Red Owl Book Organization Tool (ROBOT)

Joshua Christie, Jostyde Lekadou, Montserrat Santillan-Rodriguez, and Amanda Walenciak

Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — This paper presents the Red Owl Book Organization Tool (ROBOT), a novel approach to book inventory management designed to make the process of book organization and location more efficient. To this end, an approach using computer vision for optical character recognition is described. A strip of LED lights controlled by a microprocessor is used in conjunction with computer vision to highlight detected book locations. Using this technology, users will be able to take real-time inventory of the books available using an Android app. This project demonstrates the ability of computer vision to enhance the book retrieval process.**

***Index Terms* — Computer vision, inventory management systems, optical character recognition, embedded systems, user interface design, real-time database management.**

I. Introduction

The need to transform book inventory management due to the declining appeal of physical libraries and bookstores is what motivates the Red Owl Book Organization Tool (ROBOT) project. Our project seeks to provide a smart bookshelf that is integrated with Computer Vision (CV) technology, in an effort to alleviate the aggravation caused by inaccurate inventory management. With the use of an Android application and database access, this invention will allow for the real-time tracking of book availability and locations within libraries and bookstores, improving the user experience.

One of the main goals is to build a bookshelf with wide-angle cameras installed on the shelves that are linked to a programmable database to enable real-time book movement detection. CV is preferred over traditional barcode scanners due to its efficiency and dependability, as well as its capacity to scan over longer distances, detect several items at once, and update in real-time. In addition to having advanced scanning capabilities, the project seeks to improve user experience by including several unique elements. Specifically, by turning on matching LEDs on the actual bookshelf, the mobile application's LED search feature will make it simple for users to find the books they want. This user-friendly interface aims to improve customer convenience and satisfaction by simplifying the book browsing experience. The use of cameras, CV, database, and the Jetson Nano to analyze camera pictures and scan books quickly, are all part of the implementation process.

II. Engineering requirements

For our project, we initially set 6 main goals to achieve. However, after beginning the implementation of our project, we realized only 3 of these main goals would be reasonable. These are shown below in Table 1. From the table we can see that book detection accuracy and effectively helping to quickly locate books are some of the most critical expectations for the project. Both things were limited by the Jetson Nano memory and capabilities.

Table 1 - Engineering Requirements

Engineering Requirement	**Specification**
Accuracy of book titles detection	90 percent or more
Accuracy of book locations detection	90 percent or more
Time for all systems to update including database and application	40 seconds or less

III. System Concept

The basic structure of our project is made up of several complex software elements, each necessary to its general functionality. Every member of our team has been assigned particular tasks inside the software architecture. Four essential elements have been identified: computer vision, embedded firmware, database, and application.

Taking a closer look at the details of our setup, Figure 1 demonstrates how the cameras, computer, and database will interact with softwares. The program that runs on our Jetson Nano is the central component that manages dataset updates and picture processing. As the center of the system, the CPU analyzes incoming data streams continuously. When a missing book is detected,

the computer quickly sends orders to the PCB, which sets off an alert on the app, and as a stretch goal, a series of alarms.

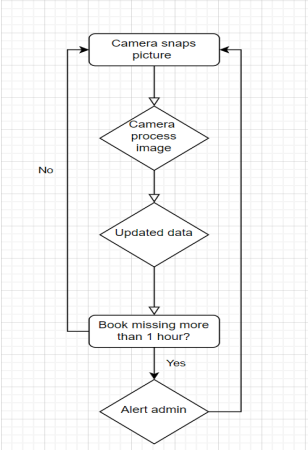

Figure 1: Software Block Diagram

Furthermore, the PCB serves as an adaptable bridge, translating orders from the computer and the user application, allowing users to quickly enable book searches and turn on LEDs when needed. Figure 2 aims to show how the overall system will work alongside sub systems and user interaction.

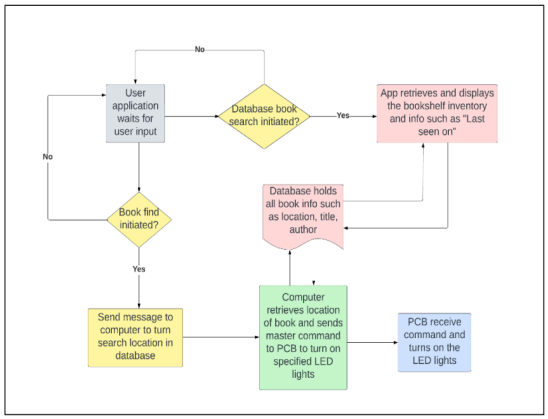

Figure 2: General Procedure Block Diagram

Two of our team members are allocated to the embedded firmware implementation. They are also responsible for managing the communication channels that connect the Jetson Nano, the custom PCB, and the LED lights.

In the meantime, two other team members are working on creating the database and application components. They have two goals in mind: creating a database that will satisfy the requirements of the Jetson Nano while providing different users with an enjoyable user experience. Their scope includes a wide range of functions, from making sure the computer processing photos is accessible to creating user-friendly interfaces for mobile and online apps.

Lastly, the whole team has been working on putting computer vision technology into practice. Equipped with the Jetson Nano and Raspberry Pi cameras, we patiently adjust algorithms to handle photos taken by our cameras. Our objective is very clear: we want to easily identify and extract text from photographs so that our system may have greater accuracy.

Our system's design relies heavily on the custom PCB, which makes communication between a wide range of devices easy. Its function is important because it can decipher orders from the Jetson Nano and transfer power to necessary parts including the Bluetooth modules and controllers for LED lights.

All these skills are aimed at creating a reliable and adaptable system that combines hardware and software to provide great functionality and user satisfaction. Each team member brings a unique set of skills and gains knowledge from the others, ensuring that each aspect of our project is efficiently carried out.

IV. App and database

The application and database work together to allow users to check out and search for books. The application is able to retrieve information from the database such as book title, authors, and last seen on shelve dates and times. The database is also readable and writable from the Jetson Nano running a Python script.

The database we are using is a Cloud Firestore database provided by Google. The service is free and Google provides a vast amount of documentation and examples for using it alongside Kotlin, Java, and Python, which are the main programming languages we are using. Changes made to the database through external sources are quickly updated, almost automatically. This is a useful feature since our goal is to implement real-time book tracking.

The application has been built using Kotlin and some Java on Android Studio. The database can be used by the application to initiate a book search, check out and check back in a book. Searching for a book can be done by creating a user entered variable ‘search’. When the book Title is given by the user, it will be added to a query command that will search the titles in the database for a match. Because of this, capitalization is important as a given title may be available but will not

show up due to grammar mistakes. However, we have implemented a capitalization-safe technique that converts all text to lowercase. If a book is found, it will notify the user and will illuminate the corresponding LEDs with the approximate location as well as the shelf location. For this feature to function, the Python script on the Nano must also have the database information and connect successfully. Then, it can also read and write data to the database just as the application. Furthermore, the application is able to add and remove books to the database as well. This is a feature that only an 'administrator' can use. To identify between different types of users, each user has a 'isAdmin' field that can only be changed within the Google Cloud Firestore online platform. Then, no one besides the owner of the database can change the type of account for a user.

V. Computer Vision

To implement the computer vision portion of the project, several approaches were considered. Although MATLAB and Python were both taken into consideration as possible programming languages for ROBOT, Python was selected due to the libraries and resources available for it. Additionally, Python can be used directly on the terminal of the Jetson Nano, making for easier system integration.

A. Optical Character Recognition

Initially, the team attempted to implement several different methods of optical character recognition. During early phases of the project, Tesseract was used as the optical character recognition (OCR) engine. This approach involved using grayscale conversion, a gaussian blur, and image rotation to preprocess the image before using k-means clustering to separate the individual book spines using bounding boxes. However, due to continuously yielding subpar results, the team eventually implemented easyocr instead.

Because easyocr proved to be more accurate and robust in real-time environments, the preprocessing steps were altered to be less rigorous. Simple image rotation and resizing was implemented to clarify the text for recognition, and the OCR was applied to the entire image instead of solely on detected bounding boxes. The program's effectiveness was further improved by the addition of fuzzy matching, a technique that allows matching to occur despite minor discrepancies.

B. Coordinate Location Detection

After the program captures the image and applies OCR, it then determines the location of each book. To do this, bounding boxes are drawn around each detected title. The code takes only the lowermost coordinates of the horizontal portion of the bounding boxes into consideration for further processing, as the rest of the coordinates are extraneous. Using these coordinates, the position of each book is extracted and sorted by order of appearance.

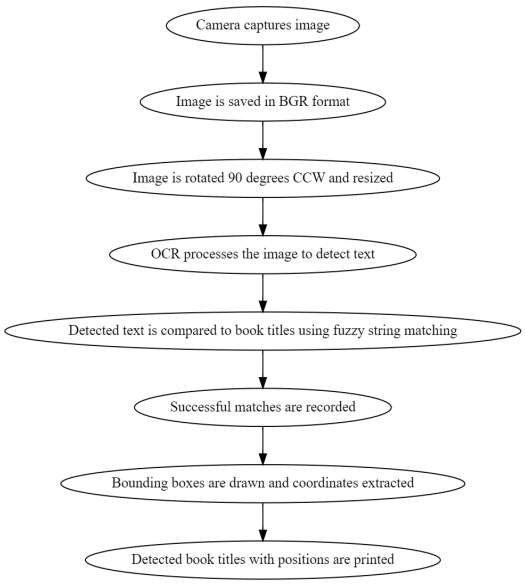

Fig. 3: Computer Vision Flowchart

C. Setting up the Jetson Nano

After completing our book titles and locations detection section of the code finished, we then transferred the code to the Nano. The Nano, by default, runs Ubuntu 18.04, which has Python 3.6.9 as its default Python 3. We kept this the same but found that it was not as simple to download dependencies and libraries as it was on our own desktops. To begin, the OpenCV version installed by default is version 4.4.0 that does not support CUDA, the GPU of our system. Furthermore, it also does not support GStreamer pipeline, which we were counting on to access and control our cameras. Then, we had to install OpenCV from source with CUDA and GStreamer support which took about 9 hours to complete. In addition, EasyOCR was also not as simple to download. For one, when you do a single command such as 'pip3 install easyocr,' you get the latest version that requires a Numpy version that Python 3.6.9 does not support. Then we had to uninstall Numpy

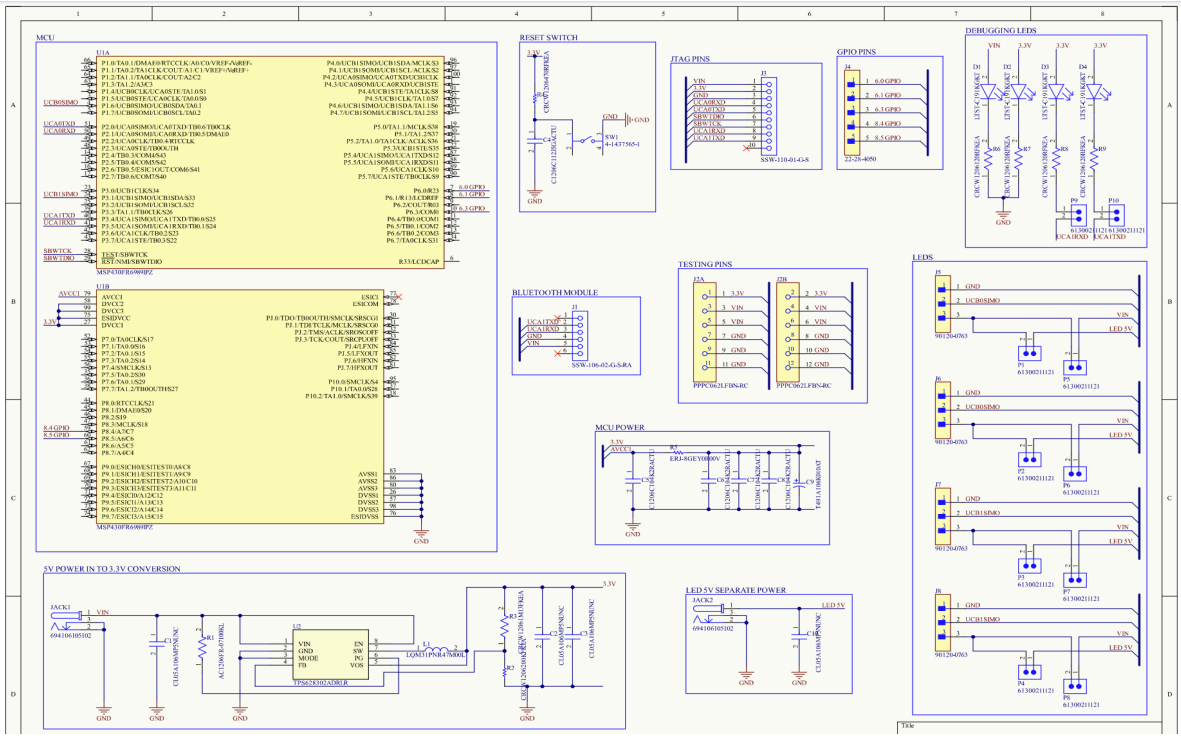

Figure 4: Overall PCB Schematic

and easyOCR, then reinstall older versions from source. The last issue we had was that easyOCR also requires PyTorch, and the pre-installed version of PyTorch does not support CUDA 10.2, which is the GPU that we have. Then, we had to find an older version of PyTorch that had CUDA 10.2 support and install it from source.

In summary, figuring out all the necessary and compatible libraries for our code to function on the Jetson Nano took about 5 days, which is much more time than we expected it to take. Factors to take into account is that the group has minimal exposure and experience to Linux and Ubuntu operating systems. It was a huge learning curve.

D. Testing

While developing the computer vision code, several rounds of testing were necessary. Initially, the program was developed in Google Collab. After the

aforementioned changes were made to accommodate the Jetson Nano, the code was run through a series of tests to ensure proper functionality.

To test for accuracy, several books at random were chosen to have their titles obscured. After obscuring the titles, the camera captured the image, and the OCR was run. Because the books with the obscured titles were not detected in any of the tests, it was clear that the code did not detect false positives.

Next, the code was tested for efficiency. Initially, the code took over a minute to run on the Jetson Nano. While technically functional, this was not ideal as per the engineering design specifications. After decreasing the resolution of the images captured, it was found that the code was able to complete the OCR in approximately ten seconds, providing a significant improvement.

VI. PCB Design & Schematic

The PCB design of ROBOT was designed, routed and modeled in Altium and most of the footprints for the components were found on ultralibrarian. For the footprints that were found there, SnapEDA was used. The full PCB schematic contains 11 smaller segments shown above.

A. DC Power Jacks

The first version of the ROBOT PCB included one DC power jack. This jack is used to power the MSP4306989, bluetooth module, status leds, and LED strip lights for the book find feature. However, when testing this design presented problems. The problem

being the LED strips require a large amount of current to light up large segments of the strip, this causes the bluetooth module to receive less power and malfunction. To resolve this issue, another DC jack has been added to the design. This extra DC input will be used to power the LED strips only while the other will be used to power the other components on the PCB. The LED strips will now be allowed to pull as much current as needed while not impeding on the function of the other parts of the design. Paired with this second power jack is a capacitor, which helps smooth out the DC input coming from the external power supply.

B. Voltage Regulation

The power supplies that will be used for the ROBOT pcb are 5V-1A, and 5V-4A for each respective DC jack. This voltage is enough to power the activity LEDs, LED lighting strips, as well as the bluetooth module. However, the microcontroller on the PCB requires a voltage of 3.3 and a current of no more than 200mA, meaning a voltage regulator is needed to step-down the 5v input to 3.3v. In the ROBOT design this is accomplished with a switching voltage regulator from Texas Instruments (TPS628302ADRLR). This regulator was found on WEBENCH and is based on the specifications entered into an online configurator. This project. The device features 1.8-V to 3.6-V operation [1].

This microcontroller has several different modes meant for saving power, such as standby, hibernate, and real-time clock modes. This MSP430 is capable of reaching as low as 100uA/MHz in active mode, 0.4 uA in standby mode, and uses an extremely low amount of power in hibernate mode [1]. Because of the low power consumption, it can even be operated on battery power.

The MSP430FR6989 uses Ferroelectric Random Access Memory (FRAM). FRAM is a non-volatile memory technology that combines the best aspects of both Flash and SRAM. FRAM provides fast read and write speeds and lower power consumption. This will make programming simpler and is known to be especially useful in projects involving data logging, configuration storage, and frequent data updates. Debugging and communication with the PC is made easy with the eZ-FET, which provides a "backchannel" UART-over-USB connection with the host [1].

The MSP430FR6989 Launchpad Development kit can be used with this microcontroller. This kit adds significant functionality to the MSP. The kit has an LCD display, meaning users will be able to see information displayed on it instantly. The MSP430FR6989 comes with a wide range of peripherals, including GPIO pins switching regulator has 8 pins corresponding to: Vin, GND, Mode, FB, VOS, PG, SW, and EN. Vin and EN are both connected to the DC jack in addition to a capacitor connected to ground to filter the input. Without the enable pin being pulled high the regulator will not function. While the Mode and GND pins are connected together putting the regulator into PWM mode based on the incoming load. The FB pin is a feedback pin that is connected to a voltage divider, and the PG pin is an open drain output that is connected to the Vin and EN pin in series with a resistor. Finally, the VOS pin and the SW pin are connected to an inductor in series, in addition to resistors and capacitors in series to the 3.3V line. This 3.3V line is then connected to a power LED as well as the MSP4306989. This 3.3V line is also present on the JTAG header as well as extra headers for testing other peripherals.

C. MSP4306989 Microcontroller

The MSP430FR6989 is a Texas Instruments MSP430 microcontroller. It is known for its power efficiency, meaning it will be a good choice for projects with power constraints. This microcontroller has a clock speed of 16 MHz and has 2 KB RAM, which is a decent amount of computing power that should be sufficient for our

and five timers. These peripherals mean that it can be used for many applications, such as sensor interfacing, communication with other devices, and precise timing control. The MSP430FR6989 is compatible with all three major communication protocols: I2C, UART, and SPI. This means a wide variety of peripherals and devices are able to be used with this MSP430 easily. Additionally, it offers high-resolution analog-to-digital converters (ADCs) for applications that require accurate analog data acquisition [1]. This MSP430 is also compatible with a large number of BoosterPacks that can be developed ourselves or purchased through the Texas Instrument website.

Another benefit is that the group already has access to an MSP430FR6989 due to prior class requirements. This prevents us from having to get a new microcontroller, saving money. Furthermore, the group has access to a comprehensive book on MSP430 microcontroller basics from an earlier class. Because of our past experiences, multiple group members are comfortable with using and programming this device. We are also comfortable using Code Composer Studio, which is compatible with this microcontroller.

One downside is that the microcontroller uses relatively more power in active mode compared to alternatives. Additionally, its limited community

support, when compared to certain other microcontrollers, can pose challenges in finding resources and troubleshooting.

D. Bluetooth Module

The Bluetooth module will be implemented using the HC-05 sensor. It will be wired up to the MSP430FR6989 and other traces on the printed circuit board as shown in Figure 7 and Table 9. The sensor and MSP430 must have a receiving and transmitting line connected to one another in opposite form. The standard baud rate they will both be set at will be 9600 baud rate which should be sufficient for the requirements of our goal of sending simple commands and acknowledgements. A problem to consider once we receive the initial prototype is to ensure that the Bluetooth module is not being interfered with by other signals on the board. If so, it is important to consider changing the board layout and possibly spacing out the components more.

E. LED Strip Headers

The individually addressable LEDs, WS2812B, will connect to the PCB through the ground and UCB0SIMO signals. This will allow the MSP430 to control which lights on the strips turn on. Three of these strips will be connected to an external power source of 5V 15A for testing. For the final PCB, we aim to have the same 5V source power the LEDs and PCB by adding a voltage regulator circuit that can regulate 5V to 3.3V and can handle 20A of current. As previously discussed briefly, it will be a more difficult voltage conversion circuit than the one we currently have as the current demands for the LEDs is much higher than typical linear voltage regulators. Nonetheless, through tools such as the TI Power Bench Designer, the team hopes to accomplish this design by the beginning of Senior Design II. It is possible that the new conversion circuit will also be able to power the Jetson Nano and cameras.

F. Jumper Connections

As stated in a previous section, the ROBOT pcb uses two DC jacks for powering all the components present on the design. With these dual jacks comes the issue of using an external power supply that exceeds the voltage and current ratings of the devices on the pcb and may cause damage. To ensure we double check the power supply being used a jumper has been added to the output line of the voltage regulator, which we can then connect after verifying the rating of the power supply. In addition, there are multiple jumpers present for the LED strip headers present. This is to enable the use of either the same power as the microcontroller and bluetooth module, or an entire separate jack solely for the LEDs. It was done this way for testing as when configuring code on CCS (Code Composer Studio) only a minimal amount of LEDs will be lit and less power is required. However in the final demo the second DC jack may be used due to its higher current output, so the jumper cover will be moved from the Vin to 5V external.

G. JTAG Header

The MSP430 launchpads used throughout our undergraduate courses have a debugging and flashing module attached that can be used on external MSP430 MCUs after proper disconnections and connections have been made. The module is called the ez-FET emulator. To complete these connections, we used the official MSP430FR6989 launchpad datasheet that has a specific section that gives instructions on how to use the launchpad debug attachment on external devices. This greatly simplified the complexity and cost of our custom PCB since we could eliminate our bootloader and flashing circuit module. The module now consists of a female pin header that we will plug our PCB MCU pins to the debugger module on the launchpad as shown below in the table.

It is important to note that from reading the datasheet, Texas Instruments only ensures this is a valid procedure for any microcontroller in their MSP430 series that has the pins listed in the table below with the same functionalities. The microcontroller we are using fits the criteria. Additionally, the RXD and TXD pins were optional, but the team decided it is a good idea to add them in case we want to later implement wired serial communication for any reason. It can also help to test faulty Bluetooth implementation and/or sensor.

VII. Bookshelf Design.

A. Bookshelf

The shelf being used in the ROBOT project is the 3-Cube Storage Organizer purchased from WalMart. Each of the 3 sections of the shelf have a height of 13 ", a depth of 15” and a width of 13”. The depth measurement is crucial for the functionality of the camera system, as if the depth is beyond 15” or below 10” the camera will not be able to see the entire spine of each book, hindering accuracy. The top and bottom shelves will be used to hold the books while the center section will be used for storing the electronic components. In addition the color that was chosen specifically for accuracy, as the white of the shelf will

reflect light increasing the effectiveness of the camera lighting.

B. Raspberry Pi Camera & Mount

The Raspberry Pi Camera will be used to capture an image of the books placed on the shelf. It requires a ribbon cable for both power and sending information. This ribbon cable will be run through the back of the shelf and connect to the Jetson Nano. The camera angle is also very important for the code to function. To keep the angle consistent, a camera mount has been added to the front of the shelf. This mount is made of a metal dowel and a mounting plate with a predetermined angle which the camera is glued to.

C. Lighting for Camera Accuracy

When testing with the Raspberry Pi Camera, problems began to arise with the clarity of the output image. Steps were taken to fix this issue such as increasing the exposure and adjusting the white balance. Tweaking these factors made a difference but the images taken did not meet the Engineering Specifications for accuracy. The solution for the accuracy issue was introducing an LED light above the books which made a significant difference in clarity and book identification accuracy. At first, this light was a simple LED powered by AAA batteries, but now has been updated to a rechargeable low profile LED bar that can be quickly removed and replaced.

D. Jetson Nano

The Jetson Nano is a small computer designed by NVIDIA for AI and CV applications. It will be used for running our book title detection code, which uses CV. It will be powered with an external DC power supply and be mounted in the middle of the shelf with the rest of the electronics.

E. Raspberry Pi Touchscreen

The HAMTYSAN Touchscreen will be used for this project as both a display and interface. It will be connected to the Jetson Nano for both display and power. After considering possible locations for the screen our team decided that mounting it on the right side of the bookshelf would be the most optimal location for library patrons to access.

F. PCB Mount

The PCB for our LED book find will also need a place to reside on the shelf. This mount will be located on the middle section of the shelf, and will be created with standoff posts located on each corner of the PCB. To maximize the length of the connected LED strips, the PCB will be mounted on the left side and will be rotated 90°.

VIII. Communication

Implementation of our project requires several types of communication between each subsystem. Our custom PCB supports both UART and Bluetooth serial communication. For our system to work, the Jetson Nano and the phone with the application must be connected to the WiF. Additionally, our LEDs require SPI serial communication.

A. UART Serial Communication

Universal Asynchronous Receiver-Transmitter (UART) is a popular serial communication protocol used to exchange data between microcontrollers, computers, and embedded systems. It allows for the transfer of serial data between two devices through two lines, Rx and Tx. UART is asynchronous, meaning that data is transmitted without a shared clock signal between the devices. Instead, both devices must agree on a specific baud rate to ensure proper data synchronization [2]. Although we are not using this communication method at the moment, our PCB is capable of implementing this with another device. It was useful in our debugging phase when our Bluetooth would at times not communicate properly. We compared our Bluetooth results with our UART results to see what could be improved or diagnose a problem. For example, when we connected our computer to UART, we would receive the data fine, but when we tried it with Bluetooth letters and characters would be jumbled up. We were soon able to confirm that the HC-05 Bluetooth module we are using was configured for a different baud rate than we wanted.

B. Bluetooth Communication

By integrating a Bluetooth module directly into our custom PCB, we can establish communication between the Jetson Nano and the microcontroller driving the LED lights. Bluetooth's Serial Port Profile (SPP) mode facilitates straightforward serial communication that will allow the Jetson Nano to send commands to control the LED lights. SPP replaces UART serial communication in our system and “is great for sending bursts of data between two devices” [3]. To implement Bluetooth, we used the HC-05 Bluetooth module that is

compatible with the MSP430 family of microcontrollers. The module's Rx and Tx lines have a 3.3V level that can easily be detected by the MCU without any shifting necessary.

C. Wireless Communication

In order for communication and updates between each of our subsystems, the phone with the application and the Jetson Nano running our Python script, must be connected to the Internet. We are connecting both of these devices to the Internet through WiFi. Our Jetson Nano has wireless and Bluetooth connection capabilities and we successfully have connected both. The phone we are testing our application on is a Samsung Galaxy 8 that is capable of also connecting to WiFi. Our PCB did not have a need for WiFi.

D. SPI Serial Communication

Serial Peripheral Interface (SPI) is another versatile and very popular communication protocol. SPI is different from UART; while UART is asynchronous and is known for being simple and easy to use, SPI operates in a synchronous fashion and often requires a more complex wiring configuration. SPI communication typically involves multiple wires or lines, including the SCK for clock synchronization, MOSI (Master Out Slave In) for data transmission from the master to the slave device, MISO (Master In Slave Out) for data transmission from the slave to the master device, and one or more select (SS/CS) lines to enable communication with multiple slave devices [51]. We will be using SPI so that our LEDs can be controlled by our MSP430.

IX. Conclusion

In conclusion, ROBOT can be used to make the process of book inventory management more efficient. By using technologies such as computer vision and a user-friendly app, books can be quickly located. The team successfully achieved a balance between functionality and design constraints.

Acknowledgement

The group would like to extend its gratitude to Dr. Sonali Das, Dr. Zakhia Abichar, and Dr. Ronald DeMara for assisting in reviewing our project. In addition we thank Dr. Lei Wei for his experienced advice and criticism throughout the completion of our project.

References

[1]"MSP430FR6989." Texas Instruments ,https://shorturl.at/biCFL.

[2] Electronics | 69, Circuit Basics | DIY. "Basics of UART Communication." Circuit Basics, 13 Feb. 2016, www.circuitbasics.com/basics-uart-communication/.

[3]"Bluetooth Basics - SparkFun Learn," learn.sparkfun.com. https://learn.sparkfun.com/tutorials/bluetooth-basics/bluetooth-profiles

Biography

Amanda Walenciak is a 24-year–old electrical engineering student at UCF. She is pursuing a PhD at Purdue University after graduation. Her interests include electromagnetism and neuroscience.

Montserrat Santillan-Rodriguez is a 23-year-old electrical engineering student at the University of Central Florida. Her interests include embedded systems and digital electronics design. After graduation, she will be working at Ford as an electrical engineer.

Jostyde Lekadou, a 23-year-old student at UCF, is pursuing a degree in both electrical engineering and a minor in mathematics. Her academic interests span the fields of linear control systems and AutoCAD design.

Joshua Christie is a 22-year-old electrical engineering student at UCF. He will be working for FPL after graduation as an Associate Engineer. His interests include Microsoldering and building Computer systems.