






# Technology-Dependent Synthesis and Optimization of Circuits for Small S-boxes

Zihao Wei<sup>1,2</sup> , Siwei Sun<sup>1,3</sup> , Fengmei Liu<sup>2</sup>, Lei Hu<sup>4,5</sup> and Zhiyu Zhang<sup>1</sup> 

<sup>1</sup> School of Cryptology, University of Chinese Academy of Sciences, Beijing, China

<sup>2</sup> Data Communication Science and Technology Research Institute, Beijing, China

<sup>3</sup> State Key Laboratory of Cryptology, Beijing, China

<sup>4</sup> Key Laboratory of Cyberspace Security Defense, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China

<sup>5</sup> School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

**Abstract.** Boolean formula minimization is a notoriously hard problem that is known to be  $\Sigma_2^P$ -complete. Circuit minimization, typically studied in the context of a much broader subject known as synthesis and optimization of circuits, introduces another layer of complexity since ultimately those technology-independent representations (e.g., Boolean formulas and truth tables) has to be transformed into a netlist of cells of the target technology library. To manage those complexities, the industrial community typically separates the synthesis process into two steps: technology-independent optimization and technology mapping. In each step, this approach only tries to find the local optimal solution and relies heavily on heuristics rather than a systematic search. However, for small S-boxes, a more systematic exploration of the design space is possible. Aiming at the global optimum, we propose a method which can synthesize a truth table for a small S-box directly into a netlist of the cells of a given technology library. Compared with existing technology-dependent synthesis tools like LIGHTER and PEIGEN, our method produces improved results for many S-boxes with respect to circuit area. In particular, by applying our method to the  $\mathbb{F}_{2^4}$ -inverter involved in the tower field implementation of the AES S-box, we obtain the currently known lightest implementation of the AES S-box. The search framework can be tweaked to take circuit delay into account. As a result, we find implementations for certain S-boxes with both latency and area improved.

**Keywords:** Circuit minimization · Latency · Logic synthesis · Technology mapping · S-box · AES

## 1 Introduction

Chips designed for a particular, limited product or application (ASIC) can be found everywhere in our digital world. Arguably, the importance of ASICs cannot be overemphasized in almost all aspects (from daily lives to military technologies) of our information society. Logic synthesis, which transforms a relatively high-level functional or behavioral description of a digital circuit into an optimized gate-level representation, plays a vital role in the production of cost-effective and high-performance ASICs. Therefore, the development of methodologies and tools for logic synthesis is one of the fundamental topics in digital

---

This work is supported by National Key Research and Development Program of China (2022YFB2701900) and the National Natural Science Foundation of China (62032014).

E-mail: [wei\\_z\\_h@163.com](mailto:wei_z_h@163.com) (Zihao Wei), [siweisun.isaac@gmail.com](mailto:siweisun.isaac@gmail.com) (Siwei Sun), [lfmei@sina.com](mailto:lfmei@sina.com) (Fengmei Liu), [hulei@iie.ac.cn](mailto:hulei@iie.ac.cn) (Lei Hu), [zhangzhiyu14@mails.ucas.ac.cn](mailto:zhangzhiyu14@mails.ucas.ac.cn) (Zhiyu Zhang)



design, extensively studied by both the industrial and academic communities. In the context of lightweight symmetric-key cryptography, an effective and efficient tool for logic synthesis has direct and immediate impact on the quality of the selected building blocks and their hardware implementations.

*Guiding the Design Choices.* Let us consider the typical working flow for designing a lightweight block cipher. After the decision of the high-level structure being made, we need to instantiate the structure with concrete components like MDS matrices or S-boxes. At this step, typically we try to find a building block from a large set of candidates such that some metric (e.g., area) is optimized under certain constraints (e.g., security and latency). The quality of this selection process largely depends on the effectiveness of the employed synthesis methodology.

This is evidenced by the development of the open literature on the construction of lightweight MDS matrices, where the goal is to minimize the number of XOR gates required in the implementations. At the early stage, due to the lack of efficient tools for synthesizing linear functions, people relied on intuitive and ad-hoc local optimization heuristics. For instances, we may guide the search based on the statement that the matrix whose binary representation has a lower Hamming weight or whose entries enjoying lower hardware footprints can be implemented with less XOR gates [SKOP15, BKL16, LW16, LW17, SS16, JPST17a, ZWS18, GLWL16]. While such heuristics greatly simplify the searching process, they hardly reflect the real cost of a matrix and thus leave a large room for improvements. With more advanced synthesizing methods (e.g., SAT-based approach [Sto16, FS10], LIGHTER [JPST17b], PEIGEN [BGLS19], SLP heuristics [BMP13, LSL<sup>+</sup>19]) and global optimizations, more compact MDS matrices are identified recently [KLSW17, JPST17b, DL18, LSL<sup>+</sup>19]. In the case of finding lightweight S-boxes with strong security properties, we will face similar synthesis problems [BGG<sup>+</sup>16, BGG<sup>+</sup>17, GJN<sup>+</sup>16]. In short, a good synthesis tool can guide the designers to pick out high quality and cost effective cryptographic components.

*Optimizing the implementations.* For a given cipher, the ability of the synthesis tool to produce implementations fulfilling various area and timing constraints imposed by the target platforms and specific application scenarios partly determines the competitiveness of the final product and the range of applications of the cipher. One way to improve the overall implementation of a cipher is to optimize the implementations of its building blocks. For example, we have seen an endless endeavor in reducing the size (or latency) of the circuits for the MDS matrix [LSL<sup>+</sup>19, Max19, TP20] and the S-box [Can05, RTA18, ME19, MPL<sup>+</sup>11] involved in the Advanced Encryption Standard (AES).

As the complexity and diversity of cryptographic applications are increasing, the effect of circuit synthesis and optimizations becomes even more profound. New technology process requires synthesis tools to take technology-specific characteristics into account to fully exploit the technological advancement and new applications scenarios may bring new optimization objectives. For example, in fully homomorphic encryptions people are interested in reducing the so-called multiplicative complexity (the number of AND gates used in the circuit) [ARS<sup>+</sup>15a]. Also, it is known that any method for reducing the number of multiplicative complexity for classical circuits can be employed to reduce the so-called  $T$ -depth of quantum circuits constructed using the Clifford+ $T$  group of gates. In short, circuit synthesis and optimization is an important field of study for both the academic and industrial communities, and any progress in this field has a potential to impact many applications.

Although eventually a circuit has to be constructed with a specific technology library, most synthesis tools do not translate a given design into the cells of a target library directly due to complexity issues. Instead, the process of circuit synthesis is typically done in a two-step approach, where the first step is technology independent and works on abstract

mathematical representations involving a highly restricted gate set (e.g., AND, OR, and NOT) decoupled from any specific technology. Techniques such as Quine-McCluskey method for two-level logic optimization [McC56] or methods for multilevel logic optimization [BHS90] are often employed in this step by many CAD tools. The resulting design of the technology-independent optimization can be ultimately implemented in any specific libraries of cells by means of the so-called technology mapping. This approach is quite efficient for even moderately large building blocks and is extensively employed in practice. However, splitting the process into these somewhat independent phases leads to quality loss, e.g., the output of the first phase may fit badly onto the target architecture. For small circuits, it is possible to perform technology-dependent optimizations directly, which may result in more fine-tuned optimizations.

**Related Work.** The two most relevant technology-dependent synthesis tools for small cryptographic building blocks of this work are LIGHTER [JPST17b] and PEIGEN [BGLS19]. LIGHTER is an open-source tool developed by Jean, Peyrin, Sim, and Tourteaux [JPST17b], which can search for efficient technology-dependent implementations of small vectorial Boolean functions. For small non-linear S-boxes, LIGHTER applies a meet-in-the-middle approach with breath-first search starting from the two root nodes encoding the identity function and the target function respectively. In the search for an implementation of an  $n \times n$  S-box, new nodes are generated from old ones by applying *invertible*  $\mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$  instructions which are compositions of atomic logic gates from the underlying technology library. This set of invertible instructions is named as  $\mathcal{B}$ -set and LIGHTER is expected to find the optimal  $\mathcal{B}$ -implementation (an implementation using instructions restricted to the  $\mathcal{B}$ -set) of the S-box. PEIGEN can be regarded as an extended and enhanced version of LIGHTER.<sup>1</sup> Apart from some programming-level improvements for boosting its time and memory efficiency, PEIGEN [BGLS19] expands the  $\mathcal{B}$ -set with full support for ANDN and ORN and optimizes away the meet-in-the-middle strategy by matching in one search tree rooted at the node encoding the identity function. We note that neither LIGHTER or PEIGEN can synthesize irreversible S-boxes due to the property of the  $\mathcal{B}$ -set.

**Our Contribution.** We present a technology-dependent synthesis tool that can find optimized implementations of small S-boxes in terms of the core cells of a specific technology libraries. Compared with LIGHTER and PEIGEN, our tool enjoys several advantages. Firstly, our tool is able to handle any small S-boxes, including those irreversible ones. Secondly, the basic version of our algorithm can identify the circuit with the lowest area for a  $3 \times 3$  S-box, while the results given by LIGHTER or PEIGEN are not guaranteed to be optimal. Thirdly, a tweaked version of our algorithm can find optimized circuit in terms of area subject to timing (latency) constraints. By the way, some minor bugs in LIGHTER and PEIGEN are identified.

We apply the method to a series of small ( $3 \times 3$  and  $4 \times 4$ ) S-boxes with several common CMOS technology libraries, and improved results are obtained. In particular, we manage to produce the currently known lightest implementation of the AES S-box by applying our tool to a  $\mathbb{F}_2^4 \rightarrow \mathbb{F}_2^4$  subcomponent of the tower field implementation of the AES S-box. In certain cases, the circuits produced by our method are superior to existing results with respect to both area and latency. Also, we apply our method to some  $8 \times 8$  S-boxes constructed from  $4 \times 4$  S-boxes and improved results are obtained. Moreover, our tool can be applied in the context of synthesis and optimization of quantum circuits to reduce the so-called  $T$ -count or employed to reduce the multiplicative complexities which are concerned in the context of multi-party computation, zero-knowledge proofs, or fully homomorphic encryption schemes [ARS<sup>+</sup>15b].

<sup>1</sup>PEIGEN also has functionalities for evaluating many security-related properties of the underlying S-boxes. In this work, we only focus on its technology-dependent synthesis aspect.

Finally we would like to note that our algorithm is more memory extensive than LIGHTER and PEIGEN, and sometimes it may fail to output a circuit within practical memory consumption. Moreover, in the majority of cases, our algorithm consumes a longer time span than LIGHTER and PEIGEN. Therefore, we regard our method as a complement of LIGHTER and PEIGEN. The source code of this work is available via [https://github.com/zihawei/CiC2024\\_TechDepSboxCircuits](https://github.com/zihawei/CiC2024_TechDepSboxCircuits).

## 2 Notations and Preliminaries

An ordered set with  $n$  elements is denoted by  $\mathcal{A} = [z_0, \dots, z_{n-1}]$ . Note that in our notation, we do not allow duplicated elements, and thus all elements in the ordered set are distinct. When the order is not important, we use  $\mathcal{A}.\text{SET}()$  to represent the set  $\{z_0, \dots, z_{n-1}\}$ . For an ordered or unordered set  $\mathcal{B}$ ,  $\mathcal{C} \subseteq \mathcal{B}$  signifies that  $\mathcal{C}$  is an ordered subset of  $\mathcal{B}$ . For example, let  $\mathcal{B} = \{0, 1, 2\}$ , then all its 2-element ordered subsets are  $[0, 1] \subseteq \mathcal{B}$ ,  $[0, 2] \subseteq \mathcal{B}$ ,  $[1, 2] \subseteq \mathcal{B}$ ,  $[1, 0] \subseteq \mathcal{B}$ ,  $[2, 0] \subseteq \mathcal{B}$ , and  $[2, 1] \subseteq \mathcal{B}$ . In addition, the number of elements in a set  $\mathcal{B}$  is denoted by  $|\mathcal{B}|$ .

**Logic Gates and Combinatorial Circuits** Logic gates are the basic building blocks of combinational logic circuits.<sup>2</sup> A logic gate corresponds to a Boolean function with one or more input signals (typically less than or equal to four bits). A list of frequently used logic gates found in common CMOS technology libraries are listed in Table 1. Hereafter, the number of input bits of a gate  $\beta$  is denoted by  $\#\beta$ . For example, according to Table 1, we have  $\#\text{NOT} = 1$ ,  $\#\text{NAND} = 2$ , and  $\#\text{OAI21} = 3$ . Also, for a gate  $\beta$  with  $\#\beta = l$ ,  $\beta(x_0, \dots, x_{l-1})$  represents the Boolean function associated with  $\beta$  in variables  $(x_0, \dots, x_{l-1})$ . For instance,  $\text{MUX}(x_0, x_1, x_2) = x_0x_1 + x_0x_2 + x_2$ . Note that the order of the variables (also called signals in this work) matters, and  $\text{MUX}(x_0, x_2, x_1) = x_0x_2 + x_0x_1 + x_1$  is different from  $\text{MUX}(x_0, x_1, x_2)$ .

Loosely speaking, logic synthesis is the process of composing a logic circuit that implements a given vectorial Boolean function using a predefined finite set of basic logic gates. It is not hard to imagine that the design space of a moderately complex vectorial Boolean function can be intractably large, and how to identify the optimal implementation with respect to certain objective is a fundamental problem in logic design. To facilitate our discussion, we introduce some formal notations.

Let  $\mathcal{G}$  be a set of logic gates, a combinatorial circuit constructed with the gates in  $\mathcal{G}$  is called a  $\mathcal{G}$ -circuit. A  $\mathcal{G}$ -circuit with an  $n$ -bit input  $(x_0, x_1, \dots, x_{n-1})$  can be specified by a sequence of signals

$$\mathcal{S} = [x_0, x_1, \dots, x_{n-1}; x_n, x_{n+1}, \dots, x_{n+t-1}]$$

with  $t \geq 0$  and a sequence of gate-signal pairs

$$\mathcal{I} = [(\beta_0, \mathcal{X}_0), \dots, (\beta_{t-1}, \mathcal{X}_{t-1})]$$

with  $\beta_j \in \mathcal{G}$ ,  $\mathcal{X}_j \subseteq \{x_0, x_1, \dots, x_{n-1+j}\}$ , and  $\#\beta_j = \#\mathcal{X}_j$ , such that for  $1 \leq i < t$ ,  $x_{n+i-1} = \beta_{i-1}(\mathcal{X}_{i-1})$ . Under these notations, the symbolic object  $x_j$  with  $n \leq j < n+t$  can be regarded as a Boolean function in the input variables  $\{x_0, x_1, \dots, x_{j-1}\}$ , which in turn can be regarded as a Boolean function in variables  $\{x_0, \dots, x_{n-1}\}$ . We call  $x_0, \dots, x_{n-1}$  the input signals and  $x_n, \dots, x_{n+t-1}$  the derived signals. Note that the input signals and derived signals are separated with “;” in  $\mathcal{S}$ . We emphasize again that in our notation, when the signals in  $\mathcal{S}$  are regarded as Boolean functions in variables  $\{x_0, \dots, x_{n-1}\}$ , no duplicated elements are allowed. This requirement will not rule out the optimal solutions

<sup>2</sup>In this work we restrict our attention to combinatorial logics where the output of the logic circuit is a pure function of the *present* inputs only.

Table 1: Frequently-used logic gates in common CMOS technology libraries. The same list of gates whose functionalities represented with logic formulas is given in Table 8 in Appendix A.

Gate	Domain $\rightarrow$ Range	Functionality (algebraic normal form)
NOT	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0 \mapsto x_0 + 1$
AND	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto x_0x_1$
NAND	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto x_0x_1 + 1$
NANDN	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto x_0x_1 + x_1 + 1$
OR	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto x_0x_1 + x_0 + x_1$
NOR	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto x_0x_1 + x_0 + x_1 + 1$
NORN	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto x_0x_1 + x_0$
XOR	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto x_0 + x_1$
XNOR	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto x_0 + x_1 + 1$
AND3	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0x_1x_2$
NAND3	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0x_1x_2 + 1$
NANDN3	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0x_1x_2 + x_1x_2 + 1$
OR3	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0x_1x_2 + x_0x_1 + x_0x_2 + x_1x_2 + x_0 + x_1 + x_2$
NOR3	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0x_1x_2 + x_0x_1 + x_0x_2 + x_1x_2 + x_0 + x_1 + x_2 + 1$
NORN3	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0x_1x_2 + x_0x_1 + x_0x_2 + x_0$
XOR3	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0 + x_1 + x_2$
XNOR3	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0 + x_1 + x_2 + 1$
MUX	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0x_1 + x_0x_2 + x_2$
MUXI	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0x_1 + x_0x_2 + x_2 + 1$
AO21	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0x_1x_2 + x_0x_1 + x_2$
AOI21	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0x_1x_2 + x_0x_1 + x_2 + 1$
OAI21	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0x_1x_2 + x_0x_2 + x_1x_2$
OAI21	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0x_1x_2 + x_0x_2 + x_1x_2 + 1$
ANDN	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto x_0x_1 + x_1$
ORN	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto x_0x_1 + x_0 + 1$
MAOI1	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2, x_3 \mapsto x_0x_1x_2x_3 + x_0x_1x_2 + x_0x_1x_3 + x_2x_3 + x_2 + x_3$
MOAI1	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2, x_3 \mapsto x_0x_1x_2x_3 + x_0x_2x_3 + x_1x_2x_3 + x_0x_1 + x_0 + x_1 + 1$

since for each circuit not fulfilling our requirement, we can find a circuit satisfying our rules whose cost is less or equal to it. Also, we call  $\mathcal{S}$  and  $\mathcal{I}$  the *signal sequence* and *gate sequence*, respectively. The circuit corresponding to a given signal sequence  $\mathcal{S}$  and a gate sequence  $\mathcal{I}$  is denoted by  $\text{CIRCUIT}(\mathcal{S}, \mathcal{I})$ . This tedious description is best illustrated by an example.

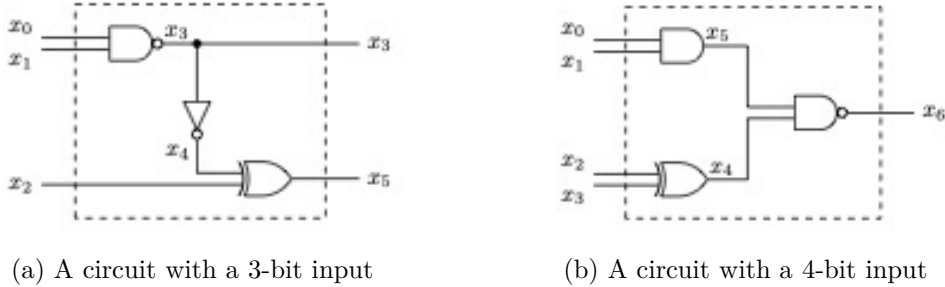


Figure 1: Two simple circuits illustrating the signal sequence and gate sequence notations

**Example 1.** Let  $\mathcal{G} = \{\text{NOT}, \text{AND}, \text{NAND}, \text{XOR}\}$ ,  $\mathcal{S} = [x_0, x_1, x_2; x_3, x_4, x_5]$ , and

$$\mathcal{I} = [(\text{NAND}, [x_0, x_1]), (\text{NOT}, [x_3]), (\text{XOR}, [x_4, x_2])].$$

Then the  $\mathcal{G}$ -circuit  $\text{CIRCUIT}(\mathcal{S}, \mathcal{I})$  is depicted in Figure 1(a). Considering the circuit given in Figure 1(b), it can be specified by  $\text{CIRCUIT}(\mathcal{S}, \mathcal{I})$  with

$$\begin{cases} \mathcal{S} = [x_0, x_1, x_2; x_3, x_4, x_5, x_6] \\ \mathcal{I} = [(\text{XOR}, [x_2, x_3]), (\text{AND}, [x_0, x_1]), (\text{NAND}, [x_5, x_4])] \end{cases}.$$

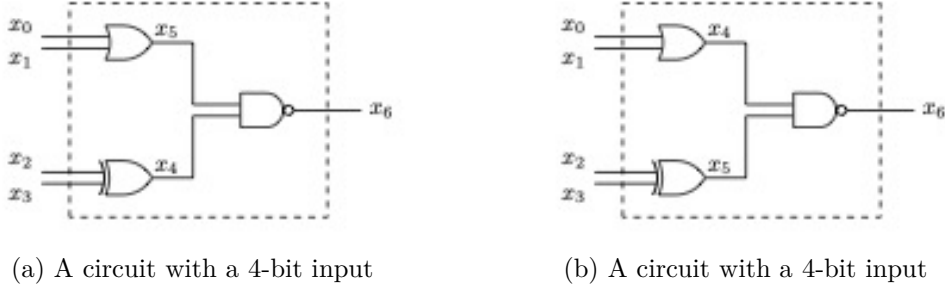


Figure 2: One circuit can have multiple different signal-gate sequence representations

We warn the reader that for a given a circuit, its signal-gate representation is not unique. Taking the circuit depicted in Figure 2 for example, it can be specified as  $\text{CIRCUIT}(\mathcal{S}, \mathcal{I})$  or  $\text{CIRCUIT}(\mathcal{S}', \mathcal{I}')$  with

$$\begin{cases} \mathcal{S} = [x_0, x_1, x_2, x_3; x_4, x_5, x_6] \\ \mathcal{I} = [(\text{XOR}, [x_2, x_3]), (\text{OR}, [x_0, x_1]), (\text{NAND}, [x_5, x_4])] \end{cases}$$

or

$$\begin{cases} \mathcal{S}' = [x_0, x_1, x_2, x_3; x_4, x_5, x_6] \\ \mathcal{I}' = [(\text{OR}, [x_0, x_1]), (\text{XOR}, [x_2, x_3]), (\text{NAND}, [x_4, x_5])] \end{cases}.$$

Therefore, we need to carefully rule out such repetitions in our search algorithms.

**Area and Latency.** Given a gate set  $\mathcal{G}$ , for each gate  $\beta \in \mathcal{G}$ , a positive integer called *area* and denoted by  $\|\beta\|$  is associated with it. We should not take the notion so literally since what it represents is some relative cost that may well deviate from the absolute area of the gate. For example, let  $\mathcal{G} = \{\text{NOT}, \text{AND}, \text{NAND}, \text{OR}, \text{NOR}\}$  whose absolute areas are  $0.532 \mu\text{m}^2$ ,  $1.064 \mu\text{m}^2$ ,  $0.798 \mu\text{m}^2$ ,  $1.064 \mu\text{m}^2$  and  $0.798 \mu\text{m}^2$ , respectively. In terms of gate equivalence (GE), these corresponds to 0.67 GE, 1.33 GE, 1.00 GE, 1.33 GE, and 1.00 GE, respectively. In our work, only the relative cost is essential and therefore we may set the areas of the gates to  $\|\text{NOT}\| = 0.67 \times 100 = 67$ ,  $\|\text{AND}\| = 1.33 \times 100 = 133$ ,  $\|\text{NAND}\| = 1.00 \times 100 = 100$ ,  $\|\text{OR}\| = 1.33 \times 100 = 133$ , and  $\|\text{NOR}\| = 1.00 \times 100 = 100$ .

Similarly, we associate each gate a parameter called delay to capture its latency characteristic, and the delay of a gate  $\beta$  is denoted by  $\mathfrak{D}(\beta)$ . In common technology libraries, the delay or latency characteristic of a gate is typically specified over ranges of temperature and operating voltages, and thus this parameter is more complicated than area. How to approximate the delay of a gate in our algorithm is described in Section 4. Moreover, defining the overall delay of a signal and a circuit represents a formidable challenge, considering that a multitude of factors can exert an influence. To streamline the model, we define the delay of a signal as the cumulative sum of the delays of the gates involved in its generation. Meanwhile, the delay of a circuit is defined as the maximum value among the delays of all signals within the circuit.

Let  $\mathcal{C} = \text{CIRCUIT}(\mathcal{S}, \mathcal{I})$  be a  $\mathcal{G}$ -circuit with

$$\begin{cases} \mathcal{S} = [x_0, x_1, \dots, x_{n-1}; x_n, \dots, x_{n+t-1}] \\ \mathcal{I} = [(\beta_0, \mathcal{X}_0), \dots, (\beta_{t-1}, \mathcal{X}_{t-1})] \end{cases}$$

with  $\beta_j \in \mathcal{G}$ ,  $\mathcal{X}_j \sqsubseteq \{x_0, x_1, \dots, x_{n-1+j}\}$ , and  $\#\beta_j = \#\mathcal{X}_j$ , such that for  $1 \leq i < t$ ,  $x_{n+i-1} = \beta_{i-1}(\mathcal{X}_{i-1})$ . The area of the circuit is defined as  $\|\mathcal{C}\| = \sum_{i=0}^{t-1} \|\beta_i\|$ . Also, We

associate each signal  $x_j \in \mathcal{S}$  a parameter  $\mathfrak{X}(x_j)$  called delay with the following rule:

$$\begin{cases} \mathfrak{X}(x_0) = \cdots = \mathfrak{X}(x_{n-1}) = 0 \\ \mathfrak{X}(x_{n+i-1}) = \mathfrak{X}(\beta_{i-1}) + \max\{\mathfrak{X}(x_j) : x_j \in \mathcal{X}_{i-1}\} \end{cases} . \quad (1)$$

The delay of the circuit  $\mathfrak{C}$  is defined as  $\mathfrak{X}(\mathfrak{C}) = \max\{\mathfrak{X}(x_j) : x_j \in \mathcal{S}\}$ .

### 3 Exploring the Search Space with Single-gate Extensions

Let  $\mathcal{G}$  be a set of logic gates. A  $\mathcal{G}$ -circuit  $\mathfrak{C} = \text{CIRCUIT}(\mathcal{S}, \mathcal{I})$  with

$$\mathcal{S} = [x_0, \cdots, x_{n-1}; x_n, \cdots, x_{n+t-1}]$$

is said to implement a vectorial Boolean function  $F : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$  sending  $(x_0, \cdots, x_{n-1})$  to  $(f_0(x_0, \cdots, x_{n-1}), \cdots, f_{m-1}(x_0, \cdots, x_{n-1}))$  if and only if

$$\{f_0(x_0, \cdots, x_{n-1}), \cdots, f_{m-1}(x_0, \cdots, x_{n-1})\} \subseteq \mathcal{S},$$

where the elements of  $\mathcal{S}$  are regarded as Boolean functions in  $(x_0, \cdots, x_{n-1})$  as discussed in Section 2. Our basic goal is to find a  $\mathcal{G}$ -circuit implementing a given vectorial Boolean function  $F : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$  whose area is minimized. Our search algorithm starts with the circuit  $\text{CIRCUIT}(\mathcal{S}, \mathcal{I})$  with  $\mathcal{S} = [x_0, \cdots, x_{n-1}]$  and  $\mathcal{I} = []$  and proceeds with the so-called *single-gate extensions*.

**Definition 1.** Given a  $\mathcal{G}$ -circuit  $\mathfrak{C}$  with an  $n$ -bit input specified by its signal sequence  $\mathcal{S} = [x_0, x_1, \cdots, x_{n-1}; x_n, x_{n+1}, \cdots, x_{n+t-1}]$  and gate sequence  $\mathcal{I} = [(\beta_0, \mathcal{X}_0), \cdots, (\beta_{t-1}, \mathcal{X}_{t-1})]$ . We can extend it by applying a gate  $\beta_t \in \mathcal{G}$  with an ordered set  $\mathcal{X}_t \subseteq \{x_0, \cdots, x_{n+t-1}\}$  such that  $\#\beta_t = |\mathcal{X}_t|$  as its input to produce a new circuit  $\mathfrak{C}'$  with instruction sequence

$$[(\beta_0, \mathcal{X}_0), \cdots, (\beta_{t-1}, \mathcal{X}_{t-1}), (\beta_t, \mathcal{X}_t)]$$

and signal sequence  $[x_0, x_1, \cdots, x_{n-1}; x_n, x_{n+1}, \cdots, x_{n+t-1}, x_{n+t}]$  such that  $x_{n+t} \neq x_j$  (treated as Boolean functions in the input signals) for  $0 \leq j < n+t$ . The above process of deriving  $\mathfrak{C}'$  from  $\mathfrak{C}$  is called a *single-gate extension* of  $\mathfrak{C}$  with  $(\beta_t, \mathcal{X}_t)$ .

Hereafter, we will use the notation  $\mathfrak{C}.\mathcal{S}$  or  $\mathfrak{C}.\mathcal{I}$  to emphasize the ownership of the signal sequence  $\mathcal{S}$  and instruction sequence  $\mathcal{I}$ . For example, in Definition 1, we have

$$\begin{cases} \mathfrak{C}.\mathcal{S} = [x_0, x_1, \cdots, x_{n+t-1}] \\ \mathfrak{C}.\mathcal{I} = [(\beta_0, \mathcal{X}_0), \cdots, (\beta_{t-1}, \mathcal{X}_{t-1})] \end{cases} \quad \text{and} \quad \begin{cases} \mathfrak{C}'.\mathcal{S} = [x_0, x_1, \cdots, x_{n+t-1}, x_{n+t}] \\ \mathfrak{C}'.\mathcal{I} = [(\beta_0, \mathcal{X}_0), \cdots, (\beta_{t-1}, \mathcal{X}_{t-1}), (\beta_t, \mathcal{X}_t)] \end{cases} .$$

We now illustrate Definition 1 with a simple example.

**Example 2.** Consider the circuit  $\mathfrak{C}$  given in Figure 3(a) with

$$\begin{cases} \mathfrak{C}.\mathcal{S} = [x_0, x_1, x_2; x_3] \\ \mathfrak{C}.\mathcal{I} = [(\text{AND}, [x_1, x_0])] \end{cases} .$$

The circuit  $\mathfrak{C}'$  depicted in Figure 3(b) with

$$\begin{cases} \mathfrak{C}'.\mathcal{S} = [x_0, x_1, x_2; x_3, x_4] \\ \mathfrak{C}'.\mathcal{I} = [(\text{AND}, [x_1, x_0]), (\text{XOR}, [x_3, x_2])] \end{cases}$$

is a single-gate extension of  $\mathfrak{C}$ .

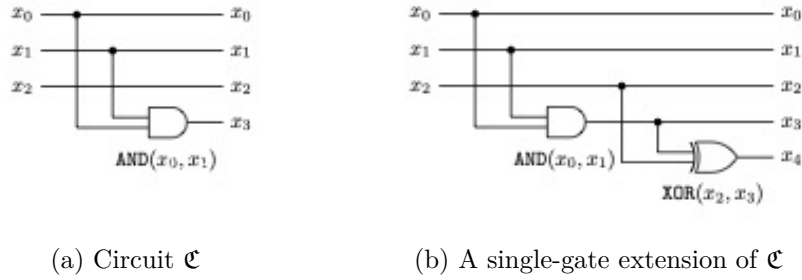


Figure 3: Two simple circuits illustrating the single-gate extension

Before setting up the algorithmic framework, we state a simple theorem.

**Theorem 1.** *Let  $\mathcal{G} = \{\beta_0, \dots, \beta_{s-1}\}$  be a gate set and  $\mathcal{C}$  be a  $\mathcal{G}$ -circuit. Then, the area  $\|\mathcal{C}\|$  is a multiple of  $\zeta(\mathcal{G}) = \gcd(\|\beta_0\|, \dots, \|\beta_{s-1}\|)$ . Consequently, for any  $\mathcal{G}$ -circuit,  $\|\mathcal{C}\| = k \cdot \zeta(\mathcal{G})$  for some non-negative integer  $k$ .*

*Proof.* It comes from  $\|\mathcal{C}\| = \sum_{i=0}^{s-1} \|\beta_i\|$  with the fact that  $\zeta(\mathcal{G})$  divides  $\|\beta_i\|$  for all  $i$ .  $\square$

### 3.1 Technology-Dependent Synthesis: The Algorithmic Framework

Let  $\mathbb{L}^{(k \cdot \zeta(\mathcal{G}))}$  be the set of all  $\mathcal{G}$ -circuits with area  $k \cdot \zeta(\mathcal{G})$  for some non-negative integer  $k$ , none of which can be implemented with area less than  $k \cdot \zeta(\mathcal{G})$ . For a vectorial Boolean function  $F : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$ , our basic algorithm systematically produces all  $\mathcal{G}$ -circuits in  $\mathbb{L}^{(k \cdot \zeta(\mathcal{G}))}$  with increasing  $k \in \{1, 2, \dots\}$  through single-gate extensions until a circuit implementing  $F$  is encountered. The framework of our algorithm for technology-dependent synthesis is described in Algorithm 1. Before showing the details of the algorithm, we first familiarize the reader with the “shape” of the search space illustrated in Figure 4.

We call the circuits visited during the exploration of the search space nodes. Conceptually, the search space of our algorithm (for a circuit with an  $n$ -bit input) forms a tree with a single root node representing the circuit with  $\mathcal{S} = [x_0, \dots, x_{n-1}]$  and  $\mathcal{I} = []$ . The nodes are organized into different layers and the circuits in the same layer are of the same area. Two nodes may be connected by a directed edge, indicating that the target node is a single-gate extension of the source node.

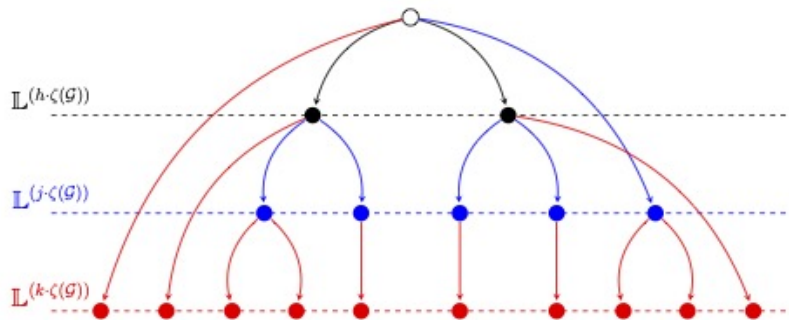


Figure 4: A visualization of the search space of Algorithm 1

Now, let us look at Algorithm 1. From *Line 1* to *Line 5*, we create the root node, and put it into the list  $\mathbb{L}^{(0)}$ . From *Line 9* to *Line 21*, the algorithm creates a list  $\mathbb{L}^{(k \cdot \zeta(\mathcal{G}))}$  of



**Algorithm 1:** The algorithmic framework

---

**Input:** A universal gate set  $\mathcal{G}$  and a vectorial Boolean function  $F : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$  mapping  $(x_0, \dots, x_{n-1})$  to  $(f_0(x_0, \dots, x_{n-1}), \dots, f_{n-1}(x_0, \dots, x_{n-1}))$

**Output:** A  $\mathcal{G}$ -circuit implementing  $F$  with minimal area

```

1 /* Initialization */
2  $\mathbb{L}^{(0)} \leftarrow []$ 
3  $\mathcal{S} \leftarrow [x_0, x_1, \dots, x_{n-1}]$ ,  $\mathcal{I} \leftarrow []$  /* The signal and instruction sequences */
4  $\mathcal{C} \leftarrow \text{CIRCUIT}(\mathcal{S}, \mathcal{I})$ 
5  $\mathbb{L}^{(0)}.APPEND(\mathcal{C})$ 

6  $k \leftarrow 0$ 
7 while  $k \geq 0$  do
8    $k \leftarrow k + 1$ 
9    $\mathbb{L}^{(k \cdot \zeta(\mathcal{G}))} \leftarrow []$ 
10  for  $\beta \in \mathcal{G}$  with  $k \cdot \zeta(\mathcal{G}) - \|\beta\| \geq 0$  do
11     $w \leftarrow k \cdot \zeta(\mathcal{G}) - \|\beta\|$ 
12    if  $\mathbb{L}^{(w)} \neq []$  then
13      for  $\mathcal{C} \in \mathbb{L}^{(w)}$  do
14         $\mathcal{S} \leftarrow \mathcal{C}.\mathcal{S}$ 
15         $\mathcal{I} \leftarrow \mathcal{C}.\mathcal{I}$ 
16        for all ordered  $\mathcal{X} \subseteq \mathcal{S}$  with  $\#\mathcal{X} = \#\beta$  do
17          if  $\beta(\mathcal{X})$  is a new signal with respect to  $\mathcal{S}$  then
18             $\mathcal{S}' \leftarrow \mathcal{S}.APPEND(\beta(\mathcal{X}))$ 
19             $\mathcal{I}' \leftarrow \mathcal{I}.APPEND((\beta, \mathcal{X}))$ 
20             $\mathcal{C}' \leftarrow \text{CIRCUIT}(\mathcal{S}', \mathcal{I}')$ 
21             $\mathbb{L}^{(k \cdot \zeta(\mathcal{G}))}.APPEND(\mathcal{C}')$ 
22            if  $\mathcal{C}'$  implements  $F$  then
23              return  $\mathcal{C}'$ 

```

---

nodes such that for each circuit  $\mathcal{C}$  in  $\mathbb{L}^{(k \cdot \zeta(\mathcal{G}))}$  we have  $\|\mathcal{C}\| = k \cdot \zeta(\mathcal{G})$ . Moreover, each node in  $\mathbb{L}^{(k \cdot \zeta(\mathcal{G}))}$  is constructed from some node in  $\mathbb{L}^{(h \cdot \zeta(\mathcal{G}))}$  for some  $h < k$  by the so-called single-gate extension. We now explain this process in more details.

The creation of a node in  $\mathbb{L}^{(k \cdot \zeta(\mathcal{G}))}$  starts by selecting a logic gate  $\beta \in \mathcal{G}$  with  $w = k \cdot \zeta(\mathcal{G}) - \|\beta\| \geq 0$  (see *Line 10*). Note that we require that  $w \geq 0$  since new nodes in  $\mathbb{L}^{(k \cdot \zeta(\mathcal{G}))}$  will be single-gate extensions of the nodes in  $\mathbb{L}^{(w)}$ , whose area will be  $w + \|\beta\| = k \cdot \zeta(\mathcal{G})$ . Then, for each node  $\mathcal{C}$  in  $\mathbb{L}^{(w)}$  (see *Line 13*), we select an ordered subset  $\mathcal{X}$  of the signal set of the node and wire the signals in  $\mathcal{X}$  to the input pins of the gate  $\beta$ . This corresponds to a single gate extension of  $\mathcal{C}$  (see *Line 18 to Line 20*), and the new node is appended to the list  $\mathbb{L}^{(k \cdot \zeta(\mathcal{G}))}$  (see *Line 21*).

Note that in our implementation of Algorithm 1, the gates in  $\mathcal{G}$  are sorted in increasing order of their sizes. When two gates are of the same size, their order can be arbitrary. Let  $\{\beta_0, \beta_1, \dots, \beta_{s-1}\}$  be the ordered gate set. When  $\beta_j$  is selected, if  $k \cdot \zeta(\mathcal{G}) - \|\beta_j\| < 0$ , then we can stop inspecting the remaining gates since for any  $t > 0$ , we have  $k \cdot \zeta(\mathcal{G}) - \|\beta_j\| < 0$  according to the ordering of the gates. If  $w = k \cdot \zeta(\mathcal{G}) - \|\beta_j\| \geq 0$  and  $\mathbb{L}^{(w)}$  is nonempty, we will generate all possible single-gate extensions of the circuits in  $\mathbb{L}^{(w)}$  with the gate  $\beta_j$ . All these circuits are of area  $k \cdot \zeta(\mathcal{G})$  and will be stored in  $\mathbb{L}^{(k \cdot \zeta(\mathcal{G}))}$ . During this process, the algorithm will return whenever a circuit implementing  $F$  is encountered (see *Line 22* and *Line 23*). Our algorithm systematically explore the space of circuits with area  $k \cdot \zeta(\mathcal{G})$

in the increasing order of  $k$ . Therefore, the first circuit we encounter in the search that implements a given vectorial Boolean function  $F$  is the smallest circuit for  $F$ .

We are also able to prove that the circuit returned by Algorithm 1 does not possess any redundant nodes. Suppose there is a circuit  $\mathcal{C}_1$  with redundant nodes that realizes a given  $F$ , and is the outcome of our algorithm. It can be readily demonstrated that the circuit  $\mathcal{C}_2$ , which consists of all the nodes in  $\mathcal{C}_1$  except the redundant ones, also implements  $F$ . Moreover, the area of  $\mathcal{C}_2$  is smaller than that of  $\mathcal{C}_1$ , and this conflicts with the conclusion in our previous paragraph. In other words,  $\mathcal{C}_1$  will not be the circuit that is returned.

### 3.2 A Comparison with LIGHTER, PEIGEN, and Our Algorithm

We highlight the critical difference between our algorithm and LIGHTER [JPST17b] together with its improved version PEIGEN [BGLS19].

**The creation of new circuits.** In our algorithm, any new circuit (node) created during the search is a single-gate extension of some previously created circuit which is called the source node. The signal sequence of the newly created node is obtained by appending a new signal derived from the application of a single basic gate in  $\mathcal{G}$  to some signals in the signal sequence of the source node. With this approach, both reversible and irreversible circuits are covered by the search space.

In contrast, when LIGHTER or PEIGEN is executed for an  $n \times n$  S-box, any new node is represented as a vectorial Boolean function from  $\mathbb{F}_2^n$  to  $\mathbb{F}_2^n$ , which is obtained by applying a *reversible* transformation to the  $n$  output bits representing the source node. Sometimes the reversible transformations are not basic gates but compositions of certain basic gates, which form the so-called  $\mathcal{B}$ -set. Each transformation in the  $\mathcal{B}$ -set of LIGHTER and PEIGEN corresponds to a *reversible* vectorial Boolean function  $F : \mathbb{F}_2^m \rightarrow \mathbb{F}_2^m$  mapping  $(x_0, \dots, x_{m-2}, x_{m-1})$  to  $(x_0, x_1, \dots, x_{m-2}, x_{m-1} + f(x_0, x_1, \dots, x_{m-2}))$ , where  $f : \mathbb{F}_2^{m-1} \rightarrow \mathbb{F}_2$  is an arbitrary Boolean function. With this approach, the search space of LIGHTER or PEIGEN is intrinsically *incomplete* and they are only applicable in the context of reversible S-boxes without any modifications of the search algorithm. Therefore, even with unlimited computational power, LIGHTER and PEIGEN may miss the optimal circuit for a given vectorial Boolean function. Let us check a concrete example.

**Example 3.** Consider the  $3 \times 3$  S-box with substitution table [6, 2, 0, 7, 3, 4, 1, 5], whose algebraic representation is

$$\begin{cases} y_0 = x_2x_0 + x_1x_0 + x_2 \\ y_1 = x_2x_0 + x_1x_0 + x_1 + 1 \\ y_2 = x_2x_1 + x_2 + x_1 + x_0 + 1 \end{cases},$$

where  $x_0$  is the least significant input bit and  $y_0$  is the least significant output bit. Our algorithm with the full gate set of the TSMC 65nm library gives a circuit with 8.5 GE:

$$\begin{aligned} x_3 &= \text{MUXI}(x_0, x_2, x_1), & x_4 &= \text{NOR}(x_2, x_1), & x_5 &= \text{XOR}(x_0, x_4), \\ x_6 &= \text{AOI21}(x_2, x_1, x_3), & x_7 &= \text{NOR}(x_4, x_6). \end{aligned}$$

Note that we need to set  $y_2 = x_5$ ,  $y_1 = x_3$ , and  $y_0 = x_7$ . With the TSMC 65nm library PEIGEN generates the following circuit:

$$\begin{aligned} t_1 &= \text{NOR}(x_1, x_2) & x_0 &= \text{XOR}(x_0, t_1) & x_1 &= \text{XNOR}(x_1, x_2) \\ t_2 &= \text{NOR}(x_1, x_0) & x_2 &= \text{XNOR}(x_2, t_2) & x_1 &= \text{XOR}(x_1, x_2) \end{aligned}$$

with  $y_2 = x_0$ ,  $y_1 = x_2$ ,  $y_0 = x_1$ , whose area is 12 GE.

Even if we apply our algorithm with reduced gate set {NOT, AND, NAND, NANDN, OR, NOR, NORN, XOR, XNOR, AND3, NAND3, OR3, NOR3} which is fully supported by LIGHTER or PEIGEN, our algorithm still leads to better circuit:

$$\begin{aligned} x_3 &= \text{NOR}(x_2, x_1) & x_4 &= \text{XOR}(x_0, x_3) & x_5 &= \text{NAND}(x_2, x_1) \\ x_6 &= \text{NAND3}(x_0, x_4, x_5) & x_7 &= \text{XNOR}(x_2, x_6) & x_8 &= \text{XOR}(x_1, x_6) \end{aligned}$$

with  $y_2 = x_4, y_1 = x_8, y_0 = x_7$ , whose area is 11 GE.

**The order of the search space exploration.** In each round of the top loop of Algorithm 1, a list of circuits with area  $k \cdot \zeta(\mathcal{G})$  for some fixed  $k$  is created from a subset of circuits from

$$\bigcup_{h=0}^{k-1} \{\mathcal{C} \text{ is a } \mathcal{G}\text{-circuit} : \|\mathcal{C}\| = h \cdot \zeta(\mathcal{G})\}.$$

Therefore, in our algorithm, all newly created circuits within each top loop iteration is of the same area, but the source circuits of the created nodes can come from different layers whose corresponding areas are less than  $k \cdot \zeta(\mathcal{G})$ . This order of the search space exploration is illustrated in Figure 5(a). While in LIGHTER and PEIGEN, within each top loop iteration, new nodes in

$$\bigcup_{h=k+1}^l \{\mathcal{C} \text{ is a } \mathcal{G}\text{-circuit} : \|\mathcal{C}\| = h \cdot \zeta(\mathcal{G})\}$$

for some integer  $l > h$  are created from a subset of circuits in  $\{\mathcal{C} \text{ is a } \mathcal{G}\text{-circuit} : \|\mathcal{C}\| = k \cdot \zeta(\mathcal{G})\}$ , that is, the source circuits are of the same size while the created circuits are of different sizes and larger than the source circuits. This search order is visualized in Figure 5(b).

Notably, in our algorithm, the selected layers having areas less than  $k \cdot \zeta(\mathcal{G})$  are also arranged in a specific order. Given that single-gate extension gives rise to new nodes with the area of gates in an increasing order, the layer chosen first is the one whose area is closest to  $k \cdot \zeta(\mathcal{G})$ , while the last-chosen layer has the smallest area. This ordering runs counter to intuitive expectations. In other words, the blue-colored layer in Figure 5(a) is the first to be selected, and the grey-colored layer is the last. This underlying reason accounts for the fact that the circuit yielded by our algorithm generates the signal associated with MUXI gate earlier than the signal associated with NOR gate in Example 3.

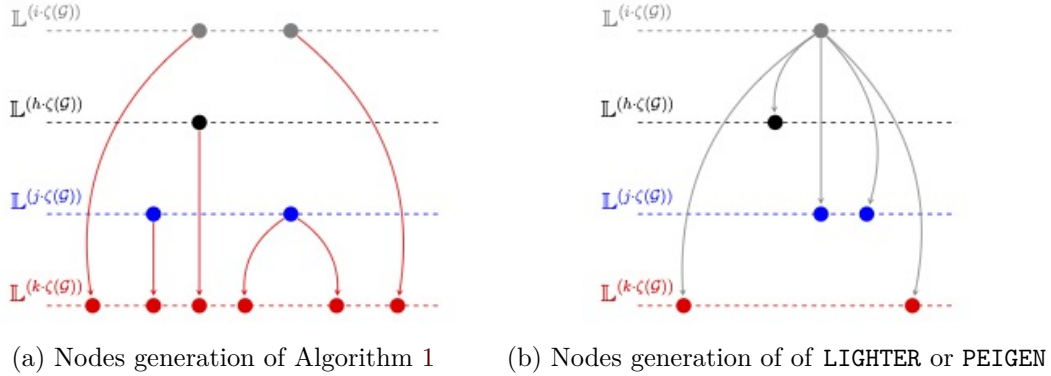


Figure 5: A comparison of Algorithm 1 and LIGHTER/PEIGEN

**Bugs identified in LIGHTER and PEIGEN.** The functionalities of MAOI1 and MOAI1 can be described as

$$\begin{cases} \text{MAOI1}(a, b, c, d) = \neg((a \wedge b) \vee (\neg(c \vee d))) \\ \text{MOAI1}(a, b, c, d) = \neg((a \vee b) \wedge (\neg(c \wedge d))) \end{cases}.$$

Under the condition that  $a = c$  and  $b = d$ , MAOI1 is equivalent to XOR and MOAI1 is equivalent to XNOR. Both LIGHTER and PEIGEN exploit these equivalences to reduce the circuit size. For example, if  $\|\text{MAOI1}\| < \|\text{XOR}\|$  in a specific technology library, then all XOR gates will be replaced by MAOI1 gates in the optimized circuits. This type of replacement is considered in the construction of the so-called  $\mathcal{B}$ -set. However, the  $\mathcal{B}$ -sets constructed in LIGHTER and PEIGEN are both incomplete.

- In the  $\mathcal{B}$ -set of LIGHTER, the instructions formed by MAOI1 miss the combinations of ANDN and ORN (fixed by PEIGEN).
- In the  $\mathcal{B}$ -set of LIGHTER and PEIGEN, the instructions formed by XOR miss the combinations of AND3, NAND3, OR3, NOR3, NAND and NOR.

The latter issue implies that the set of instructions formed by XOR is a strict subset of that formed by MAOI1. When  $\|\text{XOR}\| > \|\text{MAOI1}\|$ , this will not cause any problems. However, when  $\|\text{XOR}\| < \|\text{MAOI1}\|$ , this may lead to suboptimal results. We note that the case  $\|\text{XOR}\| < \|\text{MAOI1}\|$  indeed happens (e.g., in the STD90/MDL90 350nmnm library).

Finally, we would like to emphasize that the  $\mathcal{B}$ -set in LIGHTER and PEIGEN is constructed manually and it is quite likely to miss some meaningful combinations when the number of basic gates is large. For our method, all basic gates are considered separately and thus we do not face such difficulties.

## 4 Optimizing the Implementation of $4 \times 4$ S-boxes

Although the algorithmic framework given in Algorithm 1 is guaranteed to find the optimal solution, it has two drawbacks. Firstly, it is too memory-extensive to practically synthesize most  $4 \times 4$  S-boxes. Secondly, it does not consider the circuit delay, an equally important factor that affects the performance of the synthesized circuit. In this section, we are going to shrink the search space of Algorithm 1 based on some heuristics to make it more efficient without losing too much in the qualities of the solutions. Moreover, we will modify the algorithmic framework such that it can synthesize area-optimized circuit under given timing constraints.

**Reducing the Search Space.** The most memory consuming part of Algorithm 1 is the storage of  $\mathbb{L}^{(k, \zeta(\mathcal{G}))}$ . Now, we impose an order among the circuits in  $\mathbb{L}^{(k, \zeta(\mathcal{G}))}$ . Let  $\mathcal{C} = \text{CIRCUIT}(\mathcal{S}, \mathcal{I})$ . The distance from  $\mathcal{C}$  to the target vectorial Boolean function  $F = (f_0, \dots, f_{m-1})$ , denoted by  $d(\mathcal{C}, F)$ , is defined as the number of elements in the set  $\mathcal{S} - \{f_0, \dots, f_{m-1}\}$ . The circuits in  $\mathbb{L}^{(k, \zeta(\mathcal{G}))}$  are listed in increasing order of  $d(\mathcal{C}, F)$ . If there is a tie, the circuits with less number of gates are listed first. If there still is a tie, the circuits are listed according to its generation order. Let  $\delta(\mathbb{L}^{(k, \zeta(\mathcal{G}))}, F) = \min\{d(\mathcal{C}, F) : \mathcal{C} \in \mathbb{L}^{(k, \zeta(\mathcal{G}))}\}$ . In our algorithm, only circuits with size less or equal to  $\delta(\mathbb{L}^{(k, \zeta(\mathcal{G}))}, F) + e$  are stored and all other circuits are dropped, where  $e$  is a small integer. In practice, we typically set  $e$  to 0 or 1. Obviously, this approach makes the search space incomplete. However, experimental results show that we do not lose much in the quality of the solutions, which will be seen in Section 5.

**Taking Timing Constraints into Account.** According to Equation (1), the delay of each node visited in our algorithm can be computed if the delay of each gate is known. For a specific technology library, the latency or delay of a gate at a typical condition (e.g., 1.2V, 25°C) is modeled as the following formula:

$$\Delta(\beta) = D_I + K_{\text{Load}} \cdot C, \quad (2)$$

where  $D_I$  measured in **ns** is the so-called intrinsic delay of the gate  $\beta$ ,  $K_{\text{Load}}$  is the load delay factor (**ns/pF**), and  $C$  is the total output load capacitance (**pF**). Typically, the term  $K_{\text{Load}} \cdot C$  is much smaller than  $D_I$ , and thus in this work we ignore  $K_{\text{Load}} \cdot C$  and use  $D_I$  to approximate  $\mathfrak{D}(\beta)$ . Note that the intrinsic delay  $D_I$  of a specific gate is a variable depending on factors like drive strength and input-output conditions. In practice, we select the maximum possible value of  $D_I$  to approximate the delay of the gate.

Given a timing constraint that the delay of the circuit should be less than or equal to  $T$ , we can fire up the algorithm and before output the implementation we check its delay. If the delay is less than or equal to  $T$ , we output the circuit. Otherwise, we continue the search until a area threshold is reached. We emphasize that unlike area, a circuit with lower delay may appear in a layer that is farther away from the root node.

*Remark.* To guarantee that the approximation employed is sound, we always feed the circuits with gate-level representations returned by our method to some CAD tools like design compiler to confirm that the latency is indeed reduced as expected.

**Further optimization with ABC.** Following Jean et al.’s approach [JPST17b], we try to further optimize the circuit by applying ABC [abc, BM10] to the implementation produced by our method, i.e., the output circuit of our method is used as a starting point for further optimization by ABC. However, in most cases, the application of ABC makes the implementation less efficient in terms of area.

The reader may wonder why the application of ABC makes the implementation worse than the input circuit. This is because internally ABC transforms the input circuit into a technology-independent representation called And-Inverter Graph (AIG), and then performs some technology-independent optimizations. Finally, the technology-independent representation is mapped to an implementation based on a specific technology library. This process may introduce some quality loss due to the conversion between the technology-dependent and technology-independent domains. Moreover, experimental results show that our method leads to highly optimized technology-dependent implementations such that the room for further improvements (e.g., by ABC) is quite limited.

## 5 Applications and Results

Firstly, we apply our method to a  $\mathbb{F}_2^4 \rightarrow \mathbb{F}_2^4$  subcomponent of the tower field implementation of AES. Then we go through a list of  $3 \times 3$  and  $4 \times 4$  S-boxes. Finally, we optimize the implementations of some  $8 \times 8$  S-boxes constructed from  $4 \times 4$  S-boxes.

**Optimizing the Circuit for the AES S-box.** Currently, the most compact implementation of the AES S-box follows the tower field approach [ME19]. In this implementation, we can identify a subcomponent which can be regarded as a  $4 \times 4$  S-box (see [ME19, Sect. 5.3]). Significantly, this S-box serves as a pivotal element in both the circuit optimized for achieving minimal area and the circuit engineered to optimize logic depth in [ME19]. Any improvement of the implementation of this subcomponents directly leads to improvement of the overall AES S-box. The truth table of the subcomponent is given by [0, c, 8, 4, 3, a, 7, 6, 2, d, 5, e, 1, 9, b, f]. Applying our method with STD90/MDL90 350nm library gives the following circuit

$$\begin{aligned} x_4 &= \text{NOR}(x_2, x_0), & x_5 &= \text{NAND}(x_3, x_1), & x_6 &= \text{XNOR}(x_4, x_5), \\ x_7 &= \text{MUX}(x_3, x_6, x_2), & x_8 &= \text{MUX}(x_1, x_6, x_0), & x_9 &= \text{NAND}(x_2, x_0), \\ x_{10} &= \text{NAND}(x_7, x_9), & x_{11} &= \text{MUX}(x_{10}, x_3, x_2), & x_{12} &= \text{NAND}(x_8, x_9), \\ x_{13} &= \text{MUX}(x_{12}, x_1, x_0). \end{aligned}$$

with  $y_0 = x_{11}$ ,  $y_1 = x_7$ ,  $y_2 = x_{13}$ , and  $y_3 = x_8$ , whose area is 16.65 GE. In [ME19], the authors give a circle under STD90/MDL90 350nm library:

Table 2: S-boxes used for comparison

S-box	Lookup table (in hexadecimal)																Ref.
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	
3-way	7	2	4	5	1	6	3	0									[DGV93]
ctc2	7	6	0	4	2	5	1	3									[Cou07]
PRINTcipher	0	1	3	6	7	4	5	2									[KLP10]
SEA	0	5	6	7	4	3	1	2									[SPGQ06]
Joltik	e	4	b	2	3	8	0	9	1	a	7	f	6	c	5	d	[JNP14]
Joltik <sup>-1</sup>	6	8	3	4	1	e	c	a	5	7	9	2	d	f	0	b	[JNP14]
RECTANGLE	6	5	c	a	1	e	7	9	b	0	3	d	8	f	4	2	[ZBL <sup>+</sup> 15]
RECTANGLE <sup>-1</sup>	9	4	f	a	e	1	0	6	c	7	3	8	2	b	5	d	[ZBL <sup>+</sup> 15]
SKINNY	c	6	9	0	1	a	2	b	3	8	5	d	4	e	7	f	[BJK <sup>+</sup> 16]
SKINNY <sup>-1</sup>	3	4	6	8	c	a	1	e	9	2	5	7	0	b	d	f	[BJK <sup>+</sup> 16]
TWINE	c	0	f	a	2	b	9	5	8	3	d	7	1	e	6	4	[ZJB06]
PRESENT	c	5	6	b	9	0	a	d	3	e	f	8	4	7	1	2	[BKL <sup>+</sup> 07]
LBlock s <sub>0</sub>	e	9	f	0	d	4	a	b	1	2	8	3	7	6	c	5	[WZ11]
LBlock s <sub>1</sub>	4	b	e	9	f	d	0	a	7	c	5	6	2	8	1	3	[WZ11]
LBlock s <sub>3</sub>	7	6	8	b	0	f	3	e	9	a	c	d	5	2	4	1	[WZ11]
GIFT	1	a	4	c	6	f	3	9	2	d	b	7	5	0	8	e	[BPP <sup>+</sup> 17]
Midori s <sub>0</sub>	c	a	d	3	e	b	f	7	8	9	1	5	0	2	4	6	[BBI <sup>+</sup> 15]
Saturnin s <sub>0</sub>	0	6	e	1	f	4	7	d	9	8	c	5	2	a	3	b	[CDL <sup>+</sup> 20]
Orthros	1	0	2	4	3	8	6	d	9	a	b	e	f	c	7	5	[BIL <sup>+</sup> 21]

$$\begin{aligned}
x_4 &= \text{NAND}(x_3, x_1), & x_5 &= \text{NOR}(x_2, x_0), & x_6 &= \text{XNOR}(x_4, x_5), \\
x_7 &= \text{MUX}(x_2, x_1, 1), & x_8 &= \text{MUX}(x_0, x_3, 1), & x_9 &= \text{MUX}(x_1, x_6, x_0), \\
x_{10} &= \text{MUX}(x_6, x_0, x_7), & x_{11} &= \text{MUX}(x_3, x_6, x_2), & x_{12} &= \text{MUX}(x_6, x_2, x_8).
\end{aligned}$$

with  $y_0 = x_{12}$ ,  $y_1 = x_{11}$ ,  $y_2 = x_{10}$ , and  $y_3 = x_9$ , whose area is 18.31 GE. If we apply LIGHTER/PEIGEN to this component, the circuit obtained costs 21.65 GE.

**Optimizing  $3 \times 3$  and  $4 \times 4$  S-boxes.** We apply our method to the  $3 \times 3$  and  $4 \times 4$  S-boxes listed in Table 2 with several CMOS technology libraries, where the programs are executed within a server that is furnished with an Intel Xeon w9-3475X(72) processor, operating at a frequency of 4.6 GHz and equipped with 503 GB of memory. Some sample results can be found in Table 3 and 4, and their running time can be found in Table 5 and 6. In certain cases, the improvement is significant. From Table 3 we can see that the area of the S-box Midori s<sub>0</sub> under the TSMC 28nm library is reduced from 28 GE to 13.98 GE. However, since the search space for  $4 \times 4$  S-boxes is incomplete and the memory usage is extensive, in practice we may encounter the following possibilities. Firstly, the search may fail to output any result before exhausting the system memory (see the row for TWINE in Table 3). Secondly, the circuits produced by our method may be inferior to the circuit generated by LIGHTER or PEIGEN (see the row for GIFT in Table 4). Besides, despite the fact that we have incorporated several heuristic strategies to curtail the search space, the search space of our method remains larger than that of LIGHTER and PEIGEN. Consequently, our method invariably requires a significantly longer running time, which might pose challenges in terms of overall efficiency and practical usability. Therefore, our method should be regarded as a complement of LIGHTER and PEIGEN. We refer the reader to Appendix C for more results with concrete implementations.

Moreover, if we take the latency into account as discussed in Section 4, we obtain some circuits which are superior to previous results with respect to both area and delay. For example, for the S-box Midori s<sub>0</sub>, we obtain a circuit with area  $76.38 \mu\text{m}^2$  and delay 1.36 ns (see Table 7), while the area and delay of the circuit generated by PEIGEN are  $115.42 \mu\text{m}^2$  and 2.70 ns, respectively. For the S-box LBlock s<sub>0</sub>, RECTANGLE and Orthros, we obtain

Table 3: Synthesis results with the TSMC library, where "✗" indicates that the S-box is a 3-bit S-box and it is not supported by LIGHTER, "Error" indicates that LIGHTER returns wrong circuits, "No res" indicates that the algorithm can not return any result.

S-box	TSMC 65nm			TSMC 28nm		
	LIGHTER	PEIGEN	Ours	LIGHTER	PEIGEN	Ours
3-way	✗	11.50 GE	11.00 GE	✗	12.66 GE	10.65 GE
ctc2	✗	10.50 GE	9.50 GE	✗	12.00 GE	8.99 GE
PRINTcipher	✗	10.50 GE	10.50 GE	✗	12.00 GE	9.98 GE
SEA	✗	10.50 GE	10.00 GE	✗	12.00 GE	9.65 GE
Joltik	14.00 GE	14.00 GE	14.00 GE	16.00 GE	16.00 GE	13.99 GE
Joltik <sup>-1</sup>	14.00 GE	14.00 GE	14.00 GE	16.00 GE	16.00 GE	15.66 GE
RECTANGLE	21.50 GE	22.00 GE	20.50 GE	25.00 GE	No res	19.31 GE
RECTANGLE <sup>-1</sup>	21.50 GE	22.00 GE	19.00 GE	25.00 GE	No res	20.98 GE
SKINNY	14.00 GE	14.00 GE	14.00 GE	16.00 GE	16.00 GE	13.99 GE
SKINNY <sup>-1</sup>	14.00 GE	14.00 GE	14.00 GE	16.00 GE	16.00 GE	15.66 GE
TWINE	25.00 GE	No res	No res	28.34 GE	No res	No res
PRESENT	24.00 GE	24.50 GE	No res	27.33 GE	No res	No res
LBlock s <sub>0</sub>	19.00 GE	19.00 GE	16.50 GE	22.00 GE	22.00 GE	16.32 GE
LBlock s <sub>1</sub>	19.00 GE	19.00 GE	16.50 GE	22.00 GE	22.00 GE	16.32 GE
LBlock s <sub>3</sub>	19.00 GE	19.00 GE	16.50 GE	22.00 GE	22.00 GE	16.65 GE
GIFT	19.00 GE	19.00 GE	19.50 GE	22.00 GE	22.00 GE	21.99 GE
Midori s <sub>0</sub>	24.00 GE	24.00 GE	16.50 GE	28.00 GE	No res	13.98 GE
Saturnin s <sub>0</sub>	21.00 GE	21.00 GE	19.50 GE	24.00 GE	24.00 GE	20.31 GE
Orthros	27.50 GE	No res	16.00 GE	Error	No res	15.31 GE

similar results (see Table 7). Note that in our algorithm, the delay is approximated according to Equation (2), to confirm the validity of this approximation, the rows denoted as "Design Compiler" in Table 7 report the synthesis results generated by Design Compiler when the gate-level description of the circuits are fed into it.

Finally, we would like to mention that our tool can be applied to those  $8 \times 8$  S-boxes constructed from  $4 \times 4$  S-boxes [BGG<sup>+</sup>16, BGG<sup>+</sup>17, CDL15, LW14], since the improvement of the implementations of the involved  $4 \times 4$  S-boxes directly leads to improvement of the  $8 \times 8$  S-boxes constructed from them.

## 6 Conclusion and Discussion

We present a technology-dependent synthesis tool which can transform a small S-box specified with a truth table directly into a netlist of cells of the target CMOS technology library. For  $3 \times 3$  S-box, our tool performs a systematic search and therefore is able to identify the global optimal implementation. For  $4 \times 4$  S-boxes, we prune the search space based on some heuristics and thus the tool is not guaranteed to reach the global optimum. However, experimental results show that our tool still outperforms LIGHTER and PEIGEN for some  $4 \times 4$  S-boxes. In particular, we obtain the most compact implementation of the  $\mathbb{F}_{24}$ -inverter, leading to improved implementation of the AES S-box. For certain S-boxes, our method generates circuits superior to known implementations with respect to both area and latency. Our tool can be regarded as a complement of LIGHTER and PEIGEN.

Table 4: Synthesis results with the SMIC library, where "✗" indicates that the S-box is a 3-bit S-box and it is not supported by LIGHTER, "Error" indicates that LIGHTER returns wrong circuits, "No res" indicates that the algorithm can not return any result.

S-box	SMIC 130nm			SMIC 65nm		
	LIGHTER	PEIGEN	Ours	LIGHTER	PEIGEN	Ours
3-way	✗	10.65 GE	10.32 GE	✗	10.75 GE	10.25 GE
ctc2	✗	9.99 GE	9.33 GE	✗	9.75 GE	9.00 GE
PRINTcipher	✗	9.99 GE	9.99 GE	✗	9.75 GE	9.75 GE
SEA	✗	9.99 GE	9.33 GE	✗	9.75 GE	9.25 GE
Joltik	13.32 GE	13.32 GE	13.32 GE	13.00 GE	13.00 GE	13.00 GE
Joltik <sup>-1</sup>	13.32 GE	13.32 GE	13.32 GE	13.00 GE	13.00 GE	13.00 GE
RECTANGLE	20.31 GE	20.31 GE	18.98 GE	19.75 GE	19.75 GE	18.75 GE
RECTANGLE <sup>-1</sup>	20.31 GE	20.31 GE	20.65 GE	19.75 GE	19.75 GE	20.00 GE
SKINNY	13.32 GE	13.32 GE	13.32 GE	13.00 GE	13.00 GE	13.00 GE
SKINNY <sup>-1</sup>	13.32 GE	13.32 GE	13.32 GE	13.00 GE	13.00 GE	13.00 GE
TWINE	23.97 GE	23.97 GE	No res	23.25 GE	23.25 GE	No res
PRESENT	22.64 GE	22.64 GE	No res	22.25 GE	22.25 GE	No res
LBlock s <sub>0</sub>	17.98 GE	17.98 GE	15.65 GE	17.50 GE	17.50 GE	16.00 GE
LBlock s <sub>1</sub>	17.98 GE	17.98 GE	15.65 GE	17.50 GE	17.50 GE	16.00 GE
LBlock s <sub>3</sub>	17.98 GE	17.98 GE	15.65 GE	17.50 GE	17.50 GE	16.00 GE
GIFT	17.98 GE	17.98 GE	18.31 GE	17.50 GE	17.50 GE	18.25 GE
Midori s <sub>0</sub>	22.64 GE	22.64 GE	15.00 GE	22.00 GE	22.00 GE	15.00 GE
Saturnin s <sub>0</sub>	19.98 GE	19.98 GE	19.99 GE	19.50 GE	19.50 GE	18.75 GE
Orthros	Error	25.97 GE	15.65 GE	25.75 GE	25.75 GE	15.50 GE

Table 5: Running time with the TSMC 65nm library, where PEIGEN consumes  $6.73 \times 10^{-2}$  s and 164.36 s respectively for pre-computing  $3 \times 3$  and  $4 \times 4$  S-boxes.

S-box	LIGHTER	PEIGEN	Ours
3-way	✗	$2.73 \times 10^{-4}$ s	48.82 s
ctc2	✗	$2.75 \times 10^{-3}$ s	28.12 s
PRINTcipher	✗	$6.93 \times 10^{-4}$ s	40.92 s
SEA	✗	$2.11 \times 10^{-4}$ s	43.93 s
Joltik	16.92 s	0.12 s	73.41 s
RECTANGLE	1269.47 s	10.93 s	10199.40 s
SKINNY	15.98 s	0.10 s	70.60 s
TWINE	4065.69 s	No res	No res
PRESENT	2282.10 s	29.67 s	No res
LBlock s <sub>0</sub>	225.17 s	1.94 s	1758.62 s
GIFT	368.17 s	1.50 s	29009.40 s
Midori s <sub>0</sub>	6663.90 s	23.36 s	4477.20 s
Saturnin s <sub>0</sub>	364.95 s	5.40 s	36356.00 s
Orthros	6826.29 s	No res	73660.7 s

## References

- [abc] ABC: A System for Sequential Synthesis and Verification. <https://people.eecs.berkeley.edu/~alanmi/abc/>.
- [ARS<sup>+</sup>15a] Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen,



Table 6: Running time with the SMIC 65nm library, where PEIGEN consumes  $1.67 \times 10^{-4}$  s and  $3.14 \times 10^{-4}$  s respectively for pre-computing  $3 \times 3$  and  $4 \times 4$  S-boxes.

S-box	LIGHTER	PEIGEN	Ours
3-way	$\times$	$1.61 \times 10^{-2}$ s	50.14 s
ctc2	$\times$	$3.26 \times 10^{-3}$ s	28.15 s
PRINTcipher	$\times$	$7.44 \times 10^{-2}$ s	41.30 s
SEA	$\times$	$2.60 \times 10^{-4}$ s	1.10 s
Joltik	11.61 s	335.54 s	115.07 s
RECTANGLE	1364.44 s	439.82 s	85236.50 s
SKINNY	11.76 s	0.14 s	118.61 s
TWINE	3133.17 s	95.92 s	No res
PRESENT	2081.53 s	497.96 s	No res
LBlock $s_0$	228.70 s	2.35 s	4566.73 s
GIFT	281.00 s	2.38 s	36016.9 s
Midori $s_0$	5568.49 s	48.95 s	8943.34 s
Saturnin $s_0$	396.53 s	10.23 s	47245.10 s
Orthros	7493.46 s	150.06 s	23874.70 s

and Michael Zohner. Ciphers for MPC and FHE. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 430–454. Springer, 2015. doi:10.1007/978-3-662-46800-5\_17.

- [ARS<sup>+</sup>15b] Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. Ciphers for MPC and FHE. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 430–454. Springer, 2015. doi:10.1007/978-3-662-46800-5\_17.
- [BBI<sup>+</sup>15] Subhadeep Banik, Andrey Bogdanov, Takanori Isobe, Kyoji Shibutani, Harunaga Hiwatari, Toru Akishita, and Francesco Regazzoni. Midori: A block cipher for low energy. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II*, volume 9453 of *Lecture Notes in Computer Science*, pages 411–436. Springer, 2015. doi:10.1007/978-3-662-48800-3\_17.
- [BGG<sup>+</sup>16] Erik Boss, Vincent Grosso, Tim Güneysu, Gregor Leander, Amir Moradi, and Tobias Schneider. Strong 8-bit sboxes with efficient masking in hardware. In Benedikt Gierlichs and Axel Y. Poschmann, editors, *Cryptographic Hardware and Embedded Systems - CHES 2016 - 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings*, volume 9813 of *Lecture Notes in Computer Science*, pages 171–193. Springer, 2016. doi:10.1007/978-3-662-53140-2\_9.
- [BGG<sup>+</sup>17] Erik Boss, Vincent Grosso, Tim Güneysu, Gregor Leander, Amir Moradi, and Tobias Schneider. Strong 8-bit sboxes with efficient masking in hardware

Table 7: Optimization with timing constraints under the SMIC 130nm library. Note that the "Delay" columns record the delay of the corresponding signals in a scaled unit, where  $\mathbb{X}(\text{XOR}) = 1$ , while the "Design Compiler" row records the results reported by Design Compiler

S-box	Lighter	Delay	Peigon	Delay	Our method	Delay	
Midori $s_0$	$x_4 = \text{XNOR}(x_0, x_2)$	0.94	$x_4 = \text{XOR}(x_2, x_0)$	1.00	$x_4 = \text{XNOR}(x_3, x_0)$	0.94	
	$x_5 = \text{NOR}(x_4, x_2)$	1.38	$x_5 = \text{NAND}(x_4, x_0)$	1.34	$x_5 = \text{MUX}(x_4, x_2, x_0)$	2.41	
	$x_6 = \text{XOR}(x_3, x_5)$	2.38	$x_6 = \text{XOR}(x_3, x_5)$	2.34	$x_6 = \text{AO21}(x_3, x_2, x_1)$	2.24	
	$x_7 = \text{XOR}(x_1, x_6)$	3.38	$x_7 = \text{NAND}(x_4, x_6)$	2.68	$x_7 = \text{NOR}(x_3, x_0)$	0.44	
	$x_8 = \text{NOR}(x_6, x_4)$	2.82	$x_8 = \text{XNOR}(x_0, x_7)$	3.62	$x_8 = \text{NANDN}(x_7, x_6)$	3.36	
	$x_9 = \text{XNOR}(x_2, x_8)$	3.76	$x_9 = \text{XOR}(x_6, x_1)$	3.34	$x_9 = \text{NOR}(x_2, x_4)$	1.38	
	$x_{10} = \text{NOR}(x_7, x_9)$	4.20	$x_{10} = \text{XOR}(x_4, x_8)$	4.62	$x_{10} = \text{NOR3}(x_1, x_7, x_9)$	2.39	
	$x_{11} = \text{XNOR}(x_6, x_{10})$	5.14	$x_{11} = \text{NAND}(x_9, x_{10})$	4.96	$x_{11} = \text{NOR}(x_9, x_{10})$	2.83	
	$x_{12} = \text{XOR}(x_4, x_9)$	4.76	$x_{12} = \text{XNOR}(x_1, x_{11})$	5.90	$x_{12} = \text{NOR}(x_1, x_{10})$	2.83	
	$x_{13} = \text{NOR}(x_{11}, x_7)$	5.58	$x_{13} = \text{NAND}(x_9, x_{12})$	6.24	$x_{13} = \text{AOI21}(x_3, x_2, x_{12})$	3.58	
	$x_{14} = \text{XOR}(x_9, x_{13})$	6.58	$x_{14} = \text{XOR}(x_{10}, x_{13})$	7.24			
	$x_{15} = \text{XNOR}(x_7, x_{11})$	6.08	$x_{15} = \text{XOR}(x_9, x_{12})$	6.90			
	Design Compiler	$\ \mathcal{C}\  = 115.423197 \mu\text{m}^2$ $\mathbb{X}(\mathcal{C}) = 2.66 \text{ ns}$		$\ \mathcal{C}\  = 115.423197 \mu\text{m}^2$ $\mathbb{X}(\mathcal{C}) = 2.70 \text{ ns}$		$\ \mathcal{C}\  = 76.382999 \mu\text{m}^2$ $\mathbb{X}(\mathcal{C}) = 1.36 \text{ ns}$	
	LBlock $s_0$	$x_4 = \text{XNOR}(x_1, x_0)$	0.94	$x_4 = \text{XOR}(x_0, x_1)$	1.00	$x_4 = \text{NOT}(x_0)$	0.19
		$x_5 = \text{NOR}(x_3, x_2)$	0.44	$x_5 = \text{XNOR}(x_1, x_3)$	0.94	$x_5 = \text{MUXI}(x_2, x_1, x_4)$	1.11
$x_6 = \text{XOR}(x_4, x_5)$		1.94	$x_6 = \text{NOR}(x_4, x_2)$	1.44	$x_6 = \text{XNOR}(x_3, x_5)$	2.05	
$x_7 = \text{NAND}(x_2, x_6)$		2.28	$x_7 = \text{XNOR}(x_5, x_6)$	2.38	$x_7 = \text{MUXI}(x_2, x_0, x_6)$	2.97	
$x_8 = \text{XNOR}(x_0, x_7)$		3.22	$x_8 = \text{NOR}(x_2, x_3)$	0.44	$x_8 = \text{XOR}(x_1, x_7)$	3.97	
$x_9 = \text{NOR}(x_3, x_8)$		3.66	$x_9 = \text{XNOR}(x_4, x_8)$	1.94	$x_9 = \text{MUXI}(x_8, x_3, x_0)$	4.89	
$x_{10} = \text{XOR}(x_2, x_9)$		4.66	$x_{10} = \text{NOR}(x_7, x_9)$	2.82	$x_{10} = \text{NOR}(x_3, x_5)$	1.55	
$x_{11} = \text{XNOR}(x_8, x_3)$		4.16	$x_{11} = \text{XNOR}(x_3, x_{10})$	3.76	$x_{11} = \text{XOR}(x_2, x_{10})$	2.55	
$x_{12} = \text{NOR}(x_{11}, x_6)$		4.60	$x_{12} = \text{NAND}(x_{11}, x_7)$	4.10			
$x_{13} = \text{XNOR}(x_3, x_{12})$		5.54	$x_{13} = \text{XNOR}(x_2, x_{12})$	5.04			
Design Compiler		$\ \mathcal{C}\  = 91.659597 \mu\text{m}^2$ $\mathbb{X}(\mathcal{C}) = 1.86 \text{ ns}$		$\ \mathcal{C}\  = 91.659597 \mu\text{m}^2$ $\mathbb{X}(\mathcal{C}) = 1.75 \text{ ns}$		$\ \mathcal{C}\  = 79.777798 \mu\text{m}^2$ $\mathbb{X}(\mathcal{C}) = 1.51 \text{ ns}$	
RECT-ANGLE		$x_4 = \text{NOR}(x_0, x_1)$	0.44	$x_4 = \text{NOR}(x_0, x_1)$	0.44	$x_4 = \text{XNOR}(x_2, x_1)$	0.94
		$x_5 = \text{XNOR}(x_4, x_3)$	1.38	$x_5 = \text{XNOR}(x_4, x_3)$	1.38	$x_5 = \text{MUXI}(x_0, x_4, x_2)$	1.86
	$x_6 = \text{XOR}(x_2, x_5)$	2.38	$x_6 = \text{NOR}(x_2, x_5)$	1.82	$x_6 = \text{XNOR}(x_3, x_5)$	2.80	
	$x_7 = \text{NOR}(x_5, x_6)$	2.82	$x_7 = \text{XOR}(x_0, x_6)$	2.82	$x_7 = \text{MUXI}(x_1, x_0, x_3)$	0.92	
	$x_8 = \text{XOR}(x_0, x_7)$	3.82	$x_8 = \text{XNOR}(x_2, x_1)$	0.94	$x_8 = \text{MUXI}(x_6, x_7, x_4)$	3.72	
	$x_9 = \text{XOR}(x_6, x_1)$	3.38	$x_9 = \text{XOR}(x_7, x_1)$	3.82	$x_9 = \text{XOR}(x_6, x_7)$	3.80	
	$x_{10} = \text{XOR}(x_1, x_8)$	4.82	$x_{10} = \text{NAND}(x_5, x_8)$	1.72	$x_{10} = \text{MUX}(x_4, x_7, x_9)$	5.27	
	$x_{11} = \text{NAND}(x_9, x_5)$	3.72	$x_{11} = \text{XNOR}(x_7, x_{10})$	3.76	$x_{11} = \text{XOR}(x_9, x_1)$	4.80	
	$x_{12} = \text{XNOR}(x_8, x_{11})$	4.76	$x_{12} = \text{XNOR}(x_5, x_8)$	2.32			
	$x_{13} = \text{NAND}(x_{10}, x_9)$	5.16	$x_{13} = \text{NAND}(x_9, x_{12})$	4.16			
	$x_{14} = \text{XNOR}(x_5, x_{13})$	6.10	$x_{14} = \text{XNOR}(x_5, x_{13})$	5.10			
	Design Compiler	$\ \mathcal{C}\  = 110.330997 \mu\text{m}^2$ $\mathbb{X}(\mathcal{C}) = 2.26 \text{ ns}$		$\ \mathcal{C}\  = 103.541397 \mu\text{m}^2$ $\mathbb{X}(\mathcal{C}) = 1.94 \text{ ns}$		$\ \mathcal{C}\  = 96.751798 \mu\text{m}^2$ $\mathbb{X}(\mathcal{C}) = 1.67 \text{ ns}$	
	Orthros			$x_4 = \text{XOR}(x_0, x_1)$	1.00	$x_4 = \text{MUXI}(x_0, x_3, x_1)$	0.92
				$x_5 = \text{NAND}(x_4, x_3)$	1.34	$x_5 = \text{MUXI}(x_2, x_0, x_4)$	1.84
			$x_6 = \text{XNOR}(x_0, x_5)$	2.28	$x_6 = \text{NOR}(x_3, x_4)$	1.36	
			$x_7 = \text{NAND}(x_2, x_6)$	2.62	$x_7 = \text{NAND}(x_2, x_1)$	0.34	
			$x_8 = \text{XNOR}(x_3, x_7)$	3.56	$x_8 = \text{MUXI}(x_0, x_7, x_6)$	2.28	
			$x_9 = \text{NAND}(x_6, x_8)$	3.90	$x_9 = \text{NAND}(x_3, x_2)$	0.34	
			$x_{10} = \text{NAND}(x_9, x_4)$	4.24	$x_{10} = \text{NAND}(x_1, x_0)$	0.34	
			$x_{11} = \text{XNOR}(x_2, x_{10})$	5.18	$x_{11} = \text{NAND3}(x_7, x_9, x_{10})$	0.89	
			$x_{12} = \text{ORN}(x_6, x_{11})$	6.31	$x_{12} = \text{NAND3}(x_2, x_0, x_4)$	1.47	
			$x_{13} = \text{XOR}(x_4, x_{12})$	7.31	$x_{13} = \text{NAND}(x_3, x_7)$	0.68	
			$x_{14} = \text{NOR}(x_8, x_{13})$	7.75	$x_{14} = \text{NAND}(x_{12}, x_{13})$	1.81	
			$x_{15} = \text{NOR}(x_{14}, x_{11})$	8.19			
			$x_{16} = \text{XNOR}(x_6, x_{15})$	9.13			
			$x_{17} = \text{ORN}(x_{16}, x_8)$	10.26			
			$x_{18} = \text{NAND}(x_{17}, x_{13})$	10.60			
			$x_{19} = \text{XOR}(x_{11}, x_{18})$	11.60			
Design Compiler			$\ \mathcal{C}\  = 132.397196 \mu\text{m}^2$ $\mathbb{X}(\mathcal{C}) = 4.32 \text{ ns}$		$\ \mathcal{C}\  = 79.777798 \mu\text{m}^2$ $\mathbb{X}(\mathcal{C}) = 0.80 \text{ ns}$		

extended version. *J. Cryptogr. Eng.*, 7(2):149–165, 2017. URL: <https://doi.org/10.1007/s13389-017-0156-7>, doi:10.1007/S13389-017-0156-7.

- for evaluation, implementation, and generation of s-boxes. *IACR Trans. Symmetric Cryptol.*, 2019(1):330–394, 2019. URL: <https://doi.org/10.13154/tosc.v2019.i1.330-394>, doi:10.13154/TOSC.V2019.I1.330-394.
- [BHS90] Robert K. Brayton, Gary D. Hachtel, and Alberto L. Sangiovanni-Vincentelli. Multilevel logic synthesis. *Proc. IEEE*, 78(2):264–300, 1990. doi:10.1109/5.52213.
- [BIL<sup>+</sup>21] Subhadeep Banik, Takanori Isobe, Fukang Liu, Kazuhiko Minematsu, and Kosei Sakamoto. Orthros: A low-latency PRF. *IACR Trans. Symmetric Cryptol.*, 2021(1):37–77, 2021. doi:10.46586/tosc.v2021.i1.37-77.
- [BJK<sup>+</sup>16] Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. The SKINNY family of block ciphers and its low-latency variant MANTIS. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 123–153. Springer, 2016. doi:10.1007/978-3-662-53008-5\_5.
- [BKL<sup>+</sup>07] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: an ultra-lightweight block cipher. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, volume 4727 of *Lecture Notes in Computer Science*, pages 450–466. Springer, 2007. doi:10.1007/978-3-540-74735-2\_31.
- [BKL16] Christof Beierle, Thorsten Kranz, and Gregor Leander. Lightweight multiplication in  $\text{gf}(2^n)$  with applications to MDS matrices. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 625–653. Springer, 2016. doi:10.1007/978-3-662-53018-4\_23.
- [BM10] Robert K. Brayton and Alan Mishchenko. ABC: an academic industrial-strength verification tool. In Tayssir Touili, Byron Cook, and Paul B. Jackson, editors, *Computer Aided Verification, 22nd International Conference, CAV 2010, Edinburgh, UK, July 15-19, 2010. Proceedings*, volume 6174 of *Lecture Notes in Computer Science*, pages 24–40. Springer, 2010. doi:10.1007/978-3-642-14295-6\_5.
- [BMP13] Joan Boyar, Philip Matthews, and René Peralta. Logic minimization techniques with applications to cryptology. *J. Cryptol.*, 26(2):280–312, 2013. URL: <https://doi.org/10.1007/s00145-012-9124-7>, doi:10.1007/S00145-012-9124-7.
- [BPP<sup>+</sup>17] Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Yosuke Todo. GIFT: A small present - towards reaching the limit of lightweight encryption. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, volume 10529 of *Lecture Notes in Computer Science*, pages 321–345. Springer, 2017. doi:10.1007/978-3-319-66787-4\_16.

- [Can05] David Canright. A very compact s-box for AES. In Josyula R. Rao and Berk Sunar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*, volume 3659 of *Lecture Notes in Computer Science*, pages 441–455. Springer, 2005. doi:[10.1007/11545262\\_32](https://doi.org/10.1007/11545262_32).
- [CDL15] Anne Canteaut, Sébastien Duval, and Gaëtan Leurent. Construction of lightweight s-boxes using feistel and MISTY structures. In Orr Dunkelman and Liam Keliher, editors, *Selected Areas in Cryptography - SAC 2015 - 22nd International Conference, Sackville, NB, Canada, August 12-14, 2015, Revised Selected Papers*, volume 9566 of *Lecture Notes in Computer Science*, pages 373–393. Springer, 2015. doi:[10.1007/978-3-319-31301-6\\_22](https://doi.org/10.1007/978-3-319-31301-6_22).
- [CDL<sup>+</sup>20] Anne Canteaut, Sébastien Duval, Gaëtan Leurent, María Naya-Plasencia, Léo Perrin, Thomas Pornin, and André Schrottenloher. Saturnin: a suite of lightweight symmetric algorithms for post-quantum security. *IACR Trans. Symmetric Cryptol.*, 2020(S1):160–207, 2020. doi:[10.13154/tosc.v2020.iS1.160-207](https://doi.org/10.13154/tosc.v2020.iS1.160-207).
- [Cou07] Nicolas T. Courtois. How fast can be algebraic attacks on block ciphers? In Eli Biham, Helena Handschuh, Stefan Lucks, and Vincent Rijmen, editors, *Symmetric Cryptography, 07.01. - 12.01.2007*, volume 07021 of *Dagstuhl Seminar Proceedings*. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2007. URL: <http://dr ops.dagstuhl.de/opus/volltexte/2007/1013>.
- [DGV93] Joan Daemen, René Govaerts, and Joos Vandewalle. A new approach to block cipher design. In Ross J. Anderson, editor, *Fast Software Encryption, Cambridge Security Workshop, Cambridge, UK, December 9-11, 1993, Proceedings*, volume 809 of *Lecture Notes in Computer Science*, pages 18–32. Springer, 1993. doi:[10.1007/3-540-58108-1\\_2](https://doi.org/10.1007/3-540-58108-1_2).
- [DL18] Sébastien Duval and Gaëtan Leurent. MDS matrices with lightweight circuits. *IACR Trans. Symmetric Cryptol.*, 2018(2):48–78, 2018. URL: <https://doi.org/10.13154/tosc.v2018.i2.48-78>, doi:[10.13154/TOSC.V2018.I2.48-78](https://doi.org/10.13154/TOSC.V2018.I2.48-78).
- [FS10] Carsten Fuhs and Peter Schneider-Kamp. Synthesizing shortest linear straight-line programs over GF(2) using SAT. In Ofer Strichman and Stefan Szeider, editors, *Theory and Applications of Satisfiability Testing - SAT 2010, 13th International Conference, SAT 2010, Edinburgh, UK, July 11-14, 2010. Proceedings*, volume 6175 of *Lecture Notes in Computer Science*, pages 71–84. Springer, 2010. doi:[10.1007/978-3-642-14186-7\\_8](https://doi.org/10.1007/978-3-642-14186-7_8).
- [GJN<sup>+</sup>16] Jian Guo, Jérémy Jean, Ivica Nikolic, Kexin Qiao, Yu Sasaki, and Siang Meng Sim. Invariant subspace attack against midori64 and the resistance criteria for s-box designs. *IACR Trans. Symmetric Cryptol.*, 2016(1):33–56, 2016. URL: <https://doi.org/10.13154/tosc.v2016.i1.33-56>, doi:[10.13154/TOSC.V2016.I1.33-56](https://doi.org/10.13154/TOSC.V2016.I1.33-56).
- [GLWL16] Zhiyuan Guo, Renzhang Liu, Wenling Wu, and Dongdai Lin. Direct construction of lightweight rotational-xor MDS diffusion layers. *IACR Cryptol. ePrint Arch.*, page 1036, 2016. URL: <http://eprint.iacr.org/2016/1036>.
- [JNP14] Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. Joltik v1. *CAESAR competition*, 2014.

- [JPST17a] Jérémy Jean, Thomas Peyrin, Siang Meng Sim, and Jade Tourteaux. Optimizing implementations of lightweight building blocks. *IACR Trans. Symmetric Cryptol.*, 2017(4):130–168, 2017. URL: <https://doi.org/10.13154/tosc.v2017.i4.130-168>, doi:10.13154/TOSC.V2017.I4.130-168.
- [JPST17b] Jérémy Jean, Thomas Peyrin, Siang Meng Sim, and Jade Tourteaux. Optimizing implementations of lightweight building blocks. *IACR Trans. Symmetric Cryptol.*, 2017(4):130–168, 2017. URL: <https://doi.org/10.13154/tosc.v2017.i4.130-168>, doi:10.13154/TOSC.V2017.I4.130-168.
- [KLPR10] Lars R. Knudsen, Gregor Leander, Axel Poschmann, and Matthew J. B. Robshaw. Printcipher: A block cipher for ic-printing. In Stefan Mangard and François-Xavier Standaert, editors, *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, volume 6225 of *Lecture Notes in Computer Science*, pages 16–32. Springer, 2010. doi:10.1007/978-3-642-15031-9\_2.
- [KLSW17] Thorsten Kranz, Gregor Leander, Ko Stoffelen, and Friedrich Wiemer. Shorter linear straight-line programs for MDS matrices. *IACR Trans. Symmetric Cryptol.*, 2017(4):188–211, 2017. URL: <https://doi.org/10.13154/tosc.v2017.i4.188-211>, doi:10.13154/TOSC.V2017.I4.188-211.
- [LSL<sup>+</sup>19] Shun Li, Siwei Sun, Chaoyun Li, Zihao Wei, and Lei Hu. Constructing low-latency involutory MDS matrices with lightweight circuits. *IACR Trans. Symmetric Cryptol.*, 2019(1):84–117, 2019. URL: <https://doi.org/10.13154/tosc.v2019.i1.84-117>, doi:10.13154/TOSC.V2019.I1.84-117.
- [LW14] Yongqiang Li and Mingsheng Wang. Constructing s-boxes for lightweight cryptography with feistel structure. In Lejla Batina and Matthew Robshaw, editors, *Cryptographic Hardware and Embedded Systems - CHES 2014 - 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings*, volume 8731 of *Lecture Notes in Computer Science*, pages 127–146. Springer, 2014. doi:10.1007/978-3-662-44709-3\_8.
- [LW16] Yongqiang Li and Mingsheng Wang. On the construction of lightweight circulant involutory MDS matrices. In Thomas Peyrin, editor, *Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers*, volume 9783 of *Lecture Notes in Computer Science*, pages 121–139. Springer, 2016. doi:10.1007/978-3-662-52993-5\_7.
- [LW17] Chaoyun Li and Qingju Wang. Design of lightweight linear diffusion layers from near-mds matrices. *IACR Trans. Symmetric Cryptol.*, 2017(1):129–155, 2017. URL: <https://doi.org/10.13154/tosc.v2017.i1.129-155>, doi:10.13154/TOSC.V2017.I1.129-155.
- [Max19] Alexander Maximov. AES mixcolumn with 92 XOR gates. *IACR Cryptol. ePrint Arch.*, page 833, 2019. URL: <https://eprint.iacr.org/2019/833>.
- [McC56] Edward J. McCluskey. Minimization of boolean functions. *The Bell System Technical Journal*, 35(6):1417–1444, 1956. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/j.1538-7305.1956.tb03835.x>, doi:10.1002/j.1538-7305.1956.tb03835.x.
- [ME19] Alexander Maximov and Patrik Ekdhahl. New circuit minimization techniques for smaller and faster AES sboxes. *IACR Trans. Cryptogr. Hardw. Embed.*

- Syst.*, 2019(4):91–125, 2019. URL: <https://doi.org/10.13154/tches.v2019.i4.91-125>, doi:10.13154/TCHES.V2019.I4.91-125.
- [MPL<sup>+</sup>11] Amir Moradi, Axel Poschmann, San Ling, Christof Paar, and Huaxiong Wang. Pushing the limits: A very compact and a threshold implementation of AES. In Kenneth G. Paterson, editor, *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *Lecture Notes in Computer Science*, pages 69–88. Springer, 2011. doi:10.1007/978-3-642-20465-4\_6.
- [RTA18] Arash Reyhani-Masoleh, Mostafa M. I. Taha, and Doaa Ashmawy. Smashing the implementation records of AES s-box. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(2):298–336, 2018. URL: <https://doi.org/10.13154/tches.v2018.i2.298-336>, doi:10.13154/TCHES.V2018.I2.298-336.
- [SKOP15] Siang Meng Sim, Khoongming Khoo, Frédérique E. Oggier, and Thomas Peyrin. Lightweight MDS involution matrices. In Gregor Leander, editor, *Fast Software Encryption - 22nd International Workshop, FSE 2015, Istanbul, Turkey, March 8-11, 2015, Revised Selected Papers*, volume 9054 of *Lecture Notes in Computer Science*, pages 471–493. Springer, 2015. doi:10.1007/978-3-662-48116-5\_23.
- [SPGQ06] François-Xavier Standaert, Gilles Piret, Neil Gershenfeld, and Jean-Jacques Quisquater. SEA: A scalable encryption algorithm for small embedded applications. In Josep Domingo-Ferrer, Joachim Posegga, and Daniel Schreckling, editors, *Smart Card Research and Advanced Applications, 7th IFIP WG 8.8/11.2 International Conference, CARDIS 2006, Tarragona, Spain, April 19-21, 2006, Proceedings*, volume 3928 of *Lecture Notes in Computer Science*, pages 222–236. Springer, 2006. doi:10.1007/11733447\_16.
- [SS16] Sumanta Sarkar and Habeeb Syed. Lightweight diffusion layer: Importance of toeplitz matrices. *IACR Trans. Symmetric Cryptol.*, 2016(1):95–113, 2016. URL: <https://doi.org/10.13154/tosc.v2016.i1.95-113>, doi:10.13154/TOSC.V2016.I1.95-113.
- [Sto16] Ko Stoffelen. Optimizing s-box implementations for several criteria using SAT solvers. In Thomas Peyrin, editor, *Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers*, volume 9783 of *Lecture Notes in Computer Science*, pages 140–160. Springer, 2016. doi:10.1007/978-3-662-52993-5\_8.
- [TP20] Quan Quan Tan and Thomas Peyrin. Improved heuristics for short linear programs. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2020(1):203–230, 2020. URL: <https://doi.org/10.13154/tches.v2020.i1.203-230>, doi:10.13154/TCHES.V2020.I1.203-230.
- [WZ11] Wenling Wu and Lei Zhang. Lblock: A lightweight block cipher. In Javier López and Gene Tsudik, editors, *Applied Cryptography and Network Security - 9th International Conference, ACNS 2011, Nerja, Spain, June 7-10, 2011. Proceedings*, volume 6715 of *Lecture Notes in Computer Science*, pages 327–344, 2011. doi:10.1007/978-3-642-21554-4\_19.
- [ZBL<sup>+</sup>15] Wentao Zhang, Zhenzhen Bao, Dongdai Lin, Vincent Rijmen, Bohan Yang, and Ingrid Verbauwhede. RECTANGLE: a bit-slice lightweight block cipher suitable for multiple platforms. *Sci. China Inf. Sci.*, 58(12):1–15, 2015. doi:10.1007/s11432-015-5459-7.

- [ZJB06] Junlan Zhou, Zhengrong Ji, and Rajive L. Bagrodia. TWINE: A hybrid emulation testbed for wireless networks and applications. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 23-29 April 2006, Barcelona, Catalunya, Spain*. IEEE, 2006. doi:[10.1109/INFOCOM.2006.183](https://doi.org/10.1109/INFOCOM.2006.183).
- [ZWS18] Lijing Zhou, Licheng Wang, and Yiru Sun. On efficient constructions of lightweight MDS matrices. *IACR Trans. Symmetric Cryptol.*, 2018(1):180–200, 2018. URL: <https://doi.org/10.13154/tosc.v2018.i1.180-200>, doi:[10.13154/TOSC.V2018.I1.180-200](https://doi.org/10.13154/TOSC.V2018.I1.180-200).

## A Gates in Common CMOS Technology Libraries

Table 8: Frequently-used logic gates in common CMOS technology libraries.

Gate	Domain $\rightarrow$ Range	Functionality (Logic formulas)
NOT	$\mathbb{F}_2 \rightarrow \mathbb{F}_2$	$x_0 \mapsto \neg x_0$
AND	$\mathbb{F}_2^2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto x_0 \wedge x_1$
NAND	$\mathbb{F}_2^2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto \neg(x_0 \wedge x_1)$
NANDN	$\mathbb{F}_2^2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto \neg(\neg x_0 \wedge x_1)$
OR	$\mathbb{F}_2^2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto x_0 \vee x_1$
NOR	$\mathbb{F}_2^2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto \neg(x_0 \vee x_1)$
NORN	$\mathbb{F}_2^2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto \neg(\neg x_0 \vee x_1)$
XOR	$\mathbb{F}_2^2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto x_0 \oplus x_1$
XNOR	$\mathbb{F}_2^2 \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto \neg(x_0 \oplus x_1)$
AND3	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0 \wedge x_1 \wedge x_2$
NAND3	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto \neg(x_0 \wedge x_1 \wedge x_2)$
NANDN3	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto \neg(\neg x_0 \wedge x_1 \wedge x_2)$
OR3	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0 \vee x_1 \vee x_2$
NOR3	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto \neg(x_0 \vee x_1 \vee x_2)$
NORN3	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto \neg(\neg x_0 \vee x_1 \vee x_2)$
XOR3	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto x_0 \oplus x_1 \oplus x_2$
XNOR3	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto \neg(x_0 \oplus x_1 \oplus x_2)$
MUX	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto (x_0 \wedge x_1) \vee (\neg x_0 \wedge x_2)$
MUXI	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto \neg((x_0 \wedge x_1) \vee (\neg x_0 \wedge x_2))$
AO21	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto (x_0 \wedge x_1) \vee x_2$
AOI21	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto \neg((x_0 \wedge x_1) \vee x_2)$
OAI21	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto (x_0 \vee x_1) \wedge x_2$
OAI21	$\mathbb{F}_2^3 \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2 \mapsto \neg((x_0 \vee x_1) \wedge x_2)$
ANDN	$\mathbb{F}_2^n \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto \neg x_0 \wedge x_1$
ORN	$\mathbb{F}_2^n \rightarrow \mathbb{F}_2$	$x_0, x_1 \mapsto \neg x_0 \vee x_1$
MAOI1	$\mathbb{F}_2^n \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2, x_3 \mapsto \neg((x_0 \wedge x_1) \vee (\neg(x_2 \vee x_3)))$
MOAI1	$\mathbb{F}_2^n \rightarrow \mathbb{F}_2$	$x_0, x_1, x_2, x_3 \mapsto \neg((x_0 \vee x_1) \wedge (\neg(x_2 \wedge x_3)))$



## B Common CMOS Technology Libraries

Table 9: Used libraries, NAND and NOR are always 1 in all cases, so we omit them in the table. Scaled with  $\times 100$

Gate	UMC 180nm	TSMC 65nm	TSMC 28nm	SMIC 130nm	SMIC 65nm	Nangate 45nm	Nangate 15nm	STD 350nm	STM 65nm
NOT	0.67	0.50	0.67	0.67	0.75	0.67	0.75	0.67	0.50
AND	1.33	1.50	1.33	1.33	1.50	1.33	1.50	1.33	1.50
NAND	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
NANDN	1.67	1.50	1.33	1.33	1.50	N/A	N/A	N/A	1.50
OR	1.33	1.50	1.33	1.33	1.50	1.33	1.50	1.33	1.50
NOR	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
NORN	1.67	1.50	1.33	1.33	1.50	N/A	N/A	N/A	1.50
XOR	2.67	2.50	3.00	2.33	2.25	2.00	2.25	2.33	2.00
XNOR	2.00	2.50	3.00	2.33	2.25	2.00	2.25	2.33	2.00
AND3	2.33	2.00	1.67	1.67	1.75	1.67	2.00	1.67	2.00
NAND3	1.33	1.50	1.33	1.33	1.25	1.33	1.50	1.33	1.50
NANDN3		2.00	1.67	1.67	1.75	N/A	N/A	N/A	
OR3	2.33	2.00	1.67	2.00	1.75	1.67	2.00	1.67	2.00
NOR3	1.33	1.50	1.33	1.33	1.50	1.33	1.50	1.33	1.50
NORN3		2.00	1.67	1.67	N/A	N/A	N/A	N/A	
XOR3	4.67	5.50	4.33	5.67	4.75	N/A	N/A	4.00	
XNOR3	4.67	5.50	4.67	5.67	4.75	N/A	N/A	4.00	
MUX		3.00	2.33	2.67	2.75	2.33	3.25	2.33	
MUXI		2.50	2.33	2.33	2.50	N/A	N/A	2.67	
AO21		2.00	1.67	1.67	2.00	N/A	N/A	1.67	
AOI21		1.50	1.33	1.67	1.50	1.33	1.50	1.33	
OA21		2.00	1.67	2.00	1.75	N/A	N/A	1.67	
OAI21		1.50	1.33	1.67	1.50	1.33	1.50	1.33	

## C Applications and Results

Table 10: Synthesis results with the UMC 180nm library

S-box	UMC 180nm		
	LIGHTER	PEIGEN	Ours
3-way	✗	10.34 GE	9.67 GE
ctc2	✗	9.66 GE	9.32 GE
PRINTcipher	✗	9.00 GE	9.00 GE
SEA	✗	9.00 GE	9.00 GE
Joltik	12.99 GE	12.99 GE	12.99 GE
Joltik <sup>-1</sup>	12.99 GE	12.99 GE	12.99 GE
RECTANGLE	18.33 GE	18.33 GE	20.34 GE
RECTANGLE <sup>-1</sup>	18.33 GE	18.33 GE	20.34 GE
SKINNY	13.32 GE	13.32 GE	13.32 GE
SKINNY <sup>-1</sup>	13.32 GE	13.32 GE	13.32 GE
TWINE	21.66 GE	21.66 GE	No res
PRESENT	21.32 GE	21.32 GE	No res
LBlock s <sub>0</sub>	16.33 GE	16.33 GE	16.33 GE
LBlock s <sub>1</sub>	16.66 GE	16.66 GE	17.00 GE
LBlock s <sub>3</sub>	16.66 GE	16.66 GE	16.67 GE
GIFT	16.00 GE	16.00 GE	19.34 GE
Midori s <sub>0</sub>	20.66 GE	20.66 GE	19.99 GE
Saturnin s <sub>0</sub>	18.00 GE	18.00 GE	21.99 GE
Orthros	24.33 GE	No res	No res

Table 11: Synthesis results with the Nangate library

S-box	Nangate 45nm			Nangate 15nm		
	LIGHTER	PEIGEN	Ours	LIGHTER	PEIGEN	Ours
3-way	✗	10.34 GE	9.67 GE	✗	11.25 GE	10.50 GE
ctc2	✗	9.00 GE	8.33 GE	✗	9.75 GE	9.00 GE
PRINTcipher	✗	9.00 GE	8.99 GE	✗	9.75 GE	9.75 GE
SEA	✗	9.00 GE	8.33 GE	✗	9.75 GE	9.75 GE
Joltik	12.00 GE	12.00 GE	12.00 GE	13.00 GE	13.00 GE	13.00 GE
Joltik <sup>-1</sup>	12.00 GE	12.00 GE	12.00 GE	13.00 GE	13.00 GE	13.00 GE
RECTANGLE	18.00 GE	18.00 GE	No res	19.75 GE	19.75 GE	22.00 GE
RECTANGLE <sup>-1</sup>	18.00 GE	18.00 GE	No res	19.75 GE	19.75 GE	19.75 GE
SKINNY	12.00 GE	12.00 GE	12.00 GE	13.00 GE	13.00 GE	13.00 GE
SKINNY <sup>-1</sup>	12.00 GE	12.00 GE	12.00 GE	13.00 GE	13.00 GE	13.00 GE
TWINE	21.33 GE	21.33 GE	No res	23.50 GE	23.50 GE	No res
PRESENT	20.33 GE	20.33 GE	No res	22.25 GE	22.25 GE	No res
LBlock s <sub>0</sub>	16.00 GE	16.00 GE	15.33 GE	17.50 GE	17.50 GE	16.75 GE
LBlock s <sub>1</sub>	16.00 GE	16.00 GE	15.33 GE	17.50 GE	17.50 GE	16.75 GE
LBlock s <sub>3</sub>	16.00 GE	16.00 GE	15.33 GE	17.50 GE	17.50 GE	17.50 GE
GIFT	16.00 GE	16.00 GE	17.99 GE	17.50 GE	17.50 GE	17.50 GE
Midori s <sub>0</sub>	20.00 GE	20.00 GE	13.65 GE	22.00 GE	22.00 GE	14.50 GE
Saturnin s <sub>0</sub>	18.00 GE	18.00 GE	17.65 GE	19.50 GE	19.50 GE	20.00 GE
Orthros	23.67 GE	24.00 GE	15.98 GE	25.75 GE	No res	17.75 GE

Table 12: Synthesis results with the STD90/MDL90 350nm library

S-box	STD90/MDL90 350nm		
	LIGHTER	PEIGEN	Ours
3-way	$\times$	11.33 GE	10.66 GE
ctc2	$\times$	9.99 GE	8.99 GE
PRINTcipher	$\times$	9.99 GE	9.32 GE
SEA	$\times$	9.99 GE	8.99 GE
Joltik	13.32 GE	13.32 GE	13.32 GE
Joltik <sup>-1</sup>	13.32 GE	13.32 GE	13.32 GE
RECTANGLE	20.31 GE	20.31 GE	18.98 GE
RECTANGLE <sup>-1</sup>	20.31 GE	20.31 GE	17.98 GE
SKINNY	13.32 GE	13.32 GE	13.32 GE
SKINNY <sup>-1</sup>	13.32 GE	13.32 GE	13.32 GE
TWINE	23.97 GE	23.97 GE	No res
PRESENT	22.64 GE	22.64 GE	No res
LBlock s <sub>0</sub>	17.98 GE	17.98 GE	15.66 GE
LBlock s <sub>1</sub>	17.98 GE	17.98 GE	15.99 GE
LBlock s <sub>3</sub>	17.98 GE	17.98 GE	15.99 GE
GIFT	17.98 GE	17.98 GE	18.65 GE
Midori s <sub>0</sub>	22.64 GE	22.64 GE	13.65 GE
Saturnin s <sub>0</sub>	19.98 GE	19.98 GE	19.66 GE
Orthros	26.31 GE	No res	15.65 GE

Table 13: Synthesis results with the STM 65nm library

S-box	STM 65nm		
	LIGHTER	PEIGEN	Ours
3-way	$\times$	10.00 GE	9.50 GE
ctc2	$\times$	9.00 GE	9.00 GE
PRINTcipher	$\times$	9.00 GE	9.00 GE
SEA	$\times$	9.00 GE	9.00 GE
Joltik	12.00 GE	12.00 GE	12.00 GE
Joltik <sup>-1</sup>	12.00 GE	12.00 GE	12.00 GE
RECTANGLE	19.00 GE	18.50 GE	No res
RECTANGLE <sup>-1</sup>	19.00 GE	18.50 GE	No res
SKINNY	12.00 GE	12.00 GE	12.00 GE
SKINNY <sup>-1</sup>	12.00 GE	12.00 GE	12.00 GE
TWINE	21.50 GE	No res	No res
PRESENT	21.00 GE	21.00 GE	No res
LBlock s <sub>0</sub>	16.50 GE	16.00 GE	16.00 GE
LBlock s <sub>1</sub>	16.00 GE	16.00 GE	16.00 GE
LBlock s <sub>3</sub>	17.00 GE	16.00 GE	16.00 GE
GIFT	16.50 GE	16.00 GE	16.00 GE
Midori s <sub>0</sub>	20.50 GE	20.00 GE	No res
Saturnin s <sub>0</sub>	18.50 GE	18.00 GE	No res
Orthros	Error	No res	No res

Table 14: Optimized circuits implemented with the UMC 180nm library

S-box	Implementation in UMC 180nm			
3-way	$x_3 = \text{NORN}(x_1, x_2)$ $x_7 = \text{NOR}(x_0, x_6)$	$x_4 = \text{XNOR}(x_0, x_3)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{NOR}(x_1, x_4)$	$x_6 = \text{XNOR}(x_2, x_5)$
ctc2	$x_3 = \text{OR}(x_1, x_0)$ $x_7 = \text{NAND3}(x_3, x_5, x_6)$	$x_4 = \text{XNOR}(x_2, x_3)$ $x_8 = \text{NAND}(x_0, x_5)$	$x_5 = \text{NAND3}(x_2, x_1, x_0)$ $x_9 = \text{NAND}(x_6, x_8)$	$x_6 = \text{OR}(x_2, x_1)$
PRINTcipher	$x_3 = \text{NOR}(x_2, x_1)$ $x_7 = \text{NAND}(x_2, x_4)$	$x_4 = \text{XNOR}(x_0, x_3)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{NAND}(x_1, x_0)$	$x_6 = \text{XNOR}(x_2, x_5)$
SEA	$x_3 = \text{NOR}(x_1, x_0)$ $x_7 = \text{NAND}(x_2, x_6)$	$x_4 = \text{XNOR}(x_2, x_3)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{NAND}(x_2, x_1)$	$x_6 = \text{XNOR}(x_0, x_5)$
Joltik	$x_4 = \text{OR}(x_3, x_2)$ $x_8 = \text{OR}(x_5, x_7)$	$x_5 = \text{XNOR}(x_0, x_4)$ $x_9 = \text{XNOR}(x_1, x_8)$	$x_6 = \text{OR}(x_2, x_1)$ $x_{10} = \text{NOR}(x_1, x_5)$	$x_7 = \text{XNOR}(x_3, x_6)$ $x_{11} = \text{XNOR}(x_2, x_{10})$
Joltik <sup>-1</sup>	$x_4 = \text{OR}(x_3, x_2)$ $x_8 = \text{OR}(x_5, x_7)$	$x_5 = \text{XNOR}(x_0, x_4)$ $x_9 = \text{XNOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{OR}(x_7, x_9)$	$x_7 = \text{XNOR}(x_1, x_6)$ $x_{11} = \text{XNOR}(x_3, x_{10})$
RECTANGLE	$x_4 = \text{NANDN}(x_3, x_1)$ $x_8 = \text{XOR3}(x_3, x_6, x_7)$ $x_{12} = \text{XNOR}(x_5, x_{11})$	$x_5 = \text{XNOR}(x_0, x_4)$ $x_9 = \text{NOR}(x_5, x_8)$	$x_6 = \text{XNOR}(x_2, x_5)$ $x_{10} = \text{XOR3}(x_1, x_6, x_9)$	$x_7 = \text{NAND}(x_1, x_5)$ $x_{11} = \text{OR}(x_8, x_{10})$
RECTANGLE <sup>-1</sup>	$x_4 = \text{OR}(x_3, x_0)$ $x_8 = \text{NAND}(x_5, x_7)$ $x_{12} = \text{XNOR}(x_5, x_{11})$	$x_5 = \text{XNOR}(x_2, x_4)$ $x_9 = \text{XOR3}(x_1, x_0, x_8)$	$x_6 = \text{NOR}(x_0, x_5)$ $x_{10} = \text{XNOR}(x_1, x_5)$	$x_7 = \text{XOR3}(x_3, x_1, x_6)$ $x_{11} = \text{NANDN}(x_9, x_7)$
SKINNY	$x_4 = \text{OR}(x_3, x_2)$ $x_8 = \text{OR}(x_1, x_5)$	$x_5 = \text{XNOR}(x_0, x_4)$ $x_9 = \text{XNOR}(x_2, x_8)$	$x_6 = \text{OR}(x_2, x_1)$ $x_{10} = \text{OR}(x_5, x_7)$	$x_7 = \text{XNOR}(x_3, x_6)$ $x_{11} = \text{XNOR}(x_1, x_{10})$
SKINNY <sup>-1</sup>	$x_4 = \text{OR}(x_3, x_2)$ $x_8 = \text{OR}(x_5, x_7)$	$x_5 = \text{XNOR}(x_0, x_4)$ $x_9 = \text{XNOR}(x_2, x_8)$	$x_6 = \text{OR}(x_3, x_5)$ $x_{10} = \text{OR}(x_7, x_9)$	$x_7 = \text{XNOR}(x_1, x_6)$ $x_{11} = \text{XNOR}(x_3, x_{10})$
LBlock s <sub>0</sub>	$x_4 = \text{OR}(x_3, x_2)$ $x_8 = \text{NAND}(x_2, x_7)$ $x_{12} = \text{OR}(x_5, x_7)$	$x_5 = \text{XNOR}(x_0, x_4)$ $x_9 = \text{XNOR}(x_5, x_8)$ $x_{13} = \text{NAND}(x_{10}, x_{12})$	$x_6 = \text{XNOR}(x_1, x_5)$ $x_{10} = \text{NANDN}(x_3, x_9)$	$x_7 = \text{XNOR}(x_3, x_6)$ $x_{11} = \text{XNOR}(x_2, x_{10})$
LBlock s <sub>1</sub>	$x_4 = \text{OR}(x_2, x_0)$ $x_8 = \text{NOR}(x_3, x_2)$ $x_{12} = \text{NOR}(x_7, x_9)$	$x_5 = \text{NAND}(x_2, x_1)$ $x_9 = \text{XNOR3}(x_1, x_0, x_8)$ $x_{13} = \text{XNOR}(x_3, x_{12})$	$x_6 = \text{NAND}(x_4, x_5)$ $x_{10} = \text{NOR}(x_3, x_6)$	$x_7 = \text{XNOR}(x_3, x_6)$ $x_{11} = \text{XNOR}(x_2, x_{10})$
LBlock s <sub>3</sub>	$x_4 = \text{XNOR}(x_1, x_0)$ $x_8 = \text{NOR}(x_3, x_6)$ $x_{12} = \text{NOR}(x_7, x_{11})$	$x_5 = \text{NAND}(x_2, x_4)$ $x_9 = \text{XNOR}(x_2, x_8)$ $x_{13} = \text{XNOR}(x_3, x_{12})$	$x_6 = \text{XNOR}(x_1, x_5)$ $x_{10} = \text{NANDN}(x_3, x_2)$	$x_7 = \text{XNOR}(x_3, x_6)$ $x_{11} = \text{XNOR}(x_4, x_{10})$
GIFT	$x_4 = \text{NOR}(x_1, x_0)$ $x_8 = \text{NAND}(x_3, x_7)$ $x_{12} = \text{XNOR}(x_5, x_7)$	$x_5 = \text{XOR3}(x_3, x_2, x_4)$ $x_9 = \text{XNOR}(x_0, x_8)$	$x_6 = \text{NAND}(x_2, x_0)$ $x_{10} = \text{NAND}(x_5, x_9)$	$x_7 = \text{XNOR}(x_1, x_6)$ $x_{11} = \text{XNOR3}(x_2, x_7, x_{10})$
Midori s <sub>0</sub>	$x_4 = \text{OR}(x_3, x_0)$ $x_8 = \text{OR}(x_3, x_2)$ $x_{12} = \text{NAND3}(x_1, x_5, x_{11})$	$x_5 = \text{NAND}(x_3, x_2)$ $x_9 = \text{NAND}(x_2, x_0)$ $x_{13} = \text{XOR3}(x_1, x_4, x_{12})$	$x_6 = \text{NAND}(x_1, x_4)$ $x_{10} = \text{XNOR3}(x_4, x_8, x_9)$ $x_{14} = \text{NAND}(x_{11}, x_{12})$	$x_7 = \text{AND}(x_5, x_6)$ $x_{11} = \text{NAND3}(x_0, x_5, x_8)$
Saturnin s <sub>0</sub>	$x_4 = \text{NOR3}(x_3, x_1, x_0)$ $x_8 = \text{NOR3}(x_3, x_4, x_7)$ $x_{12} = \text{XNOR3}(x_0, x_4, x_{11})$	$x_5 = \text{NOR}(x_2, x_0)$ $x_9 = \text{NOR3}(x_2, x_6, x_8)$ $x_{13} = \text{NAND3}(x_0, x_{10}, x_{11})$	$x_6 = \text{NOR}(x_1, x_4)$ $x_{10} = \text{XNOR}(x_3, x_9)$ $x_{14} = \text{XNOR3}(x_2, x_8, x_{13})$	$x_7 = \text{XNOR}(x_5, x_6)$ $x_{11} = \text{NOR3}(x_5, x_8, x_9)$

Table 15: Optimized circuits implemented with the TSMC 65nm library

S-box	Implementation in TSMC 65nm			
3-way	$x_3 = \text{XNOR}(x_2, x_0)$ $x_7 = \text{NOR}(x_1, x_3)$	$x_4 = \text{MUXI}(x_1, x_3, x_0)$ $x_8 = \text{AOI21}(x_2, x_1, x_7)$	$x_5 = \text{NAND}(x_2, x_4)$	$x_6 = \text{XOR}(x_1, x_5)$
ctc2	$x_3 = \text{NOR}(x_1, x_0)$ $x_7 = \text{NOR}(x_0, x_4)$	$x_4 = \text{XOR}(x_2, x_3)$ $x_8 = \text{AOI21}(x_2, x_1, x_7)$	$x_5 = \text{NAND}(x_2, x_0)$	$x_6 = \text{XOR}(x_1, x_5)$
PRINTcipher	$x_3 = \text{NOR}(x_2, x_1)$ $x_7 = \text{NAND}(x_2, x_4)$	$x_4 = \text{XNOR}(x_0, x_3)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{NAND}(x_1, x_0)$	$x_6 = \text{XNOR}(x_2, x_5)$
SEA	$x_3 = \text{MUXI}(x_2, x_0, x_1)$ $x_7 = \text{XNOR}(x_0, x_6)$	$x_4 = \text{NOR}(x_1, x_0)$ $x_8 = \text{NOT}(x_3)$	$x_5 = \text{XNOR}(x_2, x_4)$	$x_6 = \text{NAND}(x_2, x_1)$
Joltik	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_2, x_1)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_3, x_8)$	$x_6 = \text{NOR}(x_1, x_5)$ $x_{10} = \text{NOR}(x_5, x_9)$	$x_7 = \text{XNOR}(x_2, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
Joltik <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{NOT}(x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XNOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
RECTANGLE	$x_4 = \text{XNOR}(x_2, x_1)$ $x_8 = \text{MUXI}(x_6, x_7, x_4)$	$x_5 = \text{MUXI}(x_0, x_4, x_2)$ $x_9 = \text{XOR}(x_6, x_7)$	$x_6 = \text{XNOR}(x_3, x_5)$ $x_{10} = \text{MUX}(x_4, x_7, x_9)$	$x_7 = \text{MUXI}(x_1, x_0, x_3)$ $x_{11} = \text{XOR}(x_1, x_9)$
RECTANGLE <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_0)$ $x_8 = \text{XNOR}(x_3, x_7)$ $x_{12} = \text{MUXI}(x_8, x_{10}, x_5)$	$x_5 = \text{XNOR}(x_2, x_4)$ $x_9 = \text{NOT}(x_2)$	$x_6 = \text{XOR}(x_1, x_5)$ $x_{10} = \text{MUXI}(x_0, x_3, x_9)$	$x_7 = \text{MUXI}(x_0, x_1, x_6)$ $x_{11} = \text{MUXI}(x_5, x_7, x_{10})$
SKINNY	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_1, x_5)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_2, x_1)$ $x_{10} = \text{NOR}(x_5, x_7)$	$x_7 = \text{XOR}(x_3, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
SKINNY <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
LBlock s <sub>0</sub>	$x_4 = \text{NOT}(x_0)$ $x_8 = \text{XOR}(x_1, x_7)$	$x_5 = \text{MUXI}(x_2, x_1, x_4)$ $x_9 = \text{MUXI}(x_8, x_3, x_0)$	$x_6 = \text{XNOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_3, x_5)$	$x_7 = \text{MUXI}(x_2, x_0, x_6)$ $x_{11} = \text{XOR}(x_2, x_{10})$
LBlock s <sub>1</sub>	$x_4 = \text{NOT}(x_0)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{MUXI}(x_2, x_1, x_4)$ $x_9 = \text{MUXI}(x_8, x_3, x_4)$	$x_6 = \text{XOR}(x_3, x_5)$ $x_{10} = \text{NAND}(x_5, x_6)$	$x_7 = \text{MUXI}(x_2, x_4, x_6)$ $x_{11} = \text{XOR}(x_2, x_{10})$
LBlock s <sub>3</sub>	$x_4 = \text{NOT}(x_0)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{MUXI}(x_2, x_4, x_1)$ $x_9 = \text{MUXI}(x_8, x_3, x_4)$	$x_6 = \text{XOR}(x_3, x_5)$ $x_{10} = \text{NAND}(x_5, x_6)$	$x_7 = \text{MUXI}(x_2, x_6, x_4)$ $x_{11} = \text{XOR}(x_2, x_{10})$
GIFT	$x_4 = \text{NOR}(x_1, x_0)$ $x_8 = \text{XOR}(x_1, x_7)$ $x_{12} = \text{MUXI}(x_{11}, x_9, x_5)$	$x_5 = \text{XOR}(x_2, x_4)$ $x_9 = \text{XOR}(x_5, x_8)$	$x_6 = \text{XOR}(x_3, x_5)$ $x_{10} = \text{NAND}(x_3, x_9)$	$x_7 = \text{MUXI}(x_0, x_3, x_6)$ $x_{11} = \text{XNOR}(x_0, x_{10})$
Midori s <sub>0</sub>	$x_4 = \text{AOI21}(x_3, x_2, x_0)$ $x_8 = \text{NAND}(x_4, x_7)$ $x_{12} = \text{AOI21}(x_3, x_2, x_1)$	$x_5 = \text{NORN}(x_4, x_1)$ $x_9 = \text{MUX}(x_8, x_2, x_0)$ $x_{13} = \text{NAND}(x_7, x_{12})$	$x_6 = \text{AOI21}(x_3, x_2, x_5)$ $x_{10} = \text{NOR}(x_2, x_8)$	$x_7 = \text{OR}(x_3, x_0)$ $x_{11} = \text{MUXI}(x_1, x_{10}, x_7)$
Saturnin s <sub>0</sub>	$x_4 = \text{MUXI}(x_1, x_2, x_3)$ $x_8 = \text{NORN}(x_7, x_0)$ $x_{12} = \text{NOR}(x_2, x_{11})$	$x_5 = \text{NOR}(x_2, x_1)$ $x_9 = \text{XOR}(x_1, x_8)$ $x_{13} = \text{MUXI}(x_3, x_{10}, x_{12})$	$x_6 = \text{NOR}(x_0, x_5)$ $x_{10} = \text{MUXI}(x_3, x_4, x_0)$	$x_7 = \text{XNOR}(x_4, x_6)$ $x_{11} = \text{XOR}(x_5, x_{10})$
Orthros	$x_4 = \text{NOR}(x_1, x_0)$ $x_8 = \text{NAND}(x_2, x_1)$ $x_{12} = \text{NAND}(x_{10}, x_{11})$	$x_5 = \text{NOR}(x_3, x_4)$ $x_9 = \text{MUXI}(x_0, x_8, x_5)$ $x_{13} = \text{NAND3}(x_2, x_0, x_5)$	$x_6 = \text{NOR}(x_2, x_5)$ $x_{10} = \text{AOI21}(x_3, x_1, x_2)$ $x_{14} = \text{NAND}(x_3, x_8)$	$x_7 = \text{MUXI}(x_6, x_4, x_0)$ $x_{11} = \text{NAND}(x_1, x_0)$ $x_{15} = \text{NAND}(x_{13}, x_{14})$

Table 16: Optimized circuits implemented with the TSMC 28nm library

S-box	Implementation in TSMC 28nm			
3-way	$x_3 = \text{NANDN}(x_0, x_2)$ $x_7 = \text{MUXI}(x_3, x_1, x_6)$	$x_4 = \text{MUX}(x_0, x_2, x_3)$ $x_8 = \text{AOI21}(x_2, x_1, x_6)$	$x_5 = \text{MUXI}(x_1, x_4, x_0)$	$x_6 = \text{NOR}(x_1, x_4)$
ctc2	$x_3 = \text{NAND}(x_2, x_1)$ $x_7 = \text{AOI21}(x_2, x_1, x_6)$	$x_4 = \text{NAND}(x_2, x_0)$ $x_8 = \text{NAND}(x_6, x_7)$	$x_5 = \text{MUXI}(x_4, x_1, x_3)$ $x_9 = \text{NAND3}(x_3, x_4, x_8)$	$x_6 = \text{NAND}(x_0, x_3)$
PRINTcipher	$x_3 = \text{NOR3}(x_2, x_1, x_0)$ $x_7 = \text{MUX}(x_6, x_2, x_4)$	$x_4 = \text{AOI21}(x_2, x_1, x_3)$ $x_8 = \text{NORN}(x_6, x_3)$	$x_5 = \text{MUX}(x_0, x_1, x_4)$	$x_6 = \text{AOI21}(x_2, x_1, x_0)$
SEA	$x_3 = \text{AOI21}(x_2, x_1, x_0)$ $x_7 = \text{NAND3}(x_2, x_1, x_0)$	$x_4 = \text{NOR}(x_2, x_1)$ $x_8 = \text{NORN}(x_7, x_3)$	$x_5 = \text{MUXI}(x_3, x_4, x_2)$	$x_6 = \text{MUX}(x_2, x_0, x_1)$
Joltik	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{MUXI}(x_7, x_6, x_2)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{MUX}(x_6, x_4, x_3)$	$x_6 = \text{NOR}(x_2, x_1)$ $x_{10} = \text{NOR}(x_5, x_9)$	$x_7 = \text{NOR}(x_1, x_5)$ $x_{11} = \text{MUX}(x_{10}, x_7, x_1)$
Joltik <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{XNOR}(x_1, x_7)$ $x_{12} = \text{MUX}(x_{11}, x_5, x_3)$	$x_5 = \text{NOR}(x_3, x_0)$ $x_9 = \text{NOR}(x_6, x_8)$	$x_6 = \text{MUX}(x_4, x_5, x_0)$ $x_{10} = \text{XOR}(x_2, x_9)$	$x_7 = \text{NOR}(x_3, x_6)$ $x_{11} = \text{NOR}(x_8, x_{10})$
RECTANGLE	$x_4 = \text{MUXI}(x_1, x_0, x_3)$ $x_8 = \text{XOR3}(x_1, x_4, x_7)$	$x_5 = \text{NANDN}(x_3, x_1)$ $x_9 = \text{NANDN}(x_8, x_6)$	$x_6 = \text{MUX}(x_5, x_0, x_4)$ $x_{10} = \text{MUXI}(x_9, x_4, x_3)$	$x_7 = \text{XNOR}(x_2, x_6)$ $x_{11} = \text{MUX}(x_8, x_6, x_4)$
RECTANGLE <sup>-1</sup>	$x_4 = \text{OR}(x_3, x_0)$ $x_8 = \text{XNOR}(x_6, x_7)$	$x_5 = \text{XOR3}(x_2, x_1, x_4)$ $x_9 = \text{MUX}(x_5, x_7, x_6)$	$x_6 = \text{MUXI}(x_0, x_1, x_2)$ $x_{10} = \text{MUX}(x_8, x_5, x_6)$	$x_7 = \text{MUX}(x_0, x_3, x_1)$ $x_{11} = \text{XOR}(x_2, x_{10})$
SKINNY	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_1, x_5)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{MUX}(x_8, x_6, x_2)$	$x_6 = \text{NOR}(x_2, x_1)$ $x_{10} = \text{NOR}(x_5, x_7)$	$x_7 = \text{MUX}(x_6, x_4, x_3)$ $x_{11} = \text{MUX}(x_{10}, x_8, x_1)$
SKINNY <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{XOR}(x_1, x_7)$ $x_{12} = \text{MUX}(x_{11}, x_5, x_3)$	$x_5 = \text{NOR}(x_3, x_0)$ $x_9 = \text{NOR}(x_6, x_8)$	$x_6 = \text{MUX}(x_4, x_5, x_0)$ $x_{10} = \text{XOR}(x_2, x_9)$	$x_7 = \text{NOR}(x_3, x_6)$ $x_{11} = \text{NOR}(x_8, x_{10})$
LBlock s <sub>0</sub>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{MUX}(x_2, x_5, x_0)$	$x_5 = \text{NOT}(x_1)$ $x_9 = \text{NOR}(x_3, x_8)$	$x_6 = \text{XOR3}(x_0, x_4, x_5)$ $x_{10} = \text{MUX}(x_9, x_4, x_2)$	$x_7 = \text{MUXI}(x_6, x_3, x_0)$ $x_{11} = \text{MUX}(x_3, x_8, x_9)$
LBlock s <sub>1</sub>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{MUX}(x_2, x_1, x_5)$	$x_5 = \text{NOT}(x_0)$ $x_9 = \text{NOR}(x_3, x_8)$	$x_6 = \text{XOR3}(x_1, x_4, x_5)$ $x_{10} = \text{MUXI}(x_9, x_4, x_2)$	$x_7 = \text{MUXI}(x_6, x_3, x_5)$ $x_{11} = \text{MUX}(x_3, x_8, x_9)$
LBlock s <sub>3</sub>	$x_4 = \text{NANDN}(x_3, x_2)$ $x_8 = \text{MUX}(x_2, x_6, x_1)$ $x_{12} = \text{NAND}(x_9, x_{11})$	$x_5 = \text{XOR3}(x_1, x_0, x_4)$ $x_9 = \text{OR}(x_3, x_8)$	$x_6 = \text{NOT}(x_0)$ $x_{10} = \text{MUXI}(x_9, x_2, x_4)$	$x_7 = \text{MUXI}(x_5, x_3, x_6)$ $x_{11} = \text{NAND}(x_3, x_8)$
GIFT	$x_4 = \text{NOR}(x_1, x_0)$ $x_8 = \text{XNOR3}(x_0, x_5, x_7)$	$x_5 = \text{XOR3}(x_3, x_2, x_4)$ $x_9 = \text{MUX}(x_3, x_7, x_0)$	$x_6 = \text{NORN}(x_0, x_2)$ $x_{10} = \text{MUXI}(x_9, x_8, x_4)$	$x_7 = \text{XOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_2, x_{10})$
Midori s <sub>0</sub>	$x_4 = \text{NAND}(x_3, x_2)$ $x_8 = \text{MUX}(x_1, x_7, x_4)$ $x_{12} = \text{NAND}(x_1, x_{10})$	$x_5 = \text{AOI21}(x_3, x_2, x_4)$ $x_9 = \text{NANDN}(x_5, x_0)$ $x_{13} = \text{NANDN}(x_7, x_{12})$	$x_6 = \text{MUX}(x_5, x_0, x_2)$ $x_{10} = \text{NAND3}(x_1, x_4, x_9)$	$x_7 = \text{NOR}(x_3, x_0)$ $x_{11} = \text{NAND}(x_9, x_{10})$
Saturnin s <sub>0</sub>	$x_4 = \text{NANDN}(x_3, x_0)$ $x_8 = \text{MUX}(x_2, x_4, x_7)$ $x_{12} = \text{MUX}(x_{11}, x_1, x_5)$	$x_5 = \text{XOR}(x_1, x_4)$ $x_9 = \text{MUX}(x_0, x_6, x_2)$	$x_6 = \text{MUXI}(x_2, x_3, x_5)$ $x_{10} = \text{XOR3}(x_3, x_7, x_9)$	$x_7 = \text{NOR}(x_3, x_5)$ $x_{11} = \text{NANDN}(x_0, x_{10})$
Orthros	$x_4 = \text{MUXI}(x_0, x_2, x_3)$ $x_8 = \text{AOI21}(x_3, x_2, x_1)$ $x_{12} = \text{OR}(x_0, x_8)$	$x_5 = \text{MUXI}(x_1, x_4, x_0)$ $x_9 = \text{NOR}(x_2, x_0)$ $x_{13} = \text{NAND}(x_{11}, x_{12})$	$x_6 = \text{NAND3}(x_3, x_2, x_1)$ $x_{10} = \text{NOR}(x_8, x_9)$	$x_7 = \text{MUX}(x_4, x_3, x_6)$ $x_{11} = \text{AOI21}(x_3, x_2, x_4)$

Table 17: Optimized circuits implemented with the SMIC 130nm library

S-box	Implementation in SMIC 130nm			
3-way	$x_3 = \text{NORN}(x_1, x_2)$ $x_7 = \text{NAND}(x_2, x_4)$	$x_4 = \text{XNOR}(x_0, x_3)$ $x_8 = \text{XOR}(x_1, x_7)$	$x_5 = \text{NOR}(x_1, x_4)$	$x_6 = \text{XNOR}(x_2, x_5)$
ctc2	$x_3 = \text{NOR}(x_1, x_0)$ $x_7 = \text{NOR}(x_0, x_4)$	$x_4 = \text{XOR}(x_2, x_3)$ $x_8 = \text{AOI21}(x_2, x_1, x_7)$	$x_5 = \text{NAND}(x_2, x_0)$	$x_6 = \text{XOR}(x_1, x_5)$
PRINTcipher	$x_3 = \text{NOR}(x_2, x_1)$ $x_7 = \text{NAND}(x_2, x_4)$	$x_4 = \text{XNOR}(x_0, x_3)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{NAND}(x_1, x_0)$	$x_6 = \text{XNOR}(x_2, x_5)$
SEA	$x_3 = \text{MUX}(x_2, x_0, x_1)$ $x_7 = \text{XNOR}(x_0, x_6)$	$x_4 = \text{NOR}(x_1, x_0)$	$x_5 = \text{XNOR}(x_2, x_4)$	$x_6 = \text{NAND}(x_2, x_1)$
Joltik	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_2, x_1)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_3, x_8)$	$x_6 = \text{NOR}(x_1, x_5)$ $x_{10} = \text{NOR}(x_5, x_9)$	$x_7 = \text{XNOR}(x_2, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
Joltik <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XNOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
RECTANGLE	$x_4 = \text{NOR}(x_2, x_1)$ $x_8 = \text{MUXI}(x_6, x_7, x_4)$	$x_5 = \text{MUXI}(x_0, x_4, x_2)$ $x_9 = \text{XOR}(x_6, x_7)$	$x_6 = \text{XNOR}(x_3, x_5)$ $x_{10} = \text{MUX}(x_4, x_7, x_9)$	$x_7 = \text{MUXI}(x_1, x_0, x_3)$ $x_{11} = \text{XOR}(x_1, x_9)$
RECTANGLE <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_0)$ $x_8 = \text{MUXI}(x_7, x_0, x_5)$	$x_5 = \text{XNOR3}(x_2, x_1, x_4)$ $x_9 = \text{XNOR}(x_6, x_8)$	$x_6 = \text{MUXI}(x_0, x_3, x_2)$ $x_{10} = \text{MUXI}(x_7, x_5, x_6)$	$x_7 = \text{XNOR}(x_1, x_6)$ $x_{11} = \text{XNOR}(x_0, x_{10})$
SKINNY	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_1, x_5)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_2, x_1)$ $x_{10} = \text{NOR}(x_5, x_7)$	$x_7 = \text{XOR}(x_3, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
SKINNY <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
LBlock s <sub>0</sub>	$x_4 = \text{NOT}(x_0)$ $x_8 = \text{XOR}(x_1, x_7)$	$x_5 = \text{MUXI}(x_2, x_1, x_4)$ $x_9 = \text{MUXI}(x_8, x_3, x_0)$	$x_6 = \text{XNOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_3, x_5)$	$x_7 = \text{MUXI}(x_2, x_0, x_6)$ $x_{11} = \text{XOR}(x_2, x_{10})$
LBlock s <sub>1</sub>	$x_4 = \text{NOT}(x_0)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{MUXI}(x_2, x_1, x_4)$ $x_9 = \text{MUXI}(x_8, x_3, x_4)$	$x_6 = \text{XOR}(x_3, x_5)$ $x_{10} = \text{NAND}(x_5, x_6)$	$x_7 = \text{MUXI}(x_2, x_4, x_6)$ $x_{11} = \text{XOR}(x_2, x_{10})$
LBlock s <sub>3</sub>	$x_4 = \text{NOT}(x_0)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{MUXI}(x_2, x_4, x_1)$ $x_9 = \text{MUXI}(x_8, x_3, x_4)$	$x_6 = \text{XOR}(x_3, x_5)$ $x_{10} = \text{NAND}(x_5, x_6)$	$x_7 = \text{MUXI}(x_2, x_6, x_4)$ $x_{11} = \text{XOR}(x_2, x_{10})$
GIFT	$x_4 = \text{NOR}(x_1, x_0)$ $x_8 = \text{XOR}(x_1, x_7)$ $x_{12} = \text{MUXI}(x_{11}, x_9, x_5)$	$x_5 = \text{XOR}(x_2, x_4)$ $x_9 = \text{XOR}(x_5, x_8)$	$x_6 = \text{XOR}(x_3, x_5)$ $x_{10} = \text{NAND}(x_3, x_9)$	$x_7 = \text{MUXI}(x_0, x_3, x_6)$ $x_{11} = \text{XNOR}(x_0, x_{10})$
Midori s <sub>0</sub>	$x_4 = \text{XNOR}(x_3, x_0)$ $x_8 = \text{NANDN}(x_7, x_6)$ $x_{12} = \text{NOR}(x_1, x_{10})$	$x_5 = \text{MUX}(x_4, x_2, x_0)$ $x_9 = \text{NOR}(x_2, x_4)$ $x_{13} = \text{AOI21}(x_3, x_2, x_{12})$	$x_6 = \text{AO21}(x_3, x_2, x_1)$ $x_{10} = \text{NOR3}(x_1, x_7, x_9)$	$x_7 = \text{NOR}(x_3, x_0)$ $x_{11} = \text{NOR}(x_9, x_{10})$
Saturin s <sub>0</sub>	$x_4 = \text{XNOR}(x_1, x_0)$ $x_8 = \text{MUXI}(x_6, x_3, x_7)$ $x_{12} = \text{MUX}(x_{11}, x_4, x_1)$	$x_5 = \text{MUXI}(x_2, x_0, x_4)$ $x_9 = \text{MUXI}(x_1, x_2, x_3)$	$x_6 = \text{MUX}(x_3, x_2, x_5)$ $x_{10} = \text{NANDN}(x_0, x_5)$	$x_7 = \text{AOI21}(x_3, x_1, x_2)$ $x_{11} = \text{XOR}(x_9, x_{10})$
Orthros	$x_4 = \text{MUXI}(x_0, x_3, x_1)$ $x_8 = \text{MUXI}(x_0, x_7, x_6)$ $x_{12} = \text{NAND3}(x_2, x_0, x_4)$	$x_5 = \text{MUXI}(x_2, x_0, x_4)$ $x_9 = \text{NAND}(x_3, x_2)$ $x_{13} = \text{NAND}(x_3, x_7)$	$x_6 = \text{NOR}(x_3, x_4)$ $x_{10} = \text{NAND}(x_1, x_0)$ $x_{14} = \text{NAND}(x_{12}, x_{13})$	$x_7 = \text{NAND}(x_2, x_1)$ $x_{11} = \text{NAND3}(x_7, x_9, x_{10})$

Table 18: Optimized circuits implemented with the SMIC 65nm library

S-box	Implementation in SMIC 65nm			
3-way	$x_3 = \text{NORN}(x_1, x_2)$ $x_7 = \text{NAND}(x_2, x_4)$	$x_4 = \text{XNOR}(x_0, x_3)$ $x_8 = \text{XOR}(x_1, x_7)$	$x_5 = \text{NOR}(x_1, x_4)$	$x_6 = \text{XNOR}(x_2, x_5)$
ctc2	$x_3 = \text{NOR}(x_1, x_0)$ $x_7 = \text{NOR}(x_0, x_4)$	$x_4 = \text{XOR}(x_2, x_3)$ $x_8 = \text{AOI21}(x_2, x_1, x_7)$	$x_5 = \text{NAND}(x_2, x_0)$	$x_6 = \text{XOR}(x_1, x_5)$
PRINTcipher	$x_3 = \text{NOR}(x_2, x_1)$ $x_7 = \text{NAND}(x_2, x_4)$	$x_4 = \text{XNOR}(x_0, x_3)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{NAND}(x_1, x_0)$	$x_6 = \text{XNOR}(x_2, x_5)$
SEA	$x_3 = \text{MUX}(x_2, x_0, x_1)$ $x_7 = \text{XNOR}(x_0, x_6)$	$x_4 = \text{NOR}(x_1, x_0)$	$x_5 = \text{XNOR}(x_2, x_4)$	$x_6 = \text{NAND}(x_2, x_1)$
Joltik	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_2, x_1)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_3, x_8)$	$x_6 = \text{NOR}(x_1, x_5)$ $x_{10} = \text{NOR}(x_5, x_9)$	$x_7 = \text{XNOR}(x_2, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
Joltik <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XNOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
RECTANGLE	$x_4 = \text{NORN}(x_1, x_3)$ $x_8 = \text{XNOR}(x_1, x_7)$ $x_{12} = \text{XOR}(x_5, x_{11})$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{MUXI}(x_5, x_8, x_6)$	$x_6 = \text{XNOR}(x_2, x_5)$ $x_{10} = \text{XOR}(x_3, x_9)$	$x_7 = \text{MUXI}(x_5, x_2, x_3)$ $x_{11} = \text{NOR}(x_8, x_{10})$
RECTANGLE <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_0)$ $x_8 = \text{MUXI}(x_7, x_0, x_5)$	$x_5 = \text{XNOR3}(x_2, x_1, x_4)$ $x_9 = \text{XNOR}(x_6, x_8)$	$x_6 = \text{MUXI}(x_0, x_3, x_2)$ $x_{10} = \text{MUXI}(x_7, x_5, x_6)$	$x_7 = \text{XNOR}(x_1, x_6)$ $x_{11} = \text{XNOR}(x_0, x_{10})$
SKINNY	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_1, x_5)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_2, x_1)$ $x_{10} = \text{NOR}(x_5, x_7)$	$x_7 = \text{XOR}(x_3, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
SKINNY <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
LBlock s <sub>0</sub>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{MUXI}(x_2, x_5, x_0)$	$x_5 = \text{XNOR}(x_1, x_4)$ $x_9 = \text{XOR}(x_3, x_8)$	$x_6 = \text{XOR}(x_0, x_5)$ $x_{10} = \text{NAND}(x_7, x_9)$	$x_7 = \text{MUXI}(x_6, x_3, x_0)$ $x_{11} = \text{XNOR}(x_2, x_{10})$
LBlock s <sub>1</sub>	$x_4 = \text{NOT}(x_0)$ $x_8 = \text{XOR}(x_2, x_7)$	$x_5 = \text{MUXI}(x_2, x_1, x_4)$ $x_9 = \text{MUXI}(x_2, x_4, x_6)$	$x_6 = \text{XOR}(x_3, x_5)$ $x_{10} = \text{XNOR}(x_1, x_9)$	$x_7 = \text{NAND}(x_5, x_6)$ $x_{11} = \text{MUXI}(x_{10}, x_3, x_4)$
LBlock s <sub>3</sub>	$x_4 = \text{NOT}(x_0)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{MUXI}(x_2, x_4, x_1)$ $x_9 = \text{MUXI}(x_8, x_3, x_4)$	$x_6 = \text{XOR}(x_3, x_5)$ $x_{10} = \text{NAND}(x_5, x_6)$	$x_7 = \text{MUXI}(x_2, x_6, x_4)$ $x_{11} = \text{XOR}(x_2, x_{10})$
GIFT	$x_4 = \text{NOR}(x_1, x_0)$ $x_8 = \text{XOR}(x_1, x_7)$ $x_{12} = \text{MUXI}(x_{11}, x_9, x_5)$	$x_5 = \text{XOR}(x_2, x_4)$ $x_9 = \text{XOR}(x_5, x_8)$	$x_6 = \text{XOR}(x_3, x_5)$ $x_{10} = \text{NAND}(x_3, x_9)$	$x_7 = \text{MUXI}(x_0, x_3, x_6)$ $x_{11} = \text{XNOR}(x_0, x_{10})$
Midori s <sub>0</sub>	$x_4 = \text{NAND}(x_3, x_2)$ $x_8 = \text{NAND3}(x_1, x_4, x_7)$ $x_{12} = \text{NAND}(x_1, x_8)$	$x_5 = \text{OAI21}(x_3, x_2, x_4)$ $x_9 = \text{NAND}(x_7, x_8)$ $x_{13} = \text{NANDN}(x_{10}, x_{12})$	$x_6 = \text{MUX}(x_5, x_2, x_0)$ $x_{10} = \text{NOR}(x_3, x_0)$	$x_7 = \text{NAND}(x_0, x_5)$ $x_{11} = \text{MUX}(x_1, x_{10}, x_4)$
Saturnin s <sub>0</sub>	$x_4 = \text{MUXI}(x_1, x_2, x_3)$ $x_8 = \text{MUXI}(x_3, x_4, x_0)$ $x_{12} = \text{NORN}(x_7, x_0)$	$x_5 = \text{NOR}(x_2, x_1)$ $x_9 = \text{XOR}(x_5, x_8)$ $x_{13} = \text{XOR}(x_1, x_{12})$	$x_6 = \text{NOR}(x_0, x_5)$ $x_{10} = \text{NOR}(x_2, x_9)$	$x_7 = \text{XNOR}(x_4, x_6)$ $x_{11} = \text{MUXI}(x_3, x_8, x_{10})$
Orthros	$x_4 = \text{NOR}(x_1, x_0)$ $x_8 = \text{NAND}(x_2, x_1)$ $x_{12} = \text{NAND3}(x_8, x_{10}, x_{11})$	$x_5 = \text{NOR}(x_3, x_4)$ $x_9 = \text{MUXI}(x_0, x_8, x_5)$ $x_{13} = \text{NAND3}(x_2, x_0, x_5)$	$x_6 = \text{NOR}(x_2, x_5)$ $x_{10} = \text{NAND}(x_3, x_2)$ $x_{14} = \text{NAND}(x_3, x_8)$	$x_7 = \text{MUXI}(x_6, x_4, x_0)$ $x_{11} = \text{NAND}(x_1, x_0)$ $x_{15} = \text{NAND}(x_{13}, x_{14})$



Table 19: Optimized circuits implemented with the Nangate 45nm library

S-box	Implementation in Nangate 45nm			
3-way	$x_3 = \text{NOT}(x_2)$ $x_7 = \text{XOR}(x_1, x_6)$	$x_4 = \text{NAND}(x_1, x_3)$ $x_8 = \text{NAND}(x_0, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NAND}(x_2, x_5)$
ctc2	$x_3 = \text{NOR}(x_1, x_0)$ $x_7 = \text{NOR}(x_0, x_4)$	$x_4 = \text{XOR}(x_2, x_3)$ $x_8 = \text{AOI21}(x_2, x_1, x_7)$	$x_5 = \text{NAND}(x_2, x_0)$	$x_6 = \text{XOR}(x_1, x_5)$
PRINTcipher	$x_3 = \text{NOR}(x_2, x_1)$ $x_7 = \text{XNOR}(x_0, x_3)$	$x_4 = \text{AOI21}(x_2, x_1, x_3)$	$x_5 = \text{MUX}(x_0, x_4, x_2)$	$x_6 = \text{MUX}(x_0, x_1, x_4)$
SEA	$x_3 = \text{MUX}(x_2, x_0, x_1)$ $x_7 = \text{XNOR}(x_0, x_6)$	$x_4 = \text{NOR}(x_1, x_0)$	$x_5 = \text{XNOR}(x_2, x_4)$	$x_6 = \text{NAND}(x_2, x_1)$
Joltik	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_2, x_1)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_3, x_8)$	$x_6 = \text{NOR}(x_1, x_5)$ $x_{10} = \text{NOR}(x_5, x_9)$	$x_7 = \text{XNOR}(x_2, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
Joltik <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XNOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
SKINNY	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_1, x_5)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_2, x_1)$ $x_{10} = \text{NOR}(x_5, x_7)$	$x_7 = \text{XOR}(x_3, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
SKINNY <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
LBlock s <sub>0</sub>	$x_4 = \text{NOT}(x_1)$ $x_8 = \text{XOR}(x_4, x_7)$ $x_{12} = \text{NOT}(x_{11})$	$x_5 = \text{MUX}(x_2, x_4, x_0)$ $x_9 = \text{NOR}(x_3, x_5)$	$x_6 = \text{XNOR}(x_3, x_5)$ $x_{10} = \text{XOR}(x_2, x_9)$	$x_7 = \text{MUX}(x_2, x_0, x_6)$ $x_{11} = \text{MUX}(x_8, x_3, x_0)$
LBlock s <sub>1</sub>	$x_4 = \text{NOT}(x_0)$ $x_8 = \text{XNOR}(x_2, x_7)$ $x_{12} = \text{NOT}(x_{11})$	$x_5 = \text{MUX}(x_2, x_1, x_4)$ $x_9 = \text{MUX}(x_2, x_4, x_6)$	$x_6 = \text{XNOR}(x_3, x_5)$ $x_{10} = \text{XOR}(x_1, x_9)$	$x_7 = \text{NOR}(x_3, x_5)$ $x_{11} = \text{MUX}(x_{10}, x_3, x_4)$
LBlock s <sub>3</sub>	$x_4 = \text{NOT}(x_0)$ $x_8 = \text{XNOR}(x_2, x_7)$ $x_{12} = \text{NOT}(x_{11})$	$x_5 = \text{MUX}(x_2, x_4, x_1)$ $x_9 = \text{MUX}(x_2, x_6, x_4)$	$x_6 = \text{XNOR}(x_3, x_5)$ $x_{10} = \text{XOR}(x_1, x_9)$	$x_7 = \text{NOR}(x_3, x_5)$ $x_{11} = \text{MUX}(x_{10}, x_3, x_4)$
GIFT	$x_4 = \text{XNOR}(x_2, x_1)$ $x_8 = \text{XOR}(x_3, x_7)$ $x_{12} = \text{XOR}(x_7, x_{11})$	$x_5 = \text{MUX}(x_0, x_4, x_1)$ $x_9 = \text{XNOR}(x_0, x_5)$	$x_6 = \text{MUX}(x_3, x_5, x_0)$ $x_{10} = \text{XOR}(x_8, x_9)$	$x_7 = \text{MUX}(x_0, x_2, x_4)$ $x_{11} = \text{NAND}(x_6, x_{10})$
Midori s <sub>0</sub>	$x_4 = \text{XNOR}(x_3, x_0)$ $x_8 = \text{OR}(x_6, x_7)$ $x_{12} = \text{NOR}(x_1, x_{10})$	$x_5 = \text{MUX}(x_4, x_2, x_0)$ $x_9 = \text{NOR}(x_2, x_4)$ $x_{13} = \text{AOI21}(x_3, x_2, x_{12})$	$x_6 = \text{AOI21}(x_3, x_2, x_1)$ $x_{10} = \text{NOR3}(x_1, x_7, x_9)$	$x_7 = \text{NOR}(x_3, x_0)$ $x_{11} = \text{NOR}(x_9, x_{10})$
Saturnin s <sub>0</sub>	$x_4 = \text{MUX}(x_1, x_2, x_3)$ $x_8 = \text{XOR}(x_1, x_7)$ $x_{12} = \text{NOR3}(x_2, x_4, x_{10})$	$x_5 = \text{NOR3}(x_2, x_1, x_0)$ $x_9 = \text{MUX}(x_0, x_1, x_8)$ $x_{13} = \text{XNOR}(x_3, x_{12})$	$x_6 = \text{NOR}(x_0, x_5)$ $x_{10} = \text{NOR3}(x_3, x_5, x_8)$	$x_7 = \text{XOR}(x_4, x_6)$ $x_{11} = \text{XOR}(x_2, x_{10})$
Orthros	$x_4 = \text{AOI21}(x_3, x_2, x_1)$ $x_8 = \text{NOR3}(x_2, x_0, x_4)$ $x_{12} = \text{XOR}(x_{10}, x_{11})$	$x_5 = \text{OR}(x_3, x_4)$ $x_9 = \text{NOR}(x_4, x_8)$ $x_{13} = \text{MUX}(x_3, x_6, x_{10})$	$x_6 = \text{AND}(x_2, x_1)$ $x_{10} = \text{AOI21}(x_3, x_2, x_0)$ $x_{14} = \text{NOT}(x_{13})$	$x_7 = \text{MUX}(x_0, x_6, x_5)$ $x_{11} = \text{NOR}(x_2, x_8)$

Table 20: Optimized circuits implemented with the Nangate 15nm library

S-box	Implementation in Nangate 15nm			
3-way	$x_3 = \text{NOT}(x_2)$ $x_7 = \text{XOR}(x_1, x_6)$	$x_4 = \text{NAND}(x_1, x_3)$ $x_8 = \text{NAND}(x_0, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NAND}(x_2, x_5)$
ctc2	$x_3 = \text{NOR}(x_1, x_0)$ $x_7 = \text{NOR}(x_0, x_4)$	$x_4 = \text{XOR}(x_2, x_3)$ $x_8 = \text{AOI21}(x_2, x_1, x_7)$	$x_5 = \text{NAND}(x_2, x_0)$	$x_6 = \text{XOR}(x_1, x_5)$
PRINTcipher	$x_3 = \text{NOR}(x_2, x_1)$ $x_7 = \text{NAND}(x_2, x_4)$	$x_4 = \text{XNOR}(x_0, x_3)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{NAND}(x_1, x_0)$	$x_6 = \text{XNOR}(x_2, x_5)$
SEA	$x_3 = \text{MUX}(x_2, x_0, x_1)$ $x_7 = \text{XNOR}(x_0, x_6)$	$x_4 = \text{NOR}(x_1, x_0)$	$x_5 = \text{XNOR}(x_2, x_4)$	$x_6 = \text{NAND}(x_2, x_1)$
Joltik	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_2, x_1)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_3, x_8)$	$x_6 = \text{NOR}(x_1, x_5)$ $x_{10} = \text{NOR}(x_5, x_9)$	$x_7 = \text{XNOR}(x_2, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
Joltik <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XNOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
RECTANGLE	$x_4 = \text{XNOR}(x_2, x_0)$ $x_8 = \text{XNOR}(x_5, x_7)$ $x_{12} = \text{XOR}(x_1, x_{11})$	$x_5 = \text{XNOR}(x_3, x_4)$ $x_9 = \text{MUX}(x_2, x_1, x_8)$	$x_6 = \text{MUX}(x_1, x_5, x_4)$ $x_{10} = \text{XOR}(x_4, x_9)$	$x_7 = \text{NAND}(x_1, x_0)$ $x_{11} = \text{MUX}(x_8, x_6, x_2)$
RECTANGLE <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_0)$ $x_8 = \text{XOR}(x_1, x_7)$ $x_{12} = \text{XOR}(x_9, x_{11})$	$x_5 = \text{XOR}(x_2, x_4)$ $x_9 = \text{XOR}(x_1, x_0)$	$x_6 = \text{XNOR}(x_1, x_5)$ $x_{10} = \text{MUX}(x_8, x_9, x_5)$	$x_7 = \text{MUX}(x_0, x_3, x_2)$ $x_{11} = \text{NAND}(x_5, x_8)$
SKINNY	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_1, x_5)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_2, x_1)$ $x_{10} = \text{NOR}(x_5, x_7)$	$x_7 = \text{XOR}(x_3, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
SKINNY <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
LBlock $s_0$	$x_4 = \text{XNOR}(x_1, x_0)$ $x_8 = \text{AOI21}(x_3, x_2, x_7)$ $x_{12} = \text{NAND}(x_9, x_{11})$	$x_5 = \text{NOR}(x_3, x_2)$ $x_9 = \text{XNOR}(x_1, x_8)$ $x_{13} = \text{XNOR}(x_2, x_{12})$	$x_6 = \text{XOR}(x_4, x_5)$ $x_{10} = \text{NOR}(x_6, x_9)$	$x_7 = \text{NOR}(x_2, x_6)$ $x_{11} = \text{XNOR}(x_3, x_{10})$
LBlock $s_1$	$x_4 = \text{XNOR}(x_1, x_0)$ $x_8 = \text{AOI21}(x_3, x_2, x_7)$ $x_{12} = \text{NAND}(x_9, x_{11})$	$x_5 = \text{NOR}(x_3, x_2)$ $x_9 = \text{XOR}(x_1, x_8)$ $x_{13} = \text{XOR}(x_2, x_{12})$	$x_6 = \text{XOR}(x_4, x_5)$ $x_{10} = \text{NOR}(x_6, x_9)$	$x_7 = \text{NOR}(x_2, x_6)$ $x_{11} = \text{XNOR}(x_3, x_{10})$
LBlock $s_3$	$x_4 = \text{NOT}(x_3)$ $x_8 = \text{XOR}(x_1, x_4)$ $x_{12} = \text{OR}(x_6, x_8)$	$x_5 = \text{NAND}(x_2, x_4)$ $x_9 = \text{MUX}(x_2, x_6, x_8)$ $x_{13} = \text{NAND}(x_{10}, x_{12})$	$x_6 = \text{XOR}(x_0, x_5)$ $x_{10} = \text{NAND}(x_4, x_9)$	$x_7 = \text{XOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_2, x_{10})$
GIFT	$x_4 = \text{NOR}(x_1, x_0)$ $x_8 = \text{XNOR}(x_1, x_7)$ $x_{12} = \text{NAND}(x_9, x_{11})$	$x_5 = \text{XOR}(x_2, x_4)$ $x_9 = \text{XNOR}(x_6, x_8)$ $x_{13} = \text{XOR}(x_5, x_{12})$	$x_6 = \text{XOR}(x_3, x_5)$ $x_{10} = \text{NAND}(x_3, x_8)$	$x_7 = \text{NAND}(x_2, x_0)$ $x_{11} = \text{XNOR}(x_0, x_{10})$
Midori $s_0$	$x_4 = \text{AOI21}(x_3, x_2, x_1)$ $x_8 = \text{NOR}(x_0, x_5)$ $x_{12} = \text{NOR3}(x_1, x_5, x_{11})$	$x_5 = \text{NOR}(x_3, x_0)$ $x_9 = \text{NOR3}(x_2, x_7, x_8)$ $x_{13} = \text{NOR}(x_{11}, x_{12})$	$x_6 = \text{OR}(x_4, x_5)$ $x_{10} = \text{NOR}(x_8, x_9)$ $x_{14} = \text{NOR}(x_1, x_{12})$	$x_7 = \text{NOR}(x_3, x_5)$ $x_{11} = \text{NOR}(x_2, x_9)$ $x_{15} = \text{AOI21}(x_3, x_2, x_{14})$
Saturnin $s_0$	$x_4 = \text{NOR3}(x_3, x_1, x_0)$ $x_8 = \text{NOR3}(x_3, x_4, x_7)$ $x_{12} = \text{MUX}(x_2, x_{11}, x_8)$	$x_5 = \text{NOR}(x_2, x_0)$ $x_9 = \text{NOR3}(x_2, x_6, x_8)$ $x_{13} = \text{NOR3}(x_5, x_8, x_9)$	$x_6 = \text{NOR}(x_1, x_4)$ $x_{10} = \text{XNOR}(x_3, x_9)$ $x_{14} = \text{NOR}(x_0, x_4)$	$x_7 = \text{XNOR}(x_5, x_6)$ $x_{11} = \text{NAND}(x_0, x_{10})$ $x_{15} = \text{XOR}(x_{13}, x_{14})$
Orthros	$x_4 = \text{AND}(x_2, x_0)$ $x_8 = \text{NOR}(x_2, x_0)$ $x_{12} = \text{NOR3}(x_4, x_{10}, x_{11})$	$x_5 = \text{NAND}(x_2, x_1)$ $x_9 = \text{NOR}(x_7, x_8)$ $x_{13} = \text{NOR3}(x_3, x_0, x_7)$	$x_6 = \text{MUX}(x_3, x_5, x_4)$ $x_{10} = \text{NOR3}(x_3, x_2, x_8)$ $x_{14} = \text{AND}(x_0, x_5)$	$x_7 = \text{AOI21}(x_3, x_2, x_1)$ $x_{11} = \text{AND}(x_7, x_8)$ $x_{15} = \text{NOR}(x_{13}, x_{14})$

Table 21: Optimized circuits implemented with the STD90/MDL90 350nm library

S-box	Implementation in STD90/MDL90 350nm			
3-way	$x_3 = \text{XNOR}(x_2, x_0)$ $x_7 = \text{NOR}(x_1, x_3)$	$x_4 = \text{MUXI}(x_1, x_3, x_0)$ $x_8 = \text{AOI21}(x_2, x_1, x_7)$	$x_5 = \text{NAND}(x_2, x_4)$	$x_6 = \text{XOR}(x_1, x_5)$
ctc2	$x_3 = \text{NOR}(x_1, x_0)$ $x_7 = \text{NOR}(x_0, x_4)$	$x_4 = \text{XOR}(x_2, x_3)$ $x_8 = \text{AOI21}(x_2, x_1, x_7)$	$x_5 = \text{NAND}(x_2, x_0)$	$x_6 = \text{XOR}(x_1, x_5)$
PRINTcipher	$x_3 = \text{NOR}(x_2, x_1)$ $x_7 = \text{XNOR}(x_0, x_3)$	$x_4 = \text{AOI21}(x_2, x_1, x_3)$	$x_5 = \text{MUX}(x_0, x_4, x_2)$	$x_6 = \text{MUX}(x_0, x_1, x_4)$
SEA	$x_3 = \text{MUX}(x_2, x_0, x_1)$ $x_7 = \text{XNOR}(x_0, x_6)$	$x_4 = \text{NOR}(x_1, x_0)$	$x_5 = \text{XNOR}(x_2, x_4)$	$x_6 = \text{NAND}(x_2, x_1)$
Joltik	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_2, x_1)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_3, x_8)$	$x_6 = \text{NOR}(x_1, x_5)$ $x_{10} = \text{NOR}(x_5, x_9)$	$x_7 = \text{XNOR}(x_2, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
Joltik <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XNOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
RECTANGLE	$x_4 = \text{XOR}(x_2, x_1)$ $x_8 = \text{XNOR3}(x_1, x_6, x_7)$	$x_5 = \text{MUX}(x_3, x_2, x_4)$ $x_9 = \text{MUX}(x_8, x_7, x_4)$	$x_6 = \text{XNOR}(x_0, x_5)$ $x_{10} = \text{NAND}(x_4, x_8)$	$x_7 = \text{MUX}(x_1, x_0, x_3)$ $x_{11} = \text{XOR}(x_7, x_{10})$
RECTANGLE <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_0)$ $x_8 = \text{XOR}(x_3, x_7)$ $x_{12} = \text{MUX}(x_8, x_{10}, x_5)$	$x_5 = \text{XOR}(x_2, x_4)$ $x_9 = \text{NOT}(x_2)$	$x_6 = \text{XNOR}(x_1, x_5)$ $x_{10} = \text{MUX}(x_0, x_3, x_9)$	$x_7 = \text{MUX}(x_0, x_1, x_6)$ $x_{11} = \text{MUX}(x_5, x_{10}, x_7)$
SKINNY	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_1, x_5)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_2, x_1)$ $x_{10} = \text{NOR}(x_5, x_7)$	$x_7 = \text{XOR}(x_3, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
SKINNY <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
LBlock <sub>s0</sub>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{AOI21}(x_3, x_2, x_7)$	$x_5 = \text{XNOR3}(x_1, x_0, x_4)$ $x_9 = \text{XNOR}(x_1, x_8)$	$x_6 = \text{MUXI}(x_5, x_3, x_0)$ $x_{10} = \text{NAND}(x_6, x_9)$	$x_7 = \text{NOR}(x_2, x_5)$ $x_{11} = \text{XNOR}(x_2, x_{10})$
LBlock <sub>s1</sub>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{MUX}(x_2, x_6, x_5)$	$x_5 = \text{XNOR3}(x_1, x_0, x_4)$ $x_9 = \text{XOR}(x_1, x_8)$	$x_6 = \text{NOT}(x_3)$ $x_{10} = \text{NAND}(x_6, x_9)$	$x_7 = \text{MUX}(x_5, x_6, x_0)$ $x_{11} = \text{XOR}(x_2, x_{10})$
LBlock <sub>s3</sub>	$x_4 = \text{NOT}(x_3)$ $x_8 = \text{MUX}(x_2, x_6, x_4)$	$x_5 = \text{NAND}(x_2, x_4)$ $x_9 = \text{XOR}(x_1, x_8)$	$x_6 = \text{XOR3}(x_1, x_0, x_5)$ $x_{10} = \text{NAND}(x_4, x_9)$	$x_7 = \text{MUX}(x_6, x_4, x_0)$ $x_{11} = \text{XOR}(x_2, x_{10})$
GIFT	$x_4 = \text{NAND}(x_2, x_0)$ $x_8 = \text{MUX}(x_5, x_0, x_3)$ $x_{12} = \text{XOR}(x_8, x_{11})$	$x_5 = \text{XNOR}(x_1, x_4)$ $x_9 = \text{XOR3}(x_2, x_7, x_8)$	$x_6 = \text{NAND}(x_3, x_5)$ $x_{10} = \text{XNOR}(x_5, x_9)$	$x_7 = \text{XNOR}(x_0, x_6)$ $x_{11} = \text{NOR}(x_7, x_{10})$
Midori <sub>s0</sub>	$x_4 = \text{AOI21}(x_3, x_2, x_1)$ $x_8 = \text{NOR}(x_0, x_5)$ $x_{12} = \text{NOR3}(x_1, x_5, x_{11})$	$x_5 = \text{NOR}(x_3, x_0)$ $x_9 = \text{NOR3}(x_2, x_7, x_8)$ $x_{13} = \text{NOR}(x_{11}, x_{12})$	$x_6 = \text{OR}(x_4, x_5)$ $x_{10} = \text{NOR}(x_8, x_9)$ $x_{14} = \text{NOR}(x_1, x_{12})$	$x_7 = \text{NOR}(x_3, x_5)$ $x_{11} = \text{NOR}(x_2, x_9)$ $x_{15} = \text{AOI21}(x_3, x_2, x_{14})$
Saturnin <sub>s0</sub>	$x_4 = \text{XNOR}(x_1, x_0)$ $x_8 = \text{MUXI}(x_6, x_3, x_7)$ $x_{12} = \text{MUX}(x_{11}, x_4, x_1)$	$x_5 = \text{MUXI}(x_2, x_0, x_4)$ $x_9 = \text{NOR}(x_0, x_7)$	$x_6 = \text{MUX}(x_3, x_2, x_5)$ $x_{10} = \text{NAND}(x_1, x_8)$	$x_7 = \text{AOI21}(x_3, x_1, x_2)$ $x_{11} = \text{XNOR3}(x_3, x_9, x_{10})$
Orthros	$x_4 = \text{AOI21}(x_3, x_2, x_1)$ $x_8 = \text{NAND}(x_2, x_1)$ $x_{12} = \text{NOR3}(x_3, x_0, x_4)$	$x_5 = \text{NOR3}(x_2, x_0, x_4)$ $x_9 = \text{MUX}(x_3, x_8, x_7)$ $x_{13} = \text{AND}(x_0, x_8)$	$x_6 = \text{NOR}(x_4, x_5)$ $x_{10} = \text{NOR}(x_2, x_5)$ $x_{14} = \text{NOR}(x_{12}, x_{13})$	$x_7 = \text{OA21}(x_3, x_2, x_0)$ $x_{11} = \text{XNOR}(x_7, x_{10})$

Table 22: Optimized circuits implemented with the STM 65nm library

S-box	Implementation in STM 65nm			
3-way	$x_3 = \text{NANDN}(x_2, x_1)$ $x_7 = \text{NAND}(x_0, x_6)$	$x_4 = \text{XOR}(x_0, x_3)$ $x_8 = \text{XOR}(x_2, x_7)$	$x_5 = \text{NAND}(x_2, x_4)$	$x_6 = \text{XOR}(x_1, x_5)$
ctc2	$x_3 = \text{NOR}(x_1, x_0)$ $x_7 = \text{NAND}(x_4, x_6)$	$x_4 = \text{XOR}(x_2, x_3)$ $x_8 = \text{XNOR}(x_0, x_7)$	$x_5 = \text{NAND}(x_2, x_0)$	$x_6 = \text{XOR}(x_1, x_5)$
PRINTcipher	$x_3 = \text{NOR}(x_2, x_1)$ $x_7 = \text{NAND}(x_2, x_4)$	$x_4 = \text{XNOR}(x_0, x_3)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{NAND}(x_1, x_0)$	$x_6 = \text{XNOR}(x_2, x_5)$
SEA	$x_3 = \text{NOR}(x_1, x_0)$ $x_7 = \text{NAND}(x_2, x_6)$	$x_4 = \text{XNOR}(x_2, x_3)$ $x_8 = \text{XNOR}(x_1, x_7)$	$x_5 = \text{NAND}(x_2, x_1)$	$x_6 = \text{XNOR}(x_0, x_5)$
Joltik	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_2, x_1)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_3, x_8)$	$x_6 = \text{NOR}(x_1, x_5)$ $x_{10} = \text{NOR}(x_5, x_9)$	$x_7 = \text{XNOR}(x_2, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
Joltik <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XNOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
SKINNY	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_1, x_5)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_2, x_1)$ $x_{10} = \text{NOR}(x_5, x_7)$	$x_7 = \text{XOR}(x_3, x_6)$ $x_{11} = \text{XOR}(x_1, x_{10})$
SKINNY <sup>-1</sup>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NOR}(x_5, x_7)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_2, x_8)$	$x_6 = \text{NOR}(x_3, x_5)$ $x_{10} = \text{NOR}(x_7, x_9)$	$x_7 = \text{XOR}(x_1, x_6)$ $x_{11} = \text{XOR}(x_3, x_{10})$
LBlock <sub>s0</sub>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NAND}(x_2, x_7)$ $x_{12} = \text{NANDN}(x_7, x_5)$	$x_5 = \text{XNOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_5, x_8)$ $x_{13} = \text{NAND}(x_{10}, x_{12})$	$x_6 = \text{XOR}(x_1, x_5)$ $x_{10} = \text{NANDN}(x_3, x_9)$	$x_7 = \text{XNOR}(x_3, x_6)$ $x_{11} = \text{XNOR}(x_2, x_{10})$
LBlock <sub>s1</sub>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NAND}(x_2, x_7)$ $x_{12} = \text{NANDN}(x_7, x_5)$	$x_5 = \text{XOR}(x_0, x_4)$ $x_9 = \text{XOR}(x_5, x_8)$ $x_{13} = \text{NAND}(x_{10}, x_{12})$	$x_6 = \text{XNOR}(x_1, x_5)$ $x_{10} = \text{NANDN}(x_3, x_9)$	$x_7 = \text{XNOR}(x_3, x_6)$ $x_{11} = \text{XOR}(x_2, x_{10})$
LBlock <sub>s3</sub>	$x_4 = \text{NOR}(x_3, x_2)$ $x_8 = \text{NAND}(x_2, x_7)$ $x_{12} = \text{NANDN}(x_9, x_6)$	$x_5 = \text{XNOR}(x_1, x_4)$ $x_9 = \text{XOR}(x_5, x_8)$ $x_{13} = \text{NAND}(x_{10}, x_{12})$	$x_6 = \text{XNOR}(x_0, x_5)$ $x_{10} = \text{NANDN}(x_3, x_9)$	$x_7 = \text{XOR}(x_3, x_6)$ $x_{11} = \text{XOR}(x_2, x_{10})$
GIFT	$x_4 = \text{NOR}(x_1, x_0)$ $x_8 = \text{XNOR}(x_1, x_7)$ $x_{12} = \text{NAND}(x_9, x_{11})$	$x_5 = \text{XOR}(x_2, x_4)$ $x_9 = \text{XNOR}(x_6, x_8)$ $x_{13} = \text{XOR}(x_5, x_{12})$	$x_6 = \text{XOR}(x_3, x_5)$ $x_{10} = \text{NAND}(x_3, x_8)$	$x_7 = \text{NAND}(x_2, x_0)$ $x_{11} = \text{XNOR}(x_0, x_{10})$