10 HIGH-RADIX MULTIPLIERS

Chapter Goals

Study techniques that allow us to handlemore than one multiplier bit in each cycle(two bits in radix 4, three in radix 8, . . .)

Chapter Highlights

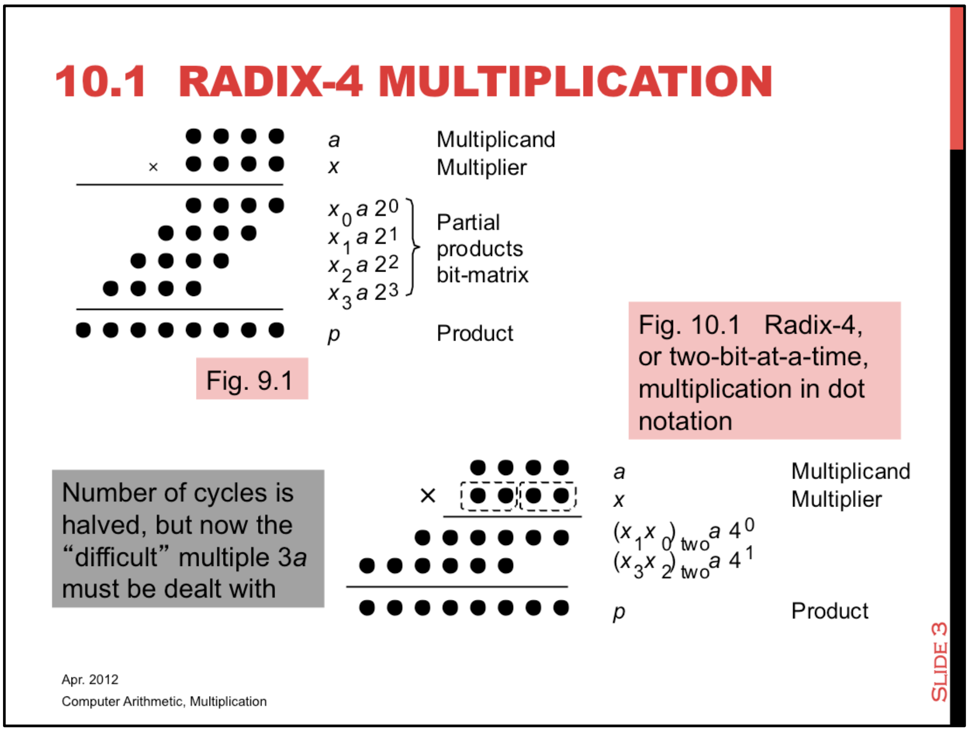
High radix gives rise to “difficult” multiples
Recoding (change of digit set) as remedy
Carry-Save addition reduces cycle time
Implementation and optimization methods

Vejamos agora o Capítulo 10.

HIGH-RADIX MULTIPLIERS: TOPICS

Topics in This Chapter
10.1 Radix-4 Multiplication
10.2 Modified Booth's Recoding
10.3 Using Carry-Save Adders
10.4 Radix-8 and Radix-16 Multipliers

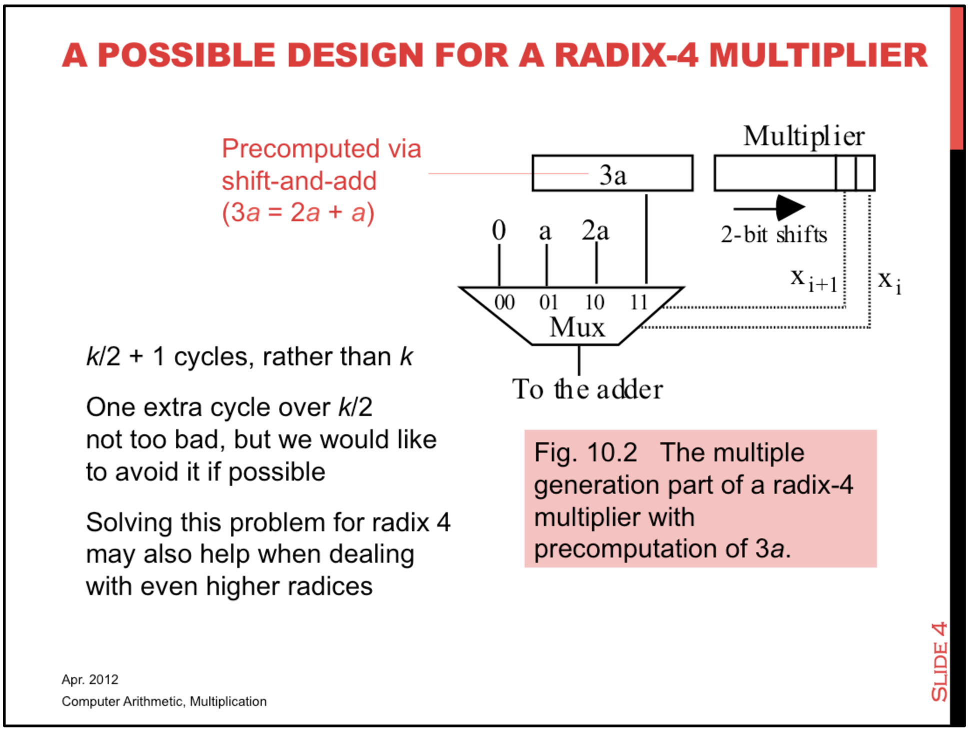
Vejamos os tópicos que serão abordados.



O uso da multiplicação de "high radix" envolve essencialmente lidar com mais de 1 bit do multiplicador em cada ciclo.

Para o exemplo radix-4 (Fig 10.1), em cada etapa, o termo de produto parcial (x(i + 1), xi) precisa ser formado e adicionado ao produto parcial cumulativo. Note que o número de ciclos é minimizado pela metade em relação à figura 9.1, quando realizamos a construção dos produtos parciais utilizando um 2 bits por vez do multiplicador.

Na multiplicação da figura ,cada linha de pontos na matriz de produtos parciais representa 0 ou uma versão deslocada de "a", então precisaremos dos múltiplos 0a, 1a, 2a e 3a. Os três primeiros múltiplos não apresentam problemas (2a é simplesmente a versão deslocada de a). Mas o cálculo de 3a precisa de pelo menos uma operação de adição (3a = 2a + a).



Uma primeira opção para solucionar o problema da multiplicação por "3a" é calcula-lo uma vez no início e armazená-lo em um registro para uso futuro.

A partir de então utilizamos um mux 4:1 para selecionar o valor (0, a, 2a, 3a).

10.2 MODIFIED BOOTH'S RECODING

Table 10.1 Radix-4 Booth's recoding yielding $(z_{k/2} \ldots z_1 z_0)_{four}$

x_{i+1}	x_i	x_{i-1}	y_{i+1}	y_i	$z_{i/2}$	Explanation
0	0	0	0	0	0	No string of 1s in sight
0	0	1	0	1	1	End of string of 1s
0	1	0	0	1	1	Isolated 1
0	1	1	1	0	2	End of string of 1s
1	0	0	-1	0	-2	Beginning of string of 1s
1	0	1	-1	1	-1	End a string, begin new one
1	1	0	0	-1	-1	Beginning of string of 1s
1	1	1	0	0	0	Continuation of string of 1s

Context — Recoded radix-2 digits — Radix-4 digit

Example

	1 0	0 1	1 1	0 1	1 0	1 0	1 1	1 0	Operand x
(1)	-1 0	1 0	0 -1	1 0	-1 1	-1 1	0 0	-1 0	Recoded version y
(1)	-2	2	-1	2	-1	-1	0	-2	Radix-4 version z

Apr. 2012
Computer Arithmetic, Multiplication

SLIDE 5

Para evitar o fato de ter de fazer a operação 3a podemos fazer uso do algoritmo de Booth para a recodificação do multiplicando. Seu resultado não terá "1"s ou "-1"s consecutivos. Assim, se a multiplicação de radix-4 for realizada com o multiplicador recodificado, apenas os múltiplos "±a" e "±2a" do multiplicando serão necessários, todos os quais são facilmente obtidos por deslocamento e / ou complementação.

Observe no exemplo que o início de uma sequencia de bits "1" teremos a recodificação para o valor igual a "-1". Seguiremos completando com zeros até atingir o final da sequencia e então completaremos com o valor igual a "1" na posição do bit mais a esquerda do final desta sequencia. Faremos esse processor sempre que encontrarmos um sequência (no mínimo dois bits 1 sequenciais). Quando houver um bit 1 seguido de zeros observe que iremos substituir por "-1" (posição deste bit 1 encontrado) e por "1" (posição do bit zero encontrado).

Se utilizarmos radix-4, agrupamos esse resultado de dois em dois bits e achamos a versão do resultado em radix-4.

EXAMPLE MULTIPLICATION VIA MODIFIED BOOTH'S RECODING

a	0 1 1 0	
x	1 0 1 0	
z	$^-1$ $^-2$	Radix-4
$p^{(0)}$	0 0 0 0 0 0	
$+z_0a$	1 1 0 1 0 0	
$4p^{(1)}$	1 1 0 1 0 0	
$p^{(1)}$	1 1 1 1 0 1 0 0	
$+z_1a$	1 1 1 0 1 0	
$4p^{(2)}$	1 1 0 1 1 1 0 0	
$p^{(2)}$	1 1 0 1 1 1 0 0	

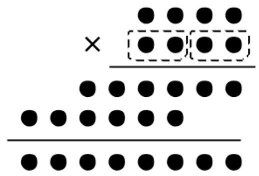


Fig. 10.5 Example of radix-4 multiplication with modified Booth's recoding of the 2's-complement multiplier.

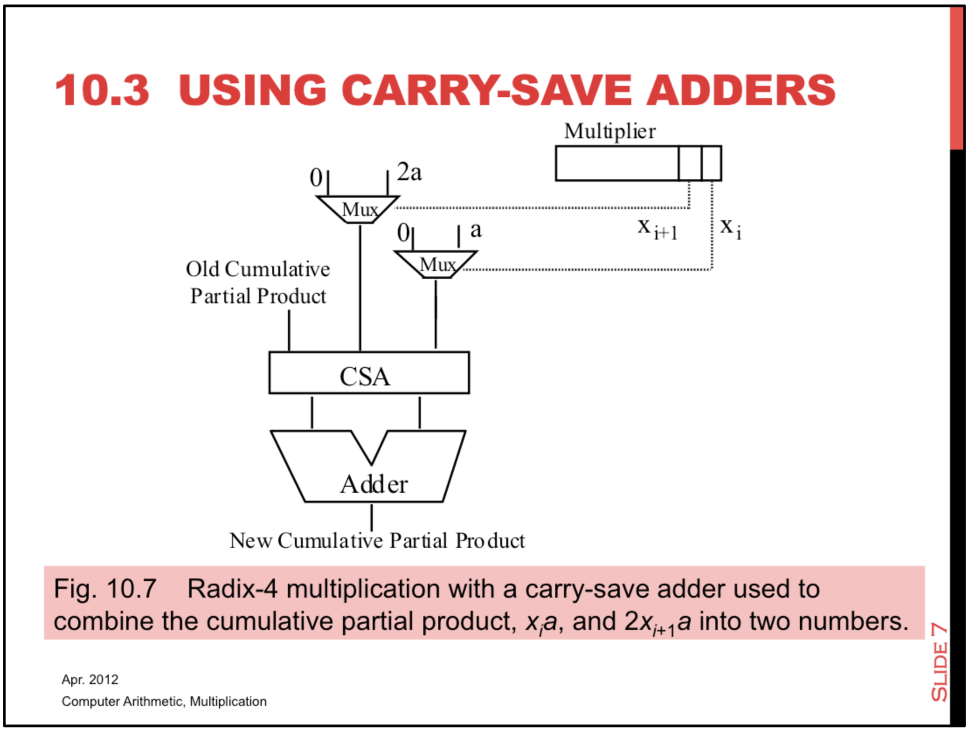
Apr. 2012
Computer Arithmetic, Multiplication

SLIDE 6

O processo de conversão de conjunto de dígitos definido pela recodificação de Radix-4 Booth não envolve propagação de carry.

Um exemplo de multiplicação de radix-4, usando a recodificação de Booth, é demonstrado na Fig. 10.5. O multiplicador de complemento de 2 de 4 bits x = (1010) é recodificado como um número radix-4 de 2 dígitos [z = (-1-2)], que por sua vez representarão os múltiplos $z_0 a$ = −2a e
$z_1 a$ = −a para ser adicionado ao produto parcial cumulativo em 2 ciclos.

Observe que em todas as etapas intermediárias, a metade superior do produto parcial cumulativo é estendida de 4 bits para 6 bits para acomodar a extensão de sinal necessária para o tratamento adequado dos valores negativos.



Somadores (CSAs) podem ser usados para reduzir o número de ciclos de adição e também tornar cada ciclo mais rápido dado o problema tratado. Por exemplo, a multiplicação de radix-4 sem recodificação de Booth pode ser implementada usando um CSA para lidar com o múltiplo 3a, como mostrado na Fig. 10.7. Veja que podemos ter a seleção de “2a” no primeiro mux e de “a” no segundo, que por sua vez, irão ser tratados como soma no CSA, podendo realizar (2a + a) mais um bit de carry do produto parcial anterior cumulativo.

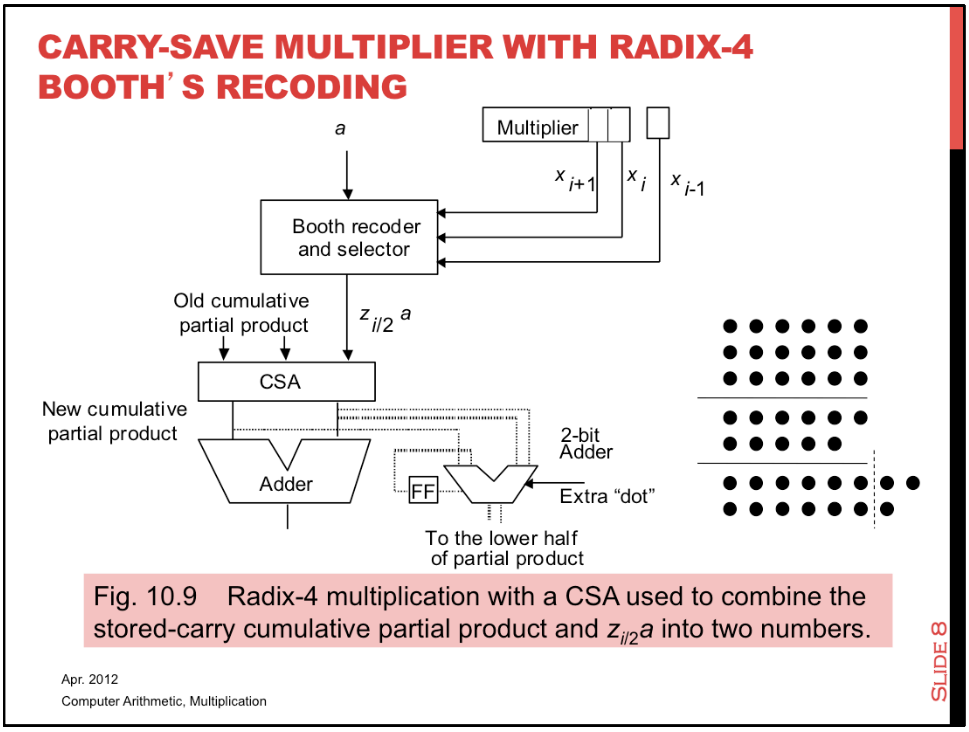
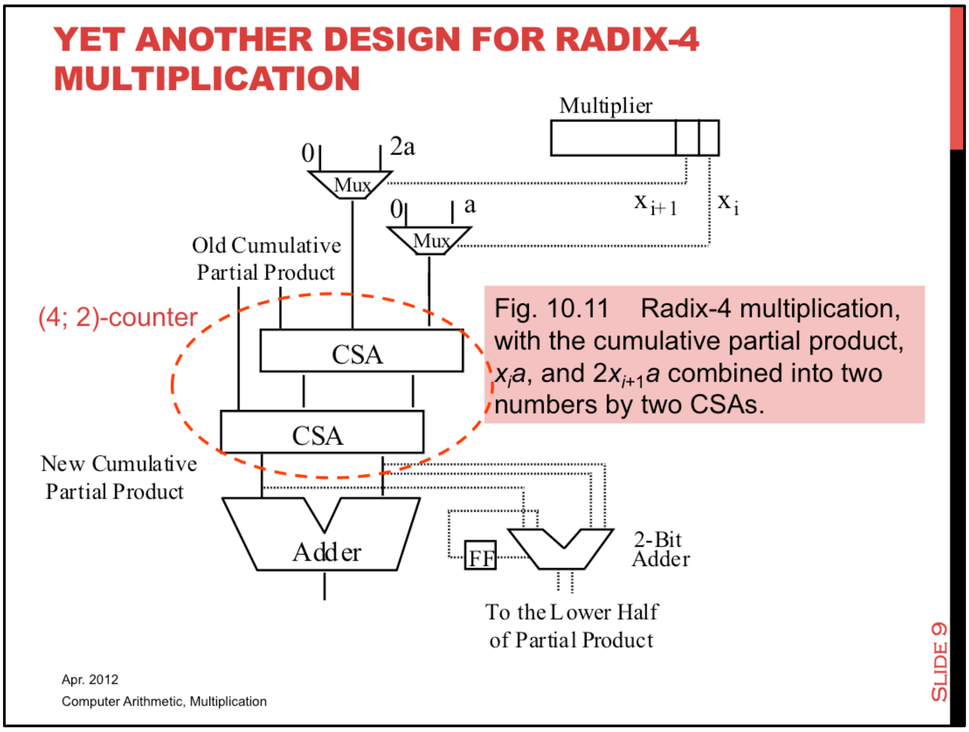


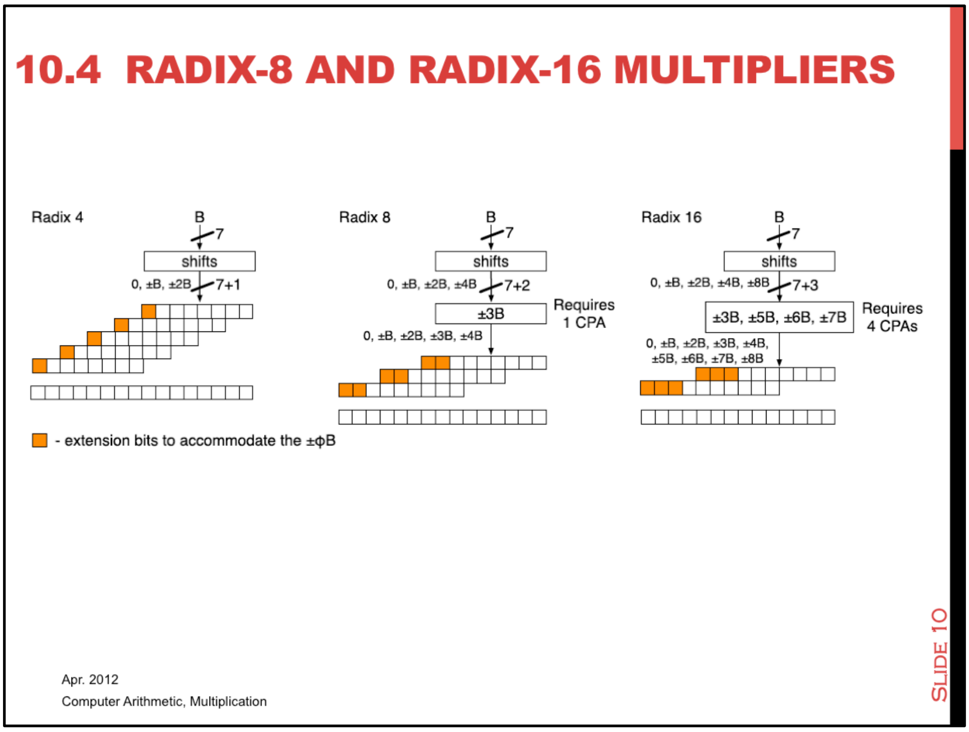
Fig. 10.9 Radix-4 multiplication with a CSA used to combine the stored-carry cumulative partial product and $z_{i/2}a$ into two numbers.

O design baseado em CSA pode ser combinado com a recodificação de Radix-4 de Booth para reduzir o número de ciclos em 50%, bem como tornando cada ciclo consideravelmente mais curto. Basta realizar a recodificação de Booth antes da entrada do CSA. Lembremos que com a recodificação não teremos mais um valor "3a", apenas multiplos de "±a" e "±2a".

Um somador de 2 bits é necessário para combinar 2 bits da soma, 1 bit de carry e um carry de um ciclo anterior para 2 bits que são deslocados para a metade inferior do registrador de produto parcial cumulativo (PP) e um carry que é mantido para o próximo ciclo.



De forma alternativa, pode-se ainda eliminar a recodificação de Booth e usar o esquema representado na Fig. 10.7 para acomodar o múltiplo 3a necessário. Agora, quatro números (a soma e os carries do produto parcial cumulativo, x_i a , e [2x_i a + 1a]) precisam ser combinados, sendo necessária uma árvore CSA de dois níveis (Fig. 10.11).



A partir do multiplicador de radix-4 da Fig. 10.11, seria possível visualizar multiplicadores de radix mais alto. Um multiplicador radix-8, por exemplo, pode ter uma árvore CSA de três níveis para combinar o produto parcial cumulativo com os três múltiplos x_ia , $2x_{i+1}$a e $4x_{i+2}$a em um novo produto parcial cumulativo.

Uma vez que passamos para três níveis de CSA, podemos também investir em mais um CSA para implementar um multiplicador radix-16 (ou 4 bits por vez).

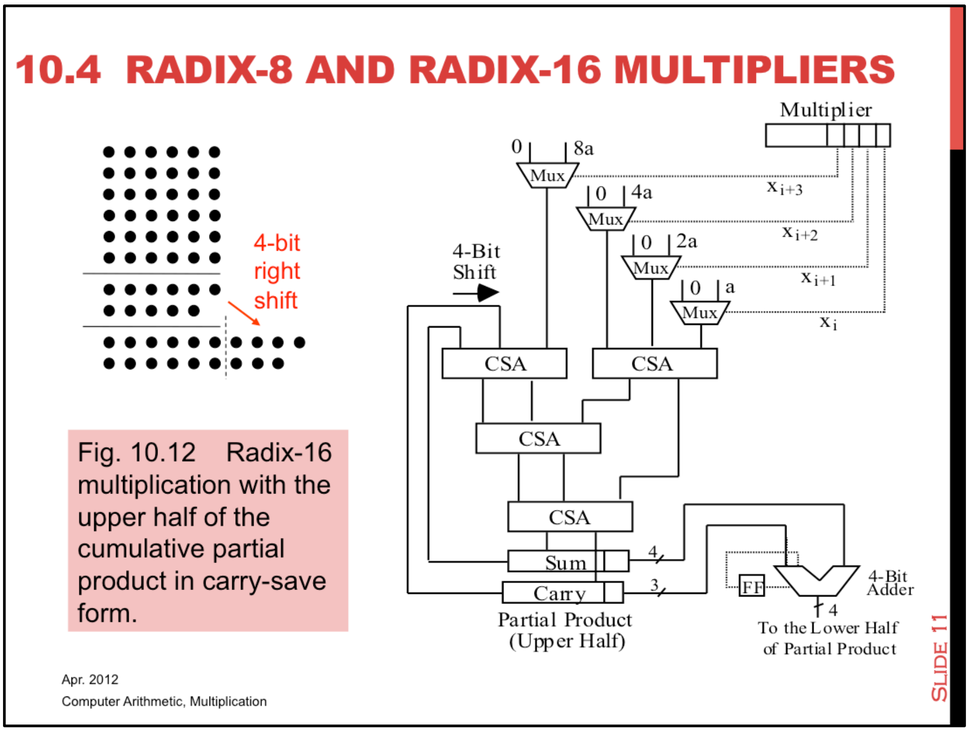
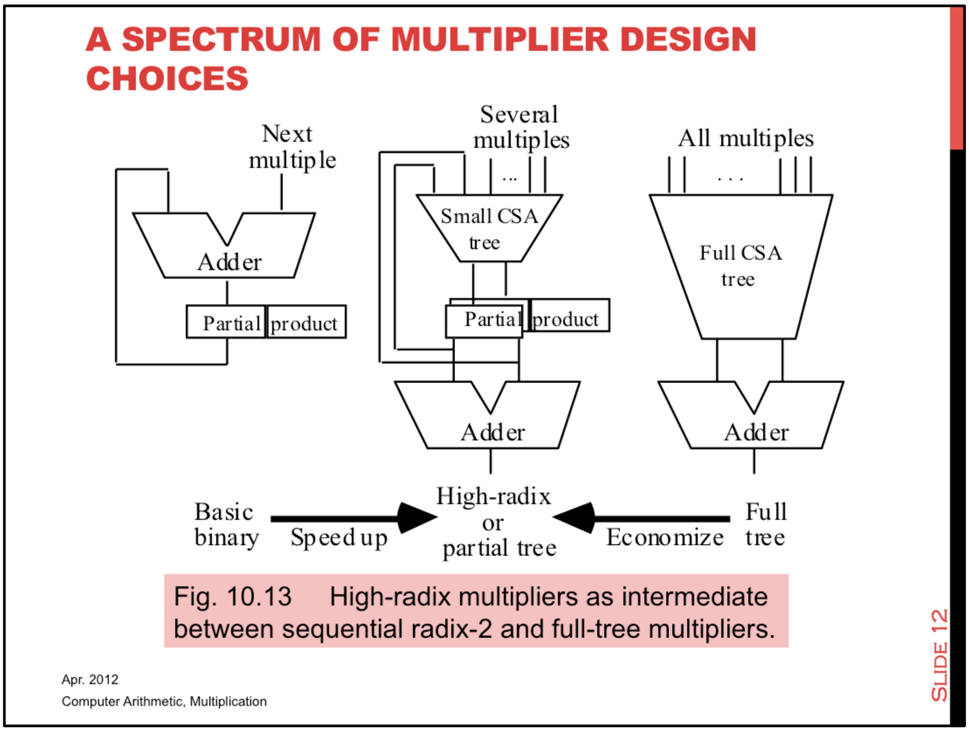


Fig. 10.12 Radix-16 multiplication with the upper half of the cumulative partial product in carry-save form.

Aqui vemos a implementação daquilo que foi comentado no slide anterior.



Projetos como os representados nas Figs. 10.11 e 10.12 podem ser vistos como intermediários entre a multiplicação sequencial básica (1 bit por vez) e multiplicadores de árvore totalmente paralelos, os quais serão discutidos no slide do Capítulo 11.

Assim, multiplicadores “high radix” podem ser vistos como projetos que oferecem aceleração sobre a multiplicação sequencial ou economia sobre multiplicadores de árvore totalmente paralelos (Fig. 10.13).

PROBLEMAS

Problema 10.1. Para uma multiplicação de dois operandos $A \times B$ de 24 bits, aplique o método e Radix-4, 8 e 16 determine o custo e caminho critico dos blocos considerando A_{FA} e T_{FA} como a área e atraso por *Full-Adder*, e $0{,}5 \times A_{FA}$ e $0{,}5 \times T_{FA}$, para o *Half-Adder*, $(a/2) \times A_{FA}$ e $(a/2) \times T_{FA}$ para o $(2^a{:}1)$ MUX.

Observação: Considere que a multiplicações $3 \times A$, $5 \times A$, $14 \times A$, $15 \times A$, $18 \times A$, $26 \times A$, e $44 \times A$ estão previamente computadas.

SLIDE 13

Gabarito no Moodle

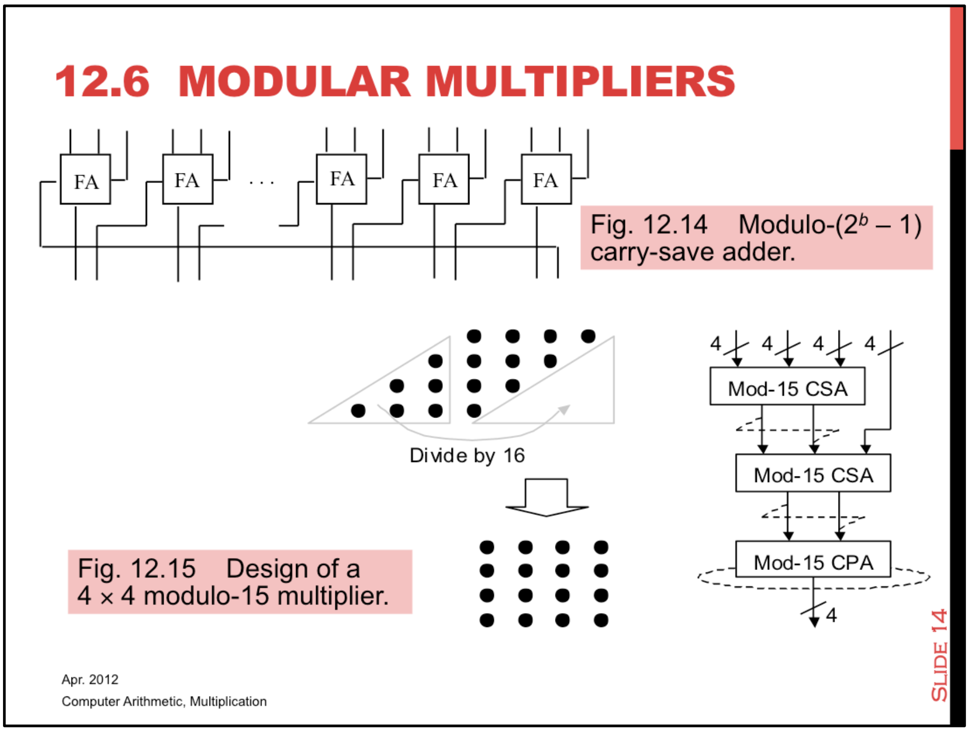
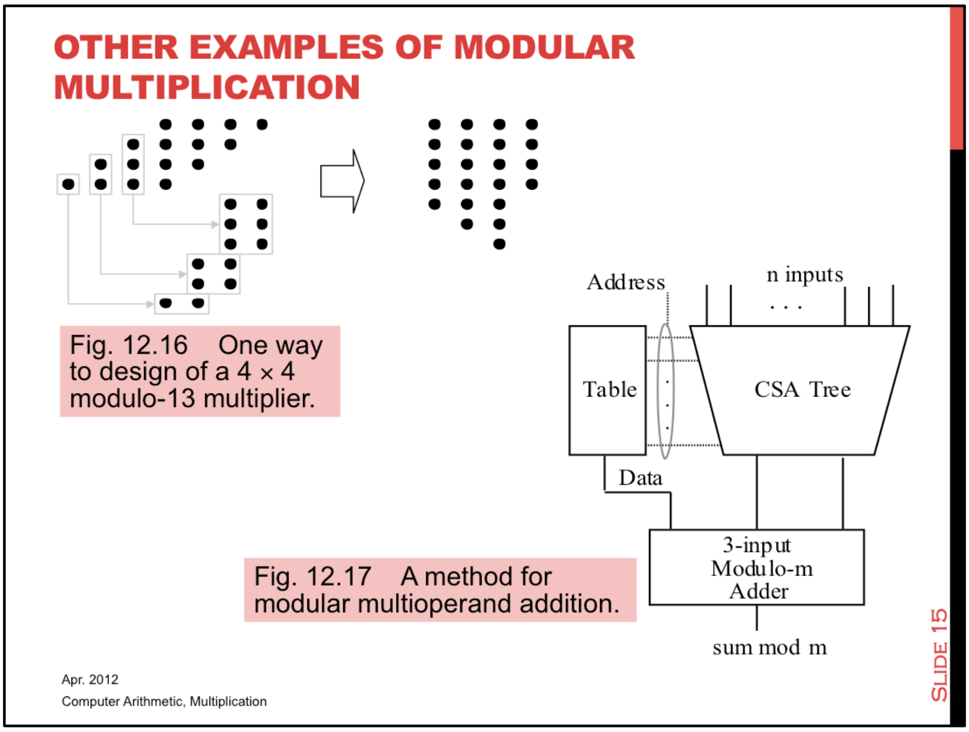


Fig. 12.14 Modulo-(2^b – 1) carry-save adder.

Fig. 12.15 Design of a 4 × 4 modulo-15 multiplier.

Agora veremos como implementar a operação de multiplicação para sistemas de numeração residual (RNS) (Visto no capítulo 4).

Como exemplo, considere o projeto de um multiplicador módulo 15 para operandos de 4 bits. Como 16 = 1 mod 15, os seis pontos mais significativos delimitados pelo triângulo cinza no canto esquerdo da Fig. 12.15 podem ser movidos conforme mostrado, levando à matriz quadrada de produtos parciais. Os quatro valores de 4 bits podem ser reduzidos em dois níveis de CSA (como o circuito de soma da Fig. 12.14) seguido por um somador de 4 bits. Vemos que este multiplicador modular particular é de fato mais simples do que um multiplicador binário 4 × 4 comum



Técnicas semelhantes podem ser usadas para lidar com outros casos de multiplicação modular. Por exemplo, um multiplicador módulo-13 pode ser projetado utilizando as identidades (16 = 3 mod 13), (32 = 6 mod 13) e (64 = 12 mod 13). Cada ponto dentro do triângulo na Fig. 12.15 deve agora ser substituído por dois pontos nas quatro colunas de ordem inferior (Fig. 12.16).
Pode-se perceber então que neste caso adiciona-se alguma complexidade tendo em vista o maior número de pontos a serem reduzidos e a necessidade de o ajustes finais do resultado.

Para completar o projeto deste multiplicador 4 × 4 módulo-13, os valores mostrados no lado direito da Fig. 12.16 devem ser adicionados módulo 13. É realizada uma pequena simplificação, consistindo em remover um ponto da coluna 1 e substituí-lo por dois pontos na coluna 0.

Então, para construção deste circuito devemos manter alguns dos bits que emergem da extremidade esquerda (por exemplo, aqueles que não podem ser acomodados na matriz de ponto sem aumentar sua altura) e reduzi-los módulo 13 por meio de uma "lookup table" ou circuito lógico especialmente projetado.

PROBLEMAS

Problema 10.2. Projete a estrutura do multiplicador RNS para os seguintes módulos:

a) 29;

b) 31;

c) 13.

SLIDE 16

Gabarito no Moodle