Number Systems (B)

Number System Characteristics

(Tinder) What's important in choosing a number system? Basically, there are four important characteristics desirable in a number system with a fifth one when using computers. They are:

- **The ability to distinguish between symbols,**
- ***Arithmetic* operations capability,**
- **Error control capability,**
- **Tractability and speed,**
- **When using computers, it should have a minimum number of easily identifiable states.**

For the most part, the **decimal number system** satisfies the first four characteristics for transfer of information between humans. **Binary** and **Roman numbers** don't meet the criteria at all.

Even though the **Binary system** doesn't meet the 1^{st} four at all, it is great for digital computer applications, mainly because currently digital machines **<u>recognize only two identifiable states</u>**. **Binary** is made to order for this problem. However, the digital computer still needs to interface with humans. **Binary** data has to be converted into either a decimal or character-based form that can be recognized by humans. While having a minimum number of identifiable characters is nice, it's just not practical when dealing with humans. Over the next few pages, we will discuss methods of converting between number systems. First, let's review how the number system we are most comfortable with (decimal) is built.

The Radix point

The **total number of digits in a number system is called the radix (r) or "base" of the system.** The highest number in the system is equal to (r-1). The radix of the decimal system is 10.

Selected Number Systems

Name	Base (radix)	Count	
Decimal	10	0-9	
Binary	2	0-1	
Octal	8	0-7	
Hex	16	0-15	0-9, A, b, C, d, E, F
BCD	9	0-9	

Note that: $r - 1 = 10 - 1 = 9$, which is the highest number in the decimal system. What you know as the "**decimal point**" **is also known as** the "**radix point**".

The radix point separates the integer portion of the number from the fraction portion.

Each position in the number has a weight attached to it. This weight is equivalent to the radix of the system **raised to a power**. **This power is the position number of the digit**. It starts with a '**0**' immediately to the left of the radix point and increases by 1 as you move to the left. It decreases by one as you move to the right.

Number System Conversions

The Base 10 (decimal) number system

Let's look at an example of the decimal number system. Note that the base number of the system is raised to the power of the position of the digit.

radix pt.

$$\underbrace{3}_{10^2}\ \underbrace{4}_{10^1}\ \underbrace{2}_{10^0}\ .\ \underbrace{1}_{10^{-1}}\ \underbrace{5}_{10^{-2}}$$

We can represent this number by the polynomial

$$\left(3*10^2\right)+\left(4*10^1\right)+\left(2*10^0\right)+\left(1*10^{-1}\right)+\left(5*10^{-2}\right)$$

There are many number systems other than **Decimal (Base 10.)** We will stick to three of the most common: **Binary (Base 2)**, **Octal (Base 8)**, and **Hex (Base 16).** The term "**Base**" is derived from the fact that all number systems are logarithmic in nature. For instance, the decimal number system is **log to the Base 10** ($\log_{10}x$). Binary is "**log to the Base 2**", ($\log_2 x$).

The Binary System

The **radix** of the Binary system is **2**, thus binary is known as "Base **2**" and the highest number in the system is **r - 1 = 1**. The **BInary digiT** is abbreviated as "**BIT**". Just like the decimal systems, it can be shown in positional notation.

Binary to Decimal Conversion

Conversion between the Binary number system to the Decimal number system is quite easy. Just take each bit and multiply it by two raised to the power of the position value of the bit as can be seen in Example 1 below.

Binary – to – Base 10 Conversion Example 1

Convert **10011.1001$_2$** to **Base 10**

$$\underbrace{1}_{2^4}\ 0\ 0\ \underbrace{1}_{2^1}\ \underbrace{1}_{2^0}.\underbrace{1}_{2^{-1}}\ 0\ 0\ \underbrace{1_2}_{2^{-4}}$$

$$\left(1 * 2^4\right) + \left(1 * 2^1\right) + \left(1 * 2^0\right) + \left(1 * 2^{-1}\right) + \left(1 * 2^{-4}\right)$$

$$16 + 2 + 1 + .5 + .0625 = 19.5625_{10}$$

A couple of things to note about the above:

- **First, note that the radix of the system is placed as a subscript of the number. This has to be done for all systems except base 10.**
- **In fact, if more systems than base 10 are being used, you should do it for base 10 as well. If this is not done, I will assume the number to be base 10, which will make it wrong!**

You have just performed a **Binary to Decimal conversion**. Didn't hurt a bit; did it! Let's do it one more time, but this time in the other direction.

Binary – to – Base 10 Conversion Example 2

Convert **101101$_2$** to **Base 10**

$$\overbrace{1}^{5}\quad \overbrace{0}^{4}\quad \overbrace{1}^{3}\quad \overbrace{1}^{2}\quad \overbrace{0}^{1}\quad \overbrace{1_2}^{0}$$

$$2^5 \qquad 2^3 \quad 2^2 \qquad 2^0$$

$$32 + 8 + 4 + 1 \Rightarrow 45_{10}$$

Decimal to Binary Conversion: The Sum of Weights Method

Most books teach a method called "**radix divide/multiply method**" to perform conversions between Decimal and other number systems. I have found this method to be cumbersome and the user is very prone to making mistakes with it. In addition, it tends to take up a lot of space on a paper. The method used in this course is called the "**Sum of Weights**" method. **This is the preferred method and you are required to use this method on tests and homework in this course.** The same method can be used **ANY TIME** you desire to convert from the decimal system to some other number system.

Sum of Weights Base 10-to-Base 2 Conversion Example 1

The Sum of Weights procedure is as follows:

- Record the list of powers of 2 up to the point of going over the decimal number to be converted.

Convert 182_{10} to Binary (Base 2)

The decimal #	$\Rightarrow$	182_{10}							
Powers of 2	$\Rightarrow$	**128**	**64**	**32**	**16**	**8**	**4**	**2**	**1**

- Divide that number into the decimal number. It will go in '1' time on the 1st division. If it goes in more than once then a mistake has been made. Record the remainder.

		182_{10}	*remainder* 54						
		1							
Powers of 2	$\Rightarrow$	**128**	**64**	**32**	**16**	**8**	**4**	**2**	**1**

Example continued on the next page)

Example Continues)

- Divide the next power of 2 into this remainder. This next time, it will either go into the decimal number '**0**' or '**1**' times.

				remainder					
		182_{10}	54	54					
		1	0						
Powers of 2	⇒	128	64	32	16	8	4	2	1

- Repeat until finished.

		182_{10}	54	54	22	6	6	2	0		
		1	0	1	1	0	1	1	0	=	$\boxed{10110110_2}$
Powers of 2	⇒	128	64	32	16	8	4	2	1		

Sum of Weights Base 10-to-Base 2 Conversion Example 2

Convert **42**$_{10}$ to Binary (Base 2) using the **Sum of Weights Method**

- Record the list of powers of 2 up to the point of going over the decimal number to be converted.

The decimal #	⇒	42					
Powers of 2	⇒	32	16	8	4	2	1

Example continues on the next page)

Example Continues)

- Divide that number into the decimal number. It will go in '1' time on the 1st division. If it goes in more than once then a mistake has been made. Record the remainder.

			remainder				
		42	10				
		1					
Powers of 2	⇒	32	16	8	4	2	1

- Divide the next power of 2 into this remainder. This next time, it will either go into the decimal number '0' or '1' times.

				remainder			
		42	10	10			
		1	0				
Powers of 2	⇒	32	16	8	4	2	1

- Repeat until finished.

		42	10	10	2	2	0	
		1	0	1	0	1	0	⇒ $42_{10} = \boxed{101010_2}$
Powers of 2	⇒	32	16	8	4	2	1	

Very often it is very important to keep track of the number of bits. A very simple way of performing the conversion while keeping track of the digits is demonstrated below in Example 3.

Sum of Weights Base 10-to-Base 2 Conversion Example 3

Convert 273_{10} to Base 2

- Set up the **Sum of Weights** conversion matrix as before. This time, add the bit number along the top. Note that this time the order of the rows has changed to place the binary at the bottom. This is just instructor preference. Feel free to continue with the previous format if desired.

bit	$\Rightarrow$	8	7	6	5	4	3	2	1	0
#	$\Rightarrow$	273								
power of 2	$\Rightarrow$	256	128	64	32	16	8	4	2	1
273_{10}	=									

- Now just repeat the earlier **Sum of Weights** procedure until completion:

bit	$\Rightarrow$	8	7	6	5	4	3	2	1	0
#	$\Rightarrow$	273	17	17	17	17	1	1	1	1
power of 2	$\Rightarrow$	256	128	64	32	16	8	4	2	1
273_{10}	=	1	0	0	0	1	0	0	0	1

$\Rightarrow 273_{10} = \boxed{100010001_2}$

The additional step of keeping track of the bit positions tends to limit the mistakes made during the conversion.

Sum of Weights Method with a Fraction

It is also possible to convert base 10 numbers possessing fractions to other base values. In the next example, the bit numbers have been changed to power of two representations.

Sum of Weights Base 10-to-Base 2 Conversion Example 3 (with a fraction)

Let's look at the previous example but this time lets add in a fraction.

Convert 273.75_{10} to Base 2.

bit	⇒	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	.	2^{-1}	2^{-2}
#	⇒	273										.75	
power of 2	⇒	256	128	64	32	16	8	4	2	1	.	.5	.25
273.75_{10}	=												

- Since the whole number portion of the example has been completed previously, we will add that part of the solution in now.

bit	⇒	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	.	2^{-1}	2^{-2}
#	⇒	273	17	17	17	17	1	1	1	1	.	.75	
power of 2	⇒	256	128	64	32	16	8	4	2	1	.	.5	.25
273.75_{10}	=	1	0	0	0	1	0	0	0	1	.		

- Now we will start to convert the fractional portion into base 2.

bit	⇒	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	.	2^{-1}	2^{-2}
#	⇒	273	17	17	17	17	1	1	1	1	.	.75	.25
power of 2	⇒	256	128	64	32	16	8	4	2	1	.	.5	.25
273.75_{10}	=	1	0	0	0	1	0	0	0	1	.	1	

- And now the last portion is completed.

bit	⇒	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	.	2^{-1}	2^{-2}
#	⇒	273	17	17	17	17	1	1	1	1	.	.75	.25
power of 2	⇒	256	128	64	32	16	8	4	2	1	.	.5	.25
273.75_{10}	=	1	0	0	0	1	0	0	0	1	.	1	1

$$273.75_{10} = \boxed{100010001.11_2}$$

Obviously this one worked out well for us but quite often the fraction might go on and on. In this class you are to stop after:

- **Three bits or digits** to the right of the radix point when the value will not be converted to any other base systems down the line.
- **Five bits** to the right of the radix point when the **conversion will continue into the OCTAL or HEX number systems**.

The Base 8 (Octal) System

The **radix of the Octal system is 8**, thus Octal is known as **"Base 8"** and the highest number in the system is: 8 - 1 = 7. The use of Octal is falling off these days, with Hex as its replacement. However, it's still used enough to warrant learning about it. **(The armed forces still use it in a large number of its fire control and tracking computers).**

Octal to Decimal Conversion

Just as in Binary, Octal can be converted directly to decimal as shown.

Octal to Decimal Conversion Example 1

Convert **1750_8** to **Base 10**:

- This time we will formalize the method and set things up vertically instead of horizontally the way it was done in Binary. We do this to take up less space and make it easier to catch mistakes.

Base 8 #		Powers of 8		
1	•	8^3	=	
7	•	8^2	=	
5	•	8^1	=	
0	•	8^0	=	

Example continues on the next page)

Example 1 continues)

- Convert the powers of 8 to actual whole numbers:

Base 8 #		Powers of 8		
1	•	8^3	=	1 • 512
7	•	8^2	=	7 • 64
5	•	8^1	=	5 • 8
0	•	8^0	=	0 • 1

- Perform the multiplications:

Base 8 #		Powers of 8						
1	•	8^3	=	1 • 512	=	512		
7	•	8^2	=	7 • 64	=	448	=	$\boxed{1000_{10}}$
5	•	8^1	=	5 • 8	=	40		
0	•	8^0	=	0 • 1	=	0		
1750_8						1000_{10}		

Note that the radix of the system is placed as a subscript of the number.

Base 8 to Base 10 Conversion Example 2 (with a fraction)

This time we take a look at an **Octal** number with a fraction. We set up the conversion matrix as before but this time with the fractional powers of 8.

- Convert **236.57_8** to **Base 10**

Base 8 #		Powers of 8		
2	•	8^2	=	
3	•	8^1	=	
6	•	8^0	=	
.			=	
5	•	8^{-1}	=	
7	•	8^{-2}	=	
236.57_8			=	

- Now add in the power of 8 results

Base 8 #		Powers of 8			
2	•	8^2	=	2 • 64	
3	•	8^1	=	3 • 8	
6	•	8^0	=	6 • 1	
.			=		
5	•	8^{-1}	=	5 • 0.125	
7	•	8^{-2}	=	7 • 0.015625	
236.57_8			=		

- And complete the example.

2	•	8^2	=	2 • 64	128
3	•	8^1	=	3 • 8	24
6	•	8^0	=	6 • 1	6
.			=		
5	•	8^{-1}	=	5 • 0.125	0.625
7	•	8^{-2}	=	7 • 0.015625	0.10938
236.57_8			=		158.73_{10}

Binary to Octal Conversion

Conversion from **Binary to Octal** or vice versa is easy. The largest number represented by a single Octal bit is 7. Since it takes 3 bits to represent a 7, all you need to do to convert from **Binary to Octal** is to group the bits into groups of three. As shown below, you start at the radix point and work outwards in each direction.

Binary to Octal Example #1

Convert **10110111_2** to **Octal**:

- Start at the radix point and group into **groups of 3**. Feel free to pad the left most group to make a group of 3.

$$\underbrace{0\ 1\ 0}\quad \underbrace{1\ 1\ 0}\quad \underbrace{1\ 1\ 1}_2. \quad \Rightarrow$$

- Now, all that is necessary is to convert each of the three bit groups to the equivalent octal number.

$$\underbrace{0\ 1\ 0}_{2}\quad \underbrace{1\ 1\ 0}_{6}\quad \underbrace{1\ 1\ 1_2}_{7}. \quad \Rightarrow \boxed{267_8}$$

Binary to Octal Conversion Example 2 (with a fraction)

Convert **10111.1101_2** to **Octal**

- The **left side** of the radix point is grouped into **groups of 3.** As before, padding of the **left most group** with zeros is allowed **IF DESIRED** so that the group is **3 bits wide** like the other groups.

$$10111.1101_2 = \underbrace{010}\ \underbrace{111}\ .\ 1101_2 \Rightarrow$$

- The **right side** of the radix point is also grouped into **groups of 3**. The difference here is that the right most group **MUST BE PADDED WITH 0's** so that it will be **3 bits wide** as well. There is **NO CHOICE. IT MUST BE DONE!**

$$10111.1101_2 = \underbrace{010}\ \underbrace{111}\ .\ \underbrace{110}\ \underbrace{100_2} \Rightarrow$$

- Now the number is set up to complete the conversion:

$$10111.1101_2 = \underbrace{010}_{2}\ \underbrace{111}_{7}\ .\ \underbrace{110}_{6}\ \underbrace{100_2}_{4} \Rightarrow \boxed{27.64_8}$$

Note in the example that if the right most group had not been padded with 0's, the answer would have been **27.61_8** instead of **27.64_8** and **it would have been WRONG**!

Octal to Binary Conversion

Converting to **Binary from Octal** is just as easy as can be seen in the following examples:

Octal to Binary Conversion Example 1

Convert **52.37_8** to **Binary:**

- Each Octal number is converted to its equivalent 3 bit binary number. It is that simple!

$$\underbrace{5}_{101}\ \underbrace{2}_{010}\ .\ \underbrace{3}_{011}\ \underbrace{7}_{111}{}_8 \Rightarrow \boxed{101010.011111_2}$$

- Note that the groups of 3 **MUST be padded with 0's** to make a each a complete **group of 3**.

Decimal to Octal Conversion – Sum of Weights Method

As shown earlier, conversion from **Decimal to Octal** is performed the same way as **Decimal to** Binary was, **except we use 8 in the power base vice 2**. Let's take a look at an example of **Decimal to Octal conversion**. Again, this method is called the **Sum of Weights method**.

Base 10 to Octal Sum of Weights Conversion Example 1 (with a fraction)

Convert **800.1407**$_{10}$ to **Octal** using the Sum of Weights Method

- The Sum of Weights conversion matrix is set up as before but this time powers of 8 are used.

bit	⇒	8^3	8^2	8^1	8^0	.	8^{-1}	8^{-2}
#	⇒	800				.	0.1407	
power of 8	⇒	512	64	8	1	.	0.125	0.015625
800.1407_{10}	=					.		

- The 1st division is performed as before, but this time the result can be any value from 1 to 7. In this case, it is a 1.

bit	⇒	8^3	8^2	8^1	8^0	.	8^{-1}	8^{-2}
#	⇒	800				.	0.1407	
power of 8	⇒	512	64	8	1	.	0.125	0.015625
800.1407_{10}	=	1				.		

- The remainder is placed to the right and the division is performed again.

bit	⇒	8^3	8^2	8^1	8^0	.	8^{-1}	8^{-2}
#	⇒	800	288	32		.	0.1407	
power of 8	⇒	512	64	8	1	.	0.125	0.015625
800.1407_{10}	=	1	4			.		

- The conversion can now be completed:

bit	⇒	8^3	8^2	8^1	8^0	.	8^{-1}	8^{-2}
#	⇒	800	288	32	0	.	0.1407	0.01057
power of 8	⇒	512	64	8	1	.	0.125	0.015625
800.1407_{10}	=	1	4	4	0	.	1	1

$= \boxed{1440.11_8}$

In the previous example we stopped after two digits to the right of the radix point in the interests of saving space. **You should go at least 3 digits in your work in this course.**

The Hexidecimal (HEX) System

The treatment of the **Hex number system** is a lot like the **Octal system**. However, there are two differences:

- The largest number represented in Hex is **r - 1 = 16 - 1 = 15.** Since decimal numbers only go to **9**, we need to find a way to represent **10** thru **15**. We do this with capital letters **A(10) thru F(15)**.
- It takes **4 bits** to represent a **15**, so the conversion from **Binary to Hex** requires **groups of four** vice **groups of three**.

Hex to Base 10 conversion

The process of converting from the Hex system to the decimal system is the same process as with Octal except of course we will be using powers of 16 this time.

Hex to Base 10 Conversion Example 1

Convert **$Ad85_{16}$** to **base 10**

- Note that in the problem statement above, the **lower case 'd'** was substituted for the **upper case 'D'**. This was done to prevent any possibility with confusing it with a '0'. The same thing can be done with the **lower case 'b'** instead of the **upper case 'B'** to avoid confusion with the number '8'. Of course, the lower case 'b' can now be confused with a six, so care must still be taken.
- This is primarily of importance when writing numbers but can be used in word processed calculations as well.

Example continues on the next page)

Example Continues)

- The conversion matrix is set up as before:

Base 16 #		*Power of* 16	
A	•	16^3	=
d	•	16^2	=
8	•	16^1	=
5	•	16^0	=
$Ad85_{16}$			

- The conversion can now be completed as it was in earlier examples:

Base 16 #		*Power of* 16				
A	•	16^3	=	$10 \bullet 4,096$	=	40,960
d	•	16^2	=	$13 \bullet 256$	=	3,328
8	•	16^1	=	$8 \bullet 16$	=	128
5	•	16^0	=	$5 \bullet 1$	=	5
$Ad85_{16}$					=	$44,421_{10}$

$= \boxed{44,421_{10}}$

Hex to Base 10 Conversion Example 2 (with a fraction)

Convert **AC3.2d**$_{16}$ to **Base 10.**

- Set the conversion matrix up as before. This time, the base 16 fraction has to be taken into account.

Base 16 #		*Power of* 16		
A	•	16^2	=	=
C	•	16^1	=	=
3	•	16^0	=	=
.				
2	•	16^{-1}		
d	•	16^{-2}	=	=
$Ad85_{16}$				=

Example continues on the next page)

Example Continues)

- **The conversion can now be completed carefully.**

A	$\bullet$	16^2	$=$	$10 \bullet 256$	$=$	2560	
C	$\bullet$	16^1	$=$	$12 \bullet 16$	$=$	192	
3	$\bullet$	16^0	$=$	$3 \bullet 1$	$=$	3	
.							$= \boxed{2755.176_{10}}$
2	$\bullet$	16^{-1}		$2 \bullet .0625$	$=$	0.125	
d	$\bullet$	16^{-2}	$=$	$13 \bullet .0039063$	$=$	0.050782	
$Ad85_{16}$					$=$	2755.176_{10}	

Decimal to Hex Sum of Weights Conversion Method

As with other systems discussed in this course, when a conversion is necessary from Base 10 to the Hex system, we will use the **Sum of Weights method**.

Base 10 to Hex Sum of Weights Conversion Example 1 (with a fraction)

Convert **2748.7735**$_{10}$ to **Hex**

- The **Sum of Weights** matrix constructed as before using the power of 16's.

bit	$\Rightarrow$	16^2	16^1	16^0	.	16^{-1}	16^{-2}
#	$\Rightarrow$	2748			.	0.7735	
power of 16	$\Rightarrow$	256	16	1	.	0.0625	0.0039063
2748.7735_{10}	$=$				.		

- The problem is now ready to be converted:

bit	$\Rightarrow$	16^2	16^1	16^0	.	16^{-1}	16^{-2}	
#	$\Rightarrow$	2748	188	12	.	0.7735	.0235	
power of 16	$\Rightarrow$	256	16	1	.	0.0625	0.0039063	$= \boxed{0xAbC.C6_{16}}$
2748.7735_{10}	$=$	A	b	C	.	C	6	

- Note that the answer has an alternate way of designating the number as belonging in the Base 16 system. The **0x** identification will be used in your microcontroller courses and may be used in this course if desired.

Binary to Hex conversion

We convert from Binary to Hex in much the same way that we converted from Binary to Octal. The significant difference is the fact that the largest number in the Hex system is a 15, which takes 4 binary bits to represent. So, this time we have to group the binary bits into groups of 4, vice the groups of 3 with Octal.

Binary to Hex Example 1 (with a fraction)

Convert **101101.101_2** to **Base 16**:

- Start out by grouping the bits into groups of four, starting at the radix point. The left most group is padded with zero's if desired while the right most group HAS to be padded with zero's to make a group of 4.

$$101101.101_2 \Rightarrow \underbrace{0010}\quad \underbrace{1101}.\underbrace{1010}_2$$

- It is now very easy to covert each 4 bit number into its equivalent Hex digit.

$$101101.101_2 \Rightarrow \underbrace{0010}_{2}\quad \underbrace{1101}_{d}.\underbrace{1010_2}_{A} = \boxed{0x2d.A_{16}}$$

Hex to Binary conversion

Again, we can convert back to the binary system from the Hex system by converting each Hex digit into its equivalent binary value as seen in the following example.

Hex to Binary Conversion Example 1 (with a fraction)

Convert **$5AC.3_{16}$** to **Base 2:**

$$\underbrace{5}_{0101}\quad \underbrace{A}_{1010}\quad \underbrace{C}_{1100}\;.\;\underbrace{3_{16}}_{0011} \Rightarrow \boxed{10110101100.0011_2}$$

Hex to Octal and Octal to Hex conversions

We now get to the subject of converting between **Hex and Octal** and **Octal to Hex**. Unlike conversions to and from Binary, you can't go straight to the desired system. **You must 1st GO THRU Binary.**

Hex to Octal Conversion Example 1

Convert **6E.d₁₆** to **Octal**

- First the number has to be converted to Binary:

$$\underbrace{6}_{0110}\ \underbrace{E}_{1110}\ .\ \underbrace{d_{16}}_{1101} = 1101110.1101_2$$

- Once in the binary system, it is possible to continue the conversion into **Octal**:

$$\underbrace{001}_{1}\underbrace{101}_{5}\underbrace{110}_{6}\underbrace{.}_{.}\underbrace{110}_{6}\underbrace{100_2}_{4} = \boxed{156.64_8} \Leftarrow$$

Note that two 0's had to be padded on the right.
The two 0's padded on the left were optional.

Hex to Octal Conversion Example 2

Convert **F9₁₆** to **Octal**:

- First the number has to be converted to Binary:

$$\underbrace{F}_{1111}\ \underbrace{9_{16}}_{1001} \Rightarrow 11111001_2$$

- Once in the binary system, it is possible to continue the conversion into **Octal**:

$$\underbrace{011}_{3}\underbrace{111}_{7}\underbrace{001_2}_{1} \Rightarrow \boxed{371_8}$$

Binary Coded Decimal (BCD) (8421 code)

- **BCD** is actually a **number code**, not a system, but we will treat it like a number system.
- It is a special Binary Code known as a "**Weighted Code**". It gets its name from the fact that it represents each decimal digit with its binary equivalent.
- The **BIG ADVANTAGE OF BCD over BINARY** is the **relative ease of conversion between BCD and DECIMAL compared to BINARY and DECIMAL.**
- The **BIG DISADVANTAGE** is that **it takes more bits to represent a decimal # in BCD.**
- Since it takes 4 bits to represent 8 and 9, BCD is a "4 bit code" (8421).
- **Any binary numbers above 1001 are undefined.**

Decimal to BCD conversion

BCD conversions in this direction are fairly simple. All that is necessary is to take each individual decimal digit and convert it to its equivalent 4-bit binary number. However, just because **BCD** is made up of **4-bit binary numbers**; do not confuse it with a binary number itself. It is just what its name implies. It is a decimal number CODED with Binary values.

Decimal to BCD Conversion Example 1

Convert $\mathbf{35}_{10}$ into its equivalent **BCD** value:

- Take each individual decimal value and convert it into its 4-bit equivalent:

$$\underbrace{3}_{0011}\;\underbrace{5}_{0101}{}_{10} \Rightarrow 00110101_{BCD} = \boxed{110101_{BCD}}$$

- Note that the result is identified with the **BCD subscript**. This is **NOT OPTIONAL**!

Decimal to BCD Conversion Example 2 (with a fraction)

Convert **2469.3**$_{10}$ into its equivalent **BCD** value:

- This is solved in the same way as the previous example. Note that the fraction doesn't even complicate the matter.

$$\underbrace{2}_{0010}\ \underbrace{4}_{0100}\ \underbrace{6}_{0110}\ \underbrace{9}_{1001}\ \underbrace{.}_{.}\ \underbrace{3_{10}}_{0011} \Rightarrow 0010010001101001.0011_{BCD}$$

$$= \boxed{10010001101001.0011_{BCD}}$$

BCD to Decimal conversion

Care must be taken in these conversions due to the fact that **BCD LOOKS LIKE BINARY** but it is **NOT BINARY**! Just remember that any value above a 9 is **<u>NOT defined in BCD</u>**.

BCD to Decimal Conversion Example 1

Convert **110111.011001**$_{BCD}$ to its **decimal** equivalent:

- Starting at the radix point, group the bits into groups of 4. As usual, the padded zeros on the left are optional while the ones on the right are mandatory.

$$\underbrace{0011}_{3}\ \underbrace{0111}_{7}\ \underbrace{.}_{.}\ \underbrace{0110}_{6}\ \underbrace{0100_{BCD}}_{4} = 37.64_{10}$$

HEX to BCD conversion

This is an example of **Hex** to decimal conversion. You can use the same procedure for **Octal** to **BCD** conversion. The key here as that the only way to perform these conversions is to first convert to decimal and then to **BCD**.

Hex to BCD Conversion Example 1

Convert **0x1BC** to **BCD**.

- The 1st task is to convert to decimal. This can be accomplished in a couple of ways. We can go to binary and then to decimal, or we can go straight to decimal. WE will go to go thru binary on the way to decimal.

$$\overbrace{0001}^{1}\ \overbrace{1011}^{B}\ \overbrace{1100}^{C}{}_2 = 110111100_2$$

- Now, convert to decimal:

$$\overbrace{1}^{256}\ \overbrace{1}^{128}\ 0\ \overbrace{1}^{32}\ \overbrace{1}^{16}\ \overbrace{1}^{8}\ \overbrace{1}^{4}\ 0\ 0_2$$

$$256 + 128 + 32 + 16 + 8 + 4 = 444_{10}$$

- Finally, convert to **BCD** as shown earlier.

$$\overbrace{4}^{0100}\ \overbrace{4}^{0100}\ \overbrace{4}^{0100}{}_{10} = \boxed{010001000100_{BCD}}$$

Hex to BCD Conversion Example 2

Repeat the last example but this time; don't go thru binary 1st.

- Convert the **Hex** number to decimal:

1	1	*	16^2	= 1 * 256	256
B	11	*	16^1	= 11 * 16	176
C	12	*	16^0	= 12 * 1	12
0x1BC					444_{10}

- Now, convert the **BCD**. Since we have already converted from **444**, the result should not be a surprise.

$$\underbrace{0100}_{4}\ \underbrace{0100}_{4}\ \underbrace{0100}_{4_{10}} = \boxed{10001000100_{BCD}}$$

- Note that the result was the same form both conversion paths.