Representação dos Dados na RAM

Mário Leite

Todo computador trabalha internamente apenas com **valores numéricos**. Assim, para que a CPU consiga reconhecer letras, números, símbolos e comandos digitados pelo usuário através do teclado, é necessário que cada caractere seja representado por um **código numérico padronizado**. Esses códigos são definidos por uma tabela denominada **Tabela ASCII** (*American Standard Code for Information Interchange*), amplamente utilizada na indústria de computadores para a troca de informações entre sistemas. Essa tabela estabelece um valor numérico único para cada caractere, permitindo que diferentes equipamentos e programas interpretem os dados de forma consistente. Quando o usuário pressiona uma tecla é gerado um código que é convertido pelo sistema em um valor numérico correspondente ao caractere desejado. Por exemplo, ao pressionar a tecla **[Backspace]** o sistema a reconhece pelo código **8**; a tecla **[Esc]** possui o código **27**, e assim por diante. Esses códigos permitem que programas identifiquem exatamente qual tecla foi acionada, possibilitando a execução das ações apropriadas.

Representação dos caracteres em *bytes*

Embora a Tabela ASCII padrão utilize originalmente **7 bits**, os caracteres são normalmente armazenados na memória em **1 *byte***, ou seja, **8 bits**. Isso ocorre porque o *byte* é a menor unidade de endereçamento utilizada pela maioria dos sistemas computacionais. Cada caractere, portanto, ocupa um *byte* na memória RAM. Como exemplo, a letra **A** (maiúscula) possui o código decimal **65** na tabela ASCII. Sua representação binária pode ser observada no esquema apresentado no **quadro 1**.

Estrutura de um *byte*

Um *byte* é composto por **oito bits** numerados convencionalmente do **bit 0** (menos significativo) ao **bit 7** (mais significativo). Por se tratar de uma base de numeração binária, cada bit pode assumir apenas dois valores possíveis: **0** ou **1**, o que, em termos de hardware, representa estados de "desativado" e "ativado", respectivamente. O valor posicional de cada *bit* dentro do *byte* segue uma progressão geométrica de razão **2**, conforme mostrado abaixo:

- bit 0 → 1
- bit 1 → 2
- bit 2 → 4
- bit 3 → 8
- bit 4 → 16
- bit 5 → 32
- bit 6 → 64
- bit 7 → 128

Cada bit contribui para o valor total do *byte* apenas quando está ativado (valor 1). Caso esteja desativado (valor 0), sua contribuição é nula.

Exemplo: representação da letra "A"

No quadro **1**, os bits **1, 2, 3, 4, 5 e 7** encontram-se desativados (0), enquanto os bits **0** e **6** estão ativados (1). Assim, seus valores posicionais correspondentes são **1** e **64**, respectivamente.

Somando-se apenas os valores dos bits ativados, temos: **0 + 64 + 0 + 0 + 0 + 0 + 0 + 1 = 65**

O valor **65**, em base decimal, corresponde ao código ASCII da letra **A** (maiúscula). De forma semelhante, a letra **a** (minúscula) possui o código ASCII **97**, evidenciando que letras maiúsculas e minúsculas são tratadas como caracteres distintos pelo computador.

Capacidade de representação de um *byte*

Como cada *byte* é formado por **8 bits** e cada *bit* assumindo dois estados possíveis, temos um total de: $2^8 = 256$ combinações distintas que podem ser representadas em um único *byte*. Isso significa que um *byte* pode armazenar até **256 valores diferentes**, suficientes para representar letras, números, símbolos e caracteres de controle definidos pela tabela ASCII estendida.

Representação de palavras na memória

Quando um texto é armazenado na memória, cada caractere ocupa um *byte* distinto. Assim, para representar a palavra **"Ana"**, são necessários **três *bytes***, conforme ilustrado no esquema apresentado no **quadro 2**.

Cada letra da palavra é armazenada em uma posição de memória consecutiva, permitindo que o computador reconheça a sequência correta de caracteres e a interprete como uma palavra.

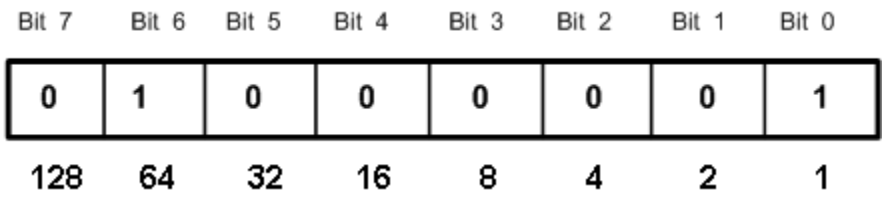
Cadeias de caracteres e alocação de memória

Pode-se então questionar: e se o nome da pessoa não fosse simplesmente **"Ana"**, mas sim **"Ana Paula Foz Rodrigues Leite"**, como isso seria tratado pelo computador?

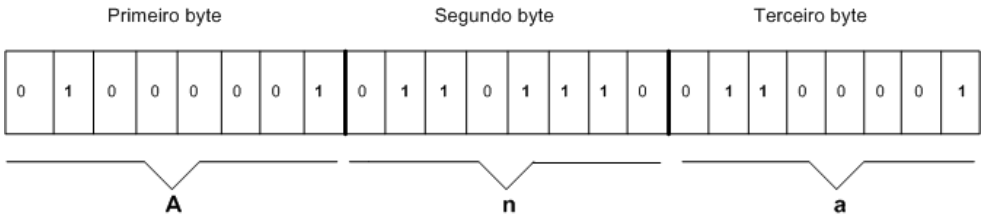
A resposta é simples: nesse caso, a CPU, por intermédio do sistema operacional e do programa em execução, alocaria **tantos *bytes* quanto fossem necessários** para armazenar todos os caracteres do nome completo. Cada letra, espaço em branco e símbolo ocuparia um *byte* adicional na memória, formando uma sequência contínua de dados conhecida como **cadeia de caracteres** (*string*). Desta forma, o tamanho ocupado na memória cresce proporcionalmente à quantidade de caracteres que compõem o texto, mantendo sempre o mesmo princípio básico: **um caractere por *byte***.

Considerações finais

A compreensão da forma como os dados são representados na memória RAM é fundamental para o entendimento do funcionamento interno dos computadores e das linguagens de programação. Ao perceber que letras, números e símbolos são, na realidade, valores binários organizados em *bytes*, o programador passa a ter uma visão mais clara de como a informação é armazenada, manipulada e transmitida pelos sistemas computacionais. Esse conhecimento serve de base para o estudo de estruturas de dados, manipulação de strings, entrada e saída de dados, além de conceitos mais avançados relacionados à arquitetura de computadores e ao funcionamento das linguagens de programação.



Quadro 1 - Representação esquemática da letra "A" em um *byte*.



Quadro 2 – Representação esquemática da palavra "Ana" em um *byte*.