```
/*****************************************************************************
* ATxmega library for the Devantech LCD03/LCD05 with 3x4 keypad - V2.0       *
* © Copyright 2012-2019 Filippo Pardini - filippo@robotica.eng.br            *
*                                                                            *
* This program is free software: you can redistribute it and/or modify it    *
* under the terms of the GNU Lesser General Public License as published by   *
* the Free Software Foundation, either version 3 of the License, or any      *
* later version.                                                             *
*                                                                            *
* This program is distributed in the hope that it will be useful,            *
* but WITHOUT ANY WARRANTY; without even the implied warranty of             *
* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the               *
* GNU Lesser General Public License for more details.                        *
*                                                                            *
* You should have received a copy of the GNU Lesser General Public License   *
* along with this program. If not, see <http://www.gnu.org/licenses/>.       *
*****************************************************************************

Remember that the TWI bus must have pull-up resistors on SDA and SCL.
As LCD03/LCD05 are 5V, if the microcontroller is 3.3V a logic level
converter must be used. I recommend 1.8K or 4.7K resistors.

The TWI must be initialized as in the example below. In the example
has been used TWIC, Peripheral Clock frequency: 32000000 Hz,
SCL Rate: 100000 bps and Master interrupt: Low Level

*****************************************************************************

#include <twilcd.h>
.  .  .

// TWIC initialization
// structure that holds information used by the TWIC Master
// for performing a TWI bus transaction
TWI_MASTER_INFO_t twic_master;

// TWIC Master interrupt service routine
#pragma optsize- // optimize for speed
interrupt [TWIC_TWIM_vect] void twic_master_isr(void)
{
    twi_master_int_handler(&twic_master);
}
```

```
43: #pragma optsize_default
44:
45: void main(void)
46: {
47:
48:     // General TWIC initialization
49:     // External Driver Interface: Off
50:     // SDA Hold: Off
51:     twi_init(&TWIC,false,0);
52:
53:     // TWIC Master initialization
54:     // Master interrupt: Low Level
55:     // Peripheral Clock frequency: 32000000 Hz
56:     // SCL Rate: 100000 bps
57:     // Real SCL Rate: 100000 bps, Error: 0,0 %
58:     twi_master_init(&twic_master,&TWIC,TWI_MASTER_INTLVL_LO_gc,TWI_BAUD_REG(32000000,100000));
59:
60:     // TWIC Slave is disabled
61:     TWIC.SLAVE.CTRLA=0;
62:
63:     // set the MMA7455 functions to use TWIC
64:     MMA7455_twi_init(&twic_master,0x1D);
65:
66:     // enable interrupts
67:     #asm("sei")
68:
69:     // now the rest of the TWILCD functions can be used
70:
71:     // ....
72: }
73:
74: *****************************************************************************/
75:
76: #ifndef _TWILCD_INCLUDED_
77: #define _TWILCD_INCLUDED_
78:
79: #pragma used+
80:
81: //Initializes LCD03
82: //addr - address
83: //rel - pointer to the relogio variable (in milliseconds)
84: //tot - keyboard time (in milliseconds)
```

```
 85: void lcd03_twi_init(unsigned char addr,unsigned long int *rel,unsigned char tot);
 86:
 87: //writes 1 byte to LCD03 (adress ender) register 0
 88: //data - Byte to write
 89: void lcd_byte_write(unsigned char data);
 90:
 91: //writes 2 bytes to LCD03 (adress ender) register 0
 92: //data1 and data2 - bytes to write
 93: void lcd_2byte_write(unsigned char data1, unsigned char data2);
 94:
 95: //writes 3 bytes to LCD03 (adress ender) register 0
 96: //data1, data2 e data3 - bytes to write
 97: void lcd_3byte_write(unsigned char data1, unsigned char data2, unsigned char data3);
 98:
 99: //Cleans the lin line and places the cursor at the beginning of the line
100: void lcd_clear_line(unsigned char lin);
101:
102: //Sends the str string to the LCD03
103: void lcd_char_string_write(char *str);
104:
105: //Reads the LCD03 reg register
106: unsigned char lcd_register_read(unsigned char reg);
107:
108: //Reads the keypad character, sends it to the LCD03 and returns it
109: //mostra = 1 - send to the LCD03 the own characters
110: //mostra = 0 - send to the LCD03 the characters '_'
111: char read_display_keypad(unsigned char mostra);
112:
113: //Receives the command code from keypad
114: char receive_keypad_cod(unsigned char t_o);
115:
116: //Receives a command parameter and returns its size
117: //t_o - Maximum wait time in seconds
118: //mostra = 1 - displays the real characters
119: //mostra = 0 - displays '_' characters
120: //ptro - pointer to the command area
121: //nmax - maximum bytes number to receive
122: unsigned char receive_keypad_par(unsigned char t_o, unsigned char mostra, char *ptro, unsigned char nmax);
123:
124: //Reads the keypad character
125: char read_keypad(void);
126:
```

```
127: //Converts the keypad bits to the corresponding characters
128: char convert_keypad(unsigned char tipo, unsigned char car);
129:
130: //Displays a message
131: //If aguarda>=1 => waits the '*' keying
132: //If aguarda=1 => do not waits
133: void lcd_talk(char *fr1,char *fr2,char *fr3,unsigned char aguarda);
134:
135: //Prepares the LCD03 to receive the keyed command
136: unsigned char receive_command(char *cmdo);
137:
138: //Calculates the command code value. If OK, returns the value. On the contrary returns 0
139: unsigned char calcula(char *cmp);
140:
141: //Displays lines 1,2,3
142: //If seg > 0 => waits seg seconds and clears the display
143: void lcd_display123(char *fr1,char *fr2,char *fr3, unsigned char seg);
144:
145: //Displays lines 1,2,3,4
146: //If seg > 0 => waits seg seconds and clears the display
147: void lcd_display1234(char *fr1,char *fr2,char *fr3,char *fr4, unsigned char seg);
148:
149: //Clears the display buffer
150: void limpa_display_buffer(void);
151:
152: //Displays the str string from flash in the linha line
153: //limpa=1 => clears the display before
154: //limpa=0 => do not clears the display before
155: //seg>=0 => displays the line for seg seconds
156: //seg<0 => do not returns
157: //center=0 => line aligned to the left
158: //center=1 => line aligned to the center
159: //center<0 => line aligned at the present cursor position
160: //If remainder(linha/4) = 0 => line 4
161: //If remainder(linha/4) > 0 => this is the line
162: void display_flash(unsigned char limpa, unsigned char linha, signed char center, flash unsigned char *str, signed char
     seg);
163:
164: //Displays the str string in the linha line
165: //limpa=1 => clears the display before
166: //limpa=0 => do not clears the display before
167: //seg>=0 => displays the line for seg seconds
```

```
168: //seg<0 => do not returns
169: //center=0 => line aligned to the left
170: //center=1 => line aligned to the center
171: //center<0 => line aligned at the present cursor position
172: //If remainder(linha/4) = 0 => line 4
173: //If remainder(linha/4) > 0 => this is the line
174: void display(unsigned char limpa, unsigned char linha, signed char center, unsigned char *str, signed char seg);
175:
176: #pragma used-
177:
178: #pragma library twilcd.lib
179:
180: #endif
```