USC Kit Directions

Introduction

These directions will hopefully be all you need to build your own controller from the kit. As always, feel free to contact Codapop with any questions.

Materials

Some materials are necessary for all modules and others are specific to some modules. Specific modules will be noted.

- Soldering iron and all relevant materials (solder, flux, etc.)
- Wire (standard hobby gauge, ~22-24 AWG)
- Wire strippers/cutters
- Pliers
- Screwdriver, with different size tips
- Vise/vice or some reliable way to compress objects
- Tweezers (For SMT chips on all joystick modules)
- 8mm wrench (for all switches and 3mm LEDs)
- 9mm wrench (Analog Throttle and Rotation Throttle modules)
- 15mm wrench (Executive Actions, Stage, Abort modules)
- Ink (for lettering, black or your choice)
- A suitable way to clean the ink (I use acetone-based nail polish remover)
- Silicone caulk (clear or the color of your container)
- Multimeter
- A breadboard
- Arduino IDE software or similar

Pre-flight

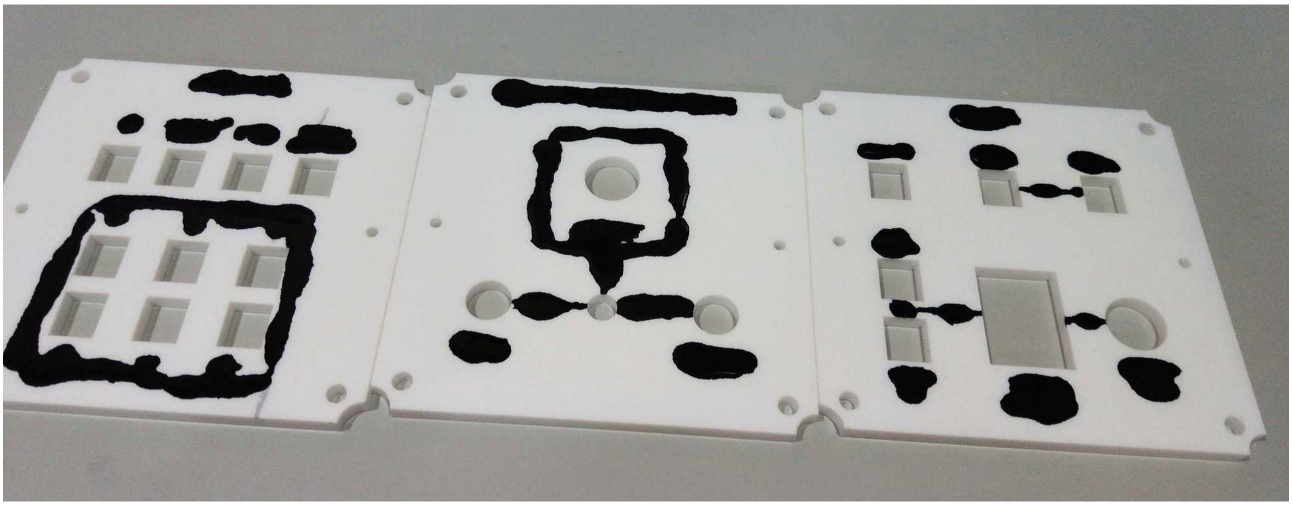
Before you begin, it is recommended that you do a test fit of all the components by fitting them into the PCBs and acrylic. Keep in mind things such as the direction certain button pins might be facing relative to their corresponding holes in the PCB. If something doesn't fit, then it might not go there. When in doubt, just ask.

In general, the two most devastating failure modes are failing to properly solder the Data Bus Connection and putting a magnet in the wrong way, so it's a good idea to focus on these specifically. The magnets are also powerful enough to ruin the IC chips and arduino, so be careful.

Inking

With the kit should be a small piece of acrylic with different size font etched into it. You can use this for testing your ink and cleaning solution. I recommend using ink for a permanent marker, but you can use dry-erase markers for a very easy cleanup.

I personally just drip ink into the etchings with the paper still on. I dry the ink with a tissue and then peel off all the paper. I then clean up any extra ink using an acetone-based nail polish remover. Pure acetone will ruin the mirror finish.



Container

The container houses the hub and wiring of the controller. It also needs to be fitted with magnets to hold the modules in place.

The placement of the hub should be pretty straightforward, but make sure the arduino's USB port and the 5V DC barrel jack are both pointing out of the holes in the back of the container. The arduino shouldn't be soldered directly to the board but instead should be inserted into two 15-pin sockets. The hub connects to the acrylic piece with one nut used as a spacer, so it is recommended to clip the bottoms of the wires for a clean fit.

The magnets should be pressed into place using a vise/vice. Ensure they are all going in the same way and that they are facing the correct orientation to accommodate the magnets in the modules. You should also try to keep the module side flush with the acrylic. Don't worry if the acrylic breaks around the magnet. As long as the magnet is in place, there is more than enough structural redundancy to maintain its integrity. If a larger piece snaps off, you may use super glue to reattach it, but be careful as the glue will ruin the finish. For the rest of the container, you may follow the container assembly guide.

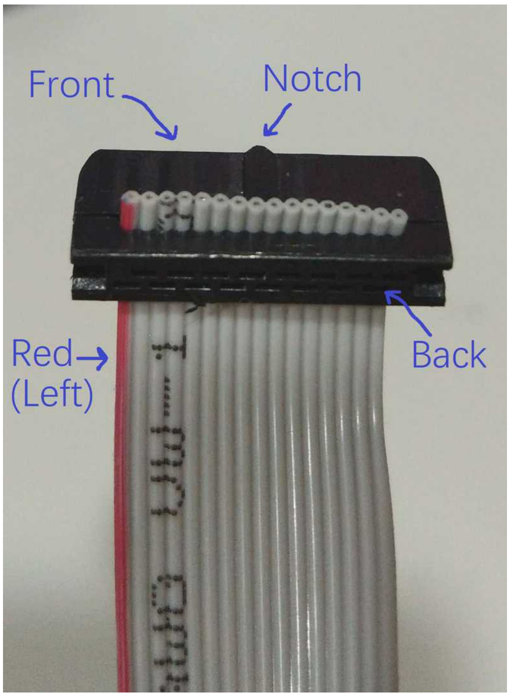
Wiring

Constructing the wiring is a very crucial step with very little room for error. Your wiring should have a red line along it that you can use for orientation. First you should make sure that your ribbon cable is 16 wires rather than 20. If it's 20, then you will need to remove 4 from the non-red side.

After that you should cut the wire into segments to fit your container. This can be done by counting how many modules removed a particular module is away from the hub. The module that sits on top of the hub is 1. Every module that is adjacent to that module is 2. And every module that is adjacent to a 2 is a 3, and so on. Determining that, you should cut the wires as so:

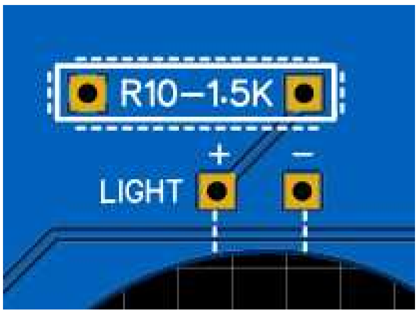
1: 12cm
2: 24cm
3: 36cm
4: 48cm

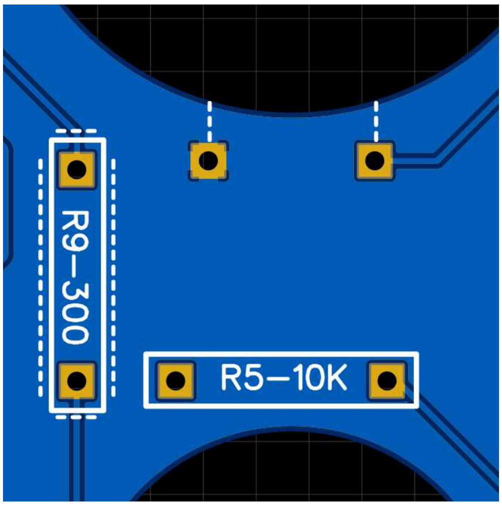
If you hold the wire in front of you with the red line towards the left, then the connector should be facing away from you and with the notch facing up. Insert the cable into the connector and then press the connector together in the vise, causing it to combine with the cable. Every connector should be added in the same way, with the connectors always facing the same way in relation to the red line. This ensures that the wires can be used in either orientation and interchangeably. If any wires are crossed here, it can be disastrous for your ICs.



General PCB Standards

Most of the components are clearly marked on the silkscreen of the PCBs. Typically, resistors are considered 'special' if they aren't 10k Ohms. Resistors that are 10k Ohms are marked with a plain white rectangle (with 10k written in the middle). Special resistors, typically used for LEDs, are marked with a white rectangle with a dotted line outlining it. Parts that include an annode and cathode (LEDs, capacitors) are marked with a + for the annode and - for the cathode. The annode is sometimes marked on the component or sometimes indicated by a longer pin. Unmarked holes are usually for jumper wires, with their destination component indicated by dotted lines. You may need to refer to an online resistor color calculator to determine which resistors you have. All ICs should use sockets rather than being soldered to the board.





Building the Modules

This is the general workflow for most of the modules. This does not include the analog joystick modules, Throttle, Telemetry, or EVA.

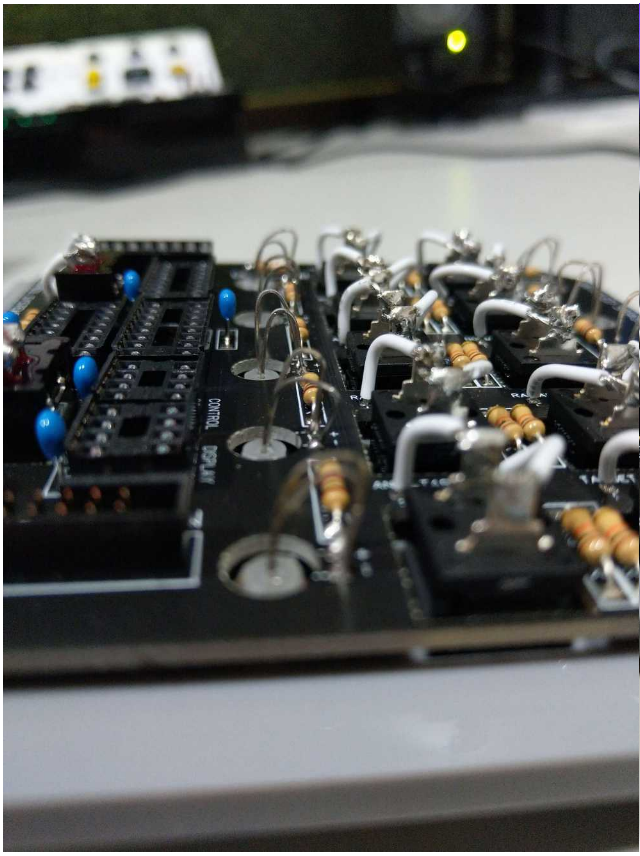
After your fit check, it is recommended that you solder all of the PCB components first, all resistors, capacitors, chip sockets, Data Bus Conn, and diodes (but NOT the potentiometer for the Throttle module). They should be soldered to the side that has the writing. You can also attach all of the buttons, LEDs, and switches to the acrylic plates, leaving the joysticks for later. The switches with two pins should always have the two pins facing down (or away from the name of the module on the acrylic).

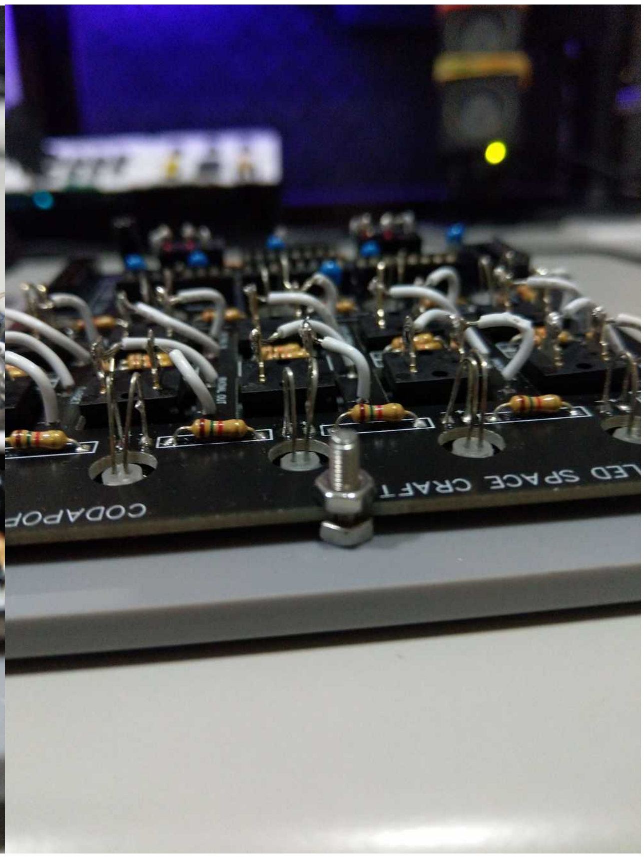
Please note: the square buttons have a tendency to prefer one side over the other. Now is a good time to look at them and pop them out and rotate them to make sure you are happy with everything.

At this point, you can cut/strip the jumper wires and solder them to the PCBs. They need to be soldered to the same side as all of the components. The result should be a PCB with many wires sticking out from the face with the writing. You will probably want to clip the ends on the bottom side and clean the excess flux with 99% isopropanol.

This is the point of no return for certain parts of the boards. You must ensure that the Data Bus Conn and ideally the chip sockets and electrolytic capacitors are all soldered correctly and securely, as they can be very difficult to fix later.

The PCBs attach to the acrylic using the screws, with one nut used as a spacer between the acrylic and the PCB. With everything secure, you can solder all of the jumper wires to their corresponding components.



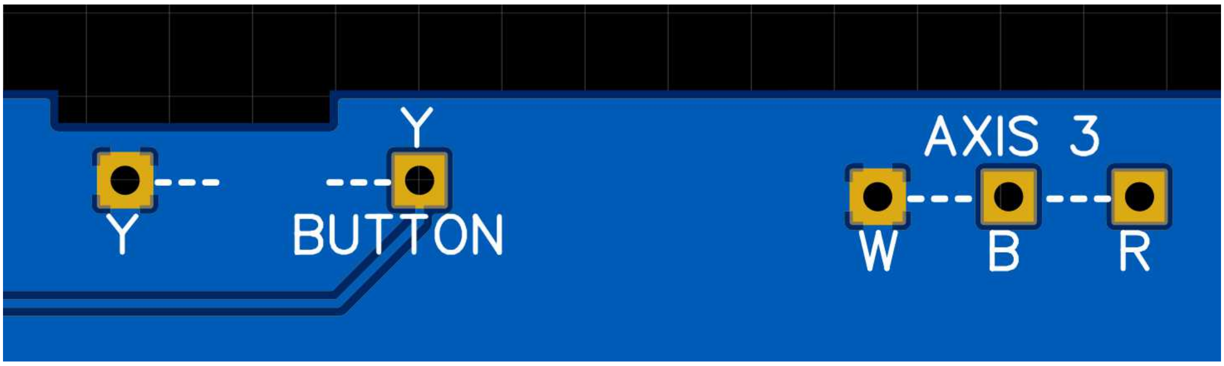


Of note: the 3mm LEDs with sockets (found on Control Systems, Action Groups, and Executive Actions) must be installed before securing the PCB to the acrylic, but the pins must remain straight until the PCB is secured. The pins can then be bent and inserted into the holes on the PCB that correspond with the annode and cathode and then soldered on the top side (opposite of the side that is typically soldered).

On the topic of these LEDs, there may be a few discrepancies in the resistor values, particularly with the blue LEDs which denote action groups. These might be labeled as 91k, but a recent change in the blue LED supplier has shifted these to requiring 62k resistors. Any and all 91k resistor markers on your PCBs should instead be 62k. The Executive Actions module that contains either 5 action groups or a mix of action groups with SAS/RCS controls might contain resistor markers that are labeled as both 91k and 10k. 91k (actually 62k) should be used if the LED it powers is a blue LED, and 10k should be used if the LED is orange.

Building the Analog Joystick Modules

The modules that contain joysticks are fairly similar to the standard modules, but the joysticks require special attention. Each joystick has 6 pins sticking out the sides (from the potentiometers) and should be fairly straightforward to wire up, as they just connect to the PCB with jumpers from Axis 1 and Axis 2. There are also 5 wires sticking out the side of the joystick: red, black, white, and two yellow. These go in the remaining empty holes next to the joystick labeled "Button" and "Axis 3". These holes are also labeled with letters that match the name of the color of the correct wire. R for red, B for black, and so on. I typically do this step after soldering the other wires to the PCB but before securing the PCB to the acrylic.



The rubber part of the joystick as well as the plastic ring cover must go on top of the acrylic, not underneath. These use screws that are easy to strip, so be careful. The two potentiometers on the sides of the joysticks can also be finetuned with a screwdriver and by checking that they are centered with a multimeter, but this shouldn't normally be necessary.

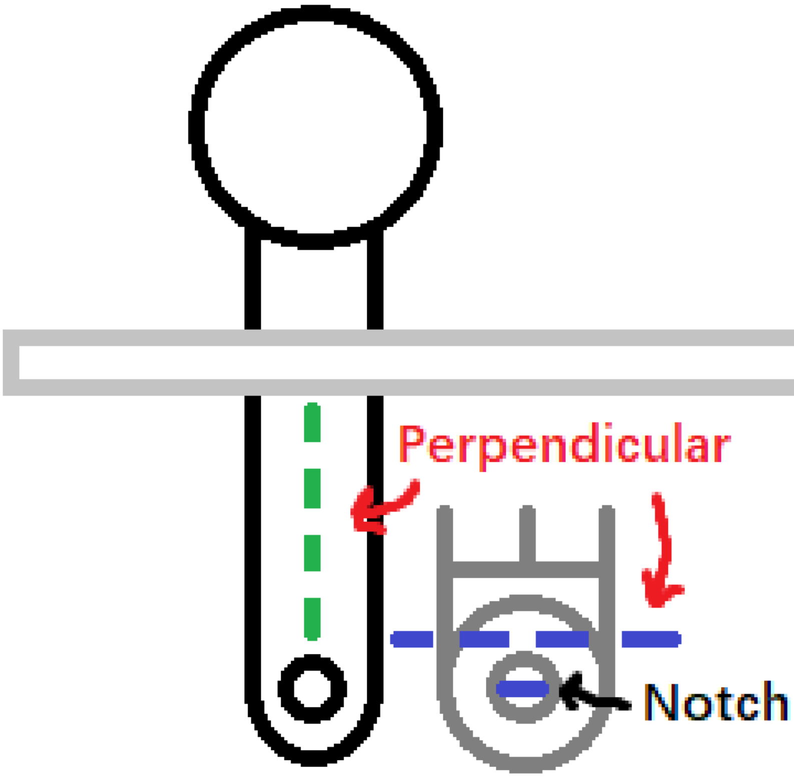
Building the Throttle Module

The Throttle module is perhaps the trickiest one and must be done with great care. Notably, this module requires a diode, which needs to be oriented with the dark line side of the diode on the same side as the white line printed on the PCB. Otherwise, you can solder all of the PCB components and wires like the standard modules, but skipping the potentiometer in particular.

The lever simply sticks onto the potentiometer, but this must be done after all other components are soldered and ready to be joined with the acrylic. Then you must stick the lever through both the acrylc and the PCB before attaching it to the potentiometer.

The lever must attach to the potentiometer with the lever in a particular orientation relative to the degree of rotation of the potentiometer dial. With the length of the lever providing an imaginary line, the cutout in the dial of the potentiometer must be perpendicular to this line, at least within a few degrees. The result should be, if the lever is oriented vertically, that the line in the potentiometer dial is horizontal. Be careful not to have it oriented upside down; the correct way should be obvious if you rotate the dial, with the bulk of the range of motion of the lever being on the side of the potentiometer with the pins. If you have already stuck the lever to the dial and aren't happy with the outcome, you can use pliers to wedge them apart and try again.

On production versions of this module, I like to add a piece of webbed material across the gap in the acrylic to prevent dust and debris from entering the hole as well as to break up the hole visually. It's not necessary, but it might be appealing to you; however, such material is not included with the kit.



The Throttle module needs to be calibrated so that the game throttle reads full when the lever is all the way forward, and off when the lever is all the way back. This requires the arduino to be placed into the hub socket with the "Calibrate_Throttle" code loaded onto it.

Open the connection in the serial monitor. With the throttle switch in the 'enable' position, push the throttle all the way forward and note the value, then push it all the way back and note the value. Do the same with the switch in the 'precision' position.

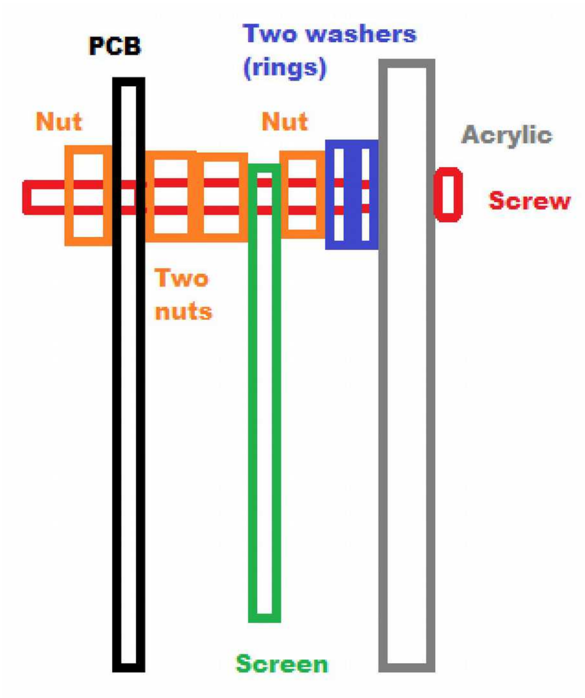
These four values will need to be added to the main hub code in the definitions.h file. Open the file and locate the four variables named throttle_min, throttle_max, throttle_pmin, and throttle_pmax. Change the values corresponding to the values that you recorded. Add 5 to both of the min variables and subtract 5 from each of the max variables to ensure that the throttle can both successfully max out and reach zero.

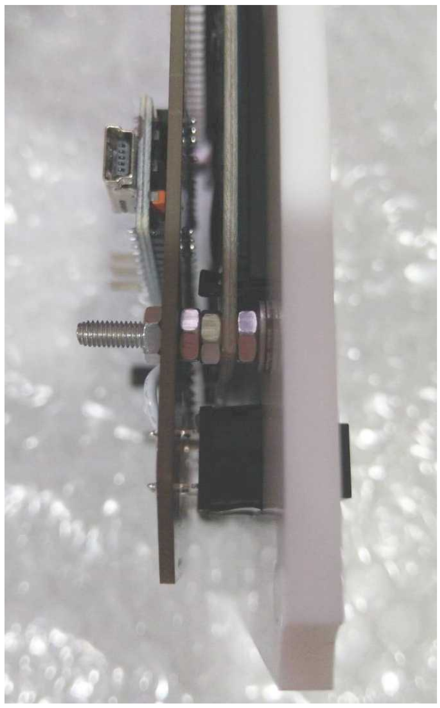
Building the EVA Module

The EVA module is, for the most part, the same as the standard modules. The only difference is the 10 LED array for displaying EVA propellant. This array is organized with all of the annodes of the LEDs on one side (the side with the serial number sticker) and the cathodes on the other. The array should be soldered to the PCB after the PCB and acrylic have already been joined. Soldering it first might prevent it from fitting into the hole in the acrylic later. The resulting array should stick up out of the acrylic slightly. Notably, the resistors for this array might be shown as "??" rather than a value. That's because the LEDs from different suppliers display different brightness levels. If your kit contains 30k resistors, then those should be used here. If it contains several 1.2k resistors (more than a small number, as they're also used for a few other LEDs), then they should be used here.

Building the Telemetry Module

The Telemetry module includes a large LCD screen and requires an arduino nano to drive it. It is the only part of the controller that requires longer screws (M3 screws 25mm in length). After soldering the 20 pins to the back side of the screen, the screen can be secured to the acrylic, separated from the acrylic by two M3 washers and one M3 nut. Then the PCB must be separated from the screen by two more nuts. The final order should be: acrylic, washer, washer, nut, screen, nut, nut, PCB, nut. The arduino will need to be soldered directly to the board as it doesn't have enough clearance to use a socket. It is a good idea to test the arduino first before committing to soldering it. The USB port on the arduino should face away from center, towards the outside edge of the PCB.





Adding the IC Chips

The ICs come in four different varieties. 74HC165, 74HC595, ATtinyX5, and ATtinyX4. The ATtiny chips might be listed on the PCB as, for instance, 'ATtiny25', but it is always fine to use a higher-number chip, in this case an ATtiny45. It is recommended to use sockets for all your ICs to greatly reduce the time cost of fixing a module in event of a failure, with the exception of the surface-mounted ATtinyX4.

The 74HC165 and 74HC595 chips require no preparation. Just bend the pins in slightly so they're nearly parallel and pop them into the sockets. Note the orientation of the chips: the half circle in the chip should match the half circle on the PCB. Mixing up the orientation could fry your chips. They also might make some unsatisfying clicks and pops while being pressed in, but in most cases this is normal.

The ATtinyX5 and ATtinyX4 chips are microcontrollers and need to be programmed. You can do this on a breadboard with directions for a simple ISP programmer circuit at the following link: https://www.circuits-diy.com/how-to-program-attiny85-with-arduino-uno/

The microcontroller (MC) chips are divided, if perhaps inefficiently, based on their function on each module. One ATtinyX5 controls the buttons (CTRL) while one drives the LED displays (DSPL). Analog joysticks are controlled with ATtinyX4. Open the included software in your arduino IDE and load it onto the ATtinyX5 chips, and then pop the chips into the sockets on the modules, being careful to note the orientation of the chips.

The ATtinyX4 chips on the Analog modules, being surface-mounted, typically need to be soldered onto the PCB before you can program them (unless you have an SMT chip socket of the right size). While soldering the chips to the PCBs, be sure to note the orientation; the small dot on the chip should match the small dot on the PCB. The programming holes on the PCB allow you to use the same circuit as you used for the ATtinyX5s, but you must plug the programming arduino into the Data Bus Conn to power the chip with a shared ground, or you can use alligator clips or similar to connect the 5V and GND. The connections on the arduino should go in the following order: Pin 10, 13, 12, 11, with pin 10 being the hole closest to the X.

Wrapping Up

After finishing a module, you may test it by loading the Hub code onto the arduino and connecting it to the module via the Data Bus Conn. To get it to connect in-game, you will need to make sure the 5V PSU is plugged in and that you have followed the Installation Guide to find the right COM port and update the .cgf file. It is a good idea to plug in the USB before plugging in the PSU and to unplug the PSU before unplugging the USB.