

# On the circuit complexity of the standard and the Karatsuba methods of multiplying integers\*

Igor S. Sergeev<sup>†</sup>

The goal of the present paper is to obtain accurate estimates for the complexity of two practical multiplication methods: standard (school) and Karatsuba [1]. Here, complexity is the minimal possible number of gates in a logic circuit implementing the required function over the basis  $\{AND, OR, XOR, NAND, NOR, XNOR\}$ .

One can find upper estimates for the said methods e.g. in [2]. The standard method has complexity  $M(n) \leq 6n^2 - 8n$ . In the case  $n = 2^k$ , the complexity  $K(n)$  of the Karatsuba method can be deduced from the recursion  $K(2n) \leq 3K(n) + 49n - 8$  as  $K(2^k) \leq 26\frac{2}{9} \cdot 3^k - 49 \cdot 2^k + 4$ .

We intend to show that the above estimates may be improved with the help of the result [3] stating that a sum of  $n$  bits may be computed via  $4.5n$  operations instead of  $5n$  as in the naive approach. The resulting bounds are  $M(n) \leq 5.5n^2 - 6.5n - 1 + (n \bmod 2)$  and

$$K(2^k) \leq 25\frac{83}{405} \cdot 3^k - 38 \cdot 2^k - \frac{81}{5}\Phi_{k+2} - \frac{37}{5}\Phi_{k+1} + 20, \quad (1)$$

where  $\{\Phi_k\}$  is the Fibonacci sequence:  $\Phi_1 = \Phi_2 = 1$ ,  $\Phi_{k+2} = \Phi_{k+1} + \Phi_k$ .

**Auxiliary circuits.** The circuits below are built from the following sub-circuits: half-adders  $HA$ ,  $HA^\pm$ ,  $(3, 2)$ -carry save adders (CSA)  $FA_3$ ,  $FA_3^-$ ,  $FA_3^0$ ,  $SFA_3$ ,  $SFA_3^-$ ,  $(5, 3)$ -CSA  $M DFA$ ,  $M DFA^-$  (the  $M DFA$  circuit was proposed in [3]). Specifically, they implement the functions:

$$\begin{aligned} HA: (x_1, x_2) &\rightarrow (u; v), \text{ where } x_1 + x_2 = 2u + v; \\ HA^\pm: (x_1, x_2) &\rightarrow (u; v), \text{ where } x_1 - x_2 = -2u + v; \\ FA_3: (x_1, x_2, x_3) &\rightarrow (u; v), \text{ where } x_1 + x_2 + x_3 = 2u + v; \\ FA_3^-: (x_1, x_2, x_3) &\rightarrow (u; v), \text{ where } x_1 + x_2 - x_3 = 2u - v; \\ FA_3^0: (x_1, x_2, x_3) &\rightarrow (u; v), \text{ where } x_1 + x_2 - x_3 = 2u + v, \text{ if } x_1 + x_2 - x_3 \geq 0; \end{aligned}$$

---

\*Translated from the Russian original published in: Proc. of XXII Conf. "Information means and technology" (Moscow, November 18–20, 2014). Vol. 3. Moscow, MPEI, 2014, 180–187.

<sup>†</sup>e-mail: isserg@gmail.com

$SFA_3$ :  $(x_1, x_1 \oplus x_2, x_3) \rightarrow (u; v)$ , where  $x_1 + x_2 + x_3 = 2u + v$ ;  
 $SFA_3^-$ :  $(x_1, x_1 \oplus x_2, x_3) \rightarrow (u; v)$ , where  $x_1 - x_2 + x_3 = 2u - v$ ;  
 $M DFA$ :  $(x_1, x_1 \oplus y_1, x_2, x_2 \oplus y_2, z) \rightarrow (u_1, u_1 \oplus u_2; v)$ , where  $x_1 + y_1 + x_2 + y_2 + z = 2(u_1 + u_2) + v$ ;  
 $M DFA^-$ :  $(x_1, x_1 \oplus y_1, x_2, x_2 \oplus y_2, z) \rightarrow (u_1, u_1 \oplus u_2; v)$ , where  $x_1 - y_1 + x_2 - y_2 + z = 2(u_1 - u_2) + v$ .

These circuits are shown on Fig. 1. Gates  $AND$ ,  $OR$ ,  $XOR$  are denoted by symbols  $\wedge$ ,  $\vee$ ,  $\oplus$ , respectively. Inverted inputs are marked by small circles.

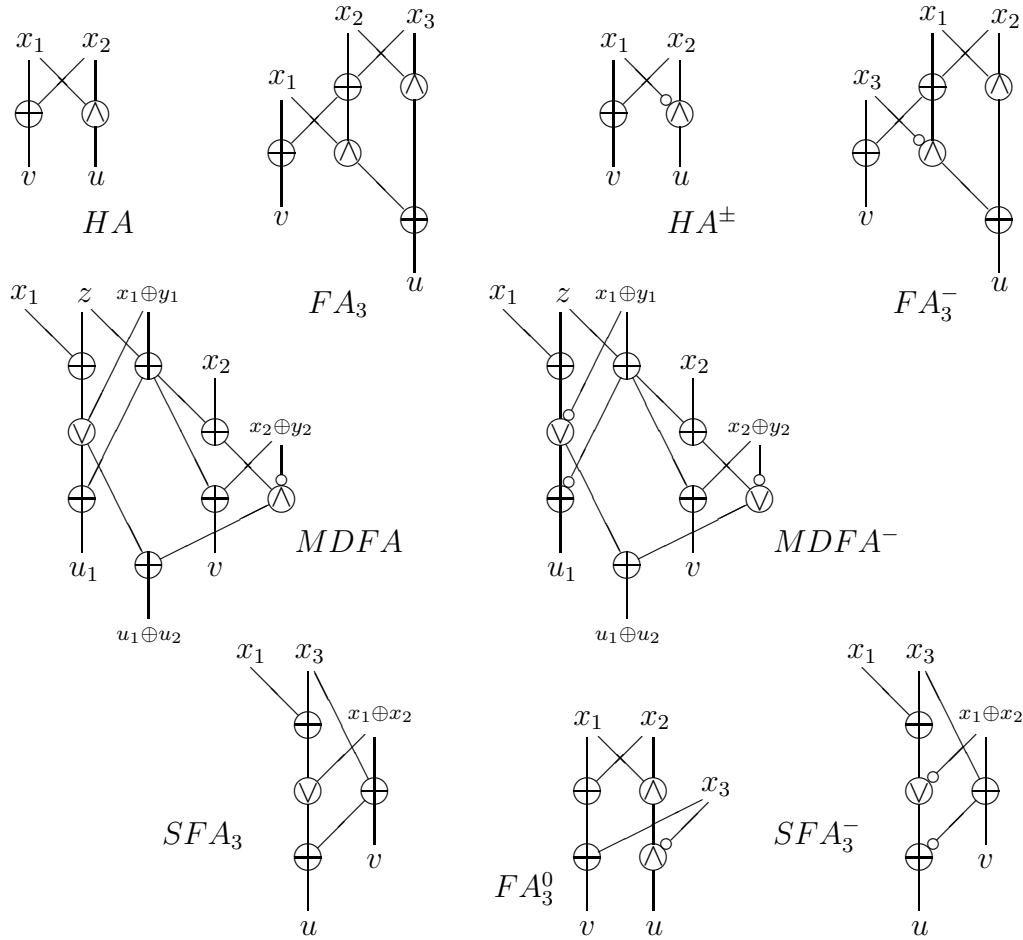


Figure 1: Auxiliary circuits

**Standard method.** The first stage of the standard method of multiplication of  $n$ -bit integers involves  $n^2$  bit multiplications. The next stage — multiple addition — may be performed via summation of bits in consecutive columns (column is a set of bits of the same order). The summation utilizes the aforementioned auxiliary subcircuits. The result of a column summation

is a bit of the product and a set of carries to the next order.

Let us index columns from 1 in increasing order. Then, after the first phase one has  $n - |n - k|$  bits in a  $k$ -th column,  $k = 1, \dots, 2n - 1$ . Consider the following rule of column summation: if there exist 5 summand bits, use *MDFA*; else, if there exist 3 summand bits, use  $FA_3$ ; else, if there exist 2 summand bits, use *HA*.

Denote by  $h(k)$  number of summand bits in the  $k$ -th column after completion of summation in all lower-order columns. Clearly,  $h(1) = 1$ . Let us check by induction that  $h(k) = 2k - 2$  for  $2 \leq k \leq n$ . Obviously, the statement holds for  $k = 2$ . Assume, it also holds for  $k = t$  and consider summation in the  $t$ -th column. By the declared strategy, summation of  $2t - 2$  bits involves  $\lfloor t/2 \rfloor - 1$  circuits *MDFA*, one circuit *HA*, and in the case of odd  $t$ , one more circuit  $FA_3$ . In total, it produces  $t - 1$  carries to the next order. Hence,  $h(t + 1) = t + 1 + t - 1 = 2(t + 1) - 2$ , as required.

By analogy, we conclude that  $h(n + 1) = 2n - 2$  and  $h(2n - k) = 2k + 1$  for  $0 \leq k \leq n - 2$ . For summation in the  $(n + 1)$ -th column one use the same set of circuits as for the  $n$ -th column. For summation in  $(2n - k)$ -th column we use  $\lfloor k/2 \rfloor$  circuits *MDFA*, and in the case of odd  $k \neq 1$  an additional circuit  $FA_3$ . For  $k = 1$  we need a circuit  $SFA_3$  instead of  $FA_3$ , since summation in the previous column involves *MDFA*.

The use of *MDFA* requires a conversion to the special bit encoding  $(x, y) \rightarrow (x, x \oplus y)$ . All *MDFA* outputs encoded this way may be connected to *MDFA* inputs of the same encoding, with the exception of the last *MDFA*, in the  $(2n - 2)$ -th column. Therefore, to execute all summations we need additionally  $q + 1$  *XOR* gates, where  $q$  is a number of *MDFA*.

Therefore, if  $n \geq 4$ , then the second stage of multiplication utilizes  $n$  circuits *HA*,  $n - 3 + 2(n \bmod 2)$  circuits  $FA_3$ , one circuit  $SFA_3$ ,  $q = (n^2 - 3n)/2 + 1 - (n \bmod 2)$  circuits *MDFA* and  $q + 1$  *XOR* gates (the number of *MDFA* is easy to derive from the number of  $(3, 2)$ -CSA, since *MDFA* reduces the total number of summand bits by 2,  $FA_3$  or  $SFA_3$  reduces it by 1; the number of summand bits before summation stage is  $n^2$ , and at the end it is  $2n$ ). Summing up the complexities of subcircuits we can bound the complexity  $M(n)$  of the multiplication circuit as

$$M(n) \leq 5, 5n^2 - 6, 5n - 1 + (n \bmod 2).$$

The estimate holds also for  $2 \leq n \leq 3$ .

**Karatsuba method.** Represent two  $m$ -bit multiplication operands as  $A_1 2^n + B_1$  and  $A_2 2^n + B_2$ , where  $n = \lceil m/2 \rceil$ ,  $0 \leq B_i < 2^n$ ,  $0 \leq A_i < 2^{m-n}$ . Then, the product may be computed by the formula:

$$A_1 A_2 2^{2n} + ((A_1 + B_1)(A_2 + B_2) - A_1 A_2 - B_1 B_2) 2^n + B_1 B_2.$$

The implied circuit consists of two addition circuits computing  $A_1 + B_1$  and  $A_2 + B_2$ , three multiplication subcircuits for  $(n + 1)$ -bit,  $(m - n)$ -bit and  $n$ -bit operands, and a subcircuit for the final addition-subtraction. The structure of this final addition is shown on the pattern below (see Fig. 2). Symbols “+” and “−” denote summand and subtrahend bits, respectively. Columns are indexed so that an index  $i$  corresponds to a bit with weight  $2^i$ . Pairs of bits in brackets are missing when  $m$  is odd.

$$\begin{array}{cccccccc}
(++) + \cdots + & + & + & + & + & + & \cdots + & A_1 A_2 2^{2n} \\
& + & + & + & + & + & \cdots + & (A_1 + B_1)(A_2 + B_2) 2^n \\
& & (-) & - & \cdots & - & - & - A_1 A_2 2^n \\
& & & - & - & \cdots & - & - B_1 B_2 2^n \\
& & & & & + & \cdots + & B_1 B_2 \\
& & & & & + & \cdots + & \\
4n-1 & & 3n-1 & & 2n-1 & & n & 0
\end{array}$$

Fig. 2. Pattern of the final addition-subtraction in the Karatsuba method

Consider the following column summation rule for the final addition: if there exist 3 summand bits and 2 subtrahend bits, use  $M DFA^-$ ; else, if there exist 3 bits, use a suitable circuit from the  $FA_3$  family; else, if there exist 2 bits, use an appropriate circuit from the  $HA$  family.

Denote by  $h^+(k)$ ,  $h^-(k)$  numbers of summand and subtrahend bits in the  $k$ -th column after completion of summation in all lower-order columns. One can easily verify that  $h^+(n) = h^-(n) = 2$ ,  $h^+(n + 1) = h^-(n + 1) = 3$  and  $h^+(k) = 3$ ,  $h^-(k) = 4$  for  $n + 2 \leq k \leq 2m - n - 1$ . We use one  $FA_3^-$  and one  $HA$  in the  $n$ -th column, one  $M DFA^-$  and one  $HA^\pm$  in the  $(n + 1)$ -th column, one  $M DFA^-$  and one  $FA_3^-$  in any subsequent column up to  $(2m - n - 1)$ -th.

When  $m$  is odd, we have  $h^+(k) = h^-(k) = 3$  for  $k = 3n - 1, 3n - 2$ . Therefore, one  $M DFA^-$  and one  $HA^\pm$  should be used in the corresponding columns.

Further,  $h^+(3n) = 3$ ,  $h^-(3n) = 2$  (use  $M DFA^-$ ),  $h^+(3n + 1) = 3$ ,  $h^-(3n + 1) = 1$  (use  $SFA_3^-$  and  $HA^\pm$ ),  $h^+(3n + 2) = 2$ ,  $h^-(3n + 2) = 1$  (use  $FA_3^0$ , since the value  $(A_1 + B_1)(A_2 + B_2) - A_1 A_2 - B_1 B_2$  is non-negative). At last,  $h^+(k) = 2$ ,  $h^-(k) = 0$  for  $3n + 3 \leq k \leq 2m - 1$ : use  $HA$  elsewhere, but use  $XOR$  in the most significant column, since no carry is required there.

As in the standard method, conversion to the special pair-of-bits encoding requires  $q + 1$  additional  $XOR$  gates, where  $q$  is the number of  $M DFA^-$  subcircuits.

Now, we're going to estimate the complexity  $K(m)$  of the multiplication circuit. It's a common knowledge, that the complexity of the addition of two  $n$ -bit numbers is  $5n - 3$ , the addition of an  $n$ -bit number and an  $(n - 1)$ -bit number has complexity  $5n - 6$  (see e.g. [2]). If  $m \geq 10$ , then the final

Karatsuba adder-subtractor contains  $n - 1$  circuits  $HA$  or  $HA^\pm$ ,  $2m - 2n - 1$  circuits  $FA_3^-$ , one circuit  $SFA_3^-$ , one circuit  $FA_3^0$ ,  $2n$  circuits  $M DFA^-$  and  $2n + 2$   $XOR$  gates.

Note, that one can save some gates. Columns indexed by  $n + i$  and  $2n + i$ , for  $i = 0, \dots, n - 1$ , contain identical pairs of bits (from summands  $B_1 B_2$ ,  $-A_1 A_2 2^n$  and  $-B_1 B_2 2^n$ ,  $A_1 A_2 2^{2n}$ , respectively). Arrange the computation process to pass these bits to the inputs of  $FA_3^-$  in columns indexed by  $n$  and  $n + 2, \dots, 2m - n - 1$ , and to the inputs of  $M DFA^-$  encoded by  $(x, x \oplus y)$  in other columns (that is, in  $(n + 1)$ -th column, and in the case of odd  $m$ , also in  $(3n - 2)$ -th and  $(3n - 1)$ -th columns).

Thus, for  $i = 0$  and  $2 \leq i \leq n - 1 - 2(m \bmod 2)$  we can save two gates  $XOR$  and  $ANDNOT$  via exploiting a CSA from Fig. 3a in a lower-order column and a CSA from Fig. 3b in a higher-order column (CSA's are different since the signs of bits in low-order and high-order columns are rearranged). For  $i = 1$  and  $n - 2(m \bmod 2) \leq i \leq n - 1$ , one  $XOR$  gate is to be saved.

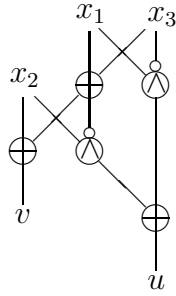


Fig. 3a

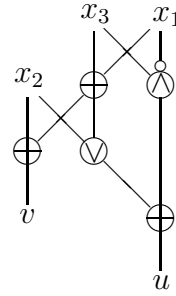


Fig. 3b

So, the following recurrent formulae hold:

$$\begin{aligned} K(2n - 1) &\leq K(n + 1) + K(n) + K(n - 1) + 38n - 16, & n \geq 6, \\ K(2n) &\leq K(n + 1) + 2K(n) + 38n - 2, & n \geq 5. \end{aligned} \quad (2)$$

A halving iteration of the Karatsuba method provides an advantage when  $m = 16$  or  $m \geq 18$ . Bounds on the complexity  $L(m)$  of multiplication of  $m$ -bit numbers for  $m \leq 18$  are collected in the Table 1 (symbol \* indicates values obtained via Karatsuba method).

For the convenience of comparison, let us derive the complexity of the Karatsuba multiplication circuit in an explicit form for  $m = 2^k$ . Denote

$$X_k = \begin{bmatrix} K(2^k + 2) \\ K(2^k + 1) \\ K(2^k) \end{bmatrix}, \quad A = \begin{bmatrix} 1 & 2 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 2 \end{bmatrix}, \quad b_k = \begin{bmatrix} 38 \cdot 2^k + 36 \\ 38 \cdot 2^k + 22 \\ 38 \cdot 2^k - 2 \end{bmatrix}.$$

Table 1: Bounds for the complexity of multiplication of  $m$ -bit numbers

$m$	1	2	3	4	5	6	7	8	9
$L(m)$	1	8	30	61	105	158	224	299	387
$m$	10	11	12	13	14	15	16	17	18
$L(m)$	484	594	713	845	986	1140	1287*	1479	1598*

Recurrences (2) imply  $X_{k+1} \leq AX_k + b_k$  for  $k \geq 4$ . Via common calculations, we obtain (1) as the solution of the latter inequality (initial values of the complexity should be taken from the Table 1).

Research supported in part by RFBR, grant 14-01-00671a.

## References

- [1] A.A. Karatsuba, Yu.P. Ofman. Multiplication of multidigit numbers on automata. *DAN USSR* **145**(2) (1962), 293–294 (in Russian). Eng. transl. in *Soviet Phys. Dokl.* **7** (1963), 595–596.
- [2] S.B. Gashkov. Entertaining computer arithmetic. Fast algorithms for operations with numbers and polynomials. Moscow, Librocom, 2012 (in Russian).
- [3] E. Demenkov, A. Kojevnikov, A. Kulikov, G. Yaroslavl'tsev. New upper bounds on the Boolean circuit complexity of symmetric functions. *Inf. Proc. Letters* **110**(7) (2010), 264–267.