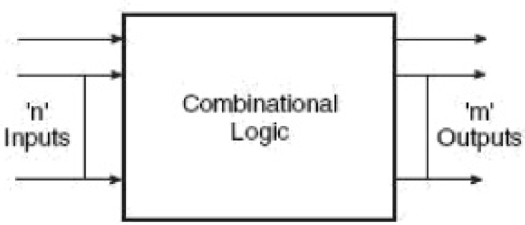
Computer Science 1000: Part #4

Digital Circuits

Digital Circuits: An Overview

- Given binary memory, need to build **digital circuits** that implement computational operations (cf. **analog circuits**).
- Digital circuit as n-input $\Longrightarrow$ m-output transformation:



- Focus here on **combinatorial circuits** that do not involve feedback (cf. feedback-based **sequential circuits**).

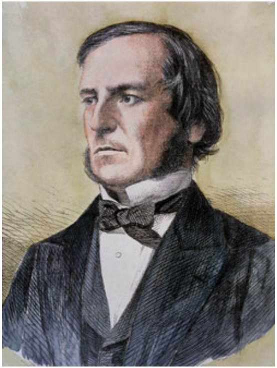
Digital Circuits: An Overview (Cont'd)

- Two types of circuits: **arithmetic** and **control**.
- Specify circuit in terms of input-output behavior, e.g.,

Inputs			Outputs	
A	B	C	OUT_1	OUT_2
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

. . . But how do we design circuits from specifications? . . .

Boolean Logic: An Overview



George Boole
(1815-1864)

- Self-taught mathematician.
- 1854 book *The Laws of Thought* developed algebraic approach to logic (**Boolean logic**); part of algebraic formalization of other areas of mathematics, e.g., geometry, probability.
- Based on variables with values $True$ or $False$ and three operators: AND ($\cdot$), OR ($+$), and NOT ($\overline{A}$).

Boolean Logic: An Overview (Cont'd)

Specify behavior of operators as **truth tables.**

A	B	A AND B	A OR B	NOT A
False	False	False	False	True
False	True	False	True	True
True	False	False	True	False
True	True	True	True	False

Boolean Logic: An Overview (Cont'd)

- Operators and variables can be combined to create expressions, e.g.,

$$\overline{((A \cdot B) + \overline{C})}$$

NOT ((A AND B) OR (NOT C))

- Each expression in turn has an associated truth table, which is created by applying the operators in the expression to each possible combination of values for the variables in the expression, e.g.,

$$A = False, B = True, C = False \Longrightarrow False$$

- What about deriving an expression for a given truth table?

Boolean Logic: An Overview (Cont'd)

OR together the AND-expressions corresponding to variable values in the rows of the truth table that yield result *True*, e.g.,

A	B	C	$RESULT$	
False	*False*	*False*	*False*	
False	*False*	*True*	*True*	$\Longleftarrow$
False	*True*	*False*	*False*	
False	*True*	*True*	*True*	$\Longleftarrow$
True	*False*	*False*	*False*	
True	*False*	*True*	*True*	$\Longleftarrow$
True	*True*	*False*	*False*	
True	*True*	*True*	*False*	

$$\Longrightarrow (\overline{A} \cdot \overline{B} \cdot C) + (\overline{A} \cdot B \cdot C) + (A \cdot \overline{B} \cdot C)$$

Digital Logic Design: Beginnings

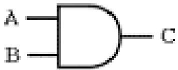
Claude Shannon (1916-2001)

Electromechanical Telephone Switch (1930s))

Digital Logic Design: Beginnings (Cont'd)

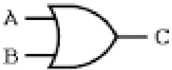
- Shannon MSc thesis (MIT, 1937): Boolean logic can be used to design telephone switching networks!
- Let 0 and 1 in circuits correspond to *False* and *True* and abstract logic gates to Boolean operators.

AND



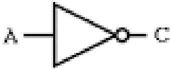
Inputs		Output
A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

OR



Inputs		Output
A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

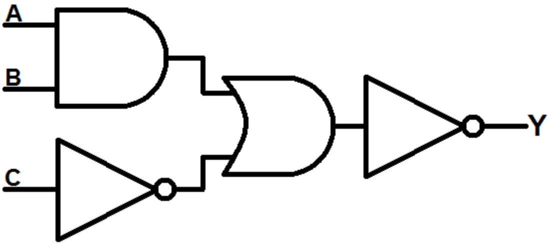
NOT



Input	Output
A	C
0	1
1	0

Digital Logic Design: Beginnings (Cont'd)

Represent network circuits as logic gate diagrams, e.g.,



Boolean expression	$\iff$	logic gate diagram
Truth table	$\iff$	logic gate behavior specification

Digital Circuit Design: The Sum-of-Products Algorithm

Get behaviour specification for circuit
for each output column in specification do
    Construct AND subexpressions for rows with output 1
    Use ORs to combine the constructed AND subexpressions
    Create logic circuit diagram corresponding to OR expressions

Digital Circuit Design: The Sum-of-Products Algorithm (Cont'd)

A	B	C_{IN}	SUM	C_{OUT}	
0	0	0	0	0	
0	0	1	1	0	$\Longleftarrow SUM$
0	1	0	1	0	$\Longleftarrow SUM$
0	1	1	0	1	$\Longleftarrow C_{OUT}$
1	0	0	1	0	$\Longleftarrow SUM$
1	0	1	0	1	$\Longleftarrow C_{OUT}$
1	1	0	0	1	$\Longleftarrow C_{OUT}$
1	1	1	1	1	$\Longleftarrow SUM, C_{OUT}$

$$SUM = (\overline{A} \cdot \overline{B} \cdot C_{IN}) + (\overline{A} \cdot B \cdot \overline{C_{IN}}) + (A \cdot \overline{B} \cdot \overline{C_{IN}}) + (A \cdot B \cdot C_{IN})$$
$$C_{OUT} = (\overline{A} \cdot B \cdot C_{IN}) + (A \cdot \overline{B} \cdot C_{IN}) + (A \cdot B \cdot \overline{C_{IN}}) + (A \cdot B \cdot C_{IN})$$

Arithmetic Circuit Design: 1-Bit Adder

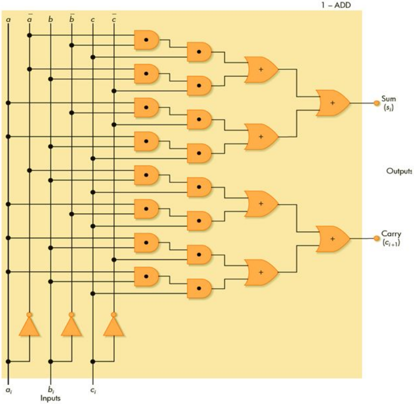
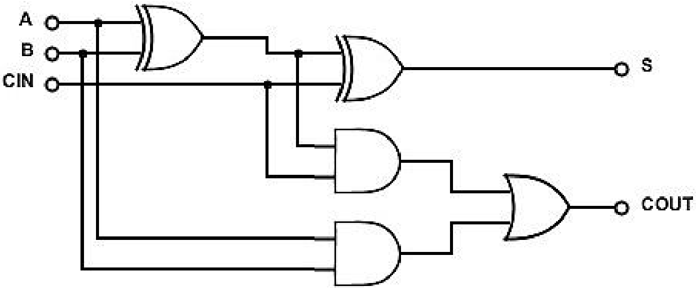


Figure 4.26 Complete 1-ADD Circuit for 1-Bit Binary Addition

Arithmetic Circuit Design: 1-Bit Adder (Cont'd)

Can dramatically decrease the number of gates by applying rules of Boolean algebra and using advanced logic gates such as XOR (Exclusive OR), e.g.,



Arithmetic Circuit Design: n-Bit Adder

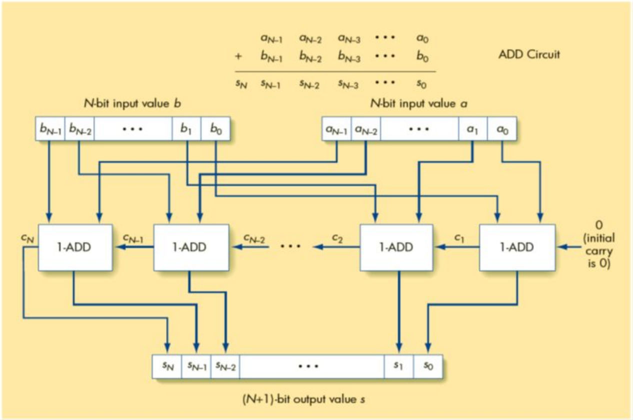


Figure 4.27 The Complete Full Adder ADD Circuit

Arithmetic Circuit Design: 1-Bit Compare-For-Equality

- This type of circuit returns 1 if the two given bits A and B have the same value.

A	B	$RESULT$	
0	0	1	$\Longleftarrow$
0	1	0	
1	0	0	
1	1	1	$\Longleftarrow$

$$RESULT = (\overline{A} \cdot \overline{B}) + (A \cdot B)$$

- Again, there are several implementations of this circuit.

Arithmetic Circuit Design: 1-Bit Compare-For-Equality (Cont'd)

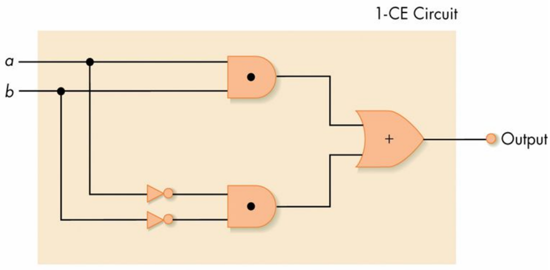
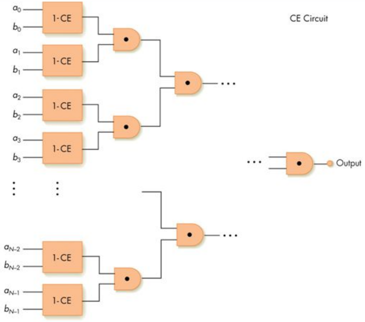


Figure 4.22
One-Bit Compare for Equality Circuit

Arithmetic Circuit Design: n-Bit Compare-For-Equality

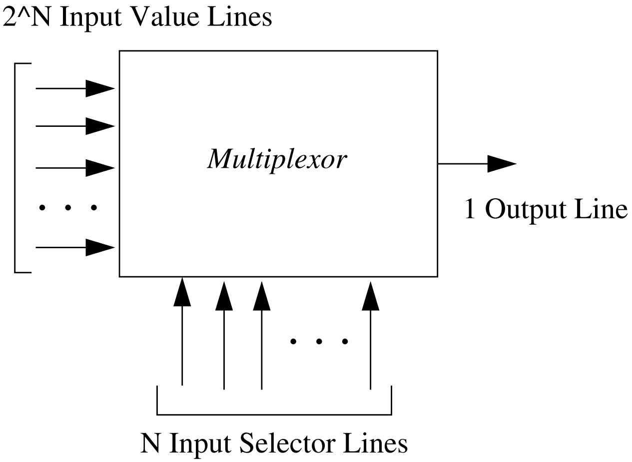


N-bit Compare for Equality Circuit

Control Circuits: Overview

- Control circuits determine the order in which operations are carried and select the correct data values to be processed $\Longrightarrow$ they are sequencing and decision-making circuits.
- Two main types:
  1. **Multiplexor**: Use N input selector lines to determine which of 2^N input value lines has its value copied to the single output line (see Textbook, Figure 4.33, p. 211).
  2. **Decoder**: Use N input selector lines to determine which of 2^N output lines is set to 1 (with all others being set to 0) (see Textbook, Figure 4.35, p. 213).

Control Circuit Design: Multiplexor (Abstract)



Control Circuit Design: Two-input Multiplexor Circuit

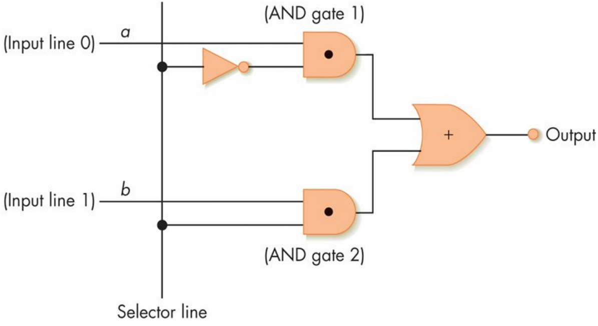


Figure 4.28
A Two-Input Multiplexor Circuit

Control Circuit Design: Data-routing Multiplexor

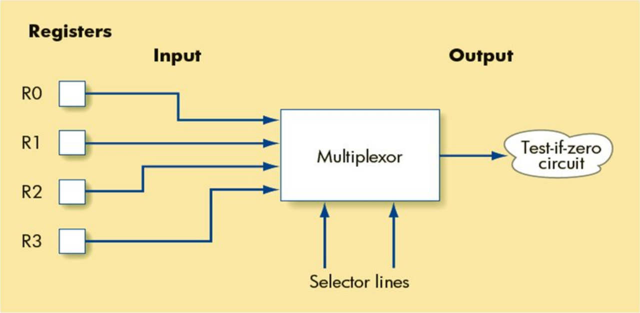
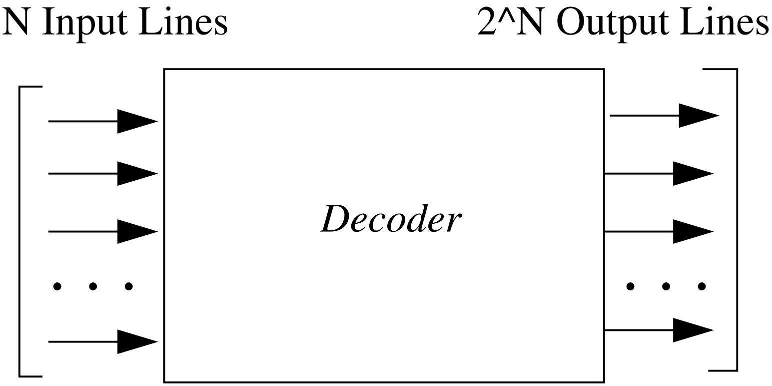


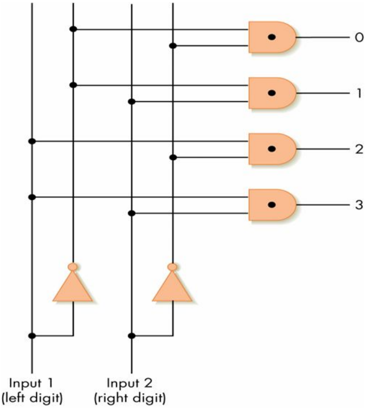
Figure 4.31 Example of the Use of a Multiplexor Circuit

Control Circuit Design: Decoder (Abstract)



Control Circuit Design: 2-to-4 Decoder Circuit

Figure 4.29
A 2-to-4 Decoder Circuit



Control Circuit Design: Op-code Decoder

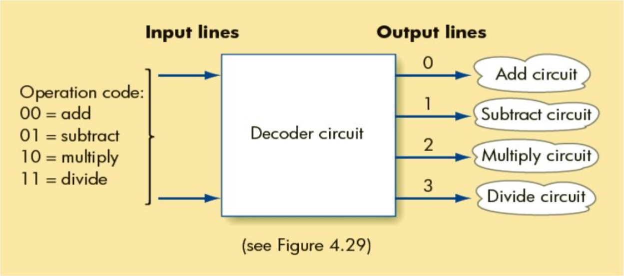


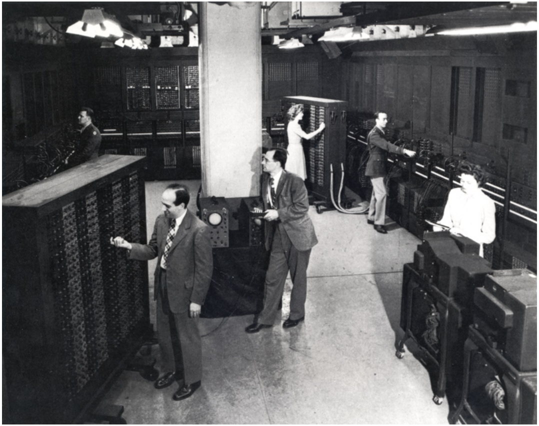
Figure 4.30 Example of the Use of a Decoder Circuit

Implementing Digital Circuits

Vacuum Tube (1904)

- Invented by John Ambrose Fleming (1849–1945).
- Basic component of early / mid 20th century electronics, e.g., radio, TV, radar.
- Require a lot of power and output a lot of heat; prone to burnout if used continuously.

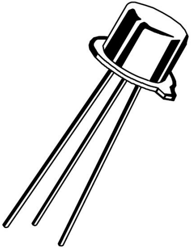
Implementing Digital Circuits (Cont'd)



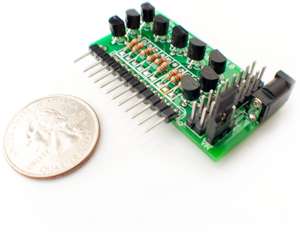
ENIAC (1945)

Used 18,000 vacuum tubes; did 5000 calculations / sec.

Implementing Digital Circuits (Cont'd)



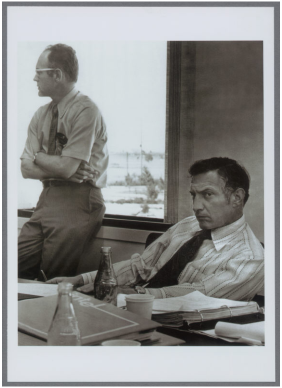
Transistor (1947)



Transistor Board

Traditional electronics businesses based on US East Coast. William Shockley (1910–1989) establishes first transistor manufacturer on West Coast (Palo Alto, CA) in 1955; trend continued by spinoff (Fairchild Semiconductor) in 1957.

Implementing Digital Circuits (Cont'd)

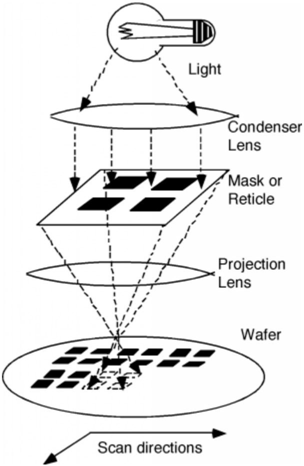


Gordon Moore (1929–) and Robert Noyce (1927–1990)

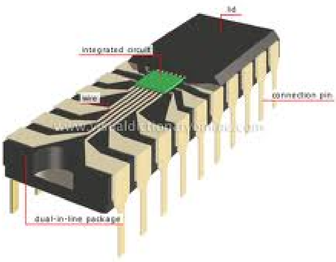
Co-founders of Fairchild Semiconductor; in 1959, Noyce develops planar process for creating integrated circuits.

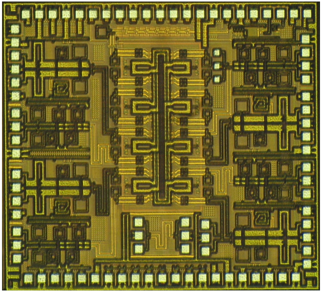
Implementing Digital Circuits (Cont'd)

- Silicon is a natural semiconductor whose electrical conductivity can be chemically modified by doping.
- In the planar process, electrical components based on silicon and deposited metals are "micro-printed" photographically in separate stacked layers on wafers of pure silicon.



Implementing Digital Circuits (Cont'd)





Integrated Circuit (IC) (1959) IC Internals ("Chip")

MOORE'S LAW (1965): 2X TRANSISTOR DENSITY EVERY 18 MONTHS

. . . And If You Liked This . . .

- MUN Computer Science courses on this area:
  - COMP 1002: Introduction to Logic for Computer Scientists
  - COMP 2003: Computer Architecture
  - COMP 4723: Introduction to Microprocessors
- MUN Computer Science professors teaching courses / doing research in in this area:
  - Miklos Bartha
  - Rod Byrne
  - Ashoke Deb
  - Antonina Kolokolova
  - Manrique Mata-Montero