IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received June 22, 2021, accepted July 9, 2021, date of publication July 15, 2021, date of current version July 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3097348

A Secure Scan Architecture Protecting Scan Test and Scan Dump Using Skew-Based Lock and Key

HYUNGIL WOO, SEOKJUN JANG, AND SUNGHO KANG, (Senior Member, IEEE)
Department of Electrical and Electronic Engineering, Yonsei University, Seoul 03722, South Korea

Corresponding author: Sungho Kang (shkang@yonsei.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2019R1A2C3011079).

ABSTRACT Scan-based Design for Testability (DFT) is widely used in industry as it consistently provides high fault coverage. However, scan-based DFT is prone to security vulnerabilities where attackers use the scan design to obtain secret information from the system-on-chip. Existing countermeasures for such attacks contribute to enhancing the security of the scan design but cannot prevent the loss of debuggability because scan dumps are also treated as a type of attack even though they provide debuggers with high observability after a chip has been packaged. Therefore, it is necessary for secure scan architectures to effectively defend various existing attacks without losing the debuggability of the scan dump. In this paper, we propose a secure scan architecture using a skew-based lock and key to enhance the security of the scan design while maintaining the debuggability of the scan dump. The proposed architecture builds up an invisible barrier against the attacker by combining the physical information into a lock and key scheme. While the security is improved effectively with the new barrier by only a few key flip-flops, the scan dump, which has been treated as an attack, is protected separately in the acquisition and analysis steps using secure software in the proposed architecture. Performance evaluations show that the area and the test time overhead in the proposed architecture are negligible while achieving both a high level of security and scan dump protection.

INDEX TERMS Secure scan architecture, secure scan dump, skew-based lock and key, lock and key.

I. INTRODUCTION

Scan architecture is a common and dominant Design for Testability (DFT) methodology in modern system-on-chip (SOC) devices as it is beneficial for obtaining high test coverage [1]. The controllability and observability provided by the scan design facilitate high-quality testing of chips to discover manufacturing defects. In addition to weeding out faulty chips, the scan design can be utilized in the form of a debugging technique called ''scan dump'' to determine the root cause of abnormal behavior in mission mode by observing the internal state of the entire circuit [2]–[7]. Upon attaining the desired condition, the functional clocks are stopped and registers in the scan chains are set to scan-shift mode. By toggling the scan clock, the internal states can be shifted out as binary data to provide ''pseudo'' run-time observability. Testing and debugging SOCs have far less visibility in the field; hence, these abilities provided by the scan architecture are important.

Utilizing the observability and controllability advantages of the scan design implies that the hardware structure may be exposed, so the algorithms for intellectual property and data being processed, which are confidential, must be protected against malicious attackers. Mode-switching attack is one method used by attackers to obtain internal register values [8]–[10]; here, attackers can obtain the data being processed by switching the chip from function mode to test mode. This type of attack assumes that mode switching is available to attackers and that the registers retain their values during mode switching. Thus, the internal circuit structure or confidential data may be leaked by illegally extracting scan data. Another attack method that does not require mode switching, called ''test-mode-only attack,'' obtains information in the test mode [11]–[13] where the response of the combinational logic is obtained by scan capture after shifting the input vector to the scan chain. Then, attackers can obtain and analyze response data to identify the relationship between shifted input data and the output result.

To prevent such attacks, several countermeasures have been proposed in the literature. The simplest approach

The associate editor coordinating the review of this manuscript and approving it for publication was Sedat Akleylek.

involves unbounding the test interface permanently after performing manufacturing tests to block all further test access. To protect chips against nonintrusive attacks targeting the data in operation, the mode-reset countermeasure [14] resets the chip when its mode changes from function mode to test mode. To block the transfer of secure key information on scan data, a solution using shadow registers called mirror key registers (MKRs) was introduced [15]. A secure DFT architecture in [16] allows the test only after the system reset, and it hides the key in the test mode. To thwart the attack through the scan design, in [17] and [18], the authors proposed a lock and key technique to assess authentication for a scan test. In [19], the authors proposed inserting dummy flip-flops in the scan chain to locate the input key for the lock and key technique. In [20], the authors used primary inputs of the cryptographic chip as a test password to protect the scan design. To avoid the vulnerability of a common key, a secure scan design that adopts physically unclonable function (PUF) was proposed, which generates a unique key for each chip [21]. Scan chain encryption was proposed in [22] using ciphers on the test interface, and the scan design in [23] utilized the hash module to make the test response irreversible.

In lock and key schemes for secure manufacturing tests, longer keys enhanced the security of the lock in most cases; however, extending the bit width of the key incurs an additional cost because of the additional area required to embed the related logic in the chip. Further, some of the countermeasures using key-based authentication are weak against existing attacks. Countermeasures that are as yet unbroken can also be vulnerable if new attack mechanisms can break known lock and key schemes. That is why security architectures should consider both the efficiency of the key and the securing technique. At the same time, protection schemes should cover not only attacks in the test mode but also mode-switching attacks while allowing scan dumps to authorized engineers for debugging. Furthermore, when abnormal behaviors in the function mode are debugged, the scan dump must preserve data in the flip-flops as close as possible to the state at which operations were halted to maintain precision for analysis [24], [25]. These are critical and difficult requirements for the scan dump because functional operations are executed at very fast asynchronous clocks in modern SOCs that can even run multiple operating systems simultaneously. Therefore, it is necessary to consider the characteristics of a scan dump for secure scan architecture to obtain maximum observability and precision while protecting content against illegal extraction or analysis.

If there is a new barrier that is invisible, the security of scan design can be improved. When the invisible barrier has a high security level, the attacker must find the method to overcome the barrier even the barrier becomes visible. While the security is improved by the new barrier, the debuggability inside the barrier has to be maintained and must be allowed to the authorized engineers only. In this paper, we aim to propose a new architecture that can build a new barrier against the attacker while scan dump is allowed only for authorized engineers. Of course, all of these should be implemented with minimal overheads in terms of area and testability.

To achieve the goals for the above motivations, we propose a secure scan architecture using skew-based lock and key (SLAKE) that can protect both the scan test and scan dump from malicious access while maintaining the debuggability of the scan dump. The main contribution of this paper is an introduction of a new scheme that can protect the scan design using the skew implementation which is an invisible barrier to attackers. Another main contribution is that scan dump, which has been recognized as an attack, is defined as a target to be protected. The scan dump in the proposed architecture is allowed to authorized engineers only, so the security is enhanced while the debuggability is maintained.

The remainder of this paper is organized as follows. Section 2 provides some relevant background for this study such as countermeasures against scan-based attacks and scan dump. Section 3 presents the proposed SLAKE and secure scan dump. Section 4 describes the proposed secure scan architecture in detail. Section 5 explains the scan dump protection using secure software. Section 6 presents some performance evaluations for the proposed secure scan architecture. Finally, Section 7 summarizes the conclusions of this study.

II. PREVIOUS WORKS

A. PREVIOUS COUNTERMEASURES

Since scan-based attacks are known and available, many countermeasures have been proposed to thwart scan-based attacks.

A mode-reset countermeasure was proposed [14] to prevent mode-switching attacks by clearing sensitive information from the scan chain; while this simple-yet-effective method can prevent information exposure, test-mode-only attacks avoid this countermeasure, and this scheme renders scan dumps impossible.

In [15], the authors proposed the architecture that can block the leakage of the secret key based on its security state. In this architecture, the key value can be loaded to MKRs only if the chip is in the secure mode. The chip can enter the secure mode by the verified software in function mode. Once the chip is in secure mode, it cannot return to the insecure mode before power-on reset. Therefore, the key value does not appear on the scan chain in the test mode. The secure DFT architecture reported in [16] also isolates the encryption key in the test mode with a smaller area cost. The mode switching from the functional mode to the test mode is forbidden, and the mendacious key is propagated to the scan chain if the chip does not start in the test mode after reset. However, MKR hides only the key registers and cannot launch scan dump to debug the chip in the secure mode. As the architecture in [16] does not allow the mode change from the function mode to the test mode at all, utilizing scan dump for debugging is not available.

In [17] and [18], a lock and key technique was proposed to prevent scan-based attacks by authentication. The test security controller (TSC) and linear-feedback shift register (LFSR) were introduced to compare test inputs to an embedded golden key and scramble the chain order. If the input key to the TSC is correct, the decoder in TSC generates a one-hot selection signal to choose a scan chain to be shifted in/out. Otherwise, the LFSR scrambles the order of scan chains unpredictably; however, it is possible for a scrambling technique to leak data while the scan chain is available [26], [27]. The same authors proposed integrating the input key into the scan test pattern as a low-cost solution [19]; however, modifying the scan chain is required to insert dummy flip-flops for the key bits in the test pattern, and this scheme is vulnerable to bit-role identification attacks [28] which determines the location of the key by observing the difference on the response for bit-flipped test patterns.

In [29], the authors proposed the countermeasure obfuscating Hamming weight against the differential attack. Since the differential attack observes Hamming weight to figure out the round key from AES, this methodology masked the response to give incorrect Hamming weights. The pre-computed masks for available input states stored in a tamper-proof memory are loaded by the controller to give corresponding incorrect Hamming weight. However, this methodology only focused on the differential attack on AES, and it can be vulnerable to other attacks (e.g., flushing attack without the capture operation to determine the masking locations).

Test key randomization using an LFSR was proposed in [30] where the LFSR is embedded to generate the authentication key. The input key in the dummy flip-flops or the ‘‘don’t care’’ bits of the test pattern are compared with the key generated by the LFSR. Without an input key generated by off-chip simulation, test responses are randomized. However, this scheme is known to be weak at bit-role identification and randomizes scan dump data.

A static obfuscation scheme using a shift register was introduced in [31], where the scheme overrode the test controls for some modified scan cells if the input key was incorrect. The test-mode-only signature attack, as discussed in [32] by the same authors, can identify misconfigured scan cells in the static obfuscation scheme in [31]. The attack assumed that the cell blocking the scan chain can be determined by correcting and analyzing the response from the crypto chip with the simulation result. To avoid the weak point that came from the static obfuscation, the authors in [32] also proposed the dynamic obfuscation scheme. The scheme defended against signature attacks by performing a cyclic permutation of the shift register. However, the schemes in both [31] and [32] could not protect the scan dump against the unauthorized access from tester. The architecture proposed in [28] also used the dynamic obfuscation without the authentication using the input key. Instead, the architecture integrated the LFSR to generate the obfuscation key. The seed of the LFSR was delivered from the control vector in non-volatile memory (NVM) by read-only direct memory access. This method does not have any test time overhead since the authentication is not required, however, the scan dump is not available in this architecture because the scan out data is forced to zero for the first pattern to avoid the differential attack.

The PUF-based lock and key solution introduced in [21] generated a unique key for each chip that was taped out to avoid security neutralization due to a leaked common key. When the user decides to activate the lock, the response of the PUF is stored in the OTP for the reliability. Then, the obfuscation logic is activated differently for each chip because the golden key is generated using the hashed PUF response and the permuted PUF response stored in the OTP. After the lock and key is activated, the tester must input the key into the chain to unlock the scan design for in-field tests. In [33], the use of symmetric fuse read only memories (ROM) was proposed as it considered differential power analysis [34], [35] targeting the PUF key stored in the memory; however, once the chip-unique PUF key was delivered to the tester, it was always possible for the tester to obtain the scan dump data even the confidential software was running on the chip.

The scan architecture proposed in [20] receives the test password consisting of the load password and the scan password using primary inputs of the cryptography chip. If the load password is correct, the scan password can be loaded into the cyclic shift register (CSR). If the scan password is also correct, the test can be performed without any obfuscation. If any of the passwords is wrong, then the clock for the CSR keeps enabled to obfuscate the scan chain using the output value of the CSR. The test time overhead is only two cycles to check the password coming through primary inputs, however, utilizing IO can be limited if the chip loads the plaintext in a different way. The scan dump data can be obtained without any loss in this architecture, but unauthorized testers who know the password for the test can access the chip and obtain the scan dump data.

In [22], the authors proposed scan chain encryption using dedicated cipher blocks for the test interface. When the encrypted test data were shifted into the chip, decryption was achieved by an on-chip cipher and the test response was shifted out after being re-encrypted by an on-chip cipher; however, this procedure requires an additional fast clock for the ciphers and two on-chip ciphers running in the test time, which should be avoided in the scan dump situation. Even though the test response is encrypted, the tester can do off-chip decryption for this scheme, this solution cannot protect the scan dump data against the access from unauthorized engineers.

An access authorization protocol based on challenge-response authentication was proposed in [36], where the tester must obtain the challenge from the chip and request an answer from the server. The challenge is a random number produced by the random number generator in the chip, and the corresponding response can be obtained from the hashed result by the server. Shifting the server’s response value to the test interface unlocks the target test instruments when the response sent back to the chip is valid. Such authentication by access to the server is effective for providing authentication to testers but this requires a functional operation that can affect the scan dump results.

In [23], the authors proposed to utilize the hash module without the lock and key. In this approach, the test response was always hashed by the hash module connected through WISHBONE protocol. Since the test response hashed responses are irreversible, the architecture is secure against the attacks using the test results. However, the scan dump is not possible in this architecture because the dictionary to analyze the scan dump data cannot be made for all cases.

B. SCAN DUMP

As the SOC complexity increases, the possibility of a design bug increases and may be unavoidable. In SOC debugging, the scan dump is an essential methodology for obtaining a snapshot of the internal state at a specific point in time [24]. Bugs can be simple design mistakes at the intellectual property (IP) level, architectural bugs at the system level, or specification bugs between the hardware and software. Moreover, bugs can be wear-out failures of certain wires or cells that did not exist at the time of manufacturing tests. Depending on the severity, a chip revision or software workaround which is time-consuming and costly may be required to fix bugs in the produced chip. In this case, the later the bug is analyzed, the higher the repair cost. Unlike register-transfer level (RTL) or netlist level debugging at the development stage, packaged SOCs provide almost no visibility for the functional behaviors of the internal logics [37]; the only method of recognizing such problems is via system hang or unexpected behaviors. It is difficult to analyze the internal state using only information from the design-for-debug (DFD) logic because DFD also has limited scope due to its cost. In such a situation, the scan dump's uniqueness and utility are important; by shifting out almost all register values in the SOC, a designer can analyze the state of IPs, system configuration, the state of buffer in the system bus, or even a program counter and general-purpose registers in the processors. With the abovementioned indicators, designers can localize bugs, which helpfully narrows the scope for debugging such that debugging can be performed in RTL or netlist simulations. Without such localization, bug reproduction in a simulator is almost impossible since the SOC usually performs multiple functions simultaneously (e.g., image processing, display control, and machine learning).

To obtain scan dump data, the system should freeze itself by gating all clocks and holding power-related signals to drive appropriate values, including primary inputs and outputs. Thereafter, by toggling the scan clock with scan-enable (SE) asserted, the internal state of the entire scan-stitched flip-flops can be shifted out. Some of the logic—such as the control logic for the abovementioned signals—cannot be a target of a scan dump. With the exception of the essential part of the scan dump operation, minimizing the use of logic maximizes the observability of the scan dump.

From a security perspective, the scan dump itself presents a vulnerability because it can expose confidential data in SOCs. For example, if the boot code in the embedded ROM includes a signature to verify the next-level software code during a secure boot [38], [39], the signature value can be placed in the booting processor—such as in the general-purpose register—at the time of signature comparison. If the booting processor is stitched to the scan chain for high test coverage, an attacker can stop the system and extract data from the scan chain. Even though the scan dump could be exploited by attackers, it must be maintained to provide the observability of the flip-flop levels while debugging packaged or installed chips operating applications. Due to this debuggability provided by the scan dump, it is not possible to calculate the additional cost for debugging in its absence. In addition to the execution of a scan dump, scan dump data analysis should be protected. Since the installed chip requiring a scan dump can be located anywhere depending on its application, obtaining scan dump data should be possible for field service engineers even without permission to analyze such results. Otherwise, it may be necessary to deliver the chip or the device to an authorized engineer with the risk of failure of bug reproduction or to let an authorized engineer debug the system on-site; both these options are undesirable as they increase debugging time and cost. Therefore, scan dump data analysis must be secured separately from data acquisition.

III. PROPOSED IDEA

A. SECURITY THREATS FOR SCAN DESIGNS IN SOCs

For scan designs implemented in SOCs, attackers can challenge the security at various stages of the SOC supply chain. Fig. 1 illustrates the possible threats in the stages from manufacture to market. The SOC is first tested at both the wafer and package levels; at this stage, malicious access by an unauthorized engineer in the foundry or test facility is possible. After screening for failures in the SOCs, they are mounted on boards and programmed to assess functionality. Functional defects are identified with functional test patterns; In addition to functional tests, software development and validation can also be done in this stage. Therefore, both testing and debugging could be solutions to identify the root causes of abnormal behaviors of SOCs at this stage. Since SOCs can operate in function mode, confidential information can be accessed on the scan-stitched circuit. This information may be exposed to attackers who can control the scan chain. After software development, SOCs are installed in products for the end customers. When products equipped with SOCs are sold in the market, the sensitive information stored in them can be illegally extracted through scan access by malicious end customers (or hackers). During the lifetime of an SOC installed in a product, wear-out failure and performance degradation by aging are possible and can be identified by in-field tests. In addition, bugs that were not detected at either the verification or validation stages could be found in products because devices could be exposed to an almost infinite number of use conditions. To identify and solve failures by in-field tests

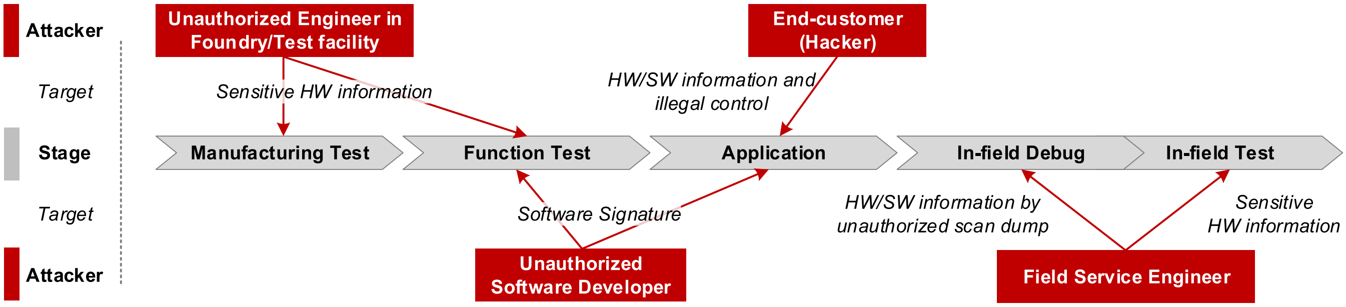

FIGURE 1. **Possible security threats through scan design in the SOC supply chain.**

and scan dumps, scan access should be allowed for field service engineers. While scan tests are available in remote sites to the field service engineers, confidential information can also be accessed and analyzed by engineers who lack the required authorization. Nonetheless, scan dumps should be available at remote sites because bugs could be affected by use environments that cannot be reproduced in other places. Therefore, considering both in-field tests and the security of the scan dumps at a remote site is required.

B. SECURE SCAN ARCHITECTURE CONSIDERING THE SCAN DUMP

The situation of debugging a malfunction in an SOC implies that the SOC is programmed to operate in function mode. Therefore, scan dump utilization during debugging requires loading appropriate software for bug reproduction, which must be available to field service engineers at remote sites. However, as mentioned above, the contents of scan dumps need to be protected from malicious access because it may contain confidential data in function mode. To allow field test engineers to use a scan dump while protecting its contents from malicious use, we propose launching a scan dump using only secure software that was verified during the secure boot process. We assume that the secure boot process established by the hardware-validated root of trust [40] ensures the integrity and the security of firmware in the proposed architecture. As verification of the included signature proves its integrity, secure software can be used to configure a security-controlled area that is only accessible to privileged software in the system. Fig. 2 illustrates a simplified example of the flow for a scan dump using the secure software configuration. After the code has been verified and allowed to access the secure hardware resources in the SOC, it can control the registers related to the scan dump in the system; the scan dump configuration includes activating the scan dump trigger (unmasking), the reordering configuration for obfuscation, and setting the trigger condition of the scan dump. After the configuration, the SOC operates as programmed. When a bug is reproduced and the scan dump mode is triggered, the system works in the scan dump mode to preserve the internal state; thereafter, the value in each flip-flop is obtained by shifting out the data in the entire chip. To read this data correctly, the configuration by the secure software for the scan dump must be known. With information about the obfuscation scheme used in the scan dump, the scan data can be parsed; then, by mapping the data to the list of the scan chain order, a designer can obtain information about the internal state at the moment of bug reproduction. With this observability afforded by the scan dump, bug analysis and localization become available. Since the scan dump in this architecture cannot be triggered without the configuration, help is required from an authorized engineer who can create privileged secure software. Once the software including the scan dump configuration has been released, field service engineers can obtain the scan dump by executing this software code; however, the data obtained from the scan dump are obfuscated, so it is impossible to analyze the data without knowledge of the configuration and obfuscation scheme. Moreover, the values of the scan dump data obtained using the same software can differ each time due to the fast functional operating frequency. Therefore, it is not possible to infer the original value from the obtained results before identifying whether the difference arises from obfuscation or some change in the internal state.

C. SKEW-BASED LOCK AND KEY

While the scan dump is protected by the secure software's configuration, attacks through scan tests remain possible. Since a test-mode-only attack can occur without loading any software, it is necessary to deploy guards such as the lock and key scheme to prevent attacks through the test interface. In the lock and key scheme, the input key is captured in designated flip-flops. Therefore, the existing secure scan architectures and attacks assume that the key is correctly captured in the flip-flops. Irrespective of the architecture having key flip-flops (KFFs) separated from the original scan chain or using the designated points in the original scan chain, the value inputted through the SI at a certain moment is expected to be stored in a single flip-flop. To improve the security of the scan design by adding a new factor, we propose SLAKE in our secure scan architecture. When scan design is implemented, the destination of SI is the input of a single flip-flop. In SLAKE, the SI is connected to multiple KFFs with an intentional skew between each path. If the paths from the SI to the KFFs have different latencies on the data

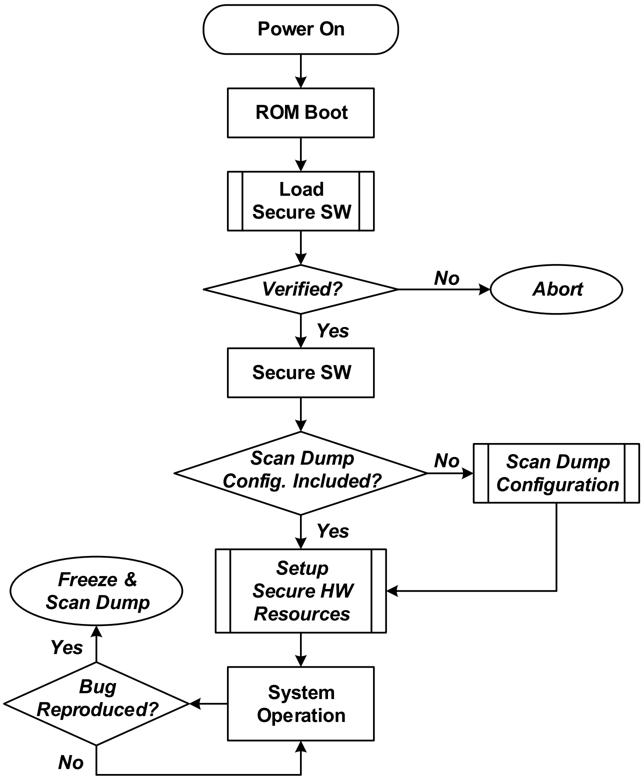

FIGURE 2. Example scan dump configuration flow by secure software.

paths due to intentional skew, the values captured at the KFFs are determined according to the timing of the clock edge. To capture the desired value, the implemented latency on each path and timing information are required.

Fig. 3 shows an example of a SLAKE with four-bit KFFs. If we assume that there is no clock skew between the flip-flops and that each buffer has the same delay, the transition of the input SI arrives in each KFF at a different time. As a result, the value captured between the rising of D0 and the falling of D3 can have various value sets depending on the clocking time. Therefore, the key input can be parallelized without additional primary inputs. After the values are captured by toggling TCK, they are compared with the golden key. Without knowledge of the use of this scheme, an attacker cannot unlock the test logic, so there is nothing an attacker can do with existing attack methods. Even if SLAKE is exposed to the attacker, the implemented skew value is required to capture the desired value. Therefore, the knowledge of the scheme, implemented timing information, and golden key value are required to unlock the secure scan architecture with this scheme. Unless all information of SLAKE are leaked, unlocking is impossible. Moreover, comparing the key at multiple stages increases the complexity of the key exponentially. Therefore, the complexity of SLAKE with n KFFs and m stages is $2^{(n\times m)} \times (C_{Ps}^{Cs})^{m}$, where Ps is the total possible skew between the clock and data and Cs is the correct skew to capture the desired value. Since Ps is infinite, SLAKE can be made complex only with a small number of KFFs.

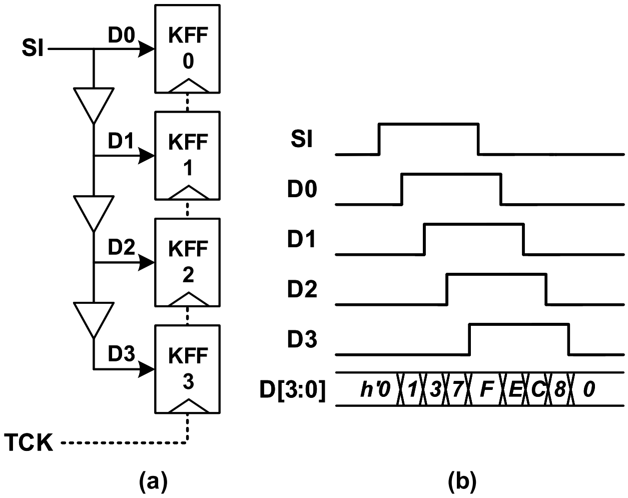

FIGURE 3. (a) Example of skew-based lock and key with four-bit KFF and (b) timing diagram.

In multistage SLAKE, the minimum delay between key capture is the latency of a single buffer chain and the maximum delay between key capturing is required when the same key should be captured consecutively. As this scheme can achieve high complexity with few KFFs, manual implementation of the skew would not be a problem for modern SOCs.

IV. PROPOSED SECURE SCAN ARCHITECTURE

A. ARCHITECTURE OVERVIEW

Fig. 4 illustrates the overall architecture. It is composed of a secure scan controller (SSC) controlling the overall sequence, secure key infusion unit (SKI), key comparator (KC), and scan-out remapper (SR). At the start of the test mode, the SSC selects SKI as the destination of the scan-in (SI) to check the key. The SKI circuit has intended skews on the paths from the input SI to the destination KFFs. Due to the skew between the SI and each KFF, the captured value in each KFF can be different for the same timeframe. The captured value in the SKI is compared with a golden key value in the KC. For a manufacturing test, the KC compares the input key with the hard-coded golden key. For an in-field test in which the golden key is changed by an authorized user who can write an OTP, the KC compares the input key with the new golden key selected by the value written in the OTP. If the input key is correct, scan access is allowed and the test through the scan chain operates correctly. If the input key is incorrect, the failed key information is delivered to the SR, which obfuscates the output data with dual shadow register sets and the LFSR seeded by the captured key value of the last stage.

The scan dump in the proposed architecture uses memory-mapped registers. An address that only secure software can access is used to store the configuration of the scan dump. When the scan dump mode is triggered by this secure register, the SSC commands the SR to reorder the scan output without input key comparison. Reordering is controlled by the configuration of the secure software.

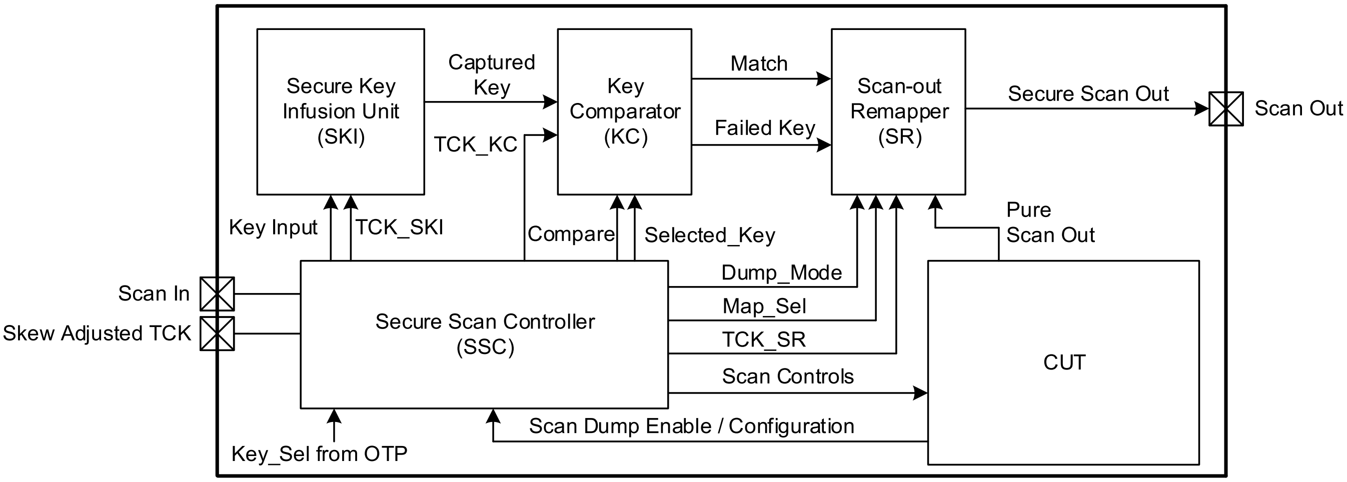

FIGURE 4. Overview of proposed secure scan architecture.

The main advantages of the proposed architecture are the security with the invisible barrier and the debuggability with the scan dump. In the SKI, the intentional skew which is a physical characteristic becomes a part of the key to unlock the scan design. The use of the SKI builds an invisible barrier against the attackers. While the scan design is protected with the new scheme, the scan dump is still available under the control of the SSC. For the scan dump launched by the secure software, the SR which can obfuscate the test response reorders the scan dump data based on the secure configuration. This reordering protects the contents from unauthorized testers who can launch and obtain the scan dump with secure software released by the authorized engineer. With the proposed architecture, these advantages can be achieved with negligible area cost and the test time overhead. The following sections describe each component in more detail.

B. SECURE KEY INFUSION UNIT

The development of the lock and key technique has mainly focused on the robustness of the golden key or on concealing the location to be filled with key input. The former could be achieved with a longer key and diversifying the golden key with dynamicity. The latter could be achieved by finding a new location for the key on the input test pattern. Instead of enhancing the characteristics in existing schemes, in this paper, we propose utilizing the physical delay on the path from the scan input to the KFFs. Including a physical delay in the secure scan architecture improves the security of the lock and key scheme because attackers are required to know not only the value of the key but also physical information.

To graft a physical characteristic onto the lock and key scheme, we propose placing intentional delay lines on the paths from the SI to the KFFs. Depending on the implemented delay, the change in SI appears differently at the input of each KFF. As a result, the captured values in KFFs are determined by the skew adjustment between the change of SI and clock edge. By adjusting the point of the clock edge based on the knowledge of the structure and implementation, it is possible to obtain results in various combinations. Fig. 5 illustrates an example SKI structure. The SI from SSC is connected to multiple KFFs; each path from the SI to KFFs has a different latency based on the designer's choice. The clock for the KFFs is a test clock (TCK) from the primary IO. Here, we assume that the clock skew between the clock paths is zero. Due to the intentional delay implemented on each path, the captured value is determined by the position of the TCK's rising edge. After the SI is captured by the KFFs, the captured values are delivered to the KC.

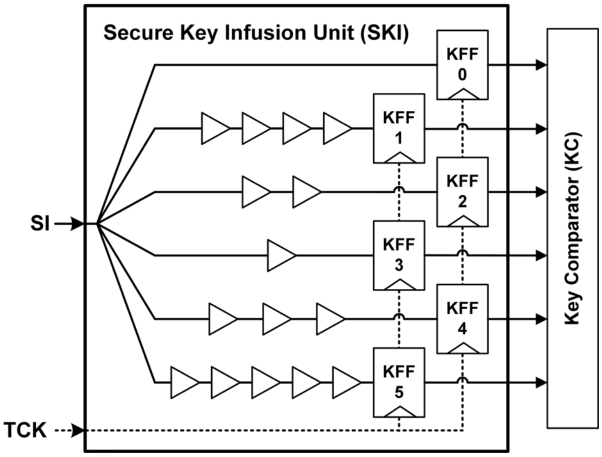

FIGURE 5. An example structure of the SKI.

As previously stated, the SSC selects the destination of the SI. When the test mode signal is asserted, the SSC selects SKI first as the destination of SI. Unless the clock is toggled as defined in the design, the SI and TCK are not transferred to the original scan chain. Therefore, it is possible to implement multistage SLAKE to evaluate the captured result several times. If SLAKE is designed to have multiple stages, the captured input key is delivered to the KC for each clock edge to check the correctness.

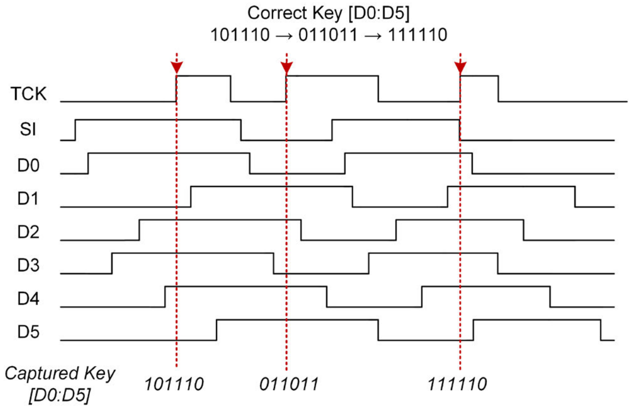

FIGURE 6. Timing diagram of the SKI operation.

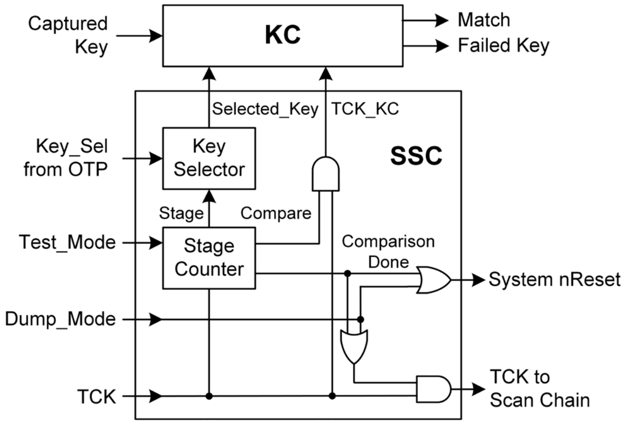

FIGURE 7. Clock and reset control in SSC.

Fig. 6 illustrates an example waveform. In this example, the key is six bits long and there are three stages. The amount of intentional delay cell is the same as in Fig. 5 and it is assumed that the delay for each cell is the same. As SI rises and falls, the inputs of the KFFs follow the transition with different delays. To capture the required key values, the rising edge of the TCK should be adjusted with consideration of the implemented latency. Hence, if the desired values to capture are "101110," "011011," and "111110" for each stage, then the first rising edge of the TCK must be between the rising of D1 and D4. Similarly, the TCK should rise between the falling of D3 and D2 to capture the second-stage key. The third-stage key should be captured before the falling of D0 but after the rising of D1. Therefore, if the TCK rises at the same positions with red dotted lines, the KFFs can capture the desired key values.

Enhancing the complexity of the security usually requires the lock and key technique to extend the length of the key. In SKI, it is possible to enhance the complexity in two ways: using additional KFFs and increasing the number of stages. Adding an extra stage increases the complexity by two to the power of the key length, whereas existing schemes have less effect with an additional key bit. If SKI has n KFFs, the combination of keys that can be obtained with m stages is $2^{(n\times m)}$.

C. SECURE SCAN CONTROLLER AND KEY COMPARATOR

The overall operation of the secure scan is organized by the SSC. Fig. 7 illustrates the clock and reset control for the KC and CUT in SSC. When the test mode is asserted from the external pin, a system reset is asserted to prevent scan-based side-channel attacks until the key comparison is complete. During key comparison, the TCK and SI remain connected to the SKI. If the architecture has m stages, the TCK for the SKI is gated after the first m cycles by the SSC to maintain the values. The golden key for each stage is selected in the key selector and delivered to the KC. While the scan data are defended against scan-based side-channel attacks, debugging requires preserving the snapshot at the scan dump mode. To preserve data in the scan dump mode, the system reset is not asserted if a mode change occurs from the scan dump trigger. Since the scan dump trigger can only be controlled by secure software, attackers cannot obtain the data being processed in the function mode.

Since it is possible for SLAKE to have multiple stages, the key selector delivers the golden key for each stage to KC as illustrated in Fig. 8. If the input keys for all stages are correct, the "Match" signal is asserted to high and remains activated until the next test mode assertion. This "Match" signal is delivered to SR with the last captured key value, so the test output is obfuscated according to these values if the input key for any stage is incorrect. In the manufacturing test, it is possible for the tester to infer the presence of SLAKE. If the tester knows the use of SLAKE, it is possible to unlock the chip with the initial test pattern even after manufacturing if the key is not changed. If the manufacturing tester is not a trustworthy entity, updating the key will be required. To update the key, we employ hard-coded key sets and the OTP embedded in the system. The hard-coded key sets are the candidates of the new key for each stage, and the OTP drives selection for key sets. Before being sold to the market, an authorized entity can write the OTP to select a set for the key among the hard-coded key sets. According to the pointer value driven by the OTP, SSC selects a key set to be compared in KC. By cascading the selection logic, it is possible to update the selection when updating the software released by the authorized developer as many times as the OTP assigned. Until it is updated to select another map by the written value in OTP, SSC keeps the selection as the default key set.

D. SCAN-OUT REMAPPER

If the input key is incorrect for the test mode, the output is obfuscated by SR. Fig. 9 illustrates the details of SR. SR is composed of a remap controller (RC) and dual shadow register sets. RC controls the SE and TCK of shadow registers. If the number of flip-flops in a single shadow register set is r, SE driven by RC alternates for each r cycle to select a shadow register set to be shifted out. When SE_A is high,

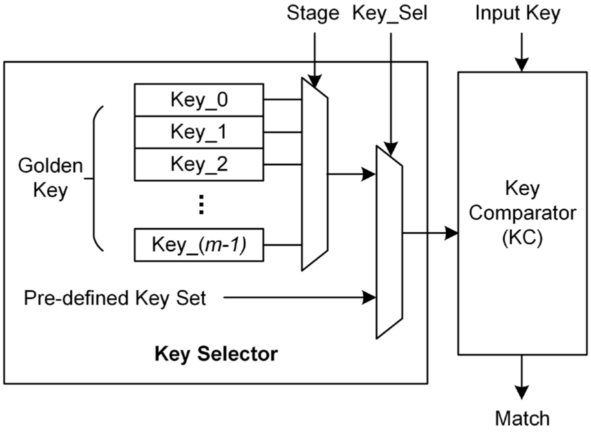

FIGURE 8. Key selection for multiple stages and updating the golden key.

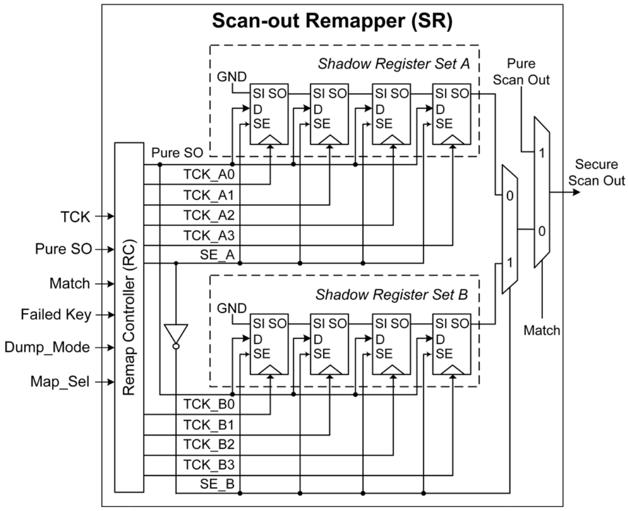

FIGURE 9. Output obfuscation using shadow register sets in SR.

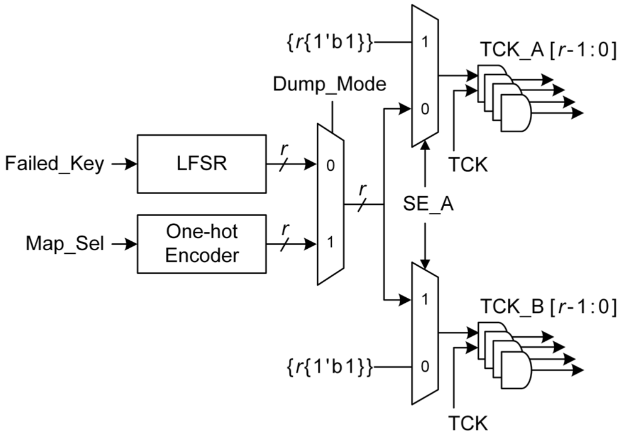

FIGURE 10. TCK control for shadow registers in RC.

the data in the shadow register set A are shifted out while the shadow register set B is updated. The update for each shadow register to capture the scan-out data from CUT is controlled by clock gating. The scan-out coming from the last flip-flop of the original scan chain is connected to the D pins of all shadow registers. Each clock for the flip-flops has its own clock gating signal. The clock gating signals are driven by LFSR seeded with the failed key. Depending on the LFSR state, pure SO data can be captured in one or more flip-flops. After r cycles, the SE signal for the updating set is activated to high and swaps the role with the shifting set. To shift out the scan data captured in the other set, the clock for the shifting set is not gated. Fig. 10 illustrates the clock gating control to obfuscate the output in RC. When the input key is incorrect in the test mode (i.e., "Dump_mode" is zero), SR obfuscates the scan data by updating the shadow register sets using the r-bit clock gating signal generated by LFSR. The clock gating signals for the shifting set are driven high to allow shifting out the captured data. By seeding LFSR using the failed key information, an attacker obtains the same result for the same test vector. Table 1 shows the example state of each signal in SR with four-bit shadow register sets. The first column is the cycle count from the beginning of the obfuscation, and the LFSR output starts from 4'b0011. The columns for TCK_$*$ denote the clock gating status. "-" and "En" represent that the clock is gated and enabled, respectively. For the first four cycles, SE_A is low and SE_B is high. Therefore, secure scan out is driven by SR_B[3:0] which is initialized to zero. While the data in SR_B are shifted out, SR_A is updated by the commonly connected input "pure SO". For the first capture operation, TCK_A1 and TCK_A0 are enabled because the LSB two bits of the LFSR are high. Therefore, the pure SO at that moment p0 is captured in SR_A[1] and SR_A[0] which are filled with red letters in the SR_A column. After set A is updated for four cycles, SE_B becomes zero, so set B becomes an updating set from the fifth cycle. The captured data in set A starts to be shifted out from its MSB. The positions highlighted with the blue letter are the next data for the secure SO. In the same manner with set A, set B is updated for four cycles. The sets swap their role for every four cycles. As the clocks can be enabled more than once in the updating cycles, the obfuscation in the test mode can be reordering, duplicating, or removing.

In scan dump mode, the information from KC is not used. The only information used by SR in this mode is "Map_Sel" signal driven by SSC to select the reordering scheme. This was originally driven by the memory-mapped secure register written by the authorized software. "Map_Sel" indicates the update order of all flip-flops in a set, and the one-hot encoder in RC controls corresponding clock gating signals based on the value of "Map_Sel". When r-bit shadow register sets are used, r different one-hot clock gating (enabling) is generated for every clock cycle using the one-hot encoder in SR. Therefore, the width of "Map_Sel" is $r \times \log_2 r$ to select which one to update among r registers during r cycles. By clocking only one register among the flip-flops in the updating set, only one selected flip-flop can capture the pure SO at that moment. Therefore, the data obfuscation in the scan dump mode does not have any loss of data. As long as "Map_Sel" is correctly

TABLE 1. The example of the obfuscation in SR in the test mode with four-bit shadow register sets.

Cycle count	Pure SO	LFSR output	SE _A	TCK _A3	TCK _A2	TCK _A1	TCK _A0	SR_A [3:0]	SE _B	TCK _B3	TCK _B2	TCK _B1	TCK _B0	SR_B [3:0]	Secure SO
0	p0	0011	0	-	-	En	En	X, X, p0, p0	1	En	En	En	En	4'b0	0
1	p1	0001	0	-	-	-	En	X, X, p0, p1	1	En	En	En	En	4'b0	0
2	p2	1000	0	En	-	-	-	p2, X, p0, p1	1	En	En	En	En	4'b0	0
3	p3	0100	0	-	En	-	-	p2, p3, p0, p1	1	En	En	En	En	4'b0	0
4	p4	0010	1	En	En	En	En	p3, p0, p1, 0	0	-	-	En	-	0, 0, p4, 0	p2
5	p5	1001	1	En	En	En	En	p0, p1, 0, 0	0	En	-	-	En	p5, 0, p4, p5	p3
6	p6	1100	1	En	En	En	En	p1, 0, 0, 0	0	En	En	-	-	p6, p6, p4, p5	p0
7	p7	0110	1	En	En	En	En	4’b0	0	-	En	En	-	p6, p7, p7, p5	p1
8	p8	1011	0	En	-	En	En	p8, 0, p8, p8	1	En	En	En	En	p7, p7, p5, 0	p6
9	p9	0101	0	-	En	-	En	p8, p9, p8, p9	1	En	En	En	En	p7, p5, 0, 0	p7
10	pa	1010	0	En	-	En	-	pa, p9, pa, p9	1	En	En	En	En	p5, 0, 0, 0	p7
11	pb	1101	0	En	En	-	En	pb, pb, pa, pb	1	En	En	En	En	4'b0	p5

configured to cover r cases, the reordered scan data can be restored with the value of “Map_Sel” and the structure of the implemented one-hot encoder. Table 2 shows the example of scan dump reordering in four-bit SR. The only difference with the obfuscation in the test mode is the source of the clock gating signals. In the example, “Map_Sel” is encoded with the priority to update each flip-flop. Since the MSB two bits of “Map_Sel” are 2’b00, SR_A[3] is the first flip-flop to be updated, followed by SR_A[0], SR_A[2], SR_A[1] in that order. The swapping of the role occurs as the same manner with that in the test mode. Because “Map_Sel” expresses the priorities of four flip-flops without duplication, all the data can be reordered without any loss.

V. SCAN DUMP PROTECTION

The massive increase in complexity for practical operation in SOC increases the difficulty of debugging for the SOC executing software after its validation. Since the root cause can be a design bug in the hardware, a coding bug in the software, or even a wear-out failure in-field, various root causes should be considered for bug localization and analysis. Moreover, reproducing the bug of installed SOC can be affected by the physical environment and its fast operating frequency can be an obstacle. The debuggability of a scan dump can be utilized to overcome these conditions.

Reproducing and debugging a bug by scan dump require modified software code. Modifying the software for a scan dump makes it possible to freeze the SOC as quickly as touching the trigger condition. For rarely reproduced bugs, it is also possible to accelerate bug reproduction by using modified software code. As the usage cases of SOCs diversify, bug reproduction can be affected by the product in which the SOC is installed or the environment on which it runs. Thus, field service engineers should be allowed to obtain scan dump data at remote sites to minimize the effect of these variables on bug reproduction. Nevertheless, the content obtained by a scan dump must be protected against unauthorized entities—including field service engineers—because it can contain confidential data. Therefore, our proposed architecture protects the content of a scan dump while still allowing the execution of a scan dump.

Fig. 11 illustrates the overall sequence of a scan dump in the proposed architecture. When a scan dump is required to debug abnormal behavior in an SOC, the software code for bug reproduction including the scan dump configuration needs to be prepared by an authorized engineer who can make secure software with the correct signature. The configuration includes the trigger condition and reordering scheme for the scan dump operation. The trigger condition sets the point to enter the scan dump mode, and the reordering scheme sets “Map_Sel” to determine the order for updating the dual shadow register sets. This configuration is implemented by writing secure registers on the memory-mapped address region that is generally used in modern SOCs. As secure software only can access this secure address region, it is not available for other software to configure a scan dump before privileged. By executing the code delivered from an authorized engineer, a field test engineer can obtain the scan dump data from the chip. Since the result is obfuscated according to the configuration, it can only be analyzed by an engineer who knows the reordering information in the secure software code. Even if attackers acquire the scan dump code and try to determine the scan dump configuration by repeatedly executing the code, it is impossible using such a method because attackers cannot figure out whether changes in the data have come from the experimental variation or

TABLE 2. The example of the obfuscation in SR in the scan dump mode with four-bit shadow register sets.

Cycle count	Pure SO	Map_Sel [7:0]	SE _A	TCK _A3	TCK _A2	TCK _A1	TCK _A0	SR_A [3:0]	SE _B	TCK _B3	TCK _B2	TCK _B1	TCK _B0	SR_B [3:0]	Secure SO
0	p0	{ 2'b00, 2'b10, 2'b11, 2'b01 }	0	En	-	-	-	p0, 0, 0, 0	1	En	En	En	En	4'b0	0
1	p1		0	-	-	-	En	p0, 0, 0, p1	1	En	En	En	En	4'b0	0
2	p2		0	-	En	-	-	p0, p2, 0, p1	1	En	En	En	En	4'b0	0
3	p3		0	-	-	En	-	p0, p2, p3, p1	1	En	En	En	En	4'b0	0
4	p4		1	En	En	En	En	p2, p3, p1, 0	0	En	-	-	-	p4, 0, 0, 0	p0
5	p5		1	En	En	En	En	p3, p1, 0, 0	0	-	-	-	En	p4, 0, 0, p5	p2
6	p6		1	En	En	En	En	p1, 0, 0, 0	0	-	En	-	-	p4, p6, 0, p5	p3
7	p7		1	En	En	En	En	4'b0	0	-	-	En	-	p4, p6, p7, p5	p1

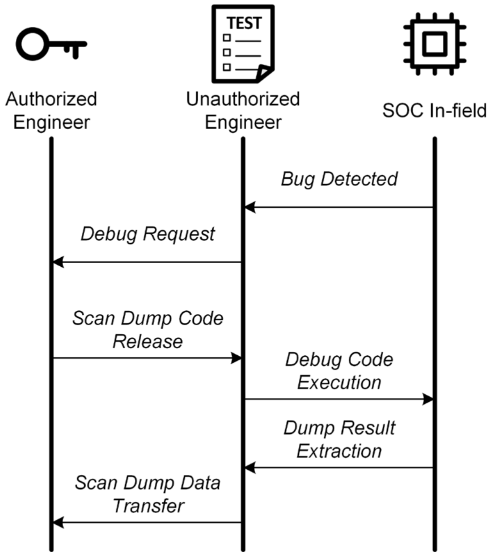

FIGURE 11. Flow of proposed scan dump sequence.

the reordering configuration. Analyzing and debugging the results can begin after the authorized engineer has parsed the results with the reordering configuration.

VI. PERFORMANCE EVALUATION

This section evaluates the security and overheads of the proposed secure scan architecture and reports on comparisons with existing secure scan architectures.

A. SECURITY

1) SCAN-BASED SIDE-CHANNEL ATTACK

This attack tries to obtain the confidential data being processed in functional mode. By switching the chip from function mode to test mode, confidential information appears in the scan chain and can be analyzed by attackers after it is shifted out. In our proposed architecture, the data is reset first when external test access occurs. The obfuscated data or the reset value is shifted out according to the key comparison result for this attack.

2) TEST MODE ONLY SCAN-BASED ATTACK

An attacker using this method infers sensitive information from the change in test responses for the input patterns. This attack does not require mode-switching but relies on the test response. With our proposed method, an attacker cannot get the correct response without knowing both the correct skew and the key. If an attacker has knowledge about SLAKE, they require information about the key value and the number of stages. An attacker who knows the correct key value cannot solve the lock without the implemented skew information, so the response is obfuscated by the key captured in the last stage that the attacker does not know. Unless an attacker figures out both the key value and the skew information, the responses will be obfuscated and cannot be reconstructed without knowing the entire architecture.

3) RESETTING ATTACK AND FLUSHING ATTACK

In a resetting attack, the attacker can try to determine the obfuscation scheme by observing the response right after resetting the CUT. Similarly, the attacker can keep flushing the scan chain to determine the obfuscation scheme. However, as SR in the proposed architecture does not scramble but invalidates the scan out data for incorrect key input in the test mode, the attacker cannot figure out the sequence of LFSR without any knowledge about the SR structure. Even if the attacker figures out the structure of LFSR, the attacker does not have any chance to infer the original value because the obfuscated data is invalidated by re-ordering, duplicating, and removing in SR.

4) BRUTE FORCE KEY ATTACK

When using a serialized n-bit key, the probability of correctly guessing the key value in the simple scheme is $1/2^n$. In case of authentication using an input key, the total length of the key should be sufficiently long for the security. Otherwise, the secure architecture can be unlocked by successive attempts or coincidence. In our proposed architecture with n-bit KFFs, a single-bit key input is translated into an

n-bit-wide parallel key. Since the potential result in SKI can be affected by the implementation, we assume here that all paths have different delays for evaluation. With the skew-adjusted TCK, the n-bit KFFs in a single stage produce 2^n possible results. At the same time, if the architecture is implemented to have m stages, there are $2^{(n\times m)}$ possible results. If it is required to have the equivalent robustness of a 32-bit serial key, then a four-bit key with eight stages is sufficient in SKI.

Since an attacker must determine not only the value of the correct key but also the skew to capture the value correctly, it is impossible to unlock the chip without either information. If an attacker who knows the golden key figures out the presence of the SKI, the attacker must find the skew value to capture it. Unlike the usual target in a brute force attack for existing architecture, the combination of skew between SI and TCK can be infinite. Since the requisites for figuring out the skew between SI and TCK to capture the desired key includes the latency on each path and the timing constraints, this cannot be solved unless the timing database is leaked. Even if an attacker accidentally captures the desired key for a stage, the attacker cannot know if the key for a certain stage is correctly captured. Moreover, the skew value determined for a certain stage cannot be used for other stages.

5) BIT-ROLE IDENTIFICATION ATTACK

If a countermeasure with lock and key scheme obfuscates the test response based on the authentication status, attackers can identify the location of the key bit by observing the difference in the results for single bit flipped input. In the proposed architecture, the obfuscation logic uses only the input key value of the last stage and does not use the input key values of the other stages. Therefore, the attacker cannot identify the location of other key bits.

6) MEMORY ATTACK

For countermeasures using its key value in NVMs can be vulnerable for the attack targeting the memory. In the proposed architecture, the golden key and the seed of the LFSR are not placed in the memory. The configuration for the scan dump comes from the secure region in the memory-mapped address but it is written in flip-flops. The only information stored in the memory is ''Key_Sel'' in the OTP to change the golden key, but it only selects key sets to use and does not include any key value.

7) ARCHITECTURE EXPOSURE ISSUE

In case that the attacker figures out the number of buffers in SKI by the side-channel attack (e.g., electromagnetic attack), the attacker requires the library information to obtain the value of the implemented skew. Even though the attacker determines the skew value by observing the radiation many times, the attacker must determine the golden key to unlock the chip.

If it is determined that the manufacturer, third-party designer, or anyone who knows the architecture cannot be trusted, the original key and method of unlocking for the manufacturing test should be updated to neutralize their knowledge about the architecture. In our proposed architecture, it is possible to update the key map using the OTP. After the golden key has been changed by a trustworthy user, malicious attempts are blocked by SKI with the updated key. The attacker then has to determine the updated golden key value and required skew to capture it properly.

8) SCAN DUMP PROTECTION

As scan dump plays an important role in the SOC debug, it is necessary to maintain the functionality of a scan dump while protecting the content since its observability can be a double-edged sword. It is essential for engineers to debug SOC but it must not be analyzed by unauthorized engineers due to the importance of the content it contains. In our proposed architecture, a scan dump is not allowed without the configuration in secure software released by an authorized engineer. Obtaining the scan dump data can be done by any engineers who have software code containing the scan dump configuration. However, the obtained data is obfuscated by the reordering configuration in the secure software, so unauthorized engineers cannot analyze the results. Even if an unauthorized engineer analyzes the results of several scan dumps to infer the obfuscation scheme, they have to determine the reordering map among 40,320 cases for the architecture having eight-bit SR. Moreover, they will be unable to figure out whether the changes in the data are due to the functional operation by programs or obfuscation. While scan dumps are protected against unauthorized access and analysis, the observability and precision are kept high because the architecture does not use any functional logic for authentication or obfuscation except a few dedicated configuration registers.

9) COMPARISON WITH THE PREVIOUS APPROACHES

Table 3 presents a comparison between the security of the proposed secure scan architecture and the previous approaches. The reported results for scan chain encryption [22] are assumed to use SKINNY-64-128, and the results of improved DFT [20] are assumed to use the 128-bit password assigning 64 bits for the scan password. Secure DFT [16] and HBSD-low [23] do not use an authentication key. DOS-10% [28] does not have an authentication key and its results assume that it uses a 64-bit obfuscation key driven by the LFSR. The results for other designs are based on implementing a 64-bit key. The proposed secure scan architecture using SLAKE denoted ''SLAKE-8-8'' is based on an eight-stage eight-bit key to have the equivalent key complexity of a 64-bit serial key without considering skew complexity.

DOSD-64 [32] can achieve a 2^{64} security level when the key is 64 bits. DOSD-64 allows scan dumps when the input key is correct. FTSL-64 [21] also has 2^{64} complexity against exhaustive key search with a 64-bit key, but it does not protect the chip against attacks at manufacturing test. Furthermore, neither can protect scan dumps because the content of scan

TABLE 3. Security comparison with the previous approaches.

Architecture	Key Length	Security Level	The Known Vulnerability	Scan Dump Protection
DOSD-64 [32]	64-bit	2^{64}	None	△
FTSL-64 [21]	64-bit	2^{64}	None	△
DOS-10% [28]	64-bit	$2^{64\times k}$	Memory attack	×
Secure DFT [16]	N/A	N/A	None	×
Improved DFT [20]	128-bit	2^{128}	None	△
HBSD-low [23]	N/A	10^{174}	None	×
Scan chain encryption [22]	128-bit for cipher	128 bits 36 rounds	Memory attack	△
SLAKE-8-8	8-bit	$2^{64}(C_{Ps}^{Cs})^8$	None	○

○: the architecture protects the scan dump without any vulnerability.
△: the architecture protects the scan dump with vulnerabilities.
×: the architecture cannot protect the scan dump.

dumps obtained with the correct key can be exposed to an unauthorized person. The security level of DOS-10% with the 64-bit obfuscation key can be expressed as has $2^{64\times k}$ where k represents the number of parallel scan chains. Since the seed in the NVM is used to initialize the LFSR in DOS-10%, it is vulnerable to attacks targeting the memory. The scan dump is not available in DOS because it drives the scan out to zero for the length of its scan chain. Secure DFT protects the crypto chip based on the mode selected in the beginning. It isolates the cipher key of the crypto chip in the test mode and blocks the mode change from the function mode to the test mode. Since any brute force attack is not applicable to this architecture, the security level cannot be expressed quantitatively. While this architecture effectively protects the scan design, the scan dump cannot be used in this architecture because the mode change from the function mode is not allowed. Improved DFT using a 128-bit password can achieve a 2^{128} security level, and it protects scan dump using the test password. However, this architecture does not distinguish between the tester and the debugger. Therefore, all the testers who are not authorized can investigate the internal state of the chip. HBSD-low includes a low area version of SHA3 to hash the test response. The reported security level of HBSD-low is 10^{174}. In this architecture, the original response cannot be recovered from the hashed response. Therefore, the diagnosis for this architecture requires a fault dictionary. However, analyzing the scan dump data requires the debugger to have a dictionary of all hashed results for all the possible scan dump data, which is not practical. Scan chain encryption uses SKINNY-64-128 whose security level is $2^{88.5}$ by 128-bit 36-round encryption [41]. This approach is known to be vulnerable to memory attack as it stores the key in the NVM embedded in the IC. As scan chain encryption encrypts all data from its scan chains, the scan dump data in this approach needs off-chip decryption to be analyzed. However, this approach requires additional logic operations such as decryption and encryption, which should be avoided in the scan dump situation. The encryption also requires an additional fast clock to operate the ciphers for multiple scan chains, which makes the preservation of snapshots difficult in the scan dump situation. Moreover, a tester who can do off-chip decryption can analyze scan dump data. Nevertheless, scan chain encryption is marked as a triangle because the scan dump data acquired by an unauthorized person can be decrypted by an authorized person without loss of data, unlike other previous approaches. SLAKE-8-8 uses only eight KFFs for the key to achieve a $2^{64} \times (C_{Ps}^{Cs})^8$ security level. Even if an attacker successfully finds the key from 2^{64} combinations, they will need to find the position of the clock edge for every stage to capture the desired value. As it is impossible for an attacker to figure out whether a certain stage is captured correctly, the attacker cannot unlock if any of the prerequisites are not ready. In addition to the high security level with only an eight-bit key, SLAKE-8-8 can protect scan dumps. In SLAKE-8-8, the scan dump mode can only be asserted when it is configured by secure software released by an authorized engineer. Therefore, even field test engineers who have the authority for in-field tests cannot attempt to launch a scan dump. After obtaining scan dump data by the secure software, it can only be analyzed by an authorized engineer who knows the obfuscation scheme programmed into the software. As the proposed architecture does not need any dynamic operation of other logics except SR, preserving the scan data for debugging is not difficult with the proposed architecture.

B. OVERHEADS

To evaluate the area overhead of the proposed secure scan architecture, it is coded in the RTL and synthesized independently as can be expected in practical cases. Table 4 shows the area overhead of our proposed architecture. The experiments are configured to evaluate the impact of the number of KFFs and the reorder depth of SR. The experiments were performed with the Synopsys SAED 32 nm library. The gate count of the proposed architecture with four KFFs and a four-bit dual shadow register is about 216. When eight KFFs are integrated, the gate count is about 240. With eight-bit dual shadow registers, the gate count becomes 324 for four KFFs and 356 for eight KFFs each. When compared to the scan-inserted ITC'99 benchmark circuits b18 and b19, the overhead of additional secure scan logic is 0.11–0.38%. Except for the OTP and the routing channel for SKI, the gate count results include the entire architecture depicted in Figure 4. The area required for the routing channel for the SKI is not included in the experimental result since buffering is only required for four or eight key lines.

Since SKI should use the skew implemented on the paths from SI to KFFs, its routing and clocking should be cared for manually during the implementation flow. The skew between KFFs should be enough to guarantee not having setup/hold violations for the desired key combination. Although some special care is necessary for SKI implementation, this is only

TABLE 4. Area overhead of the proposed architecture.

# of KFFs	# of Stages	Reorder Depth	Gate Count	vs. b18	vs. b19
4	8	4	216	0.23 %	0.11 %
8	8	4	240	0.26 %	0.13 %
4	8	8	324	0.35 %	0.17 %
8	8	8	356	0.38 %	0.19 %

TABLE 5. Overhead comparison with the previous approaches.

Architecture	Area Overhead	Test Time Overhead
DOSD-64 [32]	531 gates	64 cycles
FTSL-64 [21]	6,562 gates	2 + 64 cycles + 48 cycles
DOS-10% [28]	1,436 gates	zero
Secure DFT [16]	244 gates	zero
Improved DFT [20]	1,020 gates	2 cycles
HBSD-low [23]	5,136 gates	zero
Scan chain encryption [22]	1,696 gates for cipher excl. control logic	$2 \times 2 \times 64 + (64 - R)(K + 1)$
SLAKE-8-8	356 gates	8 cycles + T_i

a few paths since the proposed architecture can achieve a high security level with a small number of KFFs. The control signals for the scan dump are configured by software at the beginning of functional operation so the paths for those signals can be treated as false paths. Therefore, there is no implementation overhead related to scan dump control.

For the proposed architecture, additional test time is required to unlock through SKI. If the number of stages is m, the test time overhead is the duration of m cycles + T_i, where T_i means the accumulation of additional required time to achieve the key combination at each stage. The test time is only affected once at the initial test mode assertion, and it is not affected until the chip is locked again by power-on-reset or mode-change. Unlocking is not required in scan dump mode, but it takes a few more cycles depending on the length of the shadow register set. However, the number of additional cycles is not a problem since it is negligible in practical debugging situations that handle the snapshot of the entire SOC.

Table 5 shows the comparison result for the area overhead and the test time overhead with the previous approaches. The options are the same with the security analysis to compare the impacts on the area and test time. DOSD-64 costs 531 gates and requires 64 cycles before the start of the test. FTSL-64 requires 6,562 additional gates and requires an additional 114 cycles for the in-field test. The overheads of FTSL-64 can be increased or decreased according to the type of its hash function. The area overhead reported in [28] was 0.75% for b19, so we estimated the area overhead using the number of gates. DOS-10% costs 1,436 gates and does not have any test time overhead. Secure DFT costs 244 gates and does not have any test time overhead. This architecture is the only one with smaller overheads for both area and test time than SLAKE-8-8, but it cannot protect the scan dump. The area overhead of improved DFT was reported as transistor count, so we converted it to 1,020 gates. The test time overhead is 2 cycles to compare the password. HBSD-low costs 5,136 gates including low area version of the SHA3-512 module, and it does not take additional test time. Scan chain encryption employs ciphers on the scan interface and the cipher SKINNY-64-128 costs 1,696 gates. The total area overhead of scan chain encryption including the control logic is not reported in gate count on [22]. Scan chain encryption needs 256 more cycles, and it takes time to fill in the cipher blocks for every test pattern. R and K in the table denote the total number of flip-flops in the original circuit modulo with segment size 64 and the total number of patterns. The area overhead of SLAKE-8-8 is 356 gates, so the cost is less than other countermeasures using a 64-bit key. For the proposed architecture, the test time overhead is determined by the number of stages and accumulated duration when waiting for the skew. Therefore, SLAKE-8-8 takes eight cycles and an additional duration to wait for the desired moment to capture for eight-stage SKI once at the start of the scan test.

C. EVALUATION SUMMARY AND DISCUSSION

The evaluation in Sections 6.1 and 6.2 shows that SLAKE can provide a high security level and scan dump protection with fewer key bits. In addition to the security, the area overhead and test time overhead required for SLAKE are smaller than most of the previous approaches. To analyze the evaluation intuitively, the results of the security analysis using the security comparison in a three-dimensional space is shown in Fig. 12. The ideal secure scan architecture means the architecture achieving the infinite security level while it protects both scan test and scan dump without any known vulnerability. In Fig. 12, the ideal secure scan architecture is placed on the lower-left corner, and the closer to this point means that the architecture has better overall security performance. Each architecture is placed in the space based on the evaluation in Section 6.1. Secure DFT is not included in this visualized comparison since it is not possible to express its security level. Improved DFT is assumed to have the same length of the password as the length of the key in the other architectures. For scan test protection, although FTSL-64 does not recognize a manufacturing test as a target to protect, it does not have any known vulnerability and is robust against attack at the in-field test. Therefore, FTSL-64 is located between DOSD-64 and other architectures that have known vulnerabilities. DOS-10% and HBSD-low do not have test time overhead, and improved DFT takes only two cycles. Therefore, HBSD-low and improved DFT are placed closer than SLAKE-8-8 for scan test protection, and DOS-10% is located closer than scan

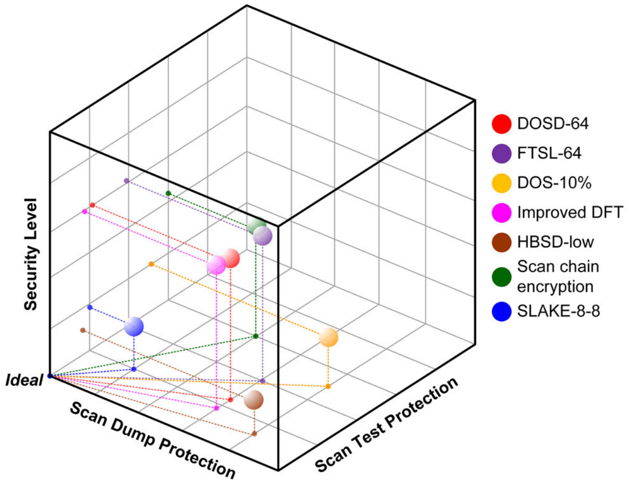

FIGURE 12. Visualized security comparison with the previous architectures.

chain encryption. Since HBSD-low and DOS-10% cannot use scan dump, it is located further from the ideal position than DOSD-64, FTSL-64, and improved DFT for scan dump protection. Scan chain encryption is located closer than DOSD-64 for scan dump protection because its scan dump data is encrypted even though it does not distinguish between the tester and the debugger.

In Fig. 12, the closest architecture to the ideal point is SLAKE-8-8. SLAKE-8-8 achieves a high security level and protects both scan test and scan dump without any known vulnerability. By involving skew in the lock and key scheme, SLAKE achieves high complexity with fewer key flip-flops. Most importantly, the proposed architecture provides scan dump protection so the debuggability is maintained after the chip has been shipped. Furthermore, the analysis in Section 6.2 shows that SLAKE is efficient in terms of area and test time overhead. The only concern of SLAKE is the physical implementation of SKI which may require iterations due to the chip's floorplan. However, the accumulated experiences on the processes and libraries can reduce the iterations, making the physical implementation easier. Furthermore, it can be used as a formulated technique for each process and library. Even taking the drawback at the physical implementation stage, the high security performance and efficiency provided by SLAKE is sufficient to be chosen as a practical solution.

VII. CONCLUSION

In this paper, a secure scan architecture called SLAKE is proposed for secure scan test and secure scan dump. Without impacting the testability and the debuggability, SLAKE defends scan-based attacks effectively by combining the skew information into the lock and key. The attackers who have the attacking method targeting previous methodologies will face an invisible barrier when they attempt to unlock the scan design. Moreover, the high complexity beyond the barrier can be achieved with only a few key flip-flops in the proposed architecture. Unless both the correct key value and skew information are leaked, scan-based attacks are not available for the scan design with the proposed architecture. In addition to the high level of security against scan-based attacks, the proposed architecture provides the secure scan dump. Scan dump has not been protected in the previous works despite its importance in SOC debug. With our proposed architecture, a scan dump can only be executed with the secure software released by authorized engineers. The scan dump data can be extracted even at remote sites by unauthorized engineers to accelerate debugging, but the content is protected against unauthorized engineers. The proposed architecture can be individually synthesized and integrated into DFT inserted design without affecting the design flow. In the experiments, the area overhead and test time overhead of the proposed architecture are very small while it could effectively achieve a high security level.

REFERENCES

[1] M. Bushnell and V. Agrawal, *Essentials of Electronic Testing for Digital Memory and Mixed-Signal*. Boston, MA, USA: Kluwer, 2000, p. 14.

[2] J. Katz, "A case-study in the use of scan in microSPARC testing and debug," in *Proc. Int. Test Conf.*, Washington, DC, USA, 1994, pp. 456–460.

[3] D. D. Josephson, S. Poehhnan, and V. Govan, "Debug methodology for the McKinley processor," in *Proc. Int. Test Conf.*, Baltimore, MD, USA, 2001, pp. 451–460.

[4] B. Vermeulen and S. K. Goel, "Design for debug: Catching design errors in digital chips," *IEEE Design Test Comput.*, vol. 19, no. 3, pp. 35–43, May/Jun. 2002.

[5] O. Caty, P. Dahlgren, and I. Bayraktaroglu, "Microprocessor silicon debug based on failure propagation tracing," in *Proc. IEEE Int. Conf. Test*, Austin, TX, USA, Nov. 2005, pp. 284–293.

[6] M. Abramovici, P. Bradley, K. Dwarakanath, P. Levin, G. Memmi, and D. Miller, "A reconfigurable design-for-debug infrastructure for SoCs," in *Proc. 43rd ACM/IEEE Design Automat. Conf.*, San Francisco, CA, USA, Jul. 2006, pp. 7–12.

[7] M. E. Amyeen, S. Venkataraman, and M. W. Mak, "Microprocessor system failures debug and fault isolation methodology," in *Proc. Int. Test Conf.*, Austin, TX, USA, Nov. 2009, pp. 1–10.

[8] B. Yang, K. Wu, and R. Karri, "Scan based side channel attack on dedicated hardware implementations of data encryption standard," in *Proc. Int. Test Conf.*, Charlotte, NC, USA, Oct. 2004, pp. 339–344.

[9] J. Da Rolt, G. Di Natale, M.-L. Flottes, and B. Rouzeyre, "New security threats against chips containing scan chain structures," in *Proc. IEEE Int. Symp. Hardw.-Oriented Secur. Trust*, San Diego, CA, USA, Jun. 2011, p. 110.

[10] H. Kodera, M. Yanagisawa, and N. Togawa, "Scan-based attack against DES cryptosystems using scan signatures," in *Proc. IEEE Asia–Pacific Conf. Circuits Syst. (APCCAS)*, Kaohsiung, Taiwan, Dec. 2012, pp. 599–602.

[11] S. S. Ali, O. Sinanoglu, S. M. Saeed, and R. Karri, "New scan-based attack using only the test mode," in *Proc. IFIP/IEEE 21st Int. Conf. Very Large Scale Integr. (VLSI-SoC)*, Istanbul, Turkey, Oct. 2013, pp. 234–239.

[12] S. S. Ali, S. M. Saeed, O. Sinanoglu, and R. Karri, "Novel test-mode-only scan attack and countermeasure for compression-based scan architectures," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 5, pp. 808–821, May 2015.

[13] S. S. Ali, S. M. Saeed, O. Sinanoglu, and R. Karri, "Scan attack in presence of mode-reset countermeasure," in *Proc. IEEE 19th Int. Line Test. Symp. (IOLTS)*, Chania, Greece, Jul. 2013, pp. 230–231.

[14] D. Hely, F. Bancel, M. Flottes, and B. Rouzeyre, "Test control for secure scan designs," in *Proc. Eur. Test Symp. (ETS)*, Tallinn, Estonia, 2005, pp. 190–195.

[15] B. Yang, K. Wu, and R. Karri, "Secure scan: A design-for-test architecture for crypto chips," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 10, pp. 2287–2293, Oct. 2006.

[16] W. Wang, J. Wang, W. Wang, P. Liu, and S. Cai, "A secure DFT architecture protecting crypto chips against scan-based attacks," *IEEE Access*, vol. 7, pp. 22206–22213, 2019.

[17] J. Lee, M. Tehranipoor, C. Patel, and J. Plusquellic, "Securing designs against scan-based side-channel attacks," *IEEE Trans. Dependable Secure Comput.*, vol. 4, no. 4, pp. 325–336, Oct./Dec. 2007.

[18] J. Lee, M. Tehranipoor, C. Patel, and J. Plusquellic, "Securing scan design using lock and key technique," in *Proc. 20th IEEE Int. Symp. Defect Fault Tolerance VLSI Syst. (DFT)*, Monterey, CA, USA, Oct. 2005, pp. 51–62.

[19] J. Lee, M. Tehranipoor, and J. Plusquellic, "A low-cost solution for protecting IPs against scan-based side-channel attacks," in *Proc. 24th IEEE VLSI Test Symp.*, Berkeley, CA, USA, Apr. 2006, pp. 94–99.

[20] W. Wang, X. Wang, J. Wang, N. N. Xiong, S. Cai, and P. Liu, "Ensuring cryptography chips security by preventing scan-based side-channel attacks with improved DFT architecture," *IEEE Trans. Syst., Man, Cybern. Syst.*, early access, Dec. 3, 2021, doi: 10.1109/TSMC.2020.3036879.

[21] A. Cui, C.-H. Chang, W. Zhou, and Y. Zheng, "A new PUF based lock and key solution for secure in-field testing of cryptographic chips," *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 2, pp. 1095–1105, Apr. 2021.

[22] M. D. Silva, M.-L. Flottes, G. Di Natale, and B. Rouzeyre, "Preventing scan attacks on secure circuits through scan chain encryption," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 3, pp. 538–550, May 2019.

[23] A. Cui, M. Li, G. Qu, and H. Li, "A guaranteed secure scan design based on test data obfuscation by cryptographic hash," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 12, pp. 4524–4536, Dec. 2020.

[24] S. K. Goel and B. Vermeulen, "Hierarchical data invalidation analysis for scan-based debug on multiple-clock system chips," in *Proc. Int. Test Conf.*, Baltimore, MD, USA, 2002, pp. 1103–1110.

[25] H. Yi, S. Kundu, S. Cho, and S. Park, "A scan cell design for scan-based debugging of an SoC with multiple clock domains," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 7, pp. 561–565, Jul. 2010.

[26] J. Da Rolt, G. Di Natale, M.-L. Flottes, and B. Rouzeyre, "Are advanced DfT structures sufficient for preventing scan-attacks?" in *Proc. IEEE 30th VLSI Test Symp. (VTS)*, Maui, HI, USA, Apr. 2012, pp. 246–251.

[27] J. D. Rolt, A. Das, G. D. Natale, M.-L. Flottes, B. Rouzeyre, and I. Verbauwhede, "Test versus security: Past and present," *IEEE Trans. Emerg. Topics Comput.*, vol. 2, no. 1, pp. 50–62, Mar. 2014.

[28] X. Wang, D. Zhang, M. He, D. Su, and M. Tehranipoor, "Secure scan and test using obfuscation throughout supply chain," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 9, pp. 1867–1880, Sep. 2018.

[29] S. S. Ali, Y. Sao, and S. Biswas, "Opacity preserving countermeasure using finite state machines against differential scan attacks," in *Proc. Eur. Test Symp.*, Bruges, Belgium, 2021, pp. 1–2.

[30] M. A. Razzaq, V. Singh, and A. Singh, "SSTKR: Secure and testable scan design through test key randomization," in *Proc. Asian Test Symp.*, New Delhi, India, Nov. 2011, pp. 60–65.

[31] Y. Luo, A. Cui, G. Qu, and H. Li, "A new countermeasure against scan-based side-channel attacks," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Montreal, QC, Canada, May 2016, pp. 1722–1725.

[32] A. Cui, Y. Luo, and C.-H. Chang, "Static and dynamic obfuscations of scan data against scan-based side-channel attacks," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 2, pp. 363–376, Feb. 2017.

[33] Q. Wang, A. Cui, G. Qu, and H. Li, "A new secure scan design with PUF-based key for authentication," in *Proc. IEEE 38th VLSI Test Symp. (VTS)*, San Diego, CA, USA, Apr. 2020, pp. 1–6.

[34] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology—CRYPTO*, M. Wiener, Ed. Berlin, Germany: Springer, 1999, pp. 388–397.

[35] E. Konur, Y. Ozelci, E. Arikan, and U. Eksi, "Power analysis resistant SRAM," in *Proc. World Automat. Congr.*, Budapest, Hungary, Jul. 2006, pp. 1–6.

[36] R. Baranowski, M. A. Kochte, and H. J. Wunderlich, "Fine-grained access management in reconfigurable scan networks," *IEEE Trans. Comput.-Aided Design Integr.*, vol. 34, no. 6, pp. 937–946, Jun. 2015.

[37] D. Josephson, "The good, the bad, and the ugly of silicon debug," in *Proc. 43rd Annu. Conf. Design Automat. (DAC)*, San Francisco, CA, USA, 2006, pp. 3–6.

[38] (2020). *UEFI Forum, Unified Extensible Firmware Interface Specification: Version 2.8*. Accessed: Jun. 15, 2021. [Online]. Available: http://www.uefi.org/sites/default/files/resources/UEFI%20Spec%202_6.pdf

[39] N. Sklavos, R. Chaves, G. Di Natale, and F. Regazzoni, *Hardware Security and Trust*. Cham, Switzerland: Springer, 2017.

[40] V. Zimmer and M. Krau. (2016). *Establishing the Root of Trust*. Accessed: Jun. 15, 2021. [Online]. Available: https://uefi.org/node/3544

[41] C. Beierle, J. Jean, S. Kölbl, G. Leander, A. Moradi, T. Peyrin, Y. Sasaki, P. Sasdrich, and S. M. Sim, "The SKINNY family of block ciphers and its low-latency variant MANTIS," in *Advances in Cryptology—CRYPTO*, vol. 9815, M. Robshaw and J. Katz, Eds. Berlin, Germany: Springer, 2016, pp. 123–153.

HYUNGIL WOO received the B.S. degree in electrical and electronic engineering from Chung-Ang University (CAU), Seoul, South Korea, in 2012. He is currently pursuing the M.S. degree with the Department of Electrical and Electronics Engineering, Yonsei University, Seoul. Since 2012, he has been a SOC Design Engineer with Samsung Electronics, Hwaseong, South Korea. His current research interests include VLSI/SOC design and testing, design for testability, and design for debug.

SEOKJUN JANG received the B.S. degree in electrical and electronic engineering from Yonsei University, Seoul, South Korea, in 2018, where he is currently pursuing the combined M.S. degree with the Department of Electrical and Electronics Engineering. His current research interests include scan chain diagnosis, test pattern reordering, and design for testability.

SUNGHO KANG (Senior Member, IEEE) received the B.S. degree in control and instrumentation engineering from Seoul National University, Seoul, South Korea, in 1986, and the M.S. and Ph.D. degrees in electrical and computer engineering from The University of Texas at Austin, Austin, TX, USA, in 1988 and 1992, respectively.

He was a Research Scientist with Schlumberger Laboratory for Computer Science, Schlumberger Inc., Austin, and a Senior Staff Engineer with Semiconductor Systems Design Technology, Motorola Inc., Austin. Since 1994, he has been a Professor with the Department of Electrical and Electronic Engineering, Yonsei University, Seoul. His current research interests include VLSI/SOC design and testing, design for testability, design for manufacturability, and fault tolerant computing.