EE 486 lecture 9: Multiply

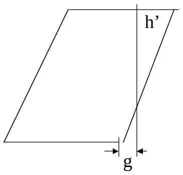
M. J. Flynn

Multiply

- Generating the partial products (pp's)
  - Booth encoding
  - Direct sub multipliers
- Reducing (or assimilating) the pp's
  - Arrays (2D)
  - Higher order arrays
  - Trees (3D)
- Iteration on the pp tree (or array)

Truncated multiply

- The multiplication operation forms a 2n bit product, but only needs to store an n bit result.
- Can truncate low order bits, if guard lsb with g=[$\log_2$ n] bits plus k
- k=1/2 h' (the tree height at lsb of g)



IEEE multiply and truncation

- In IEEE need full tree, but low order bits need only a zero detect for each bit.
- Then sticky bit, s= $\sim(\Sigma z_i)$ for each of the low order bits
- If we truncate $(X\ xY)^t$ then it should be that $(X\ xY)^t = (Y\ xX)^t$ if non Booth is used then this is true for all X, Y since we have the same number of 1's in each column

Truncated Booth

- This may not provide $(X\ xY)^t = (Y\ xX)^t$
- Consider Booth:2X – X should = X
- If X=00010$(1)^t$
- 2X=00101
- C(X^t) =11110 or C(X)=111010,
- Not the same results

Reducing the pp's

- The basic unit of reduction is the CSA (carry save adder)
- This is simply a binary full adder that takes 3 inputs of the same weight and provides 2 outputs, sum and carry. It's also called a (3,2) counter.
- In adding 3 numbers X,Y and Z we reduce the 3 operand to 2 and then do a single CPA

CSAs

- By using CSAs we reduce the 3 input operands to 2 in 2 gate delays, then do a CPA
- Can extend to n operands
- In case of multiply these operands are the pp's

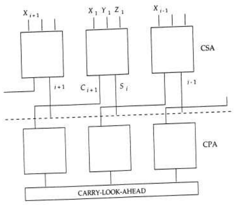
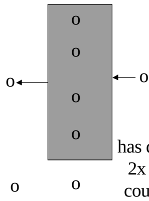


Figure 3.7: Addition of three n-bit numbers.

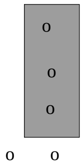
Compressors and Counters

- [4:2] compressor

(3,2) counter



has delay 2x (3,2) counters



Multiplier topology

- Refers to the way bit positions in the pp reduction are interconnected.
- Can be either arrays or trees
- Two important measures of a topology are 1) the minimum number of wires needed to interconnect the counters within a single bit position, and 2) the number of counter delays required to reduce the pp's to the 2 inputs to the CPA.

Using CSAs in linear arrays

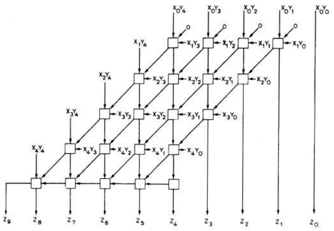


Figure 4.15: 5 × 5 unsigned multiplication.

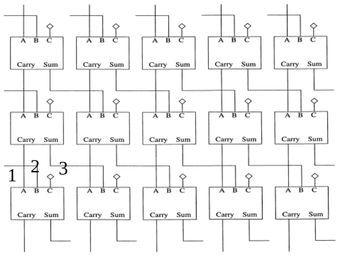
Linear arrays

- In a simple array each cell is a CSA
- To add n operands takes n-2 CSA levels plus a final CPA or 2(n-2) + CPA gate delays.
- While relative slow linear arrays use a minimum width of wiring channel.
- Number of wiring tracks/bit channel is w.
- w is determined by the topology. Some circuit family may require 2w to implement the design.

Wires vs. gates

- Arrays minimize w at the expense of gate delays.
- Wiring channel can determine the column bit pitch, forcing larger designs with more overall wire.
- Some circuit techniques use 2 wires/signal (eg dual rail domino), while fast they can only be used when w can accommodate it.

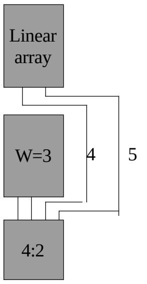
Simple array, w=3



Double arrays

- A single linear arrays has w=3 and h=n-2.
- A double array create 2 sub arrays of h=(n-2)/2; provides 4 pp's to be summed by a [4:2] and then a CPA.
- The double array has w=5

Double Array, w=5


Linear array

W=3

4

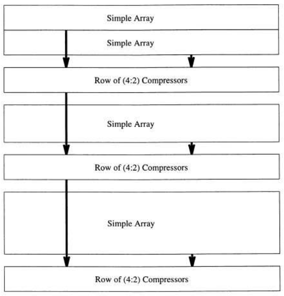
5

4:2

Higher order arrays (Al-Twaijry'94)

- More than 2 sub arrays summed by counters or compressors.
- Optimum configurations arrange unequal length linear sub arrays into a string of [4:2] compressors so that the shortest sub array has the longest compressor chain and all paths are balanced.

Structure of higher order array



Higher order array, w=5

- As in the case of the double array a linear (or simple) array with w =3 is bypassed with the 2 outputs of a [4:2] compressor.
- Gives a total w =5.
- So, by construction, all arrays should have no more than w = 5.

Arrays, width and depth in (3,2) levels

Array type	w, lines per bit	d, # of (3,2)counters
simple	3	n-2
double	5	((n-4)/2) + 2 = n/2
higher order	5	**O** (2 sqrt(n))

Trees

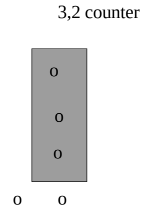
- Trees differ from arrays in that they optimize depth (the number of counter delays at the expense of width,w.
- The width of a tree (number of wires per bit position) is a function of n, the height of the pp reduction array.
- Trees are either regular whose width is a known function of n or irregular where w is determined by design layout.

Types of trees

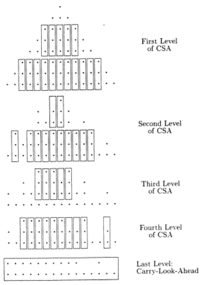
- Wallace tree: a fast irregular tree using (3,2) counters; w depends on layout. Variations use (m,n) counters
- Binary tree: regular tree using [4:2]s, w is predictable
- ZM trees: regular tree with w=5 (becomes a higher order array)
- OS (overturned staircase) trees: regular, can achieve Wallace tree delay.

Wallace ('64) big, old, trees

- The basic tree is the Wallace tree. It uses the 3,2 counter to reduce n operands to 2
- Requires about [$\log_{3/2}n/2$] levels
- No concern about w



Wallace tree for 8x8 using (3,2)



Other counters can be used

- Dadda- Stenzel counters have the form (C_{R-1},…,C_0,d) each C_i is the height of the ith column and d is the number of output bits
- (5,5,4) and (7,3) are the usual alternatives to (3,2)

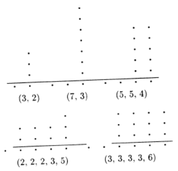
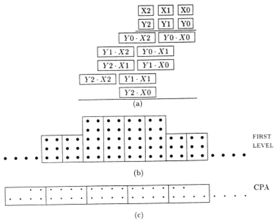


Figure 4.8: Some generalized counters from Stenzel [3

A 12x12 using 4x4 for pp generation and (5,5,4) counters for reduction

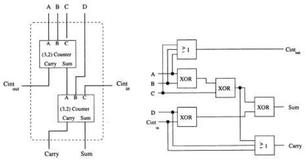


Counter considerations

- A ROM can be used to implement any counter: (Σc_i)x d, e.g. (2,2,2,2,2,6)
- Popular counters usually maximize the output states, so that d, almost a power of 2
- Counters are usually realized from (3,2)'s perhaps somewhat reconfigured.
- Can also have direct realizations using custom logic, PLAs tables, etc.

Compressors and counters

- Wallace trees are irregular in structure and are difficult to layout as their wiring requirements are determined only at layout
- A more regular tree is a binary tree. This is implemented using [4:2] compressors, this is a form of (5,3) counter. It can be implemented using 2 (3,2) counters, although it's possible to reconfigure these to create a [4:2] delay faster than 2 x (3,2) delays



(a) (4:2) Compressor constructed from two (3,2) Counters

(b) Redesigned (4:2) compressor

Binary tree(Weinberger '81): 16 pp's showing 1 bit

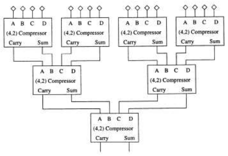
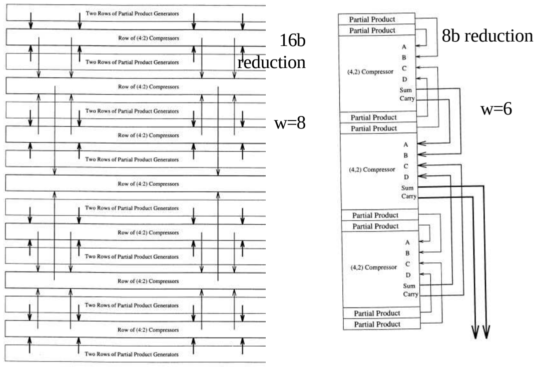


Figure 4.13: Bit slice of a binary tree that reduces 16 partial products.

Binary trees

- Delay in CSA levels is 2([$\log_2 n$] –1)
- Wire channel size seems to be n (the number of pp's), but it can be improved by bringing in each pp at an optimum point on the bit position of the tree.
- w= 2[$\log_2$ n]



16b reduction

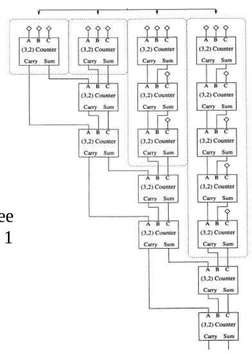
w=8

8b reduction

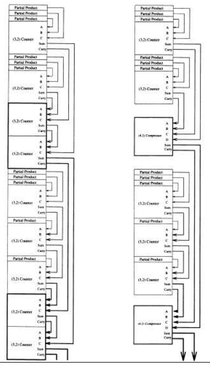
w=6

Z M and OS trees

- Other regular trees are possible using (3,2) or other counters
- The ZM (Zuras-McAllister ‘86) tree is also called a balanced delay tree. It’s recursively defined by a tree body and a chain. The connection is by a [4:2]. So the tree grows as 1,1,3,5,7,.. in CSA levels
- It effectively becomes a high order array with w=5 and d= **O** (2 sqrt n)



ZM tree
Type 1



ZM tree, type1
Showing w=5

Same, replacing
2 (3,2) with a [4:2]

Overturned staircase (OS) tree (Mou and Jutnad ’90)

- Similar to ZM but achieves Wallace tree type depth for most values of n.
- Recursively defined to connect branch of depth k-2 (CSAs) to body of depth k-1 to form tree of depth k.
- As with ZM higher order (types) are possible, with less regularity.

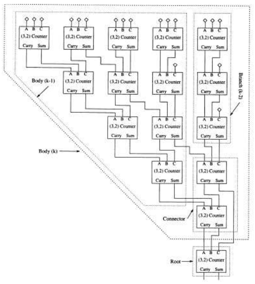
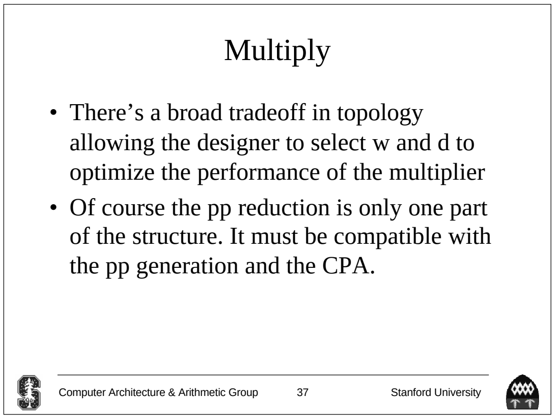


Figure 4.17: Overturned staircase tree that reduces 18 partial product bits.



Multiply

- There’s a broad tradeoff in topology allowing the designer to select w and d to optimize the performance of the multiplier
- Of course the pp reduction is only one part of the structure. It must be compatible with the pp generation and the CPA.