Proposed Work

As per the scenario, special considerations have been made for the proposed algorithms to be as simple as possible so as to run efficiently with slow processors and low memory. At the same time, attention has been given to design robust algorithms to face the challenges of various types of cryptanalytic attacks.

During this research work, seven different techniques/algorithms have been developed and implemented through microprocessor-based system, which are as follows :

[1] Prime Position Orientations (PPO)
[2] Selective Positional Orientation of Bits (SPOB)
[3] Block Exchange Technique (BET)
[4] Modulo-Arithmetic Technique (MAT)
[5] Overlapped Modulo-Arithmetic Technique (OMAT)
[6] Modified Modulo-Arithmetic Technique (MMAT)
[7] Bit-pair Operation and Separation (BOS)
[8] Decimal Equivalent Positional Substitutions (DEPS)

The first three algorithms are transposition ciphers, whereas the rest are substitution ciphers. The BOS technique also involves some transpositions in addition to substitutions.

All the proposed algorithms are block ciphers and have been implemented for bit streams of 256 or 512 bits. The algorithms can be easily enhanced to work with a higher stream size for better security. All these ciphers have several rounds, each round working on a particular block size. The encryption/decryption is started with a small block-size, say 8bits, and doubling it in each round thereafter reaching the last round with block size 256 or 512. In some cases, the decryption starts with the maximum block size and moves down to the minimum block size. All the algorithms have been put through several statistical tests for analysing their weaknesses, and hence the strength. A good degree of confusion and diffusion within the bit stream has been noticed in all of these ciphers. The following process was adopted for each proposed block cipher.

a) Writing a C language program for the purpose of analysis.
b) Write an 8085 assembly language program for implementation.
c) Analyze the strength of the cipher by running several test algorithms.

The C language programs were needed to encrypt a set of chosen files and analyze the blocks ciphers through various perspectives. The methods used for testing the algorithms are discussed at the end of this chapter. Outlines of the proposed block cipher techniques are presented in the following sections.

Prime Position Orientations (PPO)

In this technique, the bit stream is first divided into blocks of 8 bits each. The positions of the bits within a block are numbered 1 to 8 starting from the MSB. The bits whose positions are found to be prime numbers (2, 3, 5, and 7 in this case) are picked up for transposition. Three different techniques have been developed depending on whether the prime positional bits are pushed to the front, or the rear, or divided among the front and the rear portions of the block being considered. The three options can be used in a cascaded manner, or one of them can be chosen for each block depending on the key.

The process is repeated for several rounds, each time doubling the block size. It was also noticed that the original block of a particular round would be regenerated if that round was reiterated several number of times. Each round was given several iterations and the number of iterations formed a part of the key. During decryption, the key was used to give the remaining iterations to get back the original bit stream.

Selective Positional Orientation of Bits (SPOB)

In this technique, the stream of bits are divided into a number of blocks each containing n bits, where n is one of 8, 16, 32, 64, 128, or 512, depending on the round. Within each block, a pair of bits is selected using the rules of proposed technique, and swapping is performed among the bits in each pair. The selection of the bits in each pair is altered in every pass. Several passes will constitute a round.

If the whole process is performed repeatedly for a particular block size, the original block is regenerated after a finite number of iterations. One of these iterations is selected to generate the encrypted block and hence the corresponding encryption key. As the technique is symmetric, the same operation is performed for decryption.

Block Exchange Technique (BET)

In this technique, the encryption is performed through a multi-round cascaded system. In the first round, the message is taken as blocks of 1 bit each, such as (a), (b), (c), (d), (e), (f), (g), (h) and so on, the letters a, b, c, d etc. denoting either 0 or 1. These blocks are divided among several sets, each set containing 4 contiguous blocks. The second and the third elements (blocks) in each set are then exchanged. So the message is converted to (a), (c), (b), (d), (e), (g), (f), (h) and so on. In the second round, the block size is doubled making blocks as {(a, c), (b, d), (e, g), (f, h)}. The same exchange technique is applied once again to this intermediate stream converting the message to (a, c), (e, g), (b, d), (f, h). The process is repeated, each time doubling the block size, up to 512-bit block size or more. The same process is applied for decryption.

Modulo-Addition Technique (MAT)

In this case, the original message is considered as a stream of bits, which is then divided into a number of blocks, each containing n bits, where n is any one of 8, 16, 32, 64, 128, 256. If B_1, B_2, B_3, B_4,… are the blocks in the stream, then they are paired as (B_1, B_2), (B_3, B_4), and so on. The two adjacent blocks in each pair are then added where the modulus of addition is 2^n. The result replaces the second block, first block remaining unchanged. The modulo-addition has been implemented in a very simple manner where the carry out of the MSB is discarded to get the desired result. The technique is applied in a cascaded manner by varying the block size from 8 to 256. The whole technique has been implemented by using a modulo subtraction technique for decryption.

Overlapped Modulo-Addition Technique (OMAT)

This is also an enhancement of the simple MAT algorithm. In this case, the way the blocks are paired is different. The block-pairs overlap with each other and hence the name of the technique. Operations are carried out in block-pairs (B_1, B_2), (B_2, B_3), (B_3, B_4), and so on. Except for fist and the last block, each block is common to two adjacent pair of blocks.

Modified Modulo-Addition Technique (MMAT)

This is a modification of the MAT algorithm where the modulo-addition is carried out twice for each pair of blocks. Unlike the MAT algorithm, both the blocks are replaced with the result of the addition. The result of the fist addition replaces one block while the other block is replaced by the result of the second addition. The degree of confusion and diffusion is higher than simple MAT.

Bit-pair Operation and Separation (BOS)

In this technique also, the original stream of bits is divided into blocks of $n=2^k$ bits each, where k is 3, 4, 5, 6, 7, 8, 9, and so on, for each round. Within each block, two adjacent bits are paired and two different operations are performed in each pair. The result of the first operation is placed in the front and of the second operation in the rear. The encryption is started with block size of 8 bits and repeated for several times and the number of iterations also forms a part of the key. The technique is applied in a cascaded manner doubling the block size each time. The same process is used for decryption.

Decimal Equivalent Positional Substitutions (DEPS)

The original stream of bits, in this case also, is divided into a number of blocks, each containing n bits, where n is any one of 8, 16, 32, 64, 128, or 256. The decimal equivalent of the block under consideration is evaluated and checked whether the integer value is even or odd. The position of that integer in the series of natural even or odd numbers is evaluated. A '0' or '1' is pushed to the output stream depending on whether the integer is even or odd, respectively. The process is carried out recursively with the positional values for a finite number of times, equal to the length of the source block. During decryption, bits in the target block are to be considered

along LSB-to-MSB direction after which we get an integer value, the binary equivalent of which is the source block. The sweetness of the technique lies in its microprocessor-based implementation where no conversion to decimal and no calculations to ascertain whether the decimal value is even or odd and to find its position in the series of odd or even numbers are needed, no matter how long the block is.

Methods of Evaluation

Although no standard methods are available for testing the strength of a symmetric cipher, several tests suggested in popular journals have been used to examine the vulnerability of the encryption algorithms proposed in this thesis. These methods are explained briefly in the following sections. The results of these tests for the proposed ciphers have been compared with those obtained for Triple DES algorithm, which has been used as a benchmark.

Character Frequency Distribution

One way to judge a symmetric cipher is to examine the frequency distributions of all the 256 ASCII characters in the source and the encrypted files. For this purpose, a number of files in each of the categories like .txt, .exe, .dll, .jpg etc. have been chosen and are subjected to encryption using the proposed algorithms. The frequencies of characters in the source and the encrypted files are computed and then analysed to check whether the characters in the encrypted file are more or less equally distributed in the 0 to 255 range compared to that of the source file.

Heterogeneity of the Source and the Encrypted Files

This test has been applied to test whether the source and the encrypted files are heterogeneous or not. There many ways for comparing any two files. Merely comparing the two files will not be enough. The degree of difference between the files being compared is the actual measure of heterogeneity.

The best way to do this is to compare the frequencies of each character in the source and the encrypted files to ensure there is a good amount of difference. In other words, the frequencies obtained in the previous test may be used to apply the most popular χ^2-test (chi-square test). A high χ^2 value for a pair of source and encrypted files will indicate heterogeneity between those files.

Avalanche Test

In this test, a binary string is encrypted several times, each time with a small modification. At first, the original string is encrypted without any modification. In the subsequent steps, the binary string is encrypted for a number of times, equal to the length of the string, each time complementing one bit. The encrypted strings are examined to ensure that the change in one bit of the source string more or less affects the whole encrypted string. Simply speaking, this test checks the diffusion property of the encryption algorithm.

Runs Test

It must be ensured that the attempts of a cryptanalyst to look for patterns in the cipher-text, so as to deduce the corresponding plain-text, are frustrated. This achieved by creating a good amount of confusion as discussed in section 1.5. This test uses the strings obtained in the avalanche test to compare the number of *runs*. A *run* of length *i* in a binary *n*-tuple is an *i*-tuple of consecutive bits (1's or 0's) not preceded or succeeded by the same bit. The number of runs in an *n*-tuple ranges from 1 to *n*.

Microprocessor-based Implementation

The proposed algorithms have been implemented in an Intel 8085 microprocessor-based system. The schematic view of the implementation is given by the block diagram in figure 1.6.

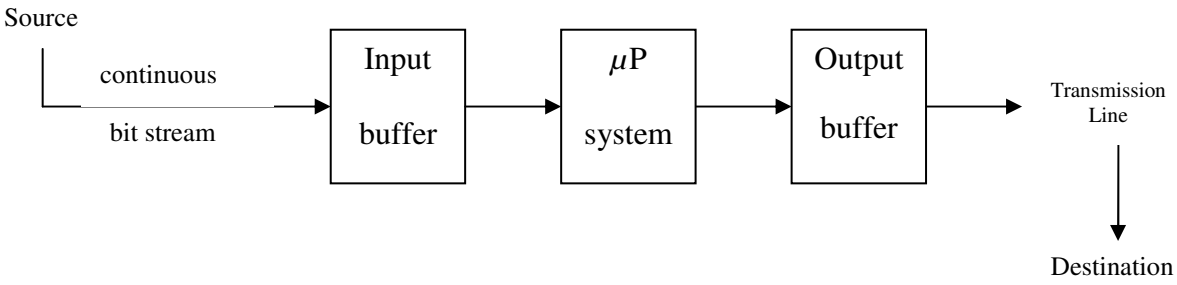

Figure 1.6: A microprocessor- based implementation

The incoming bit stream from any source will be captured and stored in the input buffer. The microprocessor-based system takes the input from the buffer, generates cipher-text following the particular algorithm being used and sends to output buffer. The cipher-text is sent for the destination through a transmission line.