Information Volume

Information volume has been doubled every year:

Total digital information generated in the world:

In 2012, 2.8 ZB (= $2.8 \times 10^{21}$ B)

→ In 2020, 8.59 ZB (5.25 TB / person)


Potential Market Growth in Nanotechnology

Nanotechnology growth:

Market size in major storage:

* http://www.fuji-keizai.com/e/report/ww_nano_product_e.html;
** http://www.fcr.co.jp/, http://www.yano.co.jp/
Contents of Information Storage and Spintronics

Lectures : Atsufumi Hirohata (atsufumi.hirohata@york.ac.uk, P/Z 019)

Advancement in information storages and spintronics (Weeks 2 ~ 9)

All lectures will be uploaded weekly in advance at

http://www-users.york.ac.uk/~ah566/lectures/lectures.html

13:30 ~ 14:30 Mons. (B/B 006 & online – Zoom)
12:00 ~ 13:00 Thus. (online – Zoom)

I. Introduction to information storage (01 & 02)
II. Magnetic information storages (03 ~ 06)
III. Solid-state information storages (07 ~ 11)
IV. Spintronic devices (12 ~ 18)

Practicals :

Analysis on a spintronic device using XRD, VSM, MFM and MR (Weeks 3 ~ 8)
Operation, data and instruction will be uploaded weekly in advance at

http://www-users.york.ac.uk/~ah566/lectures/lectures.html

09:00 ~ 11:00 Weds. (online – Zoom)

Continuous Assessment :

Assignment to be submitted via VLE (Week 10).

References

Magnetic storages :
- S. X. Wang and A. M. Taratorin, Magnetic Information Storage Technology
- C. D. Mee and E. D. Daniel, Magnetic Recording

Semiconductor storages :
- D. Richter, Flash Memories: Economic Principles of Performance, Cost and Reliability
- J. Brewer and M. Gill, Nonvolatile Memory Technologies with Emphasis on Flash:
  A Comprehensive Guide to Understanding and Using Flash Memory Devices

Spintronics :

Lecture notes / slides :
http://www-users.york.ac.uk/~ah566/lectures/lectures.html

Q & A : Slack page as invited
Miniaturisation and Integration in Semiconductor Devices

Moore’s law: *

“The number of transistors on a chip will double every 18 months.” (1965)

10 years later he revised this to “every 24 months.”

→ The development speed becomes even faster!

* http://www.intel.com/
Fabrication rules and technology:

Areal density in a hard disc drive (HDD) doubles every months. (~ 1992)

After giant magnetoresistance (GMR) implementation, it doubles less than every months. (1992 ~)
Similar to Moore’s law: Data transfer becomes faster.

Further miniaturisation is too expensive:

Analogue to our life:

Brain =  
• Rapid thinking enables fast operation.

Desktop =  
• A wide desk enables more operations in parallel.

Drawers =  
• More drawers enable more data storage.

* http://support.nifty.com/tsushin/cs/column/detail/090831543366/1.htm

Information Technology Pyramid

Layered structures between CPU and storages:

* http://www.howstuffworks.com/computer-memory1.htm
In 1940s, John von Neumann developed a basic model for a computer. *

Von Neumann categorised into 5 key components:
- CPU
- Input
- Output
- Working storage
- Permanent storage

Connections between the Components

Connections between CPU, in/outputs and storages:

* http://karbosguide.com/books/parchitecture/chapter02.htm

** http://karbosguide.com/books/parchitecture/chapter06.htm
How Does a CPU Work?

Data transfer between CPU and working storage:

A 32-bit CPU can handle data in different sized packets:
- **bytes** (8 bits)
- **half-words** (16 bits)
- **words** (32 bits)
- **blocks** (larger groups of bits)

CPU Architecture

For example, VIA Nano processor:

* [http://karbosguide.com/books/parchitecture/chapter10.htm](http://karbosguide.com/books/parchitecture/chapter10.htm)

Recent development by AMD, Trinity CPU / GPU architecture:

* https://arstechnica.com/information-technology/2012/05/amd-ships-trinity-processor-aims-for-a-piece-of-intels-ultrabook-market/

Instruction Set Architecture for CPU

Assembly language:

CPU can only handle:
- Binary code

Mnemonic:
- ADD / SUB etc.

Number of syntax:
- Typically 100 ~ 200
- Maximum ~ 400

* http://www.wikipedia.org/
Memory Access

In principle, only the CPU can (re-)write memory contents:


![Diagram of memory access](image)

Read Memory

In a 32-bit CPU, the address is typically 32-bit integer:


0x00000000 ~ 0xFFFFFFFF

= 4,294,967,296 = 4G

Address:

32-bit integer data

= Byte (1 Byte = 8 bit)

= addresses to be stored

Read 32-bit integer data

Read four 4 Byte data from the addresses of a, (a+1), (a+2), (a+3) and construct them into one data.
Write Memory

For example, write 32-bit integer data of “0x12345678”:

- 8-bit data: 0x12, 0x34, 0x56, 0x78

To store these data:
- Most significant bit is stored at the lowest address. → **endianness**
- Least significant bit is stored at the lowest address. → **endianness**

<table>
<thead>
<tr>
<th>Memory</th>
<th>0</th>
<th>a</th>
<th>(a+1)</th>
<th>(a+2)</th>
<th>(a+3)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>12</td>
<td>34</td>
<td>56</td>
<td>78</td>
<td></td>
</tr>
</tbody>
</table>

**endianness**

Endianness is the opposite.

<table>
<thead>
<tr>
<th>Memory</th>
<th>0</th>
<th>a</th>
<th>(a+1)</th>
<th>(a+2)</th>
<th>(a+3)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>78</td>
<td>56</td>
<td>34</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>

**endianness**

These are named after "Gulliver’s Travels".

* A. Nalamori, *Interface* Feb., 44 (2006);

** http://www.wikipedia.org/

---

Memory Access for Big / Little Endian

Data bus holds the same data:

![Memory Access Diagram]

Nowadays most CPU can handle both endianness.

→ **endianness**

Bit and Byte

Bit:

"Binary digit" is a basic data size in information storage.

1 bit : $2^n$ combinations ; digit in binary number

<table>
<thead>
<tr>
<th>$n$</th>
<th>$2^n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
</tbody>
</table>

Byte:

A data unit to represent one letter in Latin character set.

1 byte (B) = $2^n$ bit

<table>
<thead>
<tr>
<th>$n$</th>
<th>$2^n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>kB</td>
<td>×</td>
</tr>
<tr>
<td>1</td>
<td>kB</td>
</tr>
<tr>
<td>MB</td>
<td>×</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
</tbody>
</table>