

BINARY MULTIPLICATION

(1) 4-BIT UNSIGNED

$$\begin{array}{r}
 1101 = 13 \\
 x 1011 = 11 \\
 \hline
 1101 \\
 0000 \\
 0000 \\
 1101 \\
 \hline
 10001111 = 143
 \end{array}$$

(2) above example with blanks filled in

$$\begin{array}{r}
 1101 = 13 \\
 x 1011 = 11 \\
 \hline
 00001101 \\
 00011010 \\
 00000000 \\
 01101000 \\
 \hline
 10001111 = 143
 \end{array}$$

(3) 4-BIT / SIGNED TWOS-COMPLMNT

$$\begin{array}{r}
 1101 = -3 \\
 x 1011 = -5 \\
 \hline
 00001101 \\
 00011010 \\
 00000000 \\
 01101000 \\
 \hline
 10001111 = -113???
 \end{array}$$

(4) PROBLEMS

- (a) 0 0 0 0 1 1 0 1 does not represent -3
 The partial products must be "sign-extended"

$$\begin{array}{r}
 1101 = -3 \\
 x 1011 = -5 \\
 \hline
 11111101 \\
 11111010 \\
 00000000 \\
 11101000 \\
 \hline
 11011111 = -33??? \quad (-3 \times +11)
 \end{array}$$

- (b) to make 1 0 1 1 equal -5, the MSB must have a negative weight. $-1 \times 8 + 0 \times 4 + 1 \times 2 + 1 \times 1 = -5$. This means we need to take the twos complement of the last partial product.

differently.

<p>The above example is repeated using a register than is $2n+1$ bits wide (n is the number of bits in the numbers). We start with UPPER AC and B set to 0 and LOWER AC the multiplier. Using the LSB of LOWER AC and B, we may use Booth's rule for adding, subtracting or doing nothing. Then we do an arithmetic right shift. We repeat for all bits of the multiplier. UPPER and LOWER AC then contain the product.</p>	OP.	UPPER AC				LOWER AC				B
	start	0	0	0	0	1	0	1	1	0
	10: sub	0	0	1	1					
	shift	0	0	0	1	1	1	0	1	1
	11: noop									
	shift	0	0	0	0	1	1	1	0	1
	01: add	1	1	0	1					
	shift	1	1	1	0	1	1	1	1	0
	10: sub	0	0	0	1					
	shift	0	0	0	0	1	1	1	1	1