GPIO 軟體防彈跳

NuMicro® 32 位系列微控制器範例代碼介紹

文件資訊

代碼簡述	使用定時器週期模式，每 1 毫秒產生一次定時器中斷，來完成軟體按鈕防彈跳
BSP 版本	M031_Series_BSP_CMSIS_V3.02.000
開發平台	NuMaker-M032SE V1.3

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1 功能介紹

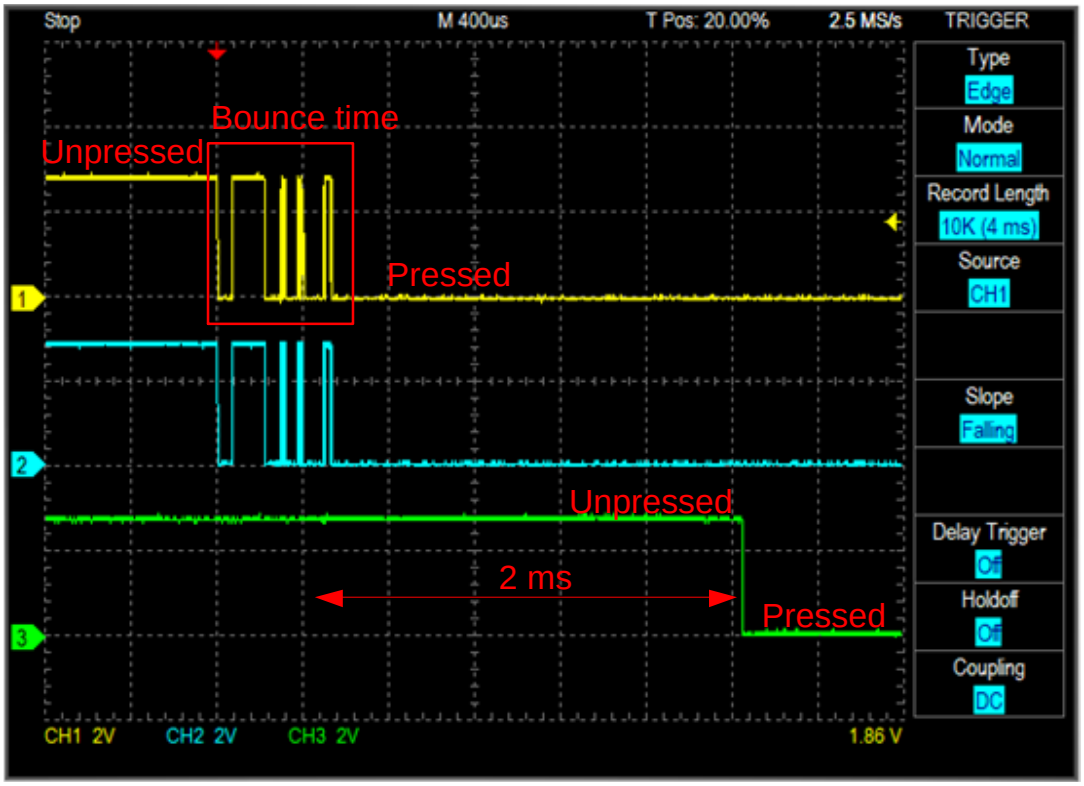
1.1 簡介

彈跳是由電子設備中觸點的閉合或斷開產生的多個脈衝信號。 最常見的是機械按鈕，當按下按鈕一次時，會產生多個脈衝信號， MCU會認為是多次按下並放開按鍵，而導致誤動作，因此需要防彈跳，以確保獲得正確的信號。 本文主要使用計時器和GPIO進行軟體防彈跳實驗。

1.2 原理

此軟體防彈跳是使用Timer每1ms產生一個中斷並將1ms旗標設置為1，然後根據此旗標每1ms確認按鈕的狀態。 在按鈕狀態一段時間未改變之後，即可以確定按鈕已被穩定地按下或放開。然後MCU可以將其判斷為正確的信號，並將該信號輸出到LED。 另外，可以根據不同按鈕的穩定時間設置防彈跳時間。

1.3 執行結果



CH1: 按鍵輸入 (PC.14),

CH2: 沒有防彈跳的訊號輸出 (PB.15),

CH3: LED, 有防彈跳的訊號輸出(PB.14)

註: 這個波型的防彈跳時間設為 2 ms.

2 代碼介紹

```
/*---------------------------------------------------------------------------*/
/* Define                                                                    */
/*---------------------------------------------------------------------------*/
#define DEBOUNCE_TIME 50            //Press and release button settling time ( ms )
#define BUTTON1 PC14                //Button input port
#define LED PB14                    //Indicator light used to indicate that a button is
                                      pressed(debounced)
#define TEST_PIN PB15               //Used to measure button state without debouncing

/*---------------------------------------------------------------------------*/
/* Global variables                                                          */
/*---------------------------------------------------------------------------*/
uint8_t bTimer_1ms;                 //Timer 1ms flag
uint8_t g_ButtonState;              //BUTTON1's current state
uint8_t g_ButtonStateOld;           //BUTTON1's previous state
uint8_t g_ButtonStateDebounced = 1; //BUTTON1's de-bounced state
uint32_t g_ButtonDebounceCnt = 0;

/*---------------------------------------------------------------------------*/
/* Functions                                                                 */
/*---------------------------------------------------------------------------*/
uint8_t GPIO_Debounce(void);

void TMR0_IRQHandler(void)
{
    /* Set 1ms flag */
    bTimer_1ms = 1;
    /* clear timer interrupt flag */
    TIMER_ClearIntFlag(TIMER0);
}

void SYS_Init(void)
{
    /* Unlock protected registers */
    SYS_UnlockReg();

    /* Enable HIRC */
    CLK_EnableXtalRC(CLK_PWRCTL_HIRCEN_Msk);
```

```
    /* Waiting for HIRC clock ready */
    CLK_WaitClockReady(CLK_STATUS_HIRCSTB_Msk);

    /* Switch HCLK clock source to HIRC */
    CLK_SetHCLK(CLK_CLKSEL0_HCLKSEL_HIRC, CLK_CLKDIV0_HCLK(1));

    /* Set both PCLK0 and PCLK1 as HCLK/2 */
    CLK->PCLKDIV = (CLK_PCLKDIV_APB0DIV_DIV2 | CLK_PCLKDIV_APB1DIV_DIV2);

    /* Switch UART0 clock source to HIRC */
    CLK_SetModuleClock(UART0_MODULE, CLK_CLKSEL1_UART0SEL_HIRC, CLK_CLKDIV0_UART0(1));

    /* Enable UART peripheral clock */
    CLK_EnableModuleClock(UART0_MODULE);

    /* Enable IP clock */
    CLK_EnableModuleClock(TMR0_MODULE);

    /* Select IP clock source */
    CLK_SetModuleClock(TMR0_MODULE, CLK_CLKSEL1_TMR0SEL_HIRC, 0);

    /* Update System Core Clock */
    /* User can use SystemCoreClockUpdate() to calculate PllClock, SystemCoreClock and
       CycylesPerUs automatically. */
    SystemCoreClockUpdate();

    /*---------------------------------------------------------------------*/
    /* Init I/O Multi-function                                             */
    /*---------------------------------------------------------------------*/
    /* Set GPB multi-function pins for UART0 RXD and TXD */
    SYS->GPB_MFPH = (SYS->GPB_MFPH & ~(SYS_GPB_MFPH_PB12MFP_Msk |
                     SYS_GPB_MFPH_PB13MFP_Msk)) |
                    (SYS_GPB_MFPH_PB12MFP_UART0_RXD | SYS_GPB_MFPH_PB13MFP_UART0_TXD);

    /* Lock protected registers */
    SYS_LockReg();
}
```

```
/*---------------------------------------------------------------------------*/
/* Init UART0                                                                */
/*---------------------------------------------------------------------------*/
void UART0_Init(void)
{
    /* Reset UART0 */
    SYS_ResetModule(UART0_RST);

    /* Configure UART0 and set UART0 baud rate */
    UART_Open(UART0, 115200);
}

int main(void)
{
    /* Init System, IP clock and multi-function I/O. */
    SYS_Init();

    /* Init UART0 for printf */
    UART0_Init();

    printf("\nThis sample code use timer to generate interrupt every 1 ms for software
              debouncing test \n");

    /* Set timer frequency to 1000 Hz */
    TIMER_Open(TIMER0, TIMER_PERIODIC_MODE, 1000);

    /* Enable timer interrupt */
    TIMER_EnableInt(TIMER0);
    NVIC_EnableIRQ(TMR0_IRQn);

    /* Start Timer 0 */
    TIMER_Start(TIMER0);

    /* Set PC4 to QUASI mode for BUTTON1 debouncing test*/
    GPIO_SetMode(PC, BIT14, GPIO_MODE_QUASI);

    /* Set PB14 and PB15 to output mode for BUTTON1's debouncing test*/
    GPIO_SetMode(PB, BIT14, GPIO_MODE_OUTPUT);//LED
    GPIO_SetMode(PB, BIT15, GPIO_MODE_OUTPUT);//TEST_PIN



    while (1)
```

```
    {
        /* Output BUTTON1 state to TESE_PIN */
        TEST_PIN = BUTTON1;

        /* Check the BUTTON1 state every 1ms */
        if (bTimer_1ms)
        {
            /* Clear 1ms flag */
            bTimer_1ms = 0;
            /* Output debounced BUTTON1 state to LED */
            LED = GPIO_Debounce();
        }
    }
}


uint8_t GPIO_Debounce(void)
{
    /* Read the BUTTON1 state */
    g_ButtonState = BUTTON1;

    /* Increase g_ButtonState if the previous state is the same as the current state */
    if (g_ButtonStateOld == g_ButtonState)
    {
        g_ButtonDebounceCnt++;

        /*The state is held longer than the button settling time,
          so this state can be treated as the correct state of the button. */
        if (g_ButtonDebounceCnt >= DEBOUNCE_TIME)
        {
            g_ButtonStateDebounced = g_ButtonState;
        }

        /* Clear the count to zero if the current state of the button is different from
           the previous state */
    }
    else
    {
        g_ButtonDebounceCnt = 0;
    }

    g_ButtonStateOld = g_ButtonState;
```

```
    /* Return the correct state of the button */
    return  g_ButtonStateDebounced;
}
```

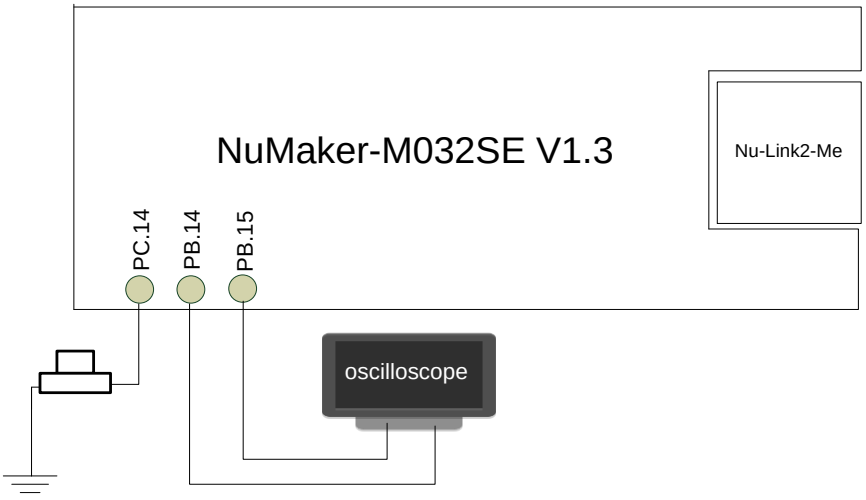
3 軟體與硬體環境

● 軟體環境

- ■ BSP 版本
  - ◆ M031_Series_BSP_CMSIS_V3.02.000
- ■ IDE 版本
  - ◆ Keil uVersion 5.28

● 硬體環境

- ■ 電路元件
  - ◆ NuMaker-M032SE V1.3
- ■ 示意圖



4 目錄資訊

- EC_M031_GPIO_Software_Debounce_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 如何執行範例程式

1. 根據目錄資訊章節進入 ExampleCode 路徑中的 KEIL 資料夾，雙擊 EC_M031_GPIO_Software_Debounce.uvproj。
2. 進入編譯模式介面
   a. 編譯
   b. 下載代碼至記憶體
   c. 進入 / 離開除錯模式
3. 進入除錯模式介面
   a. 執行代碼

6 修訂紀錄

Date	Revision	Description
Nov. 01, 2019	1.00	1. 初始發布.

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.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.