Chapter 8: Bits and the "Why" of Bytes: Representing Information Digitally

Fluency with Information Technology
Third Edition
by
Lawrence Snyder

Digitizing Discrete Information

- **Digitize**: Represent information with digits (normally base 10 numerals 0 through 9)
- **Limitation of Digits**
  - Alternative Representation: Any set of symbols could represent phone number digits, as long as the keypad is labeled accordingly
- **Symbols, Briefly**
  - Digits have the advantage of having short names (easy to say)
  - But computer professionals are shortening symbol names (exclamation point is pronounced "bang")

Ordering Symbols

- Advantage of digits for encoding info is that items can be listed in numerical order
- To use other symbols, we need an ordering system (**collating sequence**)
  - Agreed order from smallest to largest value
- In choosing symbols for encoding, consider how symbols interact with things being encoded

The Fundamental Representation of Information

- The fundamental patterns used in IT come when the physical world meets the logical world
- The most fundamental form of information is the presence or absence of a physical phenomenon
- In the logical world, the concepts of true and false are important
  - By associating true with the presence of a phenomenon and false with its absence, we use the physical world to implement the logical world, and produce information technology

Analog vs. Digital

- **Analog is continuous data/information**
  - Sound waves

Figure 8.1. Three symbol assignments for a telephone keypad.
Analog vs. Digital

- Digital is discrete data/information
  - Many distinct samples of data
  - Stored in binary (0's and 1's)
    - All data in a computer is represented in binary

---

The PandA Representation

- PandA is the mnemonic for "presence and absence"
- It is discrete (distinct or separable)—the phenomenon is present or it is not (true or false; 1 or 0). There is no continuous gradation in between.

---

A Binary System

- Two patterns make a binary system
  - Base 2 (0 or 1)
- The basic binary unit is known as a "bit" (short for binary digit)
- 8 bits are grouped together to form a byte
  - Memory accessed by byte addresses
- We can give any names to these two patterns as long as we are consistent
  - PandA (Presence and Absence can represent 1 and 0, respectively)

---

Encoding Bits on a CD-ROM

- Bump, when viewed from top
- Pit, when viewed from bottom

---

Table 8.1 Possible interpretations of the two PandA patterns

<table>
<thead>
<tr>
<th>Present</th>
<th>Absent</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Black</td>
<td>White</td>
</tr>
<tr>
<td>For</td>
<td>Against</td>
</tr>
<tr>
<td>Yang</td>
<td>Yin</td>
</tr>
<tr>
<td>Lisa</td>
<td>Bart</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Bits in Computer Memory

- Memory is arranged inside a computer in a very long sequence of bits (places where a phenomenon can be set and detected)
- Analogy: Sidewalk Memory
  - Each sidewalk square represents a memory slot (bit), and stones represent the presence or absence
  - If a stone is on the square, the value is 1, if not the value is 0

Alternative PandA Encodings

- There are other ways to encode two states using physical phenomena
  - Use stones on all squares, but black stones for one state and white for the other
  - Use multiple stones of two colors per square, saying more black than white means 0 and more white than black means 1
  - Stone in center for one state, off-center for the other
  - etc.

Combining Bit Patterns

- Since we only have two patterns, we must combine them into sequences to create enough symbols to encode necessary information
- Binary (PandA) has 2 patterns, arranging them into n-length sequences, we can create $2^n$ symbols

Hex Explained

- Recall in Chapter 4, we specified custom colors in HTML using hex digits
  - e.g., `<font color=#FF8E2A>`
  - Hex is short for hexadecimal, base 16
- Why use hex? Writing the sequence of bits is long, tedious, and error-prone

Table 8.2 Number of symbols when the number of possible patterns is two

<table>
<thead>
<tr>
<th>n</th>
<th>$2^n$</th>
<th>Symbols</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>128</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>256</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>512</td>
</tr>
<tr>
<td>10</td>
<td>1,024</td>
<td>1,024</td>
</tr>
</tbody>
</table>
The 16 Hex Digits

- 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
  - A = 10, B = 11, ..., F = 15
- Sixteen values can be represented perfectly by 4-bit sequences ($2^4 = 16$)
- Changing hex digits to bits and back again:
  - Given a sequence of bits, group them in 4's and write the corresponding hex digit
    - 0101 1100
      - 5 C
  - Given hex, write the associated group of 4 bits

Hex (0-9,A-F)

<table>
<thead>
<tr>
<th>Decimal</th>
<th>Hex</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0000</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0001</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0010</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>0011</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>0100</td>
<td>100</td>
</tr>
<tr>
<td>5</td>
<td>0101</td>
<td>101</td>
</tr>
<tr>
<td>6</td>
<td>0110</td>
<td>110</td>
</tr>
<tr>
<td>7</td>
<td>0111</td>
<td>111</td>
</tr>
<tr>
<td>8</td>
<td>1000</td>
<td>100</td>
</tr>
<tr>
<td>9</td>
<td>1001</td>
<td>101</td>
</tr>
<tr>
<td>A</td>
<td>1010</td>
<td>1010</td>
</tr>
<tr>
<td>B</td>
<td>1011</td>
<td>1011</td>
</tr>
<tr>
<td>C</td>
<td>1100</td>
<td>1100</td>
</tr>
<tr>
<td>D</td>
<td>1101</td>
<td>1101</td>
</tr>
<tr>
<td>E</td>
<td>1110</td>
<td>1110</td>
</tr>
<tr>
<td>F</td>
<td>1111</td>
<td>1111</td>
</tr>
</tbody>
</table>

Digitizing Text

- Early binary representation—one and 0—encoded numbers and keyboard characters
- Now representation for sound, video, and other types of information are also important
- For encoding text, what symbols should be included?
  - We want to keep the list small enough to use fewer bits, but we don't want to leave out critical characters

Assigning Symbols

- 26 uppercase and 26 lowercase Roman letters, 10 Arabic numerals, 10 arithmetic characters, 20 punctuation characters (including space), and 3 non-printable characters (new line, tab, backspace) = 95 characters, enough to represent English
- For 95 symbols, we need 7-bit sequences
  - $2^6 = 64$
  - $2^7 = 128$
- A standard 7-bit code is ASCII (American Standard Code for Information Interchange)
Extended ASCII: An 8-bit Code

• By the mid-1960’s, it became clear that 7-bit ASCII was not enough to represent text from languages other than English
• IBM extended ASCII to 8 bits (256 symbols)
• Called “Extended ASCII,” the first half is original ASCII with a 0 added at the beginning of each group of bits
• Handles most Western languages and additional useful symbols

ASCII Coding of Phone Numbers

• How would a computer represent in its memory, the phone number 888 555 1212?
• Encode each digit with its ASCII byte

Unicode

• Several languages around the world have more than 256 individual characters
• Unicode uses 16 bits; \(2^{16} = 65536\) characters
  – 1st 7 bits (128 chars) are ASCII chars
  – Different locales – different characters beyond 1st 7 bits

NATO Broadcast Alphabet

• The code for broadcast communication is purposefully inefficient, to be distinctive when spoken amid noise

<table>
<thead>
<tr>
<th>Table 8.4 NATO broadcast alphabet designed not to be minimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>D</td>
</tr>
<tr>
<td>E</td>
</tr>
<tr>
<td>F</td>
</tr>
<tr>
<td>G</td>
</tr>
</tbody>
</table>

The Oxford English Dictionary

• Extended ASCII encodes letters and characters well, but most documents contain more than just text.
  – Format information like font, font size, justification
• Formatting characters could be added to ASCII, but that mixes the content with the description of its form (**metadata**)
• Metadata is represented using tags, as in HTML
Using Tags to Encode

- Oxford English Dictionary (OED) printed version is 20 volumes
- We could type the entire contents as ASCII characters (in about 120 years), but searching would be difficult
  - Suppose you search for the word "set." It is included in many other words like closet, horsetail, settle, etc.
  - How will the software know what characters comprise the definition of set?
- Incorporate metadata

Structure Tags

- Special set of tags was developed to specify OED's structure
  - <hw> means headword, the word being defined
  - Other tags label pronunciation <pr>, phonetic notation <ph>, parts of speech <ps>
- The tags do not print. They are there only to specify structure so the computer knows what part of the dictionary it is looking at

Why "BYTE"

- Why is BYTE spelled with a Y?
- The Engineers at IBM were looking for a word for a quantity of memory between a bit and a word (usually 32 bits). Bite seemed appropriate, but they changed the i to a y, to minimize typing errors.