Lecture 4 – Binary representation

Dr. Aftab M. Hussain,

Assistant Professor, PATRIoT Lab, CVEST

Chapter 2

Representing negative binary

- In ordinary arithmetic, a negative number is indicated by a minus sign and a positive number by a plus sign
- Because of hardware limitations, computers must represent everything with binary digits
- It is customary to represent the sign with a bit placed in the leftmost position of the number
- The convention is to make the sign bit 0 for positive and 1 for negative
- This can be done using:
    1. Signed magnitude representation
    2. Signed complement representation
        1. Signed 1’s complement representation
        2. Signed 2’s complement representation

Signed magnitude representation

- In this notation, the number consists of a magnitude and a symbol (+ or -) or a bit (0 or 1) indicating the sign
- This is similar to the representation of signed numbers used in ordinary arithmetic
- For example, the string of bits 01001 represents +9, and 11001 represents -9 in signed magnitude representation
- In signed-magnitude, -9 is obtained from +9 by changing only the sign bit in the leftmost position from 0 to 1
- Weird: +0 is represented as 0000 and minus 0 is represented as 1000. So, two representations for zero – inefficient and may cause errors

Signed complement representation

- When arithmetic operations are implemented in a computer, it is more convenient to use a different system, referred to as the *signed complement* system, for representing negative numbers
- In this system, a negative number is indicated by its complement
- Whereas the signed-magnitude system negates a number by changing its sign, the signed-complement system negates a number by taking its complement
- Since positive numbers always start with 0 (plus) in the leftmost position (in all representations), it follows that the complement will always start with a 1, indicating a negative number
- In signed-1's-complement, -9 is obtained by taking the 1's complement of all the bits of +9, including the sign bit
- The signed-2's-complement representation of -9 is obtained by taking the 2's complement of the positive number, including the sign bit

Reading and Writing signed complements

- Write into memory the following numbers in signed 2’s complement prestation in 4 bits:
- +3
- -7
- 0

- We read these numbers from memory knowing its in signed 2’s complement prestation in 4 bits:
- $(1100)_2$
- $(1111)_2$
- $(0000)_2$
- $(1000)_2$

Interpretations for 4 bit binary numbers

Decimal	Signed-2's Complement	Signed-1's Complement	Signed Magnitude
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000
−0	—	1111	1000
−1	1111	1110	1001
−2	1110	1101	1010
−3	1101	1100	1011
−4	1100	1011	1100
−5	1011	1010	1101
−6	1010	1001	1110
−7	1001	1000	1111
−8	1000	—	—

Signed addition

- Here is some magic: if the numbers are represented in memory in 2’s complement form, we just need to add the two numbers, the sign takes care of itself!
- Bigger magic: the result is also in 2’s complement representation
- The sign bit is to be included in the addition and if there is a carry, it is discarded
- Examples in 4-bit signed 2’s complement representation:

1. 3+1
2. 1+(-7)
3. (-8)+5
4. 7+(-3)

- In order to obtain a correct answer, we must ensure that the result has a sufficient number of bits to accommodate the sum
- If we start with two n-bit numbers and the sum occupies $n + 1$ bits, we say that an overflow occurs

Signed subtraction

- Subtraction of two signed binary numbers when negative numbers are in 2’s-complement form is simple and can be stated as follows:
  - Take the 2’s complement of the subtrahend (including the sign bit) and add it to the minuend (including the sign bit)
  - A carry out is discarded, if any
- This works because: $M - N = M + (-N)$
- Examples in 4-bit signed 2’s complement representation:

1. 3-5
2. 6-2
3. 1-7