AT91SAM7X-EK softpack 1.5 代码解读之 USB mouse

——基于 SAM7X-EK 和 IAR EWARM

文档编号	MAN2014A_CH				
文档版本	Rev. A				
文档摘要	基于 AT91SAM7X-EK 开发板的代码解读，USB MOUSE 代码解读				
关键词	AT91SAM7X256 SAM7X-EK J-LINK IAR EWARM USB				
创建日期	2010-06-22	创建人员	Cust126	审核人员	Robin
文档类型	公开发布/开发板配套文件				
版权信息	Mcuzone 原创文档，转载请注明出处				

更新历史

版本	时间	更新	作者
Rev. A	2010-06-22	初始创建	Cust126

微控电子 乐微电子
杭州市登云路 639 号 2B143
销售 TEL：86-571-89908193 13957118045
支持 TEL：18913989166 13957118045
FAX：86-571-89908193
www.mcuzone.com www.atarm.com

1.概述

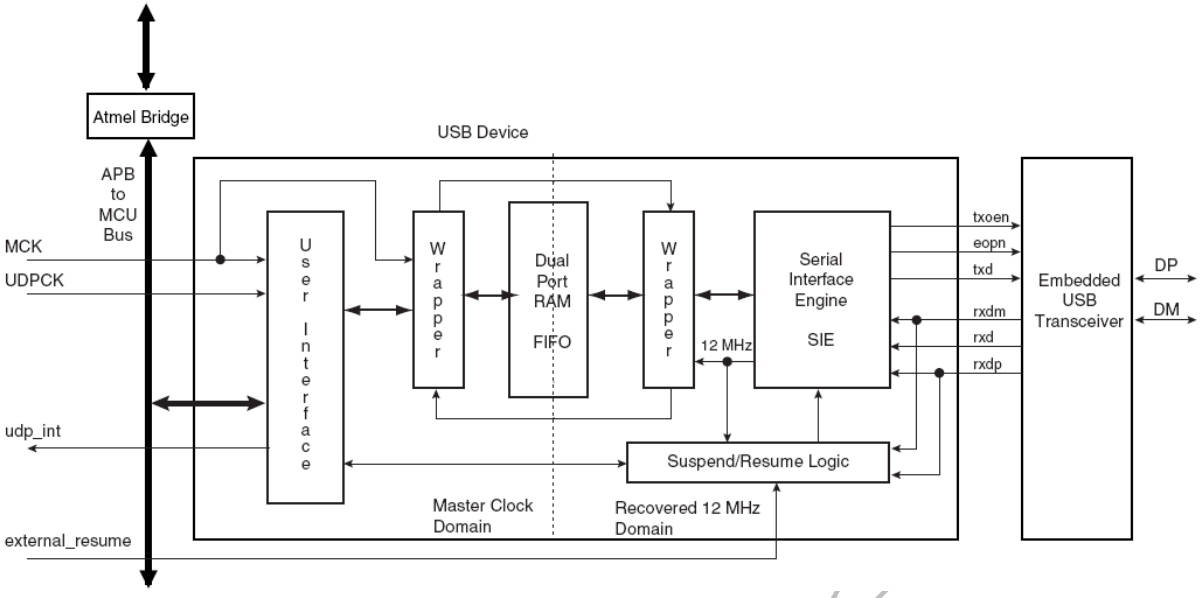
本文档以 SAM7X-EK 为硬件平台，IAR EWARM 为编译器平台，使用 J-Link 作为调试工具，演示并解读 AT91SAM7X256 的 USB MOUSE 的 device 操作流程。

2. USB 操作

2.1 USB 操作流程介绍

USB 器件端口(UDP)适用于通用串行总线(USB) V2.0 全速器件规范。每个端点可配置为 USB 传输类型中的一种。它可以联合双向 RAM 的一个或是两个 BANK 来存储当前有效的数据。若使用两个 BANK，当被另一个 USB 读写时，一个 DPR BANK 被处理器读、写。因此两个 DPR BANK 工作的端点设备可维持最大带宽(1M 字节/s)。

方框图：



2.2 USB HID MOUSE 作的目的与功能描述

AT91 softpack 1.5 的 SAM7X-EK 包内的 usb-device-hid-mouse-project 工程的功能是模拟 USB 鼠标，功能和普通的 USB 鼠标相似。

2.3 示例代码

以下代码截取自 AT91 softpack 1.5 的 SAM7X-EK 包内的 usb-device-hid-mouse-project ，基于 IAR EWARM 平台

2.3.1 main 函数代码注释解读

下面对 usb-device-hid-mouse-project 的主要代码进行注释解读，首先是 main 函数内容：

```
int main(void)
{
    unsigned char bmButtons = 0;
    signed char dX, dY;
    unsigned char isChanged;

    TRACE_CONFIGURE(DBGU_STANDARD, 115200, BOARD_MCK);
    printf("-- USB Device HID Mouse Project %s --\n\r", SOFTPACK_VERSION);
```

```
    printf("-- %s\n\r", BOARD_NAME);
    printf("-- Compiled: %s %s --\n\r", __DATE__, __TIME__);


    // If they are present, configure Vbus & Wake-up pins          //配置 PIO 引脚
    PIO_InitializeInterrupts(0);

    WAKEUP_CONFIGURE();

    // Initialize key statuses and configure push buttons          //初始化按键
    PIO_Configure(pinsJoystick, PIO_LISTSIZE(pinsJoystick));

    // HID driver initialization                                   //HID 驱动初始化
    HIDDMouseDriver_Initialize();

    // connect if needed
    VBUS_CONFIGURE();
    while (USBD_GetState() < USBD_STATE_CONFIGURED);

    // Infinite loop
    while (1) {

        isChanged = ButtonsMonitor(&bmButtons, &dX, &dY); //鼠标输入变化标志位,如果输入有变化，改变坐标值

        if (isChanged) {

            unsigned char status;

            do {

                status = HIDDMouseDriver_ChangePoints(bmButtons,
                                                      dX, dY);

            } while (status != USBD_STATUS_SUCCESS);
        }

        if( USBState == STATE_SUSPEND ) {
            TRACE_DEBUG("suspend    !\n\r");
            LowPowerMode();
            USBState = STATE_IDLE;
        }
        if( USBState == STATE_RESUME ) {
            // Return in normal MODE
```

```
                TRACE_DEBUG("resume !\n\r");
                NormalPowerMode();
                USBState = STATE_IDLE;
            }
        }
}
```

2.3.2 重要子函数代码注释解读

以下是对几个重要的子函数进行分析：

```
static unsigned char ButtonsMonitor(unsigned char *pBtnStatus,
                                    signed char *pDx,
                                    signed char *pDy)
{
    unsigned char isChanged = 0;

#if defined(PINS_JOYSTICK)
    // Left Click                                          //鼠标左键
    if (PIO_Get(&pinsJoystick[JOYSTICK_LCLIC]) == 0) {

        if ((*pBtnStatus & HIDDMouse_LEFT_BUTTON) == 0) {

            printf("LDn ");                                 //输出 LDn
            *pBtnStatus |= HIDDMouse_LEFT_BUTTON;
            isChanged = 1;
        }
    }
    else if (*pBtnStatus & HIDDMouse_LEFT_BUTTON) {

        printf("Lup ");                                     //输出 Lup
        *pBtnStatus &= ~HIDDMouse_LEFT_BUTTON;
        isChanged = 1;
    }

  #if   defined(JOYSTICK_RCLIC)
        // Right Click                                      //右键
        if (PIO_Get(&pinsJoystick[JOYSTICK_RCLIC]) == 0) {

            if ((*pBtnStatus & HIDDMouse_RIGHT_BUTTON) == 0) {

                printf("RDn ");
                *pBtnStatus |= HIDDMouse_RIGHT_BUTTON;
```

```
                isChanged = 1;
            }
        }
        else if (*pBtnStatus & HIDDMouse_RIGHT_BUTTON) {

            printf("Rup ");
            *pBtnStatus &= ~HIDDMouse_RIGHT_BUTTON;
            isChanged = 1;
        }
  #endif
#endif

        // - Movment buttons, Joystick or Push buttons
        // Left                                                    //按下 JOYSTICK_LEFT 鼠标左移
        if (PIO_Get(&pinsJoystick[JOYSTICK_LEFT]) == 0) {

            *pDx = -SPEED_X;
            isChanged = 1;
        }
        // Right                                                   //按下 JOYSTICK_RIGHT 鼠标右移
        else if (PIO_Get(&pinsJoystick[JOYSTICK_RIGHT]) == 0) {

            *pDx = SPEED_X;
            isChanged = 1;
        }
        else {
            *pDx = 0;
        }

#if defined (PINS_JOYSTICK)
        // Up                                                      //同上上移
        if (PIO_Get(&pinsJoystick[JOYSTICK_UP]) == 0) {

            *pDy = -SPEED_Y;
            isChanged = 1;
        }
        // Down                                                    //同上下移
        else if (PIO_Get(&pinsJoystick[JOYSTICK_DOWN]) == 0) {

            *pDy = SPEED_Y;
            isChanged = 1;
        }
        else {
```

```
            *pDy = 0;
        }
#endif

    return isChanged;
}
```

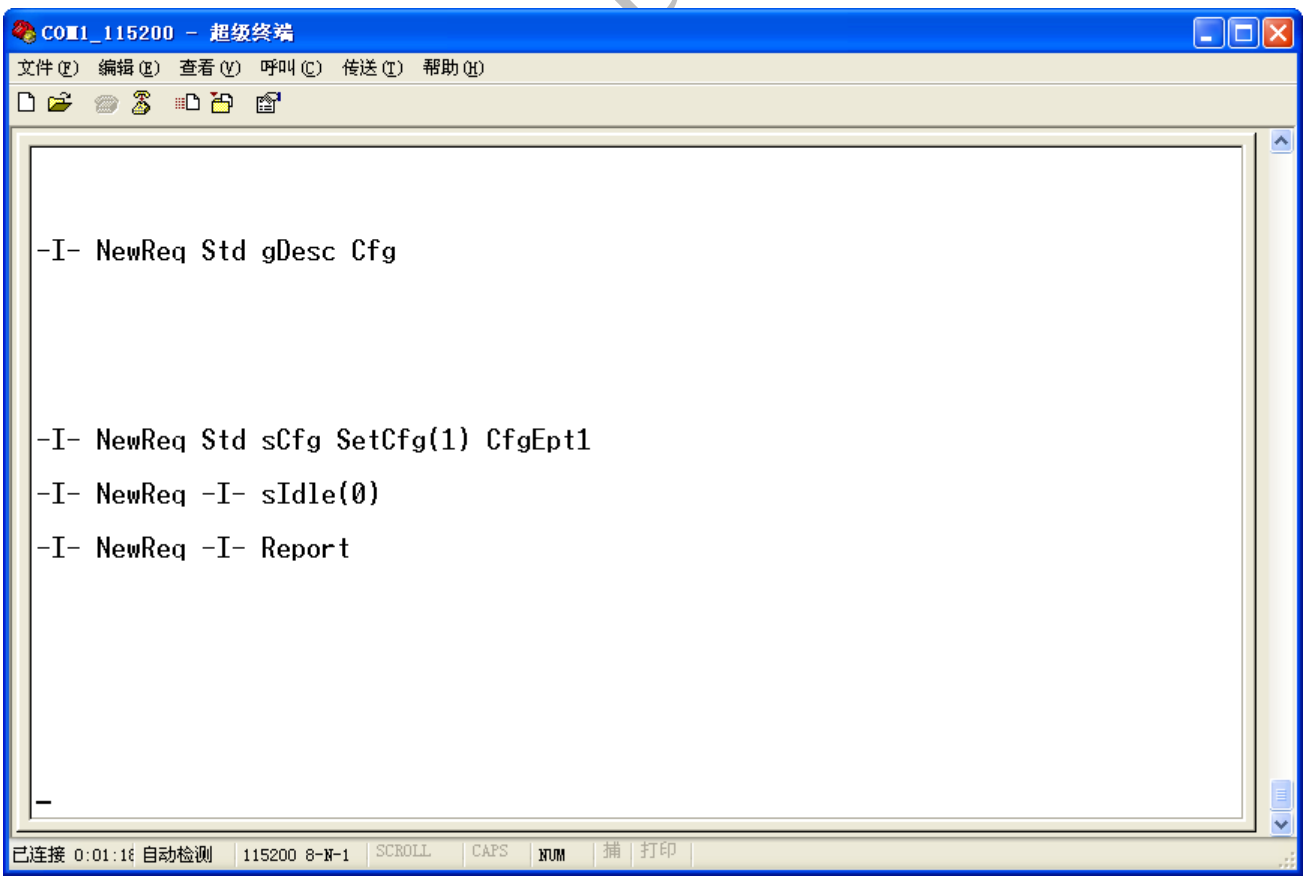
注释：

```
/// Joystick UP.                //鼠标向上移动的按键定义
#define PIN_JOYSTICK_UP      {1 << 21, AT91C_BASE_PIOA, AT91C_ID_PIOA, PIO_INPUT, PIO_PULLUP}
/// Joystick DOWN.        //鼠标向下移动的按键定义
#define PIN_JOYSTICK_DOWN    {1 << 22, AT91C_BASE_PIOA, AT91C_ID_PIOA, PIO_INPUT, PIO_PULLUP}
/// Joystick LEFT.            //鼠标向左移动的按键定义
#define PIN_JOYSTICK_LEFT    {1 << 23, AT91C_BASE_PIOA, AT91C_ID_PIOA, PIO_INPUT, PIO_PULLUP}
/// Joystick RIGHT.       //鼠标向右移动的按键定义
#define PIN_JOYSTICK_RIGHT {1 << 24, AT91C_BASE_PIOA, AT91C_ID_PIOA, PIO_INPUT, PIO_PULLUP}
/// Joystick PUSH button.    ////鼠标左键的按键定义
#define PIN_JOYSTICK_PUSH    PIN_JOYSTICK_LCLIC
```

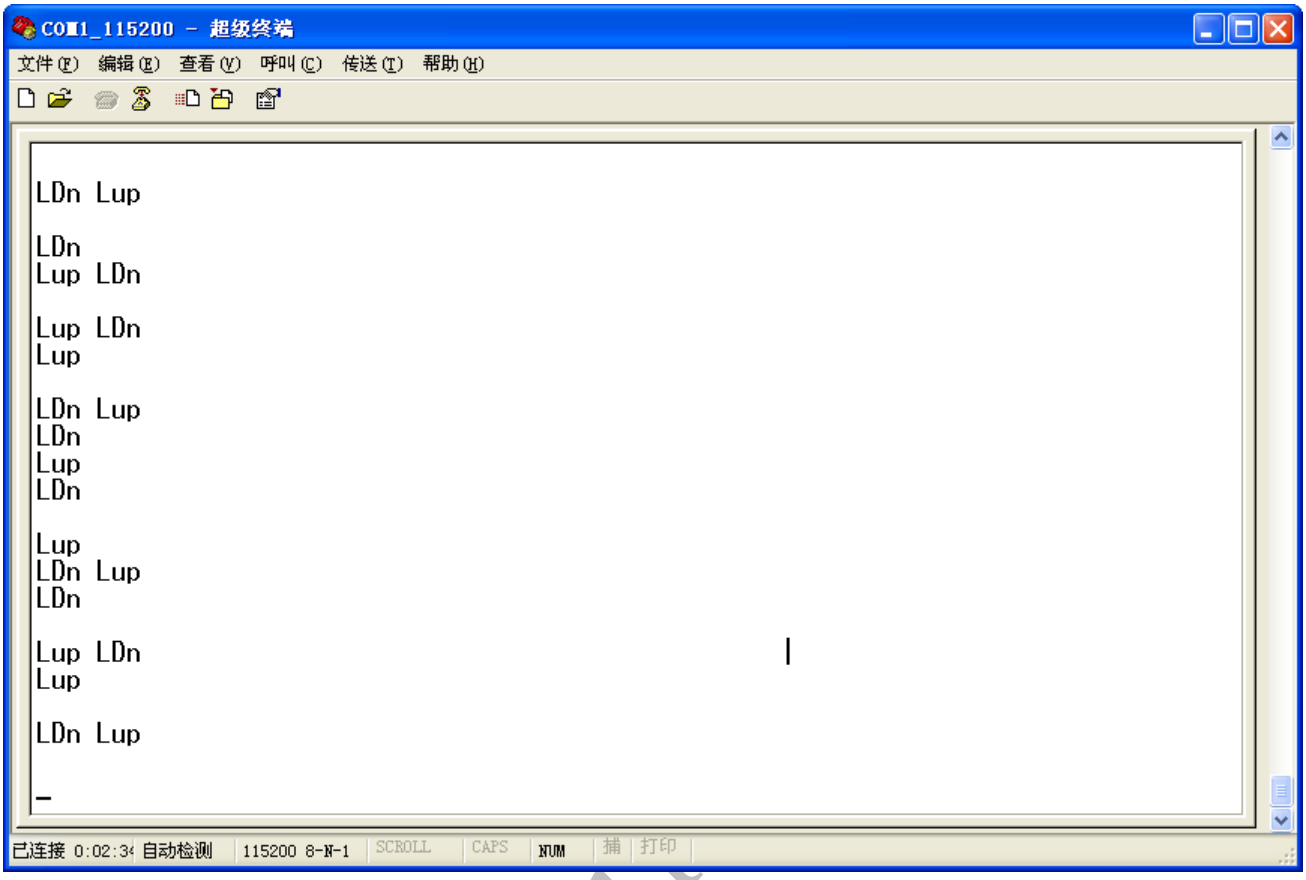
可以在此处修改引脚来切换鼠标的功能键。

2.3.3 运行结果

代码在终端上运行的结果为：



按上下左右键功能和普通的鼠标一样，按下 PUSH 键在终端上显示如下：



功能和鼠标左键一样，在设备管理器里显示鼠标信息如下(HID-compliant mouse，其中一个为标准鼠标，另一个为 SAM7X-EK 模拟的鼠标)：