pied5544_443_p4

1.0

Generated by Doxygen 1.8.16

Chapter 1

File Index

1.1 File List

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

Chapter 2

File Documentation

2.1 IRlib.c File Reference

Contains functions to communicate with the SMBus IR sensor.

```
#include <plib.h>
#include "CerebotMX7cK.h"
#include <stdint.h>
#include "IRlib.h"
```

Functions

- int I2C1_IR_Read (uint8_t slave_addr, uint8_t command, uint8_t *data, int data_len)
  Reads from the IR sensor using SMBus.

2.1.1 Detailed Description

Contains functions to communicate with the SMBus IR sensor.

Author

Parker Piedmont

Date

04 Oct 2021

2.1.2 Function Documentation

2.1.2.1 I2C1_IR_Read()

```
int I2C1_IR_Read (
            uint8_t slave_addr,
            uint8_t command,
            uint8_t * data,
            int data_len )
```

Reads from the IR sensor using SMBus.

Parameters

in	*slave_addr*	Slave address of the IR sensor
in	*command*	Instructions sent to sensor
out	*data*	Data read from sensor
in	*data_len*	Number of bytes to read

Returns

Whether an error occurred

2.2 IRlib.h File Reference

Contains functions to communicate with the SMBus IR sensor.

Functions

- int I2C1_IR_Read (uint8_t slave_addr, uint8_t command, uint8_t ∗data, int data_len)
  Reads from the IR sensor using SMBus.

2.2.1 Detailed Description

Contains functions to communicate with the SMBus IR sensor.

Author

Parker Piedmont

Date

04 Oct 2021

2.2.2 Function Documentation

2.2.2.1 I2C1_IR_Read()

```
int I2C1_IR_Read (
            uint8_t slave_addr,
            uint8_t command,
            uint8_t * data,
            int data_len )
```

Reads from the IR sensor using SMBus.

Parameters

in	*slave_addr*	Slave address of the IR sensor
in	*command*	Instructions sent to sensor
out	*data*	Data read from sensor
in	*data_len*	Number of bytes to read

Returns

Whether an error occurred

2.3 LCDlib.c File Reference

Contains functions to communicate with the LCD using the PMP.

```
#include <plib.h>
#include "CerebotMX7cK.h"
#include "LCDlib.h"
```

Functions

- void LCD_init (void)

  Initializes the LCD by clearing it and setting the cursor in the top left corner.
- char readLCD (int addr)

  Reads from one of the LCD's registers.
- void writeLCD (int addr, char c)

  Writes to one of the LCD's registers.
- void LCD_putc (char c)

  Prints one character to the LCD and advances the cursor. Sets cursor to the start of the next line on \r or to the start of the current line on

  .
- void LCD_puts (char ∗char_string)

  Prints an entire string to the LCD.
- void LCD_clear (void)

  Clears the LCD and sets the cursor to the top left corner.
- void LCD_clearline (int line)

  Clears one line of the LCD and sets the cursor to the beginning of the line.
- void LCD_set_cursor_pos (int line, int pos)

  Sets the position of the cursor.
- char busyLCD (void)

  Checks whether the LCD is busy writing to a register.
- void LCD_delay (unsigned int ms)

  Waits a specified time.

2.3.1 Detailed Description

Contains functions to communicate with the LCD using the PMP.

Author

Parker Piedmont

Date

02 Nov 2020

2.3.2 Function Documentation

2.3.2.1 busyLCD()

```
char busyLCD (
             void  )
```

Checks whether the LCD is busy writing to a register.

Returns

Non-zero if busy, zero if not busy

2.3.2.2 LCD_clear()

```
void LCD_clear (
             void  )
```

Clears the LCD and sets the cursor to the top left corner.

Returns

None

2.3.2.3 LCD_clearline()

```
void LCD_clearline (
             int line )
```

Clears one line of the LCD and sets the cursor to the beginning of the line.

Parameters

in	*line*	Row to clear

Returns

None

2.3.2.4 LCD_delay()

```
void LCD_delay (
            unsigned int ms )
```

Waits a specified time.

Parameters

in	*ms*	Number of milliseconds to wait

Returns

None

2.3.2.5 LCD_init()

```
void LCD_init (
            void  )
```

Initializes the LCD by clearing it and setting the cursor in the top left corner.

Returns

None

2.3.2.6 LCD_putc()

```
void LCD_putc (
            char c )
```

Prints one character to the LCD and advances the cursor. Sets cursor to the start of the next line on \r or to the start of the current line on

.

Parameters

in	*c*	Character to print

Returns

None

2.3.2.7 LCD_puts()

```
void LCD_puts (
             char * char_string )
```

Prints an entire string to the LCD.

Parameters

in	*char_string*	String to print

Returns

None

2.3.2.8 LCD_set_cursor_pos()

```
void LCD_set_cursor_pos (
             int line,
             int pos )
```

Sets the position of the cursor.

Parameters

in	*line*	Row
in	*pos*	Column

Returns

None

2.3.2.9 readLCD()

```
char readLCD (
            int addr )
```

Reads from one of the LCD's registers.

Parameters

in	*addr*	Address of the register to read

Returns

Value of the register

2.3.2.10 writeLCD()

```
void writeLCD (
            int addr,
            char c )
```

Writes to one of the LCD's registers.

Parameters

in	*addr*	Address of the register to write
in	*c*	Value to write

Returns

None

2.4 LCDlib.h File Reference

Declares functions to communicate with the LCD using the PMP.

Functions

- void LCD_init (void)

  Initializes the LCD by clearing it and setting the cursor in the top left corner.
- char readLCD (int addr)

  Reads from one of the LCD's registers.
- void writeLCD (int addr, char c)

  Writes to one of the LCD's registers.
- void LCD_putc (char c)

Prints one character to the LCD and advances the cursor. Sets cursor to the start of the next line on \r or to the start of the current line on

.

- void LCD_puts (char *char_string)

  Prints an entire string to the LCD.
- void LCD_clear (void)

  Clears the LCD and sets the cursor to the top left corner.
- void LCD_clearline (int line)

  Clears one line of the LCD and sets the cursor to the beginning of the line.
- void LCD_set_cursor_pos (int line, int pos)

  Sets the position of the cursor.
- char busyLCD (void)

  Checks whether the LCD is busy writing to a register.
- void LCD_delay (unsigned int ms)

  Waits a specified time.

2.4.1 Detailed Description

Declares functions to communicate with the LCD using the PMP.

Author

Parker Piedmont

Date

02 Nov 2020

2.4.2 Function Documentation

2.4.2.1 busyLCD()

```
char busyLCD (
             void  )
```

Checks whether the LCD is busy writing to a register.

Returns

Non-zero if busy, zero if not busy

2.4.2.2 LCD_clear()

```
void LCD_clear (
            void  )
```

Clears the LCD and sets the cursor to the top left corner.

Returns

None

2.4.2.3 LCD_clearline()

```
void LCD_clearline (
            int line )
```

Clears one line of the LCD and sets the cursor to the beginning of the line.

Parameters

in	*line*	Row to clear

Returns

None

2.4.2.4 LCD_delay()

```
void LCD_delay (
            unsigned int ms )
```

Waits a specified time.

Parameters

in	*ms*	Number of milliseconds to wait

Returns

None

2.4.2.5 LCD_init()

```
void LCD_init (
            void  )
```

Initializes the LCD by clearing it and setting the cursor in the top left corner.

Returns

None

2.4.2.6 LCD_putc()

```
void LCD_putc (
            char c )
```

Prints one character to the LCD and advances the cursor. Sets cursor to the start of the next line on \r or to the start of the current line on
.

Parameters

in	*c*	Character to print

Returns

None

2.4.2.7 LCD_puts()

```
void LCD_puts (
            char * char_string )
```

Prints an entire string to the LCD.

Parameters

in	*char_string*	String to print

Returns

None

2.4.2.8 LCD_set_cursor_pos()

```
void LCD_set_cursor_pos (
             int line,
             int pos )
```

Sets the position of the cursor.

Parameters

in	*line*	Row
in	*pos*	Column

Returns

None

2.4.2.9 readLCD()

```
char readLCD (
             int addr )
```

Reads from one of the LCD's registers.

Parameters

in	*addr*	Address of the register to read

Returns

Value of the register

2.4.2.10 writeLCD()

```
void writeLCD (
             int addr,
             char c )
```

Writes to one of the LCD's registers.

Parameters

in	*addr*	Address of the register to write
in	*c*	Value to write

Returns

None

2.5 main.c File Reference

Main program file for ECE 443 Project 4 using FreeRTOS V202104.00.

```
#include "FreeRTOS.h"
#include "task.h"
#include "stream_buffer.h"
#include "message_buffer.h"
#include "CerebotMX7cK.h"
#include <plib.h>
#include <stdio.h>
#include <stdint.h>
#include "LCDlib.h"
#include "IRlib.h"
```

Macros

- #define **FSCK** 80000

Functions

- static void generateCNInt (void ∗pvParameters)

  Generates a virtual change notice interrupt every 6 ms by setting the CN interrupt flag.
- static void readAndSaveTemperature (void ∗pvParameters)

  Reads the temperature from an IR sensor using SMBus and sends it to a message buffer.
- static void printToLCD (void ∗pvParameters)

  Reads the temperature from a message buffer and prints it to an LCD. Temperature is passed using a string pointer.
- static void toggleLEDA (void ∗pvParameters)

  Toggles LEDA every 3 ms.
- **void** ((interrupt(ipl2), vector(_CHANGE_NOTICE_VECTOR)))
- void CN_ISR_handler (void)

  Unblocks readAndSaveTemperature when the CN interrupt is triggered.
- static void prvSetupHardware (void)

  Configures the PIC32 hardware to support the operations performed by this project.
- void PMP_init (void)

  Configures the parallel master port to communicate with the LCD.
- void cn_interrupt_initialize (void)

  Enables the CN interrupt on BTN1 at priority level 2.
- void **vApplicationMallocFailedHook** (void)
- void **vApplicationIdleHook** (void)
- void **vApplicationStackOverflowHook** (TaskHandle_t pxTask, char ∗pcTaskName)
- void **vApplicationTickHook** (void)
- void **_general_exception_handler** (unsigned long ulCause, unsigned long ulStatus)
- void **vAssertCalled** (const char ∗pcFile, unsigned long ulLine)

Variables

- const int **BRG_VAL** = ((FPB / 2 / FSCK) - 2)
- const uint8_t **SLAVE_ADDRESS** = 0x5A

2.5.1 Detailed Description

Main program file for ECE 443 Project 4 using FreeRTOS V202104.00.

Demonstrates the use of SMBus, direct task notifications, and message buffers. Simulates a change notice interrupt every 6 ms, unblocking a task that reads from an IR sensor over SMBus. Calculates the temperature and sends it through a message buffer to a task that prints it to the LCD.

Author

Parker Piedmont

Date

04 Oct 2021

2.5.2 Function Documentation

2.5.2.1 cn_interrupt_initialize()

```
void cn_interrupt_initialize (
            void  )
```

Enables the CN interrupt on BTN1 at priority level 2.

Returns

None

2.5.2.2 CN_ISR_handler()

```
void CN_ISR_handler (
            void  )
```

Unblocks readAndSaveTemperature when the CN interrupt is triggered.

Returns

None

2.5.2.3 generateCNInt()

```
static void generateCNInt (
            void * pvParameters )  [static]
```

Generates a virtual change notice interrupt every 6 ms by setting the CN interrupt flag.

Parameters

in	*pvParameters*	Unused but required by FreeRTOS

Returns

None

2.5.2.4 PMP_init()

```
void PMP_init (
            void  )
```

Configures the parallel master port to communicate with the LCD.

Returns

None

2.5.2.5 printToLCD()

```
static void printToLCD (
            void * pvParameters )  [static]
```

Reads the temperature from a message buffer and prints it to an LCD. Temperature is passed using a string pointer.

Parameters

in	*pvParameters*	Unused but required by FreeRTOS

Returns

None

2.5.2.6 prvSetupHardware()

```
static void prvSetupHardware (
            void  )  [static]
```

Configures the PIC32 hardware to support the operations performed by this project.

Returns

None

2.5.2.7 readAndSaveTemperature()

```
static void readAndSaveTemperature (
            void * pvParameters )  [static]
```

Reads the temperature from an IR sensor using SMBus and sends it to a message buffer.

Parameters

in	*pvParameters*	Unused but required by FreeRTOS

Returns

None

2.5.2.8 toggleLEDA()

```
static void toggleLEDA (
            void * pvParameters )  [static]
```

Toggles LEDA every 3 ms.

Parameters

in	*pvParameters*	Unused but required by FreeRTOS

Returns

None

Index