#### 21.5 USB电路连接图和编程示例

当实际应用USB接口时，必须使用频率准确稳定的主振荡器，将8MHZ的晶体振荡器连接到芯片的OSC1和OSC2引脚上。见图21-2所示。

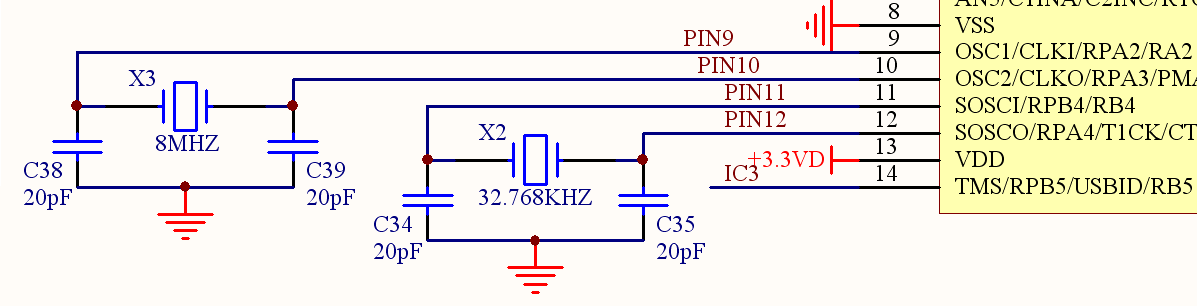


图21-2： USB必须的晶体振荡器电路

本节描述了在微芯PIC32MX220F032B型芯片上的USB\_HID程序示例。该例程实现了该单片机通过USB\_HID方式与计算机通信，示例包括单片机和PC两部分程序。

**示例功能描述**：PC与单片机之间通过USB通信，在PC端程序面板上勾选或取消“Light”复选框，来点亮便携式开发板上的LED1指示灯；在便携式开发板上按下或弹起按键1，则PC端程序面板上的“Button”复选框用勾选或取消的方式指示按键状态。

**适用范围**：本节所描述的代码适用于PIC32MX220F032B型芯片（28 引脚SOIC封装），对于其他型号或封装的芯片，未经测试，不确定其可用性。

USB引脚和USB插头电路见图21-3所示。

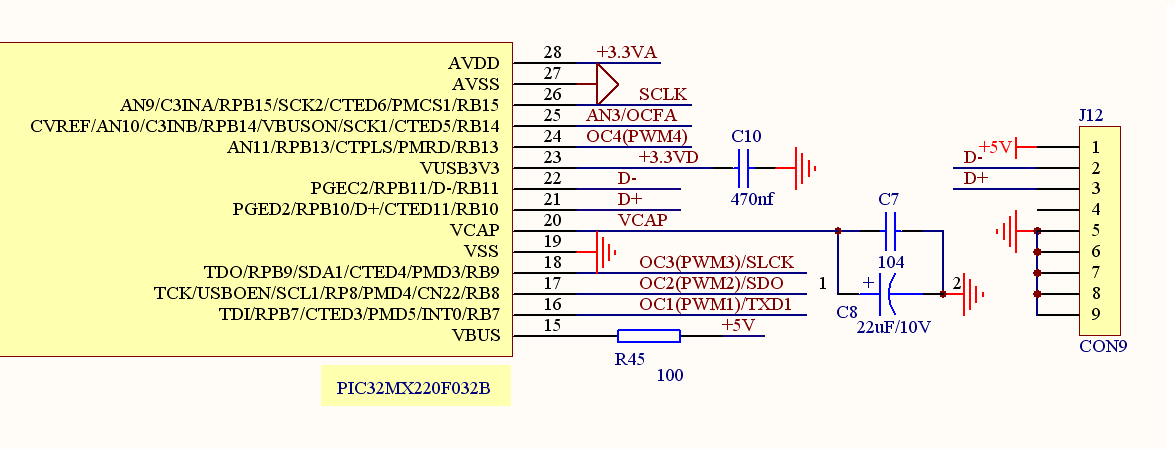


图21-3： USB引脚和USB插头电路图

本示例中用到了2个指示灯和1个按键。其中LED1（D10）用来指示PC端软件面板的“Light”状态，LED4（D4）在运行过程中以1Hz左右的频率闪烁，指示程序正在运行；按键K1的状态会在PC端软件面板上实时显示。

表21-1 USB硬件配置表

| 序号 | 功能符号 | 引脚号 | 复用端口选择指定功能所用代码 | 说明 |
| --- | --- | --- | --- | --- |
| 1 | D+ | 21 | 由USB模块指定 | USB数据+ |
| 2 | D- | 22 | 由USB模块指定 | USB数据- |
| 3 | RB7 | 16 | PORTSetPinsDigitalOut(IOPORT\_B, BIT\_7) | Light指示 |
| 4 | RB13 | 24 | PORTSetPinsDigitalOut(IOPORT\_B, BIT\_13) | 程序工作状态指示灯 |
| 5 | RA0 | 2 | ANSELAbits.ANSA0 = 0 | 按键1（K1）输入 |

指示灯及按钮输入的电路见图21-4所示。

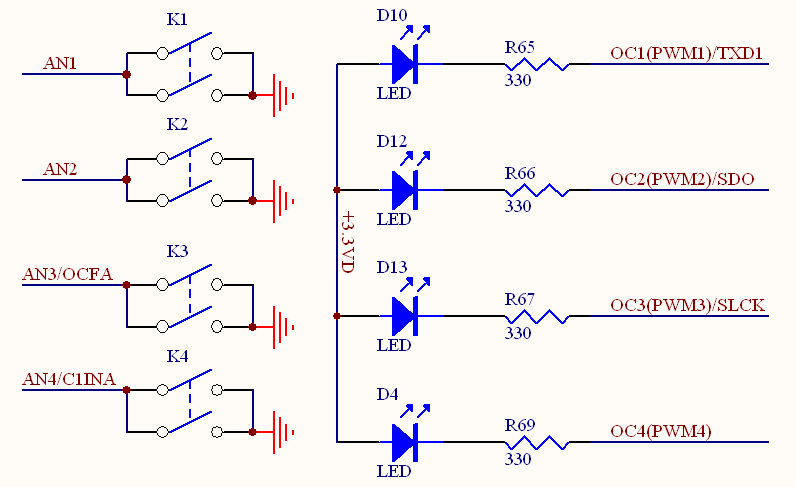


图21-4： 指示灯及按键输入电路图



图21-5：主函数流程框图

**1、主函数例程（程序流程框图见21-5所示）**

|  |
| --- |
| int **main**(void)  {  UINT pbClk;  int task=0;  // Setup configuration  pbClk = SYSTEMConfig(SYS\_FREQ, SYS\_CFG\_WAIT\_STATES | SYS\_CFG\_PCACHE);  InitLED();  INTDisableInterrupts();  INTConfigureSystem(INT\_SYSTEM\_CONFIG\_MULT\_VECTOR);  BtnInit();  Timer1Init();  TRANS\_LAYER\_Init(pbClk);    INTEnableInterrupts();  // Enter firmware upgrade mode if there is a trigger or if the application is not valid  while (1)  {  switch(task)  {  case 0:  TRANS\_LAYER\_Task();  FRAMEWORK\_FrameWorkTask();  BlinkLED();  break;  case 1:  if(btn\_flag > 0)  {  btn\_flag = 0;  ButtonScan();  }  break;  case 2:  if(update\_flag > 0)  {  update\_flag = 0;  UpdateData(Btn\_Status);  }  break;  default:  break;  }  task ++;  if(task > 2) task = 0;  }  return 0;  } |



图21-6：按键扫描函数流程框图

**2、按键扫描函数例程（程序流程框图见21-6所示）**

|  |
| --- |
| void **ButtonScan**(void)  {  static int btn1=0;  if(PORTAbits.RA0 == 0)  {  btn1 ++;  if(btn1 == BTN\_DELAY) //Button1 Pressed  {  ButtonPress(1,TRUE);  }  }  else if(btn1 > 0)  {  btn1 = 0;  ButtonPress(1,FALSE);  }  } |



图21-7：灯Light更新函数流程框图

**3、灯Light状态更新函数例程（程序流程框图见21-7所示）**

|  |
| --- |
| void **Light**(int light,BOOL on)  {  if(on)  PORTClearBits(IOPORT\_B,BIT\_7);  else  PORTSetBits(IOPORT\_B,BIT\_7);  } |

#### 附件1：代码

|  |
| --- |
| /\*  \* USB\_HID\_Example.c  \*/  #include "Include\GenericTypeDefs.h"  #include "Include\HardwareProfile\HardwareProfile.h"  #include "Include\USB\_HID\_Example.h"  #include "Include\FrameWork\Framework.h"  #include <stdlib.h>  #include <plib.h>  // Configuring the Device Configuration Registers  // 40Mhz Core/Periph, Pri Osc w/PLL, Write protect Boot Flash  #if defined(TRANSPORT\_LAYER\_USB) || defined(TRANSPORT\_LAYER\_USB\_HOST)  #pragma config UPLLEN = ON // USB PLL Enabled  #pragma config UPLLIDIV = DIV\_2 // USB PLL Input Divider = Divide by 2  #endif  #pragma config DEBUG = OFF // Background Debugger disabled  #pragma config FPLLMUL = MUL\_20 // PLL Multiplier: Multiply by 20  #pragma config FPLLIDIV = DIV\_2 // PLL Input Divider: Divide by 2  #pragma config FPLLODIV = DIV\_2 // PLL Output Divider: Divide by 2  #pragma config FWDTEN = OFF // WD timer: OFF  #pragma config POSCMOD = HS // Primary Oscillator Mode: High Speed xtal  #pragma config FNOSC = PRIPLL // Oscillator Selection: Primary oscillator w/ PLL  #pragma config FPBDIV = DIV\_1 // Peripheral Bus Clock: Divide by 1  #pragma config BWP = OFF // Boot write protect: OFF  #pragma config ICESEL = ICS\_PGx1 // ICE pins configured on PGx1 (PGx2 is multiplexed with USB D+ and D- pins).  #pragma config JTAGEN = OFF //JTAG disable  #define PERIOD 40000 //40000/40000000 = 0.001s = 1ms  #define BTN\_DELAY 5 //2\*1=2ms  #define BTN\_PUSH 200 //200ms  #define SWITCH\_PRESSED 0  unsigned int btn\_cnt=0,btn\_flag=0,update\_flag=1,counter=0;  UINT8 Btn\_Status = 0;  void **Timer1Init**()  {  // Timer1@1ms  OpenTimer1(T1\_ON | T1\_SOURCE\_INT | T1\_PS\_1\_1, PERIOD);  // Set up the timer interrupt with a priority of 2  INTEnable(INT\_T1, INT\_ENABLED);  INTSetVectorPriority(INT\_TIMER\_1\_VECTOR, INT\_PRIORITY\_LEVEL\_2);  INTSetVectorSubPriority(INT\_TIMER\_1\_VECTOR, INT\_SUB\_PRIORITY\_LEVEL\_0);  }  void \_\_ISR(\_TIMER\_1\_VECTOR, ipl2) **Timer1Handler**(void)  {  // Clear the interrupt flag  INTClearFlag(INT\_T1);  btn\_cnt++;  if(btn\_cnt > 5) //5ms  {  btn\_cnt = 0;  btn\_flag = 1;  counter++;  }  }  void **BtnInit**()  {  ANSELAbits.ANSA0 = 0; //Button  }  void **ButtonScan**(void)  {  static int btn1=0;  if(PORTAbits.RA0 == 0)  {  btn1 ++;  if(btn1 == BTN\_DELAY) //Button1 Pressed  {  ButtonPress(1,TRUE);  }  }  else if(btn1 > 0)  {  btn1 = 0;  ButtonPress(1,FALSE);  }  }  void **ButtonPress**(int button,BOOL press)  {  UINT8 set=0,clear=0xff;  switch(button)  {  case 1:  set = 0x01;  clear = 0xfe;  break;  default:  break;  }  if(press)  Btn\_Status |= set;  else  Btn\_Status &= clear;  update\_flag = 1;  }  void **Light**(int light,BOOL on)  {  if(on)  PORTClearBits(IOPORT\_B,BIT\_7);  else  PORTSetBits(IOPORT\_B,BIT\_7);  }  int **main**(void)  {  UINT pbClk;  int task=0;  // Setup configuration  pbClk = SYSTEMConfig(SYS\_FREQ, SYS\_CFG\_WAIT\_STATES | SYS\_CFG\_PCACHE);  InitLED();  INTDisableInterrupts();  INTConfigureSystem(INT\_SYSTEM\_CONFIG\_MULT\_VECTOR);  BtnInit();  Timer1Init();  TRANS\_LAYER\_Init(pbClk);    INTEnableInterrupts();  // Enter firmware upgrade mode if there is a trigger or if the application is not valid  while (1)  {  switch(task)  {  case 0:  TRANS\_LAYER\_Task();  FRAMEWORK\_FrameWorkTask();  BlinkLED();  break;  case 1:  if(btn\_flag > 0)  {  btn\_flag = 0;  ButtonScan();  }  break;  case 2:  if(update\_flag > 0)  {  update\_flag = 0;  UpdateData(Btn\_Status);  }  break;  default:  break;  }  task ++;  if(task > 2) task = 0;  }  return 0;  } |

#### 附件2：示例工程的使用方法

**1、打开项目并选择编程器**

在MPLAB X IDE环境下，打开USB\_HID\_Example\Firmware\MPLAB\_X\_Workspace\USB\_HID\_Btl\_StarterKit.X工程，将USB线连接到便携式开发板的编程器一端，另一端与计算机相连。

在X IDE的项目浏览窗口，右键单击USB\_HID\_Example，并选择Properties，打开项目属性。如图1所示：

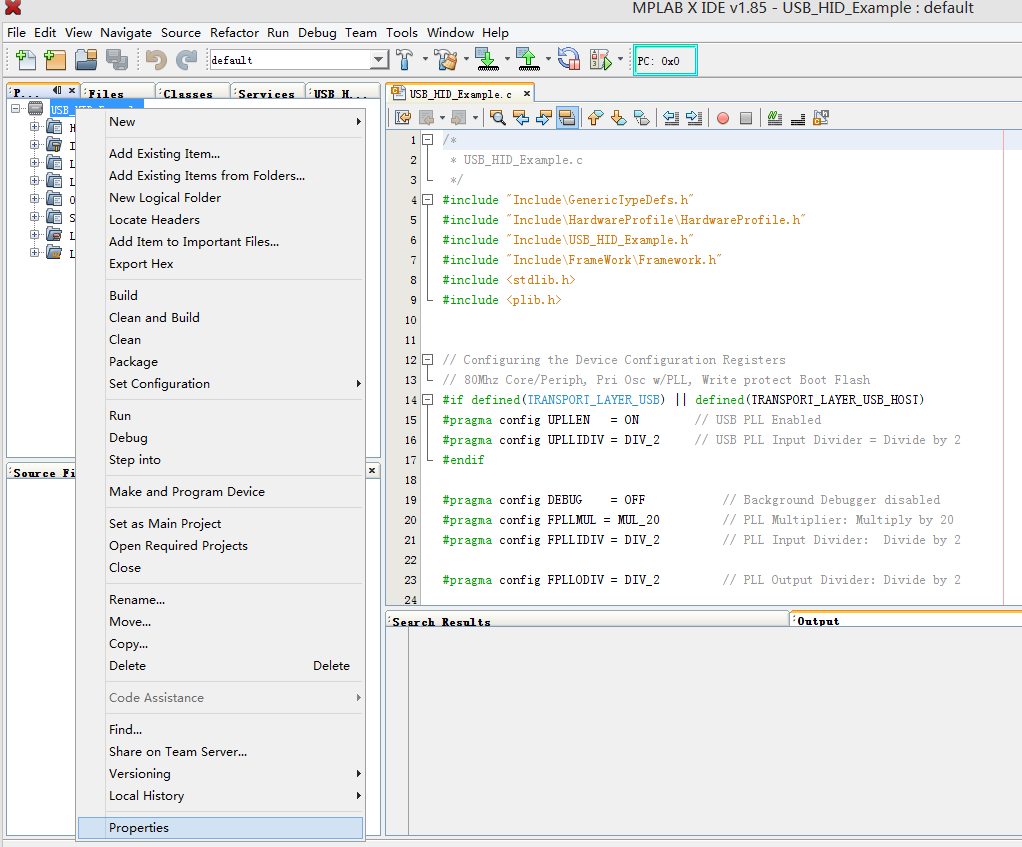


图1：在X IDE环境下打开项目属性

在项目属性界面，将编程器指定为便携式开发板，其默认路径在编程器下面的PICKit3->SN:Default\_PK3，如图2所示：

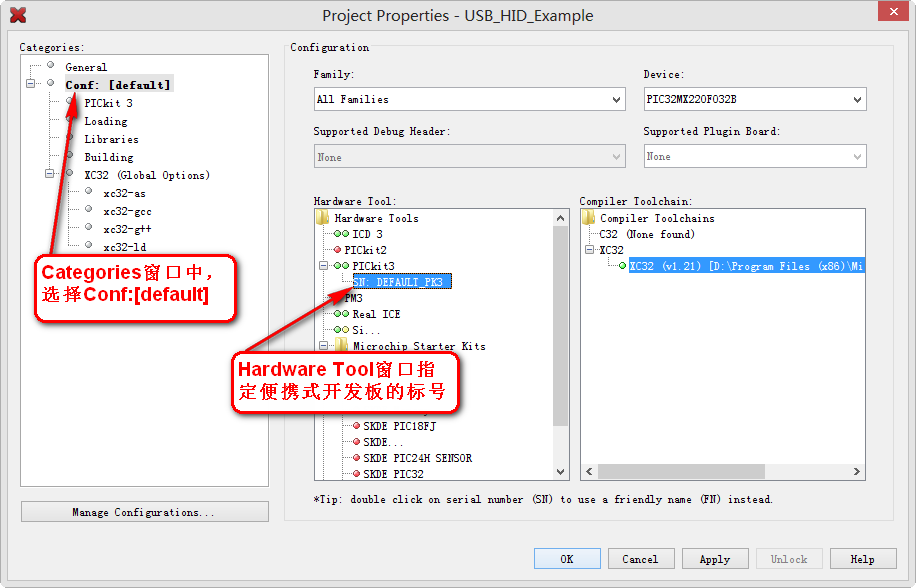


图2：在项目属性窗口指定编程器为便携式开发板

**2、编译并烧写程序代码到PIC**

要烧写代码到PIC便携开发板，必须具备以下硬件条件：

1）便携式开发板的编程端与计算机通过USB相连

2）便携式开发板上的编程跳针已连接（共5个，跨在开发板中间白色竖线上）

在MPLAB X IDE环境下，项目浏览窗口中选中USB\_HID\_Example，然后点击工具栏上的“编译并下载”按钮，开始编译并烧写代码到PIC，如图3所示：

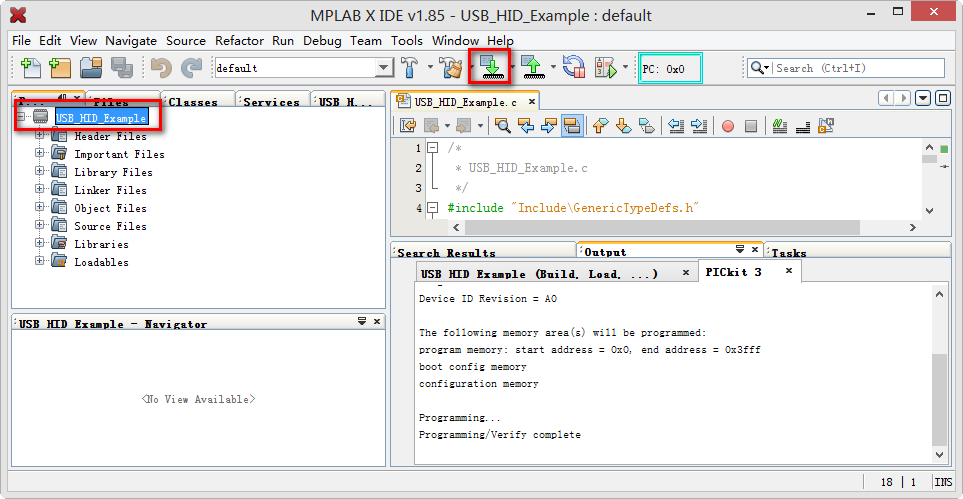


图3：编译并烧写代码

**3、通信测试**

将下载好代码的PIC便携式开发板的示例端USB口计算机连接，注意不要编程端与示例端的两个USB口同时连接到计算机上。

第一次连接时，计算机可能会弹出发现新设备对话框，选择自动安装驱动程序即可。

打开USB\_HID\_Example\PC application\USBHIDEXP.exe，初始界面如图4所示：

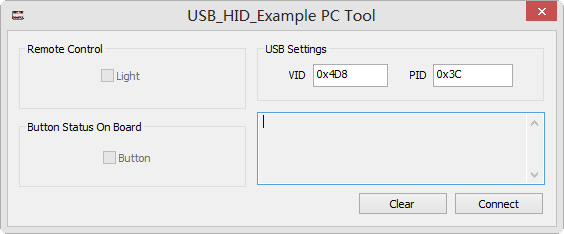


图4：PC端工具主界面

在PC工具主界面上点击“Connect”按钮连接便携式开发板。注意，便携式开发板的USB硬件ID默认为VID=0x4D8，PID=0x3C，如未改变，请在连接时保持该值。

连接成功后，在提示框里提示“Device Connected.”，图如5所示：

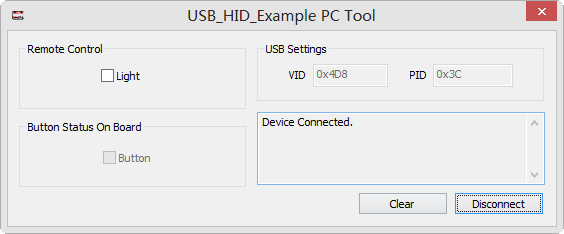


图5：连接成功

勾选或取消界面上的Light，PIC便携式开发板上的LED1会跟着点亮或熄灭。

按动便携式开发板上的按键1（K1），PC软件界面上的“Button”会勾选或取消，分别指示K1的按下与抬起动作。并在提示框中指示接收到的Button按键状态更新指令。如图6所示：

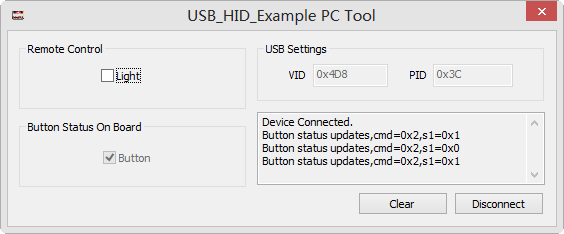


图6：通信测试界面

#### 附件3：通信协议

PC主机应用程序与PIC便携式开发板之间使用通信协议进行交互。

**帧格式**

通信协议遵循例1所示的帧格式。帧格式双向（即，从主机应用程序到便携式开发板以及从便携式开发板到主机应用程序）相同。

例1：帧格式

|  |
| --- |
| [<SOH>…]<SOH>[<DATA>…]<CRCL><CRCH><EOT>  其中：  <...> 表示一个字节  [...] 表示可选或可变数量的字节 |

帧以一个控制字符（帧头开始（Start of Header，SOH））开始，以另一个控制字符（传输结束（End of Transmission， EOT））结束。帧的完整性由两个循环冗余校验（Cyclic Redundancy Check，CRC）-16 字节保护，分别表示为CRCL（低字节）和CRCH（高字节）。

**控制字符**

数据字段中的某些字节可能与控制字符SOH 和EOT 相似。数据链路转义（Data Link Escape， DLE）字符用于转义此类会被解释为控制字符的字节。自举程序总是接受<DLE>之后的字节作为数据，并且总是在发送任何控制字符之前先发送一个<DLE>。

表1：控制字符说明

|  |  |  |
| --- | --- | --- |
| 控制 | 十六进制值 | 说明 |
| <SOH> | 0x01 | 标识帧开始 |
| <EOT> | 0x04 | 标识帧结束 |
| <DLE> | 0x10 | 数据链路转义符 |

**命令**

通信中的命令如表2所示。数据字段中的第一个字节用于承载命令。

表2：命令说明

|  |  |  |
| --- | --- | --- |
| 十六进制值 | 传输方向 | 说明 |
| 0x01 | PC🡪便携式开发板 | 更新Light状态 |
| 0x02 | 便携式开发板🡪 PC | 更新Button状态 |

**示例**

更新Light状态由PC端发送至开发板。

例2：更新Light状态示例

|  |
| --- |
| [<SOH>…]<SOH>[<0x01><STATUS>]<CRCL><CRCH><EOT>  其中：  <STATUS> ——Light状态，=0x01点亮LED1，=0x00熄灭LED1 |

更新Button状态由开发板发送至PC端。

例3：更新Button状态示例

|  |
| --- |
| [<SOH>…]<SOH>[<0x02><STATUS>]<CRCL><CRCH><EOT>  其中：  <STATUS> ——Button（K1）状态，=0x01表示K1按下，=0x00表示K1弹起 |