ELEC 2200 Digital Logic Circuits

Charles E. Stroud, Professor
Dept. of Electrical & Computer Engineering
Office: 325 Broun Hall
Email: cestroud@eng.auburn.edu

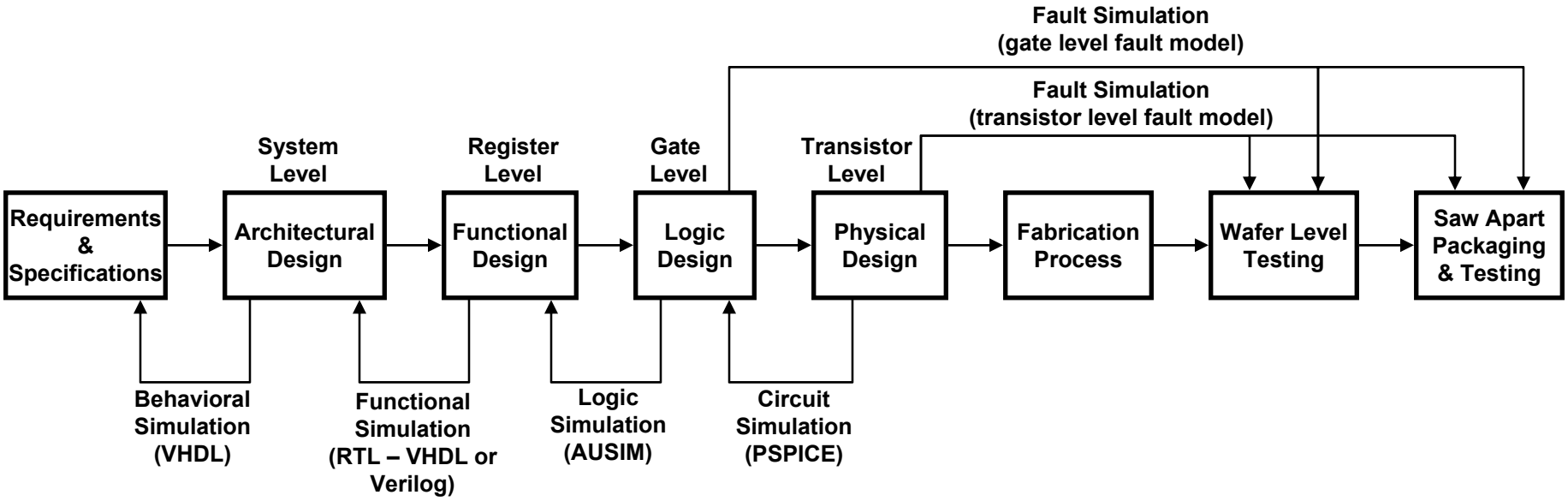
Text: Digital Logic Circuit Analysis & Design
Nelson, Nagle, Irwin & Carroll, Prentice Hall, ‘95
Class web page: www.eng.auburn.edu/~strouce

Some Basic Digital System Concepts

- Levels of design abstraction & hierarchy
  - System (behavioral) level – highest level
  - Register level – widely used for design in industry today
  - Gate level – level we will deal with most in this class
  - Transistor level – lowest level
- Top-down design
  - Begins at system level & moves toward transistor level
    - Typical way complex digital systems are designed in industry
  - CAD tools can synthesize lower levels of design abstraction from higher level descriptions
    - So what is the point of logic design?
      - CAD tools are not a magic wand, they don't design the circuit for you!

Example of Digital System Design

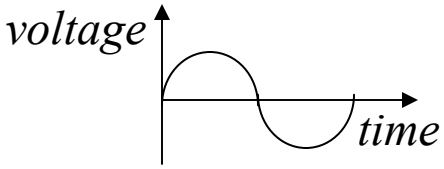
- The integrated circuit design process

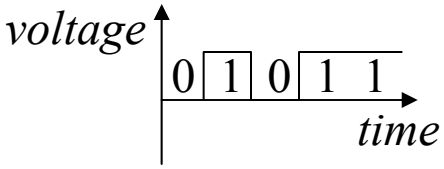


Hierarchical System Design

- Hierarchy is everywhere
  - Systems consist of units
  - Units consist of printed circuit boards (PCBs)
  - PCBs consist of integrated circuits (ICs)
  - ICs consist of logic gates
  - Logic gates consist of transistors
- Allows us to partition big designs into manageable components
- Once the circuit design works, why redesign it?
  - Instead, reuse it through hierarchical design
  - Reduces design time and design errors

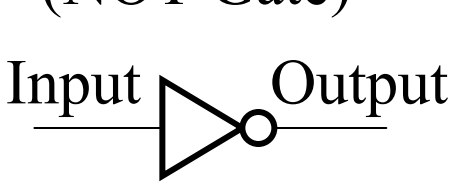
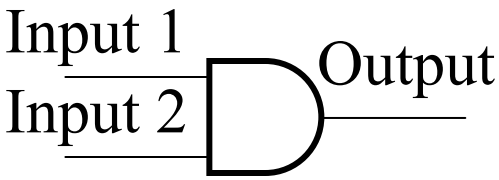
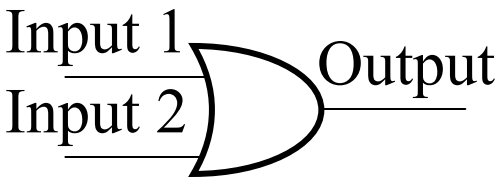
Analog vs. Digital Systems





- Analog
  - Continuous time-varying voltages and/or currents
  - Basic elements of analog circuits:
    - Resistors
    - Capacitors
    - Inductors
    - Transistors
- Digital
  - Discrete signals sampled in time
  - Two possible values
    - 0V, low, false (logic 0)
    - 5V, high, true (logic 1)
  - Basic elements of digital circuits:
    - Logic gates: AND, OR, NOT

Elementary Logic Gates

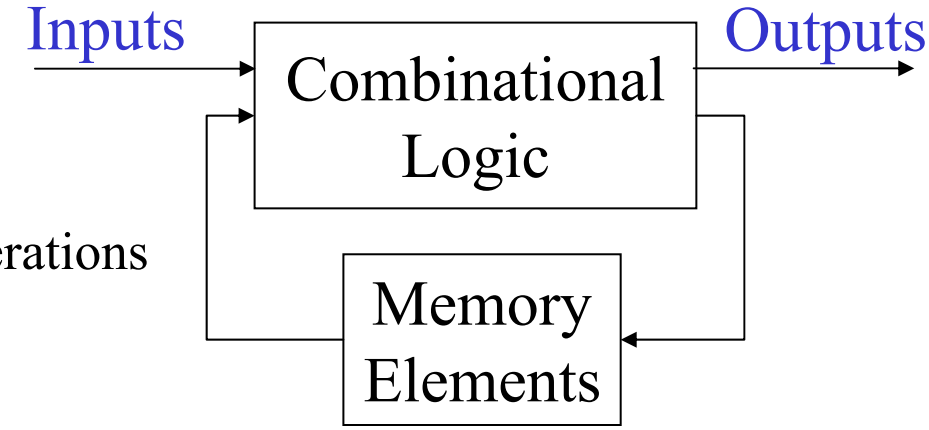
	Inverter (NOT Gate)	AND Gate	OR Gate
Name			
Symbol			
Truth Table	In | Out 0 | 1 1 | 0	In 1 | In 2 | Out 0 | 0 | 0 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1	In 1 | In 2 | Out 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 1
Logic Equation	Out = In’	Out = In1 • In2	Out = In1 + In2

Advantages of Digital Systems

- Reproducible results
- Relative ease of design
- Flexibility and functionality
- High speed
- Small size
- Low cost
- Low power
- Steadily advancing technology
- Programmable logic devices

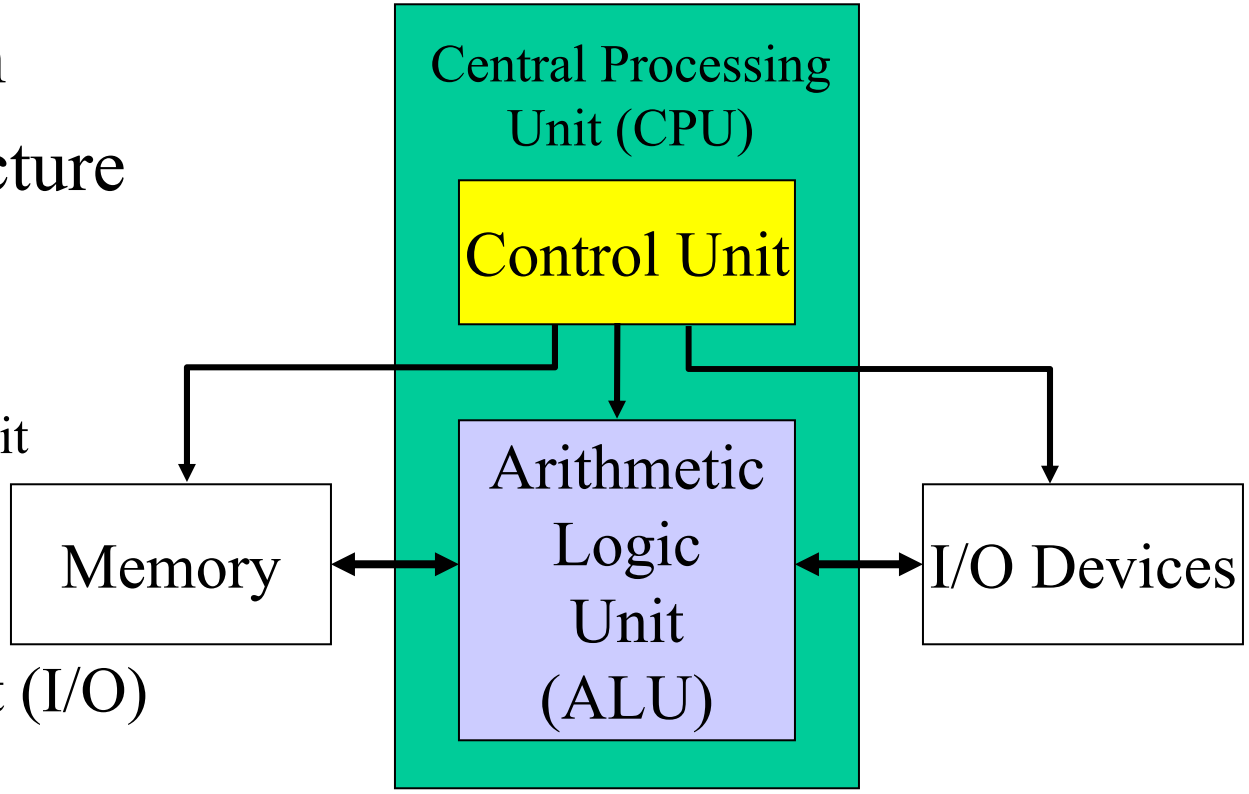
Digital Systems

- Most general model of a digital system
  - Often referred to as *General Sequential Logic*
    - aka *Huffman model*
  - Consists of:
    - Combinational logic
      - Performs logical operations
    - Memory elements
      - Stores data
    - These two items will be the focus of this course
      - Beginning with combinational logic

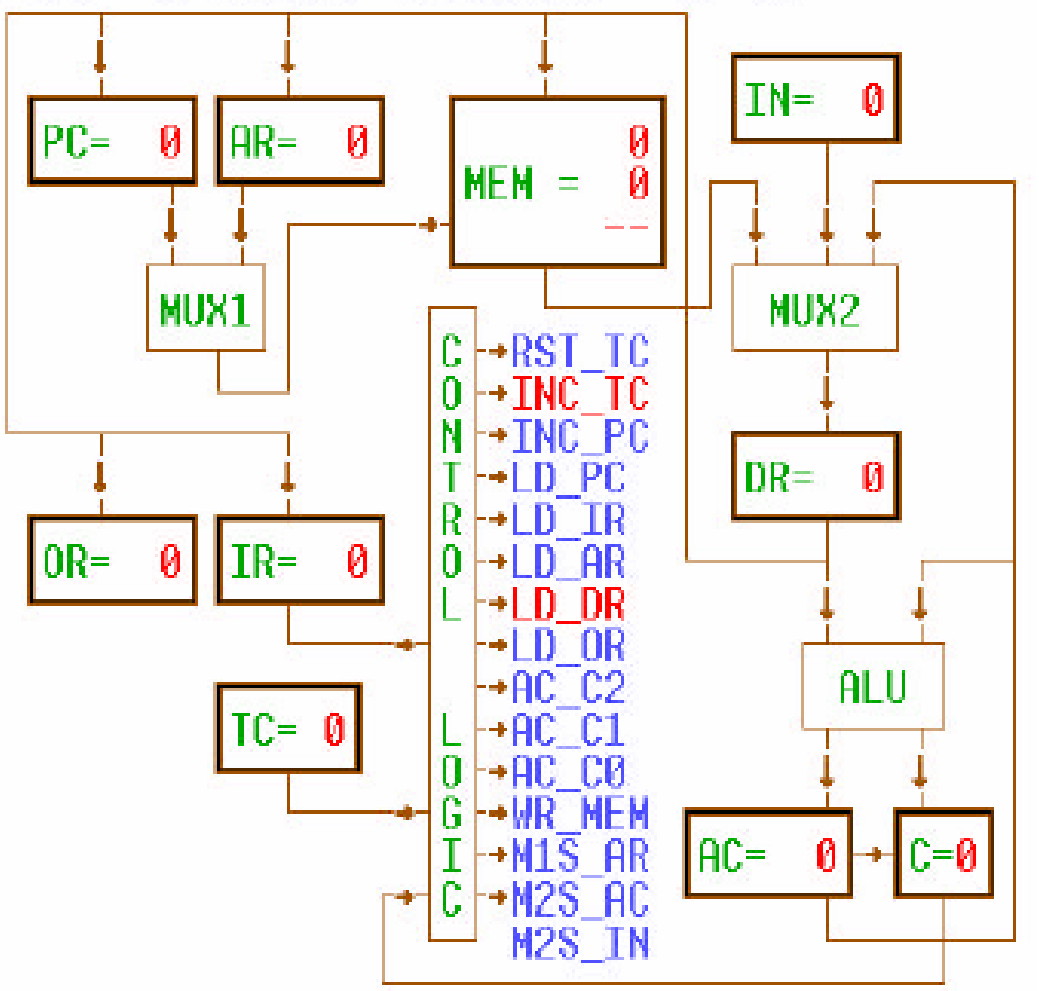


Stored Program Digital Computer

- Good example of a digital system
- Basic architecture consists of:
  - CPU
    - Control Unit
    - ALU
  - Memory
  - Input/Output (I/O) Devices



Stored Program Digital Computer



Memory:

Program Memory (MEM)

Data Register (DR)

I/O Devices:

Input Register (IN)

Output Register (OR)

Arithmetic-Logic Unit:

Arithmetic/Logic Unit (ALU)

Accumulator (AC)

ALU Carry Register (C)

Control Unit:

Program Counter (PC)

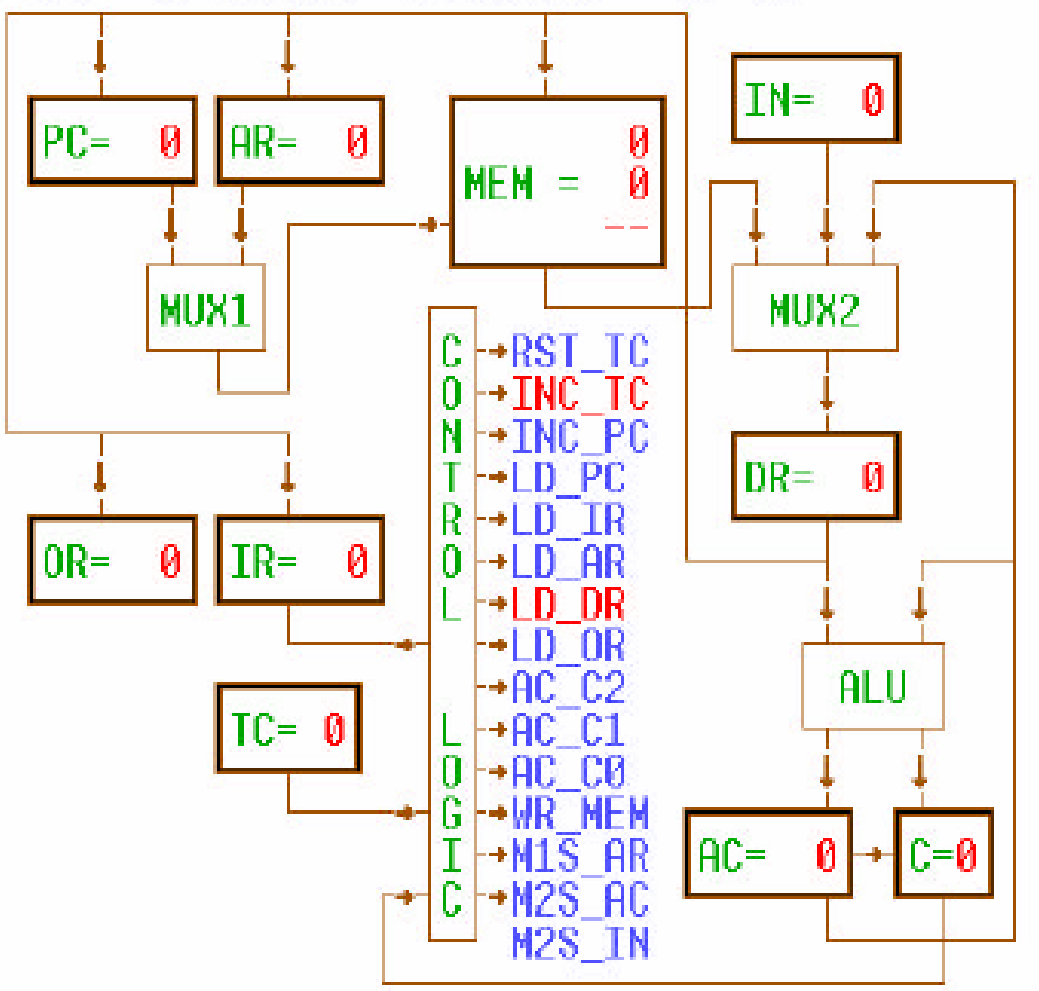
Address Register (AR)

Instruction Register (IR)

Timing Counter (TC)

Control Logic

Stored Program Digital Computer



Sequential Logic:

Program Memory (MEM)
Program Counter (PC)
Address Register (AR)
Data Register (DR)
Input Register (IN)
Output Register (OR)
Accumulator (AC)
ALU Carry Register (C)
Instruction Register (IR)
Timing Counter (TC)

Combinational Logic:

Control Logic
Arithmetic/Logic Unit (ALU)
Multiplexers 1&2 (MUX)

Digital Computer Basic Operation

- Consists of a series of instructions cycles, each consisting of:
  - Fetch
    - Fetch instruction from Program Memory (MEM) to Data Register (DR)
  - Decode
    - Pass instruction from DR to Instruction Register (IR) and decode using Control Logic
  - Execute
    - Perform operations decoded by Control Logic such as:
      - Get operands from MEM or Input Register (IN)
      - Arithmetic/logic operations
      - Store results in MEM or Output Register (OR)