效能定義

(1). 計算机效能評估兩指標：

① response time：工作從開始到完成所需時間

② throughput：單位時間內完成工作量

(2). 若為 PC，則：$\text{Performance} = \dfrac{1}{\text{execution time}}$

(2). execution time 分類：

- elapsed time
  - I/O time
  - CPU time
    - system CPU time
    - user CPU time
      - absolute：wall time
      - relative：clock cycle time

∵ I/O time，system CPU time 會受 program 數目影响

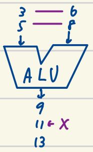
在分析上為不確定因素，故下面 execution time 皆以 user CPU time 為主

(3). user CPU time 量測：

在現今計算机中用固定頻率的 clock 來協調硬体动作同步訊号

同步訊号是為了同步快慢有別訊号

e.g.

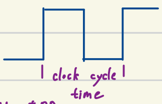


⇒ ALU 左边电路可能延遲時間較短

訊号會較 ALU 右边訊号快送達

若無 clock 來同步運算元，運算上會出錯

一个完整 clock 如右圖中所示：



clock cycle time 為一个 clock 所花的時間

即從邏輯 0 和 1 間切換時間

而 clock rate 為 1s 有多少个 clock

(4). 以 Clock Cycle Time 計算 CPU time

$$\text{CPU Time} = \text{CPU clock cycles} \times \text{clock cycle time}$$

$$= \frac{\text{CPU clock cycles}}{\text{clock rate}}$$

$$\overset{☆}{=} \underbrace{IC}_{①} \times \underbrace{CPI}_{②} \times \text{clock cycle time}$$

① 对相同机器上運行兩 program 分析 execution time 時

∵ clock cycle time 相同，只需看 CPU clock cycles

② 对兩机器但運行相同 ISA 相同 program 分析 execution time 時

∵ Instruction count 相同，只需看 instruction time

Summary：

Execution Time (CPU time)

- CPU clock cycles $= \sum CPI_i \times C_i$
  - Instruction Count
  - $CPI = \sum CPI_i \times Freq_i$
- cycle time $= \dfrac{1}{\text{cycle rate}}$
- Instruction Time $= \dfrac{1}{\text{instruction rate}} = \dfrac{1}{MIPS \times 10^6}$

e.g.

	+	×	÷
CPI_i	2	5	3
C_i	50	20	30
$Freq_i$	0.5	0.2	0.3
CPI	$\frac{2\times50+5\times20+3\times30}{100}$		

$= 2\times0.5 + 5\times0.2 + 3\times0.3$

MIPS 作為效能評估的謬誤

MIPS (Million instruction per second)：每秒可執行之百萬个指令數

Instruction rate

$$= \frac{IC}{\text{execution time}} = \frac{IC}{IC \times CPI \times \text{cycle time}} = \frac{\text{cycle rate}}{CPI}$$

MIPS

$$= \frac{IC}{\text{execution time} \times 10^6} = \frac{IC}{IC \times CPI \times \text{cycle time} \times 10^6} = \frac{\text{cycle rate}}{CPI \times 10^6}$$

$$\therefore MIPS = \frac{\text{instruction rate}}{10^6}$$

MIPS 越大，表示 1s 可執行指令數較多

但，不能做為 performance 評估標準原因有三：

1. 沒有考量每一指令能力 (e.g. MIPS 皆為 1 的 CISC, RISC 顯然 CISC 較強)
2. 同一电腦不同程式之 MIPS 可能不同
3. MIPS 有時甚至和效能成反比

影响 Performance 的軟硬体因素　P.284 表格

Algorithm：將問題解決的思維 ⇒ 要透過 programming language 實現

Programming language：實現演算法的高階語言 ⇒ 需由 complier translation 成机器可執行的 instruction

Complier：將 programming language translate 成 machine code，而所有指令的集合 + 一些硬体資訊為 ISA

ISA：軟，硬体間 interface ⇒ 需透過 computer organization 來 implement

Computer Organization：實際硬体設計 ⇒ 由电子元件構成，需 VLSI 技術

VLSI 技術：∵ 电子元件越小，之間距離越短，傳遞信号時間越短

Hardware or software component	Affects what?	How?
Algorithm	Instruction count, possibly CPI	The algorithm determines the number of source program instructions executed and hence the number of processor instructions executed. The algorithm may also affect the CPI, by favoring slower or faster instructions. For example, if the algorithm uses more divides, it will tend to have a higher CPI.
Programming language	Instruction count, CPI	The programming language certainly affects the instruction count, since statements in the language are translated to processor instructions, which determine instruction count. The language may also affect the CPI because of its features; for example, a language with heavy support for data abstraction (e.g., Java) will require indirect calls, which will use higher CPI instructions.
Compiler	Instruction count, CPI	The efficiency of the compiler affects both the instruction count and average cycles per instruction, since the compiler determines the translation of the source language instructions into computer instructions. The compiler's role can be very complex and affect the CPI in complex ways.
Instruction set architecture	Instruction count, clock rate, CPI	The instruction set architecture affects all three aspects of CPU performance, since it affects the instructions needed for a function, the cost in cycles of each instruction, and the overall clock rate of the processor.

① Computer Organization

	CPI	cycle time
single cycle machine	小	大
multi cycle machine	大	小
pipeline machine	小	小

② ISA

	IC	CPI	cycle time
RISC	大	小	小 ⇒ RISC 較簡單 同步訊号量較小
CISC	小	大	大

Complier 不影响個別指令CPI，而是平均CPI，個別指令CPI 只由 HW 決定

Ⅲ Amdahl's Law

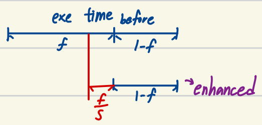
Amdahl's Law 用作計算一机算机中某一部份改善後，对整体的執行時間的改善

公式為：改善後 execution time $= \dfrac{\text{受改善影响的 exe time}}{\text{改善倍率}}$ + 未受影响部份之 execution time

$$Speedup = \frac{\text{exe time}_{before}}{\text{exe time}_{after}} = \frac{\text{exe time}_{before}}{\frac{\text{exe time}_{enhanced}}{s} + \text{exetime}_{non\text{-}enhanced}}$$

定義：$\boxed{f = \dfrac{\text{exe time}_{enhanced}}{\text{exe time}_{before}}}$

$$= \frac{1}{\frac{f}{s} + (1-f)}$$



$\therefore Speedup = \dfrac{1}{\frac{f}{s} + (1-f)}$　令 $s = \infty$ 則 $speedup = \dfrac{1}{1-f}$ #

效能總評方法

1. 利用算術平均 (AM) (假設 workload 中每一 program 執行頻率相同)
2. 利用權重算術平均 (WAM)
3. 利用 Normalize 後，再作几何平均 (GM)

e.g. 1. (unit: s)

	machine A	machine B
program A	10.0	100.0
program B	1000.0	10.0

則由 AM 可得，machine A 為：$\frac{1010}{2} = 505$ (s)

machine B 為：$\frac{110}{2} = 55$ (s)

∴ machine B 比 A 快 $\frac{505}{55}$ 倍

2. (unit: s)

	machine A	machine B	Freq
program A	10.0	100.0	0.9
program B	1000.0	10.0	0.1

則 WAM 為：machine A = 109 (s)

B = 91 (s)

3. 選定一部 machine 為 reference machine

利用 SPEC ratio $= \dfrac{\text{Ex Time}_{ref}}{\text{Ex Time}_{measured}} = \dfrac{\text{Performance}_{mea}}{\text{Performance}_{ref}}$

∴ SPEC ratio 越大，效能越好

但 ∵ $\dfrac{AM(X_i)}{AM(Y_i)} \neq AM\left(\dfrac{X_i}{Y_i}\right)$，無法用 AM 來对 Normalize 後的 SPEC ratio

做評估，得用 $GM = \sqrt[n]{\prod_{i=1}^{n} \text{Exe Time ratio}_i}$

但 GM (SPEC ratio) 無法預測 ExeTime，較不準