Pontificia Universidad Católica del Ecuador

Sede Ibarra

ESCUELA DE INGENIERÍA

INFORME FINAL DEL PROYECTO

TEMA:

CONTROLADOR DE PROCESAMIENTO DE IMÁGENES BASADO EN PLATAFORMAS OPEN SOURCE DESTINADO A LA VISUALIZACIÓN EN PANELES DE LEDS RGB

PREVIO A LA OBTENCIÓN DEL TÍTULO DE

INGENIERO EN SISTEMAS

LÍNEAS DE INVESTIGACIÓN:

INGENIERÍA EN SOFTWARE Y EMPRENDIMIENTO EN TICS

AUTOR: CRISTIAN JAVIER CUASPA BENAVIDES

ASESOR: MGS. LUIS DAVID NARVAEZ ERAZO

IBARRA, ENERO - 2019

Ibarra, 21 de enero del 2019

Mgs. Luis David Narváez Erazo
ASESOR

CERTIFICA:

Haber revisado el presente informe final de investigación, el mismo que se ajusta a las normas vigentes en la Escuela de Ingeniería, de la Pontificia Universidad Católica del Ecuador Sede Ibarra (PUCESI); en consecuencia, autorizo su presentación para los fines legales pertinentes.

(f.) ……

Mgs. Luis David Narváez Erazo

C.C.: 1002868378

PÁGINA DE APROBACIÓN DEL TRIBUNAL

El jurado examinador, aprueba el presente informe de investigación en nombre de la Pontificia Universidad Católica del Ecuador Sede Ibarra (PUCESI):

(f): ……

Mgs. Darwin Marcelo Pillo Guanoluisa

C.C.: 100331966-0

(f): ……

Mgs. Stalin Marcelo Arciniegas Aguirre

C.C.: 100349681-5

(f): ……

Mgs. Luis David Narváez Erazo

C.C.: 100286837-8

ACTA DE CESIÓN DE DERECHOS

Yo Cristian Javier Cuaspa Benavides, declaro conocer y aceptar la disposición del Art. 165 del Código Orgánico de la Economía Social de los Conocimientos, Creatividad e Innovación, que manifiesta textualmente: "Se reconoce facultad de los autores y demás titulares de derechos de disponer de sus derechos o autorizar las utilizaciones de sus obras o prestaciones, a título gratuito u oneroso, según las condiciones que determinen. Esta facultad podrá ejercerse mediante licencias libres, abiertas y otros modelos alternativos de licenciamiento o la renuncia".

Ibarra, 21 de enero de 2019

f): ……

Cristian Javier Cuaspa Benavides

C.C.: 1004637896

AUTORÍA

Yo, Cristian Javier Cuaspa Benavides, portador de la cédula de ciudadanía N° 1004637896, declaro que la presente investigación es de total responsabilidad del (los) autor (es), y eximo expresamente a la Pontificia Universidad Católica del Ecuador Sede Ibarra de posibles reclamos o acciones legales.

f): ……

Cristian Javier Cuaspa Benavides

C.C.: 1004637896

Pontificia Universidad Católica del Ecuador
PUCE
COORDINACIÓN DE TITULACIÓN DE LA ESCUELA DE INGENIERÍA
Sede Ibarra

AGRADECIMIENTO

A todas las personas quienes me han brindado apoyo, y darme aliento para poder culminar el proyecto, mi más sincero agradecimiento.

A mis padres Marcelo Cuaspa y Lilia Benavides, quienes nunca dejaron de confiar en mis capacidades.

A mis hermanas Vanessa Cuaspa, Viviana Cuaspa y Fernanda Cuaspa quienes me han brindado el apoyo moral siempre que lo necesité.

A María Fernanda Pinto Cañizares, siendo una de mis mayores motivaciones fue el ingrediente perfecto para no darme por vencido y finalmente alcanzar mi objetivo.

A mi asesor Mgs. David Narváez, por brindarme el tiempo y apoyo necesario en el desarrollo de la tesis.

Cristian Javier Cuaspa Benavides

DECLARACIÓN Y AUTORIZACIÓN

Yo: Cristian Javier Cuaspa Benavides, con CC: 1004637896, autor del trabajo de grado intitulado: “Controlador de procesamiento de imágenes basado en plataformas open source destinado a la visualización en paneles de leds RGB”, previo a la obtención del título profesional de Ingeniería en Sistemas, en la Escuela de Ingeniería.

1.- Declaro tener pleno conocimiento de la obligación que tiene la Pontificia Universidad Católica del Ecuador Sede- Ibarra, de conformidad con el artículo 144 de la Ley Orgánica de Educación Superior de entregar a la SENESCYT en formato digital una copia del referido trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la Pontificia Universidad Católica del Ecuador Sede Ibarra a difundir a través de sitio web de la Biblioteca de la PUCESI el referido trabajo de graduación, respetando las políticas de propiedad intelectual de Universidad.

Ibarra, 21 de enero del 2019

(f.)…….

Cristian Javier Cuaspa Benavides

C.C. 1004637896

CERTIFICACIÓN ANTIPLAGIO

Yo Luis David Narváez Erazo, declaro que luego del proceso de revisión en el sistema antiplagio URKUND el porcentaje de similitud del trabajo de titulación denominado:" CONTROLADOR DE PROCESAMIENTO DE IMÁGENES BASADO EN PLATAFORMAS OPEN SOURCE DESTINADO A LA VISUALIZACIÓN EN PANELES DE LEDS RGB", es del 3%, de acuerdo al documento D40686018.

En base a lo anterior, considero que el trabajo de titulación NO ☐ SÍ ☒ cumple los requisitos de originalidad y autenticidad, de acuerdo con los requisitos establecidos por la ley.

Ibarra, 21 de enero del 2019

(f:) ……
Luis David Narváez Erazo
C.C.: 1002868378

Contenido

RESUMEN **11**
ABSTRACT **13**
INTRODUCCIÓN **14**
CAPÍTULO I: ESTADO DEL ARTE **1**
1.1 Sistemas Embebidos 1
1.1.1 Arduino 1
1.2 LED 16
1.2.1 LED RGB 16
1.2.2 Matriz de LED RGB 17
1.2.3 Multiplexación 20
1.3 Procesamiento de imágenes 21
1.3.1 Imagen digital 21
1.3.2 Procesamiento digital de imágenes 22
1.4 Alternativas de presentación de imágenes 2d existentes 25
CAPÍTULO II: MATERIALES Y MÉTODOS **27**
2.1 Marco metodológico 27
2.2 Contexto de uso de sistemas embebidos y algoritmos de procesamiento de imágenes en la PUCE-SI 28
2.3 Funcionalidades del controlador 28
2.4 Flujograma de funcionamiento general del controlador 29
2.5 Estructura del controlador 30
2.5.1 Estructura de software 30
2.5.2 Estructura de Hardware 31
2.6 Metodología del Diseño 32
2.6.1 Definición Integración de hardware y software del controlador 32
2.6.2 Desarrollo de componentes 34
2.6.3 Conexión 43
2.6.4 Ensamblaje 47
CAPÍTULO III: RESULTADOS **50**
3.1 Pruebas Unitarias 50
3.1.1 Control de Archivos 50
3.1.2 Procesador de imágenes 53
3.1.3 Presentación 56
3.1.4 Interfaz de control de usuario 56

3.2 Pruebas de integración 59
CONCLUSIONES 60
RECOMENDACIONES 62
BIBLIOGRAFIA 63
ANEXOS 65
GUIA GENERAL PARA EL USO DEL CONTROLADOR 67
INSTALACION DE LIBRERIAS NECESARIAS PARA EL CONTROLADOR ... 67
CARGA DE CODIGO A ARDUINO MAESTRO Y ESCLAVOS 69
VERIFICACION DE HARDWARE 72
INTERFAZ GRAFICA PYTHON 73
EJECUCION DE INTERFAZ GRAFICA 74
USO DE INTERFAZ GRAFICA 76
ERRORES COMUNES 78
Fallas de reconocimiento de MicroSD: 78
Error en la carga de imágenes desde la interfaz de control 78
Falla en la conexión con interfaz de control 79
Falla de representación en los paneles Led RGB 79
Falla de comunicación de Arduino con Monitor Serial 79

RESUMEN

El presente proyecto de investigación se realizó para la Pontificia Universidad Católica del Ecuador Sede Ibarra (PUCE-SI), una institución de educación superior que contribuye a la formación de profesionales para el norte del país.

El objetivo de la investigación es el desarrollo de una variante de algoritmo de "Compresión del rango dinámico", tomando como referencia los algoritmos ya existentes, adaptado a un controlador de procesamiento de imágenes que permita la manipulación de los pixeles de imágenes digitales bajo plataformas Arduino, para finalmente visualizar los datos de las imágenes generadas por el controlador en paneles de leds RGB.

Este proyecto usa la metodología de investigación documental, puesto que se ha recopilado información de diferentes fuentes bibliográficas, para apoyar el desarrollo e integración de los componentes del controlador, así mismo para la composición de la variante de algoritmo de corrección de rango dinámico; por otra parte, la investigación aplicada permite un enfoque hacia la aplicación del conocimiento adquirido para la creación del hardware y software de un controlador funcional.

Para el desarrollo del controlador se inicia aplicando la metodología ABC-Besoins-SEM, concebida para la gestión de requisitos de Sistemas Embebidos, determinando así la creación de los componentes: Control de archivos, Procesador de imágenes, Presentación e Interfaz de control de usuario, todos con características específicas para su funcionamiento ya sea en conjunto o independientemente, con esto se logra una integración óptima.

La simplicidad de uso de los componentes de software de este proyecto es muy importante, ya que se considera que estos pueden ser reutilizados en otros proyectos, por tal motivo, se ha optado por desarrollar los componentes control de archivos y procesador de imágenes como librerías, siguiendo la estructura de una librería para Arduino.

El control de usuario que se ha desarrollado es versátil, ya que se puede utilizar medios como: comunicación serial, botones físicos y bluetooth para acceder a las funciones que proporciona este componente, aprovechando dicha característica, se desarrolló una

interfaz gráfica en Python que usa comunicación serial RS232 para acceder a los comandos y hacer amigable el manejo del controlador.

Palabras Clave: Procesamiento de imágenes, Arduino, LED RGB, Compresión de rango dinámico, Python.

ABSTRACT

The present research project was carried out for the Pontificia Universidad Católica del Ecuador Sede Ibarra (PUCE-SI), a higher education institution that contributes to the training of professionals for the north of the country.

The objective of the research is the development of a variant algorithm of "Dynamic Range Compression", as a reference the already existing algorithms, and an image processing controller to allow the handling of digital image pixels under Arduino platforms, to finally visualize the data of the images generated by the controller in RGB LED panels.

This project covers documentary research, information has been gathered from different bibliographic sources, to support the development and integration of controller components, as well as the composition of the dynamic range correction algorithm; On the other hand, applied research allows a focus towards the application of acquired knowledge for the creation of hardware and software of a functional controller.

For the development of the controller the ABC-Besoins-SEM application is started, it was designed for the management of the requirements of the Embedded Systems, the creation of the components was determined: File Control, Image Processor, Presentation and Control Interface of User, All the specific characteristics for its operation either together or independently, with this an optimal integration is achieved.

The simplicity of use of the software components of this project is very important, since it is considered that these can be reused in other projects, for this reason, it has been chosen to develop the components of file control and image processing as libraries, Following the structure of a library for Arduino.

The control of the user that has been developed is versatile, that means can be used such as: serial communication, physical buttons and bluetooth to access the functions that use this component, the use of this characteristic feature, the graphical interface in Python that uses RS232 serial communication to access the commands and make a friendly handling of the controller.

Keywords: Image processing, Arduino, RGB LED, Dynamic range compression, Python.

INTRODUCCIÓN

El Procesamiento de digital de imágenes es un conjunto de técnicas que se aplican a las imágenes con el objetivo de mejorar la interpretabilidad de la misma, ya sea para búsqueda de información o mejora de calidad de imagen, existen muchos algoritmos dentro de la teoría, los cuales son aplicados en varios campos de la ciencia, un ejemplo clásico es la aplicación en imágenes de medicina, este campo ya ha sido cubierto desde hace ya muchos años atrás.

En la actualidad se necesita de software especializado con el fin de probar dichos algoritmos, MATLAB tiene una capacidad diversa para resolver problemas computacionalmente, el cual además proporciona un Tolbox de procesamiento de imágenes, la desventaja es que este software es privativo (Ritesh Reddy N. K., 2017).

Ahora bien, el procesamiento de imágenes en otras plataformas como las que utilizan lenguaje Java o C#, puede ser aplicado mediante librerías, sin embargo, no se puede explorar a profundidad y entender cómo funciona realmente; lo cual puede llegar a ser más difícil cuando se utiliza plataformas privativas.

Por otro lado, Arduino es una plataforma libre que en el ámbito multimedia específicamente de imagen digitales, ha sido utilizada casi siempre como un medio de captura o presentación de imágenes, mas no como un mecanismo de procesamiento o edición de la imagen, a pesar de que posee la capacidad para realizarlo, es en este punto cuando surge este proyecto, el de aprovechar el conocimiento de los estudiantes de la escuela de ingeniería en sistemas de la PUCE-SI en la plataforma Arduino, y crear un controlador que permita la aplicación de algoritmos de procesamiento de imágenes empleando esta plataforma y representarlas en paneles de leds RGB, el cual permitiría la manipulación directa de una unidad mínima de imagen (pixel).

La finalidad de este proyecto es desarrollar un controlador que permita la manipulación de los pixeles de imágenes digitales, y aplicar algoritmos de procesamiento de imagen sobre dichos pixeles, presentando los resultados a través de paneles de LED RGB; para lo cual se empleara una variante del algoritmo de Compresión del rango dinámico, con el fin de demostrar el funcionamiento del controlador.

Existen investigaciones que contienen características similares a las propuestas en este proyecto, tal es el caso de “The LED-matrix library is (c) Henner Zeller” (Zeller,

github.com, 2016), en el cual se utiliza Raspberry Pi y cuatro paneles LED RGB, semejante a este proyecto ya que se usa el mismo número de paneles y se enfoca a la presentación de imágenes 2D, sin embargo, una de las diferencias es que en el presente proyecto se da la posibilidad no solo de representar imágenes , sino de modificarlas mediante la aplicación de algoritmos de procesamiento de imágenes sobre los pixeles, cabe destacar que la integración con software programado en otros lenguajes es posible en el presente proyecto.

CAPÍTULO I: ESTADO DEL ARTE

1.1 Sistemas Embebidos

Un sistema embebido o integrado es la combinación de hardware y software, diseñado para realizar tareas específicas, para lo cual posee únicamente el poder de cálculo, memoria y almacenamiento necesario para ejecutar dichas tareas; esta característica es lo que diferencia de ordenador personal, ya que estos están diseñados para ejecutar varias tareas y aplicaciones, con mayores prestaciones para ello.

Por lo general este tipo de sistemas es invisible, ya que puede ser parte de un sistema o componente más grande, por ende, el sistema embebido ejecuta una función pequeña para apoyar el objetivo principal de todo el componente o sistema.

Estos sistemas se los puede encontrar en casi toda herramienta o maquinaria utilizada para reducir el esfuerzo humano como por ejemplo brazos robóticos de ensamble; así como también en dispositivos que se emplean en el diario vivir, ejemplo lavadoras, microondas, celulares, etc. (Urresta Cueva, 2010).

1.1.1 Arduino

Arduino es una plataforma de hardware libre, basada en una placa con un microcontrolador y un entorno de desarrollo, ambos diseñados para facilitar el uso de la electrónica en proyectos multidisciplinarios (Mario Ibarra, 2013), el propósito de esta placa es romper la barrera de acceder a la programación de microcontroladores sin tener conocimientos profundos de ellos, y ser accesible a todos los programadores, aficionados de electrónica, personas comunes con curiosidad por entrar al mundo de la electrónica, cabe destacar que el acoplamiento de las entradas y salidas tales como sensores, botones, potenciómetros, leds, pantallas led, etc. Es sencilla de tal manera que no se necesita ser un ingeniero profesional para manejar el hardware de Arduino y componentes externos que ayuden a emprender y completar un proyecto, ya sea aficionado, o en muchos casos de domótica, las posibilidades son infinitas y al alcance de todos.

Arduino como plataforma electrónica de bajo costo basada en el microcontrolador Atmel se utiliza en muchas aplicaciones y en varios campos. De hecho, sus características en términos de entradas, salidas, CPU y RAM hacen posible desarrollar sistemas inteligentes rápidos (Fabrice Morganti, 2017) esta plataforma está compuesta por hardware y software en los cuales se debe mencionar los principales elementos para que el emprendimiento de un proyecto de Arduino pueda iniciar, cabe destacar que, en cuestión de hardware, la placa está disponible en diferentes modelos, unos con más capacidad o funcionalidades que otros, pero en general todos tienen un mismo principio y que tanto el software de Arduino como IDE y lenguaje no cambian para diferentes tipos de Arduino.

1.1.1.1 Software:

El software Arduino ha sido acoplado de manera que la programación sea fácil para los usuarios. Está diseñado para introducir programación a personas que no están familiarizadas con el desarrollo de software, lo que hace es conveniente para los estudiantes (Galadima, 2014). Toda la documentación del lenguaje, librerías, etc. están en la página web oficial de Arduino.

a) IDE

Para programar el Arduino se debe utilizar el programa (software) Arduino IDE (Integrated Development Environment), el cual es una pieza de software en donde se desarrolla el código del lenguaje original de Arduino.

El IDE permite escribir código que después será compilado y enviado hacia la placa Arduino correspondiente, una vez cargado el código en la placa, esta ejecuta las instrucciones programadas en un bucle e interactuando con dispositivos externos, según se haya programado.

El IDE, por defecto guarda los proyectos con el nombre de Sketch.

b) Lenguaje de programación Arduino

Arduino Programming Lenguaje (APL): Cuando se habla de Arduino se cree que tiene un lenguaje de programación propio, esto no es cierto, la programación se la realiza en C++, pero Arduino facilita el uso de este lenguaje con librerías, las cuales

hacen que la programación de pines de entrada, salida y los puertos de comunicación sea sencilla, el IDE de Arduino mencionado anteriormente ya incluye estas librerías de forma automática y no hay necesidad de hacer declaraciones de <include>, salvo que se necesite una librería externa para controlar algún tipo de sensor.

c) Bootloader

El bootloader o gestor de arranque es un pequeño programa almacenado en la memoria flash del microcontrolador en la que además se almacena el sketch, este bootloader en primera instancia prepara todo para que el sketch pueda ser cargado e inicie cuando el Arduino se enciende o reinicia, Es por eso que cuando un programa se carga desde el IDE, el Arduino se reinicia con el fin de ejecutar el bootloader y que este se encuentre unos segundos a la espera de que llegue un nuevo sketch y cargarlo, caso contrario el bootloader determina que no se trata de un cambio de sketch y carga el que estaba anteriormente almacenado, un dato importante del bootloader de Arduino es que este se encuentra liberado, es decir, que su código se encuentra en internet y se puede reprogramar un microcontrolador compatible con el bootloader de Arduino cuantas veces sea necesario, esto es muy útil cuando se termina un proyecto y se desea implementarlo, así se puede cambiar el microcontrolador con el sketch del proyecto por uno nuevo y cargar el bootloader, el microcontrolador con el sketch es implementado y el nuevo está listo para nuevos proyectos.

1.1.1.2 Hardware

Para este apartado se define el hardware de Arduino Mega y Arduino 101, que son las placas que se utilizan en el desarrollo de la presente investigación.

a) Arduino 101

Esta placa contiene las prestaciones necesarias para ejecutar tareas que requieren un nivel alto de procesamiento con mayor rapidez.

Ilustración 1:Arduino 101
Fuente: Página de la tienda oficial de Arduino en Estados Unidos (Arduino, 2018)

Esta versión de Arduino, mantiene el modelo del Arduino uno como se puede apreciar en la ilustración1, pero con mejores características de hardware las cuales se detallan en la tabla 1.

Tabla 1: Especificaciones técnicas de Arduino 101

Microcontrolador	Intel Curie
Tensión de funcionamiento	3.3V (E / S tolerante a 5V)
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (límite)	7-17V
Pines de E / S digitales	14 (de los cuales 4 proporcionan salida PWM)
Pines de E / S digitales de PWM	4
Clavijas de entrada analógica	6
Corriente DC por Pin E / S	20 mA
Memoria flash	196 kB
SRAM	24 kB
Velocidad de reloj	32 MHz
LED_BUILTIN	13
Características Especiales	Bluetooth LE, acelerómetro / giroscopio de 6 ejes
Longitud	68.6 mm
Anchura	53.4 mm
Peso	34 gr.

Fuente: Adaptado el resumen de características de hardware de Arduino 101 (Arduino, 2018) sitio web oficial de Arduino de Estados Unidos.

b) Arduino Mega 2560

Arduino Mega es una placa que tiene todo lo necesario para soportar la programación del capaz de realizar muchas tareas debido a su microcontrolador Atmega1280 y la gran cantidad de pines, los cuales se pueden apreciar en la ilustración 2.

Ilustración 2:Arduino Mega 2560
Fuente: página de la tienda oficial de Arduino en Estados Unidos (Arduino, 2018)

Las especificaciones técnicas que posee Arduino Mega 2560 lo convierten en un dispositivo estable y robusto que soporta un uso prolongado en la ejecución de tareas que requieren un nivel moderado de procesamiento, dichas especificaciones se detallan en la tabla 2.

Tabla 2: Características de hardware de Arduino

Microcontrolador	ATmega1280
Tensión de funcionamiento	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (límites)	6-20V
Pines de E / S digitales	54 (de los cuales 15 proporcionan salida de PWM)
Clavijas de entrada analógica	dieciséis
Corriente DC por Pin E / S	40 mA
Corriente DC para 3.3V Pin	50 mA
Memoria flash	128 KB de los cuales 4 KB utilizados por el *bootloader*
SRAM	8 KB
EEPROM	4 KB
Velocidad de reloj	16 MHz

Fuente: Adaptado el resumen de características de hardware de Arduino Mega (Arduino, 2018) .

1.1.1.3 Licencia Creative Commons Attribution ShareAlike

La licencia para creaciones en Arduino está contemplada dentro del proyecto Creative Commons que es una organización no gubernamental sin fines de lucro la cual tiene como principal objetivo es “restablecer un equilibrio entre los derechos de los autores, las industrias culturales y el acceso del público en general a la cultura.” (Vercelli, 2009).

El proyecto Creative Commons ha diseñado un sistema web de licencias en el cual los autores/creadores, empresas e instituciones públicas pueden utilizarlo de forma gratuita y en todo el mundo para realizar la gestión de los derechos de sus obras intelectuales,

con el objetivo de “aumentar la cantidad y calidad de las obras intelectuales que circulan por las redes electrónicas digitales y que tienen un carácter común”, cuando se menciona “carácter común” se refiere a obras que se pueden copiar, compartir, derivar o mesclar.

Con Creative Commons se puede licenciar todo tipo de obras intelectuales en el cual entra las creaciones con Arduino, además se puede licenciar otras obras tales como fotografías, dibujos, música, animaciones, contenidos educativos, etc. Es decir, todo lo que sea considerado obra de intelecto humano.

En el caso del software Creative Commons recomienda utilizar la Licencia Pública General [GPL] de la Fundación para el Software Libre [FSF], debido a que el software por sus especiales características debe ser beneficiado por el uso de licencias copyleft.

Existen 4 condiciones para construir una licencia Creative Commons las cuales se detallan a continuación:

- Atribución: se debe reconocer siempre la autoría.
- No Comercial: No se puede explotar la obra de forma comercial.
- Sin obras derivadas: No se permite modificar la obra original para crear una obra derivada.
- Compartir Obras Derivadas Igual: Permite crear obras derivadas siempre que mantengan la misma licencia al ser divulgadas.

Entonces “Creative Commons Attribution ShareAlike” cuya traducción al español es “Creative Commons Atribución – Compartir Obras Derivadas” Igual es una combinación de las condiciones “Atribución” y “Compartir Obras Derivadas” mencionadas anteriormente, esto quiere decir que las licencias para las creaciones en Arduino permiten uso comercial de la obra y de las obras derivadas siempre que mantengan la misma licencia al ser divulgadas y se reconozca la autoría.

Como ya se ha mencionado, el proyecto Creative Commons ha desarrollado un sistema en la web que permite el acceso a nivel mundial. Ecuador no es la excepción, se puede acceder a la plataforma a través de creativecommons.org y encontrar los tipos de licencias existentes contempladas en el proyecto.

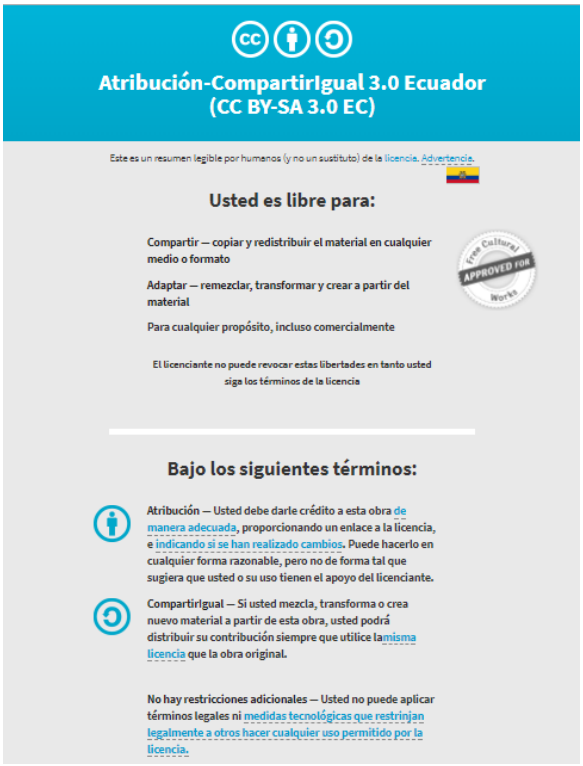

Ilustración 3:Licencia Atribución-Compartir Igual 3.0 Ecuador
Fuente: *(Creative Commons, 2017)*

"Con una licencia Creative Commons mantiene sus derechos de autor, pero permite a otras personas copiar y distribuir su obra, siempre y cuando reconozcan la correspondiente autoría y solamente bajo las condiciones que especifique en la licencia." (Biblioteca de la Universidad de Sevilla, 2015).

1.1.1.4 Librerías de Arduino

Una librería en general es un código hecho por terceros, ya sea empresas o programadores normales, estas tienen la finalidad de facilitar la programación y ser implementadas de una manera muy sencilla.

Las librerías son una colección de código funcional que facilita el trabajo con hardware o la manipulación de datos (V. Tipsuwanpornl, 2014) en un determinado lenguaje, en el caso de Arduino dichas librerías por lo general ayudan a facilitar el manejo de hardware exterior como una pantalla, matrices de leds, módulos bluetooth y un sinfín más de elementos.

Claro está que no todas las librerías pueden utilizarse libremente como por ejemplo librerías desarrolladas por Apple, pero si se habla de Arduino, todas las librerías que

han sido desarrolladas y probadas bajo esta plataforma heredan la licencia “Creative Commons AttributionShareAlike 3.0”

Existen muchas librerías de Arduino, algunas oficiales de la misma plataforma y otras que han sido desarrolladas por programadores que decidieron publicarlas para que la comunidad mundial pueda hacer uso de ellas.

Para fines de este proyecto se ha optado por hacer uso de librerías las cuales faciliten el trabajo y acorten el código, para dedicarse únicamente al objetivo, sin tener que crear algo que ya este hecho y probado, estas librerías se detallan a continuación con su respectiva explicación.

a) Librería SPI Arduino

Para entender el funcionamiento de la librería en Arduino se debe entender que es SPI. (Serial Peripheral Interface) es una interfaz que facilita la transferencia de datos seriales sincrónicos (M.K. Md Arshad, 2006), es decir, permite la transferencia de datos desde un hardware exterior a la placa, este resulta ser indispensable por hacer posible la comunicación con un módulo SD entonces este trabaja conjuntamente con la librería SD.

SPI (Serial Peripheral Interface bus) es un protocolo usado para la comunicación a corta distancia principalmente en sistemas integrados, para que la comunicación con hardware exterior sea posible, dicho hardware debe ser un dispositivo SPI, por ejemplo, pantallas de cristal líquido o tarjetas SD.

La comunicación con dispositivos SPI se realiza en modo full dúplex, utilizando una arquitectura de Master-Slave (Maestro-Esclavo), esta interfaz especifica 4 salidas lógicas para su funcionamiento los cuales se detallan a continuación:

1. SCLK: reloj en serie (salida del maestro).
2. MOSI: entrada esclava de salida maestra, o entrada esclava maestra fuera (salida de datos del maestro).
3. MISO: Salida esclava de entrada maestra, o Salida esclava maestra (salida de datos del esclavo).
4. SS: Selección de esclavo (a menudo activo bajo , salida del maestro)

SPI a veces se denomina bus serie de cuatro hilos por su número de salidas.

La arquitectura de SPI permite que la comunicación se establezca únicamente con un maestro, en cuanto a los esclavos pueden ser uno o más dependiendo del microcontrolador, en ilustración 4 se muestra el diagrama de conexión general de SPI independientemente del microcontrolador.

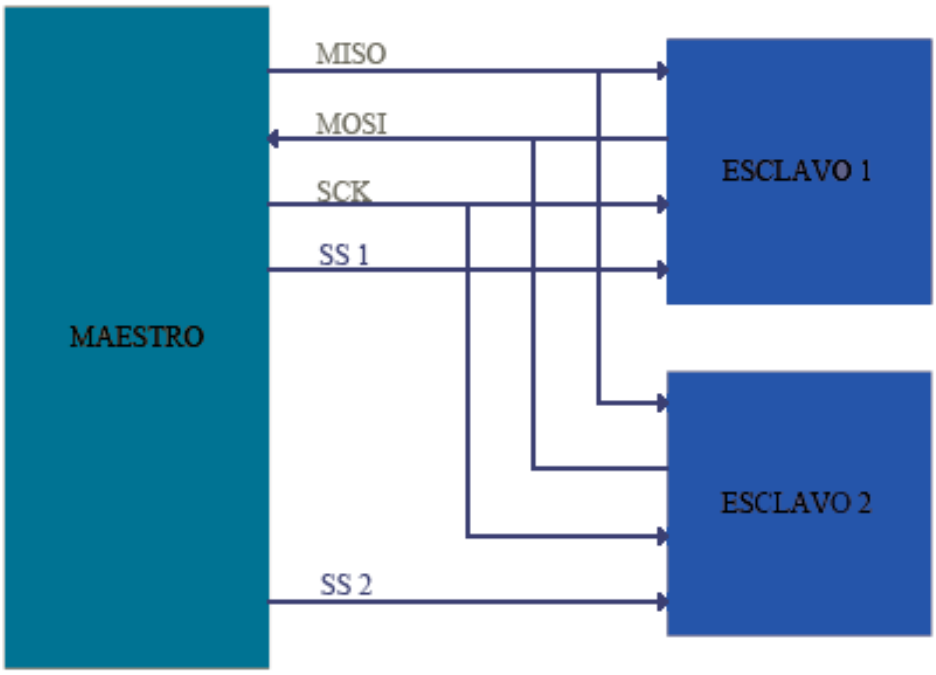

Ilustración 4:diagrama de conexión general SPI
Fuente: Cristian Cuaspa

Para algunos dispositivos SPI los nombres de las entradas cambian aquí se describen algunos de los nombres conocidos.

Reloj en serie:

SCLK: SCK.

Salida maestra => Entrada esclava:

MOSI: SIMO, SDO, DI, DIN, SI, MTSR.

Entrada maestra => Salida esclava:

MISO: SOMI, SDI, DO, DOUT, SO, MRST.

Selección del esclavo:

SS: $\overline{SS}$, SSEL, CS, $\overline{CS}$, CE, nSS, / SS, SS #.

SPI como librería de Arduino implementa la interfaz de comunicación serial SPI, para que la placa pueda comunicarse con los dispositivos externos SPI siendo Arduino el maestro y los periféricos los esclavos, un ejemplo básico sería la de la conexión de un módulo SD con Arduino como se muestra en la ilustración 5.

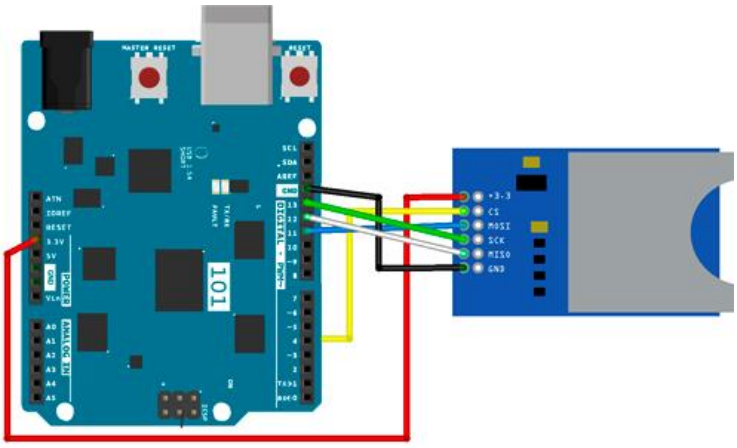

Ilustración 5:comunicación SPI de Arduino con modulo SD
Fuente: (Indexxit, 2016)

Como se puede observar en la ilustración 5 el módulo SD tiene los pines necesarios para una comunicación SPI, en cuanto a la placa Arduino los pines están definidos por la librería, cabe recalcar que los pines varían en los diferentes modelos de Arduino.

Tabla 3: Comparativa de pines de conexión SPI entre Arduino 101 y Mega 2560

Arduino	**MOSI**	**MISO**	**SCK**	**SS (esclavo)**	**Voltaje**
101	11 o ICSP-4	12 o ICSP-1	13 o ICSP-3	10	5V
Mega2560	51 o ICSP-4	50 o ICSP-1	52 o ICSP-3	53	5V

Fuente: Adaptación de la tabla de conexiones SPI de las diferentes placas de Arduino (Arduino, 2017)

La librería SPI de Arduino proporciona los siguientes métodos para realizar comunicación SPI:

- begin(): Inicializa el bus SPI configurando los pines correspondientes a SCK, MOSI, y SS
- end(): Deshabilita el bus SPI.

- setBitOrder(): Configura el orden de los bits enviados como el menos significativo primero o el más significativo primero.
- setClockDivider(): Configura del divisor de reloj en el bus SPI. ES decir, configura la velocidad del bus.
- setDataMode(): Configura el modo de dato del bus SPI, es decir, polaridad y fase del reloj.
- transfer () Transfiere un byte sobre el bus SPI, tanto de envío como de recepción.

b) Librería SD Arduino

Esta librería en particular permite el manejo de un módulo SD (Secure Digital) en Arduino, con la cual se pretende guardar archivos de imagen para su posterior tratamiento, es decir, realizar el proceso de lectura y escritura de archivos.

Para hacer fácil la comprensión de esta librería a continuación se muestra el diagrama de clases en el que se puede observar las funciones y atributos con los cuales funciona esta librería.

Ilustración 6:Diagrama de Principales Clases de Librería SD
Fuente: Cristian Cuaspa

La ilustración 6 muestra 3 principales clases de la librería, existen otras clases de la librería que no se han incluido debido a que son innecesarias para el desarrollo del proyecto, estas 3 clases trabajan conjuntamente para permitir inicialización de diferentes tipos de tarjetas SD, lectura, escritura de archivos.

Los métodos de las clases que se utilizan para el desarrollo del proyecto son descritos a continuación:

- SDClass
    - SD.begin(csPin uint8_t): inicializa la tarjeta SD, *csPin* es el parámetro en el cual hay que especificar el pin SS de comunicación SPI del Arduino, este método retorna un booleano el cual indica si se inicializo o no la tarjeta
    - SD.open(filename: string): Este método devuelve un objeto de clase file para poder manipular el archivo, *filename* es el parámetro de entrada en el que se debe especificar el nombre del archivo.

- File

  La clase file permite el manejo de un objeto como archivo, los métodos que se utilizaran son:

  - file.avaliable(): indica si el archivo abierto está disponible o no, este retorna un valor entero
  - file.close(): cierra el archivo, un archivo es abierto por la clase SD con la función open(), sin embargo el cierre se lo realiza al objeto file.
  - file.name(): este método retorna un valor de tipo char representando el nombre del archivo, esto se utiliza para cuestiones de presentación.

- Sd2Card

  Esta clase se puede definir como una especie de puente entre las librerías SD y SPI ya que hace posible la comunicación, es en esta clase donde se definen los pines, dependiendo del modelo de Arduino y determina que tipo de tarjeta de memoria se está intentando utilizar para proporcionar los métodos adecuados.

Se han desarrollado gran cantidad de proyectos usando módulos SD que utilizan comunicación SPI como se puede apreciar en el artículo "*Development of an Arduino-based automated household utility power monitoring system*" (Alan L. Vergara, 2014), este articulo describe la realización de un proyecto en el cual se utiliza una placa Arduino que se conecta a un sensor de corriente y almacena los datos en bruto a una Memoria SD mediante un módulo SD/MMC, además cuenta con un chip DS1307 que proporciona información de fecha y hora actual, esta estructura ha sido diseñada para analizar los datos en bruto proporcionados por el sensor y almacenados en la memoria SD, para así mostrar la potencia promedio consumida en vatios y su costo en tiempo real.

Este proyecto tan útil como interesante es uno de los muchos que corroboran el funcionamiento de la comunicación SPI entre placa Arduino y Módulos SD.

c) Librería JPEGDecoder Arduino

Esta librería es una de las más importantes de este proyecto, ya que es la que permite el tratamiento de imágenes descomponiéndolas en pixeles y proporcionando los valores RGB de cada pixel, esto es muy útil ya que para la aplicación de algunos algoritmos de procesamiento de imágenes se necesita conocer el valor real de cada color en un pixel.

Además, esta librería contiene métodos para la manipulación de objetos de tipo file siempre y cuando este contenga un archivo de imagen.

Esta librería proporciona varios métodos de decodificación de archivos de imagen, para diferentes escenarios o casos en el que el archivo cambie de origen, pero al final todos tienen el mismo objetivo, es por este motivo que se describe a continuación solo el método que se utiliza en este proyecto.

- JpegDec.decodeSdFile(jpgFile): Este método prepara un archivo de imagen jpg abierto desde una memoria SD para ser decodificado en partes.

Cuando se aplica este método enviando como parámetro un archivo, esto no quiere decir que la imagen se encuentra en su totalidad decodificada y convertida en matriz, esto solo quiere decir que la imagen esta lista para ser decodificada, JpegDec se puede manejar como un objeto que contiene la imagen.

- JpegDec.read(): Este Método lee un MCU (Minimum Coded Units) de la imagen, un MCU es una unidad mínima de código, este es una matriz de 8x8 de 16bits, cada vez que este ejecute este método, este recorrerá la imagen a la siguiente MCU de izquierda a derecha, y hacia abajo.
- JpegDec.pImage: Este es un atributo que cambia cada vez que se ejecuta el anterior método, pero es muy importante ya que es el que contiene la información del MCU actual.

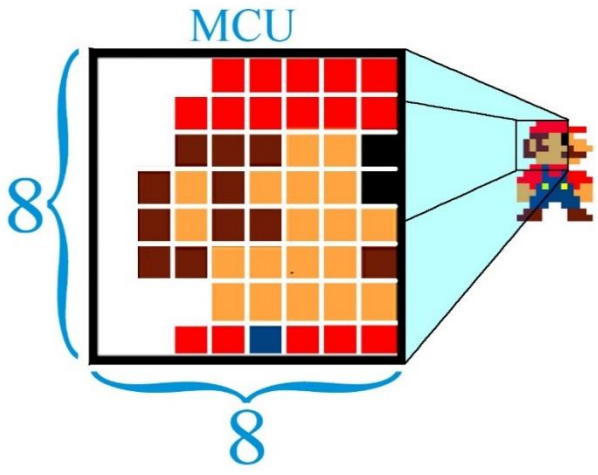

Ilustración 7:MCU de una imagen
Fuente: Cristian Cuaspa

Una vez obtenido el MCU actual la librería proporciona métodos específicos para el manejo de MCU tales como el ancho, alto, posición x o y, esto con el objetivo de facilitar la lectura de estos en forma de matriz, y su posterior ubicación en un plano.

Es aquí cuando se llega a la unidad mínima de la imagen, un pixel el cual está listo para cualquier tratamiento o algoritmo, dicho pixel está representado en el formato RGB565 que guarda las intensidades en 2 bytes o 16 bits, los colores rojo y azul tienen valores de intensidad que varían entre 0 y 31 y el verde varía entre 0 y 63, el 565 indica el número de bits que se utiliza para cada color (Juan Francisco Ordóñez, 2016), proporcionando los valores de color de cada pixel, pero primero hay que hacer una descomposición del valor para obtener cada color, esto se logra con funciones propias del lenguaje que permiten el recorrido y aplicación de máscaras de bits, para extraer bits específicos de un valor de 16 bits, la forma en la que se organizan los colores en un valor de 16 bits se muestra en la ilustración 8.

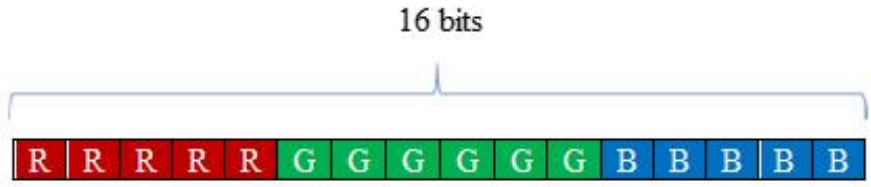

Ilustración 8:Representación RGB en 16 bits
Fuente: Cristian Cuaspa

La ilustración 8 hace referencia al modo 565 en una notación RGB en el cual se tiene un bit adicional para el color verde, porque el componente verde da más brillo de un color en el ojo humano.

d) Librería RGB-Matrix-Panel Arduino

Esta librería facilita el control de una matriz de LED's RGB, proporcionando funciones que permiten mostrar colores en una posición especifica de la matriz RGB basado en las posiciones x e y.

Mediante el uso de las funciones encontradas en la librería Adafruit GFX, que proporciona un conjunto común de primitivas gráficas (puntos, líneas, círculos, etc.) (Adafruit ®, 2017), se logra la representación de los valores de un color RGB en forma de pixel en la matriz empleada, la cual puede ser de dimensiones: 16*32, 32*32 y 32*64, de la marca Sparkfun o Adafruit.

1.2 LED

LED (Light-Emitting Diode) su traducción al español es (diodo emisor de luz), el LED es un componente electrónico que emite luz cuando es electrificado por una corriente pequeña, un diodo emisor de luz consiste en una serie de materiales semiconductores en capas. En el LED, electricidad se convierte directamente en partículas de luz, fotones, (Gustavsson, 2014) lo que da como resultado eficiencia en electricidad, ya que a comparación de otras fuentes de luz en la que la mayoría de la electricidad se convierte en calor y un porcentaje relativamente bajo en luz, como por ejemplo una lampara incandescente.

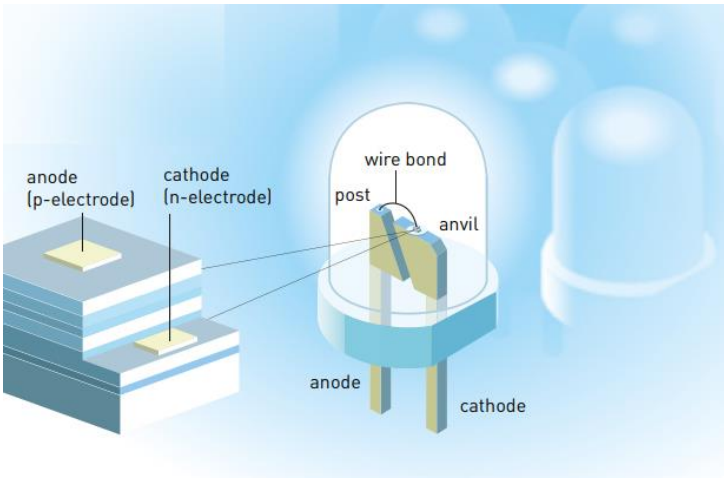

Ilustración 9:Partes de un led
Fuente: (Johan Jarnestad/The Royal Swedish Academy of Sciences, 2014)

En la ilustración 9 se hace énfasis a una parte importante del led la cual se denomina chip semi conductor, que está constituido por 2 electrodos tipo n y p los cuales determinan el color de la luz que emitirá el led, según la composición química de sus capas, existen varias composiciones químicas utilizadas en leds, a continuación, se muestras solo las composiciones de los colores más importantes.

- Arseniuro Fosfuro de Galio =Rojo
- Fosfuro de Galio = Verde
- Carburo de Silicio =Azul

1.2.1 LED RGB

Un LED RGB es la combinación de 3 diodos leds normales de colores rojo, verde y azul, comprimidos en un solo, RGB significa RED, GREEN, BLUE los cuales hacen

referencia a los colores principales que puede emitir este tipo de led. La mezcla de los colores primarios rojo, verde y azul permite obtener cualquier color dentro del espectro electromagnético visible. Los LED's RGB toman este principio con el fin de obtener luz blanca (Rodriguez, 2009).

Existen dos tipos de LED's RGB el ánodo común y cátodo común donde la única diferencia es la conexión de voltaje y tierra, esto se debe a que los leds internos comparten o bien el polo positivo (Ánodo) o el polo negativo (Cátodo) de ahí parten diferentes modelos, que varían en intensidad, diseño, tamaño, etc.

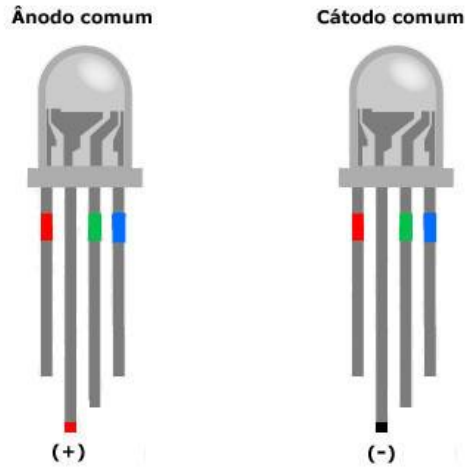

Ilustración 10:Tipos de LED's RGB Ánodo y Cátodo
Fuente: (Squids.com, 2018)

1.2.2 Matriz de LED RGB

Una Matriz LED RGB es un conjunto de leds RGB, los LED están dispuestos en filas y columnas, esto se hace para minimizar el número de pines necesarios para controlarlos (H.BOUAZZAa, 2016), estas matrices de leds son controladas comúnmente por multiplexación técnica la cual se describe más adelante.

En el mercado existen gran variedad de Matrices LED RGB, las cuales cambian en precio, marca, cantidad de leds, etc. La Matriz que se utiliza para este proyecto es 32X32 RGB LED MATRIX PANEL - 4MM PITCH de la distribuidora Adafruit

Ilustración 11:32X32 RGB LED MATRIX PANEL
Fuente: (Adafruit ®, 2017)

Esta matriz ofrece una librería que hace posible su utilización con Arduino, los detalles técnicos se describen a continuación:

DETALLES TÉCNICOS

- Dimensiones: 128 mm x 128 mm x 17 mm (5.0 "x 5.0" x 0.65 ")
- Peso del panel con cables IDC y cable de alimentación: 170 g
- Entrada de alimentación regulada de 5V, 4 A máx. (Todos los LED encendidos)
- Entrada de nivel de lógica de datos de 5V
- 2000 mcd LED en 4 mm de tono
- 1/16 tasa de escaneo
- Pantalla interior, visibilidad de 150 grados
- Las pantallas son 'conectables' - conecta una salida a la siguiente entrada - pero nuestro código de ejemplo Arduino no es compatible con esto

los detalles técnicos de este producto han sido extraídos de la web oficial de (Adafruit ®, 2017)

"Una matriz de LED's es un conjunto o arreglo de LED's dispuestos de tal forma que forman filas y columnas de la unión común de sólo cátodos y/o sólo ánodos". (Soto Almeida Juan, 2009) el manejo de leds de forma individual es muy simple, pero cuando se tiene un conjunto de varios leds, los recursos que se necesitan son excesivos, es por esto que en una matriz de LED's estos son colocados estratégicamente de manera que una fila o columna de LED's comparta voltaje o tierra dependiendo de la configuración de la matriz ánodo común o cátodo común como se muestra en la ilustración 12.

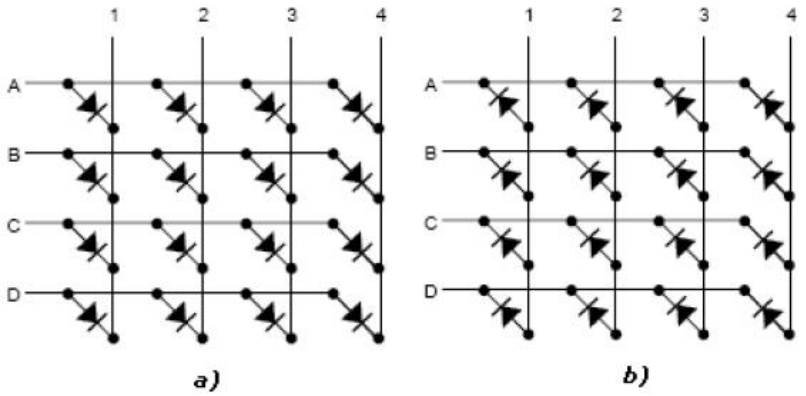

Ilustración 12:Arreglo matricial de a) ánodo común (por fila) b) cátodo común (por fila)
Fuente: extraído del artículo sobre sistema de información visual (Soto Almeida Juan, 2009)

Cuando se habla de una configuración "Ánodo común" en matrices de LED's , quiere decir que las filas de la matriz comparten líneas de voltaje y las columnas líneas de tierra, el objetivo de esto es que cuando se quiera encender por ejemplo el LED A1 de la ilustración 12, se debe conectar voltaje en la fila A y tierra en la columna 1, parece ser que daría igual tener leds separados, ya que se tiene que conectar voltaje y tierra para que se encienda un LED, pero esto no es del todo cierto ya que si se quisiera encender también el LED A2 solo se conectaría tierra en la columna 2.

En contraste cuando se habla de una matriz con configuración "Cátodo común" quiere decir que se comparte una línea de tierra por fila y una línea de voltaje por columna, lo cual es lo contrario de la configuración anterior.

Cuando se ha entendido que es una matriz de leds normales es fácil entender el concepto de "matriz de LED's RGB", ya que se puede decir que es una matriz que contiene un arreglo de 3 leds por cada espacio, como se muestra en la ilustración 13.

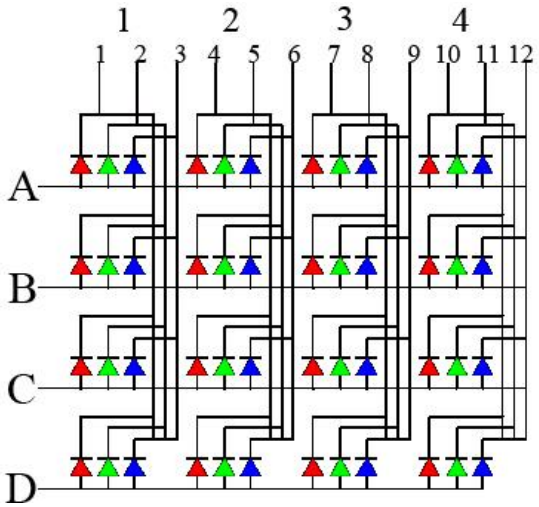

Ilustración 13:Arreglo matricial de LED's RGB de ánodo común
Fuente: Cristian Cuaspa

El manejo de una matriz de LED’s RGB no es diferente al de una matriz de LED’s normal de un solo color, sol hay que pensar en esta como una matriz que tiene más columnas, y que el conjunto de 3 pines ya sean ánodo o cátodo representan un led RGB, es así que por ejemplo si se quiere encender el LED A1 de color blanco, se debe conectar voltaje a la fila A y tierra en los 3 pines de la columna 1, ya que al encender rojo, verde y azul, se obtiene el color blanco.

1.2.3 Multiplexación

Existen varios campos de aplicación de esta técnica como por ejemplo telecomunicación, informática, electrónica, redes básicamente multiplexar consiste en compartir un mismo canal para él envió de datos a diferentes receptores, en el caso de la multiplexación para una matriz de LED’s se utiliza la Multiplexación por división de tiempo, donde solo una fila de la matriz de LED’s es activada en un intervalo de tiempo. Este método se aplica porque un terminal del LED (sea el ánodo o el cátodo) está unido a una sola fila. (Soto Almeida Juan, 2009)

El proceso consiste en electrificar una fila y una columna en un intervalo de tiempo corto, entonces se enciende un LED, si se requiere encender varios leds a la vez, se electrifican las filas y columnas a las que corresponden los LED’s en un periodo muy corto de tiempo dentro de un bucle, el tiempo debe ser tan corto, para así engañar al ojo humano y simular que los LED’s seleccionados están encendidos al mismo tiempo, este proceso se conoce también como barrido.

Con esto se logra una administración eficiente de los recursos para encender leds de una matriz, si esta es manejada por un microcontrolador, se necesitan un numero de pines equivalentes a la suma de filas y columnas, que por lo contrario si no se utilizaría esta técnica, el microcontrolador necesitaría un numero equivalente a la multiplicación de filas por columnas.

Con el uso de microcontroladores especializados se puede optimizar aún más la multiplexación ya que no hace falta tener un pin por cada línea y por cada columna, sino que utiliza la señal de reloj para llegar a un led en una matriz, un ejemplo de microcontrolador especializado es el MAX7219, el cual proporciona el control de 64 diodos LED, dicho microcontrolador tiene una característica especial de proporcionar

soporte para la conexión en serie de 2 o más microcontroladores, logrando con esto que se pueda controlar más diodos LED usando solamente 4 pines si se utiliza Arduino.

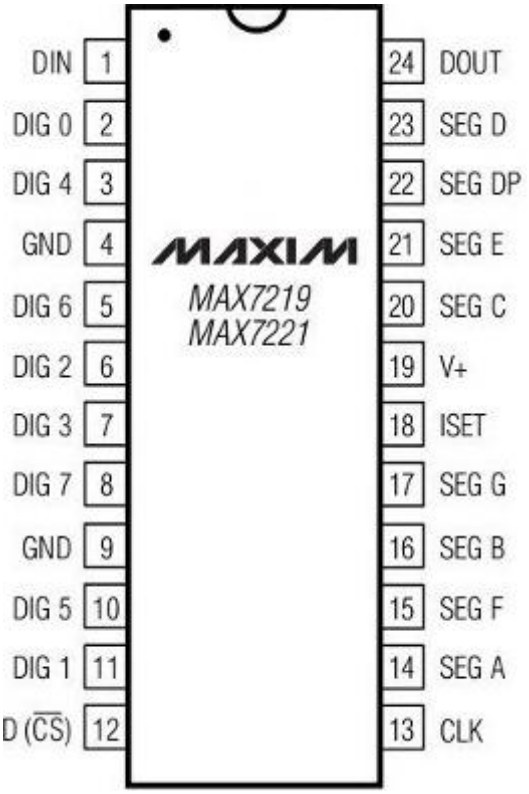

Ilustración 14: Representación gráfica de microcontrolador MAX7219
Fuente: Arduino and the MAX7219 LED Display Driver IC

1.3 Procesamiento de imágenes

1.3.1 Imagen digital

Una imagen puede ser definida matemáticamente como una función bidimensional,

$$f(x, y)$$

donde (x; y) son coordenadas espaciales (en un plano), y f en cualquier par de coordenadas es la intensidad o nivel de gris de la imagen en esa coordenada. (Vilet, 2005). Es una imagen digital cuando los valores de x, y, f son discretos finitos, esta imagen se compone de un numero finito de elementos los cuales ocupan un espacio y tienen un valor, a estos se los denomina pixeles, en el caso de imágenes a color la función se extiende en 3, ya que cada canal como el rojo, verde y azul tiene su valor discreto y finito, para objetivos de este proyecto se considerara las siguientes funciones para cada pixel: r(x, y), g(x, y) y b(x, y).

1.3.2 Procesamiento digital de imágenes

Cuando se habla de procesamiento digital de imágenes, básicamente se refiriere a como se cambia la naturaleza de la imagen, ya sea para mejorar su representación gráfica para una mejor comprensión humana o para una interpretación más adecuada de la máquina. (Ritesh Reddy N. K., 2018). El procesamiento digital de imágenes se puede entender además como un conjunto de procesos en los cuales se tiene como entrada una imagen y como salida una imagen o bien como entrada una imagen y como salida datos.

No existe un acuerdo general entre los autores con respecto a dónde se detiene el procesamiento de imágenes y comienzan otras áreas relacionadas, como el análisis de imágenes y la visión por computadora (Rafael C. Gonzalez, 2018), sin embargo, existe una clasificación que cubre gran parte del campo, y es muy fácil de entender.

Esta clasificación divide la materia en 3 diferentes tipos de procesos los cuales se detallan en tabla 3.

Tabla 3: Procesos que cubren el campo del procesamiento digital de imágenes

Nombre	Descripción
Procesos de Bajo Nivel:	Utilizan operaciones como el preprocesamiento de imagen para reducir el ruido, mejora del contraste, y filtros de enfoque. Se caracterizan por que sus entradas son imágenes y sus salidas también.
Procesos de Nivel Medio:	Operaciones como segmentación y clasificación de objetos individuales. Se caracterizan por que sus entradas son generalmente imágenes, pero sus salidas son atributos extraídos de esas imágenes (contornos, bordes, identidad de objetos individuales).
Procesos de Alto Nivel:	Implica el obtener algún significado de un conjunto de objetos reconocidos – análisis de imágenes – y, finalmente, realizar las funciones cognitivas asociadas con la vista. (ejemplo de símbolos de tráfico)

Fuente: Adaptado de Procesamiento Digital de Imágenes primer borrador (Vilet, 2005)

1.3.2.1 Compresión del Rango dinámico

La compresión de rango dinámico hace referencia a la transformación logarítmica que tiene su lugar en procesos de bajo nivel, ya que forma parte del mejoramiento de imagen, dicha transformación está dada por la ecuación de transformación logarítmica, según (Rafael C. Gonzalez, 2018):

$$s = c\,log(1 + r)$$

Ecuación 1:Ecuacion para transformación logarítmica
Fuente: (Rafael C. Gonzalez, 2018)

donde c es una constante, y es el valor del pixel. Se usa este tipo de transformación para exponer los valores de los pixeles oscuros de una imagen, mientras que los pixeles de alto valor se comprimen, como se puede apreciar en la siguiente imagen.

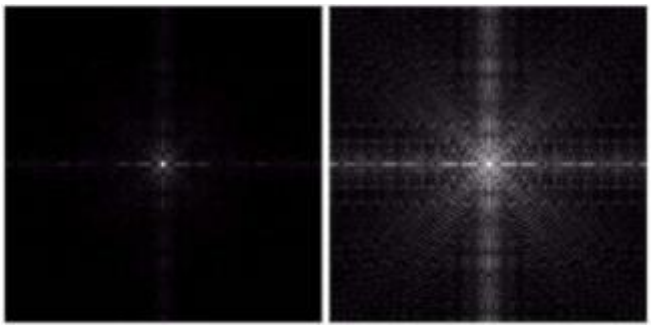

Ilustración 15:Espectro de Fourier y el resultado de una transformación logarítmica con c = 1
Fuente: (Rafael C. Gonzalez, 2018)

1.3.2.2 Adquisición de imagen digital

Se puede obtener una imagen digital mediante Herramientas informáticas que permitan dibujar con el mouse u otro dispositivo de señalamiento o generarlas a partir de la combinación de una fuente de iluminación y la absorción de energía de esta fuente por parte de un sensor (Rafael C. Gonzalez, 2018)

Existen tres tipos de configuración de sensores utilizados para adquirir imágenes

- Simple: esta es una adquisición de imagen con un solo sensor, el cual requiere movimiento mecánico en los ejes x e y para capturar la imagen, esta configuración es una forma barata pero lenta de obtener imágenes de buena calidad
- Banda: Adquisición con bandas o líneas de sensores, esta configuración es comúnmente encontrada en los escáneres de cama, en el cual se necesita un movimiento horizontal de la banda para capturar la imagen
- Arreglo: es la configuración que se encuentra en la mayoría de cámaras digitales, la cual consiste en que La energía de la fuente de iluminación es reflejada o transmitida por un elemento de la escena, la energía obtenida se colecta y se enfoca al plano focal por medio de una lente.

El resultado de estas configuraciones se denomina sensor de imagen, o en el caso de cámaras digitales Matriz CCD

a) Transformación a binario

La luz que capta cada fotodiodo genera una pequeña señal eléctrica y es convertida en datos digitales por un conversor ADC, que proporciona cadenas de ceros y unos (bytes), por cada fotodiodo, de ahí se puede determinar la profundidad de cada pixel.

b) Representación de imagen

Los valores captados y transformados de analógicos a digitales, se presentan como pequeños cuadros denominados pixeles en forma de mosaico, los pixeles suelen ser muy pequeños, por ende, el ojo y el cerebro se encarga de formar una imagen, ignorando los diminutos cuadros que conforman dicha imagen.

c) Formatos de imagen

Con el uso masivo de la información digital en la vida cotidiana, los medios digitales han pasado a ser importante no solo en informática sino en casi cualquier ámbito, ya sea para llamar la atención en publicidad, hasta ser de mucha ayuda en el área médica.
Tal es la necesidad del uso de medios digitales, que en los últimos años se han desarrollado múltiples formatos de compresión de medios digitales, ya sea en sonido, en fotografía, video, archivos etc. Para satisfacer la necesidad de portabilidad de estos medios, es decir, que se puedan utilizar cuando y donde sea.
La Imagen como medio digital de información se divide en 2 tipos:

- **Mapa de bits**: una matriz con información de colores en cada campo, que engañan al ojo humano, y este interpreta el conjunto de estos valores como una imagen.
- **Imagen vectorial**: a comparación de un mapa de bits, su tamaño es reducido, ya que los elementos que conforman la imagen se determinan por cálculos y formulas.

Las cuales pueden ser archivadas en diferentes formatos de archivos de imagen, existen 2 tipos de formatos, los formatos sin perdida los cuales a breve descripción desechan información redundante, y los formatos con perdida que eliminan información no relevante.
Entre los formatos de imagen con perdida más comunes se encuentran

- **BMP**: Bit Map (Mapa de bits) el cual brinda una profundidad de color de hasta 24 bits y soporta los formatos de color RGB, color indexado y escala de grises.

- **TIF**: Tag Image File Format contiene las mismas características que BMP con la diferencia que tiene mayor calidad por lo cual es utilizado en la industria grafica
- **GIF**: Graphics Interchange Format lo más relevante de este formato es que permite usar transparencia y la posibilidad de ver animaciones ya que puede hacer que distritos frames se ejecuten secuencialmente
- **JPG**: utilizado mayormente en la industria de la fotografía, ya que reduce el tamaño del archivo eliminando información irrelevante, pero compensando la perdida con técnicas de suavizado, este formato permite una profundidad de color de hasta 24 bits (Santiago, 2005).

1.4 Alternativas de presentación de imágenes 2d existentes

Las escasas alternativas open source existentes tales como: “The LED-matrix library is (c) Henner Zeller” (Zeller, 2016) , ofrecen características las cuales se puede considerar para el aprendizaje, pero no incluye técnica alguna de procesamiento de imágenes que se necesita, evitando así que esta tecnología sea objeto de estudio en su totalidad, por ende se tiene una deficiencia en la enseñanza que tenga como objetivo abordar el tema de procesamiento de imágenes ya que su estudio quedaría en una forma teórica y no aplicada como precisa esta materia.

Al no existir variedad en este tipo de soluciones en open source como efecto se tiene una escasa investigación en el ámbito educativo ya que al no tener bases de dónde empezar dificulta el avance y creación de nuevas técnicas de representación de calidad mediante la aplicación de algoritmos de procesamiento de imágenes ya desarrollados.

El costo elevado de algunas alternativas de presentación de imágenes en paneles de leds ya sea para publicidad o información (véase Tabla 4), frente a las formas convencionales de publicidad (véase Tabla 5), causa que en los países del tercer mundo exista un desconocimiento de las mismas, que influye indirectamente en la investigación, ya que al no existir demanda de algo que en países del primer mundo es común, un artículo publicado en el presente año establece que “el promedio el uso de esta tecnología es de 40 pantallas led luminosas por país, que funcionan en calles y autopistas de los países de Latinoamérica.” (Contaminacion Luminica En Chile Y Latinoamerica, 2017) Una cantidad relativamente baja a comparación a países

desarrollados, esto debilita la predisposición de investigar para producir dichas alternativas para estos países.

Tabla 4. Precio referencial de pantalla publicitaria

URBAN/DELTA P10mm URBAN SMD 1920mm x 1280mm	
NOMBRE DEL PRODUCTO	URBAN/DELTA
DESCRIPCIÓN	Pantallas exteriores múltiples usos y ubicaciones
LONGITUD	1920mm
ALTO	1280mm
RESOLUCIÓN PÍXEL	10mm URBAN
TIPO DE LED	SMD
DURACIÓN DEL LED	120000 horas
ANGULO DE VISIÓN	160°x140°
LUMINOSIDAD	Normal
PROTECCIÓN AMBIENTAL	Resistente a lluvia
CONSUMO	350W/m2 (Hora)
TENSIÓN DE TRABAJO	110V - 220V
PESO	45kg/m2
TIEMPO DE FABRICACIÓN	Consultar
PRECIO PANTALLA:	2.433,02 USD

Fuente: Adaptado de la proforma online de "visualled" (Cotizacion Online, 2017)

Fabricante de pantallas publicitarias proporciona el un valor que incluye costo de impuestos y envió a Ecuador

Tabla 5. Precio referencial de Gigantografía

LONA STARFLEX $10/M^2	
MATERIAL	LONA STARFLEX
LONGITUD	1920mm
ALTO	1280mm
PRECIO INCLUIDO IVA	$27.53 USD

Fuente: Adaptado de la proforma online de “blueink stants y gigantografías” (Cotizador en línea, 2017) fabricante de gigantografías.

CAPÍTULO II: MATERIALES Y MÉTODOS

2.1 Marco metodológico

El presente proyecto se basa en dos tipos de investigación, en primer lugar la investigación documental, puesto que se recopila información de diferentes fuentes bibliográficas, para apoyar el desarrollo e integración de los componentes del controlador, así mismo para la composición de la variante de algoritmo de corrección de rango dinámico; por otra parte, la investigación aplicada permite un enfoque hacia la aplicación del conocimiento adquirido para la creación de un controlador funcional, y que cumpla con los objetivos de este proyecto.

La ingeniería de requisitos aplicada descrita en este capítulo es donde se determinan los componentes de software a desarrollar, tales como Control de archivos, Procesador de imágenes, Presentación e Interfaz de control de usuario.

Como parte del desarrollo se incluye la creación de librerías ImageProcesor y FileControl.h, que hacen referencia a los componentes de software Procesador de imágenes y Control de archivos.

La descripción del desarrollo de la interfaz de control de usuario incluida, explica como este componente puede utilizarse con diferentes medios para acceder a los comandos que proporciona, dando la posibilidad de desarrollar una aplicación de escritorio que usa como medio la comunicación serial para acceder a los comandos de este componente, la cual también es descrita en este capítulo.

A través de una pila de pruebas se puede observar el comportamiento de los componentes del controlador, y además determinar cualitativamente si cumplen con los siguientes atributos de calidad: Escalabilidad y Reusabilidad.

En el caso del componente de interfaz de control de usuario se realiza pruebas de funcionamiento con diferentes medios, ejemplo: bluetooth, comunicación serial y botones.

En cuanto al ensamblaje se muestra los materiales utilizados y el proceso de armado presentando así una estructura final que se sostiene por si sola facilitando su uso en presentaciones.

2.2 Contexto de uso de sistemas embebidos y algoritmos de procesamiento de imágenes en la PUCE-SI

Dentro del pensum de Ingeniería en Sistemas se encuentran las materias: Electrología y Diseño de Computadoras las cuales hacen uso de placas electrónicas de código abierto con lenguaje C++ y *open source* como Arduino para desarrollar sus clases con una orientación práctica, sin embargo, al finalizar estas asignaturas el conocimiento adquirido de esta plataforma, no se vuelve a aprovechar en futuras materias por parte de los estudiantes.

En ese sentido, en la asignatura de Procesamiento de Imágenes, el conocimiento de los estudiantes en programación Arduino, puede ser utilizado para la aplicación de algoritmos de procesamiento de imágenes, de esta manera el aprendizaje se manifiesta de una manera más cómoda para el estudiante, ya que está familiarizado con la programación.

2.3 Funcionalidades del controlador

Tomando como referencia algunos pasos de la metodología ABC-Besoins para análisis de requerimientos en sistemas embebidos se determina que se debe establecer primero las funcionalidades del controlador, que posteriormente servirá para realizar el diagrama de flujo, y finalmente fijar los componentes de hardware y software que se necesiten desarrollar para cumplir los requerimientos.

Basándose en el contexto y problemática expuesta en el punto anterior, se da como solución de componentes de hardware Arduino y software que unidos forman un controlador que tendrá las siguientes funcionalidades:

- Manejo de Archivos de imagen jpg: permita al usuario navegación y carga de archivos.
- Modificación de Pixeles de imágenes mediante algoritmos de procesamiento de imágenes
- Visualización de imágenes en paneles de LED RGB

2.4 Flujograma de funcionamiento general del controlador

A partir de las funcionalidades descritas en el punto anterior como análisis de requerimientos del controlador, se define el flujograma de la ilustración 16, que explica el comportamiento del controlador a desarrollar.

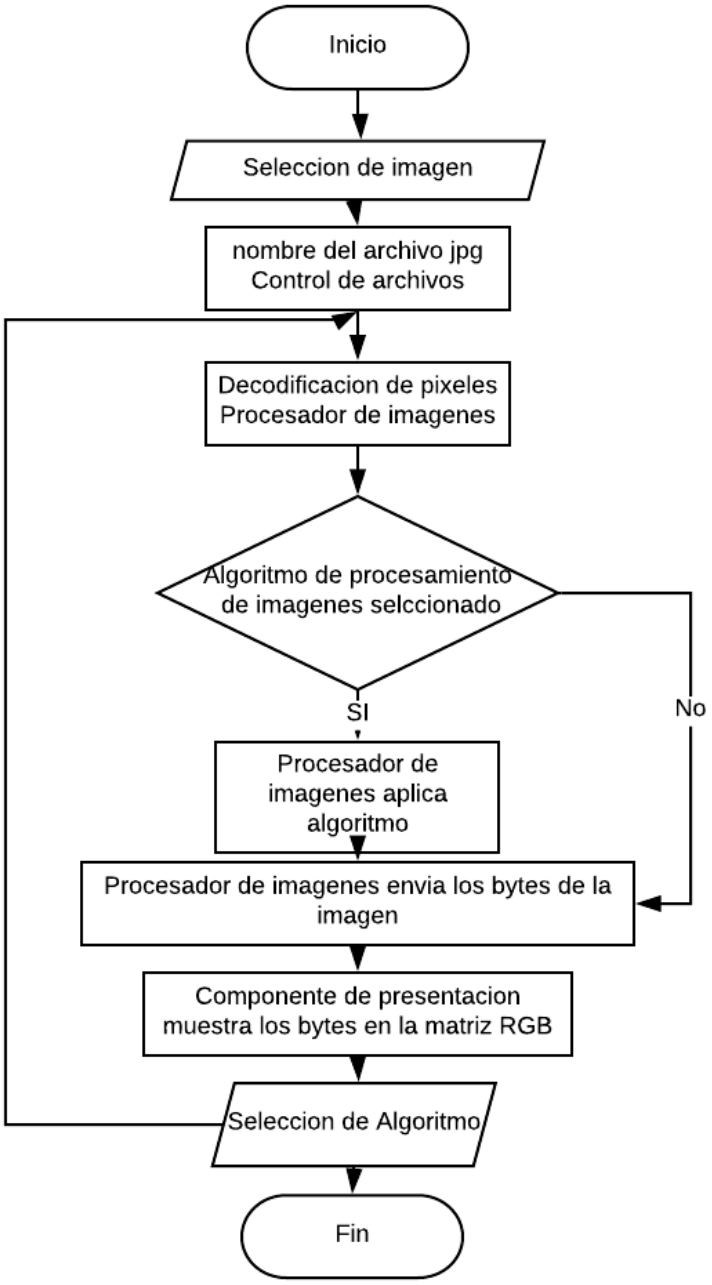

Ilustración 16:flujograma de funcionamiento general del controlador
Fuente: Cristian Cuaspa

2.5 Estructura del controlador

Para esta parte se describe la estructura de hardware y software que intervienen en el controlador, terminando así la ingeniería de requisitos.

2.5.1 Estructura de software

Para la ejecución del proceso descrito en la ilustración 16 se divide el trabajo los siguientes componentes de software:

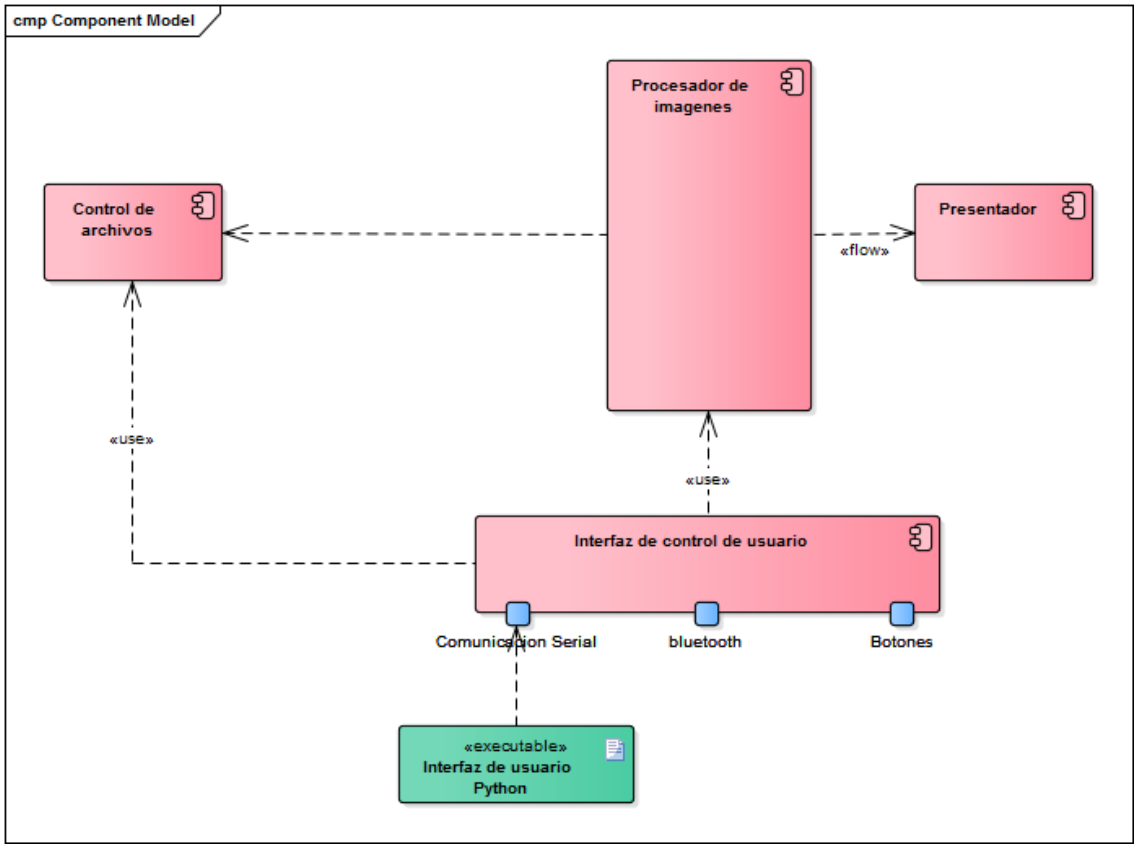

Ilustración 17:Diagrama de Componentes de software del controlador
Fuente: Cristian Cuaspa

- **Control de Archivos**: Facilita la navegación, acceso y carga de archivos de imágenes en formato jpg.
- **Procesador de imágenes**: Realiza la decodificación de archivos jpg y extrae los valores de los canales de cada pixel en un archivo jpg, para así poder modificar los valores de los pixeles mediante algoritmos de procesamiento de imágenes, y posteriormente enviarlos ya sea por comunicación serial o comunicación SPI.
- **Interfaz de control de usuario:** Permite al usuario acceder a las diferentes opciones que proporciona tanto el procesador de imágenes como el control de

archivos mediante comunicación serial, bluetooth y botones, ejemplo: siguiente imagen, aplicar algoritmo, restaurar imagen, etc.

  - **Interfaz de usuario Python**

    Aplicación que accede mediante comunicación serial a los comandos proporcionados por el componente Interfaz de control de usuario para un manejo amigable del controlador.

- **Presentación:** Toma la información enviada desde el Procesador de imágenes, y calcula la posición que tendrá cada valor (pixel) y lo muestra en las matrices led RGB.

2.5.2 Estructura de Hardware

Los componentes de software antes descritos se distribuyen en el hardware Arduino usando el concepto maestro-esclavo de tal manera que:

- **Arduino 101 (MAESTRO):** Contiene los componentes Control de Archivos, Procesador de imágenes e Interfaz de control, que trabajan en conjunto para generar los datos de una imagen, además proporciona el módulo bluetooth para la integración con la Interfaz de control de usuario.

- **Modulo SD (ESCLAVO):** Realiza el almacenamiento de las imágenes que utiliza el componente Control de archivos.

- **Arduino Mega 2560 (ESCLAVO):** Presentación de imágenes, para mostrar los datos enviados desde el maestro en forma de pixeles en un conjunto de matrices led RGB multiplexadas, por cuestiones de memoria y procesamiento se ha optado por utilizar 2 de estas placas.

- **Matriz LED RGB:** controlada por Arduino Mega 2560 para mostrar los pixeles que se envían desde Arduino 101, Arduino mega soporta la multiplexación de 2 matrices de 32x32, teniendo como resultado una sola matriz de 32x64, al final se espera mostrar una imagen en 2 paneles de 32x64 dando así una imagen cuadrada de 64x64.

2.6 Metodología del Diseño

Este proyecto se desarrolló con los principios iterativo e incremental de la metodología SCRUM, donde un incremento es presentado al cliente para comprobar la funcionalidad de una etapa del desarrollo, a la vez que se analiza los resultados obtenidos, para mejorar o agregar especificaciones al desarrollo de la etapa.

En el presente proyecto se considera como una etapa a cada componente que conforma el resultado final, priorizando el desarrollo de las etapas según la dependencia que exista entre ellas.

El resultado del desarrollo de todas las etapas debe cumplir los atributos de calidad escalabilidad con respecto a incremento de nuevos algoritmos de procesamiento de imágenes en el componente respectivo, y reusabilidad para que ciertos componentes puedan utilizarse en diferentes proyectos.

El diseño final del proyecto podrá mostrar cualquier imagen que posea las dimensiones que especifica el controlador, sin embargo, la efectividad de los resultados se comprueba por medio de la aplicación de la variante de algoritmo de compresión de rango dinámico y su extracción de histograma, antes y después de su aplicación, de tal manera que se observe la expansión de dichos histogramas de la misma imagen.

2.6.1 Definición Integración de hardware y software del controlador

Basándose en las estructuras de hardware y software antes descritas se puede establecer un diagrama de integración general de todos estos elementos, detallando a que componente de software corresponde el hardware descrito o como se relaciona, además de terceras aplicaciones que se vinculan al controlador a través de del componente de interfaz de control de usuario, esto se puede apreciar en la ilustración 17, el desarrollo o integración de cada elemento del diagrama se describe más adelante.

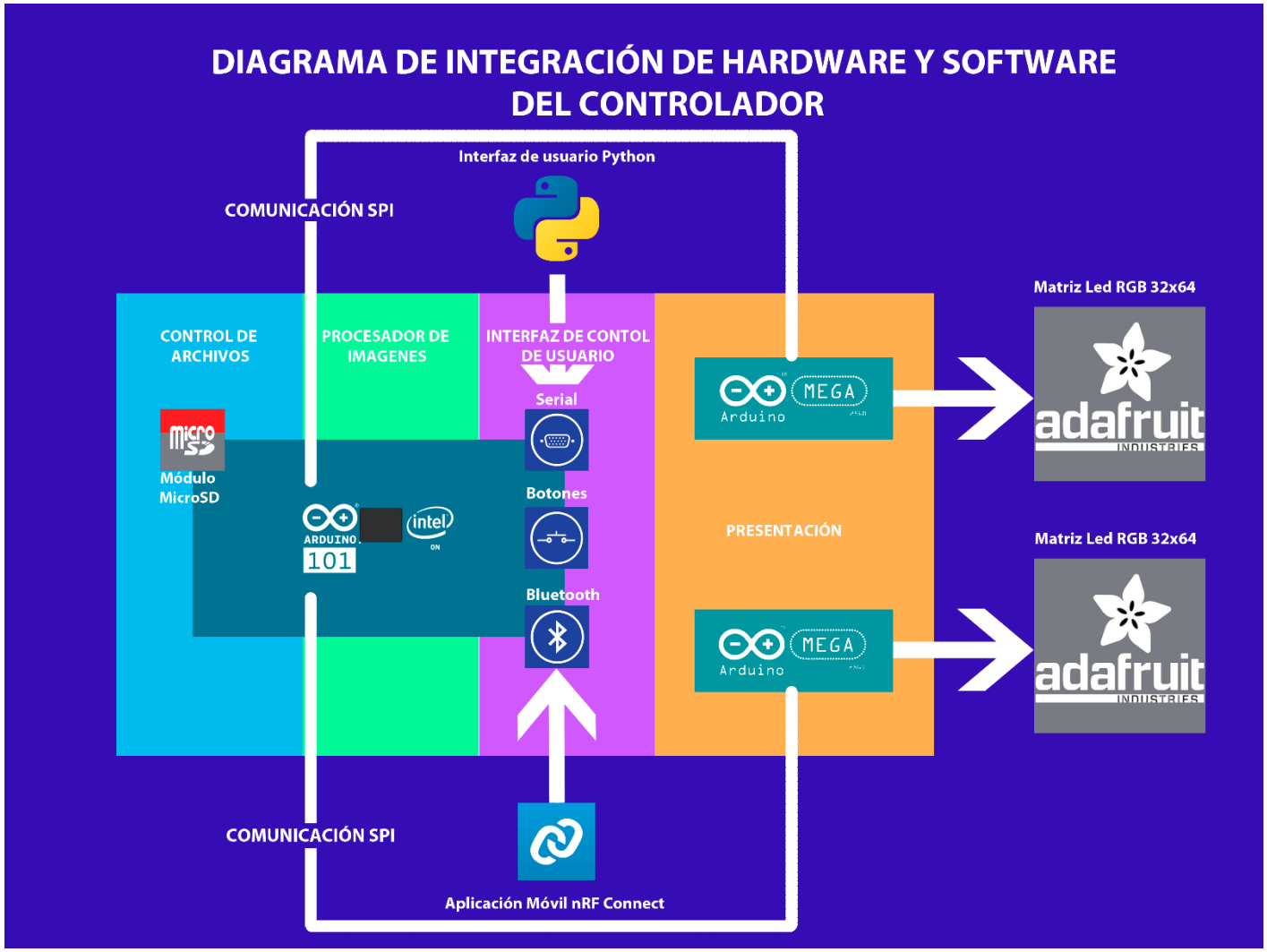

Ilustración 18: Diagrama de Integración de Hardware y software
Fuente: Cristian Cuaspa

2.6.2 Desarrollo de componentes

Cada componente de software hace referencia a una etapa de desarrollo, estos se han hecho en forma de librerías y fragmentos de código, dependiendo del número de funciones que proporcione cada componente y su dependencia con librerías externas.

Los componentes Control de Archivos y Procesador de imagen se han desarrollado como librerías propias para el proyecto, siguiendo la estructura que tienen las librerías de Arduino, esto quiere decir contiene cabecera y cuerpo, esto se logró entendiendo la estructura de la librería RGB-matrix-Panel-master, ya que contiene características de estructura similares a las que se requiere para el desarrollo de las librerías propias para el proyecto dichas características son:

- Dependencia de librerías externas
- Retorno de objeto para trabajar con una sola instancia
- Funciones públicas y privadas
- Funciones que cambian de forma

2.6.2.1 Control de archivos

Este componente se ha desarrollado en forma de librería, ya que el número de funciones que proporciona, es suficiente para optar por empaquetar el código, de tal manera que el programador pueda integrar la navegación entre archivos, simplemente incluyendo, instanciando y utilizando las funciones de la librería denominada FileControl.

a) Desarrollo de la librería FileControl.h

Esta librería hace posible la navegación y transferencia de archivos, basándose en la librería SD por ende depende también de la librería SPI de Arduino para la conexión con la tarjeta MicroSD, para proporcionar funcionalidades ya mencionadas de esta librería se desarrolló funciones que se especifican más adelante.

i) Propiedades:

- String *fileNames
  Arreglo que contiene los nombres de los archivos en una determinada carpeta

- String FolderFiles
  Nombre de la carpeta seleccionada para leer los archivos
- byte sizeElements
  Numero de archivos en la carpeta seleccionada
- byte indexImage
  position actual del nombre de archivo

ii) Métodos:

- void init(String FolderName="")
  Inicia el módulo SD y recolecta en un arreglo los nombres de los archivos que se encuentran en la carpeta con el nombre que contiene el parámetro FolderName, si no se especifica dicho parámetro, el módulo inicializara por defecto en la raíz de la Tarjeta SD
- String getCurrentFileName(void)
  Retorna la cadena de texto que se encuentra almacenada en *fileNames, en la posición indexImage
- void NextFile(void)
  incrementa en uno el valor de indexImage, para asi cuando se invoque la función getCurrentFileName el nombre de archivo sería el siguiente en la SD
- void BeforeFile(void)
  disminuye en uno el valor de indexImage, para asi cuando se invoque la función getCurrentFileName el nombre de archivo el anterior al actual
- void NextFileCircle(void)
  incrementa en uno el valor de indexImage, para asi cuando se invoque la función getCurrentFileName el nombre de archivo sería el siguiente en la SD, con la diferencia que, si al último nombre de archivo, este regresa al primero.
- void SetCurrentFile(String FileName)
  Determina la posicion indexImage dependiendo del nombre de archivo, es decir, busca el nombre de archivo igual al valor del parámetro FileName y cambia el indexImage en la posición en la que se encuentre dicho nombre
- SDImageFileControl::NewImage()

Esta función permite agregar un nuevo archivo de imagen mediante el uso de comunicación serial, en breve resumen cuando esta función es llamada, se inicia una secuencia la cual es descrita en el diagrama de flujo de la ilustración 19

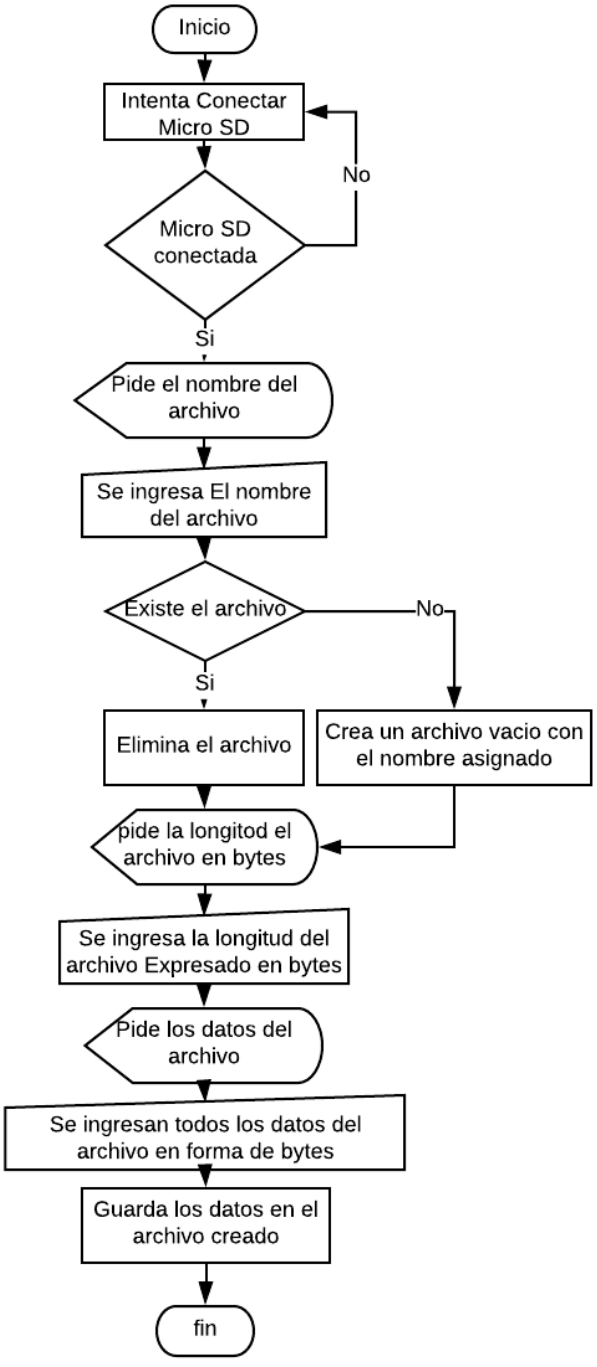

Ilustración 19: Flujograma de secuencia de transferencia de un nuevo archivo de imagen

Fuente: Cristian Cuaspa

2.6.2.2 Procesador de Imágenes

Este componente está desarrollado como librería, ya que contiene un número suficiente de métodos, para decidir empaquetar el código, esto con el objetivo de que el programador tenga la posibilidad de aplicar el procesamiento de imágenes a archivos de imagen, mediante la inclusión de la librería ImageProcesor.

a) Desarrollo de la librería ImageProcesor.h

Esta librería se encarga de decodificar los archivos jpg almacenados en la tarjeta de memoria, de manera que estos archivos puedan ser representado en un panel led RGB mediante él envió de los valores del pixel por SPI, además proporciona funciones que modifican los pixeles mediante algoritmos de procesamiento de imágenes los cuales pueden ser o no aplicados antes del envió SPI

Esta librería es independiente de la anterior, pero trabaja mejor con ella, ya que permite un manejo fácil, del manejo de los archivos.

Para el desarrollo se utiliza librerías externas que proporcionan ciertas características que son de mucha utilidad para el proyecto, tal es el caso de JPEGDecoder, que es la que permite la decodificación y manipulación de pixeles de un archivo de imagen, al igual que FileControl.h ya antes descrita, también se utiliza la librería de Arduino SPI para realizar la transferencia de datos de imagen.

Se implementa también funciones a las cuales se accede con diferente número de parámetros, esta característica se imita de la librería RGB-matrix-Panel-master que contiene constructores para diferentes tipos de matriz.

finalmente, esta librería contiene un vector de funciones, el cual contiene los apuntadores para acceder a los algoritmos de procesamiento de imágenes, esto con el objetivo de tener la posibilidad de aplicación de múltiples algoritmos en una imagen.

i) Propiedades

```
void(*processorAlgorithm[])()={dinamicRangeCorrection,grayscale,negative};
```

Este es un arreglo que contiene las referencias a los 3 algoritmos de procesamiento de imágenes de demostración:

- dinamicRangeCorrection: Calcula el histograma a partir de los pixeles y aumenta el rango dinámico de la imagen, multiplicando cada pixel por un factor, determinado por el histograma de la imagen.
- grayscale: Modifica el pixel, asignándole el promedio de sus valores: R, G, B.
- negative: Resta los valores de los pixeles, con el valor máximo 255.

ii) Funciones

- void startProcesingSPI(String FileName)
  Decodifica un archivo jpg que contenga como nombre el valor del parámetro FileName, y envía la información de los pixeles por comunicación SPI.
- void startProcesingSPI(String FileName,int uniqueReference)
  Decodifica un archivo con el nombre del valor del parámetro FileName, y aplicando el algoritmo de procesamiento que se encuentra en la posición del valor uniqueReference del arreglo processorAlgorithm
- void startProcesingSPI(String FileName, T(&AlgorithmReferences)[N])
  Decodifica un archivo con el nombre del valor del parámetro FileName, y aplicando los algoritmos de procesamiento que el encuentran en las posiciónes de los valores AlgorithmReferences del arreglo processorAlgorithm
- uint16_t Color888(uint8_t r, uint8_t g, uint8_t b);
  Funcion Extraida de la librería RGB-matrix-Panel-master para convertir los 3 bytes de un pixel al formato 565

b) Variante de compresión de rango dinámico

Para el desarrollo de la variante del algoritmo de compresión de rango dinámico se debe entender que el objetivo de este algoritmo es estirar el histograma de una imagen, en una imagen en escala de grises expande el histograma de manera que se puede apreciar niveles de gris que antes no se podían ver como en la ilustración 19.

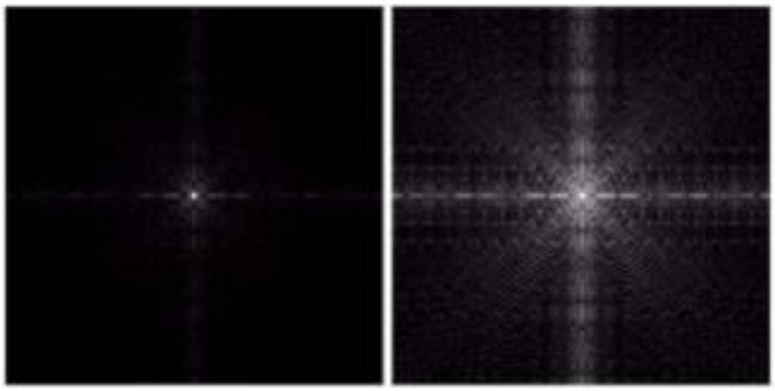

Ilustración 20: Espectro de Fourier y el resultado de una transformación logarítmica con c = 1
Fuente: (Rafael C. Gonzalez, 2018)

Se puede apreciar una expansión del histograma de cada imagen de la ilustración 20, esto mediante la aplicación del algoritmo normal de compresión de rango dinámico, pero este es válido para imágenes en escala de grises y lo que se busca es comprimir el rango dinámico de una imagen a color.

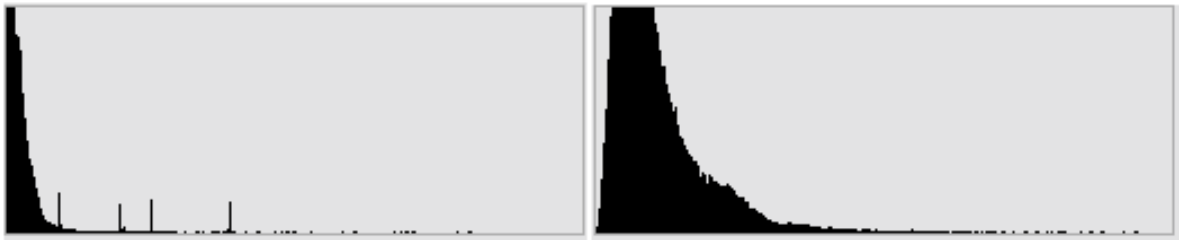

Ilustración 21: histograma de las imágenes de la ilustración 20
Fuente: Cristian Cuaspa

Lo que se busca en la variante de este algoritmo es que expanda los histogramas de los 3 canales de color de una imagen, teniendo así por ejemplo una imagen inicial como en la ilustración 22. En la cual sus histogramas están agrupados, dando como resultado que la imagen no tenga colores reales.

Ilustración 22:Imagen sin corrección de rango dinámico en sus canales
Fuente: m.genbeta.com

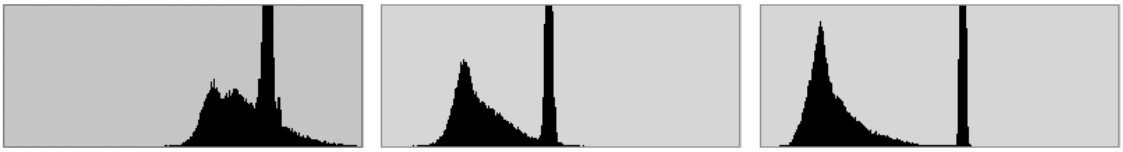

Ilustración 23:Histogramas de los canales Rojo, Verde y Azul de la imagen de la ilustración 22
Fuente: Cristian Cuaspa

Se espera como resultado de la aplicación de la variante de algoritmo de compresión de rango dinámico una imagen como la de la ilustración 24 en la que los histogramas de cada canal de color están expandidos

Ilustración 24: Imagen con corrección de rango dinámico en sus canales
Fuente: Cristian Cuaspa

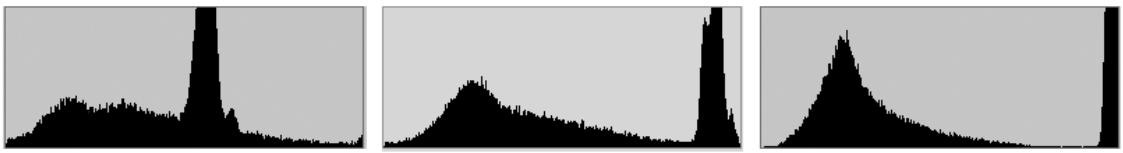

Ilustración 25:Histogramas de los canales Rojo, Verde y Azul de la imagen de la ilustración 24

Para realizar este algoritmo hay que entender que, si se suma un valor fijo a los canales de cada pixel, hará que la imagen se aclare, es decir, el histograma se desplaza hacia la derecha, sucederá lo contrario si se resta un valor fijo.

Pero no basta con desplazar el histograma, además hay que expandirlo para obtener un contraste mayor, esto se logra multiplicando cada canal por un valor fijo, tomando en cuenta que entre más alto sea el valor, mayor será el contraste.

Los valores a sumar o restar y multiplicar se obtienen generando un histograma por cada canal contando cuantos puntos existen en los tonos de color en el rango 0-255 por cada

canal de color RGB, lo que traducido a programación quiere decir 3 vectores de 256 valores en los cuales se almacena un valor en cada posición.

Se obtiene los rangos máximo y mínimo de cada histograma, y se resta a cada canal de los pixeles de la imagen, a esto se multiplica el factor para que cubra todos los tonos en el histograma este factor se obtiene dividiendo 255 que es el valor máximo en un histograma para la resta entre el valor máximo y mínimo, todo lo mencionado se resume en la ecuación 2.

Ecuación 2: Expansión de histograma

$$valor\ de\ canal\ de\ pixel = (valor\ de\ canal\ de\ pixel - minimo) * (\frac{255}{maximo - minimo})$$

Fuente: extraído del tema equilibrio de color de m.genbeta.com

Finalmente se aplica esta ecuación a cada valor de los canales de cada pixel de la imagen.

2.6.2.3 Presentador de Imágenes

El componente de presentación a diferencia de los anteriores, es un fragmento de código que cumple una única función, la de mostrar bytes enviados por comunicación SPI en un par de matrices de led RGB de 32x32.

Este fragmento de código recibe bytes desde el componente procesador de imágenes y calcula la posición en la matriz RGB y muestra el color determinado por el byte recibido haciendo uso del as funciones de la librería oficial de matrices LED RGB de adafruit: RGB-matrix-Panel-master, que ha sido modificada para que sea compatible con comunicación SPI.

2.6.2.4 Interfaz de control de usuario

Este módulo es el más sencillo de todos, ya que solo vincula eventos con las funciones de los componentes Control de Archivos y Procesador de imágenes, formando un comando, para proporcionar al usuario un manejo sencillo del controlador, esto puede ser mediante cualquier medio que produzca un evento, ejemplo: Botones físicos, Comunicación Serial, Comunicación bluetooth, comunicación Wireless, etc.

Este componente proporciona comandos vinculados a las librerías desarrolladas para este proyecto, por medio de sus métodos, como se describe en la tabla 6:

Tabla 6: Pila de comandos que proporciona el componente interfaz de control de usuario

Comando	**Descripción**	**Funciones de FileControl.h**	**Funciones de ImageProcessor.h**
next	Siguiente	NextFile(), getCurrentFileName()	startProcesingSPI(String FileName)
before	Anterior	BeforeFile(), getCurrentFileName()	startProcesingSPI(String FileName)
applyrc	Aplicar algoritmo de corrección de rango dinámico	getCurrentFileName()	startProcesingSPI(String FileName,int uniqueReference)
applyn	Aplicar algoritmo de negativo de una imagen	getCurrentFileName()	startProcesingSPI(String FileName,int uniqueReference)
applyg	Aplicar algoritmo de escala de grises de una imagen	getCurrentFileName()	startProcesingSPI(String FileName,int uniqueReference)
restore	Restaurar imagen	getCurrentFileName()	startProcesingSPI(String FileName)

Fuente: Cristian Cuaspa

a) Interfaz de usuario Python

Esta es una utilidad creada para facilitar el manejo del controlador al usuario, usando el lenguaje Python, esta aplicación se conecta a Arduino por medio del puerto serial, haciendo uso del componente Interfaz de control de usuario, envía comandos para que el componente los interprete y ejecute una acción.

Para el desarrollo de esta aplicación se utilizó las librerías Serial, PIL Y WX del lenguaje Python necesarias para que la aplicación cumpla con su objetivo, las funciones que cumplen cada librería se detallan a continuación:

- **Serial:** utilizada para realizar la conexión con Arduino y el envío de comandos al mismo.
- **PIL:** se usa para el manejo de imágenes, permite la carga, recorte y rotación de imágenes.
- **WX Python:** esta librería proporciona la interfaz gráfica para Windows, es decir, se encarga de mostrar las ventanas y controles.

2.6.3 Conexión

2.6.3.1 Conexión SPI

En este proyecto la conexión SPI se utiliza tanto para la comunicación entre Arduino 101(Maestro) con Arduino Mega 2560(Esclavo), como para la comunicación del módulo Micro SD con Arduino 101

a) **Conexión Arduino 101- Arduino Mega 2560**

Tabla 7:Conexion SPI entre Arduino 101 y mega 2560

PINES	ARDUINO 101 PINS	ARDUINO MEGA 2560 PINS	Cable
SCK	ICSP-3	52	
MISO	ICSP-1	50	
MOSI	ICSP-4	51	
CS	4,5	53	
FLAG	2	49	

Fuente: Adaptación de la tabla de conexiones SPI de las diferentes placas de Arduino (Arduino, 2017)

El pin CS es el que determina si se recibe información por SPI, por lo tanto, para la conexión con dos Arduino esclavos, se tendrá dos pines diferentes en el Arduino Maestro.

El pin FLAG establecido para el proyecto, indica fin de la comunicación, es la señal que le da el Arduino maestro al esclavo de que debe empezar a proyectar.

La conexión SPI MAESTRO-ESCLAVO puede estar como en la ilustración 25, las conexiones de voltaje y tierra pueden variar ya que existen varios de estos pines, cabe destacar que la conexión es de múltiples esclavos y los cables parten de un pin a diferentes destinos.

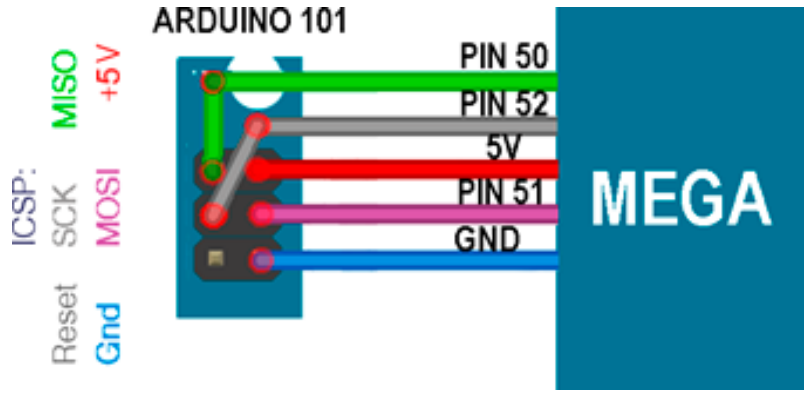

Ilustración 26:Conexion SPI Maestro esclavo entre placas Arduino
Fuente: Cristian Cuaspa

2.6.3.1.1 Conexión SPI Arduino- Modulo Micro SD

Tabla 8: Conexión SPI Arduino 101 con modulo micro SD

Pines Arduino 101	Modulo SD	Cable
13	SCK	
11	MOSI	
12	MISO	
10	SS	

Fuente: Adaptación de la tabla de conexiones SPI de las diferentes placas de Arduino y modulo micro SD (Arduino, 2017)

La conexión de un módulo MicroSD puede variar de acuerdo al modelo, pero se muestra la utilizada en el proyecto en la ilustración.

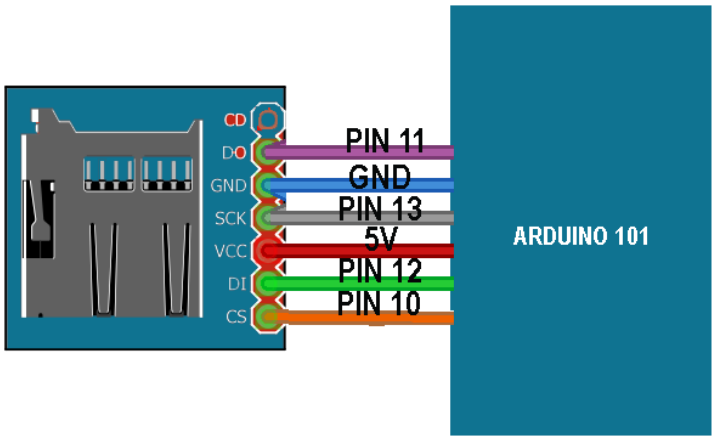

Ilustración 27:Conexión SPI Arduino-Modulo SD
Fuente Cristian Cuaspa

2.6.3.2 Conexión Arduino Mega 2560 y Matrix Panel RGB 32x32

Para la conexión entre Arduino mega y una matriz se utiliza la documentación oficial de la página de Adafruit, la cual explica el proceso de conexión, iniciando por la alimentación de las matrices.

2.6.3.2.1 Alimentación

En los diferentes modelos de matrices LED RGB puede variar la conexión de fuente de alimentación como se muestra en la ilustración 28.

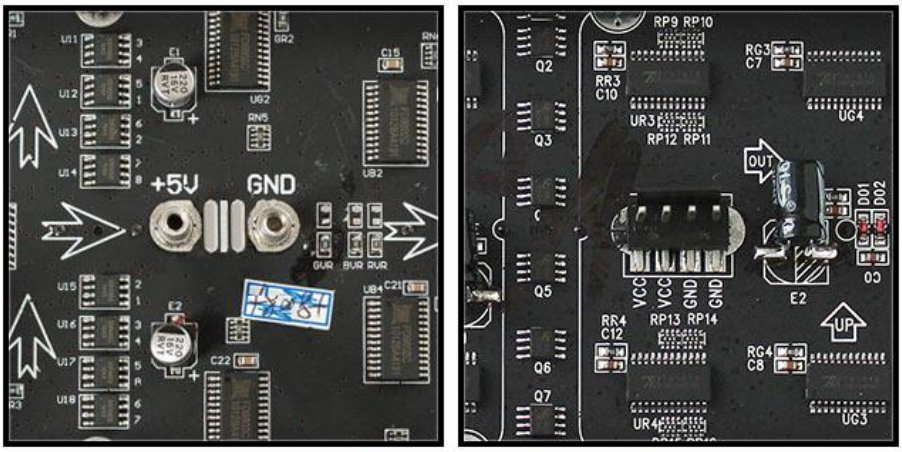

Ilustración 28:Matriz Led RGB parte posterior mostrando pines de alimentación
Fuente: extraído de página oficial de aprendizaje de Adafruit https://learn.adafruit.com

El modelo que se utiliza para el presente proyecto es el de la derecha, facilitando la conexión ya que se utiliza un cable con socket como el de la ilustración 27 para alimentar la matriz.

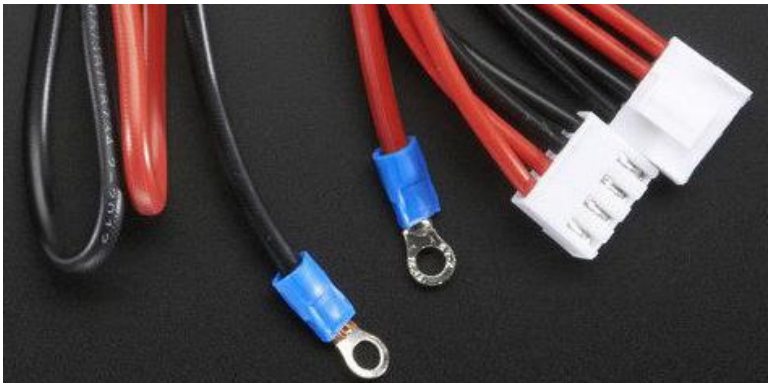

Ilustración 29:cables de alimentación matriz LED RGB
Fuente: extraído de página oficial de aprendizaje de Adafruit https://learn.adafruit.com

Estos cables deben ser conectados a una fuente de alimentación de 5 voltios como máximo 1500 miliamperios por cada matriz.

2.6.3.2.2 Conexión Arduino

Para la conexión de una matriz LED RGB hay que conocer los pines que tiene la matriz en la parte posterior como en la ilustración 30.

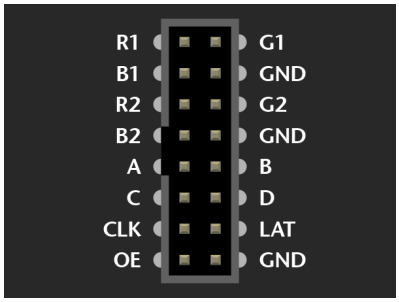

Ilustración 30:Pines de conexión de matriz led RGB
Fuente: extraído de página oficial de aprendizaje de Adafruit https://learn.adafruit.com

Utilizando el cable de datos de la matriz como en la ilustración 31 se puede conectar directamente a Arduino.

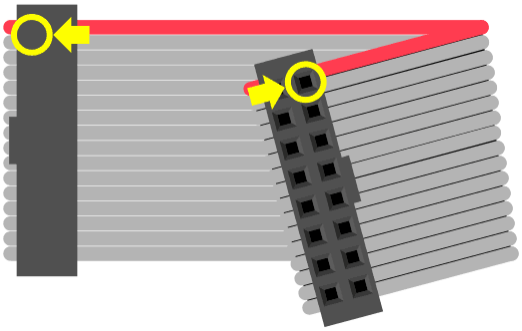

Ilustración 31: Cable de datos de matriz led RGB
Fuente: extraído de página oficial de aprendizaje de Adafruit https://learn.adafruit.com

El socket debe ser colocado en la matriz de manera que calce la muesca que tiene el socket del cable.

Para la conexión de la matriz con Arduino es necesario saber que en el cable de datos los pines se invierten, tal y como se muestra en la ilustración 32.

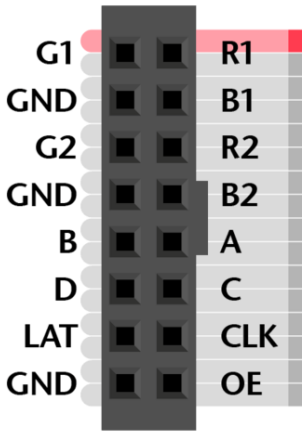

Ilustración 32: Pines del cable de datos para conexión con Arduino
Fuente: extraído de página oficial de aprendizaje de Adafruit https://learn.adafruit.com

Finalmente, para la conexión de pines del socket con Arduino se utiliza la información de la tabla 9.

Tabla 9: Conexión de Socket de matriz led RGB y Arduino Mega 2560

Socket Matriz	Arduino Mega 2560 pines	Cable
R1	24	
G1	25	
B1	26	
R2	27	
G2	28	
B2	29	
A	A0	
B	A1	
C	A2	
D	A3	
CLK	11	
LAT	10	
OE	9	

Fuente: Adaptado de la información obtenida de la página oficial de aprendizaje de Adafruit https://learn.adafruit.com

Las conexiones de tierra no se consideran en la tabla 5 ya que puede variar conforme a la preferencia del usuario.

2.6.4 Ensamblaje

Para el ensamblaje se optó por montar una estructura en acrílico blanco, a la cual se colocaron los paneles led RGB por medio de tornillos milimétricos, y las placas Arduino fueron sujetadas a la estructura con cinta doble faz.

2.6.4.1 Recorte de láminas de acrílico

Para que la estructura quede firme se utilizó dos láminas de acrílico las cuales tienen perforaciones y cortes como se muestra en la ilustración 33

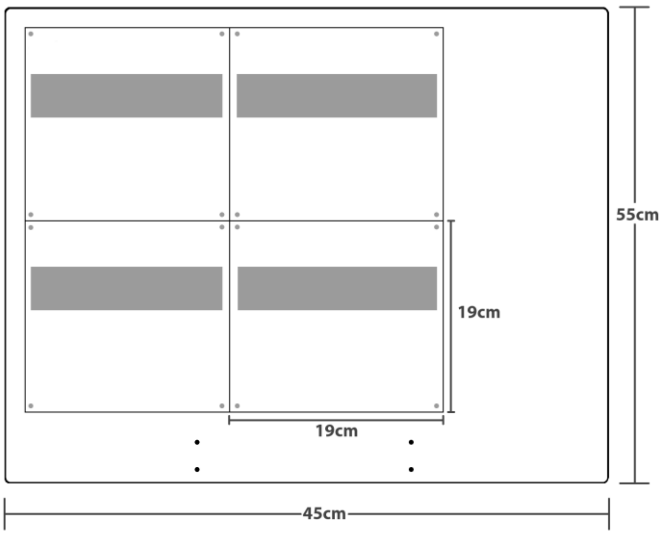

Ilustración 33: Estructura de acrílico con perforaciones y recortes
Fuente: Cristian Cuaspa

La estructura ha sido medida y perforada de manera que los tornillos milimétricos calcen con las matrices led RGB con dimensiones de 190.5 mm x 190.5 mm, en cuanto a los recortes, estos han sido realizados para que los cables de alimentación y datos puedan pasar libremente, el recorte y perforación de las láminas de acrílico se las realizo con cortadora laser especializada.

2.6.4.2 Soporte para la Estructura

El soporte para esta estructura ayuda a que las láminas de acrílico se mantengan en forma vertical, tal como un televisor, para ello se utilizó una especie de patas de televisor llamadas peanas tal y como se muestra en la ilustración 34

Ilustración 34:Peana de televisor utilizada para sostener la estructura en vertical
Fuente: extraída de la página de compras www.ocompra.com

2.6.4.3 Ajuste de elementos a la estructura

Para que los paneles y las peanas queden sujetos la estructura se utilizó tornillos de 4 milímetros, que son los que calzan en las matrices led RGB, puestos en cada perforación de la estructura, Finalmente, para que las placas Arduino queden sujetas en la parte derecha de la estructura, se utilizó cinta doble faz.

Ilustración 35:Resultado del ensamblaje
Fuente: Cristian Cuaspa

La ilustración 35 muestra el resultado final del ensamblaje dando una especie de pantalla que puede ser colocada en cualquier superficie plana, sin que tenga que sostenerla alguien.

CAPÍTULO III: RESULTADOS

Como parte de los resultados expuestos en este capítulo se tiene las pruebas que se realiza a cada componente, según su dependencia entre ellos, cubriendo así las pruebas de funcionamiento de cada elemento del controlador.

3.1 Pruebas Unitarias

Para la comprobación del funcionamiento de cada parte del controlador, se ha considerado pequeñas pilas de pruebas para cada componente, cabe recalcar que esto va abarcando la unión de los componentes para realizar las pruebas de uno en específico de acuerdo a la dependencia que existe entre ellos.

3.1.1 Control de Archivos

Para realizar las pruebas de funcionamiento del componente de control de archivos se utiliza la comunicación serial, este primer módulo no depende de ningún otro, es así que se puede testear sin la unión de otros componentes.

Tabla 10:Pila de Pruebas para el componente Control De archivos

Prueba	Descripción	Función	Resultado
FC-1	Archivo Siguiente	NextFile() y getCurrentFileName()	Nombre del archivo
FC-2	Archivo Anterior	BeforeFile() y getCurrentFileName()	Nombre del archivo
FC-3	Archivo Siguiente con repetición	NextFileCircle()	Nombre del archivo
FC-4	Selección manual del archivo	SetCurrentFile(String FileName) y getCurrentFileName()	Nombre del archivo
FC-5	Agregar nueva imagen	NewImage()	Archivo copiado a MicroSD

Fuente: Cristian Cuaspa.

Para la aplicación de estas pruebas se programa un pequeño sketch en Arduino denominado FileControl_Test, y compilado para Arduino 101 como se muestra en la ilustración 36, la prueba “Agregar nueva imagen” debe ser realizada con un software especial que se lo muestra más adelante.

```
#include <FileControl.h>
SDImageFileControl SDI;
void setup() {
  Serial.begin(9600);
  SDI.init("images");
}
void loop() {
  SDI.NextFile();
  Serial.println(SDI.getCurrentFileName());
  SDI.BeforeFile();
  Serial.println(SDI.getCurrentFileName());
  SDI.NextFileCircle();
  Serial.println(SDI.getCurrentFileName());
  SDI.SetCurrentFile("LENA.JPG");
  Serial.println(SDI.getCurrentFileName());
  delay(2000);
}
```

Ilustración 36: Código de prueba FileControl_Test
Fuente: Cristian Cuaspa

Cuando se aplica este codigo a arduino 101, se obtiene los resultados en el monitor serial, como en la ilustracion 37

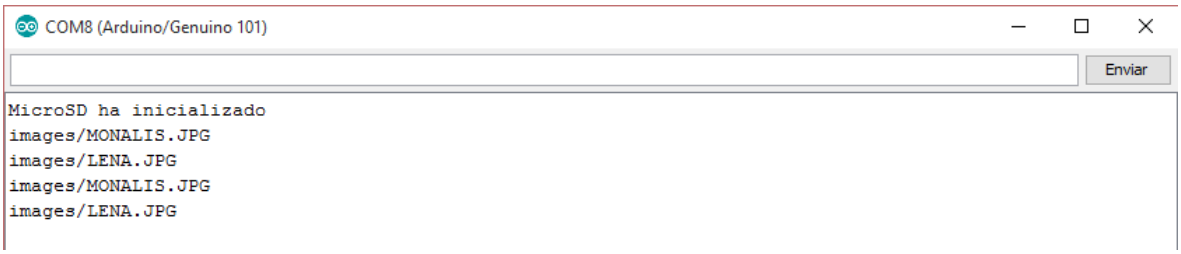

Ilustración 37: Resultado de las Pruebas FC-1, FC-2, FC-3 y FC-4
Fuente: Cristian Cuaspa

Como se puede observar, al realizar las pruebas el resultado obtenido es el mismo que el de la pila de pruebas, determinando así que las pruebas son satisfactorias.

En cuanto a la Prueba FC-5 Se necesita un programa especial desarrollado en Python, que sigue la secuencia de la función NewImage() que trabaja netamente con comunicación serial, la interfaz del programa se muestra en la ilustración 36

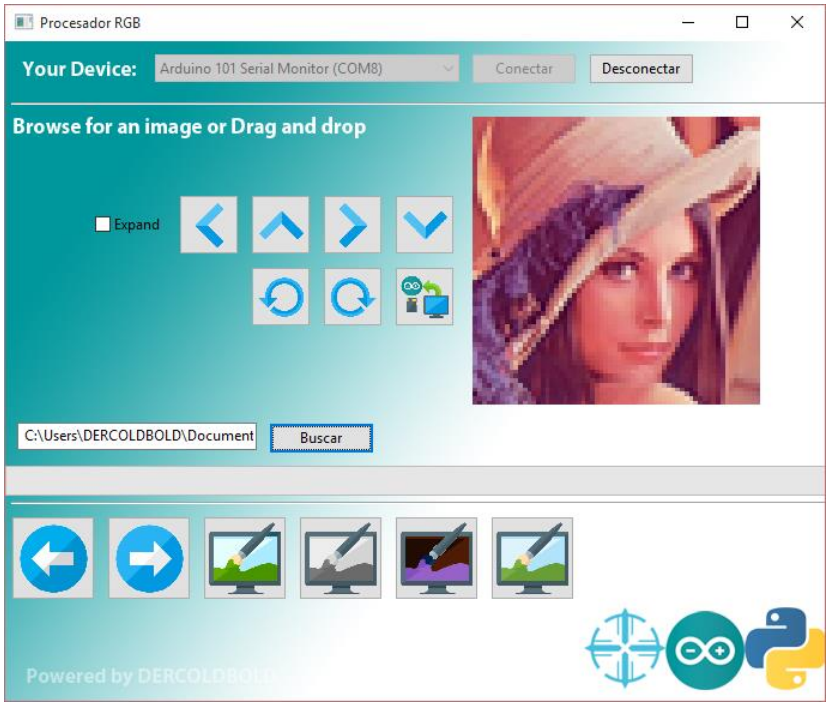

Ilustración 38:Programa para transferencia de archivos a MicroSD con Arduino
Fuente: Cristian Cuaspa

La ilustración 39 muestra la salida que genera la ejecución de la función NewImage tal como se descrita en el Capítulo 2

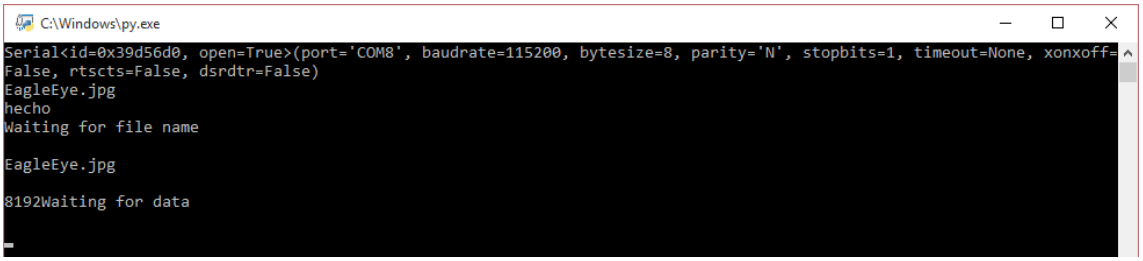

Ilustración 39:Salida de ejecución de la función NewImage

Fuente: Cristian Cuaspa

Finalmente se hace la comprobación de que la imagen efectivamente se haya copiado a la micro SD.

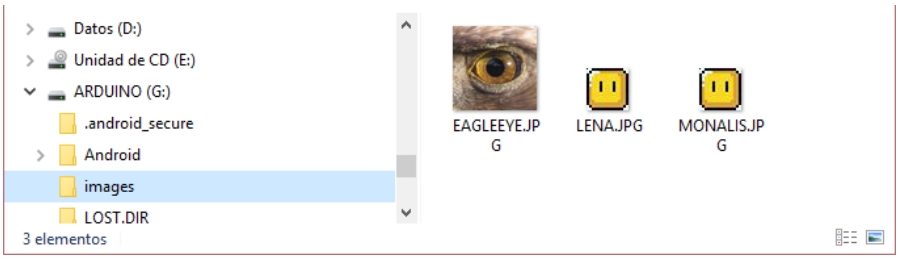

Ilustración 40: Comprobación de copia de imagen a la MicroSD
Fuente: Cristian Cuaspa

Como se puede observar en la ilustración 40 el archivo de imagen se ha copiado con el nombre especificado en la ilustración 39, determinando así que el componente pasa la prueba FC-5 de la pila de pruebas.

3.1.2 Procesador de imágenes

Para el procesador de imágenes se establece también una pila de pruebas con las funciones específicas del componente, pero debido a la dependencia que existe con el componente control de archivos se toman funciones del mismo para realizar las pruebas.

Tabla 11:Pila de pruebas para el componente Procesador de imágenes

Nombre de prueba	**Descripción**	**Función**	**Respuesta o resultado**
PI-1	Mostrar imagen normal	startProcesingSPI(String FileName)	Pixeles de imagen normal
PI-2	Mostrar imagen en escala de grises	startProcesingSPI(String FileName, int AlgorithmReference)	Pixeles de imagen en escala de grises
PI-3	Mostrar Negativo de una imagen	startProcesingSPI(String FileName, int AlgorithmReference)	Pixeles de imagen en negativo
PI-4	Mostrar imagen con corrección de rango dinámico	startProcesingSPI(String FileName, int AlgorithmReference)	Pixeles de imagen con corrección de rango dinámico

Fuente: Cristian Cuaspa

Para la comprobación de estas pruebas se necesita interpretar los valores de los pixeles de la imagen, es por eso que se ha optado por utilizar un pequeño programa desarrollado en Python, para mostrar la imagen resultante.

En la ilustración 42 se puede apreciar la interpretación de los pixeles generados por el procesador de imágenes, sin hacer ninguna modificación en los pixeles originales de la imagen guardada en la micro SD, con esto se determina que el procesador de imágenes cumple con la prueba PI-1.

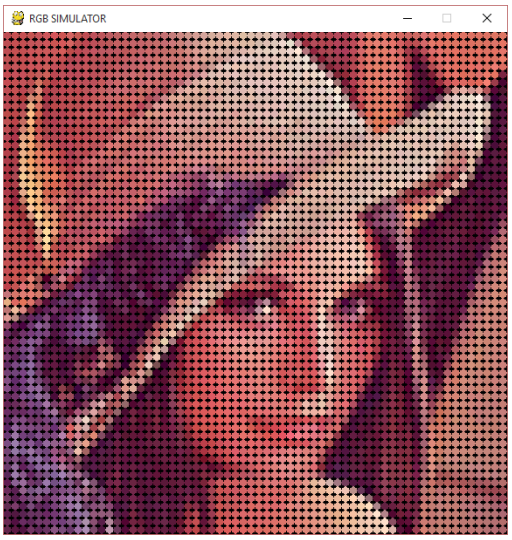

Ilustración 41:Interpretacion de los pixeles generados por el procesador sin algoritmo
Fuente Cristian Cuaspa

En la ilustración 43 se puede observar una imagen en escala de grises, estos pixeles han sido generados por el procesador de imágenes, Aplicando el algoritmo de escala de grises, se determina que cumple con la prueba PI-2 de la pila de pruebas.

Ilustración 42:Interpretacion de los pixeles generados por el procesador con algoritmo de escala de grises
Fuente: Cristian Cuaspa

La siguiente ilustración 44 muestra los pixeles de la imagen modificados con un algoritmo de negativo de imagen, generados por el procesador de imágenes, determinando así que cumple con la prueba PI-3 de la pila de pruebas.

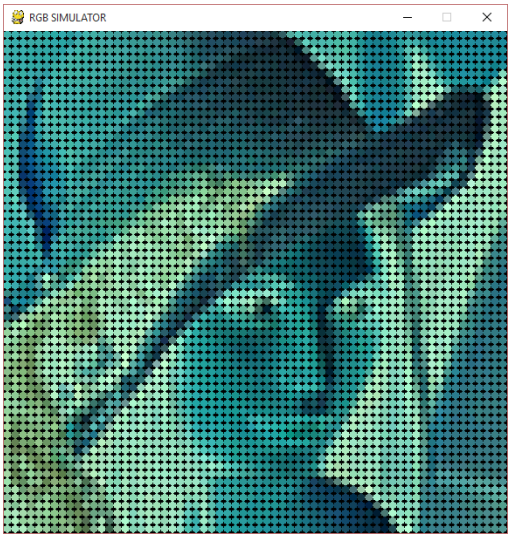

Ilustración 43:Interpretacion de los pixeles generados por el procesador con algoritmo de negativo
Fuente: Cristian Cuaspa

Finalmente, la prueba más importante PI-4 de la pila de pruebas, se determina que cumple con el resultado esperado, ya que en la ilustración 45 se muestra una imagen con corrección de rango dinámico, y se puede notar la mejora y equilibrio del color de la imagen.

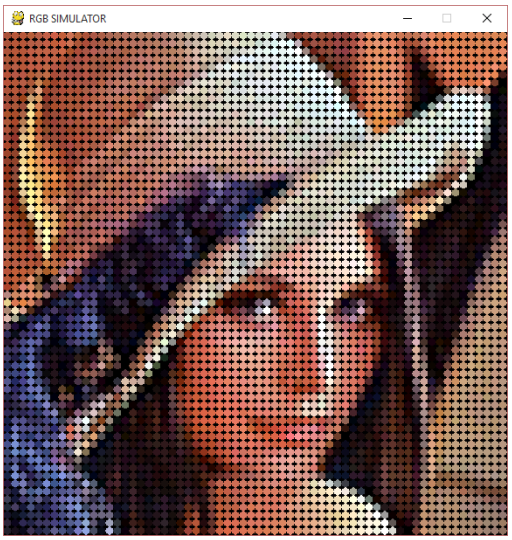

Ilustración 44:Interpretacion de los pixeles generados por el procesador con algoritmo de corrección de rango dinámico
Fuente: Cristian Cuaspa

3.1.3 Presentación

Para comprobar la funcionalidad del componente de presentación, no es necesario realizar una pila de pruebas ya que lo único que hay que probar es que se proyecte una imagen en la Matrix Led RGB, para esto se necesita utilizar los componentes Control de Archivos y Procesador de imagen, ya que depende de estos para que se le envié información de una imagen y este componente la interprete.
Dicha comprobación se la puede apreciar en la ilustración 46 que muestra una imagen presentada en 2 matrices led RGB

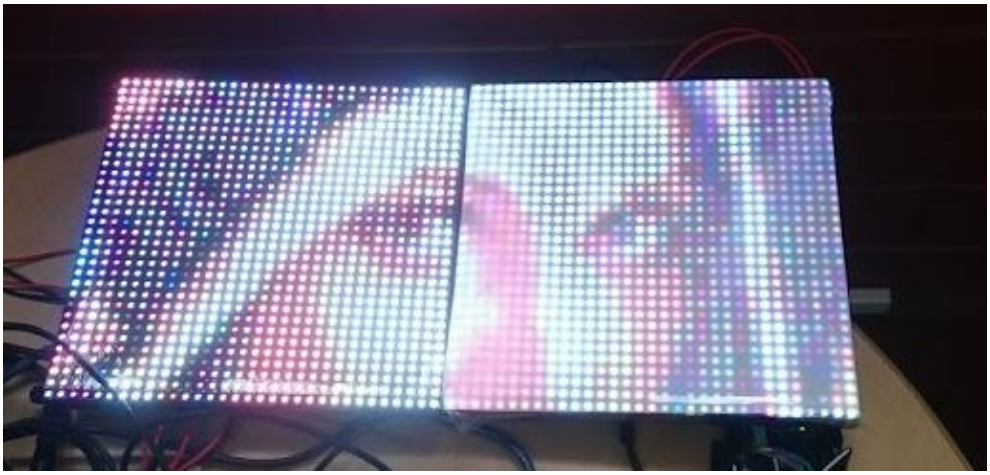

Ilustración 45:Prueba de componente de presentación
Fuente: Cristian Cuaspa

3.1.4 Interfaz de control de usuario

Finalmente, La interfaz de control de usuario estuvo presente en todas las pruebas unitarias de cada módulo, ya que proporciona el medio con el cual se da las órdenes a Arduino, es por eso que no existe una pila de pruebas para este componente, el hecho de estar presente y funcionar en las pruebas anteriores determina que su funcionamiento es satisfactorio.

Se puede también determinar las pruebas de usuario de la aplicación Python son satisfactorias ya que, para aplicar las pruebas del componente de procesamiento de imágenes, se utilizó esta aplicación, sin embargo, es necesario describir las funcionalidades de esta aplicación.

Funciones que son estáticas en la aplicación y solo modifican la imagen localmente.

- Girar a la derecha

Ilustración 46:Evidencia de prueba girar a la derecha
Fuente: Cristian Cuaspa

- Girar a la izquierda

Ilustración 47:Evidencia de prueba girar a la izquierda
Fuente: Cristian Cuaspa

- Recortar a la derecha

Ilustración 48:Evidencia de prueba recortar a la derecha
Fuente: Cristian Cuaspa

- Recortar hacia la izquierda

Ilustración 49:Evidencia de prueba recortar a la izquierda
Fuente: Cristian Cuaspa

- Recortar hacia arriba

Ilustración 50:Evidencia de prueba recortar hacia arriba
Fuente: Cristian Cuaspa

- Recortar hacia abajo

Ilustración 51:Evidencia de prueba recortar hacia abajo
Fuente: Cristian Cuaspa

Funciones que hacen uso de comandos de la interfaz de control para que esta ejecute una acción.

- Enviar imagen a Arduino: está vinculada al comando New de la interfaz de control de usuario
- Imagen Anterior: está vinculada al comando before de la interfaz de control de usuario
- Siguiente Imagen: está vinculada al comando next de la interfaz de control de usuario
- Corrección de rango dinámico: está vinculada al comando applyRC de la interfaz de control de usuario
- Imagen normal: está vinculada al comando restore de la interfaz de control de usuario
- Imagen en escala de grises: está vinculada al comando applyG de la interfaz de control de usuario
- imagen en negativo: está vinculada al comando applyN de la interfaz de control de usuario

para las pruebas con bluetooth no hay necesidad de describir los comandos que utiliza ya que son los mismos que en comunicación serial.

3.2 Pruebas de integración

Para las pruebas de integración de los componentes, se ha considerado probar el controlador completo mostrando varias imágenes en secuencia con aplicación de corrección de rango dinámico, determinando así el correcto funcionamiento del controlador, dicha prueba se puede apreciar en la ilustración.

Ilustración 52:Secuencia de imágenes con aplicación de algoritmo de compresión de rango dinámico
Fuente: Cristian Cuaspa

CONCLUSIONES

La correcta aplicación de ingeniería de requisitos permitió partir de desde una idea general de lo que necesita el usuario hasta diseñar un diagrama de componentes que ayuda al desarrollo por etapas, sin problemas de integración al final, ya que desde ese punto se definen las dependencias entre componentes, este ayuda también a determinar funciones específicas de cada componente que a partir de ello se hace fácil establecer una pila de pruebas.

Acceder a la unidad mínima en una imagen digital, plataformas especializadas es accesible ya que por mucho tiempo han evolucionado y agregado nuevas funciones para la modificación de imágenes digitales, en Arduino esto se dificulta ya que, en cuestión de imágenes digitales su principal función es como medio de presentación o captura con hardware especial, sin embargo mediante investigación bibliográfica se encontró que existe una librería denominada JPEGDecoder-master, que da la posibilidad de extraer el valor de cada pixel de una imagen, a partir de esto se logra la modificación de los valores RGB de cada canal de los pixeles de la imagen digital a partir de esto se pudo desarrollar los algoritmos de muestra y el principal que es la variante de compresión de rango dinámico.

Arduino está diseñado para funcionar de manera modular y es compatible con hardware tanto de Arduino como de otras casas comerciales, es por eso y por el diseño mismo de los componentes que la integración de hardware no es difícil, es más esto permite determinar que la interfaz de control de usuario que está directamente relacionada con el hardware que manipula el usuario es escalable, de manera que si en el futuro se requiere manejar el controlador con cualquier medio diferente de comunicación serial, esto se lo podría realizar sin problema.

La elección de comunicar los dispositivos que intervienen en el controlador con comunicación SPI es la más eficiente para este proyecto ya que se usa tanto para comunicar y transferir información del módulo MicroSD como también para comunicarse entre las placas Arduino que interviene.

La construcción de librerías permitió tener un código reusable que se pueda incluir en futuros proyectos, esto se logró mediante la comprensión de librerías de Arduino que poseen características similares a las que se necesitaba para este proyecto, las cuales ya han sido descritas en el capítulo 2 de Materiales y métodos.

Finalmente, el diseño de los componentes ayuda a determinar que son reutilizables ya que, si bien dependen de unos a otros, esto no quiere decir que no funcionen por separado en otros proyectos de Arduino, dicha dependencia solo es para el manejo fácil del controlador.

RECOMENDACIONES

Para futuras implementaciones se puede optimizar el componente de presentación utilizando Raspberry pi o Arduino Teensy ya que tienen mejores prestaciones en memoria y velocidad de procesamiento que es eficaz para para multiplexar más Matrices de pixeles al mismo tiempo.

El procesamiento de imágenes con Arduino, sin paneles RGB se puede realizar con el uso de funciones del componente de procesador de imágenes que trabajan con comunicación serial, de manera que se envían los valores de los pixeles ya sean modificados o no, y dichos valores pueden ser interpretados en cualquier lenguaje ya que la salida por seria proporciona coordenadas y valores RGB de cada pixel, de esta manera evitamos el desgaste de los componentes que intervienen en la multiplexación en etapas de desarrollo.

En cuanto al uso del controlador, se puede copiar las imágenes de tamaño 64x64 directamente en la carpeta imagen de la MicroSD, y controlar por medio de los botones, o bien utilizar la utilidad desarrollada en Python que se proporciona para este proyecto.

Para desarrollo de otros algoritmos de procesamiento de imágenes es recomendable leer primero la hoja de datos de la librería ImageProcessor desarrollada para este proyecto y que corresponde al componente Procesador de imágenes.

Se recomienda hacer investigación de aplicaciones móviles utilizando bluetooth, ya que el componente de interfaz de usuario del controlador permite fácilmente la integración con este medio.

BIBLIOGRAFIA

Adafruit ®. (2017). *Adafruit Industries, Unique & fun DIY electronics and kits*. Obtenido de https://www.adafruit.com/product/607

Adomßent, M., Barth, M., Fischer, D., Richter, S., & Rieckmann, M. (2014). Learning to change universities from within: A service-learning perspective on promoting sustainable consumption in higher education. *Journal of Cleaner Production*, 72–81.

Alan L. Vergara, H. M. (12-16 de Noviembre de 2014). *Development of an Arduino-based automated household utility power monitoring system*. Obtenido de https://ieeexplore.ieee.org/document/7016233

Arduino. (2017). *Arduino-SPI*. Obtenido de https://www.arduino.cc/en/Reference/SPI

Arduino. (2018). *Arduino 101*. Obtenido de https://store.arduino.cc/usa/arduino-101

Arduino. (2018). *ARDUINO MEGA 2560 REV3*. Obtenido de https://store.arduino.cc/usa/arduino-mega-2560-rev3

Bădică, C., Zoran , B., Hans-Dieter , B., & Mirjana , I. (2011). *Software agents: Languages, tools, platforms.* (Vol. 2). Computer Science and Information Systems.

Biblioteca de la Universidad de Sevilla. (2015). CREATIVE COMMONS PROTEGE LOS DERECHOS DE AUTOR. Sevilla, España.

Collis, B. (1996). *Tele-learning in a digital world.* Londres: Thomson Computer Press.

Contaminacion Luminica En Chile Y Latinoamerica. (2 de febrero de 2017). *Contaminación Lumínica en Chile y Latinoamérica --Pantallas LED*. Obtenido de http://pantallasled.net/contaminacion-luminica/

Cotizacion Online. (2017). *Ofertas en pantallas de LED publicitarias - Visualled.com*. Obtenido de http://www.visualled.com/precios-pantallas-led-gigantes

Cotizador en línea. (2017). *Blueink Gigantografias | Cotizador*. Obtenido de https://blueink.ec/es/cotizador

Creative Commons. (2017). *Creative Commons --Atribución-CompartirIgual 3.0 Ecuador--CC BY-SA 3.0 EC*. Obtenido de https://creativecommons.org/licenses/by-sa/3.0/ec/

Fabrice Morganti, M. O. (2017). Using ARDUINO development platform in the. *IEEE*, 1.

Galadima, A. A. (2014). *Arduino as a learning tool.* Obtenido de https://ieeexplore.ieee.org/document/6997577/

Gustavsson, S. (2014). Blue LEDs – Filling the world with new light. *THE NOBEL PRIZE IN PHYSICS 2014 ↔ THE ROYAL SWEDISH ACADEMY OF SCIENCES*, 5.

H.BOUAZZAa, b. M. (2016). New and Low Cost Product Solution for LED Matrix Display. *Conferencia Internacional de Ingeniería y MIS 2016 (ICEMIS)*.

Indexxit. (2016). *Arduino Modules-Indexxit*. Obtenido de http://indexxit.com/arduino-sd-card/

Johan Jarnestad/The Royal Swedish Academy of Sciences. (2014). Blue LEDs – Filling the world with new light. *The Royal Swedish Academy of Sciences*, 5.

Juan Francisco Ordóñez, L. O. (2016). Detector de bordes de imágenes usando un microcontrolador arm. *INGENIUS Revista de Ciencia y Tecnologia*, 6.

Klašnja-Milićević, A., Vesin, B., Ivanović, M., & Budimac, Z. (2011). *E-Learning personalization based on hybrid recommendation strategy and learning style identification.* (Vol. 3). Computers & Education.

M.K. Md Arshad, U. H. (2006). Characteristics of Serial Peripheral Interfaces (SPI) Timing. *Semiconductor Electronics, 2006. ICSE '06. IEEE International Conference on*.

Mario Ibarra, E. F. (2013). Codificación de autómatas en Arduino. *Octavo congreso Cientifico-tecnologico de las carreras de ingenieria*.

Rafael C. Gonzalez, R. E. (2018). *Digital Image Processing, 4th Edition.* Pearson.

Ritesh Reddy, N. K. (2017). Digital Image Processing through Parallel Computing in Single-Core and Multi-Core Systems using MATLAB. *IEEE*.

Ritesh Reddy, N. K. (2018). Digital image processing through parallel computing in single-core and multi-core systems using MATLAB. *IEEE*, 1-2.

Rodriguez, V. A. (2009). Diseño y construccion de un sistema de alimentacion para LEDs de potencia utilizando el convertidor CD/CD tipo Flyback. *UNIVERSIDAD TECNOLOGICA DE MIXTECA*, 94.

Santiago, C. A. (2005). *FORMATOS DE IMAGEN DIGITAL.* Mexico: DGSCA-UNAM .

Soto Almeida Juan, S. Q. (2009). SISTEMA DE INFORMACION VISUAL. *Escuela Politécnica Nacional Facultad de Ingeniería Eléctrica y Electrónica*.

Squids.com. (2018). *Projeto 23 - Criando efeito de cores com um LED RGB*. Obtenido de http://squids.com.br/arduino/index.php/projetos-arduino/projetos-basicos/95-projeto-23-usando-o-led-rgb

Urresta Cueva, L. F. (2010). Desarrollo de un sistema embebido con aplicaciones gráficas y capacidad USB ON-THE-GO para manejo de archivos en flash drive utilizando la tarjeta starter kit PIC24F. 9-10.

V. Tipsuwanpornl, A. N. (2014). Development of Redundant Bus Library for Arduino to Apply in SCADA System. *2014 14th International Conference on Control, Automation and Systems (ICCAS 2014)*.

Vercelli, D. A. (11 de septiembre de 2009). Guía de licencias Creative Commons.

Verdejo , F., & Gordon, D. (1998). *The Virtual Campus : Trends for higher education and training.* Chapman and Hall.

Vilet, J. R. (2005). *Apuntes de Procesamiento Digital de Imágenes.* San Luis Potosí.

Zeller, H. (Julio de 2016). *GitHub - hzeller/rpi-rgb-led-matrix: Controlling up to three chains of 64x64, 32x32, 16x32 or similar RGB LED displays using Raspberry Pi GPIO.* Obtenido de https://github.com/hzeller/rpi-rgb-led-matrix

ANEXOS

Cabecera FileControl.h

```
#include<Arduino.h>
#include <SD.h>
#include <SPI.h>
class SDImageFileControl{
    private:
        bool initModules=false;
      File Dir;
                String *fileNames;
                String FolderFiles;
                byte sizeElements=0;
                byte indexImage=0;
                void getFileNames();
    public:
                void init(String FolderName="");
     String getCurrentFileName(void);
     void NextFile(void);
                void BeforeFile(void);
                void NextFileCircle(void);
     void SetCurrentFile(String FileName);
                void NewImage();
                void ListFiles();
};
```

Cabecera ImageProcesor.h

```
#include<Arduino.h>
#include <JPEGDecoder.h>

class ImageProcesor{
    private:
                String CurrentFileName;
                String HistogramName;
                void resetHistogram(void);
                void generateHistogram(void);
                void startProcesing(String FileName, int Algorithm[], uint8_t size,
uint8_t Option);
                void startProcesing(String FileName,int uniqueReference ,uint8_t
Option);
          public:
                void init();
                void startProcesingSPI(String FileName);
                void startProcesingSPI(String FileName,int uniqueReference);
                template<typename T, int N>
```

```
		void startProcesingSPI(String FileName, T(&AlgorithmReferences)[N])
		{
			uint8_t nAlgorithms = sizeof(AlgorithmReferences) /
sizeof(AlgorithmReferences[0]);
			startProcesing(FileName, AlgorithmReferences, nAlgorithms, 0);
		}
		uint16_t Color888(uint8_t r, uint8_t g, uint8_t b);
		ImageProcesor();
};
```

GUIA GENERAL PARA EL USO DEL CONTROLADOR

Antes de iniciar la guía de uso del controlador es necesario leer ficha técnica de las librarías utilizadas para este controlador como buena práctica que todo programador debería aplicar antes de usar una librería.

INSTALACION DE LIBRERIAS NECESARIAS PARA EL CONTROLADOR

Existen 3 formas de instalar librarías en Arduino IDE:

1. Instalación desde Gestor de librerías: Es la forma más sencilla si la librería esta subida al repositorio oficial de Arduino (librerías oficiales).
2. importar librería desde un archivo ZIP: Es una forma de agregar librerías que no están en el repositorio oficial de Arduino.
3. instalación manual copia de archivos: Simplemente se copia las carpetas de la librería en el directorio destinado para almacenar librerías de Arduino.

Para esta guía se puede utilizar la segunda y tercera opción, ya que las librerías aún no están en repositorio oficial de Arduino, por lo tanto, no se encontraría en el gestor de librerías.

Paso 1: Descargar Librerías a instalar: Esos archivos zip serán proporcionados en forma de link en el medio que el docente crea conveniente, o entregados a los estudiantes directamente.

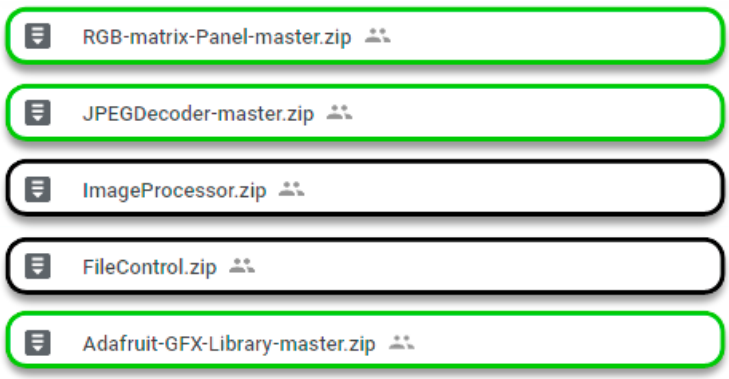

Ilustración 53: Librerías del controlador

Es necesario descargar 5 librerías, los archivos marcados con verde son librerías oficiales y se pueden encontrar en el repositorio de Arduino, pero la diferencia es que

han sido modificadas para cumplir requerimientos específicos del controlador, esto quiere decir que, si se instala del repositorio oficial, el controlador no funcionará.

Paso 2: Instalación, abra una instancia de Arduino IDE y diríjase a “Programas>Incluir Librería>Añadir Librería ZIP…”, a continuación, se busca el directorio donde se encuentran descargados los archivos ZIP correspondientes a las librerías y clic en abrir, Se repite el mismo proceso para todas las librerías del controlador como se muestra en las ilustraciones 2 y 3.

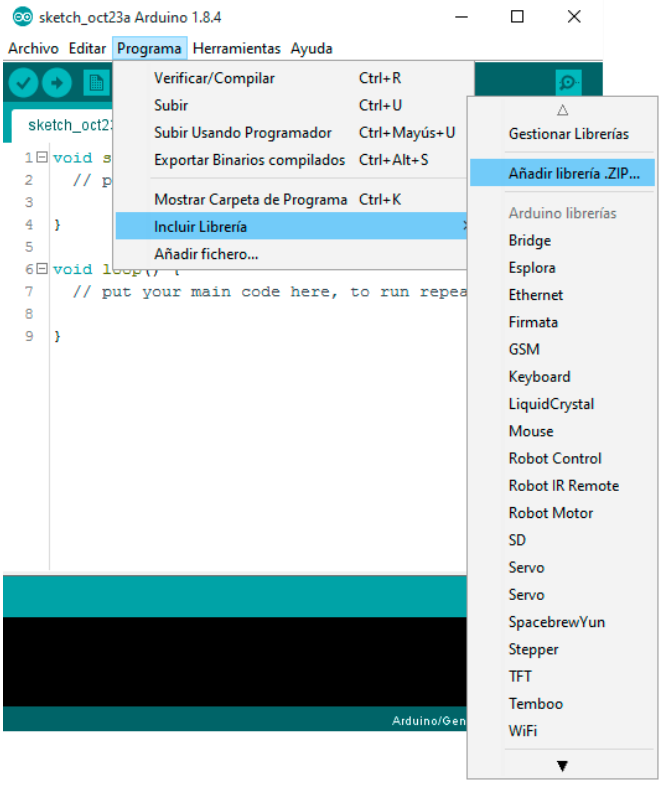

Ilustración 54:Menu Instalación de librerías ZIP Arduino

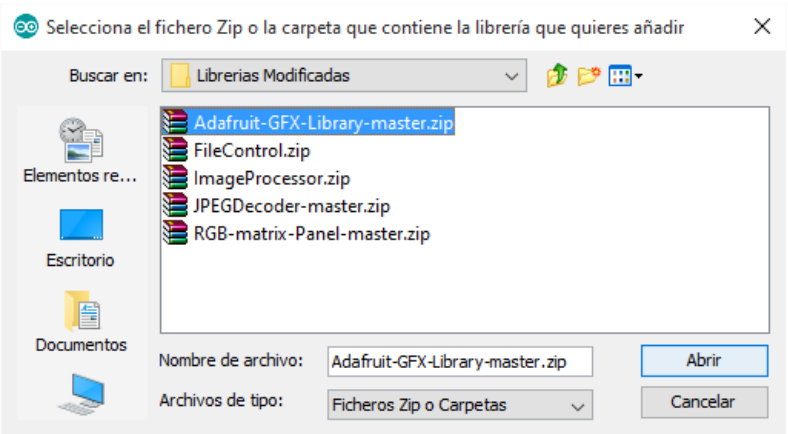

Ilustración 55: Selección de Archivo ZIP

Paso 3: Verificación, una vez instaladas las librerías se puede comprobar en Programas>Incluir Librería Como se muestra en la ilustración 4.

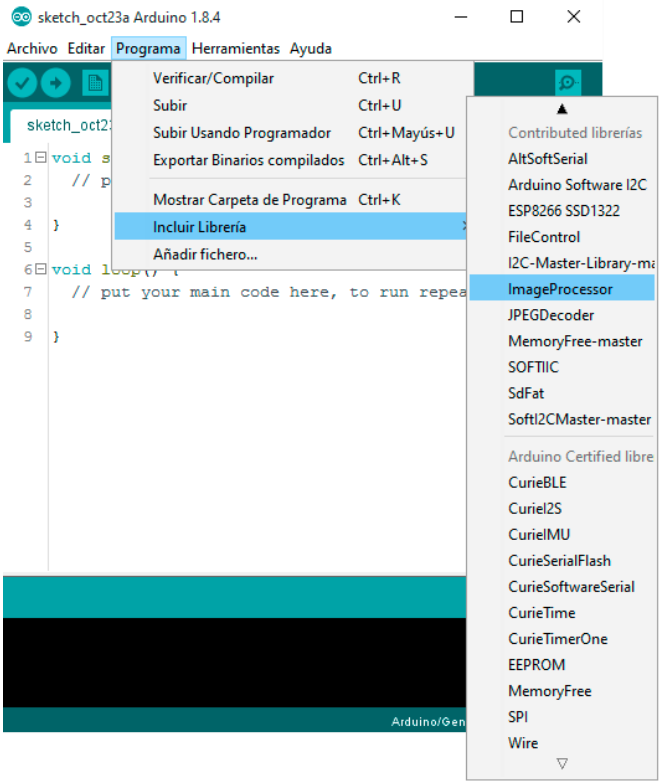

Ilustración 56: Verificación de Instalación de Librerías

CARGA DE CODIGO A ARDUINO MAESTRO Y ESCLAVOS

El controlador está compuesto por 2 tipos de placas Arduino:

1. Arduino 101 (Maestro): Encargado de realizar la decodificación y modificación de pixeles de imagen.
2. Arduino Mega 2560 (Esclavo): Encargado de la presentación en los paneles LED RGB, que por motivo de compatibilidad de librerías se utiliza dos de estas placas.

En primer lugar, es importante aclarar que para compilar código en Arduino 101, hay que instalar la placa, ya que esta no viene instalada por defecto con Arduino IDE, para eso debe dirigirse a “Herramientas> Placa> Gestor de tarjetas…”, aparecerá una ventana en la cual se deberá buscar la placa escribiendo la palabra “Intel” , cuando se hayan desplegado los resultados seleccionamos la correspondiente a Arduino 101 y en

la parte izquierda haremos clic en el botón instalar, el proceso se puede observar en las ilustraciones 5 y 6.

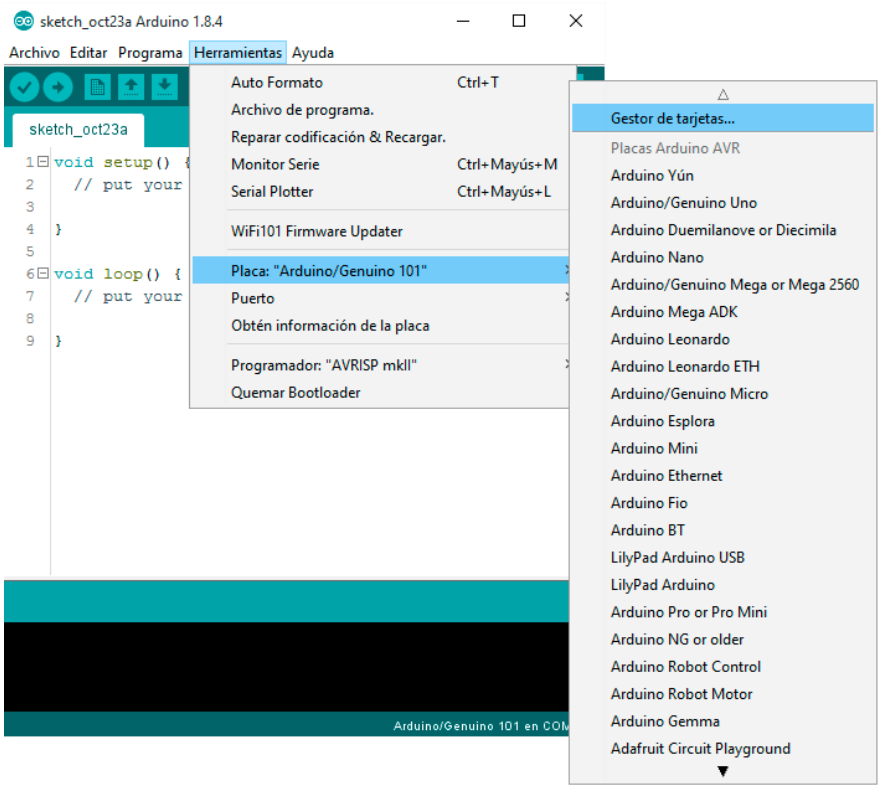

Ilustración 57:Menú instalación de placas Arduino

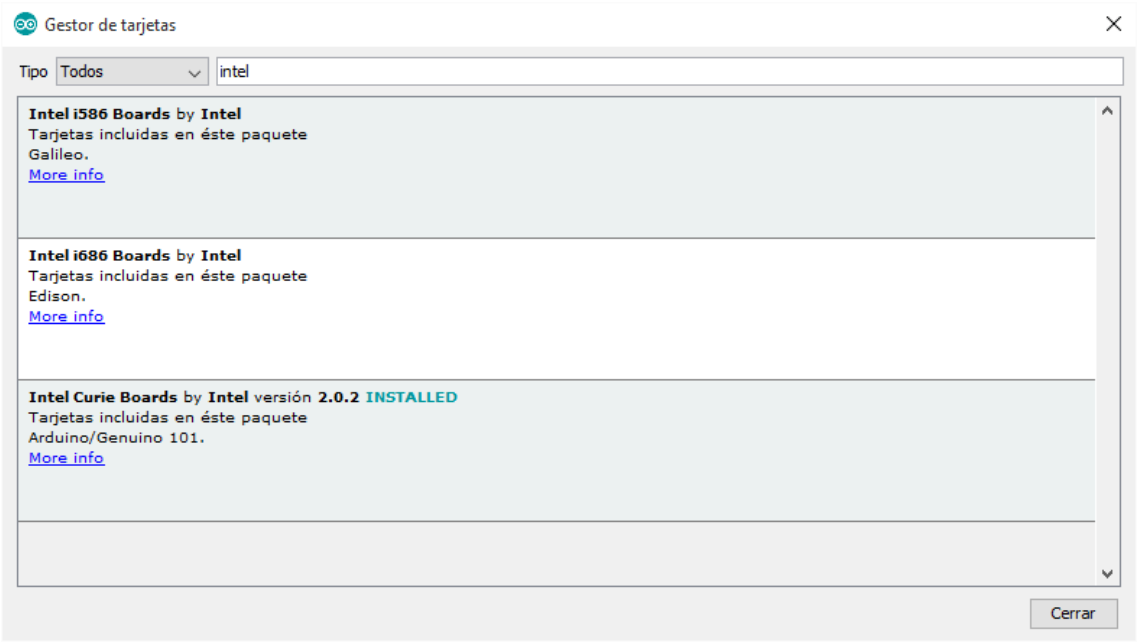

Ilustración 58:Busqueda e instalación de Placa

Una vez instalada la placa Arduino 101, ya se puede realizar la carga del código correspondiente a cada placa.

El código Arduino será proporcionado por el docente, deberá ser compilado y cargado a las placas respectivas como se indica en la ilustración 7.

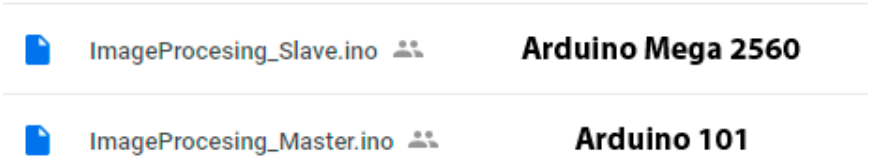

Ilustración 59: Código correspondiente a cada placa Arduino

Para cargar el código debe abrir los archivos con extensión ".ino" en el IDE de Arduino y seleccionar la placa a la cual se dirige el código, como se muestra en la ilustración 8.

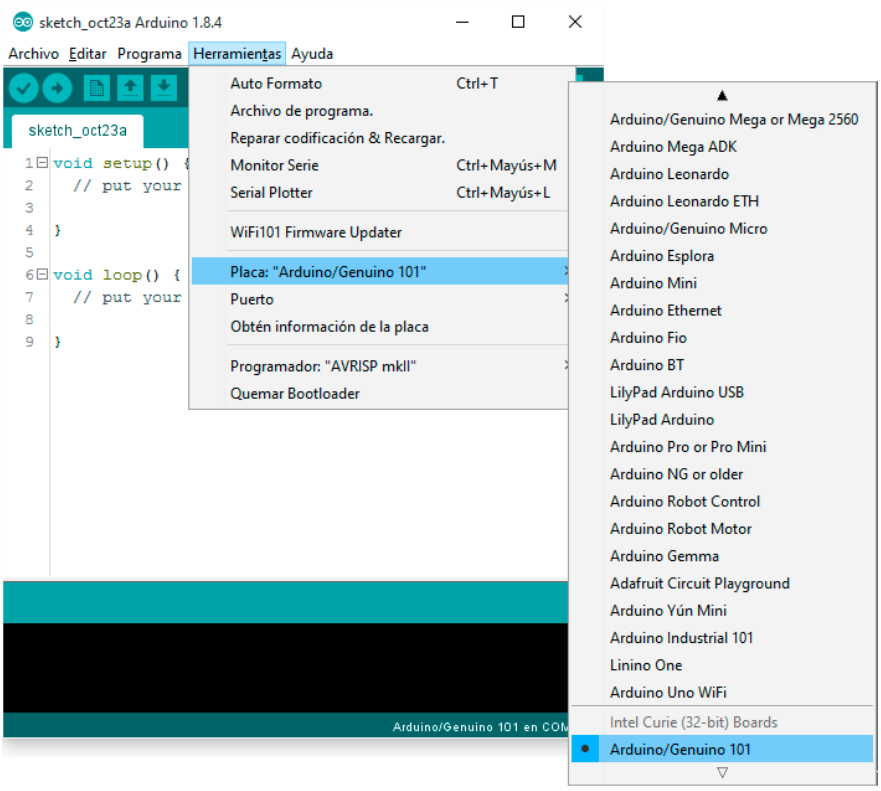

Ilustración 60: Selección de placa en Arduino IDE

Una vez seleccionada la placa, se debe compilar haciendo clic en el botón ubicado en la parte superior izquierda, luego cargar el código a la placa haciendo clic en el botón .

VERIFICACION DE HARDWARE

Inserte la MicroSD en la ranura del módulo MicroSD.

Conecte el cable USB de Arduino 101 al computador espere unos segundos a que reconozca el hardware y abra el monitor serial con velocidad 115200

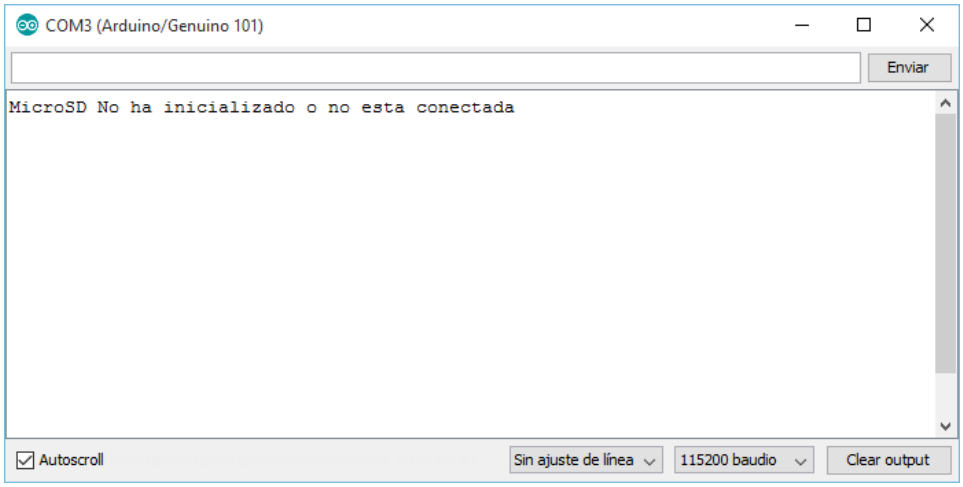

Ilustración 61:Salida de Monitor Serial

Si se muestra el mensaje de la ilustración 9 debe extraer e insertar la microSD en la ranura y ajustar la conexión del módulo microSD, cuando el monitor no muestre mensajes de error compruebe la presentación de imágenes enviando el comando “next”.

Ilustración 62:Presentación de Imágenes en paneles Led RGB

Si el controlador se ha inicializado correctamente, se debe mostrar las imágenes almacenadas en la MicroSD en la los paneles led RGB

INTERFAZ GRAFICA PYTHON

Para poder ejecutar la interfaz gráfica del controlador se debe instalar Python como lenguaje de programación y librerías externas utilizadas en la creación de la interfaz, para ello se debe seguir los siguientes pasos

1. Obtener Instalador de Python: se puede descargar la última versión de Python en la página oficial https://www.python.org/ en la sección Downloads y seleccionando su sistema operativo.
2. Ejecutar Instalador: cuando haya obtenido el instalador de realizar una instalación típica y seleccionar el checkbox de añadir PATH a Windows como se muestra en la ilustración 11

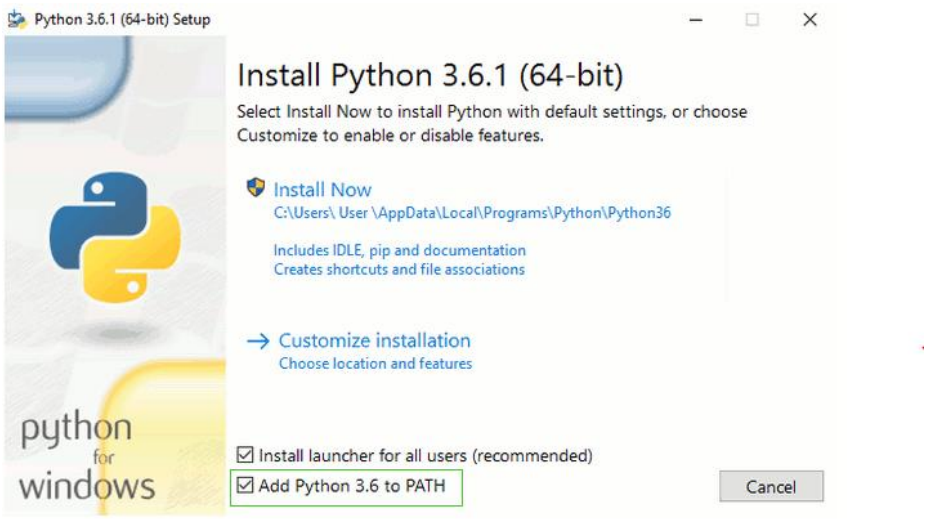

Ilustración 63: Instalación de Python ventana principal

3. Comprobación de instalación: para verificar la correcta instalación de Python se debe dirigir a una consola de símbolo del sistema e ingresar el comando “python –version” como en la ilustración 12:

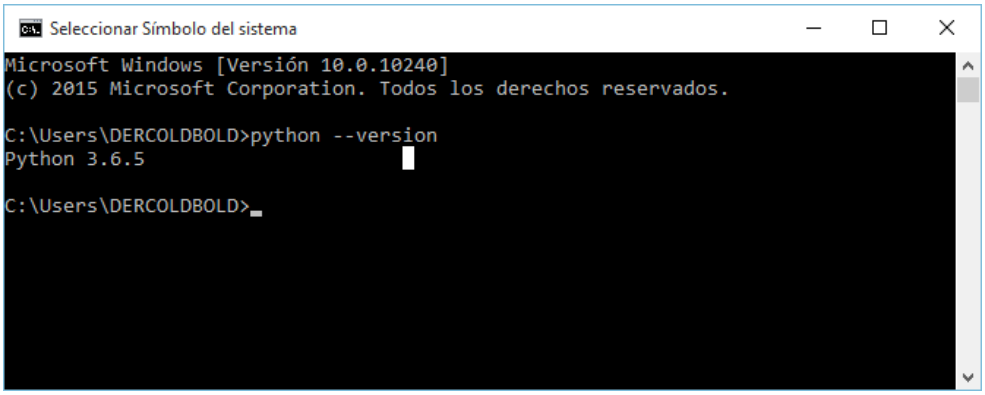

Ilustración 64:Ventana de comandos de windows

4. Instalación de librerías, ejecute uno a uno los siguientes comandos:

python -m pip install --upgrade pip

pip install PIL

pip install wxPython

pip install pygame

EJECUCION DE INTERFAZ GRAFICA

Una vez instalados todos los componentes necesarios, ya se puede ejecutar la interfaz gráfica del controlador, para eso se debe seguir los siguientes pasos.

1. Obtener Interfaz gráfica, el docente le debe proporcionar el medio para descargar la carpeta “Interfaz de control” o entregarla directamente al estudiante.

2. Ejecución De Interfaz: Una vez obtenida la carpeta, debe encontrar el archivo “Interfaz Grafica.py” y hacer doble clic, este le mostrara una ventana como en la ilustración 13

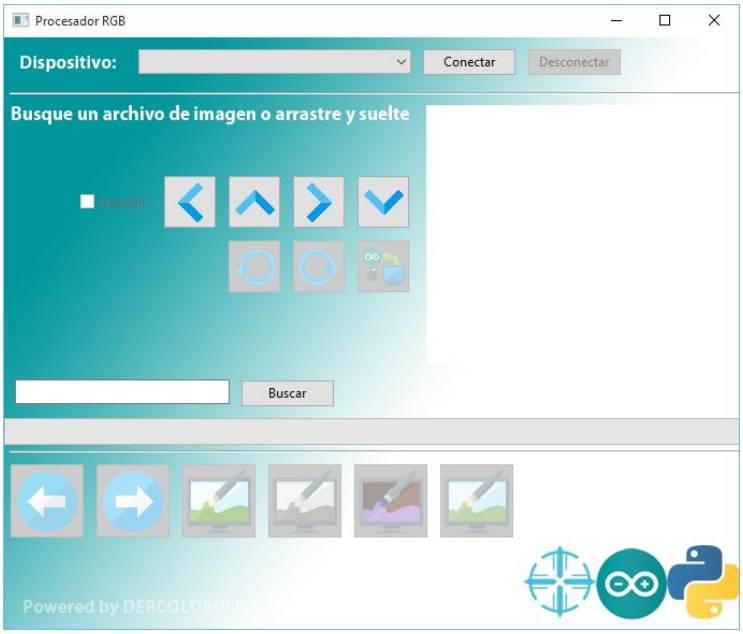

Ilustración 65:interfaz de control

3. Conectar Arduino 101 (Master): para que la interfaz de control pueda acceder a los comandos que proporciona internamente el controlador debe asegurarse de cerrar todas las ventanas de monitor serial y luego elegir el dispositivo en el dropDownList de la ventana, tal como se muestra en la ilustración 14.

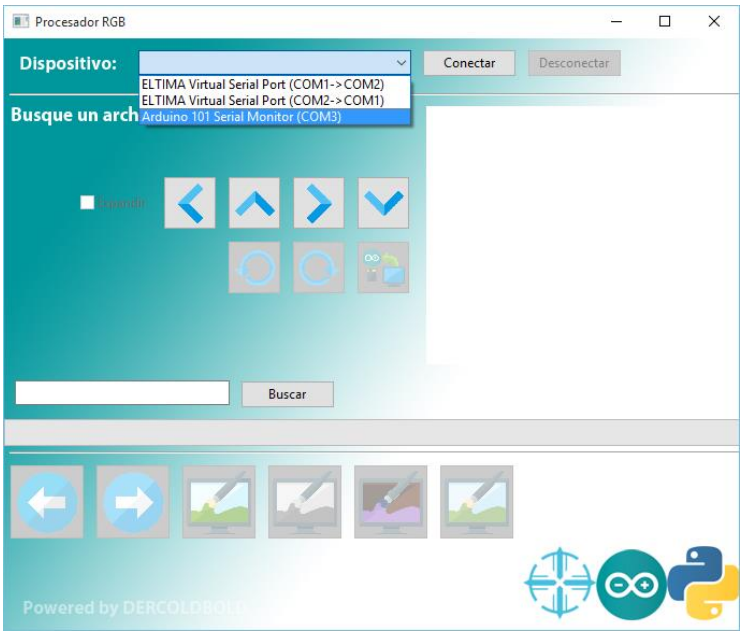

Ilustración 66:Seleccion de dispositivo

Una vez seleccionada la placa, hacer clic en conectar

USO DE INTERFAZ GRAFICA

Para el uso de la interfaz gráfica es necesario saber qué hace cada uno de sus componentes los se dividen en las secciones: conexión, edición de imagen y comandos.

1. Conexión: Permite seleccionar, conectar y desconectar el Arduino 101 en el puerto que esté conectado.

2. Edición de imagen: Permite seleccionar una imagen y realizar una edición básica con los botones y cargarla a la MicroSD conectada a Arduino.

Los elementos de esta sección se describen a continuación:

- Expandir: Estira la imagen en dimensiones de 64x64

- Recortes: Da la posibilidad de realizar recortes de una imagen en las direcciones básicas

- Rotación y carga: Permite rotar la imagen hacia la izquierda o derecha, y cargar el archivo a la Micro SD conectada a Arduino.

- Selección de imagen: puede buscar la imagen a cargar en un directorio haciendo clic en el botón buscar.

- Previsualización: Permite ver como quedara la imagen antes de cargarla a la MicroSD conectada a Arduino.

3. Comandos: Esta sección permite enviar comandos por comunicación serial, para que Arduino los interprete.

Los botones representan los siguientes comandos:

Tabla 12:Comandos de controlador

Comando	**Descripción**
next	Siguiente
before	Anterior
applyrc	Aplicar algoritmo de corrección de rango dinámico
applyn	Aplicar algoritmo de negativo de una imagen
applyg	Aplicar algoritmo de escala de grises de una imagen
restore	Restaurar imagen

Fuente: Cristian Cuaspa

ERRORES COMUNES

El buen funcionamiento del controlador puede verse afectado por errores de conexión mayormente de la MicroSD, se describen errores que pueden suceder al momento de manejar el controlador y como solucionar y volver a ponerlo en marcha.

Fallas de reconocimiento de MicroSD:

Es común que al arrancar muchas veces no se reconozca la MicroSD esto se puede verificar abriendo un monitor serial en Arduino IDE como se muestra en la ilustración 9, en el caso de que la ventana muestre el mensaje “MicroSD no ha inicializado o no está conectada” debe conectar y desconectar la MicroSD del módulo correspondiente y ajustar la conexión cableada que va desde el módulo a Arduino de la siguiente manera

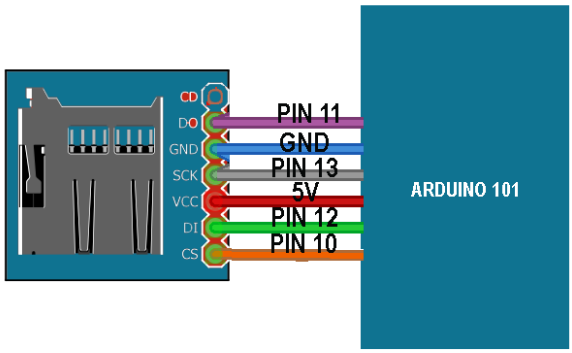

Ilustración 67:Conexion SPI Arduino-101 a modulo MicroSD

Error en la carga de imágenes desde la interfaz de control

Cuando se carga una imagen desde la interfaz de control y la imagen no se representa en los paneles se debe a que el nombre del archivo que intenta subir tiene más de 8 caracteres o lleva caracteres especiales.

Para solucionar este error realice los siguientes pasos:

- Paso 1: Cierre la interfaz de control y reinicie Arduino 101.
- Paso 2: Cambie de nombre el archivo de imagen que intenta subir quitando los caracteres especiales y reduciendo los caracteres.
- Paso 3: Finalmente vuelva a conectar Arduino 101 con la interfaz de control y vuelva a cargar la imagen.

Falla en la conexión con interfaz de control

Siempre que la conexión de la interfaz de control con Arduino 101 falle o no inicie, se debe verificar que no haya ningún monitor serial abierto, si existe una conexión abierta se debe proceder a cerrar e intentar nuevamente la conexión.

Falla de representación en los paneles Led RGB

Normalmente sucede cuando uno de los paneles no está conectado a la corriente altera, para solucionar este error se debe verificar las fuentes de energía de los paneles, luego reiniciar Arduino Mega 2560 y Arduino 101 en ese orden.

Falla de comunicación de Arduino con Monitor Serial

En el caso de que requiera enviar comandos directamente desde el monitor serial a Arduino 101 y este no responda, se debe verificar que la velocidad debe estar configurada en “115200”.

Urkund Analysis Result

Analysed Document: TRAB_TITULACION CUASPA CRISTIAN 04FEB19.pdf (D47580277)
Submitted: 2/4/2019 4:19:00 PM
Submitted By: ldnarvaez@pucesi.edu.ec
Significance: 3 %

Sources included in the report:

TESIS-FB.docx (D40967629)
TESIS-FB2.docx (D41081619)
TESIS PROTOTIPO DE CHALECO INTELIGENTE urkund.docx (D12288106)
TESIS-FB1.docx (D41058497)
TRABAJO DE PIS concluido.docx (D41087779)
URKUND-CAPITULO-I-II-III-IV-TESIS-GPS-JEFFERSON-SILVA.docx (D40822290)
https://electro2ar.wordpress.com/2014/06/19/comunicacion-i2c-entre-arduinos/
https://www.adafruit.com/product/607
https://ieeexplore.ieee.org/document/7016233
http://pantallasled.net/contaminacion-luminica/
http://squids.com.br/arduino/index.php/projetos-arduino/projetos-basicos/95-
https://github.com/hzeller/rpi-rgb-led-matrix

Instances where selected sources appear:

19

Pontificia Universidad Católica del Ecuador
SEDE IBARRA

Ibarra 21 de enero del 2019

Asunto: Aceptación y conformidad de entrega de módulo de proyecto

Doctora

Laura Rosa Guerra Torrealba

COORDINADORA DE TITULACIÓN

De mis consideraciones

Por medio de la presente me permito informarle que el Sr. Cristian Javier Cuaspa Benavides, estudiante de la Escuela de Ingeniería, ha entregado el proyecto "Controlador de procesamiento de imágenes basados en plataformas open source destinado a la visualización en paneles de leds RGB" , mismo que forma parte del proyecto tipo "A" denominado "Diseño de un prototipo de visualización 2D en paneles led RGB mediante la ejecución de la fase de emulación de técnicas de procesamiento digital de imágenes".

Comunico a Usted la Aceptación y conformidad con la entrega del proyecto ya que ha cumplido con los objetivos establecidos para el mismo.

Es todo cuanto puedo informar

Cordialmente

(f:)……
Mgs. Luis David Narváez Erazo
C.C.: 100286837-8