CSC 370 - Computer Organization and Architecture

Lecture 0: Review of Numeric Representation

Number Systems - Base 10

The number system that we use is base 10:

1734 = 1000 + 700 + 30 + 4

= 1x1000 + 7x100 + 3x10 + 4x1

= $1\text{x}10^3 + 7\text{x}10^2 + 3\text{x}10^1 + 4\text{x}10^0$

724.5 = 7x100 + 2x10 + 4x1 + 5x0.1

= $7\text{x}10^2 + 2\text{x}10^1 + 4\text{x}10^0 + 5\text{x}10^{-1}$

Why use base 10?

Number Systems - Base 2

For computers, base 2 is more convenient (why?)

$10011_2 = 1x16 + 0x8 + 0x4 + 1x2 + 1x1 = 19_{10}$

$100010_2 = 1x32 + 0x16 + 0x8 + 0x4 + 1x2 + 0x1 = 34_{10}$

$101.001_2 = 1x4 + 0x2 + 1x1 + 0x0.5 + 0x0.25 + 1x0.125$

$= 5.125_{10}$

<u>Example</u> - $1101011_2 = ?$

$10110111_2 = ?$

$10100.1101_2 = ?$

Number Systems - Base 16

Hexadecimal (base 16) numbers are commonly used because it is convert them into binary (base 2) and vice versa.

$8CE_{16} = 8x256 + 12x16 + 14x1$

$= 2048 + 192 + 14$

$= 2254$

3F9 $= 3x256 + 15x16 + 9x1$

$= 768 + 240 + 9 = 1017$

Number Systems - Base 16 (continued)

Base 2 is easily converted into base 16:

100011001110_2 = 1000 1100 1110 = 8 C E $_{16}$

11101101110101001_2 = 1 1101 1011 1010 1001 = 1 D B A 9_{16}

$10110001010000010111_2 = ?_{16}$

$1011010100101110 11_2 = ?_{16}$

Number Systems - Base 16 (continued)

Converting base 16 into base 2 works the same way:

$F3A5_{16}$ = 1111 0011 1010 0101_2

$76EF_{16}$ = 0111 0110 1110 1111_2

$AB3D_{16} = ?_2$

$15C.38_{16} = ?_2$

Converting From Decimal to Binary

19
9 R 1
4 R 1
2 R 0
1 R 0
0 R 1

10011_2

Converting From Decimal to Hexadecimal

ED_{16}

16 | 237
14 R 13
0 R 14

Converting From Decimal to Octal

8 | 237

29 R 5

3 R 5

0 R 3

355_8

Binary, Octal, Decimal and Hexadecimal Equivalents

Binary	Decimal	Octal	Hex.	Binary	Decimal	Octal	Hex.
0000	0	0	0	1000	8	10	8
0001	1	1	1	1001	9	11	9
0010	2	2	2	1010	10	12	A
0011	3	3	3	1011	11	13	B
0100	4	4	4	1100	12	14	C
0101	5	5	5	1101	13	15	D
0110	6	6	6	1110	14	16	E
0111	7	7	7	1111	15	17	F

2s Complement Representation

- In 2s complement representation, we subtract the absolute value from 2^n:

```
 100000000
  00000110
 ---------
  11111010
```

```
    -6                  11111010
 + +13                  00001101
 ------               ----------
   + 7          1       00000111  ( = +7)
```

2s Complement Representation (continued)

- The 2s complement representation can also be found by reversing the bits (into 1s complement) and then adding 1:

```
6 => 00000110 =>11111001
              +        1
               ---------
               11111010

43 => 00101011 => 11010100
                 +       1
                  --------
                  11010101
```

Overflow

- If an addition operation produces a result that exceeds our number system's range, ***overflow*** has occurred.
- Addition of two numbers of the same sign produces overflow; addition two numbers of opposite sign cannot cause overflow.

-3	1101	+5	0101
+6	0110	+6	0110
+3	1 0011 = ***+3***	+11	1011 = ***-5***

-8	1000	+7	0111
-8	1000	+7	0111
-16	1 0000 = ***0***	+14	1110 = -2

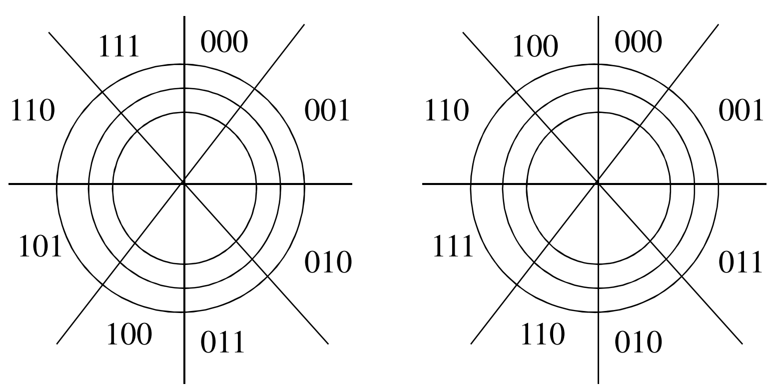
Binary Representation of Decimal Numbers

Decimal Digit	**BCD (8421)**	**2421**	**Excess-3**
0	0000	0000	0011
1	0001	0001	0100
2	0010	0010	0101
3	0011	0011	0110
4	0100	0100	0111
5	0101	1011	1000
6	0110	1100	1001
7	0111	1101	1010
8	1000	1110	1011
9	1001	1111	1100

Gray Codes

- Sometimes electromechanical applications of digital systems (machine tools, automotive brake systems and copiers) require a digital value that indicates a mechanical position.
- A standard binary code may see more than one bit change from one position to another, which could lead to an incorrect reading if mechanical assembly is imperfect.

Binary Code vs. Gray Code



Binary Code

Gray Code

ASCII representation of characters

- ASCII (***A***merican ***S***tandard ***C***ode for ***I***nformation ***I***nterchange) is a numeric code used to represent characters.
- All characters are represented this way including:
  - words (character strings)
  - numbers
  - punctuation
  - control characters
- There are separate values for upper case and lower case characters:

A	65	z	122
B	66	*blank*	32
Z	90	$	52
a	97	0	48
b	98	9	57

Control Codes

- ASCII (a 7-bit code) has $2^7 = 128$ values.
- We only need 62 for alphanumeric characters. Even after accounting for common punctuation, there are far more available code values than we need. What do we use them for?
- Control codes include DEL (for delete), NUL (for null). STX (Start of Text), CR (for carriage return), etc.

Error Detection Codes

- An error is a corruption of the data from its correct state.
- There are several codes that allow use to detect an error. These include:
  - Parity
  - CRC
  - Checksum

Parity

- Parity is an extra bit appended to our data which indicates whether the data bits add up to an even (for even parity) or odd (for odd parity) value.

Parity Generation

Message (xyz)	P(odd)	P(even)
000	1	0
001	0	1
010	1	0
011	0	1
100	1	0
101	0	1
110	1	0
111	0	1

Odd Parity