



Dual rocket altimeter using the ATmega 328 microcontroller

The “AltiDuo”

Version	date	Author	Comments
1.0	29/12/2012	Boris du Reau Boris.dureau@neuf.fr	Initial Version
1.1	17/02/2013	Boris du Reau	Updated schema
1.2	03/04/2013	Boris du Reau	Updated info regarding the jumpers for presetting the altitude

Type of rocket

Micro-max	Model rocket	Mid power	High power
no	yes	yes	yes

Category

Construction Technic	Ground Support	Electronic	Other
		X	X



Contents

Goal	3
Why build your own altimeter?	3
How does the altimeter works?	3
Altimeter specifications.....	3
Electronic Schema:.....	5
Board and components implementation	6
Components List:	6
The board.....	7
Pressure sensor:	7
Presetting the main deployment altitude	8
Connecting to the USB port:	8
The PL2303HX model	9
The CP2102 model.....	9
Altimeter programming:.....	9
Development Environment:	10
Functional diagram:.....	13
Altimeter code:	14



Goal

The primary goal of the following document is to describe the build, the programming and the use of a low cost dual ejection altimeter based on the ATmega 328 microcontroller well known from the Arduino board users.

The final goal is to be able to offer an altimeter kit for a reasonable price (less than 20 Euros) that is programmable by the user.

This document assumes that you have used and you are familiar with a dual recovery altimeter in a rocket.

Why build your own altimeter?

- Most of the altimeters are coming from the US, even if sometime their price is more than reasonable, the shipping cost can be more expensive than the altimeter itself.
- For the pleasure of doing it yourself.
- To do exactly what you need
- So that you are able to modify your program and to have full control of your electronic.

Before your start

Remember that it is a kit and that you can modify the program and behaviour of your altimeter.

The country where you live might not even allow the use of such device. You have to assume total legal responsibility for any damages or claims including personal injury that results from the use of this device. I shall not be responsible for the above. If you disagree with that, please do not build it or use it.

How does the altimeter works?

A pressure sensor measure the ground altitude to define a reference, then detect the lift off, and when the apogee is reached an ejection charge is fired to extract the drogue and a second charge is fired at a predefined altitude (usually 100 meter) before landing to deploy the main parachute.

The altimeter does a continuity test of the charges to make sure that they are good. Apogee and main altitude are “beeped” after the main has been fired.

Altimeter specifications

Dimensions: 90x32mm

Weight: approximately 30 grams

Power supply: from 6 to 10Volts

2 high power transistors pyro outputs

Separate Continuity test for each output.

Output on 2 pin terminal blocs



ATmega 328 dual altimeter – AltiDuo

Diode protection against reverse polarity.

Programable using the USB port and the C/C++ language with the Arduino development environment.

Apogee detection

4 preset altitude selectable for the main using jumpers

According the pressure sensor (ie: the BMP085) datasheet, it should be able to measure up to 9000m.

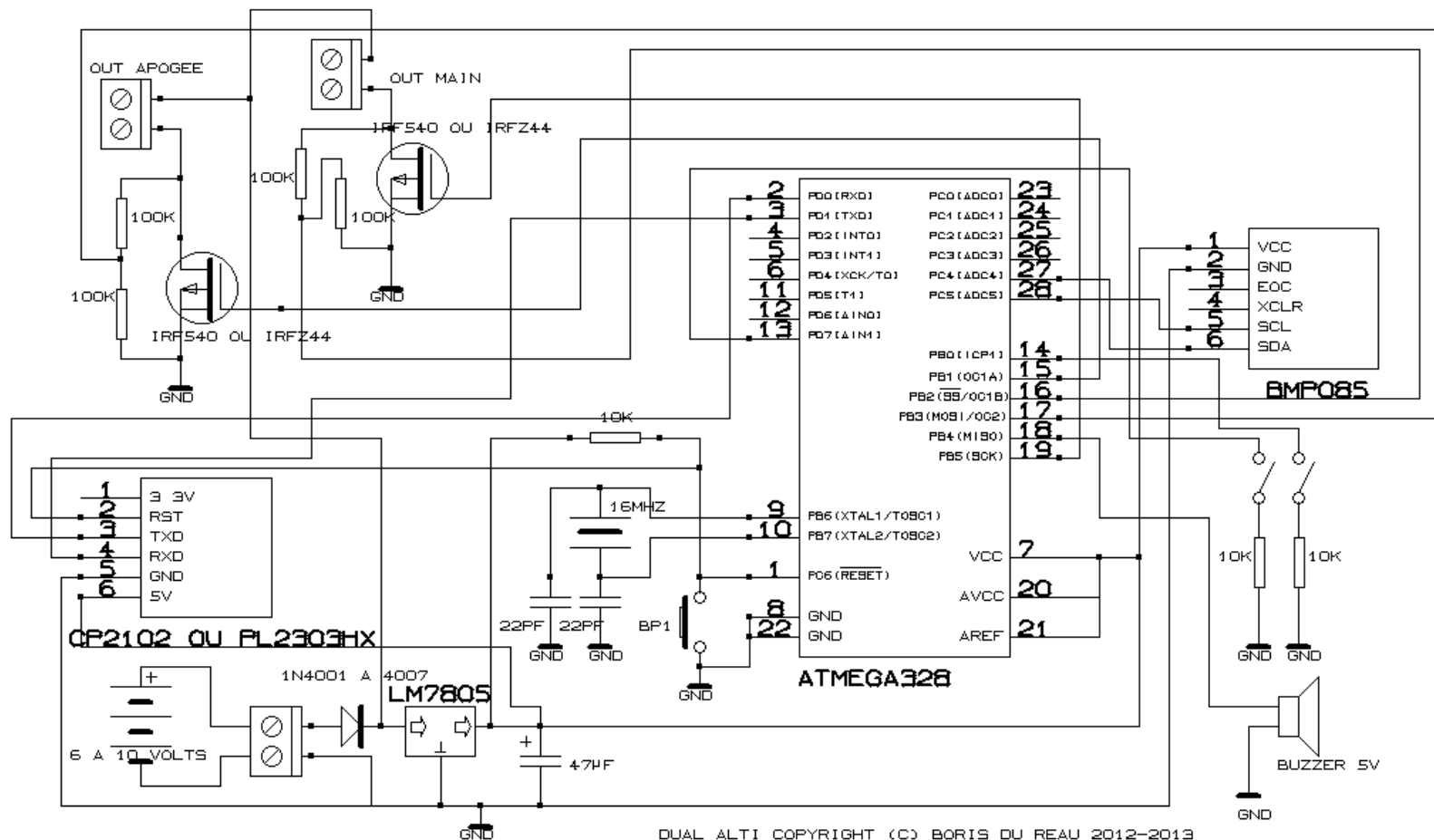
You can find the datasheet here

<http://www.daedalus.ei.tum.de/attachments/article/60/BST-BMP085-DS000-05%20Drucksensor.pdf>



ATmega 328 dual altimeter – AltiDuo

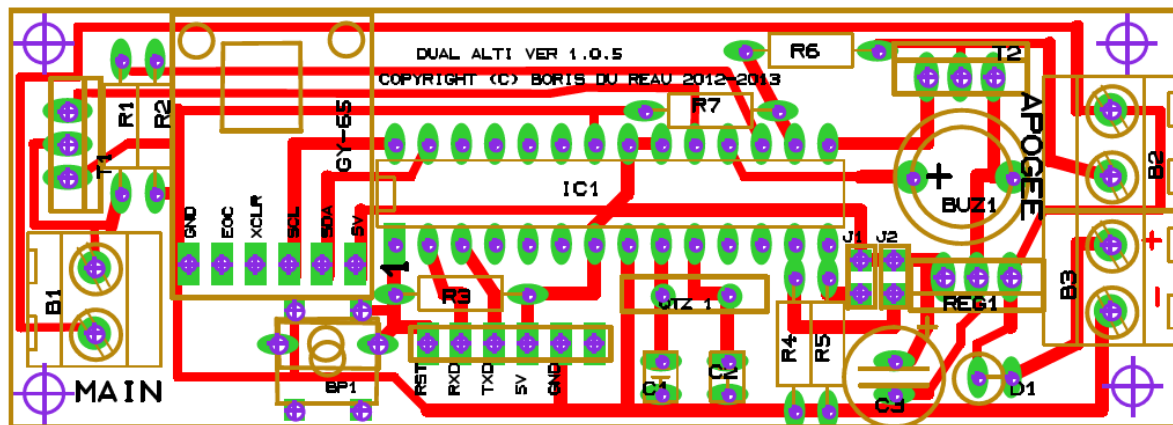
Electronic Schema:





ATmega 328 dual altimeter – AltiDuo

Board and components implementation:



Components List:

R1,R2	100Kohm to 150Kohm
R3,R4,R5	10Kohm to 15kohm
R6,R7	100Kohm to 150Kohm
C1, C2	20 or 22pf
C3	47µf (vertical)
T1, T2	Transistor (IRF540 or IRFZ44)
Reg1	7805 (TO220)
B1,B2,B3	2 pins Terminal block pitch 5,08
Q1	16Mhz cristal
Buzz1	5 volts active buzzer
IC1	Atmega 328 + 28 pins socket + Arduino bootloader
D1	1N4001 to 1N4007
Sensor	BMP085

Note that for the resistors when for R1, R2, R6, R7: you can choose a value between 100 and 150K, but all resistors must be identical. Same goes for the other resistors.

For the diode it can be anything from 1N4001 to 7; the diode will reduce the output current; note that if you do not care about protecting the altimeter it can be replaced by a wire.



ATmega 328 dual altimeter – AltiDuo

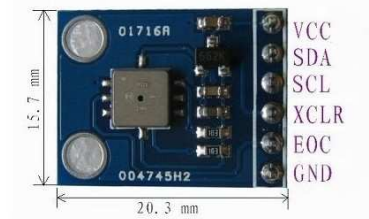
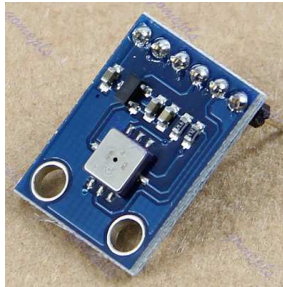
The board

The smallest board that I have made has the following dimension 30x90 mm



Pressure sensor:

The pressure sensor used is a BMP085. It has been soldered on a small board. Caution several model exists...you need to choose a board that can be powered with 5V. The model that I have chosen after trying several one is the GY-65 :



Those mini board pin layout are not identical between brands, you will have to adapt the board if you use a different one.

Be careful not to use a 3V3 board that will not work with 5V (it will be damaged).

Do not use the following one

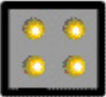


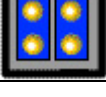




Presetting the main deployment altitude

The main deployment altitude can be preset using a couple of jumpers. With version 1.0 and 1.1 of the program you can choose from 4 different altitudes. They are 50, 100, 150 or 200 meters.

Here is a table with all possible options:

50m	
100m	
150m	
200m	

Connecting to the USB port:

It is possible to use several type of board but the order of the pins being different from one model to the other you will need to choose a model with a cable so that you can connect it to the altimeter board (crossing wires if needed). If possible choose one protected by some shrink tube. If this is not the case you will need to put some, it is mandatory because you will need to touch the card with your finger whenever you plug it to your USB port.

Cable

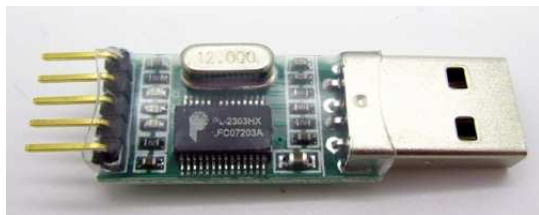


I have used the following models

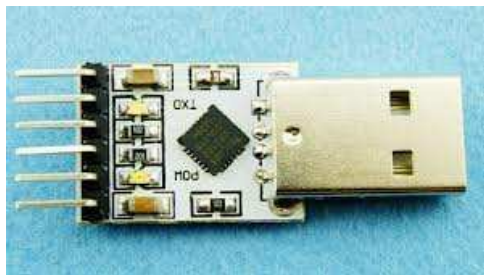


ATmega 328 dual altimeter – AltiDuo

The PL2303HX model



The CP2102 model



Altimeter programming:

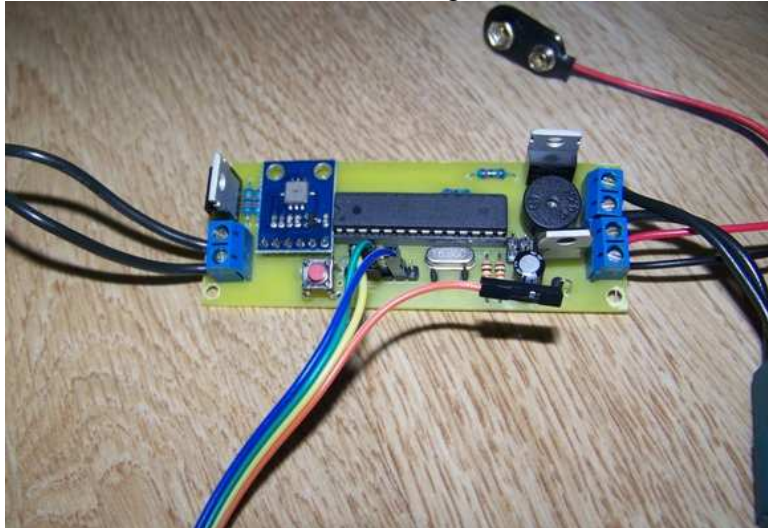
The USB adapter will be connected to your altimeter cable using a cable. Make sure that you cable it properly because the connections are not protected also, unplug the battery when you plug it to the PC or unplug the 5V power supply of the board.

Note that the continuity test does not work properly when the battery is unplugged.





ATmega 328 dual altimeter – AltiDuo



Development Environment:

In order to be able to upload the software on the altimeter you need to go to the Arduino web site.

<http://www.arduino.cc/>

Then you need to download the software

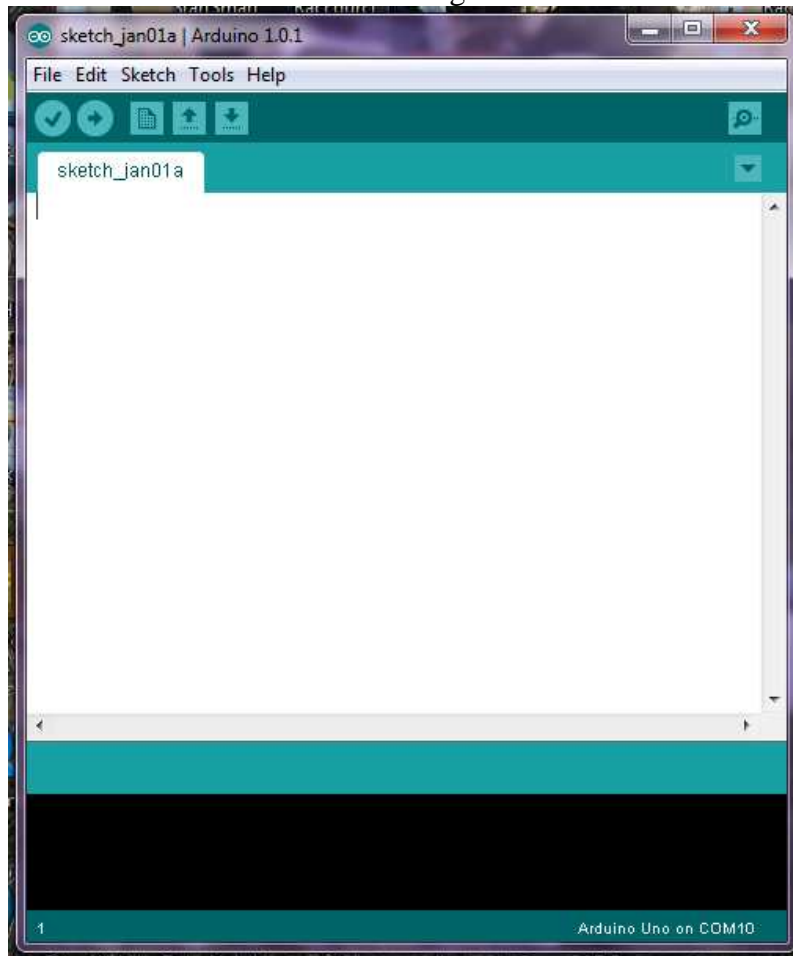
<http://arduino.cc/en/Main/Software>

Install the development environment and launch the application by clicking the icon below

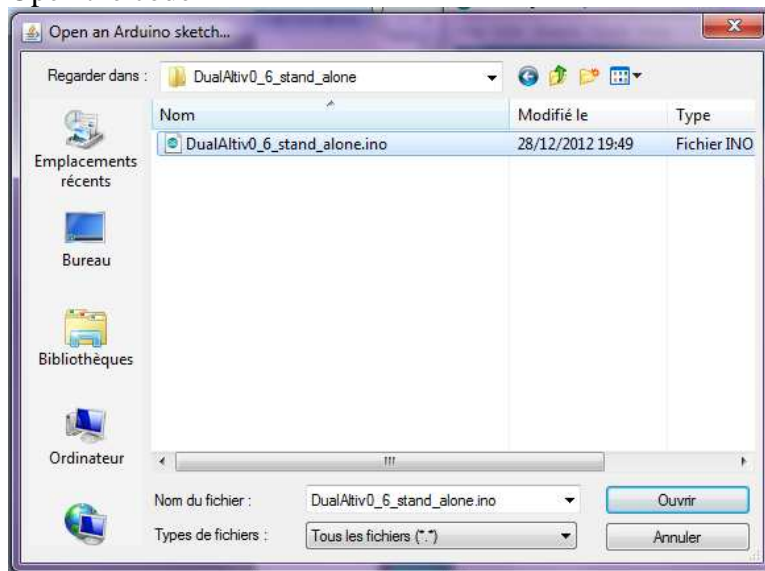




ATmega 328 dual altimeter – AltiDuo

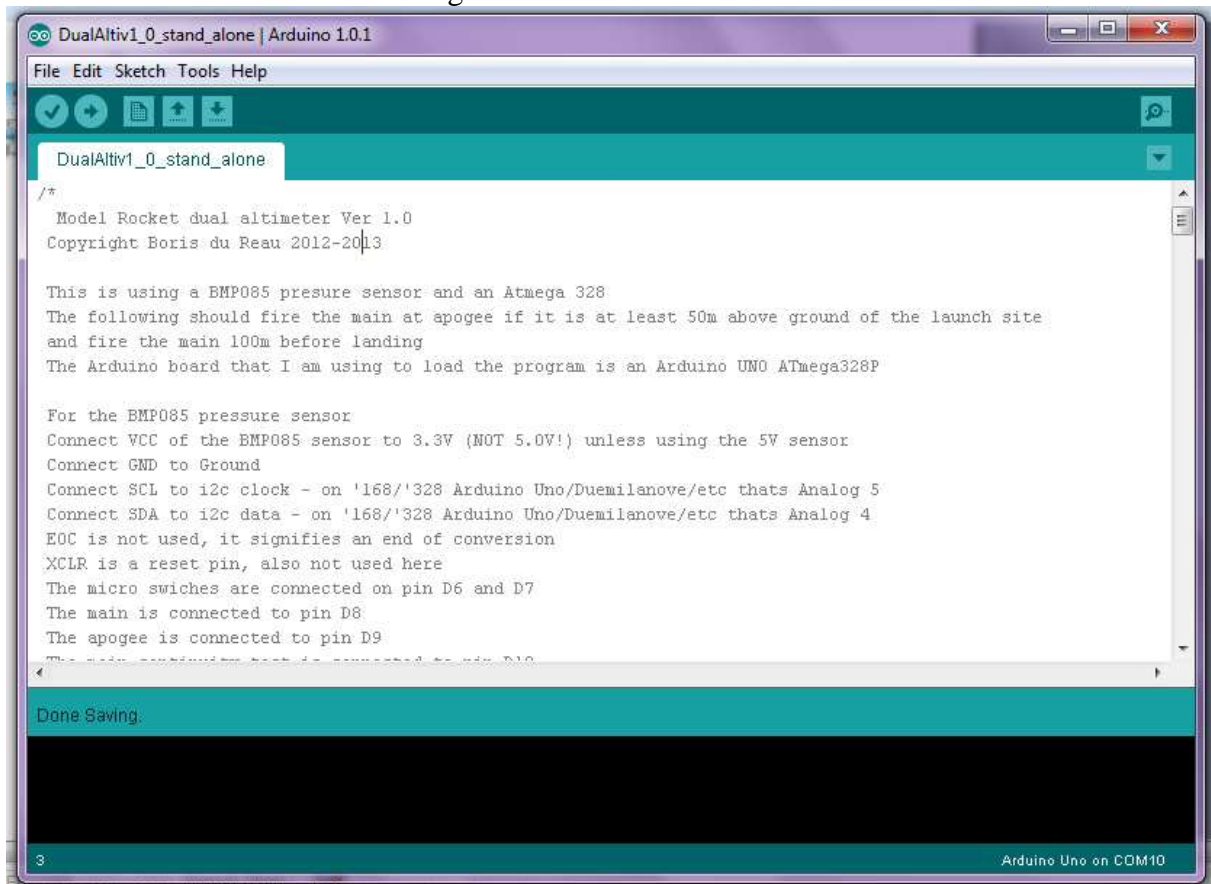


Open the code

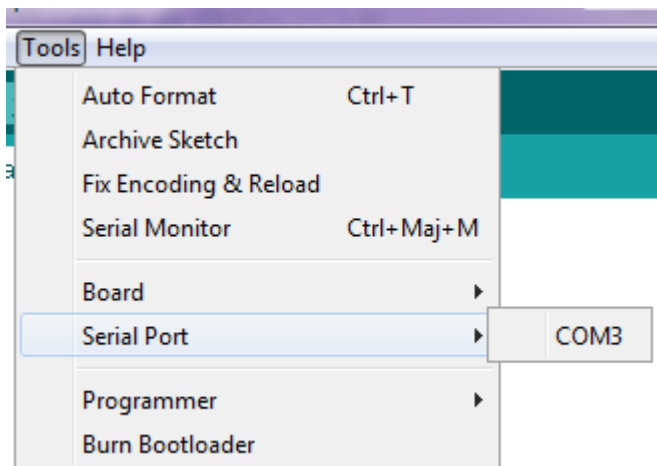




ATmega 328 dual altimeter – AltiDuo



After connecting the board using the USB adapter you need to select the correct serial port.



Then click on the horizontal arrow to upload the software.

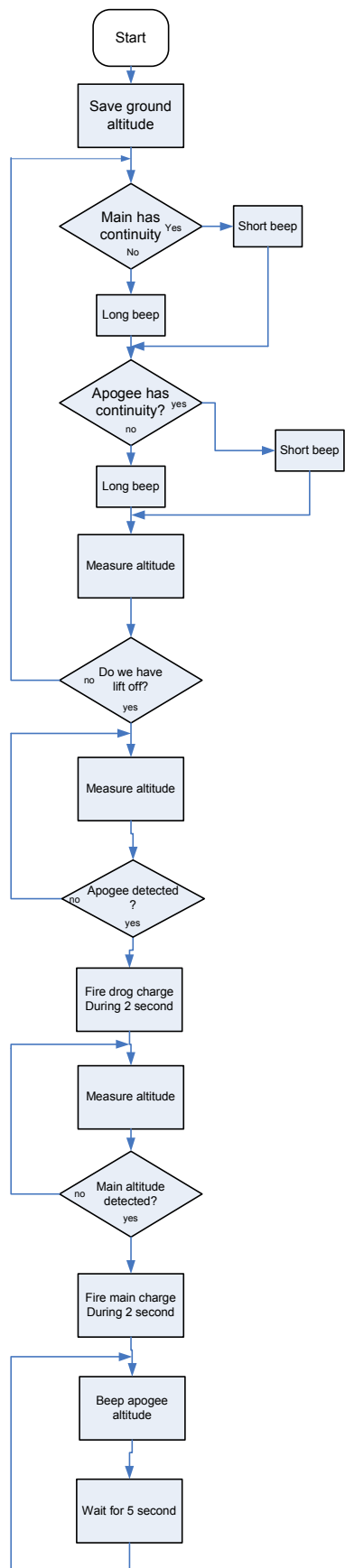


If the serial adapter used is a PL2303HX then you will need to press the card reset push button during 3 seconds before uploading the program. If you have chosen the *CP2102 model* the additional pin allow to do a reset of the card before loading the program.



ATmega 328 dual altimeter – AltiDuo

Functional diagram:





ATmega 328 dual altimeter – AltiDuo

Altimeter code:

```
/*
  Model Rocket dual altimeter Ver 1.0
  Copyright Boris du Reau 2012-2013

  This is using a BMP085 pressure sensor and an Atmega 328
  The following should fire the main at apogee if it is at least 50m above ground of
  the launch site
  and fire the main 100m before landing
  The Arduino board that I am using to load the program is an Arduino UNO ATmega328P

  For the BMP085 pressure sensor
  Connect VCC of the BMP085 sensor to 5.0V make sure that you are using the 5V
  sensor (GY-65 model)
  Connect GND to Ground
  Connect SCL to i2c clock - on 328 Arduino Uno/Duemilanove/etc thats Analog 5
  Connect SDA to i2c data - on 328 Arduino Uno/Duemilanove/etc thats Analog 4
  EOC is not used, it signifies an end of conversion
  XCLR is a reset pin, also not used here
  The micro switches are connected on pin D6 and D7
  The main is connected to pin D8
  The apogee is connected to pin D9
  The main continuity test is connected to pin D10
  The apogee continuity test is connected to pin D11
  The speaker/buzzer is connected to pin D12
  */

#include <Wire.h>

#include <Adafruit_BMP085.h>

#define DEBUG // =true

Adafruit_BMP085 bmp;

//ground level altitude
long initialAltitude;
//current altitude
long currAltitude;
//Apogee altitude
long apogeeAltitude;
long mainAltitude;
long liftoffAltitude;
long lastAltitude;
//Our drogue has been ejected i.e: apogee has been detected
boolean apogeeHasFired =false;
boolean mainHasFired=false;
//nbr of measures to do so that we are sure that apogee has been reached
unsigned long measures;
unsigned long mainDeployAltitude;

const int pinAltitude1 = 8;
const int pinAltitude2 = 7;

const int pinApogee = 9;
const int pinMain = 13;
const int pinApogeeContinuity = 10;
const int pinMainContinuity = 11;
const int pinSpeaker = 12;
const int pinLed = 13;

int nbrLongBeep=0;
int nbrShortBeep=0;
```



ATmega 328 dual altimeter – AltiDuo

```
boolean NoBeep=false;

void setup()
{
  int val = 0;      // variable to store the read value
  int val1 = 0;     // variable to store the read value

  // initialize the serial communication for debugging
  Serial.begin(9600);

  Wire.begin();
  //Pressure Sensor Initialisation
  bmp.begin();
  //our drogue has not been fired
  apogeeHasFired=false;
  mainHasFired=false;
  //let's read the launch site altitude
  initialAltitude = bmp.readAltitude();
  lastAltitude = 0; //bmp.readAltitude();
  //Initialize apogee to ground altitude + 20 meters
  liftoffAltitude = lastAltitude + 20;
  //number of measures to do to detect Apogee
  measures = 10;

  //initialise the deployment altitude for the main
  mainDeployAltitude = 100;

  //Initialise the output pin
  pinMode(pinApogee, OUTPUT);
  pinMode(pinMain, OUTPUT);
  pinMode(pinSpeaker, OUTPUT);
  pinMode(pinLed, OUTPUT);

  pinMode(pinAltitude1, INPUT);
  pinMode(pinAltitude2, INPUT);
  val = digitalRead(pinAltitude1);
  val1 = digitalRead(pinAltitude2);
  if(val == 0 && val1 ==0)
  {
    mainDeployAltitude = 50;
  }
  if(val == 0 && val1 ==1)
  {
    mainDeployAltitude = 100;
  }
  if(val == 1 && val1 ==0)
  {
    mainDeployAltitude = 150;
  }
  if(val == 1 && val1 ==1)
  {
    mainDeployAltitude = 200;
  }
  pinMode(pinApogeeContinuity, INPUT);
  pinMode(pinMainContinuity, INPUT);
  //Make sure that the output are turned off
  digitalWrite(pinApogee, LOW);
  digitalWrite(pinMain, LOW);
  digitalWrite(pinSpeaker, LOW);
  digitalWrite(pinLed, LOW);
}

void loop()
{
  //read current altitude
  currAltitude = (bmp.readAltitude()- initialAltitude);
  if (( currAltitude > liftoffAltitude) != true)
```



ATmega 328 dual altimeter – AltiDuo

```
{
  continuityCheck(pinApogeeContinuity);
  continuityCheck(pinMainContinuity);
}

//detect apogee
if(currAltitude > liftoffAltitude)
{

  if (currAltitude < lastAltitude)
  {
    measures = measures - 1;
    if (measures == 0)
    {
      //fire drogue
      digitalWrite(pinApogee, HIGH);
      delay (2000);
      apogeeHasFired=true;
      digitalWrite(pinApogee, LOW);
      apogeeAltitude = currAltitude;
    }
  }
  else
    lastAltitude = currAltitude;
}

  if ((currAltitude < mainDeployAltitude) && apogeeHasFired == true &&
mainHasFired==false)
  {
    // Deploy main chute 100m before landing...
    digitalWrite(pinMain, HIGH);
    delay(2000);
    mainHasFired=true;
    mainAltitude= currAltitude;
    digitalWrite(pinMain, LOW);
  }
  if(apogeeHasFired == true && mainHasFired==true)
  {
    beginBeepSeq();

    //#ifdef DEBUG
    Serial.println("Initial Altitude:");
    Serial.println( initialAltitude);
    Serial.println("Apogee Altitude:");
    Serial.println( apogeeAltitude);
    beepAltitude(apogeeAltitude);
    //#endif
    beginBeepSeq();

    //#ifdef DEBUG
    Serial.println("Main Altitude:");
    Serial.println(mainAltitude);
    beepAltitude(mainAltitude);
    //#endif
  }
}

void continuityCheck(int pin)
{
  int val = 0;      // variable to store the read value
  // read the input pin to check the continuity if apogee has not fired
  if (apogeeHasFired == false )
  {
    val = digitalRead(pin);
    if (val == 0)
    {
      //no continuity long beep
    }
  }
}
```




ATmega 328 dual altimeter – AltiDuo

```
        longBeep();
    }
    else
    {
        //continuity short beep
        shortBeep();
    }
}
}

void beepAltitude(long altitude)
{
    int i;
    // this is the last thing that I need to write, some code to beep the altitude
    //altitude is in meters
    //find how many digits
    if(altitude > 99)
    {
        // 1 long beep per hundred meter
        nbrLongBeep= int(altitude /100);
        //then calculate the number of short beep
        nbrShortBeep = (altitude - (nbrLongBeep * 100)) / 10;
    }
    else
    {
        nbrLongBeep = 0;
        nbrShortBeep = (altitude/10);
    }
    Serial.println("long beep:");
    Serial.println( nbrLongBeep);
    if (nbrLongBeep > 0)
    for (i = 1; i <  nbrLongBeep +1 ; i++)
    {
        longBeep();
        delay(50);
    }
    Serial.println("Short beep:" );
    Serial.println(nbrShortBeep);
    if (nbrShortBeep > 0)
    for (i = 1; i <  nbrShortBeep +1 ; i++)
    {
        shortBeep();
        delay(50);
    }

    delay(5000);
}

void beginBeepSeq()
{
    int i=0;
    if (NoBeep == false)
    {
        for (i=0; i<10;i++)
        {
            tone(pinSpeaker, 1600,1000);
            delay(50);
            noTone(pinSpeaker);
        }
        delay(1000);
    }
}

void longBeep()
{
    if (NoBeep == false)
```



ATmega 328 dual altimeter – AltiDuo

```
{
  tone(pinSpeaker, 600,1000);
  delay(1500);
  noTone(pinSpeaker);
}
void shortBeep()
{
  if (NoBeep == false)
  {
    tone(pinSpeaker, 600,25);
    delay(300);
    noTone(pinSpeaker);
  }
}
```