Number system
Outline

• Number System
  – Decimal, Binary, Octal, Hex

• Conversion (one to another)
  – Decimal to Binary, Octal, Hex & Vice Versa
  – Binary to HEX & vice versa

• Other representation
  – Signed, Unsigned, Complement

• Operation
  – Add, Sub, Mul, Div, Mod

• How to handle real number efficiently?
  – Float, Double
What Digit? => Number System

- Famous Number System: Dec, Rom, Bin
- Decimal System: 0 -9
  - May evolves: because human have 10 finger
- Roman System
  - May evolves to make easy to look and feel
  - Pre/Post Concept: (IV, V & VI) is (5-1, 5 & 5+1)
- Binary System, Others (Oct, Hex)
  - One can cut an apple in to two
Significant Digits

Binary: 11101101

Most significant digit  Least significant digit

Decimal: 1063079

Most significant digit  Least significant digit
Decimal (base 10)

• Uses positional representation
• Each digit corresponds to a power of 10 based on its position in the number
• The powers of 10 increment from 0, 1, 2, etc. as you move right to left
  \[ 1,479 = 1 \times 10^3 + 4 \times 10^2 + 7 \times 10^1 + 9 \times 10^0 \]
Binary (base 2)

- Two digits: 0, 1
- To make the binary numbers more readable, the digits are often put in groups of 4

\[-1010 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0\]
\[= 8 + 2\]
\[= 10\]

\[-1100\;1001 = 1 \times 2^7 + 1 \times 2^6 + 1 \times 2^3 + 1 \times 2^0\]
\[= 128 + 64 + 8 + 1\]
\[= 201\]
How to Encode Numbers: Binary Numbers

• Working with binary numbers
  – In base ten, helps to know powers of 10
    • one, ten, hundred, thousand, ten thousand, ...
  – In base two, helps to know powers of 2
    • one, two, four, eight, sixteen, thirty two, sixty four, one hundred twenty eight
  • Count up by powers of two

\[
\begin{array}{cccccccccc}
2^9 & 2^8 & 2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\
512 & 256 & 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1
\end{array}
\]
Octal (base 8)

- Shorter & easier to read than binary
- 8 digits: 0, 1, 2, 3, 4, 5, 6, 7,
- Octal numbers

\[136_8 = 1 \times 8^2 + 3 \times 8^1 + 6 \times 8^0\]
\[= 1 \times 64 + 3 \times 8 + 6 \times 1\]
\[= 94_{10}\]
Hexadecimal (base 16)

• Shorter & easier to read than binary
• 16 digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
• “0x” often precedes hexadecimal numbers

0x123 = 1 * 16^2 + 2 * 16^1 + 3 * 16^0
       = 1 * 256 + 2 * 16 + 3 * 1
       = 256 + 32 + 3
       = 291
<table>
<thead>
<tr>
<th>Decimal</th>
<th>Binary</th>
<th>Octal</th>
<th>Hexadecimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00000</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>00001</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>00010</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>00011</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>00100</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>00101</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>00110</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>00111</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>01000</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>01001</td>
<td>11</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>01010</td>
<td>12</td>
<td>A</td>
</tr>
<tr>
<td>11</td>
<td>01011</td>
<td>13</td>
<td>B</td>
</tr>
<tr>
<td>12</td>
<td>01100</td>
<td>14</td>
<td>C</td>
</tr>
<tr>
<td>13</td>
<td>01101</td>
<td>15</td>
<td>D</td>
</tr>
<tr>
<td>14</td>
<td>01110</td>
<td>16</td>
<td>E</td>
</tr>
<tr>
<td>15</td>
<td>01111</td>
<td>17</td>
<td>F</td>
</tr>
<tr>
<td>16</td>
<td>10000</td>
<td>20</td>
<td>10</td>
</tr>
</tbody>
</table>
Fractional Number

• Point: Decimal Point, Binary Point, Hexadecimal point

• Decimal
  \[ 247.75 = 2 \times 10^2 + 4 \times 10^1 + 7 \times 10^0 + 7 \times 10^{-1} + 5 \times 10^{-2} \]

• Binary
  \[ 10.101 = 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} \]

• Hexadecimal
  \[ 6A.7D = 6 \times 16^1 + 10 \times 16^0 + 7 \times 16^{-1} + D \times 16^{-2} \]
Converting To and From Decimal

Converting
To
and
From
Decimal

Successive Division

Weighted Multiplication

Octal
0 1 2 3 4 5 6 7

Decimal
0 1 2 3 4 5 6 7 8 9

Successive Division

Weighted Multiplication

Binary
0 1

Successive Division

Weighted Multiplication

Hexadecimal
0 1 2 3 4 5 6 7 8 9 A B C D E F
Decimal ↔ Binary

**Base\textsubscript{10}** DECIMAL ↔ **Base\textsubscript{2}** BINARY

a) Divide the decimal number by 2; the remainder is the LSB of the binary number.

b) If the quotation is zero, the conversion is complete. Otherwise repeat step (a) using the quotation as the decimal number. The new remainder is the next most significant bit of the binary number.

**Base\textsubscript{2}** BINARY ↔ **Base\textsubscript{10}** DECIMAL

a) Multiply each bit of the binary number by its corresponding bit-weighting factor (i.e., Bit-0→2\textsuperscript{0}=1; Bit-1→2\textsuperscript{1}=2; Bit-2→2\textsuperscript{2}=4; etc).

b) Sum up all of the products in step (a) to get the decimal number.
**Decimal to Binary: Subtraction Method**

- **Goal**
  - Good for human
  - Get the binary weights to add up to the decimal quantity
    - Work from left to right
    - (Right to left – may fill in 1s that shouldn’t have been there – try it).

Desired decimal number: **12**

```
<table>
<thead>
<tr>
<th></th>
<th>32</th>
<th>16</th>
<th>8</th>
<th>4</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>16</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>16</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>16</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>16</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>16</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
```

- =32 too much
- =16 too much
- =8
- ok, keep going
- =8+4=12
- DONE
- answer
Decimal to Binary : Division Method

• Good for computer: Divide decimal number by 2 and insert remainder into new binary number.
  – Continue dividing quotient by 2 until the quotient is 0.
• Example: Convert decimal number 12 to binary

\[ \begin{align*}
12 \div 2 &= (\text{Quo}=6, \text{Rem}=0) \quad \text{LSB} \\
6 \div 2 &= (\text{Quo}=3, \text{Rem}=0) \\
3 \div 2 &= (\text{Quo}=1,\text{Rem}=1) \\
1 \div 2 &= (\text{Quo}=0, \text{Rem}=1) \quad \text{MSB}
\end{align*} \]

\[12_{10} = 1100_2\]
Conversion Process Decimal ↔ Base$_N$

a) Divide the decimal number by $N$; the remainder is the LSB of the ANY BASE Number.

b) If the quotient is zero, the conversion is complete. Otherwise repeat step (a) using the quotient as the decimal number. The new remainder is the next most significant bit of the ANY BASE number.

a) Multiply each bit of the ANY BASE number by its corresponding bit-weighting factor (i.e., Bit-0 → $N^0$; Bit-1 → $N^1$; Bit-2 → $N^2$; etc).

b) Sum up all of the products in step (a) to get the decimal number.
Decimal ↔ Octal Conversion

The Process: Successive Division

- Divide number by 8; R is the LSB of the octal number
- While Q is 0
  - Using the Q as the decimal number.
  - New remainder is MSB of the octal number.

\[
\begin{array}{rcl}
11 & \equiv & 3 \\
94 & \equiv & 136_8
\end{array}
\]

\[
\begin{array}{rcl}
11 & \equiv & 3 \\
94 & \equiv & 136_8
\end{array}
\]

\[
\begin{array}{rcl}
11 & \equiv & 3 \\
94 & \equiv & 136_8
\end{array}
\]
Decimal $\leftrightarrow$ Hexadecimal Conversion

The Process: Successive Division

- Divide number by $16$; $R$ is the LSB of the $\text{hex}$ number
- While $Q$ is $0$
  - Using the $Q$ as the decimal number.
  - New remainder is MSB of the $\text{hex}$ number.

\[
\begin{array}{c}
\frac{5}{16} \\
\underline{16 \times 5} & 94 \\
0 & 0 \\
\frac{5}{16} & r = 5 \quad \leftarrow \text{MSB}
\end{array}
\]

\[
94_{10} = 5E_{16}
\]
Example: Hex → Octal

Example:

Convert the hexadecimal number $5A_{16}$ into its octal equivalent.

Solution:

First convert the hexadecimal number into its decimal equivalent, then convert the decimal number into its octal equivalent.

\[
\begin{array}{c|c}
5 & A \\
\hline
16 & 16 \\
\hline
16 & 1 \\
\hline
80 & + 10 = 90_{10}
\end{array}
\]

\[
\begin{array}{c|c}
11 & 8 \overline{)90} \quad r = 2 \quad \leftarrow \text{LSB} \\
1 & 8 \overline{)11} \quad r = 3 \\
0 & 8 \overline{)1} \quad r = 1 \quad \leftarrow \text{MSB}
\end{array}
\]

\[\therefore 5A_{16} = 132_8\]
Example: Octal → Binary

Example:

Convert the octal number $132_8$ into its binary equivalent.

Solution:

First convert the octal number into its decimal equivalent, then convert the decimal number into its binary equivalent.

\[
\begin{array}{c|cccc}
& 8^2 & 8^1 & 8^0 \\
\hline
132 & 64 & 8 & 1 \\
\end{array}
\]

\[
64 + 24 + 2 = 90_{10}
\]

\[
\begin{array}{c|c}
2 & 45 \\
2 & 22 \\
2 & 11 \\
2 & 5 \\
2 & 11 \\
2 & 2 \\
2 & 1 \\
& 1 \\
\end{array}
\]

\[
\begin{array}{c|c}
 & r \\
\hline
r = 0 & \text{← LSB} \\
r = 1 & \text{← LSB} \\
r = 0 & \text{← LSB} \\
r = 1 & \text{← MSB} \\
r = 0 & \text{← MSB} \\
r = 1 & \text{← MSB} \\
r = 1 & \text{← MSB} \\
\end{array}
\]

\[
132_8 = 1011010_2
\]
Binary ↔ Octal ↔ Hex Shortcut

• Relation
  • Binary, octal, and hex number systems
  • All powers of two
• Exploit (This Relation)
  • Make conversion easier.
Substitution Code

Convert $010101101010111001101010_2$ to hex using the 4-bit substitution code:

\[
\begin{array}{cccccc}
5 & 6 & A & E & 6 & A \\
0101 & 0110 & 1010 & 1110 & 0110 & 1010 \\
\end{array}
\]

$56AE6A_{16}$
Substitution code can also be used to convert binary to octal by using 3-bit groupings:

```
2 5 5 2 7 1 5 2
010 101 101 010 111 001 101 010
```

\[25527152_8\]
Other Representation

• Signed & Unsigned Number
• Signed number last bit (one MSB) is signed bit
  Assume: 8 bit number
  Unsigned 12 : 0000 1100
  Signed +12 : 0000 1100
  Signed -12 : 1000 1100

• Complement number
  Unsigned binary 12 = 00001100
  1’s Complement of 12 = 1111 0011
Thanks