## OFB

This section will discuss the implementation of OFB mode (Output Feedback mode). The main idea of the mode is to use initial vector and key to create a key stream, and then XOR the key stream with plain text to generate cipher text. The discussion will be divided into two part, including encryption and decryption.

# Encryption

The general encryption process of OFB mode shows as following:

**Initial vector**

Encryption

Key

Key stream

Cipher text

Plain text

## Firstly, the initial vector and key will be inputted by user into encryption process. And then the result which generated from the encryption will be XOR with plain text to get cipher text. In terms of encryption process, it consists of five round. For the first four round, substitution, permutation and XOR will be conducted step by step. As for substitution, the statement (initial vector) will be changed into hex format and search the corresponding position in S box. The string in the position will replace the original string. In permutation, the state text will be arrayed in 8 by 8 array. Each column of the array will be shifted downwards circularly by 0,1,2...7 position which depends on the column number. Finally, the state will be XOR with key. Considering round five, there is a bit difference compared to the first four round, the permutation will be canceled. Besides, at the beginning of encryption process, the state will be XOR with initial key. Additionally, the keys of each round are different.

The rule of changing key will be described as below:

Denoting each row of 8 by 8 matrix as a0,a1,a2...,a7, and each row of the processed matrix as b0,b1,b2...,b7, and i as current row. The formulation is

 i % 2 =0

 i % 2 =1

### Decryption

This section will discuss the decryption process of OFB. The general process is very similar to encryption process. Only one difference is, in decryption, the key stream will XOR with cipher text rather than plain text.

**Initial vector**

Encryption

Key

Key stream

+

Plain text

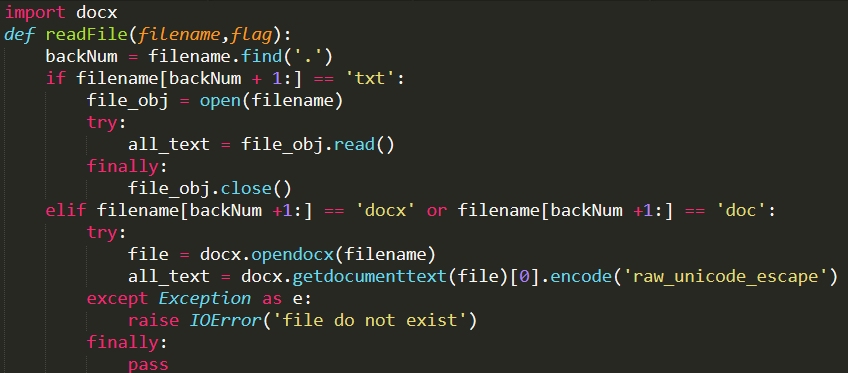
Cipher text

### Implementation

To implement the encryption and decryption, Python is chosen as program language. In order to complete the implementation, the most difficult task is to have a clear mind on the data changed in different format (binary, hex and decimal). It can be helpful for us to get some help from python internal function “bin()”, “hex()” and “int()”. As for permutation, the idea used in implementation is modular. Assuming there are 8 elements, and each elements will shift 2 positions. If modular has been used, the elements will be shifted circularly. For example, position 7 plus 2 modular 8 and the result is 1. Thus, the element in position 7 shift 2 position to position 1.

The next difficulty is encryption and decryption for a plain text consist of several blocks(64bits). To solve this problem, after user have inputted plain text, a text check function will be conducted to check the length of plain text. If the length is not met the requirement(the times of 16), function, named padding, will be applied to add zero into the rest place. After that, the plain text will be divided into several blocks( the number is the times of 16) and pass each blocks as parameters into function encryption(or decryption).

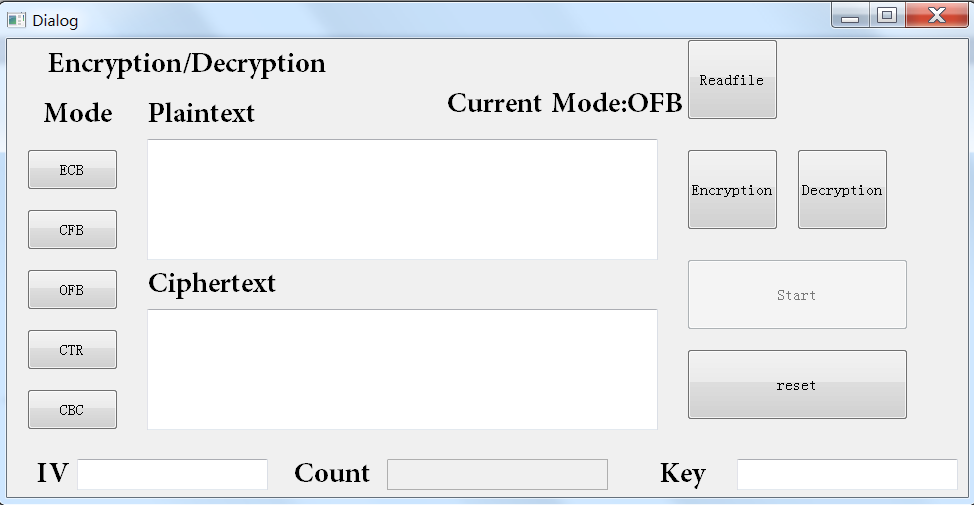
Additionally, for convenient, the encryption and decryption allow user to read file to load plain text(or cipher text). To achieve requirement, Python function “open” will be used.



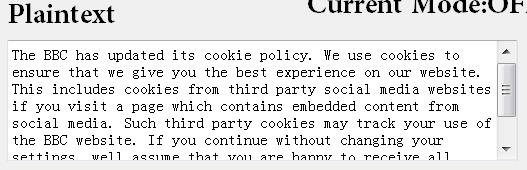
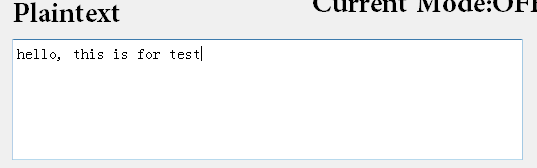
Additionally, to give a direct impression and save user operation, a GUI of encryption and decryption has been developed by using PyQt4.

### Test and Evaluate

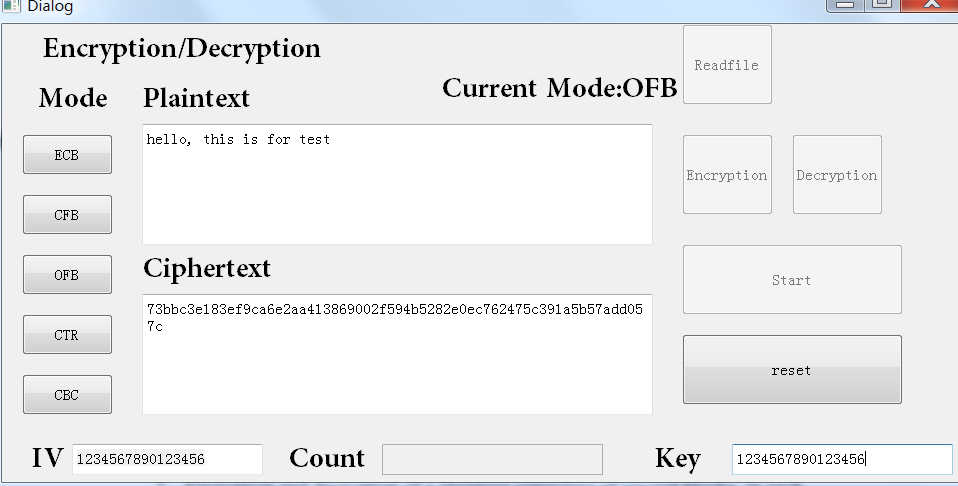
The GUI shows as following:



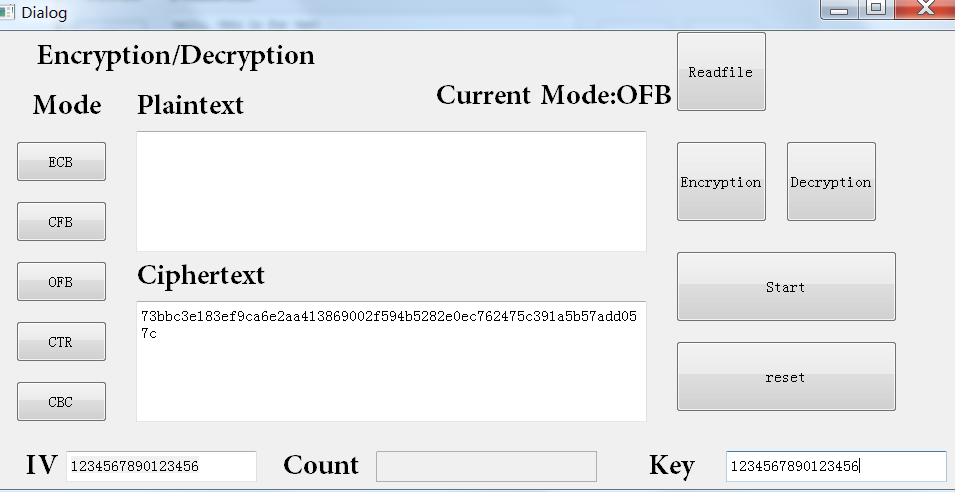
User can choose to input plain text or load plain text from file

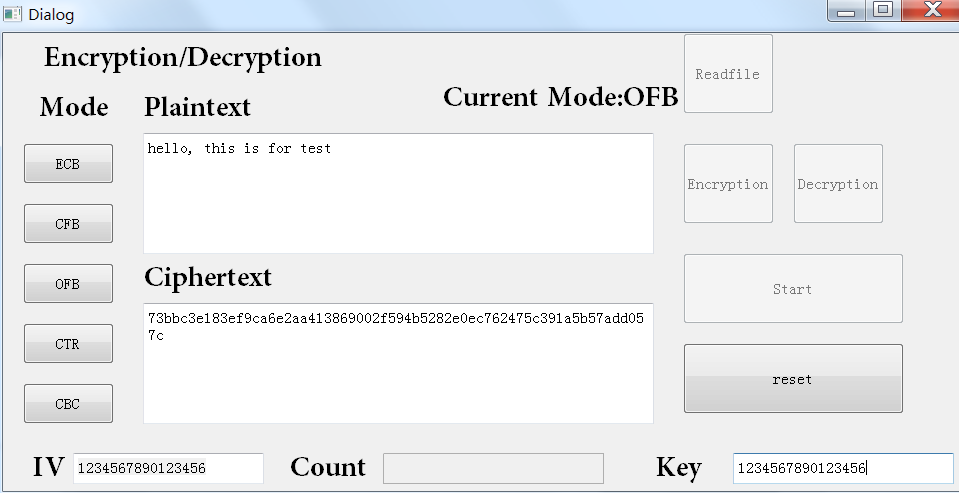
 

After user input initial text and key, button “start” can be pressed, and cipher text will be showed on edit text “Ciphertext”



The same as encryption, the cipher text in decryption can be inputted by user or load from file, and then input initial vector and key, and press start button to start decryption.





The whole time to take for OFB encryption is 0.248999834061 s, which is the same as decryption.

The time for each round is 0.0159997940063 s.