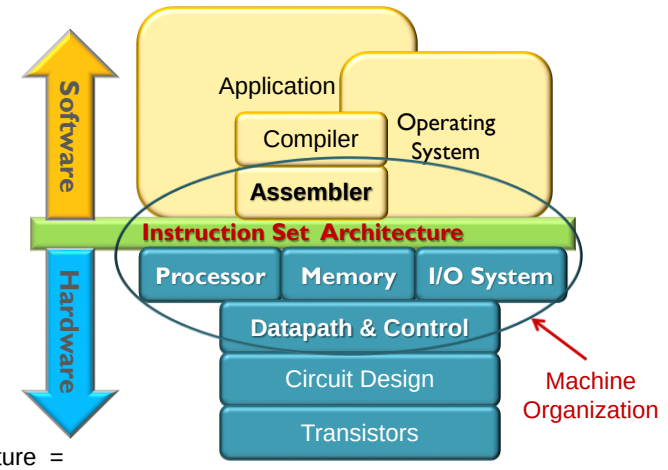
Prof. REN-SONG TSAY
NTHU

CHAPTER 1 COMPUTER ABSTRACTIONS AND TECHNOLOGY

What is Computer Architecture?



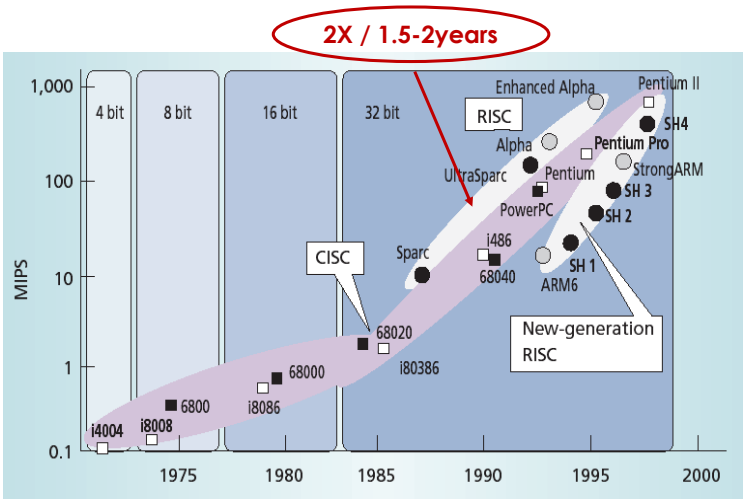
Computer Architecture =
Instruction Set Architecture
+ Machine Organization

§1.1 Introduction

The Computer Revolution

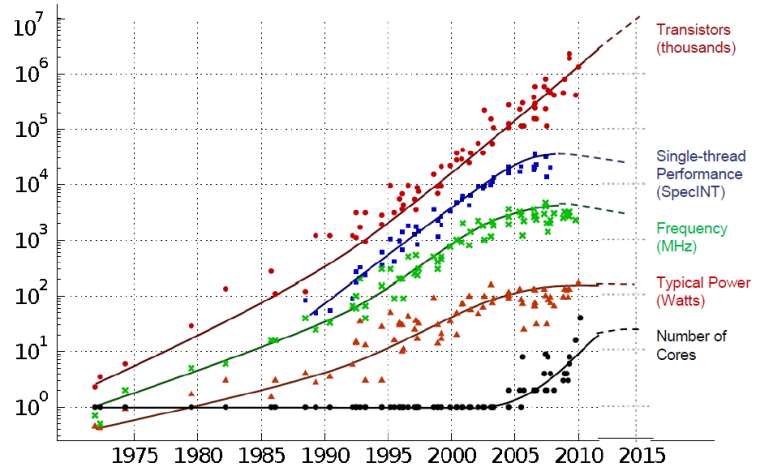
- Progress in computer technology
  - Underpinned by Moore's Law
- Makes novel applications feasible
  - Computers in automobiles
  - Smart phones
  - Human genome project
  - World Wide Web
  - Search Engines
  - Machine learning
- Computers are pervasive

Evolution of Processor Performance



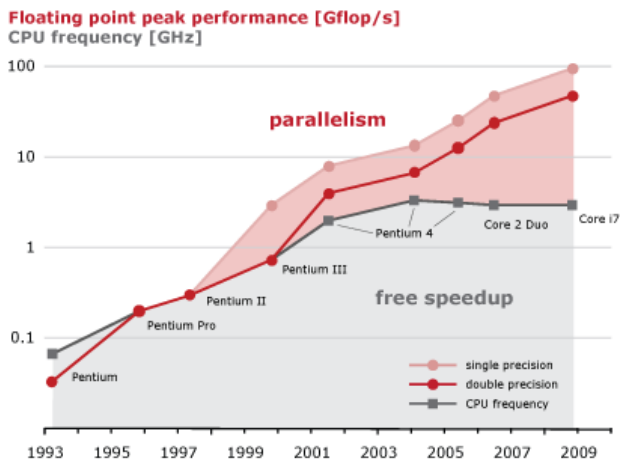
("The Cooler the Better: New Directions in the Nomadic Ages," *Computer*, April 2001.)

35 YEARS OF MICROPROCESSOR TREND DATA



Original data collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond and C. Batten
Dotted line extrapolations by C. Moore

Evolution of Processor Performance



Classes of Computers

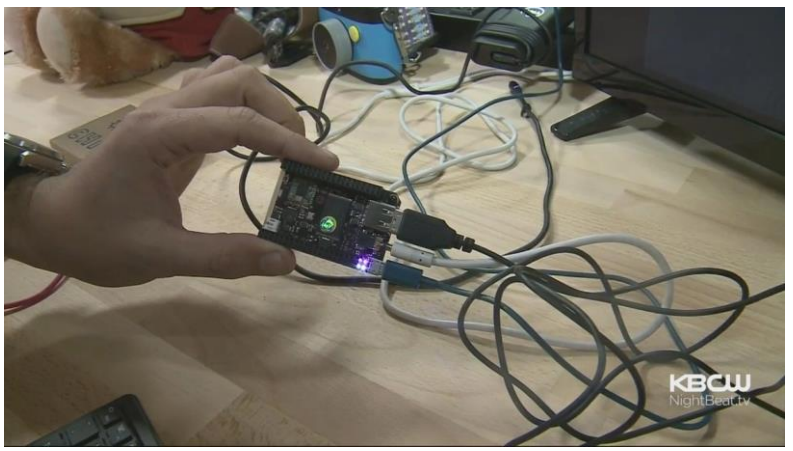
- Personal computers
  - General purpose, variety of software
  - Subject to cost/performance tradeoff
- Server computers
  - Network based
  - High capacity, performance, reliability
  - Range from small servers to building sized

2016.2.1 Microsoft testing underwater data centers

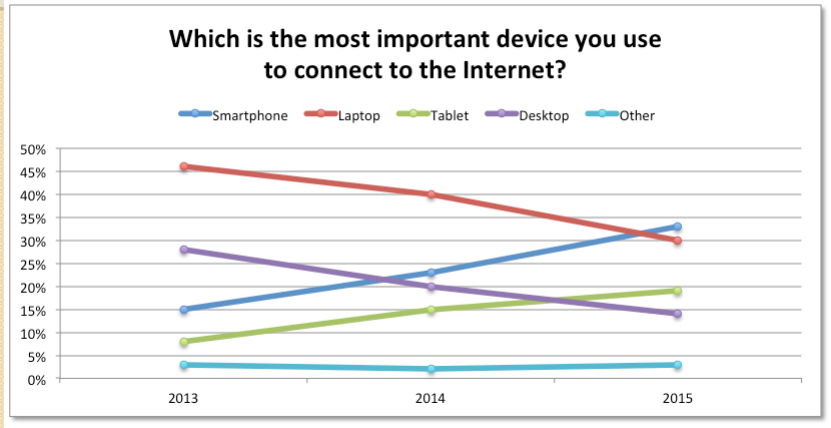
Classes of Computers

- Supercomputers
  - High-end scientific and engineering calculations
  - Highest capability but represent a small fraction of the overall computer market
- Embedded computers
  - Hidden as components of systems
  - Stringent power/performance/cost constraints

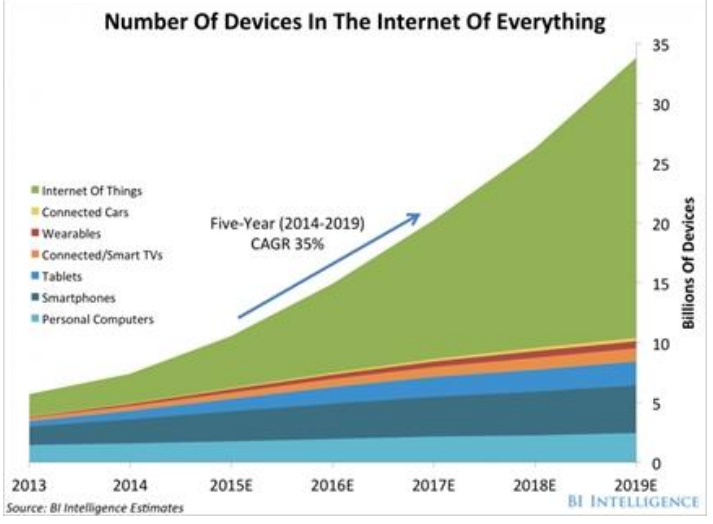
2016.2.11
$9 Computer Designed in Oakland



The PostPC Era

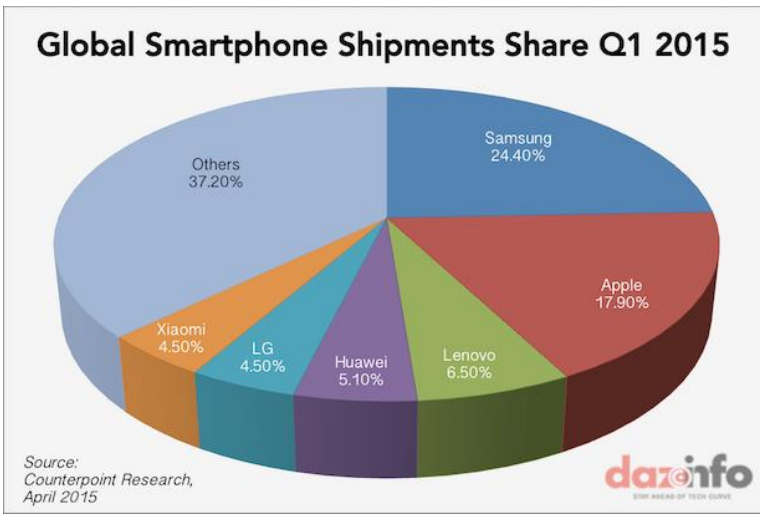


A New Trend

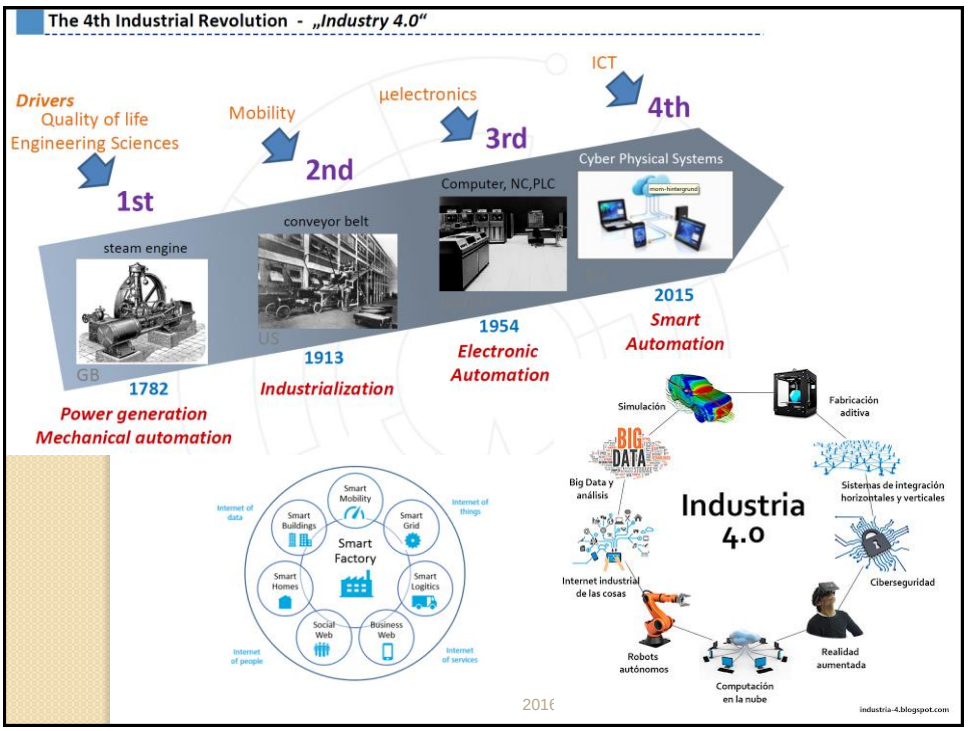


The PostPC Era

- Personal Mobile Device (PMD)
  - Battery operated
  - Connects to the Internet
  - Hundreds of dollars
  - Smart phones, tablets, electronic glasses
- Cloud computing
  - Warehouse Scale Computers (WSC)
  - Software as a Service (SaaS)
  - Portion of software run on a PMD and a portion run in the Cloud
  - Amazon and Google



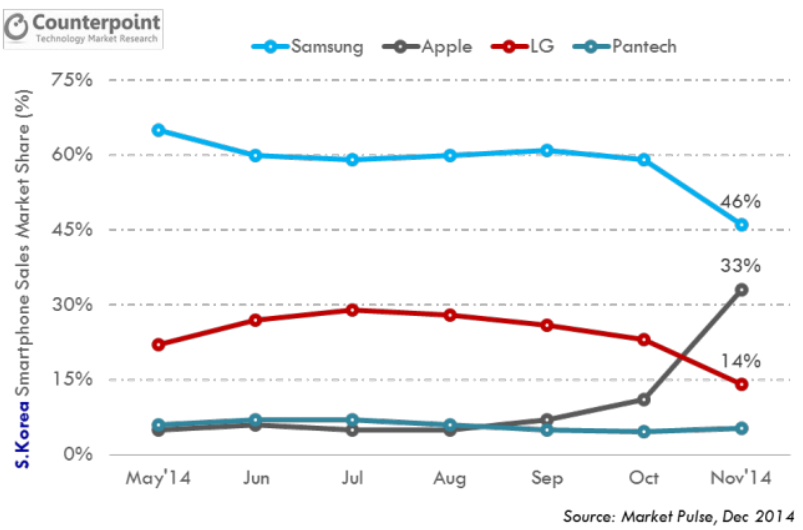

Wearables 1.0
Wearables 2.0
IRONMAN
INVISIBLE MAN
A MUCH More Diversified Market Than Investors Realize
CREDIT SUISSE
2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 16


The 4th Industrial Revolution - „Industry 4.0"
Drivers
Quality of life
Engineering Sciences
1st
Mobility
2nd
µelectronics
3rd
ICT
4th
steam engine
GB
1782
Power generation
Mechanical automation
conveyor belt
US
1913
Industrialization
Computer, NC,PLC
1954
Electronic
Automation
Cyber Physical Systems
2015
Smart
Automation
Smart Factory
Smart Mobility
Smart Buildings
Smart Grid
Smart Homes
Smart Logitics
Social Web
Business Web
Internet of data
Internet of things
Internet of people
Internet of services
Industria 4.0
Simulación
Fabricación aditiva
Big Data y análisis
Sistemas de integración horizontales y verticales
Internet industrial de las cosas
Ciberseguridad
Robots autónomos
Realidad aumentada
Computación en la nube
industria-4.blogspot.com
2016

World Largest Companies

2016.2.15 market value

- Apple $521B ($725B in 2015)
- Google $470B
- Facebook $290B
  - Taiwan 2015 budget $60 B
  - Taiwan GDP $505 B (2014)



Embedded Computer

Lego Mindstorms

- A computer inside another device used for running one predetermined application
- Uses: control (traffic, printer, disk); consumer electronics (video game, CD player, PDA)

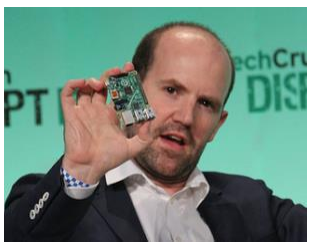
Robotic command explorer:
A "Programmable Brick",
Hitachi H8 CPU (8-bit), 32KB RAM,
LCD, batteries,
infrared transmitter/receiver,
4 control buttons, 6 connectors

Arduino

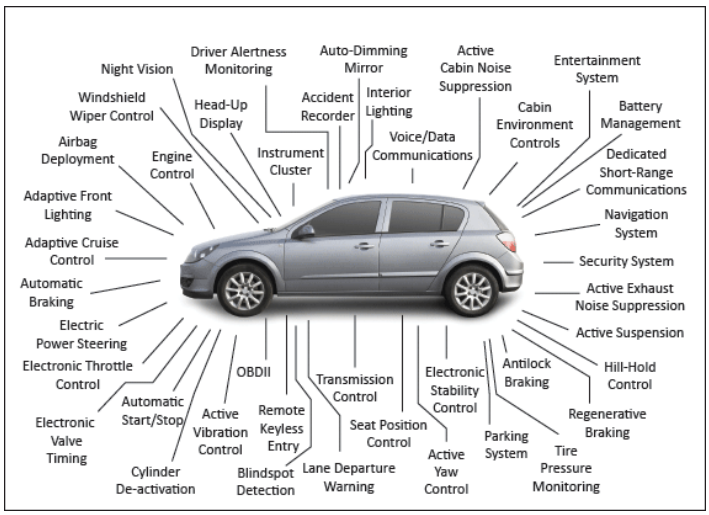
- 一個開放原始碼的單晶片微控制器
- 使用了Atmel AVR單片機
- 建構於簡易輸出/輸入（simple I/O）介面板，並且具有使用類似Java、C語言的Processing/Wiring開發環境

Raspberry Pi2

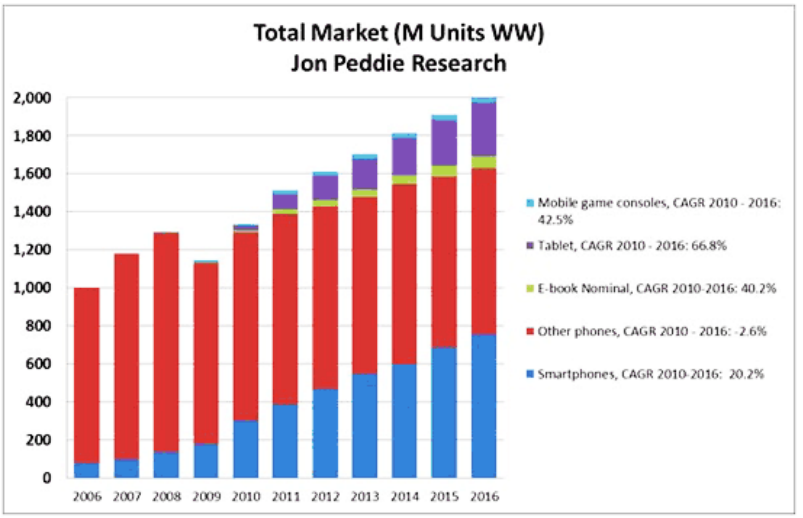
- Since 2012
- 火柴盒大小, $35
- 可搭載Open Source 的 Linux 系統
- 免費提供 Win 10 的開發者套件
- 創始人 Eben Upton

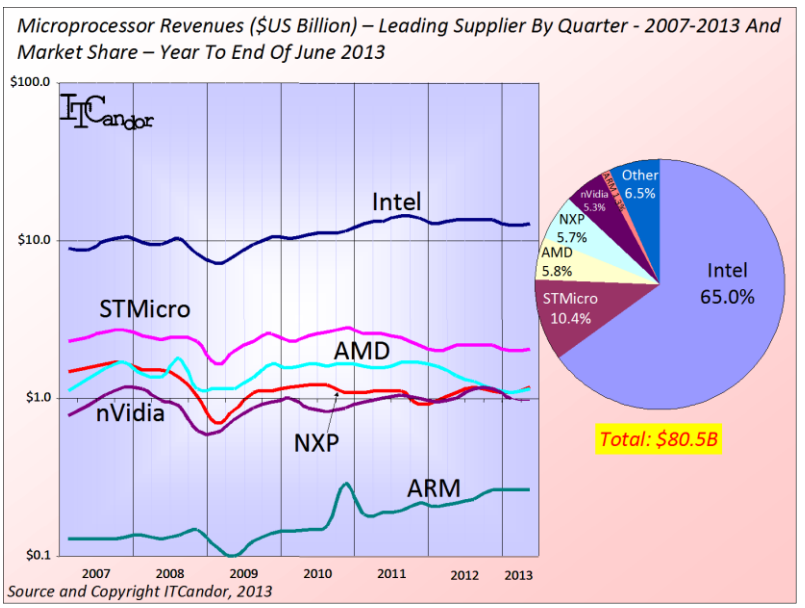


Embedded Everywhere

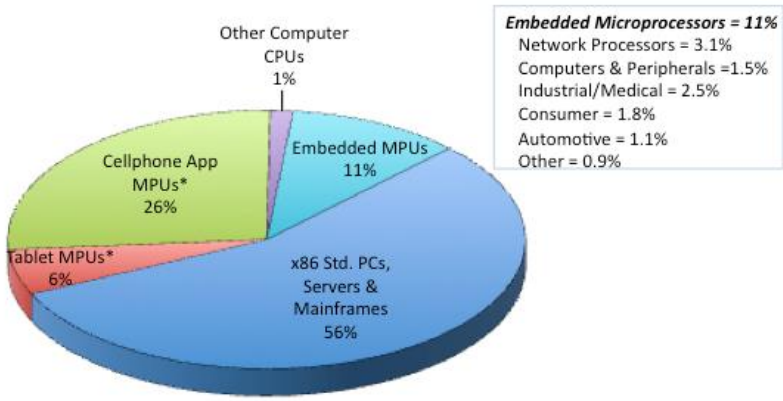


The Processor Market





2013 MPU Sales by Applications (Fcst, $61.0B)



*Includes ARM-based and x86 processors.

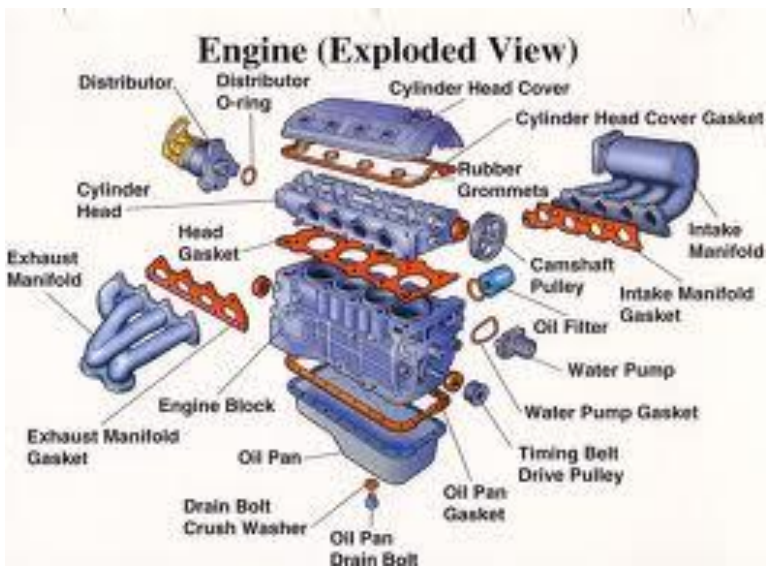
Source: IC Insights

What is coming …

- Internet-of-things
- Big data
- Intelligent life assistance
- Artificial intelligence
- Industry 4.0

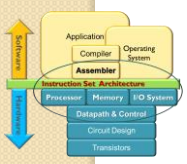
What You Will Learn

- How programs are translated into the machine language
  - And how the hardware executes them
- The hardware/software interface
- What determines program performance
  - And how it can be improved
- How hardware designers improve performance
- What is parallel processing



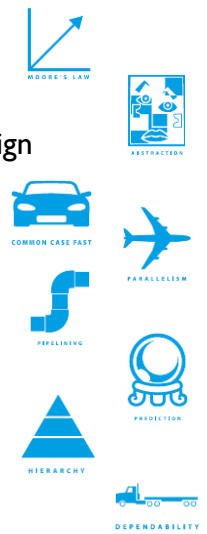
Understanding Performance

- Algorithm
  - Determines number of operations executed
- Programming language, compiler, architecture
  - Determine number of machine instructions executed per operation
- Processor and memory system
  - Determine how fast instructions are executed
- I/O system (including OS)
  - Determines how fast I/O operations are executed



Eight Great Ideas

- Design for ***Moore's Law***
- Use ***abstraction*** to simplify design
- Make the ***common case fast***
- Performance *via* ***parallelism***
- Performance *via* ***pipelining***
- Performance *via* ***prediction***
- ***Hierarchy*** of memories
- ***Dependability*** *via* redundancy

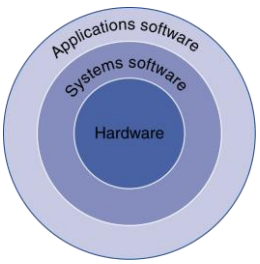


§1.2 Eight Great Ideas in Computer Architecture

Below Your Program

§1.3 Below Your Program

- Application software
  - Written in high-level language
- System software
  - Compiler: translates HLL code to machine code
  - Operating System: service code
    - Handling input/output
    - Managing memory and storage
    - Scheduling tasks & sharing resources
- Hardware
  - Processor, memory, I/O controllers



13 © Ren-Song Tsay, NTHU, Taiwan

Levels of Program Code

- High-level language
  - Level of abstraction closer to problem domain
  - Provides for productivity and portability
- Assembly language
  - Textual representation of instructions
- Hardware representation
  - Binary digits (bits)
  - Encoded instructions and data

High-level language program (in C)

```
swap(int v[], int k)
{int temp;
   temp = v[k];
   v[k] = v[k+1];
   v[k+1] = temp;
}
```

Compiler

Assembly language program (for MIPS)

```
swap:
      muli $2, $5,4
      add  $2, $4,$2
      lw   $15, 0($2)
      lw   $16, 4($2)
      sw   $16, 0($2)
      sw   $15, 4($2)
      jr   $31
```

ISA

Assembler

Binary machine language program (for MIPS)

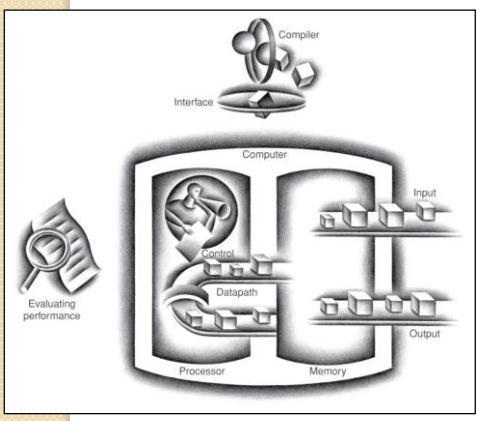
```
00000000101000010000000000011000
00000000000110000001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
```

15 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 33

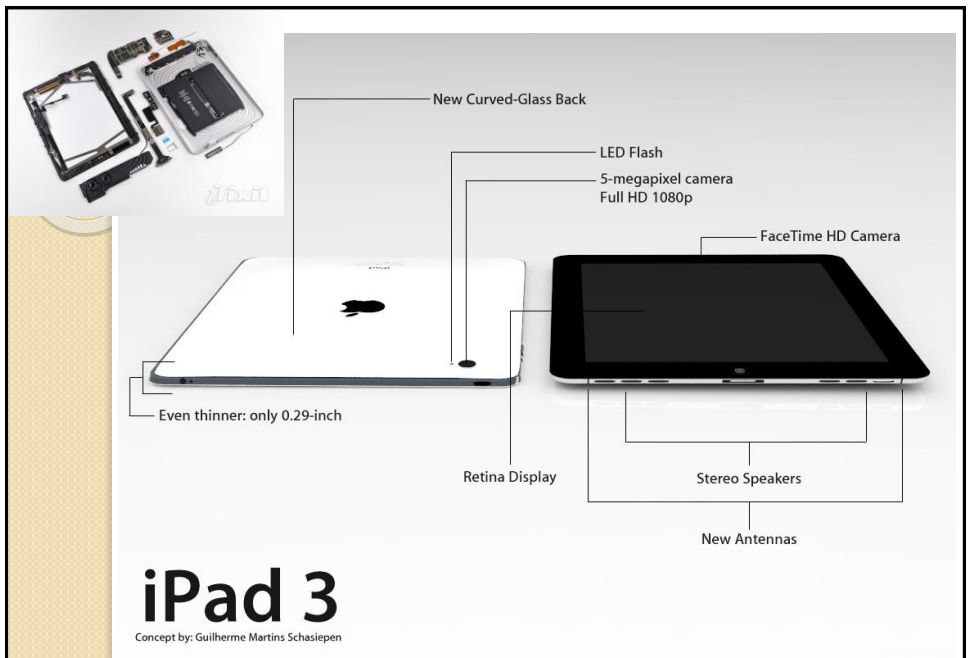
Components of a Computer

§1.4 Under the Covers

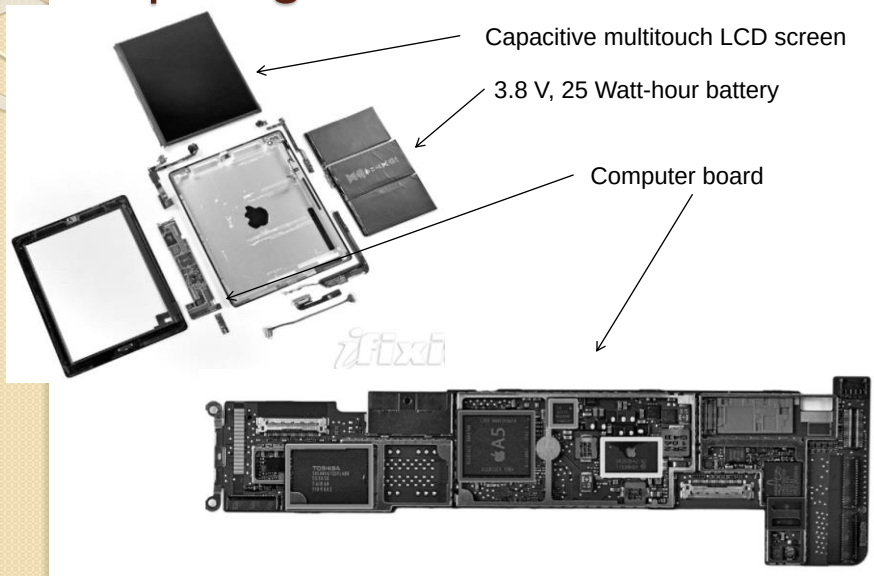
The BIG Picture



- Same components for all kinds of computer
  - Desktop, server, embedded
- Input/output includes
  - User-interface devices
    - Display, keyboard, mouse
  - Storage devices
    - Hard disk, CD/DVD, flash
  - Network adapters
    - For communicating with other computers



Opening the Box



iPhone components

- A4 processor (1GHz ARM Cortex A8)
- 512 MB RAM
- Gorilla Glass
- 1420 mAh Li-Ion
- 5M Pixel CCD
- SAMSUNG flash
- TI Touch Panel Controller
- Cirrus Logic Codec

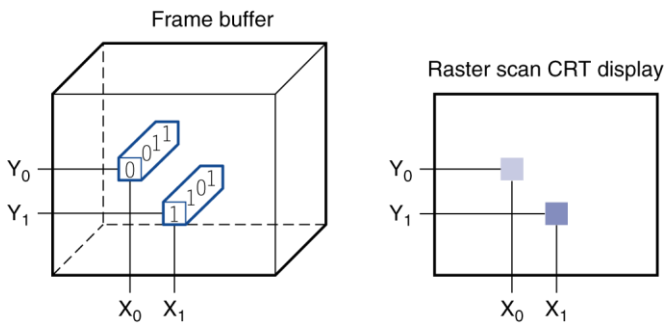
Touchscreen

- PostPC device
  Supersedes keyboard and mouse
- Resistive and Capacitive types
  - Most tablets, smart phones use capacitive
  - Capacitive allows multiple touches simultaneously
- Human Interface
  - sensors

19

Through the Looking Glass

- LCD screen: picture elements (pixels)
  - Mirrors content of frame buffer memory



18

Inside the Processor (CPU)

- Datapath: performs operations on data
- Control: sequences datapath, memory, ...
- Cache memory
  - Small fast SRAM memory for immediate access to data

Inside the Processor Apple A5

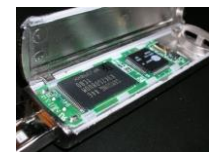
Abstractions

The BIG Picture

- Abstraction helps us deal with complexity
  - Hide lower-level detail
- Instruction set architecture (ISA)
  - The hardware/software interface
- Application binary interface
  - The ISA plus system software interface
- Implementation
  - The underlying details and interface

A Safe Place for Data

- Volatile main memory
  - Loses instructions and data when power off
- Non-volatile secondary memory
  - Magnetic disk
  - Flash memory
  - Optical disk (CDROM, DVD)

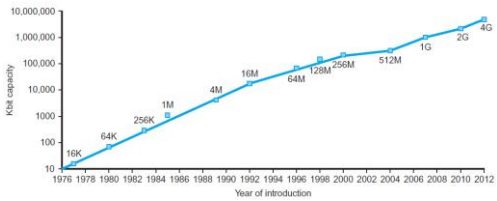


Networks

- Communication and resource sharing
- Local area network (LAN): Ethernet
  - Within a building
- Wide area network (WAN): the Internet
- Wireless network: WiFi, Bluetooth

Technology Trends

- Electronics technology continues to evolve
  - Increased capacity and performance
  - Reduced cost



DRAM capacity

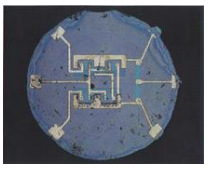
Year	Technology	Relative performance/cost
1951	Vacuum tube	1
1965	Transistor	35
1975	Integrated circuit (IC)	900
1995	Very large scale IC (VLSI)	2,400,000
2013	Ultra large scale IC	250,000,000,000

§1.5 Technologies for Building Processors and Memory

IC Evolution

Vacuum Tube

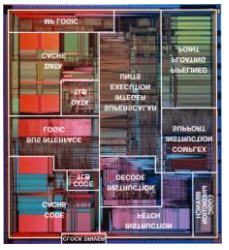
Transistor



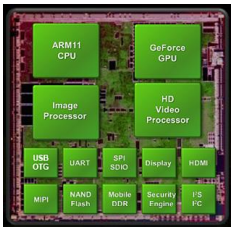
Integrated Circuit

Large Scale Integration

VLSI



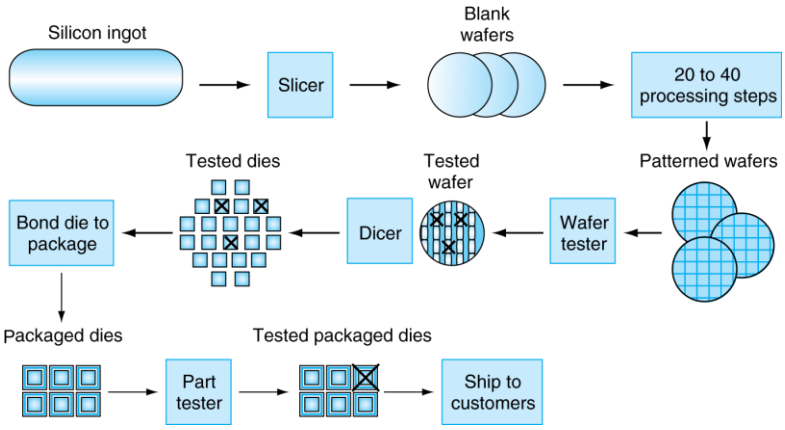
SoC



Semiconductor Technology

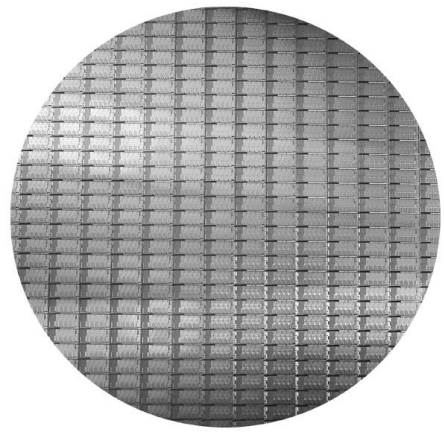
- Silicon: semiconductor
- Add materials to transform properties:
  - Conductors
  - Insulators
  - Switch

Manufacturing ICs



- Yield: proportion of working dies per wafer

Intel Core i7 Wafer



- 300mm wafer, 280 chips, 32nm technology
- Each chip is 20.7 x 10.5 mm

Integrated Circuit Cost

$$\text{Cost per die} = \frac{\text{Cost per wafer}}{\text{Dies per wafer} \times \text{Yield}}$$

$$\text{Dies per wafer} \approx \text{Wafer area}/\text{Die area}$$

$$\text{Yield} = \frac{1}{(1 + (\text{Defects per area} \times \text{Die area}/2))^2}$$

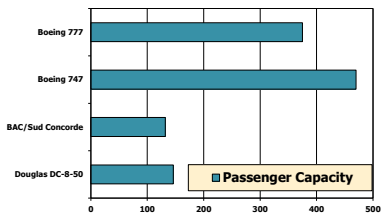
- Nonlinear relation to area and defect rate
  - Wafer cost and area are fixed
  - Defect rate determined by manufacturing process
  - Die area determined by architecture and circuit design

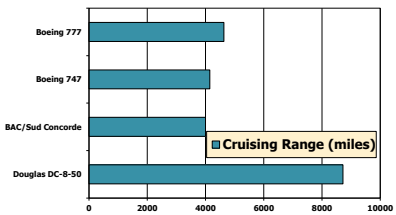
28 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 50

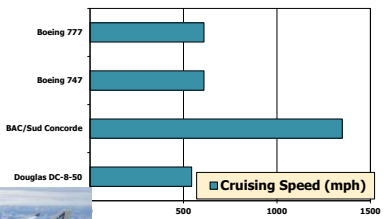
Defining Performance

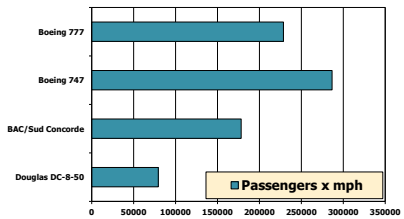
§1.6 Performance

- Which airplane has the best performance?









2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 51

Response Time and Throughput

- Response time
  - How long it takes to do a task
- Throughput
  - Total work done per unit time
    - e.g., tasks/transactions/… per hour
- How are response time and throughput affected by
  - Replacing the processor with a faster version?
  - Adding more processors?
- We'll focus on response time for now…

30 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 52

Relative Performance

- Define Performance = 1/Execution Time
- "X is n time faster than Y"

$$\text{Performance}_X / \text{Performance}_Y = \text{Execution time}_Y / \text{Execution time}_X = n$$

- Example: time taken to run a program
  - 10s on A, 15s on B
  - Execution Time_B / Execution Time_A = 15s / 10s = 1.5
  - So A is 1.5 times faster than B

31 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 53

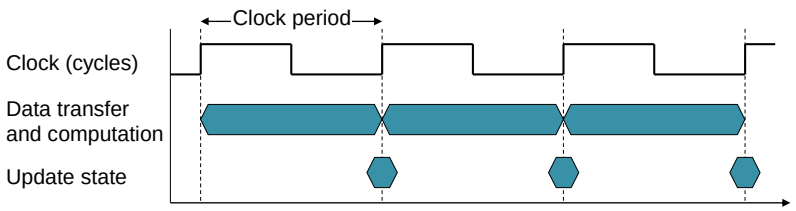
Measuring Execution Time

- Elapsed time
  - Total response time, including all aspects
    - Processing, I/O, OS overhead, idle time
  - Determines system performance
- CPU time
  - Time spent processing a given job
    - Discounts I/O time, other jobs' shares
  - Comprises user CPU time and system CPU time
  - Different programs are affected differently by CPU and system performance

32 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 54

CPU Clocking

- Operation of digital hardware governed by a constant-rate clock



- Clock period: duration of a clock cycle
  - e.g., 250ps = 0.25ns = 250×10^{-12}s
- Clock frequency (rate): cycles per second
  - e.g., 4.0GHz = 4000MHz = 4.0×10^{9}Hz

m	10^{-3}
u	10^{-6}
n	10^{-9}
p	10^{-12}

33 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 55

CPU Time

$$\text{CPU Time} = \text{CPU Clock Cycles} \times \text{Clock Cycle Time}$$
$$= \frac{\text{CPU Clock Cycles}}{\text{Clock Rate}}$$

- Performance improved by
  - Reducing number of clock cycles
  - Increasing clock rate
  - Hardware designer must often trade off clock rate against cycle count

34 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 56

CPU Time Example

- Computer A: 2GHz clock, 10s CPU time
- Designing Computer B
  - Aim for 6s CPU time
  - Can do faster clock, but requires 1.2 × clock cycles
- How fast must Computer B clock be?

$$\text{Clock Rate}_B = \frac{\text{Clock Cycles}_B}{\text{CPU Time}_B} = \frac{1.2 \times \text{Clock Cycles}_A}{6\text{s}}$$

$$\text{Clock Cycles}_A = \text{CPU Time}_A \times \text{Clock Rate}_A$$
$$= 10\text{s} \times 2\text{GHz} = 20 \times 10^9$$

$$\text{Clock Rate}_B = \frac{1.2 \times 20 \times 10^9}{6\text{s}} = \frac{24 \times 10^9}{6\text{s}} = 4\text{GHz}$$

35 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 57

Instruction Count and CPI

$$\text{Clock Cycles} = \text{Instruction Count} \times \text{Cycles per Instruction}$$

$$\text{CPU Time} = \text{Instruction Count} \times \text{CPI} \times \text{Clock Cycle Time} = \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$$

- Instruction Count for a program
  - Determined by program, ISA and compiler
- Average cycles per instruction
  - Determined by CPU hardware
  - If different instructions have different CPI
    - Average CPI affected by instruction mix

36 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 58

CPI Example

- Computer A: Cycle Time = 250ps, CPI = 2.0
- Computer B: Cycle Time = 500ps, CPI = 1.2
- Same ISA
- Which is faster, and by how much?

$$\text{CPU Time}_A = \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A = I \times 2.0 \times 250\text{ps} = I \times 500\text{ps}$$

A is faster…

$$\text{CPU Time}_B = \text{Instruction Count} \times \text{CPI}_B \times \text{Cycle Time}_B = I \times 1.2 \times 500\text{ps} = I \times 600\text{ps}$$

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

…by this much

36 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 59

CPI in More Detail

- If different instruction classes take different numbers of cycles

$$\text{Clock Cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{Instruction Count}_i)$$

- Weighted average CPI

$$\text{CPI} = \frac{\text{Clock Cycles}}{\text{Instruction Count}} = \sum_{i=1}^{n} \left(\text{CPI}_i \times \underbrace{\frac{\text{Instruction Count}_i}{\text{Instruction Count}}}_{\text{Relative frequency}} \right)$$

CPI Example

- Alternative compiled code sequences using instructions in classes A, B, C

Class	A	B	C
CPI for class	1	2	3
IC in sequence 1	2	1	2
IC in sequence 2	4	1	1

- Sequence 1: IC = 5
  - Clock Cycles
    = 2x1 + 1x2 + 2x3
    = 10
  - Avg. CPI = 10/5 = 2.0
- Sequence 2: IC = 6
  - Clock Cycles
    = 4x1 + 1x2 + 1x3
    = 9
  - Avg. CPI = 9/6 = 1.5

Performance Summary

The BIG Picture

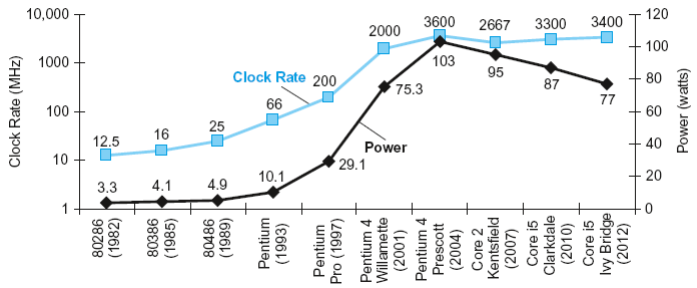
$$\text{CPU Time} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock cycle}}$$

- Performance depends on
  - Algorithm: affects IC, possibly CPI
  - Programming language: affects IC, CPI
  - Compiler: affects IC, CPI
  - Instruction set architecture: affects IC, CPI, T_c

38 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 62

Power Trends

§1.7 The Power Wall



- In CMOS IC technology

$$\text{Power} = \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency}$$

×30 5V → 1V ×1000

40 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 63

Reducing Power

- Suppose a new CPU has
  - 85% of capacitive load of old CPU
  - 15% voltage and 15% frequency reduction

$$\frac{P_{new}}{P_{old}} = \frac{C_{old} \times 0.85 \times (V_{old} \times 0.85)^2 \times F_{old} \times 0.85}{C_{old} \times V_{old}^{\ 2} \times F_{old}} = 0.85^4 = 0.52$$

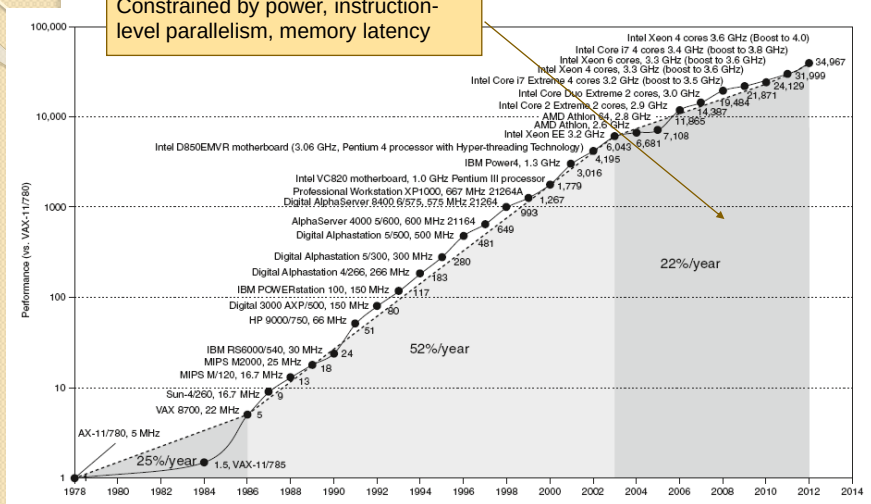
- The power wall
  - We can't reduce voltage further
  - We can't remove more heat
- How else can we improve performance?

41 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 64

Uniprocessor Performance

§1.8 The Sea Change: The Switch to Multiprocessors

Constrained by power, instruction-level parallelism, memory latency



44 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 65

Multiprocessors

- Multicore microprocessors
  - More than one processor per chip
- Requires explicitly parallel programming
  - Compare with instruction level parallelism
    - Hardware executes multiple instructions at once
    - Hidden from the programmer
  - Hard to do
    - Programming for performance
    - Load balancing
    - Optimizing communication and synchronization

SPEC CPU Benchmark

§1.9 Real Stuff"Benchmarking the Intel Core i7

- Programs used to measure performance
  - Supposedly typical of actual workload
- Standard Performance Evaluation Corp (SPEC)
  - Develops benchmarks for CPU, I/O, Web, …
- SPEC CPU2006
  - Elapsed time to execute a selection of programs
    - Negligible I/O, so focuses on CPU performance
  - Normalize relative to reference machine
  - Summarize as geometric mean of performance ratios
    - CINT2006 (integer) and CFP2006 (floating-point)

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

CINT2006 for Intel Core i7 920

Description	Name	Instruction Count x 10^9	CPI	Clock cycle time (seconds x 10^{-9})	Execution Time (seconds)	Reference Time (seconds)	SPECratio
Interpreted string processing	perl	2252	0.60	0.376	508	9770	19.2
Block-sorting compression	bzip2	2390	0.70	0.376	629	9650	15.4
GNU C compiler	gcc	794	1.20	0.376	358	8050	22.5
Combinatorial optimization	mcf	221	2.66	0.376	221	9120	41.2
Go game (AI)	go	1274	1.10	0.376	527	10490	19.9
Search gene sequence	hmmer	2616	0.60	0.376	590	9330	15.8
Chess game (AI)	sjeng	1948	0.80	0.376	586	12100	20.7
Quantum computer simulation	libquantum	659	0.44	0.376	109	20720	190.0
Video compression	h264avc	3793	0.50	0.376	713	22130	31.0
Discrete event simulation library	omnetpp	367	2.10	0.376	290	6250	21.5
Games/path finding	astar	1250	1.00	0.376	470	7020	14.9
XML parsing	xalancbmk	1045	0.70	0.376	275	6900	25.1
Geometric mean	–	–	–	–	–	–	25.7

High cache miss rates

47 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 68

SPEC Power Benchmark

- Power consumption of server at different workload levels
  - Performance: ssj_ops/sec
  - Power: Watts (Joules/sec)

$$\text{Overall ssj\_ops per Watt} = \left(\sum_{i=0}^{10} \text{ssj\_ops}_i\right) \Big/ \left(\sum_{i=0}^{10} \text{power}_i\right)$$

48 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 69

SPECpower_ssj2008 for Xeon X5650

Target Load %	Performance (ssj_ops)	Average Power (Watts)
100%	865,618	258
90%	786,688	242
80%	698,051	224
70%	607,826	204
60%	521,391	185
50%	436,757	170
40%	345,919	157
30%	262,071	146
20%	176,061	135
10%	86,784	121
0%	0	80
Overall Sum	4,787,166	1,922
Σssj_ops/Σpower =		2,490

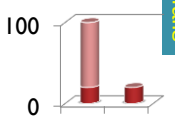
48 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 70

Pitfall (陷阱): Amdahl's Law

1.10 Fallacies and Pitfalls

- Improving an aspect of a computer and expecting a proportional improvement in overall performance

$$T_{improved} = \frac{T_{affected}}{\text{improvement factor}} + T_{unaffected}$$



- Example: multiply accounts for 80s/100s
  - How much improvement in multiply performance to get 5× overall?

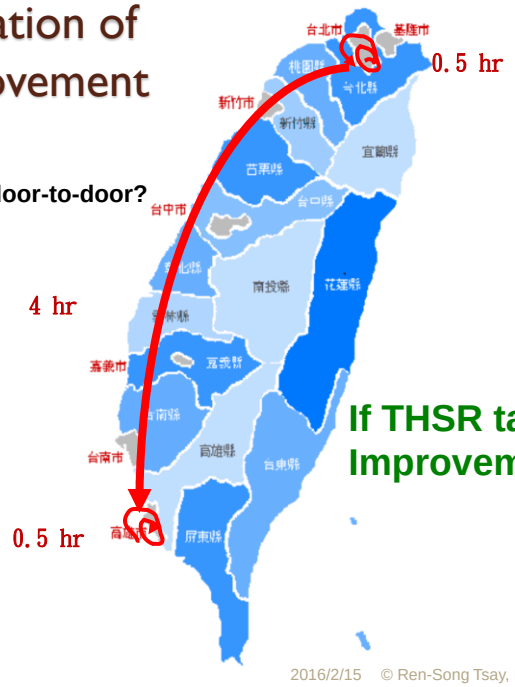
$$20 = \frac{80}{n} + 20$$

  - Can't be done!
- Corollary: make the common case fast

49 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 71

Limitation of improvement

Time for door-to-door?



If THSR takes 1 hour, Improvement?

Fallacy (迷思): Low Power at Idle

- Look back at i7 power benchmark
  - At 100% load: 258W
  - At 50% load: 170W (66%)
  - At 10% load: 121W (47%)
- Google data center
  - Mostly operates at 10% – 50% load
  - At 100% load less than 1% of the time
- Consider designing processors to make power proportional to load

Pitfall: MIPS as a Performance Metric

- MIPS: Millions of Instructions Per Second
  - Doesn't account for
    - Differences in ISAs between computers
    - Differences in complexity between instructions

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

- CPI varies between programs on a given CPU

51 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 74

Concluding Remarks

§1.11 Concluding Remarks

- Cost/performance is improving
  - Due to underlying technology development
- Hierarchical layers of abstraction
  - In both hardware and software
- Instruction set architecture
  - The hardware/software interface
- Execution time: the best performance measure
- Power is a limiting factor
  - Use parallelism to improve performance

52 2016/2/15 © Ren-Song Tsay, NTHU, Taiwan 75