SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COURSE NAME :19IT301 COMPUTER ORGANIZATION AND ARCHITECTURE

II YEAR /III SEMESTER

Unit – **Arithmetic operations**

Topic 4 : Signed operand multiplication

Signed Multiplication

Considering 2’s-complement signed operands, what will happen to (-13)×(+11) if following the same method of unsigned multiplication?

					1	0	0	1	1	(- 13)
					0	1	0	1	1	(+ 11)
1	1	1	1	1	1	0	0	1	1	
1	1	1	1	1	0	0	1	1		
0	0	0	0	0	0	0	0			
1	1	1	0	0	1	1				
0	0	0	0	0	0					
1	1	0	1	1	1	0	0	0	1	(- 143)

Sign extension is shown in blue

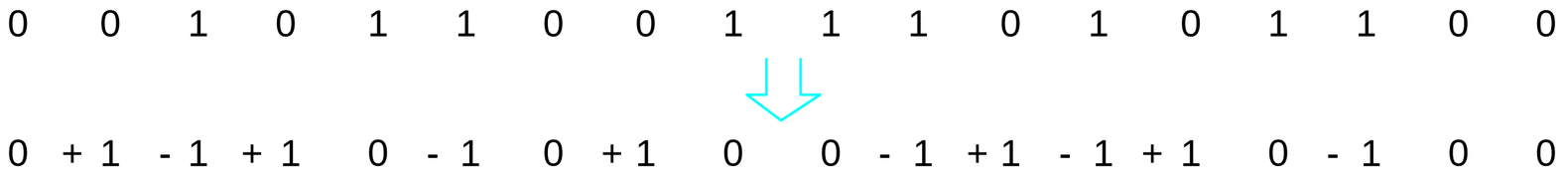
Sign extension of negative multiplicand.

Signed Multiplication

For a negative multiplier, a straightforward solution is to form the 2’s-complement of both the multiplier and the multiplicand and proceed as in the case of a positive multiplier.
This is possible because complementation of both operands does not change the value or the sign of the product.
A technique that works equally well for both negative and positive multipliers – Booth algorithm.

Booth Algorithm

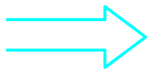
In general, in the Booth scheme, -1 times the shifted multiplicand is selected when moving from 0 to 1, and +1 times the shifted multiplicand is selected when moving from 1 to 0, as the multiplier is scanned from right to left.



Booth recoding of a multiplier.

Booth Algorithm

```
    0  1  1  0  1   (+13)
X   1  1  0  1  0   (- 6)
```



```
                    0  1  1  0  1
                    0 -1 +1 -1  0
                 ------------------
0  0  0  0  0    0  0  0  0  0
1  1  1  1  1    0  0  1  1
0  0  0  0  1    1  0  1
1  1  1  0  0    1  1
0  0  0  0  0    0
-------------------------------------
1  1  1  0  1  1  0  0  1  0   (- 78)
```

Booth multiplication with a negative multiplier.

Booth Algorithm

Multiplier		Version of multiplicand selected by bit *i*
Bit *i*	Bit *i*-1	
0	0	0 x M
0	1	+ 1 x M
1	0	– 1 x M
1	1	0 x M

Booth multiplier recoding table.

Booth Algorithm

Best case – a long string of 1’s (skipping over 1s)
Worst case – 0’s and 1’s are alternating

Worst-case multiplier

0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
+1	-1	+1	-1	+1	-1	+1	-1	+1	-1	+1	-1	+1	-1	+1	-1

Ordinary multiplier

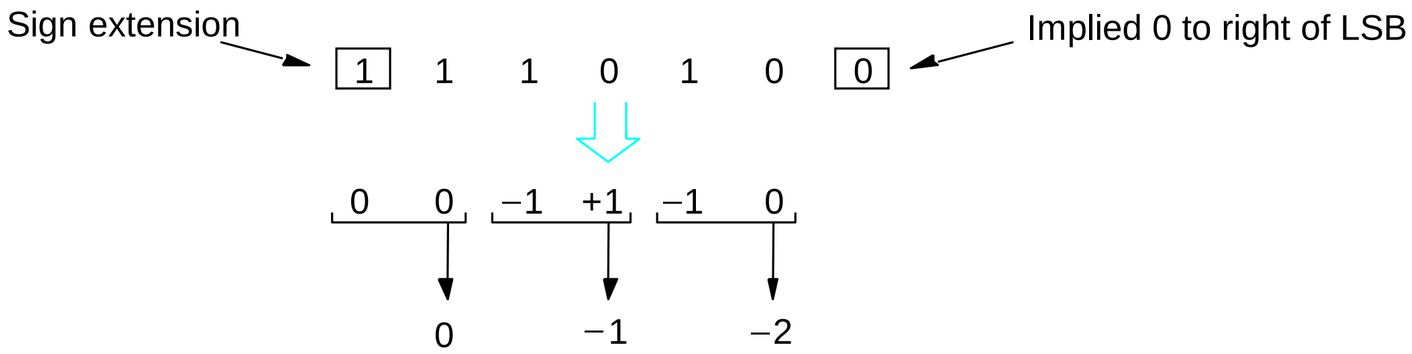
1	1	0	0	0	1	0	1	1	0	1	1	1	1	0	0
0	-1	0	0	+1	-1	+1	0	-1	+1	0	0	0	-1	0	0

Good multiplier

0	0	0	0	1	1	1	1	1	0	0	0	0	1	1	1
0	0	0	+1	0	0	0	0	-1	0	0	0	+1	0	0	-1

Bit-Pair Recoding of Multipliers

Bit-pair recoding halves the maximum number of summands (versions of the multiplicand).



(a) Example of bit-pair recoding derived from Booth recoding

Bit-Pair Recoding of Multipliers

Multiplier bit-pair		Multiplier bit on the right	Multiplicand selected at position i
$i+1$	i	$i-1$	
0	0	0	0 X M
0	0	1	+1 X M
0	1	0	+1 X M
0	1	1	+2 X M
1	0	0	−2 X M
1	0	1	−1 X M
1	1	0	−1 X M
1	1	1	0 X M

(b) Table of multiplicand selection decisions

Bit-Pair Recoding of Multipliers

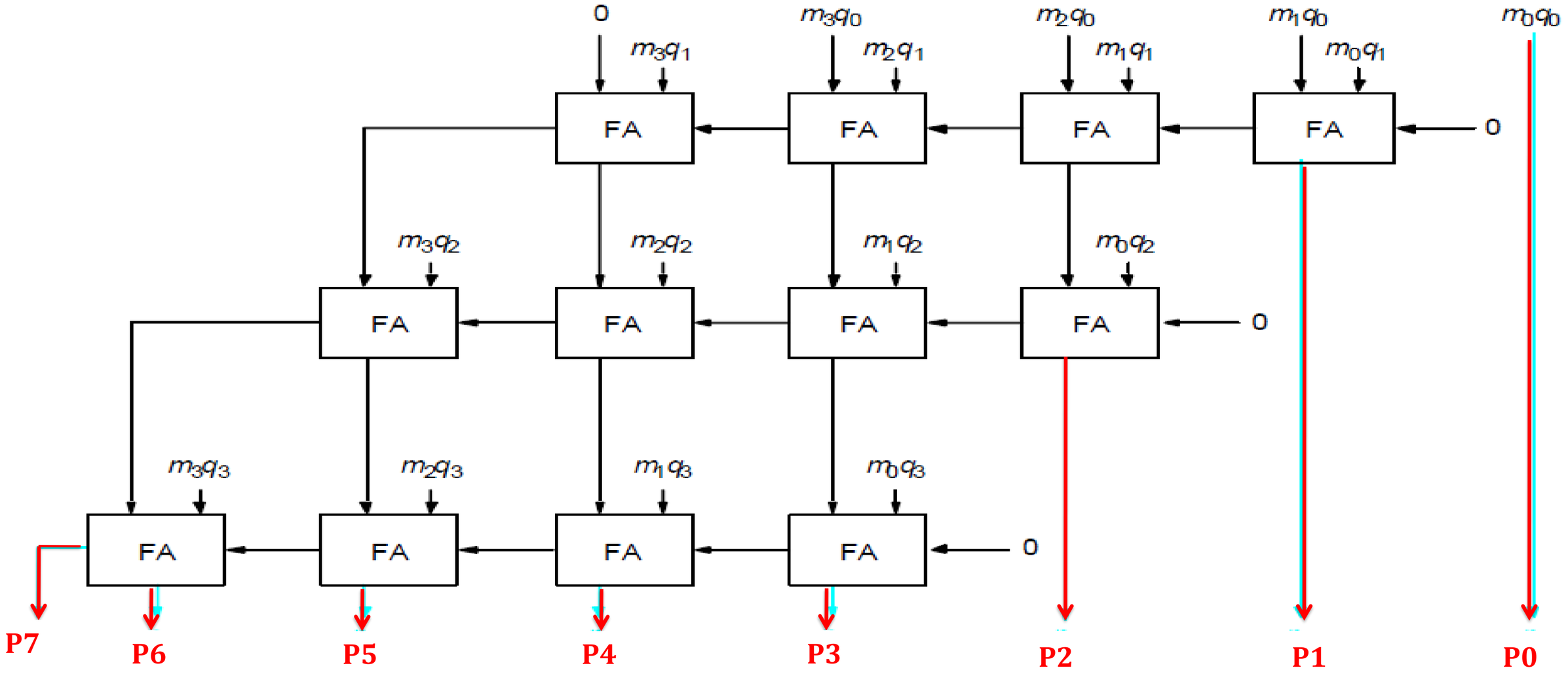
	0	1	1	0	1	(+13)
×	1	1	0	1	0	(- 6)

					0	1	1	0	1		
					0	-1	+1	-1	0		
0	0	0	0	0	0	0	0	0	0		
1	1	1	1	1	0	0	1	1			
0	0	0	0	1	1	0	1				
1	1	1	0	0	1	1					
0	0	0	0	0	0						
1	1	1	0	1	1	0	0	1	0		(- 78)

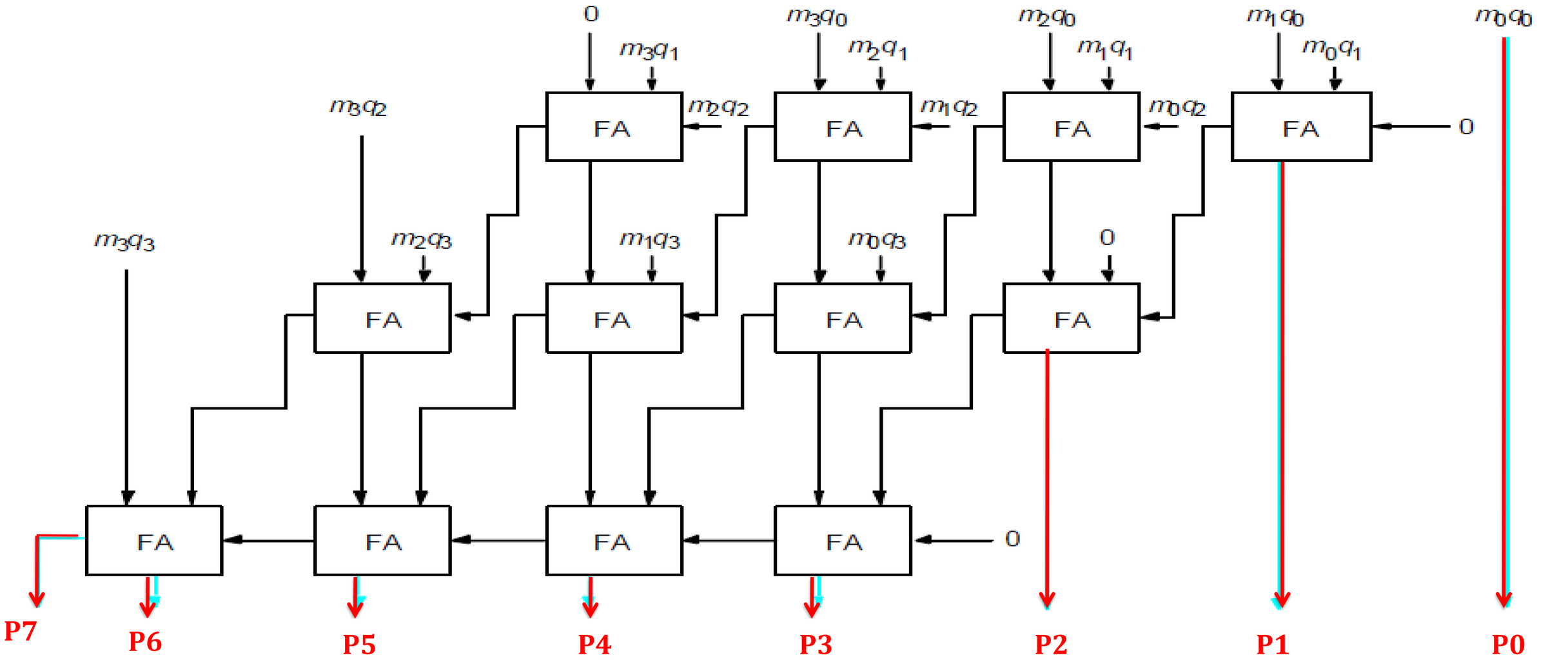
					0	1	1	0	1	
					0		-1		-2	
1	1	1	1	1	0	0	1	1	0	
1	1	1	1	0	0	1	1			
0	0	0	0	0	0					
1	1	1	0	1	1	0	0	1	0	

Figure 6.15. Multiplication requiring only *n*/2 summands.

Ripple carry array



Carry-Save Addition of Summands(Cont.,)



Carry-Save Addition of Summands(Cont.,)

Consider the addition of many summands, we can:

- Group the summands in threes and perform carry-save addition on each of these groups in parallel to generate a set of S and C vectors in one full-adder delay
- Group all of the S and C vectors into threes, and perform carry-save addition on them, generating a further set of S and C vectors in one more full-adder delay
- Continue with this process until there are only two vectors remaining
- They can be added in a RCA or CLA to produce the desired product

Carry-Save Addition of Summands

						1	0	1	1	0	1	(45)	M
					x	1	1	1	1	1	1	(63)	Q
						1	0	1	1	0	1	A	
					1	0	1	1	0	1		B	
				1	0	1	1	0	1			C	
			1	0	1	1	0	1				D	
		1	0	1	1	0	1					E	
	1	0	1	1	0	1						F	
1	0	1	1	0	0	0	1	0	0	1	1	(2,835)	Product

Figure 6.17. A multiplication example used to illustrate carry-save addition as shown in Figure 6.18.

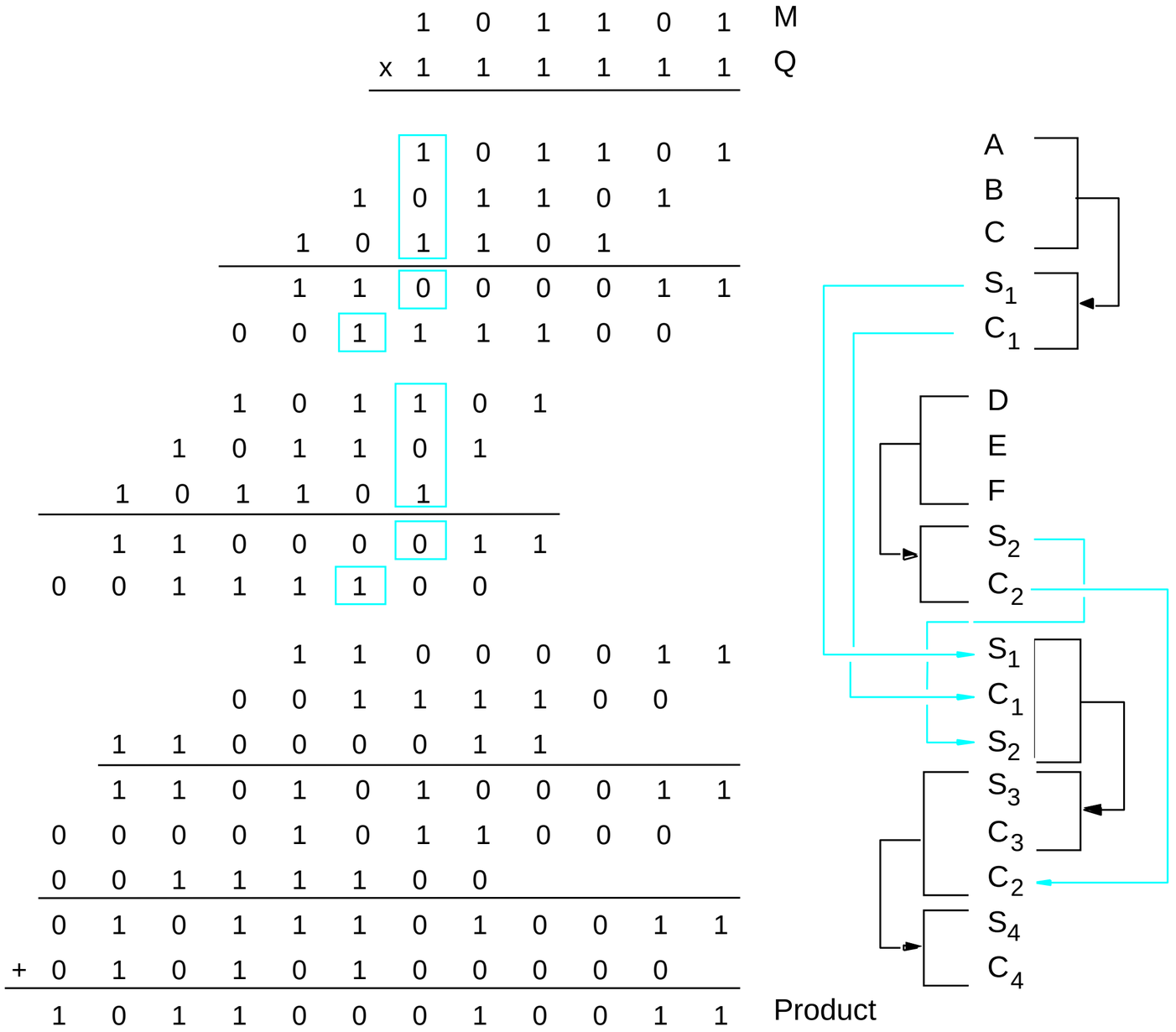


Figure 6.18. The multiplication example from Figure 6.17 performed using carry-save addition.

Assessment

a). What is Booth Algorithm?

_____

_____

_____

b) Mention the purpose of

1.Bit pair recoding.

2.Booth algorithm

Reference

1. Carl Hamacher, Zvonko Vranesic and Safwat Zaky, “Computer Organization”, McGraw-Hill, 6th Edition 2012.
2. David A. Patterson and John L. Hennessey, “Computer organization and design”, MorganKauffman /Elsevier, 5th edition, 2014.
3. William Stallings, “Computer Organization and Architecture designing for Performance”, Pearson Education 8th Edition, 2010
4. John P.Hayes, “Computer Architecture and Organization”, McGraw Hill, 3rd Edition, 2002
5. M. Morris R. Mano “Computer System Architecture” 3rd Edition 2007