Lecture 13

Booth Multiplier (additional material)

Khaza Anuarul Hoque
ECE 4250/7250

Booth multiplier

For the standard add-shift operation, each multiplier bit generates one multiple of the multiplicand to be added to the partial product. If the multiplier is very large, then a large number of multiplicands have to be added. In this case the delay of multiplier is determined mainly by the number of additions to be performed. If there is a way to reduce the number of the additions, the performance will get better.

Booth algorithm is a method that will reduce the number of multiplicand multiples. For a given range of numbers to be represented, a higher representation radix leads to fewer digits. Since a k-bit binary number can be interpreted as K/2-digit radix-4 number, a K/3-digit radix-8 number, and so on, it can deal with more than one bit of the multiplier in each cycle by using high radix multiplication. This is shown for Radix-4 in the example below.

Radix-4

Table .1 Radix-4 Booth recoding

X_{i+1}	X	X_{i-1}	$Z_{i/2}$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	2
1	0	0	-2
1	0	1	-1
1	1	0	-1
1	1	1	0

$$\mathbf{Z_i = -2x_{i+1} + x_i + x_{i-1}}$$

Example-1

$$
\begin{array}{rll}
 & \quad\;\; 111101 & (-3) \\
\times & \quad\;\; 100011\boxed{0} & (-29) \\
 & \quad\;\;\; -2\ +1\ -1 & \\
\hline
 & 00000000011 & \\
\text{Shifted 2s complement} \rightarrow & 1111111101 & \\
 & 00000110 & \\
\hline
1 \leftarrow & 000001010111 & (+87)
\end{array}
$$

Example-2

Example

Using Booth algorithm multiply A and B.

A= 20

B=30

A= 0010100
B= 0011110

Please note that both numbers are extended to cover 2A or 2B and the sign bit (whichever is larger).

A * B =

A= 0 0 1 0 1 0 0

-0

B= 0 0 1 1 1 1 0 0

+2 -2

2A = 40 = 00101000

-2A = 11011000

Now performing the addition we have

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

512 + 64 + 16 + 8 = $(600)_{10}$