# 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

# 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 |

# 10.1 RADIX-4 MULTIPLICATION

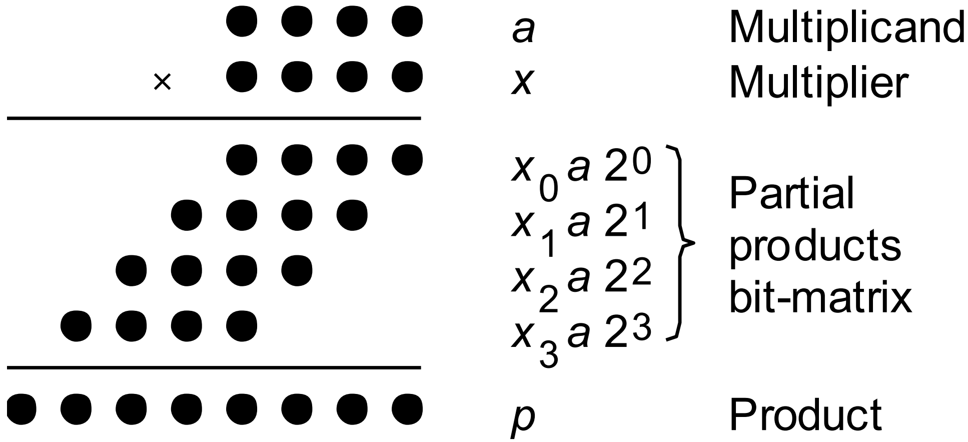
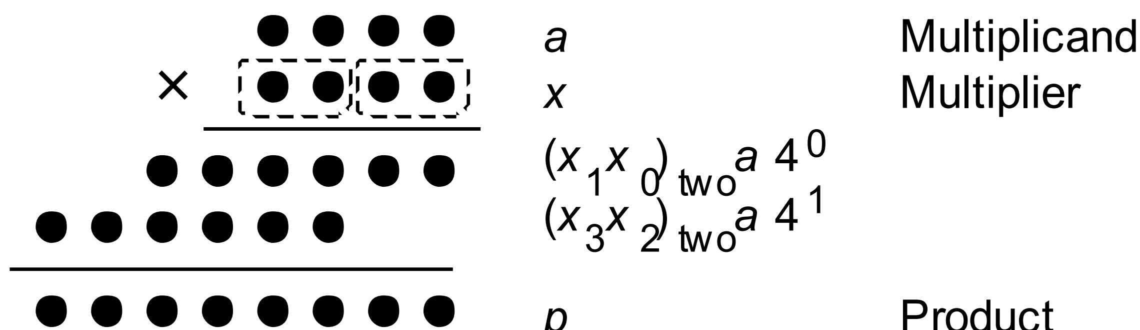



Fig. 9.1

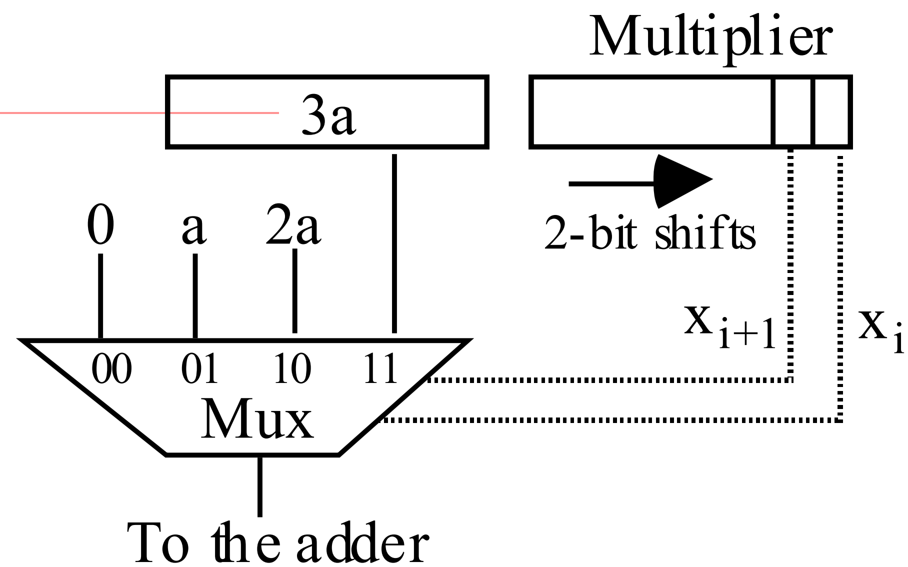
Fig. 10.1 Radix-4, or two-bit-at-a-time, multiplication in dot notation

Number of cycles is halved, but now the "difficult" multiple 3*a* must be dealt with



# A POSSIBLE DESIGN FOR A RADIX-4 MULTIPLIER

Precomputed via shift-and-add ($3a = 2a + a$)




$k/2 + 1$ cycles, rather than $k$

One extra cycle over $k/2$ not too bad, but we would like to avoid it if possible

Solving this problem for radix 4 may also help when dealing with even higher radices

Fig. 10.2 The multiple generation part of a radix-4 multiplier with precomputation of $3a$.

# 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 $z$ |

# 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 | |
| ================================ | | |

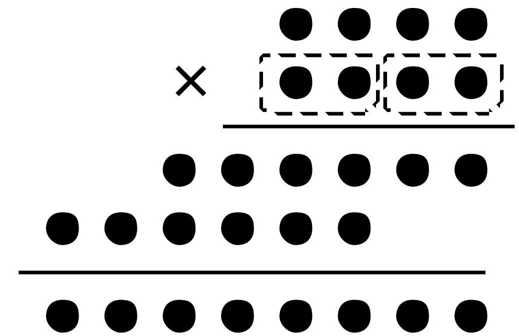



Fig. 10.5 Example of radix-4 multiplication with modified Booth' s recoding of the 2' s-complement multiplier.

# 10.3 USING CARRY-SAVE ADDERS

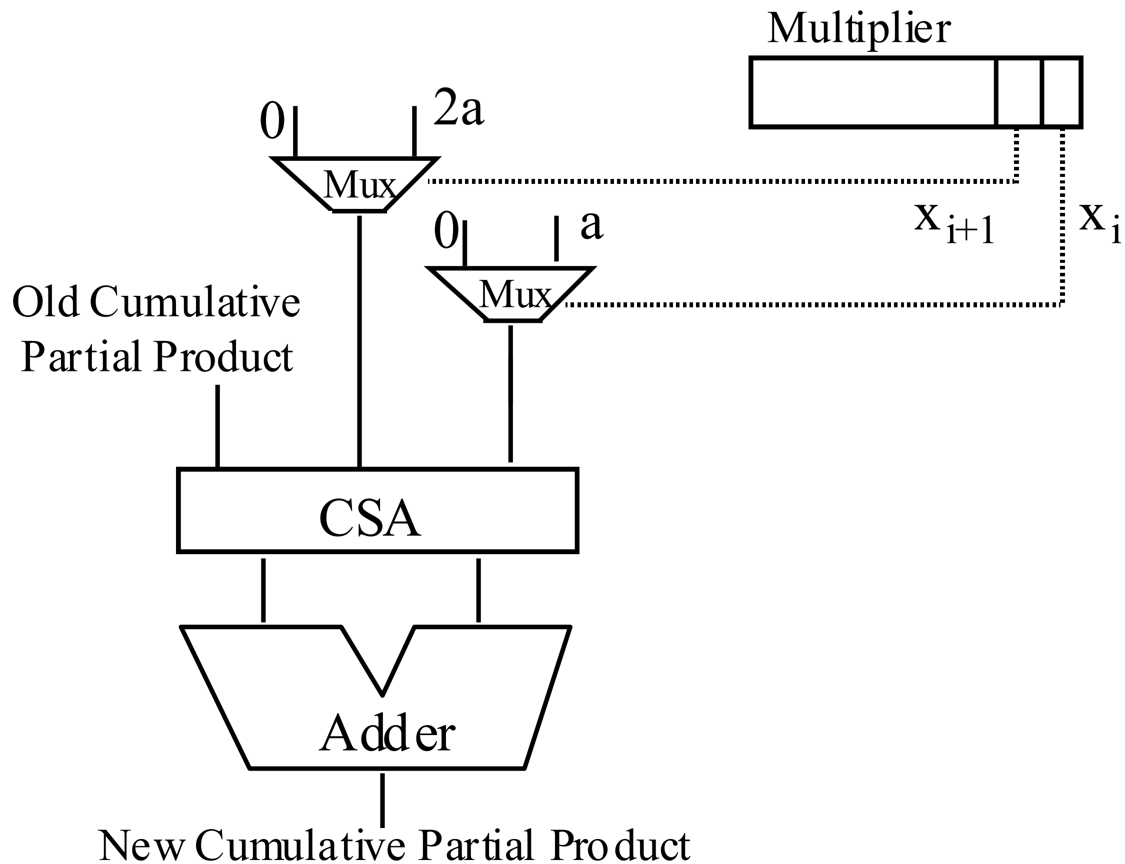



Fig. 10.7 Radix-4 multiplication with a carry-save adder used to combine the cumulative partial product, $x_i a$, and $2x_{i+1}a$ into two numbers.

# CARRY-SAVE MULTIPLIER WITH RADIX-4 BOOTH'S RECODING

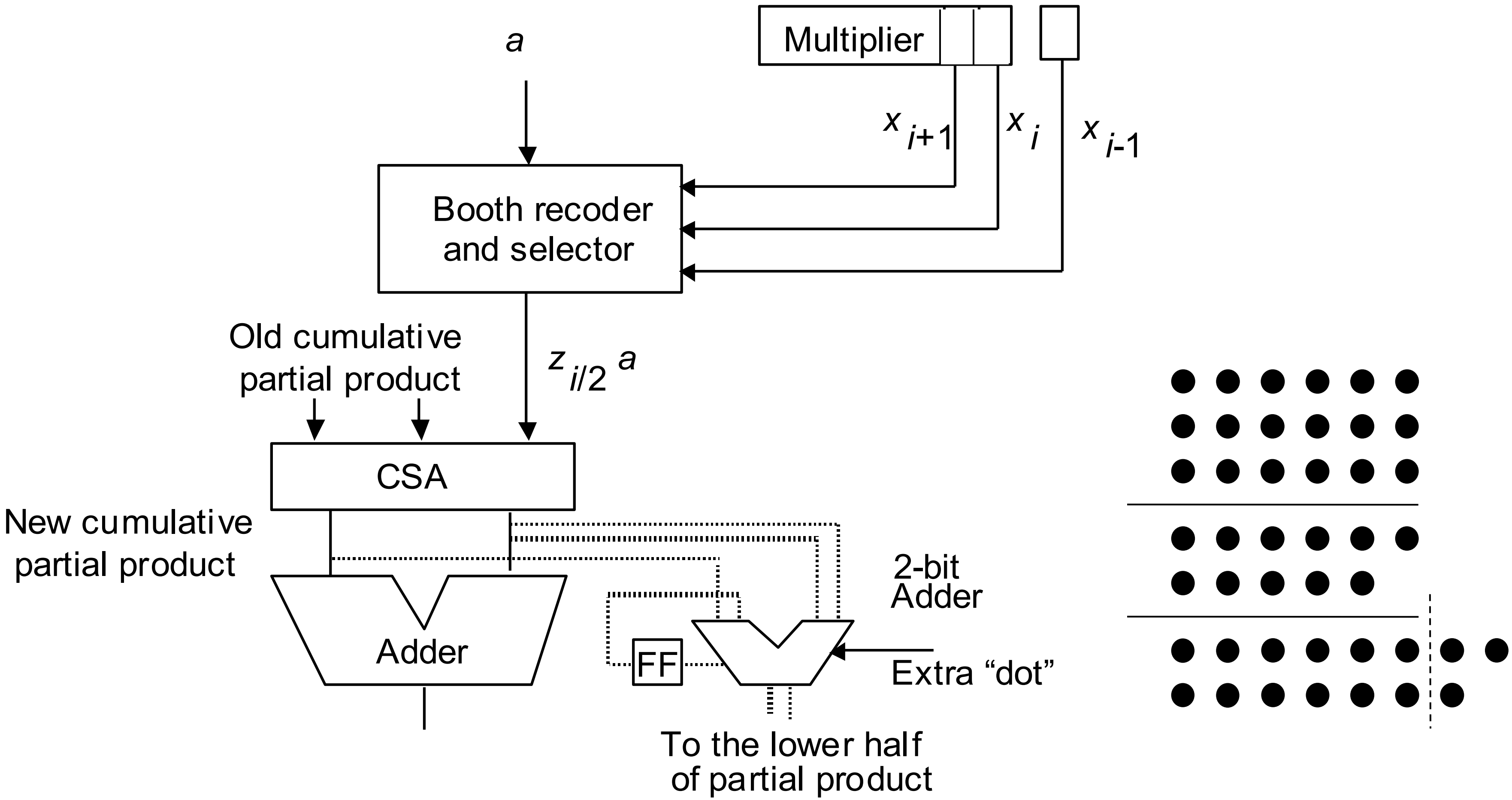



Fig. 10.9 Radix-4 multiplication with a CSA used to combine the stored-carry cumulative partial product and $z_{i/2}a$ into two numbers.

# YET ANOTHER DESIGN FOR RADIX-4 MULTIPLICATION

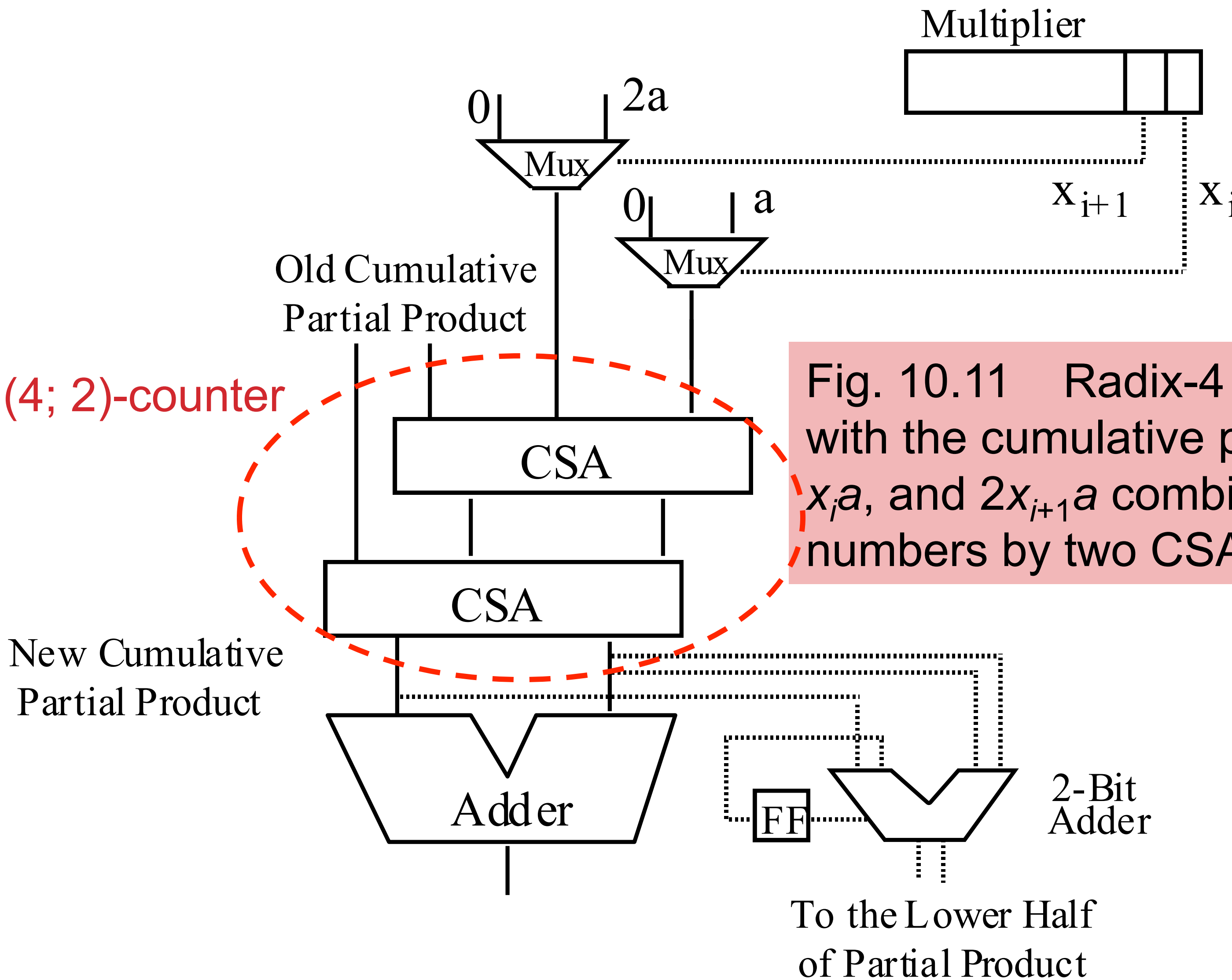
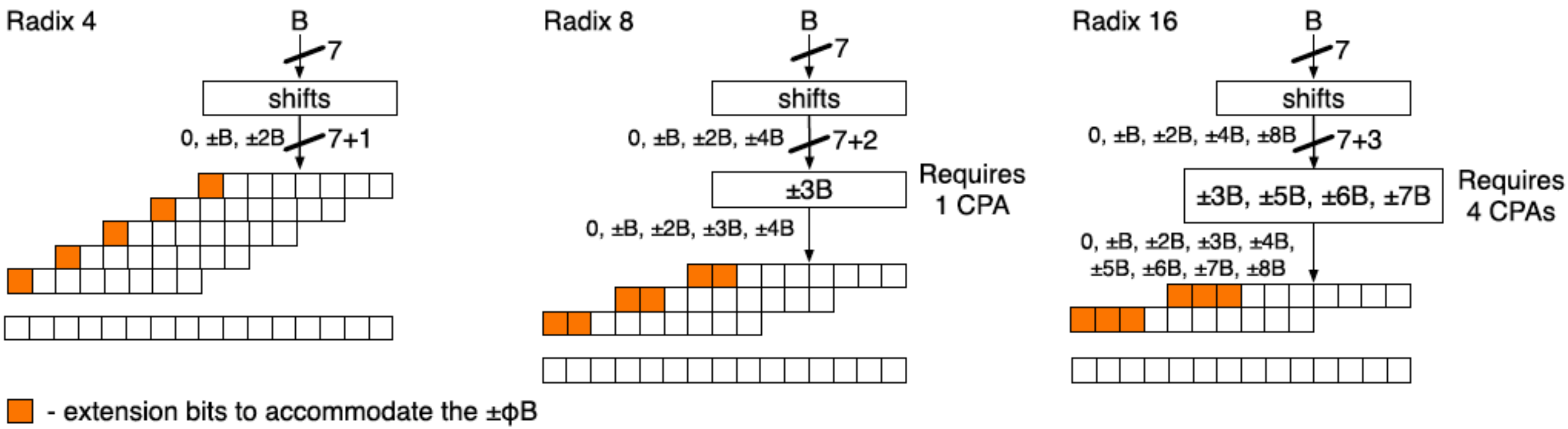
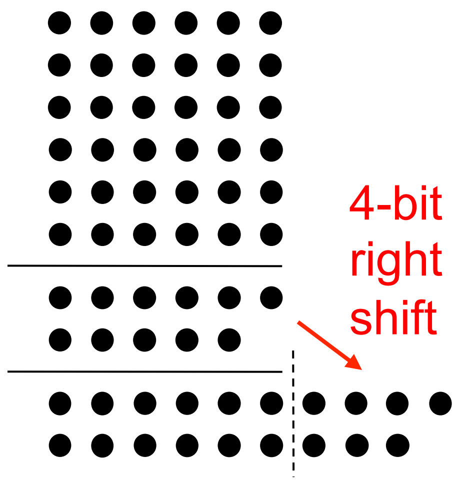



Fig. 10.11 Radix-4 multiplication, with the cumulative partial product, $x_i a$, and $2x_{i+1}a$ combined into two numbers by two CSAs.

# 10.4 RADIX-8 AND RADIX-16 MULTIPLIERS



# 10.4 RADIX-8 AND RADIX-16 MULTIPLIERS



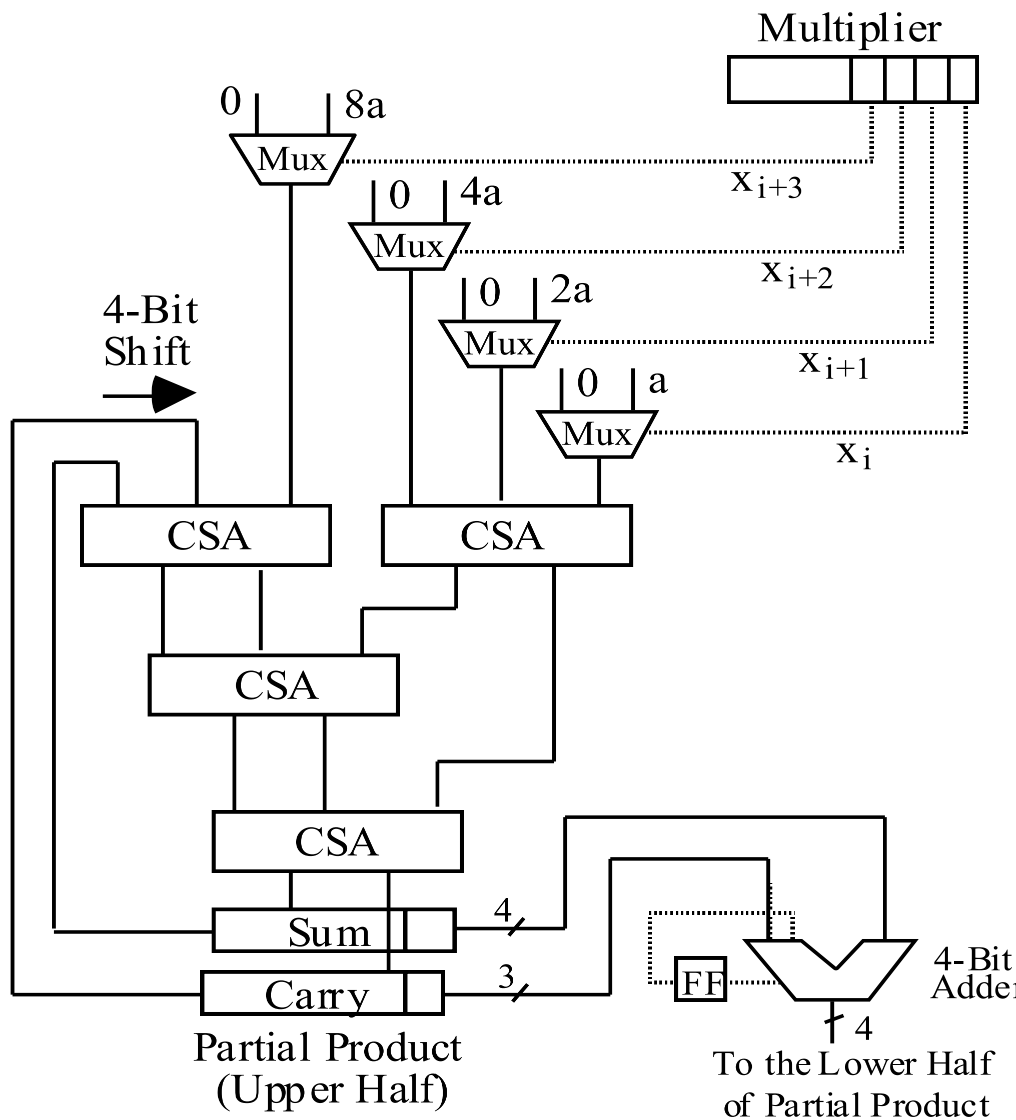




Fig. 10.12 Radix-16 multiplication with the upper half of the cumulative partial product in carry-save form.

# A SPECTRUM OF MULTIPLIER DESIGN CHOICES

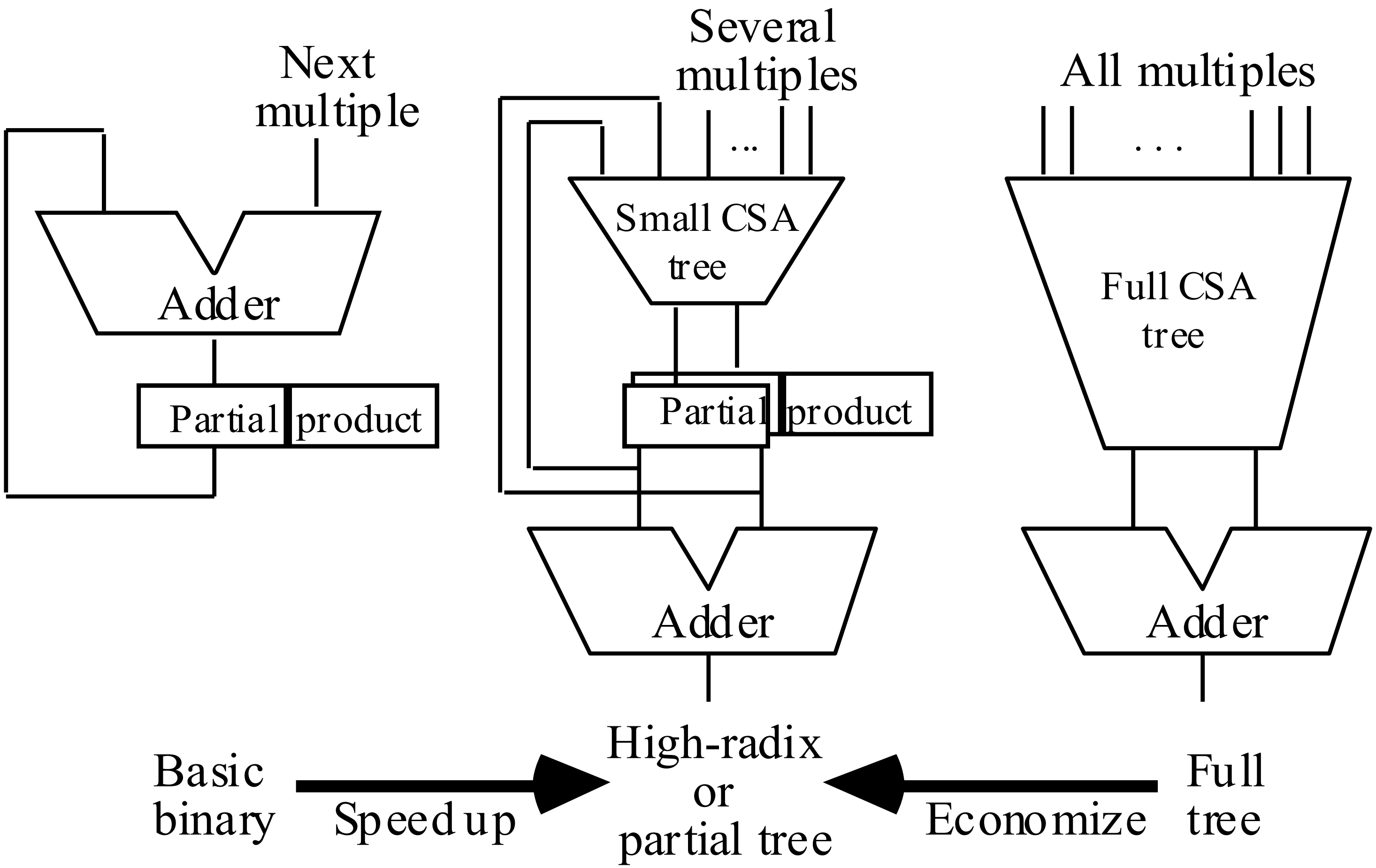



Fig. 10.13 High-radix multipliers as intermediate between sequential radix-2 and full-tree multipliers.

# 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.

# 12.6 MODULAR MULTIPLIERS

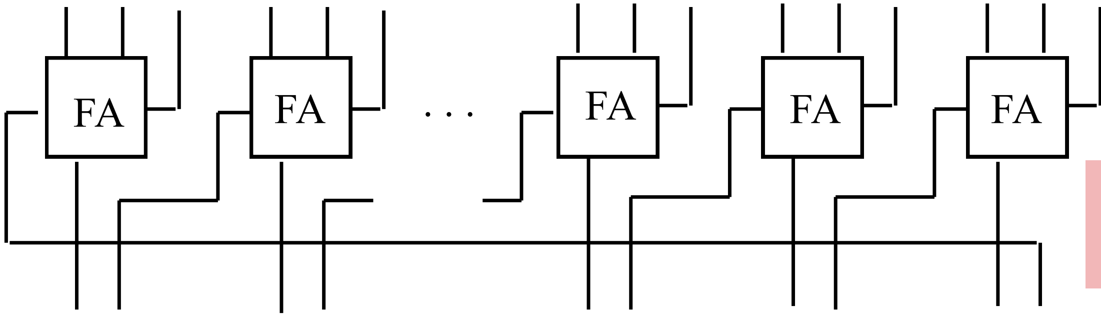



Fig. 12.14 Modulo-($2^b$ – 1) carry-save adder.

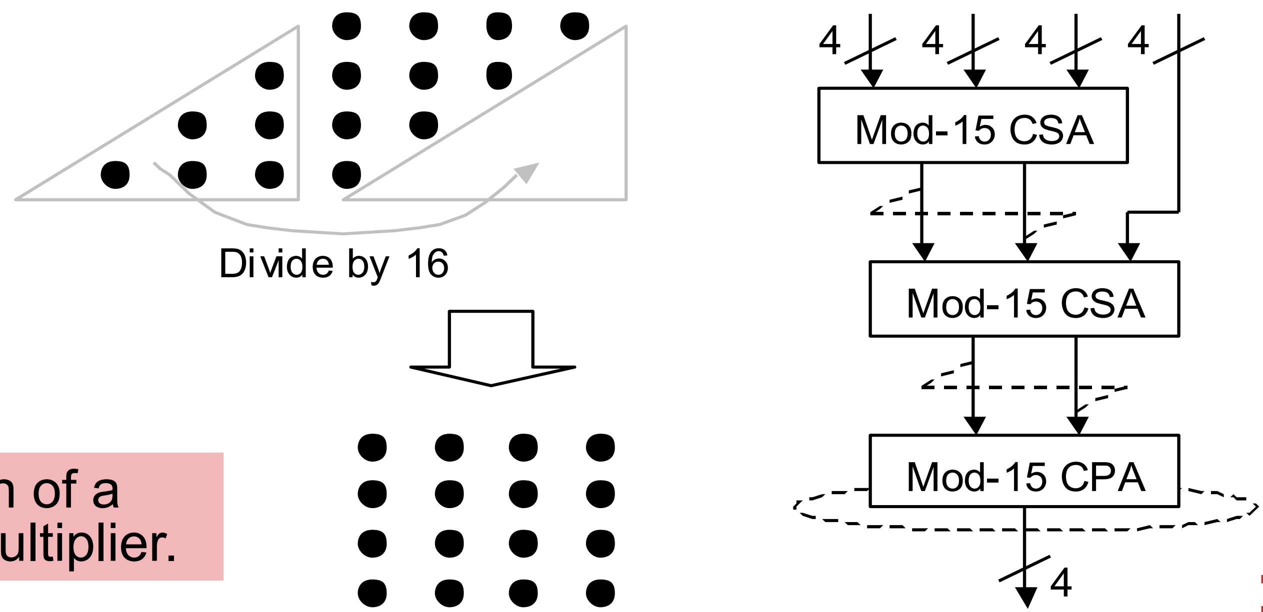



Fig. 12.15 Design of a 4 × 4 modulo-15 multiplier.

# OTHER EXAMPLES OF MODULAR MULTIPLICATION

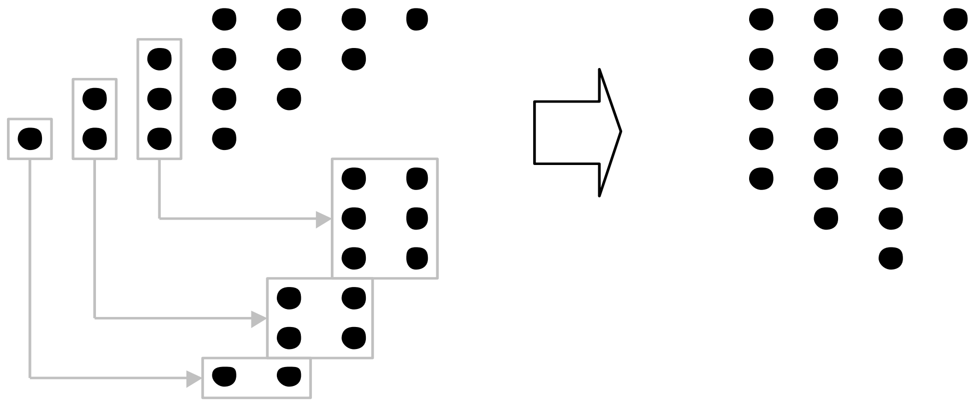


Fig. 12.16 One way to design of a 4 × 4 modulo-13 multiplier.

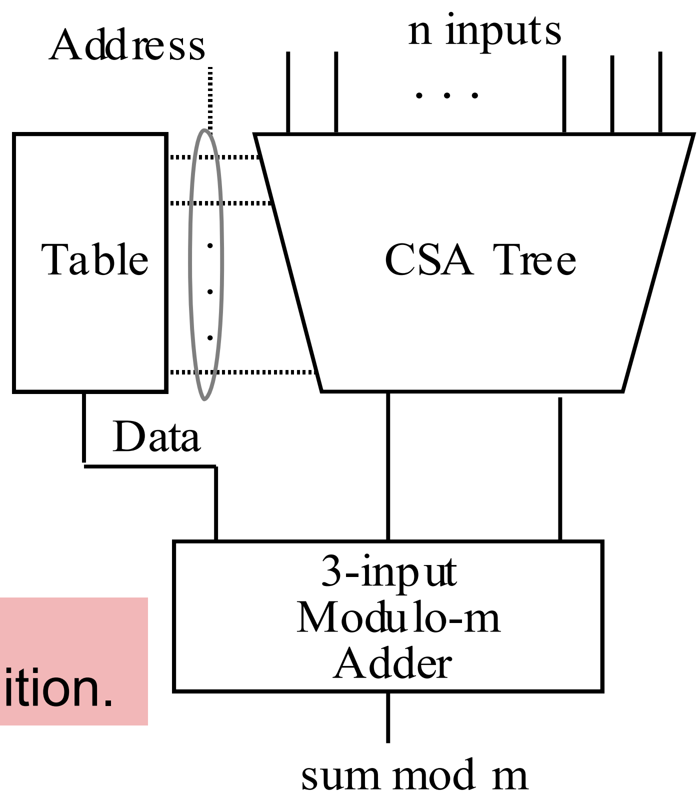



Fig. 12.17 A method for modular multioperand addition.

# PROBLEMAS

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

a) 29;

b) 31;

c) 13.