DIY CO2 INCUBATOR BIOREACTOR FOR MAMMALIAN CELL CULTURE

Project originally developed by Dr. Andrew Pelling (Dec 2014)

Update: Nov 17, 2024

This is a copy of a blog post originally hosted on the website pellinglab.net which no longer exists. The blog post related to this project was adapted into pdf format for posterity as I know that many people found the information useful for their own builds. Hyperlinks in the text below are kept as-is, however there are no guarantees that they will still be working.

Please note that this project is no longer being maintained, however the GitHub repository is still available: https://github.com/pellinglab/DIY-Incubator-Dec2014

1. INTRODUCTION

Biological incubators/bioreactors are a vital piece of equipment required for the growth of many cell types. Incubators are ubiquitous in any cell biology or DIYBio lab and at their heart they are simply warm, humid boxes with temperature and atmosphere regulation. Incubators for bacterial culture are cheap and easy to come by because they are essentially warm, humid boxes. Many DIY designs exist and are routinely employed by DIYBio and academic labs. These incubators are great for anyone wanting to grow, genetically modify and manipulate bacteria. However, if one wants to grow mammalian cells (eg. human cells, mouse cells, etc), incubators are required that control both temperature and the CO2 content of the atmosphere. Maintaining an atmospheric CO2 concentration of 5% is essential for maintaining the pH of common mammalian cell culture media (ex. DMEM) . There are some CO2-independent media formulations on the market but they can be expensive, the cells never seem to grow very well and they often look “a little funny”.

So what is a cheap/poorly funded academic and/or DIYBio person to do? Commercial incubators routinely cost between ~$5,000 (and up!) and remain fairly expensive even on the used market. The high cost of these incubators is not only prohibitive for the DIYBio community, but also for academic labs. In our current funding landscape, countries are continually slashing their science funding budgets yet these these incubators remain a vital resource to researchers in the biomedical sciences. Moreover, scientists in emerging/developing nations may not have access to levels of funding that would permit them to purchase, and maintain, CO2 incubators.

My lab is often asked if we have developed plans for a DIY CO2 incubator because of these issues. After spending some time looking around at what is available on the web already, I found myself unsatisfied with existing forum discussions and projects. So I've decided to give it a shot myself.

There are really three main challenges that need to be solved in order to implement a DIY CO2 Incubator:

- Obtaining the right CO2 sensor.
- Controlling the CO2 content of the incubator atmosphere at 5%.
- Finding a CO2 source that is easy to acquire.

Over the past few months I have developed a working DIY CO2 incubator while on sabbatical at SymbioticA, University of Western Australia. I gave myself the challenge of building it for under $500 and using as many recycled/found materials as possible. The design presented here addresses the three challenges above and I built mine for ~$350 (by far the most expensive part was the actual CO2 sensor at ~$230). Hopefully the information on this site is of use to the DIYBio and global academic communities.

The completed DIY CO2 Incubator for Mammalian Cell Culture Incubator maintains 37.0 ± 0.1 Celsius and 5.0 ± 0.2% CO2

Disclaimer: The plans and code provided here represent the first shot at a workable solution. They are seriously "Quick and Dirty". They are not perfect but the incubator functions well and will support the culture of mammalian cells. You are encouraged to use the information on this site as a starting point for your own projects. Please share any changes you have made to improve the implementation of this incubator and contact us if you have questions.

These instructions are presented in 5 different sections:

1. **INTRODUCTION**
2. **DESIGN AND BUILD**
3. **ARDUINO AND CIRCUITS**
4. **PROOF IS IN THE PUDDING**
5. **POSSIBLE IMPROVEMENTS**

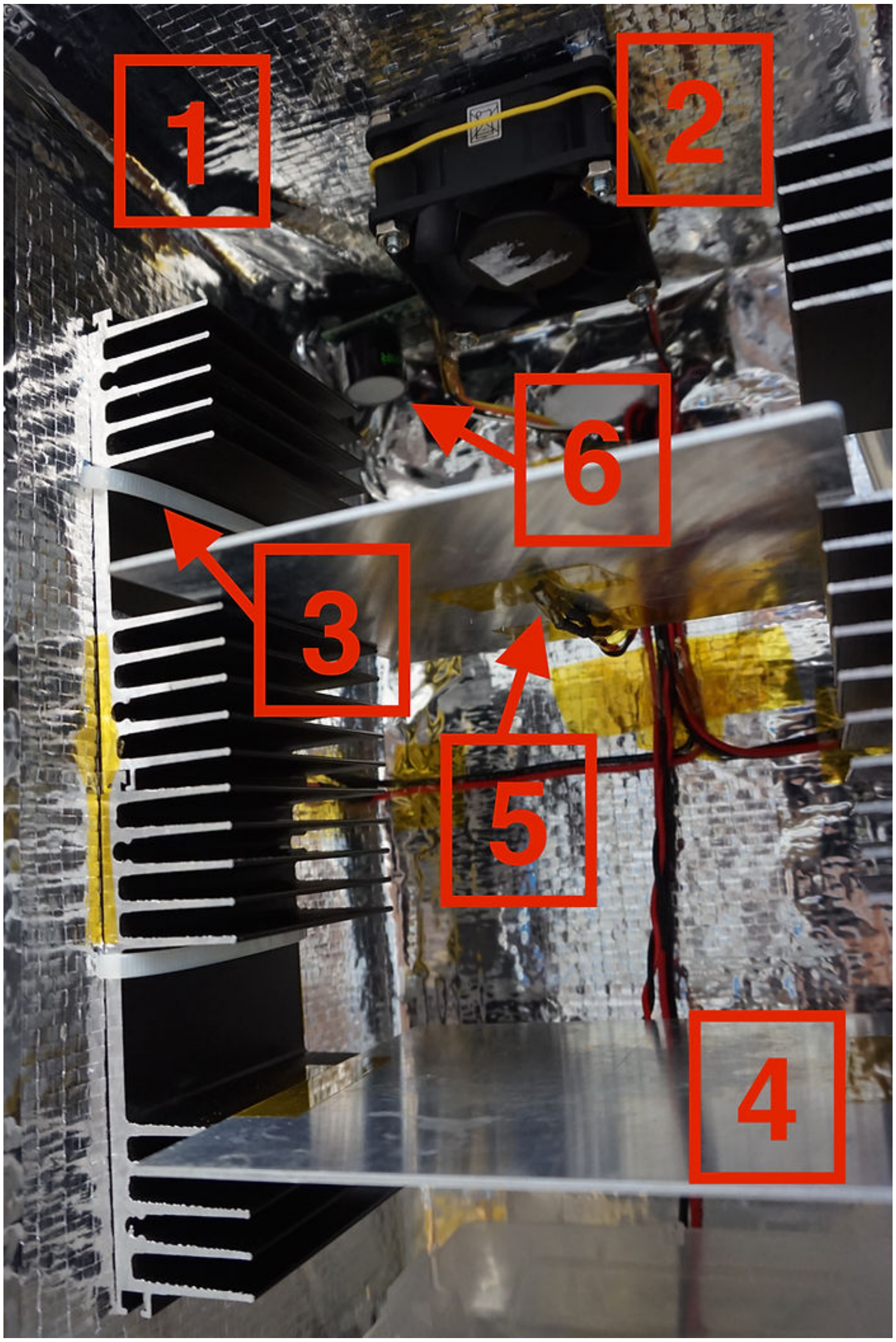
This work was made possible through financial and in-kind support from the Raine Medical Research Foundation, SymbioticA, The School of Anatomy, Physiology and Human Biology (UWA), The Natural Sciences and Engineering Research Council and The University of Ottawa.

2. DESIGN AND BUILD

An example parts list is here. You can find all sorts of similar items for free or cheap if you look hard. I did not buy all of these items, I found most of them. The list is only provided as a reference.

First you need a box of some sort. I found a Styrofoam box in the garbage one that is commonly used to ship stuff to labs. You could also consider using an old cooler/fridge or even buying a Styrofoam box from any hardware store. The box was initially lined with a space blanket I got after finishing a marathon. However, there are easier ways to acquire this material by going to any camping/outdoors/building supply store (get something like radiant thermal insulation). This material is nice because it reflects heat and can be wiped down. It also forms a barrier between the heating pads and the Styrofoam. I also considered cutting up some old cookie tins or air ducts (might still do this at some point). I then sealed any seams with foil tape (the silver foil for ducts, you can get it at any hardware store). This isn't really "functional" it just makes everything look like its from space which is always worth doing.

I pulled a 12V DC fan (to move air around the incubator) and some heatsinks from an old desktop computer and a dead amplifier. At the top of the box I mounted the fan with screws. A nice bonus was that by mounting the heatsinks on the walls of the box they can do double duty as mounts for shelving. I used kapton tape to attach some cheap flexible heaters to the backside of the heatsinks (so you cant really see the heaters in the pictures). The heatsinks (and heaters) were mounted to the walls with cable ties. Heaters were wired in parallel with 18 AWG wire (this is important because they draw a lot of current and thin wire will not cut it). I also found some scrap metal that was the right size for shelves. Since the shelves are metal, they conduct the heat from the heatsinks very well. Finally, I got myself some 1-wire DS18B20 digital temperature sensors. You can find cheaper temperature sensors but these are nice because they are addressable, can be daisy chained and only use one pin on the Arduino. I chose to attach my sensors to the underside of each shelf in order to measure the temperature as close to the cells as possible.



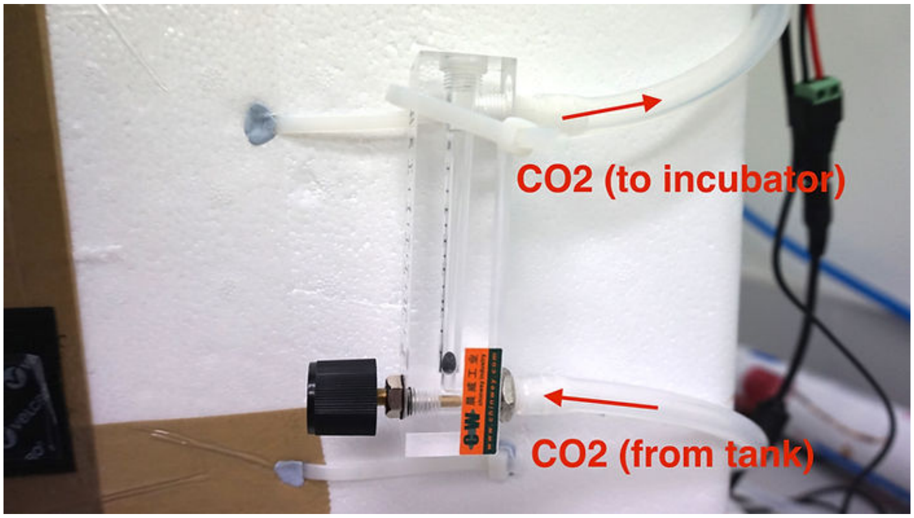
[1] Thermal insulation glued to inside of styrofoam box. [2] 12V DC fan from old desktop computer. [3] Heatsink with unseen flexible heater fixed to the underside (between heat sink and thermal insulation). The heatsink is mounted with cable ties that run through the walls and are tightened from the outside. [4] Scrap metal shelving. [5] DS18B20 temperature sensor (1 of 2), fixed to underside of shelf with Kapton tape. [6] GC-0017 CO2 sensor.

Finally CO2. There is a very common and cheap MG811 CO2 sensor on the market. I bought this without looking closely at the specs and found out it only goes up to 1% (10000ppm). So its not useful for our purposes. After spending a lot of time looking around, I settled on a company in the US (CO2Meter.com). They sell a sensor that works really well with Arduino (G-0017 0-20% SprintIR) for ~$230. This is the most expensive part of the incubator. I was not able to find an equivalent sensor cheaper anywhere else, or reverse engineer this one.

For the CO2 source – I have successfully utilized 60L CO2 tanks from SodaStream. After buying your first tank they are ~$20 to refill. I also considered larger Paintball tanks but never tried them. One might also acquire a large tank from a commercial source, but these can represent a significant safety risk and you should be prepared to take appropriate precautions (lots of instructions can be found online). If you use a SodaStream/Paintball tank you will need an appropriate adaptor to put on an easily acquired 2-stage CO2 regulator. I found an old one lying around in the building I'm working in. I used the regulator to ensure an outlet pressure of ~15kPa. And last but not least, I decided to just use a 12V solenoid to turn on/off the flow of CO2 to the incubator.

I have found my SodaStream tank to last ~1.5 weeks under normal use. From the CO2 canister I ran tubing to a syringe filter, then to the solenoid and finally to a flow meter. From the flow meter I ran more tubing into the incubator at a point as far as possible from the CO2 sensor. You can always give yourself an excess length of tubing to have flexibility for placing the outlet wherever you like inside the box.

[1] 60L SodaStream CO2 Canister. [2] CO2 Regulator. [3] 12V Solenoid Valve. [4] Tubing and 0.2um filter.

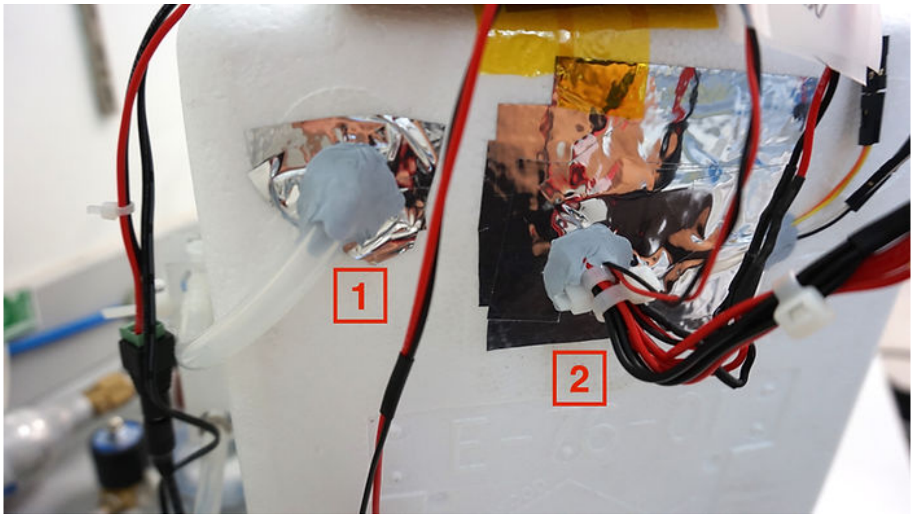


Flow valve mounted to the side of the incubator. This is used to maintain a slow bleed of CO2 into the box.

Hint: A visit to a local aquarium shop/forum, or a shop/forum for home beer brewing, might be worthwhile. They should have advice on all sorts of CO2 equipment, tubing, regulators and solenoids.

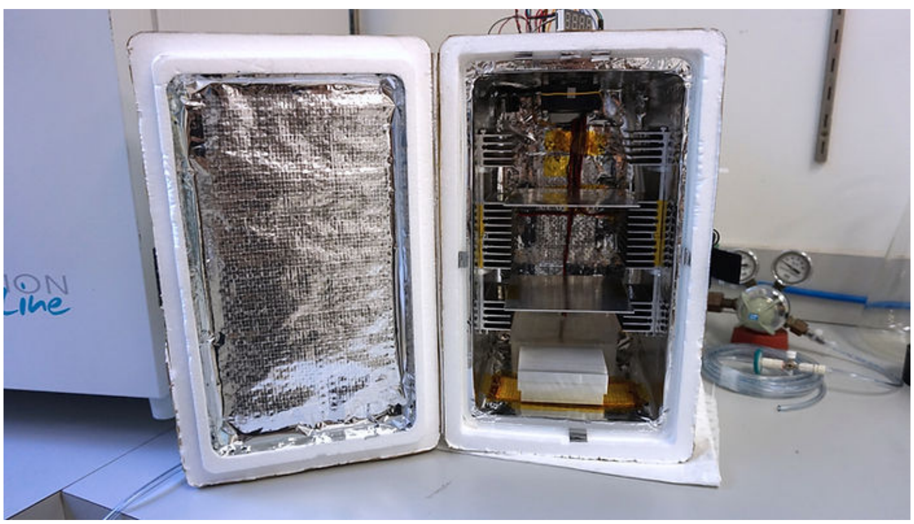
All the wires were passed through a small hole in the back of the box I cut out with a utility knife (keep the piece you cut out in order to fill the hole afterwards!). After passing the wires, I wedged the Styrofoam back into the hole and used the foil tape to seal it up inside/outside the box.

Making the incubator airtight is a good idea. You use much less CO2 and your SodaStream lasts longer. I found that Blue Tak was a great way to seal up small holes (from the outside) in the Styrofoam which were created to run wires, tubing and cable ties. In my setup, I found observed a CO2 consumption of ~0.2L/hr when the door remains closed.



Blu-tak worked really well to plug up small holes in order to make the incubator more air tight. [1] CO2 into the incubator. [2] Wiring from sensors, fan and heaters running out to the Arduino control.

In the back of the incubator I placed a small reservoir of water to keep the environment humid in order to minimize evaporation of cell culture media. To prevent condensation you really want to insulate the box as much as possible, especially around any openings (like the door!). I used some velcro in order to to pull the door tight.



[Lots of Random Pictures at Various Stages of the Build]

... also many more pictures in the following pages ...

3. ARDUINO AND CIRCUITS

The electronics and code are embarrassingly primitive. In general terms, an Arduino UNO simply monitors temperature and CO2 content and turns on/off the heaters or open/closes the solenoid valve as necessary to maintain the various setpoints. I am 100% sure the whole setup can be improved and made more efficient (some areas for improvement are listed later on). However, even though this embodiment is pretty dumb it works and mammalian cells can be grown and differentiated

The Arduino is essentially operating four simple circuits:

1. Control of a 12V DC fan.
2. Reading temperature sensors.
3. Reading the CO2 sensor.
4. Relay control to supply 12V to the heaters (on/off) or to the solenoid (open/closed).

That's it! Below these circuits will be described. It is recommended to build/test each circuit before throwing it all on a single board. Here, the whole thing was prototyped on a breadboard and then locked down on perfboard. Depending on your setup, this is probably a good ideas as the current draw by heaters can be high and may eventually damage your breadboard.

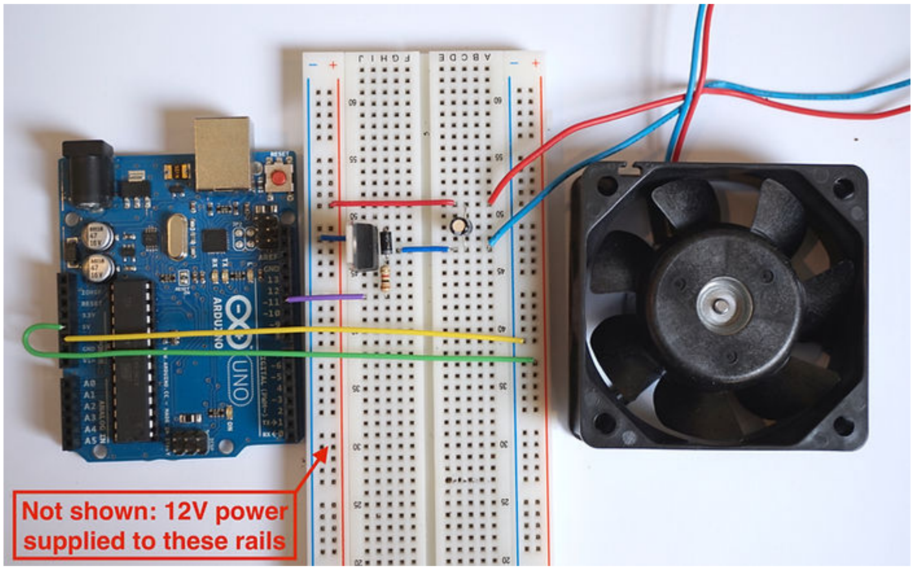
IMPORTANT NOTE: In the breadboard images below I have not shown the 12V power supply. However, you will need it! Using a DC barrel adaptor, connect a 12V supply to one set of rails and the Arduino 5V to the other (with common ground). In the images I indicate which set of rails the 12V supply should be connected to.

Download the Entire Arduino Control Code

PWM DC Fan Control

Components needed: 1x TIP120 transistor, 1x 1kOhm resistor, 1x 100uF cap, 12V DC Fan, 12V power supply

This is a very common setup. You can visit this Instructable for more info. In the current Arduino code the fan is connected to a PWM Pin, however, the fan is always on so PWM control is not really being exploited. You could just run the fan off the 12V supply and remove any Arduino control completely. In potential redesigns of the setup it might useful to have the fan come on at certain times, or change speed, etc. For this small box setup, I found that having the fan continuously on helped to keep the atmosphere well mixed and the temperature fairly stable.



Circuit for PWM Fan Control

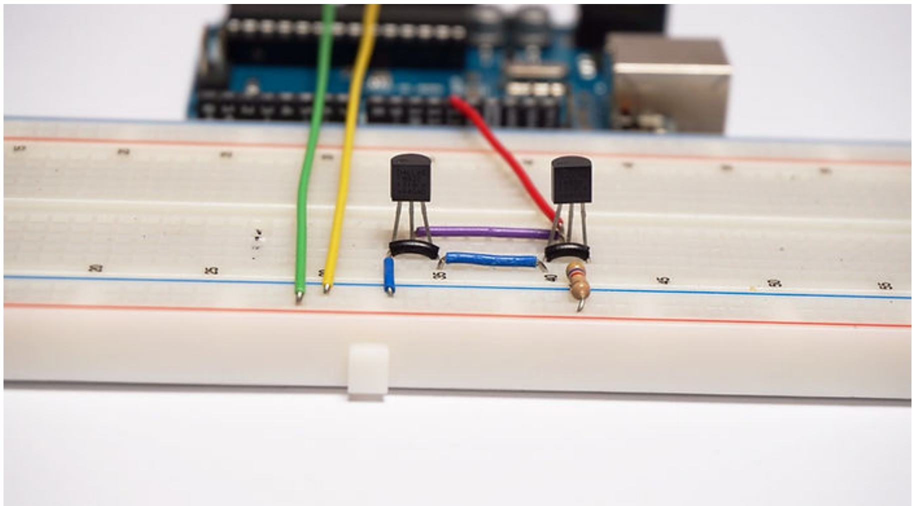
More Pictures and Circuit Diagram

Arduino Test Code for Fan Only

Reading the Temperature Sensors

Components needed: 2x DS18B20 sensor (or as many as you like), 1x 4.7kOhm resistor.

For temperature I decided on using two DS18B20 sensors wired parasitically. Below is a circuit for two sensors, however the wiring to add as many sensors as you like is very simple (follow the link in the previous sentence). In general, the outer legs of each sensor (legs 1 & 3) are wired together and then to ground. Data is collected on the middle leg of each sensor. These are wired together and then to 5V via a 4.7 kOhm resistor. In this specific example, readings are made from Pin 12 of the Arduino which is wired to leg 2 of one of the sensors. Each sensor identifies itself automatically on the bus and data from individual sensors is easily recorded.



Circuit for reading two DS18B20 Temperature Sensors

To use the sensors in the Arduino you will need the OneWire and Dallas Temperature Control Libraries. The test code provided here is for two sensors and the data is being collected on Pin 12. Its is straightforward to change the code for a single sensor, or to add more.

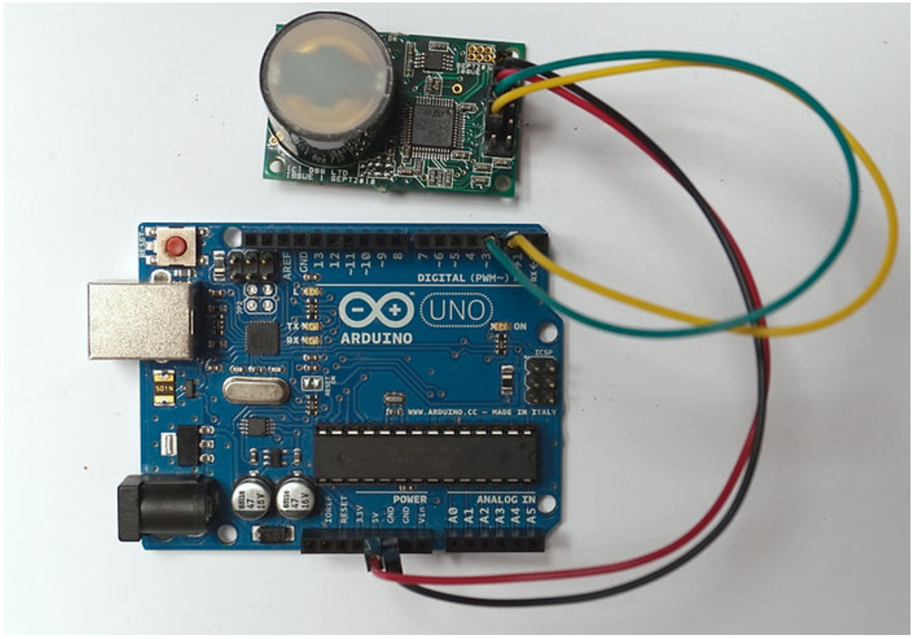
More Pictures and Circuit Diagram

Arduino Test Code for Two DS18B20 Temperature Sensors

Reading the CO2 Sensor

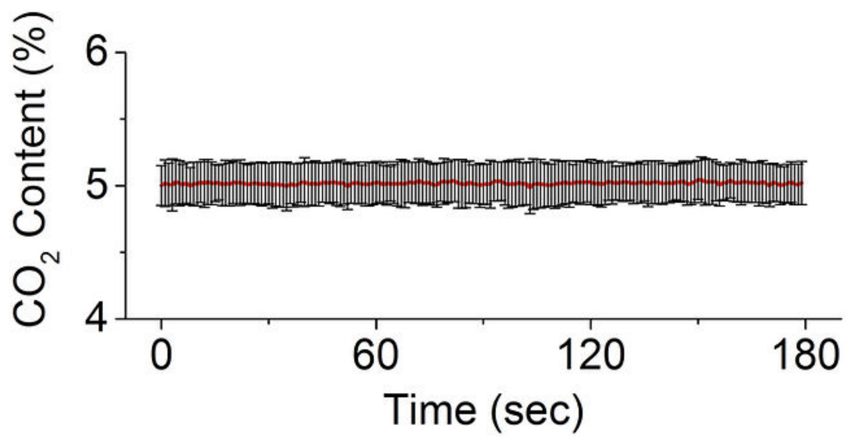
Components needed: 1x GC-0017 CO2 Sensor.

For CO2 sensing I decided to employ an NDIR based sensor from co2meter.com (GC-0017, 0-20%). There are only 4-pins we care about on the sensor (GND, 3.3-5.5VDC, Rx, Tx) and its very easy to hook up.



Luckily an Arduino library was developed for this CO2 sensor so I didn't have to do much work to get it up and running. I recommend reading this forum which has the library and some example code (also maintained on github a big shoutout to DirtGambit and Rob Tillaart for developing this). I am currently using the 1.0 version of the library on page 1 of the forum with no problems. If you decide to add more CO2 sensors to your setup you just need to define more software serial ports.

To convince myself the library and Arduino were working, I made sure measurements from the Arduino matched values obtained by the official company software. I also checked the calibration by exposing the sensor to a commercial premix of 5%/95% CO2/Air gas from BOC. Upon exposure to the 5% CO2 premixed gas I obtained CO2 readings from 3 different sensors. To do this I placed each sensor inside of an airtight container with a gas inlet and a syringe acting as an small outlet. Under pressure, I flooded the premixed gas into the box and started recording. For each sensor I made three 180sec recordings and then averaged all 9 measurements together to produce the plot below. On average, exposure to a commercial premixed gas of 5%/95% CO2/Air, resulted in a stable CO2 reading of 5.01±0.15%. This is well within the noise characteristics of the sensor. Not bad.



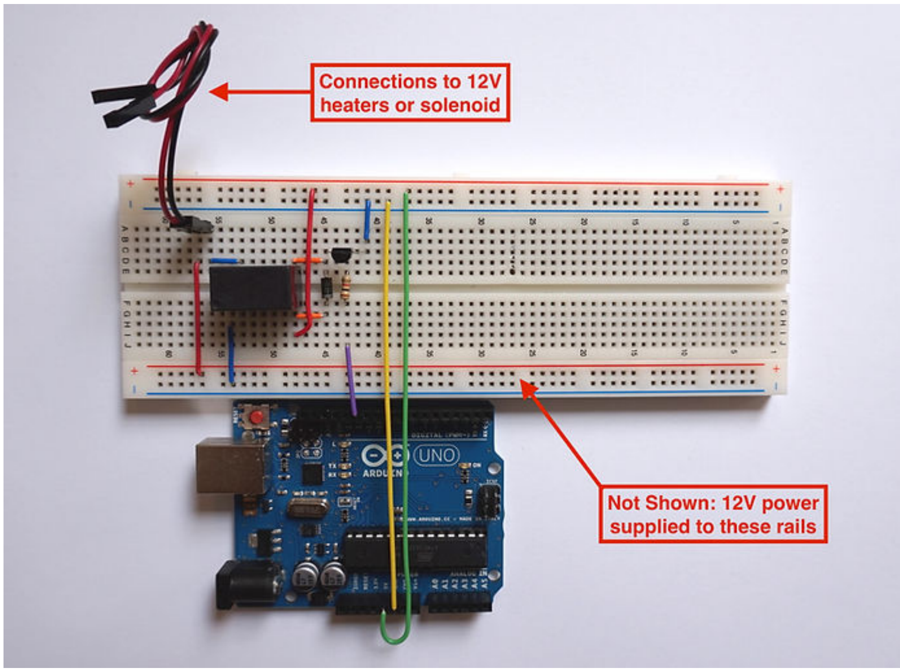
More Pictures

Arduino Test Code for CO2 Sensor Only

Relay Control over the Heaters and Solenoid

Components needed: 1x 2N2222 transistor, 1x 1kOhm resistor, 1x 1N4007 diode, SPDT relay, 12V power supply.

I employed two identical relay switches to supply 12V power to the heaters or the solenoid. The pictures and code here are for a single relay, so you will have to double up. I am using a pretty standard SPDT relay from sparkfun which is very easy to setup. This one is the 5-pin variety so note that the wring will change very slightly for 6-pin SPDT relays. There are also lots of pre-built relay modules out there that are simple to implement.



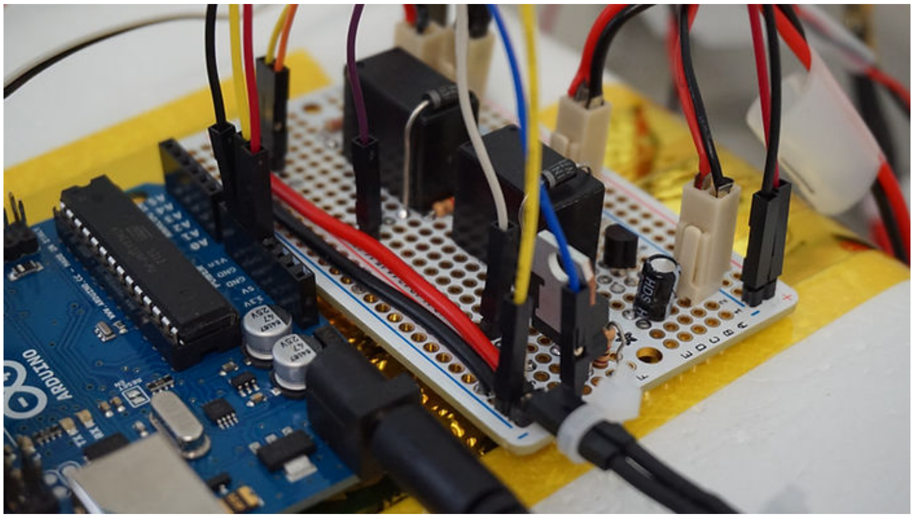
Controlling a relay is very easy. Setting an Arduino digital pin HIGH allows one to employ a transistor to trigger the relay switch with 5V from the Arduino. Once the switch has been triggered, 12V power can supply your load. Setting the digital pin LOW closes the switch. Therefore, the relays can be used to selectively supply power to a heater or to a solenoid.

More Pictures and Circuit Diagram

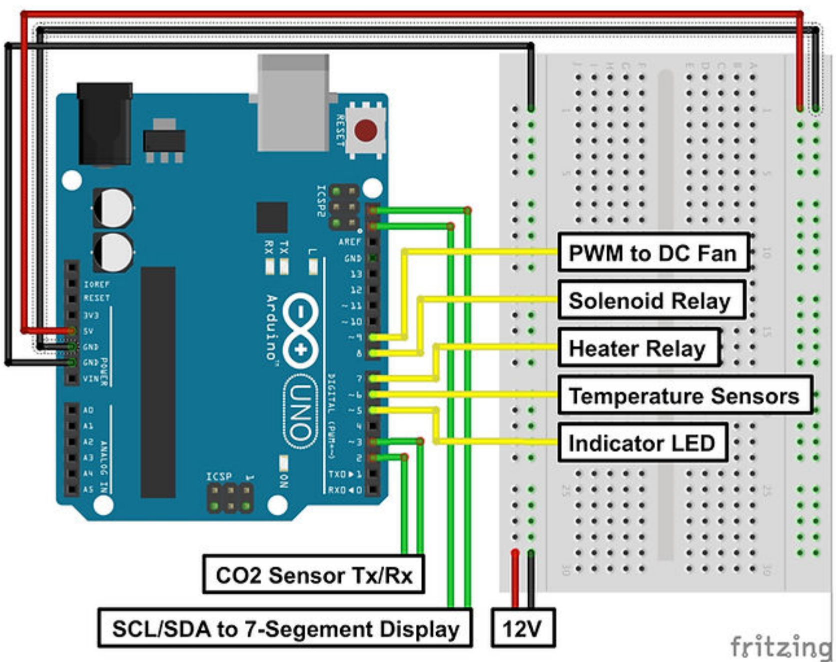
Arduino Test Code for Single Relay

Perfboard Setup

After testing and prototyping all the circuits together on a breadboard I locked it down on some perfboard. Check the full arduino code and the picture below to see what pins I used for reading/controlling everything. Depending on how much power your heaters consume, perfboard is probably a good idea. Breadboards arent rated to handle a lot of current and can be damaged. I also soldered some 18AWG wire onto the 12V rails of my perfboard (power rails closest to the Arduino in the pic below) and over the relay supplying my heaters. I had lots of headers lying around so I threw some on the board to make connecting/disconnecting all the sensors/hardware and 12V power supply a little easier.



Perfboard Setup



General Arduino control/sensor pin layout without showing the details of the individual circuits.

[More Pictures]

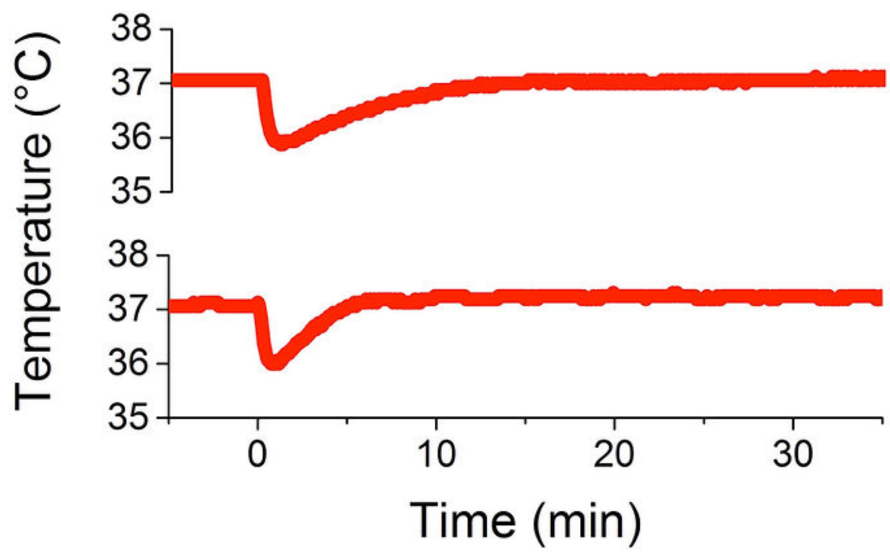
Control and Performance

The Arduino control code is currently setup to measure the average temperature across the two DS18B20 sensors and turn the heaters on if the average falls below the setpoint. The heating is controlled in two ways. If the temperature is lower than 98% of the setpoint (for example, 36°C with a setpoint of 37°C) the heaters come on continuously. If the temperature is greater than 98% of the setpoint (for example, 36.5°C with a setpoint of 37°C) the heaters turn on for only 1sec and another reading is made. This cycle continues until the setpoint is reached. This approach allows the system to step up to the setpoint and minimize overshooting.

CO2 control is managed in a similar fashion. When the CO2 content of the incubator drops below 80% of setpoint (for example 3.5% with a setpoint of 5%), the solenoid opens allowing CO2 to rapidly fill the incubator. If the CO2 level is above 80% of the setpoint (for example 4.5% with a setpoint of 5%), the solenoid only opens for 0.2 sec, closes and another reading is taken. This cycle continues until the setpoint is reached. This approach allows the CO2 content to step up to the setpoint and minimizes over-shooting.

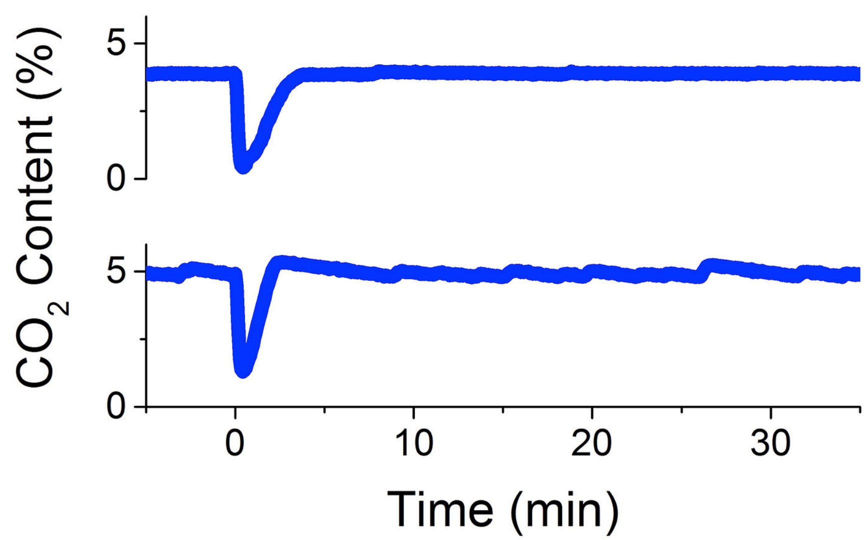
All the parameters (setpoint, thresholds, relay on times) modulating the control of the system are defined by the user. The default values in the Arduino control code work well for the incubator being described here. Values for the Temperature and CO2 setpoints (36.9 and 4.8, respectively), thresholds (0.98 and 0.8, respectively) and on times (1000 and 200, respectively) were chosen to achieve a stable reading of 37°C and 5% CO2. These parameters are defined at the beginning of the control code. It is extremely important that the user experiments and determines the best values to use. These values will depend very much on the size/shape of the incubator, pressure/flow rate of the CO2, type of heating and heat sinks, etc.

Using this setup, an experiment was then performed in which the temperature and CO2 content of the incubator was measured over time. A second Arduino connected to a data logging shield, a DS18B20 temperature sensor and another CO2 sensor was placed on the middle of the DIY incubator after it had been running for at least a day. The door was closed and the incubator was allowed to stabilize for three hours. At this point the Arduino was switched on and began logging temperature and CO2 every 5 seconds. After 10 minutes, the door was opened for 15 seconds and closed again. The recording then continued for several hours. As a comparison, the same experiment was performed on a fairly standard sized commercial incubator (170L volume). This is ~10 times larger than the DIY incubator. The same set of sensors were placed in the middle of the incubator. As well, the CO2 and temperature sensors were checked against known samples (water bath with analog thermometer and a 5% CO2 premixed gas from BOC).



Temperature recovery and stability of a standard commercial incubator (top curve) and the DIY incubator (bottom curve).

In the data above, it is clearly visible that opening the door results in a small but sudden drop in temperature by ~1°C in both incubators. Given that the commercial incubator possesses a number of temperature stabilization strategies (heated door, water jacket, etc) it was surprising the temperature drop was identical to the DIY incubator. On the other hand, due to its small volume, the DIY incubator displayed a significantly faster recovery time compared to the commercial incubator. However, both incubators recover and hold the temperature at 37°C after opening/closing the door with an acceptable temperature fluctuation of ~1°C.



CO2 recovery and stability of a standard commercial incubator (top curve) and the DIY incubator (bottom curve).

Opening the door of either incubator results in a rapid decrease in CO_2, approaching regular atmospheric levels. Surprisingly, the recovery time of both incubators was observed to be quite similar, however the DIY incubator did tend to display a small overshoot. What is clear from the data, is that the commercial incubator was able to maintain a more stable CO_2 content over time. The DIY incubator displays fluctuations of about ±0.2% (in other words, ~4% of the target value of 5%). In this case, the average stabilized CO_2 level in the middle of the incubator was ~4.9%.

However, of great concern was the fact that the CO_2 content in the middle of a commercial incubator was routinely less than 5%. This is very much due to their large size and an apparently poor ability to properly mix the gas. In the data above, the CO_2 content was ~3.9% on average, which was highly consistent with ~5 other makes/models that were examined. In one case, a commercial incubator was observed to maintain a CO_2 level of ~6%. Taken together, this represents a deviation of ~20% (or more in some cases) away from the target of 5% CO_2 content.

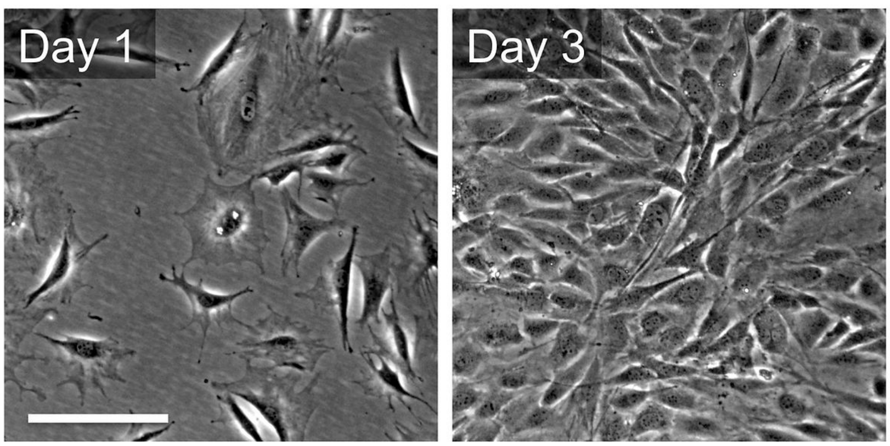
Conclusions: Although the DIY incubator has some room for improvement (described in the Possible Improvements section), it actually performs very well compared to commercial incubators that can easily be a factor of 15 or 20 higher in price. Most importantly however, mammalian cells (mouse cell lines and primary human cells) did not appear to be affected by the small CO_2 fluctuations that were observed in the DIY system. These cells exhibited normal proliferation dynamics, morphologies and the ability to undergo complex biological processes. These findings are described in the next section.

4. PROOF IS IN THE PUDDING

After building everything and testing the code its time to see if this incubator actually supports life. I suggest letting the incubator stabilize for a few hours before putting cells in. Its also important to put a small reservoir of water in the incubator to keep the environment humid in order to minimize evaporation from the culture dishes.

Mouse Fibroblast Proliferation

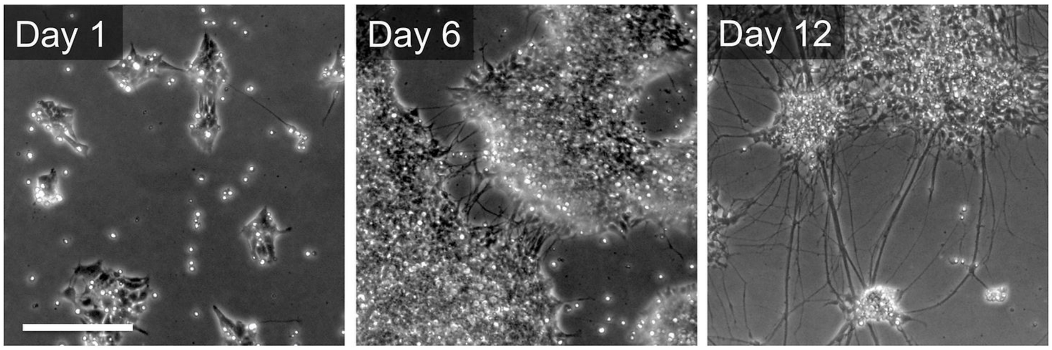
To test out the ability for this incubator to support the proliferation of mammalian cells I started with a very common mouse fibroblast cell line known as NIH3T3 (ATCC). The cells were originally derived in the 1960s and are used worldwide in cell biology research. What you see below are images of these cells proliferating over the course of a few days. Those who work with NIH3T3 cells will notice that the growth rate is indistinguishable from the growth rate of these cells in a commercial incubator. I also want to note the absence of any obvious signs of infection.



NIH3T3 cells proliferating over the course of 3 days in standard culture medium (DMEM, 10% FBS, 1% PenStrep). Scale bar = 200 µm and applies to all.

Human Neural Stem Cell Proliferation and Differentiation

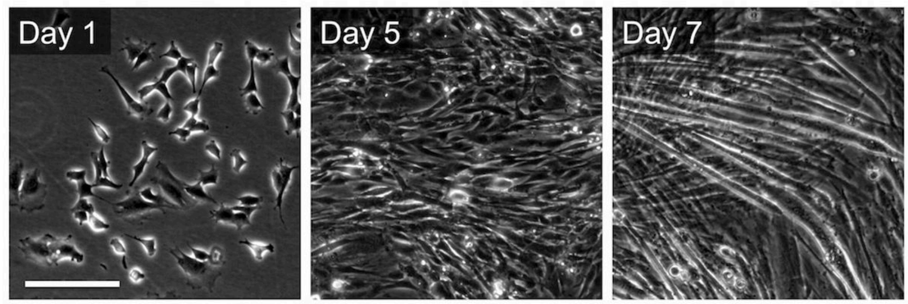
I decided to then try out a more sensitive culture of cells. In this case, I was able to get my hands on some human neural stem cells. These were derived from a skin biopsy of Guy Ben-Ary, artist and member of SymbtioticA. Following the Yamanaka protocol, his cells were modified to become induced pluripotent stem cells (iPSCs) and then driven towards neuronal progenitors. Some images of their growth over the course of several days in the DIY Incubator are below.



Human NSCs proliferating in maintenance medium (Invitrogen Neural Induction Media mixed with 50% NeuroBasel Media and 50% Advanced DMEM/F12) over 6 days. After adding retinoic acid (RA) to differentiate the cells, they begin to acquire a neuronal morphology by Day 12. This does not happen if RA is not added. Scale bar = 200 μm and applies to all.

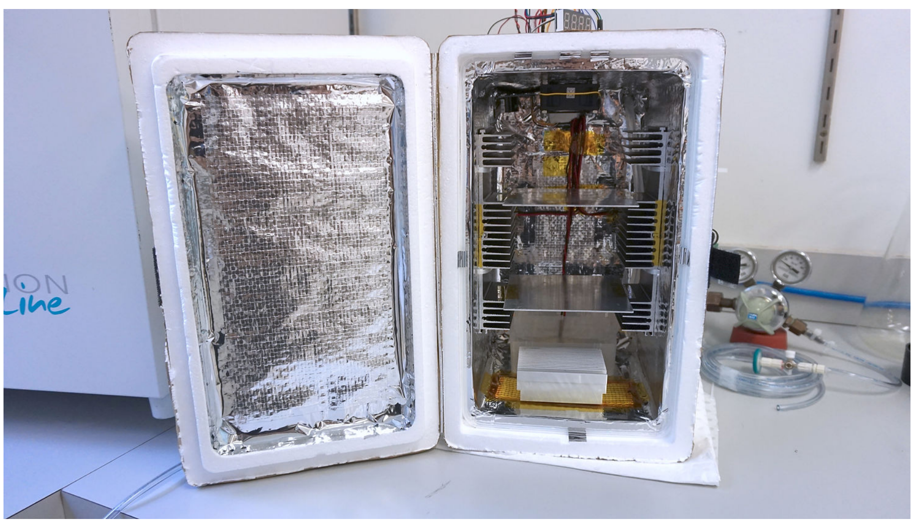
Mouse Myoblasts

Finally, I employed the common C2C12 mouse muscle myoblast cell line (ATCC) to see if I could perform a fairly standard differentiation protocol. These cells were originally derived in the 1970’s and are used routinely in many labs. They grow quickly and will also fuse and differentiate into muscle fibres. With this cell line it is possible to not only examine their proliferation but also the more complex process of myotube differentiation. What we see in the images below are C2C12 cells proliferating over a few days. Following standard protocols I allowed them to grow until fully confluent and then switched to a low serum differentiation medium. Keeping these particular cells in low serum promotes cell fusion and the differentiation of these cells into of myotubes. In vivo, this process is typically stimulated in response to stress (damage, exercise, etc) in which muscle satellite cells are activated, proliferate as myoblasts, which then fuse to form new muscle fibre. What we see is that over the course of a weeks in the DIY Incubator, I can reproduce the results from the same cells grown in standard commercial incubators. No infections occurred and the timing of cell proliferation, fusion and differentiation appeared no different than when cultured in a commercial incubator.



C2C12 Mouse Myoblasts growing in the DIY Incubator (scale bar = 200 μm and applies to all). Single cells are visible on Day 1, by Day 5 the culture has grown to confluence in standard culture medium (DMEM, 10% FBS, 1% PenStrep). On day 5 differentiation media (DMEM, 2% Horse Serum, 1% PenStrep) is added and myotubes are visible by Day 7.

5. POSSIBLE IMPROVEMENTS



This incubator is very much a result of trial and error, not enough time, and a self-imposed restriction to build it out of as much garbage/found goods as possible. However, it was put together for ~$350 and it actually works!

Even though the incubator sustains life, it is far from perfect and it is worth looking at possible improvements. Below is a non-exhaustive list of some potential changes:

1. Code/Circuits can be more efficiently implemented. Im hoping others in the community might have time to improve the implementation and will share it. The current code was put together by brute force and is quick and dirty. In particular, potentially employing threading or protothreading in the Arduino code might be fun to try out. Please let us know if you make use of these plans/code.

2. Find a cheaper CO2 sensor or coming up with a DIY version. This needs some thought, but starting with the components required for an NDIR sensor is a place to start. The sensor used in this project is by far the most expensive part and represents about ~60-70% of the total project cost. Cheap CO2 sensors would not only decrease the cost, but would also allow for multi-point measurements around the incubator volume which could be used to ensure a more uniform CO2 concentration.

3. Optimizing CO2 stability over time. Right now, when the solenoid opens CO2 (at ~100%) enters the incubator. This results in the CO2 content quickly ramping up to the setpoint. Although the time-averaged CO2 content from this setup is in the range of 5.0 ± 0.2%, the level can slowly fluctuate due to the stepping mechanism. There are a variety of ways to solve this, including: pre-mixing the CO2 with atmosphere prior to pumping into the incubator, finer control over the input pressure/flow rate, faster gas mixing, experimenting with incubator size, improving air tightness, etc.

4. Possible implementation of the Arduino PID feedback library to control CO2 and Temperature. PID might be worth examining for finer and more stable temperature/CO2 control. PID/PWM control was briefly tested but not in great detail.

5. There are lots of alternative items one can use for the incubator chamber. There may be some better choices out there. Would be kind of cool to have windows.

6. Integration with DIY microscopes to allow for monitoring of cell growth/dynamics.

7. Twitter enabled. Get status updates, send control messages, control any DIY microscopes etc. Send temp/CO2 data to thingspeak.

8. For point #6 & #7, Raspberry Pi control might be a better option with a little more flexibility and internet integration.

9. When the heaters are on the incubator draws ~65W (give or take). Camping solar panels that supply 80W at 12V might be really cool for cell culture in the field and/or remote/resource-scarce regions.

10. Better ports for passing wires into and out of the incubator.

11. Devise of a better mechanism for latching the door closed.

12. Alarms and/or LEDs to indicate low levels or the activation of heaters/solenoids.

13. Sparkles.

That’s it for now. Please let us know if you build your own!