PWM Controlled Motor

Generated by Doxygen 1.9.1

Chapter 1

PWM_Motor_Controller

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Module Documentation

4.1 Masks for configuring GPIO and ports used for PWM and clocks.

Macros

- #define **PB3_0_OUTPUT** (0x000000FF)

  Mask to set PB 3 through 0 as output.
- #define **PB7_4_OUTPUT** (0x0000FF00)

  Mask to set PB 7 through 4 as output.
- #define **GPIO_OUTPUT** (0x55555555)

  Mask for setting GPIO pins as output.
- #define **GPIO_INPUT** (0x00000000)

  Mask for setting GPIO pins as input.
- #define **GPIO_PULLUP** (0x55555555)

  Mask for setting GPIO pins with internal pull-up resistors.
- #define **GPIO_PULLDOWN** (0xAAAAAAAA)

  Mask for setting GPIO pins with internal pull-down resistors.
- #define **BIT0** (0x0001)
- #define **BIT1** (0x0002)
- #define **BIT2** (0x0004)
- #define **BIT3** (0x0008)
- #define **CK_INT** (0x00200000)

  Defualt configuration for Clock speed.
- #define **PIN_LOW** (0x00)

  Set pin to LOW state.
- #define **PIN_HIGH** (0xFF)

  Set pin to HIGH state.
- #define **PIN_PW** (0x0F)

  Set pin to period required.
- #define **AF3** (0x33333333);

  Alternate function 3 (PWM)

Functions

- void **pinSetup** (void)
- void **setupKeypadPins** (uint32_t)
- void **enableInterrupts** (void)
- uint16_t **desipherButton** (uint16_t)
- void **delay** (void)
- void **updateLEDs** (uint16_t)
- void **EXTI1_IRQHandler** (void)
- void **TIM10_PWM_setup** (uint32_t)
- void **TIM10_pinSetup** (uint8_t)
- void **PWM10_update** (uint8_t)

Variables

- uint16_t **LEDS**

  Variable to show the needed state to display number 0 - 10 in binary (ex: 2 -> 0010)
- uint16_t **colums**

  Variable to store the state of the columns when the keypad is pressed.
- uint16_t **rows**

  Variable to store the state of the rows when the keypad is pressed.
- uint16_t **output**
- uint32_t **pulseWidthLookup** [9] = {0}

  Lookup table to calculate the exact CCP register value needed to set PWM dutycycle 0 - 100%.
- uint32_t **period_cc**

  Value needed to set the PWM period requested.
- uint32_t **pulsewidth_base**

  Base number used to calculate pulseWidthLookup[] table.
- uint8_t **i**

4.1.1 Detailed Description

4.1.2 Function Documentation

4.1.2.1 delay()

```
void delay (
            void )
```

Generate a short delay.

Generate a small software delay. WARNING: some compilers optimize this function out.

Returns

None.

4.1.2.2 desipherButton()

```
uint16_t desipherButton (
            uint16_t code )
```

Take in the button pressed variable and return the required state for the LEDs.

Parameters

code	button pressed value

Returns

state required to be displayed by LEDs

Take in the row and column pressed in the keypad and return the exact key pressed. Keypad is layed out as follows:

```
1 2 3 A
4 5 6 B
7 8 9 C
* 0 # F
```

Parameters

code	16-bit number where the MSB contains the row pressed as comming from keypad. (i.e. if row 3 pressed, then MSB shows 1011)

Returns

None.

4.1.2.3 enableInterrupts()

```
void enableInterrupts (
            void  )
```

Enable all interrupts required.

Eable all interrupts needed to operate full system. It eneables external interrupt vector 1 with a falling edge and internal pull-up resitors. This is to detect when a key is pressed and pulls PA1 low.

Returns

None.

4.1.2.4 EXTI1_IRQHandler()

```
void EXTI1_IRQHandler (
            void  )
```

External interrupt vector handler. This function has the logic required to take in the key pressed.

IRQ handler for external interrupt vector 1. This function is triggered when PA1 sees a low state. It then reads the state of pins PB7-4, changes PB7-4 to output-high and PB3-0 to input and reads the input. This inidcates the value of the rows and columns when the button is pressed.

Returns

None.

4.1.2.5 pinSetup()

```
void pinSetup (
            void  )
```

Setup all pins needed to display LEDs and control the motor.

Initialize all pins needed to operate Keypad and DC motor PWM signal. This module first eneables all clocks needed to configure the GPIO ports. It then sets PA1 as an input port (used to read when any key is pressed), PC as output (display LEDs), and part of PB as output (pull-up for keypad) and the other part as input (read keypad)

Returns

None.

4.1.2.6 PWM10_update()

```
void PWM10_update (
            uint8_t pulseWidth_idx )
```

Update the state of the PWM10 module to a new duty cycle.

Parameters

pulsewidth	index 0 - 10. Indicates 0 - 100% duty cycle.

Update the duty cycle of the PWM signal out. It takes the clock counts necessary from the lookup table. If duty cycle is 0 or 100%, then it just sets pin to low or high respectivetly.

Parameters

pulseWidth_inx	Index to indicate which duty cycle to choose from lookup table

4.1.2.7 setupKeypadPins()

```
void setupKeypadPins (
            uint32_t outputPinsMask )
```

Setup all pins needed to take in input from the keypad. Provide which keys will be used as input and which will be used as output.

Parameters

outputPinsMask	pin group to be used as output. Valid inputs: PB3_0_OUTPUT or PB7_4_OUTPUT

Setup all the pins required to operate the Keypad. Pin Mask provided will be used indicate which part of PB is to serve as input and which part to serve as output.

Parameters

outputPinsMask	pins in PB to serve as output. Rest will serve as input.

Returns

None.

4.1.2.8 TIM10_pinSetup()

```
void TIM10_pinSetup (
            uint8_t state )
```

Setup timer 10 pins necessary to extract the internal PWM module signal

Parameters

state	valid valuse: PIN_LOW (duty cycle = 0%), PIN_HIGH (duty cycle = 100%), or PIN_PW (duty_cycle 10 - 90%)

Set the pin output to the required state.

Parameters

State	to set pin. Valid inputs: -PIN_LOW: Duty cycle = 0% -PIN_PW: Duty cycle = 10 - 90% -PIN_HIGH: Duty cycle = 100%

Returns

None.

4.1.2.9 TIM10_PWM_setup()

```
void TIM10_PWM_setup (
            uint32_t period )
```

Setup timer 10 to function as a PMW module with the provided period.

Parameters

period	Period to set to.

Initialize all hardware and variables required to operate TIM10 as a PWM signal output. This function takes in the desired period and calculates all values for a 0 - 100% duty cycle signal. It then initializes the module with the desired period and sets it to operate as a PWM signal.

Parameters

period	period for the PWM signal to operate in

Returns

None.

4.1.2.10 updateLEDs()

```
void updateLEDs (
            uint16_t display )
```

Update the state of the pins controlling the LEDs.

Parameters

display	value to set LEDs to

Update state of PC to show the provided LED state

Parameters

display	LED state to be displayed.

Returns

None.

Chapter 5

File Documentation

5.1 main.c File Reference

Source file to implement all functionality found in **main.h** (p. 14).

```
#include "main.h"
#include "stm32l1xx.h"
```

Functions

- int **main** ()
- void **pinSetup** ()
- void **enableInterrupts** ()
- void **setupKeypadPins** (uint32_t outputPinsMask)
- void **updateLEDs** (uint16_t display)
- uint16_t **desipherButton** (uint16_t code)
- void **TIM10_PWM_setup** (uint32_t period)
- void **TIM10_pinSetup** (uint8_t state)
- void **PWM10_update** (uint8_t pulseWidth_idx)
- void **delay** (void)
- void **EXTI1_IRQHandler** ()

5.1.1 Detailed Description

Source file to implement all functionality found in **main.h** (p. 14).

Author

Ivan Ramos `iz0006@auburn.edu`

Version

1.3

5.1.2 DESCRIPTION

This source file implements all the logic to prvide the dunctionality outline in the header file. This file is for the STM32L100 family. Port A1 is used as the input for the keypad, Port B is used to set-up the active-high state needed for the keypad, and Port C as the output for the LEDs. The LEDs display the last pressed key. Finally, timer TM10 is used to provide the PWM signal needed to control the DC-motor.

5.2 main.h File Reference

Interface for defining all variables and functions used for implementing the PWM interface to a motor controller. The duty cycle is controlled by a 16-key keypad. @description.

```
#include "stm32l1xx.h"
```

Macros

- #define **PB3_0_OUTPUT** (0x000000FF)

  Mask to set PB 3 through 0 as output.
- #define **PB7_4_OUTPUT** (0x0000FF00)

  Mask to set PB 7 through 4 as output.
- #define **GPIO_OUTPUT** (0x55555555)

  Mask for setting GPIO pins as output.
- #define **GPIO_INPUT** (0x00000000)

  Mask for setting GPIO pins as input.
- #define **GPIO_PULLUP** (0x55555555)

  Mask for setting GPIO pins with internal pull-up resistors.
- #define **GPIO_PULLDOWN** (0xAAAAAAAA)

  Mask for setting GPIO pins with internal pull-down resistors.
- #define **BIT0** (0x0001)
- #define **BIT1** (0x0002)
- #define **BIT2** (0x0004)
- #define **BIT3** (0x0008)
- #define **CK_INT** (0x00200000)

  Defualt configuration for Clock speed.
- #define **PIN_LOW** (0x00)

  Set pin to LOW state.
- #define **PIN_HIGH** (0xFF)

  Set pin to HIGH state.
- #define **PIN_PW** (0x0F)

  Set pin to period required.
- #define **AF3** (0x33333333);

  Alternate function 3 (PWM)

Functions

- void **pinSetup** (void)
- void **setupKeypadPins** (uint32_t)
- void **enableInterrupts** (void)
- uint16_t **desipherButton** (uint16_t)
- void **delay** (void)
- void **updateLEDs** (uint16_t)
- void **EXTI1_IRQHandler** (void)
- void **TIM10_PWM_setup** (uint32_t)
- void **TIM10_pinSetup** (uint8_t)
- void **PWM10_update** (uint8_t)

Variables

- uint16_t **LEDS**

  Variable to show the needed state to display number 0 - 10 in binary (ex: 2 -> 0010)
- uint16_t **colums**

  Variable to store the state of the columns when the keypad is pressed.
- uint16_t **rows**

  Variable to store the state of the rows when the keypad is pressed.
- uint16_t **output**
- uint32_t **pulseWidthLookup** [9] = {0}

  Lookup table to calculate the exact CCP register value needed to set PWM dutycycle 0 - 100%.
- uint32_t **period_cc**

  Value needed to set the PWM period requested.
- uint32_t **pulsewidth_base**

  Base number used to calculate pulseWidthLookup[] table.
- uint8_t **i**

5.2.1 Detailed Description

Interface for defining all variables and functions used for implementing the PWM interface to a motor controller. The duty cycle is controlled by a 16-key keypad. @description.

Author

Ivan Ramos `izr0006@auburn.edu`

Version

1.3

This is the file for defining all macros and functions required to interface a 16-key keypad, a PWM generator, and a DC motor. This module also comes with a way to add LEDs which displays the current speed of the motor (0 - 100%). To start, you need to call **pinSetup()** (p. 9), **setupKeypadPins()** (p. 10), TIM10_PWM_setup(_freq) where _freq is the PWM frequency, and finally **enableInterrupts()** (p. 9). Pooling buttonPressed will indicated if a button in the keypad has been pressed. The function **desipherButton()** (p. 8) will then take the variable buttonPressed to determine the state of the LEDs and dutyCycle.

Index