# AR-LCM02 User Manual

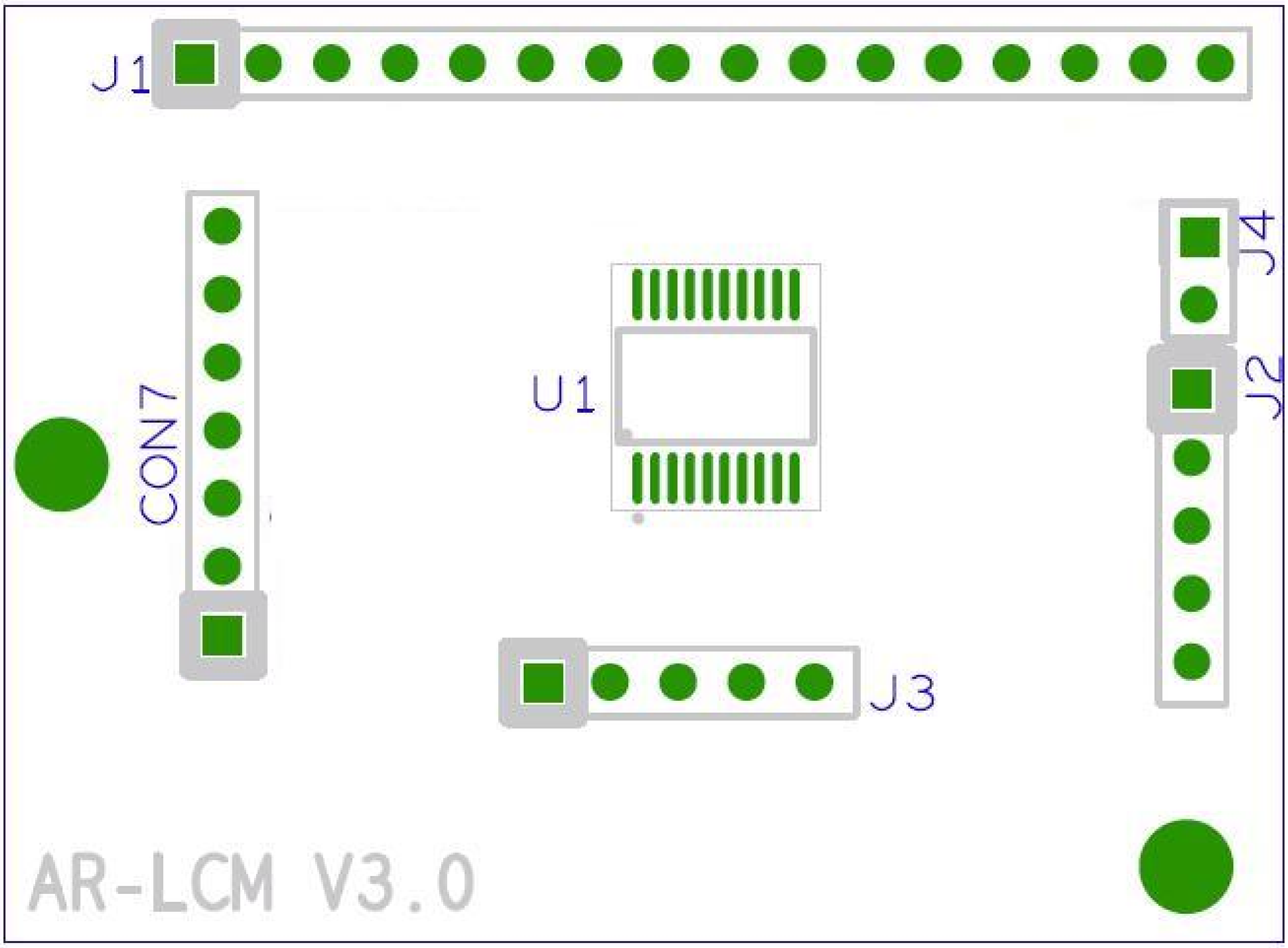
Document History

| Version | Date | Description |
|---|---|---|
| 1.0 | 2010/12/09 | Initial release |
| | | |
| | | |

# Contents

# 1. AR-LCM02 V3.0 User Manual

## 1.1. Board illustration (Top Side)



| | |
|---|---|
| ❶ | **J1**<br>LCM Connector (Linking LCM Control-Board data to LCM). |
| ❷ | **J2**<br>Key Pad input source. |
| ❸ | **J3**<br>Micro-Controller Updated Interface (updated U1’s Flash interface). |
| ❹ | **CON7**<br>Main System Connector (Communication with host and LCM control-board). |
| ❺ | **J4**<br>Output Type Select. |

## 1.2. Connector and Jumper Set-up

**1. J1**
LCM Connector (Linking LCM Control-Board data to LCM).

| | | | |
|---|---|---|---|
| 1: GND | 2: VCC | 3: BRDADJ | 4: LCMRS |
| 5: LCMRW | 6: LCMENA | 7: DB0 | 8: DB1 |
| 9: DB2 | 10: DB3 | 11: DB4 | 12: DB5 |
| 13: DB6 | 14: DB7 | 15: VCC | 16: BLCTL |

**2. J2**
Key Pad input source.

| | | | |
|---|---|---|---|
| 1: GND | 2: KEY0 (Enter Button) | 3: KEY1 (Down Button) | 4: KEY2 (Up Button) |
| 5: KEY3 (ESC button) | | | |

**3. J3**
Micro-Controller Updated Interface (updated U1's Flash interface).

| | | | |
|---|---|---|---|
| 1: VDD | 2: ISPDATA. | 3: ISPCLK | 4: VPP |
| 5: GND | | | |

**4. CON7**
Main System Connector (Communication with host and LCM control-board).

| | | | |
|---|---|---|---|
| 1: POR | 2: VCC | 3: RXD | 4: TXD |
| 5: CTS | 6: RTS | 7: GND | |

**5. J4**
Output Type Select.

| | |
|---|---|
| Open | For all products(New) |
| Short | For 9936 |

# 2. LCM API Software Programming Guide

## 2.1. File Descriptions

Please find these files from Product CD.

1. lcmdemo.c
   This file is the source code of the demo program. This program displays the user interface, processes user's input, and invokes LCM APIs to demonstrates the functions of LCM.

2. lcm.c
   This file includes the hardware independent implementation of LCM APIs. All the APIs in this file invoke the hardware dependent functions 'InitSerialPort( )', 'WriteSerial( )', 'ReadSerial( )' and 'CloseSerialPort( )' for accessing the serial port

3. lcm.h
   This file includes the declarations and macro definitions needed by lcm.c.

4. serialport.c
   This file includes the hardware dependent implementation of 'InitSerialPort( )', 'WriteSerial( )', 'ReadSerial( )' and 'CloseSerialPort( )' for accessing the serial port.

5. serialport.h
   This file includes the declarations and macro definitions needed by serialport.c.

6. Makefile
   This is the instruction script for GNU make system.

## 2.2. The Descriptions of LCM APIs

1. int lcmClrscr( void )

   **Description:** Clear the screen of the LCM.
   **Return value:** 0 after the screen is cleared.

2. int lcmCursor( bool mode )

   **Description:** According to the argument 'mode', show the cursor on the LCM screen or eliminate the cursor on the LCM screen. The position of the cursor is unchanged.
   mode = true, show the cursor.
   mode = false, eliminate the cursor.
   **Return value:** 0 after the cursor has been shown or eliminated.

3. int lcmCursorAction( int type)

   **Description:** According to the argument 'type', move the cursor to the indicated position. The displayed text is not altered.
   type = HOME, move the cursor to row 0, column 0.
   type = MOVERIGHT, move the cursor to the column which is to the right of its original position if the original column < 15.
   type = MOVELEFT, move the cursor to the column which is to the left of its original position if the original column > 0.
   type = MOVEBACK, move the cursor to the column which is to the left of its original position and delete the character at the new position if the original
   column > 0.
   **Return value:** 0 after the cursor is moved.

4. int lcmDisplay( bool mode )

   **Description:** Show the text on the LCM screen or eliminate the text on the LCM screen. The content of the text is not altered.
   mode = true, show the text.
   mode = false, eliminate the text.
   **Return value:** 0 after the text has been shown or eliminated.

5. int lcmGetKey( void )

**Description:** Scan the LCM and return the identification of the pressed direction key.
**Return value:** 'UP' if the 'up' direction key is pressed.
'RIGHT' if the 'right' direction key is pressed.
'LEFT' if the 'left' direction key is pressed.
'DOWN' if the 'down' direction key is pressed.
'NONE' if none of the keys is pressed.

6. int lcmGetPosition( int *row, int *column )

**Description:** Get the position of the cursor and write the coordinate to the memory pointed at by arguments 'row' and 'column'.
**Return value:** 0 if the request for the coordinate has been served.

7. int lemSetPosition( int row, int column )

**Description:** Set the position of the cursor according to the arguments 'row' and 'column'.
**Return value:**
0 after the position has been set.
-1 if the argument 'row' or 'column' meets any of the following conditions:
(1) row is not 0.
(2) row is not 1.
(3) column is less than 0.
(4) column is greater than 15.

8. int lcmShow( int length, unsigned char *info )

**Description:** Start from the current position of the cursor; print the text pointed at by 'info' to the LCM screen. The number of characters to be printed is at most 'length'. If the remaining columns available for printing the text is less than 'length', the number of the characters to be printed is:
16 – ( column number of the current position of the cursor ).
**Return value:** 0 after the text is printed.