

pied5544\_443\_p4

1.0

Generated by Doxygen 1.8.16



<b>1 File Index</b>	<b>1</b>
1.1 File List	1
<b>2 File Documentation</b>	<b>3</b>
2.1 IRlib.c File Reference	3
2.1.1 Detailed Description	3
2.1.2 Function Documentation	3
2.1.2.1 I2C1_IR_Read()	3
2.2 IRlib.h File Reference	4
2.2.1 Detailed Description	4
2.2.2 Function Documentation	4
2.2.2.1 I2C1_IR_Read()	4
2.3 LCDlib.c File Reference	5
2.3.1 Detailed Description	6
2.3.2 Function Documentation	6
2.3.2.1 busyLCD()	6
2.3.2.2 LCD_clear()	6
2.3.2.3 LCD_clearline()	6
2.3.2.4 LCD_delay()	7
2.3.2.5 LCD_init()	7
2.3.2.6 LCD_putc()	7
2.3.2.7 LCD_puts()	8
2.3.2.8 LCD_set_cursor_pos()	8
2.3.2.9 readLCD()	9
2.3.2.10 writeLCD()	9
2.4 LCDlib.h File Reference	9
2.4.1 Detailed Description	10
2.4.2 Function Documentation	10
2.4.2.1 busyLCD()	10
2.4.2.2 LCD_clear()	11
2.4.2.3 LCD_clearline()	11
2.4.2.4 LCD_delay()	11
2.4.2.5 LCD_init()	12
2.4.2.6 LCD_putc()	12
2.4.2.7 LCD_puts()	12
2.4.2.8 LCD_set_cursor_pos()	13
2.4.2.9 readLCD()	13
2.4.2.10 writeLCD()	13
2.5 main.c File Reference	14
2.5.1 Detailed Description	15
2.5.2 Function Documentation	15
2.5.2.1 cn_interrupt_initialize()	15

2.5.2.2 CN_ISR_handler() . . . . .	15
2.5.2.3 generateCNInt() . . . . .	15
2.5.2.4 PMP_init() . . . . .	16
2.5.2.5 printToLCD() . . . . .	16
2.5.2.6 prvSetupHardware() . . . . .	16
2.5.2.7 readAndSaveTemperature() . . . . .	17
2.5.2.8 toggleLEDA() . . . . .	17

<b>Index</b>	<b>19</b>
--------------	-----------

# Chapter 1

## File Index

### 1.1 File List

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

<b>CerebotMX7cK.h</b>	??
<b>config_bits.h</b>	??
<b>FreeRTOSConfig.h</b>	??
<a href="#">IRlib.c</a>	
Contains functions to communicate with the SMBus IR sensor	3
<a href="#">IRlib.h</a>	
Contains functions to communicate with the SMBus IR sensor	4
<a href="#">LCDlib.c</a>	
Contains functions to communicate with the LCD using the PMP	5
<a href="#">LCDlib.h</a>	
Declares functions to communicate with the LCD using the PMP	9
<a href="#">main.c</a>	
Main program file for ECE 443 Project 4 using FreeRTOS V202104.00	14
<b>trcConfig.h</b>	??
<b>trcSnapshotConfig.h</b>	??



## 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	<i>slave_addr</i>	Slave address of the IR sensor
in	<i>command</i>	Instructions sent to sensor
out	<i>data</i>	Data read from sensor
in	<i>data_len</i>	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	<i>slave_addr</i>	Slave address of the IR sensor
in	<i>command</i>	Instructions sent to sensor
out	<i>data</i>	Data read from sensor
in	<i>data_len</i>	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 \n.*
- 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**

<i>in</i>	<i>line</i>	Row to clear
-----------	-------------	--------------

**Returns**

None

**2.3.2.4 LCD\_delay()**

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

Waits a specified time.

**Parameters**

<i>in</i>	<i>ms</i>	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	<i>c</i>	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	<i>char_string</i>	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	<i>line</i>	Row
in	<i>pos</i>	Column

**Returns**

None

### 2.3.2.9 readLCD()

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

Reads from one of the LCD's registers.

#### Parameters

in	<i>addr</i>	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	<i>addr</i>	Address of the register to write
in	<i>c</i>	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 \n.*

- 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	<i>line</i>	Row to clear
----	-------------	--------------

#### Returns

None

### 2.4.2.4 LCD\_delay()

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

Waits a specified time.

#### Parameters

in	<i>ms</i>	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	<i>line</i>	Row
in	<i>pos</i>	Column

#### Returns

None

### 2.4.2.9 readLCD()

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

Reads from one of the LCD's registers.

#### Parameters

in	<i>addr</i>	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	<i>addr</i>	Address of the register to write
in	<i>c</i>	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	<i>pvParameters</i>	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	<i>pvParameters</i>	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	<i>pvParameters</i>	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	<i>pvParameters</i>	Unused but required by FreeRTOS
----	---------------------	---------------------------------

#### Returns

None



# Index

- busyLCD
  - LCDlib.c, [6](#)
  - LCDlib.h, [10](#)
- cn\_interrupt\_initialize
  - main.c, [15](#)
- CN\_ISR\_handler
  - main.c, [15](#)
- generateCNInt
  - main.c, [15](#)
- I2C1\_IR\_Read
  - IRlib.c, [3](#)
  - IRlib.h, [4](#)
- IRlib.c, [3](#)
  - I2C1\_IR\_Read, [3](#)
- IRlib.h, [4](#)
  - I2C1\_IR\_Read, [4](#)
- LCD\_clear
  - LCDlib.c, [6](#)
  - LCDlib.h, [10](#)
- LCD\_clearline
  - LCDlib.c, [6](#)
  - LCDlib.h, [11](#)
- LCD\_delay
  - LCDlib.c, [7](#)
  - LCDlib.h, [11](#)
- LCD\_init
  - LCDlib.c, [7](#)
  - LCDlib.h, [11](#)
- LCD\_putc
  - LCDlib.c, [7](#)
  - LCDlib.h, [12](#)
- LCD\_puts
  - LCDlib.c, [8](#)
  - LCDlib.h, [12](#)
- LCD\_set\_cursor\_pos
  - LCDlib.c, [8](#)
  - LCDlib.h, [12](#)
- LCDlib.c, [5](#)
  - busyLCD, [6](#)
  - LCD\_clear, [6](#)
  - LCD\_clearline, [6](#)
  - LCD\_delay, [7](#)
  - LCD\_init, [7](#)
  - LCD\_putc, [7](#)
  - LCD\_puts, [8](#)
  - LCD\_set\_cursor\_pos, [8](#)
  - readLCD, [8](#)
  - writeLCD, [9](#)
- LCDlib.h, [9](#)
  - busyLCD, [10](#)
  - LCD\_clear, [10](#)
  - LCD\_clearline, [11](#)
  - LCD\_delay, [11](#)
  - LCD\_init, [11](#)
  - LCD\_putc, [12](#)
  - LCD\_puts, [12](#)
  - LCD\_set\_cursor\_pos, [12](#)
  - readLCD, [13](#)
  - writeLCD, [13](#)
- main.c, [14](#)
  - cn\_interrupt\_initialize, [15](#)
  - CN\_ISR\_handler, [15](#)
  - generateCNInt, [15](#)
  - PMP\_init, [16](#)
  - printToLCD, [16](#)
  - prvSetupHardware, [16](#)
  - readAndSaveTemperature, [16](#)
  - toggleLEDA, [17](#)
- PMP\_init
  - main.c, [16](#)
- printToLCD
  - main.c, [16](#)
- prvSetupHardware
  - main.c, [16](#)
- readAndSaveTemperature
  - main.c, [16](#)
- readLCD
  - LCDlib.c, [8](#)
  - LCDlib.h, [13](#)
- toggleLEDA
  - main.c, [17](#)
- writeLCD
  - LCDlib.c, [9](#)
  - LCDlib.h, [13](#)