António José Estêvão Cabrita

1002404

**TI - RESUMOS**

Table of Contents

[1. Introdução aos Computadores 2](#_Toc296903754)

[1.1 Parte 1 - Conceitos básicos das TIC 2](#_Toc296903755)

[1.1.1 Informação 2](#_Toc296903756)

[1.1.2 Números 2](#_Toc296903757)

[1.1.3 Representação binária 3](#_Toc296903758)

[1.1.4 Digital vs. Analógico 4](#_Toc296903759)

[1.1.5 Hardware e software 4](#_Toc296903760)

[1.1.6 . Definições de informação 5](#_Toc296903761)

[1.1.7 Breve História das Tecnologias de Informação e Comunicação 5](#_Toc296903762)

[2 Parte 2 - Estrutura e Funcionamento dos Computadores 6](#_Toc296903763)

[2.1 Organização de um computador 6](#_Toc296903764)

[2.2 Unidade Central de Processamento 6](#_Toc296903765)

[2.3 O “fetch cycle” 7](#_Toc296903766)

[2.4 As instruções do processador 7](#_Toc296903767)

[2.5 A memória 8](#_Toc296903768)

[2.6 Memória secundária 9](#_Toc296903769)

[2.7 Dispositivos de entrada e saída 9](#_Toc296903770)

[2.8 Conflitos de hardware 10](#_Toc296903771)

[3 - Sistemas Operativos 11](#_Toc296903772)

[3.1 A “shell” 11](#_Toc296903773)

[3.2 Tipos de SO 11](#_Toc296903774)

[3.3 Ficheiros 12](#_Toc296903775)

[3.4 Processos 13](#_Toc296903776)

[3.5 Gestão de memória 14](#_Toc296903777)

[3.6 Gestão de dispositivos 14](#_Toc296903778)

[3.7 “Boot strapping” 15](#_Toc296903779)

[Índice Remissivo 16](#_Toc296903780)

# Introdução aos Computadores

## Parte 1 - Conceitos básicos das TIC

### Informação

Os tipos de informação com que interagimos mais frequentemente nos sistemas informáticos são os seguintes:

• **Texto**: uma sequência de caracteres (símbolos) que codificam um sistema de escrita.

• **Imagem**: um padrão bidimensional de luz e cor, que reflecte o que a nossa visão capta. Esse padrão pode ser estático, quando não se altera com o tempo, ou dinâmico, quando depende do passar do tempo – neste caso designa-se vídeo.

• **Som**: um padrão de vibração do ar que reflecte o que é captado pelo nosso sentido da audição.

Destes três tipos de informação, o texto, é o mais fácil de representar num computador. Basta atribuir um código numérico a cada símbolo e representar o texto através da sequência de códigos correspondente à sequência de caracteres do texto.

A luz é representada por um número que identifica a intensidade luminosa. O tom de cor pode ser representado pelas percentagens relativas de três cores básicas (vermelho, verde e azul nos monitores; cyan, magenta e amarelo nas impressoras). A disposição espacial da luz e da cor é representada por um sistema de coordenadas cartesianas. Para representar o vídeo, há que adicionar uma terceira coordenada ao sistema: o tempo.

O tempo também é um parâmetro na representação do som, além dos parâmetros típicos das ondas sonoras, como amplitudes, frequências e comprimentos de onda.

Representamos assim uma grande variedade de informação através de números.

### Números

O matemático Georg Cantor demonstrou que existem muito mais números reais do que números inteiros. Este facto reflecte-se nos tipos de infinitos presentes nos números reais: o infinitamente grande e o infinitamente pequeno. Para efeitos de representação, o infinitamente grande é limitado pela **amplitude**, ou seja, o maior número real. O infinitamente pequeno traduz-se na **precisão**, ou seja, no número de casas decimais.

Os números reais podem ser representados no computador de duas formas distintas: a representação em **vírgula fixa** e em **vírgula flutuante**. A representação em vírgula fixa é a mais natural: a parte inteira do número é representada por um grupo de dígitos de dimensão fixa (tal como nos números inteiros) e a parte decimal também.

A representação quase universalmente usada é em vírgula flutuante. Consideram-se apenas os dígitos significativos (eliminam-se os zeros à esquerda e à direita), e guarda-se a posição da vírgula decimal. A sequência de dígitos significativos chama-se **mantissa** e a posição da vírgula é o **expoente**.

A representação em vírgula flutuante tem a vantagem de não desperdiçar dígitos que representam zero o que aumenta a gama de valores que podem ser representados com os mesmos dígitos. A amplitude da representação em vírgula flutuante corresponde ao número de dígitos usados no expoente e a precisão corresponde ao número de dígitos da mantissa.

O grande inconveniente dos números em vírgula flutuante é a complexidade das operações e a imprecisão dos resultados.

### Representação binária

Os números inteiros e reais são a base da representação da informação nos sistemas informáticos. Os computadores utilizam apenas dois dígitos (representação binária). O número de dígitos de um sistema de numeração designa-se por **base**.

Cada dígito de uma representação binária é denominado **bit** (binary digit). Se dispusermos de N bits, é possível representar 2x2x…x2 = 2N números diferentes. Um grupo de 8 bits é designado por **byte**. Enquanto que um quilograma são 1000 gramas, um quilobyte são 1024 bytes (210). A razão desta diferença é que se torna muito mais conveniente usar uma potência.

No sistema binário temos a casa das unidades, dos 2, dos 4, dos 8, etc. São as sucessivas potências de 2 (1 = 20, 2 = 21) e correspondem ao “peso” de cada dígito binário. Se queremos determinar quanto vale o número binário 10011011, atribuamos a cada dígito o seu peso:



Multiplica-se o valor de cada dígito pelo seu peso e soma-se tudo. Assim, para o exemplo acima, o resultado é: **128 + 16 + 8 + 2 + 1 = 155.** O número binário 10011011 vale, então, 155.

A operação inversa, a **conversão de um número decimal para binário** é mais complicada: exige uma série de divisões sucessivas por 2. Como exemplo, para se converter o número 232 em binário temos de dividir o número por 2 e olhar para o resto da divisão.

O processo termina quando chegamos a um quociente 0. O número binário que pretendemos é, finalmente, 11101010, a sequência, por ordem inversa, dos restos obtidos.

Para além das bases 2 e 10, são importantes outras bases de numeração como 8 e 16, pela facilidade de conversão de e para a base 2. Para converter um número na base 2 para uma base 2N basta agrupar cada N dígitos do número binário e substituí-los pelo correspondente dígito em base 2N. Por serem muito usadas, estas bases recebem nomes especiais – a **octal** e a **hexadecimal**.

###  Digital vs. Analógico

A informação representada desta forma diz-se **digital**, já que é constituída por uma série de dígitos binários. Os números reais têm uma característica que os distingue dos números inteiros: é impossível representar digitalmente, com exactidão, a maioria dos números. Este facto tem a ver com a natureza infinita dos números reais, tal como foi descoberta por Cantor: amplitude infinita e precisão infinita.

As grandezas físicas medem-se recorrendo a números reais. A informação veiculada por números inteiros é **discreta**, podendo ser representada de forma digital, enquanto que a informação veiculada por números reais é **contínua**, podendo ser representada apenas aproximadamente em forma digital. Um dispositivo que representa um número real de forma contínua diz-se **analógico**.

Ao processo de conversão de informação analógica em informação digital chama-se **digitalização** ou **discretização**.

###  Hardware e software

O hardware são os equipamentos físicos e o software são programas que correm sobre o hardware. O software é, de facto, a informação digital manipulada por e para os dispositivos.

São conceitos distintos por serem conceptualmente independentes. Em princípio:

a) Um programa com determinada funcionalidade pode correr em qualquer computador;

b) Um computador pode correr qualquer tipo de programa;

Os computadores cumprem o requisito b), por serem máquinas de carácter genérico com capacidade para executar qualquer processo automático (algoritmo).

As linguagens de programação respondem ao outro requisito, permitem codificar um algoritmo independentemente da máquina onde vai ser executado. Com a expansão da Internet, foi desenvolvida a linguagem Java que permite a execução de um programa em várias máquinas, interpondo uma “máquina virtual” entre o software e o hardware. A máquina virtual Java é também um programa mas “esconde” os detalhes do hardware ao programador.

### . Definições de informação

A informação contida numa imagem em branco é quase nula. Na representação em computador esta imagem irá ocupar o mesmo espaço que uma paisagem. Há a sensação que a imagem está a desperdiçar espaço por não conter qualquer informação. É nesta ideia que se baseiam as definições de informação propostas por **Claude Shannon** e mais tarde por **Chaitin e Kolmogorov**. A teoria da informação de **Shannon** permite substituir qualquer sequência de dígitos por outra, mais reduzida.

A teoria da informação algorítmica, desenvolvida por **Chaitin** e por **Kolmogorov**, diz que a informação contida numa sequência de dígitos é igual ao tamanho do menor programa de computador que consegue gerar essa sequência. O problema com esta definição é que não é possível construir automaticamente esse programa. A definição de **Chaitin** é interessante do ponto de vista matemático, já que propõe uma medida absoluta de informação, enquanto que a definição de **Shannon** mede apenas a informação média.

###  Breve História das Tecnologias de Informação e Comunicação

Em 1931 Kurt **Gödel** demonstrou que é impossível formalizar qualquer teoria que envolva conjuntos infinitos.

**Alan Turing** reformulou o teorema de **Gödel** como um dispositivo teórico a que chamou “**máquina de Turing**”. A ideia revolucionária de **Turing** foi codificar o programa sob a forma de zeros e uns na fita. Esta **máquina universal de Turing** constitui o modelo teórico dos computadores que usamos actualmente. **Turing** traduziu o resultado de **Gödel** em linguagem computacional da seguinte forma: não existe nenhum processo automático capaz de determinar se um programa da máquina universal de Turing termina ou não a sua execução.

As ideias de **Turing** foram desenvolvidas por **John Von Neumann** que propôs a arquitectura em que os programas e os dados são guardados na mesma memória.

Os primeiros computadores baseados neste princípio, fabricados nos anos 40, usavam tecnologia de válvulas electromecânicas. A invenção do **transístor** deu origem à **2ª geração** de computadores. A 3ª geração introduziu o **circuito integrado**: um componente que incorpora um circuito electrónico numa pastilha de silício.

As tecnologias que mais contribuíram para a expansão das TIC foram os sistemas operativos de janelas e a Internet. Em meados dos anos 80 surgiram os primeiros serviços de Internet e a World Wide Web foi inventada como forma de partilhar documentos hiperligados.

Estes desenvolvimentos de software não seriam possíveis sem o correspondente avanço do hardware, embora a ideia básica se mantenha desde os anos 40.

# Parte 2 - Estrutura e Funcionamento dos Computadores

## Organização de um computador

Os computadores são constituídos por três grandes subsistemas: a **unidade central de processamento** (CPU), a **memória** e os **dispositivos de entrada e saída** (I/O).

A fig. mostra os três subsistemas e os canais de transferência de informação (***bus)***.



**Figura 1 - Arquitectura do computador**

O CPU está encarregue da gestão da memória e do subsistema de I/O. Por vezes há interesse em que o subsistema de I/O aceda à memória sem passar pelo CPU. Criaram-se então canais privilegiados chamados **DMA** que permite libertar o CPU para tarefas mais complexas.

O CPU e a memória encontram-se na “**motherboard**”, uma placa de circuitos integrados que é o elemento central do computador, que permite a substituição do CPU e das memórias através da inserção de placas em suportes que se ligam a um *bus* de acesso ao CPU. As placas de expansão mais comuns são a **placa gráfica** e a **placa de som**.

## Unidade Central de Processamento

O processador, o motor de um computador, é uma máquina de estados, que executa instruções a um determinado ritmo, que em cada momento são determinados por um conjunto de registos internos. Em cada passo, o processador lê a próxima instrução, executa-a, modifica os seus registos e passa para o estado seguinte.

O ritmo a que o processador executa instruções é determinado pela sua **velocidade**, que se mede em ciclos por segundo (Hz) porque um ciclo tem sempre uma duração fixa enquanto que as instruções de um processador têm durações de execução diferentes, que é imposta por um cristal que emite impulsos eléctricos a um ritmo constante – o mecanismo de geração desses impulsos chama-se o **relógio** (clock). Às vezes é possível aumentar a velocidade do processador aumentando o ritmo do relógio (processo chamado *overclocking)*.

## O “fetch cycle”

O processador tem um modo de funcionamento muito simples e que pode ser ilustrado pela seguinte “receita”:

Passo 1: ler a próxima instrução

Passo 2: descodificar a instrução lida

Passo 3: executar a instrução

Passo 4: voltar ao passo 1

Este procedimento chama-se “fetch cycle” e é tudo o que o processador faz. É simples, se tivermos em conta que apenas existem duas ou três centenas de instruções diferentes.

Toda a diversidade de funções que um computador exibe advém da sequência específica de instruções que é fornecida ao computador a que chamamos **programa**.

## As instruções do processador

Um **programa** de computador é constituído por uma sequência de instruções. Um programa é visto por um utilizador como uma caixa fechada onde entram dados e saem os resultados pretendidos pela outra extremidade, como se ilustra na figura 2:



**Figura 2 - O programa**

Não é difícil compreender o tipo de instruções que um processador executa:

- **instruções de transferência de informação**: servem para copiar e mover dados de um lado para o outro.

- **instruções de transformação de informação**: são de quatro tipos:

* **Operações aritméticas**: efectuam cálculos numéricos sobre os dados
* **Operações lógicas**: aplicam operadores da lógica aos dados
* **Deslocação e rotação**: alteram a sequência dos bits de um registo ou posição de memória.
* **Comparações**: efectuam comparações quantitativas sobre os dados

- **instruções condicionais**: conferem capacidade de decisão aos programas permitindo executar uma instrução A se uma determinada condição for verdadeira ou uma instrução B caso contrário. As condições são normalmente baseadas no resultado de uma instrução de comparação.

- **instruções de salto e chamada**: permitem alterar a ordem de execução das instruções saltando para outro ponto do programa. São muitas vezes associadas às instruções condicionais, quando é preciso executar um grupo de várias instruções sob determinada condição.

- **outras instruções**: entrar em modos especiais de funcionamento do processador, parar o processador, não fazer nada durante algum tempo, etc.

A seguinte sequência exemplifica um programa que pode ser executado num CPU:

* *copiar* conteúdo da posição 2412 da memória para o registo A
* *somar* 6 ao registo A
* *comparar* o conteúdo do registo A com o número 10
* *se* for menor, *saltar* para a posição <reprovado> do programa
* *transferir* a frase “aprovado” para o ecrã
* - <reprovado>: transferir a frase “reprovado” para o écrã

Os dados entram na posição2412 da memória e os resultados saem pelo écrã.

## A memória

O termo “posição” para indicar o local onde se situam tanto dados como instruções do programa não é acidental, mas reflecte o duplo papel da memória num sistema informático: o de armazenar dados e programas.

No início da história dos computadores, os dados e os programas eram guardados em memórias diferentes. A memória comum para guardar dados e programas constituiu uma mudança muito importante na forma de organizar a informação no computador. A esta organização chama-se arquitectura de **Von Neumann**, e os computadores actuais baseiam-se nela.

A memória de um computador é uma sequência de “compartimentos” identificados pela sua “posição” ou **endereço** que contêm informação (ver figura 3). Cada compartimento contém uma quantidade fixa de bits (8, 16, 32 ou 64, conforme o computador). A capacidade da memória mede-se em bytes ou nos seu múltiplos.



**Figura 3 - A memória**

A comunicação entre o processador e a memória é sempre feita nos seguintes termos: “dá-me os dados que estão no endereço x” ou “coloca os dados y no endereço z”. A informação que circula entre o processador e a memória é, portanto, de três tipos:

* endereços
* dados
* informação de controlo

A informação de controlo corresponde ao tipo de operação que o processador quer executar na memória: ler dados ou escrever dados. Os três tipos de informação são transmitidos em simultâneo, pelo que o bus que liga o processador à memória está dividido em três partes: o **bus de endereços**, o **bus de dados** e o **bus de controlo**.

* $bus endereços \left(bits\right)=Log\_{2}2^{y} = y Log\_{2}2=y x 1=y$
* $2^{y}(bits)=\frac{Memória total}{arquitectura}$
* **Bus dados = “Arquitectura” = 8, 16, 32 ou 64 bits**

O bus de controlo pode ter funções adicionais.

## Memória secundária

A memória principal tem a vantagem de ser acessível com grande rapidez pelo processador. Infelizmente, como é constituída por circuitos electrónicos, só cumpre a sua função (a de memorizar os dados) enquanto for alimentada por corrente eléctrica.

Assim, existem dispositivos de **memória secundária** que armazenam dados de forma mais duradoura mas também têm um aspecto menos bom: o acesso processa-se de uma forma muito mais lenta. O computador normalmente usa a memória principal e só ocasionalmente transfere dados de/para a memória secundária.

Dos vários dispositivos de memória secundária, o **disco duro** é o mais importante. Tem uma capacidade muito grande e uma velocidade de acesso mais rápida que os restantes.

Os CDs e DVDs são preferíveis para se guardar informação de forma permanente por terem uma durabilidade maior e não serem influenciáveis por campos magnéticos.

## Dispositivos de entrada e saída

Os dispositivos I/O são controlados pelo processador, da mesma forma que a memória, via um bus de endereços, um bus de dados e um bus de controlo. Na realidade, são os mesmos buses de acesso à memória, e existe uma linha do bus de controlo que indica se o processador está a aceder à memória ou a um dispositivo de I/O.

Cada dispositivo está associado a um endereço onde o computador lê ou escreve informação.

Existe, no entanto, uma diferença fundamental entre a memória e os dispositivos de entrada/saída. Enquanto que a memória é passiva os dispositivos de entrada e saída podem ser activos, iniciando processos de comunicação com o processador. A forma que os dispositivos têm para fazer esta indicação é através de um pedido de interrupção. Um **pedido de interrupção** (*interrupt request*) é efectuado por um canal especial que liga um dispositivo de entrada/saída ao processador, a que se chama linha de **IRQ** (interrupt request). Quando o processador recebe um pedido de interrupção, pára temporariamente o que estava a fazer e executa um pequeno programa que atende o pedido.

## Conflitos de hardware

Os conflitos podem surgir quando há coincidência nas seguintes especificações para dispositivos diferentes:

* pedidos de interrupção (IRQ)
* endereços dos dispositivos de entrada/saída
* canais de DMA
* memória principal usada pelos dispositivos

Se dois dispositivos tiverem a mesma linha de IRQ, poderá surgir um conflito. Há casos em que dois dispositivos podem partilhar a mesma linha de IRQ sem haver problemas. Em geral, o sistema operativo sabe informar se dois dispositivos estão em conflito ou não.

Se houver conflito, a forma de o resolver é atribuir outra linha de IRQ ao dispositivo em questão (ou endereço de I/O, ou canal de DMA ou seja o que for que está a causar o conflito). A resolução nem sempre é pacífica, já que em alguns casos, os dispositivos têm parâmetros definidos de fábrica que impedem que sejam alterados.

# - Sistemas Operativos

O **sistema operativo** (SO) é a primeira camada de software de um sistema informático e todo o restante **software de aplicação** depende dele.

Os SO mais antigos não obrigavam as aplicações a dependerem dele, e muitos programas manipulavam directamente o hardware. Este tipo de comportamento não é aconselhado porque o hardware varia muito de máquina para máquina e não é viável escrever n versões de um mesmo programa para acomodar diversos tipos de hardware. Por outro lado, a manipulação directa dos recursos de hardware pode levar a que programas construídos com fins maliciosos possam afectar directamente esses recursos.

Os sistemas operativos cumprem de forma eficaz as suas duas funções essenciais, que são as seguintes:

1. **Disponibilizar uma máquina virtual, que é uma extensão da máquina física.**
2. **Gerir os recursos do sistema.**

Além de cumprirem estas funções, os actuais SO não permitem que um programa aceda directamente ao hardware nem que faça uma gestão própria dos recursos.

Nas funções do SO, a disponibilização de uma máquina virtual facilita a tarefa do programador, na medida em que este não tem de se preocupar com os detalhes de funcionamento de cada pedaço de hardware, nem com a gestão da memória ou do disco.

## A “shell”

Nos SO actuais a shell é essencialmente uma interface gráfica, em que os objectos são manipulados através de uma representação pictórica (ícones) no ecrã. No entanto, a interface com o utilizador não tem de ser gráfica

A shell é uma componente do sistema operativo que é independente das restantes componentes, no sentido em que é possível substituí-la por outra, mais conveniente.

## Tipos de SO

Os sistemas operativos são classificados quanto ao número de utilizadores que comportam, em **monoutilizador** e **multiutilizador**. Os sistemas monoutilizador estão preparados para suportar apenas um utilizador e não promovem a separação efectiva dos dados e processos de vários utilizadores.

Os sistemas multiutilizadores estão construídos para suportar vários utilizadores e efectivamente separam os ficheiros e programas que cada utilizador executa. Garantem a protecção dos dados de cada utilizador e permitem que cada utilizador execute apenas os programas para os quais possui autorização.

Nos sistemas multiutilizador, existe um utilizador com mais privilégios do que os outros, o chamado **super-utilizador**, que tem autorização para executar programas que configuram e alteram o próprio sistema operativo, que gerem os recursos do sistema, e que definem as características dos outros utilizadores.

Independentemente do número de utilizadores, um SO é classificado quanto ao número de programas que pode executar ao mesmo tempo, em **monoprocessamento** e **multiprocessamento**.

Um CPU funciona sequencialmente, isto é, uma instrução de cada vez.

Para executar vários programas ao mesmo tempo ou existem vários CPU e os programas são distribuídos pelos CPU ou só há um e terá de haver uma partilha do tempo que o CPU dedica a cada programa. Esta segunda solução designa-se por “**time-sharing**”.

## Ficheiros

O subsistema que gere os ficheiros chama-se **sistema de ficheiros** (filesystem). Tal como a shell, os actuais SOs disponibilizam vários sistemas de ficheiros, embora só seja possível escolher um para cada disco. Muitas vezes um mesmo sistema de ficheiros é usado em sistemas operativos diferentes.

Os ficheiros em disco são normalmente arrumados em **directorias** ou **pastas**, que se organizam numa estrutura em árvore. Uma cadeia de pastas é designada por **caminho** (path), e normalmente representada pelos nomes das pastas separadas por / ou \.

Compete ao sistema de ficheiros gerir esta estrutura de pastas e ficheiros. A estrutura é virtual, não correspondendo à forma como está organizado o disco fisicamente. O disco é, tal como a memória central, uma sequência linear de espaços para guardar bytes, acessíveis através de um endereço numérico.

Esta gestão automática por parte do sistema de ficheiros pode gerar uma situação de alguma desarrumação no disco. A esta condição chama-se **fragmentação** do disco .

## Processos

Um **processo** contém informação sobre a posição corrente no programa a ser executado, bem como sobre os conteúdos dos registos e posições de memória a serem utilizados pelo programa. Isto é, um processo contém informação sobre o estado de execução do programa.

É possível coexistirem vários processos a correrem o mesmo programa, uma vez que se distinguem pelo respectivo estado de execução. Num sistema multiprocessamento, em que os vários processos têm de partilhar o CPU, o sistema operativo atribui, rotativamente, um período de tempo a cada processo, findo o qual guarda o seu estado de execução e passa ao processo seguinte na fila. Quando chega novamente a vez do primeiro processo, o SO repõe o estado de execução guardado e volta a colocar o processo em funcionamento. Num sistema rápido, o método rotativo transmite a ilusão de que todos os processos correm ao mesmo tempo. Esta ilusão é formalizada na noção de **processador virtual**, em que cada processo corre no seu próprio processador, embora este na realidade corresponda a uma fatia de tempo do processador real.

Cada processo utiliza recursos do sistema, como a memória, ficheiros em disco, periféricos, que, num sistema multiprocessamento podem estar a ser utilizados simultaneamente por outros processos, que também podem comunicar entre si e partilhar recursos.

A partilha de recursos pode originar conflitos entre os processos, a que se chamam problemas de **concorrência**. A resolução deste tipo de problemas é da responsabilidade do sistema operativo e também do programador. Um dos mecanismos à disposição do programador para evitar um conflito é o **semáforo**. Quando um programa pretende utilizar um recurso de forma exclusiva, deve assinalar com um semáforo que mais ninguém deve executar operações sobre esse recurso. O sistema operativo disponibiliza e verifica os semáforos, garantindo a **exclusão mútua** de processos na região crítica.

Um outro problema que resulta da **concorrência** é o chamado “**deadlock**”, ou o impasse perante recursos bloqueados. Uma forma de resolver um deadlock é eliminar um dos processos, permitindo ao outro prosseguir. Mais tarde, o processo eliminado será reiniciado. Outra forma, utilizada no caso das impressoras, é o “**spooling**”. Neste método, não é permitido a um processo bloquear a impressora, mas a informação a imprimir deve ser enviada a um processo especial (o spooler) que guarda essa informação em disco até a impressora estar disponível ficando a impressão a cargo do spooler.

## Gestão de memória

É a velocidade de acesso da memória central que faz com que seja a forma mais eficaz de temporariamente guardar a informação associada a um programa em execução.

A componente do sistema operativo que se encarrega de gerir este compromisso entre o uso de memória central e secundária é o **gestor de memória**.

O gestor de memória coopera com o gestor de processos ao guardar em memória a informação associada aos processos em execução e copiando temporariamente em disco a informação associada a processos que não estão em execução. Este processo de cópia é designado por “**swapping**” e o espaço em disco que guarda a informação designa-se por ficheiro de “**swap**”. O swapping liberta espaço na memória para alojar informação para outro processo entretanto posto em execução. O processo de swapping tem o mérito de manter o sistema em funcionamento, mesmo com pouca memória.

Outro problema com a gestão de memória surge quando um único programa necessita de mais memória do que aquela que está disponível. O gestor de memória inclui um mecanismo de **memória virtual** que permite que tal programa seja executado. A memória virtual é uma abstracção que faz com que qualquer programa tenha a ilusão de ter a memória suficiente para funcionar.

A técnica mais usada de memória virtual é a **paginação**, onde toda a informação associada a um programa é dividida em partes relativamente pequenas a que se chamam **páginas**. Das várias estratégias, uma minimamente eficiente guarda em memória as páginas que são mais usadas enquanto que no disco ficam as que só ocasionalmente são acedidas.

## Gestão de dispositivos

É o sistema operativo que faz a ponte entre o hardware e o software de aplicação.

Dada a enorme variedade de dispositivos de I/O que um computador tem de controlar não é prático que o sistema operativo tenha em conta todas as possibilidades de periféricos.

Este problema é resolvido por recurso a componentes do SO que se designam por **gestores de dispositivos** (device drivers).

Os gestores de dispositivos têm um papel importante a cumprir na função 1 do SO (disponibilizar uma máquina virtual), porque apresentam ao programador uma funcionalidade uniforme para cada tipo de dispositivo. Este método tem a grande vantagem de não ser necessário alterar o programa para que funcione, por exemplo, com outra impressora: basta modificar o gestor de dispositivos da impressora, fornecido pelo respectivo fabricante.

##  “Boot strapping”

O sistema operativo controla todo o hardware, gere os processos e coordena as actividades do computador.

O sistema operativo é uma colecção de programas, muitos dos quais têm de ser lançados quando se liga o computador. Estes programas residem em disco, e têm de ser carregados em memória antes de serem executados, como qualquer outro programa.

A solução existe num tipo de memória de sigla ROM (read-only memory). Esta memória, por oposição à memória de tipo RAM, não perde a sua informação quando a energia eléctrica é desligada por ser gravada de fábrica e o seu conteúdo ser fixo. É esta ROM que contém o primeiro programa a ser executado pelo computador, a que se chama programa de “**bootstrapping**” ou de “boot”. É um pequeno programa que, após consulta de uma tabela situada no disco rígido, carrega o sistema operativo para a memória RAM.

A tabela consultada pelo programa de boot (designada nos PC por MBR – master boot record) contém informação sobre os sistemas operativos disponíveis e onde se situam. O MBR permite que o utilizador possa escolher entre vários sistemas operativos instalados no disco rígido.

Um computador com um SO Windows e um SO Linux é uma instalação com a designação de “dual boot”.

# Índice Remissivo

**analógico**, 4

**base**, 3

**bit**, 3

bits, 8

boot, 15

**bootstrapping**, 15

***bus***, 6

**bus de controlo**, 9

**bus de dados**, 9

**bus de endereços**, 9

**byte**, 3

**caminho**, 12

**Chaitin**, 5

**Claude Shannon**, 5

**Comparações**, 7

**concorrência**, 13

conflitos, 10

**contínua**, 4

CPU, 6

**deadlock**, 13

desfragmentador, 12

**Deslocação e rotação**, 7

device drivers, 14

**digital**, 4

**digitalização**, 4

**directorias**, 12

**disco duro**, 9

**discreta**, 4

**discretização**, 4

disponibilizar uma máquina virtual, 14

**Disponibilizar uma máquina virtual**, 11

dispositivo de I/O, 9

**dispositivos de entrada e saída**, 6

**DMA**, 6, 10

dual boot, 15

**endereço**, 8

**exclusão mútua** de processos, 13

**expoente**, 3

fetch cycle, 7

ficheiros em disco, 12

**fragmentação**, 12

**Gerir os recursos do sistema**, 11

**gestor de memória**, 14

**gestores de dispositivos**, 14

**Imagem**, 2

**instruções**, 7

**instruções condicionais**, 7

**instruções de salto e chamada**, 8

**instruções de transferência de informação**, 7

**instruções de transformação de informação**, 7

intensidade luminosa, 2

interface gráfica, 11

*interrupt request*, 10

**IRQ**, 10

Java, 4

**John von Neumann**, 5

**Kolmogorov**, 5

**mantissa**, 3

máquina virtual, 4

MBR, 15

**memória**, 6, 8

**memória secundária**, 9

**memória virtual**, 14

**monoprocessamento**, 12

**monoutilizador**, 11

**motherboard**, 6

**multiprocessamento**, 12

**multiutilizador**, 11

ondas sonoras, 2

**Operações aritméticas**, 7

**Operações lógicas**, 7

**outras instruções**, 8

*overclocking*, 6

**paginação**, 14

**páginas**, 14

partilha de recursos, 13

**pastas**, 12

path, 12

**pedido de interrupção**, 10

**placa de som**, 6

**placa gráfica**, 6

**precisão**, 2

**processador virtual**, 13

**programa**, 7

RAM, 15

**relógio**, 6

representação binária, 3

ROM, 15

**semáforo**, 13

**sistema de ficheiros**, 12

**software de aplicação**, 11

som, 2

**Som**, 2

spooler, 13

**spooling**, 13

**super-utilizador**, 12

**swap**, 14

**swapping**, 14

tempo, 2

**Texto**, 2

**time-sharing**, 12

**Turing**, 5

**velocidade**, 6

vídeo, 2

**vírgula fixa**, 3

**vírgula flutuante**, 3

**Von Neumann**, 8