Arinc 429 Portable Receiver and Flutter Application

ENJNRs (Group 9):

Eduardo Contreras, **N**ate Trotter, **J**ared Staskal, **N**ick Morgan, **R**iley Millam

Advisors:

Daji Qiao & Mathew Wymore

Client:

Collins Aerospace - Colin Cox

Outline

Problem Statement - 3

Market Survey - 4

Constraints and Requirements - 5, 6, 7

Hardware, Software, and Bluetooth - 8, 9, 10

Users and Use Cases - 11

Interfaces - 12

Proposed Design - 13, 14, 15

Risks - 16

Testing - 17, 18

Project Plan - 19

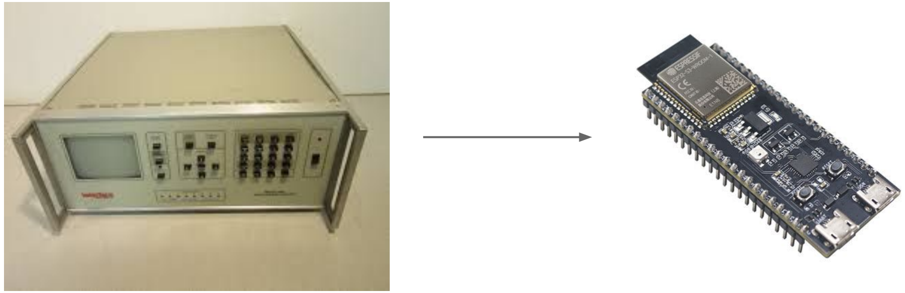
Conclusion - 20

Problem Statement

We are trying to solve the need for a bulky and expensive data receiver and transmitter. This will be done by altering a size and cost-effective bus reader to meet a variety of requirements so it can be used in the avionics industry and replace a bulky, $10,000 system with a phone and a $15 microcontroller.

- ESP32
  - Read Arinc 429 data from Holt transceiver
  - Send data over Bluetooth Low Energy

- Flutter Application
  - Read data from Bluetooth Low Energy
  - Properly Display Data

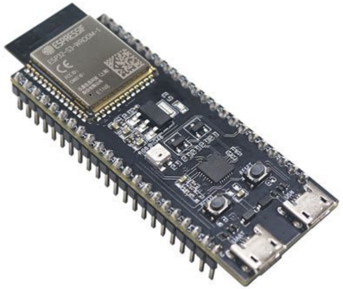


Market Survey

Why our project is unique

- Current ARINC429 Readers are expensive
  - ~$10,000
- Bulky and inconvenient
- Ours is using a small chip and a phone
  - ~$15
  - ESP32S3

TP10-A429



DAC International GDC75W
ARINC 429 Wi-Fi Bus
Reader/Analyzer

Constraints

Constraints	
1	The project must use the ESP32S3 Microprocessor
2	The mobile app needs to use the Flutter framework
3	The microcontroller should communicate with the mobile app over Bluetooth Low Energy

Flutter Requirements

Flutter Functional Requirements	
1	Read and send Arinc labels from the BLE
2	Decode the Arinc labels
3	Define and store new labels

Flutter Non-Functional Requirements	
1	Reliably connect to the chip
2	Send, receive, decode labels quickly
3	Needs to be available for android and iOS

Firmware Requirements

Firmware Functional Requirements	
1	ESP32S3 to read in data from Holt 429 receiver
2	ESP32S3 to send data over Bluetooth Low Energy
3	ESP32S3 to receive labels over Bluetooth Low Energy

Firmware Non-Functional Requirements	
1	Ability to quickly and easily update firmware
2	Send, receive, decode labels quickly
3	Receive multiple labels at a time from the Arinc429 and decide which to send

HW/SW/Technology Platforms

Hardware:

- ESP32-S3 Development Kit
- Holt 429 Receiver

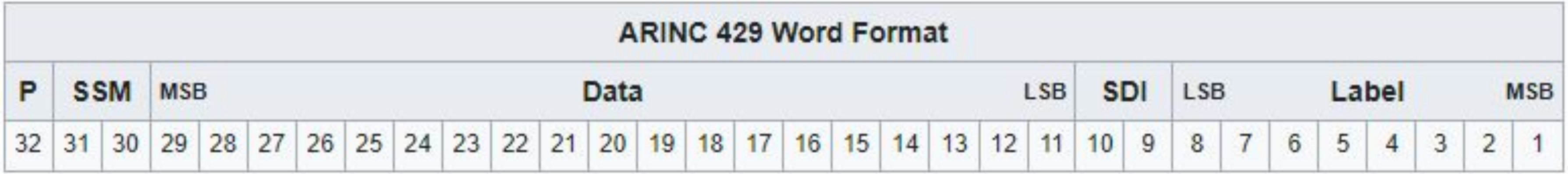
Software:

- ESP-IDF Development Framework
  - Program the firmware on ESP32-S3 chip
- Flutter Framework
- BLE API

Hardware Resources	
Part	**Quantity**
ESP32 Devkit	5
ESP32 WROOM	5
Arinc429 Evaluation Board	1
HI-3593	1
Software Resources	
Visual Studio Code	
ESP-IDF (C, Micropython)	
Flutter Framework	
Dart	

Arinc 429 Serial Bus

Arinc 429 is a serial bus commonly used in the avionics industry.



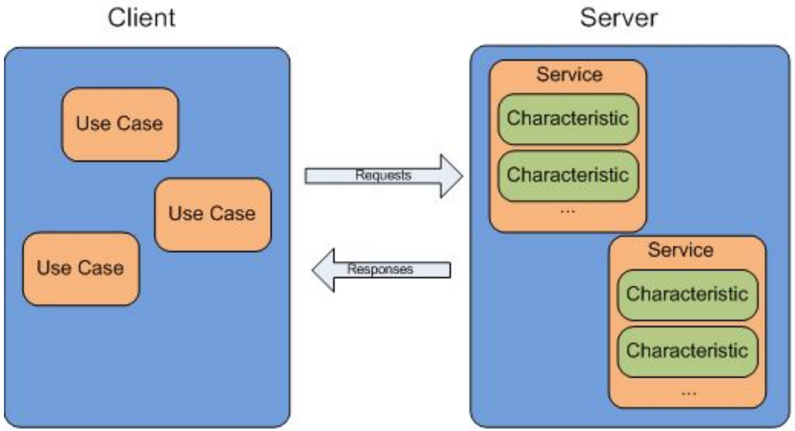
ARINC 429 Word Format																															
P	SSM		MSB									Data								LSB	SDI		LSB			Label				MSB
32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

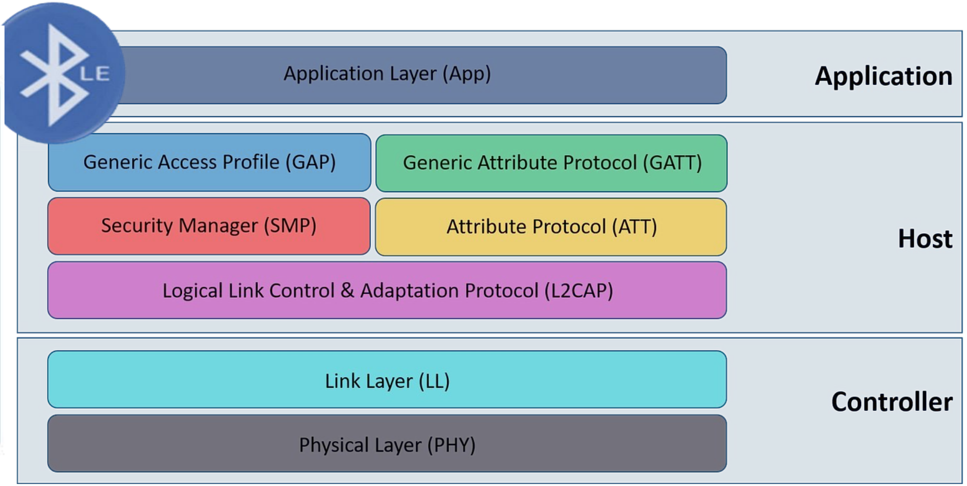
- **P bit:** Parity bit
- **SSM bits:** Sign/Status Matrix
- **SDI bits:** Source/Destination Identifier
    - Location of device sending data
    - Can have different meanings depending on label type
- **Label bits:** Describes how to interpret the other data in the word such as the actual data bits, and the SDI
    - 8 bits grouped into a 2 bit octal digit and two 3 bit octal digits.

Bluetooth

We’ll use an ESP32 MCU to connect to our smartphone through Bluetooth Low Energy (BLE) to transmit data to and from our Flutter Application.

- Our ESP32 and Flutter App will act as GATT server and GATT client respectively

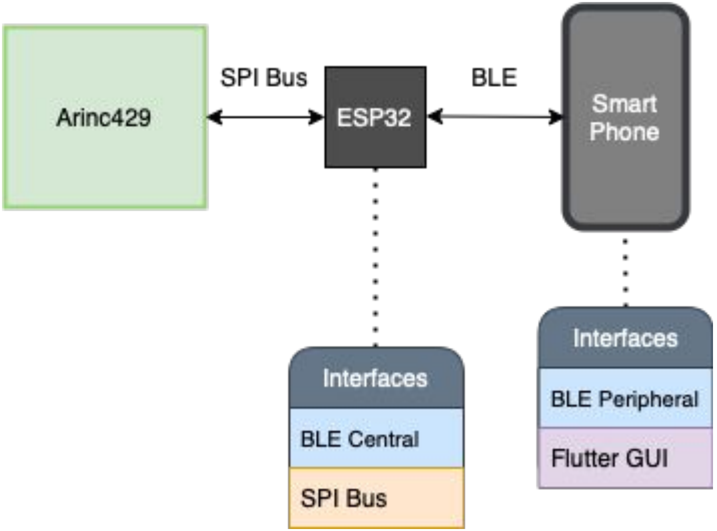




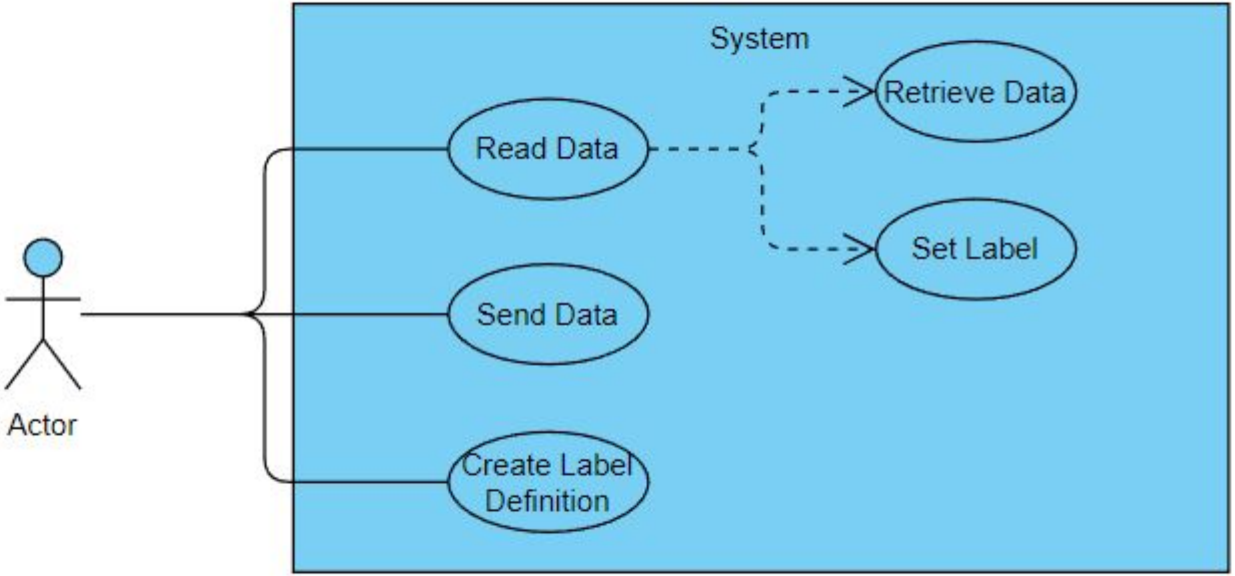
Interfaces

- ESP32
  - BLE Central
    - Send/Receive data to/from smartphone
  - SPI Bus
    - Send/Receive Arinc429 data

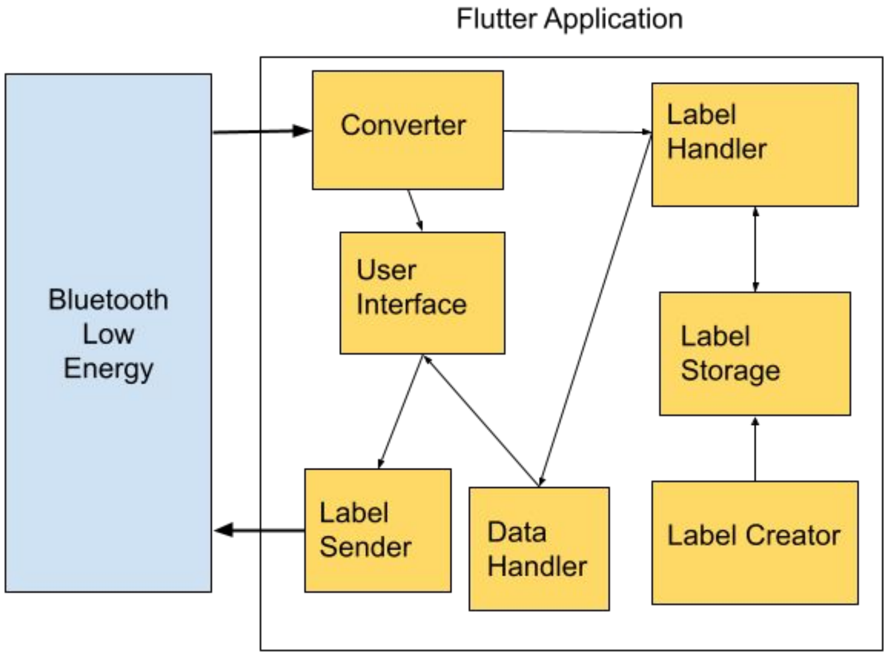
- Smartphone
  - BLE Peripheral
    - Send/Receive data to/from ESP32
  - Flutter Application
    - Flutter GUI
      - Interface with User



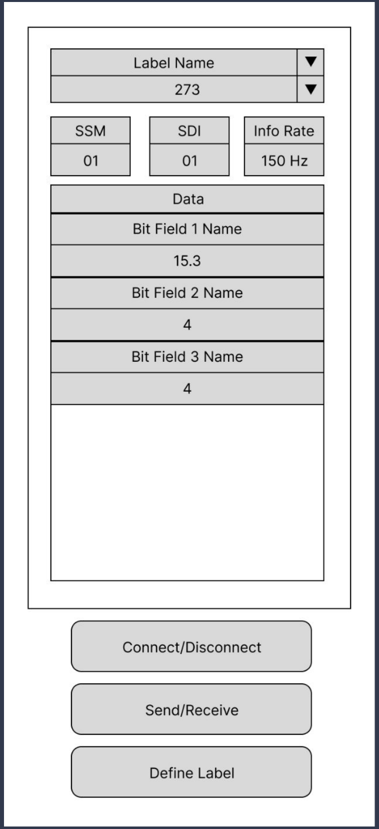
Users and Use Cases



Functional Decomposition



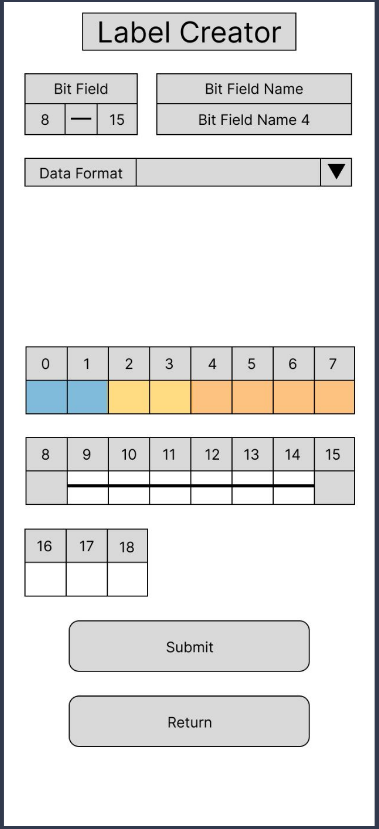
Proposed Design – Mock Up



Label Name	▼
273	▼

SSM	SDI	Info Rate
01	01	150 Hz

Data
Bit Field 1 Name
15.3
Bit Field 2 Name
4
Bit Field 3 Name
4



Label Creator

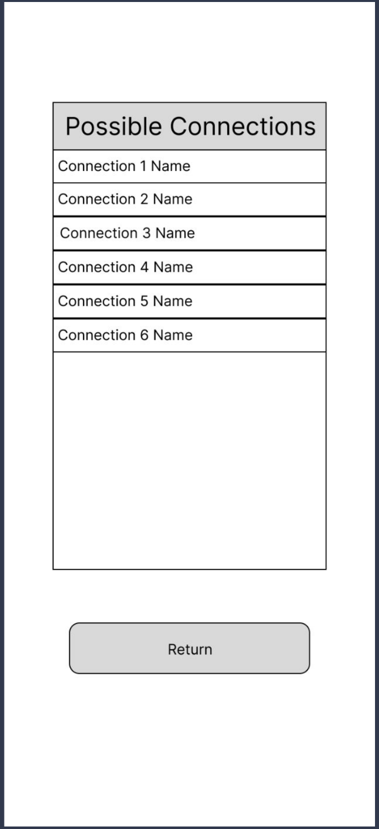
Bit Field			Bit Field Name
8	—	15	Bit Field Name 4

Data Format ▼

0	1	2	3	4	5	6	7
🟦	🟦	🟨	🟨	🟧	🟧	🟧	🟧

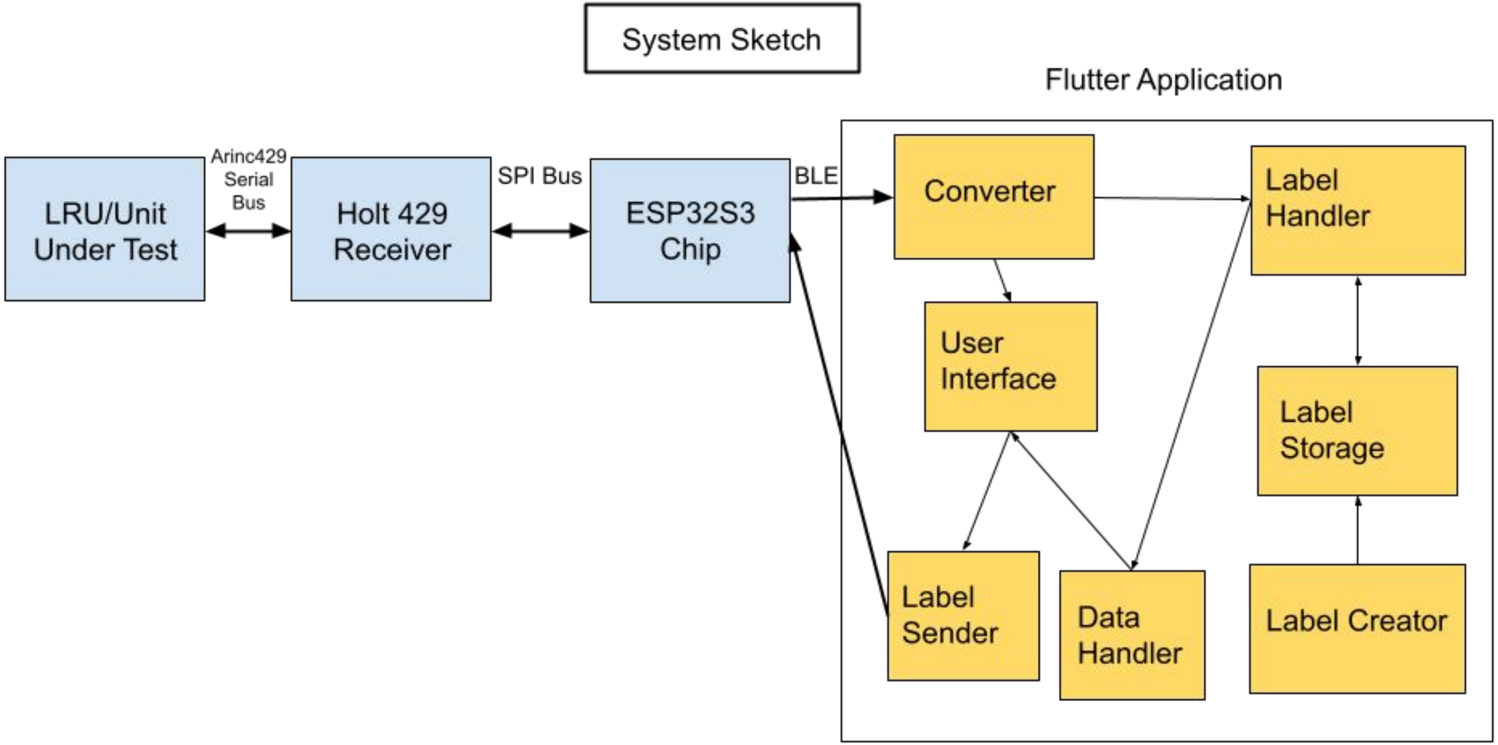
8	9	10	11	12	13	14	15

16	17	18



Possible Connections
Connection 1 Name
Connection 2 Name
Connection 3 Name
Connection 4 Name
Connection 5 Name
Connection 6 Name

Proposed Design - Design Visual and Description

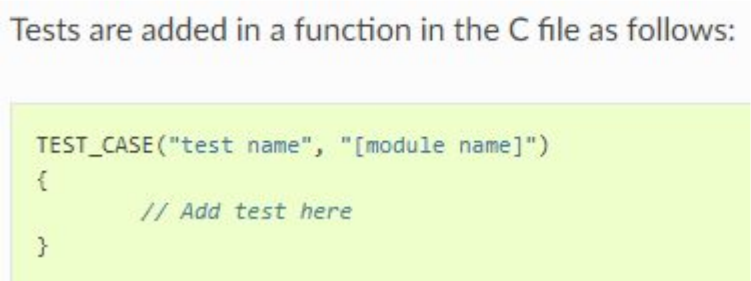


Potential Risks

Task	Sub Task	Risk Factor	Probability(0-1)	Cost(1-5)
Firmware	Read Data	Loss of data during transmission	0.01	4
	Transmit Data	Loss of data during transmission	0.01	4
Flutter Application	Converter	Reading data over BLE fails	0.1	5
	Label Handler	Binary Search Takes Longer than Expected	0.2	2
	Label Storage	Creating files works differently on different operating systems	0.2	2
	Label Creator	User created custom labels are inaccurate to the Arinc429 word format	0.2	3
	Data Handler	Casting errors when handling different data types	0.1	1
	User Interface	On-screen display is significantly out of place on different mobile platforms	0.1	1

Test Plan (ESP32S3)

- Unity Testing Framework
- Test cases will include
  - Data coming from the Arinc429 Simulator
  - Commonly used 32 bit words
  - Edge cases (under or over 32 bits)
- If app successfully gets word, then success

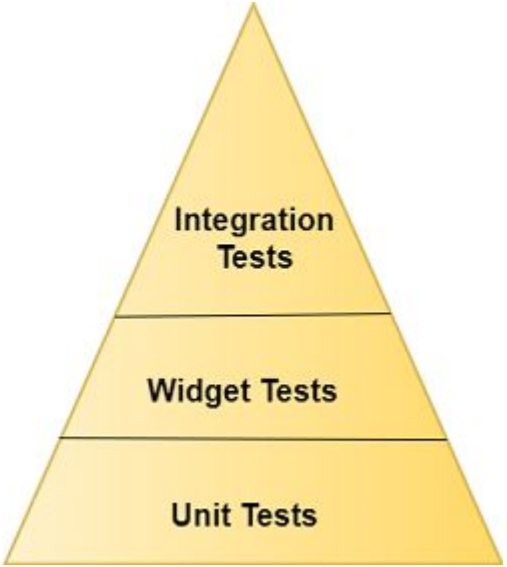


Tests are added in a function in the C file as follows:

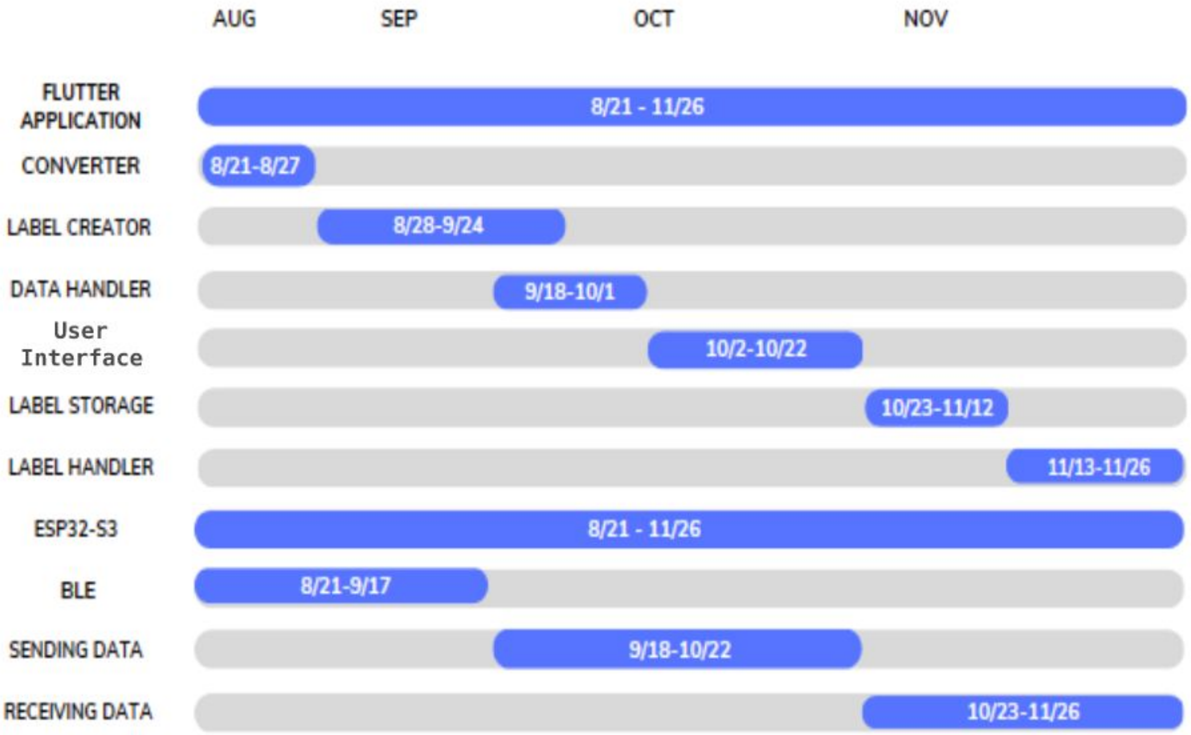
```
TEST_CASE("test name", "[module name]")
{
        // Add test here
}
```

Test Plan (Flutter)

- Flutter Testing Framework
- Unit Tests
  - Create Test cases for each component
  - ex.) Label Handler, Label Sender
- Widget Tests
  - Testing the buttons and text boxes
  - ex.) pressing “Make Label” button takes user to Make Label screen
- Integration testing
  - Create Test cases that makes labels, then send words to chip
  - If chip receives word, then success



Project Plan



FLUTTER APPLICATION	8/21 - 11/26
CONVERTER	8/21-8/27
LABEL CREATOR	8/28-9/24
DATA HANDLER	9/18-10/1
User Interface	10/2-10/22
LABEL STORAGE	10/23-11/12
LABEL HANDLER	11/13-11/26
ESP32-S3	8/21 - 11/26
BLE	8/21-9/17
SENDING DATA	9/18-10/22
RECEIVING DATA	10/23-11/26

Conclusion

Prototyping

- Hardware
  - Successful connection to device from ESP32 over BLE
- Software
  - FIGMA
  - API Document

Individual Contributions	
Riley	Research, BLE Firmware, Communications
Nate	Research, BLE Firmware
Eduardo	BLE Research, Firmware
Jared	Research, App Development, BLE API
Nick	Research, App Development, BLE API

Next Semester:

We plan to execute our aforementioned project schedule and focus entirely on the design.