GPIO Software Debounce

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	Use the timer periodic mode to generate timer interrupt every 1ms for software button debouncing.
BSP Version	M031_Series_BSP_CMSIS_V3.02.000
Hardware	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 Function Description

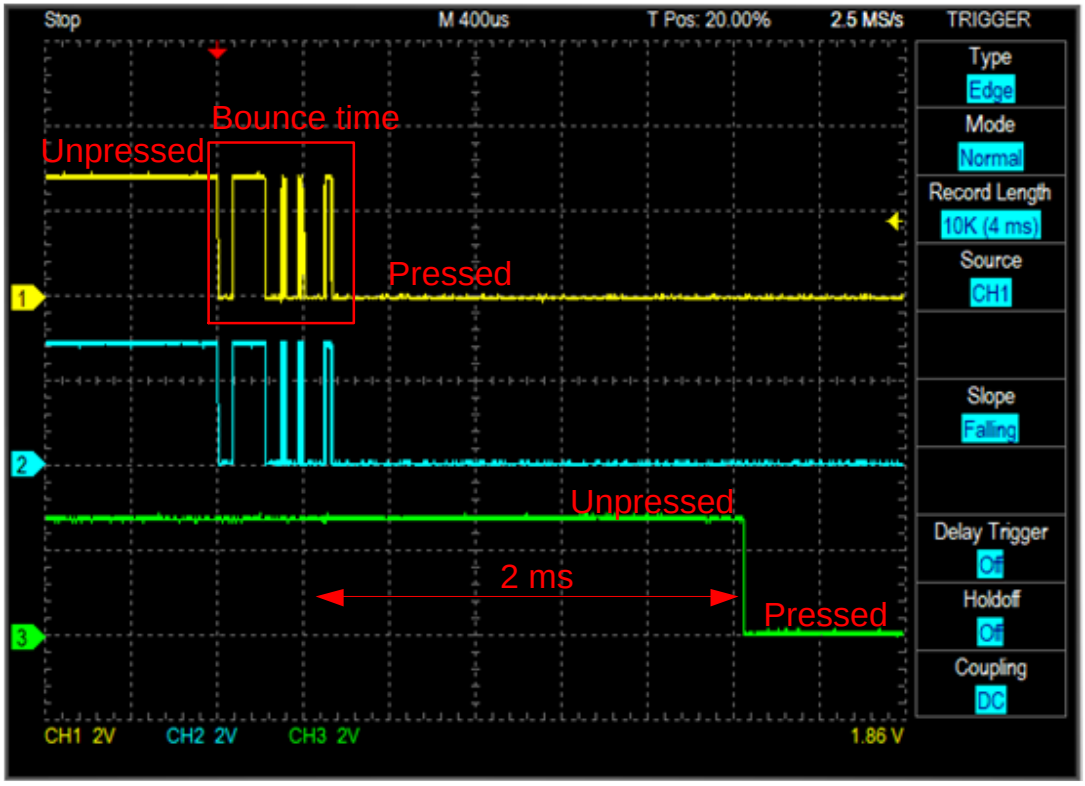
1.1 Introduction

Bouncing is a multiple of pulse signals generated by the closing or opening of contacts in an electronic device. The most common is the mechanical button. When the button is pressed once, multiple pulse signals are generated, and the MCU recognizes that it is pressed and released multiple times, causing malfunction, so it is necessary to prevent bounce to ensure that the correct signal can be obtained. This article mainly uses a timer and GPIO for software debounce experiments.

1.2 Principle

This software bounce is to use Timer to generate an interrupt every 1ms and set the 1ms flag, and then confirm the state of the button every 1ms according to this flag. After the button state has not changed for a while, it can be determined that the button has been stably pressed or released. Then MCU can judge it as the correct signal and output this signal to the LED. In addition, the debounce time can be set according to the settling time of different buttons.

1.3 Demo Result



CH1: Button input (PC.14),
CH2: Without debounce signal (PB.15),
CH3: LED, with debounce signal (PB.14)
Note: The debounce time for this demo is 2 ms.

2 Code Description

```
/*---------------------------------------------------------------------------*/
/* 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;
}
```

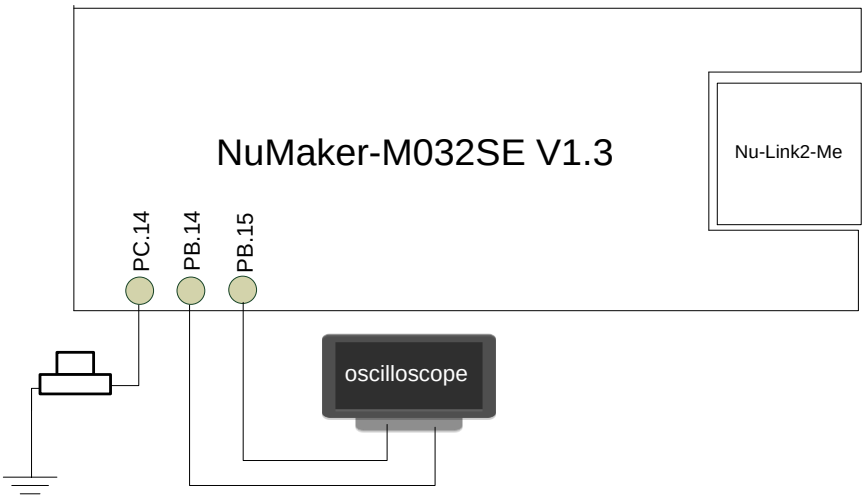
3 Software and Hardware Environment

Software Environment

- BSP version
  - M031_Series_BSP_CMSIS_V3.02.000
- IDE version
  - Keil uVersion 5.28

Hardware Environment

- Circuit components
  - NuMaker-M032SE V1.3
- Diagram



4 Directory Information

- 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 How to Execute Example Code

1. This project supports Keil uVersion 5.25 and higher version
2. Browsing into sample code folder by Directory Information (section 4) and double click EC_M031_GPIO_Software_Debounce_V1.00.uvproj
3. Enter the Keil compile mode window
    a. Build the project
    b. Download
    c. Start / Stop the debug mode
4. Enter debug mode
    a. Run

6 Revision History

Date	Revision	Description
Nov. 01, 2019	1.00	1. Initially issued.

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.