Multi-Purpose Car Scrolling Display
Part 3: User Software Instructions and Features Descriptions

BY: Mauro Grassi.

Introduction

Assuming you have built and tested the PC boards and installed the driver and software as explained in last month's article, you will now need to know how to use the host program for calibration, to change preferences and to set up the features of the display. All this is done using a PC (a laptop will be preferable if you've already installed it in a car), a USB cable and the provided command line program for Windows: **cardisplay.exe** (downloadable from the SILICON CHIP website at www.siliconchip.com.au.)

There are many user-changeable settings, but the software is organised into multiple-choice menus which should make it clear to use. In most cases, you will only need to change a setting if the default value is not suitable for your application, in other words, for customisation. And we have made the Car Scrolling Display highly customizable meaning it should be very versatile, it is truly a multipurpose display.

Aside from the main features described in the December 2008 issue of SILICON CHIP, there are features not previously mentioned, like battery protection (if using a car's battery as the power source), dependent variables and limit condition messages.

The latter are messages that, in the two scrolling display modes, will alert you when a variable is too low or too high (for example, a message will scroll saying "Battery Too Low!" when the battery level is below the set minimum). As explained in the December 2008 issue of SILICON CHIP, in static mode, flashing and reversed modes indicate the limit conditions.

Dependent variables have all the features of a normal variable measuring a sensor, except they take their reading from another variable rather than a sensor! This is quite a powerful spreadsheet-like feature. It allows you to form arbitrary polynomial functions of other readings. With dependent variables, for example, you can display temperature in both degrees Celcius and Fahrenheit. More generally, you can display the same reading in different units, or show a true and correct speed reading, among other uses. You can be very creative with dependent variables…

When using the display in a car, with the car's battery supplying power, you can enable battery protection, which will switch off the display automatically when the battery level drops below a set minimum. The Car Scrolling Display will then go into standby mode with minimal battery current drain (in the order of 20mA at worst, with typical current draw around 10mA). This feature requires a slightly different power supply connection as explained below in the section "Connecting Power". This mode is also recommended if you want to use running totals that persist over many on/off cycles.

All this, as well as all the standard features, is explained in this month's article. The calibration instructions and typical set up examples are left to next month's final article.

Connecting The Sensors

As mentioned in previous articles, the Car Scrolling Display can measure up to six different sensors in your car.

Four of these can be voltage/resistance sensors (CON3), which vary their output voltage or resistance to indicate the measured quantity. For example, a thermistor measures temperature by varying its resistance with temperature; alternatively, an oxygen sensor varies its output voltage around the stoichiometric point.

The other two inputs are for frequency/duty cycle sensors in your car (CON2). These sensors output a digital signal, indicating the varying quantity by either the frequency of the signal or its duty cycle.

Recall that the positive duty cycle of such a digital signal is the fraction of the time that the signal is at a high level, while the negative duty cycle is the fraction of the time that the signal is at a low level.

For example, fuel injector firing can be measured by the duty cycle of the signal going to the fuel injectors (and it can be either positive or negative duty cycle that indicates the proportion of the time that the fuel injectors are firing).

On the other hand, the tachometer signal in most cars is a signal whose frequency is proportional to the engine's RPM (revolutions per minute). Remember that there are two built-in readings, the battery voltage and the ambient light, so these six inputs are in addition to the built in readings.

So once you've decided which sensors to monitor, you connect them to either CON2 or CON3, as follows.

Connecting a Voltage/Resistance Sensor

For a Voltage/Resistance sensor, connect the output of the sensor to one of the four input channels on CON3 (referred to as AN0-AN3).

For a sensor with a varying voltage, you should also connect its ground reference to pin 6 of CON3. For a sensor with a varying resistance, however, you should connect its other end to pin 1 of CON3 (the +5V rail). Thermistors are examples of the latter type, while a narrowband oxygen sensor is an example of the former.

Note that you can always convert a resistance-based sensor to output a voltage by adding an external resistor and forming a voltage divider. However, the Car Scrolling Display can measure resistance directly without the need for an external resistor.

Connecting a Frequency/Duty Cycle sensor

For a Frequency/Duty Cycle sensor, connect the output of the sensor to one of the two input channels on CON2 (referred to as FQ0-FQ1). You should also connect the ground reference to pin 1 of CON2. If you want to measure fuel injector duty, for example, you would connect the fuel injector signal to one of the input pins of CON2. Refer to Table.1. for the pinout mapping physical channels to terminal block pins.

Physical Channel	Physical Input Pin	Measurement Type
0 (AN0)	Pin 5 of CON3	Voltage/Resistance
1 (AN1)	Pin 4 of CON3	Voltage/Resistance
2 (AN2)	Pin 3 of CON3	Voltage/Resistance
3 (AN3)	Pin 2 of CON3	Voltage/Resistance
4 (FQ0)	Pin 2 of CON2	Frequency/Duty Cycle
5 (FQ1)	Pin 3 of CON2	Frequency/Duty Cycle

Table.1: shows the pin-outs of the six physical channels. You connect the output of your sensor to either CON2 or CON3 depending on whether it is a voltage/resistance sensor or a frequency/duty cycle sensor.

Connecting Relays and Buzzers

If you are planning on using at least one of the two digital output channels for switching a relay or buzzer on a limit condition (a reading that is too high or too low), you should connect the relay or buzzer to CON4. Both the relay and buzzer should be rated at the same voltage as the power supply you are using. If you are using a 12V power supply you should use a 12V relay and 12V buzzer, while if you are using a 9V supply, you should use a 9V relay and 9V buzzer. Note that if you are powering the Car Scrolling Display solely from USB power (+5V) then neither a relay nor a buzzer will work on the output channels, because they take their positive supply from pin 4 of CON1 (the power supply input pin). Using USB power only does allow you to do everything else though, including data logging and powering the display.

There are two relay/buzzer outputs, referred to here as Output 0 and Output 1. Output 0 corresponds to pins 1 (-) and 2 (+) of CON4 while Output 1 corresponds to pins 3 (+) and 4 (-) of CON4. For relays, which are not polarised, the coil is connected across pins 1 and 2 or across pins 3 and 4. You should connect buzzers with the correct polarity, however.

Connecting the Power Supply

There are two options when connecting power. One way is to connect the ignition-switched +12V supply from your car’s battery to pin 4 of CON1 while connecting the negative terminal of the battery to either pin 2 or 3 of CON1 (it is immaterial which). You then connect the positive battery terminal to pin 1 of CON1 (for the battery level measurements). With this configuration, the “off” state current drain (the current drain when the Car Scrolling Display has shut down) is very small, it is essentially determined by the 66k# loading on the battery sense input (in parallel with the input impedance of the ADC input, which is very high). It works out to be roughly 200uA.

If you have the display connected this way in your car, and your car is unused for long periods (say a couple of weeks’) it prevents over-discharging your car’s battery.

Unfortunately, with this power supply connection, there are three features of the Car Scrolling Display that will not be available to you: one is battery protection, the second is automatic shutdown and wakeup, and the last is persistent running totals (see explanations below for an explanation of these).

For cars that are driven frequently, say at least once a week, and to ensure all features of the Car Scrolling Display will function correctly, you connect the power supply reversed.

The only penalty is that when the Car Scrolling Display shuts down (for example, when you switch off the ignition-key to your car), the display will still draw standby current, around 10mA. Note that most car batteries are rated for at least 20AH, so it would take weeks for this standby current to drain the car's battery (this could be worse if you have other devices installed in your car that also have standby power).

In this case, you connect the ignition-switched +12V supply from your car's battery to pin 1 of CON4 while connecting the positive battery terminal to pin 4 of CON4. You of course connect the negative supply the same way: the negative battery terminal still connects to either pin 2 or 3 of CON4 (it doesn't matter which).

In summary, for full features but higher standby current drain connect the power supply as just explained, otherwise reverse the positive connections to obtain minimal standby current drain while losing some features.

PC Host Program Overview

After having connected the sensors, relays/buzzers and power supply as explained above, the next thing to do is to connect the Car Scrolling Display to a computer running Windows, using a USB cable. You can do this at any time to change settings, calibrate the sensors (or when you add a new sensor) and to do real time data-logging.

Setting Up The Command Prompt Window

In last month's article, we guided you through the installation of the Windows Driver, and the PC host program. You may want to create a shortcut to go on your Windows Desktop to the command window. Suggested settings are shown in Fig.X. Change these by right clicking on the shortcut icon and going to "Properties".

The line width should be at least 128 characters, as the program can output some very long lines, which will otherwise wrap around and be more difficult to read.

When cardisplay.exe is run without command line options, it will display the Help menu with all the available command line options. The full list of command line options is shown in Table.2.

Command	Function	Example	Result
cardisplay or cardisplay –h	Show all available command line options (help).	cardisplay	The available command line options will be shown.

cardisplay –x:N or cardisplay –q:N	View real time information from variable number N, where N is in the range 0-5. The –x option is useful for	cardisplay –x:0	Shows information on variable 0 including all real time readings.
cardisplay –v (‘v’ is for variable)	Go To the variable setup menu.	cardisplay –v	You can change all variable settings, like minimum and maximum values, enable or disable averaging mode, change the name and unit strings for the variable, among others. The menu options are listed in Fig.X.
cardisplay –d (‘d’ is for display)	Go To the display menu.	cardisplay –d	You can change which readings are displayed and in what order, among other things. The menu options are listed in Fig.X.
cardisplay –i (‘i’ is for information)	Display Information about the Car Scrolling Display, including system settings.	cardisplay –i	Displays the current values of the settings, including system settings.
cardisplay –e (‘e’ is for extended)	Display extended information about the Car Scrolling Display.	cardisplay –e	Displays extended information including internal settings useful for troubleshooting, if needed.
cardisplay –c (‘c’ is for calibration)	Go To the calibration menu.	cardisplay –c	Follow the on screen prompts. The menus options are listed in Fig.X.
cardisplay –o (‘o’ is for outputs)	Go To the output configuration menu.	cardisplay –o	Allows you to configure the two digital outputs for connecting relays or buzzers. The menu options are listed in Fig.X.
cardisplay –u:N	Start real-time data logging up to N samples or until interrupted by the user.	cardisplay –u:100	Capture 100 samples in real time at the set logging frequency and dump the data to a .csv

			file.
Cardisplay –l:N:<filename>	Start real-time data logging up to N samples and store as file <filename> in .csv format	Cardisplay “–l:2000:my Data Log”	Capture 2000 samples in real time at the set logging frequency and dump the data to a file.
cardisplay -s (‘s’ is for system) or cardisplay -s:<N> (change extended system settings)	Go To the system settings menu.	cardisplay –s	Change any system settings. The menu options are listed in Fig.X.
Cardisplay –w:<filename>	Write all settings to a file for later recall.	cardisplay “-w:mySettings”	Saves all settings to the file mySettings.
Cardisplay –r:<filename>	Read all settings from a previously created file.	cardisplay “-r:mySettings”	Restores all settings from the file mySettings.
cardisplay –r	Reset the Car Scrolling Display and restore all default values.	cardisplay –r	Resets the car scrolling display and restores all default values.

Table.2: showing the command line options for the PC program “cardisplay.exe” running on your Windows OS. You can use a laptop to connect to the Car Scrolling Display if it is already installed in your car. You will need a USB cable and the PC host program will allow you to log your car’s signals in real time.

The main ones to keep in mind are (these have their own submenus):

(1) “cardisplay –d” (‘d’ is for display) to enter the Display Settings Menu, allowing you to change the brightness of the display, the scrolling speed, which readings are displayed and in what order;
(2) “cardisplay –v” (‘v’ is for variable) to enter the Variable Settings Menu, allowing you to change all variable-specific settings;
(3) “cardisplay –o” (‘o’ is for outputs) to enter the Output Settings Menu, allowing you to change the settings for the two digital outputs;
(4) “cardisplay –c” (‘c’ is for calibration) calibrate for a new sensor or to improve the accuracy by fine tuning the ADC supply reference.

Setting Name and Unit Strings

Each of the four values of each of the six variables has an associated name and unit string. The default name strings are of the form VarD(N) where D is the variable number (0-5) and N is the value number (0-3). For example, the default name of reading 2 of variable 3 is Var3(2). The default unit strings are similarly of the form UnitD(N).

You should change the default name and unit strings for your variable to something more meaningful. For example, if you are measuring engine RPM, you could choose the name “Engine Revs” and the unit “RPM”. To do this, run the command: **cardisplay -v** and choose the “Change the Name and Unit of the Variable”. You will be able to enter arbitrary strings. Once you've done that, you should choose the "Exit and Save" option and the strings will be saved to non-volatile memory.

Remember that you will need to add the reading to the display queue as explained in the section titled "Changing which Readings are Displayed and the Order" above, for it to be visible.

Setting Minimum and Maximum Values

Once you've done that, you can also set the minimum and maximum values for the reading, by running the command: **cardisplay -v** and choosing the "Change the Minimum and Maximum values & Hysteresis" option. Visible cues depend on the minimum and maximum values you enter. Recall from the first part of the article, that in static display mode, the readings will flash when below the set minimum and will flash reversed when above the maximum. In the other two scrolling display modes, there will be periodic warning messages like shown in Fig.X. instead.

If you are also planning to use one of the two digital outputs to monitor a reading and have a relay switch on or off or buzzer sound on the extremes, these are also the values that determine the switching on or off of the relay or buzzer.

For example, for a 12V car battery, a typical minimum value would be 11V and a typical maximum value would be 13.8V. For cabin temperature you may choose 16 °C for the minimum and 32 °C for the maximum, etc.

Configuring an Output Channel for Monitoring Limit Conditions

If you would like to monitor a reading and have a relay switch on or off on a limit condition, or have a buzzer beep once for a maximum condition and twice for a minimum condition, you should configure one of the two digital outputs.

Setting Up A Relay Output

Setting up a relay to switch on or off on a limit condition is easy. Run the command: cardisplay -o to enter the output configuration menu as shown in Fig.X. Note that the current settings of the two digital outputs will also be shown.

Choose the "Set Up a Relay" option. Then choose the digital output connected to the relay you wish to control. Then choose one of the four operating modes:

(a) Relay Switches on on a maximum condition
(b) Relay Switches off on a mimimum condition
(c) Relay Switches on on a minimum condition
(d) Relay Switches off on a minimum condition

Then choose the variable to monitor and the relay output will be set up. A complete procedure is shown in the screen grab of Fig.X.

Note that each variable has its own hysteresis value (as a percentage) which is settable by running the command: **cardisplay -v** and choosing the **"Change the Minimum**

and Maximum values & Hysteresis" option**.** A typical value will be 10%. This affects how a relay connected to one of the two digital outputs is switched on or off.

Suppose for example, that the minimum and maximum values for the battery voltage are set at 11V and 13.8V respectively, with the hysteresis at 5%.

If the relay is set to switch on on a maximum condition, then the relay will switch on whenever the battery voltage is above 13.8V and switch off whenever the battery voltage is below 95% of the maximum value, that is, below 13.1V.

Similarly, if the relay is set to switch off on a minimum condition, then it will be off whenever the battery voltage drops below 11V and will switch on again when the battery voltage is above 105% of the minimum value, that is, above 11.5V.

Setting Up A Buzzer Output

Again, run the command: **cardisplay -o**. Choose the "Set Up a Buzzer" option. Then choose one of the two digital outputs and the variable the buzzer applies to. That's all there is to it. Remember that the buzzer will emit two short beeps when the reading is below the minimum and one long beep when the reading is above the maximum. Recall also that beeping is only enabled in the **Static Display** mode.

Changing System Settings

There are a few system settings which can be set by running the command: **cardisplay -s**. You will be given the current value of each system settings and prompted to enter a new value. In all cases, pressing ENTER retains the current value. A typical screen grab is shown in Fig.X. Note that some are Enable/Disable settings and you simply type 'e' or 'd' to enable or disable respectively.

The system settings are:

(a) **Minor delay**: this is a number in 16ms units that sets the delay for flashing and reverse flashing in **Static Display** Mode. The smaller the number the shorter the time that the readings will be off (if flashing to indicate minimum condition) or reversed (if flashing in reverse to indicate a maximum condition).

(b) **Major delay**: this is a number in 16ms units that sets the time that the actual reading is displayed in Static Display Mode when a limit condition occurs. The smaller the number the shorter the time that the reading will be seen.

(c) **Readings Rounding Off Mode**: you can enable or disable rounding off of readings. If enabled rounding will be applied to all readings depending on how many digits after the decimal point are set for that variable. For example, if a variable is set to show 2 digits after the decimal point and the reading is 123.5565 with rounding enabled the display will show 123.56 rather than 123.55 with rounding disabled.

(d) **Zero Accumulators at Power On**: you can enable or disable setting all accumulators to 0.0 when the display is powered on. Disabling is useful if you want persistent running totals, ie, totals that retain their value when you switch off the ignition. To use persistent running totals you need a different power supply connection, as explained in the section titled “Connecting the Power Supply” above.

(e) **Automatic Wake Up On Battery Sense**: enable this to have the display turn on automatically when power is detected at the battery sense input. Note that this feature requires a different power supply connection as explained in the section titled “Connecting the Power Supply” above.

(f) **Readings Update Frequency (Logged):** you can set the readings update frequency for data logging from 0.2Hz to 8Hz. Note that the readings update frequency for normal display mode is always half of the logging frequency.

Changing Variable Settings

To set up a new variable, you should run the command: **cardisplay -v**. The variable setup menu will appear as shown in Fig.X.

You should run through the four options, in order:

(a) Change the Variable Type & Acquisition Parameters: here you can change the type of the variable:

- **a. Voltage/Resistance** variable**:** to monitor a sensor that relies on a varying voltage or resistance;
- **b. Frequency/Duty Cycle** variable: to monitor a sensor that outputs a digital signal with varying frequency or duty cycle;
- **c. Dependent** variable: to monitor a reading from another variable;
- **d. Silent** variable: this is a variable that has no readings attached to it. It is used to simply display text strings (up to four strings can be displayed);

This menu also allows you to change the physical channel the variable monitors (for a voltage/resistance variable there are 4 channels while for a frequency/duty cycle variable there are 2 channels). You can choose to enable or disable averaging mode. In averaging mode, the readings are averaged thus increasing noise immunity. You can choose to enable or disable averaging on the delta/integral values of the variable, choose whether the delta/integral values apply to the first or second value of the variable and choose which of the four values of the variable the accumulator will add up.

(b) **Change the Name and Unit of the Variable:** as previously explained, you can change the name and unit strings for each of the four values of the variable.

(c) Change the Minimum and Maximum Values & Hysteresis: as previously explained, you can change the minimum and maximum values for each of the four values of the variable, as well as the hysteresis.

(d) Change the Variable Display Parameters: here you can change the number of digits after the decimal point to display.

Display Settings Menu

Once you’ve set up the variable settings, you can choose which readings to display. The display settings menu allows you to choose which readings to display, in what order, to change the minimum and maximum display brightness as well as the scrolling speed.

Ambient Light Level Response

Note that the ambient light level is used to automatically dim the display when the light level drops. This can be overridden by setting the minimum and maximum brightness levels. The higher the number the brighter is the display. If you set the minimum to be equal to the maximum then the brightness will be fixed.

If for example, you set the minimum to 100 and the maximum to 250, the brightness will vary between these extremes depending on the ambient light level. It is usually desirable to have a dimmer display when the ambient level drops. It will also result in lower current drain.

The scrolling speed (applicable to the two scrolling display modes) can be varied. The lower the scrolling delay (measured in milliseconds), the faster the scrolling will be.

To set the minimum and maximum brightness as well as the scrolling speed, run the command: **cardisplay -d** and choose the “Change Display Settings” option.

Changing which Readings are Displayed and the Order

The display menu also allows you to select up to 10 readings to display and in what order they will appear. Note that the first reading cannot be hidden, so there will always be at least one reading displayed. The accumulator of each variable can also be displayed. It continuously sums any one of the four values of the variable.

When the accumulator is displayed in any of the two scrolling modes, the name string will be that of the value which it is summing. There will then be a ‘(+)’ after the name to indicate that the accumulator rather than the value is being displayed. This is shown in Fig.X.

To change which readings are displayed and in what order, or to hide or add a reading, run the command: **cardisplay -d** and choose the appropriate option as shown in the screen grab in Fig.X.

For example, choose “Change Order of Display Object” and select an object from the list of currently displayed readings. Then use the arrow keys to move the reading up or down in order.

Now that you’ve chosen which readings to display, you can proceed to calibrate the variable for a new sensor.

Battery Protection, Automatic Wake up on Battery Sense

With the correct power supply connection, you can enable automatic wake up on battery sense and battery protection. Recall that the Car Scrolling Display will go into standby mode (drawing very small currents) when it detects that the battery has dropped below the minimum (it will however, not go into standby mode if USB power is available).

The delay from detecting the low battery condition and the Car Scrolling Display shutting down with the message as shown in Fig.X can be set by running the command: **cardisplay -s**. The setting to change is the "Battery Protection Delay". This delay is useful, for example, if you are running a cooling pump from a relay on the digital outputs and you want it to operate for a while after the ignition key has been switched off.

When you enable automatic wake up on battery sense, the Car Scrolling Display will turn on when it detects that the battery level is above the minimum. You can enable or disable this by running the command: **cardisplay -s**. The setting to change is "Automatic Wake Up on Battery Sense".

Note that when you see the message in Fig.X. and the Car Scrolling Display goes into standby mode, you can manually force the display to wake up by pressing the pushbutton on the front panel. Keep it pressed for 10 seconds or so and the display should turn on.

Real Time Data Logging

For data logging, run the command: **cardisplay -u:<N>** or **cardisplay -l:<N>:<filename>** , where <N> is the number of samples and <filename> is the name of the file to save the data to.

Note that when specifying a command line option with spaces, you should use double quotes to enclose the argument. For example, to collect 2000 samples and store the result to a file called 'my Output Log' you would type at the command prompt: **cardisplay "-l:2000:my Output Log"** enclosing the argument in double quotes.

The produced file will be a .csv (comma separated values) file which can be opened with any spreadsheet. Only values which have been made loggable will be logged at the set logging frequency. To change the loggable values of a variable, run the command: **cardisplay -v** and choose the "Set the Logging Parameters For the Variable". You can choose to toggle whether each of the four values of the variable will be logged. If at least one value is made loggable, you can further choose to log the accumulator for the variable and to log the limit conditions.

A screen shot showing a log file opened in Microsoft Excel is shown in Fig.X. The battery level has been logged and a graph has been created.

Saving And Restoring Settings to File

To write settings to a file for later restoring, run the command: **cardisplay -w:<filename>**. You can restore settings by running the command: **cardisplay -r:<filename>**.

Restoring Default Values

To restore all default values, as well as clearing all strings, you run the command: **cardisplay -r**.

System Messages

From time to time, you will should see a number of messages scroll by on the display which are generated by the firmware in response to a number of events. These are shown in Fig.X.

Out of Range Values

In **Static Display** Mode, out of range values will be shown as four + signs. When a displayed value comes from a variable that has not been updated since being last displayed, the value will appear as 'N/A' in the two scrolling modes and as four – signs in the **Static Display** Mode (see Fig.X).

Changing the Welcome Message

When the Car Scrolling Display first powers on, a welcome message is shown. The default welcome message includes the firmware version. You can change the welcome message by running the command: **cardisplay -v** and choosing the "Change Welcome Message" option.

Multi-Purpose Car Scrolling Display
Part 4: Calibration and Recommended Setups.

BY: Mauro Grassi.

Introduction

As we had no space to talk about the calibration instructions last month, we give the details in this article.

Automatic or Manual ADC system Calibration

The Car Scrolling Display has a single +5V supply rail for all logic, including the microcontroller.

The accuracy of the six readings that depend on the ADC (Analog to Digital Converter) in IC1 (these being the four Voltage/Resistance inputs in CON3, the battery sense input at pin 1 of CON1 and the LDR sense input coming from the display PC board) depends directly on the accuracy of the +5V rail.

This is because the +5V rail is the positive supply reference for the ADC system.

Since the +5V rail is derived from REG1 and since small manufacturing variations can result in small deviations from 5V it is prudent to calibrate the ADC system once before performing any further sensor calibrations.

This allows you to set the actual voltage of the '+5V' rail (referred to as the "Supply Reference Voltage") and this will be used to correct all subsequent ADC conversions. Because the setting is stored in non-volatile memory, you need only do this once.

If you are powering the Car Scrolling Display primarily from USB power only, remember that the supply rail can then be quite much lower than +5V, in fact, because of the schottky diode drop and the tolerance in the USB power specifications, it can be as low as 4.5V. Be aware that if you calibrate the supply reference for a 12V car battery and then use USB power without the +12V input, the readings will be off, because the +5V rail will be lower than the set "Supply Reference".

The default value for the Supply Reference is of course 5.0V, so you need only perform this step if you are particularly fuzzy about the accuracy of the battery and ambient level readings. In some way, the calibration procedure for variables can correct discrepancies in both the +5V rail as well as tolerance errors in the values of the resistors comprising the voltage dividers on the analog inputs from CON3.

To do this run the cardisplay -c command and choose either the "Automatic Calibration of the ADC System" or "Manual Calibration of the ADC system".

The manual calibration option allows you to enter the actual voltage of the +5V rail. You should measure this with a multimeter and enter it when prompted. Easy.

The automatic calibration option can be used if you don't have a multimeter handy. You will need to connect pin 1 of CON3 (the +5V rail) to the battery sense input at pin 4 of CON1. This is shown in the screen grab in Fig.X. After sampling the voltage for 10 seconds, the supply reference correction will be automatically set by the firmware.

Once you've done that, you are ready to calibrate the variable for your sensor.

How to Calibrate a Variable

Once you've finished calibrating the ADC system, you can follow this section every time you need to calibrate a newly connected sensor. This needs only be done once, as the settings are stored in non-volatile memory.

For calibration purposes, you will possibly need a DMM (digital multi meter). One that can measure frequency and duty cycle will be preferable, a computer with USB port and a USB cable (a laptop will be needed if you have already installed the display in your car).

Calibrating a Sensor Involves Following These steps:

(1) Determine if your sensor depends on either a varying voltage or resistance to represent the changing quantity OR whether it relies on a varying frequency or duty cycle to represent the changing quantity. For example, a temperature sensor will most likely offer a varying voltage or resistance, while if you want to measure fuel injector duty, the signal's duty cycle will vary in proportion to the time the injectors are firing. Alternatively, a speed or tachometer signal from the ECU will usually be a digital signal of varying frequency.

(2) Run the command: **cardisplay -c** to enter the calibration main menu (you should have connected the Car Scrolling Display to your PC using a USB cable and have previously installed the Windows driver, as explained in last month's article).

(3) Choose the "Perform Calibration on a Variable" option. A new menu will be shown listing the six variables, the options will look similar to those shown in Fig.X. Note that it is possible that they may not be exactly the same, depending on which physical channels the variables are set up to monitor. Remember that two variables can monitor the same physical channel! The default however, is for each variable to monitor a different physical channel and that's what's shown. You should now choose which variable to calibrate. If for example, you connected your sensor's output to pin 4 of CON3 you should select 'b', as this is the channel AN1 (as shown in Fig.X). If your physical channel is not currently monitored by any variable, you can change which physical channel a variable monitors by first running the command: **cardisplay -v** and then choosing the "Change the Variable Type & Acquisition Parameters" option.

(4) Once you've chosen the variable to calibrate, you will see the next menu as shown in Fig.X. Choose the "Calibrate Variable for a New Sensor" option. If you know the "transfer function" of your sensor (for example, if you happen to have the datasheet for the sensor), you can enter calibration points manually. In this case, choose the "Perform a Manual Calibration" option. You can also enter a polynomial directly, by choosing the "Enter a Polynomial Manually" option. You can increase the accuracy of the readings of a previously calibrated sensor by adding more calibration points too. In this case, choose the "Add Calibration Points to Existing Variable". You can also import calibration points from a file (see section below, titled "Importing Calibration Points").

(5) Having connected your sensor's output to the corresponding input on CON2 or CON3, you will need some way of measuring the value the sensor monitors. For example, if your sensor's output is a square wave of frequency proportional to the speed of the car, you will need some way of knowing the instantaneous speed of the car. Similarly, if your sensor measures temperature, you will need some way of measuring the actual temperature, for example, with a thermometer.

(6) You will then be asked which of the two values of the variable to calibrate. These measure different things depending on whether the variable is measuring a voltage/resistance sensor or a frequency/duty cycle sensor. A typical menu is shown in Fig.X. In this example, where we have chosen AN1, we would select 'b' if the sensor is resistance based, and 'a' if it is voltage based. Note that you can also calibrate the Delta and Integral values, by entering a single scaling factor (see section below, titled "Delta and Integral Values").

(7) You will next be asked what the degree of the polynomial related to the variable and reading should be. Use 0 when the sensor response is linear through the origin, use 1 when the sensor response is linear but not through the origin, and use higher numbers for quadratic, cubic or higher degree polynomials etc. Remember that you will need to provide *at least* N+1 calibration points where N is the chosen degree of the polynomial. Providing more points than this will simply over-determine the polynomial and may waste space in the polynomial memory (which holds 50 points in total, shared among the 12 polynomials). Whenever providing calibration points, a point with the same X coordinate (ie, raw value) as a previously entered calibration point will not be accepted. This is because each valid calibration point must be taken at a different X coordinate. The uniqueness of the point is checked by the software; you will be asked to enter another calibration point if the point you provided had an X-value equal to a previously entered point. There is one other constraint: you cannot enter a calibration point with a zero X-value for a degree 0 (linear function through the origin) calibration.

(8) The next step is to enter the N+1 calibration points. You will see a message like shown in Fig.X (which may be different depending on what name you've chosen for your variable). The figure shown assumes you

have named the variable “Fuel Tank” and the unit “Litres” (and the sensor is presumably a fuel tank gauge). The calibration is done as follows. The raw value reading will be shown on the Car Scrolling Display and on the window on your PC. Then press a key to capture the point and then enter the correct Y-value for that point. For example, if calibrating a thermistor, you would capture the point while looking at a thermometer and noting the current temperature. You would then enter the value noted.

(9) Repeat step (7) for as many calibration points that you need. If you’ve chosen a quadratic polynomial, you will need to enter three points, and so on.

(10) Once the points have been accepted, the software calculates the polynomial of degree N that passes through the N+1 calibration points you entered. The polynomial will be shown and you will be asked whether you wish to commit the polynomial to memory. Then choose the Exit and Save option and the polynomial will be committed to memory.

That’s all there is to it!

Implementing a “Speedo Corrector”

Note that the dashboard speed-readings of cars will in general be calibrated, from the factory, to read slightly higher than the true speed you are travelling. This is done deliberately to take into account speeding infringements.

In some cases, you may be interested in knowing the true speed of your car. With this Car Scrolling Display, you can implement a kind of “speedo corrector” using the general calibration procedure described above.

Because it requires you to enter the “correct” value, you can enter a value of 100km/h when the dashboard is indicating 110km/h, for example. In other words, the “correct” reading is in fact, the reading you wish the sensor reading to correspond to on the Car Scrolling Display.

Improving the Vertical Sensitivity of a Voltage/Resistance Input

You can replace the 22k# resistor on the input channel with a smaller resistor. The formula for selecting the correct value is given in Part 1 of the article published in the December 2008 issue of SILICON CHIP. You will then need to run the command: **cardisplay -c** and choose the “Perform Calibration on a Variable” option. Choose the variable corresponding to your input channel and then choose the “Adjust For a New Horizontal Scale (X-scale)” option. Then follow the on screen prompts. A new polynomial will be calculated to take into account the new horizontal scale. Once you save the new polynomial, the readings should have the same response as when you initially calibrated the variable (with the old resistor in place). Or you could perform a whole new calibration, by running cardisplay -c and choosing the “Perform Calibration on a Variable” option, choosing your variable and then choosing the “Calibrate Variable for a New Sensor” option.

Delta and Integral Values

Each of the six variables contains two special values, which measure “Delta” and “Integral”. The delta value measures the difference in the reading from one update to the next. In other words:

Delta_t = alpha*(Value_t – Previous_Value_t)

where Delta_t is the value of the delta reading at time t, Value_t is the value of the reading at time t and Previous_Value_t is the value of the previous reading at time t and alpha is a constant scaling factor. The value of alpha can be set in order to convert to a different unit, by running the command: **cardisplay -c** and choosing the “Adjust Scaling For Delta/Integral Unit Conversion” option. For example, if the variable measures speed in km/h, then an alpha value of 0.27 can be set to have the delta value be in m/s instead of in km/h (ie, 10km/h=2.7m/s).

For example if the speed reading changes from 20km/h to 25km/h in one update period, then the delta value will be 5 (with alpha=1.0).

The updates occur at the set update frequency, which is always half of the set data logging frequency. The update frequency ranges from 0.1Hz to 4Hz (the logging frequency can be set from 0.2Hz to 8Hz by running the command: **cardisplay -s**).

The Integral value, on the other hand, is equal to:

Integral_t = beta * Value_t * dt

where Integral_t is the integral value at time t, value_t is the value of the reading at time t, dt is the time between updates in seconds and beta is a scaling factor. Again, the value of beta can be set as with alpha above. For example, if the update frequency is 2Hz, then dt will be equal to 0.5.

Note that the integral value can be used to perform a discrete integration on a reading. If you remember your calculus, the integral is the area under a curve. For example, the integral of velocity is displacement. If for example, you have set up a variable to measure speed, you can set the integral to apply to the speed reading and then setup the accumulator of the variable to apply to the integral value. The result will be that the accumulator will measure “distance” travelled! Note, however, that the accuracy of the integral will not be very high, because of the relatively low maximum sampling frequency (4Hz). The accuracy of the integral will be better the higher the sampling frequency is.

You can also choose whether to enable or disable averaging on the delta and integral values (to reduce noise) by running the command: **cardisplay -v** and choosing the “Change the Variable Type and Acquisition Parameters” option.

Note that for both delta and integral values, the Value_t can be set to apply to either value 0 or value 1 of the variable. You can set which value (0 or 1- these measure different things- as explained in the section “Firmware Overview” below) the delta and integral values of a variable apply to, as well as which of the four readings of the variable the accumulator applies to, by running the command: **cardisplay -v** and choosing the “Change the Variable Type & Acquisition Parameters” option.

For example, the delta value can be used to measure “Delta Throttle Position” from a throttle position sensor, indicating when the accelerator pedal is too abruptly pushed.

Because the delta and integral values have their own settable minimum and maximum values, you can even make a relay switch on delta and integral limit conditions. You can also have a relay switch off when the temperature of the engine rises too abruptly, for example!

A Typical Example: Fully Setting Up and Calibrating a Variable

For example, suppose you want to display engine revolutions. Connect the tachometer sensor to one of the two frequency/duty cycle inputs on CON2 and run the command: **cardisplay -c**.

Choose the "Perform Calibration on a Variable" option. Choose the corresponding variable the sensor is connected to. It will be either channel FQ0 or FQ1. Now choose the "Calibrate Variable for New Sensor" option. Now choose the value that measures Frequency (rather than the value that measures duty cycle).

Now choose the "Linear Function Not Passing Through the Origin" option (you can save one point in the polynomial memory by choosing the "Linear Function Through the Origin" option if you know that your car's tachometer signal is at 0Hz to register 0 RPM. Now follow the on screen prompts to enter either 2 points or 1 point (2 points for a linear function not passing through the origin, 1 point otherwise). You do this by noting the RPM reading as you capture the point by pressing a key, and entering it when prompted. Note that you may need to rev the car (in neutral gear!) to get a different calibration X-value if you are entering two points!

Once the points have been entered, the computed polynomial will be displayed and you should choose to commit the polynomial to memory. Then choose the "Exit and Save" option. That's all there is to it.

You can then name the variable by using the command: **cardisplay -v**. Choose the "Change Name and Unit of the Variable" option. Then choose the variable (either FQ0 or FQ1) and enter the strings. In our case, we may choose "Engine Revolutions" for the name string and "RPM" for the unit.

Once that is done, you can further customise the settings by running the command: **cardisplay -v**. Now all that remains to be done is to add the reading to the display queue.

Run the command: **cardisplay -d** and choose the "Add Display Object" option. Choose the variable and the frequency measurement when prompted. Then choose "Exit and Save". That's all there is to it. You should now be able to see the Engine Revolutions displayed (you may have to scroll past some other readings by pressing the pushbutton first).

Visualising a Polynomial

You can create a .csv file of any of the twelve polynomials by running the command: **cardisplay -c**. Then choose the "Perform Calibration on a Variable" option. Choose the variable and then the "Dump Polynomial to Spreadsheet" option. You will be asked to enter a filename to store the polynomial data. You can then open this file using a spreadsheet program like Excel or Sun's Open Office and create a graph of the polynomial. This will give you a graphical depiction of the polynomial that is currently stored for that variable and value. An example screen grab is shown in Fig.X.

Enabling or Disabling Visible Cues (Advanced Feature)

You can disable limit condition visible cues by running the command: **cardisplay -s:3** (change extended system settings).

When prompted, enter a value of 0x00 for “Visible Cues Mask” to disable all visible cues. On the other hand, enter a value of 0xFF to enable all visible cues. You can enable/disable only minimum or only maximum visible cues too. To enable only minimum cues enter a value of 0xAA, to enable only maximum cues enter a value of 0x55 instead.

Dependent Variables

Dependent variables function in the same way as other variables, the only difference is that they take their reading from another variable rather than from a sensor. You can enter a polynomial function by running the command: cardisplay –c. Then choose the variable and the “Enter a Polynomial Manually” option.

Note that it is quite possible to create circular references and deadlocks using dependent variables. In this case, the value will not be updated.

Messages using Silent Variables

You can use silent variables to display a scrolling message. Recall that silent variables do not display their values, and so you can use the name string to display a message.

Break Out Panel: Firmware Overview

In this section, we give an overview of the internal operation of the firmware, something to keep in mind in order to better understand the settings and their effects. This is also useful in order to surmise exactly what is possible with this multi-purpose display.

The firmware implements six “variables”. Each of these can be set to monitor one of the six “physical” input channels, comprising the two Frequency/Duty Cycle inputs in CON2 and the four Voltage/Resistance inputs in CON3.

The default is for each of the variables to monitor different physical channels, but you may want to have two variables monitoring the same physical channel (ie, the same sensor).

Note that there also two other built-in readings, the battery voltage and the ambient light level (these can be displayed too).

Each Variable stores, among other things, the following floating-point numbers:

- Four Readings (here referred to as Value(0)-Value(3)), each with its own:
    - Minimum and Maximum values; as well as:
    - Name and Unit character strings;
- Two Polynomials applying to Value(0) and Value(1);
- Two Scaling Factors applying to Value(2) and Value(3);
- One Accumulator which continuously sums any of the four values;
- Its own settings, including:
    - Hysteresis as a percentage, used for the output channel switching;
    - Number of Digits to Display After The Decimal Point;
    - Acquisition Mode: AVGERING mode or DIRECT mode;
    - Logging Settings: allowing you to select which of the four values are logged (any number of these can be logged);
    - Mode and Physical Channel the variable monitors; Each variable can operate in one of four modes:
        - ANALOG/RESISTANCE mode (when measuring a Voltage/Resistance input from CON3); In this case there are 4 possible physical channels corresponding to the analog/resistance inputs in CON3;
        - FREQUENCY/DUTY CYCLE mode (when measuring a Frequency/Duty Cycle input from CON2); In this case there are 2 possible physical channels corresponding to the frequency/duty cycle inputs in CON2;
        - DEPENDENT mode; In this case there are 7 possible channels to choose from as the changing variable, comprising the five other variables and the two built in readings of battery voltage and ambient light level. In this mode, the readings depend on the readings of other variables!; and
        - SILENT mode (where the variable is not used to measure a reading but only to show a scrolling message on the display);

The four readings of the variable function differently depending on which of the four modes the variable is operating in, as follows:

In Analog/Resistance mode:

Value(0) is linearly proportional to the voltage at the input; and
Value(1) is linearly proportional to the reciprocal of the voltage at the input.

In this mode, Value(0) should be used to measure voltages, Value(1) is used to measure resistances directly. For example, Value(0) can be used to measure the voltage output of a narrowband oxygen sensor, while Value(1) can be used to measure the resistance of a thermistor to measure temperature (note that you can't connect both of these sensors at the same time. You would need to use another variable for the other sensor).

In Frequency/Duty Cycle mode:

Value(0) is linearly proportional to the frequency of the signal at the input; and
Value(1) is linearly proportional to the positive duty of the signal at the input

In this mode, Value(0) should be used to measure the frequency of the sensor output, while Value(1) should be used to measure the positive duty cycle of the signal. For example, Value(0) can be used to measure the speed of your car, while Value(1) can be used to measure Fuel Injector Duty (again, you can't connect both of these sensors at the same time, you would need to use another variable for the other sensor. Only if your sensor represents information with both the duty and the frequency can you then use the two Values simultaneously).

In Dependent mode:

Value(0) is a polynomial function of any of the four values of another variable; and
Value(1) is a polynomial function of any of the four values of another variable.

Note that it is quite possible, when using dependent variables, to create a circular reference. In this case, the value will not be updated as indicated in Fig?.

In Silent mode, neither of the four values are displayed.

For all four modes, the other two values (ie, Value(2) and Value(3)) have the functions:

Value(2) measures the "delta" (change from previous reading to current reading) of either Value(0) or Value(1); and
Value(3) measures the "discrete integral" (the trapezoidal rule is used) of either Value(0) or Value(1) of the variable;

(Note that Value(2) and Value(3) either both apply to Value(0) or both apply to Value(1), you cannot set one to apply to Value(0) and the other to apply to Value(1) at the same time).

This is shown in the graph in Fig.1.

Finally, each variable has an accumulator that simply sums one of the four values (you select which by going to the variables menu “cardisplay.exe –v” and select the “Change the Variable Acquisition Parameters” and follow the prompts).

Break Out Panel: Polynomial Interpolation in a Nutshell

What makes this Car Scrolling Display different to most other displays designed for cars (including many published previously in SILICON CHIP) is that it implements polynomial interpolation.

This makes it highly adaptable to many different sensors found in cars. Although most sensors exhibit a linear response, meaning that the output of the sensor grows as a linear function of the monitored quantity, there are some sensors like narrowband oxygen sensors that exhibit an S-curve response.

Many different sensor “transfer functions” can be approximated by polynomials.

This feature also allows you to compute the negative rather than positive duty cycle, as well allowing you to measure resistances directly (ie, without a voltage divider).

One more benefit is that you can “correct” readings, for example, you can display true speed (rather than using the speed indicated on the dashboard of your car, which usually shows a higher than true reading). We here explain all you need to know about polynomials for the purposes of this project.

Recall that a polynomial is a function of the form:

p(x)=a(n).x^n + a(n-1).x^(n-1) + … + a(0)

where a(0), a(1), … a(n) are real numbers (like 3.4 or –10.23), referred to as the coefficients of the polynomial. The number n is called the degree of the polynomial (provided that the number a(n) is not zero).

For example, a line of the form p(x)=ax + b is a polynomial of degree 1 (its gradient is the number a and the y-intercept is the number b). In this case a(1)=a and a(0)=b.

The Car Scrolling Display uses polynomial interpolation for each of the variables, although only two of the four values of the variable have polynomials associated to them. This means that there are in total 12 polynomials in polynomial memory (2 for each of the six variables).

The polynomials are computed by the software program running on your PC (cardisplay.exe). To determine a degree n polynomial you need at least n+1 calibration points.

For example, two points determine a line, three points determine a quadratic polynomial (degree 2 polynomial), etc.

You will need to know the degree of the polynomial to use for your sensor. Most car sensors will exhibit a linear response. If you are not sure, you should choose a high number, like 8. You will need to provide 9 calibration points, but if the sensor response can be approximated by a lower degree polynomial, the program will store only the lower degree polynomial in memory. In other words, you should overestimate as a general rule, if you are not sure of the sensor’s response.

Now a degree 0 polynomial is technically a constant function $p(x)=a(0)$. You only need one point to determine such a polynomial, the point will be of the form $(x, a(0))$ where x is any real number.

Since a degree 0 polynomial is quite useless for a variable that is supposedly changing, *the firmware interprets a degree 0 polynomial as a degree 1 polynomial with a y-intercept of 0.* In other words, a degree 0 polynomial is interpreted as a function $p(x)=a(0)x$ which is a line through the origin with gradient $a(0)$.

The polynomial computed is the polynomial that passes through the calibration points entered and this polynomial is then used as the function to compute the value of the variable from the raw sensor readings.

All polynomial memory is shared (the polynomial memory can store up to 50 floating point numbers, which is shared among the twelve polynomials).

Car Scrolling Display Errata

In the circuit diagram on pages 32-33 of the December 2008 issue, pin 2 of CON2 should be labelled FQ0 while pin 3 of CON2 should be labelled FQ1.

In Fig.10 of last month's article (page 69) the caption should read: "this is the dialog that appears the first time the Car Scrolling Display is connected to the PC".

The low pass filter will need to be modified on the two frequency inputs (FQ0 and FQ1) if you are measuring frequencies above 5kHz. Change to a 3.9k resistor as the pull up to replace the 10k resistors near Q18 & Q19 and adjacent to the 1k resistors.

This is because the variable will not update properly otherwise, because the slope of the 10nF capacitor charging curve during the high period of a signal above 5kHz is too gentle to trigger an edge detection at the CCP1 and CCP2 inputs of IC1.