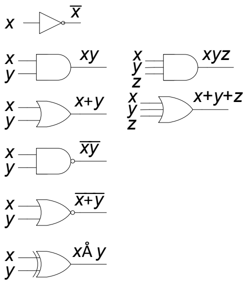
Basic logic gates

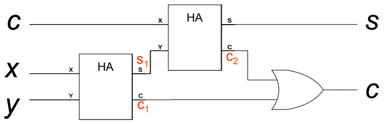
- Not
- And
- Or
- Nand
- Nor
- Xor



The full adder

- The “HA” boxes are half-adders

x	*y*	*c*	s_1	c_1	c_2	carry	sum
1	1	1	0	1	0	1	1
1	1	0	0	1	0	1	0
1	0	1	1	0	1	1	0
1	0	0	1	0	0	0	1
0	1	1	1	0	1	1	0
0	1	0	1	0	0	0	1
0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	0



Number System Conversion

- A9BF$_{16}$ = ?$_2$
- 1010100110111111$_2$
- 456$_{10}$ = ?$_2$
- 111001000$_2$
- 456$_{10}$ = ?$_7$
- 1221$_7$

Two’s Complement

- Given a positive integer *a*, the two’s complement of *a* is the n-bit representation of $2^n - a$
- $2^8 - 35 = 256 - 35 = 221 = 11011101_2$
- *a*’s two’s complement represents -*a*
- Always relative to a fixed bit length
- Bit length of 32 and 64 are most commonly used

One’s Complement

- An easier way to calculate two’s complement
- $2^8 - a = (2^8 - 1) - a + 1$
- $2^8 - 1 = 11111111_2$
- Subtracting any binary number from all 1’s is equivalent to negating all bits, i.e. taking the one’s complement

Example

$$2^8 - 35 = (2^8 - 1) - 35 + 1 =$$

$$11111111_2$$

$$-$$

$$00100011_2$$

$$11011100_2 + 1 = 11011101_2 = 221$$

Two’s Complement Again

- To find the two’s complement of a positive integer *a*:
  - Write the n-bit binary representation for *a*
  - Negate all bits
  - Add 1 to the resulting binary notation

8-bit Representations

Integer	8-bit Binary	2’s complement
127	01111111	
126	01111110	
…	…	
2	00000010	
1	00000001	
0	00000000	
-1	11111111	2^8 - 1
-2	11111110	2^8 - 2
-3	11111101	2^8 - 3
…	…	
-127	10000001	2^8 - 127
-128	10000000	2^8 - 128

Addition with Negative Numbers

$$64-15=64+(-15)=$$
$$01000000_2+((11111111_2-00001111_2)+1_2)=$$
$$01000000_2+11110001_2$$
$$01000000_2$$
$$11110001_2$$
$$00110001_2=49$$

From ThinkGeek (http://www.thinkgeek.com)