ezLCD+

ezLCD+103 Product Manual
The ezLCD+ documentation consists of:

"ezLCD+10x Manual"
Specific for each ezLCD+ device (ezLCD+101, ezLCD+102, .. etc.).
- Provides "Quick Start" instructions.
- Describes the hardware of the particular device.
- Describes how to load a new firmware and how to customize your ezLCD+ device.

"ezLCD+ External Commands Manual"
Common for all ezLCD+ products.
- Describes the set of commands, which can be sent to the ezLCD+ through any of the implemented interfaces (USB, RS232, SPI, etc.). Those commands may be sent by an external host (PC or microcontroller).
- Describes the API of the ezLCD+ Windows USB driver.

"ezLCD+ Lua API Manual"
Common for all ezLCD+ products.
All ezLCD+ products have an embedded Lua interpreter. The ezLCD+ Lua API has been developed to access all graphic and I/O capabilities of the ezLCD+ device using the Lua language.

* Programming in Lua (second edition) By Roberto Lerusalimschy
Common for all ezLCD+ products.
The official book about the Lua programming language. It is available at:
http://www.amazon.com/exec/obidos/ASIN/8590379825/lua-docs-20
More information about Lua can be found at:
http://www.lua.org/

* Not included. Must be downloaded or purchased separately.
# Table of Contents

ezLCD-103 Product Manual

1 Overview .......................................................................................................................... 4

2 Design Concept .............................................................................................................. 6

3 Quick Start ..................................................................................................................... 7
   Quick Start: External Commands ............................................................................... 8
   Quick Start: Lua ........................................................................................................... 10

4 Hardware & Interfaces ................................................................................................... 11
   Specifications .............................................................................................................. 11
   Dimensions ................................................................................................................ 12
   Connectors, Jumpers and Switches ........................................................................... 13
      Assignments ............................................................................................................. 13
      PWR and PB1 ......................................................................................................... 14
      CN4 and CN2 .......................................................................................................... 15
      USB1 ....................................................................................................................... 17
      J1 ............................................................................................................................ 18
      CN1 ......................................................................................................................... 19
      CN3 ........................................................................................................................ 20
      RS232 ..................................................................................................................... 21
      SPI ......................................................................................................................... 23
      SD .......................................................................................................................... 24
      USB ....................................................................................................................... 25
      I2C ........................................................................................................................ 26
      Audio DAC ............................................................................................................ 27

5 Firmware ....................................................................................................................... 28
   Firmware Upgrade ................................................................................................... 28

6 Display ........................................................................................................................... 29
   Organization .............................................................................................................. 29
   Frames ....................................................................................................................... 30

7 Touch Screen .................................................................................................................. 31
   Introduction .............................................................................................................. 31
   Calibration ............................................................................................................... 32

8 ezLCD+ Executable Files ............................................................................................... 33

9 ezLCD+ Customization .................................................................................................. 34
   User ROM ............................................................................................................... 36
   User Configuration ................................................................................................. 38
      User Configuration Keys ..................................................................................... 40

10 Demo Mode .................................................................................................................... 48

11 GLOSSARY .................................................................................................................. 49

© 2008 Earth Computers Tech. Inc.
ezLCD-103 Product Manual

1 Overview

Congratulations on your purchase of ezLCD+103!

The ezLCD+103 is an all-in-one programmable color LCD panel which includes:
- 320x240 pixel, 262144 color, 3.5" TFT LCD
- Embedded 32bit processor (Atmel AT32AP7000) with LCD Controller
- 4 Mega Bytes of embedded flash for storing custom fonts and bitmaps
- MicroSD Card slot for storing bitmap, fonts and other user data up to 2 Giga Bytes
- Power supply, which generates all the voltages needed by the logic and the display itself
- Touch screen
- Interface drivers and other circuitry

The ezLCD+103 communicates with the outside world through several interfaces:
- RS232 TTL
- USB
- I2C
- SPI
- SD/MMC

The ezLCD+103 firmware is in-field updatable and contains an extensive command set:
- Graphic commands
- Double buffering
- True Type and Open Type font rendering
- Bitmap font rendering
- Unicode support
- SD file I/O (FAT12, FAT16, and FAT32)
- Touch Screen commands

The ezLCD+103 firmware is in-field updatable and contains an extensive command set:
- Graphic commands
- Double buffering
- True Type and Open Type font rendering
- Bitmap font rendering
- Unicode support
- SD file I/O (FAT12, FAT16, and FAT32)
- Touch Screen commands

The ezLCD+103 may be in-field customized by:
- Adding custom fonts
- Adding custom bitmaps
- Customizing startup screen
- Modifying interface parameters like RS232 baudrate, Ethernet MAC and IP address, etc.
- Modifying pin functions

The ezLCD+103 belongs to the ezLCD+ family of intelligent displays.
All ezLCD+ devices (including ezLCD+103) have two distinctive user interfaces:
1. The ezLCD+ External Commands. The ezLCD+ is driven by a set of commands, which can be fed through any of the implemented interfaces apart from I2C. The ezLCD+ External Commands Manual
describes in detail those commands.

2. Lua programming language interpreter. The ezLCD+ contains an embedded Lua programming language interpreter. The ezLCD+ Lua library is described in the *ezLCD+ Lua API Manual*. The ezLCD+103 may be used as an "intelligent" display, or as a stand alone device. There is enough flash memory left to incorporate additional graphical instructions, or to customize the software for particular tasks. Possible applications include automotive, avionics, nautical, industrial control, hobby, etc.
2. Design Concept

The ezLCD+103 consists of:

- **FAVR-32 Module.** Stores and executes the firmware. Contains 32AP7000 32-bit microcontroller with LCD controller, RAM, ROM, Clocks and a local voltage regulator.
- **Personality Board.** Interfaces the FAVR-32 module with the outside world. Contains several I/O connectors, DC/DC voltage regulator, and **Configuration EEPROM.**
- LCD Panel
- LCD Backlight Inverter

The Configuration EEPROM stores information about all the hardware outside the AVR-32 Module:
- LCD parameters (resolution, number of colors, size, voltages, timings, etc.)
- Connectors and drivers aboard the Personality Board
- Touch Screen type and parameters
- etc.

FAVR-32 module is connected to the Personality Board by three 1.5 mm height 50-pin connectors. Upon power-up, FAVR-32 Module reads the LCD parameters and other configuration data from the Configuration EEPROM. The read parameters and data is used to configure firmware for driving the particular LCD display and peripherals.

The design concept described above enables the Firmware and FAVR-32 Module to be identical for several types of LCD graphic displays available on the market today. The LCD controller embedded in the FAVR-32 module is capable of driving TFT, STN, Color, or Monochrome LCD displays with the resolution of up to 2048x2048. The small size of the FAVR-32 Module enables it to fit on the back of an LCD panel starting with the size of 2.7".

Incorporating a new LCD panel would require the following:
1. Design a Personality Board for the particular LCD panel
2. Program the Configuration EEPROM with the parameters of the required LCD panel
3 Quick Start

Quick Start Requirements:
- PC Computer with at least 1 USB 2.0 port
- Windows XP SP2, or Windows Server 2003, or any Windows Vista or Windows Server 2008
Note: The ezLCD+ products do not need a PC computer to work. The above requirements are for the "Quick Start" only.

Quick Start
1. Download the latest USB FAVR-32 driver from http://www.ezlcd.com/support/

2. Run the downloaded driver installation executable before connecting ezLCD+103 to the USB of your computer.

3. Connect ezLCD+103 USB to your computer and turn the ezLCD+103 power on by sliding the PWR switch into "ON" position. "New Hardware Found" wizard should appear. Select automatic driver installation. Turn-off ezLCD+103 after the driver have successfully been installed.

4. Go to chapter: "Quick Start: External Commands" or "Quick Start: Lua".
3.1 Quick Start: External Commands

1. Make sure, that USB FAVR-32 driver is installed on your PC

2. Download the setup of "Learn_ezLCD" utility from http://www.ezlcd.com/support/

3. Install "Learn_ezLCD" utility by running the downloaded setup

4. Turn-on ezLCD+103 and make sure that it is connected to your computer through USB.

5. Run "Learn_ezLCD" utility. Press "Display this window" button. "Learn_ezLCD" utility window should appear on the ezLCD.

6. Read "Learn_ezLCD" Help and experiment with ezLCD commands and "Learn_ezLCD" buttons

About Learn_ezLCD

"Learn_ezLCD" is a simple utility, designed to help beginners learn how to use ezLCD commands. The picture, below, shows the main window of "Learn_ezLCD", with some short descriptions of it's components.

Upon start, "Learn_ezLCD" dynamically generates Command Buttons, based on the data read from the file: Buttons.txt. File Buttons.txt contains the button definitions. It resides in the same folder as Learn_ezLCD.exe (application executable).

Besides that, the button definitions can be loaded from any other Button Definition File by selecting File- >Open from the menu.

Generated buttons are shown in the upper-left part of the picture above.

As an example, the script defining button: ellipse 200,100 is shown to the right of it.

Permanent components of "Learn_ezLCD" are located in the right part of it's window: They include:
• The Data Sent Table, which shows the first 16 bytes of the data sent to the ezLCD upon pressing the button
• Three Utility Buttons: 'Display this window', 'Capture Screen' and 'Ping ...'
• ezLCD Backlight Control Slider

For more information about ezLCD+ External Commands, please refer to the "ezLCD+ External Commands Manual".
3.2 Quick Start: Lua

1. Make sure, that USB FAVR-32 driver is installed on your PC

2. Download the setup of "ezLuaIDE" from http://www.ezlcd.com/support/

3. Install "ezLuaIDE" by running the downloaded setup

4. Turn-on ezLCD+103 and make sure that it is connected to your computer through USB.

5. Run "ezLuaIDE". From the Menu, select "File" - "Open"

6. Select HelloWorld.lua file from the folder "Program Files\ezLuaIDE\Examples".

```
1    -- Select Display & Draw Frames
    ez.SetDispFrame(0)
    ez.SetDrawFrame(0)
    -- Fill screen with navy color
    ez.Cls(ez.RGB(0, 0, 128))
    -- Select True Type font no 6, height - 64 pixels, Width - Automatic
    ez.SetTFont(6, 64, 0)
    -- Set golden color for drawing
    ez.SetColor(ez.RGB(255, 215, 0))
    -- Set screen position for drawing
    ez.SetXY(10, 10)
    -- Print Hello World!
    print("Hello World!")
```

7. Press button. The ezLCD+103 should display "Hello World!" in golden color over navy background:

![Hello World!]

For more information about Lua on ezLCD+ and ezLuaIDE, please refer to the "ezLCD+ Lua API Manual".
4 Hardware & Interfaces

4.1 Specifications

### Electrical and Environmental Characteristics

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Unit</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Supply Voltage</td>
<td>Vcc</td>
<td>4.5</td>
<td>5.0</td>
<td>6.0</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Power Supply Current</td>
<td>Icc</td>
<td>360</td>
<td>420</td>
<td></td>
<td>mA</td>
<td>Vcc = 5V, maximum backlight</td>
</tr>
<tr>
<td>Hi Level Discrete Input Voltage</td>
<td>VH</td>
<td>2.2</td>
<td>3.3</td>
<td>3.6</td>
<td>V</td>
<td>Note: Discrete signals are not +5V tolerant. This includes SPI, I2C, SD</td>
</tr>
<tr>
<td>Lo Level Discrete Input Voltage</td>
<td>VL</td>
<td>-0.3</td>
<td>0</td>
<td>1</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>Topa</td>
<td>-20</td>
<td>70</td>
<td></td>
<td>°C</td>
<td>Max. storage time at -30°C: 48hrs</td>
</tr>
<tr>
<td>Storage Temperature</td>
<td>Tstg</td>
<td>-30</td>
<td>80</td>
<td></td>
<td>°C</td>
<td>Max. storage time at 80°C: 120hrs</td>
</tr>
<tr>
<td>Operating Ambient Humidity</td>
<td>HOP</td>
<td></td>
<td>75</td>
<td></td>
<td>%RH</td>
<td>Background color changes slightly depending on ambient temperature. This phenomenon is reversible.</td>
</tr>
<tr>
<td>Storage Humidity</td>
<td>HST</td>
<td></td>
<td>75</td>
<td></td>
<td>%RH</td>
<td></td>
</tr>
</tbody>
</table>

### Display Specifications

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Specification</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Resolution</td>
<td>320(W) x 240(H)</td>
<td>pixel</td>
</tr>
<tr>
<td>Color Depth</td>
<td>8-bit, 16,777,216 colors</td>
<td></td>
</tr>
<tr>
<td>Active Screen Size</td>
<td>3.5 (Diagonal)</td>
<td>inch</td>
</tr>
<tr>
<td>Contrast Ratio</td>
<td>300:1</td>
<td></td>
</tr>
<tr>
<td>Pixel Pitch</td>
<td>0.219(W) x 0.219(H) mm</td>
<td>mm</td>
</tr>
<tr>
<td>Luminance</td>
<td>350</td>
<td>Cd/m²</td>
</tr>
<tr>
<td>Operating Mode</td>
<td>Transmissive, normally white</td>
<td></td>
</tr>
</tbody>
</table>

**Backlight**

Backlight lamp life: 50000 hrs until brightness is at 70% of original brightness.
4.2 Dimensions
# Connectors, Jumpers and Switches

## Assignments

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Matching Connectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>CN1</td>
<td>LCD/Touchscreen Signals</td>
<td>Standard flex cable 54x0.5mm</td>
</tr>
<tr>
<td>CN2</td>
<td>MicroSD</td>
<td>Standard MicroSD Card</td>
</tr>
<tr>
<td>CN3</td>
<td>Ethernet MAC Signals (back of the board)</td>
<td>Hirose DF23C-50DS-0.5V</td>
</tr>
<tr>
<td>CN4</td>
<td>Power and Interfaces (I2C, SPI, RS232 TTL, SD, Audio, discrete)</td>
<td>Hirose DF11-32DS</td>
</tr>
<tr>
<td>USB1</td>
<td>USB</td>
<td>Standard Mini USB-B Plug</td>
</tr>
<tr>
<td>J1</td>
<td>USB Power Jumper</td>
<td>Standard 0.1” Jumper Block</td>
</tr>
<tr>
<td>PWR</td>
<td>Power On/Off Switch</td>
<td>N/A</td>
</tr>
<tr>
<td>PB1</td>
<td>Programming Button</td>
<td>N/A</td>
</tr>
</tbody>
</table>

*ezLCD+103 Connectors, jumpers and Switches*
4.3.2 PWR and PB1

PWR: Low current, power On/Off switch. Sends the signal to the ezLCD+103 power supply to switch the power on or off. It has the same function as ON/OFF signal (CN4).

PB1: Programming button. Starts ezLCD+103 bootloader, when pressed during power-up. Starts touch screen calibration, when momentarily pressed. Starts Demo Mode when pressed for more than 2 seconds. It has the same function as PROG# signal (CN4).
4.3.3 CN4 and CN2

Connector CN2 is used as a slot for MicroSD card. CN2 signals are repeated on CN4.
 Connector CN4 contains power and interface signals (I2C, SPI, RS232 TTL, SD, Audio and discrete).
 Pin0 to Pin16 specify pins, which can be reconfigured by Lua applications.

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ext Pwr+</td>
<td>Ext. Power</td>
<td>External power voltage (Vcc) 4.5V to 6V</td>
</tr>
<tr>
<td>GND</td>
<td>Gnd</td>
<td>Common power return and signal gnd</td>
</tr>
<tr>
<td>ON/OFF</td>
<td>Input</td>
<td>ON/OFF signal. The same function as PWR (ON) switch. +2.2 to +6V turns the power ON. 0 to +0.8V turns the power OFF. Rin &gt; 100 kOhm This signal is pulled to GND by an internal resistor</td>
</tr>
<tr>
<td>PROG#</td>
<td>Discrete Input</td>
<td>Firmware download signal. The same function as PB1 pushbutton. Starts ezLCD+103 bootloader, when connected to GND during power-up. Starts touch screen calibration, when momentary connected to GND. Starts Demo Mode when connected to GND for more than 2 seconds This signal is pulled to +3.3V by an internal resistor</td>
</tr>
<tr>
<td>+3.3V out</td>
<td>Pwr/Out</td>
<td>+3.3V/0.5A regulated voltage output. May be used as a power supply for external devices.</td>
</tr>
<tr>
<td>Spare0 to Spare4</td>
<td>Discrete I/O</td>
<td>Spare I/O discrete signals. May be used by Lua applications</td>
</tr>
<tr>
<td>SPI MISO</td>
<td>Discrete Output</td>
<td>SPI Master Input Slave Output signal</td>
</tr>
<tr>
<td>SPI SS#</td>
<td>Discrete Input</td>
<td>SPI Slave Select input</td>
</tr>
<tr>
<td>SPI SPCK</td>
<td>Discrete Input</td>
<td>SPI Clock input</td>
</tr>
<tr>
<td>SPI MOSI</td>
<td>Discrete Input</td>
<td>SPI Master Output Slave Input signal</td>
</tr>
<tr>
<td>Signal Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>RS232 TTL CTS</td>
<td>Discrete Input (0 - 3.3V)</td>
<td>RS-232 TTL Clear To Send input&lt;br&gt;Note: This signal has different voltage levels than RS-232 standard</td>
</tr>
<tr>
<td>RS232 TTL RTS</td>
<td>Discrete Output (0 - 3.3V)</td>
<td>RS-232 TTL Request To Send output&lt;br&gt;Note: This signal has different voltage levels than RS-232 standard</td>
</tr>
<tr>
<td>RS232 TTL RXD</td>
<td>Discrete Input (0 - 3.3V, 1.5mA)</td>
<td>RS-232 TTL Received Data input&lt;br&gt;Note: This signal has different voltage levels than RS-232 standard</td>
</tr>
<tr>
<td>RS232 TTL TXD</td>
<td>Discrete Output (0 - 3.3V)</td>
<td>RS-232 TTL Transmitted Data output&lt;br&gt;Note: This signal has different voltage levels than RS-232 standard</td>
</tr>
<tr>
<td>SD WPR</td>
<td>Discrete Input (0 - 3.3V)</td>
<td>SD Write Protect&lt;br&gt;This signal is not used by ezLCD+103</td>
</tr>
<tr>
<td>CD CIN#</td>
<td>Discrete Input (0 - 3.3V)</td>
<td>SD Card Inserted&lt;br&gt;This signal is not used by ezLCD+103</td>
</tr>
<tr>
<td>SD DAT1</td>
<td>Discrete I/O (0 - 3.3V or Opened)</td>
<td>SD Data 1 signal&lt;br&gt;This signal is pulled to +3.3V by an internal resistor</td>
</tr>
<tr>
<td>SD DAT2</td>
<td>Discrete I/O (0 - 3.3V or Opened)</td>
<td>SD Data 2 signal&lt;br&gt;This signal is pulled to +3.3V by an internal resistor</td>
</tr>
<tr>
<td>SD DAT3</td>
<td>Discrete I/O (0 - 3.3V or Opened)</td>
<td>SD Data 3 signal&lt;br&gt;This signal is pulled to +3.3V by an internal resistor</td>
</tr>
<tr>
<td>SD CMD</td>
<td>Discrete I/O (0 - 3.3V or Opened)</td>
<td>SD Command signal&lt;br&gt;This signal is pulled to +3.3V by an internal resistor</td>
</tr>
<tr>
<td>SD DAT0</td>
<td>Discrete I/O (0 - 3.3V or Opened)</td>
<td>SD Data 0 signal&lt;br&gt;This signal is pulled to +3.3V by an internal resistor</td>
</tr>
<tr>
<td>SD CLK</td>
<td>Discrete Output (0 - 3.3V)</td>
<td>SD Clock</td>
</tr>
<tr>
<td>I2C SDA</td>
<td>Discrete I/O (Open - Gnd)</td>
<td>I2C interface SDA (Serial Data) signal</td>
</tr>
<tr>
<td>I2C SCL</td>
<td>Discrete I/O (Open - Gnd)</td>
<td>I2C interface SCL (Serial Clock) signal</td>
</tr>
<tr>
<td>DAC-R</td>
<td>0 - 3.3V PWM Output</td>
<td>Audio Bitstream DAC Right</td>
</tr>
<tr>
<td>DAC-L</td>
<td>0 - 3.3V PWM Output</td>
<td>Audio Bitstream DAC Left</td>
</tr>
</tbody>
</table>
4.3.4 USB1

Connector USB1 is used for USB interface. For signal description, please refer to the USB 2.0 specification.
Jumper J1 is used to power ezLCD+103 from the USB.

When J1 is **opened**: ezLCD+103 is powered from the voltage between CN4 pins: Ext Pwr+ and GND.

When J1 is **closed**: ezLCD+103 is powered from the voltage between USB1 pins: VBUS and GND, or between CN4 pins: Ext Pwr+ and GND, whichever voltage is greater.

*ezLCD+103 Input Power Circuitry*
Connector CN1 is used for LCD and Touchscreen signals. The pin assignments fit the EarthLCD display ECT-TG320240S03E only. The information below is provided for reference only.

<table>
<thead>
<tr>
<th>Pin No</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,2</td>
<td>LED-</td>
<td>20mA current source</td>
<td>Backlight current source</td>
</tr>
<tr>
<td>3,4</td>
<td>LED+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5,6,7</td>
<td>N/C</td>
<td></td>
<td>Not connected</td>
</tr>
<tr>
<td>8</td>
<td>Ty-</td>
<td>Resistance</td>
<td>Touch screen y- input</td>
</tr>
<tr>
<td>9</td>
<td>Tx+</td>
<td>Resistance</td>
<td>Touch screen x+ input</td>
</tr>
<tr>
<td>10</td>
<td>Ty+</td>
<td>Resistance</td>
<td>Touch screen y+ input</td>
</tr>
<tr>
<td>11</td>
<td>Tx-</td>
<td>Resistance</td>
<td>Touch screen x- input</td>
</tr>
<tr>
<td>12 to 19</td>
<td>B0 to B7</td>
<td>Discrete Output (0 - 3.3V)</td>
<td>8 bits of RGB Blue component</td>
</tr>
<tr>
<td>20 to 27</td>
<td>G0 to G7</td>
<td>Discrete Output (0 - 3.3V)</td>
<td>8 bits of RGB Green component</td>
</tr>
<tr>
<td>28 to 35</td>
<td>R0 to R7</td>
<td>Discrete Output (0 - 3.3V)</td>
<td>8 bits of RGB Red component</td>
</tr>
<tr>
<td>36</td>
<td>HSYNC</td>
<td>Discrete Output (0 - 3.3V)</td>
<td>Horizontal Synchronization</td>
</tr>
<tr>
<td>37</td>
<td>VSYNC</td>
<td>Discrete Output (0 - 3.3V)</td>
<td>Vertical Synchronization</td>
</tr>
<tr>
<td>38</td>
<td>DCLK</td>
<td>Discrete Output (0 - 3.3V)</td>
<td>Data Clock</td>
</tr>
<tr>
<td>39,40</td>
<td>N/C</td>
<td></td>
<td>Not connected</td>
</tr>
<tr>
<td>41, 42</td>
<td>+3.3V Out</td>
<td>Pwr/Out</td>
<td>+3.3V/0.5A regulated voltage output</td>
</tr>
<tr>
<td>44 to 47</td>
<td>N/C</td>
<td></td>
<td>Not connected</td>
</tr>
<tr>
<td>48</td>
<td>GND</td>
<td>Gnd</td>
<td>Common power return and signal gnd</td>
</tr>
<tr>
<td>49,50,51</td>
<td>N/C</td>
<td></td>
<td>Not connected</td>
</tr>
<tr>
<td>52</td>
<td>ENB</td>
<td>Discrete Output (0 - 3.3V)</td>
<td>Data enable</td>
</tr>
<tr>
<td>53,54</td>
<td>GND</td>
<td>Gnd</td>
<td>Common power return and signal gnd</td>
</tr>
</tbody>
</table>
Connectors CN3 is used to connect ezLCD+103 to ethernet network through the 10/100 Mb/s Ethernet Physical Layer Transceiver (PHY)

**MAC Signals**
Standard Media Access Control (MAC) Data Interface signals. For more information, please refer to IEEE 802.3 specification.

**SMI Signals**
Standard Serial Management Interface signals. For more information, please refer to IEEE 802.3 specification.

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHY PD#</td>
<td>Discrete Output</td>
<td>PHY Power Down signal. Active Lo. (0 - 3.3V or Opened)</td>
</tr>
<tr>
<td>PHY RST#</td>
<td>Discrete Output</td>
<td>PHY Reset signal. Active Lo. (0 - 3.3V or Opened)</td>
</tr>
<tr>
<td>+3.3V out</td>
<td>Pwr/Out</td>
<td>+3.3V/0.5A regulated voltage output.</td>
</tr>
<tr>
<td>GND</td>
<td>Gnd</td>
<td>Common power return and signal gnd</td>
</tr>
</tbody>
</table>
4.4 RS232

**Warning:** RS232 TTL uses logic level signals: Min = 0V, Max = +3.3V. Connecting RS232 TTL to "standard" RS232 interface with the bipolar signal levels of (±3 V, ±5 V, etc.) may damage the ezLCD+103 and void the warranty.

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GND</td>
<td>Gnd</td>
<td>signal gnd</td>
</tr>
<tr>
<td>RS232 TTL CTS</td>
<td>Discrete Input</td>
<td>RS-232 TTL Clear To Send input</td>
</tr>
<tr>
<td></td>
<td>(0 - 3.3V, 1.5mA)</td>
<td></td>
</tr>
<tr>
<td>RS232 TTL RTS</td>
<td>Discrete Output</td>
<td>RS-232 TTL Request To Send output</td>
</tr>
<tr>
<td></td>
<td>(0 - 3.3V)</td>
<td></td>
</tr>
<tr>
<td>RS232 TTL RXD</td>
<td>Discrete Input</td>
<td>RS-232 TTL Received Data input</td>
</tr>
<tr>
<td></td>
<td>(0 - 3.3V, 1.5mA)</td>
<td></td>
</tr>
<tr>
<td>RS232 TTL TXD</td>
<td>Discrete Output</td>
<td>RS-232 TTL Transmitted Data output</td>
</tr>
<tr>
<td></td>
<td>(0 - 3.3V)</td>
<td></td>
</tr>
</tbody>
</table>

The ezLCD+103 uses "RS232 TTL" signals, which have different logic voltage levels than "standard" RS232. The drawings below show the differences between both RS232 signals:

**Default Communication Parameters**
Baudrate: 115200 bps
No of Data Bits: 8
No of Stop Bits: 1
Parity: None
Handshake: None

Most of the above parameters can be permanently modified by changing User Configuration. See Chapter ezLCD+ Customization / User Configuration.

**ezLCD+103 Power-Up Ready Transmission**
If the RS232 is set as the "Default Transmitter", it is used by ezLCD to send EZLCD_READY byte (EA hex, 234 dec). The EZLCD_READY byte is sent one time only, upon the power-up when the ezLCD+103 RS232 interface is ready to receive the commands.
The "Default Transmitter" can be set by changing User Configuration. See Chapter ezLCD+ Customization / User Configuration.
4.5 SPI

Communication Parameters
Max \( f_{\text{SPCK}} \): 4MHz
SPCK Idle: Low
No of Data Bits: 8
Bit Order: MSB goes first
Data sampled on: Leading edge of the SPCK (rising edge)
ezLCD+103 is: SPI Slave

### SPI Communication Parameters

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gnd</td>
<td>Gnd</td>
<td>Common power return and signal gnd</td>
</tr>
<tr>
<td>SPI MISO</td>
<td>Discrete Output (0 - 3.3V)</td>
<td>SPI Master Input Slave Output signal</td>
</tr>
<tr>
<td>SPI SS#</td>
<td>Discrete Input (0 - 3.3V)</td>
<td>SPI Slave Select input</td>
</tr>
<tr>
<td>SPI SPCK</td>
<td>Discrete Input (0 - 3.3V)</td>
<td>SPI Clock input</td>
</tr>
<tr>
<td>SPI MOSI</td>
<td>Discrete Input (0 - 3.3V)</td>
<td>SPI Master Output Slave Input signal</td>
</tr>
</tbody>
</table>

Receiving the data from the ezLCD+103

Since:
- The ezLCD+103 is configured as an SPI Slave and
- All transmissions through the SPI interface have to be initiated by the Master,

it is the user's responsibility to query the ezLCD for any new data, for example: touch screen coordinates.

Each time, the byte is sent through the SPI interface to the ezLCD+103, the unit responds on the MISO pin. if you want to query the ezLCD without sending any command: send 0 to the ezLCD.

ezLCD+103 Power-Up Ready Transmission

If the SPI is set as the "Default Transmitter", it is used by ezLCD+103 to send EZLCD_READY byte (EAhex, 234dec). The EZLCD READY byte is sent one time only, upon the power-up when the ezLCD+103 SPI interface is ready to receive the commands.

The "Default Transmitter" can be set by changing User Configuration. See Chapter ezLCD+ Customization / User Configuration.
4.6 SD

The SD Interface hardware supports the SD Memory Card Specification V1.0. SD Memory Card operations are supported by the firmware.

The ezLCD+103 firmware automatically adjusts the baudrate and other timing parameters by reading the SD Card parameters in the slow clock mode.

**Max Available Baudrate:**
- 96 Mbps (read from the SD Card)
- 24 Mbps (write to the SD Card)

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GND</td>
<td>Gnd</td>
<td>Common power return and signal gnd</td>
</tr>
<tr>
<td>+3.3V out</td>
<td>Pwr/Out</td>
<td>+3.3V/0.5A regulated voltage output.</td>
</tr>
<tr>
<td>SD DAT1</td>
<td>Discrete I/O</td>
<td>SD Data 1 signal</td>
</tr>
<tr>
<td></td>
<td>(0 - 3.3V or Opened)</td>
<td>This signal is pulled to +3.3V by an internal resistor</td>
</tr>
<tr>
<td>SD DAT2</td>
<td>Discrete I/O</td>
<td>SD Data 2 signal</td>
</tr>
<tr>
<td></td>
<td>(0 - 3.3V or Opened)</td>
<td>This signal is pulled to +3.3V by an internal resistor</td>
</tr>
<tr>
<td>SD DAT3</td>
<td>Discrete I/O</td>
<td>SD Data 3 signal</td>
</tr>
<tr>
<td></td>
<td>(0 - 3.3V or Opened)</td>
<td>This signal is pulled to +3.3V by an internal resistor</td>
</tr>
<tr>
<td>SD CMD</td>
<td>Discrete I/O</td>
<td>SD Command signal</td>
</tr>
<tr>
<td></td>
<td>(0 - 3.3V or Opened)</td>
<td>This signal is pulled to +3.3V by an internal resistor</td>
</tr>
<tr>
<td>SD DAT0</td>
<td>Discrete I/O</td>
<td>SD Data 0 signal</td>
</tr>
<tr>
<td></td>
<td>(0 - 3.3V or Opened)</td>
<td>This signal is pulled to +3.3V by an internal resistor</td>
</tr>
<tr>
<td>SD CLK</td>
<td>Discrete Output</td>
<td>SD Clock</td>
</tr>
<tr>
<td></td>
<td>(0 - 3.3V)</td>
<td></td>
</tr>
</tbody>
</table>
4.7 USB

For the detailed description of the USB signals, please refer to the USB 2.0 specification.

The ezLCD+103 USB interface is compatible with the USB 2.0 specification and supports Hi speed (240Mbps) communication.

**USB Drivers**

The drivers for PC are compatible with Windows XP, Server 2003, Vista and Server 2008. Both, 32 and 64 bit (ia64 and amd64) platforms are supported. The drivers are available in the "Drivers" directory of the ezLCD+103 CD and in the support section of the ezLCD web site: http://www.ezlcd.com/support/

Drivers are distributed in the form of windows applications:

- Favr32_USB_x86.exe - 32-bit Windows (all processors)
- Favr32_USB_amd64.exe - 64-bit Windows run by amd64 processor
- Favr32_USB_ia64.exe - 64-bit Windows run by ia64 processor

**Note:** Please, run the driver installation executable before connecting ezLCD+103 to the USB of your computer.

**ezLCD+103 Power-Up Ready Transmission**

If the USB is set as the "Default Transmitter", it is used by ezLCD+103 to send EZLCD_READY byte (EA\textsubscript{hex}, 234\textsubscript{dec}). The EZLCD\_READY byte is sent one time only, upon power-up, when ezLCD+103 USB interface is ready to receive commands.

The "Default Transmitter" can be set by changing User Configuration. See Chapter ezLCD+ Customization / User Configuration.
4.8 I2C

Since the ezLCD+103 is configured as I2C Master, the I2C interface cannot be used to receive ezLCD+ External Commands. On the other hand the I2C interface can be used by Lua to communicate with external devices like A/D converters, temperature sensors, serial memories, etc.

Internally, the I2C interface is used by ezLCD+103 to communicate with Configuration EEPROM as shown on the drawing to the left.

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I2C SDA</td>
<td>Discrete I/O (Open - Gnd)*</td>
<td>I2C interface SDA (Serial Data) signal</td>
</tr>
<tr>
<td>I2C SCL</td>
<td>Discrete I/O (Open - Gnd)*</td>
<td>I2C interface SCL (Serial Clock) signal</td>
</tr>
<tr>
<td>+3.3V out</td>
<td>Pwr/Out</td>
<td>+3.3V/0.5A regulated voltage output. May be used as a power supply for external devices.</td>
</tr>
<tr>
<td>GND</td>
<td>Gnd</td>
<td>Common power return and signal gnd</td>
</tr>
</tbody>
</table>

*Important:* Although I2C signals are of Open/Gnd type, their pins are internally connected to +3.3V through 10k pull-up resistors, as show on the drawing to the left.

**4.9 Audio DAC**

For audio playback, the outputs DAC-R and DAC-L can be connected to a class D amplifier output stage, or they can be low pass filtered and connected to a high input impedance amplifier. A simple 1st order or higher low pass filter that filters all the frequencies above 50 kHz should be adequate.
5 Firmware

5.1 Firmware Upgrade

The firmware upload to the ezLCD+103 is performed through the SD card.

The ezLCD+103 firmwares are distributed as the ezLCD+ executable files. They have an extension .eze.

To upgrade the ezLCD+103 firmware:
1. Make sure that the SD card is formatted in FAT-32, FAT-16 or FAT-12.
2. Copy the firmware file (.eze) to the SD card.
3. Make sure that the firmware is the only file with the .eze extension on the SD root directory.
4. Power off the ezLCD+103.
5. Insert the SD card into the ezLCD SD slot.
6. Hold down the PB1 button on the ezLCD+103 board (or connect the PROG# pin to the ground).
7. Power on the ezLCD+103.
8. After 2-3 seconds release the PB1 button (or disconnect the PROG# pin from the ground).
9. Wait until the programming is finished and the ezLCD restarts. Note: this may take up to 2 minutes. Make sure that the PROG# is not active, otherwise the programming will be repeated.
6 Display

6.1 Organization

The ezLCD+103 display frame is organized in 240 rows of 320 columns of pixels, as shown on the picture to the right. Each pixel of 4 bytes (32 bits): one byte for the red color, one for green, one for blue and one spare. The whole display frame takes 320*240*4 = 307,200 bytes.

Since the last byte of the pixel is not used, this arrangement may seem like a waste of the memory, however 32 bits per pixel are processed faster by the 32 bit ezLCD CPU than 24 bits would do.

The drawing to the right shows the organization of the display frame memory. Rotated numbers show the byte offset from the beginning of the frame.
6.2 Frames

In order to support special effects (animation, double buffering, etc) the whole ezLCD video RAM is divided into the 20 separate frames. Each frame has the capacity of the full ezLCD screen: 320x240x32bits.

The ezLCD commands draw on the frame selected by the ezLCD+ Command: SET_DRAW_FRAME or ezLCD+ Lua instruction: SetDrawFrame

The screen displays image from the frame selected by the ezLCD+ Command: SET_DISP_FRAME or ezLCD+ Lua instruction: SetDispFrame.

The same frame can be used to draw in display data. Upon power-up both draw and display frames are set to the Frame 0.

The entire data from one frame can be copied to the another by using one of the ezLCD+ Commands: COPY_FRAME
MERGE_FRAME
or equivalent ezLCD+ Lua instructions:
CopyFrame
MergeFrame

The portion of one frame can be copied to the another by using one of the ezLCD+ Commands:
COPY_RECT
MERGE_RECT
or equivalent ezLCD+ Lua instructions:
CopyRect
MergeRect
7 Touch Screen

7.1 Introduction

The ezLCD+103 has a 4-wire resistive analog touch screen. This touch screen consists of two layers of transparent resistive material with silver ink for electrodes. These two layers are stacked on an insulating layer of glass, separated by tiny spacer dots. They are interfaced electrically to the dedicated ezLCD+103 A/D converter. During measurement of a given coordinate, one of the resistive planes is powered along its axis and the other plane is used to sense the location of the coordinate on the powered plane.

For example, in case of the X coordinate measurement, the X plane is powered, as shown on the drawing below. The Y plane is used to sense where the pen is located on the powered plane as follows: At the location where the pen depresses the touch screen, the planes are shorted. The voltage measured on the sensed plane is proportional to the location of the touch on the powered plane. This voltage is then converted by the dedicated ezLCD+103 ADC as shown on the drawing below.
7.2 Calibration

The touch screen calibration starts upon pressing PB1 button (or connecting PROG# pin to ground) when the unit power is already on.

To calibrate the touch screen:
1. Power on the ezLCD+103.
2. Hold down PB1 button on the ezLCD+103 board (or connect the PROG# pin to the ground) for up to 2 seconds
3. Release the PB1 button (or disconnect the PROG# pin) as soon as the calibration page appears. The ezLCD+103 will switch to the Demo Mode if PB1 (or PROG#) is not deactivated in time.
4. Follow the instructions displayed on the ezLCD+103 screen

The calibration data is saved on the ezLCD+103 board Configuration EEPROM.
8 ezLCD+ Executable Files

The ezLCD+ executable files are the binary files, which can be directly executed by the ezLCD+ CPU (AT32AP7000) on the ezLCD+ platform.

The ezLCD+ executable files are used for special tasks, which cannot be performed by the ezLCD+ command set. Those include:

- Firmware upgrade
- ezLCD customization
- Loading special programs (Linux OS, etc.)
- etc.

The ezLCD+ executable files have the filename extension: ".eze". They can be executed only from the SD card during power-up when the PB1 button is pressed or PROG# input (CN4) is grounded. The .eze has to reside on the SD root directory. If the SD root directory contains more than one .eze file, only the first found is loaded and executed.

To execute the .eze file:
1. Make sure that the SD card is formatted in FAT-32, FAT-16 or FAT-12.
2. Copy the .eze file to the root directory of the SD card
3. Make sure that the copied file is the only file with the .eze extension on the SD root directory
4. Power Off the ezLCD+103
5. Insert the SD card into the ezLCD+103 SD slot
6. Press and hold down the PB1 button on the ezLCD board (or connect the PROG# pin to the ground)
7. Power On the ezLCD+103
8. After 2-3 seconds release the PB1 button (or disconnect the PROG# pin from the ground).
ezLCD+ Customization

ezLCD+103 settings, bitmaps, fonts, protocols etc. may be modified by running from the SD the ezLCD executable file User.eze.

User.eze modifies the ezLCD defaults by reading the required changes from the following files:

- **UserRom.txt** - Contains the SD paths to the files which are to be embedded into the ezLCD+103 ROM (fonts, bitmaps, scripts, etc.) The files are saved on the CPU ROM. There are 3.8 megabytes available for the user files. The syntax of this file is described in the chapter: User ROM.

- **UserConf.txt** - Contains the ezLCD+103 User Configuration (communication parameters, start-up screen, etc.) The syntax of this file is described in the chapter: User Configuration. The User Configuration is stored on the ezLCD+103 board Configuration EEPROM.

User.eze, UserRom.txt and UserConf.txt should reside on the SD root directory. Both UserRom.txt and UserConf.txt are optional.

To customize ezLCD+103:

1. Edit **UserRom.txt** and **UserConf.txt** or only one of them as needed.
2. Make sure that the SD card is formatted in FAT-32, FAT-16 or FAT-12.
3. Copy the following files to the SD card:
   - User.eze
   - UserRom.txt and the files to be embedded into ezLCD ROM, if needed.
   - UserConf.txt, if needed
4. Make sure that the User.eze is the only file with the .eze extension on the SD root directory
5. Power off the ezLCD+103
6. Insert the SD card into the ezLCD SD slot (CN9)
7. Press and hold down the PB1 button on the ezLCD+103 board (or connect the PROG# pin to the ground)
8. Power on the ezLCD+103
9. After 2-3 seconds release the PB1 button (or disconnect the PROG# pin from the ground)
10. Wait until the programming is finished. Note: this may take up to 2 minutes.
11. Recycle the ezLCD+103 power. Make sure that the PROG# is not active, otherwise the programming will be repeated.

**Warning:** Do not turn off the power while programming of the User ROM is in progress.

The progress of the operation is displayed in form of messages on the ezLCD+103. Also, the messages are logged into User.log file, if the SD is not write-protected.

Example of the generated messages:

Opening File: UserConf.txt ... OK
Set SplashScreen to None
Set BackLight to 255
Set BacklightOn to True
Set TouchProtocol to None
Set Rs232Enabled to Yes
Set Rs232Baudrate to 115200
Set Rs232StopBits to 1
Set Rs232Parity to Even
Set Rs232Handshake to None
Set SpiEnabled to Yes
Set I2cEnabled to No

Opening File: UserRom.txt ... OK
Adding SD File: /Fonts/Arial_14.ezf
Added file type: Font Bitmap no: 0 size: 3560 bytes
Adding SD File: /Fonts/Arial_14_B.ezf
Adding SD File: /Fonts/DejaVuSans.ttf
Added file type: Font True Type no: 0 size: 568408 bytes
Adding SD File: /Fonts/DejaVuSans-Bold.ttf
Adding SD File: /Bitmaps/FAVR-32_Bottom.jpg
Added file type: Bitmap no: 0 size: 120112 bytes
Adding SD File: /Bitmaps/FAVR-32_Top.jpg
Adding file type: Bitmap no: 1 size: 108481 bytes
Adding SD File: /Bitmaps/map1.jpg
Adding file type: Bitmap no: 2 size: 120225 bytes

Saving User Configuration ... OK

Programming User ROM ... OK

No of Errors: 0
9.1 User ROM

The contents of the User ROM is specified by the UserROM.txt file, which contains the SD paths to the files which are to be saved on the CPU ROM. There are 3.8 mega bytes available for the user files.

Supported file types:

<table>
<thead>
<tr>
<th>File Type</th>
<th>Supported Extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bitmaps (Icons)</td>
<td>.bmp, .jpg, .ezp</td>
</tr>
<tr>
<td>Bitmap Fonts</td>
<td>.ezf</td>
</tr>
<tr>
<td>Free Type (True Type) Fonts</td>
<td>.ttf, .otf</td>
</tr>
<tr>
<td>Scripts</td>
<td>.lua</td>
</tr>
</tbody>
</table>

An index is automatically assigned to each of the embedded files of the particular type. This index is used as an argument in ezLCD+ Commands and Lua instructions. The indexes are assigned in the order the files of the particular type appear in UserRom.txt.

UserROM.txt is read by the ezLCD+ executable: User.eze. The whole procedure is described in the chapter: ezLCD+ Customization.

UserRom.txt Syntax

- UserRom.txt consists of a series of SD paths to the files.
- UserRom.txt is an ASCII text file.
- Each text line is terminated either with <CR><LF> (DOS, Windows, etc.) or the <LF> alone (Unix, Linux, MAC OS X, etc.).
- Comments start with ‘#’ and end with the end of the line.
- Each text line can contain only one path to the SD file.

Example of the UserRom.txt file contents:

```
# +--------------+
# | BITMAP FONTS |
# +--------------+
/Fonts/Arial_14.ezf       # Bit Font 0
/Fonts/Arial_14_B.ezf     # Bit Font 1
/Fonts/Times New Roman_34_B.ezf # Bit Font 2
/Fonts/Fonte_26.ezf       # Bit Font 3
/Fonts/Script MT Bold_29_B.ezf # Bit Font 4
/Fonts/ISO_6x10.ezf       # Bit Font 5
/Fonts/ISO_8x13.ezf       # Bit Font 6
# +-----------------+
# | FREE TYPE FONTS |
# +-----------------+
/Fonts/DejaVuSans.ttf     # FT Font 0
/Fonts/DejaVuSans-Bold.ttf # FT Font 1
/Fonts/DejaVuSans-BoldOblique.ttf # FT Font 2
/Fonts/DejaVuSerif.ttf    # FT Font 3
```

© 2008 Earth Computers Tech. Inc.
/Fonts/DejaVuSerif-Bold.ttf  # FT Font 4
/Fonts/DejaVuSerif-Italic.ttf  # FT Font 5
/Fonts/Quig.ttf  # FT Font 6

# +---------+
# | BITMAPS |
# +---------+
/Bitmaps/FAVR-32_Bottom.jpg  # Icon 0
/Bitmaps/FAVR-32_Top.jpg  # Icon 1
/Bitmaps/map1.bmp  # Icon 2
/Bitmaps/map2.ezp  # Icon 3
9.2 User Configuration

The User Configuration is used to modify some of the ezLCD+103 default parameters like:
- Communication parameters (RS232 baudrate, parity, IP address, etc.)
- Startup screen
- Startup backlight
- Pin functions

Upon power-up the ezLCD+103 CPU configures the ezLCD+103 according to the data read from the User Configuration. The User Configuration is stored on the ezLCD+103 board Configuration EEPROM.

The User Configuration can be modified, by editing the UserConf.txt file. The UserConf.txt file is read by the ezLCD+ executable: User.eze. The whole procedure is described in the chapter: ezLCD+ Customization.

UserConf.txt Syntax

The philosophy of UserConf.txt is similar to the one used in the Windows .ini files. It consists of the set of keys and values assigned to them. The keys are described in the section: User Configuration Keys.

- UserConf.txt is an ASCII text file.
- Each text line is terminated either with <CR><LF> (DOS, Windows, etc.) or the <LF> alone (Unix, Linux, MAC OS X, etc.).
- Comments start with '#' and end with the end of the line.
- Each text line can contain only one key assignment.
- Leading and trailing spaces are ignored.
- Keys and their values are separated by any combination of the following characters: '=' or ':', '<SPACE>', '<TAB>'.
- Keys and their values are case insensitive.
- Values maybe specified in the following formats:
  - decimal
  - hexadecimal: 0x followed by the number, for example: 0x2D0
  - binary: 0b followed by the number, for example: 0b00103103
  - boolean: True, False or Yes, No or On, Off or Enabled, Disabled
  - IP address: four decimal numbers (0 to 255) separated by dots, for example: 192.168.1.6
  - Special values: Even, Odd, hw, h/w, sw, s/w, Xon/Xoff, USB, Rs232, SPI, I2C, Net, Mac, Ethernet, Wnet, Wireless, Automatic, DHCP, ezButton, cuButton, CalibratedXY

Note: Only the key values specified in the UserConf.txt are modified. Others are left unchanged.
Examples of the correct key assignments:
Rs232Baudrate = 115200
Rs232Baudrate: 115200
Rs232Baudrate 115200
Rs232Baudrate=115200
Rs232Parity: None
Rs232Parity = Even
SpiEnabled = True
SpiEnabled Enabled
NetIpAddress = 192.168.1.6
NetIpAddress = DHCP
NetIpAddress = Automatic
StartupTxInterface = SPI
9.2.1 User Configuration Keys

Key: BackLight
Description: Power-up backlight brightness
Value Type: Number (decimal, hexadecimal, or binary)
Value Range: 0 to 255
Example: BackLight = 255

Key: BacklightOn
Description: Power-up backlight state
Value Type: Boolean
Example: BacklightOn = On

Key: DacEnabled
Description: Enable or disable stereo audio
Value Type: Boolean
Example: DacEnabled = No

Key: DemoInterval
Description: Interval in seconds between screens, when the ezLCD is in the Demo Mode
Value Type: Number (decimal, hexadecimal, or binary)
Value Range: 0 to 4,294,967
Example: DemoInterval = 2

Key: Demos
Description: Specifies which demos should be shown in the Demo Mode. Bits 0 to 4 of the key value specify, which demos should be shown:
- Bit 0: System Info
- Bit 1: SD slide demo
- Bit 2: User ROM icons slide demo
- Bit 3: Free Type (True Type) Font demo
Bit 4: Bitmap Font demo

**Value Type:** Number (decimal, hexadecimal, or binary)

**Value Range:** 0 to 31

**Example:** Demos = 0b10111 # Show all of the demos except the Bitmap Font

---

**Key:** I2cEnabled

**Description:** Enable or disable I2C interface

**Value Type:** Boolean

**Example:** I2cEnabled = No

---

**Key:** MacAddressHi

**Description:** The highest 16 bits of the 48 bit MAC Address (Ethernet Hardware Address)

**Value Type:** Number (decimal, hexadecimal, or binary)

**Value Range:** 0 to 0xFFFF

**Example:** MacAddressHi = 0x181F

---

**Key:** MacAddressLo

**Description:** The lowest 32 bits of the 48 bit MAC Address (Ethernet Hardware Address)

**Value Type:** Number (decimal, hexadecimal, or binary)

**Value Range:** 0 to 0xFFFFFFFF

**Example:** MacAddressLo = 0x008C0001

---

**Key:** MacEnabled

**Description:** Enable or disable Ethernet interface

**Value Type:** Boolean

**Example:** MacEnabled = True
Key: NetAltDns
Description: IP Address of the Alternate DNS
Value Type: IP Address, or Special Value
Special Values: None, Automatic, DHCP. All of them are equivalent.
Example: NetAltDns = None

Key: NetDns
Description: IP Address of the DNS
Value Type: IP Address, or Special Value
Special Values: None, Automatic, DHCP. All of them are equivalent.
Example: NetDns = DHCP

Key: NetGateway
Description: IP Address of the default Gateway
Value Type: IP Address, or Special Value
Special Values: None, Automatic, DHCP. All of them are equivalent.
Example: NetGateway = 192.168.1.1

Key: NetIpAddress
Description: IP Address of the ezLCD
Value Type: IP Address, or Special Value
Special Values: None, Automatic, DHCP. All of them are equivalent.
Example: NetIpAddress = 192.168.1.6
Key: NetSubnetMask
Description: Subnet Mask
Value Type: IP Address, or Special Value
Special Values: None, Automatic, DHCP. All of them are equivalent.
Example: NetSubnetMask = 255.255.255.0

Key: ParEnabled
Description: Enable or disable Parallel Interface (provision)
Value Type: Boolean
Example: ParEnabled = No

Keys: Pin_00_Funct
      Pin_01_Funct
      Pin_02_Funct
      Pin_03_Funct
      Pin_04_Funct
      Pin_05_Funct
      Pin_06_Funct
      Pin_07_Funct
      Pin_08_Funct
      Pin_09_Funct
      Pin_10_Funct
Description: Configurable pins functions
Value Type: Number (decimal, hexadecimal, or binary) or a Special Value
Value Range: TBD
Special Values: None
Example: Pin_00_Funct: None

Key: Rs232Baudrate
Description: The baud rate in bit/s of the RS-232 interfaces
Value Type: Number (decimal, hexadecimal, or binary)
Value Range: 9,600 to 562,500
Example: Rs232Baudrate = 115200

Key: Rs232Enabled
Description: Enable or disable RS-232 interfaces
Value Type: Boolean
Example: Rs232Enabled = Yes

Key: Rs232Handshake
Description: Handshake specification of the RS-232 interfaces
Value Type: Special Value
Special Values:
- None - no handshake
- hw, h/w - hardware handshake (RTS and CTS)
- sw, s/w, Xon/Xoff - software handshake (xon/xoff)
Example: Rs232Handshake = None

Key: Rs232Parity
Description: The parity bit logic of the RS-232 interfaces
Value Type: Special Value
Special Values:
- None - no parity bit
- Even - even parity bit
- Odd - odd parity bit
Example: Rs232Parity = None

Key: Rs232StopBits
Description: Number of stop bits of the RS-232 interfaces
Value Type: Number (decimal, hexadecimal, or binary)
Value Range: 1 or 2
Example: \( \text{Rs232StopBits} = 1 \)

**Key:** Rs232ReadyOn (available starting at firmware ver:2.20)

**Description:** Used only for RS232 handshake (if enabled). Specifies the number of bytes in the input data buffer, at which RTS is reset to Lo or Xon is sent by the ezLCD+.

**Value Type:** Number (decimal, hexadecimal, or binary) or Special Value

**Special Values:** None - use the default value (0)

**Value Range:** 0 to 983,040 (0xF0000) or None. Internally rounded down to the closest multiplication of 4. Should be lower than RS232ReadyOff. Otherwise the default value will be used.

**Example:** \( \text{Rs232ReadyOn} = 16 \)

**Key:** Rs232ReadyOff (available starting at firmware ver:2.20)

**Description:** Used only for RS232 handshake (if enabled). Specifies the number of bytes in the input data buffer, at which RTS is set to Hi or Xoff is sent by the ezLCD+.

**Value Type:** Number (decimal, hexadecimal, or binary) or Special Value

**Special Values:** None - use the default value (983,040)

**Value Range:** 0 to 983,040 (0xF0000) or None. Internally rounded down to the closest multiplication of 4. Should be higher than RS232ReadyOn. Otherwise the default value will be used.

**Example:** \( \text{Rs232ReadyOff} = 128 \)

**Key:** SerialNo

**Description:** User Device Serial Number

**Value Type:** Number (decimal, hexadecimal, or binary) or

**Value Range:** 0 to 4,294,96,294. Invalid Serial Number: 4,294,96,295 (FFFFFFFfHex)

**Example:** SerialNo = 1234

**Key:** SpiEnabled

**Description:** Enable or disable SPI interface

**Value Type:** Boolean
Example: \( \text{SpiEnabled} = \text{Enabled} \)

Key: SplashScreen

Description: Power-up screen image. The value specifies the index of the User ROM bitmap to be displayed upon power-up

Value Type: Number (decimal, hexadecimal, or binary), or Special Value

Special Values: None - default screen with ezLCD info in the bottom

Value Range: 0 to 65535 or None

Example: SplashScreen = None

Key: StartupTxInterface

Description: Specifies through which interface the EZLCD_READY byte (0xEA) will be sent by the ezLCD when it is ready to process commands upon the power-up. The EZLCD_READY byte (0xEA) is sent only once and only after the power-up. If the TouchProtocol key is set to cuButton or CalibratedXY, StartupTxInterface also specifies through which interface the touch screen data is sent upon the power-up.

Value Type: Special Value

Special Values: None - ezLCD will not transmit 0xEA upon the power-up
USB
RS232
SPI
I2C

Example: StartupTxInterface = RS232

Key: StartUpLua (available starting at firmware ver:2.20)

Description: Lua program executed at Power-up. The value specifies the index of the User ROM Lua program to be executed upon power-up

Value Type: Number (decimal, hexadecimal, or binary), or Special Value

Special Values: None - default screen with ezLCD info in the bottom

Value Range: 0 to 65535 or None

Example: StartupLua = None
Key: TouchProtocol

Description: Specifies the start-up touch protocol

Value Type: Special Value

Special Values: None - touch screen data is not sent
ezButton
cuButton
CalibratedXY

Example: TouchProtocol = None

Example of the UserConf.txt file contents:
  SerialNo = 0x1234
  SplashScreen = None
  BackLight = 255
  BacklightOn = True
  TouchProtocol = None
  StartupTxInterface = None
  StartupLua = None
  Rs232Enabled = Yes
  Rs232Baudrate = 115200
  Rs232StopBits = 1
  Rs232Parity = None
  Rs232Handshake = None
  Rs232ReadyOn = None
  Rs232ReadyOff = None
  SpiEnabled = Yes
  I2cEnabled = No
  Demos = 0b11111
  DemoInterval = 5
10  **Demo Mode**

While in Demo Mode, the ezLCD displays some demonstration screens.

The ezLCD+103 enters Demo Mode when:
1. The ezLCD+103 power is on, and
2. PB1 button is held down (or the PROG# pin is connected to ground) for more than 2 seconds

To exit Demo Mode:
1. Make sure that PROG# is deactivated
2. Double-tap on the touch screen

In Demo Mode each screen is displayed for at least 3 seconds. This can be changed by modifying the User Configuration Key: DemoInterval

Any of the displayed screens can be frozen on display by pressing the touch screen in any place. The screen will be displayed as long as the touch screen is held pressed.

The following demos can be shown:
- ezLCD+ System Info. Displays installed interfaces, communication parameters, User ROM summary, etc.
- Bitmap Font Demo. Displays Bitmap Fonts installed in the User ROM.
- Free Type (True Type) Font Demo. Displays True Type Fonts installed in the User ROM.
- SD Slide Demo. Displays .bmp, .jpg and .ezp pictures from the SD root directory.
- User ROM Slide Demo. Displays bitmaps installed in the User ROM.

The demos for display can be selected by modifying the User Configuration Key: Demos
## 11 GLOSSARY

**Configuration Keys**  
Part of ezLCD+ Customization. Set of text words and values assigned to them. They are specifying the *User Configuration*. Similar, in concept, to the keys used in Windows .ini files.  
Described in the “ezLCD+10x Manual”.

**ezLCD+ Customization**  
Modification of the default power-up parameters. Addition of custom fonts, bitmaps, Lua programs, etc.  
Described in the “ezLCD+10x Manual”.

**Firmware**  
Operating software of the ezLCD+103. Can be in-field upgraded.  
Described in the “ezLCD+10x Manual”.

**Lua**  
Powerful, fast, light-weight, embeddable scripting language. By embedding Lua interpreter, the ezLCD+ become a true independent system (computer), which does not need any external host to drive it.  
Described in the “ezLCD+ Lua API Manual”.

**User Configuration**  
Part of ezLCD+ Customization. Modifies some of the ezLCD+ default parameters like: communication parameters, start-up screen, etc. Upon the power-up the ezLCD+ CPU configures the ezLCD+ according to the data read from the User Configuration.  
Described in the “ezLCD+10x Manual”.

**User ROM**  
Part of the ezLCD+ Customization. A place in the ezLCD+ flash, where user can store custom fonts, bitmaps, Lua programs, etc.  
Described in the “ezLCD+10x Manual”.

© 2008 Earth Computers Tech. Inc.