CMT2150A & CMT2210A Communication Example (1527coding)

This chapter will guide the user to carry out the communication experiment on a pair of single transmitting and single receiving chips CMT2150A and CMT2210A. CMT2150A is the OOK modulated single transmitting chip with coding function belonging to HopeRF's CMOSTEK wireless product line below. CMT2210A is the OOK modulated single receiving chip, all support Sub-1G applications.

1. Tools and software needed to be prepared

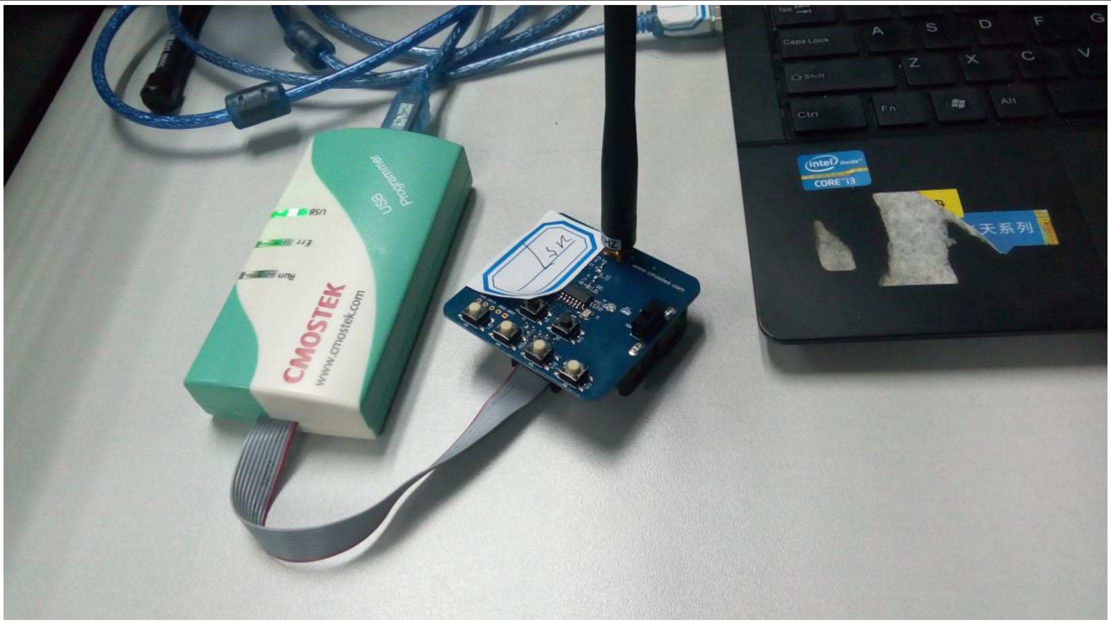
- Arduino IDE version 1.0.5
- HopeDuino board
  (If you have not used the HopeDuino board, please refer to the 《AN0002-HopeDuino Platform Construction Guideline》)
- USB cable(Type A to Type B)
- CMT2150A-EM board（Or product based on CMT2150A chip design）
- CMOSTEK USB Programmer
- CMOSTEK RFPDK V1.38（Pay attention to using the latest version. The latest version is V1.38 in the paper）
- Module RFM210 (Based on chip CMT2210A) and the matching conversion board
-

RFM210

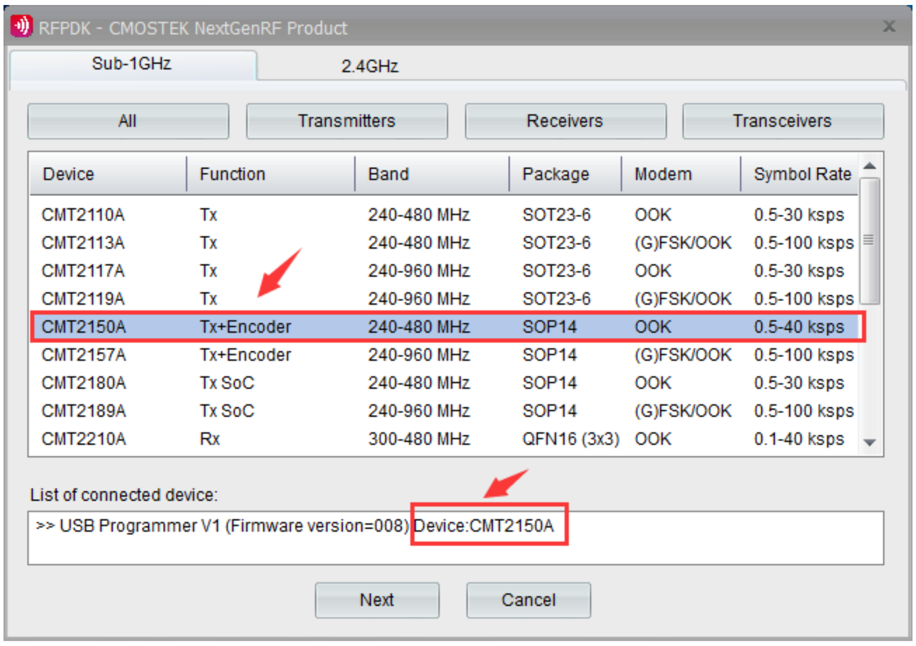
CMT2150A - EM

2. CMT2150A parameter configuration and burning

- CMT2150A configuration details refer to CMOSTEK《AN112 CMT2150A Configuration Guideline》
- This paper focuses on the experiment for the purpose, configuring parameters and demonstrating effect simply.
  - Connect CMT2150A-EM to PC with USB Programmer



- Open CMOSTEK RFPDK interface, select“CMT2150A”as below, click and enter:

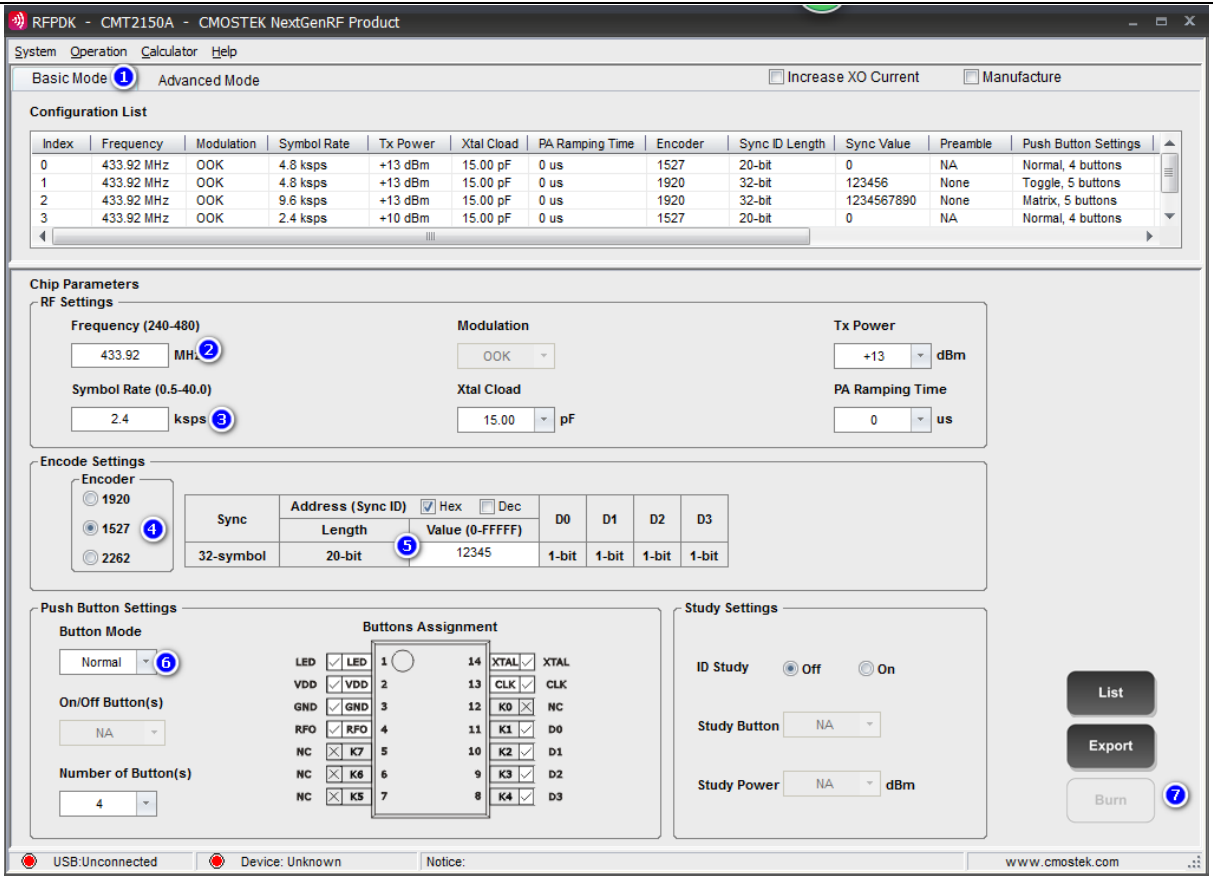


Device	Function	Band	Package	Modem	Symbol Rate
CMT2110A	Tx	240-480 MHz	SOT23-6	OOK	0.5-30 ksps
CMT2113A	Tx	240-480 MHz	SOT23-6	(G)FSK/OOK	0.5-100 ksps
CMT2117A	Tx	240-960 MHz	SOT23-6	OOK	0.5-30 ksps
CMT2119A	Tx	240-960 MHz	SOT23-6	(G)FSK/OOK	0.5-100 ksps
CMT2150A	Tx+Encoder	240-480 MHz	SOP14	OOK	0.5-40 ksps
CMT2157A	Tx+Encoder	240-960 MHz	SOP14	(G)FSK/OOK	0.5-100 ksps
CMT2180A	Tx SoC	240-480 MHz	SOP14	OOK	0.5-30 ksps
CMT2189A	Tx SoC	240-960 MHz	SOP14	(G)FSK/OOK	0.5-100 ksps
CMT2210A	Rx	300-480 MHz	QFN16 (3x3)	OOK	0.1-40 ksps

List of connected device:

```
>> USB Programmer V1 (Firmware version=008) Device:CMT2150A
```

- Configuring parameters and burning



Basic Mode | Advanced Mode | ☐ Increase XO Current | ☐ Manufacture

Configuration List

Index	Frequency	Modulation	Symbol Rate	Tx Power	Xtal Cload	PA Ramping Time	Encoder	Sync ID Length	Sync Value	Preamble	Push Button Settings
0	433.92 MHz	OOK	4.8 ksps	+13 dBm	15.00 pF	0 us	1527	20-bit	0	NA	Normal, 4 buttons
1	433.92 MHz	OOK	4.8 ksps	+13 dBm	15.00 pF	0 us	1920	32-bit	123456	None	Toggle, 5 buttons
2	433.92 MHz	OOK	9.6 ksps	+13 dBm	15.00 pF	0 us	1920	32-bit	1234567890	None	Matrix, 5 buttons
3	433.92 MHz	OOK	2.4 ksps	+10 dBm	15.00 pF	0 us	1527	20-bit	0	NA	Normal, 4 buttons

Chip Parameters

RF Settings

Frequency (240-480): 433.92 MHz
Modulation: OOK
Tx Power: +13 dBm
Symbol Rate (0.5-40.0): 2.4 ksps
Xtal Cload: 15.00 pF
PA Ramping Time: 0 us

Encode Settings

Encoder: ○ 1920 [● 1527] ○ 2262

Sync	Address (Sync ID) ☑ Hex ☐ Dec		D0	D1	D2	D3
	Length	Value (0-FFFFF)				
32-symbol	20-bit	12345	1-bit	1-bit	1-bit	1-bit

Push Button Settings

Button Mode: Normal
On/Off Button(s): NA
Number of Button(s): 4

Study Settings

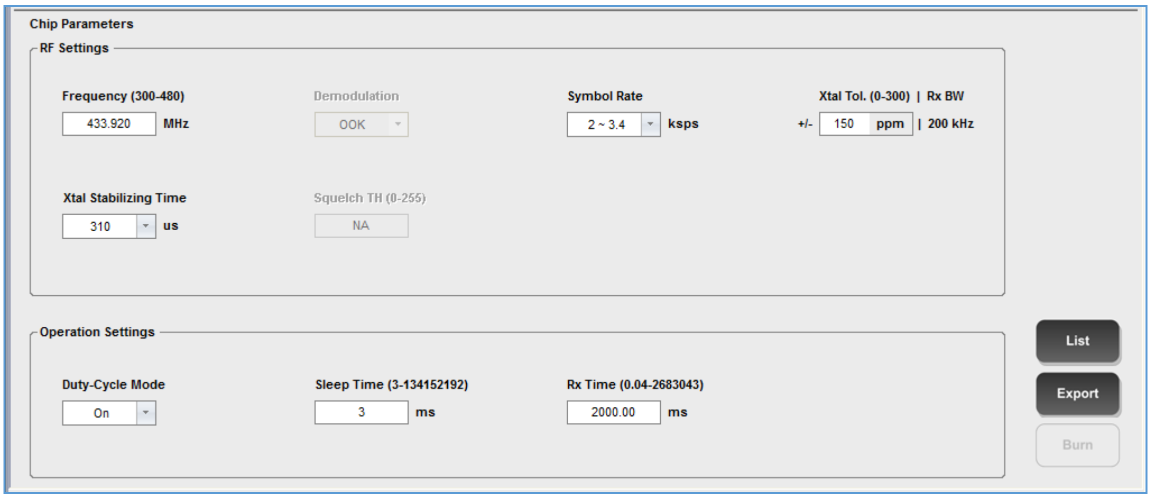
ID Study: [● Off] ○ On
Study Button: NA
Study Power: NA dBm

1. Select Basic Mode (If you use AdvanceMode, please refer to AN112 instructions)
2. Configure work frequency. Here we select 433.92MHz to match better the RFM210A default parameter to test.
3. Configure wireless rate. Here we select 2.4Ksps. RFM210A default rate is also corresponding to it.
4. Select 1527 coding format.
5. Configure ID. Here we use Hex format and configure ID for 0x12345.
6. Select the "Normal" button mode. K1~K4 is effective in the CMT2150A-EM.
7. Click“Burn”button, burn parameters.

■ Pull out CMT2150A-EM from USB Programmer and toggle the switch to “VBAT” after burning (Prior to this, please install 2 AA batteries). At this point, press any button of K1～K4, LED on the CMT2150A-EM will be lit, it indicates the key transmitting is effective.

3. CMT2210A parameter configuration and burning

➢ CMT2210A configuration details refer to CMOSTEK《AN108 CMT2210-17A Configuration Guideline》
➢ Default parameters of module RFM210A are as below:


Chip Parameters

RF Settings

Frequency (300-480): 433.920 MHz
Demodulation: OOK
Symbol Rate: 2 ~ 3.4 ksps
Xtal Tol. (0-300) | Rx BW: +/- 150 ppm | 200 kHz
Xtal Stabilizing Time: 310 us
Squelch TH (0-255): NA

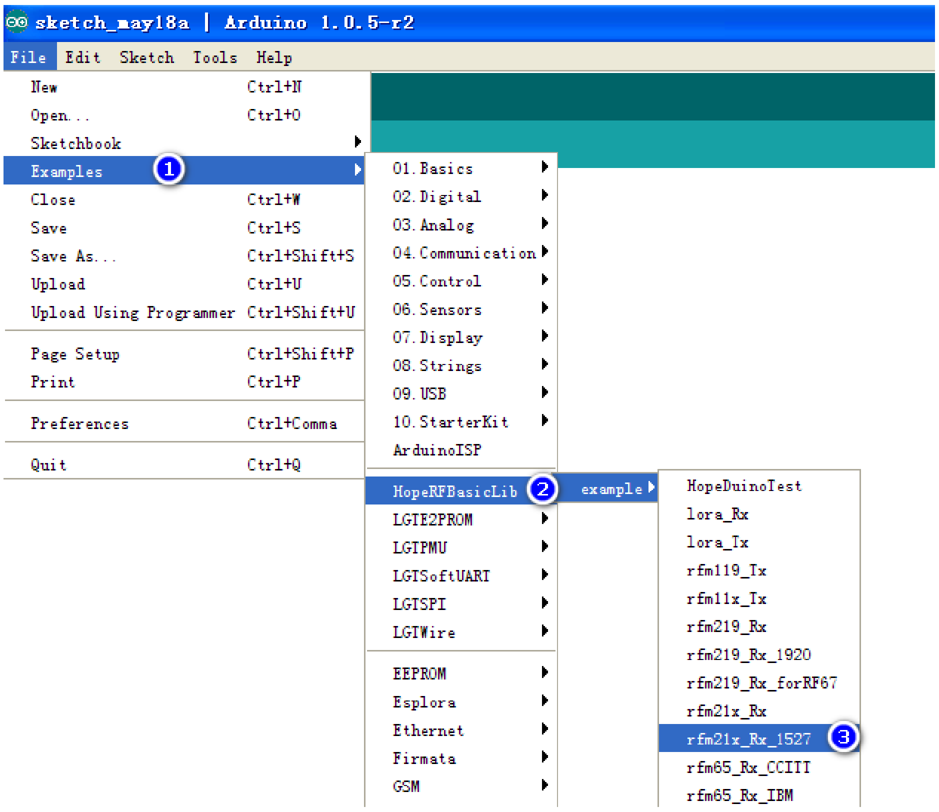
Operation Settings

Duty-Cycle Mode: On
Sleep Time (3-134152192): 3 ms
Rx Time (0.04-2683043): 2000.00 ms

4. Hands-on Experiment

- Insert module RFM210（with conversion board）into HopeDuino board
- Connect the HopeDuino boards to PC with USB cable.
- Open Arduino IDE interface, Click 【File】→【Examples】→【HopeRFBasicLib】→【example】→【rfm21x_Rx_1527】, as shown below.

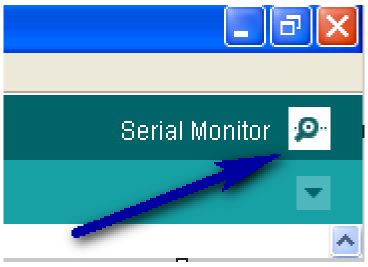
Notice: You couldn't find [HopeRFBasicLib] in [Examples] because you didn't install the HSP provided by HopeRF. Please refer to 《AN0002-HopeDuino Platform Construction Guideline》.



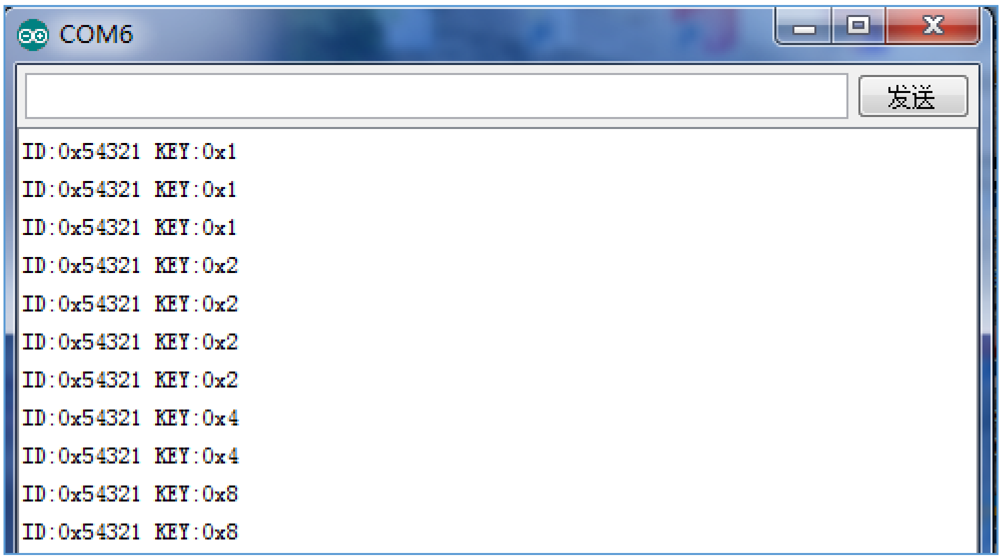
- At this time the program has been opened, please compile the download program according to the corresponding COM port.

Notice: Do not know how to compile the download code, please refer to 《AN0002-HopeDuino Platform Construction Guideline》

- After the programs are downloaded, the Tx board will transmit a packet of data by pressing any key of K1～ K4 on the CMT2150A-EM board. The Rx board will receive a packet of data through module RFM210 periodically, and parse the packet data, and upload the data to PC through UART (USB). At this point, you can set the COM of Arduino IDE as the port connected with Rx board. Open the "Serial Monitor" , as shown below.



- Click the "Serial Monitor", pop up the serial port assistant interface, as shown below. Window will display the received data message.



```
ID:0x54321 KEY:0x1
ID:0x54321 KEY:0x1
ID:0x54321 KEY:0x1
ID:0x54321 KEY:0x2
ID:0x54321 KEY:0x2
ID:0x54321 KEY:0x2
ID:0x54321 KEY:0x2
ID:0x54321 KEY:0x4
ID:0x54321 KEY:0x4
ID:0x54321 KEY:0x8
ID:0x54321 KEY:0x8
```

Notice:

1. The receiving program enables UART library function. On the description of library function UART, please refer to the "HopeDuino_UART" library file. It is also stored in the HopeRFLib.
2. ID is 0x12345 configured by RFPDK. On the receiving side the order is reversed, the ID is 0x54321.

5. Program Explanation

- rfm21x_Rx_1527.ino Case explanation

```
#include <HopeDuino_CMT221xA.h>    // Call the corresponding library file.
                                       //Calling UART is added because of using UART.
#include <Hopeduino_UART.h>
cmt221xaClass radio;                   // Define variable radio for CMT211x
uartClass uart;                    // Define variable uart for UART
byte getstr[3];                    // Define 527 coding pending data buffer
byte sendstr[32];                        // Define serial port message data buffer
byte length;                             // Define the serial port message length

void setup()
{
radio.Chipset          = CMT2210A;    // Define chip as CMT2210A
radio.SymbolTime       = 416;         // Symbol time is 2.4Ksps (that is 416us)
radio.Decode           = E527;            //Decoding format is 527
radio.vCMT221xAInit();                // Initialize CMT221xA.
uart.vUartInit(9600, _8N1);              //Initialize UART, parameters are 9600 baud rate and 8N1 format.
}

void loop()
{
byte i;
if(radio.bGetMessage(getstr, 3)!=0)      //Check radio whether to receive data function,
                                         //analyze data received.
    {
    for(i=0; i<3; i++)                   // Reverse the received 527 data
        getstr[i] = bInverseByte(getstr[i]);
    vAnalysisMsg(getstr, sendstr);       //Analyze 527 message, set up the message
    uart.vUartPutNByte(sendstr, length); //Send the message from serial port
    uart.vUartNewLine();                // UART newline is easy to read.
    for(i=0; i<3; i++)                   //Clear buffer
        getstr[i] = 0x00;
    }
}

void vAnalysisMsg(byte inptr[], byte outptr[])           // Analyze 527 message, set up the message
{
byte i, j;
i = 0;
outptr[i++] = 'I';
outptr[i++] = 'D';
outptr[i++] = ':';
outptr[i++] = '0';
outptr[i++] = 'x';
for(j=0; j<2; j++)
    {
    outptr[i++] = bChangeToAscii(inptr[j]);
    outptr[i++] = bChangeToAscii((inptr[j]>>4));
    }
outptr[i++] = bChangeToAscii(inptr[j]);
outptr[i++] = ' ';
outptr[i++] = 'K';
outptr[i++] = 'E';
outptr[i++] = 'Y';
outptr[i++] = ':';
outptr[i++] = '0';
outptr[i++] = 'x';
outptr[i++] = bChangeToAscii((inptr[j]>>4));
length = i;
}

byte bInverseByte(byte inver)          //Inverse input data, that is Bit0 and Bit7 exchange,
                                          //Bit1 and Bit6 exchange, and so on.
{
byte i, j, z;
z = 0;
j = 0x80;
for(i=0x01;i!=0;i<<=1)
    {
    if(inver&i)
        z |= j;
    j >>= 1;
    }
return(z);
}

byte bChangeToAscii(byte ch)                     //Change character into ASCII code
{
ch &= 0x0F;
if(ch<0x0A)
    ch += '0';
else
    {
    ch -= 0x0A;
    ch += 'A';
    }
return(ch);
}
```

The following focuses on explaining the specific functions in the program.

- **vAnalysisMsg**

  Type: Function

  Input: inptr[ ]， pointer, the pending data entrance.

  outptr[ ], pointer, the storage entrance of completing the pending data.

  Output: None

Function: 1527 coding data is 24bit（3 Bytes） including 20bit ID and 4bit key. This function converts the received 24bit data into a specific ASCII message. Observe the data conveniently in the serial port debugging window.

- **bInverseByte**
  Type: Function
  Input: inver， unsigned char, the pending data
  Output: Inverse data value
  Function: Inverse byte high low, for example: the pending data is 0x80, the inversing data is 0x01; the pending data is 0x1D, the inversing data is 0xB8.

- **bChangeToAscii**
  Type: Function
  Input: ch， unsigned char, character to be converted
  Output: Convert ch into ASCII code
  Function: Convert ch into ASCII code and return. Call it for vAnalysisMsg in the program. The purpose is converting the received 1527 value into the corresponding ASCII code.

6. CMT221xA Library Function Description

"CMT221xA.h"and"CMT221xA.cpp"library files are stored in the Arduino IDE files\ libraries \ HopeRFLib.

- **chipsetType**
  Type: Enumeration type
  Function: Select chip model
  Contents: CMT2210A、CMT2213A、CMT2217A（Due to the reasons for the version design, CMT2219A is packaged as another library file）

- **decodeType**
  Type: Enumeration type
  Function: Select decoding format
  Contents: E527、 E201(Due to the reasons of the version design, the corresponding decoding process is only aimed at E527 and E201).

- **Chipset**
  Type: chipsetType, enumeration type
  Function: Define chip model

- **Decode**
  Type: Enumeration type, enumeration type
  Function: Select decoding format

- **SymbolTime**
  Type: unsigned int
  Function: Define rate, which is symbol time (SYM for short). The unit is microsecond (us). The setting range is

10~4000.

- **vCMT221xAInit**
  Type: Function
  Input: None
  Output: None
  Function: The initialization is suitable for module CMT2210A、CMT2213A、CMT2217A. Call it at the beginning of program. The initialization is mainly for the IO configuration of MCU not to configure the working parameters of the CMT221xA. These parameters must be burned by USB Programmer through RFPDK software.

- **bReadRssi**
  Type: Function
  Input: None
  Output: Signal intensity range is from 0 to 255, the stronger signal, the greater value.
  Function: Make a RSSI (signal strength) conversion and read the results.

- **bGetMessage**
  Type: Function
  Input: msg[ ], pointer variable, array buffer entrance (pointer) to be received.
  Pklen , unsigned char, the length of received data. The unit is byte.
  Output: Return the received data length. The unit is byte. If the result is 0, it indicates that there is no data received
  Function: Detecting whether to receive data. The function is to query the data flow mechanism and not to interrupt processing mechanism. It needs to be embedded in the loop function when the software calls it. At the same time, if there is other process function in the repeated cycle, and the process function takes up more time, it will affect the reception decoding effect. So you need to ensure that this function is the highest priority call.

7. Pin Assignment Table:

HopeDuino	MCU	CMT2210A
13	PB5	
12	PB4	
11	PB3	
10	PB2	
9	PB1	
8	PB0	
7	PD7	CSB
6	PD6	DOUT
5	PD5	SDA
4	PD4	SCL

8. Version Records:

Version	Revised Contents	Date
1.0	Initial version	2016-03-29
1.1	Add watermarks, program explanations and descriptions	2016-04-06