Digital Design & Computer Arch.

Reading: Binary Numbers

Required Reading
for Week 1
25-26 February 2021
Spring 2021
Binary Numbers

Design of Digital Circuits 2016
Srdjan Capkun
Frank K. Gürkaynak

http://www.syssec.ethz.ch/education/Digitaltechnik_16
In This Lecture

- How to express numbers using only 1s and 0s
- Using hexadecimal numbers to express binary numbers
- Different systems to express negative numbers
- Adding and subtracting with binary numbers
Number Systems

- Decimal Numbers

\[ 5374_{10} = \]

- Binary Numbers

\[ 1101_{2} = \]
Number Systems

- **Decimal Numbers**

\[ 5374_{10} = 5 \times 10^3 + 3 \times 10^2 + 7 \times 10^1 + 4 \times 10^0 \]

- **Binary Numbers**

\[ 1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13_{10} \]
## Powers of two

<table>
<thead>
<tr>
<th>$2^0$</th>
<th>=</th>
<th>$2^8$</th>
<th>=</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^1$</td>
<td>=</td>
<td>$2^9$</td>
<td>=</td>
</tr>
<tr>
<td>$2^2$</td>
<td>=</td>
<td>$2^{10}$</td>
<td>=</td>
</tr>
<tr>
<td>$2^3$</td>
<td>=</td>
<td>$2^{11}$</td>
<td>=</td>
</tr>
<tr>
<td>$2^4$</td>
<td>=</td>
<td>$2^{12}$</td>
<td>=</td>
</tr>
<tr>
<td>$2^5$</td>
<td>=</td>
<td>$2^{13}$</td>
<td>=</td>
</tr>
<tr>
<td>$2^6$</td>
<td>=</td>
<td>$2^{14}$</td>
<td>=</td>
</tr>
<tr>
<td>$2^7$</td>
<td>=</td>
<td>$2^{15}$</td>
<td>=</td>
</tr>
</tbody>
</table>
## Powers of two

<table>
<thead>
<tr>
<th>$2^0$</th>
<th>=</th>
<th>1</th>
<th>$2^8$</th>
<th>=</th>
<th>256</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^1$</td>
<td>=</td>
<td>2</td>
<td>$2^9$</td>
<td>=</td>
<td>512</td>
</tr>
<tr>
<td>$2^2$</td>
<td>=</td>
<td>4</td>
<td>$2^10$</td>
<td>=</td>
<td>1024</td>
</tr>
<tr>
<td>$2^3$</td>
<td>=</td>
<td>8</td>
<td>$2^11$</td>
<td>=</td>
<td>2048</td>
</tr>
<tr>
<td>$2^4$</td>
<td>=</td>
<td>16</td>
<td>$2^12$</td>
<td>=</td>
<td>4096</td>
</tr>
<tr>
<td>$2^5$</td>
<td>=</td>
<td>32</td>
<td>$2^13$</td>
<td>=</td>
<td>8192</td>
</tr>
<tr>
<td>$2^6$</td>
<td>=</td>
<td>64</td>
<td>$2^14$</td>
<td>=</td>
<td>16384</td>
</tr>
<tr>
<td>$2^7$</td>
<td>=</td>
<td>128</td>
<td>$2^15$</td>
<td>=</td>
<td>32768</td>
</tr>
</tbody>
</table>

Handy to memorize up to $2^{15}$
Binary to Decimal Conversion

- Convert $10011_2$ to decimal
Binary to Decimal Conversion

- Convert $10011_2$ to decimal

$$2^4 \times 1 + 2^3 \times 0 + 2^2 \times 0 + 2^1 \times 1 + 2^0 \times 1 =$$
Binary to Decimal Conversion

Convert $10011_2$ to decimal

\[2^4 \times 1 + 2^3 \times 0 + 2^2 \times 0 + 2^1 \times 1 + 2^0 \times 1 =
\]

\[16 \times 1 + 8 \times 0 + 4 \times 0 + 2 \times 1 + 1 \times 1 =
\]

\[16 + 0 + 0 + 2 + 1 = 19_{10}\]
Decimal to Binary Conversion

- Convert $47_{10}$ to binary
Decimal to Binary Conversion

- Convert $47_{10}$ to binary
  - Start with $2^6 = 64$ is $64 \leq 47$? no do nothing
  - Now $2^5 = 32$
Decimal to Binary Conversion

**Convert 47\textsubscript{10} to binary**

- Start with $2^6 = 64$ is $64 \leq 47$? no do nothing
- Now $2^5 = 32$ is $32 \leq 47$? yes subtract $47 - 32 = 15$
- Now $2^4 = 16$ is $16 \leq 15$? no do nothing
- Now $2^3 = 8$ is $8 \leq 15$? yes subtract $15 - 8 = 7$
- Now $2^2 = 4$ is $4 \leq 7$? yes subtract $7 - 4 = 3$
- Now $2^1 = 2$ is $2 \leq 3$? yes subtract $3 - 2 = 1$
- Now $2^0 = 1$ is $1 \leq 1$? yes we are done
Decimal to binary conversion

Convert $47_{10}$ to binary

- Start with $2^6 = 64$ is $64 \leq 47$ ? no 0 do nothing
- Now $2^5 = 32$ is $32 \leq 47$ ? yes 1 subtract $47 - 32 = 15$
- Now $2^4 = 16$ is $16 \leq 15$ ? no 0 do nothing
- Now $2^3 = 8$ is $8 \leq 15$ ? yes 1 subtract $15 - 8 = 7$
- Now $2^2 = 4$ is $4 \leq 7$ ? yes 1 subtract $7 - 4 = 3$
- Now $2^1 = 2$ is $2 \leq 3$ ? yes 1 subtract $3 - 2 = 1$
- Now $2^0 = 1$ is $1 \leq 1$ ? yes 1 we are done

Result is $0101111_2$
Binary Values and Range

- **N-digit decimal number**
  - How many values? \(10^N\)
  - Range? [0, \(10^N - 1\)]
  - Example: 3-digit decimal number
    - \(10^3 = 1000\) possible values
    - Range: [0, 999]

- **N-bit binary number**
  - How many values? \(2^N\)
  - Range: [0, \(2^N - 1\)]
  - Example: 3-digit binary number
    - \(2^3 = 8\) possible values
    - Range: [0, 7] = [000_2 to 111_2]
# Hexadecimal (Base-16) Numbers

<table>
<thead>
<tr>
<th>Decimal</th>
<th>Hexadecimal</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td></td>
</tr>
</tbody>
</table>
# Hexadecimal (Base-16) Numbers

<table>
<thead>
<tr>
<th>Decimal</th>
<th>Hexadecimal</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0000</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0001</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0010</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0011</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>0100</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>0101</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>0110</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>0111</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>1000</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>1001</td>
</tr>
<tr>
<td>10</td>
<td>A</td>
<td>1010</td>
</tr>
<tr>
<td>11</td>
<td>B</td>
<td>1011</td>
</tr>
<tr>
<td>12</td>
<td>C</td>
<td>1100</td>
</tr>
<tr>
<td>13</td>
<td>D</td>
<td>1101</td>
</tr>
<tr>
<td>14</td>
<td>E</td>
<td>1110</td>
</tr>
<tr>
<td>15</td>
<td>F</td>
<td>1111</td>
</tr>
</tbody>
</table>
Hexadecimal Numbers

- Binary numbers can be pretty long.
- A neat trick is to use base 16
- How many binary digits represent a hexadecimal digit?
  4 (since $2^4 = 16$)
- Example 32 bit number:
  0101 1101 0111 0001 1001 1111 1010 0110
Hexadecimal Numbers

- Binary numbers can be pretty long.
- A neat trick is to use base 16
- How many binary digits represent a hexadecimal digit? 4 (since $2^4 = 16$)
- Example 32 bit number:

```
0101 1101 0111 0001 1001 1111 1010 0110
5    D    7    1    9    F    A    6
```
Hexadecimal Numbers

- Binary numbers can be pretty long.
- A neat trick is to use base 16
- How many binary digits represent a hexadecimal digit?
  - 4 (since $2^4 = 16$)

- Example 32 bit number:
  - 0101 1101 0111 0001 1001 1111 1010 0110
  - 5  D  7  1  9  F  A  6

- The other way is just as simple
  - C  E  2  8  3  5  4  B
Hexadecimal Numbers

- Binary numbers can be pretty long.
- A neat trick is to use base 16
- How many binary digits represent a hexadecimal digit?
  4 (since $2^4 = 16$)

Example 32 bit number:

```
0101 1101 0111 0001 1001 1111 1010 0110
5    D    7    1    9    F    A    6
```

The other way is just as simple

```
C   E   2   8   3   5   4   B
1100 1110 0010 1000 0011 0101 0100 1011
```
Hexadecimal to Decimal Conversion

- Convert $4AF_{16}$ (or 0x4AF) to decimal
Hexadecimal to decimal conversion

Convert $4AF_{16}$ (or $0x4AF$) to decimal

$$16^2 \times 4 + 16^1 \times A + 16^0 \times F =$$

$$256 \times 4 + 16 \times 10 + 1 \times 15 =$$

$$1024 + 160 + 15 = 1199_{10}$$
Bits, Bytes, Nibbles...

10010110

- most significant bit
- least significant bit

byte

10010110

nibble

CEBF9AD7

- most significant byte
- least significant byte
Powers of Two

- $2^{10} = 1$ kilo $\approx 1000$ (1024)

- $2^{20} = 1$ mega $\approx 1$ million (1,048,576)

- $2^{30} = 1$ giga $\approx 1$ billion (1,073,741,824)
Powers of Two (SI Compatible)

- $2^{10} = 1 \text{kibi} \approx 1000$ \hspace{1cm} (1024)
- $2^{20} = 1 \text{mebi} \approx 1 \text{million}$ \hspace{1cm} (1,048,576)
- $2^{30} = 1 \text{gibi} \approx 1 \text{billion}$ \hspace{1cm} (1,073,741,824)
Estimating Powers of Two

- What is the value of $2^{24}$?

- How many values can a 32-bit variable represent?
Estimating Powers of Two

What is the value of $2^{24}$?

$2^4 \times 2^{20} \approx 16$ million

How many values can a 32-bit variable represent?

$2^2 \times 2^{30} \approx 4$ billion
Addition

- **Decimal**
  
  \[
  \begin{array}{c}
  3734 \\
  + 5168 \\
  \hline
  8902 \\
  \end{array}
  \]
  
  \[11 \xleftarrow{\text{carries}}\]

- **Binary**
  
  \[
  \begin{array}{c}
  1011 \\
  + 0011 \\
  \hline
  1110 \\
  \end{array}
  \]
  
  \[11 \xleftarrow{\text{carries}}\]
Add the Following Numbers

\[
\begin{array}{c}
1001 \\
+ 0101 \\
\hline
\end{array}
\quad
\begin{array}{c}
1011 \\
+ 0110 \\
\hline
\end{array}
\]
Add the Following Numbers

\[
\begin{array}{c}
1 \\
1001 \\
+ 0101 \\
\hline
1110
\end{array}
\]

\[
\begin{array}{c}
111 \\
1011 \\
+ 0110 \\
\hline
10001
\end{array}
\]

OVERFLOW!
Overflow

- Digital systems operate on a fixed number of bits
- Addition overflows when the result is too big to fit in the available number of bits
- See previous example of $11 + 6$
Overflow (Is It a Problem?)

- Possible faults
- Security issues

The $7 billion Ariane 5 rocket, launched on June 4, 1996, veered off course 40 seconds after launch, broke up, and exploded. The failure was caused when the computer controlling the rocket overflowed its 16-bit range and crashed.

The code had been extensively tested on the Ariane 4 rocket. However, the Ariane 5 had a faster engine that produced larger values for the control computer, leading to the overflow.
Binary Values and Range

- **N-digit decimal number**
  - How many values? \(10^N\)
  - Range? \([0, 10^N - 1]\)
  - Example: 3-digit decimal number
    - \(10^3 = 1000\) possible values
    - Range: \([0, 999]\)

- **N-bit binary number**
  - How many values? \(2^N\)
  - Range: \([0, 2^N - 1]\)
  - Example: 3-digit binary number
    - \(2^3 = 8\) possible values
    - Range: \([0, 7] = [000_2\text{ to } 111_2]\)
Signed Binary Numbers

- Sign/Magnitude Numbers
- One’s Complement Numbers
- Two’s Complement Numbers
Sign/Magnitude Numbers

- 1 sign bit, N-1 magnitude bits

- Sign bit is the most significant (left-most) bit
  - Positive number: sign bit = 0
  - Negative number: sign bit = 1

- Example, 4-bit sign/mag representations of ±6:
  - +6 =
  - -6 =

- Range of an N-bit sign/magnitude number:
Sign/Magnitude Numbers

- 1 sign bit, N-1 magnitude bits

- Sign bit is the most significant (left-most) bit
  - Positive number: sign bit = 0
  - Negative number: sign bit = 1

- Example, 4-bit sign/mag representations of ±6:
  - +6 = 0110
  - -6 = 1110

- Range of an N-bit sign/magnitude number:
  \([-\left(2^{N-1}-1\right), 2^{N-1}-1]\)
Problems of Sign/Magnitude Numbers

- Addition doesn’t work, for example -6 + 6:

  \[
  \begin{array}{c}
  1110 \\
  + 0110 \\
  \hline
  10100\text{ wrong!}
  \end{array}
  \]

- Two representations of 0 (± 0):

  1000
  0000

- Introduces complexity in the processor design (Was still used by some early IBM computers)
One’s Complement

A negative number is formed by reversing the bits of the positive number (MSB still indicates the sign of the integer):

<table>
<thead>
<tr>
<th>2^7</th>
<th>2^6</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>One’s Complement</th>
<th>Unsigned</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>127</td>
<td>127</td>
</tr>
</tbody>
</table>
One’s Complement

- A negative number is formed by reversing the bits of the positive number (MSB still indicates the sign of the integer):

<table>
<thead>
<tr>
<th></th>
<th>2⁷</th>
<th>2⁶</th>
<th>2⁵</th>
<th>2⁴</th>
<th>2³</th>
<th>2²</th>
<th>2¹</th>
<th>2⁰</th>
<th>One’s Complement</th>
<th>Unsigned</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>127</td>
<td>127</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-127</td>
<td>128</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-126</td>
<td>129</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-2</td>
<td>253</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>254</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-0</td>
<td>255</td>
</tr>
</tbody>
</table>
One’s Complement

The range of n-bit one’s complement numbers is:

\[-2^{n-1} - 1, 2^{n-1} - 1\]

8 bits: [-127, 127]

Addition:

Addition of signed numbers in one's complement is performed using binary addition with end-around carry. If there is a carry out of the most significant bit of the sum, this bit must be added to the least significant bit of the sum:

Example: 17 + (-8) in 8-bit one’s complement

\[
\begin{array}{c}
0001 \ 0001 \\ (+)
\end{array}
\begin{array}{c}
1111 \ 0111 \\ (-8)
\end{array}
\hline
1 \ 0000 \ 1000
\hline
1
\hline
0000 \ 1001 = \ (9)
\]
Two’s Complement Numbers

- Don’t have same problems as sign/magnitude numbers:
  - Addition works
  - Single representation for 0

- Has advantages over one’s complement:
  - Has a single zero representation
  - Eliminates the end-around carry operation required in one's complement addition
Two’s Complement Numbers

- A negative number is formed by reversing the bits of the positive number (MSB still indicates the sign of the integer) and adding 1:

<table>
<thead>
<tr>
<th>$2^7$</th>
<th>$2^6$</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>Two’s Complement</th>
<th>Unsigned</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>127</td>
<td>127</td>
</tr>
</tbody>
</table>
Two’s Complement Numbers

A negative number is formed by reversing the bits of the positive number (MSB still indicates the sign of the integer) and adding 1:

<table>
<thead>
<tr>
<th>$2^7$</th>
<th>$2^6$</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>Two’s Complement</th>
<th>Unsigned</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>= 0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>= 1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>= 2</td>
<td>2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>= 127</td>
<td>127</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>= -128</td>
<td>128</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>= -127</td>
<td>129</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>= -3</td>
<td>253</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>= -2</td>
<td>254</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>= -1</td>
<td>255</td>
</tr>
</tbody>
</table>
Two’s Complement Numbers

- Same as unsigned binary, but the most significant bit (msb) has value of \(-2^{N-1}\)
  \[
  I = \sum_{i=0}^{i=n-2} b_i2^i - b_{n-1}2^{n-1}
  \]
  - Most positive 4-bit number:
  - Most negative 4-bit number:

- The most significant bit still indicates the sign
  (1 = negative, 0 = positive)

- Range of an N-bit two’s comp number:
Two’s Complement Numbers

- Same as unsigned binary, but the most significant bit (msb) has value of $-2^{N-1}$
  \[ I = \sum_{i=0}^{i=n-2} b_i 2^i - b_{n-1} 2^{n-1} \]
  - Most positive 4-bit number: 0111
  - Most negative 4-bit number: 1000

- The most significant bit still indicates the sign
  (1 = negative, 0 = positive)

- Range of an N-bit two’s comp number:
  \[ [-2^{N-1}, 2^{N-1}-1] \]
  8 bits: [-128,127]
“Taking the Two’s Complement”

- How to flip the sign of a two’s complement number:
  - Invert the bits
  - Add one

- Example: Flip the sign of $3_{10} = 0011_2$
“Taking the Two’s Complement”

- How to flip the sign of a two’s complement number:
  - Invert the bits
  - Add one

- Example: Flip the sign of $3_{10} = 0011_2$
  - Invert the bits $1100_2$
“Taking the Two’s Complement”

- How to flip the sign of a two’s complement number:
  - Invert the bits
  - Add one

- Example: Flip the sign of $3_{10} = 0011_2$
  - Invert the bits $1100_2$
  - Add one $1101_2$
“Taking the Two’s Complement”

- How to flip the sign of a two’s complement number:
  - Invert the bits
  - Add one

- Example: Flip the sign of $3_{10} = 0011_2$
  - Invert the bits
  - Add one

- Example: Flip the sign of $-8_{10} = 11000_2$
“Taking the Two’s Complement”

- How to flip the sign of a two’s complement number:
  - Invert the bits
  - Add one

- Example: Flip the sign of \(3_{10}\) = \(0011_2\)
  - Invert the bits \(1100_2\)
  - Add one \(1101_2\)

- Example: Flip the sign of \(-8_{10}\) = \(11000_2\)
  - Invert the bits \(00111_2\)
  - Add one \(01000_2\)
Two’s Complement Addition

- Add 6 + (-6) using two’s complement numbers

\[
\begin{align*}
0110 \\
+ & \quad 1010 \\
\hline
1010
\end{align*}
\]

- Add -2 + 3 using two’s complement numbers

\[
\begin{align*}
1110 \\
+ & \quad 0011 \\
\hline
1000
\end{align*}
\]
Two’s Complement Addition

- Add $6 + (-6)$ using two’s complement numbers
  \[
  \begin{array}{c}
  111 \\
  0110 \\
  + 1010 \\
  \hline
  10000
  \end{array}
  \]

- Add $-2 + 3$ using two’s complement numbers
  \[
  \begin{array}{c}
  111 \\
  1110 \\
  + 0011 \\
  \hline
  10001
  \end{array}
  \]

- Correct results if overflow bit is ignored
Increasing Bit Width

- A value can be extended from N bits to M bits (where M > N) by using:
  - Sign-extension
  - Zero-extension
Sign-Extension

- Sign bit is copied into most significant bits
- Number value remains the same
- Give correct result for two’s complement numbers

**Example 1:**
- 4-bit representation of 3 = 0011
- 8-bit sign-extended value: 00000011

**Example 2:**
- 4-bit representation of -5 = 1011
- 8-bit sign-extended value: 11111011
Zero-Extension

- Zeros are copied into most significant bits
- Value will change for negative numbers

**Example 1:**
- 4-bit value = $0011_2 = 3_{10}$
- 8-bit zero-extended value: $00000011_2 = 3_{10}$

**Example 2:**
- 4-bit value = $1011_2 = -5_{10}$
- 8-bit zero-extended value: $00001011_2 = 11_{10}$
## Number System Comparison

<table>
<thead>
<tr>
<th>Number System</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unsigned</td>
<td>([0, 2^{N-1}])</td>
</tr>
<tr>
<td>Sign/Magnitude</td>
<td>([-2^{N-1} - 1, 2^{N-1} - 1])</td>
</tr>
<tr>
<td>Two’s Complement</td>
<td>([-2^{N-1}, 2^{N-1} - 1])</td>
</tr>
</tbody>
</table>

### For example, 4-bit representation:

- **Unsigned:**
  - 0000: 0
  - 0001: 1
  - 0010: 2
  - 0011: 3
  - 0100: 4
  - 0101: 5
  - 0110: 6
  - 0111: 7
  - 1000: -8
  - 1001: -7
  - 1010: -6
  - 1011: -5
  - 1100: -4
  - 1101: -3
  - 1110: -2
  - 1111: -1

- **Two’s Complement:**
  - 0000: 0
  - 0001: 1
  - 0010: 2
  - 0011: 3
  - 0100: 4
  - 0101: 5
  - 0110: 6
  - 0111: 7
  - 1000: -8
  - 1001: -7
  - 1010: -6
  - 1011: -5
  - 1100: -4
  - 1101: -3
  - 1110: -2
  - 1111: -1

- **Sign/Magnitude:**
  - 0000: 0
  - 0001: 1
  - 0010: 2
  - 0011: 3
  - 0100: 4
  - 0101: 5
  - 0110: 6
  - 0111: 7
  - 1000: -8
  - 1001: -7
  - 1010: -6
  - 1011: -5
  - 1100: -4
  - 1101: -3
  - 1110: -2
  - 1111: -1
Lessons Learned

- How to express decimal numbers using only 1s and 0s
- How to simplify writing binary numbers in hexadecimal
- Adding binary numbers
- Methods to express negative numbers
  - Sign Magnitude
  - One’s complement
  - Two’s complement (the one commonly used)
Required Reading
for Week 1
25-26 February 2021
Spring 2021