# Chapter 1.Introduction

## 1.1 TURING MODEL(圖靈模型)

- **Alan Turing**首次在**1936年**提出
  - 他提出所有的計算都可以由一種特殊的機器來執行，現在稱為“**圖靈機**”。

## 1.1.3 The universal Turing machine(通用圖靈機)

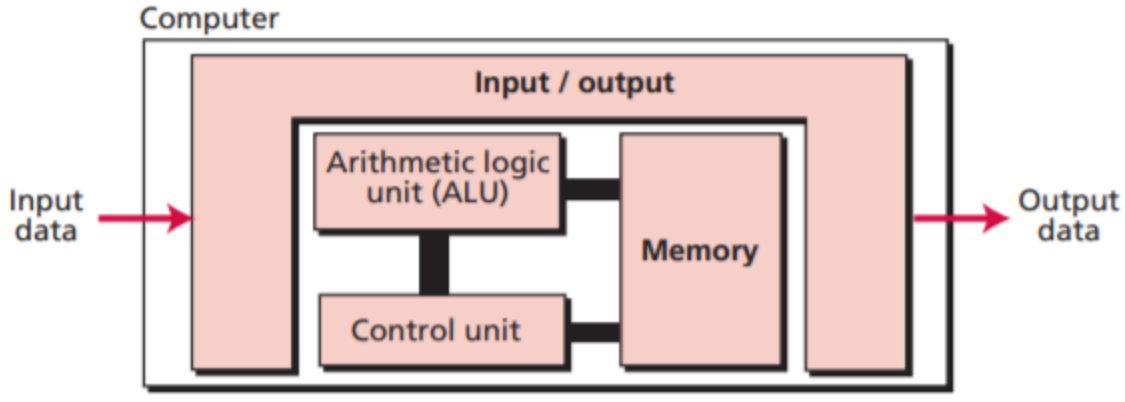
- 通用圖靈機，如果合適，可以進行任何計算的機器
  提供的程序，是對現代計算機的**第一次描述**。

## 1.2 VON NEUMANN MODEL(馮諾依曼模型)

- 建立在圖靈通用機上的計算機將數據存儲在它們的內存中。
- 大約**1944-1945年**，**John von Neumann**提出，由於程序和數據在邏輯上是同樣，程序也應該存儲在計算機的內存中。

## 1.2.1 四個子系統

- 建立在馮諾依曼模型上的計算機將計算機硬件分為四部分子系統：存儲器(**memory**)、算術邏輯單元(**arithmetic logic unit**)、控制單元(**control unit**)和輸入/輸出(**input/output**)




---

- 存儲器(memory)
  - 內存是存儲區域。這是處理過程中存儲程序和數據的地方
- 運算邏輯單元(Arithmetic logic unit)
  - 算術邏輯單元 (ALU) 是進行計算和邏輯運算的地方
- 控制單元(Control unit)
  - 控制單元控制存儲器、ALU 和輸入/輸出的操作

- 輸入/輸出(input/output)
    - 輸入子系統接受來自計算機外部的輸入數據和程序，而輸出子系統將處理結果發送給外界。

## 1.2.2 The stored program concept(存儲程序概念)

- 馮諾依曼模型指出程序必須存儲在**內存**中。 這是完全不同於早期計算機的體系結構，其中只有**數據**存儲在內存中：他們的任務程序是通過操作一組開關或通過改變接線系統。
- 現代計算機的內存承載著**程序**及其相應的**數據**。
- 這意味著數據和程序都應該具有**相同的格式**，因為它們存儲在**內存**中。

## 1.3 COMPUTER COMPONENTS(計算機組件)

- 計算機由三部分組成：計算機硬件、數據和計算機軟件。

## 1.4 HISTORY

- 計算和計算機的歷史分為三個部分-**機械機器時期（1930年之前）**，**電子計算機時期（1930-1950）**，以及包括五個現代計算機世代在內的時期。

## 1.4.1 Mechanical machines(機械機器時期) (before 1930)

- 在此期間，發明了幾種幾乎沒有相似之處的計算機器。
- 17世紀，法國數學家和哲學家**布萊斯·帕斯卡**（**Blaise Pascal**），發明了**Pascaline**，一種用於加法和減法運算的機械計算器。
- 17 世紀後期，德國數學家**戈特弗里德萊布尼茨**發明了更複雜的機械計算器，可以進行乘法和除法以及加法和減法。它被稱為萊布尼茨輪。
- 20世紀，當**Niklaus Wirth**發明了**結構化編程語言**，他稱它為帕斯卡，以紀念第一台機械計算器的發明者。

補充資料(可以略過)

- 第一台使用存儲和編程思想的機器是提花機織機，由 Joseph-Marie Jacquard 在 19 世紀初發明。織機使用穿孔卡片（就像一個存儲程序）來控制紡織品製造中的經線。
- 1890 年，在美國人口普查局工作的 Herman Hollerith 設計並建造了一種可以自動讀取、統計和排序存儲在計算機上的數據的編程器機器穿孔卡片。

## 1.4.2 The birth of electronic computers (1930-1950)(電子計算機時期)

- 在**1930年**到**1950年**之間，科學家們發明了幾台計算機，他們可以被認為是電子計算機行業的先驅，早期的電子計算機並沒有將程序存儲在內存中，所有的都是**外部編程**。

- 第一台對信息進行電編碼的專用計算機是由**John V. Atanasoff**和他的助手**Clifford Berry**在**1939年**。它被稱為**ABC（Atana-soff Berry Computer）**，專門用於求解線性方程組。
- 第一台基於**馮諾依曼思想**的計算機於**1950年**製造於賓夕法尼亞州，被稱為**EDVAC**。與此同時，一台類似的計算機叫做**EDSAC**由英國劍橋大學的**Maurice Wilkes**建造。

## 1.4.3 Computer generations (1950-至今)

- **1950年**後製造的計算機或多或少遵循馮諾依曼模型。他們有變得更快、更小、更便宜，但原理幾乎相同。
  - 第一代（大約**1950~1959年**）的特點是出現了商業電腦。 在此期間，計算機僅由**專業人員**使用。
  - 第二代計算機（大約**1959~1965年**）使用晶體管代替真空管。這減小了計算機的**尺寸**和**成本**，並使它們負擔得起到**中小企業**。
  - 第三代(大約**1965年~1975年**)集成電路（晶體管、佈線和其他元件在一個電路上)，以及單芯片的發明進一步降低了計算機的成本和尺寸。小型機出現在市場上。罐頭程序（俗稱軟件包）變得可用。
  - 第四代（大約**1975~1985年**）見證了微型計算機的出現。第一台桌面計算器**Altair 8800**於**1975年**面世。電子工業允許整個計算機子系統安裝在單個電路板上。
  - 第五代這個開放的一代始於**1985年**，見證了**筆記本電腦**的出現和**掌上電腦**，**二級存儲介質**（CD-ROM、DVD等等)，多媒體的使用，以及虛擬實境。

# Chapter 2.

## 2.2 Positional number system

- binary system—->2進位
- octal system—->8進位
- decimal system—->10進位
- hexadecimal system—->16進位

| Decimal | Binary | Octal | Hexadecimal |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| 2 | 10 | 2 | 2 |
| 3 | 11 | 3 | 3 |
| 4 | 100 | 4 | 4 |
| 5 | 101 | 5 | 5 |

| Decimal | Binary | Octal | Hexadecimal |
|---|---|---|---|
| 6 | 110 | 6 | 6 |
| 7 | 111 | 7 | 7 |
| 8 | 1000 | 10 | 8 |
| 9 | 1001 | 11 | 9 |
| 10 | 1010 | 12 | A |
| 11 | 1011 | 13 | B |
| 12 | 1100 | 14 | C |
| 13 | 1101 | 15 | D |
| 14 | 1110 | 16 | E |
| 15 | 1111 | 17 | F |

## 2.3 Nonposaitional number systems

- Roman number system

| I | V | X | L | C | D | M |
|---|---|---|---|---|---|---|
| 1 | 5 | 10 | 50 | 100 | 500 | 1000 |

- 兩數相比小數字放左為減，放右為加
  - ex : XIX–->10+(-1+10)=19

# Chapter 3.

## 3.2 Storing Number

1. unsigned integers
   - 空位補0
   - ex : (7)10–->(111)2–->(0111)24bits
2. sign and magnitude number
   - 首位數字決定正負
     - 0–->正
     - 1–->負
3. overflow 溢位
   - 若數字到達bits所能表達的上限，下一個數字將進位導致溢位

| | A+B=C | dec |
|---|---|---|
| | A | 14 |
| | B | 9 |
| | C(正解，但溢位) | 23 |
| | 顯示錯誤 | 7 |

- 負數同理

4. two's complement (NOT+1)

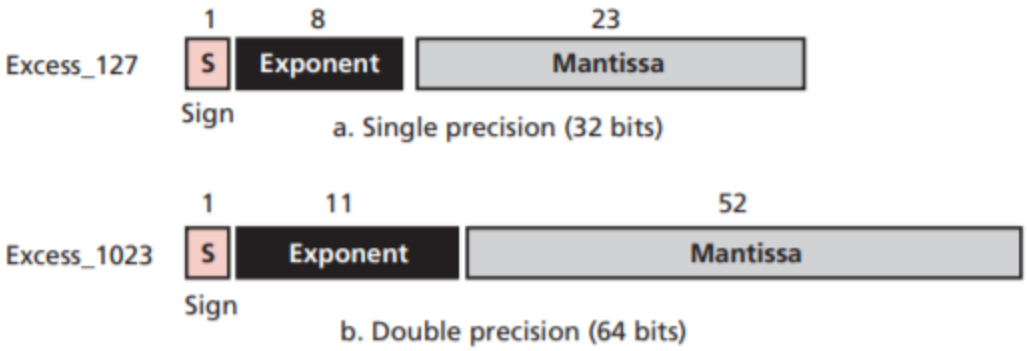
| 8-bit bin | 1 1 0 0 1 0 0 1 |
|---|---|
| two's operation | 0 0 1 1 0 1 1 0 + 1 |

5. sign + two's complement (現今代換正負最常用的方法)

| dec | 8-bit bin |
|---|---|
| 55 | 00110111 |
| -55 | 11001001 |

## 3.2.4. IEEE standards for Floating-point representation

- Sign + Shifter (or Exponent) + Fixed-point number
- **S**ign–->標誌
  - 0–->正
  - 1–->負
- **E**xponent–->指數
  - 原數指數8bits=>127−原數指數
  - 原數指數11bits=>1023−原數指數
- **M**antissa–->尾數




  - 空位補0

**例題 1.**

Show the Excess_127 (single precision) representation of the decimal number 5.75.

***Solution***

a. The sign is positive, so S = 0.
b. Decimal to binary transformation: $5.75 = (101.11)_2$.
c. Normalization: $(101.11)_2 = (1.0111)_2 \times 2^2$.
d. $E = 2 + 127 = 129 = (10000001)_2$, M = 0111. We need to add 19 zeros at the right of M to make it 23 bits.
e. The presentation is shown below:

| S | E | M |
|---|---|---|
| 0 | 10000001 | 01110000000000000000 |

The number is stored in the computer as 01000000101110000000000000000000.

## 3.3 Storing Text

### ASCII(American Standard Code for Information Interchange)

- 7bits
- 128 Symbols

### Unicode

- 32bits
- 4294967296 Symbols

## 3.5 Storing images

- 圖像使用兩種不同的技術存儲在計算機中：**光柵圖形**(raster graphics)和**矢量圖形**(vector graphics)。
- 紅、綠、藍三基色（通常稱為 RGB）
- 本色(True-Color)
  - 用於編碼像素的技術之一稱為**真彩色**，它使用 **24位元**來編碼一個像素。
  - 數字只有0~255

| *Color* | *Red* | *Green* | *Blue* | *Color* | *Red* | *Green* | *Blue* |
|---|---|---|---|---|---|---|---|
| Black | 0 | 0 | 0 | Yellow | 255 | 255 | 0 |
| Red | 255 | 0 | 0 | Cyan | 0 | 255 | 255 |
| Green | 0 | 255 | 0 | Magenta | 255 | 0 | 255 |
| Blue | 0 | 0 | 255 | White | 255 | 255 | 255 |

# Chapter 4.

## Chapter 4.1 Logic Opearations

### NOT (one input 在計算中較其他三種有優先權)

| X | output |
|---|---|
| 0 | 1 |
| 1 | 0 |

| NOT | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | Input |
| | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | Output |

### OR (類似電路並聯，其一通則通)

| X | Y | output |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | Input 1 |
| OR | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | Input 2 |
| | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | Output |

### AND (類似電路串聯，需要兩個都通過)

| X | Y | output |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | Input 1 |
| AND | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | Input 2 |
| | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | Output |

## XOR (exclusive-or,互斥或)

| X | Y | output |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | Input 1 |
| XOR | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | Input 2 |
| | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | Output |

(找不一樣的為1)

# ex:(99)16XOR[(33)16AND[NOT(00)16]]=(AA)16

1. NOT(00)16 : 00000000–->11111111=(FF)16

| NOT | Binary |
|---|---|
| (00)16 | 00000000 |
| 輸出 | 11111111 |

2. (FF)16AND(33)16–->00110011=(33)16

| AND | Binary |
|---|---|
| (FF)16 | 11111111 |
| (33)16 | 00110011 |
| 輸出 | 00110011 |

3. (99)16XOR(33)16–->10101010=(AA)16

| XOR | Binary |
| --- | --- |
| (99)16 | 10011001 |
| (33)16 | 00110011 |
| 輸出 | 10101010 |

# Chapter 5.

## 5.1 Introduction

- a computer
  - the central processing unit
  - the main memory
  - the input/output subsystem

### 5.2.1 The **C**entral(中央) **P**rocessing(處理) **U**nit(單元) and The **A**rithmetic(算術) **L**ogic(邏輯) **U**nit(單元)

- Logic operation(**邏輯運算**) : 計算邏輯運算。例如:**NOT**、**OR**、**AND**和**XOR**。
- Shift operation (**位移運算**) : 包括邏輯移位運算和算術移位運算。
- Arithmetic operation (**算術運算**) : 整數和實數的計算技巧（標準化）(**Standardization**)。

### 5.2.2 Registers (暫存器)

- Data registers (**資料暫存器**) :
  - 存儲輸入數據、中間結果和輸出數據。
- Instruction registers (**指令暫存器**) :
  - 為了存儲指令，解碼它們，然後執行它們。
- Program counter (**程式計數器**) :
  - 繼續執行指令並尋找下一條指令的地址。

### 5.2.3 Control Unit (控制單元)

- 控制各個子系統的運行。

## 5.3 Main Memory (主要記憶體)

### 5.3.1 Address space

- |Unit|Exact Number of Bytes|Approximation|
  |---|---|---|
  |kilobyte|210=1024|103|
  |megabyte|220=1048576|106|
  |gigabyte|230=1073741824|109|
  |terabyte|240|1012|

## 5.3.2 Memory types

1. RAM(**R**andom **A**ccess **M**emory) :
   又稱「**暫存記憶體**」，用於儲存接下來要執行的資料，**可隨時讀寫，讀寫速度快**，通常做為臨時資料儲存媒介。

- SRAM (靜態隨機存取記憶體 , **S**tatic **R**andom **A**ccess **M**emory) :
  - 使用傳統的正反閘來保存資料
  - 只要電源是開啟狀態，則資料就被保存
  - 斷電後就**喪失資料**。
- DRAM (動態隨機存取記憶體 , **D**ynamic **R**andom **A**ccess **M**emory) :
  - 使用電容來做資料儲存
    - 如果電容是充滿電的，則狀態為 1
    - 如果電容未充電，則狀態為 0
  - 但電容會隨時間漏電，所以DRAM記憶體單元**需要週期性地加以更新**。

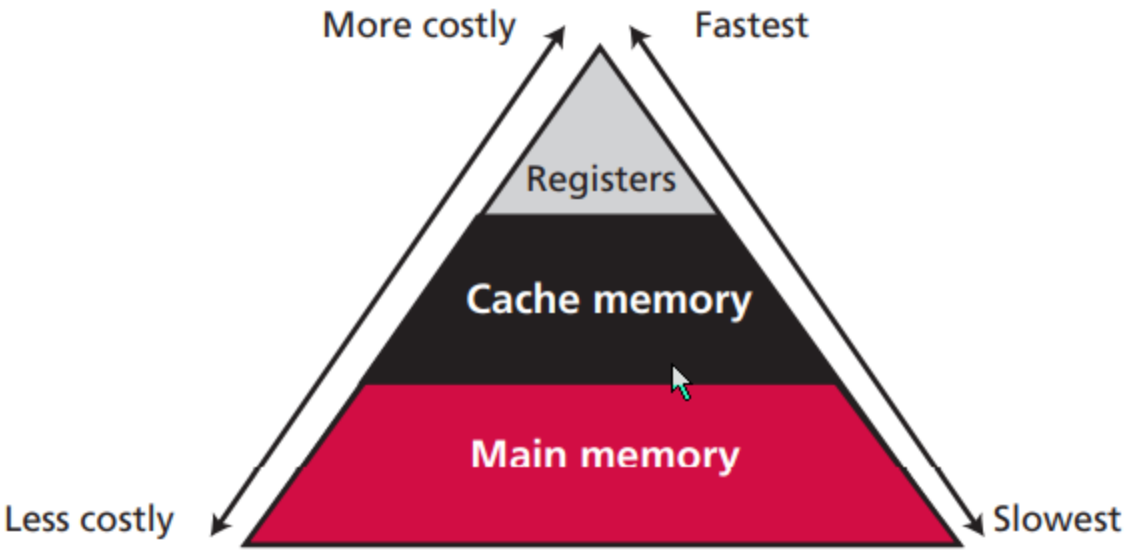
2. ROM (唯讀記憶體 , **R**ead-**O**nly **M**emory) :
   非易失性 , 可以斷電保存 , CPU不能寫入資料 , 通常用來**存放開機資料及啟動程式**。

- PROM (
  **P**rogrammable可程式規劃 **R**ead-**O**nly唯讀 **M**emory記憶體
  )
  - 為另類的 ROM
  - 一開始為空白的記憶體，使用者可用特殊設備將程式儲存於其中
  - 放入後**不可再覆寫**。
- EPROM (
  **E**rasable可清除 **P**rogrammable可程式規劃 **R**ead-**O**nly唯讀 **M**emory記憶體
  ) :
  - 為另類的 PROM
  - 使用者可用特殊設備將程式儲存於其中，也可用特殊的紫外光設備清除資料
  - 可以使用**電子脈衝**對 EEPROM 進行編程和擦除，而**無需從計算機中取出**。
- EEPROM (
  **E**lectrically **E**rasable電子抹除式 **P**rogrammable可程式規劃 **R**ead-**O**nly唯讀 **M**emory記憶體

) :

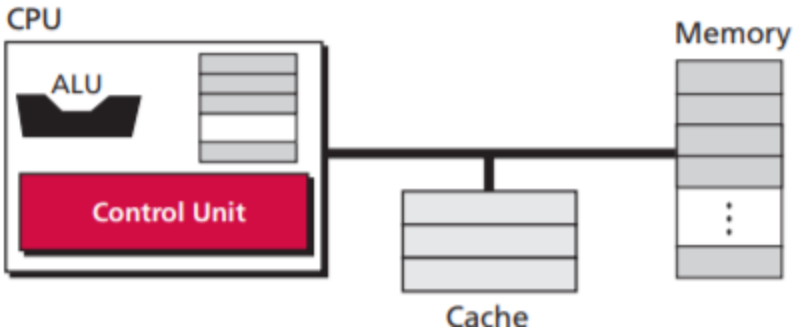
- 為另類 EPROM
- 可藉由**電子脈衝來加以程式化和清除，而不需從電腦中移除**。

### 5.3.3 Memory hierarchy




### 5.3.4 Cache Memory (快取記憶體)

- 比主記憶體快，但是比CPU及其內部的暫存器慢。快取記憶體通常數量**少**，並置於**CPU與主記憶體之間**。




## 5.4 Input/Output(I/O) Subsystem

### 5.4.1 Nonstorage devices

- keyboard
- monitor
- printer

### 5.4.2 Storage devices

- magnetic storage devices
  - magnetic disks
  - magnetic tape
- optical storage devices
  - CD-ROMs
  - CD-R
    - R=recordable
  - CD-RW
    - RW=rewritable
  - DVD

## 5.5 Subsystem Interconnection

### 5.5.1 Connecting CPU and memory

1. Data dus
2. Address bus
3. Control bus

### 5.5.2 Connecting I/O devices

1. Controllers
2. FireWire
3. **U**niversal **S**erial **B**us(USB)
4. **H**igh-**D**sfinition **M**ultimedia **I**nterface(HDMI)

### 5.5.3 Addressing I/O devices

- Isolated I/O
- Memory-mapped I/O

## 5.6 Progam Execution

### 5.6.1 Machine cycle

1. fetch:取出
2. decode:解析/碼
3. execute:執行

Created with Raphaël 2.2.0startinstructionsFetchDecodeExecuteendyesno

## 5.6.2 I/O operation

- programmed I/O
    - 在ready階段執行busy waiting–->不斷詢問(主動)

Created with Raphaël 2.2.0startmore wordsIssue I/O commandCheck device statusreadyTransfer a wordendyesnoyesno

- interrupt-driven I/O
    - 在interrupt階段等待通知(被動)

Created with Raphaël 2.2.0startmore words?Issue I/OcommandinterruptTransfer a wordendyesno

- **D**irect **M**emory **A**ccess(DMA)
    - ready and finished

Created with Raphaël 2.2.0startIssue I/OcommandreadyWaitfinishedend

## 5.7 Differnt Architetures

1. CISC
2. RISC
3. Pipelining

## 5.7.4 Parallel processing

- CPU核心分工
    - SISD(一對一)
    - SIMD(一對多)
    - MISD(多對一)
    - MIMD(多對多)