CHAPTER V

NUMBER SYSTEMS AND ARITHMETIC
• Decimal number expansion

\[ 73625_{10} = (7 \times 10^4) + (3 \times 10^3) + (6 \times 10^2) + (2 \times 10^1) + (5 \times 10^0) \]

• Binary number representation

\[ 10110_2 = (1 \times 2^4) + (0 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0) = 22_{10} \]

• Hexadecimal number representation

\[ 3E4B8_{16} = (3 \times 16^4) + (14 \times 16^3) + (4 \times 16^2) + (11 \times 16^1) + (8 \times 16^0) \]
\[ = 255160_{10} \]
Radix-10 Representation

\[ 73625.4385_{10} \]

\[
\begin{array}{cccccccccccc}
10^5 & 10^4 & 10^3 & 10^2 & 10^1 & 10^0 & 10^{-1} & 10^{-2} & 10^{-3} & 10^{-4} & 10^{-5} \\
\hline
& 0 & 7 & 3 & 6 & 2 & 5 & . & 4 & 3 & 8 & 5 & 0 & \ldots
\end{array}
\]

\[ 73625.4385_{10} = (7 \times 10^4) + (3 \times 10^3) + (6 \times 10^2) + (2 \times 10^1) + (5 \times 10^0) + (4 \times 10^{-1}) + (3 \times 10^{-2}) + (8 \times 10^{-3}) + (5 \times 10^{-4}) \]
Radix-2 Representation

\[ 10110.0011_2 \]

### Radix-2 Representation

<table>
<thead>
<tr>
<th></th>
<th>(2^5)</th>
<th>(2^4)</th>
<th>(2^3)</th>
<th>(2^2)</th>
<th>(2^1)</th>
<th>(2^0)</th>
<th>(2^{-1})</th>
<th>(2^{-2})</th>
<th>(2^{-3})</th>
<th>(2^{-4})</th>
<th>(2^{-5})</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSB</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>.</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>LSB</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

\[
10110.0011_2 = (1 \times 2^4) + (0 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0) \\
+ (0 \times 2^{-1}) + (0 \times 2^{-2}) + (1 \times 2^{-3}) + (1 \times 2^{-4}) \\
= 22.1875_{10}
\]
Radix-8 Representation

\[ 26516.1731_8 \]

\[
\begin{array}{cccccccccc}
8^5 & 8^4 & 8^3 & 8^2 & 8^1 & 8^0 & 8^{-1} & 8^{-2} & 8^{-3} & 8^{-4} & 8^{-5} \\
0 & 2 & 6 & 5 & 1 & 6 & . & 1 & 7 & 3 & 1 & 0 \\
\end{array}
\]

\[
26516.1731_8 = (2 \times 8^4) + (6 \times 8^3) + (5 \times 8^2) + (1 \times 8^1) + (6 \times 8^0) \\
+ (1 \times 8^{-1}) + (7 \times 8^{-2}) + (3 \times 8^{-3}) + (1 \times 8^{-4}) \\
= 11598.24_{10}
\]
Radix-16 Representation

19AD6.F411\textsubscript{16}

\[16^5 16^4 16^3 16^2 16^1 16^0 16^{-1} 16^{-2} 16^{-3} 16^{-4} 16^{-5}\]

\[
\begin{array}{cccccccccc}
0 & 1 & 9 & A & D & 6 & . & F & 4 & 1 & 1 & 0 & \cdots \\
\end{array}
\]

\[
19AD6.F411\textsubscript{16} = (1 \times 16^4) + (9 \times 16^3) + (A \times 16^2) + (D \times 16^1) + (6 \times 16^0) \\
+ (F \times 16^{-1}) + (4 \times 16^{-2}) + (1 \times 16^{-3}) + (1 \times 16^{-4})
\]

\[\approx 105174.95\textsubscript{10}\]
NUMBER SYSTEMS

BINARY <-> HEXADECIMAL

BINARY <-> HEXADECIMAL

BINARY -> HEXADECIMAL

Group binary by 4 bits from radix point

Examples:

0111 1011₂ = 7B₁₆

0111 1011₂ = 7B₁₆

10 1010 0110.1100 01₂ = 2A6.C4₁₆

2 A 6 C 4
NUMBER SYSTEMS
BINARY <-> OCTAL

BINARY <-> OCTAL

Group binary bits by 3 from LSB

Examples:

000₂ = 0₈
001₂ = 1₈
010₂ = 2₈
011₂ = 3₈
100₂ = 4₈
101₂ = 5₈
110₂ = 6₈
111₂ = 7₈

010100110₂ = 246₈

1010111011.01111₂ = 2573.36₈

R.M. Dansereau; v.1.0
• Perform radix-2 expansion

  • Multiply each bit in the binary number by 2 to the power of its place.
  
  Then sum all of the values to get the decimal value.

Examples:

\[ 10111_2 = (1 \times 2^4) + (0 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) = 23_{10} \]

\[ 10110.0011_2 = (1 \times 2^4) + (0 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0) + (0 \times 2^{-1}) + (0 \times 2^{-2}) + (1 \times 2^{-3}) + (1 \times 2^{-4}) = 22.1875_{10} \]
Example: Convert $41.828125_{10}$

**Integer part:**
- Modulo division of decimal integer by 2 to get each bit, starting with LSB.

**Fraction part:**
- Multiplication decimal fraction by 2 and collect resulting integers, starting with MSB.

Therefore $41.828125_{10} = 101001.110101_{2}$
Floating point numbers can be represented with a sign bit, a fraction (often referred to as the mantissa), and an exponent.

- Example 1: $-267.426 = -0.267426 \times 10^3$, where the sign is negative, the fraction is 0.267426 and the exponent is 3.
- Example 2: $0101011.1001 = 0.1010111 \times 2^6$, where the sign is positive, the fraction is 0.1010111, and the exponent is 0110.

Sample IEEE Floating-Point Formats

<table>
<thead>
<tr>
<th>32-bit</th>
<th>s</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>8</td>
<td>23</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>64-bit</th>
<th>s</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>11</td>
<td>52</td>
</tr>
</tbody>
</table>
The range for an $n$-bit radix-$r$ unsigned integer is

$$[0, r_{10}^n - 1]$$

Example: For a 16-bit binary unsigned integer, the range is

$$[0, 2^{16} - 1] = [0, 65535]$$

which has a binary representation of

- $0000 0000 0000 0000 = 0$
- $0000 0000 0000 0001 = 1$
- $0000 0000 0000 0010 = 2$
- \ldots
- $1111 1111 1111 1110 = 65534$
- $1111 1111 1111 1111 = 65535$
The range for an $n$-bit radix-$r$ signed integer is

$$[-r_{10}^{n-1}, r_{10}^{n-1} - 1]$$

The most-significant bit is used as a sign bit, where 0 indicates a positive integer and 1 indicates a negative integer.

Example: For a 16-bit binary signed integer, the range is

$$[-2^{16-1}, 2^{16-1} - 1] = [-32768, 32767]$$
There are a number of different representations for signed integers, each which has its own advantage

- Signed-magnitude representation:
  - \(1010\ 0001\ 0110\ 1111\)

- Signed-1’s complement representation:
  - \(1101\ 1110\ 1001\ 0000\)

- Signed-2’s complement representation:
  - \(1101\ 1110\ 1001\ 0001\)

The above examples are all the same number, \(-8559_{10}\).
The **signed-magnitude** binary integer representation is just like the **unsigned representation** with the addition of a **sign bit**.

For instance, using 8-bits, the number \(-6_{10}\) can be represented as the 7-bit magnitude of \(6_{10}\) using

\[
000\ 0110
\]

and then the sign bit appended to the MSB to form

\[
1000\ 0110
\]
The **radix complement**, or \( r \)'s complement, of an integer representation for an \( n \)-digit integer is defined as

\[
10^n_{10} - \text{number}_{10}
\]

The **diminished radix complement**, or \((r - 1)'\text{s complement}\), of an integer representation for an \( n \)-digit integer is defined as

\[
(r^n_{10} - 1_{10}) - \text{number}_{10}
\]

**Example:** Find the \( r \)'s and \((r - 1)'\text{s complement}\) for \(3764_{10}\)

\[
\begin{align*}
\text{r's complement} & \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad (r - 1)'\text{s complement} \\
10^5 - 3764 &= 96236 & (10^5 - 1) - 3764 &= 96235
\end{align*}
\]
The 1’s complement (diminished radix complement) binary integer representation for an \( n \)-bit integer is defined as

\[
(2^n_{10} - 1_{10}) - \text{number}_{10}
\]

In essence, this takes the positive version of the number and flips all of the bits.

- For instance, using 8-bits, the number \(-6_{10}\) can be represented as the 8-bit positive number \(6_{10}\) using

\[
0000\ 0110
\]

and then each of the bits flipped to form the 1’s complement

\[
1111\ 1001
\]
The **2’s complement** (radix complement) binary integer representation for an $n$-bit integer is defined as

$$2^n_{10} - \text{number}_{10}$$

In essence, this takes the 1’s complement and adds one.

For instance, using 8-bits, the number $-6_{10}$ can be represented as the 8-bit positive number $6_{10}$ using

$$0000\ 0110$$

and then each of the bits flipped to form the 1’s complement

$$1111\ 1001$$

and then add 1 to form the 2’s complement

$$1111\ 1010$$
Below are some examples for the signed binary numbers using 6 bits.

<table>
<thead>
<tr>
<th>Decimal</th>
<th>Signed-magnitude</th>
<th>1’s complement</th>
<th>2’s complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00 0000</td>
<td>00 0000</td>
<td>00 0000</td>
</tr>
<tr>
<td>1</td>
<td>00 0001</td>
<td>00 0001</td>
<td>00 0001</td>
</tr>
<tr>
<td>-1</td>
<td>10 0001</td>
<td>11 1110</td>
<td>11 1111</td>
</tr>
<tr>
<td>5</td>
<td>00 0101</td>
<td>00 0101</td>
<td>00 0101</td>
</tr>
<tr>
<td>-5</td>
<td>10 0101</td>
<td>11 1010</td>
<td>11 1011</td>
</tr>
<tr>
<td>12</td>
<td>00 1100</td>
<td>00 1100</td>
<td>00 1100</td>
</tr>
<tr>
<td>-12</td>
<td>10 1100</td>
<td>11 0011</td>
<td>11 0100</td>
</tr>
<tr>
<td>15</td>
<td>00 1111</td>
<td>00 1111</td>
<td>00 1111</td>
</tr>
<tr>
<td>-15</td>
<td>10 1111</td>
<td>11 0000</td>
<td>11 0001</td>
</tr>
<tr>
<td>16</td>
<td>01 0000</td>
<td>01 0000</td>
<td>01 0000</td>
</tr>
<tr>
<td>-16</td>
<td>11 0000</td>
<td>10 1111</td>
<td>11 0000</td>
</tr>
</tbody>
</table>

Notice that all representations are the same for positive numbers!!!!
Unsigned binary addition follows the standard rules of addition.

Examples

\[
\begin{align*}
1111 & 0100 \quad \text{Carries} & 0000 & 0010 \quad \text{Carries} \\
0011 & 1011 & & 1011 \ 1001 \\
+ & 0111 & 1010 & + & 0100 \ 0101 \\
\hline
1011 & 0101 & & 1111 \ 1110
\end{align*}
\]

\[
\begin{align*}
1111 & 0000 \quad \text{Carries} & 1110 & 0000 \ 0000.0000 \quad \text{Carries} \\
1111 & 1001 \ & & 0101 \ 1000 \ 1001.1001 \\
+ & 0100 & 1000 & + & 0011 \ 0011 \ 0100.01 \\
\hline
1 \ 0100 \ 0000 \ & & 1000 \ 1011 \ 1101.1101
\end{align*}
\]
Unsigned binary subtraction follows the standard rules.

Examples

\[
\begin{array}{cccc}
0000\ 0000 & \text{Borrows} & 1000\ 1000 & \text{Borrows} \\
1111\ 1001 & & 1011\ 1001 & \\
-\ 0100\ 1000 & & -\ 0100\ 0101 & \\
\hline
1011\ 0001 & & 0111\ 0100 & \\
\end{array}
\]

\[
\begin{array}{cccc}
1000\ 0000 & \text{Borrows} & 0100\ 1110\ 1000.1000 & \text{Borrows} \\
0011\ 1011 & & 0101\ 1000\ 1001.1001 & \\
-\ 0111\ 1010 & & -\ 0011\ 0011\ 0100.01 & \\
\hline
1100\ 0001 & & 0010\ 0101\ 0101.0101 & \\
\end{array}
\]
Signed-magnitude

- Add magnitudes if signs are the same, give result the sign
- Subtract magnitudes if signs are different. Absence or presence of an end borrow determines the resulting sign compared to the augend. If negative, then a 2’s complement correction must be taken.

2’s complement

- Add the numbers using normal addition rules. Carry out bit is discarded.

1’s complement

- Easiest to convert to 2’s complement, perform the addition, and then convert back to 1’s complement. This is done as follows:
  - Add 1 to each integer, add the integers, subtract 1 from the result
Typically want to do addition or subtraction of $A$ and $B$ as follows.

\[
\text{SUM} = A + B \\
\text{DIFFERENCE} = A - B
\]

If we use 2’s complement, we can make life easy on us since addition and subtraction are done in the same manner: with addition only!!!

A subtraction can be re-represented as follows.

\[
\text{SUM} = A + (-B)
\]

Or in general any two numbers can be added as follows.

\[
\text{SUM} = (\pm A) + (\pm B)
\]
Subtraction of signed numbers can best be done with 2’s complement. Performed by taking the 2’s complement of the subtrahend and then performing addition (including the sign bit).

Example:

\[
\begin{align*}
59 & \quad 0011\,1011 \\
-122 & \quad 0111\,1010 \\
\hline
-63 & \quad 1100\,0001
\end{align*}
\]

Binary:

\[
\begin{align*}
0011\,1100 \quad \text{Carries} \\
0011\,1011 & + 0111\,1010 \\
\hline
1100\,0001 & = -(0011\,1111) = -63
\end{align*}
\]