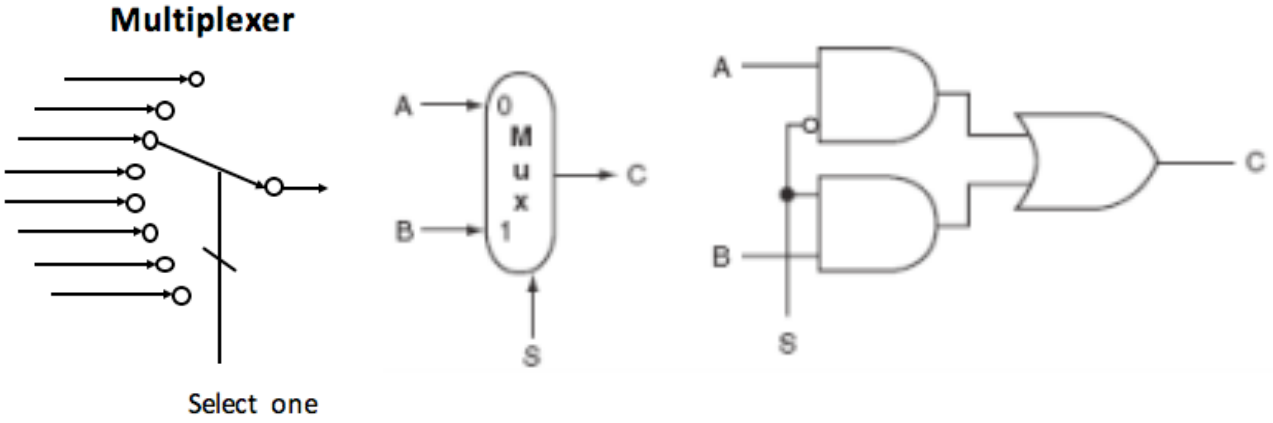
CS 240 Lab 3
Combinational and Arithmetic Circuits

- **Multiplexer/Demutiplexer**
- **Decoder/Encoder**
- **Adder**

Multiplexer

A multiplexer can be thought of as a **selection circuit**, which steers a single input from a set of inputs through to the output, based on the select line.



- n select lines
- 2^n input lines
- 1 output

One of the possible 2^n inputs is chosen by the n select lines, and gated through to the output of a multiplexer. The truth table for an 8x1 MUX is:

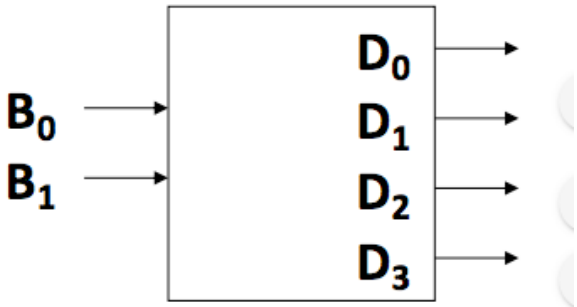
S2	S1	S0	Q
0	0	0	D0
0	0	1	D1
0	1	0	D2
0	1	1	D3
1	0	0	D4
1	0	1	D5
1	1	0	D6
1	1	1	D7

Decoder

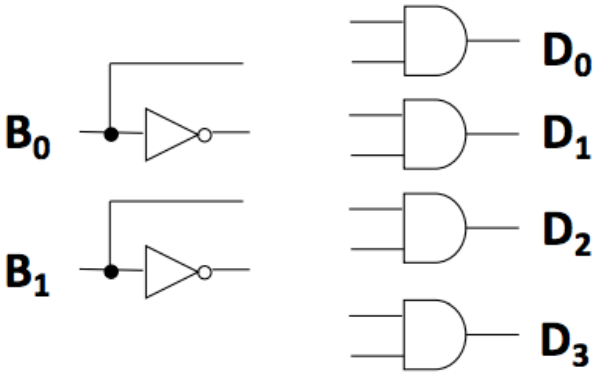
A decoder takes an n-bit binary number as an input, and asserts the corresponding numbered output from the set of 2^n outputs.

- n input/select lines
- 2^n outputs
- only one of the outputs is active at any given time, based on the value of the n select lines.

2x4 Decoder



Built with code detectors:



Truth table for an 3x8 decoder

S2	S1	S0		Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7
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

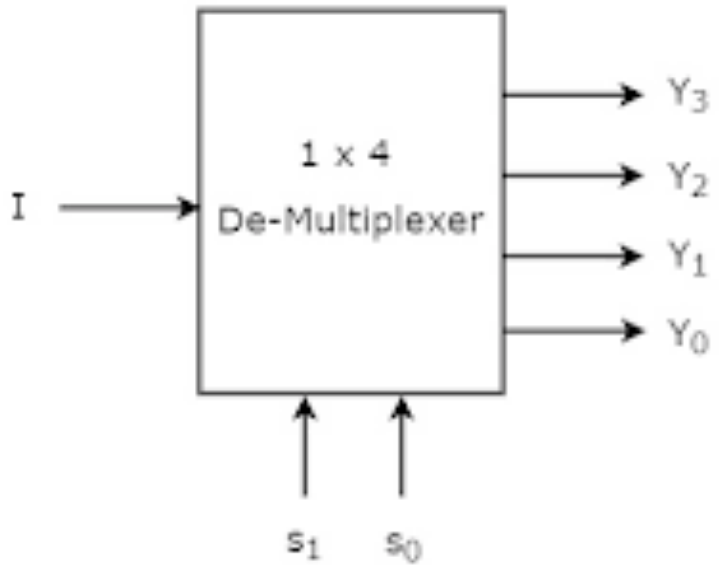
Demultiplexer

Opposite of multiplexer

Single input data line

Input through to a single one of the 2^n output lines

Output line is determined by the n select inputs



S1	S0	I	Y3	Y2	Y1	Y0
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

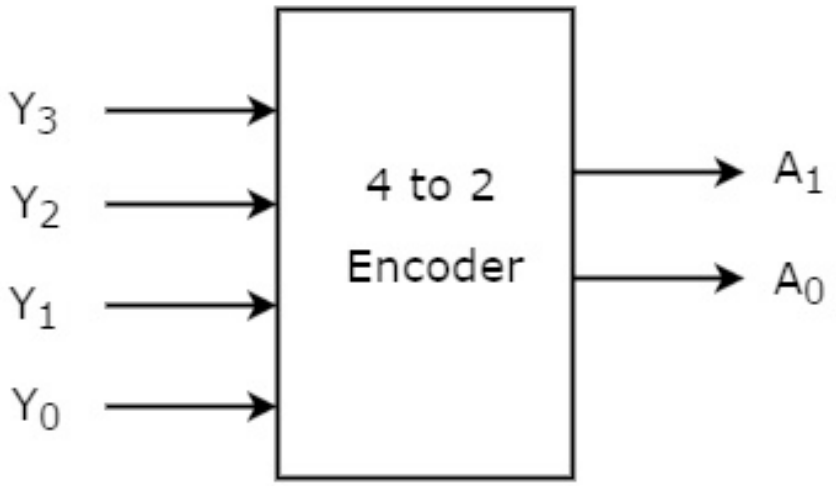
Encoder

Opposite of decoder

2^n inputs - only 1 of the inputs can be active at a time

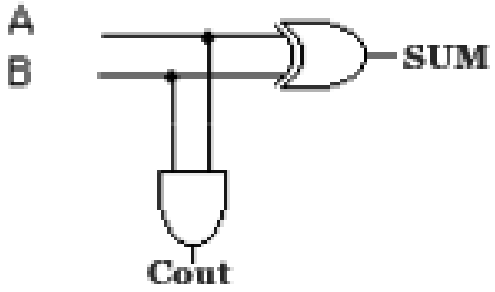
Input selected specifies a decimal number which corresponds to the number of the input (3, 2, 1, or 0)

n outputs represent the corresponding binary representation of the decimal value specified by the input



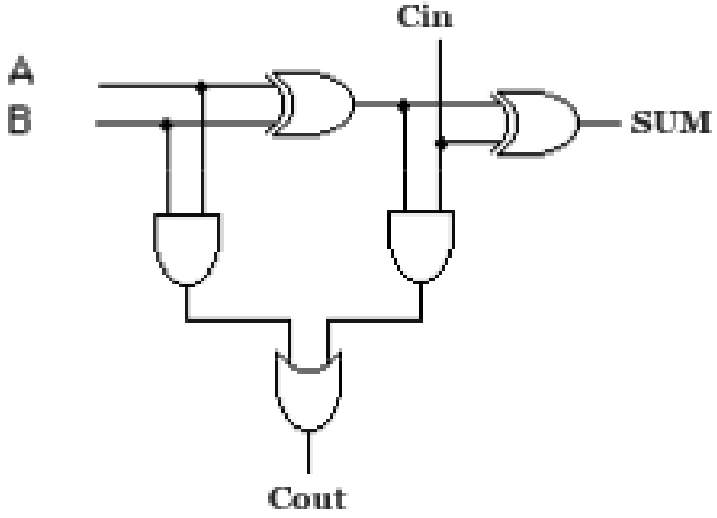
Y3	Y2	Y1	Y0	A1	A0
0	0	0	1		
0	0	1	0		
0	1	0	0		
1	0	0	0		

Half-Adder – adds two one-bit values



A	B	Cout	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Full Adder – uses two half-adders and incorporates a carry-in



Cin	A	B	Cout	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Sum = A⊕B⊕Cin

Cout = AB+(A⊕B)Cin

4-bit Ripple-Carry Adder