Capítulo 5

Circuitos combinacionais II

Dando continuidade à construção dos blocos básicos de processamento de informações com as técnicas de projeto digital, vamos trabalhar neste capítulo com a subtração de números binários.

Um ponto importante que será abordado é a introdução da representação de números negativos binários. Esse tópico é fundamental para que se compreenda, em contextos mais avançados e em linguagens de programação como C ou C++, por que temos de declarar se um inteiro é representado com ou sem sinal.

A compreensão desse tópico também é fundamental para que, em uma situação de projeto, se tenha conhecimento dos elementos

necessários à decisão de como representar os números, e ainda para que se entendam as consequências disso.

1 Representando números negativos

Conforme já abordado anteriormente, os números em um sistema digital são comumente representados por bits e bytes; quer dizer, podemos representar um valor de 0 a 255 usando um byte de 8 bits.

Entretanto, quando estamos realizando uma operação de subtração, é comum, e perfeitamente natural, que o resultado da operação seja um número negativo; por conta disso, é necessário estabelecer uma padronização para representar um número negativo.

Um ponto importante, neste caso, é pensar um pouco sobre qual a melhor forma de "avisar" aos circuitos eletrônicos, que farão as operações matemáticas, que um número negativo está sendo representado. Para isso, vamos adotar aqui a padronização descrita na técnica de complemento de 2 (IDOETA; CAPUANO, 1999), cujos fundamentos são descritos a seguir.

Uma das formas mais comuns de representar um número negativo é ter um bit para identificar se o número é positivo ou negativo. Vamos adotar que esse bit é o mais significativo. Assim, em um circuito em que representamos um número com 8 bits, o bit 7 será o nosso bit de sinal – lembrando que os bits de uma representação binária começam em 0, ou seja, um número de 8 bits vai ter seu bit menos significativo com índice 0 e seu bit mais significativo, com índice 7.

Outro ponto importante é considerar esta questão: qual valor desse bit indica que estamos representando números positivos e qual valor indica que estamos representando números negativos? Vamos adotar que, quando o bit de sinal for 0, estamos representando um número positivo, e quando o bit de sinal for 1, estamos representando um número negativo.

A primeira consequência de adotar um bit de sinal é que a representação será dividida em números positivos e negativos. Se considerarmos que zero é um número positivo, metade dos valores que podemos representar serão negativos, e a outra metade, positivos.

Para exemplificar isso, vamos usar um byte de 8 bits. Se esse byte representar apenas números positivos, será possível representar valores de 0 a 255. Entretanto, se esse byte também representar números negativos, serão representados valores de -128 até +127.

A forma mais prática e direta de inverter o sinal de um número binário, dentro da técnica de complemento de 2, é composta por duas operações booleanas. A primeira delas é a simples negação de todos os bits, ou seja, quando um número é positivo, um determinado bit é zero, ele se torna um e vice-versa. Isso também vale para o bit de sinal, ou seja, o bit mais significativo do número que estamos representando.

É importante notar que essa inversão deve ser feita quando já se sabe o número de bits com o qual o circuito como um todo trabalha. Ou seja, se um circuito é feito para trabalhar com 4 bits que representem uma grandeza, isso também vale para os números negativos. Sem isso, o número de bits seria infinito, o que não seria viável para um circuito implementado fisicamente.

A segunda operação booleana necessária para fazer troca de sinal, de positivo para negativo, é somar 1 ao resultado da primeira operação, usando as mesmas regras de soma do capítulo anterior.

PARA PENSAR

Você já parou para pensar em como representar números muito pequenos (como 0,00000001) ou muito grandes (como 100.000.000.000) usando os mesmos circuitos para representar e fazer as operações matemáticas? Para se aprofundar no assunto, faça uma pesquisa sobre como são representados os números em ponto flutuante.

Para tornar mais palpável o processo de inverter um número binário usando a técnica do complemento de 2, confira o exemplo da tabela 1. Nela, invertemos o número binário 0110 (6 em decimal).

Tabela 1 – Operação da operação de inversão de sinal

	Bit de sinal			
	Bit 3	**Bit 2**	**Bit 1**	**Bit 0**
Positivo	0	1	1	0
Complemento de 1	1	0	0	1
Soma do complemento de 2				1
Complemento de 2	1	0	1	0

Se o mesmo valor 6 fosse representando em 6 bits, a operação seria representada da forma expressa pela tabela 2.

Tabela 2 – Operação de inversão de sinal com um número representado em 6 bits

	Bit de sinal					
	Bit 5	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Positivo	0	0	0	1	1	0
Complemento de 1	1	1	1	0	0	1
Soma do complemento de 2						1
Complemento de 2	1	1	1	0	1	0

Observe que quem determina o número de bits de um número negativo é a pessoa que está fazendo o projeto do circuito eletrônico que vai manipular essas grandezas.

Note, também, que o maior número positivo que poderá ser representado tem um bit a menos do que o total de bits com que o circuito

trabalha, pois, se ultrapassarmos esse valor, vamos representar um valor negativo, e não um valor positivo.

Assim, no exemplo em que estamos representando o número 6 em 4 bits, o maior valor positivo que podemos representar em 4 bits seria o valor 7, que em binário corresponde a 0111. A mesma regra vale para qualquer quantidade de bits.

2 Meio subtrator

Dando continuidade às operações matemáticas utilizando as técnicas de projeto digital, vamos trabalhar a subtração de números binários, começando pela subtração mais simples possível, de apenas um bit.

Para entendermos bem, vamos começar com uma subtração na representação decimal. Ao subtrairmos B de A e guardarmos o resultado em S, podemos ter os resultados expressos pela tabela 3.

Tabela 3 – Subtração em decimal de 1 bit

A	B	S
0	0	0
0	1	-1
1	0	1
1	1	0

Porém, em binário, a forma de representar o negativo — ou o "empresta 1", ou ainda o borrow — deve ser um segundo bit. Como estamos usando a letra B para representar um dos números da operação, vamos representar o borrow com o T de transporte, em que Te é a entrada do transporte e Ts, a saída do transporte do circuito que estamos modelando. Assim, a representação da subtração da tabela 3 é apresentada em binário na tabela 4, a seguir.

Tabela 4 – Subtração de um bit com o "empresta 1" representado em T

A	B	S	Ts
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

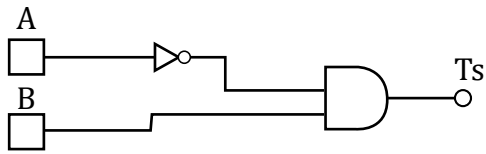
A tabela 4 é a tabela-verdade do meio subtrator. Se a observarmos, teremos a seguinte equação para descrever o comportamento que queremos do circuito:

$$S = (\overline{A} \,.\, B) + (A \,.\, \overline{B})$$

Essa equação corresponde exatamente a uma porta XOR.

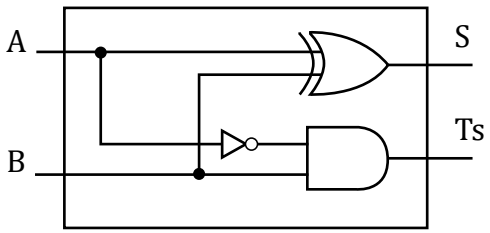
Continuando a análise da tabela 4, podemos descrever a equação de Ts como Ts = $\overline{A}$. B, a qual pode ser transformada no circuito a seguir, representado pela figura 1.

Figura 1 – Ts = $\overline{A}$. B



Ao unir os dois circuitos em um mesmo bloco, obtemos o circuito do meio subtrator, representado na figura 2.

Figura 2 — Meio subtrator



Fonte: adaptado de Idoeta e Capuano, 1999.

3 Subtrator completo

O meio subtrator, como o próprio nome já indica, não é capaz de fazer uma subtração completa, sendo necessário um circuito um pouco mais complexo para realizar essa tarefa.

Para evidenciar essa situação, vamos analisar uma subtração de dois números de 5 bits, no caso, A = 01100 e B = 00011. Podemos representar a subtração de acordo com a tabela 5, a seguir, em que Ts é o "empresta 1" que descrevemos no meio subtrator.

Tabela 5 – Subtraindo dois números binários de 5 bits com sinal

	Bit de sinal				
	Bit 4	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
A	0	1	1	0	0
B	0	0	0	1	1
"Empresta 1"			1	1	
S	0	1	0	0	1
	Ts = 0	Ts = 0	Ts = 0	Ts = 1	Ts = 1

Dessa forma, para implementarmos um subtrator que avalie o "empresta 1" do bit anterior e para fazermos a subtração de uma das colunas da subtração da tabela 5, temos a seguinte tabela-verdade.

Tabela 6 – Tabela-verdade do subtrator completo de 1 bit

A	B	Te	S	Ts
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Na tabela 6, Te é o "empresta 1" de entrada, obtido do bit anterior, e que é o menos significativo em relação ao bit cuja saída S estamos gerando, e Ts é o "empresta 1" de saída, que será o "empresta 1" de entrada do próximo bit cuja saída vamos gerar.

Explicado isso, vamos gerar as equações que descrevem S e Co:

$$S = (\overline{A} \,.\, \overline{B} \,.\, Te) + (\overline{A} \,.\, B \,.\, \overline{Te}) + (A \,.\, \overline{B} \,.\, \overline{Te}) + (A \,.\, B \,.\, Te)$$

$$Ts = (\overline{A} \,.\, \overline{B} \,.\, Te) + (\overline{A} \,.\, B \,.\, \overline{Te}) + (\overline{A} \,.\, B \,.\, Te) + (A \,.\, B \,.\, Te)$$

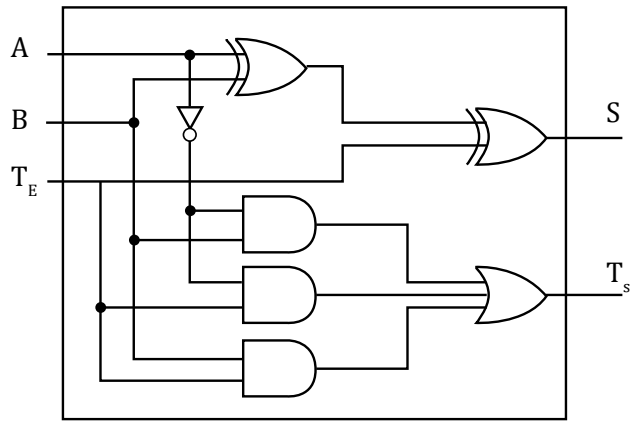
Ao simplificarmos essas equações, obtemos:

$$S = A\ (+)\ B\ (+)\ Te$$

$$Ts = (\overline{A} \,.\, B) + (\overline{A} \,.\, Te) + (B \,.\, Te)$$

Implementando essas equações, obtemos o circuito a seguir, que corresponde ao circuito que pode implementar qualquer uma das colunas da subtração binária da figura 3.

Figura 3 – Subtrator completo



Fonte: adaptado de Idoeta e Capuano, 1999.

O subtrator completo gera o resultado de apenas um bit, entretanto, a combinação de vários subtratores completos nos permite construir subtratores no tamanho necessário à aplicação.

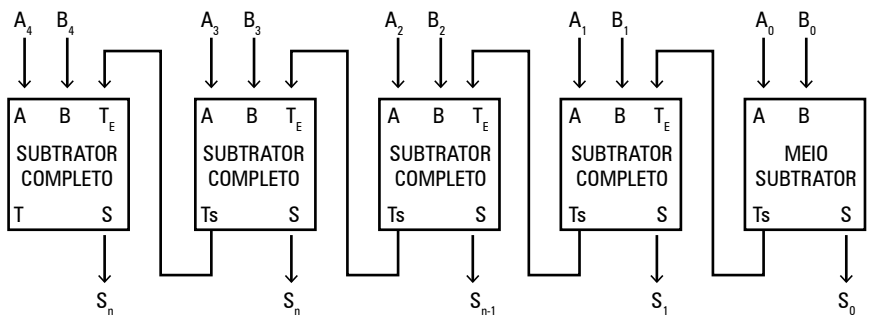
Para ilustrar isso, vamos implementar um somador de dois números de 5 bits, que vai gerar uma saída de 5 bits, sendo o primeiro número, A, desmembrado nos bits A_3, A_2, A_1 e A_0, e o segundo número, B, desmembrado em B_3, B_2, B_1 e B_0. A saída S, por sua vez, é desmembrada em S_4, S_3, S_2, S_1 e S_0. Esquematizando a soma, teremos o quadro expresso pela figura 4.

Figura 4 – Montando a subtração de binários

	A_4	A_3	A_2	A_1	A_0
-	B_4	B_3	B_2	B_1	B_0
	S_4	S_3	S_2	S_1	S_0

Usando o subtrator completo, podemos implementar a subtração da figura 4 ligando a saída Ts de um bit na entrada Te do bit imediatamente mais significativo, à exceção dos bits A_0, B_0 e S_0, que são os menos significativos. Nesse caso, podemos usar o meio somador, como expresso na figura 5.

Figura 5 — Implementando um subtrator de 5 bits



Fonte: adaptado de Idoeta e Capuano, 1999.

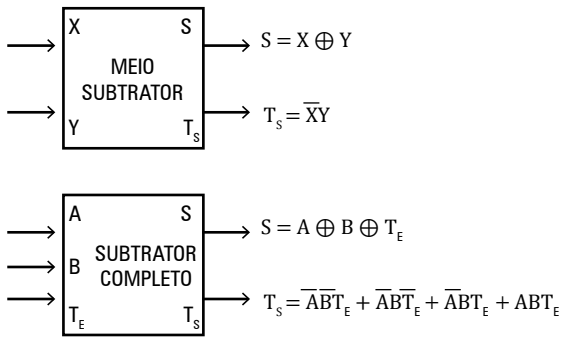
Além disso, os números negativos são representados usando a notação de complemento de 2, que discutimos no início deste capítulo.

4 Subtrator completo a partir do meio subtrator

Em muitas situações de projeto, temos uma boa vantagem ao quebrarmos o projeto de blocos muitas vezes repetidos no circuito que estamos construindo, principalmente em termos de otimização do projeto físico. Essa situação se aplica aos subtratores, visto que são blocos que se repetem pelo número de bits manipuláveis do circuito.

Ao analisarmos as expressões lógicas que descrevem o meio subtrator e o subtrator completo, percebemos que elas podem ser reescritas de uma forma diferente.

Figura 6 – Comparação entre as expressões lógicas do meio subtrator e do subtrator completo



Fonte: adaptado de Idoeta e Capuano, 1999.

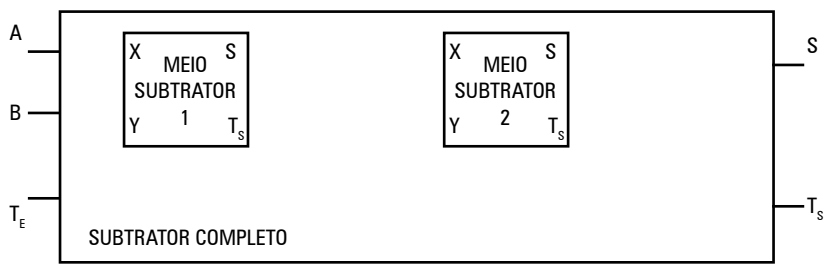
$$Ts = Te \,.\, (\overline{A} \,.\, \overline{B} + A \,.\, B) + \overline{A} \,.\, B \,.\, (\overline{Te} + Te)$$

Podemos simplificar essa equação da seguinte forma:

$$Ts = Te \,.\, (\overline{A\,(+)\,B}) + \overline{A} \,.\, B \rightarrow \text{Expressão “empresta 1”}$$

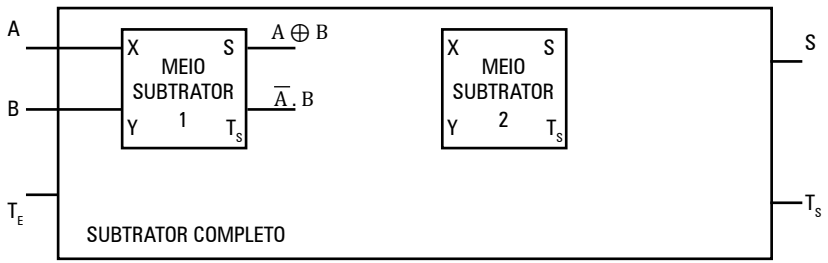
Vamos montar nosso subtrator completo usando dois subtratores, e faremos a ligação deles passo a passo.

Figura 7 – Subtrator completo usando dois meio subtratores



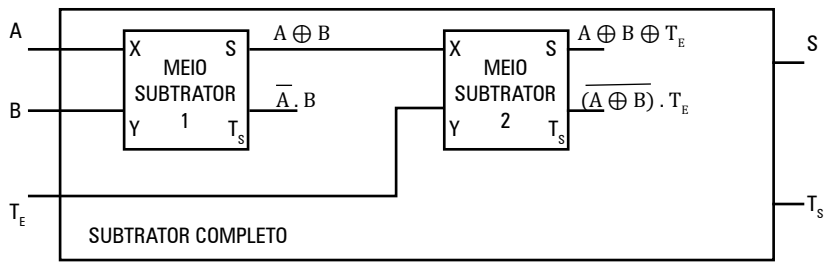
Ao ligarmos as entradas A e B do somador completo a X e Y do meio subtrator 1, respectivamente, teremos a situação expressa pela figura 8.

Figura 8 – Ligando as entradas A e B do subtrator completo no meio subtrator 1



Ao ligaremos a saída S do meio subtrator 1 à entrada X do meio subtrator 2, e a entrada Te do subtrator completo à entrada Y do meio subtrator 2, obteremos, na saída S do meio subtrator 2, a expressão que descreve a saída do subtrator completo.

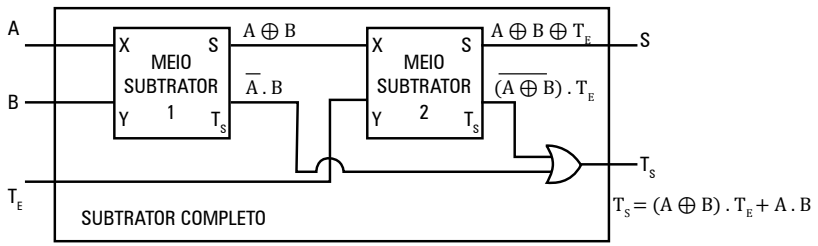
Figura 9 – Ligando o meio subtrator 2



Analisando a equação "empresta 1", concluímos que os dois termos da equação já estão presentes no circuito, na saída Ts do meio subtrator 1 e na saída Ts do meio subtrator 2.

Assim, a equação "empresta 1" do subtrator completo é obtida por meio de uma operação OR entre as saídas Ts dos dois meios subtratores, como podemos conferir na figura 10, a seguir.

Figura 10 – Subtrator completo totalmente montado a partir de dois meios subtratores



Considerações finais

Neste capítulo, pudemos conferir como são representados os binários negativos e como eles refletem no projeto dos circuitos digitais destinados a fazer as operações de subtração, as quais podem ter como resultado números negativos.

Além disso, tivemos um segundo exemplo de como implementar um circuito digital escalável, no qual se pode aumentar a quantidade de bits que ele é capaz de manipular apenas aumentando-se o número de blocos que se repetem no circuito. Junto com o circuito somador, esse é um dos circuitos combinacionais mais úteis na manipulação de dados.

Referências

IDOETA, Ivan V.; CAPUANO, Francisco G. **Elementos de eletrônica digital**. São Paulo: Érica, 1999.