chapter one

Summary

it contains:-

- Summary for total topics in chapter one
- Suggested practical questions and their Solutions

Sondos abu idaq

→ Eight great ideas

* design for moore's law
* use abstraction to simplify design : improve design
* make common case fast. : Amdah's law
* performance via parallelism. : زيادة العدد بقلل الانتاجية
* performance via pipelining : step بتخلص وبروح عال step التالي
* performance via prediction :- مواضيع بحثية وتنبؤ
* Hierarchy of memories. : غير قادرين على تحقيق capacity عالية بسعر رخيص وبسرعة عالية توازي سرعة الـ cpu، لذلك الحل يكون بتقسيم الميموري لـ (cach, DRAM, SSD..)
* dependability via redundancy : ← احتمالية الخطأ وإجراء حسابات

* Technology Trends:

← تعطل في تطوير في الـ Frequency بسبب الـ Transistors فيزيائياً الأهم نوعين من الـ power
- static
- dynamic

← dynamic power : طاقة بصرفها الـ Transistor لما يعمل عمليات "تبديل" [تحويل من 1 لـ 0]

تعتمد على: ① مربع الـ Voltage
② Frequency

← زيد الـ Freq ← بتزيد الـ pow ← مش قادرين تبرد

← ليش صرنا نخفض لـ Frequency ؟

١← ما بقدر أخفض أقل من الـ Voltage رح يسببلي عطل بالكهربا

٢← صارت الـ Freq شبه ثابتة

← آخر مرحلة من 2005 تقريباً : صرنا نزبط الـ performance فزادت الـ cores

* Tech trends : $P_{dynamic} = N * C * V^2 f * A$

* كل سنة وزن لستين تقريباً كانت تضاعف عدد الـ chips بشكل مش لينير بل exponential

* عند فترة معينة ثبت الـ power بسبب مشاكل مثل Leak current

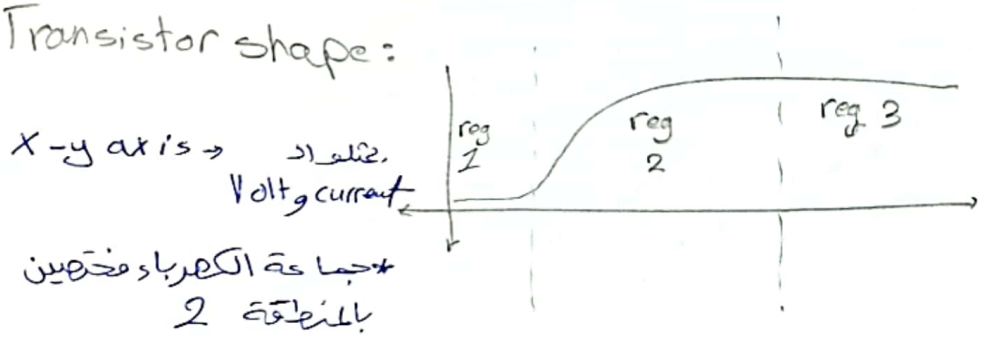
* هاظو Leakage current : المقصد به عند تصغير الـ Transistors، يحدث تسريب الـ e من مسارها بعد تسريبها تتجمع مع بعضها

* بسبب صعوبة التحكم بـ Leak current خلال الـ dynamic pow process وصل إلى حد استنزاف الـ dynamic pow بنسبة 50 - 60%

→ electronics technology continues to evolve : ① increased capacity and performance

② reduced cost

Transistor shape :



---

* المادة الرئيسية : Silicon ← semiconductor ← Transistors الـ تبنى عليها

---

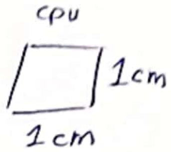
* Manufacturing ICs :

Silicon ingot → Slicer → Blank Wafers → patterned wafer → wafer tester → Tested wafer → Tested dies

Ship to customers ← Tested packaged dies ← part tester ← Bond die packages

yield:

عدد الـ dies الصالحات / العدد الكلي



عليه ← 1 مليار Transistors

← كلما زادت مساحة الـ die صارت تكلفتها أعلى .

300 mm wafer , 280 chips , 32nm tech , Each chip is 20.7 * 10 mm

- 300 mm → القطر
- 280 chips → مطبوع عليها بقدر هذا العدد → مش شرط كلهم صالحين
- 32nm tech → كلما كان الرقم أصغر يكون الـ Transis أقل ، بتحدد → ① حجم Transis ② صغار أو كبار
- 20.7 * 10 mm → أبعادها

→ Cost per die = $\frac{\text{Cost per wafer}}{\text{die per wafer} * \text{yield}}$

→ die per wafer = wafer area / die area ⇒ لكن اللي على الـ wafer عدد الـ chip ← المفروض أكثر من 280 chip الأطراف مش كاملين الشكل لذلك ← cancel them

wafer area: πr^2 , die area: الطول * العرض

→ defect area → بتحددها تكنولوجيا التصنيع

• die area → بتتحكم فيها التصميم نفسه

→ yeild = $\frac{1}{(1 + (\text{defect per area} * \text{die area} / 2))^2}$

→ defects per area = $\frac{\text{\# of defects}}{\text{wafer area}}$

→ NonLinear relation to area and defect rate

- wafer cost & area are fixed
- defect rate determined by ⇒ manufacturing process
- die area determined by architecture and circuit design

Response time ⇒ instruction كم الوقت لينفذ

throughput time ⇒ يقيس كمية الانجاز خلال فترة زمنية معينة

→ replacing processor with faster version

respo: يقل

Throughput: يزيد

→ adding more processors

resp: اسرع

thraugh: No → لما تاسك تكون على واحدة cpu
yes
↳ كل cpu رح ياخذ request ⇒ يتأثر

→ performance = 1 / Execution time

→ "X faster than y" ⇒ $\frac{\text{perf X}}{\text{perf y}} = \frac{\text{execut.t y}}{\text{execut.t X}} = N$ times faster

→ Throughput = $\frac{P}{T}$ → عدد الـ prll units . ⇒ prll / pip لما اشتغل على pip تكون فش دقيق

→ measuring execution time :-

→ elapsed time : total response time, including all aspects الوقت التقعي
[processing, I/O, Os, idle, determine Sys performance]

→ Cpu time : time spent processing given Job [user cpu, Sys cpu time]

→ clock period: duration of clock cycle

→ clock frequency: cycles per second

→ cpu time = cpu clock cycles * C.C. Time

cpu clock cycles = Ic * cpi

$$\text{cpu time} = \frac{Ic * cpi}{\text{clock rate}}$$

→ to improve performance
- reduce # of clock cycles
- increasing clock rate
- Hardware designer must often trade off clock rate against cycle count

→ Ic determines by
[program, ISA & compile]

→ cpi determines by
[cpu hardware]

→ if different instructions have different cpi ∴ cpi affected by instruction mix

---

$$\rightarrow CPI = \frac{\text{clock cycles}}{\text{Instruction count}} = \sum_{i=1}^{n} \left(cpi * \frac{Ic_i}{Ic_{total}} \right)$$

$$\rightarrow \text{relative frequency} = \frac{Ic_i}{Ic_{Tot}}$$

$$\rightarrow \text{clock cycles} = \sum_{i=1}^{n} (cpi * Ic_i)$$

→ performance depends on:
- Algorithms → Ic, cpi
- programming Language → Ic, cpi
- compiler → Ic, cpi
- ISA → Ic, cpi, Tc

→ power trends ∴ In cMos Ic technology

→ power = $\frac{1}{2}$ capasitive Load * Voltage^2 * Frequency

dynamic power

← كل ما زاد عدد الـ transistors بقل حجمها بالتالي تقل الـ power

→ Static power = supply voltage * Leakage current

∴ multiprocessor حسن الـ performance طريقة ← تحسين

parallel + multicore = high performance

∴ Hard to

① programming for performance

② Load balancing

③ Optimizing communication and synchronization

→ Reducing power

Suppose a new cpu has ① 85% of capasitive Load of old cpu

② 15% voltage and 15% frequency reduction

Then $\frac{P_{new}}{P_{old}} = 0.52$

→ power wall : ① we can't reduce voltage fuo ther

② we can't remove more heat

→ How else we can improve performance ?

* بنزيد الـ performance عن طريق multiprocessor

SPEC cpu Benchmark

→ SPEC : develops benchmarks for cpu , I/O , Web
→ elapsed time for execute selection of programs (focus on cpu performance)

→ Normalize relative to refrence machine

→ Summarize as geometric mean of performance ratios
CINT 2006 → integer CFP 2006 → floating

⇒ $\sqrt[n]{\prod_{i=1}^{n} \text{execution time ratio}_i}$

جذر ←

→ SPEC Power Benchmarks
performance : ssj_ops /sec
power : watts (Joul/sec)

→ power bench work = $\left(\sum_{i=0}^{20} ssj\_ops\right) / \left(\sum_{i=0}^{10} power_i\right)$

→ SPEC [speed] integer and floating ⇒ are used for comparing time to complete single tasks

→ SPEC [rate] integer and floating ⇒ measure the throughput or work per unit of time

---

* كل ما زاد الـ target load بزيد الـ average power فإنه : بعبر في شغل وكل ما زاد بزيد الاستهلاك للطاقة

Amdahl's Law :

→ execution time
- → $T_{affected}$
- → $T_{unaffected}$

→ Improvement factor $= \frac{T_{old}}{T_{imp}}$

→ $T_{improved} = \frac{T_{affected}}{imp\ factor} + T_{unaffected}$

Mips
- → does not account for differences in ISAs btw computers
- → does not account for differences in Complexity btw instructions

→ $Mips = \frac{I_c}{Execution\ time * 10^6} = \frac{clock\ rate}{cpi * 10^6}$

→ Cpi varies btw programs on given CPU.

Improving Performance

Our favorite program runs in 10 seconds on computer A, which has a 2 GHz clock. We are trying to help a computer designer build a computer, B, which will run this program in 6 seconds. The designer has determined that a substantial increase in the clock rate is possible, but this increase will affect the rest of the CPU design, causing computer B to require 1.2 times as many clock cycles as computer A for this program. What clock rate should we tell the designer to target?

Let's first find the number of clock cycles required for the program on A:

$$\text{CPU time}_A = \frac{\text{CPU clock cycles}_A}{\text{Clock rate}_A}$$

$$10\,\text{seconds} = \frac{\text{CPU clock cycles}_A}{2 \times 10^9 \frac{\text{cycles}}{\text{second}}}$$

$$\text{CPU clock cycles}_A = 10\,\text{seconds} \times 2 \times 10^9 \frac{\text{cycles}}{\text{second}} = 20 \times 10^9 \text{ cycles}$$

CPU time for B can be found using this equation:

$$\text{CPU time}_B = \frac{1.2 \times \text{CPU clock cycles}_A}{\text{Clock rate}_B}$$

$$6 \text{ seconds} = \frac{1.2 \times 20 \times 10^9 \text{ cycles}}{\text{Clock rate}_B}$$

$$\text{Clock rate}_B = \frac{1.2 \times 20 \times 10^9 \text{ cycles}}{6 \text{ seconds}} = \frac{0.2 \times 20 \times 10^9 \text{ cycles}}{\text{second}} = \frac{4 \times 10^9 \text{ cycles}}{\text{second}} = 4 \text{ GHz}$$

To run the program in 6 seconds, B must have twice the clock rate of A.

Using the Performance Equation

Suppose we have two implementations of the same instruction set architecture. Computer A has a clock cycle time of 250 ps and a CPI of 2.0 for some program, and computer B has a clock cycle time of 500 ps and a CPI of 1.2 for the same program. Which computer is faster for this program and by how much?

We know that each computer executes the same number of instructions for the program; let's call this number I. First, find the number of processor clock cycles for each computer:

$$\text{CPU clock cycles}_A = I \times 2.0$$
$$\text{CPU clock cycles}_B = I \times 1.2$$

Now we can compute the CPU time for each computer:

$$\begin{aligned}\text{CPU time}_A &= \text{CPU clock cycles}_A \times \text{Clock cycle time} \\ &= I \times 2.0 \times 250 \text{ ps} = 500 \times I \text{ ps}\end{aligned}$$

Likewise, for B:

$$\text{CPU time}_B = I \times 1.2 \times 500 \text{ ps} = 600 \times I \text{ ps}$$

Clearly, computer A is faster. The amount faster is given by the ratio of the execution times:

$$\frac{\text{CPU performance}_A}{\text{CPU performance}_B} = \frac{\text{Execution time}_B}{\text{Execution time}_A} = \frac{600 \times I \text{ ps}}{500 \times I \text{ ps}} = 1.2$$

We can conclude that computer A is 1.2 times as fast as computer B for this program.

Amddal's Law

1.15 Assume a program requires the execution of 50×10^6 FP instructions, 110×10^6 INT instructions, 80×10^6 L/S instructions, and 16×10^6 branch instructions. The CPI for each type of instruction is 1, 1, 4, and 2, respectively. Assume that the processor has a 2 GHz clock rate.

1.15.1 [10] <§1.11> By how much must we improve the CPI of FP instructions if we want the program to run two times faster?

1.15.2 [10] <§1.11> By how much must we improve the CPI of L/S instructions if we want the program to run two times faster?

1.15.3 [5] <§1.11> By how much is the execution time of the program improved if the CPI of INT and FP instructions is reduced by 40% and the CPI of L/S and Branch is reduced by 30%?

1.15.1 Clock cycles = $CPI_{fp} \times$ No. FP instr. + $CPI_{int} \times$ No. INT instr. + $CPI_{l/s} \times$ No. L/S instr. + $CPI_{branch} \times$ No. branch instr.

T_{CPU} = clock cycles/clock rate = clock cycles/2×10^9

clock cycles = 512×10^6; $T_{CPU} = 0.256$ s

To have the number of clock cycles by improving the CPI of FP instructions:

$CPI_{improved\ fp} \times$ No. FP instr. + $CPI_{int} \times$ No. INT instr. + $CPI_{l/s} \times$ No. L/S instr. + $CPI_{branch} \times$ No. branch instr. = clock cycles/2

$CPI_{improved\ fp}$ = (clock cycles/2 − ($CPI_{int} \times$ No. INT instr. + $CPI_{l/s} \times$ No. L/S instr. + $CPI_{branch} \times$ No. branch instr.)) / No. FP instr.

$CPI_{improved\ fp} = (256 - 462)/50 < 0 ==>$ not possible

1.15.2 Using the clock cycle data from a.

To have the number of clock cycles improving the CPI of L/S instructions:

$CPI_{fp} \times$ No. FP instr. + $CPI_{int} \times$ No. INT instr. + $CPI_{improved\ l/s} \times$ No. L/S instr. + $CPI_{branch} \times$ No. branch instr. = clock cycles/2

$CPI_{improved\ l/s}$ = (clock cycles/2 − ($CPI_{fp} \times$ No. FP instr. + $CPI_{int} \times$ No. INT instr. + $CPI_{branch} \times$ No. branch instr.)) / No. L/S instr.

$CPI_{improved\ l/s} = (256 - 198)/80 = 0.725$

1.15.3 Clock cycles = $CPI_{fp} \times$ No. FP instr. + $CPI_{int} \times$ No. INT instr. + $CPI_{l/s} \times$ No. L/S instr. + $CPI_{branch} \times$ No. branch instr.

T_{CPU} = clock cycles/clock rate = clock cycles/2×10^9

$CPI_{int} = 0.6 \times 1 = 0.6$; $CPI_{fp} = 0.6 \times 1 = 0.6$; $CPI_{l/s} = 0.7 \times 4 = 2.8$; $CPI_{branch} = 0.7 \times 2 = 1.4$

T_{CPU} (before improv.) = 0.256 s; T_{CPU} (after improv.) = 0.171 s

Solutions:

1.12.1 CPI = clock rate × CPU time/instr. count

clock rate = 1/cycle time = 3 GHz

CPI(bzip2) = $3 \times 10^9 \times 750/(2389 \times 10^9) = 0.94$

1.12.2 SPEC ratio = ref. time/execution time

SPEC ratio(bzip2) = 9650/750 = 12.86

1.12.3 CPU time = No. instr. × CPI/clock rate

If CPI and clock rate do not change, the CPU time increase is equal to the increase in the number of instructions, that is 10%.

1.12.4 CPU time(before) = No. instr. × CPI/clock rate

CPU time(after) = 1.1 × No. instr. × 1.05 × CPI/clock rate

CPU time(after)/CPU time(before) = 1.1 × 1.05 = 1.155. Thus, CPU time is increased by 15.5%.

1.12.5 SPECratio = reference time/CPU time

SPECratio(after)/SPECratio(before) = CPU time(before)/CPU time(after) = 1/1.1555 = 0.86. The SPECratio is decreased by 14%.

1.12.6 CPI = (CPU time × clock rate)/No. instr.

CPI = $700 \times 4 \times 10^9/(0.85 \times 2389 \times 10^9) = 1.37$

1.12.7 Clock rate ratio = 4 GHz/3 GHz = 1.33

CPI @ 4 GHz = 1.37, CPI @ 3 GHz = 0.94, ratio = 1.45

They are different because, although the number of instructions has been reduced by 15%, the CPU time has been reduced by a lower percentage.

Benchmarks

*For the reference machine results for the **bzip2**, and **libquantum** benchmarks, refer to the slides that we explained in class on how to measure SPECRatios*

1.12 The results of the SPEC CPU2006 bzip2 benchmark running on an AMD Barcelona has an instruction count of 2.389E12, an execution time of 750 s, and a reference time of 9650 s.

1.12.1 [5] <§§1.6, 1.9> Find the CPI if the clock cycle time is 0.333 ns.

1.12.2 [5] <§1.9> Find the SPECratio.

1.12.3 [5] <§§1.6, 1.9> Find the increase in CPU time if the number of instructions of the benchmark is increased by 10% without affecting the CPI.

1.12.4 [5] <§§1.6, 1.9> Find the increase in CPU time if the number of instructions of the benchmark is increased by 10% and the CPI is increased by 5%.

1.12.5 [5] <§§1.6, 1.9> Find the change in the SPECratio for this change.

1.12.6 [10] <§1.6> Suppose that we are developing a new version of the AMD Barcelona processor with a 4 GHz clock rate. We have added some additional instructions to the instruction set in such a way that the number of instructions has been reduced by 15%. The execution time is reduced to 700 s and the new SPECratio is 13.7. Find the new CPI.

1.12.7 [10] <§1.6> This CPI value is larger than obtained in 1.11.1 as the clock rate was increased from 3 GHz to 4 GHz. Determine whether the increase in the CPI is similar to that of the clock rate. If they are dissimilar, why?

MIPS and Actual Speed

A certain computer program is compiled to run on two different machines A and B with different ISAs and Designs. The compiled program therefore has different instruction counts on machines A and B, and due to the different internal designs, it has different CPIs. The measurements are summarized in the table below:

Measurement	Computer A	Computer B
Instruction count	10 billion	8 billion
Clock rate	4 GHz	4 GHz
CPI	1.0	1.1

a. Which computer has the higher MIPS rating?

b. Which computer is faster?

Solution of a:

$$\text{MIPS} = \frac{\text{Instruction count}}{\frac{\text{Instruction count} \times \text{CPI}}{\text{Clock rate}} \times 10^6} = \frac{\text{Clock rate}}{\text{CPI} \times 10^6}$$

$MIPS_A = 4 \times 10^9 / (1 \times 10^6) = 4000$

$MIPS_B = 4 \times 10^9 / (1.1 \times 10^6) = 3636.36..$

Based on the MIPS metric, machine A is faster as it has higher MIPS

Solution of b:

Actual program **execution time** on A is

$10{,}000{,}000{,}000 \times \frac{1}{4 \times 10^9} \times 1 =$ **2.5 seconds**

Actual program **execution time** on A is

$8{,}000{,}000{,}000 \times \frac{1}{4 \times 10^9} \times 1.1 =$ **2.2 seconds**

In fact, computer B is faster running the program than computer A by 13.63%. The MIPS rating is misleading because it does not take into account the complete picture. In this example, it did not take into account that due to machine B better ISA, design, and compiler, the total number of instructions was 20% less than machine B which heavily affects execution time.