Endianness
Lab Schedule

Activities

- **This Week**
  - Python introduction
  - Networking introduction
  - **Lab 8** (HTTP, TCP sockets)

Assignments Due

- **Lab 7**
  - Due by Mar 19th 5:00am

- **Lab 8**
  - Due by Mar 26th 5:00am
In typical computer memory, each address (location) stores one byte.

If we have a one-byte integer, how is that stored in memory?

If we have a two-byte integer, how is that stored in memory?

If we have a four-byte integer, how is that stored in memory?

Endianness = Byte Ordering
Endianness Example

- 32-bit hexadecimal number 0x12345678
- Composed of 4 bytes: 0x12 0x34 0x56 0x78
  - (MSB)
  - (LSB)
- Two possible arrangements:

<table>
<thead>
<tr>
<th>Address</th>
<th>“Option A”</th>
<th>“Option B”</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0x12</td>
<td>0x78</td>
</tr>
<tr>
<td>1</td>
<td>0x34</td>
<td>0x56</td>
</tr>
<tr>
<td>2</td>
<td>0x56</td>
<td>0x34</td>
</tr>
<tr>
<td>3</td>
<td>0x78</td>
<td>0x12</td>
</tr>
</tbody>
</table>
Endianness Example

- 32-bit hexadecimal number
  0x12345678

- Composed of 4 bytes:
  0x12 0x34 0x56 0x78
  (MSB)  (LSB)

- Two possible arrangements:
  - Big Endian
  - Little Endian

<table>
<thead>
<tr>
<th>Address</th>
<th>Big Endian</th>
<th>Little Endian</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0x12 (MSB)</td>
<td>0x78 (LSB)</td>
</tr>
<tr>
<td>1</td>
<td>0x34</td>
<td>0x56</td>
</tr>
<tr>
<td>2</td>
<td>0x56</td>
<td>0x34</td>
</tr>
<tr>
<td>3</td>
<td>0x78</td>
<td>0x12</td>
</tr>
</tbody>
</table>
How is DEADBEEF<sub>16</sub> stored in little and big endian formats at address 21C<sub>16</sub>?

**Little endian**
- 21C<sub>16</sub>=EF<sub>16</sub>
- 21D<sub>16</sub>=BE<sub>16</sub>
- 21E<sub>16</sub>=AD<sub>16</sub>
- 21F<sub>16</sub>=DE<sub>16</sub>

**Big endian**
- 21C<sub>16</sub>=DE<sub>16</sub>
- 21D<sub>16</sub>=AD<sub>16</sub>
- 21E<sub>16</sub>=BE<sub>16</sub>
- 21F<sub>16</sub>=EF<sub>16</sub>
Big Endian –vs– Little Endian

**Big-Endian CPU**
- Most significant byte (MSB) comes first (stored in lower memory address)

**Examples**
- Motorola 68000
- Java virtual machine
- IBM PowerPC (by default, can also be little endian)

**Little-Endian CPU**
- Least significant byte (LSB) comes first (stored in lower memory addresses)

**Examples**
- Intel x86/x86-64
- DEC Alpha
- ARM (by default, also can be big endian)
Etymology of “Endiann”

- Origin in 1980s
- Reference to Swift's *Gulliver's Travels*, in which the Lilliputians were divided into two camps:
  - Those who ate their eggs by opening the ‘big’ end
  - Those who ate them by opening the ‘little’ end
- In other words, a trivial distinction
Do I Care?

- When do I need to care that some computers are big-endian and others are little endian?
  - What happens if I open big-endian data on a little-endian computer?

- Endianness must be considered whenever you are sharing data between different computer systems
  - Reading/writing data files to disk
  - Reading/writing data files to network
Best Practices

- Pick one format and stick with it!
  - Example: Data sent over the network will always be in \textit{big-endian} format regardless of who sends it
    - \textit{Networks are big-endian “by tradition”}
  - Example: Data written to disk will always be in \textit{little-endian} format regardless of who writes it

- Convert between data storage/transfer format and internal representation as needed
  - Example: Little-endian machines convert to big-endian before sending data onto the network (and convert back upon receiving data from the network)
### Examples in Industry

<table>
<thead>
<tr>
<th>Little-Endian Format</th>
<th>Big-Endian Format</th>
<th>Variable or Bi-Endian Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMP</td>
<td>PSD</td>
<td>DXF (AutoCAD*)</td>
</tr>
<tr>
<td>GIF</td>
<td>IMG</td>
<td>PS (Postscript*, 8 bit interpreted text, no Endian issue)</td>
</tr>
<tr>
<td>FLI</td>
<td>JPEG, JPG</td>
<td>SGI (Silicon Graphics*)</td>
</tr>
<tr>
<td>PCX</td>
<td>MacPaint</td>
<td>Sun Raster</td>
</tr>
<tr>
<td>QTM</td>
<td>SGI</td>
<td>Sun Raster</td>
</tr>
<tr>
<td>RTF</td>
<td>WPG</td>
<td>Sun Raster</td>
</tr>
</tbody>
</table>

#### Bus Protocols

<table>
<thead>
<tr>
<th>Infiniband</th>
<th>PCI Express</th>
<th>PCI-32/PCI-64</th>
<th>USB</th>
</tr>
</thead>
</table>

#### Network Protocols

| TCP/IP | UDP | GMII (8 bit wide bus, no Endian issue) |

#### Table 2- Common file formats