CECS 225 NUMBERING SYSTEMS AND CONVERSIONS

Many number systems are used in digital technology. The most common are decimal, binary, octal, and hexadecimal systems. The decimal system is the most familiar to us. First, examine certain characteristics of the decimal system to understand how it relates to the other systems. The other systems we will study are:

- Decimal
- Binary
- Hexadecimal

Decimal System

The decimal system is composed of 10 symbols: (0,1,2,3,4,5,6,7,8,9). Using these symbols as digits of a number, we can express any quantity. The decimal system is also called the base-10 system because it has 10 digits. A decimal value is assumed if no prefix or suffix is given. Or a subscript of 10 can be used.

Analyze the decimal quantity **6,209**$_{10}$ and the value each digit contributes to the quantity.

Digit→	6	2	0	9
Column	thousands	hundreds	tens	ones
Positional Weight	10^3	10^2	10^1	10^0
	Most Significant Digit			Least Significant Digit

$\mathbf{6}*10^3 + \mathbf{2}*10^2 + \mathbf{0}*10^1 + \mathbf{9}*10^0 \quad = 6000 + 200 + 0 + 9 \quad = 6209_{10}$

This method in evaluating a decimal quantity can be applied to any numbering system.

Binary System

In the binary system, there are only two symbols or possible digit values, 0 and 1. A binary digit is known as a bit. This base-2 system can be used to represent any quantity that can be represented in any other number system. Some notations used to identify a binary quantity are the prefix 0b which is seen in some console outputs or the subscript of 2. (In NASM syntax, a suffix b is used i.e. 1101b)

Analyze the binary quantity **1101** and the value each bit contributes to the quantity.

Bit→	1	1	0	1
Column value	8	4	2	1
Positional Weight	2^3	2^2	2^1	2^0
	MSB (Most Significant Bit)			LSB (Least Significant Bit)

$\mathbf{1}*2^3 + \mathbf{1}*2^2 + \mathbf{0}*2^1 + \mathbf{1}*2^0 \quad = 8 + 4 + 0 + 1 \quad = 13$

Binary Counting

The Binary counting sequence is shown in the table below:

Binary				Decimal
2^3	2^2	2^1	2^0	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

Hexadecimal System

The hexadecimal system uses base 16. This means it has 16 possible digit symbols. It uses the digits 0 through 9 plus the letters A, B, C, D, E, and F as the 16 digit symbols. The table below shows the decimal value of each hexadecimal letter

Hex letter	Decimal Value
A	10
B	11
C	12
D	13
E	14
F	15

Hexadecimal quantities are commonly indicated with the prefix 0x or the subscript 16.

Analyze the hex quantity **0xA20B** and the value each bit contributes to the quantity.

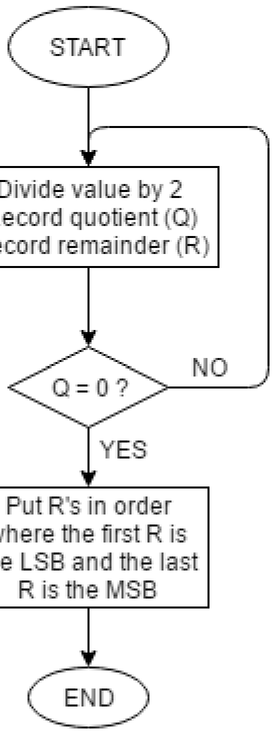
Digit→	A	2	0	B
Column Value	4096	256	16	1
Positional Weight	16^3	16^2	16^1	16^0
	Most Significant Digit			Least Significant Digit

10*163 + 2*162 + 0*161 + 11*160 = 10*4096 + 2*256 + 11*1 = 41483_{10}

In our introduction to the binary and hexadecimal numbering systems we have analyzed quantities using positional weight to perform conversion to decimal. Next, we will see how to convert from Decimal to Binary, convert Decimal to Hex, and convert between Binary and Hex.

Conversion from Decimal to Binary or Hex

A technique known as repeated division can be used to convert from Decimal to Binary or Hex. The flow chart below outlines the logic behind the conversion process.



The given flow chart is in the context of Decimal to Binary conversion but the process is the same for conversion to Hex but the Decimal value will be divided by 16 instead. Next observe the application of the repeated division technique.

Decimal To Binary Conversion Example:

Convert 35 to binary using repeated division

35 / 2: Q = 17 R = 1 🡐LSB
17 / 2: Q = 8 R = 1
8 / 2: Q = 4 R = 0
4 / 2: Q = 2 R = 0
2 / 2: Q = 1 R = 0
1 / 2: Q = 0 R = 1 🡐MSB
100011

Decimal to Hex Conversion Example:

Convert 123 from decimal to HEX using repeated division.
Perform repeated division by 16 to convert Decimal to HEX

123 / 16 =	Q = 7	R = 11	R in Hex = 0xB	🡐LSD
7 / 16 =	Q = 0	R = 7	R in Hex = 0x7	🡐MSD

The decimal value 123 converted to HEX is **0x7B**

Conversion between Hex and Binary

Conversion from Hex to Binary or from Binary to Hex is easy and doesn't involve math. Rather one must know the 4-bit binary equivalent of a Hex number to make the conversion.

Binary to Hex Conversion Example:

Convert the binary value 0001001000110100 to Hex
There are 16 bits in the binary value

0001	0010	0011	0100
0x1	0x2	0x3	0x4

0001001000110100 in binary is equivalent to **0x1234** in Hexadecimal

Hex to Binary Conversion Example:

Convert the Hex value F3A7 to binary

F	3	A	7
1111	0011	1010	0111

The HEX value F3A7 is **1111 0011 1010 0111** in binary

Negative values in binary:

Negative values are most practically represented in binary in a form known as sign-complement form. The sign-complement form uses the MSB as a sign bit. If the sign bit = 1 then the value is negative, if the sign bit = 0 then the value is positive.

If the sign bit = 0 then the binary quantity is evaluated as it normally would be and positional weighting can be used to convert to decimal.

If the sign bit = 1 then the binary quantity must first be converted using a process known as **2's complement** before positional weighting can be used to convert to decimal.

The 2's complement process is outlined as follows:

1. Keep the least significant zeros (if there are any)
2. Keep the least significant one
3. Invert the most significant bits (i.e. turn a 1 into a 0 or turn a 0 into a 1)

8-bit 2’s Signed Binary to Decimal Conversion Example:

Given the signed binary quantity 11000100, find the equivalent decimal value.
Step 1. In **1**1000100, The MSB = 1 so the quantity is negative
Step 2. Apply the 2’s complement process to the lower 7-bits ➔ 1000100

Given quantity	1	0	0	0	1	0	0
	Invert 1 to 0	Invert 0 to 1	Invert 0 to 1	Invert 0 to 1	Keep least significant one	Keep least significant zeros	
2’s complement	0	1	1	1	1	0	0

Step 3. Now convert the quantity to decimal to find the absolute power of the given binary quantity.

$0*2^7 + 1*2^6 + 1*2^5 + 1*2^4 + 1*2^3 + 1*2^2 + 0*2^1 + 0*2^0 \quad = 64 + 32 + 16 + 8 + 4 \quad = 124$

Step 4. Apply the sign that was determined in Step 1 to get the final decimal value **-124**

Signed Decimal to 8-bit Binary Conversion Example:

Convert the signed decimal value -120 to 8-bit signed binary
Convert 120 to binary using repeated division

120 / 2: Q = 60 R = 0 LSB
60 / 2: Q = 30 R = 0
30 / 2: Q = 15 R = 0
15 / 2: Q = 7 R = 1
7 / 2: Q = 3 R = 1
3 / 2: Q = 1 R = 1
1 / 2: Q = 0 R = 1 MSB
The absolute power of -120 in 8-bit binary is 01111000
The value is negative so 2's complement: **10001000**

Zero Fill

Sometimes a small unsigned binary quantity will need to be expanded to fill more bit positions. This must be done in such a way that the value of the binary quantity is not changed. The method to expand the number of bits in an unsigned quantity without changing its value is to fill most significant bit positions with 0’s. This is illustrated in the following examples:

Expand the unsigned binary quantity 1001 to 8-bits: 1001 ➔ **00001001**

Expand the unsigned binary quantity 0110 to 8-bits: 0110 ➔ **00000110**

- In each of the previous examples, adding more zero’s to the left in the most significant bit positions does not change the value of the binary quantity.

Sign Extension

Sometimes a small signed binary quantity will need to be expanded to fill more bit positions. This must be done in such a way that the value of the signed binary quantity is not changed. The method to expand the number of bits in a signed binary quantity without changing its value is to copy the most significant bit value and use it to fill positions to the left. This is illustrated in the following example:

Expand the signed binary quantity 1001 to 8-bits: 1001 ➔ **11111001**

- In this example, the quantity 1001 is equivalent to -7 in decimal. The MSB of 1001 serves as the sign bit making this a negative quantity. Replicating the sign bit 4 positions to the left preserves the negative sign and value of the binary quantity i.e. 11111001 also has a value of -7 in decimal

Expand the signed binary quantity 0110 to 8-bits: 0110 ➔ **00000110**

- In this example, the quantity 0110 is equivalent to 6 in decimal. The MSB of 0110 serves as the sign bit making this a positive quantity. Replicating the sign bit 4 positions to the left preserves the positive sign and value of the binary quantity i.e. 00000110 also has a value of 6 in decimal