DIGITAL SYSTEMS

I Real world systems and processes

A. Mostly continuous (at the macroscopic level): time, acceleration, chemical reactions
B. Sometimes discrete: quantum states, mass (# of atoms)
C. Mathematics to represent physical systems is continuous (calculus)
D. Mathematics for number theory, counting, approximating physical systems can be discrete

II Representation of continuous information

A. Continuous—represented analogously as a value of a continuously variable parameter
   1. position of a needle on a meter
   2. rotational angle of a gear
   3. amount of water in a vessel
   4. electric charge on a capacitor
B. Discrete—digitized as a set of discrete values corresponding to a finite number of states
   1. digital clock
   2. painted pickets
   3. on/off, as a switch

III Representation of continuous processes

A. Analogous to the process itself
   1. Great Brass Brain—a geared machine to simulate the tides
   2. Slide rule—an instrument which does multiplication by adding lengths which correspond to the logarithms of numbers.
   3. Differential analyzer (von Neumann)—variable-size friction wheels to simulate the behavior of differential equations
   4. Electronic analog computers—circuitry connected to simulate differential equations
   5. Phonograph record—wiggles in grooves to represent sound oscillations
   6. Electric clocks
   7. Mercury thermometers

B. Discretized to represent the process
   1. Finite difference formulations
   2. Digital clocks
   3. Music CD's

IV Manipulation

A. Analog
   1. adding the length-equivalents of logarithms to obtain a multiply e.g., a slide-rule
   2. adjusting the volume on a stereo
   3. sliding a weight on a balance-beam scale
   4. adding charge to an electrical capacitor

B. Discrete
   1. counting—push-button counters
   2. digital operations—mechanical calculators
   3. switching—open/closing relays
   4. logic circuits—true/false determination

V Analog vs. Discrete

Note: "Digital" is a form of representation for discrete

A. Analog
1. infinitely variable--information density high
2. limited resolution--to what resolution can you read a meter?
3. irrecoverable data degradation--sandpaper a vinyl record

B. Discrete/Digital
1. limited states--information density low
   e.g., one decimal digit can represent only one of ten values
2. arbitrary resolution--keep adding states (or digits)
3. mostly recoverable data degradation, e.g., if information is encoded as painted/not-painted pickets, repainting can perfectly restore data

VI Digital systems

A. decimal--not so good, because there are few 10-state devices that could be used to store information fingers. . .?
B. binary --excellent for hardware; lots of 2-state devices: switches, lights, magnetics
   --poor for communication: 2-state devices require many digits to represent values with reasonable resolution
   --excellent for logic systems whose states are true and false
C. octal --base 8: used to conveniently represent binary data; almost as efficient as decimal
D. hexadecimal--base 16: more efficient than decimal; more practical than octal because of binary digit groupings in computers

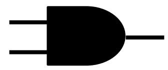
VII Binary logic and arithmetic

A. Background
1. George Boole(1854) linked arithmetic, logic, and binary number systems by showing how a binary system could be used to simplify complex logic problems
2. Claude Shannon(1938) demonstrated that any logic problem could be represented by a system of series and parallel switches; and that binary addition could be done with electric switches
3. Two branches of binary logic systems
   a) Combinatorial—in which the output depends only on the present state of the inputs
   b) Sequential—in which the output may depend on a previous state of the inputs, e.g., the “flip-flop” circuit

B. Logic operations and truth tables

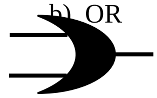
1. Logic gates:

	input	input	output
a) AND	0	0	0
	0	1	0
	1	0	0
	1	1	1

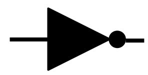


b) OR 0 0 0

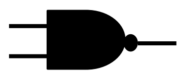
	0	1	1
	1	0	1
	1	1	1



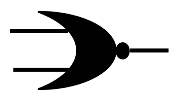
c) NOT	0		1
	1		0



d) NAND	0	0	1
	0	1	1
	1	0	1
	1	1	0



e) NOR	0	0	1
0	1	0	
1	0	0	
1	1	0	



f) XOR, XNOR

2. Boolean algebra (* = AND; + = OR; ‘ = NOT)

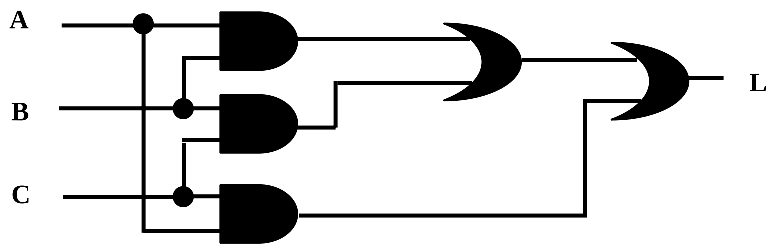
AND rules	OR rules	
A*A = A	A +A = A	
A*A' = 0	A +A' = 1	
0*A = 0	0+A = A	
1*A = A	1 +A = 1	
A*B = B*A	A + B = B+A	
A*(B*C) = (A*B)*C	A+(B+C) = (A+B)+C	
A(B+C) = A*B+B*C	A+B*C = (A+B)*(A+C)	
A'*B' = (A+B)'	A'+B' = (A*B)'	(DeMorgan’s theorem)

C. Uses

1. Logic problems: e.g., George is elected chairman only if he gets a majority of the three votes

A	B	C	L
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

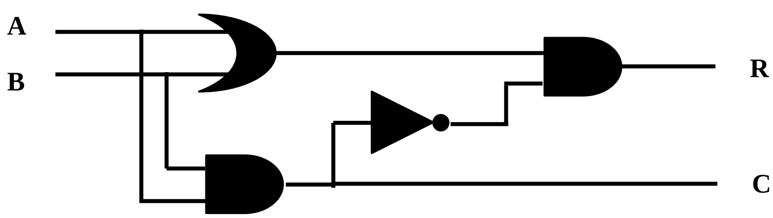
A*B*C' + A*B'*C + A'*B*C + A*B*C =
A*B*(C' + C) + B*C*(A' + A) + A*C*(B' + B) =
A*B + B*C + A*C = L



2. Binary arithmetic: e.g., adding two binary digits

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

A*B = C
(A + B)*(A*B)′ = R



3. Control systems: e.g., car will start only if doors are locked, seat belts are on, key is turned

D	S	K	I
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

D*S*K = I