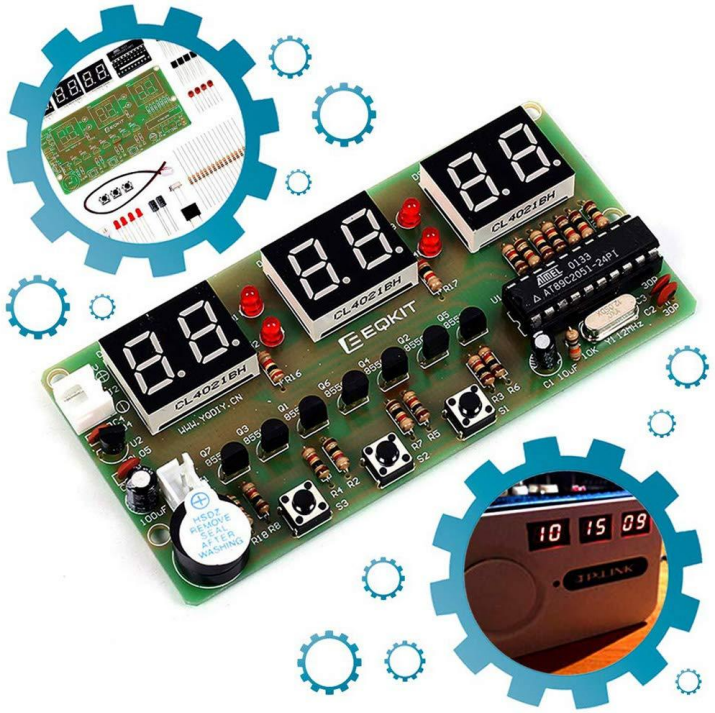
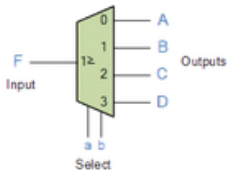



Dr. Hiran Ekanayake

# COMBINATIONAL LOGIC CIRCUITS

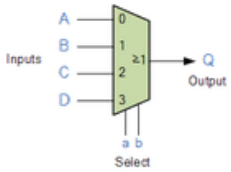
# References

- Combinational Logic Circuits
  - https://www.electronics-tutorials.ws/category/combination



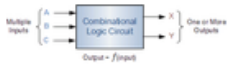

### The Demultiplexer

The data distributor, known more commonly as a Demultiplexer or "Demux" for short, is the exact opposite of the Multiplexer we saw in the previous tutorial. The demultiplexer takes one single input data line and then switches it to any one of a number of individual output lines ...




### The Multiplexer

Multiplexing is the generic term used to describe the operation of sending one or more analogue or digital signals over a common transmission line at different times or speeds and as such, the device we use to do just that is called a Multiplexer. The multiplexer, shortened to "...




### Combinational Logic Circuits

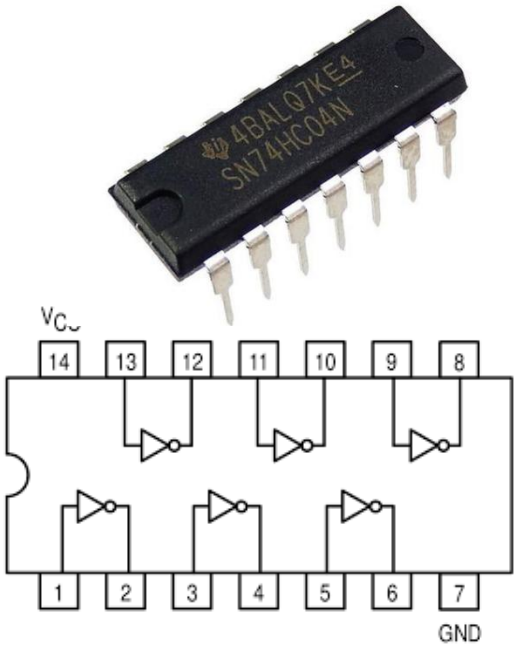
Unlike Sequential Logic Circuits whose outputs are dependant on both their present inputs and their previous output state giving them some form of Memory. The outputs of Combinational Logic Circuits are only determined by the logical function of their current input state, logic "...

# Lesson Outline

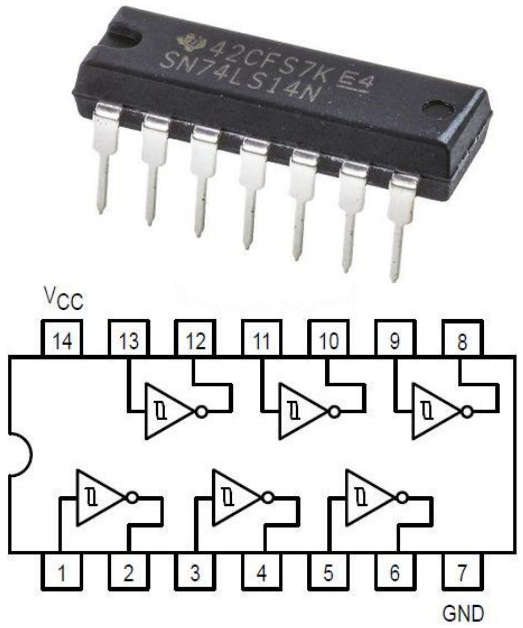
- Schmitt Trigger (Re.)
- IC Testers
- Combinational Logic Circuits Introduction
- Multiplexer
- Demultiplexer
- Encoder
- Decoder
- Binary Adder
- Digital Comparator
- Binary Subtractor
- ALU

# SCHMITT INVERTER

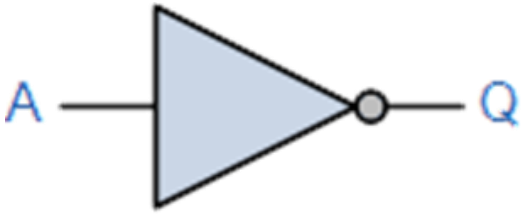
# Regular vs. Schmitt Inverters

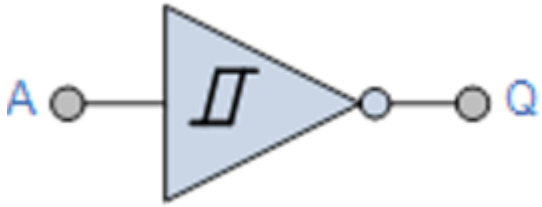



**Hex Inverter**

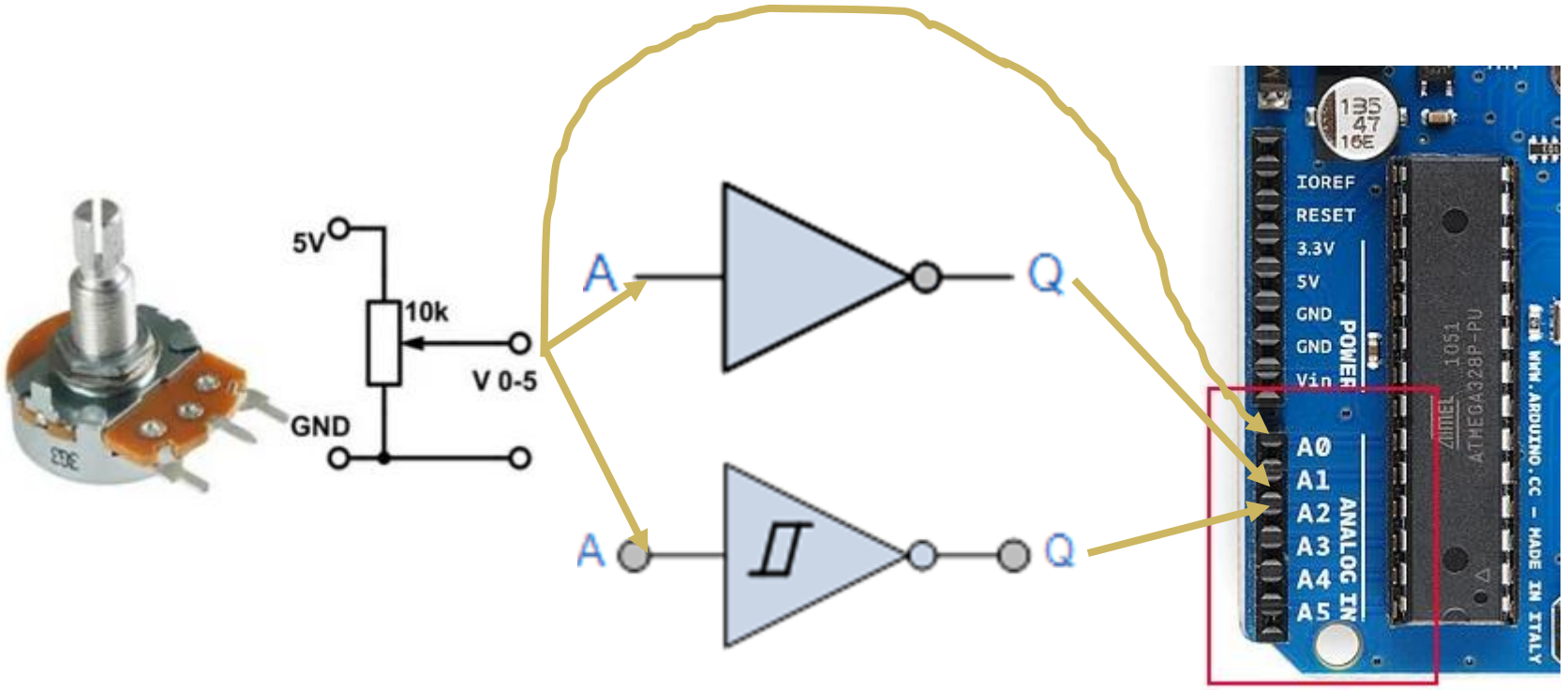



**Hex Schmitt Inverter**

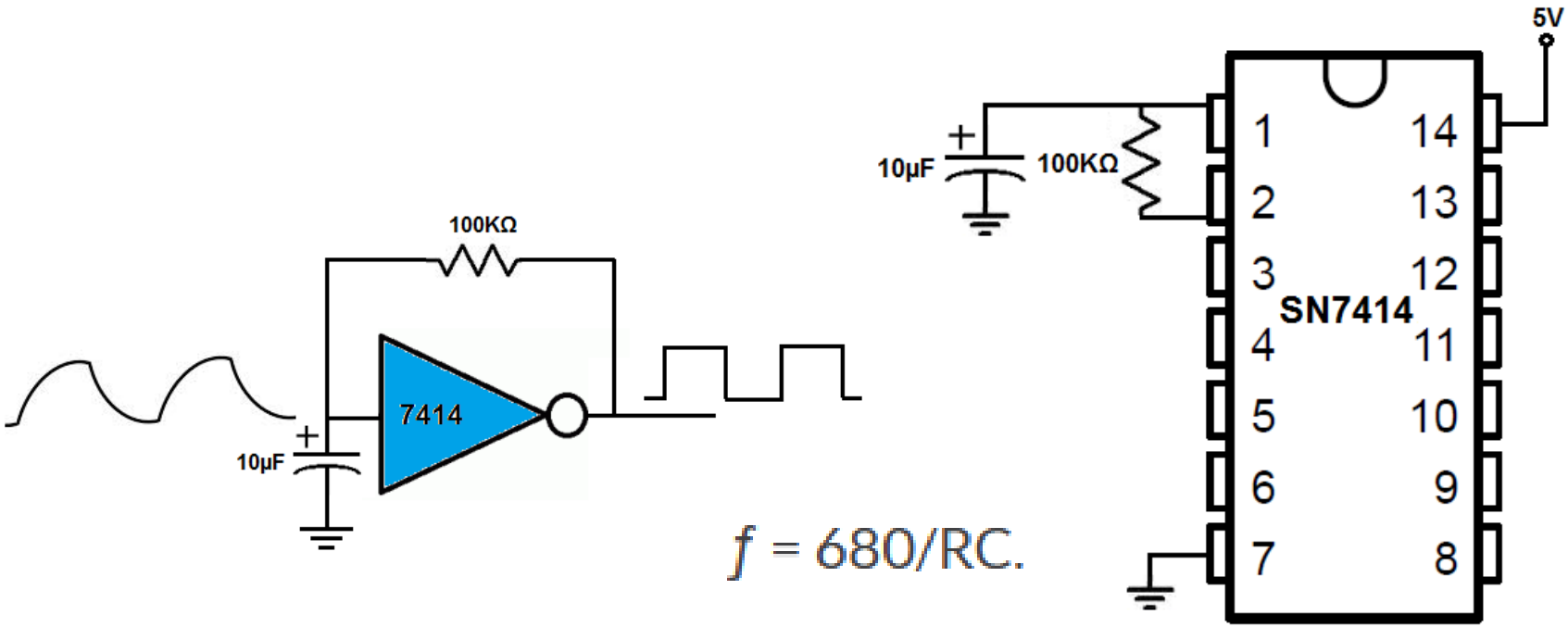


# Testing a Schmitt Inverter 1



# Testing a Schmitt Inverter 2




How do you build a capacitor meter using a Schmitt inverter and Arduino?

# IC TESTER

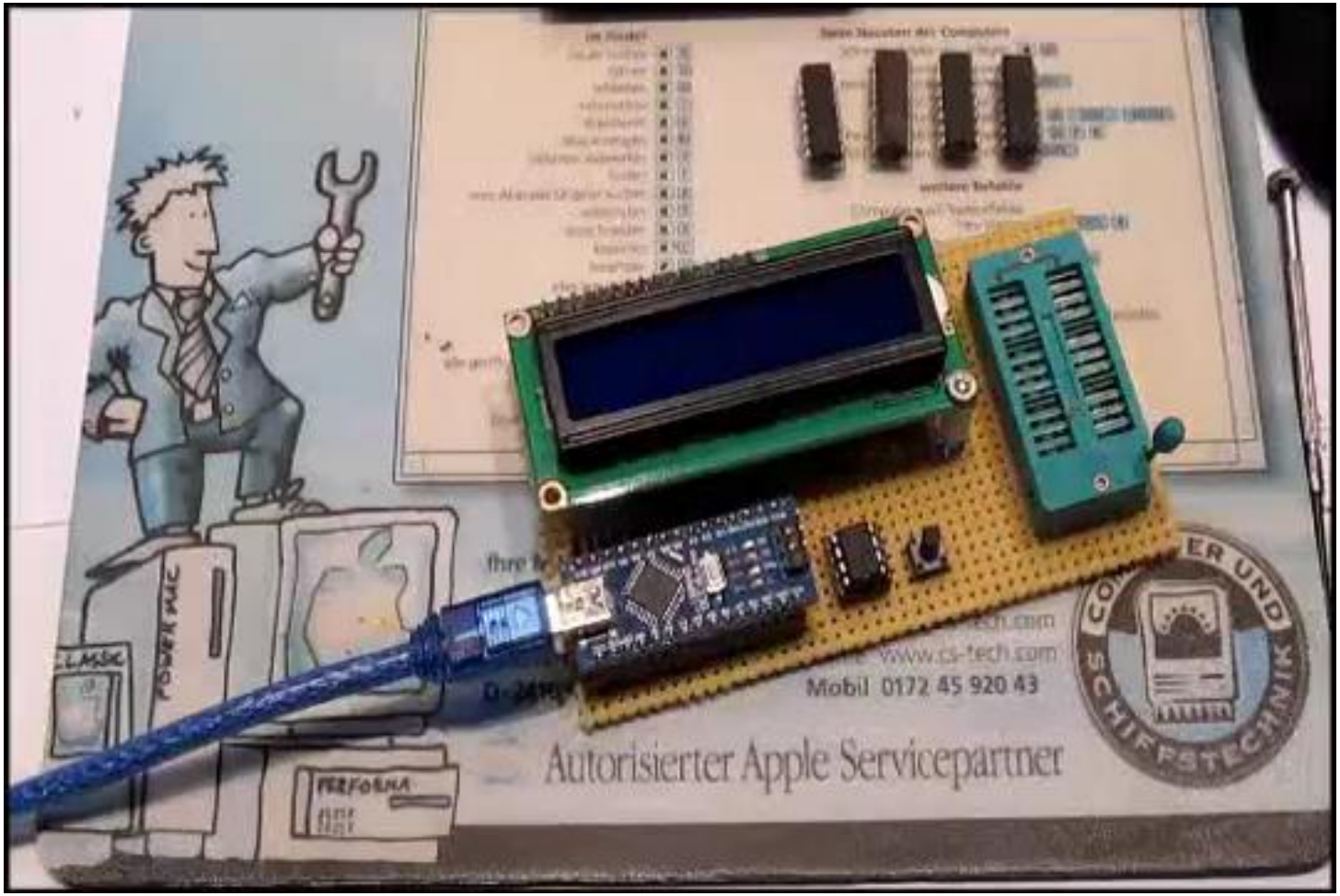
# TES200/210 IC Tester

https://www.youtube.com/watch?v=cELS7T_Eldg


Rs. 5,000-10,000/=

# DIY IC Tester Using Arduino

https://www.youtube.com/watch?v=kd2wyjB4ZwM



# Question

- Why it is important to learn digital electronics?
- Why you should learn digital electronics?

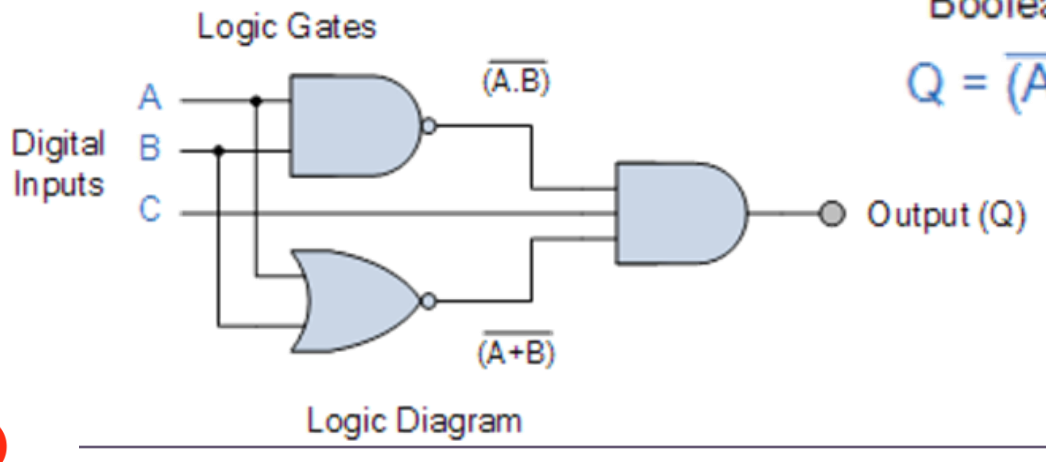
# COMBINATIONAL LOGIC CIRCUITS

# Combinational Logic Circuits

- What is a combinational logic circuit?
  - A combinational logic circuit is made up from basic logic gates to produce more complicated switching circuits
  - The output of a combinational logic circuit at any instant in time depends only on the combination of its inputs

# Combinational Logic Circuits

- How do you specify the function of a combinational logic circuit?
  - Using Boolean algebra
  - Using a truth table
  - Using a logic diagram




Boolean Expression

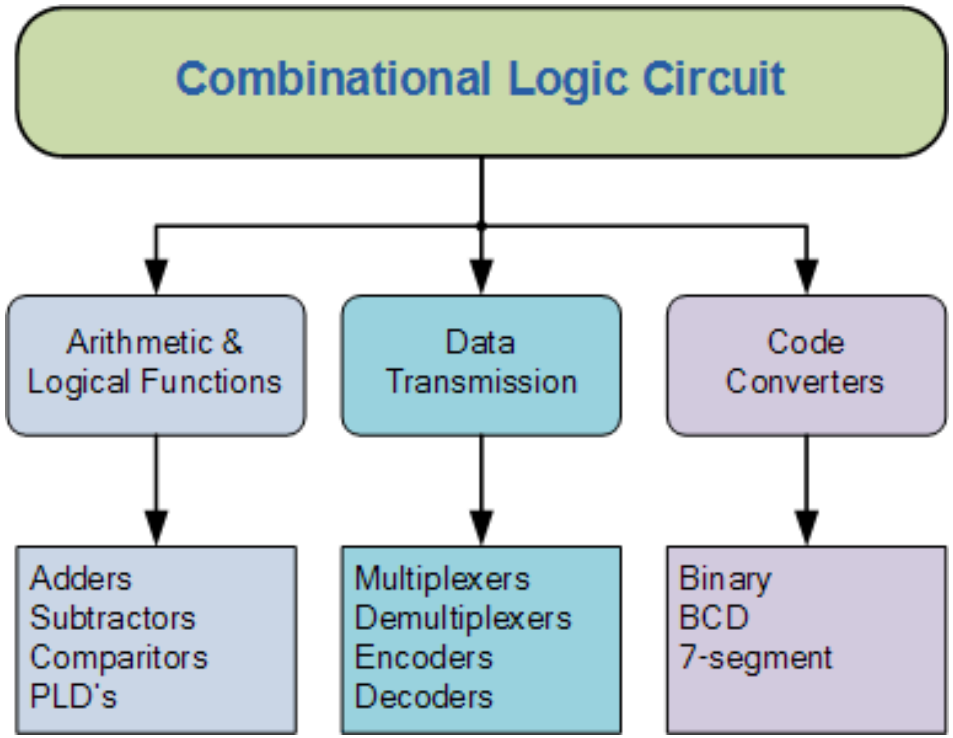
$$Q = \overline{(A.B)}.\overline{(A+B)}.C$$

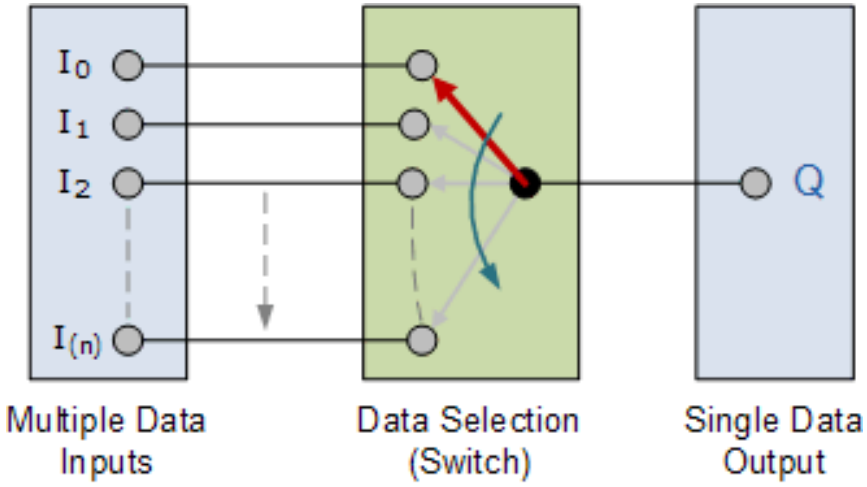
Typical Truth Table

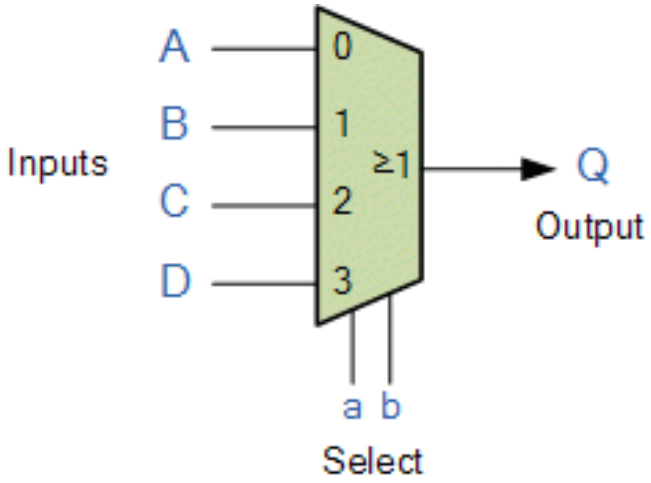
| C | B | A | Q |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

# Combinational Logic Circuits

- What are different types of combinational logic circuits?





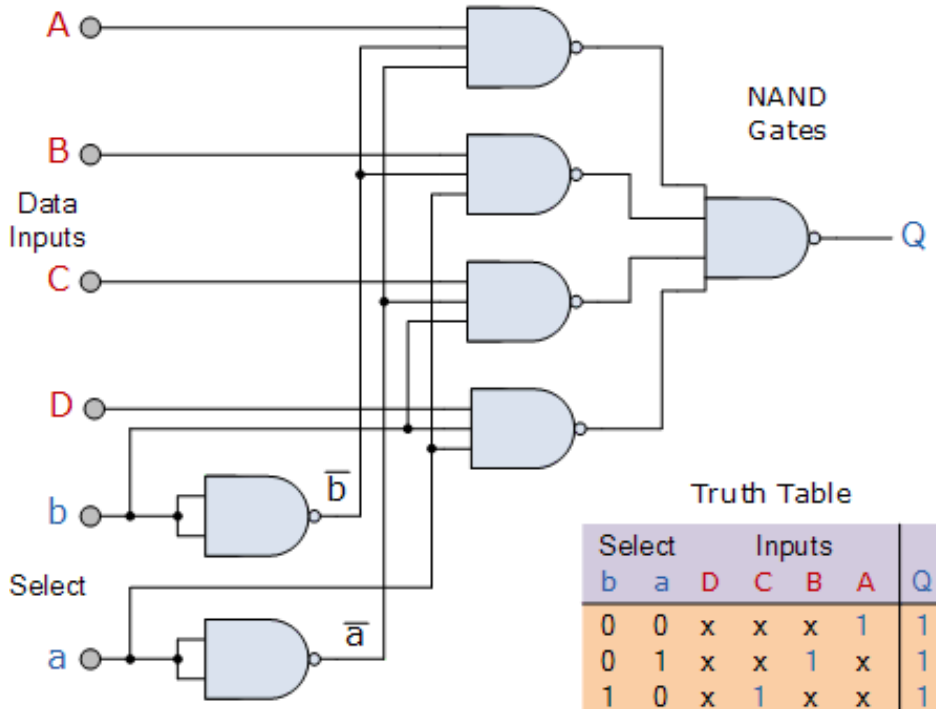

# MULTIPLEXER (MUX, MPX, DATA SELECTOR)

"is a combinational logic circuit designed to switch one of several input lines through to a single common output line by the application of a control signal"

# Multiplexer

- What are the applications of a multiplexer?
- Briefly describe the operation of a multiplexer.
- How does an encoder differ from a multiplexer?
- Give the logic diagram of a 4-to-2 multiplexer.

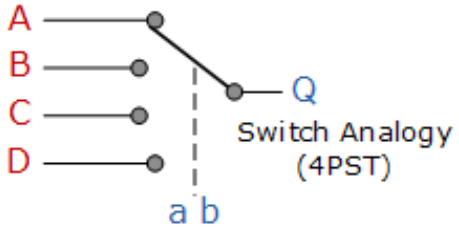
# Multiplexer 4-to-1

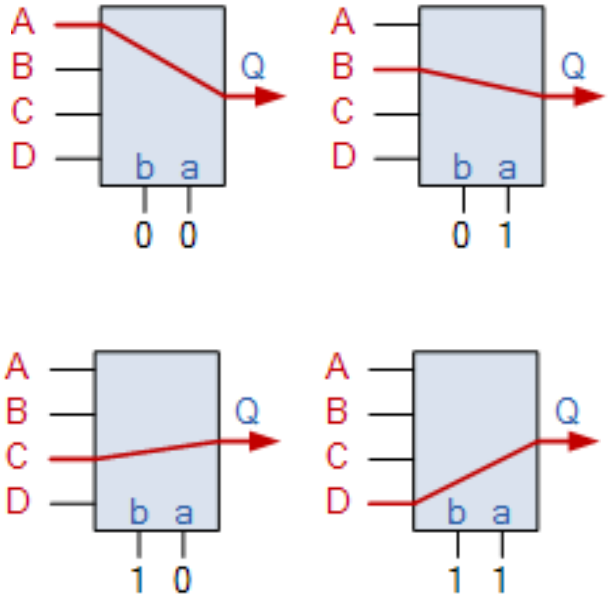



Truth Table

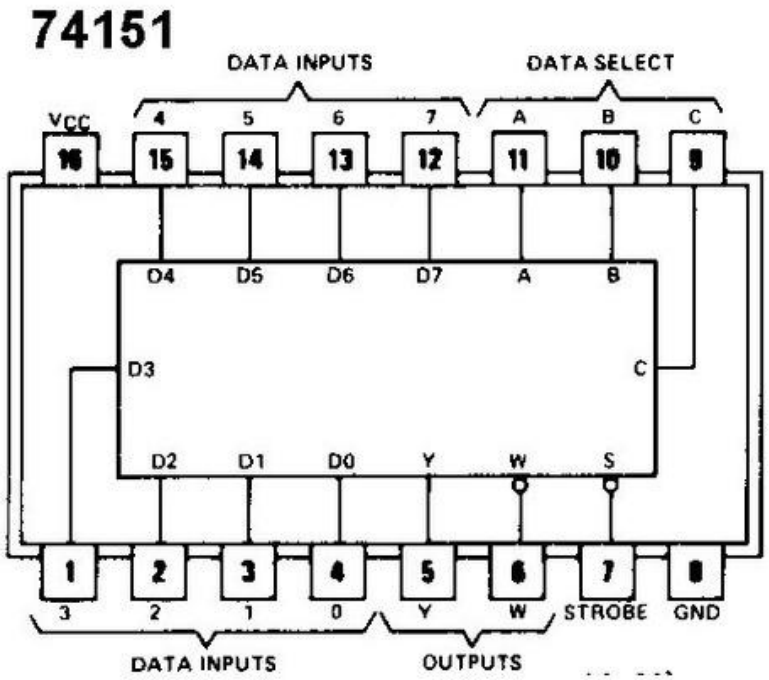
| Select | | Inputs | | | | |
|---|---|---|---|---|---|---|
| b | a | D | C | B | A | Q |
| 0 | 0 | x | x | x | 1 | 1 |
| 0 | 1 | x | x | 1 | x | 1 |
| 1 | 0 | x | 1 | x | x | 1 |
| 1 | 1 | 1 | x | x | x | 1 |

$$Q = \bar{a}\bar{b}A + a\bar{b}B + \bar{a}bC + abD$$

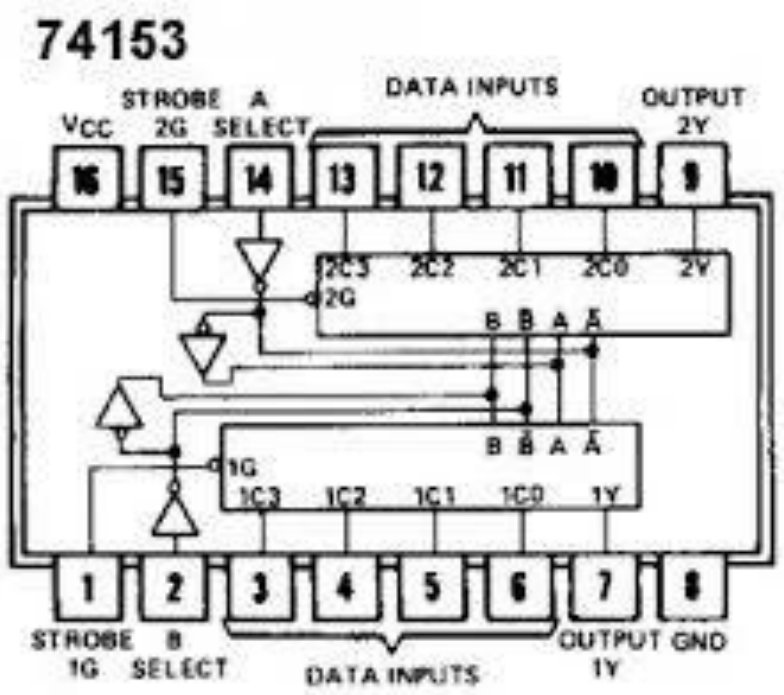


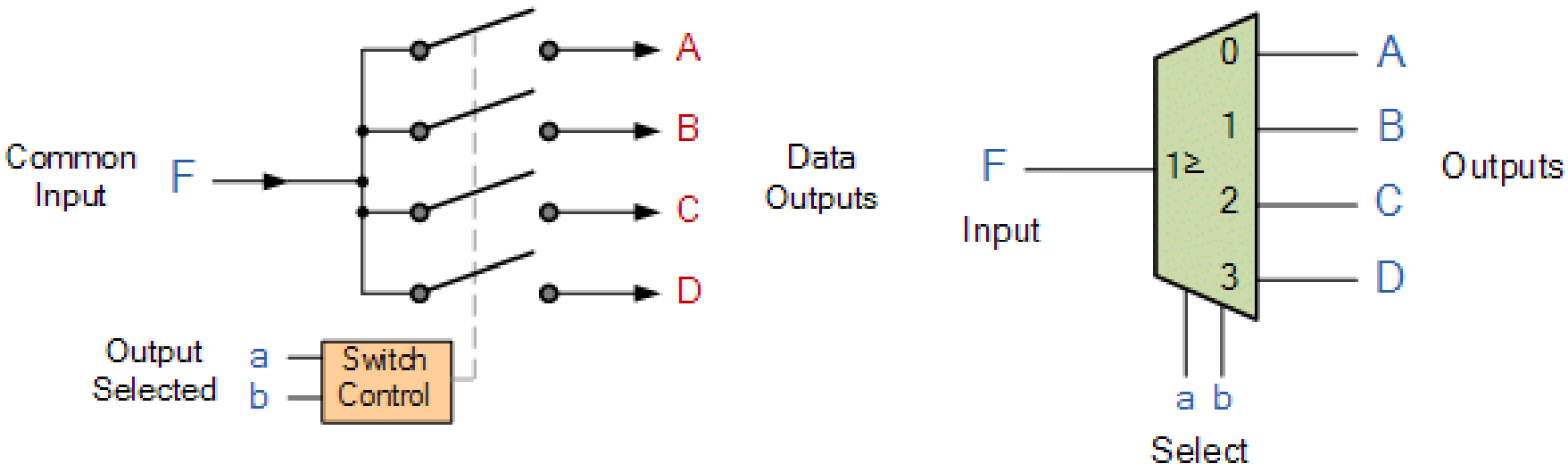
# Multiplexer ICs




8-to-1 MUX




Dual 4-to-1 MUX

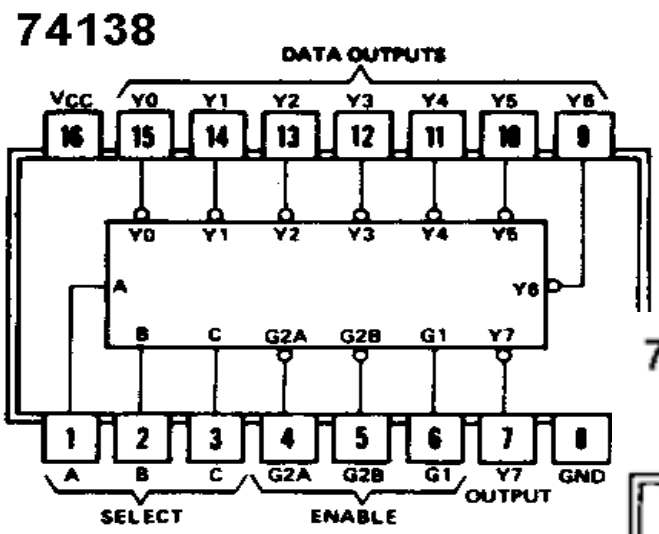



# DEMULTIPLEXER (DEMUX, DATA DISTRIBUTOR)

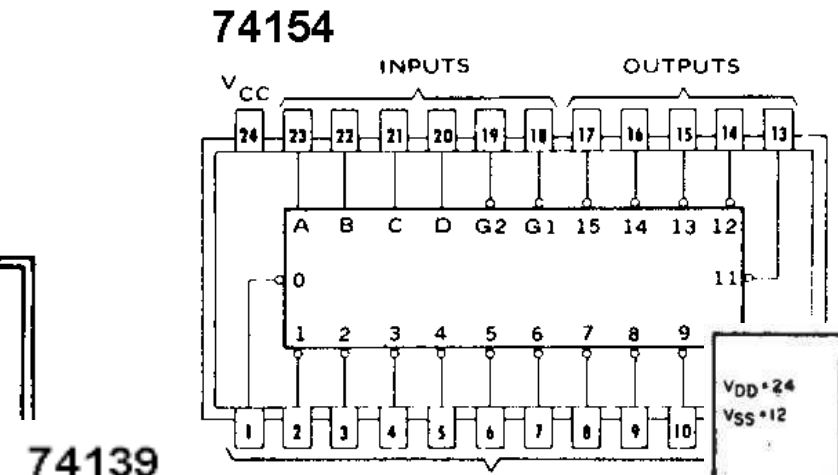
Exact opposite of the multiplexer

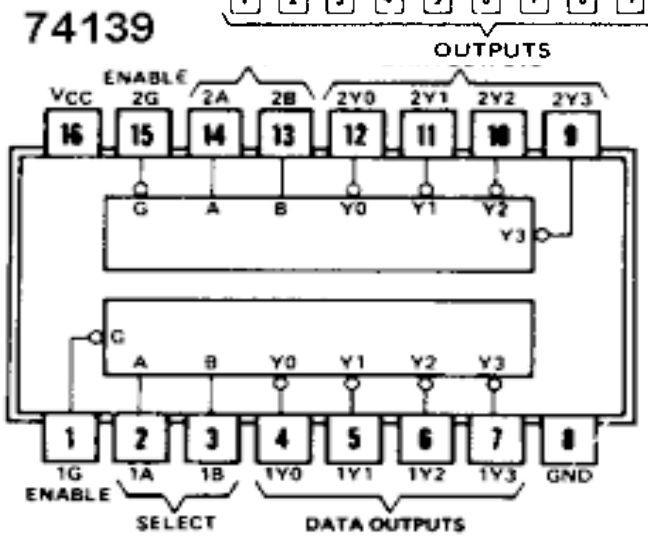
"takes one single input data line and then switches it to any one of a number of individual output lines one at a time"

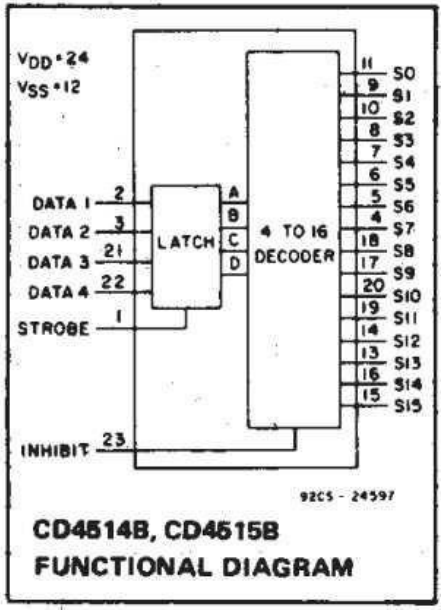
# Demultiplexer ICs




1-to-8 DeMux

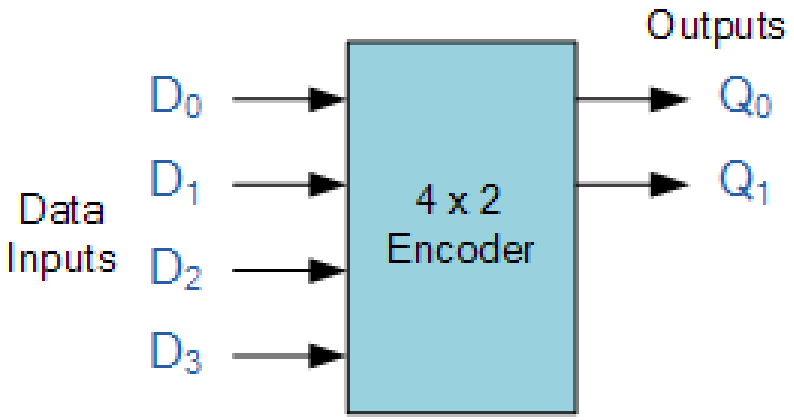



**Demultiplexers or decoders?**

Dual 1-to-4 DeMux




| Inputs | | | | Outputs | |
|---|---|---|---|---|---|
| $D_3$ | $D_2$ | $D_1$ | $D_0$ | $Q_1$ | $Q_0$ |
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | x | x |

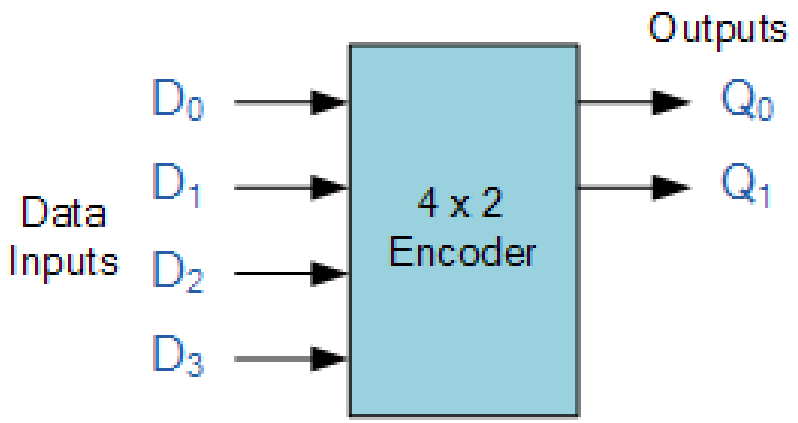
# BINARY ENCODER

"takes all its data inputs one at a time and then converts them into a single encoded output"

"usually an "n-bit" binary encoder has $2^n$ input lines and n-bit output lines"

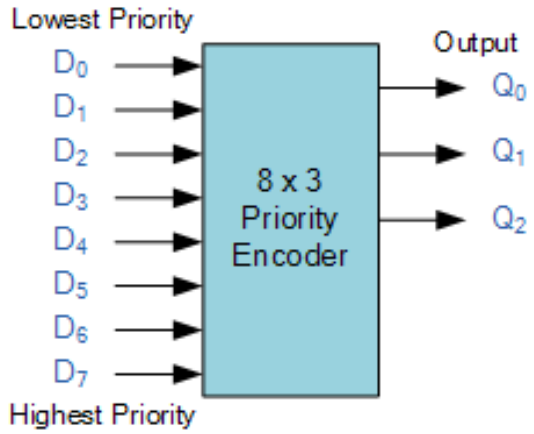
# Priority Encoder (P-encoder)

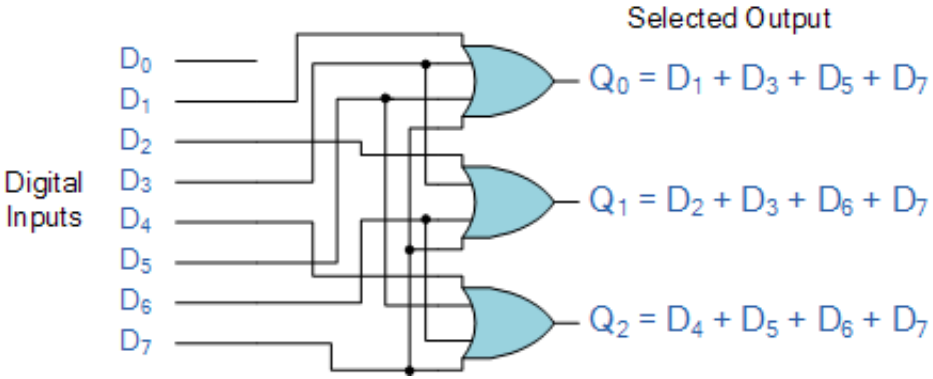
- If there is more than one input at logic level "1" at the same time, the actual output code would only correspond to the input with the highest designated priority




| Inputs | | | | Outputs | |
|---|---|---|---|---|---|
| $D_3$ | $D_2$ | $D_1$ | $D_0$ | $Q_1$ | $Q_0$ |
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | x | x |

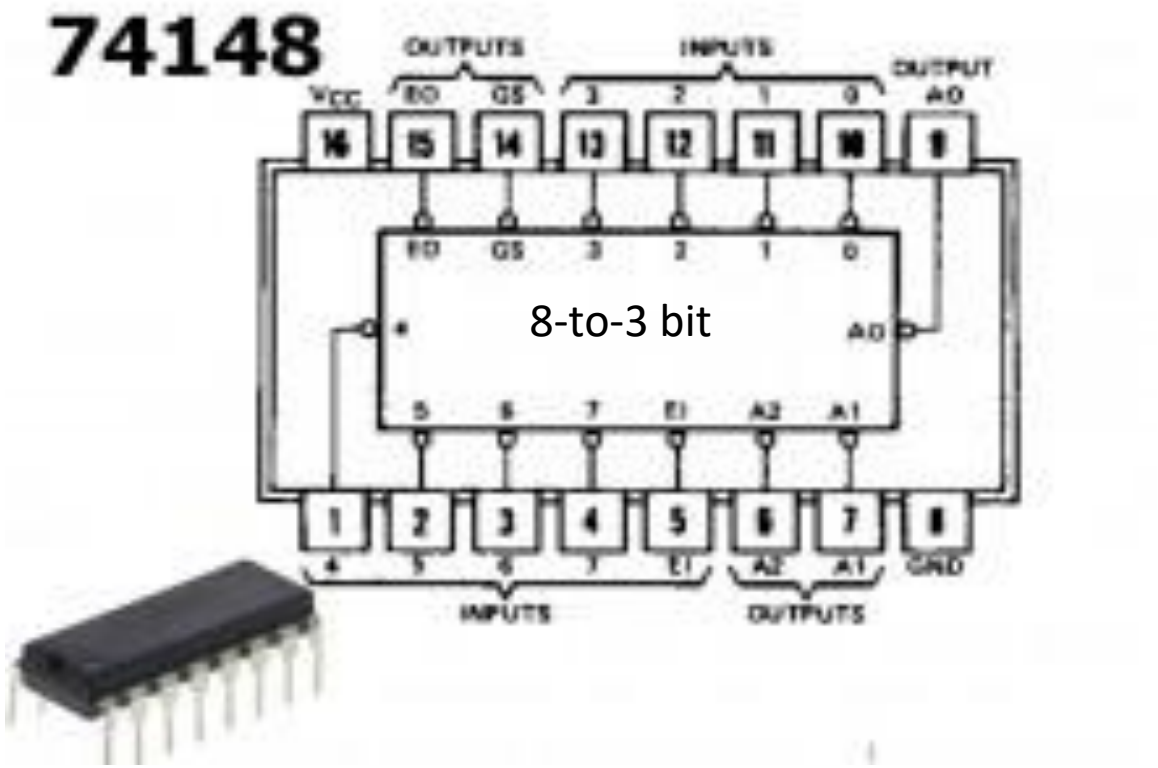
# Priority Encoder



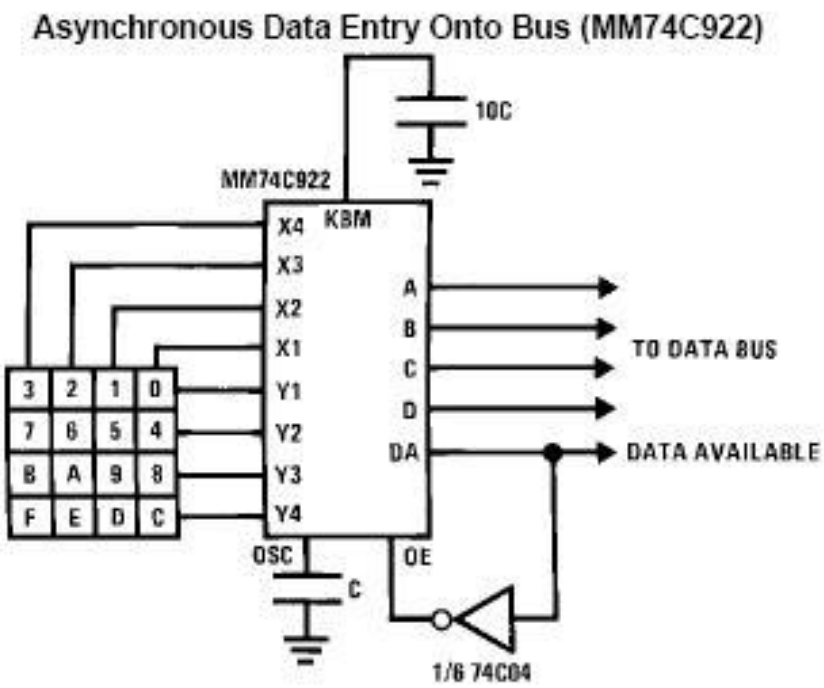



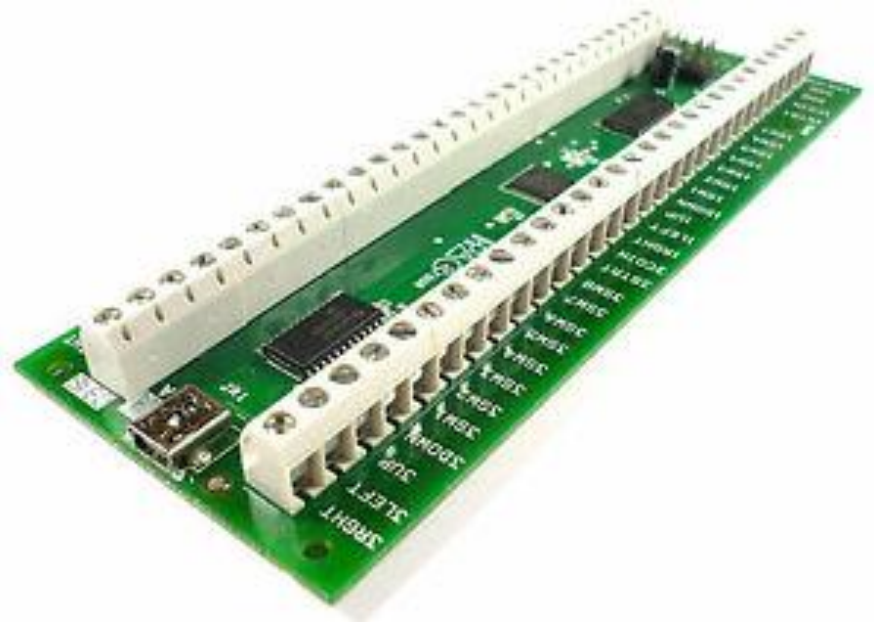

| Digital Inputs | | | | | | | | Binary Output | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | $Q_2$ | $Q_1$ | $Q_0$ |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | **1** | X | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | **1** | X | X | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | **1** | X | X | X | 0 | 1 | 1 |
| 0 | 0 | 0 | **1** | X | X | X | X | 1 | 0 | 0 |
| 0 | 0 | **1** | Lower priority bits are ignored | | | | | 1 | 0 | 1 |
| 0 | **1** | X | | | | | | 1 | 1 | 0 |
| **1** | X | X | X | X | X | X | X | 1 | 1 | 1 |

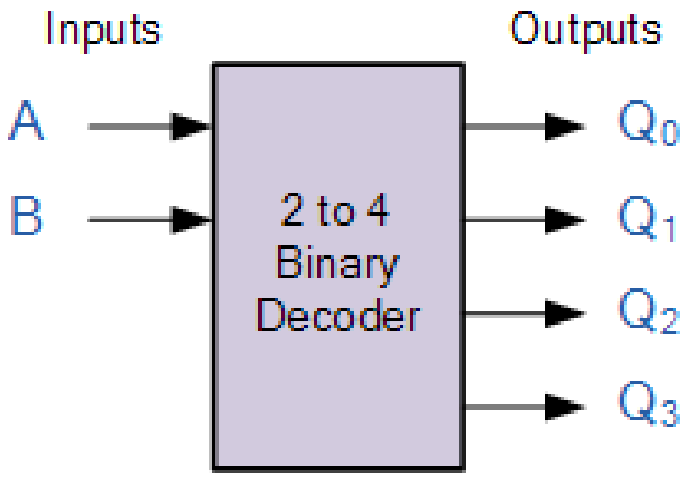
# Priority Encoder ICs



# Application: Keyboard Encoder





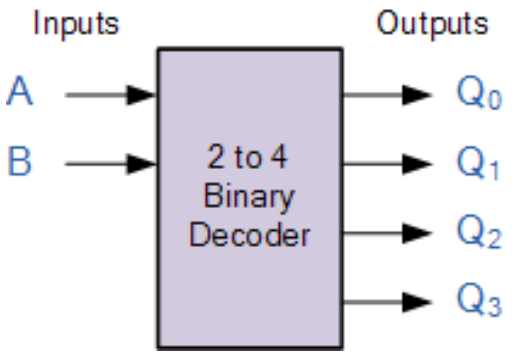

Truth Table

| A | B | $Q_0$ | $Q_1$ | $Q_2$ | $Q_3$ |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 |

# BINARY DECODER

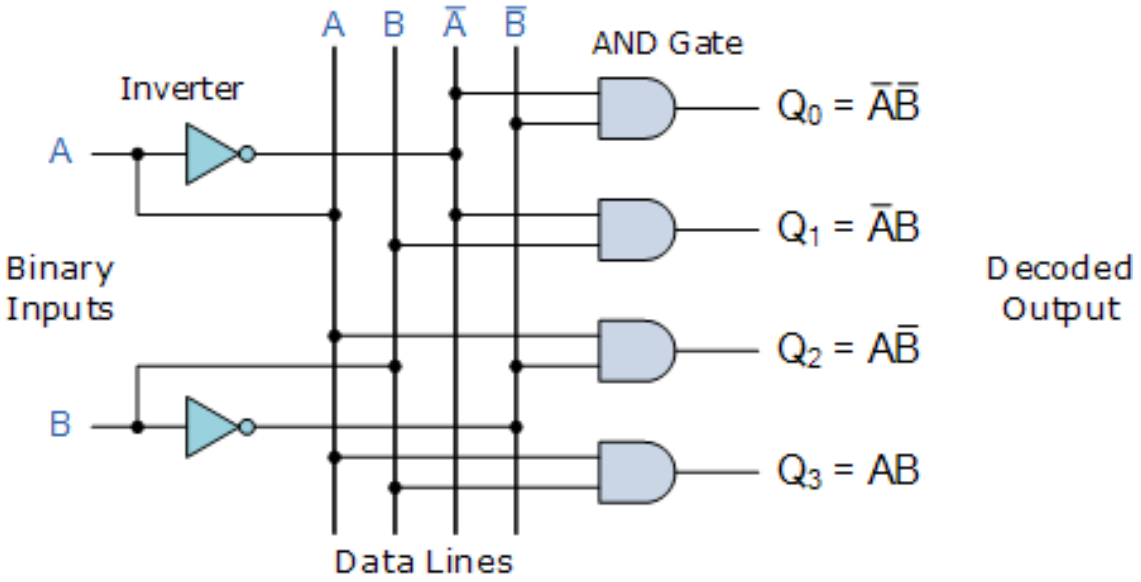
"translate or decode coded information from one format into another, so a binary decoder transforms "n" binary input signals into an equivalent code using $2^n$ outputs"

# Binary Decoder

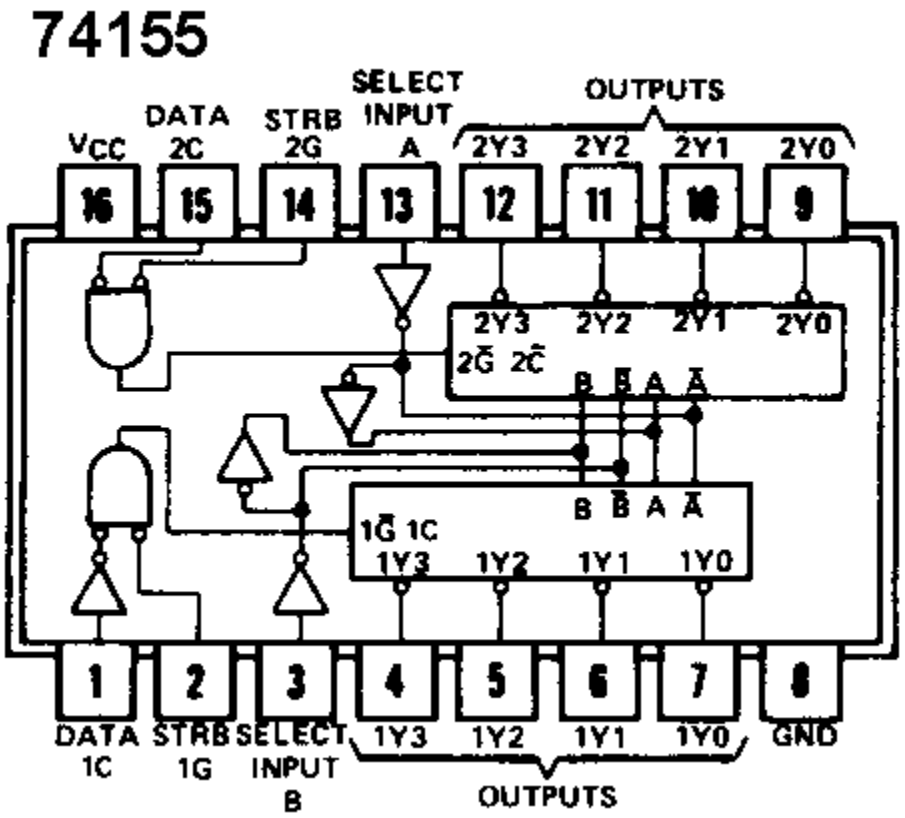



Truth Table

| A | B | $Q_0$ | $Q_1$ | $Q_2$ | $Q_3$ |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 |

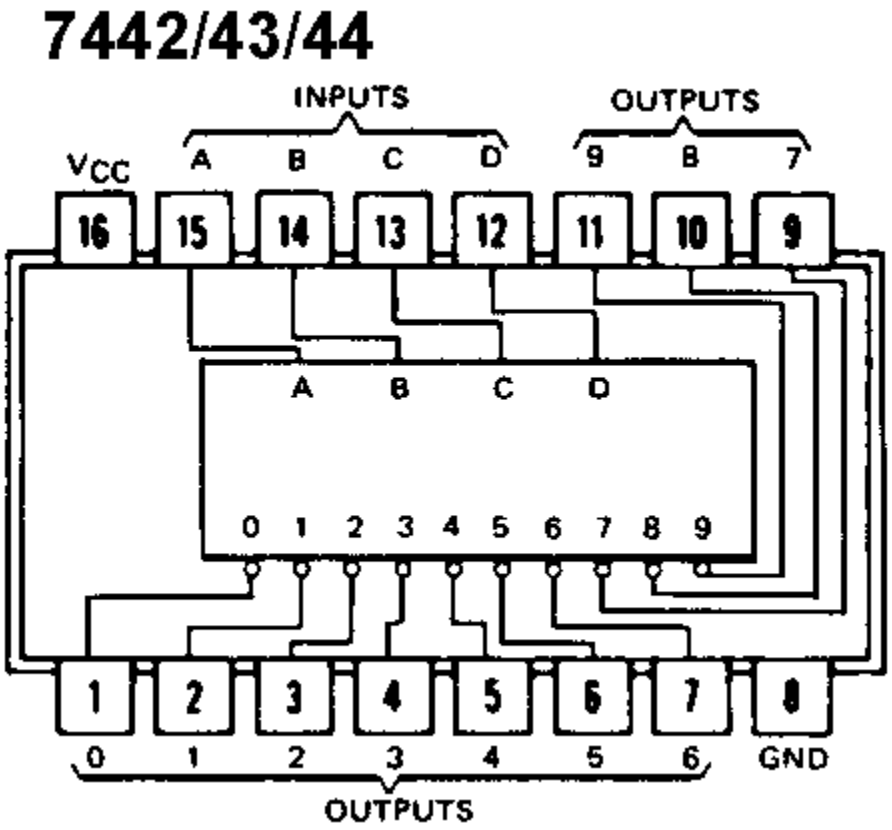


# Binary Decoder ICs

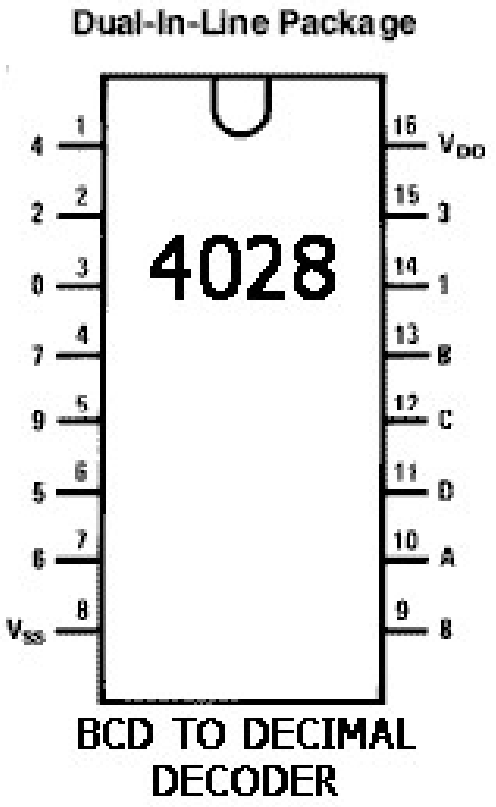



Dual 2-to-4 Decoder/Mux

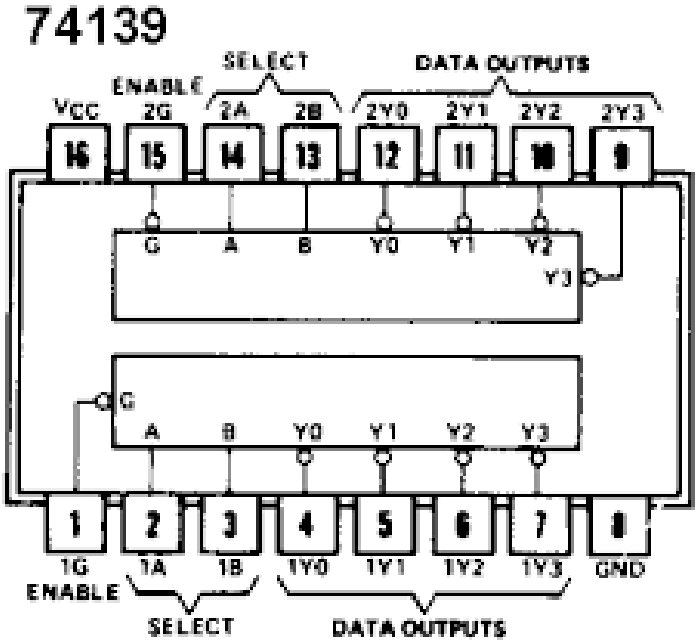
# Binary Decoder ICs

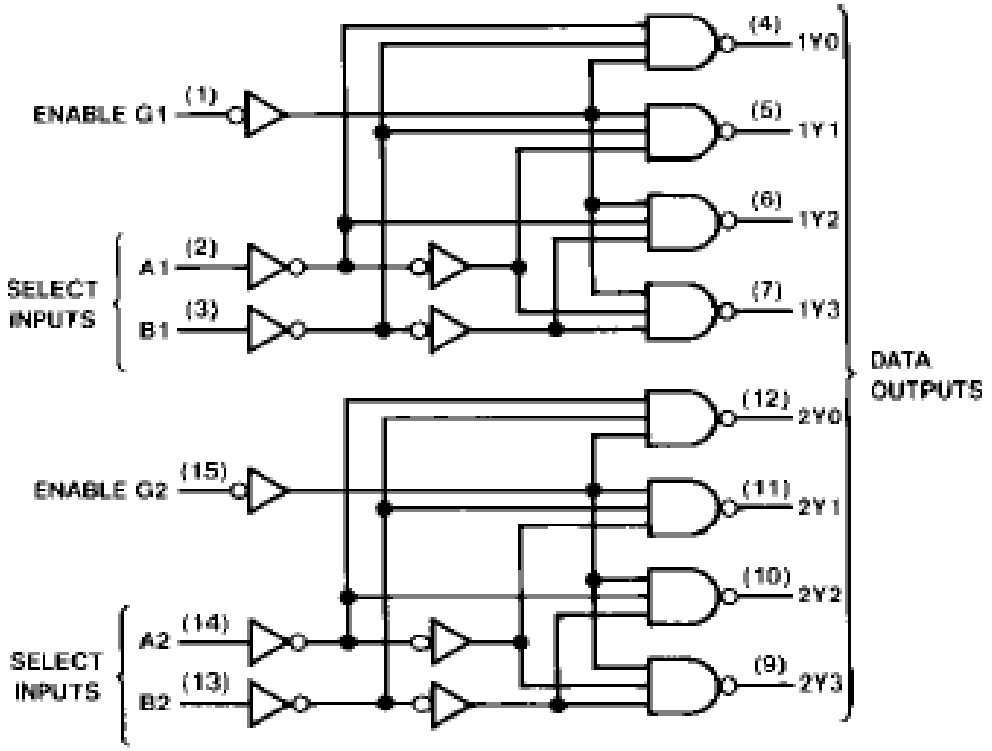



BCD-to-Decimal



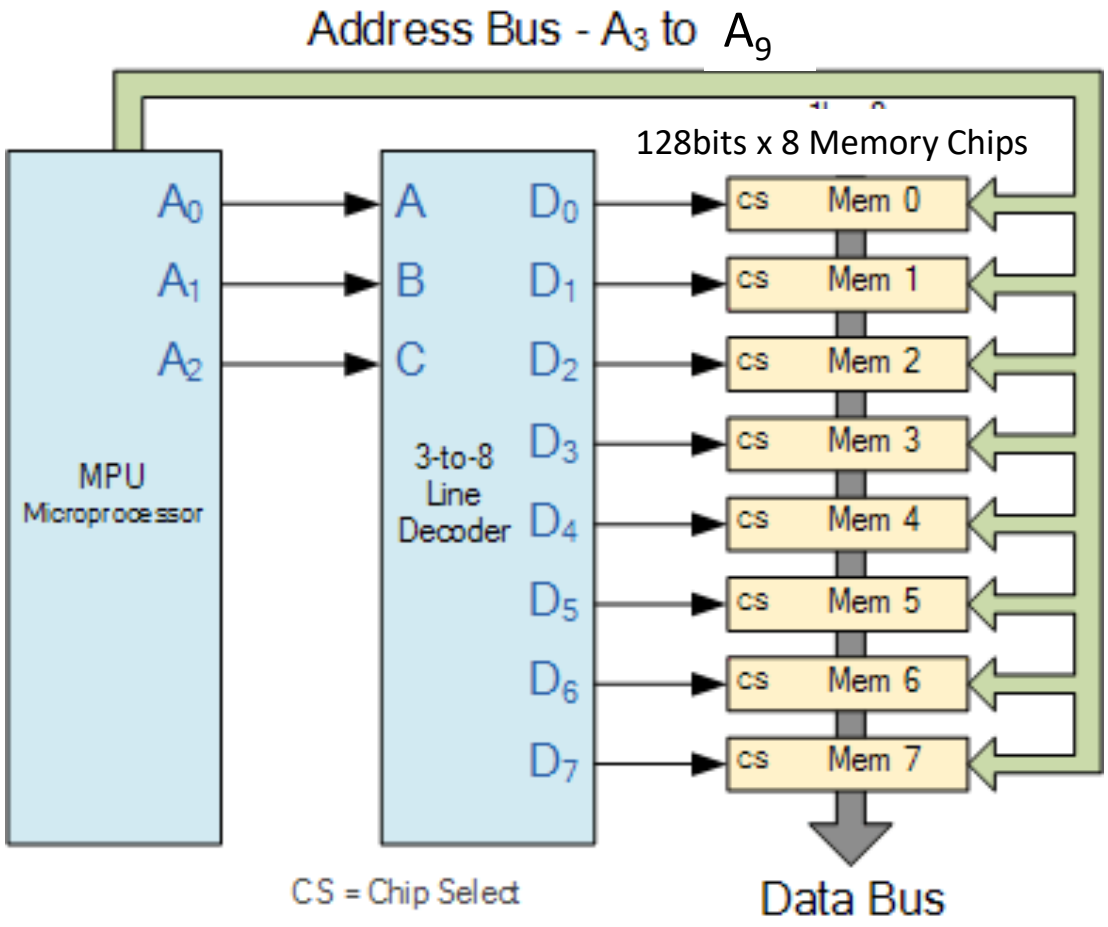
# Testing 74LS139 Dual 2-to-4 Decoder/Demux




| Inputs | | | Outputs | | | |
|---|---|---|---|---|---|---|
| Enable | Select | | | | | |
| G | B | A | Y0 | Y1 | Y2 | Y3 |
| H | X | X | H | H | H | H |
| L | L | L | L | H | H | H |
| L | L | H | H | L | H | H |
| L | H | L | H | H | L | H |
| L | H | H | H | H | H | L |

# Application: Memory Address Decoder

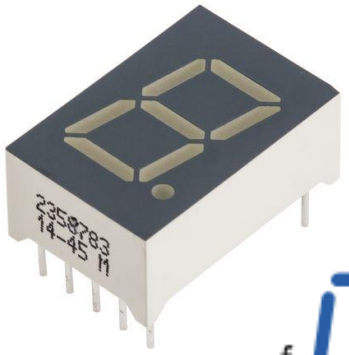


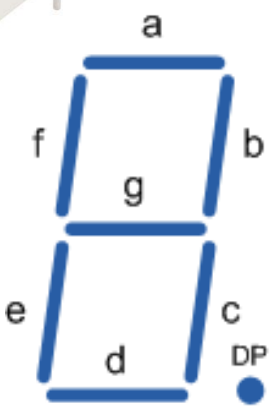

- Which memory chip will be accessed by the following addresses?
  - 1000101001
  - 1100101110
  - 0100101001

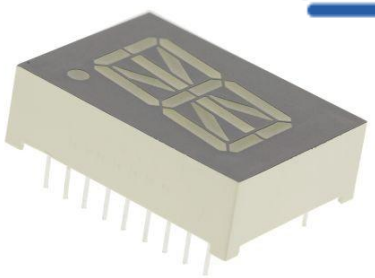
# Application: Display Decoder

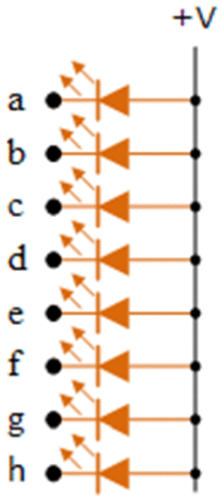
- What is a display decoder?
- What is an SSD?
- How a display decoder is used to drive an SSD?
- What are the advantages of display drivers?
- How do you construct a display decoder using basic logic gates?

# Seven-Segment Display (SSD) Unit



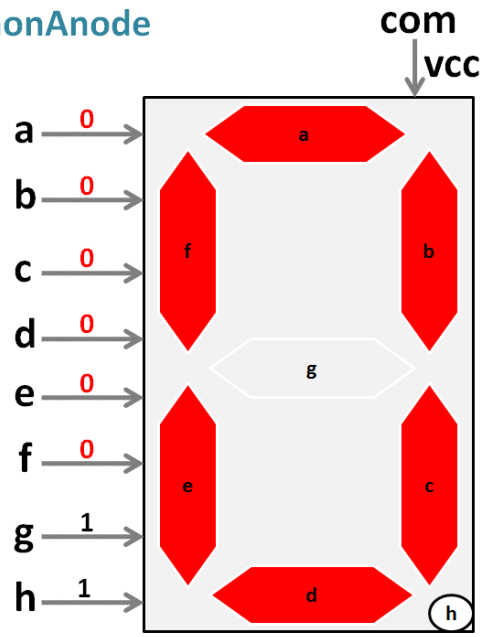


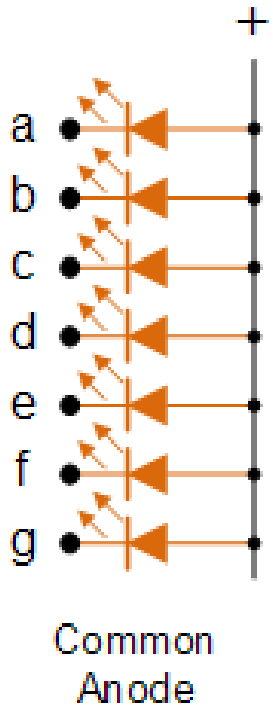





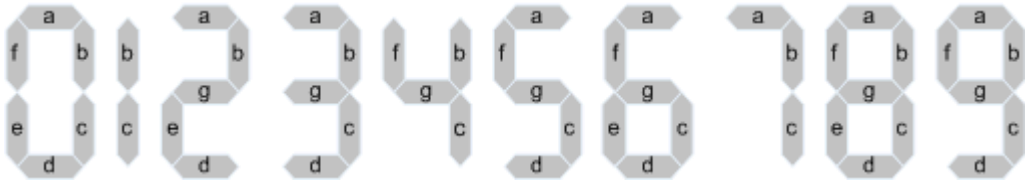

ExploreEmbedded

CommonAnode

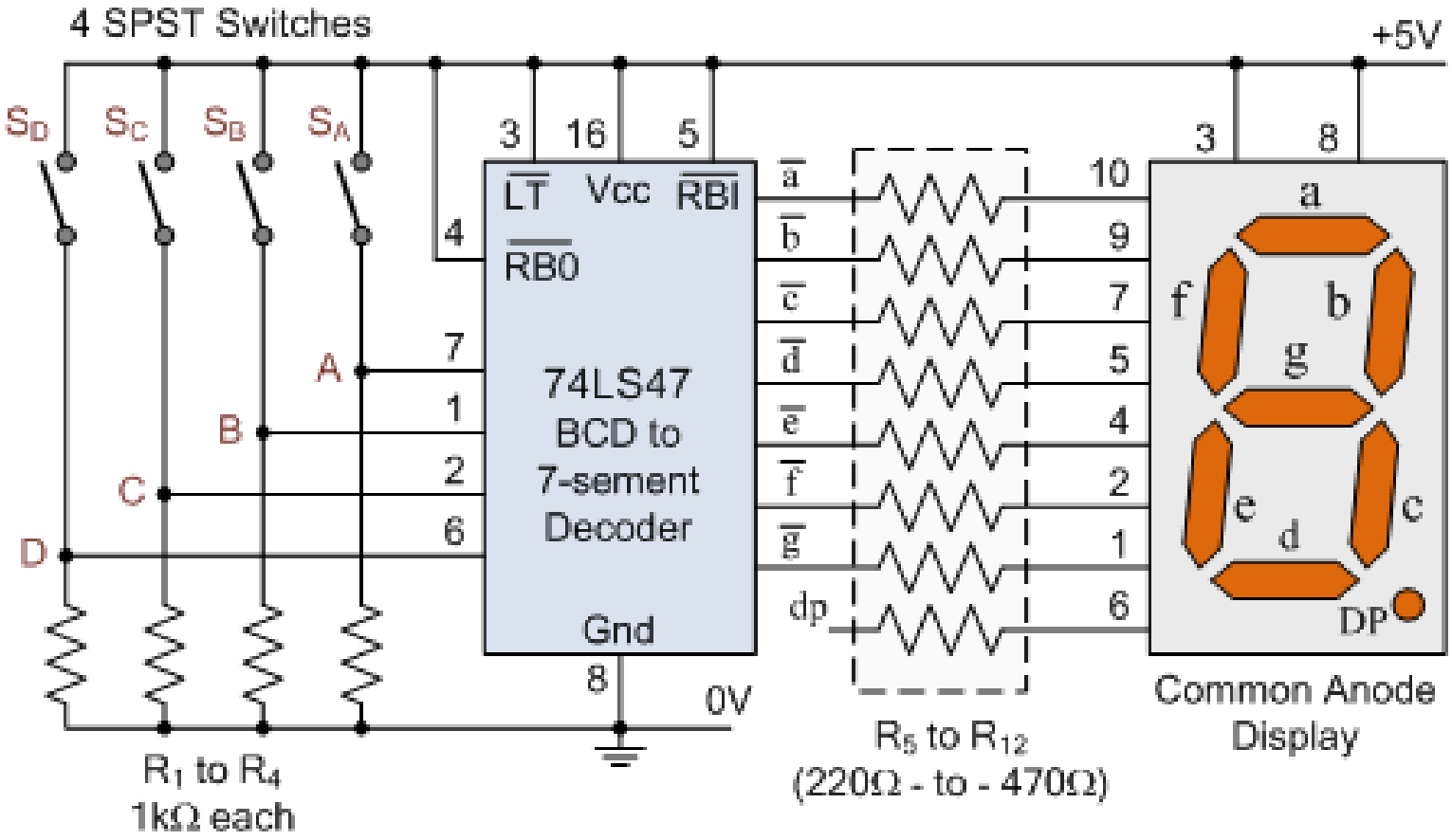


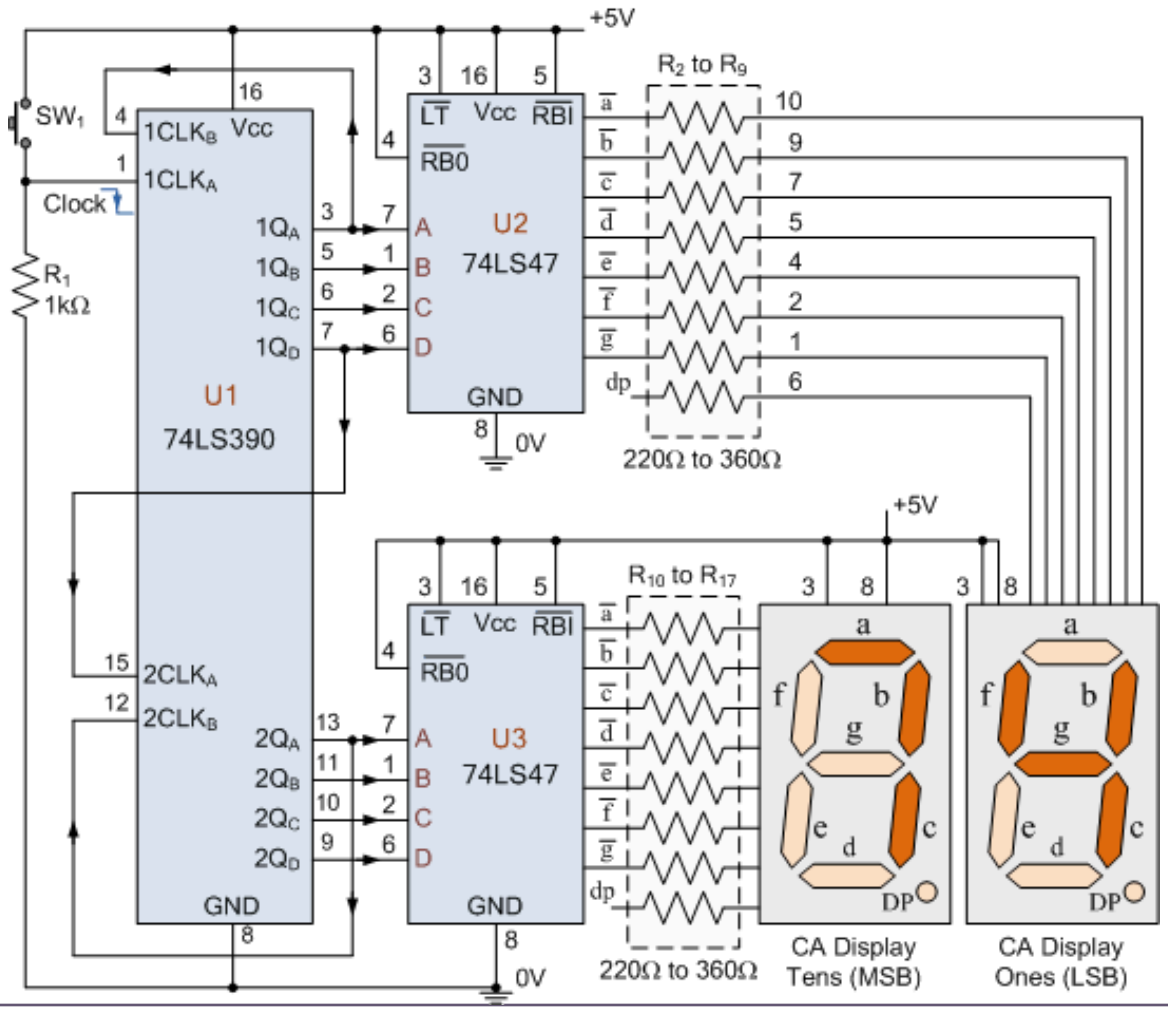

Common Anode

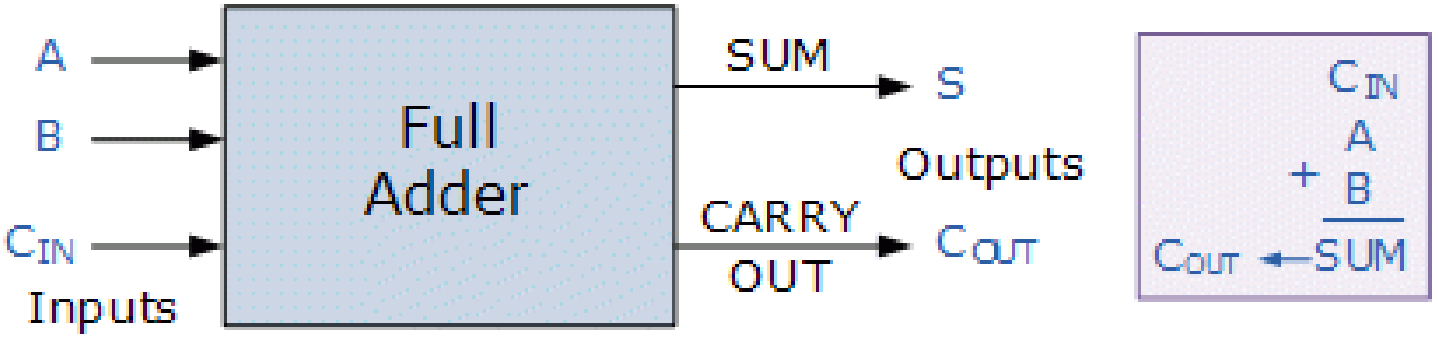


# BCD-to-SSD Decoder



# Counting Circuit





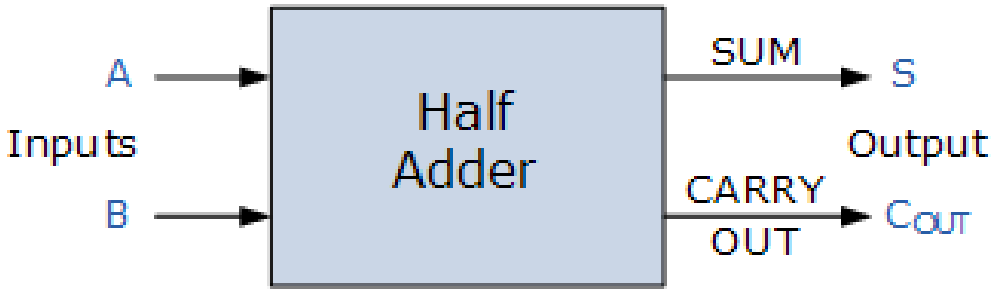

# BINARY ADDER

# Binary Adder

- How the binary addition differs from denary (base 10) addition?
- How do you implement a binary adder using logic gates?
- What is the difference between a half-adder and a full-adder?
- What are the disadvantages of ripple adders?

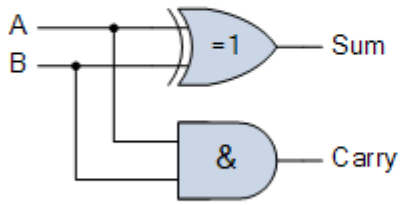
# Binary Addition: Half-Adder

| | | |
|---|---|---|
| 123 | A | |
| + 789 | B | (Addend) |
| 912 | SUM | |




$$\begin{array}{r} A \\ +\ B \\ \hline C_{OUT} \leftarrow SUM \end{array}$$

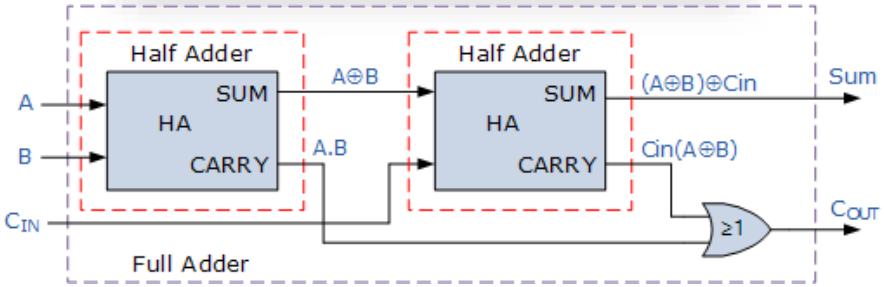
| | | | |
|---|---|---|---|
| 0 | 0 | 1 | 1 |
| +0 | +1 | +0 | +1 |
| 0 | 1 | 1 (carry) | 1←0 |

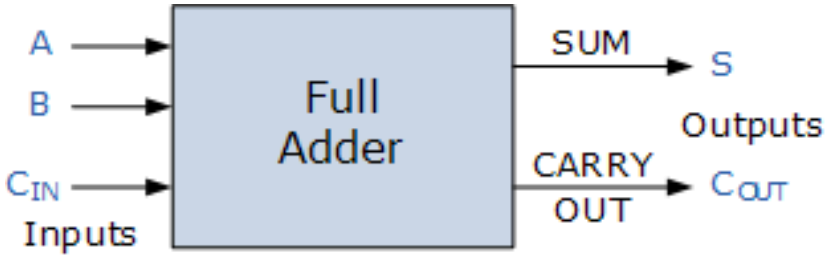
| Symbol | Truth Table | | | |
|---|---|---|---|---|
|  | B | A | SUM | CARRY |
| | 0 | 0 | 0 | 0 |
| | 0 | 1 | 1 | 0 |
| | 1 | 0 | 1 | 0 |
| | 1 | 1 | 0 | 1 |

# Binary Addition: Full-Adder

| 123 | A | |
|---|---|---|
| + 789 | B | (Addend) |
| 912 | SUM | |

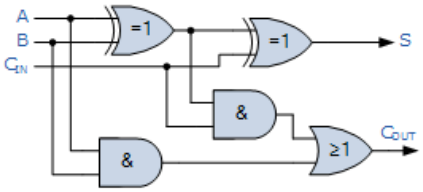
| 0 | 0 | 1 | 1 |
|---|---|---|---|
| + 0 | + 1 | + 0 | + 1 |
| 0 | 1 | 1 (carry) | 1←0 |



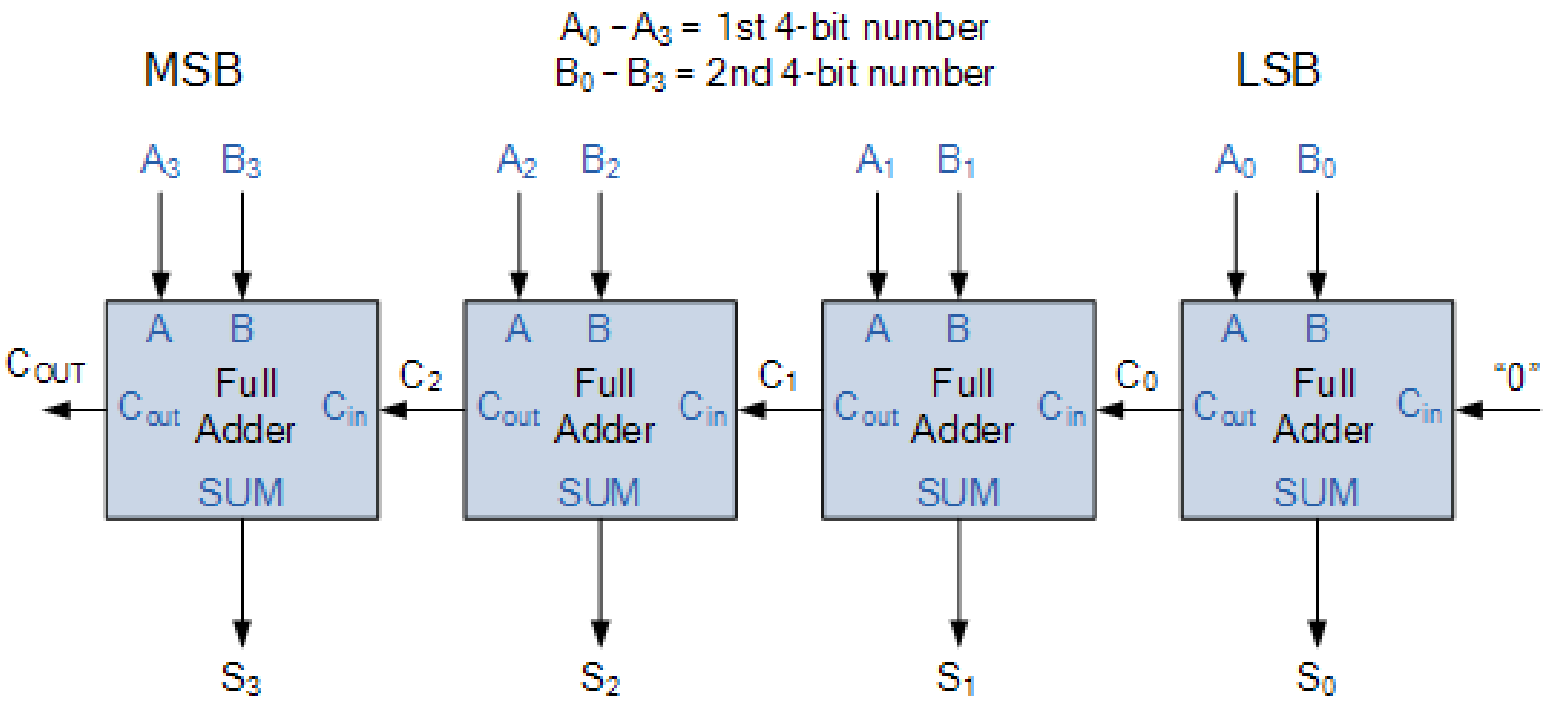




$C_{IN}$
A
\+ B
$C_{OUT}$ ←SUM

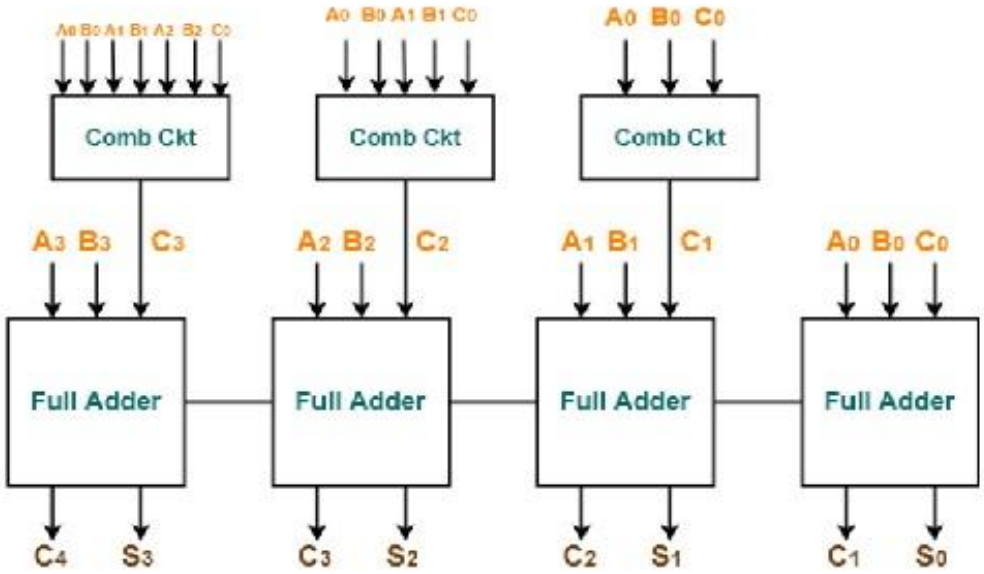
| Symbol | Truth Table | | | | |
|---|---|---|---|---|---|
| | C-in | B | A | Sum | C-out |
| | 0 | 0 | 0 | 0 | 0 |
| | 0 | 0 | 1 | 1 | 0 |
| | 0 | 1 | 0 | 1 | 0 |
| | 0 | 1 | 1 | 0 | 1 |
| | 1 | 0 | 0 | 1 | 0 |
| | 1 | 0 | 1 | 0 | 1 |
| | 1 | 1 | 0 | 0 | 1 |
| | 1 | 1 | 1 | 1 | 1 |

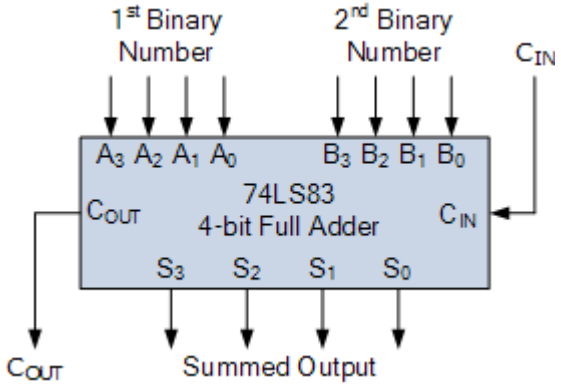


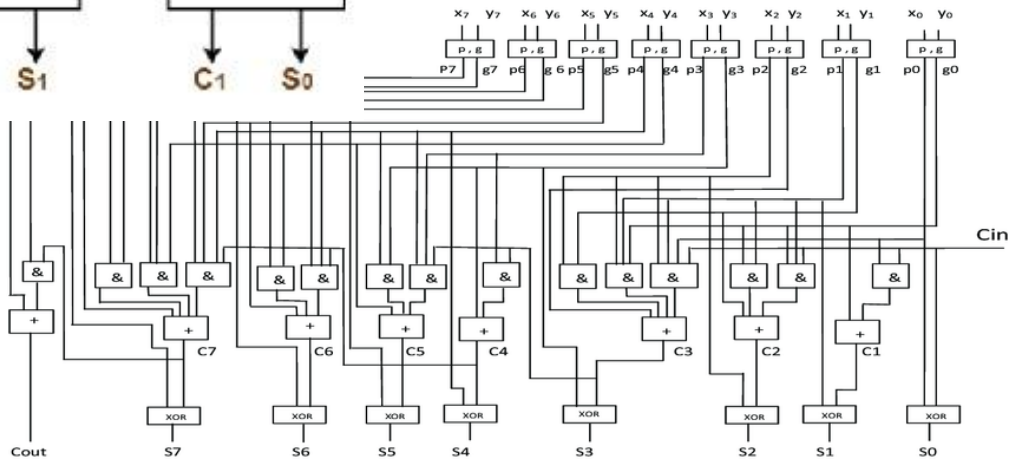
# 4-Bit Ripple Carry Adder

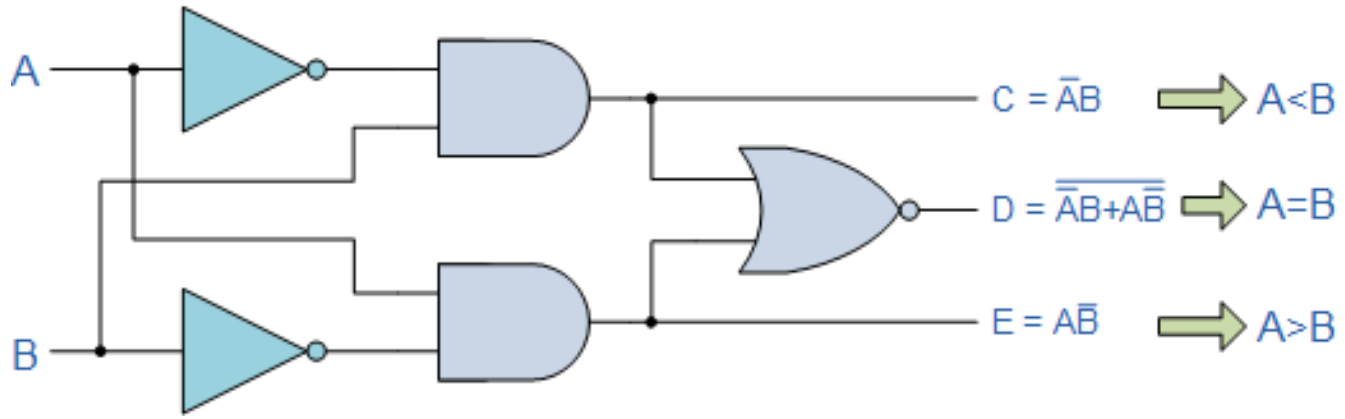


# Hierarchical Carry Look Ahead Binary Adders

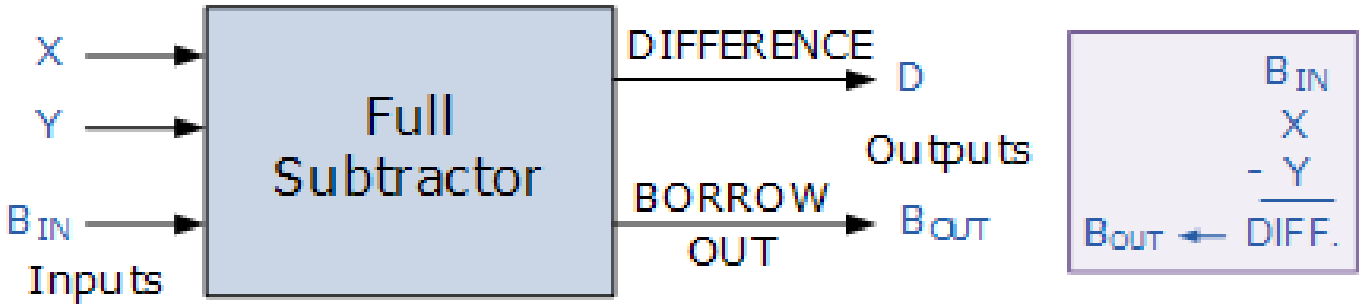





# DIGITAL COMPARATOR




# BINARY SUBTRACTOR

# APPLICATION: ALU

# How Computers Calculate

https://www.youtube.com/watch?v=1I5ZMmrOfnA&feature=emb_logo

# Arithmetic & Logic Unit