```c
#include "prototypes.h"



/* Encrypt a single block with 10 rounds */
void AesEncrypt(unsigned char *blk, unsigned char *key, int Nr)
{
    printf("Round 0:\n");
    printf("-----Start:  ");
    for (int i = 0; i < 16; i++)
        printf("%02x ", blk[i]);

    printf("\n");
    printf("----Output:  ");
    AddRoundKey(blk, key, 0);

    for (int i = 0; i < 16; i++)
        printf("%02x ", blk[i]);

    for (int x = 1; x <= (Nr - 1); x++)
    {
        SubBytes(blk);
        ShiftRows(blk);
        MixColumns(blk);
        AddRoundKey(blk, key, x);
        printf("\nRound %d:\n", x);
        printf("----Output:  ");
        for (int i = 0; i < 16; i++)
            printf("%02x ", blk[i]);

    }

    printf("\nRound 10:\n");
    SubBytes(blk);
    ShiftRows(blk);
    AddRoundKey(blk, key, Nr);
    printf("----Output:  ");
    for (int i = 0; i < 16; i++)
        printf("%02x ", blk[i]);
}






/* The AES Substitution Table */
static const unsigned char sbox[256] = {
    0x63, 0x7c, 0x77, 0x7b, 0xf2, 0x6b, 0x6f, 0xc5, 0x30, 0x01, 0x67, 0x2b, 0xfe, 0xd7, 0xab, 0x76,
    0xCA, 0x82, 0xC9, 0x7D, 0xFA, 0x59, 0x47, 0xF0, 0xAD, 0xD4, 0xA2, 0xAF, 0x9C, 0xA4, 0x72, 0xC0,
    0xB7, 0xFD, 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC, 0x34, 0xA5, 0xE5, 0xF1, 0x71, 0xD8, 0x31, 0x15,
    0x04, 0xC7, 0x23, 0xC3, 0x18, 0x96, 0x05, 0x9A, 0x07, 0x12, 0x80, 0xE2, 0xEB, 0x27, 0xB2, 0x75,
    0x09, 0x83, 0x2C, 0x1A, 0x1B, 0x6E, 0x5A, 0xA0, 0x52, 0x3B, 0xD6, 0xB3, 0x29, 0xE3, 0x2F, 0x84,
    0x53, 0xD1, 0x00, 0xED, 0x20, 0xFC, 0xB1, 0x5B, 0x6A, 0xCB, 0xBE, 0x39, 0x4A, 0x4C, 0x58, 0xCF,
    0xD0, 0xEF, 0xAA, 0xFB, 0x43, 0x4D, 0x33, 0x85, 0x45, 0xF9, 0x02, 0x7F, 0x50, 0x3C, 0x9F, 0xA8,
    0x51, 0xA3, 0x40, 0x8F, 0x92, 0x9D, 0x38, 0xF5, 0xBC, 0xB6, 0xDA, 0x21, 0x10, 0xFF, 0xF3, 0xD2,
    0xCD, 0x0C, 0x13, 0xEC, 0x5F, 0x97, 0x44, 0x17, 0xC4, 0xA7, 0x7E, 0x3D, 0x64, 0x5D, 0x19, 0x73,
    0x60, 0x81, 0x4F, 0xDC, 0x22, 0x2A, 0x90, 0x88, 0x46, 0xEE, 0xB8, 0x14, 0xDE, 0x5E, 0x0B, 0xDB,
    0xE0, 0x32, 0x3A, 0x0A, 0x49, 0x06, 0x24, 0x5C, 0xC2, 0xD3, 0xAC, 0x62, 0x91, 0x95, 0xE4, 0x79,
    0xE7, 0xC8, 0x37, 0x6D, 0x8D, 0xD5, 0x4E, 0xA9, 0x6C, 0x56, 0xF4, 0xEA, 0x65, 0x7A, 0xAE, 0x08,
    0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4, 0xC6, 0xE8, 0xDD, 0x74, 0x1F, 0x4B, 0xBD, 0x8B, 0x8A,
    0x70, 0x3E, 0xB5, 0x66, 0x48, 0x03, 0xF6, 0x0E, 0x61, 0x35, 0x57, 0xB9, 0x86, 0xC1, 0x1D, 0x9E,
    0xE1, 0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94, 0x9B, 0x1E, 0x87, 0xE9, 0xCE, 0x55, 0x28, 0xDF,
    0x8c, 0xa1, 0x89, 0x0d, 0xbf, 0xe6, 0x42, 0x68, 0x41, 0x99, 0x2d, 0x0f, 0xb0, 0x54, 0xbb, 0x16 };


static const unsigned char modifiedsbox[256] = {
    0x63, 0x7c, 0x77, 0x7b, 0xf2, 0x6b, 0x6f, 0xc5, 0x30, 0x01, 0x67, 0x2b, 0xfe, 0xd7, 0xab, 0x76,    //  0
    0xCA, 0x82, 0xC9, 0x7D, 0xFA, 0x59, 0x47, 0xF0, 0xAD, 0xD4, 0xA2, 0xAF, 0x9C, 0xA4, 0x72, 0xC0,    //  1
    0xB7, 0xFD, 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC, 0x34, 0xA5, 0xE5, 0xF1, 0x71, 0xD8, 0x31, 0x15,    //  2
    0x51, 0xA3, 0x40, 0x8F, 0x92, 0x9D, 0x38, 0xF5, 0xBC, 0xB6, 0xDA, 0x21, 0x10, 0xFF, 0xF3, 0xD2,    //  3 ------------------ swap with 7
    0x09, 0x83, 0x2C, 0x1A, 0x1B, 0x6E, 0x5A, 0xA0, 0x52, 0x3B, 0xD6, 0xB3, 0x29, 0xE3, 0x2F, 0x84,    //  4
    0x53, 0xD1, 0x00, 0xED, 0x20, 0xFC, 0xB1, 0x5B, 0x6A, 0xCB, 0xBE, 0x39, 0x4A, 0x4C, 0x58, 0xCF,    //  5
    0xD0, 0xEF, 0xAA, 0xFB, 0x43, 0x4D, 0x33, 0x85, 0x45, 0xF9, 0x02, 0x7F, 0x50, 0x3C, 0x9F, 0xA8,    //  6
    0x04, 0xC7, 0x23, 0xC3, 0x18, 0x96, 0x05, 0x9A, 0x07, 0x12, 0x80, 0xE2, 0xEB, 0x27, 0xB2, 0x75,    //  7 ------------------ swap with 3
    0xCD, 0x0C, 0x13, 0xEC, 0x5F, 0x97, 0x44, 0x17, 0xC4, 0xA7, 0x7E, 0x3D, 0x64, 0x5D, 0x19, 0x73,    //  8
    0x60, 0x81, 0x4F, 0xDC, 0x22, 0x2A, 0x90, 0x88, 0x46, 0xEE, 0xB8, 0x14, 0xDE, 0x5E, 0x0B, 0xDB,    //  9
    0xE0, 0x32, 0x3A, 0x0A, 0x49, 0x06, 0x24, 0x5C, 0xC2, 0xD3, 0xAC, 0x62, 0x91,
        0x95, 0xE4, 0x79,    // 10
    0xE7, 0xC8, 0x37, 0x6D, 0x8D, 0xD5, 0x4E, 0xA9, 0x6C, 0x56, 0xF4, 0xEA, 0x65, 
        0x7A, 0xAE, 0x08,    // 11
    0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4, 0xC6, 0xE8, 0xDD, 0x74, 0x1F, 0x4B, 
        0xBD, 0x8B, 0x8A,    // 12
    0x70, 0x3E, 0xB5, 0x66, 0x48, 0x03, 0xF6, 0x0E, 0x61, 0x35, 0x57, 0xB9, 0x86, 
        0xC1, 0x1D, 0x9E,    // 13
    0xE1, 0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94, 0x9B, 0x1E, 0x87, 0xE9, 0xCE, 
        0x55, 0x28, 0xDF,    // 14
    0x8c, 0xa1, 0x89, 0x0d, 0xbf, 0xe6, 0x42, 0x68, 0x41, 0x99, 0x2d, 0x0f, 0xb0, 
        0x54, 0xbb, 0x16 };  // 15


/* The key schedule rcon table */
static const unsigned char Rcon[10] =
{ 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80, 0x1B, 0x36 };

/* The *x function */
static unsigned char xtime(unsigned char x)
{
    if (x & 0x80) { return ((x << 1) ^ 0x1B) & 0xFF; }
    return x << 1;
}



/* MixColumns: Processes the entire block */
static void MixColumns(unsigned char *col)
{
    unsigned char tmp[4], xt[4];

    for (int x = 0; x < 4; x++, col += 4)
    {
        xt[0]  = xtime(col[0]);
        xt[1]  = xtime(col[1]);
        xt[2]  = xtime(col[2]);
        xt[3]  = xtime(col[3]);
        tmp[0] = xt[0]  ^ xt[1]  ^ col[1] ^ col[2] ^ col[3];
        tmp[1] = col[0] ^ xt[1]  ^ xt[2]  ^ col[2] ^ col[3];
        tmp[2] = col[0] ^ col[1] ^ xt[2]  ^ xt[3]  ^ col[3];
        tmp[3] = xt[0]  ^ col[0] ^ col[1] ^ col[2] ^ xt[3];
        col[0] = tmp[0];
        col[1] = tmp[1];
        col[2] = tmp[2];
        col[3] = tmp[3];
    }
}




/* ShiftRows: Shifts the entire block */
```

```c
static void ShiftRows(unsigned char *col)
{
    unsigned char t;

    /* 2nd row */
    t       = col[1];
    col[1]  = col[5];
    col[5]  = col[9];
    col[9]  = col[13];
    col[13] = t;

    /* 3rd row */
    t       = col[2];
    col[2]  = col[10];
    col[10] = t;
    t       = col[6];
    col[6]  = col[14];
    col[14] = t;

    /* 4th row */
    t         = col[15];
    col[15]   = col[11];
    col[11]   = col[7];
    col[7]    = col[3];
    col[3]    = t;
}



/* SubBytes */
static void SubBytes(unsigned char *col)
{
#if DEMO
    for (int x = 0; x < 16; x++)
        col[x] = sbox[col[x]];
#else
    for (int x = 0; x < 16; x++)
        col[x] = modifiedsbox[col[x]];
#endif // DEMO
}



/* AddRoundKey */
static void AddRoundKey(unsigned char *col, unsigned char *key, int round)
{
    for (int x = 0; x < 16; x++)
        col[x] ^= key[(round << 4) + x];
}
```