VOLTAR

Introdução aos processos de operação aritmética - Subtração

Cálculo de conversão de bases para responder às questões pertinentes à execução das especificações nas configurações de sistemas, comunicação remota e linguagem de máquina.

NESTE TÓPICO

Marcar tópico

Um sistema numérico pode ser usado para realizar duas operações básicas: adição e subtração. Pelo uso de adição e subtração, você pode então realizar multiplicações, divisões e qualquer outra operação numérica. Nesta aula, a aritmética binária (adição, subtração, multiplicação e divisão) será examinada, usando a aritmética decimal como um guia.

Empréstimo	7	12	9	13
Minuendo	8	3	0	3
Subtraendo	- 5	4	8	6
Diferença	2	8	1	7

Subtração binária

A subtração binária é realizada exatamente como subtração decimal. Portanto, antes de realizarmos a subtração binária, vamos revisar a subtração decimal. Você sabe que, se 5486 é subtraído de 8303, a diferença 2817 é obtida.

Como o dígito 6 no subtraendo é maior que o dígito 3 no minuendo, um 1 é emprestado do próximo dígito de maior ordem no minuendo. Se esse dígito é zero, como no nosso exemplo, 1 é emprestado do próximo dígito de ordem maior que contenha um número diferente de zero. Aquele dígito é reduzido de 1 (de 3 para 2 no nosso exemplo), e aos dígitos pulados no minuendo é dado o valor 9. Isso é equivalente a remover 1 de 30 com o resultado de 29, por exemplo.

No sistema decimal, o dígito emprestado tem o valor de 10. Portanto, o dígito do minuendo agora tem o valor 13, e 6 de 13 resulta 7.

Na segunda coluna 8 de 9 resulta 1. Desde que o subtraendo é maior que o minuendo na terceira coluna, 1 é transportado do próximo dígito de ordem superior. Isso suspende o valor do minuendo de 2 para 12, e 4 de 12 resulta 8. Na quarta coluna, o minuendo foi reduzido de 8 para 7 por causa do empréstimo prévio, e 5 de 7 resulta 2. Toda vez que 1 é emprestado de um dígito de ordem superior, o empréstimo é igual, em valor, à base do sistema numérico. Portanto, um empréstimo no sistema numérico decimal é igual a 10, enquanto um empréstimo no sistema numérico binário é igual a 2.

Quando se subtrai um número binário de outro, você usa o mesmo método descrito para subtração decimal.

A figura a seguir resume as quatro regras para subtração binária:

1: 0 − 0 = 0	
2: 1 − 1 = 0	
3: 1 − 0 = 1	
4: 0 − 1 = 1	empresta 1

Para ilustrar o processo da subtração binária, vamos subtrair 1101 de 11011.

A linha "empréstimo" nos mostra o valor de cada dígito do minuendo depois da ocorrência de cada transporte. Lembre-se de que o binário 10 é igual ao decimal 2.

Empréstimo	0	10	10	0	0
Minuendo	1	1	0	1	1
Subtraendo	-	1	1	0	1
Diferença		1	1	1	0

Na primeira coluna, 1 de 1 resulta 0 (regra 2). Então, 0 de 1 na segunda coluna resulta 1 (regra 3). Na terceira coluna, 1 de 0 necessita de um empréstimo da quarta coluna. Assim, 1 de 10_2 resulta 1 (regra 4).

O minuendo na quarta coluna é agora 0, por causa do empréstimo. Portanto, um empréstimo é necessário da quinta coluna, de maneira que 1 de 10_2 na quarta coluna resulta 1 (regra 4). Por causa do empréstimo anterior, o minuendo na quinta coluna é agora 0, e o subtraendo é 0 (não existe), de modo que 0 de 0 resulta 0 (regra 1). O 0 na quinta coluna não é mostrado na diferença, pois não é um bit significativo. Assim, a diferença entre 11011_2 e 1101_2 é 1110_2.

Pode-se verificar isso convertendo os números binários para decimal. Como exemplo de subtração binária, subtraia 00100101_2 de 11000100_2, como mostrado a seguir.

Quando um empréstimo ("borrow") é necessário, 1 é obtido do próximo bit de ordem superior que possui 1. Aquele bit, então, torna-se 0 e a todos os bit pulados (bits de valor 0) damos o valor 1. Isso é equivalente a remover 1 de 1000_2.

Empréstimo		0	0	1	1	1	10	1	10
Minuendo		1	1	0	0	0	1	0	0
Subtraendo	-	0	0	1	0	0	1	0	1
Diferença		1	0	0	1	1	1	1	1

Como na adição binária, os microprocessadores geralmente realizam subtrações em grupos de números de 8 bits. No exemplo anterior, a resposta contém apenas 6 bits significativos, mas dois 0 foram acrescentados para manter o grupo de 8 bits. Isso será verdade também para o minuendo e o subtraendo.

Para conhecer um pouco mais sobre essa sequência, veja o infográfico abaixo. Este infográfico faz parte da sequência desta aula e, portanto, é essencial para a aprendizagem.

SUBTRAÇÃO COM
NÚMERO BINÁRIOS

Para realizar a subtração dos números binários, temos que entender as regras:

1 : 0 - 0 = 0

2 : 1 - 1 = 0

3 : 1 - 0 = 1

4 : 0 - 1 = 1

Vamos realizar a subtração dos seguintes números binários:

$$1001_2 - 111_2$$

Montamos o cálculo :

```
 1001
- 111
```

Nos primeiros casos a partir da direita utilizamos para o cálculo a regra 2 (1-1=0) onde o resultado é 0.

```
 1001
- 111
-----
    0
```

Na segunda casa utilizamos para o cálculo a regra 4, onde temos que emprestar a base da casa ao lado. No caso, a primeira casa ao lado vale 0, por isso temos que emprestar da próxma casa que vale 1

```
 1001
- 111
-----
    0
```

Emprestamos a base da quarta casa para a terceira, ficando a quarta casa com 0 e a terceira casa com 2 em binário 10_2, precisamos ainda, emprestar para a segunda casa.

```
 0
 1001
- 111
-----
    0
```

A terceira casa vale 10_2 e empresta a base para a segunda casa e fica com 1. A segunda casa fica com 2 em binário (10_2)

```
 0 1 1
 1001
- 111
-----
    0
```

Utilizando a regra 4 (equivalente a 2-1), temos o resultado de 1.

```
 0 1 1
 1001
- 111
-----
   10
```

A terceira casa temos agora a regra 2 resultando em 0.

```
 0 1 1
 1001
- 111
```

E na quarta e última casa, temos 0, mas nada é igual a 0.

```
 0 1 1
 1001
- 111
```

010	0010
	Como resultado, temos: 0010_2

Aritmética do complemento de dois

Uma característica do sistema de complemento de dois é que tanto os números com sinal quanto os números sem sinal podem ser somados pelo mesmo circuito. Por exemplo, suponha que você deseja somar os números sem sinal 132_{10} e 14_{10}.

O microprocessador tem um circuito ALU que pode somar números binários sem sinal. Quando aparece o padrão **10000100_2** em uma entrada e **00001110_2** na outra entrada, resulta **10010010_2** na saída.

	$1\ 0\ 0\ 0\ 0\ 1\ 0\ 0_2$	$+132_{10}$
$+$	$0\ 0\ 0\ 0\ 1\ 1\ 1\ 0_2$	$+14_{10}$
	$1\ 0\ 0\ 1\ 0\ 0\ 1\ 0_2$	$+146_{10}$

Surge a pergunta: como a ALU sabe que os padrões de bits nas entradas representam número sem sinal e não números em complemento de dois?

E a resposta é: não sabe. A ALU sempre soma como se as entradas fossem números binários sem sinal. Sempre produzirá o resultado correto, mesmo se as entradas forem números em complemento de dois.

Observe o exemplo anterior. Se você assumir que as entradas são números com sinal em complemento de dois, então:

Verifique que os padrões de bits são os mesmos. Apenas o significado mudou.

Na primeira linha, nós assumimos que o padrão de bits representan números sem sinal e o somador produz o resultado sem sinal conveniente. Na segunda linha, nós assumimos que os padrões de bits representam números com sinal. Novamente, o somador fornece o resultado correto.

Transporte					1	1				
	1	0	0	0	0	0	1	0	0_2	260_{10}
	+	0	0	0	0	1	1	1	0_2	14_{10}
	1	0	0	0	1	0	0	1	0_2	274_{10}

Isso comprova um ponto muito importante. O somador na ALU sempre soma padrões de bits como se eles fossem números binários sem sinal.

É a nossa interpretação desses padrões que decide se números com ou sem sinal estão sendo indicados. O bom do complemento de dois é que os padrões de bits podem ser interpretados de qualquer maneira. Isso nos permite trabalhar com números com e sem sinal sem requerer diferentes circuitos para cada padrão.

Complemento de 2

Vamos utilizar o número binário :

11001011_2

O Complemento de 2 de um número consiste no complemento de 1 de um número mais a soma de 1.

Para tirarmos o complemento de 1 de um número, invertemos todos os dígitos do número, em que o que é 0vira 1 e o que é 1 vira 0.

00110100_2

O resultado do complemento de 1 de **11001011_2** é **00110100_2**

Para chegarmos no complemento de 2 de um número, utilizamos o resultado do complemento de 1 mais a soma do número 1.

$$\begin{array}{r} 00110100 \\ + \quad\quad\quad 1 \\ \hline 00110101 \end{array}$$

O Complemento de 2 de 11001011_2 é 00110101_2

A aritmética de complemento de dois também simplifica a ALU em outro ponto. Todo o microprocessador tem uma instrução de subtração. Assim, a ALU deve ser capacitada a subtrair um número de outro. Entretanto, se isso necessitar de um circuito de subtração separado, a complexidade e o custo da ALU seriam aumentados. Felizmente, a aritmética de complemento de dois permite a ALU realizar operações de subtração usando um circuito somador. Ou seja, a CPU usa o mesmo circuito tanto para soma como para subtração.

A CPU realiza a subtração por processo de adição binária. Para ver como isso funciona, será útil estudar um processo similar com o sistema numérico decimal. O equivalente decimal do complemento de dois é chamado complemento de dez. Desde que você esteja mais familiarizado com o sistema numérico decimal, examinaremos conjuntamente a aritmética do complemento de dez.

Subtração em complemento de dois

Existe outra forma de fazer a subtração em binário. Pode-se obter o resultado da subtração somando-se o complemento de 2 do valor, e ignorando-se o último algarismo. Veja como obter o complemento de 2 de um número binário:

O complemento de 2 de um número binário é obtido tomando-se o complemento de 1 do número e somando-se 1 na posição do bit menos significativo. Esse processo está ilustrado a seguir para $101101_2 = 45_{10}$

- 010010 – complementa-se (inverte-se os 0 por 1 e os 1 por 0) cada bit para obter o complemento de 1

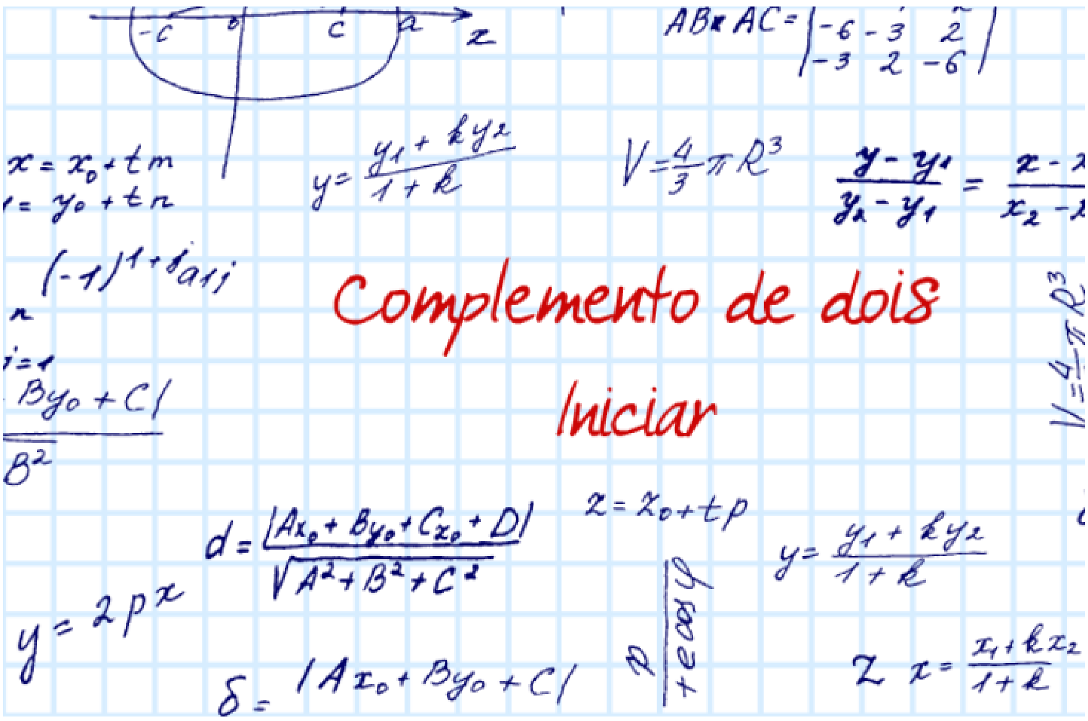
- + 1 – adiciona-se para obter o complemento 2
- 010011 – complemento de 2 do número binário original
- depois disso, basta somar os números e ignorar o dígito mais significativo para obter o resultado esperado

Veja o exemplo:

Subtração normal

		1		1	1	1		
	1	1	0	1	1	1	0	(110)
−			1	0	1	1	1	(23)
	1	0	1	0	1	1	1	(87)

Para entender o conceito, veja o infográfico abaixo. Este infográfico faz parte da sequência desta aula e, portanto, é essencial para a aprendizagem.



Referências

STALLINGS, Willian. *Arquitetura e organização de computadores*. 5. ed. Prentice Hall. São Paulo, 2006.

TANENBAUM. Andrew S. *Organização estruturada de computadores*. 5. ed. Rio de Janeiro: LTC, 2007.

MACHADO, Francis B.; MAIA, Luiz P. *Arquitetura de sistemas operacionais*. 4. ed. Rio de Janeiro: LTC, 2007.

WEBER, Raul Fernando. *Arquitetura de computadores pessoais*. 2. ed. Porto Alegre: Sagra Luzzatto, 2003.

______. *Fundamentos de arquitetura de computadores*. 3. ed. Porto Alegre: Sagra Luzzatto, 2004.

Avalie este tópico

ANTERIOR
Introdução aos processos de operação aritmética - Soma

Índice

PRÓXIMO
Funções e portas lógicas - Definição, representação, tabela verdade e expressões booleanas

Biblioteca (https://www.uninove.br/conheca-a-uninove/biblioteca/sobre-a-biblioteca/apresentacao/)
Portal Uninove (http://www.uninove.br)
Mapa do Site

Ajuda? (https://ava.un recurso=)