Encryption Design and Lightweight Development

Robert Herndon, Computer Engineering

Mentor: Dr. Mohamed El-Hadedy Aly

Kellogg Honors College Capstone Project

Introduction

Encryption is used in everyday life, but it has not been optimized for small area and low power devices. It is used to secure communication. This means that other people, unless they are able to crack the encryption, will be unable to understand the message. This hardware design process was done in a team, but encryption is little taught within our curriculum and I had some background with some prior experience. This poster plans to show that encryption at its most basic state is easy to explain and understand when properly introduced. Our complex approach will modify how industry secures small networks and how the encryption can be implemented on any device.

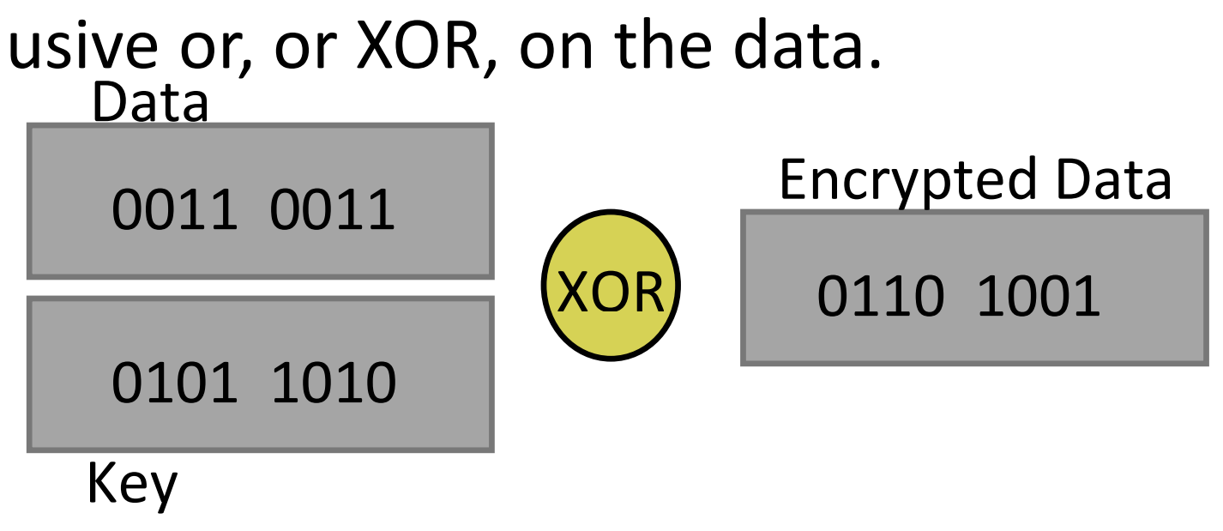
Definitions

Encryption at its most basic level is taking in data and manipulating it to be secured during the transfer of the data through cable or air. The inverse process, decryption, takes the secured data and reverts it back to the data at the beginning of the process. Encryption can be processed on a the physical or the software level, where typically the software approach is used. However, physical encryption systems are more secure for data transfer as it uses a separate processor[1]. The project was focused on design of a physical system.

Lightweight encryption is the application of encryption and decryption to be highly portable and secure. However, it is lightweight because it will take as few resources as needed to complete the process. While modern encryptions are strong, they aren't designed for low power devices, as they are too large to fit on the devices[2].

Teaching Encryption

Encryption is not typically in the curriculum for engineers until graduate level as it is a complex topic to understand and implement. Being able to explain the process and have other people understand the basic principles is key to the design process. The most basic encryption to learn is using an exclusive or, or XOR, on the data.


Data

0011 0011

0101 1010

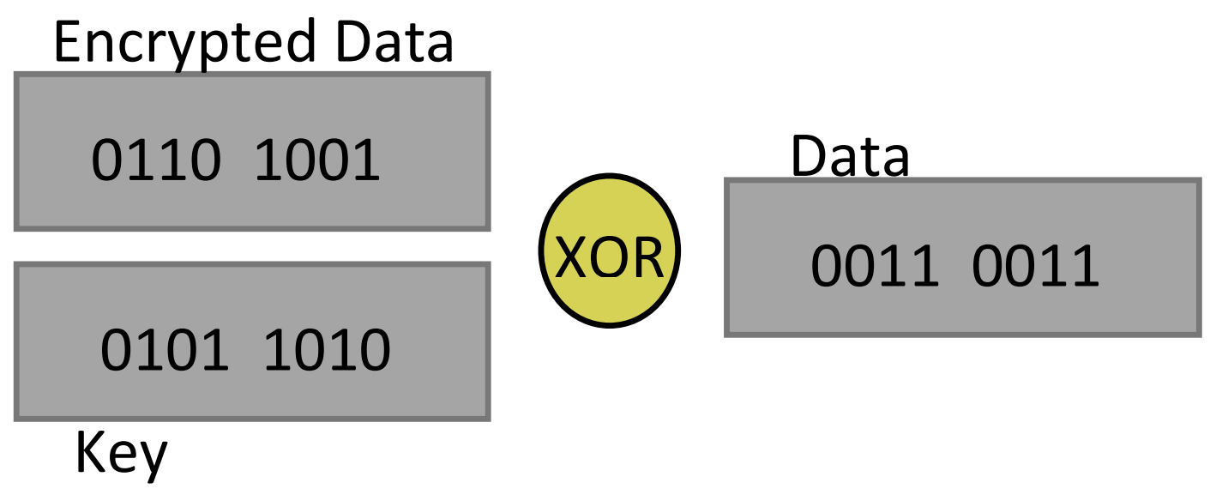
Key

XOR

Encrypted Data

0110 1001

To encrypt the data, a specific key is used to act on the data and change it. In this case, the key is the bottom block. If the same key is used for the encrypted data, the original data will be made at the other side of the XOR.


Encrypted Data

0110 1001

0101 1010

Key

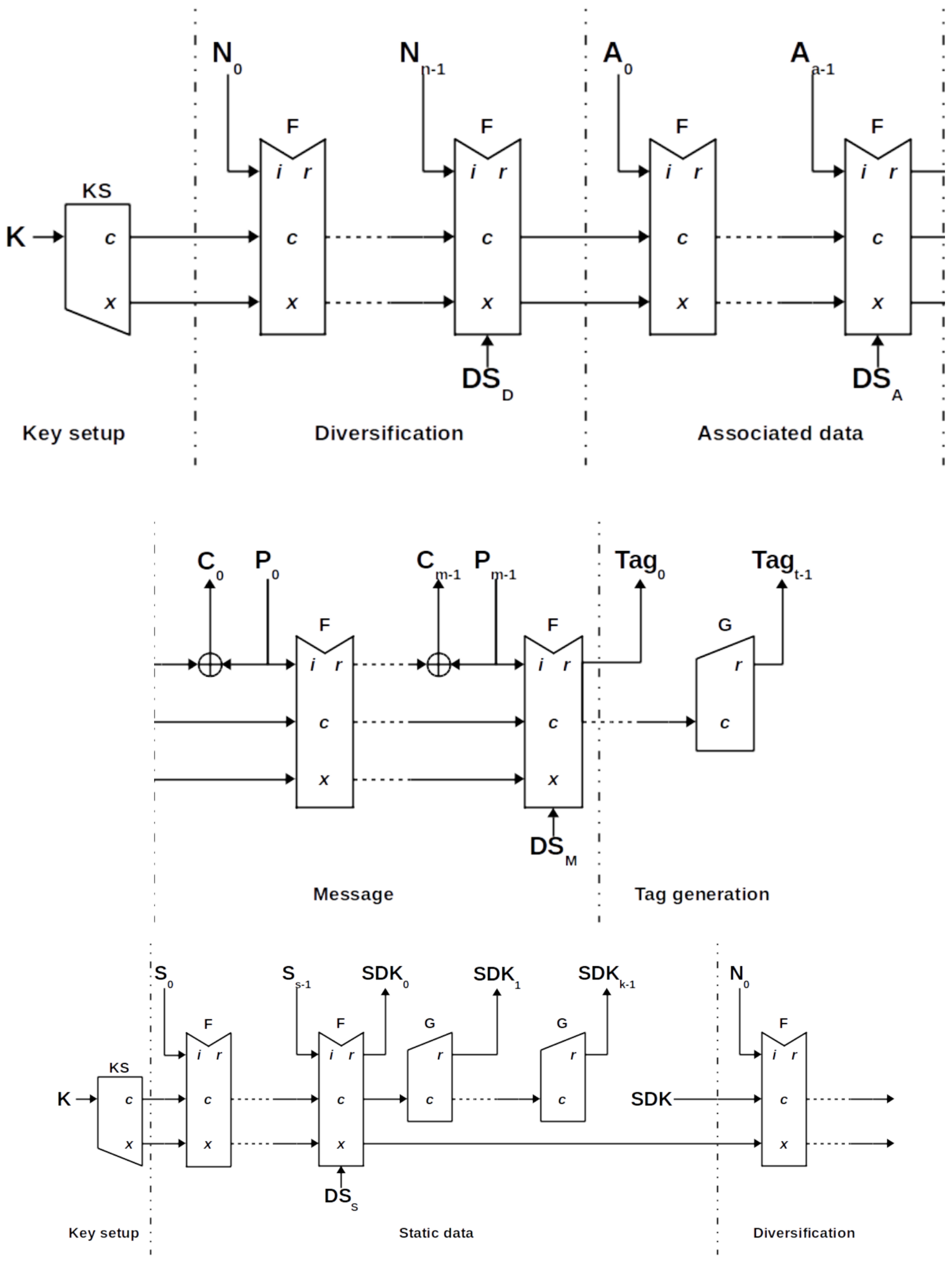
XOR

Data

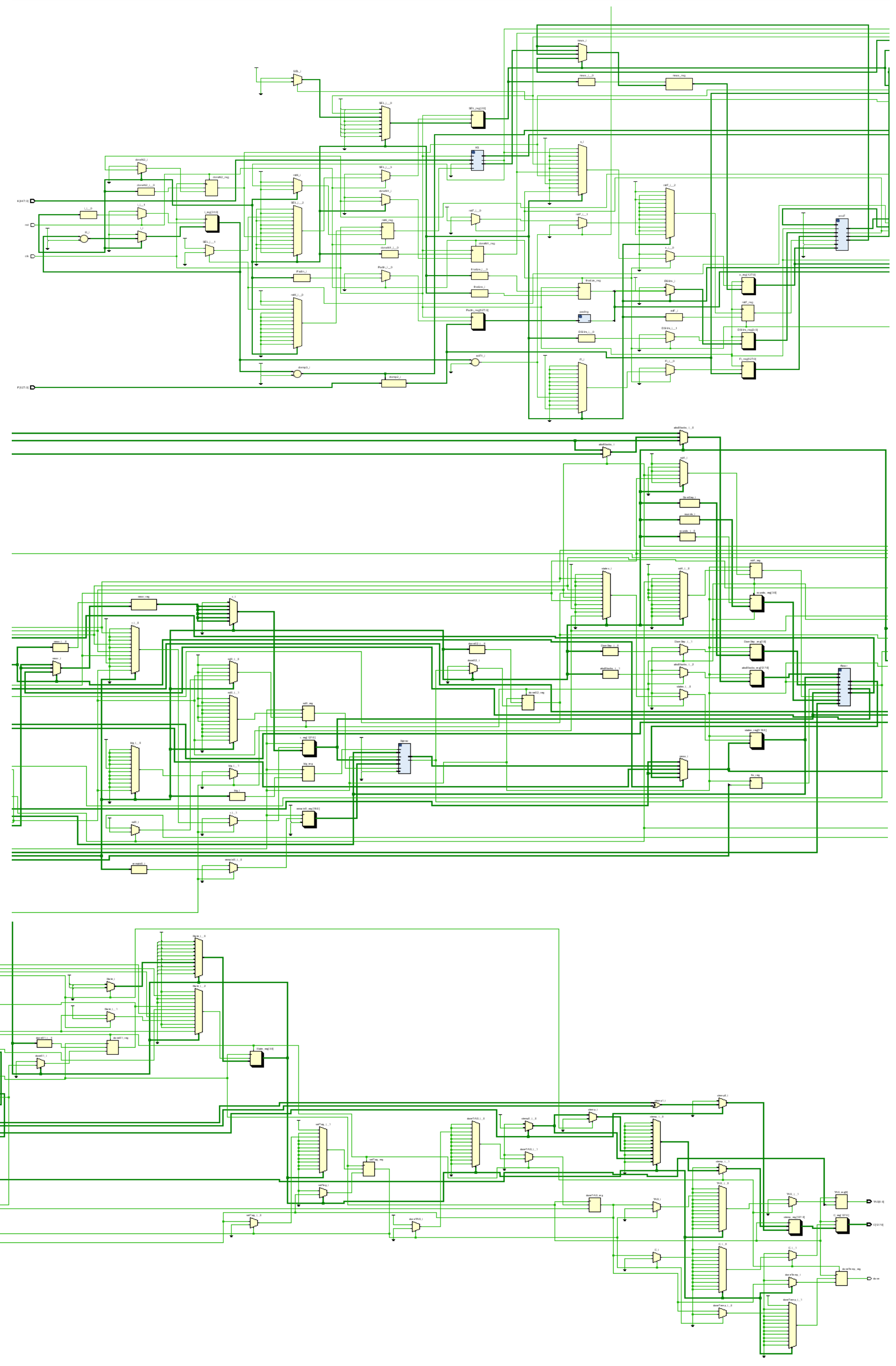
0011 0011

This basic principle of XORing values applies to all encryptions. Using a key, along with other set data, a message can be encrypted and decrypted with the same key and set data. There are specific encryption algorithms that use separate keys to encrypt and decrypt, but the concept of reverting data is still used.

Top Level Design



Top Level Hardware



DryGASCON

The National Institute of Standards and Technology, or NIST, released a competition in 2019 asking for teams to design lightweight algorithms. Sébastien Riou submitted DryGASCON as a design for the competition and the project used this implementation to create a hardware encryption[3]. The images on the left show his main design.

1)Key Setup – This process created a capacity and secret state

2)Optional Static Data – Static data is used to differentiate the capacity between systems

3)Diversification – A nonce, used only once per key, is changing the capacity

4)Associated Data – Metadata is mixed in to the state to diversify the capacity per message

5)Message – In encryption, the cipher text, or encrypted data is made. In decryption, the data returns to normal.

6)Tag Generation – Verification stage to verify in a decryption is properly done, generated in encryption.

The team designed this entire system, however, individual parts were split up and designed separately by members. The parts designed need to be efficient, low power, and low area. As the model is written in C, which runs procedurally, the design needed to be abstracted in order to run as the Hardware Description Language creates parallel hardware[4]. The plan was to place this encryption algorithm on two FPGAs and use them to communicate and send data between two devices.

Conclusion

While the project is not completely finished, a full simulation of the system is completed and has been verified to work. However, when place on the FPGA, the design does not match the simulation and more work is needed to fully complete this algorithm. Work is being done to fix the pieces to be able to run the algorithm on the board. This simulated system is faster and more secure than the one Sebastian made for his FPGA as it runs in a shorter time and uses the static stage. Additional measures that are planned was drone communication, as it is currently not fully secured. When properly implemented, this algorithm should be able to successfully achieve this goal.

References

[1] "What is the difference between hardware vs software-based encryption for secure USB flash drives?" *Kingston Technologies*. Available: https://www.kingston.com/us/solutions/data-security/hardware-vs-software-encryption, Accessed on: March 17, 2020.

[2] "Lightweight Cryptography." *NIST.* Available: https://csrc.nist.gov/projects/lightweight-cryptography, Accessed on: March 20, 2020.

[3] Riou Sebastien, DryGASCON, (2019), DryGASCON,https://github.com/sebastien-riou/DryGASCON

[4] John Sanguinetti, "Abstraction Levels and Hardware Design." *EE Times*. July 27, 2007. Available: https://www.eetimes.com/abstraction-levels-and-hardware-design/, Accessed on: March 20, 2020.