2022

Saturn Engineering Solutions

Gas Turbine Engine and Software

INSTRUCTION MANUAL

Mary Lichtenwalner
Barrett McDonald
Ryan McMillan

Table of Contents

How to Use This Guide

This instruction manual has been created so that whether you are a casual user or an advanced programmer, you will be able to effectively use all features installed on the Mercer University gas turbine engine. This includes running the engine, measuring data with the provided Arduino, and recording data with the program Engine Control Tool.

This guide is divided into two main sections. The hardware section provides information on running the engine, that is, starting the engine, controlling the propane flow, etc. The second section is devoted to helping you understand the software being used to record data while running the engine. There are two programs that will be discussed: the Arduino code and Engine Control Tool.

If you have any difficulties while running the engine, you may reach the Saturn Engineering Solutions team at any of the following email addresses.

- Mary Lichtenwalner maryelichtenwalner@gmail.com
- Barrett McDonald barrettmc77@gmail.com
- Ryan McMillan mcmillanryan55@gmail.com

Navigating the Hardware Section

If you are looking for instructions on how to safely operate the gas turbine engine, refer to the operation section of this manual, located under the Hardware Section. This content breaks operation into a variety of steps for each subsystem of the engine.

If you are looking for guidance with troubleshooting mechanical issues with the engine, refer to the Troubleshooting section under Hardware. This content focuses on issues with the spark plug and combustion instability.

Navigating the Software Section

If this is your first time using Engine Control Tool on your laptop, refer to the Installation section to learn how to download Engine Control Tool onto your laptop.

If you have already downloaded Engine Control Tool, we recommend that you begin by familiarizing yourself with the components of the Engine Control Tool GUI. These are shown in the Engine Control Tool Diagram section. Once you are familiar with the layout of Engine Control Tool, you may begin collecting data while running the engine by referring to the Engine Control Tool Quick Start section.

If you would like to learn about each component of Engine Control Tool in more detail, refer to the Engine Control Tool Component Descriptions section.

If you are a programmer and would like to make changes to Engine Control Tool or the Arduino source code, refer to the Programmer's Manual section. Finally, if you are experiencing a specific problem with either Engine Control Tool or the Arduino sensor array, refer to the Troubleshooting section.

Hardware

Engine

Operation

The following section outlines the proper steps required to safely start the gas turbine engine.

Setup

There are several key factors to consider before starting the engine. These considerations are outlined in the following sections.

1. Using a voltmeter, measure the potential difference generated from the battery. For successful operation, the battery should be producing between 10 and 12 volts.
    a. If the voltage reading is outside this range, connect the battery to a trickle charger and leave overnight.
2. Ensure that there is oil in the oil tank using the built-in dipstick.
    a. If there is not enough oil in the tank, consult Dr. Sumner.
3. Ensure there is propane in the propane tank.
    a. If you cannot hear propane sloshing in the tank when moving the tank, replace the propane tank with a filled propane tank from a store like Lowe's or Home Depot.
    b. Confirm the valve is completely closed.
4. Collect the following equipment:
    a. Leaf Blower(s)
    b. Extension Cord
    c. Windows Laptop with USB connections
    d. Fire Extinguisher
5. Locate the test site.
    a. Saturn Engineering Solutions recommends the sidewalk underneath the bridge connecting the Mercer Science and Engineering Building to the parking lot.
6. Move the engine, along with the equipment listed in Step 4, to the test site.
7. Remove the propane tank from the engine and position it as far away from the combustion chamber as possible.

Oil System and Heat Exchanger

The following section outlines the steps required to properly start the flow of oil through the engine.

1. Turn the Oil Pump and Heat Exchanger switch to the ON position.
    a. This switch is found on the electrical box.
2. Ensure that the fan on the Heat Exchanger is blowing.
3. After about 30 seconds, check the oil pressure gauge.
    a. The oil pressure should be in the range of 30-45 psi.

Glow Plug

The following section outlines the proper use of the spark generation system.

1. Turn the glow plug switch to the ON position.

a. The operators should hear a winding sound coming from the glow plug followed by a spark.
b. If the time between sparks is greater than two to three seconds, do not run the engine.
    i. The battery will need to be charged to reduce the amount of time between sparks.

Note: Be ready to turn the Glow plug switch to the OFF position shortly after ignition.

Spool-Up

The following section outlines the proper use of leaf blowers to start the engine.

1. For an electric leaf blower, plug in the leaf blower to an outlet or an extension cord.
2. For a gas leaf blower, start the blower.
3. When the operator controlling the fuel is in position, start blowing air into the compressor intake.
    a. The end of the leaf blower should butt against the compressor intake housing.
        i. NOT THE COMPRESSOR BLADES
4. Do not remove the leaf blower until a blue flame is visible through the combustion chamber window.

Propane

The following section outlines the proper use of the propane system.

1. Ensure propone regulator is turned all the way off.
2. Once the air flow is induced and glow plug is on, slowly twist regulator until engine starts around 2 psi.
3. Shortly after introducing the propane, ignition should occur.
4. Once the engine starts running begin closing the emergency shut off valve to throttle down the fuel consumption.
5. Turn off the glow plug switch to conserve the charge on the battery.

Shut Down

The following section outlines to proper method required to stop the engine.

1. Close the propane emergency shut off valve.
2. Close the propane regulator valve.
3. Close the valve on the propane tank.
4. Confirm there is no combustion in the combustion chamber.
5. Leave the Oil Pump and Heat Exchanger ON for 5 minutes to assist in engine cooling.
6. Turn off the oil system.
7. Let the cart cool for 15 to 30 minutes depending on the duration of the test run.

CAUTION: The engine components will be hot after the test is complete and can burn the operator.

Safety

The following section outlines potential safety concerns regarding operation and modification of the gas turbine engine.

Operation

Gas turbine engines are inherently dangerous. High revolutions per minute, high temperature, and combustible fuels require close attention for safe operation.

Propane

1. Storage
    a. Do not store propane in areas where the tank can be exposed to temperatures significantly higher than atmospheric temperature. Store inside as much as possible.
    b. Do not store the propane tank on the cart.
        i. After the cart has been moved to a storage location, remove the tank from the cart to eliminate any issues with electrical components arcing to the tank.
2. Operation
    a. Do not operate the engine with the tank on the cart.
        i. Move the tank as far from the engine as the tubing will allow.
    b. Note that the regulator valve does not completely cut off flow.
        i. Do not rely on this valve alone to completely restrict the flow of propane.
    c. Do not turn on the flow of propane without the glow plug turned on and air flowing through the engine.
        i. This will cause an explosion.

Compressor

1. Do not reach any part of your body into the compressor housing during operation.
    a. The compressor could suck body parts or clothing into the compressor blades which can cause bodily harm.
2. Do not stand in the radial direction of the compressor.
    a. Compressor blades could break off from the rotor and launch through the compressor housing, sending debris in the radial direction.

Turbine

1. The exhaust from the turbine is VERY HOT.
2. Do not stand behind the turbine exhaust.
    a. Debris could shoot out of the exhaust.
    b. Exhaust gasses are extremely hot.
3. Do not stand in the radial direction of the turbine.
    a. Turbine blades could break off from the rotor and launch through the turbine housing, sending debris in the radial direction.

Other Engine Components

1. Do not touch the combustion chamber, turbine housing, or combustion chamber adaptor.

a. These components are VERY HOT.

Modification

Some subsystems on the cart pose additional safety concerns to operators.

Electrical

1. Do not work on the wiring of the cart when the wiring is connected to the battery.
    a. The wiring can shock you.
2. Be careful of wiring connections that have been recently soldered.
    a. These joints might still be hot.

Glow Plug

The glow plug system is known to have significant grounding issues. Do not work on the glow plug when the battery is connected to the wiring for any reason.

1. Again, do not work on the glow plug when the battery is connected.
2. Before working on the glow plug, ground the body of the main electrical box to the cart to dissipate any charge build up in the system.

Oil System

1. Do not work on the oil system while the oil pump is on.
2. Do not work on the oil system while the battery is connected to the wiring.
3. Be cautious of the oil in the oil lines.
    a. Oil will be in the lines connecting components of the oil system.
        i. Be ready for oil to leak from the open connections.

Troubleshooting

This section covers common problems that may arise during engine operation.

Propane

1. If there is an odor of unburnt propane:
    a. Utilize the shut off valve to throttle the amount of propane entering the system.
    b. Ensure that all connections are leak free within the propane lines.
        i. Use Teflon tape on all propane connections to ensure leak free connections and tighten with wrenches, not by hand.
2. No propane fuel entering the system:
    a. Ensure that the propane tank is not empty
        i. If propane tank is empty, replace tank refill tank at Lowes, Home Depot, etc.
        ii. If the propane tank is not empty, ensure there are no blockages within propane lines

Glow Plug

1. No spark being generated by glow plug.
    a. Turn off all electrical components connected to the battery.
    b. Connect batter to a trickle charger for 3 hours to charge the battery.
    c. Reconnect battery and try glow plug again.
2. Still no spark being generated by glow plug.
    a. Disconnect battery from the system.
    b. Remove the elbow connection connecting the glow plug to the combustion chamber.
        i. Ensure that the wires do not get twisted and entangle during this process which could result in harming the wire connections.
    c. Remove electrical tape around the elbow connection and ensure that all wire connections are intact.
        i. The metal sheath encasing the wires grounds the system. Ensure that this metal sheath is grounded to the elbow connection. This connection is currently made by a wire soldered between the metal sheath and the elbow connection.
    d. Once all wire connections are intact, reconnect the battery and glow plug. A spark should ensue every couple seconds.

Software

Engine Control Tool

Arduino Code

Installation

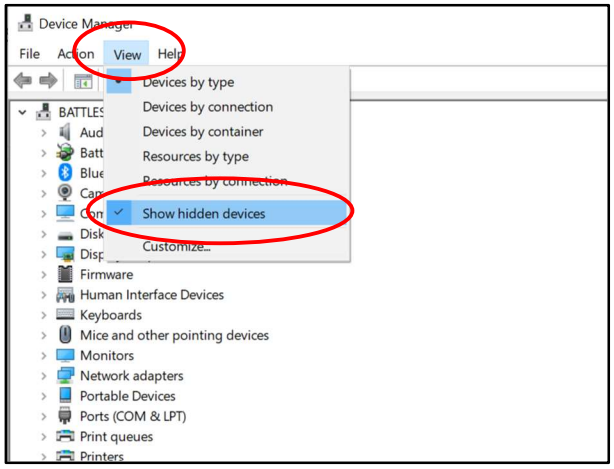
The following sections describe how to access the Arduino program and Engine Control Tool from either the provided USB drive or a Github repository.

USB Option

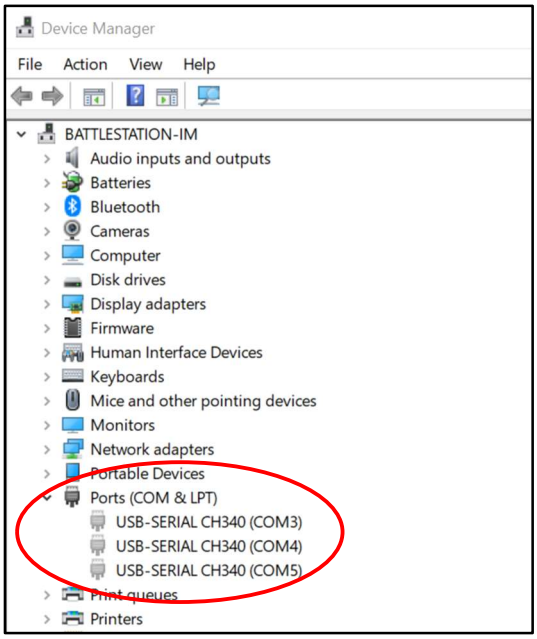
The following steps describe the installation process when using the provided USB drive.

Arduino Source Code

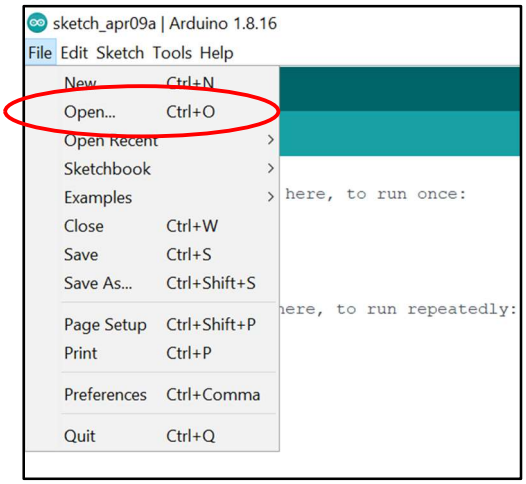
1. First, download the correct version of the Arduino IDE for your operating system from https://www.arduino.cc/en/software.
2. Follow the instructions to install the Arduino IDE.
3. The Arduino must be connected to USB serial port COM3, COM4, or COM5. To confirm that your laptop has one of these ports open, go to device manager by searching “Device Manager” in the start menu. Once Device Manager is open, select View < Show hidden devices.



4. Open the drop down next to “Ports (COM & LPT)” and confirm that either COM3, COM4, or COM5 is available. You may connect to the Arduino with any of these ports.



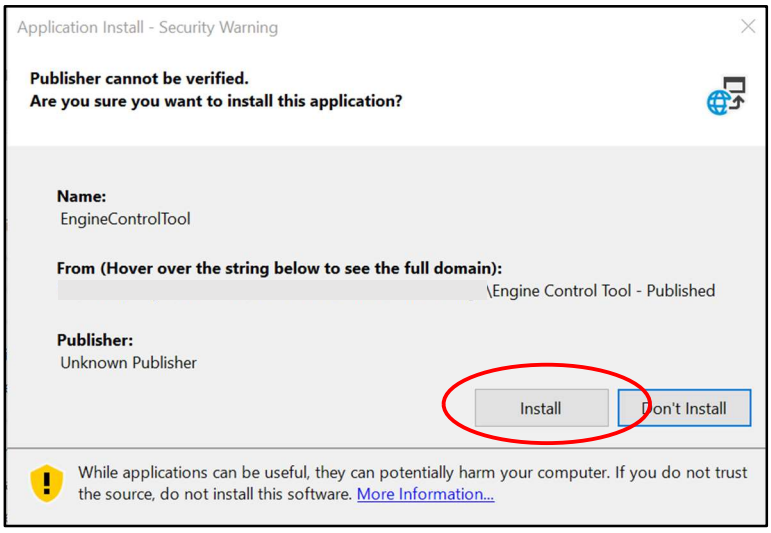
5. If none of these ports are available, you will need to change the source code for both the Arduino and Engine Control Tool to match the port you have available. This problem is addressed in the Troubleshooting section of this manual. If you have one of these ports available, simply connect to the Arduino through one of these ports with the provided USB cable and the Arduino will be ready to send data to Engine Control Tool.
6. To access the Arduino source code, open the USB drive in File Explorer, and select the folder *CombinedProgram*. Copy this folder to the desired location on your device.
7. Once the Arudino IDE is open, select File < Open and navigate to the location where you saved the folder *CombinedProgram*. Open this folder and select the file *CombinedProgram.ino*.



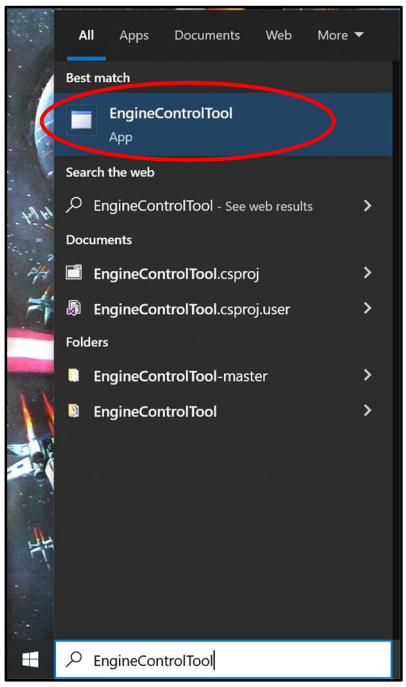
8. The Arduino source code is now opened. Information on the structure of the source code may be found in the section Programmer's ManualProgrammer's Manual.

Engine Control Tool – No Source Code

1. To install Engine Control Tool to your laptop, copy the folder *Engine Control Tool – Published* from the USB drive to the desired location on your laptop.
2. Navigate to the location where you moved this folder and open the file *EngineControlTool.application*. Click "Install" on the notice that pops up.

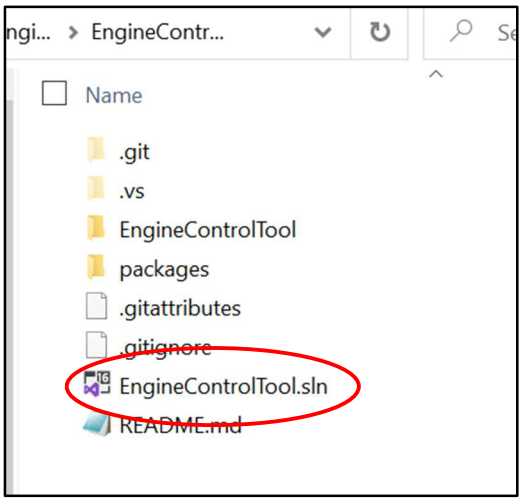


3. After the program installs, you may open the program by searching "EngineControlTool" in the start menu. It may be uninstalled at any time in Settings or Control Panel.

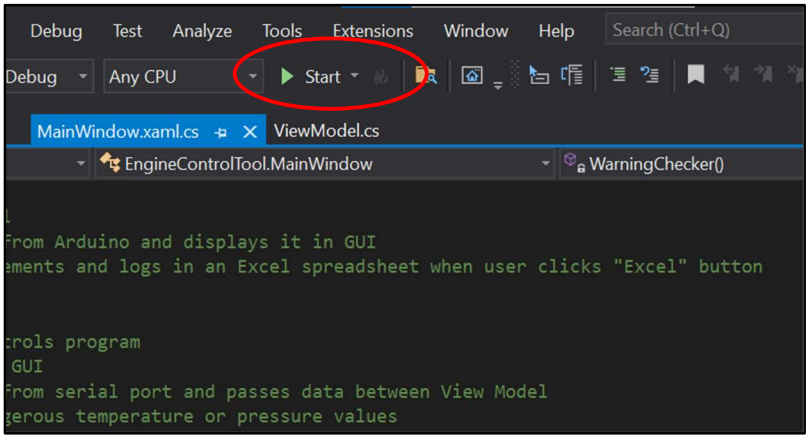


Engine Control Tool – Source Code

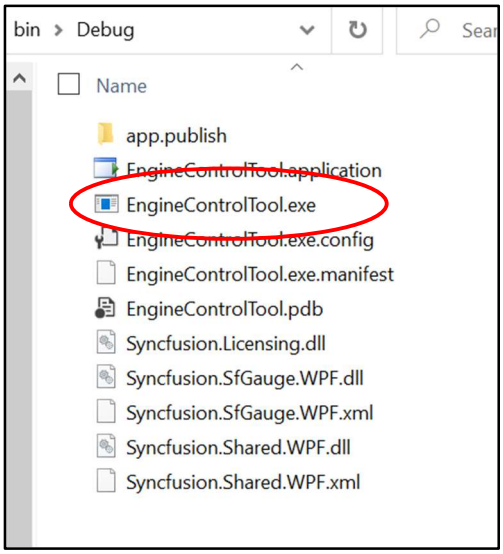
1. If you would like access to the Engine Control Tool source code, you will need to first download Microsoft Visual Studio from https://visualstudio.microsoft.com/. Select the correct version for your operating system. The community version of Visual Studio is free to download and use.
2. Once Visual Studio is downloaded, copy the folder *Engine Control Tool* from the USB drive to the desired location on your laptop.
3. To open in Visual Studio, go into the folder *Engine Control Tool* wherever you saved it, and then select …\EngineControlTool\EngineControlTool.sln. This is the Visual Studio solution file, and it will open the program in Visual Studio.



4. To run the program directly from the source code without installing it to your device, you have two options. With the program open in Visual Studio, you may select the "Start" button at the top center of Visual Studio.



5. Alternatively, you may run the program by selecting the file *EngineControlTool.exe* found at the file path …\EngineControlTool\EngineControlTool\bin\Debug\EngineControlTool.exe where you begin inside the folder *Engine Control Tool* originally copied from the USB drive.

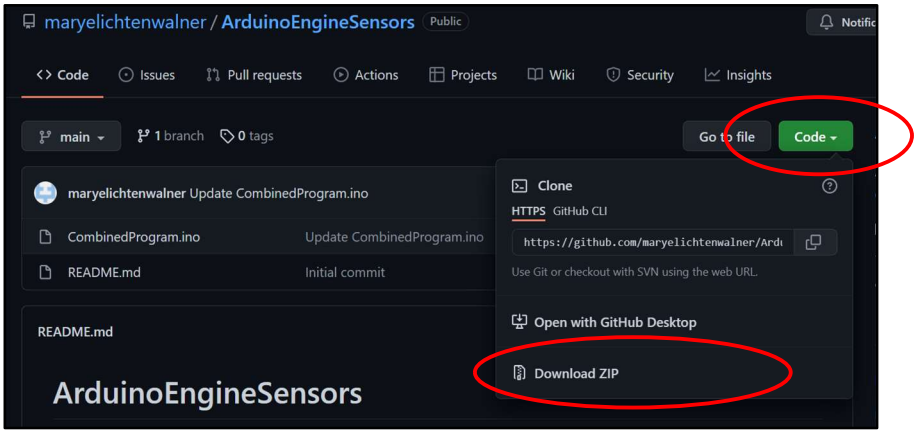


Github Option

If preferred, you may access both the Arduino code and Engine Control Tool through Github. To use this method, you will need both the Arduino IDE (https://www.arduino.cc/en/software) and Visual Studio (https://visualstudio.microsoft.com/) installed on your laptop. Install these programs before proceeding with this method.

Arduino Source Code

1. You will again need to confirm that you have the correct ports open to use the Arduino. Follow the first five steps of the USB installation portion above to confirm that you have connected the Arduino to a compatible USB port.
2. The Arduino source code may be accessed at https://github.com/maryelichtenwalner/ArduinoEngineSensors.
3. Once on this page, select Code < Download ZIP and save the ZIP folder to your desired location.

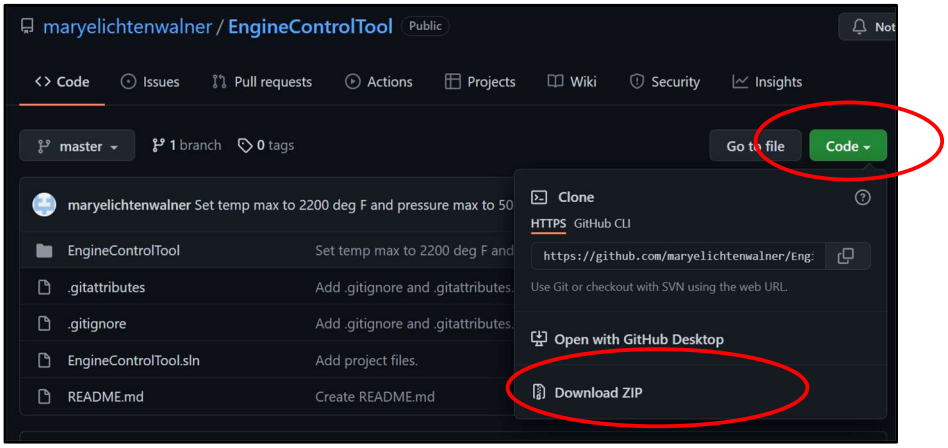


4. Extract the files from the ZIP folder once it has downloaded.
5. Open the Arduino IDE and select File < Open. Navigate to the file *CombinedProgram.ino*.

6. You now have the Arduino source code opened.

Engine Control Tool – Source Code

1. You may access the Engine Control Tool Github repository at https://github.com/maryelichtenwalner/EngineControlTool.
2. Once on this page, select Code < Download ZIP and download a ZIP folder of the repository to your desired location.



3. Extract the files from the ZIP folder once it has downloaded.
4. Open the Visual Studio solution file by navigating to …\EngineControlTool-master\EngineControlTool-master\EngineControlTool.sln.
6. The source code is now open in Visual Studio, and Engine Control Tool may be run by selecting the "Start" button at the top center of Visual Studio.
7. Alternatively, you may run the program by selecting the file *EngineControlTool.exe* found at the file path …\EngineControlTool-master\EngineControlTool-master\EngineControlTool\bin\Debug\EngineControlTool.exe.

Engine Control Tool Diagram

The following image shows the Engine Control Tool GUI. The names for these visual elements given below will be used throughout the rest of this manual.

The following list corresponds to the numbered items on the figure above.

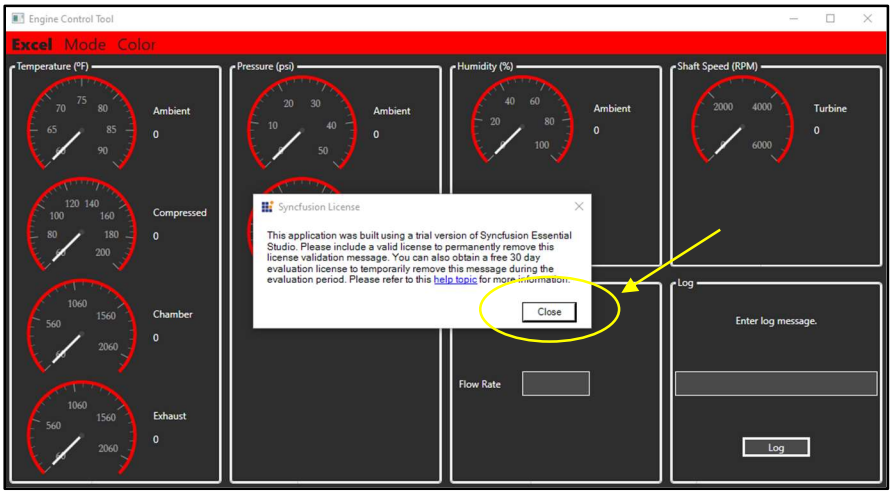
1. Menu Bar
   1.1. Excel Button
   1.2. Mode Button
   1.3. Color Button
2. Temperature Groupbox
   2.1. Ambient Air Temperature Gauge/Label
   2.2. Compressed Air Temperature Gauge/Label
   2.3. Chamber Temperature Gauge/Label
   2.4. Exhaust Temperature Gauge/Label
3. Pressure Groupbox
   3.1. Ambient Air Pressure Gauge/Label
   3.2. Compressed Air Pressure Gauge/Label
4. Humidity Groupbox
   4.1. Ambient Air Humidity Gauge/Label
5. Shaft Speed Groupbox
   5.1. Turbine Shaft Speed Gauge/Label
6. Flow Rate Groupbox
   6.1. Flow Rate Textbox
7. Log Groupbox
   7.1. Log Message Textbox
   7.2. Log Button

Engine Control Tool Component Descriptions

The following sections detail the different components of the Engine Control Tool GUI. All numbers refer to the corresponding labels in the Engine Control Tool Diagram section above.

Startup

Upon startup, Engine Control Tool will pop up a notice about the “Syncfusion License.” This refers to the C# library used to initialize the gauges used to display measurements. You may simply close this message.

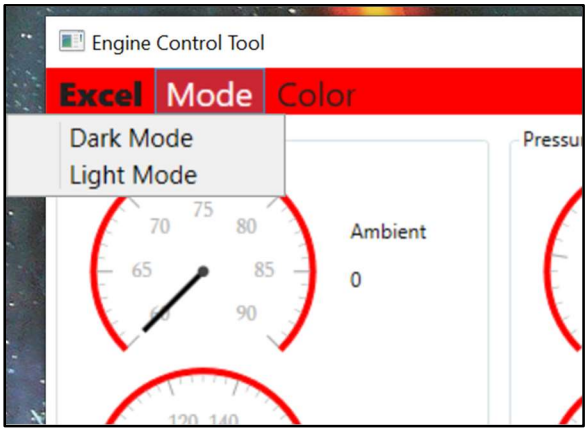


Menu Bar

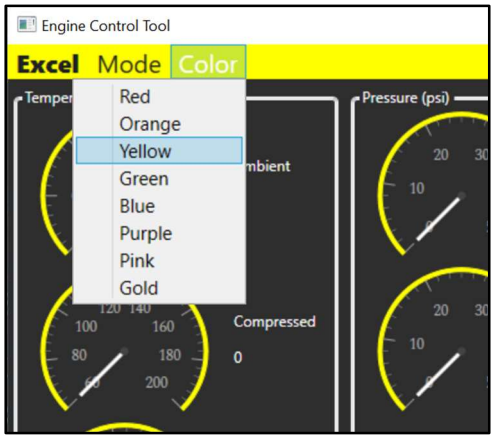
Engine Control Tool has a Menu Bar (1) across the top of the window with three buttons: Excel Button (1.1), Mode Button (1.2), and Color Button (1.3).

The Excel Button exports the recorded data to an Excel spreadsheet. Engine Control Tool continues to operate and record data even after this button is pressed, so it is possible to generate multiple Excel spreadsheets during one engine test run.

The Mode Button opens a drop-down menu where you may choose to display Engine Control Tool in dark mode or light mode.



The Color Button opens a drop-down menu where you may choose from a list of eight accent colors. The selected accent color will change the color of the menu bar and the rims on the gauges.



Both the Mode Button and Color Button make purely aesthetic changes to Engine Control Tool and do not affect the functionality.

Temperature Groupbox

The Temperature Groupbox (2) contains gauges and labels to read four temperature measurements: ambient air (2.1), compressed air (2.2), combustion chamber (2.3), and exhaust temperatures (2.4). The measurements are displayed in degrees Fahrenheit, and each gauge has an upper limit that is high enough to display the expected temperature for that location. For example, the Ambient Air Temperature Gauge has an upper limit of 90°F while the Chamber Temperature Gauge has an upper limit of 2200°F.

Pressure Groupbox

The Pressure Groupbox (3) contains gauges and labels to read two pressure measurements: ambient air (3.1) and compressed air (3.2). Both pressure measurements are displayed in psi.

Humidity Groupbox

The Humidity Groupbox (4) contains a gauge and label (4.1) to display percent humidity of the ambient air. This gauge has a range of 0 to 100 %.

Shaft Speed Groupbox

The Shaft Speed Groupbox (5) contains a gauge and label (5.1) to display the shaft speed of the turbine.

Flow Rate Groupbox

The Flow Rate Groupbox (6) contains the Flow Rate Textbox (6.1). You must record the flow rate of the exhaust using the provided handheld anemometer. Enter this value into this textbox. This textbox will only accept a numerical value. If a nonnumerical input is attempted, Engine Control Tool will change the value in this textbox to 0.

Log Groupbox

The Log Groupbox (7) contains the Log Message Textbox (7.1) where you may enter a custom message to mark some specific timestamp in the Excel output. If you would like to mark some timestamp with a custom message, type your message into the Log Message Textbox. Click the Log Button (7.2) at the moment you would like to log, and the textbox input will be added to that timestamp in your final Excel spreadsheet. If you would like to mark some timestamp but not include a custom message, simply click the Log Button at the moment you would like to mark while leaving the textbox empty, and that timestamp will be marked in the Excel output.

Engine Control Tool Quick Start

The following steps describe how to use Engine Control Tool assuming it has already been installed on your laptop. Numbered items correspond to the numbering used in the Engine Control Tool Diagram section above.

1. Connect the provided USB cable to the Arduino and your laptop. The lights on the Arduino should turn on.
2. Open Engine Control Tool and click “Close” on the Syncfusion License notice that pops up.
3. At this point, Engine Control Tool should be displaying the data from the sensors. If not, refer to the troubleshooting section of this guide.
4. Run the engine. Engine Control Tool will be reading in new data from the sensors once every second. You will see this real time data displayed on the Engine Control Tool screen.
5. Once the engine is running, measure the flow rate of the exhaust using a handheld anemometer, and enter this value into the Flow Rate Textbox (6.1) on Engine Control Tool.
6. When ready to export the recorded data to Excel, click the Excel Button (1.1).

Programmer’s Manual

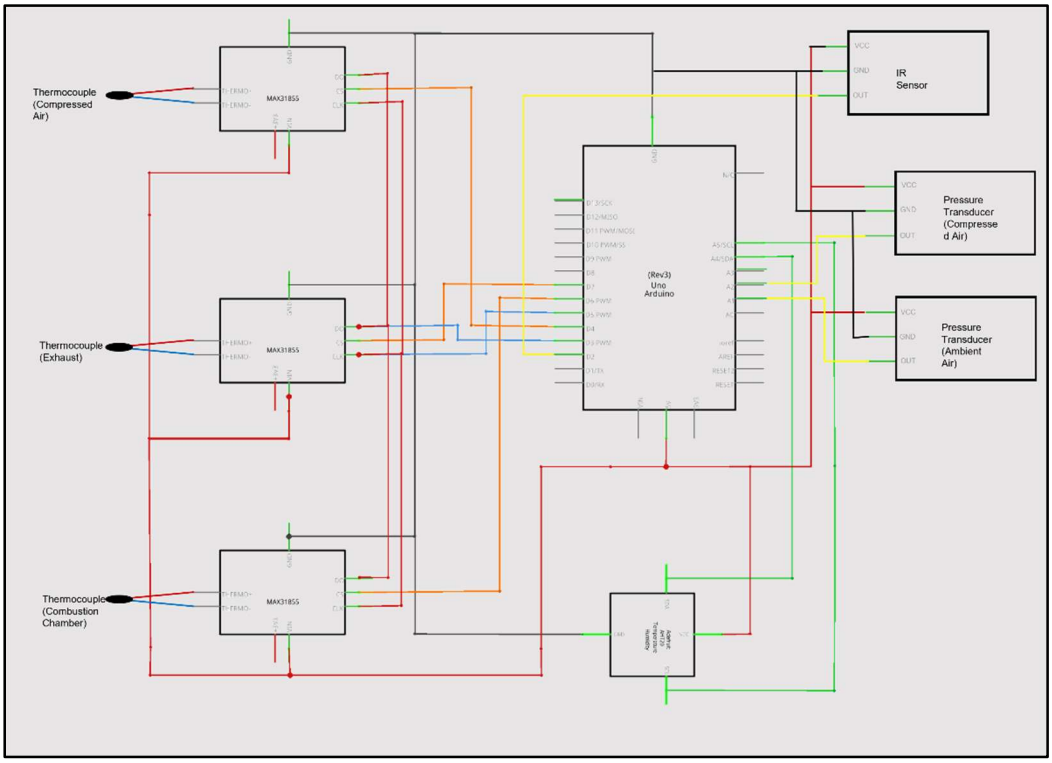
The following sections explain the different sections in both the Arduino source code and Engine Control Tool source code. You may use this information to alter the source code to fit your requirements.

Arduino Source Code

This section should be referenced if you are changing the source code used by the Arduino. The Arduino source code is written in the file *CombinedProgram.ino*. You may access this file by following either method described in the Installation section of this manual. The Arduino source code is divided into four sections. In the source code, these are labeled Section 0 to Section 3 with comments. So, you will be able to easily find the section being discussed by using the find function (CTRL + F). We begin with Section 0 in *CombinedProgram.ino*.

Section 0. Arduino Uno Connections

This section contains only comments. It describes the required pin connections for each sensor used on the engine. A diagram of these connections can be seen below. The wires in the diagram are color coded to match the actual wires installed in the electrical box on the cart.



Section 1. Set Up Sensors

This section contains libraries and definitions that are required to set up each sensor. First, three libraries are included. These are:

- SPI.h is the serial peripheral interface library and is used by the Arduino to communicate with the connected sensors.
- Adafruit_MAX31855.h is a library that allows the Arduino to use the built-in functions associated with the Adafruit MAX31855 thermocouple amplifier breakout board.
- Adafruit_AHTX0.h is a library that allows access to built-in functions that control the Adafruit AHT20 temperature/humidity sensor.

After the library definitions, the thermocouples are the first sensors that are initialized. Each thermocouple is connected to its own Adafruit MAX31855 thermocouple amplifier breakout board. All three MAX31855 boards share a clock and data out pin, and they each have their own chip select pin.

Next, the pressure transducers are set up by allocating two analog pins and setting their initial reading to 0. Following the pressure transducers pin assignment, there is one line of code that creates an instance of the Adafruit AHT20 temperature/humidity sensor. Lastly, the shaft speed sensor is prepared

by defining variables and an interrupt routine that will be used to count the revolutions made by the turbine shaft.

Section 2. Set Up Program

This section contains the setup() function included in any Arduino program. First, the two analog pins for the pressure transducers are designated as input pins. This allows the Arduino to read the value from these transducers.

Next, the attachInterrupt function is called. This causes the infrared shaft speed sensor to begin counting revolutions. Finally, the three MAX31855 boards and the AHT20 sensor are all started based on how they were defined in the previous section.

Section 3. Loop Section

This is the final section of source code in *CombinedProgram.ino*. This section contains the loop() function, another requirement in Arduino code. This section is the code that continuously runs if the Arduino is connected to a power source, in this case, if the Arudino is connected to your laptop.

First, a measurement is taken from each of the three thermocouples and stored in three strings. Next, a reading from each pressure transducer is taken. After receiving this analog reading for each transducer, there are five lines of code that convert the reading to psi. The pressure transducers are calibrated to read ambient pressure correctly, and from there, a linear relationship between the transducer's voltage output and pressure is assumed. If the pressure sensor seems to be reading incorrectly, you may need to change the calibration so that it correctly reads ambient pressure. To do this, there are two variables that you may adjust. The first is a float called "calibration1" and the second is another float named "calibration2". The value "calibration1" corresponds to the ambient air pressure transducer, and "calibration2" corresponds to the compressed air pressure transducer. To increase the reading from the pressure transducer, decrease its calibration variable and vice versa.

Next, a reading is taken from the AHT20 temperature/humidity sensor, and finally, the shaft speed sensor is read and converted to an RPM measurement.

Lastly, all these measurements are combined into one string with a space between each measurement, and this string is sent to the serial port, where it is received by Engine Control Tool. A one second delay occurs, and then, the Arduino repeats this section.

Engine Control Tool Source Code

This section should be used if you would like to make changes to the Engine Control Tool source code. Engine Control Tool source code is in four files, three are written in C# and one is written in Extensible Application Markup Language, or XAML. Each file will be examined in the following sections. Like the Arduino source code, the sections discussed below are commented with the same titles in the source code, so you may easily locate the desired section with CTRL + F.

File 1. MainWindow.xaml

The file *MainWindow.xaml* is written in XAML, and it defines the visual elements of Engine Control Tool. The source code for this file is divided into three main sections.

Section Title	Description	Approximate Lines in Code
Section 1	**Defines multiple style for the components on the GUI. These style definitions are used in the later sections to determine the visual aspects of different components.**	**19 – 236**
Section 2	**Contains the source code to define the Menu Bar (Section 2.1) and the Body Grid (Section 2.2). The Menu Bar contains the Excel Button, Mode Button, and Color Button while the Body Grid contains the six groupboxes that display measurements and receive user input**	**238 – 569**
Section 3	**Defines the label that will flash on the screen if a dangerous temperature or pressure level is measured**	**571 – 613**

File 2. MainWindow.xaml.cs

The class *MainWindow.xaml.cs* contains the C# code controlling the overall functioning of Engine Control Tool. When the program is run, this file creates instances of the other two classes, *ViewModel.cs* and *CreateExcel.cs*. It also contains methods that are called whenever the user clicks a button on the GUI. These include the Mode Button, Color Button, Excel Button, and Log Button. This class also opens the correct serial port and begins reading data from the Arduino as soon as the program is started. Finally, it starts a separate thread that continuously monitors the temperature and pressure measurements. If either of these reach a dangerously high level, it flashes a warning label to the user and instructs them to terminate the flow of propane. Rather than having comments dividing this file into sections, comments divide this file into each method that it contains. The methods contained in this file are discussed in the table below. You may find these methods in the source code by using CTRL + F and searching for the method number.

Method Title	Description	Approximate Lines in Code
Method 1	**Constructor for the MainWindow class. Initializes GUI, connects MainWindow view to view model for passing data between classes, starts a background thread to check for high temps/pressures**	**54 – 69**
Methods 2 – 11	**Called whenever the user selects from the Mode Button or the Color Button. Simply set correct variable in view model to change colors on GUI**	**71 – 132**
Method 12	**Called when Engine Control Tool starts. Searches for connected serial port**	**134 – 172**
Method 13	**Receives data from connected serial port**	**174 – 192**
Method 14	**Takes received data, updates the GUI to display data, and records data in a list variable in the view model**	**194 – 315**
Method 15	**Called when user clicks the Excel Button. This calls methods from the CreateExcel class to generate the final Excel spreadsheet**	**317 – 342**

Method 16	**Called when user changes the entry in Flow Rate Textbox. Confirms that the entry is numerical, and if not, changes the content of the Flow Rate Textbox to 0**	**344 – 362**
Method 17	**Runs on a loop beginning when Engine Control Tool is started. Checks for a temperature or pressure that is over the allowed threshold, and notifies user if so**	**364 – 409**
Method 18	**Checks to see if Log Button has been clicked. If so, set required variables in view model**	**411 – 419**

File 3. CreateExcel.cs

The *CreateExcel.cs* class is responsible for generating two Excel spreadsheets, one in imperial units and one in metric units. This class contains four methods besides the constructor. Three of these methods are called from *MainWindow.xaml.cs* when the user clicks on the Excel Button. This class uses the library *Microsoft.Office.Interop.Excel* to interface with the installed version of Excel.

Method Title	Description	Approximate Lines in Code
Method 1	**Constructor for the CreateExcel class. Gives this class access to the view model in case data must be passed between this class and other classes**	**28 – 34**
Method 2	**Starts new Excel application. Creates two Excel spreadsheets and names them**	**36 – 53**
Method 3	**Populates one of the Excel spreadsheets with data given in metric units**	**55 – 176**
Method 4	**Populates the other Excel spreadsheet with data given in Imperial units**	**178 – 297**
Method 5	**Draws a thick border around Excel cells. Called by Methods 3 and 4**	**299 – 305**

File 4. ViewModel.cs

The final class*, ViewModel.cs*, is a model class. Its only function is to store data so that data may be passed between the XAML view and other classes. This is essential because the only thread that can interact with the GUI elements is the main thread. However, the serial data being read from the Arduino is being received in a separate thread. This thread cannot access the GUI elements, so it instead passes its received data to the view model. The GUI elements are bound to the view model, so they can see the stored data. This is how the gauges and labels in the GUI are updated to show the current data.

The view model also contains lists that are constantly being added to as new data is received. These lists are accessed by the *CreateExcel.cs* class and are used to populate the Excel spreadsheets. The *ViewModel.cs* class is organized by sections.

Section Title	Description	Approximate Lines in Code
Section 1	**Contains definitions that allow the GUI elements to be altered when new data is received.**	**18 – 39**
Section 2	**Contains 3 variables that are changed based on the selections made from the Mode Button or Color Button**	**41 – 67**
Section 3	**Contains 11 variables that store the real-time measurements and mark whether the user would like to log some specific timestamp**	**69 – 165**
Section 4	**Stores all the lists that are used to record data and to populate the Excel spreadsheet**	**167 – 257**

Section 5	**Contains 2 variables that are changed if the warning label needs to be visible**	**259 – 285**

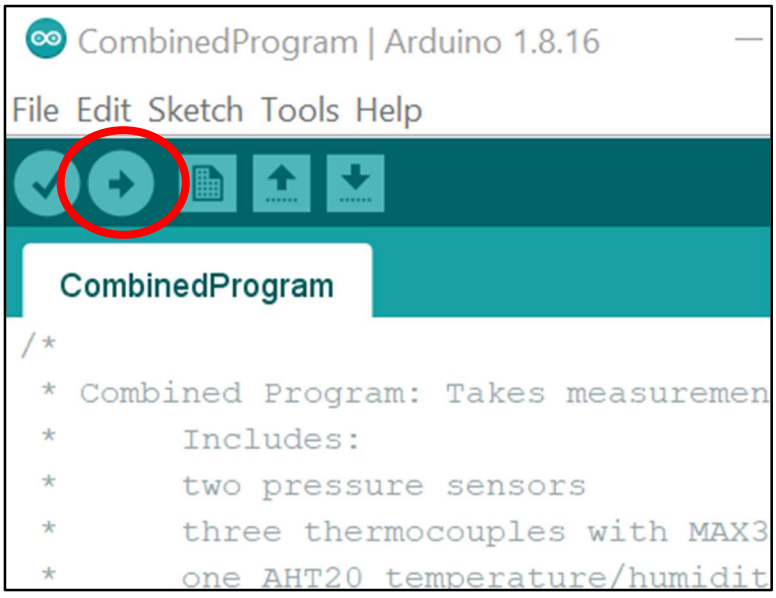
Troubleshooting

The ports “COM3”, “COM4”, and “COM5” are not available on my device.

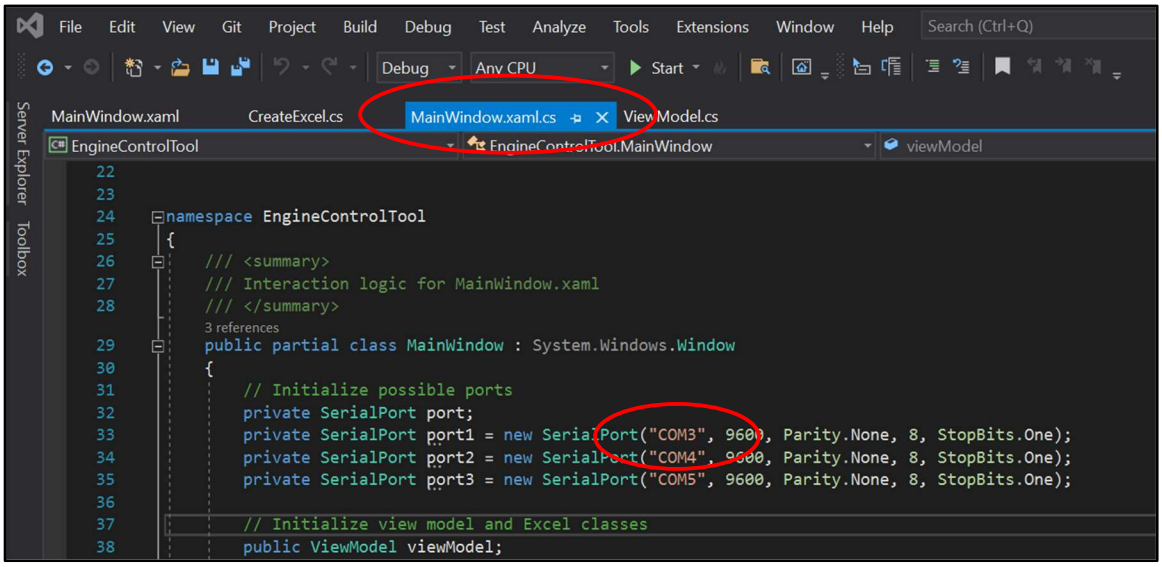
The Arudino installed on the engine is compatible with serial ports “COM3”, “COM4” or “COM5”. If none of these are available on your device, you will have to upload the Arduino code to the Arduino using your available port, and you will need to change the Engine Control Tool source code to read from an alternative port.

Begin by uploading the Arduino code through your chosen port. First, connect the Arduino to your laptop with the provided USB cable. Open the Arduino code (*CombinedProgram.ino*) in the Arudino IDE. On the menu bar, select Tools < Port < COMx. The Arduino IDE will automatically recognize which port you connected to the Arduino with. For example, if you connected to COM6, you would select Tools < Port < COM6.

Next, click on the upload arrow to upload the code to the Arudino.



Now, you must alter the Engine Control Tool source code to recognize data from your selected port. Open the Engine Control Tool source code in Visual Studio following the steps described in the Installation section. Go to the file *MainWindow.xaml.cs*. On line 33, change “COM3” to whichever port you selected in the Arduino program. For example, if you chose “COM6” in the Arduino IDE, change “COM3” to “COM6” on line 33 of this file.



The program may now be run on your device following the steps described in the Installation section.

Occasionally, two of the thermocouple readings are lost.

The MAX31855 thermocouple amplifier breakout boards that are used to take measurements from the thermocouples do not accept readings from grounded thermocouples. So, if any of the thermocouples are touching each other, it will register as an error in the reading. The thermocouple installed directly after the combustion chamber has no metal sheath on it. Thus, when the air in the combustion chamber is too turbulent, it is possible that this thermocouple touches the edge of the combustion chamber. When this happens, since the exhaust thermocouple is mounted to the metal cart, it becomes grounded with the combustion chamber thermocouple, causing an error in the measurement.

This problem will typically fix itself in a matter of seconds, as the combustion chamber thermocouple is blown away from the inside wall of the combustion chamber. As soon as the thermocouples are not grounded, accurate measurements will be recorded again.

The Excel spreadsheet was not fully populated.

If you click inside the Excel spreadsheet before it is finished generating, it will cause an error and stop the spreadsheet generation. Simply keep Engine Control Tool running, click the Excel Button to generate the spreadsheet a second time, and let the spreadsheet fully populate before you begin manipulating and analyzing the data.

Engine Control Tool is not displaying any data.

Restart the Engine Control Tool Application.

If this does not fix your problem, check that the Arduino is connected to the proper port by following the steps described in the first part of this Troubleshooting section.

If the problem persists, open the electrical box, and confirm that no connections have become loose in the Arduino circuit.