Capacitance: 0.1pF to >1F ✓
Inductance: 10nH to >1H ✓
Easy to build/low cost ✓
Arduino based ✓
Ultra low drift ✓
Auto drift compensation ✓
Auto L or C identification ✓

WOW! That's what we'd call an LC METER you really should build!

By Tim Blythman

Not only is this new digital Inductance-Capacitance Meter easy to build (it's based on a custom Arduino shield and a standard 4-line alphanumeric LCD display), it features very low drift due to a constant self-calibration procedure. Best of all, it has an extended measurement range from less than 1pF to over 1F (yes, 1 FARAD!!) for capacitors and under 100nH to several henries for inductors. You simply must add this one to your test equipment arsenal!

A **wide range LC Meter** is a very worthwhile device to have on your workbench. But have you tried to buy a good one lately? (Hint – mortgage the kids first!).

Many DMMs have a capacitance meter built in but their range is usually (very!) limited. And most cannot measure inductance at all.

But now you can have an ultra-wide range LC meter which has all the "most wanted" features – high accuracy, very low drift . . . and best of all, it won't cost you a lot to build.

That's mainly because it is based on an Arduino processor. You can choose to build it in a case as a genuine piece of test gear . . . or assemble it with your Arduino Uno whenever you need to measure a capacitor or inductor.

This design is very accurate because it automatically compensates for its own thermal drift and it can measure very small and very large capacitance and inductance values.

It automatically senses the component type, so you can connect virtually any capacitor or inductor, big or small, to the device and it will quickly tell you its value on the LCD screen. You can even measure supercapacitors!

This is actually the third LC Meter that we have published in the last twelve months. This latest iteration is a big improvement over the last two, both in both performance and ease-of-use while being only slightly more complex.

Most importantly, it solves the drift problem that has plagued most DIY LC Meter designs and many commercial designs as well.

Why "measurement drift" is a problem

This type of LC Meter design has a long history. Our projects in May 2008, June 2017 and January 2018 were all based on an earlier design by Neil Heckt from around 1998, which used a PIC16C622 microcontroller.

These are all based on an oscillator which incorporates the unknown device (inductor or capacitor) to be measured.

The parameters of the device to be measured (eg, inductance or capacitance) affect the oscillator frequency and by measuring the change in frequency, we can estimate its inductance or capacitance.

This approach requires us to measure the initial oscillator frequency, then the frequency with the unknown device in circuit and then calculate the difference. A formula is then used to compute the inductance or capacitance.

The problem is that all those previous designs only measure the initial (default) oscillator frequency when the unit is first powered up. That's because the user has to manually disconnect any components from the test terminals. Unfortunately, as the unit warms up, the oscillator frequency shifts.

So unless you disconnect the device under test (DUT) and "reboot" the Meter every time you want to make a new measurement, it won't necessarily be accurate.

That inevitable drift in oscillator frequency means that regardless of how precise the initial calibration may be, each successive measurement is likely to be progressively less accurate.

Our solution is simple: get the microcontroller to frequently disconnect and re-connect the DUT from the circuit. So it can measure the oscillator frequency with and without the DUT (device under test) at very short time intervals and compute the difference on this basis.

So it's constantly compensating for any drift due to temperature, ageing or other factors.

The microcontroller uses reed relays to switch the DUT in and out of circuit. You don't have to do anything to select or control this process; the microcontroller does it automatically.

At the same time, the micro can decide to add some extra components which allow it to make measurements using a different method that's more suitable for higher values of capacitance and inductance.

This is how we've greatly extended its measurement range.

We have not made any changes to the oscillator circuit around IC1, compared to our last LC Meter (January 2018; **siliconchip.com.au/Article/10934**).

We spent some time looking for ways to improve this but there doesn't appear to be any easy way to improve it.

If you want more details on how the oscillator is used to measure the inductor or capacitor value, see Jim Rowe's detailed description in the June 2017 issue at: **siliconchip.com.au/Article/10676**

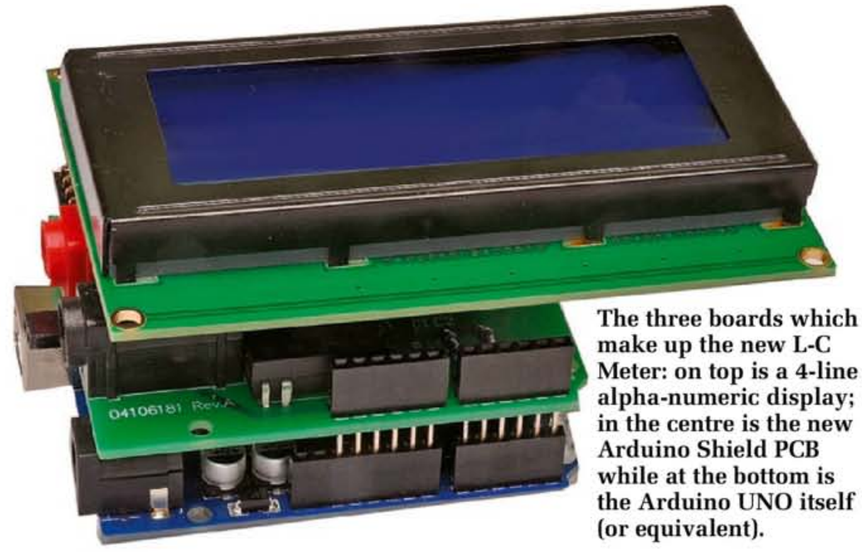
Extending the measurement range

Our previous designs and indeed any LC Meters based on this circuit configuration are limited to handling a maximum capacitance value of slightly more than 1µF and a maximum inductance of around 100mH. Values

Features & specifications

Advanced calibration
Continuous drift compensation
Long-term averaging
Inductance range: 10nH to 1H+
Capacitance range: 0.1pF to 1F+ (minimum rating 5V)
Measurement resolution: four significant figures
Component detection: automatic
Sampling rate: Once every two seconds (approx)
Accuracy (when calibrated): within ±1% of reading
Supply voltage: 5-12V DC @ <100mA
Easy-to-assemble, low cost Acrylic case available.

If you're a typical hobbyist – or even a repair centre – you've probably got a pile of capacitors in your junk box and don't know if they're good, bad or indifferent. And even your DMM can't tell you because it won't go high enough, especially for electros. Build this LC meter and you can check them all (even supercaps) – plus all those inductors with no markings!



The three boards which make up the new L-C Meter: on top is a 4-line alpha-numeric display; in the centre is the new Arduino Shield PCB while at the bottom is the Arduino UNO itself (or equivalent).

higher than this tend to prevent the oscillator from functioning.

Since there is no easy way to fix the oscillator to solve this range problem, we've used a different method to measure large component values, based on measuring the time constant of an RC or RL circuit.

A separate panel in this article explains how that method works.

To allow for this extra measurement mode, we needed a way to disconnect the DUT from the oscillator but luckily, we had already added that capability using reed relays for the drift cancellation feature.

We also needed a way to disconnect the time constant measurement circuitry from the DUT so that it doesn't affect the oscillator – but that turned out to be easy.

The pins on the micro which perform the new measurement function are simply set into a high-impedance state when not being used and they then have virtually no effect on the behaviour of the oscillator.

In the end, the only changes required to add this new measurement mode were two resistors and some extra software routines.

Aaaagh – I just built the earlier LC Meter!

Stay calm and don't panic! You haven't wasted your money . . . We thought about those readers who have a previous version of an LC Meter and made it easy (and cheap!) for you to upgrade.

Just transfer most of the existing components from your previous shield board to our new custom shield, install the four new reed relays and extra resistors, port in the new software and it's done.

Or you you can simply install fresh components on the new shield and go from there.

Circuit description

The full circuit diagram is shown in Fig.1. Everything except the Arduino and 20x4 alphanumeric LCD is mounted on the shield board for the Arduino.

Let's look first at how the oscillator-based measurements are made.

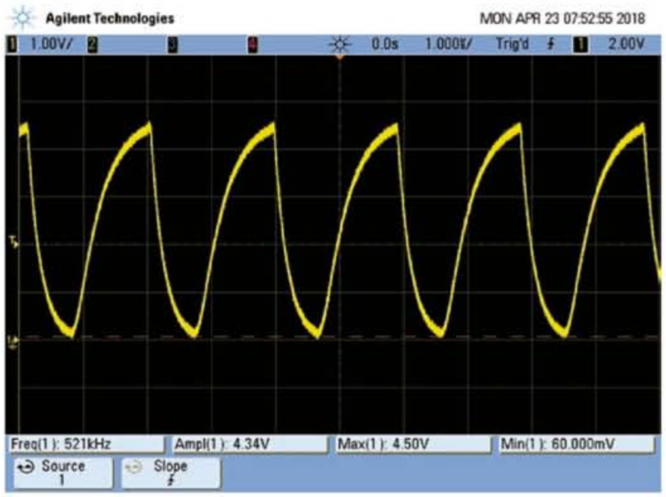
IC1 is a high-speed LM311 comparator, used to drive the resonant circuit into oscillation. The resonant circuit consists of inductor L1 (100µH) and capacitor C1 (1nF). The junction of these two components is coupled to the non-inverting input of IC1, pin 2, via a 10µF capacitor.

Positive feedback is provided around IC1 by a 100kΩ resistor from its pin 7 output to the pin 2 non-inverting input. Pin 2 is also connected to a divider across the 5V rail so that when pin 7 of IC1 goes high, the voltage at pin 2 will be pulled up to around 2/3 of the 5V supply, or 3.3V.

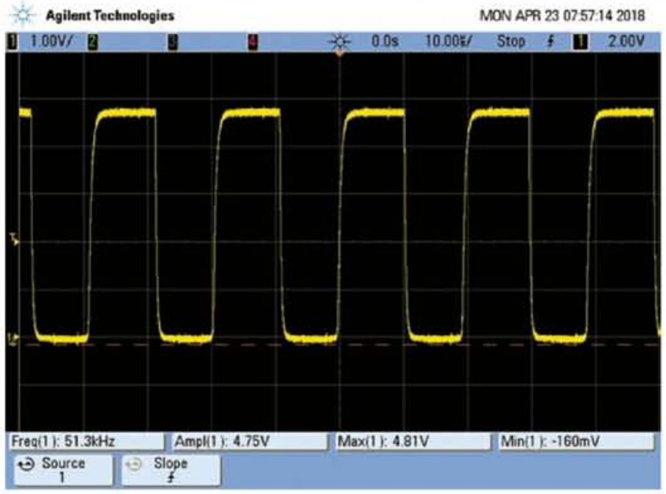
Since IC1's output is a transistor collector, a 4.7kΩ pull-up resistor is used to take it to 5V when that transistor switches off and it goes low, to 0V, when the transistor switches on.

The pin 3 inverting input of IC1 is connected to the output via an RC low-pass filter (47kΩ/10µF) and the capacitor charges up to the average output voltage so that the oscillator should stabilise with a reasonably symmetrical waveform.

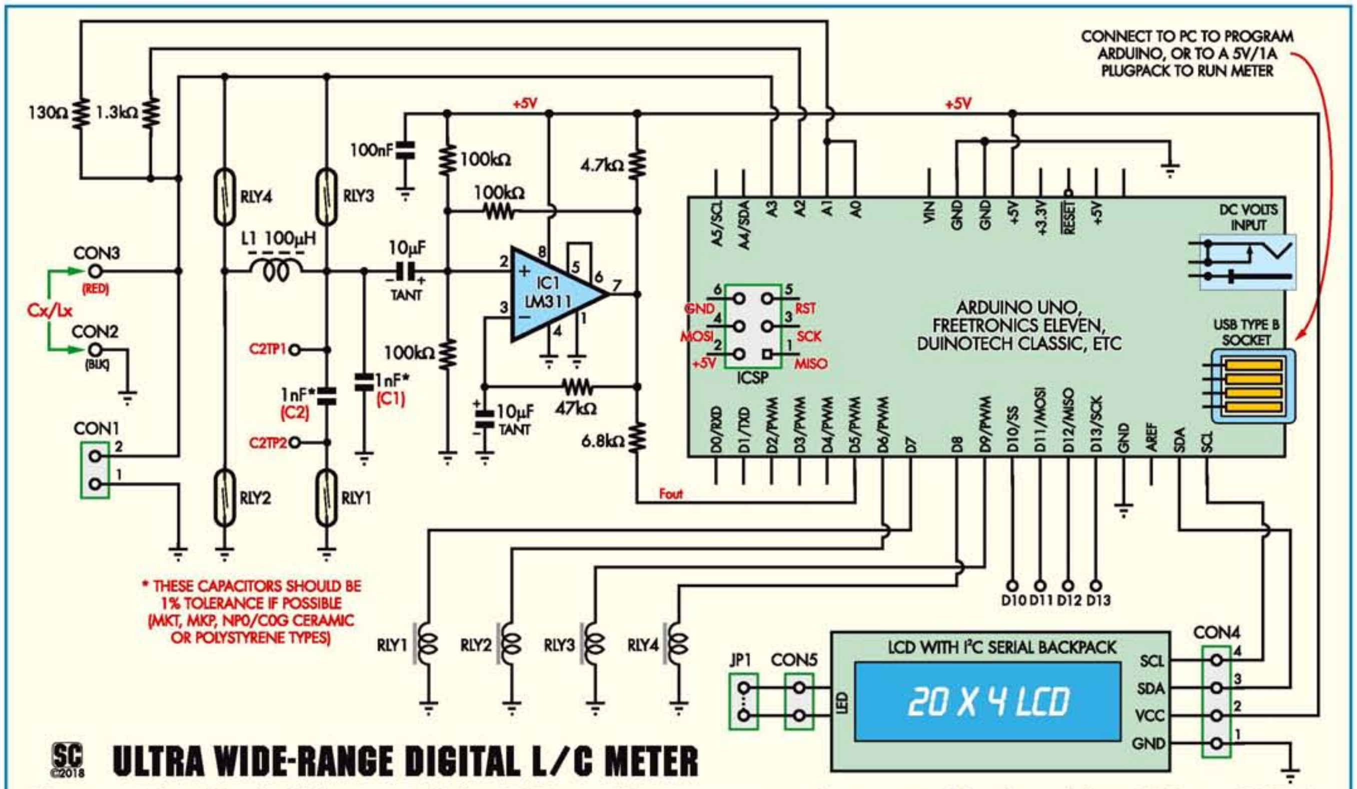
This waveform is fed to digital input pin D5 of the Arduino via a 6.8kΩ current-limiting resistor.



Scope1: measurement of the F1 frequency during the calibration phase. As detected by the scope, the frequency is 521kHz with an amplitude of 4.34V.



Scope2: measurement of a 100nF capacitor. The frequency has dropped to 51.5kHz. Note the amplitude is up to 4.75V, and the oscillator seems quite stable.



CONNECT TO PC TO PROGRAM ARDUINO, OR TO A 5V/1A PLUGPACK TO RUN METER
+5V
130Ω
1.3kΩ
100nF
100kΩ
4.7kΩ
RLY4
RLY3
L1 100µH
CON3
(RED)
Cx/Lx
CON2
(BLK)
10µF TANT
IC1 LM311
C2TP1
C2TP2
1nF* (C2)
1nF* (C1)
100kΩ
10µF TANT
47kΩ
6.8kΩ
CON1
RLY2
RLY1
Fout
ARDUINO UNO, FREETRONICS ELEVEN, DUINOTECH CLASSIC, ETC
ICSP
GND MOSI +5V RST SCK MISO
DC VOLTS INPUT
USB TYPE B SOCKET
D10 D11 D12 D13
* THESE CAPACITORS SHOULD BE 1% TOLERANCE IF POSSIBLE (MKT, MKP, NP0/C0G CERAMIC OR POLYSTYRENE TYPES)
JP1
CON5
LCD WITH I²C SERIAL BACKPACK
20 X 4 LCD
SCL SDA VCC GND
CON4
ULTRA WIDE-RANGE DIGITAL L/C METER

Fig.1: complete circuit of the new Arduino LC Meter. The component to be measured is plugged into CON1 or CON2/CON3 which is then connected to the test oscillator by RLY3 or RLY4, depending on its type. The Arduino can also perform time-constant based value measurements using its A0-A3 pins and the 130Ω and 1.3kΩ resistors.

Oscillator configurations

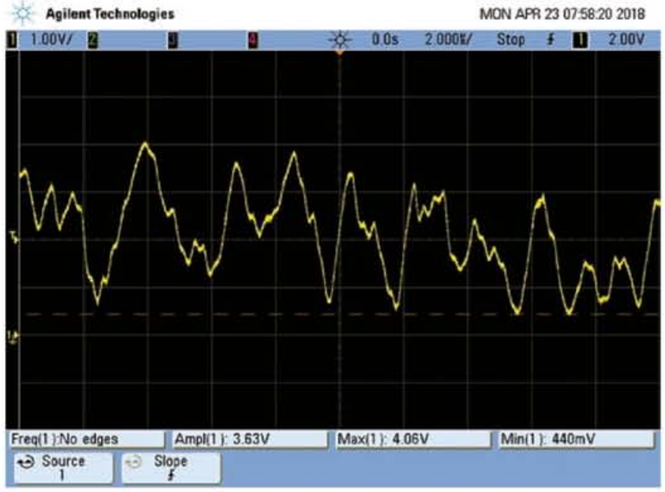
Four reed relays, RLY1-4, each connect either to one end of L1 or to the calibration capacitor C2 and are used by the micro to select one of five different modes.

The state of each relay in each mode is shown in Table.1. Each mode works as follows.

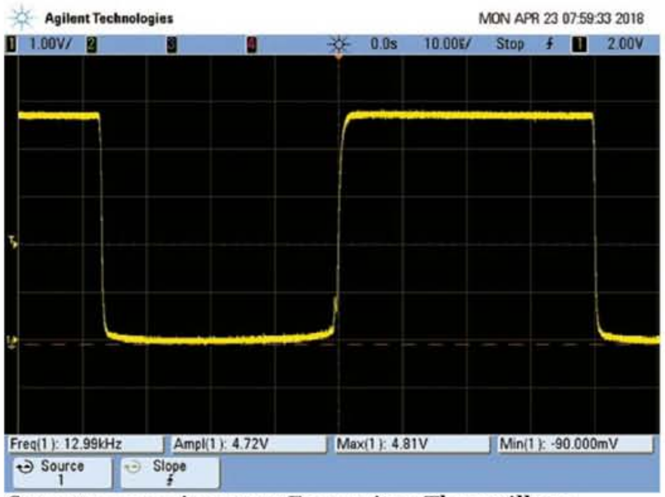
Mode 1 is for oscillator calibration. RLY2 is energised and its contacts are closed, effectively connecting L1 and C1 in parallel. The other relays are not energised. L1 and C1 resonate and cause IC1's output to produce a square wave of around 500kHz, which can be measured by the Arduino using its internal timer hardware.

Mode 2 is also for calibration. Both RLY1 and RLY2 are now energised but RLY3 and RLY4 are not. This is identical to the first mode except that now, C2 is connected in parallel to C1. The oscillator frequency drops to around 370kHz, due to the doubled capacitance. This allows the unit to measure the stray capacitance on the PCB and provide more accurate component value measurements.

Mode 3 is for measuring the value of a capacitor connected either between the pins of CON1 or between banana sockets CON2 and CON3 (these are effectively in parallel). In this case, RLY2 and RLY3 are energised and RLY1 and RLY4 are not. This is similar to the first



Scope3: in this case the oscillator is not stable when measuring a 407µF capacitor. To get an accurate measurement you need to use the RC method (see Scope6).



Scope4: measuring a 1.5µF capacitor. The oscillator appears stable, but a small glitch appears at the start of the leading edge.

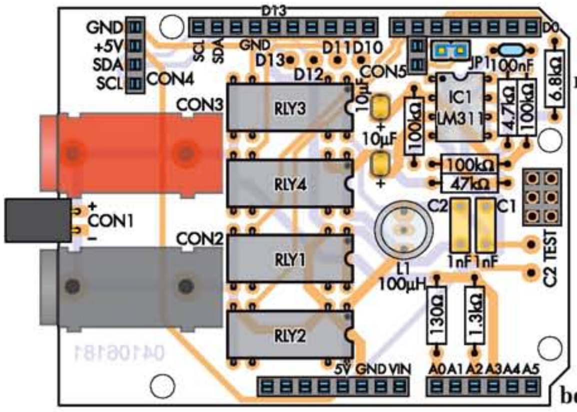
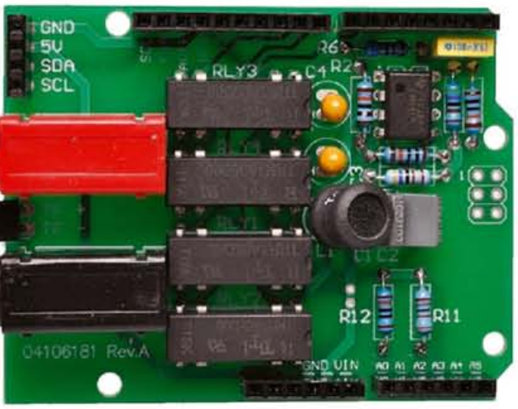


Fig.2: follow this overlay diagram – and the photo at right – to build the LC Meter shield PCB. Be careful with the placement of L1 as an extra pad is provided for larger inductors; also ensure that RLY1-RLY4, IC1 and the two tantalum capacitors are fitted with the orientation shown. The PCB at right is an early prototype; some minor adjustments have been made in the final version.



mode except that now the DUT is connected in parallel with C1.

Since the capacitance has increased, this should result in a lower oscillator frequency and by measuring the change, we can calculate the capacitor value, using the method explained previously.

Mode 4 is for measuring inductance. In this case, only RLY4 is energised. The DUT is connected in series with L1 and its opposite end is connected to ground via CON1 or CON2.

This means that the resonant circuit inductance has effectively increased (by the value of the DUT) and so once again, the oscillator frequency should drop and the difference can be used to calculate the inductance of the unknown component.

As explained earlier, in the above modes, the A0-A3 pins on the Arduino are kept in a high impedance state so they won't interfere with the oscillator. Their only influence is in their (small) pin capacitance and this is compensated for during calibration.

Mode 5 is used for measuring high values of inductance and capacitance, and now the A0-A3 pins become active.

In this mode, all four relays are off and the DUT is not connected to the oscillator. Instead, the 130Ω and 1.3kΩ resistors are used to drive the DUT and A0, A1 and A2 become outputs at different times.

A0 and A1 are connected together to measure the internal pin resistance, as this appears in series with the 130Ω resistor and can cause measurement errors.

The A3 pin is used as an analog input, to measure how the voltage across the DUT changes in response to the current from the other two pins.

The two different resistors value are used to provide two different ranges, to improve the unit's accuracy. In this mode, capacitance and inductance measurements are based on charge time measurement, as described below.

You don't have to select any of these modes or tell the micro that you are measuring capacitance or inductance. The micro does everything automatically.

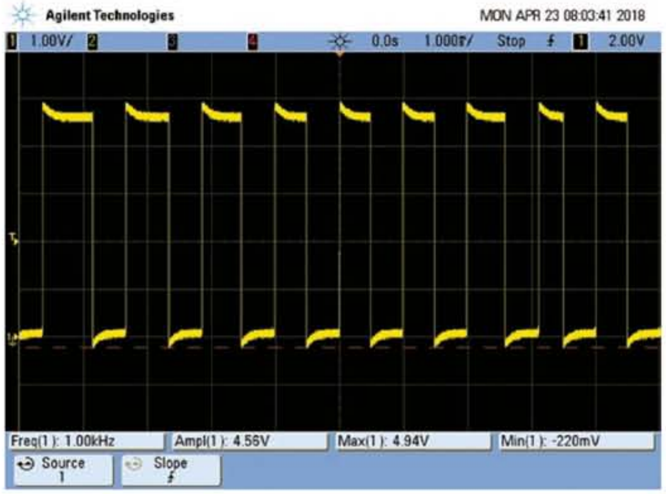
Displaying the results

Results are displayed on a 20x4 Alphanumeric LCD which is fitted with an I^{2}C adaptor so that only four connections to the Arduino are required: +5V power and ground, and the SDA and SCL pins for I^{2}C communications.

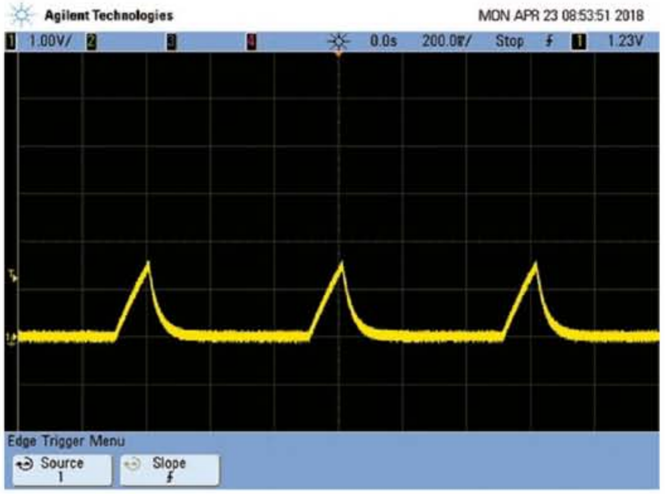
The four-line display constantly displays all the measurement data that you need to see and it even keeps a running average for super-accurate measurements.

It's powered from the Arduino's 5V rail, which can be derived from a USB charger, PC USB port or 9-12V DC plugpack via the on-board barrel connector.

The four reed relay coils are driven directly from the Arduino's digital output pins D6-D9. These outputs can provide more than enough current to latch a reed relay (up to 40mA) and the back-EMF at switch-off is sufficiently low that the ATmega328 IC's



Scope5: measuring a 0.5H (approximate) inductor. The frequency is down to 1kHz, but the oscillator is less stable – note that the pulse widths are varying significantly.



Scope6: a 220μF capacitor in RC mode. The measurement is repeated (about every second depending on the capacitor value) and the capacitor is discharged at the end of each cycle, to be ready for the next cycle.

internal clamp diodes are sufficient for absorbing it.

By the way, don't be tempted to substitute a different type of Arduino board. You need to use the Uno or equivalent.

This is because the Frequency Counter library that we use depends on pin D5 being fed into one of the hardware timers and this is only the case with the Uno.

Construction

The custom shield is built on a double-sided PCB measuring 68.5 x 53mm (ie, standard shield size) and coded 04106181.

The overlay diagram is shown in Fig.2 – use this and the matching photo as a guide during construction.

Start by fitting the resistors. While their values are printed on colour-coded bands, it's safer to simply measure the values with a DMM before soldering them in place. Next, mount the MKT/ceramic capacitors and inductor L1. None of these components are polarised.

What you wouldn't give to have an accurate L-C meter on hand right now!

While there are three holes for L1, only two are needed; the extra hole is to allow for variations in component size.

Ensure that one of the inductor leads goes through the hole closest to to the bottom edge of the PCB. The other lead can be soldered to either of the other pads.

Now fit the two tantalum capacitors. These are polarised; their positive leads will be identified with a "+"

Parts list – Ultra wide range, Arduino-based LC Meter

1 double-sided PCB, 68.5 x 53mm [SILICON CHIP code 04106181]
1 set of Arduino stackable headers (1x6 pin, 2x8 pin, 1x10 pin, Jaycar HM3208)
1 Arduino Uno or equivalent [Altronics Z6280, Jaycar XC4410]
1 20x4 alphanumeric LCD [Jaycar QP5522, SILICON CHIP SC4203]
1 I^{2}C Port Expander module [Jaycar XC3706]
1 100µH bobbin-style inductor [Altronics L6222, Jaycar LF1102]
4 5V coil DIL reed relays [Altronics S4100, Jaycar SY4030]
2 2-pin female header sockets (CON1,CON5)
1 black PCB-mounting right-angle banana socket (CON2) [Altronics P9201]
1 red PCB-mounting right-angle banana socket (CON3) [Altronics P9200]
1 4-pin female header socket (CON4)
1 2-pin header with shorting block (JP1)

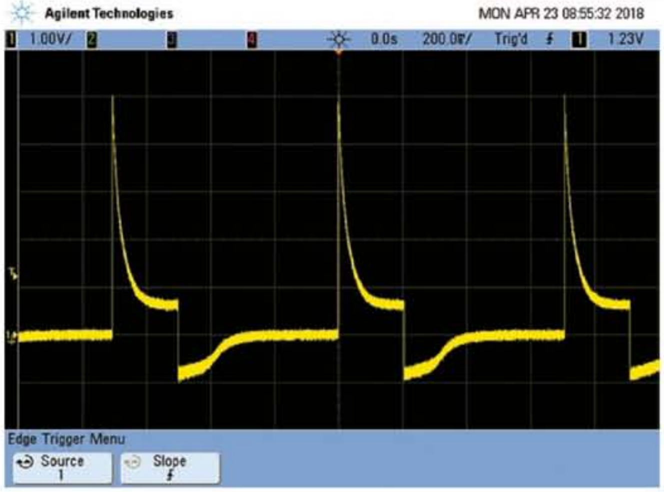
Semiconductors
1 LM311 high-speed comparator, DIP-8 [Altronics Z2516, Jaycar ZL3311]

Capacitors
2 10µF 6.3V tantalum
1 100nF MKT or ceramic *(code 0.1µf; 104 or 100n)*
2 1nF 1% NP0/C0G ceramic or polystyrene* *(code 0.001µf; 102 or 1n)*
[SILICON CHIP SC4273] **1% tolerance if possible*

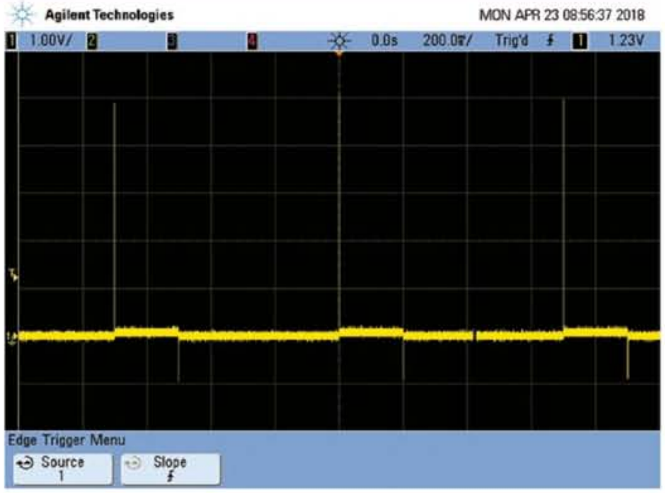
Resistors (all 0.25W, 1% metal film)

	4-band code (1%)	*5-band code (1%)*
3 100kΩ	*brown black yellow brown*	*brown black black orange brown*
1 47kΩ	*yellow violet orange brown*	*yellow violet black red brown*
1 6.8kΩ	*blue grey red brown*	*blue grey black brown brown*
1 4.7kΩ	*yellow violet red brown*	*yellow violet black brown brown*
1 1.3kΩ	*brown orange red brown*	*brown orange black brown brown*
1 130Ω	*brown orange brown brown*	*brown orange black black brown*

Case components (if required – see text)
1 laser-cut clear acrylic case (6 pieces) [SILICON CHIP Part No SC4609]
4 small self-adhesive rubber feet
2 M3 x 15mm Nylon machine screws
4 M3 Nylon nuts
2 M3 Nylon washers
4 M3 x 25mm machine screws
2 M3 x 32mm machine screws
2 M3 x 15mm tapped spacers
4 M3 nuts



Scope7: measuring a 0.5H inductor in RL mode. Again, the cycle is repeated about once a second. The effects of the inductor's intrinsic resistance can be seen in that the voltage across the inductor does not fall completely to 0V.



Scope8: measuring a 26mH inductor in RL mode. Note that the spikes are very brief (of the order microseconds) and the slightly raised trace indicating the intrinsic resistance of the inductor is significant in this case too.

symbol printed on their bodies. Ensure the lead on this side goes to the pad marked "+" on the PCB, ie, closer to inductor L1.

We recommend that you solder IC1 directly to the board (ie, don't use an IC socket) to avoid socket contact resistance. Ensure the notched end is facing the top of the board before soldering it in place, as shown in Fig.2.

Now mount reed relays RLY1-RLY4. They are all identical and all have the same orientation. While they are in DIL packages, they should be soldered directly to the PCB. Again, this is to avoid socket contact resistance.

In each case, pin 1 faces towards the right-hand end of the PCB, away from the test sockets.

SIL socket CON1 is provided to allow small components to easily be tested as their leads can simply be pushed into the socket spring contacts. Take a two-way header socket and bend its pins by 90° close to the socket body, then solder it so that the socket projects off the end of the PCB. You can now solder the two banana sockets in place; CON3 is red while CON2 is black.

Mount the four-way and two-way header sockets for CON4 and CON5 next, along with the pin header for JP1. You can use the I^{2}C module board as a jig to line up the sockets correctly. You may also need to straighten out the pins on the I^{2}C breakout board, so they are facing downwards.

The final step in the shield construction is installing the headers to connect the shield to the Uno. We have used stackable headers, as they are slightly longer, giving some clearance between the Uno's USB socket and the TP+ test point.

The easier way to install the headers accurately is to assemble the headers, shield and Uno together and use the Uno's headers to line up the shield headers. Then turn it all upside down so that the shield rests on the headers as far away from the Uno as possible.

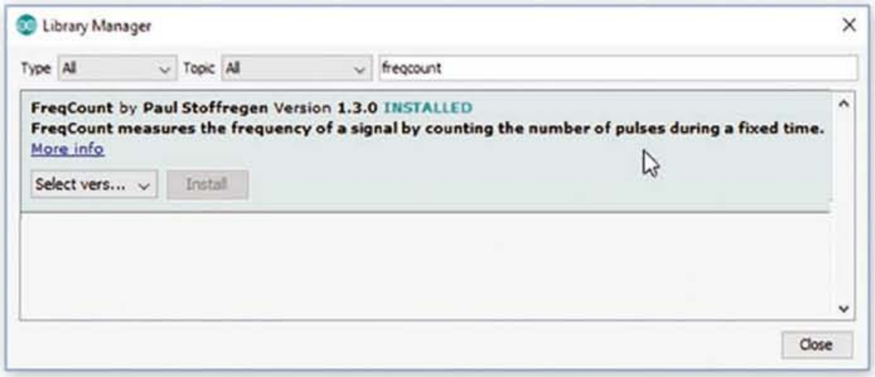
Tack solder the corner pins in place, then remove the Uno main board to allow easier access to the rest of the pins for soldering. Then refresh the corner pin solder joints.

If you don't already have the I^{2}C breakout board fitted to the LCD, carefully line up pin 1 of the LCD with the I^{2}C header end of the adaptor.

Solder one pin, then confirm that the boards are parallel and straight but not touching anywhere except the header before soldering the rest of the pins. The LCD assembly can now be plugged into CON4 and CON5. Alternatively, run four breadboard jumper leads between the I^{2}C adaptor and CON4 for testing.

If you haven't yet adjusted the contrast on the LCD, this is easier to do when the board is connected by jumper leads.

Finally, push the assembled shield onto the Uno. This completes construction. Plug the Uno into your computer using a standard USB cable.



FreqCount by Paul Stoffregen Version 1.3.0 INSTALLED
FreqCount measures the frequency of a signal by counting the number of pulses during a fixed time.
More info

Fig.3: this shows how to find and install the frequency counter library in the Arduino IDE Library Manager. Type "freqcount" in the box at the upper right-hand corner of the window and then click the Install button that appears below (it's grey here because we have already installed it).

Loading the software

To compile and upload the software that runs on the Uno board, you need to have the Arduino IDE (Integrated Development Environment) software installed on your computer. The IDE can be downloaded from www.arduino.cc/en/main/software and it is available for Windows, macOS X and Linux. Download and install a version to suit your operating system and start the software.

The "sketch" or program that runs on the Uno needs two external libraries. One is used to count the pulses that are generated by the oscillator and the other to interface to the I^{2}C LCD.

They are both supplied as ZIP files in the download package, along with the sketch itself.

Installing the first library is as simple as going to the Sketch -> Include Library -> Manage Libraries... menu, and searching for "freqcount", and clicking on the install option that is presented (see Fig.3).

To install the second library, search for "liquidcrystal_pcf8574" in the Library Manager, and install the version by Matthias Hertel.

Now open the sketch file, select "Arduino/Genuino Uno" under the Tools -> Board type... menu and then use the Tools ->Port menu to select the serial port that the Arduino is plugged into. Most versions of the Uno will display as COMx: (Arduino/Genuino Uno) in the dropdown menu, so you can use this hint to find the correct serial port if you are unsure.

Press Ctrl-U to compile and upload the sketch. If you see the message "Done Uploading" at the bottom of the window then everything has compiled and uploaded successfully.

If you get an error message, check that the libraries are installed correctly, and check that the correct serial port is selected.

Testing and set-up

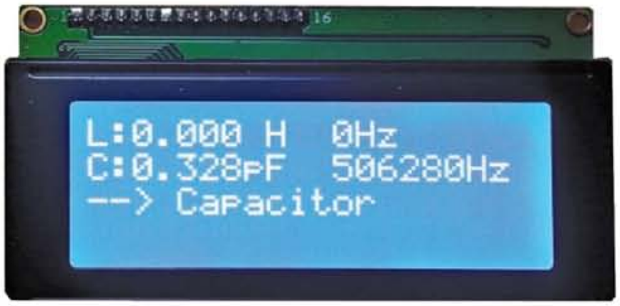
If there is no back-light on the LCD, check that the LCD back-light jumper is installed on the I^{2}C breakout board. If you are using the back-light header for mounting, the jumper is installed on the two pin header next to the mounting header.

If the back-light is working but there is no text, check and adjust the contrast pot on the back of the I^{2}C breakout board. If you have no text or faint text, trying turning the pot clockwise. If you can only see white squares, try turning the pot anti-clockwise.

You should see text similar to that shown in Fig.4.

The unit stores calibration data in EEPROM. The first time you power it on after uploading the sketch, it will load a sensible set of defaults so you can start using it straight away. In the unlikely case that this does not happen, you can reset the calibration data

Fig.4: typical display on the LC Meter with no component connected. The small amount of residual capacitance shown can be adjusted for in calibration.



using the following procedure:

Open the serial monitor at 115,200 baud, and type "C" followed by Enter. When the menu appears, press "L", Enter, "S", Enter, then "X", Enter, and press the reset button on the Uno.

Now attach a component to the test terminals and check that you get a reading of its value.

Note that polarised components, such as electrolytic and tantalum capacitors should match their polarity to the test terminal markings. The test terminals (CON1, CON2/CON3) may have up to 5V present, so take care not to attach any components with lower ratings.

You may wish to improve the accuracy of the meter by measuring and entering specific values into the calibration values (assuming you have the means to do so). The values of C2 and the 130Ω and 1.3kΩ resistors are initially assumed to be very close to expected.

If they are not exact, and you have the means to measure them, you can improve the unit's accuracy by entering these into the calibration data, as we'll explain later.

Using it

On startup, the Uno performs the same calibration tests as the previous LC meter, storing the F1 reference oscillator frequency (just C1) and the F2 calibration frequency (C2 in parallel with C1).

During this time, the six calibration constants which are stored in EEPROM are displayed as they are loaded.

Initially, instead of waiting a full second to count the number of cycles out of the oscillator, the meter only counts for 100ms to save time.

This initial measurement is simply used to detect whether an inductor or capacitor is connected and whether its value falls in the range best measured by the time-constant method or the oscillator method.

If the frequency appears to be stable for two consecutive readings, the LC meter performs another test each cycle, this time taking one second for improved accuracy.

As long as the component remains connected, an average value is displayed by accumulating the results and dividing by the number of samples recorded. In this way, a highly accurate reading can be made.

The LC meter uses the first two lines to display its initial estimate for inductance and capacitance.

It displays both, as we found there were a small number of cases (with very large capacitance and inductance values) where the LC meter would detect one type of component as the other.

The measured oscillator frequencies are displayed to assist the user in following the operation of the LC meter. The third line displays whether the component is a capacitor or inductor, while the fourth line shows the averaged values and number of samples taken.

To use, simply connect the component across the leads and allow the reading to stabilise. Check that the component has been properly identified in the third line, and if you need an accurate value, allow a few readings to be recorded and read the average displayed on the fourth line.

Manual calibration

To access the calibration constants, you will need to connect to the serial port using a terminal program such as the Arduino IDE's serial monitor. The baud rate is 115,200 baud, with the standard Arduino defaults of 8bits, no parity and one stop bit.

After opening the terminal program send a "C" (capital) to enter calibration mode, pressing Enter if necessary to trigger sending of the line of data. The LC meter may take second or two to respond, as it only checks the serial port once every test loop.

The following menu appears:

```
Calibration Mode:
A:Enter R12 value – 130Ω
B:Enter R11 value – 1.3kΩ
C:Enter C2 value
D:Enter L1 value
E:Enter Cparasitic value
F:Enter Lparasitic value
G:Auto detect Cparasitic
  (leave terminals open circuit)
H:Auto detect Lparasitic
  (short circuit leads)
L:Load defaults
P:Print current values
S:Save to EEPROM
X:Exit calibration
Choose an option
```

So the procedure is select one of the 12 options from A to X.

Options A-F correspond to each of the six calibration constants. For A-D, the best way to improve the calibration is to measure the value of the component with an accurate meter and enter it. For example, to change the value of C2 to 1.1nF, type "C" (and Enter if necessary). You will be prompted:

```
C selected.
Enter a value:
```

Type a value, including any of the SI multipliers from p (pico) to G (giga). For 1.1nF, we simply type "1.1n", with the units being assumed. If the wrong units are included, or you enter a negative number, an error message will appear. Otherwise, you will see:

```
0.0000000011000F
C2 changed to 1.100000nF
```

The LC meter displays the entered value both with and without SI multipliers for clarity. At this point, the value is loaded into the program and will be used for measurements but it will not be saved in EEPROM for later use. To save the changes, select option "S".

See text on pages 35-36 for a full description of the LC Meter's modes.

Mode	RLY1	RLY2	RLY3	RLY4
1. Calibration (C1 only)	OFF	ON	OFF	OFF
2. Calibration (C1 and C2)	ON	ON	OFF	OFF
3. Capacitor measurement	OFF	ON	ON	OFF
4. Inductor measurement	OFF	OFF	OFF	ON
5. Time constant mode	OFF	OFF	OFF	OFF

Table.1: relay states in each test mode

Measuring the value of large inductors and capacitors

If we connect a large inductor or capacitor to our test oscillator, it will fail to oscillate. Hence, we need an alternative method to handle these sorts of components to make the Meter truly useful. The simplest method is charge time measurement and we use digital output pins A0-A2, two fixed resistors (130Ω and 1.3kΩ) and analog input pin A3 to perform this function.

If we know the value of the resistor and capacitor in a series RC circuit, we can calculate its "time constant". This is simply the product of the resistance in ohms and the capacitance in farads, giving a value with units of seconds.

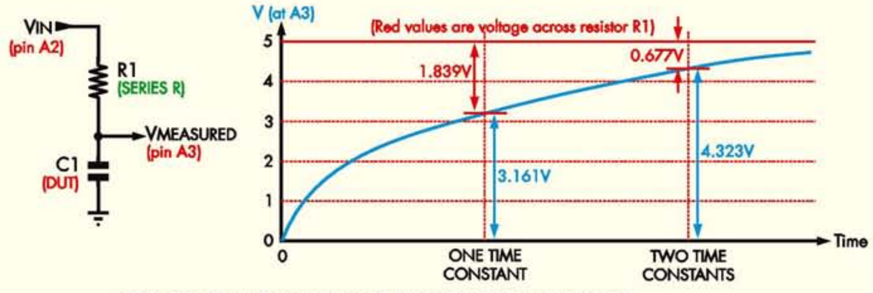


Fig.5: when testing high-value capacitors, C1 is charged via R1 and the voltage at pin A3 follows a curve similar to the blue one shown here. By measuring the voltage twice, and the time interval between measurements, we can determine its capacitance.

Refer to Fig.5. This shows the resistor and capacitor connected in the classic low-pass filter type arrangement. Assuming the capacitor starts fully discharged, the full applied voltage (Vin) appears across the resistor. After a period of one time constant, the voltage across the resistor will have fallen to Vin/e, where e is Euler's constant (2.718...).

In fact, regardless of what the voltage across the resistor is at the start, the ratio of the voltage at the start of the time constant period to the voltage at the end will be e.

This means we don't necessarily have to have the capacitor fully discharged to make our measurement, although we do get better accuracy if we work near this end of the curve, given the ~5mV resolution of the 10-bit ADC in the ATmega328 micro ($5V \div 2^{10}$).

To explain further, measuring over the first time constant period, the voltage across the resistor will drop from a nominal 5.000V down to 1.839V, a change of 3.161V or 647 ADC steps. Over the second time constant period, the voltage will change from 1.839V to 0.677V, a change of 1.163V or 238 ADC steps. Hence, starting with a mostly-discharged capacitor gives us better measurement resolution.

We don't even need to measure an exact multiple of the time constant to complete the calculations, as we can use a logarithmic function to convert a ratio of voltages into a corresponding number of time constants.

For example, if we measure the voltage across the resistor at two different times and compute the ratio between the two of 4.95, this is equivalent to $e^{1.6}$ and that tells us that the period between the two measurements is equal to 1.6 time constants. So if the period happens to be 80ms, we can calculate the time constant is 50ms.

And if we know the value of the resistance, then we can easily compute the capacitance. In this case, if the resistance is 1.3kΩ, since t = RC, C = t / R or 38.5µF.

A similar method for measuring the value of an inductor is shown in Fig.6. In this case, the curve is reversed and the equation for the time constant is the inductance in henries divided by the resistance in ohms.

Unlike capacitors, the ohmic resistance of a typical inductor can be quite significant and has to be taken into account in the calculation. With a capacitor, the voltage across the resistor will eventually get very close to 0V, only falling short due to leakage current, which is normally quite small. But with the inductor, in the steady state, the current is at a maximum and the voltage across it can be quite large.

Regardless, the method used is substantially the same; our starting state is with the inductor short-circuited to ensure no current is flowing, after which the series resistor is connected to 5V and the measurements of time and voltage begin.

After the time period has been measured, the series resistor is used to help determine the ESR of the inductor, and this is added to the series resistor's value before the time constant calculations are made, for improved accuracy.

Practical measurement ranges

For capacitors above 1µF with a series resistor around 1kΩ, the time constant is around 1ms, which is well within the realm of what an Uno can measure with reasonable accuracy. This design has no theoretical upper limit to what capacitance it can measure, given enough time. In practice, we had no trouble reading capacitor values up to 100,000µF. (That's as big as we had!).

We also tried measuring really high-value inductors (in the henries). The oscillator-based method actually works with many of these but its operation does malfunction sometimes. Our time-constant method consistently gave results within the spread measured by an expensive commercial LC meter.

Note that the same commercial LC meter, using different test frequencies, gave wildly different values for inductance, varying by 10% or more. So it seems that the values of some components can vary quite dramatically depending on the frequency that they are tested at.

Keep that in mind when using this L-C Meter as the oscillator-based measurements are not at a fixed frequency.

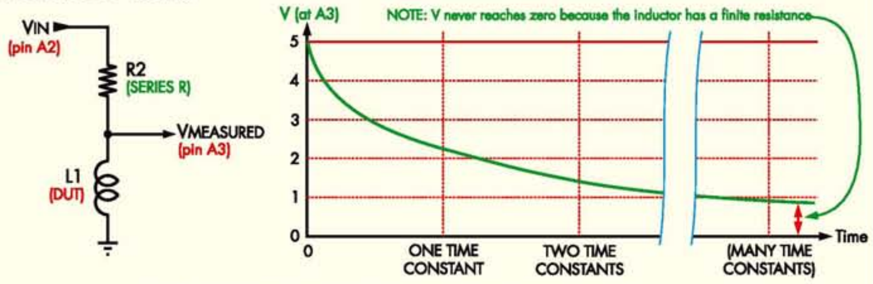
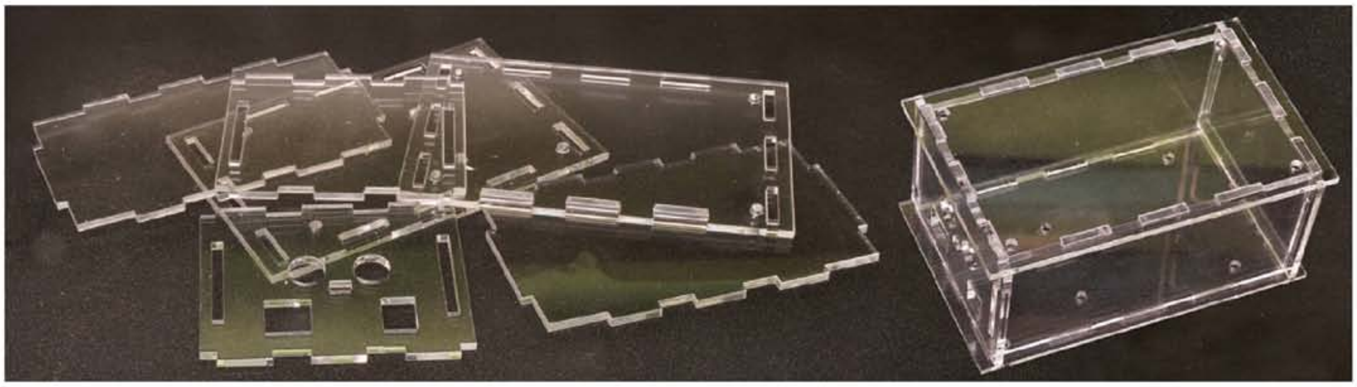


Fig.6: similarly, when testing high-value inductors, the voltage at A3 initially starts out at 5V and drops to a value determined by its series resistance once L1's magnetic field is fully charged. By measuring the voltage twice, and the time interval between measurements, we can determine its inductance.



On the left are the six pieces which make up the case specifically designed for this LC Meter. They simply slot together and the PCB mounting screws hold them in place. The assmbled case is shown on the right. No cutout is necessary for the display as the case is clear acrylic. This case is available from the SILICON CHIP Online Shop for just $7.50 (Cat SC 4609)

Options "G" and "H" can be used to automatically detect stray capacitance and inductance.

When using option "G" , ensure that no components are attached, although it is fine to leave leads attached if they are normally used for measuring – they will contribute to stray capacitance, so this can be accounted for in this calibration. If you have leads, make sure they are open-circuit.

Option "G" simply runs a capacitance measurement on the leads (if connected), then saves this value to null future measurements.

Option "H" works in a similar fashion, although a shorting bar will need to be installed to create the effect of a zero inductance.

If you are using leads to measure in inductance mode, they can be shorted before selecting this option. Again, the value is saved and used to null inductance measurements. As with other values, these will need to be saved with option "S".

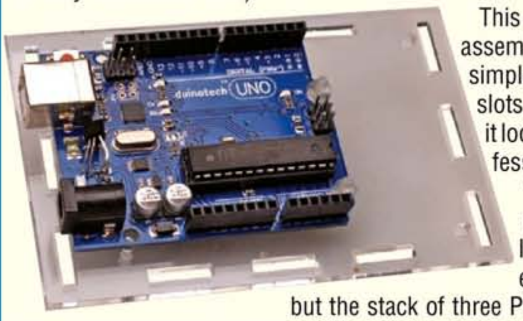
Option "L" loads sensible default values (but does not save them to EEPROM) and gets you back to the initial condition.

Option "P" displays the current values in use, while "X" returns to measurement mode.

You will notice that the LCD also echoes what is occurring on the serial monitor. SC

Assembly in the SILICON CHIP acrylic case

This assumes that you are using the recommended purpose-designed clear acrylic case* (available from the SILICON CHIP Online Shop at siliconchip.com.au/shop – cat no is SC-4609; price is very reasonable too!).



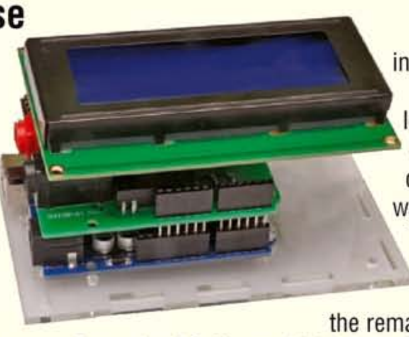
This case makes assembly so much simpler. It simply slots together. And it looks really professional!

You *could* use a jiffy box like our earlier L-C meters but the stack of three PCBs does not easily lend itself to a UB3 case – and you also have a number of slots/holes to prepare.

The SILICON CHIP case has all required slots and holes already prepared for you.

Carefully remove the protective film from the acrylic pieces, and mount the LCD module to the underside of the top panel (which has four holes to suit the LCD module) using the 25mm machine screws and M3 nuts.

Thread the tapped spacers onto the end of the rightmost (furthest away from the I²C breakout board) machine screws by about 4mm.

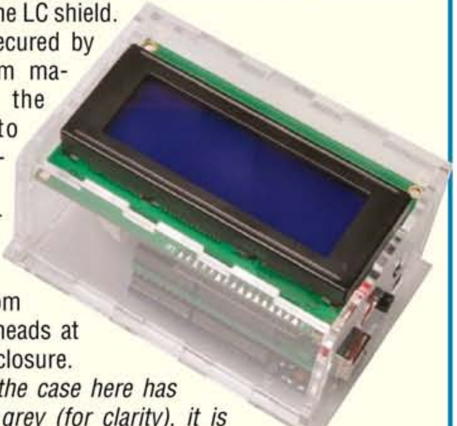


Attach the 15mm Nylon machine screws to the bottom panel in the middle holes using one Nylon washer and nut on each. This acts as a spacer.

Attach the Uno to the machine screws using the remaining two Nylon nuts.

Press the LC Meter shield onto the Arduino. Slot the end piece with the holes onto the Arduino, and drop it into the slots in the bottom panel, then fit the other three side panels so that the sides are all resting in the base.

Set the top onto the sides, ensuring the LCD headers push into the headers on the LC shield.



The halves are secured by threading the 32mm machine screws into the tapped headers to sandwich the assembly together.

We also recommend adding self-adhesive rubber feet to protect your desk from the exposed screw heads at the bottom of the enclosure.

**Note that while the case here has been photographed grey (for clarity), it is actually crystal clear, as seen at top of page.*