**LICI Cipher** **Encryption and Decryption Keil code**

#include<LPC21xx.h>

unsigned int sbox1(unsigned int);

unsigned int sbox2(unsigned int);

void tx(unsigned int c); //function that transmit char (user see on uart terminal)

void tx\_string(const unsigned char \*str);

char rx(void); // function that receive anything

unsigned char s\_e[16] = {3,0xf,0xe,1,0,0xa,5,8,0xc,4,0xb,2,9,7,6,0xd};//Encryption S-Box

unsigned char s\_d[16] = {4,3,0xb,0,9,6,0xe,0xd,7,0xc,5,0xa,8,0xf,2,1};//Decryption S-Box

int main()

{

unsigned long long int k[2]={0x00,0x00},temp,p[31]={0x00};

unsigned int data[2]={0x0,0x0},j1,i1,m,r,j2,i2,i,j;

PINSEL0 = 0x05; // configuration for tx and rx in pinsel

U0LCR = 0x83; // configure the line control register

U0DLL = 0x61;

U0DLM = 0x00;// config baud rate 9600 (calculate in bps ---> convert into hex --> then load into DLL & DLM)

U0LCR = 0x03; // lock baud rate by resetting baud rate bit in U0LCR register

tx\_string("\n\n-------------Encryption System--------------\n");

tx\_string("Plain Text= ");

//-----------------------Plain text entry and display-----------------------------

for (j1=0;j1<2;j1++)

{

{

for (i1=0;i1<=7;i1++)

{

m = rx();

tx(m);

if (m>0x2f && m<0x3a)

{

temp = m-0x30;

}

if(m>=97 && m<=102)

{

temp = m-87;

}

data[j1] += (temp & 0xf) << ((7-i1)\*4);

}

}

}

//-----------------------Key entry and display-------------------------------------------------

tx\_string("\nKey= ");

for (j1=0; j1<2; j1++)

{

//k[j1] =0;

for (i1=0;i1<=15;i1++)

{

m = rx();

tx(m);

if (m>0x2F && m<0x3a)

{

temp = m-0x30;

}

if(m>=97 && m<=102)

{

temp = m-87;

}

k[j1] += (temp & 0xf) << ((15-i1)\*4);

}

}

//---------------------------key scheduling-------------------------------------------------------

for(r=0; r<31;r++)

{

temp=k[1];

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

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

//Sbox

m=0;

for(i1=0; i1<2; i1++)

{

m|=(s\_e[(k[1]>>(i1\*4))&0XF]<<i1\*4)&0xff;

}

k[1] &= (0Xffffffffffffff00);

k[1] |= ((m&0xFF));

//Round Counter xor

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

k[1] &= 0X07FFFFFFFFFFFFFF;

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

p[r]=k[1];

}

//--------------------------------- Encryption Logic -----------------------------------------------

for(j1=0;j1<31;j1++)

{

data[0]=sbox1(data[0]);

data[1]=data[1] ^ data[0] ^ (p[j1] & 0xffffffff);

data[1]=(((data[1]<<3) | (data[1]>>(32-3))) & 0xffffffff);

data[0]=data[1] ^ data[0] ^ ((p[j1] & 0xffffffff00000000)>>32);

data[0]=(((data[0]>>7) | (data[0]<<(32-7))) & 0xffffffff);

temp=data[1];

data[1]=data[0];

data[0]=temp;

}

//--------------------------------- Display Cypher Text----------------------------------------------

tx\_string("\nCypher Text= ");

for(j1=2; j1>0; j1--)

{

for (i1=0; i1<=7; i1++)

{

temp=(data[j1-1]>>(7-i1)\*4)&0xf;

if(temp<10)

tx(0x30+temp);

else

tx(87+temp);

}

}

//--------------------------------- Decryption Logic -----------------------------------------------

tx\_string("\n\n------------------Decryption System---------------------\n");

for(j1=0;j1<31;j1++)

{

temp=data[0];

data[0]=data[1];

data[1]=temp;

data[0]=(((data[0]<<7) | (data[0]>>(32-7))) & 0xffffffff);

data[0]=data[0] ^ data[1] ^ ((p[30-j1] & 0xffffffff00000000)>>32);

data[1]=(((data[1]>>3) | (data[1]<<(32-3))) & 0xffffffff);

data[1]=data[0] ^ data[1] ^ (p[30-j1] & 0xffffffff);

data[0]=sbox2(data[0]);

}

//--------------------------------- Decrypted Plain Text----------------------------------------------

tx\_string("Decrypted Plain Text= ");

for(j1=1; j1<3; j1++)

{

for (i1=0; i1<=7; i1++)

{

temp=(data[j1-1]>>(7-i1)\*4)&0xf;

if(temp<10)

tx(0x30+temp);

else

tx(87+temp);

}

}

//----------------------------------------------------------------------------------------------------

}// main end

unsigned int sbox1(unsigned int c)

{

unsigned int i,z=0;

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

{

z |= s\_e[((c>>(28-i))&0xf)]<<(28-i);

}

return (z);

}

unsigned int sbox2(unsigned int c)

{

unsigned int i,z=0;

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

{

z |= s\_d[((c>>(28-i))&0xf)]<<(28-i);

}

return (z);

}

void tx\_string(const unsigned char \*str) //A function to send a string on UART0

{

while(1)

{

if( \*str == '\0' ) break;

tx(\*str++);

}

}

void tx(unsigned int c)

{

while(!(U0LSR & 0x20)); // check the transmitter data available bit in U0LSR

U0THR = c; // if data is available it will transmit by jumping from above line (while !)

}

char rx(void)

{

unsigned char m;

while(!(U0LSR & 0x01)); // check the transmitter data available bit in U0LSR

m = U0RBR; // if data is available it will transmit by jumping from above line (while !)

return(m);

}