STM JOURNALS

Recent Trends in
Electronics & Communication Systems

ISSN: 2393-8757
Volume 12, Issue 3, 2025
September–December
DOI (Journal): 10.37591/RTECS

https://journals.stmjournals.com/rtecs

Review RTECS

VHDL-Based Strategies for Protecting IoT Devices from Power and Electromagnetic Side-Channel Attacks: A Study

Pathan Muskan Ibrahim[1], Shaikh A. Hakim A. Razzaque[2], Heena T. Shaikh[3], Kazi Kutubuddin Sayyad Liyakat[4,*]

Abstract

The environment of the Internet of Things (IoT) is rapidly developing, linking billions of devices across a wide variety of disciplines. Furthermore, despite the fact that it provides an unprecedented level of convenience and efficiency, this interconnection also presents a fertile ground for security weaknesses. Side-channel attacks, also known as SCAs, are one of the dangers that pose a considerable risk. These attacks specifically target the cryptographic implementations that are the foundation of Internet of Things security. In order to obtain sensitive information such as secret keys, SCAs make use of information that is released from the physical execution of cryptographic algorithms. This information includes things like power usage, fluctuations in timing, and electromagnetic radiation. It is of the utmost importance to protect Internet of Things devices against security compromise attacks (SCAs). One promising strategy involves embedding countermeasures directly into the hardware design by utilizing hardware description languages such as VHDL. The role of VHDL programming in the implementation of SCA countermeasures within the realm of Internet of Things security is investigated in this paper. Cryptographic methods are frequently utilized by Internet of Things devices by means of authentication, encryption, and data integrity. Nevertheless, even cryptographic algorithms that are technically sound can be susceptible to SCAs if the implementation of the algorithm's implementation reveals information about the algorithm's internal processes. During a power analysis attack, for instance, an adversary can keep track of the amount of power that a device consumes while it is responsible for carrying out a cryptographic algorithm. The attacker is able to extrapolate information about the crucial bits that are being processed by analyzing the patterns of power consumption, which ultimately results in the device's security being compromised.

***Author for Correspondence**
Kazi Kutubuddin Sayyad Liyakat
E-mail: drkkazi@gmai..com

[1]Student, Department of Electronics and Telecommunication Engineering, Brahmdevdada Mane Institute of Technology, Solapur, Maharashtra, India
[2-3]Assistant. Professor, Department of Electronics and Telecommunication Engineering, Brahmdevdada Mane Institute of Technology, Solapur, Maharashtra, India
[4]Professor, Department of Electronics and Telecommunication Engineering, Brahmdevdada Mane Institute of Technology, Solapur, Maharashtra, India

Received Date: October 04, 2025
Accepted Date: October 06, 2025
Published Date: December 13, 2025

Citation: Pathan Muskan Ibrahim, Shaikh A. Hakim A. Razzaque, Heena T. Shaikh, Kazi Kutubuddin Sayyad Liyakat. VHDL-Based Strategies for Protecting IoT Devices from Power and Electromagnetic Side-Channel Attacks: A Study. Recent Trends in Electronics & Communication Systems. 2025; 12(3): 30–40p.

Keywords: IoT, security, side channel attack, VHDL programming, VHDL

INTRODUCTION

The Internet of Things is transforming numerous facets of human existence, ranging from intelligent residences to industrial automation. This interconnection presents considerable security issues. IoT devices, typically limited in resources and situated in precarious locations, are excellent targets for hostile entities. Among the several assault vectors, side-channel attacks (SCAs) represent a notably insidious and formidable threat. SCAs can circumvent conventional cryptographic safeguards by extracting sensitive information

through the analysis of a device's physical properties during operation.

This section examines the function of VHDL (VHSIC Hardware Description Language) programming in executing effective countermeasures against side-channel attacks in IoT device security. We will examine the significance of hardware-level protections, the various types of SCAs that pose threats to IoT devices, and the application of VHDL to alleviate these vulnerabilities.

Software-based security solutions may be beneficial; yet, they frequently prove inadequate against advanced supply chain attacks (SCAs). Attackers utilizing side-channel attacks observe power usage, electromagnetic radiation, timing discrepancies, or auditory emissions to infer secret keys or algorithms. These signals are inherent to the hardware implementation and can be utilized without direct access to the software. Consequently, integrating countermeasures directly into the hardware design with VHDL constitutes an essential security layer [1–4]. Figure 1 shows the IoT Security by electronics means.

Various forms of SCAs can jeopardize the security of IoT devices, as seen in Figure 2. Power Analysis Attacks (PAAs): These attacks observe the variations in power usage of a device to deduce information regarding the processes executed. Simple Power Analysis (SPA) can disclose the fundamental algorithmic framework, but Differential Power Analysis (DPA) use statistical techniques to associate power traces with confidential keys.

- *Timing Attacks:* By scrutinizing the duration of cryptographic processes, attackers can infer details regarding the utilized key. Discrepancies in execution duration contingent upon important parameters can be leveraged in timing analysis.
- *Electromagnetic (EM) Analysis:* Analogous to power analysis, EM attacks observe the electromagnetic radiation released by the equipment throughout its functioning. Electromagnetic emanations can disclose information regarding the device's internal condition and cryptographic operations.
- *Acoustic Attacks:* In certain instances, sound emissions from the device may be utilized to infer details on the computations being executed.

VHDL enables designers to incorporate countermeasures at the hardware level, so substantially enhancing the robustness of IoT devices against side-channel attacks. Here is the application of VHDL:

- *Masking:* This method entails the incorporation of random values (masks) to conceal the correlation between the device's functions and the secret key. VHDL can be utilized to execute masked cryptographic algorithms, wherein all sensitive variables are amalgamated with random masks prior to processing. The masks are subsequently eliminated, guaranteeing accurate computational results while concealing power usage and other side-channel information. The masking must be executed meticulously to prevent the disclosure of the secret key.

Figure 1. IoT applications by electronics means.

```
-- Example of masking in VHDL
signal key_masked: std_logic_vector(127 downto 0);
signal random_mask: std_logic_vector(127 downto 0);

process(clk, rst)
begin
```



```
  if rst = '1' then
    key_masked <= (others => '0');
  elsif rising_edge(clk) then
    -- Generate a new random mask
    random_mask <= generate_random_mask();

    -- Mask the key
    key_masked <= key XOR random_mask;

    -- Perform operations on the masked key
    -- ...
  end if;
end process;
```

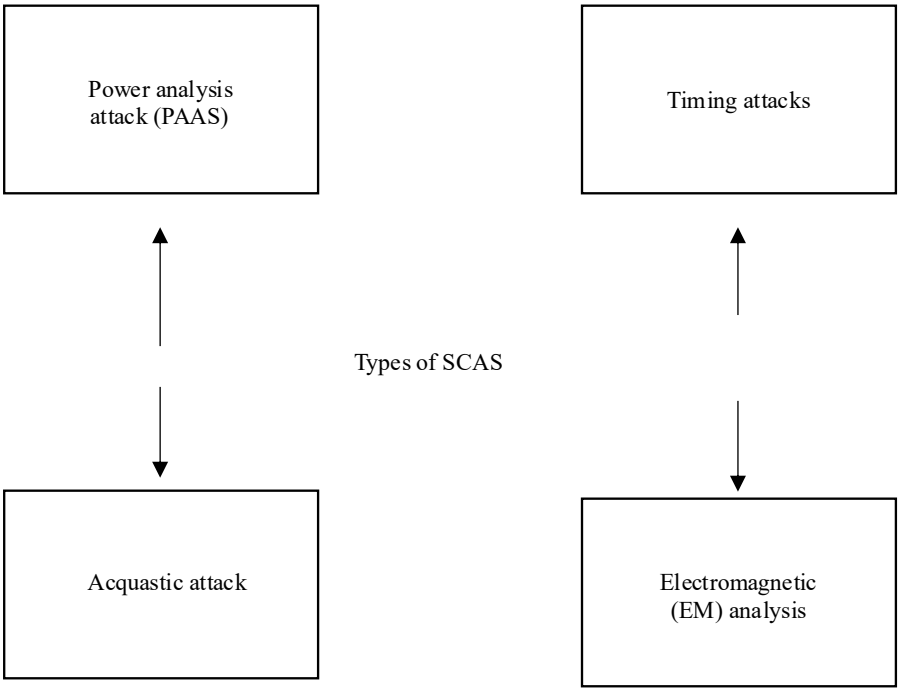

Figure 2. Types of SCAs.

- *Hiding*: This approach aims to make the device's operations appear constant, regardless of the actual calculations being performed. This involves techniques such as:
  - *Dual-Rail Logic:* This uses two wires to represent each bit, with one wire being the true value and the other being its complement. This ensures that the power consumption remains constant, regardless of the data being processed. VHDL can be used to implement dual-rail logic gates and circuits.
  - *Constant Power Consumption Circuits:* Designing circuits that consume a constant amount of power, regardless of the operations performed, can effectively hide information leakage.
  - *Dummy operations:* Inserting dummy operations to obfuscate the timing and power consumption patterns.
- *Algorithm-Level Countermeasures:* Specific cryptographic algorithms are designed to be more resistant to SCAs. Implementing these algorithms in VHDL can provide a strong defense against attacks. Examples include:
  - *Threshold Implementations:* A variant of masking where data is split into multiple shares. The number of shares required to reconstruct the original data is greater than the threshold, preventing information leakage.

  - *Boolean Masking:* Applying bitwise operations to secret data with random booleans to obscure the relationship between underlying secret data and resulting side-channel leakage.
- *Randomization:* Introducing randomness into the execution flow of cryptographic algorithms can make it harder for attackers to correlate side-channel signals with the secret key. This can involve randomizing the order of operations or using random delays. VHDL can be used to implement random number generators and control the timing of operations.

Implementing side-channel attack (SCA) countermeasures in VHDL poses numerous challenges:
- *Performance Overhead:* Countermeasures may considerably affect the performance of IoT devices, which frequently possess constrained resources. Meticulous optimization is essential.
- *Complexity:* The implementation of advanced countermeasures may augment the intricacy of the VHDL code, hence complicating verification and maintenance.
- *Verification:* Comprehensive validation of the efficacy of countermeasures against SCAs necessitates specific tools and methodologies, including power analysis simulation and fault injection.
- *Expense:* The creation and validation of hardware resistant to side-channel attacks can be costly.

Side-channel attacks represent a significant risk to the security of IoT devices. Although software defenses are significant, the implementation of countermeasures at the hardware level through VHDL is essential for constructing resilient and secure systems. Utilizing approaches such as masking, concealing, and algorithmic defenses, developers can substantially mitigate the susceptibility of IoT devices to side-channel attacks. Nonetheless, it is crucial to recognize that the implementation of such countermeasures entails complexity, performance overhead, and necessitates thorough testing and validation. With the ongoing expansion of the IoT landscape, the demand for secure hardware is set to rise, hence accentuating the importance of VHDL programming as a crucial talent for safeguarding the confidentiality and integrity of sensitive data [5–8].

VHDL IN FORTIFYING IOT SECURITY AGAINST SIDE-CHANNEL ATTACKS

The Internet of Things (IoT) has proliferated in recent years, linking billions of devices from smart home appliances to essential infrastructure. This interconnection presents a substantial security challenge. A significant concern to IoT security is the side-channel attack (SCA), which leverages a device's physical properties during cryptographic processes to disclose critical information. Hardware description languages like as VHDL are essential in the development and implementation of effective remedies against these threats [9–11].

In contrast to conventional cryptographic assaults that directly target algorithms, side-channel attacks (SCAs) concentrate on obtaining information revealed through quantifiable physical occurrences. These "side-channels" may include:
- *Power Consumption:* Fluctuations in power consumption during cryptographic operations can disclose information on processed data and key bits.
- *Timing Analysis:* Assessing the execution duration of cryptographic processes may reveal dependencies on key values.
- *Electromagnetic Radiation:* Analyzing electromagnetic emissions can uncover correlations with the processed data.
- *Acoustic Emissions:* Sounds produced during computation may, in certain instances, disclose information regarding the activities being performed.

These assaults are especially effective against embedded and IoT devices, which frequently possess constrained processing power and memory, rendering software-based mitigation strategies ineffective.

VHDL (VHSIC Hardware Description Language) is a robust language utilized for the description and simulation of digital electrical systems. Its capacity to simulate hardware at a granular level renders

it particularly adept for executing SCA countermeasures:

- *Exacting control over hardware implementation:* VHDL permits developers to meticulously optimize the hardware execution of cryptographic algorithms, facilitating the direct incorporation of countermeasures into the design. This differs from software-based methods that frequently lack the necessary granularity to adequately address SCAs.
- *Execution of hardware-level countermeasures:* VHDL facilitates the implementation of many hardware-level countermeasures, encompassing:
  - *Masking:* This technique involves introducing random values (masks) during computations to hide the correlation between the data and the side-channel signal. VHDL allows precise control over the implementation of masking schemes, ensuring their effectiveness.
  - *Hiding:* Hiding techniques aim to make the side-channel signal constant and independent of the data. This can be achieved through techniques like constant power consumption circuits, which are effectively implemented using VHDL.
  - *Dual-Rail Logic:* This technique encodes each bit with two wires, representing both the true and complement values. This helps to balance power consumption and reduces the information leakage. VHDL allows for the precise design and verification of dual-rail logic implementations.
  - *Randomized Clocking:* Introducing randomness in the clock signal can disrupt timing-based attacks. VHDL allows for the implementation of randomized clock generators and their integration into the cryptographic core.
- *Simulation and Verification:* VHDL facilitates comprehensive simulation and verification of the engineered hardware. This is essential for validating the efficacy of the executed countermeasures prior to deployment. Simulation can detect potential flaws and enhance the design for SCA resistance.
- *Flexibility and Portability:* VHDL is a standardized language, enabling the transfer of designs across many hardware platforms and technologies. This adaptability is crucial for IoT devices, which frequently possess varied hardware specifications.

Numerous practical applications illustrate the efficacy of VHDL in executing SCA countermeasures for IoT devices:

- *Secure Bootloaders:* VHDL facilitates the development of secure bootloaders that integrate SCA-resistant cryptographic methods for firmware update authentication, thereby thwarting the injection of malicious code into the device.
- *Hardware Security Modules (HSMs):* VHDL can be utilized to create HSMs that safeguard sensitive keys and execute cryptographic operations within a secure environment, insulated from Side-Channel Attacks (SCAs). These HSMs are essential for safeguarding communication and data at rest in IoT devices.
- *Lightweight Cryptography:* VHDL can be utilized to create lightweight cryptographic algorithms with integrated side-channel attack countermeasures for resource-constrained IoT devices. These algorithms are engineered for efficiency and security, even when used on low-power devices.

Notwithstanding the benefits of VHDL, some obstacles persist in the development of SCA-resistant IoT devices:

- *Complexity:* The implementation of effective SCA countermeasures can substantially elevate the intricacy of hardware designs.
- *Verification:* Assessing the efficacy of SCA countermeasures necessitates specialized equipment and knowledge.
- *Overhead:* Countermeasures may incur additional costs regarding area, electricity, and performance.

Future research aims to create automated tools and procedures for the design and verification of hardware resistant to side-channel attacks. This encompasses the investigation of novel countermeasures, formal verification methodologies, and optimized hardware architectures [12].

Side-channel attacks represent a significant risk to the security of IoT devices. VHDL provides a potent and adaptable instrument for the development and execution of resilient countermeasures at the hardware level. Utilizing VHDL's functionalities, developers can design secure and robust IoT devices capable of enduring the always changing landscape of security threats. With the expansion of the IoT, VHDL's role in safeguarding linked devices will become progressively vital [13].

DESIGNING VHDL FOR SIDE-CHANNEL ATTACK RESISTANCE

IoT offers a connected world; yet, its interconnectivity presents considerable security risks. Among the most pernicious are side-channel attacks (SCAs), which capitalize on inadvertent information leakage from the physical implementations of cryptographic algorithms. Assessing power usage, electromagnetic emissions, or timing discrepancies during cryptographic processes can expose hidden keys, regardless of the mathematical integrity of the encryption algorithm. This is especially troubling in resource-limited IoT devices, where prioritizing performance and power frequently eclipses security concerns.

This article explores the VHDL programming design for the implementation of countermeasures against side-channel attacks in IoT security. VHDL, a prevalent hardware description language, provides control over the exact hardware implementation of cryptographic algorithms, rendering it an essential instrument for integrating SCA protection. Prior to implementing countermeasures, it is essential to comprehend the panorama of SCA threats in IoT. Prevalent SCA methodologies encompass:

- *Simple Power Analysis (SPA):* Directly analyzes power traces to differentiate between various operations, such as multiplications and additions, thereby exposing the algorithm's execution sequence and potentially the key.
- *Differential Power Analysis (DPA):* Utilizes statistical examination of power traces associated with presumed key bits to derive the confidential key.
- *Electromagnetic Analysis (EMA):* Analogous to power analysis, yet employs electromagnetic radiation emitted by the instrument.
- *Timing Attacks:* Assess the duration of cryptographic processes, capitalizing on discrepancies influenced by the key.
- *Fault Injection Attacks:* Deliberately introducing faults into the system during cryptographic processes to induce certain errors that disclose information regarding the key.

IoT devices, frequently functioning in physically vulnerable settings, are great targets for such attacks. Their restricted processing capabilities, minimal energy demands, and frequently inadequate physical security render them more susceptible than conventional computing platforms.

VHDL offers the necessary granularity for the implementation of SCA countermeasures at the hardware level. By meticulously regulating the hardware implementation, leakage can be minimized or rendered statistically negligible. Essential strategies for SCA resistance in VHDL design, as illustrated in Figure 3, comprise:

- *Masking:* Randomly obscures intermediate values during cryptographic operations, thereby dissociating power consumption from the data being processed. This necessitates the generation of random integers and the execution of masking procedures within the VHDL code.
- *VHDL Implementation:* Utilizes VHDL's ability to define and manipulate signals to XOR the data with a random mask before and after sensitive operations. Special care is required to ensure the mask is truly random and uniformly distributed.

```
signal masked_data: std_logic_vector(DATA_WIDTH - 1 downto 0);
signal random_mask: std_logic_vector(DATA_WIDTH - 1 downto 0);
process(clk, rst)
begin
  if rst = '1' then
```



```
    -- Initialization
  elsif rising_edge(clk) then
    -- Generate a new random mask
    random_mask <= generate_random_mask(); -- Assuming a function generate_random_mask is defined
    -- Mask the data
    masked_data <= input_data XOR random_mask;
    -- Perform the operation on the masked data
    result <= perform_cryptographic_operation(masked_data);
    -- Unmask the result (if required)
    output_data <= result XOR random_mask;
  end if;
end process;
```

- *Hiding:* Aims to make the power consumption independent of the processed data by making all operations take the same amount of time and consume the same amount of power, regardless of the input. This involves techniques like balancing operations, using dual-rail logic, and inserting dummy operations.
- *VHDL Implementation:* Requires careful consideration of the hardware implementation of the cryptographic algorithm. For example, using balanced logic gates where the number of transitions is consistent regardless of the data input. VHDL can be used to meticulously control the gate-level implementation.

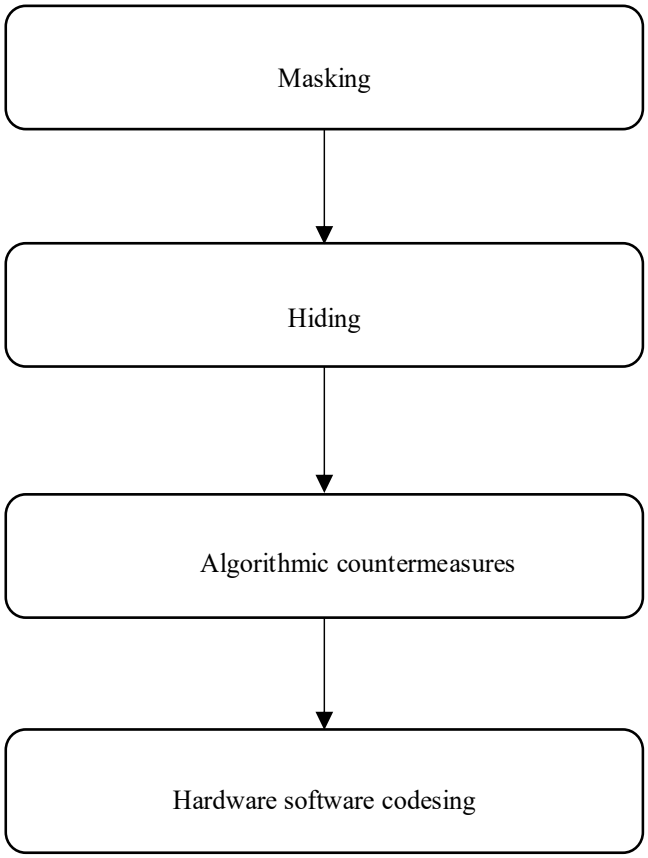

Figure 3. VHDL design for SCA.

```
-- Example: Dummy operation for timing equalization
process(clk)
```



```
begin
   if rising_edge(clk) then
      if condition then
         -- Perform actual computation
         result <= operation_a(input_data);
      else
         -- Perform dummy operation with similar complexity
         result <= dummy_operation(input_data);
      end if;
   end if;
end process;
```

- *Algorithmic Countermeasures:* Modifying the cryptographic algorithm itself to improve SCA resistance. Examples include using randomized execution or secure multiplication techniques.
- *VHDL Implementation*: Requires a deep understanding of the cryptographic algorithm and how to implement its SCA-resistant variant in hardware using VHDL. This might involve complex state machines, custom arithmetic units, and careful control of data flow.
- *Hardware-Software Co-design:* Optimizing the overall system, distributing tasks between hardware and software to leverage the strengths of each platform. This could involve offloading computationally intensive but less sensitive operations to software, while implementing critical cryptographic functions with hardware-level SCA protections.

Implementing SCA Countermeasures in VHDL Presents Several Challenges

- *Enhanced Complexity:* SCA-resistant systems frequently exhibit greater complexity compared to their unprotected equivalents, necessitating additional hardware resources and consequently affecting performance and power consumption.
- *Verification and Validation:* Comprehensive testing and validation are essential to ascertain the efficacy of the countermeasures. This necessitates specific instruments and proficiency in SCA analysis. Conventional functional verification is inadequate.
- *Performance Overhead:* Countermeasures frequently have performance overhead that must be meticulously weighed against the security benefits. This is especially crucial in resource-limited IoT devices.
- *Implementation Expenses:* The design, execution, and evaluation of SCA-resistant hardware can be expensive and labor-intensive.

Protecting IoT devices from side-channel attacks is essential for safeguarding sensitive information and maintaining the integrity of the overall ecosystem. VHDL offers a robust framework for executing SCA countermeasures at the hardware level. By employing meticulous design that integrates masking, concealing, and algorithmic strategies, developers can markedly diminish the vulnerability of IoT devices to SCA attacks. A comprehensive methodology that evaluates the trade-offs among security, performance, and cost is crucial for the effective and practical implementation of SCA protection in real-world scenarios. As the IoT landscape evolves, investing in hardware security expertise and researching advanced VHDL design methodologies are essential for developing secure and reliable connected devices.

DISCUSSION

Internet of Things (IoT) is swiftly proliferating, linking billions of devices, ranging from intelligent thermostats to industrial sensors. This interconnection offers considerable ease and efficiency, yet also presents substantial security threats. Side-channel assaults represent a significant threat, capitalizing on information divulged through a device's physical attributes during operation. VHDL programming provides an effective mechanism for developing resilient countermeasures against these threats at the hardware level.

Conventional security depends on robust cryptographic techniques. Nonetheless, side-channel attacks circumvent these algorithms by scrutinizing physical emanations from the device, including:

- *Power Usage:* Fluctuations in power usage during cryptographic processes may expose confidential keys.
- *Electromagnetic Radiation:* Analogous to power analysis, the examination of electromagnetic emissions might reveal confidential information.
- *Timing Analysis:* Assessing the duration of particular activities can expose data dependencies and compromise the key.
- *Acoustic Emissions:* The sound waves produced by the instrument may occasionally correlate with internal computations.

These attacks frequently employ advanced statistical methods and necessitate physical access to the device, rendering them especially alarming in resource-limited IoT settings where physical protection is typically inadequate.

VHDL (VHSIC Hardware Description Language) is a robust hardware description language employed for modeling and designing digital circuits. Utilizing VHDL, developers can incorporate countermeasures directly into the hardware, enhancing resilience against side-channel attacks. Mechanisms by which VHDL Facilitates Countermeasures Against Side-Channel Attacks:

1. Masking:
   - *Concept:* Introducing random values (masks) during cryptographic operations to obscure the correlation between power consumption and the actual data.
   - *VHDL Implementation:* VHDL allows for the precise control and manipulation of data paths. Random number generators (RNGs) can be implemented in VHDL to generate masks. These masks are then applied to sensitive data using bitwise operations, ensuring that the power consumption is dependent on the mask rather than the actual data.
2. Hiding:
   - *Concept*: Ensuring that the power consumption is constant regardless of the data being processed.
   - *VHDL Implementation:* Techniques like dual-rail logic and clock gating can be implemented in VHDL to equalize power consumption. Dual-rail logic encodes each bit using two wires, one representing the true value and the other the complement. This ensures constant switching activity, and therefore, constant power consumption. Clock gating can be used to selectively disable parts of the circuit not currently in use, reducing power consumption and making it less dependent on the processed data.
3. Balancing:
   - *Concept:* Designing circuits where the transitions between logic states are balanced, leading to a more uniform power consumption profile.
   - *VHDL Implementation:* Careful VHDL coding practices can minimize data-dependent transitions. This involves optimizing the circuit design to reduce the correlation between data values and signal transitions.
4. Timing Randomization:
   - *Concept:* Introducing random delays in the execution of cryptographic operations to disrupt timing analysis attacks.
   - *VHDL Implementation:* VHDL allows for the insertion of variable delays using techniques like inserting random numbers of pipeline stages. This makes it difficult for attackers to precisely measure the execution time of specific operations.
5. Hardware Tamper Detection:
   - *Concept:* Building mechanisms to detect physical tampering attempts on the device.
   - *VHDL Implementation*: VHDL can be used to incorporate sensors and detectors into the hardware design. These could include sensors for voltage deviations, temperature changes, or physical intrusion. If tampering is detected, the device can be configured to erase sensitive data or even self-destruct.

Advantages of Hardware-Based Countermeasures

- *Enhanced Security:* Hardware-based countermeasures are more challenging to circumvent than software-based alternatives.
- *Performance Efficiency:* Countermeasures embedded within the hardware frequently incur negligible performance overhead.

sDecreased Complexity: Delegating security responsibilities to hardware helps streamline software development and minimize the attack surface.

Obstacles and Factors to Consider

- *Complexity of Design:* Implementing effective safeguards against side-channel attacks in VHDL is a complex and demanding endeavor. It necessitates a profound comprehension of hardware architecture and the intricacies of side-channel assaults.
- *Verification and Validation:* Comprehensive verification and validation are essential to ascertain that the applied countermeasures are effective and do not create new vulnerabilities.
- *Cost Overhead:* The implementation of defenses against side-channel attacks may elevate the expenses associated with hardware design.
- *Resource Limitations:* In resource-limited IoT devices, it is essential to equilibrate security with power consumption and performance.

As IoT devices proliferate, the demand for solid security solutions intensifies. VHDL programming provides an effective method for constructing secure hardware capable of resisting advanced side-channel assaults. As researchers advance novel side-channel attack methodologies, ongoing innovation in VHDL-based countermeasures is crucial to uphold the security and integrity of the IoT ecosystem. The future involves creating more advanced VHDL-based systems that are flexible, energy-efficient, and able to tackle the changing threat environment. By adopting VHDL and its functionalities, IoT manufacturers can markedly improve the security of their devices and provide a more reliable and robust IoT infrastructure.

CONCLUSION

For the purpose of designing defenses against side-channel assaults in Internet of Things devices, VHDL programming is a very useful tool. In order to create strong and efficient defenses against a variety of side-channel assaults, VHDL programming makes it possible to design and verify hardware circuits at the digital level. This design and verification process is made possible by the programming language. In addition, using VHDL programming allows for the development and testing of hardware designs prior to their implementation, which helps to ensure that the designs are both secure and efficient. It is impossible to exaggerate the significance of adopting VHDL programming in order to take preventative precautions against side-channel attacks, as the demand for secure Internet of Things devices continues to increase. IoT device makers have the option to enhance the security and dependability of their goods by introducing VHDL programming into the hardware design and development processes. This allows them to earn the trust of customers and ensures the continued success of their products over the long term.

REFERENCES

1. Shaikh HT, Liyakat KKS. Robust access control mechanisms in IoT security using VHDL programming. J VLSI Des Signal Process. 2025;11(2):31–40.
2. Liyakat KKS. VHDL programming for secure true random number generators in IoT security. Res Rev Electron Commun Eng. 2025;2(1):38–47.
3. Ravale PM, et al. Retinal image decomposition using variational mode decomposition. Int Res J Eng Technol. 2018;5(6).
4. Khadake S, Kawade S, Moholkar S, Pawar M. A review of 6G technologies and its advantages over 5G technology. In: Pawar PM, et al, editors. Techno-Societal 2022 (ICATSA 2022). Cham: Springer; 2024.

5. Patil VJ, Khadake SB, Tamboli DA, Mallad HM, Takpere SM, Sawant VA. Review of AI in power electronics and drive systems. In: Proc 3rd Int Conf Power Electron IoT Appl Renew Energy Control (PARC). Mathura, India; 2024. p. 94–9.
6. Khadake SB, Chounde AB, Suryagan AA, MHM, Khadatare MR. AI-driven IoT-based decision-making system for high-blood pressure patient healthcare monitoring. In: Proc Int Conf Sustainable Commun Netw Appl (ICSCNA). Theni, India; 2024. p. 96–102.
7. Mulani AO, Bang AV, Birajadar GB, Deshmukh AB, Jadhav HM. IoT-based air, water, and soil monitoring system for pomegranate farming. Ann Agri-Bio Res. 2024;29(2):71–86.
8. Parihar B, Kiran A, Valaboju S, Rashid SZ, Liz ADRAS. Enhancing data security in distributed systems using homomorphic encryption and secure computation techniques. ITM Web Conf. 2025;76:02010.
9. Veena C, Sridevi M, Liyakat KKS, Saha B, Reddy SR, Shirisha N. HEECCNB: An efficient IoT-cloud architecture for secure patient data transmission and accurate disease prediction in healthcare systems. In: Proc 7th Int Conf Image Inf Process (ICIIP). Solan, India; 2023. p. 407–10.
10. Tamboli DA, Sawant VA, MHM, Sathe S. AI-driven IoT-based decision-making KSK approach in drones for climate change study. In: Proc 4th Int Conf Ubiquitous Comput Intell Inf Syst (ICUIS). Gobichettipalayam, India; 2024. p. 1735–44.
11. Rajendra Prasad K, Karanam SR, et al. AI in public–private partnership for IT infrastructure development. J High Technol Manag Res. 2024;35(1):100496.
12. Liyakat KKS. Detecting malicious nodes in IoT networks using machine learning and artificial neural networks. In: Proc Int Conf Emerg Smart Comput Inform (ESCI). Pune, India; 2023. p. 1–5.
13. Kasat K, Shaikh N, Rayabharapu VK, Nayak M. Implementation and recognition of waste management system with mobility solution in smart cities using Internet of Things. In: Proc 2nd Int Conf Augment Intell Sustain Syst (ICAISS). Trichy, India; 2023. p. 1661–5.

