CONCEITO DE COMPUTADOR E ORGANIZAÇÃO

TeSP de Aplicações Móveis

André Martins Pereira

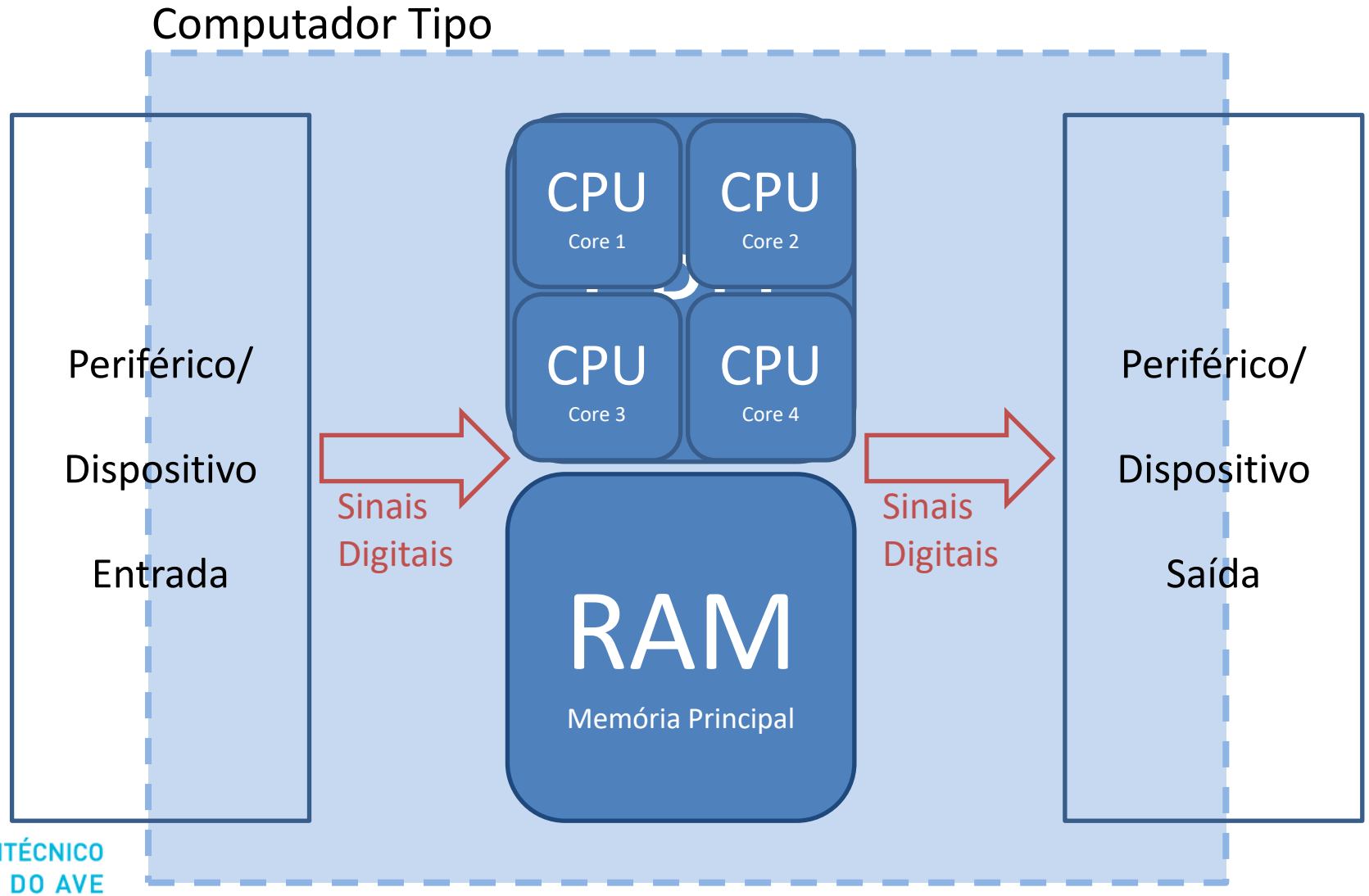
IPCA
INSTITUTO POLITÉCNICO
DO CÁVADO E DO AVE
ESCOLA SUPERIOR DE TECNOLOGIA

O QUE É UM COMPUTADOR?

- Sistema físico que:
  - Recebe informação, processa-a/arquiva-a, transmite-a e…
  - É programável!
    - A funcionalidade do sistema pode ser modificada, sem alterar fisicamente o sistema
  - Sistemas embebidos (p.e., *smartphones*, máq. fotográfica, …):
    - Funcionalidade sistema é fixada no fabrico
    - Reprogramar o sistema implica alterá-lo fisicamente
- Como se organiza um computador?
- Como se representa a informação num computador?
- Como se processa a informação num computador?

ORGANIZAÇÃO

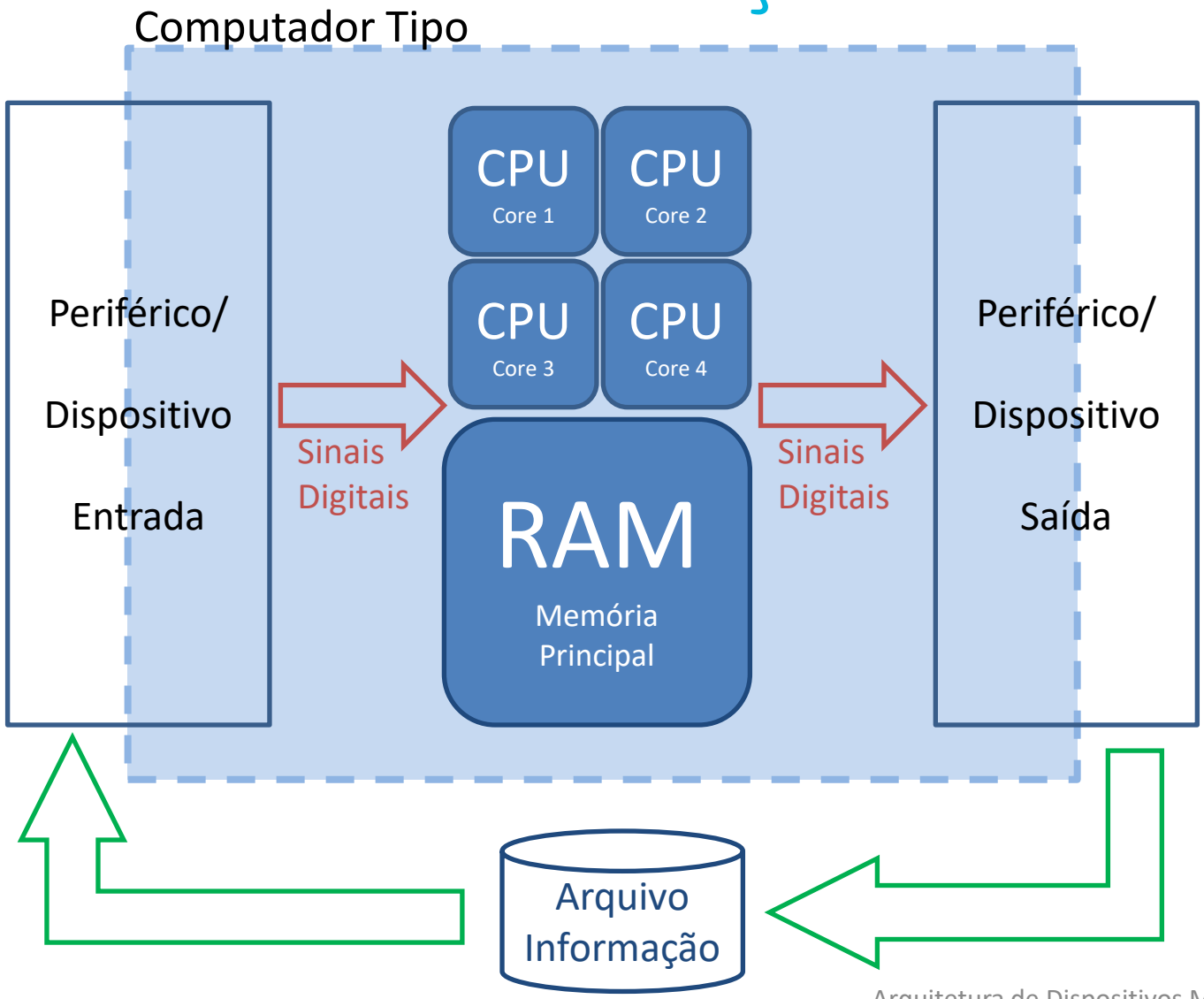
ORGANIZAÇÃO



ORGANIZAÇÃO

- Dispositivo de entrada:
  - Fornecem informação para as operações num programa de computador
  - Também conhecidos como "*input devices*"
  - Ex: Teclado, Rato
- Dispositivo de Saída
  - Exibem dados e informações processadas pelo computador
  - Também conhecidos como "*output devices*"
  - Ex: Monitor, impressora
- Fluxo de informação num programa
  - *Input*: instruções/código
  - CPU: armazena na RAM; corre instruções **uma a uma**; devolve o resultado
  - *Output*: Mostra/Imprime/(grava?) o resultado

ORGANIZAÇÃO



REPRESENTAÇÃO DE INFORMAÇÃO

REPRESENTAÇÃO DE INFORMAÇÃO

- Como se representa a informação num computador?
  - Com *bits* (*binary digits*)
- O que é possível representar?
  - números (para cálculo)
    - inteiros: S+M, Compl. p/ 1, Compl. p/ 2, Excesso
    - reais (*fp*): norma IEEE 754
  - textos (caracteres alfanuméricos)
    - Baudot, Braille, ASCII, Unicode, ...
  - conteúdos multimédia
    - imagens fixas: BMP, JPEG, GIF, PNG, . . .
    - audio-visuais: AVI, MPEG/MP3, ...
  - código para execução no computador

REPRESENTAÇÃO DE INFORMAÇÃO

- 1532.54_{10} (base 10)
  - $1*10^3 + 5*10^2 + 3*10^1 + 2*10^0 + 5*10^{-1} + 4*10^{-2} = 1532.54_{10}$
- 1532_6 (base 6)
  - $1*6^3 + 5*6^2 + 3*6^1 + 2*6^0 = 416_{10}$
- 1532_{13} (base 13)
  - $1*13^3 + 5*13^2 + 3*13^1 + 2*13^0 = 3083_{10}$
- 110110.01_{12} (base 2)
  - $1*2^5 + 1*2^4 + 0*2^3 + 1*2^2 + 1*2^1 + 0*2^0 + 0*2^{-1} + 1*2^{-2} + 1*2^{-3} = 54.375_{10}$

REPRESENTAÇÃO DE INFORMAÇÃO

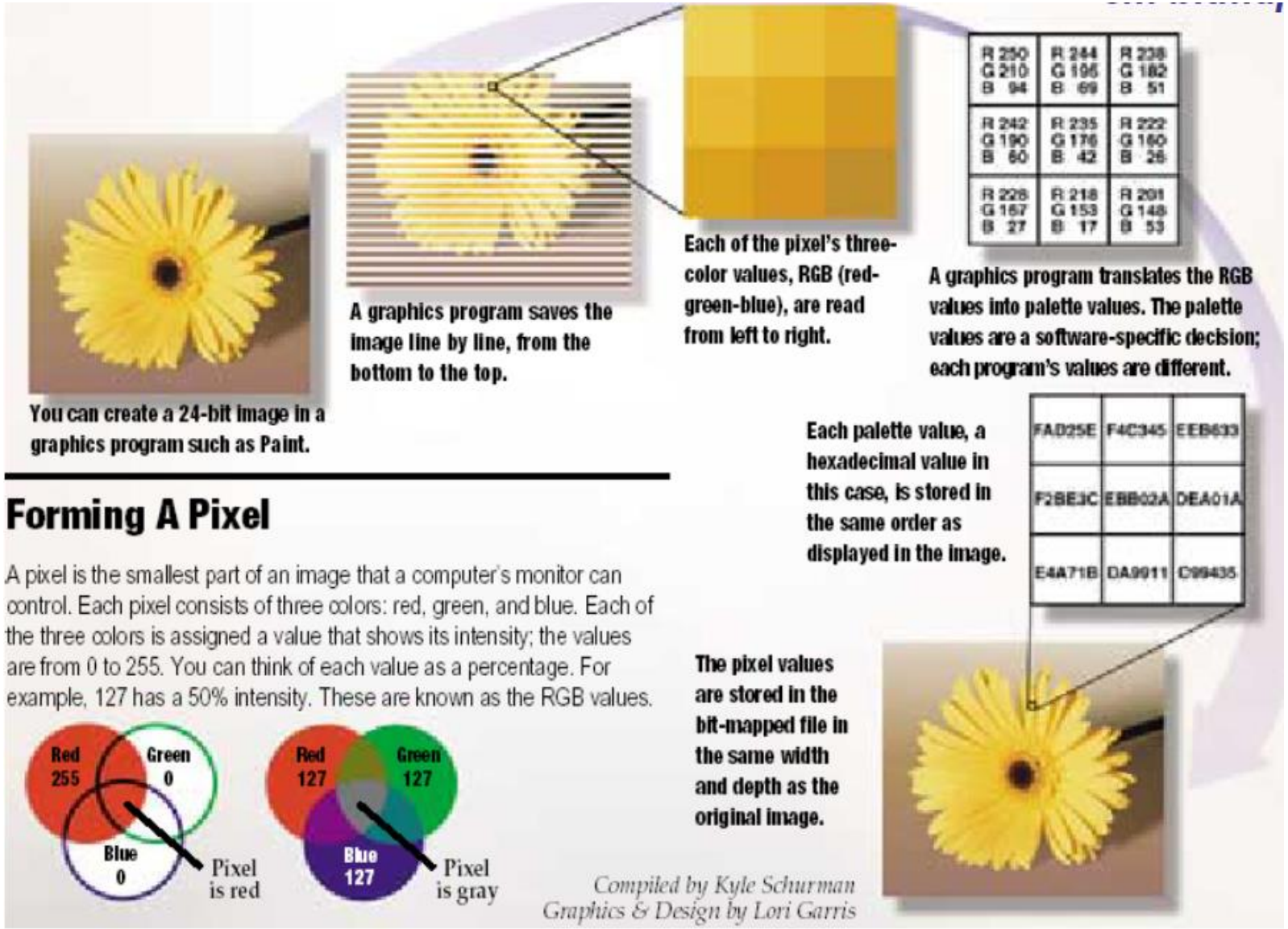
Tabela ASCII, 7 bits

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	SP	!	“	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

H	e	l	l	o		w	o	r	l	d	!
48	65	6c	6c	6f	20	77	6f	72	6c	64	21

REPRESENTAÇÃO DE INFORMAÇÃO

Imagem bitmap



REPRESENTAÇÃO DA INFORMAÇÃO

Código numa linguagem de alto nível (*C*)

- int x = x+y;
  - Soma à variável *x* o valor da variável *y*

Código numa linguagem próxima do processador

- addl 8(%ebp),%eax
  - x: em registo (*%eax*)
  - y: em memória (*%ebp + 8*)
  - Soma ao registo *%eax* o valor em memória guardado na posição *%ebp + 8*

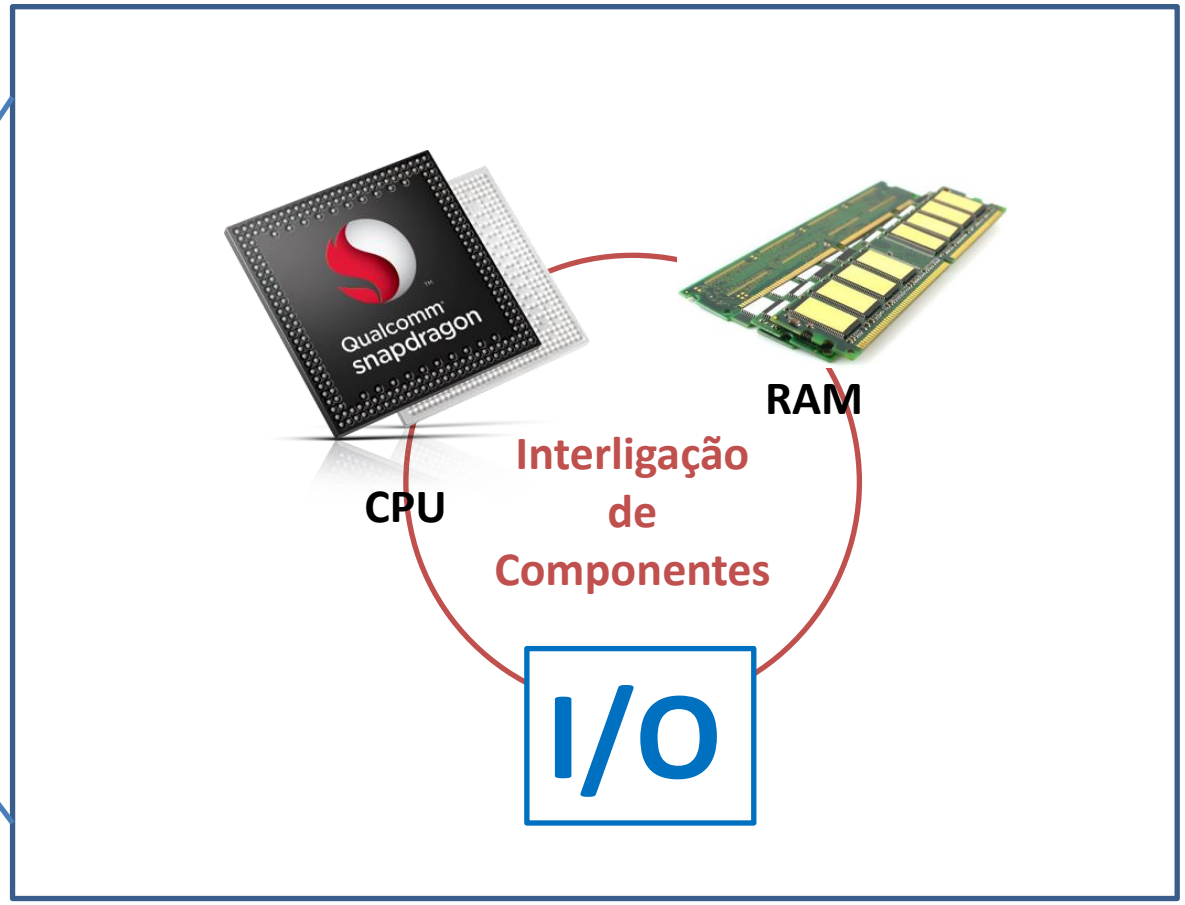
Código "objeto" (em hexadecimanl)

- 0x401046: 03 45 08
  - Instrução com 3 bytes
  - Em memória na posição *0x401046*

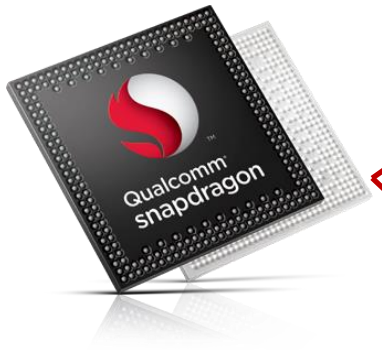
PROCESSAMENTO DE INFORMAÇÃO

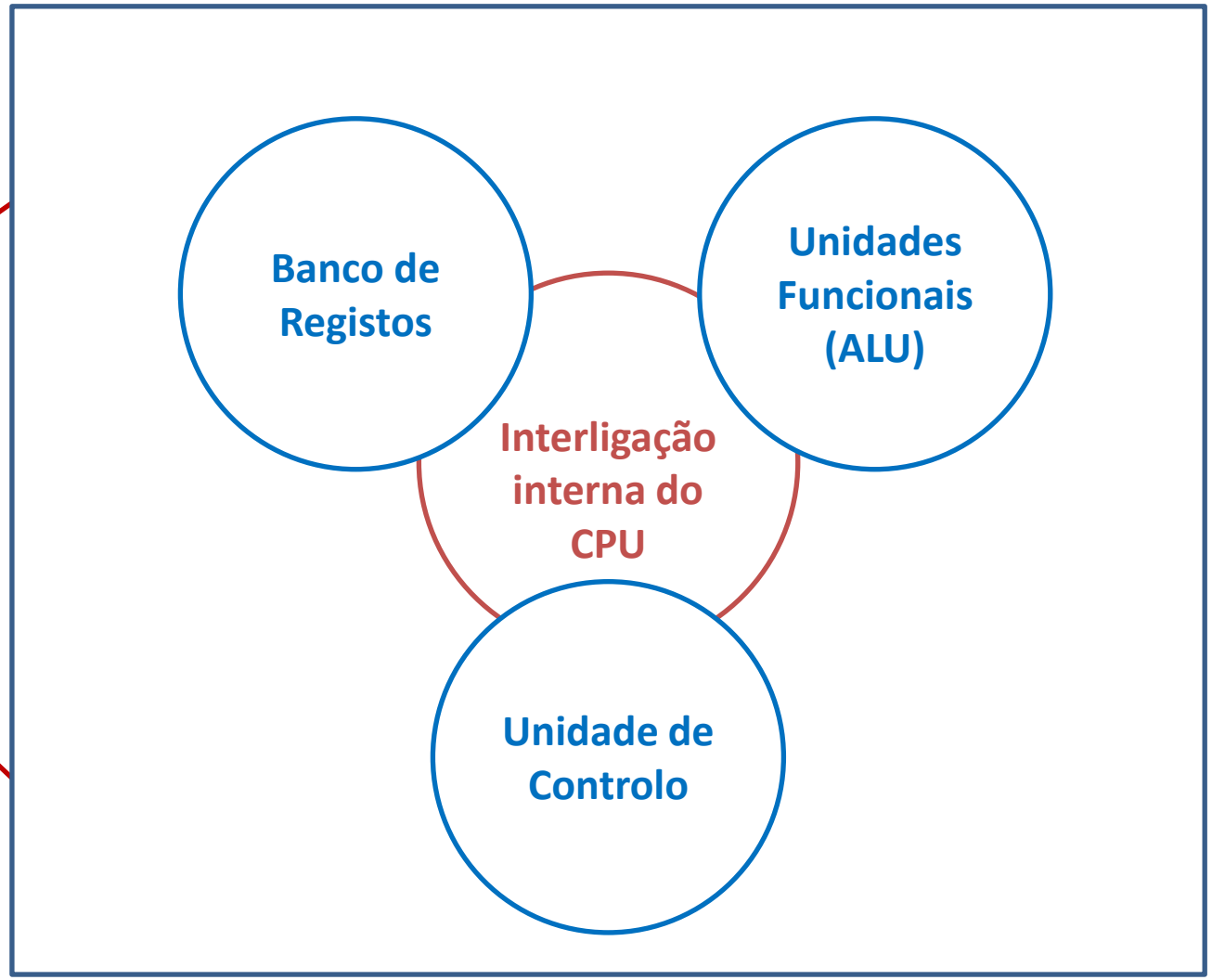
PROCESSAMENTO DE INFORMAÇÃO

Computador/ Dispositivo



PROCESSAMENTO DE INFORMAÇÃO





CPU

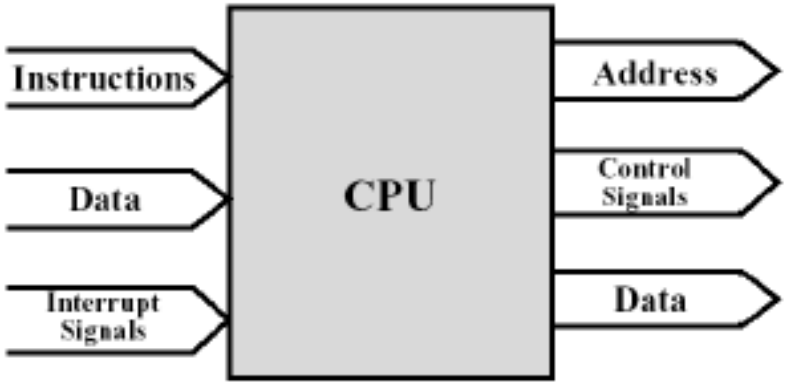
- Motor que continuamente:
  - **Lê da memória** interna um comando
  - **Interpreta**-o e
  - **Executa**-o
- De/para onde lê o comando?
  - Da posição de memória definida no registo apontador de instruções (**IP** – **I**nstruction **P**ointer; ou **PC** – **P**rogram **C**ounter)
  - Para o registo de instrução (**IR** – **I**nstruction **R**egister)
- Tipos básicos de comandos:
  - **Operações** com dados;
  - **Mover** dados de/para registos, memória ou I/O
  - **Decidir** qual o (local do) próximo comando a executar
- Onde estão estes registos?

CPU

- Unidade de Controlo:
  - Responsável por gerar todos os sinais que controlam as operações externas do CPU
  - Executa 3 ações básica intrínsecas e pré-programadas: procura (*fetch*), descodificação (*decode*) e execução (*execution*)
  - Exemplo de utilização:
    - **Procura** no registo IP qual a próxima instrução a ser executada, **descodifica**-a (transforma-a em binário) e **executa**-a
- Banco de Registos:
  - Contém todos os registos disponibilizados pelo processador
  - Depende da arquitetura…
  - IA32 tem 8 de 32 bits cada; ARM tem 31 de 64 bits cada
- Unidade aritmética e Lógica (ALU):
  - Circuito digital que executa operações de cálculo e lógica
  - Exemplo: somar dois valores; comparar conteúdo de 2 registos

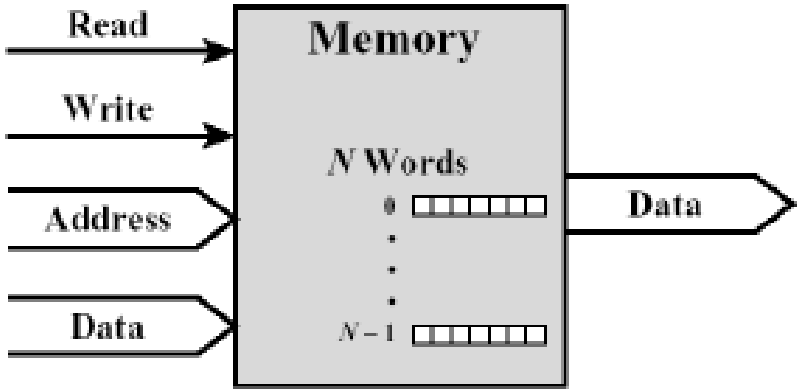
CPU

- Ligações do CPU:
  - Leitura de comandos/instruções (da memória)
  - Leitura/escrita de dados (da memória ou de I/O)
  - Envio de sinais de controlo para outros componentes
  - Receção de pedidos de interrupção (e reação)



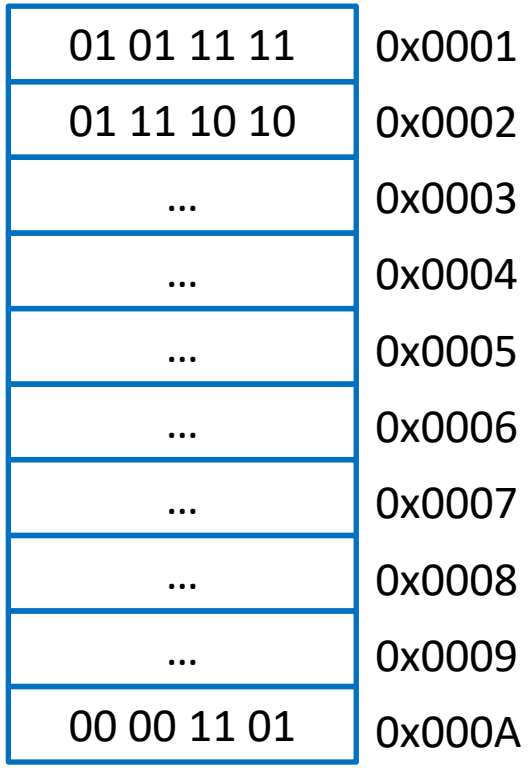
MEMÓRIA PRINCIPAL (RAM)

- Ligações da memória principal:
  - Recebe endereços (especificação de localizações)
  - Recebe sinais de controlo (read, write, timing, ...)
  - Recebe/envia dados



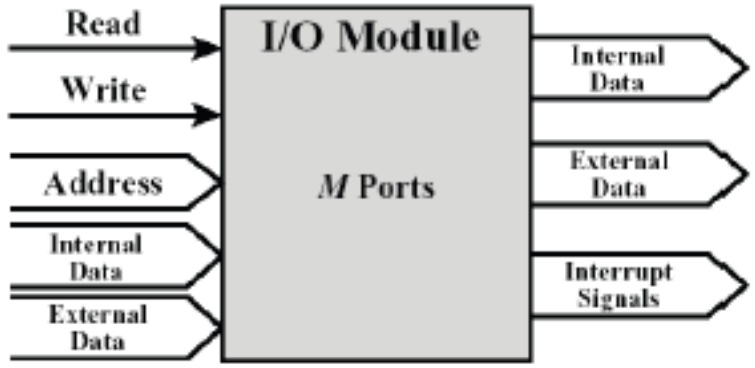
MEMÓRIA PRINCIPAL

- Componente de hardware usado para armazenamento temporário de **programas (e dados por ele manipulados)** durante a sua execução
- Organização lógica:
  - Vetor (*array* linear) de **células**, cada uma com **8 bits**
  - Cada célula tem um endereço (*address*) único
  - Dimensão máxima da memória definida pelos **n** bits do endereço
  - Arquitetura com **32 bits** para endereços podem ter 2^{32} posições de memória
    - 2^{32} bits = 4294967296 bits = 4 Gbit
    - 1 byte = 8 bits
    - Se cada posição de memória pode guardar 8 bits, então…
    - … temos 4 Gbit * 8 bits de memória = 4 GB



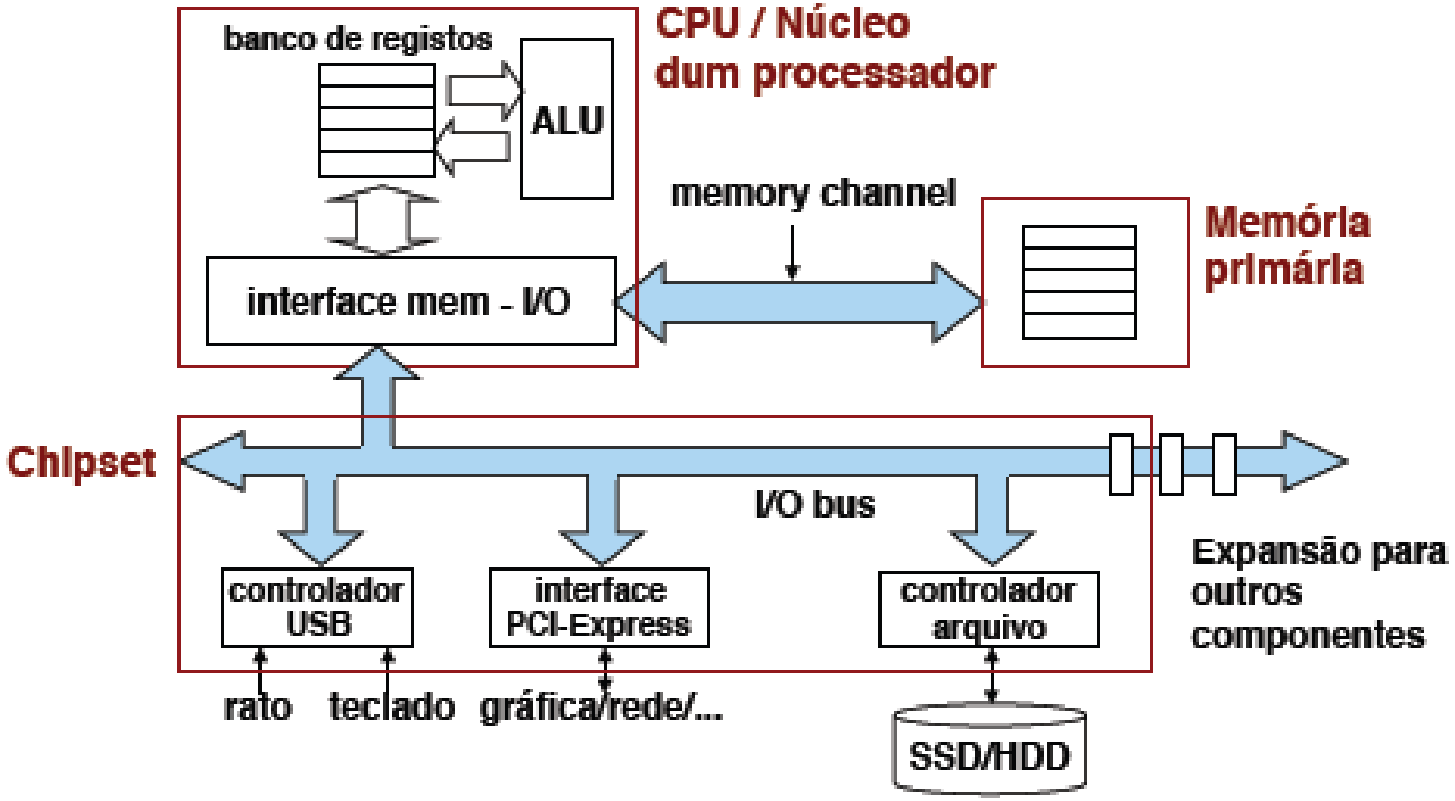
MÓDULOS DE I/O

- Ligações dos módulos de I/O:
  - Interface com CPU idêntico ao da memória
  - Dados internos incluem info de controlo e de estado (do periférico)
  - Dados externos incluem tb info de controlo e de estado
  - Sinais de *interrupt* para pedir a atenção do CPU



ESTRUTURA DETALHADA

- Todos estes componentes encaixam (e relacionam-se) da seguinte forma:



CPU E RAM

- Como comunicam?
  - Barramentos
- Existem 3 barramentos para comunicação CPU-RAM:
  - Barramento de dados (*data bus*) – responsável por transportar dados da memória para o processador, e vice-versa
  - Barramento de endereços (*address bus*) – transporta valores de posições de memória
  - Barramento de controlo (*control bus*) – transporta valores RD ou WD para indicar que os valores transportados serão de leitura ou escrita, respetivamente

CPU E RAM

- Estes 3 barramentos são usados extensivamente na execução de instruções
  - Nas 3 fases: (1) *fetch,* (2) *decode* e (3) *execution* -> a ver em breve
- Exemplos práticos:
  - Mover um valor ***X*** para uma posição de memória ***Y***:
    - *Data bus:* valor ***X***
    - *Address bus:* valor ***Y***
    - *Control bus:* ***WD***
  - Mover um valor ***X*** de uma posição de memória ***Y*** para um registo ***C***:
    - *Data bus:* valor ***X***
    - *Address bus:* valor ***Y***
    - *Control bus:* ***RD***

CONCEITO DE COMPUTADOR E ORGANIZAÇÃO