In [1]:

```
def nibblesubs(N):
    n=int(len(N)/2)
    left=N[:n]
    right=N[n:]
    l=""
    r=""

    for i in range(n):
        l=l+str(left[i])
        r=r+str(right[i])

    sbox=dict()
    sbox['0000']='1001'
    sbox['0001']='0100'
    sbox['0010']='1010'
    sbox['0011']='1011'
    sbox['0100']='1101'
    sbox['0101']='0001'
    sbox['0110']='1000'
    sbox['0111']='0101'
    sbox['1000']='0110'
    sbox['1001']='0010'
    sbox['1010']='0000'
    sbox['1011']='0011'
    sbox['1100']='1100'
    sbox['1101']='1110'
    sbox['1110']='1111'
    sbox['1111']='0111'

    s=sbox[l]+sbox[r]

    output=list()

    for i in s:
        output.append(int(i))

    return output
```

In [2]:

```
def inversenibblesubs(N):
    n=int(len(N)/2)
    left=N[:n]
    right=N[n:]
    l=""
    r=""

    for i in range(n):
        l=l+str(left[i])
        r=r+str(right[i])

    sbox=dict()
    sbox['0000']='1001'
    sbox['0001']='0100'
    sbox['0010']='1010'
    sbox['0011']='1011'
    sbox['0100']='1101'
    sbox['0101']='0001'
    sbox['0110']='1000'
    sbox['0111']='0101'
    sbox['1000']='0110'
    sbox['1001']='0010'
    sbox['1010']='0000'
    sbox['1011']='0011'
    sbox['1100']='1100'
    sbox['1101']='1110'
```

```
    sbox['1110']='1111'
    sbox['1111']='0111'

    decryptionsbox=dict()

    for k,v in sbox.items():
        decryptionsbox[v]=k

    s=decryptionsbox[l]+decryptionsbox[r]

    output=list()

    for i in s:
        output.append(int(i))

    return output
```

In [3]:

```
def shiftrow(N):
    N0=N[:4]
    N1=N[4:8]
    N2=N[8:12]
    N3=N[12:16]

    return N0+N3+N2+N1
```

In [4]:

```
def mixcolumns(N):
    N0=N[:4]
    N1=N[4:8]
    N2=N[8:12]
    N3=N[12:16]

    b=list()
    c=list()

    for i in range(4):
        b.append(N0[i])
        c.append(N2[i])

    for i in range(4):
        b.append(N1[i])
        c.append(N3[i])

    row1=[b[0]^b[6],b[1]^b[4]^b[7],b[2]^b[4]^b[5],b[3]^b[5],c[0]^c[6],c[1]^c[4]^c[7],c[
2]^c[4]^c[5],c[3]^c[5]]
    row2=[b[2]^b[4],b[0]^b[3]^b[5],b[0]^b[1]^b[6],b[1]^b[7],c[2]^c[4],c[0]^c[3]^c[5],c[
0]^c[1]^c[6],c[1]^c[7]]

    N0=row1[:4]
    N1=row2[:4]
    N2=row1[4:]
    N3=row2[4:]

    return N0+N1+N2+N3
```

In [5]:

```
def rotatenibble(N):
    n=int(len(N)/2)
    left=N[:n]
    right=N[n:]

    return right+left
```

In [6]:

```
def exor(a,b):
    out=list()
```

```
    for i in range(len(a)):
        out.append(a[i]^b[i])

    return out
```

In [7]:

```
def keyschedule(k):
  #converting string to list for easy calculations
    key=list()
    for i in k:
        key.append(int(i))

    w=list()
    w.append(key[:8])
    w.append(key[8:])
    w.append(exor(exor(w[0],[1,0,0,0,0,0,0,0]),nibblesubs(rotatenibble(w[1]))))
    w.append(exor(w[2],w[1]))
    w.append(exor(exor(w[2],[0,0,1,1,0,0,0,0]),nibblesubs(rotatenibble(w[3]))))
    w.append(exor(w[4],w[3]))

    K0=w[0]+w[1]
    K1=w[2]+w[3]
    K2=w[4]+w[5]

    return K0,K1,K2
```

In [8]:

```
def encryption(K0,K1,K2,text):
    t=list()
    for i in text:
        t.append(int(i))

    #Round 0
    round0=exor(t,K0)

    #Round 1"
    nbsub1=nibblesubs(round0[:8])
    nbsub2=nibblesubs(round0[8:])
    nbsub=nbsub1+nbsub2
    sr=shiftrow(nbsub)
    mc=mixcolumns(sr)
    round1=exor(mc,K1)

    #Round 2
    finalnbsub1=nibblesubs(round1[:8])
    finalnbsub2=nibblesubs(round1[8:])
    finalnbsub=finalnbsub1+finalnbsub2
    finalsr=shiftrow(finalnbsub)

    ciphertext=exor(finalsr,K2)

    return ciphertext
```

In [10]:

```
def getString(l):
    s=""
    for i in l:
        s=s+str(i)

    return s
```

In [13]:

```
k='0100101011110101'
plaintext='1101010110101010'

Key0,Key1,Key2=keyschedule(k)
```

```
cipher=encryption(Key0,Key1,Key2,plaintext)
ciphertext=getString(cipher)

print("16 bit Key:",k)
print("16 bit Plaintext:",plaintext)
print("Ciphertext:",ciphertext)
```

```
16 bit Key: 0100101011110101
16 bit Plaintext: 1101010110101010
Ciphertext: 0001010001100101
```

In [ ]:

```
```