cFS Design

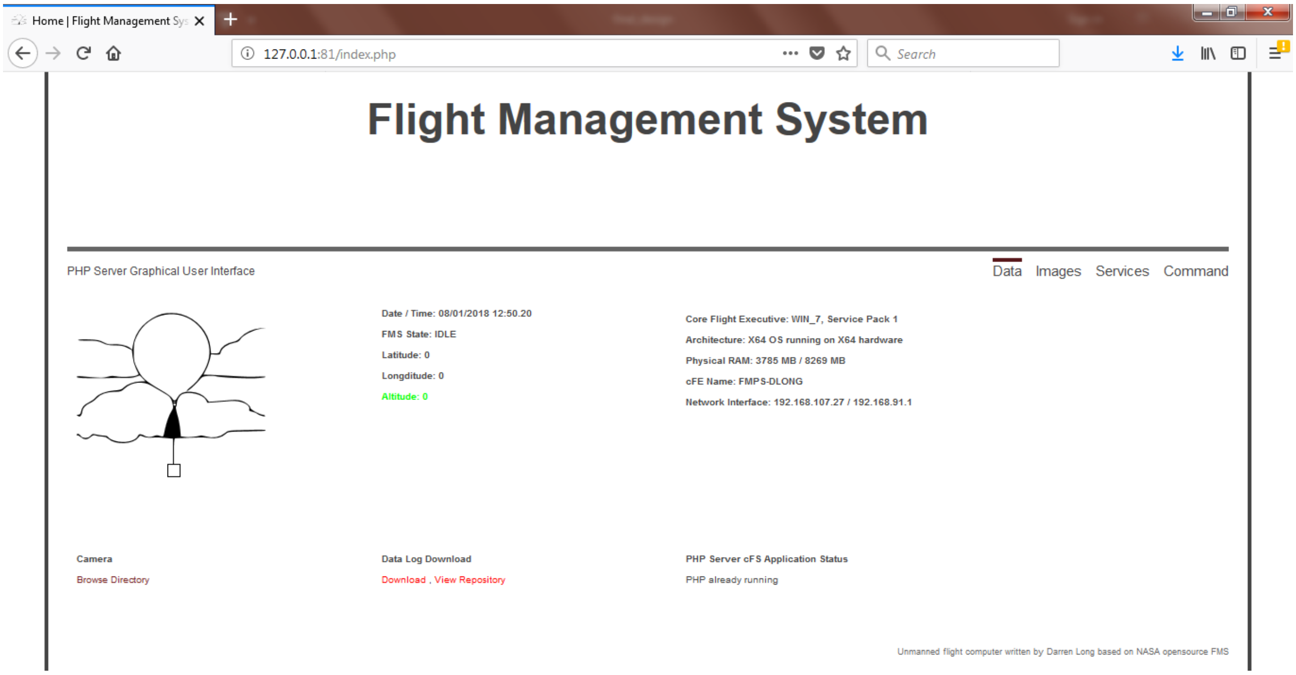
The core flight system (cFS) is the software and hardware that is running inside of the Payload. Its primary purpose is to collect data and photographs and relay them back to earth and provide a method of finding the Payload once it has landed. The software element has been designed based on information and source code from both Nasa cFS and ICAROUS. The system comprises of the following:

core Flight System (cFS) – *A Flight Software Architecture consisting of an OS Abstraction Layer (OSAL), Platform Support Package (PSP), cFE Core, cFSLibraries, and cFSApplications.* In this case fms.exe is the core application, this application like any conventional Flight Management System runs on a cycle collecting data from onboard sensors, determines the state (IDLE|TAKEOFF|CLIMB|CRUISE|DESCEND|LAND|TERMINATE) and performs additional tasks such as taking photographs at a set interval from TAKEOFF to the end of DESCENT and launching radio communication when TCP/IP connectivity to ground control is lost. All other cFS applications run as a result of fms.exe commanding them too with the exception that ground control can launch some applications remotely via the PHP server which is initiated on start-up of fms.exe.

core Flight Executive (cFE) – *A framework of mission independent, re-usable, coreflight software services and operating environment.* In this case a modified Windows Embedded Standard 7 Operating System and the drivers required to allow communication between the motherboard and the connected devices (camera, Arduino, broadband dongle).

Windows Embedded Standard 7 - A feature package with the lowest overheads needs To be used, hence an embedded windows OS being the choice. The shell will need to Be changed from explorer.exe to a command line shell to reduce CPU / RAM usage (in turn maximising battery life) and set current user to auto logon.

PHP Server



Currently Quick PHP Server is initiated on start-up of fms.exe and provides a method of transferring data and files from the Payload to ground control. However, a full php server package may be used instead going forward.

This is effectively a conventional webserver, however the landing page shows live data and system status (automatically updated by fms.exe) and allows access to file directories for the purpose of downloading data logs and images. A direct link to the highest altitude shot can be found on this page.

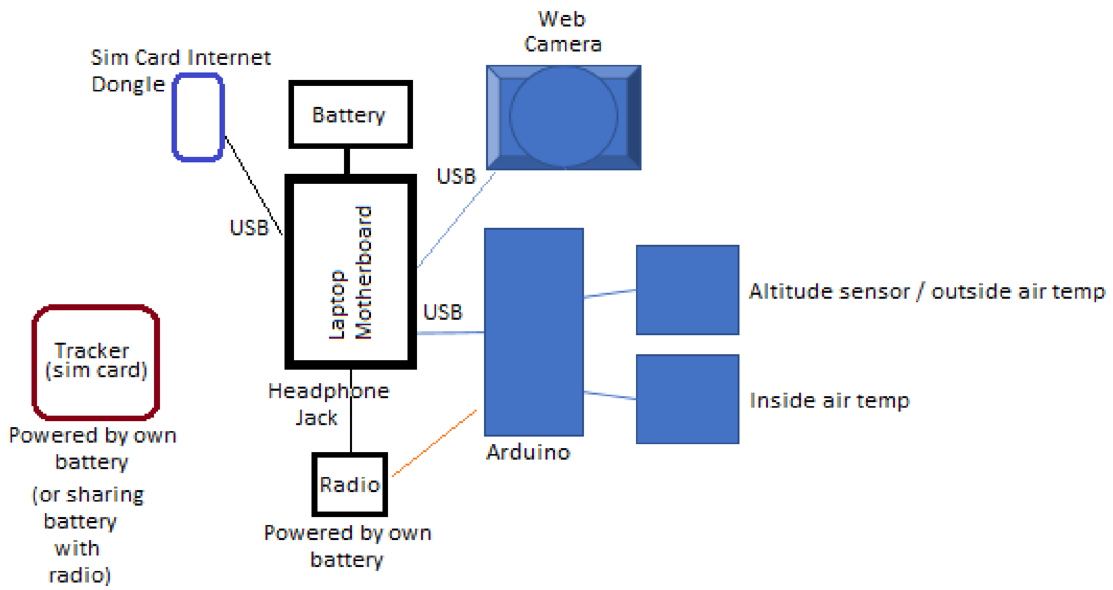
A data log showing DATE, TIME, FMS_STATE, OUTTEMP, INTEMP, ALT, BATT, EXEC_TIME can be downloaded and is auto archived in case of a reboot of fms.exe or indeed the cFE.

A directory of photographs can be viewed each showing data / time and altitude, these can then be downloaded individually or as a complete ZIP file by clicking on the Images tab which in turn commands the cFS to create the zip file.

The services tab allows ground control to check the status of key cFS applications and force the state of the FMS which could be used in case of malfunction to ensure correct communication with ground control and that photos are taken automatically.

The command tab allows ground control to remotely restart the FMS or cFE, change the configuration, run applications from the root directory remotely and upload and delete files in the PHP directory. Additionally ground control can remotely change the volume of the soundcard as to change the volume of the radio transmissions.

Communications from the Payload



Shown above is the design with a broadband dongle fitted with a SIM card to provide TCP/IP over 3g/4g between the FMS PHP Server and ground control, however the preffered method would be to utilise Spacex's Star Link satellites for internet conenction, as 3g / 4g will lose connection at around 10,000 feet (tested cell phone) – 26,000 feet (tested Nokia Lumia) (based on research from Bill Rojas at IDC Asia Pacific and the University of Southampton project ASTRA).

The FMS will identify if the connection has been lost via the status of a ping packet back to ground control (or a predefined web address) on each cycle. Once its identified that connection is lost it will continuously retry to connect and simultaneously start communicating altitude over the radio via a series of beeps. Its an option that the Arduino be linked to the power switch of the radio so that the FMS can have the Arduino switch the radio on only when required to avoid battery use and interference from the radio. A user at ground control taps in the beep codes and the software converts / logs them to a file along with a timestamp.

Once the balloon reaches maximum altitude and pops it will start its descent back to Earth aided by parachute. If using 3g / 4g then the connection will be regained at around 10,000 feet at which point the FMS will send an email to ground control announcing its Public IP Address and simultaneously disabling the encoder (perhaps even turning off the radio using the Arduino). At this point ground control can download the individual photo with the highest altitude in the file name (in case the connection is lost) before downloading all photos either in a Zip file which is created by the FMS or via Ground Controls mirror function.

A tracker fitted with another SIM card separate from all other circuitry (data and power) will be used to identify the Payloads location upon landing. Ground control simply send a text message to the tracker and await its response.

Its important to note that the radio transmitter (and broadband dongle) need to be facing down toward earth and some shielding may need to be placed between the two to stop interference. Additionally we must ensure that the battery can last the full duration of the flight or at least until all the data from the ascent has been downloaded.

The ethernet port on the motherboard will be set with a static IP so that it can be crossed over into whilst on the ground and a VNC server will be running on the OS. A power jack will also need to be available to ensure the batteries are at maximum charge before launch.

Communications from the Payload

Starlink (satellite constellation)

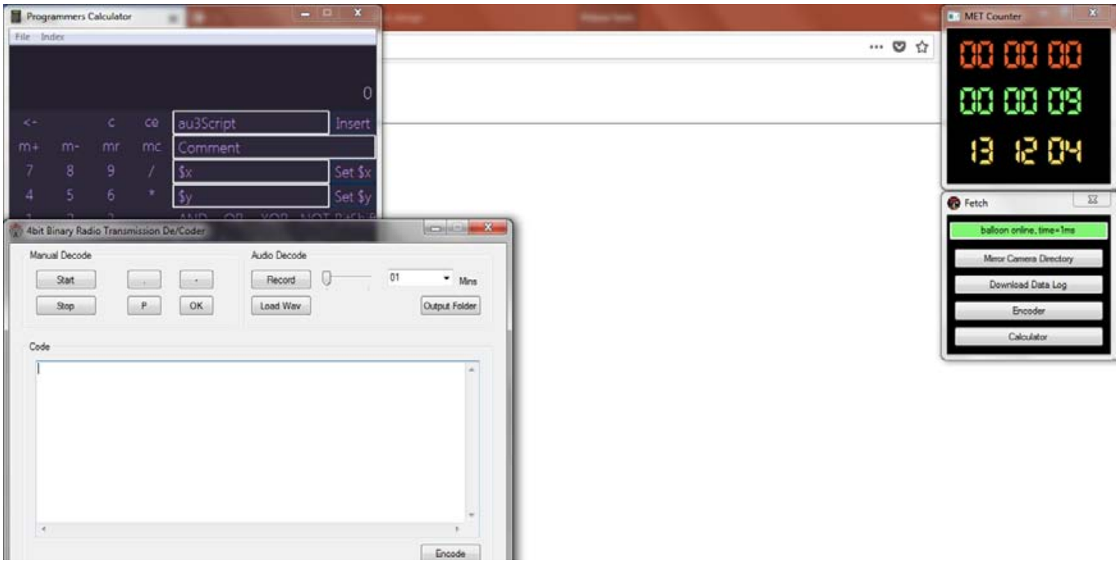
Starlink is a satellite constellation development project underway by American company SpaceX, to develop a low-cost, high-performance satellite bus and requisite customer ground transceivers to implement a new space-based Internet communication system. SpaceX also plans to sell satellites that use a satellite bus that may be used for military, scientific or exploratory purposes.
Starlink constellation, phase 1, first orbital shell: approximately 1,600 satellites at 550 km altitude

SpaceX has plans to deploy nearly 12,000 satellites in three orbital shells by the mid-2020s: initially placing approximately 1600 in a 550-kilometer (340 mi)-altitude shell, subsequently placing ~2800 Ku- and Ka-band spectrum sats at 1,150 km (710 mi) and ~7500 V-band sats at 340 km (210 mi). The total cost of the decade-long project to design, build and deploy such a network is estimated at nearly US$10 billion.

Product development began in 2015, and two prototype test-flight satellites were launched in February 2018. A second set of test satellites and the first large deployment of a piece of the constellation occurred on 24 May 2019 (UTC) when the first 60 operational satellites were launched. Initial commercial operation of the constellation could begin in 2020.

The SpaceX satellite development facility in Redmond, Washington, houses the research, development, manufacturing and on-orbit control operations for the satellite Internet project.

Ground Control



The Ground Control software performs a number of different functions and includes a number of different directories...

MET Counter – Enter the time to count from (or just press enter to set to 0), then as the balloon is launched press enter again to start counting. This counter is written to RAM so that other applications can retrieve the current mission elapsed time.

Fetch – This application shows the connectivity status of the balloon (identified by a ping packet to the IP of the balloon every 5 seconds). This enables ground control users to quickly identify when the balloon is back online. Mirror Camera Directory copies all new photos from the balloon (minimising data transfer). Download Data Log saves the current data log from the balloon as a csv, the file is named using the MET counter.

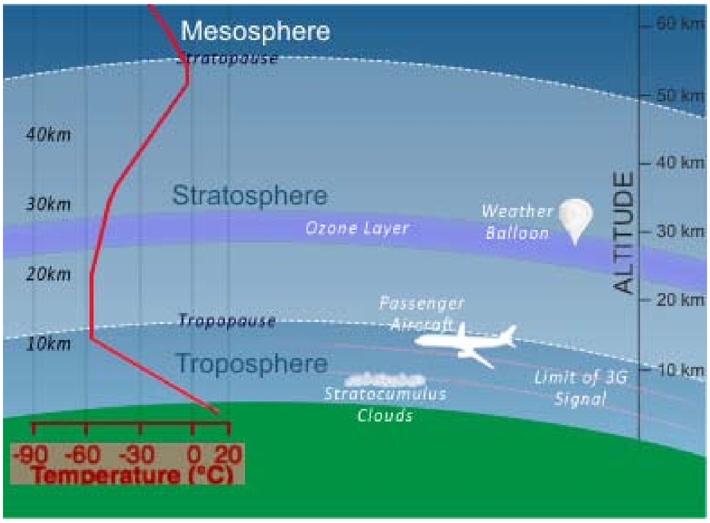
Encoder – This is used to decipher the beep codes sent down via radio. When connection to the balloon is lost the user sets a recording time interval and presses record. The beeps will be recorded to a wav file named with the MET time of the start of the recording, at the set interval a new wav will be created, thus breaking the recordings into separate files. The user slides the slider to the right to stop recording. They can then load a file to convert. This will show the deciphered output of the balloons beep codes. Additionally a csv and png file are saved into the radioWavs directory. The csv file will show the time (+ the met time of the file) that each character was sent. The png shows a visual representation of the waveform.

Firebird – This is a mix of two customized versions of Mozilla applications which are launched on startup of the Ground Control software. Firefox is used to connect to the balloons php server. Thunderbird is used to receive emails from the balloons fms.

Calculator – Used for on the fly calculations. Includes a function for retrieving the current MET time for use in calculations.

Directories – A folder named Storage is utilized by the applications to automatically create files in the associated subfolders.
rawPics – When using the mirror function all photos are saved in this directory
goodPics – Directory for user to sort good photos into
badPics – Directory for user to move unwanted photos into
dataLogs – Directory used for saving data logs when Download Data Log is clicked
radioWavs – Directory used by the encoder for saving wav, csv and png files

Reliability in the Hostile Environment



The above illustration shows the altitude that the Balloon is likely to reach and includes a graph showing the temperature changes that it will be subjected too including having to go through clouds and the Earths Ozone layer. Initially the temperature will drop very low before them warming up after going through the Ozone layer.

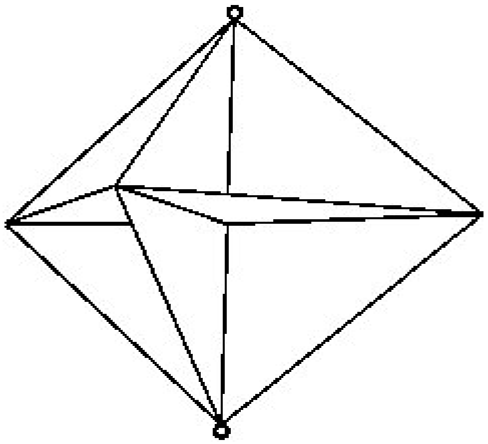
Whilst the Arduino can cope with temperatures ranging from -40c to +85c a laptop type motherboard typically operates in -5c to 35c conditions. The payload will be well sealed and the processor will be generating its own heat as it consumes power which will help to keep the payload warm especially when you consider that the atmosphere will become thin at high altitude and so will have less oxygen and pressure.

It is a consideration to use hand warmers bonded to the batteries to keep them warm as they are rated at -20c to +65c, this will also help circulate heat around the payload. This technique has been used in other Weather Balloon projects especially when launching mobile phones within a payload.

Dave Akerman with much Weather Balloon experience writes of how he instead had to fit heat sync's when launching a Raspberry Pi fearing that overheating inside the payload could actually be a problem.

Being that hot air rises it will be best to mount any hardware that isnt dependant on heat at the lowest point of the payload, the hand warmers above this and then any hardware requiring heat above those. This all needs to be done whilst keeping the weight toward the bottom of the payload for stability in flight.

Visibility to Other Aircraft



A “corner” reflector will produce a radar echo to surface radar operating in the 200 MHz to 2700 MHz frequency range. The corners return the radar energy back in the direction it originated. This will be hung beneath the payload.

Commercial radiosonde radar refelctors are about 50cm tip to tip. A very lightweight reflector (less than 30g) can be constricted using carbon fibre kite spars and metalized plastic foil (as used in space blankets).

Most importantly a NOTAM (notice to airmen) needs to be given, *20160825met_researchballoonreleaseapplication.doc* needs to be filled in and sent to arops@caa.co.uk 28 days prior to launch, see www.caa.co.uk for more information. Contact with ATC (air traffic control) is imperative to ensure the balloon is launched whilst no other aircraft are overhead.

Handling the Balloon

Before starting the inflation process, be sure everyone on your launch team is wearing latex or vinyl gloves to keep the oils on your skin away from the balloon material.

Calculating Helium Required:
http://habhub.org/calc/
http://tools.highaltitudescience.com/