**Vigenère Cipher**

Cameron C. Hingson

Table of Contents

[Overview 1](#_Toc175220838)

[Objectives 1](#_Toc175220839)

[Design 1](#_Toc175220840)

[Implementation 1](#_Toc175220841)

[Verification 1](#_Toc175220842)

[Functional 2](#_Toc175220843)

[Formal 2](#_Toc175220844)

[References 2](#_Toc175220845)

[Citations 2](#_Toc175220846)

[Glossary 2](#_Toc175220847)

[Links 2](#_Toc175220848)

# Overview

This document outlines the process I took while designing, implementing, and verifying the concept of a *Vigenère cipher*.

While pursuing my undergraduate degree, I ran across an article online (which alludes me today, else I would reference it here) of a professional who said that while learning a new programming language, they would implement a Vigenère cipher to test their understand of the given language they were studying.

This implementation of a Vigenère cipher in both VHDL and Verilog, as well as its verification using SystemVerilog, is my test-of-understanding for the tools used to successfully implement the following design.

The Vigenère cipher is a method of *encrypting* alphabetic text where each letter of the *plaintext* is encoded with a different *Caesar cipher*, whose increment is determined by the corresponding letter of another text, the *key* [[1]](#_Citations). Meanwhile, the cipher that plays into the Vigenère cipher, the Caesar cipher, is a type of *substitution cipher* in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet [[2]](#_Citations).

For example, if the plaintext is *binary digit* and the key is *bit*, then encryption is conducted by doing the following:

1. Extend the key to be as long as the plaintext’s character length.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **B** | **I** | **N** | **A** | **R** | **Y** |  | **D** | **I** | **G** | **I** | **T** |
| **B** | **I** | **T** |  |  |  |  |  |  |  |  |  |

↓

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **B** | **I** | **N** | **A** | **R** | **Y** |  | **D** | **I** | **G** | **I** | **T** |
| **B** | **I** | **T** | **B** | **I** | **T** |  | **B** | **I** | **T** | **B** | **I** |

1. Then using a Vigenère table (shown in [Table 1](#_Table_1_–)) or the algebraic representation (shown in [Formula 1](#_Formula_1_–)), encrypt the plaintext character by character.
2. The following is an example of how to encrypt the first three characters of the plaintext using the Vigenère table *(the indices of the table are referenced as (column, row))*:
3. The following is an example of how to encrypt the first three characters of the plaintext using an algebraic expression *(the characters A-Z are encoded as 0-25, respectively. Reference* [*Table 2*](#_Tabel_2_–)*)*:

To encrypt the plaintext character *b* with the key character *b*, convert the characters to their numerical equivalents; b is equivalent to *1*.

To encrypt the plaintext character *i* with the key character *i*, convert the characters to their equivalents; i is equivalent to *8*.

To encrypt the plaintext character *n* with the key character *t*, convert the characters to their equivalents; n is equivalent to *13* and t is equivalent to *19*.

1. The ciphertext ends up being *cqgbzr eqzjb*.

# Objectives

1. Establish an understanding of what a Vigenère cipher is.
2. Create and maintain a document outlining the design, implementation, and verification of a Vigenère cipher .
3. Implement a Vigenère cipher in VHDL and Verilog that can optionally be configured to be scalable as well as operate serially or in parallel.
4. Perform functional verification on the implemented designs via simulation in their respective hardware description languages.
5. Perform formal verification on the implemented designs using SystemVerilog, making sure to perform equivalence checking and assertion-based verification, aiming for ninety percent or greater in coverage.

# Design

# Implementation

# Verification

## Functional

## Formal

# References

## Tables

### Table 1 – A Vigenère Table

Also known as a Vigenère Square or a *tabula recta*.

A close-up of a crossword

Description automatically generated

### Tabel 2 – A-Z Alphabet Mapping to 0-25 Range

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **J** | **K** | **L** | **M** | **N** | **O** | **P** | **Q** | **R** | **S** | **T** | **U** | **V** | **W** | **X** | **Y** | **Z** |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** |

## Formulas

### Formula 1 – Vigenère Encryption Algorithm

The ciphertext character’s numerical equivalent at an index i (Ci) is a function of the encryption key (EK) and the plaintext character’s numerical equivalent at an index i (Mi), which is equivalent to the sum of the plaintext character’s numerical equivalent at index i and the key character’s numerical equivalent at index i (Ki), then modulo this sum with the size of the alphabet *(in this case: 26)*.

### Formula 2 – Vigenère Decryption Algorithm

The plaintext character’s numerical equivalent at an index i (Mi) is a function of the decryption key (DK) and the ciphertext character’s numerical equivalent at an index i (Ci), which is equivalent to the difference of the ciphertext character’s numerical equivalent at index i and the key character’s numerical equivalent at index i (Ki), then modulo this difference with the size of the alphabet *(in this case: 26)*.

## Examples

### Encryption

In both of the Vigenère Table and algebraic methods below, the plaintext is *BINARY DIGIT* and the key is *BIT*.

#### Vigenère Table

1. Extend the key to be as long as the plaintext’s character length.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **B** | **I** | **N** | **A** | **R** | **Y** |  | **D** | **I** | **G** | **I** | **T** |
| **B** | **I** | **T** |  |  |  |  |  |  |  |  |  |

↓

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **B** | **I** | **N** | **A** | **R** | **Y** |  | **D** | **I** | **G** | **I** | **T** |
| **B** | **I** | **T** | **B** | **I** | **T** |  | **B** | **I** | **T** | **B** | **I** |

1. Use the Vigenère Table (shown in [Table 1](#_Table_1_–)) to encrypt each character of the plaintext one-by-one.

To encrypt the plaintext character *B* with the key character *B*, find where the B column and the B row intersect in the Vigenère table. The character at this intersection is the ciphertext character that the plaintext character will be encrypted to. The table yields *C* at the (B, B) index.

A screenshot of a game

Description automatically generated

To encrypt the plaintext character *I* with the key character *I*, repeat the procedure discussed prior. The table yields *Q* at the (I, I) index.

A screenshot of a crossword puzzle

Description automatically generated

To encrypt the plaintext character *N* with the key character *T*, again, repeat the procedure discussed above. The table yields *G* at the (N, T) index.

A screenshot of a crossword puzzle

Description automatically generated

1. After encrypting each plaintext character, assemble them into the full ciphertext.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **B** | **I** | **N** | **A** | **R** | **Y** |  | **D** | **I** | **G** | **I** | **T** |
| **B** | **I** | **T** | **B** | **I** | **T** |  | **B** | **I** | **T** | **B** | **I** |
| **C** | **Q** | **G** | **B** | **Z** | **R** |  | **E** | **Q** | **Z** | **J** | **B** |

#### Algebraically

1. Extend the key to be as long as the plaintext’s character length.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **B** | **I** | **N** | **A** | **R** | **Y** |  | **D** | **I** | **G** | **I** | **T** |
| **B** | **I** | **T** |  |  |  |  |  |  |  |  |  |

↓

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **B** | **I** | **N** | **A** | **R** | **Y** |  | **D** | **I** | **G** | **I** | **T** |
| **B** | **I** | **T** | **B** | **I** | **T** |  | **B** | **I** | **T** | **B** | **I** |

1. X
2. After encrypting each plaintext character, assemble them into the full ciphertext.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **B** | **I** | **N** | **A** | **R** | **Y** |  | **D** | **I** | **G** | **I** | **T** |
| **B** | **I** | **T** | **B** | **I** | **T** |  | **B** | **I** | **T** | **B** | **I** |
| **C** | **Q** | **G** | **B** | **Z** | **R** |  | **E** | **Q** | **Z** | **J** | **B** |

### Decryption

In both of the Vigenère Table and algebraic methods below, the ciphertext is *CQGBZR EQZJB* and the key is *BIT*.

#### Vigenère Table

1. Extend the key to be as long as the ciphertext’s character length.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **C** | **Q** | **G** | **B** | **Z** | **R** |  | **E** | **Q** | **Z** | **J** | **B** |
| **B** | **I** | **T** |  |  |  |  |  |  |  |  |  |

↓

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **C** | **Q** | **G** | **B** | **Z** | **R** |  | **E** | **Q** | **Z** | **J** | **B** |
| **B** | **I** | **T** | **B** | **I** | **T** |  | **B** | **I** | **T** | **B** | **I** |

1. Use the Vigenère Table (shown in [Table 1](#_Table_1_–)) to decrypt each character of the ciphertext one-by-one.

To decrypt the ciphertext character *C* with the key character *B*, find where C resides in the B row. The column letter where the ciphertext character resides is the plaintext character that the ciphertext character will be decrypted to. The table yields *C* at the (B, B) index.

A screenshot of a game

Description automatically generated

To decrypt the ciphertext character *Q* with the key character *I*, repeat the procedure discussed prior. The table yields *Q* at the (I, I) index.

A screenshot of a crossword puzzle

Description automatically generated

To decrypt the ciphertext character *G* with the key character *T*, again, repeat the procedure discussed above. The table yields *G* at the (N, T) index.

A screenshot of a crossword puzzle

Description automatically generated

1. After decrypting each ciphertext character, assemble them into the full plaintext.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **C** | **Q** | **G** | **B** | **Z** | **R** |  | **E** | **Q** | **Z** | **J** | **B** |
| **B** | **I** | **T** | **B** | **I** | **T** |  | **B** | **I** | **T** | **B** | **I** |
| **B** | **I** | **N** | **A** | **R** | **Y** |  | **D** | **I** | **G** | **I** | **T** |

#### Algebraically

1. Extend the key to be as long as the ciphertext’s character length.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **C** | **Q** | **G** | **B** | **Z** | **R** |  | **E** | **Q** | **Z** | **J** | **B** |
| **B** | **I** | **T** |  |  |  |  |  |  |  |  |  |

↓

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **C** | **Q** | **B** | **B** | **Z** | **R** |  | **E** | **Q** | **Z** | **J** | **B** |
| **B** | **I** | **T** | **B** | **I** | **T** |  | **B** | **I** | **T** | **B** | **I** |

1. After decrypting each ciphertext character, assemble them into the full plaintext.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **C** | **Q** | **G** | **B** | **Z** | **R** |  | **E** | **Q** | **Z** | **J** | **B** |
| **B** | **I** | **T** | **B** | **I** | **T** |  | **B** | **I** | **T** | **B** | **I** |
| **B** | **I** | **N** | **A** | **R** | **Y** |  | **D** | **I** | **G** | **I** | **T** |

## Glossary

* Cipher: A secret or disguised way of writing; a code.
* Ciphertext: In cryptography, it is the result of encryption on plaintext in a cipher.
* Cryptography: The practice and study of techniques for secure communication.
* Encryption: The process of transforming or encoding information in a way that only authorized parties can decode.
* Key: In cryptography, it is a sequence of numbers and/or letters used to encode or decode cryptographic data.
* Plaintext: In cryptography, it is unencrypted information.
* Substitution cipher: A method of encryption where parts of the plaintext are replaced with ciphertext in a defined manner with the help of a key.
* Tabula recta: In cryptography and derived from a Latin word, it is a square table of alphabets where each row is made by shifting the previous one to the left.

## Citations

1. Wikipedia Contributors, “Vigenère cipher,” *Wikipedia*, Dec. 25, 2019. <https://en.wikipedia.org/wiki/Vigen%C3%A8re_cipher>.
2. ‌Wikipedia Contributors, “Caesar cipher,” *Wikipedia*, Dec. 10, 2019. <https://en.wikipedia.org/wiki/Caesar_cipher>.
3. “Vigenère Cipher - Decoder, Encoder, Solver, Translator,” Dcode.fr, 2019. <https://www.dcode.fr/vigenere-cipher>.

## Links