FREE IP CORE OVERVIEW

Integer multiplication algorithms

Methodology and implementation results

Author:
Vladimir V.Erokhin, Ph.D, MSEE.

Preface

Operation of multiplication is very important in microelectronics. Each modern microprocessor has this operation within its instruction set, and advanced microprocessors have special multiplication units, that perform multiplication during 1 synchronization period (cycle). Especially valuable multiplication is in DSP processors, where it is practically main operation. Performance of any DSP processor is defined with delays in it MAC (multiply and accumulate) unit. So efficiency of multiplication is very important.

Methodology Overview

There are different approaches of implementation of multiplication per one clock: from simple methods of multiple sums to matrix algorithms. Here we will propose the group of similar algorithms and will compare their characteristics with built-in Synopsys algorithm and well known “school” multiplication algorithm.

The idea of algorithms is as follows. Unsigned multiplicands A and D may be represented in following form: $A*D = (B * 2^n + C) * (E * 2^n + F)$, where n – any number that is satisfied with following conditions :

1) $2^n < A$;
2) $2^n < D$;
3) $C < 2^n$;
4) $F < 2^n$.

This approach is applied recursively to all multiplicands until multiplication result may be calculated easily (for example, until multiplicands have dimension of one or two bits).

For example,

$69 * 27 = (8 * 2^3 + 5)(2 * 2^3 + 9) = (8 * 2) * 2^6 + (8 * 9$
$+ 2 * 5) * 2^3 + 5 * 9;$

$8 * 2 = (1 * 2^3 + 0) (0 * 2^3 + 2) = \ldots$
$8 * 9 = (1 * 2^3 + 0) (1 * 2^3 + 1) = \ldots$
etc.

Results of different algorithms implementation based on this approach are represented below. It is supposed that m – multiplicands dimension in bits.
Synthesis of all algorithms was done with the same conditions and the same technology library. During optimization the time was critical.

Algorithms

«Pyramid» algorithm.

Have a look at basic formula $A*D = (B * 2^n + C) * (E * 2^n + F)$. In case $n=m-1$, C and D have dimension of one bit. This basic formula is applied recursively to all further multiplicands. As a result dimension of multiplicands is decreased by one at every iteration. That is why the algorithm was named as "pyramid".

Synthesis result is represented in Table 1.

Operands dimension	Delay (ns)	Combinational area (gates)	Delay*area
8*8	9.80	890	8722
16*16	19.85	2815	55878
32*32	37.34	10550	393937

Table 1 Pyramid algorithm implementation

Modified «pyramid» algorithm.

Modified «pyramid» algorithms is differ from prototype with value of $n = m-2$ and with dimension of operands C и D equal to 2 bits. As may be seen in Table 2 **Modified pyramid algorithm implementation** such small change gives valuable results improvement.

Operands dimension	Delay (ns)	Combinational area (gates)	Delay*area
8*8	9.92	700	6944
16*16	17.70	2300	40710
32*32	33.94	8580	291205
64*64	69.78	33300	2323670

Table 2 Modified pyramid algorithm implementation

«Hierarchical» algorithm.

As it follows from theory of algorithms maximum of timing efficiency should be expected when dimensions of operands **B**, **C**, **E** and **F** (see basic formula) are equal at every algorithm call, i.e. **n=m/2**. In this case number of recursions will be minimal and number of sums that take part in final result also will be minimal. Table 3 **Hierarchical algorithm implementation** represents results of the synthesis.

Operands dimension	Delay (ns)	Combinational area (gates)	Delay*area
8*8	9.56	760	7266
16*16	15.15	2505	37951
32*32	23.12	9355	216288
64*64	35.34	33805	1194669

Table 3 Hierarchical algorithm implementation

Modified «hierarchical» algorithm.

This algorithm is an attempt to improve "hierarchical" algorithm for long-dimensional operands by substitution of one multiplication with some of addition operations. This algorithm in intended for operands 128-bit width and more so the algorithm advantages appears for **m** → ∞, where possibly the algorithm may be preferable than the prototype.

The algorithm characteristics are represents in Table 4 **Modified hierarchical algorithm implementation**.

Operands dimension	Delay (ns)	Combinational area (gates)	Delay*area
8*8	14.28	1015	14494
16*16	21.76	3585	78010
32*32	33.85	11240	380474
64*64	56.48	30368	1715185

Table 4 Modified hierarchical algorithm implementation

Built-in Synopsys algorithm results.

For the comparison Synopsys built-in CSA (Carry Save Array) algorithm characteristics are represents below. The synthesis was done using the same conditions and the same library as for other algorithms.

Operands dimension	Delay (ns)	Combinational area (gates)	Delay*area
8*8	9.33	740	6904
16*16	17.01	2940	50009
32*32	32.33	10010	323623
64*64	64.67	27370	1770018

Table 5 Built-in Synopsys algorithm implementation

School («direct») algorithm.

Synthesis results of this algorithm are shown as exotics, so the young designer may understand how far modern science from the school is.

Operands dimension	Delay (ns)	Combinational area (gates)	Delay*area
8*8	13.62	1140	15527
16*16	27.16	5265	142997
32*32	65.00	13000	845000

Table 6 Direct algorithm implementation

Discussion.

Absolute leader among all algorithms is **hierarchical** multiplication algorithm. This algorithm very recommended in designs, where timing is critical.

In cases where area is more important than timing, modified **pyramid** algorithm is preferred. In comparison with built-in Synopsys algorithm for operands dimension up to 32 bits where **pyramid’**s delay is about the same while area consumption is better.

Table 7 **Delay (ns) characteristic of multiplication algorithms**represents synthesis results of all algorithms for timing and Table 8 **Area (gates) characteristic of multiplication algorithms**represents synthesis results of all algorithms for area (remember that during optimization the timing was critical).

	Algorithms					
Operands dimension	hierarchical	modified hierarchical	pyramid	modified pyramid	direct	built-in Synopsys
8*8	9.56	14.28	9.80	9.92	13.62	9.33
16*16	15.15	21.76	19.85	17.70	27.16	17.01
32*32	23.12	33.85	37.34	33.94	65.00	32.33
64*64	35.34	56.48	-	69.78	-	64.67

Table 7 Delay (ns) characteristic of multiplication algorithms

	Algorithm					
Operands dimension	hierarchical	modified hierarchical	pyramid	modified pyramid	direct	built-in Synopsys
8*8	760	1015	890	700	1140	740
16*16	2505	3585	2815	2300	5265	2940
32*32	9355	11240	10550	8580	13000	10010
64*64	33805	30368	-	33300	-	27370

Table 8 Area (gates) characteristic of multiplication algorithms

For algorithms comparison we may propose integral characteristic such as multiplication of delay and area (the less is the better). The following Table 9 represents this characteristic of algorithm. Again absolute leader in performance is “hierarchical” algorithm for all operands dimension except 8 bit operands.

	Algorithm					
Operands dimension	hierarchical	modified hierarchical	pyramid	modified pyramid	direct	built-in Synopsys
8*8	$7.25*10^3$	$14.49*10^3$	$8.72*10^3$	$6.94*10^3$	$15.52*10^3$	$6.90*10^3$
16*16	$3.80*10^4$	$7.80*10^4$	$5.59*10^4$	$4.07*10^4$	$14.30*10^4$	$5.00*10^4$
32*32	$2.16*10^5$	$3.80*10^5$	$3.94*10^5$	$2.91*10^5$	$8.45*10^5$	$3.24*10^5$
64*64	$1.19*10^6$	$1.72*10^6$	-	$2.32*10^6$	-	$1.77*10^6$

Table 9 (delay * area) characteristic of multiplication algorithms