Computer Architecture

ICT 1019Y Week 04 Lecture

Digital Logic
Combinational Circuits

Implementing Boolean Functions

- How do we **physically implement** Boolean functions?

$$F(X,Y,Z) = (X+Y)\,(X+\overline{Y})\,(\overline{X\overline{Z}})$$

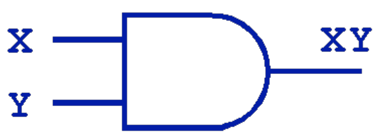
  - Using digital computer circuits called **gates**

- What is a gate?
  - Electronic device that produces a result based on two or more input values
  - Built out of 1-6 transistors (but we'll treat a gate as a single fundamental unit in this class)

- Integrated circuits contain gates organized to accomplish a specific task

Gates: AND, OR, NOT

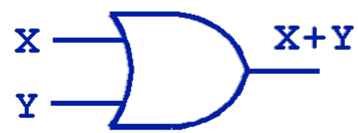
AND Gate



X AND Y

X	Y	XY
0	0	0
0	1	0
1	0	0
1	1	1

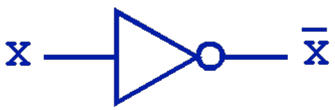
OR Gate



X OR Y

X	Y	X+Y
0	0	0
0	1	1
1	0	1
1	1	1

NOT Gate



NOT X

X	$\overline{X}$
0	1
1	0

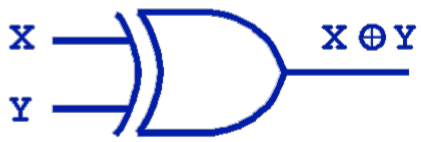
Look at the NOT gate: The ○ symbol represents "NOT". You'll see it on other gates

Gates: XOR

Exclusive OR (XOR)

X XOR Y

X	Y	X ⊕ Y
0	0	0
0	1	1
1	0	1
1	1	0



- The output of the XOR operation is **true** only when the values of the **inputs are different**
- Note the special symbol ⊕ for the XOR operation.

Gates: NAND, NOR

NAND (AND w/NOT)

- AND with NOT afterwards

X NAND Y

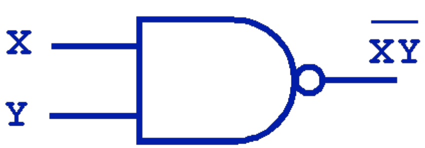
X	Y	X NAND Y
0	0	1
0	1	1
1	0	1
1	1	0

NOR (OR w/NOT)

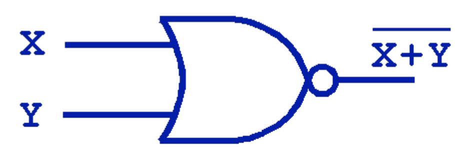
- OR with NOT afterwards

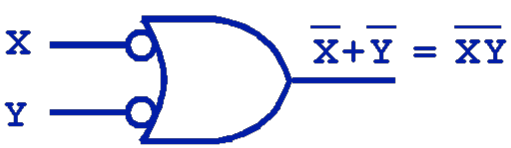
X NOR Y

X	Y	X NOR Y
0	0	1
0	1	0
1	0	0
1	1	0

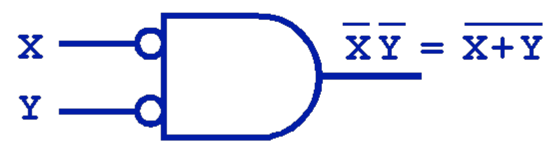


< · *Normal form* · >





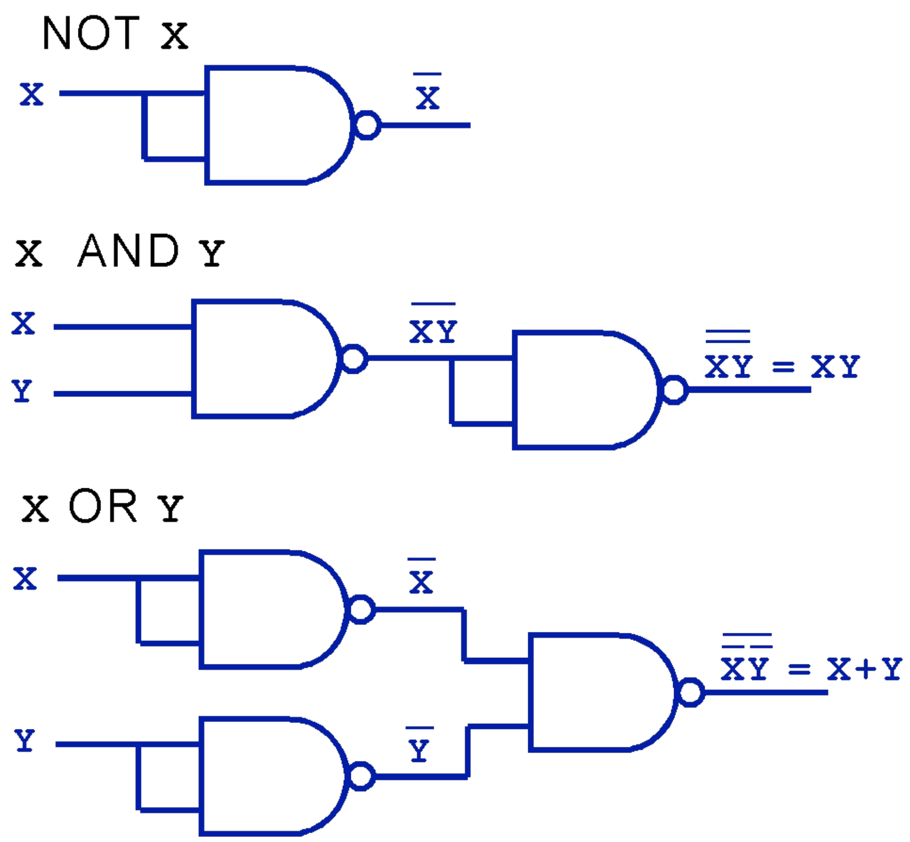
< · *DeMorgan's Law enables these alternate forms* · >



Universal Gates

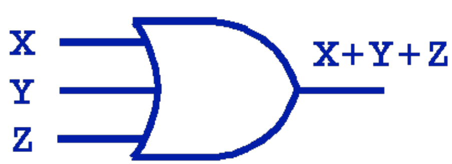
- **Why bother with NAND and NOR?**
  - Don’t they make our life more difficult compared to the obvious AND, OR, NOT?
- NAND and NOR are **universal gates**
  - Easy to manufacture
  - Any Boolean function can be constructed out of only NAND or only NOR gates

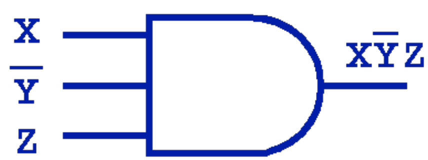
Example using only NAND gates:

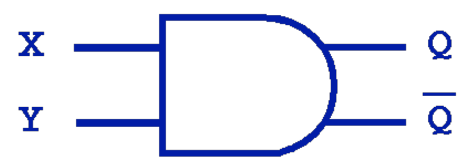


Multiple Input / Multiple Output

- We can physically build many variations of these basic gates
  - Gates with many inputs? Yes!
  - Gates with many outputs? Yes!
    - Second output might be for the complement of the operation



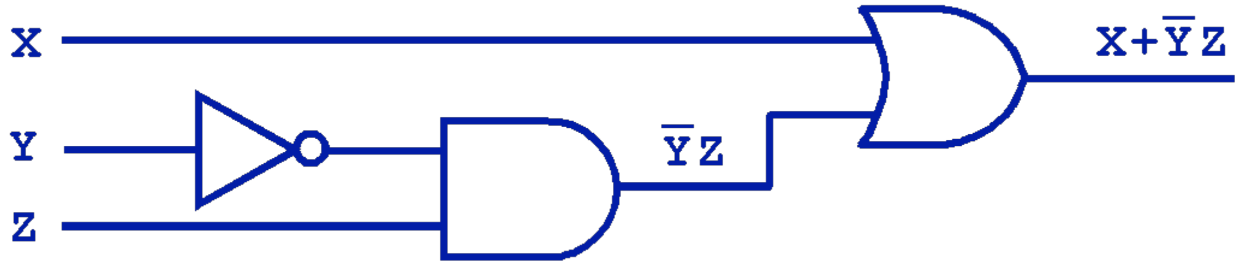




Combining Gates

- Boolean functions can be implemented by combining many gates together

$$F(X,Y,Z) = X+\overline{Y}Z$$



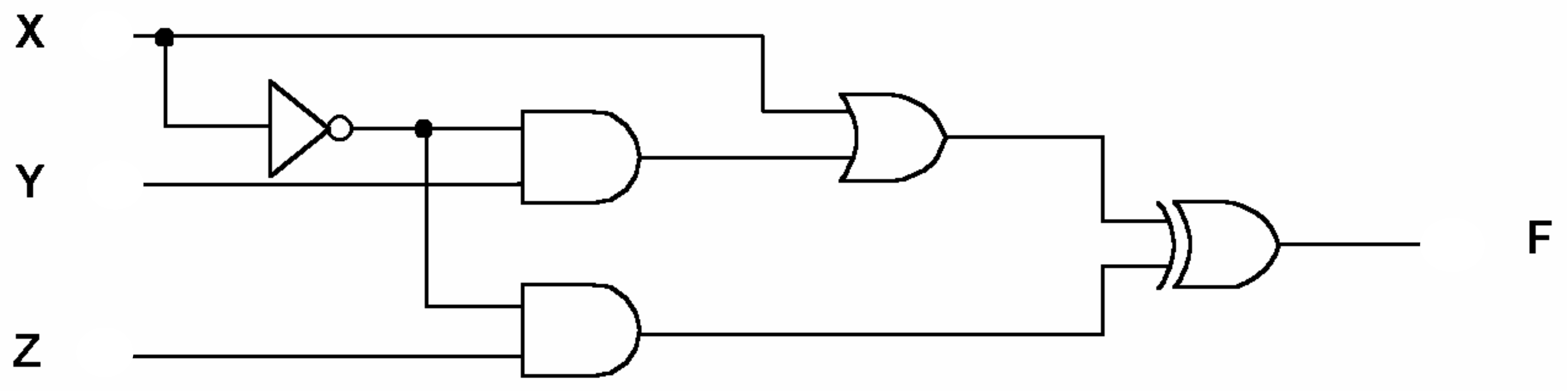
- **Why did we simplify our Boolean expressions previously?**
  - So we can build simpler circuits with fewer gates!

Combinational Circuits

Combinational Circuits

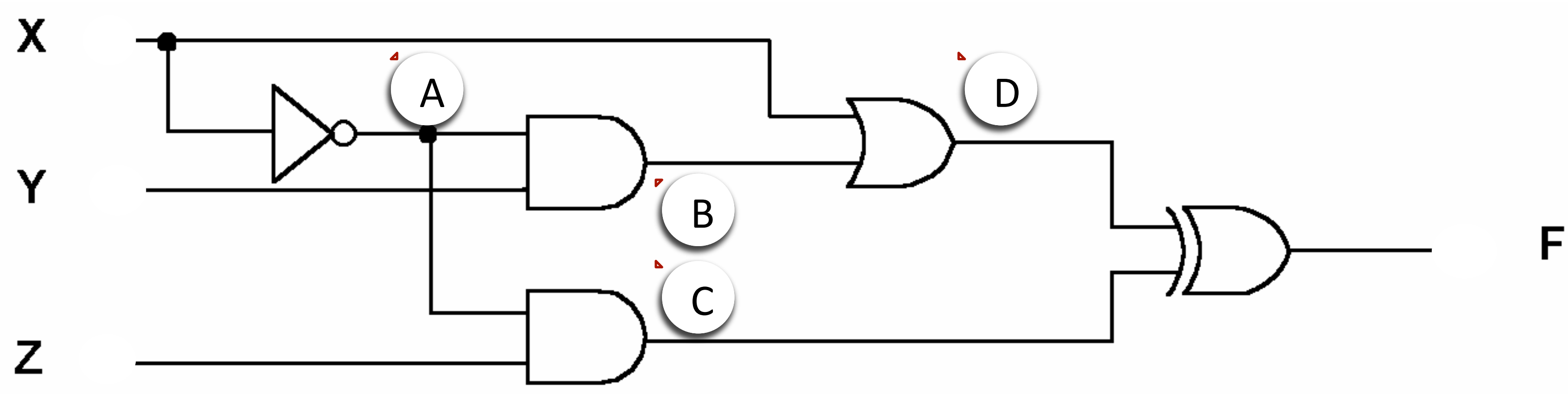
- Two general classifications of circuits
  - **Combinational** logic circuits
  - **Sequential** logic circuits
- Combinational logic circuits
  - Produce a specified output (almost) at the instant when input values are applied
  - Also known as: "Combinatorial circuits"
- Sequential logic circuits
  - Incorporate delay/"memory" elements
  - Will discuss later

Combinational Circuit



- **In teams of 2, write the truth table for this circuit**

Combinational Circuit

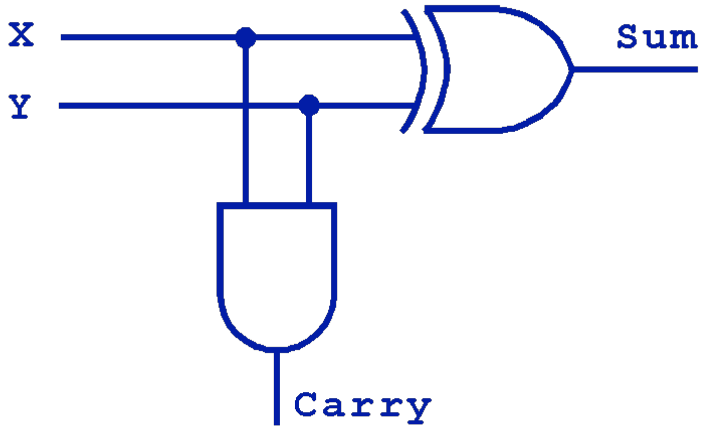


x	y	z	A	B	C	D	F(x,y,z)
0	0	0	1	0	0	0	0
0	0	1	1	0	1	0	1
0	1	0	1	1	0	1	1
0	1	1	1	1	1	1	0
1	0	0	0	0	0	1	1
1	0	1	0	0	0	1	1
1	1	0	0	0	0	1	1
1	1	1	0	0	0	1	1

Combinational Circuit – Half Adder

- **Half Adder**
  - Finds the sum of two bits
- **How can I implement the truth table?**
  - Sum = x ⊕ y (XOR)
  - Carry = x AND y

Inputs		Outputs	
X	Y	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



Combinational Circuit – Full Adder

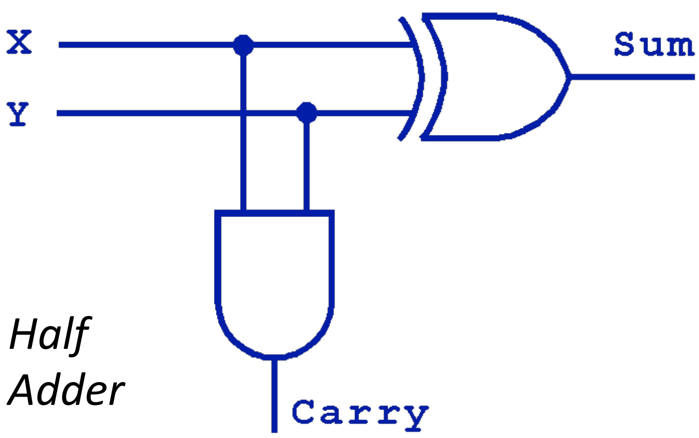
- A **full adder** is a half adder plus the ability to process a **carry-input** bit

Inputs			Outputs	
X	Y	Carry In	Sum	Carry 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

New input:

Combinational Circuit – Full Adder

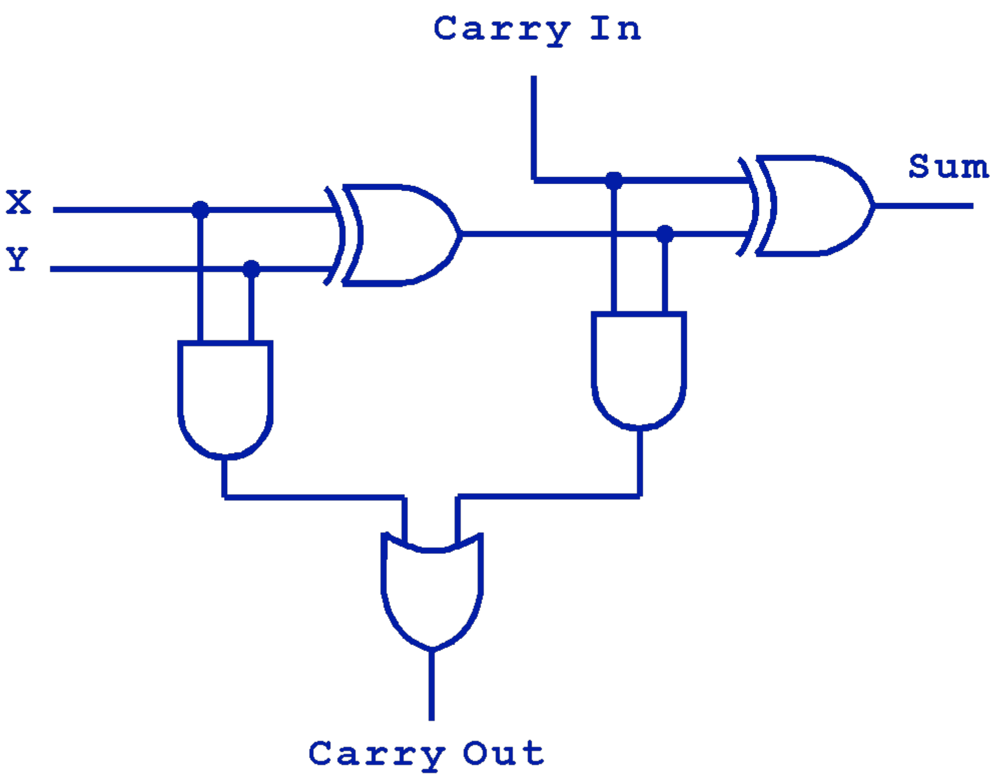
- **What do we need to add to the half adder (shown below) to make it a full adder?**



Inputs			Outputs	
X	Y	Carry In	Sum	Carry 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

Combinational Circuit – Full Adder

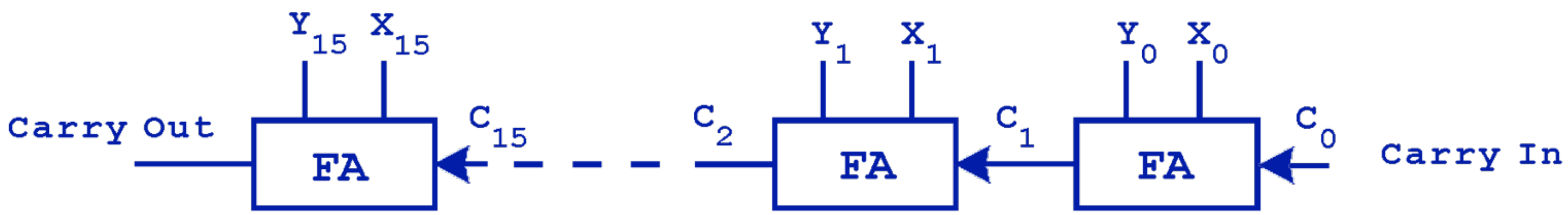
A Full Adder is really just two Half Adders in **series**



Inputs			Outputs	
X	Y	Carry In	Sum	Carry 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

Ripple Carry Adder

- Full adders can be connected in series to form a ripple carry adder
  - The carry bit "ripples" from one adder to the next



- **Why is the performance of this approach <u>slow</u>?**
  - Slow due to long propagation paths
  - Modern systems use more efficient adders

Combinational Circuit – Decoder

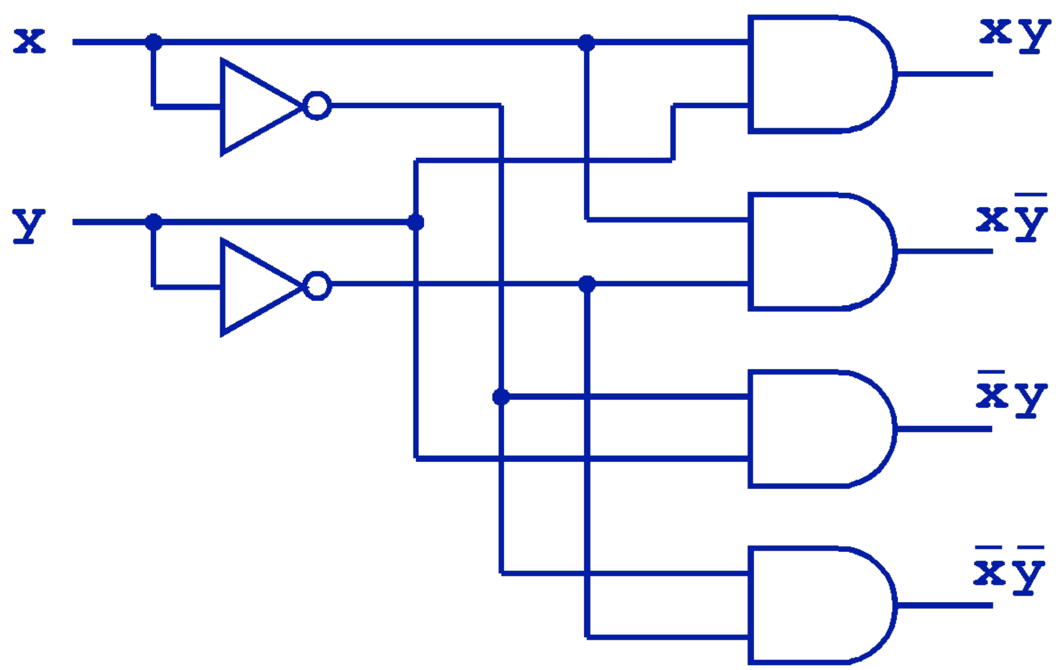
- Selects one (of many) outputs from a single input

Decoder block diagram
(black box)

Combinational Circuit – Decoder

- Implementation of a 2 input to 4 output decoder

If x = 0 and y = 1, which output line is enabled?



Memory

- Common **decoder** application: **Memory** address decoders
  - n inputs can select any of 2^n locations.
- Example: Suppose we build a memory that stores 2048 bytes using several 64x4 RAM chips
  - How do we determine **which RAM chip to use** when reading/writing a particular address?

Memory

Build this:

Full Memory
2048 total bytes
(or 2048 = 2^{11} addresses,
1 byte per address)

Data wires
(8)

Address wires
(11)

With many of these:

64x4 RAM Chip
64 (or 2^6) locations
4 bits per location

Data wires
(4)

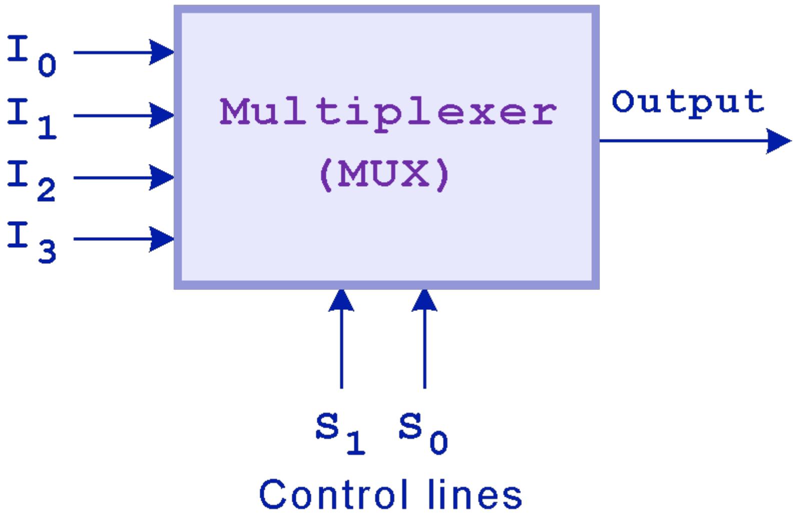
Address wires
(6)

Memory

- To get 2048 total addresses, we need 2048/64 = **32 banks** of RAM chips
- To make each address contains one byte (8 bits) we must access 8/4 = 2 chips in parallel
  - Therefore, a total of 32*2 = **64 RAM chips**
  - Picture an array of RAM chips
    - 32 rows
    - 2 columns
- To determine which of 32 possible banks to read data from, a 5-to-32 decoder is needed ($2^5 = 32$)

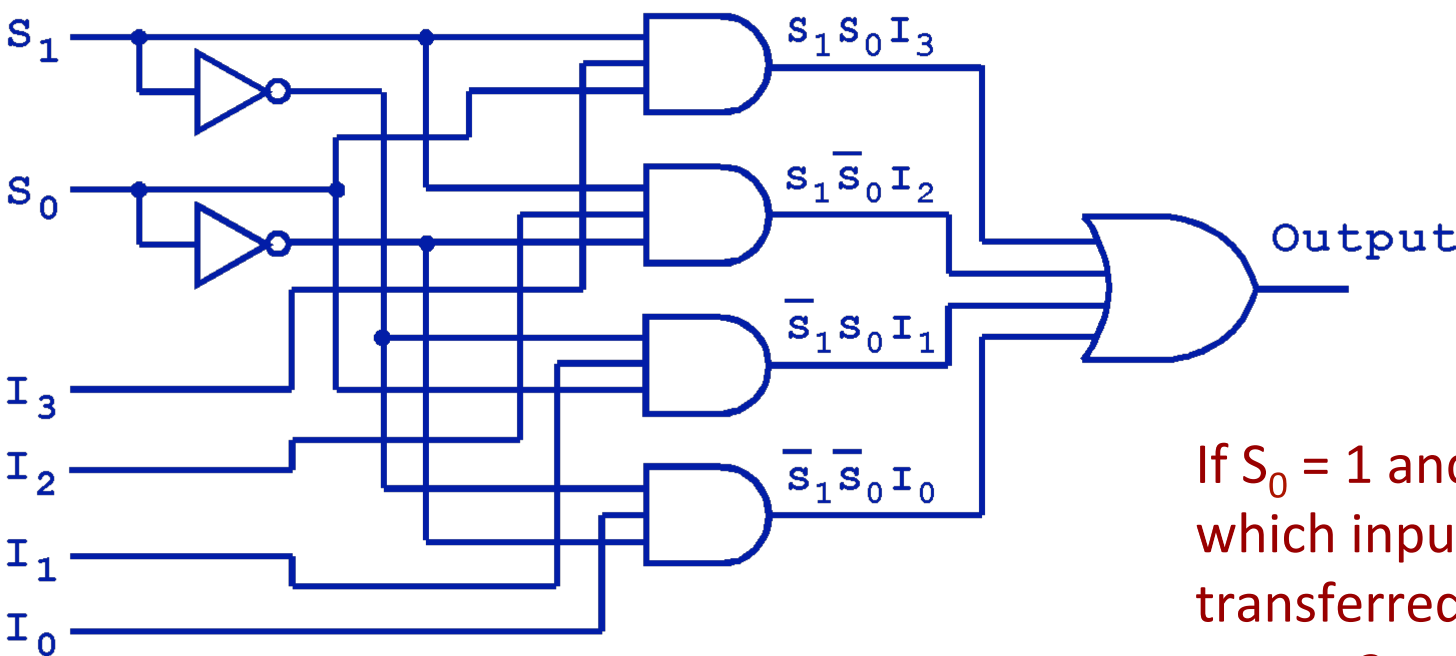
Combinational Circuit – Multiplexer

- A **multiplexer** selects a single output from several inputs
- Which input is chosen?
  - Selected by the value on the multiplexer's control lines
- To select from *n* inputs, $\log_2 n$ control lines are needed.



Combinational Circuit – Multiplexer

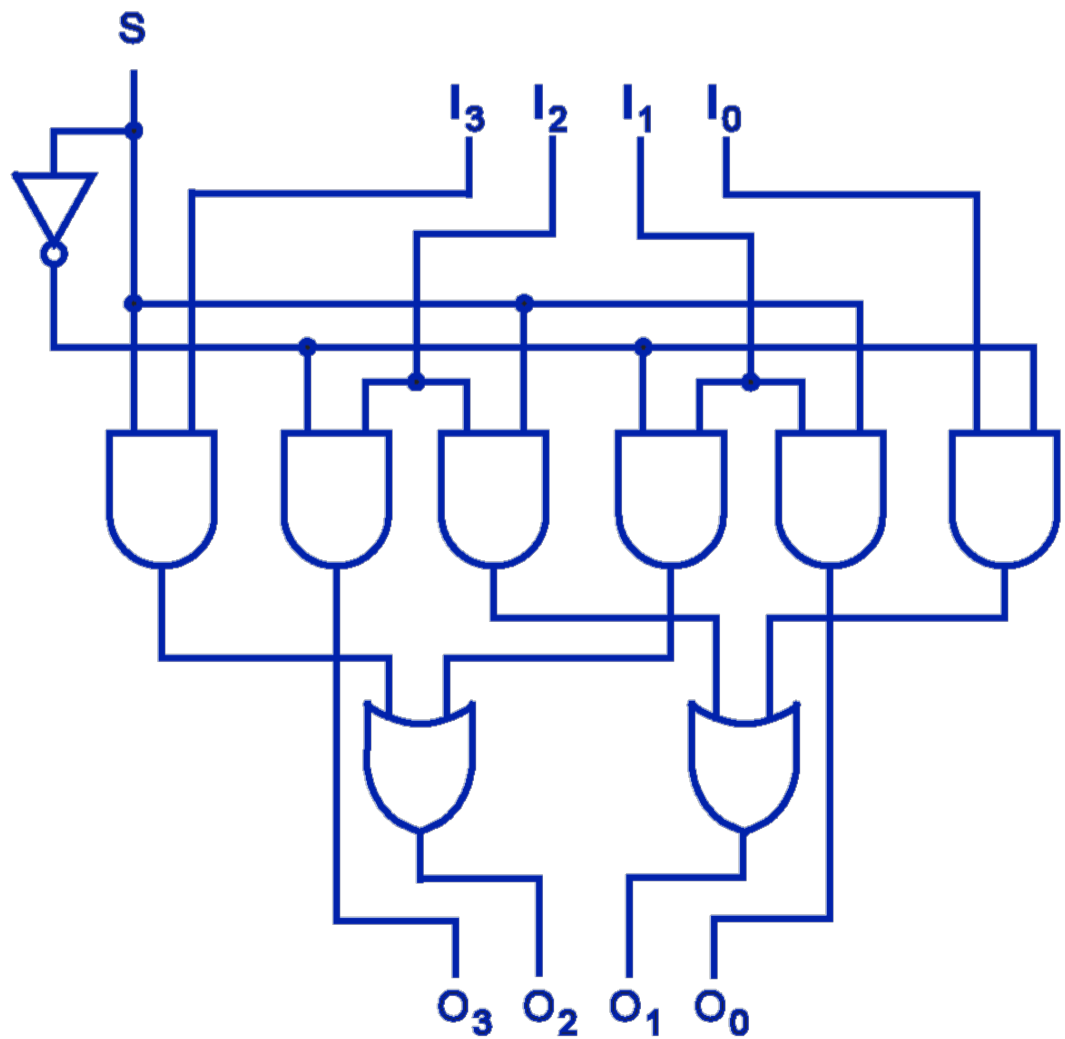
- Implementation of a 4-to-1 multiplexer



If $S_0 = 1$ and $S_1 = 0$, which input is transferred to the output?

Combinational Circuit – Shifter

- This **shifter** moves the bits of a 4-bit input one position to the left or right
- **If S = 0, in which direction do the input bits shift?**
  - Left!



Combinational Circuits

- **Does the output of a combinational circuit change instantly when the input changes?**
  - No – takes a tiny (but measurable) length of time
  - Electrical signals in a wire have a finite speed
  - A transistor takes a finite time to change state

Recap

- **What is the difference between a decoder and a multiplexer?**