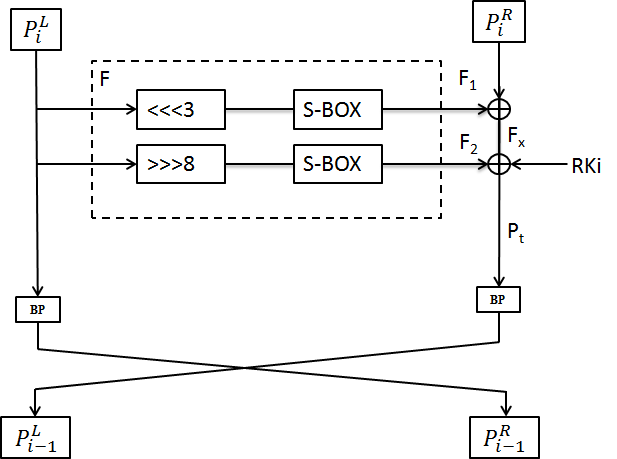
Assignment:

ANU Design : Write a C program as well as Keil program for the following design. Use the data given below.

ANU has 25 rounds cipher :-

32 – Bit MSB 32 – Bit LSB



Notations:-

* PT 64-bit input plaintext block
* CT 64-bit output cipher text block
* RKi 128-bit Round sub key for round i
* F 1 & F2 Function
* ⊕ Bitwise exclusive-OR operation
* <<<n Left cyclic shift by n bits
* >>>n Right cyclic shift by n bits
* RCiRound counter i
* **||** Concatenation of two strings
* ! Bitwise NOT operation
* BP Bit Permutation
* LCS Left circular shift
* RCS Right circular shift

S – Box :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| x | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| S(x) | 2 | 9 | 7 | E | 1 | C | A | 0 | 4 | 3 | 8 | D | F | 6 | 5 | B |

P – Layer:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 |
| **BP[** | 20 | 16 | 28 | 24 | 17 | 21 | 25 | 29 |
|  | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 15 |
| **BP[** | 22 | 18 | 30 | 26 | 19 | 23 | 27 | 31 |
|  | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 |
| **BP[** | 11 | 15 | 03 | 07 | 14 | 10 | 06 | 02 |
|  | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| **BP[** | 09 | 13 | 01 | 05 | 12 | 08 | 04 | 00 |

Key scheduling:

1. KEY <<< 13.

2. [K3 K2 K1 K0] ←S [K3 K2 K1 K0]

3. [K7 K6 K5 K4] ←S [K7 K6 K5 K4]

4. [K63 K62 K61 K60 K59] ← [K63 K62 K61 K60 K59] ⊕ RCi

**Code :**

#include<stdio.h>

unsigned long int shiftleft(unsigned long int,unsigned char);

unsigned long int sbox\_(unsigned long int);

unsigned long int shiftright(unsigned long int,unsigned int );

unsigned long int player(unsigned long long int);

unsigned char sbox[16] = {2,9,7,0xE,1,0xC,0xA,0,4,3,8,0xD,0xF,0x6,0x5,0xB};

unsigned char p[32] = {20,16,28,24,17,21,25,29,22,18,30,26,19,23,27,31,11,15,3,7,14,10,6,2,9,13,1,5,12,8,4,0};

int main()

{

unsigned int m;

unsigned long int msb\_32bit=0x00,lsb\_32bit=0x00,s\_box1,s\_box2,bn,bn1,sub\_key[1],c,r,i1,j1,i,j[2];

unsigned long long int temp,k[2]={0x00,0x00};

for(r=0;r<25;r++)

{

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

c = shiftleft(msb\_32bit,3);

s\_box1 = sbox\_(c);

c = shiftright(msb\_32bit,8);

s\_box2 = sbox\_(c);

bn = player(msb\_32bit);

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

sub\_key[0] = (k[1]&0XFFFFFFFF);

temp=k[1];

k[1]= (temp<<13)|(k[0]>>(64-13));

k[0]= (k[0]<<13)|(temp>>(64-13));

//k[1] = K\_UP1;

//k[0] = K\_UP2;

for(i=0; i<2; i++)

{

j[i]=sbox[(k[1]>>(i\*4))&0XF];

}

k[1] &= (0Xffffffffffffff00);

for(i=0;i<2;i++)

{

k[1] |= ((j[i]&0xf)<<(i\*4));

}

temp = (((k[1]&0XF800000000000000)>>(64-5))^(0x1f&r));

k[1] &= 0X07FFFFFFFFFFFFFF;

k[1] |= temp<<(64-5);

temp = (s\_box1^s\_box2^lsb\_32bit^sub\_key[0]);

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

// bn1=0;

bn1 = player(temp);

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

msb\_32bit = bn1;

lsb\_32bit = bn;

printf("\n Cipher Text= %lx %lx\n",msb\_32bit,lsb\_32bit);

printf("\n---------------------------Round %d---------------------\n",r);

}

}

unsigned long int shiftleft(unsigned long int msb\_32bit,unsigned char i)

{

unsigned long int a,b,tempr;

a = msb\_32bit<<i;

tempr = msb\_32bit>>(32-i);

b = a|tempr;

return(b);

}

unsigned long int sbox\_(unsigned long int c)

{

unsigned long int a=0,i,j[33],b=0;

for (i=0;i<32;i=i+4)

{

j[b]= sbox[((c>>(28-i)&0xf))];

a |= ((((j[b])&0xf)<<(28-i)));

b++;

}

return(a);

}

unsigned long int shiftright(unsigned long int msb\_32bit,unsigned int i)

{

unsigned long int a,b,tempr;

a = msb\_32bit>>i;

tempr = msb\_32bit<<(32-i);

b = a|tempr;

return(b);

}

unsigned long int player(unsigned long long int tempr)

{

unsigned int i, b=0;

for (i=0; i<32;i++)

{

b |= ((tempr>>i)&0x1)<<p[i];

}

return(b);

}

Output:

