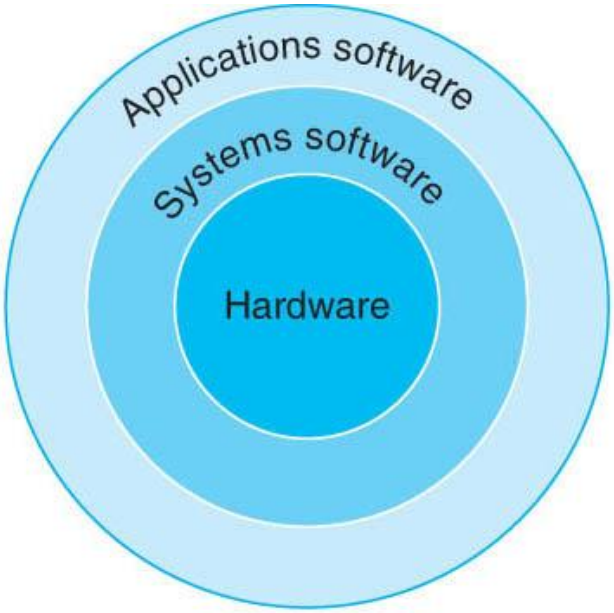
Computador: tecnologias e abstrações



História dos computadores

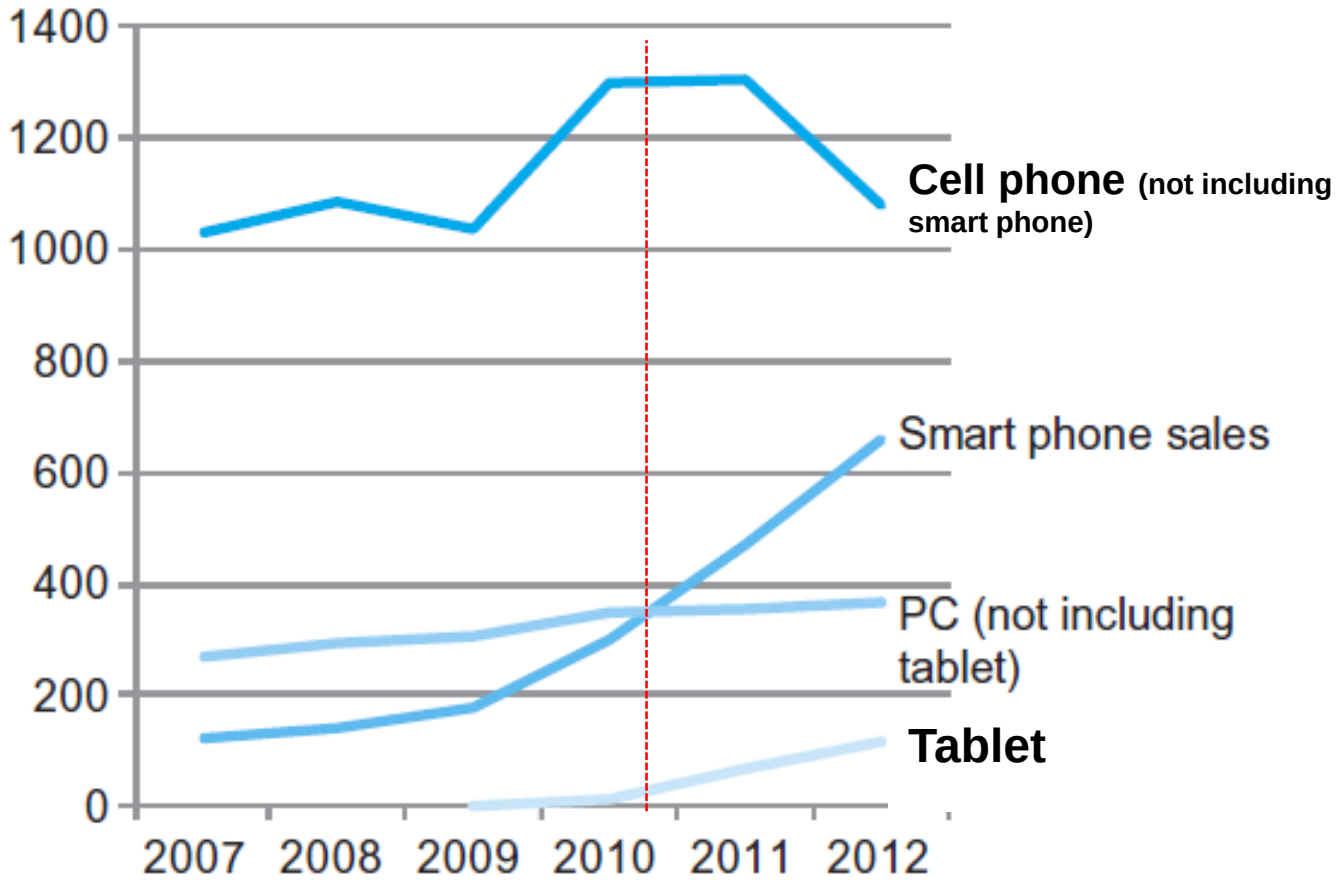
- **Pré-história:**
  - **ENIAC (1946), UNIVAC (1950)**
- **Era pré-PC (final de 1970, início 1980)**
  - **Mainframes, minicomputadores**
- **Era do PC (meados de 1980, meados 2000)**
  - **Computadores pessoais (desktops, notebooks)**
  - **Servidores**
- **Era pós-PC (final de 2000 em diante)**
  - *Personal mobile devices* (*PMDs*)
  - *Warehouse scale computers (WSC)*

Fonte: D. Patterson, *RISC versus CISC Wars in the PostPC Eras* (blog), 24/01/2011

A era Pós-PC

- ***Personal Mobile Device*** **(PMD)**
  - **Alimentado por bateria e conectado à Internet**
  - **Custa centenas de dólares**
  - **Aplicações: Web-based, media oriented**
    - » **Possivelmente "Third party software"**
  - **Exemplos: *Smart phones, tablets***
- ***Warehouse-Scale Computer*** **(WSC)**
  - **Custa milhões de dólares**
  - ***Software as a Service*** **(SaaS)**
    - » **Parte do software roda no PMD, parte em Cloud computing**
  - **Exemplos: Amazon and Google**

A era Pós-PC



Milhões

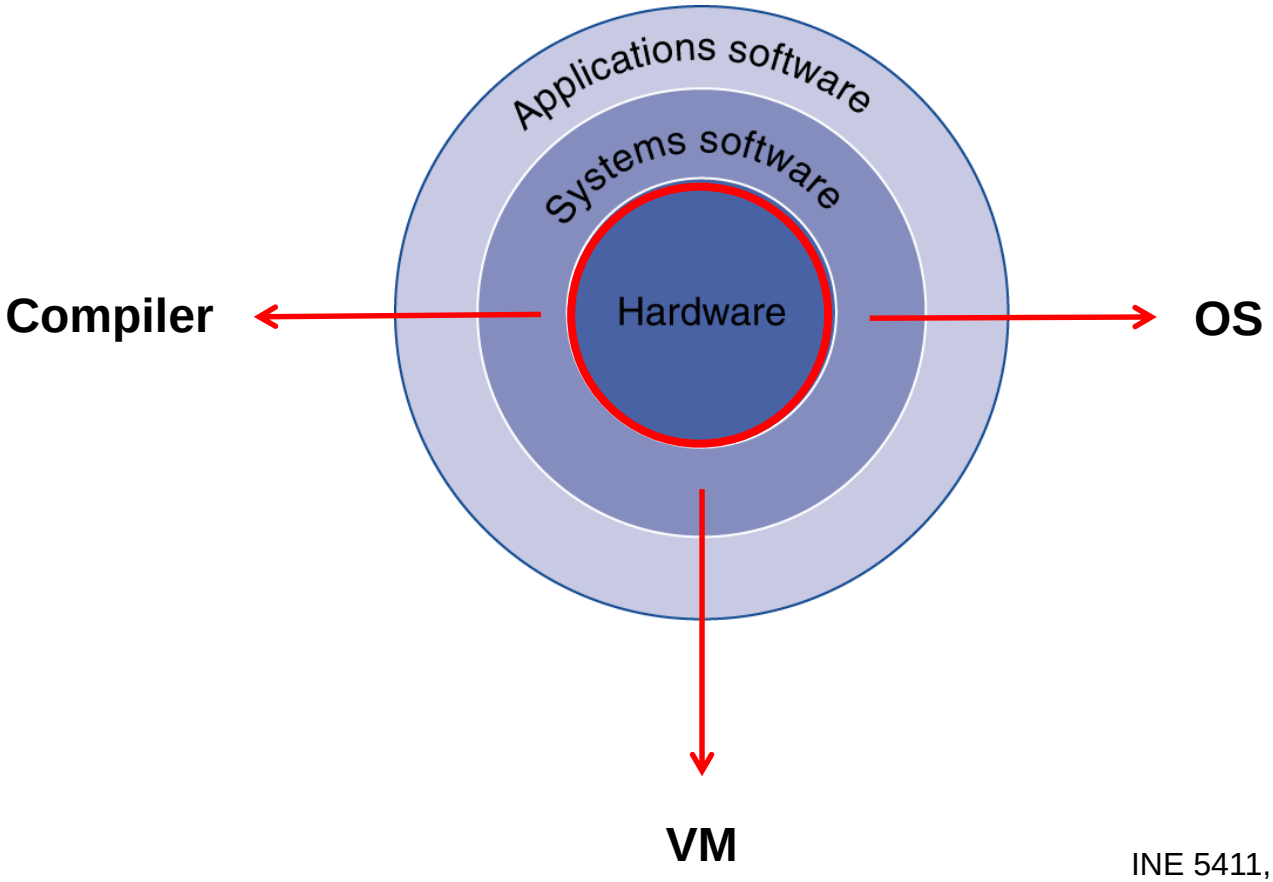
(Número de artefatos computacionais fabricados por ano)

Classes de Computadores

Class	Price		Examples
	System	µProcessor	
Personal Mobile Device (PMD)	$100 $1000	$10 $100	Cell phones Tablets
Desktop	$300 $2500	$50 $500	Netbooks, notebooks Workstations
Server	$5000 $10,000,000	$200 $2000	ATM machines, airline reservation systems
Clusters / Warehouse-scale computer (WSC)	$100,000 $200,000,000	$50 $250	Search, social network, vídeo viewing/sharing, multiplayer games online shopping
Embedded Computers / Internet of Things (IoT)	$10 $100,000	$0.01 $100	Appliances, automobiles, smart speakers, watches, cars, homes

Fonte: CAQA by Patterson & Hennessy, 6th edition, 2017

Por baixo de seu programa



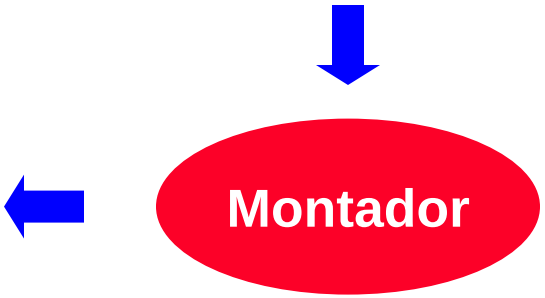
O Processo de Geração de Código

Linguagem de Alto Nível

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

Linguagem de montagem

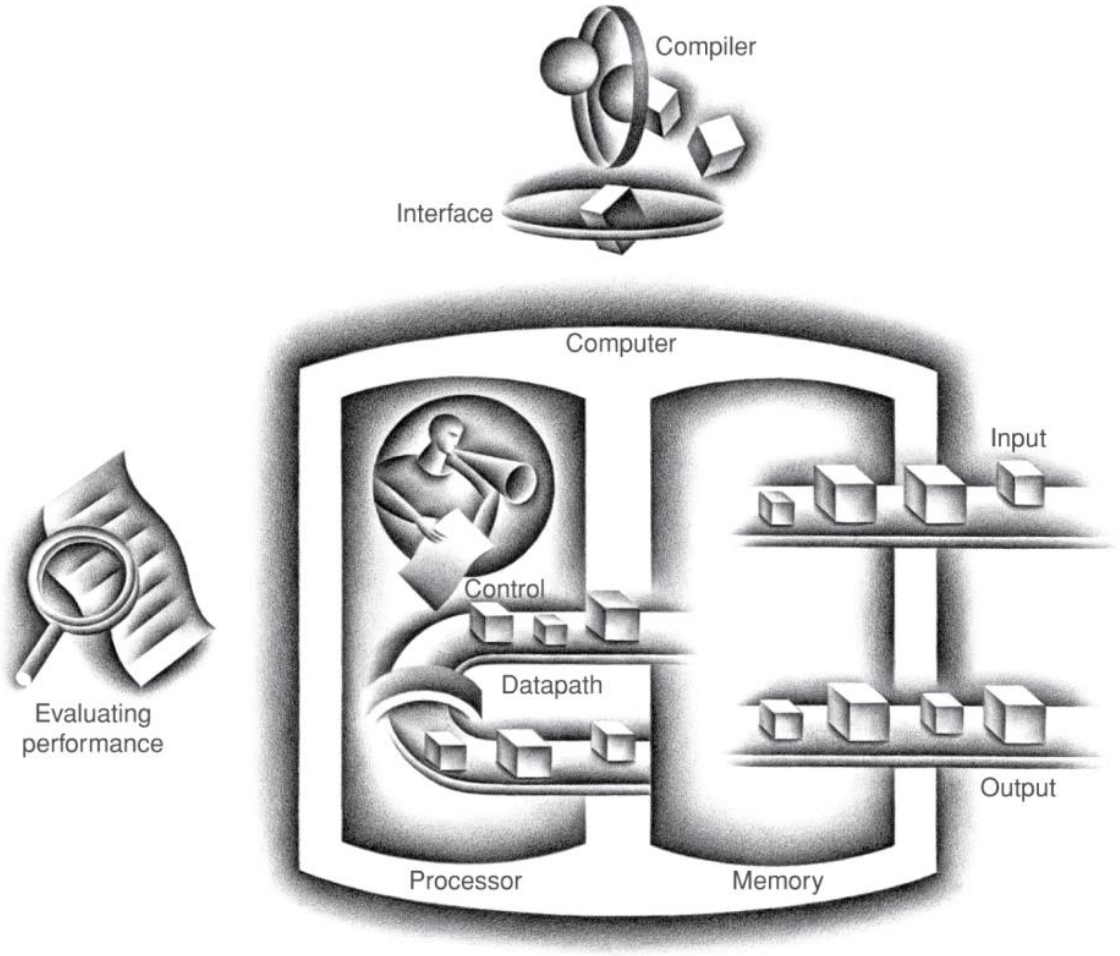
```
swap:   muli $2,$5, 4
        add $2,$4,$2
        lw $15, 0($2)
        lw $16, 4($2)
        sw $16, 0($2)
        sw $15, 4($2)
        jr $31
```




```
00000000101000010000000000011000
00000000100011100001100000100001
10001100011000100000000000000000
10001100111100000000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111100000000000000001000
```

Linguagem de máquina

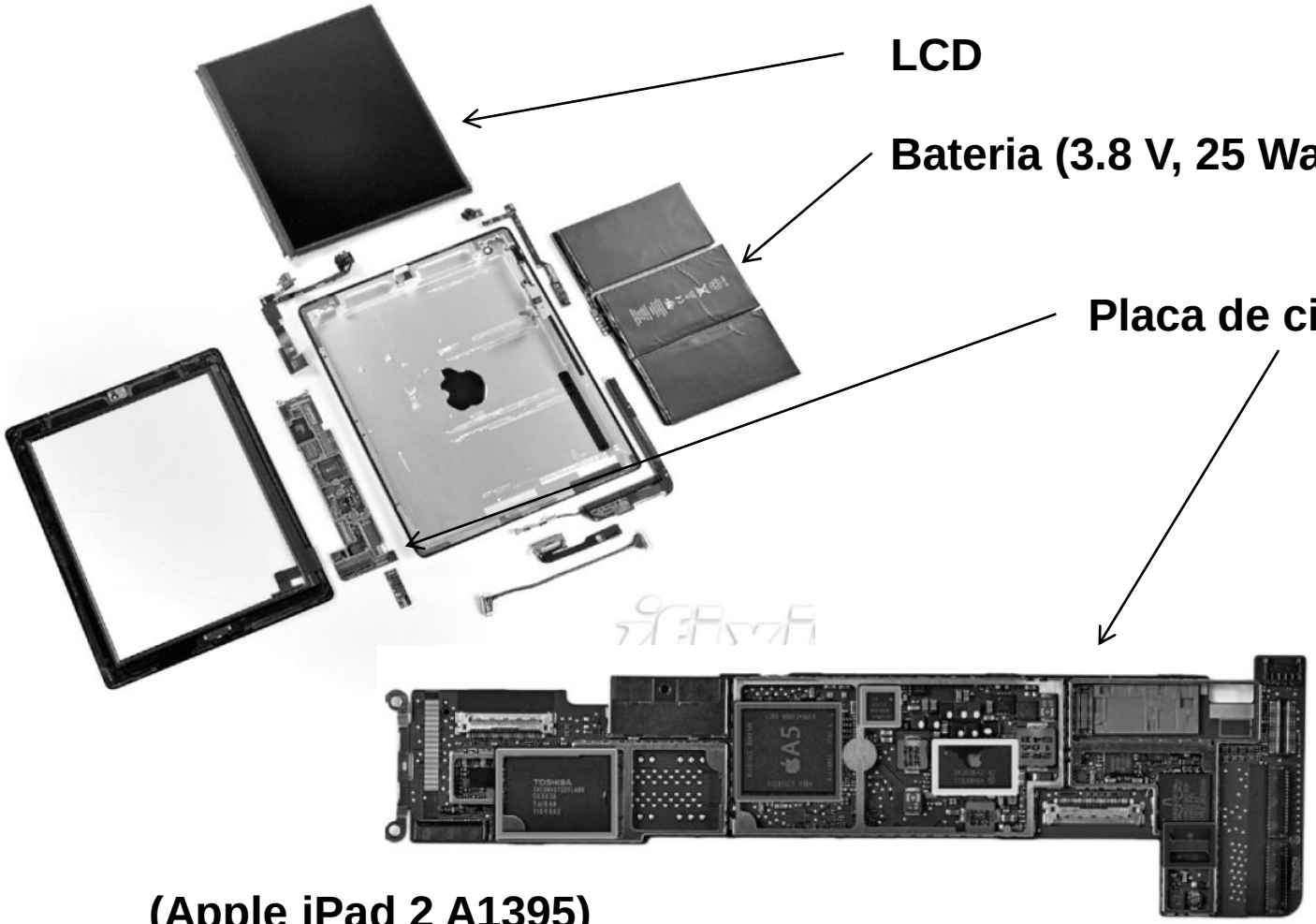
Componentes de um computador



Entrada/Saída inclui:

- **Interface com usuário**
  - » **Display, teclado, mouse**
- **Armazenamento**
  - » **Hard disk, CD/DVD, flash**
- **Adaptadores de rede**
  - » **Comunicação com outros computadores**

Abrindo a caixa



(Apple iPad 2 A1395)

Dentro do processador

- **Apple A5**
  - **(ARM Cortex A9)**

GPU

CPUs

Abstrações

- **Arquitetura**
  - **ISA: “Instruction-set architecture”**
  - **O que se precisa saber para rodar programa em linguagem binária**
  - **Utilidade: compatibilidade binária**
- **Implementação**
  - **HW que obedece à arquitetura**
  - **Utilidade: desempenho, eficiência energética**

Exemplos

ISA	Implementações
MIPS 32	BRCM 5000, MIPS 74K
MIPS 64	Godson-3
X86-32 (IA-32)	Pentium 4, Pentium M
IA-64	Itanium, Itanium2
X86-64	Core2 Duo, Core2 Quad,Opteron 2356 (Barcelona), Atom (e.g. Z3480)
ARMv7	Cortex-A8, Cortex-A9 (Apple A5), Cortex-A15
ARMv8	Cortex-A57 (Apple A7, AMD Opteron A1100)

Conclusões

Comum aos egressos de Computação e Engenharia

- **Para tradução de programas**
  - **Irrelevante como HW é implementado**
  - **Só ISA é relevante**
- **Para otimização de programas**
  - **Relevante como HW é implementado**
  - **Desempenho afetado pela implementação**
  - **Eficiência energética também**
- **Para manutenção ou projeto**
  - **Relevante como HW é construído**

O que vamos aprender?

- **Tradução de programas:**
  - **Como programas C/Java são traduzidos para a linguagem do HW**
- **Execução de programas:**
  - **Como o HW executa o programa traduzido**
- **Melhoria do desempenho de programas:**
  - **O que determina o desempenho de um programa**
  - **Como programador pode melhor o desempenho**

Desempenho de programas

Hardware or software component	How this component affects performance	Where is this topic covered?
Algorithm	Determines both the number of source-level statements and the number of I/O operations executed	Other books!
Programming language, compiler, and architecture	Determines the number of computer instructions for each source-level statement	Chapters 2 and 3
Processor and memory system	Determines how fast instructions can be executed	Chapters 4, 5, and 6
I/O system (hardware and operating system)	Determines how fast I/O operations may be executed	Chapters 4, 5, and 6

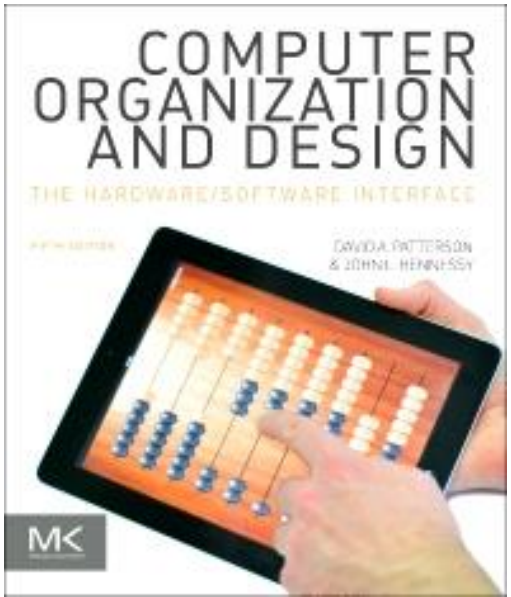
Como aumentar desempenho de um programa

- **Exemplo: multiplicação de matrizes**
  - **Data-level parallelism: x 3.8**
    - » **Sub-word parallelism via detalhes de C**
  - **Instrution-level parallelism: x 2.3**
    - » ***Loop unrolling para* emissão múltipla *e* execução *OoO***
  - **Otimização do uso da memória: x 2 a 2.5**
    - » ***Cache blocking***
  - **Thread-level parallelism: x 4 a 14**
    - » ***Parallel for in OpenMP***

Livro-texto

Computer Organization and Design: The Hardware/Software Interface, 5th edition

- **Autores: David Patterson and John Henessy**
- **MIPS edition (Não ARM edition! Nem RISC V edition!)**
  - **Exemplares na BU: *4th edition***



Computador: tecnologias e abstrações