# New Upper Bounds on the Boolean Circuit Complexity of Symmetric Functions

E. Demenkov[a,1], A. Kojevnikov[b,1], A. Kulikov[b,1,*], G. Yaroslavtsev[c,1]

[a] *St. Petersburg State University*
[b] *Steklov Institute of Mathematics at St. Petersburg*
[c] *Academic University*

**Abstract**

In this note, we present improved upper bounds on the circuit complexity of symmetric Boolean functions. In particular, we describe circuits of size $4.5n + o(n)$ for any symmetric function of $n$ variables, as well as circuits of size $3n$ for $\mathrm{MOD}_3^n$ function.

*Keywords:* Computational complexity, Boolean circuit complexity; Upper bounds; Symmetric functions; Modular functions

## 1. Introduction

By $B_n$ we denote the set of all Boolean functions $f\colon \{0,1\}^n \to \{0,1\}$. A function $f \in B_n$ is called symmetric if its value depends only on the sum of the input bits. That is, there must exist the vector $v \in \{0,1\}^{n+1}$ such that $f(x_1,\ldots,x_n) = v_s$ where $s = \sum_{i=1}^n x_i$. The set of all symmetric functions from $B_n$ is denoted by $S_n$. A typical symmetric function is a modular function $\mathrm{MOD}_{m,k}^n$ defined as follows:

$$\mathrm{MOD}_{m,r}^n(x_1,\ldots,x_n) = 1 \text{ iff } \sum_{i=1}^n x_i \equiv r \pmod m .$$

A circuit over the basis $\Omega \subseteq B_2$ is a directed acyclic graph with nodes of in-degree 0 or 2. Nodes of in-degree 0 are marked by variables from $\{x_1,\ldots,x_n\}$ and are called inputs. Nodes of in-degree 2 are marked by functions from $\Omega$ and are called gates. There are also special output gates. The size of a circuit is its

*Corresponding author
*Email addresses:* jack239@gmail.com (E. Demenkov), arist@logic.pdmi.ras.ru (A. Kojevnikov), kulikov@logic.pdmi.ras.ru (A. Kulikov), grigory@logic.pdmi.ras.ru (G. Yaroslavtsev)

[1] Research partially supported by Federal Target Program "Scientific and scientific-pedagogical personnel of the innovative Russia", RFBR (grants 08-01-00640 and 09-01-12137), RAS Program for Fundamental Research, Grant of the President of Russian Federation (MK-3912.2009.1), and Yandex.

number of gates. For a function $f \in B_n$, by $C_\Omega(f)$ we denote the minimal size of a circuit over $\Omega$ computing $f$. The two commonly studied bases are $B_2$ and $U_2 = B_2 \setminus \{\oplus, \equiv\}$. In this note, we consider circuits over $B_2$ and denote $C_{B_2}(f)$ by just $C(f)$.

While it is known that the circuit complexity of almost all Boolean functions from $B_n$ is $\Omega(2^n/n)$ [1], the best lower bound for an explicit function is $3n-o(n)$ [2]. The situation is better for symmetric Boolean functions: any symmetric function can be computed by circuits of size $5n + o(n)$ (see Sect. 2) and this is still the best known upper bound, the best lower bound is $2.5n - c$ [3]. For the basis $U_2$, the best known lower bound is $5n - o(n)$ by Iwama and Morizumi [4], the best lower bound for symmetric functions is $4n - c$ due to Zwick [5].

In this note, we present improved circuits of size $4.5n+o(n)$ for all symmetric functions. We also show that $C(\mathrm{MOD}_3^n) \leq 3n + c$ and present the sizes of optimal circuits for $\mathrm{MOD}_3^n$ for small values of $n$ that were found with the help of SAT-solvers.

On all the figures showing circuits we use only gates $\{\wedge, \vee, \oplus\}$ as there is no standard notation for other functions from $B_2$. To draw a gate computing $x \wedge \bar{y}$, we draw an $\wedge$-gate with a negation sign on the corresponding incoming edge.

## 2. Circuits of Size $4.5n + o(n)$ for All Symmetric Functions

The fact that any $f \in S_n$ depends on the sum of input bits only allows to compute $f$ in the following way:

1. Compute the binary representation ($\lceil \log n \rceil$ bits) of the sum of input bits.
2. Compute the function itself using the binary representation.

Since any $g \in B_n$ can be computed by a circuit of size $O(2^n/n)$ [6] the second stage requires $O(2^{\log n}/\log n) = o(n)$ gates. Below we first describe the standard circuit of size $5n$ for computing the binary representation of the sum of $n$ input bits and then present an improved circuit of size $4.5n$. For simplicity, throughout this section we assume that $n$ is a power of 2, if $n$ is not, the presented bounds are increased by $O(\log n)$ only.

The main building block of the well-known circuit for computing the binary representation is the binary Full Adder (FA) shown in Fig. 1(a,b). More precisely, $\mathrm{FA}(x, y, z) = (carry, sum)$, s.t. $x + y + z = 2 \cdot carry + sum$. Note that $sum = x \oplus y \oplus z$, $carry = maj(x, y, z) = xy \oplus yz \oplus zx$. Thus, two output bits of FA are the binary representation of the sum of three input bits. It is easy to verify that FA cannot be implemented by less than five gates. The sum of $n$ bits can be computed by a circuit containing $n$ FA's and $\log n$ layers as shown in Fig. 2. Its size is $5n$ and this is the best known upper bound.

The presented circuit can be easily adapted to compute $\mathrm{MOD}_{2^k}^n$. For example, to compute $\mathrm{MOD}_4^n$ we need to compute $y_1$ and $y_2$ only. This can be done by using the $n/2$ FA's from the top of the circuit above to compute $y_1$ and then taking the XOR of their carry bits to compute $y_2$. This gives a circuit of size $5 \cdot n/2 + n/2 = 3n$ for $\mathrm{MOD}_4^n$. In [3], Stockmeyer proved that $C(\mathrm{MOD}_4^n) = 2.5n + c$. He noticed that FA is an optimal block computing the

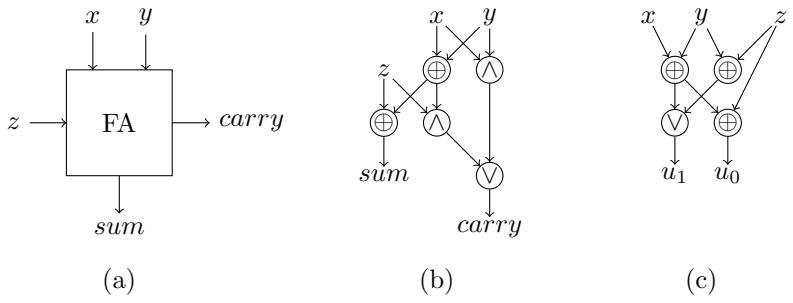


Figure 1: Full Adder (a,b) and Stockmeyer's block (c).

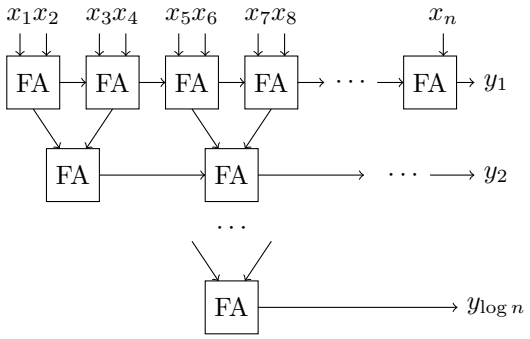


Figure 2: The well-known way of computing the binary representation of the sum of $n$ bits using $5n$ gates.

binary representation of three bits, but it is not an optimal block that encodes the sum of three bits by two bits. Stockmeyer used the block that for three input bits $x, y, z$ outputs two bits $u_1, u_0$ such that $x+y+z = 2\cdot(u_1 \oplus u_0)+u_0$. Thus, in this case $u_0 = sum$, while $u_1 = sum \oplus carry$. This block can be implemented using 4 gates, see Fig. 1(c).

Using this idea we construct a circuit of size $4.5n$ for computing the binary representation of the sum of $n$ input bits. Fig. 3 shows the main building block of the construction that we call MDFA (for Modified Double Full Adder). This block does the same as two FA's, but for a slightly different encoding of input and output bits. This allows to use 8 gates instead of 10 for implementing MDFA. It can be easily shown that for any $x_1, y_1, x_2, y_2, z \in \{0,1\}$,

$$\mathrm{MDFA}(x_1, x_1 \oplus y_1, x_2, x_2 \oplus y_2, z) = (carry_1, carry_1 \oplus carry_2, sum)\,,$$

where $sum = x_1 \oplus y_1 \oplus x_2 \oplus y_2 \oplus z$, $carry_1 = maj(x_1, y_1, z) = x_1y_1 \oplus y_1z \oplus zx_1$, $carry_2 = maj(x_2, y_2, x_1 \oplus y_1 \oplus z) = x_2y_2 \oplus x_2(x_1 \oplus y_1 \oplus z) \oplus y_2(x_1 \oplus y_1 \oplus z)$.

To compute the binary representation of $n$ input bits one first computes $n/2$ XOR's of variables and then uses $n/2$ MDFA's, see Fig. 4. The size of the

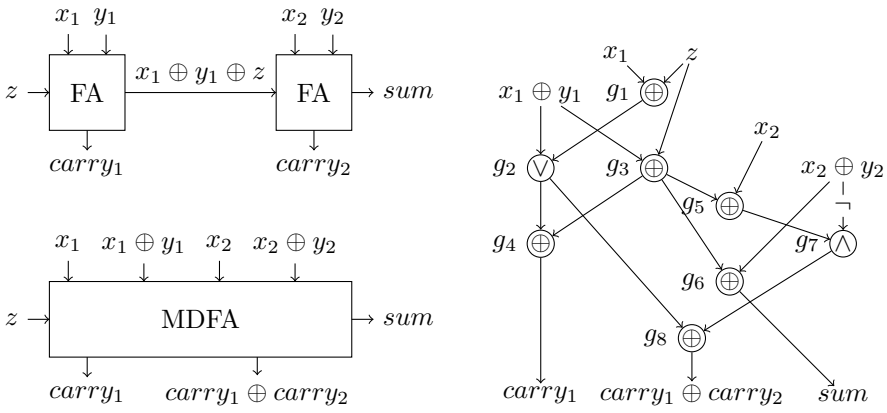


Figure 3: MDFA as a block and its implementation using 8 gates.

resulting construction is $n/2 + 8n/2 = 4.5n$. A logarithmic number of gates can be saved by noticing that the leftmost MDFA at each layer has only four inputs.

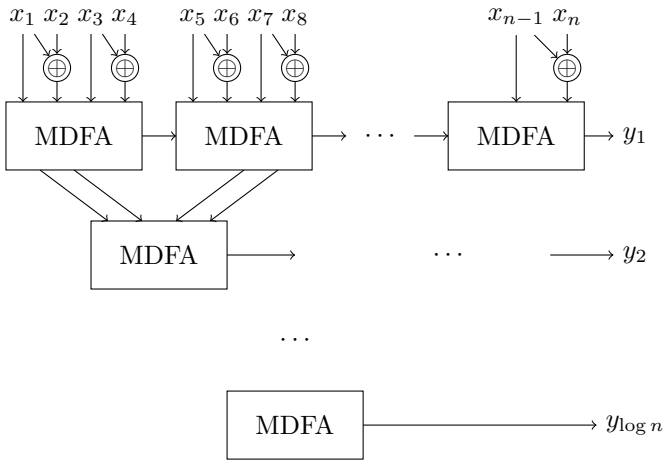


Figure 4: A circuit of size $4.5n$ for computing the binary representation of $n$ bits.

## 3. Improved Upper Bounds for MOD-functions

The exact circuit complexity (over the basis $B_2$) is known for $\mathrm{MOD}_2^n$ and $\mathrm{MOD}_4^n$ only. For all the remaining MOD-functions we only know the lower bound $2.5n - c$ [3] and the upper bound $4.5n + o(n)$. The circuit presented in the previous section implies a $(4.5 - 2^{3-k})n + O(k)$ upper bound for $C(\mathrm{MOD}_{2^k}^n)$

(for $k \geq 2$). Indeed, to compute $\mathrm{MOD}_{2^k}^n$ we need to compute $y_1, y_2, \ldots, y_k$. The first $k-1$ $y$'s are computed by $k-1$ layers of MDFA's. However, at the $(k-1)$-th layer we do not need the gates computing $carry_1$ (i.e., $g_4$ in Fig. 3). At the $k$-th layer we just compute the XOR of $n/2^k$ $carry_1 \oplus carry_2$ bits from the $(k-1)$-th layer. The overall size is

$$\frac{n}{2} + \sum_{i=1}^{k-2} 8 \cdot \frac{n}{2^{i+1}} + 7 \cdot \frac{n}{2^k} + \left(\frac{n}{2^k} - 1\right) = (4.5 - 2^{3-k})n - 1\,.$$

A circuit computing $\mathrm{MOD}_3^n$ can be built of inductive blocks shown in Fig. 5. This block takes as input the value of a remainder $r$ modulo 3 encoded by a pair of bits $(u_0, u_1)$ and outputs a pair of bits $(u_0', u_1')$ encoding $(r+x+y+z)$ mod 3. Since the block is implemented by 9 gates the upper bound $3n + c$ for $\mathrm{MOD}_3^n$ follows. The residue number encoding is the following:

$$r = \begin{cases} 0, & \text{if } (u_0, u_1) = (0,0), \\ 1, & \text{if } (u_0, u_1) = (0,1), \\ 2, & \text{if } u_0 = 1. \end{cases}$$

To help the reader to verify the construction we note that

$$g_3 = ((x \oplus u_1)(u_0 \oplus 1)) \oplus y \oplus 1, g_6 = (((x \oplus u_1)(u_0 \oplus 1)) \oplus 1)(x \oplus y \oplus u_0 \oplus 1).$$

It can be easily checked that

$$(r + x + y) \bmod 3 = \begin{cases} 1, & \text{if } (g_3, g_6) = (0,0), \\ 2, & \text{if } (g_3, g_6) = (1,0), \\ 0, & \text{if } g_6 = 1. \end{cases}$$

To see this, note that if $u_0 = 0$ (in this case $r = u_1$), then

$$g_3 = x \oplus y \oplus u_1 \oplus 1, g_6 = (x \oplus u_1 \oplus 1)(x \oplus y \oplus 1),$$

while if $u_0 = 1$ (in this case $r = 2$), then

$$g_3 = y \oplus 1, g_6 = x \oplus y.$$

Finally, it is not difficult to verify that

$$u_0' = (g_3 \oplus z)(1 \oplus g_6), u_1' = g_6 \oplus z \oplus 1$$

and that $(u_0', u_1')$ indeed encode $(r + x + y + z)$ mod 3.

This block was found using SAT-solvers after a long sequence of experiments. The fact that a circuit computing a MOD-function can be built of blocks of small constant size makes it possible to find such blocks automatically. We encoded the fact of existence of a block of size 9 computing the function described above as a CNF formula and used SAT-solvers to solve such formula, see [7] for details. The residue number encoding is essential here, as we were not able to find such a

block of size 9 for the standard encoding (binary representation). On the other hand, we also failed to prove that this block is optimal (for this, it is needed to prove that the corresponding formula is unsatisfiable), so an $8n/3$ upper bound for $\mathrm{MOD}_3^n$ is not excluded. The exact values of $C(\mathrm{MOD}_{3,k}^n)$ for $n \leq 5$ can be found in [7]. Modern SAT-solvers were not able to improve upper bounds for other $\mathrm{MOD}_k$ functions: for $k \geq 5$, the corresponding CNF formulas are too large.

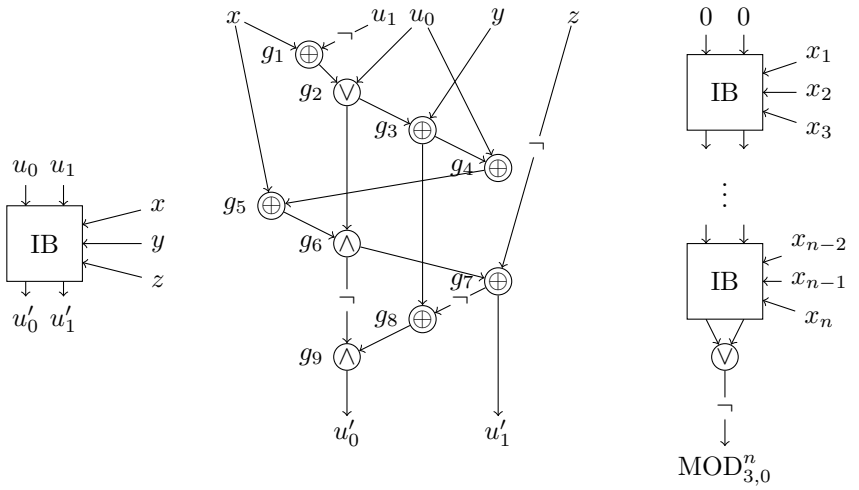


Figure 5: An inductive block for $\mathrm{MOD}_3$, its implementation using 9 gates, and a circuit for $\mathrm{MOD}_3^n$ built of $n/3$ such blocks.

## 4. Further Directions

A natural further direction is to improve lower and upper bounds on the circuit complexity of symmetric Boolean functions. An (apparently) easier problem is to close one of the following gaps (see [3], [7], [5]):

$$2.5n - c \leq C_{B_2}(\mathrm{MOD}_3^n) \leq 3n + c,\ 4n - c \leq C_{U_2}(\mathrm{MOD}_4^n) \leq 5n + c\,.$$


## Acknowledgments

We would like to thank Edward A. Hirsch and the anonymous referees for valuable comments.


## References


[1] C. E. Shannon, The synthesis of two-terminal switching circuits, Bell System Technical Journal 28 (1949) 59–98.

[2] N. Blum, A Boolean function requiring $3n$ network size, Theoretical Computer Science 28 (1984) 337–345.

[3] L. J. Stockmeyer, On the combinational complexity of certain symmetric Boolean functions, Mathematical Systems Theory 10 (1977) 323–336.

[4] K. Iwama, H. Morizumi, An explicit lower bound of $5n - o(n)$ for Boolean circuits, in: Proceedings of 27th MFCS, Lecture Notes in Computer Science, Springer, 2002, pp. 353–364.

[5] U. Zwick, A $4n$ lower bound on the combinational complexity of certain symmetric boolean functions over the basis of unate dyadic Boolean functions, SIAM Journal on Computing 20 (1991) 499–505.

[6] O. Lupanov, A method of circuit synthesis, Izvestiya VUZov, Radiofizika 1 (1959) 120–140, in Russian.

[7] A. Kojevnikov, A. S. Kulikov, G. Yaroslavtsev, Finding efficient circuits using SAT-solvers, in: Proceedings of 12th SAT, Vol. 5584 of Lecture Notes in Computer Science, Springer, 2009, pp. 139–157.