/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

File name: SM2\_ENC.h

Version: SM2\_ENC\_V1.1

Date: Sep 27,2016

Description: implementation of SM2 encryption algorithm and decryption algorithm Function List:

1.SM2\_init //initiate SM2 curve

2.SM2\_ENC //SM2 encryption, calls SM3\_KDF

3.SM2\_DEC //SM2 decryption, calls

SM2\_KDF,Test\_null,Test\_Point,SM3\_init,SM3\_process,SM3\_done

4.SM2\_ENC\_SelfTest //test whether the calculation is correct by comparing

the result with the standard data

5.Test\_Point //test if the given point is on SM2 curve

6.Test\_Pubkey //test if the given public key is valid

7.Test\_Null //test if the geiven array is all zero

8.SM2\_KeyGeneration //calculate a pubKey out of a given priKey

9.SM3\_init //init SM3 state

10.SM3\_process //compress the the message

11.SM3\_done //compress the rest message and output the hash value

12.SM3\_KDF //key deviding function base on SM3, generates key

stream Notes:

This SM2 implementation source code can be used for academic, non-profit making or non-commercial use only.

This SM2 implementation is created on MIRACL. SM2 implementation source code provider does not provide MIRACL library, MIRACL license or any permission to use MIRACL library. Any commercial use of MIRACL requires a license which may be obtained from Shamus Software Ltd.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

[#define ECC\_WORDSIZE 8](#bookmark1)

[#define SM2\_NUMBITS 256](#bookmark2)

[#define SM2\_NUMWORD (SM2\_NUMBITS/ECC\_WORDSIZE) //32](#bookmark3)

#define ERR\_INFINITY\_POINT 0x00000001

#define ERR\_NOT\_VALID\_ELEMENT 0x00000002

#define ERR\_NOT\_VALID\_POINT 0x00000003

#define ERR\_ORDER 0x00000004

#define ERR\_ARRAY\_NULL 0x00000005

#define ERR\_C3\_MATCH 0x00000006

#define ERR\_ECURVE\_INIT 0x00000007

#define ERR\_SELFTEST\_KG 0x00000008

#define ERR\_SELFTEST\_ENC 0x00000009

#define ERR\_SELFTEST\_DEC 0x0000000A

unsigned char SM2\_p[32] =

{0xFF,0xFF,0xFF,0xFE,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF}; unsigned char SM2\_a[32] =

{0xFF,0xFF,0xFF,0xFE,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFC}; unsigned char SM2\_b[32] =

{0x28,0xE9,0xFA,0x9E,0x9D,0x9F,0x5E,0x34,0x4D,0x5A,0x9E,0x4B,0xCF,0x65,0x09,0xA7,

0xF3,0x97,0x89,0xF5,0x15,0xAB,0x8F,0x92,0xDD,0xBC,0xBD,0x41,0x4D,0x94,0x0E,0x93}; unsigned char SM2\_n[32] =

{0xFF,0xFF,0xFF,0xFE,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,

0x72,0x03,0xDF,0x6B,0x21,0xC6,0x05,0x2B,0x53,0xBB,0xF4,0x09,0x39,0xD5,0x41,0x23}; unsigned char SM2\_Gx[32]=

{0x32,0xC4,0xAE,0x2C,0x1F,0x19,0x81,0x19,0x5F,0x99,0x04,0x46,0x6A,0x39,0xC9,0x94,

0x8F,0xE3,0x0B,0xBF,0xF2,0x66,0x0B,0xE1,0x71,0x5A,0x45,0x89,0x33,0x4C,0x74,0xC7}; unsigned char SM2\_Gy[32]=

{0xBC,0x37,0x36,0xA2,0xF4,0xF6,0x77,0x9C,0x59,0xBD,0xCE,0xE3,0x6B,0x69,0x21,0x53,

0xD0,0xA9,0x87,0x7C,0xC6,0x2A,0x47,0x40,0x02,0xDF,0x32,0xE5,0x21,0x39,0xF0,0xA0}; unsigned char SM2\_h[32]=

{0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01};

big para\_p,para\_a,para\_b,para\_n,para\_Gx,para\_Gy,para\_h; epoint \*G;

miracl \*mip;

int Test\_Point(epoint\* point);

int Test\_PubKey(epoint \*pubKey);

int Test\_Null(unsigned char array[],int len); int SM2\_Init();

int SM2\_KeyGeneration(big priKey,epoint \*pubKey);

int SM2\_Encrypt(unsigned char\* randK,epoint \*pubKey,unsigned char M[],int klen,unsigned char C[]);

int SM2\_Decrypt(big dB,unsigned char C[],int Clen,unsigned char M[]);

int SM2\_ENC\_SelfTest();