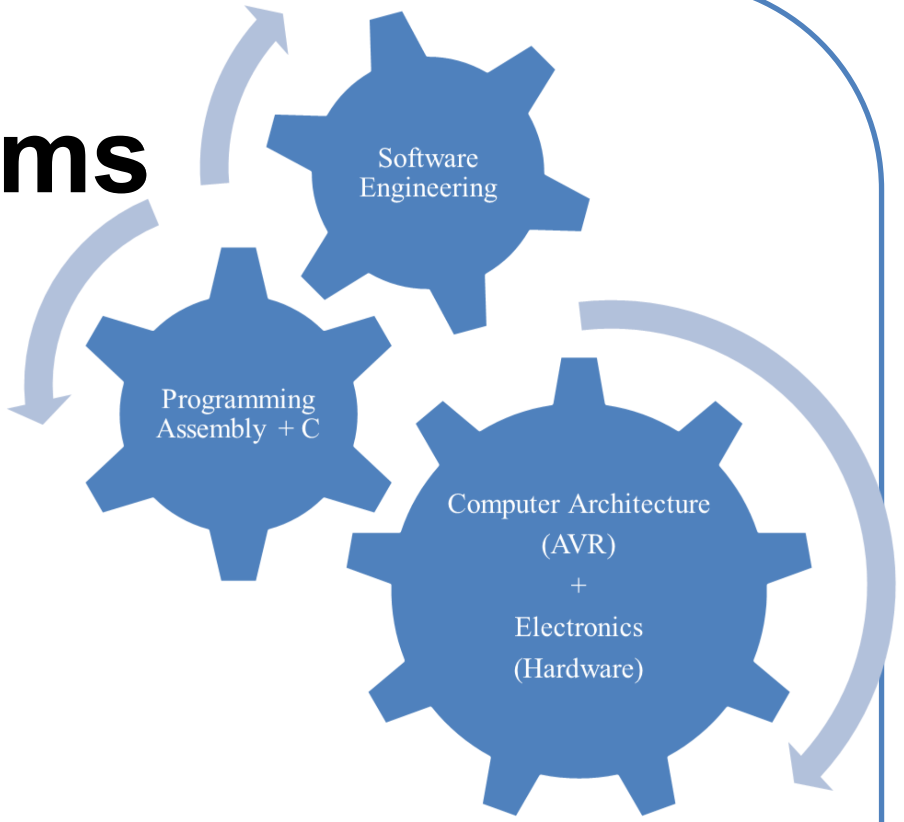
Embedded Systems



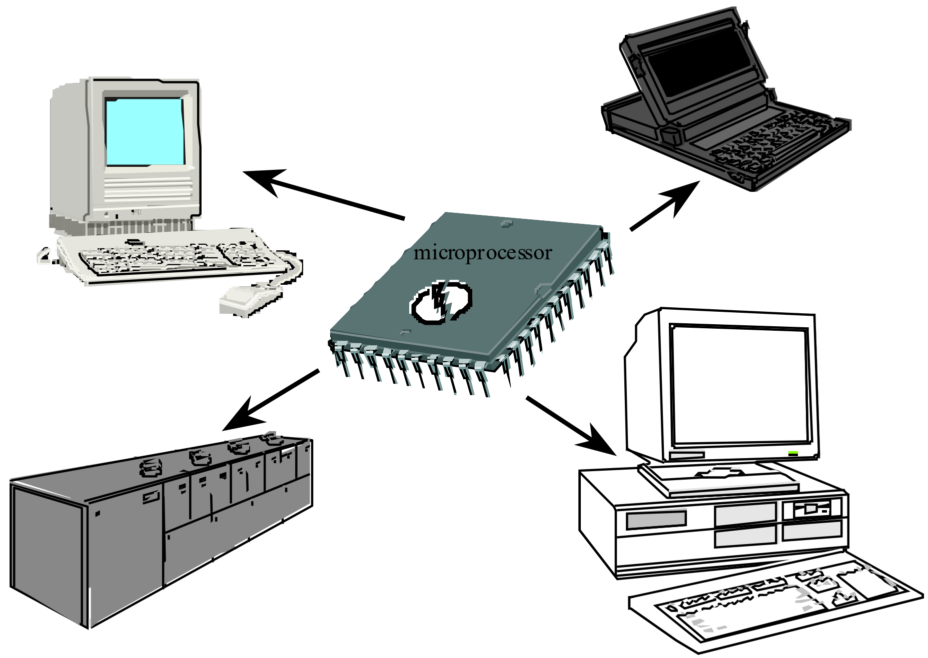
- Introduction to Embedded System.
- Review of Electronics Part (KCL, KVL, Parallel/Series Resistance, DAC and ADC).
- Review for Digital Systems (Binary Number, Logic, MOS Implantation, Computer Architecture).
- Intro to Programming Concepts (Structure and Concurrent).

What ? Embedded Systems

- **An embedded system is an electronic system that:**
  - includes a microcomputer embedded or hidden inside.
  - has software programmed into ROM.
  - has software that is not accessible to the user of the device
  - is configured to perform a specific dedicated application (software solves only a limited range of problems)

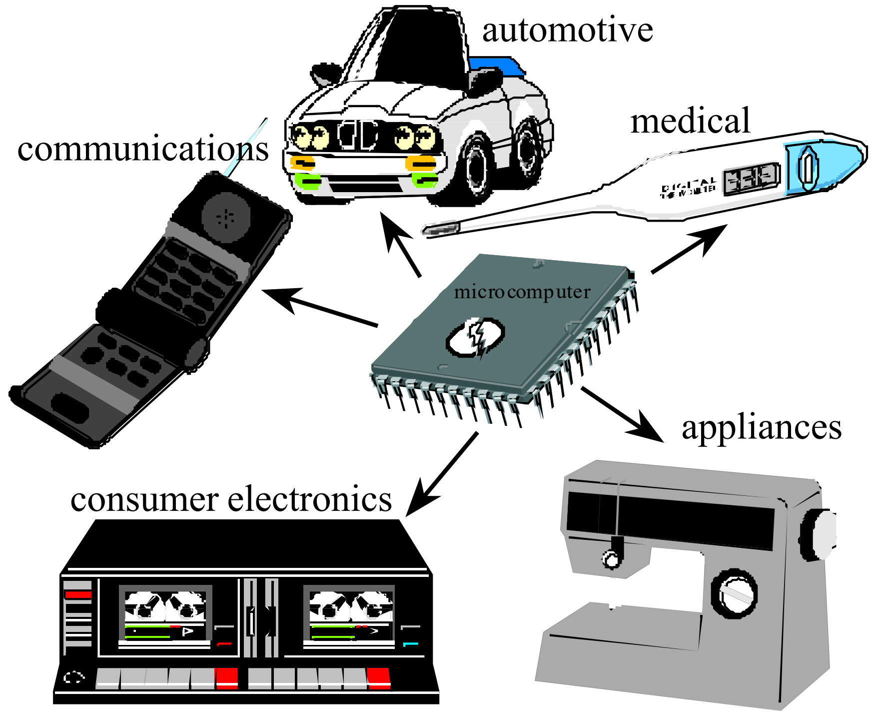
General Purpose System

- keyboard
- disk
- graphics display
- software useful for a wide variety of purposes
- software that can be changed by user



Embedded System

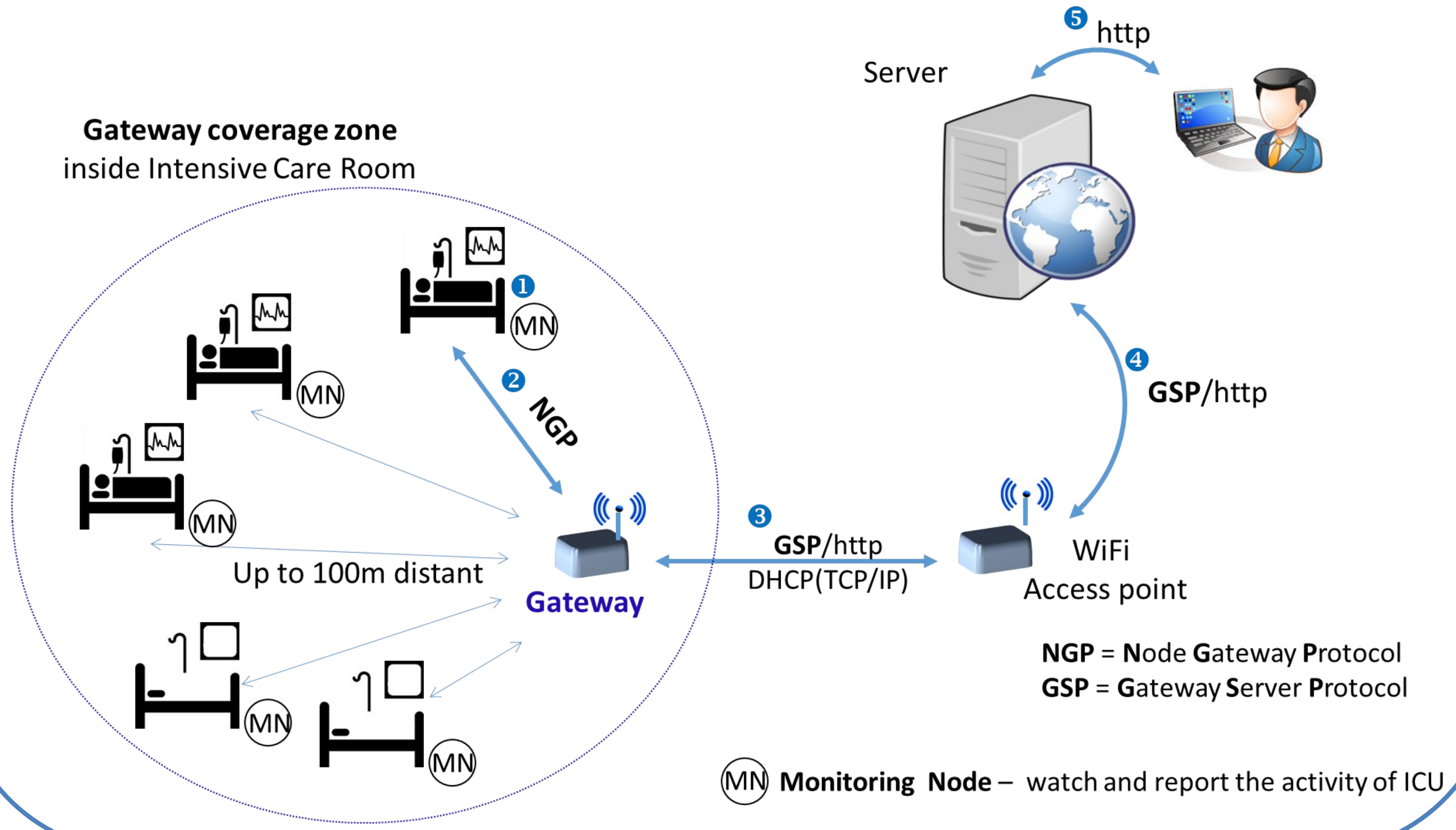
- Accepts inputs,
- Performs calculations
- Generates outputs
- Runs in “real time.”



The **internet of things** (IoT)

- Embedded systems (sensors and actuators).
- Internet.

ICU Utilization System



Embedded Systems “Big Ideas”

- **HW/SW Architecture**
  - Non processor centric view of architecture
  - Microcontroller, FPGA, analog circuits
- **Bowels of the “operating system”**
  - Specifically, the lower half of the OS
  - Concurrency, parallelism, synchronization
- **Real world design**
  - performance vs. cost tradeoffs, constraints
- **Analyzability**
  - how do you “know” that your drive-by-wire system will function correctly?
- **Application-level techniques**
  - Power Aware Programming

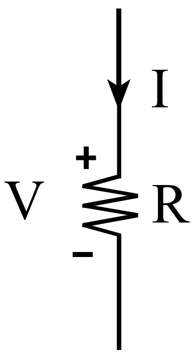
Review of Electronics

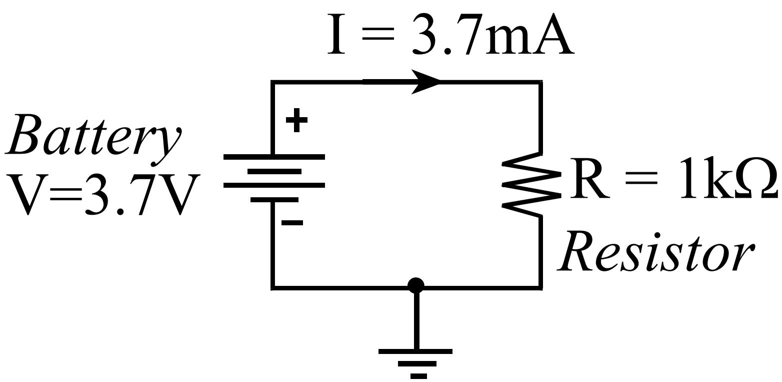
Ohm's Law

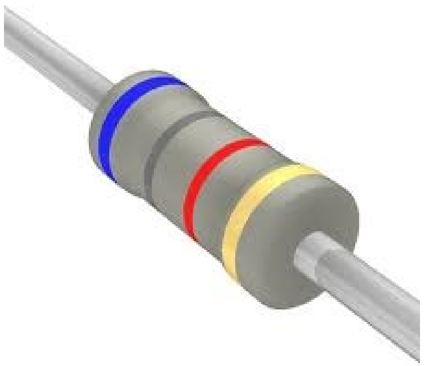
*V = I * R*   *Voltage = Current * Resistance*

I = V / R Current = Voltage / Resistance

R = V / I Resistance = Voltage / Current







*P = V * I*   *Power = Voltage * Current*

P = V2 / R   *Power = Voltage^2 / Resistance*

*P = I2 * R*   *Power = Current^2 * Resistance*

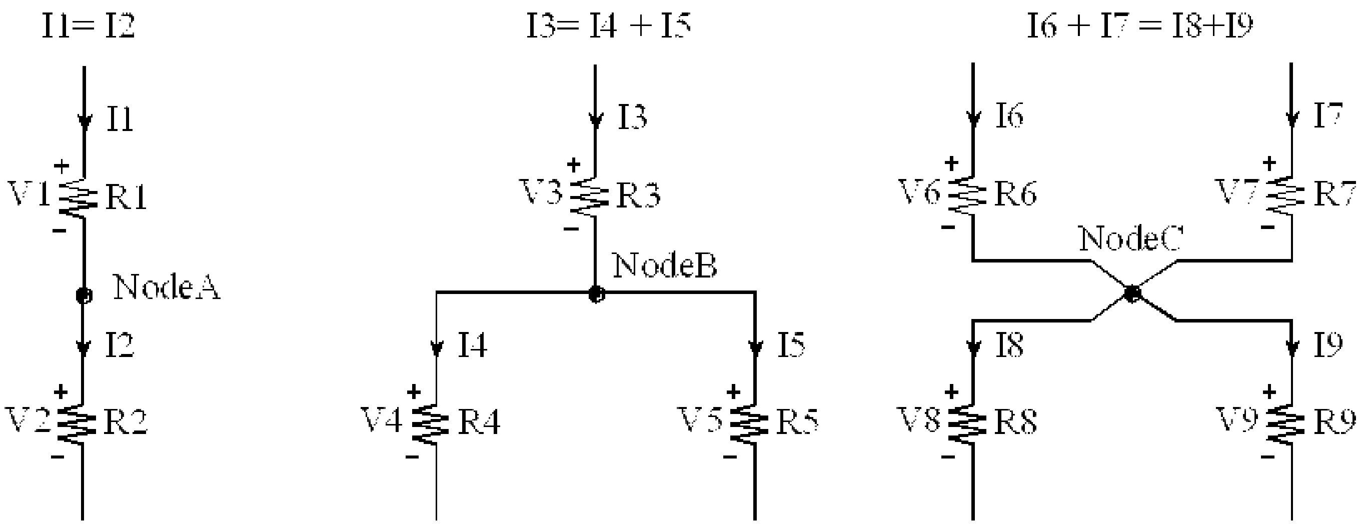
Review of Electronics (Cont...)

- Voltage: an electrical potential.
- Current: the flow of charge (electrons)
- Power: the rate of energy change.
- Energy: defines the amount of work that can be done
- Resistance: potential divided by flow

Review of Electronics (Cont...)

Kirchhoff's Current Law (KCL).

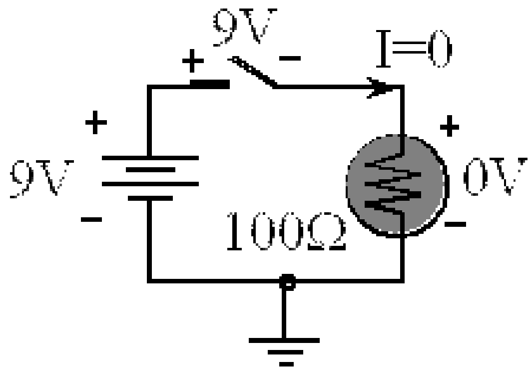
The sum of the currents **into a node** equal the sum of the currents **leaving the node**.

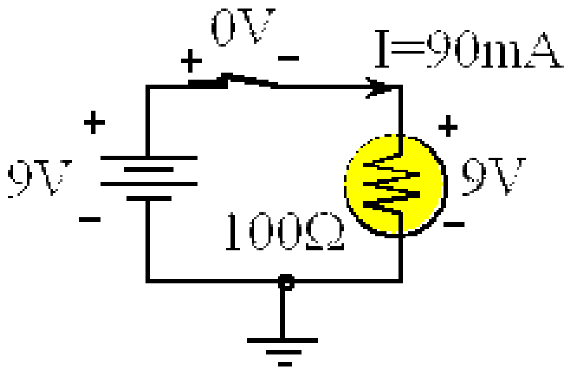


Review of Electronics (Cont...)

Kirchhoff's Voltage Law (KVL).

The sum of the voltages **around the loop is zero**.





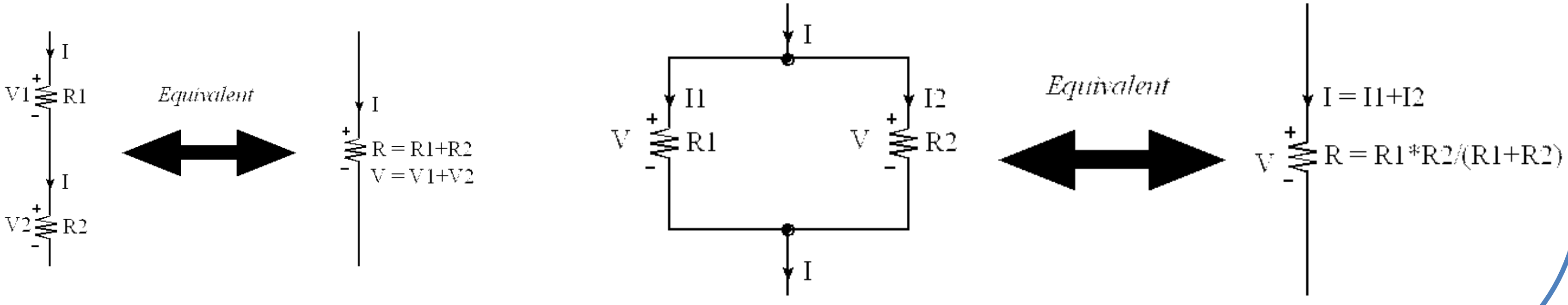
Review of Electronics (Cont...)

Series resistance

If resistor R1 is **in series** with resistor R2, this combination behaves like one resistor with a value **equal** to R1+R2

Parallel resistance

If resistor R1 is **in parallel** with resistor R2, this combination behaves like one resistor with a value **equal** to R1*R2/(R1+R2)



Review of Electronics (Cont...)

Consider this 3-bit digital to analog converter.

Define a 3-bit number n (0 to 7) which specifies the three switch positions.

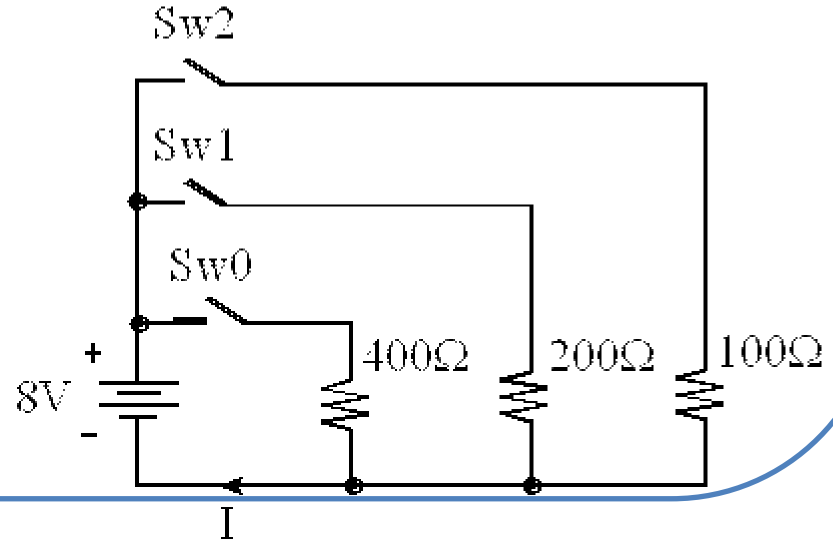
$n = 0$ means none are pushed. $n = 1$ means Sw0 is pushed.

$n = 2$ means Sw1 is pushed. $n = 3$ means Sw1 and Sw0 are pushed.

$n = 4$ means Sw2 is pushed. $n = 5$ means Sw2 and Sw0 are pushed.

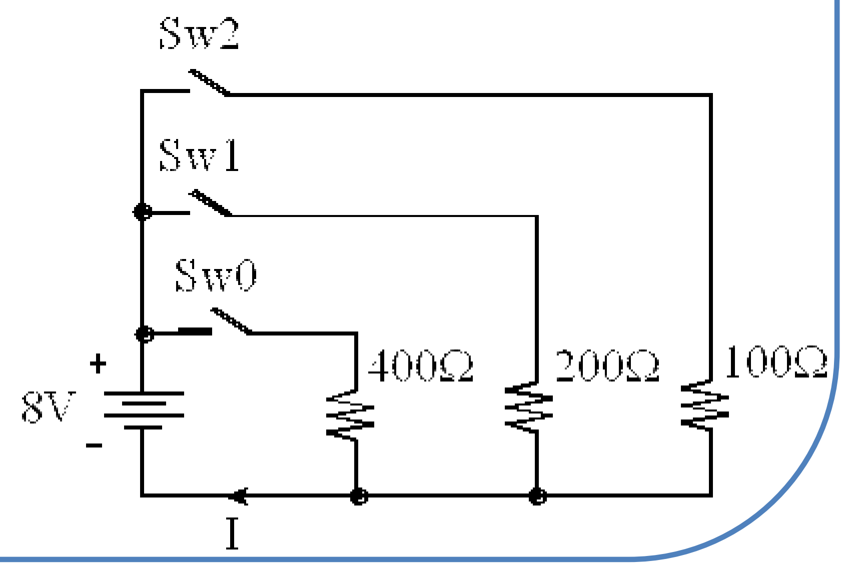
$n = 6$ means Sw2 and Sw1 are pushed. $n = 7$ means all are pushed.

Derive a relationship between the current I and the number n. Multiple choice

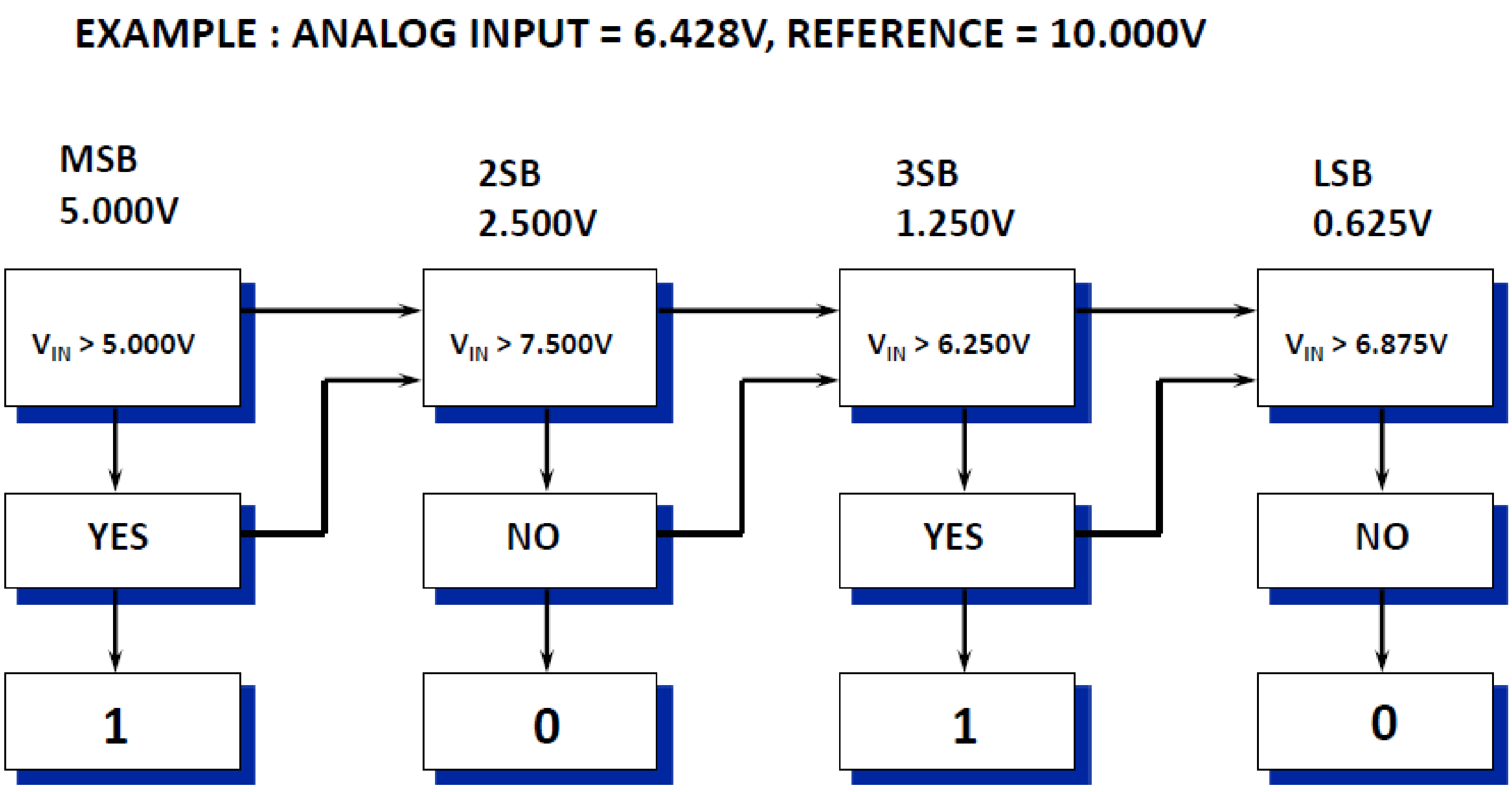


Review of Electronics (Cont...)

400	200	100	0	0	0	0	0	0	0	No- Current
400	200	100	1	0	0	400	0	0	400	0.02
400	200	100	0	1	0	0	200	0	200	0.04
400	200	100	1	1	0	400	200	0	133.3333	0.06
400	200	100	0	0	1	0	0	100	100	0.08
400	200	100	1	0	1	400	0	100	80	0.1
400	200	100	0	1	1	0	200	100	66.66667	0.12
400	200	100	1	1	1	400	200	100	57.14286	0.14



Review of Electronics (Cont...)



Assignment no. 1

In the design phase of the production line of fans. it required by the customer that fun to be controlled by three speeds (low, medium and high). your team leader asks you to identify the interface between the microcontroller and the fan motor. support your proposal by the circuit design.

Hint: motor driver, resistance ladder.

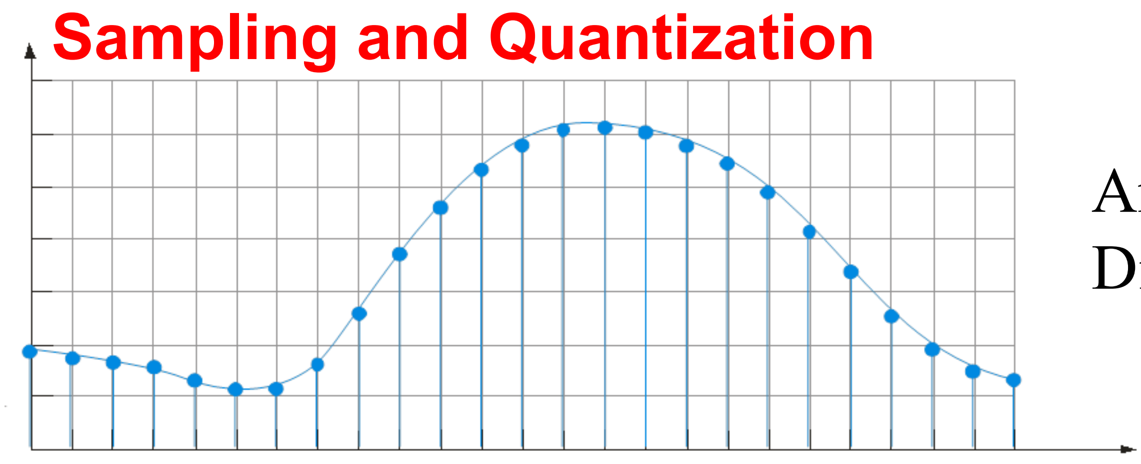
Embedded Systems

- ~~Introduction to Embedded System.~~
- ~~Review of Electronics Part (KCL, KVL, Parallel/Series Resistance, DAC and ADC).~~
- Review for Digital Systems (Binary Number, Logic, MOS Implantation, Computer Architecture).
- Intro to Programming Concepts (Structure and Concurrent).

Digital System

Analog vs. Digital

Most natural quantities (such as temperature, pressure, light intensity, …) are **analog** quantities that vary continuously.



Analog = continuous
Digital = discrete

Digital systems can process, store, and transmit data more efficiently but can only assign discrete values to each point.

Binary System

- **Two discrete values:**
  - 1's and 0's
  - 1, TRUE, HIGH
  - 0, FALSE, LOW
- **1 and 0:** voltage levels, rotating gears, fluid levels, etc.
- Digital circuits use **voltage** levels to represent 1 and 0
- ***Bit***: *B*inary dig*it*
- *N*-bit binary number
  - How many values? **2^N** - Range: **$[0, 2^N - 1]$**
  - Example: 3-digit binary number:
    - **$2^3 = 8$ possible values - Range: $[0, 7] = [000_2$ to $111_2]$**

8's column
4's column
2's column
1's column

$$1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13_{10}$$

one eight | one four | no two | one one

Hexadecimal Numbers

- Base 16
- Shorthand for binary

Hex Digit	Decimal Equivalent	Binary Equivalent
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Bits, Bytes, Nibbles…

byte

10010110

nibble

10010110

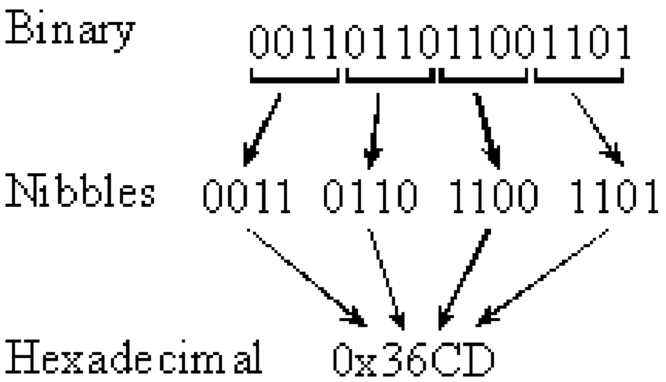
most significant bit

least significant bit

CEBF9AD7

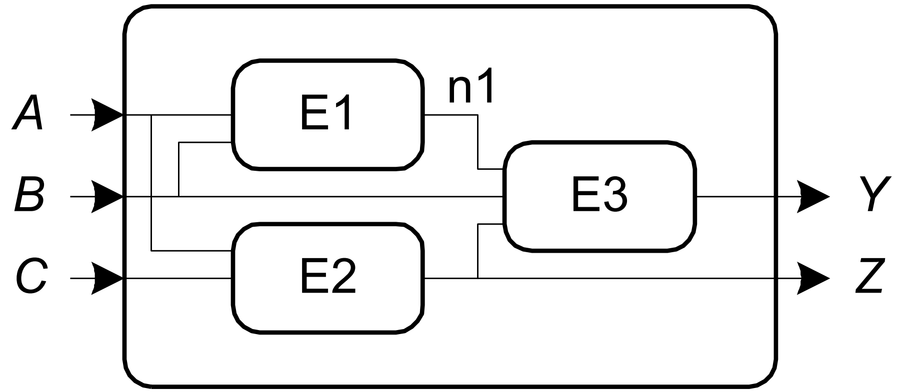
most significant byte

least significant byte



Digital Circuit

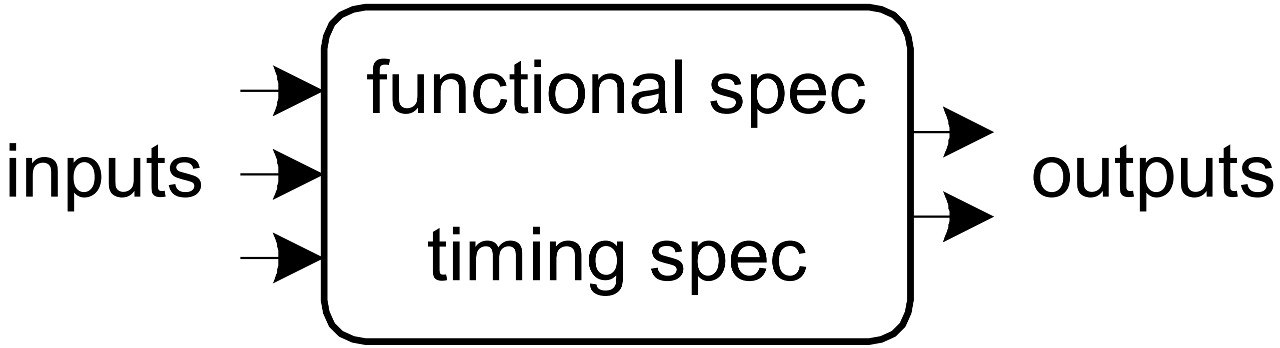
- Nodes
  - Inputs: *A*, *B*, *C*
  - Outputs: *Y*, *Z*
  - Internal: n1
- Circuit elements
  - E1, E2, E3
  - Each a circuit



Digital Circuit Specification

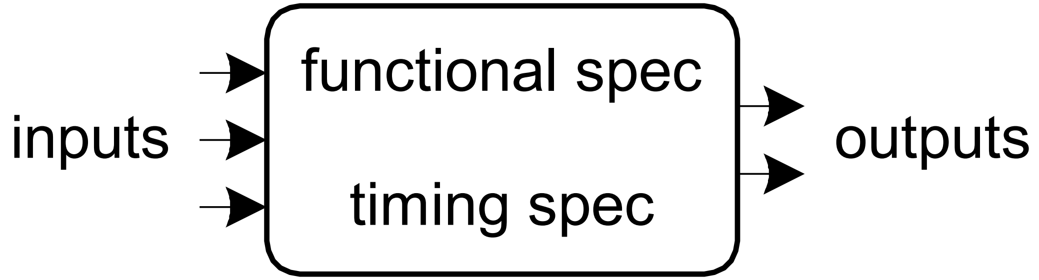
A logic circuit is composed of:

- Inputs
- Outputs
- Functional specification
- Timing specification



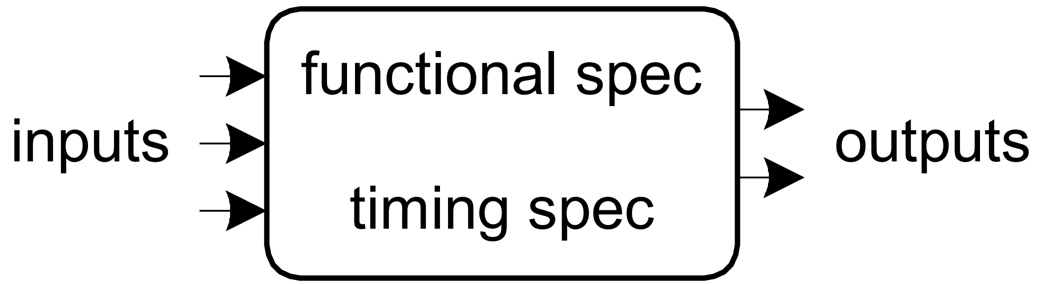
Digital Circuit Types

- **Combinational Logic**
  - Memoryless
  - Outputs determined by current values of inputs
- **Sequential Logic**
  - Has memory
  - Outputs determined by previous and current values of inputs



Digital Circuit Types

- **Combinational Logic**
  - Memoryless
  - Outputs determined by current values of inputs
- **Sequential Logic**
  - Has memory
  - Outputs determined by previous and current values of inputs



Definitions

- Complement: variable with a bar over it
  $\bar{A}, \bar{B}, \bar{C}$
- Literal: variable or its complement
  $A, \bar{A}, B, \bar{B}, C, \bar{C}$
- Implicant: product of literals
  $A\bar{B}\bar{C}, AC, BC$
- Minterm: product that includes all input variables
  $A\bar{B}C, \bar{A}\bar{B}C, ABC$
- Maxterm: sum that includes all input variables
  $(A+\bar{B}+C), (\bar{A}+B+\bar{C}), (\bar{A}+B+C)$

Sum-of-Products (SOP) Form

- All equations can be written in SOP form
- Each row has a **minterm**
- A minterm is a product (AND) of literals
- Each minterm is TRUE for that row (and only that row)
- Form function by **ORing minterms where the output is TRUE**
- Thus, a sum (OR) of products (AND terms)

A	*B*	*Y*	minterm	minterm name
0	0	0	$\overline{A}\ \overline{B}$	m_0
0	1	1	$\overline{A}\ B$	m_1
1	0	0	$A\ \overline{B}$	m_2
1	1	1	$A\ B$	m_3

$$Y = \mathrm{F}(A, B) = \overline{\mathrm{A}}\mathrm{B} + \mathrm{AB} = \Sigma(1, 3)$$

Product-of-Sums (POS) Form

- All Boolean equations can be written in POS form
- Each row has a **maxterm**
- A maxterm is a sum (OR) of literals
- Each maxterm is FALSE for that row (and only that row)
- Form function by **ANDing the maxterms for which the output is FALSE**
- Thus, a product (AND) of sums (OR terms)

A	B	Y	maxterm	maxterm name
0	0	0	$A + B$	M_0
0	1	1	$A + \overline{B}$	M_1
1	0	0	$\overline{A} + B$	M_2
1	1	1	$\overline{A} + \overline{B}$	M_3

$$Y = F(A, B) = (A + B)(\overline{A} + B) = \Pi(0, 2)$$

Boolean Equations Example

- You are going to the cafeteria for lunch
  - You won't eat lunch ($\overline{E}$)
  - If it's not open ($\overline{O}$) or
  - If they only serve chicken (C)
- Write a truth table for determining if you will eat lunch (E).

O	C	E
0	0	0
0	1	0
1	0	1
1	1	0

SOP & POS Form

- SOP – sum-of-products

O	C	E	minterm
0	0	0	$\overline{O}\ \overline{C}$
0	1	0	$\overline{O}\ C$
1	0	1	$O\ \overline{C}$
1	1	0	$O\ C$

$$E = O\overline{C}$$
$$= \Sigma(2)$$

- POS – product-of-sums

O	C	E	maxterm
0	0	0	$O + C$
0	1	0	$O + \overline{C}$
1	0	1	$\overline{O} + C$
1	1	0	$\overline{O} + \overline{C}$

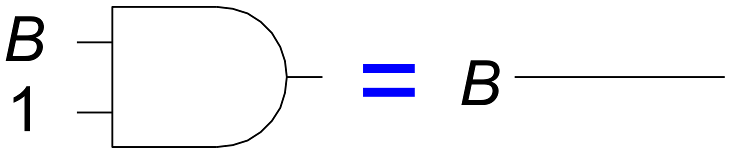
$$E = (O + C)(O + \overline{C})(\overline{O} + \overline{C})$$
$$= \Pi(0, 1, 3)$$

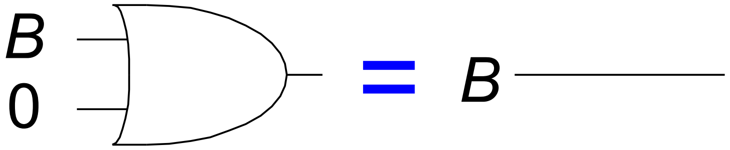
Boolean Algebra

- Axioms and theorems to **simplify** Boolean equations
- Like regular algebra, but simpler: variables have only two values (1 or 0)
- **Duality** in axioms and theorems:
  - ANDs and ORs, 0's and 1's interchanged

T1: Identity Theorem

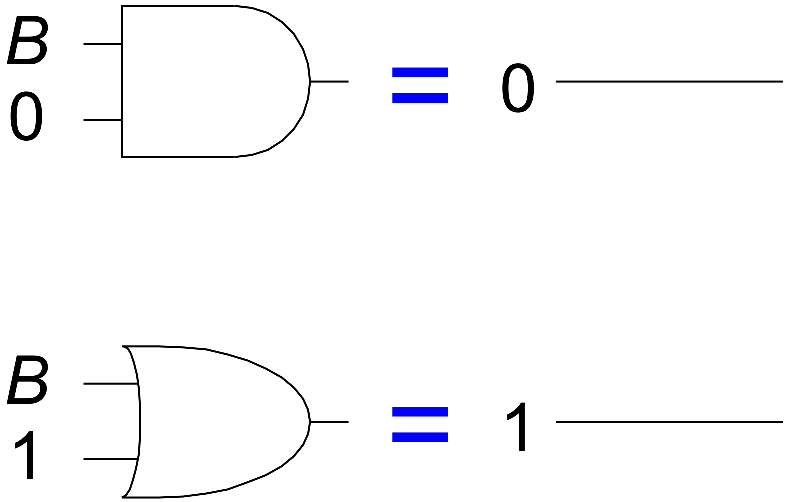
- $B \cdot 1 = B$
- $B + 0 = B$





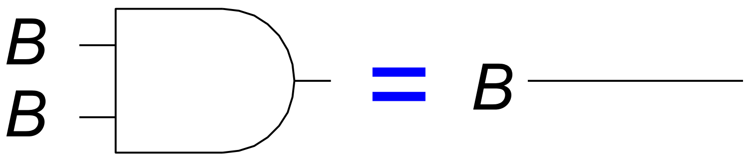
T2: Null Element Theorem

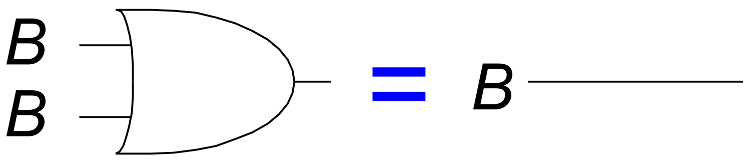
- $B \bullet 0 = 0$
- $B + 1 = 1$



T3: Idempotency Theorem

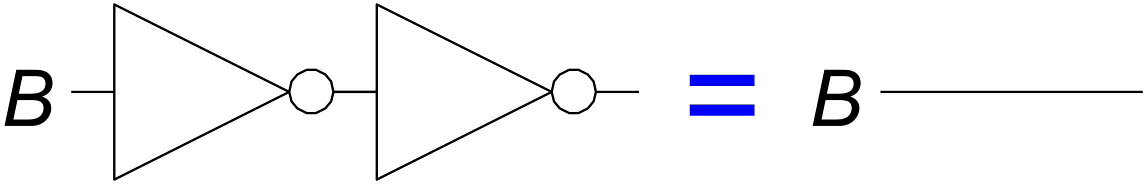
- $B \bullet B = B$
- $B + B = B$





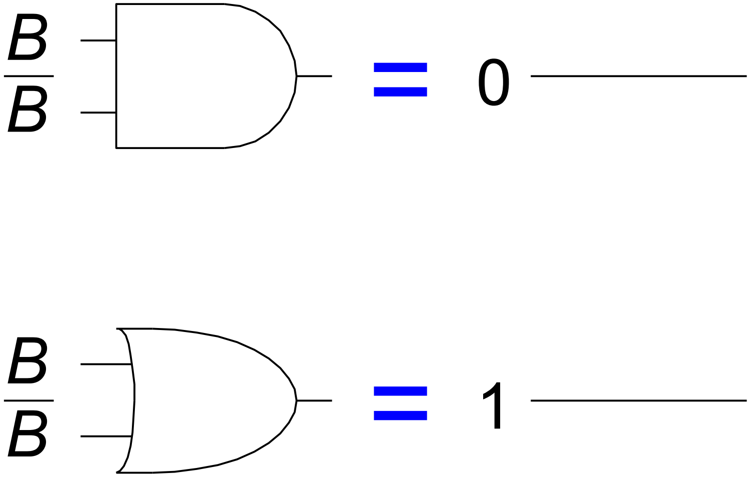
T4: Identity Theorem

- $\overline{\overline{B}} = B$



T5: Complement Theorem

- $B \bullet \overline{B} = 0$
- $B + \overline{B} = 1$



Boolean Theorems Summary

	Theorem		Dual	Name
T1	$B \bullet 1 = B$	T1′	$B + 0 = B$	Identity
T2	$B \bullet 0 = 0$	T2′	$B + 1 = 1$	Null Element
T3	$B \bullet B = B$	T3′	$B + B = B$	Idempotency
T4		$\overline{\overline{B}} = B$		Involution
T5	$B \bullet \overline{B} = 0$	T5′	$B + \overline{B} = 1$	Complements

Boolean Theorems of Several Vars

	Theorem		Dual	Name
T6	$B \bullet C = C \bullet B$	T6′	$B + C = C + B$	Commutativity
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	T7′	$(B + C) + D = B + (C + D)$	Associativity
T8	$(B \bullet C) + B \bullet D = B \bullet (C + D)$	T8′	$(B + C) \bullet (B + D) = B + (C \bullet D)$	Distributivity
T9	$B \bullet (B + C) = B$	T9′	$B + (B \bullet C) = B$	Covering
T10	$(B \bullet C) + (B \bullet \overline{C}) = B$	T10′	$(B + C) \bullet (B + \overline{C}) = B$	Combining
T11	$(B \bullet C) + (\overline{B} \bullet D) + (C \bullet D) = B \bullet C + \overline{B} \bullet D$	T11′	$(B + C) \bullet (\overline{B} + D) \bullet (C + D) = (B + C) \bullet (\overline{B} + D)$	Consensus
T12	$\overline{B_0 \bullet B_1 \bullet B_2 \ldots} = (\overline{B_0} + \overline{B_1} + \overline{B_2} \ldots)$	T12′	$\overline{B_0 + B_1 + B_2 \ldots} = (\overline{B_0} \bullet \overline{B_1} \bullet \overline{B_2})$	De Morgan's Theorem

Note: T8' differs from traditional algebra: OR (+) distributes over AND (•)

Simplifying Boolean Equations

Example 1:

$$Y = AB + \overline{A}B$$

$$= B(A + \overline{A}) \quad \text{T8}$$

$$= B(1) \quad \text{T5'}$$

$$= B \quad \text{T1}$$

Simplifying Boolean Equations

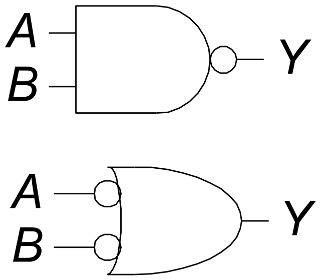
Example 2:

$$Y = A(AB + ABC)$$

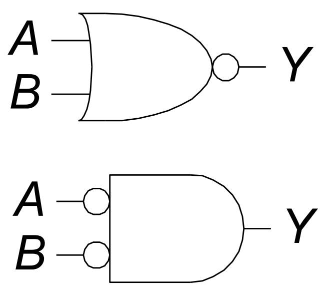
$= A(AB(1 + C))$	T8
$= A(AB(1))$	T2’
$= A(AB)$	T1
$= (AA)B$	T7
$= AB$	T3

DeMorgan's Theorem

- $Y = \overline{AB} = \overline{A} + \overline{B}$

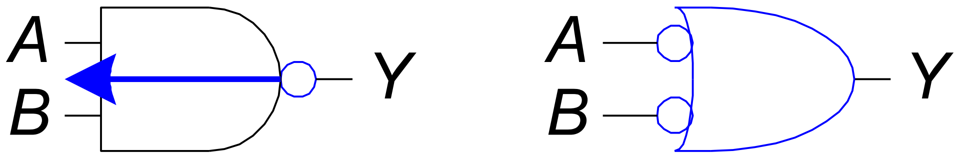


- $Y = \overline{A + B} = \overline{A} \bullet \overline{B}$



Bubble Pushing

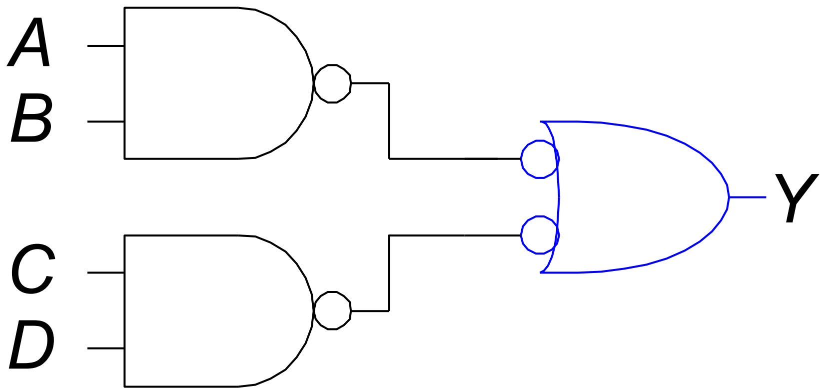
- **Backward:**
  - Body changes
  - Adds bubbles to inputs



- **Forward:**
  - Body changes
  - Adds bubble to output

Bubble Pushing

- What is the Boolean expression for this circuit?



$$Y = AB + CD$$

Boolean Algebra

- A & B = B & A … Commutative Law
- A | B = B | A … Commutative Law
- (A & B) & C = A & (B & C) … Associative Law
- (A | B) | C = A | (B | C) … Associative Law
- (A | B) & C = (A & C) | (B & C) … Distributive Law
- (A & B) | C = (A | C) & (B | C) … Distributive Law
- ~(A | B) = (~A) & (~B) … De Morgan’s Theorem
- ~(A & B) = (~A) | (~B) ... De Morgan’s Theorem

A & 0 = 0 … Identity of 0
A | 0 = A … Identity of 0
A & 1 = A … Identity of 1
A | 1 = 1 … Identity of 1

A | A = A … Property of OR
A | (~A) = 1 … Property of OR
A & A = A … Property of AND
A & (~A) = 0 … Property of AND
~(~A) = A … Inverse

Operators (Bitwise / logical)

A	B	C	A & (B | C)	A && (B || C)
01	01	11	01	True
00	01	11	00	False

Signed Binary Numbers

- Sign/Magnitude Numbers
  - 1 sign bit, *N*-1 magnitude bits
  - Positive number: sign bit = 0 , Negative number: sign bit = 1
  - Example, 4-bit sign/mag representations of ± 6:

  +6 = **0110**

  - 6 = **1110**

  - Range of an N-bit sign/magnitude number: $[-(2^{N-1}-1), 2^{N-1}-1]$
- Two's Complement Numbers
  - The most significant bit still indicates the sign (1 = negative, 0 = positive)
  - Range of an N-bit two's comp number: **[-(2N-1), 2N-1-1]**

    1. **1001**
    2. **+ 1**

    $1010_2 = -6_{10}$

```
 111
  0110
+ 1010
------
 10000
```

Signed Binary Numbers (Cont…)

- Sign Extension
  - Sign bit copied to msb's
  - Number value is same
- **Example 1:**
  - 4-bit representation of 3 = 0011
  - 8-bit sign-extended value: 00000011
- **Example 2:**
  - 4-bit representation of -5 = 1011
  - 8-bit sign-extended value: 11111011

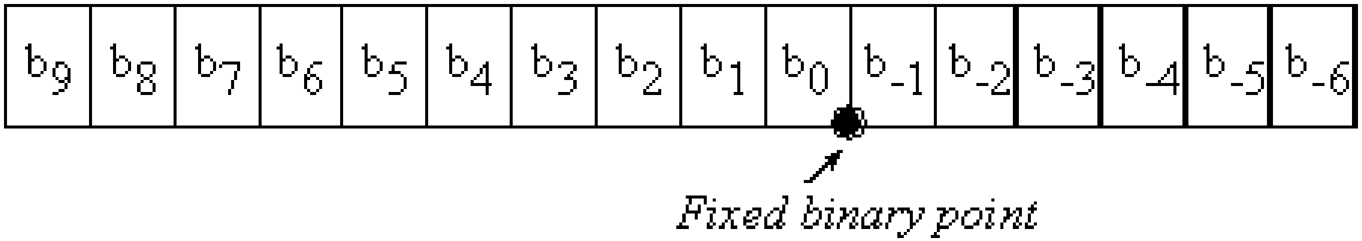
Signed Binary Numbers (Cont…)

Number System	Range
Unsigned	$[0, 2^N-1]$
Sign/Magnitude	$[-(2^{N-1}-1), 2^{N-1}-1]$
Two's Complement	$[-2^{N-1}, 2^{N-1}-1]$

For example, 4-bit representation:

Value	Unsigned	Two's Complement	Sign/Magnitude
-8		1000	
-7		1001	1111
-6		1010	1110
-5		1011	1101
-4		1100	1100
-3		1101	1011
-2		1110	1010
-1		1111	1001
0	0000	0000	0000 1000
1	0001	0001	0001
2	0010	0010	0010
3	0011	0011	0011
4	0100	0100	0100
5	0101	0101	0101
6	0110	0110	0110
7	0111	0111	0111
8	1000		
9	1001		
10	1010		
11	1011		
12	1100		
13	1101		
14	1110		
15	1111		

Fixed Point Binary Numbers



- **fixed-point numbers**: express values to the computer non-integer values.
- **A fixed-point number** contains two parts: variable integer, called *I*. fixed constant, called the resolution .
- The fixed constant will NOT be stored on the computer. The fixed constant is something we keep track of while designing the software operations. !!!
- The **precision** of a number system is the total number of distinguishable values that can be represented.
- The precision of a fixed-point number is the number of bits used to store the variable integer.

Binary fixed-point value = $I \bullet 2^n$

Fixed Point Binary Numbers

01101100

0110.1100

$2^2 + 2^1 + 2^{-1} + 2^{-2} = 6.75$

Addition

- Decimal

$$
\begin{array}{r}
{\color{blue}11\ \ \ } \leftarrow \text{carries} \\
3734 \\
+\ 5168 \\
\hline
8902
\end{array}
$$

- Binary

$$
\begin{array}{r}
{\color{blue}11\ \ \ } \leftarrow \text{carries} \\
1011 \\
+\ 0011 \\
\hline
1110
\end{array}
$$

- Digital systems operate on a **fixed number of bits**
- Overflow: when result is too big to fit in the available number of bits

$$
\begin{array}{r}
{\color{blue}111\ } \\
1011 \\
+\ 0110 \\
\hline
10001
\end{array}
$$

Overflow!

[Ariane-5 Rocket Explosion (2002)]

Multiplication

Decimal		Binary
230	multiplicand	0101
x 42	multiplier	x 0111
460 + 920	partial products	0101 0101 0101 + 0000
9660	result	0100011

230 x 42 = 9660

5 x 7 = 35

- Partial products formed by multiplying a single digit of the multiplier with multiplicand
- Shifted partial products summed to form result