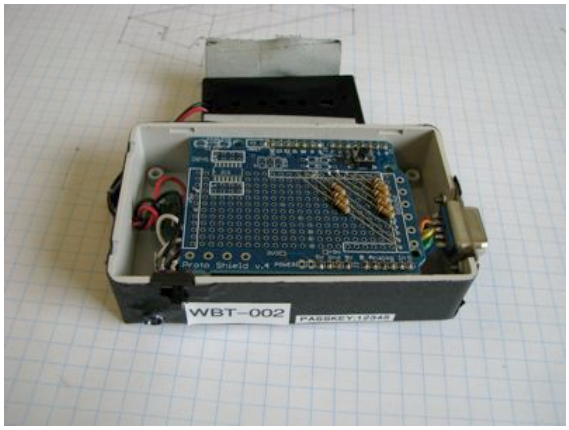
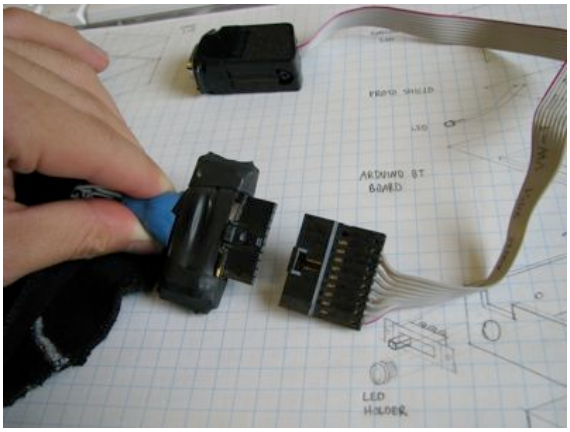
DIVA Bluetooth Modules May 2009

Johnty Wang
johntywang@gmail.com





Introduction

This document describes the design and implementation of the wireless bluetooth modules used for the DIVA system. Currently, two modules are used: one for the foot switch for controlling the system sound, and another for the left hand glove that controls the plosives and unvoiced consonants. The main purpose of each module is to detect signals from the user input and transmit them to the central computer running the DIVA software.

In the current setup, the Warbler version of DIVA ForTouch, the left hand glove consists of one common contact (thumb) and eight finger contacts. The contacts are conductive pads sewn onto the glove and wired to the connector plug via conductive thread and snaps. The snaps allow the wiring to be removed from the washable parts of the glove so it can be cleaned easily. Each one of the eight finger contacts can be connected to the thumb and are essentially 8 one-hot switches (it is possible to stretch the glove to make more than one connection at a time, but that is not required nor appropriate for the current system). The foot control is a simple on/off switch. The purpose of the wireless module is to detect the signals from the sensors and transmit them to the computer.

The module is largely based on the wireless insole designed and built by Justin Love (justinlove@gmail.com) and uses the Arduino Bluetooth as the embedded device. Therefore, the interface code already exists and can be used with slight modification.

One of the key features of the module is that it is very flexible and can be extended for future use. For example, there are analog inputs that can be used for more continuous control of signals (such as volume switching, etc).

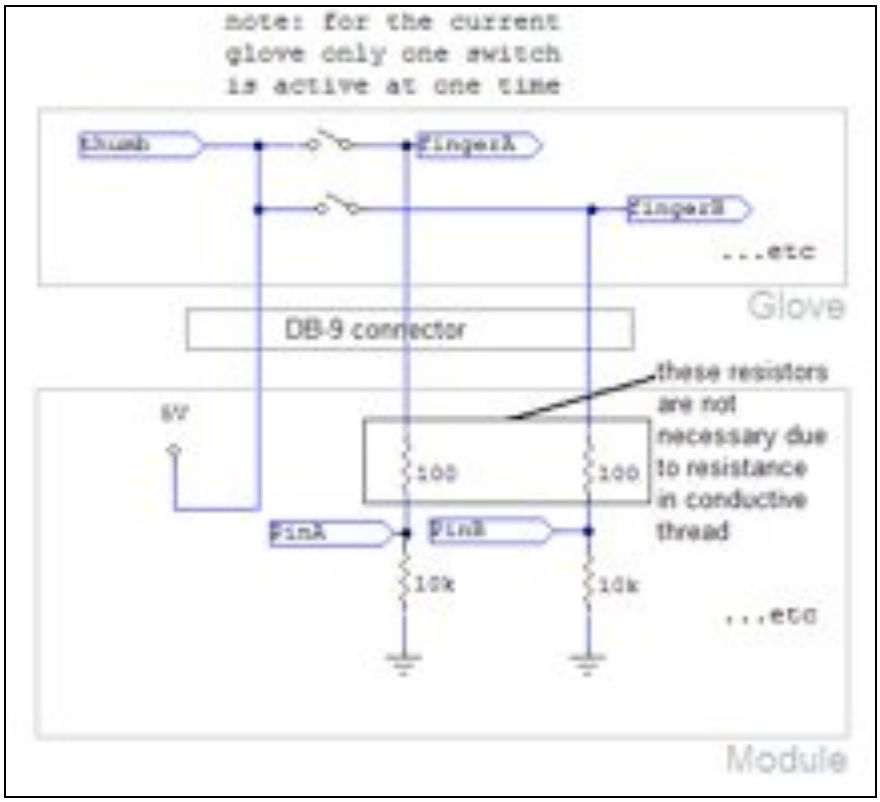
System Overview

Hardware

Circuit

Since the interfacing to the Arduino involves just a bunch of switches, the actual hardware design is relatively simple: A series of switches are hooked up to the digital input pins, and pulled down by a 10k resistor when not active. The main challenge is to build the physical interface to connect the signals in a compact and robust casing. The Arduino Proto-Shield provides a perfect platform for the switch signals to be soldered, as it fits directly onto the main board.

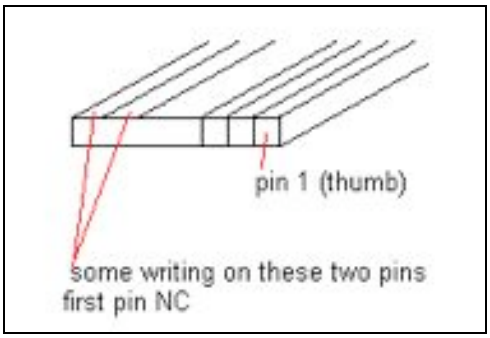
Following is a basic schematic of the wiring between the pins on the Arduino and external device (glove or foot switch). The foot switch is essentially a glove but only with a single "finger" connection and the rest are unconnected. This way the modules can be interchanged without any modifications in the hardware. The software simply needs to specify whether a particular module is used as a glove or foot switch.



There are 13 digital ports on the Arduino. Pins 2 through 13 are available for use, except for pin 7. Pins 0 and 1 are used for serial communication (via Bluetooth) and Pin 7 controls the onboard Bluetooth module.

Touchglove connection

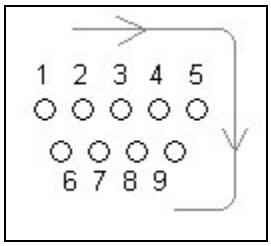
The TouchGlove has 9 pins. One for the glove and 8 for finger connections. Pin 1 on the TouchGlove connector appears as follows:



The following table shows the relationship between the connections on the DB-9 cable, the physical contacts on the gloves, and the glove connector. Assuming all TouchGloves are wired the same way, the only connections to worry about are the DB, ARDUINO and GLOVE CONNECTOR values.

DB	ARDUINO	FINGER	GLOVE CONNECTOR
1:	**5V**	thumb	**G01**
2:	**Pin 10**	pinky tip	**G02**
3:	**Pin 9**	pinky mid	**G03**
4:	**Pin 8**	ring tip	**G04**
5:	**Pin 6**	ring mid	**G05**
9:	**Pin 5**	middle tip	**G06**
8:	**Pin 4**	middle mid	**G07**
7:	**Pin 3**	index tip	**G08**
6:	**Pin 2**	index mid	**G09**

The reason for the DB-9 pin mapping is to allow for easy wrapping around of the ribbon cable, as shown below:



To sum up, the wiring that needs to be matched are:

1. Arduino board to female DB-9 connector (within module)
2. Male DB-9 cable to female Glove Connector (between module and glove)

Please see the photos of the various parts at the end of this document for more details.

Foot switch connection

The foot switch cable has only pins 1 and 6 connected, and acts the same way as a glove with only the "index mid" finger contact connected.

Power Supply

The Arduino BT board contains a built in voltage converter that works between 1.2 and 5.5 volts. There is no over current and reverse polarity protection, so any voltage beyond the correct range may cause permanent damage to the board. The power supply for the current version consists of 2 AA batteries in series, with a

protection diode to prevent reverse polarity (batteries inserted wrong way). From testing the board appears to draw between 70 and 90 mA, the latter when the bluetooth connection is active. Assuming useful capacity of a AA cell to be around 1500mAh, the battery life is expected to be around 16 hours.

Enclosure

The enclosure used was a plastic project box from Lee's Electronics. The box is 10cm x 6cm x 2.6cm and provides a snug fit for the Arduino BT module. For easy access to the batteries the clip is attached to the outside of the box. The female DB-9 connector, power switch and LED indicator are all exposed via holes and notches drilled through the casing. See drawings at the end of this document for more details.

Arduino Environment

Arduino is an open source hardware/software platform that is highly convenient for rapid prototyping of microcontroller projects. The platform includes specifications for a variety of different embedded boards based on the Atmel Amega microprocessors, a simple set of basic C programming instructions, complete IDE, and various libraries for extended functionality for interfacing with other components. Programming of the boards are done through serial/USB. There are various standard off-the-shelf boards with different packaging and extra features, but because of the core compatibility the code can be easily modified to work on different Arduino devices.

Software

Arduino Bluetooth

The Arduino Bluetooth module is very similar in features to the standard Arduino Diecimila/Duemilanove boards. It adds an integrated bluetooth module that functions as a virtual serial port. The board can be programmed via the bluetooth serial connection, and communicate with devices through it as well. Essentially the board can be operated as a regular Arduino board and the only difference is the lack of a physical wire for the serial port.

For more information, including initialization code, please see http://www.arduino.cc/en/Guide/ArduinoBT

Setup

In order to connect the Arduino BT to a computer (for programming, or interfacing), it needs to be paired just like any regular bluetooth device. The pairing process

establishes connection settings that are remembered by both devices so that the devices can initiate the connection automatically in the future. The default password of the Arduino BT is '12345'. During the pairing process the host computer will try and find various services the bluetooth device offers (e.g. file transfer on cellphones, audio for bluetooth headsets etc). The Arduino BT only supports the serial port service.

Once the Arduino BT has been paired, a new serial port will be created on the host computer. This is the port that the Arduino IDE will use to program the board. In windows the serial port is of the form COMx, where x is the index of the port in the system. In Mac OS the serial port will have a name, which is defaulted to the bluetooth name of the device appended with "-BluetoothSerial". This name can be changed in the advanced port options under the bluetooth device settings. Once the serial port is set up, the board is ready to be programmed.

Reset Button during Programming

One important, and (currently) not directly documented note is that when embedded code is programmed (uploaded) to the Arduino board, the board must be reset (During the first few seconds of bootup the Arduino goes into program mode and it is the only time it will accept programming code). This is usually done automatically on the newer Arduino boards, but for some of the older ones, including the Arduino BT, the reset button must be pressed manually at the same time the 'upload' button is pressed in the IDE. Otherwise, the board will not receive the new code and an error will be generated by the IDE.

Initialization code

The bluetooth-specific settings of the Arduino BT is stored inside the embedded bluetooth chip built into the board. These settings define the bluetooth connection speed, device name, security options, etc. They are preset from the factory, but can be changed by running a modified version of the initialization code once. The initialization code puts the bluetooth chip into config mode, and writes the settings into the flash memory. Then, the regular program can be uploaded and executed on the Arduino with the new bluetooth settings. In our situation we modified the bluetooth name in the form of "WBT-XXX" where XXX is the device name that is printed on the outside casing for easy identification.

Arduino2Max

The software consists of two parts. The embedded Arduino code scans the input ports and sends the values through the bluetooth serial. The Max code is responsible for connecting to the Arduino via bluetooth and receive the input data from it.

The version of Arduino2Max used is 0.4 and can be found here: http://www.arduino.cc/playground/Interfacing/MaxMSP

The data acquisition is done by polling. Each time the host wants to read from the Arduino, an ascii "r" is sent through the serial. The Arduino code sits in a loop that waits for serial input. As soon as the "r" is received from the serial port, the Arduino then scans all its input ports, and then outputs them through the serial.

In Max/MSP, the object responsible for reading from the Arduino is hooked up to a metro and sends out the "r" at specified intervals (in our case 10 ms) and then reads the serial data sent back.

The main thing to note with this system is that a.) it is polled, requiring a command from the host device before reading and sending the pin values and b.) it is currently read-only (does not control any output pins on the Arduino).

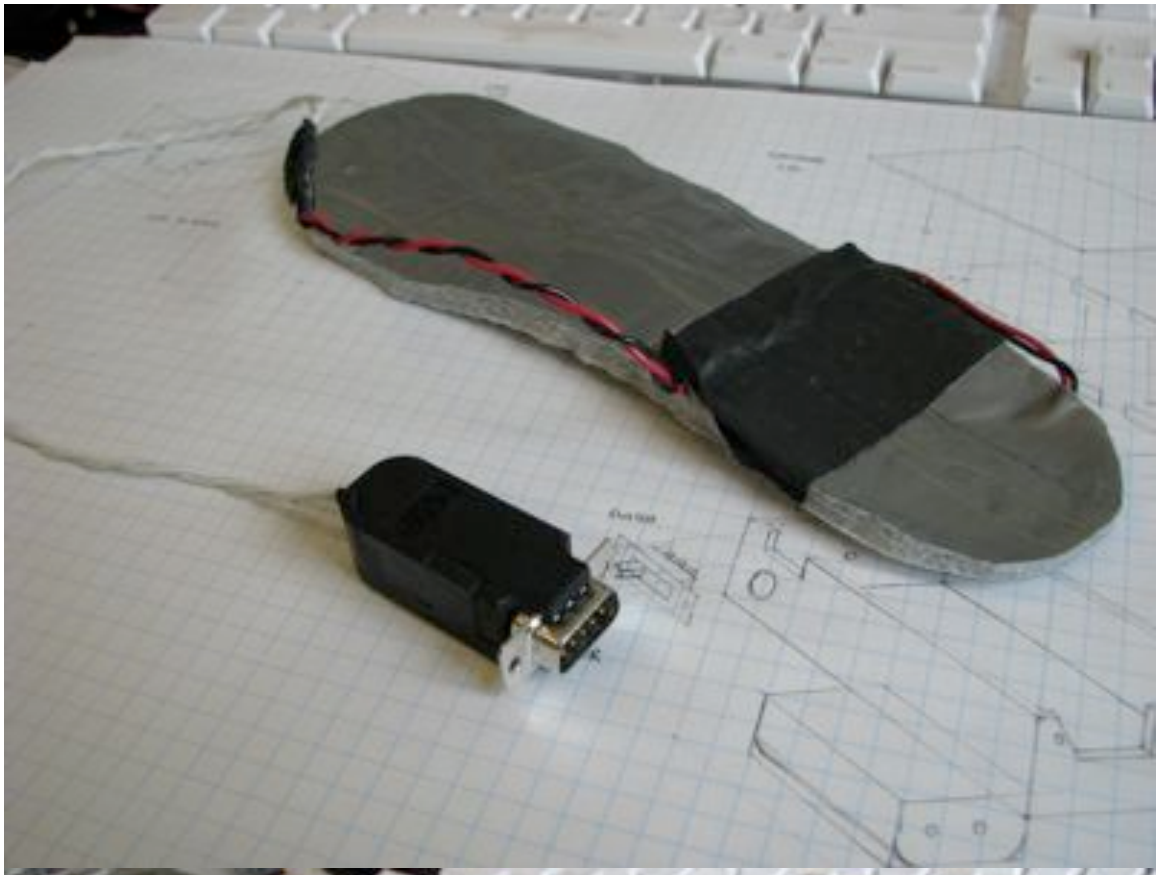
Future Development

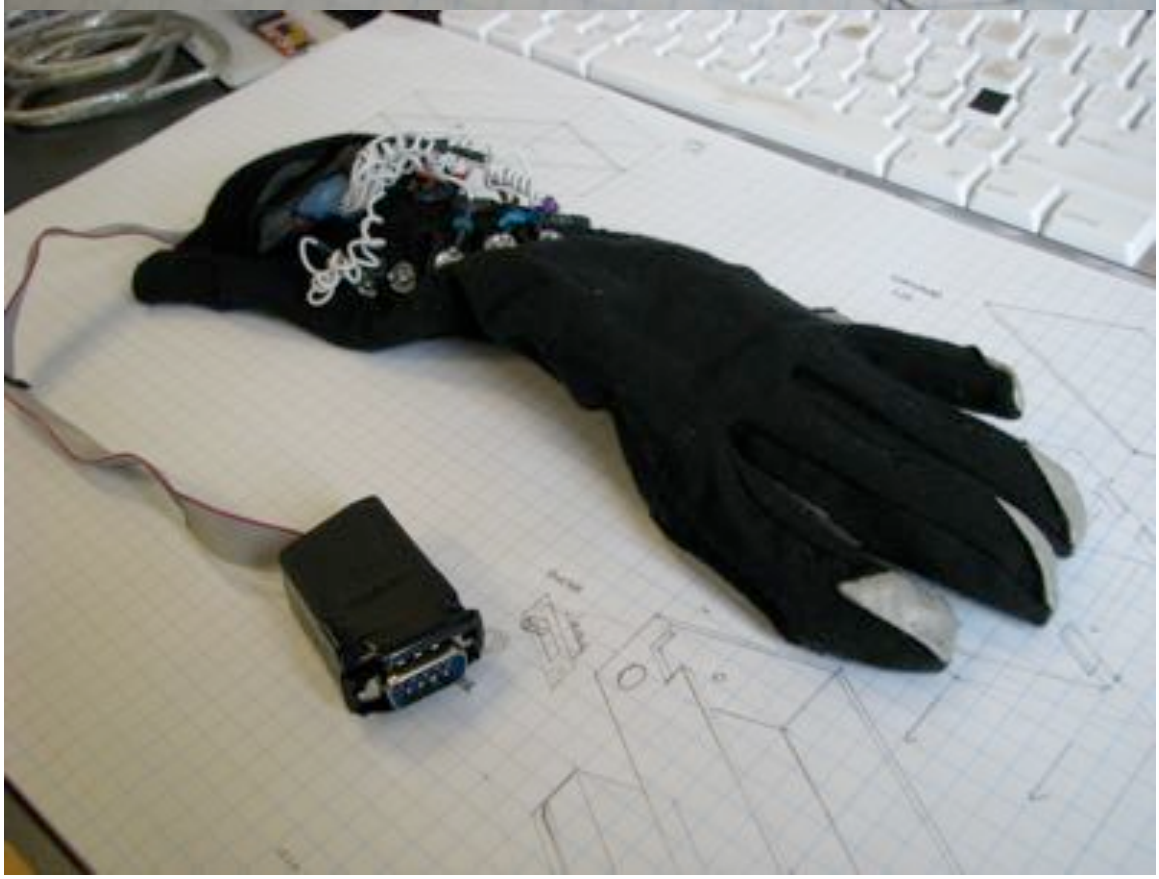
The current foot switch provides on-off control of the volume output. A continuous range of volumes may be desirable in the future. In order to achieve this an analog sensor will be built into the foot switch, and the analog inputs of the Arduino can be used without any additions to the current software.

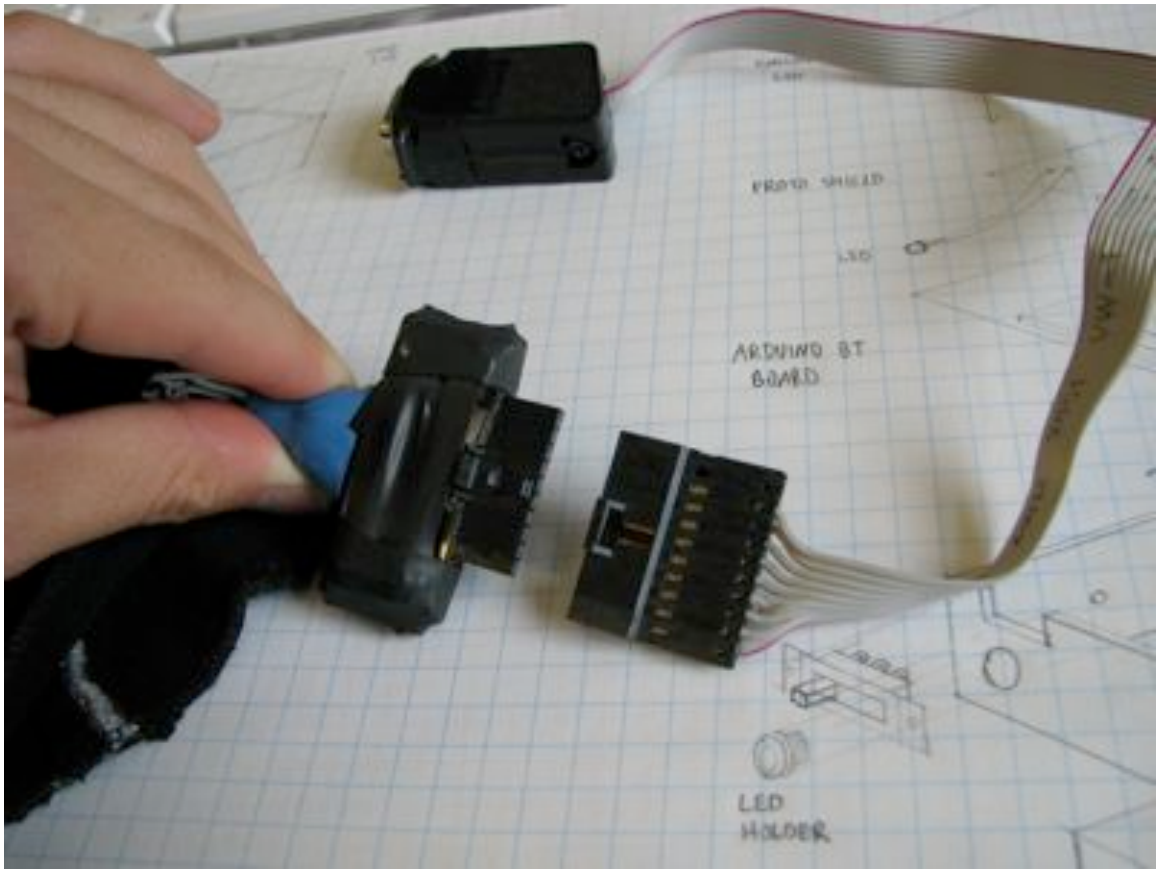
One of the motivations of the current design is a reduction in size compared to the previous version. However, using the more compact Arduino Mini it is possible to reduce the size even further. In terms of power, a Li-ion polymer cell would also reduce the bulk of the system while maintaining acceptable battery life.

Pictures and Diagrams

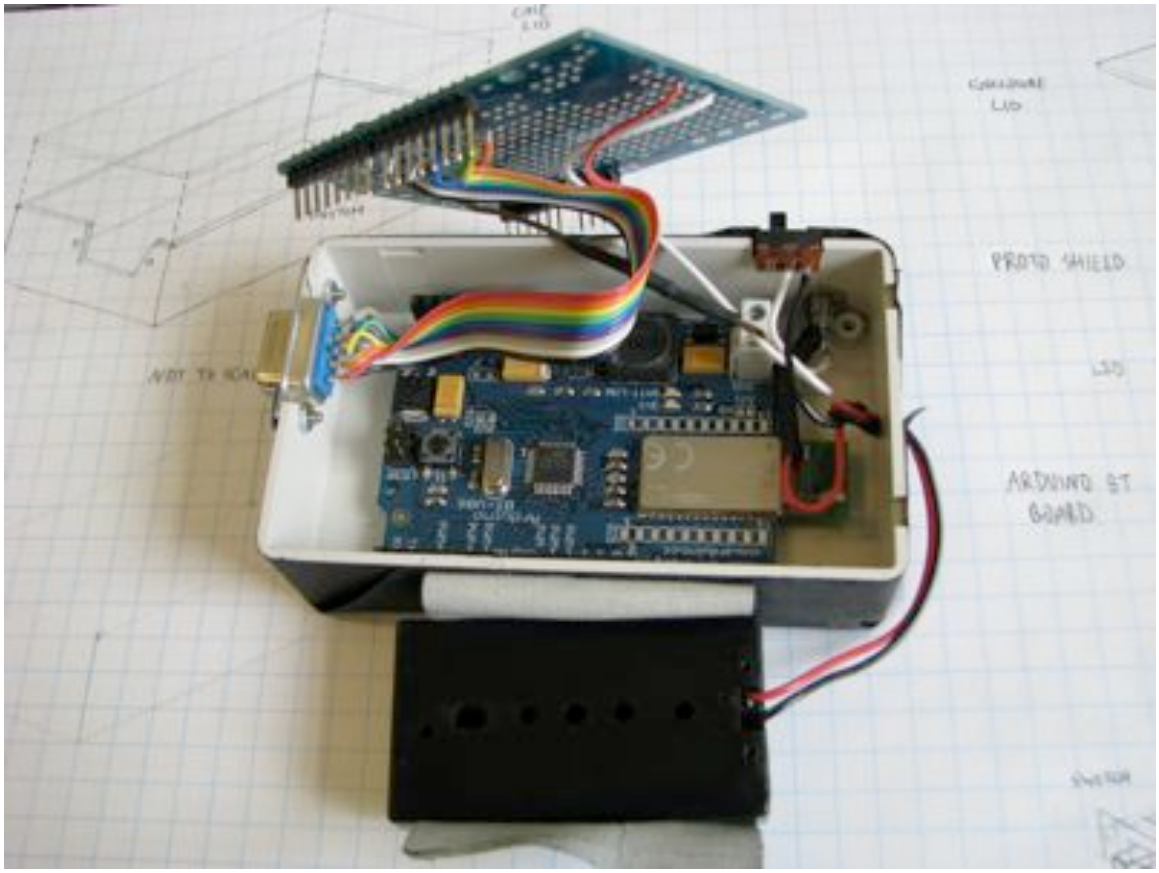
The completed WBT module, covered in black tape for costuming purposes.







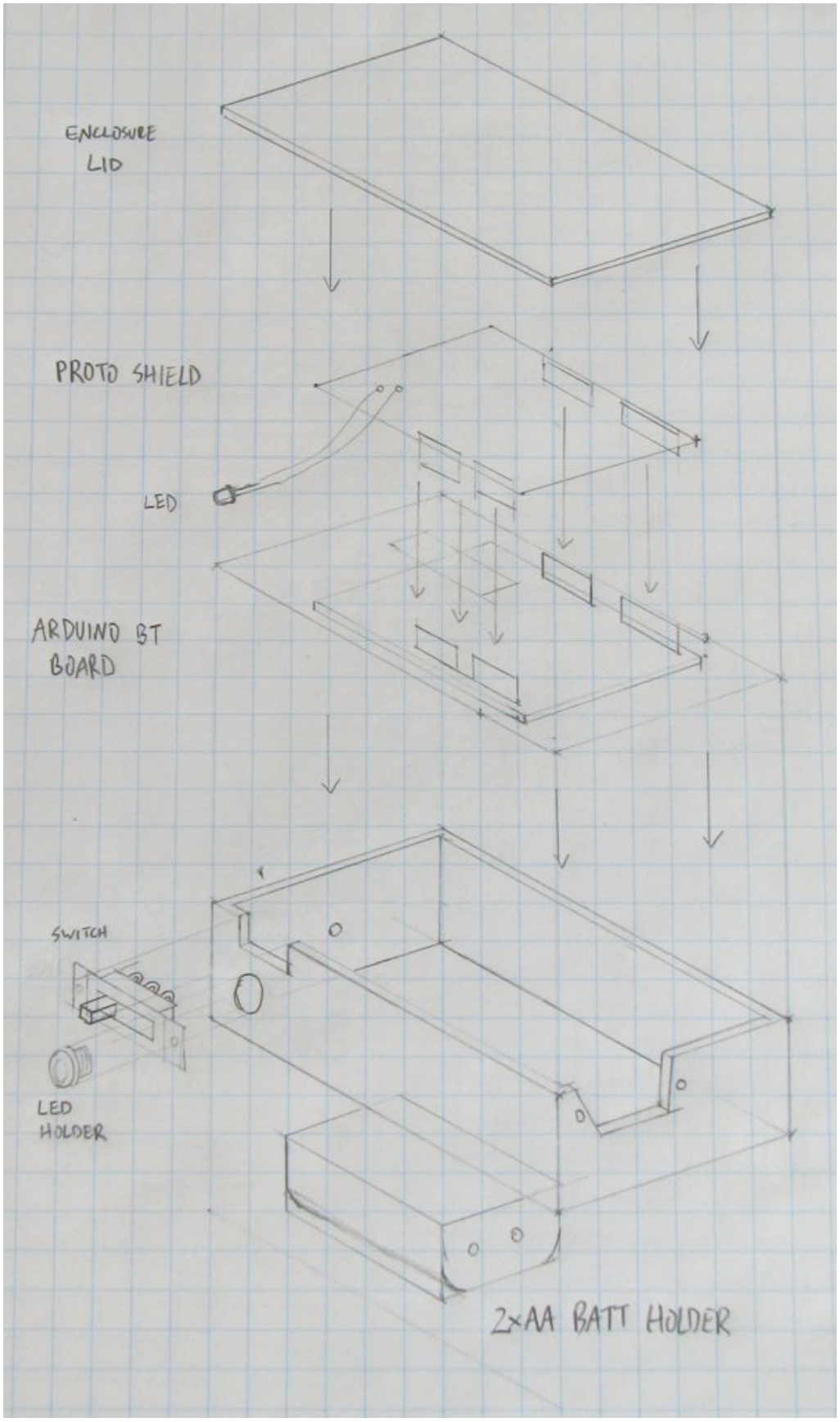
In the above image, the red cable corresponds to Pin 1 the thumb contact of the glove.



Inside the case

Mechanical Drawings

See following page.


ENCLOSURE LID
PROTO SHIELD
LED
ARDUINO BT BOARD
SWITCH
LED HOLDER
2×AA BATT HOLDER