ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

PROYECTO DE GRADO PARA LA OBTENCIÓN DEL TÍTULO EN INGENIERÍA ELECTRÓNICA

AUTORES: CLAVIJO VILLAVICENCIO, EDWIN GONZALO
PÉREZ CARRILLO, RICARDO DAVID

TEMA: DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE ROTULACIÓN RGB BASADO EN LA TÉCNICA DE PERSISTENCIA DE LA VISIÓN

DIECTOR: ING. SEGOVIA, XAVIER
CODIRECTOR: ING. TIPÁN, EDGAR

SANGOLQUÍ, NOVIEMBRE 2013

CERTIFICACIÓN

Certificamos que el presente proyecto de grado titulado "Diseño e implementación de un sistema de rotulación RGB basado en la técnica de persistencia de la visión", ha sido desarrollado en su totalidad por el Sr. Edwin Gonzalo Clavijo Villavicencio con CI 171999627-2 y el Sr. Ricardo David Pérez Carrillo con CI 171921458-5, bajo nuestra dirección

Ing. Xavier Segovia

DIRECTOR

Ing. Edgar Tipán

CODIRECTOR

DECLARACIÓN DE RESPONSABILIDAD

Edwin Gonzalo Clavijo Villavicencio

Ricardo David Pérez Carrillo

Declaramos que:

El proyecto de grado titulado "Diseño e implementación de un sistema de rotulación RGB basado en la técnica de persistencia de la visión", ha sido desarrollado con base a una investigación exhaustiva, respetando derechos intelectuales de terceros, conforme las citas que constan al pie de las páginas correspondientes, cuyas fuentes se incorporan en la bibliografía.

Consecuentemente este trabajo es de nuestra autoría.

En virtud de esta declaración, nos responsabilizamos del contenido, veracidad y alcance científico del proyecto de grado en mención.

Edwin Gonzalo Clavijo Villavicencio

Ricardo David Pérez Carrillo

AUTORIZACIÓN

Nosotros, Edwin Gonzalo Clavijo Villavicencio y Ricardo David Pérez Carrillo

Autorizamos a la Universidad de la Fuerzas Armadas – ESPE la publicación, en la biblioteca virtual de la Institución, el proyecto de grado titulado "Diseño e implementación de un sistema de rotulación RGB basado en la técnica de persistencia de la visión" cuyo contenido, ideas y criterios son de nuestra exclusiva responsabilidad y autoría.

Edwin Gonzalo Clavijo Villavicencio

Ricardo David Pérez Carrillo

DEDICATORIA

El presente proyecto de grado lo dedico a mi amada familia por ser el incentivo e inspiración para lograr cumplir esta meta, sin el sacrificio, apoyo, fuerza, palabras de aliento y sobre todo el amor que ellos me brindan cada día no hubiese sido posible llegar a este punto de mi vida. Todo el trabajo empleado para cumplir este sueño es por ustedes y para ustedes.

Ricardo Pérez C.

DEDICATORIA

El trabajo desarrollado en esta tesis y sobre todo el esfuerzo puesto en los estudios durante toda mi vida estudiantil se lo dedico a toda mi familia que siempre creyeron en mí y supieron guiarme para alcanzar mis metas propuestas en la vida, a ustedes por siempre mi corazón y mi agradecimiento.

Edwin Clavijo V.

AGRADECIMIENTO

Especialmente agradezco a mis abuelitos Mami Fabi y Papi Victor quienes son un ejemplo de humildad, trabajo y fortaleza en mi vida.

A mi padre Ricardo Pérez quien con sus conocimientos y habilidades ha sabido guiarme y aconsejarme a lo largo de toda esta etapa.

A mi madre Marcia Carrillo que con su amor, apoyo y comprensión incondicional esta a mi lado siempre velando por mi bienestar.

A mi hermana Johanna Pérez quien con sus palabras de aliento me ayuda a superar todos los momentos difíciles en mi vida.

A mi tía Lupe quien es como una segunda madre y un gran apoyo en mi vida.

A mis tíos y primos que de alguna u otra manera han contribuido para lograr esta meta.

A mi enamorada Carolina López quien con su amor es un gran respaldo en todo momento

A mí adorada mascota Neysa que siempre me brinda su compañía y todo su cariño.

Finalmente doy gracias a René Meza y Washington Carrillo, que nos brindaron su ayuda en varios aspectos para el desarrollo del proyecto.

Ricardo Pérez C.

AGRADECIMIENTO

Agradezco a mis abuelitos mamá Tere y papá Lucho por haberme guiado y apoyado en todo momento, por ser una fortaleza en los momentos duros, por brindarme todo su amor y cariño. Sobre todo por ser un ejemplo de vida.

A mi madre Sandra que es el ser más maravilloso del mundo. Gracias por el amor que me has brindado, por estar siempre guiando mi camino y por el esfuerzo realizado para verme convertido en un hombre de bien.

A mi padre Héctor quien siempre fue mi ejemplo a seguir por su sabiduría. Gracias por tu amor, apoyo, comprensión, confianza y por el sacrificio de gran parte de tu vida para educarme.

A mi hermana Jei, por su cariño, compañía y porque es un pilar fundamental en mi vida.

A mi tío Tolito por ser como mi segundo padre, quien siempre estuvo al tanto de todo, dándome fortaleza para continuar y lograr esta meta tan anhelada en mi vida.

A mis tíos, primos y amigos que estuvieron siempre a mi lado. Gracias por su confianza y lealtad.

A la familia Pérez-Carrillo por abrirme las puertas de su hogar y brindarme su amistad para que este objetivo se haya cumplido.

Para finalizar agradezco a todas la instituciones educativas especialmente al Instituto Nacional Mejía, por el conocimiento impartido para mi formación profesional.

Edwin Clavijo V.

ÍNDICE

CAPÍTULO 1

CAPÍTULO 2

CAPÍTULO 3

CAPÍTULO 4

CAPÍTULO 5

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

Los avances tecnológicos permitieron considerar la aplicación del fenómeno de la persistencia de la visión en novedosos sistemas publicitarios, que permiten proyectar texto o imágenes por medio del encendido y apagado secuencial de una sola columna de led's RGB, por tanto, se diseña e implementa un sistema de rotulación RGB, que se basa en el "Fenómeno de la persistencia de la visión" de Joseph Plateau, que demuestra como una imagen permanece en la retina humana una décima de segundo antes de desaparecer por completo. Este sistema de rotulación se realiza con el objetivo de crear un sistema publicitario llamativo e innovador. El desarrollo del sistema de rotulación RGB se realiza en cuatro etapas: investigación de los antecedentes teóricos, diseño del hardware de control de los sistemas de alimentación, visualización y soporte de los distintos dispositivos electrónicos, diseño del software y la implementación del sistema de rotulación RGB.

PALABRAS CLAVES

- Sistema de rotulación RGB.
- Persistencia de la visión.
- Sistema de publicidad LED.
- Visión del ojo humano.

CAPÍTULO 1

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE ROTULACIÓN RGB BASADO EN LA TÉCNICA DE PERSISTENCIA DE LA VISIÓN

1.1 ANTECEDENTES

En el año 130 D.C. el sabio griego Ptolomeo comenzó a explorar en el campo de la óptica, investigó las propiedades de la luz, sobre todo de la refracción, reflexión y su incidencia en la visión humana. La investigación realizada por Ptolomeo ayudó a Joseph Plateau en el año de 1830 a definir el fenómeno de la persistencia de la visión, que demuestra como una imagen permanece en la retina humana una décima de segundo antes de desaparecer por completo. El tiempo que permanece la imagen en la retina humana puede variar según la intensidad de luz, con iluminaciones fuertes se llegó a ponderar el tiempo en 1/48 seg. mientras que con iluminaciones débiles alcanza 1/20 seg. Plateau también descubrió que el ojo humano ve con una cadencia de 10 imágenes por segundo, esto permite ver la realidad como una secuencia de imágenes ininterrumpidas y puede llegar a producir sensación de movimiento.

En 1832, Plateau inventó un dispositivo estroboscópico, el primer dispositivo capaz de proporcionar la ilusión de una imagen en movimiento a partir de una secuencia de imágenes fijas. Este dispositivo estroboscópico está compuesto de dos discos coaxiales: un disco que contiene una secuencia de imágenes fijas impresas, y el otro disco que

contiene pequeñas aberturas radiales y equidistantes. Cuando los dos discos rotan a velocidad adecuada, la sincronía entre las aberturas y las imágenes crea una ilusión de animación.

El dispositivo estroboscópico de Plateau fue la base para que en el siglo XIX surjan diversos juguetes ópticos que crearon la ilusión de movimiento basados en el fenómeno de la persistencia de la visión.

Los avances tecnológicos permitieron considerar la aplicación del fenómeno de la persistencia de la visión en novedosos sistemas publicitarios, que permiten proyectar texto o imágenes por medio del encendido y apagado secuencial de una sola columna de led's RGB, con lo que las imágenes se superponen en la retina y el cerebro las enlaza como una sola imagen visual, móvil y continua.

1.2JUSTIFICACIÓN E IMPORTANCIA

En la actualidad, la mayor parte de los sistemas electrónicos para publicidad e información adolecen de un problema fundamental; su baja eficiencia. El más evidente resultado de esta baja eficiencia es el gasto innecesario y excesivo de recursos eléctricos, inciden no sólo de forma económica sino también ambiental, además las tecnologías actuales no permiten que un solo sistema publicitario logre cubrir el cien por ciento del área visual. Para abarcar la mayor parte del área de visualización de las personas es inevitable utilizar gran cantidad de pantallas, letreros o algún tipo de tecnología que despliegue mensajes luminosos. Con los adelantos tecnológicos, resulta imposible cerrar los ojos ante el futuro inmediato al que se enfrentan las empresas publicitarias, que en

cierta manera tienen la necesidad de crear sistemas llamativos al público y además cubrir la mayor parte del área visual de las personas para lograr captar su atención.

Es urgente que las empresas publicitarias posean un sistema de rotulación RGB, que permita proyectar texto o imágenes por medio del encendido y apagado secuencial de una sola columna de led's RGB, que es una forma innovadora y llamativa para la presentación de mensajes publicitarios e informativos, además ofrece precios bajos y cubre el cien por ciento del área visual de las personas porque el mensaje podrá ser visualizado desde cualquier punto alrededor del sistema publicitario.

1.3 ALCANCE

Para solventar las necesidades que tienen las empresas publicitarias de tener sistemas publicitarios que cubran el cien por ciento del área visual y sean llamativos a las personas, el presente proyecto de tesis establece la creación de un sistema publicitario rotativo, basado en el fenómeno de la persistencia de la visión, para lo cual se contempla el diseño e implementación de las siguientes etapas:

1.3.1 Mecánica

La etapa mecánica contempla el diseño y la creación del soporte mecánico para el motor y los dispositivos electrónicos del sistema de rotulación RGB.

1.3.1.1 Base metálica

Permite dar estabilidad al sistema de rotulación RGB cuando empieza su movimiento giratorio y brinda soporte al motor.

1.3.1.2 Eje rotativo

La integración del eje rotativo con el motor y la base metálica es un pilar fundamental en la implementación del sistema de rotulación RGB, puesto que es el punto de acople de todos los dispositivos electrónicos que permiten la visualización de los mensajes.

1.3.2 Partes Electrónicas

Esta etapa abarca la conexión y alimentación de los dispositivos electrónicos del sistema de rotulación RGB: microcontrolador, led's RGB, dispositivos inalámbricos, entre otros.

1.3.2.1 Diseño del circuito electrónico

Se toma en cuenta la ubicación y conexiones necesarias para los dispositivos electrónicos.

1.3.2.2 Control de velocidad del motor

El control de velocidad del motor permite establecer el tamaño y ubicación de los mensajes que aparecen en el sistema de rotulación RGB.

1.3.2.3 Alimentación de los dispositivos electrónicos

El sistema de alimentación de los dispositivos electrónicos es de vital importancia, porque debe ser constante la entrega de energía al sistema de rotulación RGB para evitar la pérdida de datos y no debe interponerse con el movimiento giratorio del eje.

1.3.2.4 Conexión inalámbrica

Envío de mensajes desde la computadora hacia el microcontrolador, a través de módulos inalámbricos.

1.3.2.5 Almacenamiento de Datos

El almacenamiento de datos permite la edición y transferencia de los mensajes al microcontrolador, para ser reutilizados en futuras aplicaciones, de esta forma se evita la necesidad de volver a diseñar un mensaje ya creado.

1.3.2.6 Interfaz de Control

La interfaz de control permite que el usuario trabaje con el sistema de rotulación RGB en el diseño de mensajes, por medio de una matriz interactiva que asemeja el área de visualización del eje rotativo.

Al culminar el diseño de cada etapa se totalizan para formar el sistema de rotulación RGB.

1.4OBJETIVOS

1.4.1 Objetivo General

Diseñar e implementar un sistema de rotulación llamativo para el mercado de publicidad, que cubra la mayor parte del área de visualización de las personas y logre captar su atención, con interfaz inalámbrica y control a través de una computadora.

1.4.2 Objetivos Específicos

1. Diseñar e implementar un sistema de rotulación RGB que se basa en la técnica de persistencia de la visión, con interfaz inalámbrica y control a través de una computadora.
2. Analizar las condiciones óptimas necesarias del sistema de rotulación para la implementación del sistema mecánico y de los dispositivos electrónicos.
3. Diseñar el hardware de control de los sistemas de alimentación, visualización y soporte de los distintos dispositivos electrónicos.
4. Diseñar e implementar la interfaz inalámbrica para los mensajes que posteriormente serán visualizados en el sistema de rotulación RGB.
5. Establecer el funcionamiento óptimo del sistema de rotulación RGB, mediante las siguientes pruebas: visualización del mensaje con el movimiento giratorio del eje rotativo, verificación de la transferencia de datos completos desde la computadora hacia el microcontrolador, comprobación de entrega de energía constante al sistema de rotulación RGB, demostración de estabilidad de la estructura mecánica del sistema de rotulación RGB.

CAPÍTULO 2

FUNDAMENTOS TEÓRICOS

2.1 INTRODUCCIÓN DE LA PERSISTENCIA DE LA VISIÓN

2.1.1 Definición

El fenómeno de la persistencia de la visión, consiste en una deficiencia del ojo humano, permite que una imagen permanezca en la retina durante un período de tiempo corto. Por ejemplo, si se coloca un objeto frente a los ojos y después de cierto tiempo de repente se retira, el ojo queda con la sensación de seguir en observación de la imagen del objeto, por lo que la visión de la imagen persiste durante un instante.

2.1.2 Efecto de la persistencia de la visión en el ser humano

La visión humana está conformada por una serie de órganos que al interactuar conjuntamente permiten que el cerebro pueda interpretar lo que el ojo ve.

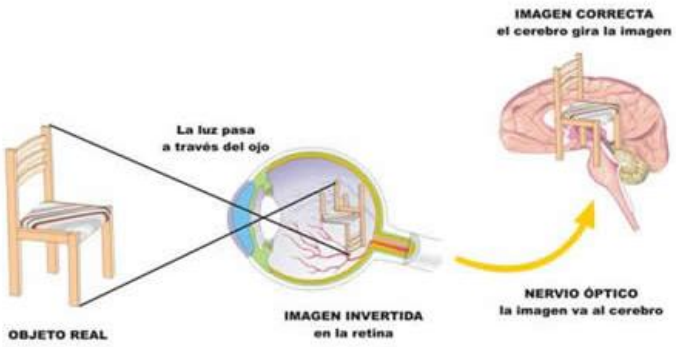


Figura. 1 Visión humana[1]

El funcionamiento de los ojos comienza cuando el haz de luz que golpea un objeto se refleja en el ojo, está conformado por rayos luminosos que pasan primero por la córnea y después por la pupila hasta llegar a la retina.

"La retina, es la pared posterior interna del ojo, contiene aproximadamente 127 millones de células sensibles a la luz que absorben los rayos luminosos y los convierten en una señal electroquímica que se transmite mediante el nervio óptico hasta llegar a la zona del cerebro donde se produce el proceso visivo".[2]

Es por este proceso visivo que el fenómeno de la persistencia de la visión se produce en los seres humanos. La excitación de la retina produce cierta cantidad de impulsos electroquímicos que nuestro cerebro interpreta como una imagen. Cuando el haz de luz que excita a la retina desaparece, las células continúan enviando impulsos electroquímicos a nuestro cerebro por un instante, lo que provoca la sensación de seguir en observación de la imagen por un intervalo de tiempo corto a pesar de que desaparezca.

1 http://www.ite.educacion.es/formacion/materiales/129/cd/unidad_1/mo1_mecanismo_de_la_vision.htm, Figura. 1 Visión humana

2 Información obtenida de la pagina web, http://www.apanovi.org.ar/iusaludparyfun.html.

2.1.3 Evolución y aplicación

2.1.3.1 Evolución

Desde la antigüedad varios científicos crearon diferentes aparatos ópticos que demuestran el fenómeno de la persistencia de la visión. Los aparatos ópticos son los instrumentos que precedieron a los elementos modernos de proyección cinematográfica que basan su funcionamiento en el fenómeno de la persistencia de la visión, algunos de estos son:

1. Taumátropo

El Dr. John Ayrton Paris inventó el taumátropo en Inglaterra en el año de 1824. Este aparato consta de un círculo de cartón con una imagen diferente en cada una de sus caras y mediante dos hilos atados a cada extremo del círculo se realiza un movimiento de rotación para observar como las imágenes se fusionan y provocan la ilusión óptica de ser una sola imagen. (Ver figura. 2)

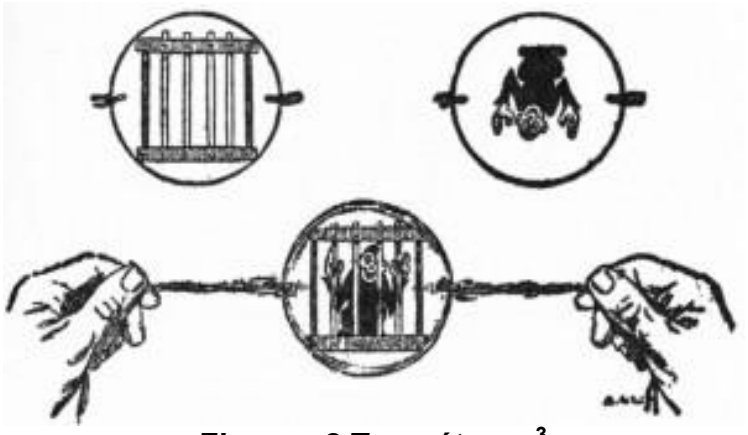


Figura. 2 Taumátropo[3]

3 http://tallerelmate.wordpress.com/2010/12/01/modelo-de-un-taller-origenes-del-cine-i/thaum/, Figura. 2 Taumátropo

2. Fenaquistiscopio y Estroboscopio

El fenaquistiscopio y estroboscopio se crearon para lograr la ilusión de movimiento continuo.

3. Fenaquistiscopio

El científico belga Joseph Plateau creó el fenaquistiscopio en el año de 1829. Este aparato óptico consta de un círculo liso con pequeñas aberturas en su borde entre las que existen imágenes de un mismo objeto pero en posiciones diferentes.

Cuando se gira el círculo con imágenes frente a un espejo, se produce la ilusión de que las aberturas se vuelven una sola, esta permite observar la imagen en movimiento. (Ver figura. 3)

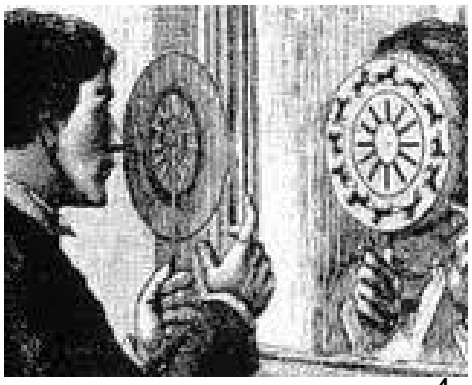


Figura. 3 Fenaquistiscopio[4]

4 http://www.fotolog.com/cineyfoto/55798200/, Figura. .3 Fenaquistiscopio

4. Estroboscopio

El austriaco Simón Von Stampfer inventó el estroboscopio en el año de 1830. Consta de un cilindro cerrado con pequeñas ranuras en su parte superior, mientras que en su parte inferior tiene imágenes de un mismo objeto pero en diferentes posiciones.

Cuando se gira el cilindro por medio de las ranuras se observa las imágenes en movimiento. (Ver figura. 4)

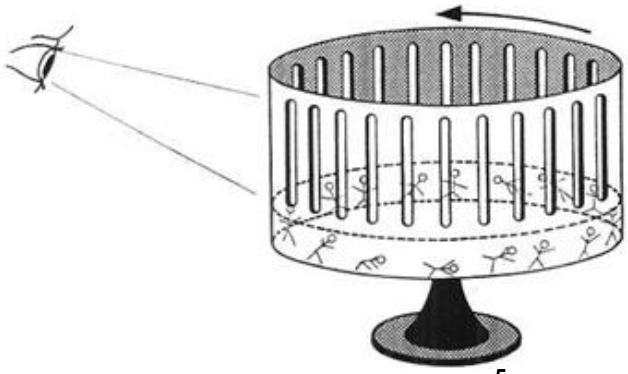


Figura. 4 Estroboscopio[5]

5. Zoótropo

El inglés William George Horner creó el zoótropo en el año de 1834 con el interés de mejorar el fenaquistiscopio de Plateau. Consiste en un cilindro con cortes que permiten al observador mantener una visión de las imágenes al interior del cilindro. (Ver figura. 5)

5 http://www.librosmaravillosos.com/comofunciona/capitulo08.html, Figura. 4 Estroboscopio

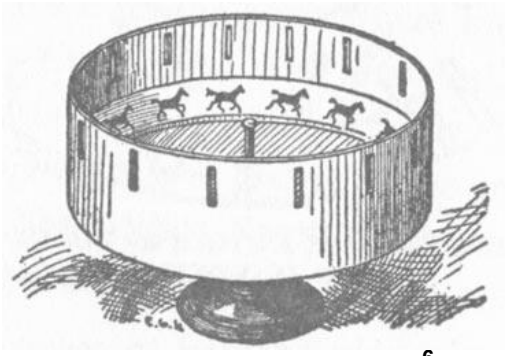


Figura. 5 Zoótropo[6]

6. Praxinoscopio

El francés Emile Reynaud realizó una modificación al zoótropo de Horner en 1877. Incluye una serie de espejos en el interior del cilindro, estos reflejan las imágenes giratorias con movimientos pausados y menos bruscos que los del zoótropo. (Ver figura. 6)

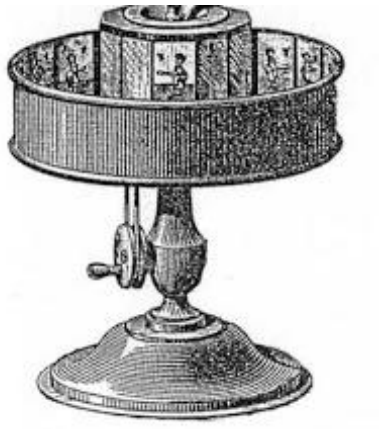


Figura. 6 Praxinoscopio[7]

[6] http://www.alternatilla.com/2009/exposiciones/13-zootropos/, Figura. 5 Zoótropo
[7] http://13gatosnegros.blogspot.com/2008/03/praxinoscopio.html, Figura. 6 Praxinoscopio

7. Mutoscopio

El norteamericano Herman Castler creó el mutoscopio en 1895. Consta de una caja que alberga en su interior un gran número de fotografías en blanco-negro. En el funcionamiento deja caer una imagen tras otra para formar la ilusión de movimiento. (Ver figura. 7)

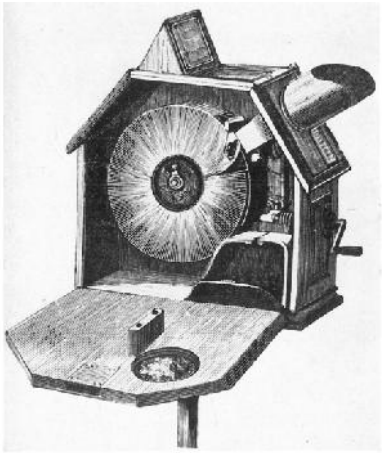


Figura. 7 Mutoscopio[8]

Los aparatos ópticos descritos son la inspiración para crear nuevos dispositivos llamativos y novedosos que ayudan en el trabajo cinematográfico y en las industrias publicitarias.

2.1.3.2 Aplicación

La principal aplicación del fenómeno de la persistencia de la visión se evidencia en el campo cinematográfico, puesto que el proyector de cine basa su funcionamiento en dicho fenómeno. El proyector de cine es un aparato con engranajes y poleas, que hacen

8 http://laedaddelpixel.blogspot.com/2013/02/la-capsula-del-tiempo-aquellos.html, Figura. 7 Mutoscopio

avanzar una cinta compuesta por fotogramas[9] frente a una lámpara. Cada fotograma se detiene frente a la lámpara durante 1/24 de segundo e inmediatamente el obturador[10] impide el paso de luz de la lámpara mientras el siguiente fotograma se coloca en su sitio. En ese lapso vemos las imágenes grabadas en la retina que da la sensación de ver las imágenes de corrido. (Ver figura. 8)

Figura. 8 Cinematografía[11]

2.2 LED DISPLAYS

Las bases de la actual tecnología LED[12] se descubrieron por azar del destino. Henry Joseph Round, en el siglo XX, realizó experimentos para mejorar el rendimiento de los receptores de radio y observó extrañas luminosidades provenientes de diferentes materiales semiconductores cuando corrientes eléctricas los atravesaban.

"Oleg Lósev, veinte años más tarde se basó en los escritos que dejó H. Round y perfeccionó un dispositivo que fuera capaz de generar energía lumínica y que permitiera

9 Fotograma: imagen obtenida sin la cámara fotográfica, por medio de un proceso que consiste en la superposición del objeto a registrar

10 Obturador: dispositivo que controla el tiempo durante el que llega la luz al dispositivo fotosensible

11 http://mistusynos.blogspot.com/2007/04/peliculas-movie-cintas-cinematograficas.html, Figura. 8 Cinematografía

12 LED: diodo emisor de luz

observar todas las ventajas que esta novedosa fuente de luz poseía por sobre las luminarias tradicionales. O. Lósev describe con bases y análisis académico la explicación del fenómeno electroluminiscente"[13]. En 1962 la industria de los semiconductores desempolvó los escritos de H. Round y O. Lósev y dio forma a los diodos LED's que hoy se conocen.

El primer LED resultó de la combinación de los materiales químicos semiconductores: galio, arsénico y fósforo. Se logró conseguir un LED ROJO con una emisión de luz relativamente baja.

En las siguientes décadas 70's, 80's y 90's la tecnología LED tuvo grades avances y permitió emplearla en: la iluminación industrial, iluminación arquitectónica, las aplicaciones automotrices, luminarias para el alumbrado público, los artículos de control de tráfico, entre otros.

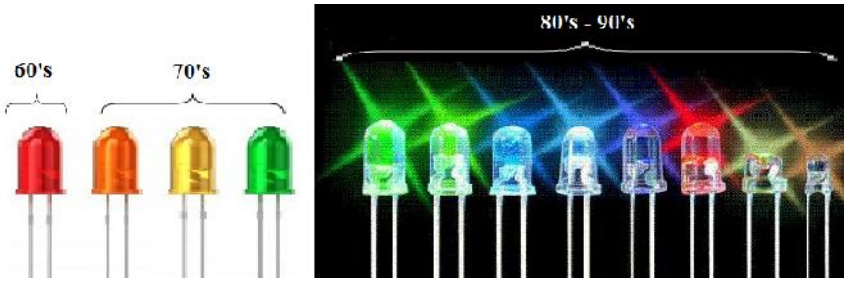


Figura. 9 Evolución LED décadas

13Información obtenida de la pagina web, http://www.neoteo.com/electronica-basica-diodos-emisores-de-luz-led

2.2.1 Matrices de LED

La matriz de LED, es un cartel formado por varias filas y columnas de LED's, esta se conoce en algunos países como "Cartel de LED's" o "Publik", recursos que se utilizan frecuentemente con fines publicitarios e informativos.

Figura. 10 Matriz de LED[14]

El mercado de LED's se ha ampliado con el avance de la tecnología, por lo que existe gran variedad de LED's, tamaños, colores y desempeños.

2.2.1.1 Clasificación

1. Pantalla modular

Es un dispositivo que contiene paneles compuestos por LED's RGB, que forman píxeles y muestran caracteres, textos, imágenes y video. La pantalla modular en la actualidad es popular, su variedad de tamaño facilita la proyección de videos, imágenes, textos, logotipos, entre otros. Es una excelente alternativa tecnológica para las empresas que deseen vender publicidad.

14 http://www.seeedstudio.com/wish/smaller-rgb-led-matrix-p2076, Figura. 10 Matriz de LED

2. Pantalla de mensajes variables

Es un dispositivo diseñado para publicidad exterior e interior, muestra mensajes compuestos por números, símbolos y letras que pueden ir variando de acuerdo a las condiciones establecidas en la pantalla.

3. Pantalla de mensajes fijos

Es una pantalla de texto fijo que no genera ningún tipo de movimiento y presenta un solo mensaje. Si se requiere cambiar el mensaje en la pantalla de mensajes fijos se debe programar de nuevo la pantalla para mostrar un nuevo mensaje.

2.2.2 Aplicaciones

En la actualidad los LED's se encuentran en una variedad de equipos electrónicos como: radios, televisores, celulares, pantallas de relojes digitales, entre otros. Los LED's evolucionaron hasta alcanzar variedad de colores y eficiencia lumínica, estas características permiten que incursionen en otros dispositivos como los letreros publicitarios.

2.2.2.1 Letrero de información

Es un dispositivo electrónico fijo que contiene varias columnas de LED's. Distribuyen información en la vía pública y ofrecen oportunidad e inmediatez. Los letreros de información sirven para manejar diversos tipos de mensajes:

1. Mensajes comerciales

En el ámbito de la comunicación comercial las ventajas de la emisión de estos mensajes son:

1. Presentación de hora, temperatura, promociones de productos o servicios especiales, etc.
2. Visualización de alto alcance.
3. Comunicación diferente donde la marca solicitante es protagonista.
4. Presentación del mensaje las 24 horas ininterrumpidas.

Figura. 11 Letreros de Información Comercial[15]

2. Mensajes deportivos

La emisión de estos mensajes en el letrero de información sirve para patrocinar eventos deportivos, sucesos del juego así como mostrar el marcador del juego deportivo, cronómetro y animaciones. (Ver figura. 12)

15 http://www.screens.ru/es/2003/4.html, Figura. 11 Letreros de Información Comercial

Figura. 12 Letreros de Información Deportiva[16]

3. Mensajes de información universitaria

En los letreros de información también se maneja la comunicación interna en establecimientos como los centros educativos, son de vital importancia para brindar la actualización diaria de la información. (Ver figura. 13)

Las ventajas de la emisión de estos mensajes pueden ser:

1. Presentación de la cartelera de actividades culturales, deportivas, académicas, etc.
2. Señalización.
3. Emisión de noticias internas y externas.
4. Programación de fechas de matrículas, exámenes, horarios, etc.

Figura. 13 Letreros de Información Universitaria[17]

16 http://cajamarca-cajamarca.nexolocal.com.pe/c690-compra-venta-p2, Figura. 12 Letreros de Información Deportiva

17 http://cuernavaca.olx.com.mx/anuncio-programable-led-32x96-display-programable-letrero-programable-led-iid-473291256, Figura. 13 Letreros de Información Universitaria

4. Mensajes de información vial

La emisión de mensajes viales en los letreros de información es de importancia para la sociedad puesto que ayudan a la identificación de situaciones de riesgo de accidente para el peatón y el conductor, muestran la velocidad vehicular y la que es permitida así como la calidad en la que se encuentra la vía, entre otras cosas. (Ver figura. 14)

Figura. 14 Letreros de Información Vial[18]

5. Mensajes de información industrial

Dentro de los sistemas electrónicos de información industrial se emiten mensajes en tiempo real que permiten una mejor comunicación con clientes y empleados, optimizan el tiempo de respuesta y reducen costos de papelería y logística. (Ver figura. 15)

Figura. 15 Letreros de Información Industrial[19]

18 http://lumtec.com.mx/Pantallas_electronicas_LED/senalizacion_vial.html, Figura. 14 Letreros de Información Vial

2.3 DISPLAY ROTATIVO

En la actualidad y con el avance de la tecnología, existen varias aplicaciones para la presentación de información, un ejemplo es el display rotativo que es un dispositivo electrónico giratorio que contiene una columna de LED's RGB. El funcionamiento del display rotativo consiste en la rotación de una columna de LED's, que al encender y apagar simultáneamente, previa programación, se puede visualizar letras, números o imágenes. (Ver figura. 16)

Figura. 16 Columna de LED's

Para la visualización de los mensajes es necesario colocar un motor o algún tipo de adaptación que haga girar a la columna de LED's. (Ver figura. 17, figura. 18 y figura.19)

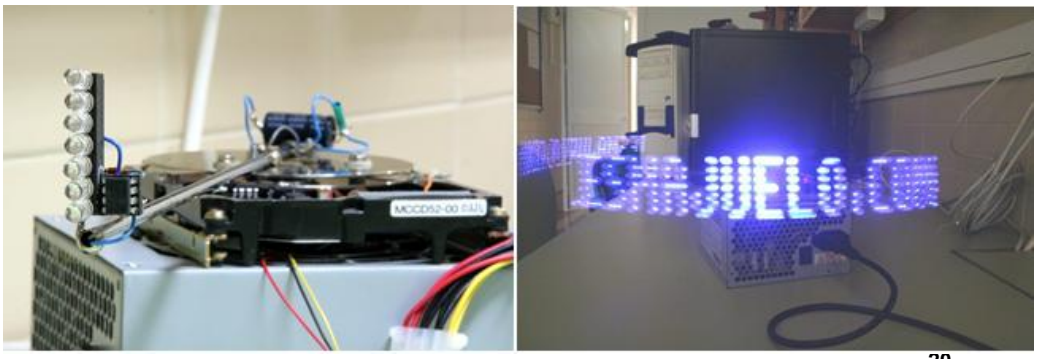


Figura. 17 Adaptación motor a la columna de LED's[20]

19 http://www.rotuloselectronicos.net/carteles-luminosos-y-pantallas-informativas-multilinea.html, Figura. 2.15 Letreros de información industrial

20 http://www.iesmajuelo.com/index.php?f=departamentos&id=354&deptno=Inform%E1tica, Figura. 2.16 Columna de LED's, Figura. 17 Adaptación motor a la columna de LED's

Figura. 18 Mensajes en rueda de bicicleta[21]

Figura. 19 Mensajes girando con la mano[22]

El desarrollo del display rotativo en el mercado actual no muestra avances. Se utiliza mínimamente para mostrar la hora, fecha o mensajes pregrabados. Un ejemplo es el dispositivo Olympia OL 3000 Infoglobe Digital Caller ID[23] (Ver figura. 20)

Figura. 20 Olympia OL 3000 Infoglobe Digital Caller ID[24]

21 http://agitpov.wordpress.com/tag/persistencia-de-vision/, Figura. 18 Mensajes en rueda de bicicleta

22 http://www.taringa.net/posts/hazlo-tu-mismo/10701574/POV-Persistence-Of-Vision-Con-Leds.html, Figura. 19 Mensajes girando con la mano

23 Olympia OL 3000 Infoglobe Digital Caller ID: aparato creado basado en el fenómeno de la persistencia de la visión.

2.4MICROCONTROLADOR

El microcontrolador es un computador digital integrado en un chip que cuenta con una unidad de procesamiento central, memoria para el almacenamiento del programa y para el almacenamiento de datos y puertos de entrada salida.

El microcontrolador se desarrolló para emplearse en aplicaciones pequeñas que no involucran el uso de gran cantidad de memoria, se utiliza en una variedad de sistemas embebidos[25], que controlan máquinas y componentes de sistemas complejos como aplicaciones industriales de automatización y robótica, equipos médicos y dispositivos de la vida diaria, por ejemplo: automóviles, hornos microondas, teléfonos, entre otros. El bajo costo económico, el mínimo consumo de energía y su alta flexibilidad son las principales ventajas que tiene el microcontrolador.

Hoy en día existen varios fabricantes de microcontroladores, que ofertan un sin número de modelos diferentes (Ver tabla. 1). La elección del microcontrolador a utilizar debe ser de acuerdo a la capacidad de las memorias, velocidad de funcionamiento, el número de líneas de entrada y salida y tipo de programación a emplear, entre otros.

AtmelDallas	Microchip (PIC)
Semiconductor	Hitachi
Intel	Motorola
Philips	National Semiconductor
Siemens	SGS-Thomson
Temic	Texas Instrument

Tabla. 1 Fabricantes de microcontroladores

24 http://todousa.co/olympia-ol-3000-infoglobe-digital-caller-id-real-time-clock, Figura. 20 Olympia OL 3000 Infoglobe Digital Caller ID

[25] Sistema embebido: diseñado para realizar una o algunas pocas funciones dedicadas

2.4.1 Componentes principales del microcontrolador

Un microcontrolador como circuito integrado de alta escala de integración dispone normalmente de los siguientes componentes:

2.4.1.1 Unidad de procesamiento Central (CPU)

La unidad de procesamiento central o procesador, es el componente principal del microcontrolador, que interpreta las instrucciones contenidas en los programas y procesa los datos. Existen versiones de 4, 8, 32 y hasta 64 bits con arquitectura Harvard o arquitectura de Von Neumann.

2.4.1.2 Memoria de programa

La memoria de programa se dedica a almacenar instrucciones del programa. Por ejemplo la memoria EEPROM (Electrically Erasable/Programable ROM) es una memoria de sólo lectura, programable y borrable eléctricamente desde el propio grabador o la memoria flash que es una memoria no volátil, pequeña y veloz, permite escribir y borrar. Funciona como una memoria ROM y una RAM, pero consume menos energía, tolera más ciclos de escritura y mismo borrado que una EEPROM.

2.4.1.3 Memoria de datos RAM (Random Access Memory)

Es una memoria de tipo volátil y su función es la de guardar las variables y datos, es de poca capacidad pues sólo debe contener las variables y los cambios de información que se produzcan en el transcurso del programa, la capacidad de la memoria RAM puede ser de 1, 2, 4, 8, 16 y 32 kilobytes.

2.4.1.4 Generador del reloj

Es un circuito oscilador, que genera una onda cuadrada de alta frecuencia y configura los impulsos de reloj de uso en la sincronización de las operaciones del sistema.

2.4.1.5 Interfaz de entrada/salida

Las líneas de entrada/salida comunican al computador interno con los periféricos exteriores y se destinan a proporcionar el soporte a las señales de entrada, salida y control.

A continuación se describen las características principales de los microcontroladores PIC de la Familia 16F87X y de la Familia 18FXX2. (Ver tabla. 2, tabla. 3)

Características	16F876	16F877	18F442	18F452
Frecuencia Máxima	DX-20Mhz	DX-20Mhz	DX-40Mhz	DX40Mh
Memoria de programa FLASH	8KB	8KB	16KB	32KB
Posiciones RAM de datos	368	368	768	1536
Posiciones EEPROM	256	256	256	256
Puertos E/S	A, B y C	A, B, C, D y E	A, B,C,DyE	A, B,C,DyE
Nº de Pines	28	40	40	40
Interrupciones	13	14	18	18
Timers	3	3	4	4
Comunicaciones Serie	MSSP,USART	MSSP,USART	MSSP,USART	MSSP,USART
Comunicación Paralelo	-	PSP	-	PSP
Convertidor A/D de 10 bits	5	8	8	8
Juego de Instrucciones	35	35	75	75

Tabla. 2 Microcontroladores PIC de la Familia 16F87x y 18Fxx2

2.4.1.6 Características generales del PIC18F452

En la Tabla. 2, el microcontrolador PIC 18F452, es el que presenta las características que se ajustan a las necesidades para el desarrollo del sistema de rotulación RGB, y se toma en cuenta el tipo de memoria de programa, capacidad, número de pines, de puertos de entrada y salida y tipo de comunicación entre otros.

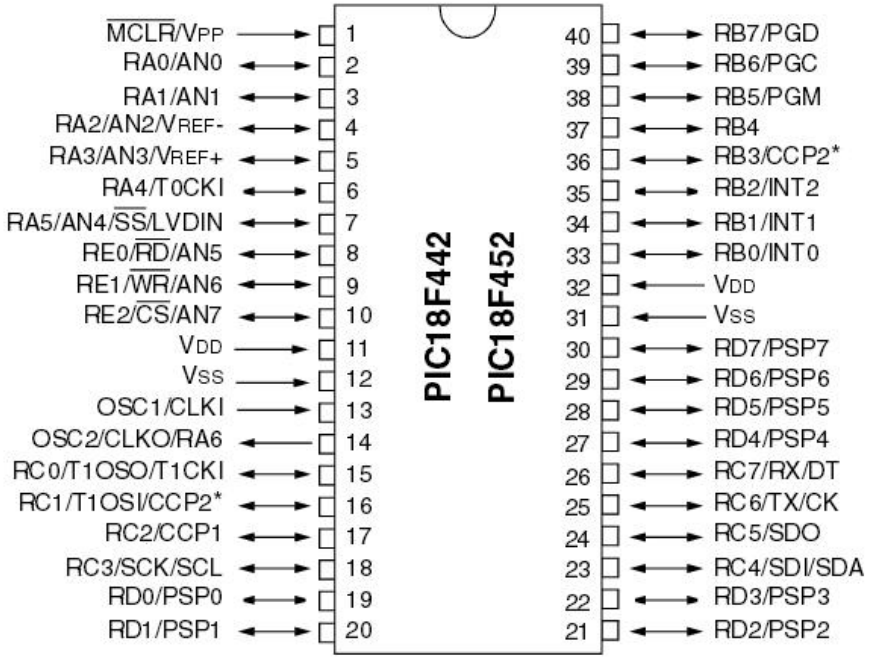


Figura. 21 PIC 18F452[26]

26 http://ww1.microchip.com/downloads/en/DeviceDoc/39576c.pdf. figura. 21 (Microcontroladores PIC) PIC18F452

Componentes Internos	Características
CPU:	- Tecnología RISC - Manejo de 75 instrucciones. - Juego de instrucciones reducido para ejecución rápida. - Frecuencia de operación de 0 a 40 MHz. - Opciones de selección del oscilador.
Memoria:	- 32Kb de memoria Flash de programa. - 1536 bytes de memoria de datos (RAM). - 256 bytes de memoria de datos EEPROM. - Lectura/escritura de la CPU a la memoria flash.
Reset e interrupciones:	- interrupciones (internas y externas). - Reset de encendido (POR). - Timer de encendido (PWRT). - Timer de arranque del oscilador (OST) - Sistema de vigilancia Watchdog timer.
Consumo de Energía	- Rango de voltaje de operación de 2.0 a 5.5 volts. - Alta disipación de corriente de la fuente: 25mA.

Tabla. 3 Características PIC18F452 componentes internos

Componentes Externos	Características	
5 Puertos Paralelos (33 pines E/S)	Puerto A (6 bits) Puerto B (8 bits) Puerto C (8 bits) Puerto D (8 bits) Puerto E (3 bits)	Con líneas digitales programables individualmente
3 Timers	Timer0 (8-16 bits) Timer1 (16 bits) Timer2 (8 bits) Timer3 (8 bits)	Contador/Temporizador de 8 bits o 16 bits con pre-escalador de 8 bits Contador/Temporizador de 16 bits con pre-escalador Contador/Temporizador de 8 bits con pre-escalador de 1, 4, 16 bits y post-escalador de 1 a 16 bits. Contador/Temporizador de 8 bits con pre-escalador
2 Módulos CCP	Captura Comparación PWM	16 bits 16 bits 10 bits
1 Convertidor A/D	AN0,…., AN7	De 10 bits, hasta 8 canales
Puertos Serie	SSP USART/SCI ICSP	Puerto Serie Sincrono Puerto Serie Universal Puerto Serie para programación y depuración “in-circuit”

Tabla. 4 Características PIC18F452 componentes externos

2.4.2 Manejo de Registros Específicos.

Address	Name	Address	Name	Address	Name	Address	Name
FFFh	TOSU	FDFh	INDF2(3)	FBFh	CCPR1H	F9Fh	IPR1
FFEh	TOSH	FDEh	POSTINC2(3)	FBEh	CCPR1L	F9Eh	PIR1
FFDh	TOSL	FDDh	POSTDEC2(3)	FBDh	CCP1CON	F9Dh	PIE1
FFCh	STKPTR	FDCh	PREINC2(3)	FBCh	CCPR2H	F9Ch	—
FFBh	PCLATU	FDBh	PLUSW2(3)	FBBh	CCPR2L	F9Bh	—
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	—
FF9h	PCL	FD9h	FSR2L	FB9h	—	F99h	—
FF8h	TBLPTRU	FD8h	STATUS	FB8h	—	F98h	—
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	—	F97h	—
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	—	F96h	TRISE(2)
FF5h	TABLAT	FD5h	T0CON	FB5h	—	F95h	TRISD(2)
FF4h	PRODH	FD4h	—	FB4h	—	F94h	TRISC
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB
FF2h	INTCON	FD2h	LVDCON	FB2h	TMR3L	F92h	TRISA
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	—
FF0h	INTCON3	FD0h	RCON	FB0h	—	F90h	—
FEFh	INDF0(3)	FCFh	TMR1H	FAFh	SPBRG	F8Fh	—
FEEh	POSTINC0(3)	FCEh	TMR1L	FAEh	RCREG	F8Eh	—
FEDh	POSTDEC0(3)	FCDh	T1CON	FADh	TXREG	F8Dh	LATE(2)
FECh	PREINC0(3)	FCCh	TMR2	FACh	TXSTA	F8Ch	LATD(2)
FEBh	PLUSW0(3)	FCBh	PR2	FABh	RCSTA	F8Bh	LATC
FEAh	FSR0H	FCAh	T2CON	FAAh	—	F8Ah	LATB
FE9h	FSR0L	FC9h	SSPBUF	FA9h	EEADR	F89h	LATA
FE8h	WREG	FC8h	SSPADD	FA8h	EEDATA	F88h	—
FE7h	INDF1(3)	FC7h	SSPSTAT	FA7h	EECON2	F87h	—
FE6h	POSTINC1(3)	FC6h	SSPCON1	FA6h	EECON1	F86h	—
FE5h	POSTDEC1(3)	FC5h	SSPCON2	FA5h	—	F85h	—
FE4h	PREINC1(3)	FC4h	ADRESH	FA4h	—	F84h	PORTE(2)
FE3h	PLUSW1(3)	FC3h	ADRESL	FA3h	—	F83h	PORTD(2)
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB
FE0h	BSR	FC0h	—	FA0h	PIE2	F80h	PORTA

Figura. 22 Registros y bancos del PIC18F452[27]

2.4.2.1 Registro de Estado (STATUS)

El registro de estado posee 5 bits, que se destinan a controlar las funciones importantes del procesador.

27 http://bibing.us.es/proyectos/abreproy/11301/fichero/Memoria%252FCap%C3%ADtulo+3.pdf. Figura..22 Registros y bancos del PIC18F452

2.4.2.2 Registros de control de interrupciones

Los registros de control de interrupciones sirven para el control global de las interrupciones del programa grabado en el microcontrolador e indica la procedencia de algunas de estas por medio de los bits de estado y su prioridad.

2.4.3 Configuración y manejo de puertos I/O

El microcontrolador PIC18F452 posee 5 puertos configurables para trabajar como entradas o como salidas a selección del programador, se identifican con los nombres de portA, portB, portC, portD y portE.

2.4.3.1 Puerto A

Dispone de 6 líneas bidireccionales (RA0-RA5), que se manejan a través de 3 registros PORTA, TRISA y ADCON1.

2.4.3.2 Puerto B

Dispone de 8 líneas bidireccionales (RB0-RB7), que se manejan a través de los registros PORTB, TRISB. El pin RB3 se puede configurar como pin periférico alternativo para el módulo CCP2.

2.4.3.3 Puerto C

Dispone de 8 líneas bidireccionales (RC0-RC7), que se manejan a través de 2 registros PORTC y TRISC.

Las líneas del puerto C están multiplexadas con varias líneas, se controlan por otros periféricos. En la Tabla. 5 se resumen las líneas del puerto C y de los periféricos que están multiplexadas con ellas.

Nombre	Función Multiplexada
RC0/T1OSO/T1CKI	Salida oscilatoria del Timer1/reloj de entrada del Timer1
RC1/T1OSI/CCP2	Entrada oscilatoria del Timer1/entrada de captura2 o salida de comparacion2 o salida PWM2
RC2/CCP1	Entrada de captura1 o salida de comparacion1 o salida PWM1
RC3/SCK/SCL	Reloj para los modos de comunicación serie síncrona SPI e I2C
RC4/SDI/SDA	Dato de entrada (en modo SPI)/ Dato de entrada-salida (modo I2C)
RC5/SDO	Dato de salida (en modo SPI)
RC6/TX/CK	Línea de transmisión asíncrona se la USART/ reloj síncrono
RC7/RX/DT	Línea de recepción asíncrona de la USART/ dato síncrono

Tabla. 5 Periféricos multiplexados con el Puerto C[28]

2.4.3.4 Puerto D

Dispone de 8 líneas bidireccionales (RD0-RD7), que se manejan a través de 3 registros PORTD, TRISD.

2.4.3.5 Puerto E

Dispone de 3 líneas bidireccionales (RE0-RE2), que se manejan a través de 3 registros PORTE, TRISE y ADCON1.

28http://ww1.microchip.com/downloads/en/DeviceDoc/39576c.pdf. Tabla.5 Periféricos multiplexados con el puerto C

2.5DESCRIPCIÓN DEL SOFTWARE

2.5.1 Java

Figura. 23 Logo de Java[29]

2.5.1.1 Introducción

Los lenguajes de programación son idiomas artificiales que se forman por un conjunto de palabras reservadas, símbolos, operadores, reglas sintácticas y reglas semánticas que definen su estructura. El proceso de programación consiste en la escritura del programa, compilación y verificación del código.

Para escoger un lenguaje de programación se debe entender completamente el problema que queremos resolver y conocer las restricciones de operación, una vez que escogemos el lenguaje de programación más adecuado, procedemos a resolver el problema desde un punto de vista algorítmico[30].

29 www.wmskill.com, Figura. 23 Logo de Java

30 Un algoritmo es un conjunto ordenado y finito de operaciones que permite hallar la solución de un problema.

2.5.1.2 Historia de Java

JAVA, surgió en la empresa Sun Microsystems. En su inicio JAVA fue un lenguaje de programación que se destinó para pequeños dispositivos electrónicos, con el surgimiento de este lenguaje sencillo, capaz de generar un código de tamaño minúsculo y en combinación con C++, se creó en 1991, un nuevo lenguaje llamado Oak [31], para solucionar el problema de los cambios continuos en los chips electrónicos, es decir se desarrolló un código neutro que no dependía del tipo de dispositivo electrónico y se ejecutó sobre una máquina virtual. La máquina virtual interpretó el código neutro y lo convirtió en un código particular para el chip electrónico que se utilizo.

El lenguaje Oak tuvo poca aceptación entre las empresas creadoras de los dispositivos electrónicos. Para el año de 1995, el lenguaje sufre un cambio de su nombre y tiene varias mejoras en su diseño para surgir como lenguaje de programación para computadoras, JAVA.

2.5.1.3 La plataforma de Java

Los programas Java, se compilan en un lenguaje intermedio, que se denomina Bytecode. Este código se interpreta por la máquina virtual de Java del entorno de ejecución (JRE) [32] y se consigue la portabilidad en distintas plataformas.

---

31 Oak es un lenguaje de programación.

32 El JRE es una pieza intermedia entre el código Bytecode y los distintos sistemas operativos existentes en el mercado.

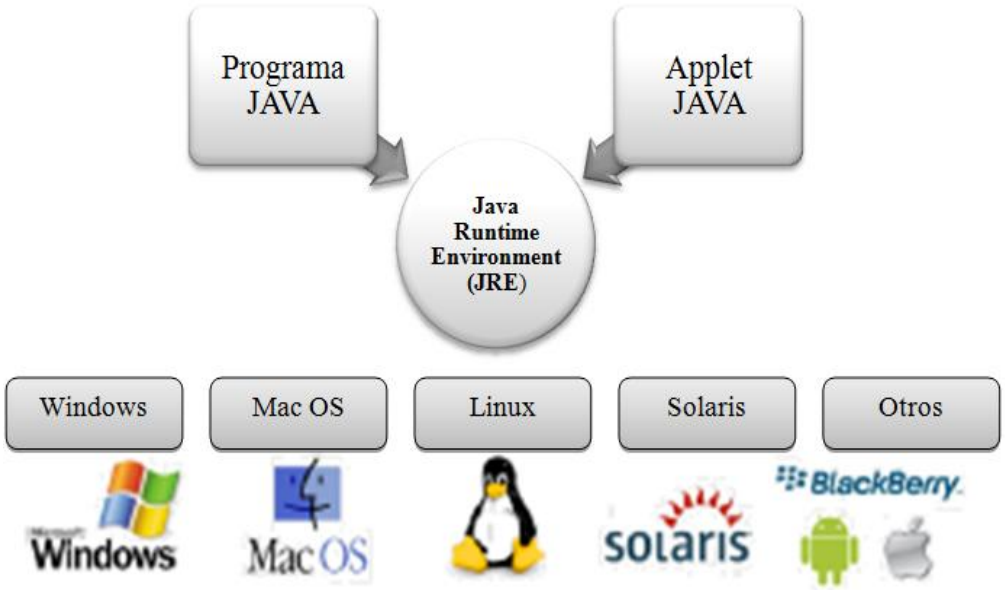


Figura. 24 Java Runtime Environment (JRE)

La evolución del lenguaje de programación Java avanzó rápidamente por lo que no solo es un lenguaje, sino también una plataforma de desarrollo (Java Development Kit JDK), un entorno de ejecución y un conjunto de librerías para desarrollo de programas sofisticados (Java Application Programming Interface API). (Ver figura 25)

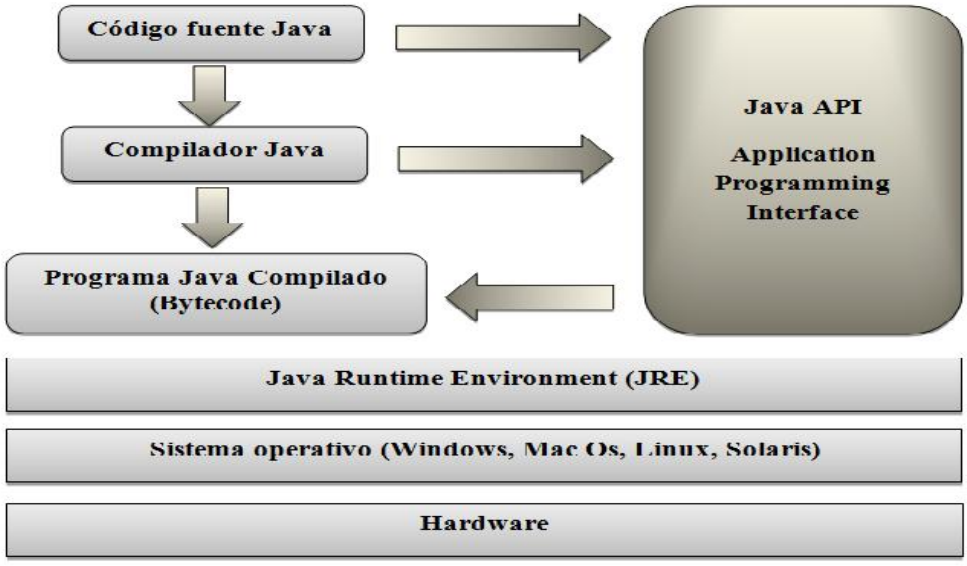


Figura. 25 Elementos de la Plataforma Java

2.5.1.3.1 Compilador de Java

Es una herramienta que se incluye en el JDK que cumple con la función de analizar los archivos fuentes de Java (extensión . java) y si no encuentra ningún error en la compilación de los archivos fuentes, genera los archivos compilados o "bytecode" [33] (extensión . class), en el caso de encontrar errores muestra la línea de código que contiene el error. (Ver figura 26)

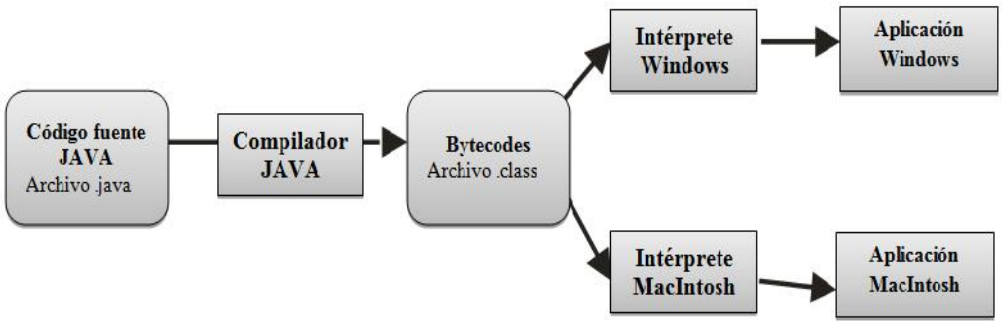


Figura. 26 Proceso de compilación de un programa Java

2.5.1.3.2 Application Programming Interface (API)

Las API son un conjunto de librerías de código JAVA, en las que se detallan las llamadas a las funciones, argumentos y los resultados que se obtienen al utilizarlas.

2.5.1.3.3 Java Virtual Machine (JVM)

La Java Virtual Machine nació por la necesidad de tener un software que no dependiera del tipo de procesador, por eso se planteó tener un código neutro, es decir

33 Byte-codes: Son el resultado de la compilación de un programa de Java y es un código de maquina virtual que es interpretado por el interprete Java.

que una vez compilado no es necesaria ninguna modificación para que este código se ejecute en otra máquina.

Una vez que se logró desarrollar los códigos neutros intervino la JVM, quien es la encargada de interpretar los códigos neutros para convertirlos en un código particular del sistema operativo que se utilizó, esto evita realizar un programa distintito para cada sistema operativo o plataforma.

2.5.1.4 Entorno de desarrollo de Java

Existen distintos programas que permiten desarrollar el código Java, Oracle es una empresa que distribuye el JDK (Java Development Kit), que es un conjunto de programas y librerías que permiten desarrollar, compilar y ejecutar aplicaciones en Java.

Otra opción para desarrollar código Java son los IDE's (Integrated Development Environment) que son los entornos de desarrollo integrado que permiten escribir el código, compilarlo y ejecutarlo sin tener que cambiar de aplicación, además tienen un conjunto de plantillas para diseñar y efectuar toda las tareas necesarias que se desea incorporar en un programa, brindado así ciertas ventajas al programador que las utilice. Los tipos de IDE's que existen en el mercado son:

Netbeans: Es un entorno gratuito que contiene un editor avanzado de código, depurador, diversos lenguajes, extensiones de todo tipo.

Eclipse: Es uno de los más utilizados por su compatibilidad con todo tipo de aplicaciones Java y sus interesantes opciones de ayuda al escribir código.

JBuilder: Es un entorno completo que se creó por la empresa Borland para la invención de todo tipo de aplicaciones Java, que incluyen aplicaciones para móviles.

JCreator: Es un editor comercial potente y de precios bajos que funciona en varias máquina, no es un IDE completo y eso lo hace ligero.

2.5.1.5 Características de JAVA

- Es un lenguaje sencillo por su facilidad de comprensión.

- Se orienta a objetos, porque los objetos agrupan en sus estructuras sus datos como los métodos que actúan sobre los mismos, es decir, ocultan los estados de los datos miembros del objeto y solo es posible modificarlos con los métodos que se definen en el mismo objeto, además permite utilizar los datos o métodos de una clase dentro de otra como si le perteneciera a esta nueva clase[34].

- Es un lenguaje de programación que permite verificar el código al mismo tiempo que se escribe y antes de ejecutarse.

- Java es un leguaje muy seguro pues controla y limita el acceso a recursos del sistema y evita daños sobre el mismo.

- El lenguaje de programación Java permite ejecutar las aplicaciones en cualquier plataforma, sin que cambie la esencia del programa que se diseñó. Quiere decir que será el mismo programa en cualquier plataforma.

---

34 Clase: Una clase es una colección de datos y métodos.

- Java permite muchas funciones simultáneas en una aplicación y obtiene mejor rendimiento interactivo y comportamiento en tiempo real.

2.5.1.6 Diferencias de Java con respecto a otros lenguajes

- Es fácil de aprender y se estructura correctamente.
- Java es un lenguaje de programación gratuito.
- Es un lenguaje que se orienta a objetos.
- Aprovecha características de la mayoría de lenguajes modernos y evita sus inconvenientes.
- Java crea programas ejecutables. Se ejecutan independientemente del microprocesador o sistema operativo que utilice la máquina.
- Permite realizar multitareas.
- Tiene una gran funcionalidad gracias a sus librerías.
- Genera aplicaciones con pocos errores.
- Es seguro porque puede controlar el acceso a los recursos del sistema.
- El manejo de la memoria no es un problema, la gestiona el propio lenguaje y no el programador.
- Permite escribir Applets, aplicaciones para intraredes, aplicaciones cliente/servidor, aplicaciones distribuidas en redes locales y en Internet.

2.5.1.7 Tipos de programas en Java

2.5.1.7.1 Aplicaciones

Son una clase de Java que funcionan como una aplicación especifica en cualquier ambiente operativo, pueden tener referencias a archivos, interfaz gráfica, entre otros.

2.5.1.7.2 Applet

Es una clase de Java que funciona dentro de un navegador y que no puede hacer referencias a archivos, también posee su interfaz gráfica.

2.5.1.7.3 Servlets

Son aplicaciones que se ejecutan en un servidor de aplicaciones web y de las que resulta una página web.

2.5.1.8 NetBeans

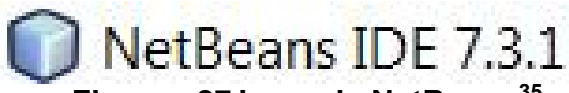


Figura. 27 Logo de NetBeans[35]

2.5.1.8.1 Introducción

Java no cuenta con un entorno de desarrollo propio, por esto se puede desarrollar programas de Java en bloc de notas o en entornos de desarrollo avanzados como Netbeans.

35 http://www.mundonets.com/herramientas/netbeans-ide/, Figura..27 Logo de NetBeans

Netbeans es un poderoso entorno de desarrollo que permite editar programas en Java, compilarlos, ejecutarlos, depurarlos y construir aplicaciones complejas con interacción web, UML, base de datos, aplicaciones para telefonía móvil e inclusive inteligencia artificial.

En la última versión Netbeans 7.3.1 se desarrolla la interfaz para el sistema de rotulación RGB. Un programador puede utilizar una versión anterior de Netbeans 7.3.1 porque todas las versiones antiguas de este son compatibles con las nuevas. Cada versión se actualiza de acuerdo a los avances de la tecnología, es por eso que en esta versión se crea y depura aplicaciones web y móviles con HTML5 y JavaScript, además incluye mejoras en IDE para C/C++, JavaFX, Groovy.

2.5.1.8.2 Historia de Netbeans

"NetBeans comenzó como un proyecto estudiantil (originalmente llamado Xelfi[36]) en la República Checa, en 1996 y estuvo bajo la tutoría de la Facultad de Matemáticas y Física en la Universidad Carolina en Praga.

Sun Microsystems en 1999 quiso tener mejores herramientas de desarrollo de Java y se interesó en NetBeans y tomó la decisión de que este entorno de desarrollo sea de código abierto[37].

La plataforma NetBeans permite que las aplicaciones se desarrollen a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de Java escritas para interactuar con las API's de NetBeans y un

36 Xelfi fue el primer entorno de desarrollo integrado escrito en Java.

37 Código abierto: Distribuido y desarrollado libremente.

archivo especial que lo identifica como módulo. Los módulos pueden desarrollarse independientemente, las aplicaciones que se basan en la plataforma NetBeans que pueden extenderse fácilmente por otros desarrolladores de software."[38]

2.5.1.9 Creación de una aplicación Java en NetBeans 7.3.1

1. Para desarrollar un programa de Java se debe abrir el entorno de desarrollo NetBeans con doble clic en el icono creado anteriormente en el escritorio de la PC. (Ver figura 28)

Figura. 28 Icono NetBeans 7.3.1

2. El programa muestra la ventana de presentación del entorno, donde se visualizan las distintas características y funciones del programa. (Ver figura 29)

38 Información obtenida de, https://netbeans.org/about/history_pt_BR.html

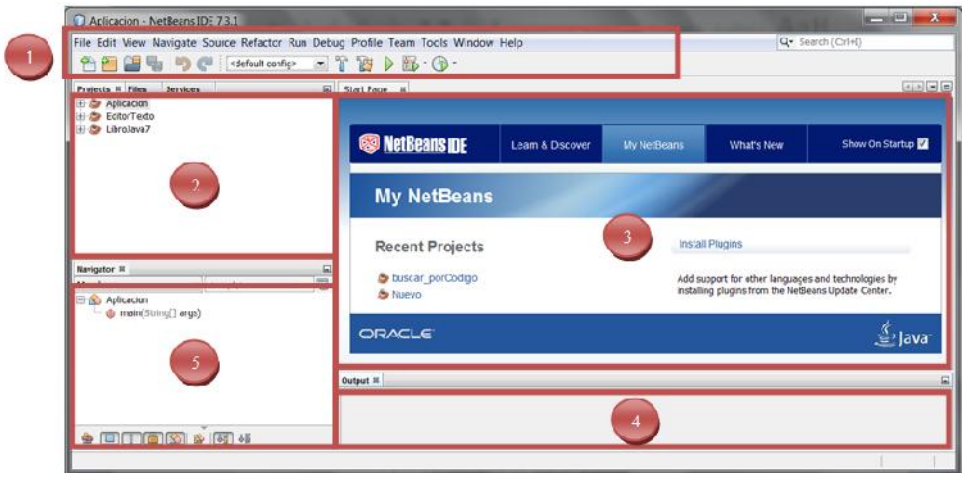


Figura. 29 Ventana principal de NetBeans

Ítem	Descripción
1	Barra de Herramientas
2	Zona de Proyectos
3	Área de Trabajo
4	Salida
5	Zona de Variables

Tabla. 6 Descripción de la ventana principal

3. Para desarrollar la interfaz del sistema de rotulación RGB se crea un nuevo proyecto con clic en la Barra de Herramientas**Archivo\Proyecto Nuevo**. (Ver figura 30)

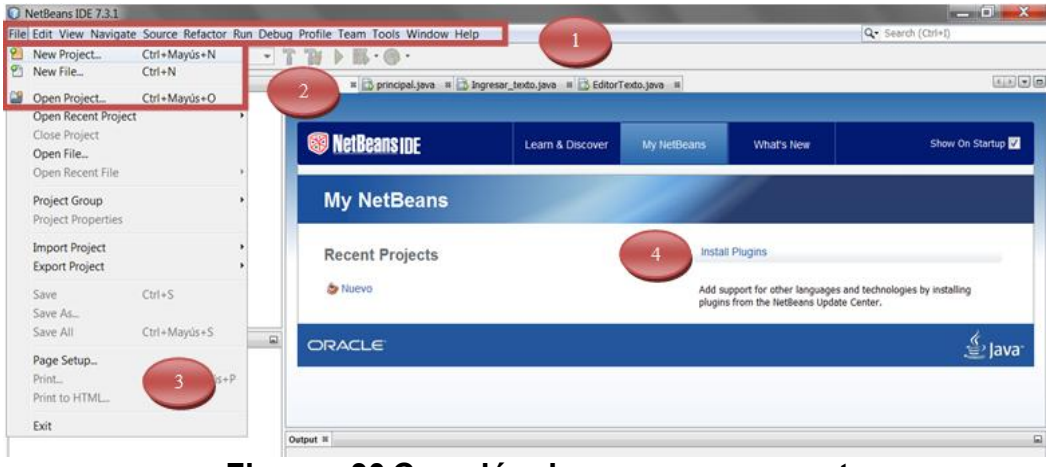


Figura. 30 Creación de un nuevo proyecto

Ítem	Descripción
1	Barra de Herramientas
2	Nuevo Proyecto, Nuevo Elemento, Abrir Proyecto
3	Menú de Archivo
4	Área de Trabajo

Tabla. 7 Descripción de los menús para crear un nuevo proyecto

4. Aparece una nueva ventana donde se indica que tipo de proyecto se desea crear. Entre las alternativas del entorno de desarrollo están aplicaciones en Java, aplicaciones web, aplicaciones para dispositivos móviles y aplicaciones en lenguaje C/C++. En este caso se elige la opción **Aplicación** Java. (Ver figura 31)

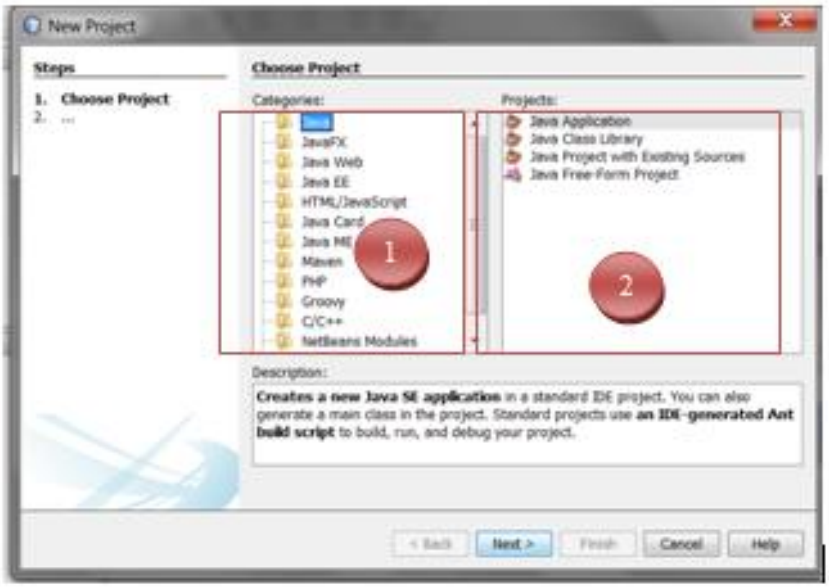


Figura. 31 Tipo de proyectos

Ítem	Descripción
1	Tipo de Proyecto
2	Aplicación del Proyecto

Tabla. 8 Descripción de la ventana del tipo de proyectos

5. Se selecciona la aplicación a utilizar, posteriormente se elige **Siguiente** para visualizar la ventana en donde se ingresa el nombre y la ubicación del proyecto.

6. Con todas las configuraciones listas se selecciona la opción **Terminar.** (Ver figura 32)

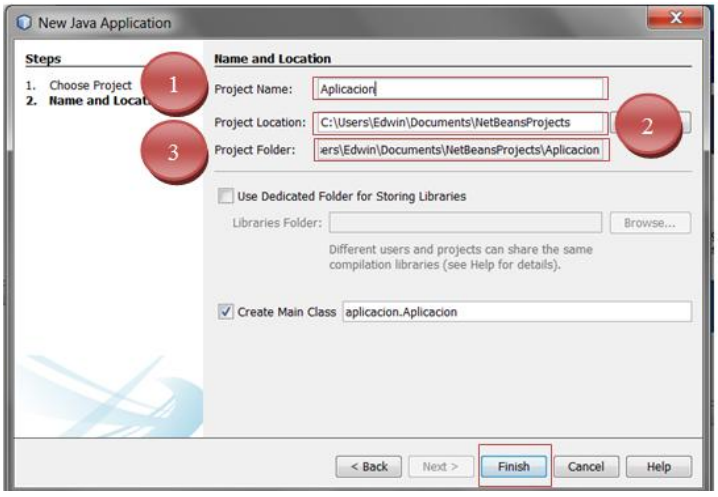


Figura. 32 Ventana nombre y ruta del proyecto

Ítem	Descripción
1	Nombre del Proyecto
2	Ruta del Proyecto
3	Nombre de la carpeta del Proyecto

Tabla. 9 Descripción de la ventana nombre y ruta del proyecto

7. Aparecerá el proyecto en la zona de proyectos. Se desarrolla la aplicación que sirve de interfaz de comunicación entre el usuario y el dispositivo rotativo. (Ver figura 33)

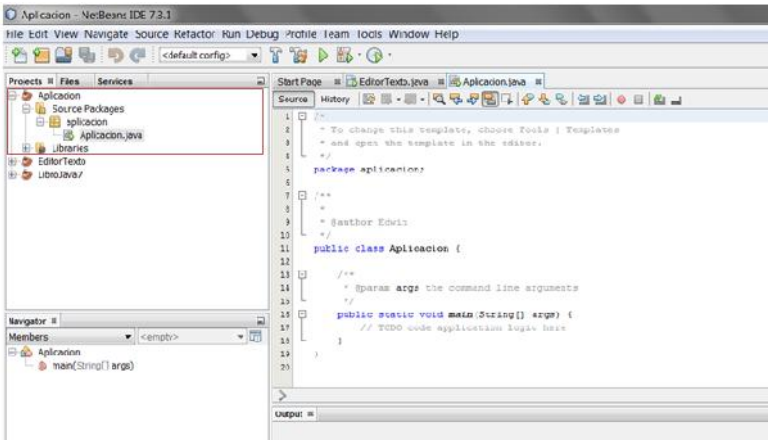


Figura. 33 Proyecto en la zona de proyectos

8. Para la aplicación es necesario un paquete en el que se almacenan las distintas imágenes que se utilizan. Con clic derecho sobre el proyecto se escoge la opción **Nuevo\Paquete Java.** (Ver figura 34)

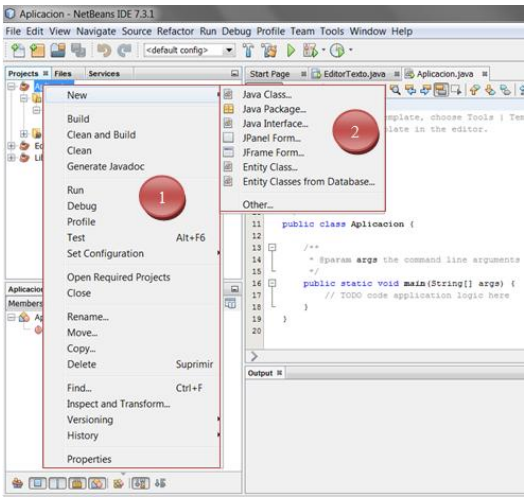


Figura. 34 Creación de un nuevo paquete

Ítem	Descripción
1	Menú del Proyecto
2	Menú de Nuevo

Tabla. 10 Descripción de los menús para crear un nuevo paquete

9. Aparece la pantalla donde se ingresa el nombre y la dirección donde se guardará el paquete.

10. Con las configuraciones listas se elige la opción **Terminar.** El paquete se agrega al proyecto. Para terminar se debe dirigir a la dirección donde se guardó el paquete para guardar todas la imágenes del programa. (Ver figura 35)

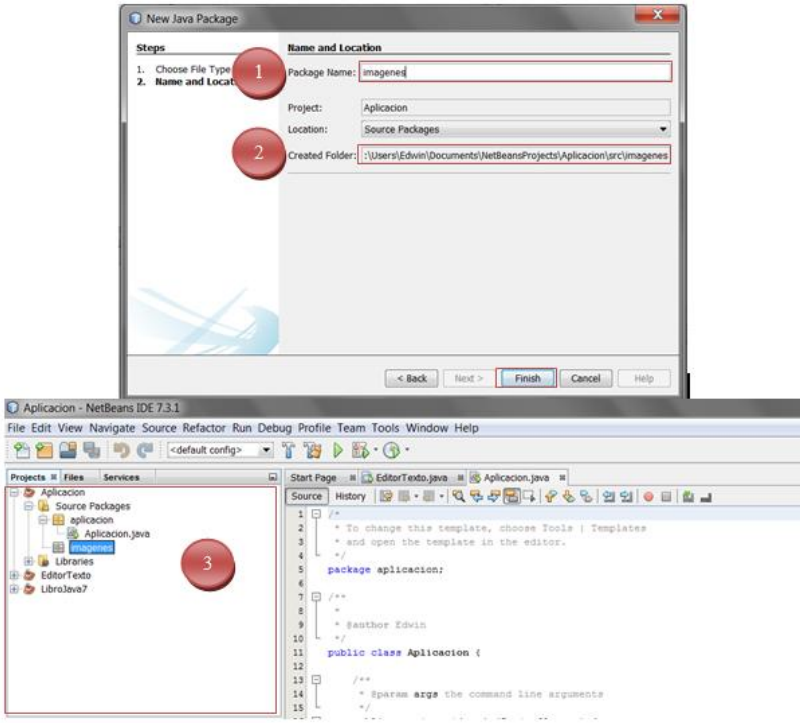


Figura. 35 Ventana nombre y ruta del paquete

Ítem	Descripción
1	Nombre del Paquete
2	Carpeta del paquete
3	Paquete en la Zona de Proyectos

Tabla. 11 Descripción de la ventana nombre y ruta del paquete

11. Otros elementos importantes para el desarrollo del programa son las **Clase Java.** Una clase Java permite desarrollar el código para la comunicación con el dispositivo

rotativo y el código para almacenar el programa en la computadora. Con clic derecho sobre el paquete de la aplicación se escoge la opción **Nuevo\Clase Java**. (Ver figura 36)

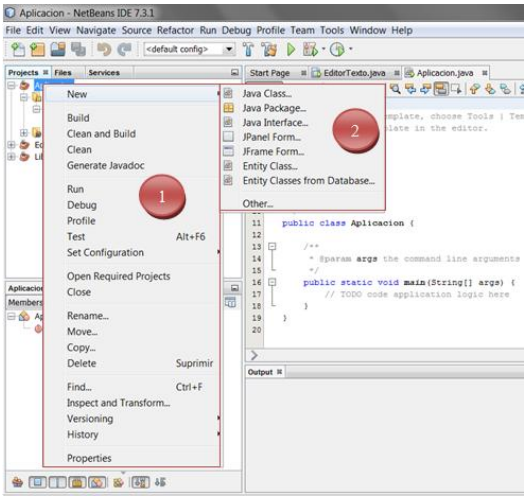


Figura. 36 Creación de una nueva clase

Ítem	Descripción
1	Menú del Paquete
2	Menú de Nuevo

Tabla. 12 Descripción de los menús para crear una nueva clase

12. Se despliega la pantalla donde se ingresa el nombre y la ruta de la clase.

13. Con las configuraciones listas se elige la opción **Terminar.** Una vez lista la clase se puede utilizar desde cualquier entidad del proyecto. (Ver figura 37)

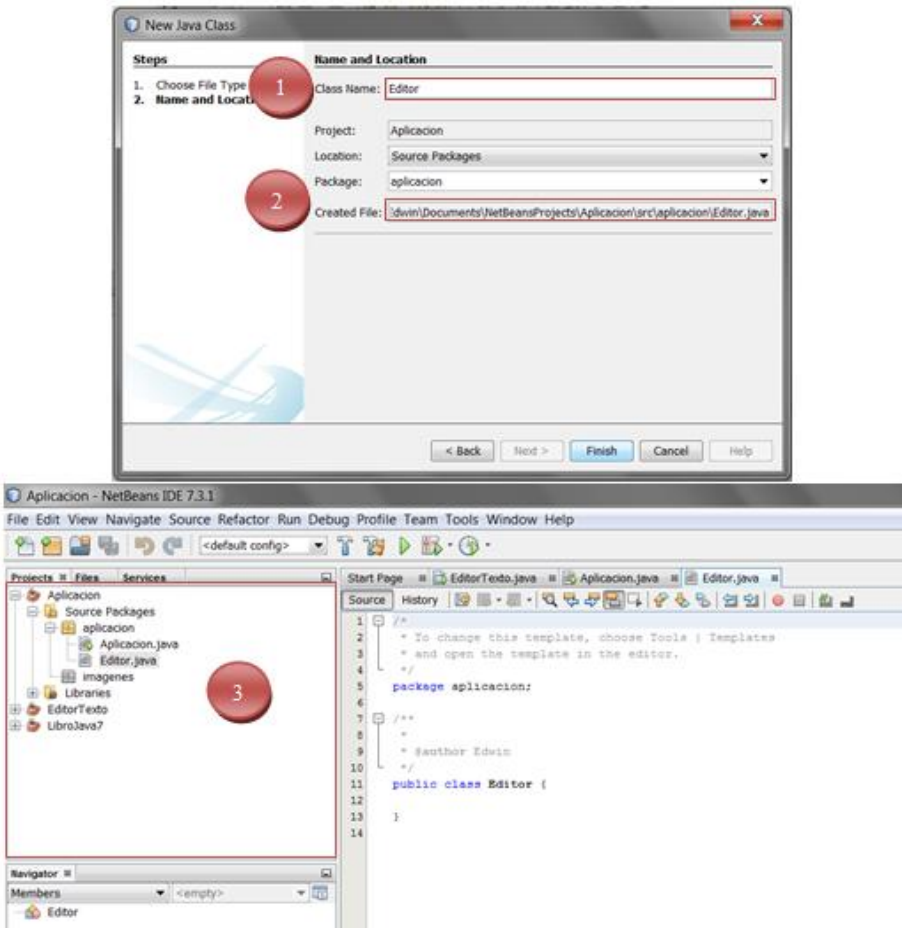


Figura. 37 Ventana nombre y ruta de la clase

Ítem	Descripción
1	Nombre de la Clase
2	Carpeta de la Clase
3	Clase en la Zona de Proyectos

Tabla. 13 Descripción de la ventana nombre y ruta de la clase

14. Se agregan los **JFrame,** que son las ventanas que le permiten al usuario trabajar con la interfaz del sistema de rotulación RGB. Con clic derecho sobre el paquete de la aplicación se escoge la opción **Nuevo\JFrame. (**Ver figura 38)

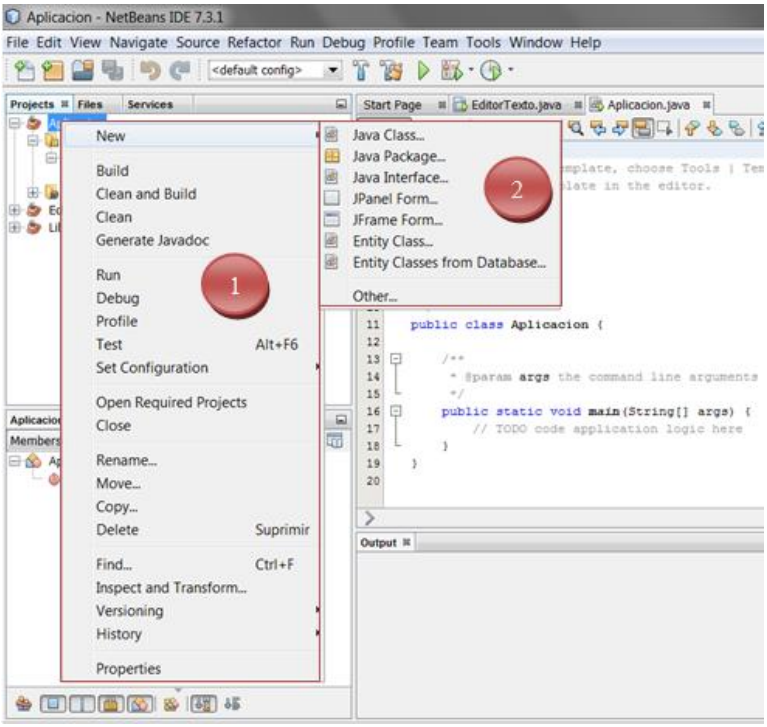


Figura. 38 Creación de un nuevo JFrame

Ítem	Descripción
1	Menú del Proyecto
2	Menú de Nuevo

Tabla. 14 Descripción de los menús para crear un nuevo JFrame

15. Se despliega la pantalla donde se ingresa el nombre y la ruta del JFrame.

16. Con las configuraciones listas se elige la opción **Terminar. (**Ver figura 39)

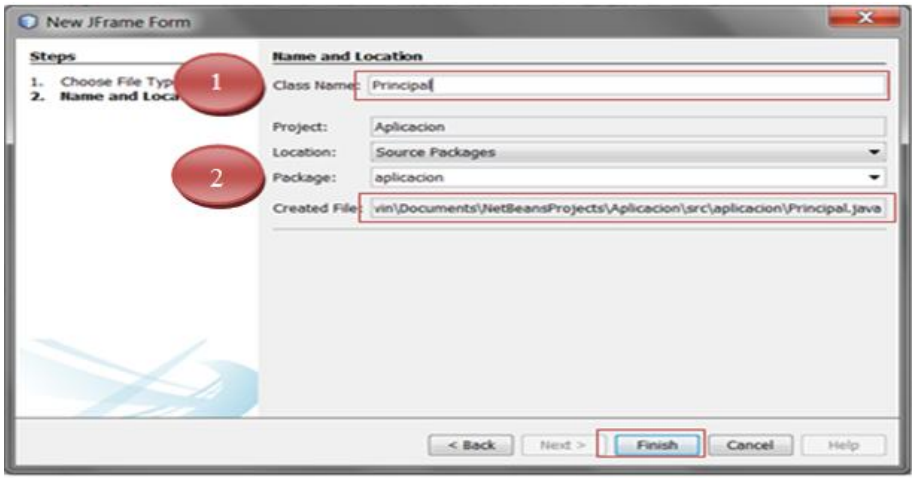


Figura. 39 Ventana nombre y ruta del JFrame

Ítem	Descripción
1	Nombre del JFrame
2	Carpeta del JFrame

Tabla. 15 Descripción de la ventana nombre y ruta del JFrame

17. Se activa la opción de diseño y fuente. La opción de diseño permite diseñar de forma gráfica la pantalla de la interfaz, mientras fuente, nos permite programar la función que tendrá cada uno de los componentes de la interfaz. (Ver figura 40)

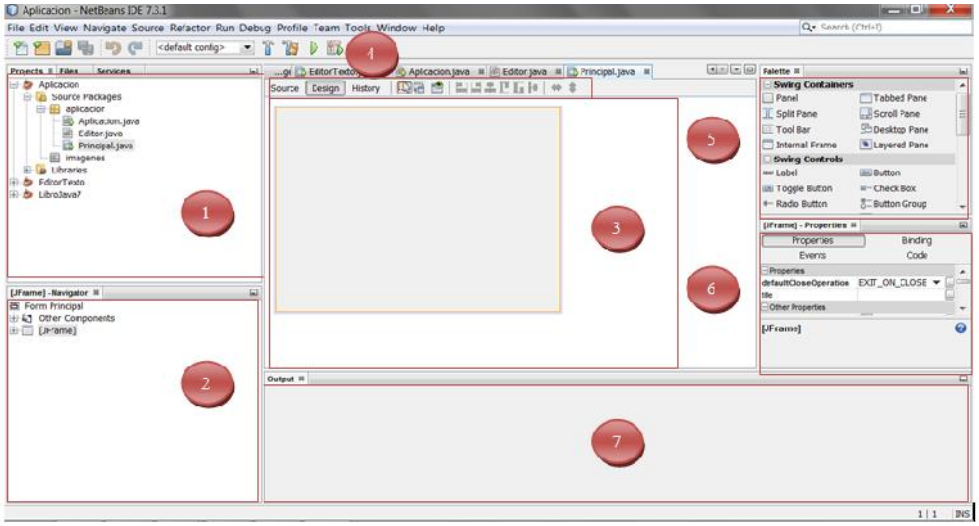


Figura. 40 Ventana principal de un JFrame

Ítem	Descripción
1	JFrame en la Zona de Proyectos
2	Zona del Nombre de los Componentes
3	Área de Diseño
4	Barra de Diseño y Fuente
5	Paleta de Componentes
6	Zona de Propiedades de los Componentes
7	Salida

Tabla. 16 Descripción de la ventana principal de un JFrame

18. Para agregar los elementos a la ventana de diseño se arrastra los componentes de la **Paleta,** entonces se puede programar su funcionalidad. (Ver figura 41)

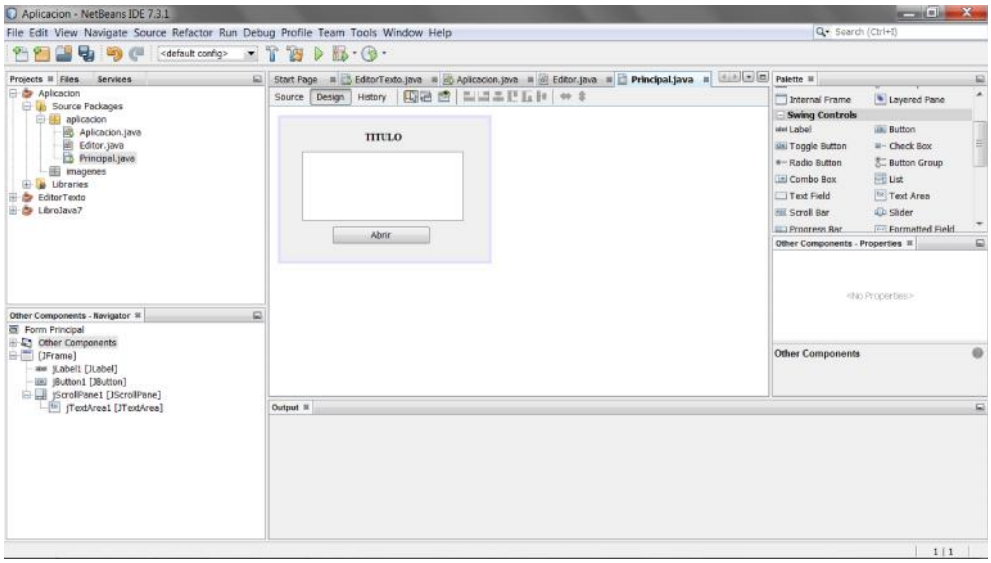


Figura. 41 Creación de los componentes del JFrame

Con la identificación de todos los elementos que permiten desarrollar un programa, se diseña cada una de las ventanas y se relacionan para poder crear un programa sólido que permita la comunicación de una manera efectiva con el sistema de rotulación RGB.

2.5.1.10 Elementos de un programa Java

2.5.1.10.1 Variables

Una variable es un área en memoria que tiene un nombre y un tipo asociado. Es obligatorio declarar las variables antes de usarlas.

Datos	Tipo de dato	Tamaño	Valores
Boolean	Boleano		True o False
Int	Enteros	32-bits	Entre -2147483648 y 2147483647
char	Caracteres		Desde '\u0000' a '\uffff'

Tabla. 17 Tipos de Datos

2.5.1.10.2 Operaciones Básicas

1. Operadores Aritméticos

Operador en Java	Significado
+	Suma
-	Resta
*	Multiplicación
/	División
%	Residuo
++	Incremento (x=x+1)
--	Decremento(x=x-1)

Tabla. 18 Operadores Aritméticos

2. Operadores Relacionales

Operador en Java	Significado
==	Igual
!=	Diferente
<	Menor que
>	Mayor que
<=	Menor o igual que
>=	Mayor o igual que

Tabla. 19 Operadores Relacionales

2.5.1.10.3 Principales Sentencias de Java

2.5.1.10.3.1 Estructuras de selección

Las estructuras de selección permiten modificar el flujo de un programa, esto se debe a que la decisión de ejecutar un bloque de sentencias se condiciona por el valor de una expresión lógica.

1. Estructura if

Ejecuta un bloque de sentencia solo cuando se cumple la condición del if y si no cumple la condición el flujo del programa continúa en la sentencia inmediatamente posterior al if.

Sintaxis

```
if ( condición)
    estatuto;
else            // la parte else es opcional
    estatuto;
```

2.5.1.10.3.2 Estructuras de Repetición

Las estructuras de repetición permiten repetir muchas veces un bloque de sentencias.

1. Estructura while

En el while se ejecutan las sentencias mientras la condición es verdadera, de ser falsa termina el ciclo.

Sintaxis

while (condición)

estatuto;

2. Estructura do – while

En el do-while, las sentencias se ejecutan al menos una vez y luego se verifican las condiciones, si son verdaderas se sigue ejecutando y si son falsas termina el ciclo.

Sintaxis

do

estatuto;

while (condición);

3. Estructura for

La estructura for repite el bloque de sentencias mientras la condición del for sea verdadera.

Sintaxis

for (inicialización ; condición ; acción)

estatuto;

2.5.2 CSC PIC-C versión 4.023

Es un compilador que traduce el código C del archivo fuente (.C) a lenguaje de máquina para los microcontroladores PIC, genera así un archivo en formato hexadecimal

(.HEX) necesario para programar un microcontrolador de 6, 8, 18 ó 40 patitas por medio de un programador de PIC.

Dispone de una amplia librería de funciones predefinidas, comandos pre-procesados, controladores o drivers para diversos dispositivos como LCD, convertidores AD, relojes en tiempo real, entre otros.

2.5.2.1 Tipo de Datos

Tipo	Descripción
unsigned	Define un número de 8 bits sin signo.
int	Define un número de 8 bits sin signo.
char	Define un número de 8 bits sin signo.
long int	Define un número de 16 bits sin signo.
signed	Define un número de 8 bits con signo.
signed long	Define un número de 16 bits con signo.
float	Define un número de 32 bits en punto flotante.
short	Define un bit.

Tabla. 20 Tipo de datos PIC-C

2.5.2.2 Funciones permitidas

Son bloques de sentencias, que deben definirse antes de utilizarse. Una función se invoca desde una sentencia de otra función y puede devolver un valor a la sentencia que la invoca.

Funciones de I/O Serie RS232	
Funciones	**Descripción**
c = GETC() **c = GETCH()** **c = GETCHAR()**	Estas funciones esperan un carácter por la pata RCV del dispositivo RS232 y retorna el carácter recibido.
GETS(char *string)	Lee caracteres (usando GETC()) de la cadena (string) hasta que encuentra un retorno de carro (valor ASCII 13). La cadena se termina con un 0.
PUTC() **PUTCHAR()**	Envían un carácter a la pata XMIT del dispositivo RS232.
PUTS(string)	Envía cada carácter de string a la pata XMIT del dispositivo RS232.
PRINTF([function], string, [values])	Saca una cadena de caracteres al estándar serie RS-232 o a una función especificada. El formato se relaciona con el argumento que se pone dentro de la cadena (string).

Tabla. 21 Funciones de I/O Serie RS232 PIC-C

Funciones de Retardos	
Funciones	**Descripción**
DELAY_CYCLES(count)	Realiza retardos según el número de ciclos de instrucción que se especifican en count; los valores posibles van desde 1 a 255.
DELAY_MS(time)	Realiza retardos del valor que se especifica en time. Dicho valor es en milisegundos y el rango es 0-65535.
DELAY_US(time)	Realiza retardos del valor que se especifica en time. Dicho valor es en microsegundos y el rango va desde 0 a 65535.

Tabla. 22 Funciones de Retardos PIC-C

2.5.2.3 Interrupciones

"Las interrupciones permiten a cualquier suceso interior o exterior interrumpir la ejecución del programa principal en cualquier momento. En el momento de producirse la interrupción, el PIC ejecuta un salto a la rutina de atención a la interrupción, previamente definida por el programador. Cuando se termina de ejecutar dicha rutina, el

PIC retorna a la ejecución del programa principal en la misma posición de la memoria de programa donde se produjo la interrupción[39].”

La familia PIC18FXX2 tiene 18 fuentes de interrupción. El enfoque de la programación está en la interrupción de la USART, cuya directiva es la #INT_RDA.

Estructura Programa	Descripción
#int_RDA	Declaración de la directiva para interrupción de la USART.
void RDA_isr() **{** **}**	Programación de la rutina de la interrupción.
void main (void) **{** **enable_interrupts(INT_RDA);** **enable_interrupts(GLOBAL);**	Habilitación de la las interrupciones.
while (true) **{** **}** **}**	Programación de la rutina del programa principal.

Tabla. 23 Estructura del programa con interrupciones

2.5.2.4 Directivas para configuración del programa

1. #USE DELAY (CLOCK=frecuencia): Indica al compilador la frecuencia del procesador en ciclos por segundo, a la vez que habilita el uso de las funciones DELAY_MS() y DELAY_US().

2. #USE FAST_IO (puerto): Ocasiona que el compilador realice E/S sin programar el registro de dirección.

[39] Información obtenida de: Garcia, Eduardo. Compilador C CCS y simulador Proteus para microcontroladores PIC. Alfaomega Grupo Editor, S.A de C.V, México. Pág. 83.

3. #USE FIXED_IO (puerto_OUTPUTS=pin_x#, pin_x#...): Causa que el compilador genere código para hacer que un pin de E/S sea entrada o salida cada vez que se utiliza. Esto ahorra el byte de RAM que se usa en E/S normal.

4. **#USE STANDARD_IO (puerto):** Causa que el compilador genere código para hacer que un pin de E/S sea entrada o salida cada vez que se utiliza, sin necesidad de configurar los TRIS.

5. #USE RS232 (BAUD=baudios, XMIT=pin, RCV=pin): Da al compilador la velocidad en baudios y los pines que se utilizan para la E/S de comunicación serial. Habilita el uso de funciones tales como: GETCH, PUTCHAR y PRINTF.

La directiva **#USE DELAY** debe aparecer antes de utilizar **#USE RS232**.

2.5.2.5 Entorno de trabajo CCS-PICC

Permite compilar los programas que se desarrollaron y suministra una gran variedad de herramientas auxiliares para facilitar el trabajo del programador. Cuando se abre el compilador aparece la ventana de trabajo principal. (Ver figura 42)

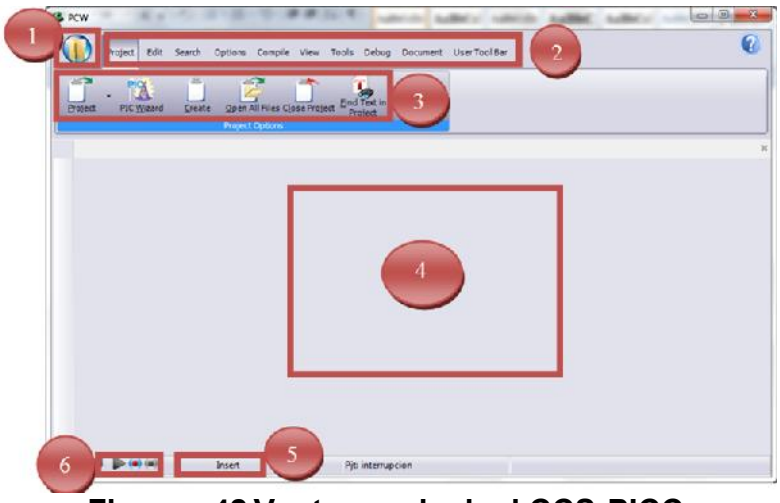


Figura. 42 Ventana principal CCS-PICC

Ítem	Descripción
1	Comando de Manejo de Ficheros
2	Barra de Comandos
3	Barra de SubComandos
4	Zona de Código
5	Barra de Información
6	Macros

Tabla. 24 Descripción ventana principal CCS-PICC

Para desarrollar un programa en el entorno de trabajo CCS-PICC los pasos a seguir son:

1. Dar clic en el icono del PIC-Wizard para desplegar una ventana que permite definir el tipo de microcontrolador a emplear, velocidad del cristal y diferentes directivas a emplear como: interrupciones, LCD, entre otras. (Ver figura 43)

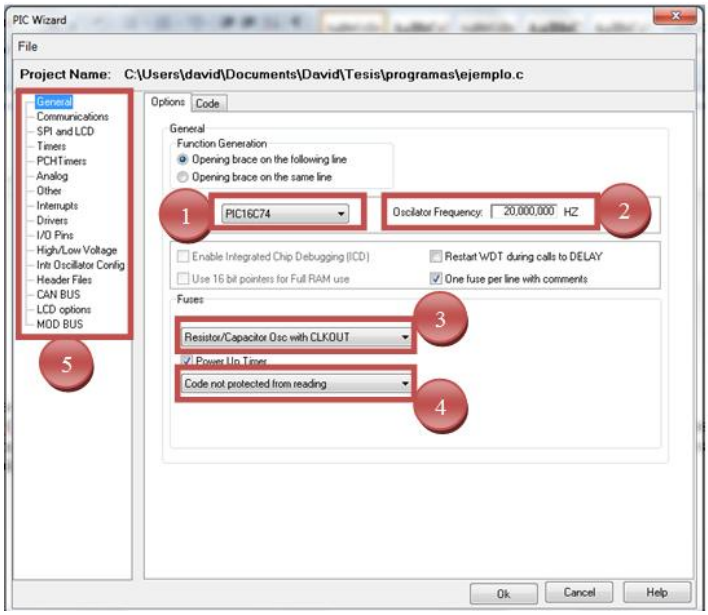


Figura. 43 Configuración PIC-Wizard

Ítem	Descripción
1	Selección del tipo de microcontrolador
2	Velocidad del Oscilador
3	Tipo de Oscilador
4	Tipo de protección del código
5	Selección de directivas a emplear en el desarrollo del programa

Tabla. 25 Descripción ventana configuración PIC-Wizard

2. Con las configuraciones listas en el PIC-Wizard, se da clic en Aceptar para volver a la ventana principal. En la zona de código (Ver figura 42), se muestran todas las directivas que se seleccionaron para el desarrollo del programa. (Ver figura 44)

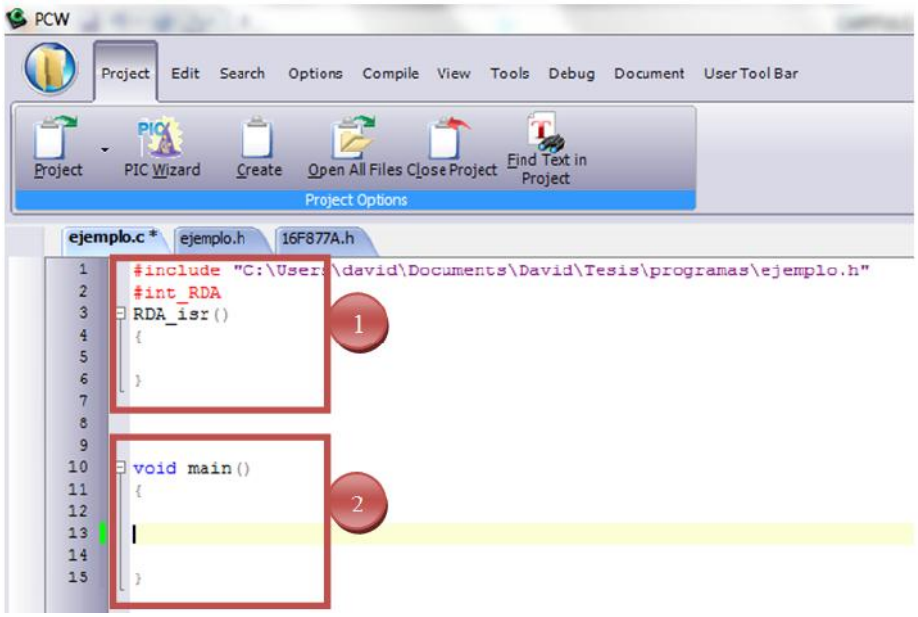


Figura. 44 Ventana desarrollo programa

Ítem	Descripción
1	Directivas seleccionadas en la configuración
2	Zona para el desarrollo del programa

Tabla. 26 Descripción ventana desarrollo programa

3. Cuando se culmina el programa, se compila para la verificación de errores. Si no existen errores se crea un archivo con la extensión .HEX, que contiene el programa para

cargarse al microcontrolador PIC por medio de un programador. Si al compilar se presenta algún tipo de error este se detalla en la barra de información. (Ver figura 45)

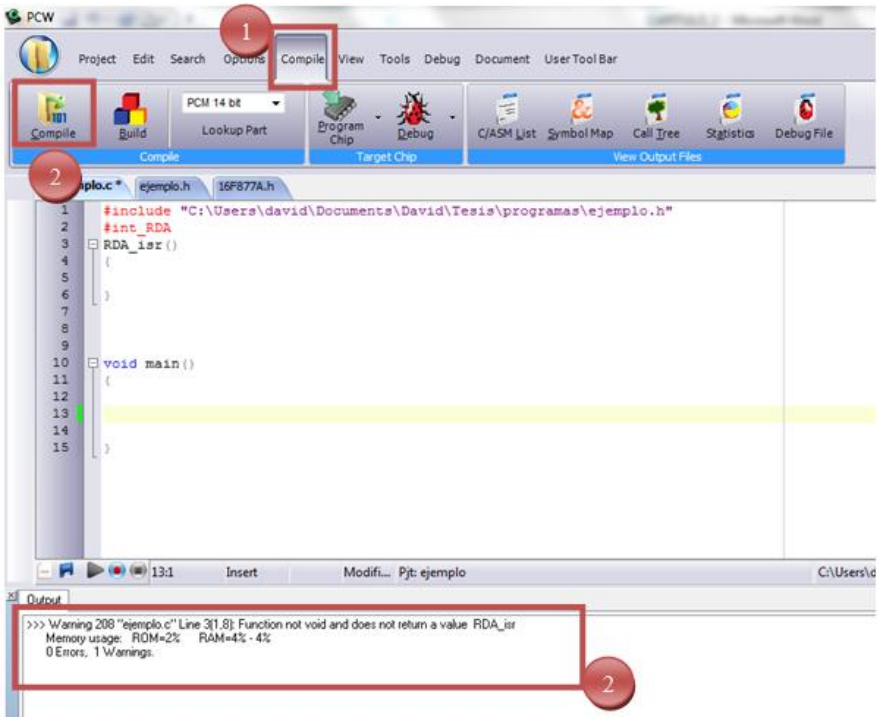


Figura. 45 Compilar el programa

Ítem	Descripción
1	Barra de Comandos Compile
2	Barra de Información
3	Compilador

Tabla. 27 Descripción compilar el programa

CAPÍTULO 3

DISEÑO DE HARDWARE

3.1 DIAGRAMA DE BLOQUES DE LA IMPLEMENTACIÓN DEL SISTEMA DE ROTULACIÓN RGB

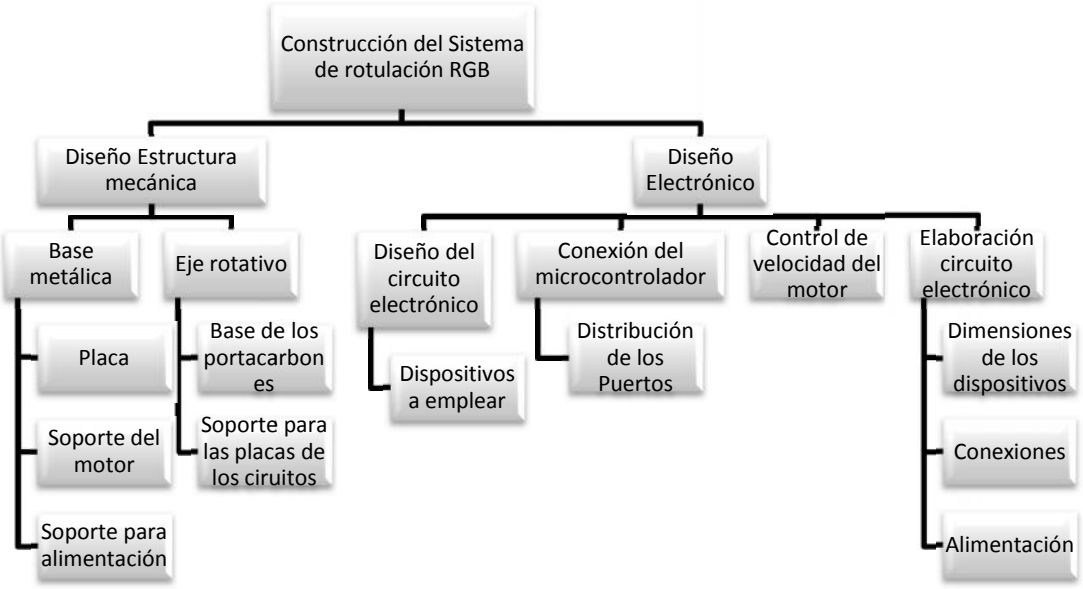


Figura. 46 Diagrama de bloques Sistema de Rotulación RGB

En el diagrama de bloques se muestra la etapa del diseño de la estructura mecánica, que contempla el diseño y la creación del soporte mecánico para el motor y los dispositivos electrónicos del sistema de rotulación RGB, y la etapa del diseño

electrónico, que abarca la conexión y alimentación de los dispositivos electrónicos: microcontrolador, led's RGB, dispositivos inalámbricos, entre otros.

3.2 DISEÑO DE LA ESTRUCTURA MECÁNICA

La estructura mecánica tiene la capacidad de soportar la hélice con las placas de circuitos, brinda alimentación y estabilidad al sistema de rotulación RGB.

3.2.1 Base metálica

Los problemas que solventa la base metálica son:

- Dar estabilidad al sistema de rotulación RGB cuando empieza su movimiento giratorio.
- Brindar soporte al motor y alimentación al sistema de rotulación RGB.

3.2.1.1 Placa

Para soportar el movimiento giratorio del sistema de rotulación RGB se crea una placa de 20cm x 20cm en metal galvanizado para evitar su corrosión. (Ver figura 47)

Figura. 47 Placa metálica

En la placa se refuerza los bordes, para ello se colocan cuatro tiras de aluminio como bases. (Ver figura 48)

Figura. 48 Placa metálica con bases

3.2.1.2 Soporte del motor

En el mercado ecuatoriano hay una gran variedad de motores de corriente continua y alterna. Entre los dos tipos se escoge el motor de corriente alterna que trabaja a altas revoluciones y posee más torque que los motores de corriente continua, sin embargo, la mayor limitante es el tamaño que este posee. Por tanto, para la selección del motor se toma en cuenta, su tamaño, voltaje de operación, las revoluciones, el torque y la carga que consume. Entonces, se utiliza un motor de corriente alterna que se emplea en máquinas de coser o bordadoras cuyo tamaño es adecuado para situarse en el sistema de rotulación RGB. (Ver tabla. 28)

Marca	FMD (sewing machine motor)
Modelo	FM 10100
Tensión y frecuencia de trabajo	110 [VAC] – 50/60 [HZ]
Corriente	1.0 [A]
Potencia	100 [W]
Velocidad	6000/7000 [RPM]

Tabla. 28 Características del motor

Con las medidas del motor de corriente alterna (Ver figura 49) se sitúan con tornillos galvanizados (Ver tabla 29), patas en forma de L que actúan como palancas de seguridad y soporte en el centro de la placa. (Ver figura 50)

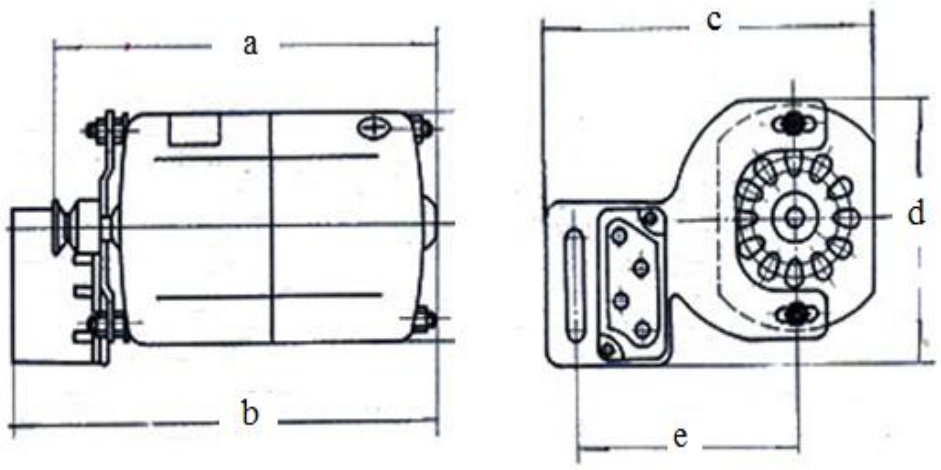


Figura. 49 Medidas motor[40]

Ítem	Dimensión
a	120[mm]
b	134[mm]
c	110[mm]
d	73.5[mm]
e	78[mm]

Tabla. 29 Dimensiones motor

40 http://www.lionball.net/Sewing-Machine-Motor/Sewing-Machine-Motor---HF.html, Figura.3.4 Medidas motor

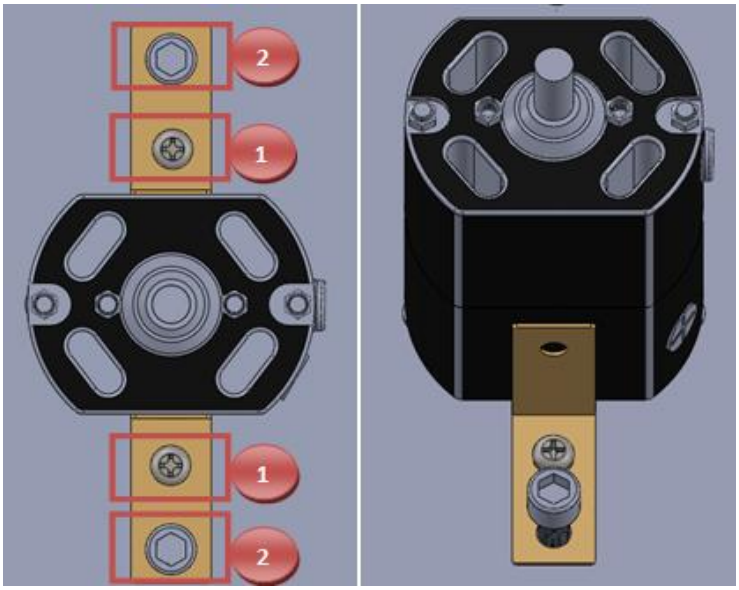


Figura. 50 Motor empotrado en la base

Ítem	Cantidad	Tornillo (mm)	Broca (in)	Machuelo
1	2	M5	3/16	-
2	2	M8 Allen	17/64	M8[41]

Tabla. 30 Descripción tornillos motor

3.2.1.3 Soporte para alimentación

Se implementa un disco en una baquelita que contiene dos anillos de 46x54x1.25mm y 52x60x1.25mm de cobre, donde se conecta el positivo y negativo de la fuente de alimentación para transferir la alimentación de energía desde la placa hacia el eje rotativo RGB. (Ver figura 51)

41 http://corteymedicion.com/site/parametros/machos-diametrohueco.php, Tabla 30 Descripción tornillos motor

Figura. 51 Anillos de cobre

El soporte para la alimentación de energía hacia la hélice se empotra con tornillos en la parte superior del motor. (Ver figura 52)

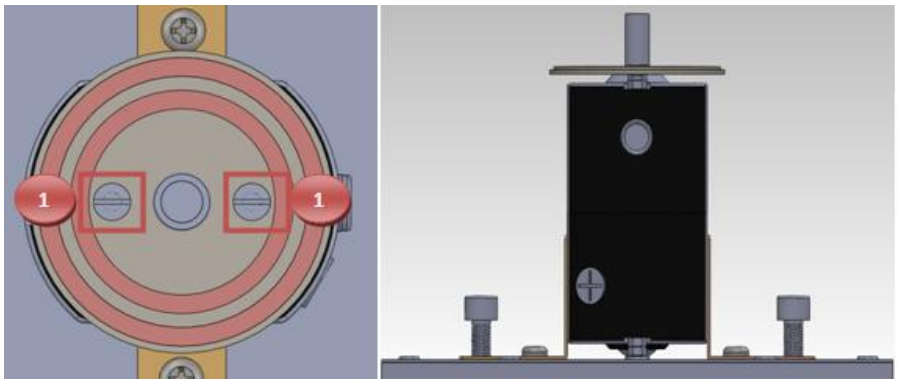


Figura. 52 Soporte para la alimentación

Ítem	Cantidad	Tornillo (mm)	Broca (in)	Machuelo
1	2	M4	30	M4

Tabla. 31 Tornillos soporte para la alimentación

3.2.2 Eje rotativo

La integración del eje rotativo con el motor y la base metálica es un pilar fundamental en la implementación del sistema de rotulación RGB, puesto que es el punto de acople de todos los dispositivos electrónicos que permiten la visualización de

los mensajes. El material que se emplea para la construcción del eje rotativo es una tira de aluminio de 56cm de largo, que se dobla en cuatro partes: dos partes con 20.5cm y las dos restantes con 6.5cm. (Ver figura 53)

Figura. 53 Eje rotativo

3.2.2.1 Base de los portaescobillas

Para la construcción de la base de los portaescobillas se utiliza plástico industrial con el que se elabora un disco de 80mm de diámetro y 10mm de alto, que posteriormente se sujeta con tornillos en la hélice. (Ver figura 54)

Figura. 54 Base portaescobillas

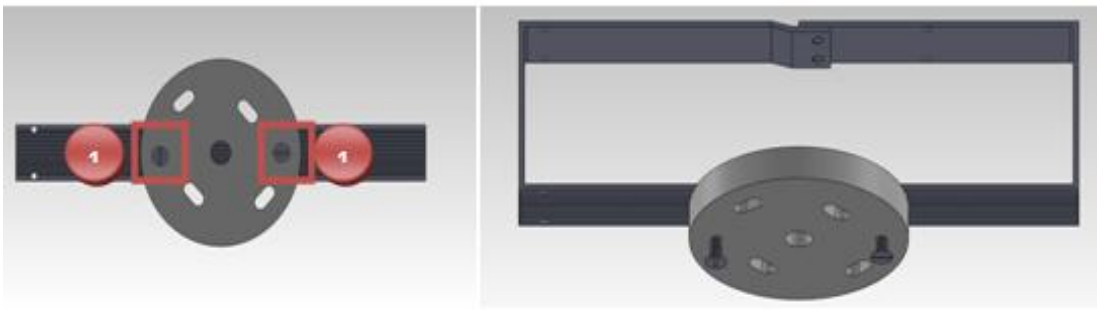


Figura. 55 Base portaescobillas acoplada al eje

Ítem	Cantidad	Tornillo (mm)	Broca (in)	Machuelo
1	2	M4	30	M4

Tabla. 32 Tornillos base portaescobillas

Se realizan cuatro agujeros en la base plástica para introducir los portaescobillas. La distancia de los agujeros se da por los anillos de cobre. (Ver figura 56)

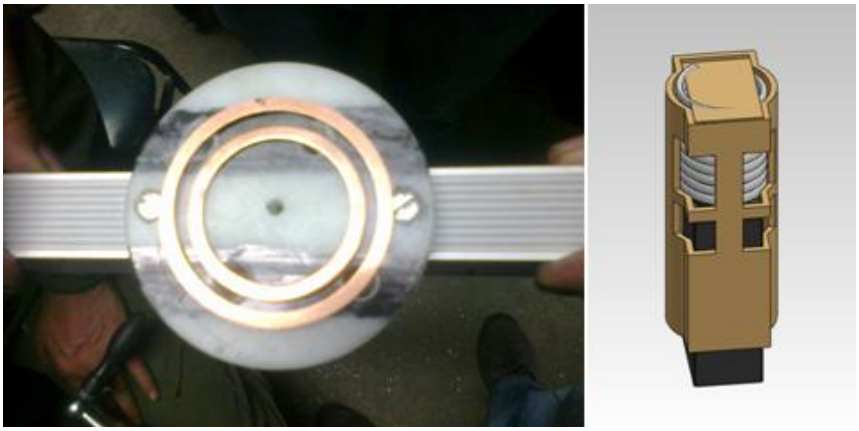


Figura. 56 Agujeros para los portaescobillas

Al término de la elaboración de las piezas se procede a ensamblar el eje rotativo y a colocar los portaescobillas. (Ver figura 57)

Figura. 57 Hélice completa

3.3 DISEÑO ELECTRÓNICO

Para el diseño electrónico se toman en cuenta las necesidades que presenta el sistema de rotulación RGB: el manejo de 16 LED RGB, cambio de mensajes inalámbricamente, entre otros.

3.3.1 Diseño del circuito electrónico

3.3.1.1 Microcontrolador PIC18F452

Para la selección del microcontrolador PIC18F452, se toma en cuenta la capacidad de las memorias, velocidad de funcionamiento, el número de líneas de entrada/salida y tipo de comunicación, entre otros. (Ver tabla. 33)

Características principales del PIC18F452	
Memorias	32Kb de memoria Flash de programa. 1536 bytes de memoria de datos (RAM). 256 bytes de memoria de datos EEPROM.
Interrupciones	18 interrupciones internas y externas
Consumo de Energía	Rango de voltaje de operación de 2.0 a 5.5 volts. Alta disipación de corriente de la fuente: 25mA.
Puertos A, B, C, D y E	Total 33 líneas de entrada/salida.
Puertos Serie	Puerto serie universal USART.
Frecuencia de operación	Frecuencia de operación de 0 a 40 MHz.

Tabla. 33 Características principales del PIC18F452

Para la selección del cristal a emplear se toma en cuenta el tiempo que utiliza un ciclo de máquina para cada oscilador, el cálculo del tiempo de los ciclos de máquina para un cristal de 4MHz y20MHz se muestran a continuación:

Cristal de 4MHz

$$T = \frac{1}{4 \square 10^{6}} \square 4 = 1[us]$$

Cristal de 20MHz

$$T = \frac{1}{20 \square 10^{6}} \square 4 = 0.2[us]$$

Por tanto se emplea un cristal de 20MHz para que la lectura del código del programa del microcontrolador sea rápida porque un ciclo de maquina se ejecuta en 0.2[us] al contrario de un cristal de 4MHz que se ejecuta en 1[us].

3.3.1.2 Circuitos Integrados DM74154

Para la administración de 16 LED RGB se necesitan 48 pines de salidas para el manejo de los colores rojo, verde y azul, por lo que es indispensable emplear demultiplexores pues el microcontrolador PIC18F452 no cuenta con suficientes líneas de entradas y salidas.

Los demultiplexores que se usan son DM74154, que por medio de cuatro señales de entrada y una señal adicional para el control, se obtienen 16 salidas. (Ver tabla 34)

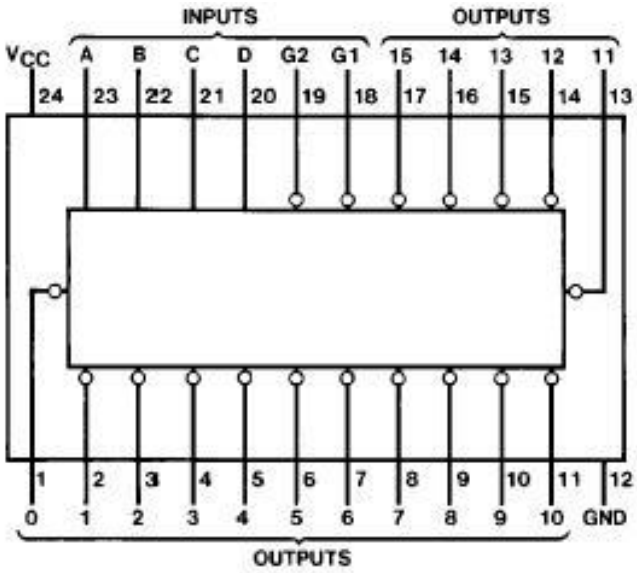


Figura. 58 Circuito Integrado DM74154

ENTRADAS						SALIDAS															
G	G2	D	C	B	A	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
L	L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	L	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H
L	L	L	H	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H
L	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H
L	L	H	L	L	L	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H
L	L	H	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H
L	L	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H
L	L	H	H	L	L	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H
L	L	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	H	-	-	-	-	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
H	L	-	-	-	-	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
H	H	-	-	-	-	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H

Tabla. 34 Tabla de verdad integrado DM74154[42]

42 http://pdf.datasheetcatalog.net/datasheets/120/236595_DS.pdf, Tabla 34 Tabla de verdad integrado DM74154

Se procede a implementar la conexión de 3 circuitos integrados DM74154, cada uno destinado al manejo de un solo color de los LED RGB.

3.3.1.2.1 Diseño del circuito para el Integrado DM74154

Se necesitan cuatro señales de entrada que se emplean para el ingreso de datos y una señal de control que habilita o deshabilita las salidas. (Ver figura 59)

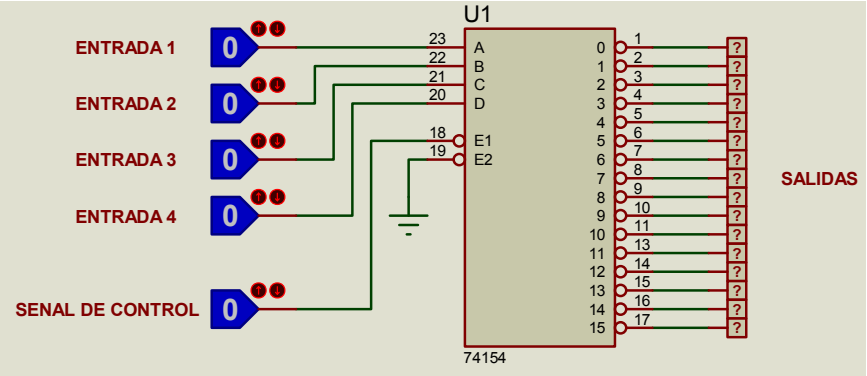


Figura. 59 Conexión integrado DM74154

Las señales de entrada son una combinación de números binarios que van en el rango de 0 a 15, el valor del número binario que se ingresa se refleja en las salidas en forma de posiciones, es decir, con el valor 0000 la primera salida se pone en nivel bajo y el resto mantiene su nivel en alto, mientras que la señal de control toma valores de 1 para deshabilitar las salidas y 0 para habilitar las salidas. (Ver figura 60 y figura 61)

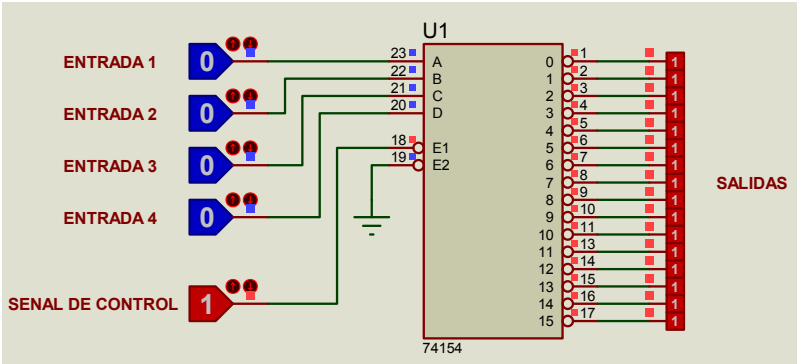


Figura. 60 Salidas deshabilitadas del integrado DM74154

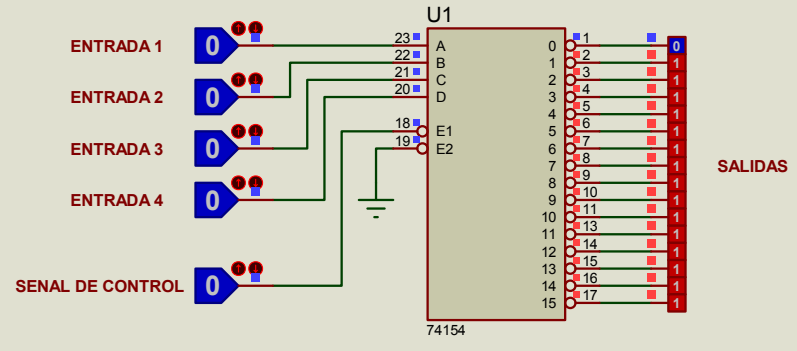


Figura. 61 Salidas habilitadas integrado DM74154

3.3.1.3 Comunicación Inalámbrica Bluetooth

Es indispensable emplear un dispositivo inalámbrico para la movilidad del sistema de rotulación RGB, para ello se utiliza la tecnología Bluetooth que es un protocolo de comunicaciones que se desarrolla específicamente para dispositivos de bajo consumo de energía y que requieren corto alcance de emisión. Posee la capacidad de emitir una señal a una distancia corta de hasta 100 metros y funciona en la banda de radio de los 2,4 GHz.

3.3.1.3.1 Módulos HC-05

Se emplean dispositivos HC-05 que manejan comunicación Bluetooth SPP para el desarrollo de la comunicación inalámbrica del sistema de rotulación RGB. Estos módulos tienen funcionamiento como dispositivo maestro o esclavo.

Características de Hardware[43]

- Típico-80dBm sensibilidad
- Hasta +4 dBm de potencia de transmisión RF
- Low Power 1.8V Operación, de 1,8 a 3,6 I / O
- UART interfaz con la velocidad en baudios programable
- Antena integrada

Características de Software[44]

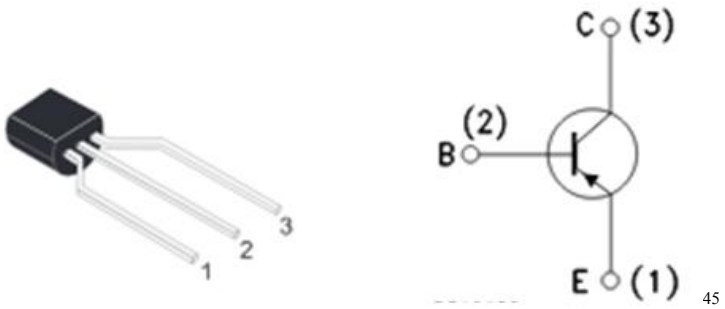
- Velocidad en baudios predeterminada: 38400, Bits de datos: 8, Bit de parada: 1, Paridad: Sin paridad, control de datos: tiene.
- Soporta tasa de baudios: 9600, 19200, 38400, 57600, 115200, 230400, 460800.
- Cuando el maestro y el esclavo se emparejan el LED rojo parpadea en intervalos de 2s y si no se conectan parpadea 2 veces por segundo.

3.3.1.4 Transistores 2n3906

Las salidas de los integrados DM74154 se activan en nivel bajo por lo que se emplean transistores PNP para la activación de los LED RGB. (Ver tabla. 35)

43 ftp://imall.iteadstudio.com/IM120417010_BT_Shield_v2.2/DS_BluetoothHC05.pdf

44 ftp://imall.iteadstudio.com/IM120417010_BT_Shield_v2.2/DS_BluetoothHC05.pdf



[45]

Figura. 62 Transistor 2N3906

Símbolo	Parámetros	Valores	Unidades
	Colector-Base Voltaje (	-60	V
	Colector-Emisor Voltaje (	-40	V
	Emisor-Base Voltaje (	-6	V
	Corriente Colector	-200	mA
	Disipación total a 25	625	mW
	Temperatura de almacenamiento	-65 a 150	
	Temperatura de operación max. en la juntura	150	

Tabla. 35 Parámetros del transistor 2n3906[46]

La configuración que se emplea en los transistores es la de interruptor, por lo que se trabaja en la zona de corte y saturación. (Ver figura 63)

1. Se calcula la corriente de base que provoca la saturación del transistor. A esta corriente se aumenta un 30% para garantizar la saturación del transistor. (Ver tabla 36)

$$I_{B(min)} = {}^{I_{Carga}}/_{HFE} \quad [47]$$

45 http://www.ece.rice.edu/~jdw/data_sheets/2N3906.pdf, Figura 62 Transistor 2N3906

46 http://www.ece.rice.edu/~jdw/data_sheets/2N3906.pdf, Tabla 35 Parámetros del transistor 2N3906

47 Thomas L. Floyd. Dispositivos Electrónicos. Octava Edición. Capítulo 4 Pág. 182-185 y 202.

Variable	Descripción	Unidades
$I_{B(min)}$	Corriente de base mínima para saturación	[A]
$I_{C(min)}$	Corriente de carga	[A]
β_{cd}	Ganancia de corriente en DC del transistor	

Tabla. 36 Fórmula corriente de base

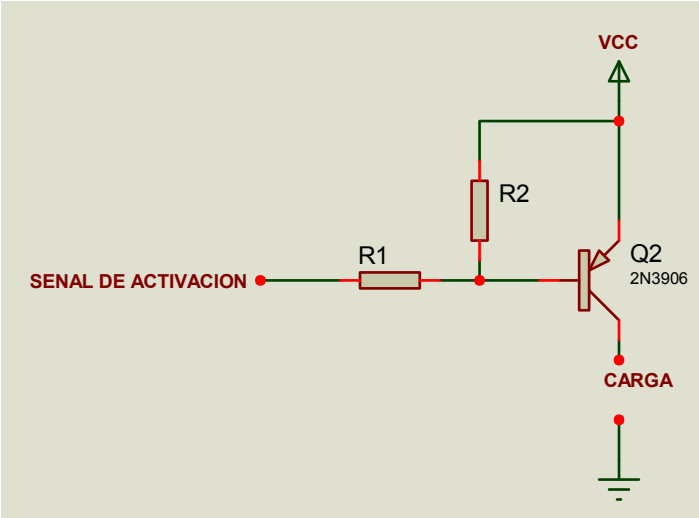


Figura. 63 Conexión modo interruptor del transistor 2N3906

2. Se calcula el valor de la resistencia de base R1.
3. Se calcula el valor de la resistencia R2, que se emplea para dar estabilidad al circuito e impide que las señales bajas que ingresan por la base provoquen que fluya una cantidad pequeña de corriente por el colector. Generalmente se la emplea con 10 veces el valor de R1.

$$R_1 = \frac{V_{sw}}{(I_{B(min)} * 1.3)} \quad [48]$$

$$R_2 = 10 \square R_1 \quad [49]$$

48 Thomas L. Floyd. Dispositivos Electrónicos. Octava Edición. Capítulo 4 Pág. 182-185 y 202.

49 Thomas L. Floyd. Dispositivos Electrónicos. Octava Edición. Capítulo 4 Pág. 182-185 y 202.

Variable	Descripción	Unidades
	Voltaje de conmutación	[V]
	Corriente de base mínima para saturación	[A]
1.3	30% más de la corriente de base mínima	-
	Resistencia base	[Ω]
	Resistencia base-emisor	[Ω]

Tabla. 37 Fórmula cálculo de resistencias

3.3.1.4.1 Diseño del circuito para los transistores 2n3906

Se calcula la corriente de saturación del transistor para diseñar la configuración en modo interruptor.

$$I_{Bm} = {}^{I_{Carga}}/_{HFE}$$

$$I_{Bm} = \frac{30mA}{80} = 0.375mA$$

Con la corriente de saturación se procede a calcular los valores de resistencias.

$$R_1 = \frac{V_{sw}}{(I_{B(min)} * 1.3)}$$

$$R_1 = \frac{5Vdc}{(0.375mA \square 1.3)} = 10.25k\Omega \approx 10k\Omega$$

$$R_2 = 10 \square R_1 = 100k\Omega$$

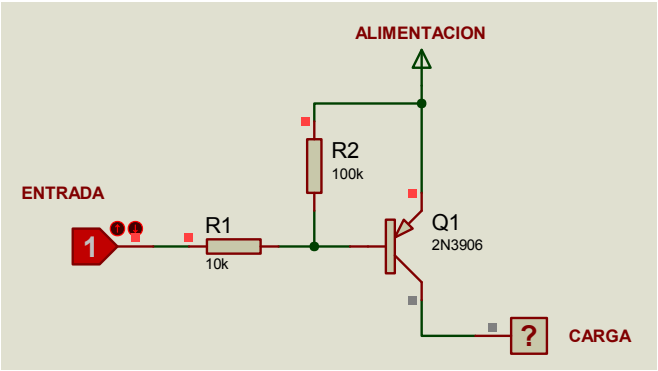


Figura. 64 Simulación transistores en modo interruptor

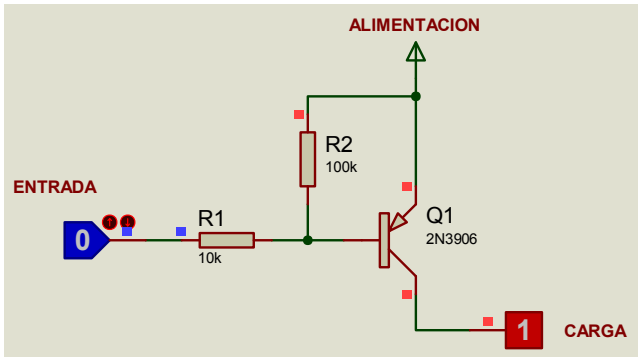


Figura. 65 Simulación transistores en modo interruptor

3.3.1.5 LED RGB SMD

Se pueden obtener siete distintos colores al trabajar con LED RGB. (Ver figura 66 y tabla 38)

50

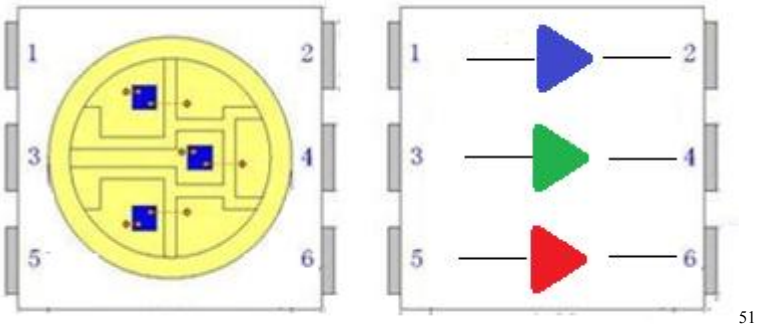
Figura. 66 Combinación de colores

50 http://proyectoplasticaprimarialossimpsons.blogspot.com/2013/04/vision-de-los-colores-beatriz-perez.html, Figura 66 Combinación de colores

Combinación Colores	Resultado
+	
+	
+	
+	
+	
+	
+ +	

Tabla. 38 Combinación de colores

Es indispensable conocer las tensiones de alimentación y el consumo de cada LED para el diseño del circuito. Como se trabaja con LED RGB, se tiene un LED rojo, verde y azul y cada uno tiene un valor de tensión y consumo de corriente. (Ver tabla 39)



[51]

Figura. 67 LED RGB SMD

LED	Tensión	Corriente
Rojo	2.1 V	20 mA
Verde	3.3 V	20 mA
Azul	3.3 V	20 mA

Tabla. 39 Tensiones y corrientes LED RGB SMD[52]

51 http://spanish.alibaba.com/product-gs/plcc6-5050-series-3chips-white-surface-high-brightnesss-white-type-smd-led-308348283.html

52 http://www.sharatronica.com/formulas_para_led.html, Tabla 39 Tensiones y corrientes LED RGB

Se calcula el valor de la resistencia limitadora en base a los valores de tensión y corriente, esto ayuda para que el LED funcione adecuadamente y prolonga su vida útil.

$$R_L = \frac{(V_{CC} - V_{TL})}{I_T}$$

Variable	Descripción	Unidades
	Resistencia limitadora	[Ω]
	Alimentación	[V]
	Tensión del LED	[V]
	Corriente de trabajo	[A]

Tabla. 40 Cálculo resistencia LED RGB SMD

3.3.1.5.1 Diseño del circuito para los LED

1. Cálculo del LED rojo:

$$R_L = \frac{(5Vcd - 2.1)}{20mA} = 145\Omega \approx 150\Omega$$

2. Cálculo del LED azul y verde:

$$R_L = \frac{(5Vcd - 3.3)}{20mA} = 85\Omega \approx 100\Omega$$

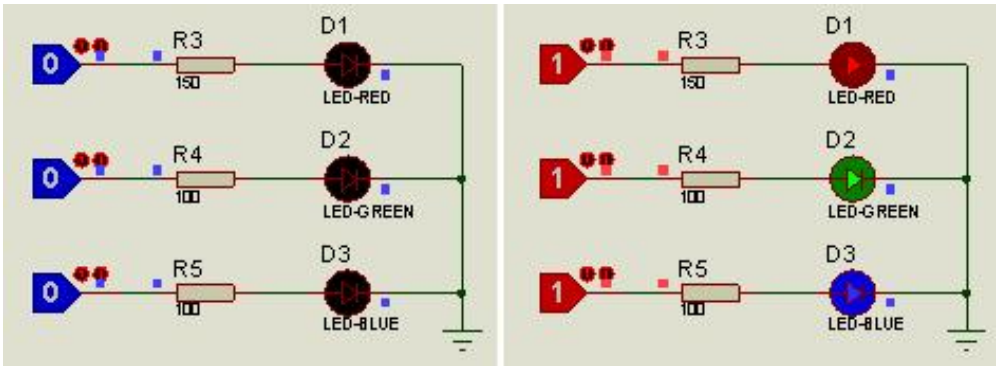


Figura. 68 Simulación LED RGB SMD

3.3.1.6 Diseño del circuito total

Con los diseños del circuito por separado, se integran en un solo circuito.

3.3.2 Conexión del microcontrolador

Con el diseño del circuito eléctrico se designa los pines de los diferentes puertos del microcontrolador PIC18F452, que dan las señales de entrada y control para los circuitos integrados DM74154 y las señales de comunicación para el módulo inalámbrico HC-05. (Ver tabla 41)

PIC18F452		Destino	Descripción
Puerto	Pines		
A	2	Pin 18 del tercer integrado DM74154.	Proporciona la señal de control para habilitar o deshabilitar el encendido de los LED's rojos.
	3	Pin 18 del segundo integrado DM74154.	Proporciona la señal de control para habilitar o deshabilitar el encendido de los LED's verdes.
	4	Pin 18 del primer integrado DM74154.	Proporciona la señal de control para habilitar o deshabilitar el encendido de los LED's azules.
B	33-36	Pines 20-23 del primer integrado DM74154.	Proporcionan las señales de entrada para controlar el encendido secuencial de los LED's azules.
	37-40	Pines 20-23 del segundo integrado DM74154.	Proporcionan las señales de entrada para controlar el encendido secuencial de los LED's verdes.
C	25	Pin 2 del módulo inalámbrico HC-05.	Transmite los datos al modulo inalámbrico HC-05 que serán reenviados a un dispositivo bluetooth exterior.
	26	Pin 1 del módulo inalámbrico HC-05.	Recibe los datos transmitidos al módulo inalámbrico HC-05 que se envían desde un dispositivo bluetooth exterior.
D	19-22	Pines 20-23 del tercer integrado DM74154.	Proporcionan las señales de entrada para controlar el encendido secuencial de los LED's rojos.

Tabla. 41 Distribución de pines puertos microcontrolador

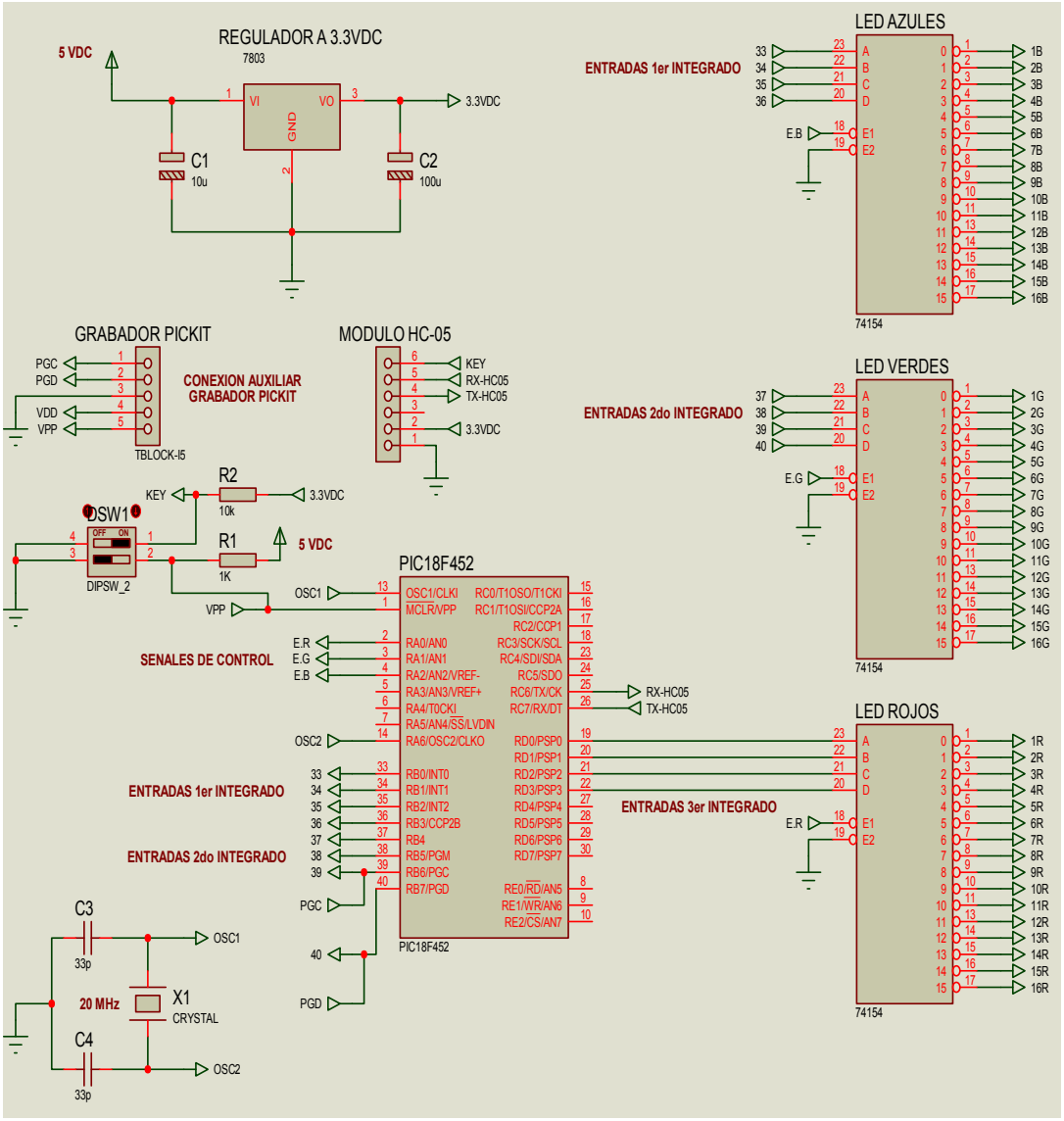


Figura. 69 Conexiones puertos del microcontrolador

Se designa un rango de valores de 0 al 7 para el manejo de los colores, por medio de las señales de control que se envían por el puerto A. (Ver tabla 42)

Valores en el Puerto A	Colores	Tono
0	Blanco	
1	Cyan	
2	Púrpura	
3	Azul	
4	Amarrillo	
5	Verde	
6	Rojo	
7	Led apagados	-

Tabla. 42 Valores en el puerto A

3.3.3 Elaboración circuito electrónico

Se traslada el diseño del circuito al ARES donde se crean las pistas y bornes para colocar los diferentes elementos físicamente. Se toma por tanto en cuenta las dimensiones de los dispositivos, alimentación y conexiones auxiliares para la grabación del PIC18F452 y configuración del modulo inalámbrico HC-05.

3.3.3.1 Dimensiones LED RGB SMD

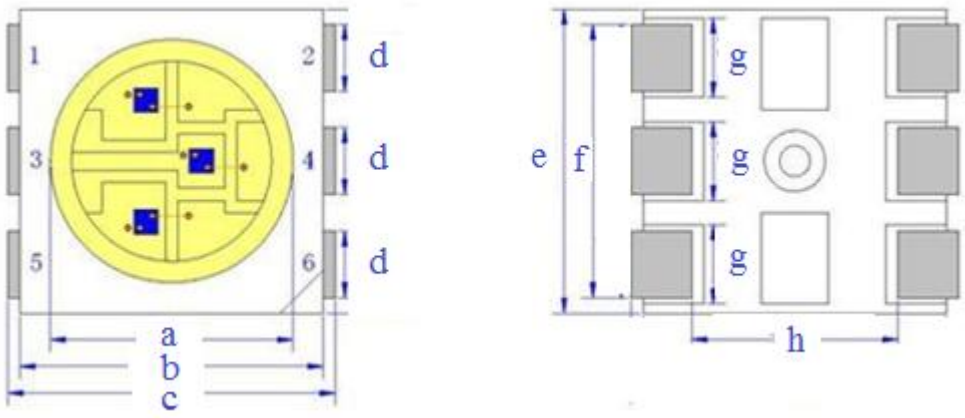


Figura. 70 Dimensiones LED RGB SMD[53]

53 http://spanish.alibaba.com/product-gs/plcc6-5050-series-3chips-white-surface-high-brightnesss-white-type-smd-led-308348283.html

Ítem	Dimensiones(mm)
a	4
b	5
c	5.40
d	1.10
e	5
f	4.50
g	1.30
h	3.40

Tabla. 43 Dimensiones LED RGB SMD

3.3.3.2 Dimensiones Transistores 2n3906

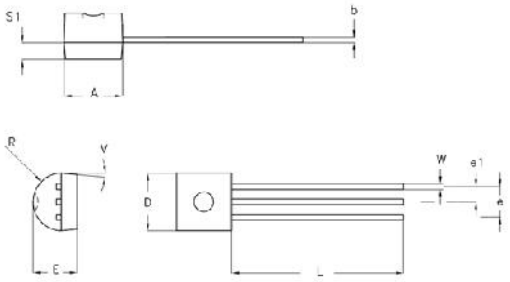


Figura. 71 Dimensiones transistores 2N3906[54]

Ítem	Min(mm)	Max(mm)
A	4.32	4.95
b	0.36	0.51
D	4.45	4.95
E	3.30	3.94
e	2.41	2.67
e1	1.14	1.40
L	12.70	15.49
R	2.16	2.41
S1	1.14	1.52
W	0.41	0.56
V	4 grados	6 grados

Tabla. 44 Dimensiones transistor 2N3906

54 http://www.ece.rice.edu/~jdw/data_sheets/2N3906.pdf, Figura 71 Dimensiones transistores 2N3906

3.3.3.3 Dimensiones dip switch

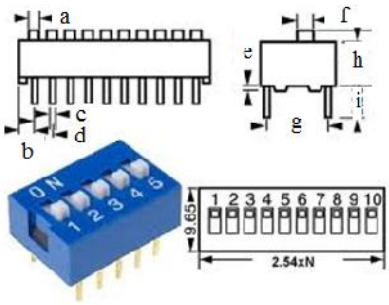


Figura. 72 Dimensiones dip switch[55]

Ítem	Dimensiones(mm)
a	1.52
b	2.54
c	0.51
d	2.29
e	0.51
f	1.78
g	7.62
h	6.22
i	3.96

Tabla. 45 Dimensiones dip switch

3.3.3.4 Dimensiones Módulo Inalámbrico HC-05

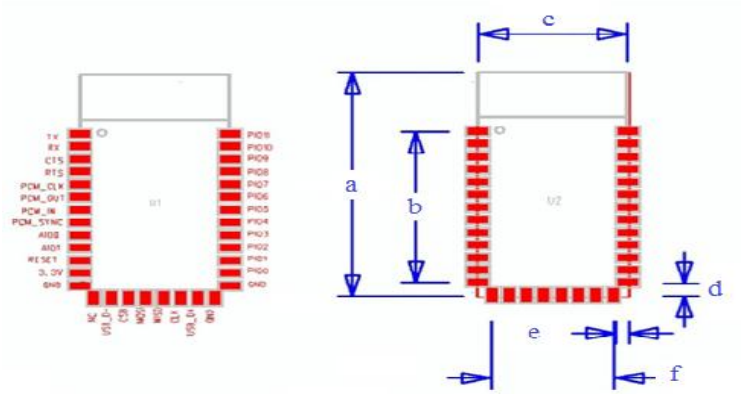


Figura. 73 Dimensiones módulo inalámbrico HC05[56]

55 http://www.shoptronica.com/576-switch-mini-dip-circuito-impreso.html, Figura 72 Dimensiones dip switch

56 ftp://imall.iteadstudio.com/IM120417010_BT_Shield_v2.2/DS_BluetoothHC05.pdf, Figura 73 Dimensiones modulo inalámbrico HC05

Ítem	Dimensiones(mm)
a	27
b	18
c	12.7
d	1.5
e	1.1
f	10.5

Tabla. 46 Dimensiones módulo inalámbrico HC05

Finalmente con todas las conexiones y dimensiones de los diferentes elementos, los resultados son tres placas:

1. Placa principal: Contiene al PIC18F452, tres integrados DM74154, módulo inalámbrico HC05, conexión auxiliar para grabación del PIC18F452 y borneras para señales de salida.

2. Placa transistores: Contiene 48 transistores que se configuran en modo interruptor y borneras para señales de entrada y salida.

3. Placa LED's RGB: Contiene 16 LED's RGB SMD.

Con el diseño en ARES listo se transfiere el circuito a una placa de cobre. (Ver figura 74)

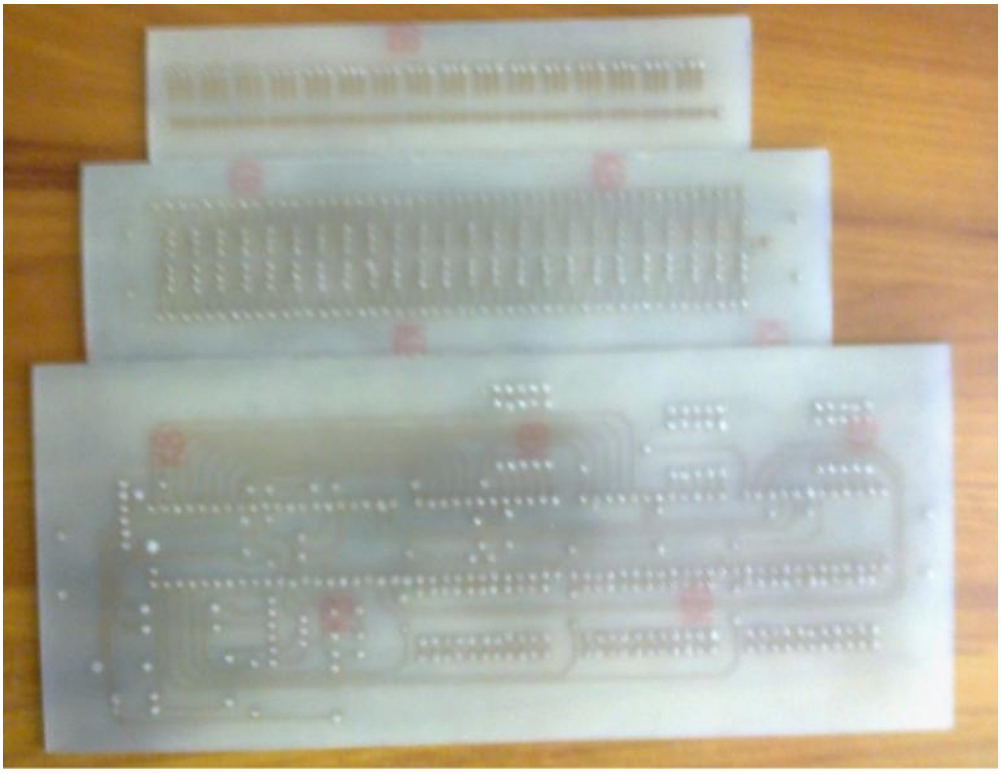


Figura. 74 Placas

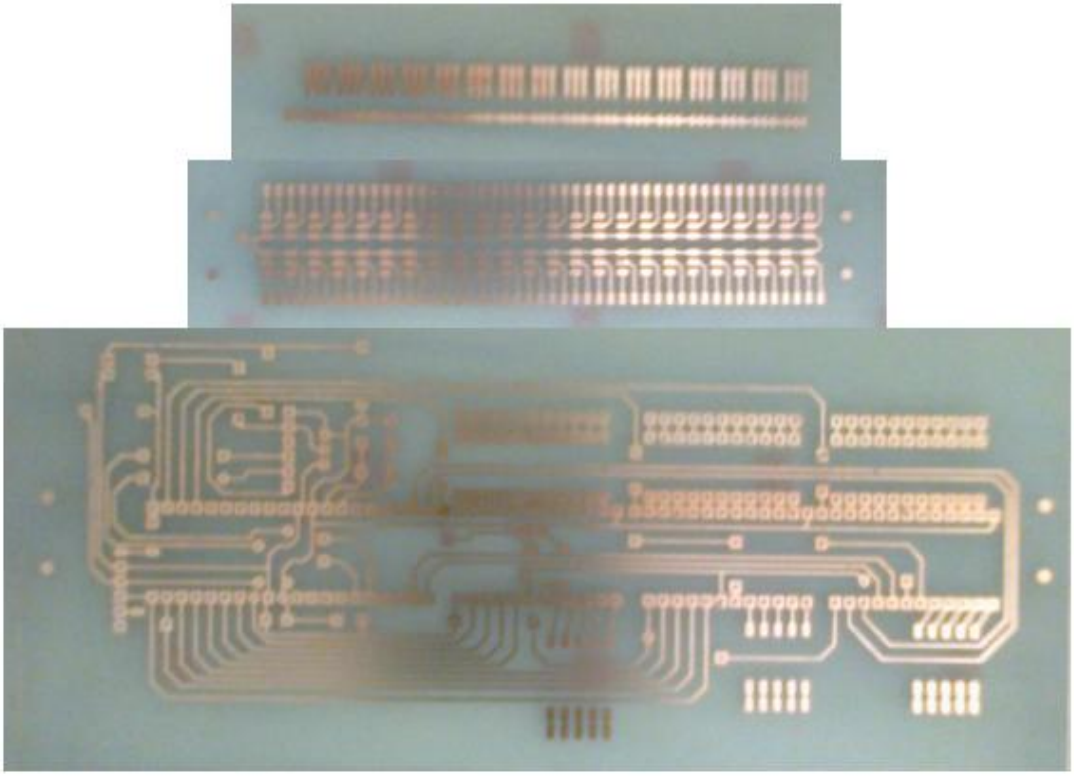


Figura. 75 Placas

3.3.4 Control de velocidad del motor

Permite variar la velocidad de giro del motor y consigue cambiar la posición del mensaje que se presenta en el sistema de rotulación RGB.

Para el diseño del control de velocidad se escoge un control a través de un dimmer que utiliza un método de regulación por triac y diac. El triac se utiliza para el control de fase de la corriente alterna que conmuta en los estados de conducción durante los semiciclos positivos y negativos de la señal de entrada, mientras que el diac se emplea para disparar al triac. La excitación del triac se realiza mediante un circuito R-C, que introduce un desfase por la constante de tiempo de carga del capacitor, que se controla por los valores de la resistencia R1, el potenciómetro POT y el capacitor C1. (Ver figura 76)

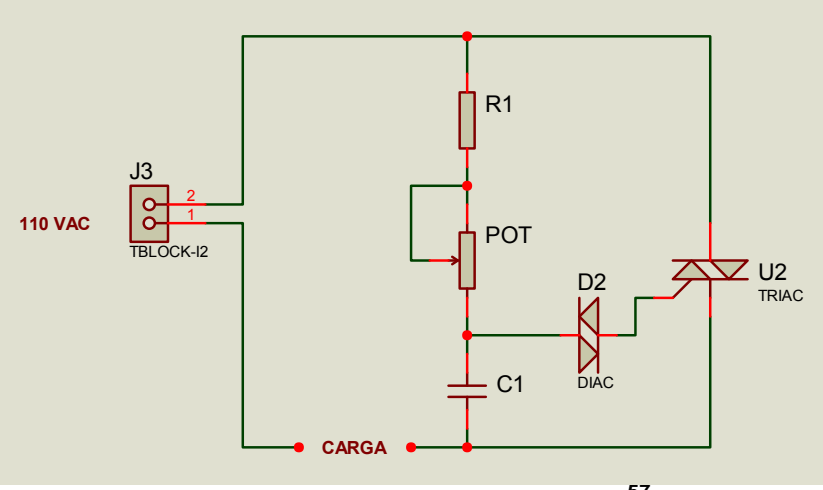


Figura. 76 Triac y circuito R-C[57]

La conexión que se muestra en la figura 3.31, no permite aplicar la misma tensión a la carga en todos los semiciclos de la señal de salida, esto, por la descarga parcial del

57 http://electronicavm.files.wordpress.com/2011/05/diacs-y-triacs.pdf, Figura 76 Triac y circuito R-C

capacitor C1, que provoca un efecto de histéresis en la señal de salida. Para resolver el efecto de histéresis se añade otro circuito R-C, que ayuda a que la tensión de excitación del triac no caiga abruptamente a causa de la recarga parcial del segundo capacitor C2. (Ver figura 77)

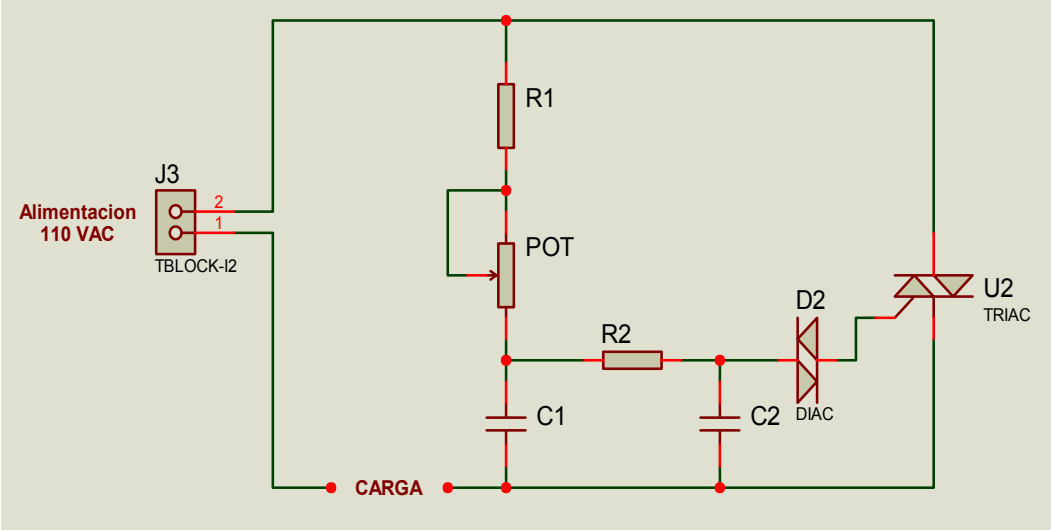


Figura. 77 Control de histéresis[58]

Finalmente se coloca un circuito amortiguador para evitar activaciones que no se deseen del triac, por tensiones parásitas. (Ver figura 78)

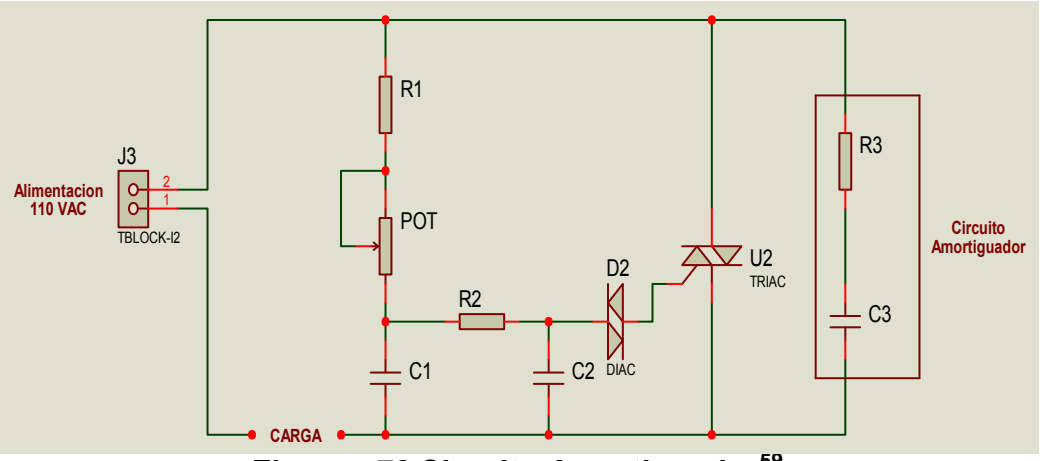


Figura. 78 Circuito Amortiguador[59]

58 http://electronicavm.files.wordpress.com/2011/05/diacs-y-triacs.pdf

3.3.4.1 Diseño del control velocidad

1. Se determina el mínimo ángulo de disparo del triac.

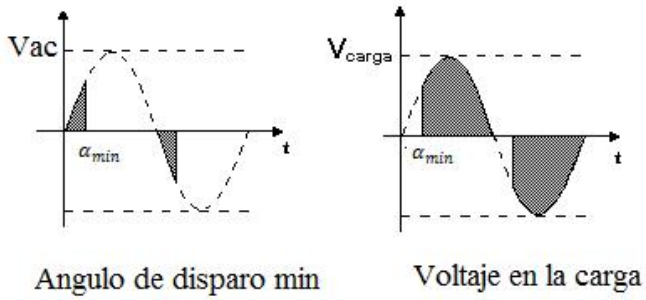


Figura. 79 Ángulo de disparo mínimo del triac

$$E_{AC} = E_P \sin wt$$

$$E_P = \sqrt{2} \cdot (V_{ACVrms})$$

$$E_P = \sqrt{2} * (120) = 169.7\ [Vpico]$$

$$wt = \sin^{-1}\left(\frac{E_{AC}}{E_P}\right)$$

$$wt = \sin^{-1}\left(\frac{32}{169.7}\right) = 10.86°$$

$$\alpha_{min} = 10.86°$$

Variable	Descripción	Unidades
	Voltaje de alimentación Diac	[V]
	Voltaje pico	[V]
	Mínimo ángulo de disparo	grados

Tabla. 47 Cálculo ángulo de disparo mínimo[60]

2. Se calcula el tiempo que corresponde al mínimo ángulo de disparo del triac.

$$t = \frac{1}{f} = \frac{1}{60[Hz]} = 16.67[ms]$$

59 http://electronicavm.files.wordpress.com/2011/05/diacs-y-triacs.pdf

60 http://vargasdaniel27.wordpress.com/2008/08/15/disparo-del-triac-mediante-diac/

$$t_{\alpha_{min}} = \frac{\alpha_{min} \cdot t}{360°} = \frac{10.86° \cdot 16.67[ms]}{360°} = 0.5[ms]$$

Variable	Descripción	Unidades
t	Tiempo de duración de la onda en AC	[s]
	Frecuencia	[Hz]
	tiempo mínimo del ángulo de disparo	[s]

Tabla. 48 Tiempo mínimo[61]

3. Se asume el valor del capacitor C1=0.1[uf]. Con el valor α_{min} , se calcula el valor de la resistencia R1. (Ver figura 78)

$$\phi = \tan^{-1}(w * C_1 * R_1) \quad [62]$$

$$\phi = \tan^{-1}(2 * \pi * f * C_1 * R_1)$$

$$10.86° = \tan^{-1}(2 \cdot \pi * 60 \cdot 0.1x10^{-6} \cdot R_1)$$

$$R_1 = 4890\Omega \approx 4.7k\Omega$$

4. Se determina el máximo ángulo de disparo del triac.

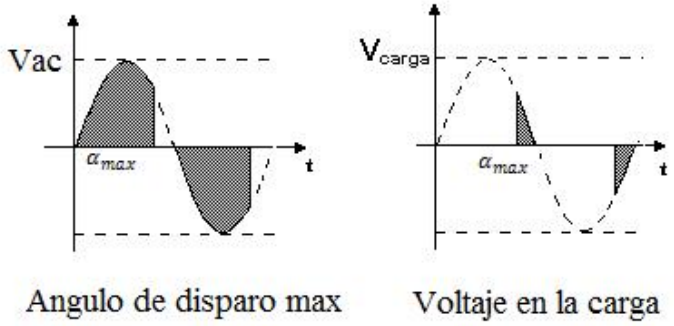


Figura. 80 Ángulo de disparo máximo

$$\alpha_{max} = 180° - \alpha_{min}$$

61 http://vargasdaniel27.wordpress.com/2008/08/15/disparo-del-triac-mediante-diac/

62 http://prof.usb.ve/bueno/E_Potencia/Laboratorio/Laboratorio.pdf

$$\alpha_{max} = 180° - 10.86° = 169.14°$$

5. Se calcula el tiempo que corresponde al máximo ángulo de disparo del triac.

$$t_{\alpha_{max}} = \frac{\alpha_{max} * t}{360°} = \frac{169.14° \cdot 16.67[ms]}{360°} = 7.83[ms]$$

6. Se calcula el valor del potenciómetro POT. (Ver figura 78)

$$X_c = \left(\frac{E_{AC}}{E_P}\right) * R^{63}$$

$$R = R_1 + POT$$

$$POT = \left(\frac{169.7}{32}\right) * \left(\frac{1}{2 * \pi * 60 \cdot 0.1x10^{-6}}\right) - 4.7k\Omega$$

$$POT = 135.96k\Omega \approx 150k\Omega$$

Finalmente los valores del capacitor C2 y la resistencia R2, se asumen con los valores de C1 y R1 respectivamente.

3.3.4.2 Posicionamiento de mensajes

Para el posicionamiento de los mensajes alrededor del sistema de rotulación RGB, se toma en cuenta las siguientes variables: la longitud de la hélice, la velocidad de giro del motor y el tiempo en que se presentan los mensajes. Para esto se sigue el siguiente procedimiento:

1. Se calcula el perímetro que abarca la hélice en su giro.

$$P_H = 2 \cdot \pi * (L_H/2)$$

63 http://prof.usb.ve/bueno/E_Potencia/Laboratorio/Laboratorio.pdf

$$P_H = 2 \pi * (20/2)$$

$$P_H = 62.83\ [cm]$$

Variable	Descripción	Unidades
[illegible]	Perímetro de la hélice	[cm]
[illegible]	Longitud de la hélice	[cm]

Tabla. 49 Perímetro de la hélice[64]

2. Se procede a segmentar el perímetro para presentar el mensaje en distintas ubicaciones.

$$P_M = P_H/s$$

$$P_M = 62.83/3$$

$$P_M \approx 21[cm]$$

Variable	Descripción	Unidades
[illegible]	Presentación mensajes	[cm]
[illegible]	Perímetro de la hélice	[cm]
s	segmentos	-

Tabla. 50 Posición del mensaje[65]

3. Se calcula el espacio que posee cada columna para presentar los mensajes.

$$E_C = P_M/\#_C$$

$$E_C = 21/30$$

$$E_C \approx 0.7\ [cm]$$

Variable	Descripción	Unidades
[illegible]	Espacio por columna	[cm]
[illegible]	Presentación mensajes	[cm]
[illegible]	Número de columnas	-

Tabla. 51 Espacio de columnas[66]

64 http://www.iesmajuelo.com/index.php?f=departamentos&id=354&deptno=Inform%E1tica,Tabla 49 Perímetro de la hélice

65 http://www.iesmajuelo.com/index.php?f=departamentos&id=354&deptno=Inform%E1tica

4. Se determina una velocidad de giro que proporciona el tiempo en que se presenta una columna del mensaje.

$$T_{tg} = \frac{1[RPM] * 60[s]}{V[RPM]}$$

$$T_{tg} = \frac{1[RPM] * 60[s]}{1000[RPM]} = 60\ [ms]$$

$$T_{pc} = \frac{E_C * T_{tg}}{P_H}$$

$$T_{pc} = \frac{0.7[cm] * 60[ms]}{62.83\ [cm]} \approx 0.66[ms]$$

Variable	Descripción	Unidades
T_{tg}	Tiempo total de giro	[ms]
T_{pc}	Tiempo presentación por columna	[ms]
V	Velocidad de giro del motor	[RPM]

Tabla. 52 Tiempo de presentación por columna[67]

3.3.5 Fuentes de alimentación

Se considera una conexión en paralelo de todos los dispositivos que conforman el sistema de rotulación RGB, para la alimentación del circuito electrónico. Las tensiones y cargas que se emplean para alimentar de energía al sistema de rotulación RGB se describen a continuación:

66 http://www.iesmajuelo.com/index.php?f=departamentos&id=354&deptno=Inform%E1tica

67 http://www.iesmajuelo.com/index.php?f=departamentos&id=354&deptno=Inform%E1tica

Elemento	Tensión	Corriente	Carga total
Motor	110 [VAC]	1 A	1 [A]
Microcontrolador PIC18F452	5 [VDC]	25 [mA]	25[mA]
Integrados DM74154	5 [VDC]	En nivel alto -0.8 [mA] En nivel bajo 16 [mA]	42[mA]
Módulos inalámbricos HC05	3.3 [VDC]	50 [mA]	50[mA]
Led RGB	Rojo 2.1 [VDC] Verde 3.3 [VDC] Azul 3.3[VDC]	20 [mA] 20 [mA] 20 [mA]	960[mA]

Tabla. 53 Características para la fuente de alimentación

Según los datos descritos, es necesario emplear una fuente de alimentación de 5 [VDC], que soporte un consumo de corriente alrededor de 1.5 [A] para los dispositivos electrónicos, y para el motor se emplea alimentación de 110 [VAC] 60Hz.

3.4 ELABORACIÓN DEL SISTEMA EN SOLIDWORKS VERSIÓN 2011

Solidworks versión 2011, es un programa que se desarrolló para la creación de diseños mecánicos en tercera dimensión, que emplea un ambiente de desarrollo gráfico, amigable con el usuario, intuitivo y fácil de manejar.

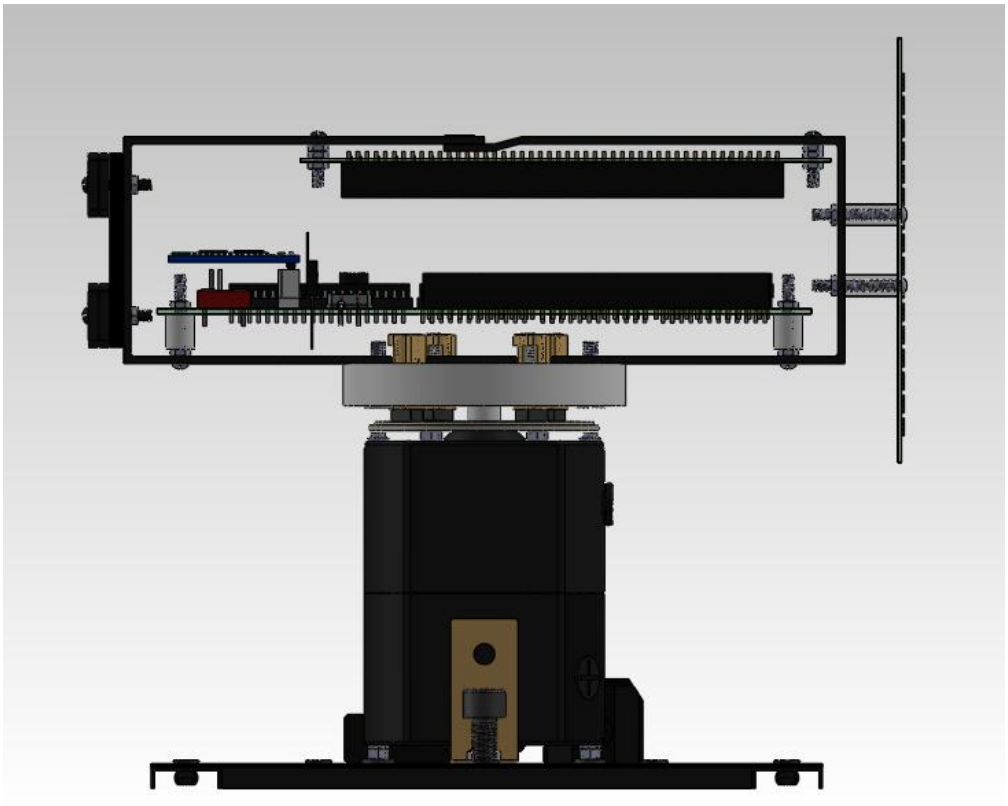


Figura. 81 Sistema de rotulación RGB vista lateral

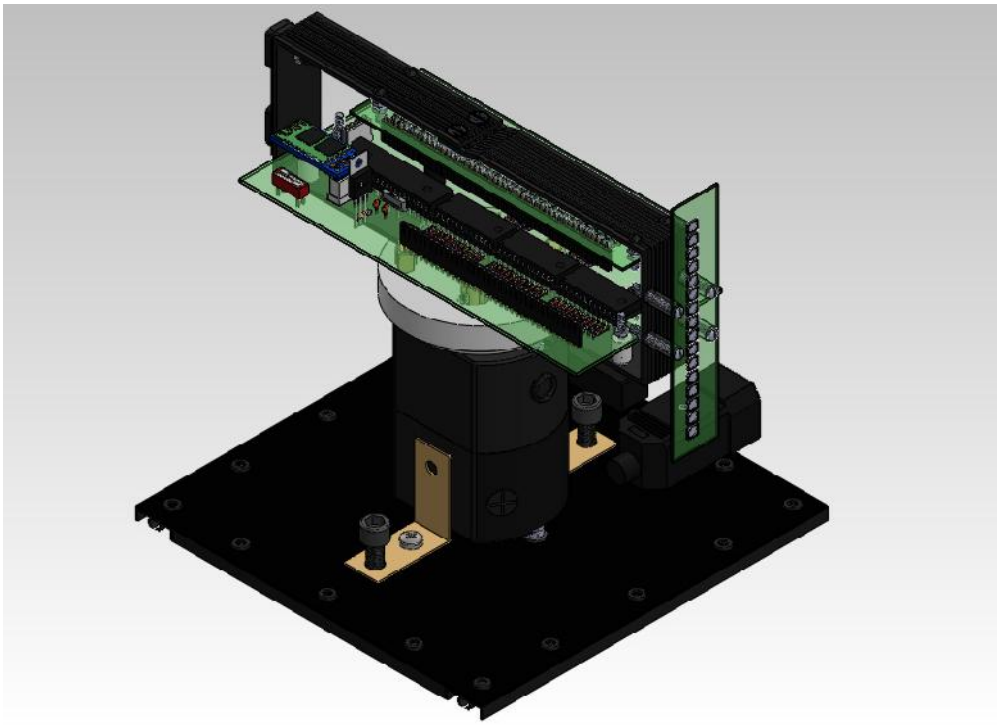


Figura. 82 Sistema de rotulación RGB vista superior

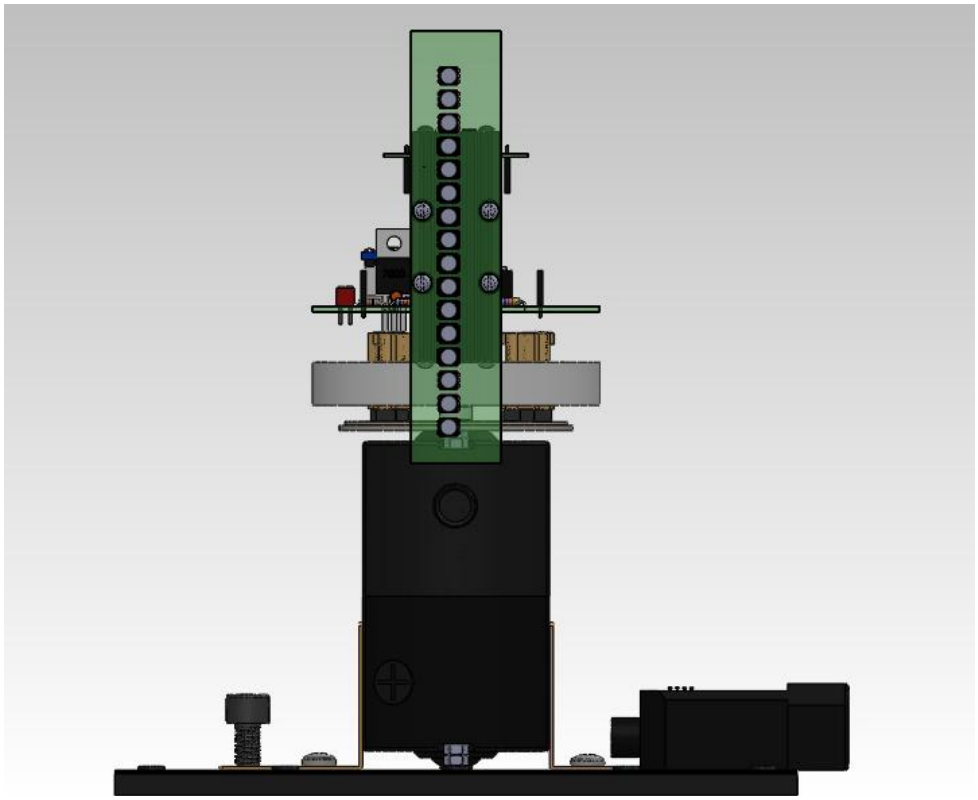


Figura. 83 Sistema de rotulación RGB vista frontal

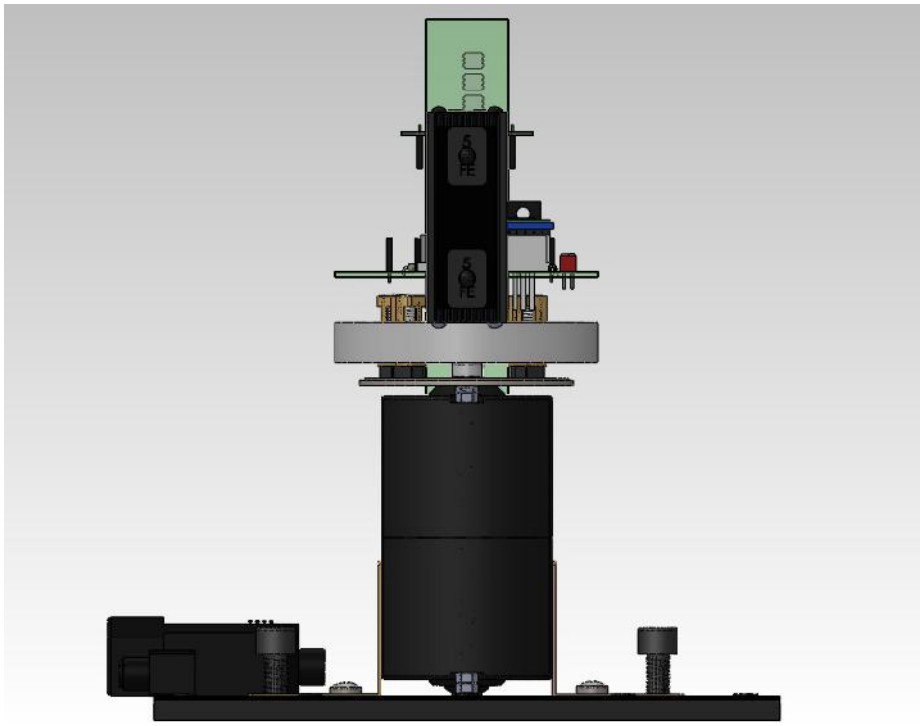


Figura. 84 Sistema de rotulación RGB vista posterior

CAPÍTULO 4

DISEÑO DEL SOFTWARE

4.1 DESARROLLO DE LA INTERFAZ Y PROGRAMACIÓN

4.1.1 Requerimientos previos

Para realizar la programación de la interfaz gráfica del dispositivo de rotulación RGB, es necesario instalar el IDE NetBeans 7.3.1 y la librería RXTX, que permite la comunicación serial entre la PC y el dispositivo de rotulación RGB.

- **Instalación de la librería RXTX**

Para realizar la comunicación serial se instala la librería RXTX.

1. Se descarga del link http://rxtx.qbang.org/wiki/index.php/Download.
2. Se abre la carpeta que se descarga y se copia el archivo RXTXcomm.jar.
3. Se pega el archivo que se copia en la dirección
   C:\ProgramFiles\Java\jdk1.7.0_25\jre\lib\ext

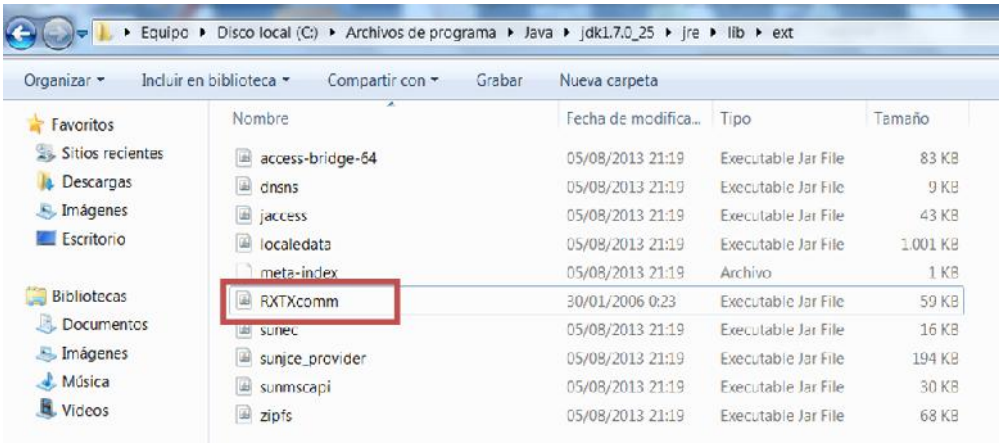


Figura. 85 Archivo RXTXcomm.jar

4. Se regresa a la carpeta que se descarga y se copia los archivos rxtxParallel.dll y rxtxSerial.dll.
5. Se pega el archivo que se copia en la dirección C:\Program Files\Java\jdk1.7.0_25\jre\bin.

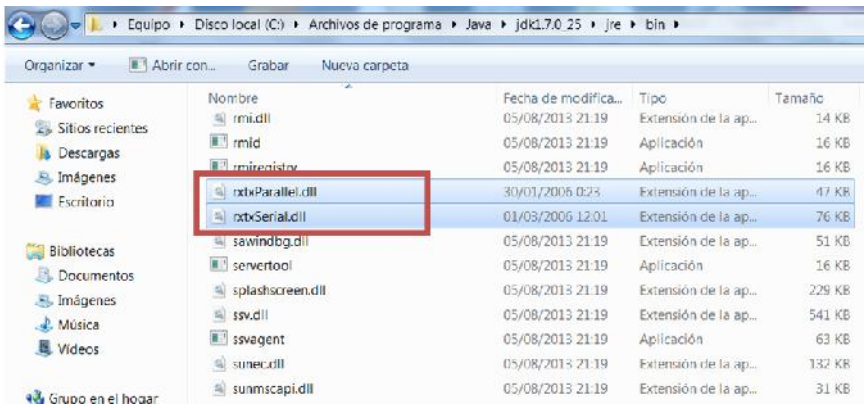


Figura. 86 Archivos rxtxParallel.dll y rxtxSerial.dll.

6. Cuando los archivos se copien en las direcciones mencionadas se hace uso de la librería en el IDE NetBeans 7.3.1.

4.1.2 Funcionalidad

La interfaz que se desarrolla para el sistema de rotulación RGB, permite diseñar los mensajes que se presentan en el dispositivo, almacenar el código de los mensajes en un bloc de notas y abrir códigos que se guardan con anterioridad.

4.1.3 Programación

4.1.3.1 Declaración de clase principal JAVA

La clase principal de JAVA establece el JFrame, que se ejecuta primero en el programa.

Objeto	Clase	Descripción
presentacion	Presentacion	Permite utilizar los métodos y funciones del JFrame Presentacion.
Programación		
presentacion.setVisible(true);		
Permite visualizar el JFrame Presentacion que se ejecuta en la interfaz del sistema de rotulación RGB.		

Tabla. 54 Declaración de clase principal JAVA

4.1.3.2 Declaración de JFrame JAVA

4.1.3.2.1 JFrame Presentación

El JFrame Presentación es la carátula de la interfaz grafica del dispositivo de rotulación RGB.

Figura. 87 JFrame Presentación

Ítem	Nombre del Componente
1	LblTitulo
2	LblLogo
3	LblTema
4	LblAutores
5	LblAutor1
6	LblAutor2
7	BtnIngresar

Tabla. 55 Descripción JFrame Presentación

Para realizar la programación del botón ingresar se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnIngresar.

Programación
Presentacion.this.dispose();
Destruye la ventana Presentación.
new Principal().setVisible(true);
Hace visible el JFrame Principal del programa.

Tabla. 56 Programación del botón ingresar

4.1.3.2.2 JFrame Principal

El JFrame Principal permite crear un mensaje nuevo o abrir un mensaje que se crea con anterioridad.

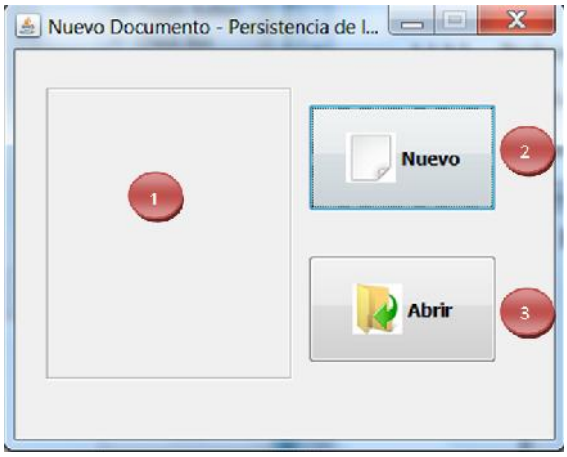


Figura. 88 JFrame Principal

Ítem	Nombre del Componente
1	LblImagen
2	BtnNuevo
3	BtnAbrir

Tabla. 57 Descripción JFrame Principal

1. Programación del Botón Nuevo

Para realizar la programación del botón **Nuevo** se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnNuevo.

Programación
Principal.this.dispose();
Destruye la ventana Principal.
new Nuevo().setVisible(true);
Hace visible el JFrame Nuevo del programa.

Tabla. 58 Programación botón nuevo

2. Programación del Botón Abrir:

Para realizar la programación del botón **Abrir** se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnAbrir.

Programación
Principal.this.dispose();
Destruye la ventana Principal.
new Abrir().setVisible(true);
Hace visible el JFrame Abrir del programa.

Tabla. 59 Programación botón abrir

4.1.3.2.3 JFrame Nuevo

La función principal del JFrame Nuevo es crear los mensajes que presenta el dispositivo de rotulación RGB.

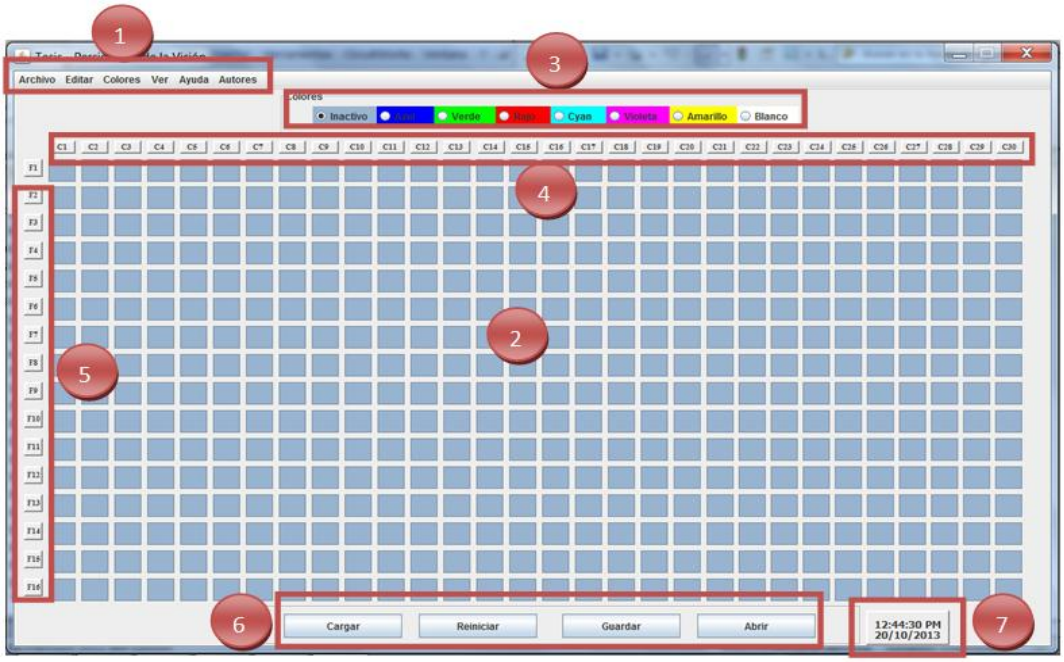


Figura. 89 JFrame Nuevo

Ítem	Nombre del Componente
1	MBarra
2	FNCN
3	PnlColores
4	LblFN
5	LblCN
6	PnlBotoneras
7	PnlHora

Tabla. 60 Descripción JFrame Nuevo

Para programar este JFrame se realiza los siguientes pasos:

1. **Importación de librerías**

Para iniciar con la programación del JFarme se importan las librerías que se utilizan en el programa. (Ver tabla 61).

Librería	Descripción
import java.awt.*	Permite utilizar los distintos métodos de clase Color.
import javax.swing.JOptionPane	Permite utilizar ventanas emergentes que muestran avisos temporales.
import java.util.Calendar; **import java.util.Date;** **import java.util.GregorianCalendar;**	Permite utilizar los métodos que capturan la fecha y hora del sistema.

Tabla. 61 Librerias del JFrame Nuevo

2. **Declaración de variables, objetos e hilos**

Variable	Tipo	Descripción
hora	String	Guarda la hora que se obtiene del sistema.
minutos	String	Guarda los minutos que se obtienen del sistema.
segundos	String	Guarda los segundos que se obtienen del sistema.
ampm	String	Guarda el valor am o pm que se obtiene del sistema.
dia	String	Guarda el día que se obtiene del sistema.
mes	String	Guarda el mes que se obtiene del sistema.
anio	String	Guarda el año que se obtiene del sistema.
matriz	String	Es un arreglo bidimensional que guarda los valores que

	[][]	se asignan a la matriz principal del JFrame Nuevo.
codigoguardar	String	Guarda el código de la matriz principal para exportar al JFrame Guardar.
codigover	String	Guarda el código de la matriz principal para exportar al JFrame Ver.
Codigocargar	String	Guarda el código de la matriz principal para exportar al JFrame Cargando.

Tabla. 62 Declaración de variables del JFrame Nuevo

Objeto	Tipo	Descripción
calendario	Calendar	Permite utilizar los métodos para capturar la fecha y hora del sistema.
pantallaver	Ver	Permite utilizar los métodos y datos del JFrame Ver.
pantallacargar	Cargando	Permite utilizar los métodos y datos del JFrame Cargando.
pantallaguardar	Guardar	Permite utilizar los métodos y datos del JFrame Guardar.
pantallaabrir	Abrir	Permite utilizar los métodos y datos del JFrame Abrir.
com	ComunicacionSerial	Permite utilizar los métodos y datos de la clase ComunicacionSerial.
INACTIVO	Color	Establece el color para la opción **desactivada** de la matriz principal.

Tabla. 63 Declaración de objetos del JFrame Nuevo

Hilo	Tipo	Descripción
hilo1	Thread	El hilo actualiza la hora y fecha del sistema en el JFrame Nuevo.

Tabla. 64 Declaración de hilos del JFrame Nuevo

3. Declaración del constructor

En todos los JFrame se crea un constructor que inicializa los componentes que se crean. En el JFrame **Nuevo** se añade la creación e inicialización del hilo que se encarga de actualizar la fecha y hora.

Constructor	Descripción
Nuevo	Inicializa los elementos que se crean en el JFrame Nuevo.
Programación	
hilo1 = new Thread(this);	
Crea el hilo que actualiza la fecha y hora.	
hilo1.start();	
Inicializa el hilo.	

Tabla. 65 Constructor del JFrame Nuevo

4. Declaración de los métodos

El JFrame Nuevo cuenta con dos métodos que son: guardar y borrar. (Ver tabla 66 y tabla 67).

Método	Tipo	Descripción
guardar	void	Este método permite guardar el código de la matriz principal en una variable que se exporta al JFrame Guardar, además muestra el JFrame Guardar.
Programación		
pantallaguardar.setVisible(true);		
Hace visible el JFrame Guardar.		

for (int filas = 0; filas <= 15; filas++){
for(int columnas = 0; columnas <= 18; columnas++){
if (columnas < 18){
codigoguardar = codigoguardar + matriz[filas][columnas];
}else if ((columnas == 18) & (filas < 15)){
codigoguardar = codigoguardar + matriz[filas][columnas] + "\n";
}else if ((columnas == 18) & (filas == 15)){
codigoguardar = codigoguardar + matriz[filas][columnas];
}
}
}		

El for se encarga de guardar los datos de la matriz principal en la variable que se exporta al JFrame Guardar.		
pantallaguardar.TxtCodigo.setText(codigoguardar);		
Envía los datos que se guardan en la variable de exportación a un área de texto del JFrame Guardar.		
codigoguardar = "";		
Borra los datos que se guardan en la variable de exportación.		

Tabla. 66 Declaración del método guardar del JFrame Nuevo

Método	Tipo	Descripción
borrar	void	Este método permite borrar el diseño que se crea en la matriz principal del JFrame Nuevo.
Programación		

for (int columnas = 0; columnas <= 18; columnas++){
for(int filas = 0; filas <= 15; filas++){
matriz[filas][columnas] = "7"; }
}		

El for se encarga de inicializar la matriz principal.		
F0C0.setBackground(INACTIVO); ……. ……. ……. F15C18.setBackground(INACTIVO);		
Cada posición de la matriz principal de diseño del JFrame Nuevo se identifica desde Fila 0 – Columna 0 (F0C0) hasta Fila 15 – Columna 18 (F15C18).		
Cuando se llama al método borrar cada posición de la matriz principal de diseño se inicializa con el color de inactivo.		

Tabla. 67 Declaración del método borrar del JFrame Nuevo

5. Programación del MBarra

En el menú MBarra se encuentran sub-menús como: Archivo, Editar, Ver, Colores, Ayuda y Autores.

- **Programación del sub-menú Archivo - nuevo**

Para realizar la programación del menú Archivo-nuevo se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el menú Archivo - nuevo.

Programación

if(JOptionPane.showConfirmDialog (){
this.dispose();
borrar();
this.show();}

El if identifica si se acepta la confirmación de crear un archivo y llama al método borrar() para inicializar la matriz principal y la interfaz principal de diseño.

Tabla. 68 Programación del sub-menú Archivo-nuevo

- **Programación del sub-menú Archivo - abrir:**

Para realizar la programación del sub-menú Archivo-abrir se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el menú Archivo - abrir.

Programación

if(JOptionPane.showConfirmDialog (){
pantallaabrir.setVisible(true);
Nuevo.this.dispose();}

El if identifica si se acepta la confirmación de abrir un archivo y permite visualizar el JFrame Abrir entonces se destruye el JFrame Nuevo.

Tabla. 69 Programación del sub-menú Archivo-abrir

- **Programación del sub-menú Archivo - guardar**

Para realizar la programación del sub-menú Archivo-guardar se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el menú archivo - guardar.

Programación
guardar();
Realiza la llamada al método guardar.

Tabla. 70 Programación del sub-menú Archivo-guardar

- **Programación del sub-menú Archivo - salir**

Para realizar la programación del sub-menú Archivo-salir se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el sub-menú Archivo - salir.

Programación
System.exit(0);
Permite salir de la aplicación.

Tabla. 71 Programación del sub-menú Archivo-salir

- **Programación del sub-menú Editar –cargar código**

Para realizar la programación del sub-menú Editar-cargar código se selecciona el evento ActionPerformed, que funciona al hacer clic sobre menú cargar código.

Programación
pantallacargar.setVisible(true);
Permite visualizar el JFrame Cargando.

for (int columnas = 0; columnas <= 18; columnas++){
for(int filas = 0; filas <= 15; filas++){
pantallacargar.cargarcodigo[filas][columnas] = matriz[filas][columnas];
}
}

El for se encarga de guardar los datos de la matriz principal en una variable del JFrame Cargando.

Tabla. 72 Programación del sub-menú Editar –cargar código

- **Programación del sub -menú Editar –borrar código**

Para realizar la programación del sub-menú Editar-borrar código se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el sub-menú borrar código.

Programación
borrar();
Realiza la llamada al método borrar.

Tabla. 73 Programación del sub-menú Editar –borrar código

- **Programación del sub-menú Ver – diseño previo**

Para realizar la programación del sub-menú Ver-diseño previo se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el sub-menú diseño previo.

Programación
pantallaver.setVisible(true);
Hace visible el JFrame Ver.

for (int filas = 0; filas <= 15; filas++){
for(int columnas = 0; columnas <= 18; columnas++){
if (columnas < 18){
codigover = codigover + matriz[filas][columnas];
}else if ((columnas == 18) & (filas < 15)){
codigover = codigover + matriz[filas][columnas] + "\n";
}else if ((columnas == 18) & (filas == 15)){
codigover = codigover + matriz[filas][columnas];
}
}
}

El for se encarga de guardar los datos de la matriz principal en la variable que se exporta al JFrame Ver.
pantallaver.vercodigo = codigover;
Guarda los datos de la variable que se exporta en una variable del JFrame Ver.
codigover = "";
Borra los datos que se guardan en la variable de exportación.

Tabla. 74 Programación del sub-menú Ver –diseño previo

- **Programación del sub-menú Ver – configuración puerto**

Para realizar la programación del sub-menú Ver-configuración puerto se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el sub-menú configuración puerto.

Programación
new ConfiguracionPuerto().setVisible(true);
Permite visualizar el JFrame ConfiguracionPuerto.

Tabla. 75 Programación del sub-menú Ver – configuración puerto

- **Programación del sub-menú Ayuda**

Para realizar la programación del sub-menú Ayuda se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el sub-menú Ayuda.

Programación
new Ayuda().setVisible(true);
Permite visualizar el JFrame Ayuda.

Tabla. 76 Programación del sub-menú Ayuda

- **Programación del sub-menú Autores**

Para realizar la programación del sub-menú Autores se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el sub-menú Autores.

Programación
new Autores().setVisible(true);
Permite visualizar el JFrame Autores.

Tabla. 77 Programación del sub-menú Autores

6. Programación del PnlColores

La programación del PnlColores escoge el color con el que se pinta la posición de la matriz principal de diseño. Este panel consta con ocho radio botones que se identifican como RbtnVerde, RbtnRojo, RbtnCyan, RbtnVioleta, RbtnAmarillo, RbtnBlanco, RbtnAzul y RbtnInactivo.

7. Programación de la matriz de diseño

Para realizar la programación de la matriz principal de diseño se toma en cuenta que cada posición es un botón que se identifica desde la Fila 0 – Columna 0 (F0C0) hasta Fila 15 – Columna 29 (F15C29). Cada botón posee el evento ActionPerformed que funciona al hacer clic sobre el botón que se elije. Todos los botones de la matriz principal de diseño poseen la misma programación.

Programación

if(RbtnInactivo.isSelected()){
F0C0.setBackground(INACTIVO);
matriz[0][0]="7";
}else if (RbtnAzul.isSelected()) {
F0C0.setBackground(Color.BLUE);
matriz[0][0]="3";
}else if(RbtnVerde.isSelected()){
F0C0.setBackground(Color.GREEN);
matriz[0][0]="5";
}else if(RbtnRojo.isSelected()){
F0C0.setBackground(Color.RED);
matriz[0][0]="6";
}else if(RbtnCyan.isSelected()){
F0C0.setBackground(Color.CYAN);
matriz[0][0]="1";
}else if(RbtnVioleta.isSelected()){
F0C0.setBackground(Color.MAGENTA);
matriz[0][0]="2";
}else if(RbtnAmarillo.isSelected()){
F0C0.setBackground(Color.YELLOW);
matriz[0][0]="4";
}else if(RbtnBlanco.isSelected()){
F0C0.setBackground(Color.WHITE);
matriz[0][0]="0";
}

El if identifica el color que se selecciona en el PnlColores, asigna el color a la posición que se escoge en la matriz principal de diseño y también sitúa el código numérico a la matriz principal.

Tabla. 78 Programación de la matriz de diseño

8. Programación de los LblFN y LblCN

Para realizar la programación de los componentes LblFN y LblCN se toma en cuenta que cada fila tiene un LblFN, que se identifica desde la Fila 1 (LblF1) hasta Fila 16 (LblF16) y cada columna tiene un LblCN, que se identifica desde la Columna 1 (LblC1) hasta Columna 16 (LblC30). Cada componente posee el evento MouseClicked que funciona al hacer clic sobre el componente que se elije. Todos los componentes LblFN y LblCN poseen la misma programación.

Programación para LblFN

if(RbtnInactivo.isSelected()){
for (int columnas = 0; columnas <= 29; columnas++){
matriz[0][columnas] = "7";
}
F0C0.setBackground(INACTIVO);
F0C18.setBackground(INACTIVO);
}else if (RbtnAzul.isSelected()) {
for (int columnas = 0; columnas <= 29; columnas++){
matriz[0][columnas] = "3";}
F0C0.setBackground(Color.BLUE);
F0C18.setBackground(Color.BLUE);
}else if(RbtnVerde.isSelected()){
for (int columnas = 0; columnas <= 29; columnas++){
matriz[0][columnas] = "5";}
F0C0.setBackground(Color.GREEN);
F0C18.setBackground(Color.GREEN);
}else if(RbtnRojo.isSelected()){
for (int columnas = 0; columnas <= 29; columnas++){
matriz[0][columnas] = "6";}
F0C0.setBackground(Color.RED);
F0C18.setBackground(Color.RED);
}else if(RbtnCyan.isSelected()){
for (int columnas = 0; columnas <= 29; columnas++){
matriz[0][columnas] = "1";}
F0C0.setBackground(Color.CYAN);
F0C18.setBackground(Color.CYAN);
}else if(RbtnVioleta.isSelected()){
for (int columnas = 0; columnas <= 29; columnas++){
matriz[0][columnas] = "2";}
F0C0.setBackground(Color.MAGENTA);
F0C18.setBackground(Color.MAGENTA);
}else if(RbtnAmarillo.isSelected()){
for (int columnas = 0; columnas <= 29; columnas++){
matriz[0][columnas] = "4";}
F0C0.setBackground(Color.YELLOW);
F0C18.setBackground(Color.YELLOW);
}else if(RbtnBlanco.isSelected()){
for (int columnas = 0; columnas <= 29; columnas++){
matriz[0][columnas] = "0";}
F0C0.setBackground(Color.WHITE);
F0C18.setBackground(Color.WHITE); }

El if identifica el color que se selecciona en el PnlColores y se asigna el color a la fila que se selecciona de la matriz principal de diseño.
El for asigna el código numérico a la posición respectiva de la matriz principal.

Tabla. 79 Programación para LblFN

Programación para LblCN

```
if(RbtnInactivo.isSelected()){
          for (int filas = 0; filas <= 15; filas++){
                matriz[filas][0] = "7";}
             F0C0.setBackground(INACTIVO);
             F15C0.setBackground(INACTIVO);
          }else if (RbtnAzul.isSelected()) {
           for (int filas = 0; filas <= 15; filas++){
                matriz[filas][0] = "3";}
             F0C0.setBackground(Color.BLUE);
             F15C0.setBackground(Color.BLUE);
          }else if(RbtnVerde.isSelected()){
           for (int filas = 0; filas <= 15; filas++){
                matriz[filas][0] = "5";}
             F0C0.setBackground(Color.GREEN);
             F15C0.setBackground(Color.GREEN);
          }else if(RbtnRojo.isSelected()){
           for (int filas = 0; filas <= 15; filas++){
                matriz[filas][0] = "6";}
             F0C0.setBackground(Color.RED);
             F15C0.setBackground(Color.RED);
          }else if(RbtnCyan.isSelected()){
           for (int filas = 0; filas <= 15; filas++){
                matriz[filas][0] = "1";}
             F0C0.setBackground(Color.CYAN);
             F15C0.setBackground(Color.CYAN);
          }else if(RbtnVioleta.isSelected()){
           for (int filas = 0; filas <= 15; filas++){
                matriz[filas][0] = "2";}
             F0C0.setBackground(Color.MAGENTA);
             F15C0.setBackground(Color.MAGENTA);
          }else if(RbtnAmarillo.isSelected()){
           for (int filas = 0; filas <= 15; filas++){
                matriz[filas][0] = "4";}
             F0C0.setBackground(Color.YELLOW);
             F15C0.setBackground(Color.YELLOW);
          }else if(RbtnBlanco.isSelected()){
           for (int filas = 0; filas <= 15; filas++){
                matriz[filas][0] = "0";}
             F0C0.setBackground(Color.WHITE);
             F15C0.setBackground(Color.WHITE);}
```

El if identifica el color que se selecciona en el PnlColores y se asigna el color a la columna que se selecciona de la matriz principal de diseño.

El for asigna el código numérico a la posición respectiva de la matriz principal.

Tabla. 80 Programación para LblCN

9. Programación del PnlBotoneras

En el PnlBotoneras se distribuyen los botones: Cargar, Reiniciar, Guardar y Abrir.

- **Programación del botón Cargar:**

Para realizar la programación del botón Cargar se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnCargar.

Programación
pantallacargar.setVisible(true);
Permite visualizar el JFrame Cargando.

for (int columnas = 0; columnas <= 29; columnas++){
for(int filas = 0; filas <= 15; filas++){
pantallacargar.cargarcodigo[filas][columnas] = matriz[filas][columnas];
}
}

El for se encarga de guardar los datos de la matriz principal en una variable del JFrame Cargando.

Tabla. 81 Programación del botón Cargar

- **Programación del botón Reiniciar**

Para realizar la programación del botón Reiniciar se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnReiniciar.

Programación
if(JOptionPane.showConfirmDialog (){ borrar();}
El if identifica si el usuario acepta la confirmación de reinicio que se presenta en una ventana emergente, en el caso de aceptar realiza el llamado al método borrar() para inicializar la matriz principal y la matriz principal de diseño.

Tabla. 82 Programación del botón Reiniciar

- **Programación del botón Guardar**

Para realizar la programación del botón Guardar se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnGuardar.

Programación
guardar();
Realiza la llamada al método guardar.

Tabla. 83 Programación del botón Guardar

- **Programación del botón Abrir:**

Para realizar la programación del botón Abrir se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnAbrir.

Programación

if(JOptionPane.showConfirmDialog (){
pantallaabrir.setVisible(true);
Nuevo.this.dispose();}

El if identifica si el usuario acepta la confirmación de abrir que se presenta en una ventana emergente, en el caso de aceptar permite visualizar el JFrame Abrir y destruye el JFrame Nuevo.

Tabla. 84 Programación del botón Abrir

4.1.3.2.4 JFrame Guardar

El JFrame Guardar permite visualizar el código numérico del mensaje que se crea en la matriz principal de diseño, además que guarda este código en un archivo txt.

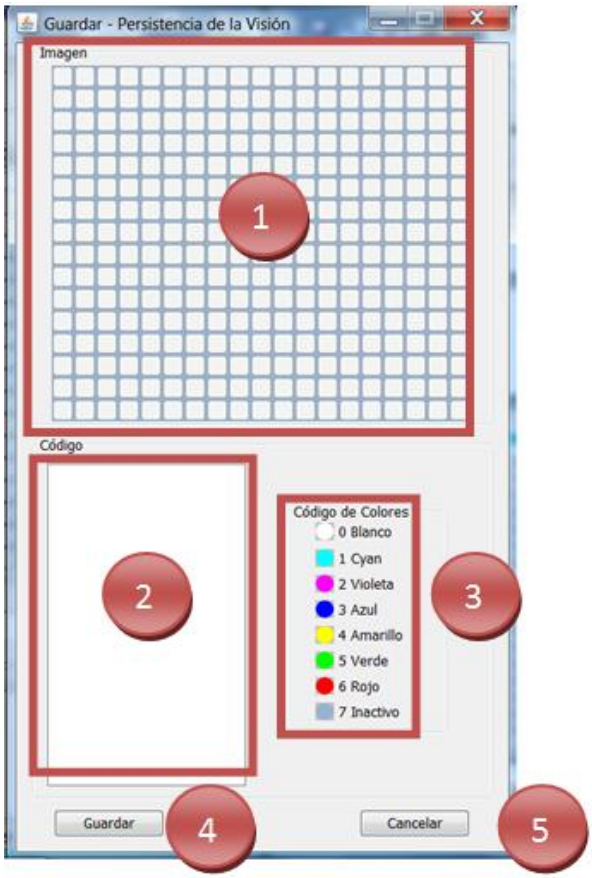


Figura. 90 JFrame Guardar

Ítem	Nombre del Componente
1	PnlImagen
2	PnlCodigo
3	PnlCodColores
4	BtnGuardar
5	BtnCancelar

Tabla. 85 Descripcion del JFrame Guardar

Para programar este JFrame se realiza los siguientes pasos:

1. Importación de librerías

Para iniciar con la programación del JFrame Guardar se importan las librerías que se utilizan en el programa de la interfaz del sistema de rotulación RGB. (Ver tabla 86)

Librería	Descripción
import java.awt.Color;	Permite utilizar los métodos de la clase Color.
import javax.swing.JFileChooser;	Permite seleccionar y/o grabar cualquier archivo dentro de la PC.
import javax.swing.JOptionPane;	Permite utilizar ventanas emergentes que muestran avisos temporales.

Tabla. 86 Librerias del JFrame Guardar

2. **Declaración de variables y objetos**

Para iniciar el desarrollo del JFrame Guardar se declara las variables y su función en el programa. (Ver tabla 87 y tabla 88)

Variable	Tipo	Descripción
codigo	String	Guarda el código numérico de la matriz principal.

Tabla. 87 Variables del JFrame Guardar

Objeto	Tipo	Descripción
editor	Editor	Permite utilizar los métodos y datos de la clase Editor.
INACTIVO	Color	Establece el color para la opción desactivada de la matriz principal

Tabla. 88 Objetos del JFrame Guardar

3. **Declaración del constructor**

En el JFrame Guardar se añade un constructor para crear un objeto de la clase Editor.

Constructor	Descripción
Guardar	Inicializa los elementos creados en el JFrame Guardar.
Programación	
editor = new Editor();	
Inicializa el objeto de la clase editor.	

Tabla. 89 Declaración del constructor del JFrame Guardar

4. Declaración de los métodos

El JFrame Guardar cuenta con métodos como: enviarcontenido, guardararchivo. Cada uno cumple funciones específicas. (Ver tabla. 90)

Método	Tipo	Descripción
enviarcontenido	String	Recoge el texto del área de texto.
Programación		
return TxtCodigo.getText();		
Retorna el texto que se captura del área de texto.		

Tabla. 90 Declaración del método enviarcontenido del JFrame Guardar

Método	Tipo	Descripción
guardararchivo	void	Guarda el texto en un archivo txt.
Programación		
If(editor.saberarchivo()){……..} else{……..}		
El if permite saber si el archivo de texto está lleno o vacío. Con el archivo vacío se realiza la siguiente programación:		
JFileChooser fc = new JFileChooser();		
Crea una ventana para buscar en los archivos de la computadora.		
if(fc.showSaveDialog(this) == JFileChooser.APPROVE_OPTION){……}		
El if permite saber si se escoge un lugar donde almacenar el archivo.		
String ruta = fc.getSelectedFile().getAbsolutePath(); String contenido = enviarcontenido();		
Cuando se escoge un lugar donde almacenar el archivo txt, se guarda la ruta y el contenido del archivo txt.		

try {
editor.guardartxt(contenido, ruta);
JOptionPane.showMessageDialog();
} catch (Exception ex) {
JOptionPane.showMessageDialog();}		

El try – catch encierra el método que permite guardar el texto en un archivo txt y también genera mensajes de información sobre si se guarda correctamente el archivo. Cuando se realiza la llamada del método de la clase Editor se envía los parámetros de ruta y contenido del archivo txt.		
Cuando el archivo este lleno, se realiza la siguiente programación:		
String contenido = enviarcontenido();		
Guardar el contenido del archivo txt.		

try {
editor.guardartxt(contenido, null);
JOptionPane.showMessageDialog();
} catch (Exception ex) {
JOptionPane.showMessageDialog();
}

El try – catch encierra el método que permite guardar el texto en un archivo txt y también genera mensajes de información sobre si se guarda correctamente el archivo. Cuando se realiza la llamada del método de la clase Editor, se envía como parámetro el contenido del archivo txt.

Tabla. 91 Declaración del método guardararchivo del JFrame Guardar

5. Programación del botón Guardar

Para realizar la programación del botón Guardar se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnGuardar.

Programación
guardararchivo();
Realiza la llamada al método guardar archivo.

Tabla. 92 Programación del botón Guardar

6. Programación del botón Cancelar

Para realizar la programación del botón Cancelar se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnCancelar.

Programación
Guardar.this.dispose(); TxtCodigo.setText(" ");
Destruye la venta guardar y borra el código numérico del área de texto del JFrame Guardar.

Tabla. 93 Programación del botón Cancelar

7. Programación del PnlImagen

Para realizar la programación del PnlImagen se selecciona el evento formWindowActivated, que funciona al activar el JFrame Ver.

Programación
Cuando los valores de la matriz principal se almacenan en la variable código, se realiza la conversión de variables.
char[] codigoarray = vercodigo.toCharArray();
Se convierte la variable tipo String a un arreglo de caracteres, que permite identificar las posiciones del PnlImagen que se activan.

if(codigoarray[0] == '0'){F0C0.setBackground(Color.WHITE);
}else if(codigoarray[0] == '1'){F0C0.setBackground(Color.CYAN);
}else if(codigoarray[0] == '2'){F0C0.setBackground(Color.MAGENTA);
}else if(codigoarray[0] == '3'){F0C0.setBackground(Color.BLUE);
}else if(codigoarray[0] == '4'){F0C0.setBackground(Color.YELLOW);
}else if(codigoarray[0] == '5'){F0C0.setBackground(Color.GREEN);
}else if(codigoarray[0] == '6'){F0C0.setBackground(Color.RED);
}else if(codigoarray[0] == '7'){F0C0.setBackground(INACTIVO);
}
…….
…….
…….
 	
if(codigoarray[318] == '0'){F15C18.setBackground(Color.WHITE);
}else if(codigoarray[318] == '1'){F15C29.setBackground(Color.CYAN);
}else if(codigoarray[318] == '2'){F15C29.setBackground(Color.MAGENTA);
}else if(codigoarray[318] == '3'){F15C29.setBackground(Color.BLUE);
}else if(codigoarray[318] == '4'){F15C29.setBackground(Color.YELLOW);
}else if(codigoarray[318] == '5'){F15C29.setBackground(Color.GREEN);
}else if(codigoarray[318] == '6'){F15C29.setBackground(Color.RED);
}else if(codigoarray[318] == '7'){F15C29.setBackground(INACTIVO);
}

Cada posición de la matriz del PnlImagen se identifica desde la Fila 0 – Columna 0 (F0C0) hasta la Fila 15 – Columna 29 (F15C29). Para saber el color que le corresponde a cada posición de la matriz del PnlImagen se coloca los ifs, que analizan cada posición del arreglo de caracteres y de acuerdo al código que se encuentra, se pinta la casilla del color correspondiente.

Tabla. 94 Programación del PnlImagen

4.1.3.2.5 JFrame Ver

El JFrame Ver es la pre-visualización del mensaje que se diseña antes de cargarse al dispositivo de rotulación RGB. Esta ventana ofrece la ventaja de ver su mensaje en una matriz compacta.

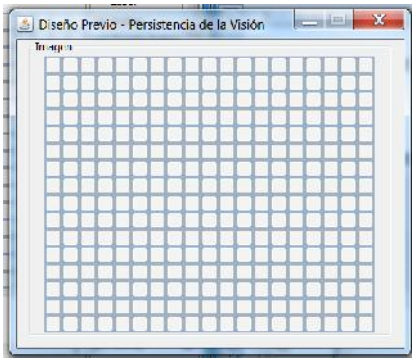


Figura. 91 JFrame Ver

Para programar este JFrame se realiza los siguientes pasos:

1. **Importación de librerías:**

Para importar la librería se escribe el código import java.awt.Color, que sirve para escoger los colores del PnlImagen.

Librería	Descripción
import java.awt.Color	Permite utilizar los distintos métodos de clase Color.

Tabla. 95 Libreria del JFrame Ver

2. **Declaración de variables y objetos**

Las variables y los objetos sirven para almacenar datos que se utilizan en el JFrame. (Ver tabla 96 y tabla 97)

Variable	Tipo	Descripción
vercodigo	String	Guarda los valores de la matriz principal del JFrame Nuevo.

Tabla. 96 Variable del JFrame Ver

Objeto	Tipo	Descripción
INACTIVO	Color	Establece el color para la opción desactivada del PnlImagen.

Tabla. 97 Objeto del JFrame Ver

3. Programación del PnlImagen:

Para realizar la programación del PnlImagen se selecciona el evento formWindowActivated, que funciona al activar el JFrame. (Ver tabla 98)

Programación
Cuando los valores de la matriz principal se almacenan en la variable vercodigo, se realiza la conversión de variables.
char[] codigoarray = vercodigo.toCharArray();
Se convierte la variable tipo String a un arreglo de caracteres, esto permite identificar las posiciones del PnlImagen que se activan.
if(codigoarray[318] == '0'){F15C29.setBackground(Color.WHITE); }else if(codigoarray[318] == '1'){F15C29.setBackground(Color.CYAN); }else if(codigoarray[318] == '3'){F15C29.setBackground(Color.BLUE); }else if(codigoarray[318] == '4'){F15C29.setBackground(Color.YELLOW); }else if(codigoarray[318] == '5'){F15C29.setBackground(Color.GREEN); }else if(codigoarray[318] == '6'){F15C29.setBackground(Color.RED); }else if(codigoarray[318] == '7'){F15C29.setBackground(INACTIVO);}
Cada posición de la matriz del PnlImagen se identifica desde Fila 0 – Columna 0 (F0C0) hasta Fila 15 – Columna 29 (F15C29).
Para saber el color que le corresponde a cada posición de la matriz del PnlImagen se colocan varios if's, para analizar cada posición del arreglo de caracteres y de acuerdo al código que se encuentra se pinta la casilla del color correspondiente.

Tabla. 98 Programación del PnlImagen

4.1.3.2.6 JFrame ConfiguracionPuerto

El JFrame ConfiguracionPuerto permite establecer los parámetros de comunicación serial para el envío de datos. (Ver figura. 92)

Figura. 92 JFrame ConfiguracionPuerto

Ítem	Nombre del Componente
1	PnlConfiguracion
2	BtnAceptar
3	BtnCancelar

Tabla. 99 Descripción JFrame ConfiguracionPuerto

Para programar este JFrame se realiza los siguientes pasos:

1. **Importación de librerías**

Para iniciar con la programación del JFrame se importan las librerías que se utilizan en el programa. (Ver tabla 100)

Librería	Descripción
import java.awt.*	Permite utilizar los métodos de la clase Image.
import gnu.io.SerialPort;	Permite utilizar los métodos de la clase SerialPort.

Tabla. 100 Librería del JFrame ConfiguracionPuerto

2. Declaración de variables y objetos

Las variables y los objetos sirven para almacenar datos que se utilizan en el JFrame ConfiguracionPuerto. (Ver tabla 101 y tabla 102)

Variable	Tipo	Descripción
BDATOS	int	Guarda la configuración de los bits de datos.
PARIDAD	int	Guarda la configuración de paridad.
BPARADA	int	Guarda la configuración de los bits de parada.

Tabla. 101 Declaración de variables del JFrame ConfiguracionPuerto

Objeto	Tipo	Descripción
configuraciom	ComunicacionSerial	Permite utilizar los métodos y datos de la clase ComunicacionSerial.

Tabla. 102 Objeto del JFrame ConfiguracionPuerto

3. Programación del botón Restaurar Valores

Para realizar la programación del botón Restaurar Valores se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnRestaurar.

Programación
CbxBsegundos.setSelectedIndex(2); CbxBdatos.setSelectedIndex(3); CbxParidad.setSelectedIndex(0); CbxBparada.setSelectedIndex(0);
Establece las listas desplegables en un valor específico.

Tabla. 103 Programación del botón Restaurar Valores

4. Programación del botón Aceptar

Para realizar la programación del botón Aceptar se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnAceptar.

Programación

if (CbxBdatos.getSelectedIndex() == 0){
BDATOS = SerialPort.DATABITS_5;
}else if(CbxBdatos.getSelectedIndex() == 1){
BDATOS = SerialPort.DATABITS_6;
}else if(CbxBdatos.getSelectedIndex() == 2){
BDATOS = SerialPort.DATABITS_7;
}else if (CbxBdatos.getSelectedIndex() == 3) {
BDATOS = SerialPort.DATABITS_8;
}

El if se encarga de verificar el índex en el que se encuentra la lista desplegable y de acuerdo a la opción que se selecciona, se guarda la configuración de los bits de datos.

if (CbxBparada.getSelectedIndex() == 0){
BPARADA = SerialPort.STOPBITS_1;
}else if(CbxBparada.getSelectedIndex() == 1){
BPARADA = SerialPort.STOPBITS_1_5;
}else if(CbxBparada.getSelectedIndex() == 2){
BPARADA = SerialPort.STOPBITS_2;
}

El if se encarga de verificar el índex en el que se encuentra la lista desplegable y de acuerdo a la opción que se selecciona, se guarda la configuración de los bits de parada.

if (CbxParidad.getSelectedIndex() == 0){
PARIDAD = SerialPort.PARITY_NONE;
}else if(CbxParidad.getSelectedIndex() == 1){
PARIDAD = SerialPort.PARITY_EVEN;
}else if(CbxParidad.getSelectedIndex() == 2){
PARIDAD = SerialPort.PARITY_ODD;
}

El if se encarga de verificar el índex en el que se encuentra la lista desplegable y de acuerdo a la opción que se selecciona, se guarda la configuración de paridad.
configuracion.parametros();
Envía los parámetros de configuración a la clase ComunicacionSerial.

Tabla. 104 Programación del botón Aceptar

5. Programación del botón Cancelar

Para realizar la programación del botón Cancelar se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnCancelar.

Programación
ConfiguracionPuerto.this.dispose();
Destruye la ventana Configuración del puerto.

Tabla. 105 Programación botón Cancelar

4.1.3.2.7 JFrame Cargando

El JFrame Cargando permite visualizar el progreso del envío de datos al dispositivo de rotulación RGB. (Ver figura 93)

Figura. 93 JFrame Cargando

Ítem	Nombre del Componente
1	LblTitulo1
2	LblTitulo2
3	LblTitulo3
4	LblCarpeta
5	LblLaptop
6	BarProgreso
7	BtnCargar
8	BtnCancelar

Tabla. 106 Descripcion JFrame Cargando

Para programar este JFrame se realizan los siguientes pasos:

1. **Importación de librerías**

Para la programación del JFrame se importan las siguientes librerías, con el fin de mostrar mensajes de información para el usuario.

Librería	Descripción
import javax.swing.JOptionPane;	Permite utilizar ventanas emergentes que muestran avisos temporales.

Tabla. 107 Librería del JFrame Cargando

2. **Declaración de variables y objetos**

Las variables y los objetos sirven para almacenar datos que se utilizan en el JFrame Cargando.

Variable	Tipo	Descripción
progreso	int	Variable que permite ver el progreso de la transferencia de datos.
cargarcodigo	String[][]	Guarda el código numérico de la matriz principal.

Tabla. 108 Declaración variables del JFrame Cargando

Objeto	Tipo	Descripción
com	ComunicacionSerial	Permite utilizar los métodos y datos de la clase ComunicacionSerial.

Tabla. 109 Objeto del JFrame Cargando

3. **Programación del botón Cargar**

Para realizar la programación del botón Cargar se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnCargar.

Programación
LblCarpeta.setVisible(true); LblTitulo2.setVisible(true); LblTitulo3.setVisible(true);
Permite visualizar la animación del JFrame Cargando.

for (int columnas = 0; columnas <= 29; columnas++){
for(int filas = 0; filas <= 15; filas++){
progreso = progreso + 1;
BarProgreso.setValue(progreso);
com.txserial(cargarcodigo[filas][columnas]);

} }
El for envia los datos de la matriz principal al dispositivo de rotulación RGB, además permite visualizar el progreso de la transmisión de datos.

if(progreso == 304){
JOptionPane.showMessageDialog();
this.dispose();
}

Cuando finaliza la transmisión de datos presenta una ventana de información y cierra el JFrame Cargando.

Tabla. 110 Programación del botón Cargar

4. Programación del botón Cancelar

Para realizar la programación del botón Cancelar se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnCancelar.

Programación
Cargando.this.dispose();
Destruye la ventana cargando y cierra la comunicación serial.

Tabla. 111 Programación del botón Cancelar

4.1.3.2.8 JFrame Abrir

El JFrame Abrir permite visualizar el código numérico del archivo abierto, además de que permite ver la imagen del archivo. (Ver figura 94)

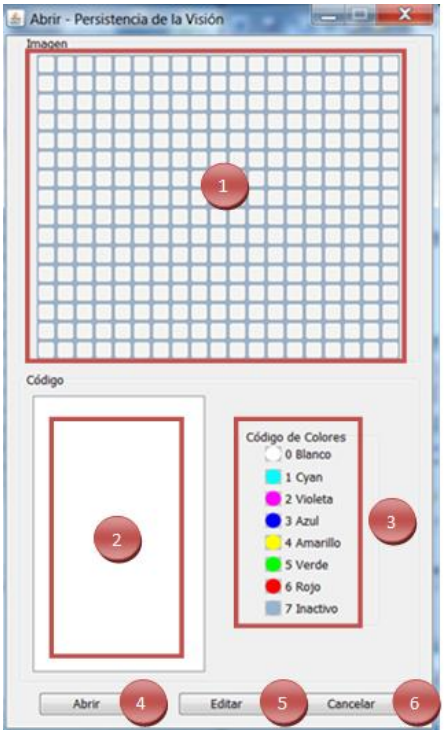


Figura. 94 JFrame Abrir

Ítem	Nombre del Componente
1	PnlImagen
2	PnlCodigo
3	PnlCodColores
4	BtnAbrir
5	BtnEditar
6	BtnCancelar

Tabla. 112 Descripción del JFrame Abrir

Para programar este JFrame se realiza los siguientes pasos:

1. **Importación de librerías:**

Para la programación del JFrame se importan las librerías que se utilizan en el programa.

Librería	Descripción
import java.awt.Color;	Permite utilizar los métodos de la clase Color.
import javax.swing.JFileChooser;	Permite seleccionar y/o grabar cualquier archivo dentro de la PC.
import javax.swing.JOptionPane;	Permite utilizar ventanas emergentes que muestran avisos temporales.
import javax.swing.filechooser.FileNameExtensionFilter;	Permite filtrar los archivos que se visualizaran en el JFileChooser.

Tabla. 113 Declaracion de librerias del JFrame Abrir

2. Declaración de variables y objetos:

Las variables y los objetos sirven para almacenar datos que se utilizan en el JFrame Abrir.

Variable	Tipo	Descripción
codigoabrir	String	Guarda el código numérico de la matriz principal.

Tabla. 114 Variable del JFrame Abrir

Objeto	Tipo	Descripción
editor	Editor	Permite utilizar los métodos y datos de la clase Editor.
INACTIVO	Color	Establece el color para la opción desactivada de la matriz principal.

Tabla. 115 Objetos del JFrame Abrir

3. Declaración del constructor

En el JFrame Abrir se añade un constructor para crear un objeto de la clase Editor.

Constructor	Descripción
Abrir	Inicializa los elementos que se crean en el JFrame Abrir.
Programación	
editor = new Editor();	
Inicializa el objeto de la clase Editor.	
BtnEditar.setEnabled(false);	
Deshabilita el botón Editar.	

Tabla. 116 Declaración del constructor del JFrame Abrir

4. Declaración de los métodos

El JFrame Abrir cuenta con métodos que permiten abrir un código numérico que se guarda con anterioridad. (Ver tabla 117 y tabla 118)

Método	Tipo	Descripción
cargarcontenido	void	Este método carga el contenido del archivo abierto al PnlImagen y área de texto del JFrame Abrir.
Programación		
TxtCodigo.setText(contenido);		
Envía el contenido del archivo txt abierto al área de texto del JFrame Abrir.		
BtnEditar.setEnabled(true);		
Habilita el botón editar.		
Cuando los valores del archivo abierto se almacenan en la variable codigoabrir, se realiza la conversión de variables.		
char[] codigoarray = vercodigo.toCharArray();		
Se convierte la variable tipo String a un arreglo de caracteres, esto permite identificar las posiciones del PnlImagen que se activan.		

if(codigoarray[0] == '0'){F0C0.setBackground(Color.WHITE);
}else if(codigoarray[0] == '1'){F0C0.setBackground(Color.CYAN);
}else if(codigoarray[0] == '2'){F0C0.setBackground(Color.MAGENTA);
}else if(codigoarray[0] == '3'){F0C0.setBackground(Color.BLUE);
}else if(codigoarray[0] == '4'){F0C0.setBackground(Color.YELLOW);
}else if(codigoarray[0] == '5'){F0C0.setBackground(Color.GREEN);
}else if(codigoarray[0] == '6'){F0C0.setBackground(Color.RED);
}else if(codigoarray[0] == '7'){F0C0.setBackground(INACTIVO);}
…….
…….
if(codigoarray[318] == '0'){F15C29.setBackground(Color.WHITE);
}else if(codigoarray[318] == '1'){F15C29.setBackground(Color.CYAN);
}else if(codigoarray[318] == '2'){F15C29.setBackground(Color.MAGENTA);
}else if(codigoarray[318] == '3'){F15C29.setBackground(Color.BLUE);		

}else if(codigoarray[318] == '4'){F15C29.setBackground(Color.YELLOW); }else if(codigoarray[318] == '5'){F15C29.setBackground(Color.GREEN); }else if(codigoarray[318] == '6'){F15C29.setBackground(Color.RED); }else if(codigoarray[318] == '7'){F15C29.setBackground(INACTIVO); }
Cada posición de la matriz del PnlImagen se identifica desde Fila 0 – Columna 0 (F0C0) hasta Fila 15 – Columna 29 (F15C29).
Para saber el color que le corresponde a cada posición de la matriz del PnlImagen se coloca los ifs para analizar cada posición del arreglo de caracteres y de acuerdo al código que se encuentra, se pinta la casilla del color correspondiente.

Tabla. 117 Método cargarcontenido del JFrame Abrir

Método	Tipo	Descripción
abrirarchivo	void	Este método sirve para abrir un archivo txt.
Programación		
JFileChooser fc = new JFileChooser();		
Crea una ventana para buscar en los archivos de la computadora.		
FileNameExtensionFilter filtro=new FileNameExtensionFilter("Archivos txt","txt"); fc.setFileFilter(filtro);		
Permite ver solo los archivos txt.		

if(fc.showOpenDialog(this) == JFileChooser.APPROVE_OPTION){
……
}		

El if permite saber si se escogió un archivo para abrir.		
File f = fc.getSelectedFile(); String contenido = "";		
Cuando se escoge un archivo para abrir se iguala a un objeto de la clase File.		

try {
contenido = editor.abrirtxt(f.getAbsolutePath());
cargarcontenido(contenido);
} catch (Exception ex) {
JOptionPane.showMessageDialog();
}		

El try – catch encierra el método que permite abrir un archivo txt y cargarlo al JFrame Abrir.		

Tabla. 118 Método abrir archivo del JFrame Abrir

5. Programación del botón Abrir

Para realizar la programación del botón Abrir se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnAbrir.

Programación
abrirarchivo();
Realiza la llamada al método abrir archivo.

Tabla. 119 Programación del botón Abrir

6. **Programación del botón Cancelar**

Para realizar la programación del botón Cancelar se selecciona el evento ActionPerformed, que funciona al hacer clic sobre el BtnCancelar.

Programación
Abrir.this.dispose(); TxtCodigo.setText(" ");
Destruye la venta abrir y borra el código numérico del área de texto del JFrame Abrir.

Tabla. 120 Programación del botón Cancelar

4.1.3.3 Declaración de clases JAVA

Para la creación de la interfaz del dispositivo de rotulación RGB, se programan tres clases JAVA:

4.1.3.3.1 Clase para comunicación serial

La clase comunicación serial consta de varias partes que se detallan a continuación:

1. **Importación de librerías**

Para iniciar con la programación de la clase para comunicación serial se importan las librerías que se utilizan en el programa.

Librería	Descripción
import java.io.*	Permite la salida de los datos por el puerto serial.
import gnu.io.*	Permite realizar la configuración del puerto serial.

Tabla. 121 Librerías Clase comunicación serial

2. **Declaración de variables y objetos**

La clase comunicación serial tiene las siguientes variables:

Variables	Tipo	Descripción
puerto	String	Guarda la identificación del puerto que se utiliza para la comunicación.
baudios	int	Guarda la información de la velocidad que se utiliza para la comunicación.
BDatos	int	Guarda la información de los bits de datos que se utiliza para la comunicación.
BParada	int	Guarda la información de los bits de parada que se utiliza para la comunicación.
Paridad	int	Guarda la información de la paridad que se utiliza para la comunicación.

Tabla. 122 Variables Clase comunicación serial

La clase comunicación serial tiene los siguientes los objetos:

Objetos	Tipo	Descripción
idPuerto	CommPortIdentifier	Identifica si el puerto que se selecciona está disponible.
puertoSerie	SerialPort	Permite realizar las configuraciones para establecer la conexión serial.
salida	OutputStream	Almacena y envía los datos por el puerto serie.

Tabla. 123 Objetos Clase comunicación serial

3. **Declaración de los métodos**

La clase comunicación serial cuenta con dos métodos que permiten establecer los parámetros de la comunicación serial para el envío de datos. (Ver tabla 124 y tabla 125)

Método	Tipo	Descripción
parametros	void	Permite modificar la configuración del puerto desde una JFrame. El método recibe del JFrame cinco variables que se igualan a las variables correspondientes del puerto serial.
Programación		**Descripción**
puerto=com		Asigna el valor com a puerto.
baudios=baud		Asigna el valor baud a baudios.
BDatos=bdatos		Asigna el valor bdatos a BDatos.
BParada=bparada		Asigna el valor bparada a BParada.
Paridad=paridad		Asigna el valor paridad a Paridad.

Tabla. 124 Método parámetros de la Clase comunicación serial

Método	Tipo	Descripción
txserial	void	Este método recibe el mensaje a enviar y realiza la configuración del puerto para enviar el mensaje que recibe.
		Programación
La programación se cierra en un try-catch, para verificar si la comunicación se realiza con éxito o caso contrario genera una excepción del problema.		
idPuerto = CommPortIdentifier.getPortIdentifier(puerto);		
Identifica el puerto que se utiliza para la comunicación serial.		

if(idPuerto.isCurrentlyOwned())
{….}
else
{….}		

El if identifica si el puerto esta en uso o libre. Si el puerto está en uso genera un mensaje de advertencia, caso contrario permite la configuración del puerto serial.		
puertoSerie = (SerialPort) idPuerto.open("ESCRITURA",2000);		
Abre el puerto serial.		
salida = puertoSerie.getOutputStream(); puertoSerie.setSerialPortParams(baudios,BDatos,BParada,Paridad);		
Configura el puerto serial.		
salida.write(mensaje.getBytes());		
Envía el mensaje que se recibe del JFrame.		
puertoSerie.close();		
Cierra el puerto después de enviar el mensaje.		

Tabla. 125 Método txserial de la Clase comunicación serial

4.1.3.3.2 Clase Archivo

La clase Archivo permite leer y guardar el código del mensaje que se crea en la interfaz gráfica. La clase Archivo consta de los siguientes partes:

1. **Importación de librerías**

Para iniciar con la programación de la clase Archivo se importan las siguientes librerías.

Librería	Descripción
import java.io.BufferedReader;	Permite utilizar los métodos de la clase BufferedReader.
import java.io.File;	Permite utilizar los métodos de la clase File.
import java.io.FileReader;	Permite utilizar los métodos de la clase FileReader.
import java.io.IOException;	Permite manejar las excepciones que se generan en la programación.
import java.io.PrintWriter;	Permite utilizar los métodos de la clase PrintWriter.

Tabla. 126 Librerías Clase Archivo

2. **Declaración de atributos**

El atributo archivo que tiene la clase Archivo permite utilizar los métodos del tipo File:

Atributo	Tipo	Descripción
archivo	File	Guarda la información del documento txt en un atributo tipo File.

Tabla. 127 Atributo Clase Archivo

3. **Declaración del constructor**

En la clase Archivo se declara el siguiente constructor que inicializa los objetos:

Constructor	Descripción
Archivo	Crea un objeto de la clase File y especifica la ruta del archivo.
Programación	
archivo = new File(nombretxt);	
Crea el archivo en una ruta que se especifica.	

Tabla. 128 Constructor de la Clase Archivo

4. **Declaración de los métodos**

La clase Archivo cuenta con dos métodos que permiten leer los archivos de texto. (Ver tabla 128 y tabla 129)

Método	Tipo	Descripción
darcontenido	String	Lee el contenido del archivo txt y retorna el contenido del archivo a un JFrame.
Programación		
String contenido = ""; Esta variable almacena el contenido completo del archivo txt.		
FileReader fr = new FileReader(archivo); BufferedReader lector = new BufferedReader(fr); El objeto fr de la clase FileReader se construye a partir de un atributo tipo File, y se encarga de leer el archivo txt. Mientras el objeto lector de la clase bufferedReader se construye a partir de un objeto FileReader, y se encarga de almacenar la información del objeto fr.		
String linea = lector.readLine(); La función lector.readLine() permite leer el archivo línea a línea, y cada una se almacena en la variable línea.		

while(linea != null){
contenido = contenido + linea + "\n";
linea = lector.readLine();
}
El while se ejecuta mientras la variable línea es diferente de nulo, cuando la sentencia while se ejecuta la variable línea se almacena en la variable contenido.		

lector.close();
fr.close();
Cuando se termina el while cierra el objeto lector y fr para leer el contenido de otro archivo.		

return contenido; Cuando la variable contenido tiene el texto completo del archivo que se analiza, retorna la variable a un JFrame.		

Tabla. 129 Método darcontenido de la Clase Archivo

Método	Tipo	Descripción
guardararchivo	void	Permite guardar el mensaje que procede del JFrame en un archivo txt.
Programación		
PrintWriter escritor = new PrintWriter(archivo);		
El objeto escritor de la clase PrintWriter se construye a partir de un atributo tipo File, y establece el archivo en modo escritura.		
escritor.write(contenidotxt);		
Escribe el texto que se obtiene de un JFrame en el archivo txt.		
escritor.close();		
Cierra el modo de escritura del archivo txt.		

Tabla. 130 Método guardararchivo de la Clase Archivo

4.1.3.3.3 Clase Editor

La clase Editor crea, abre y guarda el mensaje en un archivo txt y consta de las siguientes partes:

1. **Importación de librerías**

Para iniciar con la programación de la clase Editor se importa la librería import java.io.IOException.

Librería	Descripción
import java.io.IOException;	Permite manejar las excepciones que se generan en la programación.

Tabla. 131 Librería de la Clase Editor

2. **Declaración de atributos**

El atributo que tiene la clase Editor permite utilizar los métodos tipo Archivo:

Atributo	Tipo	Descripción
archivo	Archivo	Permite utilizar los métodos de la clase Archivo.

Tabla. 132 Declaración de atributos

3. **Declaración del constructor:**

En la clase Editor se declara el constructor Editor para inicializar los objetos tipo Archivo:

Constructor	Descripción
Editor	El constructor asigna nulo al objeto archivo.
Programación	
archivo = null;	
Crea un archivo nuevo.	

Tabla. 133 Constructor de la Clase Editor

4. **Declaración de los métodos:**

La clase Editor cuenta con cuatro métodos que sirven para abrir, guardar y crear archivos txt.

Método	Tipo	Descripción
abrirtxt	String	Abre un archivo txt existente. Este método tiene un parámetro que contiene la ruta del archivo y retorna el contenido del archivo txt.
Programación		
String contenido = ""; Esta variable almacena el contenido completo del archivo txt.		
archivo = new Archivo(nombretxt); Crea el archivo que se va abrir.		

try {contenido = archivo.darcontenido();
} catch (IOException ex) {
System.out.println("Error abrir archivo"+ ex);
}return contenido;		

Se utiliza el try – catch para manejar la excepción que se da al abrir el archivo. Dentro del try-catch se llama al método darcontenido de la clase Archivo para capturar el contenido del archivo txt que se abre. Por último al no registrar ningún problema el método darcontenido, retorna el texto a un JFrame.		

Tabla. 134 Método abrirtxt de la Clase Editor

Método	Tipo	Descripción
creartxt	void	Crea un archivo txt.
Programación		
archivo = null;		
Crea un archivo vacío.		

Tabla. 135 Método creartxt de la Clase Editor

Método	Tipo	Descripción
saberarchivo	boolean	Permite saber si un archivo está vacío o contiene texto que se guarda con anterioridad.
Programación		
return archivo = null;		
Retorna una afirmación si el archivo es vacío.		

Tabla. 136 Método saberarchivo de la Clase Editor

Método	Tipo	Descripción
guardartxt	void	Guarda un archivo txt, posee un parámetro para el contenido que recibe del JFrame y otro para la ruta del archivo txt.
Programación		

if(archivo == null){
archivo = new Archivo(rutatxt);}
try {archivo.guardararchivo(contenidotxt);
} catch (IOException ex) {
System.out.println("Error guardar archivo"+ ex);}		

El if permite saber si el archivo está vacío o no, si está vacío se crea un archivo que específica su ruta. Se utiliza el try – catch para manejar la excepción que se da al guardar el archivo. Dentro del try-catch se llama al método guardararchivo de la clase Archivo y asigna el texto que se obtiene de un JFrame.		

Tabla. 137 Método guardartxt de la Clase Editor

4.2 PROGRAMACIÓN DEL MICROCONTROLADOR

4.2.1 Requerimientos previos

Para realizar la programación del microcontrolador es necesario contar con un compilador y un grabador. En el sistema de rotulación RGB se emplea el compilador CCS PIC-C versión 4.023 y el grabador PICkit versión 2.61. (Ver figura 95 y figura 96)

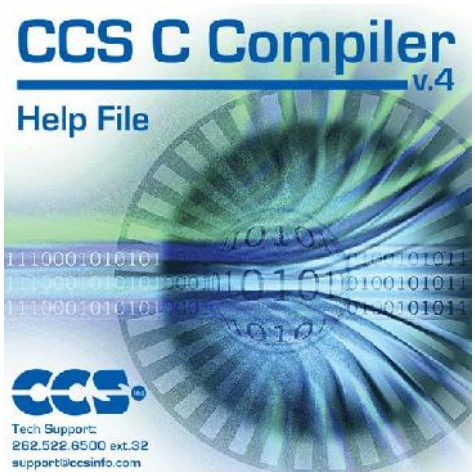


Figura. 95 Portada CCS C Compilar

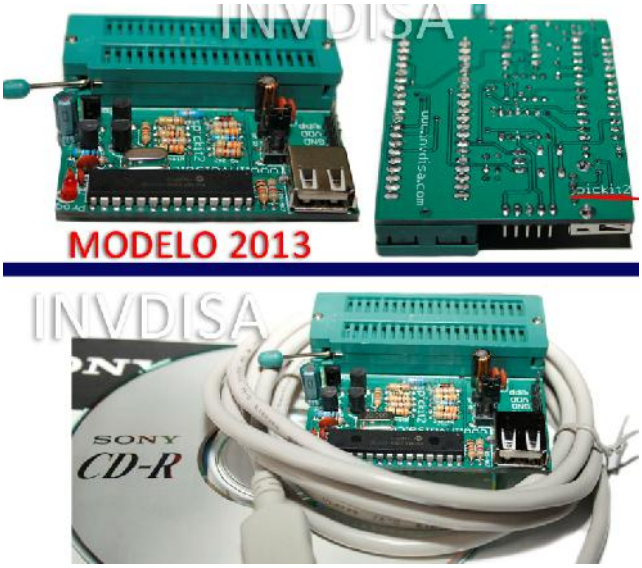


Figura. 96 Portada Pickit

4.2.2 Funcionalidad

El programa que se desarrolla en CCS PIC-C, es capaz de almacenar los datos que se reciben de la PC por medio de los módulos inalámbricos HC05, los envía hacia los puertos en un tiempo mínimo y sin perder ningún dato al manipularlos.

4.2.3 Programación

4.2.3.1 Declaración de variables

Para iniciar con la programación del microcontrolador, se declara las variables y su función en el programa. (Ver tabla 138)

Variables	Tipo	Descripción
mensaje[240]	char	Guarda la información que se recibe por la UART, para proporcionar la señal de control a los integrados DM74154.
mensaje1[240]	char	Guarda la información que se recibe por la UART, para proporcionar la señal de control a los integrados DM74154.
mensaje2[240]	char	Guarda la información que se recibe por la UART, para proporcionar la señal de control a los integrados DM74154.
mensaje3[240]	char	Guarda la información que se recibe por la UART, para proporcionar la señal de control a los integrados DM74154.
mensaje4[240]	char	Guarda la información que se recibe por la UART, para proporcionar la señal de control a los integrados DM74154.
mensaje5[240]	char	Guarda la información que se recibe por la UART, para proporcionar la señal de control a los integrados DM74154.
cont	int	Cuenta las posiciones de los arreglos al guardar los datos que se reciben por la UART.
contc	int	Cuenta las posiciones de los datos que se guardan en los arreglos, para publicarlos por el puerto A del microcontrolador PIC18F452.
contf	int	Cuenta el envío de los datos de entrada por los puertos B y D, hacia los integrados DM74154.
num	int	Guarda el dato de los arreglos y lo transforma en entero para publicarlo por el puerto A.
tiempo	int	Asigna el tiempo de retardo para que los datos se publiquen de nuevo.

Tabla. 138 Declaración de variables

4.2.3.2 Desarrollo del Programa

Las condiciones que se presentan para el desarrollo del programa del microcontrolador son:

- Recibir los datos de la UART sin que afecte la ejecución del programa principal.
- Capacidad de almacenamiento de los datos que se reciben en arreglos.
- Agilidad en el envió de los datos por los puertos A, B y D.
- Manejo de tiempos de retardo para la publicación de los datos a través de los puertos.
- Publicación de datos constante sin pérdida de ninguno de ellos.

Después de las condiciones se procede a dividir el programa en 3 partes que son:

1. **Declaración de directivas y variables**

Es indispensable emplear las directivas #use delay (clock) y #use rs232 (baud, parity, xmit, rcv, bits), porque se trabaja con trasmisión RS232 y tiempos de retardos. La declaración de las variables es de tipo global ya que se emplean tanto en la programación de la interrupción de la UART, como en la programación del programa principal. (Ver figura 97)

Figura. 97 Declaración de variables en el compilador CCS

2. Programación de la interrupción de la UART

Para cumplir con las condiciones de recibir los datos de la UART sin que afecte la ejecución del programa principal y la capacidad de almacenamiento de los datos en arreglos, se programa la interrupción de la UART por RX, que permite almacenar los datos conforme se envían a la UART sin afectar la ejecución del programa principal, ya que la interrupción solo se ejecuta cuando un dato se recibe por la UART. (Ver tabla 139)

Programación	Descripción
#int_RDA void RDA_isr()	Declaración de la interrupción de la UART.
for(meng=0;meng<6;meng++){ …}	Permite grabar los 1440 datos que se reciben, en seis arreglos de 240 posiciones.
for(cont=0;cont<240;cont++){ …}	Permite grabar los datos que se reciben, en cada una de las 240 posiciones de los arreglos.
mensaje[cont]=getchar();	Guarda el dato que se recibe según la posición que dicta el contador cont.

if ((mensaje[cont]!=0x0D) &&
(mensaje[cont]<='9') &&
(mensaje[cont]>='0')){
…}
	Verifica que el dato que se recibe sea un número que va del 0 al 9.

if(meng==0){ mensaje1[cont]=mensaje[cont];}	Realiza el almacenamiento de los primeros 240 datos que se reciben en la variable mensaje1 de acuerdo a la posición que dicta el contador cont.
if(meng==1){ mensaje2[cont]=mensaje[cont];}	Realiza el almacenamiento de los segundos 240 datos que se reciben en la variable mensaje2 de acuerdo a la posición que dicta el contador cont.
if(meng==2){ mensaje3[cont]=mensaje[cont];}	Realiza el almacenamiento de los terceros 240 datos que se reciben en la variable mensaje3 de acuerdo a la posición que dicta el contador cont.
if(meng==3){ mensaje4[cont]=mensaje[cont];}	Realiza el almacenamiento de los cuartos 240 datos que se reciben en la variable mensaje de acuerdo a la posición que dicta el contador cont.
if(meng==4){ mensaje5[cont]=mensaje[cont];}	Realiza el almacenamiento de los quintos 240 datos que se reciben en la variable mensaje de acuerdo a la posición que dicta el contador cont.
if(meng==5){ mensaje[cont]=mensaje[cont];}	Realiza el almacenamiento de los sextos 240 datos que se reciben en la variable mensaje de acuerdo

	a la posición que dicta el contador cont.
mensaje[cont]='\0';	Asigna un espacio vacío para saber que el arreglo se termina de acuerdo a la posición que dicta el contador cont.
puts(mensaje);	Envía el arreglo que se recibe por medio de la UART.

Tabla. 139 Descripción de la programación de la interrupción de la UART

3. Programación del programa principal

Para cumplir con las condiciones de agilidad en el envío de los datos por los puertos A, B y D y la publicación de datos constante sin pérdida de ninguno de ellos, se realiza una publicación de datos por medio de bucles de repetición FOR y el control de los datos a publicar solo maneja el puerto A. (Ver tabla 140)

Programación	Descripción
void main (void){…}	Declaración del programa principal.
enable_interrupts(INT_RDA); enable_interrupts(GLOBAL);	Habilitación de la interrupción de la UART.
for(cont=0;cont<240;cont++){….}	Permite incrementar las posiciones para los arreglos.
OUTPUT_A(7);	Da la señal de control para que los integrados DM74154 deshabiliten sus salidas.
OUTPUT_B(contf+(contf<<4));	Envía el dato del contador contf a los bit más y menos significativos del puerto B.
OUTPUT_D(contf)	Envía el dato del contador contf al puerto D.
num=(int)mensaje[contc]-48;	Transforma a entero la variable tipo char del arreglo y lo almacena en la variable num.
OUTPUT_A(num);	Envía el dato de la variable num al puerto A.
for(cont1=0;cont1<4;cont++){ …mensaje1[contc]; …mensaje2[contc]; …}	Da el número de veces que se muestra un mismo mensaje.
contf=contf+1;	Carga los valores que se envían por los puertos B y D
if(contf>15){ contf=0;}	Se limita al contador contf en un rango de 0 – 15.
OUTPUT_A(7); delay_ms(tiempo);	Da el tiempo de apagado de los led antes de volver a mostrar el mensaje.

Tabla. 140 Descripción de la programación del programa principal

CAPÍTULO 5

IMPLEMENTACIÓN, PRUEBAS Y RESULTADOS

5.1 UBICACIÓN DE LOS ELEMENTOS

Para la implementación del sistema de rotulación RGB y su funcionamiento se acoplan las placas de los circuitos en la hélice de la siguiente forma.

5.1.1 Montaje de placa principal y placa transistores

La placa del circuito principal y la placa de los transistores se empotran con tornillos en la parte interior superior e interior inferior de la hélice, además se incluyen soportes plásticos de 10mm de alto por 5mm de radio, para evitar que las pistas de la circuitería de las placas hagan contacto con la hélice. (Ver figura 98 y figura 99)

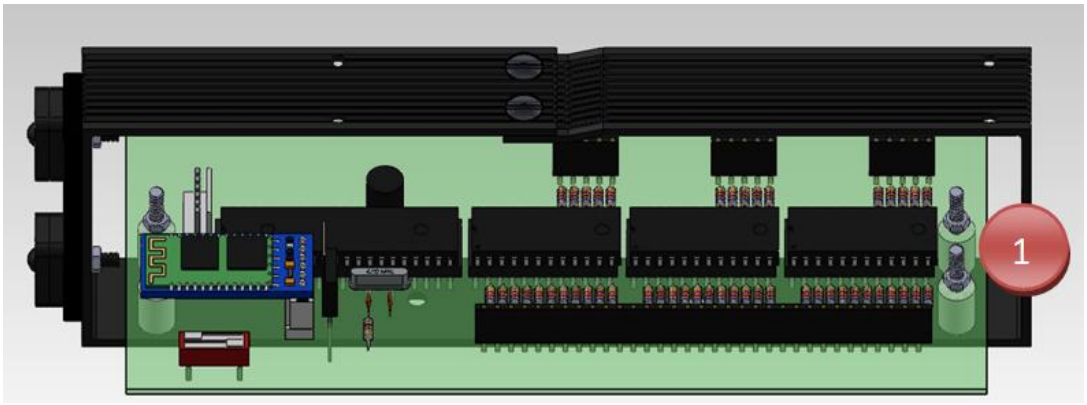


Figura. 98 Placa principal en la hélice

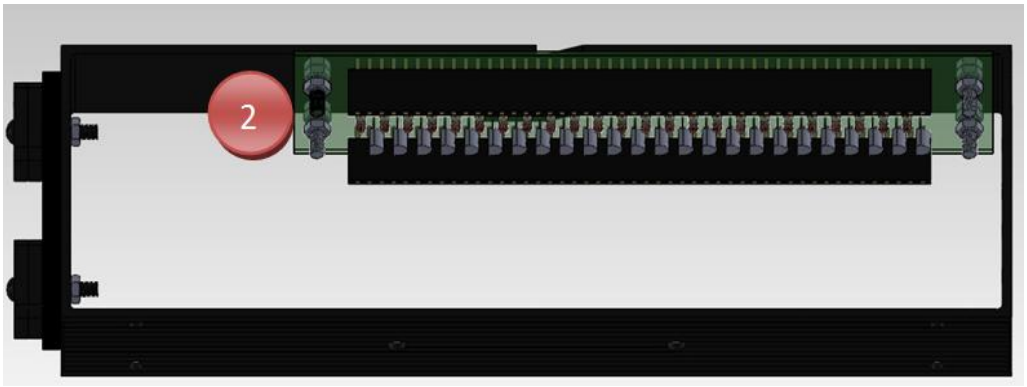


Figura. 99 Placa de los transistores en la hélice

Ítem	Cantidad	Tornillo (mm)	Largo(mm)	Broca (mm)
1	4	M3	30	2.5
2	4	M3	15	2.5

Tabla. 141 Descripción de tornillos placa principal y placa transistores

5.1.2 Montaje de la placa de led's RGB

La placa que contiene los 16 led's RGB se empotra con tornillos en la parte exterior frontal de la hélice y se incluyen soportes plásticos de 15mm de largo por 1.5mm de radio para que los cables que provienen de la placa de los transistores tengan el suficiente espacio para su conexión. (Ver figura 100)

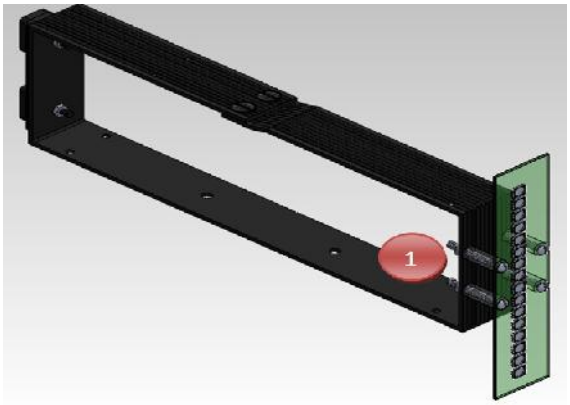


Figura. 100 Placa de los LED's en la hélice

Ítem	Cantidad	Tornillo (mm)	Largo(mm)	Broca (mm)
1	4	M3	30	2.5

Tabla. 142 Descripción tornillos placa LED's RGB

5.1.3 Montaje del dispositivo inalámbrico

Es importante que el dispositivo inalámbrico que se conecta a la placa principal sea removible puesto que se necesita configurarlo para su uso. Por esto se emplean conectores macho y hembra de 6 pines. El conector macho se suelda en la placa principal y el conector hembra se suelda directamente a los pines del módulo inalámbrico HC05. (Ver figura 101 y figura 102)

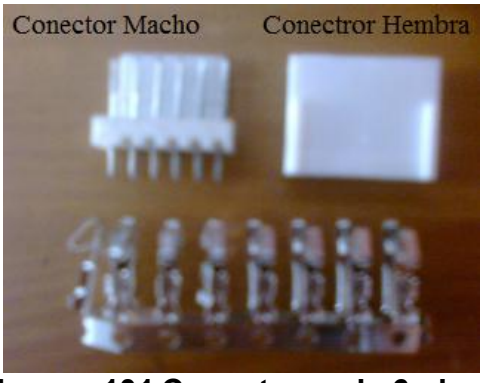


Figura. 101 Conectores de 6 pines

Figura. 102 Módulo inalámbrico con conector

5.2 CONEXIÓN ELÉCTRICA Y COMUNICACIONES

5.2.1 Conexión eléctrica

Para la conexión eléctrica se emplea una conexión en paralelo de todos los dispositivos que conforman las diferentes placas del sistema de rotulación RGB. Todos los dispositivos se alimentan con 5 [VDC], excepto el módulo inalámbrico HC05 que se alimenta con 3.3 [VDC], la carga total del circuito consume alrededor de 1.5 [A].

La conexión para la alimentación de todo el sistema de rotulación RGB, se ubica en la base metálica y se transfiere hacia la hélice por medio de los anillos de cobre y de los portaescobillas. (Ver figura 103)

Figura. 103 Conexión eléctrica

5.2.2 Comunicaciones

La comunicación entre el sistema de rotulación RGB y la PC se realiza por medio de los módulos inalámbricos HC05, estos, se configuran en modo maestro o esclavo con comandos AT. (Ver tabla 143)

Comando	Respuesta	Parámetro	Descripción
AT	OK	-	Verifica que la comunicación con el dispositivo sea exitosa.
AT+VERSION?	+VERSION:<Param> OK	Param: versión de firmware	Permite ver la versión de firmware del dispositivo.
AT+ORGL	OK	-	Vuelve a la configuración de fábrica.
AT+ADDR?	+ADDR:<Param> OK	Param: dirección del modulo Bluetooth	Devuelve la dirección MAC del dispositivo.
AT+NAME=<Param>	OK	Param: nombre del dispositivo	Permite asignar un nombre al dispositivo.
AT+ROLE=<Param>	OK	Param: 0 – Esclavo 1 – Maestro 2 – Esclavo-loop	Permite configurar el módulo como maestro o esclavo.
AT+PSWD=<Param>	OK	Param: código PIN	Permite asignar una contraseña para la conexión con otros dispositivos.
AT+UART=<Param>,<Param2>,<Param3>	OK	Param1: Baud Param2: bit parada Param3: Paridad	Permite asignar la velocidad de comunicación y los bit's de parada y paridad.
AT+CMODE=<Param>	OK	Param: 0 – conectar con una dirección especifica 1 – conectar con cualquier dirección 2 - Esclavo-Loop	Indica si el dispositivo se puede comunicar con un dispositivo o con varios.
AT+BIND=<Param>	OK	Param: MAC Dispositivo	Asigna la MAC del dispositivo con el que se va a comunicar.

Tabla. 143 Comandos AT[68]

68 ftp://imall.iteadstudio.com/IM120417010_BT_Shield_v2.2/DS_BluetoothHC05.pdf

5.2.2.1 Configuración de los módulos inalámbricos HC05

Los pasos para la configuración de los módulos inalámbricos HC05 son los siguientes.

1. Se conecta el módulo inalámbrico HC-05 a la PC a través de un adaptador USB-Serial y se coloca el PIN-KEY del módulo inalámbrico HC-05 a 3.3 VDC. (Ver figura 104)

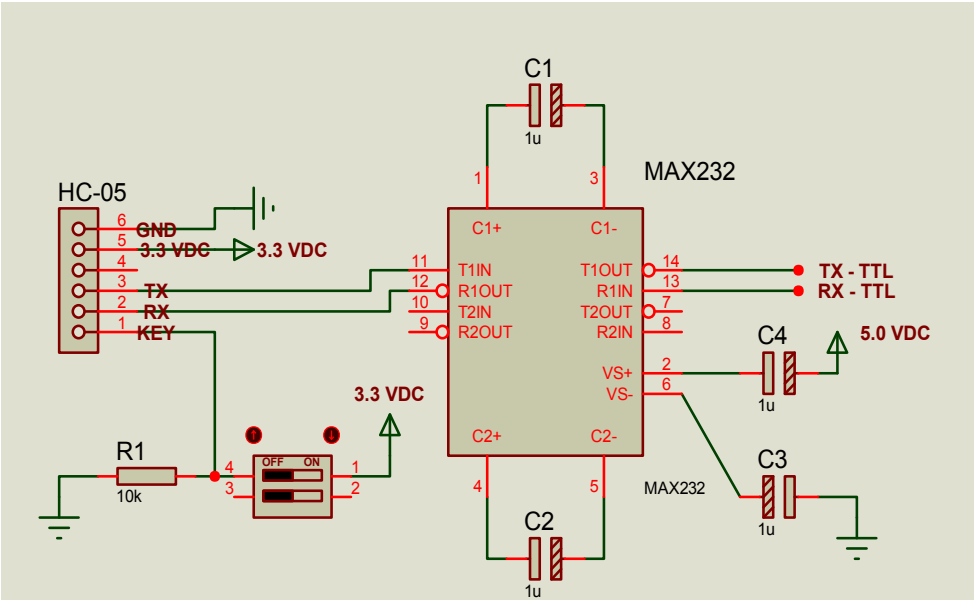


Figura. 104 Circuito para el adaptador USB – Serial

2. Se ingresa a un Terminal.exe a través de la PC y se realiza la siguiente configuración:

Bits por segundo	38400
Bits de Datos	8
Paridad	Ninguno
Bit Parada	1
Control de flujo	ninguno

Tabla. 144 Configuración Terminal.exe

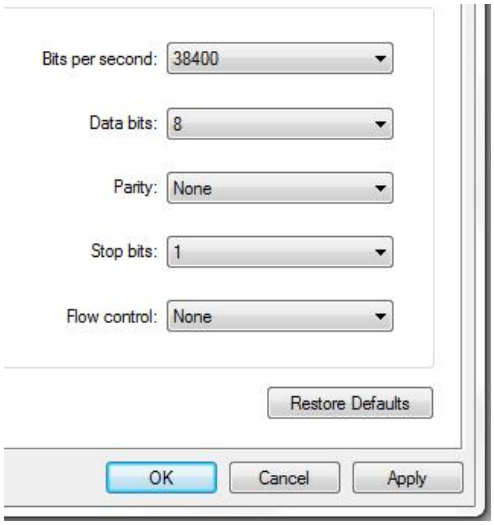


Figura. 105 Configuración Terminal.exe

3. Se envía el comando AT / ENTER y el dispositivo responde OK con esto se verifica que la conexión se realiza con éxito.

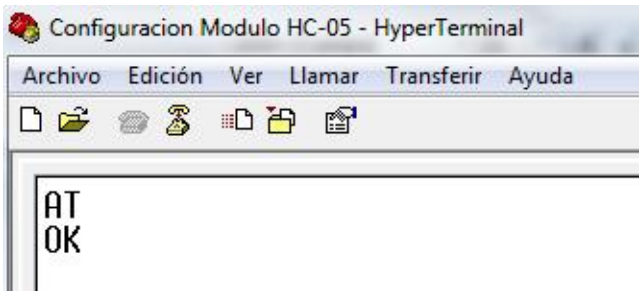


Figura. 106 Respuesta del módulo inalámbrico HC05

4. Se configura el módulo inalámbrico en modo maestro, luego se garantiza que se conecte con un único dispositivo que tenga la dirección MAC por medio del comando BIND. (Ver figura 107)

AT+NAME=MAESTRO
AT+ROLE=1
AT+CMODE=0
AT+BIND=<Dirección MAC del dispositivo Esclavo>
AT+UART=9600,0,0

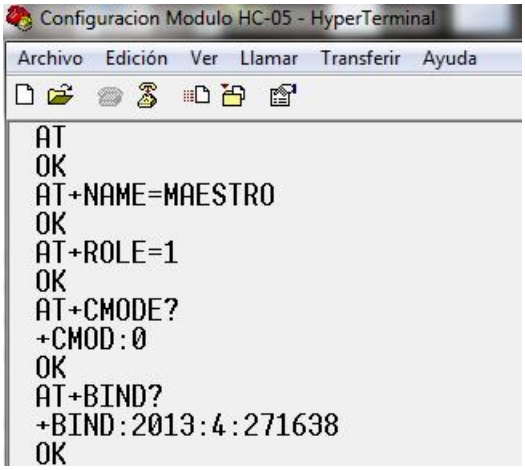


Figura. 107 Configuración módulo inalámbrico HC05 en modo maestro

5. Se configura el módulo inalámbrico en modo esclavo, para que se conecte con el módulo maestro o con una PC que tenga comunicación bluetooth. (Ver figura 108)

AT+NAME=ESCLAVO
AT+ADDR?
AT+ROLE=0
AT+CMODE=0
AT+UART=9600,0,0

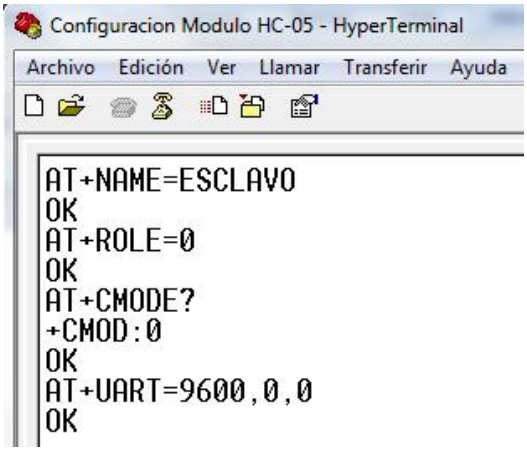


Figura. 108 Configuración módulo inalámbrico HC05 en modo esclavo

5.2.2.2 Comunicación con el módulo inalámbrico HC05/esclavo

La comunicación con el módulo HC05 en modo esclavo puede ser mediante una PC que posee comunicación bluetooth, o por medio de otro módulo inalámbrico HC05 que se configura en modo maestro y conecta a una PC.

Los pasos para vincularse con el módulo inalámbrico HC05 desde la PC son los siguientes:

1. Se ingresa a la configuración de comunicación Bluetooth de la PC y se escoge la opción de **Agregar Dispositivo**. (Ver figura 109)

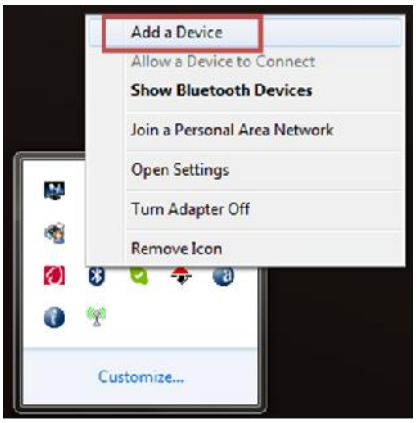


Figura. 109 Agregar nuevo dispositivo bluetooth

2. Aparece una nueva ventana donde se muestran los dispositivos que tienen activa su comunicación bluetooth, luego se escoge el dispositivo que se desea agregar y se da clic en **Siguiente**. (Ver figura 110)

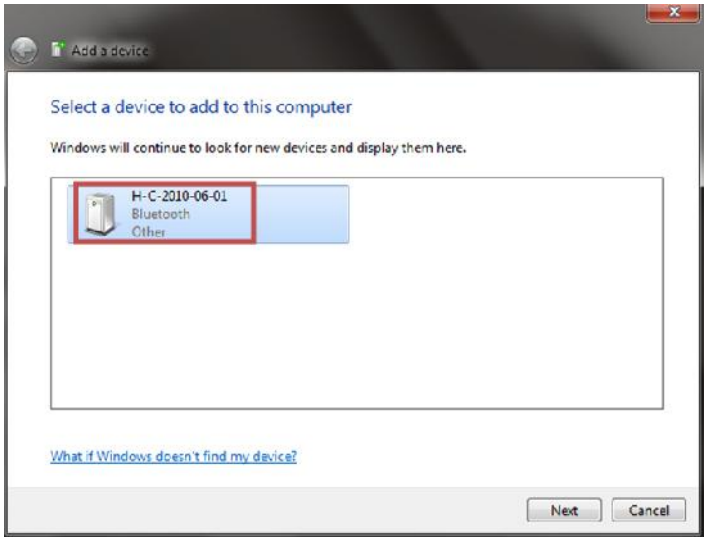


Figura. 110 Ventana de dispositivos con conexión bluetooth

3. Se despliega una ventana donde se muestran tres opciones para emparejar el dispositivo a la PC de las que se escoge la opción **Emparejar el dispositivo mediante una contraseña** y se da clic en **Siguiente**. (Ver figura 111)

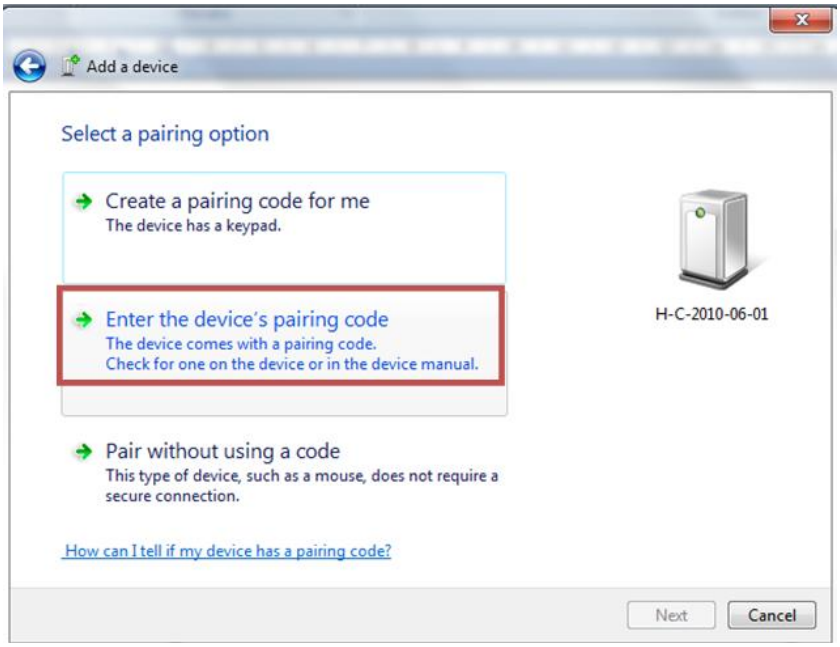


Figura. 111 Emparejar el módulo HC05 con la PC

4. Se ingresa el código de emparejamiento del dispositivo. El módulo inalámbrico HC05 tiene el código 1234 de fábrica que se puede modificar por medio de comandos AT y se da clic en **Siguiente**. (Ver figura 112)

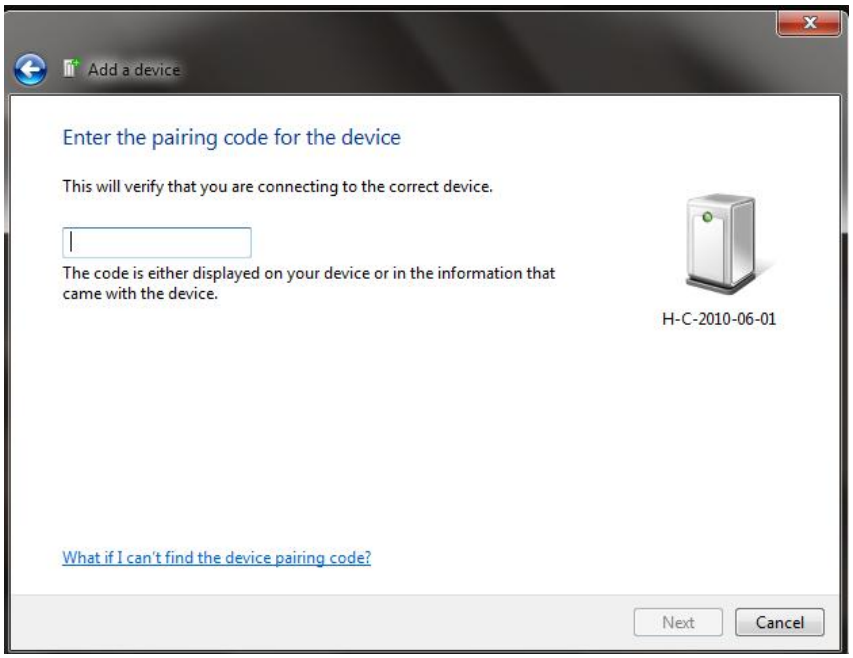


Figura. 112 Ingreso del código de emparejamiento del módulo HC05

5. Aparece una nueva ventana donde se muestran los drivers que se descargan desde la web automáticamente. (Ver figura 113)

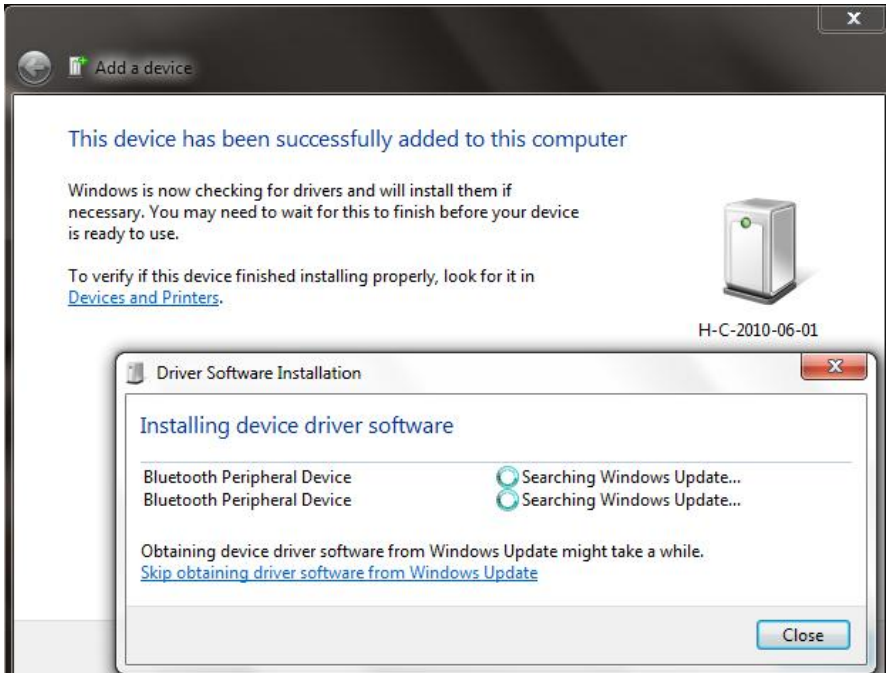


Figura. 113 Descarga de drivers para el módulo inalámbrico HC05

6. Para finalizar se asigna automáticamente un puerto de comunicación para el dispositivo inalámbrico HC05. (Ver figura 114)

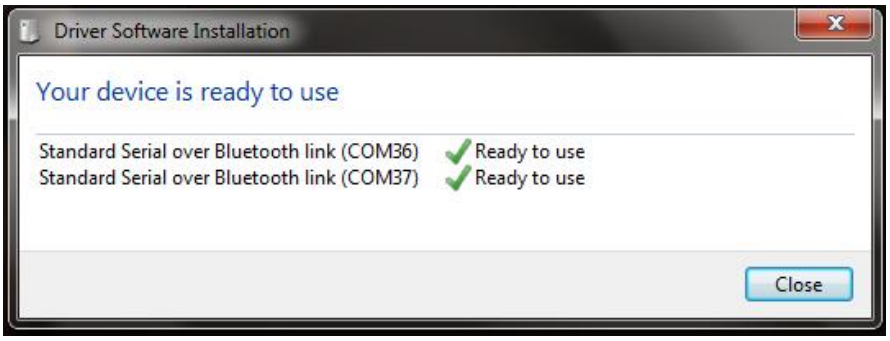


Figura. 114 Puerto de comunicación para el módulo inalámbrico HC05

5.3 PRUEBAS DE FUNCIONAMIENTO CON EL DISPOSITIVO

5.3.1 Verificación del giro del sistema de rotulación RGB

En el movimiento giratorio del dispositivo de rotulación RGB, se detecta un efecto de balanceo, vibración y recalentamiento del motor AC, que provoca inestabilidad en el sistema. El recalentamiento del motor AC se debe al exceso de fricción entre las escobillas y los anillos de cobre por la rigidez de los resortes de las escobillas.

5.3.2 Verificación de la alimentación hacia la hélice

Se evidencia que la transferencia de alimentación desde la base del sistema de rotulación RGB hacia la hélice durante el movimiento giratorio no es constante por la desconexión de las escobillas con los portaescobillas.

5.3.3 Verificación de comunicación entre la PC y el módulo inalámbrico HC05

De acuerdo a las especificaciones técnicas del módulo inalámbrico HC05 la distancia de comunicación máxima con otro dispositivo es de 10 metros, sin embargo se realizan pruebas de comunicación a varias distancias y con una línea de vista con obstáculos (paredes de concreto de 15 cm de espesor). (Ver tabla. 145)

Distancia PC – Módulo inalámbrico HC05 [m]	Resultado
1-5	Comunicación exitosa
6	Comunicación exitosa
7	Comunicación exitosa
8	Comunicación exitosa
9-10	Fallo en la comunicación

Tabla. 145 Distancia PC – Módulo inalámbrico HC05

Según los resultados que se obtienen de las mediciones de distancia entre la PC y el módulo inalámbrico HC05, se comprueba que la comunicación se interrumpe a partir de los 9 metros de distancia entre la PC y el módulo HC05, por tanto, para garantizar una comunicación exitosa se debe utilizar el sistema de rotulación RGB a una distancia máxima de 8 metros de la PC.

5.3.4 Verificación de la visualización de los mensajes en el sistema de rotulación RGB

Para la visualización del mensaje en el sistema de rotulación RGB se calcula el tiempo de encendido y apagado de los LED's RGB y la velocidad de giro del motor AC (ver capítulo 3), se establecen valores de 0.66 milisegundos para el encendido y apagado de los LED's RGB y 1000[rpm] para la velocidad de giro del motor AC.

Se evidencia que la visualización de los mensajes a través de los LED's RGB durante el movimiento giratorio de la hélice no es constante, debido a factores mecánicos que provocan cambios en la velocidad de giro del motor AC.

5.4 CORRECCIÓN DE ERRORES Y VERIFICACIÓN FINAL DE LA ESTRUCTURA

De la verificación del giro del sistema de rotulación RGB, de la verificación de la alimentación hacia la hélice y de la verificación de la visualización del mensaje, se detectan cuatro errores descritos anteriormente por lo que se proporcionan las siguientes soluciones.

5.4.1 Corrección del efecto de balanceo y vibración

Para corregir el efecto de balanceo y vibración del sistema de rotulación RGB, se colocan pesos de hierro de 5 gramos y 30 gramos, en el extremo opuesto a la placa de los LED's RGB, en la hélice. Se determina un peso aproximado de 50 gramos para lograr el equilibrio del giro de la hélice. (Ver figura 115)

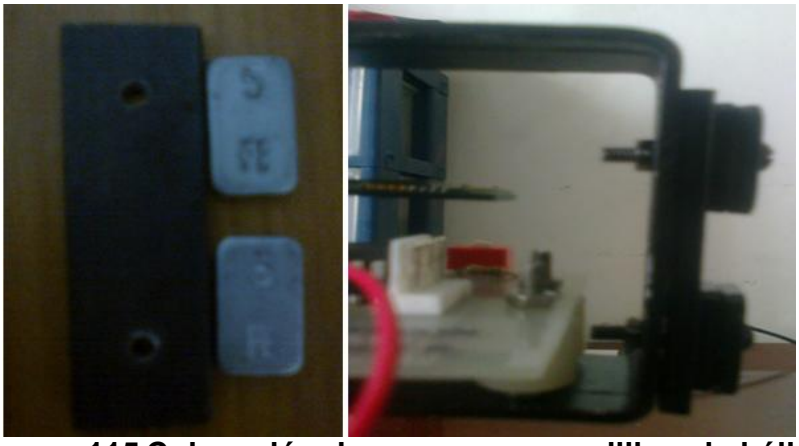


Figura. 115 Colocación de pesos para equilibrar la hélice

5.4.2 Corrección de la falla en la alimentación hacia la hélice

El movimiento giratorio que posee el sistema de rotulación RGB, no permite realizar un análisis de la entrega de energía desde la placa base hacia la hélice por medio de un osciloscopio, por lo que se realiza una prueba de fallo-corrección, que consiste en visualizar que los led's no se apaguen cuando se regule la distancia entre los portaescobillas y el disco con los anillos de cobre (distancia máxima 8[mm], mínima 4[mm]). (Ver tabla 146)

Distancia portaescobillas/disco [mm]	Resultado
8	Fallo de energía
7	Fallo de energía
6	Fallo de energía
5	Fallo de energía
4	Fallo de energía

Tabla. 146 Distancia portaescobillas/disco

Según los resultados que se obtienen de las mediciones de distancia entre los portaescobillas y el disco con los anillos de cobre, se comprueba que la interrupción de la alimentación de energía hacia la hélice se debe a la desconexión de las escobillas con los portaescobillas y mas no a la distancia entre los portaescobillas y el disco con los anillos de cobre.

Para solucionar la falla de alimentación de energía hacia la hélice, se suelda el extremo de las escobillas al tope del portaescobillas para garantizar que las escobillas no se desconecten durante el movimiento giratorio del sistema de rotulación RGB. (Ver figura 116)

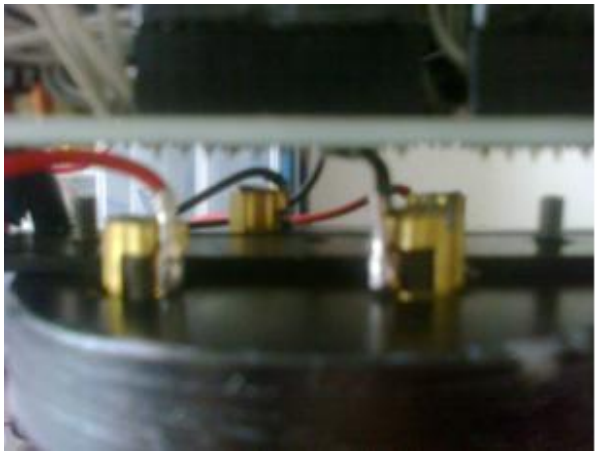


Figura. 116 Escobillas y portaescobillas soldados

5.4.3 Corrección del recalentamiento del motor AC

Para solucionar el problema del recalentamiento del motor AC, se cambian los resortes originales de las escobillas por unos más suaves que no ejercen un exceso de fricción en los anillos de cobre y se coloca un ventilador para contribuir a la circulación de aire en el motor AC. (Ver figura 117)

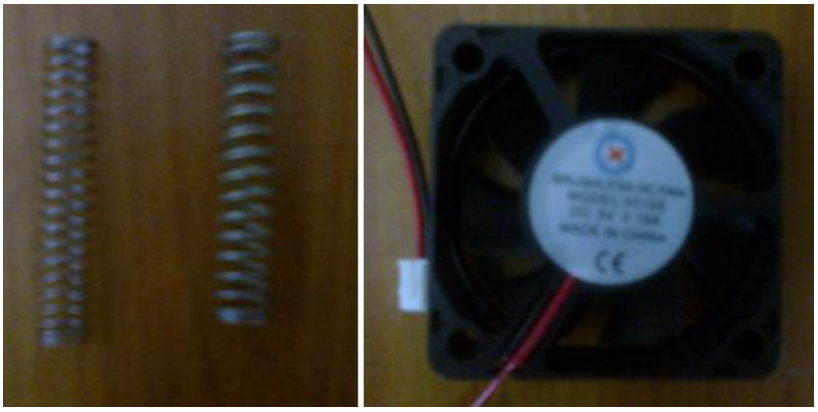


Figura. 117 Resortes de las escobillas

5.4.4 Corrección de la visualización de los mensajes en el sistema de rotulación RGB

Los factores mecánicos que provocan el recalentamiento del motor AC, no permiten realizar un cálculo exacto de los valores para el tiempo de encendido y apagado de los LED's RGB y de la velocidad de giro del motor AC, por lo que se realiza una prueba de fallo-corrección, que consiste en visualizar el mensaje que se proyecta a través de los LED's RGB, al fijar el tiempo de encendido y apagado de los LED's RGB en 100[us] y variar la velocidad de giro del motor AC. (Ver tabla 147)

Tiempo encendido/apagado LED's RGB[us]	Resistencia [kΩ]	Ángulo de disparo	% de onda senoidal de disparo del triac	Resultado
100	56	64.65°	17.95%	Se alargan y sobre montan los mensajes.
100	68	68.68°	19.07%	Se alargan y giran sin control los mensajes.
100	82	72.07°	20.01%	Se muestra tres veces el mismo mensaje sin ninguna clase de distorsión visual.
100	91	73.74°	20.48%	Se muestra el mensaje sin ninguna clase de distorsión visual y su tamaño es reducido.
100	100	75.14°	20.87%	Se detiene el motor, por tanto no se visualizan los mensajes.

Tabla. 147 Prueba de velocidad del motor AC

Según los resultados que se obtienen de los valores de resistencia, se comprueba que con una resistencia de 91[kΩ] y 82[kΩ] la visualización del mensaje que se proyecta a través de los LED's RGB es óptima, debido a que el mensaje no presenta ninguna

distorsión en su tamaño y se posiciona adecuadamente alrededor del sistema de rotulación RGB.

5.5 PUESTA EN MARCHA DEL SISTEMA DE ROTULACIÓN RGB

Una vez que se corrigen todos los errores de funcionamiento, se pone en marcha el sistema de rotulación RGB y se comprueba que las soluciones descritas resuelvan satisfactoriamente los inconvenientes.

Para finalizar se protege al sistema de rotulación RGB con una cubierta plástica transparente, para que ningún objeto del exterior golpee la hélice durante su movimiento giratorio y dañe el aparato o produzca graves lesiones en el usuario.

5.6 ANÁLISIS DE RESULTADOS

Para el análisis de resultados del prototipo que se diseña, se toma en cuenta la información de la base teórica y los antecedentes del fenómeno de la persistencia de la visión que se emplean para desarrollar el sistema de rotulación RGB.

- **Visualización del mensaje**

Para la visualización del mensaje en el sistema de rotulación RGB se establecen tres parámetros fundamentales:

Velocidad en la lectura del programa del microcontrolador: La programación se realiza en forma estructurada, para que la lectura de los datos este en orden y evitar el empleo de sentencias de condición que retrasen la lectura del programa y el envío de datos hacia los puertos.

Tiempos de retardo: Se relacionan con la distorsión del mensaje que se proyecta a través de los led's RGB. Emplear tiempos de retardo extensos, provoca que los caracteres del mensaje se deformen y provoquen una mala visualización del texto o imagen.

Velocidad del motor AC: Permite posicionar al mensaje alrededor del sistema de rotulación RGB. Si se emplea una velocidad alta el mensaje no permanece estable y dificulta su visualización.

Con los tres parámetros que se controlan, se logra como resultado que el espectador visualice el mensaje sin distorsión en diferentes posiciones alrededor de la hélice. Así se comprueba la teoría "Fenómeno de la persistencia de la visión" de Joseph Plateau, que demuestra como una imagen permanece en la retina humana una décima de segundo antes de desaparecer por completo, esto permite ver la realidad como una secuencia de imágenes ininterrumpidas y llega a producir sensación de movimiento. (Ver figura 118)

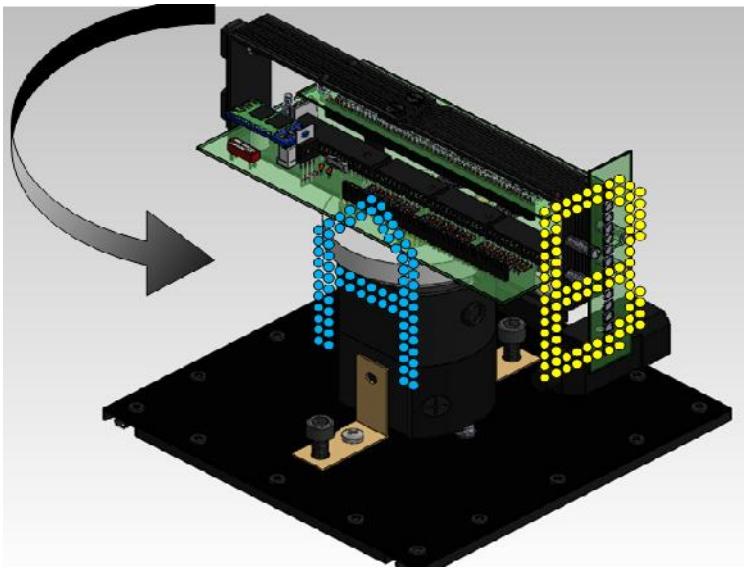


Figura. 118 Sistema de rotulación RGB

CAPÍTULO 6

CONCLUSIONES Y RECOMENDACIONES

6.1 Conclusiones

- Realizar el diseño e implementación de un sistema de rotulación RGB, que se basa en la técnica de la persistencia de la visión, ofrece una solución óptima del alcance visual, que se aplica en el mercado de la publicidad, a partir de recursos mecánicos y electrónicos actualmente disponibles en el mercado ecuatoriano.

- Analizar previamente las condiciones óptimas y necesarias para la ubicación de los dispositivos electrónicos y de los elementos mecánicos en la creación del sistema de rotulación RGB, es necesario para que no se produzcan cambios en sus diseños al momento de su acople.

- Emplear en el diseño mecánico del sistema de rotulación RGB portaescobillas y anillos de cobre, es una solución factible para transferir energía desde un punto fijo a un elemento que está en movimiento porque no representa un obstáculo en el movimiento giratorio del sistema, debido a que evita colocar cables o baterías en la hélice.

- Emplear comunicación inalámbrica en el sistema de rotulación RGB, facilita la transmisión de datos desde la PC hacia el microcontrolador, porque que se elimina la dependencia de cables y conectores, además de que se justifica la selección de los módulos HC05, que emplean comunicación Bluetooth-SSP y se diseñan para trabajar en dispositivos de bajo consumo de energía como son los microcontroladores.

- Elegir un microcontrolador PIC18F452 para el control del sistema de rotulación RGB, facilita el almacenamiento de datos por su capacidad de memoria RAM en comparación a otras gamas de microcontroladores PIC, además, favorece el bajo consumo de energía y configuraciones de Reset que posee.

- Descartar sentencias condicionales en la programación del microcontrolador del sistema de rotulación RGB, agilita la lectura del programa y da la opción de controlar los tiempos de retardo que permiten visualizar el mensaje sin ninguna deformación.

- Emplear el lenguaje de programación JAVA para el desarrollo de la interfaz gráfica en el sistema de rotulación RGB, permite crear un archivo ejecutable que puede abrirse en cualquier PC, sin la necesidad de instalar entornos de desarrollo de JAVA.

- Usar la interfaz gráfica en el sistema de rotulación RGB, proporciona al usuario un entorno visual sencillo de manejar, que permite diseñar texto o imágenes que se presentan a través de los LED's RGB en el sistema de rotulación.

- Ofrecer la opción de guardar el código del mensaje o imagen que se crea en la interfaz gráfica del sistema de rotulación RGB en un archivo con extensión .txt, da la posibilidad al usuario de reutilizar o editar mensajes que se crean sin la necesidad de volver a diseñarlos.

- Controlar los tiempos de retardo y velocidad de giro del motor del sistema de rotulación RGB, permite una correcta visualización del mensaje y evita distorsiones en el texto o imagen que se proyecta a través de los LED's RGB.

6.2 Recomendaciones

- Para la elección del dispositivo de control de cualquier sistema, se recomienda tomar en cuenta la cantidad de puertos de entrada y salida, la capacidad de memorias, los tipos de comunicación y el tamaño del dispositivo, que satisfaga las necesidades de la aplicación a desarrollar.

- Si se trabaja en alguna aplicación con transistores en lógica negativa se recomienda crear un circuito electrónico que garantice un cero lógico, debido a la presencia de señales basura que pueden provocar activaciones no deseadas.

- Se recomienda no situar el aparato a más de 10 metros de la PC, para garantizar una transmisión de datos correcta.

- Se recomienda mantener limpios y en buen estado los ventiladores del sistema de rotulación RGB, para la refrigeración del motor y evitar su daño por recalentamiento. Esta prevención alarga la vida útil del aparato.

- Se recomienda colocar el dispositivo de rotulación RGB en un soporte estable para evitar efectos de balanceo y vibración.

- Para la correcta visualización del texto o imagen que se proyecta a través de los LED's RGB, se recomienda esperar un tiempo aproximado de 30 segundos hasta que se estabilice la velocidad del motor AC del sistema de rotulación RGB.

- Se recomienda proteger al sistema de rotulación RGB con una cubierta transparente, para evitar que un objeto del exterior impacte la hélice durante su movimiento giratorio y dañe el aparato o produzca graves lesiones en el usuario.

- Para el correcto manejo y funcionamiento del sistema de rotulación RGB, se recomienda seguir las instrucciones que se establecen en el manual de usuario.

BIBLIOGRAFÍA

(s.f.). En G. Eduardo, *Compilador C CCS y simulador Proteus para microcontroladores PIC* (pág. 83). Alfaomega Grupo Editor S.A de C.V, Mexico.

(s.f.). En T. L.Floyd, *Dispositivos Electronicos* (págs. 182-185 y 202).

(s.f.). En M. H. Rashid, *Electrónica de Potencia circuitos, dispositivos y aplicaciones* (págs. 106-110, 191-198). Prentice hall hispanoamerica, S.A.

(s.f.). En R. Cadenhead, *Programacion JAVA 7* (págs. 84-95, 152-167). Anaya Multimedia.

(s.f.). En P. D. Deitel, *Como programa JAVA* (págs. 71-95). DEITEL.

Angulo de disparo del Triac. (s.f.). Obtenido de http://vargasdaniel27.wordpress.com/2008/08/15/disparo-del-triac-mediante-diac/

Cinematografia. (s.f.). Obtenido de http://mistusynos.blogspot.com/2007/04/peliculas-movie-cintas-cinematograficas.html

Combinacion de Colores. (s.f.). Obtenido de http://proyectoplasticaprimarialossimpsons.blogspot.com/2013/04/vision-de-los-colores-beatriz-perez.html

Diametro brocas y machuelos. (s.f.). Obtenido de http://corteymedicion.com/site/parametros/machos-diametrohueco.php

Dimmer por triac y diac. (s.f.). Obtenido de http://electronicavm.files.wordpress.com/2011/05/diacs-y-triacs.pdf

Efecto estroboscópico. (s.f.). Obtenido de http://www.librosmaravillosos.com/comofunciona/capitulo08.html

Ejercicios en Netbeans. (s.f.). Obtenido de http://es.scribd.com/doc/153906976/Manual-Netbeans-Guia-de-Ejercicios

FotoLog. (s.f.). Obtenido de http://www.fotolog.com/cineyfoto/55798200/

Frases flotando. (s.f.). Obtenido de http://www.iesmajuelo.com/index.php?f=departamentos&id=354&deptno=Inform%E1tica

Integrado DM74154. (s.f.). Obtenido de http://pdf.datasheetcatalog.net/datasheets/120/236595_DS.pdf

La edad del pixel. (s.f.). Obtenido de http://laedaddelpixel.blogspot.com/2013/02/la-capsula-del-tiempo-aquellos.html

LED. (s.f.). Obtenido de http://www.neoteo.com/electronica-basica-diodos-emisores-de-luz-led

LED SMD. (s.f.). Obtenido de http://spanish.alibaba.com/product-gs/plcc6-5050-series-3chips-white-surface-high-brightnesss-white-type-smd-led-308348283.html

Letreros de informacion comercial. (s.f.). Obtenido de http://www.screens.ru/es/2003/4.html

Letreros de informacion deportiva. (s.f.). Obtenido de http://cajamarca-cajamarca.nexolocal.com.pe/c690-compra-venta-p2

Letreros de informacion industrial. (s.f.). Obtenido de http://www.rotuloselectronicos.net/carteles-luminosos-y-pantallas-informativas-multilinea.html

Letreros de informacion universitaria. (s.f.). Obtenido de http://cuernavaca.olx.com.mx/anuncio-programable-led-32x96-display-programable-letrero-programable-led-iid-473291256

Letreros de informacion vial. (s.f.). Obtenido de http://lumtec.com.mx/Pantallas_electronicas_LED/senalizacion_vial.html

Matriz LED. (s.f.). Obtenido de http://www.seeedstudio.com/wish/smaller-rgb-led-matrix-p2076

Medidas motor AC. (s.f.). Obtenido de http://www.lionball.net/Sewing-Machine-Motor/Sewing-Machine-Motor---HF.html

Mensajes bicicleta. (s.f.). Obtenido de http://agitpov.wordpress.com/tag/persistencia-de-vision/

Microchip. (s.f.). Obtenido de http://ww1.microchip.com/downloads/en/DeviceDoc/39576c.pdf

Microcontroladores PIC. (s.f.). Obtenido de http://bibing.us.es/proyectos/abreproy/11301/fichero/Memoria%252FCap%C3%ADtulo+3.pdf

Modulo HC05. (s.f.). Obtenido de ftp://imall.iteadstudio.com/IM120417010_BT_Shield_v2.2/DS_BluetoothHC05.pdf

Netbeans. (s.f.). Obtenido de https://netbeans.org/about/history_pt_BR.html

Olympia OL 3000. (s.f.). Obtenido de http://todousa.co/olympia-ol-3000-infoglobe-digital-caller-id-real-time-clock

Origenes del cine. (s.f.). Obtenido de http://tallerelmate.wordpress.com/2010/12/01/modelo-de-un-taller-origenes-del-cine-i/thaum/

Partes y funcionamiento del ojo humano. (s.f.). Obtenido de http://www.apanovi.org.ar/iusaludparyfun.html

Posicionamiento del mensaje. (s.f.). Obtenido de http://www.iesmajuelo.com/index.php?f=departamentos&id=354&deptno=Inform%E1tica

POV con LED's. (s.f.). Obtenido de http://www.taringa.net/posts/hazlo-tu-mismo/10701574/POV-Persistence-Of-Vision-Con-Leds.html

Praxinoscopio. (s.f.). Obtenido de http://13gatosnegros.blogspot.com/2008/03/praxinoscopio.html

Tensiones y corrientes LED. (s.f.). Obtenido de http://www.sharatronica.com/formulas_para_led.html

Transistores 2N3906. (s.f.). Obtenido de http://www.ece.rice.edu/~jdw/data_sheets/2N3906.pdf

Visualizador Led Tipo Rotativo. (s.f.). Obtenido de http://www.scribd.com/doc/63731909/Visualizador-Led-Tipo-Rotativo

Zootropo. (s.f.). Obtenido de http://www.alternatilla.com/2009/exposiciones/13-zootropos/

FECHA DE ENTREGA: 13 DE NOVIEMBRE DE 2013

En la ciudad de Sangolquí, firman en constancia de la entrega del presente Proyecto de Grado titulado " **Diseño e implementación de un sistema de rotulación RGB basado en la técnica de persistencia de la visión**", en calidad de Autores al Sr. Edwin Gonzalo Clavijo Villavicencio y al Sr. Ricardo David Pérez Carrillo, estudiantes de la carrera de Ingeniería Electrónica en Automatización y Control, y recibe por parte del Departamento de Eléctrica y Electrónica el Director de Carrera de Automatización y Control, el Señor Ing. Luis Orozco B. Msc.

Edwin Gonzalo Clavijo Villavicencio
171999627-2

Ricardo David Pérez Carrillo
171921458-5

Ing. Luis Orozco B. Msc.
DIRECTOR DE LA CARRERA DE INGENIERÍA ELECTRÓNICA EN AUTOMATIZACIÓN Y CONTROL