Table of Contents

1. STM32WB55 PMW LED control 2
1.1. PwmLedTest.h 3
1.2. PwmLedTest.cpp 3

1. STM32WB55 PMW LED control

Our laboratory board supports an RGB LED that can be controlled with a PWM signal. Red, Green and Blue LEDs are in the same enclosure but can be controlled separately. In the stm32duino examples [1)], there is a simple program which allows us to control a single LED with PWM. It is named “All-in-one_setPWM” [2)]. It uses hardware timer output connected directly to an LED. The STM32WB55 microcontroller has two timer units built-in capable of generating PWM signals. One timer is connected to RGB LED, and another to the fan and servo. Not every pin can work as the timer output, so this example requires proper selection of the pin connected internally to the PWM timer output. On the laboratory board, the pins used are as presented in table 1.

Tab. 1: Assignment of pins

GPIO no	**LED**
D3	Red LED
D6	Green LED
D9	Blue LED

At the beginning of the code, there is a definition of pin value. Although the P-Nucleo-WB55 has the LED_BUILTIN defined it can't work as the PMW timer output. We have to override it.

```
// ********************************************************
#define LED_BUILTIN D6 //Override of the default LED_BUILTIN
// ********************************************************
#if defined(LED_BUILTIN)
#define pin  LED_BUILTIN
#else
#define pin  D2
#endif
```

In the setup part of the code, the program automatically retrieves the timer instance and PWM channel number connected to the previously defined pin.

```
  TIM_TypeDef *Instance = (TIM_TypeDef
*)pinmap_peripheral(digitalPinToPinName(pin), PinMap_PWM);
  uint32_t channel =
STM_PIN_CHANNEL(pinmap_function(digitalPinToPinName(pin), PinMap_PWM));
```

While having information on the timer instance and channel number it instantiates the HardwareTimer object.

```
 HardwareTimer *MyTim = new HardwareTimer(Instance);
```

The setup function oi finished with starting the timer to operate in PMW mode and configuring the PWM signal with frequency and duty cycle.

```
  MyTim->setPWM(channel, pin, 5, 10); // 5 Hertz, 10% dutycycle
```

It is possible to modify the duty cycle by changing the current value of the capture and compare

register for specified channel. To do it we need to use the function.

```
  MyTim->setCaptureCompare(channel, 50, PERCENT_COMPARE_FORMAT); // 50%
```

The program that controls all three LEDs has two files shown below PwmLedTest.h and PwmLedTest.cpp.

1.1. PwmLedTest.h

```
#include "Arduino.h"
#include <HardwareTimer.h>

#ifndef _PWMLED_TestLib
#define _PWMLED_TestLib

// Pins definition for RGB LED
#define LED_PWM_RED   D3  //Arduino numberring D3, STM numberring PA_10
#define LED_PWM_GREEN D6  //Arduino numberring D6, STM numberring PA_8
#define LED_PWM_BLUE  D9  //Arduino numberring D9, STM numberring PA_9

// PWM variables definitions
HardwareTimer *MyTimLED;               //Hardware Timer for PWM
uint32_t channelR, channelG, channelB; //RGB cahnnels

#endif
```

1.2. PwmLedTest.cpp

```
  #include "PwmLedTest.h"
  void PwmLedSetup()
  {
    //Create Timer instance type based on the servo pin (the same timer is
used for fan).
    TIM_TypeDef *Instance = (TIM_TypeDef
*)pinmap_peripheral(digitalPinToPinName(LED_PWM_RED), PinMap_PWM);
    //Define two separate channels for servo and fan
    channelR =
STM_PIN_CHANNEL(pinmap_function(digitalPinToPinName(LED_PWM_RED),
PinMap_PWM));
    channelG =
STM_PIN_CHANNEL(pinmap_function(digitalPinToPinName(LED_PWM_GREEN),
PinMap_PWM));
    channelB =
STM_PIN_CHANNEL(pinmap_function(digitalPinToPinName(LED_PWM_BLUE),
PinMap_PWM));

    // Instantiate HardwareTimer object. Thanks to 'new' instantiation,
HardwareTimer is not destructed when setup() function is finished.
```

```
    MyTimLED = new HardwareTimer(Instance);

    // Configure and start PWM
    MyTimLED->setPWM(channelR,LED_PWM_RED,100,1);   /* 100 Hertz, 1%
dutycycle. */
    MyTimLED->setPWM(channelG,LED_PWM_GREEN,100,1); /* 100 Hertz, 1%
dutycycle. */
    MyTimLED->setPWM(channelB,LED_PWM_BLUE,100,1);  /* 100 Hertz, 1%
dutycycle. */
  }

  void fRGBLEDSet(int red, int green, int blue)
  {
    MyTimLED->setCaptureCompare(channelR, red, PERCENT_COMPARE_FORMAT);
//modify duty cycle
    MyTimLED->setCaptureCompare(channelG, green, PERCENT_COMPARE_FORMAT);
//modify duty cycle
    MyTimLED->setCaptureCompare(channelB, blue, PERCENT_COMPARE_FORMAT);
//modify duty cycle
  }
```

1) https://github.com/stm32duino

2) https://github.com/stm32duino/STM32Examples/tree/main/examples/Peripherals/HardwareTimer/All-in-one_setPWM

From:
https://dokuwiki.zmitac.aei.polsl.pl/ - **RAU6 Dokuwiki**

Permanent link:
https://dokuwiki.zmitac.aei.polsl.pl/en:edu:zmitac:s2:aih:lab:stm32pwm?rev=1705174164

Last update: **2024/01/13 20:29**