Use timer to drive RGB LED

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	The code is a demonstration of how to use timer to drive RGB LED
BSP Version	M451 Series BSP CMSIS v3.01.001
Hardware	NuEdu-SDK-M451 v1.2

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

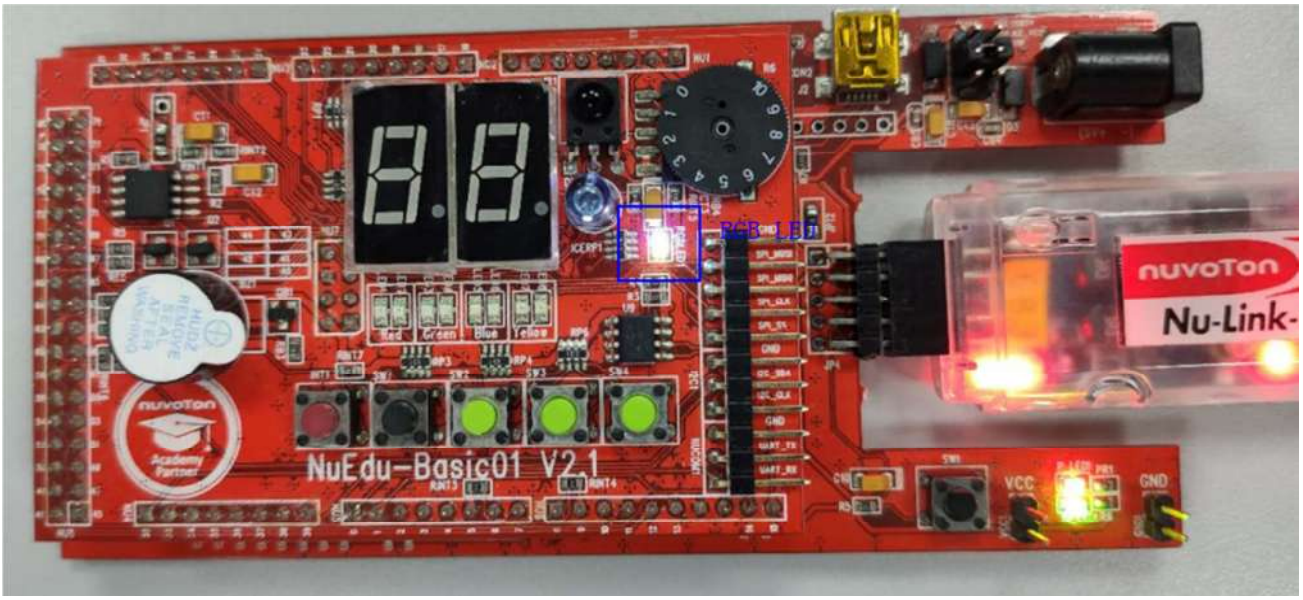
1.1 Introduction

Use Timer to control the various representations of the I/O display LED, including RGB dimming, flickering and common breathing lights, usually dimming is the way to set duty with PWM, but when there are more LEDs than PWM, it is the turn of the Timer. Only one set of Timer timing is needed to process the I/O state.

1.2 Principle

There are five parameters that can be set in the program, g_u32LED1_R, g_u32LED1_G, g_u32LED1_B are three RGB brightness settings, input value can be 0~100 corresponding to 0~100% brightness, g_u32Blink parameter is flash setting, the input value can be 0~10, the definition is to cut into 10 copies in 1 second, set how long to be bright, For example: Blink=5, that is bright for 0.5 seconds, off for 0.5 seconds for blinking, and finally g_u32brea parameter, input value is 0 or 1, set to 1 to turn on the breathing light function.

1.3 Demo Result



2 Code Description

Explain important code workflows:

```
void TMR1_IRQHandler(void)
{
    if (TIMER_GetIntFlag(TIMER1) == 1)
    {
        uint32_t u32LED_duty, u32RLED, u32BLED, u32GLED, u32LED_brea;
        /* Clear Timer1 time-out interrupt flag */
        TIMER_ClearIntFlag(TIMER1);
        g_u32LED_cnt++;
        u32LED_duty = g_u32LED_cnt % 100;
        /*1sec gradually darken, 1sec gradually brighten*/
        if ((g_u32brea == 1) && ((g_u32brea_cnt % 2) == 0))
            u32LED_brea = 100 - (g_u32LED_cnt / 100);
        else if (g_u32brea == 1)
            u32LED_brea = g_u32LED_cnt / 100;
        else
            u32LED_brea = 0;
        /*0:LED off, 1:LED on*/
        u32RLED = ((int32_t)(g_u32LED1_R - u32LED_duty - u32LED_brea) > 0) ? 1 : 0;
        u32BLED = ((int32_t)(g_u32LED1_B - u32LED_duty - u32LED_brea) > 0) ? 1 : 0;
        u32GLED = ((int32_t)(g_u32LED1_G - u32LED_duty - u32LED_brea) > 0) ? 1 : 0;
        if (g_u32LED_cnt >= (g_u32Blink * 1000))
        {
            u32RLED = 0;
            u32BLED = 0;
            u32GLED = 0;
        }
        PC->DOUT = (PC->DOUT | BIT9 | BIT10 | BIT11) & (~((u32RLED << 9) | \
                    (u32GLED << 10) | (u32BLED << 11)));
        /*1sec count*/
        if (g_u32LED_cnt == 10000)
        {
            g_u32LED_cnt = 0;
            g_u32brea_cnt++;
        }
    }
}
```

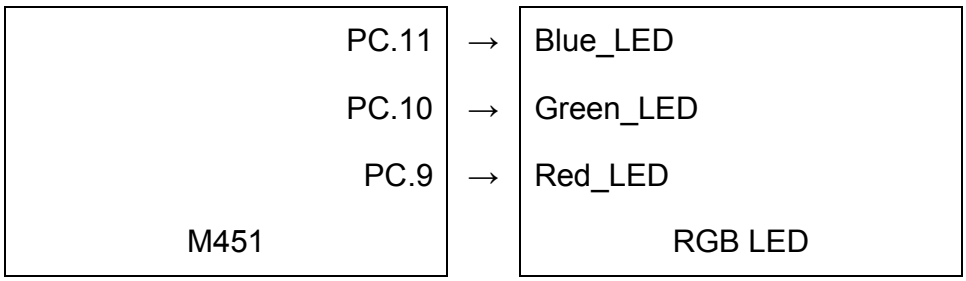
3 Software and Hardware Environment

Software Environment

- BSP version
  - M451 Series BSP CMSIS v3.01.001
- IDE version
  - Keil uVersion 5.26

Hardware Environment

- Circuit components
  - NuEdu-SDK-M451
- Diagram



4 Directory Information

- EC_M451_Drive_RGB_LED_By_Timer_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
    - NuEdu — Library for NuEdu-SDK-M451 board
  - SamlpeCode
    - ExampleCode — Source file of example code

5 How to Execute Example Code

1. Browsing into sample code folder by Directory Information (section 4) and double click M451_Drive_RGB_LED_By_Timer.uvproj.
2. Enter Keil compile mode
   a. Build
   b. Download
   c. Start/Stop debug session
3. Enter debug mode
   a. Run

6 Revision History

Date	Revision	Description
Jun. 25, 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.