Sistema binário

Introdução

Os computadores trabalham com um sistema incrível, que utiliza apenas dois valores para manipular qualquer informação. Isso quer dizer que todas as operações que o computador faz, desde permitir-nos a escrever um simples texto até jogar jogos 3D são realizados utilizando apenas dois valores, que por convenção são os dígitos “0” (zero) e “1” (um). Mas como isso é possível? Como o computador consegue dar andamento a todos os seus processos utilizando apenas os dígitos “0” e “1”? Como que isso tudo funciona na prática? Será que dentro de um processador ou em um CD/DVD veremos, literalmente, uma fileira de “0s” e “1s”? É exatamente isso que veremos nessa pequena série de artigos.

O que é binário?

De forma geral, binário é um sistema que utiliza apenas dois valores para representar suas quantias. É um sistema de *base dois*. Esses dois valores são o “0” e o “1”.

Daí podemos concluir que para 0 temos desligado, sem sinal, e para 1 temos ligado ou com sinal.

Vale ressaltar que o sistema que utilizamos diariamente é o *sistema de base dez*, chamado também por *base decimal*. Esse sistema utiliza os algarismos *indo-arábicos*, que são: 0, 1, 2, 3, 4, 5, 6, 7, 8, e 9.

Nós seres humanos fomos “treinados” para trabalhar com a base decimal. Ela é a ideal para nós. Mas, para os computadores a base binária é a ideal.

Nos computadores esses zeros (“0s”) e uns (“1s”) são chamados de *dígitos binários* ou somente *bit* (conjunção de duas palavras da língua inglesa binary digit), que é a menor unidade de informação dos computadores. Dessa forma, tanto faz dizer dígito “0” e dígito “1”, ou, bit “0” e bit “1”.

Formação de informações / O caractere.

São esses bits que formam qualquer informação, porém, um bit sozinho não faz nada, é apenas um sinal qualquer. Para que os bits possam realmente formar uma informação, precisam ser agrupados, reunidos. Esses grupos podem ser de 8, 16, 32 ou 64 bits.

8 bits

10100110

Apesar de parecer ser um sistema limitado, agrupando bits é possível fazer uma infinidade de representações. Vamos pegar como exemplo um grupo de 8 bits (tabela a seguir), onde é possível fazer as seguintes representações para os números decimais:

Tabela 01: Caracteres alfanuméricos e seus equivalentes em binário

Números Decimais	Código Binário
0	00000000
1	00000001
2	00000010
3	00000011
4	00000100
5	00000101
6	00000110
7	00000111
8	00001000
9	00001001
10	00001010
11	00001011
12	00001100
13	00001101
14	00001110

Na tabela 01 os números decimais estão representados em grupos de oito bits. Mas, acontece que, como ocorre no sistema decimal, todo zero que estiver a esquerda de dígitos binários não valem nada. Por exemplo: o decimal 14 é 1110 em binário, o mesmo que 00001110 ou 000000001110 ou ainda ...0000000000001110.

Como disse, o computador reuni grupos predefinidos de bits (8, 16, 32 ou 64) para formar uma informação, ou seja, um caractere. Um caractere é qualquer letra, número ou símbolo.

10100110 □ 8 bits = um caractere qualquer

1 + 1 = ?

Como este assunto é considerado “chato” por muitos estudantes, vamos a uma

brincadeirinha. Quanto é 1 + 1?

Bem, todos devem responder “2”.

Mas, eu não disse qual é a base (decimal ou binária). Se for decimal dará 2. E se for binária? Nesse caso dará 10.

Palavra

Agora vou explicar algo importante a saber: o conceito de “palavras”. Na terminologia dos computadores, *palavra* é um grupo de algarismos binário (bits) que podem ocupar uma localização na memória, e, que podem ser processados de uma só vez, podendo ser um número binário que é para ser manuseado como um dado, ou, uma instrução que diz ao computador que operação deve ser executada. Pode ser também um caractere ASCII representando uma letra do alfabeto, ou ainda, um endereço que diz ao processador onde se localiza um dado.

Existem tamanhos de palavras diferentes, onde cada um recebe um nome, veja:

- 4 bits = NIBBLE (2^4 =16 variações);
- 8 bits = BYTE (2^8 = 256 variações);
- 16 bits = WORD (2^{16} = 65.536 variações);
- 32 bits = DOUBLE WORD (2^{32} = 4.294.967.296 variações);
- 64 bits = QUAD WORD (2^{64} = 18.446.744.073.709.551.616 variações).

Para entender melhor, imagine que com palavras de 8 bits, as instruções, os endereços, os números e dados são representados por números binários de 8 bits. Dessa forma o menor número binário é 00000000 (ou 00 em hexadecimal), e, o maior número é 11111111 (ou FF em hexadecimal), o que corresponde de 0 a 256 valores diferentes (variações).

Quanto maior a palavra, maior será o número que se pode trabalhar. Por exemplo: com palavras de 16 bits pode-se trabalhar com números decimais até 65.536. É preciso frisar aqui que apesar de um determinado PC usar palavras de 8 bits, por exemplo, não significa que o processador desse PC ficará restringido a números decimais inferiores a 256. Simplesmente significa que será necessário usar duas ou mais palavras para representar números maiores. Dessa forma é certo dizer que um processador de 32 bits é mais rápido que um de 16 bits, pois, este último será obrigado a dividir números maiores (acima de 65.536) em números menores que sejam possíveis de se manipular com 16 bits, o que levará mais tempo.

Está confuso?

Atualmente os processadores acessam a memória a 64 bits por vez, porém, continuamos a usar o nome BYTE para referir ao tamanho de uma memória.

O correto seria usar “BYTE” para designar o tamanho de uma memória de 8 bits, e, QUAD WORD para memórias de 64 bits.

Mas, qual seria o lado prático disso? Nenhum, e pior, acaba confundindo, pois, imagine ter que usar BYTE para memórias de 8 bits (SIMM/30), WORD para memórias de 16 bits, e assim sucessivamente. Por isso até hoje o usual é o BYTE para designar o tamanho de uma memória.

Na prática

Agora um ponto importante. Como que isso tudo funciona na prática? Será que dentro de um processador ou em um CD/DVD veremos, literalmente, uma fileira de “0s” e “1s”? Não é assim que funciona.

Infelizmente eu já vi muitas publicações fazerem afirmações totalmente erradas, que só servem para confundir a cabeça de quem está aprendendo. Li uma certa vez que se você pagasse um CD-ROM e “ampliasse” a sua superfície de gravação, viria uma série de “0s” e “1s”. Ora, não é assim que funciona.

Em nível de eletrônica, os bits 0 e 1 são representados através de valores de *tensão*. Por exemplo: o bit 0 pode ser representado por valores entre 0 e 0,3 volts. Já o bit 1 pode ser representado por valores entre 2 e 5 volts. Esses números são apenas exemplos, não estamos afirmando aqui que são exatamente esses valores.

De forma geral, qualquer valor pode ser usado para representar os bits, depende do projeto, da aplicação e da tecnologia empregada. Com o avanço da tecnologia dos computadores, passou a se usar tensões cada vez menores, pois, os dispositivos eletrônicos passaram a trabalhar com tensões menores. Nos computadores são usados valores muito baixos, tais como esses que acabamos de mencionar.

Já o CD/DVD (dispositivos ópticos) armazenam as informações em forma de pequenos pontos denominados *Pits* e um espaço entre eles denominado *Lands*, que são interpretados no processo de leitura como “0s” e “1s” (bits).

Era Digital

Em nosso cotidiano é comum ouvir frases do tipo “era digital” ou “sistemas digitais” ou ainda “TV digital”. Mas, o que é digital? Resumidamente, digital é tudo aquilo que pode ser *transmitido* e/ou *armazenado* através de bits.

Um dispositivo digital é aquele que utiliza os bits para manipular qualquer tipo de informação (dados).