ELECTRICAL ENGINEERING
UNIVERSITY *of* WASHINGTON

Lecture 15: Multipliers

Based on material prepared by prof. Visvesh S. Sathe

ELECTRICAL & COMPUTER ENGINEERING

Acknowledgements

All class materials (lectures, assignments, etc.) based on material prepared by Prof. Visvesh S. Sathe, and reproduced with his permission

Visvesh S. Sathe
Associate Professor
Georgia Institute of Technology
https://psylab.ece.gatech.edu

UW (2013-2022)
GaTech (2022-present)

Roadmap for today

- Unsigned multiplication
- Signed multiplication (Baugh-Wooley)
- Booth encoding to improve performance
- Prefix computation…. **again?!!**

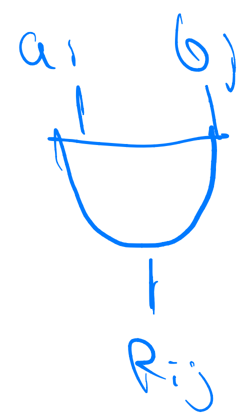
Multipliers

- How do you multiply binary numbers?

```
      1 0 1 1
        1 0 1
    ---------
      1 0 1 1
    0 0 0 0
  1 0 1 1
-------------
  1 1 0 1 1 1
```

```
  11
x  5
----

----
  55
```



1	1	1
1	0	0
0	1	0
0	0	0

- Multiplication
  - Generate partial products
  - Sum partial products together
  - M X N multiplication → M+N bit result

Multiplication

- Given multiplicand Y = $y_{n-1}\ y_{n-2}\ y_{n-3}\ \ldots\ y_1\ y_0$
- Given multiplicand X = $x_{m-1}\ x_{n-2}\ x_{n-3}\ \ldots\ x_1\ x_0$
- $X \times Y = \left(\sum_{i=0}^{m-1} x_i 2^i\right)\left(\sum_{j=0}^{n-1} y_j 2^j\right) = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} x_i y_j 2^{i+j}$
- In the case m=5, n=5

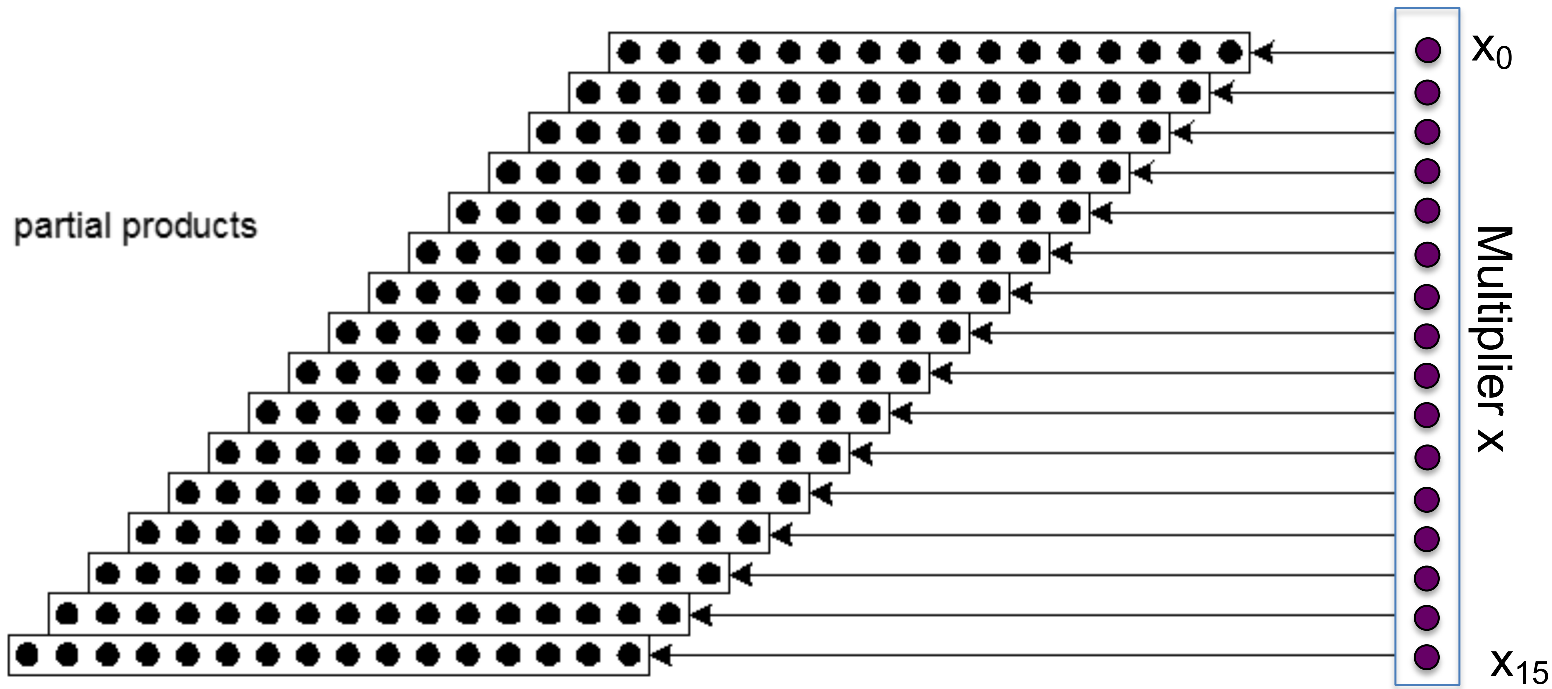
						y_5	y_4	y_3	y_2	y_1	y_0	Multiplicand
						x_5	x_4	x_3	x_2	x_1	x_0	Multiplier
						x_0y_5	x_0y_4	x_0y_3	x_0y_2	x_0y_1	x_0y_0	Partial Products
					x_1y_5	x_1y_4	x_1y_3	x_1y_2	x_1y_1	x_1y_0		
				x_2y_5	x_2y_4	x_2y_3	x_2y_2	x_2y_1	x_2y_0			
			x_3y_5	x_3y_4	x_3y_3	x_3y_2	x_3y_1	x_3y_0				
		x_4y_5	x_4y_4	x_4y_3	x_4y_2	x_4y_1	x_4y_0					
	x_5y_5	x_5y_4	x_5y_3	x_5y_2	x_5y_1	x_5y_0						
p_{11}	p_{10}	p_9	p_8	p_7	p_6	p_5	p_4	p_3	p_2	p_1	p_0	Products

Source: Weste-Harris

Dot Diagram

y $y_2 y_1 y_0$

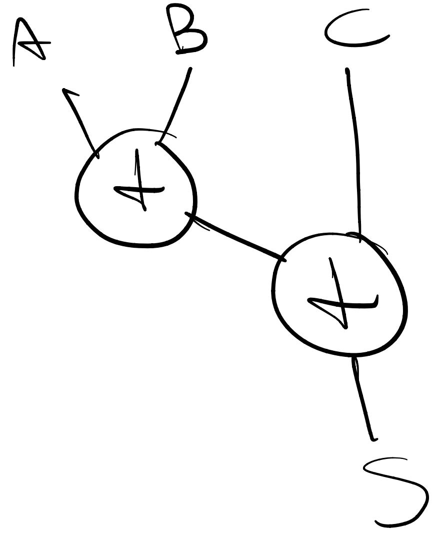
× x $x_2 x_1 x_0$

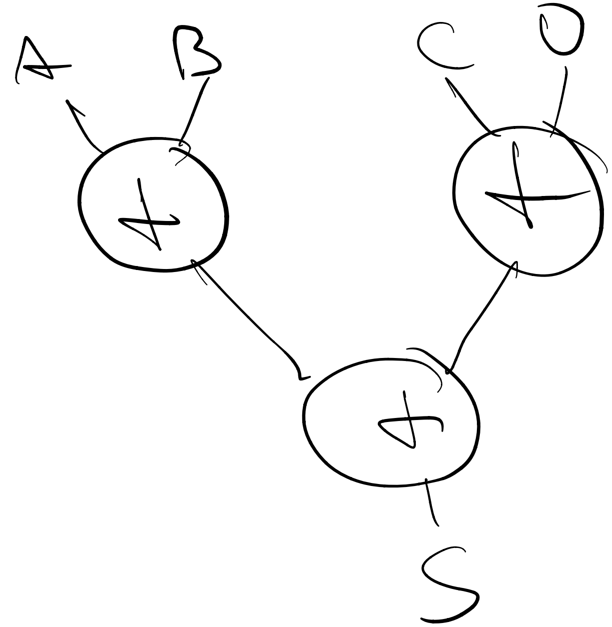


Source: Weste-Harris

- Each "●" represents a bit-product

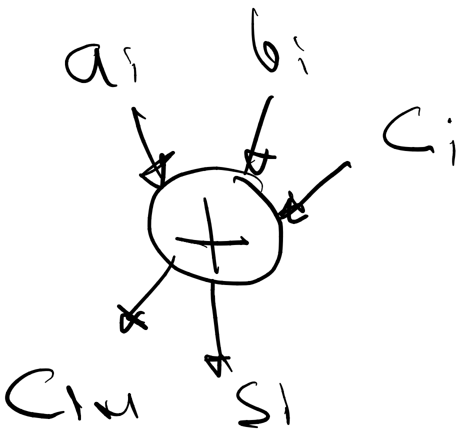
How to sum more than 2 numbers?



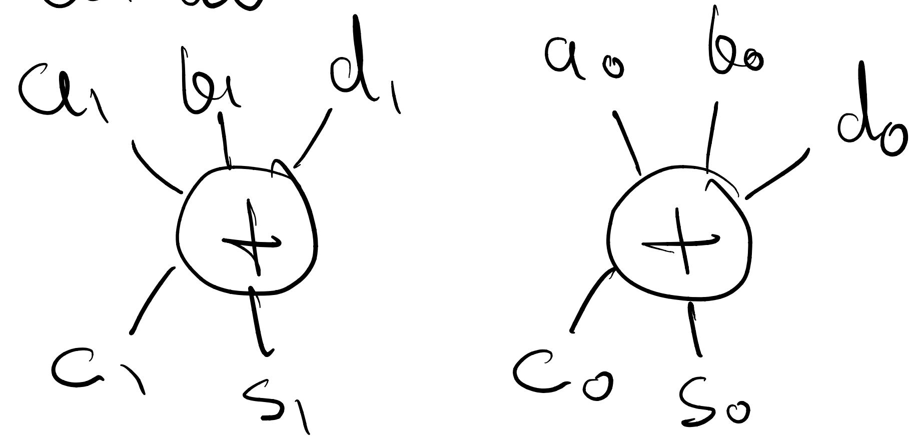


How to sum more than 2 numbers?

- Remember full-adders?



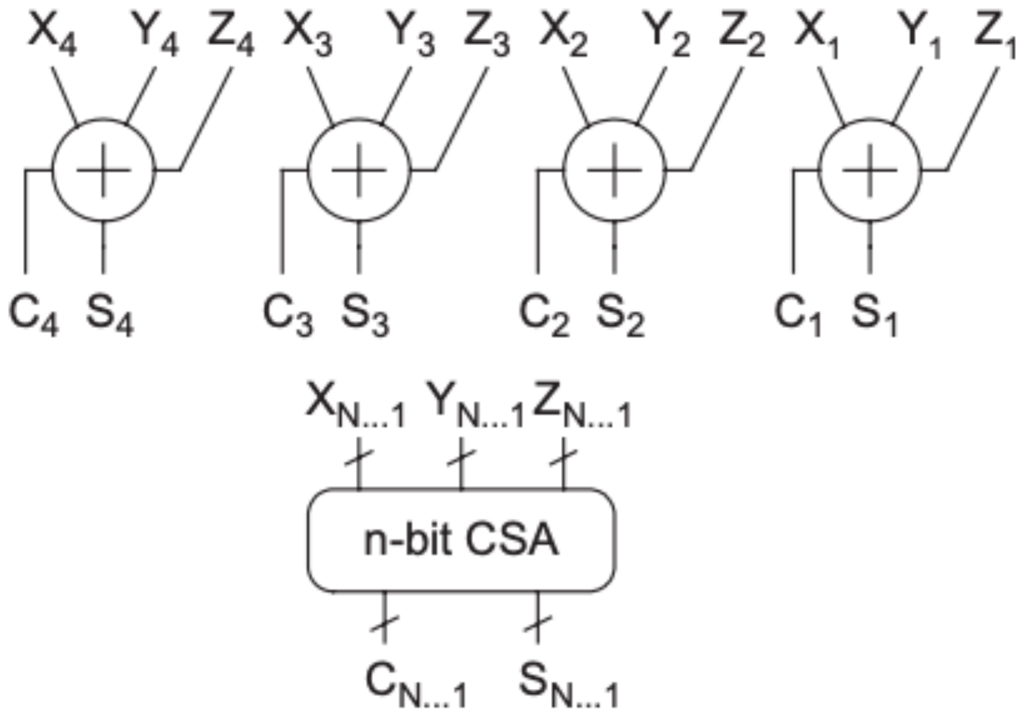
$$
\begin{array}{rcc}
 & a_1 & a_0 \\
+ & b_1 & b_0 \\
+ & d_1 & d_0
\end{array}
$$



$$A + B + C = S_1 S_0 + 2 \cdot C_1 C_0$$

$$
\begin{array}{rccc}
= & & S_1 & S_0 \\
+ & C_1 & C_0 & 0 \\
\hline
\end{array}
$$

Carry Save Adder (CSA)

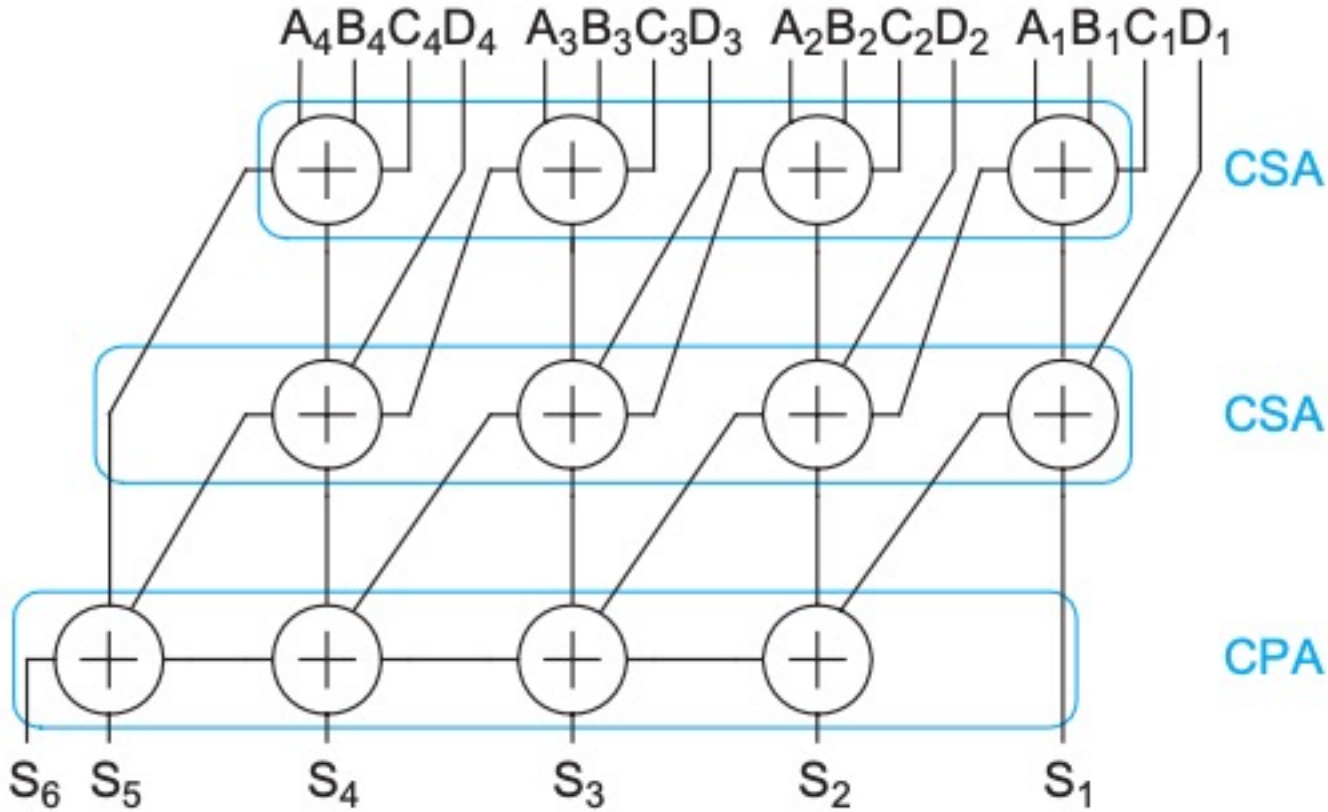


$$X + Y + Z = S + 2C$$

Source: Weste-Harris

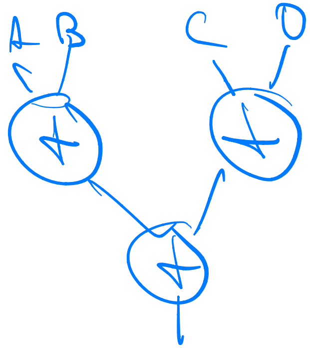
CSAs to add four numbers

$A_4 A_3 A_2 A_1$
$B_4 B_3 B_2 B_1$
$C_4 C_3 C_2 C_1$
$D_4 D_3 D_2 D_1$



Source: Weste-Harris

Array Multiplier



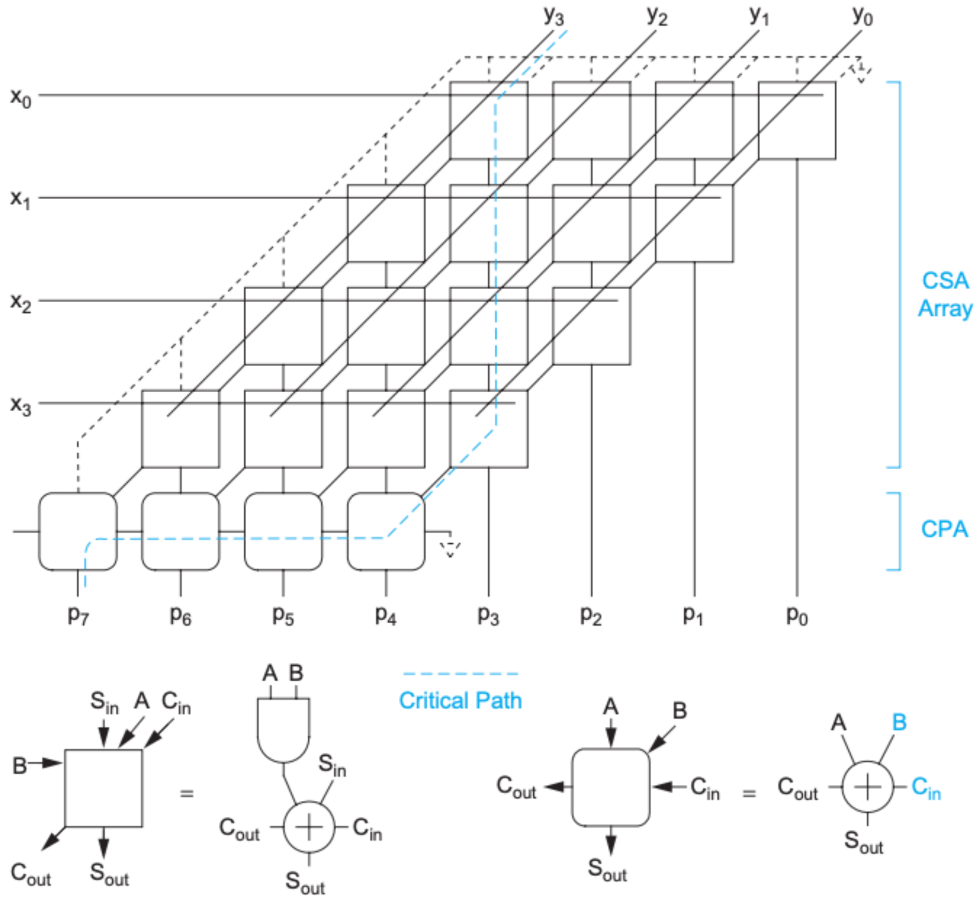


FIGURE 11.74 Array multiplier

Source: Weste-Harris

- Sequential addition of partial products (Analogous to Ripple-Carry Addition)
- Final addition of partial products is an Adder operation
  - All inputs do not arrive at the same time

Array Multiplier

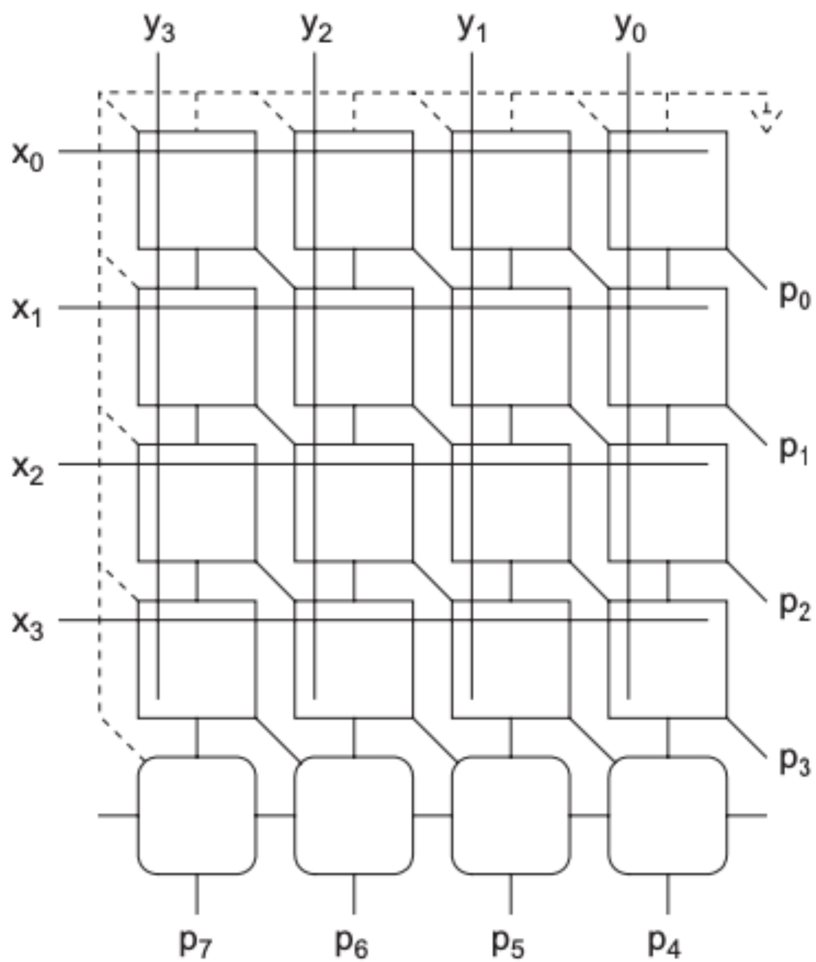


FIGURE 11.75 Rectangular array multiplier

Source: Weste-Harris

- "Straightening it" to be rectangular

Do you remember how to compute -x?

Two's complement

x . How do you compute $-x$?

$$-x = \overline{x} + 1$$

Signed Multiplication (Baugh-Wooley)

- Multiplier topology thus far has been signed multiplication

un

- $P_{signed} = X \times Y$

$$\left(-x_{m-1}2^{m-1} + \sum_{i=0}^{m-2} x_i 2^i\right)\left(-y_{n-1}2^{n-1} + \sum_{j=0}^{n-2} y_j 2^j\right)$$

$$= y_{n-1}\, x_{m-1} 2^{m+n-2}$$

$$- 2^{n-1} \sum_{i=0}^{m-2} y_{n-1} x_i 2^i \; - \; 2^{m-1} \sum_{j=0}^{n-1} x_{m-1} y_i 2^i$$

$$+ \sum_{i=0}^{m-2} \sum_{j=0}^{n-2} x_i y_j 2^{i+j}$$

$X = X_{m-1} X_{m-2} \cdots X_1 X_0$

$Y = Y_{n-1} Y_{n-2} - Y_1 Y_0$

Signed Array Multiplier (Baugh-Wooley)

$$\sum_{i=0}^{m-2} \sum_{j=0}^{n-2} x_i y_j 2^{i+j}$$

				x_0y_4	x_0y_3	x_0y_2	x_0y_1	x_0y_0
			x_1y_4	x_1y_3	x_1y_2	x_1y_1	x_1y_0	
		x_2y_4	x_2y_3	x_2y_2	x_2y_1	x_2y_0		
	x_3y_4	x_3y_3	x_3y_2	x_3y_1	x_3y_0			
x_4y_4	x_4y_3	x_4y_2	x_4y_1	x_4y_0				

Signed Array Multiplier (Baugh-Wooley)

							x_0y_4	x_0y_3	x_0y_2	x_0y_1	x_0y_0
						x_1y_4	x_1y_3	x_1y_2	x_1y_1	x_1y_0	
$\sum_{i=0}^{m-2} \sum_{j=0}^{n-2} x_i y_j 2^{i+j}$					x_2y_4	x_2y_3	x_2y_2	x_2y_1	x_2y_0		
				x_3y_4	x_3y_3	x_3y_2	x_3y_1	x_3y_0			
			x_4y_4	x_4y_3	x_4y_2	x_4y_1	x_4y_0				
$+\, y_{n-1}\, x_{m-1} 2^{m+n-2}$	x_5y_5										

Signed Array Multiplier (Baugh-Wooley)

								x_0y_4	x_0y_3	x_0y_2	x_0y_1	x_0y_0
							x_1y_4	x_1y_3	x_1y_2	x_1y_1	x_1y_0	
$\sum_{i=0}^{m-2}\sum_{j=0}^{n-2} x_i y_j 2^{i+j}$						x_2y_4	x_2y_3	x_2y_2	x_2y_1	x_2y_0		
					x_3y_4	x_3y_3	x_3y_2	x_3y_1	x_3y_0			
				x_4y_4	x_4y_3	x_4y_2	x_4y_1	x_4y_0				
$+\, y_{n-1}\, x_{m-1} 2^{m+n-2}$		x_5y_5										
												1
$-\, 2^{m-1}\sum_{j=0}^{n-1} x_{m-1} y_i 2^i$	$\overline{0}$	$\overline{0}$	$\overline{x_5y_4}$	$\overline{x_5y_3}$	$\overline{x_5y_2}$	$\overline{x_5y_1}$	$\overline{x_5y_0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$

Signed Array Multiplier (Baugh-Wooley)

								x_0y_4	x_0y_3	x_0y_2	x_0y_1	x_0y_0
							x_1y_4	x_1y_3	x_1y_2	x_1y_1	x_1y_0	
$\sum_{i=0}^{m-2}\sum_{j=0}^{n-2} x_i y_j 2^{i+j}$						x_2y_4	x_2y_3	x_2y_2	x_2y_1	x_2y_0		
					x_3y_4	x_3y_3	x_3y_2	x_3y_1	x_3y_0			
				x_4y_4	x_4y_3	x_4y_2	x_4y_1	x_4y_0				
$+\, y_{n-1}\, x_{m-1} 2^{m+n-2}$		x_5y_5										
												1
$-\, 2^{m-1} \sum_{j=0}^{n-1} x_{m-1} y_i 2^i$	$\overline{0}$	$\overline{0}$	$\overline{x_5y_4}$	$\overline{x_5y_3}$	$\overline{x_5y_2}$	$\overline{x_5y_1}$	$\overline{x_5y_0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$
$-\, 2^{n-1} \sum_{i=0}^{m-2} y_{n-1} x_i 2^i$	$\overline{0}$	$\overline{0}$	$\overline{x_4y_5}$	$\overline{x_3y_5}$	$\overline{x_2y_5}$	$\overline{x_1y_5}$	$\overline{x_0y_5}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$
												1

Signed Array Multiplier (Baugh-Wooley)

								x_0y_4	x_0y_3	x_0y_2	x_0y_1	x_0y_0
							x_1y_4	x_1y_3	x_1y_2	x_1y_1	x_1y_0	
$\sum_{i=0}^{m-2}\sum_{j=0}^{n-2} x_i y_j 2^{i+j}$						x_2y_4	x_2y_3	x_2y_2	x_2y_1	x_2y_0		
					x_3y_4	x_3y_3	x_3y_2	x_3y_1	x_3y_0			
				x_4y_4	x_4y_3	x_4y_2	x_4y_1	x_4y_0				
$+\, y_{n-1}\, x_{m-1} 2^{m+n-2}$		x_5y_5										1
$-\, 2^{m-1}\sum_{j=0}^{n-1} x_{m-1} y_i 2^i$	1	1	$\overline{x_5y_4}$	$\overline{x_5y_3}$	$\overline{x_5y_2}$	$\overline{x_5y_1}$	$\overline{x_5y_0}$	1	1	1	1	1
$-\, 2^{n-1}\sum_{i=0}^{m-2} y_{n-1} x_i 2^i$	1	1	$\overline{x_4y_5}$	$\overline{x_3y_5}$	$\overline{x_2y_5}$	$\overline{x_1y_5}$	$\overline{x_0y_5}$	1	1	1	1	1
												1

Signed Array Multiplier (Baugh-Wooley)

$\sum_{i=0}^{m-2}\sum_{j=0}^{n-2} x_i y_j 2^{i+j}$

$+\, y_{n-1}\, x_{m-1} 2^{m+n-2}$

$-\, 2^{m-1}\sum_{j=0}^{n-1} x_{m-1} y_i 2^i$

$-\, 2^{n-1}\sum_{i=0}^{m-2} y_{n-1} x_i 2^i$

							x_0y_4	x_0y_3	x_0y_2	x_0y_1	x_0y_0
						x_1y_4	x_1y_3	x_1y_2	x_1y_1	x_1y_0	
					x_2y_4	x_2y_3	x_2y_2	x_2y_1	x_2y_0		
				x_3y_4	x_3y_3	x_3y_2	x_3y_1	x_3y_0			
			x_4y_4	x_4y_3	x_4y_2	x_4y_1	x_4y_0				
	x_5y_5										1
1	1	$\overline{x_5y_4}$	$\overline{x_5y_3}$	$\overline{x_5y_2}$	$\overline{x_5y_1}$	$\overline{x_5y_0}$	1	1	1	1	1
1	1	$\overline{x_4y_5}$	$\overline{x_3y_5}$	$\overline{x_2y_5}$	$\overline{x_1y_5}$	$\overline{x_0y_5}$	1	1	1	1	1
											1

						1	$\overline{x_0y_5}$	x_0y_4	x_0y_3	x_0y_2	x_0y_1	x_0y_0
						$\overline{x_1y_5}$	x_1y_4	x_1y_3	x_1y_2	x_1y_1	x_1y_0	
					$\overline{x_2y_5}$	x_2y_4	x_2y_3	x_2y_2	x_2y_1	x_2y_0		
				$\overline{x_3y_5}$	x_3y_4	x_3y_3	x_3y_2	x_3y_1	x_3y_0			
			$\overline{x_4y_5}$	x_4y_4	x_4y_3	x_4y_2	x_4y_1	x_4y_0				
1	1	x_5y_5	$\overline{x_5y_4}$	$\overline{x_5y_3}$	$\overline{x_5y_2}$	$\overline{x_5y_1}$	$\overline{x_5y_0}$					

Removing leading 1's from signed numbers?

5 bit number

2's (5 bits) 1 1 0 1 0 $= -16 + 8 + 2 = -6$

2's (4 bits) 1 0 1 0 $= -8 + 2 = -6$

$1\ 1\ a_2\ a_1\ a_0$

$$= -2^4 + 2^3 + a_2 2^2 + a_1 2^1 + a_0$$

$$= -2^3 + a_2 2^2 + a_1 2^1 + a_0$$

$$\boxed{\begin{array}{l} 2^3 + 2^3 = 2 \cdot 2^3 = 2^4 \\ -2^4 + 2^3 = -2^3 \end{array}}$$

$1\ a_2\ a_1\ a_0$

Signed Array Multiplier (Baugh-Wooley)

$\sum_{i=0}^{m-2} \sum_{j=0}^{n-2} x_i y_j 2^{i+j}$

$+ y_{n-1}\, x_{m-1} 2^{m+n-2}$

$- 2^{m-1} \sum_{j=0}^{n-1} x_{m-1} y_i 2^i$

$- 2^{n-1} \sum_{i=0}^{m-2} y_{n-1} x_i 2^i$

					1	$\overline{x_0y_5}$	x_0y_4	x_0y_3	x_0y_2	x_0y_1	x_0y_0
					$\overline{x_1y_5}$	x_1y_4	x_1y_3	x_1y_2	x_1y_1	x_1y_0	
				$\overline{x_2y_5}$	x_2y_4	x_2y_3	x_2y_2	x_2y_1	x_2y_0		
			$\overline{x_3y_5}$	x_3y_4	x_3y_3	x_3y_2	x_3y_1	x_3y_0			
		$\overline{x_4y_5}$	x_4y_4	x_4y_3	x_4y_2	x_4y_1	x_4y_0				
1	x_5y_5	$\overline{x_5y_4}$	$\overline{x_5y_3}$	$\overline{x_5y_2}$	$\overline{x_5y_1}$	$\overline{x_5y_0}$					

unsigned

0	0	0	•	•	•	•
0	0	•	•	•	•	0
0	•	•	•	•	0	0
•	•	•	•	0	0	0

signed ?

$S = 2^{-k}$

S	S	S	S	•	•	•
S	S	S	•	•	•	0
S	S	•	•	•	0	0
S	•	•	•	0	0	0

Signed Array Multiplier (Baugh-Wooley)

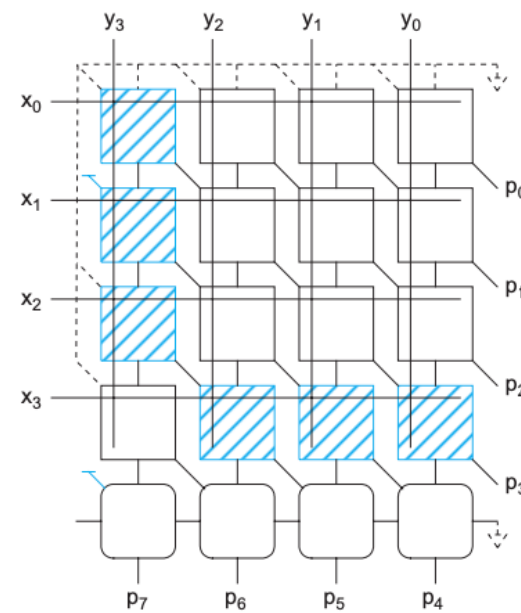
$$\sum_{i=0}^{m-2}\sum_{j=0}^{n-2} x_i y_j 2^{i+j}$$

$$+\, y_{n-1}\, x_{m-1} 2^{m+n-2}$$

$$-\, 2^{m-1}\sum_{j=0}^{n-1} x_{m-1} y_i 2^i$$

$$-\, 2^{n-1}\sum_{i=0}^{m-2} y_{n-1} x_i 2^i$$

					1	$\overline{x_0y_5}$	x_0y_4	x_0y_3	x_0y_2	x_0y_1	x_0y_0
					$\overline{x_1y_5}$	x_1y_4	x_1y_3	x_1y_2	x_1y_1	x_1y_0	
				$\overline{x_2y_5}$	x_2y_4	x_2y_3	x_2y_2	x_2y_1	x_2y_0		
			$\overline{x_3y_5}$	x_3y_4	x_3y_3	x_3y_2	x_3y_1	x_3y_0			
		$\overline{x_4y_5}$	x_4y_4	x_4y_3	x_4y_2	x_4y_1	x_4y_0				
1	x_5y_5	$\overline{x_5y_4}$	$\overline{x_5y_3}$	$\overline{x_5y_2}$	$\overline{x_5y_1}$	$\overline{x_5y_0}$					



Source: Weste-Harris

FIGURE 11.78 Modified Baugh-Wooley two's complement multiplier

Reducing the Number of Partial Products

- Array multipliers require merging M partial products
  - We're adding too many partial products!

- Choose r for "radix-2^r" Booth Encoding
- → reduction to $\lfloor M/r \rfloor$ partial products

r = 2

1 0 1 1 0 0
0 1 1 0 0 1

1 0 1 1 0 0
1 0 1 1 0 0 0
1 0 1 1 0 0

M = 5

3

5/2 = 2.5

Reducing the Number of Partial Products

- Array multipliers require merging M partial products
  - We're adding too many partial products!

Traditional: r = 1

r = 2

00 0
01 1
10 2
11 3

- Choose r for "radix-2^r" Booth Encoding
- → reduction to M/r partial products
  - Groups of r multiplier bits looked at to generate a single partial product (PP)
  - Referred to as radix-2^r encoding
  - Traditional radix-2 encoding: PP is selected as either 0 or Y (AND Array)
  - Radix-4 encoding: PP is selected as either 0, Y, 2Y or 3Y (Selection logic)

Similar ideas are relatively common in Digital VLSI. See for example:

Jeong, J., Collins, N., & Flynn, M. P. (2016). *A 260 MHz IF Sampling Bit-Stream Processing Digital Beamformer With an Integrated Array of Continuous-Time Band-Pass ΔΣ Modulators.* IEEE Journal of Solid-State Circuits, 51(5), 1168-1176.

Reducing the Number of Partial Products

- Array multipliers require merging M partial products
  - We're adding too many partial products!

- Choose r for "radix-2^r" Booth Encoding
- → reduction to M/r partial products
  - Groups of r multiplier bits looked at to generate a single partial product (PP)
  - Referred to as radix-2^r encoding
  - Traditional radix-2 encoding: PP is selected as either 0 or Y (AND Array)
  - Radix-4 encoding: PP is selected as either 0, Y, 2Y or 3Y (Selection logic)

$r=2$ $r=3$

- Radix-4 and radix-8 encoding used more often
  - Increased complexity in generating partial products with ↑r
  - Use of even higher radix justified in rare cases

Booth Encoding (Radix-4)

- Idea: Scale partial product by 0, 1, 2 or 3
  - 2Y generated from a shift <<1
  - 3Y is a problem.

0	easy
1	easy
2	easy
3	hard
4	easy

4 − 1 = 3 !!

3Y = 4Y − Y

```
  101110
11 10 01 00
--------------
      000000
    101110
   1011100
1001010
```

```
 101110
     11
--------
 101110
101110
--------
1001010
```

Booth Encoding (Radix-4)

- Idea: Scale partial product by 0, 1, 2 or 3
  - 2Y generated from a shift <<1
  - 3Y is a problem.

0	easy
1	easy
2	easy
3	hard
4	easy

4 - 1 = 3 :)

$3Y = 4Y - Y$

1 0 1 1 1 0

$\underbrace{1\ 1}\ \underbrace{0\ 0}_{+1}\ \underbrace{1\ 1}\ \underbrace{0\ 0}$

-Y

easy

Booth Encoding (Radix-4)

- Idea: Scale partial product by 0, 1, 2 or 3
  - 2Y generated from a shift <<1
  - 3Y is a problem. Try 4-1 instead of 2+1

Booth Encoding (Radix-4)

- Idea: Scale partial product by 0, 1, 2 or 3
  - 2Y generated from a shift <<1
  - 3Y is a problem. Try 4-1 instead of 2+1
- Let's try this: Detect a 3X case in the previous group and +1 to current group

01 00 11

+1 +1 -1

01 10 11

+1 +3 -1

+2 -1 -1

01 10 10

+1 +2 +2

Booth Encoding (Radix-4)

- Idea: Scale partial product by 0, 1, 2 or 3
  - 2Y generated from a shift <<1
  - 3Y is a problem. Try 4-1 instead of 2+1
- Let's try this: Detect a 3X case in the previous group and +1 to current group

- But.. what if previous group is 11 and the current group is 10?

Booth Encoding (Radix-4)

- Idea: Scale partial product by 0, 1, 2 or 3
  - 2Y generated from a shift <<1
  - 3Y is a problem. Try 4-1 instead of 2+1
- Let's try this: Detect a 3X case in the previous group and +1 to current group

- But.. what if previous group is 11 and the current group is 10?
- What if current group is 10, previous is 10, and one before is 11?

01 10 10 11

X +1 +2 +3 -1

X +1 +3 -1 -1

+2 -1 -1 -1

Booth Encoding (Radix-4)

- Idea: Scale partial product by 0, 1, 2 or 3
  - 2Y generated from a shift <<1
  - 3Y is a problem. Try 4-1 instead of 2+1
- Let's try this: Detect a 3X case in the previous group and +1 to current group

- But.. what if previous group is 11 and the current group is 10?
- What if current group is 10, previous is 10, and one before is 11?
- Ugh… Prefixes again!! ☹

Booth Encoding (Radix-4)

- Idea: Scale partial product by 0, 1, 2 or 3
  - 2Y generated from a shift <<1
  - 3Y is a problem. Try 4-1 instead of 2+1
- Let's try this: Detect a 3X case in the previous group and +1 to current group

- But.. what if previous group is 11 and the current group is 10?
- What if current group is 10, previous is 10, and one before is 11?
- Ugh… Prefixes again!! ☹

- WAIT! There's an easy fix with a clever idea ☺

Booth Encoding (Radix-4)

- Idea: Scale partial product by 0, 1, 2 or 3
  - 2Y generated from a shift <<1
  - 3Y is a problem. Try 4-1 instead of 2+1
- Let's try this: Detect a 3X case in the previous group and +1 to current group

- But.. what if previous group is 11 and the current group is 10?
- What if current group is 10, previous is 10, and one before is 11?
- Ugh… Prefixes again!! ☹

- WAIT! There's an easy fix with a clever idea ☺

$$\boxed{\mathbf{2 = 4 - 2}}$$

x0 x1 x2 x3 → x0 x1 x2 x(4-1)

x0 x1 x2 x3 → x0 x1 x2 or x(4-2) x(4-1)

Booth Encoding (Radix-4)

$$\boxed{\mathbf{2 = 4 - 2}}$$

- Idea: Scale partial product by 0, 1, 2 or 3
  - 2Y generated from a shift <<1
  - 3Y is a problem. Try 4-1 instead of 2+1
  - 2Y implemented as
    - sometimes 2Y
    - sometimes 4Y-2Y

Booth Encoding (Radix-4)

$$\boxed{\mathbf{2 = 4 - 2}}$$

- Idea: Scale partial product by 0, 1, 2 or 3
  - 2Y generated from a shift <<1
  - 3Y is a problem. Try 4-1 instead of 2+1
  - 2Y implemented as
    - sometimes 2Y
    - sometimes 4Y-2Y

+1 +1 +1 +1

$\underbrace{00}_{+1}\ \underbrace{10}_{-2}\ \underbrace{01}_{+2}\ \underbrace{10}_{-1}\ \underbrace{10}_{-1}\ \underbrace{11}_{-1}$

$\underbrace{01}_{+1}\ \underbrace{0}$

$\underbrace{01}_{+2}\ \underbrace{1}$

$\underbrace{10}_{-2}\ \underbrace{0}$

$\underbrace{10}_{-1}\ \underbrace{1}$

- Why is this a good idea?
  - We only need to look at the MSB bit of the previous 2-bit group! (No prefix ☺)

Booth-Encoding: rules and how it works

- Add extra group with leading 0s if needed
- Form partial products according to the following rules:
  - Add 4Y to next group based on MSB of current group

Inputs (numbers to multiply)			Outputs
Current Group		MSB from previous group	Partial product
x_{2i+1}	x_{2i}	x_{2i-1}	PP_i
0	0	0	0
0	1	0	Y
1	0	0	-2Y
1	1	0	-Y
0	0	1	Y
0	1	1	2Y
1	0	1	-Y
1	1	1	0

$$PP = \begin{cases} 0 \\ y \\ 2y \\ -2y \\ -y \end{cases}$$

Tree Multipliers

- Array multipliers inherently process PPs serially
- This addition can also be performed in parallel

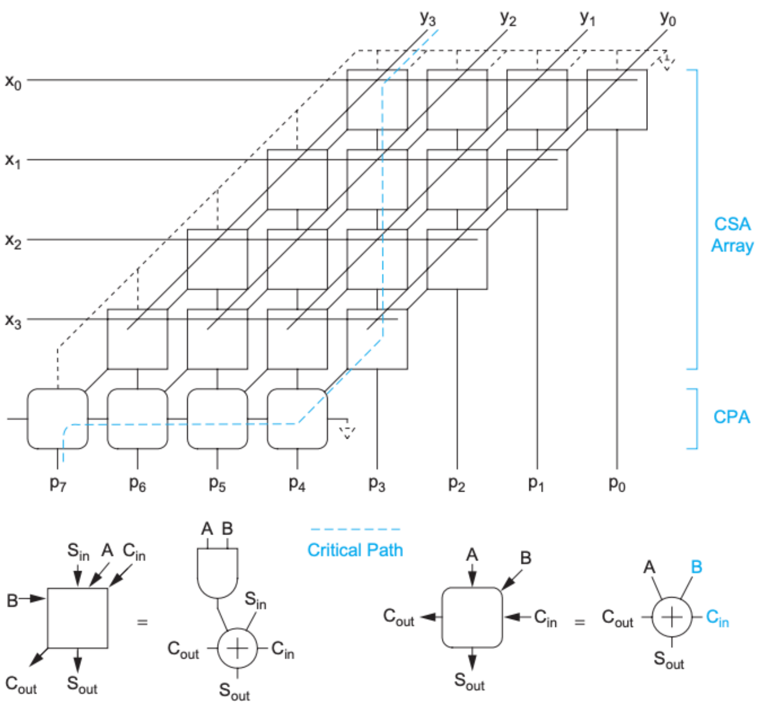


FIGURE 11.74 Array multiplier

Source: Weste-Harris

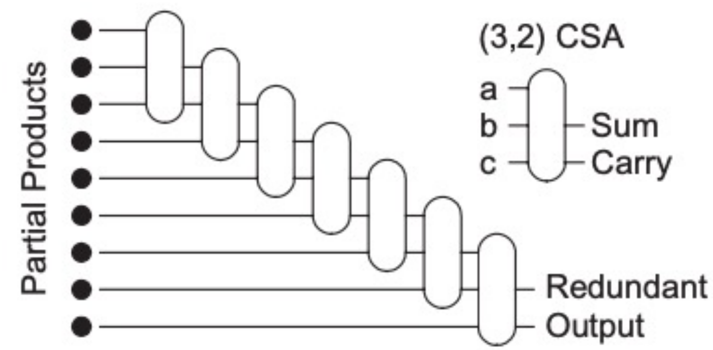


FIGURE 11.84 Dot diagram for array multiplier

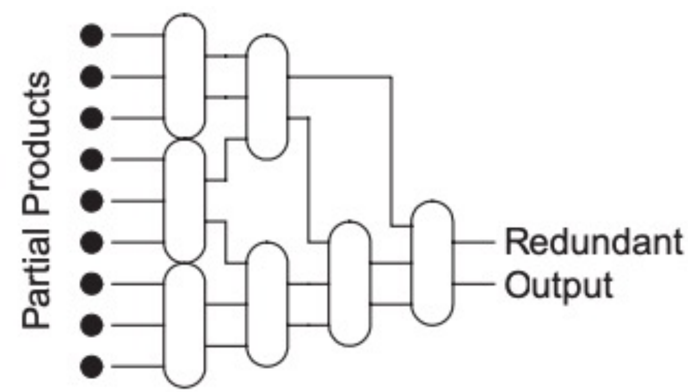


FIGURE 11.85 Dot diagram for Wallace tree multiplier

Tree Multipliers

- Array multipliers inherently process PPs serially
- This addition can also be performed in parallel

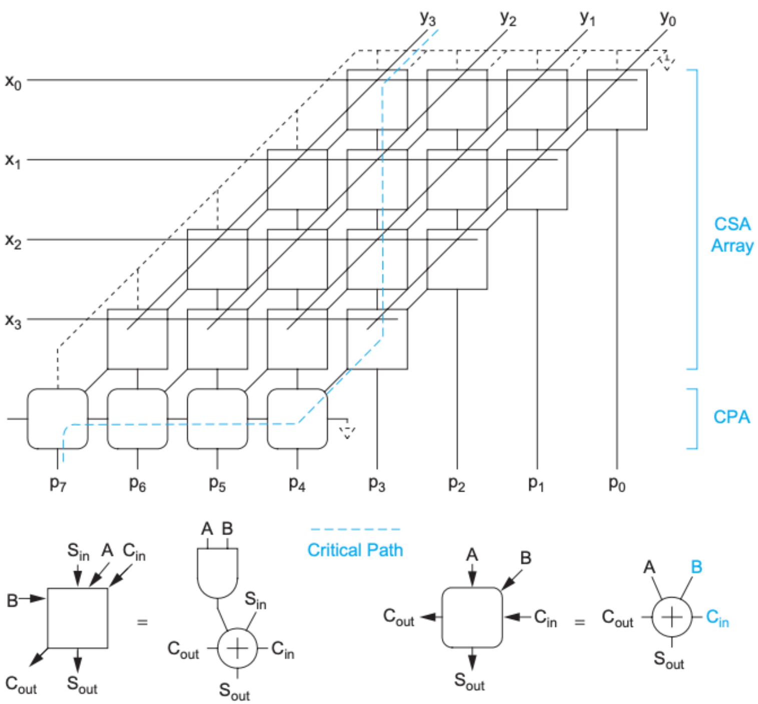


FIGURE 11.74 Array multiplier

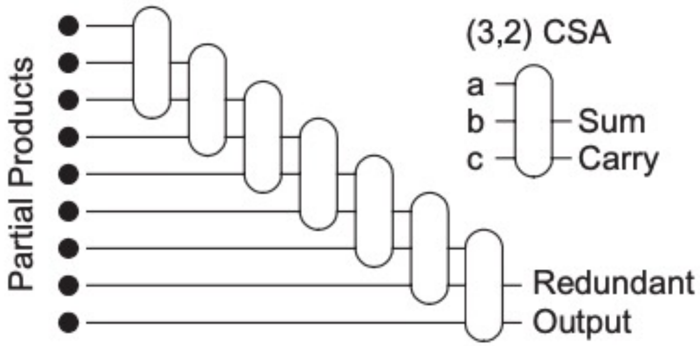


FIGURE 11.84 Dot diagram for array multiplier

Source: Weste-Harris

Questions/Assignments

- Signed booth multiplier?
- Non-equal width of the carry-save segment