SreeDakshinya /
S1-T18-Small-Scale-DES-Algorithm-Hardware-Implementation

Code Issues Pull requests Actions Projects Security Insights

This repository contains the software files relevant to the simulation of our small-scale DES Algorithm.

0 stars **1** fork **1** watching Branches Tags Activity

Public repository

2 Branches **0** Tags Go to file t Go to file + Add file Code ...

SreeDakshinya Updated README.md 71f4231 · 10 months ago

Block Diagram & Functional …	Added the Block diagram and functi…	last year
Logisim	Added the logisim file	last year
Snapshots	Collapsed the details of each title in…	last year
Verilog	Added .vcd and .gtkw files	last year
Video	Uploaded Hardware video	10 months ago
README.md	Updated README.md	10 months ago

Small Scale DES Algorithm Hardware Implementation

Team details

▼ Detail

```
Semester: 3rd Sem B. Tech. CSE
Section: S1
```

Team members

1. 221CS112, Arjun Ravisankar, arjunravisankar.221cs112@nitk.edu.in, 6360968991
2. 221CS140, Prayag Ganesh Prabhu, prayagganeshprabhu.221cs140@nitk.edu.in, 9353997270
3. 221CS154, Singaraju B V Sreedakshinya, singarajubvsreedakshinya.221cs154@nitk.edu.in, 9606180825

Abstract

▼ Detail

Encryption is the process of converting data into a code to prevent unauthorised access to it. An encryption algorithm converts the original text into an alternative, unreadable form known as ciphertext. Decryption is the reverse process in which the ciphertext is converted back into original text by an authorised user using a key or password, to access the original information.
In the digital era we live in, encryption is vital to ensure the protection of confidential information and messages, financial transactions, classified military communications and matters of national security. The global cyber security landscape has seen increased threats in recent years. Cybercrime has been exhibiting an upward trend globally. Therefore, cryptography is a field of prime importance in these times.

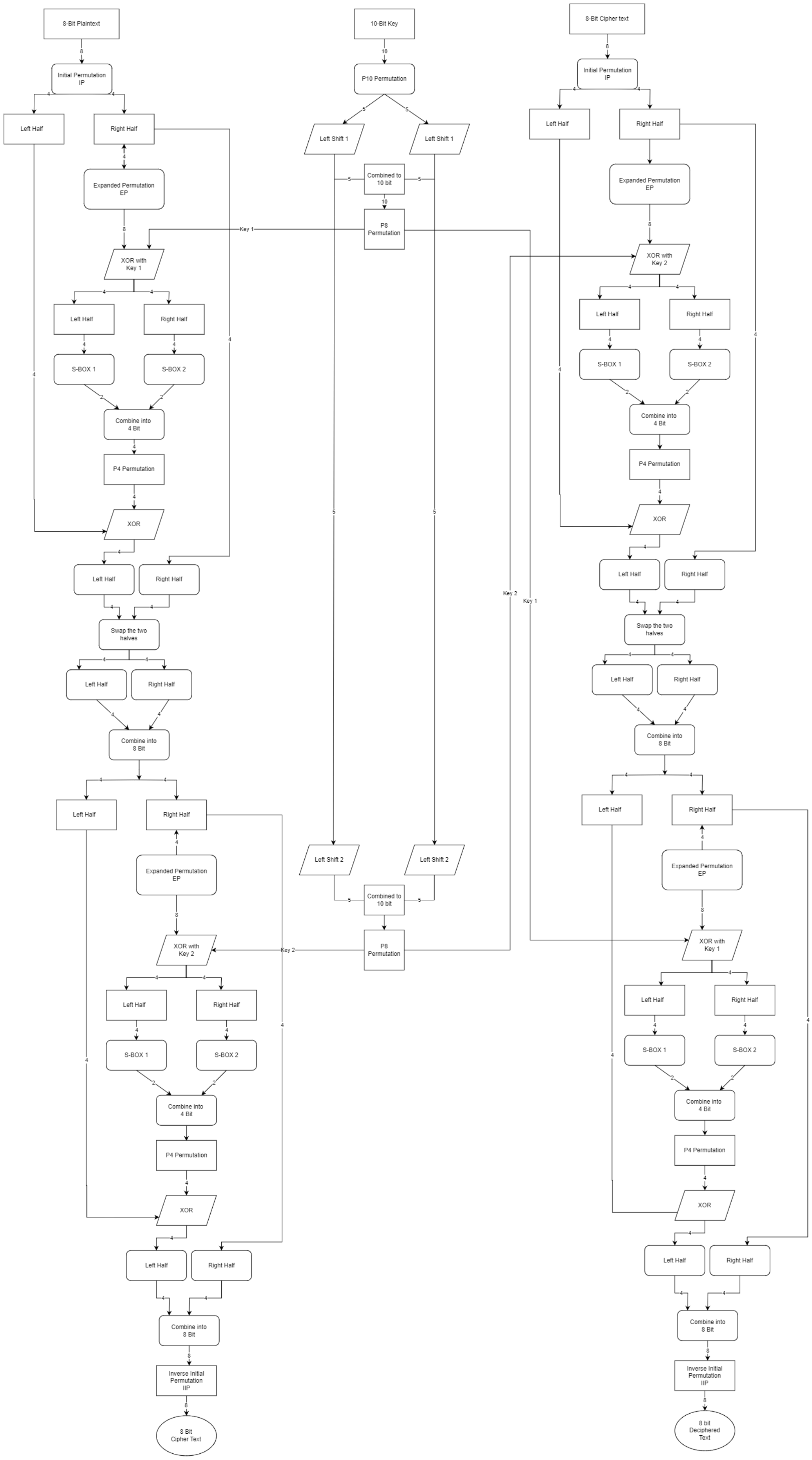
Most high-level encryption algorithms such as DES (Data Encryption Standard) are implemented as software models only. Hardware models are rare, and most of the existing ones use complex components such as FPGAs (Field Programmable Gate Arrays). We decided to implement it as a hardware model utilising simpler components. Hardware models are known to be significantly faster, more secure (resisting timing/power analysis attacks) and efficient than software models. Our model will implement a scaled-down, simpler version of the DES algorithm for the purpose of quick and urgent classified communication. Our choice of DES was due to its well-known status as a standard encryption algorithm and as a highly influential precursor in the development of modern cryptographic techniques and will be a good first choice for hardware implementation.

Working

▼ Detail

The key is passed to the key generator subcircuit. After splitting the bits, left shift and contraction permutation operations are performed to obtain subkeys K1 and K2. The plaintext is passed to the initial permutation subcircuit. Inside the subcircuit, splitting of bits and permutation is done. From the resulting 8 bits, the right half is passed to the round function subcircuit which includes (a) expanded permutation, (b) bitwise XOR with K1 (encryption)/K2 (decryption) (c) substitution using S-boxes operations, (d) transposition (P-box), (e) bitwise XOR with the left half obtained from the initial permutation (f) combination with the right half from the initial permutation. The left half is now swapped with the right half in the "4-bit swap" step. The right half of the resulting 8-bit intermediate is passed to the round subcircuit, in which only the key used for XOR is changed (K2 for encryption and K1 for decryption). The new 8 bit-intermediate undergoes inverse initial permutation and the result is the ciphertext (encryption)/decrypted text (decryption).

Block Diagram


8-Bit Plaintext
Initial Permutation IP
Left Half
Right Half
Expanded Permutation EP
XOR with Key 1
Left Half
Right Half
S-BOX 1
S-BOX 2
Combine into 4 Bit
P4 Permutation
XOR
Left Half
Right Half
Swap the two halves
Left Half
Right Half
Combine into 8 Bit
Left Half
Right Half
Expanded Permutation EP
XOR with Key 2
Left Half
Right Half
S-BOX 1
S-BOX 2
Combine into 4 Bit
P4 Permutation
XOR
Left Half
Right Half
Combine into 8 Bit
Inverse Initial Permutation IIP
8 Bit Cipher Text
10-Bit Key
P10 Permutation
Left Shift 1
Left Shift 1
Combined to 10 bit
P8 Permutation
Key 1
Key 2
Left Shift 2
Left Shift 2
Combined to 10 bit
P8 Permutation
Key 2
8-Bit Cipher text
Initial Permutation IP
Left Half
Right Half
Expanded Permutation EP
XOR with Key 2
Left Half
Right Half
S-BOX 1
S-BOX 2
Combine into 4 Bit
P4 Permutation
XOR
Left Half
Right Half
Swap the two halves
Left Half
Right Half
Combine into 8 Bit
Left Half
Right Half
Expanded Permutation EP
XOR with Key 1
Left Half
Right Half
S-BOX 1
S-BOX 2
Combine into 4 Bit
P4 Permutation
XOR
Left Half
Right Half
Combine into 8 Bit
Inverse Initial Permutation IIP
8 bit Deciphered Text

Functional Table

FUNCTIONAL TABLE													
Encryption													
Plaintext										Key			Ciphertext
Plaintext	IP	Round 1	P-box	L-R XOR 1	4-bit Swap		Round 2	P-box	L-R XOR 2	Key	Subkey K1	Subkey K2	
					Left	Right							
10110110	01111001	0010	0010	0101	1001	0101	0011	0110	1111	0100001101	01001010	10001100	11100111
01101101	11100110	0000	0000	1110	1110	0110	0011	0110	0000	1011110111	10111111	11111011	00011001
0101100	10001010	0010	0010	1010	1010	1010	1000	0001	1011	0101111010	01110001	00110110	11111000
Decryption													
Ciphertext										Key			Decrypted
Ciphertext	IP	Round 1	P-box	L-R XOR 1	4 -bit Swap		Round 2	P-box	L-R XOR 2	Key	Subkey K1	Subkey K2	
					Left	Right							
11100111	11110101	0011	0110	1010	0101	1001	0010	0010	0111	0100001101	10001100	01001010	10110110
00011001	00001110	0011	0110	1010	1110	0110	0000	0000	1110	1011110111	11111011	10111111	01101101
11111000	10111010	1000	0001	1010	1010	1010	0010	0010	1000	0101111010	00110110	01110001	0101100

Logisim Circuit Diagram

▼ Detail

The "Logisim" folder consists of the logisim files of the overall S-DES algorithm circuit.

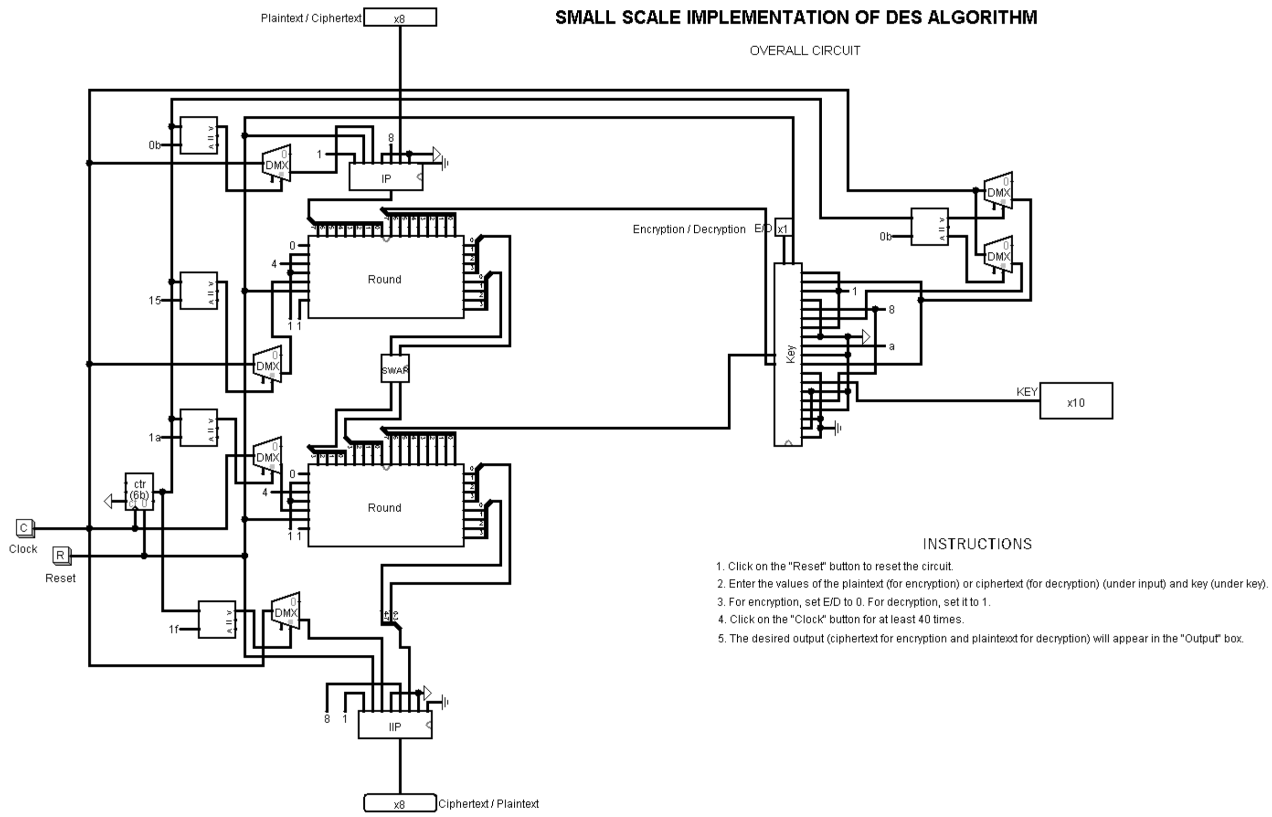
```
    To use the .circ file (Overall circuit):-
    Step 1
        Click on the "Reset" button to reset the circuit.

    Step 2
        Enter the values of the plaintext (for encryption) or ciphertext (for
decryption) (under input) and key (under key).

    Step 3
```

        For encryption, set E/D to 0. For decryption, set it to 1.

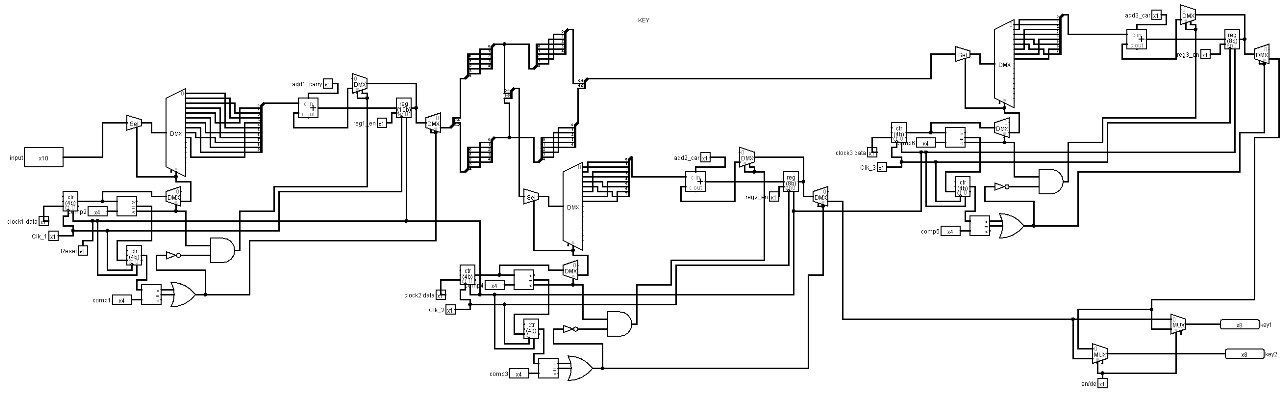
    Step 4
        Click on the "Clock" button for at least 40 times.

    Step 5
        The desired output (ciphertext for encryption and plaintext for decryption) will appear in the "Output" box.

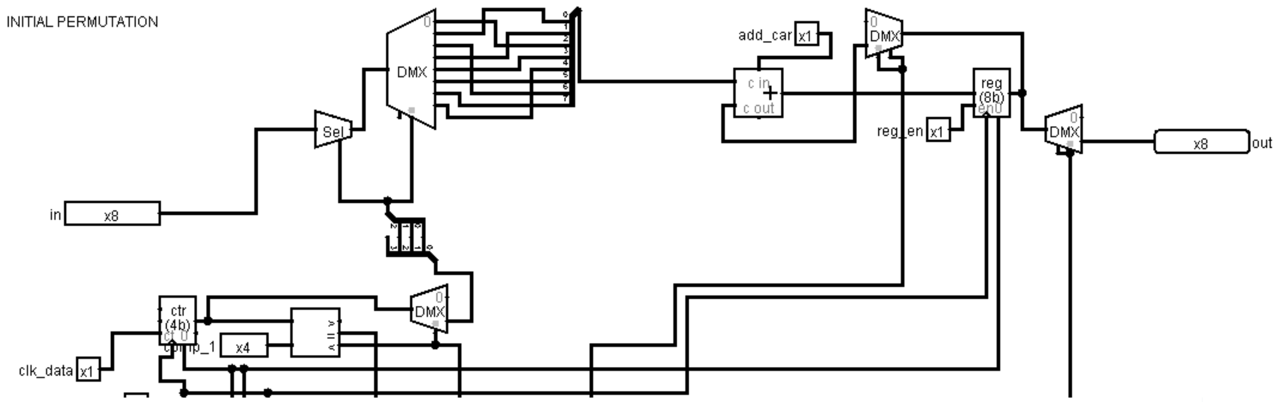
Overall Circuit



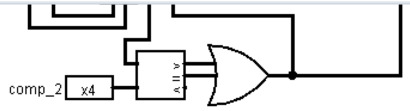
Key Generator Circuit



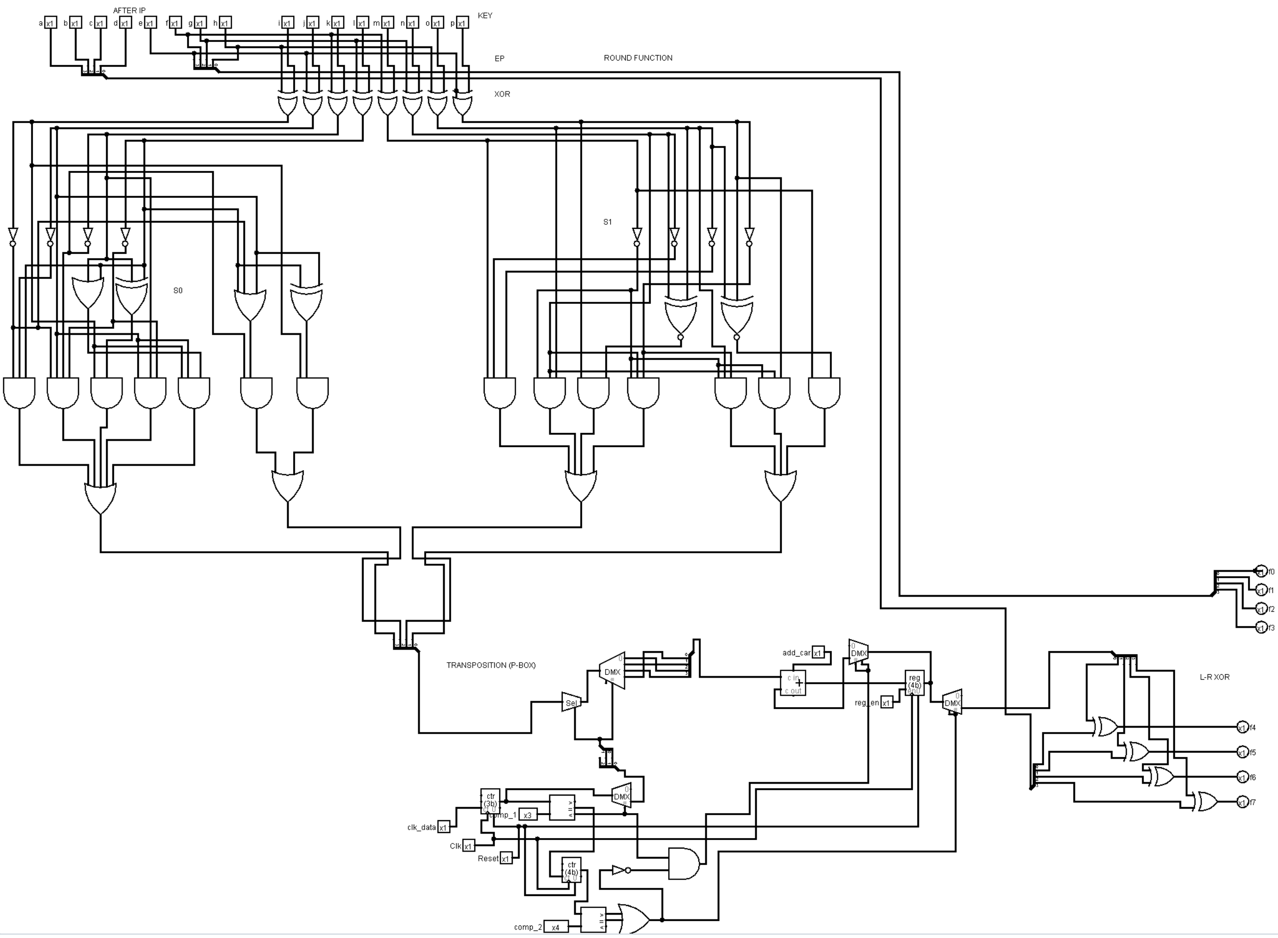
Initial Permutation Circuit



README



Round Function Circuit



Releases

No releases published

[Create a new release]

Packages

No packages published
Publish your first package

Contributors 3

- SreeDakshinya
- PrayagGP
- arjunravisankar001

Languages

- **Verilog** 100.0%