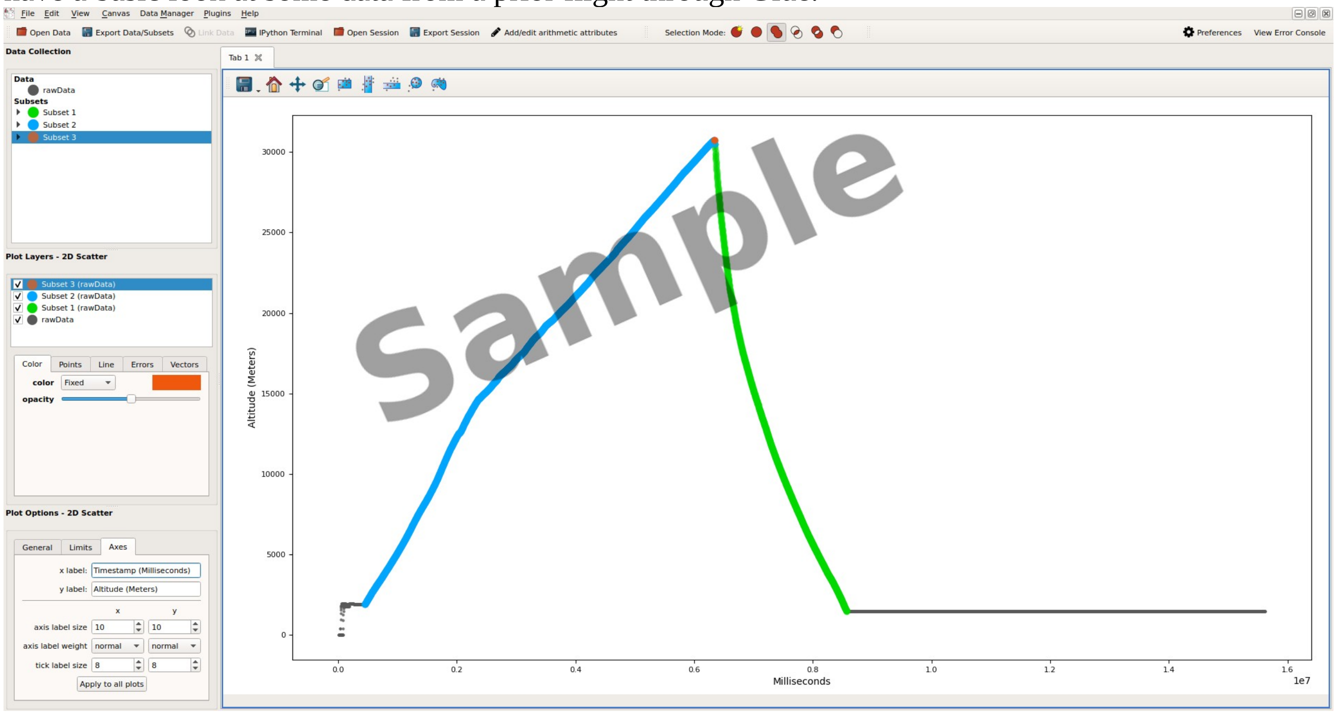
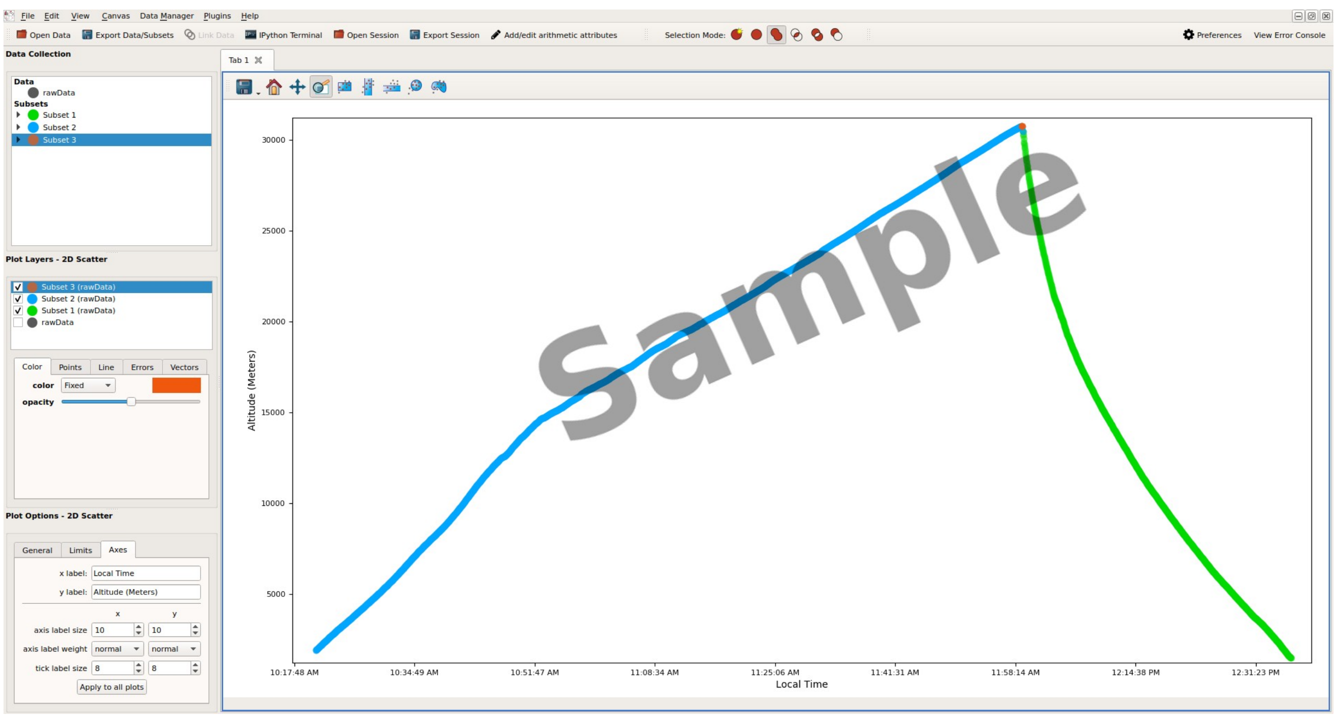
Now that we have done some surface level tests, we want to dig deeper into what our flight will be like and what is required of our system for it to survive and report accurate data the whole time. Lets have a basic look at some data from a prior flight through Glue.





Here are two screenshots of a payload flight with altitude in meters on the Y axis and timestamp on the X axis in milliseconds and local time, which was detected using a GPS that was part of this particular payload. Note that this graph is colored as opposed to the prior test graphs which was only in gray. This was done by creating new subsets by right clicking the data area and the colors were changed to be blue for ascension, green for falling, and orange to represent the peak. Data points can be added to subsets by using the selection mode and tools at the top of the chart window to zoom and select the data that belong together.

Zooming in closely on the data also changes the axes to have a more precise view. Zooming into the beginning and ends of these two charts we can pinpoint how long this payload flight was. Based on local time, the time of landing was 12:36:20pm, and the launch time was 10:20:50am, giving a duration of 2 hours 15 min 30 seconds. The millisecond timestamp is produced using the arduino's internal clock, which simply reports how long the current program has been running uninterrupted. The payload was launched 448852 milliseconds after startup, and landed 8572575 ms after startup. Subtract the launch time from the landing time gives the flight duration in milliseconds, and then dividing by 1000 converts to seconds which is then converted to minutes and hours by dividing by 60 twice. This results in 2.26 hours or 2 hours 15 min and 24 seconds, only 6 seconds off of our calculation using local time from GPS. Looking at the peak altitude of 30km (~98.5 kft) we can see that this flight was just about at the highest altitude that our balloons usually go, so this is a good case to examine for a full length flight, though it's good to prepare for some extra headroom in all designed specs. Flight dynamics are greatly influenced by the delicate balance of payload weight and how much propellant is used to inflate the balloon, and an underfilled balloon can take much longer than expected to reach a peak altitude to pop. To account for this, we will plan for a nice round 3 hour flight.

If we consider that we want a 3 hour minimum lifetime, we recall our earlier discussion of batteries usually having a "milliamp*hour" rating, so we would need 45mA * 3Hours = 135mAH minimum. Looking back at our battery information, we can now better compare how fit each battery type is for our needs. We compared 4 batteries from Energizer, Eveready and Industrial which are Alkaline batteries , a Nickel Metal Hydride rechargeable 9V battery, and a Lithium 9V battery. All of the battery technologies examined had capacities dependent on the amount of current being drawn. At around 45mA, the NIMH is 175mAH, Industrial is around 450mAH, and Lithium is 750mAH. Eveready however doesn't show the capacity rating at currents higher than 25mA, so its probably not ideal for our needs, however the others all appear to have sufficient capacity to run our system at least 3 hours.

Looking a bit closer, a NIMH battery 175mAH would power a 45mA load for (175/45) = 3.89Hours, compared to the 10 hours and 16.7 hours of the industrial and lithium batteries respectively. All of these varieties will perform differently based on their temperatures, and can cut out when they get too cold, so insulation against the cold in a final design can be beneficial. The battery capacities will also slowly drop over time while in storage, so an actual NIMH capacity could be slightly lower than what is specified by the datasheet, so we should pick a battery with a comfortable amount of head room. For our purposes, industrial 9V has much more energy than we need and a quick search on amazon will find a unit price of about $1.70 per battery, but lithium could also be favorable for the wider temperature tolerance and lighter weight at the greater cost of around $7 per battery.

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Input Voltage (Note 1)	V_{in}	20	V
Output Short Circuit Duration (Notes 2 and 3)	–	Infinite	–
Power Dissipation and Thermal Characteristics			
Case 318H (SOT−223)			
Power Dissipation (Note 2)	P_D	Internally Limited	W
Thermal Resistance, Junction−to−Ambient, Minimum Size Pad	$R_{\theta JA}$	160	°C/W
Thermal Resistance, Junction−to−Case	$R_{\theta JC}$	15	°C/W
Case 369A (DPAK)			
Power Dissipation (Note 2)	P_D	Internally Limited	W
Thermal Resistance, Junction−to−Ambient, Minimum Size Pad	$R_{\theta JA}$	67	°C/W
Thermal Resistance, Junction−to−Case	$R_{\theta JC}$	6.0	°C/W
Maximum Die Junction Temperature Range	T_J	−55 to 150	°C
Storage Temperature Range	T_{stg}	−65 to 150	°C
Operating Ambient Temperature Range	T_A		°C
NCP1117		0 to +125	
NCV1117		−40 to +125	

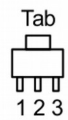
Stresses exceeding those listed in the Maximum Ratings table may damage the device. If any of these limits are exceeded, device functionality should not be assumed, damage may occur and reliability may be affected.

1. This device series contains ESD protection and exceeds the following tests:
   Human Body Model (HBM), Class 2, 2000 V
   Machine Model (MM), Class B, 200 V
   Charge Device Model (CDM), Class IV, 2000 V.
2. Internal thermal shutdown protection limits the die temperature to approximately 175°C. Proper heatsinking is required to prevent activation. The maximum package power dissipation is: $P_D = \frac{T_{J(max)} - T_A}{R_{\theta JA}}$
3. The regulator output current must not exceed 1.0 A with V_{in} greater than 12 V.

ELECTRICAL CHARACTERISTICS

(C_{in} = 10 μF, C_{out} = 10 μF, for typical value T_A = 25°C, for min and max values T_A is the operating ambient temperature range that applies unless otherwise noted.) (Note 4)

Characteristic	Symbol	Min	Typ	Max	Unit
Dropout Voltage (Measured at V_{out} − 100 mV)					V
(I_{out} = 100 mA)		–	0.95	1.10	
(I_{out} = 500 mA)		–	1.01	1.15	
(I_{out} = 800 mA)		–	1.07	1.20	
Output Current Limit (V_{in}−V_{out} = 5.0 V, T_A = 25°C, Note 6)	I_{out}	1000	1500	2200	mA



SOT−223

We now know how much power the system consumes and would require for at least a 3 hour period, but what is the upper limit for the power that can be supplied by the regulator on the arduino UNO? Checking the product page for the UNO board on arduino.cc will reveal that the arduino UNO uses a “low dropout” linear voltage regulator with a part number NCP1117. Recall from our power segment that linear regulators create a stable output voltage by absorbing excess voltage and converting that into heat. This means that they can only output voltages lower than their input, and the “low dropout” voltage refers to how low the input can go and still supply a stable 5V output. A very common LM7805 5V regulator has a 2V dropout voltage, which means that it will hold a stable 5V output as long as the input is always at least 2V above the output, or 7V. So 7V is the minimum input voltage for a 7805 regulator, while the NCP1117 on a standard UNO has around a 1V dropout voltage, setting the minimum input voltage at 6V to provide a stable 5V output. Checking the NCP1117 maximum ratings, we can see it has short circuit protection and thermal shutdown, which means that if the output is simply shorted, it will not self destruct, instead it will actively shut itself down once the regulator package reached approximately 175 degrees C. The regulator is short circuit protected up to 1.5 amps. According to the maximum power dissipation equation referenced in the maximum ratings, this regulator can dissipate around 937milliWatts of power, which corresponds to the excess power the regulator absorbs to keep the output stable. If we assume the input is 9V and the output is 5V, then the regulator will absorb 4V excess, and 937mW of power would equate to 937mW/4V =234mA of current or 230mA just for a more round number.

This gives an idea of some of the limitations of the regulator, if the current a given system needs 200mA, it may be time to use a more capable power regulator like a switching regulator rated for higher currents than this chip. Problems may also occur if a USB port is being used to power the system but has low current capabilities. Some USB ports have maximum current capabilities of only 100mA, so this limitation can definitely cause some hiccups during development, so a good USB supply is recommended, and systems should be tested using their own supply as well to make sure their regulators show no trouble. One thing to note, it is not recommended to power an arduino system from USB and an external power supply at the same time, in some situations this can feet power back into the USB port and damage the host device. For this reason, the 5V supplying power to the arduino chip should be exclusively from USB or external power, *never* from both at the same time.

This so far addresses the arduino Uno's regulator capacity, which is simply how much power can be supplied to the core and supplied to external sensors through the 5V pin. But notice from the integration section that the activity indicator is receiving its power directly from a digital pin. If we assume the led will absorb 2V, then the power drawn from a 5V pin would be (5V-2V) / 200Ohm = 15mA. This isn't very much and is safely within the 20mA limit normal leds have, but it's safe to guess that the arduino core has an upper limit on how much current can be supplied through the digital pins. The question is, how much?

The core of the arduino is a micro controller chip called Atmega328, and section 26.1 of its 400+ page datasheet is the controller's absolute maximum electrical characteristics.

26.1 Absolute Maximum Ratings*

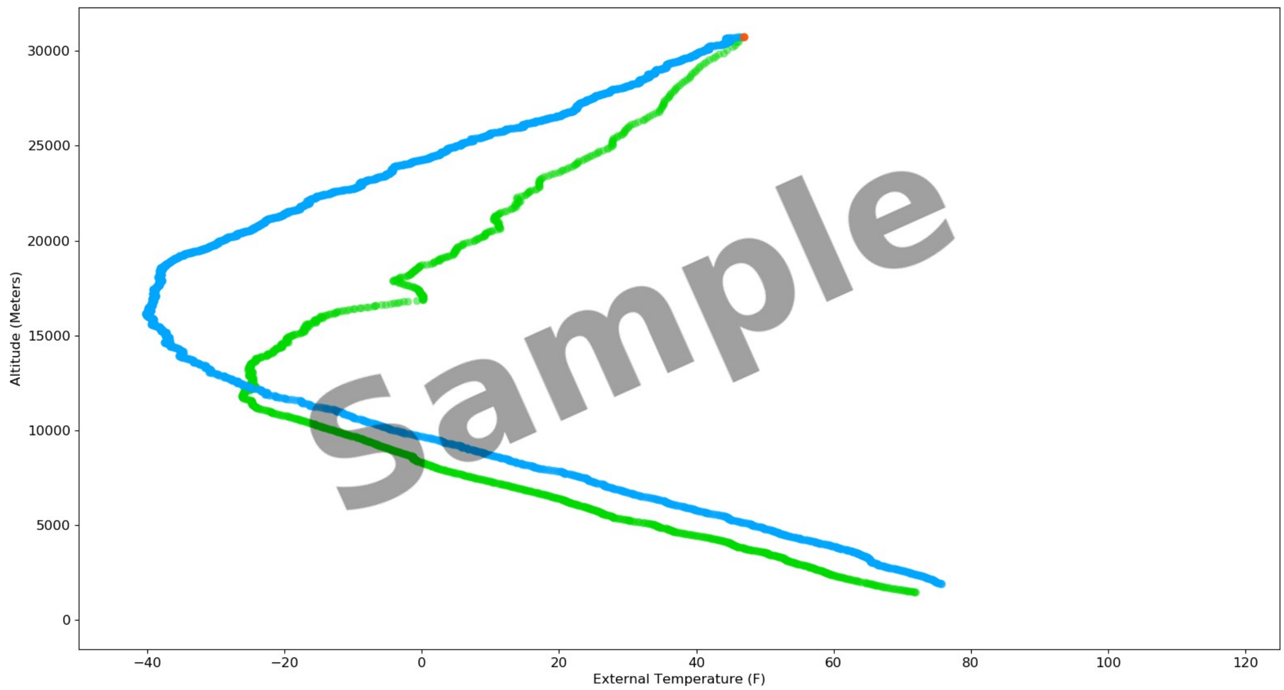
Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground	-0.5V to V_{CC}+0.5V
Voltage on $\overline{\text{RESET}}$ with respect to Ground	-0.5V to +13.0V
Maximum Operating Voltage	6.0V
DC Current per I/O Pin	40.0 mA
DC Current V_{CC} and GND Pins	200.0 mA

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

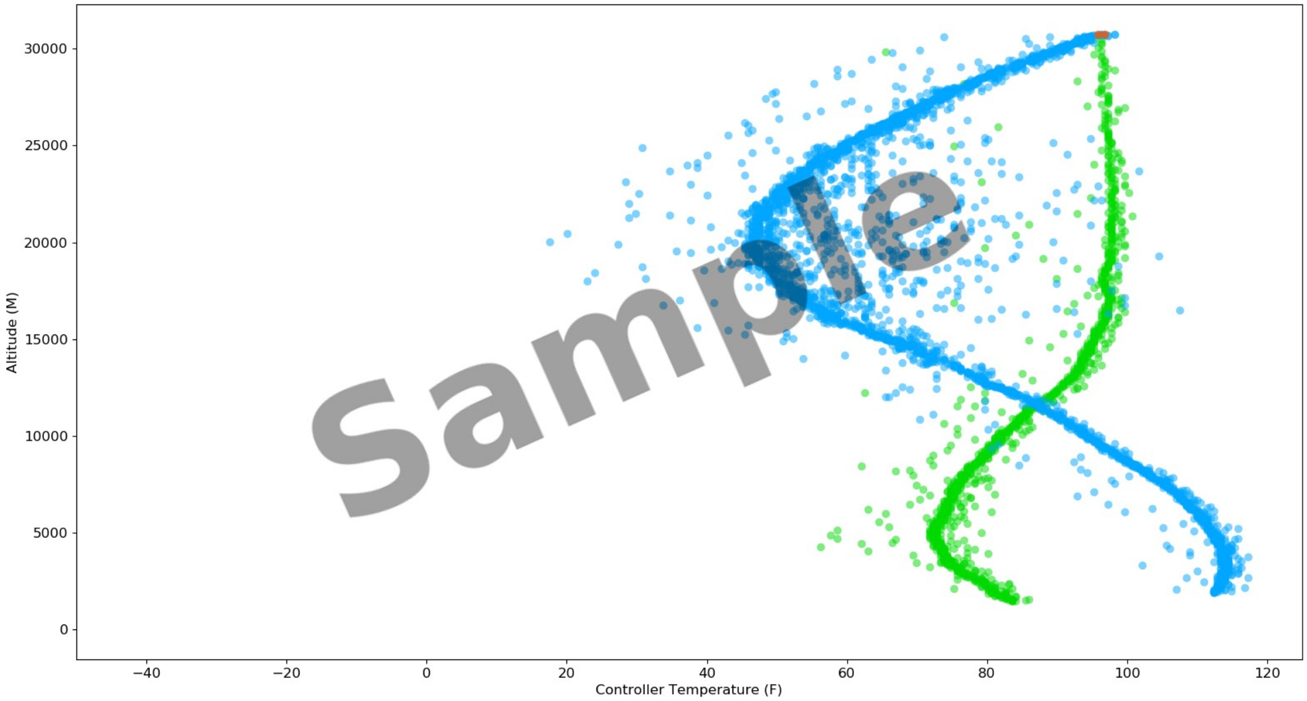
The bottom of the maximum ratings suggests that the maximum current that any of the digital pins should supply or consume is 40mA while there is a total maximum of 200mA current that can pass through the power pins. Each digital pin can act as a small 5V supply to supply current or as a ground pin to drain current, and as the chip runs through its host program it will consume some additional amount of current. Since all current used by the chip must enter or exit through the VCC or Gnd pin, the 200mA limit refers to the combination of power consumed by computation with the power consumed by each digital pin simultaneously. It then becomes obvious that even without considering the power consumed by running the program, powering 5 LEDs with 40mA or 10 LEDs at 20mA would reach the limit of the core and would damage the chip permanently. This should show that the arduino core is not meant to supply significant amounts of power, but rather is meant to control other devices while consuming as little power as possible. Common LEDs usually have 20mA as a maximum rated current, so usually it's good practice to design LED circuits using the arduino well under the 20 mA maximum current for the leds. Controlling devices however sometimes do need to control external circuits that can be higher than 5V and 20mA. In those situations transistors or relays can be used as electronic switches to allow the arduino to control systems dealing with much higher amounts of power. This course will not cover the use of those, but generally in order from low to high power capabilities, there are 2 main transistor types, bipolar junction transistors (BJTs), metal oxide semiconductor field effect transistors (MOSFETs), and then relays of various sizes. Transistors are great because they're very light weight and physically small, relays are great because they can handle enormous amounts of power, and each device will have their own datasheets that show what kinds of power they are capable of, so determine what your power requirements are, and that will guide what switching device you should use.

The arduino core chip and leds have maximum ratings, and the sensors the arduino controls have their own maximum ratings, but they usually consume a consistent amount of power as opposed to the arduino core that will consume varying amounts of power depending on the contents of the program the chip is running. Lets examine some of the other maximum ratings of this system.

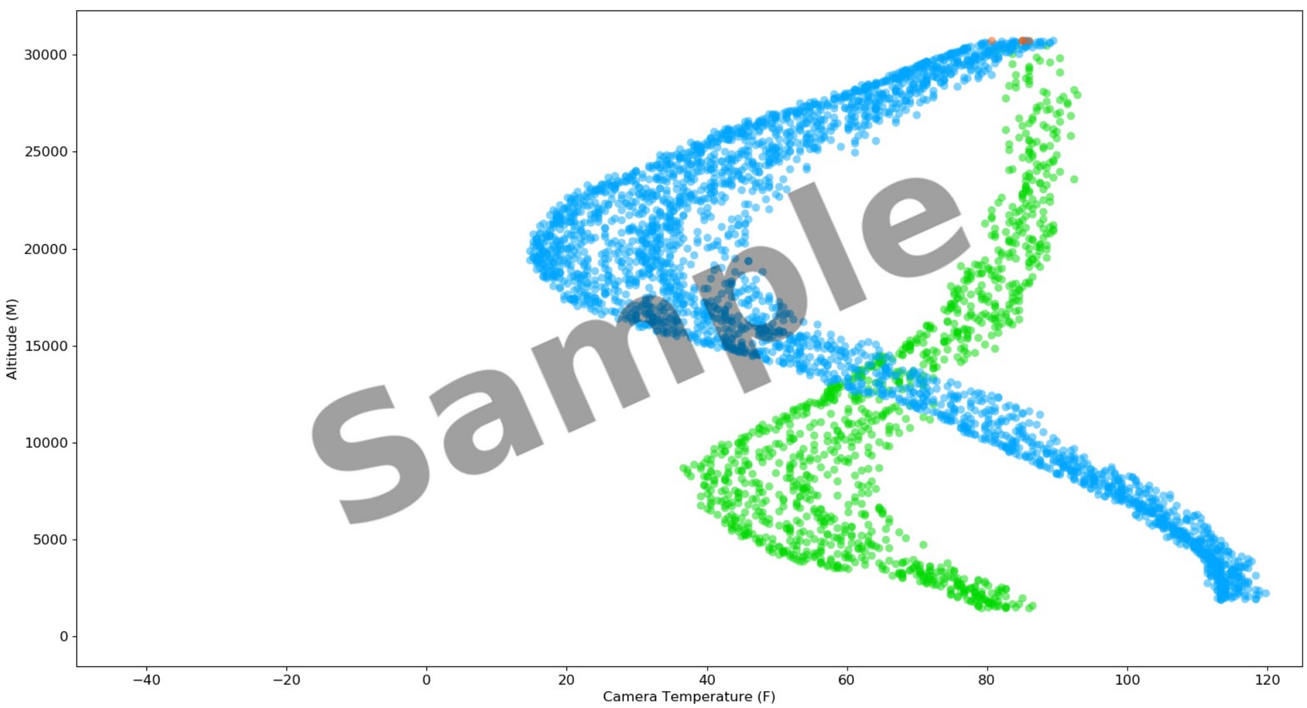
Power consumption is one type of maximum value we should consider, but we want our system to run from ground altitude up to 100kft, which has hot regions and cold regions throughout. Looking back at the arduino core chip maximum ratings, we can see that the chip can operate just fine as long as it’s between -55*C (-67F) and 125*C (257F). Lets compare this to some data from a prior balloon launch.



Here is a graph with payload external temperature in Fahrenheit on the x axis and altitude in meters on the y axis to orient altitude intuitively. Since a payload travels up, then after the balloon pops it falls back down, here those regions are colored blue for ascending, green for falling, and an orange blip to show the moment the balloon popped. We can see that the payload can experience temperature swings that approach the arduino’s cold limit. To compare, -50*C is -58*F, and 30km is 98,000ft. Sensors also have some imperfections, for example, a temperature with a lot of thermal mass will react to temperature fluctuations slower than a sensor with very little thermal mass. This means that while the temperature sensor read around -40*F, the air could have been slightly colder but the temperature sensor having moderate thermal mass could have delayed the readings of the sensor before the ambient temperature starts to climb again. Combine that with the fact that this particular payload was constructed with black carbon fiber and some elements inside the payload were producing heat, and its easy to guess that some of that heat inside the payload could have conducted heat through the walls and wires to the temperature sensor, raising its readings higher than the actual air temperature. This payload happened to contain 2 internal chambers to house the main controller and a camera that produced significant amounts of heat. The following two charts show the controller temperature and the camera chamber temperature respectively with the same color scheme as the chart for external payload temperature. Note that the green path on the external temperature chart follows the ascent profile much more closely than the internal payload temperatures, which shows how the payload will “store” heat energy that can affect temperature readings. Comparing the external readings to the internal ones makes it obvious that this heat capacity of the payload could have affected the temperature readings slightly.



30000
25000
20000
15000
10000
5000
0
Altitude (M)
−40
−20
0
20
40
60
80
100
120
Controller Temperature (F)
Sample



30000
25000
20000
15000
10000
5000
0
Altitude (M)
−40
−20
0
20
40
60
80
100
120
Camera Temperature (F)
Sample

Lets dig one layer deeper. What about the limitations of the sensors in our example payload? We have some sample data to compare to our sensor limitations, so lets see how well our example payload would perform in these conditions.

Example payload components & operating temperature ranges:

- Arduino Uno: -55C – 125C
- AM2320 Humidity Sensor: -40C – 80C & 0% - 99.9% Relative humidity
- MPL115A2 Barometer: -40C – 105C & 50kPa - 115kPa
- Photocell: -30C – 75C
- Logging
  - Sd card: -25C – 85
  - Chip on logging board: ? - 150C (logging often warms the chip, so no cold limit)

These stats were gathered from the sensor’s datasheets, and we can see that each device usually has a lower and upper limit for it’s operating temperature. Additionally, sensors usually have a range of sensitivity for the value that it can measure. For example, the MPL115A2 barometer cannot be relied to be accurate for pressures above 115kiloPascals and below 50kiloPascals. The sensor won’t necessarily be damaged when outside of its limits, but the data it reports will no longer be accurate. The lowest pressure recorded from the prior launch was 1.5kPa, which is far lower than this particular sensor’s 50kPa limit. This is because the prior launch that produced those example charts used a wider range and more expensive barometer that was designed with use as an altimeter in mind, while our kit for our example payload is optimized for low cost instead. From prior launches I can tell you that 50kPa corresponds to around 6km or 20kft, which is only around a fifth of the full 100kft range we would expect from a full range flight. This example payload is therefore not fit for a full range flight because of pressure limits and because of temperature limits from several sensors since we can see the atmosphere gets colder than several sensors can handle without protection. When designing a payload, pay careful attention to all of the maximum ratings for the sensors to ensure every component is rated to survive the extreme nature of the flight.