# UNIT II: Combinational Logic

**FACULTY**

**Dr. K. ARTHI MCA, M.Phil., Ph.D.,**
**Assistant Professor,**
**Postgraduate Department of Computer Applications,**
**Government Arts College (Autonomous),**
**Coimbatore-641018.**

# Introduction to ALU

- ALU stands for: Arithmetic Logic Unit
- ALU is a digital circuit that performs Arithmetic (Add, Sub, . . .) and Logical (AND, OR, NOT) operations.
- John Von Neumann proposed the ALU in 1945 when he was working on EDVAC.

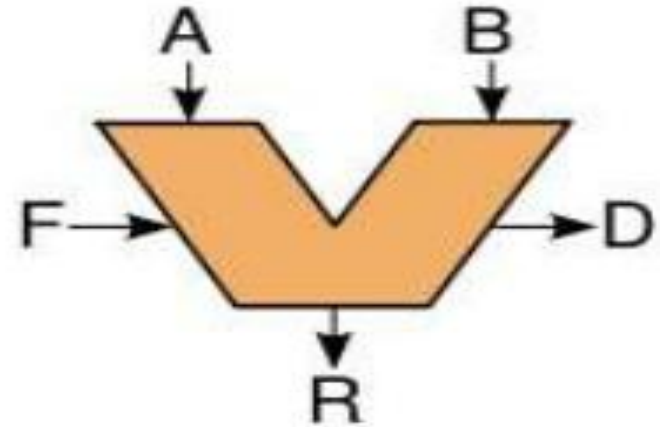
# Introduction to ALU (contd...)

- An ALU is the fundamental unit of any computing system.
- Understanding how an ALU is designed and how it works is essential to building any advanced logic circuits.
- Using this knowledge and experience, we can move on to designing more complex integrated circuits.
- The ALU is the "heart" of a processor—you could say that everything else in the CPU is there to support the ALU.

# Typical Schematic Symbol of an ALU

**A and B:** the inputs to the ALU (aka operands)
**R:** Output or Result
**F:** Code or Instruction from the Control Unit (aka as op-code)
**D:** Output status; it indicates cases such as:

- carry-in
- carry-out,
- overflow,
- division-by-zero
- And . . .



# TYPES OF DIGITAL LOGIC CIRCUITS IN ALU

- COMBINATIONAL CIRCUITS
- SEQUENTIAL CIRCUITS

# INTRODUCTION TO COMBINATIONAL CIRCUITS

- Combinational Circuits are made of logic gates.
- Doesn't contain memory element , that's why they cant store any information.
- Value of present output is determined by present input.
- Examples of combinational circuits are half adders, full adders, sub tractors etc.

# Combinational Circuits

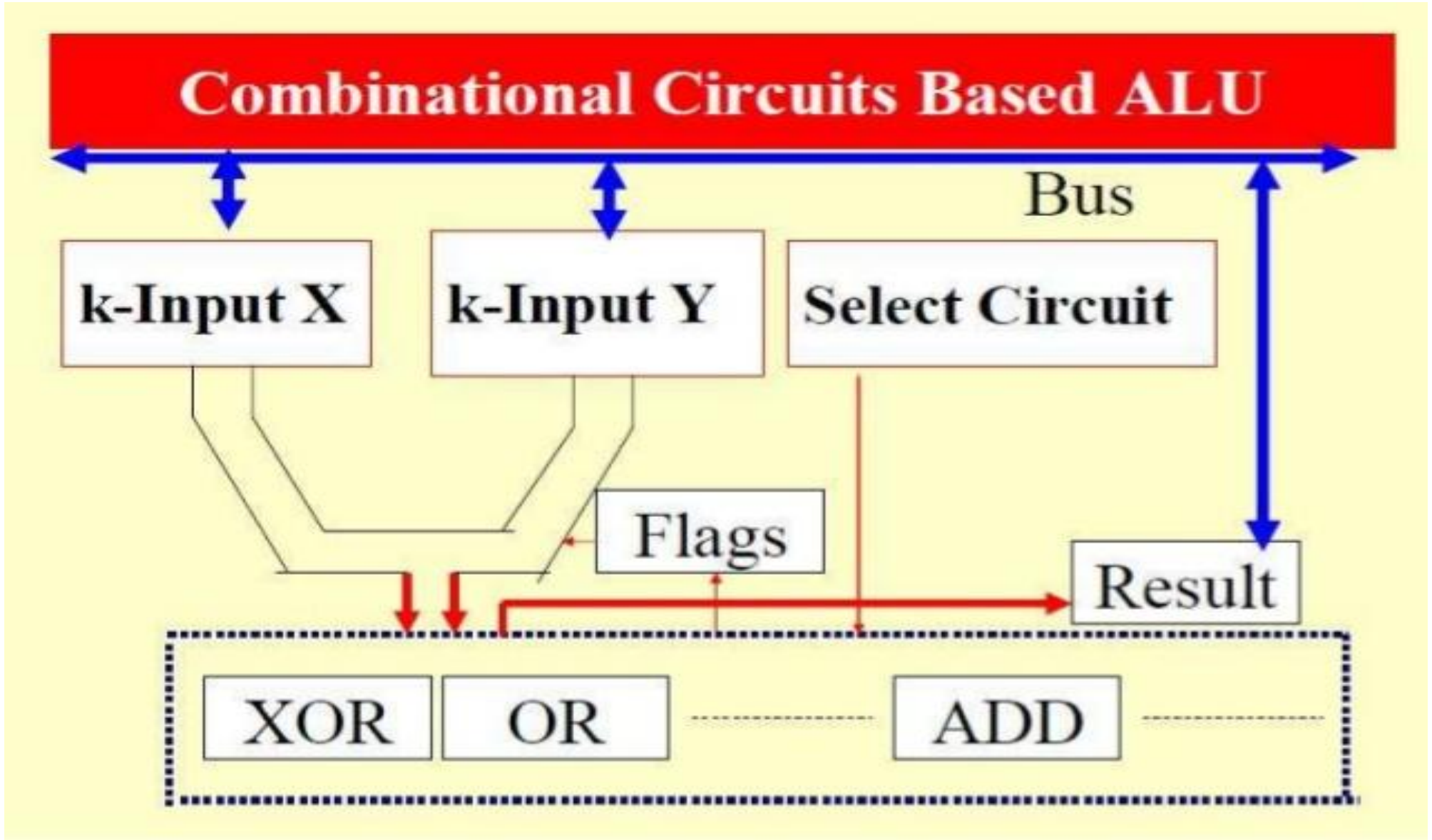

- Binary values of outputs are a function of binary combination of inputs
- Outputs at any given time are entirely dependent on inputs that are present at that time

# Adding 2 Numbers

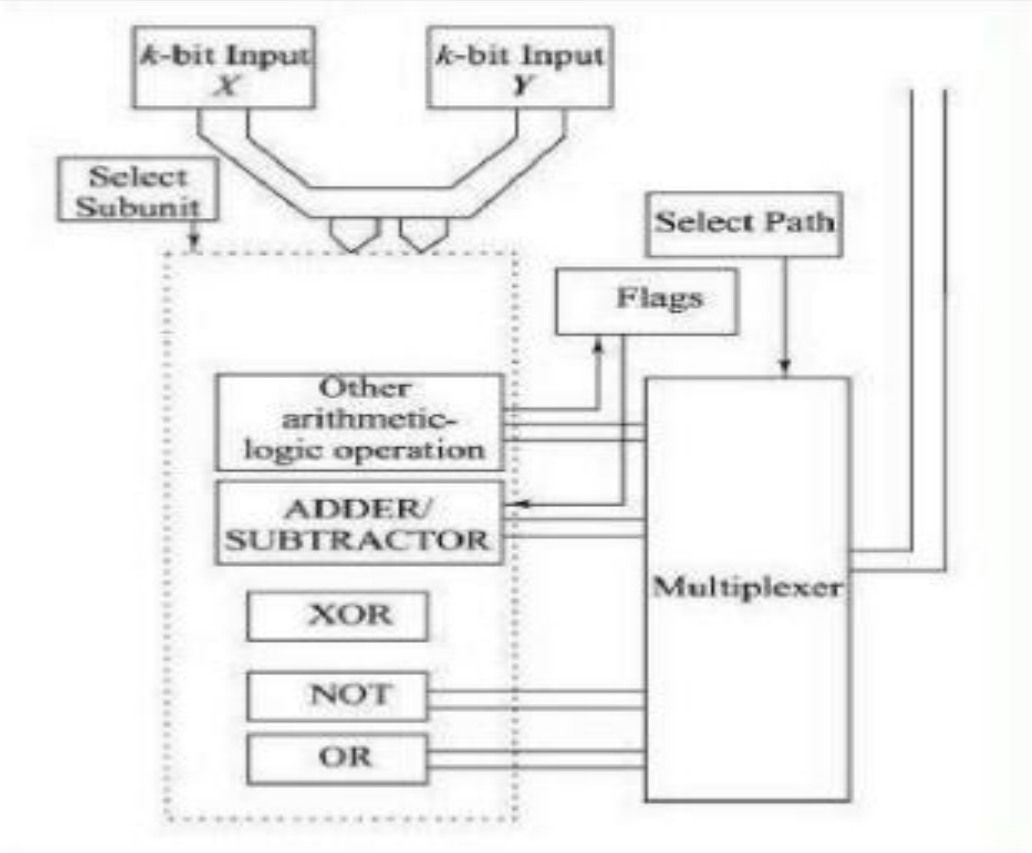
- Write the truth table for addition of 2 bits A & B

| A | B | Sum (S) | Carry (C) |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

- Write Boolean representation for Sum & Carry
  - $S = A'B + AB' = A \oplus B$
  - $C = AB$


Combinational Circuits Based ALU
Bus
k-Input X
k-Input Y
Select Circuit
Flags
Result
XOR
OR
ADD

# AN ALU USING COMBINATIONAL CIRCUITS



# Examples of Combinational Circuits:

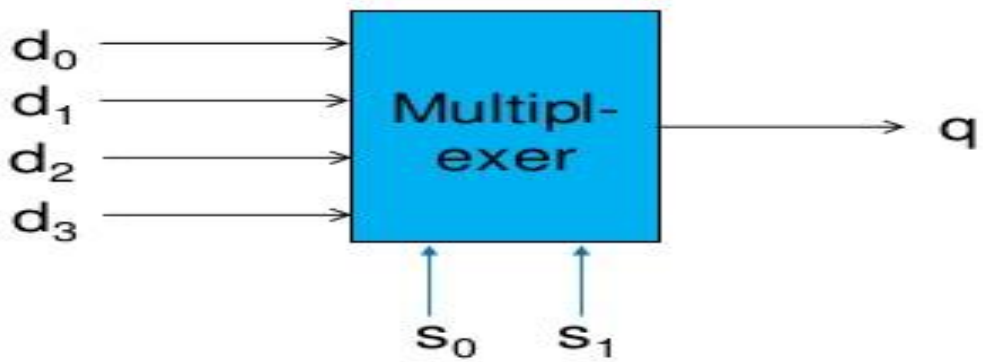
- Multiplexer
- Demultiplexer
- Encoder
- Decoder
- Half Adder
- Full Adder

# Multiplexer & Demultiplexer

- Multiplexer-
  - A multiplexer is a combinational circuit where binary information from one of many input lines is selected and directs it to a single output line.

- Demultiplexer-
  - Demultiplexing is the reverse process of multiplexing; i.e., a demultiplexer is a combinational circuit that receives information on a single line and transmits this information on one of 2n possible output lines.

# Multiplexer




- Receives binary data from $2^n$ lines & connect them to a single output line based on a selection
- By applying a control signals we can steer any input to the output

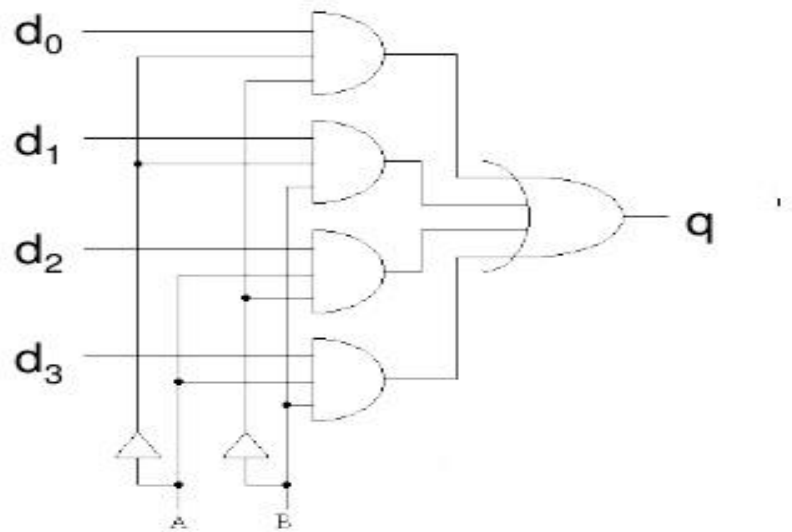
# Multiplexer (Cont.)

- Truth table

| $s_0$ | $s_1$ | q |
|---|---|---|
| 0 | 0 | $d_0$ |
| 0 | 1 | $d_1$ |
| 1 | 0 | $d_2$ |
| 1 | 1 | $d_3$ |

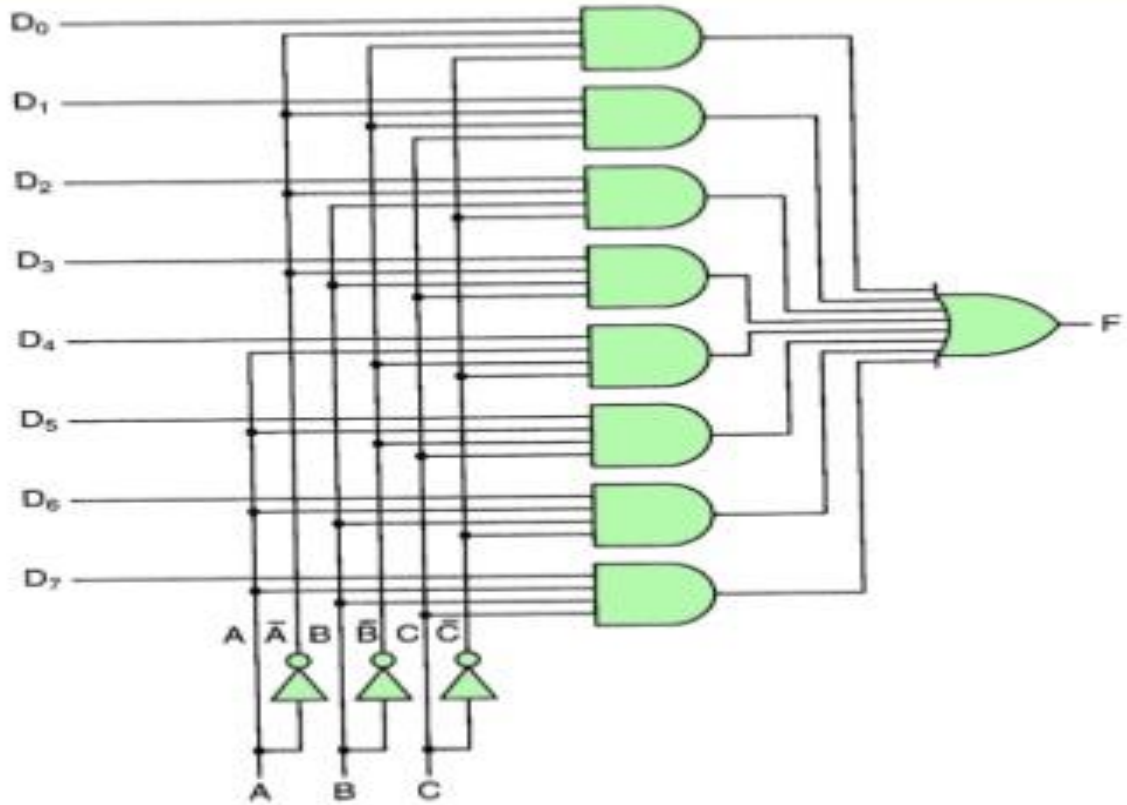
# Multiplexer (Cont.)

- Logic circuit



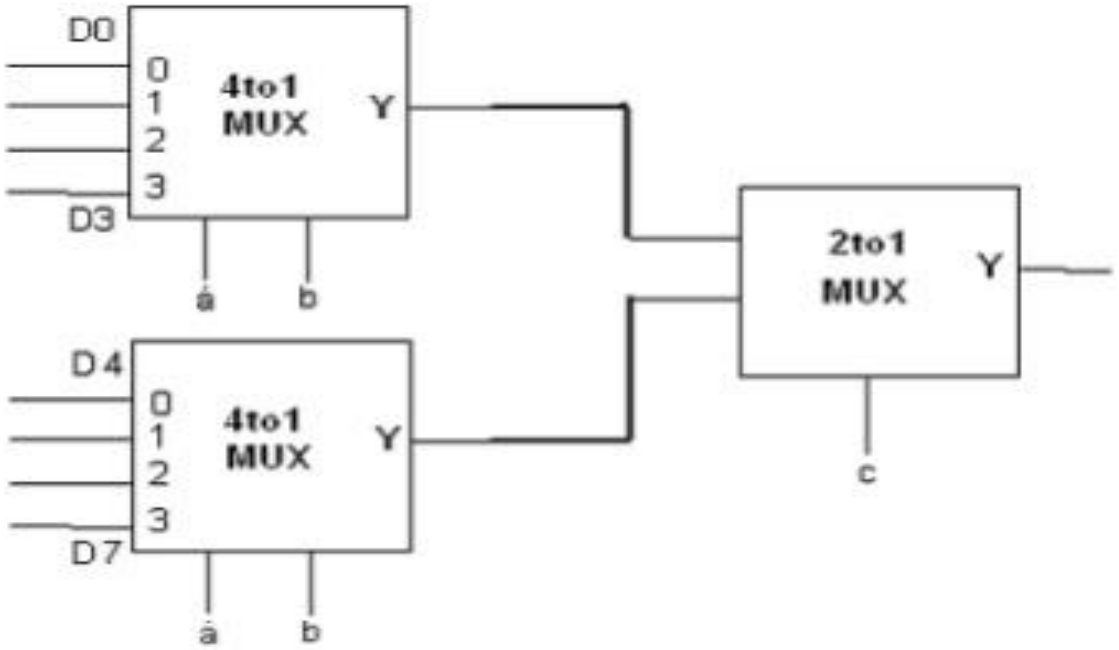

Source: www.ee.surrey.ac.uk/Projects/CAL/digital-logic/multiplexer/index.html

# 8-to-1 Multiplexer



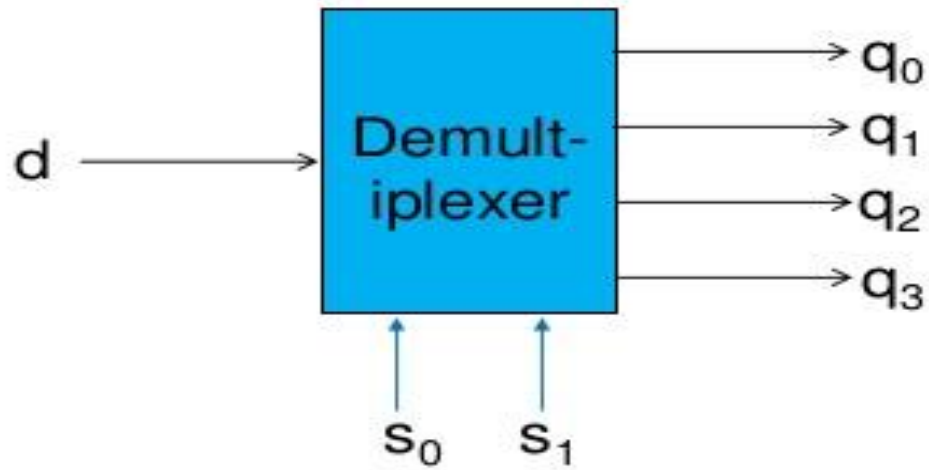

Source: http://users.cis.fiu.edu/~prabakar/cda4101/Common/notes/lecture08.html

# 8-to-1 Multiplexer using 4-to-1 & 2-to-1 Multiplexers




Source: www.exploreroots.com/dc28.html

# Demultiplexer



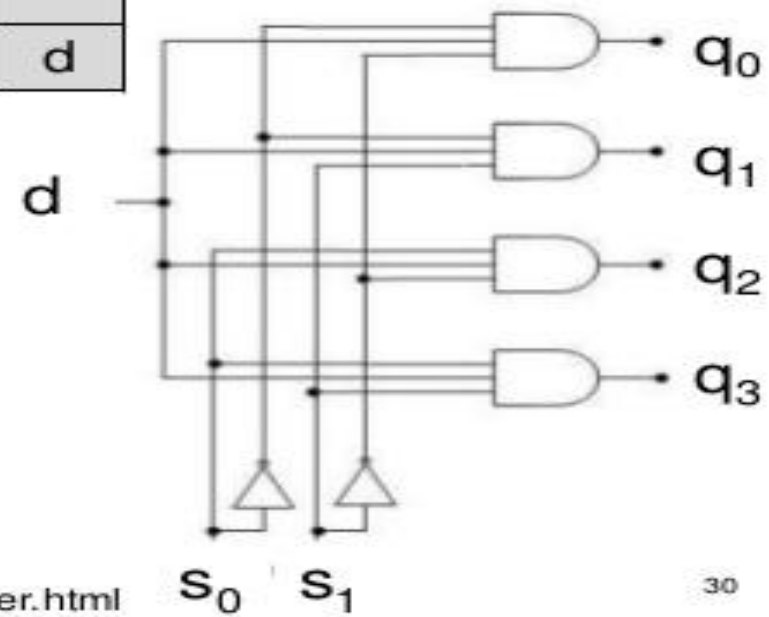

- Reverse process of a multiplexer
- By applying a control signals we can steer the input signal to one of the output lines

# Demultiplexer (Cont.)

- Truth table

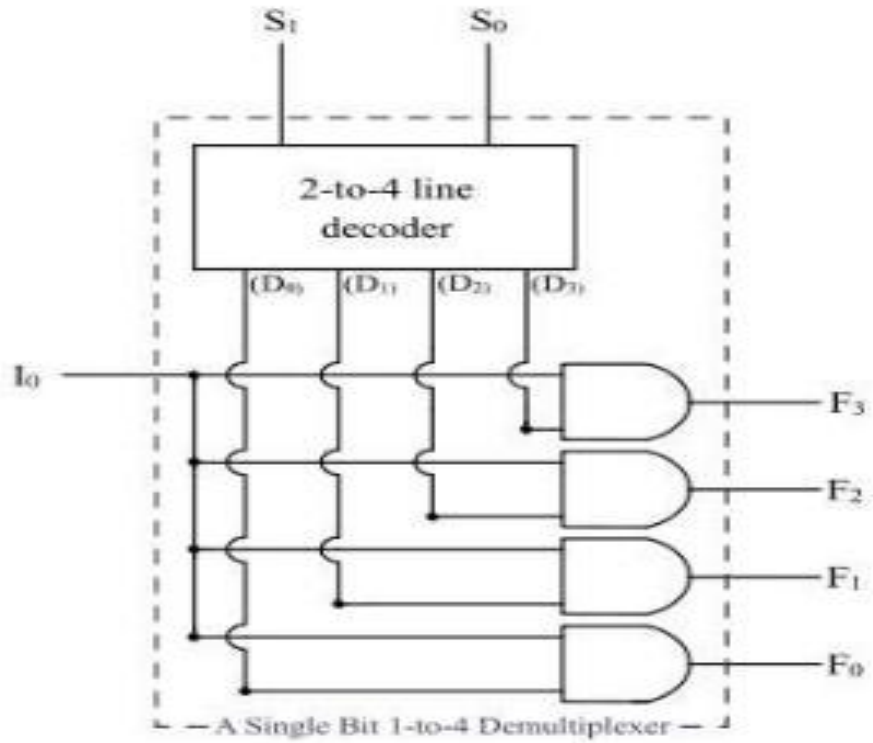
| $s_0$ | $s_1$ | $q_0$ | $q_1$ | $q_2$ | $q_3$ |
|---|---|---|---|---|---|
| 0 | 0 | d | | | |
| 0 | 1 | | d | | |
| 1 | 0 | | | d | |
| 1 | 1 | | | | d |

- Logic circuit



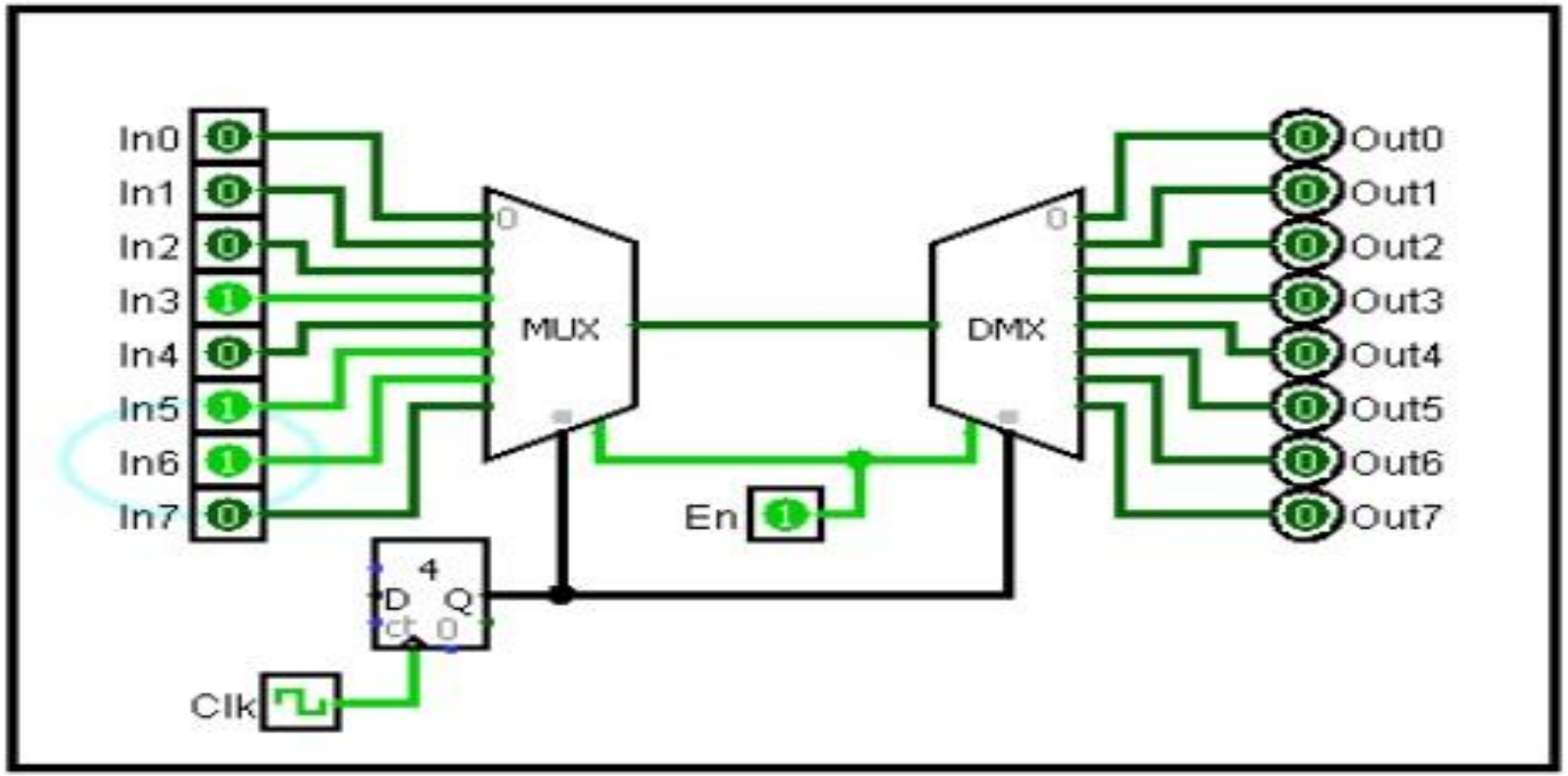

Source: http://do-area.blogspot.com/p/multiplexer-demultiplexer.html

# Demultiplexer Using Decoder




Source: http://en.wikipedia.org

# Multiplexer/Demultiplexer - Application in Telecommunication




Source: http://digilogwiki.com/index.php?title=Multiplexers/Demultiplexers

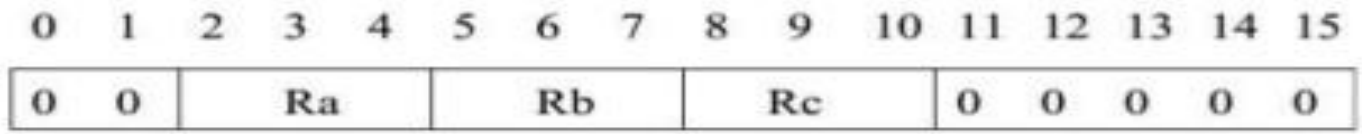
# Encoder & Decoder

- Encoder-
  - An encoder is a combinational circuit that produces the reverse function from that of a decoder.
- Decoder-
  - A decoder is a combinational logic circuit that receives coded information on n input lines and feeds them to maximum of 2n unique output lines after conversion.

# Decoders

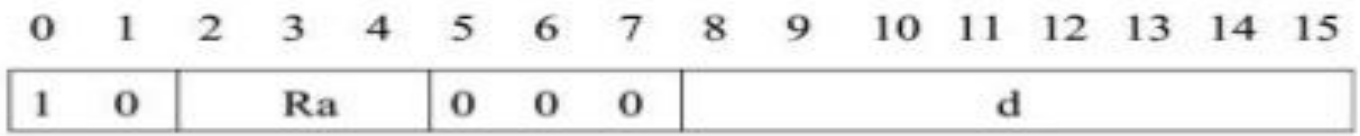
- Suppose a simple microprocessor supports following 2 instructions
  - ADD

Rc = Ra+Rb;
PC = PC+1;



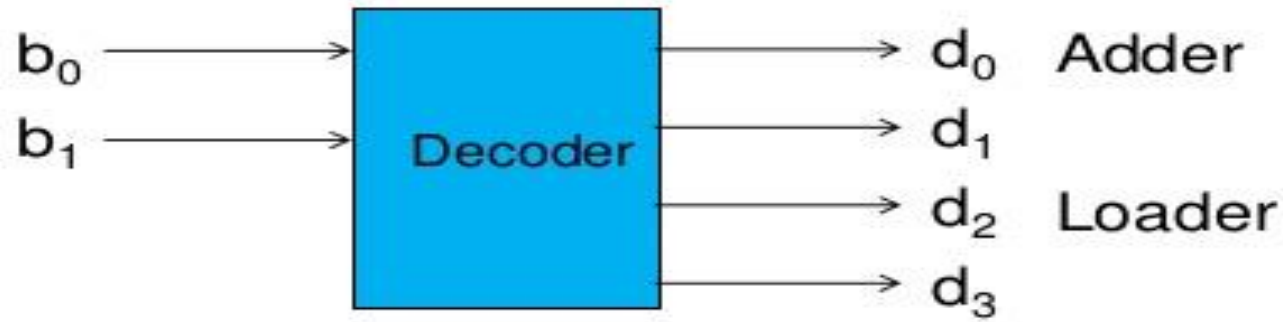

  - LOAD

Ra = d;
PC = PC+1;




- When these instructions execute they'll need to activate different circuits
  - Which circuit is determined by 2 most significant bits

# Decoders




- Converts binary data from $n$ coded inputs to a maximum of $2^n$ unique outputs
- Called $n$-to-$2^n$ decoder

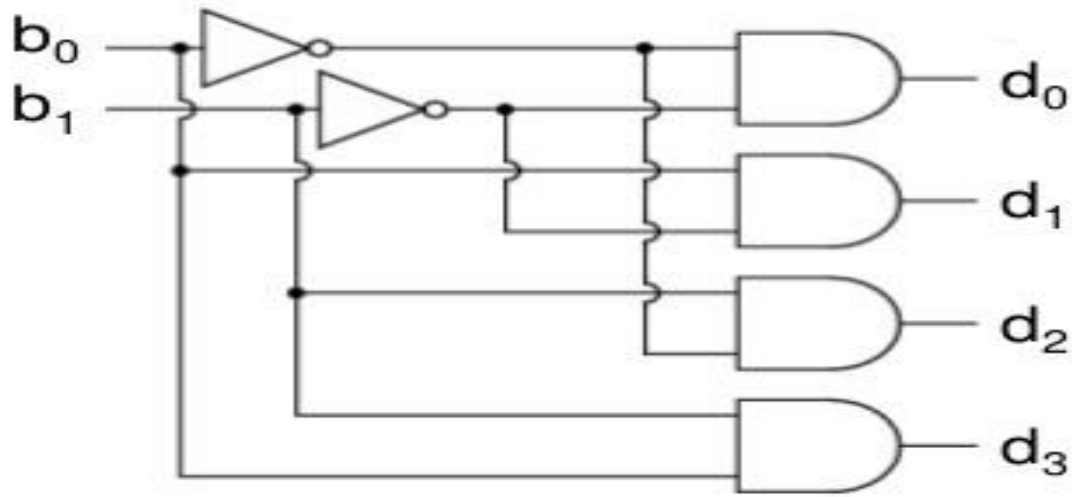
# Decoders

- Truth table for a *2*-to-4 decoder

| $b_0$ | $b_1$ | $d_0$ | $d_1$ | $d_2$ | $d_3$ |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 |

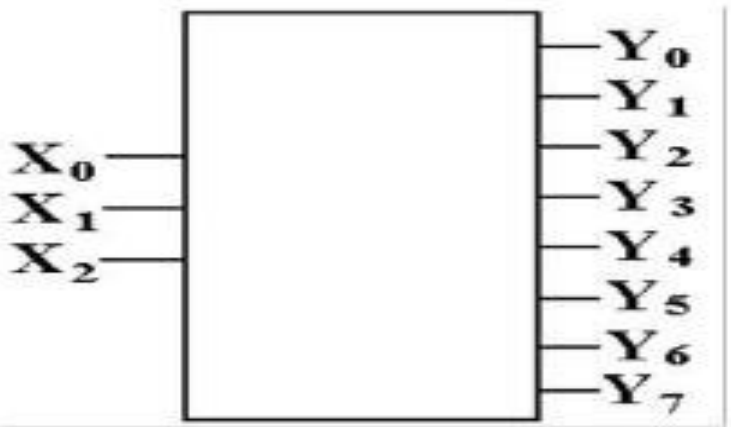
# Decoders

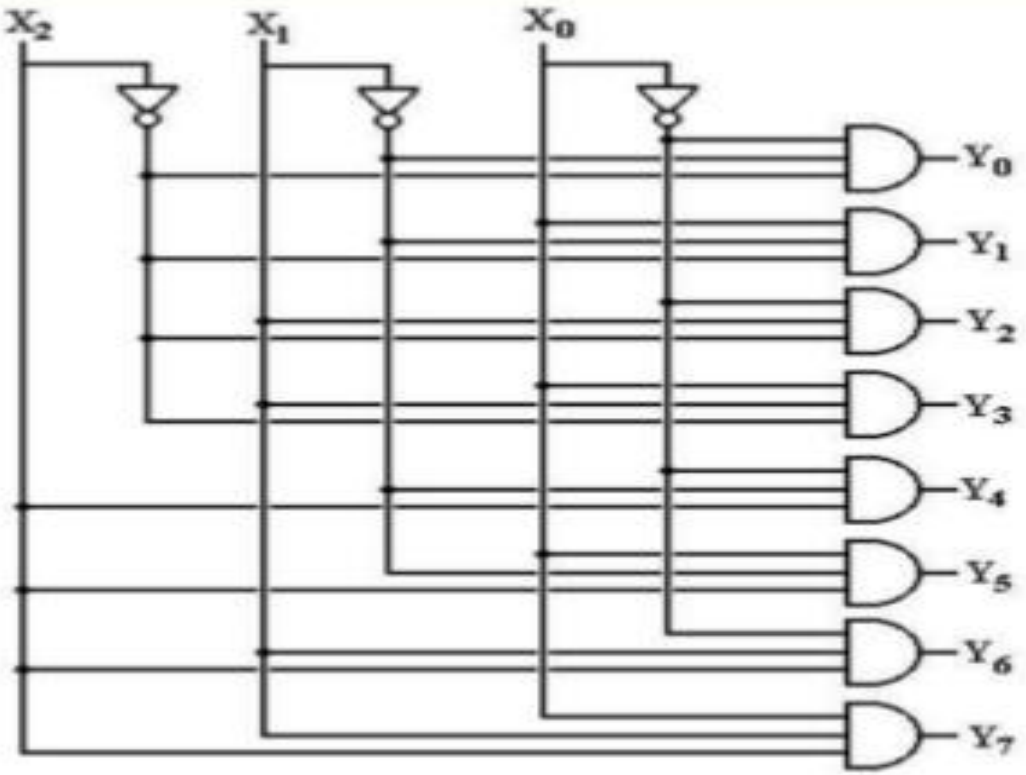
- Draw logic circuit of a 2-to-4 decoder




Source: www.allaboutcircuits.com/vol_4/chpt_9/4.html

# 3-to-8 Decoder

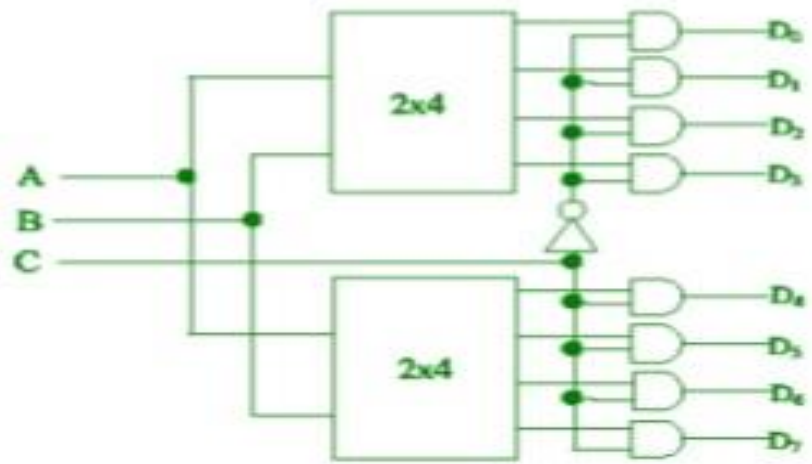


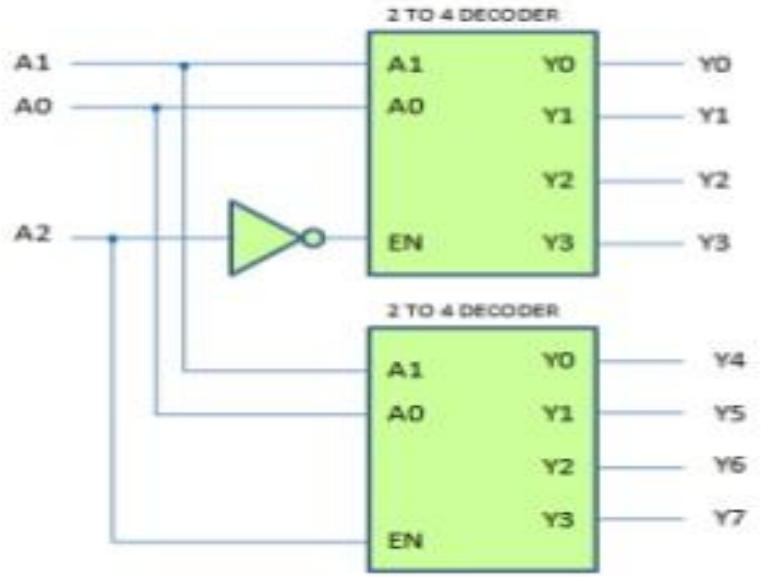

Source: www.edwardbosworth.com/CPSC2105/Lectures/Slides_05/Chapter_03/DecodersAndMux.htm

# Decoder Expansion

- Build a 3-to-8 decoder using 2-to-4 decoders




Source: http://dc167.4shared.com/doc/or00nekd/preview.html

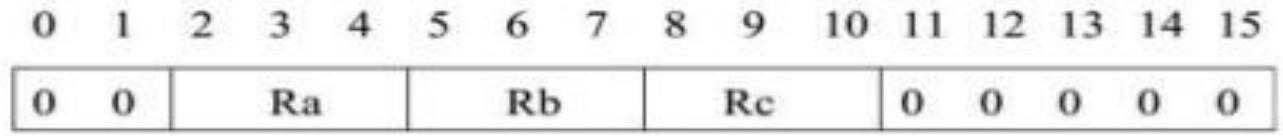



Source: www.teachurselfece.com/2012/02/decoders.html

# Decoders (Cont.)

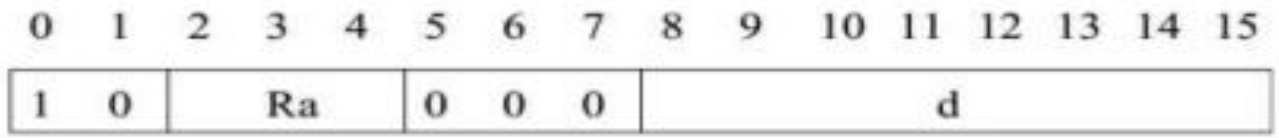
- ADD

Rc = Ra+Rb;
PC = PC+1;



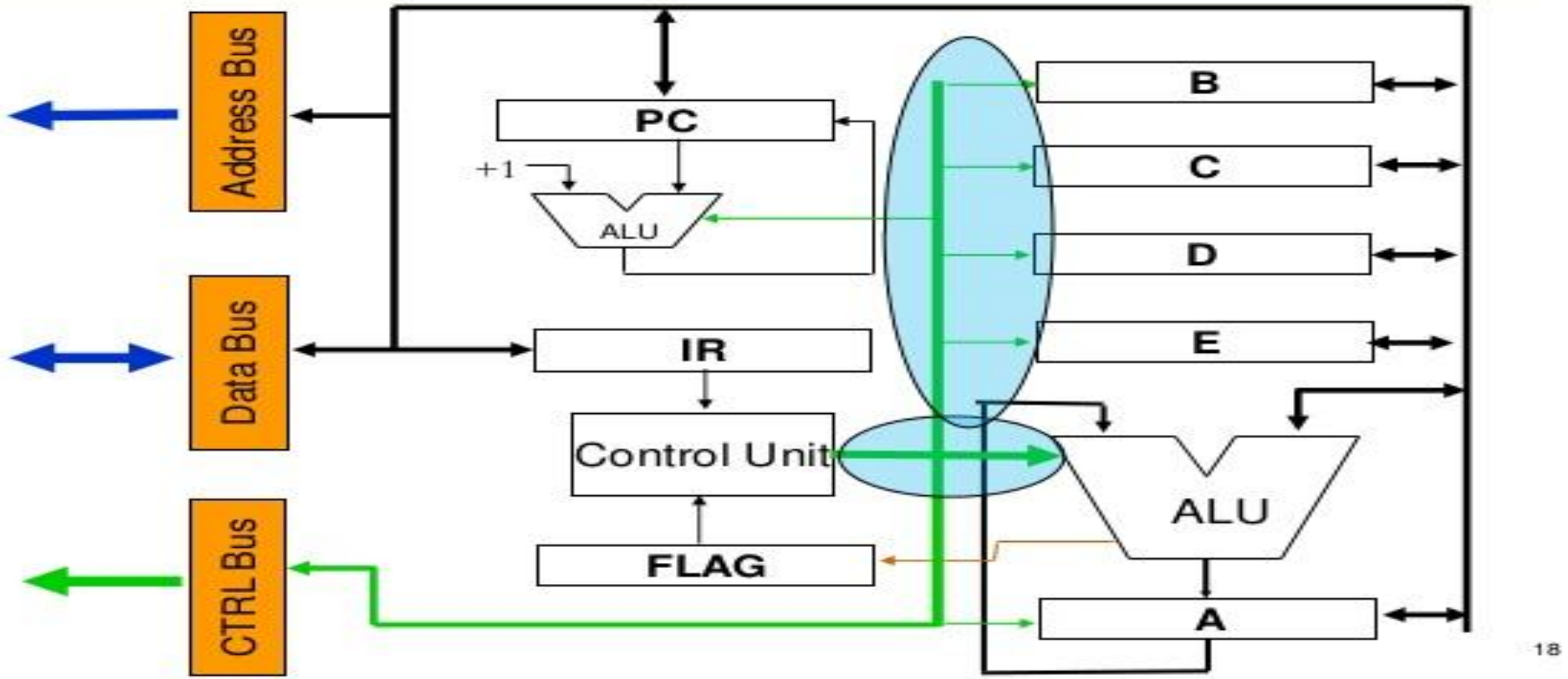

- LOAD

Ra = d;
PC = PC+1;



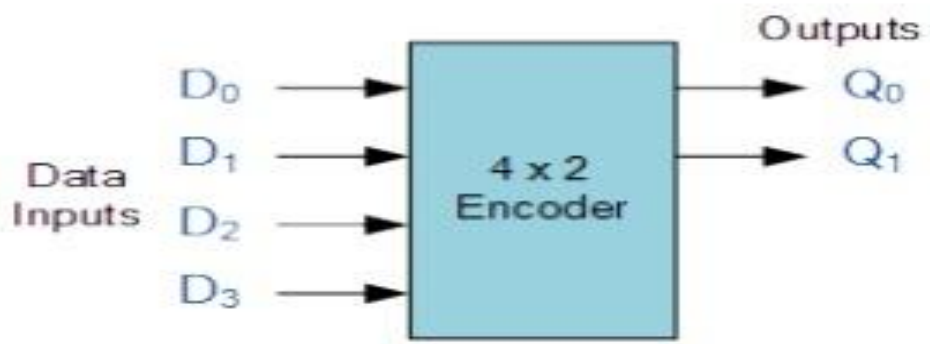

- Also helps us select which registers to use

# Use of Decoders Inside CPU



# Encoders

- Reverse process of a decoder
  - 4-to-2 encoder
  - 3-to-8 encoder

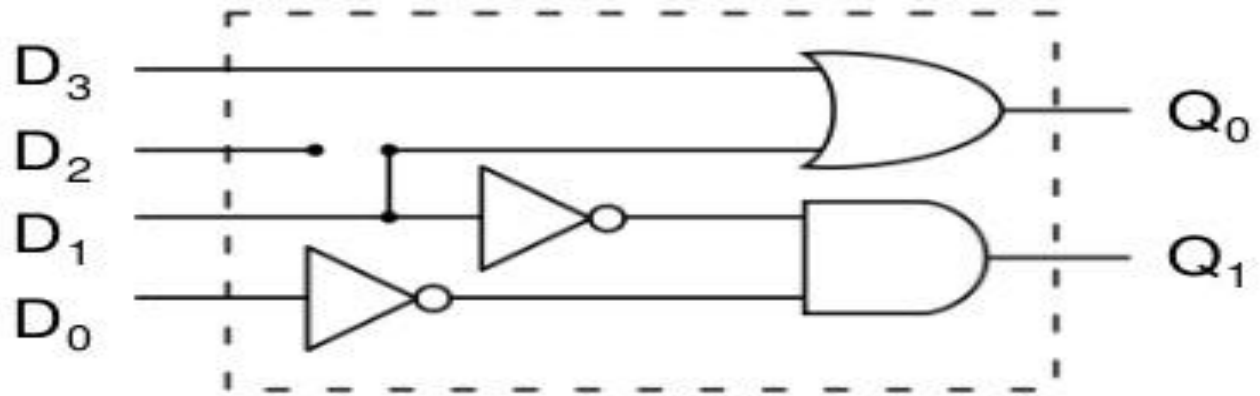



| Inputs | | | | Outputs | |
|---|---|---|---|---|---|
| $D_3$ | $D_2$ | $D_1$ | $D_0$ | $Q_1$ | $Q_0$ |
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 |

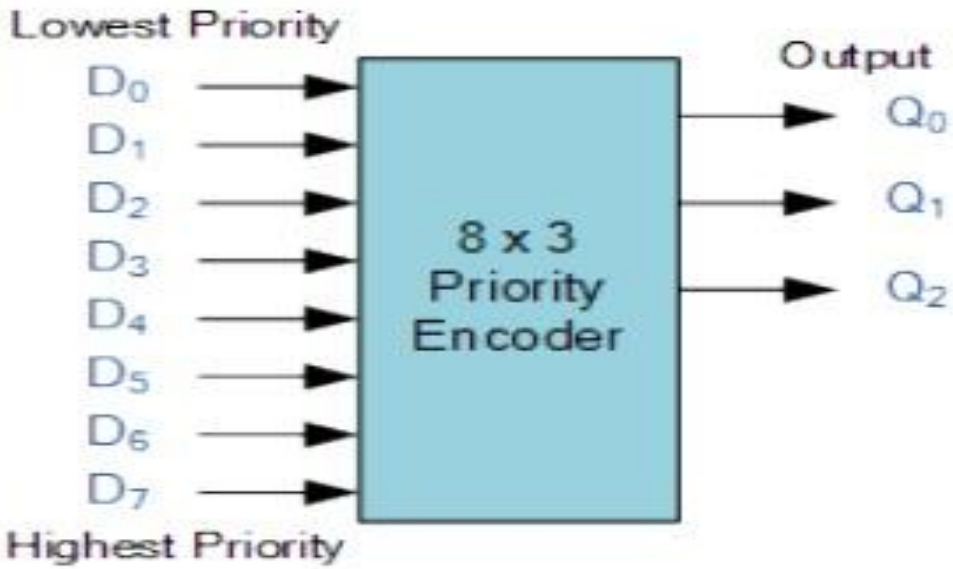
Source: www.electronics-tutorials.ws/combination/comb_4.html

# Encoders

- Draw logic circuit of a 4-to-2 encoder



# Priority Encoder

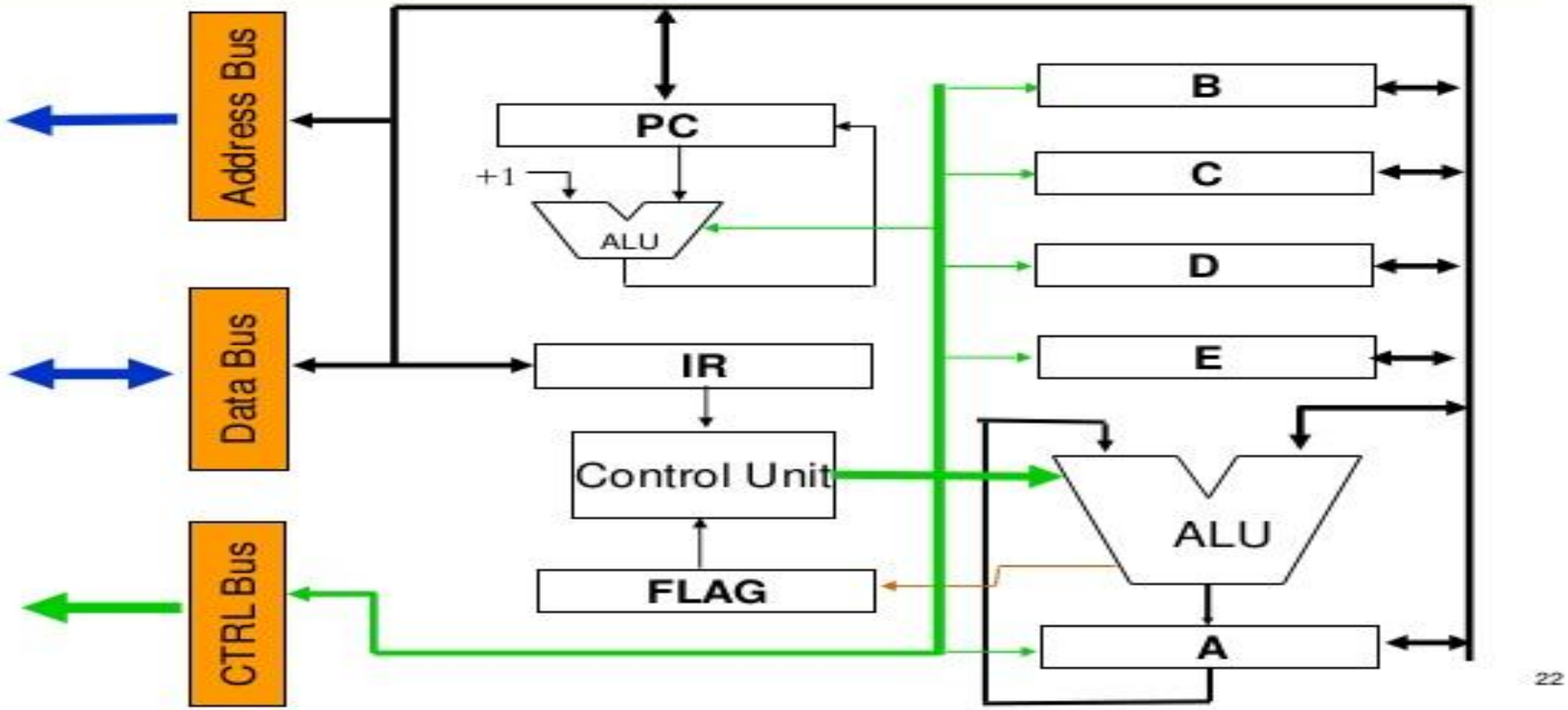



| Inputs | | | | | | | | Outputs | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | $Q_2$ | $Q_1$ | $Q_0$ |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | x | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 1 | x | x | 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 | x | x | x | x | x | 1 | 0 | 1 |
| 0 | 1 | x | x | x | x | x | x | 1 | 1 | 0 |
| 1 | x | x | x | x | x | x | x | 1 | 1 | 1 |

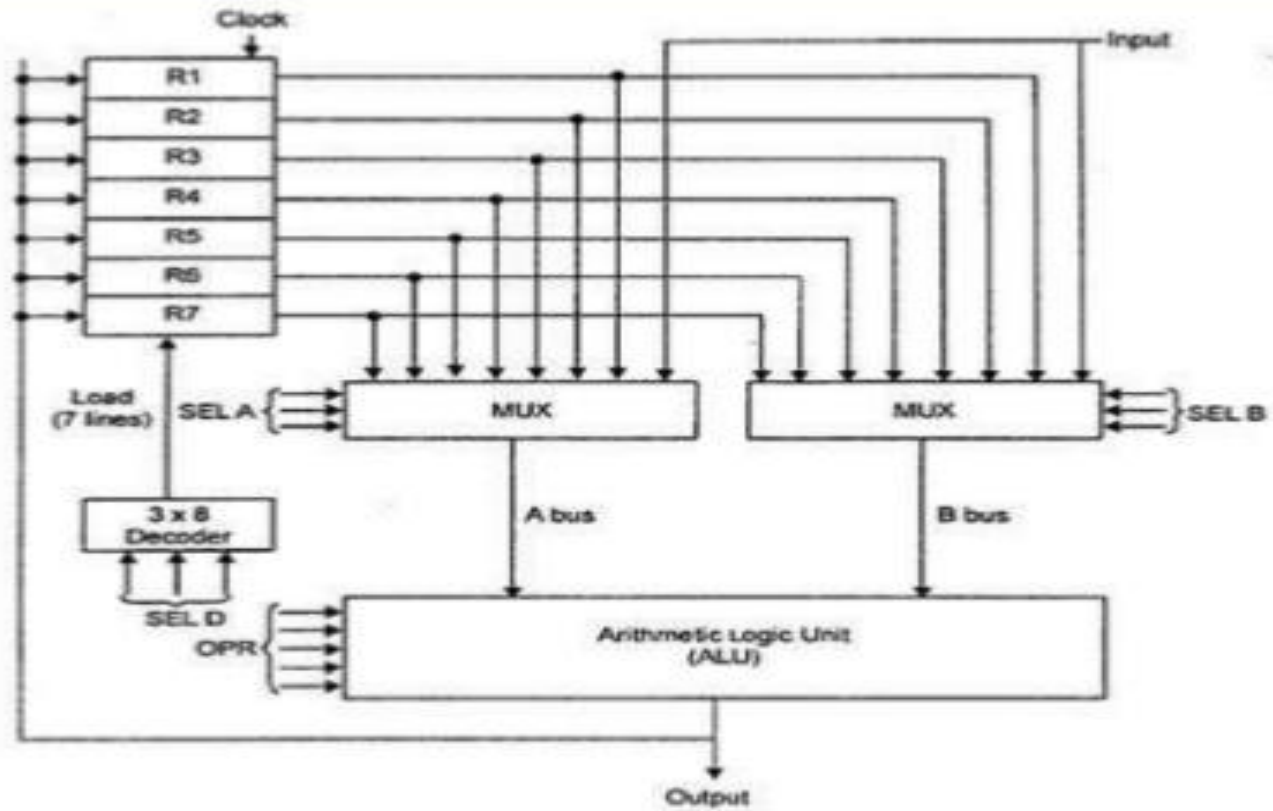
X = dont care

Source: www.electronics-tutorials.ws/combination/comb_4.html

# Internal Structure



# Internal Structure (Cont.)




Source: www.transtutors.com/homework-help/computer-science/computer-architecture/cpu/general-register-organization/

# Half-Adder & Full-Adder

- Half-Adder :
  - A half-adder is a combinational circuit that performs the addition of two bits.

- Full Adder :
  - This type of adder is a little more difficult to implement than a half-adder.
  - The main difference between a half-adder and a full-adder is that the full-adder has three inputs and two outputs.

# Adding 2 Numbers (Cont.)

- Draw logic circuit


Source: Wikipedia.org

- This is called a half adder

# Adding 2 Numbers & a Carry

- Write the truth table for addition of 2 bits A & B as well as a carry from previous low-order bit

| A | B | $C_{in}$ | S | $C_{out}$ |
|---|---|---|---|---|
| 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 |

# Adding 2 Numbers & a Carry (Cont.)

- Write Boolean representation for Sum & Carry
  - Hint – use k-maps
  - $S = (A \oplus B) \oplus C_{in}$
  - $C_{out} = AB + (A \oplus B)C_{in}$

$S =$

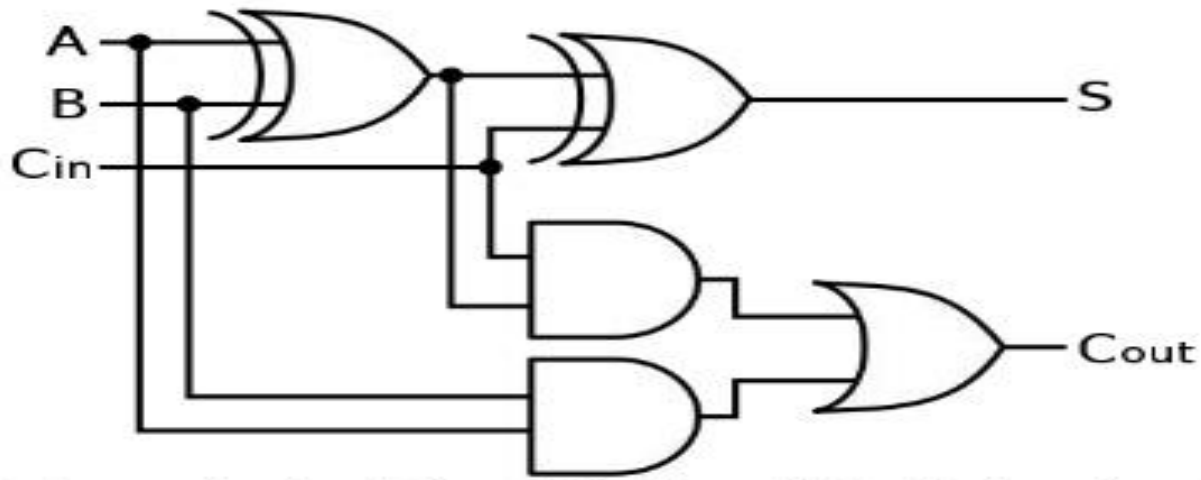
| ab \ c | 0 | 1 |
|---|---|---|
| 00 | 0 | 1 |
| 01 | 1 | 0 |
| 11 | 0 | 1 |
| 10 | 1 | 0 |

$C_{out} =$

| ab \ c | 0 | 1 |
|---|---|---|
| 00 | 0 | 0 |
| 01 | 0 | 1 |
| 11 | 1 | 1 |
| 10 | 0 | 1 |

# Adding 2 Numbers & a Carry (Cont.)

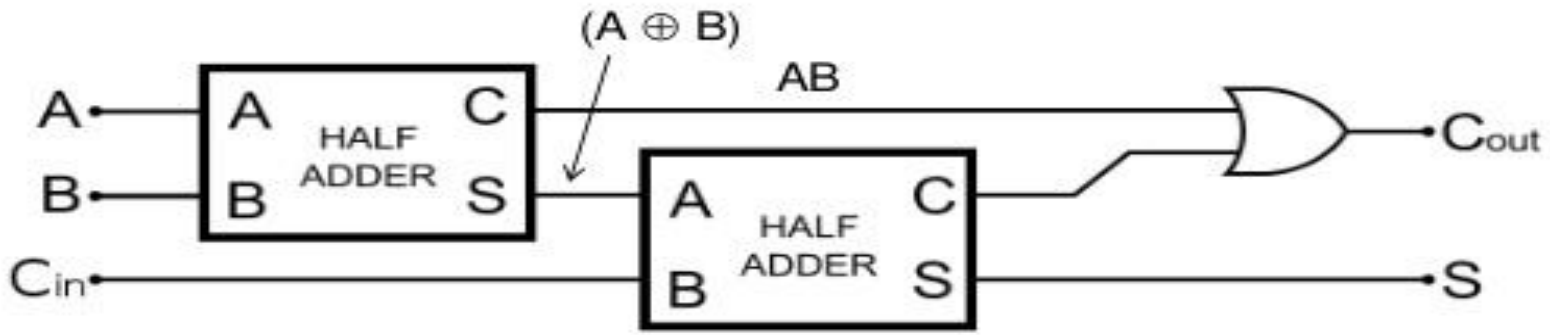
- Draw logic circuit




Source: www.setupsolution.com/how-to-design-a-half-adder-and-full-adder-in-verilog-at-gate-level-modeling/

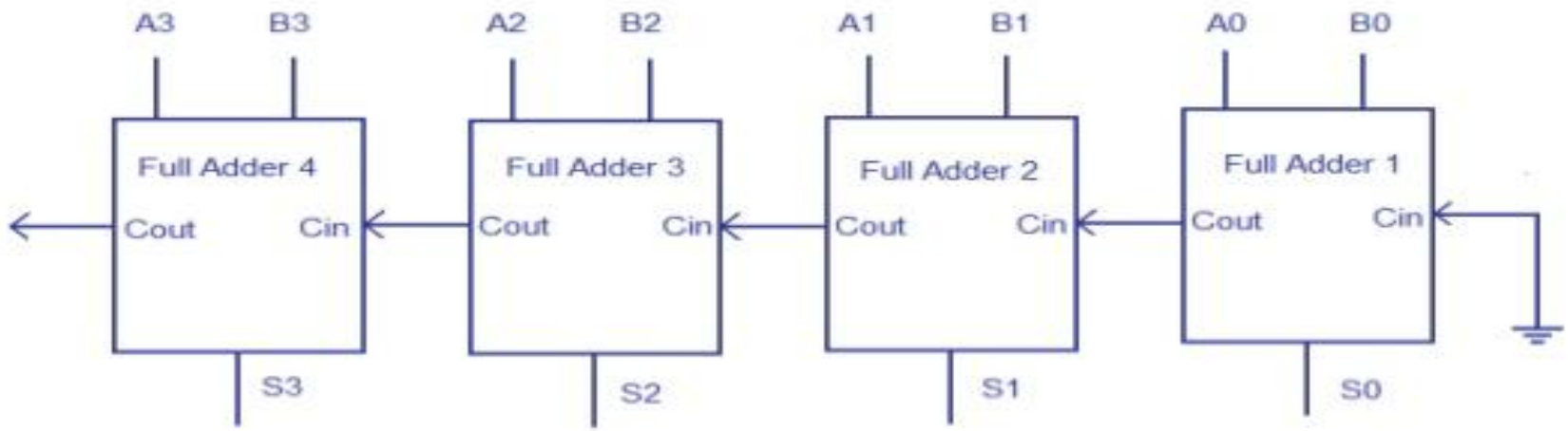
- This is called a full adder

# Schematic Representation of Full Adder




Source: http://en.wikibooks.org

# *n*-bit Adder (Ripple Carry Adder)



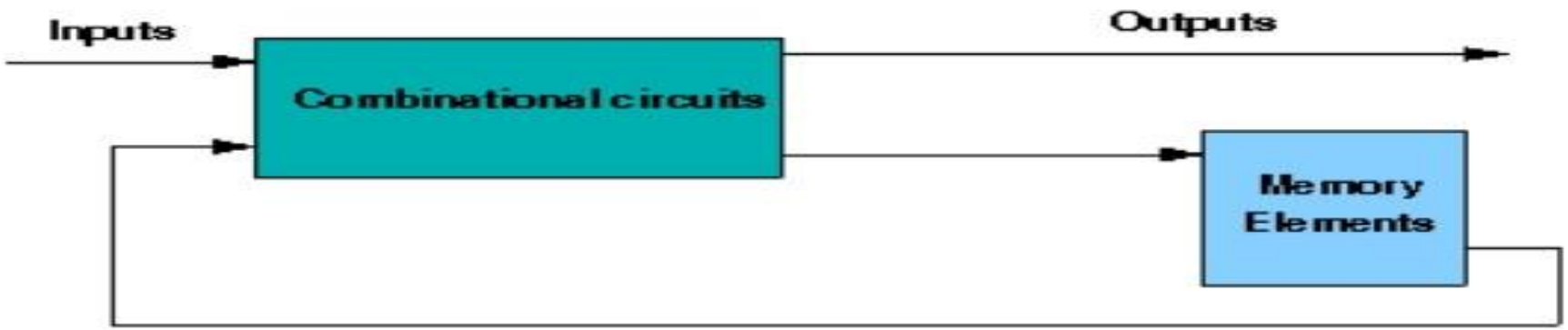

4 bit ripple carry adder

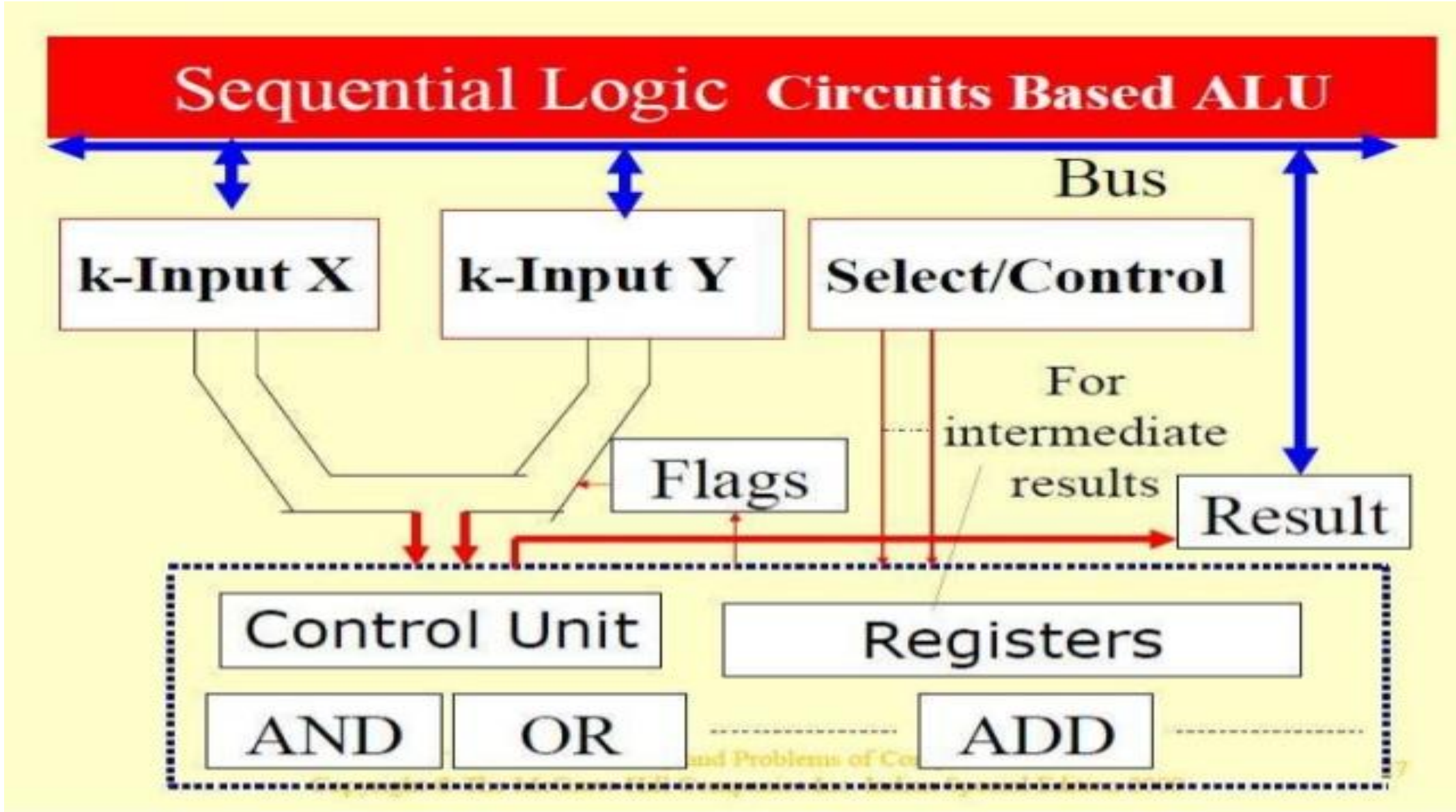
www.circuitstoday.com

# SEQUENTIAL LOGIC CIRCUITS

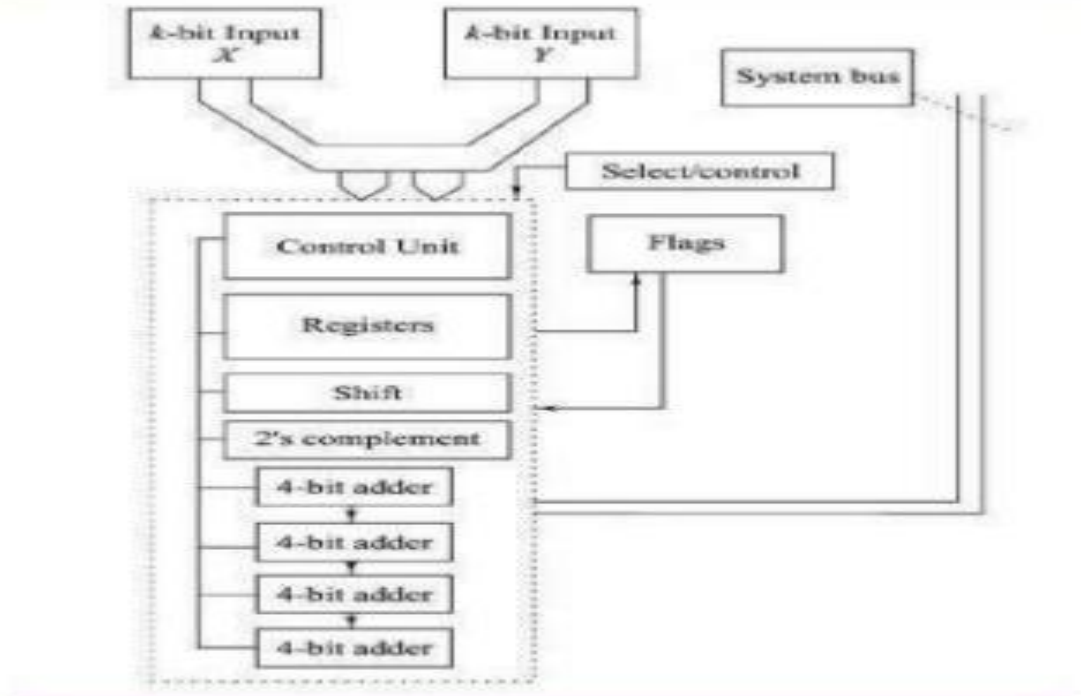
- Made up of combinational circuits and memory elements.
- These memory elements are devices capable of storing ONE-BIT information.
- Output depends on input and previous state.
- Examples of sequential circuits are flip flops, counters, shift registers

# BLOCK DIAGRAM OF A SEQUENTIAL CIRCUIT




Sequential Logic Circuits Based ALU
Bus
k-Input X
k-Input Y
Select/Control
For
intermediate
results
Flags
Result
Control Unit
Registers
AND
OR
ADD

# AN ALU USING SEQUENTIAL CIRCUITS



# Examples of Sequential Circuits :

- Flip-Flops
  - JK Flip-Flop
  - RS Flip-Flop
  - PR Flip-Flop
  - D Flip-Flop
- Registers
- Counters

# Flip-Flops

- Flip-Flops are the basic building blocks of sequential circuits.
- A flip-flop is a binary cell which can store a bit of information.
- A basic function of flip-flop is storage, which means memory. A flip-flop (FF) is capable of storing 1 (one) bit of binary data.
- It has two stable states either '1' or '0'. A flip-flop maintains any one of the two stable states which can be treated as zero or one depending on presence and absence of output signals.

# Registers and Counters

- A circuit with flip-flops is considered a sequential circuit even in the absence of combinational logic.
- Circuits that include flip-flops are usually classified by the function they perform.
- Two such circuits are registers and counters:
- **Registers-**
  - It is a group of flip-flops.
  - Its basic function is to hold information within a digital system so as to make it available to the logic units during the computing process.
- **Counters-**
  - It is essentially a register that goes through a predetermined sequence of states.

# TYPES OF SEQUENTIAL CIRCUITS

Sequential circuits are of two types:

- *SYNCHRONOUS SEQUENTIAL CIRCUITS*
- *ASYNCHRONOUS SEQUENTIAL CIRCUITS*

# → SYNCHRONOUS CIRCUITS

- In synchronous sequential circuits, the state of the device changes only at discrete times in response to a clock Pulse.
- In a synchronous circuit, an electronic oscillator called a *clock* generates a sequence of repetitive pulses called the *clock signal* which is distributed to all the memory elements in the circuit.

# → ASYNCHRONOUS CIRCUITS

- Asynchronous circuit is not synchronized by a clock signal; the outputs of the circuit change directly in response to changes in Inputs.
- The advantage of asynchronous logic is that it can be faster than synchronous logic, because the circuit doesn't have to wait for a clock signal to process inputs.
- The speed of the device is potentially limited only by the propagation delays of the logic gates used.

# THANK YOU

**This content is taken from the text books and reference books prescribed in the syllabus.**