

instructables (/)

[Let's Make ...](#)[Featured \(/featured/\)](#)[Write an Instructable \(/about/create.asp\)](#)[Log In \(/account/login/\)](#) | [Sign Up \(/account/gopro\)](#)[Classes \(/classes/\)](#)[Contests \(/contest/\)](#)[Community \(/topics/\)](#)[Teachers \(/teachers/\)](#)[Pier 9 \(/pier9/\)](#)[AUTODESK. Make anything. \(http://www.autodesk.com\)](#)

advertisement

## MAKE A HIGH-ALTITUDE BALLOON TRACKER (ARDUINO)

Featured

118,546

885

103

| Posted Jun. 18, 2014 in [Technology \(/technology/\)](#) > [Arduino \(/technology/arduino/\)](#)[\(/member/Ugifer/\)](#)By **Ugifer**[\(/member/Ugifer/\)](#)[Follow](#)More by  
the author:

About: Call me Ugi. Special offer! Make something based on one of my instructables and post a picture for a free 3-month pro membership (while stocks last). [More About Ugifer » \(/member/Ugifer/\)](#)

One of the coolest projects I have done so-far with my kids is a "near-space" balloon. That's not quite into space itself (100Km+) but so high that the sky looks black and you can start to see the curvature of the earth as the globe rolls away below you.

The Flickr set with the photos from our first launch is here:

<http://flic.kr/s/aHsjK22nDc> (<http://flic.kr/s/aHsjK22nDc>).

This is a quick-fire slideshow of the photos from the balloon, formatted as a 2.5

frames/sec video. I'm having trouble embedding it so the link is [here](http://flic.kr/p/gjzH3c) (<http://flic.kr/p/gjzH3c>).

ToGo1 Slide Show



(/member/Ugifer/)

Follow

By Ugifer (/member/Ugifer/)

Download

Favorite

In order to *take* this kind of photo you need to send a camera up to the stratosphere. Ours went 38Km (124,000 feet) straight up. This is easy enough: you attach it to a massive balloon, let it rise until the balloon bursts (due to the v. low pressure at the edge of the atmosphere) and then it will fall back to earth.

In order to *see* the photos that you have taken, you then need to find the camera afterwards. This is the trick.





Fortunately, at least in the UK and increasingly across Europe, the very helpful guys at the UK High Altitude Society ([ukhas.org.uk](http://ukhas.org.uk)) have developed a distributed network of trackers who will receive a signal from your balloon, upload the data to a server and plot the position for you on a Google Maps based page ([spaceneer.us/tracker/](http://spaceneer.us/tracker/)).

In order to take advantage of this wonderful network of helpers, we need to build a tracker that will communicate with their equipment. That is what I will outline in this instructable.

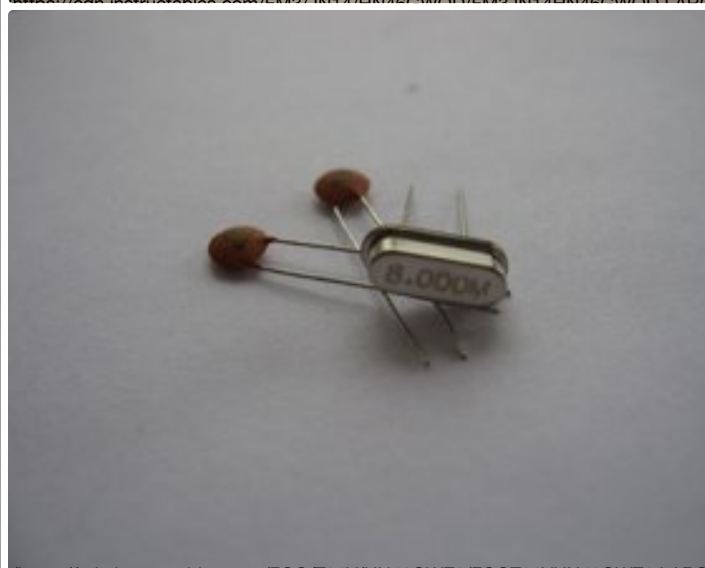
There are plenty of rules and regulations regarding what you can fly and which parts of the radio spectrum you can use for various tasks. The approach I will use in this instructable is suitable for the UK under the rules prevailing in 2013. If you live elsewhere or you are doing this significantly after I write this instructable, please check the rules that apply to you. The guys at UKHAS are fantastically helpful.

So - let's build a radio-tracker.

advertisement

 Add Tip
  Ask Question
  Comment
  Download

## Step 1: Things We Need



9 More Images

The tracker is based upon the Atmel ATMeag328 Microcontroller which forms the heart of many of the popular "Arduino" boards. We are going to make an "Arduino Compatible" board which we can program using the Arduino IDE.

Because the GPS module and SD card both require 3.3v and we have plenty of computing power, we may as well make the whole tracker run on 3v3. That means that we can't clock the '328 up to its full 16MHz but it will run happily at 8MHz on 3v3, and that's plenty for our purposes.

The tracker is designed on a custom PCB which is 5 x 5 cm in size. This means that it can be fabricated on some incredibly cheap PCB fab' sites. The design files for the tracker board and more details on the design are in the later steps.

These are the materials I used for my design. There may well be better options but this worked well for me:

**Materials:**

- 1 x Custom Tracker Board (see next step)
- 1 x ATmega 328\*\*
- 1 x 8MHz Crystal (NB not the usual 16MHz)
- 2 x 22 pf ceramic caps
- 1 x Radiometrix NTX2 (e.g. from [UPU's store \(http://ava.upuaut.net/store/\)](http://ava.upuaut.net/store/))
- 1 x GPS breakout board with antenna (the board is designed for [this one \(http://ava.upuaut.net/store/index.php?route=product/product&path=59\\_60&product\\_id=52\)](http://ava.upuaut.net/store/index.php?route=product/product&path=59_60&product_id=52))
- 1 x SD card breakout board (e-bay)\*
- 1 x board-mount DS1820 temperature sensor\*
- 1 x external DS1820 Temperature sensor (e-bay)\*
- 1 x HoneyWell HSCDANN001BA2A3 board-mount absolute pressure sensor (Mouser)\*
- 1 x tactile switch
- 3 x 100nf ceramic caps
- 1 x 100 uf electrolytic cap
- 1 x MCP1825 ultra-low dropout 3v3 regulator
- 1K resistors (around 5)
- 10K resistors (2-3)
- 4k7 resistors (4-5)
- 3mm LEDs (0-4 according to preference, different colours)\*
- Selection of 1% resistors (I used: 47K/6.8K for the battery sensor and 6.8K/10K for the radio driver )
- Male header pins (0.1") - some straight, some 90'

**Tools:**

- Soldering station and solder
- Clippers

Helping hands

Wire strippers

### **Programming / Debugging Tools:**

PC with Arduino IDE installed

USB to UART adaptor (I use a cheap CP2102. FTDI would work too) or program with ISP (see below)

### **For Bootloading/Programing\*\*:**

ISP programmer


or

Arduino + breadboard or ISP shield

\* - These sensors and indicators are not essential to the function of the board but allow you to send some interesting data and log it for analysis offline.

\*\*You can buy pre-bootloaded chips but they expect a 16MHz crystal. You would likely at least have to correct the timings in the code to make it work on 8MHz. It is certainly preferable to burn an 8MHz bootloader (see later). It's not hard to do.

 Add Tip

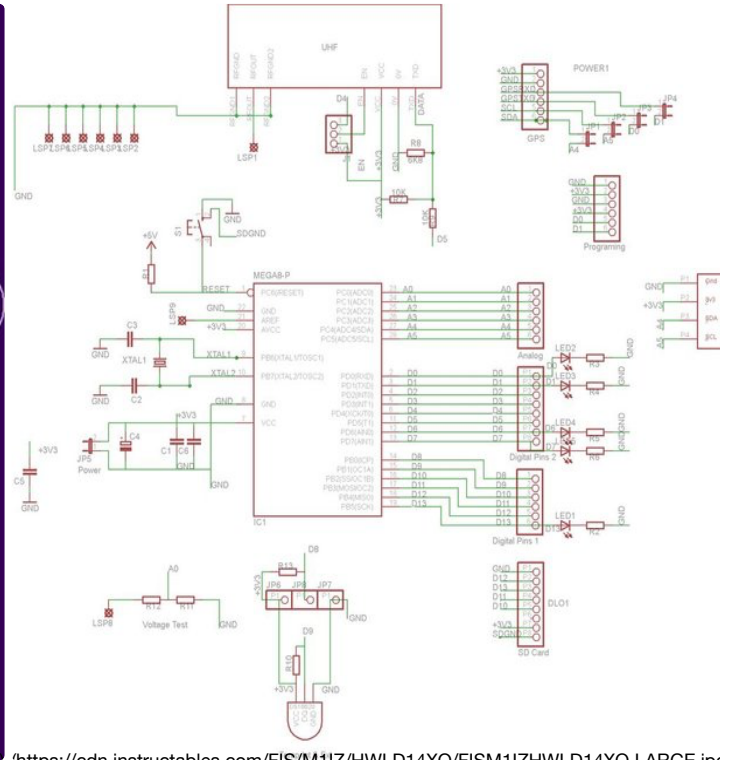
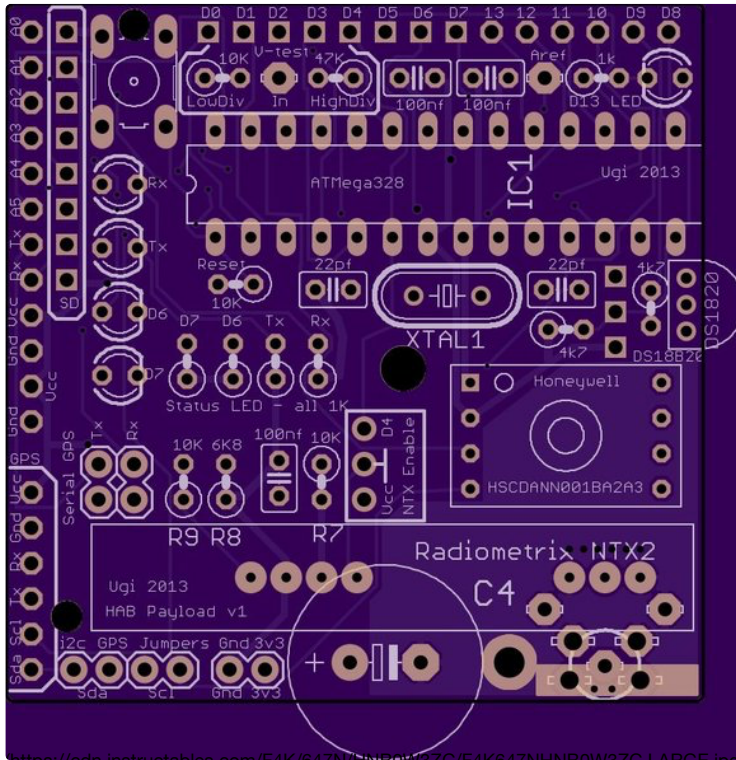
 Ask Question

 Comment

Download

---

## **Step 2: Tracker Design**



Attached to this step are the Eagle schematic and board files that I used for my tracker board, along with two .zip files: one with the board and schematic files and one containing all of the gerber files for the various layers. My board worked fine, but was not 100% perfect, for example, the LEDs on the serial port lines should really pull up to +3v3 rather than pulling down to ground so that they are off when there is not activity.

You may well prefer to design your own board, or at least modify mine. If you do this then there are a number of features that are necessary and/or useful for a High-Altitude tracker. These are some that are worth considering:

### Power:

You can't send an alkaline battery up into the atmosphere - it get's down to -50°C and they just stop working.

Even though you will insulate your payload, you want to use disposable lithium batteries - the are light and work well at low temperatures.

The GPS and SD-card run at 3.3v only. 5v will fry then. If you can run everything on the same voltage you save wasting too much power in voltage conversions.

If you use a switching power supply or an ultra-low dropout regulator then you can wring more flight-time from your power-pack.

### Robustness:

You could make your circuit up with lots of loose wires, with sockets or on stripboard, but it's going to get very cold and shaken about!

If you can it's worth designing a custom PCB and soldering everything directly to it.



No sockets to lose contact, at least for the critical parts. My PCB is small and cheap but will allow soldering of the processor, crystal, GPS and radio. I elected for direct soldering of the radio antenna too.

**Parts:**

Radio - in the UK there are only a few narrow frequencies you can use from the air and even then you are only allowed 10mW. The NTX2 is one of the few viable options for radio modules.

GPS - layout of a GPS module can be difficult and ideally you would keep it away from everything else. I opted for a pre-made module that could be soldered above and off to the side from the main board. It's important to note also that many GPSs don't operate above 14km height. Ublox GPSs have a flight mode that does work. For any others you will need to do your research before you buy/use.

Microcontroller - the ATmega 328 is a great option because it can be programmed from the very easy Arduino IDE. It will also run on 3v3 but only a 8MHz. However that's plenty of processing power for reading a GPS and running a low baud-rate radio.

Temperature - a temperature sensor provides interesting data but many are not rated down to the -50°C that we could encounter at the tropopause. The DS18B20 is a good option because they are pretty cheap and are rated to low temperatures. You can get "external" ones which are sealed in a stainless tube for added protection. We'll use one board-mounted one and one external.

Pressure - there are not too many "absolute" pressure sensors that I have found. The Honeywell HSCDANN001BA2A3 supposedly goes down to "0" mBar and in practice performed well down to 8mBar on the flight. Its i2c interface is pretty easy to handle. Honeywell make about a zillion variants but many are "relative" sensors - they measure the difference between a certain pressure and atmospheric. You need an "absolute" pressures sensor because it's atmospheric pressure we are measuring.

SD card - you could potentially surface-mount your own SD card holder but that's pretty fiddly and I didn't have enough board area anyway. SD-card breakouts are cheap and easily available so I elected to solder one underneath the main board to be compact and easy. A lighter approach would be to solder wires to the pads of a micro-SD adaptor and use it as a socket for a micro-SD card.

EDIT

Attached is now an additional .zip file. This contains version 1.1 of the board (Eagle schematic and board files, plus gerber files suitable for itead studio and major board-houses). The the following changes were made over v1:

LEDs on the Serial line pull up rather than down so should be on only when data transmitting.

Pull-up resistors now present on-board for i2c lines

Space for ultra-low dropout regulator on-board

Battery voltage now directly connected to V-test input. There is a track at bottom-middle you can cut if you don't want that.

I have not flown v1.1 of the board so there *could* be other errors or new problems with it. However, to the best of my knowledge this addresses the known issues of v1.

Ugi

**328 HAB Payload\_v1.brd**

Download (<https://cdn.instructables.com/ORIG/FYF/AD4F/HWLD14YM/FYFAD4FWLD14YM.brd>)  
(<https://cdn.instructables.com/ORIG/FYF/AD4F/HWLD14YM/FYFAD4FWLD14YM.brd>)

**328 HAB Payload\_v1.sch**

Download (<https://cdn.instructables.com/ORIG/FHN/LYKD/HWLD150F/FHNLYKDHWD150F.sch>)  
(<https://cdn.instructables.com/ORIG/FHN/LYKD/HWLD150F/FHNLYKDHWD150F.sch>)

**Hab1.zip**

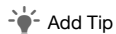
Download (<https://cdn.instructables.com/ORIG/F8G/MSA/HWLD1528/F8GMSAHWLD1528.zip>)  
(<https://cdn.instructables.com/ORIG/F8G/MSA/HWLD1528/F8GMSAHWLD1528.zip>)

**HAB\_Board.zip**

Download (<https://cdn.instructables.com/ORIG/F5B/3RZ/HWLD1532/F5B3RZHWD1532.zip>)  
(<https://cdn.instructables.com/ORIG/F5B/3RZ/HWLD1532/F5B3RZHWD1532.zip>)

**HAB\_Board\_1.1.zip**

Download (<https://cdn.instructables.com/ORIG/FAU/PUNJ/IJX7DV0S/FAUPUNJIJX7DV0S.zip>)  
(<https://cdn.instructables.com/ORIG/FAU/PUNJ/IJX7DV0S/FAUPUNJIJX7DV0S.zip>)



Add Tip



Ask Question

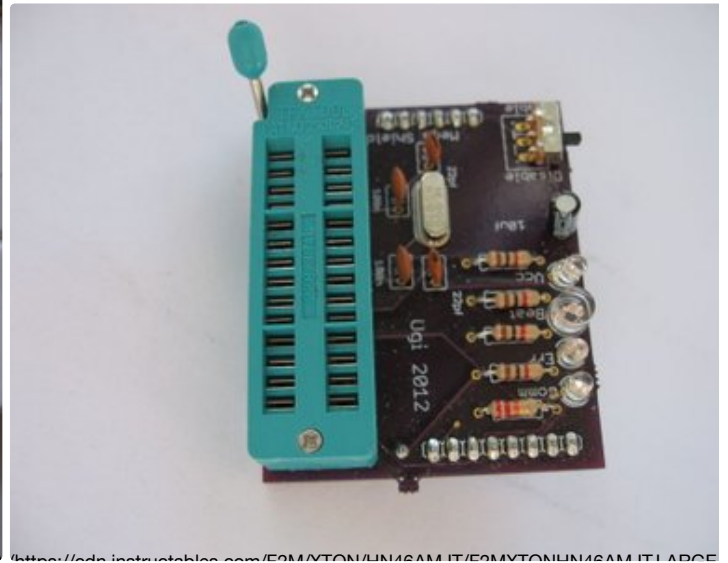
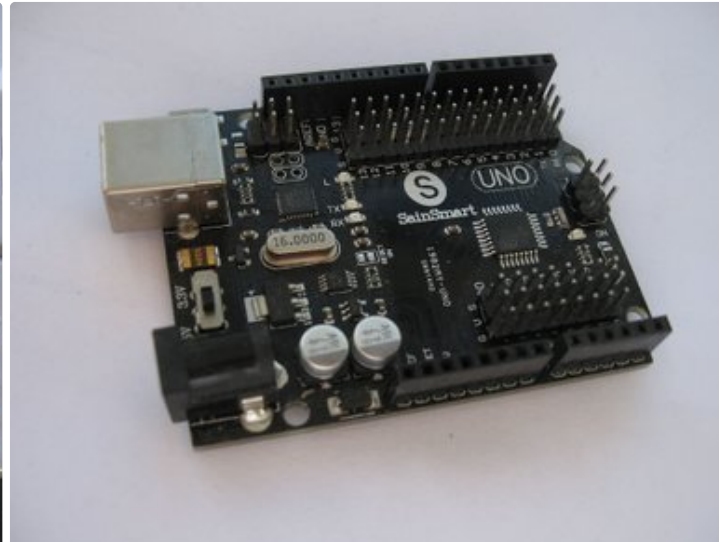
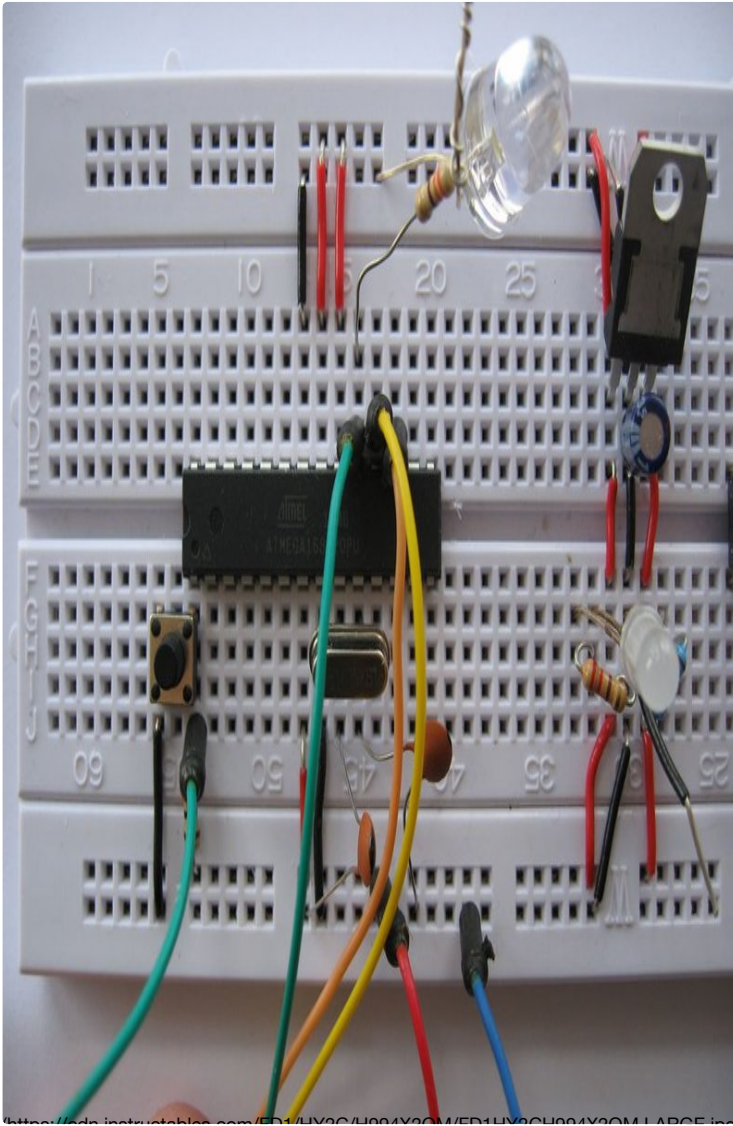


Comment

Download

## Step 3: Bootloading





In order to use the Arduino IDE reliably at 8MHz, you need to burn the "Arduino Pro Mini 8MHz" bootloader onto your ATmega328.

There are all sorts of ways of doing this, but I will assume here that you have a '328 based Arduino board (like an Uno, nano or Duemilanove). Using this you can bootstrap your '328 using either a dedicated "shield" or just a solderless breadboard.

### Breadboard method:

- 1) Place your '328 in the centre of your breadboard. Run +5v and Gnd from your arduino to the power rails on your breadboard.
- 2) Run a 10K from pin 1 on the BB (reset) to +5v.
- 3) Fit a crystal (8 or 16MHz) and cap's (22pf) on pins 9 and 10 on the BB.
- 4) Join pins 7 and 20 (on the BB) to +5v and pins 8 and 22 to Gnd with jumper wires.

- 5) Jump from D11, D12 and D13 of the Ardu' to pins 17, 18, 19 on the breadboard.
- 6) Jump from D10 on the Ardu' to pin 1 on the BB.
- 7) Plug the Ardu' into your PC and burn the the Arduino ISP sketch onto your Ardu' from the Examples.
- 8) Put a 10uf capacitor from reset to Gnd *on the Arduino (not the BB)*.
- 9) In the IDE, select "board" as Arduino Pro Mini 8MHz. Select the programmer as "Arduino ISP"
- 10) In the IDE click "burn bootloader".
- 11) The lights will flicker for a bit while the bootloader is transferred.
- 12) When it's finished, the D13 light on the Ardu should flash regularly. That's in parallel with D13 on the new chip and tells you the blink sketch, which is burned with the bootloader, is running on it.

### Shield Method

I got fed up with laying out BBs the whole time so I made an small ISP shield. The picture is of mine but if you don't want to make your own then you can buy one from EMSL (<http://shop.evilmadscientist.com/productsmenu/tinykitlist/253>).

The procedure is very similar but reduces the setting up to one step:

- 1) Burn the Arduino ISP sketch onto the Arduino & un-plug it.
- 2) Put your '328 in the shield and the shield on the Arduino.
- 3) Plug in the Arduino and disable auto-reset (jumper or switch on the shield)
- 4) In the IDE, select "board" as Arduino Pro Mini 8MHz. Select the programmer as "Arduino ISP"
- 5) In the IDE click "burn bootloader".
- 6) The lights will flicker for a bit while the bootloader is transferred.
- 7) When it's finished, the D13 light on the Ardu should flash regularly. That's in parallel with D13 on the new chip and tells you the blink sketch, which is burned with the bootloader, is running on it.

## Testing:

Your '328 should now be ready to be programmed.

If you want to test it before soldering directly onto the board then you can lay it out as a breadboard Arduino as described above. Now, disconnect the Arduino from the Breadboard and connect up the CP2102 USB to TTL converter. That will go to +5v and Gnd, with Tx on the CP2102 going to D0 and Rx to D1. Some CP2102s are marked backwards so you may need to switch these two. Finally you need to add a reset switch between pin 1 and Gnd. You might also want an LED and resistor (say 1k) between pin 19 and Gnd.

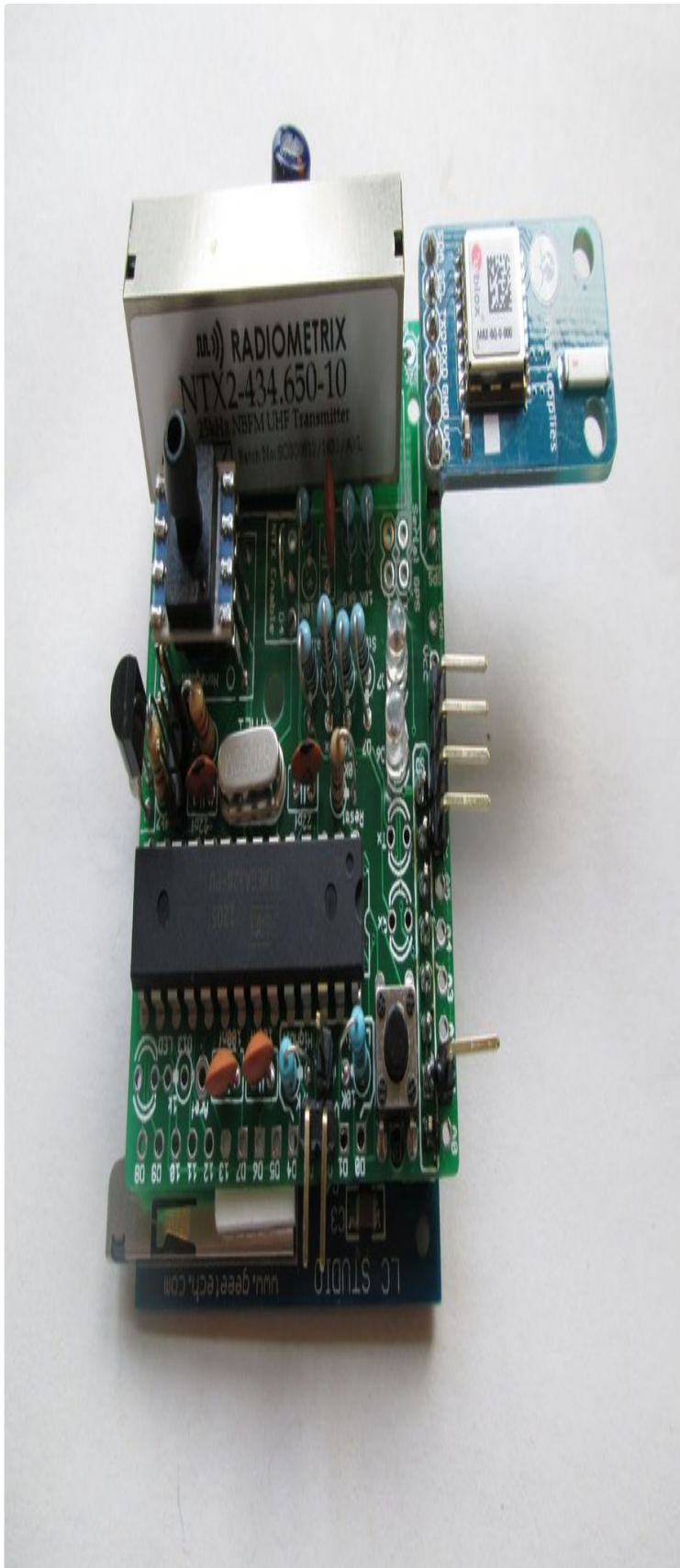
To load a sketch, select Arduino Pro Mini 8MHz as the board and upload a sketch. Just when it finishes compiling and tells you the final size at the bottom of the IDE, press and release the reset button to start the upload. You should be able to upload the blink sketch and see the D13 LED flashing with a new timing.

You can burn the final sketch now if you want. It will save you connecting the CP2102 up to the tracker board. See a later step for the sketch.

[Add Tip](#)[Ask Question](#)[Comment](#)[Download](#)

---

## Step 4: Put It Together..



9 More Images

There are several options on this board - you can control the GPS by serial or using i2c, you can tie the "control" line of the radio direct to ground or you can control from the microcontroller etc.

The board is made so that you can put pin headers on the various options and change them around with jumpers. This is very useful in development. However, for the actual flight build, I wanted to hard-solder all of the jumpers to avoid anything losing contact in the harsh conditions of flight. This build therefore solders all of the jumpers but if you are developing your tracker you may well want to use jumpers until you know which option you want to use.

In general, I started from the centre and worked outwards for ease of soldering. The SD-card breakout needs to go on late because it covers a lot of the under-side and so prevents you from soldering on very much else once it's there.

There's a lot crammed into this board to make it small and cheap and light. That means it's not terribly suitable for a first soldering project. However, it's all through-hole so apart from being a little tight in places it's not at all difficult.

### **Construction:**

The pressure-sensor mounted above the board on the ends of its legs. This just allows it to be clipped off again if we wish. It's the most expensive component and you might want to use it on another project later.

The 1Ks for the status LEDs, the pullups for the DS18B20s, header for the external DS18B20 plus the crystal and caps. Nearly all resistors are vertical on this board to save space. Remember to use an 8MHz crystal!

The NTX decoupling cap, control line and voltage divider. The NTX changes frequency according to a voltage supplied on the control pin. To transmit RTTY 50 baud, we want a shift of around 500 Hz. I used 10K and 6K8. 1% resistors are a good plan here.

Microcontroller - make sure you bootload it first! If you haven't done that yet, go back to the last step and do it now! Then solder it in. Don't worry about direct soldering it, AVRs are hard as nails, but do get the notch at the right end - it'll be a pig to un-solder.

Internal DS18B20 and reset switch. Dead easy.

Header - all pins of the '328 are broken out but most of them you don't need. If you want to re-program it in place then you will need a 4-pin header for the Tx, Rx, Vcc and Gnd lines. Try to work out what other pins you might need and whether you want to access them from the top or side.

The voltage divider for battery sensing connects to A0 and senses against the 1v1 internal standard. You need to set the range according to the maximum your battery pack can provide because this input cannot go above 1v1 without risking damage



(remember a 1.5v battery can be significantly higher when new). I used the arrangement: Batt-47K-A0-6.8K-Gnd which is safe to around 8.7v.

Now is a good time to add the status LEDs, smoothing cap and GPS jumpers. In the version of the board shown later the i2c pullups are properly on-board next to the pressure sensor. I forgot them on the first version so I added them on the underside, connected to the sda and scl jumper positions.

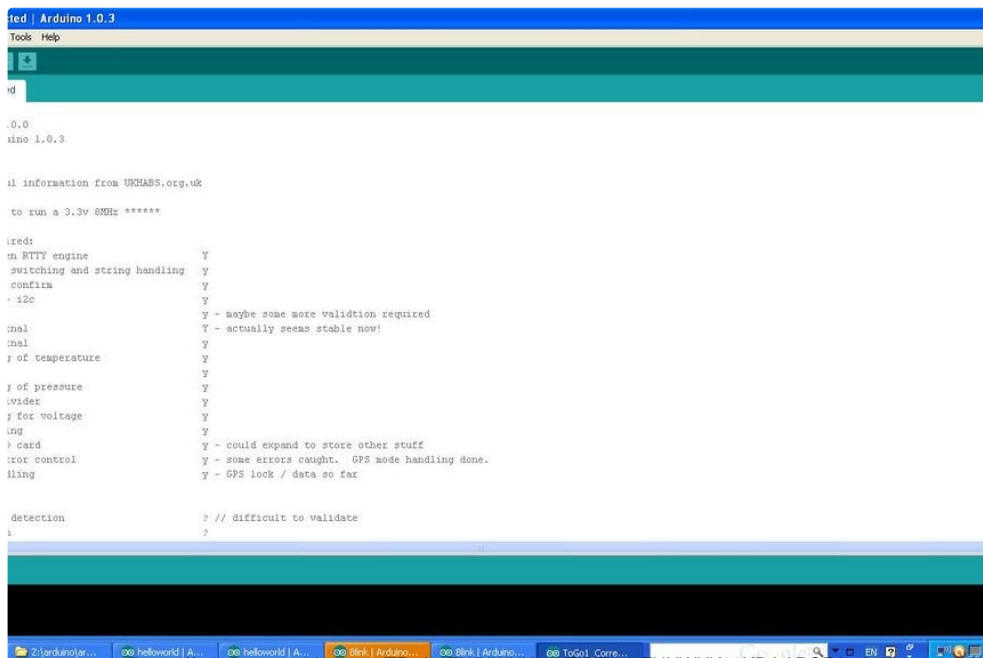
The NTX2 radio module slips in next, then have a good look over the board. I did not add all of the possible LEDs or headers. If you want any others, add them now because once the SD-card is in, it's going to get a whole lot harder to solder anything!

The SD card sits under the board with its header pins soldered in from the top. A foam sticky pad will keep the boards from shorting and hold it nicely while you solder it.

Finally, the GPS breakout board attaches to the edge on header pins. I used the full length of the pins to put the GPS physically above and to the side of the main board.

💡 Add Tip    ? Ask Question    💬 Comment    Download

## Step 5: Create a Sketch



### Sketch:

My full sketch is available in the .zip file in the next step, however, it's useful to have an idea of the overall plan to make it easier to adapt.

**Interrupt:**

We need to drive the radio by modulating pin D5 high/low at a rate of 50 baud. That is, every 20ms we need to represent a new bit of data as a high or low for a 1 or 0 respectively. We use this data stream to encode a text string. There are various methods for this, but 7-bit ASCII is most common. We'll keep a double-buffer containing the string for transmission and a copy that we are preparing with updated data.

In order to get the most efficient and reliable transmission and especially to give us accurate timing, we send the string under interrupt. This requires a short and fast piece of code simply to keep track of which bit of which byte of the string we are currently transmitting and move on to the next bit each time. When we finish one string we switch the string buffers, moving over to send the most up-to-date data from the other buffer.

**Main Loop:**

In the main loop, we circulate, reading each of the sensors and the GPS whenever they are ready. The latest values are stored in variables. When the GPS is ready to provide a new location, we read that, decode it and update the working copy of the string. We have to turn off interrupts briefly in order to update the buffer so that we don't end up transmitting a half-constructed string. As a result, we update the buffer only when a new GPS location is ready, so that we don't have interrupts off for too long.

Each time we switch buffers we set a flag telling the main loop that a new transmission is started. The old string that was just sent is then written to the SD card before opening the buffer up to being updated.

We also need to keep an eye on the GPS to make sure it stays in flight mode. The reason we use a Ublox GPS module is that it will operate up to the heights that we will be visiting - many GPSs don't operate above about 14km. In order to make the Ublox work at high altitude, we need to make sure it is in the right mode.

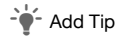
**Sensors & ports:**

The board can talk to the GPS module by either I2C or UART. I have elected to use the I2C interface so that I can use the UART for debugging by attaching to the PC serial port. We will also use the I2C to talk to the Honeywell pressure sensor, but the two seem to coexist on the I2C lines without conflict.

The board will take two DS18B20 temperature sensors. I used one internal and one external. These are very sensitive to timing and I needed to turn interrupts off for a few microseconds while reading them. They are otherwise very easy to use. I used a copy of the wire library that claimed to have fewer hangups than the native Arduino one. True or not, it worked fine.



The SD card sits on the SPI pins and is handled by the SDfat library. SD card writing takes a load of memory and one of the real tricks to this sketch was trying to monitor how much free SRAM was available.



Add Tip



Ask Question



Comment

Download

## Step 6: Burn the Sketch



I have assumed that you will want to develop a sketch and so probably want to re-program the chip on the tracker board. If you already programmed it in step 3 and don't want to do any further development then you can skip this step.

*The key thing to remember with this board is that it does not have an auto-reset so you will need to press "reset" at the right time to upload a sketch. This can take a few tries to master.*

### Hardware:

For this step, you will need a PC loaded with the Arduino software, your newly made tracker board and a CP2102 UART to USB converter.

Connect 3v3 (not 5v) from the CP2102 to Vcc on the tracker board. Connect the Gnds together.

Connect Rx on CP2102 to Tx on the tracker and Tx on the CP2102 to Rx on the tracker.\*

### Sketch:

My sketch is attached to this step. Because of the various libraries etc needed, I have attached it in the enclosed .zip file. This should contain everything that you need in addition to the normal Arduino environment.

The sketch was written in Arduino IDE v1.0.3. I have not tested it on any other version. It requires several Arduino libraries to help handle the GPS and SD card aspects. They are also in the .zip file. Place the various libraries in the libraries folder and then open the Arduino IDE. Now load the sketch.

There are two places in the sketch where I have written "Test1" as the callsign. You will want to replace that with your own callsign before you upload the sketch. Make it only a few characters because the sketch might over-run its text array otherwise.

### Upload:

To talk to the tracker board you will need to load the drivers for the CP2102 and select the new COM port in the Arduino environment. Now select the board as Arduino Pro Mini 8MHz. The 8MHz is important because you used an 8MHz crystal and so if you select a 16MHz board then your timings will all be off.

To load up the sketch, click "upload" and wait for it to compile. When the IDE displays the final compiled size, press and release the reset button on the tracker board. Lights should flicker for a while then the lights on the tracker board flash briefly as the sketch starts up.

#### UgiHABcode.zip

Download (<https://cdn.instructables.com/ORIG/FLM/U8NE/HWKJZVCV/FLMU8NEHWKJZVCV.zip>)

(<https://cdn.instructables.com/ORIG/FLM/U8NE/HWKJZVCV/FLMU8NEHWKJZVCV.zip>)



Add Tip



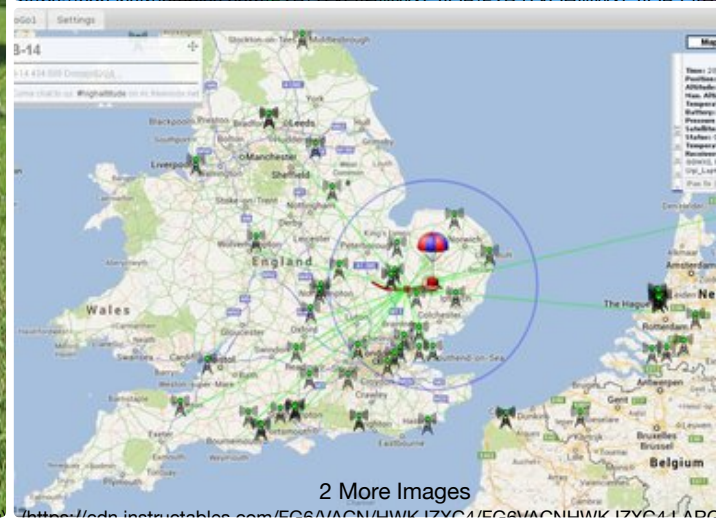
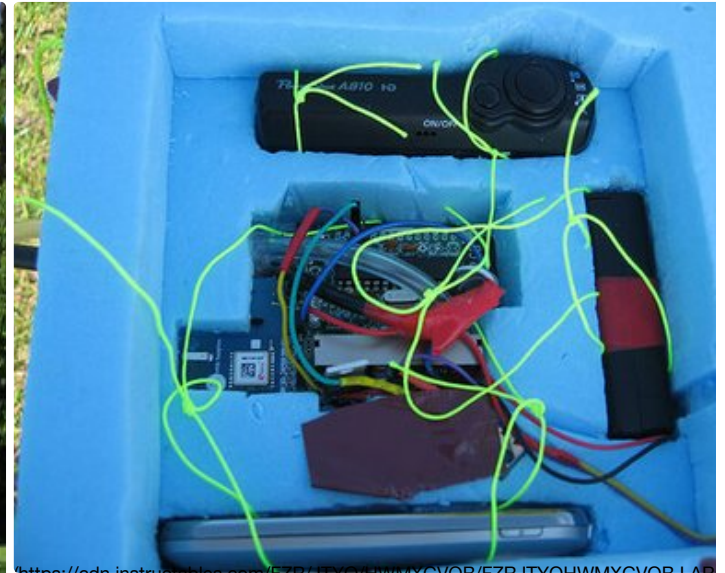
Ask Question



Comment

Download

## Step 7: Using the Tracker



Now that you have built a High Altitude Balloon tracker, you have completed the hardest part of your own High Altitude Balloon launch. The next steps are to incorporate this into a payload, add a camera (or two), a parachute and a balloon and launch them into the stratosphere. These steps are discussed by the very helpful people at UKHAS ([ukhas.org.uk](http://ukhas.org.uk)).

Before you can get information back from your tracker, you will also need to set up a "payload" document on the High-Altitude Balloon tracking server. That can be done [here](http://habitat.habhub.org/genpayload/) (<http://habitat.habhub.org/genpayload/>):

Once you have a payload document set up, you can either fly your tracker on blind faith and hope someone will track it, or you can test it yourself. I recommend the latter.

To test your payload, you will need to set up for 70cm reception. The cheapest way is to use a "Software Defined Radio" dongle attached to a 16cm bit of wire as an antenna. These parts are available from e-bay for around £10 simply by searching for "SDR dongle." The procedure for setting up an SDR tracker is described in great detail on the [ukhas.org.uk](http://ukhas.org.uk) site [here](http://ukhas.org.uk/guides/sdr_tracker) ([http://ukhas.org.uk/guides/sdr\\_tracker](http://ukhas.org.uk/guides/sdr_tracker)):



Once you have that set up and you are seeing your tracker on [spaceneer.us/tracker](http://spaceneer.us/tracker/) (<http://spaceneer.us/tracker/>), then your payload is ready to launch and be tracked.

The process of incorporating your tracker into a payload, getting CAA clearance, announcing your launch and actually launching are all described at [ukhas.org.uk](http://ukhas.org.uk) (<http://ukhas.org.uk>) and it is worth popping by at <http://webchat.freenode.net/?channels=highaltitude> (<http://webchat.freenode.net/?channels=highaltitude>) for plenty of help and advice. However, with your tracker built, the rest of the process is comparatively straightforward.

Good luck with your High-Altitude Balloon! I hope I get to track it pretty soon!

Ugi



Add Tip



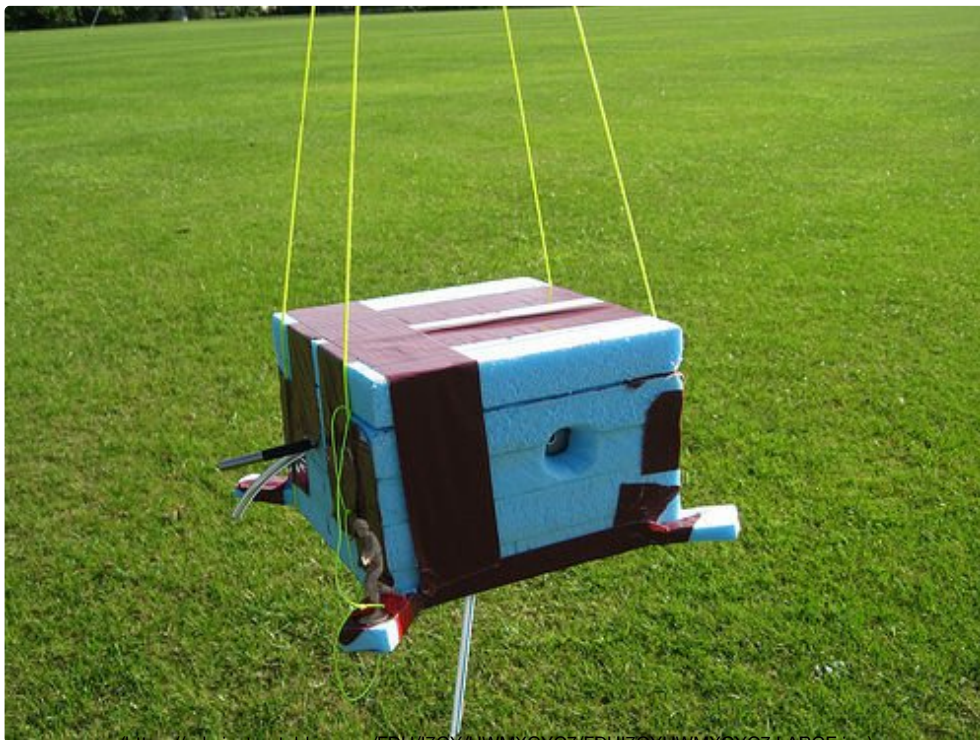
Ask Question



Comment

Download

## Step 8: Send Your Instructables Contest Prize to the Edge of Space!



Is this the first Instructables Contest prize to reach the edge of space?

The purple duct tape that you see in this photo was a prize from the Jury-rig-it Contest last year (<https://www.instructables.com/contest/juryrigit/> (<https://www.instructables.com/contest/juryrigit/>)). It must have gone closer to space than any other Indestructibles Prize so far..

Have any previous Instructables Contest prizes reached 124,000 feet? I'd love to know if they have!

If you think more Instructables Contest prizes should leave the atmosphere then please feel free to vote for this entry!

Thanks

Ugi



Add Tip



Ask Question



Comment

Download

advertisement

advertisement



Second Prize in the  
[Battery Powered Contest](/contest/batterypowered2014/)  
(/contest/batterypowered2014/).

[View Contest](#)



Grand Prize in the  
[Launch It! Contest](/contest/launchit2014/) (/contest/launchit2014/).

[View Contest](#)

### 3 People Made This Project!

maureen-selinal  
(/member/maureen-selinal/) made it!


Lloydy21  
(/member/Lloydy21/) made it!


criticalmass115  
(/member/criticalmass made it!


Did you make this project? Share it with us!


[I Made It!](#)

### Recommendations



  
Add Tip

  
Ask Question

  
Post Comment

We have a **be nice** policy.  
Please be positive and constructive.

Add Images

Post

Asked by dbiswas6 (/member/dbiswas6/) 3 months ago

0

Hey so i had this doubt cuz im sorta of a newbie to this hobby. I need a data logger as well as a tracker and both require the same gps breakout board. So is there anyway i could build both of them with one board??

**103 Comments**

<https://www.instructables.com/id/Make-a-High-Altitude-Balloon-Tracker-Arduino/> Page 21 of 23

Do I need to get my radio license before I can use this in the USA? Also what dongle do you advise to track the payload? (preferably one that connects to a PC)

NadeemH2 (/member/NadeemH2/) 11 months ago

Reply

Hi Ugi

Can I use XTend 900 radio (<https://www.sparkfun.com/products/9411>) to keep providing me GPS data even the balloon goes 34 - 37 km up & then it burst. I mean can i use this radio till 37 km height. What if I am not precisely pointing my antenna towards it. Will it still work ?

Please help

1 reply ▼

NadeemH2 (/member/NadeemH2/) 11 months ago

Reply

Hello Ugi

Can you please let me know from where you bought this balloon? Thank you

1 reply ▼

DrBoost (/member/DrBoost/) 1 year ago

Reply

Is this UK specific, or can this be used in the US as well?

pde25clicks (/member/pde25clicks/) 2 years ago

Reply

Hi Ugi, I'm also part of the team sending a balloon 25km to celebrate the uni degree anniversary. We're currently trying to get the board and radio receiver talking and running into a few problems. The radio hears the board but when we try to send it to dl-fl digi to get read and parsed out issues arise. Do you see anything that immediately strikes you as out of place?

Thank you,  
Jaren

(<https://cdn.instructables.com/FDU/UY1E/IM28UI60/FDUNY1EIM28UI60.LARGE.jpg>)

criticalmass115 (/member/criticalmass115/) 2 years ago

Reply

Starting the build :- ) #veryexcited  
Steve

(<https://cdn.instructables.com/FDU/UY1E/IM28UI60/FDUNY1EIM28UI60.LARGE.jpg>)

3 replies ▼

criticalmass115 (/member/criticalmass115/) 2 years ago

Reply

Hi Ugi,



I appreciate that this instructable is a couple of years old now, but I wondered if I could ask you a question about your board? You mention that some of the resistors need to be pull up rather than pull down. Can you confirm that only applies to LED 2 and LED 3 please? I'm about to get the board manufactured and want to make these changes before I do. This is an amazing instructable by the way. I've wanted to send up a HAB for some time, but your tracker design as well as the explanation is what has given me the motivation to get started. Thank you for that.

I'm criticalmass on the IRC channel if you ever want to say hello.

Cheers,  
Steve

3 replies ▼

---

moein (/member/moein/) 2 years ago  
(/member/moein/)

Reply

what if it makes collision with airliners ?

2 replies ▼

---

More Comments

Post Comment

None