Number Conversions – Part II

**This tutorial covers basic conversions for number systems. Floating point numbers, sign-extended numbers, fixed point numbers and any of the "compliment" methods will not be mentioned. This section is just a basic understanding of conversions and representations.***

There are many number systems in the world of computer science and mathematics. Some systems include the decimal system (base ten), octal system (base eight), binary system (base two) and hexadecimal system (base sixteen).

The below chart can certainly come in handy as these are most commonly seen when converting numbers. The table is representing the first 11 powers of 2 which includes 2 to the 0th power.

<table>
<thead>
<tr>
<th>Power</th>
<th>Result</th>
<th>Binary form</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^0$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$2^1$</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>$2^2$</td>
<td>4</td>
<td>100</td>
</tr>
<tr>
<td>$2^3$</td>
<td>8</td>
<td>1000</td>
</tr>
<tr>
<td>$2^4$</td>
<td>16</td>
<td>10000</td>
</tr>
<tr>
<td>$2^5$</td>
<td>32</td>
<td>100000</td>
</tr>
<tr>
<td>$2^6$</td>
<td>64</td>
<td>1000000</td>
</tr>
<tr>
<td>$2^7$</td>
<td>128</td>
<td>10000000</td>
</tr>
<tr>
<td>$2^8$</td>
<td>256</td>
<td>100000000</td>
</tr>
<tr>
<td>$2^9$</td>
<td>512</td>
<td>1000000000</td>
</tr>
<tr>
<td>$2^{10}$</td>
<td>1024</td>
<td>10000000000</td>
</tr>
</tbody>
</table>

**Representations**

Number systems each have different representations. For example, each number we see written out is implied to be in decimal unless otherwise stated. For example, we could write the following numbers in decimal this way:

9 $(9)_{10}$
10 $(10)_{10}$
11 $(11)_{10}$

etc...

Each number system has what is called a base. As mentioned earlier, decimal is a system of base 10; octal is a system of base 8; binary is a system of base 2 while hexadecimal is a system of base 16. The above numbers in the parenthesis show the base representation for decimal numbers. Either way you write them is acceptable but it is much more conventional to write them without the parenthesis and the base.

The base is also the number of digits or characters used to represent numbers in the system. Below is a chart of the digits or characters each number system uses:

**Binary**: 0|1

**Octal**: 0|1|2|3|4|5|6|7

**Decimal**: 0|1|2|3|4|5|6|7|8|9

**Hexadecimal**: 0|1|2|3|4|5|6|7|8|9|A|B|C|D|E|F

where A = 10; B = 11; C = 12; D = 13; E = 14; F = 15
So in turn, the range of digits or characters for each system is: $0 = d < \text{base}$

Here is a chart showing some decimal numbers and their equivalent representations in each of the mentioned number systems.

<table>
<thead>
<tr>
<th>Decimal</th>
<th>Binary</th>
<th>Octal</th>
<th>Hexadecimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000</td>
<td>000</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>001</td>
<td>001</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>010</td>
<td>002</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>011</td>
<td>003</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>010</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>101</td>
<td>011</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>110</td>
<td>012</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>111</td>
<td>013</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>1000</td>
<td>100</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>1001</td>
<td>101</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>1010</td>
<td>102</td>
<td>A</td>
</tr>
<tr>
<td>11</td>
<td>1011</td>
<td>103</td>
<td>B</td>
</tr>
<tr>
<td>12</td>
<td>1100</td>
<td>110</td>
<td>C</td>
</tr>
<tr>
<td>13</td>
<td>1101</td>
<td>111</td>
<td>D</td>
</tr>
<tr>
<td>14</td>
<td>1110</td>
<td>112</td>
<td>E</td>
</tr>
<tr>
<td>15</td>
<td>1111</td>
<td>113</td>
<td>F</td>
</tr>
</tbody>
</table>

Again, each can be written in the base representation. One example is:

$$(4)_{10} = (100)_2 = (4)_8 = (4)_{16}$$

### Conversions

What happens when someone only knows how to read numbers in a certain way? Most everyone knows how to read decimal numbers. So say we are given a binary number or a hexadecimal number that a person cannot read and understand. We can convert that given number into the appropriate system using a set of rules. The first type of conversion that is easiest to learn is a binary number to a decimal number.

### Binary to Decimal

To take a small example, the binary representation of the decimal number 9 is: $(1001)_2$

The algorithm to go about the conversion is:

1. Start at the rightmost bit.
2. Take that bit and multiply by $2^n$ where $n$ is the current position beginning at 0 and increasing by 1 each time. This represents a power of two.
3. Sum each terms product until all bits have been used.

So using the above methods, here is the resulting conversion to decimal above:

$$1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$
$$8 + 0 + 0 + 1 = (9)_{10}$$

A shortcut is to avoid counting the 0 bits in the sum. It saves space and time when converting but also remember that you need to increase the counter for the converting otherwise the number will not be correct.
Let's look at another example converting the below binary number: \((101101001)_2\)

Using the conversion steps above and skipping the 0 bits, here is the conversion:

\[
1 \times 2^8 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^3 + 1 \times 2^0 \\
256 + 64 + 32 + 8 + 1 = (361)_{10}
\]

**Decimal to Binary**

A more annoying and longer way of converting numbers is the decimal system to binary system. The best way of doing this is to follow these steps:

1. Divide the decimal number by 2.
2. Take the remainder and record it on the side.
3. REPEAT UNTIL the decimal number cannot be divided into anymore.
4. With the bits, record them in order from right to left as that will be the number in base two.

The example here with the decimal number 8 is a good way to start:

\[
\begin{align*}
8 / 2 &= 4 & 0 \\
4 / 2 &= 2 & 0 \\
2 / 2 &= 1 & 0 \\
1 / 2 &= 0 & 1
\end{align*}
\]

**Answer:** \((1000)_2\)

To check the result, use the previous method of converting from binary to decimal:

\[
1 \times 2^3 = (8)_{10}
\]

Let's see another example using \((125)_{10}\)

\[
\begin{align*}
125 / 2 &= 62 & 1 \\
62 / 2 &= 31 & 0 \\
31 / 2 &= 15 & 1 \\
15 / 2 &= 7 & 1 \\
7 / 2 &= 3 & 1 \\
3 / 2 &= 1 & 1 \\
1 / 2 &= 0 & 1
\end{align*}
\]

**Answer:** \((111101)_{2}\)

To check the result:

\[
1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^0 \\
64 + 32 + 16 + 8 + 4 + 1 = (125)_{10}
\]

**Binary to Hexadecimal**

Another easy conversion to do is binary to hexadecimal. The hexadecimal number system uses the digits 0 to 9 and A,B,C,D,E,F. And since the hexadecimal system is a power of 2 \((2^4)\), we can take a binary number in groups of 4 and use the appropriate hexadecimal digit in its place.
The steps to doing so are simple. Begin at the rightmost 4 bits. If there are not 4 bits, pad 0s to the left until you hit 4. Repeat the steps until all groups have been converted.

An example is below:

Take the binary number (1000101). Using the above steps, here is the conversion:

\[
\begin{array}{c|c}
0 & 100 \\
\hline
4 & 5 \\
\end{array}
\]

\text{Answer: (45)}_{16}

Take the binary number (11101011). Using the same steps, here is the conversion:

\[
\begin{array}{c|c|c}
1110 & 1011 \\
\hline
E & B \\
\end{array}
\]

\text{Answer: (EB)}_{16}

### Hexadecimal to Binary

This conversion is also simplistic. Given a hexadecimal number, simply convert each digit to its binary equivalent. Then, combine each 4 bit binary number and that is the resulting answer.

Take the hexadecimal number (A2F)\textsubscript{16} and convert it to its binary form:

\[
\begin{array}{c|c|c}
A & 2 & F \\
\hline
1010 & 0010 & 1111 \\
\end{array}
\]

\text{Answer: (1010 0010 1111)}_{2}

Take the hexadecimal number (55)\textsubscript{16} and convert it to its binary form:

\[
\begin{array}{c|c}
5 & 5 \\
\hline
0101 & 0101 \\
\end{array}
\]

\text{Answer: (1010101)}_{2}

### Binary to Octal

Since the octal system is again a power of two (2\(^3\)), we can take group the bits into groups of 3 and represent each group as an octal digit. The steps are the same for the binary to hexadecimal conversions except we are dealing with the octal base now.

Take the binary number (10011)\textsubscript{2} and convert it to octal:

\[
\begin{array}{c|c}
010 & 011 \\
\hline
2 & 3 \\
\end{array}
\]

\text{Answer: (23)}_{8}

Take the binary number (10010101011)\textsubscript{2} and convert it to octal:

\[
\begin{array}{c|c|c|c}
010 & 010 & 101 & 011 \\
\hline
2 & 2 & 5 & 3 \\
\end{array}
\]

\text{Answer: (2253)}_{8}
Octal to Binary
To go from octal to binary, simply reverse the above algorithm and represent each octal digit in its three bit binary form.

Take the octal number \((742)_8\) and convert it to binary:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>111</td>
<td>100</td>
<td>010</td>
</tr>
</tbody>
</table>

**Answer:** \((111100010)_2\)

Take the octal number \((1234567)_8\) and convert it to binary:

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>001</td>
<td>010</td>
<td>011</td>
<td>100</td>
<td>101</td>
<td>110</td>
<td>111</td>
</tr>
</tbody>
</table>

**Answer:** \((0010100110010111011)_2\)

Decimal to Octal
Similar to the decimal to binary method, you must divide each number by 8 and record the remainder. This continues until the number cannot be divided into anymore.

Take the decimal number \((18)_{10}\) and convert it to octal:

\[
18 / 8 = 2 \quad 2 \\
2 / 8 = 0 \quad 2
\]

**Answer:** \((22)_8\)

Take the decimal number \((46)_{10}\) and convert it to octal:

\[
46 / 8 = 5 \quad 6 \\
5 / 8 = 0 \quad 5
\]

**Answer:** \((56)_8\)

Octal to Decimal
Similar to the binary to decimal method, simply take each digit in the octal base and multiply by the power of 8.

Take the octal number \((764)_8\) and convert it to decimal:

\[
7 \times 8^2 + 6 \times 8^1 + 4 \times 8^0 \\
448 + 48 + 4 = 500
\]

**Answer:** \((500)_{10}\)

Take the octal number \((5771)_8\) and convert it to decimal:

\[
5 \times 8^3 + 7 \times 8^2 + 7 \times 8^1 + 1 \times 8^0 \\
2560 + 448 + 56 + 1 = 3065
\]

**Answer:** \((3065)_{10}\)