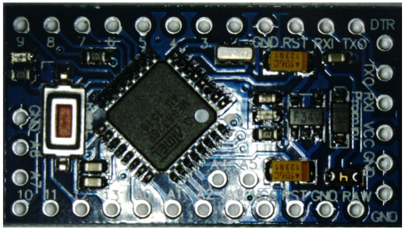
# THE 3.3$ ALTIMETER

version: 0.3

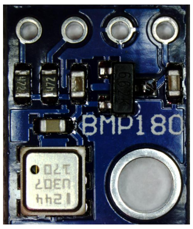
## 1 NEEDED MATERIAL

To build the altimeter you need:

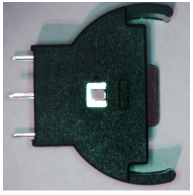
- Arduino pro Mini 8Mhz 3.3V with atmega328 (it is important to have the 8Mhz 3.3V version with the atmega328 and that the A4 and A5 pins are break out)



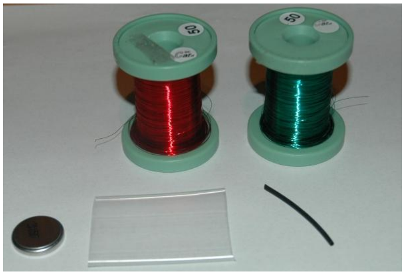
- BMP180 on breakout-board (the smaller boards with 4 pins are ideal)



- cr2032 battery holder



- cr2032 battery
- solderable magnet wire (different colors are useful)
- shrink tube (width 36mm and a small piece 1mm)



If you look on aliexpress you can find the arduino around 1.7 Dollar and the BMP180 around 1 Dollar.

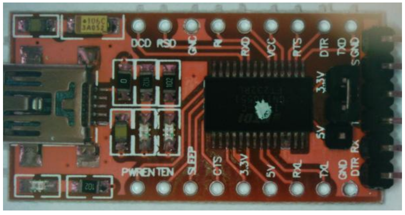
You need also the following tools:

- Soldering iron (high enough temperature to burn away the isolation of the magnet wire) + solder

- wire cutter
- cutter

- hot glue gun + hot glue

- hot air gun or lighter to shrink shrink tube
- optional: multimeter for testing connections
- 3.3V FTDI USB to UART (TTL) arduino



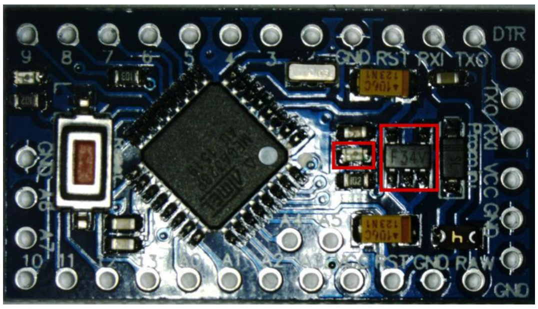
It must be a 3.3V programmer, a 5V programmer will destroy the board (best use one where you can select the voltage with a jumper)! It is also useful if it has a reset pin DTR for the arduino, the ideal pin out is DTR TXo RXI VCC GND GND, because then you can directly plug it into the arduino. I would avoid the cheapest USB to TTL converters you can find on ebay for 1 Dollar they tend to have bad ftdi chips. Look for one which has a jumper to select the voltage and is red. (search on ebay or aliexpress for 'FTDI USB to TTL' and you will find some for around 2.4 Dollars) If you want better quality look at sparkfun or dfrobot.

## 2 BUILDING THE HARDWARE

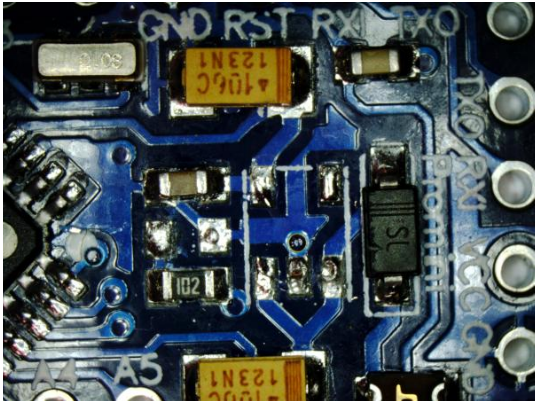
### 2.1 *Power saving modifications (optional)*

This modification is only needed to save power, it reduces the power consumption on the pad and in the air from >20 mA to 3.5 mA and after apogee to 18 $\mu$A. If you do not care about power consumption, for example because you use a LiPo instead of the cr2032 these modifications are not necessary. The modifications remove the reverse polarity protection from the arduino and allow it to run below 3.3 V.

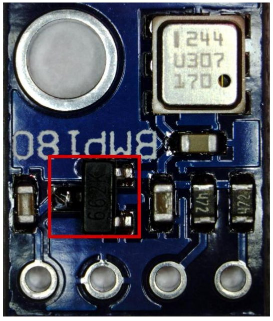
On the arduino we have to remove the regulator and the power led (the position can be different for different boards; on the sparkfun boards there is a jumper J1 to do the manipulation). This is the most power saving manipulation:



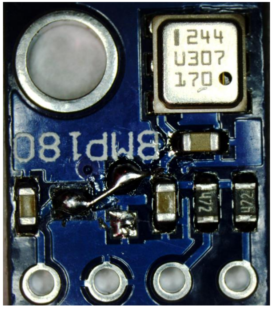
For the cheapest boards, cut the legs of the regulator and the led with the cutter until you can remove them. Then take the soldering iron and remove the rest of the legs on the soldering pads. Afterwards it should look like this:



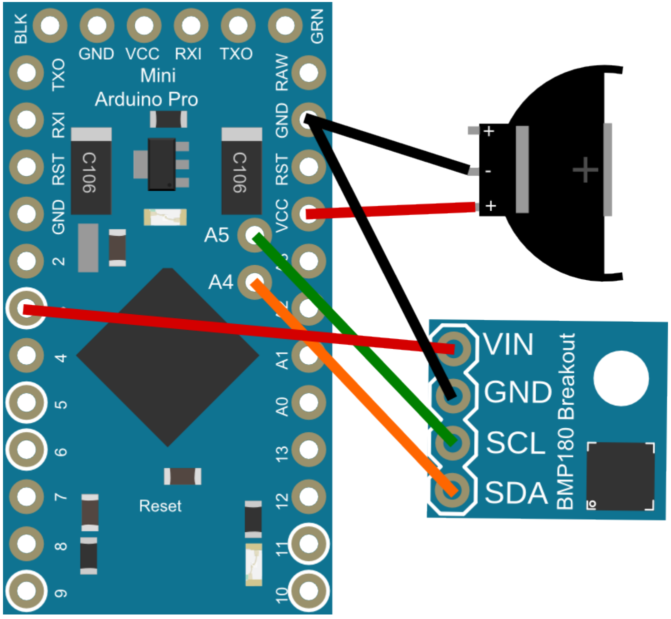
On the bmp180 board we also have to remove the regulator (this saves very few power but it allows the bmp180 to run on lower voltages):



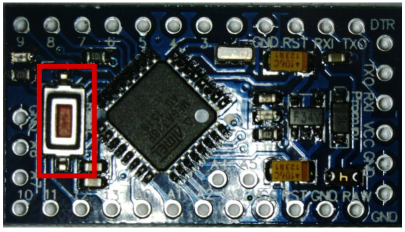
To use the soldering pins afterwards for VIN, we have to short the VIN pin with the former VOUT pin of the regulator, by a small wire.
Afterwards it should look like this:



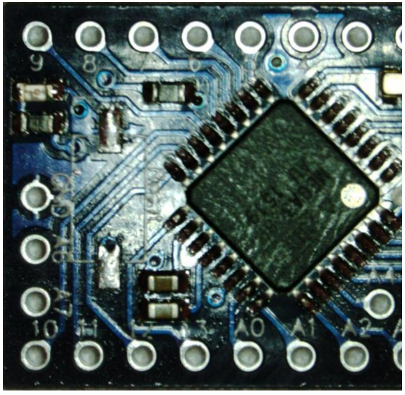
The basic wiring for the altimeter looks like this:



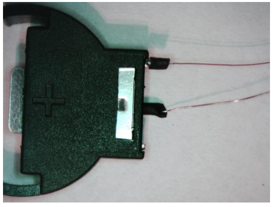
To avoid triggering the reset button of the arduino during flight, we remove it (it is optional and should only be done if you have a programmer which resets the arduino DTR pin):



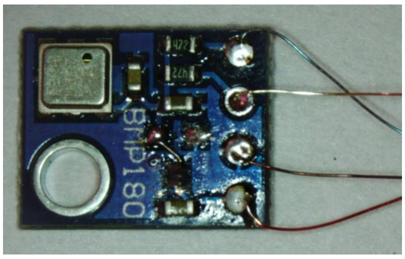
Cut the legs of the button with the cutter until you can remove it. To remove the rest of the legs from the pads use the soldering iron. It should look like this:



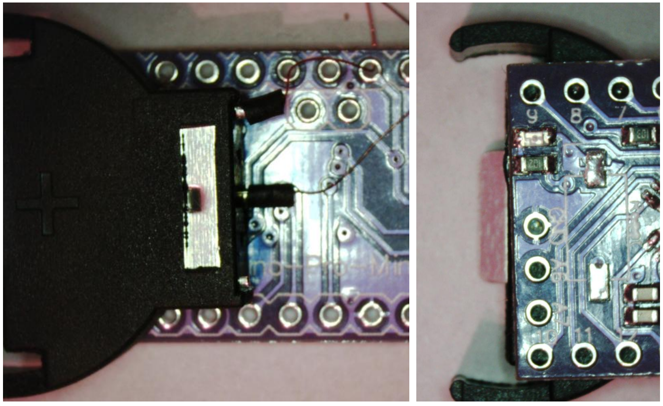
Now we start with the battery connector. Cut away one of the positive pins, solder magnet wire (5 cm are handy) to the two other pins and isolate the soldering point with shrink tube.



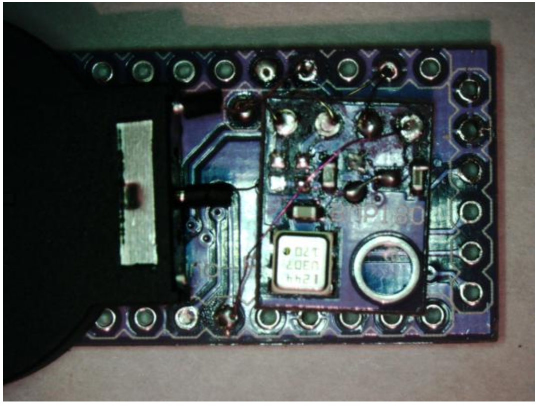
Next solder four wires to the bmp180 breakout:



In the next step you have to glue the battery holder with hot glue to the back of the arduino (on the side where the switch was) so that the main part of the holder is flush with the board and the positive terminal is facing away from the board.



Now place the bmp180 to the back of the arduino but do not glue it yet:



Fiddle the wires trough the pin holes of the arduino.
bmp180:

BMP VIN to arduino digital pin 3

BMP GND to arduino GND on the long side (not on the uart)

BMP SDA to arduino A4

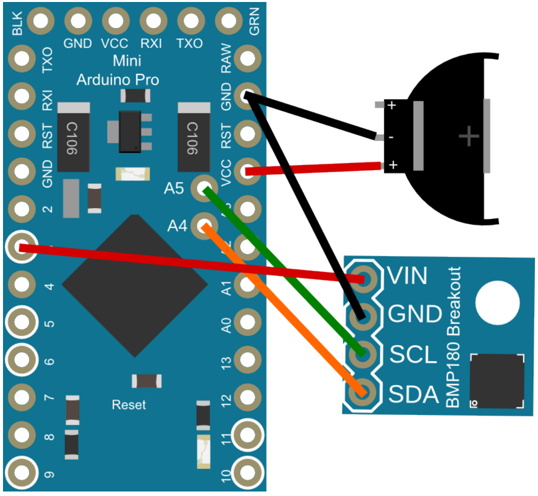
BMP SCL to arduino A5

battery:

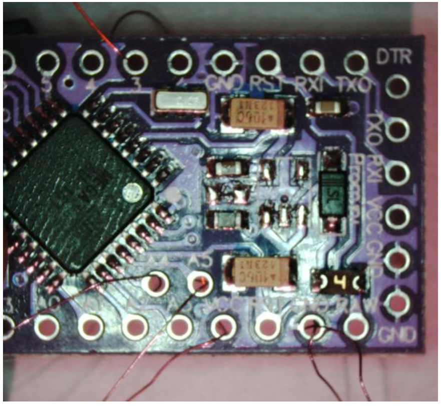
Battery + to arduino VCC (3.3V) on the long side (not on the uart)

Battery - to arduino GND on the long side (not on the uart)

The wiring should match the known sketch:

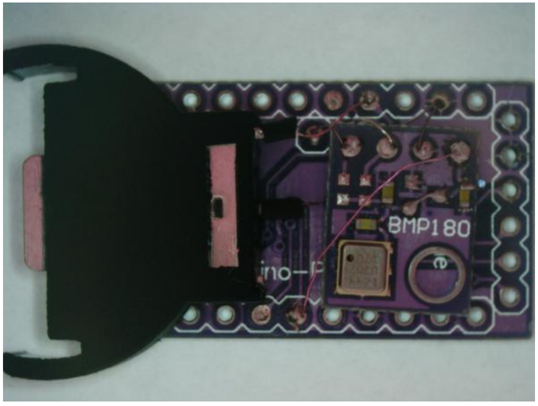


Connecting the BMP VIN to a digital pin on the arduino allows the arduino to turn the bmp180 on and off to save power.
From the front it should now look like this:



Now glue the bmp180 in place with hot glue, use enough glue so there is some isolating distance to the back of the arduino board and leave the UART pins on the short side free.

Now pull the wires as short as reasonable and solder them to the arduino pins, solder long enough to burn the isolation of the wires. If you have a multimeter you can now check the connections.
So the end result should look like this:



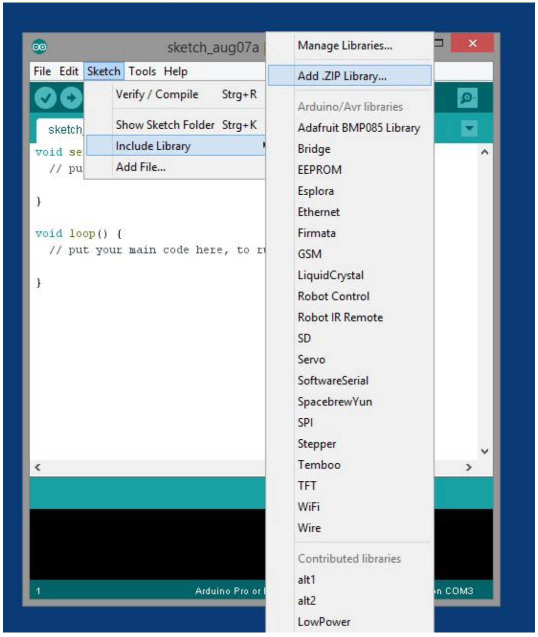
Clean all soldering joints from flux.

After programming you should add shrink tube to protect the board, leave a hole for the pressure sensor.

## 3 PROGRAMMING

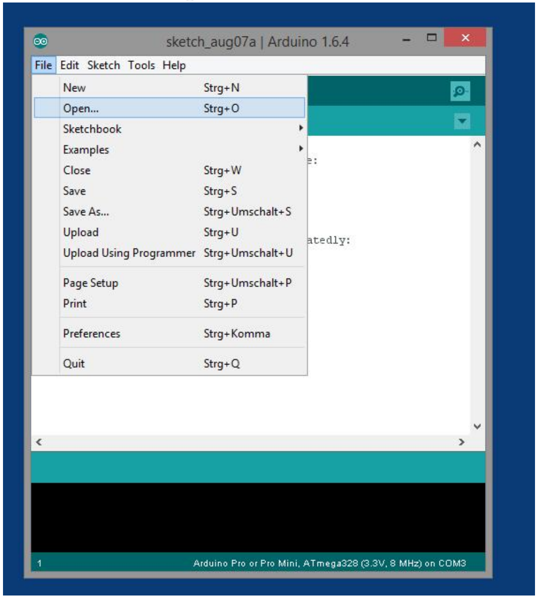
The altimeter is based on an arduino and can be programmed with the arduino IDE. Download and install the IDE:
`https://www.arduino.cc/en/main/software`
Download also the program for the altimeter and and the libary for the bmp180.

To install the libary go on 'Sketch / Include Libary / Add .ZIP Libary ...' and select 'bmp180.zip'.

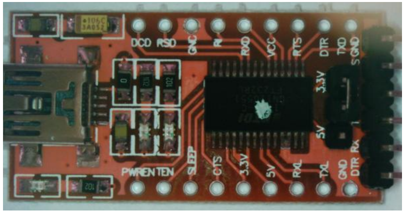


```
void se
  // pu

}

void loop() {
  // put your main code here, to r

}
```

For problems with the installation refer to:
`https://www.arduino.cc/en/pmwiki.php?n=Guide/Libraries`
Go on 'File / Open...' and load the 'altimeter.ino'.

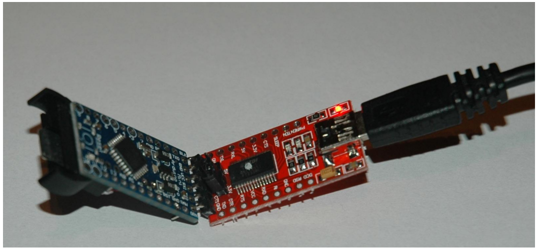

sketch_aug07a | Arduino 1.6.4
File Edit Sketch Tools Help
New Strg+N
Open... Strg+O
Sketchbook
Examples
Close Strg+W
Save Strg+S
Save As... Strg+Umschalt+S
Upload Strg+U
Upload Using Programmer Strg+Umschalt+U
Page Setup Strg+Umschalt+P
Print Strg+P
Preferences Strg+Komma
Quit Strg+Q
e:
atedly:
1
Arduino Pro or Pro Mini, ATmega328 (3.3V, 8 MHz) on COM3

Connect the FTDI USB programmer ot the USB Port

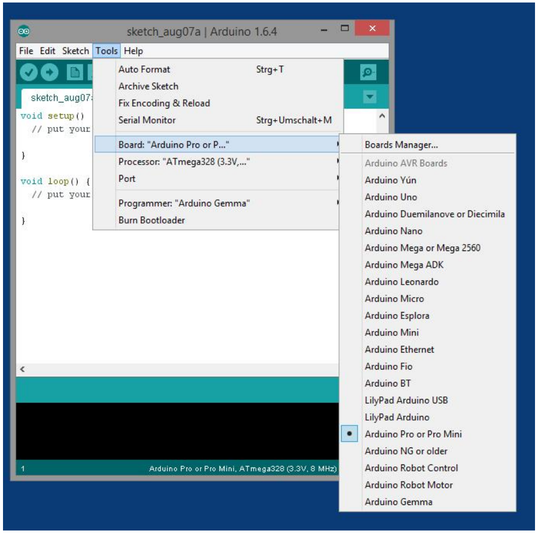


It must be a 3.3V programmer, a 5V programmer will destroy the board (best use one where you can select the voltage with a jumper)! The pin layout of

the programmer should match the one of the arduino pro mini:
DTR TXo RXI VCC DNS GND
sometimes TXo and RXI are labeled as on the arduino and sometimes they are reversed (describing their function on the programmer) The programmer should have pin headers soldered on it, simply insert them im in matching holes on the arduino and tilt the headers so they have a connection to the contacts on the arduino.



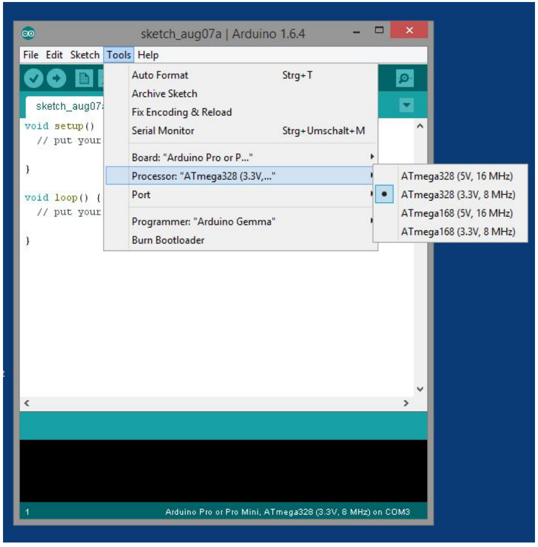
In the arduino IDE select under 'Tools' as 'board' the 'Arduino Pro or Pro mini'.



```
void setup()
  // put your

}

void loop() {
  // put your

}
```

1 Arduino Pro or Pro Mini, ATmega328 (3.3V, 8 MHz)

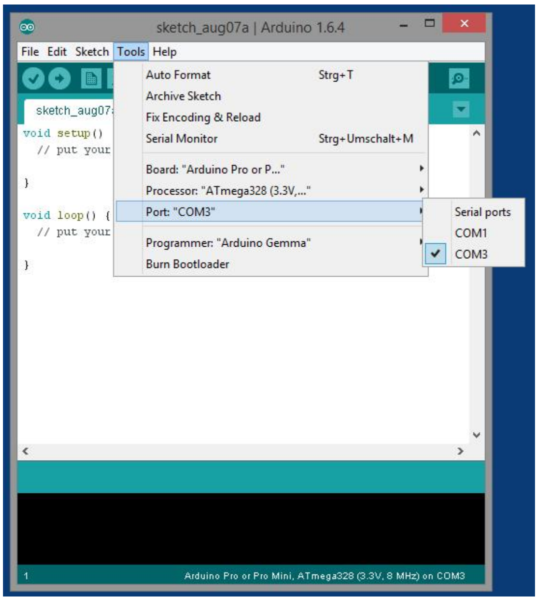
Select as 'Processor' the 'Atmega328 (3.3V 8Mhz)'.


sketch_aug07a | Arduino 1.6.4
File Edit Sketch Tools Help
Auto Format Strg+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Strg+Umschalt+M
Board: "Arduino Pro or P..."
Processor: "ATmega328 (3.3V,..."
Port
Programmer: "Arduino Gemma"
Burn Bootloader
ATmega328 (5V, 16 MHz)
● ATmega328 (3.3V, 8 MHz)
ATmega168 (5V, 16 MHz)
ATmega168 (3.3V, 8 MHz)
sketch_aug07

```
void setup() {
  // put your

}

void loop() {
  // put your

}
```

1 Arduino Pro or Pro Mini, ATmega328 (3.3V, 8 MHz) on COM3

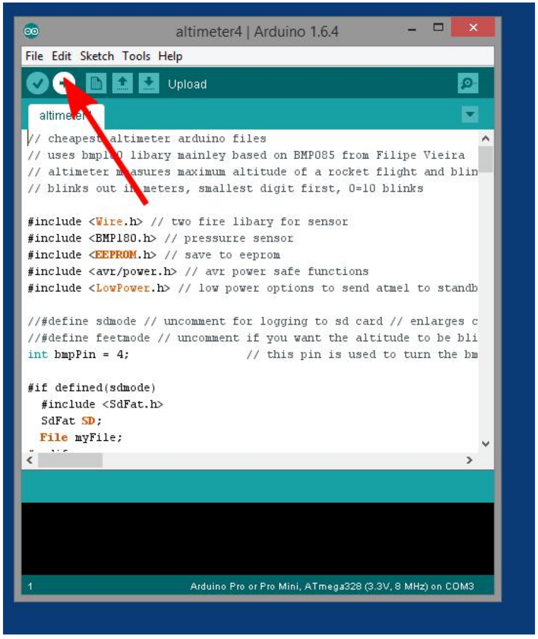
And under 'Port' select the serial port of your arduino programmer.


sketch_aug07a | Arduino 1.6.4
File Edit Sketch Tools Help
sketch_aug07
Auto Format Strg+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Strg+Umschalt+M
Board: "Arduino Pro or P..."
Processor: "ATmega328 (3.3V,..."
Port: "COM3"
Programmer: "Arduino Gemma"
Burn Bootloader
Serial ports
COM1
COM3

```
void setup()
  // put your

}

void loop() {
  // put your

}
```

```
1                Arduino Pro or Pro Mini, ATmega328 (3.3V, 8 MHz) on COM3
```

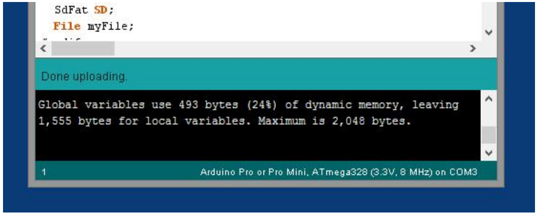
If you are not sure which port belongs to your programmer, disconnect the programmer. The port which disappears is the one of your programmer. Upload the sketch to the board, by clicking the right arrow (second symbol from the left).



```
// cheapest altimeter arduino files
// uses bmp180 libary mainley based on BMP085 from Filipe Vieira
// altimeter measures maximum altitude of a rocket flight and blin
// blinks out in meters, smallest digit first, 0=10 blinks

#include <Wire.h> // two fire libary for sensor
#include <BMP180.h> // pressurre sensor
#include <EEPROM.h> // save to eeprom
#include <avr/power.h> // avr power safe functions
#include <LowPower.h> // low power options to send atmel to standb

//#define sdmode // uncomment for logging to sd card // enlarges c
//#define feetmode // uncomment if you want the altitude to be bli
int bmpPin = 4;                  // this pin is used to turn the bm

#if defined(sdmode)
  #include <SdFat.h>
  SdFat SD;
  File myFile;
```

```
1                Arduino Pro or Pro Mini, ATmega328 (3.3V, 8 MHz) on COM3
```

Check that the status display shows 'Done uploading'.



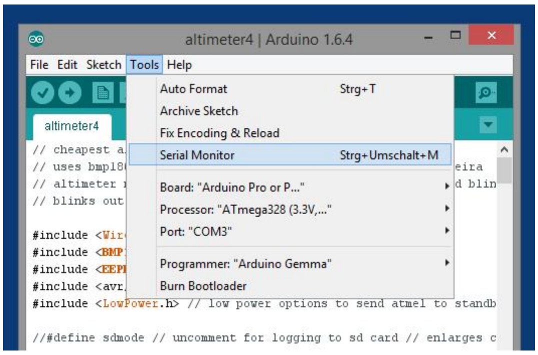
```
  SdFat SD;
  File myFile;
```

```
Done uploading.
```

```
Global variables use 493 bytes (24%) of dynamic memory, leaving
1,555 bytes for local variables. Maximum is 2,048 bytes.
```

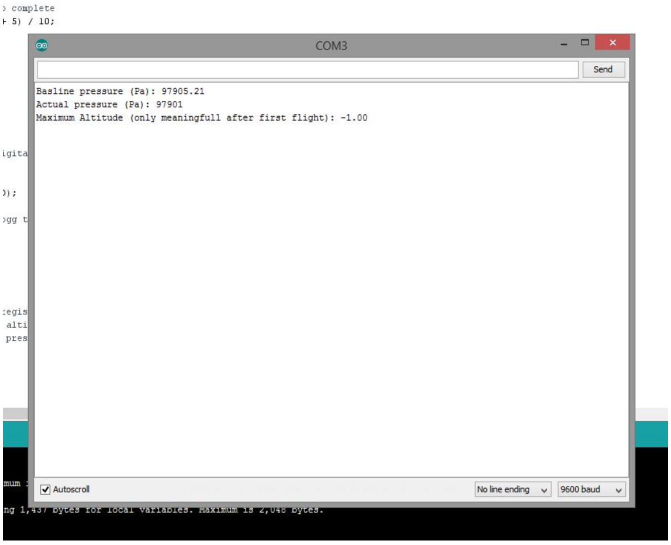
```
1                Arduino Pro or Pro Mini, ATmega328 (3.3V, 8 MHz) on COM3
```

To check if your altimeter is working properly, hook it up again on the FTDI programmer. Select the correckt com port and start 'Tools / Serial Monitor'


altimeter4 | Arduino 1.6.4
File Edit Sketch Tools Help
altimeter4
Auto Format Strg+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Strg+Umschalt+M
Board: "Arduino Pro or P..."
Processor: "ATmega328 (3.3V,..."
Port: "COM3"
Programmer: "Arduino Gemma"
Burn Bootloader

```
// cheapest a                                                    eira
// uses bmp18                                                    d blin
// altimeter i
// blinks out

#include <Wir
#include <BMP
#include <EEPI
#include <avr.
#include <LowPower.h> // low power options to send atmel to standb

//#define sdmode // uncomment for logging to sd card // enlarges c
```

Select 9600 baud on the right bottom (even if it is already selected this will restart the altimeter). After a few seconds the altimeter will report the base pressure and an actual pressure reading, check that it is meaningful. It is in Pascal, it should be close to atomospheric pressure 101300 Pa.



```
Basline pressure (Pa): 97905.21
Actual pressure (Pa): 97901
Maximum Altitude (only meaningfull after first flight): -1.00
```

It will also output the maximum altitude from the last flight. It is only meaningful if there was a flight already.

The altimeter is now ready to use.

## 4 USAGE

Insert a cr2032 battery in the battery holder, make sure the polarity is correct, reverse polarity will likely destroy the altimeter. The altimeter will start to blink out the last recoded altitude in meters, if present. Starting with the lowest digest first, 10 blinks stand for 0, then after a 2 second delay the next digit and so on. If the battery voltage is too low the altimeter will start to blink very fast. The altimeter has a 30 second delay, to allow installation into the rocket. After this 30 seconds it will collect some pressure samples to obtain the pad altitude. It will blink very short every five seconds to signal that it is turned on. The altimeter samples the pressure with 64 Hz and calculates altitudes and applies a kalman filter. After the launch was detected (by reaching a specific height normally 20 m) it will determine the maximum altitude continuously. It will check for 100 readings of decreasing pressure, which is considered to be after apogee. It will then save the maximum altitude in the eeprom to signal it after landing. Then the altimeter turns off the sensor and sends the processor to sleep so the power consumption will be 18 $\mu$, which allows the altimeter to stay in this state for a year before the battery is empty. When you have recovered the rocket simply remove the battery and reinstall it (check polarity), the altimeter will start blinking out the maximum altitude reached.

Remove the battery to turn the altimeter off.

If eeprom logging is activated, the altimeter will log the altitude continuously to the eeprom, with a rate of 8Hz before apogee and 4Hz after apogee, 1022 byte are available for this and each number will consume 2 bytes.