TB041

KEELOQ® Decryption Routines in C

Author: Lucio Di Jasio
Microchip Technology Inc.

OVERVIEW

This Technical Brief describes the implementation of the fundamental Decryption Algorithm used in most KEELOQ applications in the C programming language.

Three implementations are presented. In the first one, the code is written in the most generic form and is suitable for implementation on high end processors and desktop computers using standard C compilers (e.g. Microsoft® or Borland®). In the other two implementations, the code is optimized for speed of execution on 8-bit PICmicro® microcontroller and C compilers (e.g. HITECH PIC C and CCS PIC C).

INTRODUCTION

In KEELOQ Technology, the Decryption Algorithm is not really part of the original specification and could be in fact anything (e.g. DES, Rijndael) if it is considered solid enough for the application and conforms to a basic set of rules that goes more or less like this:

1. Block cypher
2. Key configured
3. Uses a symmetric key (the same key is used during encryption and decryption)

The algorithm discussed in this Technical Brief is referring to the most popular algorithm implemented in the first set of HCS2XX and HCS3XX Encoders and HCS5XX Decoders (and Application Notes) released from Microchip. For simplicity from now on, we will refer to it as the Decryption Algorithm,

ALGORITHM OVERVIEW

The KEELOQ Decription Algorithm was originally designed for optimal performance on 8-bit PICmicro microcontrollers (MCU) that run at 4 MHz only. The design specifications dictated a 64-bit key length and as in every good modern encryption algorithm, the security of the application had to depend uniquely on the secrecy of the key. In other words, if we assume the "enemy" knows the algorithm but not the secret key, the system must still be secure.

The KEELOQ Decryption Algorithm was designed to operate on a 32-bit message (block) and to be fully configurable by a 64-bit decription key. Its working comprised some 528 encryption stages (loops).

At the beginning, a 32-bit shift register is loaded with the message to be decrypted. Than in each stage (of 528) 5 bits from the shift register are entered in a Non Linear Function (NLF), combined with two more bits from the same shift register (XORed) and one bit from the key to produce a new bit that is then rotated into the shift register. Finally, the 64-bit key itself is rotated so that every bit in the key gets used at least 8 times.

At the end of the process, the decrypted message (plain text) is available in the 32-bit shift register itself (see Figure 1).

STANDARD C IMPLEMENTATION

The first implementation is aimed at personal computers with vast resources of RAM and program memory, and targets mainstream compilers (like Borland and Microsoft). Therefore, the source (available in appendix A), is not optimized. Optimization is left to the compilers. To hold and manipulate the shift register and the key (split in two), 32-bit integers are used. Bit manipulation is performed "defining" three simple functions (inline) based on shifting and spacing bit logic operators for maximum portability. All in all, the simplicity of the implementation is evident while the performance will be very dependent on the compiler, target processor type and clock speed.

PICmicro C IMPLEMENTATION

The second and third listings (appendix B and C) refer to versions of the same Decryption Algorithm specifically optimized for the popular HITECH and CCS C compilers for the PICmicro family of 8-bit microcontrollers.

The code has been optimized for space using the smallest number possible of RAM locations, and has been further optimized for execution speed considering the typical constraints of an embedded control design.

Eight bit variables and arithmetic have been used to better exploit the potential of the 8-bit RISC architecture of the PICmicro MCU. Since the standard C language syntax would not allow the expression of the rotation with a sufficient level of efficiency for small 8-bit microcontrollers, and to streamline the Non Linear Function computation, different optimization techniques had to be used for the two compilers.

The CCS implementation makes use of compiler specific library functions like `SHIFT_LEFT()`, `SWAP()` and `BIT_TEST()`, while the HITECH implementation requires the use of single line assembly inserts `asm()`, or multiple lines inserts `#asm .. #endasm`.

Further, in order to optimize the 64-bit key manipulation (the key is actually stored in an array of 8 x 8-bit integers), the main loop has been broken in two nested loops so that the key is accessed and rotated one byte at a time. The added advantage of this method is that at a end of the computation, the key is still aligned and ready for the next use, while in a rotation scheme like the generic C code, the key has to be reloaded or rotated 16 more times for realignment.

Considering the small amount of code, there is almost no difference in the efficiency of the two compilers in terms of code space.

In terms of speed efficiency, both versions get more or less to the same level achieving some 25 ms for a full decryption on a generic mid-range PICmicro MCU running at 4 MHz. This compares quite nicely with the speed of comparable assembly language implementations with a speed difference in the range of a 30%.

REFERENCES

An Introduction to KEELOQ Code Hopping	TB003
Secure Learning RKE Systems Using KEELOQ Encoders	TB001
KEELOQ Decryption & IFF Algorithms	TB030
Interfacing a KEELOQ Encoder to a PLL Circuit	TB042
KEELOQ CRC Verification Routines	TB043

MEMORY USAGE

Memory usage is dependent on C compiler used.

KEYWORDS

KEELOQ, Decoder, Decryption, C Language

FIGURE 1: THE KEELOQ DECRYPTION ALGORITHM

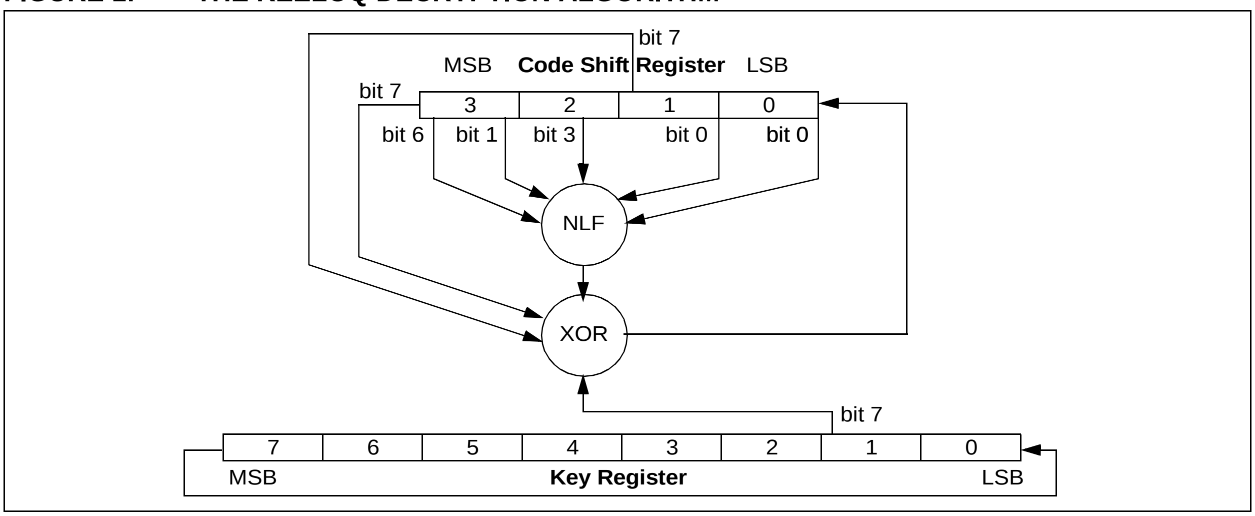

TABLE 1: NON-LINEAR FUNCTION OUTPUT

I4	I3	I2	I1	I0	NLF	I4	I3	I2	I1	I0	NLF
0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	1	1	1	0	0	0	1	0
0	0	0	1	0	1	1	0	0	1	0	1
0	0	0	1	1	1	1	0	0	1	1	1
0	0	1	0	0	0	1	0	1	0	0	1
0	0	1	0	1	1	1	0	1	0	1	0
0	0	1	1	0	0	1	0	1	1	0	1
0	0	1	1	1	0	1	0	1	1	1	0
0	1	0	0	0	0	1	1	0	0	0	0
0	1	0	0	1	0	1	1	0	0	1	1
0	1	0	1	0	1	1	1	0	1	0	0
0	1	0	1	1	0	1	1	0	1	1	1
0	1	1	0	0	1	1	1	1	0	0	1
0	1	1	0	1	1	1	1	1	0	1	1
0	1	1	1	0	1	1	1	1	1	0	0
0	1	1	1	1	0	1	1	1	1	1	0

APPENDIX A: GENERIC C SOURCE CODE

```
//
//   Module GENC.C
//
//   Keeloq Decryption Algorithm
//      generic implementation in C language
//
//
//  INPUTS:
//      mpik, mpin      64 bit decryption key (preloaded)
//      csr             32 bit shift register (preloaded with text to decrypt)
//  OUTPUT:
//      csr             32 bit plain text   (decrypted message)
//
#include <stdio.h>

typedef unsigned int  word;
typedef unsigned long dword;

#define setbit( b, n)   ((b) |= ( 1 << (n)))
#define getbit( b, n)   (((b) & (1L<<n)) ? 1 : 0)
#define ifbit(x,y)      if (getbit(x,y))


dword   mpik,mpin;  // decryption key
dword   csr;        // shift register

dword decode(dword csr)

{
  dword lut[32] =
  { 0,1,1,1, 0,1,0,0, 0,0,1,0, 1,1,1,0, 0,0,1,1, 1,0,1,0, 0,1,0,1, 1,1,0,0 };
 /*   E        2        4        7        C        5        A        3   */

  dword pik,pin,bitin,keybit,keybit2;
  word bitlu;
  int ix;


  // Load Key

  pik = mpik;
  pin = mpin;


  for(ix=0; ix < 528; ix++)
  {
     // Rotate Code Shift Register
```

```
    bitin = getbit(csr,31);
    csr<<=1;

    // Get Key Bit

    keybit2=getbit(pin,15);

    // Rotate Key Right

    keybit=getbit(pik,31);
    pik=(pik<<1)|getbit(pin,31);
    pin=(pin<<1)|keybit; /* 64-bit left rotate */

    // Get result from Non-Linear Lookup Table

    bitlu = 0;
    ifbit (csr, 1) setbit(bitlu,0);
    ifbit (csr, 9) setbit(bitlu,1);
    ifbit (csr,20) setbit(bitlu,2);
    ifbit (csr,26) setbit(bitlu,3);
    ifbit (csr,31) setbit(bitlu,4);

   // Calculate Result of XOR and shift in

    csr    = csr ^ bitin ^ ((csr&0x10000L)>>16) ^ lut[bitlu]
       ^ keybit2;
  }
    return csr;
}
```

 Confidential

APPENDIX B: CCS C COMPILER SOURCE CODE

```
//-------------------------------------------------------------------------
//  Keeloq Decryption Algorithm
//
//  optimized for CCS PIC C compiler v. 2.535
//
//  version 1.00     01/09/2001 Lucio Di Jasio
//
//  INPUTS:
//      DKEY[0..7]      64bit decryption key (pre loaded)
//      Buffer[0..3]    32bit shift register (pre loaded with text to decrytp)
//  OUTPUTS:
//      Buffer[0..3]    32bit plain text (decrypted message)
//
//=========================================================================
unsigned int DKEY[8];
unsigned int Buffer[3];

void Decrypt()
{
    unsigned int    i, j, key, aux; // 8bit variables
    signed int      p;              // 7bit +sign

    p = 1;

    for (j=66; j>0; j--)
    {
        key = DKEY[p--];
        if (p<0)
            p+=8;

        for (i=8; i>0; i--)
        {
            // NLF
            if ( BIT_TEST( Buffer[3],6))
            {
                if ( !BIT_TEST( Buffer[3],1))
                    aux = 0b00111010;   // 10
                else
                    aux = 0b01011100;   // 11
            }
            else
            {
                if ( !BIT_TEST( Buffer[3],1))
                    aux = 0b01110100;   // 00
                else
                    aux = 0b00101110;   // 01
            }


            // move bit in position 7
            if ( BIT_TEST( Buffer[2],3))
                SWAP( aux);
            if ( BIT_TEST( Buffer[1],0))
                aux<<=2;
            if (BIT_TEST( Buffer[0],0))
                aux<<=1;

            // xor with Buffer and Dkey
            aux ^= Buffer[1] ^ Buffer[3] ^ key;

            // shift in buffer
            SHIFT_LEFT( Buffer, 4, BIT_TEST( aux,7));

            key<<=1;
```

```
        } // for i

    } // for j
} // decrypt
```


APPENDIX C: HITECH C SOURCE CODE

```
//-------------------------------------------------------------------------
//  Keeloq Decryption Algorithm
//
//  optimized for HITECH PIC C compiler v.7.85
//
//  version 1.00     01/09/2001 Lucio Di Jasio
//
//  INPUTS:
//      DKEY[0..7]      64bit decryption key (pre loaded)
//      Buffer[0..3]    32bit shift register (pre loaded with text to decrytp)
//  OUTPUTS:
//      Buffer[0..3]    32bit plain text (decrypted message)
//
//=========================================================================
unsigned char DKEY[8];
unsigned char Buffer[4];

#define BIT_TEST( b, n) (( (b) & (1<<(n))) != 0)

unsigned char aux;      // keep it global for simple use in asm()

void Decrypt()
{
    unsigned char   i, j, key;  // 8 bit unsigned
    signed char     p;          // 8 bit signed

    p = 1;

    for (j=66; j>0; j--)
    {
        key = DKEY[p--];
        if ( p < 0)
            p += 8;
        for (i=8; i>0; i--)
        {
            // NLF
            if ( BIT_TEST( Buffer[3],6))
            {
                if ( !BIT_TEST( Buffer[3],1))
                    aux = 0b00111010;   // 10
                else
                    aux = 0b01011100;   // 11
            }
            else
            {
                if ( !BIT_TEST( Buffer[3],1))
                    aux = 0b01110100;   // 00
                else
                    aux = 0b00101110;   // 01
            }


            // move bit in position 7
            if ( BIT_TEST( Buffer[2],3))
                asm("swapf _aux,f");
            if ( BIT_TEST( Buffer[1],0))
                aux<<=2;
            if (BIT_TEST( Buffer[0],0))
                aux<<=1;

            // xor with Buffer and Dkey
            aux ^= Buffer[1] ^ Buffer[3] ^ key;

            // shift in buffer
```

```
            #asm
            rlf _aux,w
            rlf _Buffer,f
            rlf _Buffer+1,f
            rlf _Buffer+2,f
            rlf _Buffer+3,F
            #endasm

            // rotate Dkey
            key<<=1;
        } // for i
    } // for j
} // decrypt
```

 Confidential

NOTES:

NOTES:

Confidential

Trademarks

Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs and microperipheral products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.

MICROCHIP

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200 Fax: 480-792-7277
Technical Support: 480-792-7627
Web Address: http://www.microchip.com

Rocky Mountain
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7966 Fax: 480-792-7456

Atlanta
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770-640-0034 Fax: 770-640-0307

Austin
Analog Product Sales
8303 MoPac Expressway North
Suite A-201
Austin, TX 78759
Tel: 512-345-2030 Fax: 512-345-6085

Boston
2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848 Fax: 978-692-3821

Boston
Analog Product Sales
Unit A-8-1 Millbrook Tarry Condominium
97 Lowell Road
Concord, MA 01742
Tel: 978-371-6400 Fax: 978-371-0050

Chicago
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071 Fax: 630-285-0075

Dallas
4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423 Fax: 972-818-2924

Dayton
Two Prestige Place, Suite 130
Miamisburg, OH 45342
Tel: 937-291-1654 Fax: 937-291-9175

Detroit
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250 Fax: 248-538-2260

Los Angeles
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888 Fax: 949-263-1338

Mountain View
Analog Product Sales
1300 Terra Bella Avenue
Mountain View, CA 94043-1836
Tel: 650-968-9241 Fax: 650-967-1590

New York
150 Motor Parkway, Suite 202
Hauppauge, NY 11788
Tel: 631-273-5305 Fax: 631-273-5335

San Jose
Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408-436-7950 Fax: 408-436-7955

Toronto
6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699 Fax: 905-673-6509

ASIA/PACIFIC

Australia
Microchip Technology Australia Pty Ltd
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733 Fax: 61-2-9868-6755

China - Beijing
Microchip Technology Beijing Office
Unit 915
New China Hong Kong Manhattan Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100 Fax: 86-10-85282104

China - Shanghai
Microchip Technology Shanghai Office
Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700 Fax: 86-21-6275-5060

Hong Kong
Microchip Asia Pacific
RM 2101, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200 Fax: 852-2401-3431

India
Microchip Technology Inc.
India Liaison Office
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaugnessey Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

Japan
Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

ASIA/PACIFIC (continued)

Korea
Microchip Technology Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea
Tel: 82-2-554-7200 Fax: 82-2-558-5934

Singapore
Microchip Technology Singapore Pte Ltd.
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-334-8870 Fax: 65-334-8850

Taiwan
Microchip Technology Taiwan
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

Denmark
Microchip Technology Denmark ApS
Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45 4420 9895 Fax: 45 4420 9910

France
Arizona Microchip Technology SARL
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

Germany
Arizona Microchip Technology GmbH
Gustav-Heinemann Ring 125
D-81739 Munich, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

Germany
Analog Product Sales
Lochhamer Strasse 13
D-82152 Martinsried, Germany
Tel: 49-89-895650-0 Fax: 49-89-895650-22

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-039-65791-1 Fax: 39-039-6899883

United Kingdom
Arizona Microchip Technology Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5869 Fax: 44-118 921-5820

01/30/01