Constructing Large S-boxes with Area Minimized Implementations

Christopher A. Wood*, Stanisław P. Radziszowski†, Marcin Lukowiak‡
*Department of Computer Science, UC Irvine, CA
†Department of Computer Science, Rochester Institute of Technology, NY
‡Department of Computer Engineering, Rochester Institute of Technology, NY

***Abstract*—Block ciphers are essential cryptographic primitives used for encryption in resource constrained hardware systems. Many modern block ciphers are based on the substitution permutation network (SPN) design. The substitution step is commonly the only non-linear transformation in the cipher, and is usually comprised of a permutation which applies an S-box substitution to each element of the cipher state. In hardware, the S-box implementation has a significant impact on the area consumed by the cipher. Consequently, there have been considerable research and engineering efforts to obtain compact circuit implementations of S-boxes for hardware systems. Building on past work, we present a comprehensive methodology for (1) constructing cryptographically strong affine-power S-boxes over Galois fields and (2) minimizing the VLSI technology-independent combinational logic requirements for their circuit implementations. Motivated by the potential need for larger S-boxes with improved cryptographic properties, we use this methodology to construct area minimized circuits for novel 16-bit S-boxes.**

***Index Terms*—16-bit S-boxes, S-box design, composite Galois fields, S-box combinational circuits.**

I. Introduction

In order to enable area-efficient hardware implementations of block ciphers with substitution permutation network (SPN) designs, such as the Advanced Encryption Standard (AES) [12], the logic resources for each operation in the algorithm must be reduced as much as feasible. Traditionally, research efforts to minimize such resources have focused on the S-box - the only nonlinear operation in the algorithm. For example, minimizing the area required for the AES S-box has been the subject of intense research because combinational implementations of this component often consume a majority of the total logic needed for the algorithm [28], [29], [30], [32], [19], [5], [21], [22], [2]. To aid the implementation of future cryptographic primitives that rely on S-boxes, we present a comprehensive methodology for constructing and implementing cryptographically significant S-boxes with the goal of low-area combinational logic defined in terms of the number of VLSI technology-independent XOR and AND gate requirements.

We believe this methodology and our results may be useful in the design of future cryptographic algorithms, especially for those using large S-boxes for improved nonlinearity and algebraic complexity [11], [20], [7], [12]. Attacks on SPN block ciphers with small S-boxes, such as those used in the AES [12] and GOST [34], will only continue to become more sophisticated in the future. Intuitively, to increase the computational and algorithmic barrier for such attacks, either the number of rounds in the cipher algorithm or the internal state size can be increased. We favor the latter, since our results show that we can implement larger S-boxes with a minimal increase in logic consumption; the relatively small increase may be worth the enhanced security properties offered by 16-bit designs when creating future cryptographic algorithms. This is especially true when performance metrics such as throughput, which are affected by the number of rounds, are of critical importance.

An affine-power S-box $S(x) = A(P(x))$ is the composition of a highly nonlinear power mapping $P : GF(2^n) \to GF(2^n)$ over a binary Galois field and an affine transformation $A : GF(2)^n \to GF(2)^n$. In developing our methodology we build upon the exhaustive mixed basis approach of Canright [5] and combinational logic minimization techniques of Boyar and Peralta [2]. Our methodology is composed of two steps: finding suitable affine-power S-box constructions, and then programmatically and exhaustively searching for implementation parameters (i.e. subfield decompositions and basis representations) that permit area-optimized circuits. We applied our methodology to find area-optimized 16-bit S-boxes over binary Galois fields. Using the affine-inverse construction leveraged by the AES S-box, we exhaustively searched for area-optimized S-boxes for the 21 smallest irreducible polynomials of degree 16 over $GF(2)$. Our search led to a set of implementation parameters that permit an area-optimized circuit composed of 1238 XOR and 144 AND gates. Additionally, we exhaustively searched for S-box circuits over $GF(2^8)$ defined by all 30 irreducible polynomials over $GF(2)$ of degree 8. Our search produced an 8-bit S-box with 103 XOR and 36 AND gates using the field polynomial $t(v) = v^8 + v^6 + v^5 + v^4 + v^2 + v + 1$, surpassing Canright's optimized S-box circuit for the AES, which uses a different field polynomial, by a single XOR gate (prior to further fine-grained logic optimization techniques). We also found new implementation parameters for the AES S-box that yield a reduction in a single XOR gate prior to the application of Boyar and Peralta's logic optimization techniques. Details of these 8-bit S-box constructions can be found in [33].

We emphasize that performing further optimizations, such as gate replacement (e.g., using NAND/NOR gates instead of AND/XOR gates) and circuit-specific designs such as pipelining, are outside the scope of this work. Developing the methodology and finding foundational, technology-independent S-box constructions with ideal gate counts were the primary objectives[1]. This is different from the work done by Wolkerstorfer et al. [32] and Nogami et al. [22] which also focused on power and critical path reduction. Furthermore, the software and workflow we developed can be easily repurposed to find minimal combinational logic representations for arbitrary mappings over binary Galois fields, thereby making it generally applicable to the efficient implementation of other cryptographic algorithms for a variety of resource constrained platforms, including embedded systems, smart cards, and VLSI circuits.

II. Related Work

S-boxes are often constructed as an affine transformation composed with an inverse power mapping over some Galois field, as is the case for the AES. This particular power mapping has many desired cryptographic properties that, in practice, effectively render many known cryptanalysis attacks ineffective. Consequently, much of the research on low-area implementations for the affine-power S-boxes has focused on minimizing the combinational logic required for the multiplicative inverse calculation over binary Galois fields.

Out of all known methods to compute the multiplicative inverse in $GF(2^n)$, the use of subfield decomposition has been the most effective and accepted technique for implementing low-area circuits. The inverse can be computed using the Itoh-Tsujii inversion algorithm [15] or by direct decomposition to bitwise operations over $GF(2)$ [25]. Under the assumption that the elements in a particular field are represented using a normal basis, the number of multiplications required in the Itoh-Tsujii inversion algorithm still yields complex combinational logic for small fields. In comparison, direct decomposition to $GF(2)$ has yielded significantly smaller circuits [29], [30], [19], [5], [21], [22], [3], [2].

To date, the smallest AES S-box using composite field arithmetic and other combinational logic minimization techniques is due to Boyar and Peralta, who found an implementation that required only 83 XOR/XNOR and 32 AND gates [2]. This fell below the previous area record of 104 XOR and 36 AND gate count by Canright in 2005 [5], which was found by trying all mixed basis representations of the field $GF(((2^2)^2)^2)$ to reduce the cost of relevant arithmetic when computing the multiplicative inverse and then factoring all basis change matrices needed to map $GF(2^8)$ to $GF(((2^2)^2)^2)$. Boyar and Peralta improved upon Canright's results by swapping his $GF((2^2)^2)$ inversion circuit for their own and then performing subsequent combinational logic minimization. We model our methodology after both of these approaches.

[1]Combinational circuits to compute composite field arithmetic operations can be easily constructed given the parameters for an S-box and its composite field representation.

III. Constructing Suitable S-Boxes

With the continued improvement of cryptanalytic attacks that include different forms of linear and differential cryptanalysis [18], [1], and algebraic analysis [7], [9], among others [14], [16], [17], it is critically important that S-boxes do not exhibit weaknesses that can be exploited. For example, linear cryptanalysis of SPN-based block ciphers exploits the existence of some linear combinations of input and output bits in the substitution step that occur with high (or low) probability [18]. Therefore, highly nonlinear S-boxes are ideal to reduce the probability that such linear combinations can be found and effectively exploited. To determine if an S-box has this property, we may compute its nonlinearity, denoted $\mathcal{N}_L$ [11]. Other cryptographic metrics of interest include: the maximum and minimum entries of the linear and difference distribution table [18], [1], δ-differential uniformity [23], [24], resiliency and correlation immunity [6], component algebraic immunity [6], XL and XLS algebraic immunity [20], [7], interpolation polynomial algebraic complexity [16], [12], and branch number [12].

Most of these metrics can be easily computed for single n-bit S-boxes over fields $GF(2^n)$ when $n \leq 16$. For example, directly computing the differential uniformity can be done in $\mathcal{O}(2^{3n})$ time, which is feasible for a single 16-bit S-box. This complexity, however, prohibited such computations for all 16-bit S-boxes considered in this work. By computing these metrics for S-box candidates we may quantitatively compare the security of different constructions.

Following results from the literature, we seek to build S-boxes that have high nonlinearity, low differential uniformity, high resiliency, high algebraic immunity, and high algebraic complexity. We focus on these metrics when selecting possible S-box constructions in the form $S(x) = A(x^d)$, where $x \in GF(2^n)$, $0 \leq d < 2^n$, and $A(\cdot)$ is an affine transformation computed by some matrix $\mathbf{M}$ and constant vector c. Power mappings of the form x^d over the field $GF(2^n)$ are typically classified by their exponents d. The mappings over $GF(2^n)$ for even n with substantially high nonlinearity and good differential uniformity properties are shown in Table I.

While other possible constructions exist, such as those based on cryptographically-strong Boolean functions, there are several limitations that make these difficult to use in practice. For instance, they typically do not have compact algebraic expressions. This implies that hardware and software implementations often use lookup tables for such mappings, thereby making their application to 16-bit constructions too expensive.

Since these S-boxes are intended for block ciphers, it is natural to impose the additional requirement that they are bijective. By Fermat's Little Theorem this only occurs when $\gcd\{d, 2^n - 1\} = 1$. Interestingly, with this restriction and the constraint $n = m = 16$, many of the possible values for d are discarded and only the inverse exponent remains. See [33] for a proof of this claim.

TABLE I: Cryptographically-significant power mappings.

Name	*Exponent* (d)	*Reference*
Inverse	$-1 \equiv 2^n - 2$	[24]
Gold	$2^k + 1$, $\gcd\{k, n\} = 1$ for some $1 \leq k \leq 2^n - 1$	[8]
Kasami	$2^{2k} - 2^k + 1$, $\gcd\{k, n\} = 1$ for some $1 \leq k \leq n/2$	[8]
Dobertin	$2^{4k+3k+2k+k} - 1$ over $GF(2^n)$ with $n = 5k$	[8]
Niho	$2^m + 2^{m/2} - 1$ over $GF(2^n)$ with $n = 2m + 1$ and m even $2^m + 2^{(3m+1)/2} - 1$ over $GF(2^n)$ with $n = 2m + 1$ and m odd	[8]
Welch	$2^m + 3$ over $GF(2^n)$ with $n = 2m + 1$	[8]

After finding a candidate S-box with affine transformation matrix $\mathbf{M}$ and constant vector c, our next task was to modify the constructions to increase the algebraic complexity. In order to avoid interpolation attacks, such expressions should have more terms (i.e., be more complex). Perhaps the most common technique for increasing the complexity is to compose an affine transformation with one such power mapping. Cui and Cao [10] proved that the algebraic complexity for any affine-power S-box over $GF(2^n)$ is bounded by $n + 1$. Using the same rationale for the affine transformation selection as presented by Daemen and Rijmen in [13], this procedure searches for affine transformations that have a "complex algebraic expression if combined with the inverse mapping" and, together with the inverse operation, have "no fixed points and no opposite fixed points." In this context, a fixed point or opposite fixed point occurs when there exists an element $x \in GF(2^n)$ such that $S(x) \oplus x = 0^n$ or $S(x) \oplus x = 1^n$. Since there are no known attacks that exploit the existence of fixed points, we opted to lift this constraint in favor of more opportunities for logic optimization.

IV. Searching for Area-Efficient Tower Field Representations

Using composite arithmetic to compute the multiplicative inverse requires arithmetic operations such as addition, multiplication, squaring, and scaling (i.e. multiplication by a constant) in the subfields. The complexity of such arithmetic heavily depends on the representation of elements in the subfields. Polynomial arithmetic is generally more computationally efficient with polynomials of small degree. This can be shown by deriving the expressions for the arithmetic operations in the subfields. For example, given an element $\epsilon \in GF(((2^2)^2)^2)$ (where $r(x) = x^2 + x + \Pi$ defines $GF((2^2)^2)$) represented in a polynomial basis $[1, X]$ with subfield coefficients δ_1 and δ_2, ϵ^{-1} can be computed as [33]

$$\epsilon^{-1} = \delta_1(\delta_2^2 + \delta_1\delta_2 + \delta_1^2\Pi)^{-1}x + (\delta_1 + \delta_2)(\delta_2^2 + \delta_1\delta_2 + \delta_1^2\Pi)^{-1}.$$

If ϵ is represented in a normal basis $[X, X^{16}]$, the expression becomes

$$\epsilon_1^{-1} = ((\delta_1\delta_2 + (\delta_1 + \delta_2)^2\Pi)^{-1}\delta_2)x^{16} + (\delta_1\delta_2 + (\delta_1 + \delta_2)^2\Pi)^{-1}\delta_1)x.$$

Deriving a general expression for inversion in $GF((2^2)^4)$ depends on many factors, including the coefficients of the polynomial $r(x)$ and the basis representation. We omit such derivations here given the numerous possibilities. However, it should be clear that the higher-degree polynomials representing elements in $GF((2^2)^4)$ will lead to less compact expressions than the simple quadratic extension case wherein we can always find an irreducible polynomial $r(x)$ with a unit x coefficient. Consequently, we focus on the tower fields $GF((((2^2)^2)^2)^2)$ and $GF(((2^2)^2)^2)$ for $GF(2^{16})$ and $GF(2^8)$, respectively. Using such isomorphic representations, the cost of all arithmetic operations with respect to the subfields using a polynomial and normal basis is given in Table II.

To this end, let $t(v)$ be a degree 16 (8) irreducible polynomial over $GF(2)$ for $GF(2^{16})$ (analogously $GF(2^8)$), $s(y) = y^2 + \Psi y + \Lambda$, be the irreducible polynomial for $GF((((2^2)^2)^2)^2)$, $r(x) = x^2 + \Theta x + \Pi$ be the irreducible polynomial for $GF(((2^2)^2)^2)$, $q(w) = w^2 + \Omega w + \Sigma$ be the irreducible polynomial for $GF((2^2)^2)$, and finally $p(v) = v^2 + v + 1$ be the *only* irreducible polynomial for $GF(2^2)$. We enforce $\Psi = \Theta = \Omega = 1$ to simplify field arithmetic. Also, we denote by V, W, X, and Y roots of corresponding the polynomials for the fields $GF(2^2)$, $GF((2^2)^2)$, $GF(((2^2)^2)^2)$, and $GF((((2^2)^2)^2)^2)$, respectively, and refer to the forward and inverse basis change matrices needed to map elements from $GF(2^8)$ and $GF(2^{16})$ to their isomorphic tower field partners as $\mathbf{T}$ and $\mathbf{T}^{-1}$.

TABLE II: Cost of $GF((q^2)^2)$ arithmetic based on $GF(q^2)$ (A)ddition, (M)ultiplication, (Sq)uare, (I)nversion, (Sc)ale, and (SS)quare-scale operations for polynomial and normal basis representations.

Operation	*Polynomial Basis*	*Normal Basis*
Inverse	$3M + 2A + 1I + 1SS$	$3M + 2A + I + 1SS$
Add	$2A$	$2A$
Multiply	$3M + 4A + 1Sc$	$3M + 4A + 1Sc$
Square	$2Sq + 1Sc + 1A$	$3A + 2Sq + 1Sc$

Each irreducible polynomial for the fields $GF(2^2)$,...,$GF((((2^2)^2)^2)^2)$ will have two distinct conjugate roots, which we denote as the sets $\{V, V^2\}$, $\{W, W^4\}$, $\{X, X^{16}\}$, and $\{Y, Y^{256}\}$. A polynomial basis for any field can be formed by selecting one of these roots as a basis element in conjunction with the identity element 1, e.g.

$[1, V]$ or $[1, V^2]$ for $GF(2^2)$, whereas a normal basis requires that both roots are used. For each possible combination of basis elements we then programmatically determine the combinational complexity of subfield arithmetic needed to compute the inverse.

For each choice of basis elements we also perform several arithmetic and logic optimizations. For instance, as Satoh [30] mentions, it is possible to save on the number of gates required for a circuit if there exists two $GF((2^m)^2)$ multipliers that have a shared input. This is because both the polynomial and normal multipliers need to compute the sum of the two coefficients for the input elements. Therefore, every shared input factor will save one addition in the subfield. In addition, polynomial and normal multipliers for elements in $GF((2^2)^2)$ and $GF(((2^2)^2)^2)$ each have three subfield multipliers that will share a common factor, thus saving additional sub-subfield addition operations. We also make use of the optimizations to the square-scale operations performed by Canright [5]. At a high level, such optimizations are used to derive compact expressions for the square-scale operations given particular values of Π [5], [33].

Our S-box construction program written in Magma [4] does not support exhaustive common subexpression elimination. This is due to the fact that Magma does not support normal basis representations for finite field elements. Also, it is important to note that, because we do not automatically apply the full set of Canright's optimizations, our gate counts will be *upper bounds* on the total number of gates. That is, the software that was written to count the number of gates for each field representation and basis selection will produce a result that is larger than or equal to what is presented in Canright's work, and as shown in his detailed report, other optimizations can be applied to lower this bound even further. After the tower field implementation parameters have been identified, we then utilize the logic minimization techniques of Paar [26] and Boyar and Peralta [3] to reduce the XOR gate count for the basis change matrices, which are merely linear mappings represented as straight-line programs (SLPs). An SLP for a binary matrix-vector multiplication expression is a finite sequence of lines of the form $u := \lambda v + \mu w$, where λ and μ are elements in $GF(2)$, u, v, and w are variables, and some lines are output of the corresponding multiplication.

In addition to these algebraic and gate-level optimizations, we also follow in the footsteps of Satoh [30] and Canright [5] by performing logic minimizations on merged S-box designs. The merged S-box design simply pairs the forward and inverse S-box operations into the same circuit that use the same inversion component, where the output is determined by a simple multiplexer. We optimize the matrices $\mathbf{T}^{-1}/(\mathbf{MT})^{-1}$ and $\mathbf{MT}/\mathbf{T}$ separately. A high-level overview of the merged circuit is shown in Figure 1.

V. New S-Box Constructions

We measure the complexity, or cost, of a particular S-box as the total number of VLSI technology-independent XOR and AND gates required in a combinational circuit implementation.

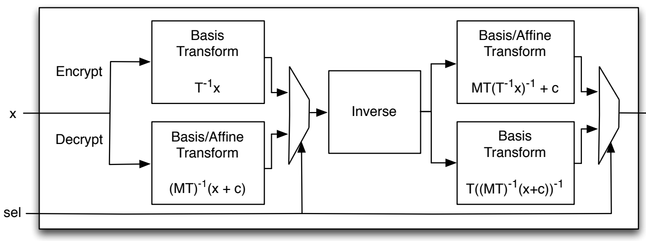

Fig. 1: High-level view of a merged S-box circuit. The sel signal toggles encryption and decryption.

To determine this cost for merged S-box circuits we measure the cost of the basis transformation matrices $\mathbf{T}$ and $\mathbf{T}^{-1}$ merged with the affine transformation matrices $\mathbf{M}$, the cost of a single inversion circuit, and the weight of the affine constant c. Measuring this cost is done in four steps: (1) finding a suitable $\mathbf{M}$ and c, (2) performing an exhaustive search over all possible mixed basis representations of the S-box field to find an *upper bound* on the gates required for the inversion circuit, (3) use the logic optimization technique of Boyar and Peralta [2] to reduce the number of XOR gates required for the merged basis change and affine transformation matrices, and (4) summing the total number of gates in the transformation matrices and inversion circuits. We implemented a standalone Magma program for the first two tasks[2]. We implemented the linear circuit minimization techniques and gate count tasks in a standalone Java program. The results of the Magma program[3], are partitioned as inputs to multiple instances of the Java program running on the Open Science Grid [27], [31] to produce an explicit gate count for each circuit variant.

For 16-bit S-boxes, there are 128 choices for $s(y)$, eight choices for $r(x)$, two choices for $q(w)$, and only one choice for $p(v)$ that have a trace of unity. Since each of these polynomials has two distinct conjugate roots that can be used to represent the respective field elements with a polynomial or normal basis, there are exactly three basis element combinations in all degree-two extension subfields of $GF(2^{16})$. Consequently, there is a total of 165888 possible cases to consider for a single polynomial $t(v)$. Since the basis change matrices depend on the representation of $GF(2^{16})$, and there are 4080 candidates for $t(v)$, this means that we must consider about 6×10^8 possible cases to find a minimal transformation. Due to computational limitations, we selectively focused on the 21 smallest $t(v)$ polynomials when searching for 16-bit S-box implementation parameters.

We applied our methodology to find new 16-bit S-boxes over $GF(2^{16})$ S-boxes over $GF(2^8)$. Our search for 16-bit S-boxes considered the 21 smallest degree 16 irreducible polynomials $t(v)$ over $GF(2)$. This search yielded several S-box constructions with small gate counts prior to (linear) logic

[2]Magma was chosen over C/C++ due to its readily available support for composite field constructions and Galois field arithmetic.

[3]Source code for all programs is available upon request.

optimizations of the basis change matrices. For the smallest irreducible polynomial $t(v) = v^{16}+v^5+v^3+v+1$, we found a set of implementation parameters that permitted a circuit with 1238 XOR and 144 AND gates. This candidate, shown in Figure 2, uses the basis sets $[1, V]$, $[1, W]$, $[1, X]$, $[Y^{256}, Y]$ to represent elements in $GF((((2^2)^2)^2)^2)$ and its respective subfields, where $\Sigma = v$, $\Pi = vw+v$, and $\Lambda = (vw+v)x+w$. The affine transformation and basis change matrices used to obtain the circuit are shown in Figure 2. A larger subset of these constructions are shown in Table III.

To parse the results in this table, we first note that a basis $B = [\beta^{n_1}, \beta^{n_2}]$ is used to represent an arbitrary element $\alpha \in GF((q^m)^2)$ as $\alpha = a_1\beta^{n_1} + a_2\beta^{n_1}$ for some $a_1, a_2 \in GF(q^m)$. The basis element powers n_1 and n_2 are chosen such that B is a polynomial or normal basis. In particular, if $n_1 = 0$, then B must be a polynomial basis where $n_2 \in \{1, q^m\}$. If $n_1 \neq 0$ and $n_2 \neq 0$ then B must be a normal basis. The order of normal basis elements in B depends on how Magma selects primitive elements. Specifically, if β^{q^m} is chosen as the primitive element then our S-box construction program will fix the basis $B = [\beta^{q^m}, \beta]$.

We encode each irreducible polynomial $t(v)$, constant c, and binary matrix ($\mathbf{T}$, $\mathbf{T}^{-1}$, and $\mathbf{M}$ in row order), which are described in Section IV, as a hexadecimal string. Σ, Π, and Λ, the irreducible polynomial coefficients described in Section IV, are shown with a polynomial basis. We also use the notation $\mathbb{F}!v$ to denote the embedding of v into the field $\mathbb{F}$, where $\mathbb{F} = GF(2^8)$ or $\mathbb{F} = GF(2^{16})$ for 8 and 16-bit S-boxes, respectively. The subfield bases are as described in Section IV. Finally, the Inv. and Total fields denote the number of XOR gates required for the multiplicative inverse and merged S-box circuits, respectively. More information about these encodings, as well as more extensive results for both 8- and 16-bit S-boxes, can be found in [33].

VI. Conclusion

In this work we presented a comprehensive methodology for identifying cryptographically significant S-box constructions based on power mappings over $GF(2^n)$ and searching for composite-field representations that permit low-area hardware implementations. Motivated by a potential need for larger S-boxes, we used our procedure to find 16-bit S-boxes that permit area-minimized implementations. We believe this methodology and our results may be useful in the design of future cryptographic algorithms.

VII. Acknowledgements

This research was done using resources provided by the Open Science Grid, which is supported by the National Science Foundation and the U.S. Department of Energy's Office of Science. We owe many thanks to Mats Rynge for his guidance throughout our use of the OSG.

References

[1] Biham, E. and Shamir, A. Differential Cryptanalysis of DES-Like Cryptosystems. *Journal of Cryptology* **4.1** (1991), 3-72.

[2] Boyar, J. and Peralta, R. A Small Depth-16 Circuit for the AES S-Box. *IFIP Advances in Information and Communication Technology, Springer Berlin Heidelberg* 376 (2012), 287-298.

[3] Boyar, J., Matthews, P., and Peralta, R. Logic Minimization Techniques with Applications to Cryptology. *Journal of Cryptology* (2012), 1-33.

[4] Cannon, J. and Steel, A. The Magma computational algebra system. *Software available online* (magma.maths.usyd.edu.au) (2005).

[5] Canright, D. A Very Compact S-Box for AES. *CHES 2005 - Cryptographic Hardware and Embedded Systems, Springer Berlin Heidelberg* (2005), 441-455.

[6] Carlet, C. and Prouff, E. On a New Notion of Nonlinearity Relevant to Multi-output Pseudo-random Generators. *Selected Areas in Cryptography, Springer Berlin Heidelberg* (2004).

[7] Courtois, N. T. and Pieprzyk, J. Cryptanalysis of block ciphers with overdefined systems of equations. *Advances in Cryptology - ASIACRYPT 2002, Springer Berlin Heidelberg*, (2002), 267-287.

[8] Courtois, N. T., Debraize, B., and Garrido, E. On Exact Algebraic [Non-]Immunity of S-Boxes Based on Power Functions. *Information Security and Privacy, Springer Berlin Heidelberg* (2006).

[9] Bard, G. V., Courtois, N. T., and Jefferson, C. Efficient Methods for Conversion and Solution of Sparse Systems of Low-Degree Multivariate Polynomials over $GF(2)$ via SAT-Solvers. *Presented at ECRYPT workshop Tools for Cryptanalysis* (2007).

[10] Cui, L. and Cao, Y. A New S-Box Structure Named Affine-Power-Affine. *International Journal of Innovative Computing, Information and Control* 3.3 (2007), 751-759.

[11] Cusick, T. W. and Stănică, P. Cryptographic Boolean Functions and Applications. *Academic Press* (2009).

[12] Daemen, J. and the, V. Advanced Encryption Standard (AES) (FIPS 197). *Technical report, Katholijke Universiteit Leuven/ESAT* (2001).

[13] Daemen, J. and Rijmen, V. The Design of Rijndael: AES-the Advanced Encryption Standard. *Springer* (2002).

[14] Gilbert, H. and Peyrin, T. Super-Sbox Cryptanalysis: Improved Attacks for AES-like Permutations. *Fast Software Encryption, Springer Berlin Heidelberg* (2010).

[15] Itoh, T. and Tsujii, S. A Fast Algorithm for Computing Multiplicative Inverses in $GF(2^m)$ Using Normal Bases. *Information and Computation* 78.3 (1988), 171-177.

[16] Jakobsen, T. and Knudsen, L. R. The Interpolation Attack on Block Ciphers. *4th International Workshop on Fast Software Encryption LNCS, Springer* 1267 (1997), pp. 28-40.

[17] Knudsen, L. R.. Truncated and Higher Order Differentials. *Fast Software Encryption, Springer Berlin Heidelberg* 1008 (1995).

[18] Matsui, M. Linear cryptanalysis method for DES cipher. *Advances in Cryptology - EUROCRYPT93, Springer Berlin Heidelberg*, (1994).

[19] Mentens, N., Batina, L., Preneel, B., and Verbauwhede, I. A Systematic Evaluation of Compact Hardware Implementations for the Rijndael S-Box. *Topics in Cryptology-CT-RSA, Springer Berlin Heidelberg* (2005), 323-333.

[20] Yassir, N., Gupta, K. C., and Gong, G. Algebraic Immunity of S-Boxes Based on Power Mappings: Analysis and Construction. *IEEE Transactions on Information Theory* 55.9 (2009), 4263-4273.

[21] Nikova, S., Rijmen, V., and Schläffer, M. Using Normal Bases for Compact Hardware Implementations of the AES S-box. *Security and Cryptography for Networks. Springer Berlin Heidelberg* (2008), 236-245.

[22] Nogami, Y., Nekado, K., Toyota, T., Hongo, N., and Morikawa, Y. Mixed Bases for Efficient Inversion in $F_{((2^2)^2)^2}$ and Conversion Matrices of SubBytes of AES. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* E94-A:6 (2011), 1318-1327.

[23] Nyberg, K. Perfect nonlinear S-boxes. *Advances in Cryptology - EUROCRYPT91*. Springer Berlin Heidelberg (1991).

[24] Nyberg, K. Differentially Uniform Mappings for Cryptography. *Advances in Cryptology - Eurocrypt93*. Springer Berlin Heidelberg (1994).

[25] Paar, C. Some Remarks on Efficient Inversion in Finite Fields. *1995 IEEE International Symposium on Information Theory* (1995).

[26] Paar, C. Optimized Arithmetic for Reed-Solomon Encoders. *Proceedings of the 1997 IEEE International Symposium on Information Theory* (1997).

[27] Pordes, R., Petravick, D., Kramer, B., Olson, D., Livny, M., Roy, A., Avery, P., Blackburn, K., Wenaus, T., Wŭrthwein, F., Foster, I., Gardner, R., Wilde, M., Blatecky, A., McGee, J., and Quick, R. The Open Science Grid. *Journal of Physics: Conference Series* **78** (2007).

[28] Rijmen, V. Efficient Implementation of the Rijndael S-box. *Katholieke Universiteit Leuven, Dept. ESAT, Belgium* (2000).

$$z = S(x) = \begin{pmatrix} 0&0&1&0&0&0&0&1&0&0&1&1&1&1&1&0\\ 1&1&0&0&0&0&0&1&0&1&1&0&1&0&1&0\\ 1&1&0&0&1&0&1&1&0&1&0&1&0&0&1&1\\ 1&1&1&0&0&0&1&0&0&1&1&0&0&0&0&0\\ 1&1&0&0&0&1&1&0&0&1&1&1&1&0&1&1\\ 0&1&0&0&0&0&1&1&0&1&1&1&1&1&0&1\\ 0&0&1&0&1&0&1&0&1&1&0&0&1&1&0&0\\ 1&0&1&1&1&0&1&1&0&0&0&1&0&1&1&1\\ 0&1&0&0&0&0&0&0&1&0&0&1&1&1&0&1\\ 1&0&1&1&0&0&0&1&0&0&1&0&1&0&0&0\\ 1&0&1&0&0&1&1&1&0&0&1&1&0&1&0&0\\ 1&0&1&1&1&0&1&1&1&1&0&1&1&0&0&1\\ 1&0&1&0&0&1&0&1&1&0&0&1&0&0&0&1\\ 0&1&0&0&0&1&1&1&1&0&0&0&0&0&0&1\\ 1&0&0&0&1&1&0&1&0&1&1&1&1&0&0&0\\ 1&1&0&1&0&1&1&0&1&0&0&1&1&0&0&0 \end{pmatrix} \begin{pmatrix} y_{15}\\ y_{14}\\ y_{13}\\ y_{12}\\ y_{11}\\ y_{10}\\ y_9\\ y_8\\ y_7\\ y_6\\ y_5\\ y_4\\ y_3\\ y_2\\ y_1\\ y_0 \end{pmatrix} + \begin{pmatrix} 0\\ 1\\ 0\\ 0\\ 0\\ 1\\ 0\\ 1\\ 1\\ 0\\ 1\\ 1\\ 0\\ 1\\ 1\\ 1 \end{pmatrix}$$

$$x = (S^{-1}(z))^{-1} = \begin{pmatrix} 0&1&0&1&0&1&1&1&0&0&1&0&0&0&0&1\\ 1&1&0&1&0&0&1&0&1&0&1&1&1&1&0&1\\ 1&0&1&1&1&1&0&1&0&1&1&0&0&0&0&0\\ 0&0&1&0&1&1&1&0&1&0&1&1&1&0&1&0\\ 1&1&1&1&1&0&0&0&1&0&0&0&0&1&0&0\\ 0&0&0&1&0&1&0&0&0&0&1&1&1&1&1&1\\ 1&0&1&0&0&0&0&0&0&1&1&0&1&0&1&1\\ 0&0&1&0&1&1&1&0&0&1&0&1&1&1&1&0\\ 0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0\\ 1&1&1&0&0&1&1&1&1&0&1&1&1&0&0&0\\ 0&1&1&0&1&1&1&1&0&0&1&0&1&1&1&1\\ 1&0&0&1&1&0&0&0&1&0&0&1&1&0&1&1\\ 1&0&0&0&0&1&0&1&1&1&0&0&1&0&1&0\\ 1&0&0&0&0&1&1&1&1&1&0&1&1&1&1&1\\ 1&1&1&0&0&1&1&0&1&0&0&1&1&1&1&1\\ 0&1&0&0&1&0&1&0&1&0&0&1&0&0&0&1 \end{pmatrix} \begin{pmatrix} z_{15}\\ z_{14}+1\\ z_{13}\\ z_{12}\\ z_{11}\\ z_{10}+1\\ z_9\\ z_8+1\\ z_7+1\\ z_6\\ z_5+1\\ z_4+1\\ z_3\\ z_2+1\\ z_1+1\\ z_0+1 \end{pmatrix}$$

$$\mathbf{T} = \begin{pmatrix} 0&1&0&1&0&0&0&0&1&0&0&0&0&1&0&0\\ 0&1&1&0&0&1&1&1&0&0&1&0&0&1&1&1\\ 0&0&0&1&1&0&0&1&0&0&0&1&0&0&0&1\\ 1&1&0&0&0&0&1&1&0&1&0&1&0&0&1&1\\ 1&1&0&0&1&0&0&1&0&0&0&1&0&1&0&1\\ 0&0&1&1&0&1&1&1&1&0&0&0&1&0&0&1\\ 0&0&0&1&1&0&1&0&1&0&1&0&0&1&0&0\\ 0&0&1&0&1&0&1&0&0&1&1&1&0&1&0&0\\ 1&0&1&0&0&1&1&1&1&0&1&1&0&0&1&1\\ 0&0&0&1&0&1&0&1&0&1&1&0&0&1&0&1\\ 1&1&0&1&1&0&1&1&0&0&1&0&0&0&0&1\\ 0&1&1&1&0&1&0&0&1&1&1&0&0&0&1&0\\ 0&1&0&1&1&1&0&0&1&1&0&1&0&0&0&0\\ 1&0&0&1&0&0&0&0&0&0&1&0&0&1&0&0\\ 0&1&0&0&0&1&0&0&0&1&0&1&0&0&1&0\\ 0&1&0&0&0&0&1&1&1&0&1&1&0&0&0&0 \end{pmatrix}$$

$$\mathbf{T}^{-1} = \begin{pmatrix} 1&0&1&0&0&0&0&1&1&0&0&0&0&0&1&0\\ 1&0&0&0&1&0&0&0&0&1&0&0&1&1&0&0\\ 1&1&0&0&1&0&0&0&1&1&0&1&1&0&1&0\\ 0&1&1&1&1&0&1&1&1&0&1&1&1&0&0&0\\ 0&0&0&1&1&0&0&1&0&0&0&1&1&0&0&0\\ 0&1&0&1&1&0&1&0&1&0&0&0&0&1&1&0\\ 1&0&1&1&0&0&1&1&0&0&0&1&1&1&0&0\\ 1&0&0&0&1&0&0&0&0&0&1&0&0&0&0&1\\ 0&0&0&0&1&0&0&1&1&0&0&0&0&0&1&0\\ 1&0&1&0&0&0&1&1&0&1&0&1&1&0&1&0\\ 1&0&1&0&0&0&0&0&0&1&0&0&1&0&0&0\\ 0&0&0&1&1&0&1&0&1&0&1&1&1&0&1&0\\ 0&0&0&0&1&1&1&1&0&1&1&0&0&0&0&0\\ 0&1&1&1&1&0&1&0&0&1&1&1&0&1&1&0\\ 0&1&1&0&1&0&1&1&0&0&1&0&1&0&0&0\\ 1&1&0&1&0&0&0&0&0&0&1&1&1&0&1&1 \end{pmatrix}$$

Fig. 2: The 16-bit S-box and basis change matrices for the polynomial $t(v) = v^{16} + v^5 + v^3 + v + 1$. The vector y is the inverse of the element x in $GF(2^{16})$ (or $\bar{0}$ if $x = 0$). Accordingly, the output $y = S^{-1}(z)$ is inverted in the same way to obtain the original element x.

TABLE III: Subset of the smallest area-optimized 16-bit S-box constructions for merged circuit implementations found using our methodology.

$t(v)$	1012F	1018F	10175	1015D
Σ	$v+1$	$v+1$	v	v
Π	$vw+1$	$vw+v$	vw	$(v+1)w+1$
Λ	$(vw+1)x$	$(vw+1)x+vw+v+1$	$((v+1)w+v)x$	$((v+1)w+1)x$
$\mathbb{F}!v$	8AA4	A477	F723	10C
$\mathbb{F}!w$	5628	6610	953F	1B79
$\mathbb{F}!x$	E432	45D1	C130	8E3D
$\mathbb{F}!y$	7FC0	A8D2	12A9	849F
Bases	$[V, V^2], [1, W^4]$ $[1, X^{16}], [Y^{256}, Y]$	$[1, V^2], [1, W]$ $[X, X^{16}], [Y^{256}, Y]$	$[1, V^2], [1, W]$ $[X, X^{16}], [1, Y^{256}]$	$[V, V^2], [1, W^4]$ $[1, X^{16}], [Y, Y^{256}]$
$\mathbf{T}^{-1}$	5604FC5A41E67B 644A40A8BEF62D	481CC788160890 C95C3826FA6AAC	2EB8C0C54644A2 26B89D3EAFE542	1040FB41564B58 37AA5A8B8CF735
$\mathbf{T}$	A03F998C29ECEC A261AA8C68D89B	A923247E3ED95F 525A598C2463DC	0617A0B82ECAFE 880051803463FD	0ED2C47C0C0704 0CE435D4064289
$\mathbf{M}$	9F5A116333100C B33A4604FD272A	66FEAC4F696AC9 C4275E7DDCBA77	A6EAF2D22BC542 875BE444ECAF5A	1BB5CC9E4C55E7 F139D7E3584A5D
c	1A2E	8EA3	39F8	F9D8
Inverse	367	376	390	367
Total	1209	1230	1231	1238

[29] Rudra, A., Dubey, P. K., Jutla, C. S., Kumar, V., Rao, J. R., and Rohatgi, P. Efficient Rijndael Encryption Implementation with Composite Field Arithmetic. *Cryptographic Hardware and Embedded Systems - CHES, Springer Berlin Heidelberg* (2001).

[30] Satoh, A., Morioka, S., Takano, K., and Munetoh, S. A Compact Rijndael Hardware Architecture with S-Box Optimization. *Advances in Cryptology - ASIACRYPT, Springer Berlin Heidelberg* (2001), 239-254.

[31] Sfiligoi, I., Bradley, D. C., Holzman, B., Mhashilkar, P., Padhi, S., and Würthwein, F. The pilot way to Grid resources using glideinWMS. *2009 WRI World Congress on Computer Science and Information Engineering* **2** (2009), 428-432.

[32] Wolkerstorfer, J., Oswald, E., and Lamberger, M. An ASIC Implementation of the AES SBoxes. *Topics in Cryptology CT-RSA 2002, Lecture Notes in Computer Science* **2271** (2002), 67-78.

[33] Wood, C. A. Large Substitution Boxes with Efficient Combinational Implementations. *M.S. Thesis, Computer Science, Rochester Institute of Technology* (August 2013).

[34] Zabotin, I. A., Glazkov, G. P., Isaeva, V. B. Cryptographic Protection for Information Processing Systems, *Government Standard of the USSR, 24GOST 28147-89, Government Committee of the USSR for Standards* (1989). In Russian, translated to English in [40].