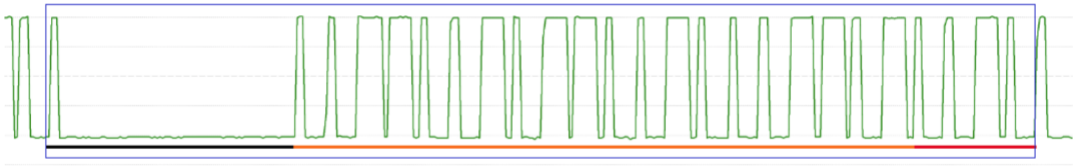
用远-R1 接收模块解码 EV1527 的方法

一、EV1527 帧结构

EV1527 每帧数据由同步码和 24 位的数据码组成，数据码又分为地址码（20 位）和按键码（4 位）。

以 433Mhz EV1527 遥控器为例，遥控波形如下。



bit0：400us 高电平+800us 低电平
bit1：1ms 高电平+200us 低电平

同步码（黑色线条部分）：高电平 400us+低电平 9ms。
地址码（橙色线条部分）：20 个数据位，共 24ms。
按键码（红色线条部分）：4 个数据位，共 4.8ms。

二、解码原理

同步码和 bit1、bit0 的低电平持续时间都不一样。通过定时器计算低电平时间来判断同步码、bit1、bit0。

三、远-R1 解码代码

设置一个 50us 中断一次的定时器，每次中断调用 soft_count()函数。
修改后面的数值就可以在其他不同的模块上用.

```
#define _start_us_min    160
#define _start_us_max    200
#define _num0_us_min     10
#define _num0_us_max     20
#define _num1_us_min     0
#define _num1_us_max     8
```

.h 文件

```
#ifndef _SOFT_DECODE_
#define _SOFT_DECODE_

#include "N76E003.h"

#define  uint   unsigned int
#define  uchar  unsigned char
```

```
#define  ulong  unsigned long

//50us
#define _start_us_min   160
#define _start_us_max   200
#define _num0_us_min    10
#define _num0_us_max    20
#define _num1_us_min    0
#define _num1_us_max    8

extern uchar IR_Key;
extern uchar Temp_addrl;
extern uchar Temp_addrh;
extern uchar Address_l;
extern uchar Address_h;

extern uint RF_Value_Cnt;
extern uint release_key;
extern bit IR_OVER;

void soft_count();
void soft_decode();

#endif
```

.c 文件

```
#include "SOFT_DECODE.h"

sbit RF_Dat = P1^7;          //接收引脚

ulong RF_data;
uchar Temp_addrl,Temp_addrh,Address_l,Address_h;//地址码
uchar cntint,IR_Key;                  //接收位数据的个数,数据
bit   start_flag=0,IR_OVER; //检测到码头 start_flag 置 1,反之置 0，接收完一组数据后 IR_OVER 置 1，反之清 0
bit   Jump_flag;                        //电平跳变标志
uint  release_key;                    //松手计数
uint  Low;                                  //低电平计数
uint  RF_Value_Cnt;                   //长按计数

/**-------------------------------------------------------------------------
-----------------**
```

```
 **函数名   ：低电平脉宽测量函数
 **功能说明：计算低电平时间(LOW),50us 调用一次
 **----------------------------------------------------------------------------
-----------------**/
void soft_count()//接收码计数函数
{
    if(RF_Dat==0)      //低电平
    {
        Low++;
        if(Jump_flag)Jump_flag=0;    //Jump_flag 由 0→1 代表 低→高 跳变
    }
    else if(RF_Dat==1)     //高电平
    {
        if(!Jump_flag)
        {
            Jump_flag=1;             //Jump_flag 由 1→0 代表 高→低 跳变
            soft_decode();
            Low=0;
        }
    }
}
/**----------------------------------------------------------------------------
-----------------**
 **函数名   ：解码函数
 **----------------------------------------------------------------------------
-----------------**/
void soft_decode()//接收码处理函数
{
//  uchar i;
    if(start_flag==0)
    {
        if(( Low > _start_us_min ) && ( Low < _start_us_max ))          //同步
码
        {
            start_flag=1;
            cntint=0;//数据长度
            IR_Key=0;
            RF_data=0;
            LED=1;
        }
        else
        {
//          LED=!LED;//1khz
            RF_Value_Cnt=0;
```

```
//            long_key_flag=0;
              release_key++;
              if(release_key>500)release_key=500;
          }
      }
      else if((start_flag==1)&&(cntint<24))
      {
          if(( Low > _num0_us_min ) && ( Low < _num0_us_max ))                //
数据 0 700us
          {
              RF_data=RF_data<<1;
              cntint++;
          }
          else if(( Low > _num1_us_min ) && ( Low < _num1_us_max ))           //
数据 1  120-200us
          {
              RF_data=RF_data<<1;
              RF_data|=1;
              cntint++;
          }
          else
          {
              start_flag=0;
              cntint=0;
          }
      }

      if(cntint==24)
      {
          release_key=0;
          RF_Value_Cnt++;
          if(RF_Value_Cnt>500)RF_Value_Cnt=500;

          cntint=0;
          start_flag=0;
          IR_Key=RF_data&0x0000000f;  //取 4 位数据
          Temp_addrl=(RF_data>>4)&0x000000ff; //地址共 20 位，只取 16 位。
          Temp_addrh=(RF_data>>12)&0x000000ff;

          IR_OVER=1;
    }
}
```

————————————————————