Lecture 3 – Binary codes

Dr. Aftab M. Hussain,

Assistant Professor, PATRIoT Lab, CVEST

Chapter 2

Radix complement

- The *r*'s complement of an *n*-digit number *N* in base *r* is defined as $r^n - N$ for $N \neq 0$ and as 0 for $N = 0$
- Comparing with the (*r* - 1)'s complement, we note that the r's complement is obtained by adding 1 to the (*r* - 1)'s complement, since $r^n - N = [(r^n - 1) - N] + 1$
- Thus, the 10's complement of decimal 2389 is 7610 + 1 = 7611 and is obtained by adding 1 to the 9's complement value
- The 2's complement of binary 101100 is 010011 + 1 = 010100 and is obtained by adding 1 to the 1's-complement value
- Examples:
  - $(66772)_{10}$
  - $(10011)_2$

Some notes on Complements

- If the original number *N* contains a radix point, the point should be removed temporarily in order to form the *r*'s or (*r* - 1)'s complement
- The radix point is then restored to the complemented number in the same relative position
- Example: 9's complement and 10's complement of $(82.314)_{10}$
- It is also worth mentioning that **the complement of the complement restores the number to its original value**
- To see this relationship, note that the *r*'s complement of *N* is $r^n - N$, so that the complement of the complement is $r^n - (r^n - N) = N$ and is equal to the original number
- *(r-1)*'s complement of *N* is $r^n - 1 - N$, so that the complement of the complement is $(r^n - 1) - (r^n - 1 - N) = N$ and is equal to the original number

Subtraction with Radix complements

- The usual method of borrowing taught in elementary school for subtraction is less efficient when subtraction is implemented with digital hardware
- Lets assume we have to perform M-N in base r
- Here is the algorithm using Radix complement:
  1. Take radix complement of N: $r^n - N$
  2. **Add** this to M: $r^n - N + M = r^n + (M - N) = r^n - (N - M)$
  3. If you get a carry in the (n+1)th digit, then the result is positive, discard the carry and you are done
  4. If you *do not* get a carry in the (n+1)th digit, then the result is **negative**. Take the radix complement of the number to get the answer, then put a negative sign
- 10's complement subtraction:
  - $(9812)_{10} – (3142)_{10}$
  - $(1423)_{10} – (7336)_{10}$

Subtraction with Diminished radix complements

- The usual method of borrowing taught in elementary school for subtraction is less efficient when subtraction is implemented with digital hardware
- Lets assume we have to perform M-N in base r
- Here is the algorithm using Diminished radix complement:
    1. Take diminished radix complement of N: $r^n - 1 - N$
    2. **Add** this to M: $r^n - 1 - N + M = r^n + (M - N - 1) = (r^n - 1) - (N - M)$
    3. If you get a carry in the (n+1)th digit, then the result is positive, ***add the carry to the result*** and you are done
    4. If you ***do not*** get a carry in the (n+1)th digit, then the result is **negative**. Take the diminished radix complement of the number to get the answer, then put a negative sign
- 9's complement subtraction:
    - $(6552)_{10} - (3145)_{10}$
    - $(2142)_{10} - (9667)_{10}$

Binary subtraction with complements

- Perform the following subtractions using 2's complement method:
- $(110001)_2 - (010100)_2$
- $(010110)_2 - (100)_2$
- $(10)_2 - (100000)_2$
- $(100001)_2 - (110100)_2$

- Perform the following subtractions using 1's complement method:
- $(110001)_2 - (010100)_2$
- $(100100)_2 - (011101)_2$
- $(1)_2 - (10100)_2$
- $(11010)_2 - (110111)_2$

Subtraction using complements

Radix Subtraction		Reduced Radix Subtraction	
• Find Radix Complement of Y • Add Y complement to X		• Find Reduced Radix Complement of Y • Add Y complement to X	
<u>Extra Leading Digit</u>	<u>No Extra Digit</u>	<u>Extra Leading Digit</u>	<u>No Extra Digit</u>
• Drop extra digit	• Take Radix Compleme nt • Attach Negative	• Drop extra digit • Add extra digit to result	• Take Reduced Radix Complement • Attach Negative

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