Arduino-based Digital Inductance/Capacitance Meter

Do you ever need to check or confirm the values of inductors or capacitors? This Arduino-based LC meter will give you a digital readout and can even measure parasitic inductance or capacitance present in a circuit. It's much more accurate than most DMM-based LC meters.

Many digital multimeters (DMMs) have capacitance ranges but they are not normally accurate for values below about 50pF. And those few DMMs that can measure inductance are often not very good at measuring inductance in the range of 1-100µH – those that are typically used in audio and RF circuits.

An inductance meter with a 10µH resolution (typical for DMMs) isn't very helpful if you want to wind a choke of say 6.8µH, for an amplifier output filter.

Professionals tend to rely on digital LCR meters for these types of measurements. They allow you to measure almost any passive component quickly and automatically, often measuring not just their primary parameter (like inductance or capacitance) but one or more secondary parameters as well. However, many of these you-beaut instruments also carry a hefty price tag, keeping them well out of reach for many of us.

Fortunately, thanks to microcontroller technology, much more affordable digital instruments are becoming available. These include both commercial and DIY instruments like the low-cost unit described here.

By JIM ROWE

Essentially it's an improved version of the PIC-based Digital LC Meter we described in the May 2008 issue of SILICON CHIP. This time, we're basing it around an Arduino Uno or equivalent module.

Main features

Our new Digital LC Meter is compact and easy to build, since the Arduino board comes pre-assembled. It also has a better LCD readout than the previous version. It fits snugly inside a UB3 utility box and you should be able to build it for under $100.

It offers automatic digital measurement of both inductance (L) and

capacitance (C) over a wide range and with 5-digit resolution. Measurement accuracy is better than ±1% of reading over most of the ranges.

It operates from 5V DC, drawing an average current of about 62mA, so it can run from a 5V USB supply (either mains or battery) or from a spare USB port on your PC.

How it works

The meter's impressive performance relies on an ingenious measurement technique developed almost 20 years ago by the late Neil Heckt in the USA.

It uses a wide-range test oscillator and its frequency is varied by connecting the unknown inductance or capacitance you're measuring. The resulting change in frequency is measured by the microcontroller and used to calculate the component's value, which is displayed directly on a small LCD panel.

To achieve reliable oscillation over a wide frequency range, the test oscillator is based on an analog comparator with positive feedback around it, as shown in Fig.1. This configuration has a natural inclination to oscillate, because of the very high gain between the comparator's input and output.

When power (+5V) is first applied, the comparator's positive input is held at +3.3V by the divider formed by the two 100kΩ resistors and the 100kΩ and 4.7kΩ resistors. Initially, the voltage at the negative input is zero because the 10µF capacitor at this input needs time to charge via the 47kΩ resistor.

So with its positive input much more positive than the negative input, the comparator initially switches its output high, to near +5V.

Once it does so, the 10µF capacitor connected to the negative input begins charging up via the 47kΩ resistor and the voltage at this input rises. As soon as it goes above +3.3V, the comparator output switches low and the positive input is brought to 1.67V due to the 100kΩ feedback resistor pulling the 100kΩ divider low.

The low comparator output voltage is also coupled through the 10µF input capacitor to the tuned circuit formed by inductor L1 and capacitor C1. This makes the tuned circuit "ring" at its resonant frequency.

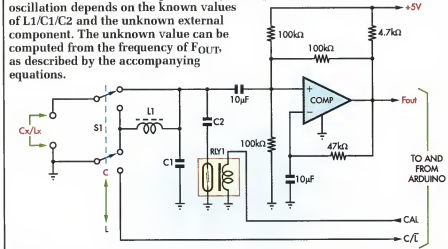
As a result, the comparator and the tuned circuit now function as an oscillator at that resonant frequency. In effect, the comparator functions as a negative resistance across the tuned circuit, to cancel its losses and maintain oscillation. Once this oscillation is established, a square wave of the same frequency is present at the comparator's output and it is this frequency (F_{OUT}) that is measured by the microcontroller.

In practice, before anything else is connected to the circuit, F_{OUT}, will simply correspond to the resonant frequency of the tuned circuit comprising L1, C1 and any stray inductance and capacitance that may be associated with them.

When power is first applied to the circuit, the microcontroller measures this frequency (F1) and stores it in memory. It then energises reed relay RLY1, which switches capacitor C2 in parallel with C1 and thus lowers the oscillator frequency. The micro then measures and stores this new frequency (F2).

Next, the micro uses these two frequencies plus the known value of C2 to accurately calculate the values of both C1 and L1. The equations it uses to do this are shown in Fig.1. Following these calculations, the micro

Fig.1: operating principle of the LC Meter. L1 and C1 form a tuned circuit in combination with an external capacitance/inductance connected via S1. Feedback for oscillation is provided by a comparator and the frequency of oscillation depends on the known values of L1/C1/C2 and the unknown external component. The unknown value can be computed from the frequency of F_{OUT}, as described by the accompanying equations.



HOW IT WORKS: THE EQUATIONS

(A) In calibration mode

(1) With just L1 and C1: $F1 = \frac{1}{2.\pi.\sqrt{L1.C1}}$

(2) With C2 added to C1: $F2 = \frac{1}{2.\pi.\sqrt{L1.(C1+C2)}}$

(3) From (1) and (2), we can find C1:

$$C1 = C2 \cdot \frac{F2^2}{(F1^2 - F2^2)}$$

(4) Also from (1) and (2), we can find L1:

$$L1 = \frac{1}{4.\pi^2.F1^2.C1}$$

(B) In measurement mode

(5) When Cx is connected: $F3 = \frac{1}{2.\pi.\sqrt{L1.(C1+Cx)}}$

so $Cx = C1 \cdot \left(\frac{F1^2}{F3^2} - 1\right)$

(6) Or when Lx is connected:

$F3 = \frac{1}{2.\pi.\sqrt{L1+Lx).C1}}$

so $Lx = L1 \cdot \left(\frac{F1^2}{F3^2} - 1\right)$

NOTE: F2 & F3 should always be lower than F1

Features & specifications

Inductance range:	10nH to 100mH+
Capacitance range:	0.1pF to 2.7µF+ (non-polarised only)
Measurement resolution:	five digits in either mode
Range selection:	automatic
Sampling rate:	approximately one measurement per second
Accuracy (when calibrated):	±1% of reading, ±0.1pF or ±10nH
Supply voltage:	5V DC @ <65mA (including backlit LCD)
Supply type:	USB charger or the USB port on a PC

turns RLY1 off again to disconnect C2, allowing the oscillator frequency to return to F1. The unit is now ready to measure the unknown inductor or capacitor (L_x or C_x).

As shown in Fig.1, the unknown component is wired to the test terminals at far left. It is then connected to the oscillator's tuned circuit via switch S1.

When measuring an unknown capacitor, S1 is switched to the "C" position so that the capacitor is connected in parallel with C1. Alternatively, for an unknown inductor, S1 is switched to the "L" position so that the inductor is connected in series with L1.

In both cases, the added L_x or C_x again causes the oscillator frequency to change to a new frequency (F3). As with F2, this will always be lower than F1.

So by measuring F3 as before and monitoring the position of S1 (which is done via the C/$\overline{L}$ line), the micro can calculate the value of L_x or C_x using one of the equations shown in the right section of the equations box in Fig.1.

From these equations, you can see that the micro has some fairly solid number-crunching to do, both in the calibration mode when it works out the values of L1 and C1 and in the measurement mode when it must work out the value of L_x or C_x.

Each of these values needs to be calculated to a high degree of resolution and accuracy, using floating-point maths.

Circuit details

The full circuit diagram is shown in Fig.2. It mainly consists of the Arduino microcontroller module and the serial I²C LCD module together with the oscillator circuit we've already introduced, built using an LM311 high-speed comparator (IC1).

The Arduino controls RLY1 to switch calibrating capacitor C2 (1nF) in and out of circuit, via its IO3 pin. Diode D1 is connected across the relay coil to prevent the Arduino's internal circuitry from being damaged by inductive spikes.

The Arduino senses which position L-C switch S1 is in using its IO2 pin, which is pulled high internally when it's not pulled low by S1b (in the L position).

The output of the oscillator at pin 7 of IC1 is taken to pin IO5 of the Arduino via a series 6.8kΩ resistor. It needs to be taken to this pin because this is also the external input pin for the 16-bit timer/counter inside the ATmega328P micro which forms the heart of the Arduino Uno.

As a result, we are able to use the Arduino to easily measure the oscillator's frequency.

The results of the Arduino's measurements and calculations are displayed on a blue back-lit 16x2 alphanumeric LCD module.

This has a serial I²C module fitted, so it can be controlled from the Arduino via its I²C port lines (SCL and SDA).

Its features were fully described in SILICON CHIP March 2017 issue.

Calibration functions

The firmware sketch running in the Arduino is designed to perform its "zero calibration" adjustment just after initial startup.

But pushbutton switch S2 is also provided to allow zero calibration to be performed at any other desired time as well (to allow for temperature drift, for example).

S2 pulls the Arduino's $\overline{\text{RESET}}$ pin (pin 4) down, so that it is forced to reset and start up again, readjusting its zero setting in the process.

LK1 and switch S3 can be used to nudge or tweak the calibration in small increments or decrements, if you have access to an accurate reference capacitor. When LK1 is fitted, pulling input pin IO7 low, the micro will increase

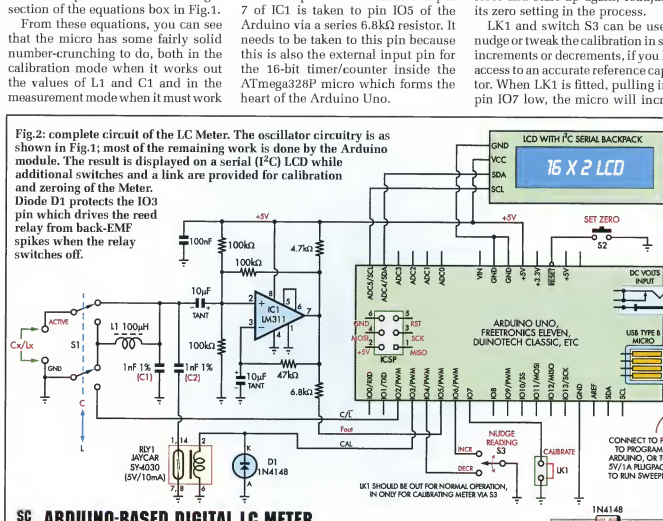


Fig.2: complete circuit of the LC Meter. The oscillator circuitry is as shown in Fig.1; most of the remaining work is done by the Arduino module. The result is displayed on a serial (I²C) LCD while additional switches and a link are provided for calibration and zeroing of the Meter. Diode D1 protects the IO3 pin which drives the reed relay from back-EMF spikes when the relay switches off.

the capacitance reading by about 0.5% each time S3 (a centre-off rocker switch) is pushed to the upper "INCR" position, or alternatively decrease the reading by the same amount if S3 is pushed to the lower "DECR" position.

So the idea is to push S3 in one direction or the other until the reading is correct.

Each time a change is made, the adjustment factor is stored in the Arduino's EEPROM memory, so it's remembered for future sessions. When link LK1 is not fitted, pressing S3 in either direction has no effect at all.

This is a safety feature, to prevent unintended changes to the meter's calibration during normal use. Although this calibration is normally done using a reference capacitor, it also improves the accuracy of inductance measurements.

Construction

There is no custom PCB used for the LC meter's circuitry; instead, most of the added circuitry is fitted on a prototype shield board which simply plugs into the top of the Arduino PCB.

There aren't that many components involved, so it's a straightforward job to wire it up as shown in the wiring diagram, Fig.3.

The only components which are not mounted on the ProtoShield are the serial LCD module, switches S1-S3, the test terminal binding posts and reference components L1 and C1.

As shown in Fig.3 and the photos, these are all mounted on the lid of the UB3 box, which forms the meter's front panel. These off-board components are all linked to the ProtoShield board via short multi-wire interconnection leads and SIL connector plugs and sockets, which are also shown in Fig.3.

You can get an idea of how everything fits together from the internal cutaway diagram of Fig.4, along with the internal photos.

The Arduino module mounts in the bottom of the box via four 9mm long M2.5 machine screws and four M2.5 nuts, with another four M3 or M2.5 Nylon nuts used as spacers.

The ProtoShield is plugged into the top of it. The rest of the meter circuitry connects via the 90° pin headers on the ProtoShield.

Follow the wiring diagram (Fig.3) and internal photos to build the ProtoShield. Start by soldering the components into place where shown

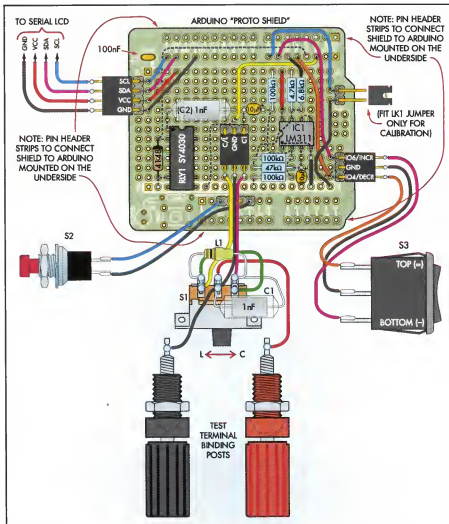
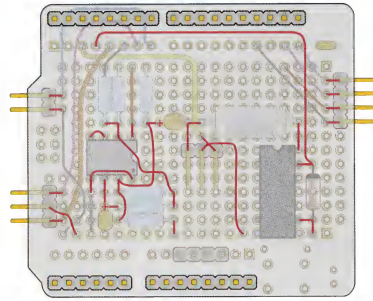
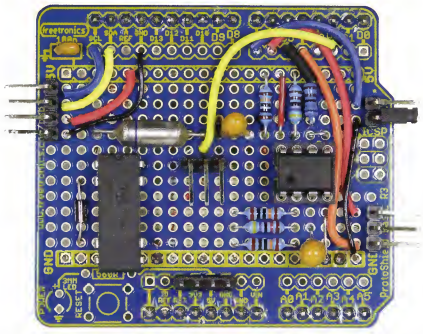


Fig.3: follow these diagrams to fit the components to the ProtoShield and also to wire up all the external connections. Connections made between component pads on the underside are shown below, significantly larger than the 1:1 diagram above, for clarity. These should be made with insulated wire to avoid short circuits.





"Larger than life" photo of the wiring on the top side of the Freetronics Arduino ProtoShield board (actual size is shown in Fig.3, below). This board "plugs in" to the Arduino Uno (etc) board via the rows of pin headers on the underside; the I²C LCD board plugs into the ProtoShield board.

in Fig.3, ensuring you use the correct orientation for polarised components: IC1, diode D1, RLY1 and the two 10µF tantalum capacitors.

Next, add the wiring on the underside, as shown in the underside wiring diagram of Fig.3. Use insulated wire because several of these wires cross over each other.

In cases where adjacent pads are connected, you can simply place a solder bridge between the two pads or alternatively, bend the component leads while fitting them and trim them so that they reach the adjacent pads. For longer connections, use component lead off-cuts, routed carefully to avoid the possibility of shorting anything else, or short lengths of light-duty hookup wire (eg, stripped from a piece of ribbon cable) or bell wire.

Here's our suggested order of fitting the components and wiring the ProtoShield board; check Fig.3 for the exact placement in each case:

1. Fit the four 90° SIL headers.
2. Fit a four-pin vertical header for switch S2.
3. Fit the four SIL pin headers to the underside, along the upper and lower edges of the ProtoShield, which connect it to the Arduino. These comprise a 10-pin header at upper left, two 8-pin headers (one at upper right and the other at lower centre) and a 6-pin header at lower right. Do not fit a 3x2 DIL pin header in the ICSP position at centre right on the ProtoShield board.
4. Fit the 8-pin DIL socket for IC1, with its notched end to the left, then relay RLY1, with its notched end towards the top.
5. Mount the six resistors, the 100nF capacitor and the two 10µF tantalum caps. Note that the last two are polarised, so make sure you fit them with the orientation shown.
6. Fit the 12 insulated wires on the top of the board and any insulated wires required to complete the wiring on the underside. This will require you to strip the insulation from each end by about 5mm or so.
7. Fit diode D1, making sure its end with the cathode band is uppermost and adjacent to pin 2 of RLY1, then plug IC1 into its socket.

Box and lid preparation

There are four holes to drill in the bottom of the box for mounting the Arduino module and two larger holes to cut in the left-hand end for the USB plug and alternative DC power plug.

The locations and dimensions of

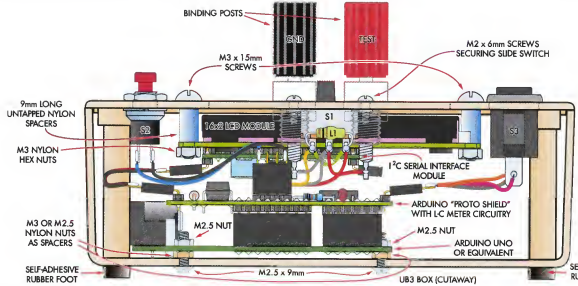


Fig.4: this shows how everything fits together inside the UB3 "Jiffy" box. The Arduino module is attached to the bottom of the case with the proto-board hosting most of the remaining circuitry plugged on top. The three switches, two binding posts and the I²C LCD module are mounted on the lid and connected to the ProtoShield via flying leads.

The Freetronics Eleven (Uno equivalent) board, mounted in the bottom of the case (see drilling template on pages 35 and 36).

all of these holes are shown in Fig.5, the drilling template, while the corresponding information for the holes to be drilled and cut in the lid/front panel are shown in Fig.6.

For best results, start the larger holes with a smaller pilot drill and enlarge with a stepped drill bit, series of larger drill bits or a tapered reamer. Rectangular or other non-round holes can be made by drilling a series of holes, knocking out the centre section and then filing the hole to shape.

We fixed four self-adhesive rubber feet to the underside of the box to protect any surface it's placed on.

Making all the required holes in the lid is rather tedious as there are twelve, including three rectangular cut-outs and two holes with flat edges.

To save time and guarantee a neat result, you can purchase a laser-cut clear acrylic lid (which replaces the lid supplied with the box) from the Silicon Chip online shop (see parts list).

As the acrylic panel is transparent the lid doesn't need a cut-out to view the LCD. Note that since the 3mm acrylic is slightly thicker than the lid supplied with the UB3 box, depending on the length of the screws that came with it, you may need to use slightly longer self-tapping screws to attach it.

We have also prepared artwork for the front panel, to give it a professional look. You can download this as a PDF file from the Silicon Chip website.

There are two ways to go here: after you print it, it can be hot laminated, then attach it to the box lid using double-sided adhesive tape or spray glue. After that, you can cut out the holes in the front panel to match those in the box lid using a sharp hobby knife.

Or, for longest life and an even more professional finish, consider fitting the label to the *underside* of the lid – it's more fiddly to fit but doesn't require laminating, nor double-sided tape to hold it in place (the switches and terminals hold it in position; a very light mist of clear spray adhesive will also ensure it stays tight against the lid).

Perhaps it's gilding the lily somewhat but if you can print the label onto clear film, you can see the "works" through the label, as we did with the photo on page 28.

Just make sure you get the right film to suit your type of printer (eg, laser printer or inkjet printer, etc).

Once the lid/front panel is finished, fit switches S1-S3 to it, along with the two test terminal binding posts and the serial LCD module.

Slide switch S1 attaches to the front centre of the lid via two 6mm long M2 machine screws, while switch S2 mounts using the spring washer and nut supplied with it and S3 simply pushes into its rectangular mounting hole until its two barbs spring outwards to hold it in place.

Just make sure that you fit it with the "=" sign on its rocker actuator uppermost (see photos).

The two binding posts are mounted using the mounting nuts and lock washers provided.

Take care doing so, however, as the upper and lower mounting bushes have D-shaped sections which should mate with the matching holes in the lid/front panel.

The serial LCD module mounts under the lid in the top centre position,

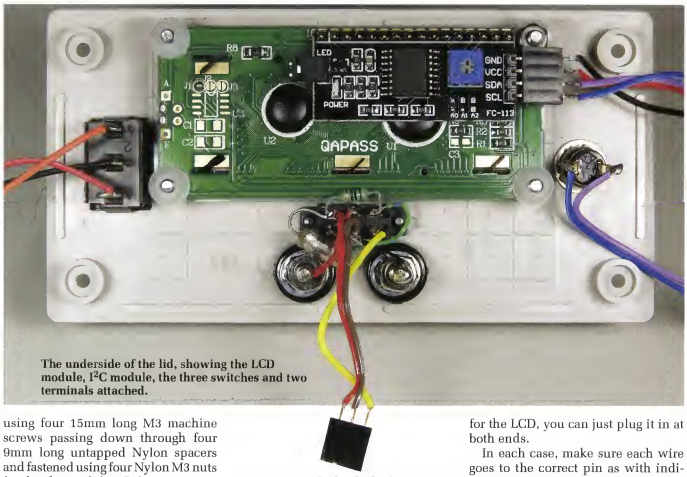
Parts list – Arduino-based LC Meter

1 Arduino Uno R3, Duinotech Classic, Freetronics Eleven or equivalent microcontroller module
1 Serial I²C 16x2 LCD module with back-lighting (Silicon Chip online shop Cat SC4198)
1 Arduino Uno Prototyping Shield (eg, Freetronics SH-Proto-Basic)
1 UB3 "Jiffy" box, 130 x 68 x 44mm
1 laser-cut clear acrylic lid for UB3 box [optional but recommended] (Silicon Chip online shop Cat SC4274)
4 self-adhesive rubber feet
1 5V/10mA DIL reed relay (RLY1; Jaycar SY4030)
1 100µH axial RF inductor (L1; Jaycar LF1534)
1 DPDT subminiature slide switch (S1; Jaycar SS0821)
1 panel-mount SPST NO momentary pushbutton switch (S2; SP0710)
1 panel-mount SPDT on-off-on momentary rocker switch (S3; Jaycar SK0987)
1 8-pin DIL IC socket
1 40-pin header, 2.54mm pitch
1 40-pin right-angle header, 2.54mm pitch
1 150mm socket-to-socket jumper ribbon cable (Jaycar WC6026)
1 jumper shunt
2 binding posts with integral banana socket (1 red, 1 black)
4 9mm Nylon untapped spacers, 3mm inner diameter
4 15mm M3 machine screws
8 M3 Nylon hex nuts
4 9mm pan head M2.5 machine screws
4 M2.5 hex nuts
2 6mm M2 machine screws (for S1)

Semiconductors
1 LM311 DIP high-speed comparator (IC1; Jaycar ZL3311)
1 1N4148 small signal diode (D1)

Capacitors
2 10µF 16V through-hole tantalum
1 100nF multilayer ceramic
2 1nF 1% NP0 ceramic, mica, MKT, polypropylene or polystyrene (Silicon Chip online shop Cat SC4273)

Resistors (all 0.25W, 1%, through-hole mounting)
3 100kΩ 1 47kΩ 1 6.8kΩ 1 4.7kΩ



The underside of the lid, showing the LCD module, I²C module, the three switches and two terminals attached.

using four 15mm long M3 machine screws passing down through four 9mm long untapped Nylon spacers and fastened using four Nylon M3 nuts (under the module PCB).

With the LCD module in position, your front panel assembly is ready to be wired up and provided with its various leads to connect to the ProtoShield board.

Refer back to Fig.3 and the internal photo, following them carefully to make the correct connections between S1, the test terminal binding posts and L1 and C1 in particular.

Note that the leads of L1 and C1 should be kept as short as possible, to keep stray capacitance low (and stable). You can then make up the various short leads which will connect the front panel components to the ProtoShield board.

Note that the lead which connects S1, L1, C1 and the test terminals to the ProtoShield ends in a three-way SIL header socket, as does the lead from S3.

In contrast, the lead which connects to the serial LCD module has a four-way SIL header socket at each end, while the lead to connect zero/reset switch S2 (although of only two wires) ends in a four-way SIL header socket, with the wires connecting only to the pins on each end.

The two pins in the centre of the socket can be either cut short or pulled out, since they are not used.

Rather than using SIL sockets like we did on the prototype, we suggest you simply split a 40-way ribbon jumper cable with individual "DuPont" sockets on each wire.

This makes the job really easy; you simply pull off the required number of wires and then cut the cable to length and strip the free end, to solder to the switch or connector.

You don't even need to cut the cable for the LCD, you can just plug it in at both ends.

In each case, make sure each wire goes to the correct pin as with individual sockets, it's easy to get them out of order.

Having made up all the required leads, complete the LC Meter assembly with the following steps:

1. Mount the Arduino module inside the bottom of the box using four 9mm M2.5 screws and nuts, using four Nylon M3 nuts as spacers.
2. Plug the LC Meter ProtoShield into the Arduino, making sure you have all four SIL pin headers lined up correctly.
3. Holding the front panel assembly close to the top of the box and orientated correctly, plug the various connection leads into their matching pin headers on the ProtoShield. Be especially careful to get the correct connections between the ProtoShield and the LCD module, as shown in Fig.3.

Resistor Colour Codes

	No.	Value	4-Band Code (1%)	5-Band Code (1%)
❑	3	100kΩ	brown black yellow brown	brown black black orange brown
❑	1	47kΩ	yellow purple orange brown	yellow purple black red brown
❑	1	6.8kΩ	blue grey red brown	blue grey black brown brown
❑	1	4.7kΩ	yellow purple red brown	yellow purple black brown brown

Here's the alternative finish using a paper-printed label fixed to the outside of the UB3 Jiffy box lid, after it has been drilled and cut to suit. (You can, of course, glue a paper label to the laser-cut lid purchased from the Silicon Chip online store). In this case the meter is measuring a nominal 100µH inductor and showing it is slightly high at 103µH.

4. Lower the lid assembly down into the box and fix it into place.
5. Program the Arduino, as described below.

Uploading the firmware

In order to do this, you need to have the Arduino IDE installed on your PC.

The latest version of the IDE can always be downloaded from the Arduino website (www.arduino.cc/en/Main/Software).

At the time of writing, the latest version is V1.8.2, dated 22/03/2017. There are various versions available to suit different operating systems: Windows (32-bit or 64-bit), macOS and Linux (32-bit, 64-bit and ARM).

After the IDE has been installed, download our firmware sketch for the LC Meter from the Silicon Chip website (www.siliconchip.com.au). It's called "Arduino_LC_meter_sketch.ino".

Now plug your LC Meter into one of your PC's USB ports, using a suitable USB cable (usually with a Type A plug on one end, and a micro Type B plug on the other). You may need to install the correct USB VCP driver for it if this is not already installed.

If you're using a Freetronics Eleven module, you can download the appropriate driver from their website (www.freetronics.com.au). All of their drivers are zipped up in a file called "FreetronicsUSBDrivers_V2.2.zip", and there's also a document which explains how to install it.

Once the USB driver has been installed and your operating system confirms that it can communicate with the Arduino in your LC Meter, use Control Panel to find out which COM port the Meter's Arduino has been allocated (eg, COM5, COM7, or whatever).

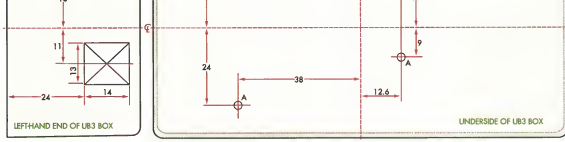


Fig.5: the drilling templates for the four Arduino mounting holes in the bottom of the box along with the USB and DC power access holes in the left-hand end.

Set the port for communication at 115,200 baud with the 8N1 "no handshaking" protocol. The COM port number should be entered into the Arduino IDE's Tools->Port pull-down menu after you start it up.

Now open the LC Meter firmware sketch in the Arduino IDE, verify and compile it, and then upload it into the LC Meter's Arduino flash memory. Soon after it has been uploaded, your meter should spring into life, flashing this message on the LCD screen:

```
Silicon Chip
Digital LC Meter
```

This should remain visible for two seconds, after which the screen should go black, before the Meter begins its initial zero calibration.

If you don't see this initial message, this may be because the contrast trimpot on your LCD display module's serial interface PCB is not set to the correct position.

The remedy is to swing open the lid of the box just enough to fit a very small screwdriver or alignment tool into the trimpot's adjustment slot, turn it and then press switch S2 to force the Arduino to reset and start again.

Try changing the pot setting in one direction or the other until the message becomes clearly visible, pressing S2 after each adjustment.

Startup & calibration

At start-up, the Meter normally expects slider switch S1 to be set in the Capacitance (C) position, and no external capacitor to be connected to the test terminals. If you have done this it will now display the message:

```
S1 set for C: OK
Now calibrating
```

But if you have set S1 in the Inductance (L) position instead, you'll see a different message:

```
Fit shorting bar
Now calibrating
```

This means that the Meter has detected that S1 is set to the L position, and is assuming that you want to do the zero calibration in this mode. As a result, it's advising you to fit a very low inductance shorting bar between the test terminals. This can be in the form of a 40mm long piece of 1.66mm diameter copper or brass rod between the terminals, or (better still) a 40 x 30mm rectangular piece of 1mm thick copper or brass sheet between them. In either case, the rod or sheet must be shiny rather than oxidised.

Note that if you have set S1 to the L position accidentally and don't have a shorting bar available, there's no harm done. Simply flick S1 to the C position and then press switch S2 to get the Meter to reset and begin over again.

Or if you do want to calibrate in inductance mode, simply fit the shorting bar between the terminals (if you haven't already done so) and then press S2 to reset and begin over again.

In either case, there will be a brief pause after which the meter will show the values for C1 and L1 it has found from the initial calibration. This will be something like:

```
C1 = 1084.2 pF
L1 = 91.24 uH
```

The actual values displayed will depend on the components in your unit, as well as the stray capacitance and inductance. They're shown at this stage mainly as reassurance that the Meter is working correctly. The measured values of C1 and L1 will be displayed for three seconds, after which this message will appear:

```
Calibration done
Ready to measure
```

This will display for one second, after which the Meter will begin making measurements. If you have done the initial calibration in C mode and S1 is still in this position but no unknown capacitor is as yet connected to the test terminals, you should now get a display like this:

```
Cx = 0.004 pF
(F3 = 515838 Hz)
```

where the value shown for C_x is very close to zero, while the frequency F3 shown on the second line is for the current oscillator frequency; essentially the same as F1 at the current ambient temperature.

The Meter's oscillator frequency does drift a little with temperature. This means that after a while, the value shown for Cx with no external capacitor connected may creep up from the almost-zero reading you get initially. At the same time, the reading for F3 would slowly decrease.

If you find the value shown for Cx

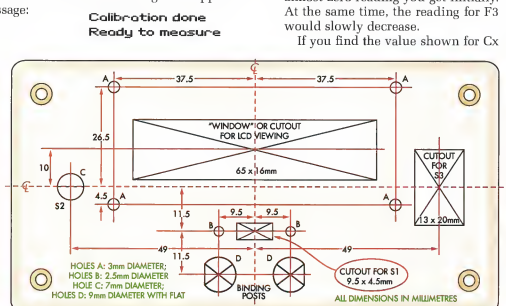


Fig.6: you can either drill and cut the twelve cut-outs required in the lid supplied with the UB3 "Jiffy" box, as shown in this diagram, or (much easier!) purchase a laser-cut acrylic lid from the SILICON CHIP online store and use that instead of the lid that came with the box.

Credit where it's due

As mentioned earlier, this Digital LC Meter, like our earlier May 2008 design, is based on a 1998 design by the late Neil Heckt, of Washington, USA.

Since then, various people have produced modified versions of the design, including Australian radio amateur Phil Rice VK3BHR, of Bendigo in Victoria. Mr Rice and others also modified the firmware and adapted it to use the PIC16F628 micro with its internal comparator. They also added a firmware calibration facility.

So a significant amount of credit for this latest version of the design must go to these earlier designers. The author is happy to acknowledge their earlier work.

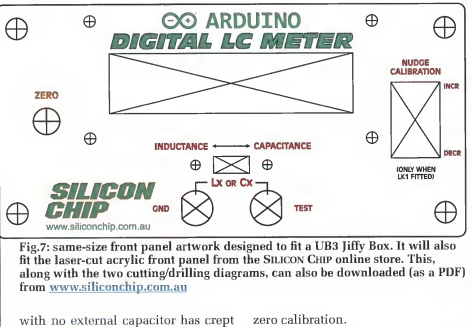


Fig.7: same-size front panel artwork designed to fit a UB3 Jiffy Box. It will also fit the laser-cut acrylic front panel from the SILICON CHIP online store. This, along with the two cutting/drilling diagrams, can also be downloaded (as a PDF) from www.siliconchip.com.au

with no external capacitor has crept up to 0.1pF or more, simply press S2 again to get the Arduino to perform a new zero calibration.

On the other hand, if you've done the initial calibration in L mode and S1 is still in this position but the shorting bar is still connected across the terminals, you should get a display like this:

```
Lx = 0.002 uH
(F3 = 516615 Hz)
```

The value shown for Lx is again very close to zero, and the frequency F3 shows the current oscillator frequency, again very close to F1 at the current ambient temperature. Now if you remove the shorting bar in this mode, you'll find the display will change to something like this:

```
Over Range!
(F3 = 2 Hz)
```

This simply shows that in this mode, an open circuit between the terminals is equivalent to a very high inductance, because it causes the oscillator frequency to drop to near zero.

When you connect a real inductance between the test terminals, the Meter will measure its inductance and display it (assuming its value is within the Meter's range, which is from 10nH to 150mH).

As before, drift in the Meter's oscillator may cause the Lx reading for the shorting bar to creep up gradually. So before making a particularly critical measurement, it's a good idea to fit the shorting bar between the test terminals and press S2 again to force the Arduino to reset and perform a new zero calibration.

Optimising accuracy

If all is well so far, your Digital LC Meter should be operating correctly and ready for use. If you have been able to procure a couple of 1% tolerance (or better) capacitors for C1 and C2, it should also be able to deliver that order of accuracy without any extra calibration.

But as mentioned earlier, it is possible to achieve even better accuracy with the meter providing you have access to a reference capacitor whose value is accurately known (because you've been able to measure it with a high-accuracy LCR meter).

Ideally, this reference capacitor should have a value of between 10nF and 100nF, but even one with a value between 1nF and 10nF would be OK.

This is achieved by tweaking or "nudging" the Meter's reading for the reference capacitor using switch S3. Here's how you do it:

1. Remove the 5V supply from the Meter
2. Lift the lid/front panel up from the box and carefully fit the jumper shunt over the pins for LK1, down on the ProtoShield.
3. Close the box and slide S1 to the C position but don't connect your reference capacitor to the test terminals.
4. Re-apply the 5V power and let the Meter go through its initial zero calibration.
5. Wait a couple of minutes, watching the reading for Cx to see if it drifts up appreciably from the initial near-zero figure. If it does, press switch S2 to force a reset and bring the reading back to less than 0.01pF.
6. Connect your known-value capacitor to the test terminals and note the Meter's measurement reading. It should be fairly close to the capacitor's known value, but may be a little higher or lower.
7. If the reading is too low, press the rocker of switch S3 at the upper ("=") end for a second or so; if it's too high, press the lower end ("-") instead. The reading should change by about 0.5%. Continue until the reading is as close as possible.
8. Remove power, open the lid and remove the jumper from LK1.
9. Re-attach the lid.

Note that since the Arduino always saves the revised calibration factor in its EEPROM after every measurement during this nudging procedure, so you only have to do the calibration once.

Also, when you calibrate the meter in this way using a known value capacitor, it's also calibrated for inductance measurements too. SC