```
1   /*
2    * Block Cipher - AES voi MDS 4x4 theo bang nhan tren truong
3    * thohd 2018
4    */
5
6   #include <stdio.h>
7   #include <stdlib.h>
8   #include <string.h>
9   #include <conio.h>
10  #include <io.h>
11  #include <dos.h>
12  #include <time.h>
13  #include <iostream>
14  #include <iomanip>
15  //#include "FMuls.h"
16  using namespace std;
17
18  int Nk=8;
19  int Nr=14;
20  unsigned char *key;
21  unsigned char invSBox[16*16];
22  unsigned char state[4*4]; /* 128 bit */
23  unsigned char w[4 * (14 + 1) * 4]; /* max */
24
25  unsigned char Sbox[16*16]={  // populate the Sbox matrix
26  /*       0     1     2     3     4     5     6     7     8     9     a     b     c
    d     e     f */
27  /*0*/  0x63, 0x7c, 0x77, 0x7b, 0xf2, 0x6b, 0x6f, 0xc5, 0x30, 0x01, 0x67, 0x2b, 0xfe,
    0xd7, 0xab, 0x76,
28  /*1*/  0xca, 0x82, 0xc9, 0x7d, 0xfa, 0x59, 0x47, 0xf0, 0xad, 0xd4, 0xa2, 0xaf, 0x9c,
    0xa4, 0x72, 0xc0,
29  /*2*/  0xb7, 0xfd, 0x93, 0x26, 0x36, 0x3f, 0xf7, 0xcc, 0x34, 0xa5, 0xe5, 0xf1, 0x71,
    0xd8, 0x31, 0x15,
30  /*3*/  0x04, 0xc7, 0x23, 0xc3, 0x18, 0x96, 0x05, 0x9a, 0x07, 0x12, 0x80, 0xe2, 0xeb,
    0x27, 0xb2, 0x75,
31  /*4*/  0x09, 0x83, 0x2c, 0x1a, 0x1b, 0x6e, 0x5a, 0xa0, 0x52, 0x3b, 0xd6, 0xb3, 0x29,
    0xe3, 0x2f, 0x84,
32  /*5*/  0x53, 0xd1, 0x00, 0xed, 0x20, 0xfc, 0xb1, 0x5b, 0x6a, 0xcb, 0xbe, 0x39, 0x4a,
    0x4c, 0x58, 0xcf,
33  /*6*/  0xd0, 0xef, 0xaa, 0xfb, 0x43, 0x4d, 0x33, 0x85, 0x45, 0xf9, 0x02, 0x7f, 0x50,
    0x3c, 0x9f, 0xa8,
34  /*7*/  0x51, 0xa3, 0x40, 0x8f, 0x92, 0x9d, 0x38, 0xf5, 0xbc, 0xb6, 0xda, 0x21, 0x10,
    0xff, 0xf3, 0xd2,
35  /*8*/  0xcd, 0x0c, 0x13, 0xec, 0x5f, 0x97, 0x44, 0x17, 0xc4, 0xa7, 0x7e, 0x3d, 0x64,
    0x5d, 0x19, 0x73,
36  /*9*/  0x60, 0x81, 0x4f, 0xdc, 0x22, 0x2a, 0x90, 0x88, 0x46, 0xee, 0xb8, 0x14, 0xde,
    0x5e, 0x0b, 0xdb,
37  /*a*/  0xe0, 0x32, 0x3a, 0x0a, 0x49, 0x06, 0x24, 0x5c, 0xc2, 0xd3, 0xac, 0x62, 0x91,
    0x95, 0xe4, 0x79,
38  /*b*/  0xe7, 0xc8, 0x37, 0x6d, 0x8d, 0xd5, 0x4e, 0xa9, 0x6c, 0x56, 0xf4, 0xea, 0x65,
    0x7a, 0xae, 0x08,
39  /*c*/  0xba, 0x78, 0x25, 0x2e, 0x1c, 0xa6, 0xb4, 0xc6, 0xe8, 0xdd, 0x74, 0x1f, 0x4b,
    0xbd, 0x8b, 0x8a,
40  /*d*/  0x70, 0x3e, 0xb5, 0x66, 0x48, 0x03, 0xf6, 0x0e, 0x61, 0x35, 0x57, 0xb9, 0x86,
    0xc1, 0x1d, 0x9e,
41  /*e*/  0xe1, 0xf8, 0x98, 0x11, 0x69, 0xd9, 0x8e, 0x94, 0x9b, 0x1e, 0x87, 0xe9, 0xce,
    0x55, 0x28, 0xdf,
42  /*f*/  0x8c, 0xa1, 0x89, 0x0d, 0xbf, 0xe6, 0x42, 0x68, 0x41, 0x99, 0x2d, 0x0f, 0xb0,
    0x54, 0xbb, 0x16 };
43
44
45  unsigned char Rcon[11*4] = {
46          0x00, 0x00, 0x00, 0x00,
47          0x01, 0x00, 0x00, 0x00,
48          0x02, 0x00, 0x00, 0x00,
49          0x04, 0x00, 0x00, 0x00,
50          0x08, 0x00, 0x00, 0x00,
51          0x10, 0x00, 0x00, 0x00,
52          0x20, 0x00, 0x00, 0x00,
53          0x40, 0x00, 0x00, 0x00,
54          0x80, 0x00, 0x00, 0x00,
55          0x1b, 0x00, 0x00, 0x00,
56          0x36, 0x00, 0x00, 0x00};
57
58  // Phep nhan tren truong GF(2^8)
59  /* Thuc hien phep nhan 0x02 * a */
60
61  unsigned char gfMultBy02(unsigned char i)  {
62      if(i<0x80)
63          i=i<<1;
64      else{
65          i=i<<1;
66          i=(unsigned char)((int)i^(int)0x1b);
67  //          i=(unsigned char)((int)i^(int)0x1b);
```

```
68  //        i=i<<1;
69      }
70      return i;
71  }
72
73  /* Thuc hien phep nhan 0x03 * a */
74  unsigned char gfMultBy03(unsigned char i)    {
75      return (unsigned char)((int)gfMultBy02(i)^(int)i);
76  }
77
78  /* Thuc hien phep nhan
79   * 0x09 * a = (a * 0x02 * 0x02 * 0x02 ) + (a * 0x01)
80   * - dung khi giai ma
81   */
82  unsigned char gfMultBy09(unsigned char i) {
83      return (unsigned char)( (int)gfMultBy02(gfMultBy02(gfMultBy02(i)))^(int)i );
84  }
85
86  /* Thuc hien phep nhan 0x0b * a
87   * 0x0b * a = (a * 0x02 * 0x02 * 0x02 ) + (a * 0x02) + (a * 0x01)
88   * - dung khi giai ma
89   */
90  unsigned char gfMultBy0b(unsigned char i)      {
91      return (unsigned char)(
    (int)gfMultBy02(gfMultBy02(gfMultBy02(i)))^(int)gfMultBy02(i)^(int)i );
92  }
93
94  /* Thuc hien phep nhan 0x0d * a
95   * 0x0d * a = (a * 0x02 * 0x02 * 0x02 ) + (a * 0x02 * 0x02 ) + (a * 0x01)
96   - dung khi giai ma*/
97  unsigned char gfMultBy0d(unsigned char i)      {
98      return (unsigned char)( (int)gfMultBy02(gfMultBy02(gfMultBy02(i))) ^
99              (int)gfMultBy02(gfMultBy02(i)) ^ (int)(i) );
100 }
101
102 /* Thuc hien phep nhan 0x0e * a
103  * 0x0e * a = (a * 0x02 * 0x02 * 0x02 ) + (a * 0x02 * 0x02 ) + (a * 0x02)
104  - dung khi giai ma*/
105 unsigned char gfMultBy0e(unsigned char i)   {
106     return (unsigned char)( (int)gfMultBy02(gfMultBy02(gfMultBy02(i))) ^
107              (int)gfMultBy02(gfMultBy02(i)) ^ (int)gfMultBy02(i) );
108 }
109 /* numkey = 0, 1, 2 */
110 void SetNkNr(int numkey) {
111
112    if ((numkey == 0)||(numkey == 128)){
113       Nk = 4;
114       Nr = 10;
115    }
116    else if ((numkey == 1)||(numkey == 192)){
117       Nk = 6;
118       Nr = 12;
119    }
120    else if ((numkey == 2)||(numkey == 256)){
121       Nk = 8;
122       Nr = 14;
123    }
124
125    /* DEBUG */
126    printf("\n Nk = %d ", Nk);
127    printf("\n Nr = %d ", Nr);
128 }
129
130
131 /* Tao hop the nguoc voi Sbox - dung khi giai ma */
132 void InitInvSBox(void) {
133     FILE *f;
134
135     for(int i=0;i<16;i++)
136         for(int j=0;j<16;j++) {
137         unsigned char t;
138         int x,y;
139         t=Sbox[i*16+j];
140         x=((unsigned char)((int)(t)>>4));
141         y=((unsigned char)((int)(t)&0x0f));
142         invSBox[x*16+y]=(unsigned char)((i<<4)+j);
143     }
144
145     f=fopen("invSbox.txt", "wb");
146     fprintf(f,"\n Hop the hoan vi nguoc: ");
147     fprintf(f,"\n invSBox[16*16] = {\n");
148
149     for(int i=0;i<16;i++) {
150         for(int j=0;j<16;j++) {
```

```
151                 fprintf(f,"%02X, ", invSBox[i*16+j]);
152             }
153             fprintf(f, "\n");
154         }
155         fprintf(f, "};");
156 
157         fclose(f);
158     }
159 
160     // Dich vong trai di 1 Byte
161     unsigned long RotWord(unsigned long t)
162     {
163         unsigned char *p;
164         unsigned char temp;
165 
166         p=(unsigned char *)&t;
167         temp=*p;
168         *p=*(p+1);
169         p++;
170         *p=*(p+1);
171         p++;
172         *p=*(p+1);
173         p++;
174         *p=temp;
175 
176         return t;
177     }
178 
179     // Dich trai cac dong (hang)
180     void ShiftRows(void) {
181         unsigned char temp[4*4];
182         for(int i=1;i<4;i++)
183             for(int j=0;j<4;j++) {
184                 temp[i*4+j]=state[i*4+j];
185             }
186 
187         /* Vi dong 0 khong dich vong. Dong thu i dich di i byte */
188         for(int i=1;i<4;i++)
189             for(int j=0;j<4;j++)  {
190                 state[i*4+j]=temp[i*4+((i+j)%4)];
191             }
192     }
193 
194     // Thay the mang 4*4 State thanh mang 4*4 qua Sbox.
195     void SubBytes(void) {
196         int x,y;
197         for(int i=0;i<4;i++)        /* cot */
198             for(int j=0;j<4;j++)    /* hang */ {
199                 /* a[i, j] = Sbox[a[i,j]] */
200     //            x= (unsigned char)((int)(state[i*4+j])>>4);
201     //            y= (unsigned char)((int)(state[i*4+j])& 0x0f);
202                 state[i*4+j]=Sbox[state[i*4+j]];
203             }
204     }
205 
206     // Thay the qua Sbox. Thay the 4 byte. Dung trong luoc do tao khoa.
207     unsigned long SubWord(unsigned long t)
208     {
209         unsigned char *p;
210         int x,y;
211 
212         p=(unsigned char *)&t;
213         x= (unsigned char)((int)(*p)>>4);
214         y= (unsigned char)((int)(*p)&0x0f);
215         *p=Sbox[x*16+y];
216 
217         p++;
218         x= (unsigned char)((int)(*p)>>4);
219         y= (unsigned char)((int)(*p)&0x0f);
220         *p=Sbox[x*16+y];
221 
222         p++;
223         x= (unsigned char)((int)(*p)>>4);
224         y= (unsigned char)((int)(*p)&0x0f);
225         *p=Sbox[x*16+y];
226 
227         p++;
228         x= (unsigned char)((int)(*p)>>4);
229         y= (unsigned char)((int)(*p)&0x0f);
230         *p=Sbox[x*16+y];
231         return t;
232     }
233 
234     /* Expantion userkey to Key Schedule */
```

```
235  /*
236     Mo rong tu 128 -> 4*(10+1)*4 = 1408 bit khoa con - 10 vong.
237     Mo rong tu 192 -> 4*(12+1)*4 = 1664 bit khoa con - 12 vong.
238     Mo rong tu 256 -> 4*(14+1)*4 = 2000 bit khoa con - 14 vong.
239
240     Vi moi vong su dung Nb = 4*4 byte khoa. Can them 1 khoa con cho
241     vong cuoi cung.
242   */
243  void KeyExpantion(unsigned char *key, unsigned char *w) {
244      /* Nk*4 byte dau duoc lay tu khoa vao -> duoc Nk dong */
245      for(int i=0;i<Nk;i++)   { /* theo dong */
246          w[i*4]=key[i*4];
247          w[i*4+1]=key[i*4+1];
248          w[i*4+2]=key[i*4+2];
249          w[i*4+3]=key[i*4+3];
250      }
251
252      /* Tu dong thu Nk den 4*(Nr+1)*4 */
253      for(int row=Nk;row<(4*(Nr+1));row++)  {
254          unsigned long temp=*((unsigned long*)&w[(row-1)*4]);/* W[i-1] = w[4*(i-1)]*/
255
256          if(row%Nk==0){
257              temp=SubWord(RotWord(temp))^(*((unsigned long*)&Rcon[(row/Nk)*11]));
258          }
259          else if(Nk > 6 && (row % Nk == 4)) {
260              temp=SubWord(temp);
261          }
262          temp=temp^(*((unsigned long*)(&w[(row-Nk)*4])));
263
264          /* Ghi lai 4 byte nay vao w ung voi dong khoa thu row*/
265          *((unsigned long*)&w[row*4])=temp;
266      }
267  }
268
269
270  /* Cong State voi khoa vong thu round */
271  void AddRoundKey(int round)
272  {
273      for(int i=0;i<4;i++)         /* cot */
274          for(int j=0;j<4;j++)     /* hang */ {
275              state[i*4+j]=(unsigned char)((int)state[i*4+j]^(int)w[(round*4+j)*4+i]);
276          }
277  }
278
279  /* MixColumns */
280  void MixColumns(void){
281      unsigned char temp[4*4];
282
283      /* Copy du lieu tu State vao temp */
284      for(int i=0;i<4;i++)
285          for(int j=0;j<4;j++) {
286              temp[i*4+j]=state[i*4+j];
287          }
288
289  //Ma tran goc === cot 1 va 2
290      for(int i=0;i<4;i++)  {
291          state[0*4+i]=(unsigned char)((int)gfMultBy02(temp[0*4+i])^
292                                       (int)gfMultBy03(temp[1*4+i])^
293                                       (int)temp[2*4+i]^
294                                       (int)temp[3*4+i]);
295
296          state[1*4+i]=(unsigned char)((int)temp[0*4+i]^
297                                       (int)gfMultBy02(temp[1*4+i])^
298                                       (int)gfMultBy03(temp[2*4+i])^
299                                       (int)temp[3*4+i]);
300
301          state[2*4+i]=(unsigned char)((int)temp[0*4+i]^
302                                       (int)temp[1*4+i]^
303                                       (int)gfMultBy02(temp[2*4+i])^
304                                       (int)gfMultBy03(temp[3*4+i]));
305
306          state[3*4+i]=(unsigned char)((int)gfMultBy03(temp[0*4+i])^
307                                       (int)temp[1*4+i]^
308                                       (int)temp[2*4+i]^
309                                       (int)gfMultBy02(temp[3*4+i]));
310      }
311  }
312
313
314  void AES_Cipher(unsigned char *input, unsigned char *output) {
315      /* 0. Neu khong co du lieu vao -> THOAT */
316      if((input==NULL)||(output==NULL))
317          return;
318
```

```
319      /* 1. Dua input vao State */
320      for(int i=0;i<16;i++)  { /* theo cot truoc */
321          state[(i%4)*4 +(i/4)]=input[i];
322      }
323
324      /* 2. AddRoundKey - vong thu 0*/
325      // printf("\n round [0]. input  ");
326      //for (int i=0; i<16; i++) printf("%02X", state[(i%4)*4 + (i/4)]);
327      AddRoundKey(0);
328      //printf("\n round [0]. k_sch  ");
329      //for (int i=0; i<16; i++) printf("%02X", w[i]);
330
331
332      /* 3. Thuc hien Nr vong lap */
333      for(int round = 1; round <= (Nr - 1); ++round)
334      {
335        //  printf("\n round [%d]. start ", round);
336          //for (int i=0; i<16; i++) printf("%02X", state[(i%4)*4 + (i/4)]);
337
338          SubBytes();
339
340          //printf("\n round [%d]. s_box ", round);
341          //for (int i=0; i<16; i++) printf("%02X", state[(i%4)*4 + (i/4)]);
342
343          ShiftRows();
344
345          //printf("\n round [%d]. s_row ", round);
346          //for (int i=0; i<16; i++) printf("%02X", state[(i%4)*4 + (i/4)]);
347
348          MixColumns();
349
350          //printf("\n round [%d]. m_col ", round);
351          //for (int i=0; i<16; i++) printf("%02X", state[(i%4)*4 + (i/4)]);
352
353          AddRoundKey(round);
354
355          //printf("\n round [%d]. k_sch ", round);
356          //for (int i=0; i<16; i++) printf("%02X", w[(round)*16+i]);
357      }
358
359      /* 4. Final Round */
360      SubBytes();
361      ShiftRows();
362      AddRoundKey(Nr);
363
364      /* Lay du lieu ra theo cot*/
365      for(int i=0;i<16;i++)  {
366          output[i]=state[(i%4)*4+(i/4)];
367      }
368  }
369
370  /* Cac ham phuc vu qua trinh giai ma */
371  /* Dich vong sang phai 1 byte */
372  void InvShiftRows(void) {
373      unsigned char temp[4*4];
374
375      for(int i=1;i<4;i++)
376          for(int j=0;j<4;j++) {
377              temp[i*4+j]=state[i*4+j];
378          }
379
380       /* lay nguoc cua phep bien doi o ShiftRows o tren */
381       for(int i=1;i<4;i++)
382           for(int j=0;j<4;j++) {
383               state[i*4+((i+j)%4)]=temp[i*4+j];
384           }
385  }
386
387  /* Thuc hien phep the nguoc - thong qua bang the nguoc invSBox */
388  void InvSubBytes(void) {
389      int x,y;
390      for(int i=0;i<4;i++)
391          for(int j=0;j<4;j++)  {
392  //            x= (unsigned char)((int)(state[i*4+j])>>4);
393  //            y= (unsigned char)((int)(state[i*4+j])&0x0f);
394              state[i*4+j]=invSBox[state[i*4+j]];
395          }
396  }
397
398  void InvMixColumns(void) {
399      unsigned char temp[4*4];
400
401      for(int i=0;i<4;i++)
402          for(int j=0;j<4;j++) {
```

```
403             temp[i*4+j]=state[i*4+j];
404         }
405 //ma tran goc
406     for(int i=0;i<4;i++)  {
407 
408 
409         state[0*4+i]=(unsigned char)((int)gfMultBy0e(temp[0*4+i])^
410                                      (int)gfMultBy0b(temp[1*4+i])^
411                                      (int)gfMultBy0d(temp[2*4+i])^
412                                      (int)gfMultBy09(temp[3*4+i]));
413 
414 
415         state[1*4+i]=(unsigned char)((int)gfMultBy09(temp[0*4+i]) ^
416                                      (int)gfMultBy0e(temp[1*4+i]) ^
417                                      (int)gfMultBy0b(temp[2*4+i]) ^
418                                      (int)gfMultBy0d(temp[3*4+i]) );
419 
420 
421         state[2*4+i]=(unsigned char)((int)gfMultBy0d(temp[0*4+i]) ^
422                                      (int)gfMultBy09(temp[1*4+i]) ^
423                                      (int)gfMultBy0e(temp[2*4+i]) ^
424                                      (int)gfMultBy0b(temp[3*4+i]) );
425 
426 
427         state[3*4+i]=(unsigned char)((int)gfMultBy0b(temp[0*4+i]) ^
428                                      (int)gfMultBy0d(temp[1*4+i]) ^
429                                      (int)gfMultBy09(temp[2*4+i]) ^
430                                      (int)gfMultBy0e(temp[3*4+i]) );
431     }
432 }
433 
434 // Ham thuc hien giai ma AES
435 void InvAES_Cipher(unsigned char *input, unsigned char *output) {
436     /* Neu input hoac output chua khoi tao -> THOAT */
437     if(input==NULL||output==NULL)
438         return;
439 
440     /* 1. Dua input vao State */
441     for(int i=0;i<16;i++) { /* theo cot */
442         state[(i%4)*4+(i/4)]=input[i];
443     }
444 
445     /* 2. AddRoundKey - Khoa cua vong thu Nr*/
446     AddRoundKey(Nr);
447 
448     /* 3. Thuc hien Nr vong lap nguoc */
449     for(int round = Nr-1; round >= 1;--round)  {
450         InvShiftRows();
451         InvSubBytes();
452         AddRoundKey(round);
453         InvMixColumns();
454     }
455 
456     /* 4. Final Round */
457     InvShiftRows();
458     InvSubBytes();
459     AddRoundKey(0);
460 
461     /* Lay du lieu ra - theo cot*/
462     for(int i=0;i<16;i++)  {
463         output[i]=state[(i%4)*4 + (i/4)];
464     }
465 }
466 
467 int main() {
468    unsigned char pBlock[100][16],cBlock[100][16],bro[16],bdich[16],bma[16];
469    unsigned char khoa[32]; /* 256 bit */
470     unsigned char testvectors2[16]={ 0xdc,0x95,0xc0,0x78,0xa2,0x40,0x89,0x89,
471                                0xad,0x48,0xa2,0x14,0x92,0x84,0x20,0x87};
472     int k;
473    printf("\n Chuong trinh mo phong ma dich AES ");
474 //
475 
476     /* ************************************/
477     /*       Mo phong qua trinh ma AES       */
478     /* ************************************/
479 
480     /* Ban ro vao */
481 
482      // for (int i=0; i<16; i++) bro[i] = testvectors2[i];
483 
484     /* Khoa ma/dich - 128 bit */
485     for (int i = 0; i < 16; i++) khoa[i] = (unsigned char)i;
486 
```

```
487      /* In ban ro */
488      printf("\n Ban ro    : ");
489      for (int i = 0; i < 16; i++) printf("%02X", bro[i]);
490      printf("\n Khoa      : ");
491      for (int i = 0; i < 16; i++) printf("%02X", khoa[i]);
492
493      /* 1. Thiet lap so khoa su dung */
494      SetNkNr(128);
495      /* 2. Thiet lap khoa con */
496      KeyExpantion(khoa, w);
497      /* 3. Goi ham ma */
498      printf("\nStarting Encrypt...");
499
500       AES_Cipher(bro, bma);
501
502      /* In ket qua ma */
503      printf("\n Ban ma    : ");
504      for (int i = 0; i < 16; i++) printf("%02X", bma[i]);
505
506      /* ************************************/
507      /*       Mo phong qua trinh dich cua AES       */
508      /* ************************************/
509
510      /* 1. Tao bang hoan vi nguoc */
511      InitInvSBox();
512
513      /* 2. Thiet lap khoa con */
514      KeyExpantion(khoa, w);
515      /* 3. Goi ham giai ma*/
516      InvAES_Cipher(bma, bdich);
517
518      /* In ket qua dich */
519      printf("\n Ban dich : ");
520      for (int i = 0; i < 16; i++) printf("%02X", bdich[i]);
521       printf("\n An phim bat ky de thoat chuong trinh : ");
522        getch();
523      return 0;
524
525  }
526
527
528
```