EE241 - Spring 2003
Advanced Digital Integrated Circuits

Lecture 20

Multipliers

Binary Multiplication

$$Z = X \times Y = \sum_{k=0}^{M+N-1} Z_k 2^k$$

$$= \left(\sum_{i=0}^{M-1} X_i 2^i \right) \left(\sum_{j=0}^{N-1} Y_j 2^j \right)$$

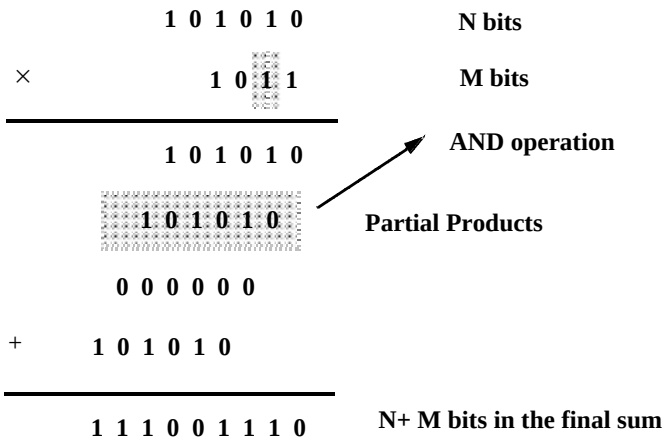
$$= \sum_{i=0}^{M-1} \left(\sum_{j=0}^{N-1} X_i Y_j 2^{i+j} \right)$$

with

$$X = \sum_{i=0}^{M-1} X_i 2^i$$

$$Y = \sum_{j=0}^{N-1} Y_j 2^j$$

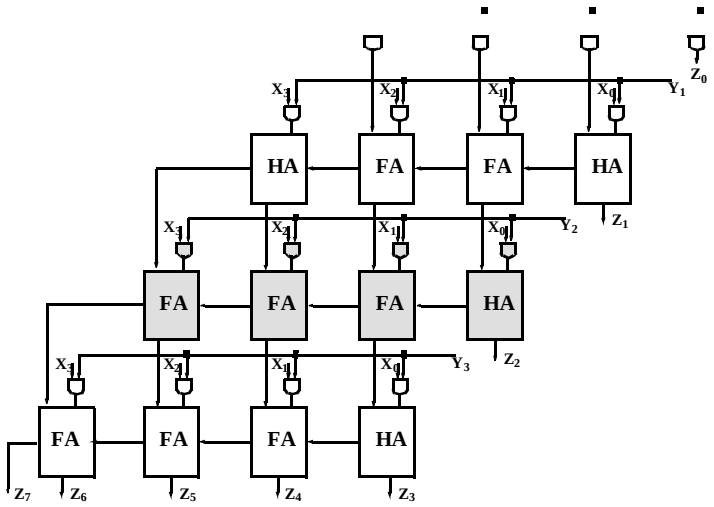
Binary Multiplication



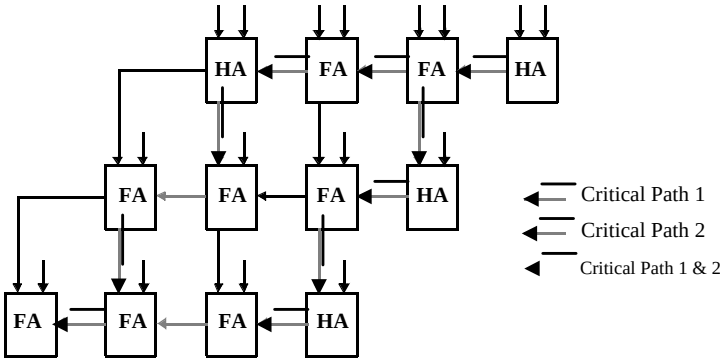
Shift-and-Add Multiplier

- Standard adder and shift-in the multiplicand
- Shift the result as well and add
- N cycles
- Parallel adders add more hardware (adders) instead.

Array Multiplier

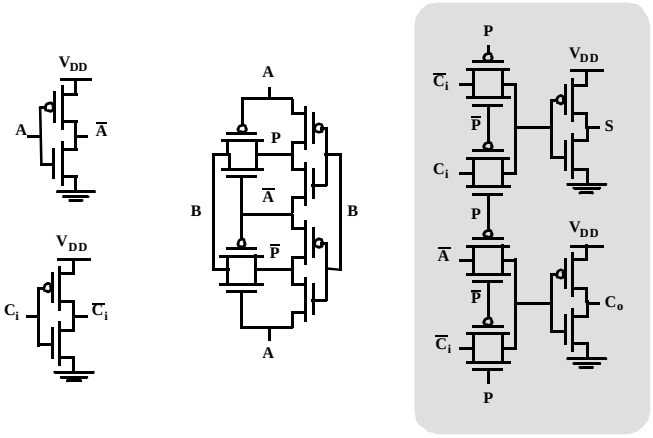


MxN Array Multiplier — Critical Path



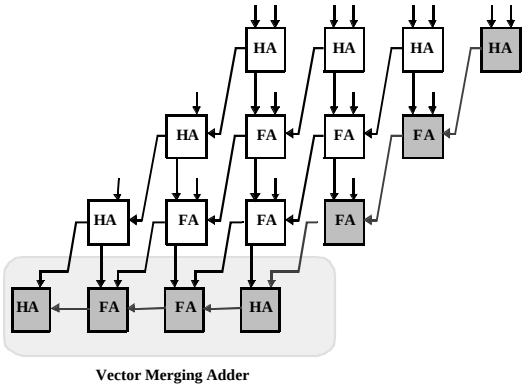
$$t_{mult} \approx [(M-1)+(N-2)]t_{carry} + (N-1)t_{sum} + (N-1)t_{and}$$

Adder Cells in Array Multiplier



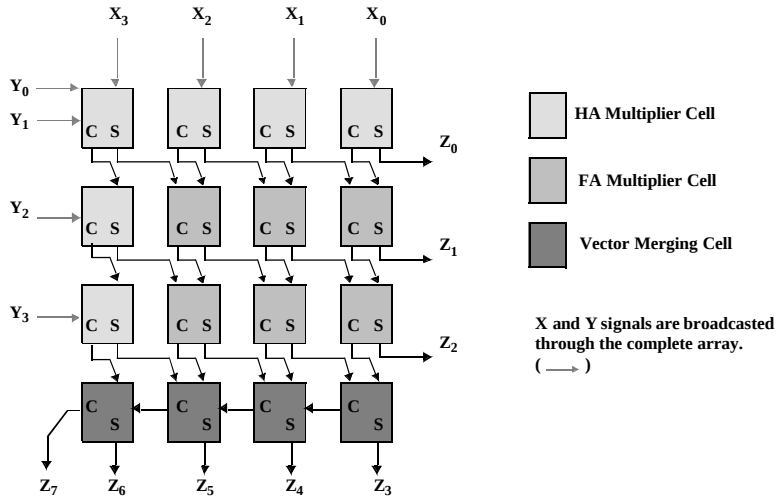
Identical Delays for Carry and Sum

Carry-Save Multiplier



$$t_{mult} = [(N-1)t_{carry} + (N-1)t_{and} + t_{merge}]$$

Multiplier Floorplan

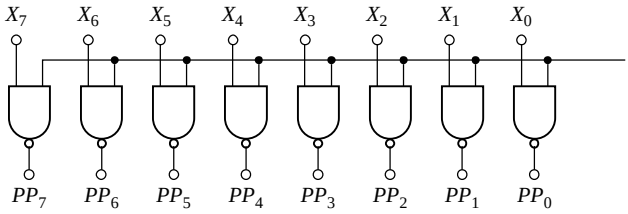


Multipliers

- Partial product generation
- Partial product accumulation
- Final summation

Generating Partial Products

- All partial products: AND



- Booth's recoding – reduction of partial product count

Booth Recoding

- Instead of generating all the partial products
  0 * x = 0 1 * x = x x={0,1}
- Reduce the number of partial products by grouping

0 0	0	
0 1	1*	
1 0	2*	(shift)
1 1	3*	(or 4* -1)

Booth'51

Booth Recoding

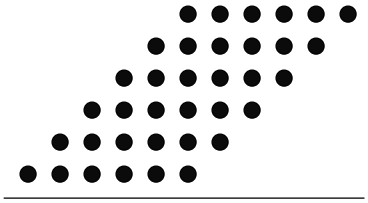
- Instead of using set {0, 1*Y, 2*Y, 3*Y}
- Use {0, 1*Y, 2*Y, 4*Y, -Y}
- Shifting and complementing
- 3*Y = 4*Y – Y
- Can be simplified by looking into three bits – modified Booth recoding

Modified Booth Recoding

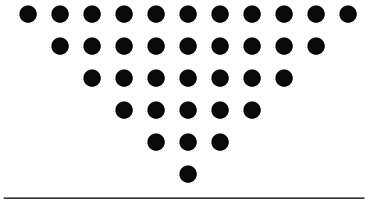
Two bits and the MSB of previous two

$x_{i+2}x_{i+1}x_i$	Add to partial product
000	+0Y
001	+1Y
010	+1Y
011	+2Y
100	-2Y
101	-1Y
110	-1Y
111	-0Y

Accumulating Partial Products

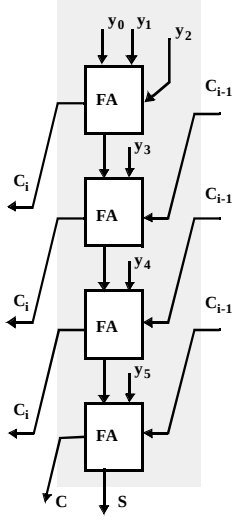


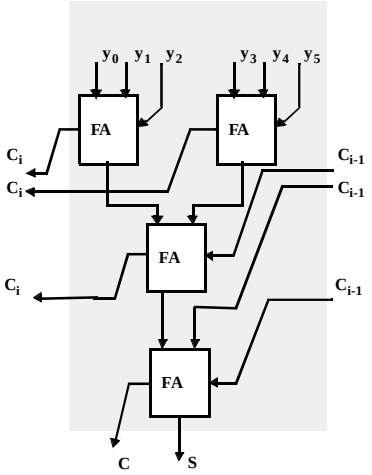
Partial product matrix



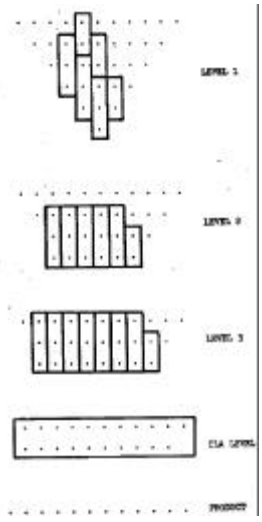
Reorganized matrix

Wallace-Tree Multiplier

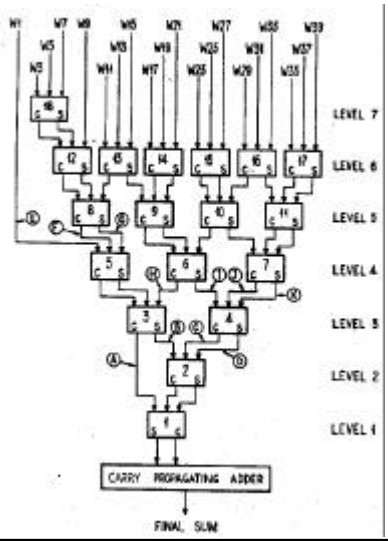




Wallace-Tree Multiplier



Wallace-Tree Multiplier



Wallace,
Trans on Comp. 2/64

Tree Multipliers

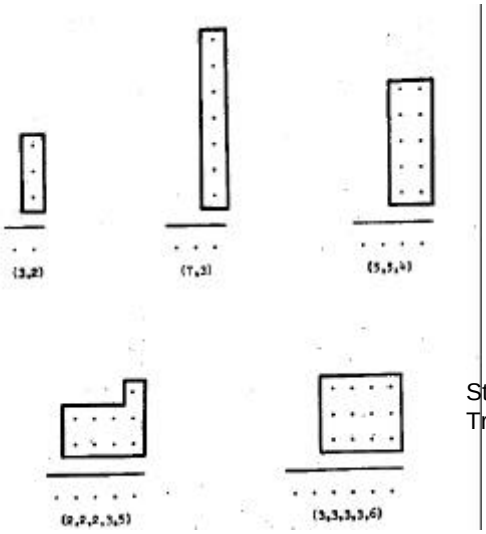
- Time is proportional to log N
- Wiring is complicated
- Different wire lengths
- Optional pipelining
- Wallace tree: reduce the number of operands at earliest opportunity
- Dadda tree: reduce the number of operands with fewest adders

Minimum Number of Stages

Number of bits in the multiplier	Number of stages
3	1
4	2
$4 < n \leq 6$	3
$6 < n \leq 9$	4
$9 < n \leq 13$	5
$13 < n \leq 19$	6
$19 < n \leq 28$	7
$28 < n \leq 42$	8
$42 < n \leq 63$	9

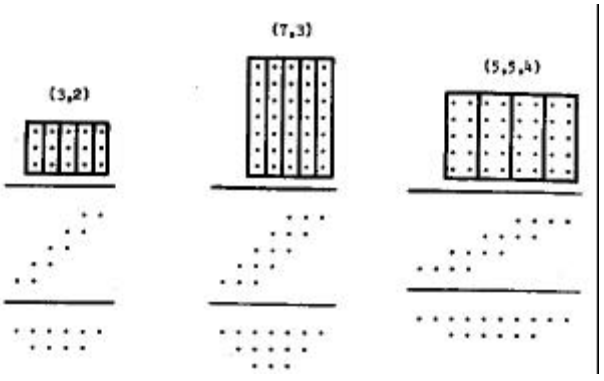
Dadda,`65

Generalized Counters



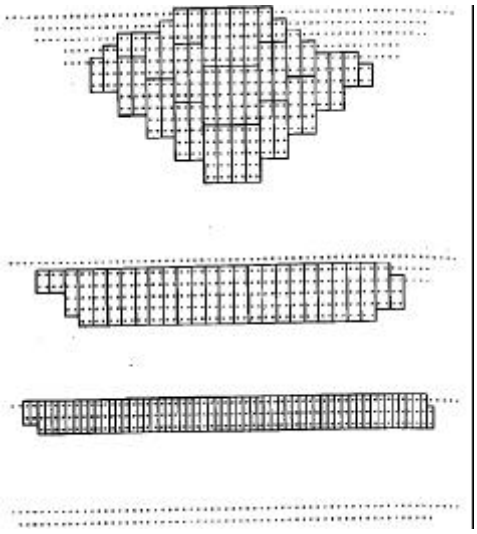
Stenzel,
Trans on Comp 10/77

Generalized Counters



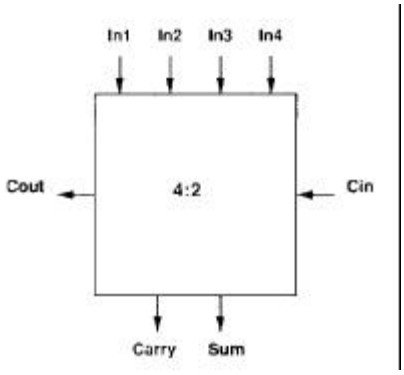
Generalized Counters

32x32b
using (5,5,4)
with (3,2) in
the last stage



4:2 Counters (Compressors)

4-2 carry-save module



Weinberger, IBM J. ResDev 1/81
Santoro, Horowitz, JSSC 4/89

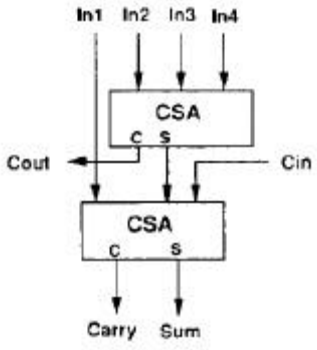
TRUTH TABLE FOR THE 4:2 ADDER
n is number of inputs (from $In1$, $In2$, $In3$, $In4$) which = 1, C_{in} is the input carry from the C_{out} of the adjacent bit slice, C_{out} and carry both have weight 2, and sum has weight 1.

n	Cin	Cout	Carry	Sum
0	0	0	0	0
1	0	0	0	1
2	0	*	*	0
3	0	*	0	1
4	0	*	1	0
0	1	0	0	1
1	1	0	1	0
2	1	*	*	1
3	1	1	1	0
4	1	1	1	1

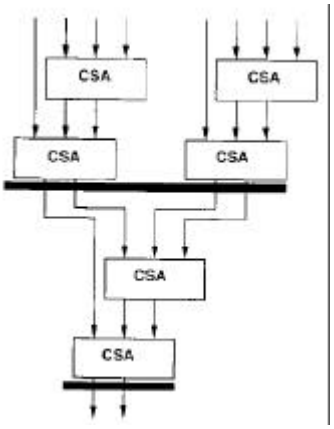
*Either C_{out} or Carry may be ONE for two or three inputs equal to 1 but NOT both.
C_{out} may NOT be a function of the C_{in} from the adjacent block or a ripple carry may occur.

4:2 Compressors

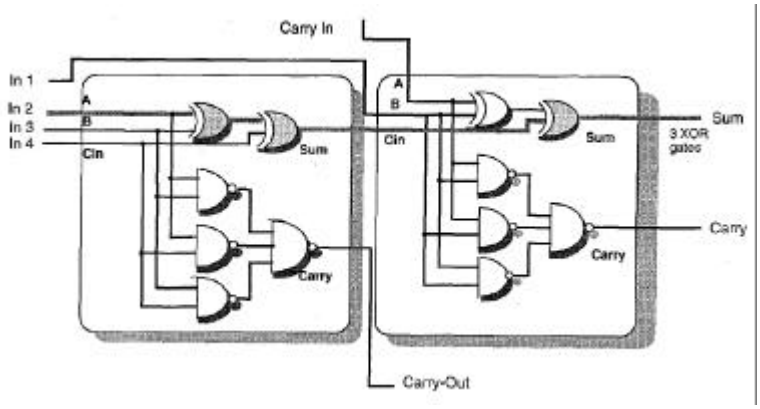
Built of CSAs



Pipelined version compresses
8 partial products per cycle

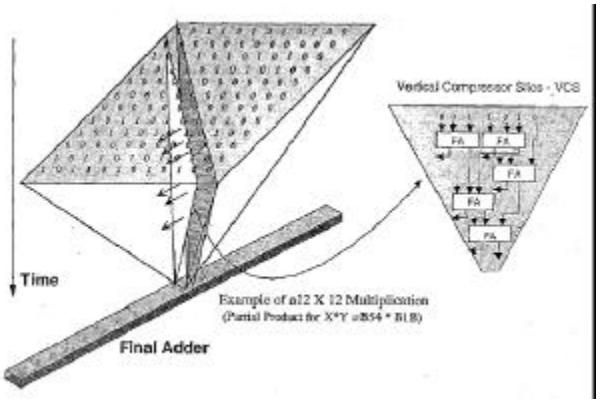


4:2 Compressors



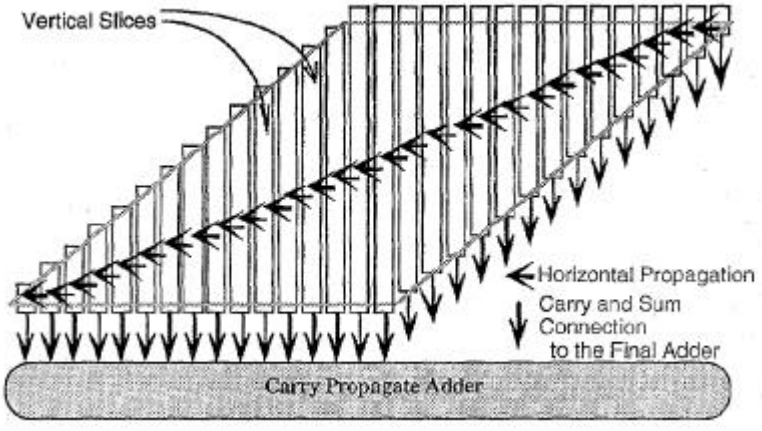
Interconnect can be more regular than in Wallace tree

Three Dimensional Optimization

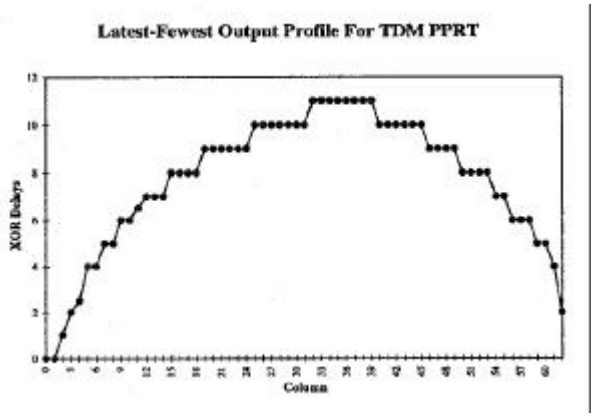


Oklobdzija, Villeger, Liu, Trans on Comp 3/96

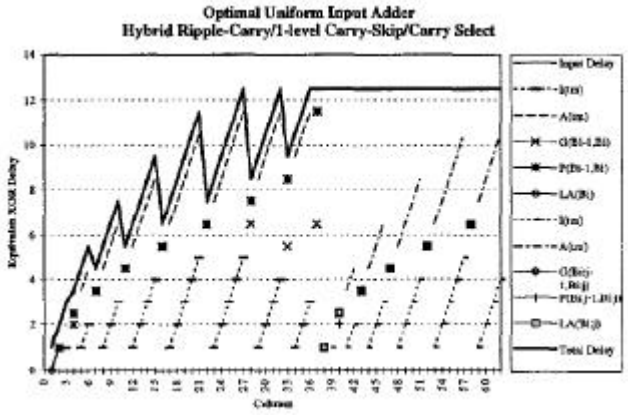
Vertical Slices in TDM



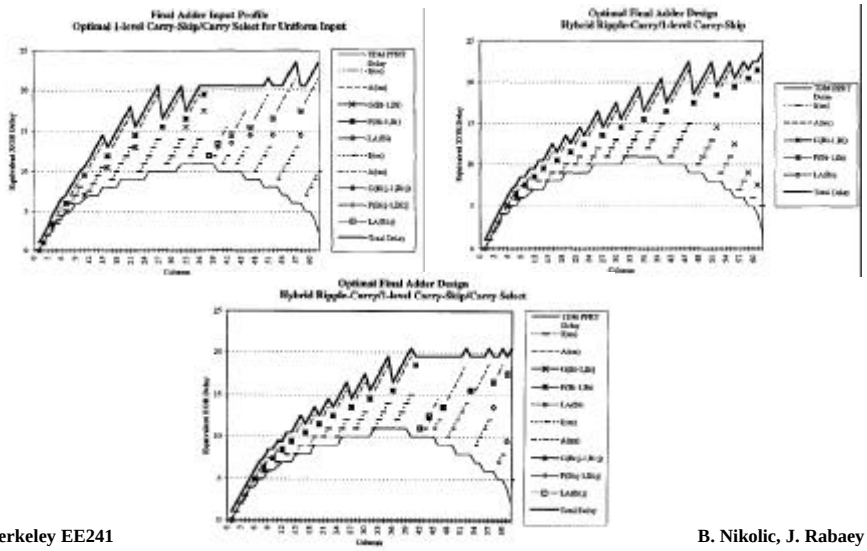
Final Addition



Final Addition

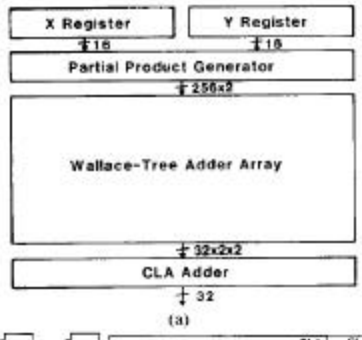


Final Addition



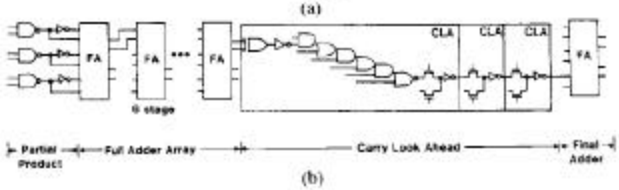
Example: CPL Multiplier

Block Diagram

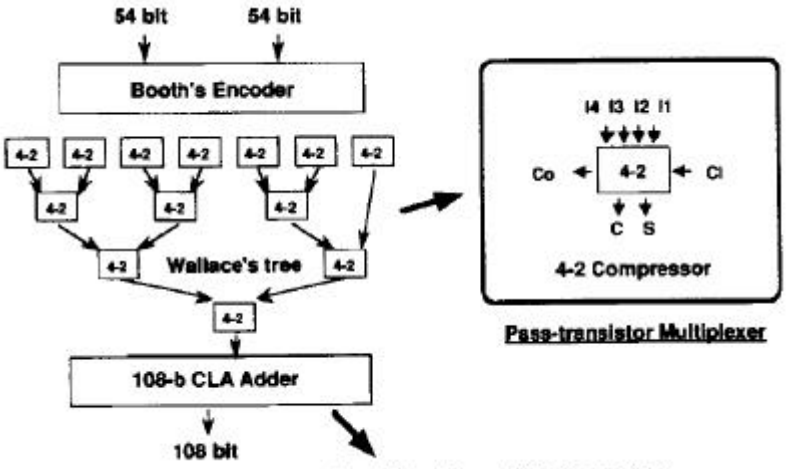


Yano, JSSC 4/90

Critical Path



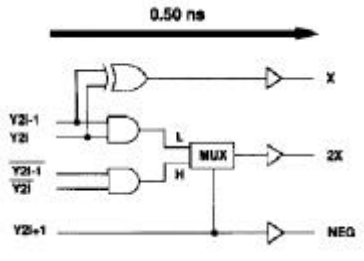
Example: DPL Multiplier



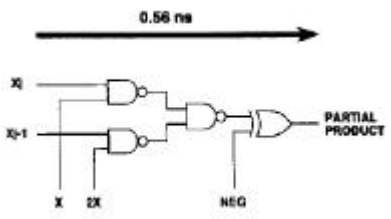
Ohkubo, JSSC 3/95

Example: DPL Multiplier

Booth encoder

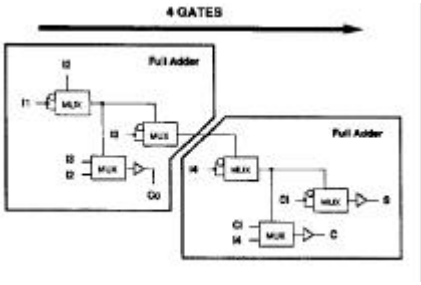


Partial product generator

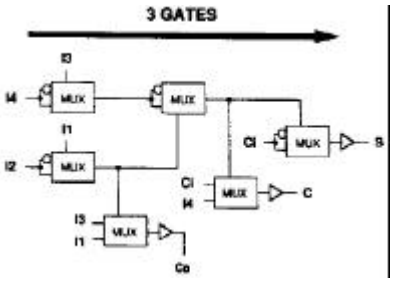


Example: DPL Multiplier

FA-based 4:2

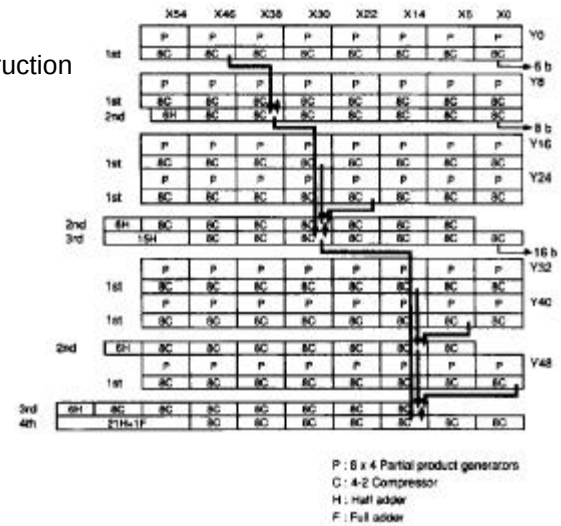


Modified 4:2



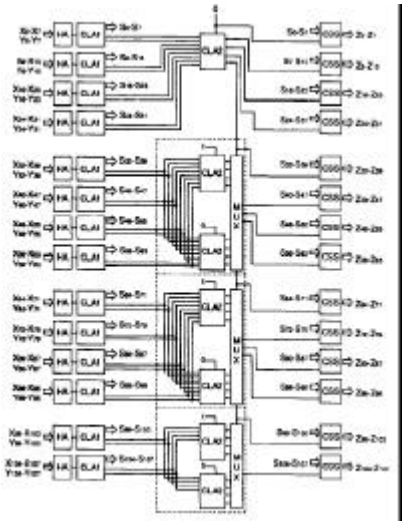
Example: DPL Multiplier

Tree construction

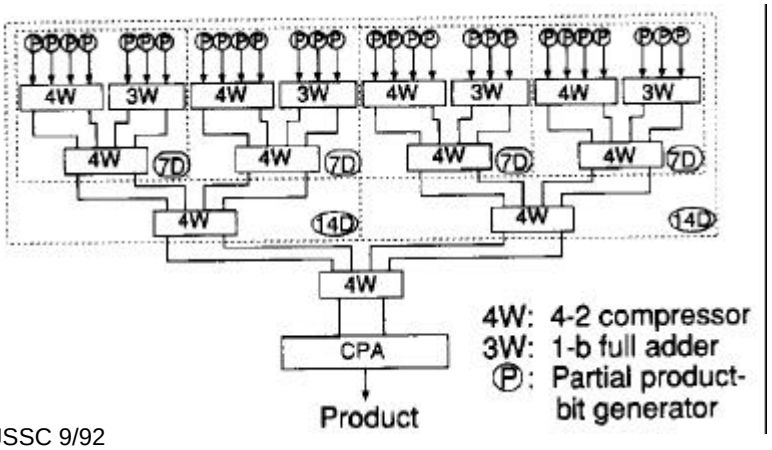


Example: DPL Multiplier

Final adder

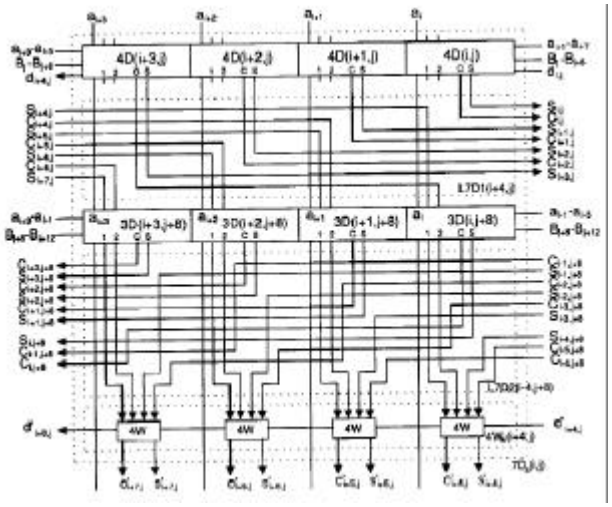


Regularly Structured Tree

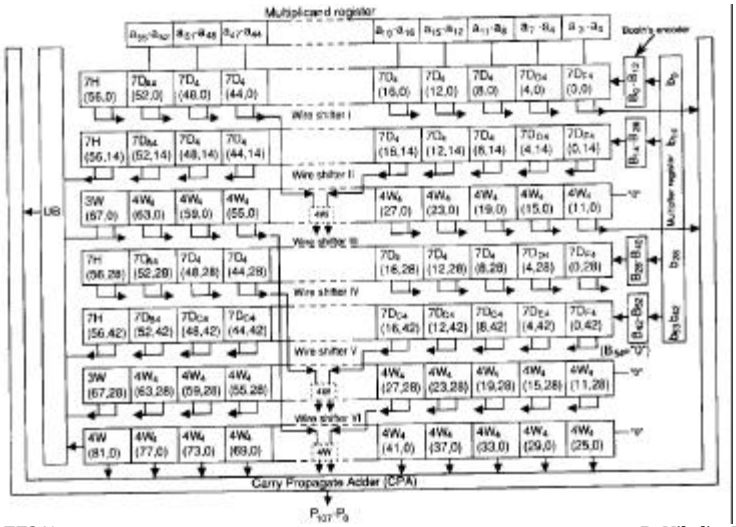


Goto, JSSC 9/92

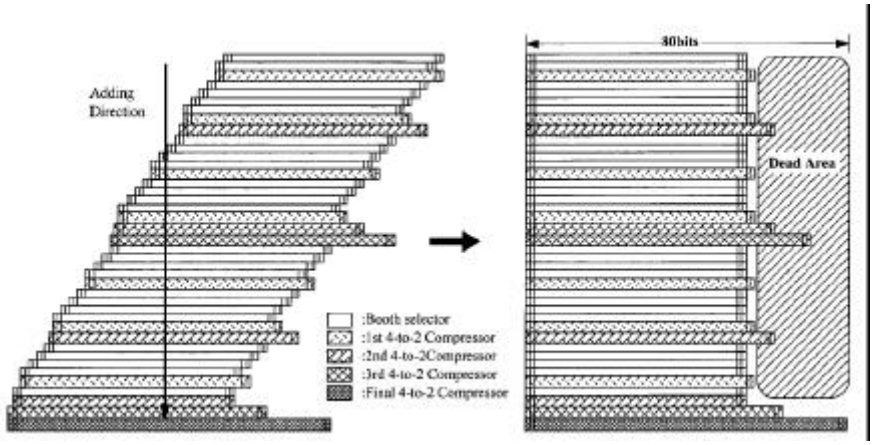
Regularly Structured Tree



Regularly Structured Tree

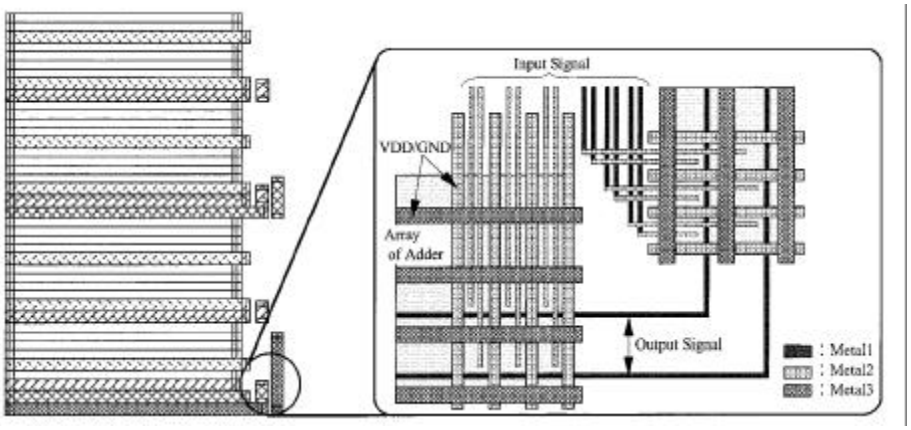


Regularly Structured Tree



Itoh, JSSC 2/01

Regularly Structured Tree



Regularly Structured Tree