AdiPadi2703 Update README.md ce5e36e · 10 months ago

Logisim	Rename Hardware1.circ to Hardw…	10 months ago
Screenshots	Put functional table image into Scr…	last year
Verilog	Rename tb.vcd to test.vcd	last year
Videos	Add files via upload	10 months ago
README.md	Update README.md	10 months ago

End-To-End Encrypted Communication

Team Details

▼ details

Semester: 3rd Semester B.Tech CSE
Section: S1
Member 1: Adithya S Ubaradka, 221CS105, adithyau.221cs105@nitk.edu.in
Member 2: Akshat Mishra, 221CS107, akshatmishra.221cs107@nitk.edu.in
Member 3: Hemang J Jamadagni, 221CS129, hemangj.221cs129@nitk.edu.in

Abstract

▼ details

The process of encoding information, which is conversion of the original representation of the information known as ‘plain text’, into an alternative form known as ‘cipher text’ is called encryption. Encryption does not itself prevent interference but denies the intelligible content to a would-be interceptor.

Problem Statement:

The goal of this project is to design a system of units that communicate between each other via end-to-end encryption. The RSA encryption algorithm will be used and the user will be able to choose the kind of encryption. The aim is to achieve secure communication between two digital systems.

Here's how the RSA algorithm works:

Key Generation:

RSA uses a pair of keys: a public key and a private key. These keys are generated as follows: Choose two distinct prime numbers, typically denoted as p and q. Compute the product of these two prime numbers: The value of n is used as the modulus for both the public and private keys. Calculate Euler's totient function, $\varphi(n) = (p-1)(q-1)$. Select a public exponent (e) such that $1 < e < \varphi(n)$, and e is coprime to $\varphi(n)$, which means they have no common factors other than 1. For ease of arbitration, we have chosen to use the smallest number coprime to $\varphi(n)$ for any given n. Calculate a private exponent (d) such that $(d * e) \% \varphi(n) = 1$. In other words, d is the modular multiplicative inverse of e modulo $\varphi(n)$. The formula we have chosen for d is $d = (k * \varphi(n) + 1) / e$, for some integer k where d is a whole number.
The public key consists of (n, e), and the private key consists of d.

Encryption:

To send an encrypted message, the sender uses the recipient's public key. The message is represented as an integer, usually by breaking it into blocks and converting those blocks to numbers. The sender then computes the ciphertext (C) using the recipient's public key: $C = (M^e) \bmod n$, where M is the plaintext message.

Decryption:

The recipient uses their private key to decrypt the ciphertext. The recipient computes the plaintext message (M) using the private key: $M = (C^d) \bmod n$.

The security of the RSA algorithm is based on the difficulty of factoring the large composite number n into its prime factors (p and q). As long as n is sufficiently large and its prime factors remain unknown, RSA encryption is considered secure. The security relies on the mathematical properties of prime factorization, which is computationally intensive for large integers.

Implementation of the project: In the project, the implementation of this algorithm was done similarly, by dividing the circuit according to the three phases into the Key Generation module, the Encryption module and the Decryption module.

In our chosen design, the Key Generation module will allow any user to choose the value of n and e by adding any two prime numbers they wish, and the output will be the both of the public key’s parts, n and e. The generated private key (d) will be directly fed to the Decryption module, while the public key will be shown to the user on a seven segment display to input into the next module.

Going to the Encryption module, here the user is asked to enter their message of choice and their public key (n,e), which the encryption module turns into the cipher text (C), which is both displayed and fed to the Decryption module.
The Decryption module performs the process of converting ciphertext (C) into the message again, using the private key (d) as input from the Key Generation module and the ciphertext as input from the Encryption module. The message is then displayed as the output of the module to show the success of the decryption.

Why this project?:

WIth the advancement of technology, security of information and data has become extremely relevant. Encryption is a way of manipulating the appearance of data and information to ensure such security from third parties. Depending on the methods of encryption, either the receiver, sender or both will be able to make sense out of encrypted data. Cryptography has become a major area of research. This project was chosen keeping in mind its current importance and relevance. End-To-End Encryption is widely used in instant messaging applications like Whatsapp.

Working

▼ details

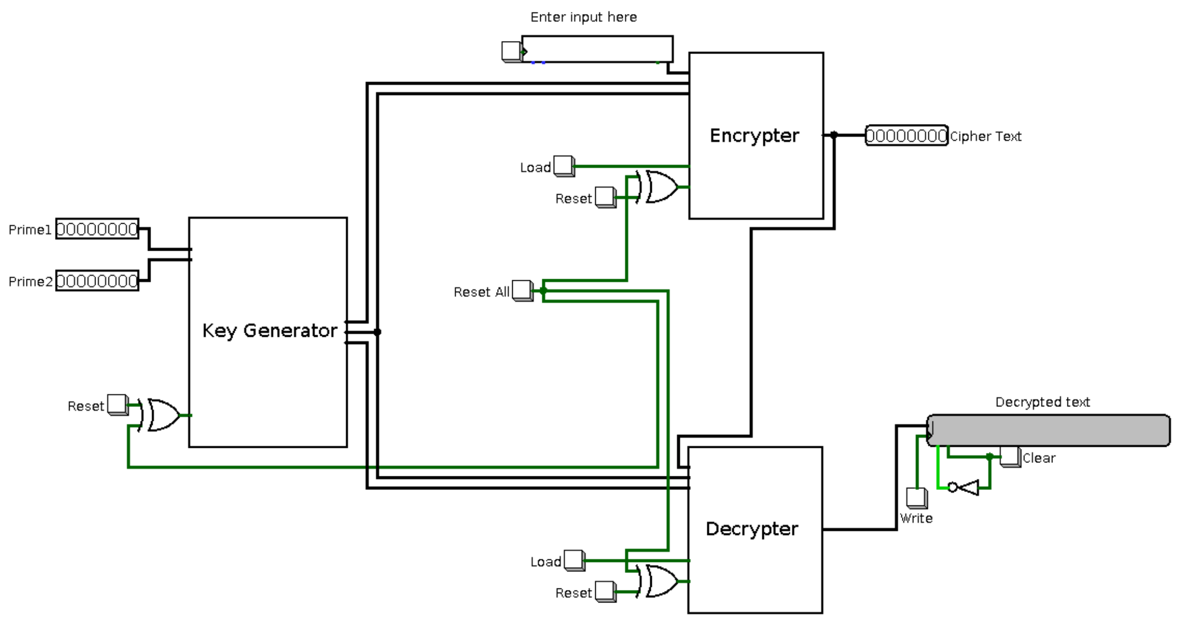
Functional Table:

p	q	e	n	d	M	C
11	101	11	1111	11	A (00)	0
11	101	11	1111	11	B (01)	1
11	101	11	1111	11	C (10)	1000
11	101	11	1111	11	D (11)	1100
11	111	101	10101	101	A (00)	0
11	111	101	10101	101	B (01)	1
11	111	101	10101	101	C (10)	1011
11	111	101	10101	101	D (11)	1100
101	111	101	100011	101	A (00)	0
101	111	101	100011	101	B (01)	1
101	111	101	100011	101	C (10)	100000
101	111	101	100011	101	D (11)	100001

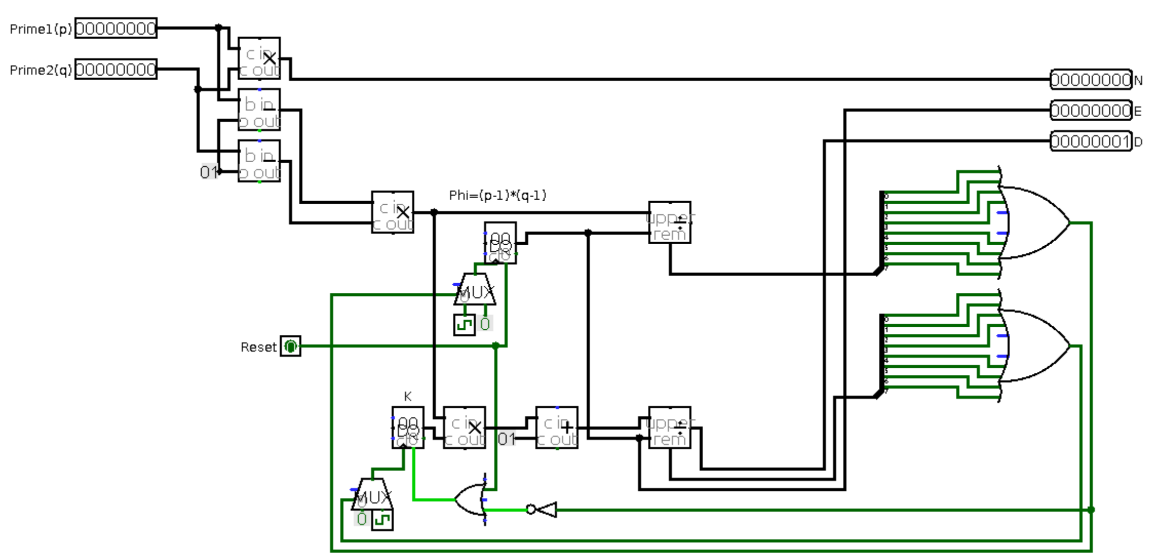
Logisim Circuit diagram

▼ details

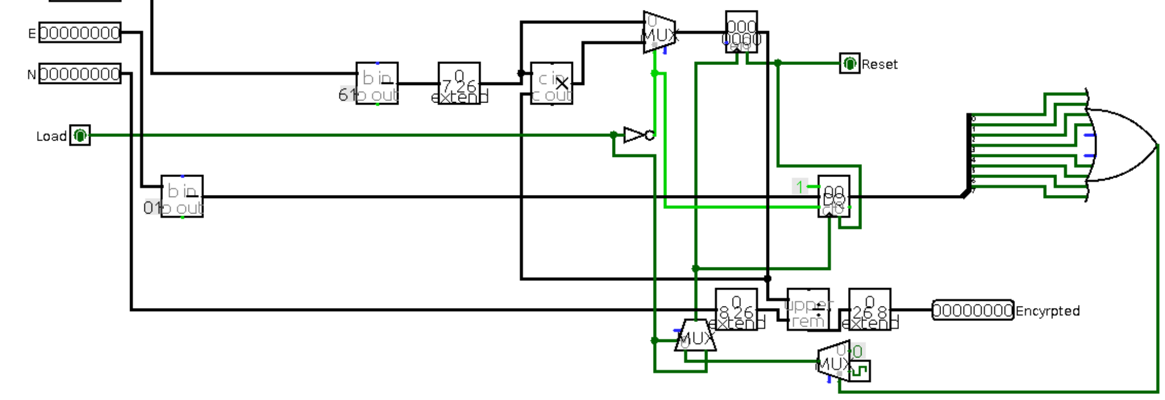
Main Diagram:



Key Generator:

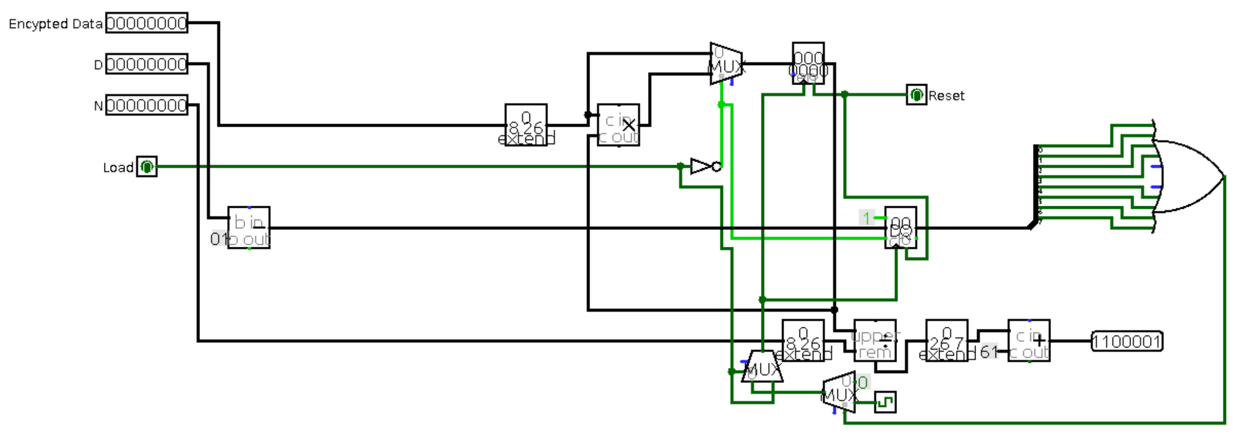


Encrypter:



Decrypter:

README



Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Contributors 4

Kazedaa Hemang J Jamadagni
AdiPadi2703 Adithya Ubaradka
Arsim612
brcnitk Dr. B R Chandavarkar

Languages

● **Verilog** 100.0%