2019 CanSat Kit – User's Manual

Irish & European CanSat Competitions

Queries to Rob O' Sullivan
CanSat Ireland Project Coordinator
CIT Blackrock Castle Observatory

Email: rob.osullivan@bco.ie

Table of Contents

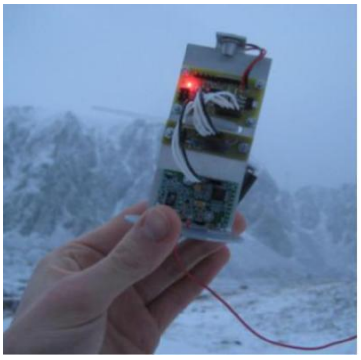
Acknowledgements

With special thanks to Eamon Connolly, Brenda Cooper and Valerie Cowman of CEIA.

We would like to thank **Prof. Jens Dalsgaard Nielsen, Simon Jensen** and other colleagues from **Aalborg University College (AAU)** in Denmark and **NAROM** in Norway for providing information regarding CanSat design, allowing us to use and providing us with the Gerber files for the AAU CanSat sensor shield, and allowing us to use information from the CanSat Handbook 2013 Edition in the preparation of this manual.

This user manual is written for CanSat teams new to the CanSat concept and can be used both as an introduction to Arduino programming/interfacing and CanSat. As a next step after this user manual, the **CanSat Handbook** is highly recommended.

1. CanSat Overview



A CanSat is a simulation of a real satellite, contained within the volume of a 330ml soft-drinks can. The ***Primary Mission*** of the CanSat is to remotely measure temperature and pressure, and transmit the data to the ground-station (laptop computer). In addition to a computer (Arduino UNO microcontroller), radio communications module, sensors and power supply, the CanSat will eventually need a parachute to land safely after launch from high altitude (eg. from a rocket, balloon, plane).

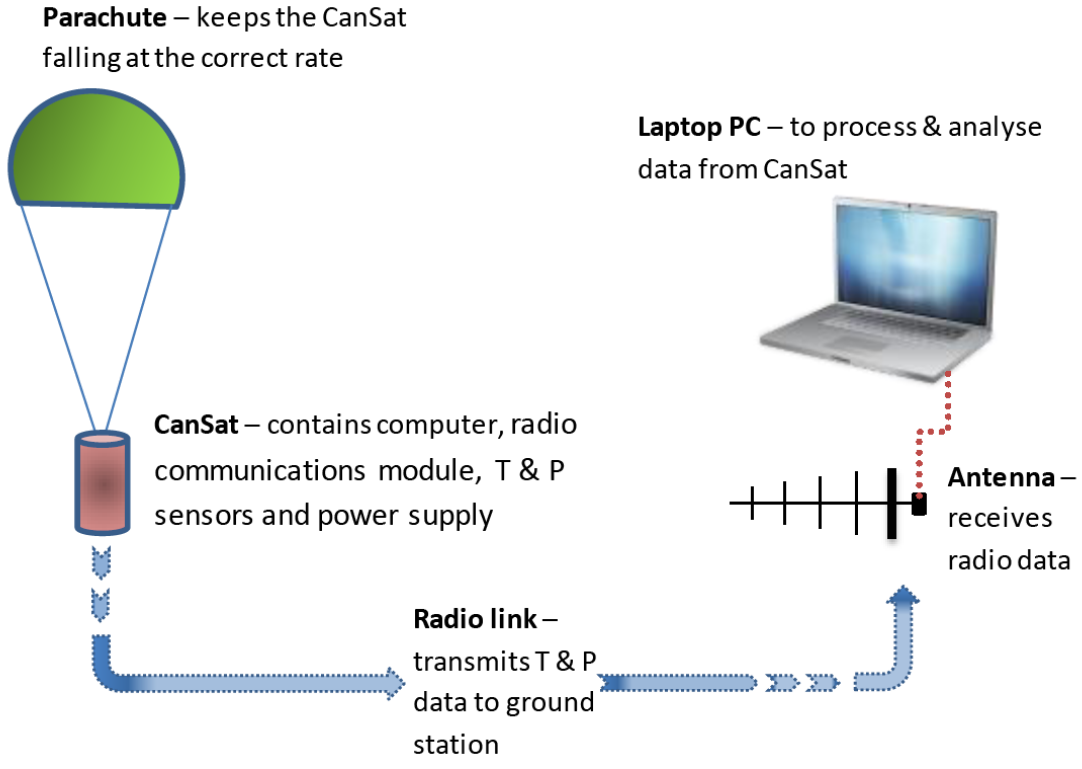


Fig.1 Overview of the Primary Mission of CanSat

As well as designing, building and testing the CanSat, a major part of the CanSat competition requires the student teams to analyse and present their data (and other work) to an audience.

Some ***Secondary Mission*** ideas for CanSat are given in *Section 11*, however this Users Manual focuses on the ***Primary Mission*** only

2. Arduino UNO for temperature and pressure measurements

*See **Appendix 1** for instructions about installing drivers for your Arduino UNO board and uploading programs from your computer to Arduino.*

In this section we will see how to program the Arduino to measure a voltage input to the Arduino from a sensor output.

2.1 Temperature measurement using a Thermistor

The resistance of a thermistor changes with temperature as seen in Fig.2. Because the resistance decreases with increasing temperature it is called a *negative temperature coefficient* (NTC) temperature sensor. *See also the datasheet for the thermistor in **Appendix 2**.*

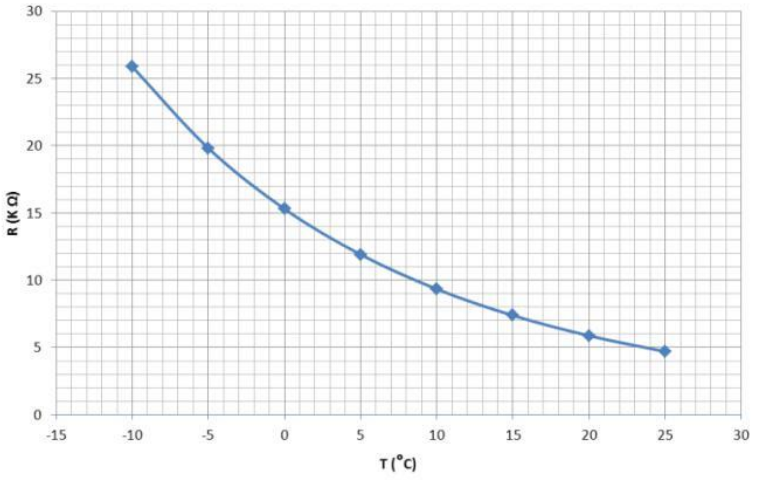


Fig.2. Calibration curve for a Thermistor temperature sensor

In the circuit schematic shown in Fig.3, we have a *potential divider circuit*. This means that the voltage (potential) at the black dot between the 2 resistors (10K and R(T)) depends on the value of R(T). Thus, the voltage at **pin A0** changes when the resistance of the thermistor changes (due to temperature change).

Similarly, to the calibration curve in Fig.1, you can make a calibration curve for your thermistor by plotting V_{A0} against Temperature.

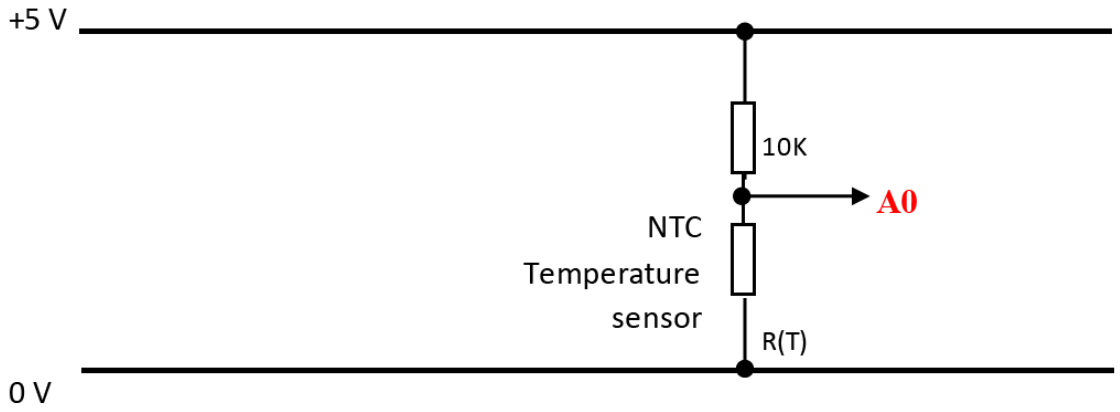


Fig.3. Circuit to measure a voltage input at* pin A0 *– analog input '0'

Solderless breadboards (see ***Appendix 3*** *for information about wiring)* are very useful for experimenting with circuits. No solder is used which means components can be easily changed around etc…

Use the solderless breadboard to set up the circuit as shown in Fig.4.

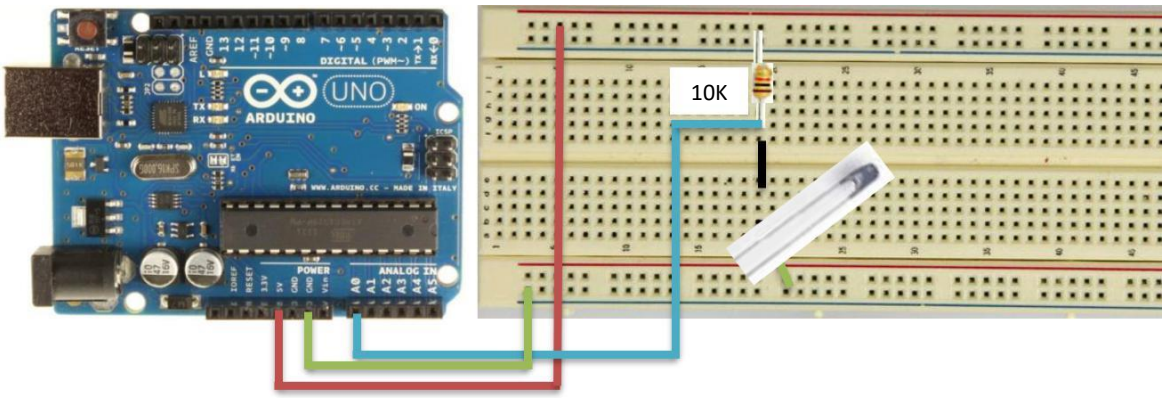


Fig.4. Breadboard prototype circuit to measure a voltage input at* pin A0 *– analog input '0'

Connect the Arduino board to your computer with the USB cable & startup the Arduino IDE on your computer.

2.1.1 COM port selection

Follow the steps indicated in Fig.5 – select **Tools → Serial Port → COM*X* –** the COM port will be different for every computer but usually not COM1 or COM2). In this case COM7 is used, and you will know from the bottom right-hand corner of the screen which COM port is connected.

Fig.5. Selecting the CanSat COM port

With the Arduino now connected to the correct COM port we can upload programs to it.

To start off we will upload a simple program which can read a voltage and print the data to screen:

select **File → Examples → 01 Basics → ReadAnalogVoltage**
This loads the program **ReadAnalogVoltage** into an Arduino IDE 'sketch' as shown in Fig.6.

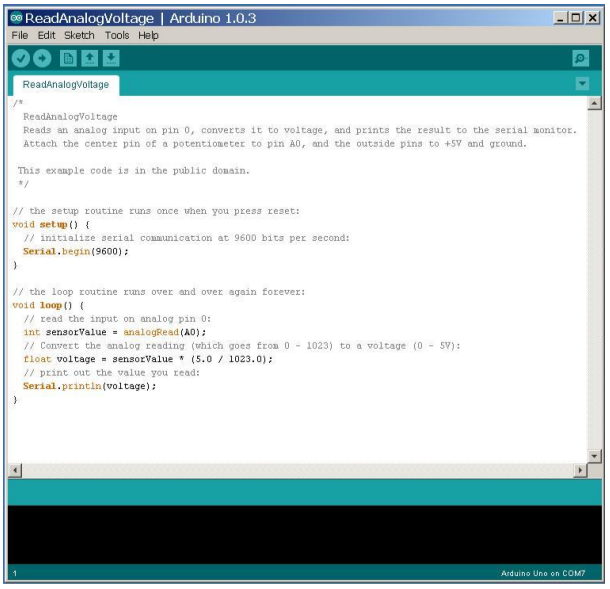


Fig.6. **ReadAnalogVoltage** program loaded to Arduino IDE

2.1.2 Explanation of ReadAnalogVoltage program

Arduino programs, or sketches, have 2 sections: ***setup & loop***. The ***setup*** section runs once, and the ***loop*** section loops until the reset button is pressed or the power (or USB cable) is disconnected.

SYNTAX NOTES

- Each line of code must be followed by a semicolon ;
- Brackets must be used after setup and loop, eg. setup ( ), loop ( )
- Code to be executed needs to be within curly brackets, eg. { code to be executed }
- lines with **//** in front of them are ignored by the program compiler

The **setup section** has just 1 line of code:

```
Serial.begin(9600);
```

This command sets up a serial communication link between the Arduino UNO (microcontroller) and the computer, with a speed (baud rate) of 9600 bits/second.

The **loop section** has 3 lines of code:

```
int sensorValue = analogRead(A0);
```

The analogRead command is used to read the voltage at **pin A0**. Specifically, this line assigns a value (0 – 1023) read from **pin A0** to the integer **sensorValue**, using the analogRead command.*

* **Note:** To process the voltage input from a sensor, the Arduino first converts this voltage from between 0 – 5V to an integer value between 0 – 1023. This enables the Arduino to deal with voltages such as 3.276V, or 2.045V for example.

```
float voltage = sensorValue * (5.0 / 1023.0);
```

This line converts the variable sensorValue from between 0 – 1023 to a value between 0 – 5. Specifically this line multiplies **sensorValue** by (5.0 / 1023.0) and then assigns this value to the variable **voltage**. In this case the variable is a 'float' meaning it has a decimal point.

```
Serial.println(voltage);
```

This line prints the value of **voltage** (eg. 2.737) to the serial monitor *and* then prints a carriage return (cursor goes to the next line).

2.1.3 Uploading the ReadAnalogVoltage program to the Arduino UNO

Clicking the  button will upload the code in the sketch area to the Arduino UNO.

On successful uploading you should see some lights flashing. Please refer to Appendix 1 if you have any difficulties uploading programs to the Arduino.

Now click on the  button to access the serial monitor.

A new window opens showing the values of the variable **voltage** in a list. This number is being continuously updated: every time the program loops it reads pin A0 and prints a new value to the

serial monitor. Because this happens so quickly and might generate too much unwanted data, a **pause command** could be used to create a delay between measurements:

Insert the following line immediately after the last line of code (the 500 indicates 500 milliseconds, or 0.5 seconds).

```
delay(500);
```

It's a good idea to try to write the complete program into another blank sketch. First close the window with the **ReadAnalogVoltage** program (don't save changes). Open up a new Arduino IDE sketch area :

Select **File → New**

The complete program listing (without comments) is:

```
void setup () {
        Serial.begin (9600);
}
void loop () {
        int sensorValue = analogRead(A0);
        float voltage = sensorValue * (5.0 / 1023.0);
        Serial.println(voltage);
        delay (500);
}
```

It is also possible to declare your variables before the setup section. The program following is the same as above but with the *variables declared before the setup section*:

```
int sensorValue;
float voltage;
void setup () {
        Serial.begin (9600);
}
void loop () {
        sensorValue = analogRead(A0);
        voltage = sensorValue * (5.0 / 1023.0);
        Serial.println(voltage);
        delay (500);
}
```

Upload this program to the Arduino and open the serial monitor again (make sure to close the other serial monitor window first). Now the data should be updated every 500ms.

2.2 Pressure measurement using the MPX4115 pressure sensor

The pressure sensor used is the MPX4115A from Freescale Semiconductor. It uses a silicon *piezoresistive* sensor element. Fig.7 shows a schematic of the sensor. For more information see the MPX4115A pressure sensor datasheet in ***Appendix 4***.

Piezoresistive means that the resistance of a material change when a mechanical stress is applied. In this case silicon is used. The changes of resistance for silicon are magnitudes of times larger than for metals, making this material very useful to use in a pressure sensor.

The MPX4115A pressure sensor has 6 pins, and that pin 1 can be identified by a small notch on the leg. **Pins 1, 2 and 3 are assigned to V_{out}, 0 V, and V_{supply} respectively**.

Fig. 7 shows a graph from the sensor datasheet with the accompanying transfer function.

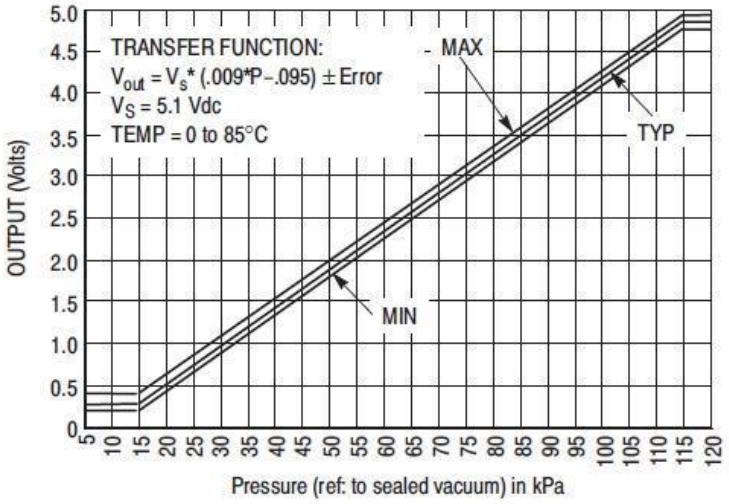


Fig.7: The MPX4115A transfer function

Fig.7 tells us that at 100kPa (=1000hPa =1000millibars), the voltage output will be about 4.2 volts

We can rearrange the transfer function

$$V_{\text{out}} = V_{\text{supply}} * (0.009 * P - 0.095) \pm Error$$

to get

$$P = \frac{\frac{V_{\text{out}}}{V_{\text{supply}}} + 0.095}{0.009}$$

This gives Pressure P in *kPa*. Change the denominator to 0.0009 to obtain P in *hPa* (or *mbars*).

Using a solderless breadboard, similarly to Fig.4, set up the circuit shown in Fig.8. Note that pins 4-6 of the MPX4115A pressure sensor are not connected.

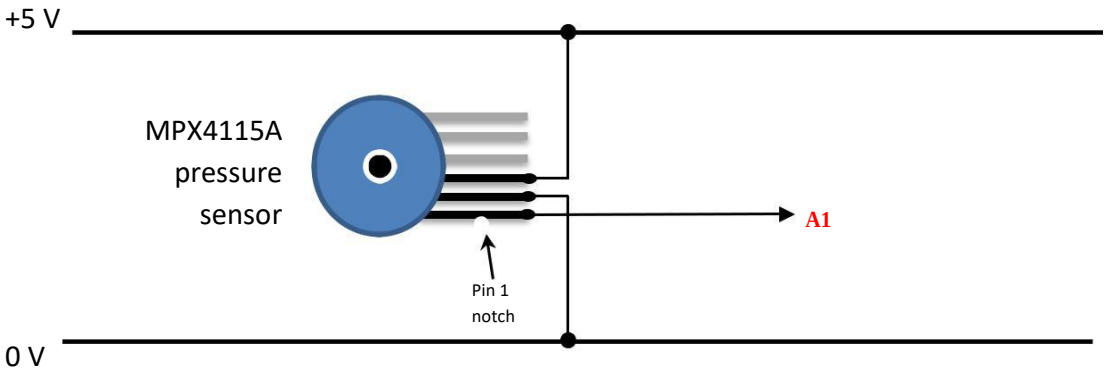


Fig.8. Circuit to measure a voltage input at* pin A1 *– analog input '1'

2.2.1 Writing an Arduino program to measure pressure

When you have the breadboard circuit ready, you can write the program. Firstly, open a new Arduino IDE sketch to enter the code.

The program will be similar to the last one to obtain reading from the thermistor:

```
int pressureValue;
float pressure;

void setup ( ) {
        Serial.begin (9600);
}
void loop ( ) {
        pressureValue = analogRead (A1);
        pressure = ((pressureValue / 1024.0) + 0.095) / 0.0009;
        Serial.print("CanSat_name / unique_ID ");
        Serial.print("Pressure = ");
        Serial.print(pressure);
        Serial.println (" millibars");
        delay (500);
}
```

This program reads an analogue voltage at pin A2 and processes it so that the value of pressure in hPa(millibars) is printed to the serial monitor.

Note: V_O/V_S is a ratio; using 1024.0 instead of V_S (=5.1V) saves calculations and processing time. Upload this program to the Arduino UNO as shown in *Section 2.1.3.*

Can you **modify this program** to **include temperature measurement** & upload it to the Arduino?

2.2.2 Calculating altitude from pressure (see Appendix 7 for details)

Air pressure and altitude are related according to the graph in Fig.9. This shows that increasing altitude results in decreasing pressure.

Air pressure can be calculated using the equation

$$P = 101325\ (1 - 2.25577\ 10^{-5}\ h)^{5.25588}$$

where **P** = air pressure (Pa) and **h** = altitude above sea level (m)

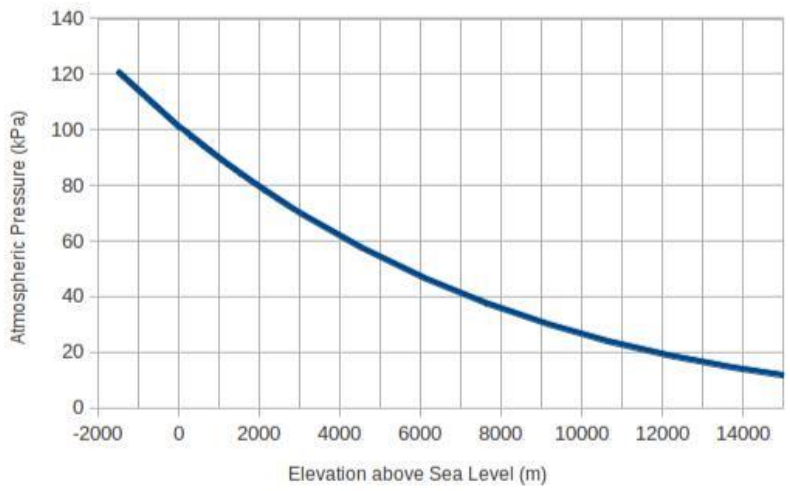


Fig.9 Elevation and atmospheric pressure *(from www.engineeringtoolbox.com)*

This equation can be re-arranged to obtain altitude:

$$log\left(\frac{P}{101325}\right) = 5.25588\ log(1 - 2.25577 \times 10^{-5} h)$$

In Arduino, the **pow** function can be used to obtain **h**.

pow(base, exponent)

See Appendix 7 for details on how to calculate altitude from pressure data.

See Also the ***Arduino Reference*** (in an Arduino window, select **Help → Reference**) for syntax info on this and other Arduino commands.

3. Radio Communications

*See **Appendix 5** for the APC220 Wireless Communication Module data sheet*

For the CanSat competition an *APC220 Wireless Communication Module* from Appcon technologies will be used – Fig.10. The APC220 is a UHF transceiver.

Figure 10: The APC220 communication module (from www.DFRobot.com)

The APC220 has 2 identical parts (radios) for transmitting and receiving data. Both radios can transmit and receive data. For the CanSat ***Primary Mission*** we will use one to transmit (TX) data from the CanSat, and one to receive (RX) data at the laptop PC – the USB converter is used to connect the RX radio to the laptop PC.

3.1 Transmitting data using the APC220 module

Fig.11 shows a schematic of one of the APC220 radios.

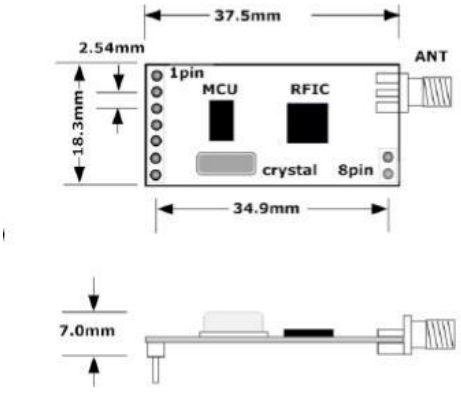


Fig.11. APC220 Wireless Comm. radio schematic

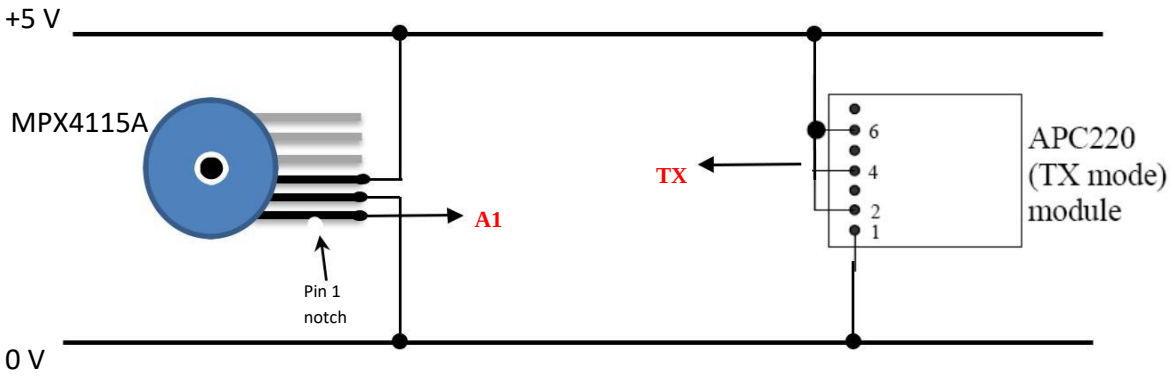


Fig.12. Circuit to connect APC220 wireless communication module

Set up the circuit shown in Fig.12. In this circuit this APC220 radio is set to be **permanently in TX mode** because pin 6 is connected with the 5V (i.e. set HIGH).

If RX was also needed (to receive data at the CanSat), we could use one of the Digital Pins on the Arduino to change the mode between TX (pin 6 = HIGH) or RX (pin 6 = LOW).

To send data via the transmitting radio pin 4 of the radio must be connected to the Arduino TX pin (Digital pin 1).

Use the same code as in *Section 2.2.1* to transmit data via the APC220 TX radio.

3.2 Receiving data using the APC220 module

To receive the transmitted signal the second APC220 radio needs to be connected to a different USB port on the computer using the USB – TTL converter.

When connecting the USB – TTL converter, a driver will need to be installed.

This can be found on the web at:
http://www.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx

Or search for “silabs cp210x usb driver” in Google.

With the **USB – TTL converter driver** installed an extra com port is created. You can check this by **using the TERMINAL program**: after running TERMINAL, click **Connect** to access data from the new COM port (NB. Make sure this is not the Arduino COM port). You may need to click on the **ReScan** button.

Now you should be receiving the data from the pressure sensor via the APC220 radio link. You can log this data by clicking **StartLog**. Stop logging by clicking **StopLog**.

3.3 Changing the APC220 operating frequency (*adapted from The CanSat Book*)

When there are several CanSats operating in close proximity (eg. during group sessions and CanSat competition), assigning different 'operating' frequencies to different CanSats will facilitate data collection – without this option, all 'ground stations' would be receiving data from all CanSats.

Connect the Arduino board to the computer and upload the program "**apc220cfg.ino**" which is found on the memory stick (or NAROM's web site - http://www.narom.no/).

Make sure you upload the program ***before*** you try to connect the transceiver to the Arduino board. Disconnect the USB cable (and battery) from the Arduino board and connect the transceiver to the Arduino as shown in Fig 13.

Figure 13. Connecting the transceiver to the Arduino UNO

The transceivers will be connected to the pins labelled GND, 8, 9, 10, 11, 12 and 13 on the Arduino board.

Reconnect the Arduino board trough the USB cable and open the Serial Monitor. In the command line at the top, type in 'm' and hit enter. This will bring up the menu shown in Figure 14.

If you type 'r' and hit enter, the program will return the current configuration for the transceiver. To reconfigure the radio, type 'w' and the 6 parameters needed, with space between each parameter.

Note that you have to configure **both** transceivers with the same settings to be able to use them together.

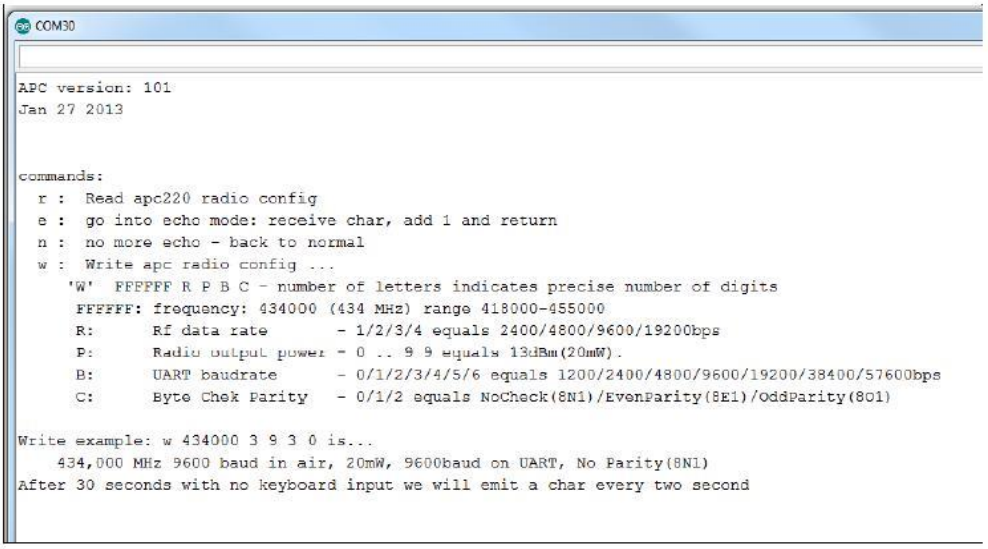
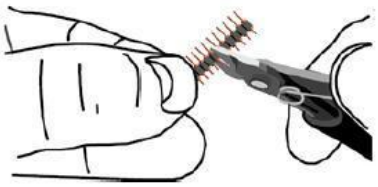


***Figure 14. Configuring the transceiver modules using* apc220cfg.ino**

4. AAU CanSat sensor board (*adapted from The CanSat Book*)

In order to fit all the components neatly inside the can we will use a printed circuit board (PCB) from AAU in Denmark. Instead of wires between the components a PCB uses layers of printed conductors (metals).

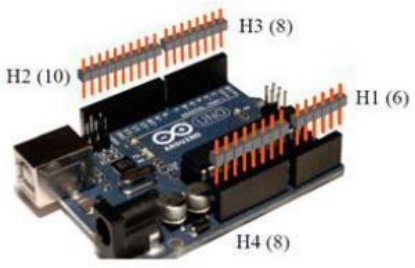
Using a PCB will save space and keep everything in its correct position. The circuits will be robust but more difficult to change than with the solderless breadboards.



Appendix 6 gives more details about the AAU PCB.

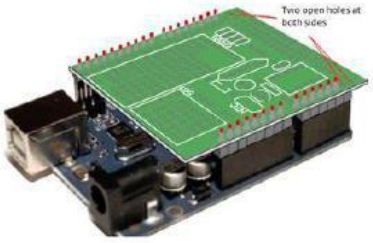
To connect the AAU PCB we first need to solder headers onto the PCB.

Cut the male pin connector header in to the following lengths: - 6 pins (H1) - 10 pins (H2) - 8 pins (H3) - 8 pins (H4).



Insert the connector headers into the Arduino board with the short end up.

Mount the AAU shield board on top of the Arduino Uno.
Note: The board should fit only one way.



Solder all the pins on the top of the circuit board and then remove it from the Arduino Uno.

Make sure not to heat the pins too long while soldering. Too long exposure to heat might damage the Arduino board.

As shown in fig. 15, we suggest using 7-way & 6-way female headers to enable easy removal/change of the rf-radio & the pressure sensor. This also allows the space under the rf-radio to be used for a gps module, for example. 2-way headers can be cut from the 8-way headers: 1 is used to enable 'jumpering' of J1 on the PCB – this is necessary to be able to send the data via the TX pin (analog input pin 1).

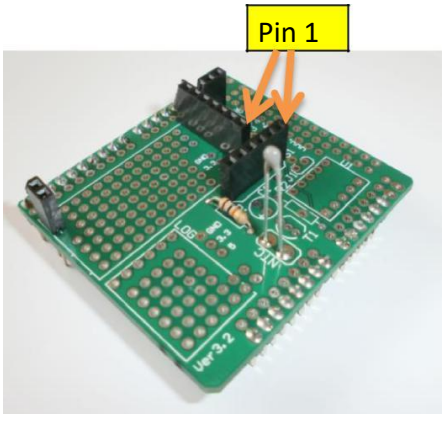
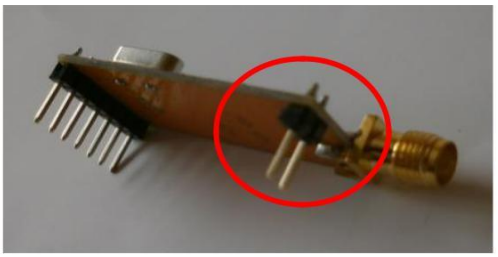
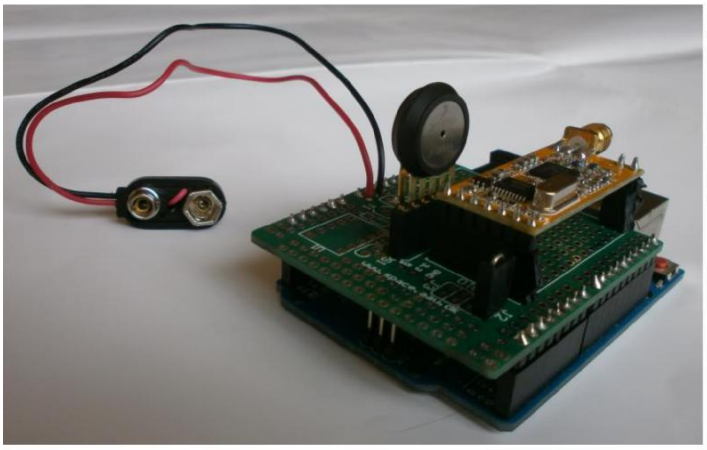


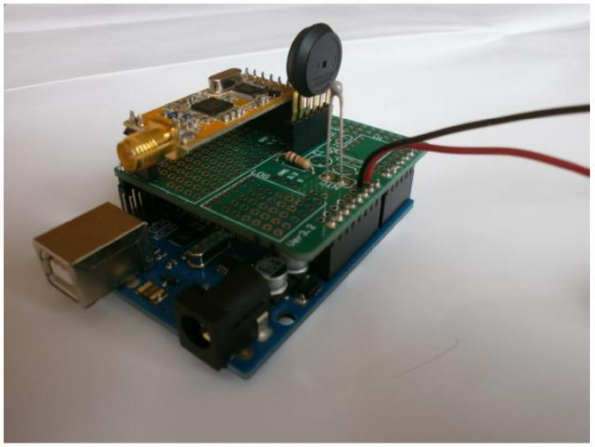
Fig. 15. Using stackable female headers on the AAU sensor board

The 10 KΩ resistor (R1) and the thermistor temperature sensor are also shown soldered in place.

A 2-pin ‘stabilising’ header should also be soldered on to the TX rf-module as shown on the right.







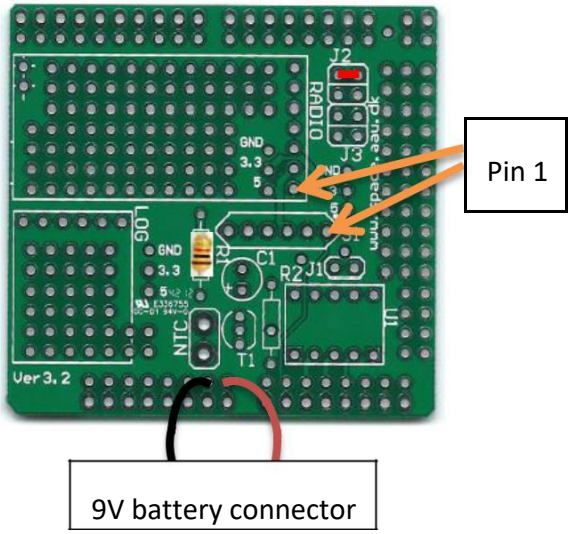


Fig. 16 Indicating pin 1 positions for the MPX4115A pressure sensor and the APC220 radio, the J2 jumper and the 9V battery connector

Fig. 16 shows the pin 1 positions for the MPX4115A pressure sensor and the APC220 radio. Also shown is the 10KΩ resistor and the connections for the 9V battery connector.

To transmit data via the Arduino TX pin, we also need to ‘jumper’ the J2 holes as shown in red.

It is a good idea to use headers to connect the MPX4115A and APC220 to the sensor board instead of soldering them in directly – this will enable easy removal/replacement of these components if necessary.

5. CanSat construction

5.1 CanSat frame and housing

A piece of aluminium is provided in the CanSat kit as a frame to attach the parachute to and mount the Arduino & electronics/sensors etc.

Of course it is not a requirement to use the Aluminium plate provided as the CanSat frame, and figure 17 shows just one way the CanSat might be designed.

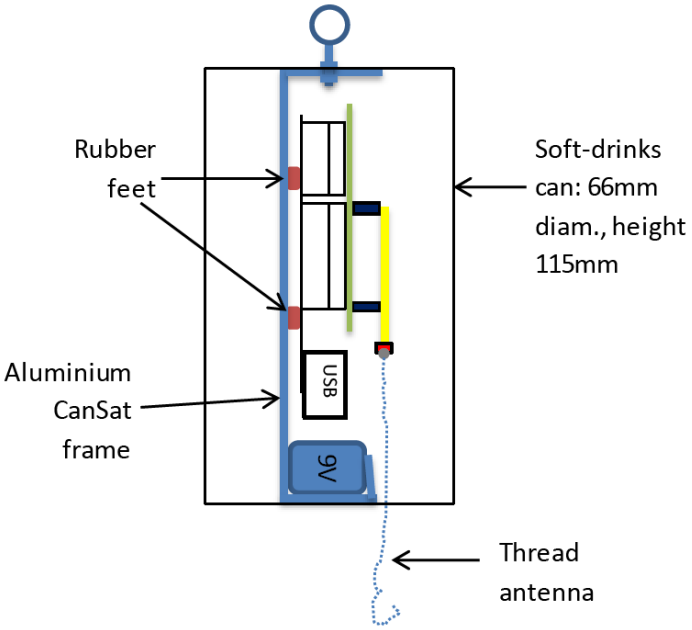


Figure 17. A possible CanSat design using the aluminium plate as a frame

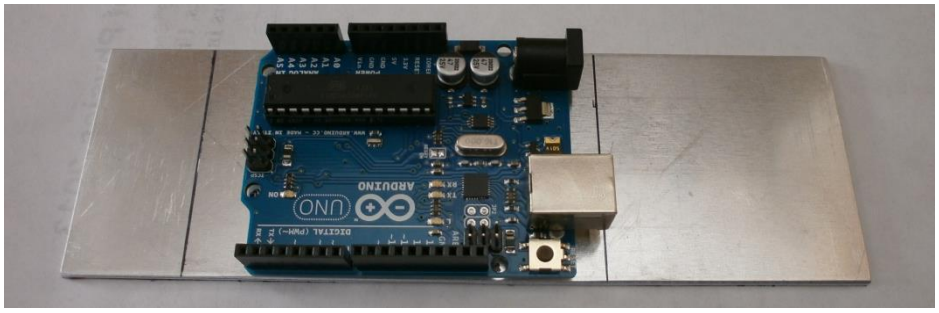


Figure 18. Using the Arduino board to mark positions to drill holes in aluminium plate

Remember, **everything** – except the eye-bolt, parachute and antenna wire, **must fit inside** the volume of a soft-drinks can: 115mm high, 66mm diameter.

Use the Arduino to mark positions to drill 3mm holes in the Al as shown in Fig. 18.

You will also need to drill a 6mm hole for the eye-bolt to attach the parachute, and a bigger hole for the USB cable (otherwise you will have to remove your Arduino from the frame every time you want to change the program – not a good design!).

The Al plate has to be bent into shape to create the battery holder.

The parachute eye-bolt section also needs to be bent into shape. Several different designs are shown in the images below.

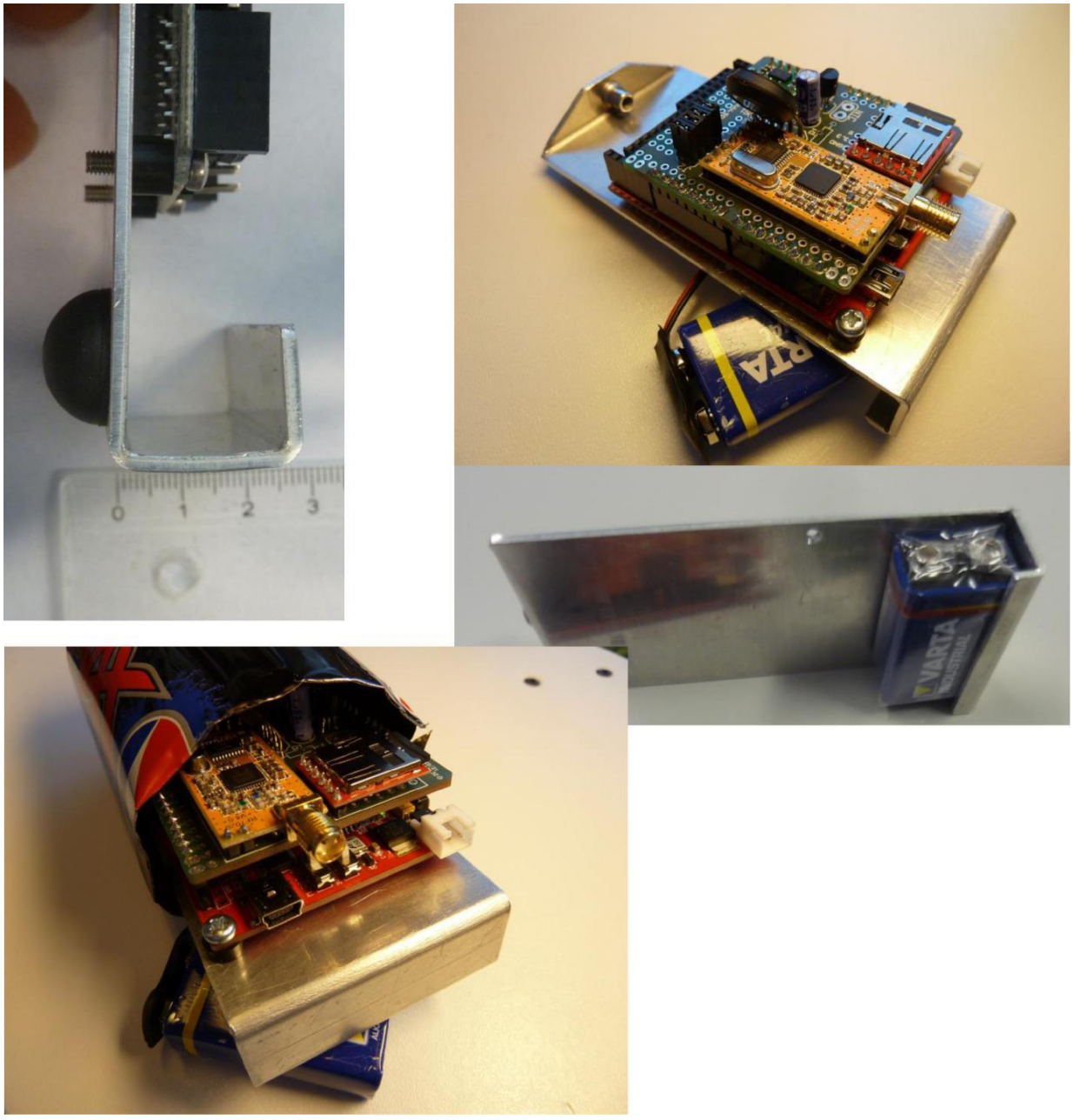


Fig.19 CanSat frame & housing construction – several ways to design the CanSat

5.2 CanSat antenna

The antenna that is included with the APC220 radio is a 433MHz Rubber Duck antenna. This antenna is robust and great for testing, but won't fit inside a soda can.

Fig.20 Duck Antenna

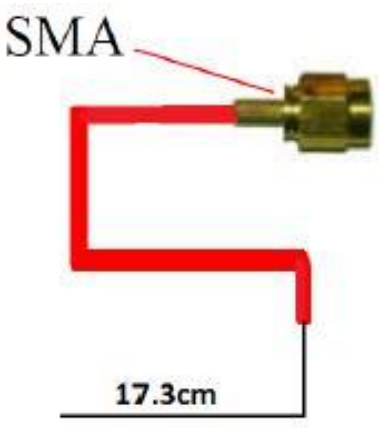
A good alternative is the simple thread antennas that can be soldered directly on to the transmitter output or attached to a SMA-connector. Normally such antennas will be a quarter-wavelength. The thickness is not critical; the most important is the flexibility and durability.

Antenna length can be calculated from equation below when the frequency (f), and velocity of light (c) is known:

If we have a frequency of 434 MHz, then the equation for calculating the length of a quarter wavelength is:

$$L = \frac{c}{4f} = \frac{3\text{x}10^8}{4(434\text{x}10^6)} = 0.173\ m$$

By this calculation we find that the antenna should be 17.3 cm long.

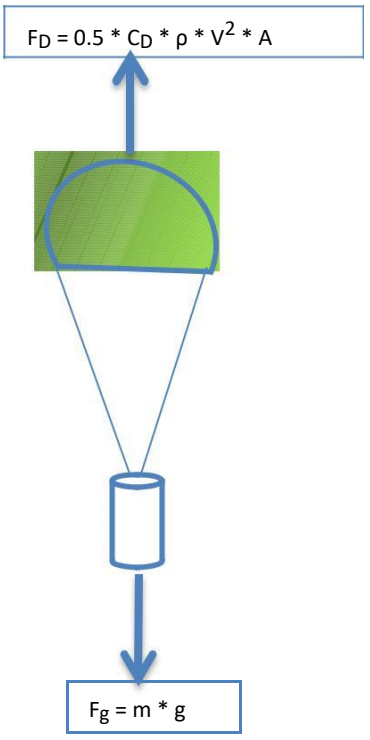


We can build this antenna by using a coaxial cable. Remove 17.3 cm of the plastic jacket and the metallic shield from the cable, leaving the centre core and dielectric insulator. Make sure that the shielded part of the cable reaches all the way out of the can before it is stripped.

Fig.21 Thread Antenna

See also the CanSat Book for more information about CanSat design.

6. Parachute design (*adapted from The CanSat Book*)



The parachute will provide a safe landing for the CanSat as it returns to Earth. The descent rate is required to be within certain values: if the CanSat descent was too slow it could drift too far with wind; too fast and the CanSat might be damaged.

Requirements Descent Parameters

Minimal descent Velocity: 8 m/s
Maximal descent Velocity: 11 m/s

Maximum allowed CanSat mass: 350 grams

The deployment of the parachute will be relatively violent, so the fabric and fibres you use need to be strong. Most often you can get nylon wires and rib stop fabric at a kiting shop. These materials are ideally suited for the parachute.
When cutting the fabric, you should take into account the fact that some of the fabric needs to be doubled to be able to sew it.

During descent 2 main forces act on the CanSat:

The force pulling the CanSat to earth is due to gravity is: $F_g = m * g$ where

m: is the mass of the CanSat

g: is the acceleration of gravity, equal to 9.81 m/s^2

As the CanSat descends through air it experiences a drag force due to the parachute:

$$F_D = 0.5 * C_D * \rho * A * V^2$$ where

A: is the total area of the parachute (not just the frontal area)

C_D: is the drag coefficient of the parachute – value depends on the **shape** of the parachute

ρ: is the local density of the air, assumed to be constant at 1.225 kg/m^3.

V: is the descent velocity of the CanSat

Drag coefficients C_D: Semi Spherical: 1.5
Flat, hexagon: 0.8

When the CanSat is deployed, the force of gravity will cause it to *accelerate*. After a few seconds the drag force from the parachute will reach *equilibrium* with the force of gravity. From that point on, the acceleration will be zero and the CanSat will descend at a *constant velocity*. This constant velocity has to be *within the min/max descent velocities* specified above.
Some more handy tips on the production of the parachute can be found here:
http://www.nakka-rocketry.net/paracon.html
http://www.nakka-rocketry.net/xchute1.html
http://www.sunward1.com/imagespara/The%20Mathematics%20of%20Parachutes%28Rev2%29.pdf

See also the CanSat Book for more information about parachute design.

7. Testing & Calibration of sensors

Sensors need to be calibrated to ensure correct measurement data. In practice, calibration involves recording a sensor output (resistance, voltage, etc) when placed in a controlled environment.

For example, to calibrate a thermistor (Junior cert. Science experiment & Leaving cert. Physics experiment), the thermistor is placed in iced water and its output recorded. The ***standard*** thermometer used is a mercury-in-glass lab. thermometer.

Heat is applied to the water/ice mixture slowly (hotplate or Bunsen burner) and the resistance of the thermistor is recorded against the temperature. A graph of resistance versus temperature should look similar to Fig. 2

Pressure sensor calibration will be a little bit more complicated. Met Eireann data (www.met.ie) might be useful.

Altitude calibration might be possible using ordinance survey data (www.osi.ie)

Many ideas also in the CanSat Book.

8. Data Analysis & Presentation

CanSat is primarily an educational activity so Data Analysis & Presentation is one of the main areas where CanSat teams will be able to impress competition judges with skills they have learned during their CanSat work. Important skills include among others: electronic design, programming, mechanical design, team work, problem solving, communication (presentation)…

TERMINAL.exe is a freeware that reads and store data coming from the serial ports.

Data saved using Terminal can be opened using Microsoft Excel to analyse & produce graphs.

In making a presentation it's important to show what you found (measurements) and to also interpret what the data means. Also check to see if there is other (unwanted?) information in the data (eg. Bernoulli ef fect on the pressure measurements due to the can moving through air).

What else can you see? What would you do differently next time? Why?

Use Microsoft PowerPoint or Prezi to make a presentation of all your work, data and data analysis, etc.

Practise the presentation in front of an audience (eg. class). Ask for feedback from the audience to see how you can improve the presentation.

9. Outreach & Communications

As well as making publicity for your CanSat team, communicating your CanSat work/progress to the public will be very useful in making the final presentation. Some tips include:

Take photos regularly to record your progress.

Keep a logbook – enter everything here (it doesn't have to be really neat, just useful/readable).

Set up a Facebook page to document your CanSat activities.

Write articles for local/national papers etc.

Check previous CanSat winning projects

10. Arduino & CanSat resources

There are many internet resources available for Arduino & CanSat. A few really useful links are:

www.arduino.cc

www.element14.com

www.cansat.eu

www.narom.no - check English version of site; download CanSat Book latest version from here

www.engineeringtoolbox.com

11. Secondary Mission ideas

The secondary mission is open to your imagination, scientific understanding and engineering skills. Some ideas from past secondary missions are:

Determine exact CanSat position using accelerometer and/or GPS module

Land CanSat at pre-determined location – difficult; requires 2-way communication to steer CanSat

Deploy an experiment outside the CanSat – eg. monitor CO_2 concentration during descent

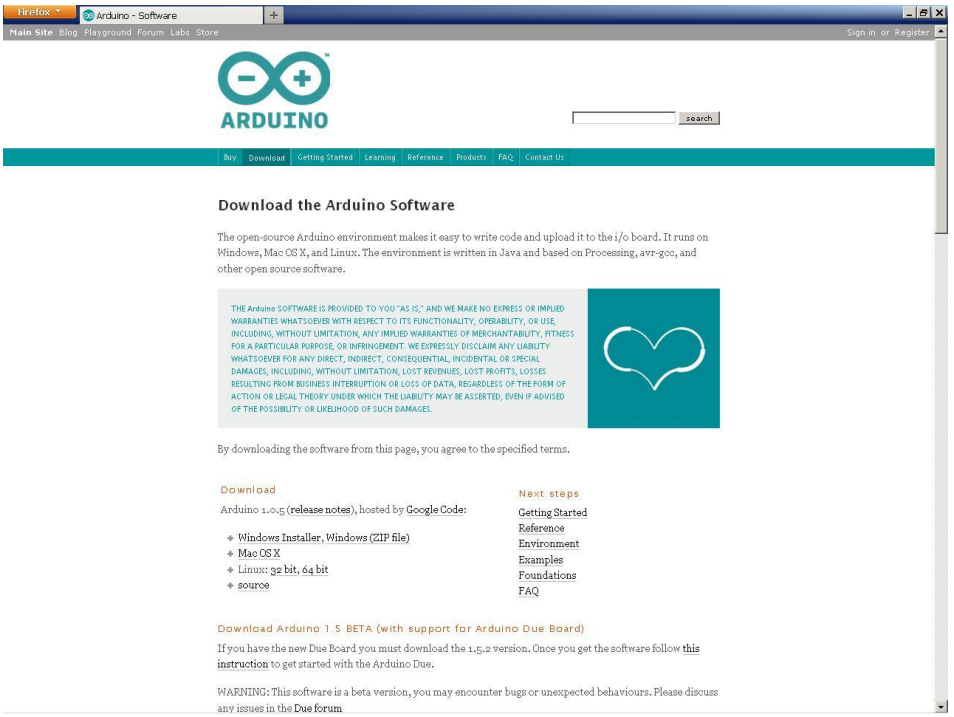
Take videos when descending

Generate power when descending

A lot of information on satellites missions can be found on the internet. Try and see what ESA is doing at the moment, or NASA. Search on the website of Arduino (www.arduino.cc/) to see what would be possible.

Appendix 1 – Getting started with Arduino

1. **Visit** http://arduino.cc/en/
2. **Click on the** Download **tab to bring you to the screen shown below.**
3. **Choose your operating system. Follow the on screen instructions to download the appropriate Arduino IDE (Integrated Development Environment) to your computer.**



4. **When the download finishes, unzip the downloaded file. Make sure to preserve the folder structure. Double-click the folder to open it. There should be a few files and sub-folders inside.**
5. **Click on the** Getting Started **tab to bring you to the guide – reprinted from website on the following pages.**

Document below copied from http://arduino.cc/en/ **website. Licensed to copy under terms** at: http://creativecommons.org/licenses/by-sa/3.0/

Getting Started w/ Arduino on Windows

1 | Get an Arduino UNO board and USB cable

You'll need a standard USB cable (A plug to B plug): the kind you would connect to a USB printer, for example. (For the Arduino Nano, you'll need an A to Mini-B cable instead.)

2 | Connect the board

Connect the Arduino to your computer using the USB cable. The green power LED (labelled **PWR**) should go on.

3 | Install the drivers

Installing drivers for the Arduino Uno or Arduino Mega 2560 with Windows7, Vista, or XP:

- Plug in your board and wait for Windows to begin it's driver installation process. After a few moments, the process will fail, despite its best efforts
- Click on the Start Menu, and open up the Control Panel.
- While in the Control Panel, navigate to System and Security. Next, click on System. Once the System window is up, open the Device Manager.
- Look under Ports (COM & LPT). You should see an open port named "Arduino UNO (COMxx)"
- Right click on the "Arduino UNO (COmxx)" port and choose the "Update Driver Software" option.
- Next, choose the "Browse my computer for Driver software" option.
- Finally, navigate to and select the driver file named **"arduino.inf"**, located in the "Drivers" folder of the Arduino Software download (not the "FTDI USB Drivers" sub-directory). If you are using an old version of the IDE (1.0.3 or older), choose the Uno's driver file named **"Arduino UNO.inf"**
- Windows will finish up the driver installation from there.

See also: step-by-step screenshots for installing the Uno under Windows XP.

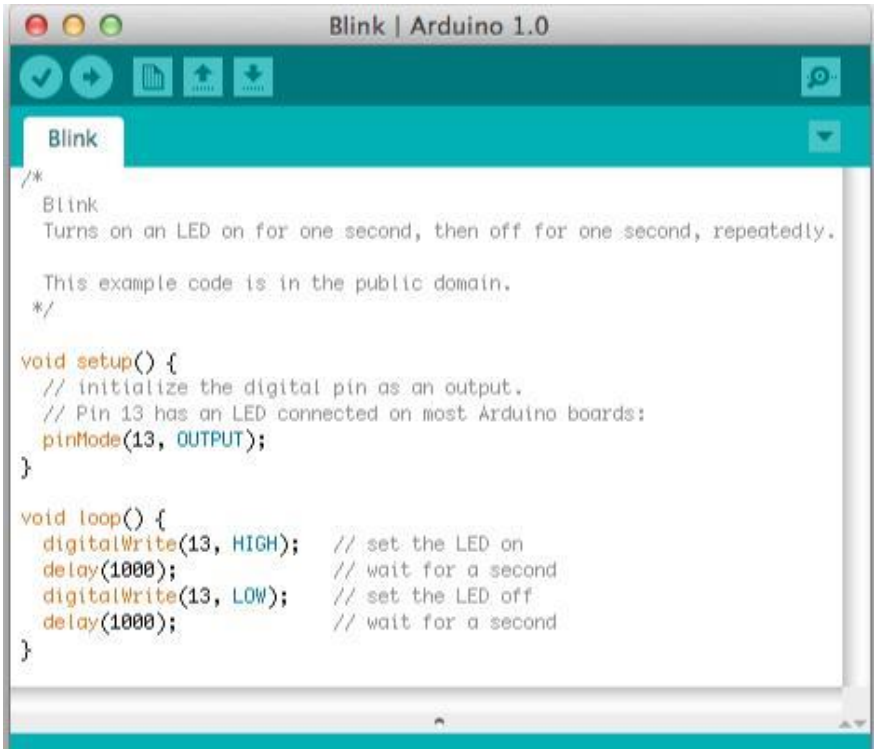
4 | Launch the Arduino application

Double-click the Arduino application. (Note: if the Arduino software loads in the wrong language, you can change it in the preferences dialog. See the environment page for details.)

Ireland esero BCO The Space For Science Science Foundation Ireland sfi For what's next ceia.ie cork's technology network

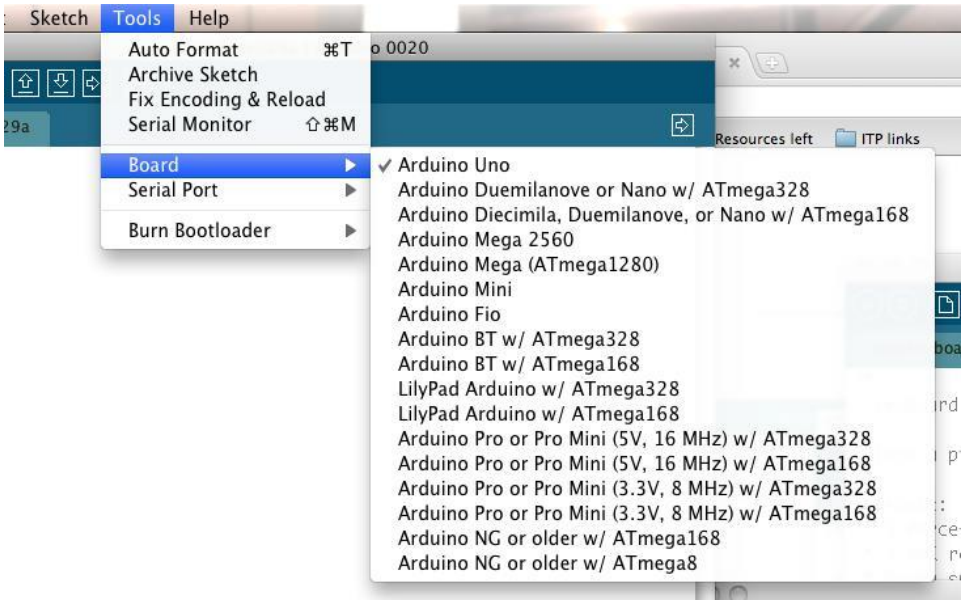
5 | Open the blink example

Open the LED blink example sketch: **File > Examples > 1.Basics > Blink**.



6 | Select your board

You'll need to select the entry in the **Tools > Board** menu that corresponds to your Arduino.



Selecting an Arduino Uno

7 | Select your serial port

Select the serial device of the Arduino board from the Tools | Serial Port menu. This is likely to be **COM3** or higher (**COM1** and **COM2** are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu; the entry that disappears should be the Arduino board. Reconnect the board and select that serial port.

8 | Upload the program

Now, simply click the "Upload" button in the environment. Wait a few seconds - you should see the RX and TX leds on the board flashing. If the upload is successful, the message "Done uploading." will appear in the status bar. (*Note:* If you have an Arduino Mini, NG, or other board, you'll need to physically present the reset button on the board immediately before pressing the upload button.)

A few seconds after the upload finishes, you should see the pin 13 (L) LED on the board start to blink (in orange). If it does, congratulations! You've gotten Arduino up-and-running.

If you have problems, please see the troubleshooting suggestions.

The text of this Arduino getting started guide is licensed under a Creative Commons Attribution-ShareAlike 3.0 License (http://creativecommons.org/licenses/by-sa/3.0/) **Code samples in the guide are released into the public domain.**

Appendix 2 – Thermistor data sheet

2381 640 10.../NTCLE101E3...SB0

Vishay BCcomponents

VISHAY.

NTC Thermistors, Radial Leaded Special Accuracy

FEATURES

- Excellent accuracy between 25 °C and 85 °C
- High stability over a long life
- Old part number was 2322 640 10...
- Compliant to RoHS directive 2002/95/EC and in accordance to WEEE 2002/96/EC

Pb e3 RoHS COMPLIANT

APPLICATIONS

- Temperature measurement, sensing and control

QUICK REFERENCE DATA

PARAMETER	VALUE
Resistance at 25 °C (1)	4.7 kΩ to 100 kΩ
Temperature measurement accuracy (between 25 °C and 85 °C)	± 0.5 °C
Climatic category	40/125/56
Maximum dissipation	250 mW
Dissipation factor δ (for information only)	7 mW/K
Response time (for information only) (2)	1.2 s
Thermal time constant τ (for information only)	11 s
Operating temperature range:	
at zero dissipation (continuously)	- 40 °C to + 125 °C
at maximum dissipation	0 °C to + 55 °C
Weight	≈ 0.22 g

Notes

(1) For values of nominal resistance value and tolerance at intermediate temperatures; see resistance values tables.

(2) Response time in silicone oil MS 200/50. This is the time needed for the sensor to reach 63.2 % of the total temperature difference when subjected to a temperature change from 25 °C in air to 85 °C in oil.

DESCRIPTION

These thermistors have a negative temperature coefficient. The device consists of a chip with two tin-plated copper leads. It is grey lacquered and not insulated. These thermistors are very accurate (± 0.5 °C) over a trajectory from 25 °C to 85 °C.

PACKAGING

The thermistors are packed in cardboard boxes, each box contains 500 units.

MARKING

Grey lacquered body.

MOUNTING

By soldering in any position.

ELECTRICAL DATA AND ORDERING INFORMATION

R_{25} (Ω)	$\Delta R_{25}/R_{25}$ (%)	R_{85} (Ω)	$\Delta R_{85}/R_{85}$ (%)	$B_{25/85}$ (K)	ΔB/B (%)	CATALOG NUMBER 2381 640	SAP MATERIAL NO. NTCLE101E3.....
4700	2.19	503.1	1.59	3977	0.75	10472	472SB0
10 000	2.19	1070	1.59	3977	0.75	10103	103SB0
47 000	2.23	4721	1.64	4090	1.5	10473	473SB0
100 000	2.29	9496	1.72	4190	1.5	10104	104SB0

Ireland esero
BCO The Space For Science
Science Foundation Ireland sfi For what's next
ceia.ie cork's technology network

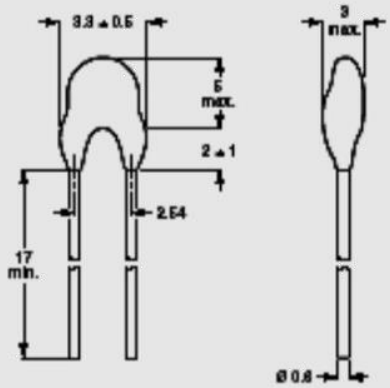
VISHAY.

2381 640 10.../NTCLE101E3...SB0

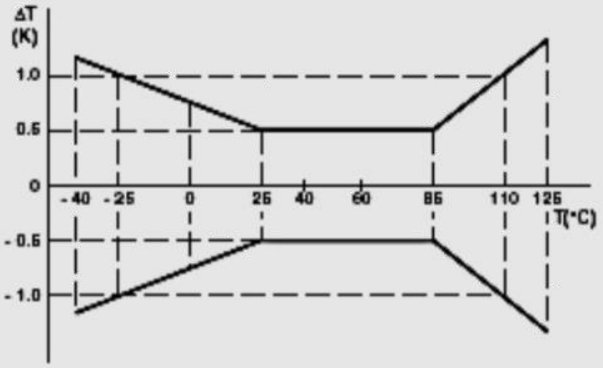
NTC Thermistors, Radial Leaded Special Accuracy

Vishay BCcomponents

DIMENSIONS in millimeters



TOLERANCE CURVE



RESISTANCE VALUES AT INTERMEDIATE VALUES with R_{25} at 4.7 kΩ and 10 kΩ

T_{oper} (°C)	R_T/R_{25}	TCR (%/K)	R_T (kΩ)	
			2381 640 10472 NTCLE101E3472SB0	2381 640 10103 NTCLE101E3103SB0
- 40	33.21	6.57	156.1	332.1000
- 35	23.99	6.36	112.8	240.0
- 30	17.52	6.15	82.35	175.2
- 25	12.93	5.95	60.77	129.3
- 20	9.636	5.76	45.30	96.36
- 15	7.250	5.58	34.08	72.50
- 10	5.505	5.40	25.87	55.05
- 5	4.216	5.24	19.81	42.16
0	3.255	5.09	15.30	32.56
5	2.534	4.92	11.91	25.34
10	1.997	4.79	9.390	19.97
15	1.570	4.64	7.378	15.70
20	1.249	4.50	5.869	12.49
25	1.000	4.37	4.700	10.00
30	0.8059	4.25	3.788	8.059
35	0.6535	4.13	3.072	6.535
40	0.5330	4.02	2.505	5.330
45	0.4372	3.91	2.055	4.372
50	0.3605	3.80	1.694	3.606
55	0.2989	3.70	1.405	2.989
60	0.2490	3.60	1.170	2.490
65	0.2084	3.51	0.9797	2.084
70	0.1753	3.42	0.8239	1.753
75	0.1481	3.33	0.6960	1.481
80	0.1256	3.25	0.5905	1.256
85	0.1070	3.16	0.5031	1.070
90	0.09154	3.09	0.4303	0.9154
95	0.07860	3.01	0.3694	0.7860
100	0.06773	2.94	0.3183	0.6773
105	0.05858	2.87	0.2753	0.5858
110	0.05093	2.80	0.2399	0.5093
115	0.04426	2.73	0.2090	0.4426
120	0.03866	2.67	0.1817	0.3866
125	0.03397	2.61	0.1592	0.3397
130	0.02977	2.55	0.1399	0.2977
135	0.02624	2.49	0.1233	0.2624
140	0.02319	2.43	0.1090	0.2319
145	0.02055	2.38	0.0966	0.2055
150	0.01826	2.33	0.0859	0.1826

Ireland esero
BCO The Space For Science
Science Foundation Ireland sfi For what's next
ceia.ie cork's technology network

2381 640 10.../NTCLE101E3...SB0

Vishay BCcomponents

NTC Thermistors, Radial
Leaded Special Accuracy

VISHAY.

RESISTANCE VALUES AT INTERMEDIATE VALUES with R_{25} at 47 kΩ

T_{oper} (°C)	R_T/R_{25}	TCR (%/K)	R_T (kΩ)
			2381 640 10473 NTCLE101E3473SB0
- 40	33.81	6.55	1589
- 35	24.50	6.34	1151
- 30	17.98	6.15	842.8
- 25	13.25	5.98	622.8
- 20	9.875	5.78	464.1
- 15	7.425	5.61	349.0
- 10	5.630	5.45	264.6
- 5	4.304	5.29	202.3
0	3.315	5.14	155.8
5	2.578	4.99	120.9
10	2.011	4.85	94.53
15	1.583	4.72	74.40
20	1.254	4.59	58.95
25	1.000	4.48	47.00
30	0.8024	4.34	37.71
35	0.6474	4.23	30.43
40	0.5255	4.12	24.70
45	0.4288	4.01	20.15
50	0.3518	3.91	16.53
55	0.2901	3.81	13.63
60	0.2408	3.71	11.30
65	0.2001	3.62	9.404
70	0.1674	3.53	7.865
75	0.1408	3.44	6.617
80	0.1186	3.36	5.573
85	0.1004	3.28	4.721
90	0.08542	3.20	4.015
95	0.07292	3.13	3.427
100	0.06248	3.06	2.936
105	0.05372	2.98	2.525
110	0.04635	2.92	2.179
115	0.04018	2.85	1.888
120	0.03495	2.79	1.638
125	0.03057	2.73	1.427
130	0.02654	2.67	1.247
135	0.02329	2.61	1.093
140	0.02044	2.55	0.9608
145	0.01802	2.50	0.8468
150	0.01592	2.44	0.7483

Ireland esero
BCO The Space For Science
Science Foundation Ireland sfi For what's next
ceia.ie cork's technology network

RESISTANCE VALUES AT INTERMEDIATE VALUES with R_{25} at 100 kΩ

T_{oper} (°C)	R_T/R_{25}	TCR (%/K)	R_T (kΩ)
			2381 640 10104 NTCLE101E3104SB0
- 40	36.66	6.70	3666
- 35	26.38	6.49	2638
- 30	19.17	6.29	1917
- 25	14.06	6.10	1406
- 20	10.41	5.92	1041
- 15	7.779	5.74	777.9
- 10	5.861	5.57	586.1
- 5	4.453	5.41	445.3
0	3.409	5.26	340.9
5	2.631	5.11	263.1
10	2.044	4.97	204.4
15	1.600	4.83	160.0
20	1.261	4.70	126.1
25	1.000	4.57	100.0
30	0.7981	4.45	79.81
35	0.6408	4.35	64.08
40	0.5175	4.22	51.74
45	0.4202	4.11	42.02
50	0.3431	4.00	34.31
55	0.2818	3.90	28.18
60	0.2322	3.80	23.22
65	0.1925	3.71	19.25
70	0.1602	3.62	16.03
75	0.1340	3.53	13.40
80	0.1126	3.45	11.26
85	0.09498	3.36	9.498
90	0.08042	3.28	8.042
95	0.06837	3.21	6.837
100	0.05835	3.13	5.835
105	0.04998	3.06	4.998
110	0.04296	2.99	4.296
115	0.03705	2.92	3.705
120	0.03206	2.86	3.206
125	0.02783	2.80	2.783

Ireland esero

BCO The Space For Science

Science Foundation Ireland sfi For what's next

ceia.ie cork's technology network

VISHAY.

Legal Disclaimer Notice

Vishay

Disclaimer

All product specifications and data are subject to change without notice.

Vishay Intertechnology, Inc., its affiliates, agents, and employees, and all persons acting on its or their behalf (collectively, "Vishay"), disclaim any and all liability for any errors, inaccuracies or incompleteness contained herein or in any other disclosure relating to any product.

Vishay disclaims any and all liability arising out of the use or application of any product described herein or of any information provided herein to the maximum extent permitted by law. The product specifications do not expand or otherwise modify Vishay's terms and conditions of purchase, including but not limited to the warranty expressed therein, which apply to these products.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document or by any conduct of Vishay.

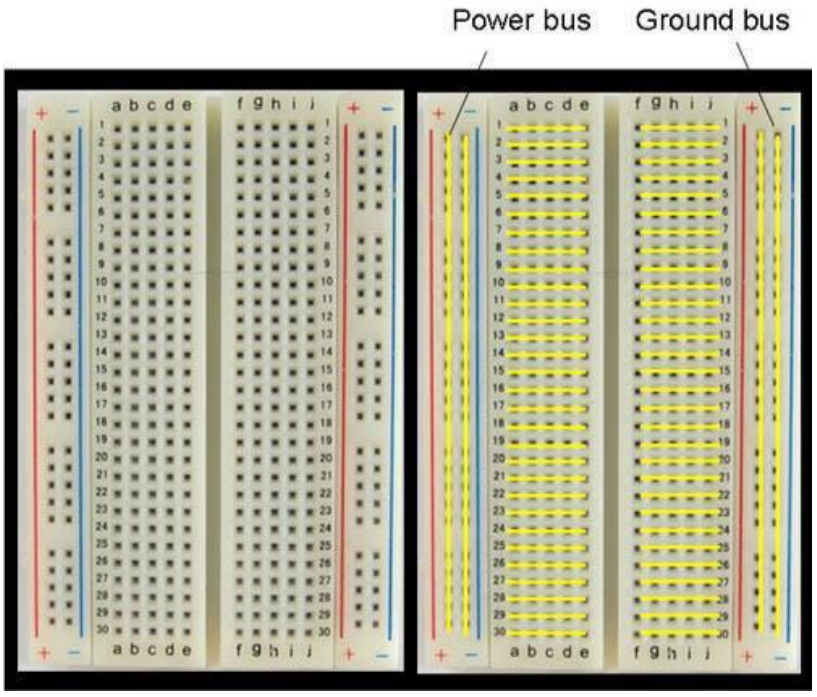
The products shown herein are not designed for use in medical, life-saving, or life-sustaining applications unless otherwise expressly indicated. Customers using or selling Vishay products not expressly indicated for use in such applications do so entirely at their own risk and agree to fully indemnify Vishay for any damages arising or resulting from such use or sale. Please contact authorized Vishay personnel to obtain written terms and conditions regarding products designed for such applications.

Product names and markings noted herein may be trademarks of their respective owners.

Ireland esero
BCO The Space For Science
Science Foundation Ireland sfi For what's next
ceia.ie cork's technology network

Appendix 3 – Solderless Breadboards

A *breadboard* is a rectangular plastic box filled with holes, which have contacts in which you can insert electronic components and wires. A breadboard is what you use to build a temporary version of your circuit. You don't have to solder wires or anything else; instead, you insert your components and wires into the little contact holes arranged in rows and connected by lines of metal; then you can connect your components together with wires to form your circuit.



Left: *Picture of holes on breadboard on left.* ***Right:*** *How holes are connected*

The yellow lines on the image on the right show how the sockets are connected. You can see that the vertical columns of holes labelled with "+" are connected to each other, as are the columns of holes labelled with "-"

The columns labelled with "+" are called the power bus, and you will connect one of them to a positive input voltage, such as the positive terminal of a 9V battery. One of the columns labelled with "-" (the ground bus) will be attached to the negative terminal of the battery.

Note that in each row (numbered 1 through 30) sockets "a" to "e" are connected to each other. And "f" to "j" are also connected to each other.

Appendix 4 – MPX4115 Pressure sensor data sheet

Freescale Semiconductor
Technical Data

Document Number: MPX4115
Rev 5, 08/2006

Integrated Silicon Pressure Sensor Altimeter/Barometer Pressure Sensor On-Chip Signal Conditioned, Temperature Compensated and Calibrated

MPX4115 SERIES

OPERATING OVERVIEW
INTEGRATED PRESSURE SENSOR
15 to 115kPa (2.18 to 16.7 psi)
0.2 to 4.8 Volts Output

The MPX4115 series is designed to sense absolute air pressure in an altimeter or barometer (BAP) applications. Freescale's BAP sensor integrates on-chip, bipolar op amp circuitry and thin film resistor networks to provide a high level analog output signal and temperature compensation. The small form factor and high reliability of on-chip integration makes the Freescale BAP sensor a logical and economical choice for application designers.

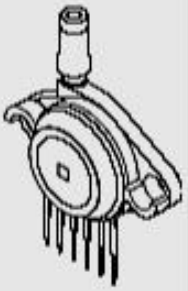
Features

- 1.5% Maximum Error over 0° to 85°
- Ideally suited for Microprocessor or Microcontroller-Based Systems
- Available in Absolute, Differential and Gauge Configurations
- Durable Epoxy Unibody Element
- Easy-to-Use Chip Carrier Option

Typical Applications

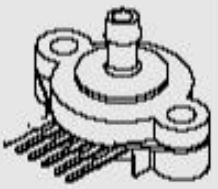
- Altimeter
- Baromete

MPX4115A
CASE 867-08



MPX4115AP
CASE 867B-04

MPX4115AS
CASE 867E-03



MPX4115ASX
CASE 867F-03

ORDERING INFORMATION[1]

Device	Options	Case No.	MPX Series Order No.	Marking
Basic Element	Absolute, Element Only	Case 867-08	MPX4115A	MPX4115A
Ported Elements	Absolute, Ported	Case 867B-04	MPX4115AP	MPX4115AP
	Absolute, Stove Pipe Port	Case 867E-03	MPX4115AS	MPX4115A
	Absolute, Axial Port	Case 867F-03	MPX4115ASX	MPX4115A

1. The MPX4115A BAP Sensor is available in the Basic Element package or with pressure port fittings that provide mounting ease and barbed hose connections.

PIN NUMBERS

1	V_{OUT}[1]	4	N/C[2]
2	GND	5	N/C[2]
3	V_S	6	N/C[2]

1. Pin 1 is noted by the notch in the lead.
2. Pins 4, 5, and 6 are internal device connections. Pin 1 is noted by the notch in the Lead. Do not connect to external circuitry or ground.

© Freescale Semiconductor, Inc., 2006. All rights reserved.

freescale semiconductor

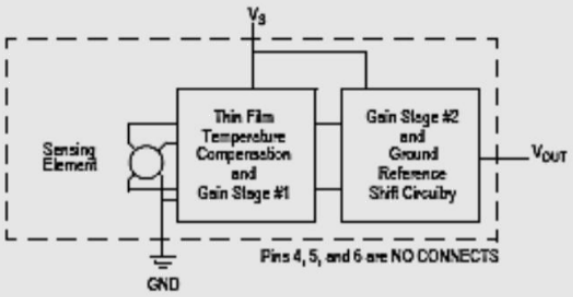


Figure 1. Integrated Pressure Sensor Schematic

Table 1. Maximum Ratings[1]

Parametrics	Symbol	Value	Unit
Overpressure[2] (P1 > P2)	P_{max}	400	kPa
Burst Pressure[2] (P1 > P2)	P_{burst}	1000	kPa
Storage Temperature	T_{stg}	-40° to +125°	°C
Operating Temperature	T_A	-40° to +125°	°C

1. T_C = 25°C unless otherwise noted.
2. Exposure beyond the specified limits may cause permanent damage or degradation to the device.

Ireland esero
BCO The Space For Science
Science Foundation Ireland sfi For what's next
ceia.ie cork's technology network

Table 2. Operating Characteristics
(V_S = 5.1 Vdc, T_A = 25°C unless otherwise noted, P1 > P2 Decoupling circuit shown in Figure 3 required to meet electrical specifications.)

Characteristic		Symbol	Min	Typ	Max	Unit
Pressure Range[1]		P_{OP}	15	-	115	kPa
Supply Voltage[2]		V_S	4.85	5.1	5.35	Vdc
Supply Current		I_o	—	7.0	10	mAdc
Minimum Pressure Offset[3] @ V_S = 5.1 Volts	(0 to 85°C)	V_{off}	0.135	0.204	0.273	Vdc
Full Scale Output[4] @ V_S = 5.1 Volts	(0 to 85°C)	V_{FSO}	4.725	4.794	4.863	Vdc
Full Scale Span[5] @ V_S = 5.1 Volts	(0 to 85°C)	V_{FSS}	—	4.59	—	Vdc
Accuracy[6]	(0 to 85°C)	—	—	—	±1.5	$\%V_{FSS}$
Sensitivity		V/P	—	46	—	mV/kPa
Response Time[7]		t_R	—	1.0	—	ms
Output Source Current at Full Scale Output		I_o+	—	0.1	—	mAdc
Warm-Up Time[8]		—	—	20	—	mSec
Offset Stability[9]		—	—	±0.5	—	$\%V_{FSS}$

1. 1.0kPa (kiloPascal) equals 0.145 psi.
2. Device is ratiometric within this specified excitation range.
3. Offset (V_{off}) is defined as the output voltage at the minimum rated pressure.
4. Full Scale Output (V_{FSO}) is defined as the output voltage at the maximum or full rated pressure.
5. Full Scale Span (V_{FSS}) is defined as the algebraic difference between the output voltage at full rated pressure and the output voltage at the minimum rated pressure.
6. Accuracy (error budget) consists of the following:
   Linearity: Output deviation from a straight line relationship with pressure, using end point method, over the specified pressure range.
   Temperature Hysteresis: Output deviation at any temperature within the operating temperature range, after the temperature is cycled to and from the minimum or maximum operating temperature points, with zero differential pressure applied.
   Pressure Hysteresis: Output deviation at any pressure within the specified range, when this pressure is cycled to and from the minimum or maximum rated pressure at 25°C.
   TcSpan: Output deviation over the temperature range of 0° to 85°C, relative to 25°C.
   TcOffset: Output deviation with minimum pressure applied, over the temperature range of 0° to 85°C, relative to 25°C.
   Variation from Nominal: The variation from nominal values, for Offset or Full Scale Span, as a percent of V_{FSS} at 25°C.
7. Response Time is defined as the time for the incremental change in the output to go from 10% to 90% of its final value when subjected to a specified step change in pressure.
8. Warm-up is defined as the time required for the product to meet the specified output voltage after the Pressure has been stabilized.
9. Offset stability is the product's output deviation when subjected to 1000 hours of Pulsed Pressure, Temperature Cycling with Bias Test.

Table 3. Mechanical Characteristics

Characteristic	Symbol	Min	Typ	Max	Unit
Weight, Basic Element (Case 867)	—	—	4.0	—	Grams
Common Mode Line Pressure[1]	—	—	—	690	kPa

1. Common mode pressures beyond what is specified may result in leakage at the case-to-lead interface.

Ireland esero
BCO The Space For Science
Science Foundation Ireland sfi For what's next
ceia.ie cork's technology network

Figure 2 illustrates the absolute sensing chip in the basic chip carrier (Case 867). A fluorosilicone gel isolates the die surface and wire bonds from the environment, while allowing the pressure signal to be transmitted to the sensor diaphragm. The MPX4115A series pressure sensor operating characteristics, and internal reliability and qualification tests are based on use of dry air as the pressure media. Media, other than dry air, may have adverse effects on sensor performance and long-term reliability. Contact the factory for information regarding media compatibility in your application.

Figure 4 shows the sensor output signal relative to pressure input. Typical, minimum, and maximum output curves are shown for operation over a temperature range of 0° to 85°C. (The output will saturate outside of the specified pressure range.)

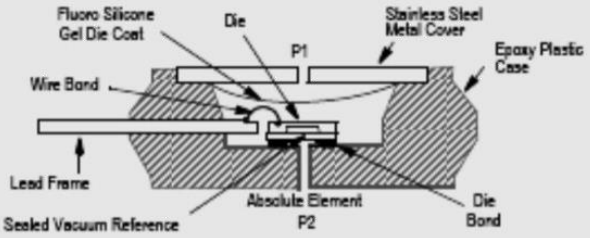


Figure 2. Cross-Sectional Diagram (Not to Scale)

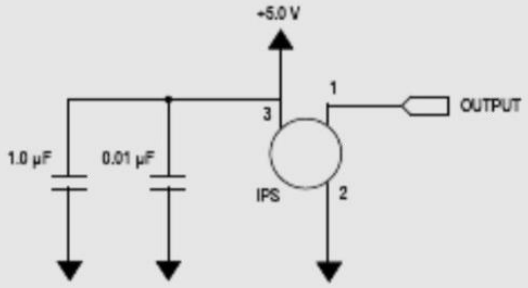


Figure 3. Recommended Power Supply Decoupling.
(For output filtering recommendations, please refer to Application Note AN1646.)

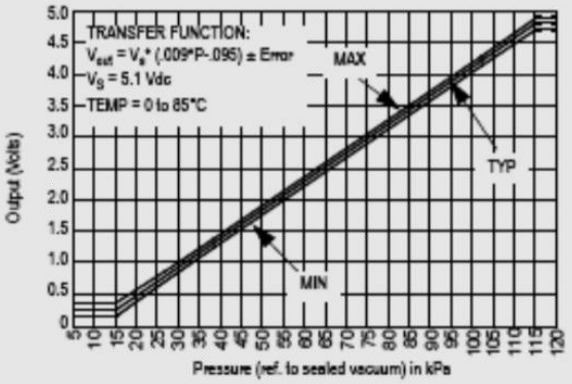


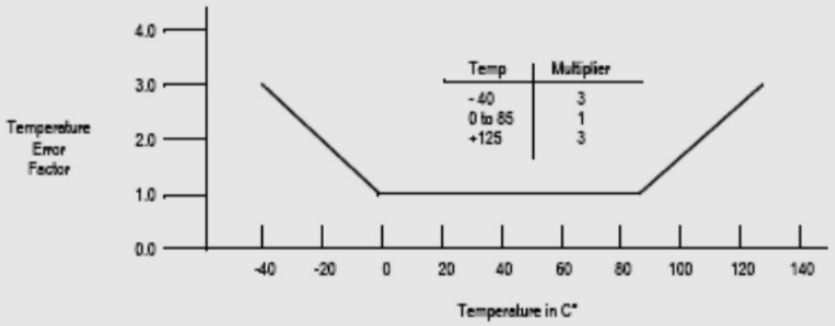
Figure 4. Output versus Absolute Pressure

Transfer Function (MPX4115)

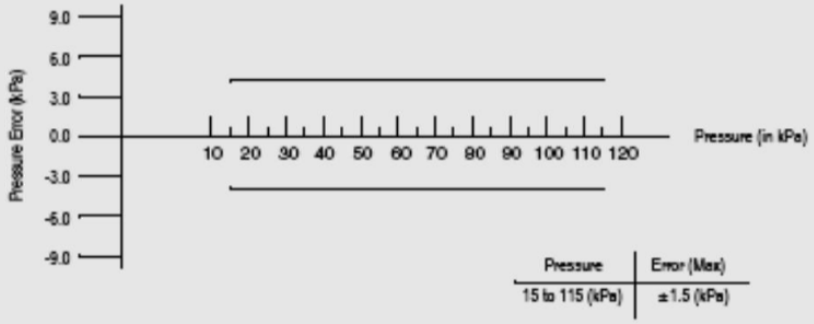
Nominal Transfer Value: $V_{out} = V_S (P \times 0.009 - 0.095)$
$\pm$ (Pressure Error x Temp. Factor x 0.009 x V_S)
$V_S = 5.1\ V \pm 0.25\ Vdc$

Temperature Error Band

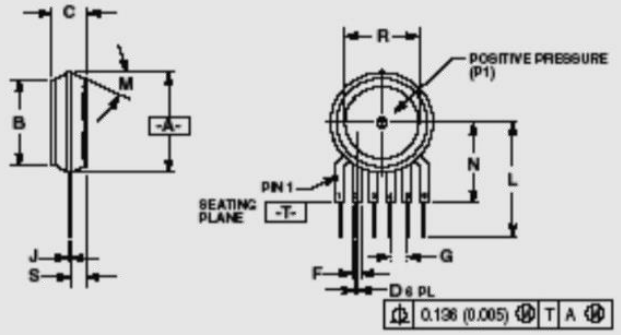
MPX4115A Series



Pressure Error Band



PACKAGE DIMENSIONS



NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.
3. DIMENSION -A- IS INCLUSIVE OF THE MOLD STOP RING. MOLD STOP RING NOT TO EXCEED 16.00 (0.630).

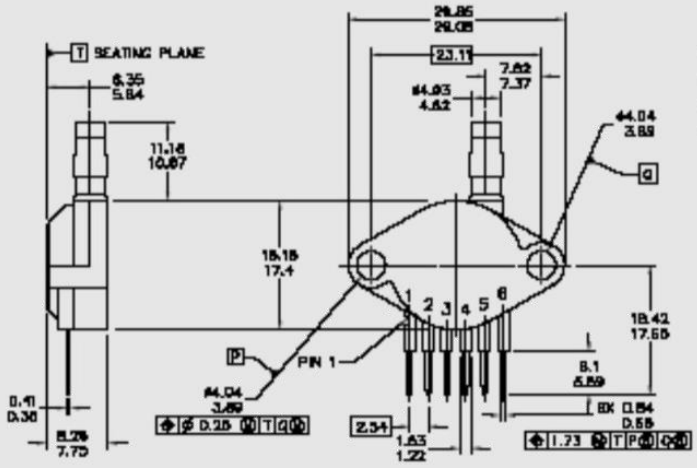
DIM	INCHES		MILLIMETERS	
	MIN	MAX	MIN	MAX
A	0.595	0.630	15.11	16.00
B	0.514	0.534	13.06	13.56
C	0.200	0.220	5.08	5.59
D	0.027	0.033	0.68	0.84
F	0.048	0.064	1.22	1.63
G	0.100 BSC		2.54 BSC	
J	0.014	0.016	0.36	0.40
L	0.695	0.725	17.65	18.42
M	30° NOM		30° NOM	
N	0.475	0.495	12.07	12.57
R	0.430	0.450	10.92	11.43
S	0.090	0.105	2.29	2.66

STYLE 1:
PIN 1. VOUT
2. GROUND
3. VCC
4. V1
5. V2
6. VEX

STYLE 2:
PIN 1. OPEN
2. GROUND
3. -VOUT
4. VSUPPLY
5. +VOUT
6. OPEN

STYLE 3:
PIN 1. OPEN
2. GROUND
3. +VOUT
4. +VSUPPLY
5. -VOUT
6. OPEN

CASE 867-08
ISSUE N
BASIC ELEMENT (A, D)



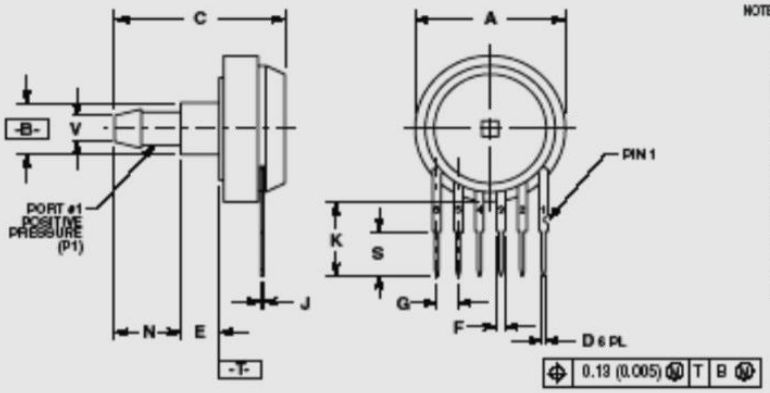
NOTES:
1. DIMENSIONS ARE IN MILLIMETERS.
2. DIMENSIONS AND TOLERANCES PER ASME Y14.5M–1994.
3. 867B–01 THRU –3 OBSOLETE, NEW STANDARD 867B–04.

STYLE 1:
PIN 1: V OUT
2. GROUND
3. VCC
4. V1
5. V2
6. V EX

CASE 867B-04
ISSUE G
PRESSURE SIDE PORTED (AP, GP)

Ireland esero
BCO The Space For Science
Science Foundation Ireland sfi For what's next
ceia.ie cork's technology network

PACKAGE DIMENSIONS

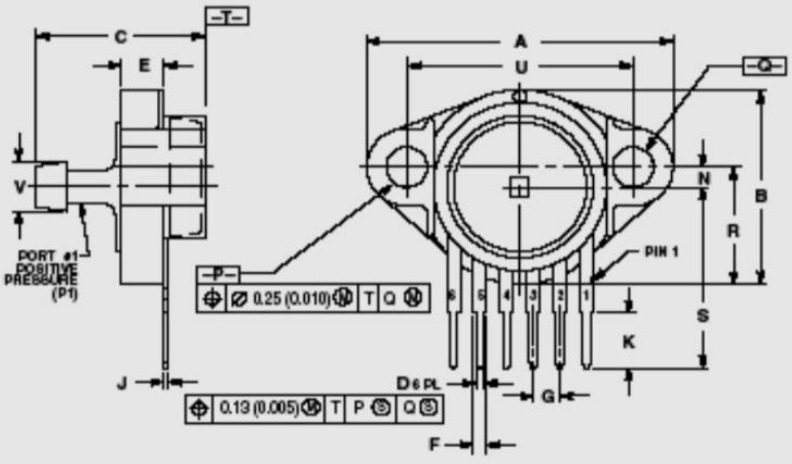


NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.

DIM	INCHES MIN	INCHES MAX	MILLIMETERS MIN	MILLIMETERS MAX
A	0.690	0.720	17.53	18.28
B	0.245	0.255	6.22	6.48
C	0.780	0.800	19.81	20.32
D	0.027	0.033	0.69	0.84
E	0.178	0.198	4.52	4.72
F	0.048	0.064	1.22	1.63
G	0.100 BSC		2.54 BSC	
J	0.014	0.016	0.36	0.41
K	0.345	0.375	8.76	9.53
N	0.300	0.310	7.62	7.87
S	0.220	0.240	5.59	6.10
V	0.182	0.194	4.62	4.93

STYLE 1:
PIN 1. V_{OUT}
2. GROUND
3. V_{CC}
4. V1
5. V2
6. V_{EX}

CASE 867E-03
ISSUE D
PRESSURE SIDE PORTED (AS, GS)



NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.

DIM	INCHES MIN	INCHES MAX	MILLIMETERS MIN	MILLIMETERS MAX
A	1.080	1.120	27.43	28.45
B	0.740	0.760	18.80	19.30
C	0.630	0.650	16.00	16.51
D	0.027	0.033	0.69	0.84
E	0.160	0.180	4.06	4.57
F	0.048	0.064	1.22	1.63
G	0.100 BSC		2.54 BSC	
J	0.014	0.016	0.36	0.41
K	0.220	0.240	5.59	6.10
N	0.070	0.090	1.78	2.29
P	0.150	0.160	3.81	4.06
Q	0.150	0.160	3.81	4.06
R	0.440	0.460	11.18	11.68
S	0.695	0.725	17.65	18.42
U	0.840	0.860	21.34	21.84
V	0.182	0.194	4.62	4.93

STYLE 1:
PIN 1. V_{OUT}
2. GROUND
3. V_{CC}
4. V1
5. V2
6. V_{EX}

CASE 867F-03
ISSUE D
PRESSURE SIDE PORTED (ASX, GSX)

Ireland esero
BCO The Space For Science
Science Foundation Ireland sfi For what's next
ceia.ie cork's technology network

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2006. All rights reserved.

freescale semiconductor

MPX4115
Rev. 5
08/2006

Ireland esero
BCO The Space For Science
Science Foundation Ireland sfi For what's next
ceia.ie cork's technology network

Appendix 5 – APC220 wireless telemetry data sheet

APPCON TECHNOLOGIES

SHENZHEN APPCON TECHNOLOGIES CO.LTD DVER 1.20

APC Series Transparent Transceiver Module

APC220-43

Product Overview:

APC220-43 is highly integrated semi-duplex low power transceiver module withhigh speed MCU and high capability RF IC. Using high efficiency forward error correction with Interleaving encoding technology , it can make anti-interference and sensitivity improved highly. It can have a good performance in strong interference circumstance as well, for example the industry field. The technique is advanced in data transfers area.

APC220-43 is a cost-effective and easy applied module that not only can transmit transparent data with large data buffer zone,but also can provide more than 100 channels . It 's parameters easily setting and small size make the module an ideal for wireless data transfer application.

Application:

- Automated Meter Reading (AMR)
- Wireless sensor
- Industrial Automation
- The control of traffic signal
- Wireless handheld terminal
- Remote control and monitoring
- The management of cars
- Wire Replacement
- Oil and Gas Detection.
- The control of robot

Ireland esero BCO The Space For Science Science Foundation Ireland sfi For what's next ceia.ie cork's technology network

APPCON TECHNOLOGIES APC220-43

Characters:

- 1000 meters of communication distance（2400bps）
- Output power is 20 mW
- Frequency is from 418MHz to 455MHz
- Size of Module 37.5mm x 18.3mm x 7.0mm
- More than 100 channels
- GFSK modulation
- UART/TTL interface
- Exceed 256 bytes data buffer
- fit to large data transfers
- The convenient software for setting

Installation and Use

APC220-43 module has 9 pins. Refers to the Table 1:

APC220-43		
Pin NO.	Pin Name	Description
1	GND	Grounding of Power Supply
2	VCC	Power supply DC 3.5V-5.5V
3	EN	Power enable, ≥1.6V or empty, ≤0.5Vsleep.
4	RXD	URAT input, TTL
5	TXD	URAT output, TTL
6	MUX	The pin is expanded for other functions
7	SET	Setting parameters, setting online supportted
8	NC	Not connected
9	NC	Not connected

Table 1 Interface definition

Ireland esero BCO The Space For Science Science Foundation Ireland sfi For what's next ceia.ie cork's technology network

APPCON TECHNOLOGIES APC220-43

Dimensions – PIN Assignments

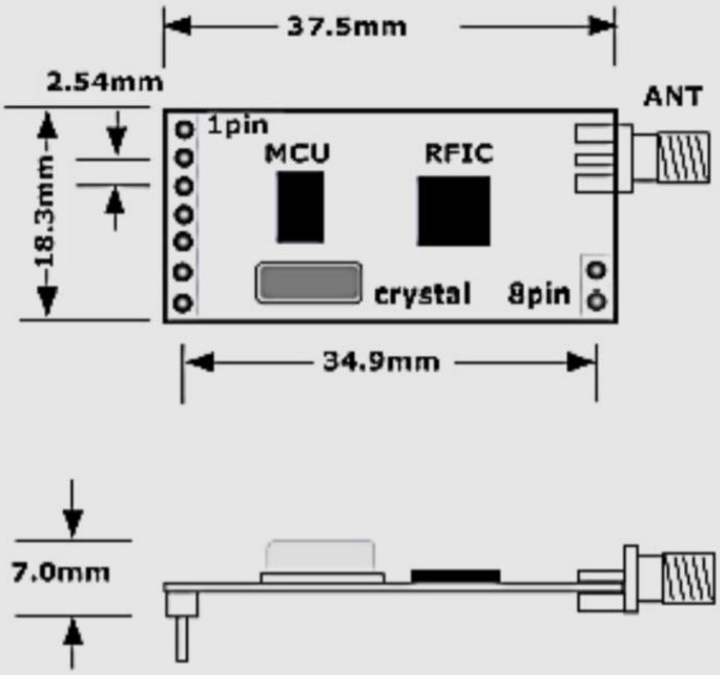


Figure 1: Size of Module

Setup parameters

With series COM or through the software Rf-Magic ,user can set up all parameters which include work frequency, UART rate, air rate,checkout mode and so on.

It is very convenient to set APC220-43 . Different options can be selected based on user needs . Please refer to Table 2 and Figure 2.

The instruction of setting parameters of module APC220-43		
Setting	options	default
UART rate	1200,2400,4800,9600b,19200,38400,57600	9600bps
Series Parity	Disable,Even Parity,Odd Parity	Disable
Frequency	418MHz-455MHz	434 MHz
Air Rate	2400bps,4800bps,9600bps,19200bps	9600bps
RF Power	0-9(9 for 20mw)	9(20mw)

Table 2 Setting Parameters of Modules

Ireland esero BCO The Space For Science Science Foundation Ireland sfi For what's next ceia.ie cork's technology network

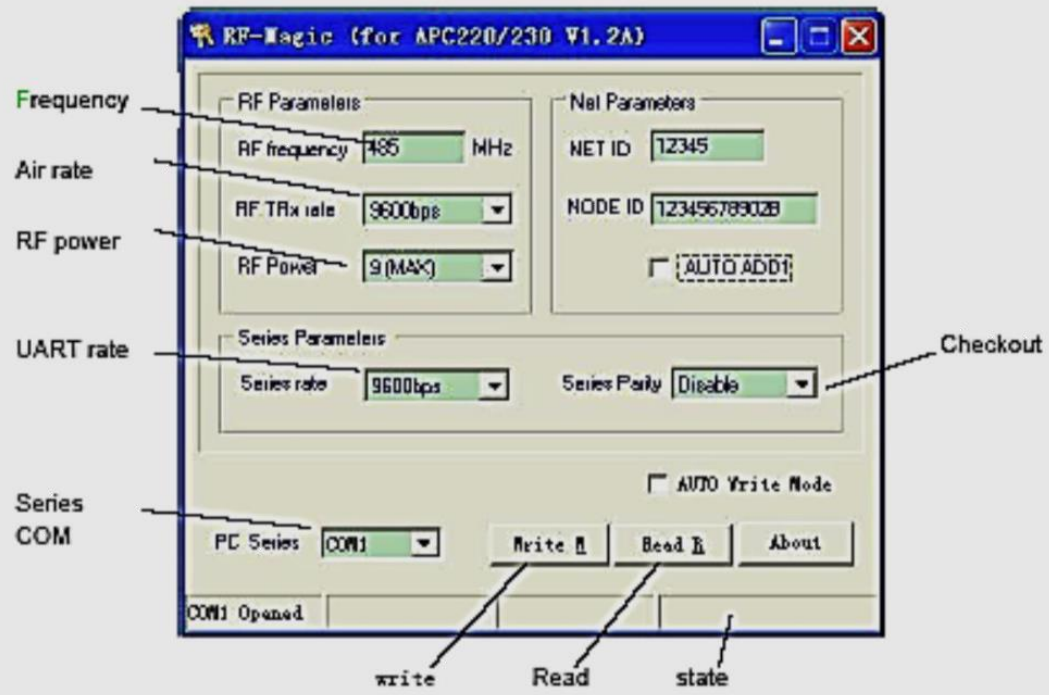


Figure 2 the software of Rf-Magic

There are two ways of setting the parameters of APC220-43. One way is to use the Rf-Magic by PC. Please look at the Figure 3

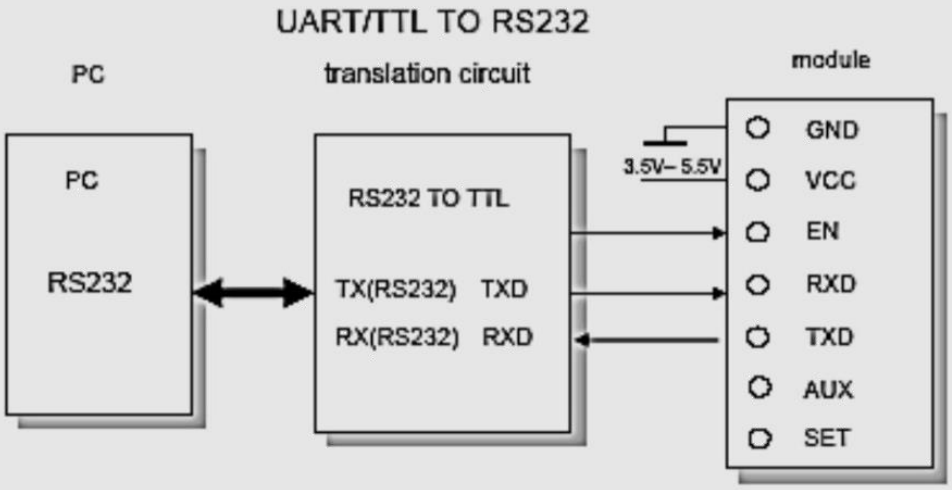


Figure 3 the connection of setting parameters

Ireland esero BCO The Space For Science Science Foundation Ireland sfi For what's next ceia.ie cork's technology network

To set the parameters,It needs a UART/TTL to RS232 interface board to connect APC220-43 with PC.At first, connect the APC220-43 and PC by interface board . Then run the software Rf-Magic and plug APC220-43 into the interface board with power supply +5V .After that ,User will see "Found Device" underside of Rf-Magic, and user can write the parameters which user selected.

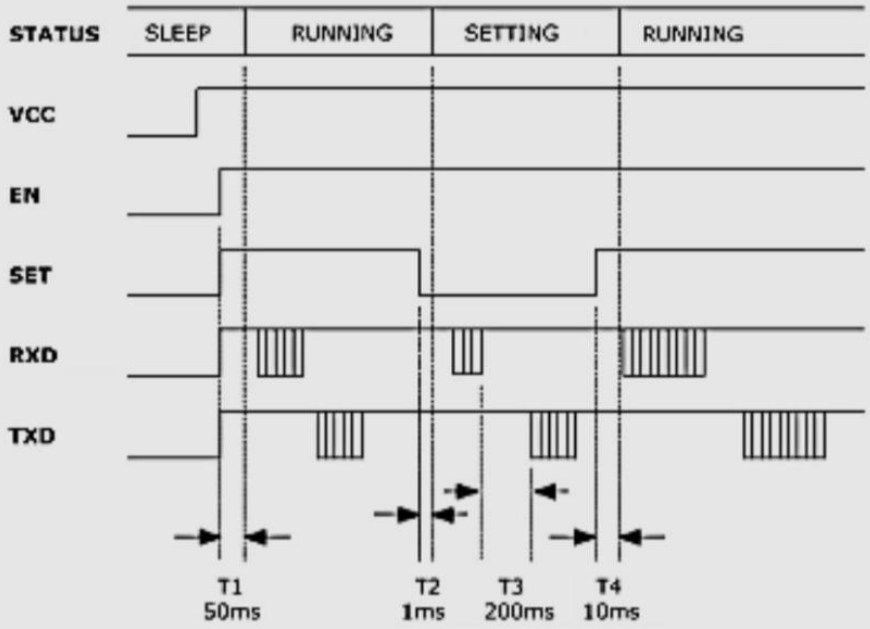


Figure 4 the figure of Setting Parameters Online

The other way of setting parameters is to use terminal on line. The parameters are setted by TTL/UART (4,5PIN) and the control pin of SET (see the Figure 5). APC220-43 will enter normal working (T1, see the Figure 4) mode after the voltage of SET 50ms later. If user wants to set parameters online, you can lower the voltage of SET firstly .Then APC220-43 will set the UART data rate at 9600bps automatically and enter setting mode (T2) with no checkout 1ms later. User should send the parameters command by RXD.After parameters checkout,the pin TXD will return the parameters information(T3) in 200ms. User may set up the voltage of SET after checking out the information that has been written.Finally, APC220-43 will work

Ireland esero BCO The Space For Science Science Foundation Ireland sfi For what's next ceia.ie cork's technology network

based on the new parameters in 10ms(T4). It must be noticed that user sent the command to APC220-43 for only one time when APC220-43 at the setting mode. If the command is wrong or the setting parameters are not completed, user should set them again. However,one more important thing that user must do firstly is to set up the voltage of the pin SET and enter the setting mode . It is the same as the way of the last setting.

APC220-43 is set by ACSII. UART rate is 9600bps and no checkout. There are two setting commands. They are reading and writing. It must use the capital letters. The parameters are parted by blank. And the enter means end.

The command of reading parameters:

RD↙

ANSWER: PARA_frequency_rf data rate_output power_UART data rate_series checkout

The command of writing parameters:

WR_frequency_ RF data rate_output power_UART rate_series_check↙

ANSWER: PARA_frequency_rf data rate_output power_UART data rate_series checkout

The Parameters Table:

The parameters table		
Parameters	Bytes	Instruction
Frequency	6	Unit is KHz,for example 434MHz is 434000.
Rf data rate	1	1,2,3 and 4 refer to 2400,4800,9600,19200bps separetely.
Output power	1	0 to 9, 9 means 13dBm(20mW).
UART rate	1	0,1,2,3,4,5 and 6 refers to 1200,2400,4800,9600, 19200,38400,57600bps separately.
Series checkout	1	Series checkout：0 means no check,1 means even parity,2 means odd parity.

Ireland esero BCO The Space For Science Science Foundation Ireland sfi For what's next ceia.ie cork's technology network

For example , one APC220-43 is set to 434MHZ; rf data rate is 9600bps; Output power is 20mW; UART data rate is 1200bps; No checkout.

WR_434000_3_9_0_0↙

(HEX code: 0x57,0x52,0x20,0x34,0x33,0x34,0x30,0x30,0x30,0x20,0x33,0x20,0x39, 0x20,0x30,0x20,0x30,0x0D,0x0A)

ANSWER: PARA_434000_3_9_0_0↙

(HEX code:0x50,0x41,0x52,0x41 0x20,0x34,0x33,0x34,0x30,0x30,0x30,0x20,0x33, 0x20,0x39,0x20,0x30,0x20,0x30,0x0D,0x0A)

The Connection between Module and Terminal(UART/TTL):

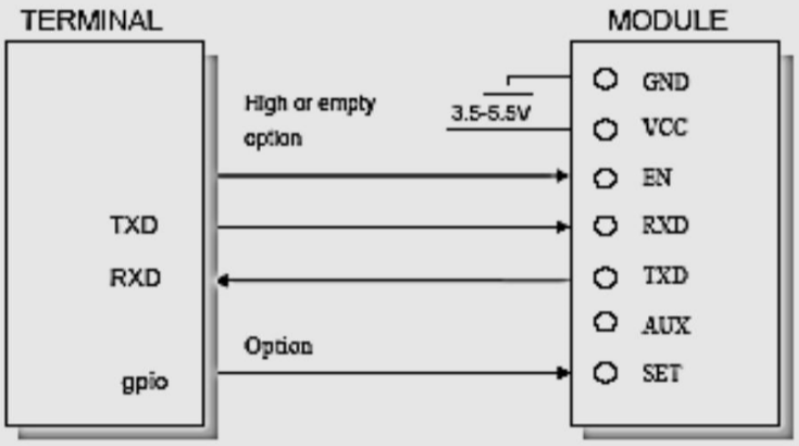


Figure 5:The Connection between Module and Terminal

Application of Constructing Networking (one point to multi-point):

APC220-43 is a semi-duplex module, which can be communicated by point to point or one point to multi-point. In the second mode, user needs to set one host module, while the others are client modules. Every module must only have one unique ID. The coordination of communication is controled by the host module, which sends datas or commands including ID. All the client modules can receive the data packets, and compare the ID with the own's. If they are the same, the module will

deal with the data packets. Otherwise, it will discard them. In order to avoid interfereing each other,all client modules must be at transmitting mode when the network is working. APC220-43 can set many different frequencies so that many networks can work in the same place and at the same time.

User should pay attention to the following questions based on the complex transfers in the air and some inherency characteristics of wireless communication:

1) The data delay of wireless communication

The wireless terminal receives some data ,or after waiting for a while to ensure no data any more, then there will be tens to hundreds milli-seconds delay from transfer to receiver (the exact delay based on the UART rate, air rate and the flow of data package). In addition, it also will cost some time to tranmsmit from module to terminal but the delay time is the same with the same condition.

2) The control of data flux

Although there is a buffer zone with 256 bytes in the wireless module, when the UART rate is higher than the air rate, there must be a problem about the data flux. It may cause to lose some data because the data overflow from the buffer. Under this condition, it must be ensured that the average UART rate is lower than 60 percent of the air rate . For instance, the UART rate is 9600bps, the air rate is 4800bps.If UART rate is the same as the air rate, the only way is to interval the transmitting time. If terminal transmits 100bytes to UART every time, it will take104ms every time. (104ms/0.6)*(9600/9600)=174ms. So when the interval time that terminal transmit 100bytes to the UART is higher than 174ms every time, those mentioned problems will be avoided.

3) The control of errors

The wireless network module has strong capability of anti-interference because of the high efficiency checking error correction with Interleaving encoding technology. However, when it is in a bad circumstance that has strong electric interference, the data may be lost or receive some error data. User can increase the development of the system link layer protocol. For instance, if user can increase TCP/IP slip window and repeat transmitting functions, it will improve the reliability and ability of wireless network

Ireland esero BCO The Space For Science Science Foundation Ireland sfi For what's next ceia.ie cork's technology network

communication.

4) Choice of antenna

Antenna is an very important element of the communication system. The quality of antenna impacts the capability of communication system. So user should think more about the quality of antenna.Generally speaking, it mainly contains two points : the kind of antenna (size) and its electric capability. The antenna must be matched with the frequency of communication system.

Specifications

The technical specifications of APC220-43:	
Work frequency	418MHz to 455MHz
Modulation	GFSK
Frequency interval	200KHz
Transmitted power	20mw (10 levels)
Received sensitivity	-113dBm@9600bps
Air rate	2400 - 19200bps
UART rate	1200 - 57600bps
The parity of series COM	8E1/8N1/8O1
The buffer of COM	256bytes
Humidity	10%~90%
Temperature	-30℃ - 85℃
Supply voltage	3.5 – 5.5V (the ripple is ±50mV)
Transmit current	≤42mA@20mW
Receiving current	≤28mA
Sleeping current	≤5uA
Transfers distance	1000m (open space)
Dimension	37.5mm x 18.3mm x 7.0mm

Ireland esero BCO The Space For Science Science Foundation Ireland sfi For what's next ceia.ie cork's technology network

Questions and Answers:

Questions and Answers	
Can not communicate between two devices	1. The communication protocol is different between two modules, for instance: data rate and checkout.
	2. The frequency or RF data rate is different between two communicated modules.
	3. They are not the same kind products.
	4. The connection between module and terminal is wrong.
	5. The module is wrong.
	6. The setting of EN is wrong.
	7. The communication distance exceeds the range, or the connection of antenna is bad.
Short communication distance	1. The supply voltage exceeds range
	2. The ripple of power is too big.
	3. The connection of antenna is bad or it is a wrong kind of antenna
	4. Antenna is too close to the surface of metal or the ground
	5. Receiving circumstance is very bad, for instance buildings and strong interference.
	6. There is interference of the same frequency
Receive wrong data	1. Wrong setting of COM, for example, Baud rate is wrong
	2. The connection of UART is wrong.
	3. The cable to the UART is too long.

Ireland esero BCO The Space For Science Science Foundation Ireland sfi For what's next ceia.ie cork's technology network

APPCON TECHNOLOGIES

APC220-43

Partner:

ANALOG DEVICES

Analog Dvices Inc

ADI Third Party Designer

APPCON TECHNOLOGIES

SHENZHEN APPCONTECHNOLOGIES CO.,LTD.

RMB1-B2,5F,112Building,JinDiindustry zone FuTian District ShenZhen(518048)

TEL:86-755-83405295

FAX:86-755-83405660

Email:appcon@126.com

http://www.appcon.com.cn

Ireland esero

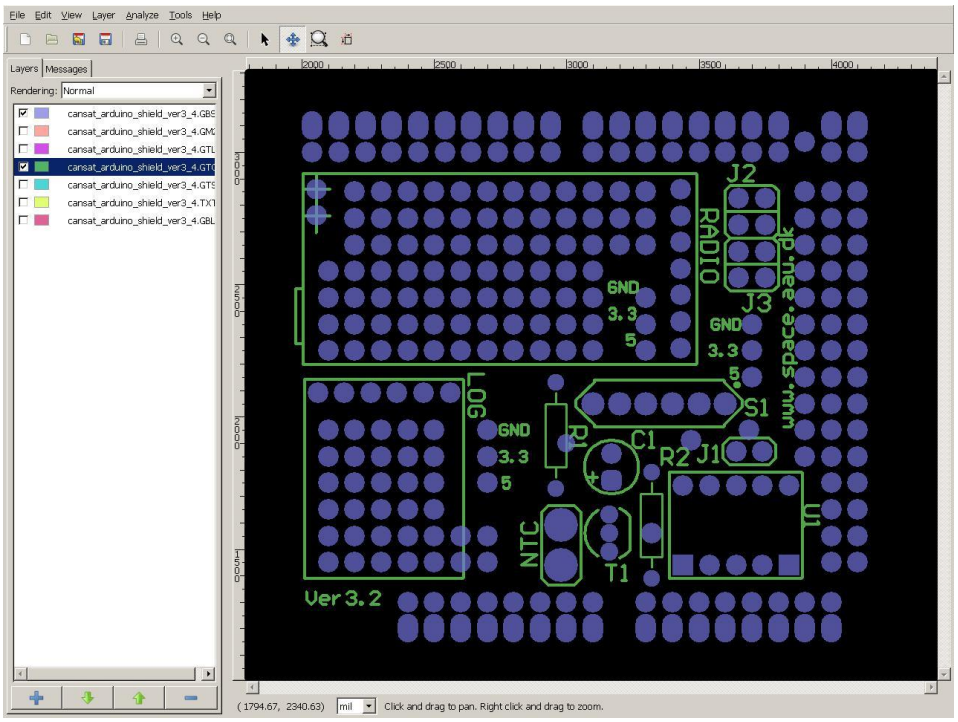
BCO The Space For Science

Science Foundation Ireland sfi For what's next

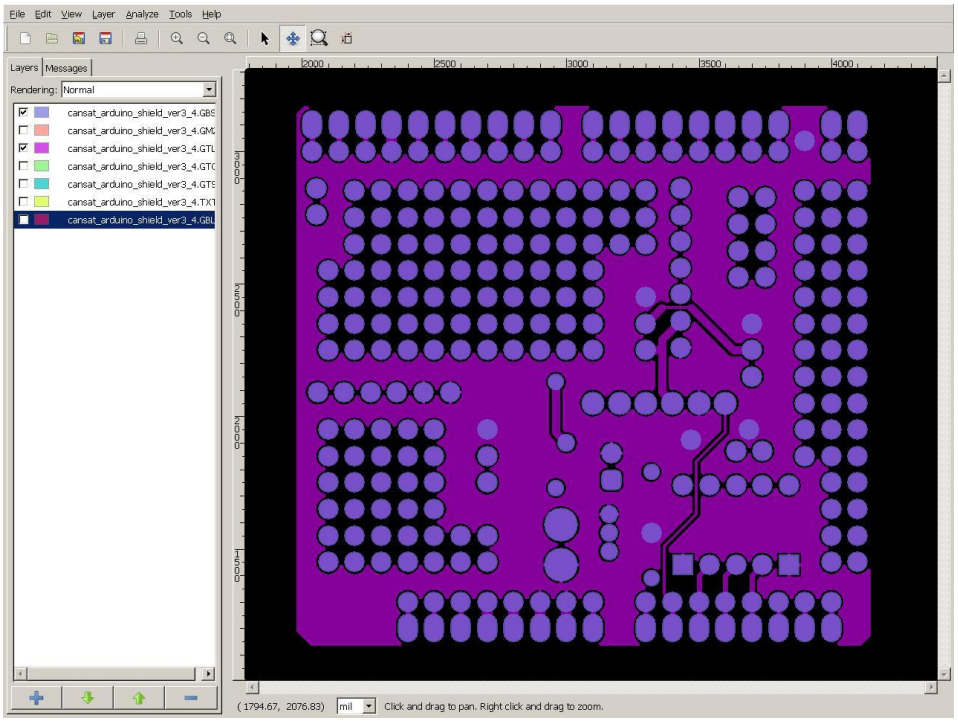
ceia.ie cork's technology network

Appendix 6 – AAU PCB sensor board

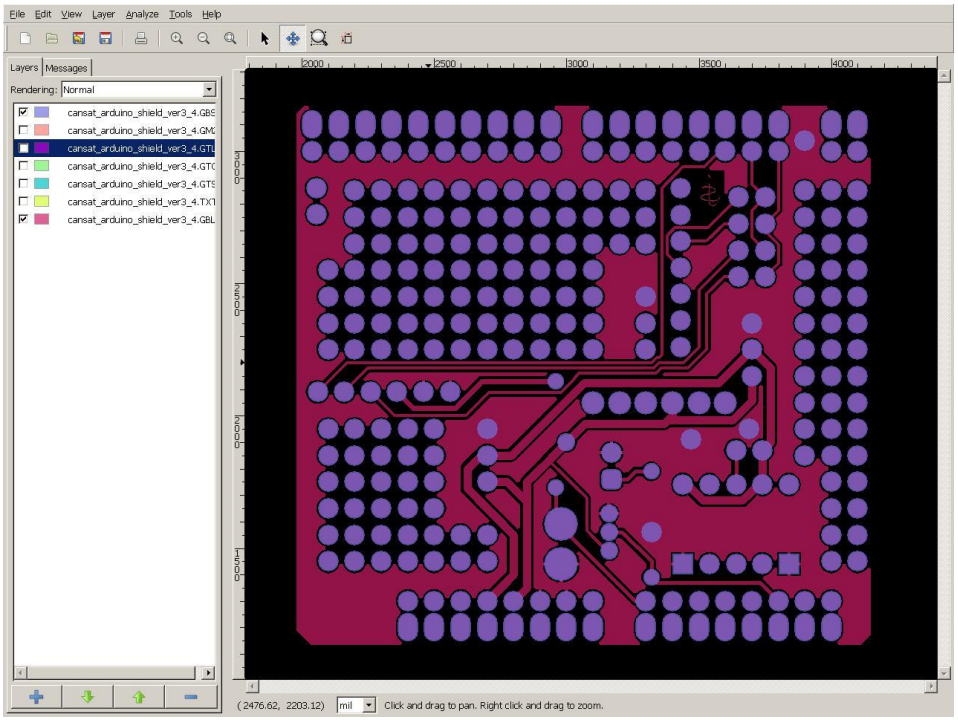
Layers 1 & 4



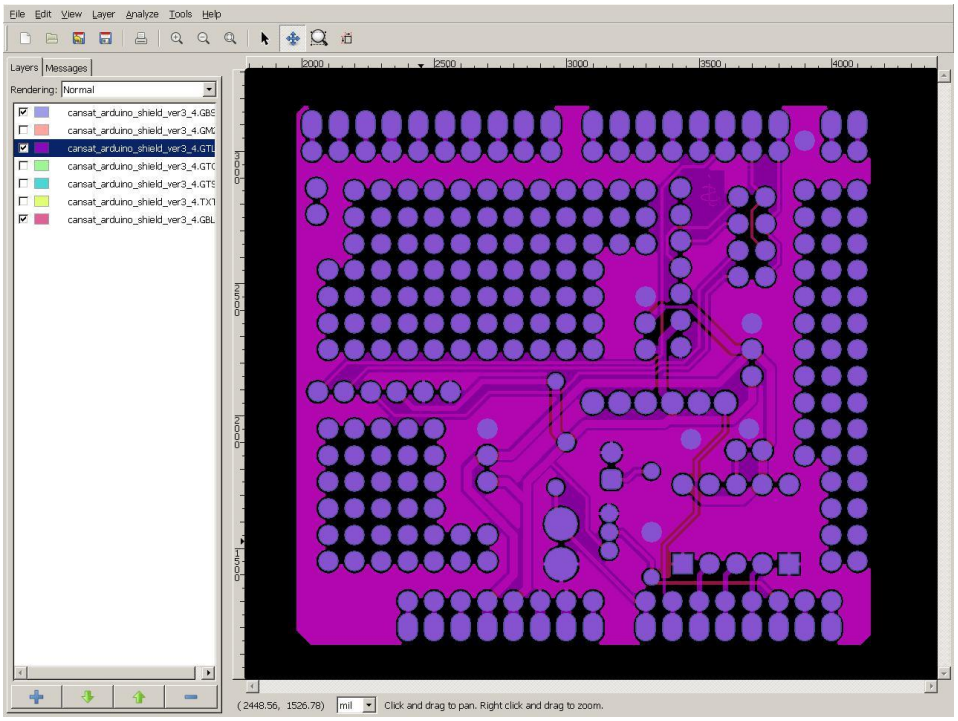
Layers 1 & 3



Layers 1 & 7



Layers 1, 3 & 7



Appendix 7 – Calculating Altitude from Pressure data

Atmospheric pressure varies with altitude according to the relation:

$$\mathbf{P = 101325\ (1 - 2.25577\ 10^{-5}\ h)^{5.25588}}$$

where **P** = air pressure (*Pa*) and **h** = altitude above sea level (*m*)

This equation can be re-arranged to obtain altitude:

$$\frac{\mathbf{P}}{\mathbf{101325}} = \mathbf{(1 - 2.25577\ 10 - 5\ h)^{5.25588}}$$

Taking logs of both sides:

$$log\left(\frac{\mathbf{P}}{\mathbf{101325}}\right) = \mathbf{5.25588}\ log(\mathbf{1 - 2.25577 \times 10^{-5}h})$$

Dividing by 5.25588:

$$\frac{log\left(\frac{\mathbf{P}}{\mathbf{101325}}\right)}{\mathbf{5.25588}} = log(\mathbf{1 - 2.25577 \times 10^{-5}h})$$

Taking anti-logs:

$$\mathbf{10}^{\left(\frac{log\left(\frac{\mathbf{P}}{\mathbf{101325}}\right)}{\mathbf{5.25588}}\right)} = \mathbf{1 - 2.25577 \times 10^{-5}h}$$

Solving for h:

$$\mathbf{h} = \frac{\mathbf{10}^{\left(\frac{log\left(\frac{\mathbf{P}}{\mathbf{101325}}\right)}{\mathbf{5.25588}}\right)} - \mathbf{1}}{\mathbf{-2.25577 \times 10^{-5}}}$$

To convert this to Arduino code, it's easiest to do it in steps (code on next page) :

Firstly, the value of **P** must be in *Pa* instead of *millibars*: **Pressure*100.0**	>>> **num1**
Next we calculate $\frac{\mathbf{P\ (Pa)}}{\mathbf{101325}}$: **num1/101325**	>>> **num2**
Calculate the log (base 10) of $\frac{\mathbf{P\ (Pa)}}{\mathbf{101325}}$: **log10 (num2)**	>>> **num3**
Divide this number by 5.25588 : **num3/5.25588**	>>> **num4**
Calculate the anti-log & subtract 1 : **POW(10.0, num4) -1**	>>> **num5**
Divide by $\mathbf{-2.25577 \times 10^{-5}}$: **num5/ - 0.0000225577**	>>> **Altitude1**

Ireland esero
BCO The Space For Science
Science Foundation Ireland sfi For what's next
ceia.ie cork's technology network

Arduino code:

```
int pressureValue;
float pressure;
float altitude;

float num1;
float num2;
float num3;
float num4;
float num5;


void setup ( ) {
Serial.begin (9600);
}


void loop ( ) {
  pressureValue = analogRead (A1);
  pressure = ((pressureValue / 1024.0)  + 0.095) / 0.0009;

  num1 = pressure*100.0;
  num2 = num1/101325.0;
  num3 = log10 (num2);
  num4 = (num3/5.25588);
  num5 = pow(10.0, num4) - 1.0;
  altitude = num5/-0.0000225577;

  Serial.print(pressureValue);
  Serial.print(" | Pressure =  ");
  Serial.print(pressure);
  Serial.print ("  millibars  ");
     |
  Serial.print(" | Altitude =  ");
  Serial.print(altitude);
  Serial.println("  meters  ");
  delay (1000);
}
```

The Arduino **pow** function is used to find the anti-log of **num4**.

It's also possible (and good programming practice?) to do the altitude calculations using fewer (only 1 if possible?) lines, but to keep things easy to de-bug it's often better to do it in steps.