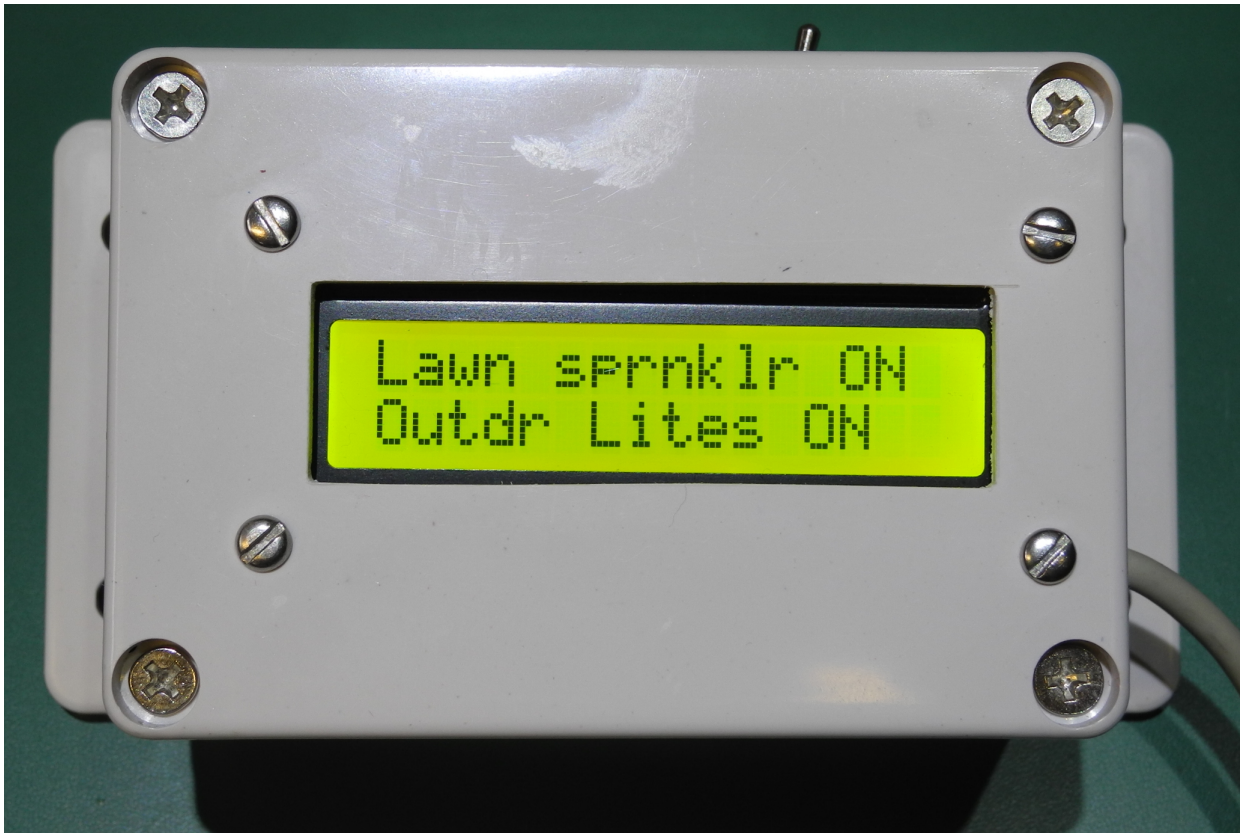
An LCD Annunciator Project



The LCD Annunciator Project describes an LCD messaging system designed to work with the uTile PLC. The annunciator uses a 2 line by 16 character LCD display to selectively display any two messages from a pre-loaded set of 8 messages stored on the annunciator. The annunciator is controlled from a uTile PLC over a serial RS-485 communications link. The annunciator is a stand-alone unit that can be remotely located from the uTile PLC by up to 300 m.

Date: 2016 October 29

Table of Contents

1 Introduction

This project describes an LCD annunciator unit for the uTile PLC. The annunciator is controlled from the uTile by the **LANN** command word and two 8-bit output port bytes **PM** and **PN**, with the associated bit write command words **.Mn** and **.Nn** (where n = 0,..,7). uTile communicates with the annunciator over a uni-directional RS485 serial interface. The annunciator can be located up t0 300 m away from the uTile controller. The annunciator is powered by an externally supplied +7 to 12 V(dc) power supply.

The annuncaitor parts list includes an Arduino Nano board (ATmega328P chip), a 2 line by 16 character LCD display with an attached Backpack TWI interface board, and an RS-485 transceiver chip. All the hardware components used for this project are readily available from numerous sellers on ebay and other on-line retailers.

The software for the LCD annunciator is provided as an executable file (m328-LCD-message.hex), ready for programming into the ATmega328P chip's flash program memory.

The Arduino Nano board is used in this design because of its small footprint and relatively low cost. The Nano board's built-in USB port is used to load up to 8 messages into the annunciator's non-volatile EEPROM memory via the annunciator's built-in message editor.

The uTile PLC master controls the slave annunciator annunciator by transmitting simple command text strings by which the annunciator determines which pre-stored message to display. The annunciator commands directs to which of the two LCD lines the message should be displayed.

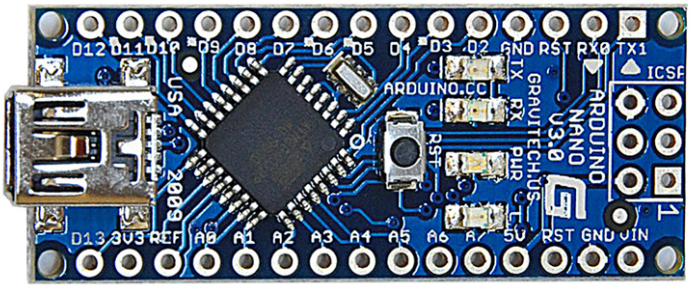
An annunciator message is displayed on the LCD when data is written from the bit stack using one of the **.Mn** and **.Nn** commands. The message is displayed on the detected rising edge of the stack bit, and erased on the falling edge of the same bit.

2 Parts list

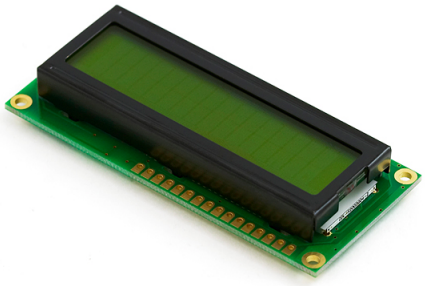
You need the following parts for the clock-timer project:

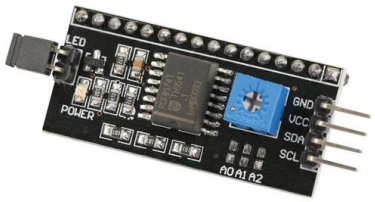
1. Arduino Nano controller board, 16 MHz clock, with ATmega328P microcontroller chip.
2. A 2line by 16 character LCD siplay module, type JHD162, with an attached Backpack TWI interface module.
3. An SN75176B RS-485 transceiver chip. Only the receiver side of the transceiver is used for the LCD annunciator project. Can also use the MAX485 transceiver chip in place of the SN75176B.
4. An external 7 to 12 V (dc) power supply (recommended) power supplied to the Nano's VIN pad powers the annunciator unit.
5. Miscellaneous hardware including 28 AWG hook-up wire, pin headers, 50 Ohm resistor, plastic project case, screws & nuts, spacers etc.

1. RS-485 interface module for use with uTile running on either the Nano or on the Mega2560 controller board.



The Arduino Nano controller board





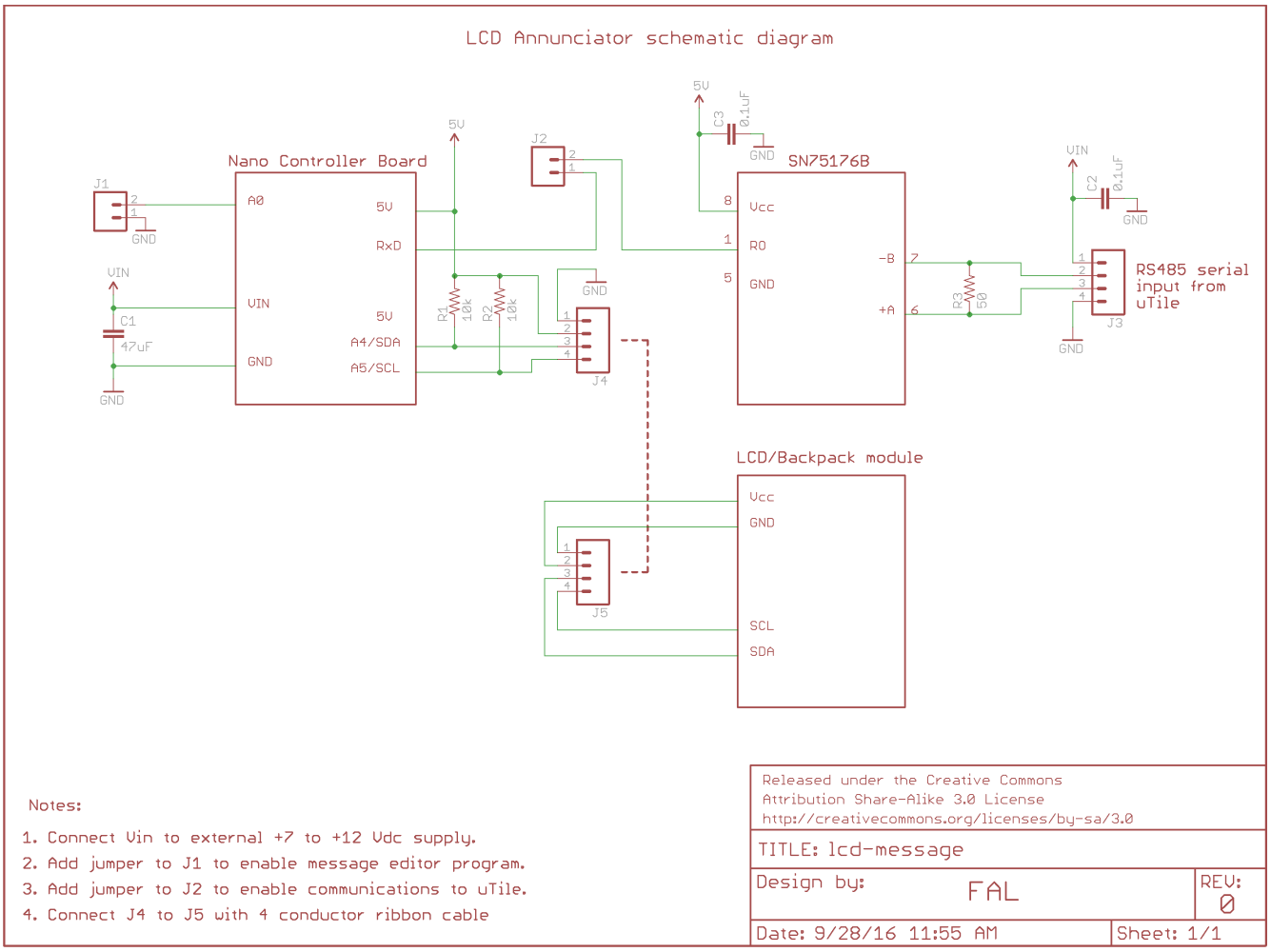
LCD Display and Backpack TWI module

RS-485 Interface module

3 Putting the project together

The schematic diagram below shows the circuit for the LCD annunciator project.

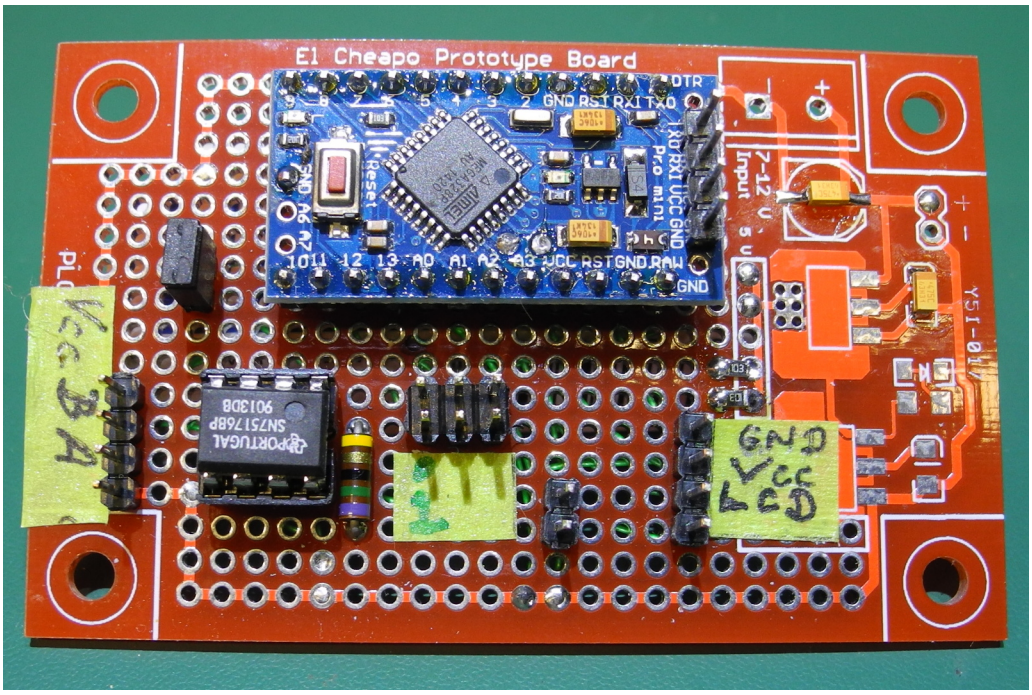
The annunciator project uses an Arduino Nano controller board because of its relatively low cost. All the required support functions are on the board, including the ISP header, USB serial port, and +5 Vdc power for the LCD and backpack module and the SN75176B transceiver chip.



Note that the prototype annunciator uses a Pro-mini board instead of the Nano board. The prototype pictures show the Pro-mini board, but the annunciator program works on Nano without modification.

3.1 Assembly steps

The Nano board and the SN75176B IC are mounted on a small prototyping circuit board. You can see the arrangement in the picture below.

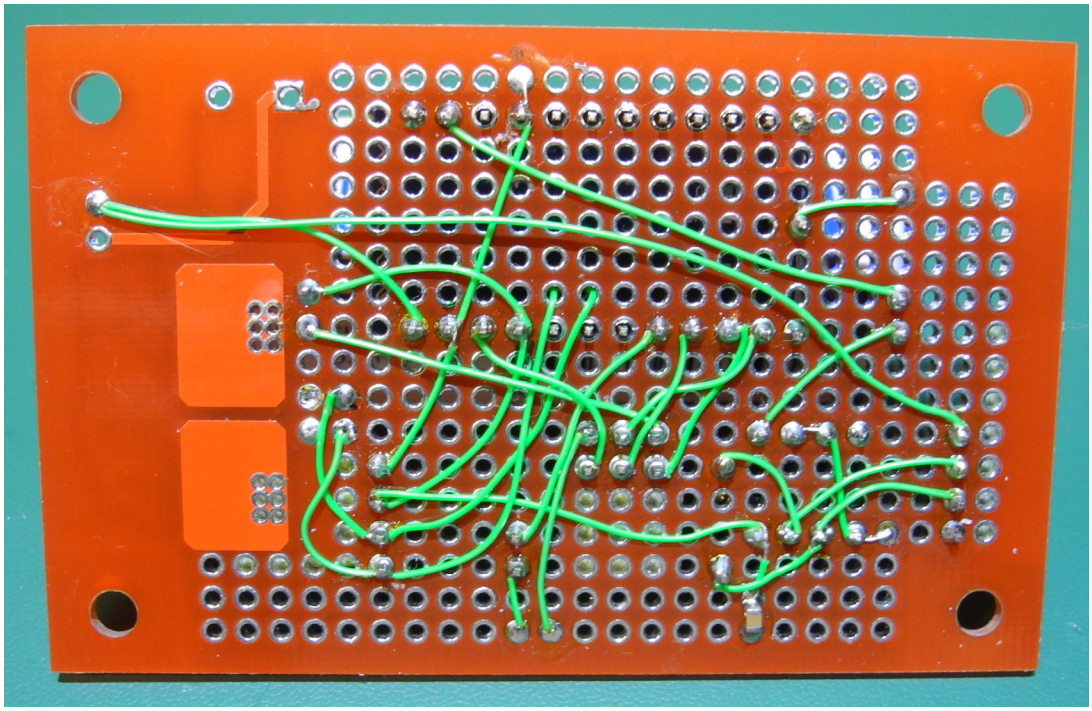


Annunciator board top view - parts layout

The 4-pin strip header on the bottom left corner is for the connections to the annunciator interface wiring to Vcc, ground and the A and B terminals of the RS-485 serial connections. Just aboove and to the right is the 2-pin header J2 with shorting jumper shown installed.

The 4-pin strip header on the bottom right corner facilitates connections to the mating 4 pin header on the LCD/backpack module, which is located away from the circuit board. A four-conductor ribbon jumper cable connects the LCD display to the Nano controller on the circuit board. The 2-pin header J1 is just to the left of the LCD header.

The 6-pin strip header in the bottom middle of the board is the ICSP programming header for attaching the USB-ASP programmer to the Pro-mini board. The Nano board already provides this header.

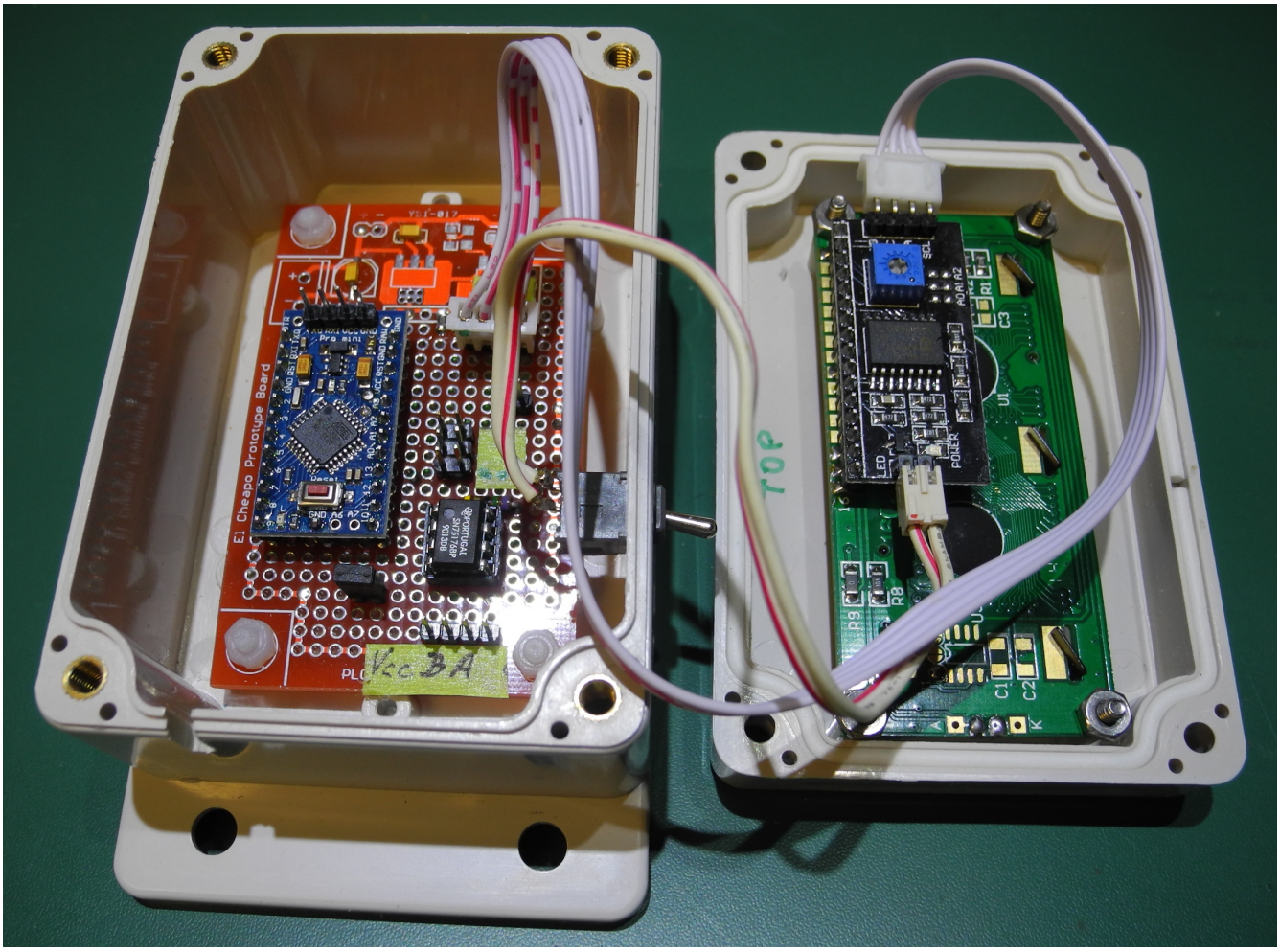


Annunciator board bottom view - wiring

30 AWG kynar insulated wire-wrap wire is used for wiring the connections on the circuit board.

Two 2 pin headers, J1 and J2 accept shorting jumpers that are used to configure the annunciator for operation either for the message editor sub-program or for the annunciator normal operation. For message editor operation add the J1 jumper and remove the J2 jumper. After the messages are loaded and saved to EEPROM, remove the J1 jumper and add the J2 jumper to put the annunciator in the normal operating mode.

The completed prototype annunciator assembly is shown in the picture below mounted in a 2.5 x 4 x 2 inch plastic utility case with ample room. You can see the ribbon cable for the LCD/Backpack connections. The grey toggle switch is for manual control of the LCD backlight LED.



Annunciator inside view

When you have completed the wiring of the annunciator circuit board, carefully check the wiring against the schematic diagram again and make sure there are no wiring errors. Look especially at the +5V power and ground wiring as inadvertent short circuits or reversed polarity connections are some of the easient ways to cause a catastrophic “smoke” event.

Do not apply power to the Nano board as yet. At this stage you are ready to program the m328-LCD-message.hex executable file you downloaded from the PLCCHIPS.com web site to the Nano's flash memory. The programming method is described in the next section.

3.2 Burning the m328-LCD-message.hex program.

The programming method used to program the LCD annunciator executable file to the Nano's flash memory uses the AVRDude programming software and a USB-ASP-programmer avaialable from many on-line retailers. The Arduino IDE is not used in this method of programming, nor is the boot-loader supplied with the Arduino Nano board.

First, download and install on your PC the AVRDude software. There are numerous excellent articles on the web describing how to install and configure AVRdude so we'll skip repeating the same thing here. If you have trouble finding the software, take a look at this site:

http://winavr.sourceforge.net/download.html

The tricky part to programming the Nano board with AVRDude is to invoke the proper command line instruction. We'll get to this in a moment.

First, you need a USB-ASP-Programmer, a typical one is shown in the picture below. You should also get a 10-pin socket to 6-pin plug adapter to make it easier to connect the 10-pin ribbon cable from the programmer to the 6 pin ISP header on the Nano board. The USB programmer is recognized by AVRDude as programmer type usbasp. See the following site for further details of this programmer.

http://www.fischl.de/usbasp/

We'll assume you are running in a Windows environment and that you placed the m328-LCD-message.hex program image file in a sub-directory such as:

c:\Users\yourname\My Documents\avr\m328-LCD-message.hex

When you invoke AVRDude be sure to specify the full path to the directory where avrdude.exe resides. We'll asume for this discussion that the avrdude files are in:

c:\opt\avr\avrdude.exe

The avrdude.conf and avrdude.rc files are also placed in this sub-directory.

When you are ready to program the Nano's flash with m328-LCD-message.hex executable image, plug the programmer into a USB port on the PC and plug the ribbon cable with the 10-to-6 pin adapter onto the 6-pin ISP programming header on the Nano board, making sure you connect the adapter the right way round (adapter socket 1 connected to ISP header pin 1). The programmer will supply 5 V power to the Nano board and the power indicator LED will light up.

Open up a command line interface in a Windows terminal and go to the subdirectory c:\Users\yourname\My Documents\avr\ where the m328-LCD-message.hex file is located. Type in the following command:

"c:\opt\avr\avrdude.exe" -C"c:\opt\avr\avrdude.conf" -p m328 -c usbasp -u -U flash:w:m328-LCD-message.hex:i

AVRDude will perform a number of programming steps and print some messages on the console screen while it uploads the m328-LCD-message.hex file to the Nano's flash memory. If the programming is successful, you'll see the program verification step completed.

As a side note to AVRDude use, a very useful companion program to have in your toolbox is the AVRdudess, a GUI for AVRDude. Check out this link if you are interested:

http://blog.zakkemble.co.uk/avrdudess-a-gui-for-avrdude/

When you are done programming, unplug the programmer from the Nano. Now go to the final step of powering up the LCD annunciator project in the following section.

4 Configuring the LCD annunciator

The first time the LCD annunciator is powered up there are no messages stored in the EEPROM memory as the EERPOM is erased by the flash memory programming operation. You need to enter and store up to eight annunciator text messages into the annunciator and then to save them in the non-volatile EEPROM memory.

You enter/edit and save messages to EEPROM by invoking the message editor sub-program of the LCD annunciator. Once you are inside the message editor program your interface to the editor is via a terminal emulator program running on a host PC and connected to the LCD annunciator via a USB cable.

4.1 Invoking the message editor

The message editor sub-program is a special program within the LCD annunciator that is invoked when you need to enter new messages or edit existing messages in the LCD annunciator.

The following procedure places the LCD annunciator into the message editor operating mode:

1. Power down the LCD annunciator. You do not need to disconnect the RS-485 serial line to the uTile controller if it is already connected.
2. Add the shorting jumper across the two J1 pins. The LCD annunciator will read this condition on a power-up or reset and will start the message editor sub-program.
3. Remove the shorting jumper across the two J2 pins. This disconnects the RS-485 Rx signal from the host uTile controller and allows the Nano's USB serial interface to operate. There is only one UART on the Nano and it has to be shared with the USB serial interface and with the RS-485 link to the host controller.
4. Connect a USB cable between the LCD annunciator Nano controller board and a host PC. When this connection is made, the LCD annunciator will be powered up via this USB cable and will enter the message editor sub-program operating mode.

4.2 Entering annunciator messages

The message editor sub-program requires a terminal emulator program running on a host PC (Windows based) to act as a terminal interface for the editor program. A good program to use is PuTTY, an open source SSH and Telnet client available from:

http://www.putty.org/

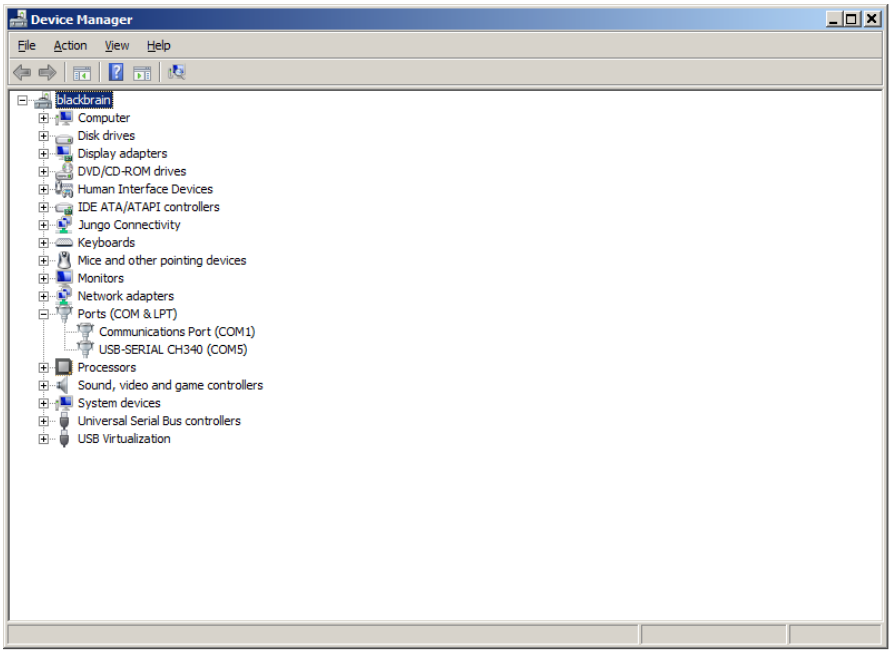
Download and install PuTTY on your PC and go through the configuration procedure described in the following section to allow PuTTY to be used with the message editor program.

4.3 PuTTY Configuration

Connect the Arduino Nano board to a PC USB port for the set-up procedure.

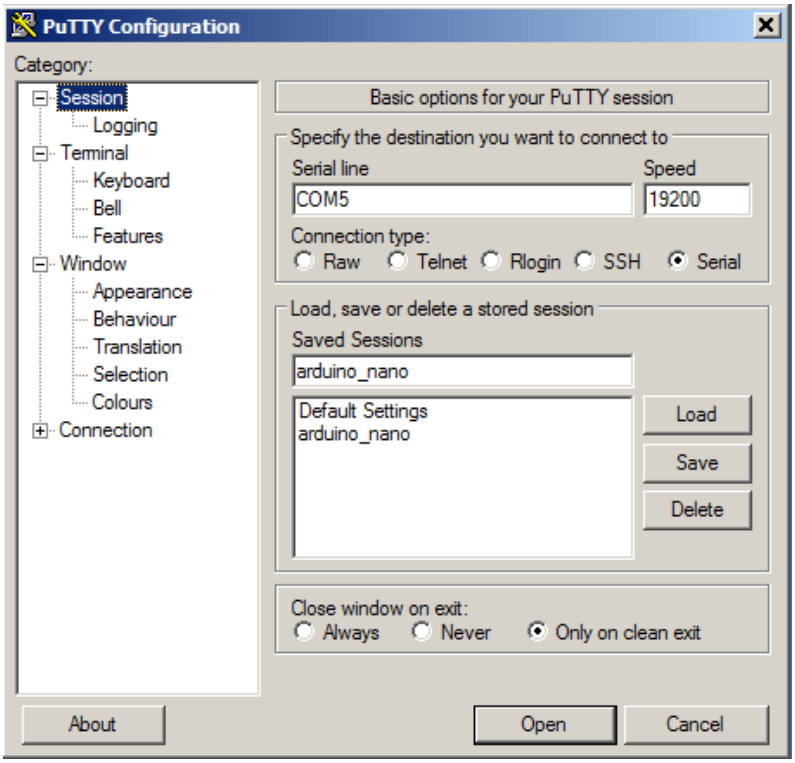
The first thing to do is to install the required USB device driver for the Nano controller board. You need to get and install the correct driver from the Arduino web site. Check other popular Arduino web sites like Lady Ada, Sparkfun, etc. for advice on setting up the Arduino board USB serial port driver.

You now need to determine which serial COM port the OS assigned to the plugged-in controller board. To find out which COM port is assigned, open the Windows Control Panel, select Hardware and Sound, and select Device Manager. You should see a screen similar to the following:



Under the Ports (COM & LPT) folder you see USB-SERIAL CH340 (COM5) listed, which identifies COM5 as the assigned port for the Arduino Nano board.

Start PuTTY and the configuration screen appears. Modify the settings as shown in the screen below:



Under the Category selection on the left pane, Connection, SSH, open the screen that shows the serial settings, and for the Flow control box select none. Go back to the Session screen and save the configuration used by providing a session name to use (arduino_nano used in this example) and clicking on the Save button.

Open the PuTTY terminal screen by clicking on the Open button. If the lcd-annunciator is operating correctly, you should see the sign-on screen shown below appear on the terminal screen.

You see 8 rows of message lines, all blank, with the first message on line 00 in reverse highlight to indicate this is the line being edited.

Editor's cursor appears at the bottom of the screen. Above the cursor is a ruler line with sixteen marks to show the length of the message. Navigating between messages is straightforward, use the Up <Up> or Down <Dn> cursor keys to move the highlighted line up or down. Or you can just enter a blank line (Carriage Return <CR> key without any other characters) to move to the next line.

Enter up to eight lines of messages by typing in the text, ending a line with a <CR>. Enter another <CR> to move to the next message line. When you are done entering the messages, the screen shows you what you have entered so far. as shown in the next example screen below:

The messages that you entered and are now displayed on the screen are stored in the controller's volatile system RAM. To permanently save the text you need to store the data in the non-volatile EEPROM memory. You do this by entering a Ctrl S (press the Ctrl key and the 'S' key at the same time). The following screen appears:

Enter a 'Y' to save the messages to EEPROM, or a <CR> if you decide not to save the messages at this time.

If you save the messages, the next time you invoke the editor program, the saved messages will be displayed. You can overwrite an existing message with a new one by simply entering the new text on the selected message line, then saving the messages again.

This concludes the message editor operation. Reconfigure the lcd annunciator to the normal operating annunciator mode by powering down the unit, removing the jumper on J1 and replacing the jumper on J2. Connect the lcd annunciator RS-485 serial communications lines to the uTile controller, and connect the wires to the external power supply for the annunciator. The annunciator is now set up for normal operation under control of an external master host uTile controller.

5 Operating the LCD annunciator

The LCD annunciator is designed to be controlled by a uTile controller over an RS-485 serial connection using a single twisted pair cable. An external +7 to +12 V(dc) power and ground provides power to the LCD annunciator unit.

The Mega2560 or Nano controller board versions of uTile support the LCD annunciator and has the new **LANN**, **.Mn** and **.Nn** annunciator control words incorporated into the control word dictionary.

You need to connect an RS-485 interface module to the Mega2560 or to the Nano controller board as described in the next section, then conect the wiring for the serial RS-485 link and external power supply to the annunciator unit to complete the LCD annunciator system wiring.

The RS-485 interface module connections vary slightly depending on which controller board version of uTile is being used. The Mega2560 controller board has three USART channels available. Channel 1 USART is used for communications to the LCD annunciator, and Tx1 is wired directly to the RS-485 module's DI input.

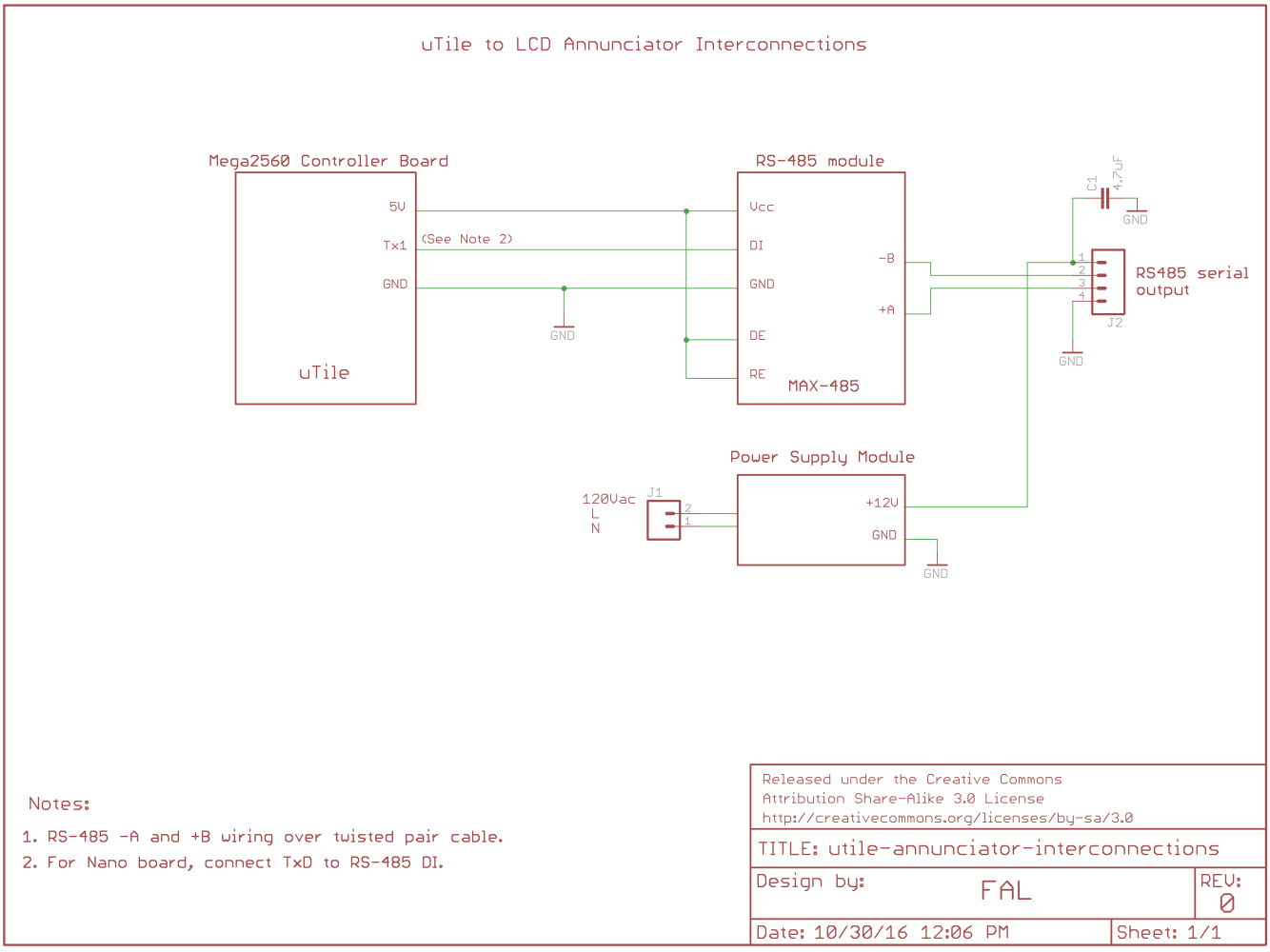
The Nano controller board has only one hardware USART so the TxD output must be shared between the USB serial port and the RS-485 module's DI input. For the Nano controller, wire the TxD pin to the RS-485 module's DI pin.

5.1 The uTile RS-485 interface

The schematic diagram below shows the required connections between a uTile controller board (Mega2560 or Nano) and the RS-485 module, Section 2, Parts list

Don't forget to wire the RS-485 module's DE and RE pins to +5V to enable the transmitter section while at the same time disabling the receiver section of the transceiver.

Note too the TX wiring differences between the Nano and Mega controller boards



There are four wiring connections between the LCD annunciator and the uTile controller, and the external power supply (usually located close to the uTile controller). A single CAT-5 cable can be used for all the wiring interconnections.

5.2 The uTile – LCD Annunciator commands

uTile for the Mega2560 and Nano controller boards includes the commands for controlling the LCD annunciator. From a programming perspective, the Mega2560 and Nano uTile programming is identical. The listings shown here are for the Mega2560 but they apply equally to the Nano board as well.

The command word to invoke the operation of the LCD annunciator is the **LANN** command. Just include **LANN** in your program listing to enable the annunciator.

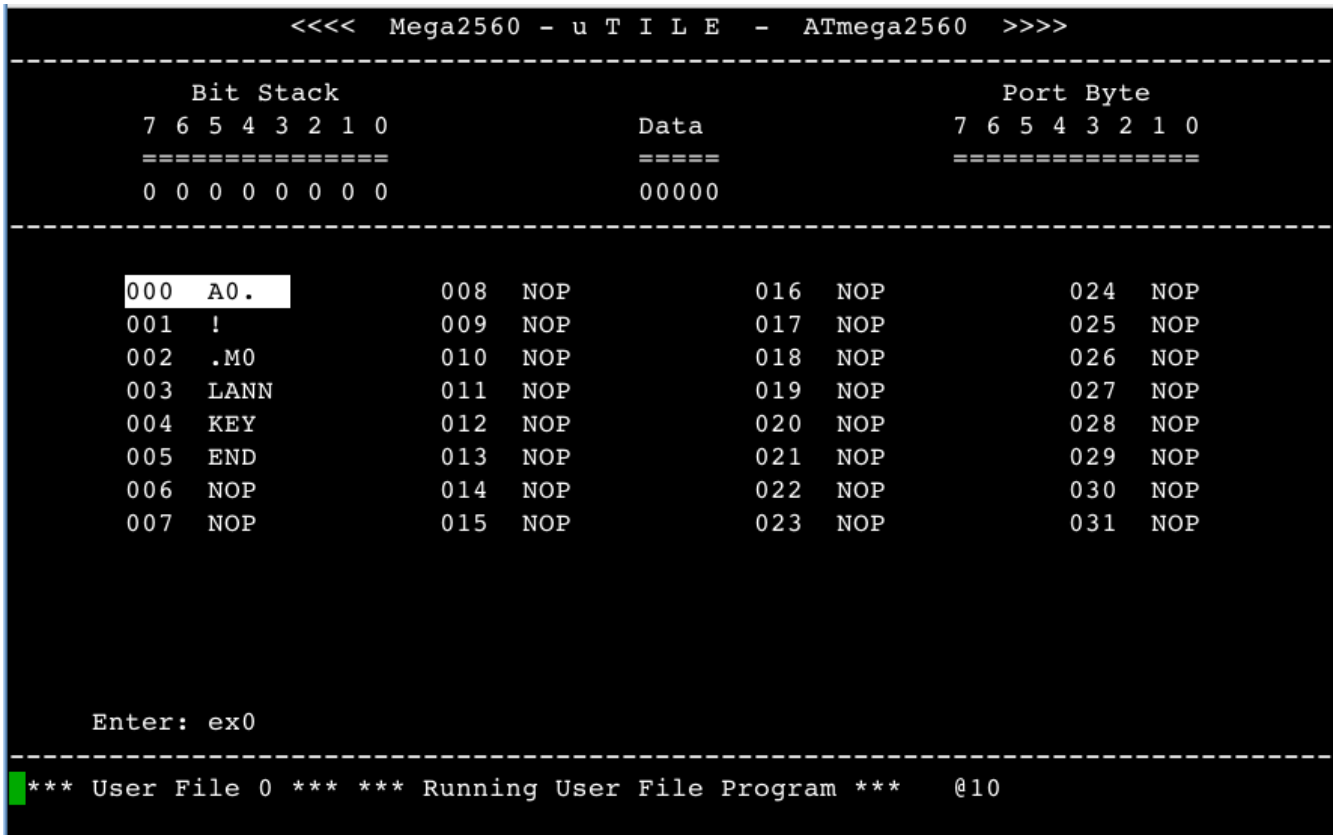
Each line of the annunciator is controlled by one 8-bit virtual port byte. Line 1 is controlled by output virtual port byte **PM**, while line 2 is controlled by output virtual port byte **PN**. Each of these port bytes have individual bit write commands **.Mn** and **.Nn**, where n = 7,..,0. The **n** selects which of the 8 stored message is displayed on the LCD. For example to print out stored message number 1 on the annunciator line 2, simply write a result using the **.N1** command. Similarly, to print out stored message number 0 on the annunciator line 1, simply write a result using the **.M0** command.

The **.Mn** and **.Nn** commands respond to the leading and trailing edge transitions of the data written to the bits and are insensitive to the static levels of the data written to the bits. When a rising level is sensed, uTile sends a command to the annunciator to write the corresponding message on the selected LCD line. The message remains on the LCD until a falling level transition is detected on the data bit, then the message is erased.

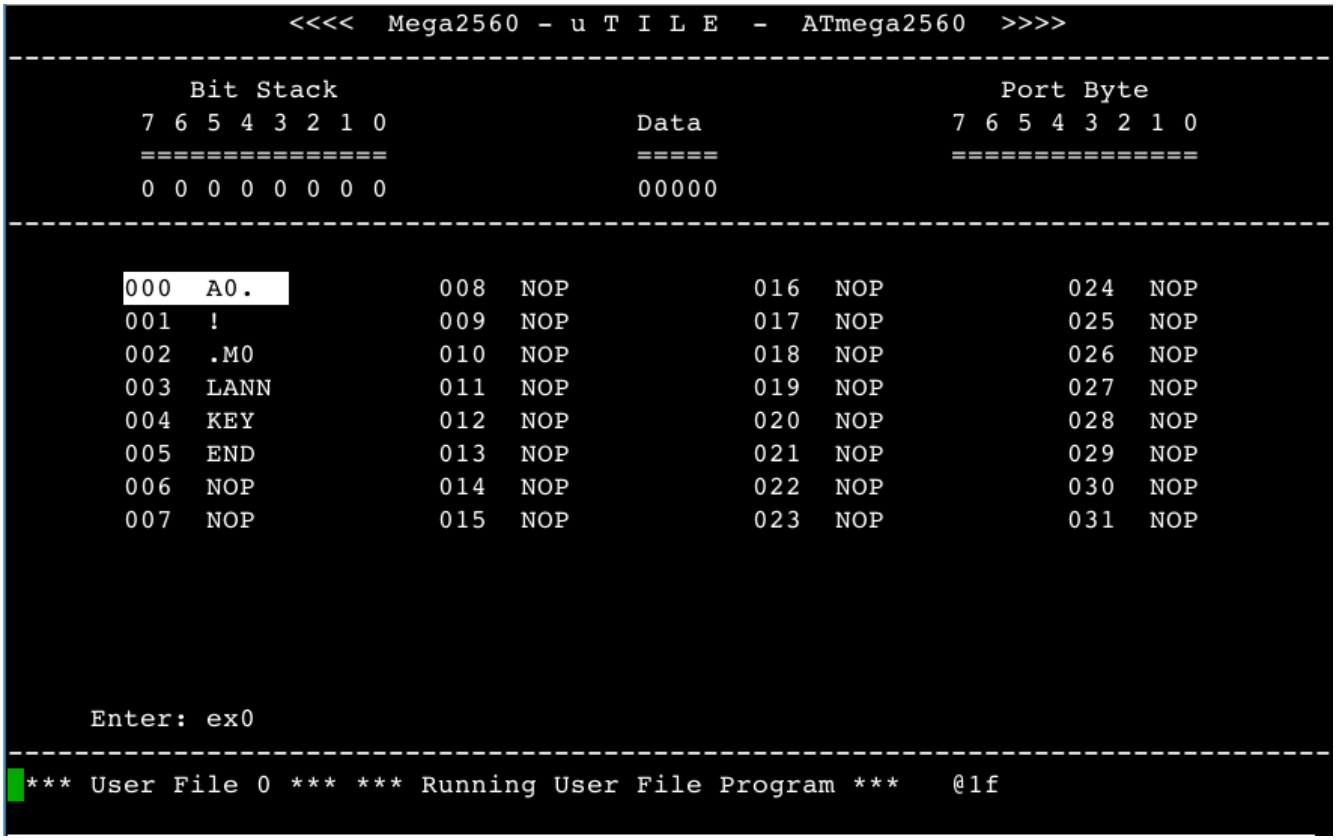
If your user file program prints one message on the annunciator, then prints a second message of the same line of the annunciator while the first message is still active, the second message will overwrite the first message.

The uTile program listing below shows how to display message 0 on line 1 of the LCD annunciator.

The input bit used to control the message is **A0**, which is read to the bit stack by the **A0.** command. The **!** command inverts the input bit value on the stack, so the bit goes to 1 whenever the input is grounded. The **.M0** command writes the stack bit to the LCD annunciator, telling it to write the stored annunciator message 0 to line 1 of the LCD display.



The **LANN** command processes the results from the PM and PN related commands and sends the proper control strings to the annunciator display. The last line of the above screen shot shows what happens when input A0 is grounded. Notice the '@10' string at the end of the line. This is the text string the is sent to the LCD annunciator when uTile recognises a rising edge on the **M0** bit (falling edge on the **A0** input). The next screen shot below shows that a command string '@1f' is sent to the LCD annunciator to erase the message when the M0 bit falls to 0.



The command string consists of a three ASCII characters, an @ character prefix, followed by two decimal digits, i (i = 1 or 2) and j (j = 0,..,7). Digit i specifies the line 1 or line 2 of the LCD display on which message will appear. Digit j specifies which of the stored annunciator message will be printed. Port **PM** selects i = 1, port **PN** selects i = 1.

5.3 Annunciator Raw Commands

The annunciator is controlled by sending it commands which consist of a 3-character ASCII string in the format “@ij”, where

- @ First character, command prefix, always “@”.
- i Second character, the LCD line number, “1” or “2”.
- j Third character, the message number to display, “0” to “7”, or the special hexadecimal number “f” to erase the line.

For example, to display message number 6 on line 1 of the LCD display, you would send the command string “@16” to the annunciator over the RS-485 serial link. To erase line 1 message, send command string “@1f” and whatever message is on the LCD line 1 is deleted.

Command message “@20” displays stored annunciator message number 0 on line 2 of the display. A subsequent command message “@2f” would then erase this message.

6 Observations

The LCD annunciator is a versatile peripheral that can be assembled by anyone with average construction skills in about two evenings work. About half the build time is attributed to the case preparation and in arranging all the component modules to fit efficiently inside the case without causing intereferences or difficult access to connections such as the USB socket.

Plan your component layout carefully, work slowly, and double check the wiring on the main board. The photographs of the prototype LCD annunciator shows one possible layout for mounting the project in a standard type of plastic utility case.

The annunciator as presented in this application note is controlled by a uTile PLC running on either the Mega2560 or Nano controller board platforms. This means it is easy to control the annunciator using simple commands within the uTile control framework, as shown in the user file program examples above.

The design of the annunciator is for a completely independent stand-alone device. The annunciator is controlled by simple 3-character ASCII control strings transmitted to the device via an RS-485 serial link. This means you can control the annunciator by sending it the desired 3-character control string. The device sending the control string can be any type of controller, even a simple terminal emulator is able to send these strings. See 5.3 Annunciator Raw Commands above for command string details.

You can use the annunciator in your own custom controller project. Just arrange for your controller to send the appropriate control strings at the required time to the annunciator and you are off to the races.

If you plan to use the LCD annunciator close to the master controller, say within a common panel, you can dispense with the RS-485 serial interface and send commands to the annunciator via the Nano board's USB port directly.