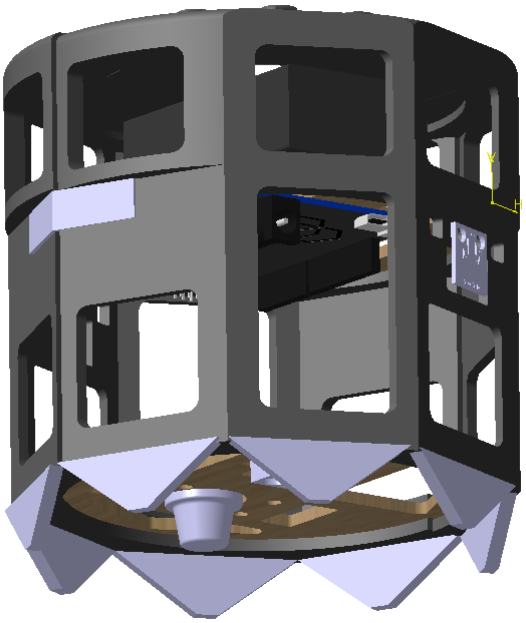
Agricultural Planting Drone Project

Final Project Report

Edwin Andrew
Kelvin Tam
Michael Latosa
Sobhan (Kian) Ebrahimi
Qingyang (Elena) Liang



AER817: Systems Engineering
Dr. Krishna Dev Kumar
Ryerson University
Fall 2018

Contents

List of Figures

List of Tables

1 Introduction

As technology advances, practical quadcopter/drone applications are steadily growing and is helping society in a variety of industries. Quadcopters excel in a number of areas such as cost, form factor, mobility, and safety compared to current or historical methods of completing certain tasks. One specific industry that can greatly benefit from its potential is "agriculture". With a fast growing population, the demand for food is rising, and ensuring that crops are grown under ideal conditions is critical. A quadcopter accompanied by a sensor package would be one such solution to meet such demands. A number of sensors and electronic hardware, alongside a rigid software will permit such applications. Drone use in the agricultural industry will help the industry grow and improve from a cost, efficiency, and performance perspective. An agricultural purposed quad-copter and sensor package would have the ability to monitor crop conditions or possibly find new areas which are ideal for agriculture, with the potential to aid in planting, pesticide/chemical deployment, and area mapping and marking in the future. Ensuring healthy crops is critical to meet the increasing global demand.

Figure 1: Importance of Crop Health

1.1 Mission Concept

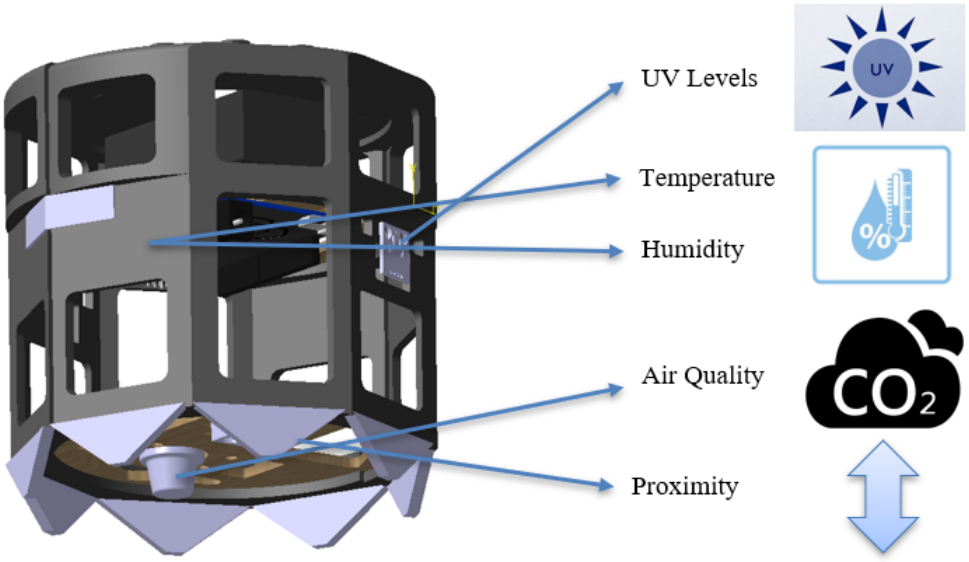


Figure 2: Enclosed View & Mission Concept

In an effort to improve the efficiency of crop production a sensor package which is to be mounted underneath a quadcopter has been carefully designed to record data based on a number of parameters such as humidity, temperature, air quality, UV levels, and proximity for safety and accuracy of data as seen in figure 2. To improve the effectiveness and efficiency of a crop, the health of the crop must be carefully maintained by the farmer through careful monitoring. Essentially, the sensor package would

utilize all of the collected data and compare them based on permissible limits. Once the data collected for the various parameters has been deemed good, bad, or marginal, the overall crop health would be presented to the pilot and farmer. At this point, from the detailed report produced, the farmer would be able to take the appropriate action to ensure the crop is flourishing in the most ideal conditions possible. Figure 3 outlines the various parameters and their respective limits.

Level	Good	Marginal	Bad
UV Level [mW/cm²]	100<UV<280	280<UV<315	315<UV<390
Temperature [°C]	T<18 or T>30	18<T<20 or 24<T<30	20≤T≤24
Humidity [%]	H<65 or H>90	65<H<80	80≤H≤90
Air Quality [ppm]	AQ<250	250<AQ<500	AQ>400
Proximity [cm]	P<40 or P>110	40<P<60 or 90<H<110	60<P<90

Figure 3: Parameters and Permissible Limits [1][2][3][4]

The following figure below presents the final working model of the sensor package from an exploded view. The general drawing of the entire payload can be found in the appendix in figure 36.

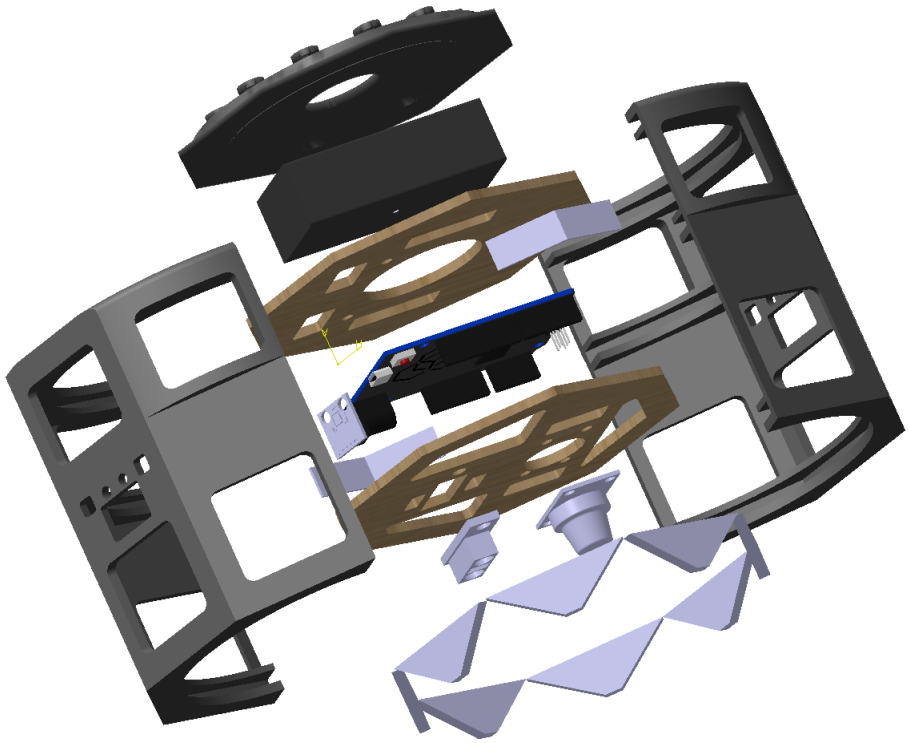


Figure 4: Exploded View of Latest Working Concept

The complete sensor package package will be composed of 13 main components, by which there are 4 primary structural parts, and 4 are electrical or power related, and 4 are sensors. The specific weight of each individual component and the total weight as a whole is listed in table 2.

2 System Objectives & Requirements

In this section, an overview of the constraints, objectives, and main requirements will be examined. A sensor package will be developed and deployed using a quadcopter. With a use of a Xbee transmitter and receiver data collected from the sensors can be sent to the main computer to be manipulated to present the information.

2.1 System Objectives

The main motivation for this project is to develop a sensor package that will be attached to a quadcopter. The sensor subsystem must be able to successfully collect data, and transfer it to the main computer where it must be presented to the client or pilot of the drone. *Woodland Tech Innovations'* main priority is to analyze air quality, temperature, humidity, sunlight, and proximity. With this information, the test area can be assessed with regards to crop growth health and potential.

- Monitor crop growth potential
  - Collect and analyze air quality data
  - Collect and analyze temperature and humidity data
  - Collect and analyze UV radiation data
- Present collected and analyzed atmospheric and surrounding data
  - Associate the various combinations of status' for each parameter with either "Good", "Marginal", "Bad" crop health
- Monitor, collect, and analyze proximity data from the ground
- Present collected and analyzed proximity data
  - Take the mean of the data within a specific time frame and display on pilot screen
  - Alert the pilot via pilot screen indicator
- Able to determine the health of a section of a specific crop

2.2 System Requirements

- The ability to maintain a safe an optimal distance above the ground for safety and data collection purposes
- Payload weight must be a maximum of 300g
- Dimensions of the payload should not exceed 20cm x 20cm x 20cm
- Payload can power on and transmit data to the ground station

2.2.1 Degree of Meeting Requirement Matrix

Based on the system requirements discused in the previous section, the integrated system degree of meeting requirement matrix is presented in the following table below.

Table 1: Integrated System Degree of Meeting Requirement Matrix

		Needs			
Req. #	**Req. Description**	Accurate Data	Optimal Alt.	Present Results	Asses Crop Health and Conditions
1	Fly a Safe Distance Above the Ground	x	x	x	x
2	Payload Must Meet the 300g Requirement		x		x
3	Payload Must Fit Within a 20 cm x 20 cm x 20 cm Box		x		x
4	Payload can power on and transmit data	x		x	x
Priority		3	2	3	1

2.3 Mass and Power Budget

2.3.1 Mass Budget

Given that the weight restriction of a payload to be mounted onto the quadcopter is 300 g, the integrated system as a whole and individual weights of components were measured, and can be seen in table 2. The weight of the sensor package was divided into three main categories; sensors, electronics, and structural. Overall, the payload resulted in a total weight of 291.00 g which is below the 300 g weight limit. This weight was mainly achieved by only keeping the necessary components on the package, along with reducing material on the structure where it was not needed or did not provide much structural support.

Table 2: Parts List & Actual Weight of Sensor Package

Part No.	**Part Name**	**Quantity**	**Weight (g)**
	Sensors		
1	Temperature & Humidity Sensor	1.00	3.00
2	Infared Proximity Sensor	1.00	4.00
3	Air Quality Sensor	1.00	5.00
4	UV Sensor	1.00	2.00
	Electronics		
5	Arduino Leonardo + Bread Board	1.00	54.00
6	Xbee Pro	1.00	4.00
7	9V Battery	1.00	70.00
8	Wiring	N/A	5.00
	Structural		
9	Casing	2.00	96.00
10	Electronics Mounting Plates	2.00	18.00
11	Drone Mounting Plate	1.00	25.00
12	Landing/Protecting Feet	8.00	2.00
13	Additional Weight (zip ties, tape, glue)	N/A	3.00
Total Weight			**291.00**

2.3.2 Power Budget

With regards to the power budget, the phase II report accurately predicts the power consumption. These calculations were determined using datasheets relative to each hardware component. Based on the *Phase I Report*, the Leonardo Arduino current and power was not determined because the prototype phase was not complete. Referring to the Leonardo Arduino datasheet, the current drawn is a function of the number of I/O pins used. With the completed electronics subsystem it was determined that 5 pins will be used for hardware components. Therefore, 200mA of current is drawn by the board. The total

fly time is 1 hour and 21 minutes based off of a 500 mAh battery. The power consumption summary can be seen in [Table 3] below.

Table 3: Power consumption chart based on individual components

Component	**Voltage (V)**	**Current (A)**	**Power (W)**
DHT-11 Sensor	5.0000	0.0025	0.0125
MQ-135 Sensor	5.0000	0.1600	0.8000
IR Sensor	5.0000	0.0050	0.0250
Sunlight Sensor	5.0000	0.0030	0.0150
Xbee Radio	3.3000	0.0050	0.0165
Leonardo Arduino	9.0000	0.2000	1.8000
TOTAL		**0.3678**	**2.6690**

2.4 Technology Readiness Level

Technology Readiness Levels (TRL) are a type of measurement system that is used to assess the maturity level of a project [5]. This method was introduced and used by NASA for the first time in 1980 and soon afterwards was adopted by other design and innovation institutions. In this section, the Agricultural Drone will be assessed using the TRL. The project was able to reach a maturity level of 6. In this stage, all the subsystems are fully integrated and the design is fully operational. The prototype was tested in a relevant environment (Kerr Hall Gymnasium) and demonstrate its performance.

3 Project Gantt Chart, Work Breakdown Structure and Work Packages

3.1 Project Gantt Chart

The gantt chart and its corresponding legend are presented in figure 5 and 6 below. The presented gantt chart establishes all major and internal project phases, project tasks, external and internal milestones, as well as internal deliverables for the project, along with the major deliverables.

Icon	Legend
	Major Project Phase
	Internal Project Phase
	Project Tasks
	External Milestones
	Internal Deliverables
	Internal Milestones

Milestone	Date
Submit Team Information Sheet	13-Sep-18
Proposal Report Submission	20-Sep-18
Proposal Presentation	21-Sep-18
Preliminary Design Review	10-Oct-18
Phase I Report Submission	12-Oct-18
Intermediate Design Review	31-Oct-18
Phase II Report Submission	1-Nov-18
Final Report Submission	29-Nov-18
Final Presentation	30-Nov-18
Showcase & Test Date	30-Nov-18

Figure 5: Project Gantt Chart Legend

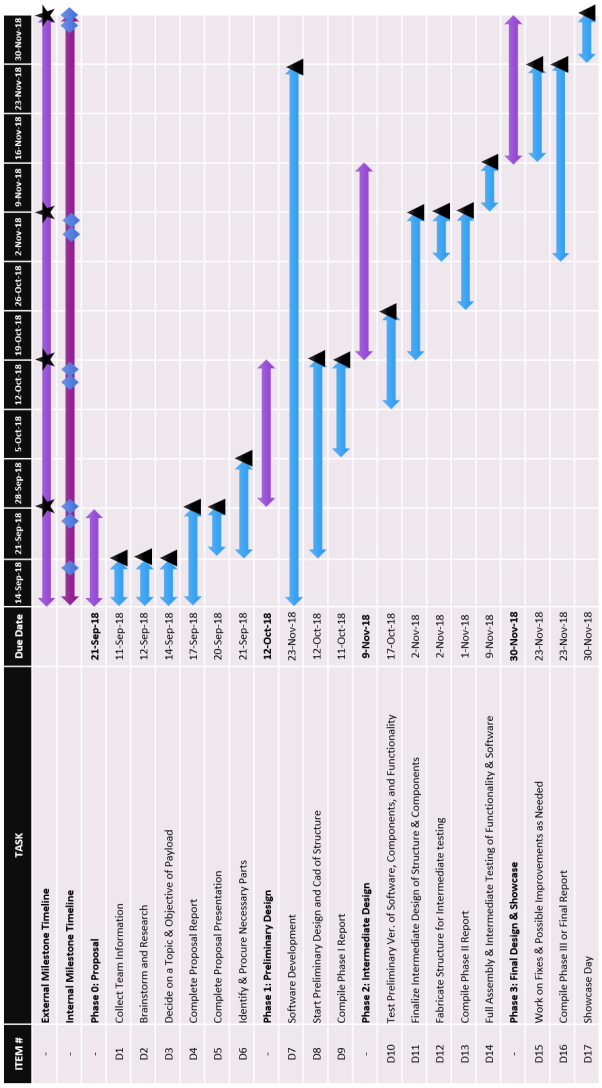


Figure 6: Project Gantt Chart

3.2 Work Breakdown Structure

The work has been divided into three main work packages; Project Management, Engineering & Design, and Simulation & Testing. The three work packages can be seen in the figure below. Individual work packages have also been broken down into its lower level structure.

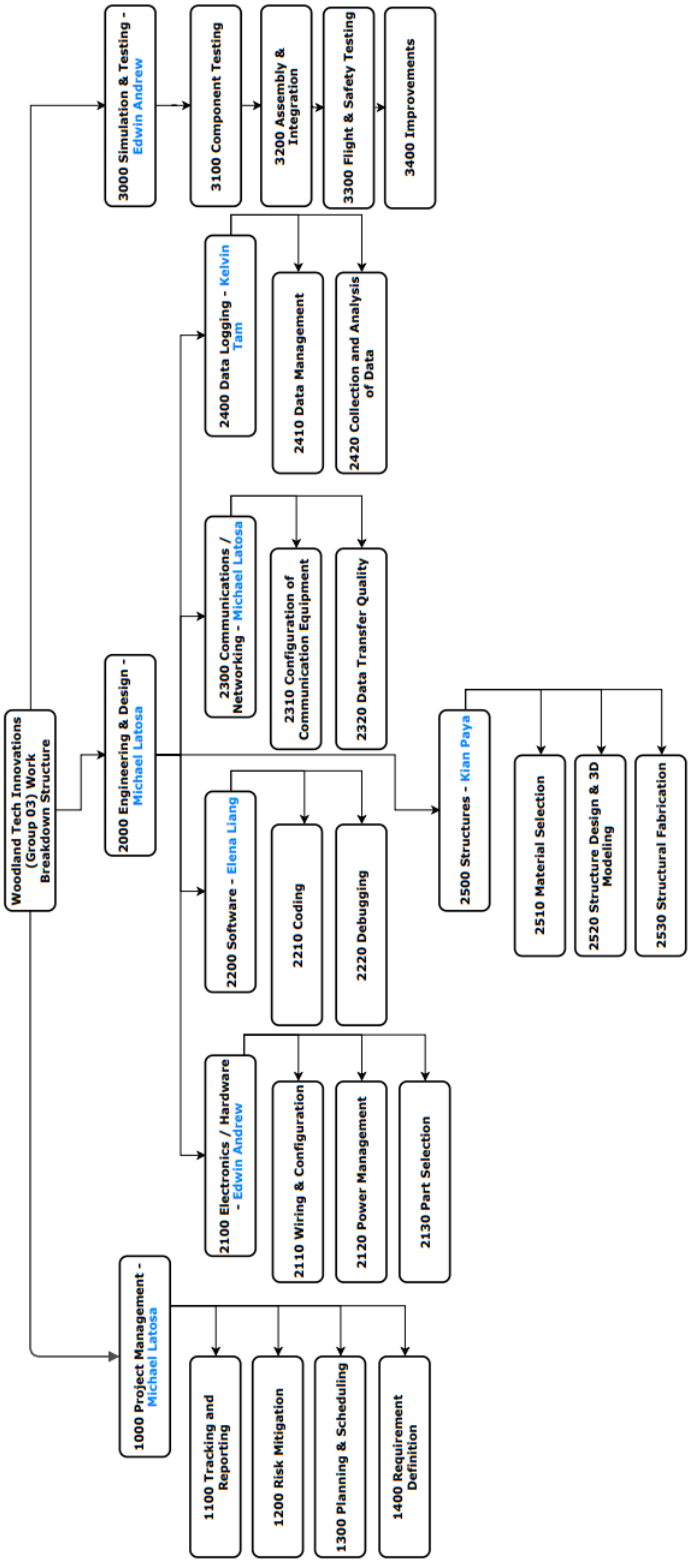


Figure 7: Work Breakdown Structure

3.3 Work Packages

The Project Management work package is managed by Michael Latosa, who also manages Engineering & Design, along with being responsible for communications and networking. With regards to the engineering and design teams, Edwin Andrew is in charge of the electronics and hardware, Elena Liang is in charge of the software, Kian Paya is responsible for structures, and Kelin Tam is in charge of Data Logging. The various work packages are presented in the following sections and are defined in terms of objectives, inputs, tasks, and outputs. The gantt chart is a critical tool used to provide visibility of how the project should proceed, and to determine if the project is on track to plan or not. In the event of missed deliverables and/or milestones, a gantt chart can be used to dictate if there will be a slip or risk to meeting the critical milestones and deadlines.

3.3.1 Project Management

Table 4 below outlines the the Project Management work package.

Table 4: Work Package 1000

Project: Agricultural Crop Health Monitoring Drone			
Work Pack Title:		**Project Management**	**WBS Ref: 1000**
Sheet	1 of 1		
Scheduled start:	September 10, 2018	Accountable Manager:	Michael Latosa
Scheduled end:	November 30, 2018	Resources	Edwin Andrew, Elena Liang, Kelvin Tam, Kian Paya
Estimated effort:	90 hours		
Objectives: - Project planning and scheduling - Project tracking and reporting - Risk mitigation - Requirement definition			
Inputs: - From Ryerson University			
Tasks: - Setup and arrange weekly meetings, milestone review, and intermediate review - Chair meetings and guide discussion - Produce meeting minutes - Ensure the team is completely aligned - Manage the project - Create agendas, plans,and schedules - Resolve internal issues			
Outputs/Deliverables: - Meeting minutes - Project plan - Progress tracking - Team coordination document - Proposal report package - Phase 1 report package - Phase 2 report package - Final report package - Presentation - Completed program			

3.3.2 Engineering & Design

Table 5 below outlines the the Engineering & Design work package.

Table 5: Work Package 2000

Project: Agricultural Crop Health Monitoring Drone			
Work Pack Title:		**Engineering & Design**	**WBS Ref: 2000**
Sheet	1 of 1		
Scheduled start:	September 10, 2018	Accountable Manager:	Michael Latosa
Scheduled end:	November 30, 2018	Resources	Edwin Andrew, Elena Liang, Kelvin Tam, Kian Paya
Estimated effort:	265 hours		

Objectives:

- Part selection and procurement
- Setup and configuration of electronics and components
- Power management
- Interfacing
- Programming & Debugging
- Ensure all equipment and data are operating and being managed properly
- Design, 3D model, and fabricate the structure of the sensor package
- Minimize effort, maximize results

Inputs:

- From component suppliers
- From data sheets
- BOM
- From engineering teams
- From flight test team
- From Ryerson University
- From programming and 3D modeling software's

Tasks:

- Choose components that maximize performance and is cost effective
- Procure, test, and program parts and components as needed
- Integration and wiring of all components
- Produce a circuit diagram
- Resolve any issues
- Interface with other teams
- Debugging
- Setup and configure all devices and systems
- Ensure data is transferred, processed and analyzed correctly
- Design, model, fabricate, and troubleshoot structure
- Ensure all components are mounted and can be integrated properly into the structure
- Produce professional part drawings

Outputs/Deliverables:

- Operational components and integrated system (All components harmonized)
- Circuit diagram
- Methodology structure for hardware
- Operating time limit
- All components operate together and unified under a single code
- Appropriate data is recorded and analyzed
- Completed rigid software
- Smooth and efficient data transfer
- Both raw and resulting data are stored, collected, analyzed and presented correctly
- Material or materials chosen for structure
- Chosen structural design
- Fabricated structure
- 100% compatible with mounting solutions and electronics

3.3.3 Simulation & Testing

Table 6 below outlines the the Simulation & Testing work package.

Table 6: Work Package 3000

Project: Agricultural Crop Health Monitoring Drone			
Work Pack Title:	**Simulation and Testing**		**WBS Ref: 3000**
Sheet:	1 of 1		
Scheduled start:	September 10, 2018	Accountable Manager:	Edwin Andrew
Scheduled end:	November 30, 2018	Resources	Elena Liang, Kelvin Tam, Kian Paya, Michael Latosa
Estimated effort:	30 hours		
Objectives: - Ensure all components are operational - All parts including structure and electronics are correctly assembled - Completed package is flight tested as a whole - Ensure safety conditions			
Inputs: - From components - From structural components - From quadcopter - From testing equipment - From tools for assembly - From engineering teams			
Tasks: - Ensure components are operational - Install all parts and components meant to be in the sensor package and on the quadcopter - Electronics are properly installed on the structure - Set up testing equipment - Conduct safety checks - Flight test the package on the quadcopter - Record observations for possible improvements			
Outputs/Deliverables: - Flight tested sensor package - Product and equipment safe for mission profile - Possible improvements - Validation of capabilities			

4 System Overview

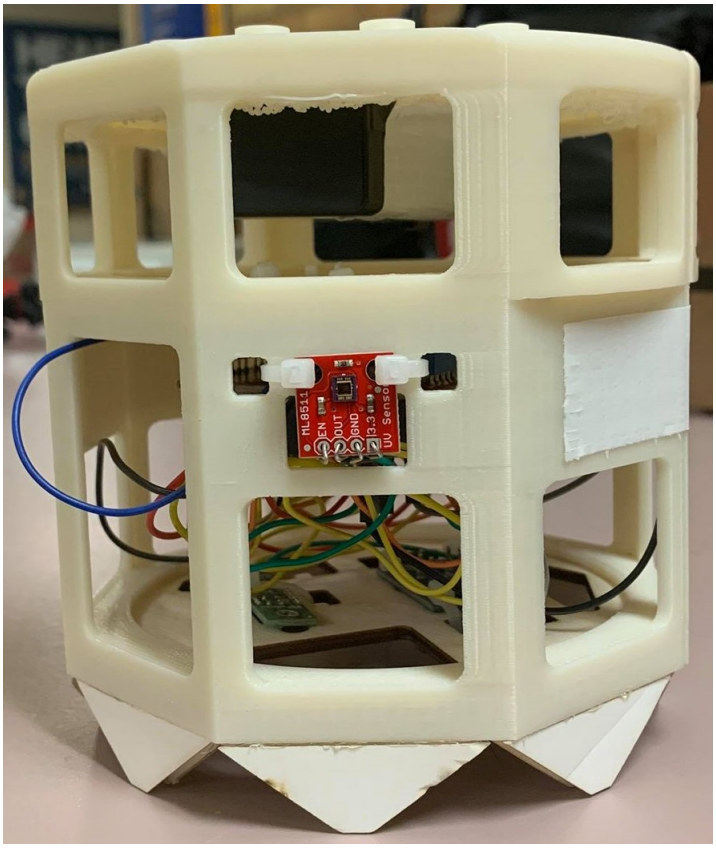


Figure 8: Complete and Integrated Actual Payload

4.1 Payload Subsystem

The payload subsystem is concerned with the sensors and instruments used for communication and data logging. The sensors are expected to work together in a harmonious way to maximize the utility of the package in a fully functional subsystem. They include: Temperature and humidity, IR proximity, Air Quality and Light Detection which are powered by a 9-volt battery. Arduino board, Xbee transmitter, wiring and Micro SD card are also part of the payload providing the interface between hardware and data logging. Data captured by the sensors will be transmitted to the ground station for further processing. The interface for this package is described in N2 diagram in more details.

4.2 Hardware

As earlier described in this report, the payload hardware consists of 4 different sensors, an Arduino board, Xbee transmitter along with wiring and an SD card for storing sensor data. Table 7 gives a brief description on each of these components. The descriptions of sensors is also provided in Table 8.

Table 7: Component Descriptions

Component	Description
Leonardo Arduino with Xbee shield	This is the microcontroller used for the sensor package. The integrated Xbee shield makes it possible for the transmitter to be directly attached.
Micro SD shield	This component is used to collect data and storing the data. In case, the livestream is disconnected for any transmitter related issues, the Micro,SD can be used to recover the data.
Xbee 1 mW Trace Antenna	This antenna is used to receive commands from ground station. It can provide a range of more than 30 meters as tested in the lab.

4.3 Block Diagram

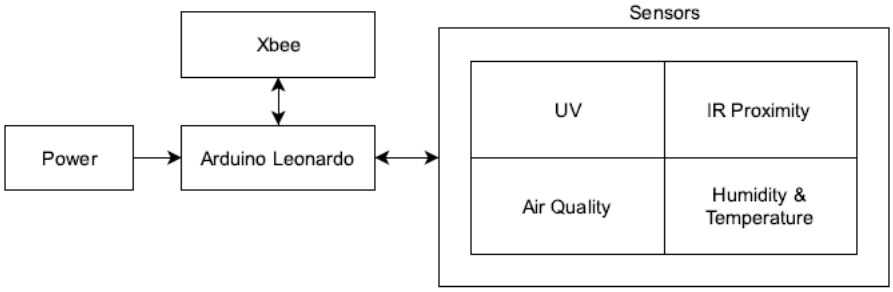


Figure 9: Payload Hardware Block Diagram

4.4 Structures and Mechanisms

Structures and mechanisms is essential for providing both the structural protection to the internal payload components and an enclosure that houses the subsystems tightly and securely. This section shall outline the design iteration process and the materials considered for the final product.

4.4.1 Requirements

The objective as mentioned above outlines the basic requirements of structural and mechanisms. The following list are detailed requirements that the structure must achieve.

- The final weight of structures and mechanisms shall be less than 300 grams
- The dimensions of the structure shall fit within a 20 x 20 x 20 cm cube
- The structure shall allow user to easily attach/detach & access the payload components within
- The external structure shall encase and provide proper protection for the electronics within

4.4.2 Design

The design of the structural component is resulted from redesigning a total of four iterations.The structural enclosure can be divided into four components. The first is the quad-copter mounting component. This plate is responsible for locking onto the provided mechanism such that the payload is secured to the copter. Next are holed internal plates that provides surface area for the electronics to be fastened onto. The third component is the external casing that houses and protects the locking plate and electronics plate. Lastly are the extra protective components. These provide extra security and shock protection in the event of crashes.

The initial designs outlined from phase report one attempts to use one maximum surface area for all internal electronics. The top and bottom plate had enough clearance for the Arduino & Xbee chip and breadboard. Since the sensor package includes sensors that must interact with the surrounding air, the sides was designed with an open concept. Flaws in this design lie in the single supporting plate and an almost complete open enclosure design. In the event of the plate cracking, the sensor packages will fall out of the enclosure. Furthermore, due to the lack of side protection, objects may reach the internals and knock sensors out of the payload. Lastly, due to the single plate design, individual sensors are less accessible as the entire package must be disassembled to acquire the desired one.

4.5 Command and Data Handling (C & DH) Subsystem

The major function of a Command and Data Handling Subsystem is to perform on-board operations and internal communication. The C&DH system consists of an Arduino Leonardo, which controls the operation of the quadcopter payload system.

4.5.1 Requirements

The objective of the command and data handling subsystem is to provide operation commands to various subsystems and data management. The C&DH subsystem should be able to perform following tasks:

- The system shall control the operation of payload sensors system by execute the command from ground station
- The remotely collected data shall be transmitted back to ground station for analysis
- The data and software shall be storage in allocated memories and location
- If any error occurred, the system shall be able to detect the failure
- The system shall weigh less than maximum allowable limits

4.5.2 Hardware

The section describes the hardware design in C&DH system. The heart of the C&DH system is the Arduino Leonardo, which runs the software responsible for managing the on-board operations. A brief breakdown of the C&DH system is listed as following:

- Arduino Leonardo consists a microcontroller and three built-in memories: EEPROM, Flash, and SRAM
  - Microcontroller is responsible for receiving and executing commands from ground operators
  - Flash memory is used to store the main operating software
  - SRAM(temporary memory) will be used to run the software and stored the measured values from the payload sensors
  - EEPROM acts as permanent memory and will be used to stored a copy of the software
- In order to communicate with the ground station, the Arduino Leonardo must be connected to the communication subsystem
  - By using the transmitter, the microprocessor sends packets of data to the ground station
- The system must be linked to power system to obtain power supply for managing on-board operations
- In order to generate crop monitoring report, the payload system consists of four sensors to collect desired data:
  - Air Quality sensor: measures environmental air quality
  - Infrared Proximity sensor: measures distance from ground to the quadcopter payload system
  - UV sensor: detects the intensity of incident ultraviolet radiation
  - Humidity&Temperature sensor: collects air humidity and temperature data in time manner

4.5.3 Software

The C&DH system software controls the operation of multiple sensors installed on the payload system. The Arduino IDE was used to develop the C&DH system software.The designed software performs following function:

- Read data from sensors
- Save data into the on-board SD card as CSV file

- Transmit reading of each sensor to ground station
  - Temperature
  - Humidity
  - Heat Index
  - Air Quality
  - IR Distance
  - UV Level
  - UV Intensity
- Transmit the IR distance condition to ground station. The range of IR sensor is 0 to 150 cm.
  - IR distance between 0 to 50 cm: Fly to low
  - IR distance between 50 to 100 cm: Fly ok (Ideal Range)
  - IR distance between 100 to 150 cm: Fly to high

The source code of C&DH system is shown in **Appendix A**. The real time reading is displayed on the XCTU console log (shown in figure 10).

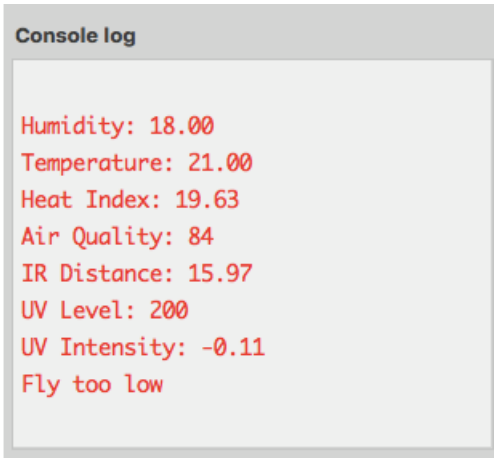


Figure 10: XCTU Console Log

4.5.4 Block Diagram

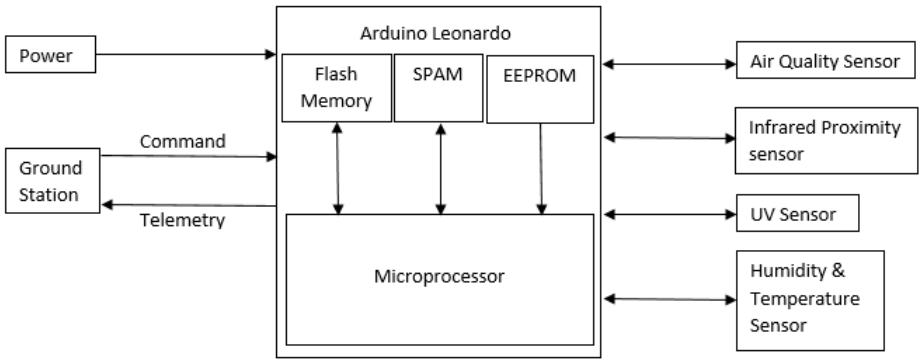


Figure 11: Block Diagram of Command and Data Handing Subsystem

4.6 Power Subsystem

4.6.1 Requirements

The purpose of having a power subsystem is to supply a continuous source of electrical power to all the hardware components. Alongside this, the power subsystem will control and distribute the electrical flow accordingly, relative to average and peak loading of the unit. The Arduino Leonardo requires a input voltage of 7-15V with a maximum tolerance of 6-20V. The main board is required to ensure that the supply voltage is consistent so that it can supply power to each component appropriately. This will require a voltage regulator which is already built into the system to regulate the voltages so that board can manage it aptly. The main power supply is a 9V battery so it must be converted into a lower voltage. The regulator will act as a buffer in regards to protecting the components from any damages. Risks are inherent in all activities. Battery leakage is a potential hazard that may occur. To prevent this, a battery holder with pressure contacts will be used to safely transmit power and encase the battery to prevent damage.

4.6.2 Hardware

The power subsystem consists three main components:

1. 9V battery
2. Battery Holder w/ Switch
3. Arduino Leonardo Board

These components are the fundamental blocks to providing the necessary power for the whole payload system. Therefore, [Figure 12] presents the block diagram for all the components that are interconnected to the power subsystem. These blocks are also necessary for the power subsystem because they all work together in unison.

4.6.3 Block Diagram

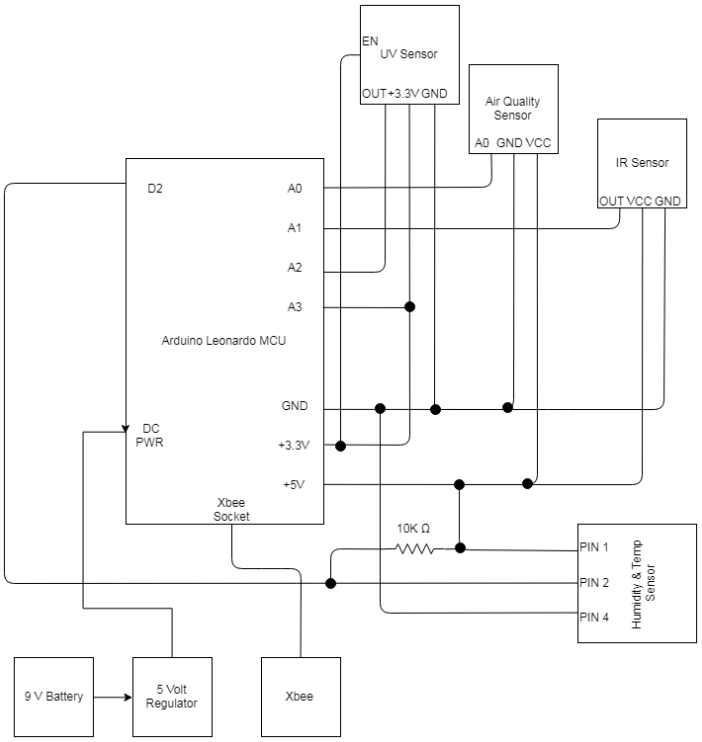


Figure 12: Power Management Block Diagram

4.7 Communication Subsystem

4.7.1 Requirements

The primary function of the communication subsystem is to provide the transmission, reception, and conditioning of the mission data to ground base for data processing. The communication subsystem is a medium for human to system interface and consists of the Xbee receiver, and transponder. This specific subsystem has the following requirements which are listed below:

- The subsystem must receive the collected data from the electronics and hardware
- The subsystem must receive collected input from the ground station
- Input data collected from the ground station must be transmitted to the electronics and hardware to perform the specific tasks that have been outlined by Command and data handling
- Data collected by the electronics and hardware during the the mission profile must be transmitted back to the ground station where command and data handling would process, store, and report on the collected data
- Constant and consistent communication must be achieved between the payload system housing the power subsystem, and the ground station where command and data handling occur
- Collected data and input data being transmitted and received must conditioned to ensure it arrives at its designated destination and still retains the entirety of the data

4.7.2 Hardware

The communication subsystem is composed of two primary components:

- Xbee Series 1 Transmitter/Receiver
- Xbee USB Dongle

The Xbee transmitter/receiver permit the interaction between the ground station and the microprocessor through collected data and input commands. One Xbee transmitter/receiver is connected to the Arduino Leonardo, while another is connected to the Xbee USB dongle. The USB dongle is a means of allowing the the ground station to be connected to the communications network.

4.7.3 Block Diagram

The communication subsystem is presented in the following block diagram

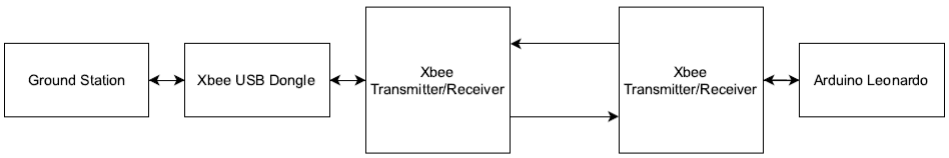


Figure 13: Communication Subsystem

4.8 Ground Station

The primary function of the ground station is to serve as the main source of operations for quad-copter and its attached payload system. The ground station is typically where the quad-copter is controlled, and where commands are sent to the payload system and data is received

4.8.1 Requirements

As the ground station essentially acts as the "brains" of the entire operation, it has the following requirements:

- Must control and monitor the progression of the operation
- Must be able to transmit and receive data to the quad-copter and payload system
- Must work with command and data handling
- Must work in conjunction with the communications subsystem

4.8.2 Hardware

The ground station is composed of the following primary components:

- Computing Station (Computer)
- Quad-copter controller
- Communications Equipment (Antenna)
- Human Input Device (Keyboard)
- User

The heart of the ground station is composed of the user, the computing station, and quad-copter controller as they are the main drivers for the success of the operation or mission profile. The human input device and communications equipment are just a medium by which the user can tell what the computing station to do and by which the computing system and controller can interact with the quad-copter and payload. All these various components are located in the same location and is what make up the ground station.

4.8.3 Block Diagram

The ground system is depicted in the following block diagram

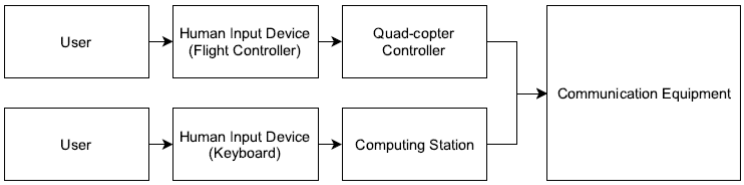


Figure 14: Ground Station Block Diagram

4.9 Bill of Materials (BOM)

Table 8: Payload Subsystem Bill of Materials

Part Name	Model	Retailer	No. of Parts Required	Price (CAD)
Temp & Humidity Sensor	SPEDE-010111	CREATRON INC	1	$11.00
Infrared Proximity Sensor	OPTDA-015000	CREATRON INC	1	$17.25
Air Quality Sensor (20 to 150 cm)	GASSE-001135	CREATRON INC	1	$10.00
Light Sensor	OPTIC-012705	CREATRON INC	1	$17.50

Table 9: Power Subsystem Bill of Materials

Item No.	QTY	Component	Cost
1	1	9V Battery Holder w/ Switch	$7.70
2	1	9V Power Adapter	$17.00
3	19	M/M Jumper Cables	Free
4	3	M/F Jumper Cables	Provided
5	1	Energizer 9V Battery	$4.00
TOTAL			**$28.70**

Table 10: Communications Bill of Materials

Item No.	QTY	Component	Cost (CAD)
1	1	Xbee Pro Transmitter/Receiver	$37.97
2	1	Xbee USB Dongle	$25.95
TOTAL			**$116.07**

Table 11: Command & Data Handling Bill of Materials

Item No.	QTY	Component	Cost (CAD)
1	1	Arduino Leonardo	$22.37
2	1	DUT-11 Sensor	$11.00
3	1	MQ-135 Sensor	$10.00
4	1	IR Sensor	$17.25
5	1	UV Sunlight Sensor	$17.50
6	1	Xbee Pro	$37.95
7	22	Cables	N/A
TOTAL			**$116.07**

Table 12: Structures Bill of Materials

Item No.	QTY	Component	Cost (CAD)
1	1	ABS Plastic (3D Printing Material)	$20.00/kg
2	1	Foam Core	$1.00/sheet
3	1	3mm MDF	$10.00/sheet
4	1	Velcro Fastener	$10.00/roll
5	1	Cable Ties (100 Pack)	$4.00
6	10	Bolts, Nuts & Pins	$2.00
TOTAL			**$53.11**

Table 13: Ground Station Bill of Materials

Item No.	QTY	Component	Cost (CAD)
1	1	Computing Station	$1450.00
2	1	Flight Controller	$100.00
3	1	Keyboard	$50.00
4	1	Communication Equipment	$130.00
5	1	User(1.5 hr fly time)	$22.50
TOTAL			**$1867.57**

Table 14: Master List Bill of Materials

Item No.	QTY	Component	Cost
1	1	SPEDE-010111 Sensor	$11.00
2	1	OPTDA-015000 Sensor	$17.25
3	1	GASSE-001135 Sensor	$10.00
4	1	OPTIC-012705 Sensor	$17.50
5	1	9V Battery Holder w/ Switch	$7.70
6	1	9V Power Adapter	$17.00
7	19	M/M Jumper Cables	-
8	3	M/F Jumper Cables	-
9	1	Energizer 9V Battery	$4.00
10	1	Xbee Pro Transmitter/Receiver	$37.97
11	1	Xbee USB Dongle	$25.95
12	1	Arduino Leonardo	$22.37
13	1	ABS Plastic (3D Printing Material)	$20.00/kg
14	1	Foam Core	$1.00/sheet
15	1	3mm MDF	$10.00/sheet
16	1	Velcro Fastener	$10.00/roll
17	1	Cable Ties (100 Pack)	$4.00
18	10	Bolts, Nuts & Pins	$2.00
Total			**$223.85**

5 System Interface

5.1 System Interface Diagram [Data and Power]

In terms of data and power and their connection/relation to one another, the system interface diagram can be found in Figure 15 below. The diagram establishes that it provides power to three main systems/components: communication subsystem, arduino micro-controller, and the payload consisting of all the sensors.

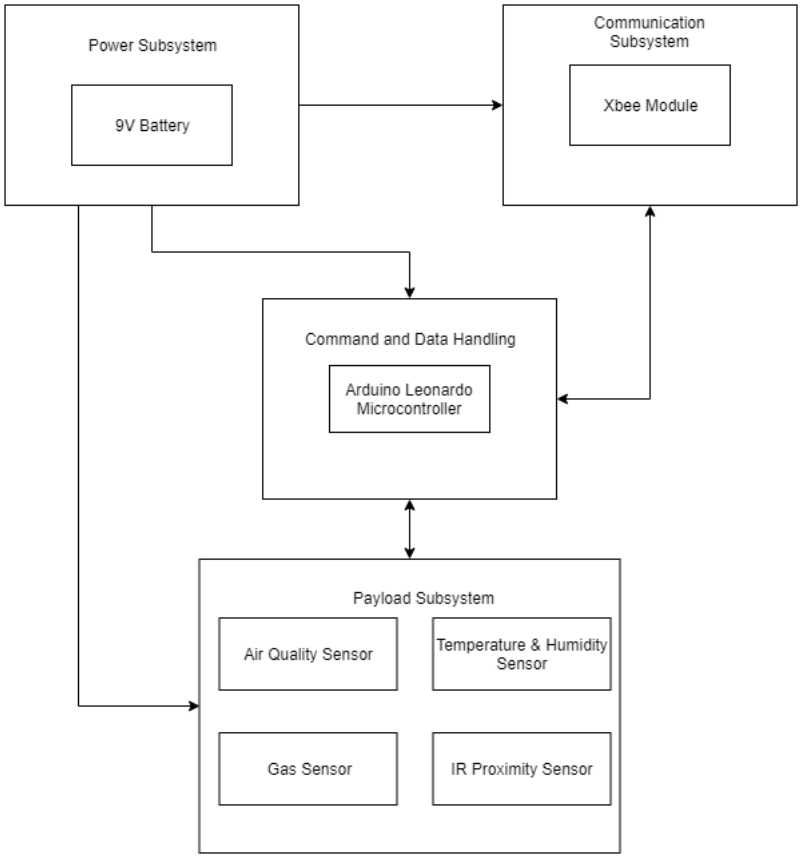


Figure 15: Data and Power System interface block diagram

5.2 System Interface between Subsystems

Table 15 illustrates the system interfaces between the six subsystems of the overall system. It also establishes the purpose each subsystem serves in relation to the other subsystems.

Table 15: System Interface

System Interface	Payload	Power	Communication	Command & Data Handling	Structures	Ground Station
Payload	-	Battery in the payload provides the power to the package.	Xbee transmitter in Aurduino board transmits the data.	Data is recorded and further analyzed from the livestream and SD card.	Structure is able to withstand the weight of the payload (300g).	The data from the sensor payload is sent to the ground station.
Power	Power is distributed to the sensor package.	-	Communications through Xbee is possible through power supply.	Command & Data Handling uses power to process data. Power is distributed.	Structures create a space for safe transition of power.	Can be notified of any malfunctions in the system at distance due to power being available in the sensor package.
Communication	Payload provides communication and transmit of data.	Xbee transmitter receives power for communication.	-	The processed data is communicated through Xbee and SD card.	Communications component are safely contained in the structure.	Ground station is in the range of communication components (SD/Xbee transmitter).
Command &Data Handling	Data is received from sensor payload and is processed.	Receive power for data handling components.	Data is recorded and made available through communication components.	-	Components are secured to the structure for a proper transmis.sion of data.	The data is processed and used in the ground station.
Structures	Proper air circulation and enough space is provided by structures to the sensor payload.	Battery holder in structures provides a fail-safe feature in the power supply.	Structure gives enough room for communication components.	Data processing is possible through a secure design provided by a sturdy structure.	-	No electrical interface between structures and ground station.
Ground Station	The data from payload is accessible in Ground Station.	The drone can be tracked in Ground Station in case of power outage.	Ground Station constantly communicate and monitors the livestream of data from communication components.	The data result from command and data handling subsystem is made available in Ground Station.	No electrical interface between ground station and structure subsystem.	-

6 Testing

6.1 Payload Subsystem

In regards to testing the sensors for the payload, the main method for analysis is the debugging technique. This technique requires the system to be broken down into smaller components and tested separately. Each of the sensors were wired individually to check for any short circuits. After each of the components were tested individually they were combined together. One by one, each sensor was implemented onto the breadboard to check for validity. This method allows for the detection of faults in the design in a way that conveniently allows them to be fixed using isolation. During the prototyping phase it is much easier to remove and replace components because nothing is permanently installed.

6.2 Structures and Mechanism

Structures testing is separated into two components. Prior to flight testing and component fitting, a preliminary CATIA stress analysis is done with the external casing to ensure proper protection. In the event of drone failure and detach the mounting unit, the payload will potentially be damaged due to ground impact. To simplify this stress analysis, half of the casing was modeled and a similar density material to the prototype plastic is mapped to it. Specifically, the impact analysis is mainly focused on the casing's lower component as it has the highest potential for damages. Force exerted on the aforementioned locations mimics reaction force from ground impact due to gravity.There were three impact scenarios listed as follow:

- Impact on surface

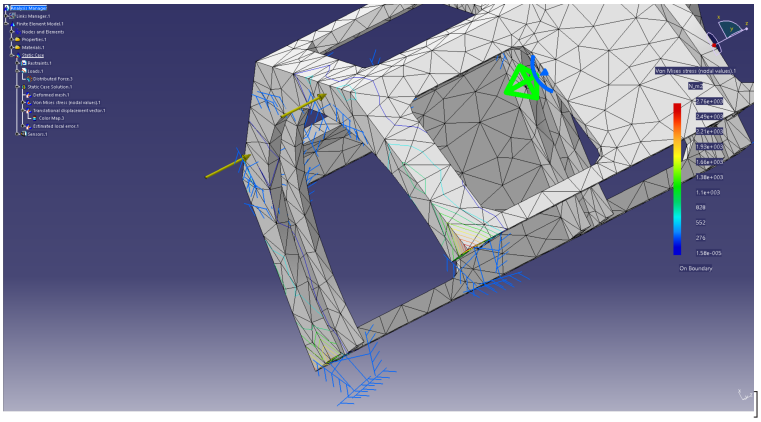


Figure 16: Surface Stress on impact)

An even force distribution is exerted on the bottom surface and the maximum stress is located on the corners of the casing. There were no evident physical deformation.

- Impact on an edge

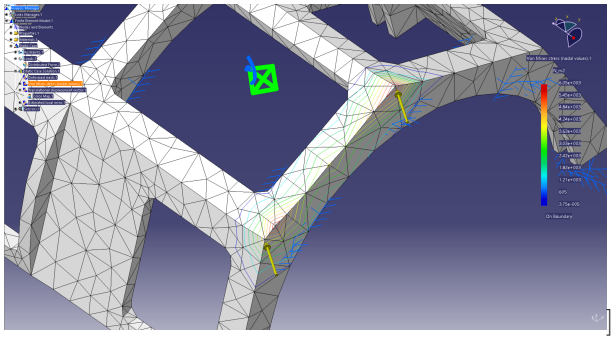


Figure 17: Edge Stress on impact)

Similar to the distributed force shown on the surface, the edge demonstrated little to no deformation on impact. Once again, the maximum stress is located at the corners of the edge.

- Impact at a corner

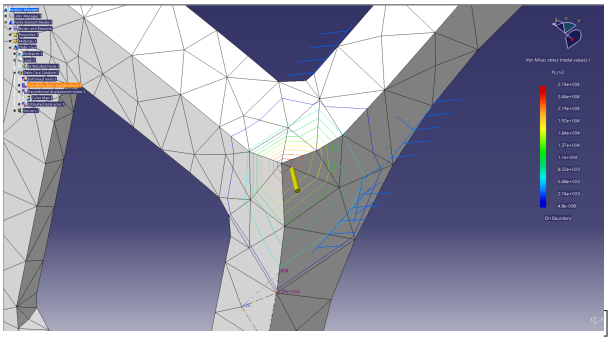


Figure 18: Corner Stress on impact)

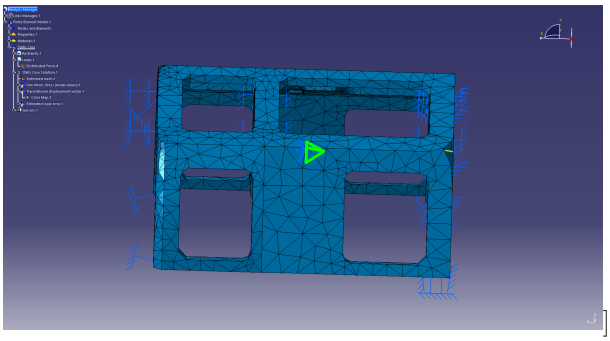


Figure 19: Corner Deformation on impact)

The corner impact is the most detrimental as shown by the dent seen in Figure 19 corner deformation. Of all the maximum Von Mises Stress magnitude, the corner has the highest stress value of $2.74 \times 10^4 N/m^2$. Though there is deformation, it is not enough for the casing to crack.

After a preliminary stress analysis, component fitting is essential to ensure that the system can be integrated properly together. The internal plates are precision fitted first with the electronics and then into the slots of the casing. The top plate is also mounted in a similar manner. To ensure the security of the top plate, small amount of resin was applied onto the edges before fitting.

6.3 Command & Data Handling Subsystem

The testing of C&DH subsystem software has been divided into three sections based on its functional requirements: read data from each sensors, transmit data to ground station through Xbee, and save data to the on-board SD card. All of the functions have been tested individually during ground testing. By connecting the Arduino board with installed sensors to the computer, the serial monitor in Arduino IDE can display the collected data. The outputs in serial monitor were used to identify any logical bugs and incorrect pin assignment.In order to ensure the sensor reading can be transmitted into the ground station, the Arduino board was connected to the computer via Xbee.If the data can be transmitted into the ground station via Xbee, the data can be seen in the XCTU console log. Since the data shown in XCTU console log should be same as the Arduino board is wired connected to the computer, any bugs can be identified by comparing the data in console log and serial monitor.Before saving the data into SD card, it is important to check the on-board SD has been properly installed. The source code contains the error handling to inform the user if the on-board SD cannot be initialized. If the SD card has been installed, all the reading should be saved in a CSV file on the SD card. By plugging in the SD card onto the computer, the results in CSV file were used to compare with the reading displayed in the serial monitor to identify error.Beside testing on the functionality of this software, the correctness of the sensors reading has been confirmed with the similar parameters obtained from other sources. For example, the collected temperature reading was compared with the reading showed on a thermometer.

During the flight test, all of the functional requirements of C&DH system have been met. The software was able to operate the sensors and collect their reading. All the required reading were successfully sent to the ground station via Xbee (shown in figure 20). The on-board SD card has been checked after the flight test. The testing data has been saved into the CSV file on the SD card (shown in figure 21).

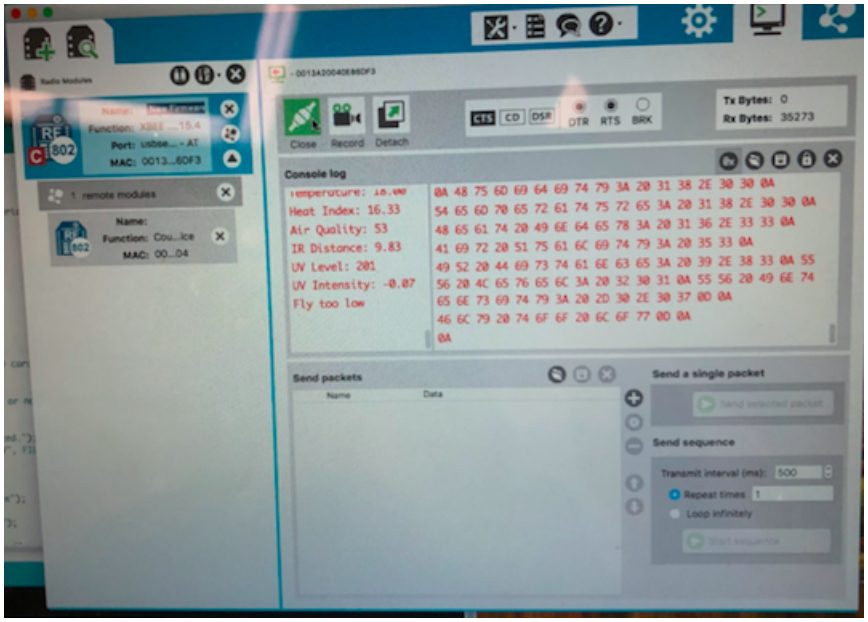
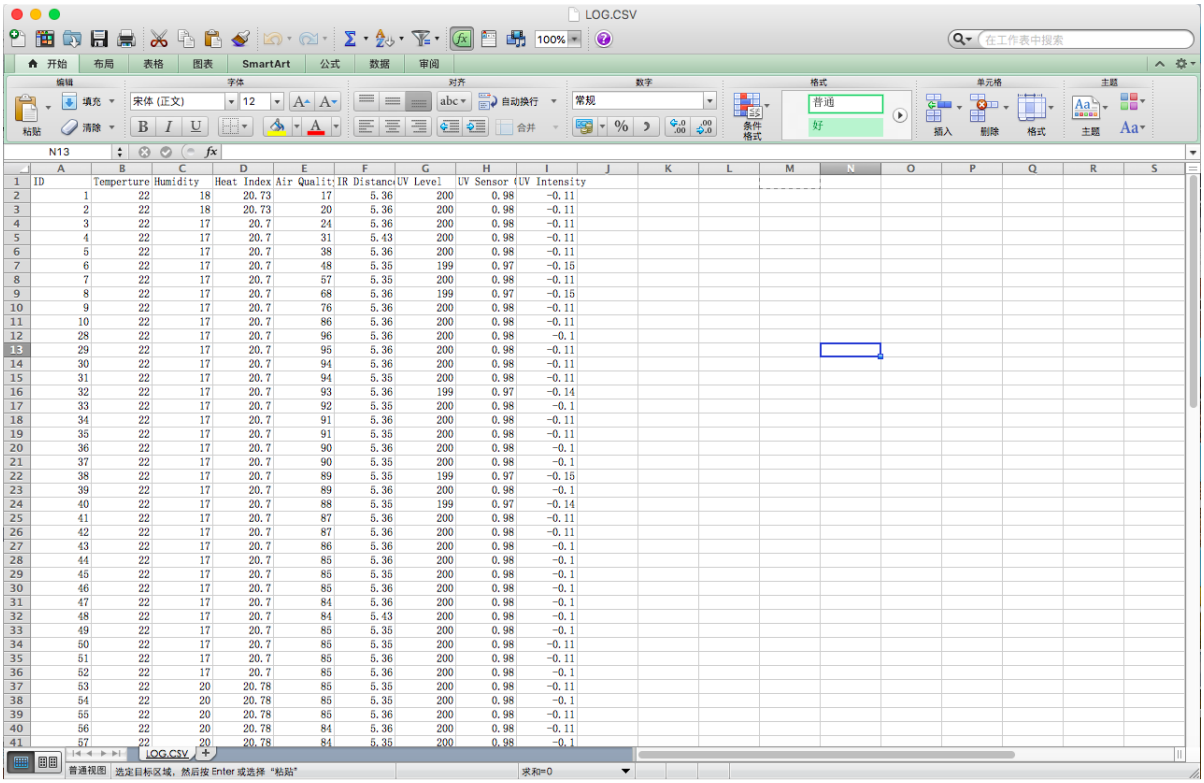


Figure 20: XCTU Console Log (During Flight Test)



ID	Temperture	Humidity	Heat Index	Air Quality	IR Distance	UV Level	UV Sensor	UV Intensity
1	22	18	20.73	17	5.36	200	0.98	-0.11
2	22	18	20.73	20	5.36	200	0.98	-0.11
3	22	17	20.7	24	5.36	200	0.98	-0.11
4	22	17	20.7	31	5.43	200	0.98	-0.11
5	22	17	20.7	38	5.36	200	0.98	-0.11
6	22	17	20.7	48	5.35	199	0.97	-0.15
7	22	17	20.7	57	5.35	200	0.98	-0.11
8	22	17	20.7	68	5.36	199	0.97	-0.15
9	22	17	20.7	76	5.36	200	0.98	-0.11
10	22	17	20.7	86	5.36	200	0.98	-0.11
28	22	17	20.7	96	5.36	200	0.98	-0.1
29	22	17	20.7	95	5.36	200	0.98	-0.11
30	22	17	20.7	94	5.36	200	0.98	-0.11
31	22	17	20.7	94	5.35	200	0.98	-0.11
32	22	17	20.7	93	5.36	199	0.97	-0.14
33	22	17	20.7	92	5.35	200	0.98	-0.1
34	22	17	20.7	91	5.36	200	0.98	-0.11
35	22	17	20.7	91	5.35	200	0.98	-0.11
36	22	17	20.7	90	5.36	200	0.98	-0.1
37	22	17	20.7	90	5.35	200	0.98	-0.1
38	22	17	20.7	89	5.35	199	0.97	-0.15
39	22	17	20.7	89	5.36	200	0.98	-0.1
40	22	17	20.7	88	5.35	199	0.97	-0.14
41	22	17	20.7	87	5.36	200	0.98	-0.11
42	22	17	20.7	87	5.36	200	0.98	-0.11
43	22	17	20.7	86	5.36	200	0.98	-0.1
44	22	17	20.7	85	5.36	200	0.98	-0.1
45	22	17	20.7	85	5.35	200	0.98	-0.1
46	22	17	20.7	85	5.36	200	0.98	-0.1
47	22	17	20.7	84	5.36	200	0.98	-0.1
48	22	17	20.7	84	5.43	200	0.98	-0.1
49	22	17	20.7	85	5.35	200	0.98	-0.1
50	22	17	20.7	85	5.35	200	0.98	-0.11
51	22	17	20.7	85	5.36	200	0.98	-0.11
52	22	17	20.7	85	5.36	200	0.98	-0.1
53	22	20	20.78	85	5.35	200	0.98	-0.11
54	22	20	20.78	85	5.35	200	0.98	-0.1
55	22	20	20.78	85	5.36	200	0.98	-0.11
56	22	20	20.78	84	5.36	200	0.98	-0.11
57	22	20	20.78	84	5.35	200	0.98	-0.1

Figure 21: CSV file in SD Card (Flight Test)

6.4 Power Subsystem

Prior to starting the installation of the sensors, the calculation for power drawn was done. The data sheets provided by the manufacturers provide adequate information about the power characteristics. The calculation gave an estimate of how much power can and is drawn by the components. This gave a good idea of how the payload will react to each component and the estimated fly time with a 9V battery. The testing procedure is very critical because power components can be damaged if they are not connected correctly or too much current is drawn. Therefore, theoretical values for current drawn was done prior to prototyping. Refer to [Table 3] for power consumption.

6.5 Communication Subsystem

The primary functional requirements of communication subsystem is to provide the data transmission and reception between the wireless payload system and the ground station via Xbee. The communication system only contains two components, Xbee transmitter and Xbee USB dongle. The testing had been divided into two main parts. The first part was to ensure the correct configuration of the Xbee transmitter and Xbee USB dongle by using XCTU. Under the correct configuration, the Xbee transmitter can be discovered and linked to the Xbee USB dongle. After the Xbee had been set up properly, the second part of testing is to ensure the data can be transmitted via Xbee. The Xbee communication code obtained from the lab manual had been used to test on the data transmission and reception functions between two connected Xbee modules.

6.6 Ground Station

Based on the requirements, the ground station should be able to allow users control and monitor the payload system operation wirelessly during the drone is running. In order to test the control on the Xbee connection, the serial connection has been opened and closed. If no error in this control, the data is received and displayed on the console log continuously while the serial connection has been opened. And, the data reception is stopped and no new data shows on the console log when the serial connection has been disconnected. When the serial connection has been connected, the console log should allow users

to monitor the progression of the operation. Any bugs can be identified if the data cannot be shown on the console log in a designed format. Also, the designed system has no any required user inputs from the ground station. The error handling has been considered when the input message has been sent by users accidentally.

6.7 Integrated System [Payload Unit]

Based on the requirements for an integrated system, all components, systems, and structure must seamlessly be integrated into one product without compromising any requirements and designed functionality. In order to test the performance of the payload unit, first the structure must be able to house and protect all components and electronics securely. Second, the payload unit carrying all the electronics and sensors must be able to be secured and fitted onto the quad-copter. Third, all electronics should be all properly connected to the arduino and able to receive power. Fourth, all the sensors must be able to collect their respective data during flight and have it saved onto the SD card. Fifth, connection to the ground station must be maintained and data parameters should be received. Finally, data collected must be analyzed to produce the flight test report. Thus, given that the payload was flight tested safely and data was collected which produced the crop health report, the payload unit was able to complete its mission objective of collecting data and reporting results of crop health to the user. The flight test report is presented in the Figure 22. From the results, the user or farmer will be able to take appropriate action to ensure the longevity of the crop.

OVERALL CROP STATUS: Bad

Average Temperature: **21.82**
Condition: Good

Average Humidity: **18.3**
Condition: Bad - Humidity was too low

Average Air Quality: **71.11**
Condition: Bad - Air Quality level is too low

Average UV Level: **199.93**
Condition: Bad - Average UV level within 100 to 280

Average IR Distance: **13.84**
Condition: Bad - Fly too low

Level	BAD	MARGINAL	GOOD
UV Level	100<UV<280	280<UV<315	315<UV<390
Temperature [deg C]	T>30 or T<18	18<T<20 or 24<T<30	20<=T<=24
Humidity [%]	H<65 or H>90	65<H<80	80<=H<=90
Air Quality [ppm,]	AQ<250	250 AQ 500	AQ>400
Proximity [cm]	P<40 or P>110	40<P<60 or 90<H<110	60<P<90

Figure 22: Flight Test Report

7 Conclusion

Overall, the project has been completed, where all the various milestones and deliverables have been achieved. One of the major milestones to mention was the testing and completion of the integrated system payload. In flight testing the actual completed product, all the requirements were met and fortunately all of the objectives listed were also met. The completed product at this stage can be considered to have reached a maturity level 6. All the work that has been outlined by the gantt chart, work breakdown structure, and work packages have reached 100 percent completion. A single product is expected to cost around $223.85 CAD. In terms of validation and testing, a fault free product was achieved, where the structure was validated to withstand impact in cases such as rotor failure, or detachment of payload through CATIA. It was also validated that all the sensors were operational, collecting data as it should, data was saved to the SD card correctly, and communication between the payload and ground station was maintained. In the completion of the flight test, it was confirmed that the post processing code in VBA was successful in quickly generating a summary report for the user. In conclusion, a successful product has been completed,and the project has been completed without any major unforeseen obstacles.

References

[1] K. Zuk-Golaszewska1. The effect of uv-b radiation on plant growth and development. https://www.agriculturejournals.cz/publicFiles/52840.pdf, 2003.

[2] Agropedia. Climatic and temperature requirement of tomato. http://agropedia.iitk.ac.in/content/climatic-and-temperature-requirement-tomato, 2012.

[3] Amy Rodriguez. Ideal humidity for indoor tomato plants. https://homeguides.sfgate.com/ideal-humidity-indoor-tomato-plants-46330.html, 2010.

[4] Chase Werner. Managing carbon dioxide in your grow space. https://fifthseasongardening.com/regulating-carbon-dioxide, 2014.

[5] NASA. Technology readiness level. https://www.nasa.gov/directorates/heo/scan/engineering/technology/txt_accordion1.html, 2012.

Appendix A: C & DH Code and Payload Code

```
unsigned long start = millis();
// IMPLEMENTATION OF DHT11, MQ-135, GP2Y0A60SZLF, and ML8511
// AER 817 Agricultural Planting Drone Project

// D2 Pin used for Humidity & Temperature Sensor (DHT-11)
// A0 Pin used for Air Quality Sensor (MQ-135)
// A1 Pin used for IR Proximity Sensor (GP2Y0A60SZLF)
// A2 Pin used for UV Sensor [OUTPUT](ML8511)
// A3 Pin used for UV Sensor [3.3V Power](ML8511)

// SD Card
#include <SPI.h>
#include <SD.h>

// DHT-11 Pin Definition
#include "DHT.h"
#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

// ML8511 Pin Definition
int UVOUT = A2;
int REF_3V3 = A3;

// Variable for MQ-135 Sensor
int MQ135SensorValue;
int digitalValue;

// GP2Y0A60SZLF Pin Definition
const int irsensorpin = A1;

//set by default for the SD card library
const int chipSelect = 5;
int id=0;

void setup() {

  Serial.begin(9600);
  Serial1.begin(9600);

  pinMode(13, OUTPUT);
  pinMode( 3, INPUT);
  dht.begin();
  pinMode(UVOUT, INPUT);
  pinMode(REF_3V3, INPUT);

  //SD Card Connection
  Serial.begin(9600);
  Serial.println("Initializing␣SD␣card...");
  pinMode(10,OUTPUT);
  if(!SD.begin(chipSelect)){
    Serial.println("Card␣failed,␣or␣not␣present");
      return;
    }
  Serial.println("card␣initialized.");
  File dataFile=SD.open("LOG.CSV", FILE_WRITE);
```

```
  if(dataFile){
      dataFile.print("ID");
      dataFile.print(",");
      dataFile.print("Temperture");
      dataFile.print(",");
      dataFile.print("Humidity");
      dataFile.print(",");
      dataFile.print("Heat Index");
      dataFile.print(",");
      dataFile.print("Air Quality");
      dataFile.print(",");
      dataFile.print("IR Distance");
      dataFile.print(",");
      dataFile.print("UV Level");
      dataFile.print(",");
      dataFile.print("UV Sensor Output Voltage");
      dataFile.print(",");
      dataFile.println("UV Intensity");
      dataFile.close();
  }
  Serial.begin(9600);
  Serial1.begin(9600);
}

void loop() {
  MQ135SensorValue = analogRead(0); // read analog input pin 0
  digitalValue = digitalRead(2);
  int IRSensorValue = analogRead(irsensorpin);
  double IRdistance = 187754 * pow(IRSensorValue, -1.51);
  delay(1000); // wait 100ms for next reading
  int uvLevel = averageAnalogRead(UVOUT);
  int refLevel = averageAnalogRead(REF_3V3);
  float outputVoltage = 3.3 / refLevel * uvLevel;
  float uvIntensity = mapfloat(outputVoltage, 0.99, 2.8, 0.0, 15.0); //
     Convert the voltage to a UV intensity level
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }

  // Compute heat index in Celsius
  float hic = dht.computeHeatIndex(t, h, false);

  // Printout for DHT-11 Sensor
  Serial.print("Humidity: ");
  Serial.print(h);
  Serial.print(",");
  Serial.print("%\n");
  Serial.print("Temperature: ");
  Serial.print(t);
  Serial.print(",");
  Serial.print(" *C ");
  Serial.print("\n");
  Serial.print("Heat index: ");
```

```
  Serial.print(hic);
  Serial.print(",");
  Serial.print("␣*C␣");
  Serial.print("␣\n");
  Serial.print("------------------------------" );
  Serial.print("␣\n");

  // Printout for MQ-135 Sensor
  Serial.print("Air␣Quality␣Value:␣" );
  Serial.println(MQ135SensorValue, DEC);
  Serial.print("------------------------------" );

  // Printout for IR Proxmity Sensor
  Serial.print("\n");
  Serial.print("Distance␣[cm]:␣");
  Serial.println(IRdistance);
  Serial.print("------------------------------" );
  Serial.print("\n");

  // Printout for UV Sensor
  Serial.print("Average␣Analog␣Read:␣");
  Serial.println(refLevel);

  Serial.print("ML8511␣UV␣Level:␣");
  Serial.println(uvLevel);

  Serial.print("ML8511␣Voltage:␣");
  Serial.println(outputVoltage);

  Serial.print("UV␣Intensity␣(mW/cm^2):␣");
  Serial.println(uvIntensity);
  Serial.print("------------------------------" );
  Serial.println();

  //Transmit data through Xbee
  Serial1.print("Humidity:␣");
  Serial1.print(h);
  Serial1.print("\n");
  Serial1.print("Temperature:␣");
  Serial1.print(t);
  Serial1.print("\n");
  Serial1.print("Heat␣Index:␣");
  Serial1.print(hic);
  Serial1.print("\n");
  Serial1.print("Air␣Quality:␣");
  Serial1.print(MQ135SensorValue, DEC);
  Serial1.print("\n");
  Serial1.print("IR␣Distance:␣");
  Serial1.print(IRdistance);
  Serial1.print("\n");
  Serial1.print("UV␣Level:␣");
  Serial1.print(uvLevel);
  Serial1.print("\n");
  Serial1.print("UV␣Intensity:␣");
  Serial1.println(uvIntensity);
  if (IRdistance>100){
    Serial1.println("Fly␣too␣high");
    Serial1.print("\n");
```

```
  }
  else if (IRdistance<50){
    Serial1.println("Fly_too_low");
    Serial1.print("\n");
  }
  else{
    Serial1.println("Fly_ok");
    Serial1.print("\n");
  }

  //Saving data into on-board SD card
  id=id+1;
  File dataFile=SD.open("LOG.CSV", FILE_WRITE);
  if(dataFile){
      dataFile.print(id);
      dataFile.print(",");
      dataFile.print(t);
      dataFile.print(",");
      dataFile.print(h);
      dataFile.print(",");
      dataFile.print(hic);
      dataFile.print(",");
      dataFile.print(MQ135SensorValue);
      dataFile.print(",");
      dataFile.print(IRdistance);
      dataFile.print(",");
      dataFile.print(uvLevel);
      dataFile.print(",");
      dataFile.print(outputVoltage);
      dataFile.print(",");
      dataFile.println(uvIntensity);
      dataFile.close();
  }


}

// UV Sensor Average Reading Script
//Takes an average of readings on a given pin
//Returns the average
int averageAnalogRead(int pinToRead)
{
  byte numberOfReadings = 8;
  unsigned int runningValue = 0;

  for (int x = 0 ; x < numberOfReadings ; x++)
    runningValue += analogRead(pinToRead);
  runningValue /= numberOfReadings;

  return (runningValue);
}
//The Arduino Map function for floats
float mapfloat(float x, float in_min, float in_max, float out_min, float
    out_max)
{
  return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
}
```

Appendix B: Ground Station Code [VBA]

```
Option Explicit
Sub Main()
'When you run this macro, all existing data will be erased
'Clear any existing data
 Reset

'import all the data from LOG.CSV file
 CSV_Import


'save each data set into individual worksheet
 Data_Set_Logging

End Sub
Sub CSV_Import()
    Dim ws As Worksheet, strFile As String

    Set ws = ActiveWorkbook.Sheets("ALL_DATA") 'save data in LOG.CSV file into "ALL_DATA" worksheet

    strFile = Application.GetOpenFilename("Text Files (*.csv),*.csv", , "Please select text file...")

    With ws.QueryTables.Add(Connection:="TEXT;" & strFile, Destination:=ws.Range("A1"))
         .TextFileParseType = xlDelimited
         .TextFileCommaDelimiter = True
         .Refresh
    End With
End Sub
Sub Data_Set_Logging()
    Worksheets("ALL_DATA").Activate
    Dim Num_Set As Integer
    Num_Set = Data_Set_Count

    Dim Set_Start() As Integer
    ReDim Set_Start(Num_Set)
    Dim LastRow As Long, i As Long, CheckValue As String
    Dim count As Integer
    CheckValue = "ID"
    count = 1
    LastRow = Cells(Rows.count, "A").End(xlUp).Row
    For i = 1 To LastRow
        If Cells(i, 1).Value = CheckValue Then
             Set_Start(count) = i
             count = count + 1
        End If
    Next i
    Dim nws As Worksheet
    For i = 1 To Num_Set
        Set nws = ActiveWorkbook.Sheets.Add(After:=Worksheets(Worksheets.count))
        nws.Activate
        If i < Num_Set Then
            nws.Name = "Data_Set_" & i
            Sheets("ALL_DATA").Range("A" & Set_Start(i) & ":I" & Set_Start(i + 1) - 1).Copy Destination:=
                Sheets("Data_Set_" & i).Range("A1")
        Else
            nws.Name = "Data_Set_" & i
            Sheets("ALL_DATA").Range("A" & Set_Start(i) & ":I" & LastRow).Copy Destination:=Sheets("
                Data_Set_" & i).Range("A1")
        End If
        Call Plotting("Sensor Temperature", "Data_Set_" & i, "Temperature", "B", "1")
        Call Plotting("Sensor Humidity", "Data_Set_" & i, "Humidity", "C", "16")
        Call Plotting("Sensor Heat Index", "Data_Set_" & i, "Heat Index", "D", "32")
        Call Plotting("Sensor Air Quality", "Data_Set_" & i, "Air Quality", "E", "48")
        Call Plotting("Sensor IR Distance", "Data_Set_" & i, "IR Distance (cm)", "F", "64")
        Call Plotting("Sensor UV Level", "Data_Set_" & i, "UV Level", "G", "80")
        Call Plotting("Sensor UV Intensity", "Data_Set_" & i, "UV Intensity", "I", "96")
        nws.Shapes.AddTextbox(msoTextOrientationHorizontal, 50, 200, 380, 300).TextFrame.Characters.Text =
             ResultAnalysis
    Next i

End Sub
Sub Plotting(PlotTitle As String, SheetName As String, PlotValueTitle As String, ColumnId As String,
     LocationRow As String)
    Dim ws As Worksheet
    Dim rgChartData As Range
    Dim myChart As Chart

    Set ws = ThisWorkbook.Worksheets(SheetName)
    Set rgChartData = ws.Range("A:A," & ColumnId & ":" & ColumnId)
    Set myChart = Charts.Add
    Set myChart = myChart.Location(xlLocationAsObject, ws.Name)
    With myChart
        .ChartWizard _
            Source:=rgChartData, _
```

```
            Gallery:=xlArea, _
            Format:=1, _
            PlotBy:=xlColumns, _
            CategoryLabels:=1, _
            SeriesLabels:=1, _
            HasLegend:=True, _
            Title:=PlotTitle, _
            CategoryTitle:="Test ID", _
            ValueTitle:=PlotValueTitle
    End With
    With myChart
       .Parent.Top = Range("K" & LocationRow).Top
       .Parent.Left = Range("K" & LocationRow).Left
    End With
    Set myChart = Nothing
    Set rgChartData = Nothing
    Set ws = Nothing

End Sub
Public Function ResultAnalysis() As String
    Dim ws As Worksheet
    Set ws = Application.ActiveSheet
    Dim aveTemperature As Double
    Dim aveHumidity As Double
    Dim aveAirQuality As Double
    Dim aveUVLevel As Double
    Dim aveIR As Double
    Dim aveOverallbad As Double
    Dim aveOverallgood As Double
    Dim aveOverallmarginal As Double
    Dim LastRow As Integer
    Dim comTemperature As String
    Dim comHumidity As String
    Dim comAirQuality As String
    Dim comUVLevel As String
    Dim comOverall As String
    Dim comIR As String

    aveOverallbad = 0
    aveOverallgood = 0
    aveOverallmarginal = 0

    LastRow = Cells(Rows.count, "A").End(xlUp).Row
    aveTemperature = Round(AVERAGE(Range("B2:B" & LastRow)), 2)
    aveHumidity = Round(AVERAGE(Range("C2:C" & LastRow)), 2)
    aveAirQuality = Round(AVERAGE(Range("E2:E" & LastRow)), 2)
    aveUVLevel = Round(AVERAGE(Range("G2:G" & LastRow)), 2)
    aveIR = Round(AVERAGE(Range("F2:F" & LastRow)), 2)

    If aveTemperature < 18 Or aveTemperature > 30 Then
        If aveTemperature < 18 Then
            comTemperature = "Bad - Temperature was too low"
        ElseIf aveTemperature > 30 Then
            comTemperature = "Bad - Temperature was too high"
        End If
        aveOverallbad = aveOverallbad + 1
    ElseIf aveTemperature >= 20 And aveTemperature <= 24 Then
        comTemperature = "Good"
        aveOverallgood = aveOverallgood + 1
    Else
        comTemperature = "Marginal - The optimum temperature range is between 20 to 24 degrees"
        aveOverallmarginal = aveOverallmarginal + 1
    End If

    If aveHumidity < 65 Or aveHumidity > 90 Then
        If aveHumidity < 65 Then
            comHumidity = "Bad - Humidity was too low"
        ElseIf aveHumidity > 90 Then
            comHumidity = "Bad - Humidity was too high"
        End If
        aveOverallbad = aveOverallbad + 1
    ElseIf aveHumidity >= 80 And aveHumidity <= 90 Then
        comHumidity = "Good"
        aveOverallgood = aveOverallgood + 1
    Else
        comHumidity = "Marginal - The optimun humidity range is between 80 to 90"
        aveOverallmarginal = aveOverallmarginal + 1
    End If


    If aveAirQuality < 250 Then
        comAirQuality = "Bad - Air Quality level is too low"
        aveOverallbad = aveOverallbad + 1
    ElseIf aveAirQuality >= 400 Then
        comAirQuality = "Good"
        aveOverallgood = aveOverallgood + 1
    Else
```

```
        comAirQuality = "Marginal - The optimum Air Quality level is greater than 400"
        aveOverallmarginal = aveOverallmarginal + 1
    End If


    If aveUVLevel > 100 And aveUVLevel < 280 Then
        comUVLevel = "Bad - Average UV level within 100 to 280"
        aveOverallbad = aveOverallbad + 1
    ElseIf aveUVLevel > 315 And aveUVLevel < 390 Then
        comUVLevel = "Good"
        aveOverallgood = aveOverallgood + 1
    ElseIf aveUVLevel >= 280 And aveUVLevel <= 315 Then
        comUVLevel = "Marginal - The optimun UV Level range is between 315 to 390"
        aveOverallmarginal = aveOverallmarginal + 1
    End If

    If aveIR > 60 And aveIR < 90 Then
        comIR = "Good"
        aveOverallgood = aveOverallgood + 1
    ElseIf aveIR < 40 Or aveIR > 110 Then
        If aveIR < 40 Then
            comIR = "Bad - Fly too low"
        ElseIf aveIR > 110 Then
            comIR = "Bad - Fly too high"
        End If
        aveOverallbad = aveOverallbad + 1
    Else
        comIR = "Marginal - The optimun fly range is between 60 to 90"
        aveOverallmarginal = aveOverallmarginal + 1
    End If

    If aveOverallgood >= 3 Or aveOverallbad >= 3 Or aveOverallmarginal >= 3 Then

        If aveOverallmarginal >= 3 Then
            comOverall = "Marginal"
        ElseIf aveOverallgood >= 3 Then
            comOverall = "Good"
        ElseIf aveOverallbad >= 3 Then
            comOverall = "Bad"
        End If

    ElseIf aveOverallgood = aveOverallbad Then
        comOverall = "Marginal"
    ElseIf aveOverallgood = aveOverallmarginal Then
        comOverall = "Marginal"
    ElseIf aveOverallbad = aveOverallmarginal Then
        comOverall = "Bad"
    End If

    ResultAnalysis = "OVERALL CROP STATUS: " & comOverall & vbCrLf _
                     & " " & vbCrLf _
                     & "Average Temperature:        " & aveTemperature & vbCrLf _
                     & "Condition:                  " & comTemperature & vbCrLf _
                     & " " & vbCrLf _
                     & "Average Humidity:           " & aveHumidity & vbCrLf _
                     & "Condition:                  " & comHumidity & vbCrLf _
                     & " " & vbCrLf _
                     & "Average Air Quality:        " & aveAirQuality & vbCrLf _
                     & "Condition:                  " & comAirQuality & vbCrLf _
                     & " " & vbCrLf _
                     & "Average UV Level:           " & aveUVLevel & vbCrLf _
                     & "Condition:                  " & comUVLevel & vbCrLf _
                     & " " & vbCrLf _
                     & "Average IR Distance:        " & aveIR & vbCrLf _
                     & "Condition:                  " & comIR & vbCrLf _

End Function
Public Function Data_Set_Count() As Integer
    Worksheets("ALL_DATA").Activate
    Dim LastRow As Long, i As Long, CheckValue As String
    Dim count As Integer

    count = 0
    CheckValue = "ID"
    LastRow = Cells(Rows.count, "A").End(xlUp).Row
    For i = 1 To LastRow
        If Cells(i, 1).Value = CheckValue Then
            count = count + 1
        End If
    Next i
    Data_Set_Count = count

End Function
Public Function AVERAGE(rng As Range) As Double
    Dim cell As Range, total As Integer, count As Integer
    total = 0
    count = 0
```

```
    For Each cell In rng
            total = total + cell.Value
            count = count + 1
    Next cell
    AVERAGE = total / count
End Function
Sub Reset()
    Dim s As Worksheet, t As String
    Dim i As Long, K As Long
    K = Sheets.count

    For i = K To 1 Step -1
        t = Sheets(i).Name
        If t <> "ALL_DATA" Then

            Application.DisplayAlerts = False
                Sheets(i).Delete
            Application.DisplayAlerts = True

        End If
    Next i

    Sheets("ALL_DATA").UsedRange.ClearContents
End Sub
```

Appendix C: CAD Drawings

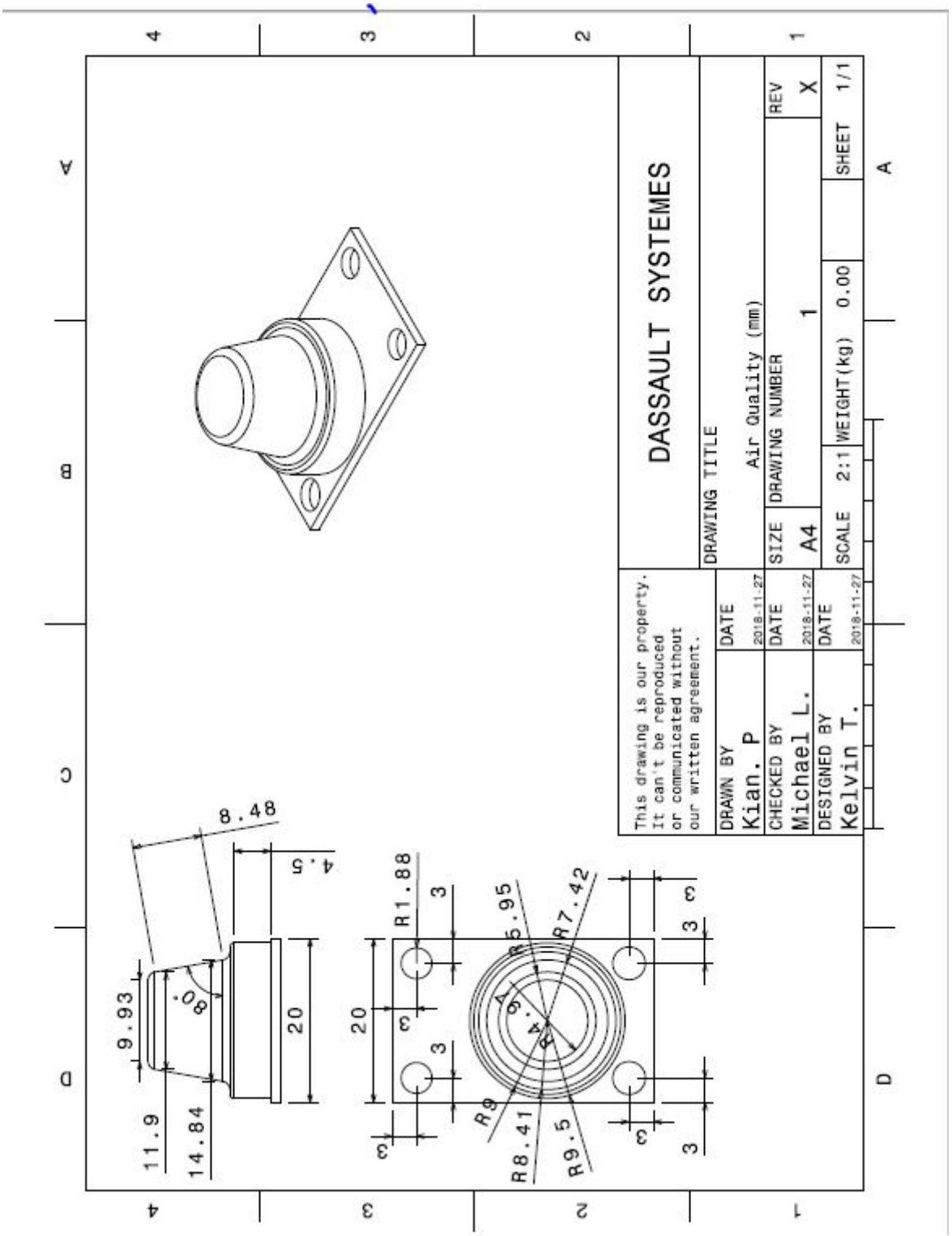


Figure 23: Air Quality Sensor

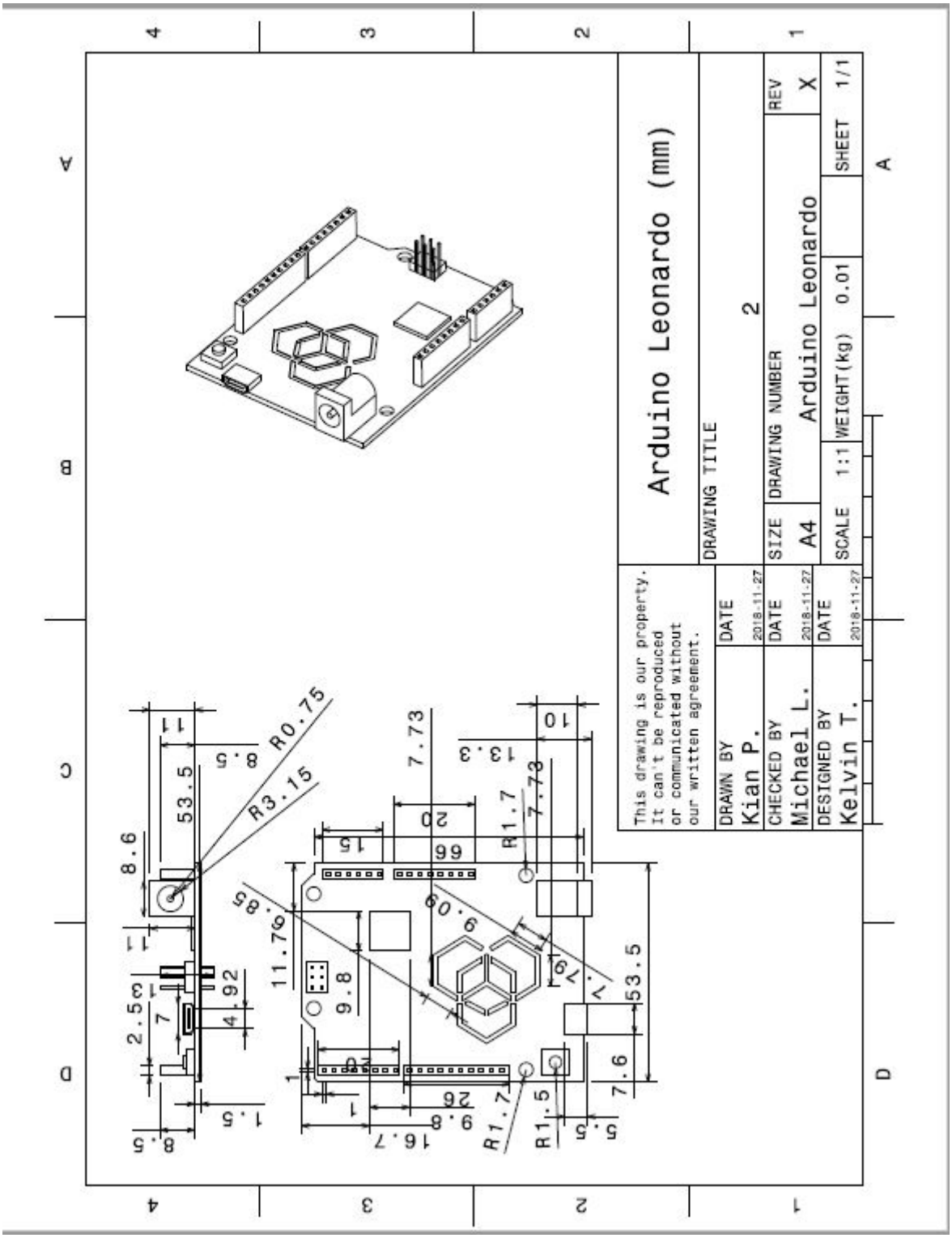


Figure 24: Arduino Leonardo

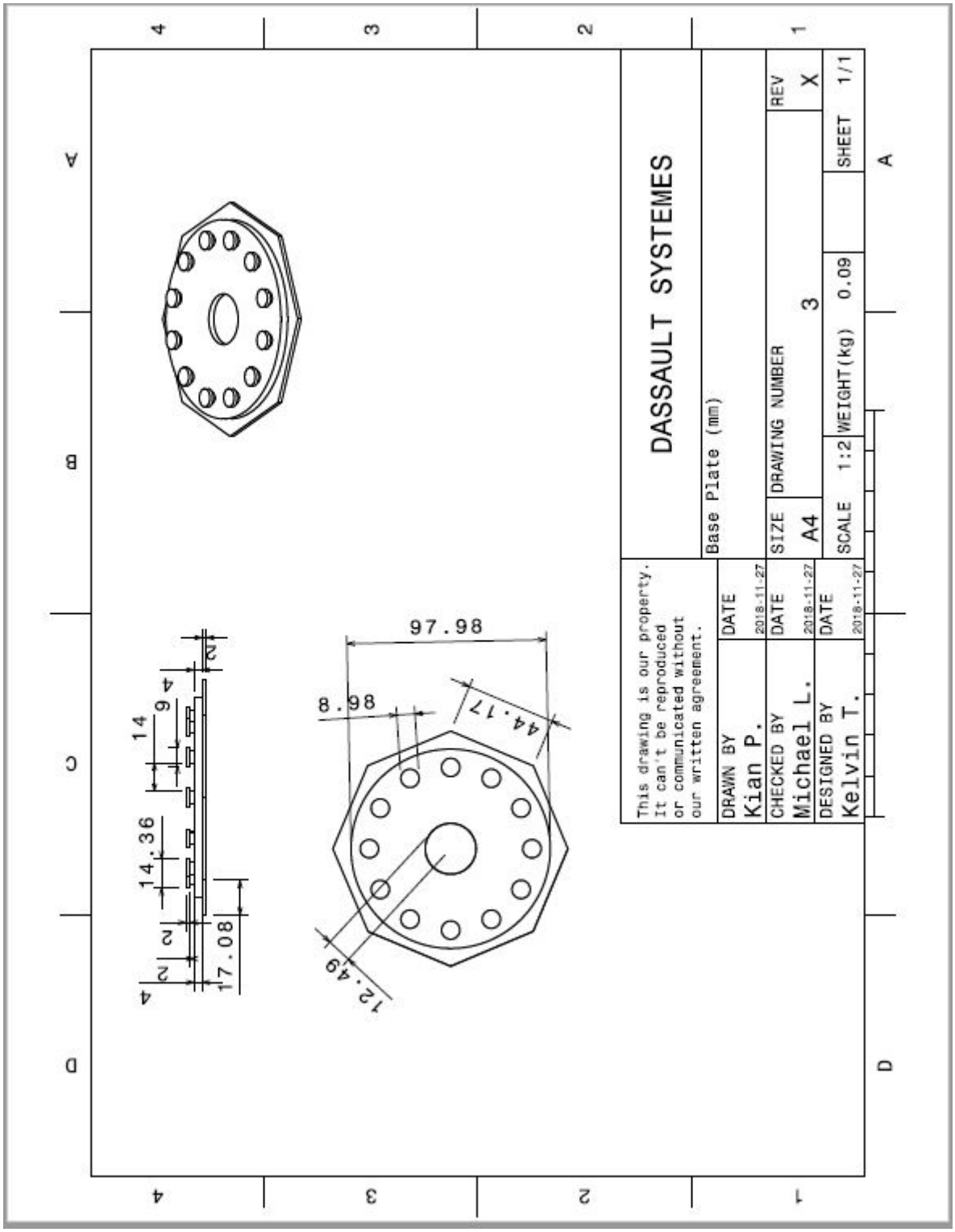


Figure 25: Base Plate

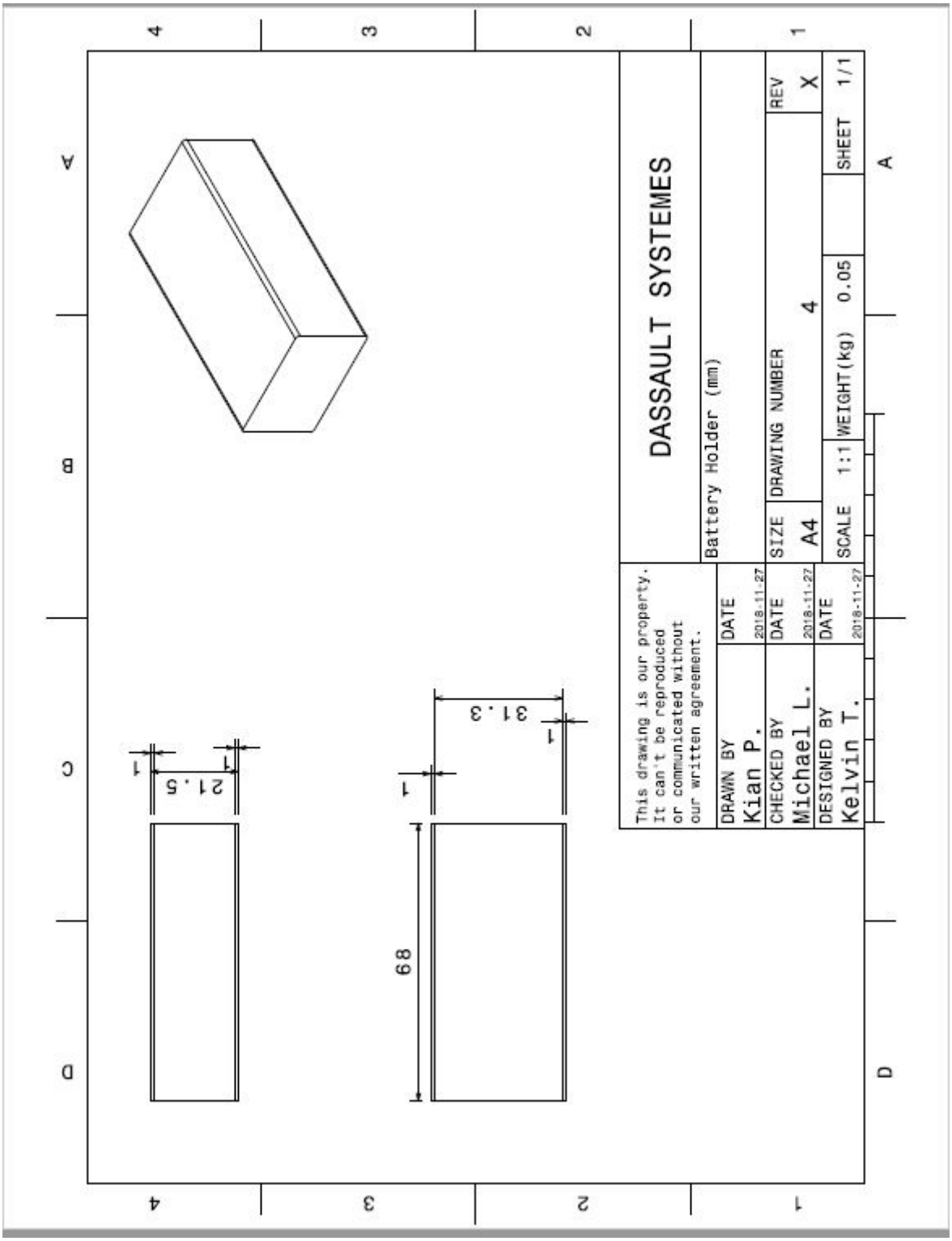


Figure 26: Battery Holder

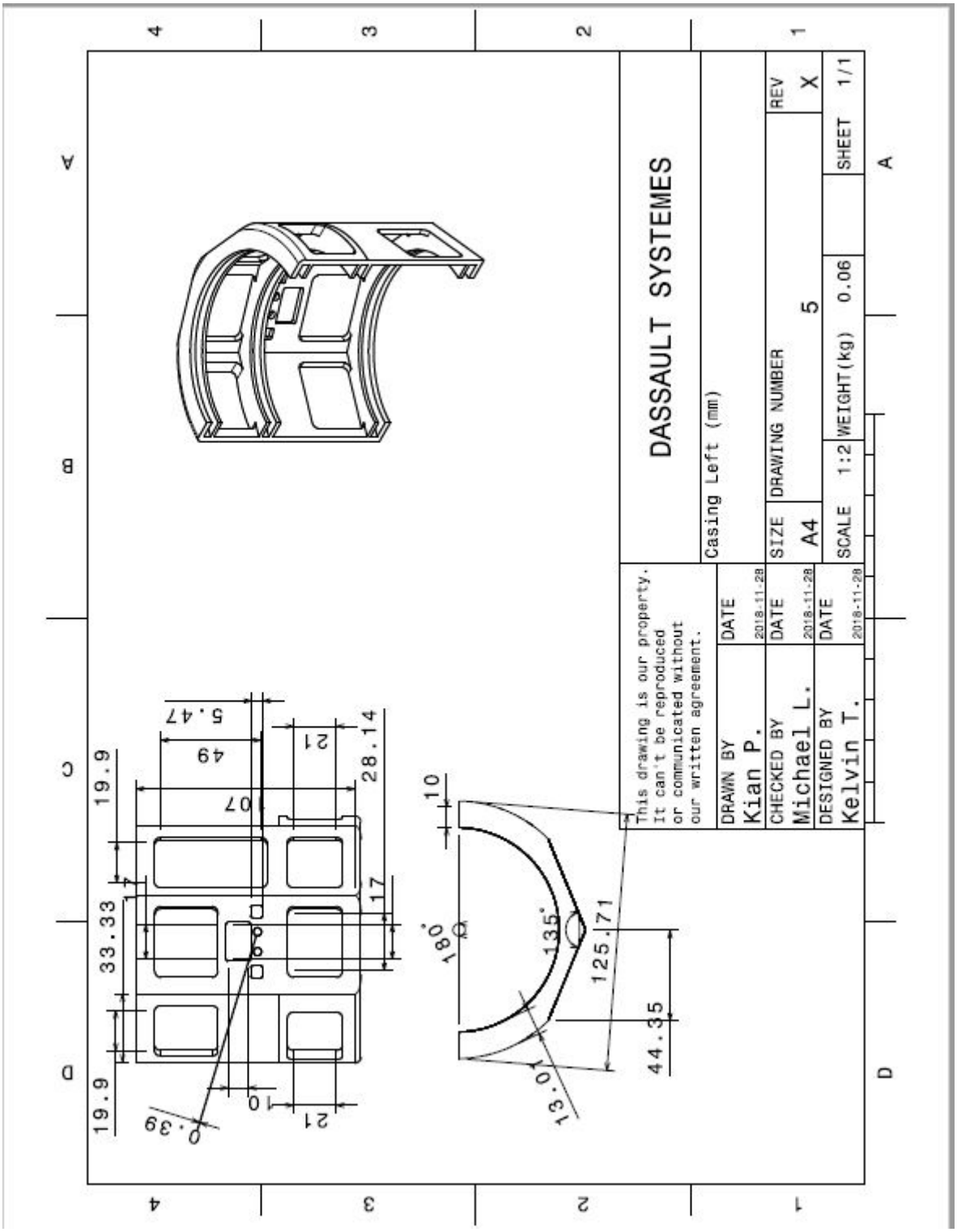


Figure 27: Casing Left

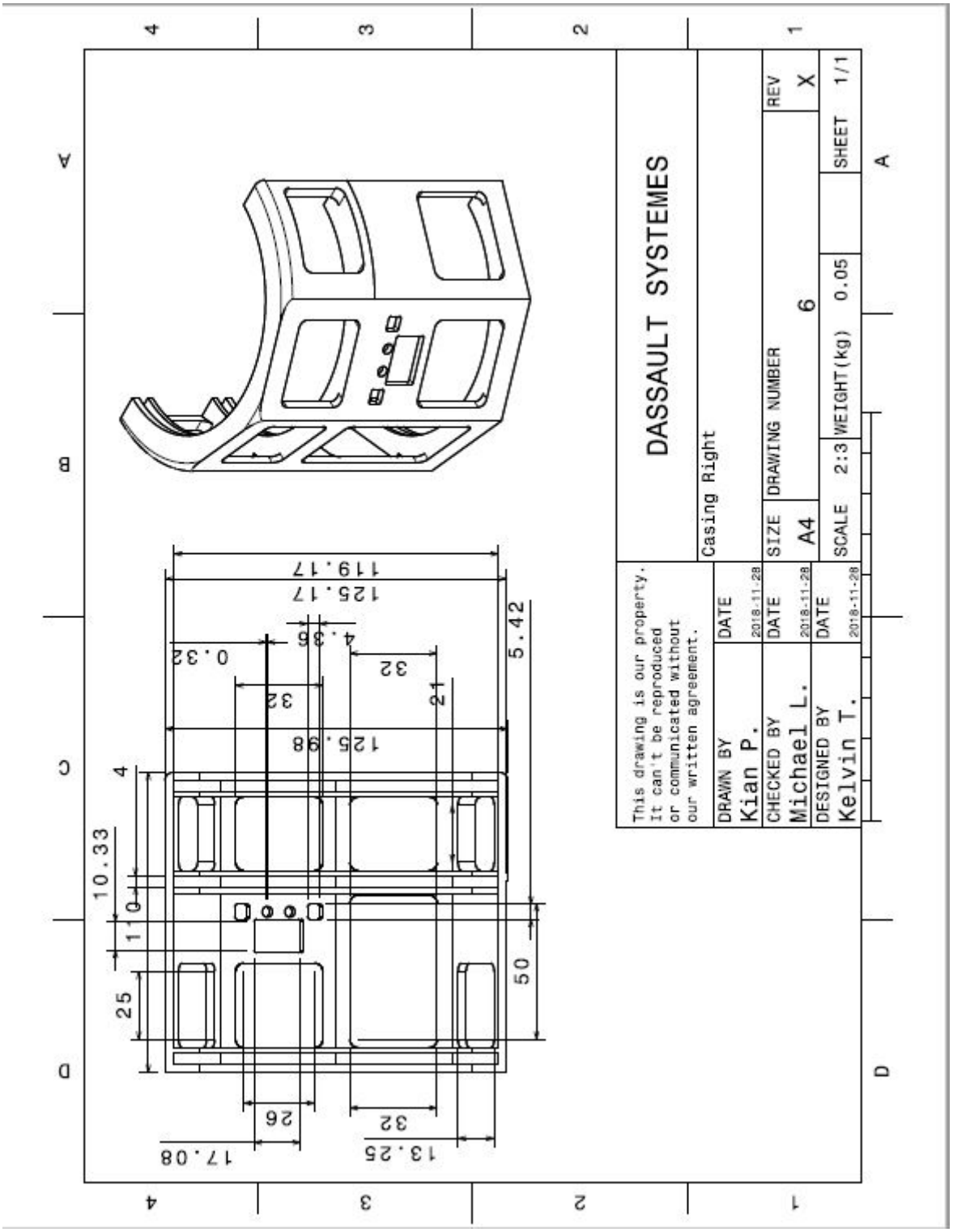


Figure 28: Casing Right

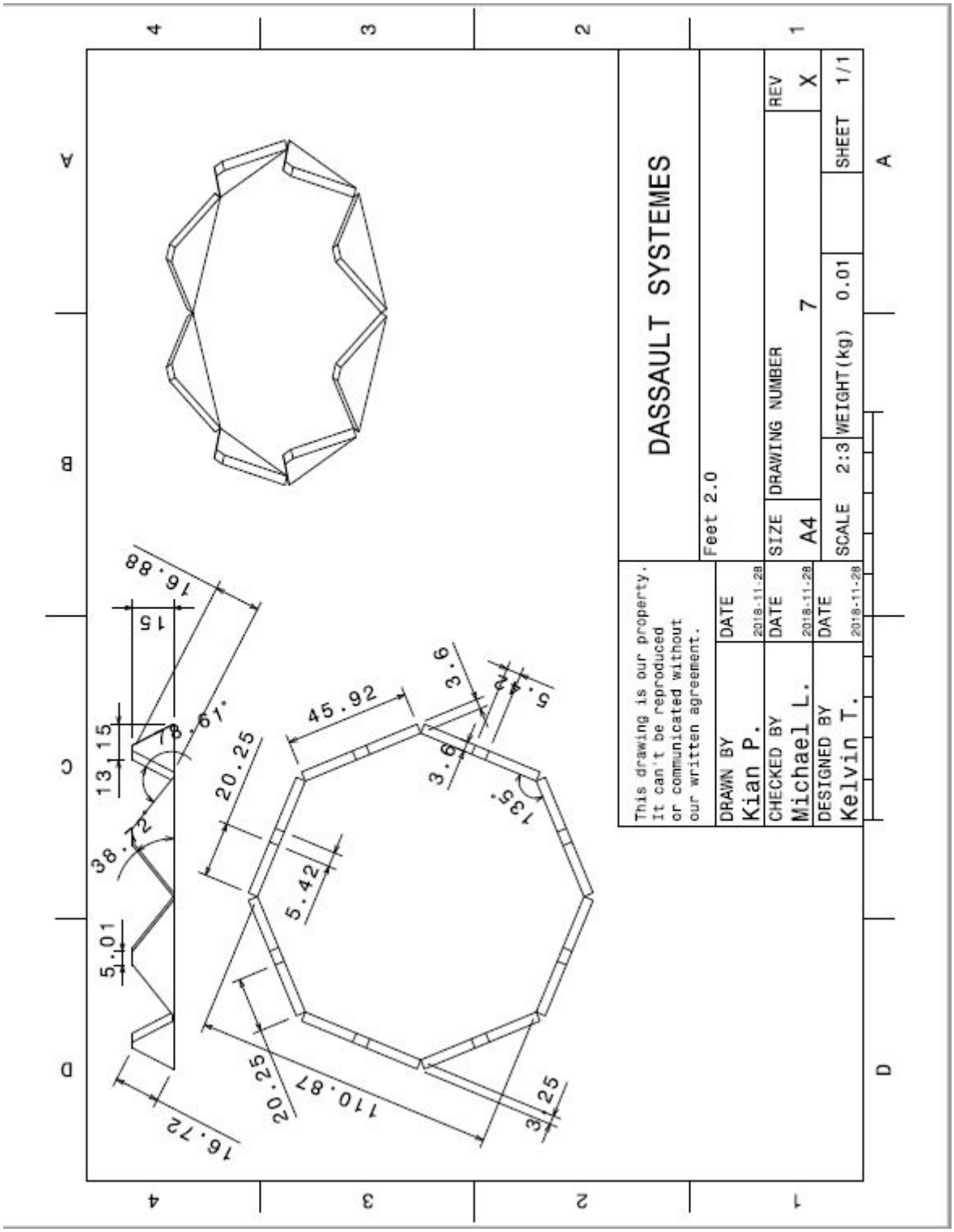


Figure 29: Feet 2.0

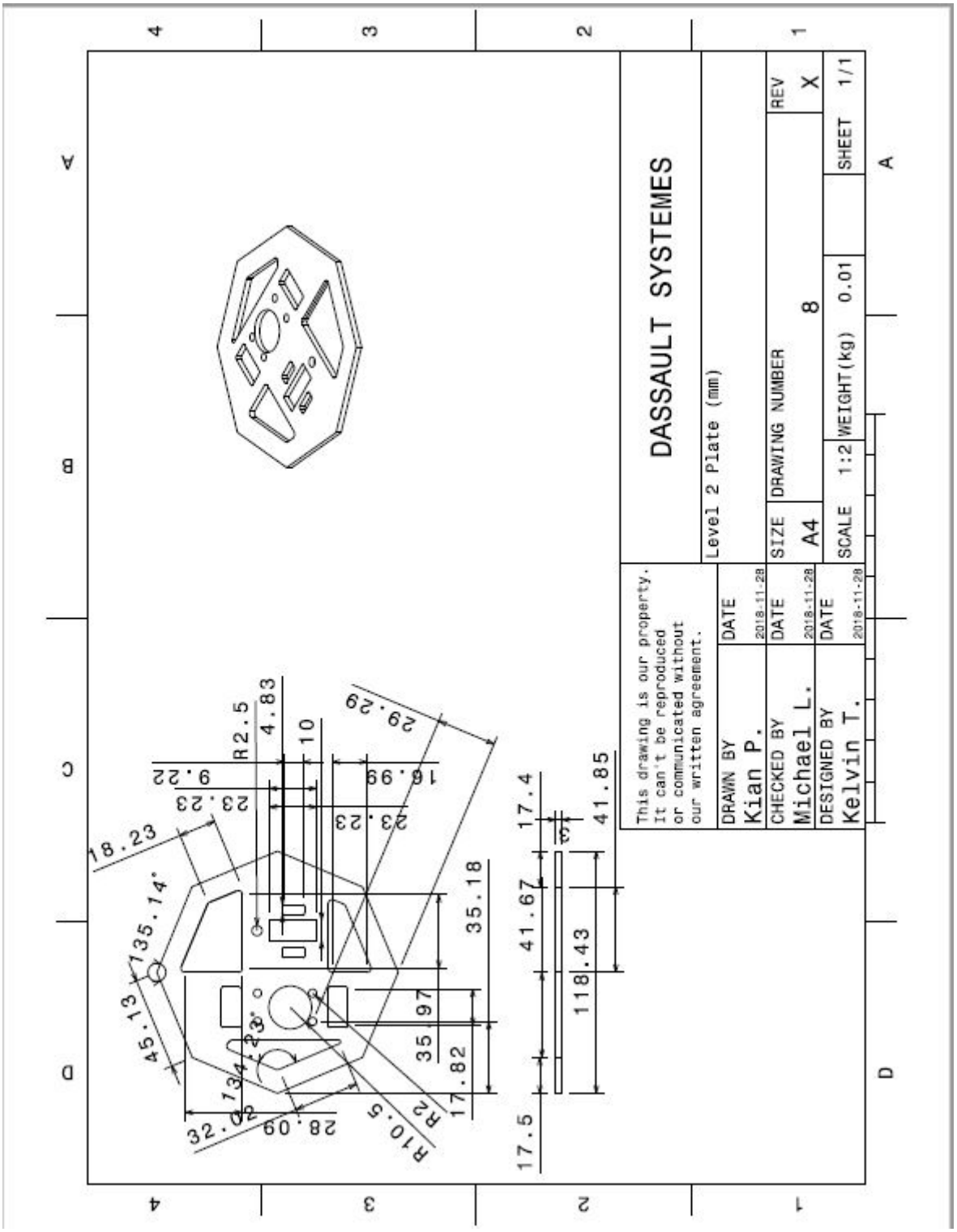


Figure 30: Level 2 Plate

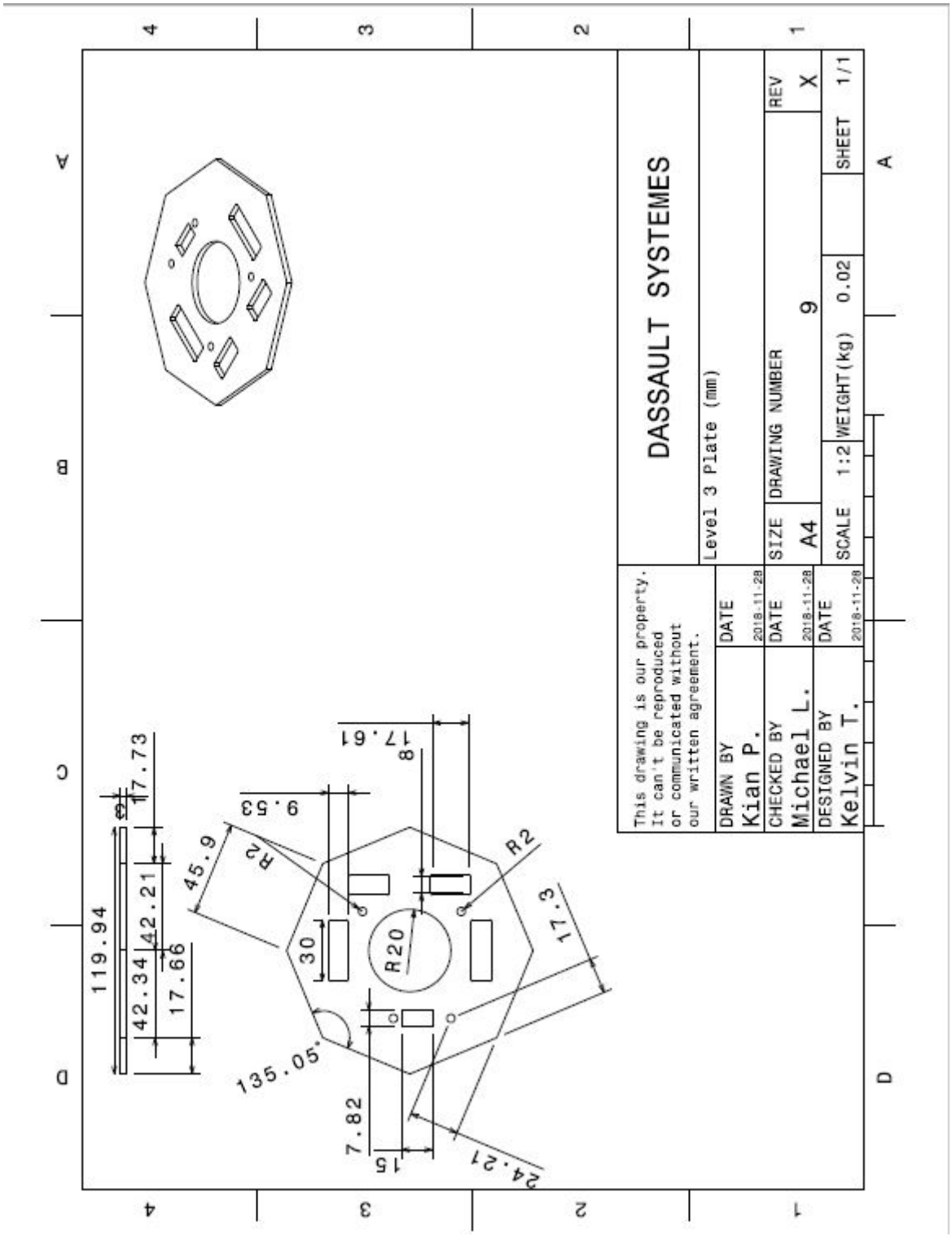


Figure 31: Level 3 Plate

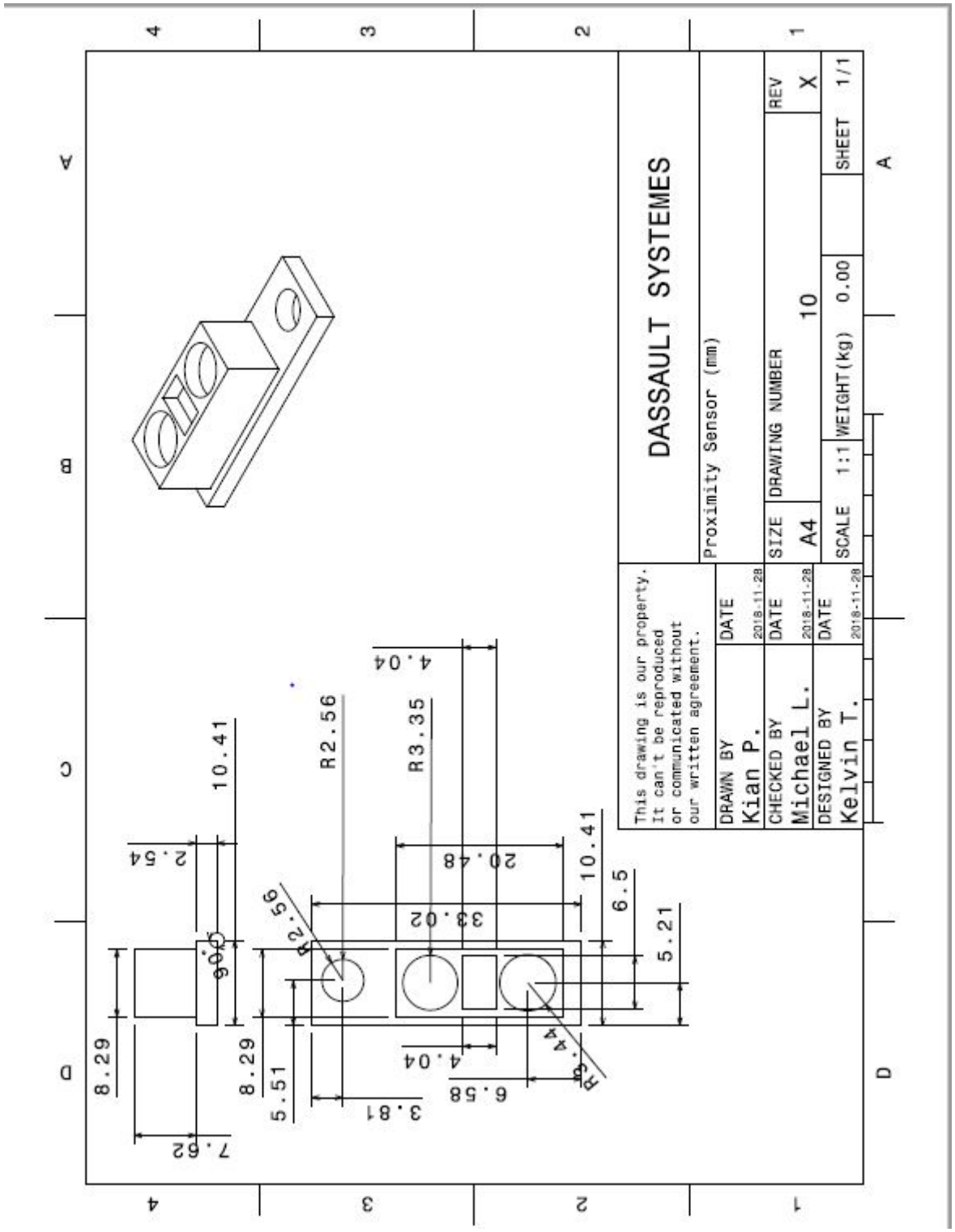


Figure 32: Proximity Sensor

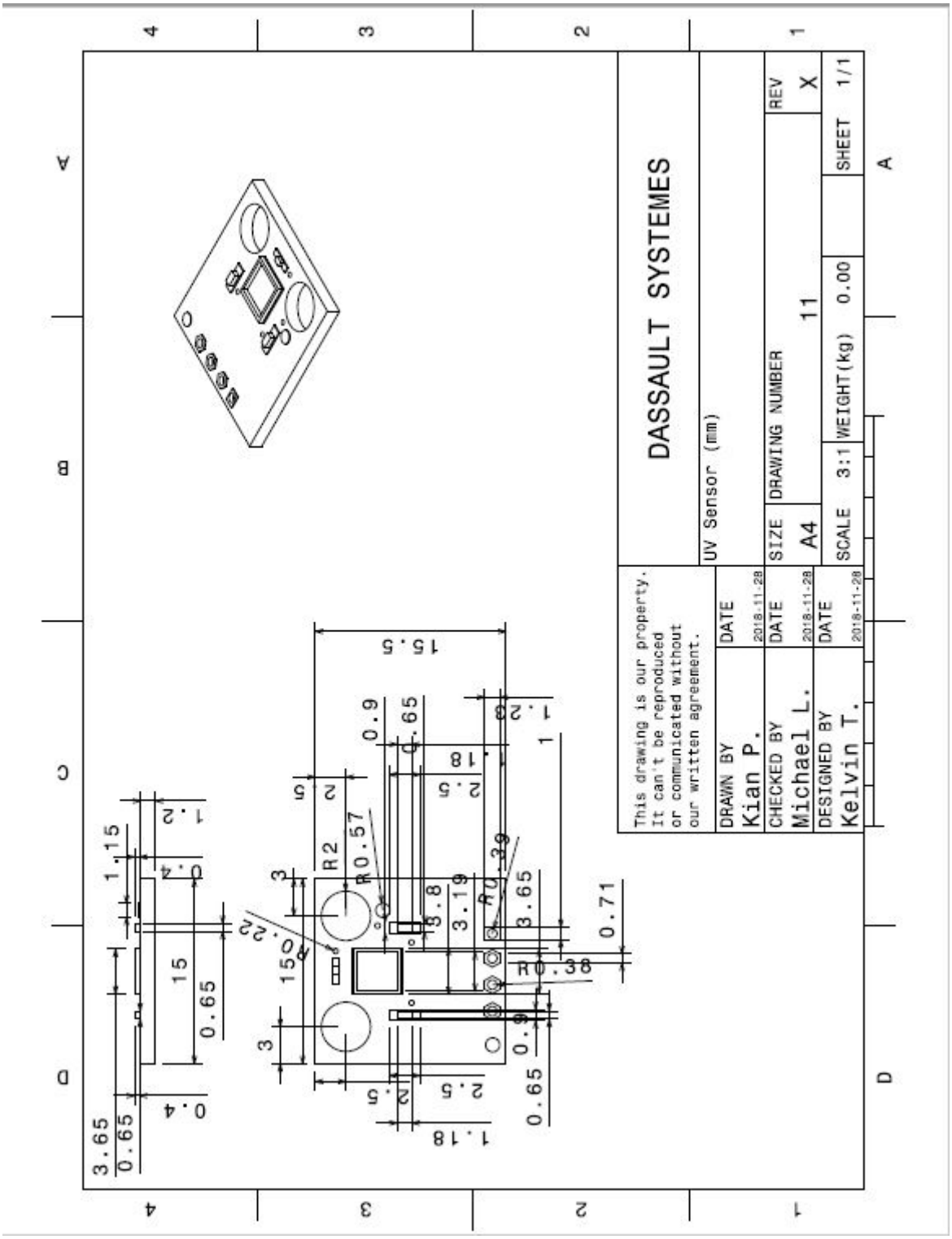


Figure 33: UV Sensor

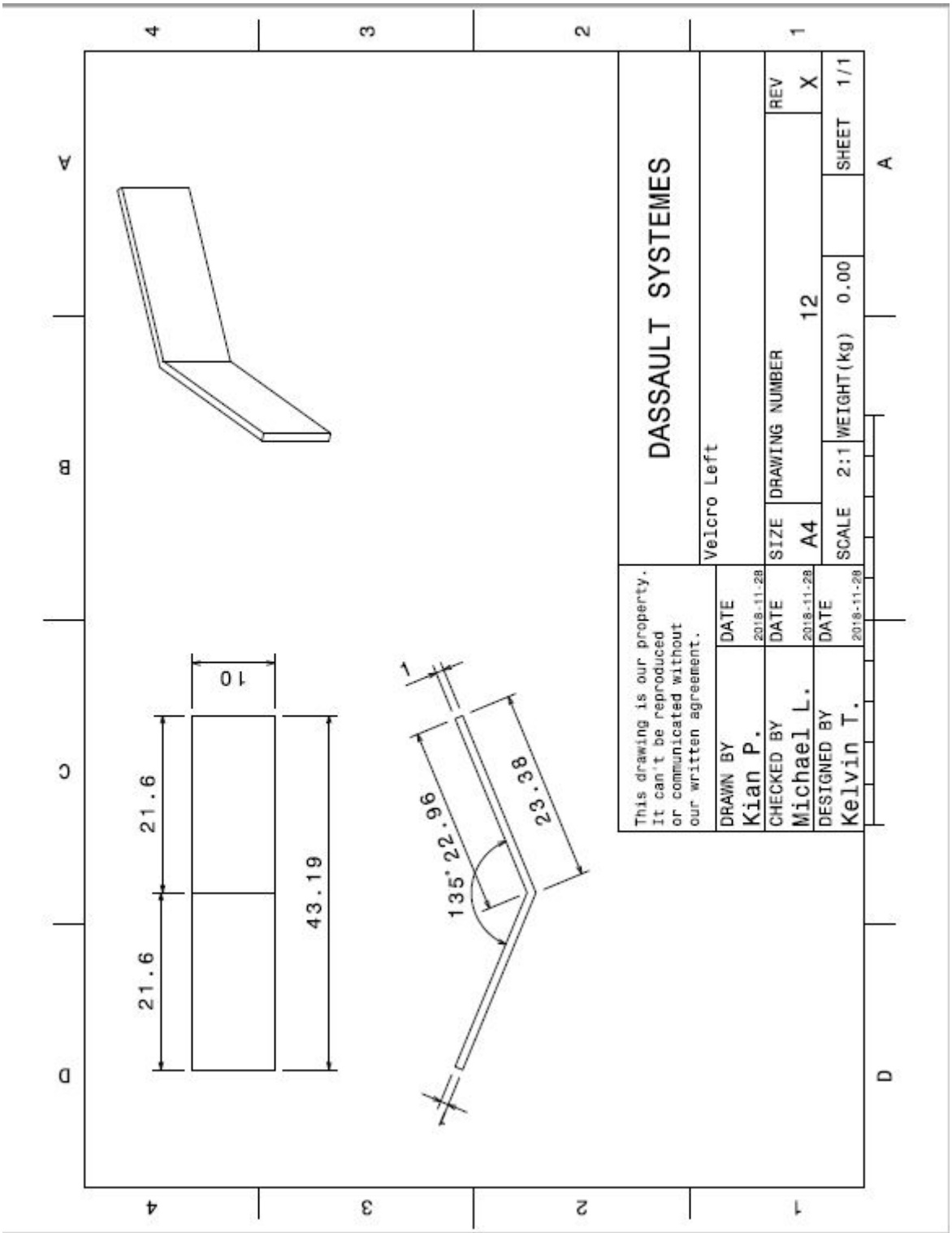


Figure 34: Velcro Left

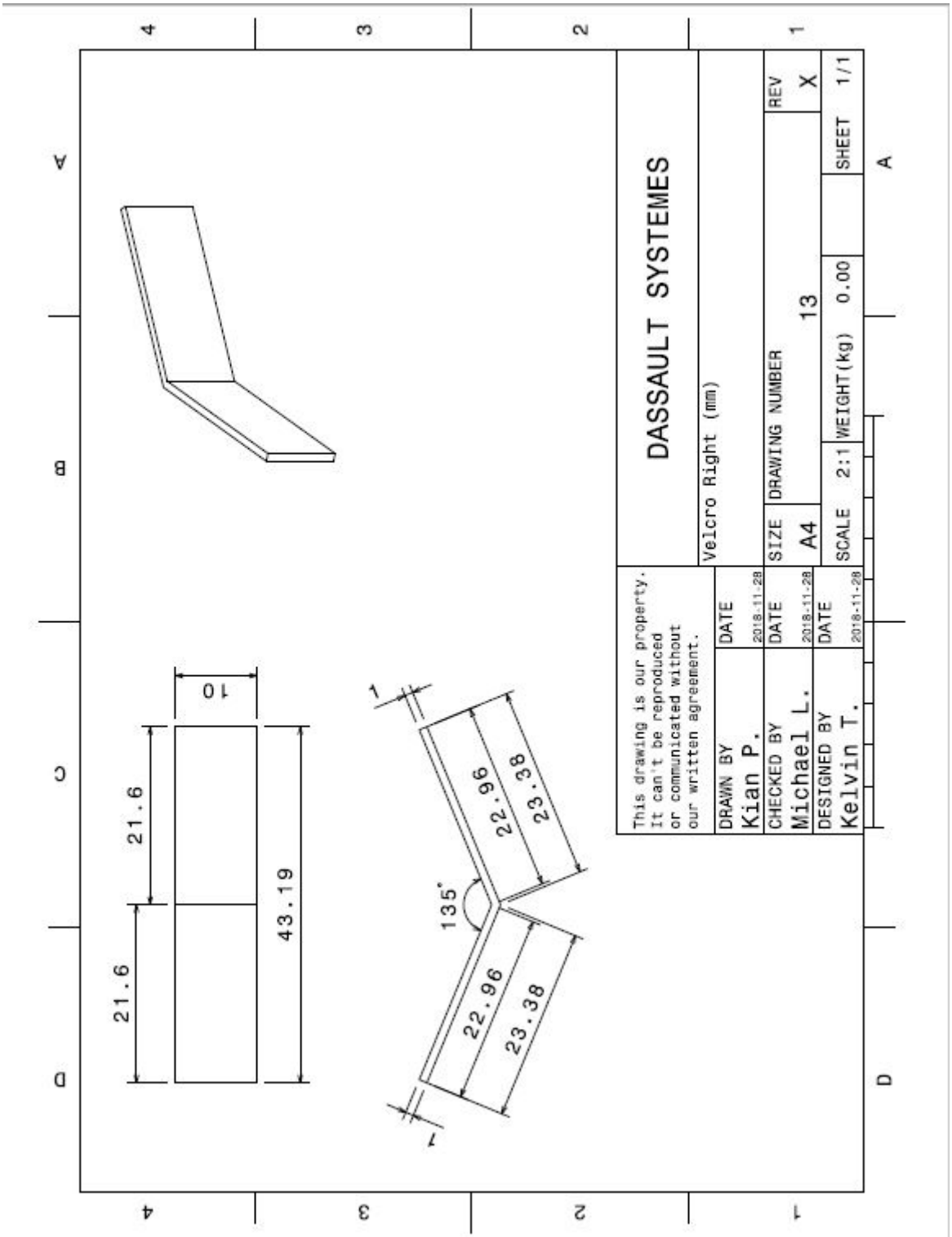


Figure 35: Velcro Right

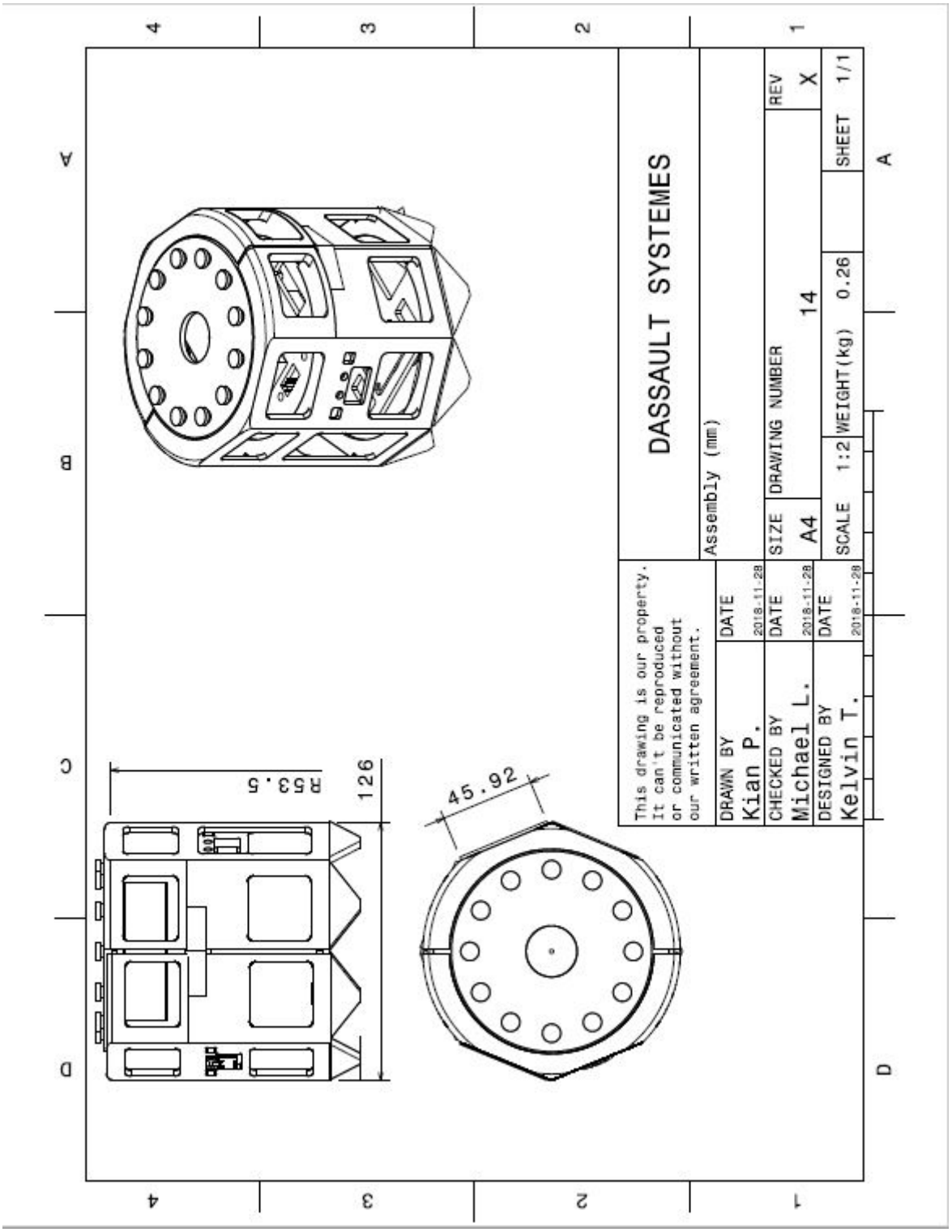


Figure 36: Assembly Drawing