Negative Binary Numbers

- In decimal we are quite familiar with placing a "-" sign in front of a number to denote that it is negative
- The same is true for binary numbers a computer won't understand that
- What happens in memory then?

Binary Negative Numbers

- There are several representations
  - Signed magnitude
  - One's complement
  - Two's complement
- Two's complement is the system used in microprocessors
- Most significant bit becomes important

Signed Magnitude

- Represent the decimal number as binary
- Left bit (MSB) used as the sign bit
- Only have 7 bits to express the number

- $12_{10} = 00001100$
- $-12_{10} = 10001100$
- How many representations are there for zero?

One's Complement

- Method: Invert the ones and zeros

- $11_{10} = 00001011$
- $-11_{10} = 11110100$

- 0 in MSB implies positive
- 1 in MSB implies negative

Two's Complement

- Method: Take the one's complement and add 1

- 11_{10} = 00001011
- -11_{10} = 11110100 one's comp
- -11_{10} = 11110101 two's comp

Why Two's Complement?

- One representation of zero (Check)

- Enables subtraction operation by considering the addition of a positive number with a two's complement number
- Only need addition
- MSB indicates the sign of the number

One Representation of 0

- 00000000

- 11111111

- (1) 00000000

Subtraction by Addition

- Consider 10 – 6 using binary subtraction

- 00001010 10
- 00000110 6
- 00000100 subtract gives 4

Two's complement

- -6
- Binary conversion of 6 = 00000110
- One's complement (invert) = 11111001
- Two's complement (+1) = 11111010

Subtraction by Addition

- Consider 10 – 6 using two's complement
- 0001010 10
- 11111010 6
- (1) 00000100 adding gives 4

Why two's complement

- Only need one type of hardware/process to add both signed and unsigned numbers.