BOOTH ENCODING OF THE "MULTIPLIER" INPUT
Booth Encoding

- Method to reduce the number of partial products
- Named after Andrew Booth (1918-2009) who published the algorithm in 1951 while at Birkbeck College, London
- Booth-$n$
  - Examines $n+1$ bits of the multiplier
  - Encodes $n$ bits
  - $n \times$ reduction in the number of partial products
- But partial products must then be more complex than simply $0$ or $+\text{multiplicand}$
Booth Encoding: Booth-2 or “Modified Booth”

- Can view the *multiplier* as being built of strings of 1’s
  - Examine multiplier bits $Y_{i+1}$, $Y_i$, and $Y_{i-1}$
  - Perspective of moving right to left towards the MSB
- There are $\left\lceil \frac{N+2}{2} \right\rceil = \left\lceil \frac{N}{2} + 1 \right\rceil$ partial products in the worst case

<table>
<thead>
<tr>
<th>$Y_{i+1}$</th>
<th>$Y_i$</th>
<th>$Y_{i-1}$</th>
<th>Partial product</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>no string of 1’s</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>$+x$</td>
<td>end of string of 1’s</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>$+x$</td>
<td>a string of 1’s</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>$+2x$</td>
<td>end of string of 1’s</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>$-2x$</td>
<td>beginning of string of 1’s</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>$-x$</td>
<td>$-2x + x$</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>$-x$</td>
<td>beginning of string of 1’s</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>center of string of 1’s</td>
</tr>
</tbody>
</table>
Booth Encoding: Booth-2 or “Modified Booth”

- There are five possible partial products compared to two with non-Booth encoding

<table>
<thead>
<tr>
<th>$Y_{i+1}$</th>
<th>$Y_i$</th>
<th>$Y_{i-1}$</th>
<th>Partial product</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>no string of 1’s</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>+x</td>
<td>end of string of 1’s</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>+x</td>
<td>a string of 1’s</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>+2x</td>
<td>end of string of 1’s</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>−2x</td>
<td>beginning of string of 1’s</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>−x</td>
<td>−2x + x</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>−x</td>
<td>beginning of string of 1’s</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>center of string of 1’s</td>
</tr>
</tbody>
</table>

[Booth Encoding: Booth-2 or “Modified Booth”](Waser and Flynn)
Booth Encoding: Booth-2 or “Modified Booth”

- Fortunately, these five possible partial products are very easy to generate
  - The key issue is to not separate the 1) negation and 2) adding “1” LSB operations during the inversion process

- Correctly generating the –x and –2x PPs requires a little care

\[
\begin{align*}
+2x & \quad \text{multiplicand} \quad 0 \\
+x & \quad \text{multiplicand} \quad 0 \\
0 & \quad 0 \\
-x & \quad \sim\text{multiplicand} \quad 1 \\
-2x & \quad \sim\text{multiplicand} \quad 1
\end{align*}
\]
Booth Encoding: Booth-2 or “Modified Booth”

- Example: \( \text{multiplier} = 0010 = 2 \)
  - Add 0 to the right of the LSB since the first group has no group with which to overlap
  - Examine 3 bits at a time
  - Encode 2 bits at a time
    \( \rightarrow \) Overlap one bit between partial products

\[
\begin{array}{cc}
\text{s} & \text{s} \\
\hline
\text{+x} & 0 0 \\
\hline
-2x & \text{0 0}
\end{array}
\]

\[
4 \times (+x) - 2x \quad = \quad +2x
\]
Booth Encoding: Booth-2 or “Modified Booth”

- Example: \( \text{multiplier} = 1001 = -7 \)
  - Add 0 to the right of the LSB since the first group has no group with which to overlap
  - Examine 3 bits at a time
  - Encode 2 bits at a time
  - Overlap one bit between partial products

\[
4 \times (-2x) + x = -7x
\]
Booth Encoding: Booth-2 or “Modified Booth”

- **Example:** \( \text{multiplier} = 01111111 = +127 \)
  - Nice example of encoding a long string of 1’s
  - Examine 3 bits at a time
  - Encode 2 bits at a time

\[ 64 \times (+2x) + 16 \times (0) + 4 \times (0) - x \]
\[ = +127x \]
Booth Encoding: Booth-2 or “Modified Booth”

- **Example:** $multiplier = 10100110 = -90$
  - Examine 3 bits at a time
  - Encode 2 bits at a time

\[
\begin{align*}
64 \times (-x) + 16 \times (-2x) + 4 \times (+2x) - 2x & = -90x \\
\end{align*}
\]
Booth Encoding: Booth-2 or “Modified Booth”

- (Left side) *End* of a string of 1’s

\[
\begin{array}{c}
0 \\
\hline
0 0 1 1 1 1 1 1 \\
+ x & 0 \\
\end{array}
\]

- (Right side) *Beginning* of a string of 1’s

\[
\begin{array}{c}
0 \\
\hline
0 1 1 1 1 1 1 1 1 \\
+2 x & 0 \\
\end{array}
\]

\[
\begin{array}{c}
\cdots \\
\hline
\cdots 1 1 1 1 1 1 0 \\
0 & - x \\
\end{array}
\]

\[
\begin{array}{c}
\cdots \\
\hline
\cdots 1 1 1 1 1 0 0 \\
0 & - 2 x \\
\end{array}
\]
Booth Encoding: Booth-3

<table>
<thead>
<tr>
<th>$Y_{i+2}$</th>
<th>$Y_{i+1}$</th>
<th>$Y_i$</th>
<th>$Y_{i-1}$</th>
<th>Partial product</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>+x</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>+x</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>+2x</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>+2x</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>+3x</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>+3x</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>+4x</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-4x</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-3x</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-3x</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>-2x</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>-2x</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>-x</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>-x</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

[Waser and Flynn]