GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

Bill Tech Dollar Identifier

Autor: Javier Martínez Liñera

Director: Arne Fliflet

Madrid

- Julio 2022 -

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Bill Tech Dollar Identifier

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2021/22 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

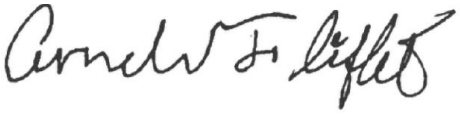
tomada de otros documentos está debidamente referenciada.

Fdo.: Javier Martínez Liñera Fecha: 07 / 07 / 2022

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Arne Fliflet Fecha: 15 / 07 / 2022



GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

Bill Tech Dollar Identifier

Autor: Javier Martínez Liñera

Director: Arne Fliflet

Madrid

- Julio 2022 -

BILL TECH DOLLAR IDENTIFIER

Autor: Martínez Liñera, Javier.
Director: Fliflet, Arne.
Entidad Colaboradora: University of Illinois Urbana-Champaign

RESUMEN DEL PROYECTO

Bill Tech Dollar Identifier es un dispositivo que identifica el valor de un billete dólar de Estados Unidos mediante clasificación de imágenes. En concreto, el dispositivo utiliza una red neuronal de convolución (CNN), una arquitectura de aprendizaje automático utilizada para clasificar imágenes, y una cámara para reconocer cualquier billete estadounidense colocado por el usuario. En el desarrollo del dispositivo, los componentes principales funcionan por separado, pero fallan cuando se integraban entre sí. El mayor problema que nos encontramos fue el tamaño de la memoria del microcontrolador. Las secciones separadas costaban demasiada memoria y cuando se juntaban impedían su funcionamiento.

Palabras clave: billete dólar, modelo clasificación, Convolutional Neural Network (CNN)

1. INTRODUCCIÓN

1.1. Problema

En Estados Unidos, los billetes no pueden ser identificados fácilmente por las personas con discapacidad visual o ceguera. Esta realidad es exclusivamente estadounidense, ya que Estados Unidos es una de los pocos países que no dispone de distintivos táctiles ni de tamaños diferentes en sus billetes [1]. Sin la ayuda de otra persona, identificar el valor de los billetes suele ser una tarea difícil y, para quienes viven solos, casi imposible. Para ayudar a las personas con deficiencias visuales, proponemos un dispositivo de sobremesa que puede identificar el valor de un billete de curso legal en Estados Unidos, el Bill Tech Dollar Identifier.

1.2. Solución

Bill Tech Dollar Identifier es un sistema integrado que permite conocer el valor del billete mediante la clasificación por imágenes. El usuario coloca el billete en una bandeja blanca y comienza el proceso pulsando un botón en el dispositivo físico. Tras un breve retardo, el dispositivo toma una fotografía del billete y, al cabo de unos segundos, el microprocesador es capaz de identificar correctamente el valor del billete y emitir mediante secuencias de pitidos su valor por el buzzer integrado.

Este dispositivo se diferencia de las soluciones como el uso de aplicaciones móviles porque no requiere un teléfono para utilizarlo. El dispositivo sigue unos estándares de diseño para facilitar su uso que lo hacen más accesible. El Bill Tech Dollar Identifier sólo utiliza un botón y un interruptor de encendido como entradas del usuario, lo que permite que éste se familiarice fácilmente con los controles.

1.3. Dispositivo Final

La Ilustración 1 muestra el diseño final del dispositivo. El dispositivo utiliza una bandeja blanca para que el usuario deposite el billete. La bandeja blanca permite que haya menos ruido de fondo para la cámara y, en última instancia, para la CNN. La cámara está montada encima de la bandeja para obtener una mejor imagen del billete. Además de la colocación de la cámara, el dispositivo también tiene un LED para iluminar el billete, lo que permite obtener una foto de mayor calidad.

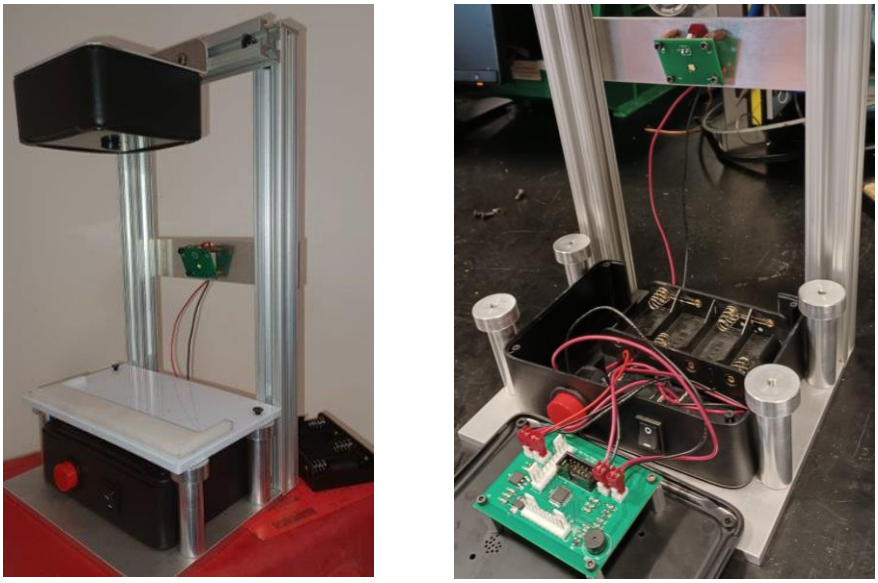


Ilustración 1. Diseño Final

2. DISEÑO

2.1. Diagrama de Bloques

En el diseño final obtenido, la Ilustración 2 muestra cómo se distribuyen los datos y la energía a través del dispositivo. Todo comienza con el subsistema de alimentación; este subsistema valida la energía que llega a todos los demás bloques, lo que permite el buen funcionamiento de los componentes. Este subsistema se encarga de alimentar el sistema del microprocesador que contiene el código que ejecuta el dispositivo, a la vez que contiene el modelo que procesa los datos que envía cámara.

El microcontrolador envía una señal de salida basada en los datos de la imagen procesada y le dice al buzzer que emita un determinado número de pitidos correspondientes al valor del billete, cada billete tiene asociado un número de pitidos. Por ejemplo, si el usuario pone un billete de 1 dólar el dispositivo emite un pitido o si el usuario pone un billete de 5 dólares el dispositivo emite dos pitidos.

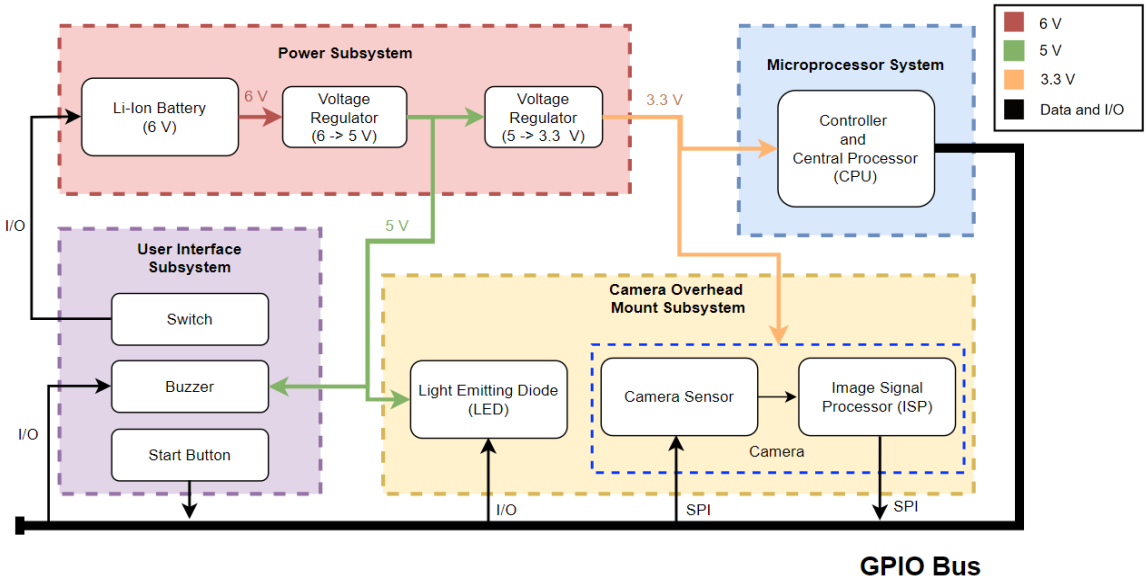


Ilustración 2. Diagrama de Bloques

El dispositivo debe ser capaz de realizar las siguientes acciones:

1. Producir diferentes respuestas de audio para cada billete: 1$, 5$, 10$, 20$ y 100$.

2. Funcionar de forma independiente con batería durante al menos 1 hora.

3. Identificar el billete depositado en la bandeja con una precisión de 0,95 +/- 0,02.

4. Proporcionar una luminosidad adecuada para tomar una buena fotografía del billete.

2.2. Diseño PCB

El dispositivo tiene dos PCBs, una principal que contiene la mayoría de los componentes y circuitos, y una segunda utilizada para soldar el LED. Para crear las PCBs, hemos utilizado KICAD, una herramienta de diseño para crear esquemas eléctricos y diseños de placas. A continuación, se muestran los esquemas y los diseños de ambas PCBs.

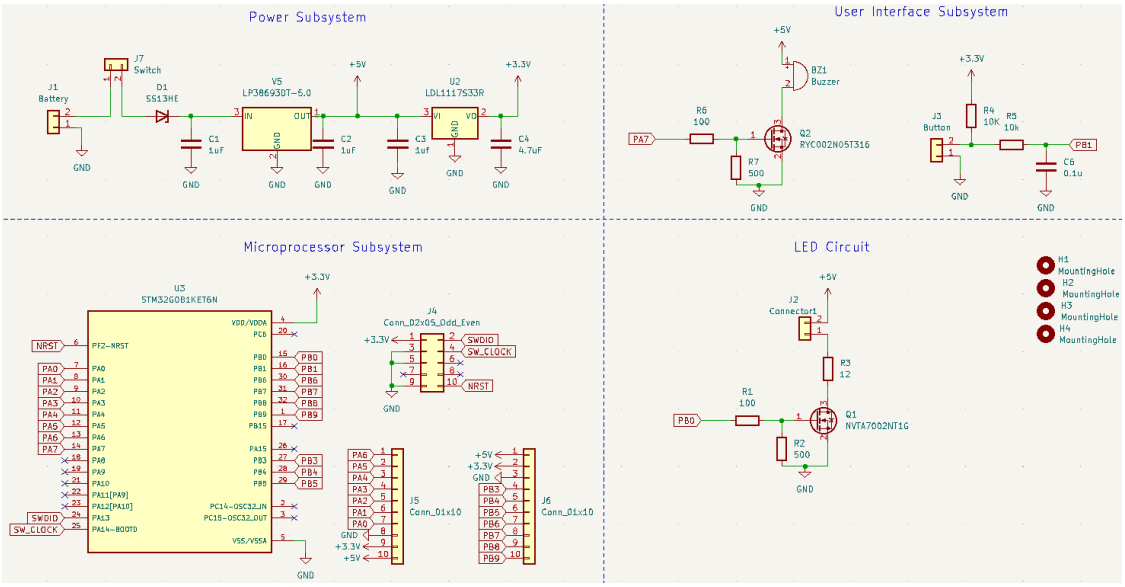


Ilustración 3. Esquema PCB Principal

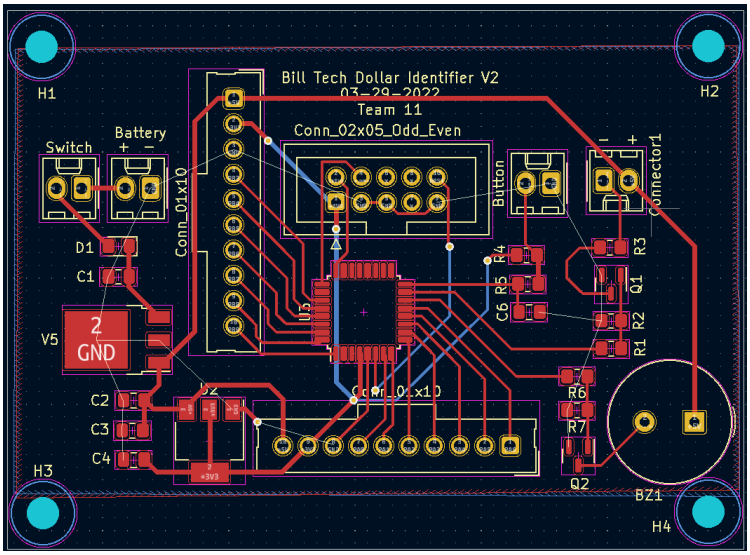


Ilustración 4. Conexiones PCB Principal

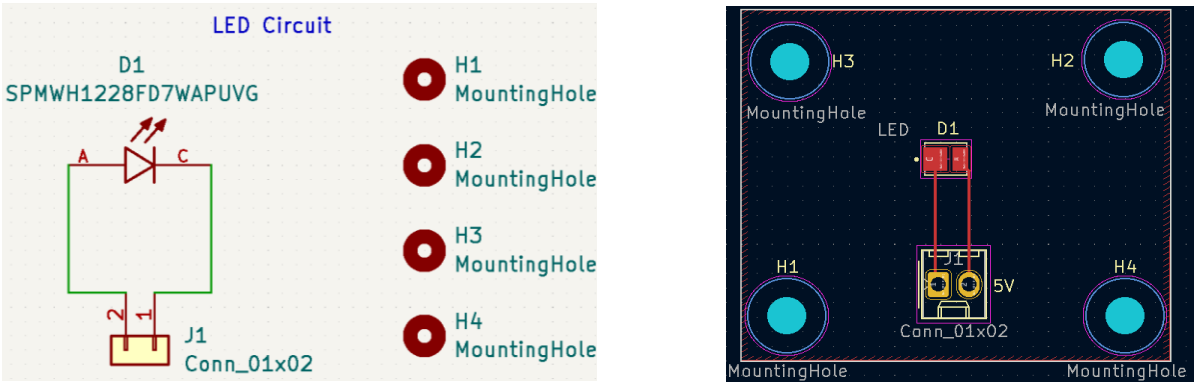


Ilustración 5. Esquema y Conexiones PCB del LED

3. RESULTADOS

3.1. Batería

Hemos utilizado 4 pilas alcalinas tipo C de 1,5 V conectadas en serie, suministrando 6 V al circuito. El primer regulador de tensión baja la tensión de 6 V a 5 V y necesita una tensión de entrada superior a 5,5 V para poder proporcionar la tensión fija de 5 V. Por lo tanto, la tensión mínima de cada pila para que el dispositivo funcione es de 1,4 V.

Para comprobar que la pila puede suministrar energía al circuito durante más de una hora sin que su tensión baje de 5,5 V, probamos su funcionamiento en la peor situación posible. Como la corriente máxima que circula por el circuito es de unos 150 mA, dejamos la batería suministrando esta corriente a través del LED durante una hora. Al cabo de una hora obtuvimos que la tensión era de aproximadamente 5,7 V, por lo que la duración de la batería es suficiente.

Además, el correcto funcionamiento también se puede comprobar con la hoja de datos de la batería. La Ilustración 6 es un gráfico de la hoja de datos de la pila [2] que muestra cómo cambia el voltaje de la pila con el tiempo en función de la corriente utilizada. Incluso si la corriente es superior a la corriente máxima de 150 mA, esta especificación de una hora se seguiría cumpliendo.

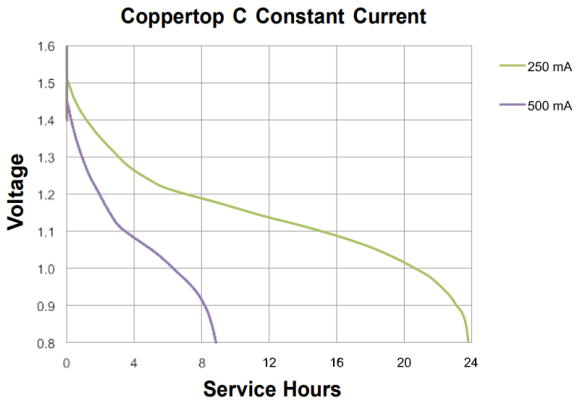


Ilustración 6. Voltaje vs. Horas de Servicio

3.2. LED

Tras realizar un análisis para saber la iluminación necesaria para obtener una foto de buena calidad obtuvimos que era necesario 50 Lm en la bandeja. Para proporcionar esos lúmenes hemos utilizado un LED de montaje superficial. Para comprobar que el LED proporciona los lúmenes necesarios hemos utilizado la aplicación Lux Light Meter. Obtuvimos que los

lúmenes que se veían en la bandeja eran de aproximadamente 67 lúmenes, superior al mínimo requerido de 50 lúmenes.

Además, el correcto funcionamiento también se puede comprobar con la hoja de datos del LED [3]. Como se puede ver en la Ilustración 7, si se suministran 150 mA al LED, éste proporciona entre 61 y 65 Lm, lo que es suficientemente alto para obtener una imagen de buena calidad.

a) Luminous Flux Bins (I_F = 150 mA, T_S= 25°C)

CRI (R_a) Min	Nominal CCT (K)	Product Code	Flux Bin	Flux Range (Φ, lm)
90	2700	SPMWH1228FD7WAW☆VG	VG	56.5 ~ 60.5
	3000	SPMWH1228FD7WAV☆VG	VG	58.0 ~ 62.0
	3500	SPMWH1228FD7WAU☆VG	VG	59.0 ~ 63.0
	4000	SPMWH1228FD7WAT☆VG	VG	61.0 ~ 65.0
	5000	SPMWH1228FD7WAR☆VG	VG	62.0 ~ 66.0
	5700	SPMWH1228FD7WAQ☆VG	VG	61.5 ~ 65.5
	6500	SPMWH1228FD7WAP☆VG	VG	61.0 ~ 65.0

Ilustración 7. LED Luminous Flux

3.3. Modelo de Clasificación de Imágenes

Para probar y verificar que el modelo de clasificación de imágenes funciona correctamente, hemos creado un conjunto de datos de validación mediante un submuestreo aleatorio del conjunto de datos de entrenamiento inicial. Para no obtener un resultado erróneo, estas imágenes se apartaron y no se utilizaron durante el proceso de entrenamiento. Tras el entrenamiento de los modelos ResNet50 y MobileNetV2, pudimos medir una precisión de validación de aproximadamente el 99,9% en el modelo ResNet50 y de aproximadamente el 80% en el modelo MobileNetV2.

MobileNetV2 permite una compresión muy alta por lo que si hay problemas de espacio en el proyecto es una buena opción, sin embargo, en nuestro caso la precisión del modelo es muy importante para evitar clasificar mal un billete. Por lo tanto, hemos decidido utilizar la arquitectura ResNet50. La Ilustración 8 muestra la precisión de los modelos ResNet50 y MobileNet50 respectivamente.

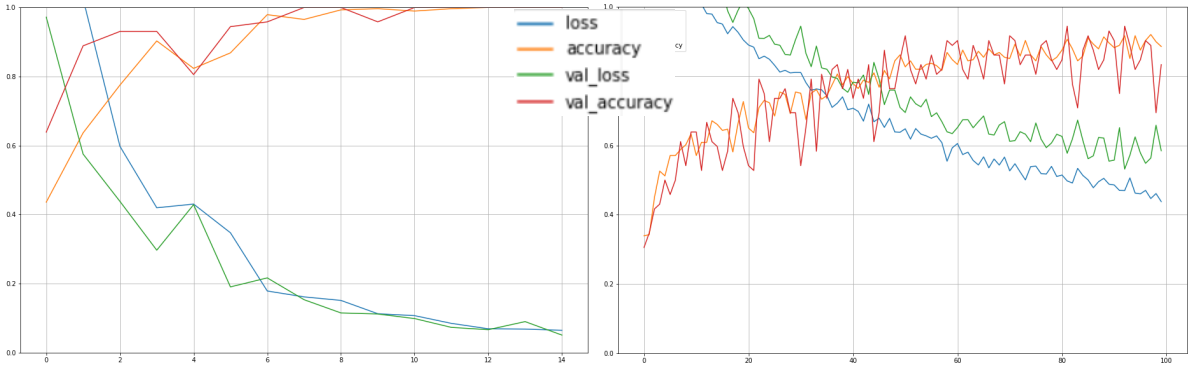


Ilustración 8. Precisión de ResNet50 y MobileNet50

4. CONCLUSIÓN

4.1. Logros

El grupo ha conseguido cumplir los cuatro objetivos. El dispositivo tiene una PCB completamente validada y en funcionamiento donde la fuente de alimentación, el buzzer, el botón, el LED y el microcontrolador cumplen los requisitos necesarios y funcionan correctamente; además, somos capaces de programar y ejecutar código desde el microcontrolador; y hemos podido desarrollar una CNN [4] con una precisión de validación del 99,9%.

4.2. Incertidumbres

La principal razón por la que nuestro proyecto final no es funcional se debe únicamente a la falta de memoria flash del microprocesador. El microprocesador que integramos en nuestro sistema sólo tiene una capacidad de memoria flash de 512 kilobytes. Esta memoria está destinada a ser compartida entre el tamaño del modelo de clasificación de imágenes, el tamaño del buffer de imágenes de la cámara y el tamaño del código utilizado para operar el sistema. Además de estas tareas de almacenamiento, la memoria flash también tendría que servir potencialmente como buffer para la RAM mientras el procesador realiza los cálculos. Esto se debe a que el microprocesador sólo tiene 148 kilobytes de RAM. Sin un chip SRAM adicional o una interfaz de almacenamiento, no existe suficiente capacidad en el microprocesador para contener todo el código necesario para la plena funcionalidad del dispositivo.

4.3. Trabajo Futuro

Para obtener un diseño final totalmente funcional se nos han ocurrido dos ideas, una que consistiría en probar el diseño que hicimos utilizando un ordenador, enviar los datos de la imagen a un ordenador, ejecutar en el ordenador el modelo y enviar una señal de vuelta a la placa para mostrar el valor del billete.

La otra consistiría en integrar algún tipo de almacenamiento adicional en nuestra PCB para que el microprocesador se pueda interconectar. Este almacenamiento adicional podría ser en forma de un chip SRAM o incluso una ranura de expansión MicroSD.

5. REFERENCIAS

[1] “How Do People Who Are Blind or Visually Impaired Identify Money?” https://chicagolighthouse.org/sandys-view/identifying-money/ (accessed 05/02/2022)

[2] Digi-Key, “C-MN1400”, Available. https://www.digikey.com/en/products/detail/duracell-industrial-operations-inc/C-MN1400/13280361 (accessed 03/20/2022)

[3] Samsung. Digi-Key, “SPMWH1228FD7WAP LED”, Available. https://cdn.samsung.com/led/file/resource/2020/07/Data_Sheet_LM281B_Plus_Pro_VG_Rev.0.8.pdf (accessed 03/23/2022)

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'12). Curran Associates Inc., Red Hook, NY, USA, 2012. https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf (accessed 04/06/2022)

BILL TECH DOLLAR IDENTIFIER

Author: Martínez Liñera, Javier.
Supervisor: Fliflet, Arne.
Collaborating Entity: University of Illinois Urbana-Champaign

ABSTRACT

Bill Tech Dollar Identifier is an all-in-one embedded system that performs image classification on currency. To be more specific, the Bill Tech Dolor Identifier uses a convolution neural network (CNN), a machine learning architecture used to classify pictures, and a camera to recognize paper US currency placed by the user. In the development of the device, core components worked separately but failed when integrated with one another. The issue that we met was the memory size of the microcontroller. The separate sections cost too much memory and when brought together broke the system.

Keywords: dollar bill, image classification model, Convolutional Neural Network (CNN)

1 INTRODUCTION

1.1 Problem

In the United States, paper currency cannot be easily identified by those with either visual impairments or blindness. This reality is a uniquely American one, as America is one of the few nations that does not have tactile markers or different sizes in its paper currency [1]. For those who are blind in the United States, many have developed identification systems that require the folding of bills in unique ways [1]. This system works when the identity of the bill is known in advance, but it can become messy when the user is given unknown bills. Without the help of another person this is often a difficult task and for those who live alone, almost impossible. In order to aid those with visual impairments who live alone, we propose a desktop device that can identify any paper legal tender in the United States, the Bill Tech Dollar Identifier.

1.2 Solution

The Bill Tech Dollar Identifier is an all-in-one embedded system that performs image classification on currency The user puts the piece of currency on a tray and pushes a button on the physical device. After a brief delay, the device will take a picture of the bill and after a few seconds, the microprocessor is able to correctly identify the value of the bill and output its value from the buzzer of the device.

This device differs from mobile app solutions because it does not require a phone to use it. The device follows design standards for ease of use that make it more accessible. The Bill Tech Dollar Identifier only uses a button and power switch as inputs from the user, allowing the user to easily become familiar with the controls.

1.3 Final Product

Illustration 1 displays the final design of the device. The device uses a white plate for the user to place currency onto. The white plate allows for less background noise to the camera and ultimately to the CNN. The camera is mounted top down to get a high-quality photo from the bill. In addition to this top mounted camera view, the device also has an LED in order illuminate the bill, resulting in a clearer photo for the actual device.

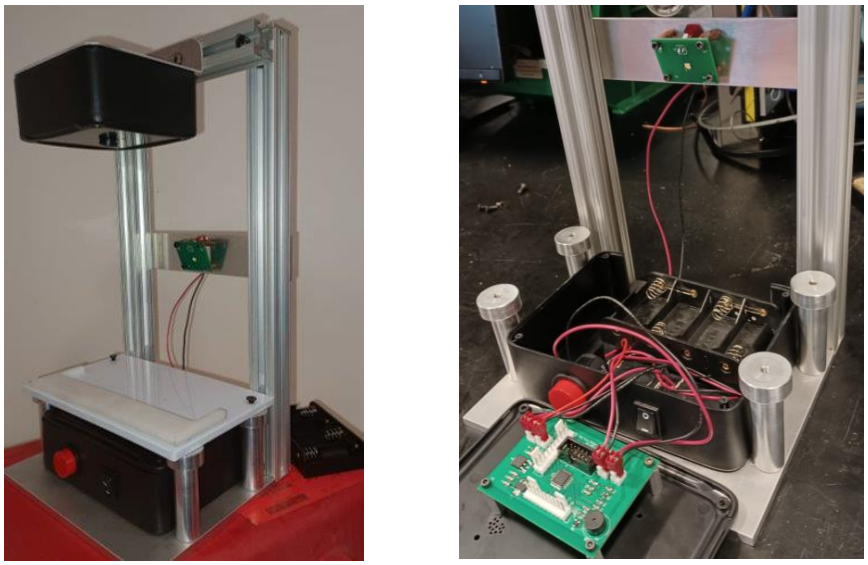


Illustration 1. Final Device

2 DESIGN

2.1 Block Diagram

The final design result in Illustration 2 displays how data and power flow through the device. Everything starts with the power subsystem; this subsystem validates the power going into all other blocks allowing for smooth operation of electrical components. This voltage feeds into the microprocessor system that holds the code that runs the actual device while also holding the model that will be processing camera data. The microcontroller sends IO based on the processed image data and tells the buzzer to output a certain number of beeps corresponding to the correctly identified value of the bill. For example, if the user puts a $1 bill the device beeps once or if the user puts down a $5 bill the device beeps twice.

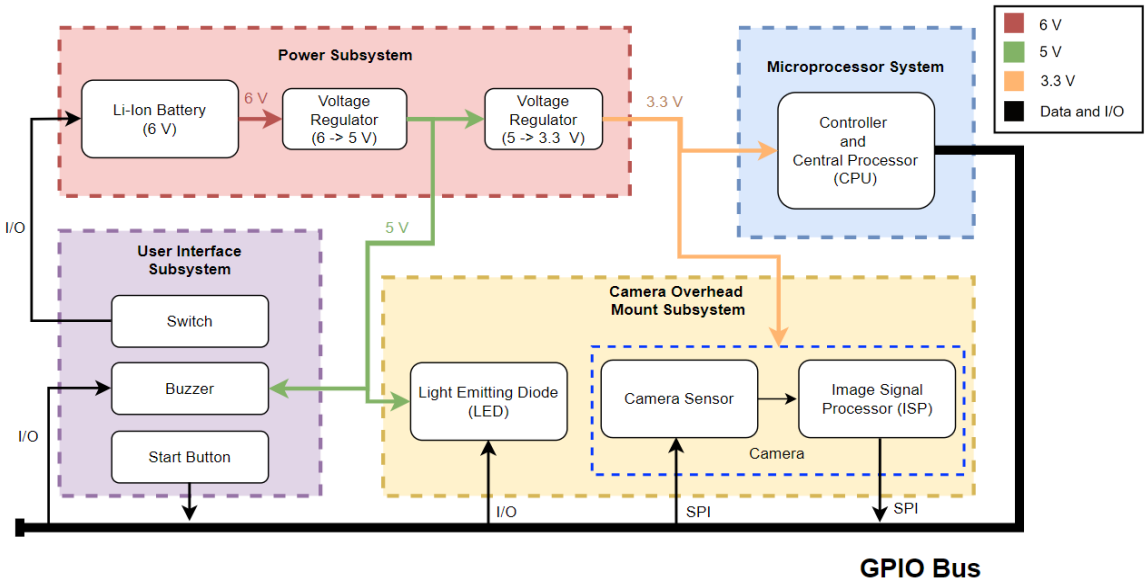


Illustration 2. Block Diagram

In the end we wanted a device that was able to do the following:

1. Produce different audio responses for each bill: $1, $5, $10, $20, and $100
2. Run independently on battery power for at least 1 hour.
3. Identify the bill presented on the loading tray with an accuracy rate of 0.95 +/- 0.02.
4. Provide an appropriate brightness to take a good picture of the bill.

2.2 PCB Design

The device has two PCBs, a main PCB containing most of the components and circuits, and a second PCB to solder the LED. To create the PCBs, we have used KICAD, a design tool for creating electrical schematics and board layouts. The schematics and PCB layouts for both PCBs are shown below.

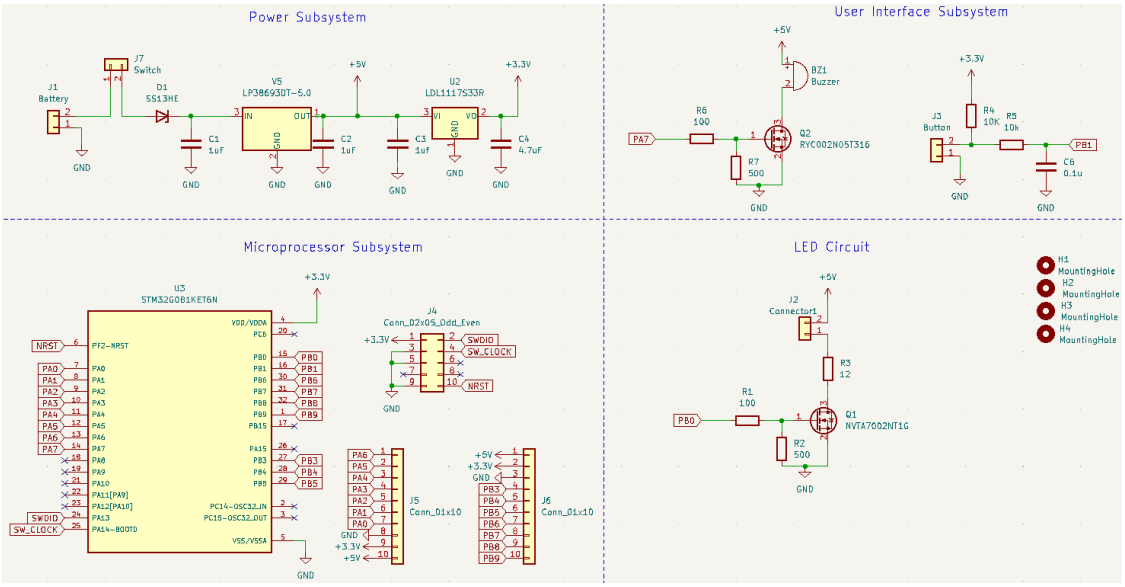


Illustration 3. Main PCB Schematics

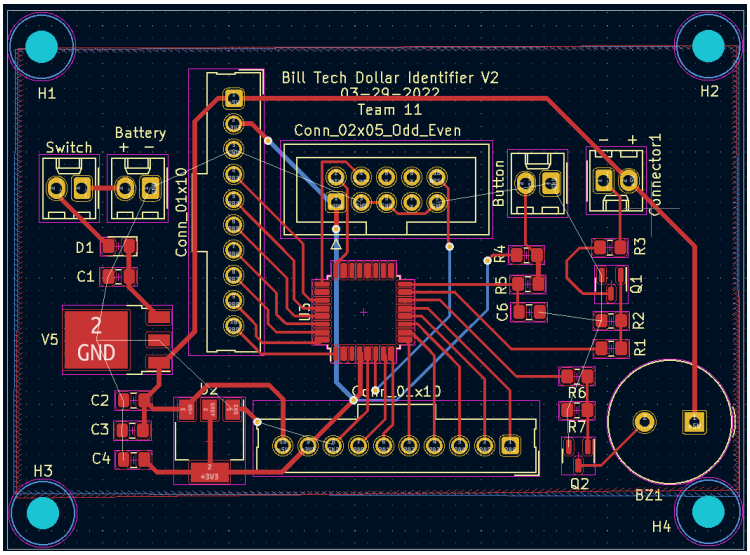


Illustration 4. Main PCB Layout

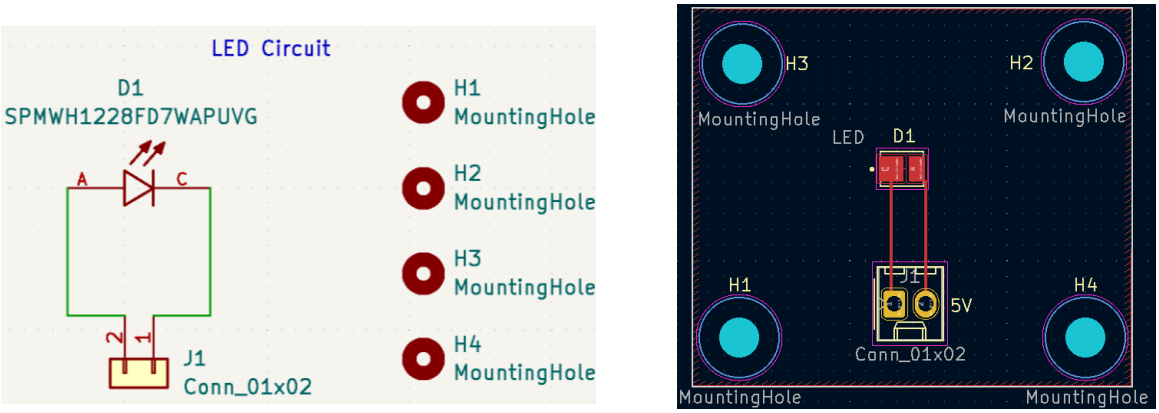


Illustration 5. LED PCB Schematics and Layout

3 DESIGN VERIFICATION

3.1 Battery

We have used 4 1.5 V C-type alkaline batteries connected in series, supplying 6 V to the circuit. Batteries discharge as they supply power to a circuit, therefore, their voltage decreases. The voltage regulator that drops the voltage from 6 V to 5 V needs an input voltage higher than 5.5 V to be able to provide the 5 V fixed voltage. As we are using 4 batteries of 1.5 V, the minimum voltage of each battery for the circuit to work is 1.4 V.

To check that the battery could supply power to the circuit for more than one hour without its voltage going below 5.5 V, we tested it in the worst possible situation. As the maximum current flowing through the circuit is about 150 mA, we left the battery supplying this current through a resistor for one hour. After one hour we obtained that the voltage was approximately 5.7 V, therefore, the battery lifetime is enough.

Moreover, it can also be verified with the battery datasheet [2]. Illustration 6 is a graph from the battery datasheet that shows how the battery voltage changes over time depending on the current used. Even if the current is greater than the maximum current 150 mA this one-hour specification would still be met.

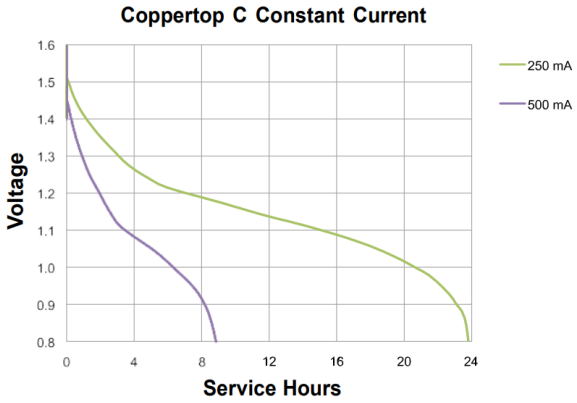


Illustration 6. Voltage vs. Service Hours

3.2 LED

The minimum luminance flux required to get a good illumination of the tray is 50 Lm. To provide those lumens we have used an LED with sufficient power to take a good picture of the bill. To check that the LED is providing the required lumens we have used the Lux Light Meter application. We obtained that the lumens seen in the tray were approximately 63 lumens, higher than the required minimum of 50 lumens.

On the other hand, we were able to verify that it is providing the required lumens by using the LED datasheet [3]. As can be seen in Illustration 7, if 150 mA is supplied to the LED, it provides between 61 and 65 Lm, which is high enough to obtain a good image.

a) Luminous Flux Bins (I_F = 150 mA, T_S= 25°C)

CRI (R_a) Min	Nominal CCT (K)	Product Code	Flux Bin	Flux Range (Φ, lm)
90	2700	SPMWH1228FD7WAW☆VG	VG	56.5 ~ 60.5
	3000	SPMWH1228FD7WAV☆VG	VG	58.0 ~ 62.0
	3500	SPMWH1228FD7WAU☆VG	VG	59.0 ~ 63.0
	4000	SPMWH1228FD7WAT☆VG	VG	61.0 ~ 65.0
	5000	SPMWH1228FD7WAR☆VG	VG	62.0 ~ 66.0
	5700	SPMWH1228FD7WAQ☆VG	VG	61.5 ~ 65.5
	6500	SPMWH1228FD7WAP☆VG	VG	61.0 ~ 65.0

Illustration 7. LED Luminous Flux

3.3 Image Classification Model

To test and verify that the image classification model works correctly we created a validation data set by randomly subsampling the initial training data set. In order not to obtain an erroneous result, these images were set aside and not used during the training process. After each iteration of the training process, the validation data set was run through the model in order to assess the accuracy of the network. After training the ResNet50 and MobileNetV2 models, we were able to measure a validation accuracy of approximately 99.9% in the ResNet50 model and approximately 80% in the MobileNetV2 model.

MobileNetV2 allows a very high compression so if there are tight space issues in the project it is a good choice, however, in our case the accuracy of the model is very important to not misclassify a bill. Therefore, we have decided to use the ResNet50 architecture. Illustration 8 shows the accuracy of the ResNet50 and MobileNet50 models.

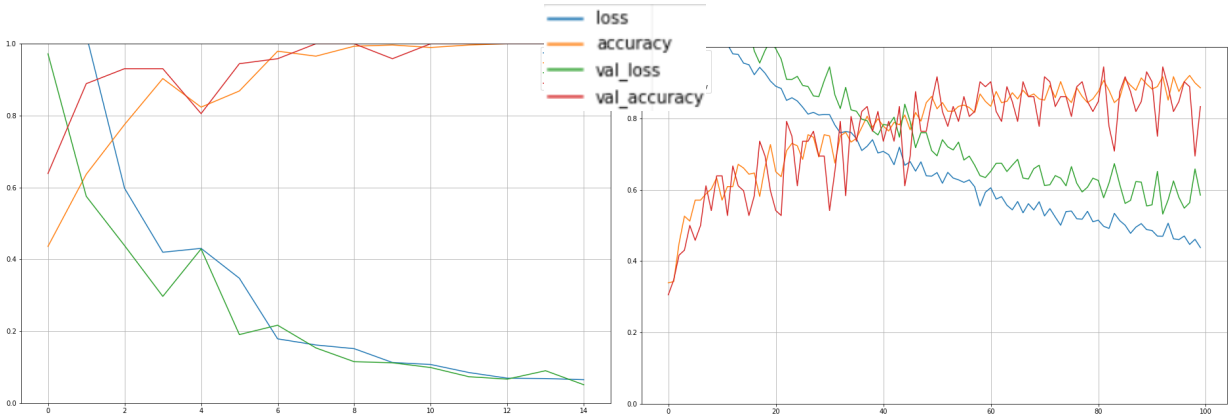


Illustration 8. ResNet50 and MobileNet50 Accuracy

4 CONCLUSION

4.1 Accomplishments

At the end of the day, the group has managed to meet these four objectives. The device has a completely validated and working PCB where the power supply, the buzzer, the button, the LED, and the microcontroller meet the requirements needed and work correctly; furthermore, we are able to program and execute code from the microcontroller; and we were able to develop a CNN [4] with 99.9% validation accuracy.

4.2 Uncertainties

The main reason why our final project wasn't functional was due solely to the lack of flash memory onboard the microprocessor. The microprocessor that we integrated into our system only had a flash memory capacity of 512 kilobytes.

This memory is meant to be shared between storing the image classification model, storing the image buffer from the camera, and storing the controller code to operate the system. In addition to these storage tasks, the flash memory would also potentially need to serve as a buffer for the RAM while the processor is performing calculations. This is because the microprocessor only had 148 kilobytes of RAM. Without an additional SRAM chip or storage interface, there was not enough capacity built into the microprocessor to contain all the necessary code required for the full functionality of the device.

4.3 Future Work

In order to obtain a fully functional final design we have come up with 2 ideas, one that would involve testing the design we made using a computer, send the image data to a computer and run the model; and the other would consist of integrating some form of additional storage onto our PCB for the microprocessor to interface with. This could be in the form of an SRAM chip or even a MicroSD expansion slot.

5 REFERENCES

[1] "How Do People Who Are Blind or Visually Impaired Identify Money?" https://chicagolighthouse.org/sandys-view/identifying-money/ (accessed 05/02/2022)

[2] Digi-Key, "C-MN1400", Available. https://www.digikey.com/en/products/detail/duracell-industrial-operations-inc/C-MN1400/13280361 (accessed 03/20/2022)

[3] Samsung. Digi-Key, "SPMWH1228FD7WAP LED", Available. https://cdn.samsung.com/led/file/resource/2020/07/Data_Sheet_LM281B_Plus_Pro_VG_Rev.0.8.pdf (accessed 03/23/2022)

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'12). Curran Associates Inc., Red Hook, NY, USA, 2012. https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf (accessed 04/06/2022)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. INTRODUCTION

In the United States, paper currency cannot be easily identified by those with either visual impairments or blindness. The fact is that all bills of this currency, from the one dollar to the one hundred dollars, have the same shape and the same texture [1].

Of the more than 180 countries that use paper money, only the bills of the United States have the same color and size in all formats. More than 100 of these nations vary the size of the bills according to their value, and all the others include some distinctive feature that helps the visually impaired to identify their value [2]. For example, euros are easy to identify, the larger the size, the higher the value of the bill.

These features to identify the bills work when the identity of the bill is known in advance, but it can become messy when the user is given unknown bills. Without the help of another person this is often a difficult task and for those who live alone, almost impossible. In order to aid those with visual impairments, we propose a desktop device that can identify any paper legal tender in the United States, the Bill Tech Dollar Identifier.

The Bill Tech Dollar Identifier is a desktop device that allows you to identify the value of an American bill in a fast, cheap and easy way. The device is powered by 4 1.5 V C-type alkaline batteries allowing the user to identify the bill without the need to connect the device to a wall plug. The user puts the piece of currency on a tray and pushes a button on the physical device. After a brief delay, the device will take a picture of the bill and after a few seconds, the microprocessor is able to correctly identify the value of the bill and output its value from the buzzer of the device.

This device differs from mobile app solutions because it does not require a phone to use it. The device follows design standards for ease of use that make it more accessible. The Bill Tech Dollar Identifier only uses a button and power switch as inputs from the user, allowing the user to easily become familiar with the controls.

CHAPTER 2. TECHNOLOGY DESCRIPTION

The device has two PCBs, a main PCB containing most of the components and circuits, and a second PCB to solder the LED. To create the PCBs, we have used KICAD, a design tool for creating electrical schematics and board layouts. The process to create a PCB has three steps: add components and define the electrical connections of our design in a schematic, assign packages/footprints to the components in the schematic and define PCB dimensions, component locations, and copper traces in a layout.

Moreover, to perform the bill classification, we tested several architectures for convolutional neural network (CNN) [3]. CNN is a class of artificial neural network used to process and identify images. It is used for image classification, identifying objects or face recognition. We tried to use the LeNet and AlexNet architectures, but these first attempts resulted in low validation accuracy. LeNet is a small architecture mostly used for detecting handwritten and AlexNet is very similar to LeNet but it is bigger, with more layers, filters and data augmentation.

Switching to the more complex ResNet50 allowed us to obtain a validation accuracy reading of 99.9%. ResNet50 is a very deep feedforward neural network that has hundreds of layers. However, with the packaged model requiring over 2.37 gigabytes of space, it was impossible to load onto our microprocessor. We finally tried to use the MobileNetV2 architecture to try to downsize the model to fit onto the microprocessor. MobileNetV2 is a smaller CNN than ResNet50 with 43 layers deep, however, the accuracy obtained was not high enough.

CHAPTER 3. STATE OF THE ART

There are other devices that are also capable of reading the value of bills and returning their value, for example, bank teller machines can read the value of money that is introduced. However, they are very inaccessible as you have to go to the street to use them and they do not give you the value of each of the bills you insert, but the total sum. It is true that you can enter a single bill to get its value, however, the process you have to go through for each reading is very long and can be complicated for visually impaired people.

On the other hand, there are some applications for mobile phones that allow the user to know the value of a bill by taking a picture with the camera. Some of these applications are Cash Reader, IDEAL Currency Identifier or LookTel Money Reader. These applications are a kind of scanner capable of instantly identifying banknotes. They speak aloud the value of each bill scanned.

It is true that having an application on the cell phone that allows the user to know the value of the bill would help identifying the value of a bill wherever the user is, since we carry our mobile phones everywhere. However, we thought about the difficulty of using a mobile phone application for a blind person. For this reason, instead of making an application, we decided to make a device that would allow to know the value of the bill in a simple way. It is much simpler and faster to use since it tells you the value of the bill by simply pressing a button. It is true that the device is not as portable as a cell phone, but it allows to reduce the inequalities for all visually impaired people thanks to its ease of use.

CHAPTER 4. FRAMEWORK

4.1 OBJECTIVES

In order to obtain a device that can fulfill its function, to show the value of a banknote, it needs to fulfill at least 4 main objectives. These objectives are:

- The machine is able to correctly identify the bill presented on the loading tray with an accuracy rate of 0.90 and an error rate of 0.03.

- The machine should produce differentiable audible responses upon identification of different bill types. For instance, a one-dollar bill would produce one beep while a five-dollar bill would produce two beeps.

- The machine should be able to run independently on battery power for at least 1 hour.

- The LED should illuminate the bill with the appropriate brightness so that a good picture can be taken.

4.2 METHODOLOGY

4.2.1 CIRCUIT DESIGN

When we designed the circuits of our device, we have always thought in making it easy to use for the user. To do this we have done research to know what components and circuits to use, and once we knew the circuits, we investigated in Digi-Key and Mouser Electronics websites to get the component that best suited our needs.

4.2.2 PCB DESIGN

The device required two PCBs, a main PCB where most of the circuitry is located and a second PCB for the LED surface mount. The PCBs have been designed using the free KiCad program. The design process required schematics of the circuits, footprint assignment and PCB layout. Once the design was done, we ordered the PCBs through PCBway, a company specialized in PCBs manufacturing.

4.2.3 EXTERNAL CONNECTIONS

The components that the device has and require external connections are the button, the switch, the LED, the battery, and the camera.

All of them have their connections determined on the PCB itself. However, we decided to use external connections for two reasons. On the one hand, as we did not know the connections that the camera will need, the PCB design was made with external output connections that allowed access to the microcontroller pins, ground and the 3.3 V and 5 V voltages sources. On the other hand, we decided to use external connections because once the microcontroller was soldered, if we had a problem with the pins used, we could still access the microcontroller using a different pin.

4.2.4 BUILDING THE MACHINE LEARNING MODEL

For classifying the bills on the platform, we used a Convolutional Neural Network (CNN) as our image classification model [3]. We gathered hundreds of training images for our dataset settled on implementing two architectures, ResNet50 and MobileNetv2, using transfer learning, to train the network.

We used the TensorFlow framework to code the network. TensorFlow is a python library that allows easy manipulation and convolution for tensors that are used in a neural network [4]. We used this library because it allowed us to introduce the model in the microcontroller.

4.2.5 PROGRAMMING

To program the microcontroller, the generated code was sent by serial wire debug. The ARM serial wire debug interface uses a single bidirectional data link. The serial interface is defined if it transfers data asynchronously to a minimum number of pins, it provides an independent clock link, and transfers data synchronously. In addition, we used the C programming language to program the microcontroller.

4.3 SCHEDULE AND ECONOMIC ESTIMATION

4.3.1 SCHEDULE

Table 1 shows how work was divided during the 10-week given for development time.

Week	Task
02/21/22	- Research image classification models - Circuit schematics
02/28/22	- Finish circuit schematics - PCB design
03/07/22	- Refine PCB design - Start training the model
03/14/22	- Order PCB and components - Prototype with development board
03/21/22	- Start programming microcontroller - Send device idea to the machine shop
03/28/22	- Solder and test PCB design - Finish Programming microcontroller
04/04/22	- Fix possible problems - Continue building the model
04/11/22	- Test software operation
04/18/22	- Finish the model - Fix possible problems
04/25/22	- Final assembly - Test the device
05/02/22	- Presentation - Write final paper
Now **05/10/2022**	- Summit final paper - Senior project finished

Table 1. Schedule

4.3.2 COSTS

This section shows the project costs. The costs have been divided into the cost of the materials used and the cost of the hours worked by both the team and the machine shop who helped us to build the physical device.

4.3.3 PART COSTS

Table 2 shows a total bill of all parts required for a single run of making the "Bill Tech Dollar Identifier". A lot of the major costs for creating this product are in the one-time purchases that are needed for production. Examples of one-time purchases are, the st-link v2 which is used for programming and can be reused, the TC-2050 Adapter which is also only needed for programming, and the programming cables which are also reusable. Without these three expenses the recurring cost for making one of these devices would be $60.99 which could be brought down even further with bulk ordering.

Part Name	***Part Manufacturer***	***Quantity***	***Price per part***	***Total***
08055C105JAT2A (1 uF Capacitor)	KYOCERA AVX	3	$0.44	$1.32
CL21A475KAQNNNE 4.7 uF Capacitor	Samsung Electro-Mechanics	1	$0.11	$0.11
RMCF0805FT100R 100 Ω Resistor	Stackpole Electronics	2	$0.10	$0.20
RC0805FR-07499RL 500 Ω Resistor	YAGEO	2	$0.10	$0.20
ERJ-P06J120V 12 Ω Resistor	Panasonic Electronic Components	1	$0.13	$0.13
ERJ-P06J103V 10 kΩ Resistor	Panasonic Electronic Components	2	$0.13	$0.26
LP38690DT-5.0/NOPB 5V LDO	Texas Instruments	1	$2.38	$2.38

Continued on next page

Part Name	***Part Manufacturer***	***Quantity***	***Price per part***	***Total***
TC2117-3.3VDBTR 3.3V LDO	Microchip Technology	1	$1.00	$1.00
SPMWH1228FD5WAPUVG White LED	Samsung Semiconductor	1	$0.17	$0.17
NVTA7002NT1G N-channel Mosfet	onsemi	1	$0.43	$0.43
MBR230LS Schottky Diode	SMC Diode Solutions	1	$0.42	$0.42
IE092505-1 Buzzer	DB Unlimited	1	$2.51	$2.51
RYC002N05T316 N-channel 5V Mosfet	Rohm Semiconductor	1	$0.46	$0.46
C-MN1400 1.5V C type batteries	Duracell Industrial Operations	4	$1.19	$4.76
4X C Cell Battery Holder	SDTC Tech Store	1	$4.24	$4.24
3-640440-2 1x02 Crimp Connectors	TE Connectivity AMP Connectors	4	$0.17	$0.68
22232021 1x02 PCB Header	Molex	4	$0.22	$0.88
61201021621 2x05 Even Odd Male Header	Wurth Elektronik	1	$0.46	$0.46
22272101 1x10 Male PCB Header	Molex	2	$0.84	$1.68
PR144C1900 Panel Mount Push Button	E-Switch	1	$2.30	$2.30
A8L-21-11N2 Panel Mount Power Switch	Omron Eletronics	1	$3.72	$3.72
ST-LINK/V2 External Programmer	STMicroelectronics	1	$21.25	$21.25
TC2050-ARM2010 ARM 20-pin to TC2050 Adapter	Tag-Connect	1	$29.95	$29.95

Continued on next page

Part Name	***Part Manufacturer***	***Quantity***	***Price per part***	***Total***
2.54mm Pitch 2 Row 10 Pin Female to Female Wires IDC Ribbon Connector	C & Xanadu	1	$4.00	$4.00
OV2640 2 Megapixels Lens	Arducam	1	$26.00	$26.00
STM32G0B1KET Microcontroller	STMicroelectronics	1	$6.93	$6.93
			Total	$116.44

Table 2. Parts Costs for Final Product

4.3.4 LABOR COSTS

Name	***Hourly Rate***	***Hours***	***Total***	***Total x 2.5***
Javier Martinez	$40.00	120	$4,800.00	$12,000.00
Pratheek Eravelli	$40.00	120	$4,800.00	$12,000.00
Justin Hsieh	$40.00	120	$4,800.00	$12,000.00
Machine Shop Personal	$50.00	20	$1,000.00	$2,500.00
Total				$38,500.00

Table 3. Labor Costs for Development

$$E.\ 1 \quad Total\ Cost = Parts\ Cost + Labor\ Costs = \$38{,}616.44$$

4.4 SUSTAINABLE DEVELOPMENT

The project is aligned with Goal 10 of the ODSs (Objetivos de Desarrollo Sostenible) [5]. Reducing inequalities and ensuring that no one is left behind is an integral part of achieving the Sustainable Development Goals.

Inequality within and between countries is a continuing cause for concern. This project is focused on reducing inequality within one country, that is the United States. In America, because all bills have the same size and are the same to the touch, blind people cannot identify their value.

To solve this problem, several changes have been proposed to the U.S. currency, including making the bills different sizes, embossing dots, or raising the printing of the script. However, these changes have not yet been implemented [2]. Today visually impaired or blind people in the United States use methods such as folding bills to recognize their value. However, this does not work if a person does not tell them the value of the bill beforehand. Therefore, without the help of another person this is often a difficult task and for those who live alone, almost impossible

It is for this reason that we have decided to design a device to reduce the inequalities. The Bill Tech Dollar Identifier is a device that allows the user to know the value of the bill by simply pressing a button. Thanks to this device the user is able to identify the value of the bill without having to rely on what someone else tells him.

CHAPTER 5. DEVELOPED DEVICE

Chapter 5 covers the entire design of the device. It explains the subsystems and the connections between the different blocks, the components used for the device hardware, the programming tools used to program the microcontroller and to build the machine learning model, and the design of the physical product.

5.1 BLOCK DIAGRAM

The device is divided into 4 blocks, power subsystem, microprocessor subsystem, user interface subsystem and camera overhead mount subsystem. Figure 1 displays the different blocks in which the device is structured and how data and power flow through the device.

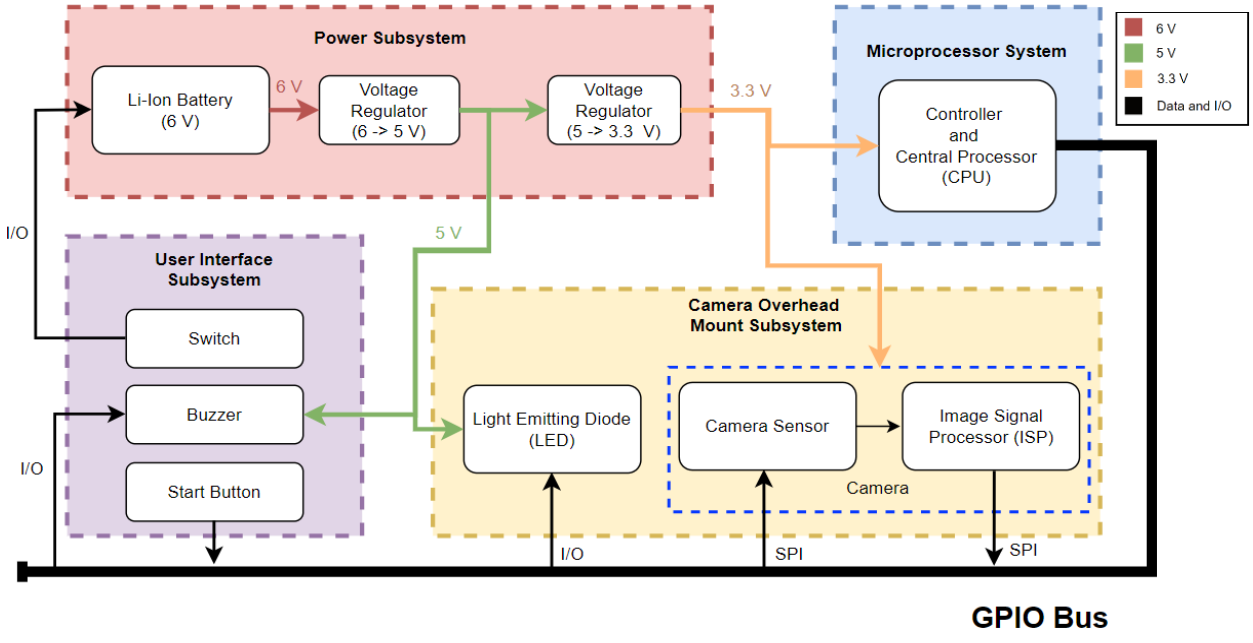


Figure 1. Block Diagram

Everything starts with the power subsystem; this subsystem validates the power going into all other blocks allowing for smooth operation of electrical components. This voltage feeds into the microprocessor system that holds the code that runs the actual device while also holding the model that will be processing camera data. The microcontroller sends IO based on the button state and the processed image data. If the microcontroller receives a signal that the button has been activated, the LED turns on and a picture of the bill is taken. On the other hand, depending on the value of the processed image data it sends a signal to the buzzer to output a certain number of beeps corresponding to the correctly identified value of the bill. For example, if the user puts a $1 bill on the tray, the device beeps once or if the user puts down a $5 bill the device beeps twice.

5.2 POWER SUBSYSTEM

This block covers the power distribution and voltage regulation required to safely operate each component. Using a battery, a Schottky diode and two voltage regulators, this block is able to safely power multiple components with varying voltage requirements. This system interacts with the LED, the buzzer, the camera, and the microprocessor. This power subsystem works in tandem with the microprocessor to both regulate voltage as well as turning off voltage levels to each of the components. As will be explained later, the microprocessor is also able to regulate the voltage on its pins. This allows outputting 3.3 V or 0 V to activate or deactivate components such as the LED or the buzzer.

A 6 V source is being used to power the device and the voltage is dropped to 5 V and to 3.3 V. The 3.3 V is used to power the microcontroller, the button debouncing circuit and the camera. The 5 V is used to power the buzzer and LED circuits. Figure 2 shows the schematic of the power subsystem circuit.

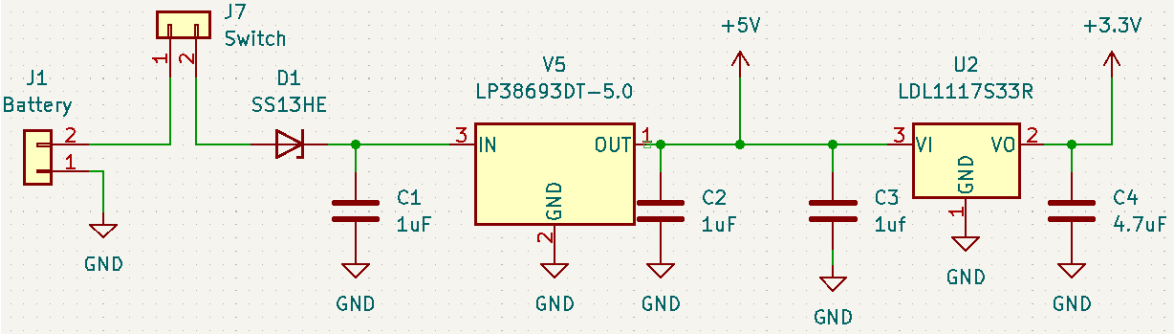


Figure 2. Power Subsystem Schematic

5.2.1 BATTERY

A 6 V battery has been used to power the device for more than 1 hour. To provide the 6 V to the circuit, we have used 4 1.5 V C-type alkaline batteries connected in series. The voltage of these batteries is added together to obtain the 6 V needed. The device uses 1.5 V C-type alkaline batteries so that when the battery runs out the user can easily replace them.

Figure 3 shows the 4X C Cell Battery Holder with Wire Leads and the batteries used to power the circuit.

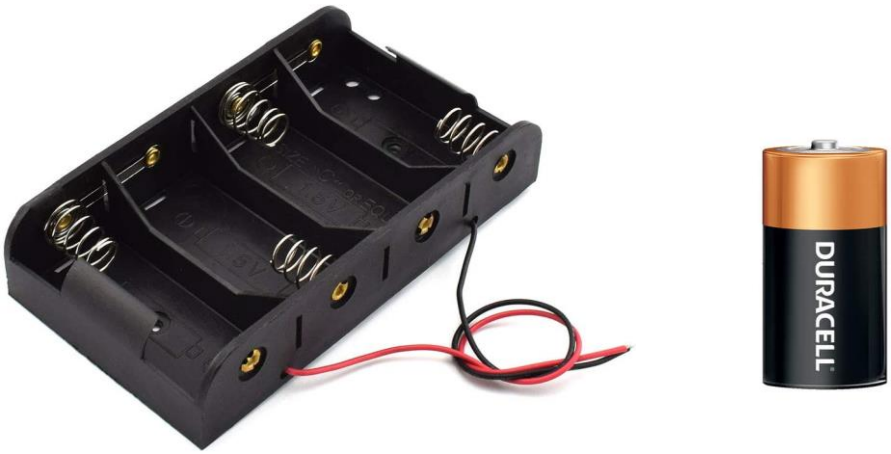


Figure 3. Battery Holder with Wire Leads and Battery

5.2.2 SCHOTTKY DIODE

In case the user connects the battery in opposite polarity, there must be a safety system that prevents the currents from flowing in the opposite direction. A simple way to protect the circuit is to use a diode in series with the source. To solve the problem, we could have used a normal diode, however, we used a Schottky diode because it reduces voltage loss and power dissipation.

The diode certainly cannot "undo" the reverse polarity, but it can isolate the rest of the circuit from this condition simply because it will not conduct current when the cathode voltage is greater than the anode voltage. Therefore, in a reverse bias situation, no harmful reverse currents can flow and the voltage across the load is not the same as the reverse supply voltage because the diode operates as an open circuit.

It has been chosen a diode that allows a high current flow of 1 A, but that its current reverse leakage is small 50 µA at 30 V to get a better protection of the circuit.

5.2.3 VOLTAGE REGULATORS

We needed 3.3 V and 5 V for our project and to obtain these voltages we are using a 6 V battery and two step-down voltage regulators. We are using LDOs instead of Buck Converters because of the simplicity of their design, the smaller device size, and the absence of switching noise. The disadvantage is that linear DC regulators must dissipate power and thus heat.

$$E.\ 2 \qquad Power\ Dissipation = (Vi - Vo) * Io$$

To minimize the power loss, given equation E. 2, the difference between the input voltage and the output voltage must be reduced. Therefore, the power subsystem has a linear structure. The LDOs are placed to step down the voltage first from 6 V to 5 V and then from 5 V to 3.3 V instead of directly converting 6 V to 3.3 V. Despite using LDOs to have the lowest possible noise, we have used capacitors following the manufacturer's recommendations to reduce the noise at the output.

5.3 USER INTERFACE SUBSYSTEM

The User Interface subsystem includes the switch, start button and the speaker. It is through these elements that the user physically interacts with our device. The switch and the button detect the user's inputs and send the signal to the microcontroller. The speaker works as an output, producing differentiable audible responses so the user can identify the bill value.

5.3.1 BUZZER

The buzzer used to reproduce the beeps needs a current of 30 mA and a voltage range between 3 V and 7 V. As the maximum output current of the microcontroller is 20 mA and the buzzer needs 30 mA, an N-Channel MOSFET has been used. Figure 4 shows the circuit designed to connect the buzzer.

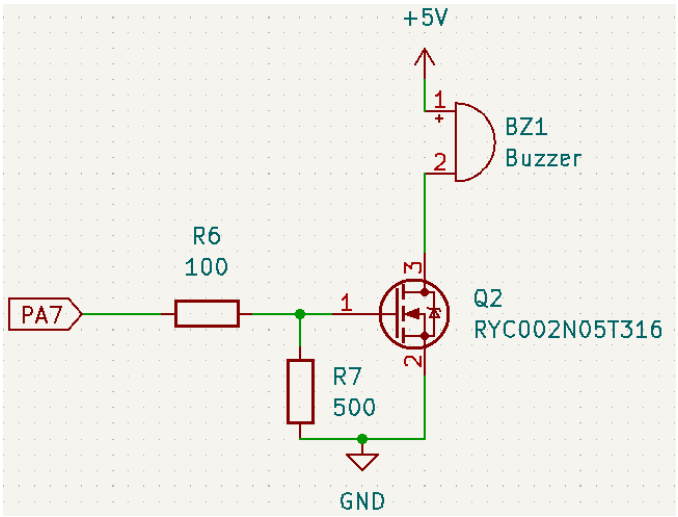


Figure 4. Buzzer Circuit Schematic

The MOSFET works as a switch and allows to provide the current that the buzzer needs. Depending on the output voltage of the microcontroller on its pin, the gate voltage of the MOSFET will be high enough to activate it, otherwise no current would flow through the buzzer. This switch consists of an NMOS transistor that is alternately driven between the triode and cutoff regions.

The transistor is in triode mode if the gate voltage is higher than the threshold voltage (Vgs > Vth) and if the drain voltage is lower than the overdrive voltage (Vds < Vgs – Vth). On the other hand, the transistor is in cutoff region if the gate voltage is lower than the threshold voltage (Vgs < Vth). When the microcontroller outputs 3.3 V on its pin the MOSFET operates in triode mode and follows the following calculations:

$$Vgs = V12 = 3.3 * \frac{500}{600} = 2.75\ V \qquad Vds = 0.1\ V$$

$$Io = \frac{3.3}{600} = 5.5\ mA < Imax = 20\ mA \qquad Id = 30\ mA$$

Given these drain and gate voltages and a threshold voltage of 0.8 V we can be sure that the transistor is in the triode region. Figure 5 is a graph taken from the component datasheet [6] showing the region in which the transistor is operating and shows that the circuit provides the buzzer the 30 mA it needs.

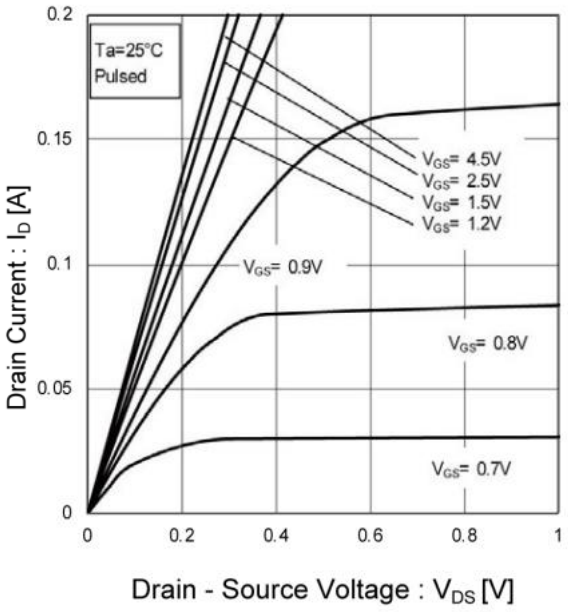


Figure 5. Buffer MOSFET (Id vs. Vds)

5.3.2 BUTTON

When the button is pressed, the button connects and disconnects to ground before finally settling down. There are three commonly used methods to prevent the circuit from switch bouncing: hardware debouncing, RC debouncing and switch debouncing IC. We ha used a low pass filter has been designed to eliminate this disturbance [7]. A simple hardware solution using a capacitor and a resistor, RC filter, is used to prevent the button bouncing. The debouncing circuit is shown in Figure 6.

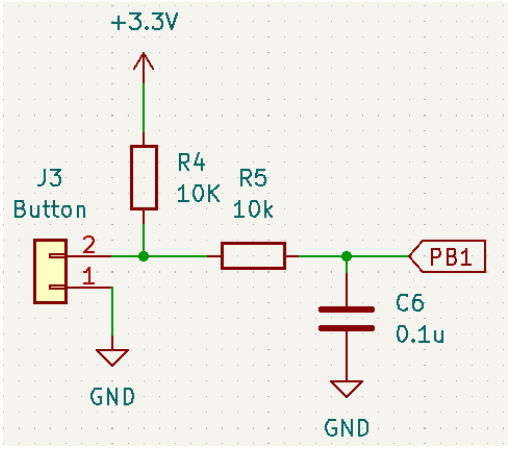


Figure 6. Button Debouncing Schematic

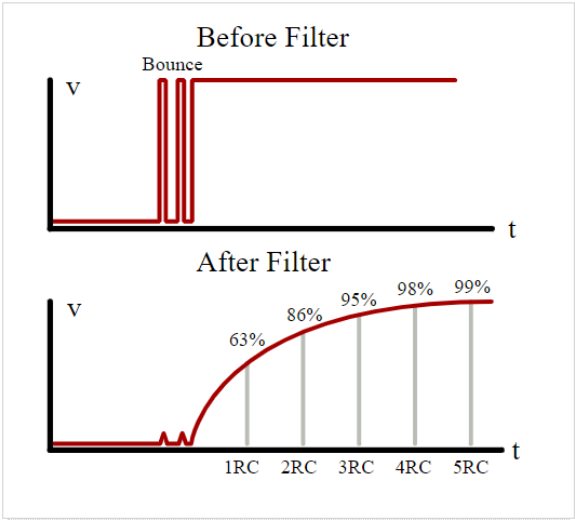


Figure 7. Filtering out button bounce with a hardware debounce circuit

5.4 CAMERA SUBSYSTEM

This system ties directly with the microprocessor and the user interface. When the user presses the start button, with the device turned on, the microprocessors system sends a control signal to the camera and the LED. The LED will then turn on to illuminate the tray, which holds the dollar, and the camera system will take the picture. The ISP on the camera will then process the photo and go through Analog to Digital Conversion and send this data through SPI and I2C to the microprocessor for image analysis.

5.4.1 LED

The Bill Tech Dollar Identifier is designed to assist the visually impaired. As our device works by taking a picture of a bill and using a model to know its value, taking a clear picture is an important component in our project. There is a possibility that the device may be in a poorly lit location, therefore, an LED has been incorporated to ensure that a high-quality photo of the bill is taken.

The LED's position and illuminance are important to get a good picture. In order to maintain its functionality in the worst-case condition—complete darkness—our system must be able to self-illuminate approximately 30 Lm/m2 at the tray. A fixture mount too close or too far away may cause overexposure or underexposure of the resultant image respectively.

Using the inverse square law shown in E. 3 we can find the illuminance (Lm/m2) of a surface with respect to a source at distance d. P is the luminous flux of the LED itself and d is the distance of source from the surface of measurement.

$$E.\,3 \quad E = \frac{P}{4\pi d^2}$$

Starting with the minimum distance of 25 cm to facilitate access to the user to place the bill on the tray and a minimum illuminance of 30 lm/m2, complete darkness, we can plot our distance to a luminous flux graph. Figure 8 shows the luminous flux as a function of distance.

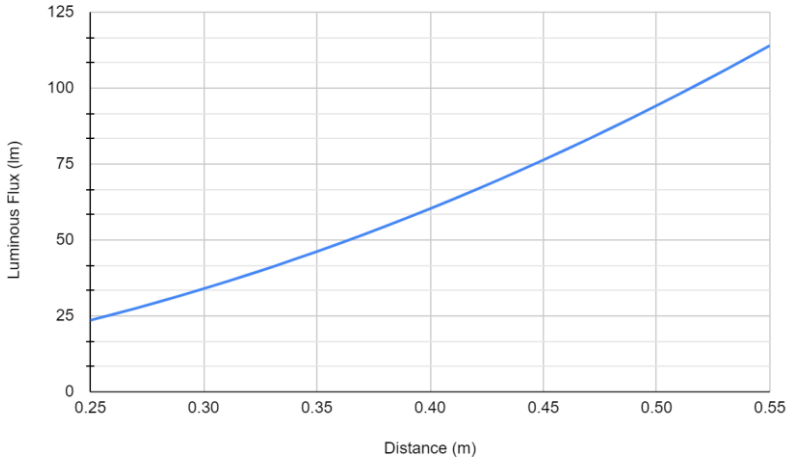


Figure 8. Luminous Flux vs. Distance from Tray

Due to the cost of the camera sensor and memory limitations of the processor, the resolution of the image is 0.3 megapixels or around 640 x 480 pixels with a focal length of 3.6 mm and a field of view of 25 degrees. In order to find the perceived resolution from a given distance, we need to first find the focal pixel F which is calculated using the image resolution and the camera's field of view in E. 4. F is the focal pixel, w is the horizontal resolution in pixels, and Φ is the field of view in degrees.

$$E.\,4 \quad F = \frac{w}{2tan\left(\frac{\phi\pi}{360}\right)}$$

Using the focal pixel given by the above equation we can then use triangular similarity to approximate the perceived resolution, P, using E. 5. P is the perceived resolution, F is the focal pixel, is the width of the object, and d is the distance of the object from the camera.

$$E.\,5 \quad P = \frac{Fw}{d}$$

Using the specifications of our camera given above and in E. 4, we found a focal pixel of 14443.43. Applying this to E. 5, Figure 9 shows the perceived image resolution as a function of distance.

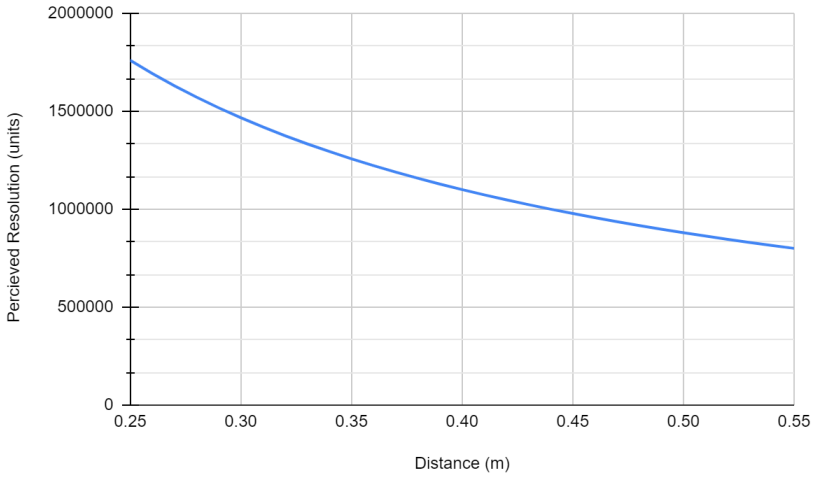


Figure 9. Perceived Resolution vs. Distance from Tray

Since the perceived resolution is not bound to a tangible reference measurement, we have normalized both the illuminance and perceived resolution charts and plot them against each other to find the theoretical optimal distance for the fixture to be mounted. Figure 10 shows this intersecting value to be roughly 0.37 meters from the tray. Therefore, the LED must have a minimum luminance flux of 50 Lm to still achieve 30 Lm/m2 on the surface of the tray.

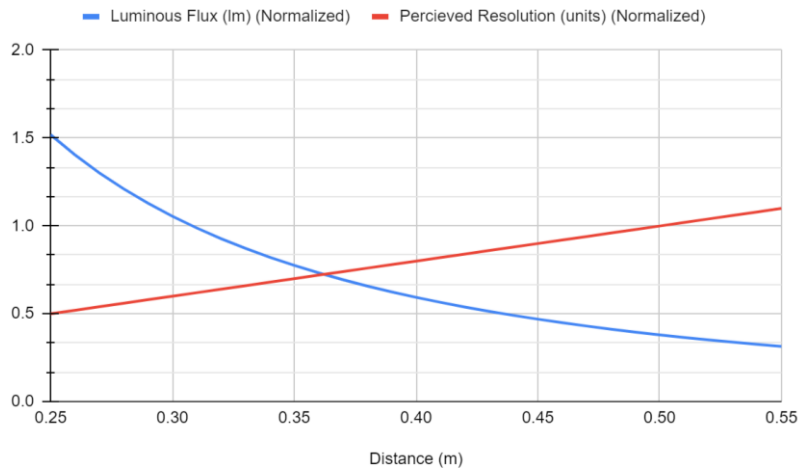


Figure 10. Normalized Illuminance vs. Normalized Perceived Resolution

The minimum luminance flux required to get a good illumination of the tray is 50 Lm. Since an LED with sufficient power to illuminate the bill is needed, we are using a surface mount LED. To illuminate the tray, the LED is placed in a different place than the main PCB, therefore a second PCB has been designed to solder the LED. The section discusses the PCBs design in more depth.

The LED used in the device provides more than the 50 Lm needed. It needs a current of 150 mA, and a voltage of 3.3 V. As the maximum output current of the microcontroller is 20 mA, an N-Channel MOSFET it is used working as a switch and allowing to provide the current that the LED needs. Figure 11 shows the LED designed circuit and the schematic used for the second PCB.

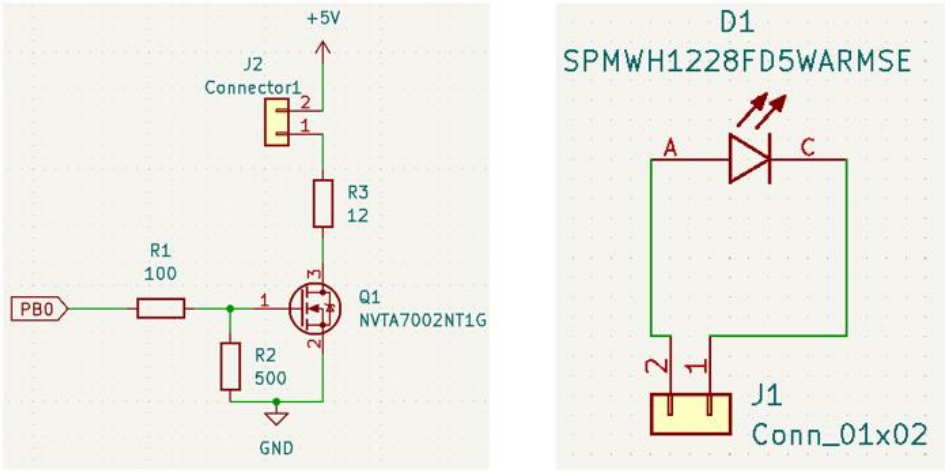


Figure 11. LED Circuit Schematic PCB1 / PCB2

The circuit designed for the LED is practically the same as the one used for the buzzer. It consists of a voltage divider, an NMOS transistor and unlike the buzzer circuit, it has a resistor and a connector since the LED is soldered on an external PCB.

The MOSFET works as a switch and allows to provide the current that the LED needs. Depending on the output voltage of the microcontroller on its pin, the gate voltage of the MOSFET will be high enough to activate it, otherwise no current would flow through the buzzer. This switch consists of an NMOS transistor that is alternately driven between the triode and cutoff regions.

The transistor is in triode mode if the gate voltage is higher than the threshold voltage (Vgs > Vth) and if the drain voltage is lower than the overdrive voltage (Vds < Vgs – Vth). On the other hand, the transistor is in cutoff region if the gate voltage is lower than the threshold voltage (Vgs < Vth). Since we want the transistor to operate in the triode region, we hace used a 12 Ω resistor to reduce the drain voltage and operate in this region. When the microcontroller outputs 3.3 V on its pin the MOSFET operates in triode mode and follows the following calculations:

$$Vgs = V12 = 3.3 * \frac{500}{600} = 2.75\ V \qquad Id = 150\ mA$$

$$Io = \frac{3.3}{600} = 5.5\ mA < Imax = 20\ mA \qquad Vds = 5 - 3 - 0.15 * 12 = 0.2\ V$$

Given these drain and gate voltages and a threshold voltage of 1 V we can be sure that the transistor is in the triode region. Figure 12 is a graph taken from the component datasheet showing the region in which the transistor is operating and shows that the circuit provides the LED the 150 mA it needs [8].

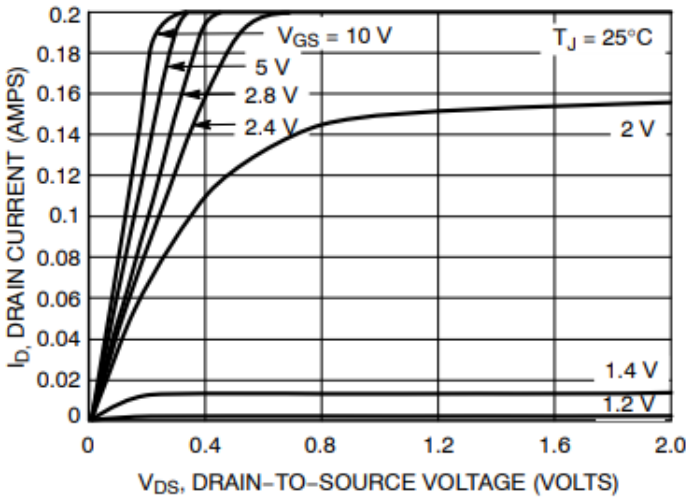


Figure 12. LED MOSFET (Id vs. Vds)

5.4.2 CAMERA

For the camera subsystem we have used the ArduCam 2MP Plus. This camera was chosen because of the microcontroller used for the final design. The camera doesn't require complicated communication protocols and was able to communicate with the microcontroller via I2C and SPI [9].

I2C was used to communicate with the physical properties of the camera like image brightness, saturation, and exposure while SPI was used to change photo resolution and data acquisition. With the use of SPI, the microcontroller is able to use a relatively simple communication protocol in order to send and receive image data. In addition to simple communication, this camera also handled a lot of the image processing that would take up valuable resources on the main microcontroller of device. Figure 13 shows the picture of the component used ArduCam 2MP Plus. It shows the image processor chip in the component and the camera lens.

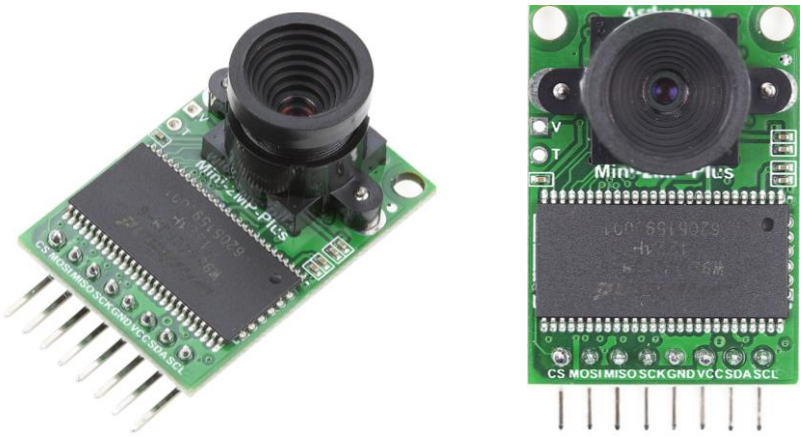


Figure 13. Arducam 2MP Plus

5.5 MICROPROCESSOR SUBSYSTEM

This block takes care of both controlling the individual elements like the LED and the buzzer but also performs most of the image analysis. This system will have a CNN (Convolutional Neural Network) loaded onto its flash memory while also handling the image buffer loaded from the camera. On the microcontroller that we will be using in this device we have 512Kb of flash as well as 148kb of ram [10].

Figure 14 shows the schematics of the different ports of the STMEGA microcontroller, the pin assignments of the 02x05 Odd_Even connector used to program the microcontroller, and two 10-pin connectors that can output some pins of the microcontroller as well as ground and the 3.3 V and 5 V voltages.

These external connections allow to connect the camera to the microcontroller, the power, and ground; to have access to other pins of the microcontroller in case any connection of the PCB fails; and it also allows to verify that each subsystem works correctly individually without the need to connect the microcontroller by doing external connections.

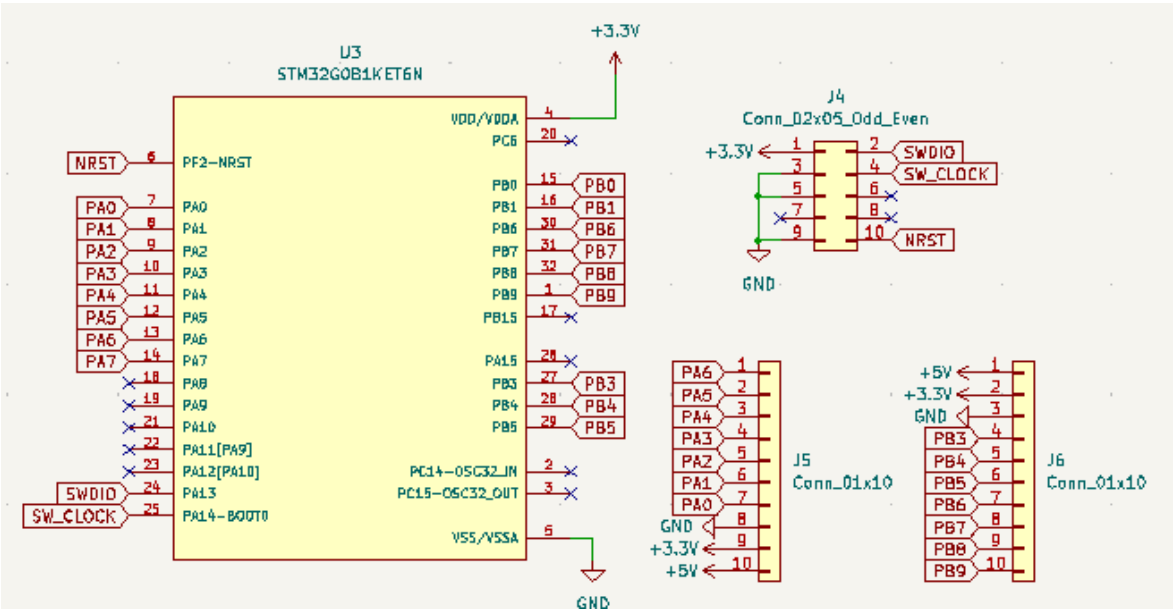


Figure 14. Microcontroller / Programming / External Connectors Schematic

In order to program the microcontroller, we had to use either JTAG or SWD. JTAG is a programming interface that interacts with several programmable hardware components like FPGAs and microcontrollers. The only problem with JTAG is that it uses a lot of pins, and it would use too many on the actual microcontroller. Moreover, if you use a larger number of pins, it takes up more space on the PCB and therefore the PCB size increases. Since we want the device to be as small as possible and if it takes up more microcontroller pins it leaves us with fewer pins to operate with, we have used SWD.

SWD or Serial Wire Debug is a programming protocol that uses 10 pins instead of 20 which allows us to save space. Using this 10-pin connection we just had to follow the pinout of SWD and make sure that each of the pins were used. Figure 15 shows the pin assignment for the 02x05 Odd_Even connector shown in used to program the microcontroller.

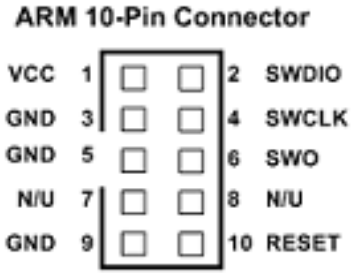


Figure 15. Pinout of SWD [11]

In addition to this we used the ST-Link which is a proprietary debugger/programmer that connects the computer to the actual microcontroller. In order to get the PCB to connect to this component we needed a 20 pin to 10 pin converter. This converter turns the ST-LINK from a 20 pin product to a 10 pin product which again allows for better space utilization on the PCB while also allowing for less pins of the actual microcontroller to be used.

The actual code was generated using STCUBEIDE which is a closed source IDE. The program uses the type of microcontroller being used and creates and sets up a development environment so that the programmer can focus on programming and not trying to set up the actual registers and communication protocols.

5.6 PCB DESIGN

The device has two PCBs, a main PCB containing most of the components and circuits, and a second PCB to solder the LED. Sections 5.6.1 and 5.6.2 show the schematics and PCB layout of the PCBs used.

To create the PCBs, we have used KICAD, a design tool for creating electrical schematics and board layouts. The process to create a PCB has three steps: add components and define the electrical connections of our design in a schematic, assign packages/footprints to the components in the schematic and define PCB dimensions, component locations, and copper traces in a layout.

5.6.1 MAIN PCB

The main PCB has almost all the components and connections of our device. It consists of the Power Subsystem, User Interface Subsystem, Microprocessor Subsystem, and the LED circuit. All the components used are surface mount and soldered to this PCB except for the battery, the button, the switch, the LED, and the camera. The components are connected to the main PCB by wires attached to their connectors. Figure 16 shows the schematics of the main PCB.

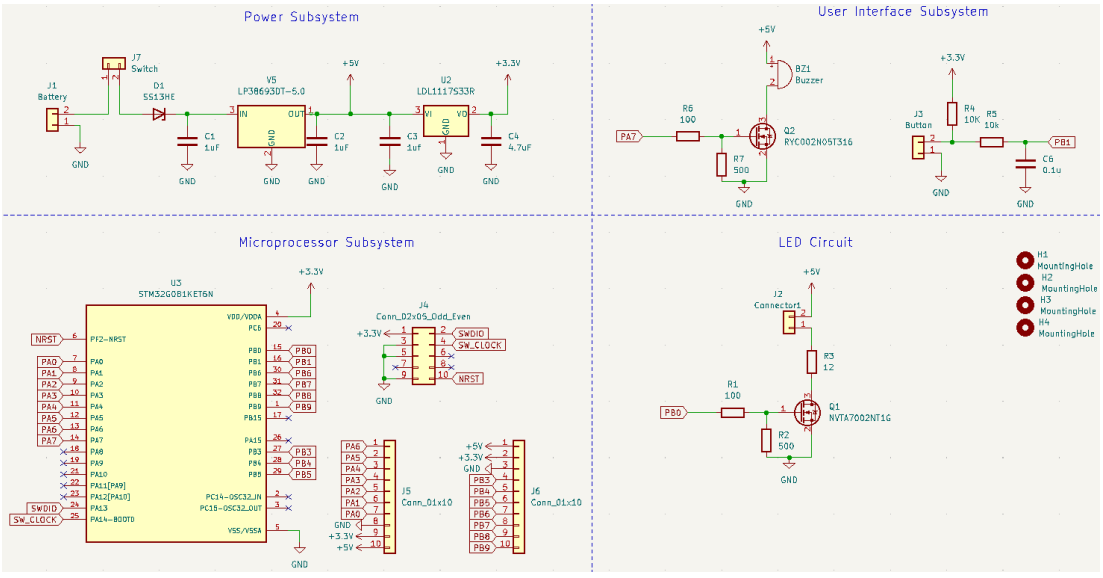


Figure 16. Main PCB Schematics

Once the circuits of our device have been designed and their schematics have been created, we made the component association. Each component has its footprint associated to it in order to be able to solder it correctly. After the assignment of the footprints, we made the PCB layout. The footprints of the components are sorted trying to keep traces short. Figure 17 shows the final design of the main PCB.

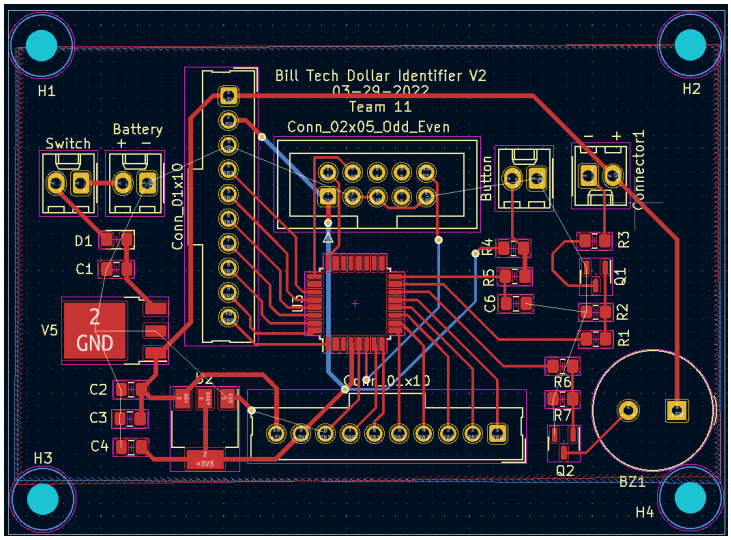


Figure 17. Main PCB Layout

5.6.2 LED PCB

As we need a good illumination of the bill to take a good quality photo, we have used a high-power LED. We have used a surface mount LED, so this second PCB has been designed to be able to solder it. The LED PCB is a simple PCB with only the LED and a connector that allows connecting it with wires to the LED Circuit of the main PCB. Figure 18 shows the schematic made.

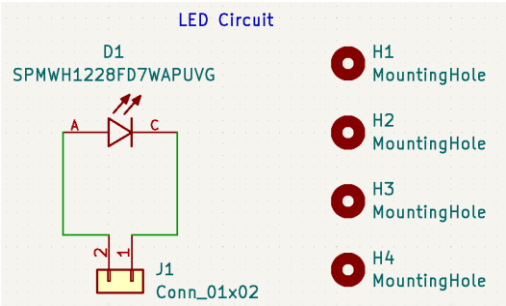


Figure 18. LED PCB Schematics

Once the simple LED circuit schematic was done, we made the component association. We assigned the LED and the connector to their footprints in order to be able to solder it correctly. After the assignment of the footprints, we made the PCB layout. Figure 19 shows the final design of the LED PCB.

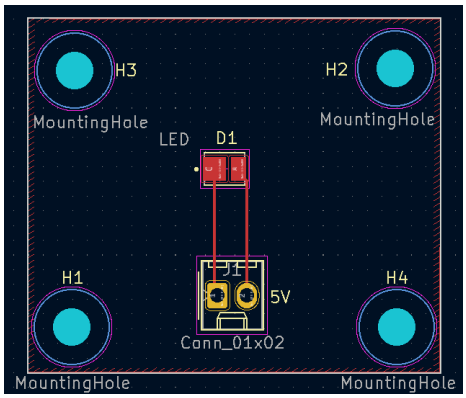


Figure 19. LED PCB Layout

5.7 PHYSICAL DESIGN

Designing the physical device, we took into account the needs and habits of the target users, those with visual impairment. We thought of an easy-to-use device, only using a button and a power switch as inputs from the user, allowing the user to easily become familiar with the controls. The switch to turn the device on and off, and the button to start the process of taking the photo and outputting the value of the bill.

In addition, we considered the convenience and simplicity when depositing the bill on the tray. For this purpose, the camera was placed at a sufficient height so that it does not disturb the user when depositing the bill on the tray. On the other hand, as the device is aimed to help those who are visually impaired, there is the possibility of being in a room with poor lighting or, in the worst case, complete darkness. To solve this problem, we decided to use an LED with enough power to illuminate the bill and take a good quality photo. Figure 20 shows a first sketch of the idea we had of the physical design:

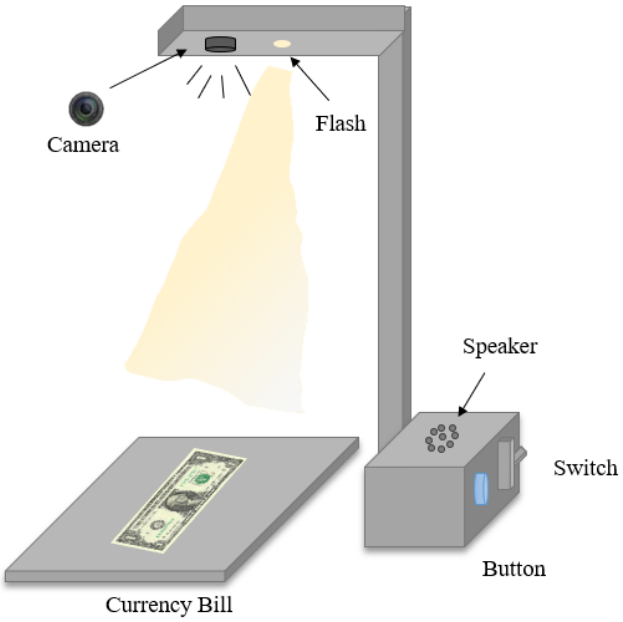


Figure 20. Bill Tech Dollar Identifier Visual Aid

The design shown in Figure 20 shows that device has a box that encloses the PCB with all the connections, the switch to turn the device on and off, the button that sends the command to take a picture and the buzzer that uses different sequences of beeps to disclose the value of the bill. In addition, it has an arm and at its end is the camera and flash. The bill should be placed on a white surface to make its reading easier.

After thinking more carefully about the design we made some changes to the first sketch. Instead of placing the LED next to the camera we decided to place it on the side. As we have used a high-power LED if it was placed next to the camera it could damage the image and prevent us from taking a good picture. By placing it on the side we avoided this problem.

On the other hand, we designed a system that allows adjusting the height of the camera, located inside the upper black case, and that allows adjusting the height of the LED. Thanks to this system, despite having some approximate theoretical calculations, the heights can be modified and placed where the best photo is taken, and the model works better.

In addition, a plastic corner has been added on top of the white tray. This plastic corner allows the user to know in which position to place the bill and helps to build the machine learning model. By limiting the positions in which the bill can be placed, the training of the model is reduced. Figure 21 shows the CAD design of the device. This design was sent to the machine shop for the fabrication of the device.

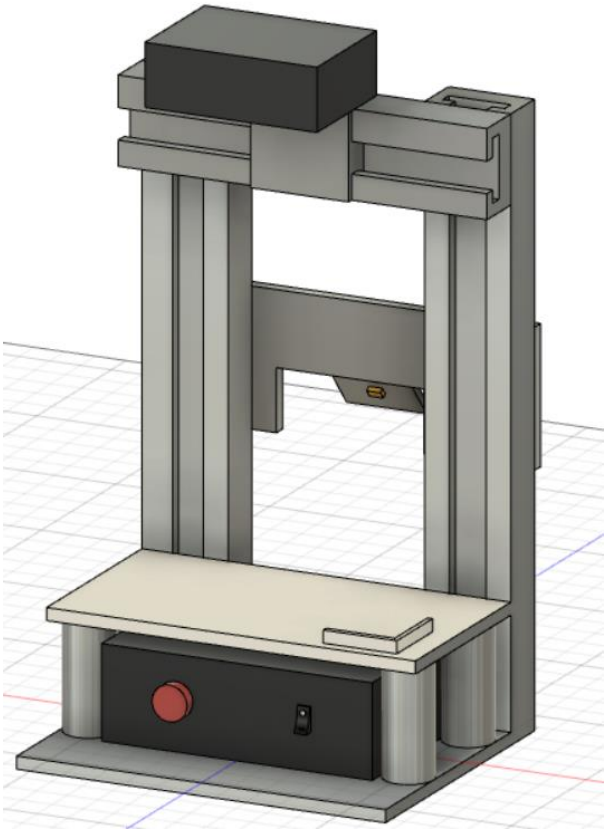


Figure 21. CAD Final Physical Design Sketch

Finally, Figure 22 displays the final physical design of our device. The device is very similar to the CAD design sent to the machine shop. Only the mechanism used to hold the LED changes.

The final design consists of a mechanism for adjusting the height and position of the camera located in a black case on top; a white tray for less background noise to the camera and ultimately to the Convolutional Neural Network (CNN); a black box under the white tray that encloses the main PCB and a LED located on the side that can also be adjusted in height.

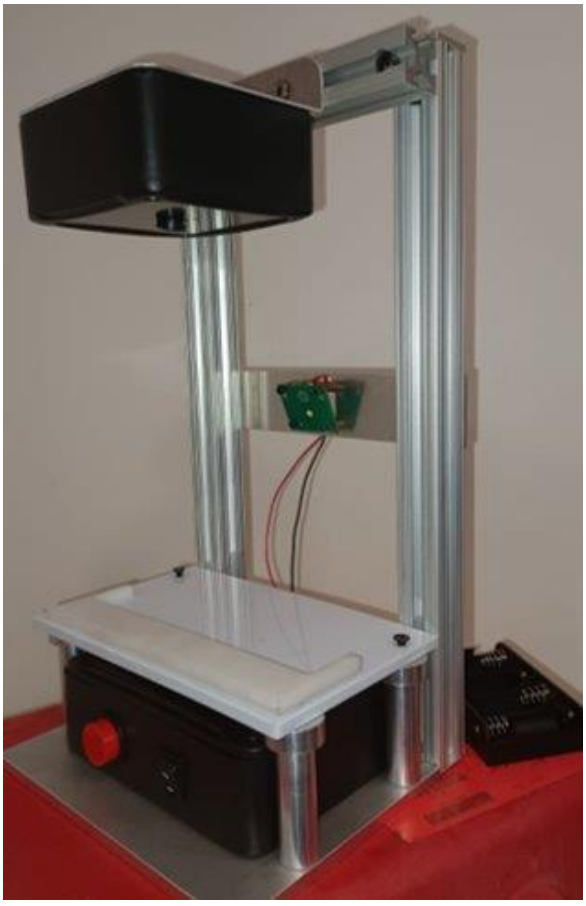


Figure 22. Bill Tech Dollar Identifier Final Model

5.8 IMAGE CLASSIFICATION MODEL

For classifying the bills on the platform, we used a Convolutional Neural Network (CNN) as our image classification model [12]. We gathered hundreds of training images for our dataset settled on implementing two architectures, ResNet50 and MobileNetv2, using transfer learning, to train the network.

5.8.1 DATASET COLLECTION

In order to begin training on the convolutional neural network, we acquired a dataset with the required information needed. The goal of this training dataset is to provide the neural network with reference images that replicate the environment of the actual device. These reference images are labeled correspondingly to their respective classes and allow the network to learn and distinguish patterns between these images. Our dataset images were taken using an APS-C crop sensor camera with a 35 mm lens 24 inches above standard letter paper as the platform on which the bills were placed upon. Around 40 images were taken for each side of the bill placed in different orientations on the surface. In total, over 80 images were obtained for each of the 5 classes. Figure 23 shows how the images were collected.

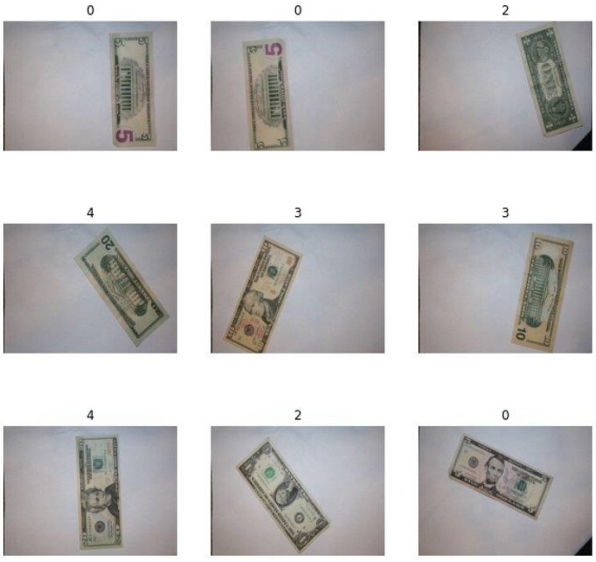


Figure 23. Example Dataset Images with Labels

5.8.2 DATASET AUGMENTATION

For a convolutional neural network to be effective, hundreds if not thousands of training images are required. Even though we obtained hundreds of images by hand, the network needs more images in order to produce our desired accuracy goal [4]. In order to achieve this, we used a technique called data augmentation as seen in Figure 24.

Each image from the training dataset was randomly rotated, flipped, and transposed in order to create hundreds of additional images for each hand collected sample. This allowed the training dataset to amass over a thousand images in total.

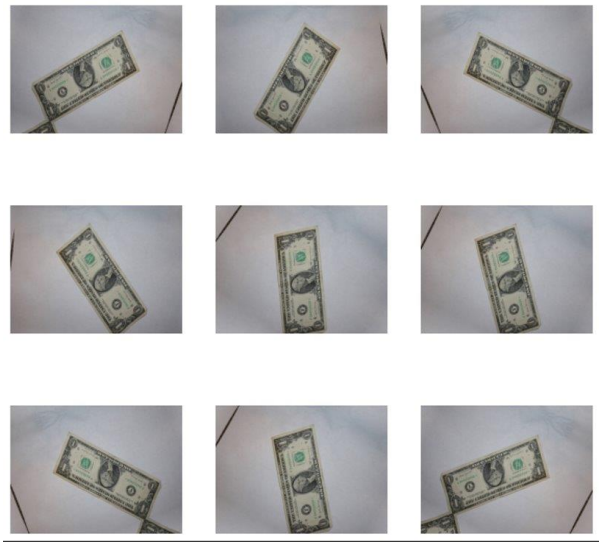


Figure 24. Example Data Augmentation on a Single Image

5.8.3 MODEL ARCHITECTURE

Our first attempts on using the LeNet and AlexNet architectures resulted in low validation accuracy. Switching to the more complex ResNet50 allowed us to obtain a validation accuracy reading of 99.9%. However, with the packaged model requiring over 2.37 gigabytes of space, it was impossible to load onto our microprocessor. Even after quantization, the process of compressing the model using smaller data structures, the model still required almost 100 megabytes of storage. We then tried to use the MobileNetV2 architecture to try to downsize the model to fit onto the microprocessor.

The advantage of the MobileNetV2 model is very high compression but suffers from worse accuracy. After quantization, we were able to get the model trained on MobileNetv2 to 2.53 megabytes. Even though this presented a huge improvement over the initial two gigabytes, it was still too large to fit onto our microprocessor.

CHAPTER 6. RESULTS ANALYSIS

Chapter 6 covers the procedures performed to check that the components and the model meet the requirements. Section 6.1, 6.2, 6.3, and 6.4 covers requirements and verification of the components used for the device. Section 6.5 covers the results obtained of the Image Classification Model.

6.1 POWER SUBSYSTEM

Requirement	Verification	Verified Status
Battery 1. The battery must supply power to the device for more than one hour.	1. Verification Process for Item 1: (a) Use the maximum current that flows through the circuit when its working. Leave the battery discharging at this maximum current and measure its voltage after one hour of use.	Yes
2. The battery must provide a minimum voltage of 5.5 V so the voltage regulator can output a fixed voltage of 5 V.	2. Verification Process for Item 2: (a) Measure the input voltage of the voltage regulator using a multimeter to ensure the voltage is higher than 5.5 V.	Yes
Schottky Diode 1. Protect the circuit if someone connects the battery in opposite polarity.	1. Verification Process for Item 1: (a) Measure the voltage across the diode and the voltage across the load when the battery is connected in opposite polarity. Calculate the reverse current flowing across the diode. It should be small.	Yes
Voltage Regulators 1. The voltage regulators must be able to drop the voltage from 6 V to 5 V and from 5 V to 3.3 V, obtaining an output voltage with a tolerance of 0.5%.	2. Verification Process for Item 1: (a) Measure the voltage after the voltage regulator using a voltmeter to ensure the voltage is in the range of 5 V +/- 0.5 %. (b) Measure the voltage after the voltage regulator using a voltmeter to ensure the voltage is in the range of 3.3 V +/- 0.5 %.	Yes

Table 4. Requirements and Verification Table for Power Subsystem

6.1.1 BATTERY

As one of our requirements is that the battery must power the device for more than one hour, we have used a battery that can supply the maximum current needed by the circuit continuously for more than one hour. We have used 4 1.5 V C-type alkaline batteries connected in series, supplying 6 V to the circuit.

Batteries discharge as they supply power to a circuit, therefore, their voltage decreases. The voltage regulator that drops the voltage from 6 V to 5 V needs an input voltage higher than 5.5 V to be able to provide the 5 V fixed voltage. As we are using 4 batteries of 1.5 V, the minimum voltage of each battery for the circuit to work is 1.375 V.

Our device has several processes: turning on the LED, taking a picture, running the model and outputting the bill value through the buzzer. These processes work in series, so they do not work at the same time. The process through which the highest current flows is the LED, which has a current of 150 mA.

To check that the battery could supply power to the circuit for more than one hour without its voltage going below 5.5 V, we tested it in the worst possible situation. As the maximum current flowing through the circuit is about 150 mA, we left the LED on for one hour.

As our device is not going to be on doing the process continuously and this current is higher than the average current that would flow when the device is on, it is enough to check that it would last more than one hour. After one hour we obtained that the voltage was approximately 5.7 V, therefore, the battery supply power for that time.

Figure 25 is a graph from the battery datasheet [13] that shows how the battery voltage changes over time depending on the current used. It can be seen that even if the current is greater than the maximum current 150 mA this one-hour specification would still be met.

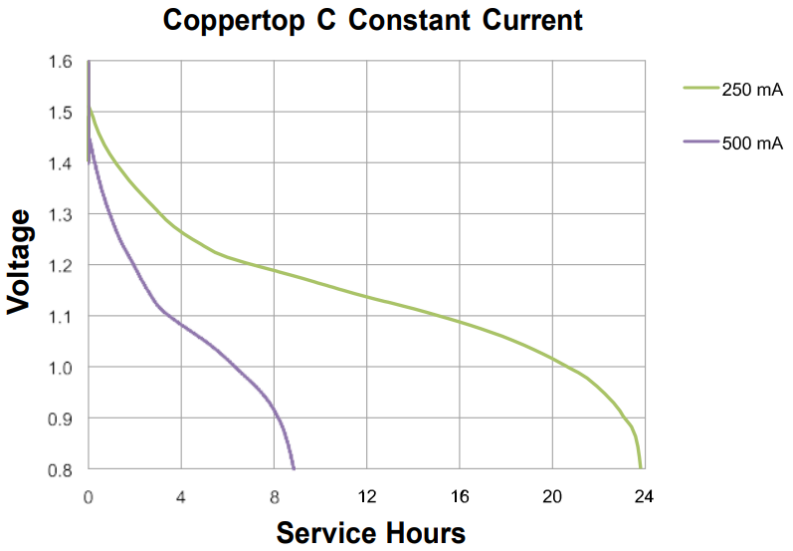


Figure 25. Voltage - Service Hours Graph

6.1.2 SCHOTTKY DIODE

To check that the schottky diode performs its function, that is protecting the circuit in case the user connects the battery in reverse polarity, we have changed the polarity of the battery and measured with a multimeter the voltage across the diode and the voltage across the load. We got a reverse current of 47 µA, therefore, the components are protected in case the user connects the battery in reverse polarity.

6.1.3 VOLTAGE REGULATORS

The voltage regulators should provide an output of 5 V and 3.3 V with a tolerance of 0.5%. To check compliance, we soldered the voltage regulators in the circuit and using a multimeter we measured their output voltage. In order to obtain the 5 V, the first LDO must have an input voltage higher than 5.5 V. Therefore, the voltage supplied by the battery and the voltage across the diode are very important. We obtained that the first voltage regulator had an output voltage of 5.0103 V and the second one had an output voltage of 3.2993 V. Therefore, both are in the desired voltage range.

6.2 USER INTERFACE SUBSYSTEM

Requirement	Verification	Verified Status
Button 1. Button should be debounced allowing for smooth transition between analog and digital signals.	1. Verification Process for Item 1: (a) Attach our mechanical switch to the debounce circuit. (b) With the circuit powered on, look at the trace once the switch is turned on and off. (c) The image on the oscilloscope should be smooth without any very rampant voltage jumps.	Yes
Buzzer 1. The buzzer should be able to produce at a minimum 7 different beep sequences as to distinguish each US bill denomination.	1. Verification Process for Item 1: (a) Attach a 10 Ω resistor in series with the speaker. (b) Set the voltage to 5 V and put 3.3 V in the microcontroller pin. See if the buzzer beeps. (c) Introduce the code to the microcontroller and prove the 7 different beep sequences	Yes
2. The buzzer should exceed 60 dB in volume	2. Verification Process Item 2: (a) Hear if the speaker can be comfortably heard in a quiet room. (b) This step can be done during step 1 C.	Yes

Table 5. Requirements and Verification Table for User Interface Subsystem

6.2.1 BUTTON

To check that the button works properly, and it is debounced allowing for smooth transitions we have used a multimeter and an oscilloscope. On the one hand, we soldered the button circuit on the PCB and once connected we measured the voltage it provides to the microcontroller pin. We obtained that if it was not pressed it had a voltage of about 3.3 V within the tolerance of 0.5 % and when the button was pressed the voltage value decreased gradually to almost 0 V.

On the other hand, we used an oscilloscope to see if the debounce system works properly. We connected channel 1 of the oscilloscope to the pin of the microcontroller and we could

verify that the system allows smooth transitions and therefore, we do not have the problem that state changes with the transitions.

6.2.2 BUZZER

The buzzer is in responsible for reproducing different sequences of beeps depending on the value of the bill. The decibels provided by the buzzer are very important. An intermediate range of decibels that allows to hear the beeps easily and is not harmful is between 50 and 60 db [14].

To check that the buzzer works we make external connections before soldering the microcontroller. Once we checked that the volume of the beeps was adequate, we soldered the microcontroller and introduced the code containing the sequences of each bill. The bills that can be identified by the model and therefore, the buzzer can output their value are:

$1 bill → 1 beep
$5 bill → 2 beeps
$10 bill → 3 beeps
$50 bill → 4 beeps
$100 bill → 5 beeps

6.3 CAMERA SUBSYSTEM

Requirement	Verification	Verified Status
Camera 1. Camera must be mounted high enough as to not hinder loading and unloading the bill by the user but close enough to capture a high-resolution image of the bill, at least 0.3 MP.	1. Verification Process for Item 1: (a) The camera must be of sufficient quality to be able to read the value of the bill at a minimum height of 25 cm. (b) Test the model at different heights. (c) Choose the highest height that doesn't negatively affect the accuracy of the model.	No
LED 1. LED must be able to provide 50 Lumens in brightness to still achieve 30 Lm/m2 on the surface of the tray.	1. Verification Process for Item 1: (a) Use a phone app like Lux Light Meter to check light levels in a dark room.	Yes

Table 6. Requirements and Verification Table for Camera Subsystem

6.3.1 CAMERA

The main objective for the camera to work is to be able to communicate with the microcontroller. Through the microcontroller we must be able to take a picture and then pass it to the model so that we can identify the value of the bill.

The camera we chosen meets the requirements needed to be able to get a good picture, however, we were not able to test it. In the process of identifying the value of the bill we have been able to communicate the microcontroller with the camera, but due to the memory problem we have not been able to process the images.

6.3.2 LED

The minimum luminance flux required to get a good illumination of the tray is 50 Lm. To provide those lumens we have used an LED with sufficient power to take a good picture of the bill. To check that the LED is providing the required lumens we have used the Lux Light Meter application. We obtained that the lumens seen in the tray were approximately 63 lumens, higher than the required minimum of 50 lumens.

On the other hand, we were able to verify that it is providing the required lumens by using the LED datasheet [15]. As can be seen in Figure 26, if 150 mA is supplied to the LED, it provides between 61 and 65 Lm, which is high enough to obtain a good image.

a) Luminous Flux Bins (I_F = 150 mA, T_S= 25°C)

CRI (R_a) Min	Nominal CCT (K)	Product Code	Flux Bin	Flux Range (Φ, lm)
90	2700	SPMWH1228FD7WAW☆VG	VG	56.5 ~ 60.5
	3000	SPMWH1228FD7WAV☆VG	VG	58.0 ~ 62.0
	3500	SPMWH1228FD7WAU☆VG	VG	59.0 ~ 63.0
	4000	SPMWH1228FD7WAT☆VG	VG	61.0 ~ 65.0
	5000	SPMWH1228FD7WAR☆VG	VG	62.0 ~ 66.0
	5700	SPMWH1228FD7WAQ☆VG	VG	61.5 ~ 65.5
	6500	SPMWH1228FD7WAP☆VG	VG	61.0 ~ 65.0

Figure 26. LED Luminous Flux

To check the current flowing through the LED, we have measured with a multimeter the real value of the resistance R3 shown in Figure 11, and the value of its voltage. We obtained that the current I = V/R is 147 mA. Therefore, the actual illumination that we are providing to the bill should be sufficient.

6.4 MICROPROCESSOR SUBSYSTEM

Requirement	Verification	Verified Status
Microprocessor 1. We must have access to the microcontroller, and we must be able to program it.	1. Verification Process for Item 1: (a) Check the connections using a multimeter. (b) Connect it and program a simple code such as turning on the LED to check that it flashes.	Yes
2. Microprocessor should be able to store the image classification model, the image buffer from the camera and the controller code to operate the system.	2. Verification Process for Item 2: (a) Evaluate size of neural net model as well as image size from camera. (b) Scale down both image size as well as model complexity. (c) While scaling down both parameters adjust for accuracy until 0.95 accuracy is attained while also being able to fit on the microprocessor.	No

Table 7. Requirements and Verification Table for Microprocessor

To verify that the microprocessor is working properly we must evaluate that the connections are correct, that we have access to it, we are able to program it, and that we have enough memory for storing the image classification model, storing the image buffer from the camera, and storing the controller code to operate the system.

To check that the connections were good we used a multimeter. We also checked that we were able to program the microcontroller by programming a simple code to turn the LED and buzzer on and off.

However, when we got the size of the image buffer from the camera, the size of the image classification model and the size of the code to operate the system we found that the

memory of the microcontroller was not enough. The microprocessor that we integrated into our system only had a flash memory capacity of 512 kilobytes.

The microprocessor not only has these storage tasks, but the flash memory would also potentially need to serve as a buffer for the RAM while the processor is performing calculations. The reason is because the microprocessor only had 148 Kilobytes of RAM. Therefore, due to some memory problems we were not able to test the full performance of the device.

6.5 IMAGE CLASSIFICATION MODEL

To test and verify that the image classification model works correctly we created a validation data set. The validation data set was created by randomly subsampling the initial training data set. In order not to obtain an erroneous result, these images were set aside and not used during the training process. After each iteration of the training process, the validation data set was run through the model in order to assess the accuracy of the network. The validation accuracy is shown in the red line in Figure 27 and Figure 28. After training the ResNet50 and MobileNetV2 models, we were able to measure a validation accuracy of approximately 99.9% in the ResNet50 model and approximately 80% in the MobileNetV2 model.

As mentioned above MobileNetV2 allows a very high compression so if there are tight space issues in the project it is a good choice having 80% accuracy. However, in our case the accuracy of the model is very important. If the device misclassifies the user's bill, e.g., the user deposits a $50 bill and the device outputs $5, instead of helping the user it makes the situation worse. Therefore, in our case we have decided to use the ResNet50 architecture. Figure 27 and Figure 28 shows the accuracy of the ResNet50 and MobileNet50 models respectively.

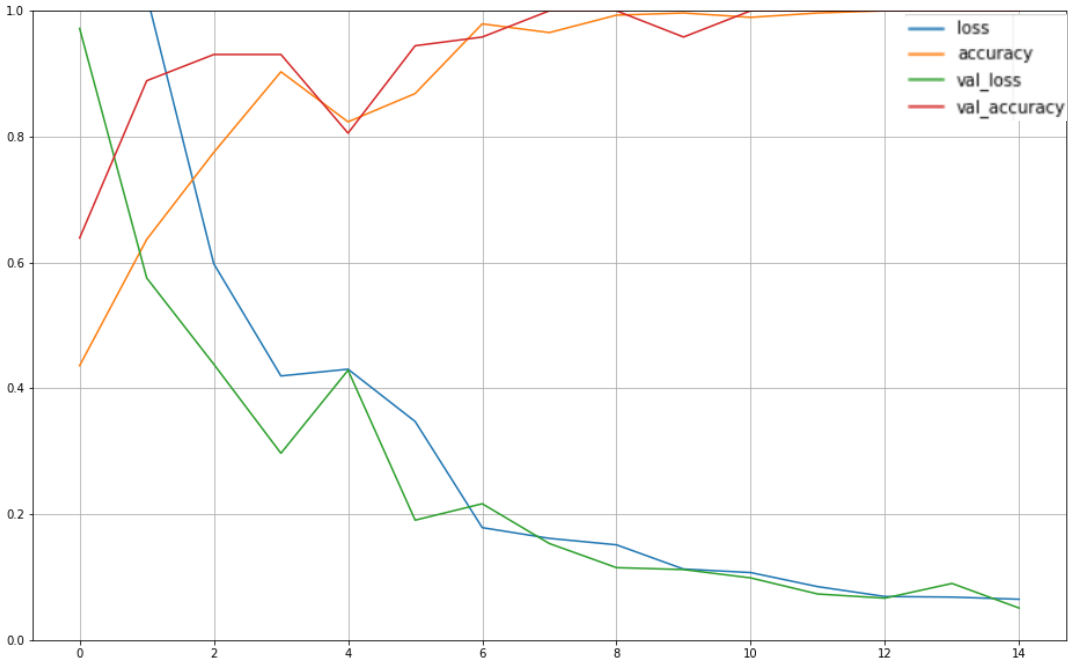


Figure 27. ResNet50 Evaluation Graph

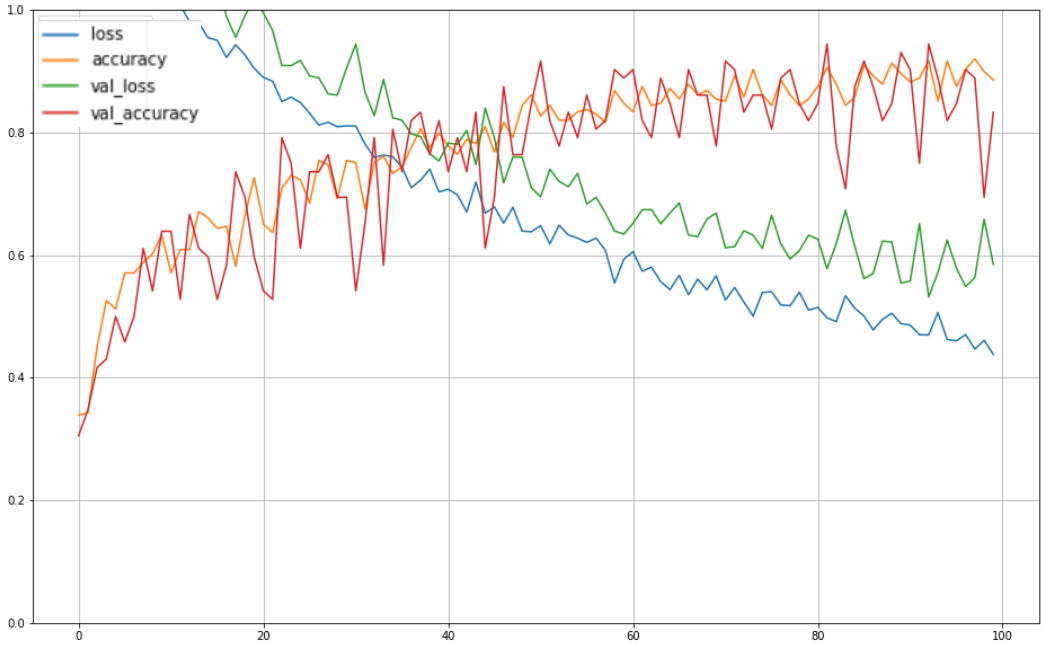


Figure 28. MobileNetV2 Evaluation Graph

CHAPTER 7. CONCLUSION AND FUTURE WORK

7.1 ACCOMPLISHMENTS

The main achievements that the project had to accomplish are those mentioned in section 4.1 Objectives: The device should be able to run independently on battery power for at least 1 hour, the LED should illuminate the banknote with the appropriate brightness so that a good picture can be taken, the machine should be able to correctly identify the bill presented on the white tray with an accuracy rate of 0.90 and an error rate of 0.03, and the machine should produce differentiable audible responses upon identification of different bill types.

At the end of the day, the group has managed to meet these four objectives. The device has a completely validated and working PCB where the power supply, the buzzer, the button, the LED, and the microcontroller meet the requirements needed and work correctly; furthermore, we are able to program and execute code from the microcontroller; and we were able to develop a CNN with 99.9% validation accuracy.

However, the device is not fully functional. Each part works separately, but as we have discussed before, some problems with this device revolved around the miss use of memory and being unable to get these blocks to combine.

7.2 UNCERTAINTIES

The main reason why our final project wasn't functional was due solely to the lack of flash memory onboard the microprocessor. The microprocessor that we integrated into our system only had a flash memory capacity of 512 kilobytes. This memory is meant to be shared between storing the image classification model, storing the image buffer from the camera, and storing the controller code to operate the system. In addition to these storage tasks, the flash memory would also potentially need to serve as a buffer for the RAM while the processor is performing calculations. This is because the microprocessor only had 148 kilobytes of RAM. Without an additional SRAM chip or storage interface, there was not enough capacity built into the microprocessor to contain all the necessary code required for the full functionality of the device.

7.3 ETHICAL CONSIDERATIONS

The main ethical concerns for this project come from taking pictures of legal currency. The United States government allows for individuals to take pictures of US currency if these pictures are either less than 75% of the original size or greater than 150% of the original size [16]. These pictures also require deletion after use. During the training and designing of this device, the pictures of the bills are never saved for very long. As soon as the images are used for training the model they are no longer needed and are deleted. Because of this practice, the development of the device followed all legal and regulatory standards provided by the United States [16].

In terms of safety concerns this device is operating with both low voltage and current levels which do not pose a risk to any of the group members. LEDs will not be of a strong enough luminance to pose a risk to vision and speakers will not produce a loud enough sound to cause hearing loss. In order to uphold safety standards, members of the group have complied with all safety precautions in the senior design lab and have been carefully following instructions on proper etiquette and technique when soldering components.

We have followed all codes of ethics from IEEE [17] by both upholding moral and ethical standards among ourselves and others, continually considering ethical solutions to problems that do not introduce conflicts of interest.

7.4 FUTURE WORK

As we have discussed before, a lot of the problems with this device revolved around the miss use of memory and being unable to get these blocks to combine. In order to obtain a fully functional final design we have come up with 2 ideas, one that would involve testing the design we made using a computer, and the other that would consist of making a small modification to the device to increase its memory.

The first option to get these components to combine would be to get the microcontroller to take a picture. Send the image data to a computer and run the model. Once the model is done running send the data back to the microcontroller. The microcontroller is then able to output the identity of the bill in a series of short beeps.

On the other hand, by leveraging the fact that all the sub systems work separately, we could use a distributed approach to achieve final functionality. Another approach would be to integrate some form of additional storage onto our PCB for the microprocessor to interface with. This could be in the form of an SRAM chip or even a MicroSD expansion slot. By adding this additional storage, the microprocessor would be able to store and compute the model. If an SRAM chip or a MicroSD expansion chip is used, the main core would not change. Simply, it would be necessary to make some small modification to the PCB and change a part of the code. With these small changes the memory problem could be solved.

CHAPTER 8. REFERENCES

[1] "How Do People Who Are Blind or Visually Impaired Identify Money?"

https://chicagolighthouse.org/sandys-view/identifying-money/ (accessed 05/02/2022)

[2] "Modify Dollar Bills to help blind people identify their value "
https://www.espaciologopedico.com/noticias/det/978/modificaran-dolares-para-que-puedan-ser-identificados-por-ciegos.html

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'12). Curran Associates Inc., Red Hook, NY, USA, 2012.

https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf (accessed 04/06/2022)

[4] Serdar Yegulalp. What is Tensor Flow?. June 3, 2022

https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explained.html (accessed 03/02/2022)

[5] Naciones Unidas. "Objetivos de Desarrollo Sostenible"

https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/ (accessed 05/20/2022)

[6] Digi-Key, "RYC002N05T316", Available.

https://www.digikey.com/en/products/detail/rohm-semiconductor/RYC002N05T316/6573141?utm_adgroup=Discrete%20Semiconductor%20Products&utm_source=google&utm_medium=cpc&utm_campaign=Shopping_Supplier_Rohm%20Semiconductor_0846_Co-op&utm_term=&utm_content=Discrete%20Semiconductor%20Products&gclid=Cj0KCQjw0PWRBhDKARIsAPKHFGgddTLXcqoI83gJt6Rt_qLqhGgyVZmGzY9nuud8lxf_5Ws44kyzcv4aAkvkEALw_wcB (accessed 04/22/2022)

[7] HACKADAY. "Debounce your noisy buttons"

https://hackaday.com/2015/12/09/embed-with-elliot-debounce-your-noisy-buttons-part-i/ (accessed 02/12/2022)

[8] Digi-Key. "NVTA7002NT1G", Available. https://www.digikey.com/en/products/detail/onsemi/NVTA7002NT1G/8538846?utm_adgroup=Discrete%20Semiconductor%20Products&utm_source=google&utm_medium=cpc&utm_campaign=Shopping_Supplier_onsemi&utm_term=&utm_content=Discrete%20Semiconductor%20Products&gclid=Cj0KCQjw0PWRBhDKARIsAPKHFGiBQZiQ7N7EAUUOkUsTYFkFUT6omQzm-t6J3LraP6gBQ2iULLlJ8xoaAjtfEALw_wcB (accessed 04/22/2022)

[9] Arducam. "ArduCAM-M-2MP Camera Shield",

https://www.arducam.com/downloads/shields/ArduCAM_Mini_2MP_Camera_Shield_Hardware_Application_Note.pdf (accessed 03/15/2022)

[10] Digi-Key, "STMEGA32", Not Available.

[11] arm Developer. SWD pinout picture. Available from: https://developer.arm.com/documentation/101455/0100/Hardware-Description/Target-Connectors (accessed 05/23/2022)

[12] LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner P. "Gradient-based learning applied to document recognition." Proc. of the IEEE, 86 (11). Dec. 1998, pp. 2278-2324. Available from: http://yann.lecun.com/exdb/publis/pdf/lecun-98.pdf

[13] Digi-Key, "C-MN1400", Available.

https://www.digikey.com/en/products/detail/duracell-industrial-operations-inc/C-MN1400/13280361 (accessed 03/20/2022)

[14] Digi-Key, "IE092505-1", Available.

https://www.digikey.com/en/products/detail/db-unlimited/IE092505-1/9990486 (accessed 04/22/2022)

[15] Samsung. "SPMWH1228FD7WAP LED", Available.

https://cdn.samsung.com/led/file/resource/2020/07/Data_Sheet_LM281B_Plus_Pro_VG_Rev.0.8.pdf (accessed 03/23/2022)

[16] U.S. Currency Education Program. Permissible Color Illustrations of U.S. Currecncy.

https://www.uscurrency.gov/media/currency-image-use (accessed 05/03/2022)

[17] IEEE Policies, Section 7 - Professional Activities (Part A - IEEE Policies).

https://www.ieee.org/about/corporate/governance/p7-8.html (accessed 05/10/2022)

ANEXO I

```
import tensorflow as tf
import tensorflow.compat.v1 as tf1
tf.config.set_visible_devices([], 'GPU')

from tensorflow import keras
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import os
import cv2

import sklearn
# import randomfrom numpy import
from PIL import Image
# import theano
import pandas as pd
```

In [4]:

```
from tensorflow.python.client import device_lib
print(device_lib.list_local_devices())
print("Num GPUs Available: ", len(tf.config.list_physical_devices('GPU')))

config = tf.compat.v1.ConfigProto()
config.gpu_options.allow_growth = True
session = tf.compat.v1.Session(config=config)
```

In [5]:

```
path_train = "train_img"

IMG_SIZE = (768, 1024)
# IMG_SIZE = (480, 640)
# IMG_SIZE = (288, 352)

CATEGORIES = ["ONE", "TWO", "TEN", "TWENTY", "FIFTY", "HUNDRED"]

# load train dataset from folder
train_ds = tf.keras.preprocessing.image_dataset_from_directory(
    "dataset_img",
    # color_mode='grayscale',
    image_size=IMG_SIZE,
    validation_split=0.2,
    subset="training",
    seed=123
)

# load validation dataset
test_ds = tf.keras.preprocessing.image_dataset_from_directory(
    "dataset_img",
```

```
    # color_mode='grayscale',
    image_size=IMG_SIZE,
    validation_split=0.2,
    subset="validation",
    seed=123
)

plt.figure(figsize=(10, 10))
for images, labels in train_ds.take(1):
    for i in range(9):
        ax = plt.subplot(3, 3, i + 1)
        plt.imshow(images[i].numpy().astype("uint8"))
        plt.title(int(labels[i]))
        plt.axis("off")

data_augmentation = tf.keras.Sequential(
    [
        tf.keras.layers.experimental.preprocessing.RandomFlip('horizontal'),
        tf.keras.layers.experimental.preprocessing.RandomRotation(0.1),
    ]
)

plt.figure(figsize=(10, 10))
for images, _ in train_ds.take(1):
    for i in range(9):
        augmented_images = data_augmentation(images)
        ax = plt.subplot(3, 3, i + 1)
        plt.imshow(augmented_images[0].numpy().astype("uint8"))
        plt.axis("off")

# create the augmented train data set
augmented_train_ds = train_ds.map(
  lambda x, y: (data_augmentation(x, training=True), y))
```

```
Found 361 files belonging to 5 classes.
Using 289 files for training.
Found 361 files belonging to 5 classes.
Using 72 files for validation.
```

In [6]:

```
model = keras.models.Sequential()

pretrained_model= tf.keras.applications.ResNet50(include_top=False,
                   input_shape=(IMG_SIZE[0],IMG_SIZE[1],3),
                   pooling='avg',classes=5,
                   weights='imagenet')
for layer in pretrained_model.layers:
        layer.trainable=False

pretrained_model.trainable = False

pretrained_model.summary()
```

In [7]:

```
model.add(pretrained_model)

model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(512, activation='relu'))
model.add(tf.keras.layers.Dense(5, activation='softmax'))
```

In [8]:

```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 2048)              23587712

 flatten (Flatten)           (None, 2048)              0

 dense (Dense)               (None, 512)               1049088

 dense_1 (Dense)             (None, 5)                 2565

=================================================================
Total params: 24,639,365
Trainable params: 1,051,653
Non-trainable params: 23,587,712
```

In [9]:

```
# compiling the model
tf.keras.backend.clear_session()
model.compile(loss = "sparse_categorical_crossentropy",
optimizer = "adam",
metrics = ["accuracy"])
```

In [10]:

```
tf.keras.backend.clear_session()
history = model.fit(augmented_train_ds,
                    epochs = 15,
                    batch_size = 16,
                    validation_data = test_ds)
```

In [11]:

```
pd.DataFrame(history.history).plot(figsize = (16, 10))
plt.grid(True)
plt.gca().set_ylim(0, 1)
plt.show()
```

In [12]:

```
model.evaluate(test_ds)
model.save("completed_model")
```

In [13]:

```
# model_test = keras.models.load_model('completed_model')
# model_test.evaluate(test_ds)
```

In [14]:

```
# Convert the model.
# converter = tf.lite.TFLiteConverter.from_keras_model(model)
# tflite_model = converter.convert()

converter = tf.lite.TFLiteConverter.from_keras_model(model)
converter.optimizations = [tf.lite.Optimize.OPTIMIZE_FOR_SIZE]
# converter.representative_dataset = representative_dataset
# converter.target_spec.supported_types = [tf.float16]
tflite_quant_model = converter.convert()

# Save the model.
with open('model.tflite', 'wb') as f:
  f.write(tflite_quant_model)
```

```
/* Includes */
#include "main.h"

/* Variables */
CRC_HandleTypeDef hcrc;

I2C_HandleTypeDef hi2c3;

SPI_HandleTypeDef hspi1;
DMA_HandleTypeDef hdma_spi1_rx;

/* Function prototypes /

void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_SPI1_Init(void);
static void MX_DMA_Init(void);
static void MX_CRC_Init(void);
static void MX_I2C3_Init(void);

int main(void)
{
  HAL_Init();

  /* Configure the system clock */

  SystemClock_Config();


  /* Initialize all configured peripherals */

  MX_GPIO_Init();
  MX_SPI1_Init();
  MX_DMA_Init();
  MX_CRC_Init();
  MX_I2C3_Init();

  /* USER CODE BEGIN 2 */

  uint8_t vid, pid, temp;
  uint8_t Camera_WorkMode = 0;
  uint8_t start_shoot = 0;
  uint8_t stop = 0;

  /* USER CODE END 2 */
```

```
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
       if(HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_1) == GPIO_PIN_RESET)
       {
       for(uint8_t x = 1; x < 6; ++x){
              HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_SET);
              HAL_Delay(2000);
              HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_RESET);

               HAL_Delay(2000);
               for(uint8_t i = 0; i < x; ++i){
               HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_SET);

               HAL_Delay(1000);
               HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_RESET);
               HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_RESET);
               HAL_Delay(1000);
         }
       }

    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */

  }
  /* USER CODE END 3 */
}
}

/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Configure the main internal regulator output voltage */

  HAL_PWREx_ControlVoltageScaling(PWR_REGULATOR_VOLTAGE_SCALE1);

  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure. */

  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.HSIDiv = RCC_HSI_DIV1;
  RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSI;
```

```
  RCC_OscInitStruct.PLL.PLLM = RCC_PLLM_DIV1;
  RCC_OscInitStruct.PLL.PLLN = 8;
  RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
  RCC_OscInitStruct.PLL.PLLQ = RCC_PLLQ_DIV2;
  RCC_OscInitStruct.PLL.PLLR = RCC_PLLR_DIV2;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }
  /** Initializes the CPU, AHB and APB buses clocks */

  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_2) != HAL_OK)
  {
    Error_Handler();
  }
}

/**
  * @brief CRC Initialization Function
  * @param None
  * @retval None
  */

static void MX_CRC_Init(void)
{
  hcrc.Instance = CRC;
  hcrc.Init.DefaultPolynomialUse = DEFAULT_POLYNOMIAL_ENABLE;
  hcrc.Init.DefaultInitValueUse = DEFAULT_INIT_VALUE_ENABLE;
  hcrc.Init.InputDataInversionMode = CRC_INPUTDATA_INVERSION_NONE;
  hcrc.Init.OutputDataInversionMode = CRC_OUTPUTDATA_INVERSION_DISABLE;
  hcrc.InputDataFormat = CRC_INPUTDATA_FORMAT_BYTES;
  if (HAL_CRC_Init(&hcrc) != HAL_OK)
  {
    Error_Handler();
  }
}

/**
  * @brief I2C3 Initialization Function
  * @param None
  * @retval None
  */
```

```
static void MX_I2C3_Init(void)
{
  hi2c3.Instance = I2C3;
  hi2c3.Init.Timing = 0x10707DBC;
  hi2c3.Init.OwnAddress1 = 0;
  hi2c3.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;
  hi2c3.Init.DualAddressMode = I2C_DUALADDRESS_DISABLE;
  hi2c3.Init.OwnAddress2 = 0;
  hi2c3.Init.OwnAddress2Masks = I2C_OA2_NOMASK;
  hi2c3.Init.GeneralCallMode = I2C_GENERALCALL_DISABLE;
  hi2c3.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
  if (HAL_I2C_Init(&hi2c3) != HAL_OK)
  {
    Error_Handler();
  }

  /** Configure Analogue filter */

  if (HAL_I2CEx_ConfigAnalogFilter(&hi2c3, I2C_ANALOGFILTER_ENABLE) != HAL_OK)
  {
    Error_Handler();
  }

  /** Configure Digital filter */

  if (HAL_I2CEx_ConfigDigitalFilter(&hi2c3, 0) != HAL_OK)
  {
    Error_Handler();
  }
}

/**
  * @brief SPI1 Initialization Function
  * @param None
  * @retval None
  */

static void MX_SPI1_Init(void)
{
  /* SPI1 parameter configuration*/

  hspi1.Instance = SPI1;
  hspi1.Init.Mode = SPI_MODE_MASTER;
  hspi1.Init.Direction = SPI_DIRECTION_2LINES;
  hspi1.Init.DataSize = SPI_DATASIZE_8BIT;
  hspi1.Init.CLKPolarity = SPI_POLARITY_LOW;
  hspi1.Init.CLKPhase = SPI_PHASE_1EDGE;
  hspi1.Init.NSS = SPI_NSS_SOFT;
  hspi1.Init.BaudRatePrescaler = SPI_BAUDRATEPRESCALER_16;
  hspi1.Init.FirstBit = SPI_FIRSTBIT_MSB;
  hspi1.Init.TIMode = SPI_TIMODE_DISABLE;
  hspi1.Init.CRCCalculation = SPI_CRCCALCULATION_DISABLE;
```

```
  hspi1.Init.CRCPolynomial = 7;
  hspi1.Init.CRCLength = SPI_CRC_LENGTH_DATASIZE;
  hspi1.Init.NSSPMode = SPI_NSS_PULSE_ENABLE;
  if (HAL_SPI_Init(&hspi1) != HAL_OK)
  {
    Error_Handler();
  }
}

/** Enable DMA controller clock */

static void MX_DMA_Init(void)
{

  /* DMA controller clock enable */
  __HAL_RCC_DMA1_CLK_ENABLE();

  /* DMA interrupt init */
  /* DMA1_Channel1_IRQn interrupt configuration */
  HAL_NVIC_SetPriority(DMA1_Channel1_IRQn, 0, 0);
  HAL_NVIC_EnableIRQ(DMA1_Channel1_IRQn);

  /* DMA1_Ch4_7_DMA2_Ch1_5_DMAMUX1_OVR_IRQn interrupt configuration */
  HAL_NVIC_SetPriority(DMA1_Ch4_7_DMA2_Ch1_5_DMAMUX1_OVR_IRQn, 0, 0);
  HAL_NVIC_EnableIRQ(DMA1_Ch4_7_DMA2_Ch1_5_DMAMUX1_OVR_IRQn);

}

/**
  * @brief GPIO Initialization Function
  * @param None
  * @retval None
  */

static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOB_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(CHIP_SELECT_GPIO_Port, CHIP_SELECT_Pin, GPIO_PIN_RESET);

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_RESET);

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_RESET);

  /*Configure GPIO pin : CHIP_SELECT_Pin */
  GPIO_InitStruct.Pin = CHIP_SELECT_Pin;
```

```
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_MEDIUM;
  HAL_GPIO_Init(CHIP_SELECT_GPIO_Port, &GPIO_InitStruct);

  /*Configure GPIO pin : PA7 */
  GPIO_InitStruct.Pin = GPIO_PIN_7;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

  /*Configure GPIO pin : LED_Pin */
  GPIO_InitStruct.Pin = LED_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(LED_GPIO_Port, &GPIO_InitStruct);

  /*Configure GPIO pin : PB1 */
  GPIO_InitStruct.Pin = GPIO_PIN_1;
  GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);
}

/**
  * @brief  This function is executed in case of error occurrence.
  * @retval None
  */

void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state */
  __disable_irq();

  while (1)
  {
  }
}

#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
  */
```

```
void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
  /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```