Fido

A farmer-built electronic tool that can monitor greenhouse temperature, record greenhouse data, and alert the farmer to problems in the greenhouse via cell phone text message.

Bill of Materials:

Arduino UNO	$30
Datalogging Shield	$20
10K Precision Epoxy Thermistor	$4
SD Card 1GB or more	$5
Enclosure	$15
9V Power Supply	$6
Phone Related	
Motorola C168i + SIM Card	$30
3/32 Minijack	$4
Logic Shifter	$1
Back-up Battery	
Backup 9V Adapter	$6
2 Diodes	$1.50
2 DC barrel jacks	$2
Protoboard	$2
	$127

How to Build

Preparation

1.Download Arduino IDE v1.0. An IDE is an "Integrated Development Environment," or in plain english, it has everything you need to program your Arduino. The full tutorial is available on the Arduino site but I will fill in a few concepts as we go.
In this Arduino environment, you will be writing code in C++. There is an upload button which automatically "compiles" your code (turns it into processor code) and then uploads it to your Arduino. If you are working on your code for a while, it is helpful to compile it periodically to catch syntax errors as you make them. If all this is overwhelming, don't worry. You don't need to write any code for this project and all you have to do is learn how to upload code!
Your IDE also keeps all your "libraries" in order. Libraries are pieces of already written code that you can call on when writing your Arduino program. They are usually device specific, so for example, with Fido we use a library that was specially written for a cell phone and another for the data logging shield.
Download the libraries and unzip them. There should be three individual folders, each containing at least two files, one with a .h suffix and one with a .cpp suffix. Open your Arduino sketchbook folder. If there is already a folder there called libraries, place the library folders in there. If not, create a folder called libraries in the sketchbook folder, and drop the library folders in there. Then re-start the Arduino programming environment, and you should see your new library in the Sketch > Import Library menu. Additionally, under File > Examples, you should now see our libraries along with their example sketches.
Finally, before your IDE will be able to see your Arduino, make sure your computer recognizes it! There are device drivers included in the Arduino folder and if you have trouble, refer to the Arduino web site.
If you just read all this and need a more detailed tutorial, check out this one from Lady Ada. It will be less specific but if you are serious about learning how to use your Arduino to its full potential, their learning system will bring you up to speed VERY quickly and efficiently.
2.Activate SIM card and load with text messages
We are using the Motorola C168i because of the 3/32 stereo jack connection - it's really easy to make the Arduino interface with it but sadly more recent cell phones don't use that kind of connection anymore.
You will need a SIM card for your C168i if you don't yet have one. Unless you specifically sought out an unlocked one, your phone will most likely be locked to the AT&T network and you will need one of their SIM cards which can be purchased on their site. You may also be able to find one in their stores (to be confirmed) and you can definitely find them at Radioshack (just buy their cheapest Go Phone and it will be included). When activating it, I recommend the 10c/min plan, and that you simply buy text message packages that are valid for 30 days ($5 for 200, $10 for 1000).
Also, since we will be leaving our cell phone charging most of the time, you'll benefit from going into the settings and silencing the phone so as not to get beeped at every few minutes.

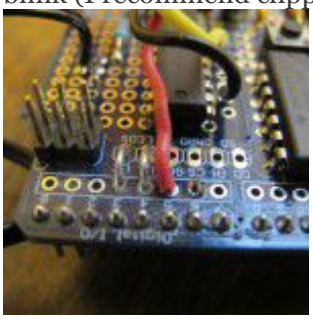
Build

1.Assemble Adafruit data logging shield following their easy tutorial
Notes:
1. If you don't have a vise, laying the board down on some bricks or anything other similar setup should do

the trick

2. You do not need to solder the RTC clock mount if you don't want to

3. You should wire a jumper between pins 3 and 4 and the LED holes right next to them if you want them to blink (I recommend clippings from the legs of a diode to do so)



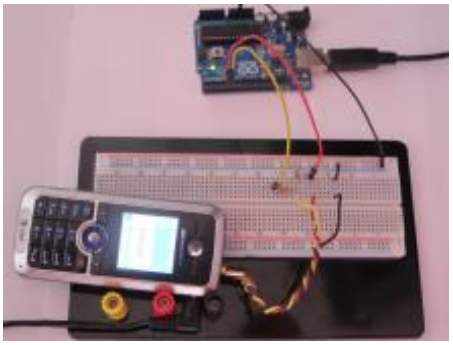
2. Make the wire to connect your cell phone to your Arduino (see Appendix A)

3. Connect the cell phone and thermistor to shield (See Appendix B)

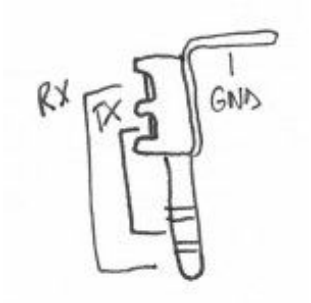
4. Build your back-up battery (See Appendix C)

5. Upload the Fido sketch from File > Examples > Fido to the shield and program via text message (See Appendix D)

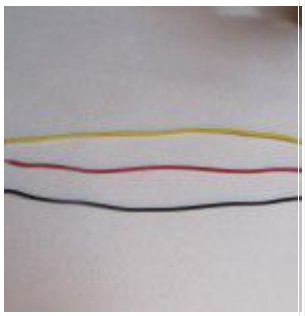
Appendix A: Cellphone to Arduino



You can find a Motorola C168i on Ebay for less than 10 dollars. The simplest way to get service for it is to buy a pay-as-you-go SIM card, at Radio Shack for example. While you're there you can also pick up the3/32 plug; unfortunately, they don't have any information on the packaging but I've included a figure here for your reference (see left).



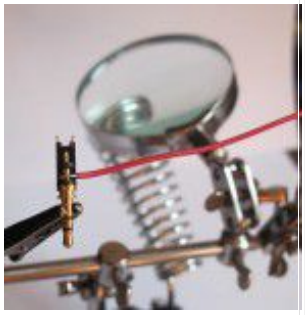
Here's a step-by-step on how to make your "breakout wire" and plug it into your Arduino.
Tools: soldering iron, wire clippers, wire strippers, a soldering stand is helpful
Bill of Materials:

	3/32 Mini-jack
	Three lengths of wire

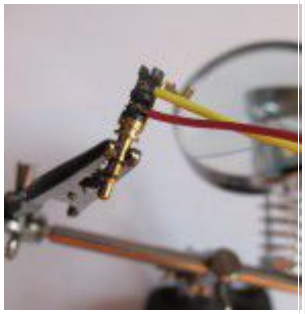
Two resistors to make a voltage divider. I used resistor values of R1=3.3kOhms and R2=5.6kOhms but plenty of other ratios will work (just be careful that you don't reduce current too much).

In general, for any R1, ideal R2 = (3.3/5*R1) / (1-3.3/5)

Steps



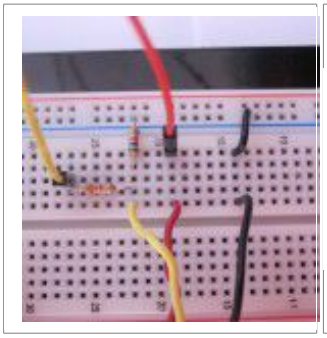
Solder one wire that will be the TX for the phone or the RX on the Arduino end to the lower-most part of the connector.Be careful! The connector is made of plastic so you'll want to avoid the need to resolder.



And another right above it (RX for phone, TX for Arduino)

Finally add the ground wire to large metal tab

Twist the wires and make them look pretty (I like to first twist two and wrap additional wires one-by-one)



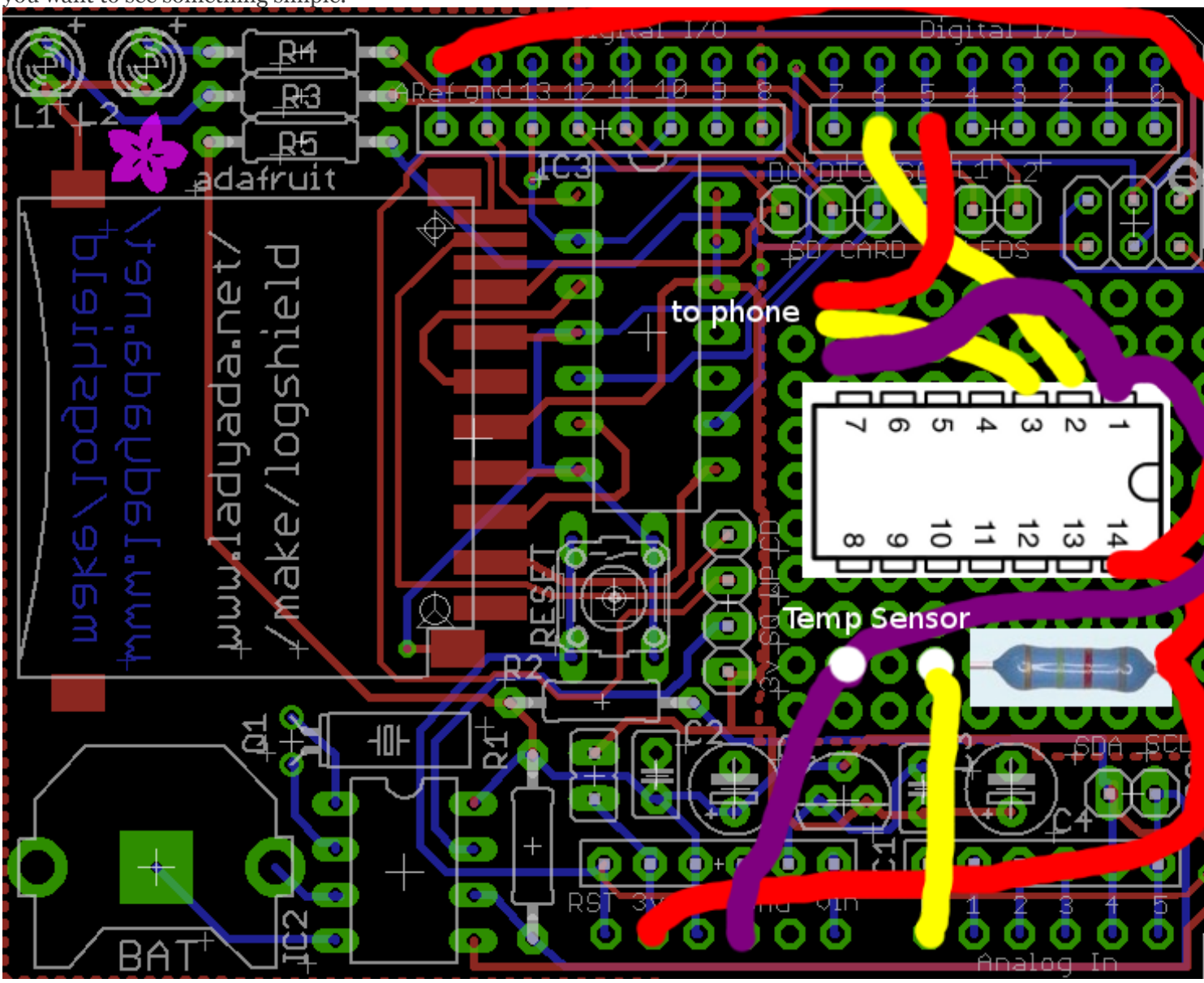
Plug things into your breadboard! Choose a resistor ratio such that you divide 3.3V out of the Arduino's 5V.

Note the resistors I used are 3.3kOhms (R1/Left/Horizontal) and 5.6kOhms (R2/Right/Vertical).

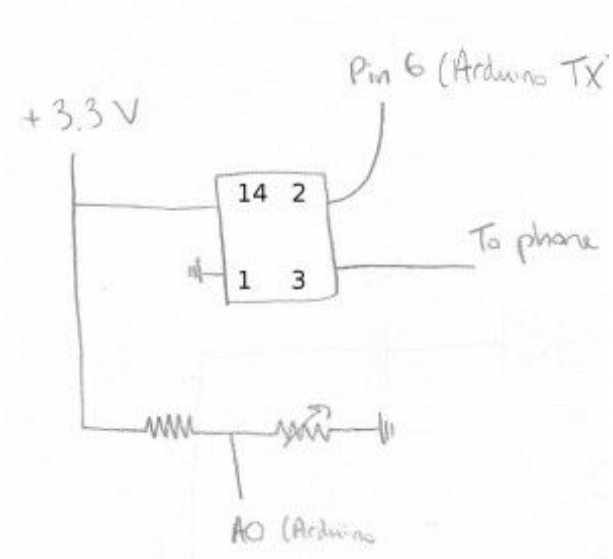
Now try out the GSMSerial example library to make sure everything works fine (in Arduino IDE, go to File>Examples>GSMSerial>GSMSerial_Example

Appendix B: Building on the Shield

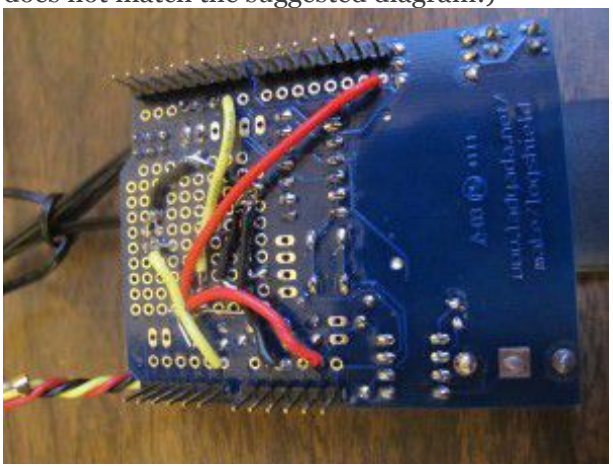
This where it all comes together! There are a lot of pieces but they're all pretty straightforward individually. There are tons of different ways to place your logic shifter and temperature circuit on the board, but this is a suggested one (please excuse the lame paint job). I've also included a picture of the circuit diagram in case you want to see something simple.



Once you have everything soldered on, run the thermistor example sketch to make sure the temperature sensor is running okay and then upload Fido!

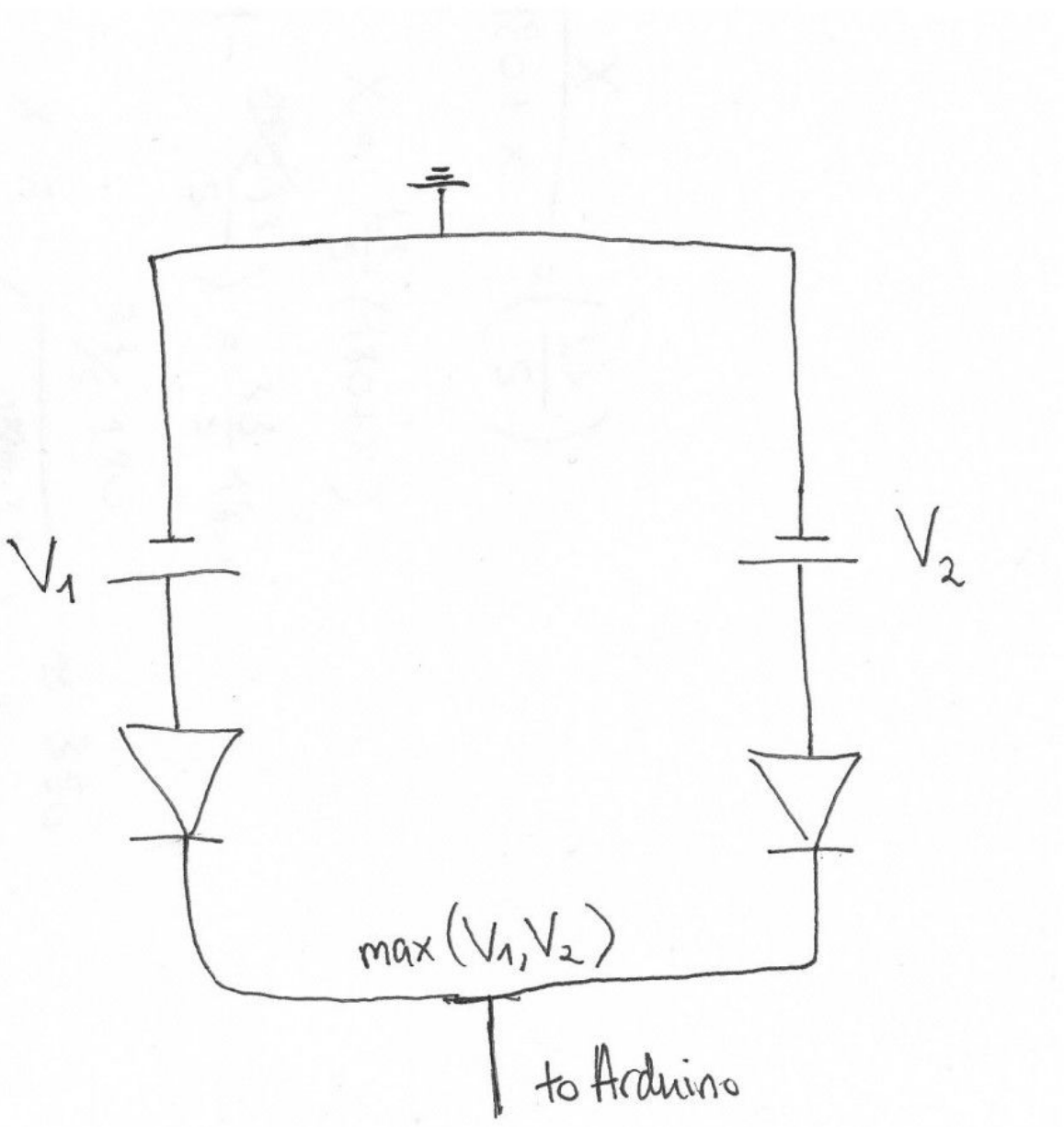


The below picture is just to illustrate what it will look like using bits of wire to make all the connections; it does not match the suggested diagram!)



Appendix C: Back-up Battery for Arduino

Building a back-up battery for your Arduino just takes a little bit of soldering! Take a look at the following circuit.

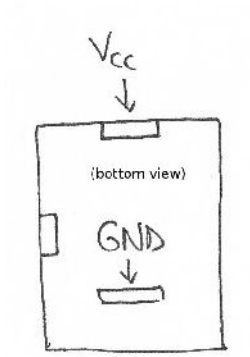


This is how the circuit works: Parallel circuits always have equal voltages (Kirchoff's Voltage Law); but when you have two parallel voltages, you can't push a high voltage down so you must push the low voltage

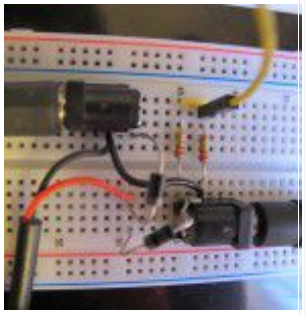
up! Our battery wouldn't appreciate that very much though, so we've adding some diodes in series of each voltage source. This means that the voltage supplied to the Arduino will be the maximum of the two voltage sources.
If you have a 9V wall wart which tends to supply roughly 9.2V and a 9V battery which tends to supply 9V or less, then only the wall wart will be drawn on (well mostly at least...)
What we're going to do is solder two barrel jacks into your protoboard. It will have the normal power supply and the back-up battery plugged into it, and will plug into the Arduino itself. Here's a step-by-step guide on how to build a back-up battery for your Arduino.
Bill of Materials

	Two barrel jacks (2.1mm)
	9V battery holder
	Two diodes
	DC cable (male)

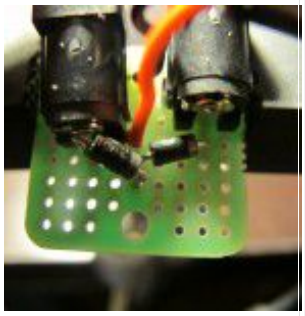
Step-By-Step

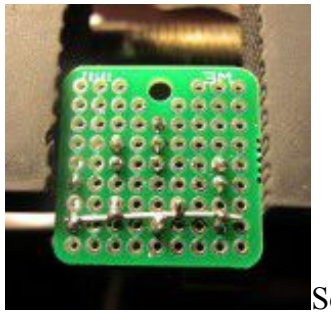


Make sure you know what's going on with the barrel jacks. The top pin is Vcc (positive) and the bottom pin is GND (negative).



Plug the circuit into your breadboard (you can ignore the part with the resistors) Make sure you have both barrel jacks and the DC cable referring to the same ground.Test it out: if you switch the battery on and off is voltage constant? if you unplug the wall wart, does the voltage drop to the battery's voltage?





Solder it into your protoboard.

Appendix D: Program Fido

First of all, make sure that all the messages on the phone have been erased. Now make sure the Arduino is reset while already plugged into the phone. After a power up, if the light blinks green, then things are fine and you can go ahead and program!
Programming Fido can be done in two text messages.
First, text Fido the default master pin, "#FIDO" - Fido will make you the owner and you will be allowed to program Fido settings.
Next, you will send a sequence of numbers seperated by commas (no spaces!) to set all of the paramaters on Fido. The numbers must be appended with "#" and go in this order:
1. How often Fido will log a measurement and save it SD card (in minutes)
2. Whether or not to activate temperature alarm (1 means true, 0 means false)
3. Lower bound on the temperature alarm, in Fahrenheit (negative numbers and Celsius not yet supported)
4. Upper bound on the temperature alarm, in Fahrenheit
5. Whether or not you want periodic updates (1 for true and 0 for false, again)
6. How frequently Fido will send you periodic updates (in minutes)
7. At what hour, in military time, you'd like updates to begin (eg: 9 for 9 am, 13 for 1pm)
8. At what hour, in military time, you'd like updates to end
9. Whether or not you want a daily summary with min, max, and avg since last update (1=true, 0=false)
10. At what hour you would like the update, in military time (eg: 20 is 8pm)
11. Whether or not you want to activate privacy (more on this later)
Here's an example text message to send Fido, if you want measurements to be logged every 5 minutes, want the lower temperature bound to be 32, upper temperature bound to be 90, want updates every 60 minutes between 11am and 3pm, want a summary at 7pm and want privacy mode on:

```
"#5,1,32,90,1,60,11,15,1,19,1"
```

Fido will send you a confirmation message telling you what the settings have been saved as! You can do this last operation again at any time to change the settings - just make sure that Fido knows you as the owner.
If you, the owner, send a text message to Fido that DOES NOT start with '#', it will automatically send you the current temperature. Ifsomeone else does, Fido will send them the current temperature only if privacy mode is off; if privacy mode is on, then strangers must register with Fido. Registering with Fido will not only allow you to request temperature even with privacy mode activated, but will subscribe you to a certain amount of alerts. Fido can remember up to 16 people including the owner; all they need to do is text "#fido" followed by a number indicating their usergroup (no space):
- nothing or '0': they will be allowed to text for current readings even if privacy is on and they will get emergency text messages
- '1': idem plus they will get daily summaries
- '2': idem plus they will get updates throughout the day (ie: they get all the communications that owner gets)
If you want to change your usergroup, simply text "#fido" followed by your new usergroup - you will automatically be removed and reentered but this does not work if you are already owner.
If you want to change ownership - text '#' followed by the master pin and the current owner will be removed and the new owner will be added.