CHAPTER 9

BOOLEAN ALGEBRAS AND COMBINATORIAL CIRCUITS

APPLICATIONS OF CIRCUIT IDEAS

- **Now we will look at some applications of these ideas**

FUNCTIONALLY COMPLETE SETS

- A set of gates is functionally complete if it can be used to build any combinatorial circuit
- {AND, IOR, NOT} is functionally complete

- A new gate to learn is the NAND gate
- NAND means "not and": x NAND y means –(x AND y)
- The gate symbol is a combination of the AND and the NOT

THE SET {NAND} IS FUNCTIONALLY COMPLETE

- **NOT x = x NAND x (We have added the NOT gate)**
- **X AND y = NOT (x NAND y) (We already have NOT, NAND)**
- **X OR y = - (-x AND -y) (Again, we already have NOT, NAND)**
- **So, we can build all logic circuits from only (perhaps a lot of!) NAND gates**

SIMPLIFYING LOGIC - A SIMPLE DEMONSTRATION

- –ABC + A(-B)C + ABC
- = BC (-A) + A(-B)C + BC(A)
- = BC (-A) + BC (A) + A(-B)C
- = (BC)(-A + A) + A(-B)C
- = (BC)(1) + A(-B)C
- = BC + A(-B)C
- =… = (B + A(-B))C

A HALF ADDER

- A half adder is a circuit that can add two bits
- It takes the two bits as input and produces a sum bit and a carry bit
  - The actual sum of the two bits is carry, sum
- Here is a picture of a half adder
- The picture is on p. 444

A FULL ADDER

- The problem with a half adder is that it needs more circuitry to be useful
- A full adder can add two bits plus the carry bit from the addition "on the right"
- It also produces a sum and a carry

- The name for the bit in a byte that act as 2^0 is the least significant bit
- To make adders useful for adding two bytes, for example, you need
  - A half adder for the least significant bits
  - A full adder for the rest of the bits