Reduced Overhead Gate Level Logic Encryption

Kyle Juretus
Drexel University
Philadelphia, Pennsylvania 19104
kjj39@drexel.edu

Ioannis Savidis
Drexel University
Philadelphia, Pennsylvania 19104
isavidis@coe.drexel.edu

ABSTRACT

Untrusted third-parties are found throughout the integrated circuit (IC) design flow resulting in potential threats in IC reliability and security. Threats include IC counterfeiting, intellectual property (IP) theft, IC overproduction, and the insertion of hardware Trojans. Logic encryption has emerged as a method of enhancing security against such threats, however, current implementations of logic encryption, including the XOR or look-up table (LUT) techniques, have high per-gate overheads in area, performance, and power. A novel gate level logic encryption technique with reduced per-gate overheads is described in this paper. In addition, a technique to expand the search space of a key sequence is provided, increasing the difficulty for an adversary to extract the key value. A power reduction of 41.50%, an estimated area reduction of 43.58%, and a performance increase of 34.54% is achieved when using the proposed gate level logic encryption instead of the LUT based technique for an encrypted AND gate.

1. INTRODUCTION

The design of an integrated circuit (IC) typically focuses on methods to increase performance and reduce power and area. More importantly, an IC must function as intended. The current monetary and time constraints placed on semiconductor companies have introduced an increasing number of third-parties into the IC design flow, creating concern over the ability to trust the given hardware. As a result, the security of the IC against malicious adversaries is becoming a critical design consideration.

As more and more IC design firms utilize third party foundries, partially due to the $5 billion dollar investment required for an in-house fabrication facility [1], the threat of a malicious foundry has gained significant attention. The level of information a third party foundry is privy to further increases the threat, as a foundry typically has access to the GDS-II file and possesses the necessary tools and knowledge required to reverse engineer a design from the GDS-II file alone [2].

GLSVLSI '16, May 18-20, 2016, Boston, MA, USA

DOI: http://dx.doi.org/10.1145/2902961.2902972

The level of information possessed by the third party foundry allows for the theft of intellectual property (IP), the ability to counterfeit and overproduce ICs, and the ability to insert malicious circuitry (hardware Trojans) into an IC design. At the very least, these threats pose a serious concern in terms of monetary loss. Pirating and counterfeiting are expected to result in losses of $1.7 trillion dollars in 2015 [3], and a 2008 analysis estimated the IC industry lost $4 billion dollars in IP revenue alone [4]. The monetary concerns are overshadowed by the potential harm caused by an IC that does not function to design specifications. An IC that fails to meet the original specifications causes increased failure rates, produces logical errors, and/or includes hardware Trojans embedded in the IC. Hardware Trojans provide additional challenges as many different attack vectors, including denial of service, stealing information, and/or corrupting functionality, are possible [5]. The challenge in security then becomes protecting against the wide variety of possible threats.

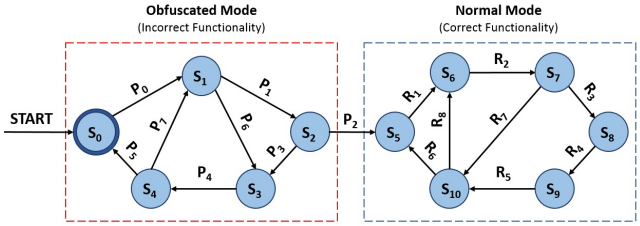

Figure 1: Modified state transition graph (STG) with an enabling sequence of P0, P1, P2 required to enter normal circuit operation [6].

One security measure that aims to prevent IP theft, IC counterfeiting and overproduction, and the insertion of hardware Trojans is logic encryption[1]. Logic encryption exists in both sequential [9, 6] and combinational [7, 10, 11, 12] methodologies. Sequential logic encryption typically involves the application of a required sequence before correct circuit operation is achieved. An example of sequential logic encryption is shown in Fig. 1, where the sequence of P_0, P_1, and P_2 must be entered before the IC operates correctly.

Combinational logic encryption adds key gates to the circuit, requiring the adversary to determine the key before correct functionality is obtained. In addition, combinational

[1] Researchers have previously referred to logic encryption as logic obfuscation [7], but [8] distinguishes between the two as logic encryption prevents black-box use of the design.

logic encryption changes the logic within the circuit, creating additional obfuscation over sequential logic encryption and therefore providing an advantage against IP theft. Without the key, it is also more difficult for an adversary to counterfeit or overproduce an IC, as the functionality of the circuit is unknown. Logic encryption also prevents the insertion of hardware Trojans, as the entire design is no longer known to an adversary, making it more difficult to insert a Trojan without causing unintended actions that are more readily detected.

While combinational logic encryption provides a method to increase IC security against a multitude of threats, current logic encryption techniques result in high overheads in performance, power, and area. A novel gate level implementation of logic encryption that significantly reduces the per-gate overhead of encrypting a gate is described in this paper. The reduction in per-gate overhead provides the ability to increase the key size without increasing the total overhead of current logic encryption methodologies, further deterring adversaries from detecting the key.

The paper begins with an introduction to current combinational logic encryption methodologies in Section 2. A per-gate overhead analysis of the described logic encryption techniques is provided in Section 3, followed by an explanation of the proposed gate topologies in Section 4. Circuit topologies to utilize key expansion, a methodology to potentially double the key search space for an adversary, are described in Section 5. A comparison between the proposed topologies and current techniques is described in Section 6. Finally, conclusions are offered in Section 7.

2. COMBINATIONAL LOGIC ENCRYPTION

Combinational logic encryption requires a key to enable the correct operation of an IC by altering the logic structure of the circuit to include key gates. The two prominent logic encryption techniques are: 1) the insertion of a LUT as a gate replacement [12], and 2) the utilization of XOR/XNOR gates [7, 10]. An additional technique is based on the insertion of 2x1 MUXes [11], which requires connecting the correct output to one MUX input and a net that carries the negated version of the correct net to the other input. The select signal of the MUX then serves as the key input, with either the correct or negated net passed to the next stage of logic. Finding a net that is always the negation of the correct input is difficult and limits the use of 2x1 MUX logic encryption.

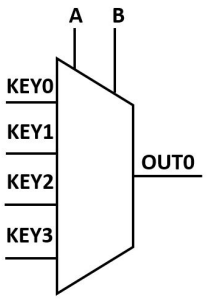

Figure 2: Logic encryption with the use of a 4x1 MUX.

2.1 LUT Based Encryption

The method proposed in [12] utilizes a MUX to encrypt a circuit in the form of a look-up table structure. The key is no longer applied to the select signals of the MUX directly. Instead the key values are passed as inputs to the MUX and set the functionality of the gate. The methodology to encrypt a single gate with a 4x1 MUX structure is shown in Fig. 2. Note that the illustration in Fig. 2 does not include any memory elements attached to the inputs of the 4x1 MUX, as memory is not explicitly required to encrypt the functionality of a gate. If memory elements are used in the circuit, the per-gate encryption overhead increases significantly.

An advantage of the 4x1 MUX based logic encryption method is that any 2-input function is realized, generating more functional combinations for an adversary to search before the correct key value is determined. In addition, the original gates are removed from the circuit and are replaced with 4x1 MUXes, leaving no remnants of the original functionality. Gate replacement with 4x1 MUXes is dissimilar to the XOR encryption discussed in Section 2.2, which requires the addition of inverters to the circuit to prevent adversaries from knowing the original gate functionality.

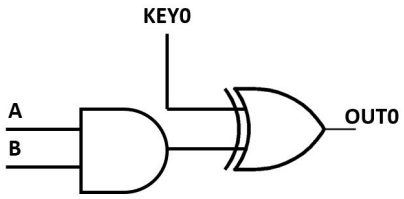

Figure 3: Logic encryption with the use of an XOR gate.

2.2 XOR Based Encryption

The XOR based encryption methodology connects the key value to one input of the XOR gate and the net to encrypt to the other input. By setting the value of the key, the output net OUT0 may either be left with the original or negated input value. An example of encrypting an AND gate with an XOR is shown in Fig. 3. The XOR behaves as an inverter when KEY0 is 1 and as a buffer when KEY0 is 0. Therefore, when KEY0 is 1, an incorrect value is obtained on OUT0. Unlike the 4x1 MUX based encryption, the original gate is not removed from the circuit. Inverters are added in select locations, essentially inverting the key, to deter adversaries from knowing the value of KEY0 that obtains the correct functionality [7]. While the additional inverters secure the key value, the area and power overhead increase.

3. OVERHEAD OF LOGIC ENCRYPTION

The per-gate overheads in performance, power, and area associated with encrypting a gate using the XOR and LUT based encryption techniques are described in this section. For comparison, the performance and area of the standard un-encrypted gates are listed in Table 1. In addition, both the average power and average leakage power of the un-encrypted standard cell gates are provided in Table 2.

The circuits presented in the paper are implemented in an IBM 180 nm technology. The power and performance were determined by matching the drive strength of the gates to ensure a fair analysis between the standard cells and the

Table 1: Analysis of the propagation delay and area of standard cells from a 180 nm fabrication process.

Standard Cell	Propagation Delay (ps)	Area (μm^2)
AND	69.79	30.73
NAND	36.71	23.25
OR	92.90	30.73
NOR	42.09	23.25
XOR	91.23	45.69
XNOR	106.7	41.62

equivalent encrypted topologies. The following simulation characteristics were applied:

1. All simulations were completed with a load capacitance of 5 fF.
2. The worst case propagation delay is determined from the low to high and high to low transient delays.
3. The area was obtained from a layout of the implemented topologies.
4. The average power is determined by matching the duration and input transitions for a given gate function.
5. The highest average leakage power is reported from all possible input combinations to a gate.

Table 2: Analysis of the average and leakage power of standard cells from a 180 nm fabrication process.

Standard Cell	Average Power (nW)	Average Leakage Power (pW)
AND	70.40	192.5
NAND	72.67	149.7
OR	64.52	252.3
NOR	96.85	193.6
XOR	148.0	455.3
XNOR	204.6	527.6

Note that for the analysis of the overhead for the XOR and LUT encryption techniques, no inverters were added after the XOR encryption gates and no memory elements were used for the LUT based approach. The values provided for the per-gate overhead of each method are therefore highly optimistic as compared to implementations that include the inverters and memory.

The per-gate overheads in power, performance, and area of the XOR based logic encryption technique are listed in Table 3. Similarly, the overheads of the LUT based logic encryption method are listed in Table 4. The leakage power for both the XOR and LUT encryption techniques is provided in Table 5.

The large overheads in power, area, and performance indicated by the results listed in Tables 3, 4, and 5 limit the use of the XOR and LUT logic encryption techniques in many IC applications. The large per-gate encryption overhead limits the acceptable signal paths to place gates without decreasing performance. In addition, the total number of encrypted gates placed in a circuit is limited by power and area constraints. As ICs must meet performance, power, and area constraints, the large per-gate overheads limit the added security when implementing the logic encryption techniques. It is therefore imperative to reduce the per-gate encryption overhead.

Table 3: Analysis of the propagation delay, power, and area of XOR based logic encryption. Per-gate overheads are provided as percent increases over the standard cell values in Tables 1 and 2.

Standard Cell	Prop. Delay (ps)	Power (nW)	Area (μm^2)
AND	151.3 (116.8%)	150.4 (113.6%)	63.73 (107.4%)
NAND	127.6 (247.6%)	142.2 (95.68%)	63.73 (174.1%)
OR	157.5 (69.54%)	174.6 (170.6%)	63.73 (107.4%)
NOR	134.3 (219.1%)	168.5 (73.98%)	63.73 (174.1%)
XOR	181.8 (99.27%)	219.3 (48.18%)	84.50 (84.94%)
XNOR	201.3 (88.66%)	226.3 (10.61%)	84.50 (101.4%)
Average	**140.2%**	**85.45%**	**124.9%**

Table 4: Analysis of the propagation delay, power, and area of 4x1 MUX based logic encryption. Per-gate overheads are provided as percent increases over the standard cell values in Tables 1 and 2.

Standard Cell	Prop. Delay (ps)	Power (nW)	Area (μm^2)
AND	122.5 (75.53%)	146.0 (107.4%)	90.58 (194.8%)
NAND	124.6 (239.4%)	157.0 (116.0%)	90.58 (289.6%)
OR	120.8 (30.03%)	142.2 (120.4%)	90.58 (194.8%)
NOR	126.8 (201.3%)	158.8 (63.96%)	90.58 (289.6%)
XOR	124.0 (35.92%)	191.8 (29.59%)	90.58 (98.25%)
XNOR	124.6 (16.78%)	191.7 (6.310%)	90.58 (115.9%)
Average	**99.82%**	**73.95%**	**197.2%**

Table 5: Analysis of the average leakage power of the XOR and LUT based logic encryption methods. Per-gate overheads are provided as percent increases over the standard cell values listed in Table 2.

Standard Cell	XOR Leakage (pW)	LUT Leakage (pW)
AND	673.8 (250.0%)	895.8 (365.3%)
NAND	581.0 (288.1%)	841.8 (462.3%)
OR	661.5 (162.2%)	863.4 (242.2%)
NOR	648.9 (235.2%)	973.7 (402.9%)

4. GATE LEVEL LOGIC ENCRYPTION

The motivation for gate level logic encryption is due to the large performance, power, and area overheads associated with the XOR (see Fig. 3) and LUT (see Fig. 2) based logic encryption methodologies. Consider an un-encrypted AND gate as a way to demonstrate the inefficiencies of current logic encryption methodologies. The XOR method shown in Fig. 3 adds an additional XOR gate after the AND gate to mask OUT0 with KEY0. However, the data must now propagate through both the AND and XOR gates, resulting in a reduction in performance. The added XOR gate also contributes power and area over the original un-encrypted AND gate.

The LUT based approach replaces the AND gate with a 4x1 MUX structure as shown in Fig. 2, which is used to implement any two input function. While the per-gate security is enhanced, the performance, power, and area are negatively affected by the additional transistors required to implement a given function. Two novel gate level logic en-

cryption methodologies are described in this section that enhance the security of a gate while also reducing the power, performance, and area overheads as compared to both the LUT and XOR techniques.

4.1 Stack-based Topology

The first topology is termed *stack-based* (see Fig. 4) as portions, or stacks, of logic are turned off/on depending on the key input. The topology depicted in Fig. 4 functions as either a NAND or NOR gate depending on the value of KEY0. When KEY0 is 0, the PMOS stack is activated allowing the gate to behave as a NAND. When KEY0 is set to 1, the NMOS stack is activated and the gate behaves as a NOR. Placing the KEY0 transistors on the OUT net of the gate reduces the capacitance connected to OUT by essentially disconnecting one of the logic stacks during execution. A smaller output capacitance reduces the performance and power overhead of implementing the stack based logic encryption.

Truth Table

KEY0	A	B	OUT
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

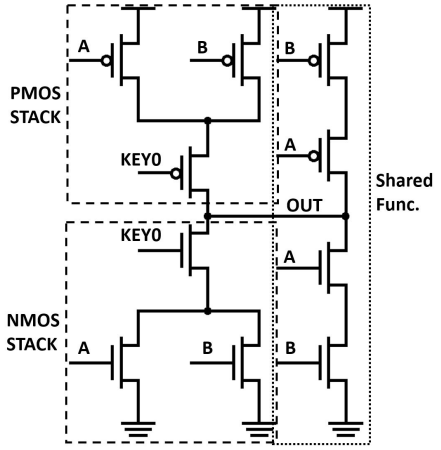

Figure 4: Stack-based topology implementing a NAND or NOR gate depending on the value of KEY0.

The NAND/NOR stack-based topology has two important characteristics: 1) The ability to share common input-output combinations between implemented logical functions, and 2) implementations that do not require negated inputs. For example, the NAND and NOR gates have the same logical output when inputs A and B are both 0 or 1, which permits shared functionality as indicated by the fine dashed box (*shared func.*) shown in Fig. 4. The shared logic does not require a key transistor as the NAND and NOR produce the same output for both the 00 and 11 input combinations. The ability to forgo the key transistor reduces the area, power, and performance overheads of utilizing the stack-based approach.

The second characteristic of the NAND/NOR topology, is not requiring negated inputs of A and B, which removes two additional inverters from the circuit and further reduces the overhead of the methodology. The negated inputs are required if a NAND/AND stack-based topology was implemented instead. The negated logic is needed as the same input combinations must either turn on a PMOS or NMOS stack depending on the key.

While both characteristics of the stack-based topology allow for the reduction in the overhead in performance, power, and area of the NAND/NOR gate, sharing the two common output cases when inputs A and B are both 0 or 1 reduces the percentage of inputs that produce incorrect outputs when an incorrect key is applied. Both characteristics that reduce the overhead must be ignored to achieve complete output corruption when an incorrect key is entered, which results in a larger overhead in power and performance.

Truth Table

KEY0	KEY1	A	B	OUT
0	0	X	X	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	X	X	1

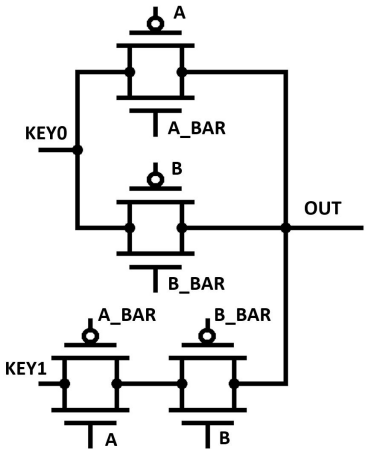

Figure 5: Transmission gate based topology implementing an AND/NAND depending on the values of KEY0 and KEY1.

4.2 Transmission Gate Topology

The transmission gate topology shown in Fig. 5 is better suited to completely corrupt the outputs of the implemented circuit functions. The key value is passed through the transmission gates, eliminating the additional key transistors required in the stack-based topology. The removal of the key transistors reduces the overhead in the performance, power, and area, making the transmission gate topology better suited to replicate the functionality of the XOR based logic encryption method.

The transmission gate topology shown in Fig. 5 implements either an AND or a NAND gate depending on the values of KEY0 and KEY1. When KEY0 is 0 and KEY1 is 1, the topology behaves as an AND gate. When KEY0 is 1 and KEY1 is 0, the topology behaves as a NAND gate. Note that if KEY0 and KEY1 are 0 the AND/NAND topology produces a constant low on OUT. Similarly, a constant high output is achieved when KEY0 and KEY1 are 1. The use cases for the two constant outputs are described in Section 5.

The transmission gate topology is modified slightly to implement an encrypted OR/NOR. For the AND/NAND topology, if logic low is applied to either input A or B the value of KEY0 is passed to OUT. Similarly, when inputs A and B are both logic high, the value of KEY1 is passed to OUT. An OR/NOR gate encryption is achieved by passing the value of KEY0 when input A or B is logic high, and the value of KEY1 when inputs A and B are both low.

While the implementation of a NAND, AND, OR, or NOR gate with transmission gate logic provides simplifications to reduce the power, performance, and area overhead, encrypting an XOR or XNOR gate does not include the same logic simplifications. The only simplification that is possible for the XOR or XNOR gates is to eliminate one of the key inputs by tying the KEY1 and KEY2 inputs in Fig. 2 together.

The reduction in the number of key values is beneficial when routing congestion is a concern.

5. KEY EXPANSION

The transmission gate topology shown in Fig. 5 has the ability to output a constant logic low or a constant logic high independent of the inputs A and B. The number of key combinations an adversary must consider effectively doubles when utilizing the constant outputs of the transmission gate topology, while still achieving the reduction in overhead provided by gate level logic encryption. The ability to utilize the constant outputs to expand the key search space of an adversary is coined key expansion. Circuit topologies that utilize the constant low and constant high output of the transmission gate logic encryption family to force an adversary to consider a larger key search space are provided in this section.

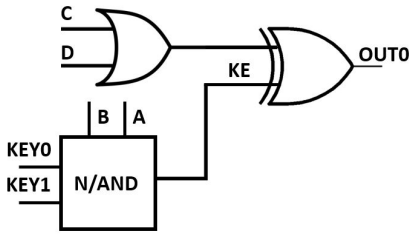

Figure 6: Key expansion utilizing net KE as a standard key input.

5.1 Key Expansion Topologies

Key expansion is successfully included into a circuit when the expanded key appears to be a standard case of logic encryption. As a result, one encrypted gate must not be directly connected to the key input of another encrypted gate. The first topology that encrypts the input keys is shown in Fig. 6, where the net labeled KE represents the key expanded net. For the topology shown in Fig. 6, A and B are inputted from nearby nets, providing the appearance that the encrypted gate is a NAND or AND. However, when KEY0 is equal to KEY1, a constant value is generated on KE, essentially replicating the XOR based encryption technique. The topology shown in Fig. 6 also functions when the XOR gate is replaced by another standard gate such as an AND or OR. For the case that the XOR is replaced with an OR, if a controlling value is generated on KE, then OUT0 is a corrupted version of the output of the OR gate.

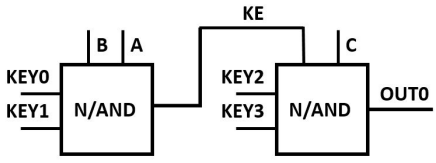

Figure 7: Key expansion topology with two NAND/AND gate level logic encryption gates.

Another topology that allows for key expansion is shown in Fig. 7. The output of a gate level logic encryption instance is connected to one of the logical inputs of another encrypted instance. The topology appears to be a standard logical connection within the circuit, however, key expansion is implemented by holding the KE net constant. When KE is held at a constant high, OUT0 is either C, the inversion of C, a constant 0, or a constant 1 depending on the values of KEY2 and KEY3. The AND/NAND structure no longer functions as an AND/NAND gate, forcing an adversary to explore more key input combinations.

The topology shown in Fig. 8 is an expanded use case of the topology in Fig. 7. Instead of directly connecting the output of one encrypted gate to another, a controllable path within the circuit is used to disguise the use of key expansion. The modified circuit makes it more difficult for an adversary to associate the given topology with key expansion.

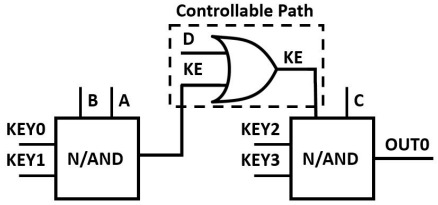

Figure 8: Controllable path insertion to mask key expansion.

While the presented key expansion topologies present opportunities to utilize the constant output cases of the transmission gate logic encryption, the overheads of the topologies are larger than a standard encrypted gate. As such, the ideal circuit candidate is one that already includes the budget for the implementation of the XOR or LUT based logic encryption techniques. For example, the c432 ISCAS benchmark implemented with the fault-based XOR logic encryption technique has a performance overhead of approximately 10%, a power overhead of 55%, and an area overhead of 57% [11]. Although the impact on performance requires further investigation, the use of gate level logic encryption reduces the power overhead to an estimated 32.3% and the area overhead to 45.7% for the same implementation. The 22.7% difference in the power overhead and 11.3% difference in the area overhead are available for key expansion. The reduced overhead of gate level logic encryption permits the addition of key expansion to further increase the key combinations an adversary must search while maintaining a similar overhead budget to current methodologies.

6. EVALUATION OF POWER, AREA, AND PERFORMANCE

The improvements in the power, propagation delay, and area for both the stack and transmission gate topologies are described in this section. As with the simulation described in Section 3, the drive strengths of the encrypted cells were matched with those of the XOR and LUT based logic encryption techniques. In order to match the drive strength, an inverter was added to the output of both the stack and transmission gate topologies. The functionality of the stack-based design is therefore an AND/OR gate as opposed to a NAND/NOR. The improvement in performance is described as a reduction in propagation delay. An analysis of the average power and average leakage power is provided for each gate. The area improvement is based on an area estimate determined from a layout of the encrypted gates as compared with layouts of the XOR and LUT encryption

Table 6: Propagation delay, power, and area analysis of the AND/OR implemented with stack-based encryption. Percent improvements over XOR and LUT logic encryption are listed.

Comp.	Standard Cell	Prop. Delay (ps)	Power (nW)	Area (μm^2)
XOR	AND	119.8 (20.82%)	80.72 (46.33%)	41.62 (34.77%)
	OR	116.7 (25.90%)	79.87 (54.26%)	
LUT	AND	119.8 (2.204%)	80.72 (44.71%)	41.62 (54.05%)
	OR	116.7 (3.394%)	79.87 (43.83%)	

methods. The improvements over the XOR and LUT based encryption techniques are listed in Tables 6, 7, and 8 for, respectively, an AND/OR, NAND/AND, and NOR/OR. The average leakage power of each of the topologies and the percentage improvement over the XOR based technique is listed in Table 9. A comparison with the LUT based technique results in larger improvements.

Table 7: Propagation delay, power, and area analysis of the NAND/AND implemented with transmission gate encryption. Percent improvements over XOR and LUT logic encryption are listed.

Comp.	Standard Cell	Prop. Delay (ps)	Power (nW)	Area (μm^2)
XOR	AND	80.19 (46.99%)	85.41 (43.21%)	51.10 (19.81%)
	NAND	98.54 (22.77%)	101.0 (28.97%)	
LUT	AND	80.19 (34.54%)	85.41 (41.50%)	51.10 (43.58%)
	NAND	98.54 (20.91%)	101.0 (35.67%)	

The results demonstrate a substantial reduction in the performance, power, and area overheads associated with encrypting a gate. The reductions allow for an expanded key search space to further deter adversaries without increasing the overheads of current logic encryption methods. In addition, the lower per-gate encryption overheads allow for the placement of encrypted gates closer to the critical timing path within a circuit.

Table 8: Propagation delay, power, and area analysis of the NOR/OR implemented with transmission gate encryption. Percent improvements over XOR and LUT logic encryption are listed.

Comp.	Standard Cell	Prop. Delay (ps)	Power (nW)	Area (μm^2)
XOR	OR	86.99 (44.77%)	91.44 (47.63%)	51.10 (19.81%)
	NOR	96.77 (27.94%)	88.00 (47.78%)	
LUT	OR	86.99 (27.99%)	91.44 (35.70%)	51.10 (43.58%)
	NOR	96.77 (23.68%)	88.00 (44.58%)	

7. CONCLUSIONS

A methodology to insert key control into a gate, referred to as gate level logic encryption, is described. Two gate level logic encryption topologies were described in the paper. Substantial reductions in the per-gate encryption overhead were achieved by utilizing gate level logic encryption. As an example, encrypting an AND gate resulted in a power reduction of 41.50%, an estimated area reduction of 43.58%, and a performance increase of 34.54% over the LUT based encryption technique. In addition, a methodology to expand the key search space of an adversary when utilizing gate level logic encryption (key expansion) was introduced. The techniques provided in the paper demonstrate the importance of considering security when designing a library of logic gates.

Table 9: Analysis of the average leakage power of the stack-based AND/OR and transmission gate based AND/NAND and OR/NOR topologies. Percent improvement over the XOR based topology are listed.

Standard Cell	Stack (pW)	Trans. Gate (pW)
AND	226.8 (66.34%)	405.1 (39.88%)
NAND	-	371.6 (36.04%)
OR	181.2 (68.81%)	528.2 (20.15%)
NOR	-	494.6 (23.78%)

8. REFERENCES

[1] DigiTimes, "Trends in the global IC design service market," www.digitimes.com/news/a20120313RS400.html?chid=2, March 2012.

[2] R. Torrance and D. James, "The State-of-the-Art in Semiconductor Reverse Engineering," *Proceedings of the IEEE Design Automation Conference*, pp. 333 – 338, June 2011.

[3] International Chamber of Commerce, "Impacts of counterfeiting and piracy to reach US $1.7 trillion by 2015," http://www.iccwbo.org/News/Articles/2011/Impacts-of-counterfeiting-and-piracy-to-reach-US$1-7-trillion-by-2015/, Feb. 2011.

[4] Semiconductor Equipment and Materials Industry, "Intellectual Property (IP) Challenges and Concerns of the Semiconductor Equipment and Materials Industry," http://www.semi.org/cms/groups/public/documents/web_content/p043701.pdf, April 2008.

[5] M. Tehranipoor and F. Koushanfar, "A Survey of Hardware Trojan Taxonomy and Detection," *IEEE Design and Test of Computers*, Vol. 27, No. 1, pp. 10 – 25, Feb. 2010.

[6] S. Chakraborty and S. Bhunia, "HARPOON: An Obfuscation-Based SoC Design Methodology for Hardware Protection," *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems*, Vol. 28, No. 10, pp. 1493 – 1502, Oct. 2009.

[7] J. Roy, F. Koushanfar, and I. Markov, "EPIC: Ending Piracy of Integrated Circuits," *Proceedings of the IEEE/ACM Design, Automation and Test in Europe*, pp. 1069 – 1074, Oct. 2008.

[8] Rajendran, J. and Sinanoglu, O. and Karri, R., "Regaining Trust in VLSI Design: Design-for-Trust Techniques," *Proceedings of the IEEE*, Vol. 102, No. 8, pp. 1266 – 1282, July 2014.

[9] Y. Alkabani and K. Farinaz, "Active Hardware Metering for Intellectual Property protection and Security," *Proceedings of the USENIX Security Symposium*, pp. 291 – 306, Aug. 2007.

[10] Rajendran, J. and Pino, Y. and Sinanoglu, O. and Karri, R., "Logic Encryption: A Fault Analysis Perspective," *Proceedings of the IEEE/ACM Design, Automation and Test in Europe*, pp. 953 – 958, Oct. 2012.

[11] Rajendran, J. and Zhang, H. and Zhang, C. and Rose, G. and Pino, Y. and Sinanoglu, O. and Karri, R., "Fault Analysis-Based Logic Encryption," *IEEE Transactions on Computers*, Vol. 64, No. 2, pp. 410 – 424, Feb. 2015.

[12] A. Baumgarten, A. Tyagi, and J. Zambreno, "Preventing IC Piracy Using Reconfigurable Logic Barriers," *IEEE Design and Test of Computers*, Vol. 27, No. 1, pp. 66 – 75, Feb. 2010.