thePIXIE

Complete User Guide

thePIXIE

Complete User Guide

By: Kyle Park
(kyle.park@dal.ca)

Complete User Guide Version 1.0

Table of Contents

Preface

This document is a compilation of instructions and design notes to help you with the component sourcing, assembly, and operation of your very own PIXIE fluorometer. It contains details to explain how the device is configured – users with basic technical skills are shown how to make the minimum-necessary configurations for their device, whereas users with advanced skills are shown how the device works at a more fundamental level to drive their own customizations.

Each section of this document is written to be as stand-alone as possible, though references to earlier sections are ultimately unavoidable. Abbreviations are freshly defined each time they are first used in a section.

The goal of the PIXIE project is to dramatically improve the accessibility of submersible *in-situ* fluorometers without sacrificing device performance over industrial alternatives. By making the device fully open source and providing documentation, the PIXIE also provides access to educational opportunities in how such a device is designed and constructed.

1 Instructions

1.1 Components

In this section, the components of the default-configuration PIXIE are described. Instructions for choosing the optical components for your application are provided. Instructions for printing the two 3D-printed parts are provided. The three custom-machined parts are described in detail. Next, some alternative hardware options are presented, and their associated pros and cons are discussed. Lastly, the complete Bill of Materials (BOM) for the PIXIE is provided.

Once the component parts have been obtained, assembly instructions can be found in the following section, Section 1.2.

1.1.1 Optics Selection

For simplicity all the optically relevant components in the PIXIE will be described as "Optics" including the LEDs and photodiodes. The optics used in the PIXIE are the external window, the condensing lenses, the focusing lenses, the excitation filters, the emission filters, the excitation LEDs, and the detecting photodiodes. The PIXIE requires one external window for the device, and one set of lenses, filters, and an LED for each channel you intend to use.

The window and lenses chosen for the PIXIE are made of glass (BOROFLOAT® and N-BK7, respectively) so that the same parts can be used for the full range of UV to Near-Infrared fluorometry. Plastics, like clear acrylic, are a very inexpensive alternative but cannot be used for UV fluorometry because of their strong UV absorbance. The photodiode selected for the PIXIE is appropriately chosen for its sensitivity to the fluorescence of compounds ranging from Crude Oil to Chlorophyll.

For each channel you intend to install in the PIXIE, the choices are therefore reduced to selecting the correct excitation filter, emission filter, and excitation LED. By default, the PIXIE assumes its four channels are equipped for Phycocyanin (PC), Rhodamine Water Tracer (RWT), Chlorophyll-a (Chl-a), and Crude Oil fluorometry, respectively.

Each fluorescent compound, or fluorophore, has an associated excitation spectrum and emission spectrum. The excitation spectrum describes which wavelengths

(colors) of light are most strongly absorbed by the fluorophore. The selected excitation LED should have an output spectrum that is well-matched to a peak in the excitation spectrum to make the resulting fluorescence both strong and energy efficient. The emission spectrum describes the wavelengths that the fluorophore emits in response to being excited. Since this is the light we are interested in collecting (at the exclusion of all other wavelengths), the emission filter is chosen to match the peak of the emission spectrum and to strongly reject all other wavelengths. The excitation filter is selected to further reduce the possibility of excitation LED light being collected and misinterpreted as fluorescent emission; a process called optical crosstalk.

Figure 1.1-1 is provided to illustrate how the spectra of the fluorophores, filters, and excitation LEDs of the default-configuration PIXIE overlap. The excitation and emission spectra are adapted from literature: Quinine Sulphate (Kristoffersen et al, 2018), the fluorescent stand-in for Crude Oil, Chl-a and PC (Poniedziałek et al, 2017), and RWT (Smart and Laidlaw, 1977). These curves are reproduced and scaled for illustration purposes only and should not be consulted for quantitative work. For further reference, the long-dashed curve in each plot represents the background spectrum of the sun in air (worst-case scenario) and the short-dotted curve in the bottom plot represents the photodiode's detection spectrum.

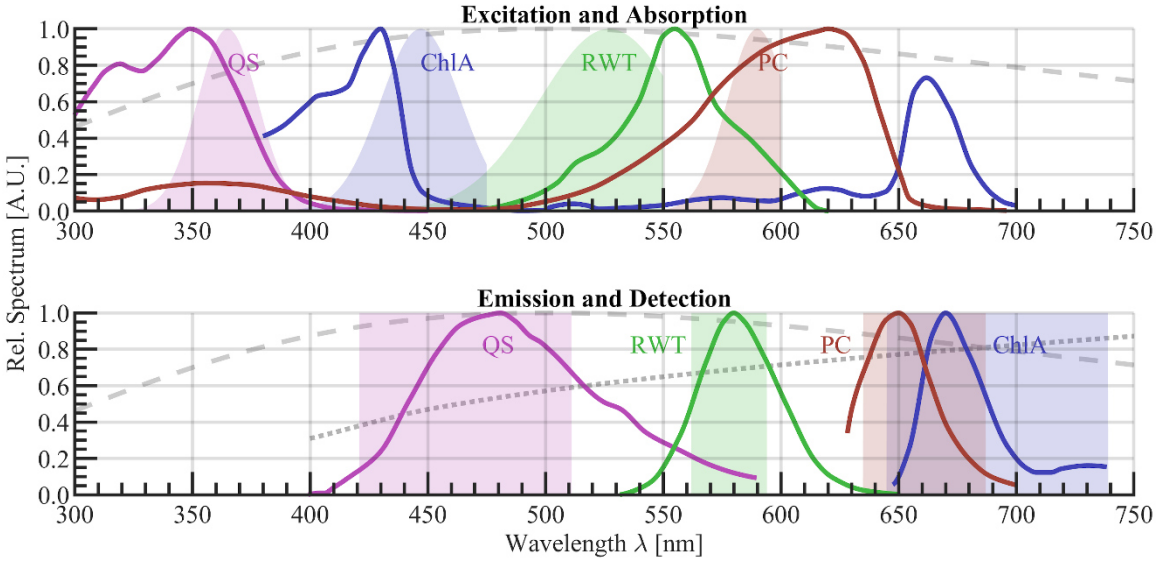


Figure 1.1-1 Spectra of default-configuration PIXIE Optical Components.

In the top plot, the shaded sections represent the spectra of the indicated excitation LEDs. The bell-shaped spectra of the LEDs are truncated by the excitation filters, as can be most clearly seen in the green RWT shaded section and red PC shaded section at 550 nm and 600 nm, respectively. The peak wavelengths of the LED spectra are placed close to the peaks of their corresponding excitation spectra, but necessarily on-top of each other. The exact placement is a balance of three factors: the availability of LEDs, the distance from other excitation peaks to prevent cross-sensitivity, and the distance from the corresponding emission peak to prevent optical crosstalk.

In the bottom plot, the shaded sections represent the passband of the emission filters. In contrast to the excitation LED spectra, the emission filters are centered over the peak of their corresponding emission spectra to maximize the amount of fluorescence collected. Another key selection feature is the lack of overlap between the excitation LED and the emission filter passband; as mentioned previously, this eliminates the possibility of optical crosstalk.

The following table, Table 1.1-1, summarizes the excitation LEDs and filters chosen for the default-configuration PIXIE. The Vendor URLs provided can be followed to obtain complete specification sheets and pricing information. These parts are also found in the Bill of Materials provided in Section 1.1.4 to follow. By following the logic outlined in this section, choose the optics required for your PIXIE.

Table 1.1-1 Summary of some default-configuration PIXIE optical components.

LEDs				
Channel	**Usage**	**Part Name/Description**	**Manufacturer**	**Vendor URL**
1	PC	MTE5900N-UY	Marktech Optoelectronics	https://tinyurl.com/yck8u5jb
2	RWT	MTE5270P-C		https://tinyurl.com/32jsaajz
3	Chl-a	MTE4600N		https://tinyurl.com/43pstk25
4	QS/Crude Oil	MT3650N3-UV		https://tinyurl.com/bdhcz6mh
Filters				
1	Excitation	600 nm short-pass filter, OD 4	Edmund Optics	https://tinyurl.com/bd4x7hyf
1	Emission	661 ± 26 nm band-pass filter, OD 6		https://tinyurl.com/ybudkw78
2	Excitation	550 nm short-pass filter, OD 4		https://tinyurl.com/2cv268wr
2	Emission	578 ± 16 nm band-pass filter, OD 6		https://tinyurl.com/acajbywd
3	Excitation	475 nm short-pass filter, OD 4		https://tinyurl.com/23ndn38a
3	Emission	692 ± 47 nm band-pass filter, OD 6		https://tinyurl.com/4aa3ymby
4	Excitation	400 nm short-pass filter, OD 4		https://tinyurl.com/mw6hvcnd
4	Emission	466 ± 45 nm band-pass filter, OD 6		https://tinyurl.com/2wyananm

1.1.2 3D Printed Components

A major design goal for the PIXIE was to incorporate the use of consumer-grade 3D printers to produce parts for the device wherever possible. The PIXIE takes full advantage of 3D printing to produce its Optical Stack; the part used to align and house the optics described in Section 1.1.1. The other 3D printed part, the Optical Cap, mates with the Optical Stack to seal in the optics and to shroud the surface-mount photodiodes on the PIXIE Printed Circuit Board (PCB) from being directly exposed to any source of light besides that which can pass through the emission filters. The complex and compact geometry of the Optical Stack, combined with its comparatively high mechanical error tolerance, make these parts a prime candidate for 3D printing as opposed to conventional machining. Photographs of these two parts are provided below, in Figure 1.1-2.

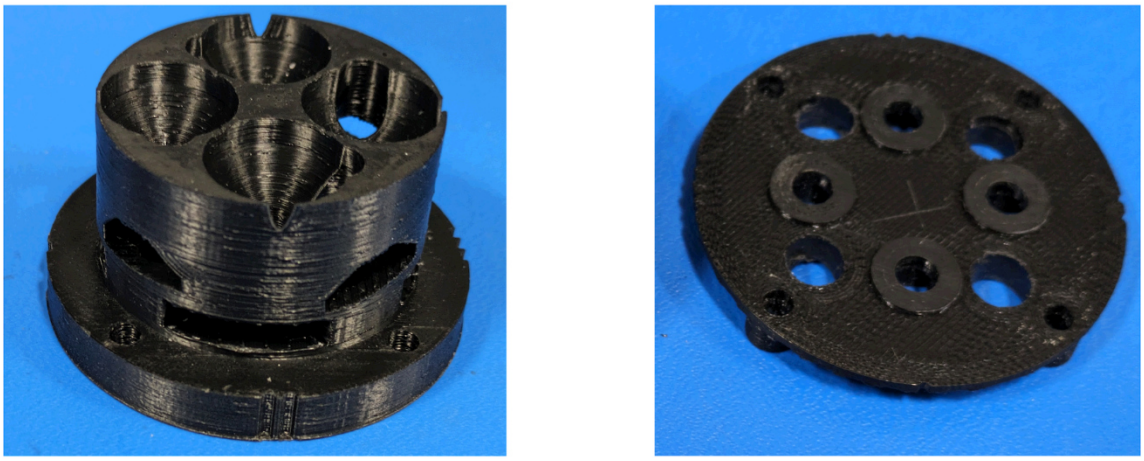


Figure 1.1-2 Photographs of 3D printed parts. (a) Optical Stack. (b) Optical Cap.

There are two primary varieties of consumer-grade 3D printers: Fused Deposition Modeling (FDM) 'filament' 3D printers, and Stereolithography 'resin' 3D printers. The two PIXIE components described above are printed with opaque-black Polylactic Acid (PLA) filament using FDM printers. Opaque-black Acrylonitrile Butadiene Styrene, or ABS, filament is also an option, but this material typically requires an enclosed, medium-to-high end 3D printer.

An unavoidably attractive thought that must be addressed is whether more of the PIXIE can be 3D printed. However attractive it may be, the common 3D printing filaments are almost universally hydroscopic plastics and cannot guarantee any degree of watertightness, even at the water's surface. Additionally, the structure of

FDM printing allows microscopic gaps to persist between layers, further compromising the watertightness of the printed part. It may be possible to print watertight components using a resin 3D printer, but this avenue was not explored for the PIXIE.

The PIXIE parts have been successfully printed using three 3D printers: the Flashforge Creator Pro (released 2016), the FLSUN Q5 (released 2019), and the Bambu Lab P1S (released 2023). The slicers used to obtain the printers' ".gcode" files were Flashforge FlashPrint 5, UltiMaker Cura 5.4, and Bambu Lab Bambu Studio 1.8, respectively. The print settings for Bambu Studio are provided here for reference; they can be adapted for your make and model of printer/slicer as needed.

The basic configurations in Bambu Studio are identifying your printer and the filament type you intend to use. In the example given in Figure 1.1-3, the selected printer is the Bambu Lab P1S, equipped with a standard 0.4 mm nozzle. The selected filament is #1, Bambu PLA Basic. The profile for this material is the stock configuration defined by Bambu Lab; it has not been modified in any way.

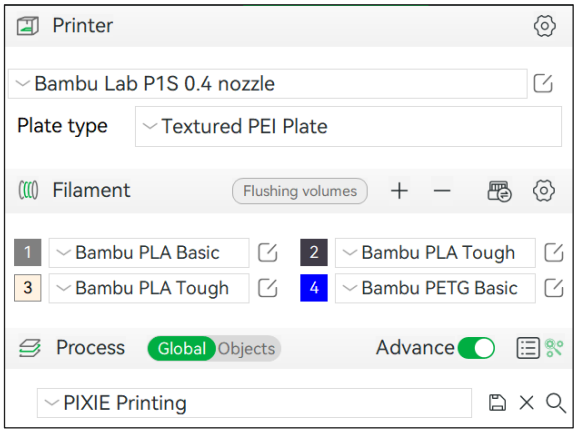


Figure 1.1-3 Bambu Lab basic settings.

Since both the Optical Stack and Optical Cap are cylindrical parts that are nearly axially symmetrical, the concentric tool path is preferred for printing these components. The profile "PIXIE Printing" is saved for repeat use, as the Optical Stack and Optical Cap are printed with the same settings. The Advanced settings, broken down by Bambu Studio into Quality, Strength, Speed, Support, and Others tabs, are provided in full in Appendix A.

When printing the Optical Stack, it is important to ensure the part is printed with the appropriate supports. The cavities used to hold the Emission Filters have vertical overhangs (flat ceilings) that most consumer grade 3D printers will have difficulty printing without support structures. The cavities used to hold the Excitation Filters have sloped overhangs that some 3D printers may have difficulty

printing without support structures. Figure 1.1-4 below provides an annotated example of the Optical Stack being printed with supports in Bambu Studio 1.8. The green material indicates support plastic that can be pulled from the part after printing, using a set of fine needle-nose pliers.

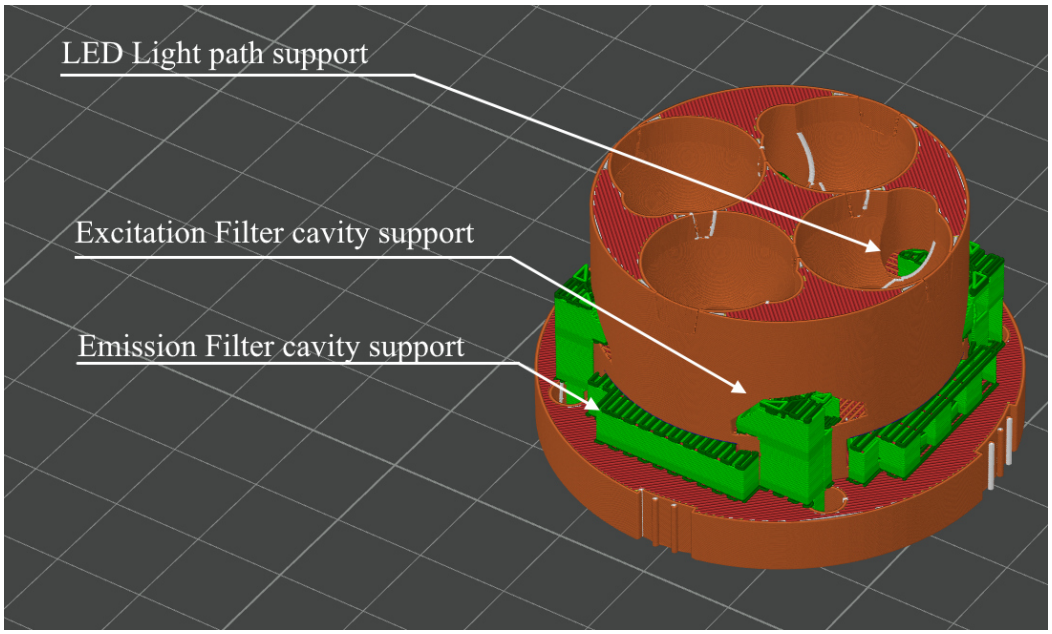


Figure 1.1-4 Optical Stack with indicated support structures.

The Optical Cap should be printed with the orientation indicated below in Figure 1.1-5. This is to ensure that the mating face of the Optical Cap is as smooth as possible to ensure a tight fit with the Optical Stack.

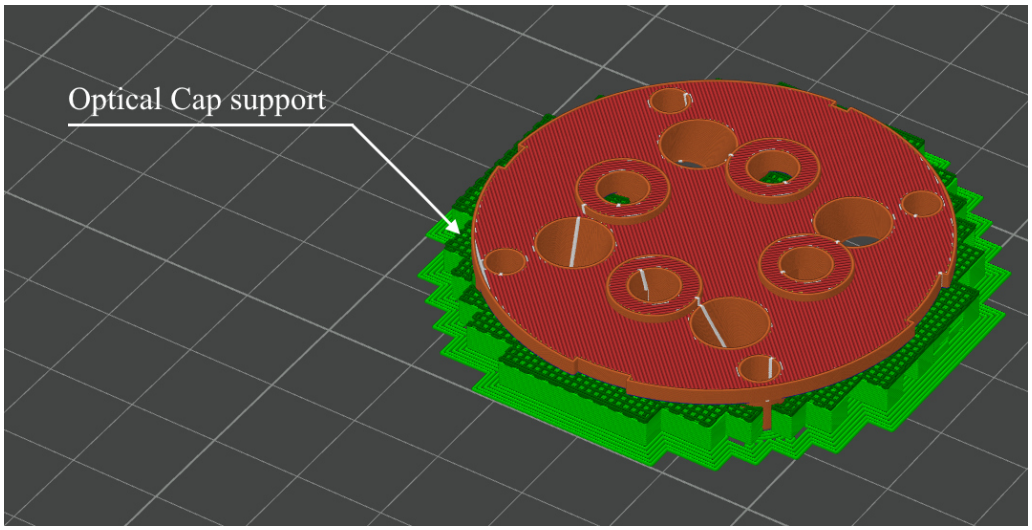


Figure 1.1-5 Optical Cap with indicated support structure.

After slicing, Bambu Studio proves a report on the amount of time and material used to print the part(s). The Stack and Cap should be printed one at a time to maximize the likelihood of a successful print. The report for each part is shown below in Figure 1.1-6; (a) provides the report for the Optical Stack and (b) provides the report for the Optical Cap. The stated total printing time is 4 hours, 34 minutes. The total material use is 37.68 g, costing just under $1.00 USD per set of parts.

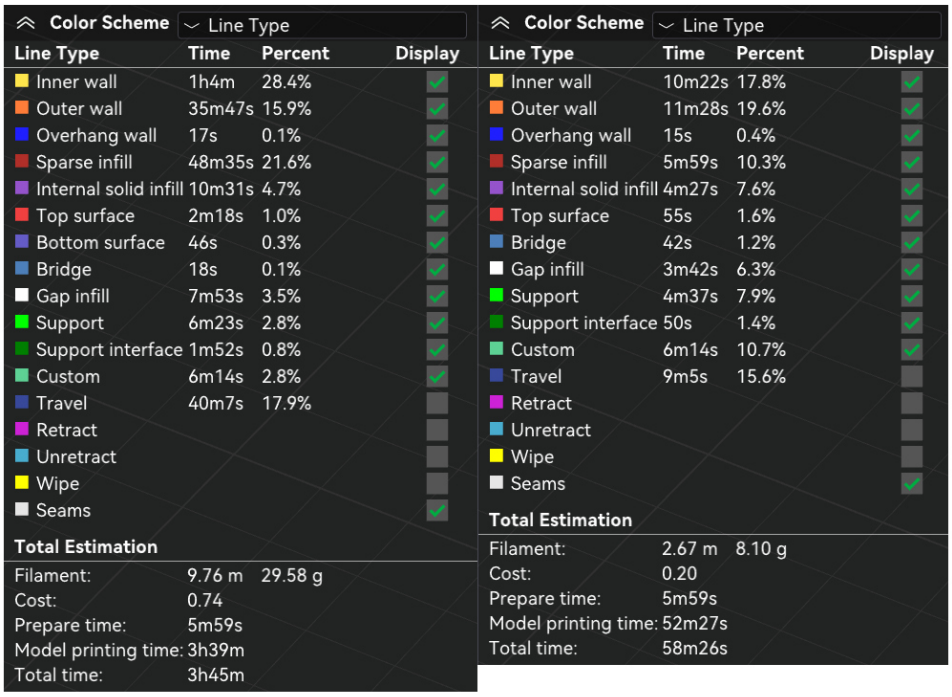


Figure 1.1-6 Print report for the (a) Optical Stack (b) Optical Cap.

Once the parts have been successfully printed, you may need to clean the cavities of residual material left over from supports. This can be done with a sharp knife or gouge. The cylindrical holes through each part can be cleaned using a slightly undersized drill bit; you are not aiming to remove significant amounts of material, just burrs or hairs of over-extruded plastic filament.

Sanding and abrasives should be avoided unless the underside of the Optical Stack is too rough to make a sealed fit with the Optical Cap. If the bottom surface is rough, a 220-grit abrasive can be used until the scratch pattern is uniform across the bottom of the Optical Stack. The sanding paper should be placed on a very flat surface, and the Stack should be drawn across it in a figure-8 pattern until the appropriate smoothness is obtained.

There are three fit tests that should be performed once the parts are cleaned. You can find these steps outlined below, along with recommendations if the fit is incorrect:

- The Optical Stack and Optical cap should be mated together. The fit should be slightly loose, as these parts will be forced together by machine screws once fully assembled. If the parts snap together and cannot be removed by hand, your 3D printer may be out of calibration.
- The Emission Filter cavity should be checked by partially inserting one of your selected emission filters in the cavities (see Figure 1.1-7). If the filter is difficult to insert in one cavity, that cavity may still have leftover support material and need additional cleaning using a sharp knife or gouge. If the filter is difficult to insert into all cavities, then the part has printed undersized, likely due to shrinkage. The overall print scale should be increased by 2%, and the print of *both* parts should be repeated. If the filter slips in too easily and rattles in place, then the part has printed oversized. The overall print scale should be reduced by 2%, and the print of *both* parts should be repeated.

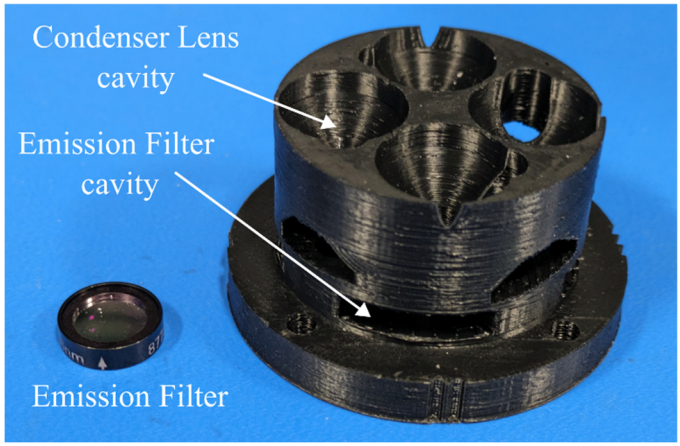


Figure 1.1-7 Optical stack with Emission Filter, Emission Filter cavity, and Condenser Lens cavity indicated.

- The top cavities should be checked by inserting a condenser lens. The planar end of the lens should fit flat with the top of the Optical Stack and should be difficult to rock from side to side. If the lens easily slips or rocks from side to side when pressed, the part was printed oversized. If the lens doesn't fit at all, the part was printed undersized.
- When balancing the fits of the Condenser Lens and the Emission Filter, the fit should favor the Condenser Lens: a firm fit for the Condenser Lens with a difficult-to-remove fit for the Emission Filter is preferable to a loose or sloppy fitting Condenser Lens with an easy-to-remove fit for the Emission Filter.

1.1.3 Machined Components

There are three machined components required for the default-configuration PIXIE. These components are manufactured by external vendors and shops. The three components are: the Custom Window, the Retaining Cap, and the PIXIE Printed Circuit Board (PCB).

The Custom Window is specified as a circular, plano-plano surfaced glass window manufactured in a commercial borosilicate (BOROFLOAT®) substrate. Borosilicate glass is robust, relatively inexpensive, chemically inert, and offers good optical transmission from 350 nm to well over 1000 nm, allowing the full range of UV to near-infrared fluorometry. The Custom Window is depicted in Figure 1.1-8.

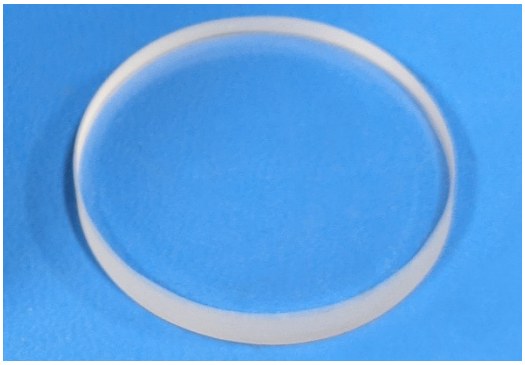


Figure 1.1-8 Photograph of PIXIE Custom Window

The nominal dimensions of the window are a diameter of 46.5 mm and a thickness of 6.5 mm. Any alterations to the dimensions or tolerances of the Custom Window would require matching changes to the dimensions and stacked tolerances of the Retaining Cap that holds the Custom Window in place.

The Custom Window depicted in Figure 1.1-8 was manufactured by Edmund Optics (New Jersey, USA). Their custom productions page can be accessed through the following URL:

https://www.edmundoptics.com/tools/borofloat/

The specifications and drawings of the Custom Window can be obtained from the PIXIE Repository:

https://github.com/KylePark0/PIXIE/tree/main/Hardware/CustomWindow

The Retaining Cap is a ring cut from Aluminum and Anodized to match the fit and finish of the BlueRobotics (Torrance, CA, USA) 2”-series O-ring Flange. The function of the Retaining Cap is to clamp the Custom Window onto the O-ring flange. The cap itself does not provide a watertight seal at depth; while it may provide some degree of ingress protection at surface pressure, the watertight seal is provided by

the interface between the Custom Window and the face O-ring housed by the flange.

The six M2 screw holes on the Retaining Cap match the pitch and pattern of the O-ring flange and are unthreaded. Corrosion-resistant stainless-steel machine screws are specified for fastening the Retaining Cap. It is unlikely that the Retaining Ring will be damaged by over-torquing these six fasteners, but it would likely damage the threads of the O-ring flange. The screws should be installed at the same time and fastened progressively and evenly. Shows the Retaining Cap in CAD isometric view and as part of a fully assembled PIXIE fluorometer.

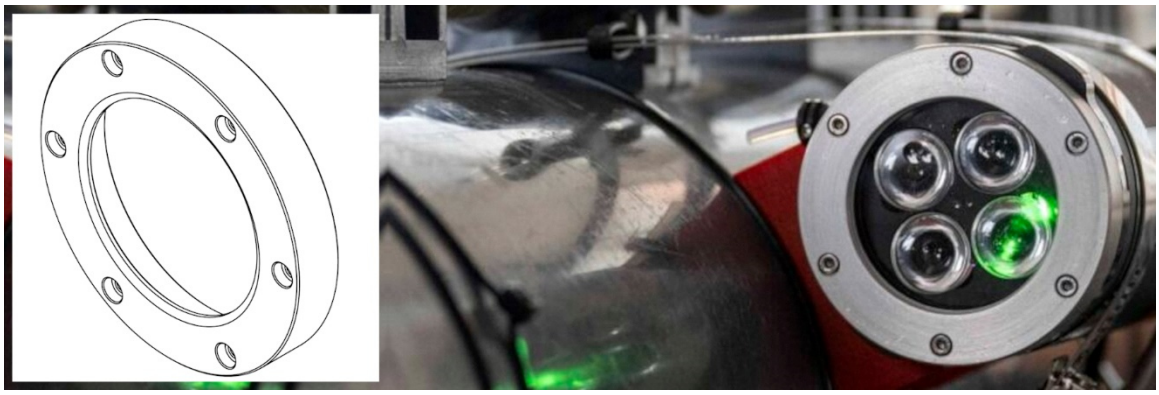


Figure 1.1-9 PIXIE Retaining Cap, in CAD and photograph as part of a fully assembled PIXIE.

Since Aluminum is a common material, obtaining the material, machining the part, and anodizing the part is left to the user to source using local or international vendors and shops. The drawings, specifications, and CAD models of the Retaining Cap can be obtained from the PIXIE Repository:

https://github.com/KylePark0/PIXIE/tree/main/Hardware/RetainingCap

The PIXIE PCB is a purpose-designed and built PCB for the PIXIE fluorometer that handles all device functions on one board. The PIXIE PCB itself is intended to be produced and populated by a PCB/Prototyping provider such as PCBWay (Hangzhou, Zhejiang, China). The PIXIE PCB is designed using the free and open-source software suite KiCAD. The schematics, gerber files, BOM, and KiCAD project files for the PIXIE PCB, along with a ready-made package to submit to PCBWay for fabrication, can be obtained from the PIXIE Repository:

https://github.com/KylePark0/PIXIE/tree/main/Electrical

The PIXIE PCB requires an additional step of hand-soldering the Excitation LEDs for each of the channels selected in Section 1.1.1. Care should be taken not to overheat the PCB when installing the LEDs, as this may damage nearby components. Good soldering practices are a necessity.

The leads of the Excitation LED should be soldered tilted, or bent afterwards, to match the pitch of the Excitation LED cavity in the Optical Stack. For this reason, it is very helpful to have already 3D printed the Optical Stack and Optical Cap when attempting to solder the Excitation LED(s).

Use the silk-screen on the PCB to ensure the Excitation LED that you are installing matches the reference designator of the channel you intend to use it for; it is easy to mistakenly install the LED to the wrong channel at this stage. The Schematics of the PIXIE PCB are included in Appendix B of this user guide for quick reference. Figure 1.1-10 shows an example of the solder pads and silk screens for diode D9, which corresponds to the Excitation LED for Channel 4. The square pad indicates the position of the cathode.

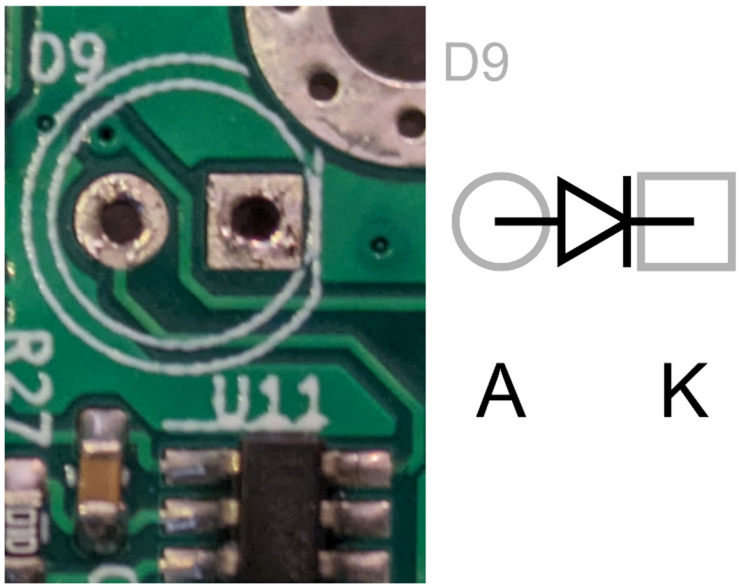


Figure 1.1-10 PIXIE PCB Solder pad and silk screen layout for Excitation LEDs.

If using the default Excitation LEDs outlined in Section 1.1.1, it is important to note that, while these parts use the same TO-18-2 Metal Can packaging, these parts differ between which indicated lead is the cathode and which is the anode. Always check your Excitation LED's datasheet before soldering it in place. Additionally, it is necessary to bend down the metal tab of these parts in order to seat them properly in the Optical Stack.

The Excitation LED cavity of the Optical Stack is angled 7° from vertical to aim light into the focus of the Condenser Lens and maximize the captured fluorescence. In Figure 1.1-11 a cutaway view of the PIXIE PCB with the Optical Cap, and with both

the Optical Cap and Optical Stack, is provided. This cutaway view illustrates the cavity that the LED must fit within.

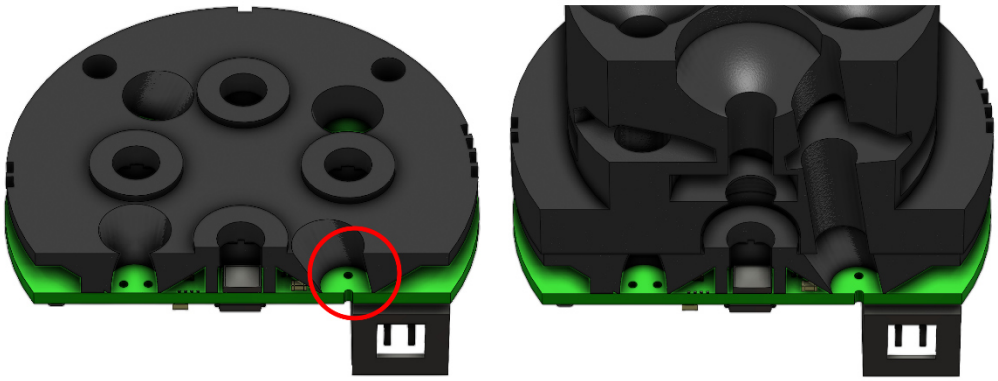


Figure 1.1-11 PIXIE PCB cutaway with Optical Cap (left) and Cap + Stack (right). LED Through-holes indicated with red circle.

Depending on the tolerance of the 3D print, it is possible the Optical Cap will not slide over the Excitation LED(s) without interference. Test this by first sliding the LED through the cavity indicated with the red circle; if the LED does not fit, the cap must be placed onto the PCB *before* soldering the LEDs in place. Alternatively, the cavity can be widened by boring out material using a drill bit, but this risks warping or damaging the Optical cap.

To allow the LED enough free play to properly locate itself when the Optical Stack is pressed in place, first orient the LED so that it is perpendicular to the face of the PCB. The distance from the bottom of the LED's metal can to the surface of the PCB should be 10 mm. Once properly oriented, solder only one of the leads in place for now. Figure 1.1-12 provides renderings of the PIXIE PCB both before and after installing the LED.

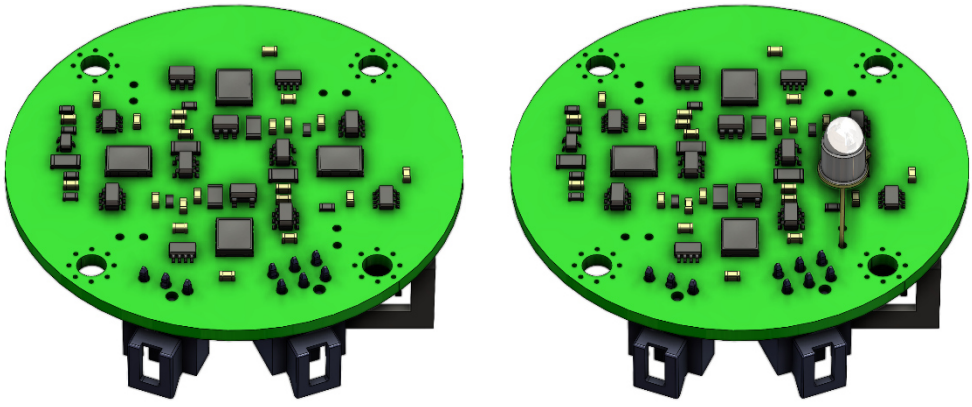


Figure 1.1-12 PIXIE PCB, both before (left) and after (right) installing Excitation LED.

Once the LED and Optical Cap are in place, press the Optical Stack into place (this is best done before the lenses and optical filters are installed, as the Optical Stack can still be easily removed at this point). As shown below in Figure 1.1-13, the LED will interfere with the Optical Stack and is thereby pushed into alignment with the cavity. The leads of the LED can also be bent by hand, if necessary.

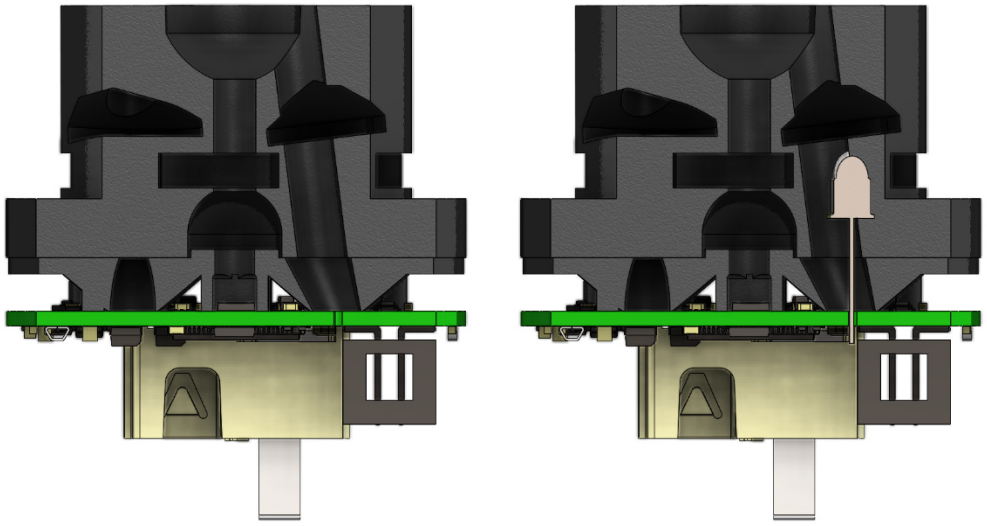


Figure 1.1-13 Section view of PIXIE PCB with Optical Cap + Stack, before (left) and after (right) Excitation LED installed.

Visually inspect the fit through the top and side of the Optical Stack so that the lens of the LED does not extend into the Excitation Filter cavity (described in the previous section, Section 1.1.2).

Once the alignment and fit are inspected, fixed the LED in place by soldering the remaining lead to the PCB. If multiple Excitation LEDs are being installed, *all* the LEDs should be checked for fit and alignment before *any* are fixed in place.

1.1.4 Alternate Hardware

The default-configuration PIXIE provides an enclosure that is robust and has good performance for depth, optical clarity and transmissibility of UV light, and chemical compatibility for calibration. If compromises can be tolerated on one or more of these metrics, then alternative hardware combinations exist that can reduce cost or add room for components. Some alternative combinations are compared here.

The sensing end of the default-configuration PIXIE uses a custom-cut glass window and a custom-machined anodized-aluminum retaining ring that mates with an off-the-shelf (OTS) O-ring flange. If the PIXIE will never be exposed to harsh solvents and will not perform UV fluorometry, the custom glass and aluminum ring can be replaced with an OTS acrylic window instead. If the custom glass window is too difficult or too expensive to obtain, whereas custom aluminum parts would be comparatively easy, the OTS O-ring flange can be replaced with a custom O-ring flange/retaining cap and an OTS glass window. In Figure 1.1-14 below, illustrative examples of these three configurations are presented: (a) shows the components for the acrylic window configuration, (b) shows the components for the default configuration, and (c) shows the components for the custom flange configuration.

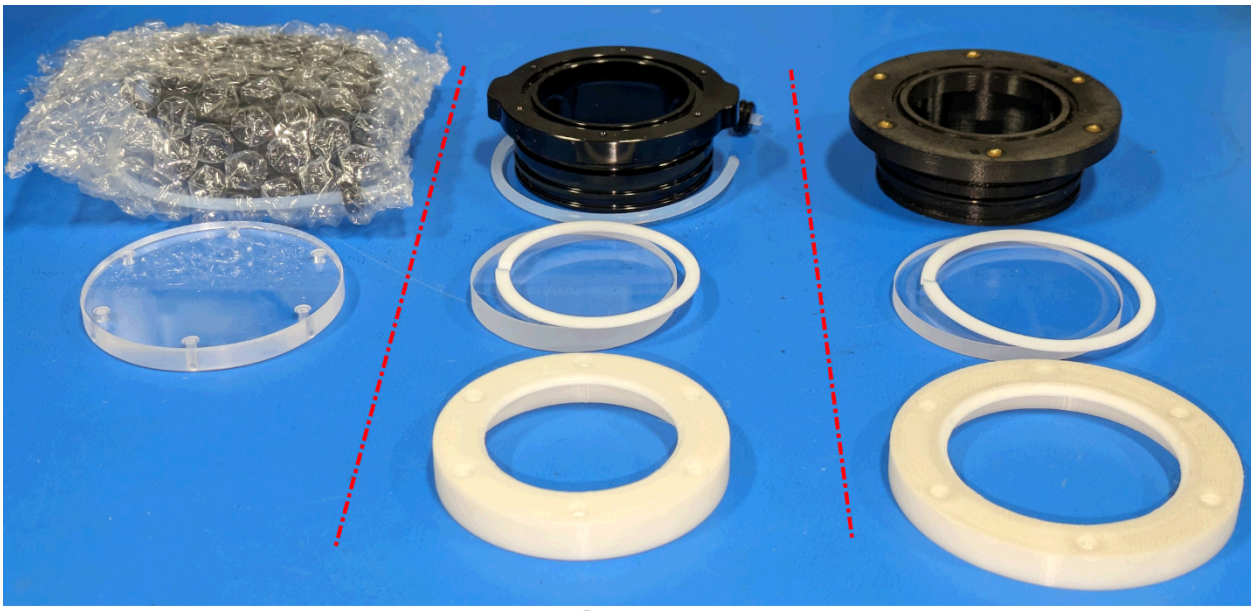


Figure 1.1-14 Alternate sensing end configurations. (a) Acrylic window configuration. (b) Default configuration. (c) Custom flange configuration.

Note that the custom flange and retaining rings depicted in Figure 1.1-14 are 3D printed for illustration purposes only. The flange must be machined from anodized aluminum, or a superior metal at higher cost (stainless steel, titanium), with smooth

O-ring channels to provide a complete seal. The retaining rings must also be machined; while not providing a seal in and of themselves, plastic FDM-printed rings may fracture at depth, causing the glass window to slip and breach the O-ring seal.

The default-configuration O-ring flange, a product of BlueRobotics, is an OTS part designed for a watertight seal with the BlueRobotics 2” inner diameter tube housing series. The default-configuration housing uses the 3.9” length option in anodized aluminum, rated for depths of 950 m. Additionally, BlueRobotics offers the same tube housing in greater lengths, and in clear acrylic tubes in the same optional lengths. The fully assembled sensing end configurations presented in Figure 1.1-14 are shown below in Figure 1.1-15, each with a separate example of alternative tube housings.

Figure 1.1-15 Alternate sensing end configurations, fully assembled, with alternate tube housings.

The acrylic housing and sensing end window are incompatible with the acetone used to calibrate for Chlorophyl-a, and harsher solvents used to calibrate for Crude oil and UV fluorometry. The aluminum housings have a consistent depth rating of 950 m, whereas the depth ratings of the acrylic housings decrease with length.

In Table 1.1-2 below, the specifications and cost of the BlueRobotics tube housing options are provided for comparison's sake.

Table 1.1-2 Comparison table for alternative housing options.

	Material	
Nominal Length	Acrylic	Aluminum
3.9"	$24.00, 225 m depth	$46.00, 950 m depth
5.9"	$28.00, 130 m depth	$52.00, 950 m depth
11.8"	$34.00, 100 m depth	$68.00, 950 m depth
https://bluerobotics.com/store/watertight-enclosures/locking-series/wte-locking-tube-r1-vp/		

For the alternate sensing ends, the acrylic window is an optical component. Compared to borosilicate glass, the acrylic window is less resistant to scratches, cannot be used for UV fluorometry due to high UV absorbance, is susceptible to fogging, and is highly susceptible to strong solvents. but is the least expensive option and requires the fewest components. The custom flange option may be less expensive than the default configuration if aluminum machining and anodizing services are easily accessible, whereas custom glass is available directly from the vendor. Table 1.1-3 below summarizes the three sensing end options described in this section. Note that only the default configuration has been formally calibrated; **the alternate configurations are considered experimental until noted otherwise**.

Table 1.1-3 Comparison table for alternative sensing end options.

Option	Total Cost	Components in Brief	Component Cost	Component URLs
Acrylic Window	$41.00	BlueRobotics Flange BlueRobotics Flange Cap	$27.00 $14.00	https://tinyurl.com/ykemz7zp https://tinyurl.com/54phjb9w
Default	$388.70	BlueRobotics Flange Custom Glass Window Custom Retaining Ring Gasket M2 screws x6	$27.00 $310.00 $50.00* $0.85 $0.85	https://tinyurl.com/ykemz7zp CUSTOM CUSTOM https://tinyurl.com/27adtff2 https://tinyurl.com/3z94u9tm
Custom Flange	$248.70	Custom Flange Edmund Optics Glass Window Custom Retaining Ring BlueRobotics Locking Cord Gasket M2 screws x6	$100.00* $87.00 $50.00* $10.00 $0.88 $2.30	CUSTOM https://tinyurl.com/5c83wudy CUSTOM https://tinyurl.com/mb3enbx6 https://tinyurl.com/2hek85fa https://tinyurl.com/59c8yffc
***Estimated Cost**				

1.1.5 Bill of Materials

The Bill of Materials (BOM) for the PIXIE is presented here as a series of Tables for each PIXIE subassembly providing component names, descriptions, prices, and URLs where applicable. The tables are ordered by their numerical callouts in the following diagram, Figure 1.1-16.

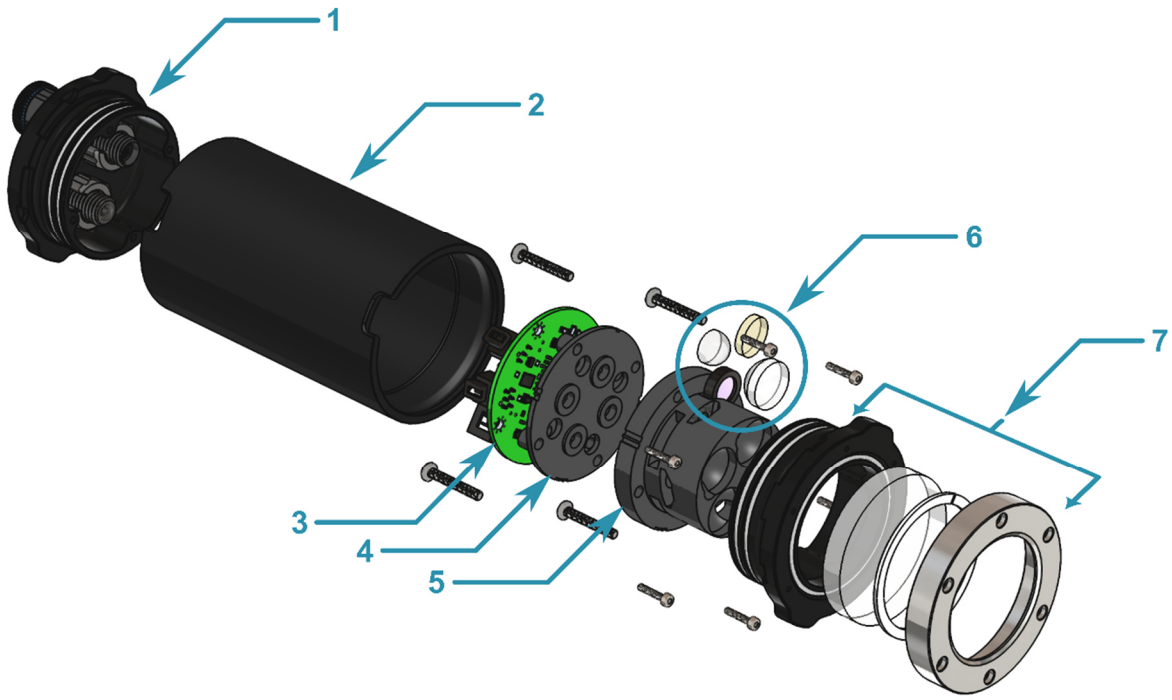


Figure 1.1-16 Exploded view of the PIXIE for BOM, with indicated sub-assembly callouts.

In respective numerical order, the subassemblies are: the End Cap assembly, the Tube Housing, the PIXIE PCB, the Optical Cap, the Optical Stack, the Channel Optics, and the Sensing End assembly.

The End Cap assembly includes cabling (not depicted). The PIXIE PCB includes Excitation LEDs (not depicted). The Channel Optics includes components for all four default channels (only one depicted). All fasteners are included with the Sensing End assembly.

An additional table includes the cost of uncategorized components, such as the MSP-FET Programming device. A final table presents the total device cost of the default configuration PIXIE and a single-channel PIXIE, along with the cost of a single-channel alternate configuration PIXIE that replaces the Tube housing and Sensing End with acrylic parts, as described in Section 1.1.4.

Table 1.1-4 below provides the BOM for the End Cap subassembly.

Table 1.1-4 End Cap subassembly BOM.

Part	Mfg./PN	Description	Cost	Qty	URLs
Cable	BlueRobotics SKU: BR-100452	PUR Subsea Cable 5 m jacketed underwater cable, 4 cond 24 AWG.	$25.00	1	https://tinyurl.com/muftk8ef
Penetrator	BlueRobotics SKU: BR-100870-045	WetLink Penetrator Watertight gland for cable entry.	$12.00	1	https://tinyurl.com/2s495w95
Vent/Plug	BlueRobotics SKU: BR-100783	M10 Enclosure Vent and Plug For pressure relief + vacuum testing.	$10.00	1	https://tinyurl.com/5btvah9m
End Cap	BlueRobotics SKU: BR-100276-002	Watertight Enclosure Flange Cap 2x M10 holes for Penetrator/Cable + Vent/Plug.	$32.00	1	https://tinyurl.com/4jahuvwv
Total					**$79.00**

Table 1.1-5 below provides the BOM (single component) for the Tube Housing.

Table 1.1-5 Tube Housing BOM.

Part	Mfg./PN	Description	Cost	Qty	URLs
Housing	BlueRobotics SKU: BR-100534-100	Watertight Enclosure Tubes 2″ series 100 mm aluminum locking tube.	$46.00	1	https://tinyurl.com/yev4cykd
Total					**$46.00**

Table 1.1-6 below provides the BOM for the PIXIE PCB and default Excitation LEDs. The PIXIE PCB is taken as one part for the purpose of this document. For the BOM of the PIXIE PCB, visit https://github.com/KylePark0/PIXIE/tree/main/Electrical. Open the "PIXIE_Packed_For_PCB_Way.zip" archive. This contains the BOM and placement listings of the electronic components of the PIXIE PCB. The price is based on the most recent quote obtained from PCBWay.

Table 1.1-6 PIXIE PCB and Excitation LED BOM.

Part	Mfg./PN	Description	Cost	Qty	URLs
PIXIE PCB	PCBWay	PIXIE PCB Board manufactured & populated by PCBWay	$145.00	1	-
Channel 1 LED	Marktech Optoelectronics MTE5900N-UY	Metal Can LED PC Excitation (Amber) LED	$7.35	1	https://tinyurl.com/yck8u5jb
Channel 2 LED	Marktech Optoelectronics MTE5270P-C	Metal Can LED RWT Excitation (Green) LED	$18.02	1	https://tinyurl.com/32jsaajz
Channel 3 LED	Marktech Optoelectronics MTE4600N	Metal Can LED Chl a Excitation (Blue) LED	$17.22	1	https://tinyurl.com/43pstk25
Channel 4 LED	Marktech Optoelectronics MT3650N3-UV	Metal Can LED Crude Oil Excitation (UV) LED	$14.32	1	https://tinyurl.com/bdhcz6mh
Total					**$201.91**

Table 1.1-7 and Table 1.1-8 below provides the BOM (single component) for the Optical Cap and Optical Stack, respectively. The URL provided for both tables is a link to the PLA filament used to print the PIXIE components and where the unit cost is derived from.

Table 1.1-7 Optical Cap BOM.

Part	Mfg./PN	Description	Cost	Qty	URLs
Optical Cap	-	3D Printed Optical Cap Printed in Opaque Black PLA Filament	$0.23	1	https://tinyurl.com/ymkvmyd5
Total					**$0.23**

Table 1.1-8 Optical Stack BOM.

Part	Mfg./PN	Description	Cost	Qty	URLs
Optical Stack	-	3D Printed Optical Stack Printed in Opaque Black PLA Filament	$0.83	1	https://tinyurl.com/ymkvmyd5
Total					**$0.83**

Table 1.1-9 below provides the BOM for the Channel Optics.

Table 1.1-9 Channel Optics BOM.

Part	Mfg./PN	Description	Cost	Qty	URLs
Condenser Lens	Edmund Optics 36-166	15 mm x 12 mm Aspheric Condenser Lens	$27.00	4	https://tinyurl.com/5n6bs9y8
Channel 1 Excitation Filter	Edmund Optics 84-697	600 nm short-pass filter, OD 4 Matched to CH1 Excitation LED	$189.00	1	https://tinyurl.com/bd4x7hyf
Channel 1 Emission Filter	Edmund Optics 87-741	661 ± 26 nm band-pass filter, OD 6 Matched to PC fluorescence	$249.00	1	https://tinyurl.com/ybudkw78
Channel 2 Excitation Filter	Edmund Optics 84-695	550 nm short-pass filter, OD 4 Matched to CH2 Excitation LED	$189.00	1	https://tinyurl.com/2cv268wr
Channel 2 Emission Filter	Edmund Optics 87-738	578 ± 16 nm band-pass filter, OD 6 Matched to RWT fluorescence	$249.00	1	https://tinyurl.com/acajbywd
Channel 3 Excitation Filter	Edmund Optics 84-692	475 nm short-pass filter, OD 4 Matched to CH3 Excitation LED	$189.00	1	https://tinyurl.com/23ndn38a
Channel 3 Emission Filter	Edmund Optics 67-024	692 ± 47 nm band-pass filter, OD 6 Matched to Chl a fluorescence	$249.00	1	https://tinyurl.com/4aa3ymby
Channel 4 Excitation Filter	Edmund Optics 84-689	400 nm short-pass filter, OD 4 Matched to CH4 Excitation LED	$189.00	1	https://tinyurl.com/mw6hvcnd
Channel 4 Emission Filter	Edmund Optics 86-341	466 ± 45 nm band-pass filter, OD 6 Matched to Crude Oil fluorescence	$249.00	1	https://tinyurl.com/2wyananm
Focusing Lens	Edmund Optics 45-937	10 mm N-BK7 Half-ball Lens	$60.00	4	https://tinyurl.com/3bvyc5k4
Focusing Lens O-Ring	McMaster-Carr 9396K15	Dash No. -010 Silicone O-Rings, 70A Hardness Mounts Focusing Lens to Optical Stack and Cap	$0.25	4	https://tinyurl.com/ff5ashze
Total					**$2,101.00 ($525.25 per CH)**

Table 1.1-10 below provides the BOM for the Sensing End subassembly. The URL provided for the Custom Window leads to the Edmund Optics custom glass form. No URL is provided for the Custom Retaining Cap; consult local machine shops for the machining and finishing of the Retaining Cap. The cost of the Retaining Cap is estimated based on local material, machining, and anodizing prices.

Obtain the optical/mechanical drawings and specifications for the Custom Window and Retaining Cap by visiting the PIXIE Hardware repository at:

https://github.com/KylePark0/PIXIE/tree/main/Hardware

Table 1.1-10 Sensing End subassembly BOM.

Part	Mfg./PN	Description	Cost	Qty	URLs
O-Ring Flange	BlueRobotics SKU: BR-100628-998	O-Ring Locking Flange Locking cord and 3x O-Rings included	$29.00	1	https://tinyurl.com/4w8hsyp5
M3 Screws	McMaster-Carr 92290A761	Stainless steel pan head, 18 mm thread-length	$0.28	4	https://tinyurl.com/y2dxzntx
M2 Screws	McMaster-Carr 92290A017	Stainless steel pan head, 10 mm thread-length	$0.15	6	https://tinyurl.com/3z94u9tm
Custom Window	Edmund Optics	Custom BOROFLOAT® Window 46.5 mm diameter x 6.5 mm thickness	$310.00	1	https://tinyurl.com/35vazptr
PTFE Gasket	McMaster-Carr 9560K74	Chemical-resistant spacing gasket Dash No. 222	$0.85	1	https://tinyurl.com/27adtff2
Retaining Cap	-	Custom 6060-T6 Aluminum Retaining Cap Anodized, compatible with O-Ring Flange	$50.00	1	-
Total					**$390.28**

Table 1.1-11below provides the BOM for uncategorized or “Other” components.

Table 1.1-11 Other components BOM.

Part	Mfg./PN	Description	Cost	Qty	URLs
Battery	Various	Backup Battery for PIXIE PCB; optional	$4.25	1	https://tinyurl.com/3s5wfnd9
Programmer	Texas Instruments MSP-FET	MSP-FET JTAG Programmer Needed to program the MSP430/PIXIE PCB	$138.00	1	https://tinyurl.com/y3euach5
O-Ring Grease	BlueRobotics SKU: BR-100484	Silicone O-Ring grease. Many alternatives available; Molykote 111 stocked by BlueRobotics	$4.00	1	https://tinyurl.com/mwsuknu7
Total					**$146.25**

Table 1.1-12 below provides the total cost of the default-configuration PIXIE. Beneath the total cost, the cost of two other configurations are offered: the cost of a default-configuration PIXIE with only one channel installed, and the cost of a single channel PIXIE using only acrylic housings and sensing ends as described in Section 1.1.4.

Table 1.1-12 Total cost of default-configuration PIXIE.

Assembly	Description	Cost
1	End Cap subassembly	$79.00
2	Tube Housing	$46.00
3	PIXIE PCB + LEDs	$201.91
4	Optical Cap	$0.23
5	Optical Stack	$0.83
6	Channel Optics	$2,101.00
7	Sensing End subassembly	$390.28
X	Other	$146.25
Total	Default-Configuration PIXIE, Four Channels	**$2,965.50**
	Default-Configuration PIXIE, One Channel	**$1,389.75**
	Acrylic housing PIXIE, One Channel	**$1,042.05**

1.2 Assembly

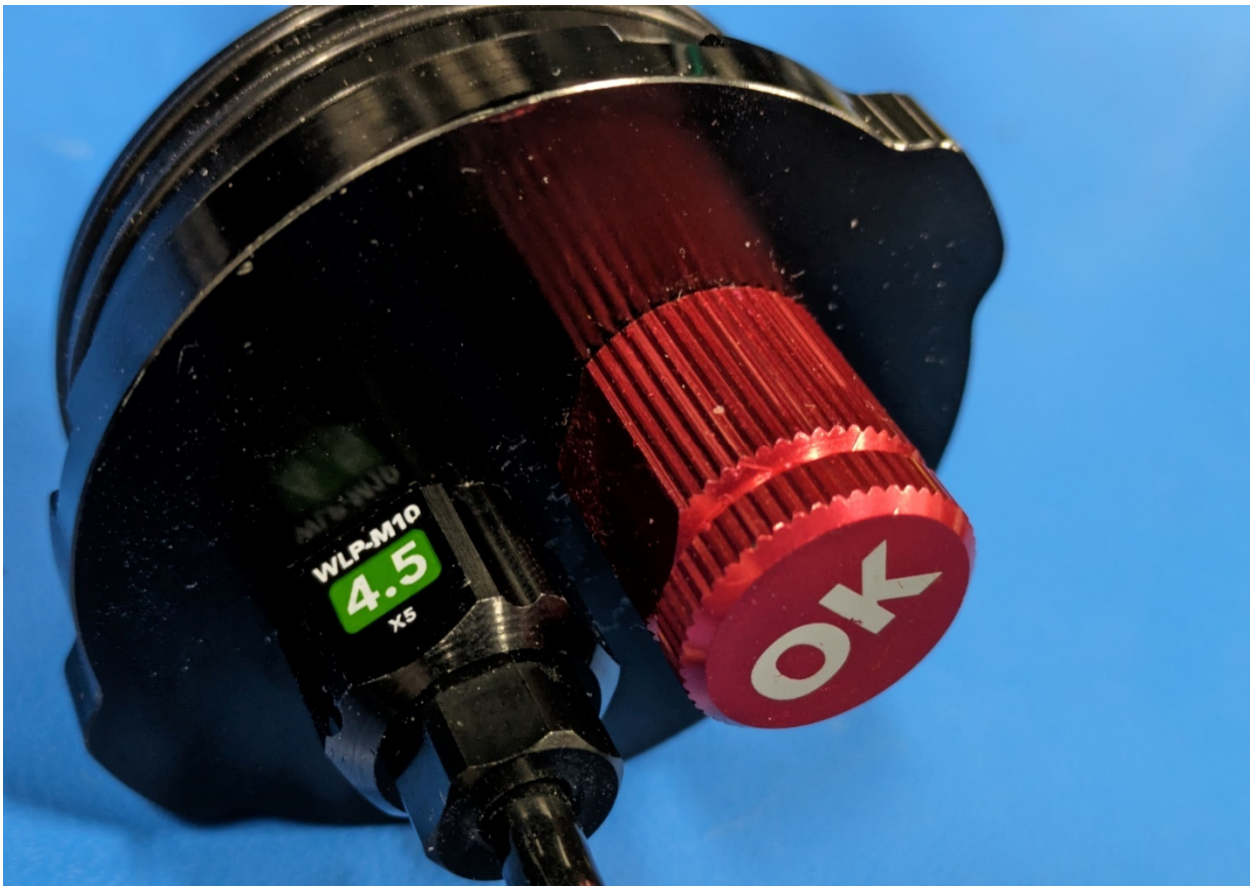
In this section, instructions for how to assemble the components of the default-configuration PIXIE are provided. This section assumes that all components have been acquired, including the machined and 3D printed parts. This assembly assumes the BlueRobotics end cap has been fully assembled. For assembly instructions for all the off-the-shelf BlueRobotics housing components, consult the user guide at:

https://bluerobotics.com/learn/watertight-enclosure-wte-assembly-new/

1.2.1 Hardware

Assembly instructions for the default-configuration PIXIE are provided here in numerical order. Only one set of optics is installed for brevity. **You must choose your intended Channels and their corresponding Optics *before* beginning assembly**.

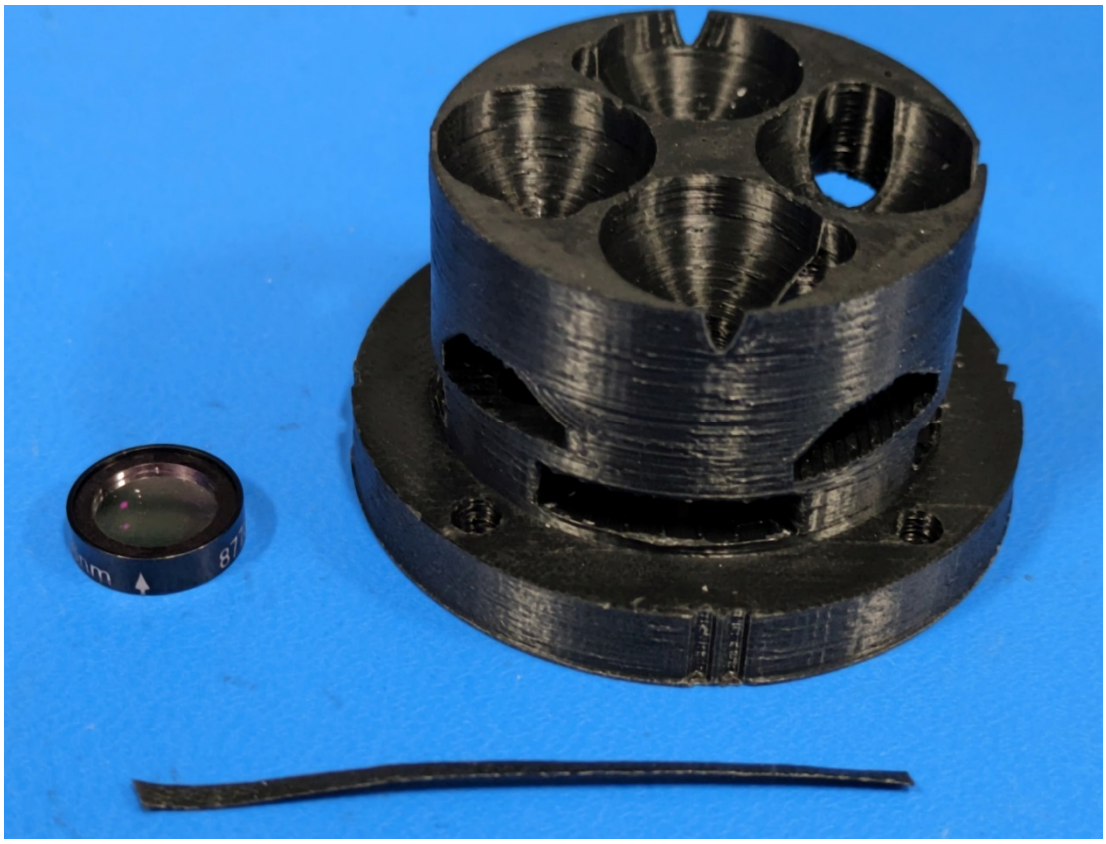
1. First, assemble the End Cap and set it aside. It should appear as in the photograph below, though component colors may vary.



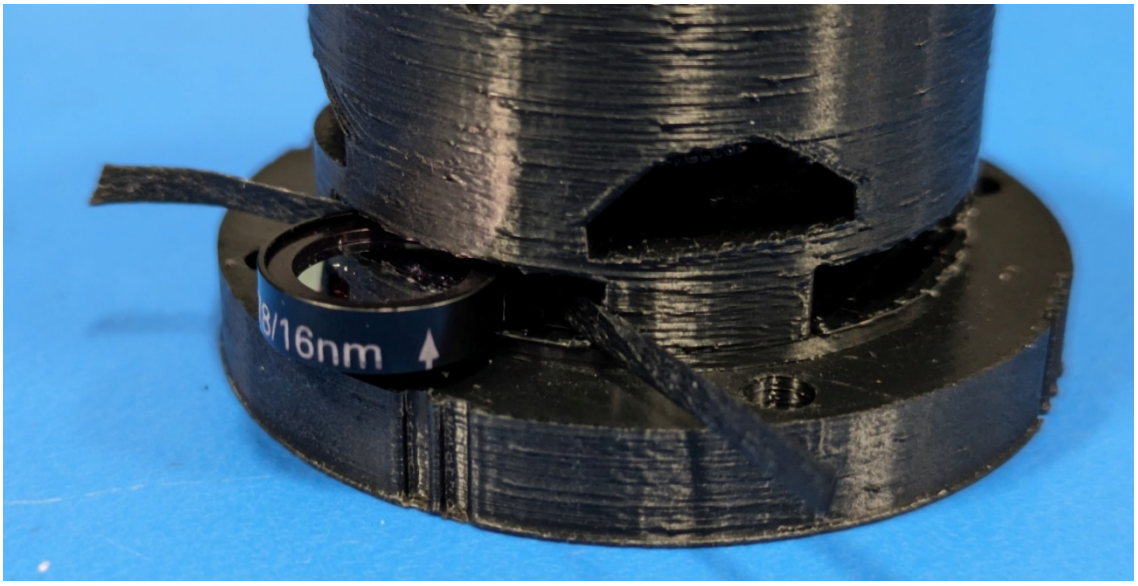
2. Next, prepare the Optical Stack. For instructional purposes, only one optical channel will be installed. The procedure for installing additional channels is identical. Gather the Optical Stack and the Optical Cap and the Emission Filter, Excitation Filter, Condenser Lens, Focusing Lens, and Lens O-ring for each channel you intend to install. The installation is fixed in place using black vinyl tape. Waxed nylon cord is helpful, but not strictly necessary. Uncoated disposable gloves should be worn when handling Optical Filters and Lenses.

   The Optical Stack and Optical Cap have notches around their outermost edge to help keep track of which Channel you are installing optics for.

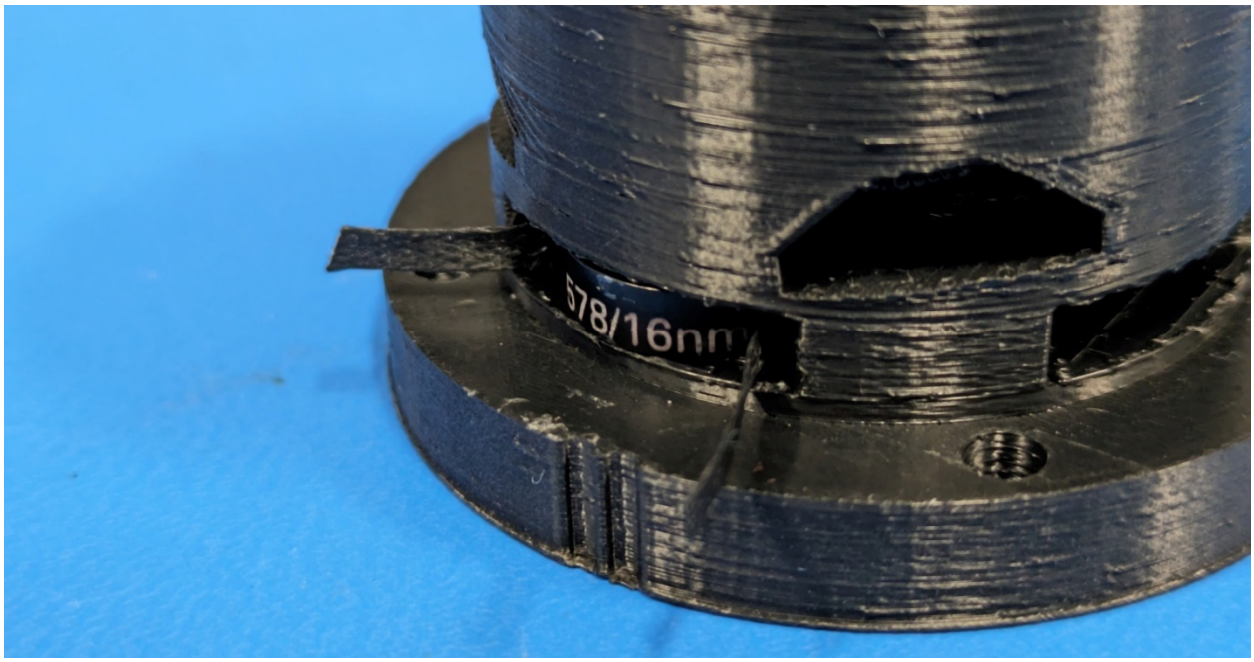
3. The Emission Filter(s) should be installed first, due to their tight fit. To install an Emission Filter, first cut a short length of nylon cord or folded-over vinyl tape. This is used to simplify future removal of the Filter.



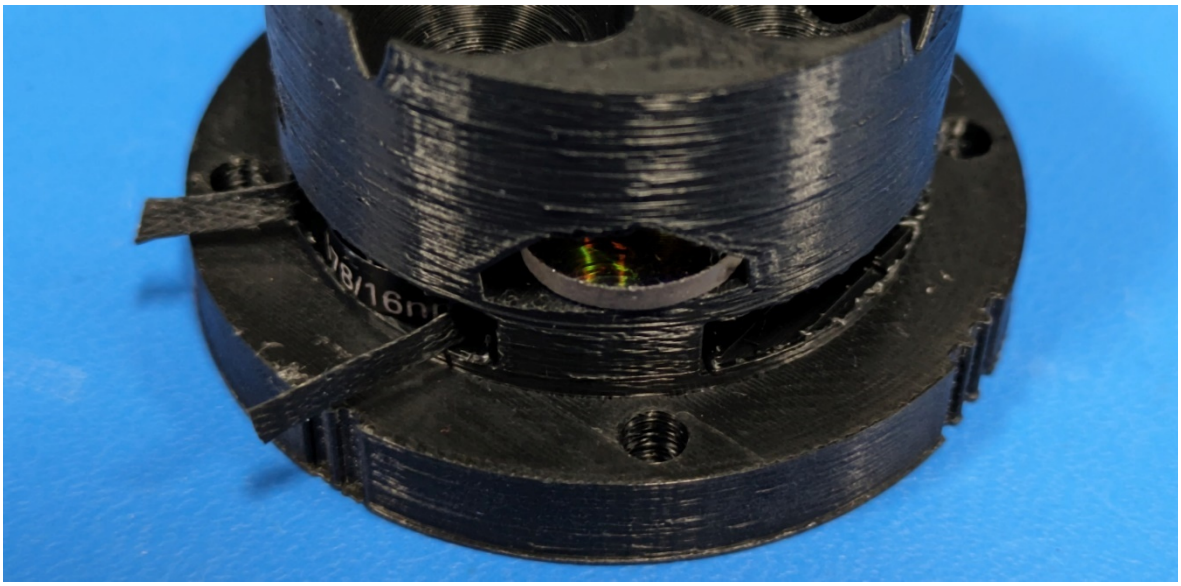
4. Wrap the filter in cord/tape and insert it into the slot indicated by the following photograph. Optical filters should be installed with their direction indicators (V-shaped notch or printed arrow) pointing *towards* the light *source*.



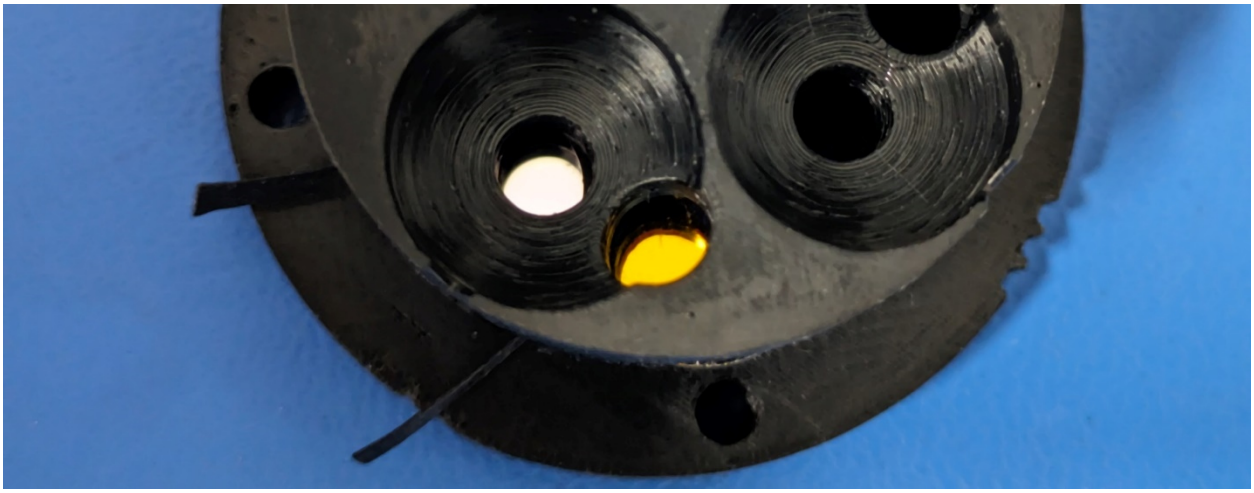
5. Gently press the Emission Filter into the slot. The fit should be tight but not require excessive manual pressure nor require clamping tools: if the fit is too tight, material may need to be removed from the Stack with a blade or gouge, or the Stack may need to be reprinted at a slightly larger scale. Pull on the cord/tape to remove the Filter. Once in place, the protruding cord/tape should be cut back, with each exposed length being about half the length of the slot.



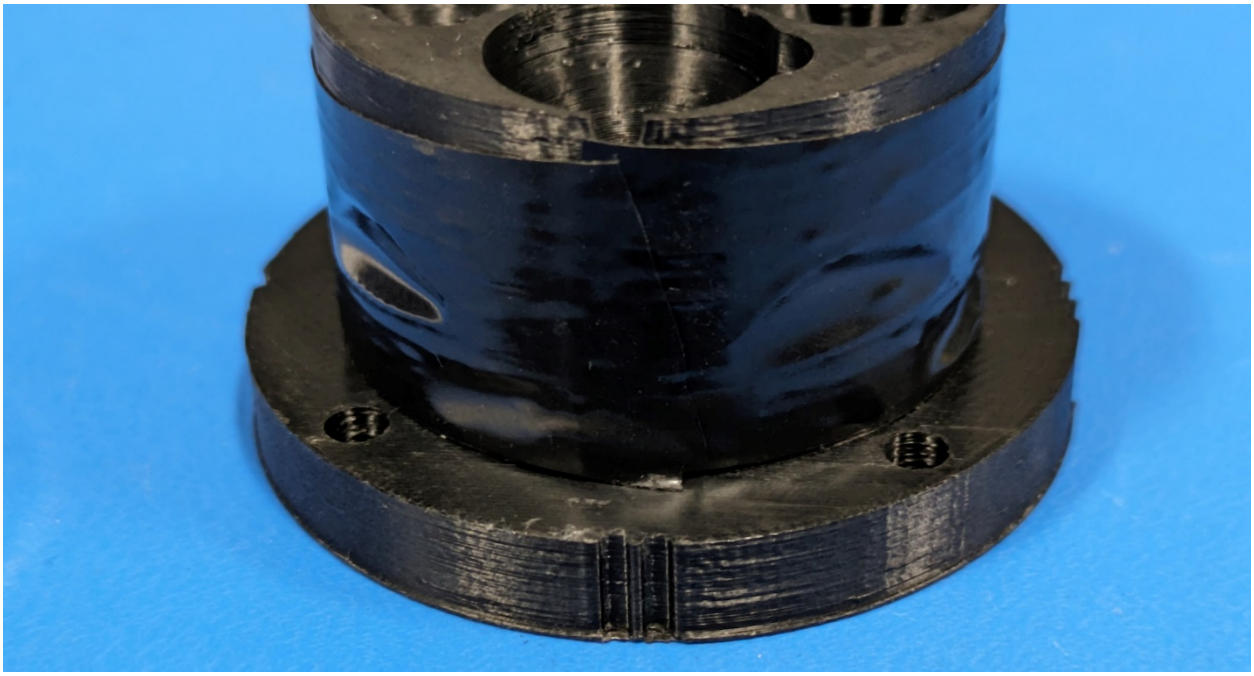
6. Install the Excitation Filter. The fit should be noticeably looser than the Emission Filter. When installing multiple channels, be careful not to tilt or spin the Stack too quickly, as the Excitation filters may slip out.



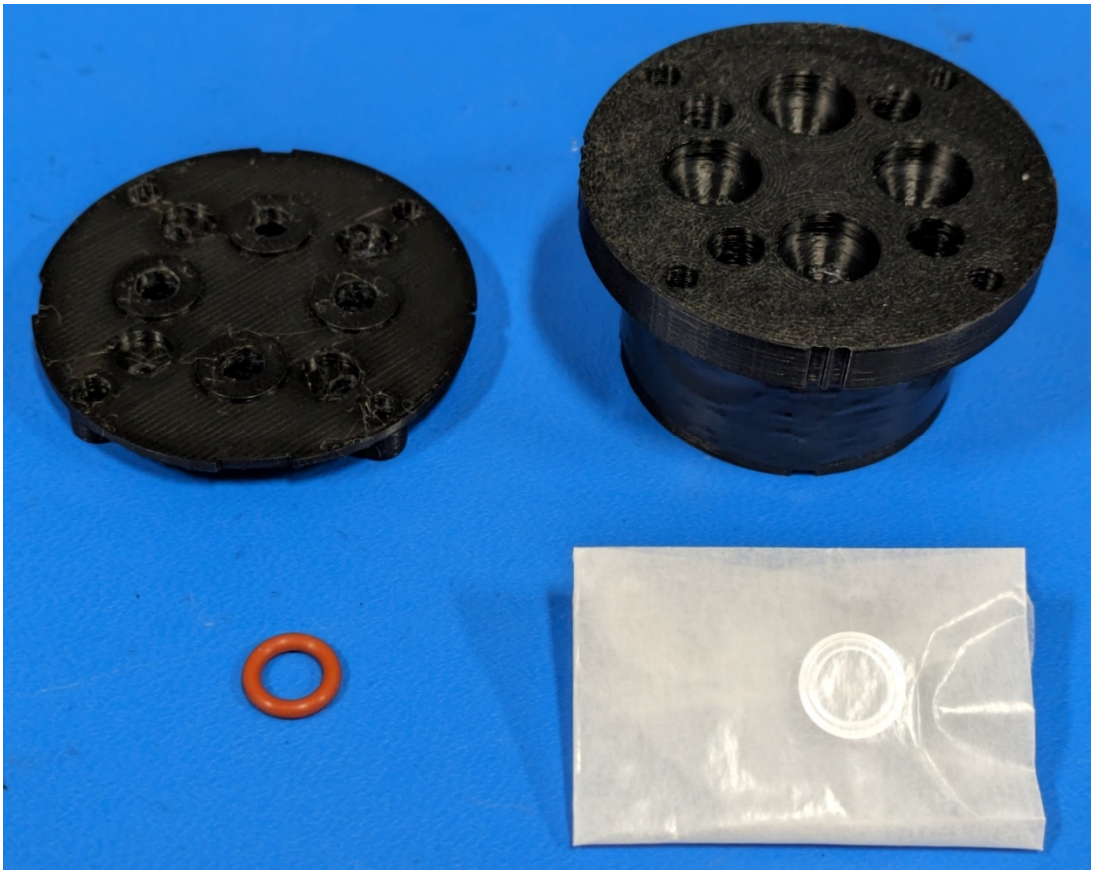
7. When correctly installed and viewed from above, both of a channels' filters should be visible. Be careful to install the correct Excitation Filters when installing multiple channels, as they will be more difficult to verify after installation. In this example, filters for Rhodamine Water Tracer (RWT) fluorometry have been installed in Channel 2, as noted by the two divots in the outer rim of the Optical Stack.



8. Once all the Filters have been installed, the Filters are fixed in place using black vinyl tape. The tape should be long enough to wrap the optical stack exactly once. Ensure that the tape does not overlap or meet ends near the screw holes in the bottom of the Stack: the Optical Stack has generous clearance with the Tube Housing everywhere *except* the screw holes.

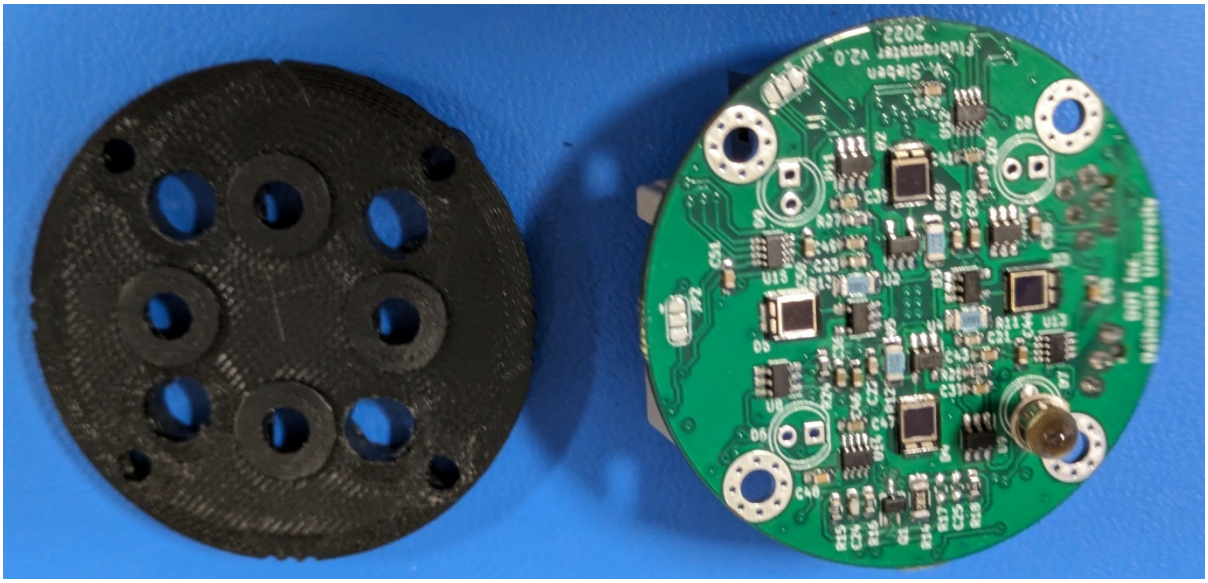


9. Next, the Focusing Lens(es) will be installed and held in place by the Lens O-ring(s) and the Optical Cap. The Optical Cap mates with the underside of the Optical Stack.

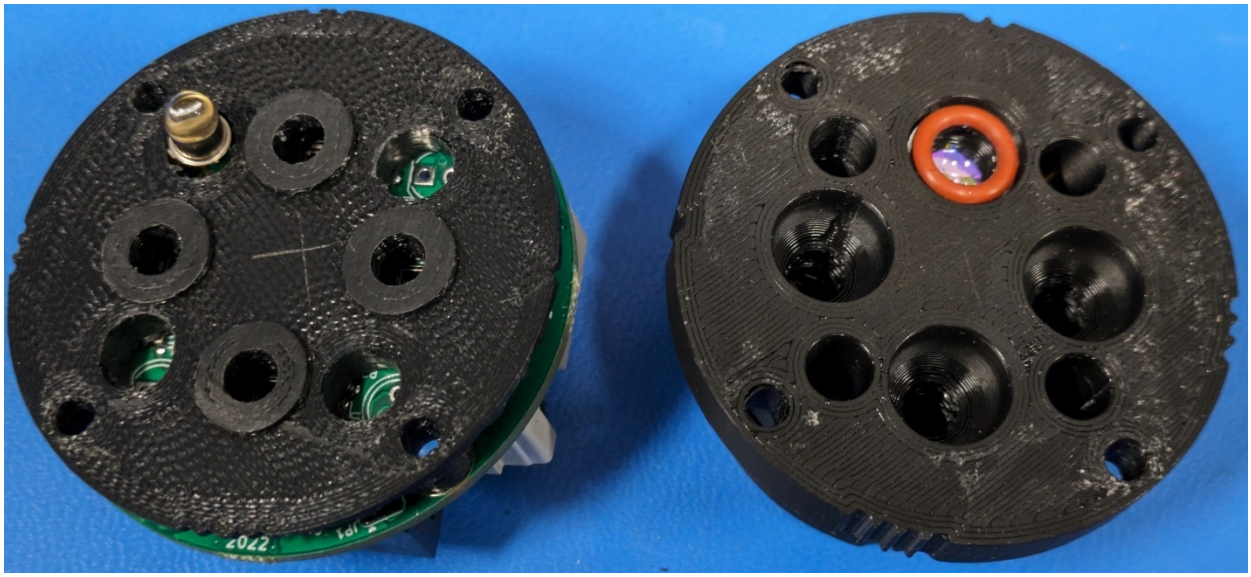


10. Insert the Focusing Lens into the indicated cavity, followed by the Lens O-ring. Repeat for every installed channel.

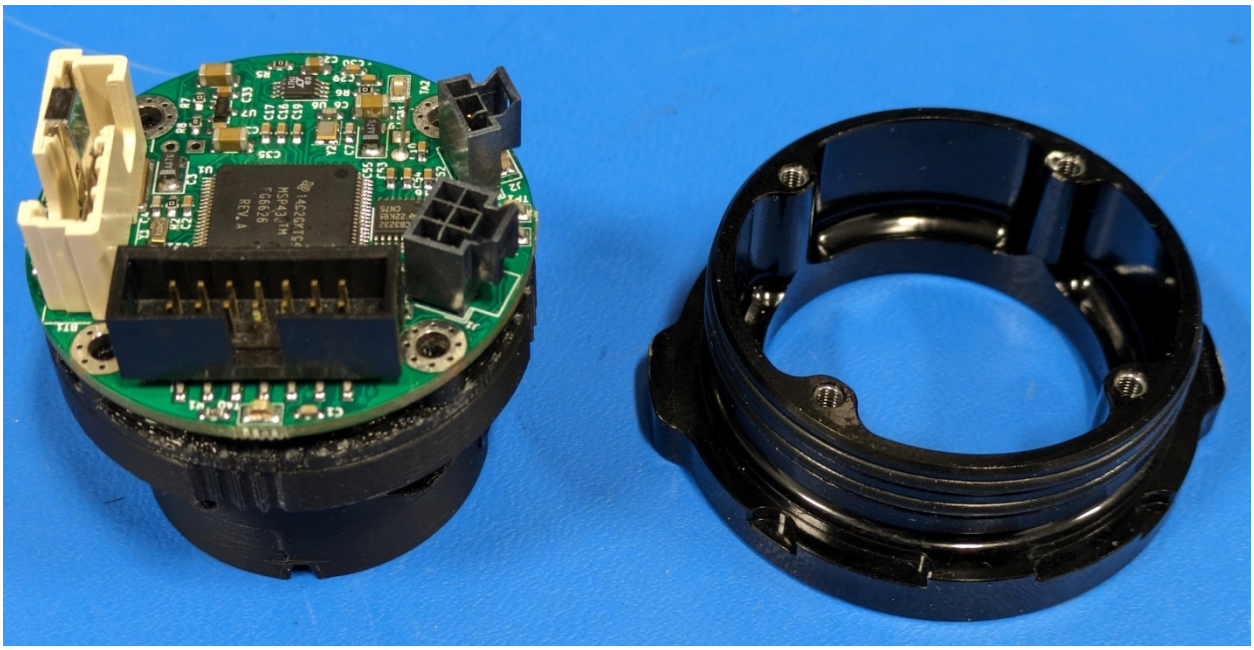
11. Place the Optical Cap onto the PIXIE Printed Circuit Board (PCB). The Optical Cap helps align the Excitation LEDs and shrouds the detector photodiodes from all light except that which arrives through the Emission Filter and paired lenses.



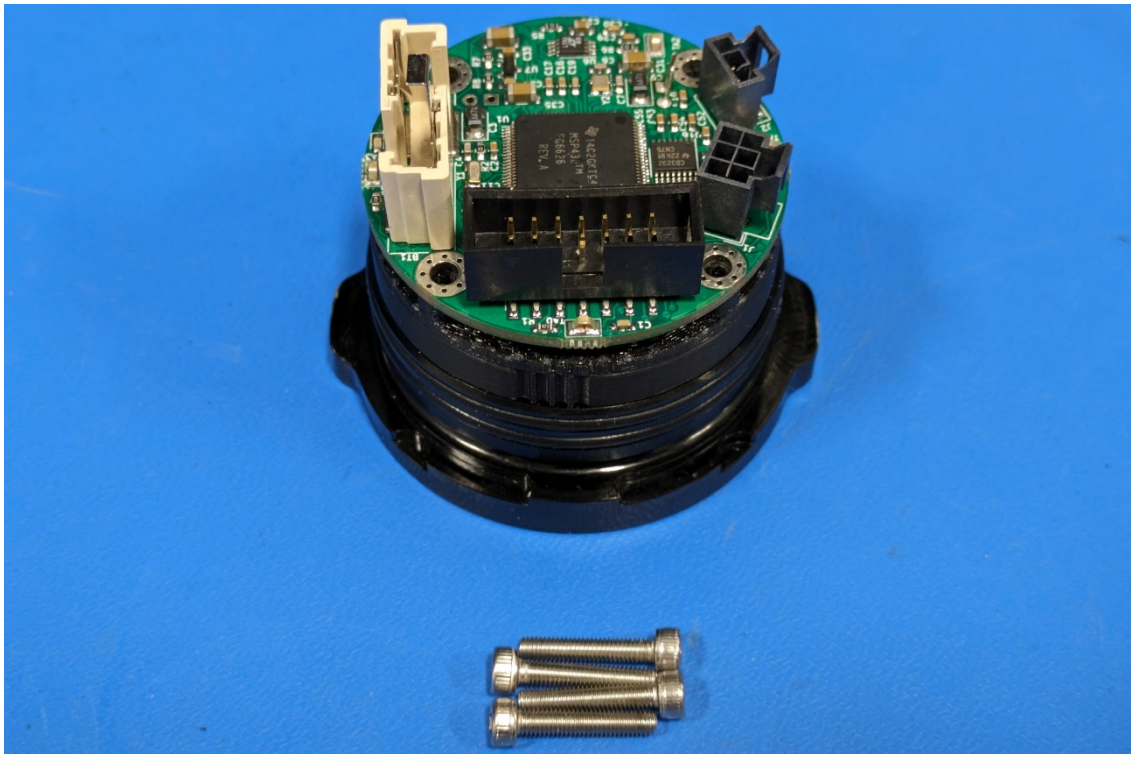
12. Once the PIXIE PCB is fitted with the Optical Cap, place the Optical Cap onto the Optical Stack. Depending on the tolerances between the two prints, the Cap will either snap into place and hold on its own or be a loose fit that falls off due to gravity. In the latter case, the whole assembly can be temporarily secured with vinyl tape until the it is secured in place with screws. The grooves on the side of each part should be aligned to help you keep track of which channel is which. Ensure that the correct Excitation LED and Filter set are aligned, as this can be difficult when installing all four channels.



13. Next, the Optical Stack assembly is mated to the BlueRobotics Locking O-ring flange. Ensure the flange's Radial O-rings are installed before assembly (consult the BlueRobotics User Guide provided in Step 1). Insert the assembly into the flange and align the screw holes.



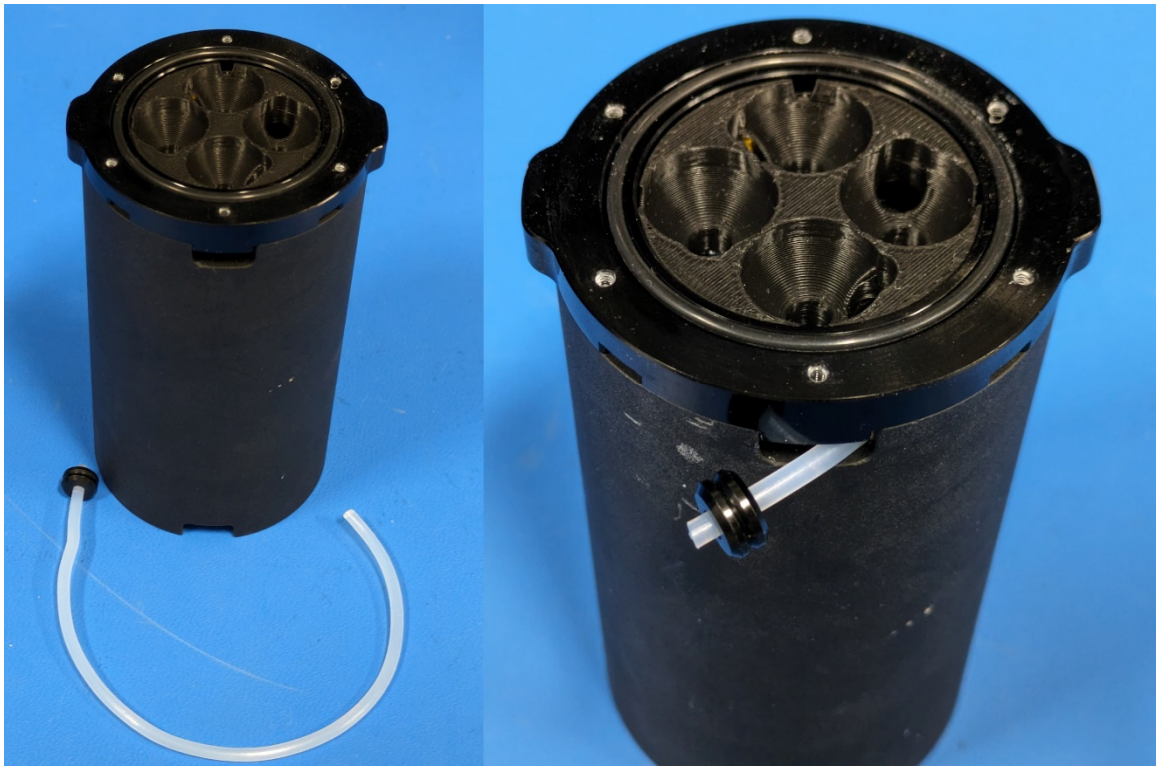
14. Using four M3 screws, screw the assembly to the flange. The specified screws are steel whereas the flange is aluminum. Over-torquing the screws may strip the flange's threads and may warp the plastic parts.



15. Insert the CR2032 Cell Backup Battery. Notice the correct orientation of the cell in the photograph below. Once this step is completed, the PIXIE is assembled enough for programming and basic testing/debugging. Consult Section 1.2.2 and Section 1.3 for early instruction on how to cable and program the device.

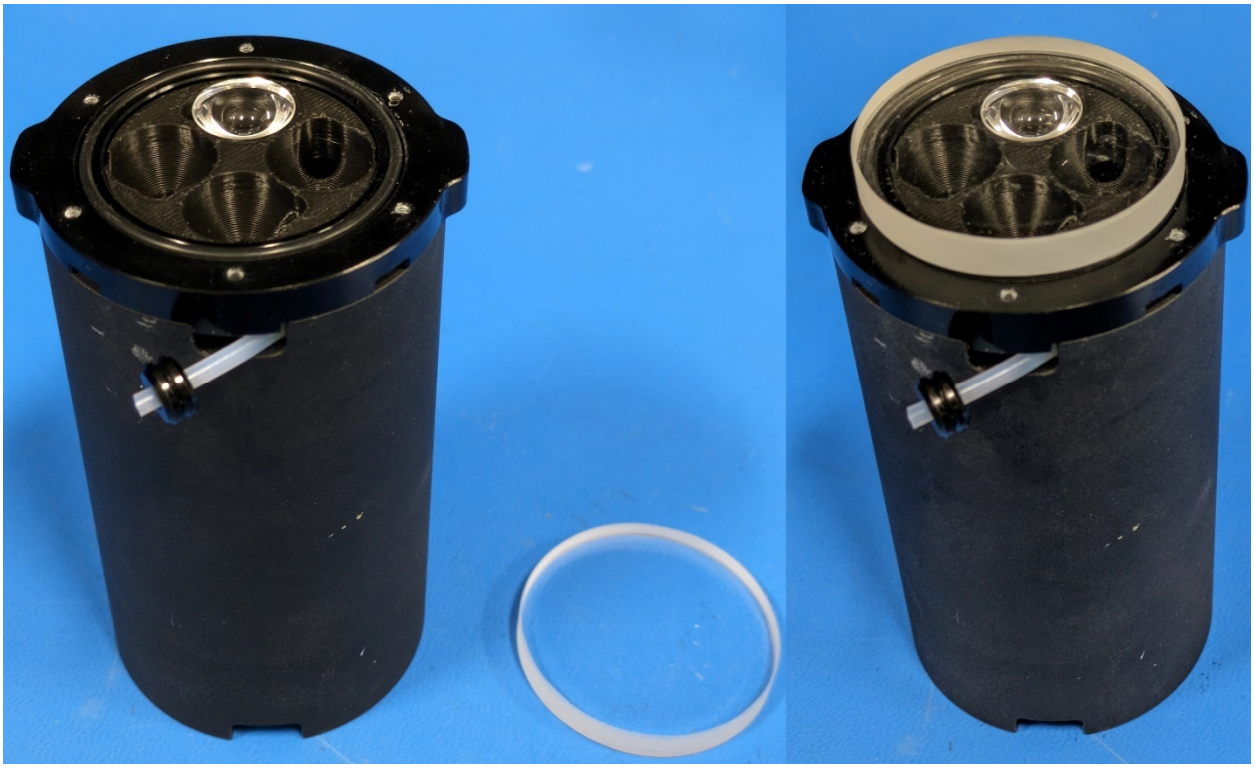
16. Fit the assembly into the BlueRobotics Tube Housing. Make sure the device has been correctly assembled before this step, as the flange can be difficult to remove later. Ensure the O-ring flange is well lubricated with silicone O-ring grease, and match the alignment tab of the flange to the slot of the tube.

17. Insert the Locking Cord to fix the flange in place. The cord is pushed through the groove until it has wrapped around flange and makes contact with itself.

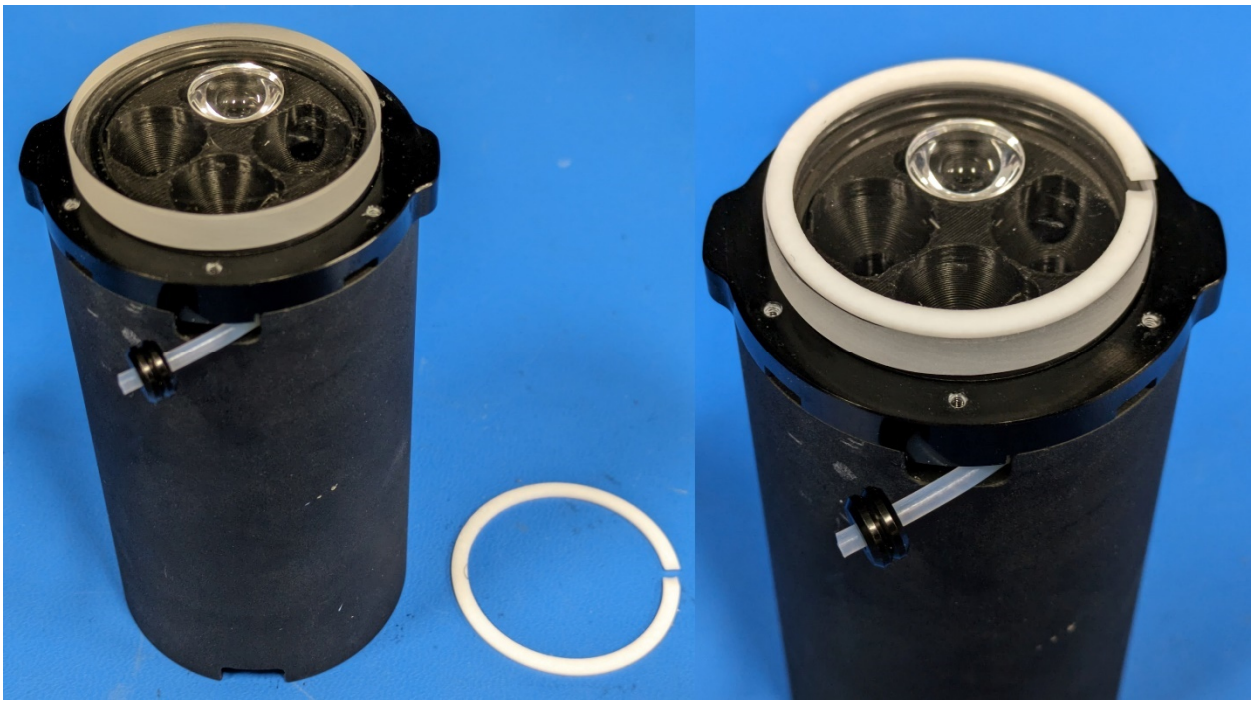


18. Place a Condenser Lens into each of the populated channels.

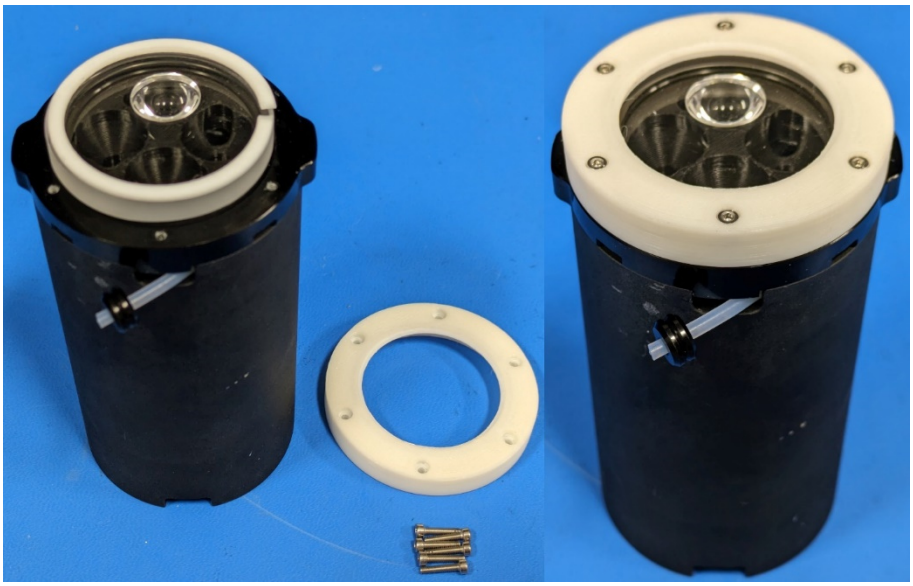
19. Place the Glass Window onto the Face O-ring of the flange. Ensure the placement is even around the O-ring and that no screw holes are covered.



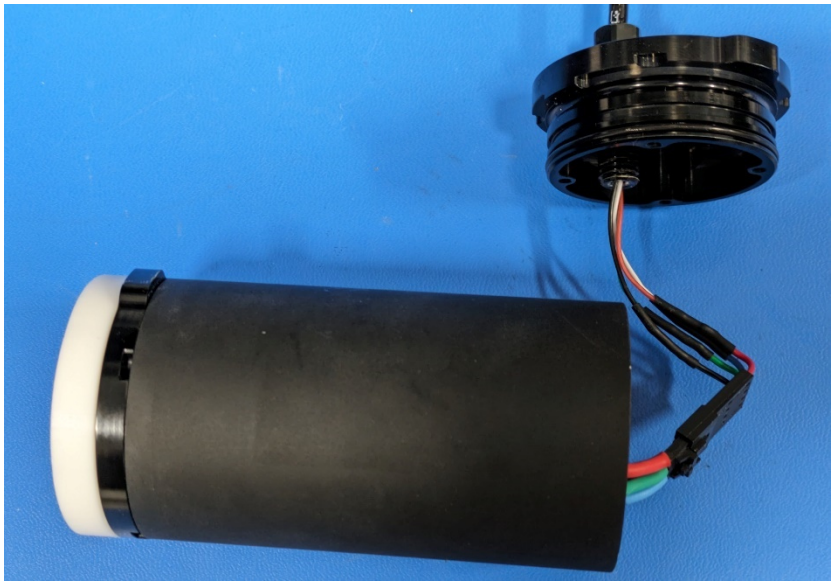
20. Place the PTFE Gasket onto the Glass Window. Do not attempt to place the Retaining Cap until the gasket is placed on the window, as the metal part may damage or scratch the optical surface of the window.



21. Place the Retaining Cap over the gasket and window, aligning the screw holes to those of the flange. The Retaining Cap is held in place with six stainless steel M2 screws. These screws will damage the flange if over-torqued. All six screws should be progressively tightened, rather than one at a time, to ensure no segment of the Glass Window is put under extreme stress.



22. The sensing end of the PIXIE is now fully assembled and can be removed from the Tube Housing as needed without disassembly (i.e. without removing the Retaining Cap or Glass Window). The End Cap set aside in Step 1 can now be installed by repeating Step 16. The connectors should be mated to the PIXIE at this point; if the cable has not been prepared in advance, consult the following section, Section 1.2.2. If the PIXIE PCB has not been programmed, consult Section 1.3.



23. Insert a second Locking Cord into the End Cap to finish assembling the PIXIE.

1.2.2 Cable and Interface

The PIXIE communicates using the RS-232 protocol through a cable connection with at least 4 conductors (power, ground, RS-232 Tx, and RS-232 Rx). The PIXIE's MSP430 microcontroller handles communication using its Universal Asynchronous Receiver/Transmitter (UART) peripheral and two-way RS-232 driver chip. You can connect the PIXIE to any device that is RS-232 compatible. Instructions for assembling a device-to-PC cable interface are provided here for reference; these can be extended to whatever watertight connection your application requires.

The Terminal (PC) end of the cable should be prepared with a standard DB-9 RS-232 plug connector. The Rx, Tx, and GND pins of the connector should be soldered to the cable conductors (green, white, and black respectively). The VCC (red) and GND (black) conductors should be left bare, or use a separate set of connectors, for connection with a benchtop power supply or battery pack. The GND conductor should be split or daisy-chained to both connectors from one conductor to ensure a common reference is used. See Figure 1.2-1 for an example configuration.

Figure 1.2-1. Example PIXIE cable, terminal side. DB9 and Power connectors.

At the device end, the four conductors are split out across two small connectors that mate with headers J1 and J2 on the PIXIE PCB. J1 is a six-position header that receives the VCC and GND conductors. J2 is a three-position header that receives the RS-232 Tx and Rx conductors. Be sure the connections to J2 are wired carefully: the Tx pin of the DB9 connector must connect to the Rx pin of J2, and the Rx pin of the DB9 connector must connect to the Tx pin of J2. This crossover connection is

the *null modem* that allows two-way communications over one port. See Figure 1.2-2 below for an example configuration.

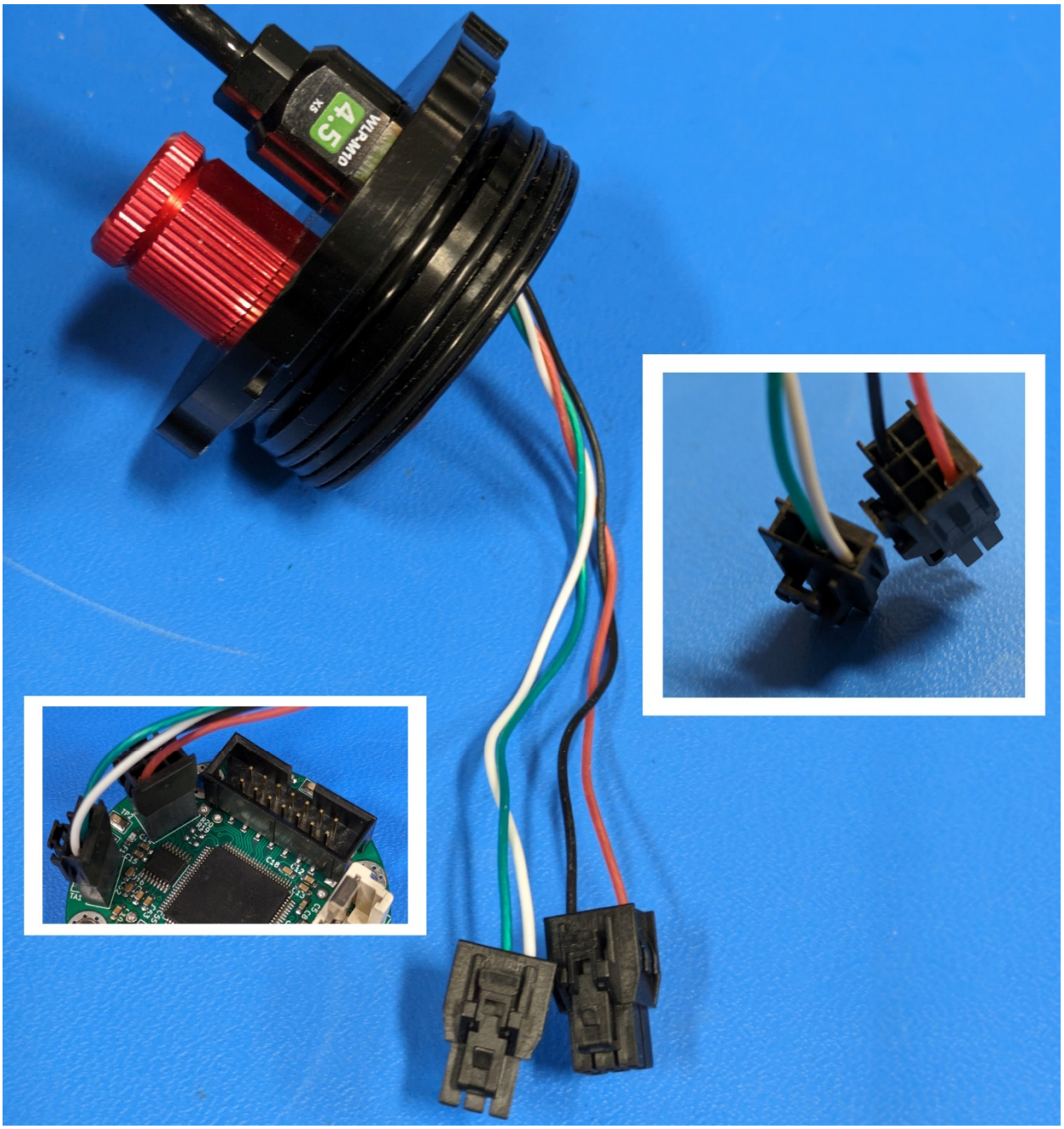


Figure 1.2-2. Example PIXIE cable, device side. End-cap assembly with three and six position connectors visible. Inset Right: rear view of connectors, for clarity. Inset Left: cables connected to their respective headers on the PIXIE PCB.

When preparing the three-position and six-position connectors for the PIXIE cable, the connectors accept socket-type wire terminations that can be easily fixed in place with a dedicated crimping tool, though this is not necessary for a one-time assembly. Instead, the wire terminators can be manually crimped with needle-nose

pilers. A method for preparing the wire terminators by hand is demonstrated below, starting with the required components in Figure 1.2-3. The method is same for both connectors, this example shows one wire prepared for the three-position connector.

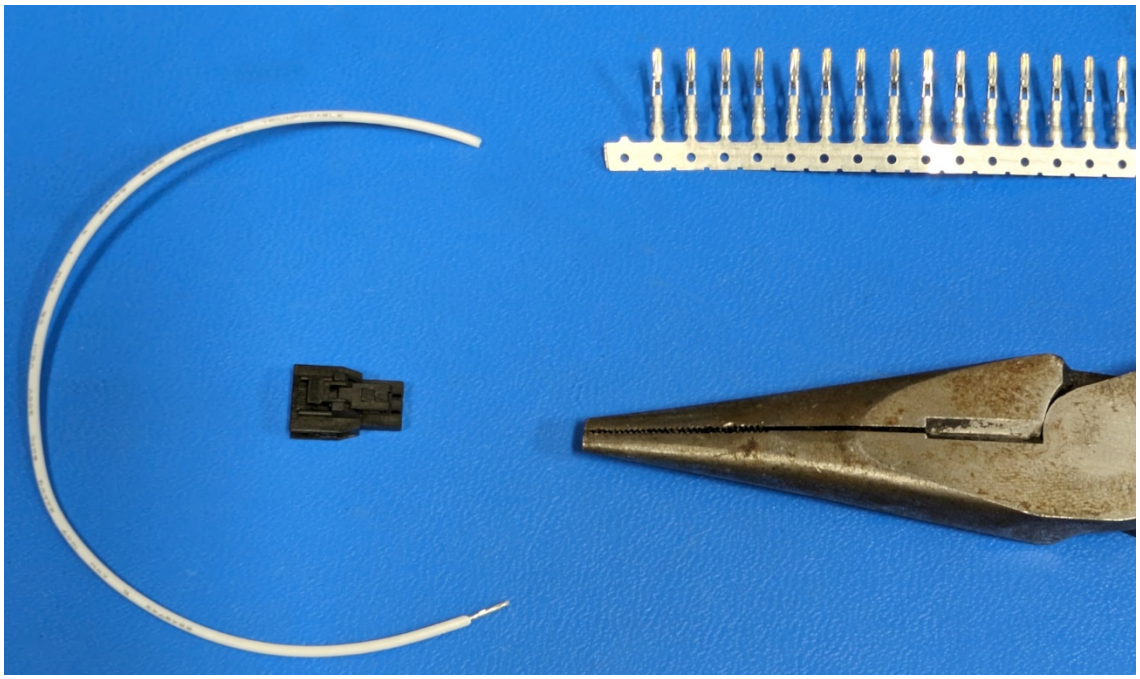


Figure 1.2-3 Preparing wire terminator sockets by hand; required components.

Strip approximately 5 mm of wire, twisting the wire strands together. Lay the bare wire across the open tin-plated side of the socket, as in Figure 1.2-4. The gold-plated end is the side that receives the receptacle pins.

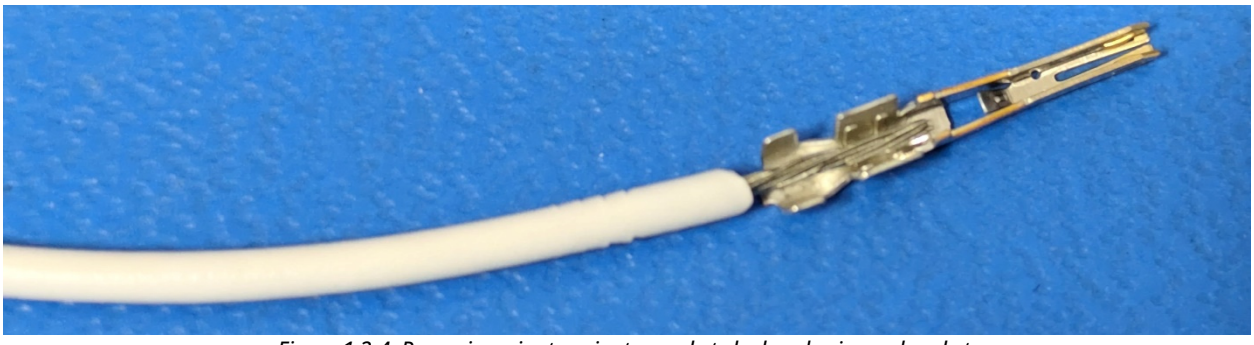


Figure 1.2-4 Preparing wire terminator sockets by hand; wire and socket.

The socket has two pairs of tabs that are bent (crimped) to hold the wire in place. The tabs each form an open “V” shaped groove. To prepare these tabs for crimping with pliers, gently pinch the tabs from above with the pliers until the tabs bend

inward slightly. *Do not* bend them to the point of making contact; this will make the tabs impossible to crimp properly.

Once the tabs are bending inward slightly, reorient the pliers so that one side rests on one tab, and the other side rests on the bottom of the socket. By squeezing the pliers, the tab will be bent downward and clamp the wire in place. Do the same for the opposite tab, completing the first crimp, as in Figure 1.2-5.

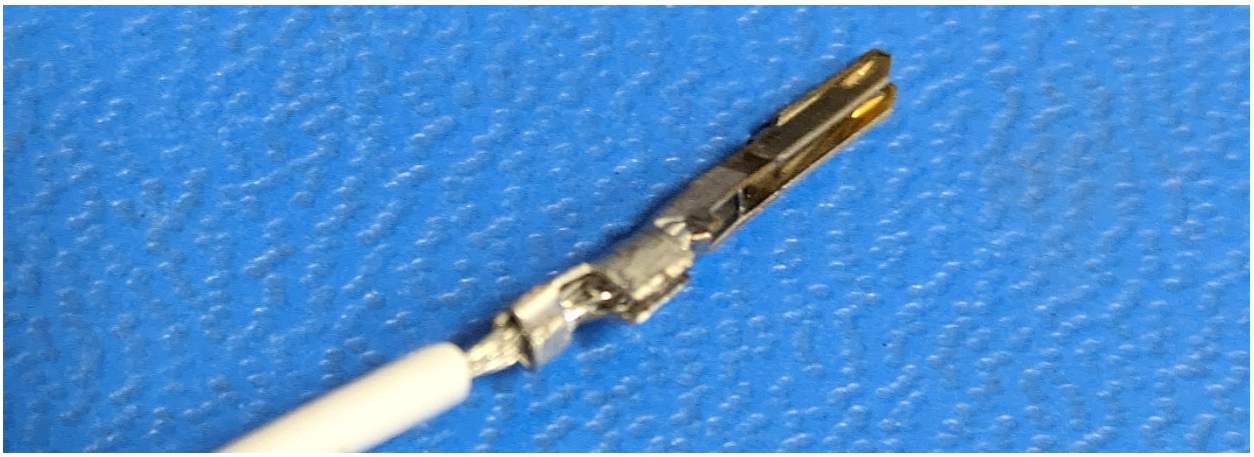


Figure 1.2-5 Preparing wire terminator sockets by hand: first crimp.

Repeat the procedure for the second set of tabs, as in Figure 1.2-6. You *may* solder the wire to the socket once the crimp connection is complete, but this is neither necessary nor recommended; too much solder will prevent the socket from seating in the connector properly.

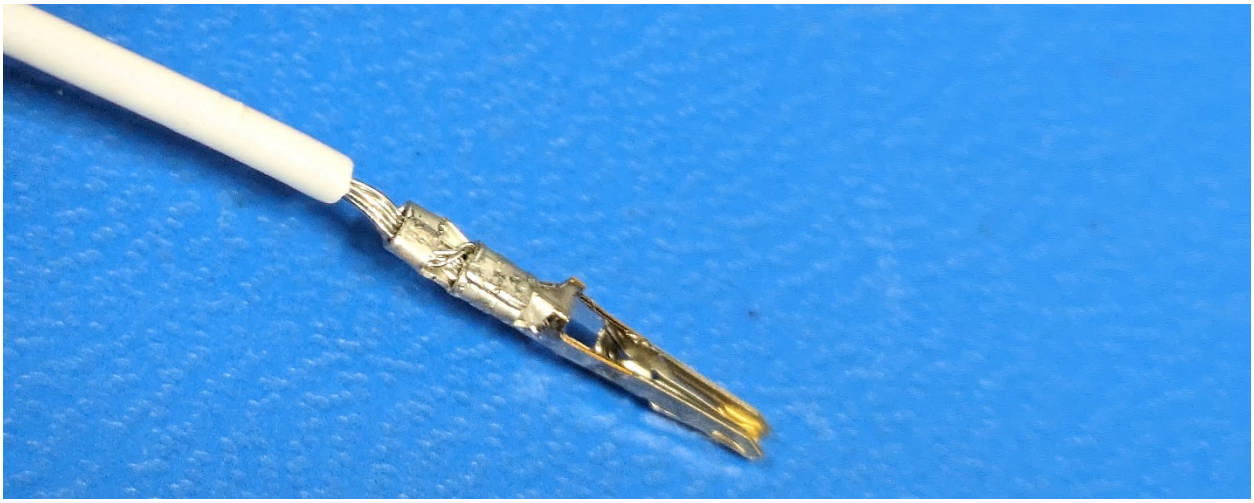


Figure 1.2-6 Preparing wire terminator sockets by hand: second crimp.

The socket is now ready to insert into the connector, as in Figure 1.2-7. **Double check the position before you insert the socket**! Once inserted, the sockets are

extremely difficult to remove without catastrophically damaging the connector. The socket will make an audible click when fully seated in the connector.

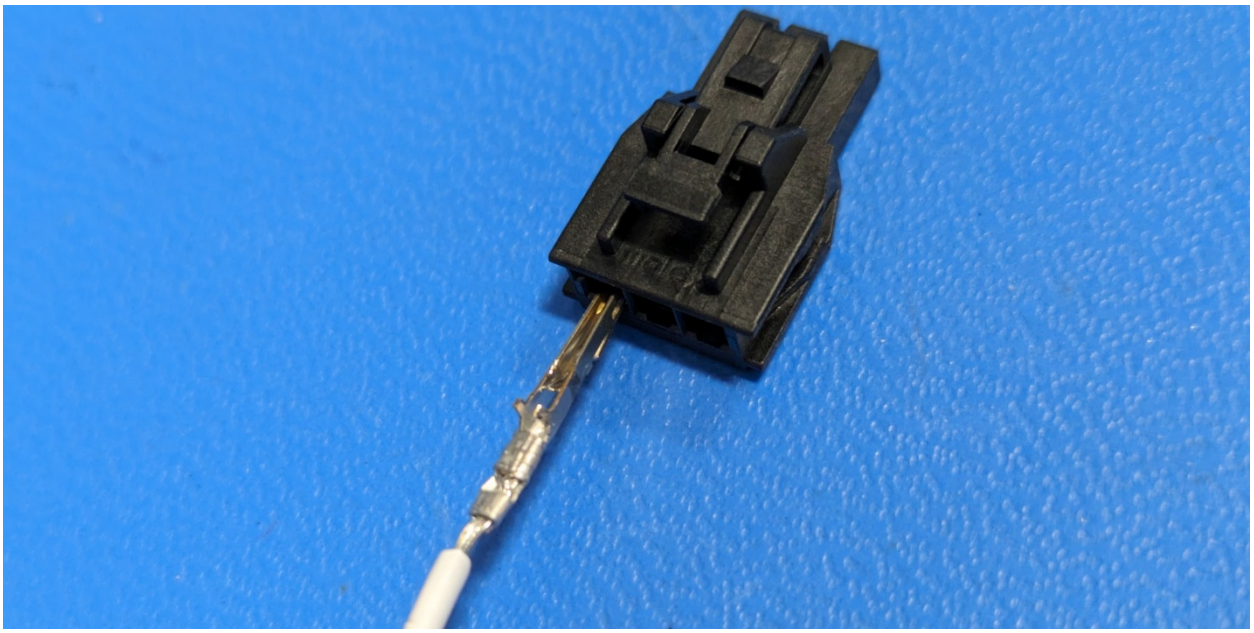


Figure 1.2-7 Preparing wire terminator sockets by hand; insert socket.

This completes one of the required connections for the PIXIE. The absolute minimum number of positions to prepare is four; two to power the PIXIE through the six-position connector, and two to communicate with the PIXIE through the three-position connector.

To power and communicate with the PIXIE PCB, simply plug the three-position and six-position connectors into their corresponding headers. **The PIXIE must be powered to flash (program) the device**.

1.3 Programming

In this section, instructions for how to retrieve the PIXIE's firmware source code, import it to the supported Integrated Development Environment (IDE), compile the code, flash it to the device, and run/debug the code are provided. This section assumes you have fully assembled your PIXIE and obtained the Texas Instruments (TI) MSP-FET Programmer device. The MSP-FET can be found at this link and through other reputable hardware suppliers:

https://www.ti.com/tool/MSP-FET

The PIXIE must be powered, and the MSP-FET Programmer must be connected to the programmer (JTAG) header on the PIXIE Printed Circuit Board (PCB).

1.3.1 Source Code

To obtain the PIXIE's firmware source code, along with all other files (including this User Guide!), visit the PIXIE's repository:

https://github.com/KylePark0/PIXIE/tree/main

Navigate to the "< > Code" button and click on it, then click on the "Download ZIP" option to acquire the files (see Figure 1.3-1). Alternatively, navigate to the "Firmware" path of the repository to obtain *only* the source code.

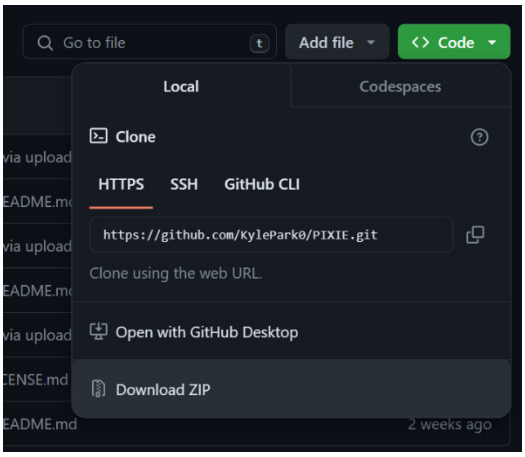


Figure 1.3-1 Downloading PIXIE Repository.

Extract the downloaded .zip file to a suitable location on your device. Open the Firmware subdirectory to view the "PIXIE_MSP430" folder. This folder is the root directory of the PIXIE's firmware source code project files. The PIXIE firmware project is written using the TI Code Composer Studio (CCS) IDE.

1.3.2 Obtaining IDE

The PIXIE firmware source code is formatted exclusively as a TI CCS project file, with the current stable version having been written using version 12.5.0. This version of the IDE can be obtained from the following link:

https://www.ti.com/tool/download/CCSTUDIO/12.5.0

The following instructions assume the use of the Windows 10 or Windows 11 operating system. These simple instructions can be easily adapted to other common operating systems.

After following the installation instructions for CCS, the first step is to import the "PIXIE_MSP430" folder as a CCS project. The first launch of CCS after a fresh installation will greet you as below in Figure 1.3-2.

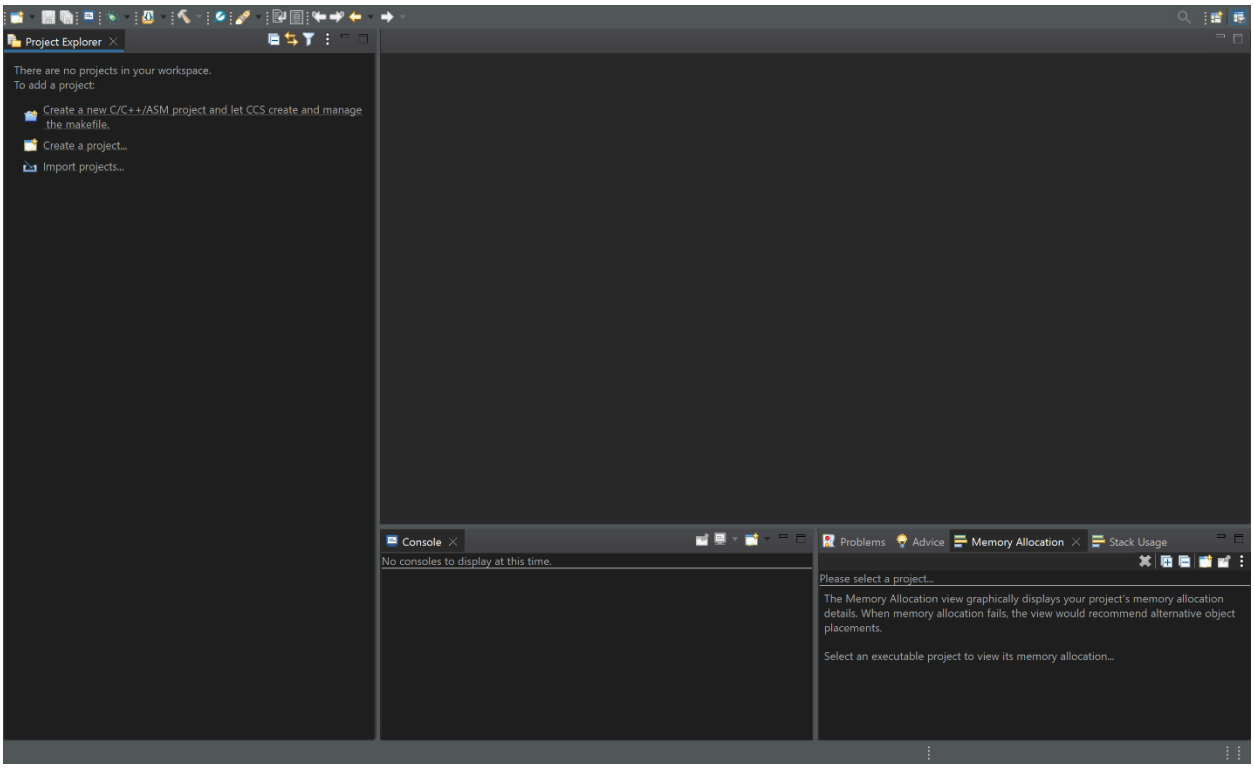


Figure 1.3-2 TI CCS v12.5.0 fresh install welcome screen.

Once imported, the "PIXIE_MSP430" project will populate the left pane with its source files. To import the project, navigate to the toolbar and click "File > Open Projects from File System...". This will automatically open the "Import Projects from File System or Archive" dialogue.

Next to the “Import Source” dropdown menu, find the “Directory…” button. Click “Directory…” and use your system browser to locate the “PIXIE_MSP430” project folder, as in Figure 1.3-3. Click the folder, then click the “Select Folder” button.

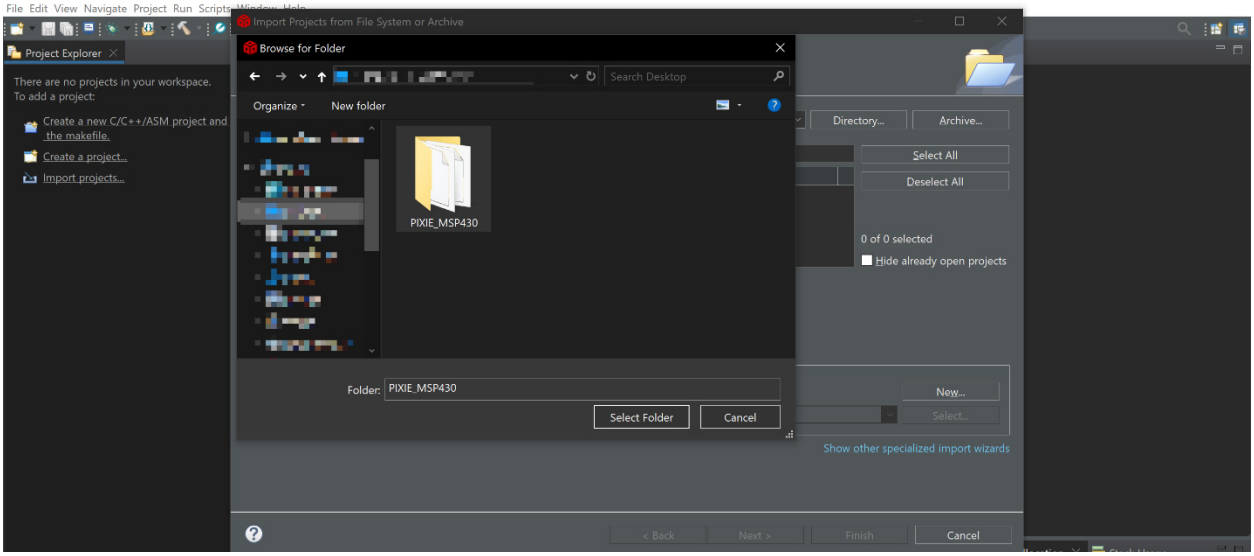


Figure 1.3-3 Importing the PIXIE_MSP430 folder as a TI CCS Project.

The folder is automatically recognized as a CCS project. Click the “Finish” button.

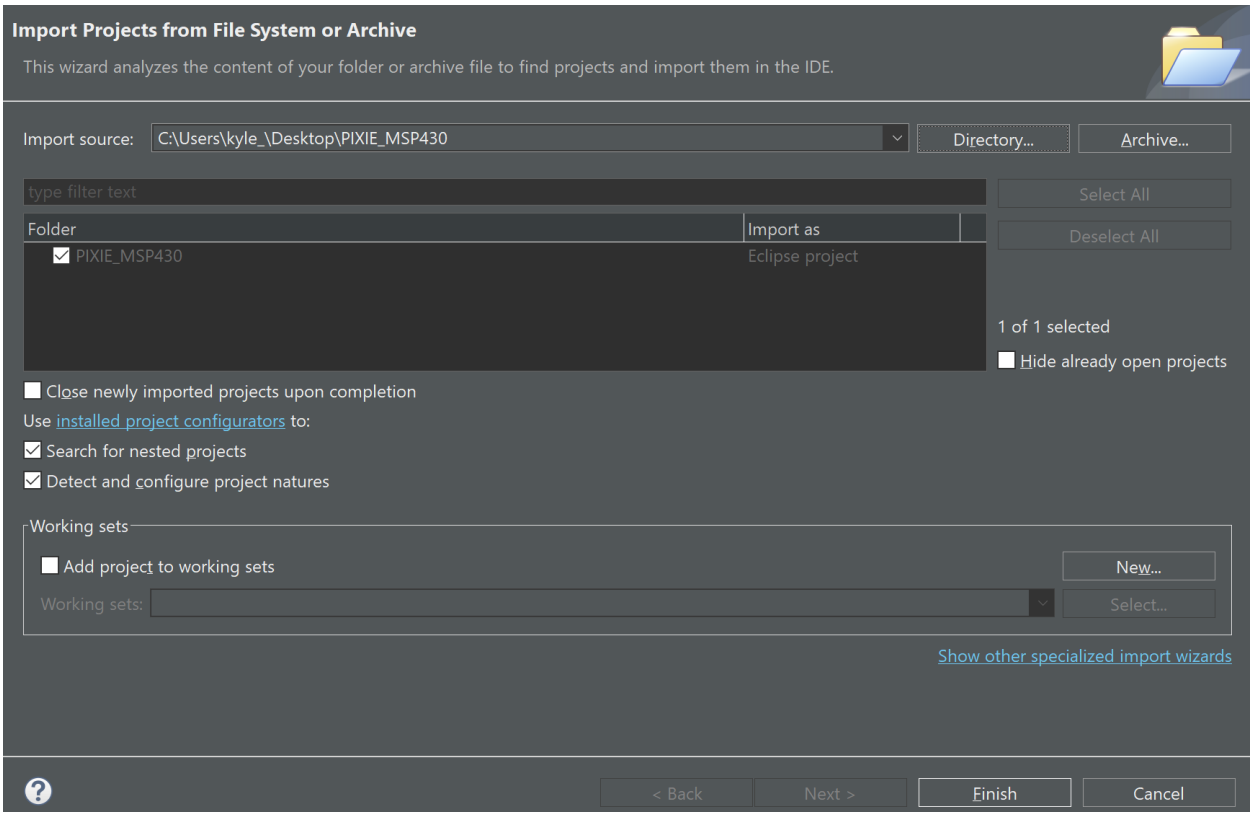


Figure 1.3-4 Finish importing the PIXIE_MSP430 folder.

The Project Explorer and Code editor panes should be automatically filled, though you may need to open “main.c” manually. Your CCS interface should now appear as below in Figure 1.3-5. The two most important source files, “main.c” and “pixie.h”, are located in the root project directory, whereas the project header files are located in the “inc” subdirectory, and the project source files are located in the “src” subdirectory.

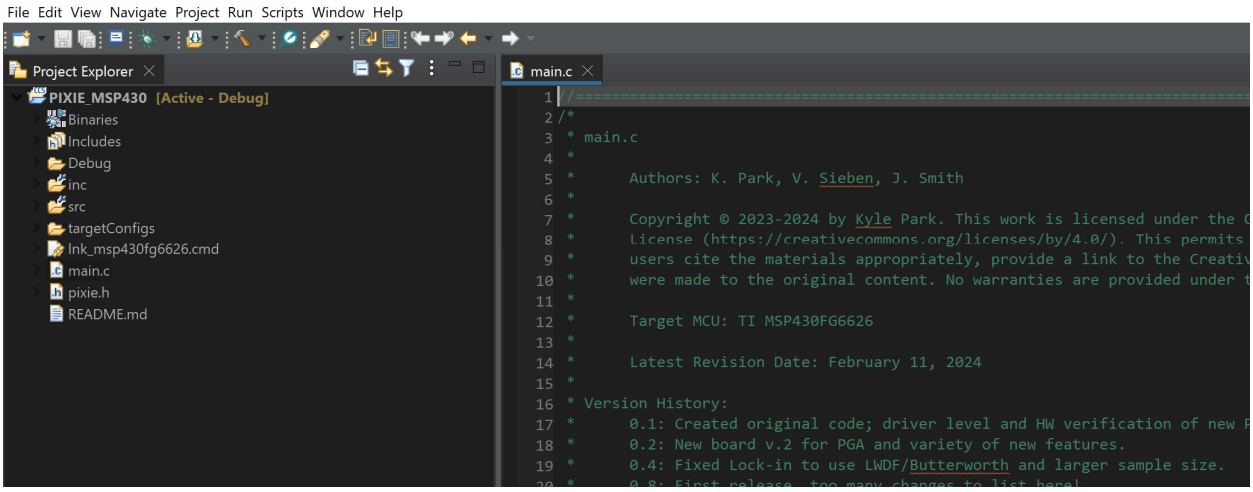


Figure 1.3-5 Successfully imported PIXIE_MSP430 into TI CCS.

To accelerate the understanding of the source code, each source file includes substantial commentary (line-by-line in most cases) to explain the function of the source code. For a system-level perspective of the firmware, consult the Firmware Design Note provided in Section 2.3 and the Commands system described in Section 1.5.2.

1.3.3 Compile and Run

To compile the firmware, navigate to the toolbar and click “Project > Build All”. The first compilation of the firmware will build every piece of source code and will take longer than subsequent builds. When finished, the Console will print a “**** Build Finished ****” message and the Advice pane will automatically populate with suggestions for code edits, as in Figure 1.3-6. These suggestions can be safely ignored.

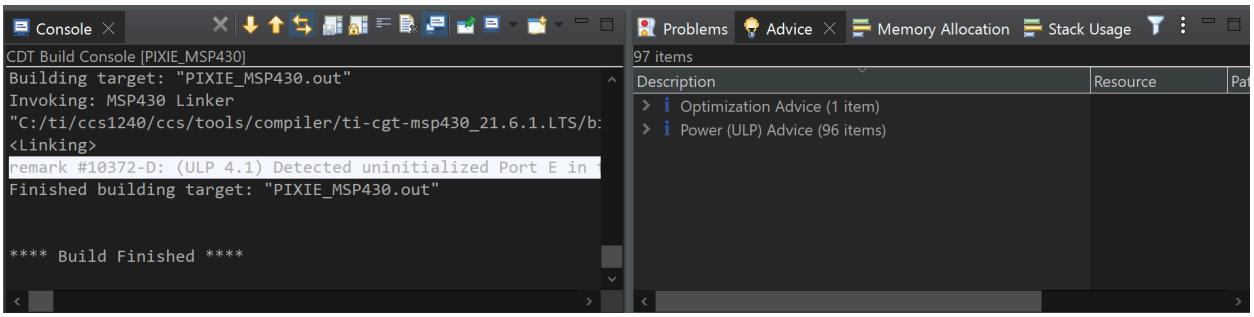


Figure 1.3-6 Messages from a successful build of the PIXIE firmware.

Upon successful compilation, the “Memory Allocation” tab can be used to inspect the current RAM and Flash memory usage of the firmware, as it appears in Figure 1.3-7 below.

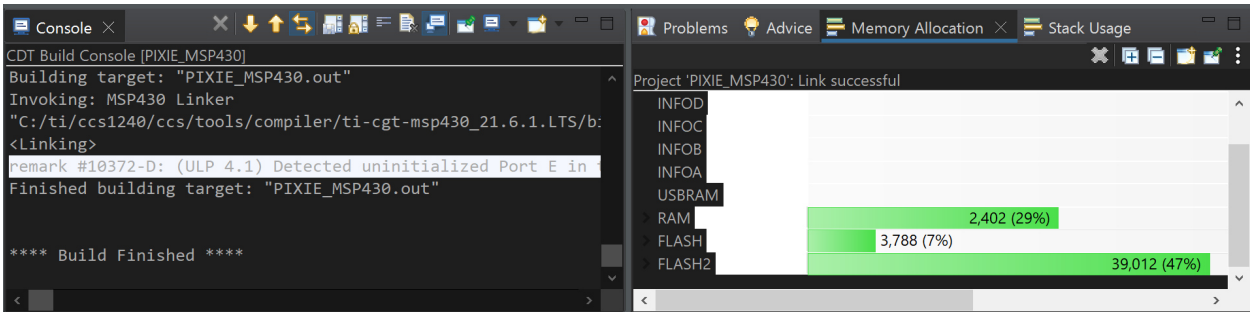


Figure 1.3-7 Successful build of the PIXIE firmware with Memory Allocation.

To run the program on the PIXIE PCB, the compiled firmware must be flashed onto the device. These instructions assume you have acquired the PIXIE PCB, the TI MSP-FET Programming device, and have assembled a cable to power the PIXIE as described in Section 1.2.2.

The TI MSP-FET Programmer connects to your computer via USB and communicates to the PIXIE PCB through the JTAG header, allowing the device to be programmed and debugged; that is, executed line-by-line. With TI CCS installed, the MSP-FET will

install its own driver through the USB connection. You may need to restart your computer if CCS fails to recognize the programmer immediately.

Connect the MSP-FET before powering the device. Once the MSP-FET is connected, you can supply power to the PIXIE. The connections should appear as below in Figure 1.3-8.

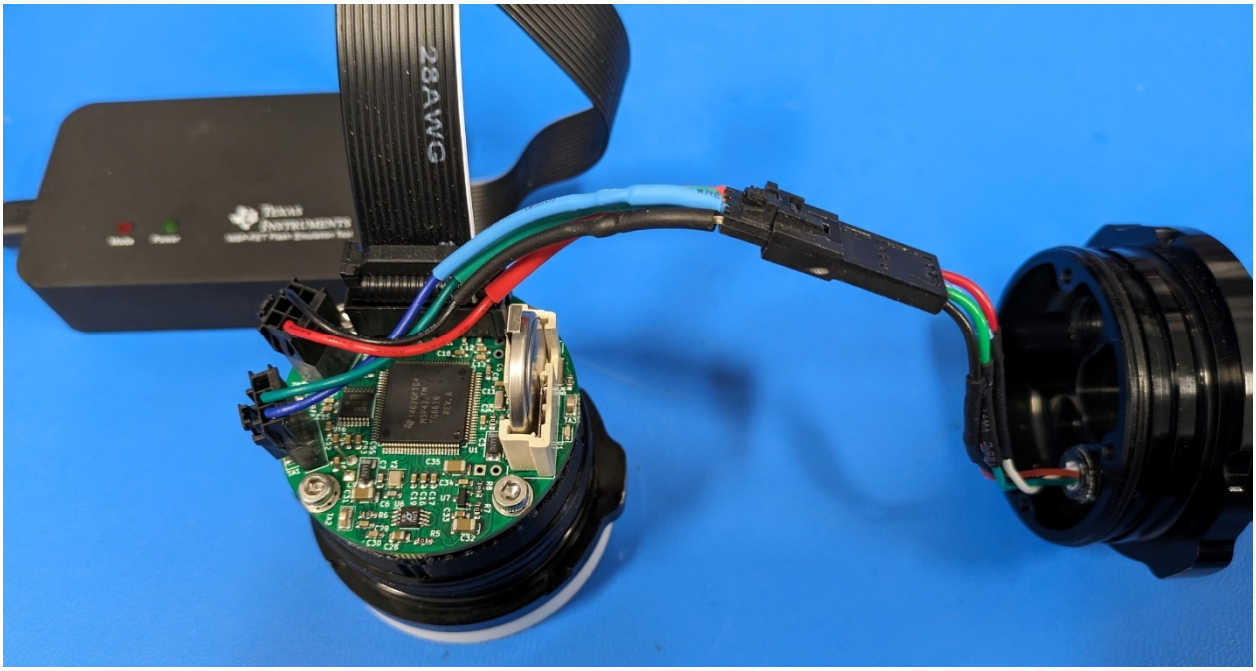


Figure 1.3-8 PIXIE PCB with Cable and MSP-FET Programmer connected.

Navigate to the toolbar and click “Run > Debug”. If the device is connected but not powered, the console will report the following error (see Figure 1.3-9).

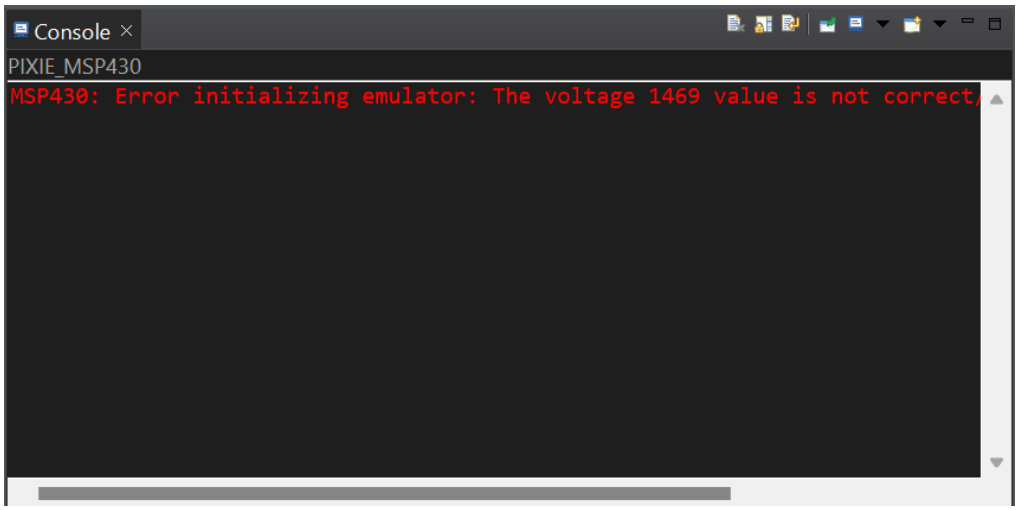


Figure 1.3-9 Error message returned on attempt to program with PIXIE unpowered.

When the MSP-FET establishes a connection, the user interface automatically swaps to Debug mode (see Figure 1.3-10). Wait for the configuration process to finish.

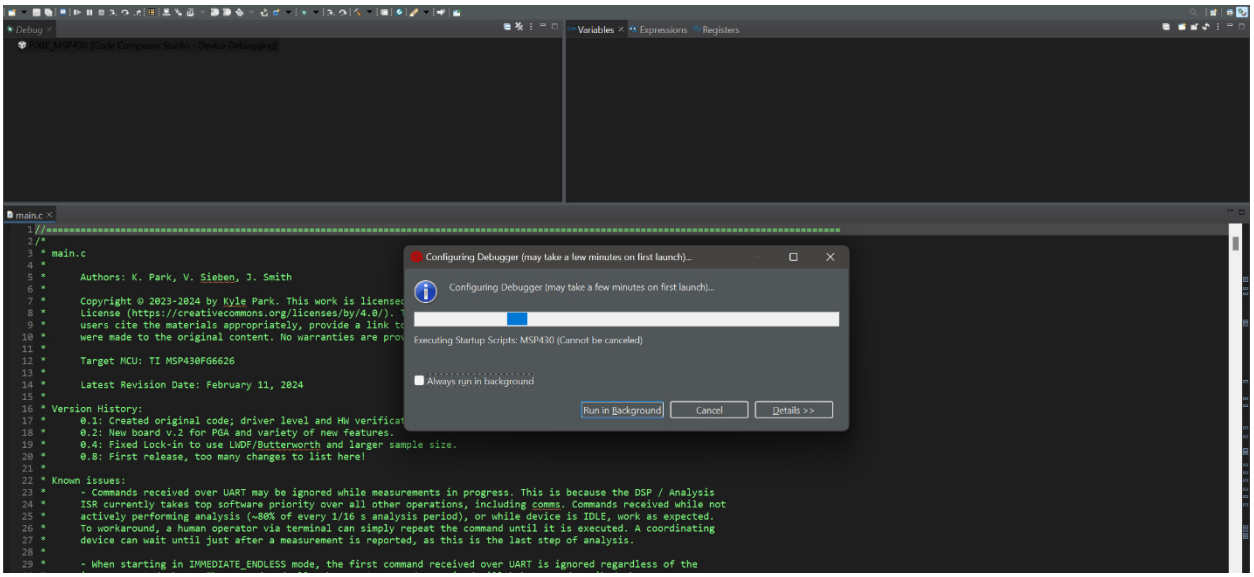


Figure 1.3-10 CCS Debug mode startup.

When configuration is finished, the PIXIE has been programmed. The CCS interface enters its proper debug mode, with code execution paused. Navigate to the toolbar and click “Run > Resume” to execute the PIXIE firmware code (see Figure 1.3-11).

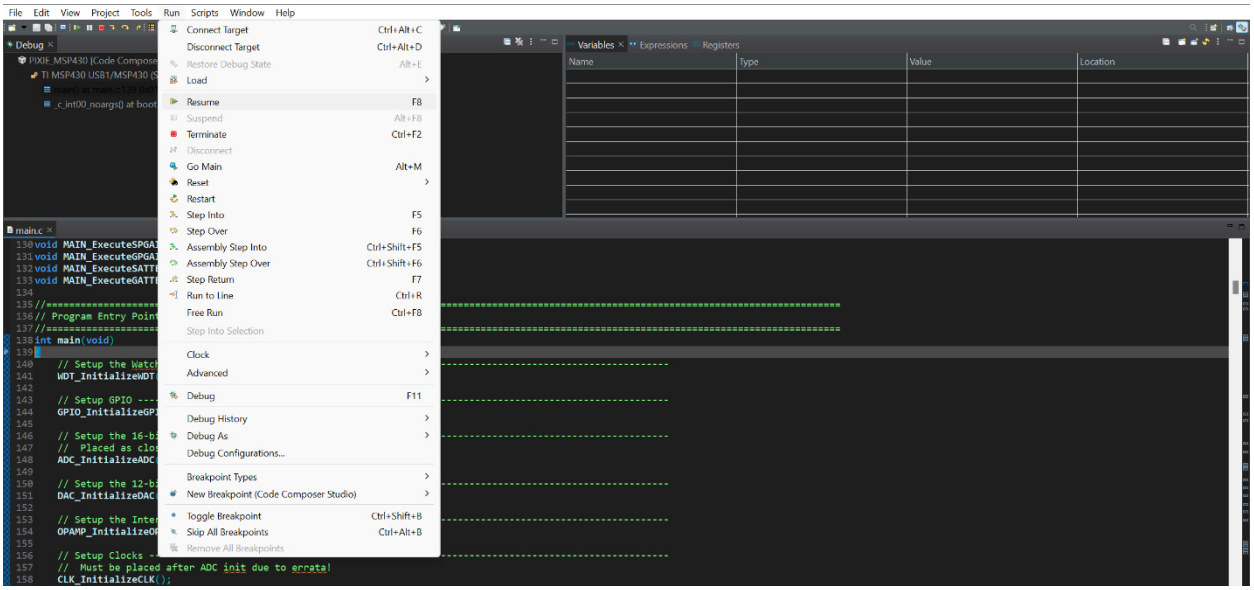


Figure 1.3-11 Begin code execution with CCS debug mode.

The debug mode can also be terminated, and the MSP-FET disconnected. The PIXIE can now be power-cycled to run independently of the MSP-FET programmer.

When the PIXIE has been successfully programmed and the firmware is allowed to run, the blue status indicator LED will toggle once every second, as in Figure 1.3-12. The PIXIE is now ready for configuration as described in the next section, Section 1.4, and for communication as described in the subsequent section, Section 1.5.

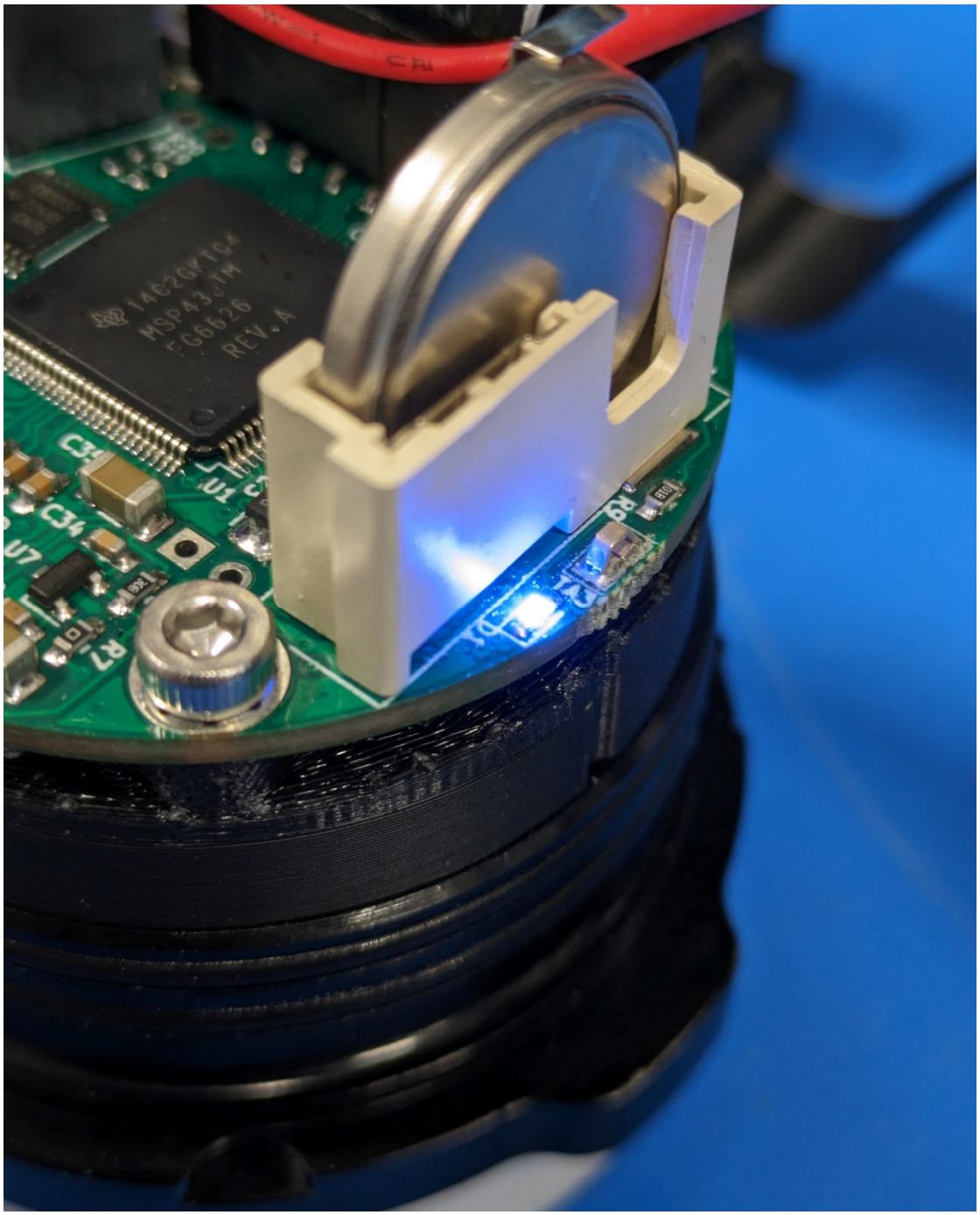


Figure 1.3-12 PIXIE successfully programmed and running, with status indicator LED lit.

1.4 Device Configuration

In this section, instructions for the minimum-necessary steps required to configure the PIXIE are provided. The default firmware configurations are presented with details on how to adjust them for your application. Configuring the device to run in Debug mode or in Immediate-Endless mode is demonstrated. Instructions for creating a Channel Sequence are provided.

You must configure your PIXIE based on the optics you have chosen and whether any deviations have been made from the default optics presented in Section 1.1.1. This section assumes you have assembled your PIXIE and are able to flash (program) the device.

1.4.1 Basic Configuration

The basic configuration of the PIXIE involves setting the identity and default gain settings for each of the device's installed channels, as well as identifying information for the device itself. Table 1.1-1 of Section 1.1.1 provides the fluorophores assigned to each channel by the default configuration. The source code, if left completely unaltered, matches the default configuration.

To properly configure the PIXIE and its channels, the channels' optics, electronics, and firmware must all be in agreement. The most difficult component to reconfigure once installed is the Excitation LED, as removing it from the PIXIE Printed Circuit Board (PCB) requires reflowing solder that may damage either the adjacent components or the PCB itself. Moving optical filters and changing firmware settings to match the placement of the Excitation LED(s) is easier and safer, though it is recommended that the default configuration is followed even when only one channel is being used (e.g. use Channel 2 for Rhodamine Water Tracer even if it is the only channel with LEDs/filters installed).

Most of the user-configurable elements in the PIXIE source code are found in the files "pixie.h" and "pixie.c" C-language files. For basic configurations, only "pixie.h" needs to be considered.

The first block of configurable elements in "pixie.h" contains identification text. You can name the individual PIXIE, assign the device a Hardware version number and Firmware version number, and assign the device a unique serial number. These assignments are all done with plain text and are not used by the device's internal

logic – they only exist to be echoed back to the user through a terminal or datalogger for identification purposes.

Figure 1.4-1 below depicts the symbols representing device name (DEVICE_NAME), hardware version (HW_VERSION), firmware version (FW_VERSION), and unique serial number (SERIAL_NUMBER). Each symbol must be set to a string with no more than 16 characters in length.

```
//=============================================================================================================
// Device-level Definitions
//=============================================================================================================
// Micro-controller platform name and clock speed. Shouldn't need to be changed.
#define PLATFORM                    "MSP430FG6626"
#define MCLOCK                      "18.432 MHz"

// Device Name, suggested. Can be changed if needed.
#define DEVICE_NAME                 "PIXIE"

// Hardware version number, suggested for personal use. Can be changed if needed.
#define HW_VERSION                  "0.80"

// Original Firmware version number, suggested for personal use. Can be changed if needed.
#define FW_VERSION                  "0.80"

// Serial number, suggested for personal use. Can be changed if needed.
#define SERIAL_NUMBER               "00000"
```

Figure 1.4-1 Configuring basic device elements in "pixie.h".

The next configuration deals with the startup message set by the PIXIE (see Section 1.5 for details). The PIXIE, by default, is set to a verbose startup. With a verbose startup, when the device is powered or is reset it will send a message containing the elements defined above, as well as the state of the devices' backup memory, current onboard time, and the date/time its current firmware was compiled on. Figure 1.4-2 below shows the device configured for a verbose startup.

```
//=============================================================================================================
// Firmware Compilation Flags
//=============================================================================================================
// __USE_VERBOSE_STARTUP ----------------------------------------------------------------------------
//  If this symbol is defined, the PIXIE will echo its name, compile time, current time, and battery backup
//  status to the terminal every time it powers on or resets. This is useful for debugging, but can be
//  a waste of space when connected to RS-232 data-logging devices.

#define __USE_VERBOSE_STARTUP
```

Figure 1.4-2 Configuring "pixie.h" for the PIXIE to use a verbose startup.

When the symbol __USE_VERBOSE_STARTUP is defined, the device will use a verbose startup. This is useful when debugging a newly assembled PIXIE, or in use-cases where the device will run uninterrupted for long periods of time.

When the symbol __USE_VERBOSE_STARTUP is undefined, by being commented out as in Figure 1.4-3, the PIXIE will not send any messages at startup. This is useful in

use-cases where the PIXIE may be power cycled frequently; turning off the verbose startup message will reduce wasted space in the terminal or datalogger.

```
//=============================================================================================================
// Firmware Compilation Flags
//=============================================================================================================
// __USE_VERBOSE_STARTUP ------------------------------------------------------------------------------------
//  If this symbol is defined, the PIXIE will echo its name, compile time, current time, and battery backup
//  status to the terminal every time it powers on or resets. This is useful for debugging, but can be
//  a waste of space when connected to RS-232 data-logging devices.

//#define __USE_VERBOSE_STARTUP
```

Figure 1.4-3 Configuring "pixie.h" to disable the PIXIE's verbose startup.

When configuring the basic operating parameters of the PIXIE's channels, the first setting applies to all four channels at once. The EXCLUSIONS symbol, as seen in Figure 1.4-4, allows the user to define the number of cycles or samples of raw data (at 16 samples per second) that are excluded from the start of each measurement. For example, if EXCLUSIONS is set to 8 and the PIXIE begins a measurement with a duration of 10 seconds, the PIXIE will begin a measurement with a total duration of 10.5 seconds, but it will only report the datapoints after the first 8 have been excluded. This is intended to allow the digital filtering transients of the device to fully stabilize before reporting data, in the case that data storage is scarce. By default, the EXCLUSIONS are set to 0, which disables this behavior entirely.

```
//=============================================================================================================
// Channel Configuration
//=============================================================================================================

// Exclusions -----------------------------------------------------------------------------------------------
//  If set to N > 0, the analysis extends the duration of the RUN command by N cycles and does not report
//  that data. This can be useful to suppress start-up transients from the data-stream and save space
//  in terminals or RS-232 data-loggers.

#define EXCLUSIONS                      (0)
```

Figure 1.4-4 Configuring "pixie.h" to set (or disable) the number of excluded data points.

Each channel has four individual operating parameters. The first parameter is an identifier that does not interact with the device logic, it is simply returned with commands to report device status for tracking purposes. The remaining parameters establish the default amplifier gain, default preamplifier gain, and default LED drive-signal attenuation level. These parameters are loaded into RAM on startup if the backup memory has been cleared or is otherwise unavailable. If the backup memory is intact, the channels will use the last gain settings saved to the device (see Section 1.5.2 and Section 1.5.3 for details). The block of code dedicated to the individual channel settings is depicted in Figure 1.4-5.

```
// Default Channel Specifications ------------------------------------------------------------------
//  These configurations are used set the default name, gain, pregain, and attenuation level for each of
//  the four channels in the device. Adjust these to fit your PIXIE!
//
//  CAUTION: At attenuation level 0, the default PIXIE hardware is designed to supply a peak current of
//  ~45 mA to the active channel LED. If the LED of a given channel cannot tolerate this peak current, the
//  default attenuation level should be set to at least 1.
//
//  Valid Pregains      : 0, 1, 2, 5, 10, 20, 50, 100
//  Valid Gains         : 1, 2, 4, 8, 16
//  Valid Attenuations  : 0, 1, 2, 3

//   Channel 1 Default Specs
#define CHANNEL_1_TYPEID               (CHANNEL_TYPEID_PHYCOCYANIN)
#define CHANNEL_1_PREGAIN              (100)
#define CHANNEL_1_GAIN                 (16)
#define CHANNEL_1_ATTENUATION          (0)

//   Channel 2 Default Specs
#define CHANNEL_2_TYPEID               (CHANNEL_TYPEID_RHODAMINE)
#define CHANNEL_2_PREGAIN              (10)
#define CHANNEL_2_GAIN                 (8)
#define CHANNEL_2_ATTENUATION          (0)

//   Channel 3 Default Specs
#define CHANNEL_3_TYPEID               (CHANNEL_TYPEID_CHLOROPHYLL)
#define CHANNEL_3_PREGAIN              (10)
#define CHANNEL_3_GAIN                 (8)
#define CHANNEL_3_ATTENUATION          (0)

//   Channel 4 Default Specs
#define CHANNEL_4_TYPEID               (CHANNEL_TYPEID_FDOM)
#define CHANNEL_4_PREGAIN              (100)
#define CHANNEL_4_GAIN                 (16)
#define CHANNEL_4_ATTENUATION          (1)
```

Figure 1.4-5 Configuring the default channel gain and attenuation settings in "pixie.h".

The amplifier gain and preamplifier gain settings are constrained to hardware-specific values. The _GAIN symbols must be set to one of the values in {1, 2, 4, 8, 16}, whereas the _PREGAIN symbols (preamplifier gain) must be set to one of the values in {0, 1, 2, 5, 10, 20, 50, 100}. The _ATTENUATION symbols must be set to one of the values in {0, 1, 2, 3}. The gain and attenuation settings are explained in detail in Section 1.5.3.

The settings presented in the unaltered source code represent reasonable values for the default configuration Excitation LEDs. The settings for Channel 2 show in Figure 1.4-5 are identical to those used to produce the Calibration results in Section 1.6. The attenuation level of Channel 4 is set to 1 because the default-configuration LED for UV fluorometry, the MT3650N3-UV, cannot handle the 50 mA unattenuated peak drive current. A setting of 1 reduces the peak drive current by a factor of 50%.

1.4.2 Startup Mode

The PIXIE has two modes it may operate in at startup, or following a device reset. These are Debug mode and Immediate Endless mode. The technical details of these modes are described in Section 2.3. Here, only the instructions for setting the startup mode are provided.

By default, compiling and running the PIXIE firmware will start the device in Debug mode. The PIXIE sends some startup information through the RS-232 connection and waits for commands received through the RS-232 connection. The details of communicating with the PIXIE via RS-232 and the list of commands are provided in the following section, Section 1.5. Debug mode is most useful for testing and debugging the PIXIE after assembly, and for collecting data from mock-measurements and calibration.

The file “pixie.h” contains the code needed to set the startup mode. The unaltered source code for starting in Debug mode is depicted below in Figure 1.4-6.

```
//=================================================================================================================
// Firmware Compilation Flags
//=================================================================================================================
// __USE_VERBOSE_STARTUP ---------------------------------------------------------------------------------------
//  If this symbol is defined, the PIXIE will echo its name, compile time, current time, and battery backup
//  status to the terminal every time it powers on or resets. This is useful for debugging, but can be
//  a waste of space when connected to RS-232 data-logging devices.

#define __USE_VERBOSE_STARTUP

// __IMMEDIATE_ENDLESS -----------------------------------------------------------------------------------------
//  If this symbol is defined, the PIXIE will begin measuring immediately on startup and will do so using
//  the ENDLESS_RUN duration. It can only be stopped by receiving a STOP command via UART/RS-232.

//#define __IMMEDIATE_ENDLESS

// __IMMEDIATE_ENDLESS_MODE ------------------------------------------------------------------------------------
//  If the __IMMEDIATE_ENDLESS is defined, this symbol MUST also be defined. This constant tells the PIXIE
//  which channel to RUN with; set to 1...4 to run from channel 1...4, or set to 0 to run the defined
//  channel sequence.

//#define __IMMEDIATE_ENDLESS_MODE    (0)
```

Figure 1.4-6 Configuring "pixie.h" to start the PIXIE in Debug mode.

The symbol “__IMMEDIATE_ENDLESS” is set as a comment by the preceding “//” comment symbol and is therefore skipped at compile time. This causes the firmware to compile in Debug mode.

If configured to start in Immediate Endless mode, the PIXIE will begin to measure and report data immediately after startup or reset and will do so for an indefinite duration. The PIXIE can be interrupted with commands sent via RS-232, at which

point the PIXIE will idle as if it had started in Debug mode. Immediate Endless mode is most useful when deploying the PIXIE, especially when its power and commands are coordinated by an external datalogger.

To start the PIXIE in Immediate Endless mode, the "__IMMEDIATE_ENDLESS" symbol must be defined. To do this, simply remove the preceding "//" comment symbol in "pixie.h". The symbol "__IMMEDIATE_ENDLESS_MODE" must also be set equal to a number from 0 to 4, or the firmware will fail to compile. The device will measure with the channel corresponding to the chosen number; Channel 1 for a choice of 1, Channel 2 for 2, and so on. If set to 0, the PIXIE will start by running its Channel Sequence. Channel sequences are described in the section immediately following this one, Section 1.4.3.

For an example of starting the PIXIE in Immediate Endless mode using Channel 2, see Figure 1.4-7 below.

```
//=================================================================================================================
// Firmware Compilation Flags
//=================================================================================================================
// __USE_VERBOSE_STARTUP ------------------------------------------------------------------------------------------
//  If this symbol is defined, the PIXIE will echo its name, compile time, current time, and battery backup
//  status to the terminal every time it powers on or resets. This is useful for debugging, but can be
//  a waste of space when connected to RS-232 data-logging devices.

#define __USE_VERBOSE_STARTUP

// __IMMEDIATE_ENDLESS --------------------------------------------------------------------------------------------
//  If this symbol is defined, the PIXIE will begin measuring immediately on startup and will do so using
//  the ENDLESS_RUN duration. It can only be stopped by receiving a STOP command via UART/RS-232.

#define __IMMEDIATE_ENDLESS

// __IMMEDIATE_ENDLESS_MODE ---------------------------------------------------------------------------------------
//  If the __IMMEDIATE_ENDLESS is defined, this symbol MUST also be defined. This constant tells the PIXIE
//  which channel to RUN with; set to 1...4 to run from channel 1...4, or set to 0 to run the defined
//  channel sequence.

#define __IMMEDIATE_ENDLESS_MODE    (2)
```

Figure 1.4-7 Configuring "pixie.h" to start the PIXIE in Immediate Endless mode.

1.4.3 Channel Sequences

The PIXIE is only capable of driving one Excitation LED at a time, and only capable of measuring with one Detector circuit at a time. To make use of all four channels of the PIXIE sequentially, there are two primary options.

For the first option, the PIXIE can be run in Debug mode and the coordinating device (datalogger, computer terminal, etc.) can issue explicit run commands to measure with each channel for a certain amount of time. This can be useful for deployments where the PIXIE will be inactive and unpowered for long periods of time, or when conditional measurements need to be made using higher level decision logic than the PIXIE can be programmed with. This sequential measurement option is controlled externally and is therefore up to the user to design and implement.

For the second option, the PIXIE has a firmware-based channel sequence that is controlled internally. The internal logic of the PIXIE treats the channel sequence as Channel 0. Commands to measure with Channel 0 will trigger the channel sequence instead of reporting an error. When starting the PIXIE in Immediate Endless mode, setting the "__IMMEDIATE_ENDLESS_MODE" symbol to 0 will trigger the channel sequence and repeat it until commanded to stop. The internal channel sequence allows the PIXIE to cycle measurements through a user-defined sequence of channel and gain settings for a user-defined amount of time per step.

The channel sequence defaults to 4 steps, repeating until the total measurement duration has elapsed or until the measurement is commanded to stop. The number of steps in the channel sequence is set using the "CHANNEL_SEQUENCE_COUNT" symbol in "pixie.h". An example of setting the number of steps to 4 is shown below in Figure 1.4-8.

```
// Channel Sequence ------------------------------------------------------------------------------
//  Configure this first, to set the number of steps in your channel sequence, or set to 0 to disable
//  channel sequencing.
//  To program the channel, gain, pregain, attenuation, and number of cycles per for each step in the
//  sequence, continue to "pixie.c".

#define CHANNEL_SEQUENCE_COUNT          (4)
```

Figure 1.4-8 Configuring "pixie.h" to have a 4-step channel sequence.

Once the number of steps in the channel sequence has been defined, the properties of each step (channel number, gain settings, duration) can be configured in the source code file "pixie.c". Specifically, each step in the channel sequence has a setting for ID (nominal channel number, e.g. Channel 1 has an ID of 1), Pregain

(preamplifier gain), Gain (amplifier gain), Attenuation (attenuation level), and SeqCycles (duration of step). The SeqCycles setting is counted in sixteenths of a second, so a setting of 16 corresponds to a step duration of one second. The settings must be made using valid C-array syntax. Each setting must be made inside of curly "{ }" braces, separated by commas, with the line terminated by a semicolon. The settings entered into the array must be in the order you intend the sequence to run through, and the number of settings must be exactly equal to "CHANNEL_SEQUENCE_COUNT".

In the default settings shown below, with Figure 1.4-9, a channel sequence that measures with Channel 2, Channel 2 again, Channel 1, and Channel 3 before repeating is configured. These settings demonstrate that channels can be repeated (Channel 2), channels can be excluded (Channel 4), and steps in the sequence can be run for different durations.

```
//=============================================================================================================================================
// Channel Sequence
//=============================================================================================================================================
//  Configure each step of the channel sequence by setting, in order, the ID values, the pregains, etc. For example, the default
//  configuration's first step uses:
//          Channel 2, with Pregain 10, Gain 8, Attenuation level 0, for 32 cycles (2 seconds),
//
//  then proceeds to the second step with:
//          Channel 2, with Pregain 50, Gain 16, Attenuation level 1, for 32 cycles (2 seconds).
//
//  You must ensure each array has many entries as the number CHANNEL_SEQUENCE_COUNT is set to. Any combination of valid combination
//  of channel ID, gain, attenuation, etc. can be used, channels may be repeated with different gain settings, channels can be excluded.
//
//  NOTE: Channel IDs use their NOMINAL values: Channel 1 has an ID value of 1. They do NOT use C-array style IDs.
//
//  Valid Channel IDs   : 1, 2, 3, 4
//  Valid Pregains      : 0, 1, 2, 5, 10, 20, 50, 100
//  Valid Gains         : 1, 2, 4, 8, 16
//  Valid Attenuations  : 0, 1, 2, 3

#if CHANNEL_SEQUENCE_COUNT > 0

const uint16_t  PIXIE_SequenceID[CHANNEL_SEQUENCE_COUNT]            = {2,   2,  1,  3};

const uint16_t  PIXIE_SequencePregain[CHANNEL_SEQUENCE_COUNT]       = {10, 50, 10, 10};

const uint16_t  PIXIE_SequenceGain[CHANNEL_SEQUENCE_COUNT]          = {8,  16,  8,  8};

const uint16_t  PIXIE_SequenceAttenuation[CHANNEL_SEQUENCE_COUNT]   = {0,   1,  0,  0};

const uint16_t  PIXIE_SequenceSeqCycles[CHANNEL_SEQUENCE_COUNT]     = {32, 32, 16, 16};
```

Figure 1.4-9 Configuring the details of a 4-step channel sequence in "pixie.c".

1.5 Operation

In this section, instructions for how to operate the PIXIE are provided. Basic instructions for establishing a communications link with the PIXIE are provided. A list of commands for communicating with the PIXIE is provided and detailed. Sample measurement data is provided, as well as sample calibration results.

1.5.1 Communication

You can connect the PIXIE to any device that is RS-232 compatible, instructions for establishing a connection to a Windows PC terminal are provided here for reference. The simple, open-source terminal program "PuTTY" is recommended, and can be obtained from the following permanent link:

https://www.chiark.greenend.org.uk/~sgtatham/putty/releases/0.80.htmlb.

Assuming the PIXIE cable is assembled as described in Section 1.2.2, and connect the DB9 connector to your PC's RS-232 port but do not turn on the power supply yet. If the PC does not have a dedicated RS-232 port, you will need a separate USB to RS-232 cable.

The device should appear in Device Manager as a generic Communications Port or with a name corresponding to your USB to RS-232 converter. Figure 1.5-1 below contains an example Device Manager window, in which the connected PIXIE appears on the COM10 port.

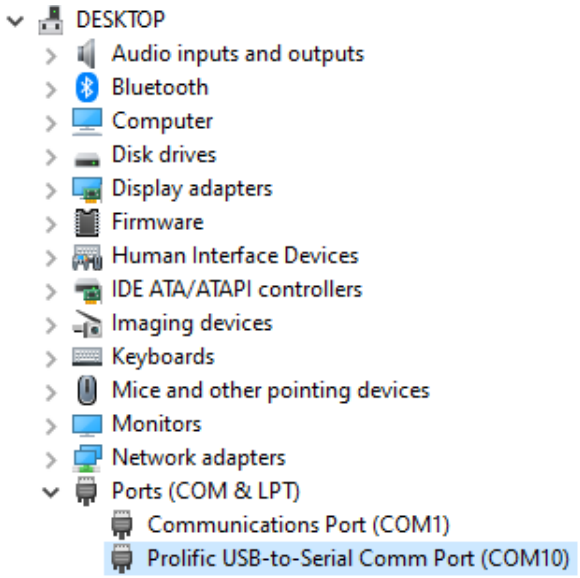


Figure 1.5-1. Example of Device Manager, with PIXIE connected to the COM10 port.

Depending on the Port's default settings, the connection may work directly, or appear to respond with junk characters in the terminal. In either case, the Port should be configured with settings that match the PIXIE's expected RS-232 specifications.

To configure the Port's settings, right click on the corresponding device (COM10 in this example) and click on the Properties option. From the dialog box that opens, click on the tab marked

"Port Settings". To configure the port, choose 115200 bits per second (115200 baud), 8 data bits, no parity, 1 stop bit, with no flow control (see Figure 1.5-2).

To configure PuTTY and open the terminal, open the program and navigate to the "Serial" page under the "Connection" category. Enter the name of the port in the top field (COM10 in this case, adjust your input to match), and enter the same settings as before into their respective fields: 115200 baud, 8 data bits, no parity, 1 stop bit, with no flow control. Once the correct settings are entered, click the "Open" button at the bottom of the dialog box to open the terminal (see Figure 1.5-3).

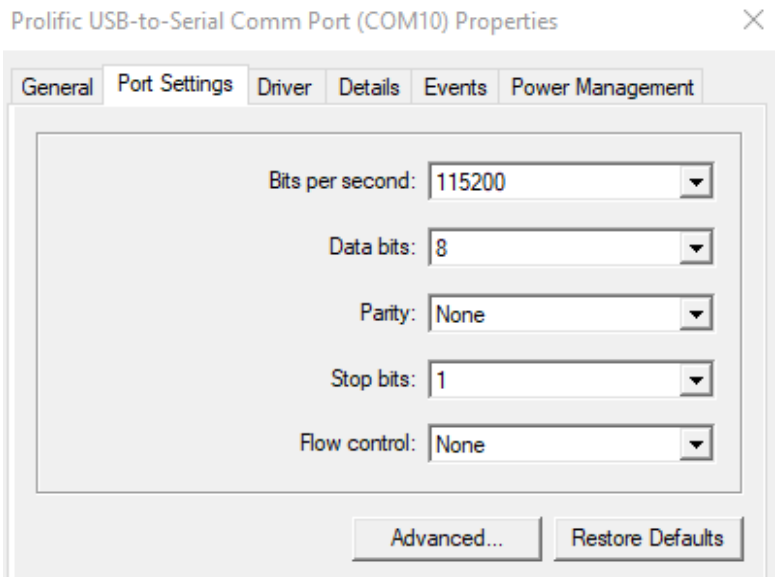


Figure 1.5-2. Device manager dialog window with PIXIE RS-232 Settings.

If the COM port and PuTTY settings have been entered properly, an empty terminal window will open. Since the PIXIE should be unpowered at this point; no response should be result from any commands sent by typing into the terminal.

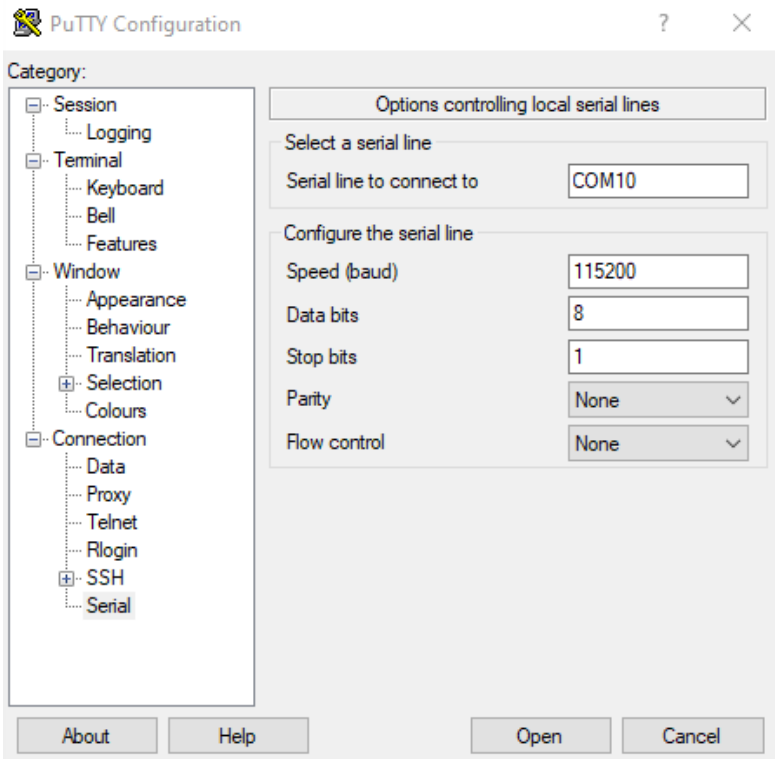


Figure 1.5-3. PuTTY configuration dialog window with PIXIE RS-232 Settings.

Once the COM port has been opened and the terminal window has appeared, you should turn on the power to the PIXIE. If the device has been programmed to run in its default Debug mode, an information header should

echo to the terminal. This header contains the name of the device (PIXIE), along with version and serial numbers, the date on which the current firmware version was compiled, whether battery-backed data has been restored, and the current time of the device's onboard clock in ISO 8601 UTC format, followed by a prompt. If the device has been programmed to run in Immediate Endless mode, data will begin to stream to the terminal after a short delay.

Figure 1.5-4 below contains a sample terminal window in which communication with the PIXIE in Debug mode has been successfully established.

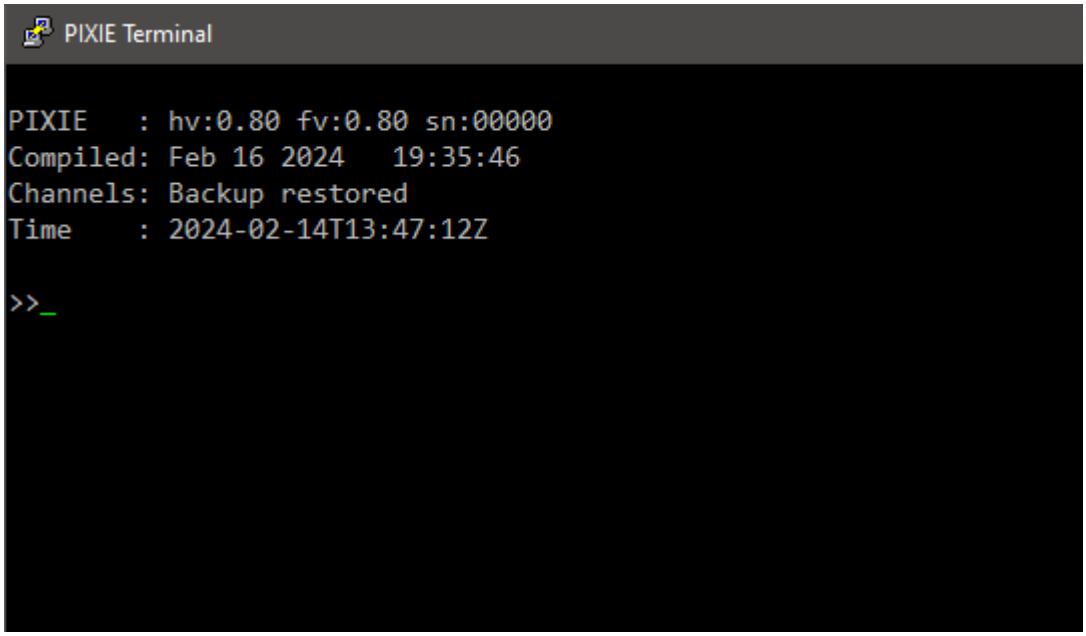


Figure 1.5-4. Terminal window with PIXIE communication established successfully.

1.5.2 Commands

While powered on, the PIXIE has four device states that dictate how it responds to commands received via RS-232. The four states are: IDLE, FLOOD, RUN, and SPOOF. Two of these, FLOOD and SPOOF, are intended for debugging purposes only. When the device is powered on (or resets) into Debug mode, the initial state is IDLE. When the device is powered on (or resets) into Immediate Endless mode, the initial state is RUN. A summary of the PIXIE's state machine is given below in Figure 1.5-5.

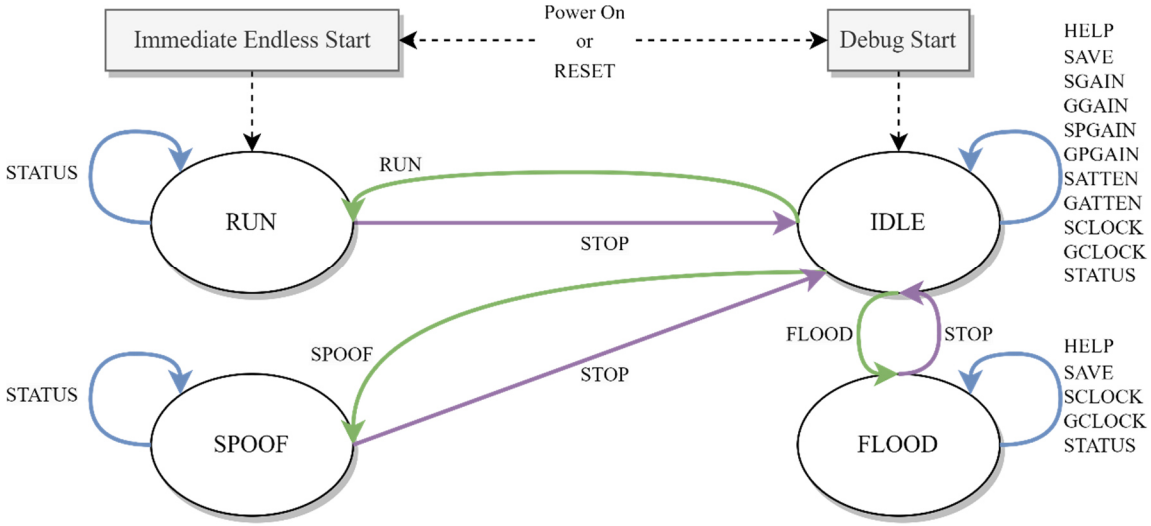


Figure 1.5-5. PIXIE Device state machine, showing transitions and valid commands.

When IDLE, the device responds to commands directly. The user can check or adjust channel settings, set or get the current time, begin measurements, or reset the device. When RUN, the device will reject all commands besides those to stop or reset the device so as not to divert computing resources away from capturing and processing measurement data.

The FLOOD state is entered with the FLOOD command, which mimics the IDLE state but with one selected LED turned on. This state blocks attempts to perform measurements unless the state is first returned to IDLE through the STOP or RESET commands. This can be used to test whether channel filters and LEDs are correctly installed.

The SPOOF state is entered with the SPOOF command, which mimics the RUN state but disables all the channel LEDs. Therefore, the measurements reported via RS-232 during this state are due to external light sources and internal electrical (i.e.:

noise, interference) sources. Inexplicable measurements and instability in this state can indicate damage to the Printed Circuit Board (PCB).

Commands sent via RS-232 are case-insensitive and may contain comma-separated arguments (data fields). Commands are terminated with a newline ('\n') or carriage return ('\r') characters. For example, STOP requires no data and can be executed at any state, whereas FLOOD requires one argument, the selected channel, and can only be executed while the device state is IDLE. The following table, Table 1.5-1, lists the PIXIE's commands, showing arguments and example usage.

Table 1.5-1. Detailed list of PIXIE RS-232 Commands.

Command	Arguments	Description	Example(s)
RUN	X Channel, Y Duration	Enter RUN state, measure channel X for Y seconds. If X is zero, runs the channel sequence for Y seconds. If Y is exactly 3678, the RUN duration is infinite.	run,2,30 RUN,0,600 Run,1,3678 run,0,3678
SPOOF	X Channel, Y Duration	Enter SPOOF state, measure channel X for Y seconds, without turning on any LEDs. SPOOF does not use channel sequences. If Y is exactly 3678, the SPOOF duration is infinite.	spoof,2,30 SPOOF,1,3678
HELP		Not currently implemented. A placeholder for on-board command help.	help
STOP		Stops any current action and returns device to IDLE state. HALT is a synonym command. Works in any state.	STOP halt
SAVE		Commits all current gain, pre-gain, and attenuation settings to battery backup. This will be reloaded on power-up / RESET.	save
STATUS	None, or X Channel	If sent with no arguments, the current device header and time are echoed. If X is provided, the current gain settings of channel X are provided instead. If measuring, instead warns the user of the current device state.	Status STATUS,3
FLOOD	X Channel	Enter FLOOD state, activate channel X's LED at its current attenuation setting. No measurement occurs, state persists until STOP or RESET.	FLOOD,1
RESET		Triggers a software reset of the PIXIE. Device restarts as if just powered on. Works in any state.	reset
SGAIN	X Channel, Y Gain	Sets the gain of channel X to value Y. Y must be a valid gain setting (1, 2, 4, 8, or 16).	Sgain,2,16 SGAIN,1,1
GGAIN	X Channel	Fetches and echoes the gain setting of channel X.	GGAIN,1
SPGAIN	X Channel, Y Gain	Sets the pre-gain of channel X to value Y. Y must be a valid pre-gain setting (0, 1, 2, 5, 10, 20, 50, or 100).	spgain,2,50
GPGAIN	X Channel	Fetches and echoes the pre-gain setting of channel X.	GpGAIN,4

SATTEN	X Channel, Y Level	Sets the attenuation level of channel X to value Y. Y must be a valid attenuation level (0, 1, 2, or 3).	satten,2,0 SATTEN,4,3
GATTEN	X Channel	Fetches and echoes the attenuation level of channel X.	gATTEN,1
SCLOCK	Time	Sets the PIXIE's onboard clock to Time. Time must be formatted in the ISO 8601 UTC time code format, or "YYYY-MM-DDThh:mm:ssZ". This time is automatically retained if a backup battery is installed.	sclock,2021-02-23T12:34:56Z
GCLOCK		Fetches and echoes the current onboard time, in ISO 8601 UTC format.	gclock

Sample output of SCLOCK, then GCLOCK, followed by a RESET command is given below in Figure 1.5-6.

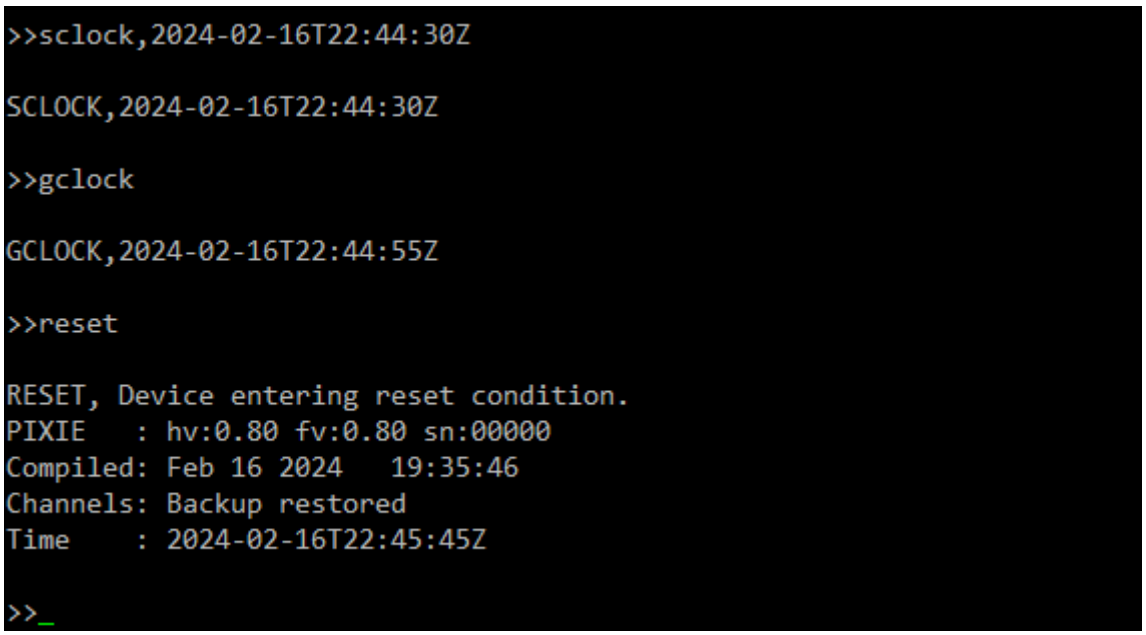


Figure 1.5-6. Sample output from setting time with SCLOCK, retrieving time with GCLOCK 25 seconds later, then resetting.

1.5.3 Measurement

Before making measurements, the gain settings for each channel you intend to use should be configured. This can either be done at the firmware level through programming, or through RS-232 commands. If configured via RS-232, these settings persist and take priority over firmware-level defaults unless the battery backup fails. You can check the settings of any channel by issuing a STATUS command, an example of which is presented in Figure 1.5-7. For this release of the PIXIE, the settings of a channel sequence can only be altered at the firmware level; they are unaffected by the settings made via RS-232 commands.

```
STATUS,
PIXIE : hv:0.80 fv:0.80 sn:00000
Compiled: Feb 16 2024   19:35:46
State: IDLE

>>status,2

STATUS, Channel 2:
Target:         RhodamineWT
Attenuation:    0
Gain:           8
Pregain:        10
```

Figure 1.5-7. STATUS command output for Channel 2 using default configurations.

The channel Gain and Pre-gain are set the total amplification of the fluorescence signal after it has been received by the photodiode. The total voltage gain is simply these two settings multiplied together.

The Pre-gain (preamplifier gain) configures the channel's LTC6910 programmable gain amplifier to one of its hardware defined values: 0, 1, 2, 5, 10, 20, 50, or 100. The zero setting is used to turn the channel off. Drop-in replacement parts for the LTC6910 exist, allowing for even more gain options.

The Gain configures the amplification of the MSP430's integrated amplifier and can be set to one of its hardware defined values: 1, 2, 4, 8, or 16.

The Attenuation Level adjusts the amount of current delivered to the channel's LED, thereby controlling its brightness. It can be set to 0, 1, 2, or 3, corresponding to a reduction by a factor of 1, 1/2, 1/4, or 1/8, respectively. Because of the inherent linearity of fluorescence, this has essentially the same impact as using a smaller gain/pre-gain combination but achieves this by manipulating the current driver.

An important note worth repeating: the attenuation setting *must* be used if your choice of Excitation LED cannot handle the peak driver current. Since the peak current is just under 50 mA, you *must* attenuate the current signal if this exceeds the Excitation LED's absolute maximum current rating.

Once you have chosen the gain settings for your application, you can commit them to the battery backed memory using the SAVE command.

To begin a single channel measurement using the RUN command, simply send the command “RUN,X,Y”, where X is the desired channel and Y is the desired duration in seconds. Valid choices of Y range from 1 to 3600, for a maximum of 1 hour. The Y value of 3678 is also accepted and is used to make the run duration infinite until stopped or reset.

You can expect the first 8 data points to be invalid at the start of every RUN command, as this is due to transient effects in the PIXIE’s signal processing. The data from RUN is formatted as “DATI,U,V” or “DATA,U,V”, where U is the channel being measured and V is the raw fluorescence measurement. The “DATI” tag is used to index every 16 datapoints, whereas the “DATA” tag marks all datapoints in between. This is simply to allow for easier post-processing. A sample RUN command is provided in Figure 1.5-8 below.

```
>>run,2,10

DATI,2,473
DATA,2,607
DATA,2,534
DATA,2,508
DATA,2,550
DATA,2,567
DATA,2,498
DATA,2,532
DATA,2,536
DATA,2,530
DATA,2,497
DATA,2,478
DATA,2,527
DATA,2,504
DATA,2,582
DATA,2,532
DATI,2,606
DATA,2,553
DATA,2,552
DATA,2,511
```

Figure 1.5-8. Sample output from a RUN command.

To begin a channel sequence measurement, send the command “RUN,0,Y”. By receiving a command to run channel zero for Y seconds, the PIXIE interprets this as

a command to run the channel sequence for Y seconds. The channel sequence is also compatible with a Y value of 3678 to have the sequence run until cancelled. **It is important to note that channel sequences are unaffected by commands such as SGAIN and SATTEN**.

You can expect the first 8 samples of each sequenced step to be invalid, even if it is the same channel as a prior step with different gain settings. The data from a channel sequence RUN is formatted as “SEQI,U,V,W” or “SEQN,U,V,W”, where U is the current step of the sequence, V is the channel being measured at that step, and W is the raw fluorescence measurement at that step. As before, the “SEQI” tag is used to index the first of every 16 datapoints for a given step, whereas “SEQN” is used to tag all datapoints in between. A sample channel sequence RUN command is provided in Figure 1.5-9 below.

```
>>run,0,10

SEQI,0,2,565
SEQN,0,2,652
SEQN,0,2,518
SEQN,0,2,562
SEQN,0,2,481
SEQN,0,2,508
SEQN,0,2,654
SEQN,0,2,595
SEQN,0,2,531
SEQN,0,2,554
SEQN,0,2,613
SEQN,0,2,529
SEQN,0,2,507
SEQN,0,2,584
SEQN,0,2,564
SEQN,0,2,524
SEQI,0,2,394
SEQN,0,2,536
SEQN,0,2,483
SEQN,0,2,485
```

Figure 1.5-9. Sample output from a channel sequence RUN command.

1.6 Calibration

In this section, instructions for how to calibrate the PIXIE are provided. A brief theoretical background is provided, from which the steps taken to calibrate the device are derived. The PIXIE used in calibration is described. Lastly, a detailed description of the calibration protocol for Rhodamine Water Tracer is provided, including the apparatus and results. The results are provided for reference purposes and, provided the default configuration is used, can be used as a rough calibration for a newly assembled PIXIE.

1.6.1 Background

The measurement data reported by the PIXIE is given in terms of unitless counts and represents the amplitude of the sinusoidal fluorescence captured (or lack thereof) by the photodiode, sampled by the device's analog to digital converter (ADC), then processed by the PIXIE's digital Lock-in amplifier. Because the PIXIE uses 16-bit samples to measure amplitude, the measurements will be reported as numbers between 0 and 32767, in units of ADC counts. There is enough unused processing time for the PIXIE to report calibrated data once the channel has been calibrated, but this feature is not implemented. It is recommended that calibration be applied in post-processing instead.

Given the proportional relationship between the concentration of a diluted fluorophore and the intensity of fluorescence it produces when exposed to a fixed excitation, and the linearity of the devices and signal processing techniques used to extract the fluorescence amplitude, the concentration is modelled as directly proportional to the fluorescence amplitude. If the concentration is given by C and the amplitude (i.e.: the raw measurement data) is given by R (see Design Note 2.4 for details), the ideal equation for concentration in terms of amplitude is given by:

$$aC = R\,,$$

where a is a temperature-dependent calibration constant. In practice, sensors are known to exhibit an offset and holds true for the PIXIE due to the presence of noise and voltage ripple. A more reasonable relationship is the affine equation:

$$aC = (R - R_0),$$

where R_0 is a non-zero, approximately constant measurement offset present even when the fluorophore concentration is zero.

Two factors that affect the amount of fluorescence generated by a fixed fluorophore concentration are quenching agents and temperature. Quenching agents – chemicals that steal excited-state energy at an atomic level – are generally compensated for externally. However, the temperature dependence of fluorescence is well studied (Smart and Laidlaw, 1976). For large differences in temperature the relationship is exponential, but for smaller differences in temperature (such as those found across ocean water columns) the relationship can be modelled with a linear approximation (see Park et al, 2023). Therefore, the relationship between temperature T and calibration constant a can be modelled with the equation:

$$a = S_0 + S_1 T,$$

where S_0 and S_1 are the linear approximation constants of the exponential relationship. A fluorophore's degree of exponential dependence is reported in literature by its temperature exponent n. It can be shown that the temperature exponent is approximated by the formula:

$$n \approx S_1 / S_0 \, .$$

From the formulae described above, the basic steps needed to calibrate a PIXIE channel can be outlined as follows:

- Determine an estimate of R_0 by measuring R at various temperatures while submerged in a volume with zero concentration and keep the mean result.
- Subtract R_0 from all future measurements of R.
- For a fixed concentration of fluorophore, measure the fluorescence at various temperatures (fix C, measure $R - R_0$, estimate a).
- Repeat the previous step for various concentrations.
- For each temperature, plot the measurements ($R - R_0$) against concentration and find the best-fit line. The slope of this line is a for this temperature.
- Repeat the previous step for each temperature.
- Plot a against T and find the best-fit line. The offset and slope of this line are S_0 and S_1, respectively.

Once R_0, S_0, and S_1 are known, the calibrated concentration measurement is given by:

$$C_{meas} = \frac{R_{meas} - R_0}{S_0 + S_1 T},$$

where R_{meas} is the raw sensor output in ADC counts and C_{meas} is the calibrated, temperature-compensated concentration measurement in the concentration units of choice (ppb, µg/L, etc.).

1.6.2 Calibrated PIXIE

The PIXIE has been calibrated for the fluorescence of Rhodamine Water Tracer (RWT). The calibrated PIXIE used the default gain settings of 8 Gain, 16 Pre-gain, with Attenuation level 0 for its RWT channel, or Channel 2 by default. The calibration results of that channel are not necessarily valid with different gain settings, so individual calibrations should be performed for each gain setting you intend to use.

The calibrated PIXIE described in this section used the default RWT excitation LED as described in Section 1.1.1 and Section 1.1.4, the Marktech MTE5270P-C. The calibrated PIXIE had only one hardware modification when compared to the default configuration. The STMicroelectronics LDLN015PU33R 3.3 V voltage regulator IC was used in place of the default Analog Devices LT1762-3.3 voltage regulator. Figure 1.6-1 below shows this hardware modification and the delicate micro-soldering required to perform it.

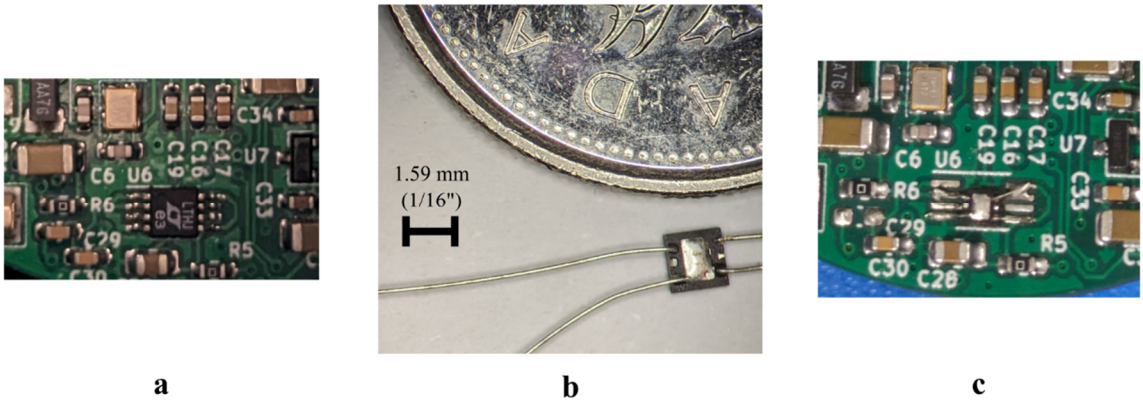


Figure 1.6-1 (a) Default-configuration PIXIE PCB showing U6 populated with LT1762-3.3. (b) Soldering micro-wires to a LDLN015PU33R. Scale bar and Canadian 25-cent piece included for reference. (c) Modified PIXIE PCB showing U6 populated with LDLN015PU33R.

After testing the modified PIXIE against default-configuration PIXIEs, there is **no evidence to suggest a difference in measurement performance between the two configurations**. Therefore: the results of this calibration are applicable irrespective of the voltage regulator chosen, provided all other choices remain identical. Because of its higher range of tolerated input voltages and active-part status, the default LT1762-3.3 is recommended in all cases.

1.6.3 Calibration Protocol

The US Environmental Protection Agency (EPA) Method 445 (see Arar and Collins, 1997) recommends the use of five standard concentrations plus the zero concentration, or blank, standard for its method on Chlorophyll-a fluorometer calibration. The calibration protocol described here adapted this method to Rhodamine Water Tracer (RWT) with temperature compensation, using six concentrations of RWT (blank plus five), across six temperatures, for a total of 36 data points. To achieve temperature stability, a laboratory chiller (NESLAB RTE-7, Thermo Scientific) and a jacketed beaker were used. These are expensive components; an alternative approach would be to cool each sample in an ordinary beaker to common refrigerator temperatures and allow them to warm to room temperature slowly, while electronically logging real-time standard temperature data with the fluorescence data. However, this would require a different analysis approach than that outlined in Section 1.6.1.

The calibration apparatus was as follows:

- The PIXIE was powered with 5 V by a laboratory DC power supply. The power supply is isolated from other electrically noisy lab equipment on the same circuit using an isolation transformer. Alternatively, a battery pack could be used, though this is an inherently less-repeatable option.
- The PIXIE's RS-232 output was connected to a PC using a RS-232 to USB converter cable. The PC's terminal window was used to send commands and log the calibration data for each trial. For further isolation, a battery-powered laptop could be used instead.
- The PIXIE was suspended above a 250 mL jacketed beaker, which sat on a magnetic stirrer. The PIXIE was suspended using a common adjustable beaker clamp. Care was taken to avoid trapping bubbles with the PIXIE.
- The jacketed beaker was plumbed to the laboratory chiller so that cold water could circulate, controlling the temperature of the standard contained in the beaker.
- A temperature probe was suspended in the beaker to measure the standard temperature directly and monitored using a multimeter.
- A simple cardboard shroud was used to cover the PIXIE and the calibration standard, to block out the worst of the ambient sunlight. The PIXIE was not

calibrated in complete darkness, to demonstrate its reliability in well-lit environments even during calibration.

The following table, Table 1.6-1, summarizes the uncommon components and instruments of the calibration apparatus. The power supply, chiller, and isolation transformer listed are rated for North American 120V mains voltage; alternatives must be sourced for other locales.

Table 1.6-1. List of uncommon Calibration Apparatus components and instruments.

Component	Name (Manufacturer)	URL
Power Supply	GPE-3323 (GW Instek)	http://tinyurl.com/muhs6n6t
Isolation Transformer	171B (Hammond Manufacturing)	http://tinyurl.com/36caa5ns
Chiller	NESLAB RTE-7 (Thermo Scientific)	http://tinyurl.com/5n9besf3
Multimeter	True RMS Multimeter 287 (FLUKE)	http://tinyurl.com/4evex66v
Jacketed Beaker	250 mL jacketed beaker (Various)	http://tinyurl.com/3wenauww

The RWT concentration standards prepared for this calibration were nominally 0 ppb (blank), 0.5 ppb, 2 ppb, 10 ppb, 30 ppb, and 60 ppb. For these purposes, the 1 ppb RWT = 1 µg/L convention is used. These standards were prepared by dilution with ultra-pure deionized water. They could also be prepared using Pharmacy-grade (USP) distilled water provided they are used quickly after preparation. Tap water should be avoided, as chloride and fluoride can cause RWT to degrade.

The calibration standards were diluted from a stock of Rhodamine WT 20% dye (Thermo Scientific, https://www.fishersci.ca/shop/products/rhodamine-wt-20-solution-water/p-4440197). The stock is 20% RWT by mass. The stock was first diluted by measuring 100 mg of dye using a sensitive mass balance, then the dye is washed into a 1000 mL volumetric flask until the fill line was reached. This created an initial stock of 20 mg/L, or 20 ppm by convention.

From the 20 ppm stock, 10 mL were taken and diluted using a 500 mL volumetric flask, which created a secondary stock of 400 ppb. Alternatively, the 400 ppb calibration standard could be purchased directly (Turner Designs, https://www.turnerdesigns.com/product-page/400-ppb-rhodamine-wt), saving the need for a sensitive mass balance and stock of concentrated dye.

From the 400 ppb stock the final concentrations of 0 ppb, 0.5 ppb, 2 ppb, 10 ppb, 30 ppb, and 60 ppb were prepared, in 500 mL quantities.

Once the apparatus was assembled and the calibration standards were prepared, the calibration data collection began. The standards were measured from lowest to highest concentration, starting with the 0 ppb (blank) standard. Figure 1.6-2 offers a photograph of the PIXIE collecting data during a 0 ppb calibration point. Approximately 220 mL of the standard was dispensed into the jacketed beaker so that the sensing end of PIXIE could be submerged without trapping bubbles or causing an overflow. At each concentration, the chiller was set to hold the standard temperature at 5, 8, 11, 14, 17, and 20 °C as reported by the submerged temperature probe. At each temperature, 15 minutes' worth of data was collected with the PIXIE by issuing the "RUN,2,900" command.

Once the measurements for each of the selected temperatures were complete, the PIXIE was removed from the beaker and the calibration standard was poured into a waste beaker. All implements in contact with the calibration standard (the probe, the beaker, and the PIXIE) were thoroughly rinsed with deionized water and gently blown dry with compressed air. The next standard was poured into the beaker and the position of the PIXIE was reset. The procedure was repeated until all six standard concentrations had been measured at all six temperature setpoints, for a total of 36 measurements. Incidentally, the 60 ppb standard at 5 °C saturated the output of the PIXIE, so one extra temperature measurement of 6.9 °C. was collected.

Figure 1.6-2. PIXIE collecting data at 0 ppb calibration point, suspended in beaker by a beaker clamp.

From the 15 minutes' worth of data collected at each point, only the last 5 minutes were used. The remaining 5 minutes worth of data, sampled at the default rate of 16 samples per second, equated to 4800 points of raw measurement data. These data were then downsampled by averaging sequential windows of 16 datapoints, such that the 5 minutes worth of data is now effectively 300 raw datapoints sampled at 1 sample per second. These steps were taken to make the results comparable to previously published work (see Park et al, 2023).

Once all the data were collected, the calibration curves were produced using the formulae described at the beginning of this section. For each 5 minute dataset, the mean was calculated and taken as the measurement R, along with the standard deviation. For the 0 ppb (blank) measurements, the mean *across* temperatures was taken as R_0. A replacement 60 ppb, 5 °C datapoint was then linearly extrapolated from the remaining 60 ppb data, including the extra 6.9 °C measurement. The data were then offset by subtracting R_0.

The best-fit lines for each temperature were calculated using the conventional approach fixing the now-offset data, $(R - R_0)$, to a vertical intercept of zero. Figure 1.6-3 below provides the calibration curves for each temperature. The level of device saturation is indicated by the dashed line, whereas extrapolated 60 ppb, 5 °C datapoint is encircled. The inset plot provides horizontal 10-sigma error bars for the midpoint, 30 ppb, in the devices' dynamic range at 5 °C, as an indicator of overall measurement precision.

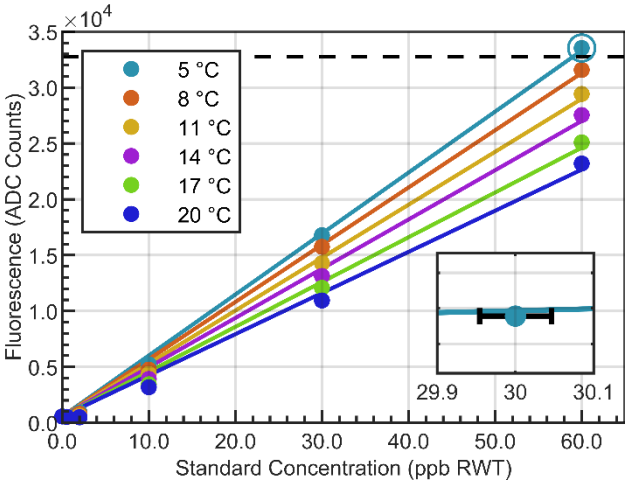


Figure 1.6-3 Plot of calibration curves, per temperature, for RWT calibration.

The coefficient of determination (r^2) for each calibration curve exceeds 0.99. The slope of each line in is an estimate of a at the indicated standard temperature. By plotting the slope of each line at its corresponding temperature, Figure 1.6-4 was obtained.

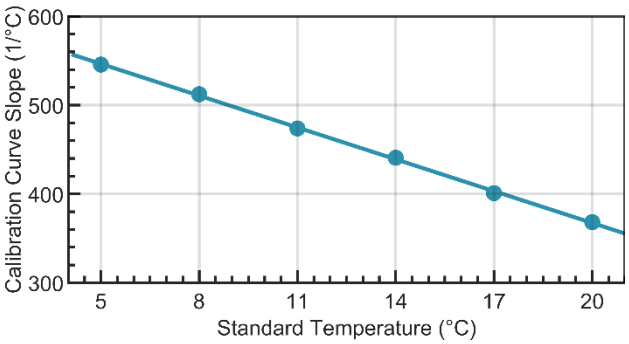


Figure 1.6-4 Plot of calibration curve slopes vs. temperature.

The best-fit line of this curve was then calculated using the conventional approach to obtain both the vertical intercept, S_0, and the slope, S_1. The coefficient of determination of this curve exceeds 0.99.

The resulting temperature compensated RWT calibration equation is given as:

$$C_{meas} = \frac{R_{meas} - (567.2771)}{(606.2103) - (11.9457)T} .$$

The approximate temperature exponent is therefore:

$$n \approx -\frac{11.9457}{606.2103} = -0.019\ ^{\circ}C^{-1} .$$

This approximate temperature exponent is consistent with the same analysis as performed on other submersible fluorometers (see Park et al, 2023), but is notably different from the RWT temperature exponent from literature of -0.026 °C^{-1} (Smart and Laidlaw, 1976). This is likely due to the temperature dependent loss of fluorescence being offset by the temperature sensitivity of the devices' optoelectronics, whereas the literature reference is isolated to the temperature sensitivity of RWT exclusively.

The PIXIE's limit of detection (LOD) is defined in the standard way (Arar and Collins, 1997; Sieben et al, 2010) as three-times the standard deviation of the 0 ppb (blank) data. Since 6 blank tests were performed, the LOD is taken as the worst-case (largest) result among the blanks *after* converting the standard deviation from raw

data, with units of ADC count, to a concentration, with units of ppb. This is achieved by simply dividing the standard deviation of the blank tests by the calibration constant. In other words, the LOD can be calculated with the following equation:

$$LOD = 3s_C = 3\frac{s_R}{(606.2103) - (11.9457)T},$$

where s_C and s_R are the standard deviations of in terms of calibrated concentration and in terms of raw measurement, respectively, at temperature T. The worst case LOD is found to be less than 0.01 ppb, putting the PIXIE's RWT measurement performance on par with industrial competitors (Park et al, 2023).

Below summarizes the resulting parameters of each calibration curve.

Table 1.6-2 Summary of calibration curve parameters.

Temperature [°C]	r^2	R_0	a	s_R [counts]	LOD [ppb]
5	0.999	567.2771	545.6360	0.4956	0.0027
8	0.998	"	512.2257	0.9874	0.0058
11	0.997	"	473.7119	0.4301	0.0027
14	0.995	"	440.7003	0.3725	0.0025
17	0.995	"	400.8652	0.9764	0.0072
20	0.994	"	368.1947	0.4964	0.0040

2 Design Notes

2.1 Optical

In this section, the design and function of the PIXIE's optical stack is briefly explored. This is presented to provide insight into the optical design and how it could be impacted by customizations.

The size of the PIXIE's aluminum housing constrains the design of optical stack, necessitating the packing of four independent optical pathways into a sub-2" diameter barrel. To save space, the condenser lens used to collect the fluorescent emissions is also used to steer the excitation light from the LED. To achieve the best alignment between the LED's radiation pattern and the condenser lens' back focal length, some of the LED light is allowed to leak. A diagram of the optical stack, with a photograph of the 3D-printed part and a ray-tracing scale diagram is provided in Figure 2.1-1. The section-view indicated in Figure 2.1-1b) is used to create the ray-tracing diagram in Figure 2.1-1c).

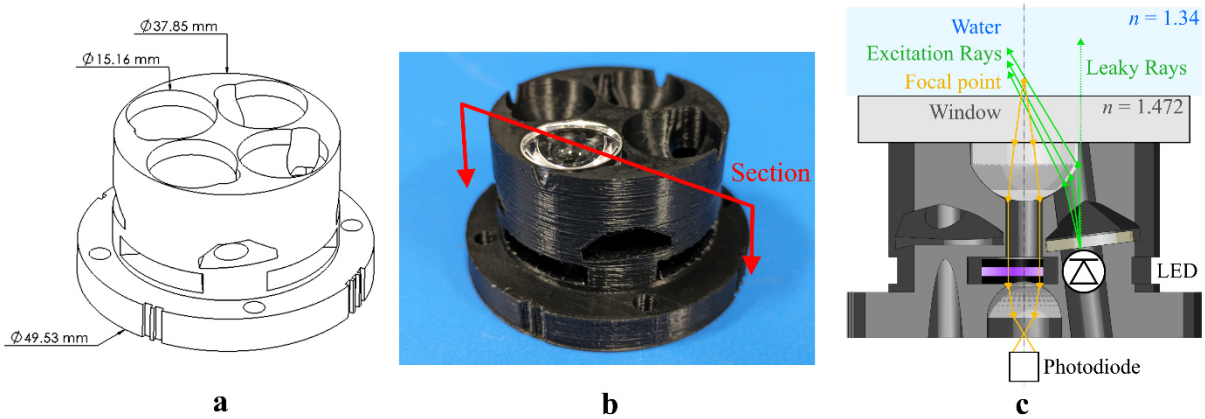


Figure 2.1-1 The PIXIE's optical stack. (a) Drawing view, showing some dimensions. (b) Photograph of 3D printed optical stack with a single condenser lens installed, with a section-view cut indicator. (c) A ray-tracing scale diagram of the optical stack, using the section-view indicated in (b). The pathway of the green excitation and resulting yellow fluorescence through the lens and foci are illustrated.

The window depicted in Figure 2.1-1c) is the default-configuration 6.5 mm borosilicate glass window. The current design respects the incidence angle and clear-aperture specifications of the filters and window, the viewing angle of the LED and photodiode, and the back focal length of each lens, without causing the parts to physically interfere with the optical pathways of the adjacent channels.

In the default configuration, the emission filters specified are optical density (OD) OD6 whereas the excitation filters are OD4. The emission and excitation share no spectral overlap, eliminating the possibility of optical crosstalk from LED to Photodiode.

2.2 Electrical

In this section, the electrical architecture of the PIXIE is briefly explored. This is presented to provide insight into the circuit design and how it could be impacted by customizations.

The electrical architecture of the PIXIE can be split into three primary subsections: the MSP430FG6626 microcontroller, the current-signal LED driver, and the Lock-in Fluorescence detector. Two secondary subsections can also be considered: connectors and I/O, and voltage regulation, but these are left out for the sake of brevity. The full schematic for the PIXIE is available.

An electrical architecture block diagram is provided below in Figure 2.2-1.

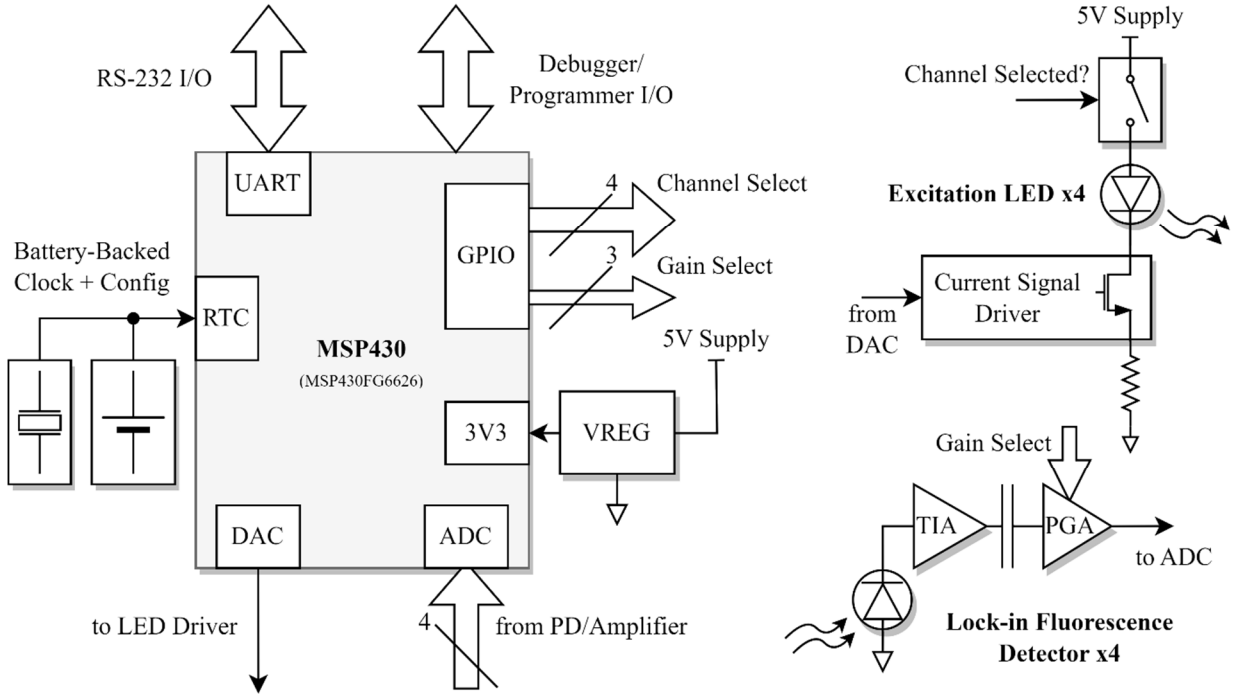


Figure 2.2-1 PIXIE Electrical Architecture block diagram.

The current-signal driver subsystem contains four individually selectable LEDs and drives them with a current-signal waveform, which is generated by the digital to analog converter (DAC). This design has a compact layout, but it cannot drive more than one LED at a time without splitting the current signal across each, which would lower the maximum-achievable brightness of each channel.

The lock-in fluorescence detectors are AC-coupled transimpedance amplifiers (TIAs) with two additional stages of programmable gain: one discrete programmable-gain

amplifier (PGA), and a second PGA integrated with the MSP430 (ADC) input. The AC coupling is used between the first and second stages of amplification to reject low-frequency or “DC” sources of detected light, allowing the PIXIE to perform well in bright environments without a physical shroud.

Since the MSP430FG6626 has only one high-performance ADC, only one detector can be indexed and measured at a time. This is ultimately why the PIXIE can only measure with one channel at a time.

2.3 Firmware

In this section, the firmware architecture of the PIXIE is briefly explored. This is presented to provide insight into the firmware and how it could be impacted by customizations.

The PIXIE firmware is composed in Texas Instruments' (TI's) Code Composer Studio (CCS) and is primarily written in the C programming language (with one exception, described later). The source code uses the native "msp430.h" library to access all chip-specific definitions for the deployed microcontroller, the TI MSP430FG6626, along with C standard libraries where required. The PIXIE does not use any other external libraries.

The firmware's source code follows an idiosyncratic Coding Standard inspired in-part by the style of the open-source *Simple DirectMedia Layer* open-project by Sam Lantinga. The intent is to ease users unfamiliar with C, but perhaps familiar with other modern programming and scripting languages, into making the minor modifications they need to configure their PIXIE. All outward facing declarations (i.e.: global variables, functions) are prefixed with a two-to-five letter namespace, whereas local functions and variables are allowed to go without a prefix. Every source file contains detailed commentary explaining definitions, variables, use cases, functions, and code blocks, on a line-by-line basis in some circumstances. The source code itself should be consulted for a detailed description of each file: this note is meant to provide a top-down overview of the firmware.

Each element of the firmware (i.e.: each source file) can be organized into one of three layers: the Soft Layer, the Firm Layer, or the Hard Layer. Soft Layer code deals strictly with objects in RAM and is essentially agnostic of the platform the code is being run on. Firm Layer code can call upon the functions of the other layers and directly interacts with peripherals and registers of the MSP430FG6626. Hard Layer code directly interacts with peripherals and registers of the MSP430FG6626, but also with the pins of the MSP430FG6626 and external electronics of the PIXIE printed circuit board.

The firmware source code is well-encapsulated and interaction between elements is kept to an absolute minimum. The only code allowed to transcend this organization rule is that of "main.c", which takes the central role of coordinating the firmware and performing the PIXIE's fluorometry.

The only source code not written in C is “dsp.asm”, which is written in MSP430 extended assembly and externalized to the C language through the header “dsp.h”. It is a hand-optimized implementation of the digital signal processing (DSP) used to implement the PIXIE’s digital lock-in amplifier. It can be considered Firm Layer code due to its use of the MSP430’s MPY32 32-bit Hardware Multiplier peripheral, which accelerates the DSP’s fixed-point arithmetic operations. The signal processing implemented by this assembly code is described in the next Design Note, Section 2.4.

To illustrate how the individual elements of the firmware are organized and when they interact, a Firmware Architecture diagram, Figure 2.3-1, is provided below. An orange arrow indicates an element has direct access to the one it points to via included headers. A blue dashed arrow indicates an element that interacts with the one it points to at a peripheral level, but not at a program/code access level. The MAIN element has direct access to all other elements, whereas all elements have direct access to PIXIE.

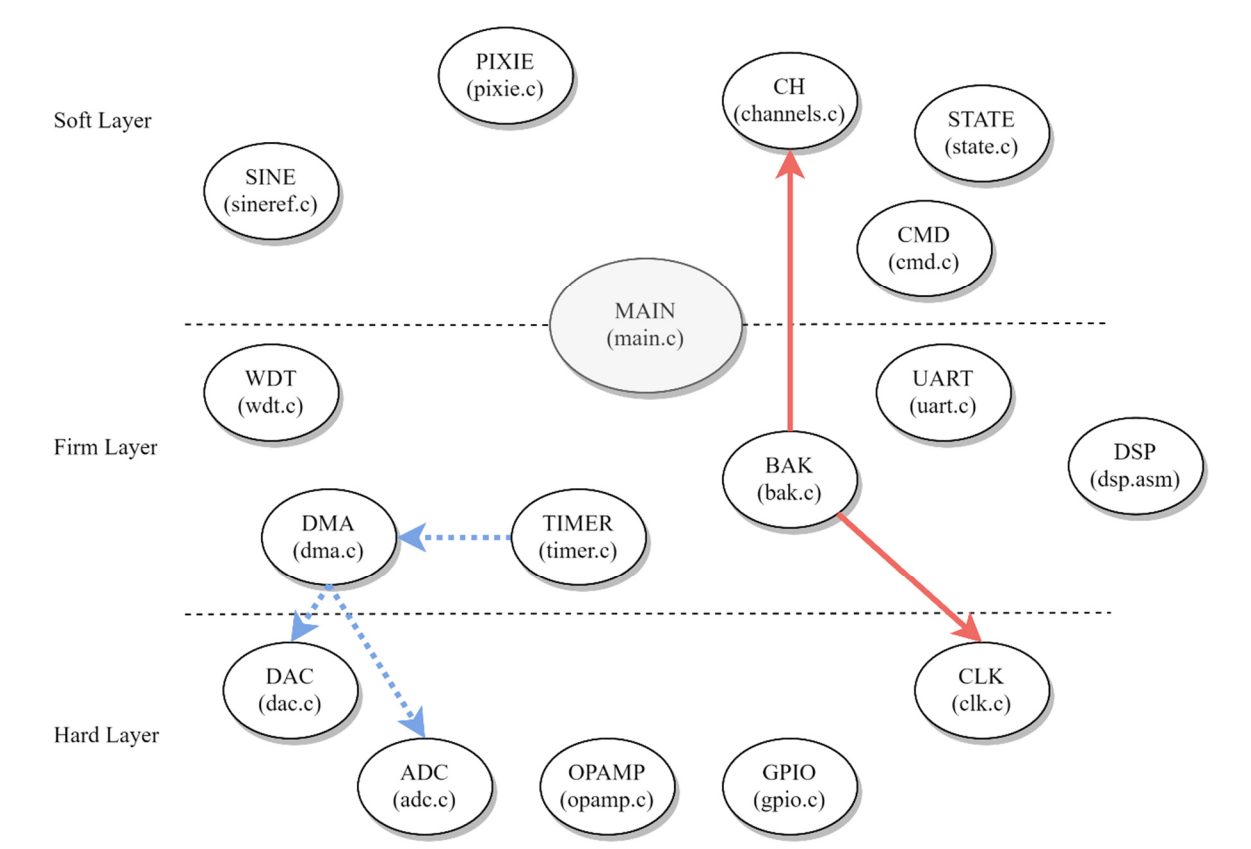


Figure 2.3-1 PIXIE Firmware architecture diagram.

The PIXIE source code (“pixie.c”, “pixie.h”) is intended to be the easiest for non-programmers to interact with, as it contains the few necessary lines of code to adjust and configure the PIXIE device. These configurations are: setting the PIXIE to run in DEBUG or IMMEDATE_ENDLESS mode, setting the name and default gain settings of each channel, and setting up the devices’ channel sequence. Instructions for making these configurations are provided in Section 1.4.

The PIXIE has four device states as described in Section 1.5.2. Only two states are intended to be used in normal operation: IDLE and RUN.

When in the IDLE state, the program flow is primarily controlled by the input Universal Asynchronous Receiver/Transmitter (UART). The main loop sleeps in a low power mode, LP0, until the UART has received a string of characters ending with a newline ‘\n’ or carriage return ‘\r’. The main loop is awoken and parses the received command to see if it’s a valid command. If it is a properly formatted command, the corresponding handler function is called, and the main loop returns to sleep. The IDLE state persists until an explicit command to RUN is received.

When in the RUN state, the program flow is primarily controlled by the Timer peripheral through triggering the Direct Memory Access (DMA) peripheral. The main loop continues to sleep and the UART may be serviced unless higher priority tasks interrupt them. Independently from the main loop, data are pulled from the Analog to Digital converter (ADC) by the DMA at a rate defined by the Timer. The DMA peripheral transfers the data into a buffer: when the buffer is full, the DMA interrupt service routine (ISR) is called. The DMA ISR has execution priority above all other processes in the PIXIE so that its primary purpose, fluorometry, is not interfered with. The DMA ISR calls “dsp.asm” to process the buffered data into a single point of raw fluorometry data. The raw fluorometry data is formatted and sent to the output UART, after which control is returned to the main loop where either sleep is resumed or the next highest priority task (i.e.: servicing the input UART) is executed. The RUN state persists until the duration of the run command has elapsed, though a special case exists for an infinite-duration, or a STOP command is received, returning the device to the IDLE state.

In all states, the Real Time Clock (RTC) requests an interrupt once every second. The RTC ISR is low priority and will not prevent the DMA or UART from operating on

time. The purpose of the RTC ISR is to save a snapshot of the current time in RAM where it can be accessed as needed, rather than having real-time related tasks access the RTC registers directly. This is because poorly timed access calls to these MSP430 registers can cause timing glitches, leading to clock drift. When serviced by the RTC ISR, access to these registers is guaranteed to be glitch-free.

To illustrate the described program flow of the PIXIE firmware, Figure 2.3-2 is provided below.

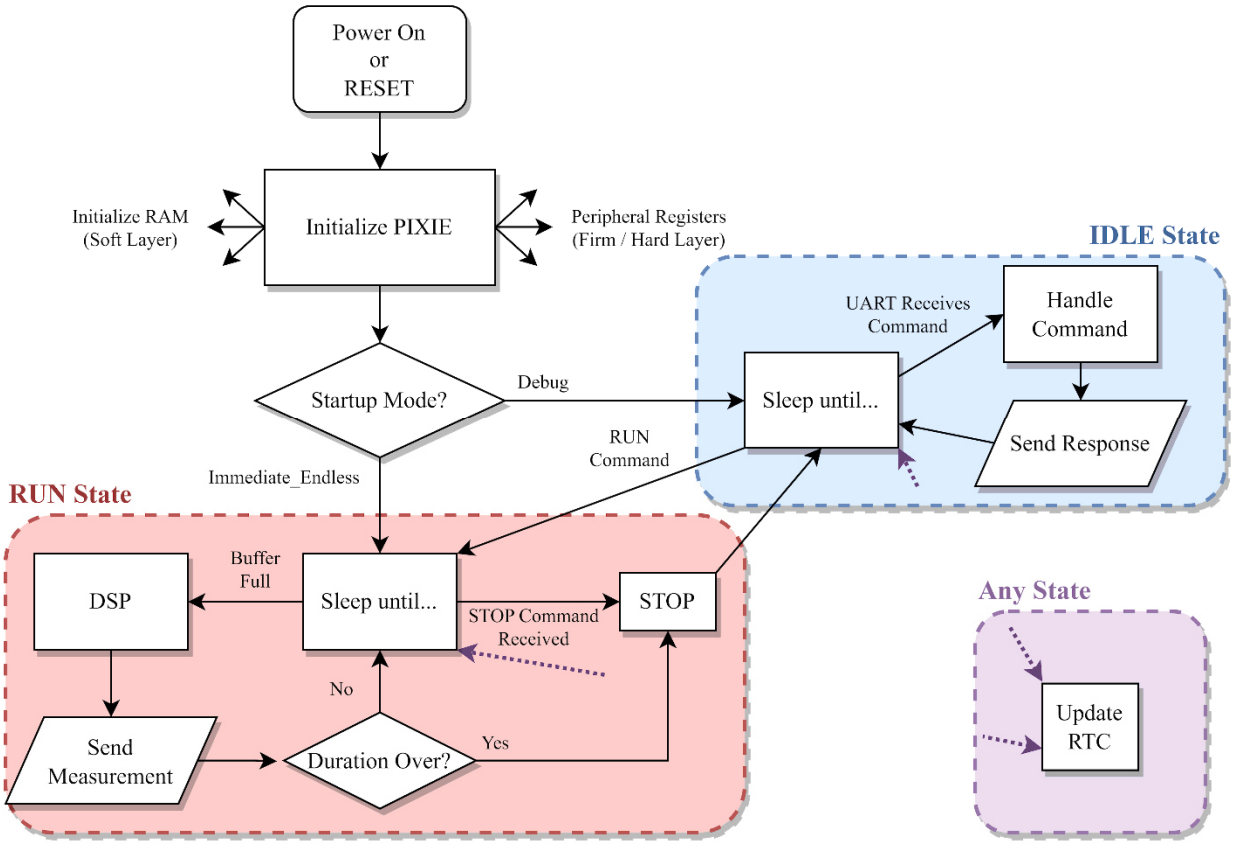


Figure 2.3-2 PIXIE Firmware program flow diagram.

2.4 Signal Processing

In this section, the digital signal processing performed by the PIXIE is briefly explored. This is presented only as an outline, as the technical content involved in this design is far beyond the scope of this user guide.

A block diagram describing the signal analysis performed by the PIXIE is provided below in Figure 2.4-1.

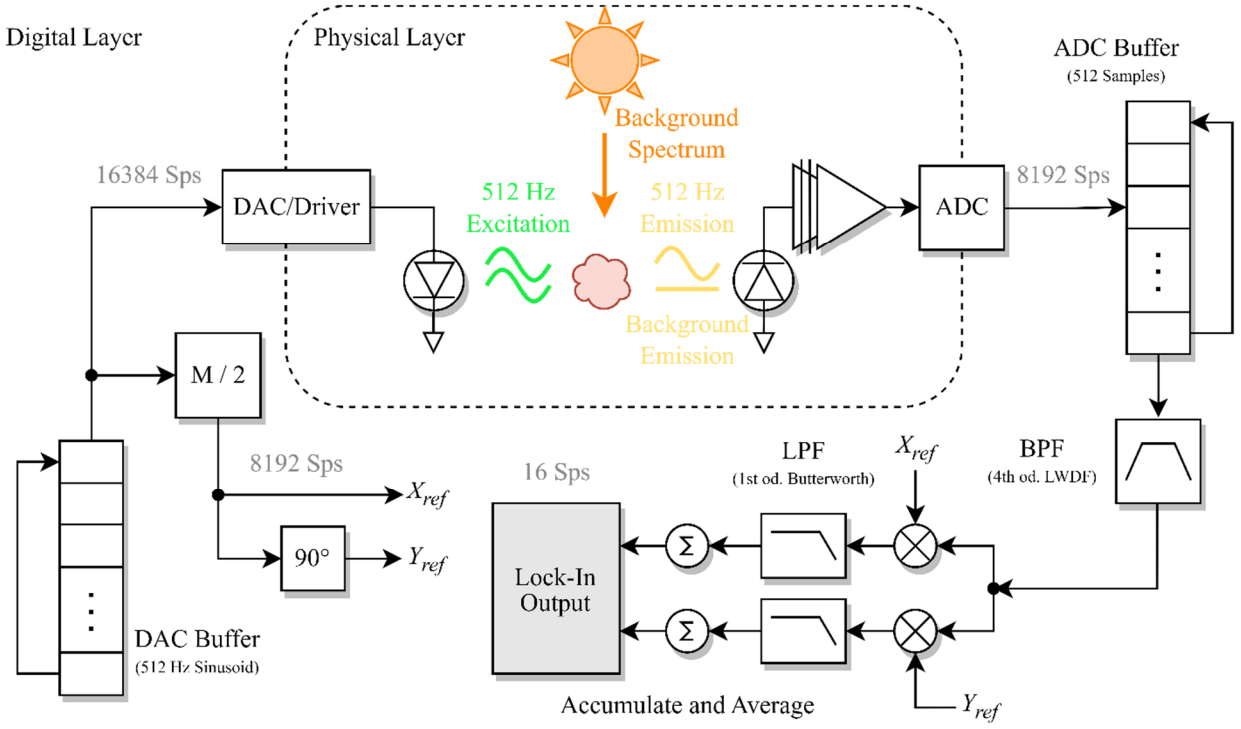


Figure 2.4-1 Block-diagram of the PIXIE's signal processing chain, illustrating the connection between physical and digital signal layers.

The Physical Layer represents the excitation of a given concentration of fluorophore using sinusoidal LED light, and the detection of the sinusoidal fluorescence emission plus any background emissions. Once detected by the AC coupled amplifier, the signal is sampled at 8192 Hz until 512 samples are buffered in the Digital Layer. This triggers the "Analysis Loop", which processes the buffered data as fast as possible.

Samples from the 16-bit Analog to Digital converter (ADC) ADC data are converted to a proprietary Q3.28 fixed point format to maximize numerical precision and minimize filter transients. The ADC data are first bandpass filtered using a 4th order Lattice-Wave Digital Bandpass filter (BPF) implemented as two 2nd order filters in

cascade. This narrow bandpass filter rejects as much of the sampled noise as possible while keeping a small bandwidth around the frequency of interest, 512 Hz. By filtering at passband instead of baseband, a much faster response time is observed. The data are then mixed with an in-phase reference signal X_{ref} and an in-quadrature reference signal Y_{ref}. This creates two mixed versions of the ADC sampled signal, each with sum-and-difference frequency components. The sum frequency component has a frequency of 1024 Hz, whereas the difference frequency component is DC.

The mixed signals are each filtered using a Direct-Form 1st order Butterworth lowpass filter (LPF), and then buffered until all 512 sample points have been processed. The mean of each buffer is calculated (sum and divide by 512), giving an in-phase measurement X and an in-quadrature measurement Y at an effective report rate of 16 measurements per second. This average fully eliminates the sum-frequency content from each measurement, since the reference signals are exactly periodic in the size of the buffer. It can be shown (Park et al, 2023) that the amplitude measurement R of the "locked in" sinusoidal fluorescence is given by:

$$R = 2\sqrt{X^2 + Y^2}$$

This amplitude is directly proportional to the concentration of fluorophore in the illuminated volume.

3 Works Cited

Arar, E.J., and G.B. Collins. 1997. *Method 445.0 in Vitro Determination of Chlorophyll a and Pheophytin Ain Marine and Freshwater Algae by Fluorescence*. EPA 445, The US Environmental Protection Agency, Washington, DC. 22 pp.

Kristoffersen, A.S., S.R. Erga, B. Hamre, and O. Frette. 2018. Testing Fluorescence Lifetime Standards using Two-Photon Excitation and Time-Domain Instrumentation: Fluorescein, Quinine Sulfate and Green Fluorescent Protein. *Journal of Fluorescence* 28:1065–1073, https://doi.org/10.1007/s10895-018-2270-z.

Park, K.T., J.J. Creelman, A.S. Chua, T.S. Chambers, A.M. MacNeill, and V.J. Sieben. 2023. A Low-Cost Fluorometer Applied to the Gulf of Saint Lawrence Rhodamine Tracer Experiment. *IEEE Sensors Journal* 23(15):16772-16787, https://www.doi.org/10.1109/JSEN.2023.3283977.

Poniedzialek, B., H.I. Falfushynska, and P. Rzymski. 2017. Flow cytometry as a valuable tool to study cyanobacteria: A mini-review. *Limnological Review* 17(2):89-95, https://doi.org/10.1515/limre-2017-0009.

Smart, P.L., and Laidlaw. 1977. An evaluation of some fluorescent dyes for water tracing. *Water Resources Research* 13(1):15-33, https://www.doi.org/10.1029/WR013i001p00015.

4 Appendices

4.1 Appendix A: Bambu Studio Settings

The Advanced settings for the Quality, Strength, Speed, Support, and Other tabs in Bambu Studio are provided here for reference.

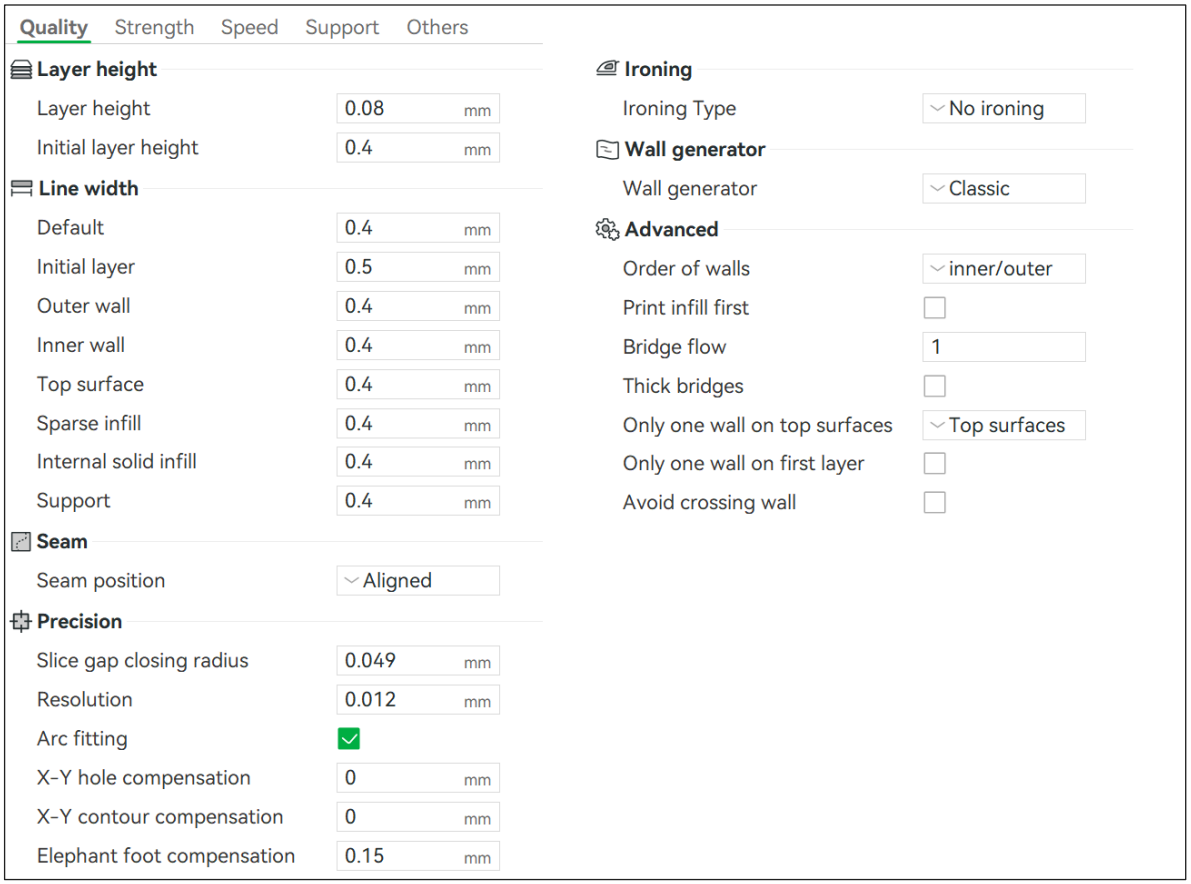


Figure 4.1-1 Bambu Studio Advanced settings for the Quality tab.

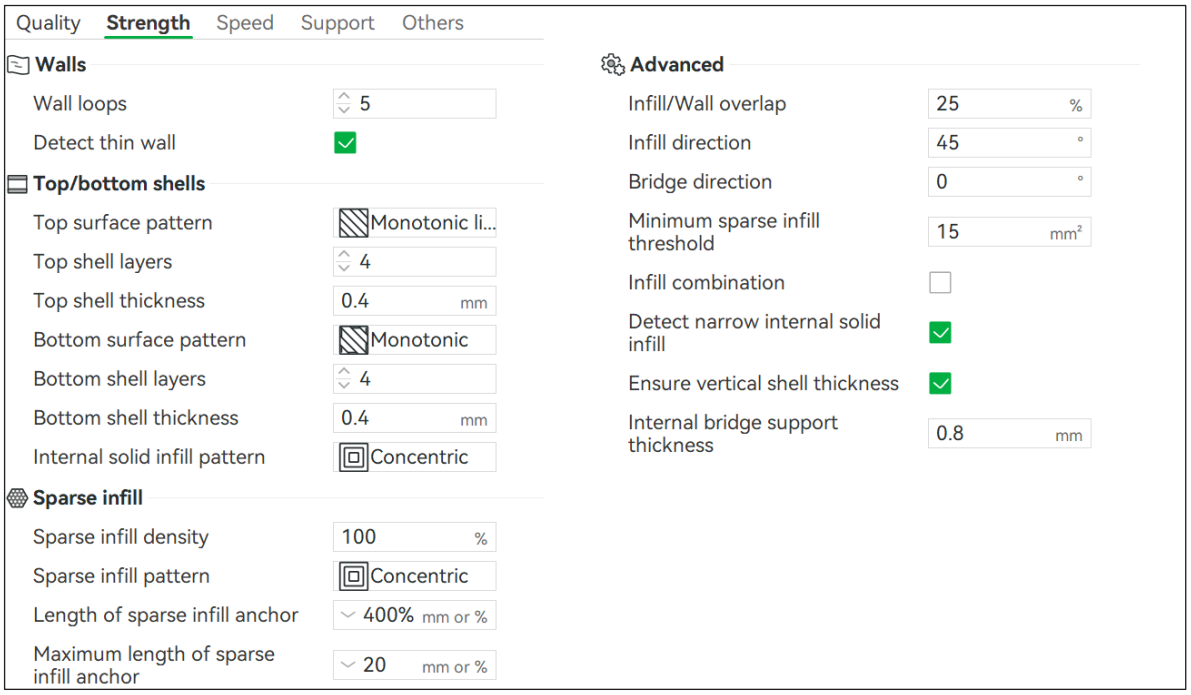


Figure 4.1-2 Bambu Studio Advanced settings for the Strength tab.

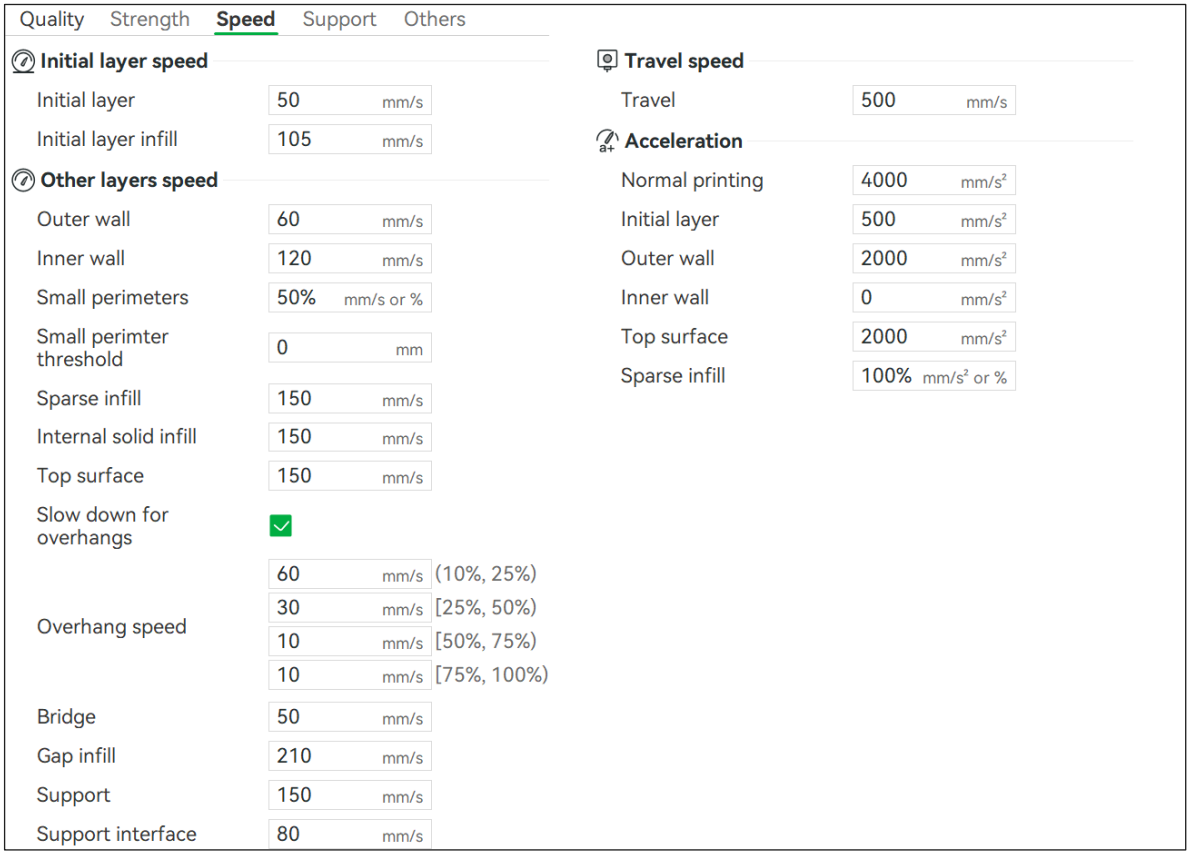


Figure 4.1-3 Bambu Studio Advanced settings for the Speed tab.

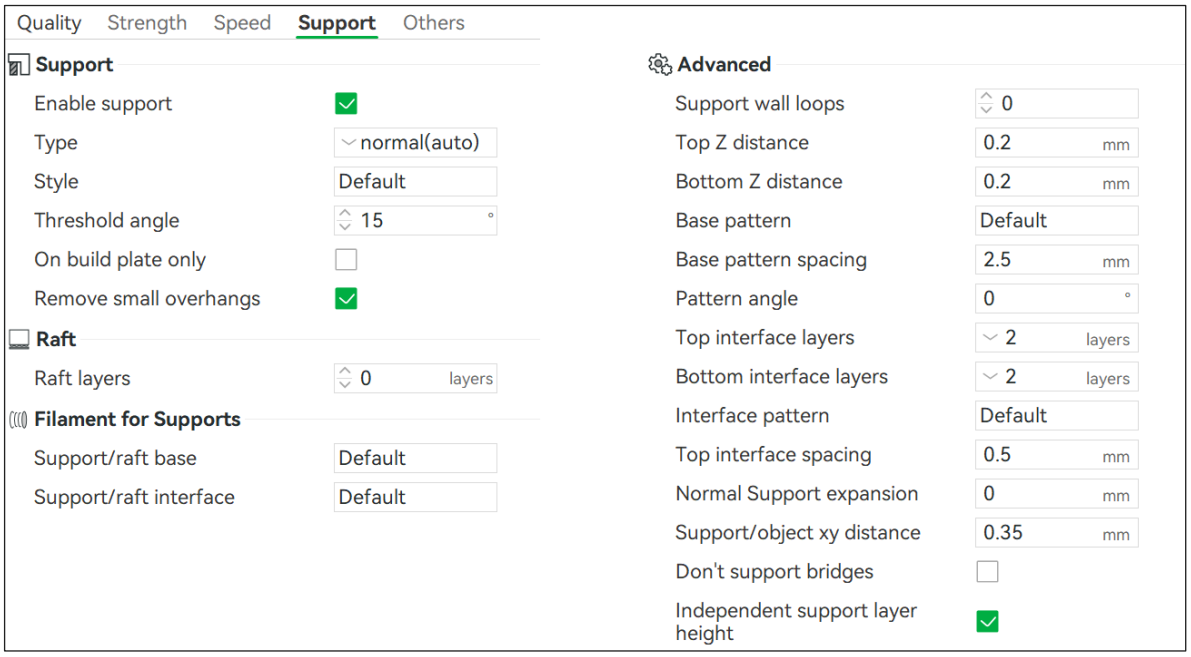


Figure 4.1-4 Bambu Studio Advanced settings for the Support tab.

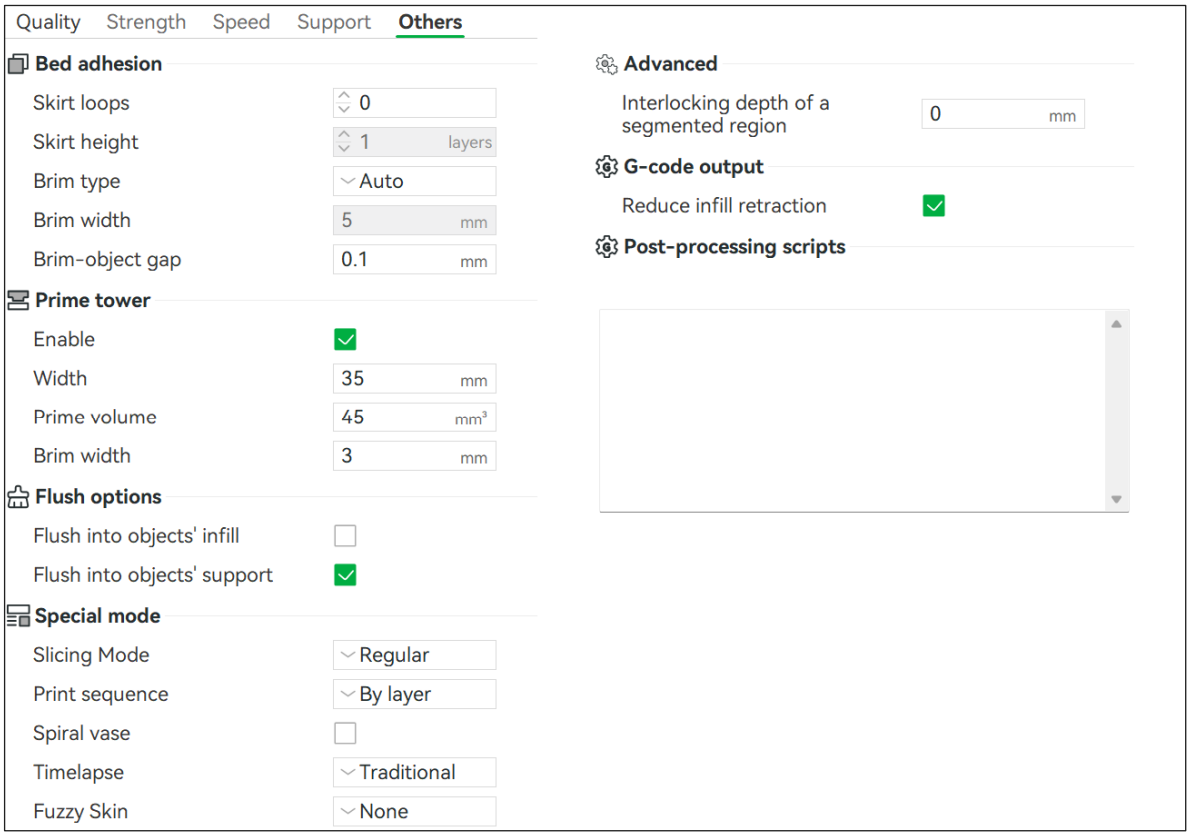
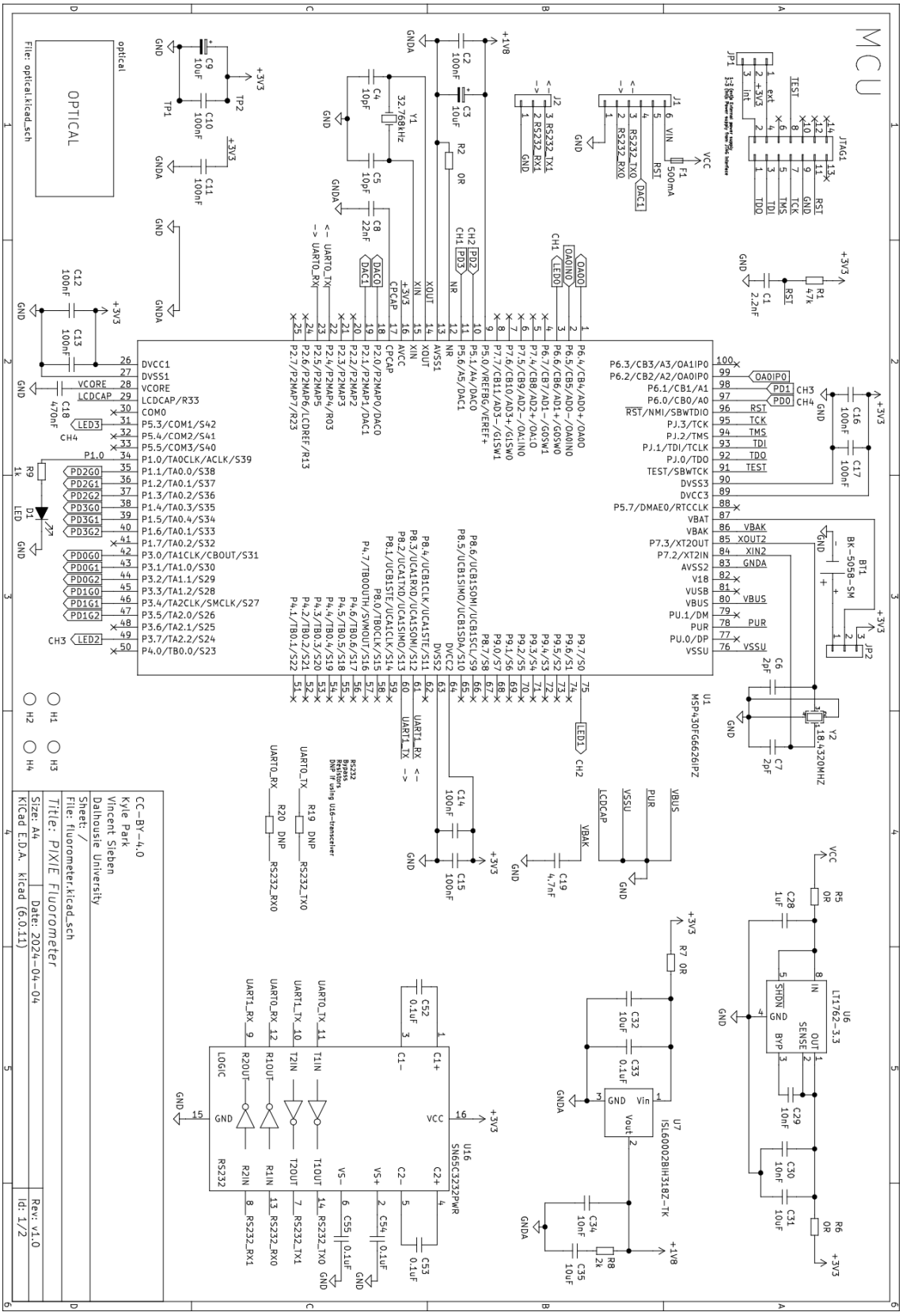
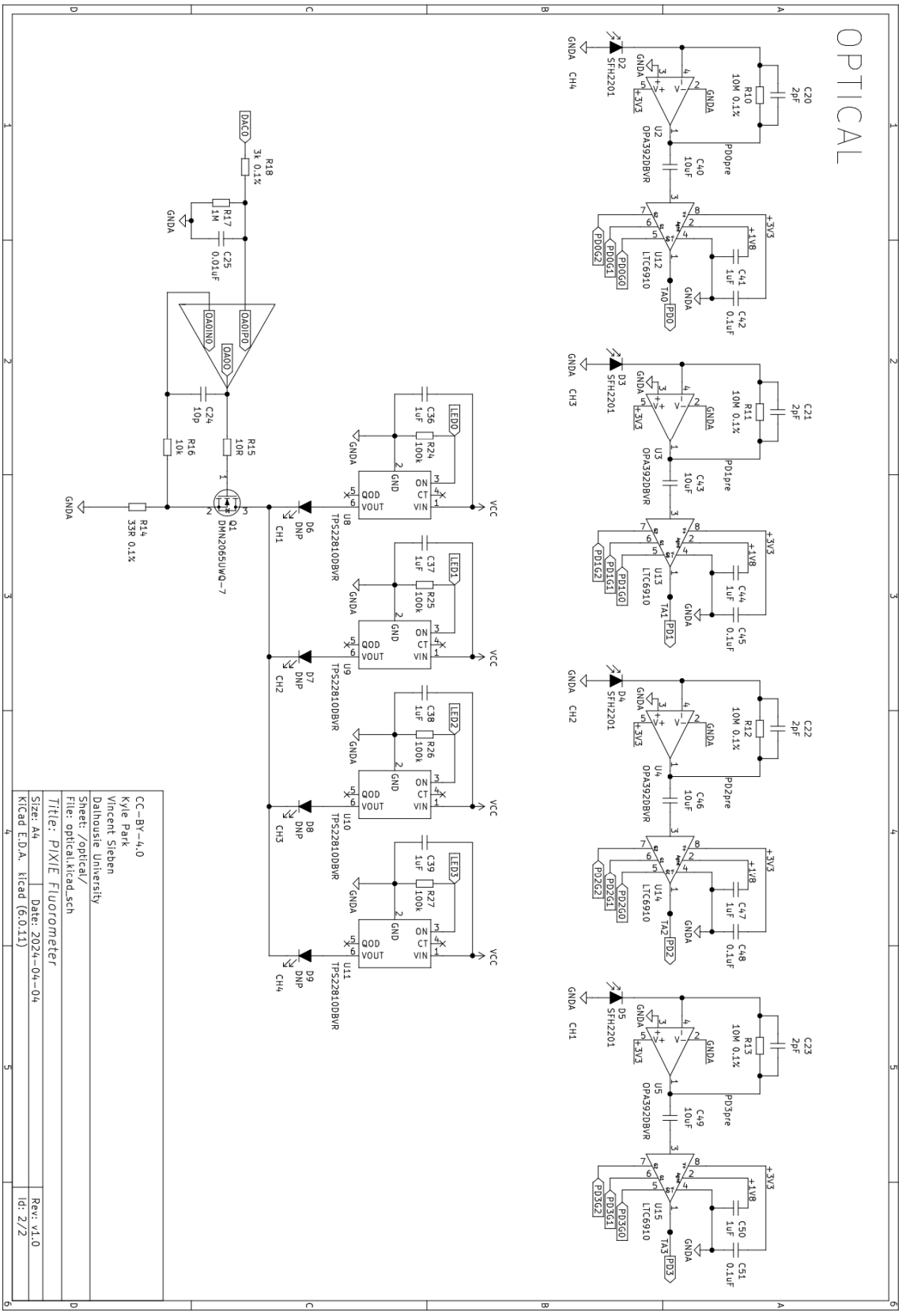


Figure 4.1-5 Bambu Studio Advanced settings for the Others tab.

4.2 Appendix B: PIXIE PCB Schematics




OPTICAL
CC-BY-4.0
Kyle Park
Vincent Sieben
Dalhousie University
Sheet: /optical/
File: optical.kicad_sch
Title: PIXIE Fluorometer
Size: A4
Date: 2024-04-04
Rev: v1.0
KiCad E.D.A. kicad (6.0.11)
Id: 2/2