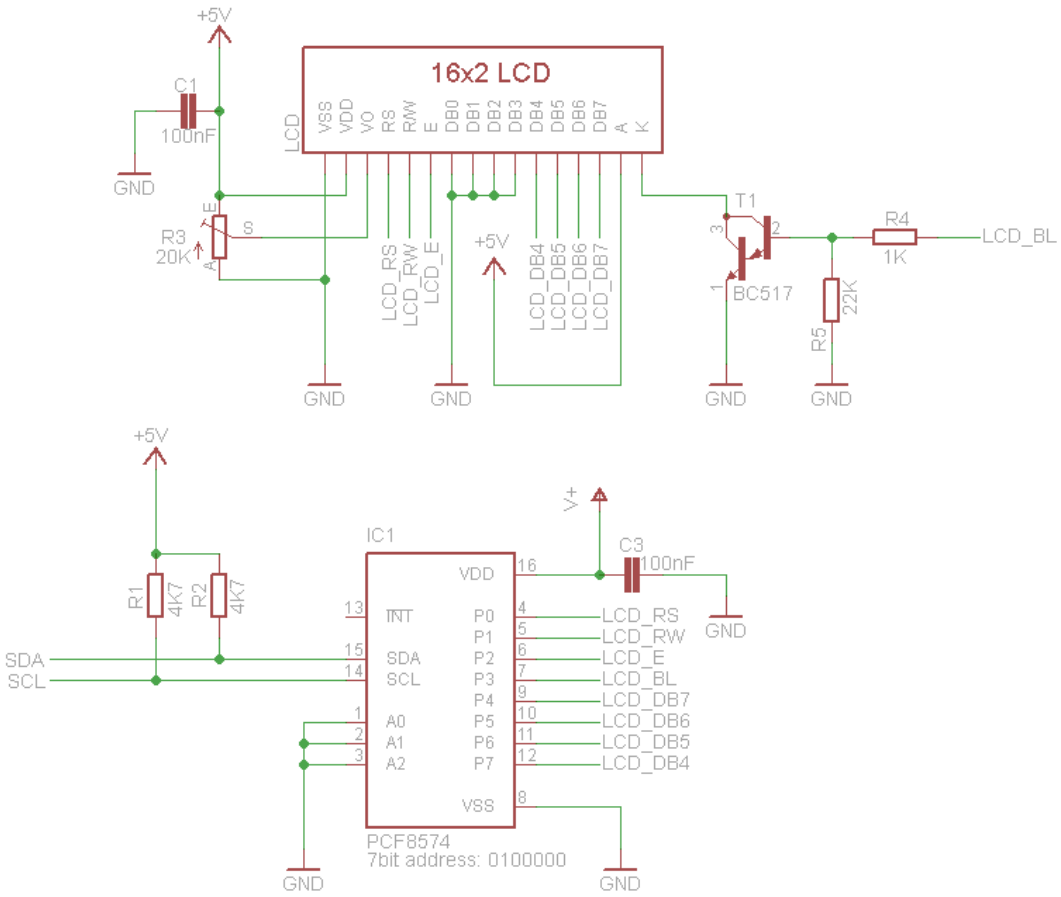
This library is used to drive an LCD with HD44780 or compatible controller through a PCF8574 IO Expander

Electrical connections



SCL and SDA must to be connected to SCL and SDA lines of ORbit16™. There are 2 I2C modules on ORbit16™. You must use the module 1 with default pins:

RB9/RP9 (BP9)	SDA1
RB8/RP8 (BP8)	SCL1

Connect SCL on BP8 and SDA on BP9. **BP8 and BP9 are 5V-tolerant inputs, you must use only this pins!** Common HD44780 LCD works at 5V and PCF8574 is used at 5V since the LCD I2C library can read the busy flag status.

SCL and SDA lines requires pull-up resistors. 4.7KΩ is a common value for 400KHz mode.

A0, A1 and A2 pins of PCF8574 are the I2C address selection pins. 7-bit address of PCF8574 is:

0	1	0	0	A2	A1	A0

if you use a PCF8574A instead of PCF8574, the 7-bit address will be:

0	1	1	1	A2	A1	A0

this 7-bit address is followed by 1 if you want to use the IO expander in read-mode, or by 0 if you want to use it in write mode.

If A2, A1 and A0 are pulled to GND as in above schematic, the 7-bit address of a PCF8574 will be 0x20 :

0	1	0	0	0	0	0

The 7-bit address of a PCF8574A will be 0x38:

0	1	1	1	0	0	0

The library uses the address in the 7bit format, the R/W bit is added by the functions.

Library settings

If you use a different IO-Expander-to-lcd connection than that showed in above schematic, you must change the following part in the **`lcd_i2c.h`** header file:

```
// connections between LCD and PCF
#define   LCD_RS    0    // Register select
#define   LCD_RW    1    // Read/Write
#define   LCD_EN    2    // Enable
#define   LCD_BL    3    // Backlight
#define   LCD_D7    4    // LCD data 7
#define   LCD_D6    5    // LCD data 6
#define   LCD_D5    6    // LCD data 5
#define   LCD_D4    7    // LCD data 4
```

You must also change the display size by editing this part in the same file:

```
// display size
#define LCD_ROWS 2
#define LCD_COLS 16
```

Library usage

This library requires the IO Expander Library. At the beginning of your program, outside the main function, in the "include" section, you must write:

```
#define IOEXP_ADDR 0x20 // required PCF8574 address definition
#include <libpic30.h> // required for delays used in the library
#include <i2c.h> // required for use of io expander
#include "ioexp.c" // required for use of LCD with PCF8574
#include "lcd_i2c.c"
```

In **`main()`** function of your program, outside the infinite loop, you must configure the I2C module. This is an example of configuration:

```
OpenI2C1(I2C_ON  &  I2C_IDLE_CON  &  I2C_SLW_EN  &  I2C_SM_DIS  &
I2C_IPMI_DIS, 37);
IdleI2C1();
```

Value 37 is used to work with devices operating at 400KHz on ORbit16™. Working at 400KHz requires the Slew Rate control enabled. This is achieved by using the **`I2C_SLW_EN`** value in the configuration.

If you want I2C module on ORbit16™ operates at 100KHz, change 37 to 157 and disable the slew rate control (**`I2C_SLW_DIS`**). If you want I2C module on ORbit16™ operates at 100KHz, change 37 to 13 and disable the slew rate control (**`I2C_SLW_DIS`**).

> ***Note***
> *Few devices can operate at 1MHz. All devices on same I2C bus must operate to the frequency allowed by the slower device on the bus. Different frequencies requires different pullup resistors values.*

then you must initialize the LCD (in the main, outside the infinite loop) by using the function **`LCDinit`**:

```
LCDinit(ON); // initialize the LCD with backlight ON
```

Now you can use the following high-level functions:

```
void LCDInit(unsigned char backlight);
void LCDGoto(char row,char col);
void LCDClear(void);
void LCDSetBacklight(unsigned char backlight);
void LCDPutch(char c);
void LCDPutun(unsigned int c);
void LCDPutsn(signed int c);
void LCDPuts(const char *s);
void LCDCustomChar(unsigned char pos, const char *b);
```

LCDInit
is used to initialize the LCD.

Parameters:
`backlight` : turns on or off the backlight at the initialization

usage example:
```
LCDInit(ON); // initialize the LCD with backlight ON
```

LCDGoto
move the cursor to the desidered row and column (1-based index)

Parameters:
`row` : row value (between 1 and LCD_ROW)
`col` : column value (between 1 and LCD_COLS)

usage example:
```
LCDGoto(1,2); // move cursor to first row, second column
```

LCDClear
Clear LCD content.

Parameters:
`none`

usage example:
```
LCDClear(); // clear LCD
```

LCDSetBackLight
turns on or off the backlight

Parameters:
`backlight` : ON or OFF (1 or 0)

usage example:
```
LCDSetBackLight(OFF); // turns off the backlight
```

LCDPutch
write a single char

Parameters:
`c` : code of the char

usage example:

```
LCDPutch('A'); // write A - note: use SINGLE quotes
LCDPutch(128); // write the char having code 128
```

> ***Note***: *for chars having a code between 32 and 127, the code would be the ascii code, for other values you must read the datasheet of the controller used by your display. Values between 0 and 7 are used for the custom characters.*

LCDPutun

write an unsigned integer

Parameters:
`c` : number between 0 and 65535

usage example:

```
LCDPutun(4532); // writes 4532
unsigned char a=256;
LCDPutun(a); // writes 256
```

LCDPutsn

write a signed integer

Parameters:
`c` : number between -32768 and 32767

usage example:

```
LCDPutsn(-4532); // writes -4532
unsigned char a=256;
LCDPutsn(a); // writes 256
a *= -1;
LCDPutsn(a); // writes -256
```

LCDPuts

write a string null terminated

Parameters:
`*s` : pointer to string

usage example:

```
LCDPuts("Hello World!"); // writes Hello World! - use DOUBLE quotes
const char string[] = "ORbit16";
LCDPuts(string); // writes ORbit16
```

LCDCustomChar
defines an user character

Parameters:
`position` : CGRAM position where to store the character, value between 0 and 7
`*b` : pointer to character values

usage example:

```
const char heart[]={0,10,31,31,14,4,0,0}; // array for custom char (an heart)
LCDCustomChar(0,heart); // defines the custom char in location 0
LCDPutch(0); // print the heart
```

For further informations on how to define custom chars you can follow this links:

http://www.settorezero.com/wordpress/corso-programmazione-picmicro-in-c-lezione-7-parte-3-interfaccia-con-lcd-come-definire-simboli-e-caratteri-personalizzati-nell-lcd-esempi-di-animazione/

This is a software can help you to paint new chars:

http://www.settorezero.com/wordpress/software/custom-char-hd44780/