UNIVERSIDAD NACIONAL DE INGENIERÍA

Facultad de Electrotecnia y Computación
Recinto Universitario Simón Bolívar

Trabajo Monográfico

Diseño e implementación de un Controlador Electrónico para Automatizar el suministro de tratamiento para metal SUPERKOTE 2000 provisto por la empresa Distribuidora de Productos y Servicios S.A. (DPS).

Para optar al título de: Ingeniero Electrónico

Autores:

Br. Carlos Eduardo Díaz Moreira	Carnet # 2009-29761
Br. Jorge Joel Mora Jara	Carnet # 2006-23868
Br. Christian Selim Soza Zeledón	Carnet # 2009-29918

Tutor:
Msc. Dora Inés Reyes Chávez

Managua-Nicaragua, noviembre del 2021.

El único modo de hacer un gran trabajo es amar lo que haces.

Dedicatoria

A DIOS por la vida, conocimiento, sabiduría y perseverancia para lograr cumplir esta etapa.

A mis PADRES por su apoyo a lo largo de mis estudios y por enseñarme a creer y confiar en mí.

A mis HERMANAS por su amor incondicional.

A mis FAMILIARES, AMIGAS y AMIGOS que me apoyaron y motivaron a seguir adelante.

A mis COMPAÑEROS de tesis por haber aceptado este desafío conmigo.

Carlos Eduardo Díaz Moreira

Dedicatoria

A Dios, por su infinita bondad y amor, por permitirme culminar mis estudios profesionales, gozar de salud y darme lo necesario para seguir adelante día a día y alcanzar mis objetivos.

A mis padres, por su amor y apoyo incondicional, sus consejos y sus constantes palabras de motivación las que me formaron como una persona de bien, especialmente gracias a mi Padre, que, aunque no estará presente para verme culminar mis estudios, fue y será ese ejemplo de perseverancia y constancia que admiraré siempre.

A mi alma mater, por abrirme las puertas desde el inicio, mostrarme el mundo y dotarme de conocimientos y valores para aportar a la consecución de una sociedad mejor.

Jorge Joel Mora Lara

Dedicatoria

A Dios en primer lugar por darme fortaleza para seguir adelante a pesar de las adversidades en el camino, ya que en cada momento estuvo a mi lado.

A mi abuelita Mariana Palma Palaviccini la mejor abuela del mundo y la que me enseñó el valor y fruto del esfuerzo; por ser el pilar y la piedra angular de mi familia.

A mis padres Claudia María Zeledón Palma y Harvy Horacio Soza Castro por ser ejemplo de valores y la humildad.

A mis amigos, familiares por ser incondicionales y que siempre un saludo o muestra de afecto, hace la diferencia.

A mi alma Mater, la Unión Nacional de Estudiantes de Nicaragua y la Jefatura del Proyecto UNI, por ser ejes fundamentales en mi desarrollo personal y profesional.

Christian Selim Soza Zeledón

Toda persona deja una enseñanza, toda enseñanza deja una experiencia y toda experiencia deja una huella.

Agradecimiento

A DIOS, por estar presente en mi vida y permitirme llegar a completar esta etapa.

A “DPS, S.A”, por la disposición brindada, la confianza puesta en nosotros y por todo el apoyo posible para la culminación del presente trabajo.

A todas las personas que nos colaboraron incondicionalmente durante el desarrollo de este trabajo.

A nuestros profesores que a lo largo de la carrera nos compartieron sus conocimientos y enseñanzas.

Carlos Eduardo Díaz Moreira

Agradecimiento

Mi agradecimiento eterno a mí universidad, Universidad Nacional de Ingeniería, después de años de esfuerzo, sacrificios, dedicación y grandes alegrías llegó el día en que miraría hacia atrás el camino recorrido por tus pasillos, aulas y me detendría para agradecerte mi alma mater.

Agradezco mucho por la ayuda de mis maestros, mis compañeros, y a la universidad en general por todos los cuantiosos conocimientos que me ha otorgado.

Jorge Joel Mora Lara

Agradecimiento

Agradezco a Dios, padre y creador de todo; por darme la oportunidad de finalizar mis estudios y por darme las fuerzas de seguir adelante.

Agradezco principalmente a mi abuelita, que se empeñó en que iniciara mis estudios en la Universidad Nacional de Ingeniería

Agradezco a Mi Alma Mater por ser mi segundo hogar, por la gratuidad y calidad de la educación, a los docentes que la componen y a las personas que desempeñan su labor en cada área.

Christian Selim Soza Zeledón

Índice

Índice

Figuras

Tablas

Introducción

La Automatización exige estándares para optimizar procesos, es por ello que se desarrollaran herramientas que sean garantes de generar mayor fiabilidad en los sistemas que se implementan.

Un sistema automatizado ajusta sus operaciones en respuesta a cambios en las condiciones externas e internas desafiando a las microempresas del país a implementar nuevos mecanismos para hacer más eficiente sus procesos productivos para así poder competir y estar a la altura de las necesidades del mercado.

La automatización es fundamental para gestionar, cambiar y adaptar no solo su infraestructura, sino también la manera en que su empresa opera en todos sus procesos, de esta manera realizan sus tareas con mayor rapidez.

Es por ello que desde la carrera de Ingeniería Electrónica se logra diseñar e implementar controladores electrónicos que permitan a la microempresa una solución de bajo costo y de calidad que sea capaz de automatizar procesos industriales; en este caso de tratamiento para metales como SUPERKOTE 2000.

Los tratamientos para metales como SUPERKOTE 2000 extienden la duración del motor, aumentan la fuerza y compresión, disminuyen el desgaste durante el encendido en frío, reducen el ruido, temperatura y más del 12% del consumo del combustible.

Por tanto, se diseñará e implementará un controlador electrónico de lazo cerrado capaz de automatizar el suministro de los tratamientos para metales en maquinarias cumpliendo con las especificaciones que el cliente determine y al mismo tiempo les permita realizar el llenado de envases comerciales de manera eficiente.

Al realizar el diseño del Control Electrónico se deben realizar varias pruebas con conocimientos ingenieriles, a la vez un análisis técnico-económico que nos llevará a concretar el proyecto, o bien a no ejecutarse. El desarrollo del controlador se realizará con elementos económicos que permitan a la microempresa una solución de bajo costo y de calidad.

En nuestro caso nos enfocaremos en una parte muy importante de la Ingeniería Electrónico como es el caso de la Automatización.

El documento se divide en V Capítulos: El Capítulo I, explica conceptos generales; Capítulo II, se definen los elementos principales de Controlador Electrónico y Automatización; Capítulo III, hace referencia al Diseño Metodológico; el Capítulo IV contiene un resumen de resultado del Diseño e Implementación del Controlador Electrónico para Automatizar cumpliendo así con los objetivos principales del tema monográfico; quedando el Capítulo V, para las conclusiones finales del estudio.

Antecedentes

La Automatización se remonta a muchos años, con el tiempo se han inventado nuevos y mejores dispositivos.

Desde la Universidad Nacional de Ingeniería, Nicaragua, se han realizado propuestas de Diseño de un sistema de automatización para el llenado de un tanque de agua por bomba con la ayuda de sensores de nivel haciendo uso de sensores de proximidad capacitivos con detectores de nivel para controlar el funcionamiento de bombas de llenado.[1]

Lograron controlar los niveles máximos y mínimos de productos líquidos, polvos o granos en un recipiente o una tolva, es relevante en la búsqueda de realizar el control de la dosis exacta de aditivo líquido.

En la Universidad San Buenaventura de Bogotá Colombia presentan proyecto para optar al título de Ingeniero Electrónico como un aporte tecnológico a la empresa Fuller Pinto donde se resuelven los problemas relacionados con las dosificaciones de fragancias a productos químicos y la exposición de los trabajadores a estas sustancias reduciendo con ello los riesgos ocupacionales. [2]

Desde la Universidad Austral de Chile presentan tesis para optar al título de Ingeniero Electrónico, Estudio y Diseño de un sistema dosificador de pigmentos, logrando utilizar elementos disponibles en el mercado local.[3]

Para el caso de la automatización industrial se destacan:

1968: La exitosa historia del PLC, empezó con el control industrial modular Dick Morley.
1997: La tecnología de la automatización consiste en un control descentralizado e inteligente con componentes de control.
2004: La funcionalidad del PLC fue descubierta en un chip.

[1] Mejía Narváez, Espinoza Martínez. 2016. Universidad Nacional de Ingeniería, Nicaragua.
[2] Beltrán Sánchez, Cepeda Sánchez, 2008. Universidad San Buenaventura, Bogotá Colombia,
[3] Heriberto Mario. Universidad Austral, Chile.

Justificación

El crecimiento en la economía de nuestro país ha propiciado la creación de micro empresas, una de estas es la microempresa "Distribuidora de Productos y Servicios" (DPS) situada en Managua, es proveedora de tratamiento para metales especialmente SUPERKOTE 2000 recomendado para usarse en sistemas de lubricación que requieren protección especial en presencia de presiones extremas, altas temperaturas friccionales y desgaste excesivo.

La mayoría de las industrias buscan automatizar debido a los riesgos que implica algunos procesos para los seres humanos y otras por el beneficio de la rapidez y aumentar su productividad dado que simplifica las operaciones.

Para el avance de sus procesos productivos en la empresa DPS se deben implementar métodos tecnológicos que ayuden a acelerar todo el proceso y lo convierta más productivo; esto se lograría por medio de un sistema electrónico capaz de suministrar el tratamiento de manera automática, el cual disminuirá los tiempos de trabajo, mano de obra y aumentara la exactitud de la producción

La Extensión Universitaria es, además, una necesidad misma de la universidad, es por ello que a través de la vinculación universidad-empresa se fortalecen los conocimiento y saberes de la comunidad universitaria que se enriquece con la práctica cotidiana que vive la sociedad, mejora el aprendizaje y desarrolla el pensamiento creativo e innovador de las y los estudiantes en su formación profesional, elementos que agrega valor al rol de la universidad en la sociedad.

La empresa DPS ha solicitado el diseño y la implementación de un controlador electrónico capaz de acelerar el proceso de llenado de las botellas, actualmente las presentaciones son de 4 y 8 onzas, además de su mercado en crecimiento que es la venta y suministro del producto a granel a maquinaria pesada de carácter agrícola y/o transporte.

Objetivos

Objetivos General

Diseñar e implementar un controlador electrónico para automatizar el suministro del tratamiento para metales SUPERKOTE 2000.

Objetivos Específicos

- ✓ Analizar los términos de referencia para automatizar el suministro del tratamiento para metales SUPERKOTE 2000.
- ✓ Diseñar el controlador electrónico con autonomía energética, independiente del suministro eléctrico comercial.
- ✓ Realizar al prototipo de controlador pruebas de rendimiento y calibración para que dosifique las cantidades exactas del tratamiento para metales propuestas por el usuario.
- ✓ Implementar el controlador electrónico garantizando el desempeño requerido por la empresa.

Capítulo 1

Fundamentos Teóricos

1.1. Introducción

El control automático es una rama de la ingeniería que se ocupa del control, automatización de procesos en un estado determinado; definiéndose como la acción o efecto de automatizar o hacer automático algo, el cual proviene del término "automática" que es la ciencia y técnica de la automatización, que agrupa el conjunto de las disciplinas teóricas y tecnológicas que intervienen en la concepción, la construcción y el empleo de los sistemas automáticos.

Podemos referirnos a la automatización como el conjunto de elementos electrónicos y mecánicos para realizar un proceso con intervención parcial o nula del ser humano.

El concepto suele utilizarse en el ámbito de la industria como referencia el sistema que permite que una máquina desarrolle ciertos procesos o realice tareas sin intervención del ser humano. La automatización permite ahorrar tiempo y en su defecto dinero.

Los orígenes de la automatización se encuentran en la prehistoria, con el desarrollo de las máquinas que minimizaban la fuerza que debían hacer las personas. La energía animal o humana, con el tiempo, comenzó a reemplazarse por energías renovables (como la energía eólica o la energía hidráulica)[4]

En la actualidad en las modernas fábricas, instalaciones industriales pequeñas y de gran escala, se hace necesario disponer de sistemas de control o de mando, que permitan mejorar y optimizar una gran cantidad de procesos, en donde la sola presencia del hombre es insuficiente para gobernarlos. La industria espacial y de la aviación, petroquímica, papelera, textil, del cemento, etc. son algunos ejemplos de lugares en donde se necesitan sistemas de control, cuya complejidad ha traído como consecuencia el desarrollo de técnicas dirigidas a su proyecto y construcción.

[4] J.P.P y M. Merino https://definicion.de, 2016 y actualizado 2017. https://definicion.de/automazacion/

1.2. Definiciones en la teoría de Control Automático[5]

Estas definiciones están basadas, en parte, en las propuestas de normas de la IEEE, obedecen a la necesidad de emplearlas en los temas de introducción general.

Planta: se designará como planta a cualquier objeto físico que pueda ser controlado, puede ser un equipo, quizás simplemente un juego de piezas de una máquina funcionando juntas, cuyo objetivo es realizar una operación determinada.

Proceso: se definirá como una operación o conjuntos de pasos con una secuencia determinada, que producen una serie de cambios graduales que llevan de un estado a otro, y que tienden a un determinado resultado final. Se denominará proceso a cualquier operación que se vaya a controlar. Ejemplos de procesos son: químicos, económicos, biológicos, etc.

Sistema: se define a un sistema como un arreglo, conjunto o combinación de cosas conectadas o relacionadas de manera que constituyen un todo. De forma científica podemos definirlo como un arreglo de componentes físicos conectados o relacionados de tal manera que formen una unidad completa o que puedan actuar como tal; en otras palabras: Un sistema es una combinación de componentes que actúan conjuntamente, con un determinado objetivo a cumplir. Cómo puede observarse el término sistema no está aplicado únicamente a objetivos físicos, el concepto de sistema puede ser aplicado a fenómenos abstractos y dinámicos como por ejemplo la economía. Por tanto, cuando se hable de sistemas implicará referirse a fenómenos físicos, biológicos, económicos, sociológicos, etc.

La planta junto con el proceso, conforman un sistema.

Control: esta palabra se usa para designar regulación, gobierno, dirección o comando.

Sistema de control: es un arreglo de componentes físicos conectados de tal manera que el arreglo pueda comandar, dirigir o regular, asimismo o a otro sistema. Estos sistemas comandan dirigen o controlan dinámicamente.

Entrada de un sistema: Es una variable del sistema elegida de tal manera que se la utiliza como excitación del mismo.

Salida de un sistema: Es una variable del sistema elegida de tal modo que se la utiliza para analizar los efectos que produjo una excitación en la entrada del mismo.

Entrada de un sistema de control: Es una variable del sistema controlado que se elige de modo tal que mediante su manipulación se logra que el sistema cumpla un objetivo determinado. Las variables de entrada, son variables que ingresan al sistema y no dependen de ninguna otra variable interna del mismo.

[5] Ogata K. Sensor de Ingeniería de Control

Salida de un sistema de control: Es una variable del sistema controlado que se elige de modo tal que mediante su estudio se analiza si el sistema cumple o no con los objetivos propuestos. Se verá más adelante que en los sistemas realimentados esta señal de salida contribuye a realizar el control propuesto.

Realimentación: es una propiedad de los sistemas que permiten que la salida del sistema o cualquier variable del mismo sea comparada con la entrada al sistema o con cualquier componente del sistema, de tal manera que pueda establecerse la acción de control apropiada entre la entrada y la salida.

Perturbaciones: es una señal que tiende a afectar adversamente el valor de la salida de un sistema. Si la perturbación se genera dentro del sistema se la denomina interna, mientras que una perturbación externa se genera fuera del sistema. Las perturbaciones actúan sobre un sistema modificando, su funcionamiento por lo que su presencia implica la necesidad de control. Normalmente las perturbaciones actúan sobre un sistema aleatoriamente.

Control de Realimentación: es una operación que, en presencia de perturbaciones, tiende a reducir las diferencias entre la salida y la entrada del sistema y lo hace sobre la base de esta diferencia, la cual se denomina señal de error. Cuando se utiliza control de realimentación se considera perturbación a aquellas que tienen carácter aleatorio (no previsible), porque las perturbaciones que pueden ser predichas siempre se puede incluir una compensación dentro del sistema de modo que sea innecesario el control.

Sistema de control realimentado: es aquel que tiende, a mantener una relación preestablecida entre la salida y la entrada de referencia, comparando ambas y utilizando la diferencia como variable de control.

Sensor: Es un dispositivo que convierte la variable de salida en otra variable manejable, como un desplazamiento, una presión o un voltaje, que pueda usar, separar comparar la salida con la señal de entrada de referencia. Este elemento está en la trayectoria de realimentación del sistema en lazo cerrado. El punto de ajuste del controlador debe convertir-se en una entrada de referencia con las mismas unidades que la señal de realimentación del sensor o del elemento de medición.

Sustancias Lubricantes6: Los aceites lubricantes minerales proceden en su mayoría de la destilación de petróleos brutos o crudos, como así se les denomina por ser extraídos de las profundidades de la tierra.
Los lubricantes minerales se pueden presentar en estado sólido, como el molibdeno, el grafito, el talco, las vaselinas, las ceras minerales y un sinfín de metales.

La lubricación consiste propiamente en la aplicación de una película de sustancia grasa, oleosa o de análogas condiciones con objeto de disminuir el rozamiento entre

[6] F.M. Pérez. Sustancias Lubricantes. La Tribología Ciencia y Técnica para el mantenimiento. México, Limusa 2002.

dos superficies, a fin de evitar en lo posible el contacto metálico entre ellas cuando una se mueve sobre la otra.

Propiedades de los lubricantes[7]

Viscosidad: Llámese viscosidad a aquella propiedad de un fluido a la cual debe su resistencia a fluir, Fundamentalmente es la fuerza tangencial o esfuerzo de fricción para rozar una capa de fluido de 1cm2 y de un centímetro de espesor, a una velocidad de 1cm por segundo.

Bomba: Es una máquina que absorbe energía mecánica que puede provenir de un motor eléctrico, térmico, etc., y la transforma en energía que la transfiere a un fluido como energía hidráulica la cual permite que el fluido pueda ser transportado de un lugar a otro, a un mismo nivel y/o a diferentes niveles y/o a diferentes velocidades.
Se pueden considerar dos grandes grupos: Dinámicas (Centrífugas, Periféricas y Especiales) y de Desplazamiento Positivo (Reciprocantes y Rotatoria)

Bombas Centrífugas: Son aquellas en que el fluido ingresa a ésta por el eje y sale siguiendo una trayectoria periférica por la tangente.

1.3. Generalidades sobre Sistemas de Control

El control automático ha jugado un papel vital en el avance de la ingeniería y la ciencia. Como los avances en la teoría y práctica del control automático brindan los medios para lograr el funcionamiento óptimo de sistemas dinámicos, mejorar la calidad y abaratar los costos de producción, liberar de la complejidad de muchas rutinas de tareas manuales respectivas, etc.; la mayoría de los ingenieros tienen contacto con los sistemas de control, aun cuando únicamente los usen, sin profundizar en su teoría.

Este tipo de sistema que se caracteriza por la presencia de una serie de elementos que permiten influir en el funcionamiento del sistema. La finalidad de un sistema de control es conseguir mediante la manipulación de las variables de control un dominio sobre las variables de salida de modo que estas alcancen unos valores prefijados.

Un sistema de control automático es una interconexión de elementos que forman una configuración denominada sistema, de tal manera que el arreglo resultante es capaz de controlarse por sí mismo, este debe ser capaz de conseguir su objetivo cumpliendo los siguientes requisitos:

1. Garantizar la estabilidad y particularmente ser robusto frente a perturbaciones y errores en los modelos.

[7] F.A. García, Características, uso y aplicación de lubricantes. Características, uso y aplicación de lubricantes. La Habana 1983.

2. Ser tan eficiente como sea posible, según un criterio preestablecido, normalmente consiste en que la acción de control sobre las variables de entrada sea realizable, evitando comportamiento bruscos e irreales.
3. De fácil implementación y cómodo de operar en tiempo real con ayuda de un ordenador.

1.4. Tipos de Computadoras

1.4.1 Computadora compacta Raspberry Pi[8]

La Raspberry Pi es una computadora de bajo costo y con un tamaño compacto, del porte de una tarjeta de crédito, puede ser conectada a un monitor de computador o un TV, y usarse con un mouse y teclado estándar. Es un pequeño computador que correo un sistema operativo Linux capaz de permitirle a las personas de todas las edades explorar la computación y aprender a programar lenguajes como Scratch y Python. Es capaz de hacer la mayoría de las tareas típicas de un computador de escritorio, desde navegar en internet, reproducir videos en alta resolución, manipular documentos de ofimática, hasta reproducir juegos.
Además, la Raspberry Pi tiene la habilidad de interactuar con el mundo exterior, puede ser usada en una amplia variedad de proyectos digitales, desde reproductores de música y video, detectores de padres, estaciones meteorológicas hasta cajas de aves con cámaras infrarrojas. Queremos que veas que la Raspberry Pi puede ser usada por niños y adultos por todas partes del mundo, para aprender a programar y entender cómo funcionan las computadoras. En la Figura No. 1 se muestra el logo oficial de Raspberry.

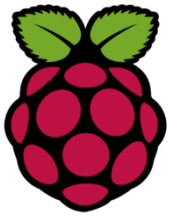


Figura 1. Logo Oficial de Raspberry

1.4.2 Linux[9]

Linux® es un sistema operativo (SO) open source. En 1991, Linus Torvalds lo diseñó y creó a modo de pasatiempo. Mientras estaba en la universidad, Linus intentó crear una versión open source, alternativa y gratuita del sistema operativo MINIX, que a su vez se basaba en los principios y el diseño de Unix. Ese pasatiempo logró convertirse en el sistema operativo con la mayor base de usuarios, el más usado en los servidores

[8] https://raspberrypi.cl/que-es-raspberry/
[9] https://www.redhat.com/es/topics/linux

de Internet disponibles públicamente y en el único utilizado en las 500 supercomputadoras más rápidas.

Quizá lo mejor de Linux es que es open source. Linux se lanza en virtud de la Licencia de uso público GNU (GPL), lo cual significa que todos pueden ejecutar, estudiar, compartir y modificar el software. El código modificado también se puede redistribuir, e incluso vender, pero todo esto se debe hacer con la misma licencia. Esta es una de las principales diferencias con los sistemas operativos tradicionales (por ejemplo, Unix y Windows) que son propietarios, están bloqueados, se distribuyen tal como están y no se pueden modificar. En la figura No. 2 se muestra la mascota de Linux.

Figura 2. Mascota de Linux

1.4.3 Raspberry Pi OS[10]

Raspbian es un sistema operativo gratuito basado en Debian optimizado para el hardware Raspberry Pi. Un sistema operativo es el conjunto de programas y utilidades básicos que hacen que su Raspberry Pi funcione. Sin embargo, Raspbian ofrece más que un sistema operativo puro: viene con más de 35,000 paquetes, software pre compilado incluido en un formato agradable para una fácil instalación en su Raspberry Pi.

La compilación inicial de más de 35.000 paquetes Raspbian, optimizados para el mejor rendimiento en Raspberry Pi, se completó en junio de 2012. Sin embargo, Raspbian todavía está en desarrollo activo con énfasis en mejorar la estabilidad y el rendimiento de tantos paquetes Debian como sea posible.
Nota: Raspbian no está afiliado a Raspberry Pi Foundation. Raspbian fue creado por un pequeño y dedicado equipo de desarrolladores que son fanáticos del hardware Raspberry Pi, los objetivos educativos de la Fundación Raspberry Pi y, por supuesto, el Proyecto Debian.

[10] https://www.raspbian.org/

1.4.4 GPIO[11]

Una característica poderosa de la Raspberry Pi es la fila de pines GPIO (entrada / salida de propósito general) a lo largo del borde superior de la placa se encuentra un encabezado GPIO de 40 pines en todas las placas Raspberry Pi actuales (sin llenar en Pi Zero y Pi Zero W). Antes del Pi 1 Modelo B + (2014), las placas comprendían un cabezal más corto de 26 pines. La figura No. 3 muestra Raspberry Pi GPIO

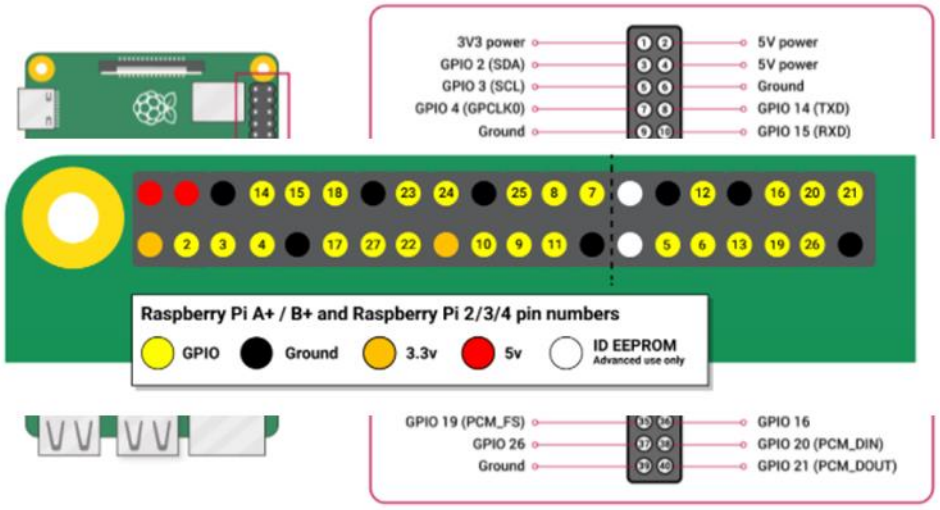


Figura 3. Raspberry Pi GPIO

Cualquiera de los pines GPIO puede designarse (en el software) como un pin de entrada o salida y usarse para una amplia gama de propósitos.

La numeración de los pines GPIO no está en orden numérico; Los pines GPIO 0 y 1 están presentes en la placa (pines físicos 27 y 28) pero están reservados para uso avanzado.

En relación al voltaje, en la placa hay dos pines de 5V y dos pines de 3.3V, así como varios pines de tierra (0V), que no son configurables. Los pines restantes son todos pines 3.3V de uso general, lo que significa que las salidas están configuradas en 3.3V y las entradas son tolerantes a 3.3V.

Un pin GPIO designado como pin de salida se puede configurar en alto (3.3V) o bajo (0V).

Un pin GPIO designado como pin de entrada se puede leer como alto (3.3V) o bajo (0V). Esto se hace más fácil con el uso de resistencias internas pull-up o pull-down.

[11] https://www.raspberrypi.org/documentation/usage/gpio/

Los pines GPIO2 y GPIO3 tienen resistencias pull-up fijas, pero para otros pines esto se puede configurar en el software.

Además de los dispositivos de entrada y salida simples, los pines GPIO se pueden usar con una variedad de funciones alternativas, algunas están disponibles en todos los pines, otras en pines específicos.

- PWM (modulación de ancho de pulso)
  - Software PWM disponible en todos los pines
  - Hardware PWM disponible en GPIO12, GPIO13, GPIO18, GPIO19
- SPI
  - SPI0: MOSI (GPIO10); MISO (GPIO9); SCLK (GPIO11); CE0 (GPIO8), CE1 (GPIO7)
  - SPI1: MOSI (GPIO20); MISO (GPIO19); SCLK (GPIO21); CE0 (GPIO18); CE1 (GPIO17); CE2 (GPIO16)
- I2C
  - Datos: (GPIO2); Reloj (GPIO3)
  - Datos EEPROM: (GPIO0); Reloj EEPROM (GPIO1)
- De serie
  - TX (GPIO14); RX (GPIO15)

1.5. Baterías

1.5.1. Tipos de Baterías

Batería de plomo: Las baterías son dispositivos donde se almacena energía eléctrica en forma de energía química que las convierte en equipos delicados, actualmente se están empleando mayoritariamente 2 tipos de baterías para aplicaciones motrices: las de plomo-ácido y las de litio.

Aunque las baterías de iones de litio están llamadas a ser el futuro de esta tecnología, por ejemplo, no se concibe la automoción eléctrica sin ellas, el menor coste de las baterías convencionales y su capacidad para conformar grandes bancos, las mantienen todavía en el mercado, las convierten en la solución obligada para la gran mayoría de las aplicaciones,

Hay que tener en cuenta que en 2017[12] se vendieron en todo el mundo baterías con una capacidad total de algo más de 500.000 MWh, de las cuales un 77% se corresponde a las baterías de plomo –ácido y el resto a baterías de iones de litio.

Al tratarse de una tecnología con más de 160 años de historia, las baterías de plomo han evolucionado y se han diversificado mucho, todos los fabricantes siguen sacando al mercado baterías cada vez más eficientes, gracias a los avances en ingeniería química y en el desarrollo de nuevas aleaciones, las baterías de plomo presentan unas desventajas inherentes a su propia naturaleza que, si bien cada vez son menores, siguen muy significativos cuando se comparan con las baterías de litio.

[12] https://dcbballester.com/tipos-de-baterias-baterias-de-plomo

Batería de plomo –ácido: En primer lugar, hay que distinguir dos grandes familias: las baterías húmedas o de electrolito líquido, y los tipos sellado. Las primeras son las típicas baterías con una disolución de ácido sulfúrico que exigen un mantenimiento periódico, pues hay que verificar con frecuencia el nivel de densidad de electrolito, y es necesario añadir agua destilada de vez en cuando. Estas baterías pueden presentar derrames de ácido y escapes de hidrógeno por los movimientos de la aplicación, sin embargo, son baterías muy utilizadas ya que ofrecen muy buenas presentaciones: elevados números de ciclos de vida útil a descargas profundas (generalmente se habla en términos de descargas del 80%).

Las baterías de elementos de electrolito líquido utilizadas para tradición suelen ser celdas de 2V unidades en serie hasta conseguir la tensión de trabajo del motor, pero también se fabrican baterías de 6V, 12 V o 24 V. Sin duda su nicho de mercado más popular es el de las carretillas elevadoras, ya que un 90% de estas máquinas de trabajo diario están equipadas con baterías de tracción de este tipo.

Batería VRLA (selladas): Este tipo de baterías son conocidas por sus siglas en inglés: VRLA, Valve Regulated Lead Acid, y son más seguras, más limpias y más fáciles de mantener que las baterías húmedas.
Sus ventajas son:

- No requieren mantenimiento
- No producen gases inflamables durante el proceso de carga (hidrógeno)
- No producen derrames

Como el electrolito no es líquido, sino que está retenido en gel o malla, el hidrógeno y el oxígeno que se producen durante la carga no pueden viajar hasta la superficie y evaporarse. La presión los obliga a desplazarse a los polos opuestos para recombinarse en agua en el polo positivo.

Aunque el coste de adquisidores de esas baterías es mayor que el de las de electrolito húmedo, su coste de operación es menor ya que no quieren mantenimiento, son fáciles de transportar y de instalar, sobre todo, son mucho más seguras al evitar tanto los escapes de gases inflamables como los derrames de ácido (las instalaciones donde se utilizan estas baterías no necesitan requisitos especiales, son idóneas para embarcaciones o para carretillas elevadoras que trabajan en ambiente sanitario).

Por eso, este tipo de baterías son más utilizadas en numerosas aplicaciones, tanto para movilidad eléctrica como para sistemas de almacenamiento de emergencia por si falla el suministro de la red, en instalaciones solares aislados o en aquellas instalaciones donde el suministro de energía no está siempre disponible.

Se puede distinguir dos tipos de baterías VRLA: las AGM y las de GEL. La primera, Absorbed Glas Mat, mantienen el electrolito absorbido en un tejido de fibra de vidrio, y destacan por aceptar corrientes elevadas y menores tiempos de cargas, en las de

tipo GEL el electrolito se encuentra en forma de gel a base de sílica y aceptan descargas más profundas y con mayor ciclo de vida de las anteriores, especialmente las que se construyen con placas tubulares.

Las más eficientes en este sentido son las baterías de 2V (denominados OPzV), que pueden tener una capacidad de hasta 5000 Ah y se pueden instalar en serie en la cantidad necesaria para alcanzar la tensión de trabajo. Para aplicaciones donde la cantidad necesaria de energía no es tan elevada, existen baterías mono bloque de 12V. En la figura No. 4 y No. 5 se muestran tipos y arreglos de baterías.

Figura 4. Batería monobloque 12 v, tipo AGM, marca Victron Energy

Figura 5. Banco de batería de 48 v, 24 baterías OpzV de 2 v cada una, Photonic Universe.

1.5.2. Capacidad de carga

Es la cantidad de electricidad que puede obtener mediante la descarga total de una batería inicialmente cargada al máximo. La capacidad de un acumulador se mide en Amperios horas (Ah) para un determinado tiempo de descarga, es decir, una batería de 130Ah es capaz de suministrar 130A en una hora o 13A en diez horas. Para acumuladores fotovoltaicos es usual referirse a tiempos de descargas de 100 horas. También al igual que para módulos solares pueden definirse el voltaje de circuito abierto y el voltaje de carga. Las baterías tienen un voltaje nominal que suele ser de 2V, 6V, 12V,24V, aunque siempre varia durante los distintos procesos de operación.

1.5.3 Tiempo de descarga

El tiempo de descarga depende de la capacidad de la batería y de la cantidad de electricidad que consuman los equipos conectados. Como regla general, cuanto antes se descarga una batería, menos electricidad suministra. Y también sucede, al contrario: Cuando más tiempo tarda en descargase una batería, más energía puede obtenerse de ella. Una batería de ácido de plomo de 100 Ah suministra una corriente de 5A durante 20 horas, periodo de tiempo en la que tensión no desciende por debajo de 10, 5 voltios Esto equivale a 100 Ah. Si se conecta un dispositivo de 100 A a la misma batería, la batería será capaz de alimentar el dispositivo solo 45 minutos. Transcurrido ese tiempo, la tensión de la batería descenderá a 10,5 voltios y la batería se vaciará, habiendo suministrado no más de 75 Ah. Al contrario que las baterías de ácido –plomo, la capacidad de las baterías de Iones de Litio no se ve afectada por el dispositivo conectado, una batería de Iones de Litio siempre suministra el 100% de su capacidad, independientemente de la carga de consumo conectada.

1.5.4 Circuito cargador de batería 12 V

Un cargador sencillo trabaja haciendo pasar una corriente continua o tensión, entre otras. El cargador sencillo no modifica su corriente de salida basándose en el tiempo de carga de la batería. Esta sencillez facilita que sea un cargador barato, pero también de baja calidad. Este cargador suele tardar bastante en cargar una batería para evitar daños por sobrecarga.
Incluso así, una batería que se mantenga mucho tiempo en un cargador sencillo pierde capacidad de carga y puede llegar a quedar inutilizable.

Capítulo 2

Control Automático, Automatización

2.1. Introducción

En el capítulo 1 se explicó las generalidades de los elementos que constituyen El Controlador Eléctrico y Automatización. Ahora se presentan detalles de los elementos constitutivos del Controlador Electrónico donde el elemento fundamental es la aplicación para Automatizar procesos. Junto con esto, están los fundamentos de cálculo que el ingeniero necesita para el diseño, definición de los criterios de seguridad y confiabilidad necesarios hasta llegar a la ejecución. Además, se aborda la ingeniería de Controladores, aspectos fundamentales, sus tipos y variantes constructivas.

Si bien, hoy día los Controladores tienen una intervención cada vez más importante en la vida diaria, desde los simples controles que hacen funcionar un tostador automático hasta los complicados sistemas de control necesarios en vehículos espaciales, en guiado de proyectiles, sistemas de pilotajes de aviones, etc. Además, el control automático se ha convertido en parte importante e integral de los procesos de manufactura e industriales modernos. Por ejemplo, el control automático resulta esencial en operaciones industriales como el control de presión, viscosidad y flujo en las industrias de procesos, maquinado en las industrias de fabricación, entre muchas otras.

2.2. Formas de Operación de Sistemas de Control

Un sistema de control es aquel en el que las variables de salida se comportan según las órdenes dadas por las variables de entrada, al cual se le aplica una señal r(t) a manera de entrada para obtener una respuesta o salida y(t), puede representarse mediante bloques, se aprecia en la figura No.6

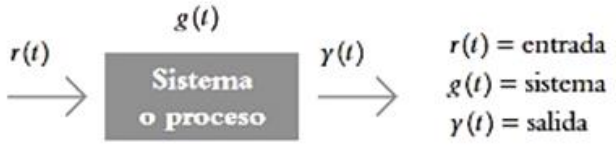


Figura 6. Sistema de Control de Bloques.

El vínculo entrada-salida es una relación de causa y efecto con el sistema, por lo que el proceso por controlar (también denominado planta) relaciona la salida con la entrada. Las entradas típicas aplicadas a los sistemas de control son: escalón, rampa e impulso, según se muestra en la figura No.7

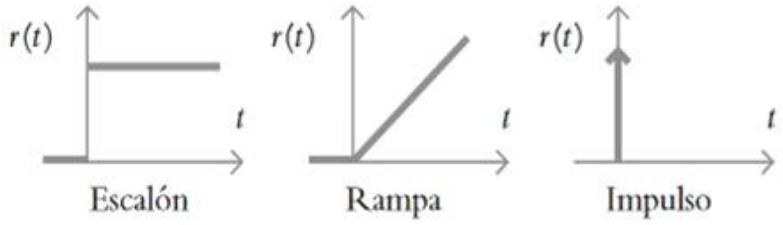


Figura 7. Entradas típicas de los Sistemas de Control.

La entrada escalón indica un comportamiento o una referencia constantes introducidos al sistema, mientras que la entrada rampa supone una referencia con variación continua en el tiempo, y la entrada impulso se caracteriza por ser una señal de prueba con magnitud muy grande y duración muy corta. La función respuesta impulso o función de transferencias la representación matemática del sistema. Básicamente, el problema de control consiste en seleccionar y ajustar un conjunto específico de elementos tal que, al interconectarse, el sistema resultante deberá comportarse de una manera específica.[13]

2.3. Representación de un sistema de control en diagramas de bloque

Se utiliza para describir, gráficamente, las partes de las que consta un sistema, así como sus interconexiones. El bloque en sí contiene la descripción, el nombre del elemento o el símbolo de la operación matemática que se ejecuta sobre la entrada r (t) para producir la salida y (t). El punto de suma se utiliza cuando a un bloque se le aplican dos o más entradas, en tanto que el bloque se sustituye por un círculo, cuya

[13] I.R.H. Gaviño, Clasificación de los sistemas de Control. Introducción a los Sistemas de Control: Conceptos, aplicaciones y simulación con Matlab, México, Pearson Educación 2010.

salida representa la suma algebraica de las entradas. El punto de reparto, representado por un punto, se usa cuando una señal se bifurca para aplicarse a más de un bloque a como se muestra en la figura No. 8, a continuación.

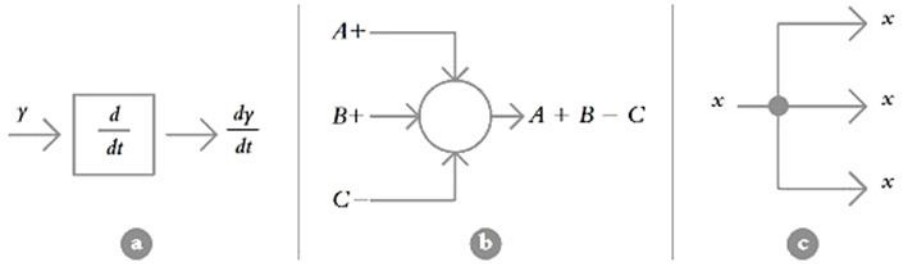


Figura 8. Sistema de Control en diagrama de bloque.

2.4. Clasificación de Sistemas de Control

En todos los sistemas de control se usan con frecuencia componentes de distintos tipos, por ejemplo, componentes mecánicos, eléctricos, hidráulicos, neumáticos y combinaciones de estos. Un ingeniero que trabaje con control debe estar familiarizado con las leyes físicas fundamentales que rigen estos componentes. Sin embargo, en muchos casos, los fundamentos existen como conceptos aislados con muy pocos lazos de unión entre ellos. El estudio de los controles automáticos puede ser de gran ayuda para establecer lazos de unión entre los diferentes campos de estudio haciendo que los distintos conceptos se usen en un problema común de control.

Podemos referirnos a la automatización como el conjunto de elementos electrónicos y mecánicos para realizar un proceso con intervención parcial o nula del ser humano, se definirán los sistemas de control de lazo abierto (no automático) y de lazo cerrado (automático)

Sistema de control de Lazo Abierto

Los sistemas de control de lazo abierto son sistemas en los que la salida no tiene efecto sobre la señal o acción de control, es decir, la salida ni se mide ni se realimenta para compararla con la entrada.

Los elementos de un sistema de control de lazo abierto se pueden dividir en dos partes: el controlador y el proceso controlado. Una señal de entrada o comando se aplica al controlador, cuya salida actúa como una señal de control o señal actuante, la cual regula el proceso controlado, de tal forma que la variable de salida o variable controlada se desempeñe de acuerdo a ciertas especificaciones o estándares establecidos. En los casos simples, el controlador puede ser un amplificador, filtro,

unión mecánica u otro elemento de control. En los casos más complejos puede ser una computadora tal como un microprocesador.

En los sistemas de control de lazo abierto, no se compara la salida con la entrada de referencia. Por lo tanto, para cada entrada de referencia corresponde una condición de operación fijada.

Un sistema de control de lazo abierto es insensible a las perturbaciones; por consiguiente, un sistema de control de este tipo es útil cuando se tiene la seguridad que no existen perturbaciones actuando sobre el mismo. En la práctica solo se puede usar el control de lazo abierto si la relación entre la entrada y la salida es conocida, y si no hay perturbaciones internas ni externas importantes.

Este tipo de sistemas por lo general utiliza un regulador o actuador con la finalidad de obtener la respuesta deseada, ver figura No.9 a continuación:

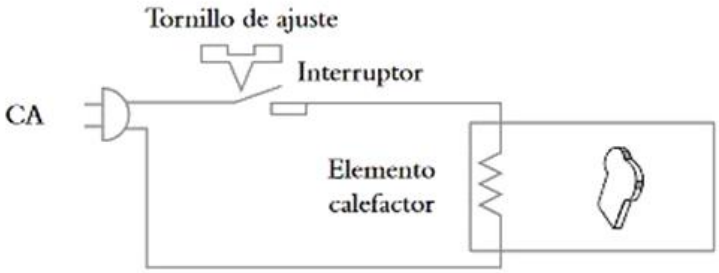


Figura 9. Sistema de Control de Lazo Abierto.

Sistema de control de Lazo Cerrado[14]

En los sistemas de control de lazo cerrado, la salida o señal controlada, debe ser realimentada y comparada con la entrada de referencia y se debe enviar una señal actuante o acción de control, proporcional a la diferencia entre la entrada y la salida a través del sistema, para disminuir el error y corregir la salida.

Un sistema de control de lazo cerrado es aquel en el que la señal de salida tiene efecto directo sobre la acción de control. Esto es, los sistemas de control de lazo cerrado son sistemas de control realimentados. La diferencia entre la señal de entrada y la señal de salida se la denomina señal de error del sistema; esta señal es la que actúa sobre el sistema de modo de llevar la salida a un valor deseado. En otras palabras, el término lazo cerrado implica el uso de acción de realimentación negativa para reducir el error del sistema.

[14] I.R.H Gaviño, Clasificación de los sistemas de Control. Introducción a los Sistemas de Control: Conceptos, aplicaciones y simulación con Matlab, México, Pearson Educación 2010.

El término retroalimentar significa comparar; en este caso, la salida real se compara con respecto al comportamiento deseado, de tal forma que si el sistema lo requiere se aplica una acción correctora sobre el proceso por controlar. La figura No. 10 muestra la configuración de un sistema retroalimentado.

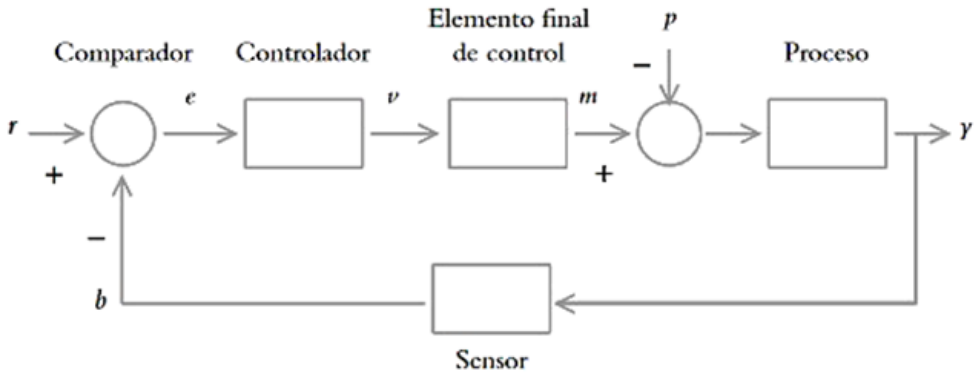


Figura 10. Sistema retroalimentado de Control de Lazo Cerrado.

Sistema de control en bucle abierto (Open loop control system)

Se caracteriza por recibir información en sus entradas sobre el valor de las variables que controlan, acerca del valor que tienen la variable del producto o proceso que quiere controlar, un ejemplo sería una lavadora "automática" común, ya que ésta realiza los ciclos de lavados en función a una base de tiempo, sin medir el grado de limpieza de la ropa, que sería la salida a considerar. Otro ejemplo sería una tostadora: Al hacer una tostada, se coloca el tiempo que suponemos suficiente para que el pan salga con el grado de tostado que queremos, más la tostadora no puede decidir si ya está suficientemente tostado o no. En la figura No.11 se detalle el sistema de control.

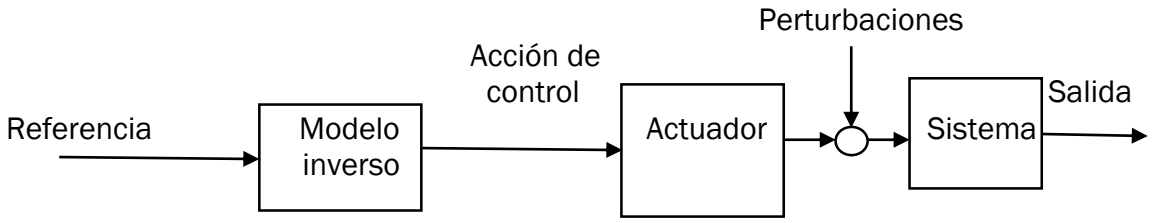


Figura 11. Sistema de Control de bucle abierto.

Estos sistemas se caracterizan por:

- Ser sencillos y de fácil mantenimiento.
- Estar afectados por perturbaciones
- Depender altamente de la calibración del sistema

2.5. Control secuencial y control continuo

Para controlar un proceso industrial, lo primero es identificar el tipo de señales con que se va a trabajar, las variables involucradas en un proceso industrial pueden ser principalmente de dos tipos:

- Señales todo-nada o binarias (on-off signals): sólo pueden tener dos valores diferentes en régimen permanente a lo largo del tiempo y por lo cual se las denominan digitales. Por ejemplo: botón pulsado/no pulsado, válvula abierta /cerrada, presencia de objeto /no presencia.

- Señales analógicas (analog signals) que pueden tener cualquier valor dentro de unos determinados márgenes y que llevan la información en sus amplitudes. Un ejemplo de variables analógica es la velocidad de un motor. Ejemplo temperatura, humedad, luminosidad.

En la figura No.12 se puede observar un controlar discreto (Computador o PLC) que controla una planta discreta mediante actuadores sensores digitales.

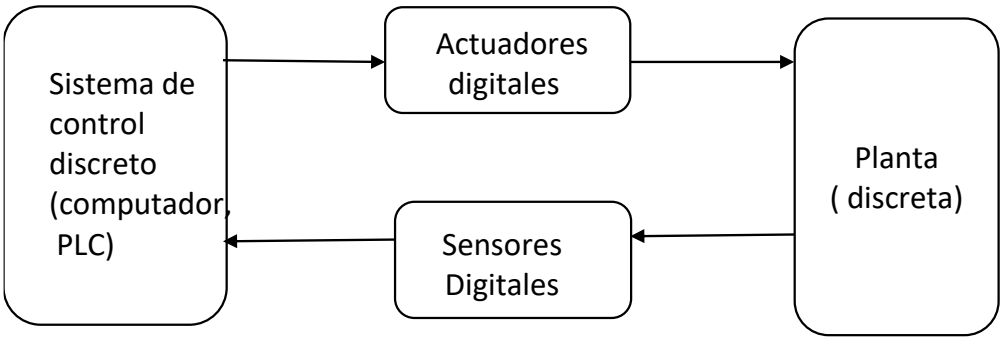


Figura 12. Control Discreto

En función del controlador tendremos los sistemas de control de procesos continuos o simplemente sistemas de control de procesos (process control systems).

- Si el controlador es continuo, se refiere a control continuo tradicional. Los sistemas analógicos de control suelen estar formados por electrónica analógica.
  Presenta características de no ser programables, es decir, que para cambiar la función que realizan hay que modificar los elementos que forman parte de ellos o el cableado entre los mismos.

- Si el controlador es digital, suele utilizar un microcontrolador como elemento central. Al control de procesos mediante un computador se le dio el nombre de control digital directo, conocido como DDC (Direct Digital Control).
  Estos sistemas utilizan convertidores analógicos digital (CAD) para digitalizar las señales obtenidas de la planta, y convertidores digitales analógico (CDA) para convertir las salidas del sistema de control a señales continuas en la planta.

En la figura No.13 se puede observar un controlador discreto (computador o PLC) que controla una planta continua. Para ellos las salidas digitales del controlador se convierten a continua a través de un CDA, y a las entradas del controlador llegan las señales analógicas de la planta a través de un CDA.

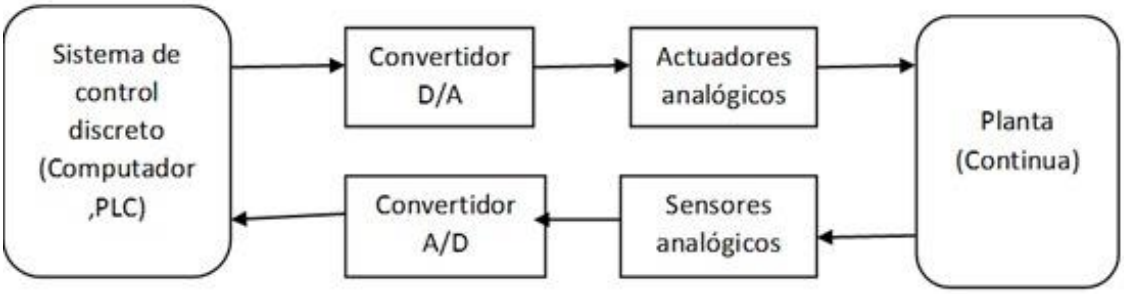


Figura 13. Controlador discreto para una planta continua.

Actualmente la mayoría de los procesos industriales son híbridos (Hybrid systems), reciben y generan tantas señales analógicas como todo - nada (digitales). Debido a ello, actualmente los procesos industriales se controlan mediante sistemas digitales que constituyen al mismo tiempo un controlador lógico y un controlador de procesos continuos también denominado regulador.

2.6. Requerimientos generales de un Sistema de Control[15]

La estabilidad, exactitud y rapidez de respuesta son características que debe tener todo sistema de control.

Estabilidad: Necesariamente, un sistema debe ser estable, esto significa que la respuesta a una señal, ya sea el cambio del punto de referencia a una perturbación, debe alcanzar y mantener un valor útil durante un período razonable.

[15] M. Pérez, A. Pérez Hidalgo, E. Pérez Berenguer, Introducción a los Sistemas de Control y Modelo Matemático para Sistemas Lineales Invariantes en el Tiempo, 2007.

Un sistema de control inestable producirá, por ejemplo, oscilaciones persistentes o de gran amplitud en la señal, o bien, puede hacer que la señal tome valores que corresponden a límites extremos. Una respuesta inestable es indeseada desde el punto de vista de control. Es necesario también, conocer la cantidad o grado de estabilidad que tiene un sistema, porque puede suceder que un sistema que sea estable, esté cerca de los límites de pasar de ser estable a inestable por el uso que se le dé al sistema en el transcurso del tiempo, o por el cambio de algún componente al realizar cualquier tipo de mantenimiento. La inestabilidad está latente en cada sistema, por eso es importante poder medir la cantidad de estabilidad.

Exactitud: Un sistema de control debe ser exacto dentro de ciertos límites especificados, esto significa que el sistema debe ser capaz de reducir cualquier error a un límite aceptable. Es conveniente hacer notar que no hay sistemas de control alguno que pueda mantener un error cero en todo tiempo, porque siempre es necesario que exista un error para que el sistema inicie la acción correctora. Aun cuando haya sistemas que matemáticamente pueden reducir a cero el error en el sistema, esto no sucede en la realidad a causa de las pequeñas imperfecciones inherentes a los componentes que forman el sistema. En muchas aplicaciones de control, no se requiere una exactitud extrema. La exactitud es muy relativa y sus límites están basados en la aplicación particular que se haga del sistema de control.

Rapidez de respuesta: Es la cualidad que debe tener un Sistema de control para que funcione a tiempo. Un sistema de control debe completar su respuesta a una señal de entrada en un tiempo aceptable. Aunque un sistema sea estable y tenga la exactitud requerida no tiene ningún valor si el tiempo de respuesta a una entrada, es mucho mayor que el tiempo entre las señales.

2.7. Reguladores

2.7.1 Consumo eléctrico del regulador

Voltaje[16]: La unidad de medición es el volt, simbolizada por V, se eligió en honor a Alessandro Volta. De forma, si se requiere un joule de energía (1 J) para mover la carga de un coulomb (1C), a como se detalla en la figura No.14 desde una posición y la diferencia de potencial o voltaje entre los dos puntos será de un volt (1 V).

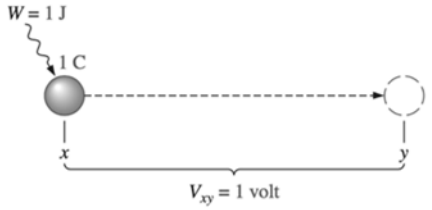


Figura 14. Unidad de medida para el voltaje.

[16] R. L. Boylestad, Introducción al análisis de circuitos, Naucalpan de Juárez: PEARSON EDUCACIÓN, 2004.

Si la energía requiere para mover la carga de 1 C se incrementa a 12 J debido a fuerzas adicionales de oposición, entonces la diferencia de potencial se incrementará a 12 V. El voltaje es entonces una señal de cuanta energía se encuentra involucrada en el movimiento de una carga entre dos puntos en un sistema eléctrico. Y de forma inversa, mientras mayor sea el nivel de voltaje de una fuente de energía tal como es una batería, más energía se encontrará disponible para mover cargas a través del sistema. Una diferencia de potencial o voltaje siempre se mide entre dos puntos en el sistema. Al cambiar cualquier punto puede cambiar la diferencia de potencial entre los dos puntos bajo análisis.

Corriente Eléctrica[17]: El movimiento de estos electrones libres del extremo negativo del material al extremo positivo es la corriente eléctrica, simbolizada mediante I. La corriente eléctrica es la velocidad que lleva el flujo de la carga, es un material conductor, el número de electrones (cantidad de carga) que fluyen más allá de ciertos puntos en una unidad de tiempo determinado la corriente. Se muestra en la figura No.15

$$I = \frac{Q}{t}$$

Ecuación 1. Corriente Eléctrica

Donde:

I= corriente en amperes (A). Un ampere (1 A) es la cantidad de corriente que existe cuando cierto número de electrones, cuya carga total es de un coulomb (1 C), para por un área de sección transversal dada en un segundo (1 s)

Q = cargas en coulombs (C)

t = tiempo en segundos (S)

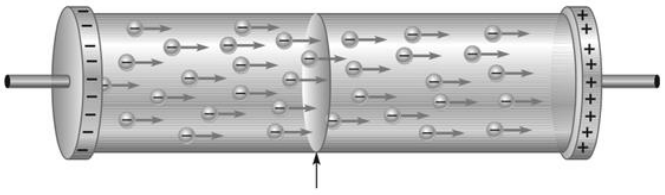


Figura 15. Ilustración de 1 A de corriente en un material.
Cuando cierto número de electrones que tienen una carga total de 1C pasan por un área de sección transversal en 1s, la corriente es de 1 A.

[17] T. L. Floyd, Principios de circuitos eléctricos. Octava Edición, Naucalpan de Juárez: Pearson Educación, 2007.

Potencia:[18] Es la magnitud física dada por el cociente entre la energía transferida en un cierto lapso y el valor de ese tiempo. La tensión o diferencia de potencial eléctrico U, como cociente entre energía y carga, en J/C o Volt.
Por otra parte, la corriente I es el cociente entre la carga y el tiempo, en C/s, o ampare. Entonces, del producto o multiplicación de la tensión por la corriente, resulta la potencia eléctrica.

$$P = UI$$

Ecuación 2. Potencia

Carga Eléctrica[19]: Si un cuerpo atrae a otros dos, estos se repelen. Si rechaza a otros dos, estos también se repelan. Y si un cuerpo atrae a otro y rechaza un tercero, estos dos últimos cuerpos se atraen. De eso se dedujo que hay dos clases de electricidad, primitivamente llamadas ambarina y vítrea, la del ámbar y la del vidrio. Después se las llamó polaridades negativa y positiva, respectivamente.
Hoy, 24 siglos después, explicamos esos efectos por la estructura atómica de la materia que está compuesta por átomos, a su vez, formados por protones positivos, electrones negativos y neutrones neutros a como se muestra en la figura No. 16

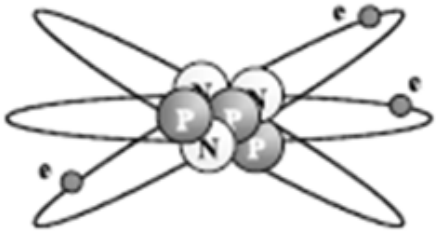


Figura 16. Átomo de litio

2.7.2 Regulador LM2596 Ajustable en voltaje y corriente

La serie de reguladores LM2596 son monolíticos integrados, circuitos que proporcionan todas las funciones activas para un reductor (buck) regulador de conmutación, capaz de impulsar una carga de 3A con excelente regulación de línea y carga. Estos dispositivos están disponibles en voltajes de salida fijos de 3.3V, 5V, 12V, y ajustable versión de salida.

Al requerir un número mínimo de componentes externos, estos reguladores son fáciles de usar e incluyen frecuencia interna compensación †, y un oscilador de frecuencia fija.

[18] Rela Agustín, Electricidad y Electrónica, Morvillo Anselmo, capítulo 4 Electricidad Fundamentos de Electrodinámica. 19/01/2010
[19] Rela Agustín, Electricidad y Electrónica, Morvillo Anselmo, capítulo 4 Electricidad Fundamentos de Electrodinámica. 19/01/2010

La serie LM2596 opera a una frecuencia de conmutación de 150 kHz, lo que permite componentes de filtro de tamaño más pequeño que lo que se necesitaría con reguladores de conmutación de frecuencia más baja. Disponible en un paquete estándar TO-220 de 5 derivaciones con varias opciones de curvatura de derivaciones diferentes y un TO-263 de 5 derivaciones paquete de montaje en superficie.

Se dispone de una serie estándar de inductores de varios fabricantes optimizados para su uso con la serie LM2596, esta característica simplifica enormemente el diseño de fuentes de alimentación conmutadas, otras características incluyen una tolerancia garantizada de ± 4% en la salida voltaje bajo voltaje de entrada especificado y carga de salida condiciones, y ± 15% en la frecuencia del oscilador.

Apagado externo incluido, con una corriente de espera típicamente de 80 μA.

Las características de autoprotección incluyen una frecuencia de dos etapas, reducir el límite de corriente para el interruptor de salida y un apagado en exceso de temperatura para una protección completa en condiciones de falla.

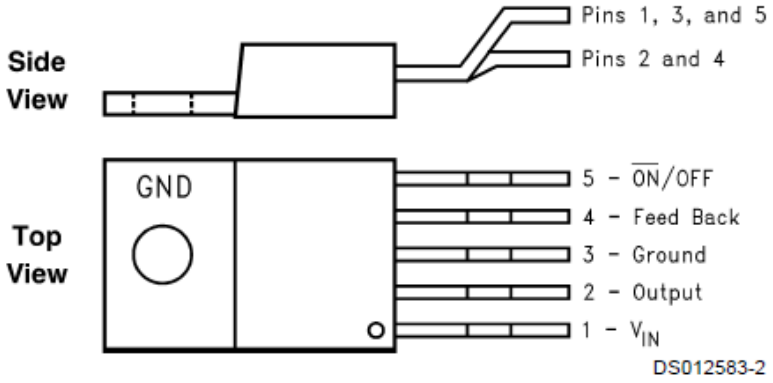


Figura 17. Regulador LM2596

Características:

- 3.3V, 5V, 12V y versiones de salida ajustable
- Rango de voltaje de salida de la versión ajustable, 1.2V a 37V ± 4% máximo en condiciones de línea y carga
- Disponible en paquetes TO-220 y TO-263
- Corriente de carga de salida garantizada de 3 A
- Rango de voltaje de entrada hasta 40 V
- Requiere solo 4 componentes externos
- Excelentes especificaciones de regulación de línea y carga
- Oscilador interno de frecuencia fija de 150 kHz

- Capacidad de apagado TTL
- Modo de espera de bajo consumo, IQ típicamente 80 µA
- Alta eficiencia
- Utiliza inductores estándar fácilmente disponibles
- Apagado térmico y protección de límite de corriente

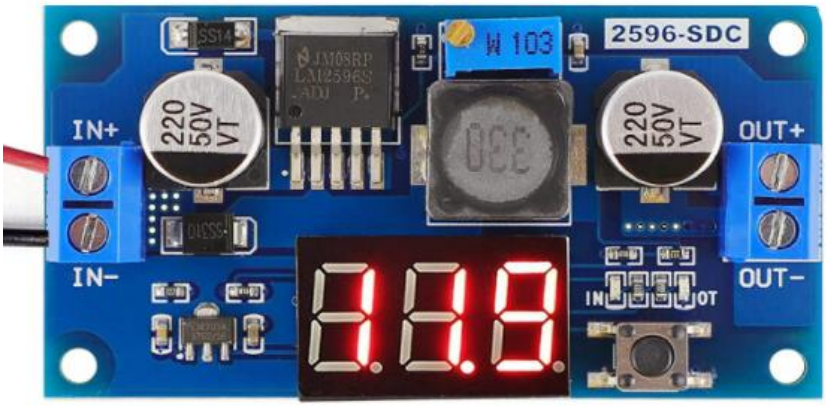


Figura 18. Fuente regulada basada en LM2596 tipo Step Down

Este módulo permite obtener un voltaje regulado a partir de una fuente de alimentación con un voltaje mayor, es así como puedes regular una fuente de 12V a 2.2V, 3.3V, 5V, etc.

Esta placa está basada en el regulador DC-DC Step Down LM2596 que es un circuito integrado monolítico adecuado para el diseño fácil y conveniente de una fuente de conmutación tipo buck. Es capaz de conducir una corriente de hasta 3A. Maneja una carga con excelente regulación de línea y bajo voltaje de rizado. Este dispositivo está disponible con voltaje de salida ajustable. El módulo reduce al mínimo el uso de componentes externos para simplificar el diseño de fuentes de alimentación.
Incluye un voltímetro digital ideal para proyectos con Arduino, PIC, Raspberry Pi, etc.

LM2596 DC-DC Step-down Adjustable CC/CV Power Supply Module

Figura 19. LM2596 DC-DC Step-down Adjustable CC/CV Power Supply Module

El convertidor reductor LM2596 que es un módulo reductor ajustable de voltaje constante de corriente constante. Eso significa que pones un voltaje más alto en un extremo del módulo y en el otro extremo del módulo obtienes un voltaje más bajo. Este módulo se usa comúnmente para cargar baterías de litio o para ejecutar LED, también puede usarlo como una fuente de alimentación de sobremesa, pero el voltaje será un poco bajo en este módulo. Es un regulador de conmutación, lo que significa que la eficiencia debería ser bastante alta, pero aún no puede generar tanta corriente.

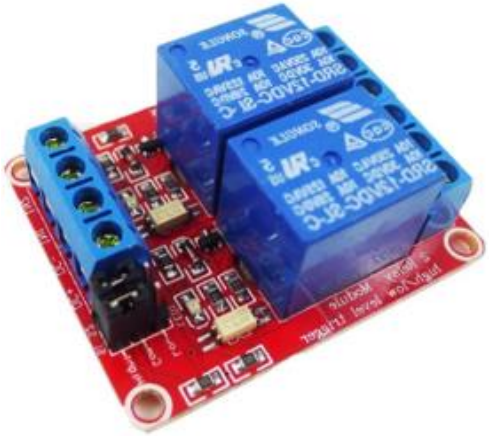


Figura 20. Módulo relé de 2 canales

Estos módulos poseen optoacopladores para activar los relés, por lo que es posible activarlo directamente con un pin de una Raspberry Pi, Arduino o microcontrolador, sin el riesgo de dañarlo. Cuenta con LED de encendido para indicar el estado de cada relé. Posee un jumper con el cual se podrá seleccionar si los relés se alimentan directamente de VCC (jumper a la izquierda), o por medio de una fuente de voltaje externa (jumper retirado y alimentar el pin JD-VCC)

Especificaciones:

- Voltaje de operación: 5V.
- Corriente en cada relé: 10A máximo.
- Voltaje en cada relé: 250VAC máximo.
- Dispone de agujeros para atornillar.
- Canales activados por medio de optoacopladores.
- Dimensiones: 5cm x 3.9cm x 2cm (altura)

Interfaz de conexión:

- VCC: Alimentación. (5V)
- GND: Tierra.
- IN1: Entrada del relé 1(Se activa con cero lógico)
- IN2: Entrada del relé 2(Se activa con cero lógico)
- NO1: Normalmente abierto del relé 1.
- COM1: Común del relé 1.
- NC1: Normalmente cerrado del relé 1.
- NO2: Normalmente abierto del relé 2.
- COM2: Común del relé 2
- NC2: Normalmente cerrado del relé 2

La unidad contiene:

- 1 x Módulo relé de 2 canales

Capítulo 3

Diseño Metodológico

3.1 Introducción

El presente trabajo de investigación se origina por el interés de dar un aporte significativo a las pequeñas empresas proveedoras de tratamientos para metales, principalmente Superkote 2000, contribuyendo a un mejor desempeño, fortaleciendo el diseño y mecanismo de suministro del aceite.

La metodología del trabajo empleada se desarrolló bajo el enfoque analítico-descriptivo utilizando técnicas de investigación de campo y bibliográficas. Se describirán las diversas etapas para desarrollar dicho proyecto con un enfoque que utiliza los principios de la ingeniería conducida por modelos para desarrollar sistemas de control.

3.2 Recopilación de la Información

Se recopila la información por medio de la encuesta como instrumento, con ayuda del cuestionario que fue dirigido a la administración, de esta manera se obtuvieron los datos necesarios que sirvieron de guía para la búsqueda de la solución a la problemática planteada.

También, se deben conocer las condiciones físicas, técnicas en donde se empleará del controlador electrónico, los volúmenes de tratamiento para metal que se suministrará, equipos necesarios para desarrollar además de su respectiva ficha técnica, los costos de los elementos para desarrollar el controlador.

3.3 Análisis de la Información

Con el análisis de la información obtenida, se pudo establecer el diseño del controlador, a la vez los equipos a utilizarse, obteniendo un sistema de bajo costo y un desempeño de calidad.

La adopción de los principios de ingeniería de software en la automatización industrial requiere la integración de métodos y herramientas que proporcionen interfaces con otras herramientas de ingeniería definiendo así la lógica de programación a emplear.

3.4 Diseño

Para el diseño del controlador se hizo uso de una bomba de bajo consumo eléctrico, las cuales se encuentran en el mercado nacional, se utiliza un suministro de energía independiente del suministro eléctrico comercial, asegurando que el diseño fuera de bajo costo, se hizo uso de un controlador basado en tecnología Raspberry que cumpliera con autonomía energética y que aplicara las dosis exactas de acuerdo a los términos de referencia analizados previamente.

3.5 Diagrama de bloque

Siendo la representación gráfica de las funciones que lleva a cabo cada componente. Éste diagrama muestra las relaciones existentes entre los componentes del controlador. A diferencia de una representación matemática puramente abstracta, el diagrama de bloque tiene la ventaja de indicar de forma más realista el funcionamiento ordenado del Controlador Electrónico.

3.6 Elaboración de Prototipo

Una estrategia de evaluación del diseño, es realizar un prototipo para someterlo a pruebas de calibración de las cantidades suministradas regularmente, en donde se documenta todo el proceso de elaboración de cada una de los equipos, lógica de programación.

3.7 Implementación

Una vez que el prototipo pasa las pruebas de calibración se termina con la construcción del diseño final, para que el usuario lo ocupe donde requiera.

Capítulo 4

Desarrollo del Trabajo

4.1 Introducción

La industria actual debe abordar un cambio tecnológico en los procesos de producción para mantener el liderazgo en el entorno económico del tercer milenio, de acuerdo al asesor industrial de la Comisión Europea[20] "Las fábricas de futuro" requieren sistemas automatizados más complejos, seguros y fiables.

El desarrollo de software de control constituye uno de los retos en el campo de la automatización para garantizar la disponibilidad de producción de alta calidad con cero defectos, siendo el controlador o regulador el elemento fundamental en un sistema de control que determina el comportamiento del bucle, ya que condiciona la acción del elemento actuador en función del error obtenido. La forma en que el regulador genera la señal de control se denomina acción de control.

El desarrollo de este tipo de software se enfrenta a la complejidad de integrar las tecnologías de la información en el entorno industrial a la hora de transformar información en acciones de dispositivos de un modo totalmente controlado. Para ello, es preciso generar, transmitir y procesar la información de modo rápido, exacto y fiable, en cantidad y calidad adecuadas a las necesidades específicas, en el momento preciso y en el lugar idóneo. Este flujo distribuido de datos debe garantizar el correcto flujo de materiales de un modo automático.

[20] Effra, Factories of the future PPP, EU, 2005

4.2 Diseño y Selección Elementos del Controlador.

4.2.1 Diseño de controlador

Para la selección del controlador se consideró como primer parámetro la autonomía del sistema, o sea que este no dependiera de una conexión de fuente comercial AC, para que el dispositivo goce de independencia energética accesible y estable, sería un acumulador de 12V 7Ah o uno de 12V 4Ah, que comúnmente encontramos en los equipos de Sistemas de Alimentación Ininterrumpida (SAI) en el mercado nacional.

A la vez se requiere de una interfaz gráfica con un dispositivo compacto, donde se pueda llevar una secuencia del proceso, se analizó la posibilidad de implementar Arduino, pero inmediatamente se descartó debido a que no cuenta con la característica solicitada por la empresa DPS, como lo es la interfaz gráfica donde se guiará al usuario en un proceso paso a paso.

Tras la búsqueda y el análisis bibliográfico se logró identificar la Raspberry Pi como el dispositivo ideal para el mando del controlador, de ahí que se seleccionó la Raspberry Pi 3 B+, siendo este nuestro componente robusto y que cumple con la interfaz gráfica, ya que se agregó una pantalla táctil para la interacción y seguimiento del proceso.

Además, es necesario un circuito con una alimentación DC estable para evitar fluctuación en el sistema y que por tal motivo se viera afectado. Partiendo de esa condición se evaluaron las propiedades físicas de Superkote 2000, haciendo pruebas de conductividad eléctrica y viscosidad del mismo para la selección de la bomba que impulsaría el líquido.

4.2.2. Selección de Elementos del Controlador

Regulador de voltaje: Este dispositivo es de tipo conmutado de retardo con alta eficiencia de conversión, excelente regulación de línea y bajo voltaje de rizado; a diferencia de los reguladores lineales no generan mucho calor, además de un apagado en exceso de temperatura para protección completa en condiciones de falla.

Se selecciona el regulador de voltaje LM2596, DC-DC para la alimentación de:

- Raspberry Pi 3 B+
- Batería 12v 7Ah
- Bomba o motor

Cada dispositivo de los antes mencionados tendrá un módulo independiente con el regulador LM2596 para la carga y encendido a quien corresponda.

La Raspberry Pi contiene un módulo de tipo Fuente regulada basada en LM2596 tipo Step Down, desde la batería se alimentará, con 12V en su entrada y a su salida tendrá la capacidad de entregar a la Raspberry Pi 3 B+ 5.3V voltaje necesario y 2A la corriente máxima especificada por el fabricante. Se observa en la figura No.21

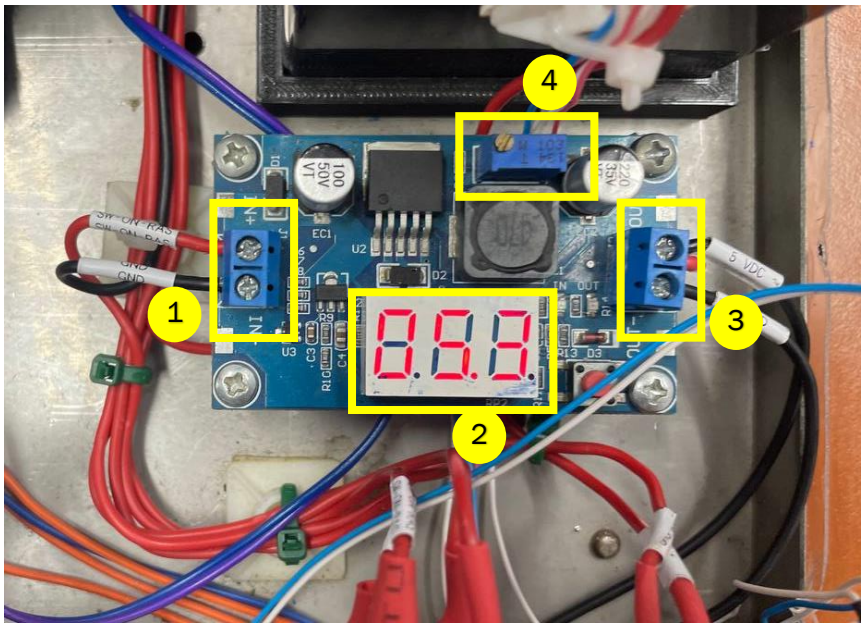


Figura 21. Módulo # 2 LM2596

Donde:

1. Voltaje de entrada 12V desde la batería.
2. Display 7 segmentos que muestra el voltaje de la salida.
3. Voltaje de Salida 5.3V para la Raspberry Pi 3 B+ y módulo de relé dos canales.
4. Potenciómetro para regular en voltaje de salida.

Batería: Esta es una fuente de alimentación de 12V DC, capaz de entregar 12V hasta 7A, según lo especificado por el fabricante (EPCOM).

Soporte para Batería: Se diseñó e imprimió soporte para ajustar el acumulador a la carcasa en el laboratorio de monografía y proyectos de innovación y desarrollo de la FEC con una impresora 3D.

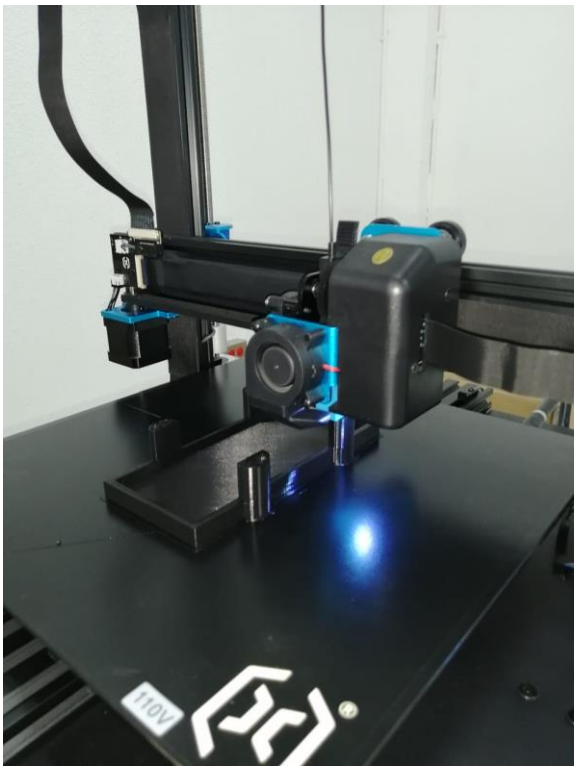


Figura 22. Elaboración de piezas para soporte de batería en Impresora 3D marca ENDER

De la batería; el módulo para carga de la batería es LM2596 DC-DC Step-Down Ajustable CC/CV Power Supply, en este módulo tenemos la posibilidad de realizar tres ajustes a través de potenciómetros:

- Regulación de voltaje de salida
- Ajuste de carga completa
- Regulación de salida de corriente

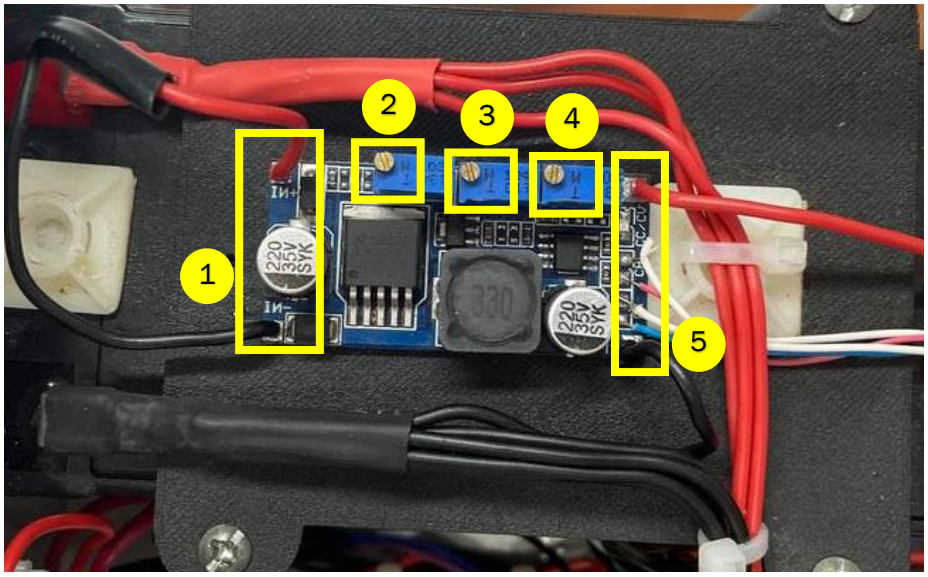


Figura 23. Módulo # 1 LM2596

Donde:

1. Voltaje de entrada desde la Fuente AC-DC de 19.5V
2. Potenciómetro Para regular el voltaje de salida que alimentará la batería
3. Potenciómetro para regular el voltaje en que se hará el corte cuando la batería llegue al voltaje correspondiente a batería cargada.
4. Potenciómetro para regular la corriente de salida para la batería
5. Conexiones de salida para la batería

Bomba: Se optó por utilizar Bomba de diafragma, dado que el flujo a dosificar es bajo, la capacidad de mover liquido es hasta 3 lt/min.

Las características:

- Bomba de agua sumergible FL-2201 12V Micro pulverizador eléctrico bomba de agua de diafragma bomba de refuerzo autocebante.

La micro bomba de diafragma CC combina los mejores aspectos de bomba autocebante y bomba química, adopta una variedad de materiales importados de síntesis de resistencia a la corrosión, tiene la función de auto cebado, protección térmica, funcionamiento estable, puede ralentí continuamente durante mucho tiempo, puede ser un funcionamiento de carga continua durante mucho tiempo, Buen efecto de atomización, corriente pequeña, bajo ruido, peso ligero, fácil de mover.

La medida de la bomba con respecto a la entrada y salida de la manguera es de 3/8”.

- Potencia: Electric
- Estructura: Submersible Pump
- Combustible: electric
- Estándar: Standard
- Aplicación: Submersible
- Componente principal: submersiblepump
- Uso: Water
- Presión: Low Pressure
- Número de modelo: FL-2201
- voltage: 12V
- Type: Diaphragm
- pipe size: 9.5mm
- weight: 0.62kg
- amp: 1.8a
- total head: 30m
- pressure: 55psi
- flow: 3L/m
-

Para el accionamiento de la bomba utilizamos el tercer módulo Fuente regulada basada en LM2596 tipo Step Down, el cuál es alimentado por la batería con la capacidad de entregar 6V, hasta 2A y su salida dependerá de la placa de relé de dos canales activado por la Raspberry Pi.

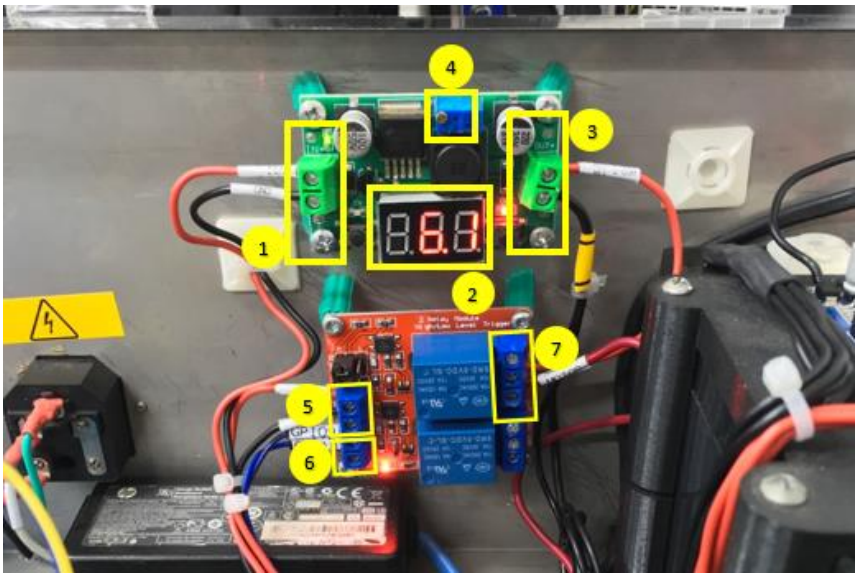


Figura 24. Módulo # 3 LM2596 y Módulo de relé de dos canales.

Donde:

1. Voltaje de entrada 12V desde la batería
2. Display 7 segmentos que muestra el voltaje de la salida
3. Voltaje de Salida 6.1V para el canal uno del módulo de relé
4. Potenciómetro para regular en voltaje de salida

5. Alimentación al módulo de relé de dos canales con 5.3V desde módulo#2 LM2596
6. Pines de activación y desactivación de la bomba desde de la Raspberry Pi 32 B+ al módulo de relé
7. Salida del módulo de relé para activar la bomba

Mangueras: Las que se utilizaron para el prototipo son de tubo neumático flexible de nailon azul de 12 mm para línea de aire, aceite y combustible; el cual succionará y suministrará el Superkote2000 al recipiente indicado.

Las características:

- Material: nailon resistente a los rayos UV PA12, polímero virgen
- Especificaciones: BS5409, SAE J1394, ISO7628
- Presión de trabajo: 12 bares
- Temperatura de funcionamiento: -40 °C a +80 °C
- Resistente al aire, combustible, aceite y abrasión

En la figura No. 24 se muestra el tipo de manguera a utilizar

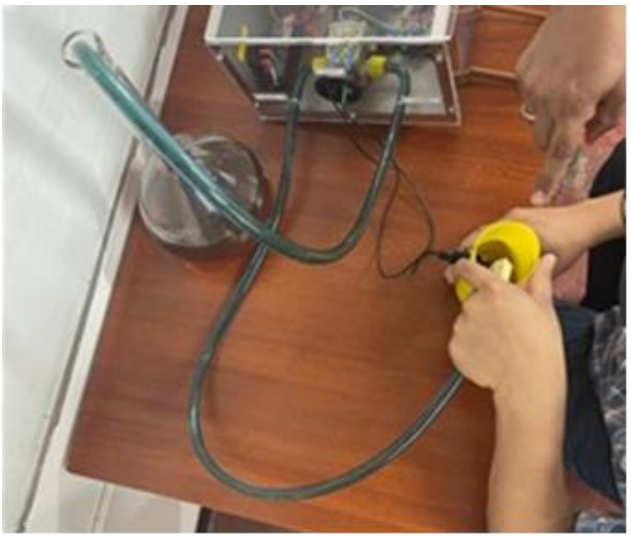


Figura 25. Manguera BS5409, SAE J1394, ISO7628

Sensores de nivel horizontal para líquidos: El sensor de nivel utilizado es de la serie LS, de interruptores de nivel de líquido, este demuestra un alto grado de confiabilidad debido al uso de componentes mojado no reactivo y un interruptor de lengüeta único diseñado específicamente para aplicaciones de detección de nivel. El sensor utiliza un flotador en movimiento con un imán incrustado para activar un interruptor de lengüeta ubicado en el cuerpo del sensor.
A medida que el nivel del líquido eleva el flotador, se aproxima al interruptor de lengüeta y lo acciona para dar un contacto abierto o indicación de interruptor de contacto cerrado; se instala desde el exterior y no necesita ser sellado.
Las características

- Voltaje del funcionamiento: 220V AC, 110V DC
- Tensión de ruptura (Máx.): 220V DC
- Corriente de conmutación (Max): 0.5A
- Corriente de carga (Máx.): 1A
- Potencia de funcionamiento (Máx.): 10W
- Resistencia máxima (Máx.): 100 ohm
- Temperatura de trabajo: -10 ~ +85°C
- Contactos: NO / NC
- Presión de funcionamiento (Max): 1.0 MPA
- Material: Plástico
- Longitud del cable: 40cm
- Dimensiones: 17x85mm
- Peso: 14g

En la Figura No. 25 podemos ver la imagen del sensor y en la Figura No. 26 el sensor instalado en el recipiente que será utilizado para la calibración.

Figura 26. Sensor de nivel

Figura 27. Sensor instalado en el prototipo

Se logró diseñar e implementar un prototipo de Controlador Electrónico que cumpla con los requerimientos de la Distribuidora de Productos y Servicios S.A, este se evaluará para ver si el sistema cumple con las especificaciones requeridas.

En Anexo 1 se muestra el Diagrama esquemático del circuito electrónico del Controlador desarrollado en el software de diseño de circuitos, Easily Aplicable Graphical Layout Editor (Eagle), siendo Eagle un potente programa para diseñar circuitos impresos y realizar esquemas electrónicos. Gracias a este programa se logra diseñar esquemas y placas de circuito impreso con AutoRoute, es decir con la función que automatiza el dibujo de pistas en la placa de circuitos impresos, y todo esto en un entorno ergonómico.

En la figura No. 27, se puede observar el prototipo del Controlador Electrónico a utilizar y las indicaciones de sus componentes.

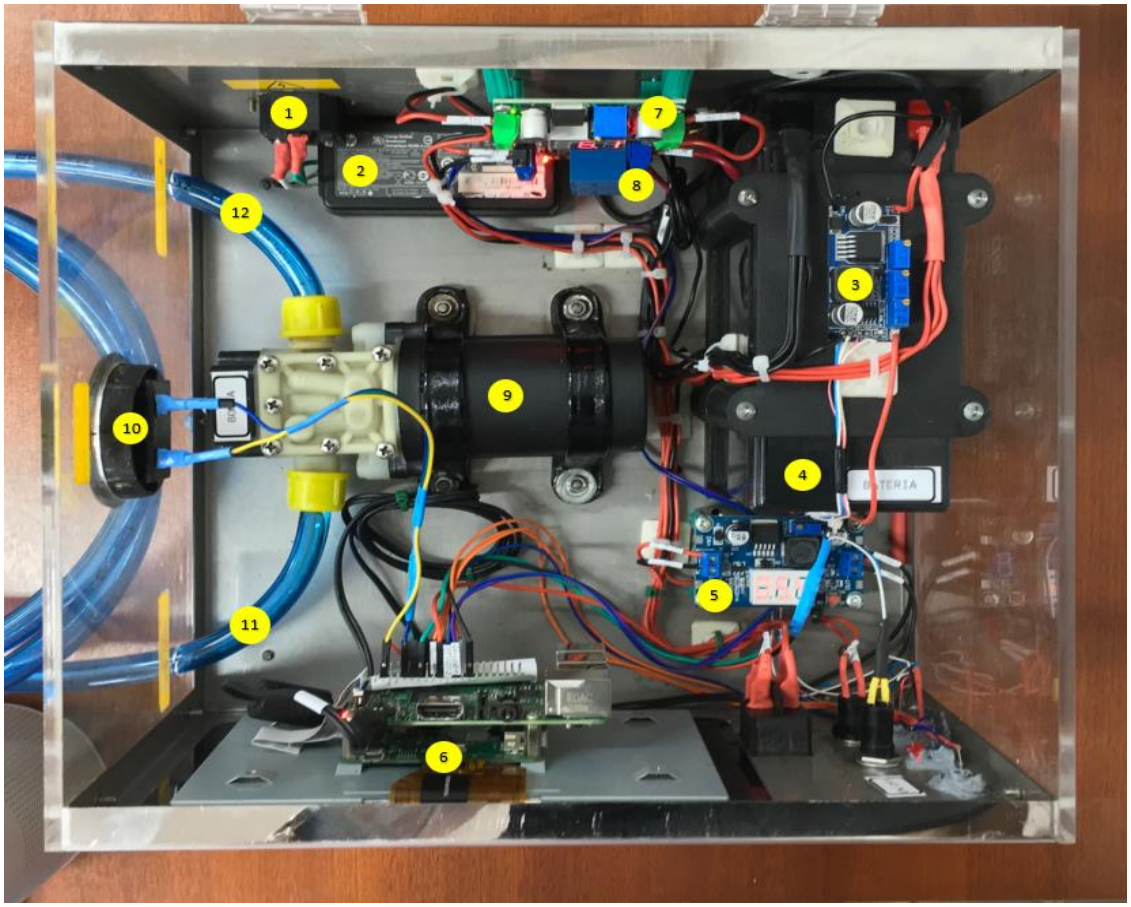


Figura 28. Controlador Electrónico a utilizar donde se indican sus componentes

Donde:

1. Suministro Comercial 120V AC
2. Fuente de alimentación del 110V -240V a 19.5V-2.5 A
3. Módulo#1 LM2596
4. Batería 12V 7Ah
5. Módulo#2 LM2596
6. Raspberry Pi 3 B+
7. Módulo#3 LM2596
8. Módulo de relé de dos canales
9. Bomba o Motor
10. Conector para el sensor de nivel
11. Manguera de succión de Superkote 2000
12. Manguera de dosificación de Superkote 2000

Para sujetar el Controlador Electrónico y sus componentes se procedió a elaborar la carcasa, en Anexo 2 se encuentran los planos de dicha carcasa y en Anexo 3 se encuentra el plano de Soporte superior e inferior de la batería.

4.2.3 Desarrollo experimental

4.2.3.1 Pruebas del consumo del Controlador Eléctrico

Se realizaron distintas pruebas y mediciones al prototipo del Controlador electrónico, con la finalidad de determinar el rango ideal de funcionamiento de la bomba para la dosificación del Superkote2000, tomando como variables la corriente, voltaje y tiempo de suministro en envases de 4 onzas.

Las pruebas consistieron:

Primero:
Ajustar el potenciómetro a la salida de voltaje del módulo LM2596 encargado de alimentar a la bomba, según las especificaciones del fabricante, el rango de operación de la bomba está entre 4V hasta 12V, corriente directa.

Segundo:
suministrar Superkote2000 tratamiento para metales dentro del recipiente de calibración y comprobar que el sensor realice el corte de suministro dentro del recipiente cuando este alcance el nivel de cuatro onzas. Seguido de eso, se almacena el tiempo de suministro en la variable *t_suminist* y se realizan el llenado de 2 suministros dentro de 2 probetas estandarizadas de 250 mililitros, para verificar que la cantidad de Superkote2000 tratamiento para metales sea equivalente a 118ml (según la conversión de onzas a mililitros).

De igual manera, se adquirieron los datos de consumo de corriente eléctrica de la bomba y la Raspberry Pi 3B+, los cuales representan los componentes de mayor demanda del prototipo del controlador.

Se efectuaron siete pruebas, de las cuales el mejor comportamiento se observa cuando el ajuste del voltaje del módulo LM2596 está a 6V, el tiempo de suministro del consumo de corriente eléctrica de la bomba es de 1.4763A, la Rapsberry Pi3 B+ se mantiene constante con un consumo de corriente eléctrica de 0.8858A.
El tiempo de desplazamiento de la bomba es de 7.180 segundos desde el accionamiento de la bomba hasta la activación del nivel del sensor y, la verificación de la exactitud del suministro en las probetas uno y dos es de 118ml.

En la figura No. 29 se muestran los resultados de las 7 pruebas realizadas y los resultados de cada una de ellas.

Voltaje DC LM2596 #3 (V)	4.5	5	6	7	8	9	10
Consumo controlador suministrando (A)	1.3748	1.4264	**1.5585**	1.6066	2.0127	2.3529	2.7817
Consumo Raspberry Pi3 (A)	0.4606	0.4606	**0.4606**	0.4606	0.4606	0.4606	0.4606
Consumo Bomba Suministrando (A)	0.9142	0.9658	**1.0979**	1.146	1.5521	1.8923	2.3211
Tiempo suministro (seg)	9.315	9.048	**7.18**	6.547	6.217	5.841	5.731
Llenado 1 (mililitros)	120	118	**118**	120	120	126	126
Llenado 2 (mililitros)	120	120	**118**	120	122	124	126

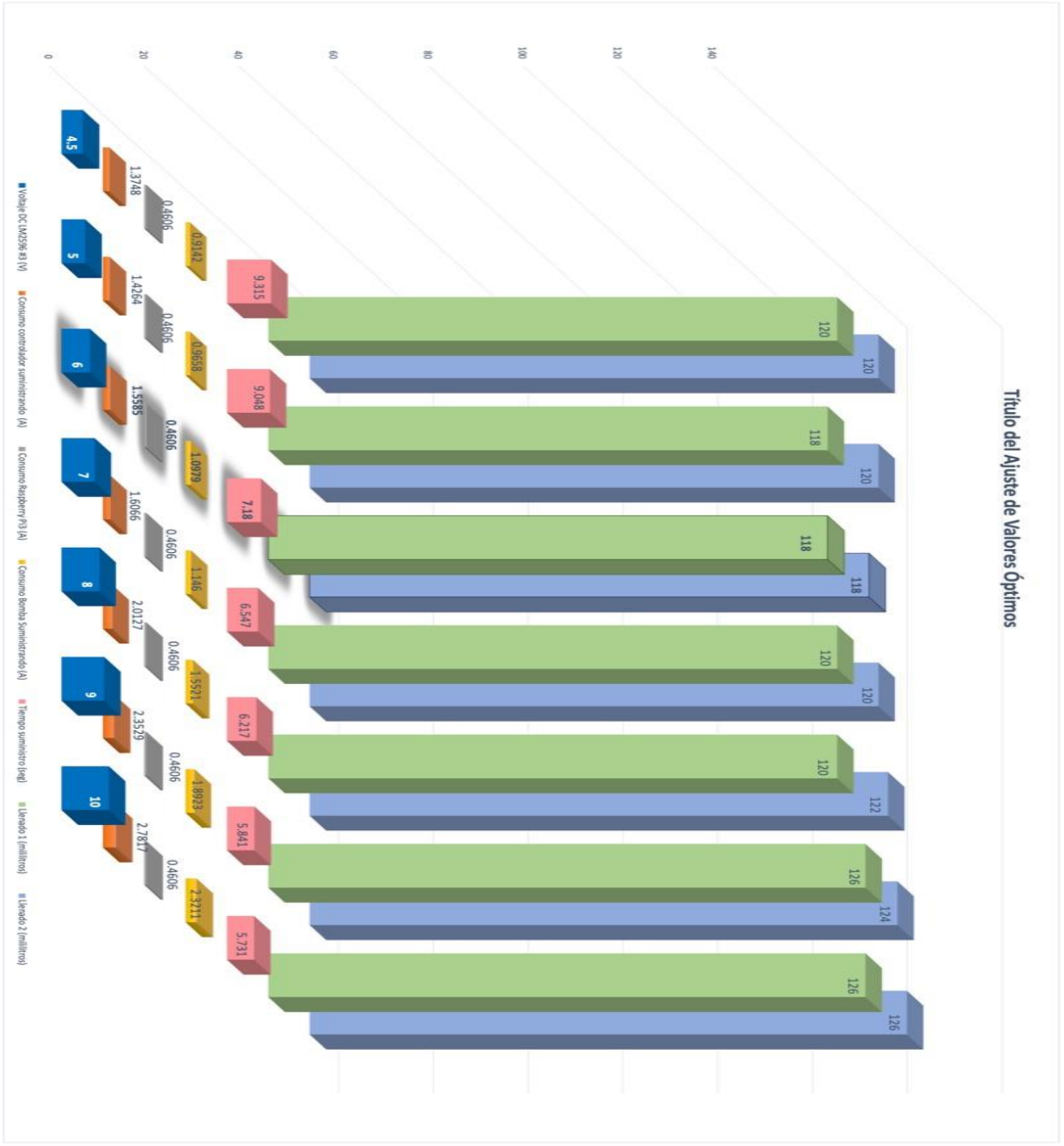


Figura 29. Pruebas realizadas con el Controlador Electrónico para suministro de Superkote 2000

4.2.3.2 Prueba para determinar el tipo de Bomba y Sensor a utilizar

El tratamiento para metales Superkote 2000 cuenta con una viscosidad alta debido a su composición y para lo que es aplicado, el cual es un fluido que se adhiere a las paredes del motor para disminuir la fricción entre las piezas.

Para determinar el tipo de bomba a utilizar es necesario determinar un parámetro importante siendo éste la viscosidad del líquido.

Se realizaron pruebas en el laboratorio de alimentos de la Facultad de Ingeniería Química en la Universidad Nacional de Ingeniería, para determinar la viscosidad del Superkote 2000 tratamiento para metales, a continuación, se describen las pruebas efectuadas con la asesoría del responsable de Laboratorio, Ingeniero Christian Sánchez.

Los elementos utilizados son:

- Superkote 2000 tratamiento para metales
- Bureta graduada con llave de pase

4.2.3.2.1 Prueba de viscosidad

La bureta es uno de los materiales de laboratorio usados en química, es fabricado de un material de vidrio de medición volumétrico empleado para la dosis exacta de un líquido, principalmente de uno de los reactivos en una titulación. El tubo de la bureta cuenta con marcas graduadas que sirven para determinar el volumen dispensado del líquido que tiene una llave de paso en su extremo inferior, acompañado con un tubo capilar afilado en la salida de la llave de paso. La llave de paso es quien controla el flujo de líquido desde el tubo a la punta de la bureta.

Figura 30. Ingeniero Sánchez responsable de Laboratorio

Para realizar la prueba de viscosidad se calcula primero el tiempo de disipación del Superkote 2000 y el tiempo de disipación del líquido de referencia(agua), la fórmula aplicada para el cálculo es:

$$Viscosidad = \frac{t_f\ superkote2000}{t_f agua}$$

Ecuación 3. Fórmula para cálculo de la Viscosidad.

Para ello se realiza el llenado de la bureta con capacidad de 25ml, tomando como muestra 1ml del Superkote 2000 para calcular el tiempo de disipación. El resultado obtenido es de 100 segundos desde el tiempo inicial, hasta el tiempo final de disipación del Superkote 2000.

Superkote2000

Vo= 0 ml
Vf= 1 ml
ti= 0 seg
tf= 100 seg

Figura 31. Llenado de Bureta con Superkote2000

También se realiza este mismo ejercicio con un líquido de referencia siendo éste el agua y tomando de muestra 1ml para calcular el tiempo de disipación. El tiempo de inicial hasta el tiempo de disipación fue de 2.64 segundos.

Agua

Vo= 0 ml
Vf= 1 ml
ti= 0 seg
tf= 2.64 seg

Figura 32. Prueba con Agua

Al tener los tiempos de disipación de ambos fluidos realizamos el cálculo de la viscosidad:

Teniendo que:

$$Viscosidad = \frac{100\ seg}{2.64\ seg}$$

Resultando que la $Viscosidad = 37.88$

Cabe señalar que la viscosidad no presenta una unidad de medida, es decir que es adimensional.

4.2.3.2.2 Medidor de conductividad

Antes de determinar el sensor ideal para la función que éste realizaría se efectuaron pruebas de conductividad en el laboratorio de Alimentos de la Facultad de Ingeniería Química.

A continuación, describimos el equipo para medir la conductividad y a luego una tabla con las mediciones arrojadas.

Elemento a utilizar es:

- Conductímetro

El medidor de mesa de calidad del agua mide todos los siguientes parámetros:

- pH
- mV
- Conductividad
- TDS (sólidos totales disueltos)
- Salinidad
- Temperatura

Carcasa resistente con pantalla LCD que muestra el parámetro que se está midiendo junto con la hora, la fecha y la temperatura (en ºC o ºF).

- Compensación de temperatura automática o manual
- Rango automático o manual
- Reconocimiento automático de búfer
- Salidas digitales y analógicas
- Min / Max y 99 puntos de datos
- Retención de datos
- 5 puntos de calibración
- Indicación del estado del electrodo

Funciona con voltaje CA (sin baterías). Viene listo para usar con electrodo de pH ATC, brazo porta electrodos, adaptador de CA, cable de computadora y software. También acepta cualquier sonda de pH u ORP con un conector BNC estándar.
Dimensiones: 217 x 168 x 58 mm (8½ "x 6½" x 2¼ "). Peso: 18 oz (533 g)

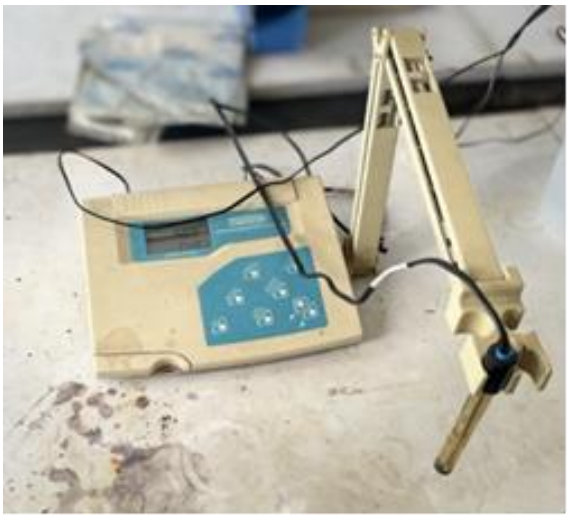


Figura 33. Medidor de conductividad

Se llenó un matraz de Erlenmeyer en el que se inserta, deposita o introduce la sonda que contiene los electrodos para realizar la medición; resulto no ser conductivo por el valor medido arrojado por el equipo.

Figura 34 Medidor de conductividad

A continuación, en la tabla 1 se describen los resultados de las pruebas realizadas para Superkote 2000 y para Agua.

Tabla Prueba de conductividad eléctrica			
Fluido	**Superkote2000**		**Agua**
Equipo	Conductivity Benchop	PH Conductivity Meter	PH Conductivity Meter
Marca	Thermo Scientifik	OAKTON	OAKTON
Modelo	Orion Star Series	510 series	510 series
Temperatura	28.3 °C	28.2 °C	28.2 °C
Conductividad	0.00us/cm aprox 10ps/cm	0.00us/cm aprox 10ps/cm	441us/cm

Tabla 1.Prueba de Conductividad Eléctrica

4.3 Desarrollo de Interfaz gráfica de programación.

4.3.1 Lógica de programación

Consiste en escribir las instrucciones para que la computadora realice una tarea; se suele decir que la computadora resuelve problemas, pero lo correcto es decir que la computadora ejecuta las instrucciones que resuelven el problema.
Basándose en los requerimientos de la empresa Distribuidora de Productos y Servicios, se requiere suministrar con exactitud el líquido Superkote2000 para realizar series de llenado en envases que la empresa dispone. Las unidades que distribuyen corresponden a 4 onzas y 8 onzas. Se requiere utilizar tecnologías emergentes de bajo costo para la empresa, pero con precisión en el llenado. Al mismo tiempo se busca un controlador que posea autonomía energética y permita realizar llenados en lugares con problemas de acceso a energía eléctrica.

Se desarrolló una lógica de programación secuencial, para ello se definen las variables de entradas, salidas requeridas y posteriormente definir la programación a emplear.

Entradas requeridas:

- Activación del sensor horizontal de llenado que enviará señal a la Raspberry Pi 3B+ y detendrá la bomba cuando se alcance el límite según recipiente de calibración.
- Seleccionar tiempo de espera entre envases.
- Digitar cantidad de envases a llenar.
- Botón de emergencia para detener el suministro.

Salidas deseadas:

- Cantidad de líquido exacto (4, 8 onzas).
- Diodos leds indicando estado del controlador.
- Diodos leds indicando estado de carga del controlador.

Procesos:

- Almacena tiempo calculado según envase de calibración.
- Almacena tiempo de espera entre envases.
- Almacena cantidad de envases a llenar.
- Realiza validación de ingreso de datos.
- Procede al llenado con los datos almacenados en las variables:
  input_state (sensor nivel de líquido)
  input_stop (botón emergencia)
  t_suminist (tiempo de suministro)
  TW (tiempo de espera entre envases)
  qty_env (cantidad de envases a llenar

4.3.2 Diseño del Algoritmo

El algoritmo[21] es un método para resolver un problema, aunque la popularización del término ha llegado con el advenimiento de la era informática, algoritmo proviene de Mohammed al- KhoWaîrizmi, matemático persa que vivió durante el siglo IX y alcanzó gran reputación por el enunciado de las reglas paso a paso para sumar, restar, multiplicar y dividir números decimales; la traducción al latín del apellido en la palabra algorismus derivó posteriormente en algoritmo. Euclides, el gran matemático griego (del siglo IV a. C.) inventó un método para encontrar el máximo común divisor de dos números, se considera con Al-Khowaîrizmi el otro gran padre de la algoritmia (ciencia que trata de los algoritmos).

En dicho algoritmo se detallan las instrucciones para realizar una tarea, en donde se presenten alternativas de solución, este debe ser preciso y el resultado depende de los datos suministrados siempre que el algoritmo se ejecute con un mismo conjunto de datos de entrada, el resultado debe ser siempre el mismo, para ello se hace uso de diagrama de flujo, luego se verifica si el diagrama de flujo cumple las características del algoritmo.

Para la debida ejecución del algoritmo se deben incorporar diferentes datos de entrada para verificar si se obtienen los datos de salida esperados, en caso contrario se procede a hacer las modificaciones necesarias al algoritmo.

4.3.2.1 Diagrama de Flujo[22]

Es una de las técnicas de representación de algoritmos más antigua y a la vez más utilizada, aunque su empleo ha disminuido considerablemente, sobro todo, desde la aparición de lenguajes de programación estructurados. Un diagrama de flujo es un diagrama que utiliza los símbolos (cajas) estándar mostrados en la Tabla 2, contiene los pasos de algoritmo escritos en esas cajas unidas por flechas, denominadas líneas de flujo, que indican la secuencia en que se debe ejecutar.

En este diagrama se describen los diferentes pasos que representa el algoritmo que se usará para el llenado de envases de SUPERKOTE 2000, representa los pasos a seguir, el mecanismo de control y descripción de procesos, la correcta construcción es muy importante porque a partir de este se escribe en lenguaje de programación, además que ofrece una herramienta básica dentro de la empresa Distribuidora de Productos y Servicios S.A. (DPS).

[21] L. Joyanes Aguilar, Fundamentos de Programación, Madrid: Mc Graw Hill, 2008.
[22] L. Joyanes Aguilar, Fundamentos de Programación, Madrid: Mc Graw Hill, 2008.

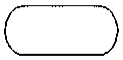
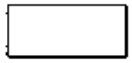
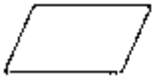
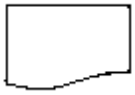
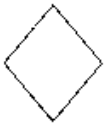
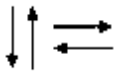
Símbolo	Descripción
	Símbolo terminal, utilizada para indicar el inicio y el fin del algoritmo.
	Símbolo de proceso, representa una operación sobre datos, tal como un cálculo, ejemplo operaciones aritméticas, asignación de valor de una variable.
	Símbolo de entrada de datos, representa datos a ser leídos.
	Símbolo de documento, se utiliza para representar salida de datos.
	Símbolo de proceso predeterminado, se utiliza para representar salida de datos (escritura)
	Símbolo de decisión, representa el cambio de curso de acción del algoritmo.
	Flujos, conecta todos los símbolos en el diagrama e indica el orden en que va a ser realizadas las instrucciones.
	Conector, conecta una parte de un diagrama de flujo con otra en una misma página.
	Símbolo de conexión entre página, utilizado para conectar una parte de un diagrama de flujo con otra en páginas diferentes.

Tabla 2. Simbología para Diagrama de Flujo

El diagrama de flujo que se detalla en la figura No 35 se presenta e indica el orden de realización del llenado de los envases para el SUPERKOTE 2000.

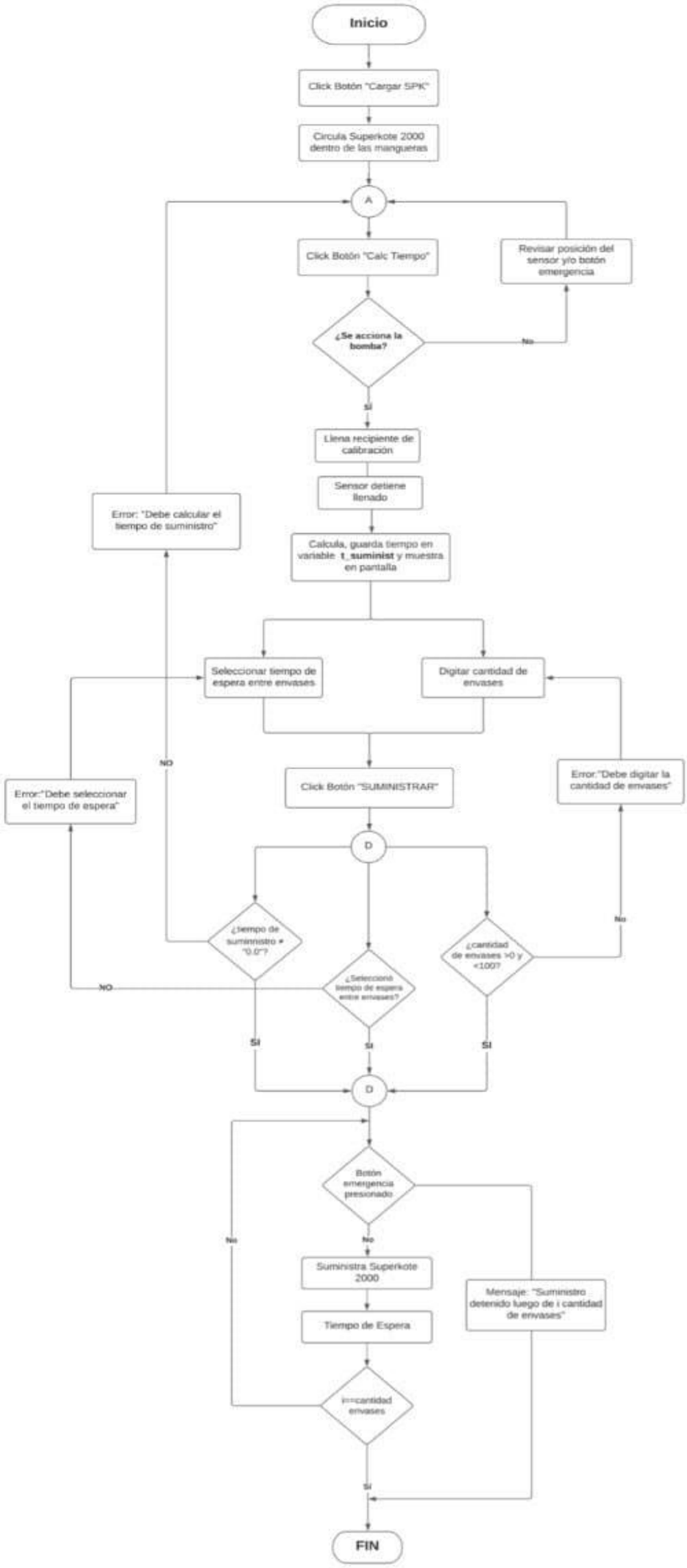


Figura 35. Diagrama de Flujo

4.3.3 Programación

Cuando el procesador es una computadora, el algoritmo se ha de expresar en un formato que se denomina programa, ya que el seudocódigo o el diagrama de flujo no son comprensibles por la computadora, aunque pueda entenderlos cualquier programador. Un programa se escribe en un lenguaje de programación y las operaciones que conducen a expresar un algoritmo en forma de programa se llaman programación. Así pues, los lenguajes utilizados para escribir programas de computadoras son los lenguajes de programación y programadores son los escritores y diseñadores de programas. El proceso de traducir un algoritmo en seudocódigo a un lenguaje de programación se denomina codificación, y el algoritmo escrito en un lenguaje de programación se denomina código fuente.

4.3.3.1 Visual Studio Code[23]

Es un editor de código fuente ligero pero potente que se ejecuta en su escritorio y está disponible para Windows, macOS y Linux. Viene con soporte integrado para JavaScript, TypeScript y Node.js y tiene un rico ecosistema de extensiones para otros lenguajes (como C ++, C #, Java, Python, PHP, Go) y tiempos de ejecución (como .NET y Unity),

4.3.3.2 Python

Python[24] es un lenguaje de programación de computadoras que nació en 1991 y que ha ido ganando adeptos por dos razones principales. La primera es que es un lenguaje de alto nivel muy fácil de usar y la segunda es que es código libre. Fue concebido y desarrollado por Guido von Rossum y a la primera versión 0.9.0, estuvo disponible en 1991. Actualmente, las versiones más usadas son la versión 2 y la versión 3.
Una de las grandes ventajas de Python es que es independiente de la plataforma que use, ya sea Windows, Mac o Unix. Los programas escritos en una plataforma corren en las otras plataformas. El único requisito es que haya sido realizado en la misma versión.

Python tiene disponible una biblioteca estándar que le da al usuario una gran cantidad de funciones para escribir código fuente muy simple pero poderoso. Además, otros usuarios han desarrollado módulos, la mayoría de los cuales son de código abierto (open source) y gratuitos. Estos módulos se pueden usar para desarrollar interfaces gráficas, interfaces a base de datos, aplicaciones de páginas de Internet, entre otras.

En Anexo No. 4 se detalla la Programación del Controlador para Superkote 2000 y en Anexo 5. Descripción del Controlador para Superkote 2000.

[23] https://code.visualstudio.com/docs

[24] Ofelia Delfina Cervantes Villa Gomez, José Miguel David Baéz Lopez, Antonio Arízaga Silva, Esteban Castillo Juárez., Python con aplicaciones a las matemáticas, ingeniería y finanzas, Ciudad de México: Alfaomega Grupo Editor, 2017.

4.4 Implementación

Un prototipo es una versión inicial compacta de la solución o parte de la solución de un sistema construido en un período de tiempo y mejorado en varias iteraciones para probar y evaluar la eficacia del diseño general que se utiliza para resolver un problema determinado.

Los prototipos se utilizan de forma dirigida para reducir el riesgo y reducir el entorno de incertidumbre:

- La viabilidad empresarial de un producto que se está desarrollando.
- La estabilidad o el rendimiento de tecnología clave.
- La confirmación o financiación del proyecto: construcción de un pequeño prototipo de prueba de concepto.
- La comprensión de requisitos.
- El aspecto y percepción del producto, su utilización.

Un prototipo puede ayudar a construir soporte para el producto mediante la muestra de algo concreto y ejecutable a los usuarios, clientes y gestores.

Para la Distribuidora de Productos y Servicios (DPS) la implementación del Controlador Electrónico para Automatizar el suministro de tratamiento para metal SUPERKOTE 2000 es de mucha utilidad ya que mejorará sus procesos de producción al tener automatizado su ciclo de llenado de los envases, además que entregará la cantidad optima en cada envase.

Una vez diseñado, desarrollado el controlador y el algoritmo se procede a realizar las diversas pruebas con aceite y agua para determinar el funcionamiento del prototipo.

En la figura No. 36 se muestra el prototipo del Controlador Electrónico para Automatizar el suministro de tratamiento para metal SUPERKOTE 2000.

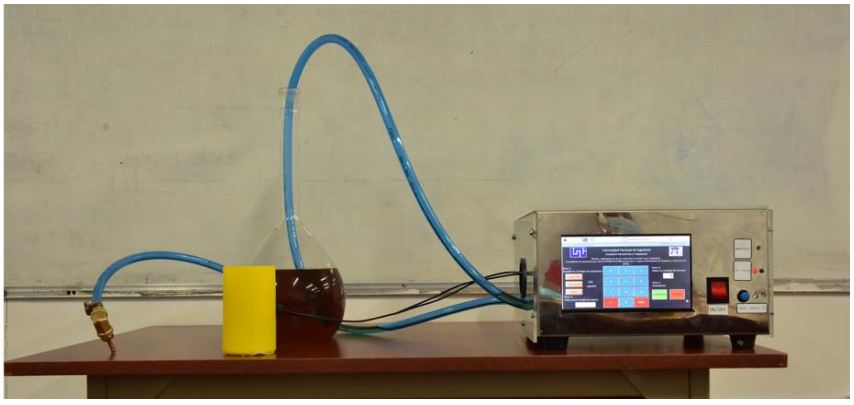


Figura 36. Controlador Electrónico para Automatizar el suministro de tratamiento para metal SUPERKOTE 2000.

4.4.1 Descripción del Funcionamiento del Controlador para la automatización del suministro de tratamiento para metal SUPERKOTE 2000

En este apartado, se explica la interacción que existirá entre la interfaz gráfica y el usuario, a manera de guía o manual para comprender el funcionamiento, los pasos a seguir y obtener los resultados.

Para iniciar el suministro de Superkote 2000 desde el Controlador Electrónico, se procede a encenderlo para cargar la interfaz gráfica de usuario, la interfaz está estructurada en cuatro partes e indica los pasos a seguir para realizar de manera correcta el suministro del líquido en los envases de las medidas establecidas por DPS.

En la figura No.37 se muestra la Interfaz gráfica de usuario para su debido uso.

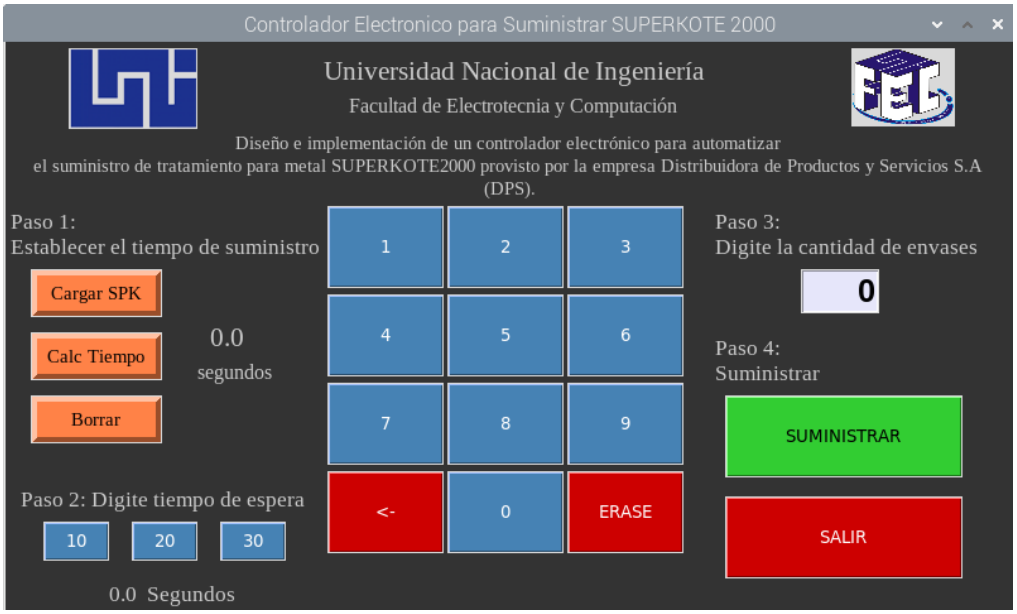


Figura 37. Interfaz gráfica de inicio.

Una vez iniciada la interfaz gráfica se deben llenar las mangueras con SUPERKOTE 2000 dando clic en el botón “Cargar SPK”, para cargar los conductos con el líquido, luego se procede a accionar el botón “Calc Tiempo” para calcular el tiempo de suministro según el volumen requerido (4 u 8 onzas), dicho paso se indica en la figura No.38

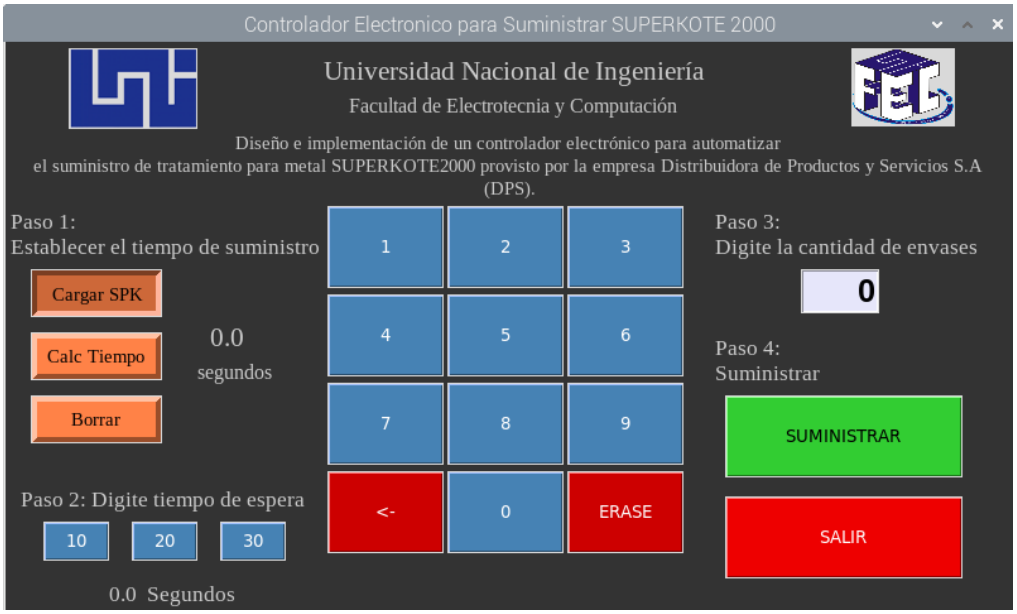


Figura 38. Cargar SPK para el llenado de las mangueras.

A continuación, se detallan los pasos para el llenado de envase de Superkote 2000

Paso 1: Establecer el tiempo de suministro.

Para calcular el Tiempo de Suministro, se lee la entrada de datos del pin GPIO4, si el sensor está en bajo (cero lógicos o en posición abierta), no conduce. Esta señal permite la activación de la bomba e iniciar a desplazar SUPERKOTE en el recipiente de calibración. Cuando el sensor está en alto (uno lógico o posición cerrada), presentará en el pin un voltaje de 3.3 voltios, indicando que alcanza el nivel deseado, lo que es utilizado por la Raspberry Pi 3 B+ para enviar un pulso por medio del pin GPIO9 hacia el módulo de relé y desactivar la bomba. Este proceso debe repetirse para cada cálculo de un nuevo tiempo de suministro. En la figura siguiente se muestra el botón Calculo de Tiempo de suministro.

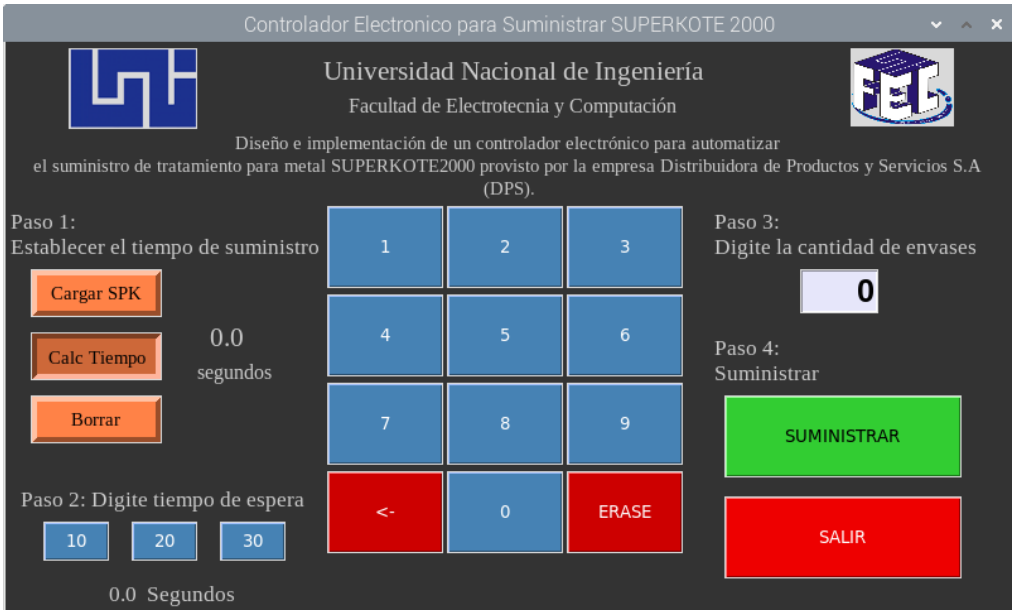


Figura 39. Cálculo de Tiempo de Suministro.

Luego de calculado el tiempo, se almacena en la variable t_suminisit (en el código de programación) y muestra al usuario ese valor en pantalla, para este caso corresponde a 13.529, ese tiempo equivale a lo que el sistema tardaría en suministrar 4 u 8 onzas del SUPERKOTE 2000.

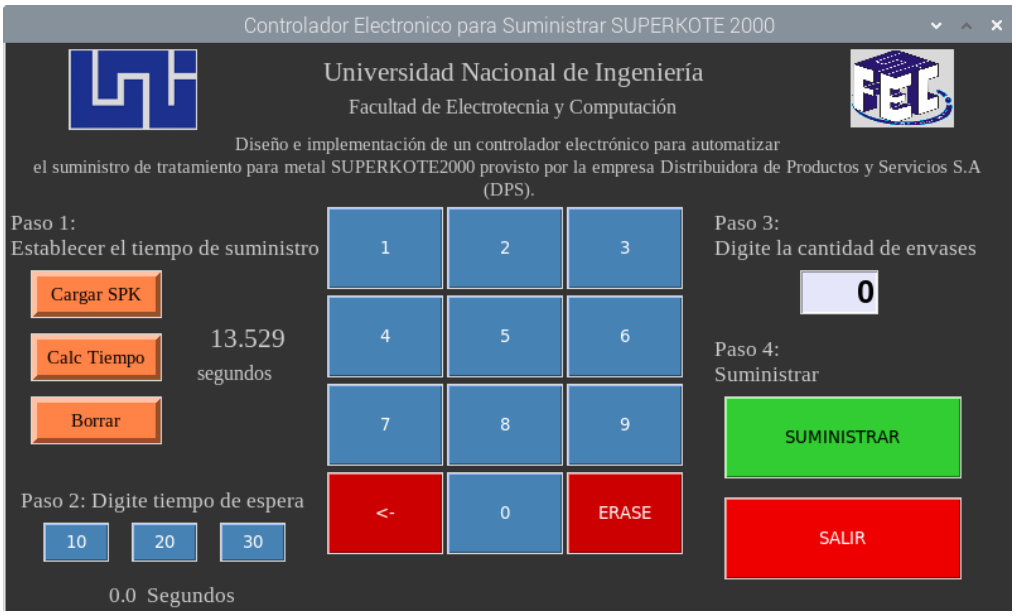


Figura 40.Tiempo de suministro.

Paso 2: Digite tiempo de espera.

Para el llenado de envases se debe seleccionar el tiempo de espera entre recipientes, este puede ser 10, 20, 30 segundos.

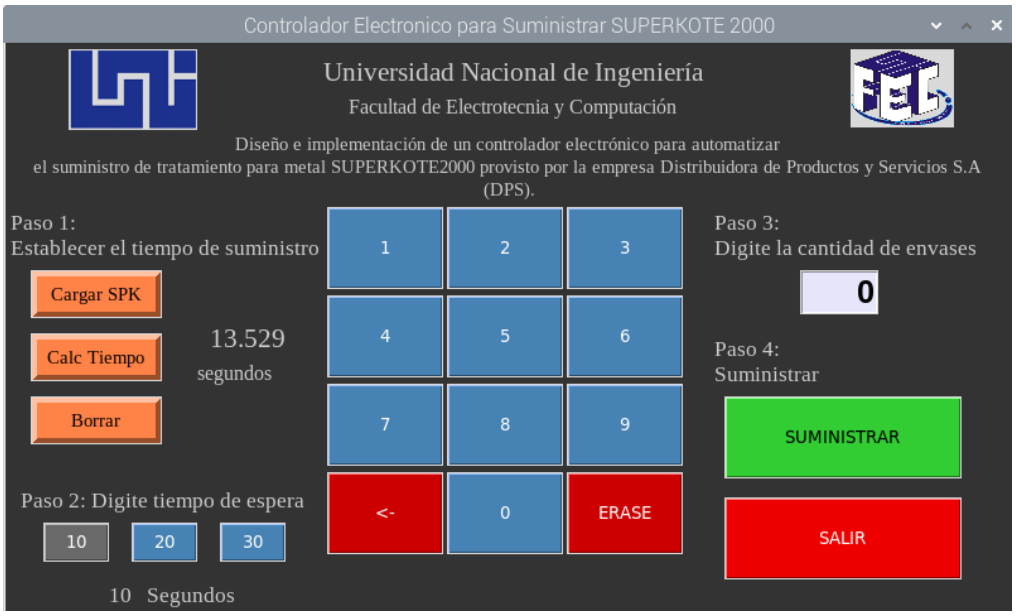


Figura 41. Tiempo de espera

Paso 3: Digite la cantidad de envases a llenar.

Una vez cumplido el paso 1 y paso 2, se procede a digitar la cantidad de envases a llenar, la cual está establecida en el rango del 1 a 99. En caso de escribir un tercer dígito, la interfaz emite un mensaje de alerta “Dígitos Permitidos 2”.

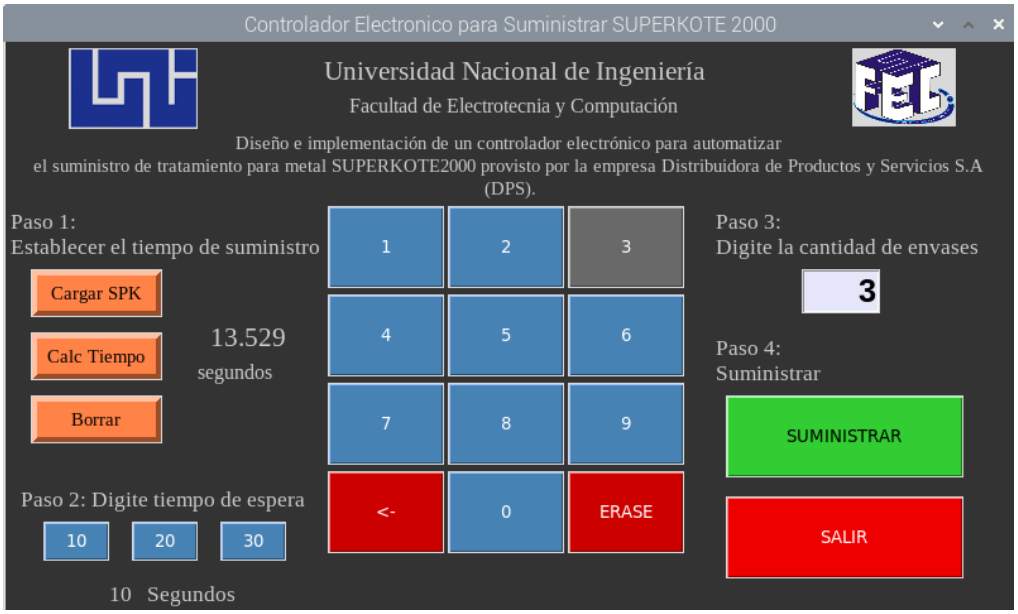


Figura 42. Digite la cantidad de envases

Paso 4: Suministrar

Al dar clic al botón Suministrar se deben de haber cumplido las condicionantes de paso 1, paso 2 y paso 3.

La interfaz está diseñada para que el usuario cumpla con todos los pasos para el suministro del Superkote2000 tratamiento para metales.

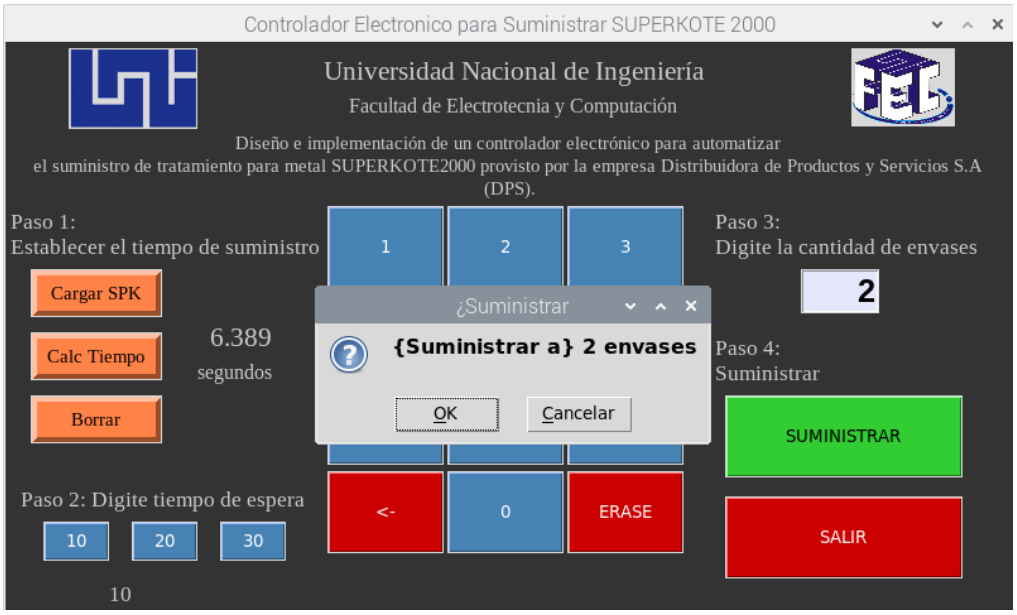


Figura 43. Confirmar Suministro

Cuando el controlador electrónico haya completado el suministro de la cantidad de envases, mostrará en pantalla el mensaje “Suministro X de X”.

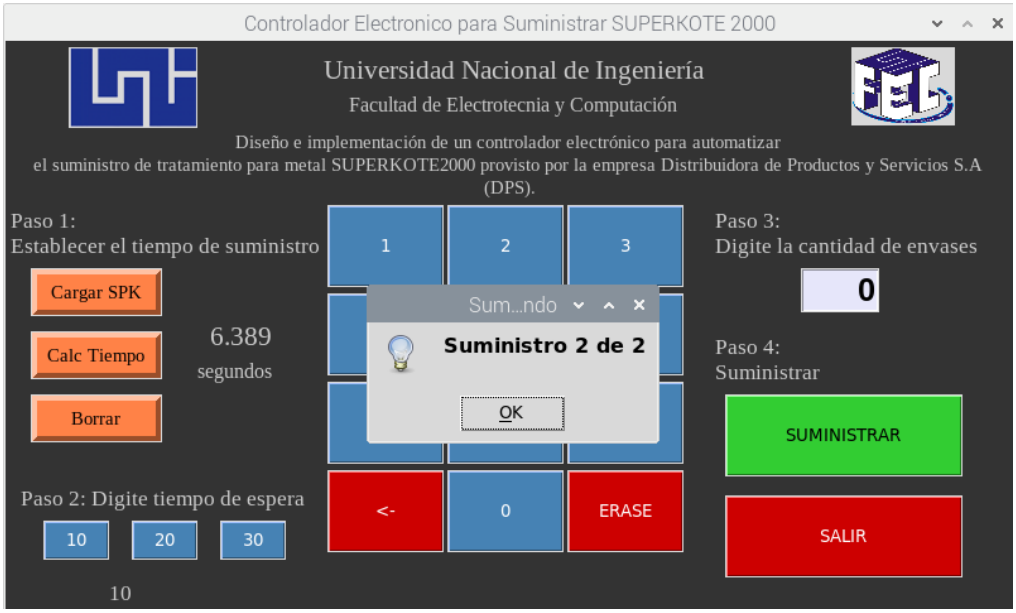


Figura 44.Suministro Completado

Si no se cumplen los pasos enviará mensaje de error, tal es el caso al no cumplir el paso 1, paso 2 y paso 3. Si el usuario por error u olvido no cumple el paso 1 “Digitar el tiempo de espera” la interfaz mostrará un mensaje emergente el cual será “Calcule el tiempo antes de suministrar”

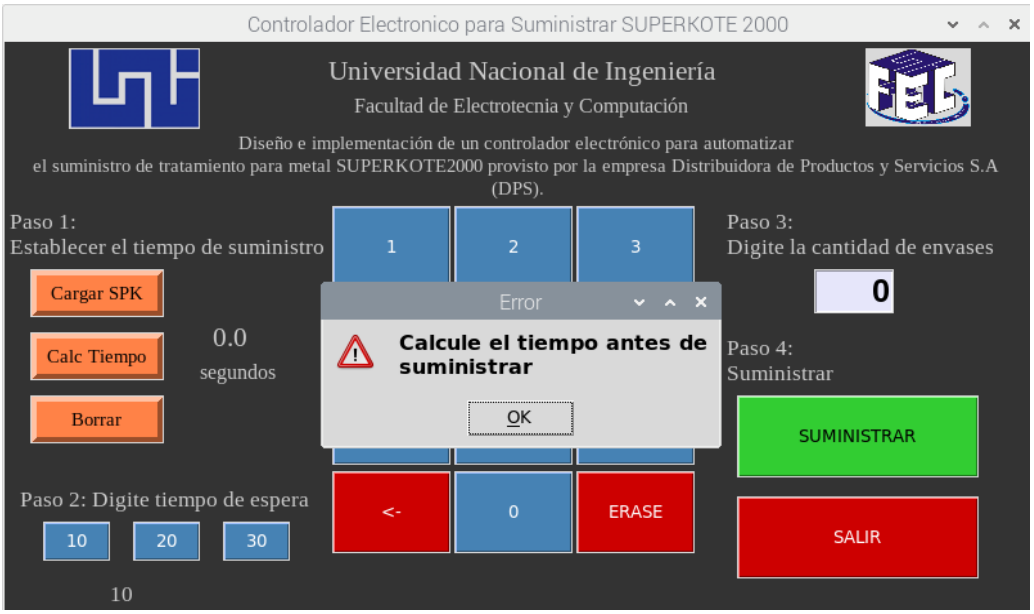


Figura 45. Mensaje emergente _“Calcule el tiempo antes de suministrar”

Si el paso 2 no se cumple envía el mensaje “Seleccione tiempo de espera”

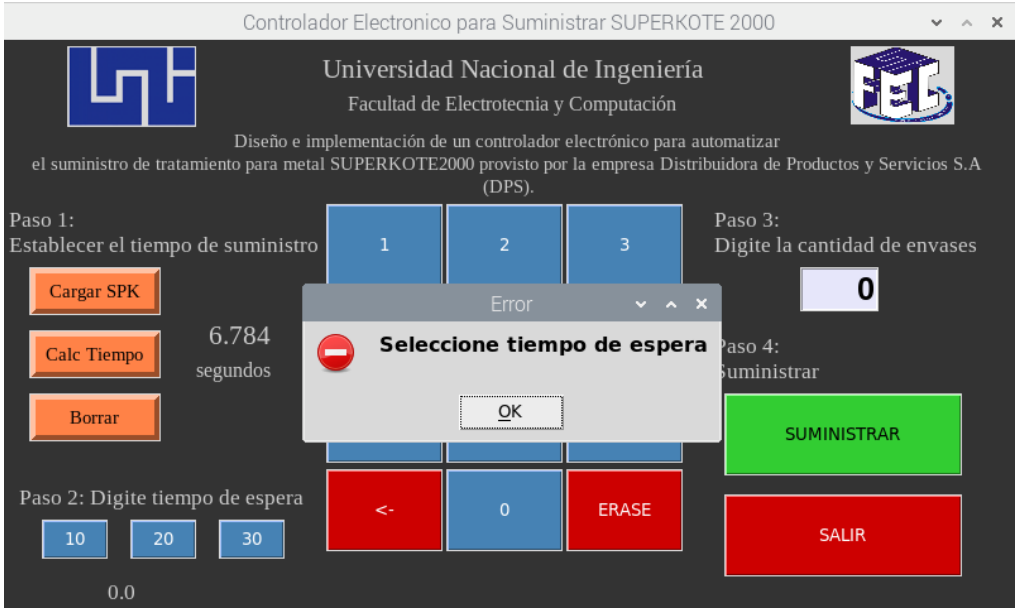


Figura 46. Mensaje_Seleccione tiempo de espera

En el paso 3 al ingresar dígitos de 3 cifras enviará mensaje “Dígitos permitidos:2” por tal razón de deben incorporar 2 dígitos, a la vez si no se incorpora la cantidad de envases a llenar debe digitalizarla.

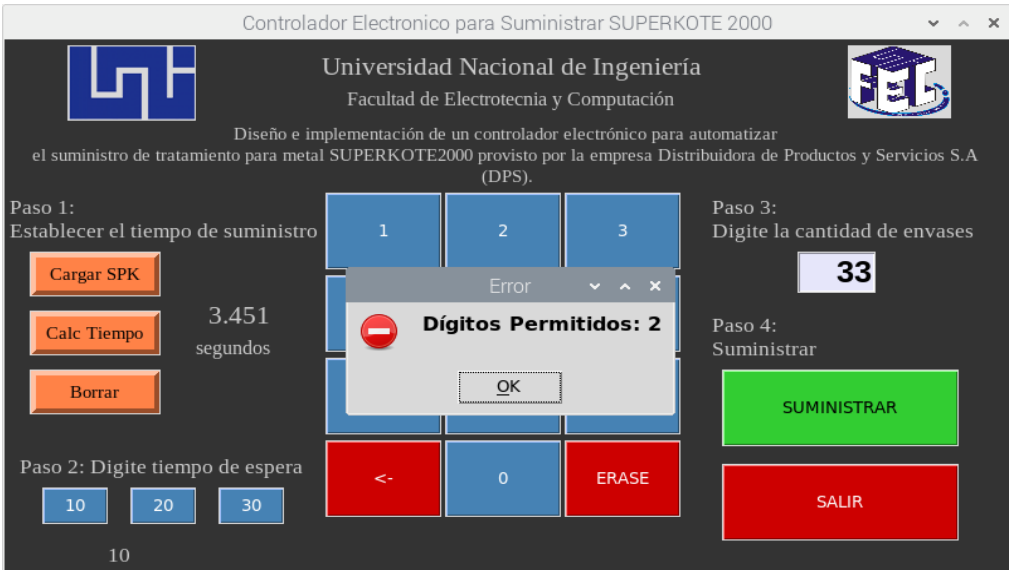


Figura 47. Mensaje_Dígitos permitidos:2

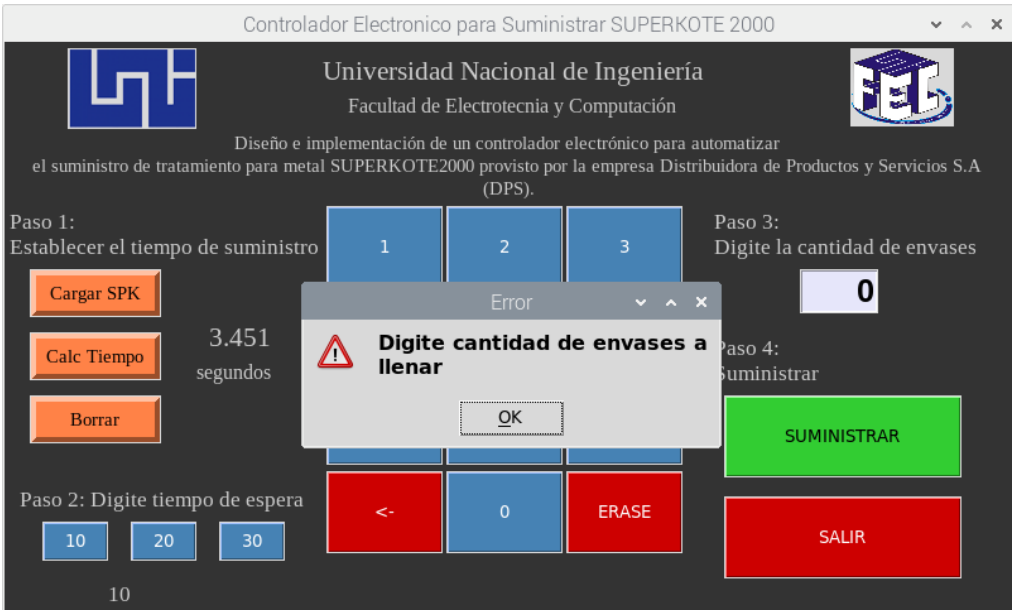


Figura 48. Mensaje_Digite la cantidad de envases a llenar

Si el usuario decide cancelar el suministro, la interfaz emitirá un mensaje indicando a partir de cual suministro se realizó la cancelación.

Figura 49. Mensaje de cancelación de Suministro

4.4.2 Diagrama de Bloques

Es la representación gráfica del desarrollo del Controlador Electrónico y la Interfaz gráfica a través de un esquema de funcionamiento para suministrar el Superkote 2000, en este se incluye terminales, funciones los cuales transfieren datos junto con otros objetos del diagrama de bloques. La ventana del diagrama contiene código de fuente gráfica, cada bloque representa una operación o una etapa completa

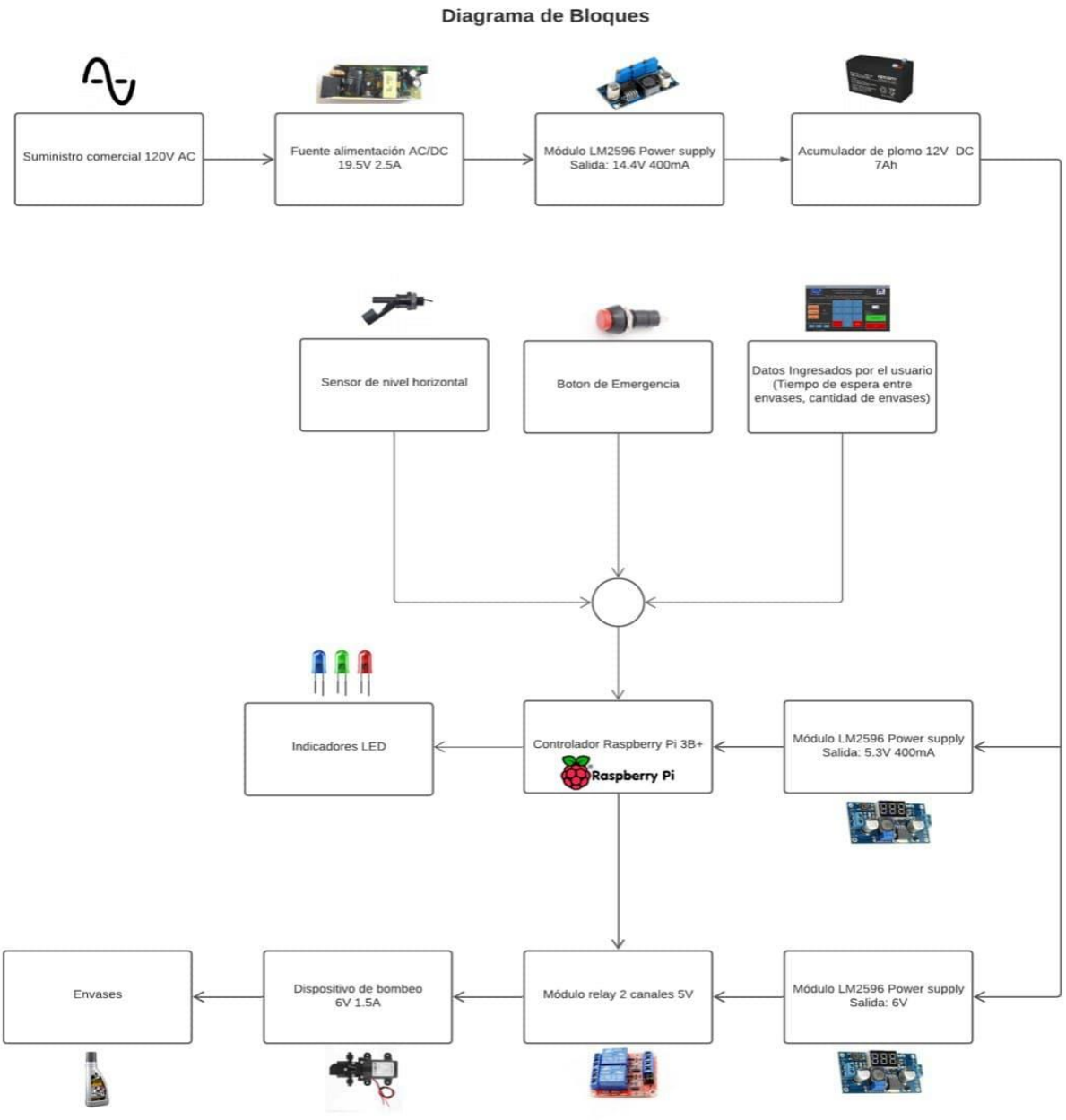


Figura 50. Diagrama de bloques.

Capítulo 5

Conclusiones y Recomendaciones

El trabajo se enfocó en dos puntos de vista, uno Proponer diseño de Controlador Electrónico considerando todas las pautas que conllevan a desarrollar un esquema que es la importancia que tiene el resultado de un buen diseño para su ejecución. El otro es la Implementación del Controlador Electrónico con la Interfaz gráfica, se tomó en cuenta el Diseño metodológico para poder llevar a efectos todas las pruebas de campo, los métodos utilizados para realizar análisis y poder

Conclusiones:

- Se ha desarrollado un controlador electrónico para automatizar el suministro Superkote 2000, utilizando componentes electrónicos, el cual ha permitido crear un algoritmo que permita ingresar parámetros para el llenado de envases.

- De acuerdo a los términos de referencia de la empresa Distribuidora de Productos y Servicios S.A. (DPS) se realizó el diseño y la programación para el suministro del Superkote 2000.

- Por medio de ejercicios y modelados se selecciona el tipo de batería a utilizar que tenga autonomía energética para que no interrumpa el funcionamiento de la empresa DPS.

- Se hace uso de una interfaz gráfica amigable en el cual se dirige el análisis del sistema, siendo esta una herramienta muy útil y sencilla de utilizar.

- Antes de declarar el prototipo funcional, este fue probado varias veces para comprobar su exactitud, se realizaron las pruebas de rendimiento y calibración para que la dosificación se exacta al tratamiento para metales propuesta por el usuario.

- A través de un prototipo se implementa la ejecución del controlador electrónico garantizando el desempeño requerido por la empresa.

- El controlador resulta viable para la empresa por la optimización en el tiempo de llenado, exactitud en la cantidad de líquido y para la dosificación de cada envase de acuerdo a los términos de referencia.

- El lenguaje de programación utilizado fue Python, ya que éste es utilizado en Raspberry por ser robusto, versátil y moderno.

Recomendaciones:

- Para llevar un control de la cantidad de envases a los que se le ha suministrado el Superkote 2000 es necesario realizar una Reporte en Excel, en donde se pueda incluir datos.

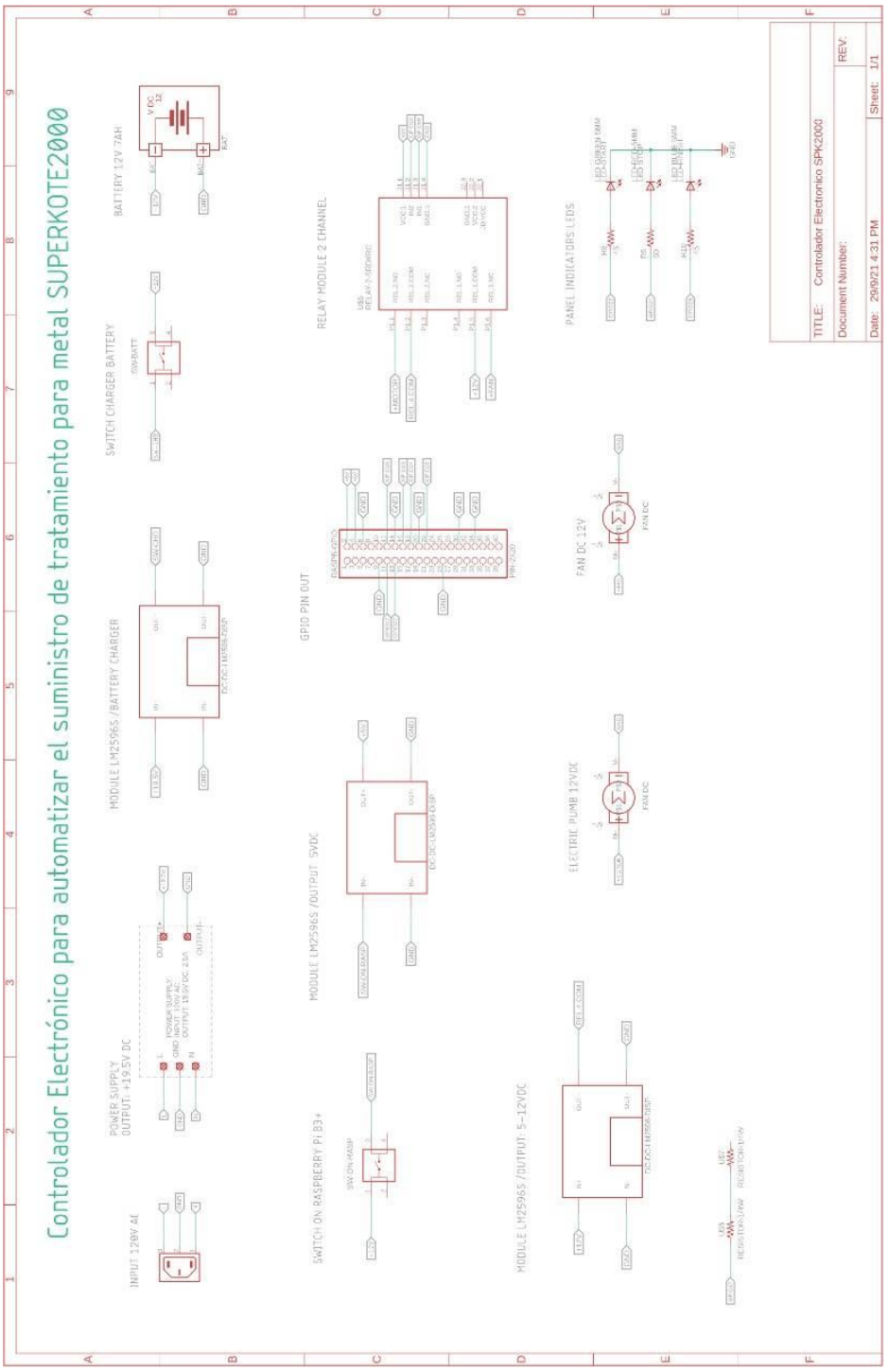
- Dado que este prototipo puede ser muy útil para desarrollar ideas de proyectos se sugiere sea incluido en el plan de estudios de la carrera de Ingeniería Electrónica.

- Se sugiere que dicha interfaz puede ser explotada para ampliar los alcances de este proyecto a fin de obtener mayores funciones.

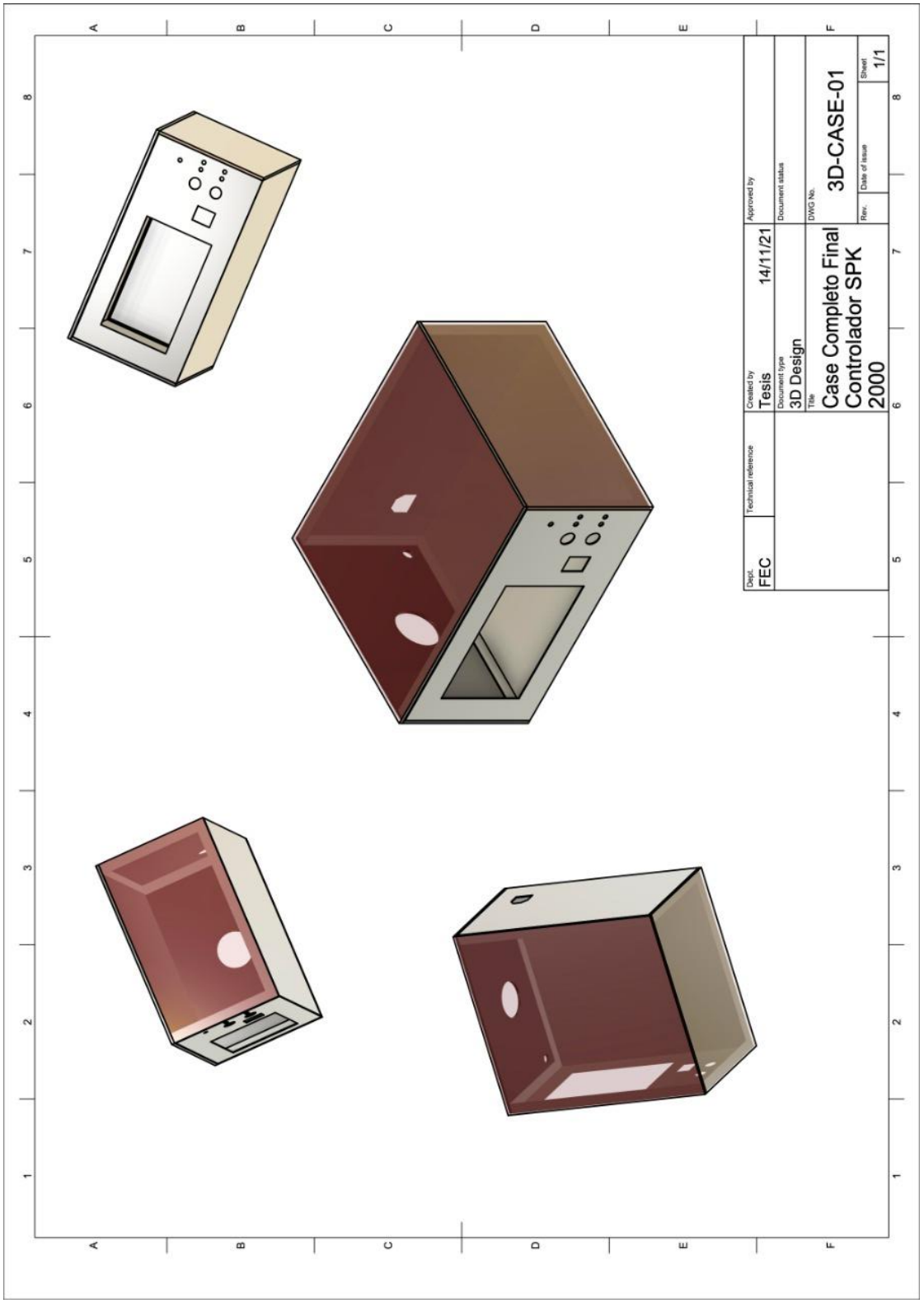
- Para el uso efectivo del Controlador Electrónico se sugiere capacitar al usuario asignado por DPS.

Anexos

Anexo No. 1 Esquema del Controlador Electrónico desarrollado y simulado en Eagle para metal Superkote 2000.

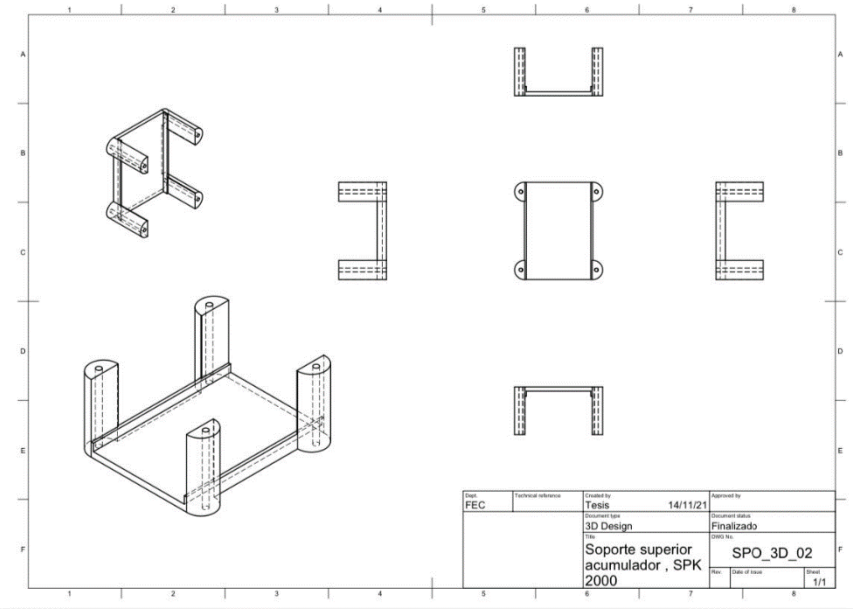


Anexo No. 2 Planos de carcasa del controlador Anexo

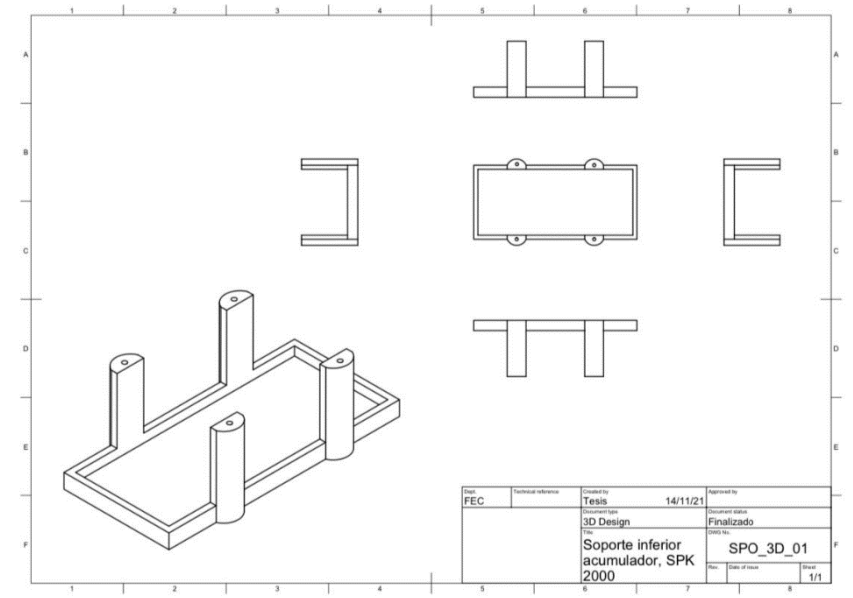


Anexo No. 3 Soporte 3D superior e inferior de la Batería

Soporte Superior



Soporte Inferior



Anexo No. 4 Programación del Controlador Superkote 2000

Controlador Electrónico SUPERKOTE2000

```
import os
from tkinter import *
from tkinter import ttk
import tkinter.messagebox
from PIL import Image, ImageTk
from time import sleep
import time
import RPi.GPIO as GPIO
Screen = Tk()
Screen.resizable(0,0)
Screen.title("Controlador Electronico para Suministrar SUPERKOTE 2000")
Screen.config(background='gray20')
Screen.geometry("800x480")
# LOGO UNI#
imguni=ImageTk.PhotoImage(file="/home/pi/uni.png")
logo_uni=Label(Screen,image=imguni)
logo_uni.pack()
logo_uni.place(x=50,y=5)
#LOGO FEC#
imgfec=ImageTk.PhotoImage(file="/home/pi/fec.png")
logo_fec=Label(Screen,image=imgfec)
logo_fec.pack()
logo_fec.place(x=670,y=5)
# Declaraciones para Botones #
width_bt=8
height_bt=3
color_bt_foreg=("white")
color_bt=("steelblue")
color_bt_func=("red3")
color_btstart=("limegreen")
color_btstart_pressed=("darkgreen")
color_btstop=("red2")
color_btstop_pressed=("red3")
color_btText=("white")
color_btActivated="dimgray"
color_txlb=("gray80")
deep=20
# Declaraciones para Etiquetas #
qty_env=StringVar()
t_wait=IntVar()
t_init=float()
t_finish=float()
t_suminist=float()
s_env=StringVar()
s_tsuminist=StringVar()
TW=float()
STW=StringVar()
# Función Setup GPIO inicio #
def gpio_init():
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(4, GPIO.IN,pull_up_down=GPIO.PUD_DOWN) #SENSOR#
    GPIO.setup(17, GPIO.IN,pull_up_down=GPIO.PUD_DOWN) #BOTON STOP#
    GPIO.setup(9,GPIO.OUT)
    GPIO.setup(10, GPIO.OUT)
    GPIO.setup(23, GPIO.OUT)
    GPIO.setup(24, GPIO.OUT)
    GPIO.setup(25, GPIO.OUT)
    GPIO.output(24,1)
    GPIO.output(9,0)
    GPIO.output(10,0)
    GPIO.output(23,0)
    GPIO.output(25,0)
def v_calib_init():
    TW=0.0
    STW.set(TW)
    t_suminist=0.0
    s_tsuminist.set(t_suminist)
```

```
# Función Limpiar valores en Pantalla #
def clear():
    global qty_env
    global l_qty_envs
    gpio_init()
    qty_env=""
    l_qty_envs=[]
    s_env.set("0")
# Función Calibración #
def calib():
    global t_suminist
    global t_finish
    global t_init
    clear()
    clear_calib()
    GPIO.output(9,0)
    t_init=time.monotonic()
    while True:
        input_sensor=GPIO.input(4)
        input_stop=GPIO.input(17)
        if input_sensor==False and input_stop==False:
            GPIO.output(9,1)
        else:
            GPIO.output(9,0)
            t_finish = time.monotonic()
            t_suminist = round((t_finish-t_init),3)
            s_tsuminist.set(t_suminist)
            break
#Función Detener Calibración#
def load_SPK():
    GPIO.output(9,1)
    sleep(10)
    GPIO.output(9,0)
# Función capturar dígito #
def digit(n):
    global qty_env
    global l_qty_envs
    l_qty_envs=s_env.set(qty_env)
    long=len(qty_env)
    if long>=2:
        error_digitos()
        clear()
    else:
        qty_env=qty_env+n
    s_env.set(qty_env)
def digitTW(m):
    global TW
    global STW
    TW=m
    STW.set(TW)
# Función Suministrar SUPERKOTE #
def suministrar():
    gpio_init()
    global qty_env
    global t_wait
    global TW
    initspk=tkinter.messagebox.askquestion('Suministrar','¿Desea iniciar suministro?')
    if initspk=='yes':
        if (t_suminist==0.0):
            error_time()
        elif (TW==0.0):
            error_tw()
        elif (qty_env==''):
            error_envases()
        else:
            for i in range (0,int(qty_env),1):
                input_stop=GPIO.input(17)
                GPIO.output(9,1)
                GPIO.output(23,1)
                GPIO.output(24,0)
                GPIO.output(25,0)
                sleep(t_suminist)
                GPIO.output(9,0)
```

```
                GPIO.output(23,1)
                GPIO.output(24,0)
                GPIO.output(25,1)
                if (i==int(qty_env)-1) or input_stop==True:
                    break
                else:
                    sleep(int(TW))
            pass
        clear()
    else:
       pass
def clear_calib():
    global t_suminist
    global s_tsuminist
    t_suminist=0.0
    s_tsuminist.set(t_suminist)
# Función eliminar último dígito #
def delete():
    global qty_env
    qty_env = qty_env.rstrip(qty_env[-1])
    s_env.set(qty_env)
# Función errores #
def error_digitos():
   tkinter.messagebox.showerror('Error','Dígitos Permitidos: 2')
   clear()
def error_tw():
   tkinter.messagebox.showerror('Error','Seleccione tiempo de espera')
   clear()
def error_time():
   tkinter.messagebox.showwarning('Error','Calcule el tiempo antes de suministrar')
def error_envases():
   tkinter.messagebox.showwarning('Error','Digite cantidad de envases a llenar')
def closeApp():
   salirMB=tkinter.messagebox.askquestion('Salir','¿Desea salir?')
   if salirMB=='yes':
       os.system("sudo shutdown -h now")
       #Screen.destroy()
   else:
       pass
def error_calib():
   tkinter.messagebox.showwarning(print("Envases a Llenar", qty_env))
# Paso1 Tiempo llenado #
lb_tx_t_suminist=Label(Screen,text="Paso 1:\nEstablecer el tiempo de suministro")
lb_tx_t_suminist.pack()
lb_tx_t_suminist.config(bg="gray20",fg=color_txlb,justify=LEFT,font=("Times New
Roman",13))
lb_tx_t_suminist.place(x=2,y=130)
# Dato Tiempo Suministro#
lb_t_suminist=Label(Screen,textvariable=s_tsuminist)
lb_t_suminist.pack()
lb_t_suminist.config(bg="gray20",fg=color_txlb,font=("Times New Roman",16))
lb_t_suminist.place(x=160,y=220)
# Segundos#
lb_t_sec=Label(Screen,text="segundos")
lb_t_sec.pack()
lb_t_sec.config(bg="gray20",fg=color_txlb,font=("Times New Roman",12))
lb_t_sec.place(x=150,y=250)
#PASO 2 TIEMPO ESPERA #
lb_tx_wait=Label(Screen,text="Paso 2: Digite tiempo de espera")
lb_tx_wait.pack()
lb_tx_wait.config(bg="gray20",fg=color_txlb,justify=LEFT,font=("Times New Roman",13))
lb_tx_wait.place(x=10,y=350)
lb_tx_wait_show=Label(Screen,textvariable=STW)
lb_tx_wait_show.pack()
lb_tx_wait_show.config(bg="gray20",fg=color_txlb,justify=LEFT,font=("Times New Roman",13))
lb_tx_wait_show.place(x=80,y=425)
lb_tx_wait_show1=Label(Screen,text="Segundos")
lb_tx_wait_show1.pack()
lb_tx_wait_show1.config(bg="gray20",fg=color_txlb,justify=LEFT,font=("Times New
Roman",13))
lb_tx_wait_show1.place(x=110,y=425)
# Texto Cantidad Envases#
lb_qty_envases=Label(Screen,text="Paso 3:\nDigite la cantidad de envases")
```

```
lb_qty_envases.pack()
lb_qty_envases.config(bg="gray20",fg=color_txlb,justify=LEFT,font=("Times New Roman",13))
lb_qty_envases.place(x=560,y=130)
# PASO 4 SUMINISTRAR#
lb_paso4=Label(Screen,text="Paso 4:\nSuministrar")
lb_paso4.pack()
lb_paso4.config(bg="gray20",fg=color_txlb,justify=LEFT,font=("Times New Roman",13))
lb_paso4.place(x=560,y=230)
# Etiqueta Universidad #
lb_uni=Label(Screen,text="Universidad Nacional de Ingeniería")
lb_uni.pack()
lb_uni.config(bg="gray20",fg=color_txlb,font=("Times New Roman",16))
lb_uni.place(x=250,y=9)
# Etiqueta Facultad #
lb_fec=Label(Screen,text="Facultad de Electrotecnia y Computación")
lb_fec.pack()
lb_fec.config(bg="gray20",fg=color_txlb,font=("Times New Roman",12))
lb_fec.place(x=270,y=39)
# Etiqueta Tema #
lb_tit=Label(Screen,text="Diseño e implementación de un controlador electrónico para
automatizar\n el suministro de tratamiento"
    " para metal SUPERKOTE2000 provisto por la empresa Distribuidora de Productos y
Servicios S.A \n (DPS).")
lb_tit.pack()
lb_tit.config(bg="gray20",fg=color_txlb,font=("Times New Roman",11))
lb_tit.place(x=16,y=69)
# INICIAR CALIBRACION #
btn_init_calib=Button(Screen,bd=4,width=10,height=1,fg="black",command=lambda:calib(),acti
vebackground="sienna3",activeforeground="black",
    bg="sienna1",relief="raised",font=("Times New Roman",11),text="Calc Tiempo")
btn_init_calib.pack()
btn_init_calib.place(x=20,y=230)
# DETENER CALIBRACION #
btn_load=Button(Screen,bd=4,width=10,height=1,fg="black",command=lambda:load_SPK(),activeb
ackground="sienna3",activeforeground="black",
    bg="sienna1",relief="raised",font=("Times New Roman",11),text="Cargar SPK")
btn_load.pack()
btn_load.place(x=20,y=180)
#BORRAR CALIBRACIÓN#
btn_clear_calib=Button(Screen,bd=4,width=10,height=1,fg="black",command=lambda:clear_calib
(),
activebackground="sienna3",activeforeground="black",
    bg="sienna1",relief="raised",font=("Times New Roman",11),text="Borrar")
btn_clear_calib.pack()
btn_clear_calib.place(x=20,y=280)
#TIEMPO DE ESPERA#
tw1=Button(Screen,text="10",bg=color_bt,fg=color_btText,activeforeground=color_bt_foreg,ac
tivebackground=color_btActivated,
    width=3,height=1,command=lambda:digitTW("10")).place(x=30,y=380)
tw2=Button(Screen,text="20",bg=color_bt,fg=color_btText,activeforeground=color_bt_foreg,ac
tivebackground=color_btActivated,
    width=3,height=1,command=lambda:digitTW("20")).place(x=100,y=380)
tw3=Button(Screen,text="30",bg=color_bt,fg=color_btText,activeforeground=color_bt_foreg,ac
tivebackground=color_btActivated,
    width=3,height=1,command=lambda:digitTW("30")).place(x=170,y=380)
# Teclado Numérico #
# Botón Cero #
btn0=Button(Screen,text="0",bg=color_bt,fg=color_btText,activeforeground=color_bt_foreg,ac
tivebackground=color_btActivated,
    width=width_bt,height=height_bt,command=lambda:digit("0")).place(x=350,y=340)
# Botón Uno #
btn1=Button(Screen,text="1",bg=color_bt,fg=color_btText,activeforeground=color_bt_foreg,ac
tivebackground=color_btActivated,
    width=width_bt,height=height_bt,command=lambda:digit("1")).place(x=255,y=130)
# Botón Dós #
btn2=Button(Screen,text="2",bg=color_bt,fg=color_btText,activeforeground=color_bt_foreg,ac
tivebackground=color_btActivated,
    width=width_bt,height=height_bt,command=lambda:digit("2")).place(x=350,y=130)
# Botón Tres #
btn3=Button(Screen,text="3",bg=color_bt,fg=color_btText,activeforeground=color_bt_foreg,ac
tivebackground=color_btActivated,
    width=width_bt,height=height_bt,command=lambda:digit("3")).place(x=445,y=130)
# Botón Cuatro#
```

```
btn4=Button(Screen,text="4",bg=color_bt,fg=color_btText,activeforeground=color_bt_foreg,ac
tivebackground=color_btActivated,
    width=width_bt,height=height_bt,command=lambda:digit("4")).place(x=255,y=200)
# Botón Cinco #
btn5=Button(Screen,text="5",bg=color_bt,fg=color_btText,activeforeground=color_bt_foreg,ac
tivebackground=color_btActivated,
    width=width_bt,height=height_bt,command=lambda:digit("5")).place(x=350,y=200)
# Botón Seis #
btn6=Button(Screen,text="6",bg=color_bt,fg=color_btText,activeforeground=color_bt_foreg,ac
tivebackground=color_btActivated,
    width=width_bt,height=height_bt,command=lambda:digit("6")).place(x=445,y=200)
# Botón Siete #
btn7=Button(Screen,text="7",bg=color_bt,fg=color_btText,activeforeground=color_bt_foreg,ac
tivebackground=color_btActivated,
    width=width_bt,height=height_bt,command=lambda:digit("7")).place(x=255,y=270)
# Botón Ocho #
btn8=Button(Screen,text="8",bg=color_bt,fg=color_btText,activeforeground=color_bt_foreg,ac
tivebackground=color_btActivated,
    width=width_bt,height=height_bt,command=lambda:digit("8")).place(x=350,y=270)
# Botón Nueve #
btn9=Button(Screen,text="9",bg=color_bt,fg=color_btText,activeforeground=color_bt_foreg,ac
tivebackground=color_btActivated,
    width=width_bt,height=height_bt,command=lambda:digit("9")).place(x=445,y=270)
# Botón DELETE #
btnDEL=Button(Screen,text="<-
",bg=color_bt_func,fg=color_btText,activeforeground=color_bt_foreg,activebackground=color_
bt_func,
    width=width_bt,height=height_bt,command=delete).place(x=255,y=340)
# Botón ERASE #
btnERASE=Button(Screen,text="ERASE",bg=color_bt_func,fg=color_btText,activeforeground=colo
r_bt_foreg,activebackground=color_bt_func,
    width=width_bt,height=height_bt,command=clear).place(x=445,y=340)
# Botón START #
btnSTART=Button(Screen,text="SUMINISTRAR",bg=color_btstart,fg="black",activeforeground=col
or_bt_foreg,activebackground=color_btstart_pressed,
    width=20,height=height_bt,command=suministrar).place(x=570,y=280)
# Botón STOP #
btnSTOP=Button(Screen,state=
ACTIVE,text="SALIR",bg=color_btstop,fg="white",activeforeground=color_bt_foreg,activebackg
round=color_btstop_pressed,
    width=20,height=height_bt,command=lambda:closeApp()).place(x=570,y=360)
### PROPIEDADESs_envases Teclado ###
Entry(Screen,font=('Arial',19,"bold"),width=4,textvariable=s_env,bd=1,insertwidth=8,bg="la
vender",justify="right").place(x=630,y=180)
clear()
gpio_init()
v_calib_init()
Screen.mainloop()
GPIO.cleanup()
KeyboardInterrupt()
```

Anexo 5. Descripción del Controlador Superkote 2000.

1. **# Controlador Electrónico SUPERKOTE2000 #**
2. De la librería **tkinter** se importan todos los módulos.
3. Se importa especificamente el módulo **ttk** que contiene los combobox.
4. Se importa el módulo **tkinter.messagebox** el cual permite crear pequeñas ventanas para mensajes para el usuario durante el proceso de suministro.
5. De la librería Python Imaging Library (**PIL**), se importan los módulos image, ImageTkcual se utilizó para cargar las imágenes(iconos) de la Universidad y la facultad en el frame del controlador. El módulo ImageTk contiene soporte para crear y modificar objetos a partir de imágenes PIL.
6. Importamos la librería **time.**
7. De la librería **time** importamos el módulo **sleep.**
8. Importamos la librería **RPI.GPIO** (Raspberry Pi General Porpose Input Output) y le asignamos el nombre de **GPIO** para simplificar el llamado de la librería a lo largo del código. Este paquete proporciona una clase para controlar el GPIO en una Raspberry Pi.
9. Creamos una ventana y le asignamos una variable llamada **Screen**. Una ventana es una instancia de la clase Tkinter Tk.
10. Las dimensiones de la ventana no serán ajustables por parte del usuario.
11. La ventana tendrá como título el texto : "Controlador Electrónico para Suministar SUPERKOTE2000".
12. La ventana tendra fondo de color gris al 20%.
13. Y tendrá dimensiones fijas de 800 pixeles de ancho por 480 pixeles de alto.
14. **#LOGO UNI#**
15. La variable **imguni** contiene el archivo en formato png con el logotipo de la universidad.
16. La variable **logo_uni** es objeto label que está ubicado en la ventana **Screen** y que un su interior contiene la imagen **imguni** en formato png.
17. Llamamos al organizador **.pack()**
18. Se posiciona la imagen en coordenadas específicas de la ventana.
19. **#LOGO FEC#**
20. La variable **imgfec** contiene el archivo en formato png con el logotipo de la facultad.
21. La variable **logo_fec** es un objeto label que está ubicado en la ventana **Screen** y que un su interior contiene la imagen **imgfec** en formato png.
22. Llamamos al organizador **.pack()**

23. Se posiciona la imagen en coordenadas específicas de la ventana.
24. **#Declaraciones para Botones#**
25. Los botones en la pantalla tendran un ancho de 8 píxeles.
26. Su altura es de 3 lineas de texto.
27. Color de primer plano cuando el boton esta debajo del cursor: 'white'.
28. Color del botón: 'steelblue'.
29. Color de botones de funciones DEL, ERASE : 'red3´
30. Color del botón start: 'limegreen'.
31. Color del botón start cuando es presionado: 'darkgreen'.
32. Color del botón stop: 'red2'.
33. Color del botón stop cuando es presionado:'red3'.
34. Color del texto en botones: 'white'
35. Color de fondo cuando el botón esta activado: 'dimgray'.
36. Color de texto en las etiquetas que muestran los textos en la ventana: 'gray80'.
37. **#Declaraciones para etiquetas#** En esta sección de código se declaran las variables a utilizar en las funciones.
38. Se declara **qt_env** (cantidad de envases) de tipo cadena
39. Se declara **t_wait** (tiempo de espera entre envases) de tipo entero.
40. Se declara **t_init** (tiempo de inicio) de tipo flotante.
41. Se declara **t_finish** (tiempo final) de tipo flotante.
42. Se declara **t_suminist** (tiempo de suministro) de tipo flotante.
43. Se declara **s_env** (envases convertidos a tipo cadena) de tipo cadena, para mostarlo en un etiqueta tkinter en la interfaz de usuario.
44. Se declara **s_tsuminist** (tiempo de suministro convertido a cadena) de tipo cadena, para mostrarlo en una etiqueta tkinter en la interfaz de usuario.
45. **#Función Setup GPIO inicio#**
46. En esta parte del código se crea la primera función del programa llamada **gpio_init()** donde inicializamos los ajustes de los pines de la raspberry.
47. Se desactivan las advertencias por pines usados en codigos ajenos al desarrollado.
48. Se configura en modo de trabajo de GPIO por el tipo **Broadcom SOC channel**.

49. Se declara el pin 22 como pin de adquisicion o entrada de datos, el cual se comportara como **Pull Down** para el sensor del recipiente calibrador.
50. Se declara el pin 9 como pin de salida de datos, este pin se usa para activar o desactivar la bomba.
51. Se declara el pin 10 como pin de salida de datos para activar el fan que reduce el calentamiento dentro controlador.
52. Se configura el pin 23 como pin de salida de datos para el led de color verde, el cual indica que el suministro esta en proceso.
53. Se configura el pin 24 como pin de salida de datos para el led de color rojo, el cual indica que el controlador no esta suministrando.
54. Se configura el pin 25 como pin de salida de datos para el led color azul, el cual indica que el suministro ha finalizado.
55. Se declara que al inicio de esta función, el pin 24 (led rojo) esta en alto.
56. Se declara que el pin 9 (bomba) esta en bajo.
57. Se declara que el pin 10(fan) esta en bajo.
58. Se declara que el pin 23 (led verde) esta en bajo.
59. Se declara que el pin 25 (led azul) está en bajo.
60. En esta area se declara la función **v_calib_init** (valores de calibración inicial).
61. Se configura la variable **t_suminist=0.0** segundos
62. A continuación, se configura para que la etiqueta que muestra en la interfaz, adquiera el valor de la variable **t_suminist** y lo presente al usuario.
63. **#Función Limpiar Valores en Pantalla#**
64. En esta sección de código se declara la función **clear** (borrar).
65. Se llama a la variable **qty_env** en modo global para poder modificarla en cualquier parte del código.
66. Se hace un llamado a la función **gpio_init()**.
67. Se asigna un valor en blanco a la variable **qt_env**.
68. Se configura la variable **s_env** (envases tipo cadena) con valor de cero(0).
69. **#Función Calibración#**
70. En esta area se declara la función **calib** (calibración) la cual realiza la calibracion de los envases.
71. Se llama a la variable **t_suminist** (tiempo suministro) en modo global.
72. Se llama a la variable **t_finish** (tiempo final) en modo global.
73. Se llama a la variable **t_init** (tiempo inicial) en modo global.
74. Se llama a la función **clear** para limpiar valores que hayan quedado en anteriormente.
75. Se configura el pin 9 (bomba) en bajo.

76. Se llama a la función **clear_calib** la cual borra el tiempo de calibrarcion calculado anteriormente y lo deja en blanco para guardar uno nuevo.
77. Al hacer clic en el botón de calibar se asigna a **t_init** el valor de **time.monotonic()** el cual mide el tiempo transcurrido en un proceso de larga duración porque se garantiza que nunca se moverá hacia atrás, incluso si se cambia la hora del sistema.
78. Iniciamos el ciclo **While True** donde se evalua la una condicion de verdad para la entrada del pin 22 (sensor de nivel).
79. Se declara la variable **input_state** la cual tiene el valor booleano del pin 22 GPIO de la raspberry.
80. Se evalua **input_state,** mientras el valor sea **cero lógico** (0), quiere decir el voltaje en el pin 22 es 0 voltios, lo cual nos indica que el sensor entre sus pines mide una alta resistencia electrica en el orden de los mega ohms.
81. Cumplida la condición se acciona la bomba por medio del pin9, iniciando a suministrar liquido en el recipiente de calibración.
82. **Else**, de lo contrario si el pin 22 obtiene un valor de 1 logico (1), un voltaje de 3.3V.
83. Se procede a desctivar la bomba en el pin 9.
84. Se calcula el **t_finish** usando time-monotic(), que seria el tiempo actual.
85. **t_suminist** obtiene el resultado redondeado a tres dígitos de la resta de **t_finish-t_init** (tiempo final menos tiempo inicial) obteniendo como resultado el tiempo en segundos desde que se dio clic al boton calibrar, hasta que se detuvo el suministro por medio del sensor.
86. Se asigna a la variable **s_tsuminist** de tipo cadena el valor del tiempo calculado y almacenado en **t_suminist** de tipo flotante.
87. Se rompe el ciclo.
88. **#Función capturar dígito#**
89. Esta seccion de código contiene la función **digit(n)**, la cual captura el digito presionado en el teclado y lo muestra en la caja de texto.
90. Se hace llamado a la variable **qty_env** en modo global la cual esta asociada al teclado numérico.
91. Se declara la variable **long** (longitud) que calcula la longitud de dígitos almacenados en la variable **qty_env.**
92. Evalua si la variable **long** es mayor que dós.
93. Si lo fuera, envia al usuario el mensaje **error_digitos.**
94. Se llama a la función **clear.**
95. De lo contrario, en caso de no cumplirse la condición.
96. **qty_env** es igual al valor contenido en si misma mas el digito que se presione.
97. Se configura **s_env**(variable de tipo cadena para mostrar al usuario) con el valor de **qty_env.**
98. **#Función Suministrar#**
99. Se crea la función **suministrar** la cual se explica acontinuación:
100. Se hace un llamado a la función **gpio_init()** garantizando que la bomba se encueentre en off, los diodos leds en su estado previo a suministro.

101. **qty_env** es llamada en modo global, lo cual nos grantiza utitlizarla aunque ha sido calculada en una función anterior.
102. De igual manera la variable **t_wait** trabaja en modo global
103. **t_wait** captura el valor seleccionado del cuadro de lista para tiempo de espera entre envases.
104. Inicia a evaluar una triple condicion anidada, si el valor en **t_suminist** devuelto es exacatamente igual a cero.
105. Llama a la función **error_time()** la cual le dira al usuario que debe calibrar el tiempo.
106. La segunda condición evalua si el usuario ha elegido el tiempo de espera entre envases, si la variable **t_wait** devuelve un valor vacio (t_wait==“”).
107. Envía al usuario **error_twait**(error de tiempo de espera).
108. La tercera condición se asegura de que el usuario haya seleccionado la cantidad de envases a llenar por medio de teclado numérico y guardado en la variable **qty_env** se revisa el dato almacenado en ella.
109. Si dicho valor devolviera a python vacio, envia al usuario **error_envases** el cual le dira al usuario que debe seleccionar la cantidad de envases a llenar.
110. De lo contrario, si todas las condiciones se cumplen,
111. Inicia el ciclo **for i in range,** para **i** con incrementos de 1, en un rango desde cero hasta **qty_env.**
112. Bomba en alto por medio del pin 9.
113. El diodo led verde(start) en alto por medio del pin 23.
114. El diodo led rojo(stop) en bajo por medio del pin 24.
115. El diodo led azul en bajo por medio del pin 25.
116. Estos pines permanecerán en ese estado hasta que **t_suminist** asignado a **sleep** haya transucrrido, que no es mas que el tiempo de calibración.
117. Una vez transucrrido el **t_suminist**, el pin 9 pasa a estar en bajo.
118. El diodo led verde(start) se comporta en bajo por medio del pin 23.
119. El diodo led rojo(stop) se activa en alto por medio del pin 24.
120. El diodo led azul(finish) se activa en alto por medio del pin 25.
121. Evalua la condición para saber cuando salir del ciclo de cantidad de envases a llenar, para lo cual, si **i** es exactamente igual a la cantidad de envases ingresados por el usuario y almacenado en **qty_env** menos 1; esto porque python inicia a contar desde cero y no en 1 como lo hace el usuario.
122. Si la condición se cumple, entonces rompe el ciclo y termina el llenado.
123. De lo contrario
124. Se aplica el tiempo de espera repitiendo el ciclo de llenado hasta que la condición anterior se cumple.
125. Sale del ciclo for

126. Llamado a la función **clear** que devolvera los valores de los diodos leds y cantidad de envases a los valores contenidos en la función.
127. En esta sección de código se describe la funcion **clear_calib,** limpiar calibración, es una opcion para el usuario borrar la calibración del recipiente actual, para proceder a llenar otra medida.
128. Se llama a la variable **t_suminist** en modo global para modificarla desde esta parte del código.
129. Se llama a la variable **s_tsuminist** en modo global para modificar el dato a presentar al usuario.
130. Se configura **t_suminist=0.0**
131. Se configura para que la etiqueta que muestra al usuario el tiempo de suministro obtenga a **t_suminist**
132. **#Función eliminar último dígito del teclado#**
133. En esta sección del codigo se desarrollo la función que para los teclado es igual a "DELETE".
134. Se hace el llamado a la variable **qty_env** en modo global.
135. Para elimiinar el último digito usa el método **rstrip ()** que elimina cualquier carácter final de una cadena así que a **qty_env** resta el último digito almacenado en si mismo.
136. Se configura para que **s_env** adopte el nuevo valor de **qty_env.**
137. **#Función errores#**
138. Se crea la función **error_digitos**
139. Se presenta en pantalla un cuadro de diálogo de tipo **messagebox.showerror** que dispone la librería tkinter. El mensaje de error le indica al usuario que ingreso mas digitos de los permitidos.
140. Se llama a la función **clear**
141. Se crea la función **error_time**
142. Se crea un cuadro de diálogo de tipo **messagebox.showwarning** que dispone la librería tkinter. El mensaje de error le indica al usuario que no ha configurado el tiempo de suministro en los envases de calibración.
143. Se crea la función **error_envases**
144. Se crea un cuadro de diálogo de tipo **messagebox.showwarning** que dispone la librería tkinter. El mensaje de error le indica al usuario que debe ingresar la cantidad de envases a llenar.
145. Se crea la función **error_twait**
146. Se crea cuadro de diálogo de tipo **messagebox.showwarning** que dispone la librería tkinter. El mensaje de error le indica al usuario que debe ingresar seleccionar una opcion de tiempo de espera entre envases.
147. **#Paso1 Tiempo de llenado#**
148. Se crea la etiqueta (**lb_tx_t_suminist)** que contendra el texto fijo "Paso 1 Establecer el tiempo de llenado"
149. Se llama al organizador de objetos de tkinter.
150. Se ajustan las propiedades(color de fondo, alineación, fuente, tamaño de fuente)
151. Se ubica la etiqueta en las coordenadas descritas en el código.
152. **#Dato tiempo de suministro#**

153. Se crea la etiqueta (**lb_t_suminist)** que contendra el valor de **t_suminst** acondiconado en la variable **s_tsuminist** de tipo cadena para su presentación al usuario.
154. Se llama al organizador de objetos de tkinter.
155. Se ajustan las propiedades(color de fondo, alineación, fuente, tamaño de fuente)
156. Se ubica la etiqueta en las coordenadas descritas en el código.
157. **#Segundos#**
158. Se crea la etiqueta (**lb_tx_t_suminist)** que contendra el texto fijo "segundos"
159. Se llama al organizador de objetos de tkinter.
160. Se ajustan las propiedades(color de fondo, alineación, fuente, tamaño de fuente)
161. Se ubica la etiqueta en las coordenadas descritas en el código.
162. **#PASO2 TIEMPO ESPERA#**
163. Se crea la etiqueta (**lb_tx_wait)** que contendra el texto fijo "Paso 2 Seleccionar el tiempo de espera"
164. Se llama al organizador de objetos de tkinter.
165. Se ajustan las propiedades(color de fondo, alineación, fuente, tamaño de fuente)
166. Se ubica la etiqueta en las coordenadas descritas en el código.
167. **#Texto Cantidad Envases#**
168. Se crea la etiqueta (**lb_qty_envases)** que contendra el texto fijo "Paso 3 Digite la cantidad de envases"
169. Se llama al organizador de objetos de tkinter.
170. Se ajustan las propiedades(color de fondo, alineación, fuente, tamaño de fuente)
171. Se ubica la etiqueta en las coordenadas descritas en el código.
172. **#PASO 4 SUMINISTRAR#**
173. Se crea la etiqueta (**lb_paso4)** que contendra el texto fijo "Paso 4 Suministrar"
174. Se llama al organizador de objetos de tkinter.
175. Se ajustan las propiedades(color de fondo, alineación, fuente, tamaño de fuente)
176. Se ubica la etiqueta en las coordenadas descritas en el código.
177. **#Etiqueta Universidad#**
178. Se crea la etiqueta (**lb_uni)** que contendra el texto fijo "Universidad Nacional de Ingeniería"
179. Se llama al organizador de objetos de tkinter.
180. Se ajustan las propiedades(color de fondo, alineación, fuente, tamaño de fuente)
181. Se ubica la etiqueta en las coordenadas descritas en el código.
182. **#Etiqueta Facultad#**
183. Se crea la etiqueta (**lb_fec)** que contendra el texto fijo "Facultad de Electrotecnia y Computación"
184. Se llama al organizador de objetos de tkinter.
185. Se ajustan las propiedades(color de fondo, alineación, fuente, tamaño de fuente, etc)
186. Se ubica la etiqueta en las coordenadas descritas en el código.
187. **#Etiqueta Tema#**

188. Se crea la etiqueta (**lb_tit)** que contendra el texto fijo “Diseño e implementación de un controlador
189. electrónico …”
190. Se llama al organizador de objetos de tkinter.
191. Se ajustan las propiedades(color de fondo, alineación, fuente, tamaño de fuente, etc)
192. Se ubica la etiqueta en las coordenadas descritas en el código.
193. **#INICIAR CALIBRACIÓN#**
194. Se crea el botón (**btn_init_calib)** que contendra el texto fijo “Iniciar”
195. Se ajustan las propiedades(color de fondo, alineación, fuente, tamaño de fuente, etc) y en esta parte se ajusta la acción a realizar cuando se presione el botón que será llamar a la función **calib()**
196. Se llama al organizador de objetos de tkinter.
197. Se ubica la etiqueta en las coordenadas descritas en el código.
198. **#BORRAR CALIBRACIÓN#**
199. Se crea el botón (**btn_clear_calib)** que contendra el texto fijo “Borrar”
200. Se ajustan las propiedades(color de fondo, alineación, fuente, tamaño de fuente, etc) y en esta parte se
201. Se ajusta la acción a realizar cuando se presione el botón que será llamar a la función **calib()**
202. Se llama al organizador de objetos de tkinter.
203. Se ubica la etiqueta en las coordenadas descritas en el código.
204. **#ListBox Tiempo de Espera#**
205. Se crea el cuadro de lista (**list_twait)** que forma parte de la librería actualiza de tkinter usando el módulo **ttk.combobox**.
206. Se definen los valores a desplegar en la lista (1,5,10)
207. Se llama al organizador de objetos de tkinter y se ubica la etiqueta en las coordenadas descritas en el código.
208. **#Teclado numérico#**
209. **#Botón Cero#**
210. Se crea el botón (**btn0)** utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
211. La acción a tomar cuando se haga clic será en la función **digit(n)** tomar el valor ”**0**”.
212. **#Botón Uno#**
213. Se crea el botón (**btn1)** utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
214. La acción a tomar cuando se haga clic será en la función **digit(n)** tomar el valor ”**1**”.
215. **#Botón Dos#**

216. Se crea el botón (**btn2**) utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
217. La acción a tomar cuando se haga clic será en la función **digit(n)** tomar el valor "**2**".
218. **#Botón Tres#**
219. Se crea el botón (**btn3**) utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
220. La acción a tomar cuando se haga clic será en la función **digit(n)** tomar el valor "**3**".
221. **#Botón Cuatro#**
222. Se crea el botón (**btn4**) utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
223. La acción a tomar cuando se haga clic será en la función **digit(n)** tomar el valor "**4**".
224. **#Botón Cinco#**
225. Se crea el botón (**btn5**) utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
226. La acción a tomar cuando se haga clic será en la función **digit(n)** tomar el valor "**5**".
227. **#Botón Seis#**
228. Se crea el botón (**btn6**) utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
229. La acción a tomar cuando se haga clic será en la función **digit(n)** tomar el valor "**6**".
230. **#Botón Siete#**
231. Se crea el botón (**btn7**) utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
232. La acción a tomar cuando se haga clic será en la función **digit(n)** tomar el valor "**7**".
233. **#Botón Ocho#**
234. Se crea el botón (**btn8**) utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
235. La acción a tomar cuando se haga clic será en la función **digit(n)** tomar el valor "**8**".
236. **#Botón Nueve#**
237. Se crea el botón (**btn9**) utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
238. La acción a tomar cuando se haga clic será en la función **digit(n)** tomar el valor "**9**".

239. **#Botón DELETE#**
240. Se crea el botón (**btnDEL)** utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
241. La acción a tomar cuando se haga clic será ejecutar la función **delete().**
242. **#Botón ERASE#**
243. Se crea el botón (**btnERASE)** utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
244. La acción a tomar cuando se haga clic será ejecutar la función **clear().**
245. **#Botón START#**
246. Se crea el botón (**btnSTART)** utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
247. La acción a tomar cuando se haga clic será ejecutar la función **suministrar().**
248. **#Botón STOP#**
249. Se crea el botón (**btnSTOP)** utilizando el módulo Button de tkinter, se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
250. La acción a tomar cuando se haga clic será ejecutar la función **clear().**
251. **#Entrada Envases#**
252. Se crea el objeto de tipo entrada de datos utilizando el módulo Button de tkinte, que muestra los dígitos presionados en el teclado a través de su propiedad **textvariable=s_env.** Se configuran sus propiedades (color de fondo, alineación, fuente, tamaño de fuente, etc) y se ubica en las coordenas descritas en el código.
253. Se llama a la función **clear()**
254. Se llama a la función **gpio_init()**
255. Se llama a la función **v_calib_init()**
256. Se carga la pantalla (**Screen)** que contiene todos los objetos creados.
257. Se hace uso de **KeyboardInterrupt** para poder detener el programa mediante teclado.
258. Se liberan los pines que se usaron en codigos anteriores y que Raspberry podria entender que esten ocupados e impedir la ejecución correcta del código.

Bibliografía

Mejía Narváez, Espinoza Martínez. 2016. Universidad Nacional de Ingeniería, Nicaragua.

Beltrán Sánchez, Cepeda Sánchez, 2008. Universidad San Buenaventura, Bogotá Colombia,

Heriberto Mario. Universidad Austral, Chile.

J.P.P y M.Merino https://definicion.de, 2016 y actualizado 2017. https://definicion.de/automazacion/

Ogata K. Sensor de Ingeniería de Control

F.M.Pérez. Sustancias Lubricantes. La Tribología Ciencia y Técnica para el mantenimiento. México, Limusa 2002.

F.A.García , Características , uso y aplicación de lubricantes. Características, uso y aplicación de lubricantes. La Habana 1983.

https://raspberrypi.cl/que-es-raspberry/

https://www.redhat.com/es/topics/linux

https://www.raspbian.org/

https://www.raspberrypi.org/documentation/usage/gpio/

https://dcbballester.com/tipos-de-baterias-baterias-de-plomo

I.R.H. Gaviño, Clasificación de los sistemas de Control. Introducción a los Sistemas de Control: Conceptos, aplicaciones y simulación con Matlab, México, Pearson Educación 2010.

M.Pérez, A.Pérez Hidalgo, E. Pérez Berenguer, Introducción a los Sistemas de Control y Modelo Matemático para Sistemas Lineales Invariantes en el Tiempo, 2007.

R. L. Boylestad, Introducción al análisis de circuitos, Naucalpan de Juárez: PEARSON EDUCACIÓN, 2004.

T. L. Floyd, Principios de circuitos eléctricos. Octava Edición, Naucalpan de Juárez: Pearson Educación, 2007.

Rela Agustín, Electricidad y Electrónica, Morvillo Anselmo, capítulo 4 Electricidad Fundamentos de Electrodinámica. 19/01/2010

Effra, Factories of the future PPP, EU, 2005

L. Joyanes Aguilar, Fundamentos de Programación, Madrid: Mc Graw Hill, 2008.

https://code.visualstudio.com/docs

Ofelia Delfina Cervantes Villa Gomez, José Miguel David Báez López, Antonio Arizaga Silva, Esteban Castillo Juárez., Python con aplicaciones a las matemáticas, ingeniería y finanzas, Ciudad de México: Alfaomega Grupo Editor, 2017.