Accumulating Partial Products

Partial product matrix

Reorganized matrix
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

Dadda’s Tree
Dadda's Tree

Generalized Counters

Stenzel, 
Trans on Comp 10/77
Generalized Counters

32x32
using (5,5,4)
with (3,2) in
the last stage
Minimum Number of Stages

<table>
<thead>
<tr>
<th>Number of bits in the multiplier</th>
<th>Number of stages</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>12</td>
<td>5</td>
</tr>
<tr>
<td>15</td>
<td>6</td>
</tr>
<tr>
<td>18</td>
<td>7</td>
</tr>
<tr>
<td>21</td>
<td>8</td>
</tr>
<tr>
<td>24</td>
<td>9</td>
</tr>
</tbody>
</table>

Dadda, '65

4:2 Counters

4-2 carry-save module

Truth table for the 4:2 adder:

- \( n \) is the number of inputs (from \( I_{in1}, I_{in2}, I_{in3}, I_{in4} \)) which \( = 1 \), \( C_{in} \) is the input carry from the \( C_{in} \) of the adjacent bit slice, \( C_{out} \) and carry both have weight 2, and sum has weight 1.

<table>
<thead>
<tr>
<th>( n )</th>
<th>( C_{in} )</th>
<th>( C_{out} )</th>
<th>( Carry )</th>
<th>( Sum )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

*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.

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

UC Berkeley EE241

B. Nikolic
4:2 Counters

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

Carry Propagate Adder

Horizontal Propagation
Carry and Sum Connection to the Final Adder

UC Berkeley EE241
B. Nikolic
Example of Delay Optimization

Booth Encoding

- Instead of generating all the partial products
  \[ 0 \times x = 0 \quad 1 \times x = x \quad x = \{0,1\} \]
- Reduce the number of partial products by grouping

\[
\begin{array}{ccc}
0 & 0 & 0 \\
0 & 1 & 1^* \\
1 & 0 & 2^* \text{ (shift)} \\
1 & 1 & 3^* \text{ (or } 4^* -1) \\
\end{array}
\]
Booth Encoding

- 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 Encoding

Two bits and the MSB of previous two

<table>
<thead>
<tr>
<th>(x_{i+2} x_{i+1} x_i)</th>
<th>Add to partial product</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>+0Y</td>
</tr>
<tr>
<td>001</td>
<td>+1Y</td>
</tr>
<tr>
<td>010</td>
<td>+1Y</td>
</tr>
<tr>
<td>011</td>
<td>+2Y</td>
</tr>
<tr>
<td>100</td>
<td>-2Y</td>
</tr>
<tr>
<td>101</td>
<td>-1Y</td>
</tr>
<tr>
<td>110</td>
<td>-1Y</td>
</tr>
<tr>
<td>111</td>
<td>-0Y</td>
</tr>
</tbody>
</table>
Using Smaller Multipliers

\[
\begin{array}{c}
X_H & X_L \\
X & Y_H & Y_L \\
\hline
X_L \cdot Y_L & X_H \cdot Y_H \\
X_H \cdot Y_L & X_L \cdot Y_H \\
X_H \cdot Y_H & X_L \cdot Y_L
\end{array}
\]

Final Addition

Latest-Fewest Output Profile For TDM PPRT

UC Berkeley EE241  B. Nikolic
Final Addition

Optimal Uniform Input Adder
Hybrid Ripple-Carry/1-level Carry-Skip/Carry Select

Final Addition

Optimal Adder Input Profile
Optimal Level Carry/Skip Input for Uniform Input

Optimal Final Adder Design
Hybrid Ripple-Carry/Skip Carry-Skip/Carry Select

Final Addition

Optimal Final Adder Design
Hybrid Ripple-Carry/Skip Carry-Skip/Carry Select

UC Berkeley EE241 B. Nikolic
Example: CPL Multiplier

**Block Diagram**

Example: DPL Multiplier

**Critical Path**

Yano, JSSC 4/90

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

Final adder
Regularly Structured Tree

Sign-Select Booth Encoders
Sign-Select Booth Encoders

Regular Booth Selector

\[ M_j \]
\[ a_i \]
\[ x_j \]
\[ 2x_j \]
\[ a_{i-1} \]

\[ P_{i,j} \]

Modified Booth Selector

\[ e_{i,j} \]
\[ a_i \]
\[ a_{i-1} \]
\[ e_{i-2,j} \]
\[ p_{i-1,j} \]
\[ M_j \]
\[ p_{i,j} \]
\[ x_j \]
\[ 2x_j \]

(2 bit combined)

\[ P_{i,j} \]
\[ P_{i-1,j} \]

10 Tr./bit

Sign-Select Booth Encoders

4:2 Compressor

\[ P_1 \]
\[ P_2 \]
\[ P_3 \]
\[ P_4 \]

\[ c_o \]

48 Tr.

S
C

XNOR

C_{in}

UC Berkeley EE241

B. Nikolic