4线电阻式触控屏幕控制

NuMicro® 32 位系列微控制器范例代码介绍

文件信息

代码简述	本范例代码基于NUC126平台，实现4线电阻式触控屏幕控制
BSP 版本	NUC126Series_BSP_CMSIS_v3.00.003
开发平台	NuTiny-NUC126

1 功能介绍

1.1 简介

目前市面上的电阻式触控屏幕，多分成四线、五线、七线或八线等架构，本范例程序以最常见的四线式ADC架构作说明与实现。
在触控屏幕上，屏幕左边至最右边(X方向)可视为一个长条电阻，屏幕上面至最下面(Y方向)可视为另一个长条电阻，当按压触控屏幕的时候，可由两组ADC分别采样X方向与Y方向得到两个电压值，并依照比例计算目前的触控点。
举例来说，触控屏幕分辨率为320x240，由于NUC126的ADC有12-bit的分辨率，因此5V可以被切成4096个等分。但电阻式屏幕并不是理想的，在屏幕最左边X坐标应为0的位置，X坐标的ADC读数可能会是500多而不是0；屏幕最右边X坐标应为320的位置，X坐标读数可能仅有3200而不是4095，因此必须将范围做正规化处理。
举个例子，假如触控后，ADC采样回来的数值为(2048, 1024)，请问对应320*240屏幕上的哪个坐标点？

1. 先将 ADC 范围做正规化处理：
   设ADC读数500时为X坐标0点，ADC读数为3200时为X坐标320点，因此能够将320个坐标点切成3200-500 = 2700个等分。
2. 计算 X 的坐标位置
   320: 2700 = X 坐标 : 2048 => X 坐标 = 177.124
3. Y 坐标位置同 X 坐标的计算方式

1.2 执行结果


串口配置
端口: COM69
波特率: 115200
數據位: 8
停止位: 1
校驗: NONE
關閉串口
線路控制
DTR
BREAK
RTS
線路狀態（只讀）
CTS
DSR
RING
RLSD
接收區：已接收3449位元組，速度0位元組/秒，接收狀態[允許]，輸出文本狀態[已停止]
(0, 16)
(1, 17)
(6, 20)
(13, 26)
(19, 32)
(25, 39)
(33, 48)
(40, 57)
(47, 64)
(53, 72)
(60, 79)
(67, 88)
(74, 95)
(80, 103)
(84, 109)
(90, 114)
(98, 121)
(105, 128)
(112, 134)
(118, 142)
(124, 147)
(127, 153)

2 代码介绍

2.1 Main 函数

在Main函数中，将会做系统初始化、UART初始化以及触控屏幕的初始化。然后在while循环中每秒回报一次触控点。

```
/*---------------------------------------------------------------------------*/
/*  Main Function                                                            */
/*---------------------------------------------------------------------------*/
int32_t main(void)
{
    uint32_t u32TouchX, u32TouchY;
    SYS_Init();
    UART0_Init();

    // Init Touch Panel HW
    Drv_NuTFT_HWInit();

    // Touch Test
    while (1)
    {
        if (App_GetTouchPoint(&u32TouchX, &u32TouchY) == Get_Success)
            printf("(%d, %d) \n", u32TouchX, u32TouchY);
        CLK_SysTickDelay(1000000);
    }
}
```

2.2 获得 X 轴方向的触控点

由于这是电阻式的触控屏幕，当感测X轴方向的触控点时，需要将Y方向的两条线设置为输入模式，并将X方向的两条线(XR, XL)设置为输出模式。令XR输出高逻辑、XL输出低逻辑，并使用ADC采样YU在线的电压，即可获得X方向上的触控坐标。

```
static uint16_t _Api_Get_TP_X(void)
{
    uint32_t u32_ADC_Convert_Result_X_Axis;
    uint16_t u16_nNormalizationFactor;

    /* Init ADC for TP */
    /* Set input mode as single-end and enable the A/D converter */
    ADC_Open(ADC, ADC_ADCR_DIFFEN_SINGLE_END, ADC_ADCR_ADMD_SINGLE, BIT6);
```

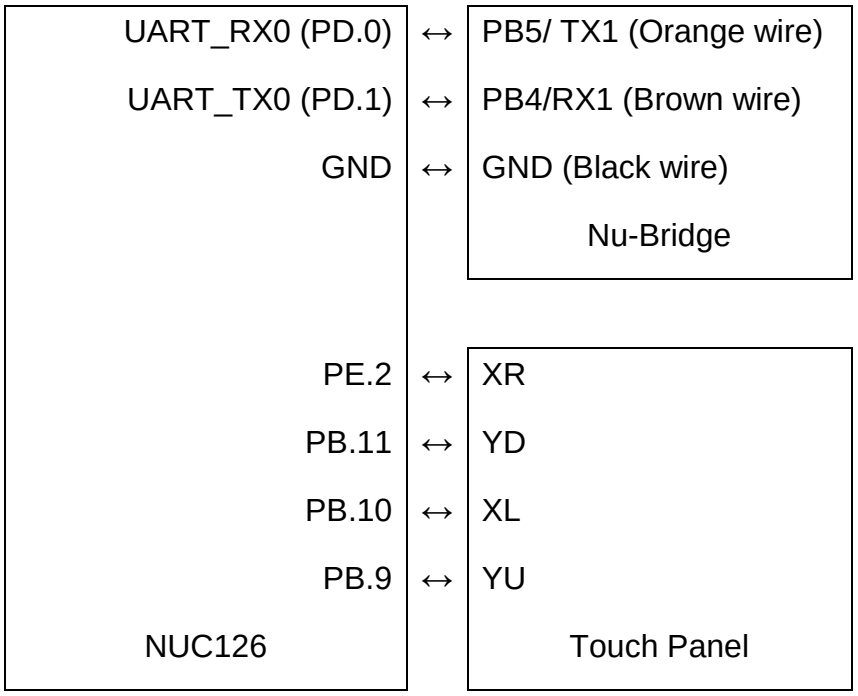
```
    /* Get X from ADC input */
    GPIO_SetMode(PE, BIT2, GPIO_MODE_OUTPUT);   // XR
    GPIO_SetMode(PB, BIT11, GPIO_MODE_INPUT);   // YD
    GPIO_SetMode(PB, BIT10, GPIO_MODE_OUTPUT);  // XL
    PE2 = 1; //XR High
    PB10 = 0; //XL Low

    /* Configure the GPB9 ADC analog input pins.  */
    SYS->GPE_MFPL &= ~(SYS_GPE_MFPL_PE2MFP_Msk);    // Disable ADC CH9
    SYS->GPB_MFPH &= ~(SYS_GPB_MFPH_PB9MFP_Msk);    // Enable ADC CH6
    SYS->GPB_MFPH |= SYS_GPB_MFPH_PB9MFP_ADC0_CH6;  //YU sample

    /* Disable the GPB8 digital input path to avoid the leakage current. */
    GPIO_DISABLE_DIGITAL_PATH(PB, BIT9);            //YU

    /* Power on ADC module */
    ADC_POWER_ON(ADC);

    /* Enable the sample module 1 interrupt.  */
    ADC_EnableInt(ADC, ADC_ADF_INT); //Enable sample module A/D ADINT1 interrupt.
    NVIC_EnableIRQ(ADC_IRQn);
    /* Clear the A/D ADINT1 interrupt flag for safe */
    ADC_CLR_INT_FLAG(ADC, ADC_ADF_INT);

    /* Reset the ADC interrupt indicator and trigger sample module 1 to start A/D
conversion */
    g_u32AdcIntFlag_TP = 0;
    ADC_START_CONV(ADC);

    /* Wait ADC interrupt (g_u32AdcIntFlag_TP will be set at IRQ_Handler function) */
    while (g_u32AdcIntFlag_TP == 0);

    /* Get x-axis ADC convert result */
    u32_ADC_Convert_Result_X_Axis = ADC_GET_CONVERSION_DATA(ADC, 6);

    /* Close ADC module */
    ADC_Close(ADC);

    //Normalization
    if (u32_ADC_Convert_Result_X_Axis <= ReportThreshold)
    {
```

```
        u32_ADC_Convert_Result_X_Axis -= Init_MinBorderCaliValue_X;
        u16_nNormalizationFactor = (Init_MaxBorderCaliValue_X - Init_MinBorderCaliValue_X)
* 1000 / LCD_Resolution_X;
        u32_ADC_Convert_Result_X_Axis *= 1000;
        u32_ADC_Convert_Result_X_Axis /= u16_nNormalizationFactor;
    }
    return u32_ADC_Convert_Result_X_Axis;
}
```

3 软件与硬件环境

- 软件环境
  - BSP 版本
    - NUC126Series_BSP_CMSIS_v3.00.003
  - IDE 版本
    - Keil uVersion 5.26
- 硬件环境
  - 电路组件
    - NuTiny-NUC126
    - Nu-Bridge
    - NuTFT v1.3 Touch Panel
  - 接线图

使用者可以将 NUC126 上的 UART_RX0(PD.0), UART_TX0(PD.1)接到 Nu-Bridge 上打印 UART 的除错讯息。使用者需要先打开 PC 上的设备管理器，查看“NuBridge Virtual Com Port (COMX)”，并打开终端机，将 Baudrate 设定为 115200 接收对应的 Com Port 讯息。相关的触控屏幕、Nu-Bridge 与 NUC126 的连接方式请参考下图。


UART_RX0 (PD.0)
UART_TX0 (PD.1)
GND
NUC126
↔
↔
↔
PB5/ TX1 (Orange wire)
PB4/RX1 (Brown wire)
GND (Black wire)
Nu-Bridge
PE.2
PB.11
PB.10
PB.9
↔
↔
↔
↔
XR
YD
XL
YU
Touch Panel

4 目录信息

- EC_NUC126_4Wired_Resistive_Touch_Panel_V1.00
  - Library — Sample code header and source files
    - CMSIS — Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
    - Device — CMSIS compliant device header file
    - StdDriver — All peripheral driver header and source files
  - SampleCode
    - ExampleCode — Source file of example code

5 如何执行范例程序

1. 根据目录信息章节进入 ExampleCode/4Wired_Resistive_Touch_Panel 路径中的 KEIL 文件夹，双击 4Wired_Resistive_Touch_Panel.uvproj。
2. 进入编译模式接口
   a. 编译
   b. 下载代码至内存
   c. 进入 / 离开除错模式
3. 进入除错模式接口
   a. 执行代码

6 修订纪录

Date	Revision	Description
Jun. 24, 2019	1.00	1. 初版

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.