

Table of contents

[Project Plan Summary 2](#_1fob9te)

[Modules and Subassemblies 3](#_3znysh7)

[Microcontroller Unit 3](#_2et92p0)

[Distance Sensor 1](#_tyjcwt)

[Stepper Motor Driver/MCU 2](#_3dy6vkm)

[Bluetooth/MCU 3](#_1t3h5sf)

[Application/Bluetooth 3](#_4d34og8)

[Hierarchical design diagrams \ Schematics (Wiring Diagrams) 3](#_2s8eyo1)

[Microcontroller Unit 3](#_17dp8vu)

[Distance Sensor 7](#_3rdcrjn)

[Stepper Motor Driver/MCU 8](#_26in1rg)

[MCU/Bluetooth 11](#_lnxbz9)

[Test Plan 12](#_35nkun2)

[Microcontroller Unit 12](#_44sinio)

[Distance Sensor 13](#_2jxsxqh)

[Stepper Motor Driver/MCU 13](#_z337ya)

[Bluetooth/MCU 13](#_3j2qqm3)

[Application/Bluetooth 14](#_1y810tw)

[Software Specification 15](#_4i7ojhp)

[Microcontroller Unit 15](#_2xcytpi)

[Distance Sensors 19](#_1ci93xb)

[Stepper Motor Driver/MCU 21](#_3whwml4)

[MCU\Bluetooth 22](#_2bn6wsx)

[Application\Bluetooth 23](#_qsh70q)

[Individual Team Member Assignments 25](#_3as4poj)

[Preliminary Cost 26](#_49x2ik5)

Oregon Institute of Technology  
junior project plan

# Project Plan Summary

Our project plan explains in detail how we intend to implement our “Focus Point System” for the junior project. The “Module and Subassemblies” section gives a high-level component overview of the modules we think need to be in place per the Junior Project requirements. The “Hierarchical diagrams and Schematics” section dives deeper into each component and shows state machines, data flow charts and other logic so far considered for implementation. The “Software Specification” section will explain the software functionality of each component with respect to the MCU and will include data flow charts and UML sequence diagrams. The team has already ordered parts and are currently testing these components. The “Test plan,” section shows who is going to test what modules and how we intend to test these mdoules. Pseudo code and programming concepts are presented in the “Software Specification” section. Although we are all working together on all aspects of the project, each team member has selected specific areas they wish to explore. This is listed in the “Team Assignments” section followed by our cost up to date in the “Preliminary Cost” section.

# Modules and Subassemblies

Our modules and subassemblies are broken down to five main modules. The MCU module has the two phases for development of device (GPIO, PWM, TIMERS, UART) configuration for connecting to peripheral devices. The distance sensor module, which consists of connecting the sensor to the MCU which will be using trigger and echo system through two GPIO pins. The stepper motor driver module consists of one subassembly which connects the driver to the MCU and to the motor itself with an additional power supply. The Bluetooth and MCU module focus on transmit and receive abilities and functionality. The application and MCU module are focused on connecting the MCU to a mobile phone application. Android is the current platform goal for the application. Our goal is to have the MCU not just receive commands but send information back to the application such as current distance to focus object.

### Microcontroller Unit

The MCU has a two-phase development plan. The first is to configure devices to interact with all peripherals. Several devices have been identified that are possibilities fordata flow and communication. The second phase is fine-tuning the configuration and applying features of the MCU that can produce a smaller code footprint and refine operations to closer tolerances for smooth accurate organic and behavioral focusing.

**Phase 1 (Configuring Device Communications)**

* GPIO
* PWM
* TIMERS
* UART
* INTERRUPTS

**Phase 2 (Fine Tune Configurations)**

* Code refinement to decrease file size.
* Use of power options, hibernation modes
* Fine tune timing and clocks
* Make use of other features of the MCU

### Distance Sensor

The distance sensor that was chosen is the HC-SR04 ultrasonic distance sensor. This was chosen for its simplicity of use and accuracy. This will take a trigger signal from the MCU, and then the sensor will send out 8 ultrasonic pulses and use time of flight (TOF) to determine the distance that it is sensing. It will then output a high signal to the MCU, and the timing of how long it is high, will determine the distance.

**Device Features:**

* 2 cm – 400 cm range
* Ultrasonic transceiver
* TTL 3.3 V logic for input and output
* 15° measuring angle

### Stepper Motor Driver/MCU

A DRV8833 module drives the stepper motor. This is our first option because of its heavily documented use with the microcontroller we are currently testing with. The team is considering using other options such as a micro stepper motor driver to give us more granularity to focus. Development of the gears for the motor will start when we have the rest of the module operating.

**Device Features:**

* Dual-H-Bridge Current-Control Motor Driver
* Output Current (at VM = 5 V, 25°C)
* Wide Power Supply Voltage Range 2.7 to 10.8 V
* PWM Winding Current Regulation and Current Protection

### Bluetooth/MCU

The HC-05 Bluetooth module will be used to communicate the data input on the application to the microcontroller. We picked this Bluetooth controller because it has an abundance of documentation to ease with programming, and the power consumption is small for a Bluetooth device. All the information will be transmitted through a UART, specifically the UART1. The information will be processed by the microcontroller, and the MCU will use the data accordingly.

### Application/Bluetooth

The application will be an Android app. It will be used as a means for the user to input information regarding specifics such as the camera lens being used if the distance will be manually set or automatically polled. If the user wants to set the distance manually, they can also set the time it takes to blend the focus to the target, allowing for more control with the focus style.

# Hierarchical design diagrams \ Schematics (Wiring Diagrams)

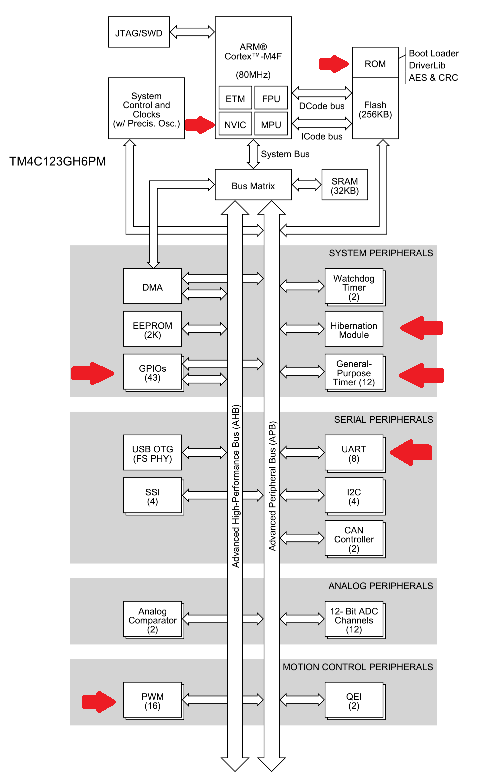
This section gives a high-level visual snapshot of each of the module’s implementation. Images and specifications are gathered from the datasheets, manuals and help documents of our chosen parts.

### Microcontroller Unit

The devices the team is currently looking to implement on the TM4C123GH6PM are highlighted by arrows in image “TM4C123GH6PM High-Level block diagram” from the MCU documentation published by Texas Instruments below. The features of these devices are listed in table “Device Features Table.

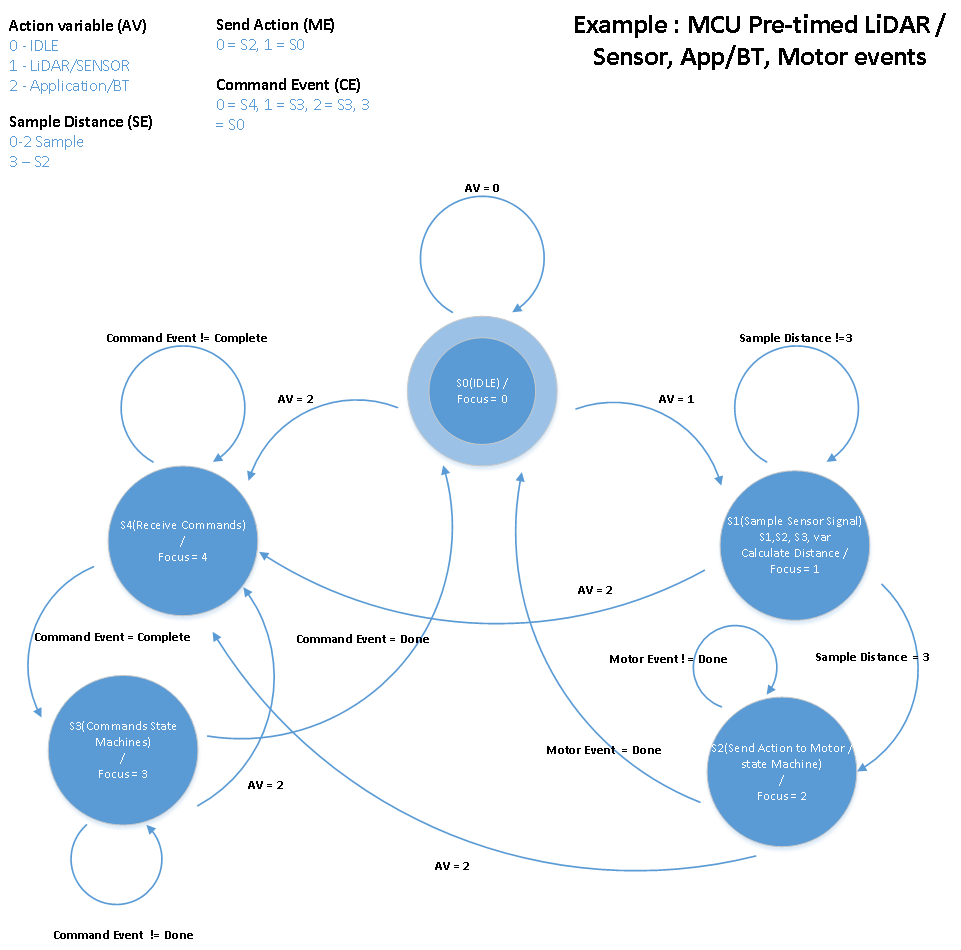
|  |  |
| --- | --- |
| Device Name | Features that Interest Team |
| ROM | Make use of libraries in ROM. Advanced Encryption Standards |
| NVIC | Nested Vector Interrupt Controller has Tail-chaining and other low-latency features |
| GPIOs | Highly configurable, can trigger on edge or level, |
| PWM | A pair of complementary signals with dead-band generation (for H-bridge circuit protection) |
| Hibernation Module | Still researching if this can be of assistance. May be a stretch goal. |
| General Purpose Timers | Any pin can be interrupt, all pin states saved during Hibernation mode |
| UART | Separate 16x8 bit transmit and receive FIFOs  Programmable baud rate generator  Stretch goal - µDMA support |

**Device Features Table**



**TM4C123GH6PM High-Level block diagram**

Image “Auto Focus State Machine” below shows the state machine diagram for the basic operation of the microcontroller unit when polling data from the distance sensor. We are also working on other possible behaviors like defined length focus or timing focus at different distance.



**Auto Focus State Machine**

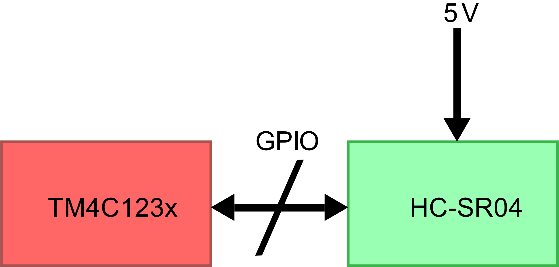
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Current State** | | **Next State** | | | | | | | | | | | | **Output** |
|  |  | **(AV) Action Variable** | | | **(SI) Sample Input** | | | **(ME) Motor Event** | | **(CE) Command Event** | | | | **Focus Status** |
|  |  | 00 | 01 | 10 | 00 | 01 | 10 | 0 | 1 | 000 | 001 | 010 | 011 |  |
| **S0** | **00001** | S0 | S1 | S4 | xx | xx | xx | xx | xx | xx | xx | xx | xx | **000** |
| **S1** | **00010** | xx | xx | S4 | S1 | S1 | S2 | xx | xx | xx | xx | xx | xx | **001** |
| **S2** | **00100** | xx | xx | S4 | xx | xx | xx | S2 | S0 | xx | xx | xx | xx | **010** |
| **S3** | **01000** | xx | xx | S4 | xx | xx | xx | xx | xx | xx | xx | S3 | S0 | **011** |
| **S4** | **10000** | xx | xx | xx | xx | xx | xx | xx | xx | S4 | S3 | xx | xx | **100** |

### Distance Sensor

The HC-SR04 ultrasonic distance sensor uses TOF to determine the distance of the object in front. The applications for this device is filmmaking, depth sensing, and robotics. Below is the “Distance Sensor Block Diagram.”

**Design Features:**

* Requires two pins for controlling the sensor, trigger and echo
* Use of change notification and timer for echo pin

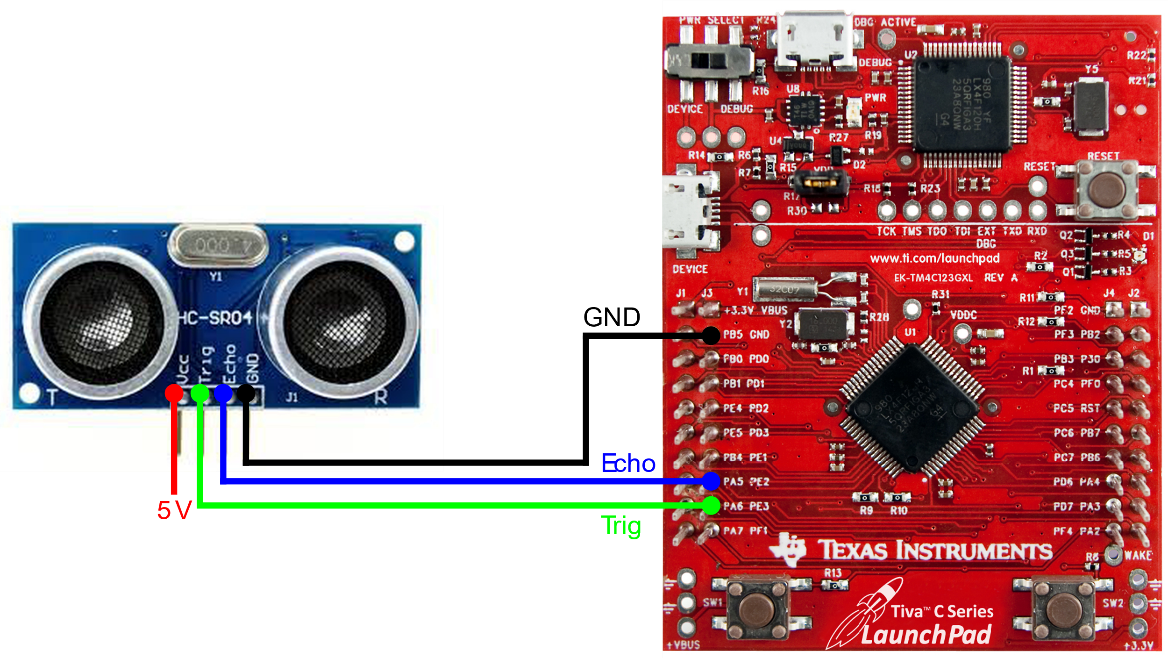


**Distance Sensor Block Diagram**

Below is the pin table and wiring diagram, showing which pins the HC-SR04 will be using on our TM4C123GH6PM. The HC-SR04 requires two GPIO pins for operation. This is a single output pin to trigger the sensor and a single input to echo. The HC-SR04 VCC and GND pins are then powered by 5V and a connected to a common ground respectively.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Device Name** | **Device Pins** | | | |
| TM4C123xxx Launchpad | N/A | PE2 | PE3 | GND |
| HC-SR04 | 5V | Echo | Trig | GND |

**Distance Sensor Pin Table**



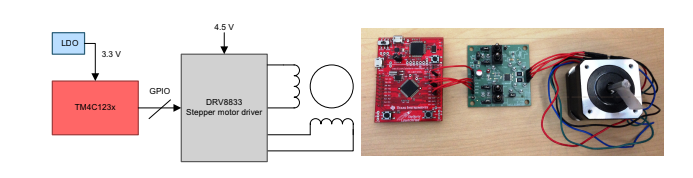
**Distance Sensor Pin Diagram**

### Stepper Motor Driver/MCU

The Dual H-Bridge Motor Driver can drive two DC Motors or One stepper. A few of its applications include Video applications and robotics. It provides internal shutdown functions for overcurrent protection as well as other protections and a low powered sleep mode to conserve energy. Image “Stepper Motor Block Diagram” Shows the high-level concept.

**Design Features:**

* Uses 4 GPIO pins to control the Output of the H-Bridge drivers in this module and uses a General-Purpose Timer to generate the PWM signals for driving the motor.
* The software is Designed to work with our MCU

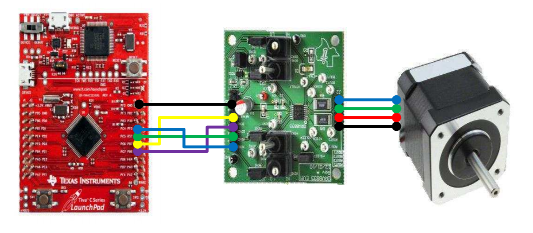


**Stepper Motor Block Diagram**

The wiring diagram below shows how the MCU, stepper motor driver, and stepper motor will be connected. Table “MCU and DRV8833 Pin Table”, shows the pin connection of the two evaluation boards.

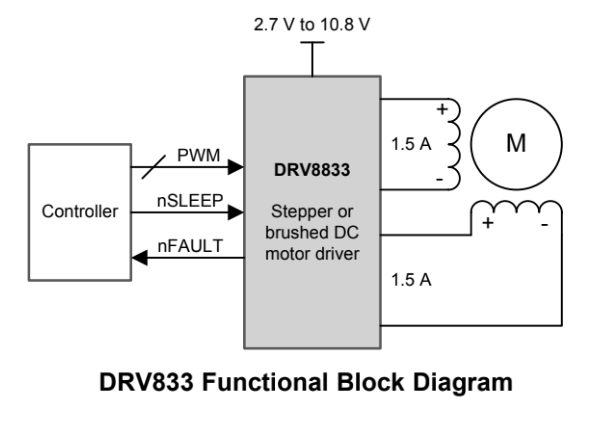
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Device Name | Device Pins | | | | |
| TM4C123xxx Launchpad | PC4 | PC5 | PC6 | PC7 | GND |
| DRV8833 | AIN1 | AIN2 | BIN2 | BIN1 | GND |

**MCU and DRV8833 Pin Table**



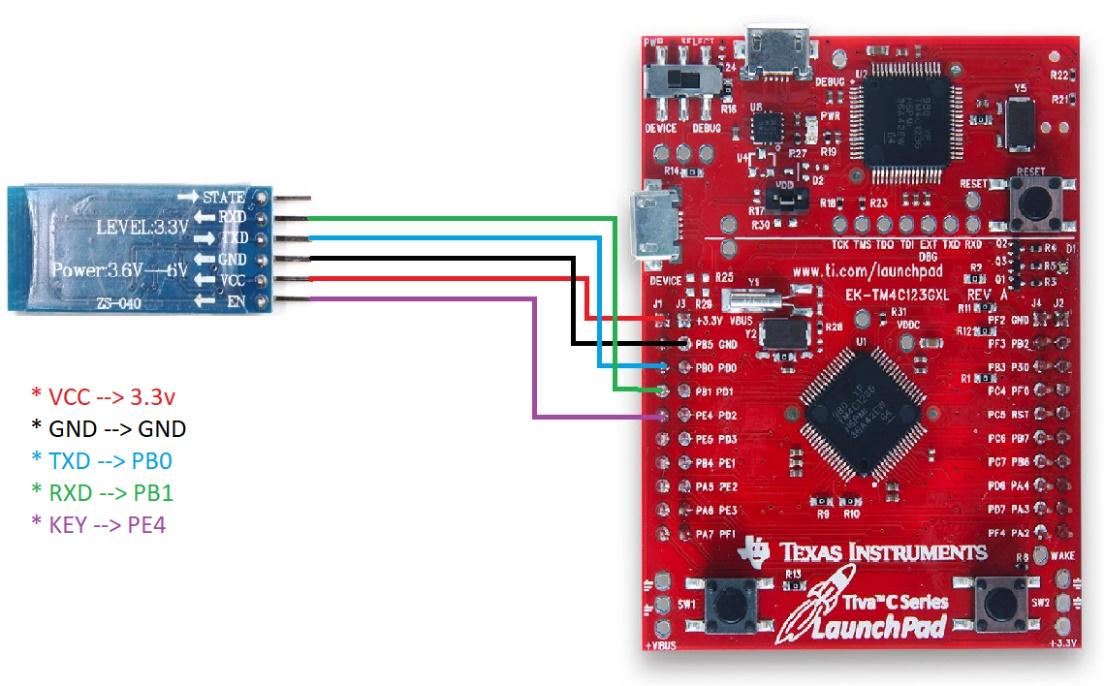
**MCU – Stepper Motor Driver - Stepper Motor (Wire Diagram)**

The DRV8833 works with brushed DC or bipolar stepper motors. This device also includes a low power sleep mode. Another possible stretch goal may be to fine-tune the features of the driver unit. Features like the nSLEEP or Overcurrent Protection.



### MCU/Bluetooth

The HC-05 Bluetooth Module uses Serial Port Protocol. The input voltage is DC 5v which may need to be supplied from an external source since the TI board can only output 3.3v over a GPIO pin. All the data transmission will be sent using a UART module, with both transmit and receive capabilities. The key will be used to toggle between Data Mode (when low) and AT mode (set high). By default, it will be low, putting the Bluetooth device into Data mode, which will put the Bluetooth module in an idle state, waiting for data to be present. AT command mode is used to program and debug the Bluetooth module, and will be used only during the development phase of the project. Being in a default state of data mode allows the user of the Android application to send data from the entry fields in the application to the microcontroller. The state pin will not be used for our application. Below is the wiring diagram and pin table for the Bluetooth to MCU module.



**Bluetooth Wiring Diagram**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Device Name | Device Pins | | | | |
| TM4C123xxx Launchpad | N/A | GND | PB0 | PB1 | PE4 |
| HC-05 | VCC | GND | TXD | RXD | KEY |

**Bluetooth Pin Table**

# Test Plan

As parts come in, we intent to use the example code from the manufactures to test with our TI ARM boards or Arduinos to ensure that the parts are not defective. We want to do this as soon as the parts arrive, so we have time to reorder if the part is defective.

### Microcontroller Unit

All team members will be working on testing their personal MCU boards. Each will be responsible for going through the Texas Instruments supplied workbook for setting up Code Composer and testing out MCU features.

* + - 1. Ensure UART modules are working
      2. Ensure PWM modules are working
      3. Ensure GPIO modules are working
      4. Ensure Timer modules are working

### Distance Sensor

Andrew K will be performing tests on the distance sensor using an Arduino Uno and example code provided by SparkFun.

Ensure distance is being sensed accurately

Ensure distance is being sensed with precision

Compare data gathered with real world measured data

This testing will ensure that the sensor is functional and within the tolerances specified by the manufacturer. This testing method will allow easy integration and understanding of the functionality of the sensor.

### Stepper Motor Driver/MCU

Michael will test with TI board using lab examples provided by Texas Instruments. The sample code provides configuration for the timers, clock and GPIOs needed.

1. Ensure motor can move using PWM
2. Test functionality of speeding up and slowing down of motor

### Bluetooth/MCU

Andrew Webb will be testing the functionality and usage of the Bluetooth module. All the testing will be done using an Arduino Uno as there is supported libraries to test the Bluetooth device.

* + - 1. Sending and receiving UART transmission to both the Microcontroller and the Application
      2. Transmission of data will include sending simple information such as a string or single character

The Arduino will also allow for testing of pairing the Bluetooth module with a mobile phone or other Bluetooth capable device. Once testing for functionality has been completed, the rest of the testing will be completed while using the TI microcontroller.

### Application/Bluetooth

Brandon will be performing a Test Plan on the Android application that follows the plan of action as listed below:

1. Ensure data can be manually set in all data fields by the user
2. Ensure that no data field can be empty upon trying to send data via Bluetooth
3. Ensure the Android Application works on a couple different Android devices
4. Ensure that once the Android Application is fully functional that the data can be transmitted via Bluetooth to the MCU
5. Ensure that the current distance reading can be received via Bluetooth to the Android device and displayed within the GUI

The Android application is a big part of making our project a unique experience for the end user. Brandon along with Andrew Webb plan on testing most of the Bluetooth functionality together when it comes to sending and receiving data to and from the Android application. Once the group feels comfortable with how the Android application functions after completing the initial Test Plan we will begin working on incorporating a data base into the project that will allow us to store different lenses, and the formulas needed to focus the lenses with the stepper motor.

# Software Specification

This section contains all the software specifications for each module. It gives detailed descriptions such as UML or Sequence Diagrams on the software implementation and some of the requirements each module needs to provide.

### Microcontroller Unit

The MCU will be in control of receiving the data from the Android application and in return parsing the data into usable data to use for the servo control function. Other functionality that the functions will perform on the MCU are as follows: calculating number of pulses to send to the stepper motor based on the distance value variable, controlling sample intervals for the sensor, handling interrupts, sending the current distance via Bluetooth to the Android Application, and control timing for stepper control. The MCU will have at the least five functions one for Stepper Control, Sensor Data Acquisition, Distance to Pulse Conversion, Bluetooth Send, and Bluetooth Receive.

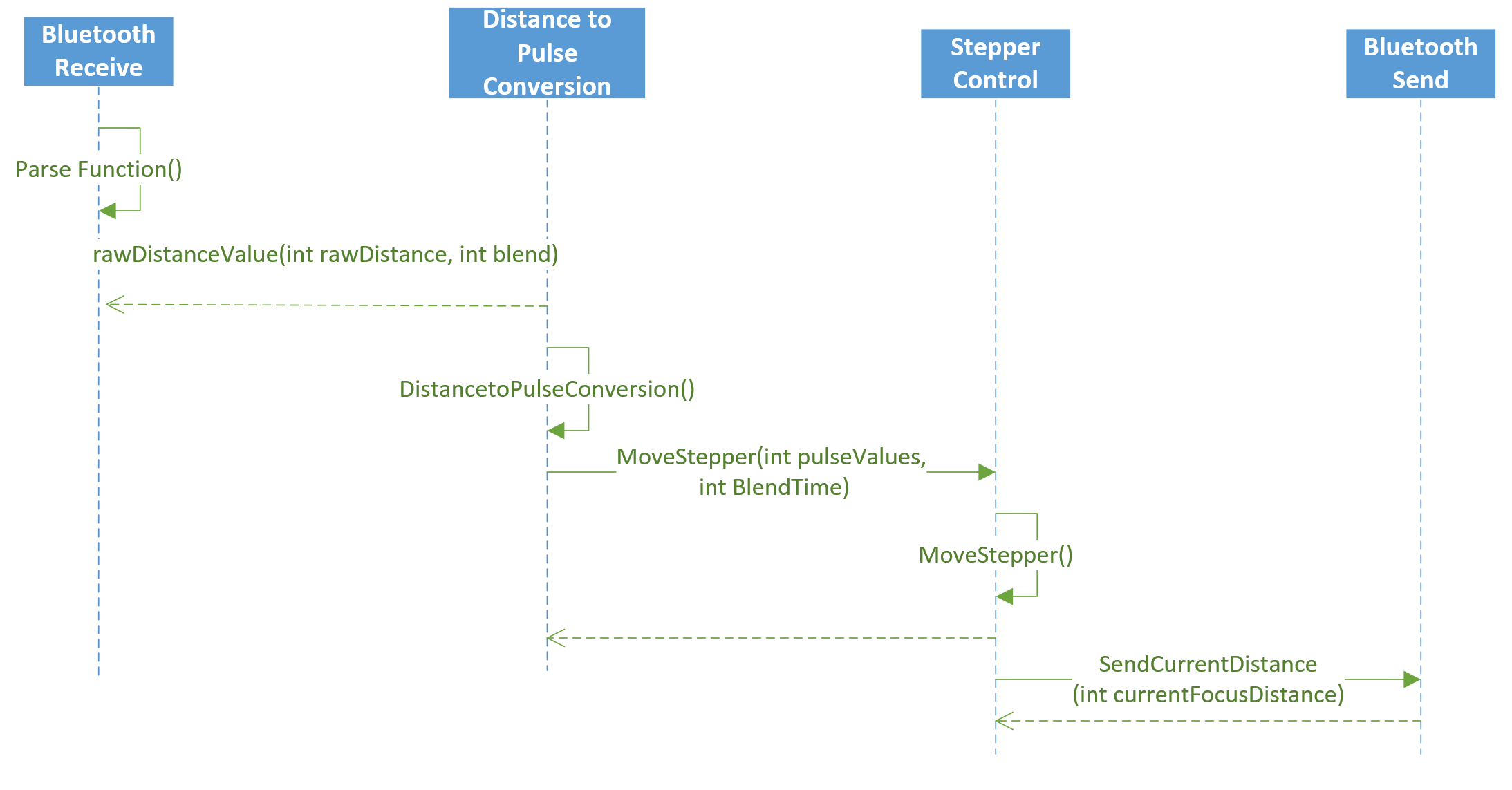
The Stepper Control function will take the data out of the Distance to Pulse Conversion function and use the setDistance variable to control the number of pulses needed to move the lens into the correct focus. If the user is passing in data from the app and there is a blend time that is received by the MCU, then the frequency at which the steps will be executed will be determined based on the number of pulses to achieve the set distance divided by the time requested to get to the distance, to ensure a smooth transition to focus. This will be controlled by using two of the MCU timers, and interrupts to ensure that the stepper motor starts and stops when it is supposed to without damaging it or the lens.

The Sensor Data Acquisition functions primary function will be to pull data from the sensor at a certain time interval and then check that each distance value is within 5% tolerance of the next distance value. If any of the distance values are off by more than 5% then all five distance variable values will be reset and five new distance values will need to be read in and collected/checked. Once five distance values have been collected and stored, the distances will be averaged together to get the distance that the lens needs to be focused to. This value will be stored in the readDistance variable and will be used in the Distance to Pulse Conversion function.

The Distance to Pulse Conversion function will use the readDistance value that was acquired by either the Sensor Data Acquisition function, or set by the user from the Android application, to determine how many pulses are needed and how frequent they need to be to achieve the correct focus in a specified or none specified time. This will be done by using a formula that is based on the distance per step on the lens divided by the readDistance variable data; which this answer will then need to be divided by the number of steps that our stepper motor has, to figure out how many pulses we need to get the lens to the correct focus. This methodology of figuring out pulse conversions based on distance values is not set in stone and is subject to change.

The Bluetooth send function will consist of only transmitting the readDistance value from the Sensor Data Acquisition function to the Android application via Bluetooth. The data that is transmitted to the Android application will be used to display the current focus distance that the sensor is picking up which will be displayed in the GUI of the Android application. The value will help the user to determine a manual distance to set as well as the blend time from the current distance to the manually set distance.

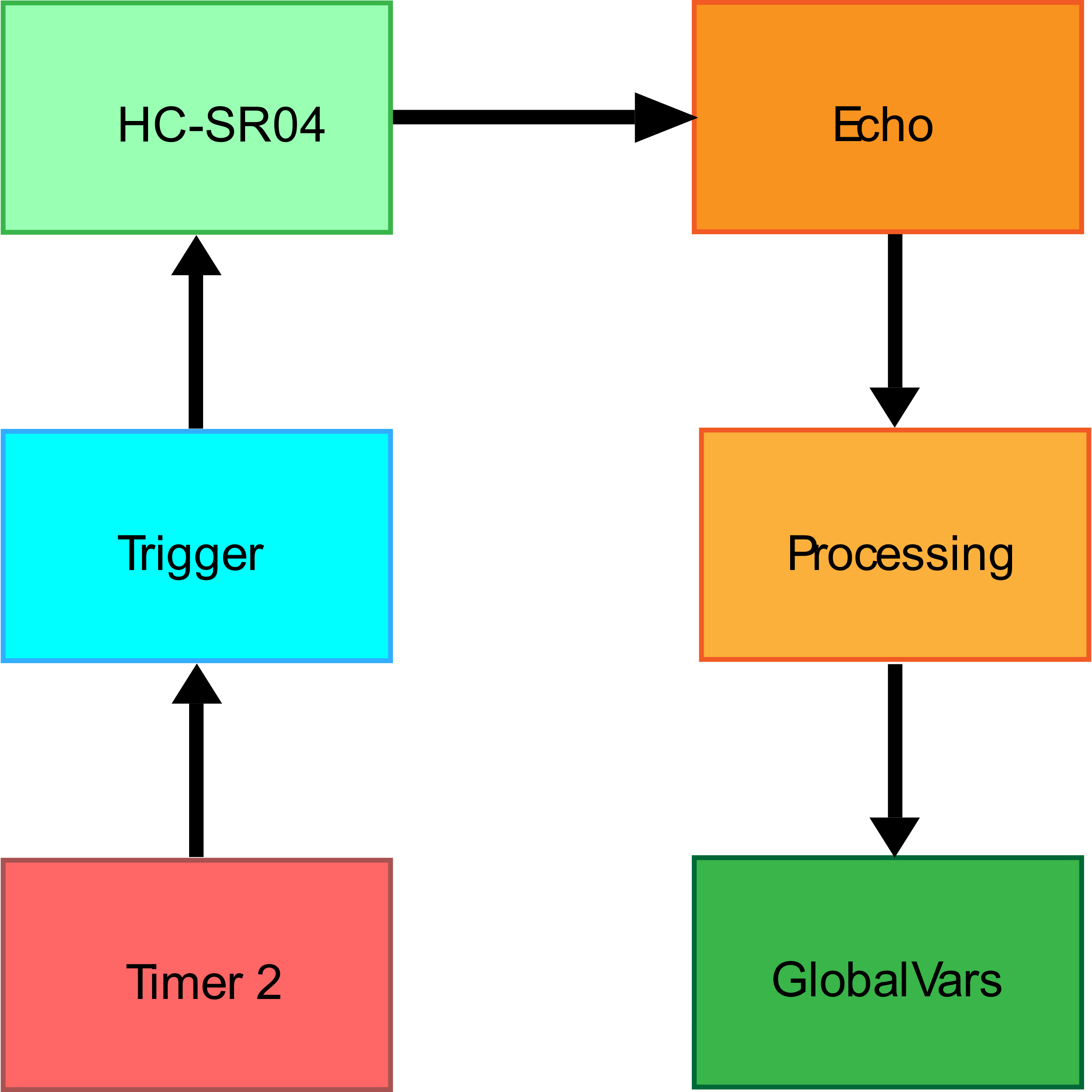
The Bluetooth read function will handle the data that is sent from the Android application to the MCU via Bluetooth. The data that is sent from the Android application will be parsed and placed into their MCU equivalent variables which will primarily be used by the Stepper Control function to ensure that the correct lens values are being used in the Distance to Pulse Conversion function before the Stepper Control function begins to focus the lens. If any preset distances are set by the user to focus to, or if a blend time is set by the user, this function will place the data in the necessary variables for the MCU to achieve the distance/speed combo to achieve the correct focus effect.

 **MCU to Bluetooth UML Sequence Diagram**

**(Assuming distance/blend data is sent via the Android Application.)**

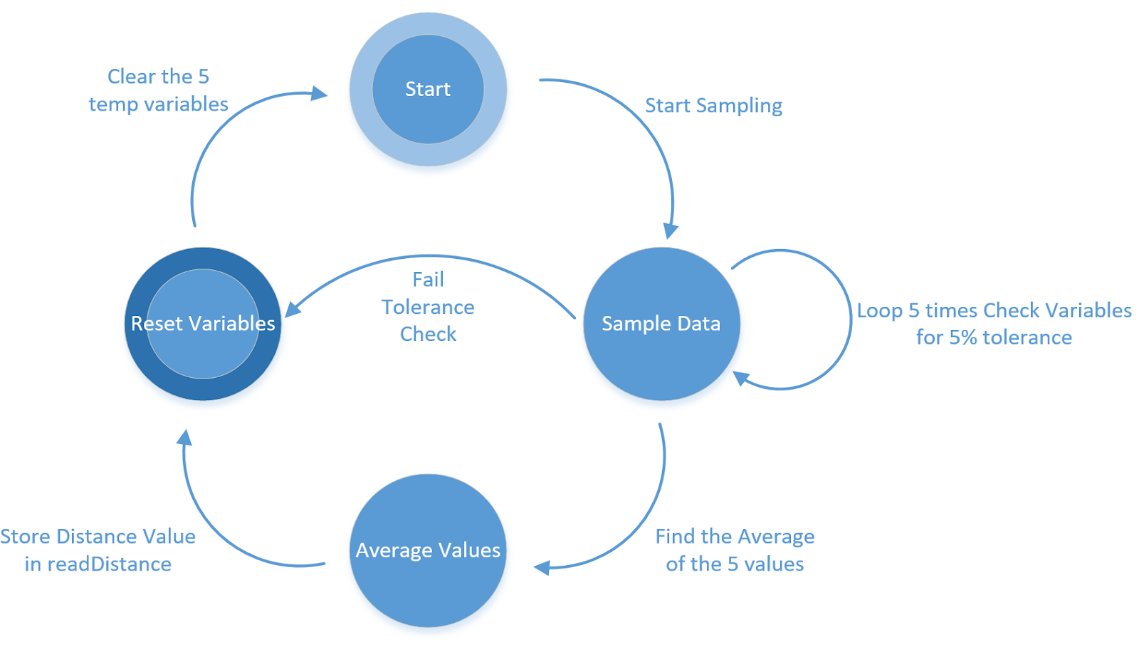
### Distance Sensors

The distance sensor will be sampled on an interval and triggered by Timer 2. When Timer 2 expires it will interrupt the MCU and send a trigger signal to the HC-SR05 for 10 microseconds. The MCU will then wait for a rising edge from the HC-SR05 and start a timer when it detects the rising edge. The MCU will then wait for a falling edge from the HC-SR05 and stop the timer. Then the MCU will process this timer value and calculate the distance from this timer value. The MCU will then update the distance global variable for the Stepper Motor Diver/MCU module to use.



**Distance Sensor Data Flow Chart**

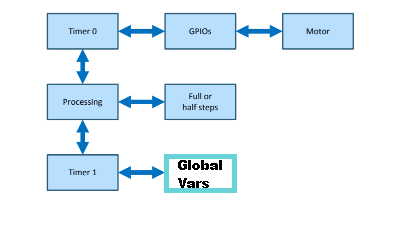
Below is the our “Distance Sensor State Machine”. This will be in a start state, until it starts sampling. While in the Sample Data state we will loop 5 times and check each variable for a 5% tolerance from the first measurement. Should the tolerance fail we will reset all variables and return to start. If the tolerance is met, then we will average the values and then calculate the distance of the object. The five temp variables will then be reset.



**Distance Sensor State Machine**

### Stepper Motor Driver/MCU

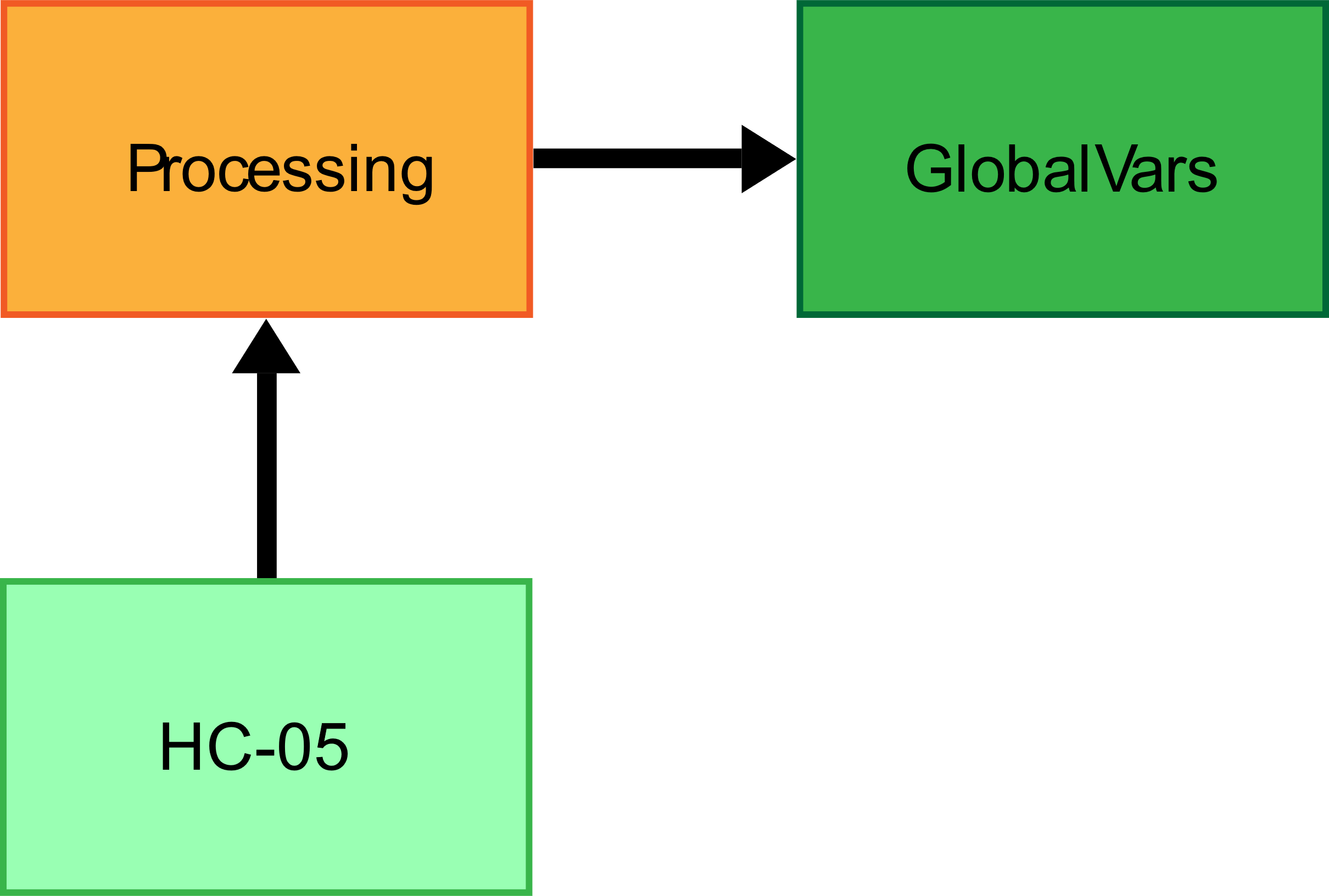
One of the timers on the TM4C123xxx drives the motor and interrupts the CPU in intervals. In the ISR the state of the four GPIO pins changes to drive the motor. The second timer checks the status of global variables to dictate motor behavior. This can come from the distance sensor or a command from the remote application via Bluetooth or other wireless communication. Time intervals and timing calculations are being researched.



**Software Architecture Block Diagram**

### MCU\Bluetooth

The Bluetooth module on the MCU will be responsible for delivering data to and from the Android application. Data would include information about configuration of the device for different lenses, mode selection, and current distance. The connection between the Bluetooth and MCU is handled by a UART controller. This will be maintained by an interrupt and will stop all device behavior to update the MCU. We chose this method because the user would not be operating the camera while they are configuring the device. This will simplify the design for data gathering from the Android application. Below is the “MCU\Bluetooth Data Flow “.

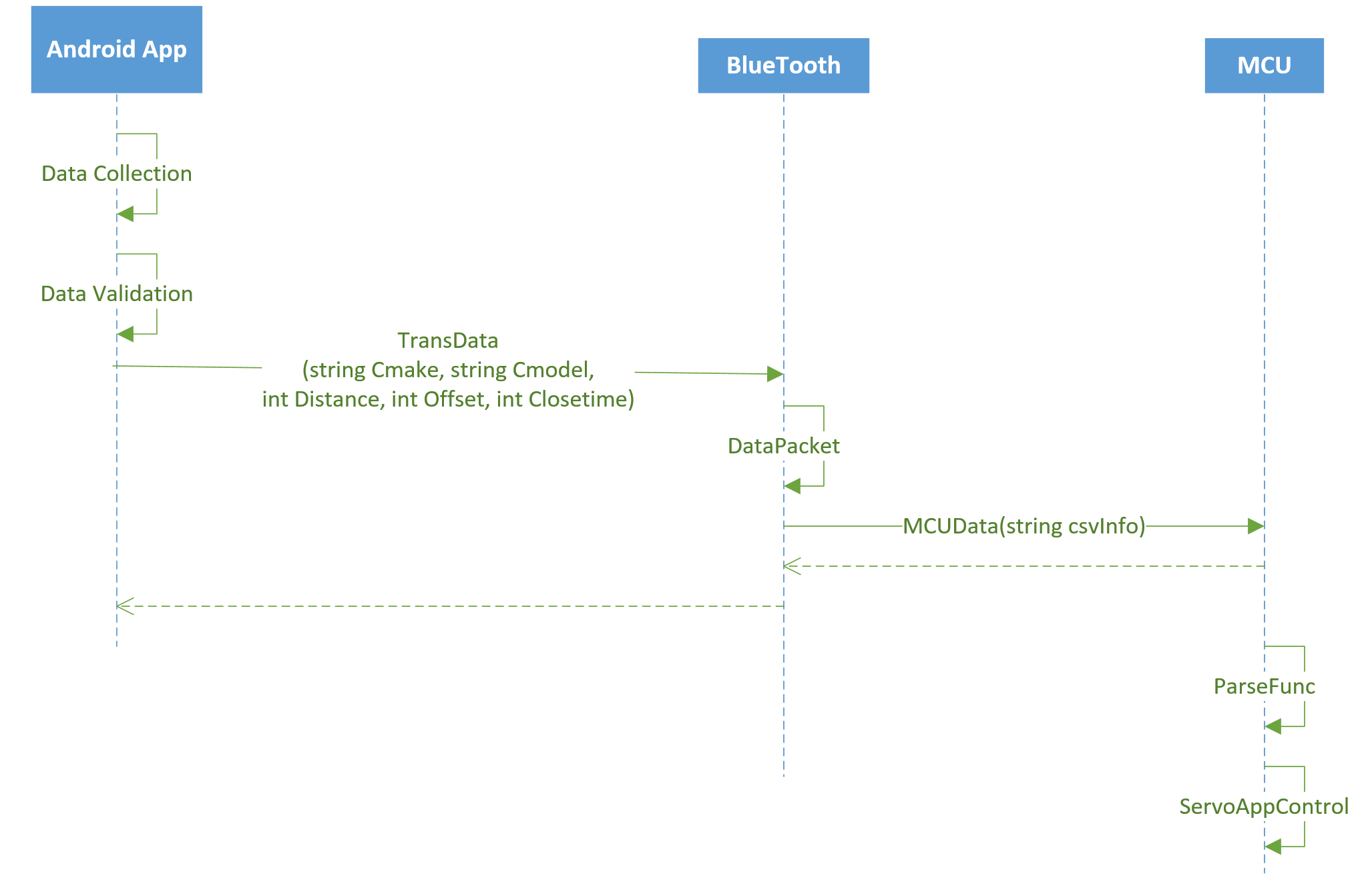


**MCU\Bluetooth Data Flow**

### Application\Bluetooth

The Android application will be the main way for the user to interface with the lens focusing system. The Android application will have different fields for the user to populate such as the following: Camera Make, Camera Model, Lens, Distance to focus to, Blend time, and whether to be in Auto Mode (where the camera focuses based on sensor data), or Manual Mode (where the data to the MCU is coming from the app). Once the fields have been populated by the user within the GUI; the user can hit the “Send Data” button located at the bottom of the GUI of the Android application. If there are any empty fields of data within the application, there will be a function that will catch that one or more of the fields has insufficient data for the MCU, and an error message will be displayed upon the GUI of the Android application stating which fields still need data.

If all fields have sufficient data, then it will pass the function that checks to make sure no data fields are blank when the send data button is clicked. Once the check function passes, the data is ready to be packaged via the Bluetooth module. The Bluetooth module will convert the data fields into a single comma separated string where each comma represents the start of a new variable/piece of data. After the data fields have been converted into a single string of data this data will be transmitted via Bluetooth to the MCU from the Android Application. The method for converting the data to bits, or seeing if we can send the data via Bluetooth in string form is still to be determined at this time.



**Android to Bluetooth UML Sequence Diagram**

# Individual Team Member Assignments

Below is a chart of part and module assignment. These are not set in stone. As the project progresses and we have more team meetings, we will all become more familiar with all the aspects of the project and encourage discussion about implementation.

|  |  |
| --- | --- |
| Name | Team Member |
| Ti TM4C123GXL | All team members have boards and will be developing and sharing code for the duration of the project. |
| Application/Bluetooth | Brandon Fletcher |
| Stepper Motor Driver | Michael Roberts |
| Distance Sensors | Andrew Kwok |
| Bluetooth/MCU | Andrew Webb |

# 

# Preliminary Cost

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Units | Part Number | Amount per unit |
| TI LaunchPad | 4 | TCM4C123GXL | $25.24 |
| Seeed TF Mini Lidar | 1 | 1597-1588-ND | $40.75 |
| HC-SR04  Ultrasonic Distance Sensor | 2 | BR010020 | $2.99 |
| Bipolar 3v Stepper Motor | 1 | 1568-1106-ND | $16.95 |
| HC-05 Bluetooth Module | 2 | CSR BC417 | $8.99 |
| Kramer BNC Male RG-g Coax Video Cable (6 ft) | 1 | KRCBMBM6 | $5.60 |
| Kamerar Universal Geared Lens Belth for FF-3 | 1 | KAKFF3B | $15.00 |
| SmallRig 15mm Black Aluminum Rod pair (8” Each) | 1 | SM1051 | $9.99 |
| Blackmagic Design Micro Converter SDI to HDMI | 1 | BLCONVCMICSH | $45.00 |
|  |  |  | Total $258.21 |