```
/**
 *  HyRAL(Hybrid Randomize Algorithm)
 *  HyRAL Algorithm 2009/11/27 Version
 *
 *  Created by Inaba on 2009/11/27.
 *  Copyright 2009 Laurel Intelligent Systems Co.,Ltd. All rights reserved.
 *
 *  Compile Option
 *        __BIG_ENDIAN__          It operates with the big endian.
 *        __LITTLE_ENDIAN__       It operates with the little endian.
 *        __STANDARD_COMP__       The macro-function is not used. function "f,F,G,rF,r
G"
 *        __TEST_VECTOR__         Monitor.  Option of "__STANDARD_COMP__" is neces
sary.
 */
/**
 * @file       HyRAL16.h
 * @brief      HyRAL Encrypt / Decrypt<br>
 * @author     LIS(Laurel Intelligent Systems Co.,Ltd.):Inaba
 * @date       2009/11/27     New making
 * @date       2009/12/22     For endian "revised a f-function table and Label and f-f
unction"
 * @date       2009/12/25     Speed improvement "abolishes loop processing", and it i
s added compile option "__STANDARD_COMP__"
 * @date       2010/01/07     Logic addition for test vector. Compile option "__TEST_V
ECTOR__"
 * @version    0.04     2010/01/07 LIS:Inaba
 */

#if !defined(_HYRAL16_H_)
    #define _HYRAL16_H_

//-------------------------------------------------------------------------
---------//
// Compile Option Define
//-------------------------------------------------------------------------
---------//
//#define __TEST_VECTOR__
//#define __STANDARD_COMP__

#if defined(__TEST_VECTOR__)
    #define __STANDARD_COMP__
#endif  //__TEST_VECTOR__

//-------------------------------------------------------------------------
---------//
// Endian judgment
//-------------------------------------------------------------------------
---------//
#if !defined(__LITTLE_ENDIAN__) & !defined(__BIG_ENDIAN__)
    //#include "endian.h"
    #if __BYTE_ORDER == __LITTLE_ENDIAN
        #define __LITTLE_ENDIAN__
    #elif __BYTE_ORDER == __BIG_ENDIAN
```

```
        #define __BIG_ENDIAN__
    #endif
#endif

//------------------------------------------------------------------------
---------//
// OS judgment
//------------------------------------------------------------------------
---------//
#if defined(__APPLE__) || defined(MACOSX)
    // MacOS
#elif defined(WIN32) || defined(WIN64)
    // WindowsOS
#else
    // OthersOS
#endif

//------------------------------------------------------------------------
---------//
// definition
//------------------------------------------------------------------------
---------//
#define HYRAL_OK                0x00000000;
#define HYRAL_BLOCK_LEN_ERR     0x00000001;
#define HYRAL_OTHERS_ERR        0xFFFFFFFF;

//------------------------------------------------------------------------
---------//
// Function definition
//------------------------------------------------------------------------
---------//
void init();
void ClearKM();
void ClearIK();
void ClearRK();

//------------------------------------------------------------------------
---------//
// Test function
//------------------------------------------------------------------------
---------//
#if defined(__STANDARD_COMP__)
    // The KM data is returned.
    void getKM(unsigned long* p_km1,unsigned long* p_km2,
        unsigned long* p_km3,unsigned long* p_km4);
    // The IK data is returned.
    void getIK(unsigned long* p_ik1,unsigned long* p_ik2,unsigned long* p_ik3,
        unsigned long* p_ik4,unsigned long* p_ik5,unsigned long* p_ik6);
    // The RK data is returned.
    void getRK(unsigned long* p_rk1,unsigned long* p_rk2,unsigned long* p_rk3,
        unsigned long* p_rk4,unsigned long* p_rk5,unsigned long* p_rk6,
        unsigned long* p_rk7,unsigned long* p_rk8,unsigned long* p_rk9);
#endif  //__STANDARD_COMP__
//------------------------------------------------------------------------
```

```
---------//
// Encrypt & Decrypt commonness function
//-------------------------------------------------------------------------
---------//
// f-function[32bit]
#if defined(__STANDARD_COMP__)
    unsigned long f1(const unsigned long Im);
    unsigned long f2(const unsigned long Im);
    unsigned long f3(const unsigned long Im);
    unsigned long f4(const unsigned long Im);
    unsigned long f5(const unsigned long Im);
    unsigned long f6(const unsigned long Im);
    unsigned long f7(const unsigned long Im);
    unsigned long f8(const unsigned long Im);
#endif  //__STANDARD_COMP__

// KeyGeneration function
void KeyGenerationSingle(unsigned long* OK);
void KeyGenerationDouble(unsigned long* OK1,unsigned long* OK2);

// KeyScheduling function
void KeySchedulingSingle();
void KeySchedulingDouble();

//-------------------------------------------------------------------------
---------//
// Encrypt function
//-------------------------------------------------------------------------
---------//
// G-function[128bit] / Encrypt
#if defined(__STANDARD_COMP__)
    void G1(unsigned long* XtoY);
    void G2(unsigned long* XtoY);
#endif  //__STANDARD_COMP__

// F-function[128bit] / Encrypt
#if defined(__STANDARD_COMP__)
    void F1(unsigned long* XtoY, unsigned long* IKi);
    void F2(unsigned long* XtoY, unsigned long* IKi);
#endif  //__STANDARD_COMP__

// HyRAL Encrypt
void HyralSingle(unsigned long* PtoC);
void HyralDouble(unsigned long* PtoC);

//-------------------------------------------------------------------------
---------//
// Decrypt function
//-------------------------------------------------------------------------
---------//
// G-function[128bit] / Decrypt
#if defined(__STANDARD_COMP__)
    void rG1(unsigned long* YtoX);
    void rG2(unsigned long* YtoX);
```

```
#endif   //__STANDARD_COMP__

// F-function[128bit] / Decrypt
#if defined(__STANDARD_COMP__)
     void rF1(unsigned long* YtoX, unsigned long* IKi);
     void rF2(unsigned long* YtoX, unsigned long* IKi);
#endif   //__STANDARD_COMP__

// HyRAL Decrypt
void rHyralSingle(unsigned long* CtoP);
void rHyralDouble(unsigned long* CtoP);

#endif   //_HYRAL16_H_
```