The Traveling Geocache!

by **williamanos** on January 31, 2011

Table of Contents

Intro: The Traveling Geocache!

For those not familiar with geocaching, it is a wonderful treasure hunt/ adventure game for grownups and kids! Traditionally one finds a set of coordinates and cryptic clues on a website that hosts geocache locations. Using a hand held GPS device, one simply tromps off into the wilderness (or busy city intersection) and uses the power of observation and clue solving to find a hidden box usually with a prize inside! This is loads of fun and free! http://www.geocaching.com/

The Traveling Geocache still requires the user to get to a specific set of GPS coordinates but the hand held device is replaced by the Traveling Geocache itself. It is a locked box with an LCD display and one button. Upon pressing the button an intro screen welcomes the user to the Traveling Geocache and then displays the distance in kilometers from the target location. If the distance is larger than the programmed margin of error radius, the LCD informs you that access is denied. The box will not unlock until it is brought to the correct location! This is a great way to give a present to a friend or loved one. You can put tickets inside and set the magic location to the museum, aquarium, stadium, train station (DO NOT set a location that requires plane travel! Unopenable homemade electronic boxes are not welcome at airport security!). Any prize will do. Because the box is intended for a specific person, the prize can be far more special than in a traditional geocache.

Apart from the prize, the box also contains a log book where recipients can sign their names and write messages. There is also a usb drive with information on all of the electronics and software. This enables the recipient to reprogram the box, change the destination, customize the LCD messages, replace the prize, and re-gift the box to another friend, loved one, or trustworthy stranger. You can even leave a url somewhere inside the box where users can enter their names and destination locations enabling every user to track the life of the box!

Thanks to sparkfun and instructables for holding the microcontroller contest. It really motivated me to document and post my first ever instructable! If you like it, please vote for this project!

Please leave comments! =)

Image Notes
1. A synthesis of human, machine, and our natural physical universe, the Traveling Geocache rejects alienation of lifeforms by machines. Rather than imbue a machine with an exploitative social relation (taylorism, that light on the fry-o-lator that tells you to hurry up), one can create a machine that changes social relations positively each time it is used!
2. background illustration from adafruit's blog post

Image Notes
1. Standard Servo
2. GPS
3. 16X2 LCD
4. Locking mechanism comprised of eye hooks and a nail attaches to the eye hook on the lid of the box.
5. eye hook on the lid of the box
6. Momentary switch
7. 5 volt relay
8. 4 AA battery pack (used the missing two batteries to take this picture!)
9. servo cable
10. secret back door override cables!
11. Arduino clone w/ Atmega328
12. A book I keep of my favorite songs. A moleskine fits perfectly in the box and makes a good log book for recording each user.
13. USB drive with arduino software, libraries, schematics. Basically a compact users' manual!
14. Can't forget a pen!
15. Space for a geocache treasure!
16. a shirt
17. warm cozy sock

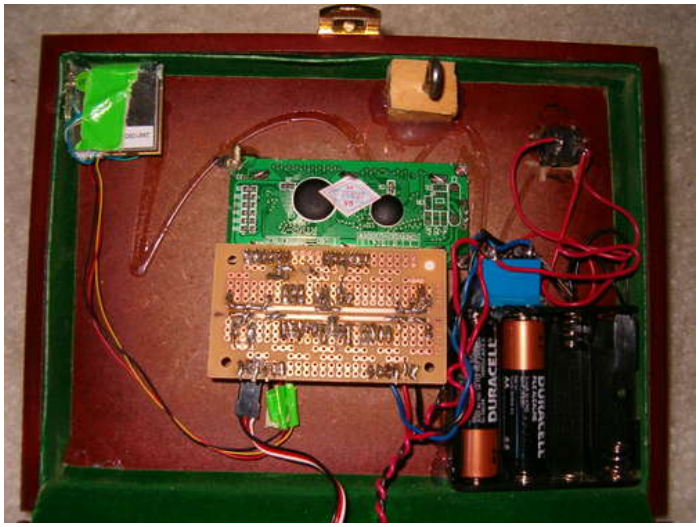
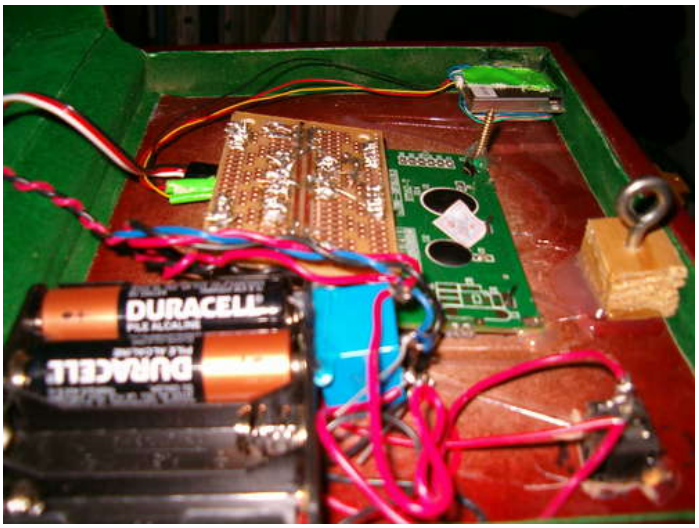
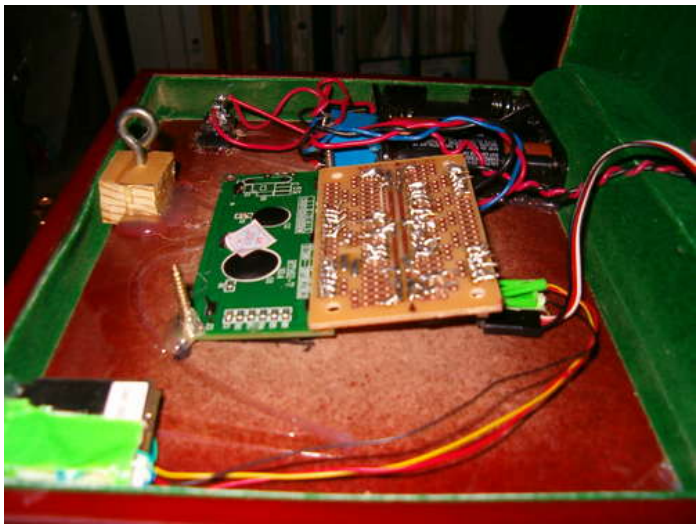


Image Notes
1. momentary switch
2. 16X2 LCD this one is white text on a blue background.





step 1: Private Property Rights are Upheld by Police with Guns

Despite our dreams of open source communism, capitalism is still a reality. With that in mind here is some info on intellectual property involved in this project.

This project was inspired by Mikal Hart's "Reverse Geocache Puzzle Box" He has been wonderful about providing information on his design. **Please note: "Reverse Geocache" and "Reverse Geocache Puzzle Box" are trademarked intellectual property of Mikal Hart.** If you think that is uncool, he DOES have all the wiring diagrams and software available for free as long as you are not profiting off of them. He did after all write several arduino libraries for the program.

http://www.youtube.com/watch?v=Lu7IysgaZf8&feature=player_embedded

The original reverse geocache:
http://arduiniana.org/projects/the-reverse-geo-cache-puzzle/

Build instructions for the original Reverse Geocache:
http://arduiniana.org/projects/the-reverse-geo-cache-puzzle/building/

Other Acknowledgements:

Thanks to lady ada for all her super helpful tutorials!

Thanks to sparkfun for keeping up the Free Day tradition. I got my very first atmega328 with my 10 free day dollars (YES, I actually made it onto their site on free day!)

Thank you as well to RickP on the arduino forums for help with the relay switch design.
http://www.arduino.cc/cgi-bin/yabb2/YaBB.pl?num=1293755082

step 2: Materials + Tools

Sparkfun Electronics is a good place to get most of what you need http://www.sparkfun.com/

List of Tools:

Soldering Iron
Wire cutters + strippers
Hot Glue gun
epoxy
dremel

List of Materials:

Cool looking box

Solder

Breadboard http://www.sparkfun.com/products/9567

Arduinohttp://www.sparkfun.com/products/9950 (you need to get one with a through hole chip. The surface mount SMD version of the arduino UNO will not work)

atmega328 chip http://www.sparkfun.com/products/9061 or http://www.sparkfun.com/products/9217

28 pin dip socket (for atmega328) http://www.sparkfun.com/products/7942

perf board http://www.sparkfun.com/products/8812

LED http://www.sparkfun.com/products/9592

1N4004 or any 1N400X diode (you can get these at radioshack)

Momentary switch

330Ohm resistor http://www.sparkfun.com/products/8377

10KOhm resistor http://www.sparkfun.com/products/8374

10k Ohm trim pot http://www.sparkfun.com/products/9806

22pf Capacitors (2) http://www.sparkfun.com/products/8571

16MHz crystal http://www.sparkfun.com/products/536

22 gauge solid core wire (multiple colors help a lot) http://www.sparkfun.com/products/8022

standard servo http://www.sparkfun.com/products/9065

gps module http://www.sparkfun.com/categories/4

LCD (I used a 16X2) http://www.sparkfun.com/products/790

5 volt relay

4 AA battery holder

4 AA batteries

female headers http://www.sparkfun.com/products/115

male headers http://www.sparkfun.com/products/116

scrap wood

large nail or metal bar

eye hooks (3)

usb stick

notebook

pen

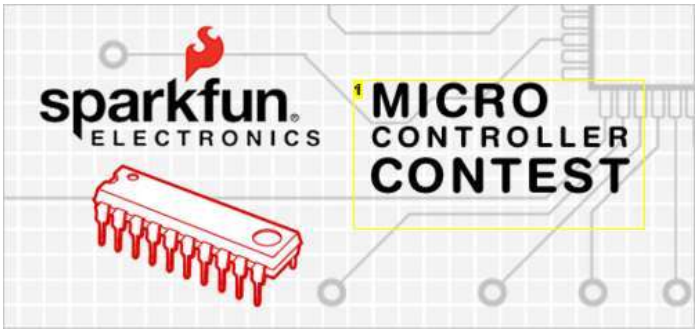


Image Notes
1. This is what motivated me to do this instructable! If you have a project you should totally enter this contest!

step 3: Cutting the Box

This picture is of the first box I attempted to cut. This is the correct pattern but I cut the LCD hole too large! Always cut a smaller hole than you think you will need and work from there. I found it easiest to use a dremel but use whatever you like.

Now is also a good time to set four screws into the bottom of your box to act as legs. Later on, we will use two of these for a secret back door entrance into the box!

Image Notes
1. LCD opening
2. switch socket

step 4: Breadboard Arduino

Here is all the info from the arduino website : http://arduino.cc/en/Tutorial/ArduinoToBreadboard

If you bought an Atmega328 with the bootloader already installed, simply follow the second half of this step. If you have a blank chip, please follow the entire step.
"

Burning the Bootloader

If you have a new ATmega328 (or ATmega168), you'll need to burn the bootloader onto it. You can do this using an Arduino board as an in-system program (ISP). If the microcontroller already has the bootloader on it (e.g. because you took it out of an Arduino board or ordered an already-bootloaded ATmega), you can skip this section.

To burn the bootloader, follow these steps:

1. Upload the ArduinoISP sketch onto your Arduino board. (You'll need to select the board and serial port from the Tools menu that correspond to your board.)
2. Wire up the Arduino board and microcontroller as shown in the diagram to the right.
3. Select "Arduino Duemilanove or Nano w/ ATmega328" from the Tools > Board menu. (Or "ATmega328 on a breadboard (8 MHz internal clock)" if using the minimal configuration described below.)
4. Run Tools > Burn Bootloader > w/ Arduino as ISP.

You should only need to burn the bootloader once. After you've done so, you can remove the jumper wires connected to pins 10, 11, 12, and 13 of the Arduino board."

Uploading Using an Arduino Board

"Once your ATmega328p has the Arduino bootloader on it, you can upload programs to it using the USB-to-serial convertor (FTDI chip) on an Arduino board. To do, you remove the microcontroller from the Arduino board so the FTDI chip can talk to the microcontroller on the breadboard instead. The diagram at right shows how to connect the RX and TX lines from the Arduino board to the ATmega on the breadboard. To program the microcontroller, select "Arduino Duemilanove or Nano w/ ATmega328" from the the Tools > Board menu (or "ATmega328 on a breadboard (8 MHz internal clock)" if you're using the minimal configuration described below). Then upload as usual."

NOTE: In addition to following all the steps from the arduino website posted above, you will need to *connect the reset pin of the arduino board to the reset pin of the atmega chip* in order to upload code.

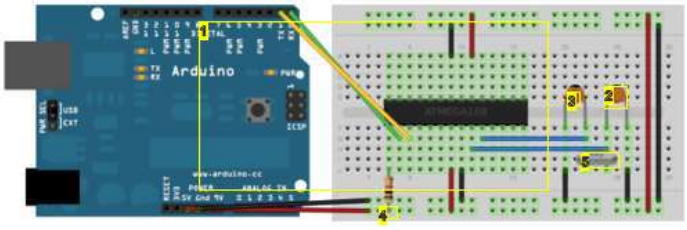
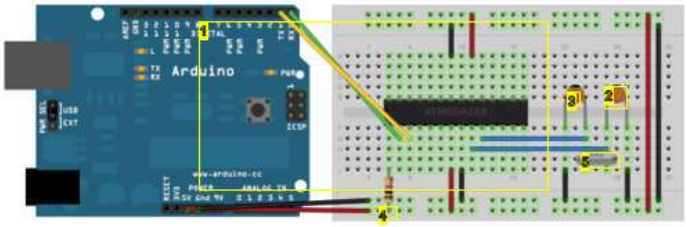


Image Notes
1. Layout for uploading code
2. 22pf capacitor
3. 22pf capacitor
4. 10k ohm resistor connects to POWER not ground. The diagram can fool you
5. 16MHz crystal



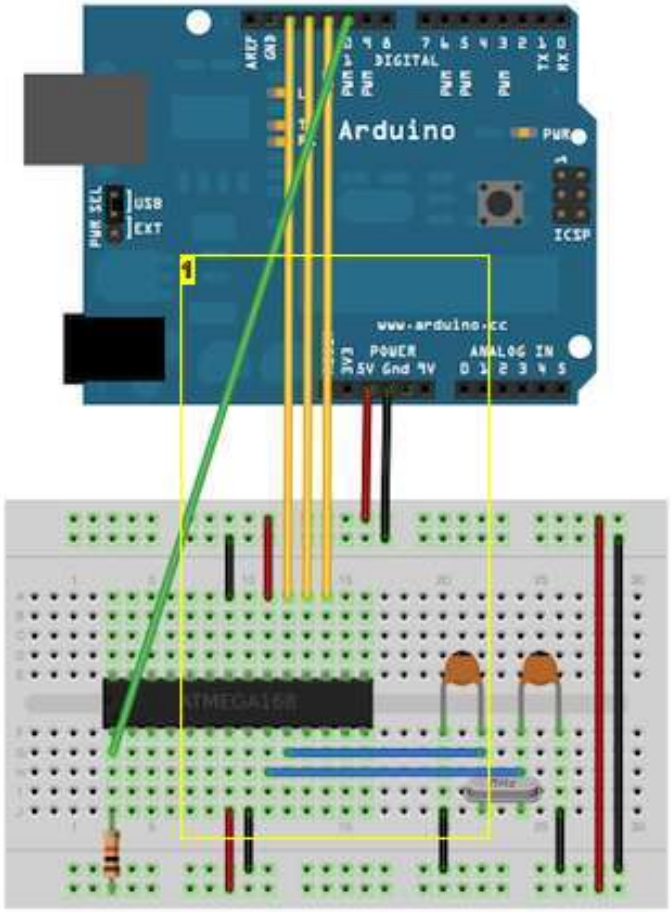


Image Notes
1. Layout for burning bootloader

Image Notes
1. works for the 328 as well
2. These are the pin names we are interested in.
3. These are the pin names we are interested in

step 5: Using a Servo

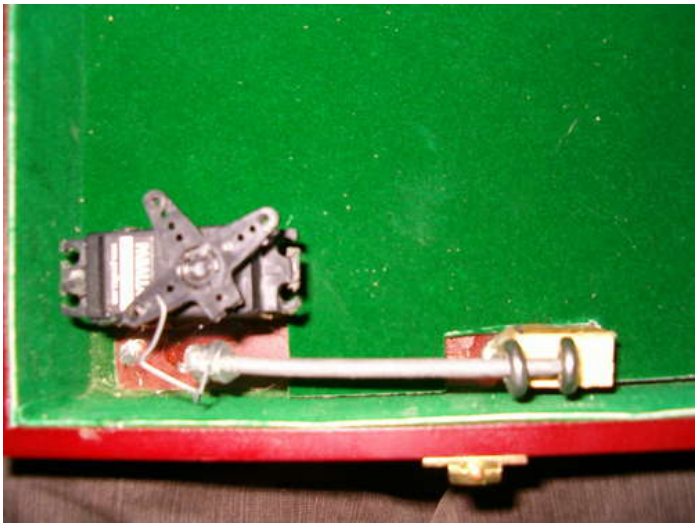
This goes for every step involving electronic components: **TRY IT ON A BREADBOARD FIRST!**

This tutorial on the arduino playground will help you out if you have never used a servo with the arduino. http://www.arduino.cc/playground/ComponentLib/Servo The code later in this instructable is already set out for you so don't worry too much about figuring out every line of code in the tutorial. What is important to know is how to connect the servo to your Atmega328 and how to adjust the rotation of the servo in the program.

A servo has three connections: Ground (black) 5 volts (red) and Signal (white). 5v and ground provide power to your servo and the signal line lets the arduino or atmega328 tell the servo what to do via one of the arduino's digital pins.

In the Traveling Geocache code there is a clearly labeled space for adjusting the angle at which your servo rotates.

This video from MAKE explains a little about servos and even shows you how to control a servo with a potentiometer (basically a knob).
http://www.youtube.com/watch?v=FKj9jJgj8Pc



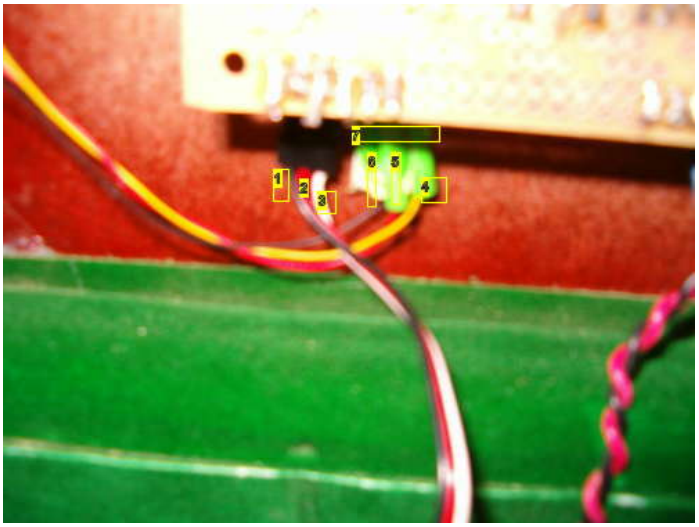


Image Notes
1. servo ground (black)
2. servo 5 volts (red)
3. servo signal (white) This wire is connected to a digital pin on the arduino.
4. gps signal pin
5. gps power
6. gps ground
7. The GPS wires were quite thin so I soldered them to some of the 22 gauge wire. This makes them fit great in the female headers. The green is some duct tape I used in lieu of heat shrink tubing.

step 6: Using a GPS

TRY IT ON A BREADBOARD FIRST

Hooking up a GPS module to the arduino is surprisingly simple. Make sure to look at the data sheet for your GPS! It should have two TTL lines for receiving and transmitting data to the arduino. Regardless of the other wires coming out of your GPS all you need to worry about are the TX or transmitting TTL line which will funnel GPS data to the arduino, and the power and ground lines. Power and ground connect to the rails on the arduino just like the servo. The TX line from the GPS attaches to a digital arduino pin just like the signal wire on the servo.

The GPS I am using used to be available from parallax but has been discontinued:
http://www.parallax.com/Store/DiscontinuedProducts/tabid/795/CategoryID/69/List/0/Level/a/ProductID/560/Default.aspx?SortField=ProductName%2CProductName

It was the cheapest I was able to find a year ago, but now you may have to spend 40 dollars on one.

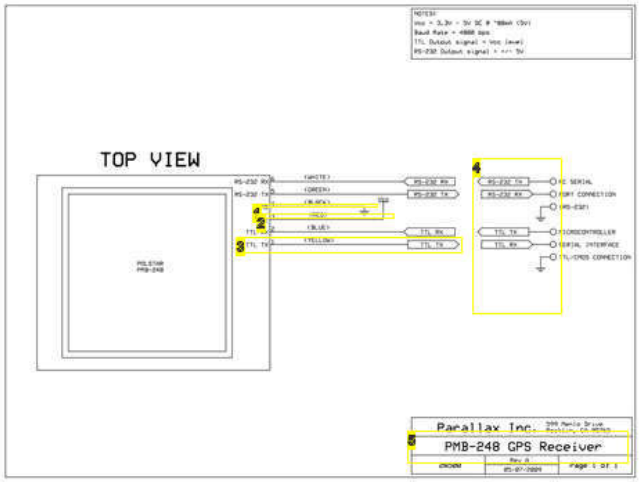
The GPS sends a string of NMEA data to the arduino. To the untrained eye it is a dizzying garble of numbers and punctuation. The arduino gps library and Mikal Hart's TinyGPS library act as interpreters for this data. They cut it up, label it, and display it in the serial terminal.

Again the arduino playground is a good place to brush up on interfacing GPS with arduino: http://www.arduino.cc/playground/Tutorials/GPS

If you get frustrated with the tutorial try using Mikal Hart's TinyGPS library. It is what the main program will be using anyways: http://arduiniana.org/libraries/tinygps/

NOTE: DO NOT connect the TX line of the GPS to the RX of the arduino as shown in the diagrams. This is because we need to use the arduino RX pin for programming and reprogramming purposes. Instead simply connect it to any of the digital pins 2 to 13.

If you have hooked everything up correctly and are just getting garbled gibberish from the serial terminal, check your baud rate and make sure it matches the rate set for the GPS device in the code you are using!



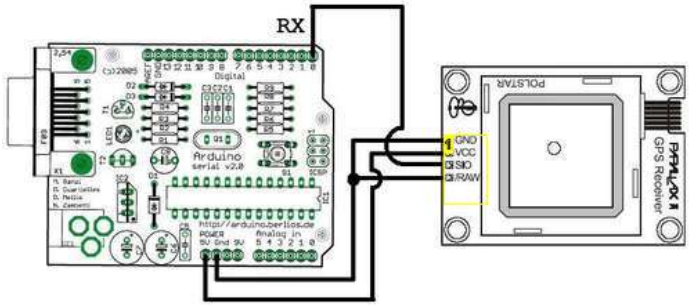


Image Notes
1. the data sheet for my gps was labeled as described above. For the gps in the schematic the SIO would be the TX TTL pin, not sure what the bottom pin is.

Image Notes
1. ground
2. power
3. tx line
4. disregard this. Tx goes to digital pin 2.
5. the module used in this project

step 7: Using an LCD

I learned how to use an LCD with arduino from the ladyada tutorial here: http://www.ladyada.net/learn/lcd/charlcd.html

This is a really in depth look at hooking up and operating LCDs with the arduino. As with every step, try this out on a breadboard before soldering anything to your perf board arduino!

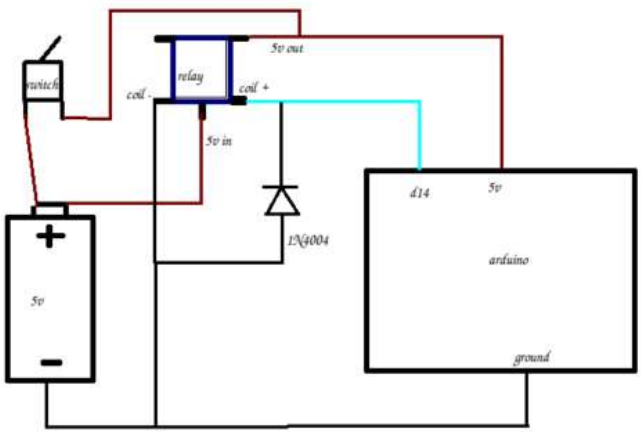
Image Notes
1. This photo is from ladyada's tutorial in the link

step 8: Using a Relay for Auto-turn-off

TRY ON A BREADBOARD FIRST

A relay works by activating an electromagnet to mechanically throw a switch. To use one you need to place it on the breadboard and test all the contacts. What you are looking for or rather listening for is a clicking sound. When you connect one end of the coil to ground and the other end to power it will activate and make a clicking noise. That click is the internal switch being physically drawn to a contact by the electromagnet. After finding the two coil leads, experiment with the remaining leads using an led with a resistor to determine where electricity can flow depending on the coil's state. The coil will allow a separate power source to flow between two other pins on the relay. You are looking for the led to turn on when you connect the coil. The led will be where your circuit will go.

Note: The relay in the schematics is viewed from overhead as if it were on a breadboard. The photo shows the relay mounted upside down in the box for easy access to the terminals.



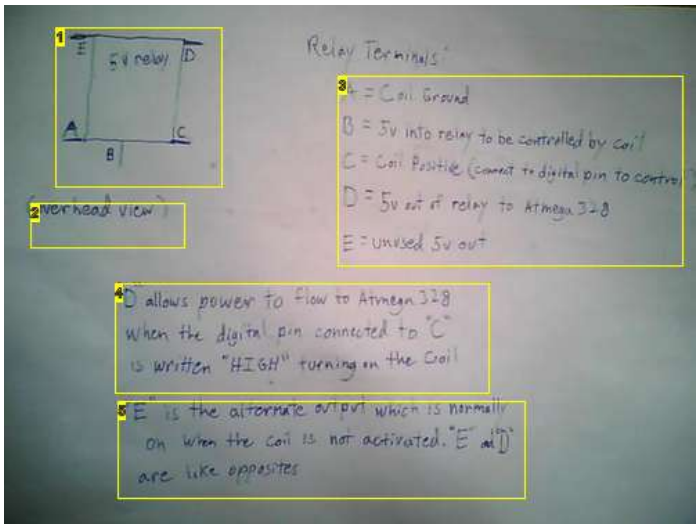


Image Notes
1. Because I could not find the data sheet for the relay online, I have arbitrarily assigned these letters to the terminals for the sake of explanation.
2. the relay is labeled as if you were looking down on it set in a breadboard.
3. A = Coil ground B = 5 volts into relay to be controlled by the coi C = Coil positive (connect to digital pin for controlling relay) D = 5 volt out of relay to power Atmega 328 E = unused 5v out
4. D allows power to flow to Atmega 328 when the digital pin connected to "C" is written HIGH turning on the coil.
5. "E" is the alternate output which is normally on when the coil is not activated. "E" and "D" are like opposites.

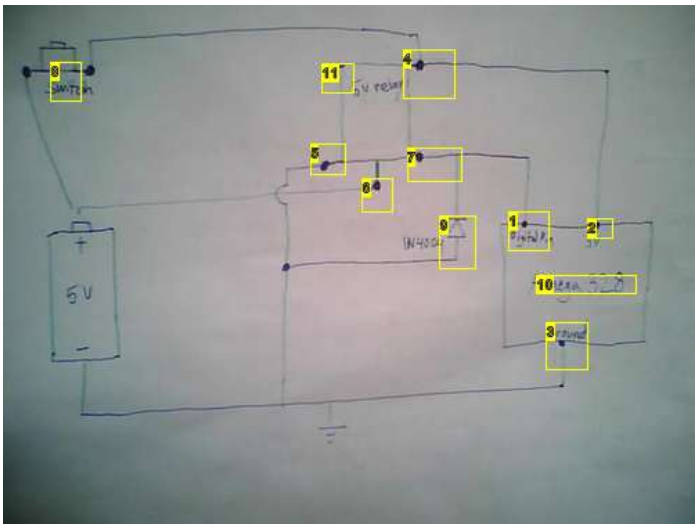


Image Notes
1. Digital Pin
2. 5 Volt to power the arduino
3. Ground
4. relay pin out latches power to the arduino. The switch supplies initial power while the arduino starts up and activates the coil, fixing it in an 'on' state until the program on the arduino decides to cut the power and shut itself down.
5. ground
6. Relay power in. The power connected here flows through the relay to the arduino if the coil is activated.
7. positive end of the coil. power for the coil comes from a digital pin on the arduino. When the pin is written HIGH the coil turns on and power flows from the battery through the relay and to the arduino allowing you to let go of the button.
8. switch or button
9. 1N4004 blocking diode to protect circuit when relay coil powers off
10. Arduino
11. Unused relay out. This is the opposite of the connection directly to the right. When the coil is off this contact is on. When the coil is on, this contact is off.

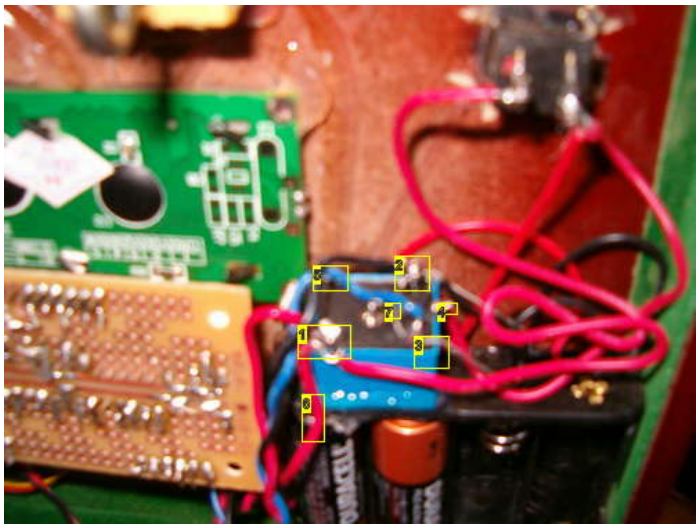


Image Notes
1. 5 volts to board
2. ground. Secret override ground also goes here. Situating the relay with the leads exposed is a great way to wire in new components!
3. coil 5v connected to digital pin
4. 5 volts into relay controlled by coil
5. unused relay out
6. secret back door power wire
7. 1N4004 diode (positive to terminal "A", negative to terminal "C")

File Downloads

relaycircuit.pdf (12 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'relaycircuit.pdf']

step 9: Perfboard Arduino

I found this instructable by jmsaavedra helpful in making the leap from breadboard to perf board: http://www.instructables.com/id/Perfboard-Hackduino-Arduino-compatible-circuit/

Basically your goal is to recreate the arduino breadboard circuit on this perf board making it smaller, stronger, and more permanent.

All you need to put on the board is the same circuit you had on the breadboard with the addition of an LED on pin 13. Everything else in the above example you can ignore.

Keep your connections farther from the board edge than jmsaavedra because we need those outer spaces for female header connections and the LCD. Try to keep it tight!

The power regulator used in the above circuit isn't really necessary for our project because the operational voltage of the atmega328 is up to 5.5 volts and we are only going to be slightly over that at the very beginning of the Traveling Geocache's life. That being said, if you are worried about damaging the circuit you will need a regulator with less than the 2 volt drop of the 7805 regulator used in jmsaayedra's hackduino. The 7805 would make the maximum voltage available to the circuit 4 volts. This is not enough to optimally power the LCD and the servo.

The perfboard is going to be the central hub of your box. The servo, GPS, LCD, batteries, relay, and switch will all be connecting to this board in one way or another. Before getting into any of those connections however make sure you can run the simple blink LED program demonstrated in jmsaavedra's instructable. This will let you know if your board is working to begin with before complicating any troubleshooting process with more wires.

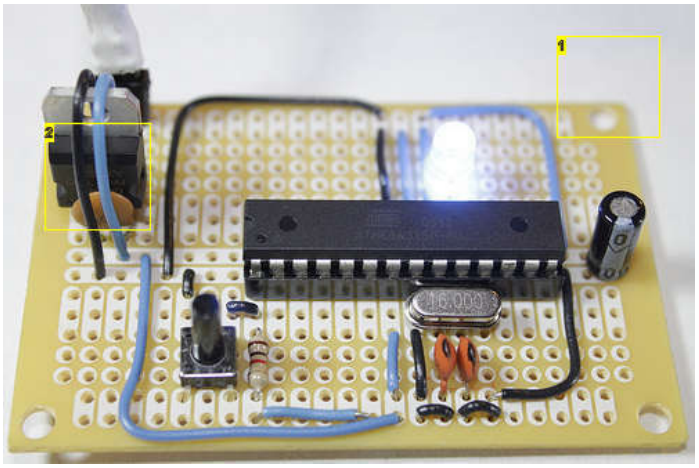


Image Notes
1. This photo is from jmsaavedra's instructable in the link
2. use a low voltage drop regulator if you want one

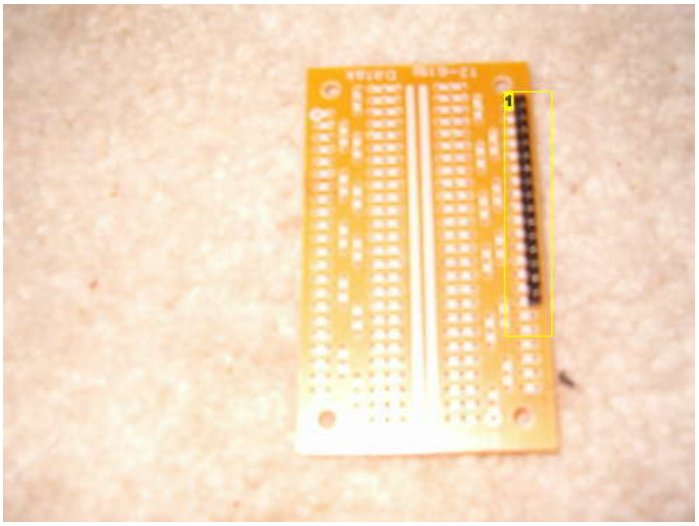


Image Notes
1. this header is paced where the LCD will connect to the board. Keep the bare bones arduino circuit away from the edges as much as possible to leave room for the LCD, gps, servo, power, ground, reset, and relay latch wires later.

step 10: Wiring it all up!

Follow the schematics and photos. If the schematic images are too blurry, they are also available in pdf form. Make sure to read the image notes though!

The important thing to remember is that the
power,
ground,
reset,
tx,
rx,
analogue 0,
digital 2,
gps power,
gps ground,
digital 9,
servo power,
and servo ground
are all connected to the board via stackable female headers. SEE THE PHOTOS!

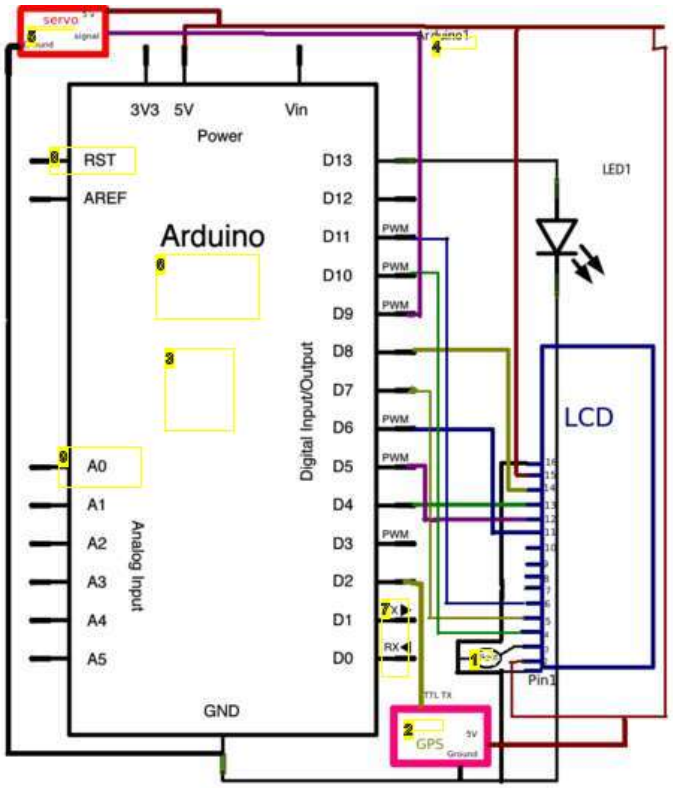


Image Notes
1. 10K Ohm potentiometer for adjusting LCD contrast
2. GPS REMEMBER: The GPS attaches to the arduino board via the stackable headers in the photos. This is a purely electrical diagram.
3. Check the PDF for a closer look!
4. Ignore this text. It means nothing.
5. Servo REMEMBER: The servo attaches to the arduino board via the stackable headers in the photos. This is a purely electrical diagram.
6. Note that this circuit is just for the modules. It does not include the break out wires from RX, TX, RESET, 5v, Ground, or Relay digital pin. It also does not contain the 10k resistor, LED resistor, 16MHz crystal, or two 22pf capacitors that should already be on your board.
7. Don't forget to connect these to the female header pins! (as shown in the photos)
8. Don't forget to connect this to the female header pins! (as shown in the photos)
9. Don't forget to connect this to a female header pin! (as shown in the photos)

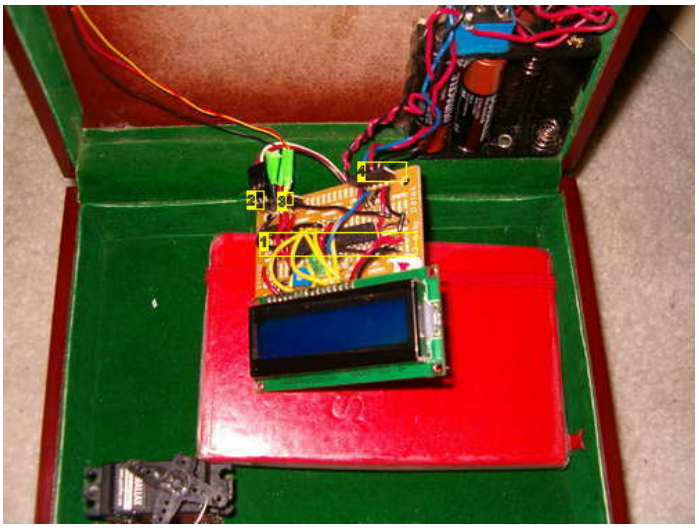


Image Notes
1. The beautiful beautiful board!
2. servo connection with a three prong male header bent at 90 degrees to fit into half of the 6 pin stackable header.
3. gps wires stuck into the other half of the 6 pin header

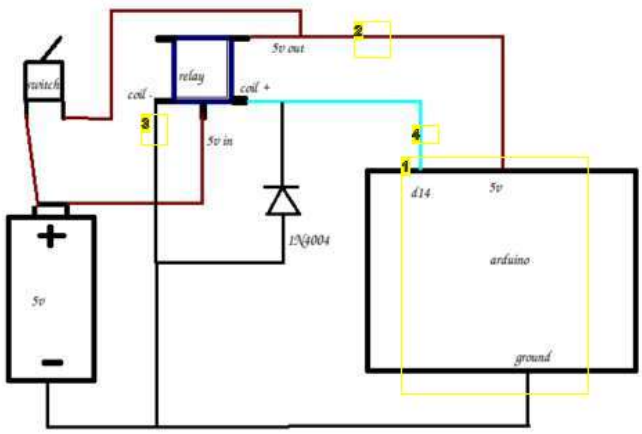


Image Notes
1. Take note that all three wires going to the arduino in this electrical diagram will attach to the board via the stackable female headers shown in the photos.
2. this is where you will eventually connect the power for secret access.
3. This is where you will eventually connect the ground for secret access.
4. d14 or digital pin fourteen is labeled as "analogue 0" in the arduino diagram and on regular arduinos.

4. during operation the only wires connected here are power, ground, and a digital pin for the relay. the other three connections on this header are for programming. they are reset, tx, and rx.

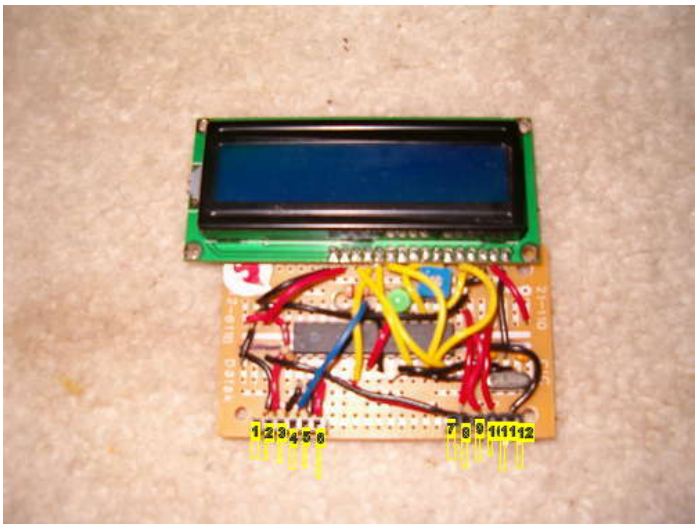


Image Notes
1. to ground
2. to power
3. to reset
4. to digital pin 14 (also known as analogue 0)
5. to rx
6. to tx
7. to digital pin two (gps data)
8. to 5 volts (GPS power)
9. to ground (GPS ground)
10. to digital pin 9 (servo signal)
11. to 5 volts (servo power)
12. to ground (servo ground)

Image Notes
1. I cut up the edges of the stickers sparkfun sends out in their shipments to make labels for these sockets
2. yellow wires are the digital pin connections for the LCD
3. Servo and GPS
4. Power and programming

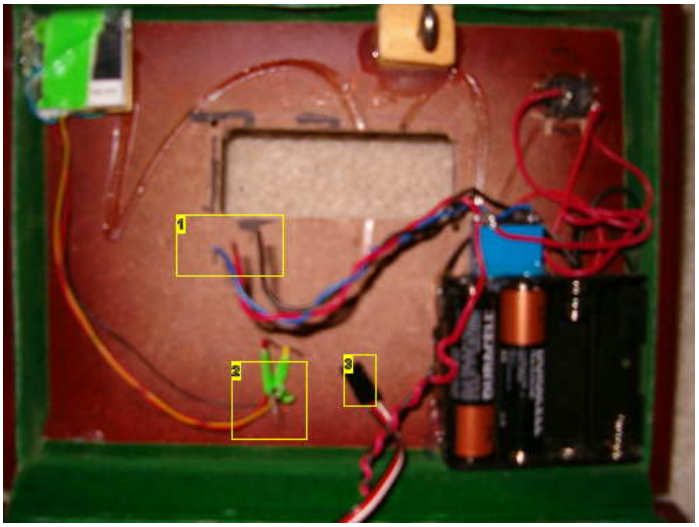


Image Notes
1. three wires coming from the relay to the board control power
2. 3 wires from the gps
3. servo wires

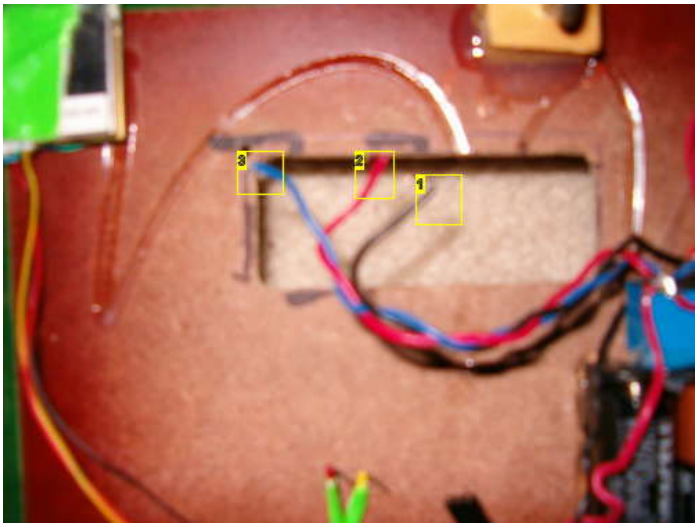
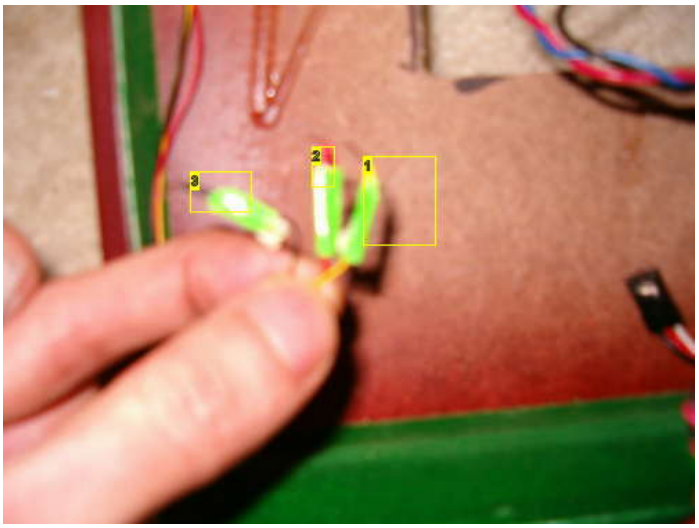


Image Notes
1. ground from relay to board
2. power from relay to board
3. coil wire from relay to board (digital pin 14 aka analogue 0)



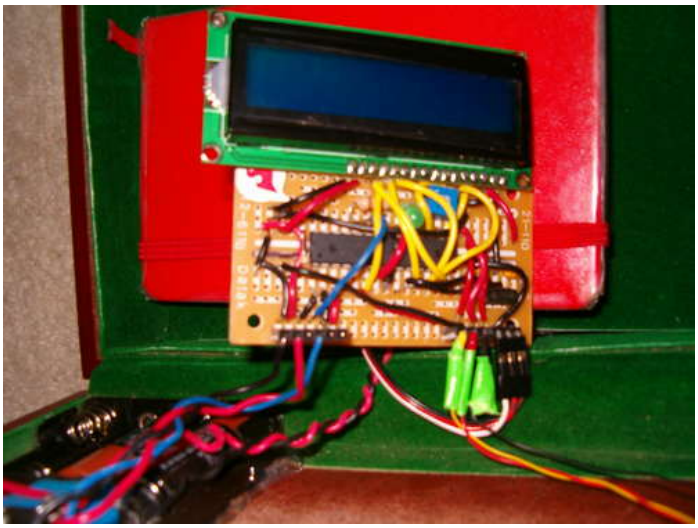


Image Notes
1. gps data goes to digital pin 2
2. gps power
3. gps ground

File Downloads

modueshookup.pdf (67 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'modueshookup.pdf']

relaycircuit.pdf (12 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'relaycircuit.pdf']

step 11: Secret Back Door Wires

This super secret wiring will allow you to open the box even when it is not at the correct location. The arduino program has a function in it which is activated only when the really switch power is being bypassed. When the program normally powers off (and says powering off) the secret wire terminals will keep the whole system running. After a few minutes the servo opens the box for 10 seconds before sealing the box back up.

If you haven't already, now is a good time to add four metal legs to the box. I used some screws that were lying around my basement. You can use any two of these for the secret wires, but the back two were closest to the circuit so I chose those ones. Simply solder a wire to each screw. Assign one to be red and one black so you know what is going where. Take the red wire and solder it to the relay pin which carries 5 volts from the relay to the arduino. Take the black wire and solder it to the ground connection on the relay. Cover the screws in hot glue and you have made your back door circuit! You can test it by applying 4 AA batteries to the screw on the bottom of the box. Be sure you remember which is ground and which is power! If it is wired correctly the circuit should turn on and stay on after the "powering down" message is displayed

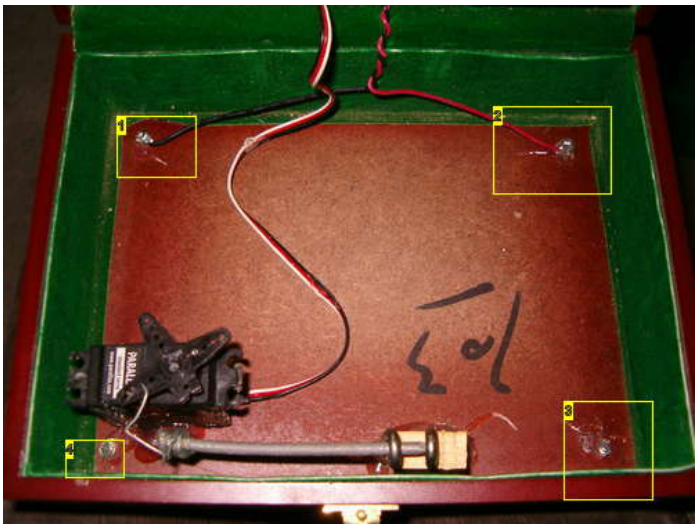


Image Notes
1. Ground for secret override power supply
2. 5 volts for secret override power supply
3. front legs so the back ones aren't so supicious
4. front legs so the back ones aren't so suspicious

Image Notes
1. secret override terminals disguised as legs!

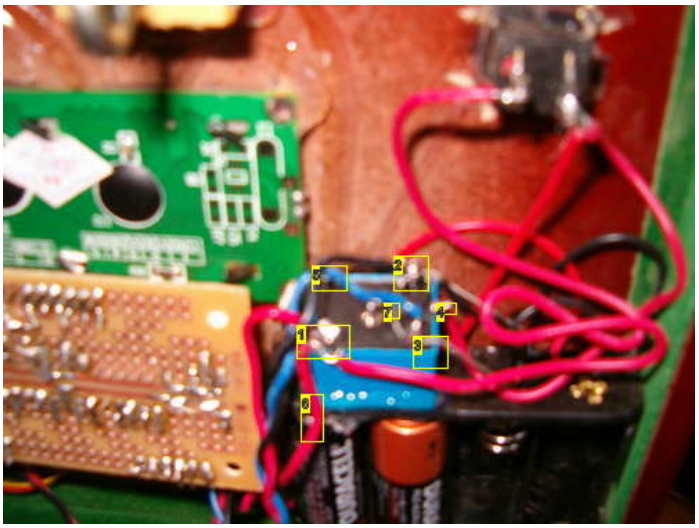
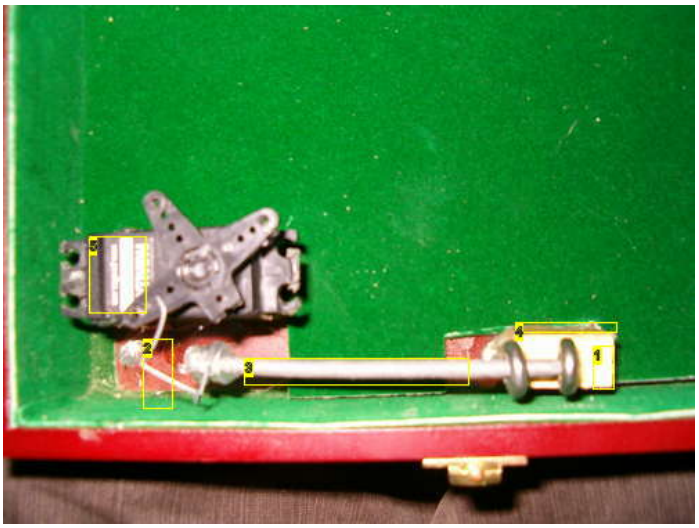


Image Notes
1. 5 volts to board
2. ground. Secret override ground also goes here. Situating the relay with the leads exposed is a great way to wire in new components!
3. coil 5v connected to digital pin
4. 5 volts into relay controlled by coil
5. unused relay out
6. secret back door power wire
7. 1N4004 diode (positive to terminal "A", negative to terminal "C")

step 12: Locking Mechanism

The locking mechanism is pretty straightforward. A servo is attached to a nail with a paper clip. One end of the paper clip is bent and inserted into a hole on the servo horn. This allows the paper clip to pivot. The other end of the paper clip is hot glued to the head of the nail.

If your box has thick sides you can screw the hooks right into the box. Otherwise a scrap of wood epoxied to the box works great for setting the hooks!



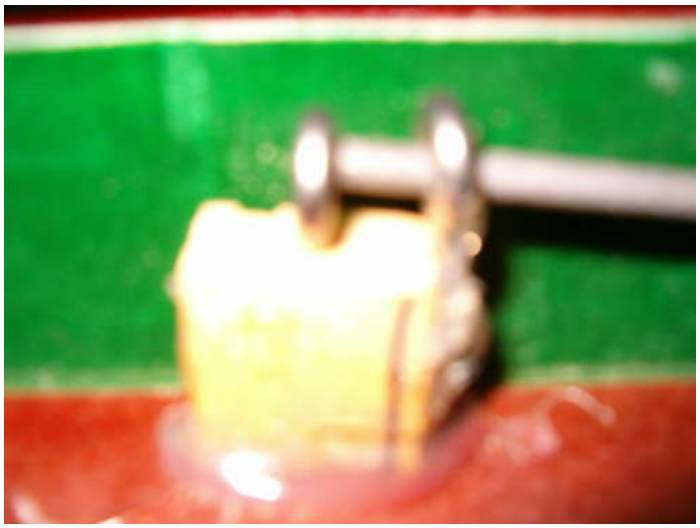


Image Notes
1. Lid hook rests here when closed.
2. paperclip
3. nail
4. random block of wood
5. SERVO!

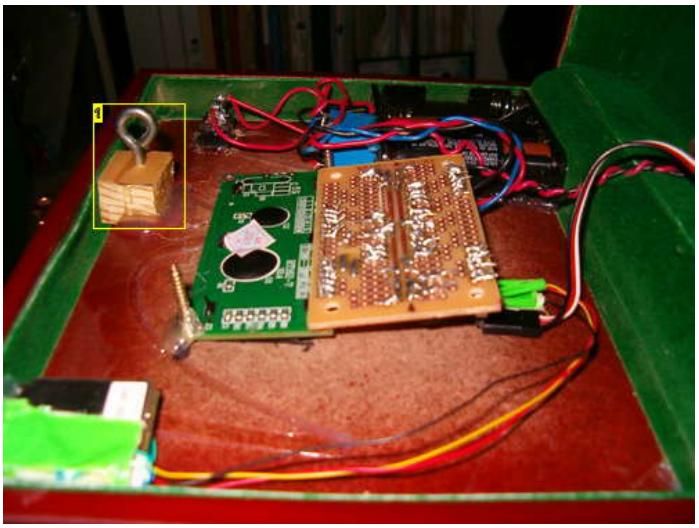


Image Notes
1. Eye hook on the lid is positioned directly next to the lower hook when lid is closed

step 13: Programming and Software

Software

The software is based on Mikal Hart's "Reverse Geocache" code from Make Vol 25 http://www.make-digital.com/make/vol25?pg=146#pg146

Everything you will need to customize is commented in the code. Remember the LCD is 16 characters across so if you want to center text on the screen, add some spaces before the words.

Things you may want to change with each use:

Destination (get latitude and longitude by finding the location on google maps!) = **line 46**

Name of recipient = **line 105**

Distance from target necessary to open box (called RADIUS in the code) = **line 48**

"Welcome to BLANK" message upon opening = **line 158**

Maximum attempt reached message (I suggest having your url display on the LCD if the maximum 50 attempts is reached.) = **line 117**

For an explanation of the arduino libraries used see Mikal Hart's blog: http://arduiniana.org/

Programming

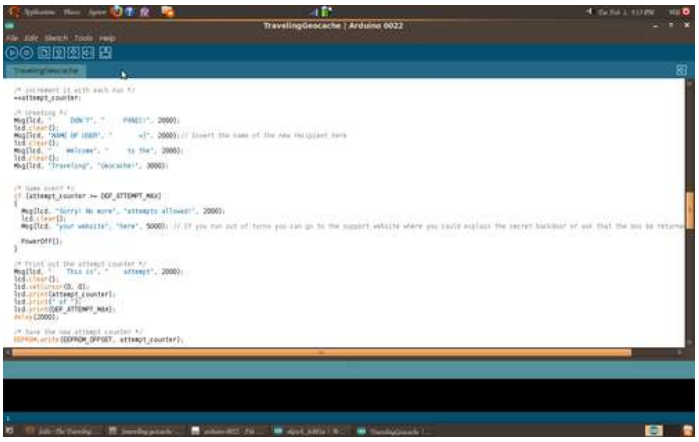
You can program your perfboard arduino the same way you programmed the arduino on a breadboard in step 4. All of the necessary connections are brought out to female headers for running wires to an arduino. If you use a full arduino instead of a perfboard arduino you can use the usb jack on the arduino itself.

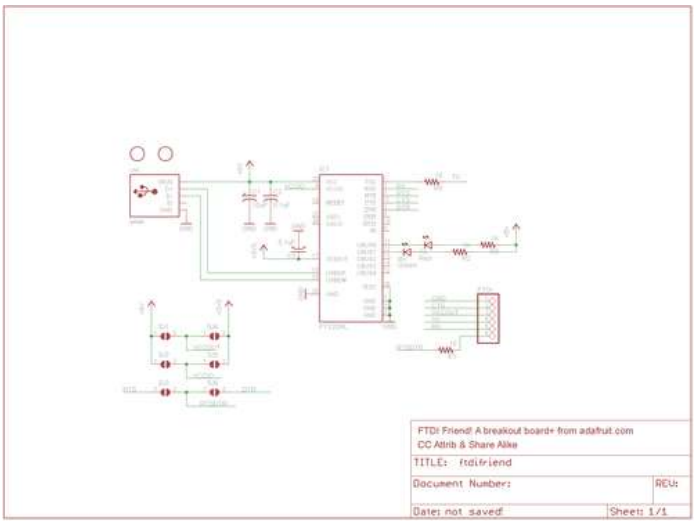
I chose a perfboard because it is smaller and cheaper than giving away a 30 dollar arduino with each box. I am currently waiting on a FTDI Basic breakout board from sparkfun: http://www.sparkfun.com/products/9716 These are cheaper than a whole arduino and can be embedded anywhere in your box.

If you have any soldering experience, I recommend making your own usb to serial module. The ftdi chip is around 4 dollars and only requires a few components. See the pdfs below for some schematics on the ftdi circuits.

NOTE: This is the top priority for this project. Info and photos on the ftdi module will be coming shortly!

When you wish to reprogram your box be sure to run the eeprom_clear example program before loading your new Traveling Geocache code. This will set the attempt counter back to 0.





File Downloads

TravelingGeocache.pde (6 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'TravelingGeocache.pde']

ARD-PROGRAMMER_Schematic.pdf (18 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'ARD-PROGRAMMER_Schematic.pdf']

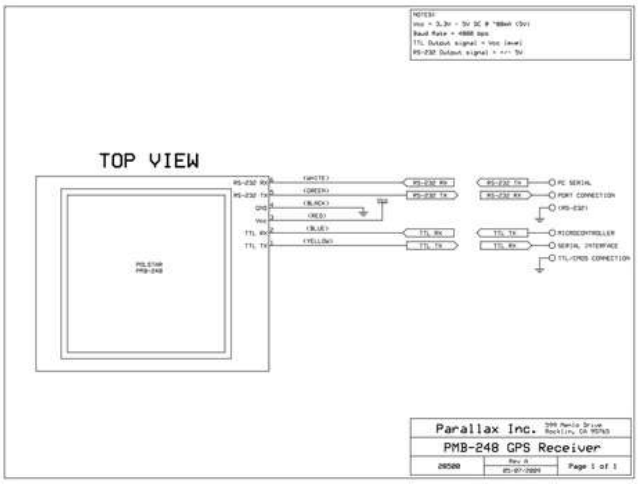
FTDI Basic-v21-5V.pdf (16 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'FTDI Basic-v21-5V.pdf']

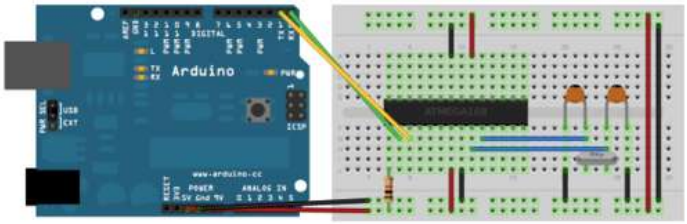
DS_FT232R.pdf (768 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'DS_FT232R.pdf']

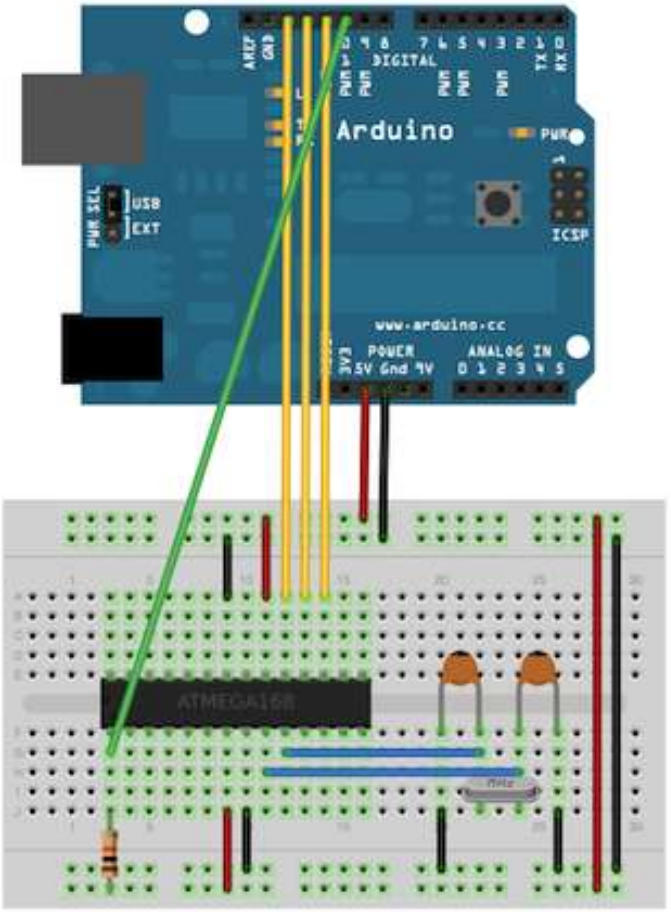
step 14: Load USB Stick

Get out your trusty USB stick (not your favorite as it will be leaving you shortly) and put the attached files on it.

Note: as of this very moment the pdf of this instructable is not yet in this step. It will be very shortly! This pdf will act as the users' manual.



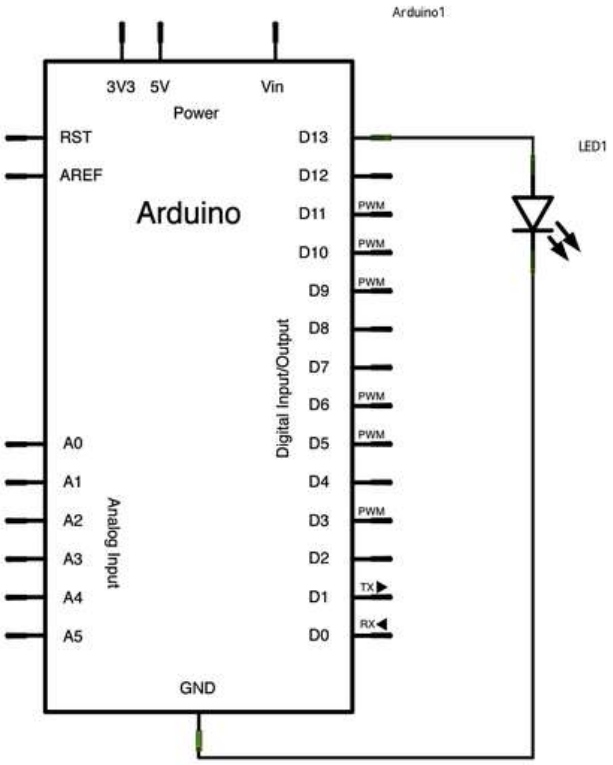




Atmega168 Pin Mapping

Arduino function				Arduino function
reset	(PCINT14/RESET) PC6	1	28 PC5 (ADC5/SCL/PCINT13)	analog input 5
digital pin 0 (RX)	(PCINT16/RXD) PD0	2	27 PC4 (ADC4/SDA/PCINT12)	analog input 4
digital pin 1 (TX)	(PCINT17/TXD) PD1	3	26 PC3 (ADC3/PCINT11)	analog input 3
digital pin 2	(PCINT18/INT0) PD2	4	25 PC2 (ADC2/PCINT10)	analog input 2
digital pin 3 (PWM)	(PCINT19/OC2B/INT1) PD3	5	24 PC1 (ADC1/PCINT9)	analog input 1
digital pin 4	(PCINT20/XCK/T0) PD4	6	23 PC0 (ADC0/PCINT8)	analog input 0
VCC	VCC	7	22 GND	GND
GND	GND	8	21 AREF	analog reference
crystal	(PCINT6/XTAL1/TOSC1) PB6	9	20 AVCC	VCC
crystal	(PCINT7/XTAL2/TOSC2) PB7	10	19 PB5 (SCK/PCINT5)	digital pin 13
digital pin 5 (PWM)	(PCINT21/OC0B/T1) PD5	11	18 PB4 (MISO/PCINT4)	digital pin 12
digital pin 6 (PWM)	(PCINT22/OC0A/AIN0) PD6	12	17 PB3 (MOSI/OC2A/PCINT3)	digital pin 11(PWM)
digital pin 7	(PCINT23/AIN1) PD7	13	16 PB2 (SS/OC1B/PCINT2)	digital pin 10 (PWM)
digital pin 8	(PCINT0/CLKO/ICP1) PB0	14	15 PB1 (OC1A/PCINT1)	digital pin 9 (PWM)

Digital Pins 11,12 & 13 are used by the ICSP header for MISO, MOSI, SCK connections (Atmega168 pins 17,18 & 19). Avoid low-impedance loads on these pins when using the ICSP header.



File Downloads

arduino-0022.tar.gz (3 MB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'arduino-0022.tar.gz']

TravelingGeocache.pde (6 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'TravelingGeocache.pde']

modueshookup.pdf (67 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'modueshookup.pdf']

relaycircuit.pdf (12 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'relaycircuit.pdf']

step 15: Fill your Box with Goodies!

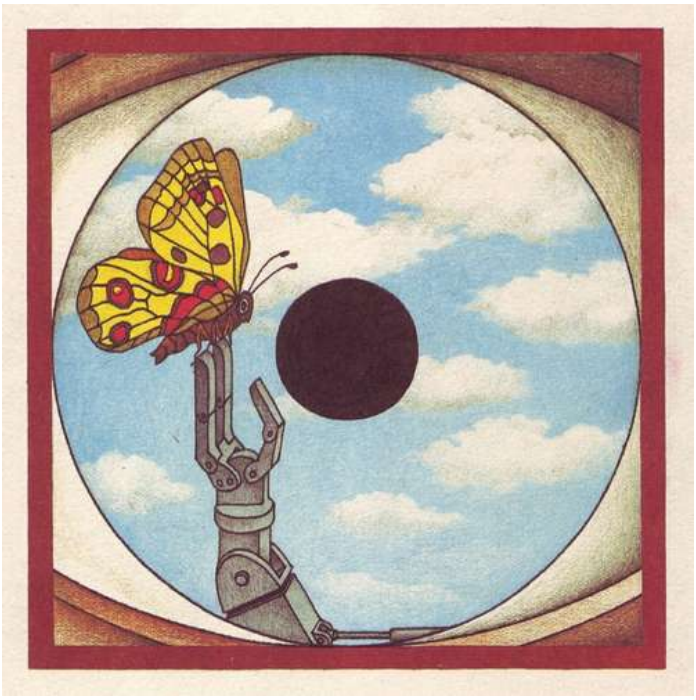
In addition to the log book, pen, and usb there is space for special geocache treasure! Keep in mind who you are giving the box to.

Wedding rings, keys to a new car, encrypted wikileaks files, cards, tickets, gold, consumables, arduinos, glow in the dark glass jellyfish, go crazy!

This part is all up to you.

step 16: Release into the Wild!

Do a test run to make sure everything is functioning properly. Make sure you can install the arduino software with the usb and that the code runs fine before releasing your Traveling Geocache into the big beautiful world! Also, install NEW batteries with each gift!



step 17: Thoughts, Concerns, Considerations

Final Thoughts:

The Traveling Geocache is not the cheapest of gifts. It is important to choose a good person to give it to. They should be trusted to actually find the location, not dismantle the box (steal components etc), and keep the box in circulation by giving it away. It is also a bonus if the recipient is interested in programming or electronics, but this is not necessary.

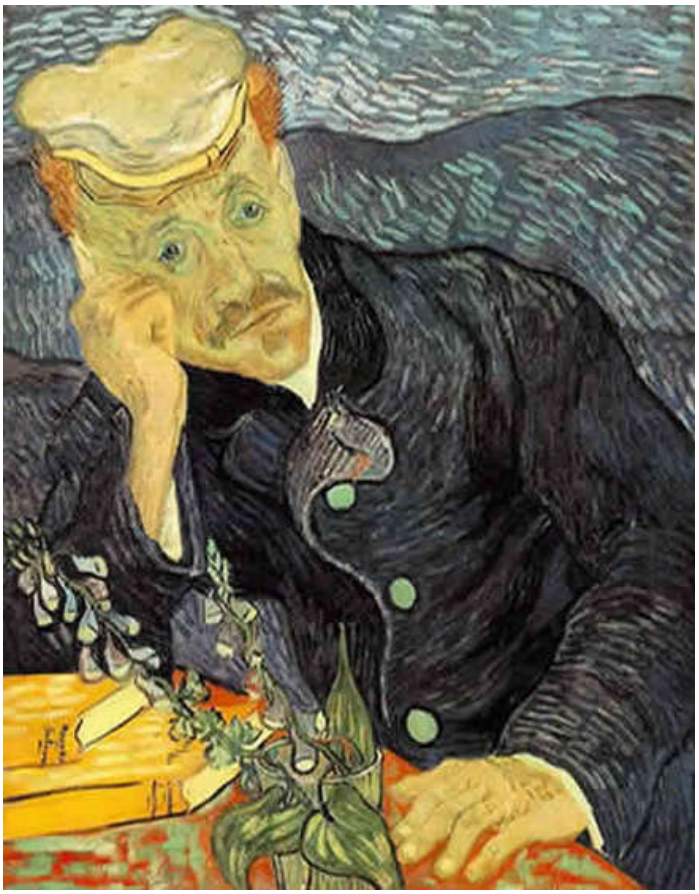
For me, the really exciting part of this project is being able to track where the box goes. To be sure this is possible you may want to write the url of your tracking site on the first page of your log book. As noted in the code, the url should also display on the LCD when the destination is reached. Your website could be open to the pubic or require a password kept somewhere in the code or box. It could have space for media (pictures and video of where the box has been) and user blog entries!

I like the contrast of the log book and website. Something about using electronics to access a physical book makes opening the box more fun.

REMINDER: **NO PLANES!**

REMINDER: **USB to serial interface coming soon! (soon as sparkfun packages make it through a continent-wide snow storm)**

NOTE: The code has a victory tune play when the destination is reached. This is played through a speaker wired to digital pin 16 (analogue 2). Pics with the speaker coming soon!



Related Instructables

The Great Guide Of Geocaching. by james.mcglashan

Steampunk Geocaching by r10n

How to Geocache by PineapplebobTheG

How to make a geocache by matthewja

Geocache Without A GPS by Derelicte

Geocache-A-Forest DIY Kit by rrarunan