

**Experiment No. 3**

**Title:** A5/1

**Batch: A3 Roll No.:16010421073 Experiment No.:3**

**Aim:** To implement stream cipher A5/1

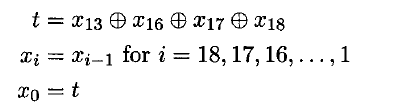
**Resources needed:** Windows/Linux

**Theory: Pre Lab/ Prior Concepts:**

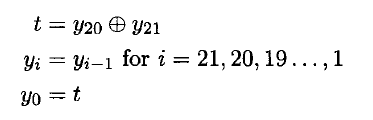
A5/1 employs three linear feedback shift registers , or LFSRs, which are labeled X, Y, and Z. Register X holds 19 bits, (x0,x1…x18)-The register Y holds 22 bits, (y0,y1…y21) and Z holds 23 bits, (z0,y1…z22) ·Of course, all computer geeks love powers of two, so it's no accident that the three LFSRs hold a total of 64 bits.

Not coincidentally, the A5/1 key K is also 64 bits. The key is used as the initial fill of the three registers, that is, the key is used as the initial values in the three registers. After these three registers are filled with the key,1 we are ready to generate the keystream. But before we can describe how the keystream is generated, we need to say a little more about the registers X, Y, and Z.

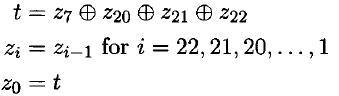
When register X steps, the following series of operations occur:



Similarly, for registers Y and Z, each step consists of



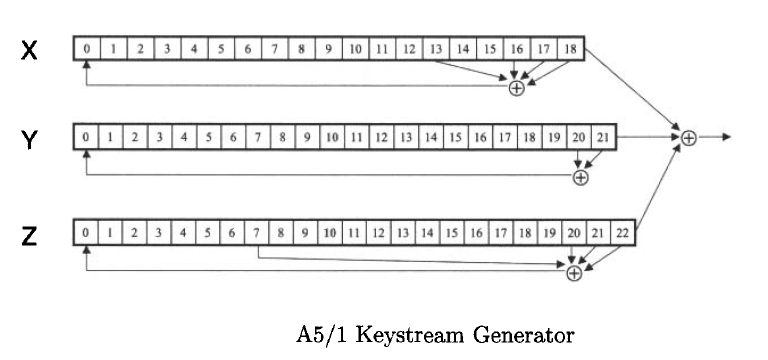
and

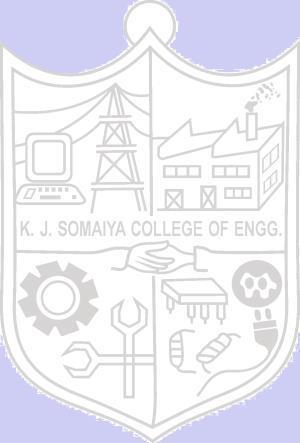


respectively.

Given three bits x, y, and z, define ma,](x,y, z) to be the majority vote function, that is, if the majority of x, y, and z are 0, the function returns 0; otherwise it returns 1. Since there are an odd number of bits, there cannot be a tie, so this function is well defined.

The wiring diagram for the A5/1 algorithm is illustrated below:





**Procedure / Approach /Algorithm / Activity Diagram:**

1. **Key Stream generation Algorithm:**

At each step: m = maj(x8, y10, z10)

-Examples: maj(0,1,0) = 0 and maj(1,1,0) = 1

If x8 = m then X steps

-t = x13 ⊕ x16 ⊕ x17 ⊕ x18

-xi = xi−1 for i = 18,17,…,1 and x0 = t

If y10 = m then Y steps

-t = y20 ⊕ y21

-yi = yi−1 for i = 21,20,…,1 and y0 = t

If z10 = m then Z steps

-t = z7 ⊕ z20 ⊕ z21 ⊕ z22

-zi = zi−1 for i = 22,21,…,1 and z0 = t

**Keystream bit is *x*18 ⊕ *y*21 ⊕ *z*22**

**Implementation:**

Implement the A5/1 algorithm. Encryption and decryption function should ask for key and a input and show the output to the user.

**Results:** (Program with output as per the format)

**Code:**

x = input("Enter the value of register X: ")

y = input("Enter the value of register Y: ")

z = input("Enter the value of register Z: ")

print("Initially ")

print("Register X: ", x)

print("Register Y: ", y)

print("Register Z: ", z)

print("\n")

keystream = []

for i in range(32):

    c = int(x[8]) + int(y[10]) + int(z[10])

    if c <= 1:

        m = 0

    else:

        m = 1

    if int(x[8]) == m:

        tx = int(x[13]) ^ int(x[16]) ^ int(x[17]) ^ int(x[18])

        x1 = x[0:18]

        x = str(tx) + x1

    if int(y[10]) == m:

        ty = int(y[20]) ^ int(y[21])

        y1 = y[0:21]

        y = str(ty) + y1

    if int(z[10]) == m:

        tz = int(z[7]) ^ int(z[20]) ^ int(z[21]) ^ int(z[22])

        z1 = z[0:22]

        z = str(tz) + z1

    print("X: ", x)

    print("Y: ", y)

    print("Z: ", z)

    k = int(x[18]) ^ int(y[21]) ^ int(z[22])

    print("For iteration ", i + 1, ", keystream bit: ", k)

    print("\n")

    keystream.append(k)

print("After all iterations: ")

print("Register X: ", x)

print("Register Y: ", y)

print("Register Z: ", z)

print("Keystream: ", keystream)

pt = input("Enter the plaintext: ")

ctr = 0

key = ''

for i in keystream:

    key = key + str(i)

    ctr = ctr + 1

    if ctr == len(pt):

        break

ct = ''

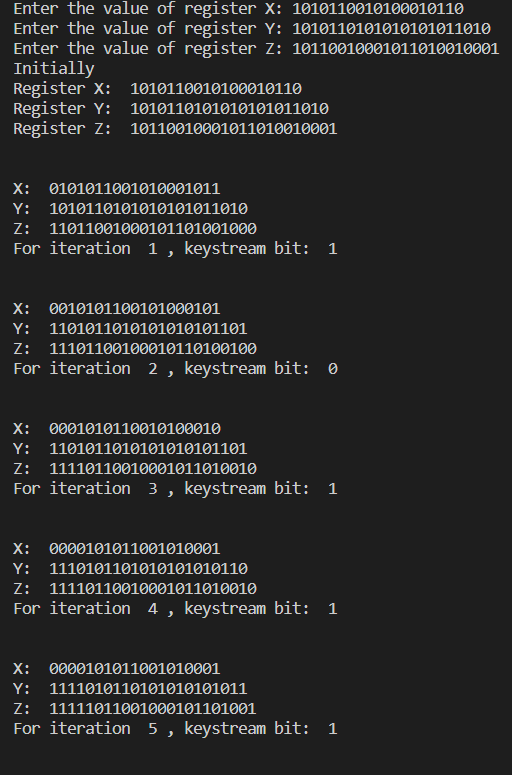
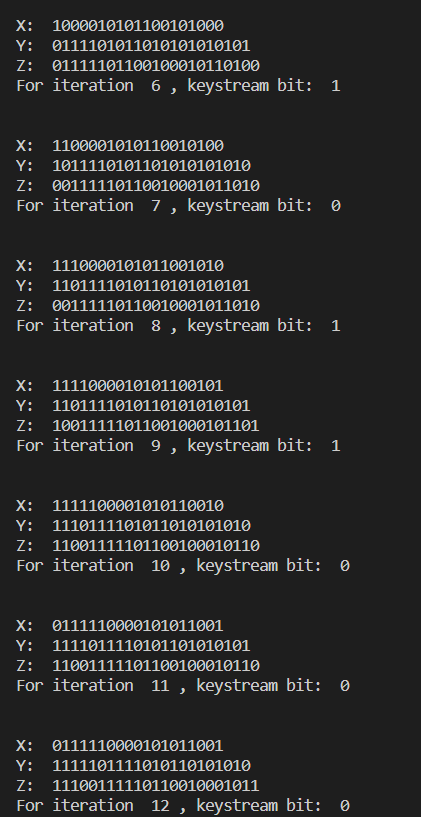
for i, j in zip(key, pt):

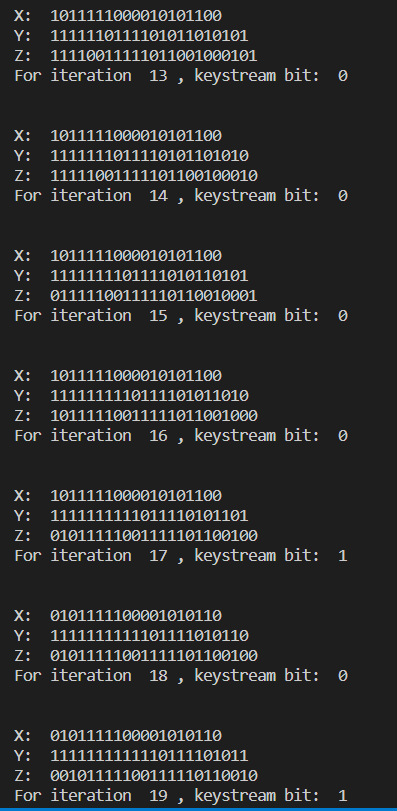
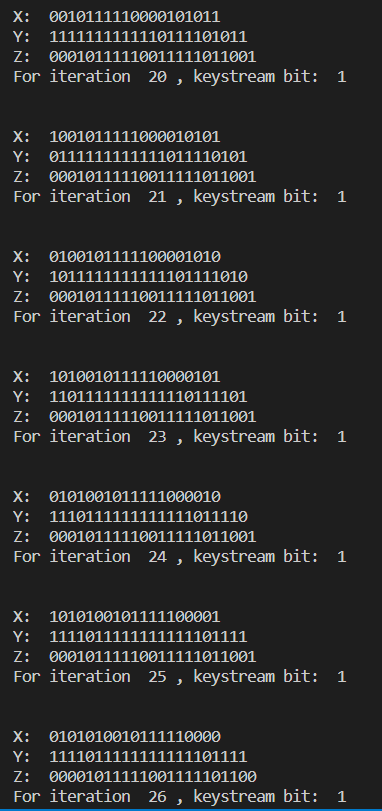
    x = int(i) ^ int(j)

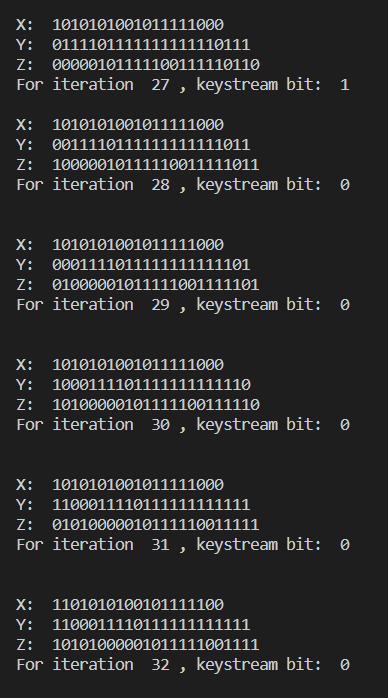
    ct = ct + str(x)

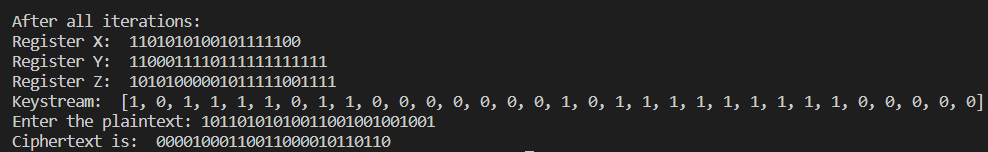
print("Ciphertext is: ", ct)

**Output:**

**** ****





**Questions:**

**1) List the stream cipher used in current date along with the name of applications in which those are used.**

**Ans:-**

**RC4 -** RC4, which stands for Rivest Cipher 4, is the most widely used of all stream ciphers, particularly in software. It's also known as ARCFOUR or ARC4. RC4 steam chiphers have

been used in various protocols like WEP and WPA (both security protocols for wireless networks) as well as in TLS. Unfortunately, recent studies have revealed vulnerabilities in RC4, prompting Mozilla and Microsoft to recommend that it be disabled where possible. In fact, RFC 7465 prohibits the use of RC4 in all versions of TLS.

These recent findings will surely allow other stream ciphers (e.g. SALSA, SOSEMANUK, PANAMA, and many others, which already exist but never gained the same popularity as RC4) to emerge and possibly take its place.

RC4 is used in various applications such as WEP from 1997 and WPA from 2003. We also find applications of RC4 in SSL from 1995 and it is a successor of TLS from 1999. RC4 is used in varied applications because of its simplicity, speed, and simplified implementation in both software and hardware.

**There are various types of RC4 such as Spritz, RC4A, VMPC, and RC4A.**

1. SPRITZ: Spritz can be used to build a cryptographic hash function, a deterministic random bit generator (DRBG), n an encryption algorithm that supports authenticated encryption with associated data (AEAD).
2. RC4A: Souraduyti Paul and Bart Preneel have proposed an RC4 variant, which they call RC4A, which is stronger than RC4.
3. VMPC: VMPC is another variant of RC4 which stands for Variably Modified Permutation Composition.
4. RC4A+: RC4A+ is a modified version of RC4 with a more complex three-phase key schedule which takes about three times as long as RC4 and a more complex output function which performs four additional lookups in the S array for each byte output, taking approximately 1.7 times as long as basic RC4.

# Outcomes:

**CO2:** Illustrate different cryptographic algorithms for security

**Conclusion: (Conclusion to be based on the objectives and outcomes achieved)**

In This experiment, we learnt to successfully implement and execute stream cipher A5/1.

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of faculty in-charge with date**

**References: Books/ Journals/ Websites:**

1. Mark Stamp, “Information Security Principles and Practice”, Wiley.
2. Behrouz A. Forouzan, “Cryptography and Network Security”, Tata McGraw Hill
3. William Stalling, “Cryptography and Network Security”, Prentice Hall