Ryan Fazal 12/14/2018

Daouda Gaye

Necko Bailey

CSCI 360-01 / Fall 2018

Professor Khodjaeva

Topic: Fast Software Implementation: Bit slicing

Programming AES Implementation on bit slicing

What is Bit slicing?

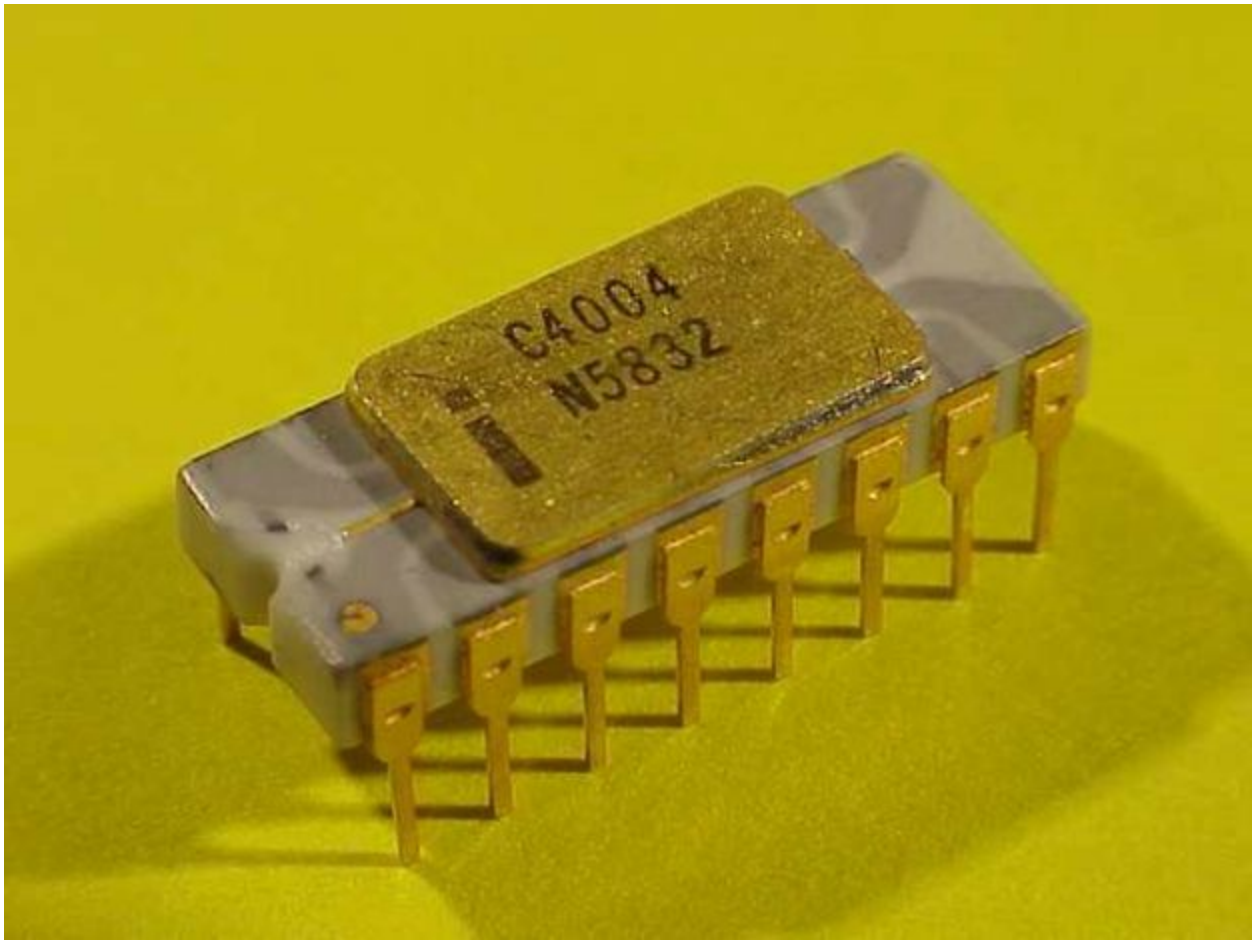
Bit slicing is a mechanism of incorporating modules to multiple the word length. A word in computer architecture, is a unit of data of a defined bit length that can be addressed and moved between storage and the computer chip. Bit slicing was mainly used with early processors specially the AMD (Advanced Micro Devices) 2900 series that was created in 1975.

Each module in a bit-sliced processor has an ALU (Arithmetic Logic Unit). The ALU is capable of handling 4 bit field. By adding more than two identical modules, it is eventually attainable to construct a chip that can hold any multiple of this values such as 8 bits, 12 bits, 20 bits, and so on.

In a bit- sliced processor, each module is named a slice. The regulations for all slices are connected adequately in parallel to share the action work evenly.

Bit slicing was already used by computer experts around 1980, to build more powerful computers. Some experts built processors with the capacity of 64 bits using bit slicing technique. Bit slicing has taken an important facets in the field of encryption methods such as the block cipher (Rouse, 2012).

The picture below is an example of the first microprocessor created in 1971 called “Gray Trace” Intel C 4004. These are some of the earliest versions of the 4004.



Why Bit slicing is important?

Bit slicing has played an important role in micro processing. There are three significant advantages by using bit slicing in micro processing. The first advantage is that the ALU`s

(Arithmetic Logic Unit) can be tied together in horizontal layouts to create computers that can support an important amount of data at a time. Let considers the AMD 2901 is a bit sliced processor which is ALU and the AMD 2910 which stand for CU (Control Unit). In fact, the 2901 was a peer of Intel processor 8080, but could not support 8-bit of data at a time. The 2901 was a 4- bit ALU, but 4-bit 2901 could be linked together to create a computer that could handle 16- bit at a time, 8 bit could also be put together to create a 32- bit computer and so on.

The second advantage is that actually the two chip design permitted the chips to use bipolar chip technology. The good thing about the bipolar chip technology is that it is very fast.

The third and last advantage of bit slice processor is the fact that it has the capacity to permit users to construct their own specification sets for their applications (Antique Tech, 2013).

Advanced Encryption Standard (AES)

In 1997, the National Institute of Standards and Technology (NIST) felt the need to develop a new block cipher. Advanced Encryption Standard (AES) is a block cipher introduced by the NIST as a replacement for Data Encryption Standard (DES) which come into being defenseless to brute-force-attacks. The NIST wanted to make sure that the new advanced encryption algorithm would be unclassified and would be efficient to protecting sensitive government data well into the next century. The development of the AES was open to the public and cryptographers were competing to meet the requirements of the AES proposed by the NIST.

In 2001, NIST announced the block cipher Rijindael as the new AES published it as a final standard (FIPS PUB 197). Rijindael was designed by two young Belgian cryptographers.

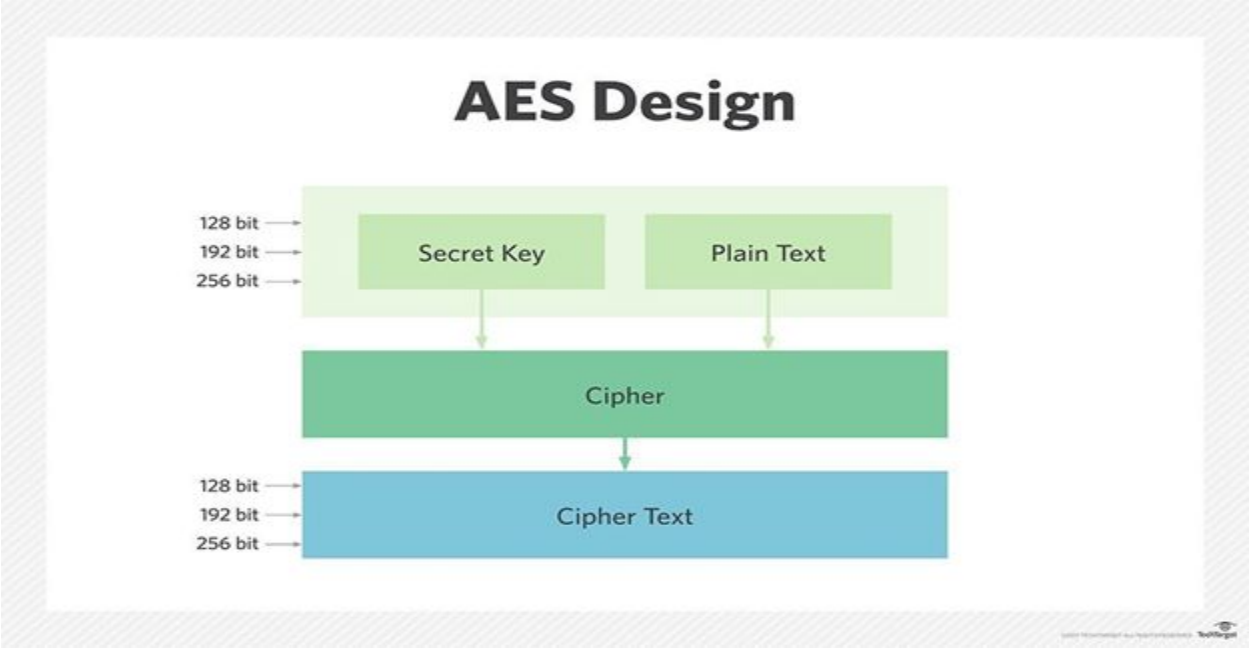
The candidates' proposals must meet the requirements for the new AES proposed by the NIST such that the block cipher with 128 bit block size, three key lengths must be supported 128, 192,

and 256 bits and security relative to other submitted algorithms. It also has to be efficiency in software and hardware (Paal, Pelzl, p88).

The AES became effective as a federal government standard in 2002. The United State government in 2003, declared that AES could be used for protecting classified data. And it later became the default encryption algorithm to protect classified information as well as the first publicly accessible and open cipher approved by NSA for top- secret information. The NSA select AES as one of the cryptographic algorithms to be used by its Information Assurance Directorate to protect national security systems (Rouse, 2017).

How does AES encryption work?

AES constitutes three blocks ciphers: 128, 192, and 256. Each cipher encrypts and decrypts data in blocks of 128 bits using cryptographic keys of 128, 192, and 256 bits respectively (Rouse, 2017).



The picture above explains the AES encryption. Encryption refers to a method of scrambling data so that it is unreadable to others. It also allows for secure storage and transportation over a network. AES encryption uses a key to scramble data. The same key must be used for its decryption. This is the symmetric key algorithm.

The four steps of AES and bit slicing

AES bit slicing is consist of four generally important steps. These step have to all be completed for us to complete the process of bit slicing. The first level the programmer gives the data/message with the key needed. Both are given to the AES Encryption which, in return, gives back a scrambled, unrecognisable message. When decrypting the data the process is very similar, the encrypted message along with its key is supplied to the AES Decryption. In return, this should output the original, decoded message (aka plaintext). diagram one below gives a vivid example of what the encryption process looks like, while diagram two shows the steps of decryption. If you place close attention you will see that the process is very similar.

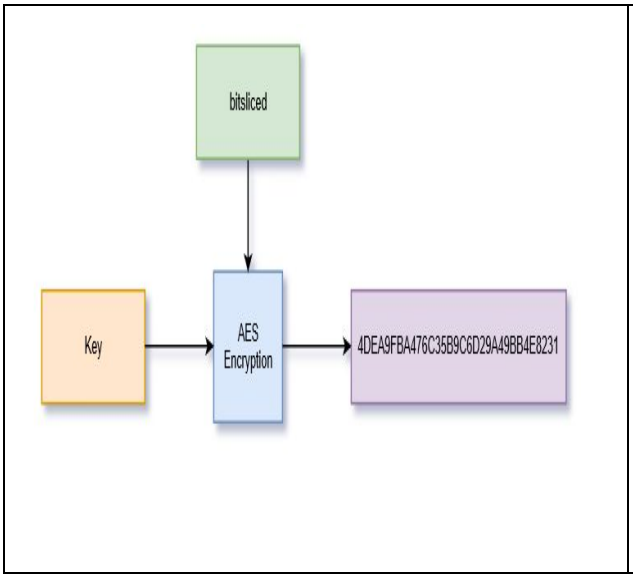


Diagram 1.

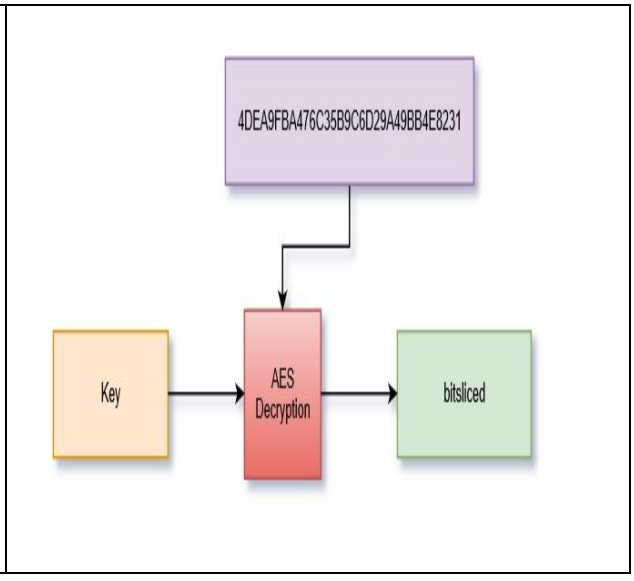


Diagram 2.

AES is made up of a key expansion round and an initial round. These rounds are called the initialisation steps, and is considered the Second Level of the AES process. In this round several rounds of encryption are done using the expanded keys. Is is normally a way to confuse and mislead potential attackers. The key size is normally very important because this is what determine the number of times we would repeat the rounds. The larger the key, the move rounds are performed. The smaller the keys the smaller the number of rounds. Therefore, if you want your encryption to be secure you would typically use more keys. The keys range from 128, 192 and 224 bits. The original AES Algorithm allows for 160 to 224 bits. Though more keys make your encryption more secure, it makes the encryption process slower. This is because it affects the number of cycles. This can be seen in table one below.

128 bit key	10 cycles
192 bit key	12 cycles
256 bit key	14 cycles

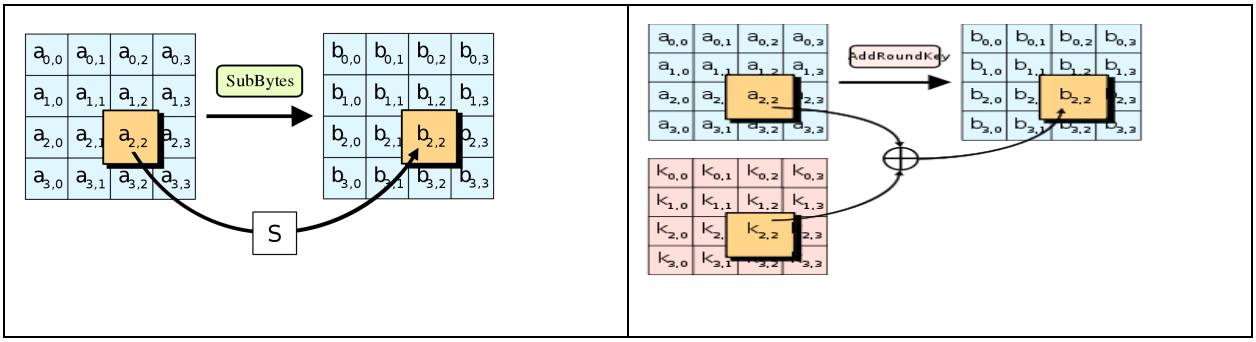
Table 1.

In the AES key expansion each round modifies the previous keys which are products of the original key. The modifications are the key Expansions, and changes the key along with the encrypted message. Each round results is a altered key. This mean the key is able to encrypt itself. The final round however, is the simplest one.

The third level is referred to as the stages within the round. where the shifting and mixing takes place. The mixing is very similar to a XOR. in this round can find your subbytes, the shiftRows mixcolumns and a round key is added. In

In The fourth and final level we do the closing subBytes. This is where we replace each byte with another, base on the key. This is normally done in the Rijndael S-box. The Rijndael S-box is comprises of 256 bit substitution in a 16 by 16 grid form. We normally shift the rows of the state to the left. 1st shift to the left, then the 2nd shift to the left by two bytes, and the third we shift to the left by three byte. These bytes then will reappear on the right, this is referred to as the rotation. The Add round key step is the last and final step in level four, and the AES encryption. This key is added in two places. It's used in the whitening step, and it's also the step in the middle of the inner loop. This process consist of adding the state and the round key using a special type of arithmetics. The arithmetic uses a binary addition modular two. Therefore, every bit in the input is added to the corresponding bit in the round key. And the results mod 2 is stores in the state. This is good because binary addition MOD 2 happens to match a paticular operation which cpu can perform in super fast speed. Below in diagram three is an example of a SubByte, and in diagram four is an

example of AddRoundKey.



Security Faults of AES

The AES, while effective, is not without flaw. Chief among these shortcomings is its vulnerability to attackers since all they require to navigate it is the xor of the plaintext and the first round key. Bit slicing, however, minimizes the impact and replaces the process of (XOR). {If we XOR the polynomial to a result, which 9th bit is 00 (i.e. result does not overflow), then it will overflow since the irreducible polynomial in AES is 100011011100011011 (in bitform). Moreover, an attacker can intercept the secret key if he attacks the key schedule.

Algorithm

The purpose of the algorithm, was to show how Bit slicing is implemented upon the AES encryption. For that, we had to revert back to a basic AES encryption without the dedication of bit slicing.

```
void AES_Encrypt()
{
    int numberOfRounds = 1;

    KeyExpansion();
    InitialRound();

    for(int i = 0; i < numberOfRounds; i++)
    {
        SubBytes();
        ShiftRow();
        MixColumns();
        AddRoundKey();
    }

SubBytes();
ShiftRow();
AddRoundKey();
}
```

Figure 1

As we see from figure 1, we see 4 key statements within the coding. These four are Subbytes, shiftrow, mix column, and addroundkey. These 4 are located within the third and fourth rounds of the AES Encryption.

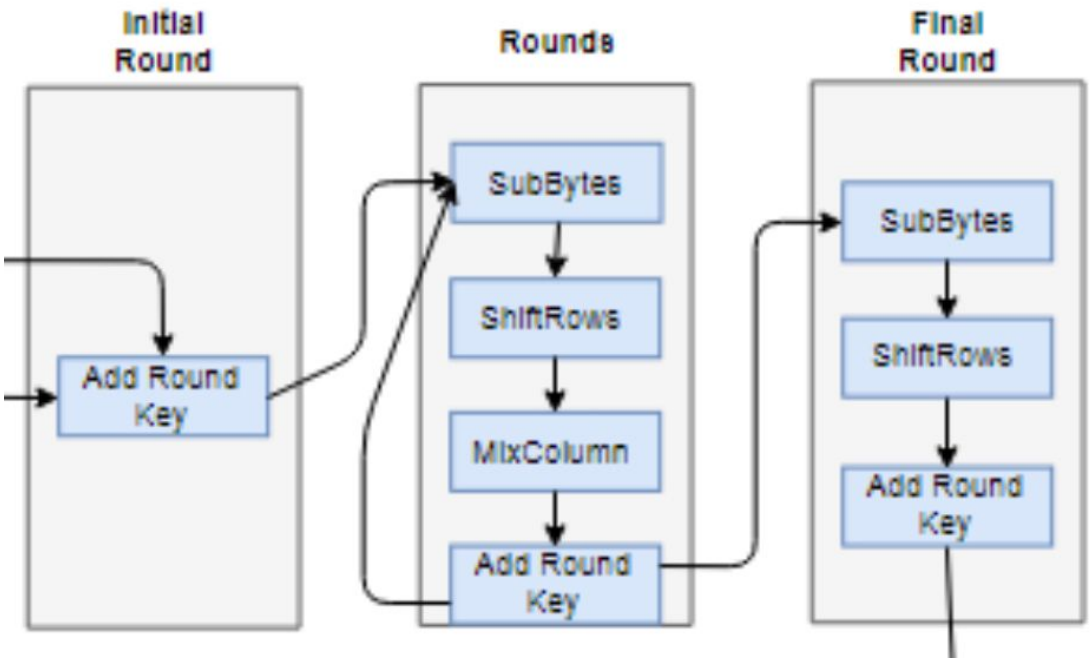


Figure 2

Each round plays an important role as to how the data provided by, say “Alice”, is to be encrypted. For the third level, the user supplies a message and the key which expands into how many keys are needed to the initial round. In this case from figure 1, it’s 128 bits for one round. Within this third level, it is moved onto the rounds, in this case the second as per figure 2, and where the data is shifted and mixed and the rounds are repeated. Here data mixing involves substituting bytes, shifting rows, and adding round key. Afterwards, the data is moved onto the final round of shifting then the encrypted message is the result. Now we move onto the fourth and final level. Within this level, It contains subbytes, shift rows and add round key. The Subbytes replaces each byte with another based on the key. The shift row, for each row after is shifted a byte to the left. The second is by 1, the third is by 2 and the last is by 3. And lastly, the add round key is used in the initial round and looped rounds. It adds the state and round keys.

This formatting of encryption allows us to create a 128,256, and 512 bit encrypted message. However, it was time to see how we can expand on this. To do so, we took the elements of standardized AES encryption, and broke It down even farther.

```
1  #include <iostream>
2  #include <iomanip>
3
4  #include "modes.h"
5  #include "aes.h"
6  #include "filters.h"
7
8  int main(int argc, char* argv[]) {
9
10     byte key[ CryptoPP::AES::DEFAULT_KEYLENGTH ], iv[ CryptoPP::AES::BLOCKSIZE ];
11     memset( key, 0x00, CryptoPP::AES::DEFAULT_KEYLENGTH );
12     memset( iv, 0x00, CryptoPP::AES::BLOCKSIZE );
```

Figure 3

Figure 3 was our first implementation of the Bit slicing on AES. The requirements were much more complex, due to the amount of encryption that was going to take place. Our setup had required a key and an IV. These two were necessary to where they determined how encrypted our file could be. In this case, we set the key length to 128 bits. This is because the 128 bits through bit slicing will be broken down into 8 blocks. Each block will contain 16 bits. However, we could've had 256, or 512 bits. But 128 bits was the easiest to demonstrate and understand. Our IV, or Initialization Vector, is used for our public information. The key created in figure 3 will be exchanged between for example, Alice/Bob before communication begins. Our "DEFAULT_KEYLENGTH" is equal to 16 bytes. Figure 3 is necessary for the allocation of the memory for the bit slicing algorithm.

```
17      std::string plaintext = "Group 6 AES Bit Slicing";
18      std::string ciphertext;
19      std::string decryptedtext;
20
```

Figure 4

For lines 17-20 of figure 4, we must set up the commands for our file. On line 17, “Group 6 AES Bit Slicing” file is partitioned. The String and sink setup is used to break apart the plain text into smaller bits of data

```
24      std::cout << "plaintxt (" << plaintext.size() << " bytes)" << std::endl;
25      std::cout << plaintext;
26      std::cout << std::endl << std::endl;
```

Figure 5

Figure 5 is where we use a dump plain text. A dump plain text is used to create an archive for our plain text.

```
31      CryptoPP::AES::Encryption aesEncryption(key, CryptoPP::AES::DEFAULT_KEYLENGTH);
32      CryptoPP::CBC_Mode_ExternalCipher::Encryption cbcEncryption( aesEncryption, iv );
33
34      CryptoPP::StreamTransformationFilter stfEncryptor(cbcEncryption, new CryptoPP::StringSink( ciphertext ) );
35      stfEncryptor.Put( reinterpret_cast<const unsigned char*>( plaintext.c_str() ), plaintext.length() + 1 );
36      stfEncryptor.MessageEnd();
```

Figure 6

Now we move onto our Cipher text. As figure 6 shows, this is where we input the commands for the creation of Cipher Text. In works with the IV, Cipher Block Chaining takes our specified data, and encrypts It into individual blocks. As we see from line 31, it is creating what is known as a memory copy. With this, it is saving the encrypted message, and the designated key length. When the file is to be located, it will have the coding for line 31 to thank, since it saved it to the system. Line 32 demonstrates what is known as a Cipher block chaining. Because bit slicing involves the sequence of blocks that hold a specified amount of bits, this command allows each block to be encrypted using a cipher key. Each block uses an IV. Line 33 is used for connecting each block. It is to ensure that each block is handled in the proper. If one block were to be lost or removed, the file could not open properly. This is because it would be missing 16 bits (depending on key).

```
41      std::cout << "ciphertxt (" << ciphertext.size() << " bytes)" << std::endl;
42
43      for( int i = 0; i < ciphertext.size(); i++ ) {
44
45          std::cout << "0x" << std::hex << (0xFF & static_cast<byte>(ciphertext[i])) << " ";
46      }
47
48      std::cout << std::endl << std::endl;
```

Figure 7

Just like in figure 5, figure 7 involves the archive of a file. However this is the dump of Cipher Text

Figure 8

For this figure, we can revert back to figure 6. However instead of encryption, we do this for decryption. Line 51, just like line 33 of figure 6 is a memory copy. In this case for figure 8, the algorithm is saving the decrypted file. Line 52 also uses CBC (Cipher block chaining). Just as mentioned in figure 6, each block is necessary so that Bob can see the decrypted message in it's true format. Without it, the data will be incomplete. Thus leaving the output unrecognizable. Lastly line 53 connects all the decrypted blocks together.

Figure 9

Figures 8 and 9 involve the decryption of the encrypted text. This is in the terms of if Bob wants to read the messages, it must be broken down for only his viewing. This is to avoid a malicious attackers from reading his files from Alice. For us, on line 60 of figure 60, we decided to name our file "Our decryptedtxt:." After the completion of the algorithm, it should successfully output "Our decryptedtxt: Group 6 AES Bit Slicing."

Citations

[1] http://cs-exhibitions.uni-klu.ac.at/index.php?id=402

[2] https://eprint.iacr.org/2009/129.pdf

[3] https://eprint.iacr.org/2009/129.pdf

[4]http://www.cs.technion.ac.il/~biham/cv.html

[5]https://www.ru.nl/english/news-agenda/news/vm/icis/cyber-security/2017/peter-schwabe-dutch-prize-ict-research/

[6]https://crypto.stackexchange.com/questions/35132/how-can-bit-slicing-be-constant-time-when-mix-columns-is-in-the-cipher

Rouse, M. (2012, July). Bit slicing. Retrieved fromhttps://whatis.techtarget.com/definition/bit-slicing

Antique Tech. (2013). Bit Slice, or Bit Slice, Microprocessor. Retrieved from http://www.antiquetech.com/?page_id=593

Rouse, M. (2017, March). Advanced Encryption Standard (AES). Retrieved from https://searchsecurity.techtarget.com/definition/Advanced-Encryption-Standard

Paar, Christof, et al. Understanding Cryptography A Textbook for Students and Practitioners. Springer Berlin, 2014.