Multiplication.

Introduction: multiplication involves two basic operations: the generation of partial products and their accumulation.

First part we will discuss multiplication with unsigned numbers and second part we will discuss multiplication with two's compliant numbers:

First part: Multiplication with unsigned numbers:

Given any inputs $a, b \in \{0, 1\}$

where $a = a_{n-1} \ldots a_0$

$b = b_{n-1} \ldots b_0$

Output of multiplier will be $p \in \{0, 1\}^{2n}$

$$\text{where} \quad \langle p \rangle = \langle a \rangle \times \langle b \rangle = \sum_{i,j=0}^{n-1} a_i \cdot b_j \cdot 2^{i+j}$$

<u>Basic sequential multiplication:</u> Multiplication consist of generating all partial products and then accumulating all the partial products.

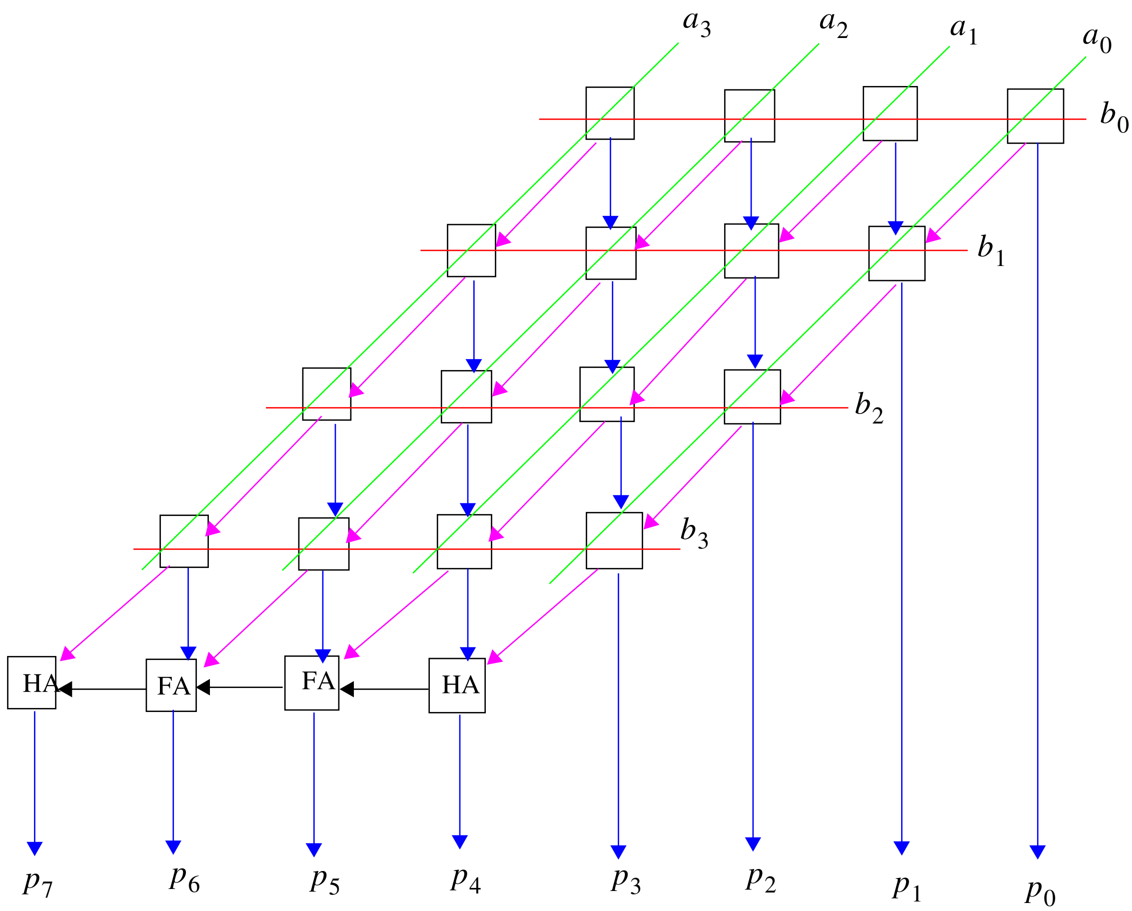
$$\begin{array}{r}
a_{n-1} \ldots\ldots\ldots\ldots\ldots\ldots\ldots a_0 \\
b_{n-1} \ldots\ldots\ldots\ldots\ldots\ldots\ldots b_0 \\
\hline
b_0 a_{n-1} \ldots\ldots\ldots\ldots b_0 a_2 b_0 a_1 b_0 a_0 \\
b_0 a_{n-1} \ldots\ldots\ldots\ldots b_0 a_2 b_0 a_1 b_0 a_0 \quad \\
\\
b_0 a_{n-1} \ldots\ldots\ldots\ldots b_0 a_2 b_0 a_1 b_0 a_0 \qquad\qquad\qquad\qquad \\
\hline
p_{2n-1} p_{2n-1} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots p_2 p_1 p_0
\end{array}$$

Note: sequential multiplication takes quadratic time to fill up the all matrix and additional quadratic time to compute all the partial products.

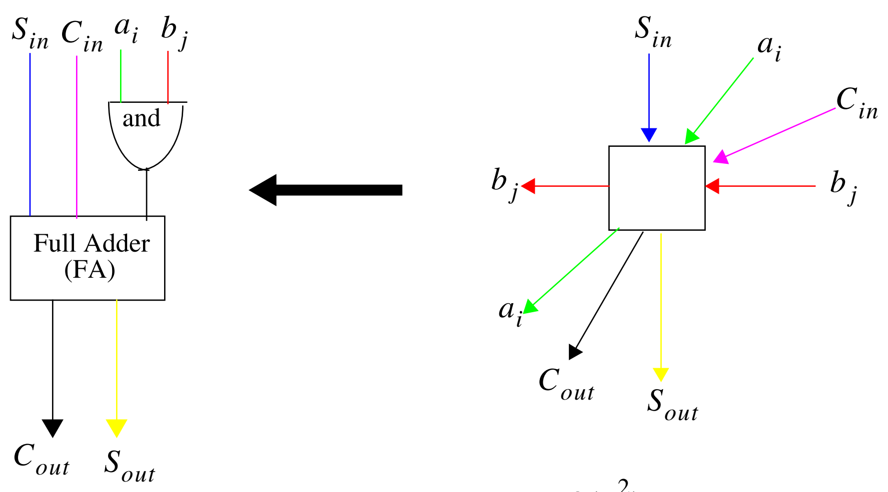
Array Multipliers:

In array multiplier we can merge the two basic operation, the generations of partial products and their summation. However, the running timing of array multipliers is linear.

Block Diagram of array multiplier for unsigned numbers:



Detailed a multiply and add box description:



$$S_{in} + C_{in} + a_{in}b_{in} = S_{out} + 2C_{out}$$

Note: Cost is $O(n^2)$

Delay $O(n)$

Multiplication of unsign numbers with logarithmic delay time and cost n-square.

First question is asked what is the difference between the addition of partial products and addition.

Answer: In addition we add up two numbers. However, in multiplication we reduce n-partial product to two, then we add two number as usual.

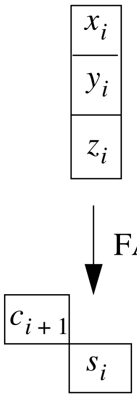
In carry save format every digit is represented by two bits a_i and b_i the value of digit is $a_i + b_i$. We can look at carry save format as the extended binary digit set {0,1,2}

Carry Save Format.

a_i	0	0	1	1
b_i	0	1	0	1
	0	1	1	2

Step I. Reduction n-partial products to Carry Save Format. we will prove by inductive approach.

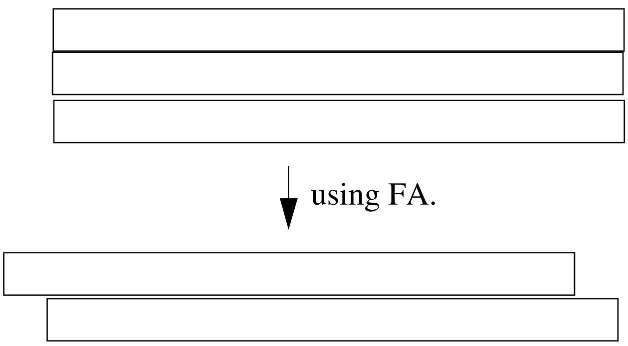
Let n=3 we want to trasform it to carry save format. The simplest way to do it, is to use full adder.



Following equation used for transfer three bit vector to carry save format.

$$x_i + y_i + z_i = s_i + 2 \cdot c_{i+1}$$

In general we can transfer given three vectors to carry save format in constant time and linear cost.



When m>3 we will divide by three m-partial products and reduced it to carry save format:

$\left\lceil \frac{m}{3} \right\rceil$ - number of groups divided by three.

$2 \cdot \left\lceil \frac{m}{3} \right\rceil$ - last operation we will end up only in carry save format number

Total time it takes to reduced from n-partial product to CSF is:

$$T(n, m) = T\left(n + 1, \left\lceil \frac{m}{3} \right\rceil\right) + Del(FA)$$

where: n is the lenght of binary string
m is a group of the lenght of binary strings.

Claim: Total delay time will be equal: $T(n, m) \le \left(lg_{\frac{3}{2}} m\right) + Del(FA)$

Cost for reduction n-partial products to carry save format is:

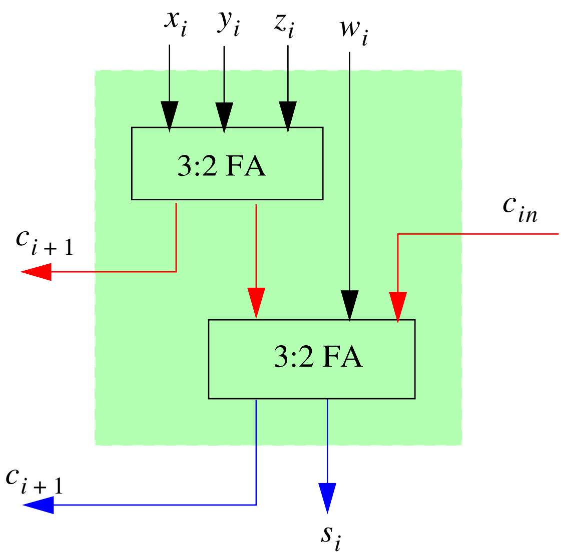
$$C(n, m) = C\left(n + 1, 2\left\lceil \frac{m}{3} \right\rceil\right) + n \cdot \cos t(FA)$$

$$C(n, m) = O\left(n \cdot lg_{\frac{3}{2}} m\right)$$

4:2 Reduction Tree.

Using 4:2 counter we can reduced 4-partial products to carry save format. The 4:2 counter must be designed so that Cout is not a function of Cin, in order to avoid a ripple-carry effect. The simplest way to built 4:2 counter is to use two 3:2 full adders in following way.

4:2->(x,y,z,w)=3:2(3:2(x,y,z),w);



A (4:2) counder using two (3:2) adders.

An Array Multiplier for Two's Complement Numbers.

Given vector a in floating point notation

$$a = a_0 a_{-1} a_{-2} \ldots a_{-(n-1)}$$

In binary format we can write the radix polynomial for a following:

$$\langle a \rangle = a_0 + \sum_{i=1}^{n-1} a_{-i} \cdot 2^{-i}$$

In two's complement radix polynomial for a is following:

$$[a] = -a_0 + \sum_{i=1}^{n-1} a_{-i} \cdot 2^{-i}$$

Our purpose to compute the p-two's complement multiplication p=[a]*[b];
Only difficulty using this technique is that we have two kinds of partial products:

1- Positive partial product

$$a_i \cdot b_i \Rightarrow i = 0, j = 0$$

or

$$a_i \cdot b_i \Rightarrow i < 0, j < 0$$

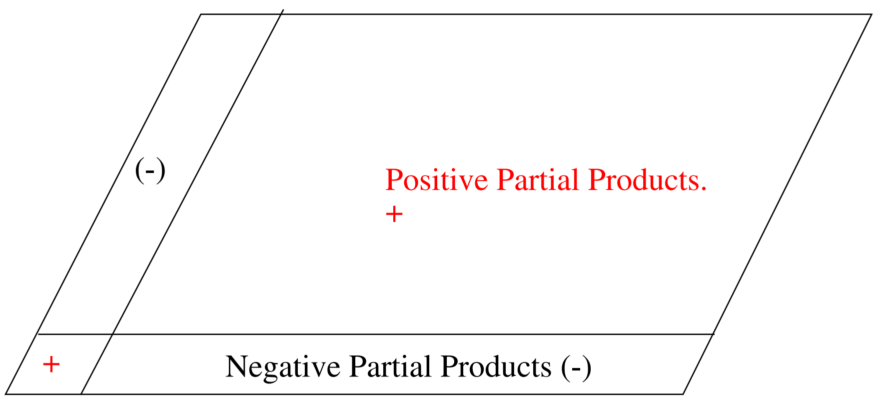
2) Negative partial product

$$a_i \cdot b_i \Rightarrow i < 0, j = 0$$

or

$$a_i \cdot b_i \Rightarrow i = 0, j < 0$$

Therefore, it is difficult to add all partial products because we have positive and negative products: see partial reduced tree:



There are two ways to accumulate the partial products.

Technique #I: We will compute the modulo 4 multiplication:

Let $\pi = [a] \cdot [b]$

Note: $\pi \in [-1, 1) \cdot [-1, 1) = (-1, 1]$

Final product will be represented following way:

Let: $p = p_1 p_0, p_{-1} \cdots\cdots p_{-(2n-2)}$

Claim: if $\pi = \langle p \rangle (mod 4)$ then $\pi = [p]$

Proof: $\pi \in (-1, 1]$ and $\langle p \rangle = [0, 4)$

If $\pi \geq 0$ then p is in the following ranges: $\pi \in [0, 1]$ => $\langle p \rangle = \pi (mod 4)$

follows that $\pi = \langle p \rangle \Rightarrow \pi = [p]$ because p is positive and less then one

we will get that $\langle p \rangle = [p]$

if $\pi < 0$ then $\pi \in (-1, 0)$ and $\langle p \rangle \in [0, 4)$

Follows that $\langle p \rangle = \pi (mod 4) \Rightarrow \langle p \rangle = \pi + 4 \Rightarrow \pi = -4 + \langle p \rangle = [1p]$

Two cases can happened:

Case I $p_1 = 1$ then $[1.p] = [p]$

1	p_1	p_0	p_{-1}	p_{-2}

because we can be able to do extension to sign.

Case II $p_1 = 0$ this case cannot be possible because $\langle p \rangle = \pi + 4 \in 4 - (1, 0) = (3, 4)$

Conclusion it is enough to calculate $p = p_1 \cdot p_0 \bullet p_{-1} \cdots p_{-2(n-1)}$

Such that $\pi \in \langle p \rangle (mod 4)$

Techniques: Given $\hat{a} = a_{-1} \cdots\cdots\cdots a_{-n-1}$

$\hat{b} = b_{-1} \cdots\cdots\cdots b_{-n-1}$

$$\pi = [a] \times [b] = (-a_0 + \langle \hat{a} \rangle) \cdot (-b_0 + \langle \hat{b} \rangle) = a_0 b_0 - a_0 \langle \hat{b} \rangle - b_0 \langle \hat{a} \rangle + \langle \hat{a} \rangle \langle \hat{b} \rangle$$

Let $\langle A \rangle = \langle \hat{a} \rangle \langle \hat{b} \rangle$

$\langle B \rangle = 2 - b_0 \langle \hat{a} \rangle$

$\langle C \rangle = 2 - a_0 \langle \hat{b} \rangle$

$\langle D \rangle = a_0 b_0$

$$\pi = (\langle A \rangle + \langle B \rangle + \langle C \rangle + D)(mod 4)$$

Graphical Representation

$$\langle B\rangle = 2 - b_0\langle a'\rangle = 1 + 1 - \langle b_0a_{-1}, b_0a_{-2}\ldots\ldots b_0a_{-n+1}\rangle$$

$$\langle B\rangle = 1 + (\overline{b_0a_{-1}}, \overline{b_0a_{-2}}\ldots\ldots\overline{b_0a_{-n+1}}) + 2^{-(n-1)}$$

$$\langle C\rangle = 1 + 2^{-(n-1)} + \langle\overline{a_0b_{-1}}\ldots\ldots\ldots\overline{a_0b_{-n+1}}\rangle$$

Partial products tree: