Document Title

A7125 Hopping reference code for FIFO mode

Revision History

Rev. No.	History	Issue Date	Remark
0.0	Preliminary	Aug. 28 , 2008	
0.1	Modify initial configuration value & delete section 3.2	Nov. 12, 2008	
1.0	Modify initial configuration value	Dec. 19, 2008	
1.1	Modify initial configuration value	Jan. 13, 2009	
1.2	Modify initial configuration value QDS=1, XCP[1:0]=[00], IGFQ[2:0]=[111], IGFI[2:0]=[111], PDL[1:0]=[00], WSEL[2:0]=[011]	Apr. 22, 2009	

Table of contents

RF Chip-A7125
Hopping reference code for FIFO mode

1. 簡介

這文件係對 RF chip -A7125 FIFO mode 做一簡單的應用範例程式 ，供使用者能夠快速應用這 RF chip。

2. 系統概述

本範例程式使用簡單的跳頻(frequency hopping)機制，時序如下圖：

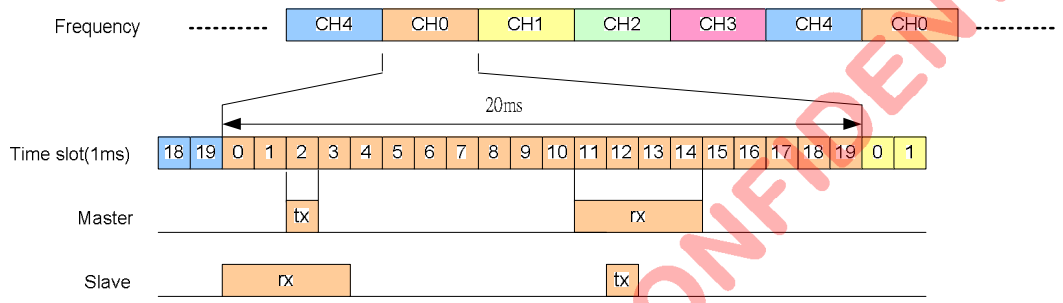

Fig1. 跳頻機制時序圖

程式主要分二個部份，一個爲 master 端，另一個爲 slave 端。

Master 端：power on、initial 系統及 RF chip 後，等待 time slot=2 時，進入 TX 狀態，傳送 64 bytes 資料。之後等待 time slot=11 時，再進入 RX 狀態，等待接收。如收到資料，會自動改變下一次的工作頻率序列，重新另一次的時序週期動作。若未收到資料，Master 端會自動改變下一次的工作頻率序列，重新另一次的時序週期動作。

Slave 端：power on、initial 系統及 RF chip 後，等待 time slot=0 時，進入 RX 狀態等待接收。若無收到 Master 端所發送的資料，則會自動改變下一次的工作頻率序列，等待下一次 time slot=0 的時序週期動作。若仍未收到資料有 5 次時序週期，則停止跳頻機制，並回到初始工作頻率，進入 RX 狀態等待接收。若有收到 Master 端所發送的資料，則重新啓動跳頻機制，依時序完成 TX 及 RX 工作。

一旦接收到封包，讀出資料、比對，計算 error bit 後，再發送封包給 Master 端。使用者可依據簡易的計算 error bit 及傳送封包數，得出 BER(bit error rate)，作爲傳輸品質的數據。

3. 硬體

3.1 系統方塊圖

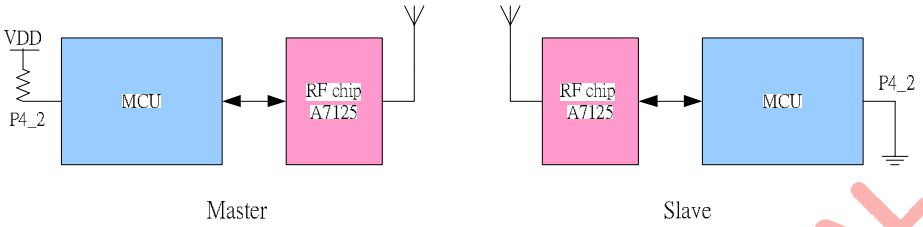

Fig2. 系統方塊圖

MCU 使用 I/O pin 4_2 的設定，判別 Master 端或 Slave 端。

使用 I/O pin 設定：
應用範例使用 I/O：
SCS, SCK, SDIO - 這 3 wire 串列介面控制 A7125 內部 register。
GIO1 - FIFO 動作完成的控制信號，MCU 可檢測該 pin 是否傳送或接收 packet 完成。

MCU 控制 A7125 RF chip 的 I/O 配置如下圖：

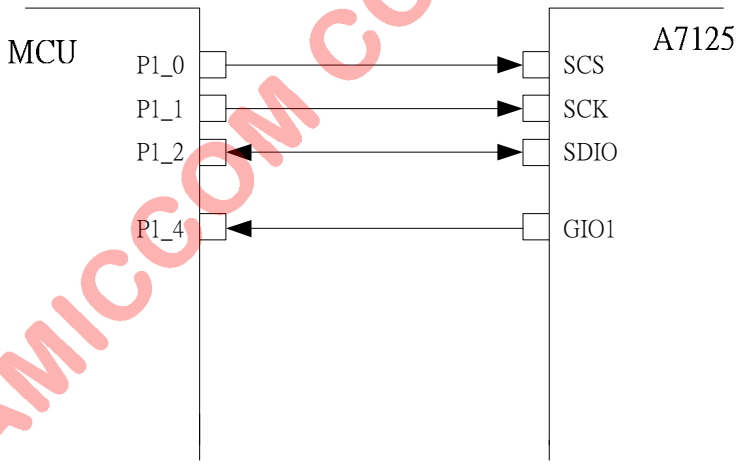

Fig3. I/O 配置圖

4. 韌體程式設計:

4.1 應用範例概述

首先初始化 Timer0、Uart0 及 A7125RF chip，之後判別 Port 4_2 =1 進入 master 端的主程式或 Port 4_2 =0 進入 slave 端的主程式。

Master 端：

1) 等待 timer=2。
2) 進入 TX state，傳送封包。完成傳送後，RF chip 會自動結束 TX state，回復到 Standby state。
3) 等待 timer=12 時，進入 RX 狀態。
4) 進入 RX 狀態。
5) 如 Time>14 時，仍未收到資料，則結束 RX 狀態。變數 seq 值加 1。回到 Step 1 動作，開始下一周期時序(另一工作頻率)的傳送。
6) 如收到封包後，RF chip 會自動結束 RX state，回復到 Standby state。
7) 從 RX FIFO 讀出，並比較 PN9 code 共 64bytes，並計算 error bit 數目。
8) 變數 seq 值加 1，重新回到 Step 1 動作，重新開始下一周期時序工作。
9) 每 500ms，將所計算的 error bit 傳送至 PC。

Slave 端：

1) 等待 timer=0。
2) 進入 RX 狀態，等待封包收到。
3) 如 timer>3 時，仍未收到資料，則結束 RX 狀態。變數 seq 值加 1，變數 Err_HopCnt 值加 1。
4) 如 Err_HopCnt 值沒有大於 5 次，則重新回到 Step 1 動作，重新開始下一周期時序工作。
5) 如 Err_HopCnt 值大於 5 次，則清除變數 seq 值爲 0，及 Err_HopCnt 值爲 0。使用 seq=0 的工作頻率進入 RX state，等待封包收到。
6) 如收到封包後，RF chip 會自動結束 RX state，回復到 Standby state。
7) 從 RX FIFO 讀出，並比較 PN9 code 共 64bytes，計算 error bit 數目。
8) 等待 timer=12 時，進入 TX 狀態，傳送封包。完成傳送後，RF chip 會自動結束 TX state，回復到 Standby state。
9) 重新回到 Step 1 動作，重新開始下一周期時序工作。
10) 每 500ms，將所計算的 error bit 傳送至 PC。

4.2 範例程式工作基本方塊

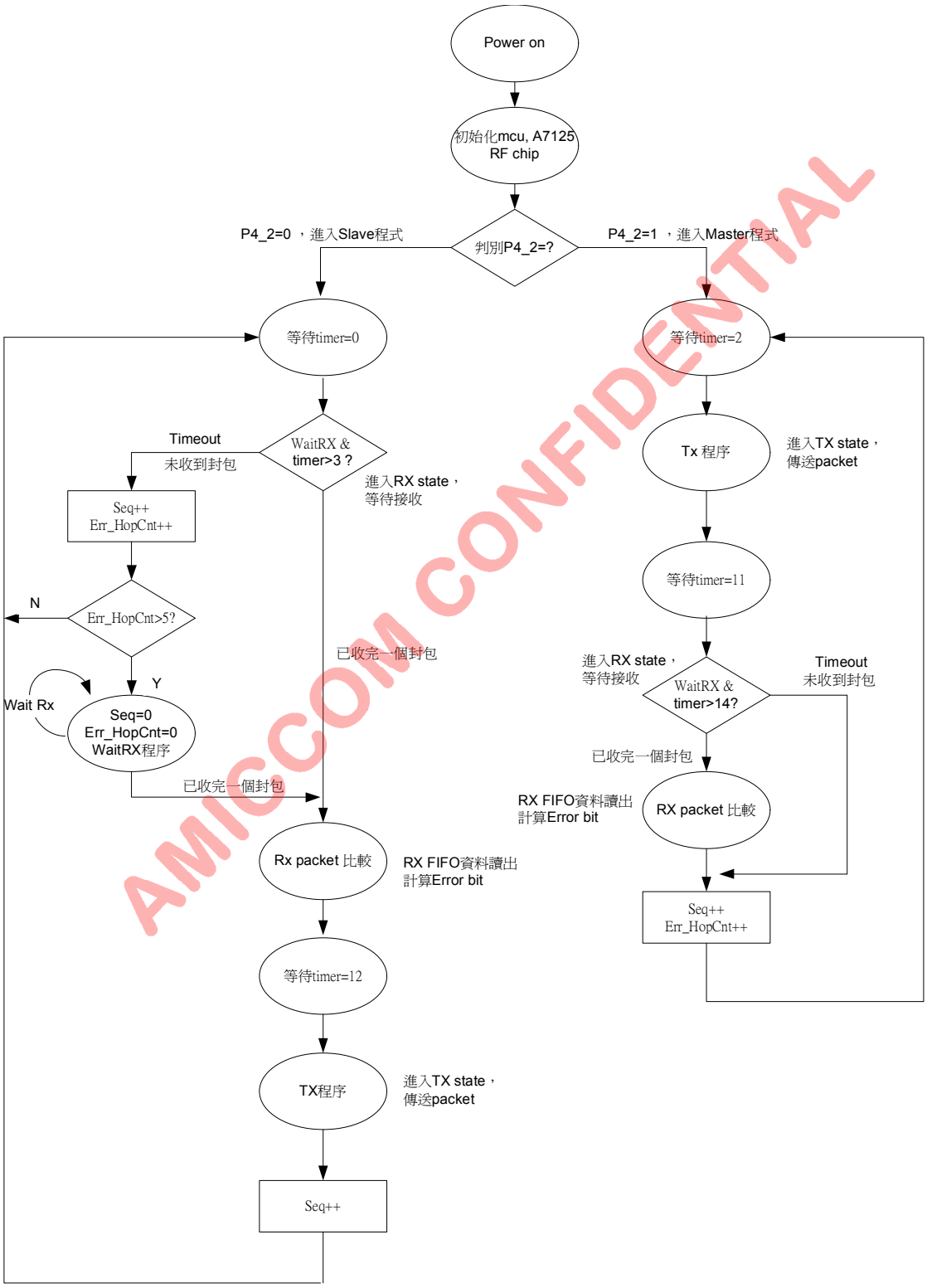

Fig4 範例程式工作基本方塊

5. 程式說明

```
1  /*********************************************************************
2   **  Device:     A7125
3   **  File:       main.c
4   **  Author:     JPH
5   **  Target:     Winbond W77LE58
6   **  Tools:      ICE
7   **  Created:    2009-04-22
8   **  Description:
9   **  This file is a sample code for your reference.
10  **
11  **  Copyright (C) 2008 AMICCOM Corp.
12  **
13  **
14  **
15  **
16  **
17  *********************************************************************/
18  #include "define.h"
19  #include "w77le58.h"
20  #include "a7125reg.h"
21  #include "Uti.h"
```

功能說明：Include 檔宣告，定義常數變數	
行數	說明
18~21	匯入程式庫設定檔

```
23  /*********************************************************************
24  **  I/O Declaration
25  *********************************************************************/
26  #define SCS         P1_0        //spi SCS
27  #define SCK         P1_1        //spi SCK
28  #define SDIO        P1_2        //spi SDIO
29  #define CKO         P1_3        //CKO
30  #define GIO1        P1_4        //GIO1
31  #define GIO2        P1_5        //GIO2
32  #define Button      P1_7        //test Button
33
34  /*********************************************************************
35  **  Constant Declaration
36  *********************************************************************/
37  #define TIMEOUT       50
38  #define t0hrel      1000
```

功能說明：MCU 對 A7125 RF chip I/O 接腳定義，常數定義	
行數	說明
26~32	MCU I/O 配置
37~38	常數定義

```
40  /*********************************************************************
41  **  Global Variable Declaration
42  *********************************************************************/
43  Uint8      data    timer;
44  Uint16     idata   RxCnt;
45  Uint32     idata   Err_ByteCnt;
46  Uint32     idata   Err_BitCnt;
47  Uint16     idata   TimerCnt0;
48  Uint8      data    *Uartptr;
49  Uint8      data    UartSendCnt;
50  Uint8      data    CmdBuf[12];
51  Uint8      xdata   tmpbuf[64];
52  Uint8      idata   Err_Frame;
53  Uint8      data    ReportTimeOut;
54  Uint8      data    Seq;
55  Uint8      data    Err_HopCnt;
56
57  const Uint8 code BitCount_Tab[16] = {0,1,1,2,1,2,2,3,1,2,2,3,2,3,3,4};
58  const Uint8 code ID_Tab[4]={0x54,0x75,0xC5,0x2A}; //ID code
59  const Uint8 code PN9_Tab[]=
60  {   0xFF,0x83,0xDF,0x17,0x32,0x09,0x4E,0xD1,
61      0xE7,0xCD,0x8A,0x91,0xC6,0xD5,0xC4,0xC4,
62      0x40,0x21,0x18,0x4E,0x55,0x86,0xF4,0xDC,
63      0x8A,0x15,0xA7,0xEC,0x92,0xDF,0x93,0x53,
64      0x30,0x18,0xCA,0x34,0xBF,0xA2,0xC7,0x59,
65      0x67,0x8F,0xBA,0x0D,0x6D,0xD8,0x2D,0x7D,
66      0x54,0x0A,0x57,0x97,0x70,0x39,0xD2,0x7A,
67      0xEA,0x24,0x33,0x85,0xED,0x9A,0x1D,0xE0
68  };      // This table are 64bytes PN9 pseudo random code.
```

功能說明：使用的整體變數宣告，常數變數的宣告

行數	說明
43~55	程式中使用的變數宣告
57	BitCount_Tab 宣告
58	ID code 宣告
59~68	PN9 data 宣告

```
70   const Uint16 code A7125Config[]=
 71  {
 72      0x00, //MODE register,          only reset, not use on config
 73      0x42, //MODE CTRL register,
 74      0x00, //CALIBRATION register,
 75      0x3F, //FIFO1 register,
 76      0x00, //FIFO2 register,
 77      0x00, //FIFO register,          for fifo read/write
 78      0x00, //IDDATA register,        for idcode
 79      0x00, //RCOSC1 register,
 80      0x00, //RCOSC2 register,
 81      0x00, //RCOSC3 register,
 82      0x00, //CKO register,
 83      0x01, //GPIO1 register
 84      0x00, //GPIO2 register,
 85      0x1F, //DATARATE register,
 86      0x50, //PLL1 register,
 87      0x0E, //PLL2 register,          RFbase 2400.001MHz
 88      0x96, //PLL3 register,
 89      0x00, //PLL4 register,
 90      0x04, //PLL5 register,
 91      0x3C, //chanel group I,
 92      0x78, //chanel group II,
 93      0xD7, //TX1 register,
 94      0x40, //TX2 register,
 95      0x10, //DELAY1 register,
 96      0x61, //DELAY2 register,
 97      0x62, //RX register,
 98      0xA0, //RXGAIN1 register,
 99      0x00, //RXGAIN2 register,
100      0x00, //RXGAIN3 register,
101      0xD2, //RXGAIN4 register,
102      0x00, //RSSI register,
103      0xE2, //ADC register,
104      0x07, //CODE1 register,
105      0x56, //CODE2 register,
106      0x2A, //CODE3 register,
107      0x06, //IFCAL1 register,
108      0x00, //IFCAL2 register,   only read
109      0x05, //VCOCCAL register,
110      0x44, //VCOCAL1 register,
111      0x80, //VCOCAL2 register,
112      0x30, //VCO DEV Cal. I register,
113      0x20, //VCO DEV Cal. II register,
114      0x80, //VCO DEV Cal. III register,
115      0x00, //VCO Mod. delay register
116      0x7A, //BATTERY register,
117      0x2F, //TXTEST register,
118      0x47, //RXDEM1 register,
119      0x80, //RXDEM2 register,
120      0xF1, //CPC1 register,
121      0x11, //CPC2 register,
122      0x04, //CRYSTAL register,
123      0x45, //PLLTEST register,
124      0x18, //VCOTEST register,
125      0x10, //RF Analog Test
126      0xFF, //IFAT register,
127      0x37, //Channel select register,
128      0xFF  //VRB register
129  };
```

功能說明：Master 端 RF chip 初始設定

行數	說明

72~128	Master 端 RF chip 的初始設定

```
131   const Uint8 HopTab[]=
132   {
133       20, //2410
134       40, //2420
135       80, //2440
136       120,//2460
137       160 //2480
138   };
```

功能說明：Hopping table 宣告

行數	說明
133~137	自行定義 5 個的 channel.

```
140   /*************************************************************************
141    **  function Declaration
142    *************************************************************************/
143   void InitTimer0(void);
144   void initUart0(void);
145   void Timer0ISR (void);
146   void Uart0Isr(void);
147   void A7125_Reset(void);
148   void A7125_WriteReg(Uint8, Uint8);
149   Uint8 A7125_ReadReg(Uint8);
150   void ByteSend(Uint8 src);
151   Uint8 ByteRead(void);
152   void A7125_WriteID(void);
153   void A7125_WriteFIFO(void);
154   void initRF(void);
155   void A7125_Config(void);
156   void A7125_Cal(void);
157   void RxPacket(void);
158   void StrobeCmd(Uint8);
159   void SetCH(Uint8);
160   void CHGroupCal(Uint8);
161   void ReportData(void);
```

功能說明：副程式檔頭宣告

行數	說明
143~161	副程式宣告

```
/*****************************************************************
 * main loop
 *****************************************************************/
void main(void)
{
    //initsw
    PMR |= 0x01; //set DME0

    //initHW
    P0 = 0xFF;
    P1 = 0xFF;
    P2 = 0xFF;
    P3 = 0xFF;
    P4 = 0x0F;

    InitTimer0();
    initUart0();
    TR0=1; //timer0 on
    EA=1; //enable interrupt

    if ((P4 & 0x04)==0x04) //if P4.2=1, master
    {
            initRF(); //init RF

            StrobeCmd(CMD_STBY);
            A7125_WriteFIFO(); //write data to tx fifo
            Seq=0;

            while(1)
            {
                //Tx time-slot
                while(timer != 2); //wait until timer=2
                SetCH(HopTab[Seq]);
                StrobeCmd(CMD_TX); //entry tx & transmit
                while(GIO1); //wait transmit completed

                //Rx time-slot
                while(timer !=11); //wait until timer=11
                SetCH(HopTab[Seq]-4);
                StrobeCmd(CMD_RX);
                while(GIO1 && timer <= 14);

                if (timer >14)
                {
                    //timeout
                    StrobeCmd(CMD_STBY);
                }
                else
                {
                    //data procedure
                    RxPacket();
                }

                Seq++;
                if (Seq > 4)
                    Seq = 0;
            }
    }
```

```
else //if P4.2=0, slave
 {
    initRF(); //init RF

    StrobeCmd(CMD_STBY);
    A7125_WriteFIFO(); //write data to tx fifo

    RxCnt = 0;
    Err_ByteCnt = 0;
    Err_BitCnt = 0;

    Seq=0;
    Err_HopCnt=0;

    while(1)
    {
       if (P1_7==0)
       {
          RxCnt = 0;
          Err_BitCnt = 0;
       }

       //Rx time-slot
       while(timer!=0);
       SetCH(HopTab[Seq]-4);
       StrobeCmd(CMD_RX);
       while(GIO1 && timer <= 3); //wait receive completed
       if (timer > 3)
       {
          StrobeCmd(CMD_PLL);

          Seq++;
          if (Seq > 4)
             Seq = 0;

          Err_HopCnt++;
          if (Err_HopCnt > 5)
          {
             Seq = 0;
             Err_HopCnt = 0;

             SetCH(HopTab[Seq]-4);
             StrobeCmd(CMD_RX);
             while(1)
             {
                   if (GIO1==0)
                   {
                      break;
                   }
                }
          }
          else
          {
                continue;
          }
       }

       timer = 2; //reSync
       TF0 = 0; // Clear Timer0 interrupt
       TH0 = (65536-t0hrel)>>8; // Reload Timer0 high byte,low byte
       TL0 = 65536-t0hrel;
```

```
282
283             RxPacket();
284
285             //Tx time-slot
286             while(timer != 12);
287             SetCH(HopTab[Seq]);
288             StrobeCmd(CMD_TX);
289             while(GIO1);
290
291             Seq++;
292             if (Seq > 4)
293                Seq = 0;
294          }
295       }
296   }
```

功能說明：主程式 main loop。Port4_2=1，進入 master 迴圈，行數 184~220 爲 master 程式。Port4_2=0，進入 slave 迴圈，行數 222~295 爲 slave 程式。

行數	說明
169	啓用 MCU on chip data SRAM
172~176	初始化 MCU I/O Port
178	呼叫副程式 initTimer0，致能中斷
179	呼叫副程式 initUart0，初始 Uart0
180~181	啓動 Timer0 ，致能中斷開啓
183	判別 port4_2=1，進入 master 迴圈。Port4_2=0，進入 slave 迴圈。
184~220	Master 迴圈程式
185	呼叫副程式 initRF，初始化 A7125 chip
187	使用 Strobe command，進入 Standby 模式。
188	清除變數 seq=0
194	等待 timer=2
195	呼叫副程式 SetCH ，依 HopTab 查表設置工作頻率
196	設置 A7125 chip 進入 TX mode
197	判別 I/O GIO1 等待是否完成資料傳送
200	等待 timer=11
201	呼叫副程式 SetCH ，依 HopTab 查表設置工作頻率
202	使用 Strobe command，進入 RX 模式
203	等待是否收妥資料或是 timer>14
205~209	判別是否 timer>14。如是，則使用 Strobe command，進入 PLL state
211~214	已收妥資料，呼叫 RxPacket 副程式，從 RX FIFO 讀出資料、比對、計算 error bit 數
216~218	變數 seq 加 1，判別變數 seq 是否大於 4。如是，則清爲 0
222~295	Slave 迴圈程式
223	呼叫副程式 initRF，初始化 A7125 chip
225	使用 Strobe command，進入 Standby 模式。
226	呼叫副程式 A7125_WriteFIFO，將 data 寫入 TX FIFO
228~233	清除變數 RxCnt, Err_ByteCnt, Err_BitCnt, seq, Err_HopCnt 值
237~241	判別 MCU pin P1_7 是否爲 0。如是，則清除變數 seq, Err_HopCnt 值
244	等待 timer=0
245	呼叫副程式 SetCH ，依 HopTab 查表設置工作頻率
246	使用 Strobe command，進入 RX 模式
247	等待資料的收妥或是 timer>3
248	判別變數 timer 是否大於 3
250	使用 Strobe command，進入 PLL state

252~254	變數 seq 加 1，判別變數 seq 是否大於 4。如是，則清爲 0
256	變數 Err_HopCnt 加 1
257	判別 Err_HpoCnt 是否大於 5
259~260	清除變數 seq=0, Err_HopCnt=0
262	呼叫副程式 SetCH ，設置工作頻率
263	使用 Strobe command，進入 RX 模式
264~271	等待資料進入
274	變數 Err_HopCnt 值沒有大於 5 次時，則回到 237 行，重新另一次周期的工作
278	設置變數 timer=2，重新同步
279~281	清除 TF0 旗標，重新設置 TH0, TL0 值
283	呼叫副程式 RxPacket，從 RX FIFO 讀出資料、比對、計算 error bit 數
286	等待 timer=12
287	呼叫副程式 SetCH，設置工作頻率
288	使用 Strobe command，進入 TX 模式，發送資料
289	等待是否完成資料傳送
291~293	變數 seq 加 1，判別變數 seq 是否大於 4。如是，則清爲 0

```
298  /*****************************************************************************
299   **  init Timer0
300   *****************************************************************************/
301   void InitTimer0(void)
302   {
303       TR0 = 0;
304       TMOD =(TMOD & 0xF0)|0x01; //timer0 mode=1
305       TH0 = (65536-t0hrel)>>8; // Reload Timer0 high byte,low byte
306       TL0 = 65536-t0hrel;
307       TF0 = 0; // Clear any pending Timer0 interrupts
308       ET0 = 1; // Enable Timer0 interrupt
309   }
```

功能說明：初始化 Timer0 程序

行數	說明
303	關閉 Timer0 計時動作
304	設置 Timer0 在 mode 1 模式
305~306	設置 TH0,TL0 的初始值
307	清除 Timer0 中斷旗標
308	致能 Timer0 中斷

```
311  /*********************************************************************
312   **  Timer0ISR
313   *********************************************************************/
314   void Timer0ISR (void) interrupt 1
315   {
316      TF0 = 0; // Clear Timer0 interrupt
317      TH0 = (65536-t0hrel)>>8; // Reload Timer0 high byte,low byte
318      TL0 = 65536-t0hrel;
319
320      timer++;
321      if (timer>=20)
322      {
323         timer=0;
324         P3_5= ~P3_5;
325      }
326
327      TimerCnt0++;
328      if (TimerCnt0 == 500)
329      {
330         TimerCnt0=0;
331         CmdBuf[0]=0xF1;
332
333         memcpy(&CmdBuf[1], &RxCnt, 2);
334         memcpy(&CmdBuf[3], &Err_ByteCnt, 4);
335         memcpy(&CmdBuf[7], &Err_BitCnt, 4);
336         memcpy(&CmdBuf[11], &Err_Frame, 1);
337
338         UartSendCnt=12;
339         Uartptr=&CmdBuf[0];
340         SBUF=CmdBuf[0];
341      }
342   }
```

功能說明：初始化 Timer0 的中斷副程式

行數	說明
316~318	清除 Timer0 中斷旗標，設置 TH0,TL0 的啓始值
320	變數 timer 加 1
321~325	判別變數 timer 是否等於 20ms。如是，清除變數 timer=0，pin P3_5 信號反向.
327	變數 TimerCnt0 加 1
328	判別變數 TimerCnt0 是否等於 500(即 500ms)
330	清除變數 TimerCnt0
331	CmdBuf[0]設置 0xF1 爲傳送啓始位元識別碼
333	CmdBuf[1] 、CmdBuf[1]設置變數 RxCnt 的值
334	CmdBuf[3] 、CmdBuf[4]、CmdBuf[5] 、CmdBuf[6]設置變數 Err_ByteCnt 的值
335	CmdBuf[7] 、CmdBuf[8]、CmdBuf[9] 、CmdBuf[10]設置變數 Err_BitCn 的值
336	CmdBuf[11] 設置變數 Err_Frame 的值
338	設置變數UartSendCnt=12
339	設置指標變數 Uartptr 指到變數 CmdBuf[0]的啓始位址
340	傳送 SBUF 至 PC

```
344  /*************************************************************************
345   **  Init Uart0
346   *************************************************************************/
347  void initUart0(void)
348  {
349      TH1 = 0xFD; //BaudRate 9600;
350      TL1 = 0xFD;
351      SCON = 0x40;
352      TMOD = (TMOD & 0x0F) | 0x20;
353      REN = 1;
354      TR1 = 1;
355      ES = 1;
356  }
```

功能說明：初始化 Uart0 的程序

行數	說明
349~351	初始 TL1,TH1,SCON1 值，設置爲 9600bps @xtal=11.0592MHz
352	設置 Timer1 爲 mode 2
353~355	設置 REN,TR1,ES 爲 1，啓用 Uart0 的功能

```
358  /*************************************************************************
359   **  Uart0 ISR
360   *************************************************************************/
361  void Uart0Isr(void) interrupt 4 using 3
362  {
363      if (TI==1)
364      {
365          TI=0;
366          UartSendCnt--;
367          if(UartSendCnt !=0)
368          {
369              Uartptr++;
370              SBUF = *Uartptr;
371          }
372      }
373  }
```

功能說明：初始化 uart0 的中斷副程式

行數	說明
363	判別 TI 旗標是否爲 Uart 已傳送完成 1byte
365	清除 TI 旗標
366	變數 UartSendCnt 減 1
367	判別變數 UartSendCnt 是否爲 0。如不爲 0，則繼續傳送下一個資料
369~370	指標變數 Uartptr 加 1，並將其位址的資料，使用 Uart0 送至 PC

```
375  /*************************************************************************
376   **  Reset_RF
377   *************************************************************************/
378  void A7125_Reset(void)
379  {
380      A7125_WriteReg(MODE_REG, 0x00); //reset RF chip
381  }
```

功能說明：A7125 RF chip Reset 程序

行數	說明
380	對 register 位址 0 ，寫入 0x00，重置 RF chip。

```
383  /*************************************************************************
384   **  WriteID
385   *************************************************************************/
386   void A7125_WriteID(void)
387   {
388       Uint8 i;
389       Uint8 d1,d2,d3,d4;
390       Uint8 addr;
391
392       addr = IDCODE_REG; //send address 0x06, bit cmd=0, r/w=0
393       SCS = 0;
394       ByteSend(addr);
395       for (i=0; i < 4; i++)
396           ByteSend(ID_Tab[i]);
397       SCS = 1;
398
399       addr = IDCODE_REG | 0x40; //send address 0x06, bit cmd=0, r/w=1
400       SCS=0;
401       ByteSend(addr);
402       d1=ByteRead();
403       d2=ByteRead();
404       d3=ByteRead();
405       d4=ByteRead();
406       SCS=1;
407   }
```

功能說明：寫入 ID 的程序。

行數	說明
392	計算變數 addr 的值
393	SCS=0，設置控制暫存器讀寫功能
394~396	寫入 ID 控制暫存器的位址，及 4 bytes 的 ID code
397	SCS=1，清除 SPI 讀寫功能
399	計算讀出 ID code 的變數 addr 值
400	SCS=0，設置控制暫存器讀寫功能
401~405	讀出 ID code
406	SCS=1，清除控制暫存器讀寫功能

```
409 /*********************************************************************
410  **  A7125_WriteReg
411  *********************************************************************/
412 void A7125_WriteReg(Uint8 addr, Uint8 dataByte)
413 {
414     Uint8 i;
415
416     SCS = 0;
417
418     addr |= 0x00; //bit cmd=0,r/w=0
419     for(i = 0; i < 8; i++)
420     {
421         if(addr & 0x80)
422             SDIO = 1;
423         else
424             SDIO = 0;
425
426         SCK = 1;
427         _nop_();
428         SCK = 0;
429         addr = addr << 1;
430     }
431     _nop_();
432
433     //send data byte
434     for(i = 0; i < 8; i++)
435     {
436         if(dataByte & 0x80)
437             SDIO = 1;
438         else
439             SDIO = 0;
440
441         SCK = 1;
442         _nop_();
443         SCK = 0;
444         dataByte = dataByte<< 1;
445     }
446
447     SCS = 1;
448 }
```

功能說明：對 A7125 控制暫存器(Control Register)寫入動作

行數	說明
416	SCS=0，致能控制暫存器讀寫功能
418	將 address Or 寫入控制暫存器命令。
419~430	寫入 address 的程序
434~445	寫入 data word 的程序
447	SCS=1，清除控制暫存器讀寫功能

```
450 /*********************************************************************
451  **  A7125_ReadReg
452  *********************************************************************/
453 Uint8 A7125_ReadReg(Uint8 addr)
454 {
455     Uint8 i;
456     Uint8 tmp;
457
458     SCS = 0;
459     addr |= 0x40; //bit cmd=0,r/w=1
460     for(i = 0; i < 8; i++)
461     {
462
463         if(addr & 0x80)
464             SDIO = 1;
465         else
466             SDIO = 0;
467
468         _nop_();
469         SCK = 1;
470         _nop_();
471         SCK = 0;
472
473         addr = addr << 1;
474     }
475
476     _nop_();
477     SDIO = 1;
478
479     //read data
480     for(i = 0; i < 8; i++)
481     {
482         if(SDIO)
483             tmp = (tmp << 1) | 0x01;
484         else
485             tmp = tmp << 1;
486
487         SCK = 1;
488         _nop_();
489         SCK = 0;
490     }
491     SCS = 1;
492     return tmp;
493 }
```

功能說明：A7125 控制暫存器(Control Register)讀出動作

行數	說明
458	SCS=0，致能控制暫存器讀寫功能
459	將 address Or 讀出控制暫存器命令。
460~474	寫入 address 的程序
477	設置 SDIO 爲輸出模式
480~490	讀出資料
491	SCS=0，清除控制暫存器讀寫功能
492	回傳 1 byte 的讀值

```
495 /*****************************************************************
496  **  ByteSend
497  *****************************************************************/
498 void ByteSend(Uint8 src)
499 {
500     Uint8 i;
501
502     for(i = 0; i < 8; i++)
503     {
504         if(src & 0x80)
505             SDIO = 1;
506         else
507             SDIO = 0;
508
509         _nop_();
510         SCK = 1;
511         _nop_();
512         SCK = 0;
513         src = src << 1;
514     }
515 }
```

功能說明：寫入 1 byte 的程序

行數	說明
502~514	寫入 1 個 byte 的程序

```
517 /*****************************************************************
518  **  ByteRead
519  *****************************************************************/
520 Uint8 ByteRead(void)
521 {
522     Uint8 i,tmp;
523
524     SDIO = 1; //sdio pull high
525     for(i = 0; i < 8; i++)
526     {
527         if(SDIO)
528             tmp = (tmp << 1) | 0x01;
529         else
530             tmp = tmp << 1;
531
532         SCK = 1;
533         _nop_();
534         SCK = 0;
535     }
536     return tmp;
537 }
```

功能說明：讀出 1byte 的程序

行數	說明
524~535	讀出 1 個 byte 的程序
536	返回 8 bit 的讀值

```
539 /*************************************************************************
540  **  Send4Bit
541  *************************************************************************/
542 void Send4Bit(Uint8 src)
543 {
544     Uint8 i;
545
546     for(i = 0; i < 4; i++)
547     {
548         if(src & 0x80)
549             SDIO = 1;
550         else
551             SDIO = 0;
552
553         _nop_();
554         SCK = 1;
555         _nop_();
556         SCK = 0;
557         src = src << 1;
558     }
559 }
```

功能說明：寫入 4 bit 的程序

行數	說明
544~558	寫入 1 個 byte 的程序

```
561 /*************************************************************************
562  ** SetCH
563  *************************************************************************/
564 void SetCH(Uint8 ch)
565 {
566     A7125_WriteReg(PLL1_REG, ch); //RF freq = RFbase + (CH_Step * ch)
567 }
```

功能說明：設置頻道的程序

行數	說明
566	呼叫副程式 A7125_WriteReg，對 PLL1 控制暫存器寫入工作頻道值。

```
569 /*********************************************************************
570  ** initRF
571  *********************************************************************/
572  void initRF(void)
573  {
574      //init io pin
575      SCS = 1;
576      SCK = 0;
577      SDIO = 1;
578      CKO = 1;
579      GIO1 = 1;
580      GIO2 = 1;
581
582      A7125_Reset(); //reset A7125 RF chip
583      A7125_WriteID(); //write ID code
584      A7125_Config(); //config A7125 chip
585      A7125_Cal(); //calibration IF,VCO,VCOC
586  }
```

功能說明：初始化 Master 端的 RF chip

行數	說明
575~580	設置 RF chip 介面 I/O 初始值
582	呼叫副程式 A7125_Reset，重置 RF chip
583	呼叫副程式 A7125_WriteID，寫入 ID code 4ytes
584	呼叫副程式 A7125_Config，初始控制暫存器
585	呼叫副程式 A7125_Cal，RSSI, IF, VCO, VCO current , VCO deviation 的校準程序

```
588 /*********************************************************************
589  ** A7125_WriteFIFO
590  *********************************************************************/
591  void A7125_WriteFIFO(void)
592  {
593      Uint8 i;
594      Uint8 cmd;
595
596      cmd = FIFO_REG; //send address 0x05, bit cmd=0, r/w=0
597      SCS=0;
598      ByteSend(cmd);
599      for(i=0; i <64; i++)
600          ByteSend(PN9_Tab[i]);
601      SCS=1;
602  }
```

功能說明：Tx FIFO 寫入資料的程序

行數	說明
596	將 FIFO 控制暫存器位址與 cmd bit, r/w bit 作運算，寫入 TX FIFO 控制暫存器命令
597	SCS=0，致能控制暫存器讀寫功能
598	送出 TX FIFO 寫入命令
599~600	寫入 64 bytes 的資料
601	SCS=1，清除控制暫存器讀寫功能

```
604 /*********************************************************************
605  ** Strobe Command
606  *********************************************************************/
607  void StrobeCmd(Uint8 cmd)
608  {
609      SCS = 0;
610      Send4Bit(cmd);
611      SCS = 1;
612  }
```

功能說明：Strobe 命令寫入的程序。

行數	說明
609	SCS=0，致能控制暫存器讀寫功能
610	呼叫副程式 Send4Bit，將控制指令寫入
611	SCS=1，清除控制暫存器讀寫功能

```
614 /*********************************************************************
615  ** RxPacket
616  *********************************************************************/
617  void RxPacket(void)
618  {
619      Uint8 i;
620      Uint8 recv;
621      Uint8 tmp;
622      Uint8 cmd;
623
624      RxCnt++;
625      cmd = FIFO_REG | 0x40; //address 0x05, bit cmd=0, r/w=1
626
627      SCS=0;
628      ByteSend(cmd);
629      for(i=0; i <64; i++)
630      {
631          recv = ByteRead();
632          tmpbuf[i]=recv;
633          if((recv ^ PN9_Tab[i])!=0)
634          {
635              tmp = recv ^ PN9_Tab[i];
636              Err_BitCnt += (BitCount_Tab[tmp>>4] + BitCount_Tab[tmp & 0x0F]);
637          }
638      }
639      SCS=1;
640  }
```

功能說明：從 RX FIFO 讀出資料，比對資料的程序

行數	說明
624	變數 RxCnt 加 1
625	將 FIFO 控制暫存器位址與 cmd bit, r/w bit 作運算，讀出 RX FIFO 控制暫存器命令
627	SCS=0，致能控制暫存器讀寫功能
628	呼叫副程式 ByteSend，送出控制命令
629~638	讀出 data ，比較 data 的正確性，計算出 error bit
639	SCS=1，清除控制暫存器讀寫功能

```
642  /*********************************************************************
643   ** Err_State
644   *********************************************************************/
645   void Err_State(void)
646   {
647       //ERR display
648       //Error Proc...
649       //...
650       while(1);
651   }
```

功能說明：Error state 處理程序	
行數	說明
647~650	Error 處理程序，由使用者自訂

```
653  /*********************************************************************
654   ** CHGroupCal
655   *********************************************************************/
656   void CHGroupCal(Uint8 ch)
657   {
658       Uint8 tmp;
659       Uint8 vb,vbcf;
660       Uint8 vcb,vccf;
661
662       A7125_WriteReg(PLL1_REG, ch);
663       Delay10us(4);
664       A7125_WriteReg(CALIBRATION_REG, 0x1C);
665      do{
666          tmp = A7125_ReadReg(CALIBRATION_REG);
667          tmp &= 0x1C;
668       }
669       while (tmp);
670
671       //for check
672       tmp = A7125_ReadReg(VCOCAL1_REG);
673       vb = tmp & 0x07;
674       vbcf = (tmp >>3) & 0x01;
675
676       tmp = A7125_ReadReg(VCOCCAL_REG);
677       vcb = tmp & 0x0F;
678       vccf= (tmp >> 4) & 0x01;
679
680       if (vbcf || vccf)
681           Err_State();//error
682   }
```

功能說明：Channel Group 校準處理程序	
行數	說明
662	呼叫副程式 A7125_WriteReg，寫入 PLL1 控制暫存器位址 。
663	延遲 40us，使 PLL setting 穩定。
664	設置 bit VCC=1, VBC=1, VDC=1，對 VCO current, VCO band, VCO deviation 作校準動作。
665~669	讀出 Calibration registerI，並判別 bit VCC, VBC, VDC 是否爲 0。如爲 0，則跳出等待迴圈。
672~678	讀出 VCO Calibration I, VCO Current Calibration 控制暫存器，並檢示其值。
680~681	判別旗標 vbcf, vccf 是否爲 0。如爲 1，則進入 Error state 處理程序。

```
684  /*********************************************************************
685   ** calibration
686   *********************************************************************/
687   void A7125_Cal(void)
688   {
689       Uint8 tmp;
690       Uint8 fb,fbcf,fcd;
691       Uint8 dvt;
692
693       //calibration RSSI,IF procedure
694       StrobeCmd(CMD_PLL);
695       A7125_WriteReg(CALIBRATION_REG, 0x03);
696       do
697       {
698           tmp = A7125_ReadReg(CALIBRATION_REG);
699           tmp &= 0x03;
700       }
701       while (tmp);
702
703       //calibration VCO dev,VBC,VCC procedure
704       CHGroupCal(30); //calibrate channel group Bank I
705       CHGroupCal(90); //calibrate channel group Bank II
706       CHGroupCal(150);//calibrate channel group Bank III
707       StrobeCmd(CMD_STBY); //return to STBY state
708
709       //for check
710       tmp = A7125_ReadReg(IFCAL1_REG);
711       fb = tmp & 0x0F;
712       fbcf = (tmp >>4) & 0x01;
713
714       tmp = A7125_ReadReg(IFCAL2_REG);
715       fcd = tmp & 0x1F;
716
717       tmp = A7125_ReadReg(VCOCAL2_REG);
718       dvt = tmp;
719
720       if (fbcf)
721           Err_State(); //error
722   }
```

功能說明：RSSI , IF, VCO, VCO current, VCO deviation 校準程序

行數	說明
694	使用 Strobe 命令，設置 RF chip 進入 PLL state
695	設置 Calibration control register 中 bit RSSC=1, bit FBC=1。
696~701	讀出 Calibration control register ，並判別 bit RSSC, bit FBC 是否爲 0。如爲 0，則跳出等待迴圈
704~706	呼叫副程式 CHGroupCal，校準 Channel group 程序。
707	使用 Strobe 命令，設置 RF chip 進入 Standby state。
710~718	讀出 IF Calibration I, IF Calibration II, VCO Calibration II 制暫存器，並檢示其值
720~721	判別旗標 fbcf 是否爲 0。如爲 1，則進入 Error state 處理程序。

```
724  /*********************************************************************
725   ** A7125_Config
726   *********************************************************************/
727   void A7125_Config(void)
728   {
729       Uint8 i;
730
731       //0x00 mode register, for reset
732       //0x05 fifo data register
733       //0x06 id code register
734       //0x24 IF calibration II, only read
735
736       for (i=0x01; i<=0x04; i++)
737           A7125_WriteReg(i, A7125Config[i]);
738
739       for (i=0x07; i<=0x23; i++)
740           A7125_WriteReg(i, A7125Config[i]);
741
742       for (i=0x25; i<=0x38; i++)
743           A7125_WriteReg(i, A7125Config[i]);
744   }
```

功能說明：初始 RF chip 的程序

行數	說明
736~737	呼叫副程式 A7125_WriteReg，寫入控制暫存器位址 0x01~0x04
739~740	呼叫副程式 A7125_WriteReg，寫入控制暫存器位址 0x07~0x23
742~743	呼叫副程式 A7125_WriteReg，寫入控制暫存器位址 0x25~0x38