• Review: Algorithms
• Using building blocks
• Binary system
• Unsigned integers
  • Unsigned addition
• Negative numbers
  • Two's complement representation
• Conversion between binary and decimal
• More on addition
  • Subtraction
  • Sign extension
  • Overflow
• Review of Hexadecimal representation
Example of Algorithm

Problem Statement:
Find the factorial of a given integer, n
\[ n! = 1 \times 2 \times 3 \times \ldots \times n-1 \times n \]
\[ = n \times (n-1) \times (n-2) \times \ldots \times 2 \times 1 \]

Algorithm:

```
IF n = 0, F = 1
STOP

OTHERWISE
F = n
N = N - 1
IF N = 0, STOP
OTHERWISE F = F \times N
```

Example:

\[ n = 4 \]
\[ F = 4 \]
\[ n = 3 \]
\[ F = 12 \]
\[ n = 2 \]
\[ F = 24 \]
\[ n = 1 \]
\[ F = 24 \]
\[ n = 0 \]
STOP
Computer to execute algorithm

- Has memory to store n, factorial
- Units which can subtract, multiply
- Unit to test whether a number is 0 or negative

Has Instructions to:
- Get data from memory
- Put data into memory
- Add, subtract, ...
- Test whether a number is zero
- Change flow of execution
- Print answer
Using “black-box” functions

Have “boxes” for + and x

Implement AxB + BxC

\[(A+C) \times B\]
NUMBERS

UNARY: 5 = 11111

DECIMAL: 29
  \[ \frac{29}{16} = 1 \text{ remainder } 13 \]

ROMAN: XXIX
  \[ 16 + 8 + 1 + 1 \]

BINARY: 11101
  \[ \frac{11101}{2} = 1 \text{ remainder } 1 \]
  \[ \frac{1}{2} = 0 \text{ remainder } 1 \]
  \[ \frac{0}{2} = 0 \text{ remainder } 0 \]
  \[ \frac{0}{2} = 0 \text{ remainder } 0 \]
  \[ 2^4 \uparrow 2^3 \uparrow 2^2 \uparrow 2^1 \uparrow 2^0 \]
Representing Data Inside a Computer: Start with Unsigned Integers

CLASS HAS 108 PEOPLE

HOW MANY BITS TO GIVE EACH PERSON A UNIQUE ID??

n = \log_2(108) + 1

8.73...
 Unsigned Integers (cont.)

An $n$-bit unsigned integer represents $2^n$ values: from 0 to $2^n - 1$.

<table>
<thead>
<tr>
<th>$2^2$</th>
<th>$2^1$</th>
<th>$2^0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Unsigned Binary Arithmetic

Base-2 addition – just like base-10!

- add from right to left, propagating carry

\[
\begin{align*}
1 & \quad 0 \quad 0 \quad 1 \quad 0 \quad (18) \\
\text{+} & \quad 1 \quad 0 \quad 1 \quad 1 \quad (11) \\
\hline
1 \quad 1 \quad 1 \quad 0 \quad 1 \quad (29) \\
1 \quad 1 \quad 1 \quad 1 \\
\text{+} & \quad 1 \\
\hline
1 \quad 0 \quad 0 \quad 0 \quad 0 \\
\end{align*}
\]
Positive and Negative Numbers
Signed Integers - Represent Negative Numbers

**Sign-Magnitude**
- **0**: Positive
- **1**: Negative

+5: 00101
-5: 10101

Problems: Two "0"s.
(+5) + (-5) ≠ 0

**1's Complement**
- Negative: Complement of the number

+5: 0.0101
-5: 1.1010

Two "0"s.

**2's Complement**

+5: 0.0101
-5: 1.1011

Is complement +1

-(-5): 00100
  +1
  00101
Two’s Complement Signed Integers

Most Significant bit is sign bit – it has weight $-2^{n-1}$.

Range of an $n$-bit number: $-2^{n-1}$ through $2^{n-1} - 1$.

- The most negative number ($-2^{n-1}$) has no positive counterpart.

<table>
<thead>
<tr>
<th>$-2^3$</th>
<th>$2^2$</th>
<th>$2^1$</th>
<th>$2^0$</th>
<th>$-2^3$</th>
<th>$2^2$</th>
<th>$2^1$</th>
<th>$2^0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 1</td>
<td>1 0 0 0</td>
<td>0 0 0 0</td>
<td>1 0 0 0</td>
<td>0 0 0 1</td>
<td>1 0 0 0</td>
<td>0 0 0 1</td>
<td></td>
</tr>
<tr>
<td>0 0 1 0</td>
<td>2 0 0 0</td>
<td>1 0 0 0</td>
<td>1 0 0 0</td>
<td>0 0 1 0</td>
<td>2 0 0 0</td>
<td>1 0 0 0</td>
<td></td>
</tr>
<tr>
<td>0 0 1 1</td>
<td>3 0 0 0</td>
<td>1 0 0 0</td>
<td>1 0 0 0</td>
<td>0 0 1 1</td>
<td>3 0 0 0</td>
<td>1 0 0 0</td>
<td></td>
</tr>
<tr>
<td>0 1 0 0</td>
<td>4 0 0 0</td>
<td>1 0 0 0</td>
<td>1 0 0 0</td>
<td>0 1 0 0</td>
<td>4 0 0 0</td>
<td>1 0 0 0</td>
<td></td>
</tr>
<tr>
<td>0 1 0 1</td>
<td>5 0 0 0</td>
<td>1 0 0 0</td>
<td>1 0 0 0</td>
<td>0 1 0 1</td>
<td>5 0 0 0</td>
<td>1 0 0 0</td>
<td></td>
</tr>
<tr>
<td>0 1 1 0</td>
<td>6 0 0 0</td>
<td>1 0 0 0</td>
<td>1 0 0 0</td>
<td>0 1 1 0</td>
<td>6 0 0 0</td>
<td>1 0 0 0</td>
<td></td>
</tr>
<tr>
<td>0 1 1 1</td>
<td>7 0 0 0</td>
<td>1 0 0 0</td>
<td>1 0 0 0</td>
<td>0 1 1 1</td>
<td>7 0 0 0</td>
<td>1 0 0 0</td>
<td></td>
</tr>
</tbody>
</table>

$7 - 3 = 4$

$7 + (-3) = 4$
ADDI NG A NUMBER 
TO ITSELF

3 + 3
0 0 1 1
+ 0 0 1 1
0 1 1 0
= (0011) SHIF TED
LEFT

≡ × 2
SIGN EXTENSION

4-BIT NUMBER: +4

PUT INTO AN 8-BIT MEMORY

+4 = 0100

↓

0000 0100.

HOW ABOUT -4 ??

SIGN

1 1 0 0 = -4

1 1 1 1 1 1 0 0

EXTEND THE SIGN