Project presentation Cryptography with Verilog on FPGAs

Presented by Verilog Vikings

Origin of the creative idea

The objective of using Verilog to implement cryptography on FPGAs is to **achieve high-performance, secure, and flexible hardware-based encryption/decryption solutions. This approach aims to overcome the limitations of traditional software-based cryptography, such as:

- Vulnerability to side-channel attacks.
- Performance bottlenecks.
- Limited flexibility

Project vision and mission

To revolutionize secure communication by delivering high-performance cryptographic hardware solutions implemented in Verilog on FPGAs, enabling faster, more efficient, and cost-effective data protection for a wide range of applications.

01. Implementing industry-standard and novel cryptographic algorithms with exceptional performance and resource efficiency.

02. Providing open-source hardware designs and tools to empower developers and researchers in the field.

03. Collaborating with academia and industry to advance the state-of-the-art in FPGA-based cryptography.

Ideation process

01

Define Project Goals and Scope

02

Research and Algorithm Selection:

03

Hardware Design and Optimization:

04

Testing and Verification:

05

Integration and Applications:

System Overview

Imagine a system where confidential data needs secure transmission over a channel. We can achieve this using an FPGA-based design with three key components:

- **Data Transmitter**: This module takes plaintext data, encrypts it using a chosen cryptographic algorithm.
- **Transmission Channel**: This could be a physical cable, wireless link, or any communication path susceptible to eavesdropping.
- **Data Receiver**: This module receives the transmitted data (potentially intercepted and modified), decrypts it using the corresponding key, and recovers the original plaintext.

Mind map

Verilog Implementation

Verilog, a Hardware Description Language (HDL), allows us to describe the logic and functionality of each module within the FPGA.

Data Transmitter --> Plaintext Input: plaintext data (text, ASCII Codes, etc.) --> Encryption --> Binary Data Formatting.

UART Interface: The formatted data is sent through a Universal Asynchronous Receiver Transmitter (UART) module. UART acts as a serial communication interface, converting parallel data bits into a single stream transmitted over the channel.

Transmission Channel: The data stream travels through the chosen channel

UART Reception: The receiver's UART module retrieves the incoming data stream and converts it back into parallel data bits.

Decryption: The received data is fed into the same cryptographic core with the corresponding decryption key. The core reverses the encryption process, transforming the ciphertext back into the original plaintext.

Weaknesses

- **Choice of algorithm**
- **Implementation errors**

Threats

- **Interception**
- **Manipulation**

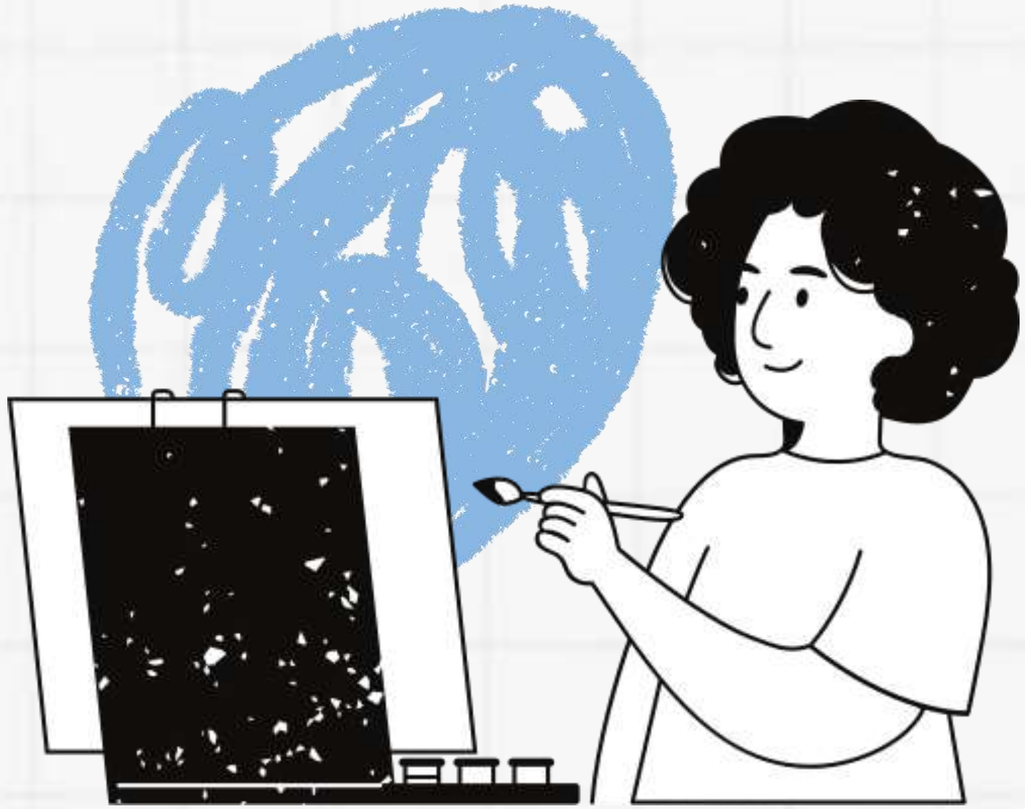
Strengths

- **Rapid Prototyping and Design Iteration**
- **Customization and Optimization**

Opportunities

- **Post-Quantum Cryptography**
- **Formal Verification**

Final reflections and future steps



Final Reflections:

- Power of Verilog & FPGAs for cryptography: Hardware acceleration, customization, parallel processing for high performance and efficiency.
- Resource optimization: Balancing security, performance, resource utilization for practical solutions.
- Advancements in FPGA technology: Increased logic density, clock speeds, on-chip memory for more efficient implementations.

Future Steps:

- Explore novel cryptographic algorithms & protocols: Adapt for efficient Verilog/FPGA implementations, improve security and performance.
- Integrate with other hardware/software: Secure communication protocols, key management systems, trusted execution environments.

References

- Xilinx Cryptographic Cores: https://www.xilinx.com/products/intellectual-property/nav-dsp-and-math/nav-cryptography.html
- Verilog Tutorial for AES Encryption: https://medium.com/@ayushdixithere/revolutionizing-data-security-a-beginners-guide-to-aes-implementation-in-verilog-e2dcc8d2a899
- Design of Secure Systems on FPGAs: https://link.springer.com/book/10.1007/978-90-481-9157-4

Thank you very much!