Operations in Number System & Boolean Algebra
Outline

• Number System & Conversion
  – Decimal, Binary, Octal, Hex
• Other representation: Signed, Complement

• Operations in Number System
  – Add, Sub, Mul, Div, Mod

• How to handle real number efficiently
  – Float, Double

• Boolean Algebra: Gates & Theorem
Operation on Numbers

• Addition
• Subtraction
• Multiplication
• Division
• Modulus
Binary Addition

• One bit

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>+</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>+</td>
<td>1</td>
</tr>
</tbody>
</table>

= 0 1 (Carry bit)

• Multibit (consider carry)

\[
\begin{array}{cccccc}
1 & 1 & 0 & 1 \\
+ & 1 & 0 & 0 & 1 \\
\hline \\
1 & 0 & 1 & 1 & -
\end{array}
\]
Binary Subtraction

• One bit
  
  \[
  \begin{array}{ccc}
  0 & - & 0 = 0 \\
  1 & - & 0 = 1 \\
  0 & - & 1 = 1 \\
  1 & - & 1 = 0 \\
  \end{array}
  \]

  1 (Carry bit)

• Multibit (consider carry)
  
  \[
  \begin{array}{cccccc}
  1 & 1 & 1 & 1 & 0 \\
  - & 1 & 0 & 0 & 1 \\
  \hline
  0 & 1 & 0 & 0 & 1 \\
  \end{array}
  \]
Binary Subtraction

- Multibit (consider carry)

\[
\begin{array}{cccc}
1 & 1 & 1 & 0 \\
- & 1 & 0 & 0 & 1 \\
\hline
& 0 & 1 & 0 & 1 \\
\end{array}
\]

- add 2’s complement = \((1001)' + 1 = 0110 + 1 = 0111\)

- Other way (add 2’s complement & discard carry)

\[
\begin{array}{cccc}
1 & 1 & 1 & 0 \\
+ & 0 & 1 & 1 & 1 \\
\hline
& 1 & 0 & 1 & 0 & 1 \\
\end{array}
\]
Binary Multiplication

- Repeated addition
- Many improved technique

\[
\begin{array}{cccccc}
1 & 0 & 0 & 0 & 0 & = 8_{10} \\
\times & 0 & 1 & 1 & 1 & 0 & = 6_{10} \\
\hline
0 & 0 & 0 & 0 & 0 & \\
+ & 1 & 0 & 0 & 0 & 0 \\
+ & 1 & 0 & 0 & 0 & 0 \\
+ & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 1 & 1 & 0 & 0 & 0 & 0 & = 48_{10} \\
\end{array}
\]
Binary Division & Modulus

• $\overline{0110010 \div 011} = \overline{50_{10} \div 3_{10}}$

\[
\begin{array}{c}
011 ) 0110010 \\
\hline
011 \\
\hline
000 \\
\hline
000 \\
\hline
000 \\
\hline
001 \\
\hline
010 \\
\hline
000 \\
\end{array}
\]

Q: $\overline{1000} = \overline{16_{10}}$
R: $\overline{10} = \overline{2_{10}}$
Hex Addition

- **Addition**
  
  \[
  \begin{array}{cccc}
  1 & A & 2 & B \\
  + & 7 & C & A & 6 \\
  \hline
  & 9 & 6 & D & 1 \\
  \end{array}
  \]
  
  Carry Value to higher significant one is 1

- **Subtraction**

  \[
  \begin{array}{cccc}
  A & 2 & B & 9 \\
  1 & C & F & 3 \\
  \hline
  8 & 5 & C & 6 \\
  \end{array}
  \]
Other Operations to skip

- Hex: Multiplication, Division & Mod
- Oct: Add, Sub, Multiplication, Division & Mod
- Read yourself
Floating Point Numbers

- Fractional Number: 2034.455
- Floating format: $2.03455 \times 10^3$
- In Binary: $1.100011 \times 2^7$
- Real numbers must be normalized using scientific notation:
  $0.1... \times 2^n$ where $n$ is an integer
- Note that the whole number part is always 0 and the most significant digit of the fraction is a 1 – ALWAYS!
Need to go beyond integers

- integer 7
- rational 5/8
- real √3
- complex 2 - 3i

Extremely large and small values:
- distance pluto - sun = 5.9 \times 10^{12} \text{ m}
- mass of electron = 9.1 \times 10^{-28} \text{ gm}
Representing fractions

- Integer pairs (for rational numbers)
  
  \[ \frac{5}{8} = \frac{5}{8} \]

- Strings with explicit decimal point
  
  -247.09

- Implicit point at a fixed position
  
  010011010110001011

- Floating point
  
  \[ \text{fraction} \times \text{base}^{\text{power}} \]
Numbers with binary point

101.11 = 1x2^2 + 0x2^1 + 1x2^0 + . + 1x2^{-1} + 1x2^{-2}
= 4 + 1 + . + 0.5 + 0.25 = 5.75_{10}

0.6 = 0.1001100110011001......

.6 x 2 = 1 + .2
.2 x 2 = 0 + .4
.4 x 2 = 0 + .8
.8 x 2 = 1 + .6
IEEE 754 standard

- **Single precision numbers**
  
  ![Single precision binary representation](image)

- **Double precision numbers**
  
  ![Double precision binary representation](image)
**Float operations**

- **Add, Sub**
  - Make same power, operate, normalize
    
    \[
    2 \times 10^6 + 3 \times 10^4 = 2 \times 10^6 + 0.03 \times 10^6 = 2.03 \times 10^6
    \]
    
    \[
    2 \times 10^6 - 3 \times 10^4 = 2 \times 10^6 - 0.03 \times 10^6 = 1.97 \times 10^6
    \]

- **Mul, Div**
  - Do operation, normalize
    
    \[
    2.0 \times 10^6 \times 3.0 \times 10^3 = 2 \times 3 \times 10^{(6+3)} = 6.0 \times 10^9
    \]
    
    \[
    2 \times 10^6 \div 3 \times 10^3 = \frac{2}{3} \times 10^{(6-3)} = 0.666 \times 10^3 = 6.66 \times 10^2
    \]
Boolean Algebra

• Computer hardware using binary circuit greatly simply design

• Binary circuits: To have a conceptual framework to manipulate the circuits algebraically

• George Boole (1813-1864): developed a mathematical structure
  — To deal with binary operations with just two values.
Basic Gates in Binary Circuit

- Element 0: “FALSE”. Element 1: “TRUE”.
- ‘+’ operation “OR”, ‘*’ operation “AND” and ‘’ operation “NOT”.

<table>
<thead>
<tr>
<th>OR</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AND</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NOT</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
OR Gate

• ‘+’ operation “OR”

\[ R = X + Y \]

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Y</td>
<td>R=X OR Y</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>----------</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ 1 + Y = 1 \]
**AND Gate**

- ‘*’ operation “AND”

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>R = X AND Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**AND**

<table>
<thead>
<tr>
<th>AND</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

0 * Y = 0
NOT Gate

- ‘operation “NOT”

\[
R = X' \\
X \quad R = \text{NOT } X
\]

<table>
<thead>
<tr>
<th>X</th>
<th>R=X’</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Boolean Algebra Defined

- Boolean Algebra $\mathbb{B}$: 5-tuple
  $$\{ \mathbb{B}, +, *, ', 0, 1 \}$$
- $+$ and $*$ are binary operators,
- $'$ is a unary operator.
Boolean Algebra Defined

- **Axiom #1: Closure**
  If a and b are Boolean
  \[(a + b)\] and \[(a * b)\] are Boolean.

- **Axiom #2: Cardinality**
  if a is Boolean then \(a'\) is Boolean

- **Axiom #3: Commutative**
  \[(a + b) = (b + a)\]
  \[(a * b) = (b * a)\]
Boolean Algebra Defined

• *Axiom #4: Associative*: If a and b are Boolean

\[(a + b) + c = a + (b + c)\]
\[(a \times b) \times c = a \times (b \times c)\]

• *Axiom #6: Distributive*

\[a \times (b + c) = (a \times b) + (a \times c)\]
\[a + (b \times c) = (a + b) \times (a + c)\]

2\text{nd} one is Not True for Decimal numbers System
Boolean Algebra Defined

• **Axiom #5: Identity Element**: 
  - B has identity to + and *
    - 0 is identity element for +: \(a + 0 = a\)
    - 1 is identity element for *: \(a * 1 = a\)

• **Axiom #7: Complement Element**
  - \(a + a' = 1\)
  - \(a * a' = 0\)
Thanks