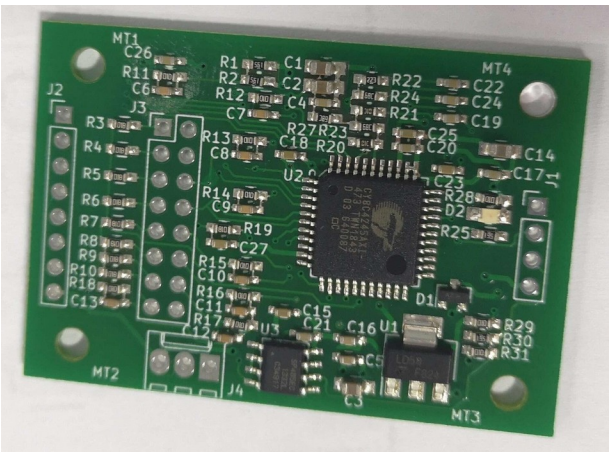
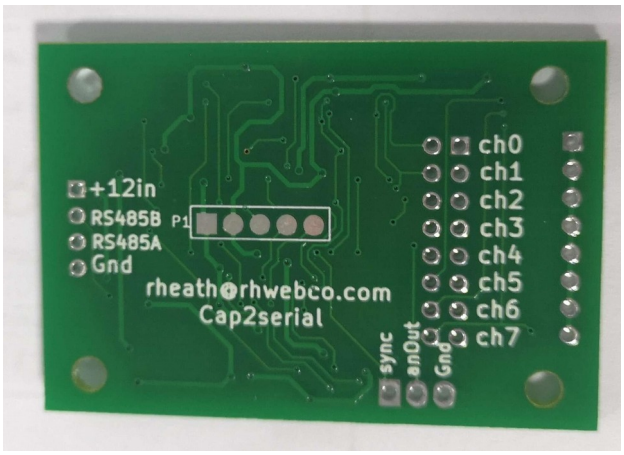
Cap2serial





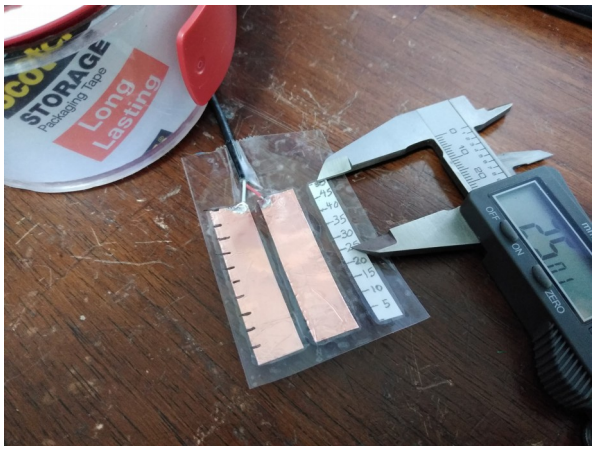
The cap2serial board will convert 8 home built (custom) capacitive sensors into serial data that can be read over RS-485 by using a simple set of commands. The user can make a large variety of different sensors, including angular position, linear position, water level, soil moisture, etc.

Calibration has been made easy. Move the custom sensor to the maximum capacitance, send the “calibrate hi” command. Move the custom sensor to minimum capacitance and send the “calibrate low” command. The cap2serial board will then read from 0.0 to 100.0 percent. Each of the 8 sensors are calibrated individually. The cap2serial board will automatically adjust it’s gain, offset and do the math. Custom sensors with ranges of 0 to 10pf will work fine, but a range of 10 to 100pf will have improved linearity and repetability.

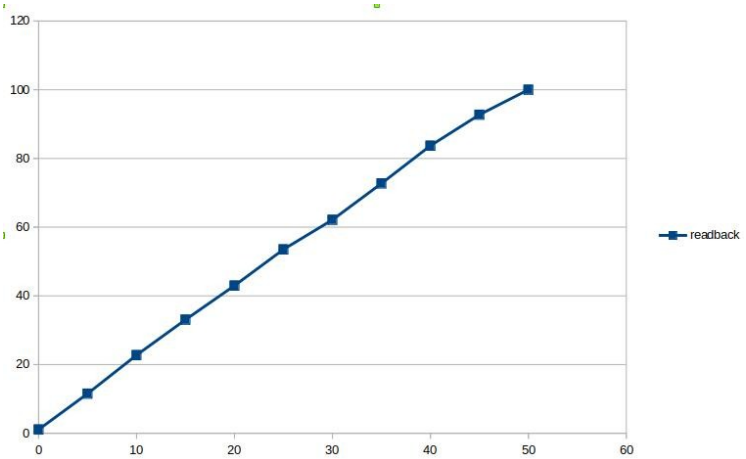
The Cap2serial pcb will remember all of it’s settings and calibrations for years, regardless of power cycling.

Example Sensors:

If the plate area changes as the sensor is moved, the sensor will be nearly linear.



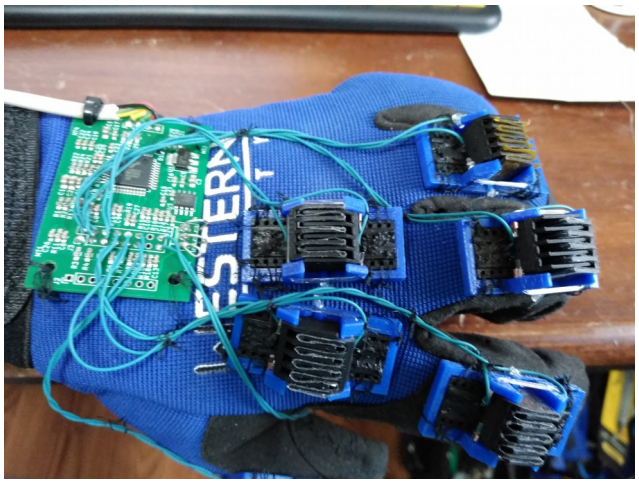
A good soil moisture/water level sensor can be made with two strips of 12.7mm wide (½ inch) copper tape, 50mm long.

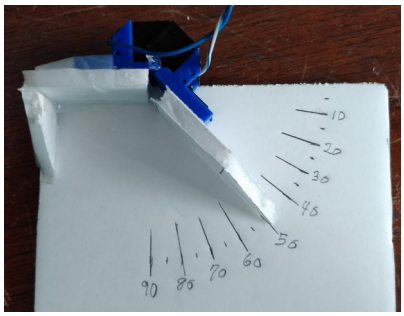


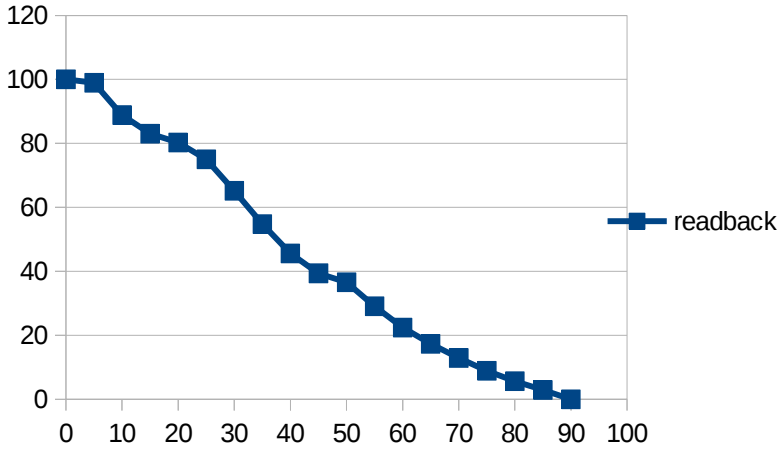
If you seal the top of the soil moisture sensor with silicon rubber sealant, this sensor can be buried and the reading will be proportional to the moisture in the soil. This sensor is inexpensive enough to replace every season because the electronic pcb will not need to be replaced. Calibration should be done each time a new sensor is installed.

As with all capacitive soil moisture sensors, a tiny amount of AC electricity is passed through the soil. It is not known if this effects the plants in a positive or negative way. The cap2serial board provides a command to silence the sensor drive (sleep mode) so that the user can wake it up, take a reading, then put it back to sleep. The readings can be taken periodically, then continuously when the water as been turned on.

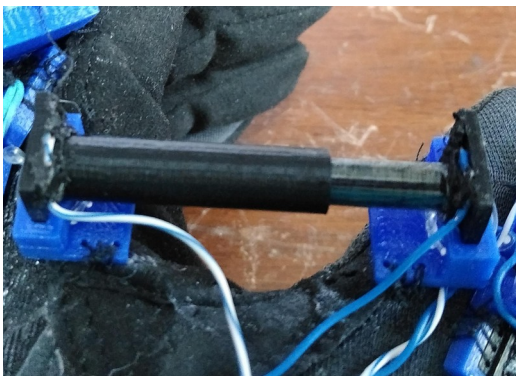
Example of angular position sensor, 3D printed with conductive PLA and a hinged insulator of non-conductive PLA:

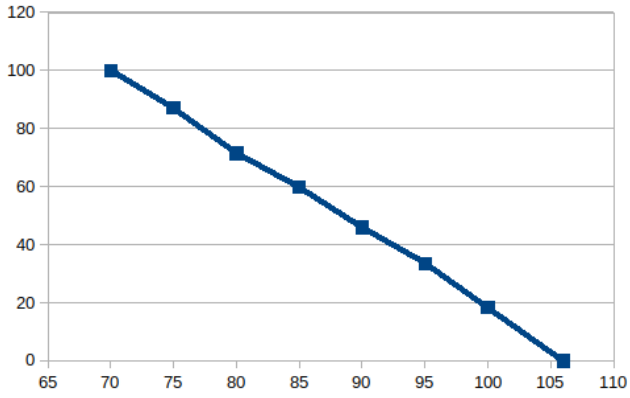




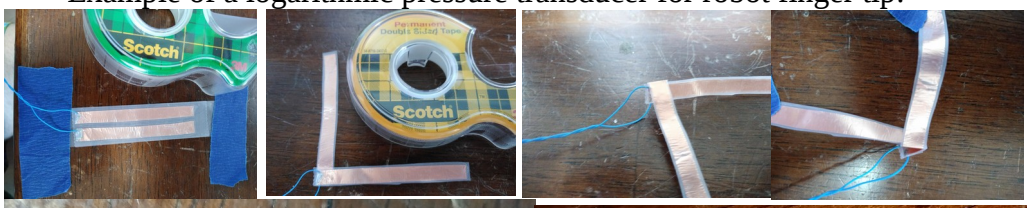


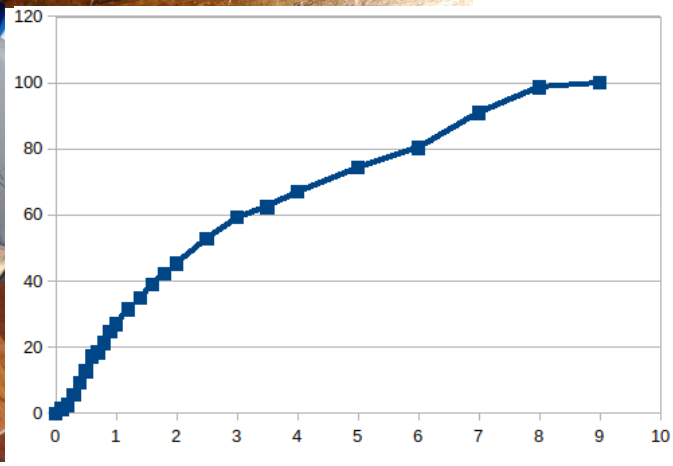
Example of Linear position sensor 3D printed with conductive PLA and an insulator of non-conductive PLA:





Logarithmic sensors can be made if the plate to plate distance varies as the sensor moves.
Example of a logarithmic pressure transducer for robot finger tip:





If the custom built capacitor is too large, it will read back as 100.0% when the sensor has not yet moved the maximum calibrated capacitance. This may happen if the capacitance is above 400 or 500 pico farads. The calibration may also fail if the cap is too large. For example: if each of the copper tape pieces in the above soil moisture sensor is larger than 60mm long and 12.7mm wide, the cap2serial will read 100.0% for anything above that area. If you need to have a large capacitance sensor, such as a pressure plate people sensor), you may add a cap from the sensor input (even pins on J3) to ground (all pins on J2). With a bit of experimentation, the sensor can be brought within the calibration range of the cap2serial board.

The distance from the board to the sensor will determine what type of wire that you can use. Twisted pair is fine for up to 10 inches, but 2 inner conductor microphone cable should be used with the shield connected to ground (available on J2), for greater distances.

Pins:

J1 is power input and communication
J1 pin 1: +12v (can be any voltage from +8v to +15v)
J1 pin 2: RS485B
J1 pin 3: RS485A
J1 pin 4: Ground (12v return)

J2 pins are ground for cable shield connection.
J2 pins 1 through 8: Ground

J3 is where the sensor plates should be connected
J3 pin 1: Sensor drive 0, J3 pin 2: Sensor 0
J3 pin 3: Sensor drive 1, J3 pin 4: Sensor 1
J3 pin 5: Sensor drive 2, J3 pin 6: Sensor 2
J3 pin 7: Sensor drive 3, J3 pin 8: Sensor 3
J3 pin 9: Sensor drive 4, J3 pin 10: Sensor 4
J3 pin 11: Sensor drive 5, J3 pin 12: Sensor 5
J3 pin 13: Sensor drive 6, J3 pin 14: Sensor 6
J3 pin 15: Sensor drive 7, J3 pin 16: Sensor 7

J4 is provided for convenience (not used in normal operation)
J4 pin 1: scope sync
J4 pin 2: anOut, time sequenced analog output of all 8 channels
J4 pin 3: ground

Connect sensor 0, plate 1 to J3 pin 1
Connect sensor 0, plate 2 to J3 pin 2
Connect sensor 0, shield of cable (if used) to J2 pin 1

Connect sensor 1, plate 1 to J3 pin 3
Connect sensor 1, plate 2 to J3 pin 4
Connect sensor 1, shield of cable (if used) to J2 pin 2

Communication:

While any operating system able to use a serial port will work, the examples on youTube and on the web page have been written under Linux. These examples will work great with the Raspberry pi computer that is running Raspbain Linux. The Raspberry pi computer is presently available for less than $50 without the monitor. A USB to RS-485 adapter should be plugged into one of the available USB ports. Many different USB to RS-485 adapters work on the Raspberry pi without having to install any special driver. USB to RS-485 adapters are presently available for less than $6 including shipping.

Setting the device address:

The device address allows multiple cap2serial boards to be connected to the same serial port. The cap2serial board will only answer to commands that are sent to the device address that has been set (and address 00).

Cap2serial is shipped with the device address set to 47 and the baud rate set to 115200 baud. Please be sure that your serial port is set to 8 bits, 1 stop bit and no parity.

Connect a single cap2serial board to the USB-to-RS-485 adapter. There are two pins on J1 that are labeled RS485A and RS485B, connect these to the USB-to-RS-485 adapter The input power is on the same connector (J1), connect +12 (can be anywhere between 8 and 15v DC) to pin 1 and Gnd to pin 4. It is a good idea to connect the ground of the cap2serial board to the ground of the computer.

Use the connected computer to send [00A47] where 47 is the desired device address. The device address can be any number from 01 to FF (hexadecimal). Please include the [], these brackets define the beginning and ending of the command. The cap2serial pcb will answer back: [00A47]. If it does not answer back, you may need to swap RS485A with RS485B. The device address is now set. Note that with some serial programs the first thing that is sent can be lost, so if there is no answer, try sending again before swapping the leads. Test by sending [47R11] to read all of the sensors on the board with device address 47.

If you have multiple cap2serial boards, disconnect the first one and connect the next one. Send [00A3B] (the device address of 3B was chosen arbitrarily). Now you can connect both boards (up to 32 boards). Connect all RS485A (J1pin3) together and also all RS485B (J1pin2) together. To read all of the sensors for the first board, send [47R11]. To read all of the sensors from the second board at address 3B, send [3BR11]. When using multiple cap2serial boards on the same port, please remove R30 (termination resistor) from all boards accept the farthest one (last in chain). The cable length from computer to the farthest cap2serial board can be up to 4000 feet. For long cables and multiple devices, it can be a good idea to reduce the baud rate (see below).

Calibration:

Move the custom built sensor on channel 0 to the largest capacitance. Send [47W30x02]

The device address is 47 in this example. Commands for calibration are 30 through 37 for channels 0 through 7. The 02 is calibrate hi.

Cap2serial will return [CalHi0] where 0 is the channel number. If the channel number is out of range (0 through 7), the return would be [Error].

Watch the LED and if it blinks just 3 times, everything is Ok. If it continues blinking (24 times), the calibration failed (maybe capacitance too large, sensor is shorted, wire is broken, etc). The status of the last calibration may also be read in response to a command. If [47R10] is sent and [R1.0] is returned, calibration was successful. If the calibration has failed, [R0.0] will be returned.

Move the custom sensor on channel 0 to the lowest capacitance. Send [47W30x01]

Note that the calibrate hi must be done before the calibrate low. The same procedure should be done for each of the other channels that are in use, [47W31x02] hi and [47W31x01] low for channel 1, [47W37x02] hi and [47W37x01] low for channel 7.

Watch the LED and if it blinks just 3 times, everything is Ok. If it continues blinking (24 times), the calibration failed. If [47R10] is sent and [R1.0] is returned, calibration was successful ([R0.0] returned if failed).

Reading the sensors:

To read just one sensor, send [47R00] where 47 is the device address and 00 is the channel number. Cap2serial will return [R100.0] where 100.0 is the sensor percent from 0.0 to 100.0.

To read all of the channels at once, send [47R11], cap2serial will return:
[R0.0,53.6,53.5,1.7,43.2,41.7,42.9,52.3]
This is the reading of each sensor in percent, separated by commas.

Setting the baud rate:

The cap2serial board will be shipped with the following settings: 115200 baud, 8 bits, one stop bit, no parity.

If you are using a device that will not support 115200 baud, connect J4 pin 1 (scope sync) to J4 pin 2 (anOut). Turn the power on for a few seconds until the LED stops blinking. Remove the connection from pin 1 to pin 2. All of the device settings will be reset to default values and the baud rate will now be 9600 with 8 bits, one stop bit, no parity. See below to select a different baud rate.

115200 rate was chosen so that a sensor glove made with the cap2serial board can smoothly control a robotic hand. If the baud rate is too low, the motion of the fingers will be jerky.

9600 baud should be used when the distance to the computer is hundreds of feet. An example of sensors that would require such a distance are soil moisture sensors and security sensors.

The command to change the baud rate must be sent using the current baud rate.

If you do not know what the current baud rate is, connect J4 pin 1 (scope sync) to J4 pin 2 (anOut). Turn the power on for a few seconds until the LED stops blinking. Remove the connection from pin 1 to pin 2. All of the device settings will be reset to default values and the baud rate will now be 9600 with 8 bits, one stop bit, no parity. The device address will be 01.

The available standard baud rates are:

baud	setting
115200	25
57600	50
38400	78
19200	155
9600	312
4800	624

Select a value from this table and send [01W13x25] where 25 is the setting from the table (115200 baud).
The setting from the table will be put into an internal divider register with no error checking, so you may use non-standard values if necessary.

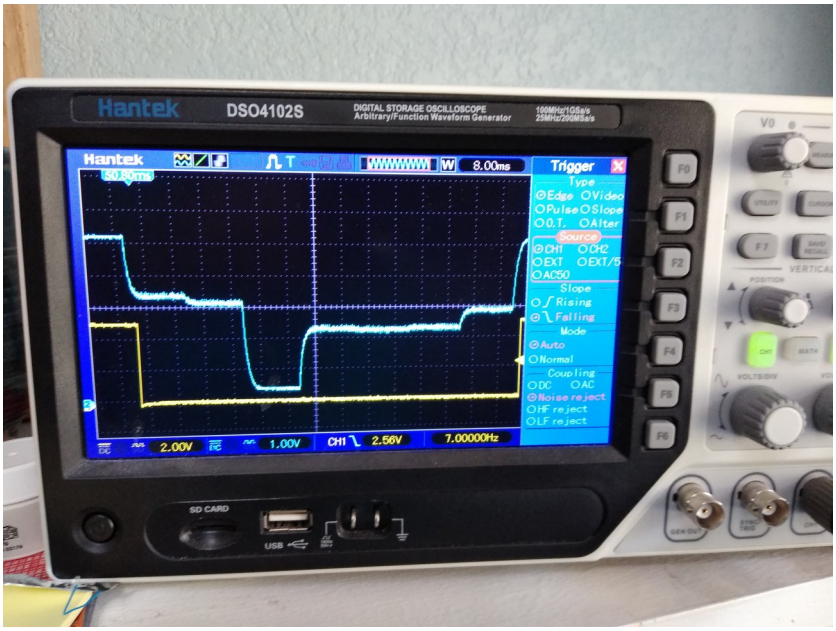
Sleep Mode:

To turn on sleep mode, send [47W12x1]. Sensor drive will be turned off and the cap2serial board sensor values will not be valid during sleep mode.

To turn off sleep mode, send [47W12x0].

Using the scope output (J4):

There is a scope output that has been provided as a convenience. This is not necessary for normal operation, but is useful to see all of the sensor values with a quick glance at the scope. You will still need to do the calibration before this will be useful. J4 has three pins, 1 (scope sync), 2 (anOut), 3 (Gnd). Connect channel 1 of an oscilloscope to pin 1, channel 2 to pin 2, and the ground for both channels to pin 3. Channel 1 should be 2v per division and channel 2 should be 1v per division. Set the timebase to 10ms per division (8ms is perfect, but 10 will do). Set the trigger to channel 1, falling edge, 2.5v and adjust the horizontal position so that you can see the falling edge of channel 1 on the left and the rising edge on the right. On channel 2 there should be a waveform that is divided up into 8 time sections that are about the same width as the positive going scope sync. The time section that happens while the scope sync is high is the analog value for channel 0 where 100% is 5v. All of the 8 channels are displayed so that you can watch the sensor values change in real time without need of sending read commands.



Commands:

All communications are done using ASCII text characters.

Commands are of the form: [XXCYY] or [XXCYYxZZ.Z] where X: device address, C: one letter label for command, Y: shared register address, x: value is present, Z: floating point number (the decimal point does not need to be present for a whole number).

XX: the device address, 00 everyone answer, 01 through FF (hexadecimal) only addressed device will answer.

C: R = read, W = write, A = set device address.

YY: shared register address. Presently 0 through 40 (decimal) are valid but not all are used. All shared registers are floating point. The readback is limited to one place after the decimal point (to decrease communications overhead).

ZZ.Z: floating point value to be written into shared register

Memory Map Of Shared Registers:

0 through 7: sensor outputs

8, 9: used for development
10: will return 1.0 if last calibrate command was successful, 0.0 if not.
11: if this address is read, all 8 sensor values will be returned (comma delimited)
12: write 1 for sleep mode, write 0 to wake up
13: will be transferred to UART divider register – see baud rate table above.
14: used for development
30 through 37: write 1 to begin calibrate Low, write 2 to begin calibrate Hi. 30 is channel 0, 37 is channel 7

Note: Most of these shared registers are not backed up and will be re-initialized on power-up. The exceptions are: device address, UART divider register (baud rate)

Return format:

A read command will return [RXXX.X] where X is a floating point number with no leading zeros and one place after the decimal point. The exception is if address 11 is read, [47R11], all 8 channel values will be returned, separated by commas: [R6.3,83.0,98.4,100.0,100.0,100.0,100.0,100.0]

The set address command, [00A47] ,will return [00A47] where 47 is the new device address.
A write command will return [W]

When the calibration registers are written to, [47W37x02] or [47W37x01], [CalHi7], [CalLow7] or [Error], where 7 is the channel number will be returned.