CSARCH1 Introduction to Computer Organization and Architecture

Binary Data Organization

Units of Digital Information

Unit	Description
Bit	**2 cells** (0 or 1). Abbreviated as lowercase b.
Crumb	A group of **2 bits**.
Nibble	A group of **4 bits**.
Byte	A group of **8 bits**. Abbreviated as uppercase B.
Word	A group of bits **defined by the computer architecture** (e.g., 16-bit, 32-bit, 64-bit). Words may also have extensions: • **Double Word**: twice the size of a word. • **Quad Word**: Four times the size of a word.

Least Significant vs. Most Significant

- **Least Significant:**
  - Refers to the rightmost position in a binary data organization.
  - Represents the **smallest weight or value**.
- **Most Significant:**
  - Refers to the leftmost position in a binary data organization.
  - Represents the **largest weight or value**.

```
Example 1: 0x42AB3156CAFE5687 where a word is defined as 16-bit.

(the prefix 0x denotes hexadecimal / base-16)

(remember that each digit in base-16 represents 4 bits)

LSNibble     -> 4 2 A B 3 1 5 6 C A F E 5 6 8 7 = 7   (last 4 bits)

MSNibble     -> 4 2 A B 3 1 5 6 C A F E 5 6 8 7 = 4   (first 4 bits)

LSByte       -> 4 2 A B 3 1 5 6 C A F E 5 6 8 7 = 78   (last 8 bits)

MSByte       -> 4 2 A B 3 1 5 6 C A F E 5 6 8 7 = 12   (first 8 bits)

LSWord       -> 4 2 A B 3 1 5 6 C A F E 5 6 8 7 = 5678   (last 16 bits)

MSWord       -> 4 2 A B 3 1 5 6 C A F E 5 6 8 7 = 12AB   (first 16 bits)
```

```
LSDoubleWord -> 4 2 A B 3 1 5 6 C A F E 5 6 8 7 = CAFE5678   (last 32 bits)

MSDoubleWord -> 4 2 A B 3 1 5 6 C A F E 5 6 8 7 = 12AB3456   (first 32 bits)

(turn the hexadecimal into binary to "see" the bit or crumb)

LSBit        -> 4 2 A B 3 1 5 6 C A F E 5 6 8 7       (expand 7)
             -> 7 -> (base-2) -> 0 1 1 1        = 1   (last 1 bit)

MSBit        -> 4 2 A B 3 1 5 6 C A F E 5 6 8 7       (expand 4)
             -> 4 -> (base-2) -> 0 1 0 0        = 0   (first 1 bit)

LSCrumb      -> 4 2 A B 3 1 5 6 C A F E 5 6 8 7        (expand 7)
             -> 7 -> (base-2) -> 0 1 1 1        = 11  (last 2 bits)

MSCrumb      -> 4 2 A B 3 1 5 6 C A F E 5 6 8 7        (expand 4)
             -> 4 -> (base-2) -> 0 1 0 0        = 01  (first 2 bits)
```

Byte Ordering Systems (Endianness)

- Refers to the convention for **interpreting the byte order of multi-byte data** in memory.
- **Little Endian**:
    - Least significant byte (LSB) is stored at the lowest memory address.
    - Multi-byte data is **stored in ascending order** of significance.
    - The address of the multi-byte is the address of the LSB.

```
Example 1: 32-bit data 0x12345678 stored at address 0000

Address    Value
0003       12    |
0002       34    |
0001       56    |
0000       78    v

address 0000 contains 16-bit data = 5678
address 0000 contains 32-bit data = 12345678
```

- **Big Endian**:
    - Most significant byte (MSB) is stored at the lowest memory address.
    - Multi-byte data is **stored in descending order** of significance.
    - The address of the multi-byte is the address of the MSB.

```
Example 1: 32-bit data 0x12345678 stored at address 0000

Address    Value
0003       78    ^
0002       56    |
0001       34    |
0000       12    |

address 0000 contains 16-bit data = 1234
address 0000 contains 32-bit data = 12345678
```

FAQ: What happens if endianness is ignored?

? If endianness is not accounted for **during data exchange between systems** with different conventions, the interpretation of multi-byte data will be incorrect. For example, `0x12345678` in Big Endian may be misinterpreted as `0x78563412` in Little Endian.