Cyduino — making our own Arduino

The Arduino is a very nice development platform for simple embedded systems. It is easy to connect some hardware — sensors, amplifiers, switches, displays, actuators, etc. — and write some software to make a working prototype in short order.

However, when it comes time deploy the system, we probably don't want to use our Arduino board — we won't be able use it for anything else! And, as nice as the Arduino is for prototyping, it is rather unwieldy for use in the field — the board is bit big and there is a lot of hardware overhead that probably isn't needed for a specific application. Plus, the boards are kind of expensive. If we needed to make 10 copies of the hardware, we would probably blow our budget buying Arduino boards.

We need to make our own Arduino that can integrated with all the other parts of a system. Fortunately, making our own is very easy — and inexpensive — to do. We will describe how to do it on the following pages. Being loyal Iowa Staters, we will call our simplified version the "Cyduino".

A simple thermometer as an example application

There is really no point in making our own Arduino if we don't an application in mind. We will use the digital thermometer that was described in an earlier club meeting.

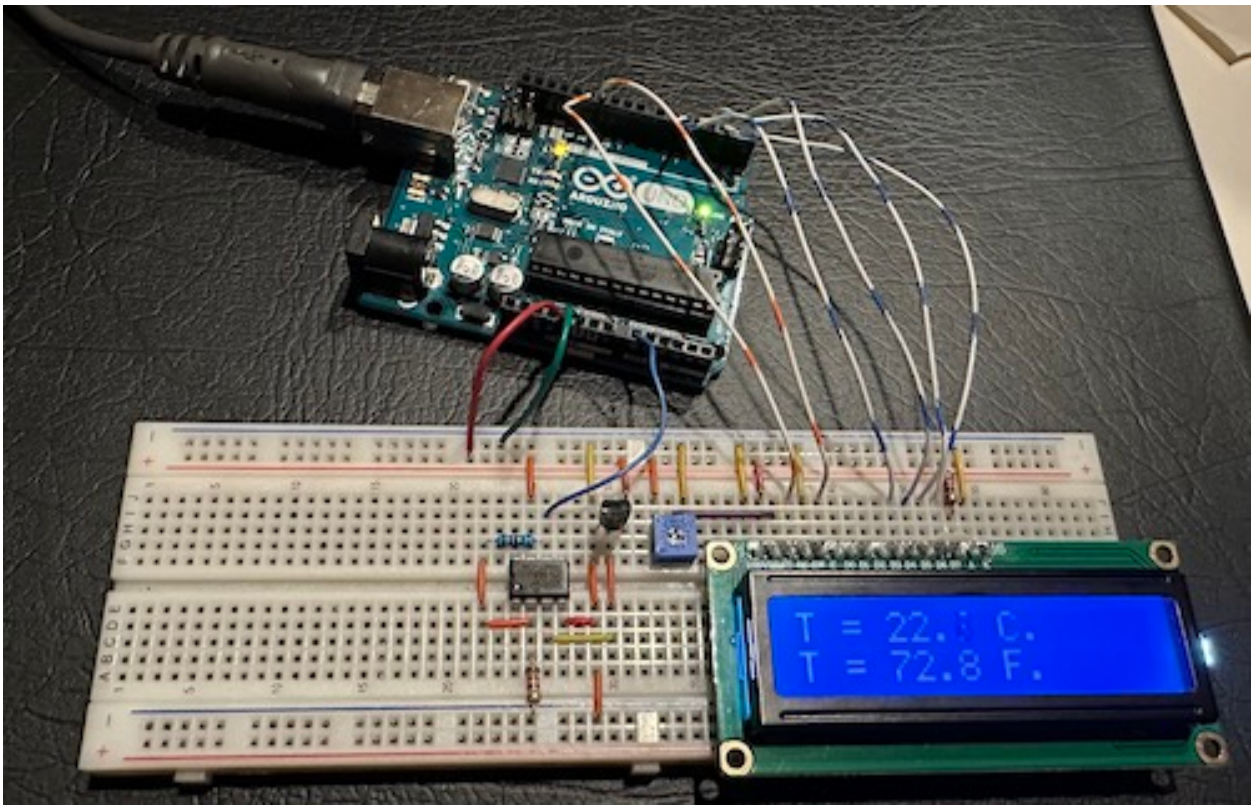


Figure 1. The digital thermometer using an Arduino controller. An LCD display shows the temperature in Celsius and Fahrenheit.

Essential hardware

We need to examine the Arduino hardware to determine what parts are essential and what parts could be left off.

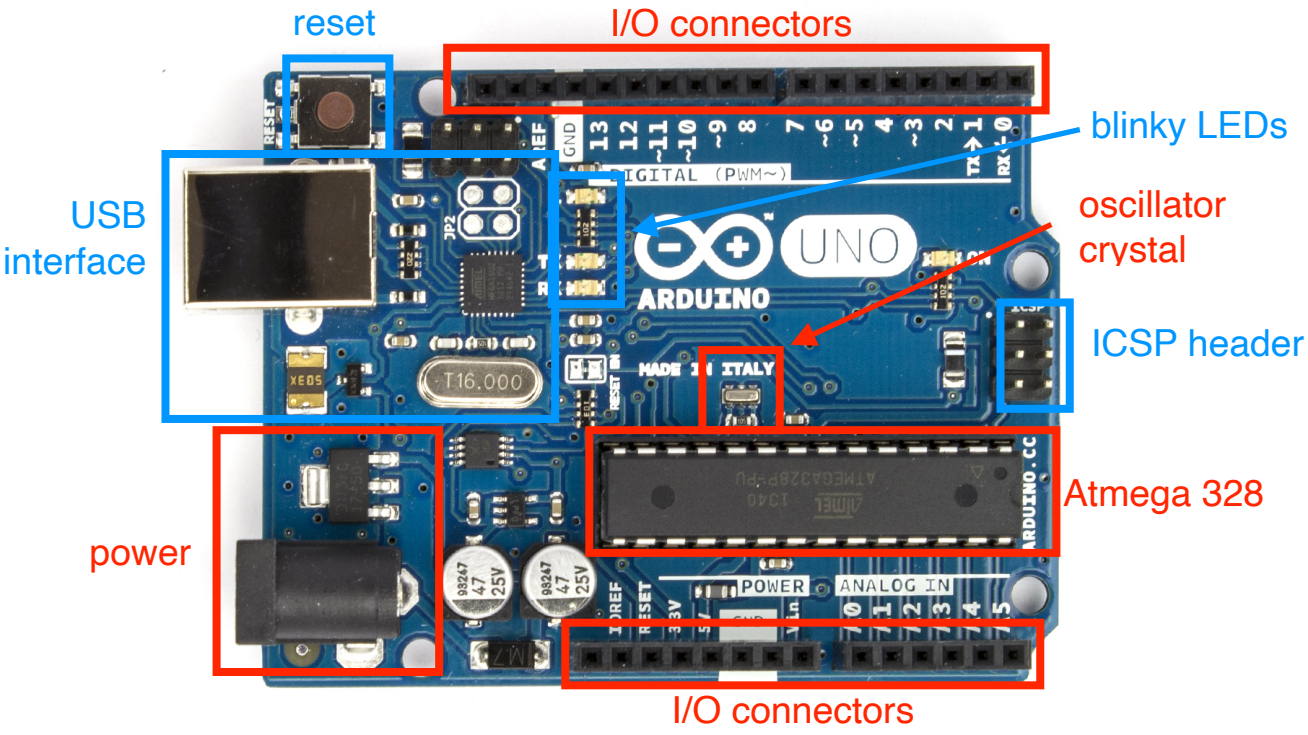


Figure 2. Basic parts of the Arduino board. Most of the pieces outlined in red are essential. Sections outlined in blue might be optional.

Essential sections include

- The Atmega microcontroller chip. (Duh.)
- The oscillator crystal and associated capacitors.
- Power input and regulation.
- Input/output connectors.

That's it. The optional parts are:

- The USB interface, which is an important part of the Arduino. It is the programming interface and also provides power. But the interface is rather complicated and duplicated would be tricky. Leaving it out means that we will have to find an alternative to send programs to the Atmega chip.
- Reset button. We can leave out the switch, but we will include a pull-up resistor to make certain that the microcontroller does not reset accidentally.
- In-circuit serial programming (ICSP) header. This is an alternative interface. We will omit it in the breadboard implementation of Cyduino. It will be included in the PCB version.
- Various blinking LEDs.
- There is also a 3.3-V regulator on board — we will it out for now.

Pin map

Because we will be connecting directly to the pins on the Atmega chip and and not to connectors at the edge of the Arduino board, we need to be able to translate between the Aduino pin nomenclature and the actuals pins on the chip.

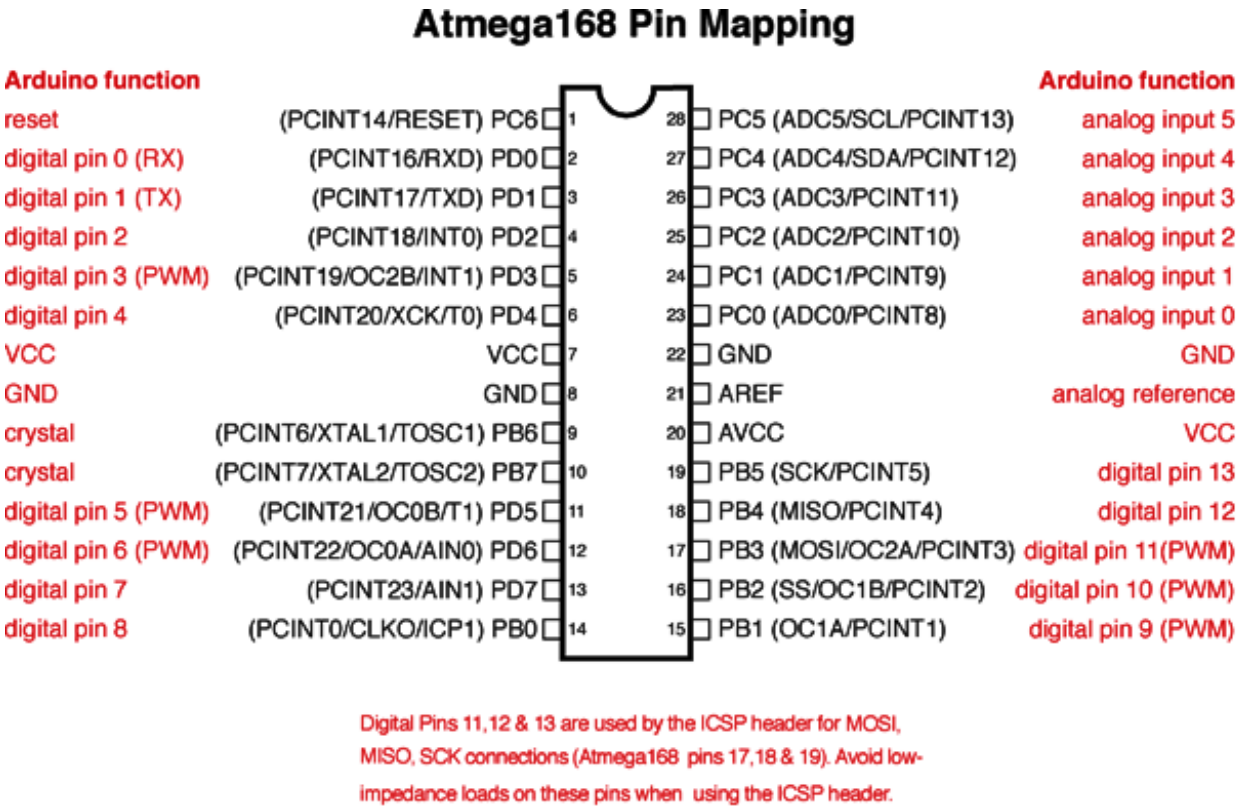


Figure 3. Atmega chip pinout with Arduino names in red. (Atmega168 and Atmega328 have the same pin arrangement.) The names used by Atmel are next to the pins. The text inside the parentheses describes the pin functions using Atmel terminology, which is a bit inscrutable to novice users.

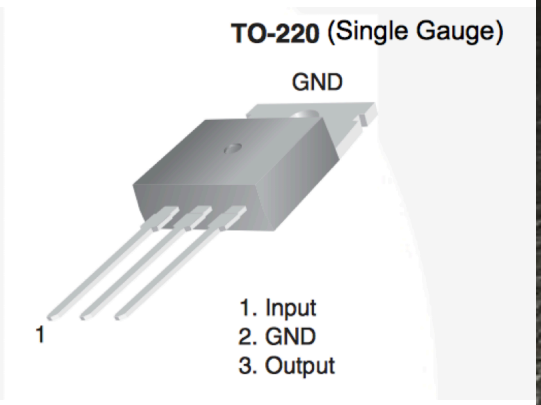
Breadboard version

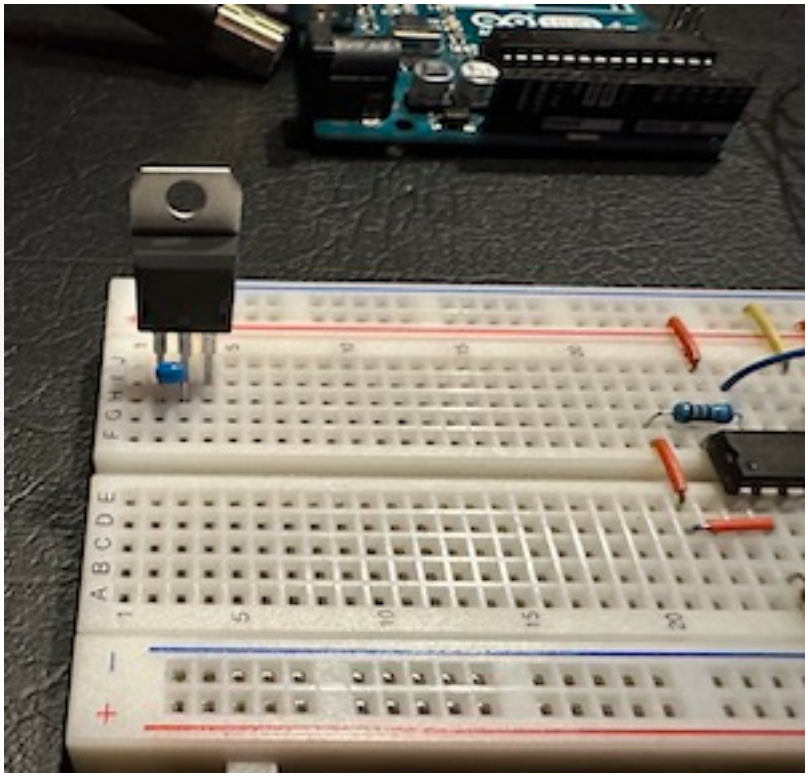
In Fig. 1, we see that there is considerable space to the left of the amplifier and sensor — more than enough to add a breadboard version of Cydunio. We will go step by step.

1. Voltage regulator

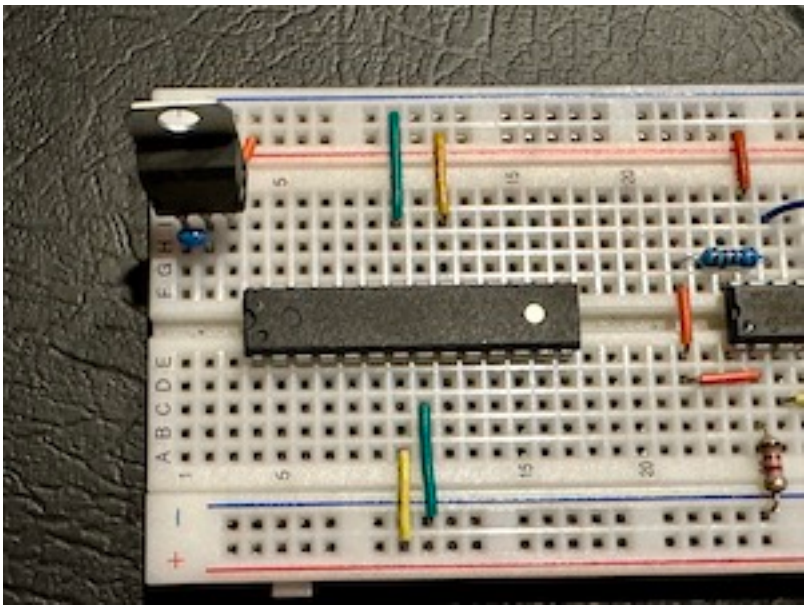
We will use a classic 7805 voltage regulator to provide the steady 5 volts that powers everything. The 7805 is a simple three-pin regulator — input, ground, and output — in TO-220 package. The pin arrangement from the data sheet is shown below left. The input voltage must be greater than about 7 V and it can probably be as big as 15 V without the regulator becoming too hot. We will assume that input voltage will be 9 V, coming from a battery or a DC wall-plug unit. The data sheet recommends adding a 0.33-μF capacitor between the input and ground to improve stability.

Below right is a photo of the regulator in the breadboard. The input voltage will be connected to pin 1 on the left. The center pin is ground and is connected to the breadboard ground rail. The right pin is the 5-V output and is connected to the 5-V breadboard rail. (The jumper wires are behind the regulator package and can't be seen in this view.) The small capacitor can be seen in front of the regulator.





2. Atmega 328



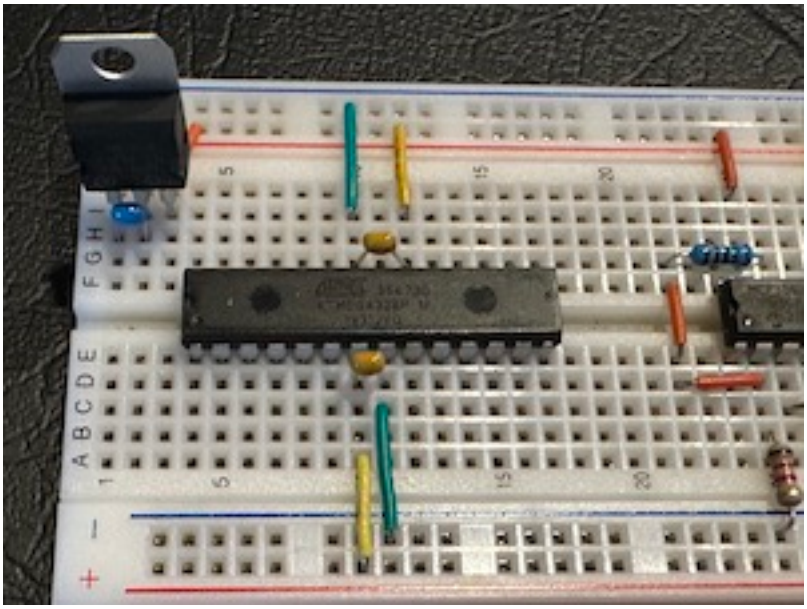
Unplug the power from Arduino. Gently pry the Atmega chip out of the socket. It is probably necessary to use a small screw driver or similar device for prying. To avoid bending the pins, pry the chip out part way on one end and then go to the other end to finish prying it out.

Insert the chip into the breadboard. (Note the orientation — pin 1 is lower left here. Pin 15 is upper right.) Then attach the power and ground connections — there are power and ground pins on each side.

Ground — pins 8 and 22 (green wires in the photo). 5 V — pins 7 and 20 (yellow wires).

Note that the power rails on the two sides of the bread board have been connected together with jumpers (out of view of this photo). And the same for the ground rails.

3. By-pass capacitors

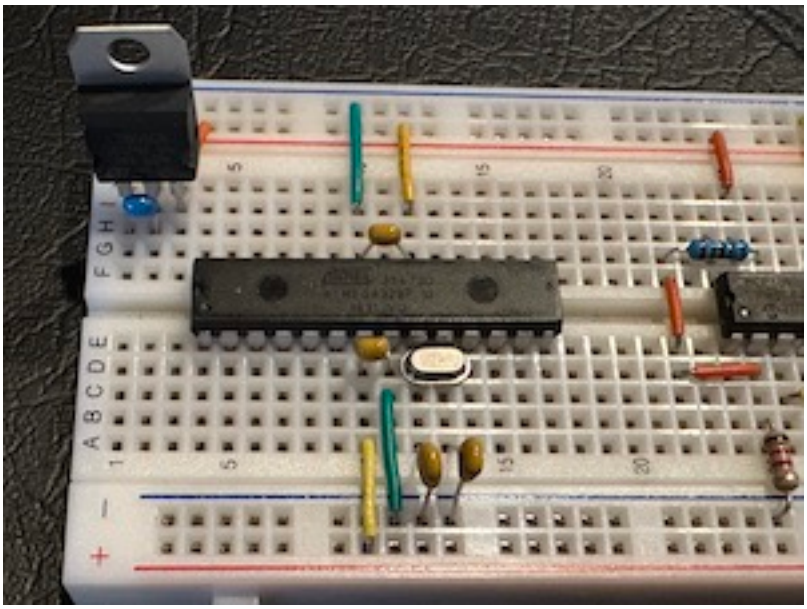


Connect 0.1-μF capacitors between the power and ground connections on each side of the chip. These bypass capacitors help smooth voltage fluctuations due to switching. (Many weird problems encountered when using digital chips can be fixed by adding by-pass capacitance.)

4. Oscillator crystal and capacitors

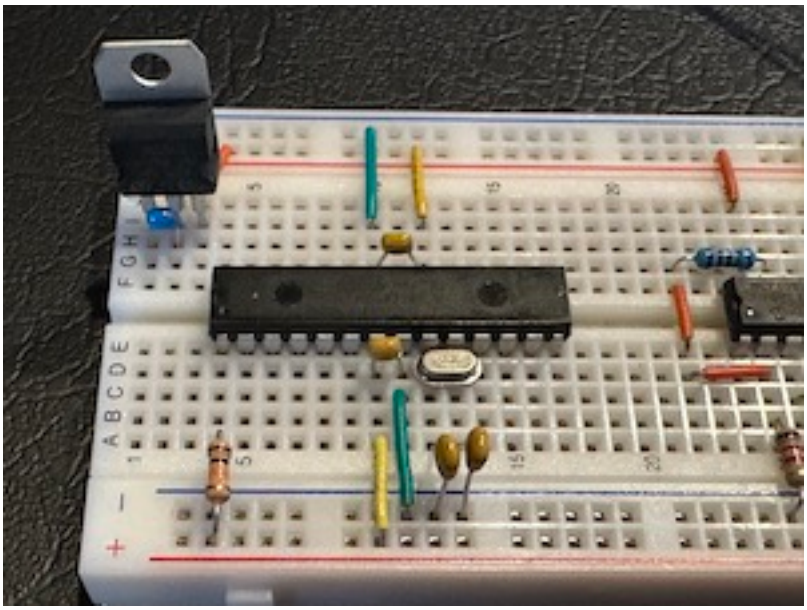
Next add the 16-MHz oscillator crystal between pins 9 and 10. Then add two 22-pF capacitors between those two pins and ground.

Note that this is slightly unusual crystal package. The spacing between the pins is 0.1-inch, allowing for a nice fit next to the chip on the board. (The part number is on the BoM, listed later.) The more common package has pins spaced 0.2 inches apart, which makes for an awkward fit on a breadboard.



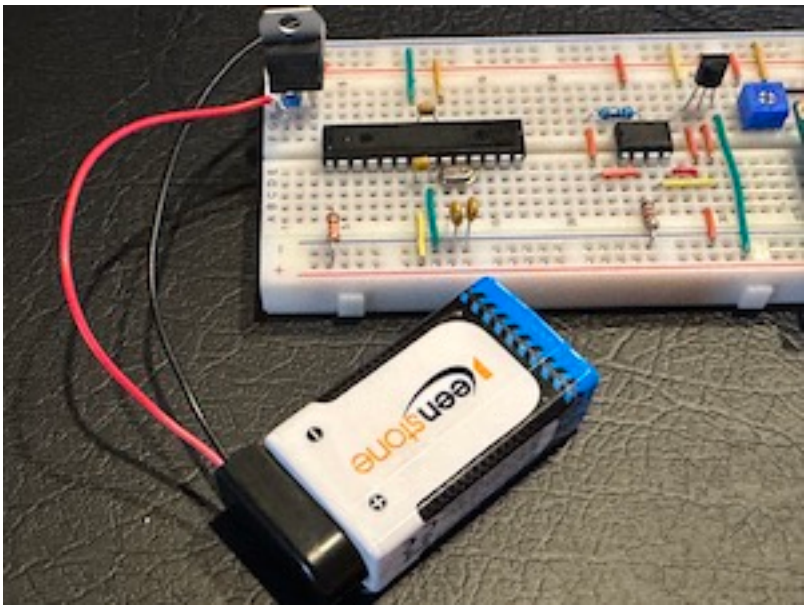
5. Pull-up resistor on reset pin.

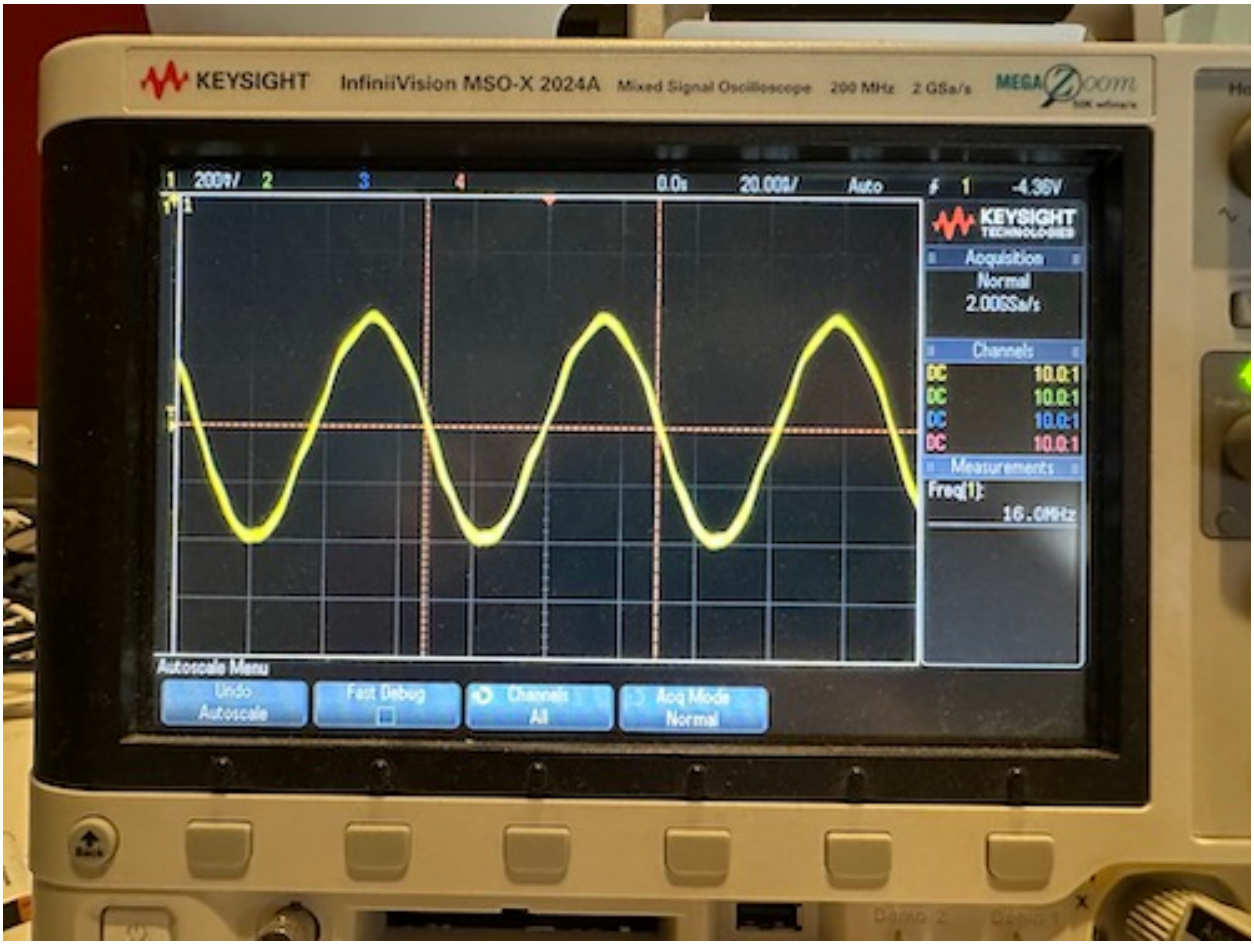
Lastly, connect a 10-kΩ resistor between pin 1 and the 5-V rail. (Not ground!) The microcontroller resets when pin 1 is pulled low to ground. By connecting to 5-V with a resistor, we can be be sure that the controller does not inadvertently reset. (If we were to leave the pin floating, we don't know what might happen.)



6. Connect power and check.

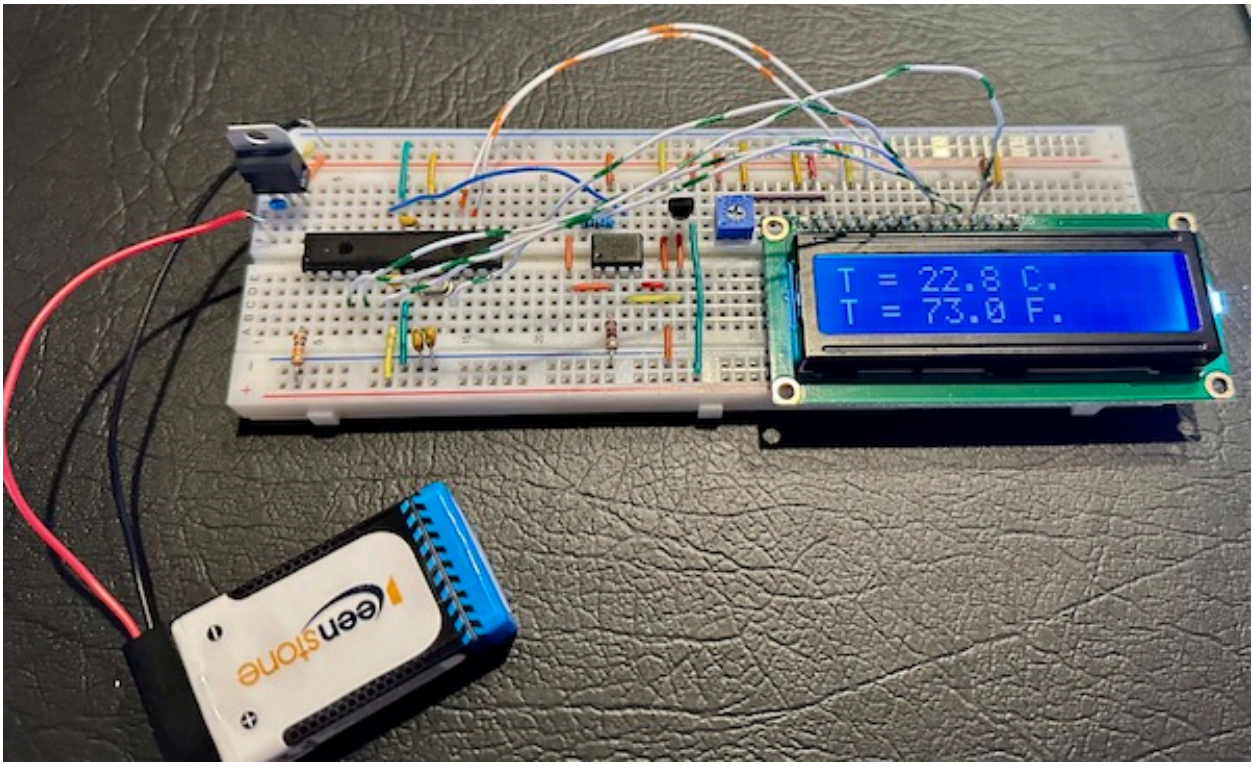
At the point, we can connect the power source — in this case a 9-V battery. It is probably a good idea to check various voltages with a multi-meter. Look for 5 V at the power connections, pins 7 and 20, and the reset, pin 1. Check for ground on pins 8 and 22. If we have an oscilloscope available, we can check the operation of the oscillator — looking for a 16-MHz sine wave between pins 9 or 10 and ground.





7. Connect the rest of the circuitry.

Disconnect the power. Use the jumpers to connect the output of the amplifier to analog input 0. Then connect jumpers from the LCD display to the six digital pins. Then re-connect the power. If all connections are good, then the thermometer should start displaying the temperature.



Success!

Cyduino PCB

As an intermediate step between the general-purpose Arduino board and hardware specifically designed for an application, we have designed a Cyduino PCB. The Cyduino has all of the components included in the breadboard version described above. It also includes an ICSP header[1] for programming, a reset switch, and an LED to indicate that the power is on. All of the extras could be left off the board, if desired. The Cydunio PCB allows for some flexibility in terms of making connections. Depending on what external components are needed, we can use in screw terminals[2], header pins, or female headers (like the Arduino).

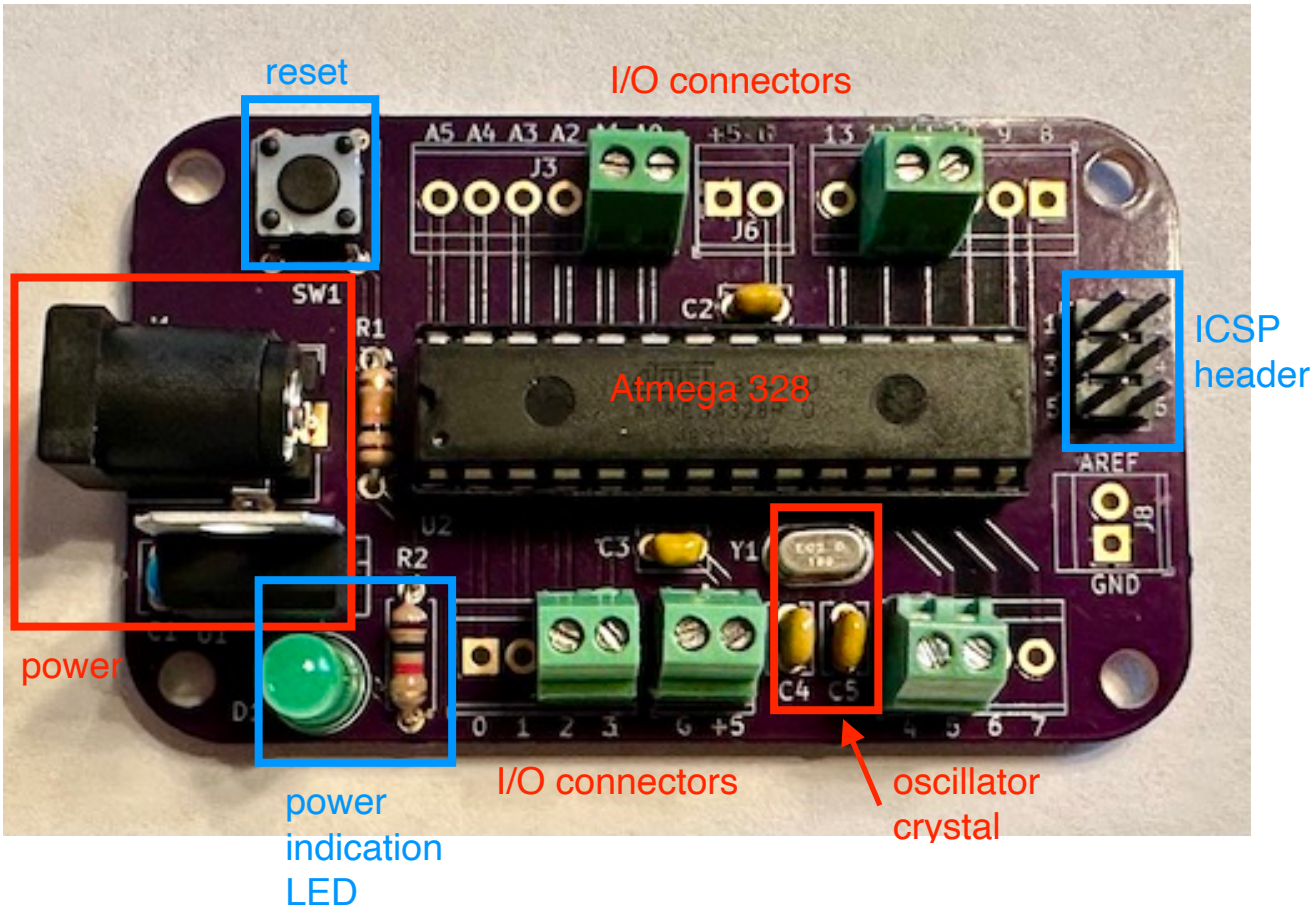


Figure 4. Photograph of the Cyduino PCB with all the components in place.

[1] Using an external programmer will described in a separate document. If available, an external programming makes downloading code to the Cyduino *much* more convenient.

[2] Screw terminals are my preferred connection method. Although a screw driver is needed, the connections are much more solid. — GT

We see the essential elements:

- Atmega328 chip in a socket
- oscillator crystal with the associated capacitors
- power input jack with 5-V regulator
- Through-holes for all of the I/O connections, allowing for a variety of connection methods. (Screw terminals are shown above.)

And there are a few optional items. These could be omitted without loss of basic functionality.

- An ICSP header for programming on the PCB directly.
- A reset switch.
- A power indication LED and current-limiting resistor.
- A connector for applying an external analog voltage reference.

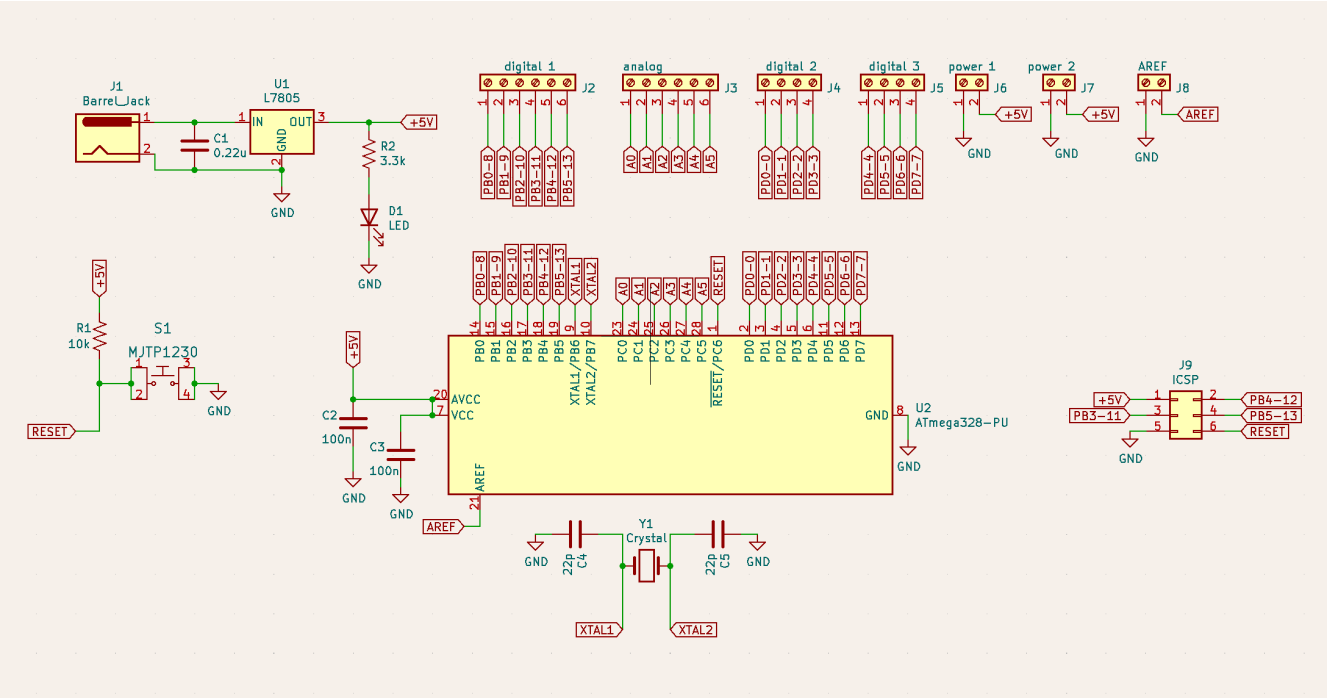


Figure 5. Kicad schematic for the Cyduino PCB.

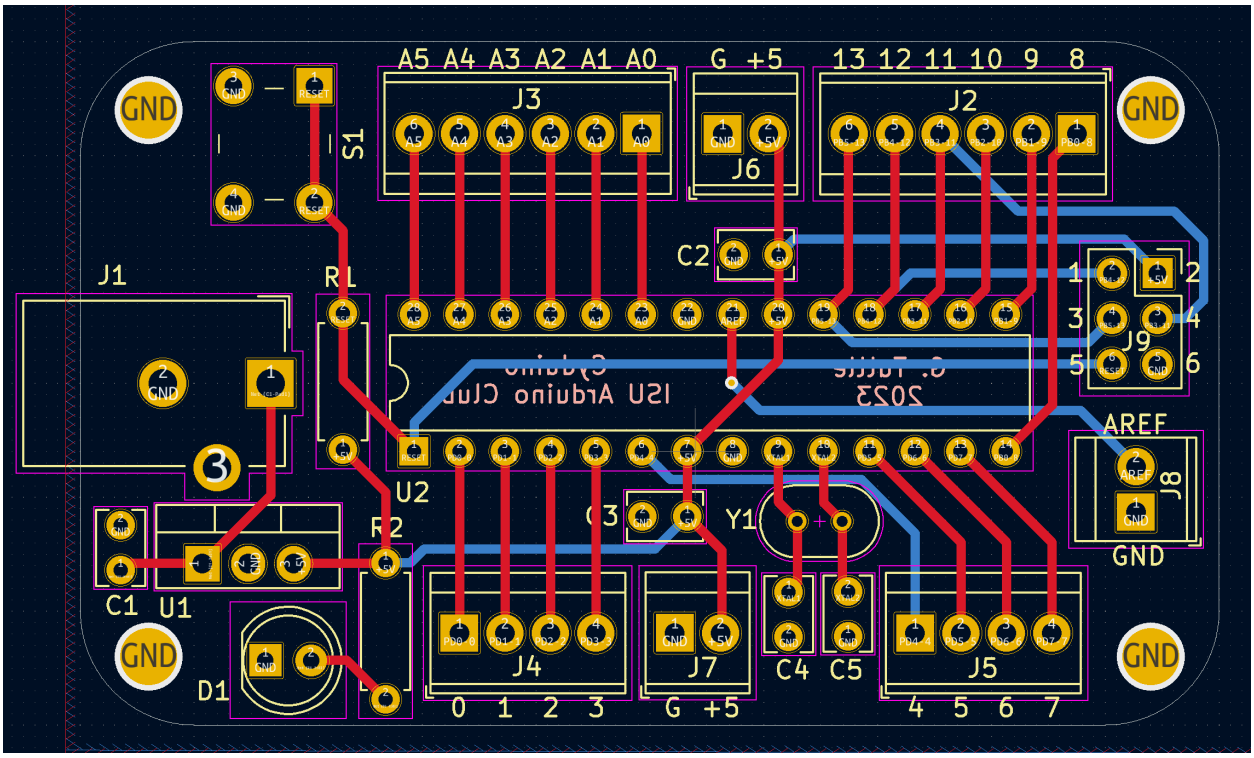


Figure 5. The Cyduino PCB layout from Kicad. The dimensions are 2.5 inches x 1.5 inches.

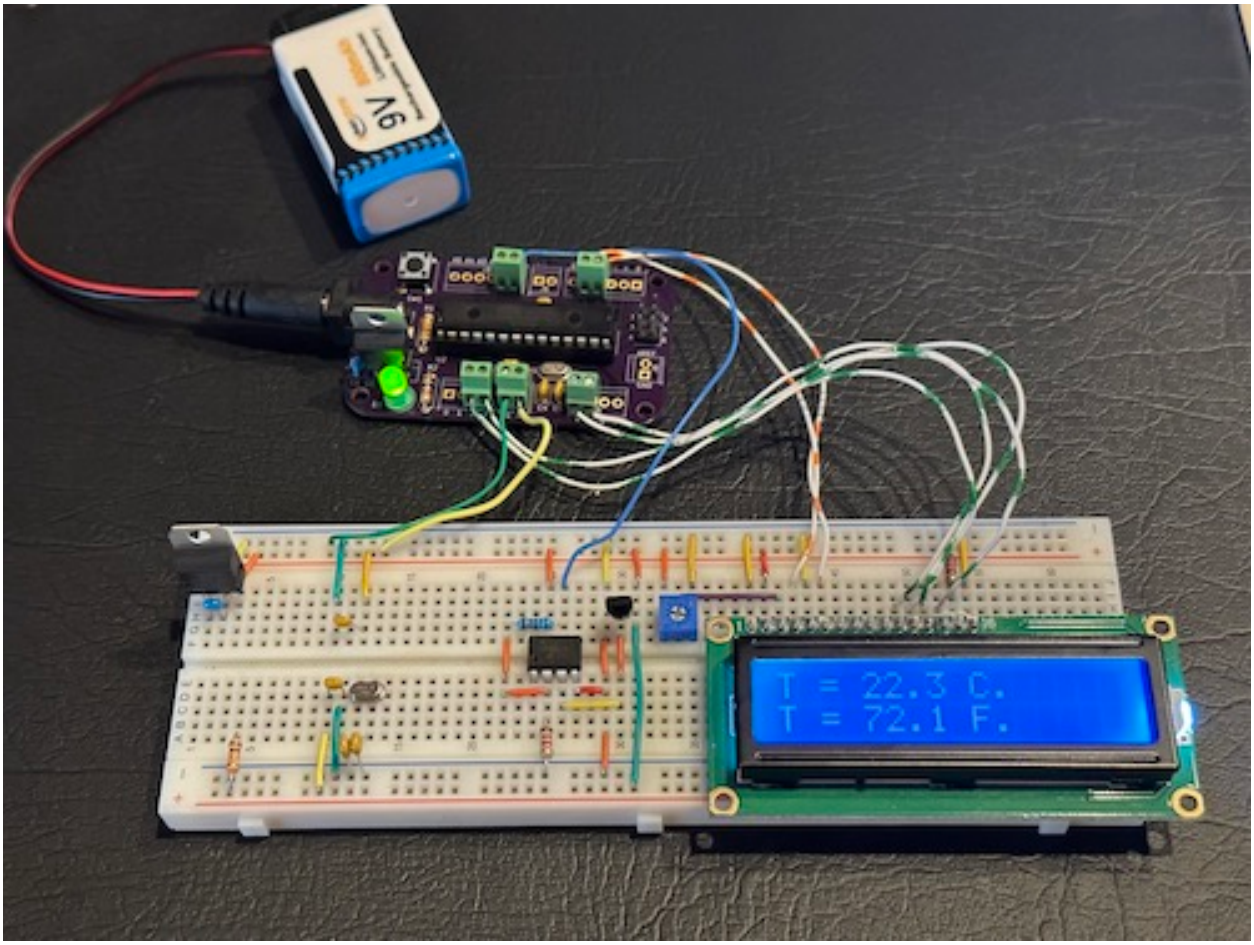
The barrel jack power connector is a standard 2.1 mm style. It works nicely with a 9-V battery strap that goes to a matching connector. These can be found on Amazon. Or it is easy to make one. A 9-V wall-plug supply probably has this type of connecter. If some other type of supply will be used, it might be best to leave the barrel connector off of the PCB and solder the leads of the other supply directly to the board.

Bill of Materials

part	part number	DigiKey number	quantity	price	total
Atmega 328	ATMEGA328P-PU	ATMEGA328P-PU-ND	1	3.32	3.32
socket	On Shore ED281DT	ED3050-5-ND	1	0.37	0.37
5-V regulator	STMicro L7805CV	497-1443-5-ND	1	0.69	0.69
100-nF cap	Vishay K104K15X7RF5TL2	BC1084CT-ND	2	0.23	0.46
220-nF cap	Vishay K224K20X7RF53L2	K224K20X7RF53L2-ND	1	0.43	0.43
16-MHz crystal	ECS-160-20-46X	XC1759-ND	1	0.39	0.39
22-pF capacitor	K220J15C0GF5TL2	BC1005CT-ND	2	0.27	0.54
red LED	Cree C5SMF-RJF-CT0W0BB1	C5SMF-RJF-CT0W0BB1-ND	1	0.17	0.17
3.3-kΩ resistor	Stackpole CF14JT3K30	CF14JT3K30CT-ND	1	0.10	0.10
10-kΩ resistor	Stackpole CF14JT10K0	CF14JT10K0CT-ND	1	0.10	0.10
reset switch	APEM MJTP1230	679-2428-ND	1	0.15	0.15
barrel jack	CUI PJ-102AH	CP-102AH-ND	1	0.82	0.82
ICSP header	Amphenol 67996-406HLF	609-3218-ND	1	0.29	0.29
screw term - 2	On Shore OSTVN02A150	ED10561-ND	3	0.92	2.76
board			1	2.50	2.50
					13.09

Table 1. Bill of Materials for the Cyduino kit. Prices as of March 2023. The kit includes 3 two-position screw-terminal connectors. Depending on the application, more connectors may be needed. The basic kit comes in at about the half the cost of a genuine Arduino. Leaving out the screw terminals, header, reset switch, and LED removes about three and half bucks from the total price.

A Cyduino board was soldered up from scratch. (See the Cyduino build document for the details.) There are enough screw terminals for: ground and 5 V, the analog input from the sensor/amplifier, and all of the digital connections for the LCD display. The programmed Atmega chip was moved from the breadboard to the Cyduino socket. Power was connected, and … it worked! (Should we be surprised?)



So when we need to prototype an Arduino-type application, we can use the Cyduino board instead. It's cheaper and maybe a bit more flexible. When we are ready to design a PCB to be used specifically for our application, the Cyduino Kicad files gives us a starting point.