UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERÍA Y ARQUITECTURA

ESCUELA DE INGENIERÍA ELÉCTRICA

DESARROLLO DE UNA PLACA ELECTRÓNICA APLICADA EN LA INDUSTRIA TEXTIL BASADA EN EL HARDWARE Y SOFTWARE DE LIBRE UTILIZACIÓN

PRESENTADO POR:

HERNÁN EDGARDO GÁLVEZ MAJANO

PARA OPTAR AL TÍTULO DE:

INGENIERO ELECTRICISTA

CUIDAD UNIVERSITARIA, MAYO 2021

UNIVERSIDAD DE EL SALVADOR

RECTOR:

MSC. ROGER ARMANDO ARIAS ALVARADO

SECRETARIO GENERAL:

ING. FRANCISCO ANTONIO ALARCÓN SANDOVAL

FACULTAD DE INGENIERÍA Y ARQUITECTURA

DECANO:

DR. EDGAR ARMANDO PEÑA FIGUEROA

SECRETARIO:

ING. JULIO ALBERTO PORTILLO

ESCUELA DE INGENIERÍA ELÉCTRICA

DIRECTOR:

ING. ARMANDO MARTÍNEZ CALDERÓN

UNIVERSIDAD DE EL SALVADOR
FACULTAD DE INGENIERÍA Y ARQUITECTURA
ESCUELA DE INGENIERÍA ELÉCTRICA

Trabajo de Graduación previo a la opción al Grado de:

INGENIERO ELECTRICISTA

Título:

DESARROLLO DE UNA PLACA ELECTRÓNICA APLICADA EN LA INDUSTRIA TEXTIL BASADA EN EL HARDWARE Y SOFTWARE DE LIBRE UTILIZACIÓN

Presentado por:

HERNÁN EDGARDO GÁLVEZ MAJANO

Trabajo de Graduación Aprobado por:

Docente Asesor:

MSC. JOSÉ WILBER CALDERÓN URRUTIA

SAN SALVADOR, MAYO 2021

Trabajo de Graduación Aprobado por:

Docente Asesor:

MSC. JOSÉ WILBER CALDERÓN URRUTIA

NOTA Y DEFENSA FINAL

En esta fecha, viernes 12 de marzo de 2021, en la Sala de Lectura de la Escuela de Ingeniería Eléctrica, a las 9:00 a.m. horas, en presencia de las siguientes autoridades de la Escuela de Ingeniería Eléctrica de la Universidad de El Salvador:

1. Ing. Armando Martínez Calderón
   Director

   Firma

2. MSc. José Wilber Calderón Urrutia
   Secretario

   Wilber Calderón
   Firma

Y, con el Honorable Jurado de Evaluación integrado por las personas siguientes:

- MSC. JOSE WILBER CALDERON URRUTIA
  (Docente Asesor)

  Wilber Calderón
  Firma

- MSC. SALVADOR DE JESÚS GERMAN

  Firma

- ING. MARVIN GERARDO JORGE HERNANDEZ

  Firma

Se efectuó la defensa final reglamentaria del Trabajo de Graduación:

DESARROLLO DE UNA PLACA ELECTRÓNICA APLICADA EN LA INDUSTRIA TEXTIL BASADA EN EL HARDWARE Y SOFTWARE DE LIBRE UTILIZACIÓN

A cargo del Bachiller:

- GALVEZ MAJANO HERNAN EDGARDO

Habiendo obtenido en el presente Trabajo una nota promedio de la defensa final: 9.5

(Nueve . cinco)

AGRADECIMIENTOS

Agradezco primeramente a Dios Todopoderoso por permitirme haber llegado a esta etapa de mi vida, por su protección cada día, por la provisión, buena salud y todos los favores que he recibido de Él. A Él sea la honra, la gloria y la alabanza por los siglos de los siglos.

También agradezco a mi madre, Teresa de Jesús Majano (Q.D.D.G), quien ya fue llamada para descansar en paz en la Presencia de Dios, estoy seguro que ella hubiese disfrutado mucho de la meta la cual estoy próximo a alcanzar, ya que con mucho esfuerzo me inculcó los valores necesarios para poder triunfar en la vida de una manera honrada y tuvo una vida de lucha y mucho trabajo arduo para que yo tuviera acceso a una educación superior y lograr mis sueños. Gracias mamá por todo el amor, el cariño, la educación de calidad en casa, por la dedicación a la familia y a hacer de mi un hombre de bien para la sociedad, este próximo éxito es para usted.

Agradezco a mi padre Pedro de Jesús Gálvez Martínez (Q.D.D.G), un hombre del cual me siento muy orgulloso de llamar papá, él me enseño que con disciplina, persistencia, fe en Dios y mucho trabajo, se pueden lograr las metas en la vida. Gracias papá por todo el esfuerzo y la entrega a la familia, por el ánimo, la ayuda emocional y económica, por el sustento diario y todos sus años de vida invertidos en mi persona. Me encantaría que pudiese vivir este triunfo que no es solo mío, sino también suyo, pero sé que junto a mi madre se encuentran mucho mejor en la presencia de Dios.

A mi amada esposa Damaris de Gálvez, ya que por mucho tiempo ha permanecido a mi lado animándome en los momentos de más grande oscuridad en mi vida, nunca ha faltado una palabra de ánimo. Gracias por la comprensión, el amor y la dedicación con que me das fuerzas y me acompañas en este camino de la vida.

A mis hermanos, Jorge Gálvez, Edwin Gálvez y Pedro Gálvez, a mis compañeros de estudio y amigos que durante la carrera fueron piezas clave en las cuales pude apoyarme, Mario Machuca, Christian Ramos (Q.D.D.G), Everson Mazariego, Juan Vázquez, Everson Vicente, Edgardo Larín. A mis amigos, Jahaziel Mena, Brenda de Mena, Carlos Miranda, Aurelio Villalta, Leonardo Nieto, Héctor Pleitez, Héctor Granados, Fabricio Abarca y William Zelaya, los cuales me han animado a continuar para lograr el éxito en esta carrera

Agradezco al MSc. Ing. Wilber Calderón por tenderme la mano en este trabajo de graduación, por la asesoría, por la enseñanza y orientación para poder superar los diferentes obstáculos que se presentaron durante la realización de este trabajo.

A la empresa TEXTUFIL S.A de C.V por la oportunidad que me brindó de realizar todo el estudio necesario en sus instalaciones y la confianza depositada en mi para poder intervenir una sus maquinarias textiles en la cual se basa este trabajo de graduación, ya que se ha puesto en práctica buena parte del aprendizaje adquirido durante mi tiempo de estudio

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Actualmente en El Salvador, buena parte de la maquinaria industrial se ha ido modernizando con el paso de los años buscando satisfacer las exigencias por parte de sus clientes, que con el tiempo se van incrementando. Este fenómeno se debe a que a medida que la demanda de los clientes crece, también es necesario acelerar la producción de un determinado producto. La industria textil Salvadoreña no ha sido la excepción, y se ha tenido que modernizar la maquinaria textil dejando atrás los métodos tradicionales de elaboración de tejidos de manera artesanal, para dar paso a la producción en cantidades industriales que suple la demanda tanto nacional como internacional. Cabe destacar que las maquinarias modernas implementan numerosas técnicas de control ya sea, mecánicas, eléctricas, electrónicas, neumáticas, o combinación de todas ellas para llevar a cabo un proceso determinado, controladas por algún dispositivo inteligente. Teniendo en cuenta que El Salvador no es un país productor de tecnologías para la elaboración de productos textiles, cuando un dispositivo esencial de la maquinaria llega al final de su vida útil, se deba solicitar la compra al exterior dejando así maquinaria inactiva por períodos indefinidos.

En este contexto es donde este trabajo de graduación enfoca sus objetivos, ya que se ha tomado como objeto de trabajo una maquinaria textil que produce tejidos estampados, en la cual se ha observado que repercute la pronta adquisición de una tarjeta electrónica que controla un sistema muy importante del cual depende la buena calidad del producto final. Además los costos de adquisición son muy altos y los tiempos de entrega se extienden mucho más de lo previsto. Se hará uso de tecnologías basadas en software y hardware de libre distribución para obtener un dispositivo controlador que sustituya una placa electrónica con software y hardware propietario, pero que sea de bajo costo y que supere las deficiencias presentadas con frecuencia.

OBJETIVOS

GENERALES

- Desarrollar una placa electrónica basada en software y hardware de libre distribución, y de bajo costo para darle solución a un problema de una maquinaria textil estampadora a través del estudio y comprensión de un proceso que se lleva a cabo en una industria textil de El Salvador, logrando así igualar funciones de la placa electrónica original y superar el rendimiento.
- Hacer uso de herramientas de software y hardware de libre distribución, para dar solución a un problema de una de las industrias textiles de El Salvador.
- Desarrollar una solución económica y especialmente confiable que iguale o supere a sus similares comerciales.

ESPECÍFICOS

- Diseñar e implementar un hardware y software capaz de igualar o superar el rendimiento de partes originales de una maquinaria de la industria textil salvadoreña.
- Evaluar un proceso industrial que genera problemas de la vida real en una industria textil y darle solución a través del software y hardware de libre distribución.
- Desarrollar una aplicación industrial en una industria textil salvadoreña, que tengan una vida útil mayor a las comercialmente equivalentes.
- Reducir los costos en la reparación de una maquinaria de una industria textil y eliminar los tiempos de entrega relacionados con compras internacionales.
- Realizar un estudio comparativo del costo beneficio en las áreas técnica y económica que acompañan la aplicación de estas tecnologías de bajo costo comparada con la compra de hardware original.

ALCANCES

Haciendo uso de tecnologías de bajo costo y confiables, tales como el software y hardware de libre distribución, lograr sustituir un componente importante de una maquinaria industrial textil que solo se encuentra en el exterior del país y por lo tanto tiene un elevado costo o su tiempo de entrega es demasiado largo para poner en marcha dicha maquinaria cuyo funcionamiento es de vital importancia para mantener la línea de producción activa las 24 horas del día y los 7 días de la semana.

Al final de este trabajo de graduación, se tendrá un hardware que controla correctamente la posición del cabezal de estampación en una maquinaria textil industrial. El operario tendrá acceso a hacer los ajustes precisos lateral y diagonal necesarios para el correcto estampado. También se tendrá control de la bomba de inyección de tinta para evitar el derrame y asegurar la calidad de color requerido. Se obtendrá mayor durabilidad y robustez con la placa desarrollada, debido a que se implementará en la misma, software de libre distribución capaz de reconocer condiciones de operación inadecuada y así corregir las condiciones que en el caso de las placas comerciales sugeridas por el fabricante no pueden reconocer.

ANTECEDENTES

Actualmente en una industria textil salvadoreña buena parte de la maquinaria se encuentra en condiciones de funcionamiento automatizado, lo que permite que la producción satisfaga los niveles de demanda que los clientes requieren, además de brindar seguridad y cierto grado de comodidad a las personas responsables de la operación de dicha maquinaria.

Con el avance acelerado de las tecnologías muchos de los fabricantes de hardware para maquinaria textil industrial deciden ya no fabricarlos y dan paso a la fabricación de tecnologías con funciones más avanzadas que puedan competir con sus rivales comerciales. Muchas de estas nuevas funciones de tecnologías de última generación a veces quedan de sobra y no se adaptan a las necesidades de la maquinaria con tecnología de generaciones anteriores, agregando a esto que el costo de los nuevos dispositivos de hardware aumenta drásticamente basado en las nuevas funciones. El fenómeno antes descrito produce poca disposición de los dirigentes de la industria textil Salvadoreña antes mencionada a actualizar su maquinaria debido a los altos costos que esto produce, teniendo como consecuencia que esta maquinaria o funciones esenciales para la misma, permanezcan deshabilitadas, produciendo gastos recurrentes y mala calidad en el producto final. Por lo tanto, es necesario tomar medidas que puedan compensar estas situaciones de manera que sean de costo accesible, confiable y seguro para la operación satisfactoria garantizando así el óptimo funcionamiento de la maquinaria.

PLANTEAMIENTO DEL PROBLEMA

Se requiere solucionar un problema que se ha planteado en una industria textil Salvadoreña, el cual está relacionado a la falta de hardware original por costo excesivo y por tiempo de entrega muy largo. Sumado a las problemáticas mencionadas anteriormente, dicho hardware comercial sufre averías recurrentes debido a errores en el diseño del mismo, y por operación inadecuada por parte de los operarios.

La solución alternativa de bajo costo basada en hardware y software de libre distribución se pondrá a prueba sustituyendo una placa electrónica de un sistema llamado *Navigator Automata*, el cual se encarga del movimiento lateral y diagonal de un cabezal perteneciente a una máquina de fabricación austriaca cuya función es estampar tela de manera milimétrica minimizando considerablemente cualquier error y por ende dar la mejor calidad posible. El problema por el cual no se ha podido dar una solución anteriormente se basa en que no se han podido controlar dos señales analógicas que indican en todo momento la posición del cabezal.

JUSTIFICACIÓN

Este tipo de proyecto es muy importante en la realidad de la industria textil Salvadoreña, debido a que de la pronta puesta en marcha o de la sustitución oportuna de los dispositivos electrónicos de la maquinaria textil industrial dependen factores importantes, ya sea para la empresa como para los trabajadores. Una maquinaria podría continuar con sus funciones aun cuando se sacrifique la calidad del producto final a falta de repuestos de forma inmediata, esto conlleva al rechazo del producto por parte de los clientes, pudiendo así provocar fenómenos sociales adversos como el desempleo, también provocado a veces por el paro indefinido de maquinaria por falta de repuestos, es aquí donde el desarrollo de placas electrónicas aplicadas en la industria textil basadas en el hardware y software de libre distribución se vuelve importante.

CAPITULO 1: MARCO TEÓRICO

1.1 SOFTWARE DE LIBRE DISTRIBUCIÓN

De acuerdo con la filosofía del software libre, la definición de este según la FSF (Free software foundation) dice textualmente:

"Software libre significa software que respeta la libertad y la comunidad de los usuarios. A grandes rasgos, significa que los usuarios tienen la libertad de ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software"[1].

Como puede observarse dentro de la definición de software libre, caben las llamadas cuatro libertades esenciales del software libre, estas son:

- Libertad 0: Esta libertad se describe como la libertad para que un usuario ejecute un programa de la manera que desee, y con el propósito para el cual considere mas conveniente. El usuario final del software puede ser una persona u organización, y de acuerdo a esta libertad lo único que interesa es el propósito del usuario final del software y no el de el programador, ya que incluso si el programa es útil o no, si tiene fallas o no, las demás libertades esenciales respaldan al usuario final para que este lo estudie, lo modifique, y pueda mejorar el programa. Como dato curioso de esta libertad, se puede decir que su numeración se debe a que en el año 1990 solo se habían planteado tres libertades esenciales, pero al darse cuenta de que la libertad de ejecución de un programa es aun mas importante que las demás, se decidió que en lugar de volver a enumerar las libertades anteriores a fin de que la libertad de ejecución permaneciera en primer lugar, se le puso el número cero en lugar de uno.

- Libertad 1: Esta libertad establece que el usuario final del software es libre de estudiar como funciona el programa para luego modificar el código fuente creado por alguien mas y hacer uso del programa modificado. Para que esta libertad se pueda llevar a cabo es necesario primeramente tener acceso completo al código fuente, es decir, que no hayan secciones del código las cuales no sean accesibles o modificables para el usuario final. Según la FSF si un código fuente no se libera completamente, entonces este no es considerado como software libre, ya que si alguna de las cuatro libertades es violada incluso en algún detalle mínimo, entonces este software es privativo.

- Libertad 2: En esta libertad el usuario puede copiar y redistribuir el software incluso sin notificar al programador que escribió el código fuente, debido a que la filosofía del software libre evita que se acarreen consecuencias legales por infringir derechos de autor. El usuario

puede hacer esta distribución ya sea de manera gratuita o cobrando el valor que él considere conveniente, ya que el software libre no debe confundirse como “software gratis”.

El software redistribuido puede ser el que se obtuvo originalmente sin modificaciones o puede incluir las modificaciones que el usuario que distribuye le haya realizado. No existe ninguna consecuencia legal para una persona que se lucre de la distribución del software libre, de hecho se insta a que se realice esta práctica y se pueda comercializar y “*cobrar tanto como se desee o tanto como sea posible ya que la redistribución se considera una actividad buena y legítima”*[2]

- Libertad 3: Establece que a diferencia del software privativo, este se puede modificar y mejorar, de manera que al distribuirlo a terceros, estos también puedan gozar de los beneficios de dichas modificaciones.

  En esta libertad también se da a conocer que al distribuir un determinado programa catalogado como software libre, este debe de distribuirse en sus formas ejecutable y como código fuente, a fin de que el usuario que decida hacer uso de dicho programa pueda ejecutarlo sin problemas y si desea modificarlo o mejorarlo tenga acceso al código fuente para realizar dichas actividades.

Se puede observar que las cuatro libertades esenciales del software libre, se respaldan unas con otras, y que de omitirse algún pequeño detalle de estas libertades, el software puede llegar a considerarse como software privativo, incluso si cumple muchas de las características del software libre.

Ya que el término original para Software libre se encuentra en el idioma inglés (Free software), suele existir la confusión de que el software libre es “gratis, debido a que en el idioma inglés la palabra “free” tiene dos significados, los cuales son, libre y gratis, de manera que es una palabra muy ambigua para transmitir la idea fundamental del software libre. Como detalla la FSF se ha hecho uso de las palabra del idioma español “gratis” y “libre” para diferenciar la filosofía del software libre.

Figura 1: Logo de la Free Software Foundation

Debido a las numerosas dudas que surgen al preguntarse si un software es libre o no, la definición suele revisarse cada cierto tiempo dejando como resultado una serie de revisiones que se han realizado a lo largo del tiempo, actualmente la última revisión es la 1.168 y data del Martes 30 de julio de 2019 [3]. Cabe mencionar que algunas revisiones no poseen cambios muy significativos, mas bien incluyen

solamente aclaraciones que de acuerdo a la FSF no afectan la definición o las interpretaciones del software libre.

Actualmente se pueden encontrar numerosas aplicaciones de software libre en internet, desde juegos, multimedia, ofirmatica, herramientas para análisis matemáticos avanzados, hasta aplicaciones avanzadas de ingeniería, las cuales incluyen diseño CAD, análisis de circuitos eléctricos y muchas mas.

1.2 HARDWARE DE LIBRE DISTRIBUCIÓN

Además del software libre o de libre distribución, también esta filosofía se aplica al hardware, pero con con la diferencia de que este último es algo tangible. A diferencia del software, el hardware se trata de componentes físicos, es decir, puede tratarse de algún objeto diseñado para ser impreso en 3D, o también circuitos integrados y/o componentes electrónicos que en determinada configuración realizan una función especifica. Ya que la adquisición de dichos componentes conlleva un costo monetario no se puede tratar exactamente bajo los mismo ideales que el software libre, pero de acuerdo con la FSF, la filosofía se aplica a los diseños de hardware, es decir, a la parte que se realiza antes de llevar a cabo la fabricación, entonces el término mas adecuado para hardware libre o de libre distribución es *“hardware de diseño libre”*, ya que es a los diseños a los cuales se les aplica la filosofía de software libre.

Como definición formal de parte de la FSF para los diseños de hardware libre, la FSF dice textualmente:

“El hardware libre significa hardware que los usuarios pueden usar y copiar y redistribuir con o sin cambios.”[4]

Básicamente en su definición se encuentran implícitas las cuatro libertades esenciales que también se aplican al software de libre distribución, con la excepción de que estas se aplican a los diseños de hardware y no al hardware en sí.

Al igual que con el software libre, la definición de hardware libre también tiende a revisarse periódicamente para aclarar y ampliar conceptos en los cuales surge confusión o necesitan reforzarse más. La revisión más reciente es la 1.23, con fecha lunes 30 de diciembre de 2019. [5]

En la actualidad existe un gran número de dispositivos electrónicos que son útiles para la vida cotidiana, tales como computadoras, reproductores de música y vídeos, teléfonos móviles inteligentes y una inmensa gama de productos y dispositivos para el hogar, oficina e industria en general de los cuales en su mayoría no se conoce la manera en la que sus componentes electrónicos están organizados, es decir, no se conocen detalles de sus diseños, de manera que cualquier persona pueda reproducirlos o modificarlos para el fin que se desee. Este tipo de hardware es llamado de tipo restrictivo, ya que solo el fabricante posee los diseños. Sin embargo en los últimos años se ha visto un incremento en la

liberación de los diseños de hardware de ciertas aplicaciones las cuales pueden ser útiles para el ámbito de la educación, doméstico, oficina e incluso algunas para industria.

Los diseños que pueden encontrarse bajo la filosofía del hardware de diseño libre pueden ser fácilmente reproducidos y distribuidos con algún o ningún costo y su ensamble depende de cuan accesibles sean los componentes que conforman el hardware.

Uno de los beneficios que conlleva el poder realizar los ensambles propios haciendo uso de hardware de diseño libre es que se puede lograr disminuir los costos de una manera notable con respecto a la obtención o compra de hardware restrictivo. Por supuesto que esto depende claramente del entorno donde se realice tal actividad, ya que hay que tener en cuenta que la situación de los diferentes países es muy diferente con respecto a la obtención de los materiales y equipo necesarios para los ensambles. Sin embargo los diseños libres para crear hardware continúan al alza, basta con hacer una búsqueda en internet e inmediatamente se puede tener acceso a múltiples diseños muy útiles para diversas aplicaciones, con las ventajas de que cada diseño puede estudiarse y modificarse de acuerdo a las necesidades que el diseñador tenga.

1.3 LICENCIAS DE SOFTWARE DE LIBRE DISTRIBUCIÓN

Con el objetivo de garantizar las cuatro libertades esenciales del software libre, se han creado licencias que protegen al software para que este se mantenga libre, de manera que aunque exista la intención de querer violar dichas libertades hayan recursos legales que eviten tales fines.

La FSF en su sitio web muestra numerosas licencias de software libre, pero las mas importantes son las GNU GPL o *GNU General Public Licence,* por sus siglas en inglés. Todas las licencias de software libre que se pueden encontrar en ese sitio buscan compatibilidad con las GNU GPL, de manera que si alguna licencia de software tiene alguna incompatibilidad con ella, ya no se considera software libre.

Es de mucha importancia resaltar que cualquier software que se escribe con el propósito de ser software libre, debe de especificarse clara y explícitamente que lo es, porque de no ser así, se convierte en un software con copyright hasta que no se especifique lo contrario.[6]

Las licencias más importantes de software de libre distribuciones y con las cuales la demás licencias de la misma categoría de software deben de buscar compatibilidad son GPLv1, GPLv2 y GPLv3.

1.3.1 GPLv1

GPLv1 ó General Public Licence version 1, esta versión de acuerdo a la FSF data de febrero de 1989 y en ella se detalla que se procede en dos pasos para poder otorgar al usuario las libertades que para ese tiempo se habían planteado explícitamente. Estos pasos son hacer copyright del software y luego ofrecer la licencia la cual le da permisos legales a los usuarios finales del software a fin de que puedan copiar, distribuir y/o modificar el software.

Con esta licencia de software se garantizaba las libertades de copiar , distribuir y/o modificar el software, y con esto se establecían las tres libertades esenciales del software libre.

Se puede aplicar esta licencia a cualquier programa que la incluya de forma explicita. Básicamente esta licencia da la libertad de que al obtener un software de tipo libre, el usuario debe de tener acceso no solo a los archivos binarios o ejecutables, sino también, si lo desea, a los archivos de código fuente a fin de poder estudiarlo y modificarlo a su conveniencia, para luego, si lo desea, distribuirlo. También la licencia protege al autor del software y lo libra de toda responsabilidad por el uso que se le dé a su obra, es decir, no existe una garantía del funcionamiento del software, ya que por la filosofía y las libertades del software libre, este puede ser modificado innumerables veces por diversos usuarios al rededor del mundo y a la vez distribuido, existiendo la posibilidad de que alguno de los cambios introducidos por alguno de los usuarios cause un mal funcionamiento en la obra original. En caso de que esta situación se de, la reputación del autor original de la obra queda libre de toda acusación.

Se da la libertad de poder obtener una ganancia económica por la distribución del software libre teniendo a opción del distribuidor dar una garantía por el software, y también se invita a los desarrolladores de software a que sus proyectos sean de ayuda a la humanidad compartiéndolos bajo los términos y condiciones de la licencia GLPv1.[7]

1.3.2 GPLv2

Esta versión de las licencias GPL se dio a conocer en Junio de 1991 de acuerdo a la FSF, en ella nuevamente se hace incapíe en que el software libre hace referencia a la libertad y no al precio. Y deja en claro el contraste que existe entre las licencias como la GPLv2 y las demás licencias de software restrictivo, las cuales buscan quitar la libertad de compartirlo. Además en esta licencia de software se extiende la explicación de que si algún usuario hace uso de todas las libertades y beneficios que la licencia GPLv2 ofrece, este esta obligado a extender los mismos beneficios a las demás personas que se le distribuya el software incluso si este ha sido modificado por el usuario en cuestión. De manera que la versión modificada del software estará bajo los mismos términos y condiciones que el software original. Se explica también claramente que de violar esta licencia puede acarrear como consecuencia que se restrinja el derecho de distribuir el software. [8]

1.3.3 GPLv3

La versión tres de la licencia GNU GPL es la última de las revisiones más recientes, presentada el 29 de junio de 2007. Como lo estipulan los términos y condiciones de las licencias GPL anteriores, la FSF puede publicar versiones revisadas o nuevas de

Figura 2: Logo oficial de la GNU GPLv3.

la GPL cada vez que sea necesario aclarar o abordar nuevos problemas o situaciones que surjan siempre y cuando la filosofía del software libre permanezca intacta.

En esta versión de licencia GPL se puede encontrar a primera vista que se ha extendido no solo para el software, sino para cualquier trabajo liberado bajo los términos y condiciones de esta licencia por el autor de la obra. Esta licencia permite que la filosofía del software libre también se extienda a cualquier tipo de obra en la que el autor este de acuerdo en ofrecer las libertades esenciales que caracterizan la filosofía libre. Se aclara también que todo usuario que modifique una obra original, debe de dejar constancia de que lo hizo, de esta manera si el usuario introduce cambios que provoquen errores, estos no serán atribuidos al autor original de la obra. En las versiones v1 y v2 de las GPL ya se había abordado este tema, pero al parecer no se había explicado con suficiente claridad. De hecho esta licencia es la que mejor se explica en comparación con las dos versiones anteriores, ya que incluso en el numeral cero de los términos y condiciones se dedica a explicar las definiciones de las palabras mas relevantes y las cuales pueden ser objeto de confusión o ambigüedad. [9]

1.3.4 OTRAS LICENCIAS DE SOFTWARE DE LIBRE DISTRIBUCIÓN.

Como se mencionó anteriormente, existen muchas mas licencias para el software libre, pero una de las mas importante en la actualidad es la GPLv3 ya que la última versión no se limita solamente a software, sino a cualquier obra que el autor desee liberar bajo la filosofía del software libre, ademas que de que todas las otras licencias de software libre buscan la compatibilidad con ella pero hay diversidad de licencias que son específicas para determinados programas e incluso una licencia completamente permisiva para pequeños proyectos que no rebasen las 300 lineas.

A continuación un pequeño listado de algunas licencias de software libre, cabe resaltar que enumerarlas todas trae consigo una gran dificultad, ya que aun la FSF se limita a mencionar unas pocas y se pone a la disposición de los usuarios que deseen consultar sobre alguna otra licencia que no se encuentre en el listado que mejor se apegue al proyecto que se encuentran desarrollando.

- Licencia Pública General Reducida de GNU (LGPL) versión 3
- Licencia Pública General Reducida de GNU (LGPL) versión 2.1
- Licencia Pública General Affero de GNU (AGPL) versión 3.
- Licencia Completamente Permisiva de GNU
- Licencia Apache, versión 2.0
- Licencia Artística 2.0
- Licencia Artística Clarificada
- Berkeley Database License
- Boost Software License
- Licencia BSD Modificada
- Licencia FreeBSD

1.4 LICENCIAS DE HARDWARE DE LIBRE DISTRIBUCIÓN

Con el surgimiento del hardware de diseño libre también se volvió necesario que se creasen licencias que protejan a los usuarios para que puedan copiar, estudiar, modificar y distribuir dichos diseños, esto con el fin de que nadie se apropie de los diseños y restrinja las libertades que la filosofía del hardware libre ofrece.

De acuerdo con la FSF la mejor manera de promover el hardware de diseño libre es publicándolo en los repositorios de hardware tal y como sucede con el software. Se puede tener acceso a diseños con un desarrollo muy avanzado para aplicaciones también avanzadas, en las cuales los desarrolladores de hardware pueden compartirlos bajo licencias que protegen dichos diseños libres.

En los sitios de internet en los cuales sirven de repositorios de hardware de diseño libre, se puede encontrar muchas veces las licencias que los respaldan [10]. Las licencias que podemos encontrar son básicamente las mismas para el software de libre distribución, ya que su filosofía se puede aplicar a los diseños de hardware tales como las GPLv2 y GPLv3, también otras dos licencias de la FSF como la LGPLv2.1 y LGPLv3.0 las cuales son licencias compatibles con las GPL pero que permiten el enlazado de módulos privativos, y que se recomienda solo usar estas licencias en circunstancias especiales. La FSF en su apartado "*Licenses and Copyright for Free Hardware Designs*" confirma que la GPLv3 fue diseñada para darle uso en el contexto del hardware de diseño libre.[11]

1.5 MICROCONTROLADOR

Un microcontrolador es un dispositivo electrónico capaz de ser programado en varias ocasiones y que puede ejecutar de manera satisfactoria la lógica de control que en él se escriba. Estos dispositivos son diseñados y producidos por diversos fabricantes tales como Microchip, Atmel Corporation, Motorola, Freescale, Texas Instrument, entre otras. Cada microcontrolador tiene una lista de instrucciones útiles que difieren entre las diferentes marca de fabricantes pero que por lo general están destinadas a realizar operaciones aritméticas, lógicas y de comunicación utilizando alguno de los protocolos para los cuales estos dispositivos están diseñados.

Los microcontroladores son muy utilizados en una gran cantidad de dispositivos electrónicos para el hogar, oficina e industria en todas sus ramas. En el hogar pueden encontrarse en dispositivos tales como televisores, control remoto, termómetros infrarrojos, calculadoras, lavadoras y hasta en juguetes. El ámbito de mayor uso para los microcontroladores es por supuesto la industria, en donde existen infinidad de aplicaciones, ya que se realizan procesos repetitivos para tareas específicas. Se pueden encontrar llevando a cabo tareas muy sencillas como activando y desactivando un diodo emisor de luz, hasta controlar variadores de frecuencia, dispositivos lógicos programables como PLC´s, controladores de temperatura, nivel, presión, temporizadores, relés programables etc.

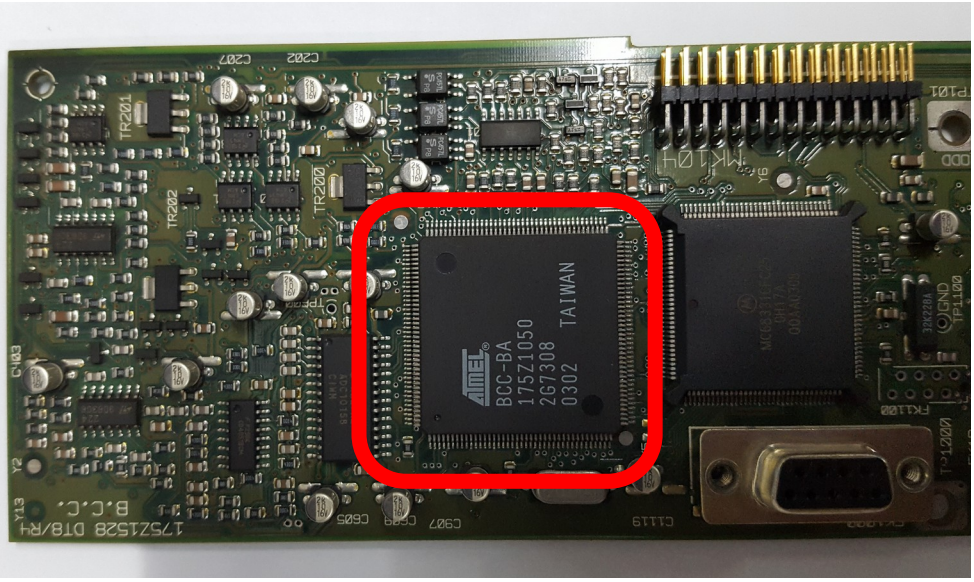


Figura 3: Microcontrolador Atmel en placa de control de VFD Danfoss VLT 6000 HVAC.

Los microcontroladores integran una unidad central de procesos (CPU), unidad aritmética lógica (ALU), puertos de entrada/salida digitales, convertidores analógico/digital, así como unidades de memoria destinadas al almacenamiento del programa y memoria de datos. Algunas de estas unidades se encuentran interconectadas mediante el bus de direcciones y/o bus de datos. No debe confundirse un microcontrolador con un microprocesador, ya que ambos tienen diferencias notorias que los hacen únicos. Por ejemplo, a un microprocesador se le deben adicionar unidades de memoria externa, dispositivos de entrada/salida y demás periféricos que lo hagan funcional como en el caso de las computadoras portátiles o las de escritorio, ademas son utilizados para aplicaciones de propósito general en los que se debe de realizar gran cantidad de procesamiento y por este hecho es que necesita también grandes cantidades de memoria de acuerdo a las tareas que deba de ejecutar.

Por otro lado un microcontrolador es utilizado para aplicaciones específicas, y a diferencia del microprocesador integra todos los periféricos necesarios, CPU, ALU, Memorias, bus de direcciones y bus de datos, es decir, un microcontrolador tiene en su interior un microprocesador. Ya que un microcontrolador posee todas estas características se tiene la posibilidad de montarlo fácilmente en una placa electrónica o dispositivo con un mínimo de periféricos externos y programarlo para una tarea determinada.

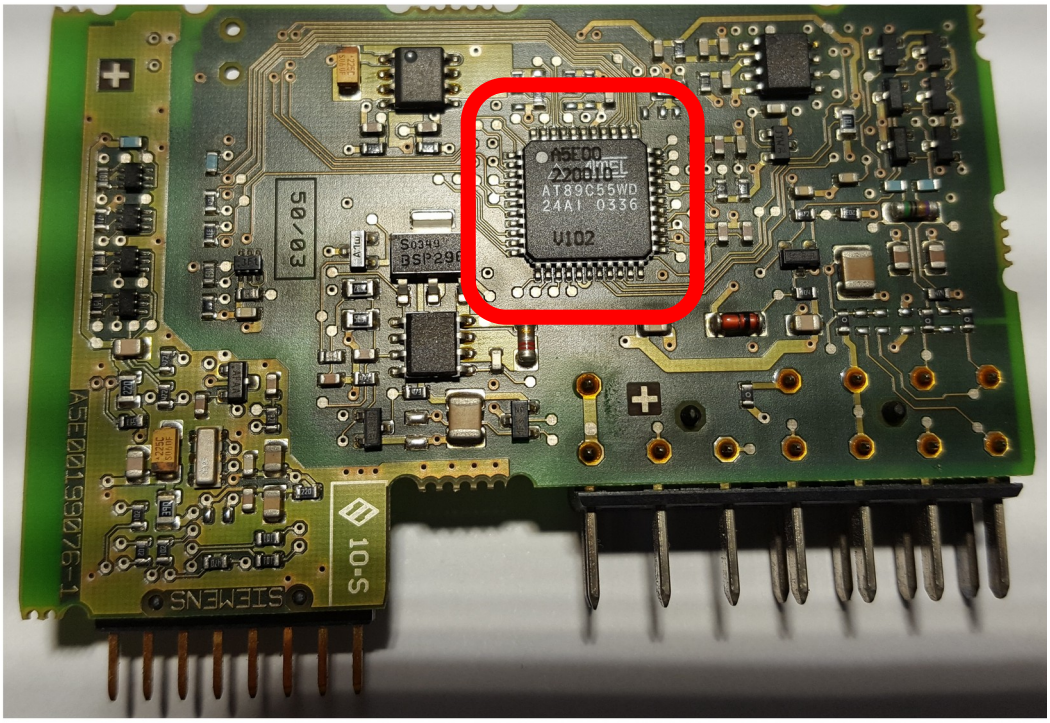


Figura 4: Microcontrolador Atmel en módulo de entradas análogas 6ES7 134-4GB-0AB0 para PLC SIMATIC S7 de Siemens.

El el campo de los microcontroladores existen dos arquitecturas más utilizadas con sus diversas ventajas y desventajas para que la CPU pueda accesar a las memorias de datos y de programa, estas arquitecturas son: Von Neuman y Harvard. En la arquitectura Von Neuman, desarrollada en la universidad de Princeton, y una de las que mas se aplicaron en las primeras computadoras inclusive en las actuales, se utiliza un solo bus para direcciones y datos a la misma vez, esto provoca que haya una baja velocidad de procesamiento en la CPU, debido a que debe de esperar a que primero se cumpla el acceso a la memoria de programa y se termine el proceso, para luego poder accesar a la memoria de datos, este proceso de espera entre accesos a memoria son efectos indeseables cuando lo que se busca es velocidad de procesamiento. Sin embargo se tiene la ventaja que al tener un solo bus para acceso, significa que se usan menor número de líneas de entrada/salida.

Por otra parte la arquitectura Harvard, desarrollada en la universidad del mismo nombre, tiene la ventaja de que posee buses separados para acceso a las memorias de datos y de programa, lo que resulta en una más alta velocidad de procesamiento, ya que para implementar esta arquitectura se requiere de cuatro conjuntos de buses, el primer conjunto para transportar los datos dentro y fuera de la CPU, el segundo es un bus de direcciones para acceder a los datos, el tercero transporta el programa a la CPU y el cuarto es un bus de direcciones para accesar a la memoria de programa. Aparentemente la arquitectura Harvard tiene la ventaja de la velocidad de procesamiento sobre la arquitectura Von Neuman, pero utiliza una gran cantidad de lineas por cada bus, en computadoras personales esto no es una opción muy práctica pero en microcontroladores esta arquitectura resulta favorable, ya que todos la CPU, ALU, periféricos y demás componentes del microcontrolador se encuentran encapsulados en un

solo dispositivo[12]. Por ejemplo los microcontroladores AVR de Atmel Corporation utilizan la arquitectura Harvard, en su hoja de datos dice textualmente:

"Para maximizar el rendimiento y el paralelismo, el AVR utiliza una arquitectura Harvard, con memorias y buses separados para programas y datos. Las instrucciones en la memoria del programa se ejecutan con una canalización de un solo nivel. Mientras se ejecuta una instrucción, la siguiente instrucción se recupera previamente de la memoria del programa. Este concepto permite ejecutar instrucciones en cada ciclo de reloj. La memoria del programa es una memoria flash reprogramable en el sistema".[13]

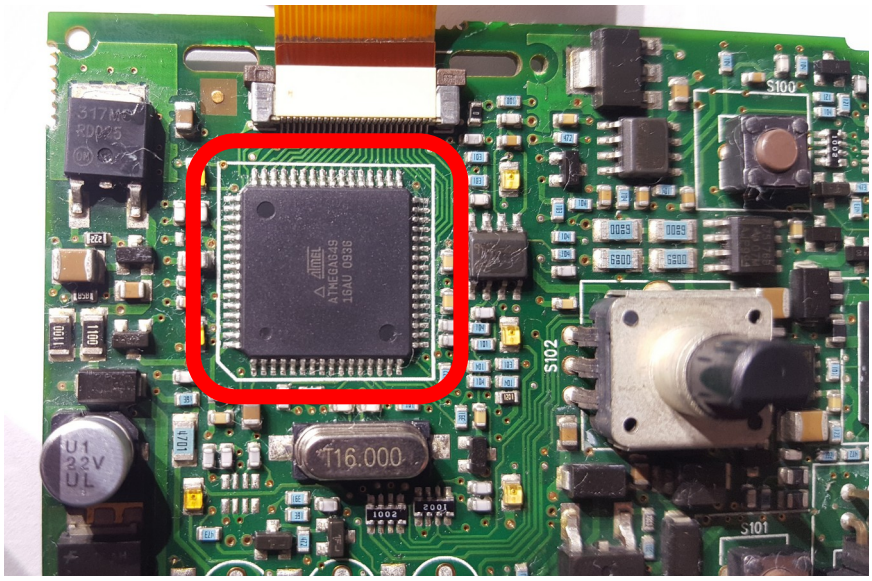


Figura 5: Microcontrolador ATmega en placa electrónica de control de VFD VACON® 20 AC Drives

1.6 PROTOCOLO DE COMUNICACIÓN I2C.

El I2C, IIC o I^2C, es un protocolo de comunicación creado por la empresa Phillips para comunicar diferentes circuitos integrados o dispositivos dentro de los aparatos electrodomésticos que fabrica como televisores y otros en donde el uso de este protocolo fuera útil. Fue introducido en el año 1982 y presentaba ciertas ventajas tales como, que a través de un bus de dos hilos se podía comunicar un gran número de dispositivos conectados a dicho bus y debido a esto se reduce en gran manera el número de líneas a utilizar en un determinado circuito electrónico. El protocolo de comunicación I2C tiene la característica de funcionamiento en modo maestro-esclavo, es decir, un dispositivo llamado maestro es el encargado de iniciar la comunicación, enviando las ordenes que los dispositivos esclavos deben de ejecutar para luego responder al maestro cuando la tarea fue realizada.

Cada uno de los dispositivos conectados al bus I2C posee una dirección con la cual se identifica dentro de la red de dispositivos, de esa manera se pueden diferenciar las ordenes correspondientes para cada dispositivo esclavo. Los dispositivos esclavos pueden ser del modo receptor o transmisor, es decir,

algunos de ellos tienen la capacidad de reportarle al maestro cierta información recolectada y otros simplemente ejecutan las ordenes del maestro.

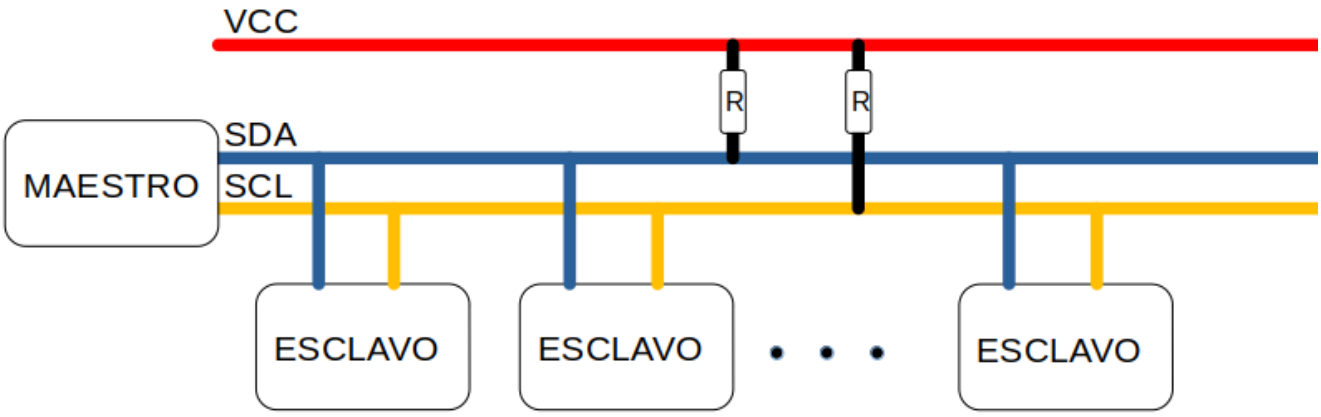


Figura 6: Esquema del bus I2C.

Dentro de los bits que conforman la trama de datos en la comunicación del bus I2C, se tiene que existen siete bits de direccionamiento, con lo cuales se tiene la posibilidad de 128 direcciones posibles, lo que significan 128 dispositivos conectados al bus ($2^7 = 128$).

Una de las desventajas que tiene este bus es que al encontrarse en lugares con demasiado ruido eléctrico se tiende a tener interferencias, también se puede mencionar que este bus solo se usa cuando las distancias no son tan largas , debido a que también se ve afectado por las largas distancias y debido a que por lo general se tiene una configuración de un maestro y muchos esclavos se pueden producir retrasos en el envío y recepción de los datos.

El bus I2C puede alcanzar velocidades de transferencia entre 100kbit/s en el modo estándar y de 3.4Mbit/s en el modo de alta velocidad.

Las dos líneas que conforman el bus de comunicación son SDA(Serial DAta) y SCL(Serial CLock), la primera se encarga de transportar la información entre los distintos dispositivos conectados, mientras que la segunda línea la cual es controlada por el maestro, mantiene a todos los dispositivos sincronizados.

1.7 ARDUINO

Arduino es una plataforma de hardware y software que consta de una serie de placas electrónicas basadas en microcontroladores AVR los cuales pueden ser programados una gran cantidad de veces. Dichas placas son catalogadas como placas de desarrollo o para prototipos, es decir, muchos proyectos se pueden iniciar en esta plataforma debido a su bajo costo y a la facilidad de programación que presenta.

Figura 7: Logo de Arduino.

Los diseños de hardware de las placas Arduino pueden encontrarse en numerosos sitios de internet principalmente en el sitio web de Arduino [14], el cual pone a disposición de la comunidad en general los diseños esquemáticos y para circuito impreso (PCB).

Buen número de las especificaciones técnicas de las placas Arduino dependen del microcontrolador que poseen, por ejemplo para el Arduino UNO se tienen las siguientes:

- Microcontrolador: ATmega328P
- Voltaje de operación: 5V
- Voltaje de entrada(recomendado): 7-12V
- Voltaje de entrada límite: 6-20V
- Pines digitales de E/S: 14 (de los cuales 6 pueden ser utilizados como salidas PWM)
- Pines de entrada análogos: 6
- Pines digitales PWM de E/S: 6
- Corriente DC para pines de E/S: 20mA
- Memoria Flash: 32KB(de los cuales 0.5KB son utilizados para el cargador de arranque)
- Memoria SRAM: 2KB
- Memoria EEPROM: 1KB
- Velocidad de reloj: 16MHz

Figura 8: Placa electrónica de desarrollo Arduino UNO

La plataforma Arduino ha alcanzado una gran popularidad debido a que con el paso de los años, desde que se inició su desarrollo, se ha vuelto mucho más accesible en muchos países del mundo, lo que le permite tanto a personas como a instituciones educativas poder adquirirlas para iniciar a los estudiantes en el software y hardware de libre distribución.

De acuerdo con las estadísticas de Google Trends, la tendencia en la búsqueda de temas sobre Arduino ha ido en aumento a nivel mundial desde septiembre de 2007 como se puede observar en el gráfico de la figura 8.

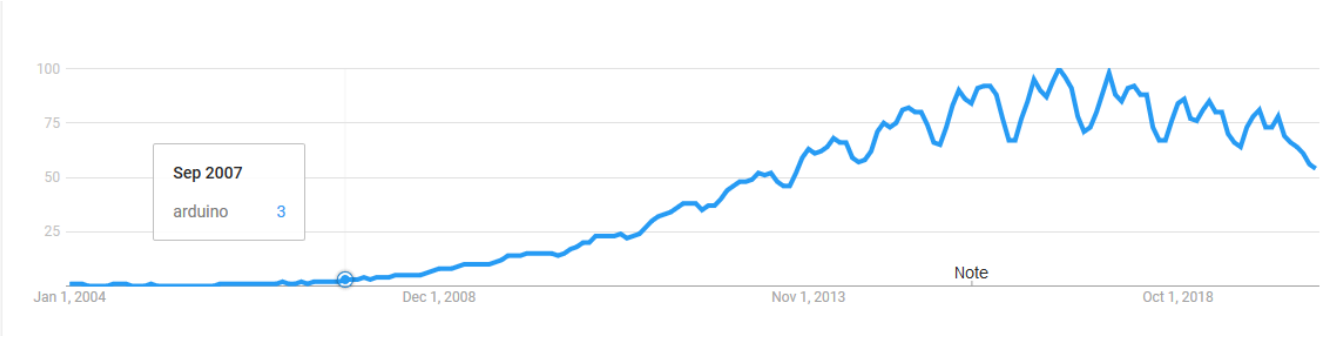


Figura 9: Búsquedas relacionadas con Arduino desde 2004 a agosto 2020. Gráfico de Google Trends

Dentro de todo el conjunto de placas electrónicas disponibles, la placa Arduino UNO es la que goza de mayor popularidad, ya que su precio es accesible y posee un número de entradas/salidas que le permiten al desarrollador poder crear diversos prototipos antes de implementar un diseño final. Ademas de la Arduino UNO, también existen otras placas con mas puertos de entrada/salida como la placa Arduino Mega 2560 y también placas más pequeñas como la Arduino nano. El diseñador de hardware debe elegir la que mejor se adapte a sus necesidades. Las tendencias de búsqueda de Google Trends demuestran que los usuarios de internet familiarizados con este tema tienen mas interés en la información acerca de la placa Arduino UNO comparada con la Mega 2560 y la Arduino Nano.

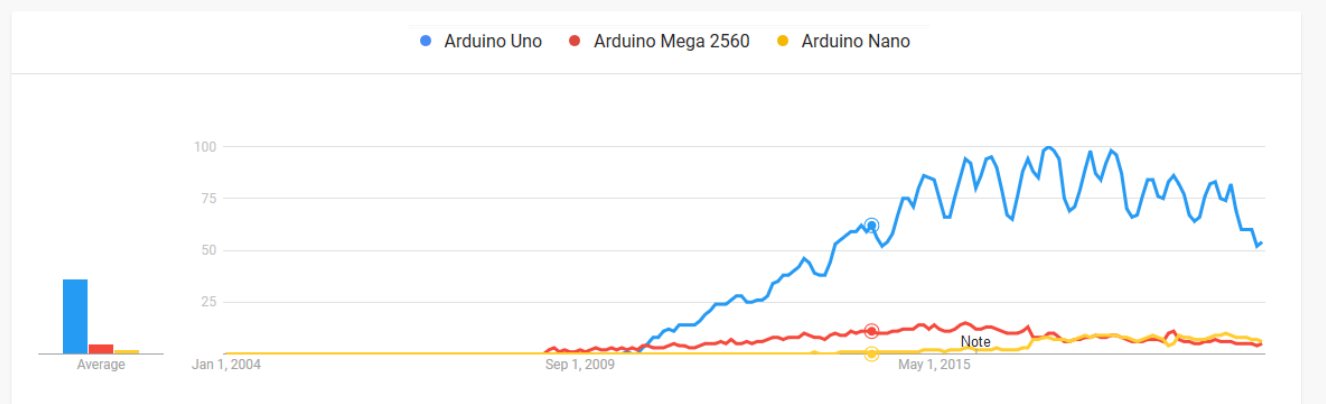


Figura 10: Tendencia de búsqueda de información de placa Arduino UNO comparada con las placas Mega 2560 Nano. Gráfico de Google Trends.

1.8 ARDUINO IDE

La plataforma de desarrollo Arduino utiliza los microcontroladores AVR de Atmel Corporation, dichos microcontrolador en su hoja de datos tienen una serie de instrucciones para realizar todos los procedimientos necesarios de operaciones lógicas, aritméticas y de comunicación.[13] Las instrucciones con las que trabaja el microcontrolador están originalmente en un lenguaje ensamblador específico para este tipo de microcontroladores, esto supone un poco mas de dificultad para los programadores menos experimentados en esa área. Una de las razones por las que la plataforma

Arduino se ha hecho tan popular es debido a que goza de la fama de tener una manera mucho mas accesible para programar, es decir, no se utilizan el listado de instrucciones en ensamblador del microcontrolador directamente, sino que se hace uso de un lenguaje de programación relativamente más sencillo parecido al lenguaje de programación C/C++.

Para programar un microcontrolador de la serie Mega de Atmel como los que se utilizan en las placa de desarrollo Arduino se puede hacer uso del entorno de desarrollo integrado (IDE) el cual se puede encontrar para su descarga y utilización en el sitio de internet de Arduino, en el se puede encontrar muchas funcionalidades como seleccionar de en medio de un gran número de placas a las cuales se puede subir el programa editado, una opción para verificar errores de sintaxis y luego compilar, selección del puerto de la computadora en la que se conectará la placa Arduino a través de un puerto USB, monitor serial para leer/escribir en el microcontrolador en tiempo de ejecución del programa y una opción para escribir el cargador de arranque en la memoria del microcontrolador en caso de utilizar un nuevo microcontrolador Atmega. [13]

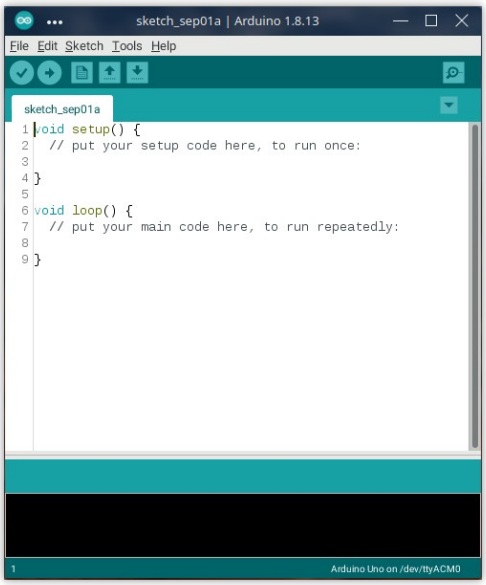


Figura 11: Interfaz gráfica de usuario de Arduino IDE.

El lenguaje de programación de Arduino, se basa en un proyecto similar llamado *Wiring*, de hecho el hardware y los programas para dicho proyecto son similares a los de Arduino [15] .

Cuando se hace uso del IDE de Arduino, por defecto muestra en un nuevo documento las funciones *setup* y *loop*, en el primero se definen las funciones que tendrán los pines del microcontrolador, ya sea de entrada o salida, también se define si habrá comunicación serial o se hará uso de algún hardware externo como por ejemplo un sensor. En pocas palabras en la función *setup* se define un punto de inicio del programa para preparar el ambiente en el que este funcionará. Esta función solo se ejecuta una vez al inicio del programa y una vez ejecutada da paso a la función *loop*. La función *loop* es en donde esta escrita la lógica del programa, todos los procedimientos, acciones y decisiones se llevan a cabo en esta función, a diferencia de la función *setup*, ésta se repite continuamente.

1.9 AUTODESK EAGLE

Autodesk Eagle es una herramientas de software para diseño de circuitos electrónicos que puede ser utilizado en las plataformas de software más utilizadas como GNU/Linux, Windows y MacOS. Este tipo de software

Figura 12: Logo de Eagle Autodesk

es conocido como EDA (Electronic Design Automation) y dentro de sus funciones esta el poder diseñar circuitos electrónicos en forma de esquemático y a partir de ellos generar el diseño del circuito impreso (PCB). Los componentes electrónicos que se pueden encontrar son muy variados y tienen el nombre de librerías, estas últimas se pueden encontrar ya incluidas en el software y de ser necesario también existe la posibilidad de descargar de internet librerías extra o incluso poder crearlas en una aplicación para este fin dentro del programa.

Este tipo de software permite el diseño de circuitos electrónicos sencillos, hasta proyectos de gran complejidad. Cada proyecto contiene los archivos necesarios para incluso poder ser manufacturados y realizar un PCB profesional o también se tiene la opción de imprimir el diseño de manera que se pueda hacer un PCB casero utilizando herramientas que no requiera una gran complejidad operar o adquirir.

Eagle permite que los diseños sean con componentes de montaje superficial (SMD), así como también con los típicos componentes de montaje DIP o PDIP.

1.10 GNU/LINUX

GNU/Linux es una serie de sistemas operativos que se desarrollaron como alternativa libre a los sistemas UNIX. Inicialmente GNU y Linux se desarrollaron como proyectos separados, por una parte GNU era desarrollado por Richard Stallman y la FSF, escribiendo aplicaciones o programas para evitar el uso de los programas de software privativo, mientras que Linux era desarrollado principalmente por Linus Torvalds, un estudiante universitario finlandés que escribía el código fuente de lo que sería el núcleo de un sistema operativo. Ambos proyectos al ser liberados bajo licencias que permitían a los desarrolladores de software alrededor del mundo poder acceder a su código fuente, fueron creciendo de manera que cuando el proyecto GNU ya se encontraba avanzado (aunque incompleto) en cuanto a aplicaciones y algunos controladores de dispositivos, el núcleo de Linux también se encontraba en una fase de desarrollo que les permitía utilizar dicho núcleo para las aplicaciones del proyecto GNU, de manera que al fusionar ambos proyectos en uno solo se obtuvo un sistema operativo funcional.[16]

Actualmente la comunidad, como es llamado el grupo de desarrolladores de software, traductores y artistas gráficos que mantienen el proyecto actualizado, hacen sus aportes a través de internet, escribiendo código fuente, haciendo revisiones y traducciones, también desarrollando nuevos programas de distintas categorías como programas para diseño gráfico, multimedia, ofirmatica, software especializado para ingeniería, etc.

Figura 13: Logo de RedHat

Cada una de las categorías de software libre que se han mencionado también se reúnen muchas veces para crear un sistema operativo funcional y práctico haciendo uso del núcleo de Linux, o también llamado *kernel*, ya que este último les brinda a los desarrolladores acceso completo al código fuente, para que de esta manera cada distribución de software, conocidos como sistemas operativos GNU/Linux, pueda adaptar las características que mejor les parezcan.

Es importante mencionar que algunas distribuciones de GNU/Linux no se pueden obtener de forma gratuita ya que son versiones especializadas que integran software libre y privativo, como controladores o aplicaciones especializadas, también ofrecen soluciones para servidores y computadoras de escritorio, así como entrenamiento, certificación y soporte técnico por un coste mensual, anual, o en los períodos que tanto los clientes como las empresas consideren necesarios.

Los ejemplos mas ampliamente conocidos de este tipo de sistemas operativos son RedHat Enterprice Linux y SUSE Linux, ambas ofrecen variedad de servicios los cuales son de paga, pero también ambas distribuciones han liberado su código fuente de manera que se cuenta con versiones libres basadas en las versiones de paga.

Figura 14: Logo de SUSE Linux

Para el caso de la distribución RedHat Enterprice Linux se encuentra disponible la versión libre la cual lleva por nombre *Fedora*, y para la SUSE Linux, la distribución libre es openSUSE Linux, ambas distribuciones libres cuentan con muchas características de la versión de paga, pero no poseen todos los paquetes de software restrictivo y soporte técnico que su similar de paga. Las versiones libres son desarrolladas por una comunidad de programadores, diseñadores gráficos, traductores y especialistas que hacen su aporte al rededor del mundo para liberar cada determinado tiempo la versión mas estable del sistema libre.

Las personas involucradas en el desarrollo de las distribuciones libres se encargan de estudiar el código fuente, modificarlo, mejorarlo y ejecutarlo junto a muchos usuarios que reportan continuamente los errores o fallos encontrados a fin de presentar al público una versión estable.

1.11 ZIMMER MASCHINENBAU GMBH

Zimmer Maschinenbau GmbH es una empresa de origen austríaco dedicada al desarrollo y fabricación de maquinaria dirigida a la industria textil. Fundada en 1874 por Franz Zimmer, Zimmer Maschinenbau se ha mantenido operando por mas de 140 años desarrollando nuevas tecnologías que facilitan la producción de tejidos impresos, lo que le ha permitido expandir sus fronteras comerciales por todo el mundo en donde sus maquinarias gozan de alta demanda.[18]

Figura 15: Logo de Zimmer AUSTRIA

Zimmer se especializa en la invención, innovación y mejora de impresoras textiles de tipo industrial. Dentro de las maquinarias con tecnologías mas demandadas están las de impresión por el método de serigrafía rotativa y el de serigrafía plana.

Además de las maquinarias de tecnología de impresión, Zimmer también desarrolla y produce maquinaria complementaria para el proceso de estampado de tejidos textiles, tales como hornos secadores, vaporizadores, y lavadoras de aplicadores de pastas de estampación.

1.12 ROTA-TG/116

La ROTA-TG/116 es una maquinaria de fabricación austriaca para la industria textil, la cual posee doce estaciones especializadas en el proceso de serigrafía rotativa las cuales pueden ser ampliadas hasta 24 estaciones de trabajo. Cada estación es conocida con el nombre de cabezal de estampación. De acuerdo con las especificaciones técnicas de la maquinaria, esta es capaz de alcanzar una velocidad de 120m/s [19], lo que significa que en un solo día es capaz de estampar miles de kilogramos de tejido.

Figura 16: Maquinaria estampadora ROTA-TG/116[19]

La maquinaria consta de tres partes esenciales las cuales se encargan de una función específica dentro del proceso de estampación, dichas partes se describen a continuación:

Figura 17: Entrada de tejido de ROTA-TG/116[19]

La entrada del tejido, en la cual los operadores ubican el tejido en forma de bobina para ser estampado, el tejido por lo general es de color blanco, aunque este puede cambiar de acuerdo al color predominante en un diseño, el cual será la base de color en el diseño. El tejido es dirigido hacia una serie de rodillos especiales que evitan que las orilla de la tela se enrolle a la vez que pasa por unos rodillos con un recubrimiento que genera fricción entre la tela y el cilindro, de manera que se genera un efecto de tracción.

En esta etapa también existe un sistema de extracción de pequeños residuos de tela, polvo y pelusa a fin de que al ser aplicada la tinta, esta última no se adhiera a la contaminación del tejido, sino al tejido limpio.

La segunda etapa de la maquinaria es conocida como mesa de estampación. Esta consiste en una cinta recubierta con una mezcla pegajosa la cual permite que el tejido se adhiera firmemente y sin arrugas para luego pasar por cada uno de los cabezales, en los que se fijan unos rodillos huecos de aluminio por donde fluye la tinta hacia el tejido, dejando impreso el diseño que se transfiere del rodillo.

Figura 18: Mesa de estampación de la ROTA-TG/116[19]

Finalmente la tela estampada aun húmeda por la tinta pasa a la tercera etapa de la máquina, un horno en cuyo interior la temperatura alcanza alrededor de 200°C, el cual bien puede ser calentado por la combustión de gas propano o por la circulación de aceite térmico en el interior de sus radiadores de calor. El tejido ingresa en esta etapa y tras una corta permanencia en su interior, el tejido sale totalmente seco, listo para posteriores procesos.

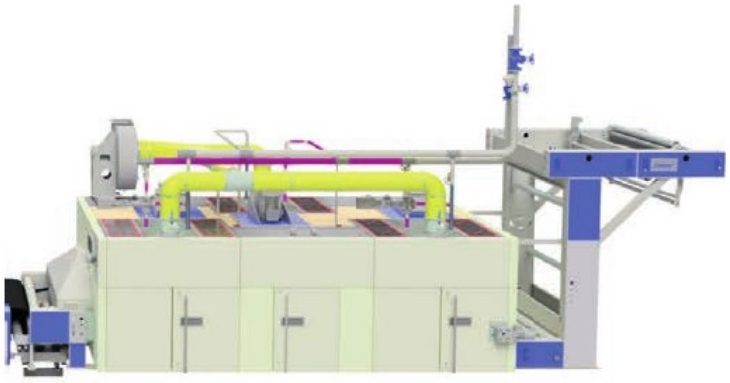


Figura 19: Horno en la salida de la ROTA-TG/116[19]

CAPITULO 2: ANÁLISIS DEL PROCESO Y DE CAUSA DE FALLAS.

2.1 DESCRIPCIÓN DEL PROCESO DE ESTAMPADO

Dentro del proceso de producción de textiles, cabe la opción por parte de los clientes de solicitar un diseño estampado en las distintas calidades de telas, para este fin el cliente debe de enviar el diseño con el número de colores de su elección. Cabe recalcar que para este tipo de maquinaria estampadora como la Zimmer ROTA-TG/116, no existe la opción de poder crear colores a partir de la combinación de los mismos en tiempo de estampación, ya que esos tipos de estampado lo realizan otros tipos de maquinarias textiles con un solo cabezal similar al de una impresora de inyección de tinta para papel de uso común, de las que se encuentran en uso doméstico o de oficina, con la diferencia en que su tamaño y complejidad son mayores. Para el caso de la ROTA-TG/116 solo se tiene la opción de elegir doce colores, es decir, diseños que contengan patrones en los cuales como máximo solo pueden usarse doce colores. Esto es debido a que la maquinaria físicamente solo consta de doce estaciones llamadas cabezales de estampación, en los cuales se instalan los cilindros con los cuales se realiza el proceso.

El diseño, también llamado dibujo, debe de ser procesado en un software especial provisto por el proveedor de la maquinaria, de manera que separe el diseño original en patrones de acuerdo a sus distintos colores, por ejemplo el siguiente mostrado en la figura 20.

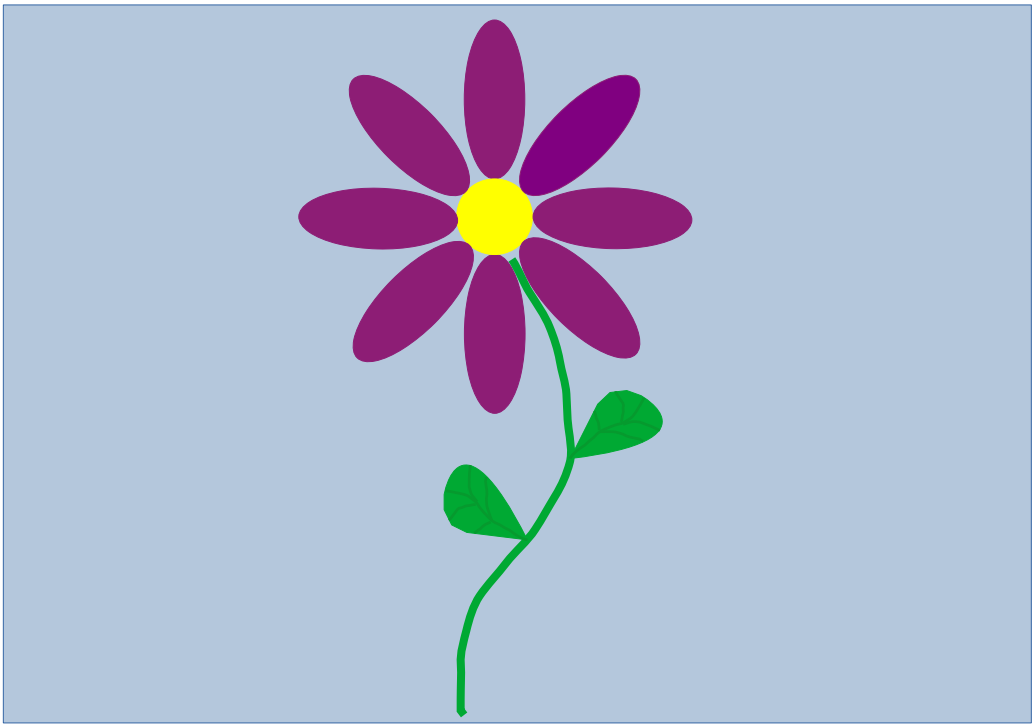


Figura 20: Diseño de cinco colores y cuatro patrones.

En el diseño sencillo mostrado en la figura 20, se puede observar que posee cinco distintos colores, y cuatro distintos patrones que conforman el diseño final. El software de procesamiento de imágenes debe de separar el diseño en sus distintos colores, con el fin de que cada color y su patrón respectivo

sean perforados en un cilindro distinto cada uno. Se puede observar en la figura 21 la separación de los colores en sus respectivos patrones.

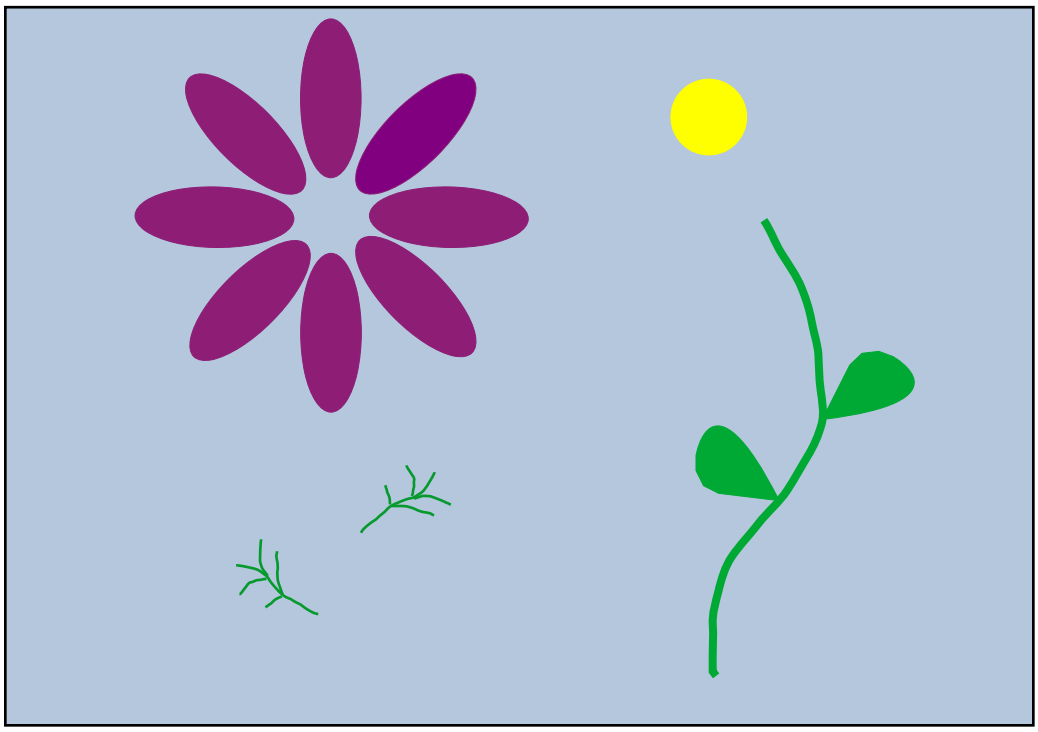


Figura 21: Diseño separado en sus distintos colores.

Tal como se puede apreciar se han separado los cuatro patrones por color, púrpura, verde oscuro, verde claro y amarillo, el fondo celeste que sirve de base es el tejido teñido de ese color, y por lo tanto no se considera un patrón más. El propósito de esta separación radica en que cada patrón por color se va a grabar en un cilindro conocido como cilindro de estampación haciendo uso de una máquina llamada grabadora láser, la cual crea los distintos patrones realizando minúsculas perforaciones de forma circular en el cilindro fabricado con una delgada lámina de aluminio color amarillo. Se debe asegurar de que cada patrón tenga continuidad, es decir, que los patrones de formas son grabados de manera tal, que justo donde inicia el dibujo, también termina el mismo, asegurándose de evitar la superposición del dibujo sobre si mismo, o mejor dicho, el diseño será impreso en el cilindro, por lo que el software especializado debe de acomodar el patrón de modo que al girar el cilindro en tiempo de estampación no se traslape la imagen, sino que por cada rotación completa se logre la impresión completa del diseño.

El proceso de acomodación del diseño a ser perforado en el cilindro, lo lleva a cabo el software especializado de la maquinaria láser la cual realiza los cálculos necesarios después de haber concluido un mapeo del patrón, para luego generar un archivo de coordenadas, y mediante un potente láser apuntando a cada coordenada perfora la pantalla de aluminio del cilindro a la vez que este gira a gran velocidad, grabando en ella un conjunto de pequeños agujeros cuyo arreglo se conoce con el nombre de

malla (*mesh)*. Dicha malla contiene las perforaciones distribuidas de manera que la distancia entre agujeros no deje espacios en blanco a la hora del proceso de estampación. La calidad del producto es directamente proporcional a la calidad del cilindro de estampación.

Como ejemplo, para los pétalos de la figura 21 se tendría como resultado algo similar a lo mostrado en la figura 22. El color amarillo es el tono del color del material del cilindro.

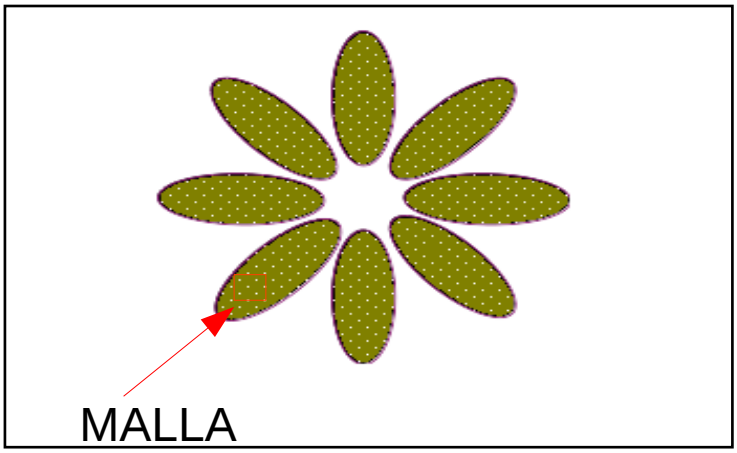


Figura 22: Diseño perforado en cilindro de estampación.

El proceso de perforación láser se debe de realizar por cada color que el diseño contenga, lo que significa que para el diseño mostrado en la figura 20, se debe de realizar este procedimiento en cuatro distintos cilindros, uno para el color purpura, uno para el verde claro, uno para el verde oscuro y por último uno para el color amarillo. Pero se dan casos en los cuales existen diseños de doce colores.

En cada uno de los agujeros perforados se filtrará la tinta que será luego inyectada por la bomba de tinta en el proceso de estampación al rotar el cilindro, este proceso es conocido como serigrafía rotativa.

Uno de los detalles que es necesario entender es que cada cilindro de estampación es hueco, es decir, se trata de una lámina de aluminio, dispuesta en forma de tubo, de esta manera, la tinta se vierte en su interior, a la vez que una barra metálica se introduce en el cilindro hueco, y es atraída por unos potentes electroimanes de manera que al girar el rodillo, la barra metálica presione la tinta contra las paredes internas del cilindro, provocando así la salida de la tinta por medio de los agujeros perforados en el proceso descrito con la máquina láser. Entre menor sea el diámetro de la barra, la calidad de aplicación de la tinta y la penetración de esta en el tejido también será menor, siempre considerando que el diámetro máximo de la barra no debe de sobrepasar los veinticinco milímetros.

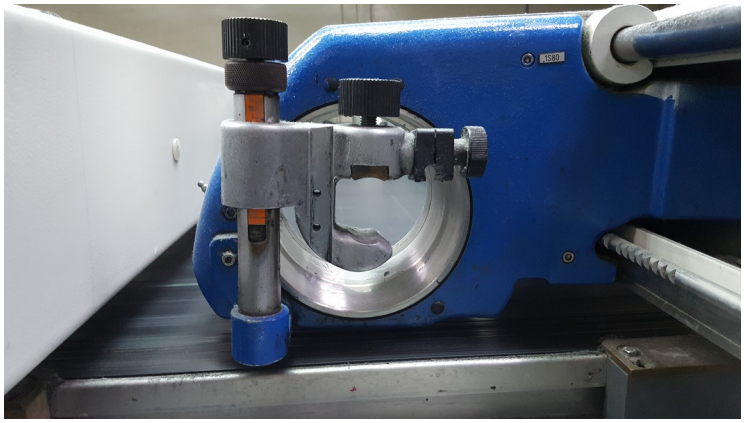


Figura 23: Vista lateral del cabezal donde se fijan los cilindros de estampación.

2.2 AJUSTE LATERAL

En tiempo de estampación es muy común que al imprimir el diseño, algunos de los patrones no coincida perfectamente con los demás colores en el diseño total, quedando superpuesto a otro color y provocando producto de mala calidad.

Por lo general no se da con mucha frecuencia que el diseño perforado en el cilindro no tenga errores en la grabación del patrón, aunque no está exento de pequeños errores, pero más bien las fallas de estampación por desfase y superposición de los colores como se mencionó anteriormente se deben a errores humanos a la hora de fijar el cilindro en el cabezal de estampación. Es en este caso en donde se vuelve importante hacer ajustes precisos para lograr que cada patrón y color encajen perfectamente en el diseño. Un ejemplo de este tipo de fallas de estampación se puede observar en la figura 24.

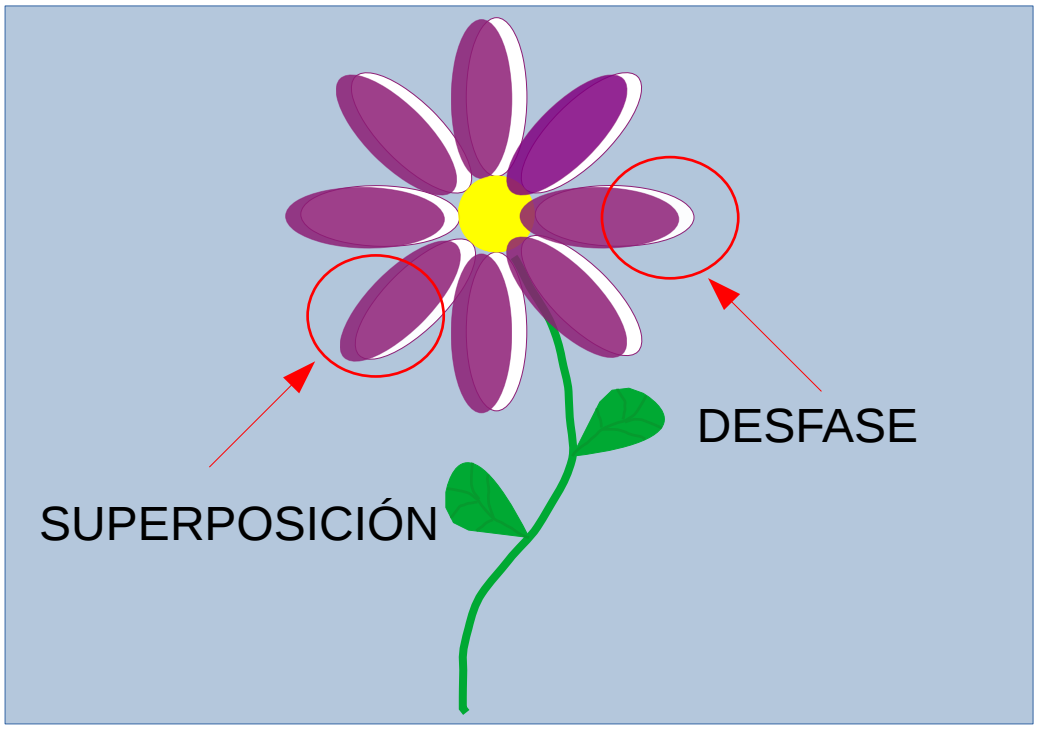


Figura 24: Defecto por desviación de alineación lateral.

Como se puede observar en la figura 24, el cilindro correspondiente al patrón del color purpura, se encuentra mal ubicado, es decir, esta movido a la izquierda, debido a que se ha producido una superposición, que a la vez da como resultado un desfase del patrón con respecto al diseño total. Este defecto tiene el nombre de defecto por desviación de alineación lateral.

Para efectos demostrativos se ha exagerado el efecto de superposición y desfase, pero en la práctica estos defectos son mucho menores (aunque con posibilidad de ser extremadamente grandes, hasta 20mm), ya que incluso un valor tan pequeño como 0.5mm de estos defectos son considerados como producto que perdió su categoría de alta calidad y por lo tanto pasa a categorías más bajas que también le restan su valor de venta. La corrección de este problema es posible, mediante el movimiento del cilindro completo, moviendo el cilindro hacia la derecha o hacia la izquierda en tiempo de estampación, a esto se le conoce como ajuste lateral, el cual permite mover cuanto sea necesario el cilindro hasta lograr que el diseño quede alineado completamente evitando la superposición de los colores. La figura 25 muestra una representación sencilla de cómo funciona el ajuste mencionado.

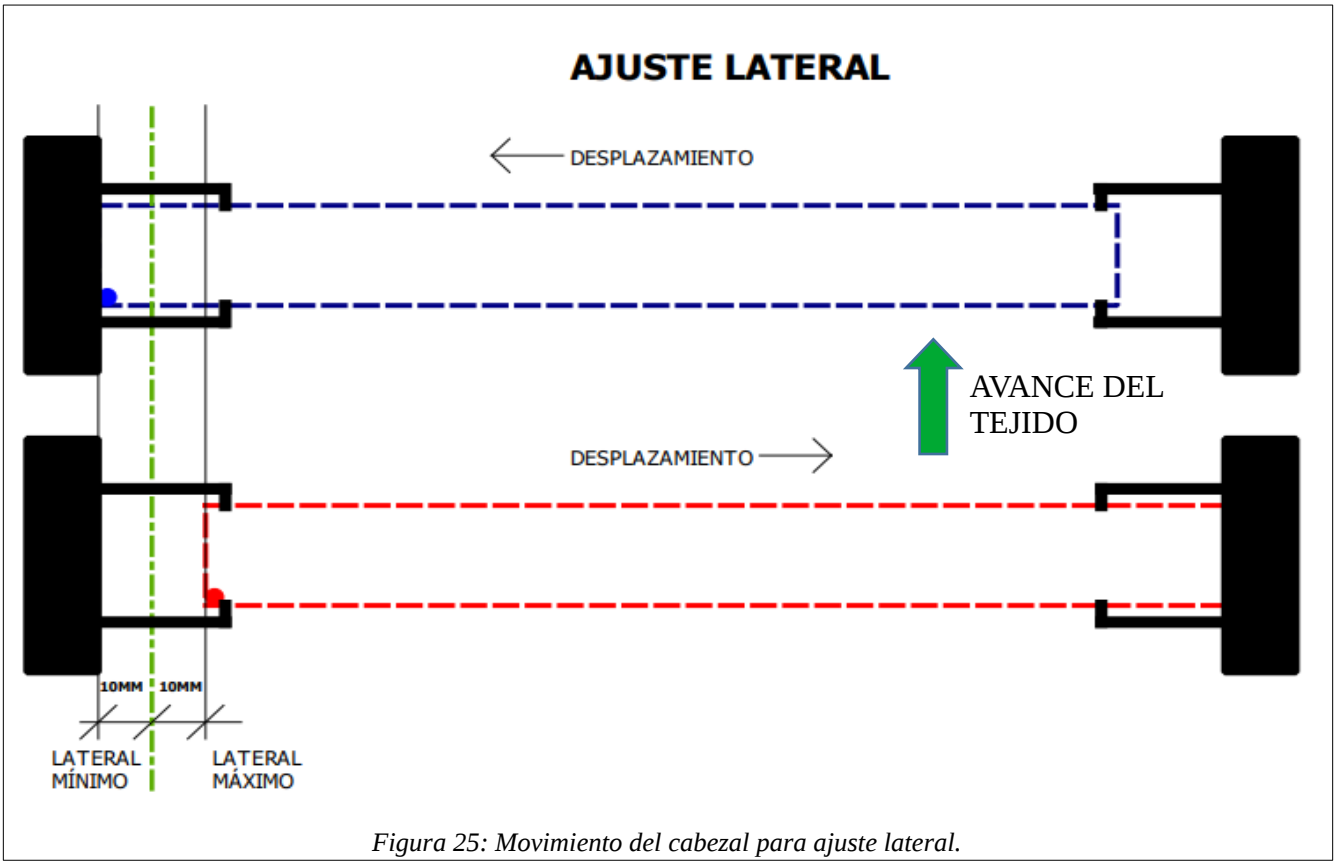


Figura 25: Movimiento del cabezal para ajuste lateral.

En la figura 25 se puede observar que las partes de color negro, representan piezas del cabezal de estampación que sostienen el cilindro perforado para la serigrafía rotativa. Las líneas punteadas representan al cilindro por donde se transfiere a presión la tinta al tejido, para el caso del cilindro en color azul se trata de la posición lateral mínima que puede se puede alcanzar y en complemento el

cilindro en color rojo representa la posición lateral máxima a la que se puede llegar en el ajuste lateral. Tal como se mencionó entre la posición mínima y la máxima hay un rango de 20mm, este es el valor ajustable en el cilindro de estampación.

2.3 AJUSTE DIAGONAL

El ajuste diagonal, en este contexto, no es más que una variante del ajuste lateral descrito en la sección dos del capítulo dos (2.2 AJUSTE LATERAL) de este reporte de trabajo de graduación.

Debido a que con frecuencia se presentan los errores humanos al intervenir en las tareas que no se pueden realizar de manera automática, tales como la fijación del cilindro al cabezal de estampación, al igual que las fallas por corrimiento a la izquierda o a la derecha del cilindro, las cuales se pueden corregir con el ajuste lateral provisto por el sistema *Navigator Automata*, también son posibles los defectos producto del corrimiento en diagonal del cilindro. En teoría el cabezal de estampación y por ende el cilindro perforado para la serigrafía rotativa, deberían de permanecer perpendiculares a la dirección de avance del tejido en el cual se está trabajando, pero esto no es posible cuando ha ocurrido un problema de alineación del diseño. En este caso se obtienen defectos por desviación de alineación diagonal, y como se observa en la figura 26 los resultados son los siguientes:

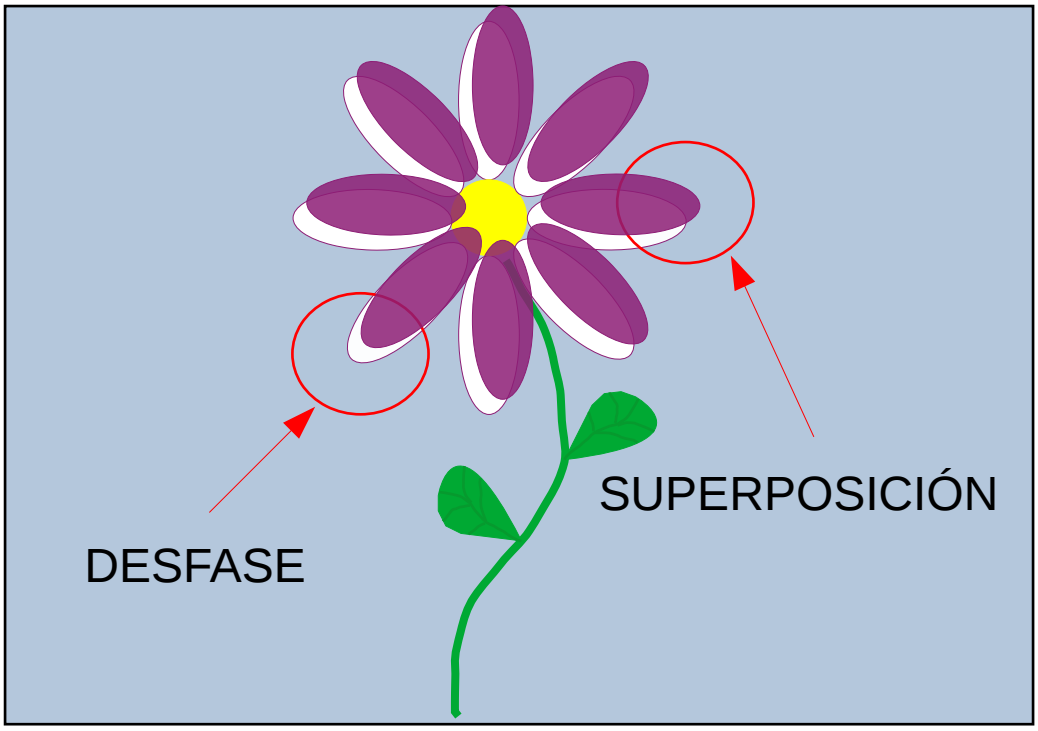


Figura 26: Defecto por desviación de alineación diagonal.

Puede observarse que para este caso el patrón de los pétalos no se ha movido exactamente a la derecha, ni a la izquierda como lo ocurrido en la figura 24 de la sección 2.2 correspondiente a los defectos de

alineación lateral, sino esta vez se ha movido en forma diagonal. En este contexto se dice que el defecto mostrado en la figura 26 ha sufrido un movimiento hacia la diagonal superior.

El sistema *Navigator Automata* también tiene la capacidad de corregir estos defectos, mediante el ajuste de forma parecida al ajuste lateral, pero para este tipo de problema el cilindro de estampación se mueve solo del extremo derecho en movimiento diagonal, hacia la diagonal superior o inferior según sea al caso, quedando el extremo izquierdo como punto fijo o pivote. En ambos casos, ajuste lateral y diagonal, son realizados con precisión por el sistema *Navigator Automata* mediante el uso de un pequeño teclado el cual es operado por una persona, mientras que otro operador observa que la alineación encaje perfectamente con el diseño.

En la figura 27 se muestra una representación de los cilindros de estampación en sus posiciones de diagonal máximo y diagonal mínimo.

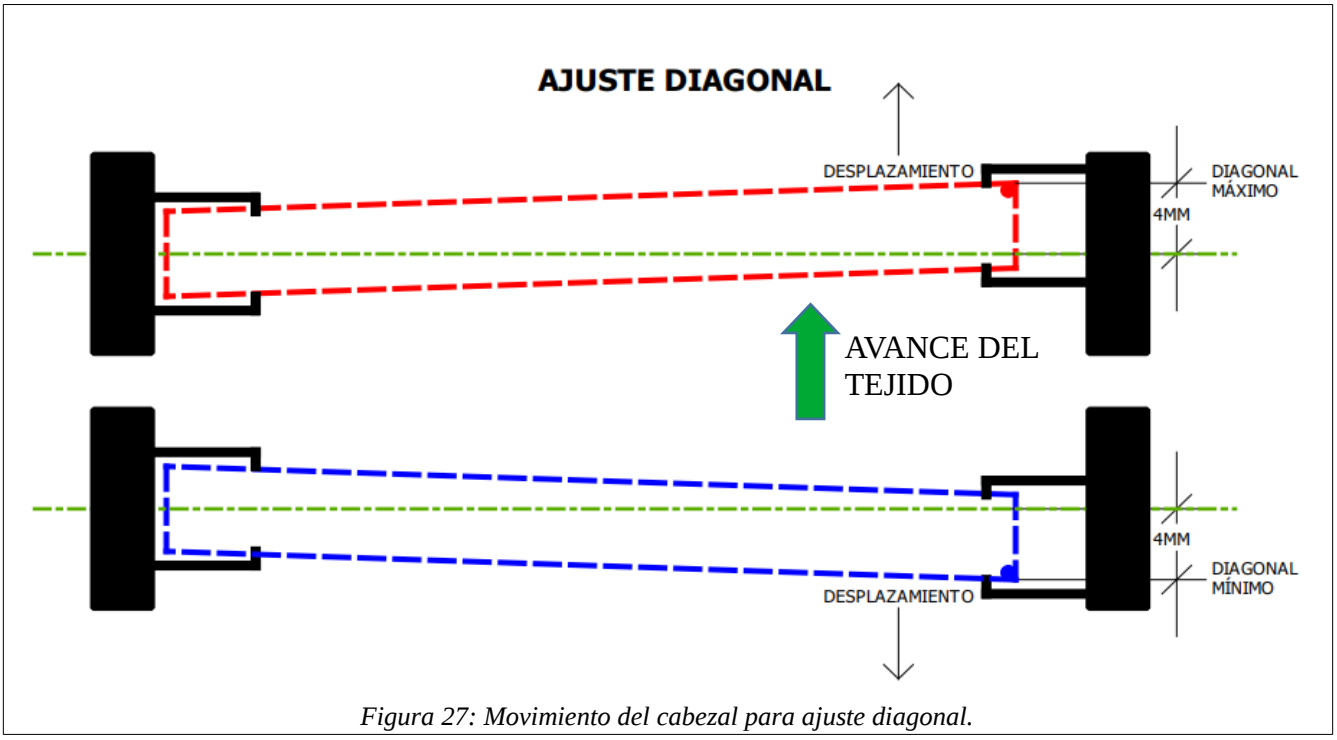


Figura 27: Movimiento del cabezal para ajuste diagonal.

2.4 SUMINISTRO DE TINTAS

El suministro de las tintas para el proceso de estampación puede sonar un tema trivial, o una función muy lógica y que por ende debería de ser muy fácil manejar o restarle algo de importancia, puede ser que este sea el pensamiento que se maneje sobre el tema de las tintas, pero realmente resulta todo lo contrario, ya que el proceso de estampado es básicamente aplicar tinta sobre el tejido, pero no basta con solo aplicarla, este proceso debe de ser controlado, ordenado y limpio, ya que las expectativas sobre el

producto final siempre es obtener la más alta calidad a fin de que el precio del producto se mantenga a la altura de la calidad del producto, lo cual asegura también clientes satisfechos y la continuidad dentro del mercado textil nacional e internacional.

Figura 28: Cocina de tintas.

La preparación de las tintas inicia en un lugar conocido como “cocina de tintas”(figura 28), es aquí donde se hacen los ensayos respectivos para la obtención de los colores que el cliente desea ver en su diseño. Las mezclas de colores y los demás procedimientos que competen a esta área son realizados en ese lugar.

Una vez que se han obtenido los tonos requeridos en los colores de las tintas y son aprobados tanto por el departamento de estampado como por los clientes, se procede a preparar las cantidades necesarias para la cantidad de tela que se pretende estampar. La tinta se almacena en barriles de material plástico a fin de evitar cualquier tipo de contaminación y se trasladan hacia la maquina estampadora, se dispone un barril de cada color por cada cabezal, y se introduce una manguera de succión dentro del barril, esta manguera se encuentra conectada a una bomba que se encarga de verter la tinta en el interior del cilindro perforado, la tinta debe de tener las propiedades necesarias de viscosidad para que su aplicación sea lo más óptima posible, ya que debe de distribuirse de manera uniforme a lo largo del cilindro perforado sin acumularse en un solo lugar, ya que esto produciría que en áreas del tejido el color resulte más intenso que en otras, y por ende más húmedo y por la consistencia de la tinta, más pegajoso y difícil de secar en el paso posterior de secado.

Figura 29: Barriles con tinta o también llamada pasta de color.

La aplicación de la tinta debe de ser un proceso controlado de manera automática, debe de aplicarse con medida y ser monitoreado continuamente de manera que no haya exceso de tinta, ni escasez de esta, ya que de darse el caso que la aplicación de la tinta se haga de acuerdo a la percepción del ojo humano, se corre el riesgo muy alto de atentar contra la calidad del producto final, aunque también existe la opción de accionar las bombas de tinta de forma manual, cuya lógica se explicará en el capítulo tres de este trabajo de graduación.

Una vez que la tinta se encuentra en el interior del cilindro perforado, esta es desplegada uniformemente con un dispositivo llamado raqueta, cabe mencionar que este despliegue uniforme no depende solamente de la raqueta, sino, como se mencionó anteriormente, de la consistencia de la tinta, es decir de su viscosidad. Se introduce una barra metálica dentro del cilindro perforado, junto a la tinta y la raqueta, y se activa un electroimán en forma de barra, cuya longitud es igual a la barra metálica introducida, atrayendo fuertemente la barra metálica de manera que la tinta es presionada y obligada a salir por las perforaciones del cilindro dejando su marca en el tejido, estampando el patrón con su respectivo color.

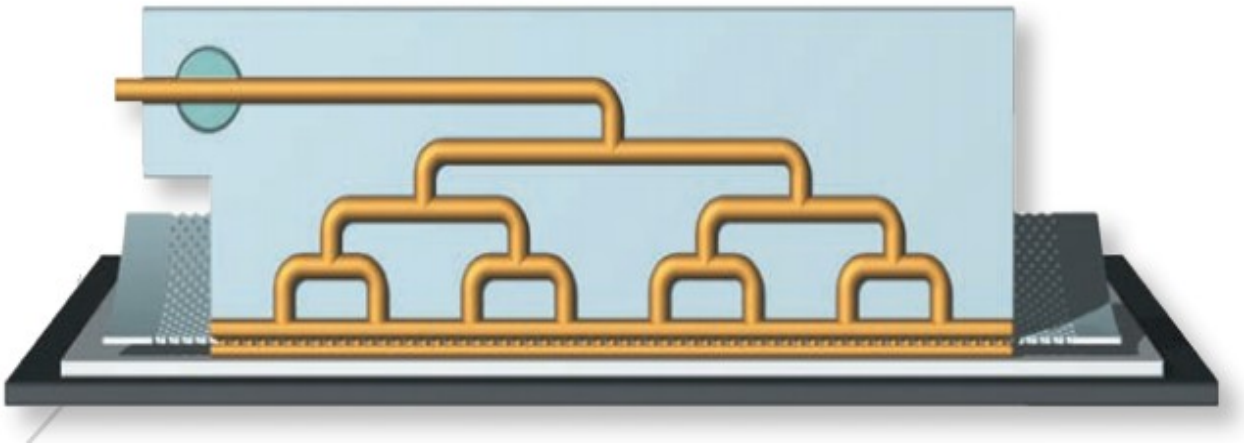


Figura 30: Representación de tecnología de distribución de tintas BVS micro channel system de Zimmer Austria [19].

2.5 ANÁLISIS DEL AJUSTE LATERAL

2.5.1 HARDWARE PARA LA OPERACIÓN DEL AJUSTE LATERAL

A la hora de realizar el estampado de un nuevo diseño, a los operadores se les entrega una hoja en donde se encuentran todos los parámetros que deben ajustarse en la maquinaria para la producción, a la cual se le denomina receta.

Dentro de estos parámetros se encuentran, los colores de las tintas, tipo y calidad del tejido, velocidad de la máquina, temperatura del horno de secado, niveles de líquidos en tanques, tiempo de permanencia de la tela en el horno, entre otro como los parámetros de interés para este trabajo de graduación, tales como, la posición de los cabezales para un estampado exitoso, la cual para este caso es la longitud recorrida de izquierda a derecha o viceversa en un intervalo de cero a diez milímetros, que hasta este momento se ha manejado con el nombre de ajuste lateral.

Es la placa electrónica AUTOMATA 70036700 220L3670 Rev0.1 la encargada de recibir la orden de un ajuste lateral (y diagonal del cual se explicará con detalle en la sección 2.6) y procesar las condiciones en sus entradas para luego ejecutar la acción deseada.

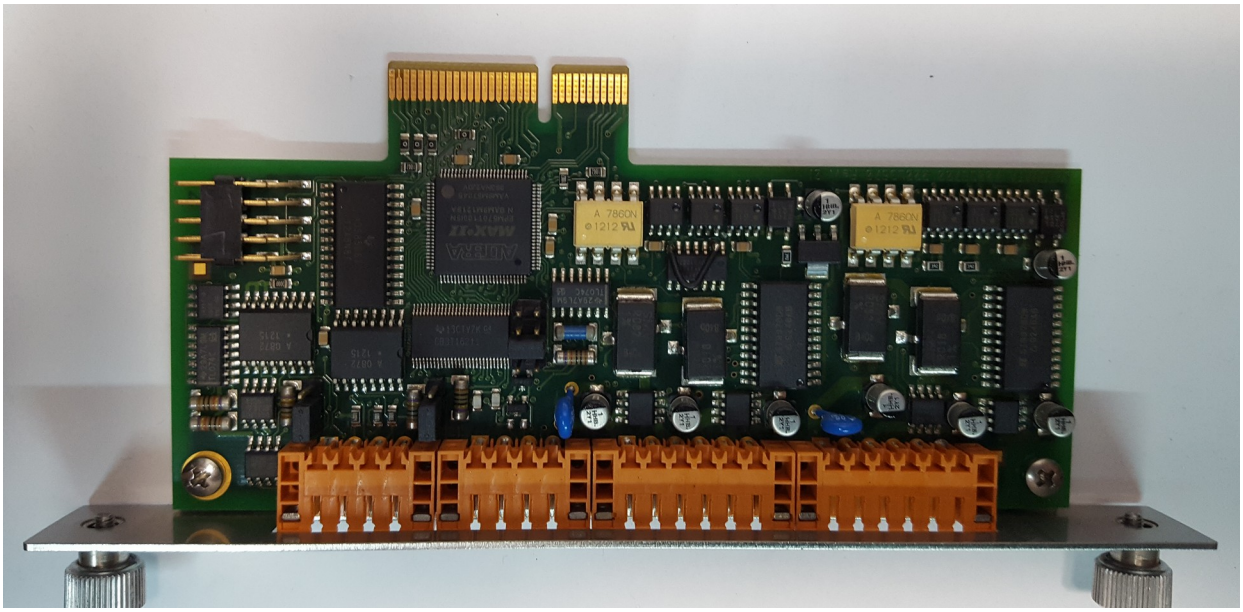


Figura 31: Placa electrónica AUTOMATA 70036700 220L3670 Rev0.1

El operador debe de escribir estas posiciones en la llamada receta para cada lote de tela que se estampa con su respectivo diseño, ya que de no guardarse la receta correspondiente a cada diseño, habría que hacer las pruebas de ajuste a prueba y error cada vez que se necesite hacer diferentes estampados, y dependiendo del número de colores que el diseño contenga, esto puede llegar a ser tan fácil como ajustar estos parámetros de posición para uno o dos colores, o tan complicado como ajustarlos para un diseño de doce colores. El problema de no guardar una receta radica en que para cada ajuste debe de utilizarse tejido virgen el cual se descarta para la venta debido a que solo se utiliza para hacer las pruebas de alineación hasta obtener un diseño limpio y sin defectos. Este procedimiento de ajuste y alineación iniciales puede consumir largos tramos de tejido que bien pudieran ser comercializados, es por este motivo que el ajuste debe de ser un proceso rápido, preciso y eficaz.

En la figura 32 se muestra un segmento del diagrama eléctrico correspondiente al ajuste lateral, se puede observar que un motor de corriente continua -1M84 es el encargado de mover el cabezal hacia la izquierda o hacia la derecha dependiendo del ajuste que se necesite hasta lograr alinear el diseño de manera satisfactoria. Dicho motor de corriente continua trabaja cuando en sus terminales se aplica una tensión de 18VDC provenientes de la placa AUTOMATA 70036700 220L3670 REV.01. En la cual ya se han tomado todas las decisiones tomando en cuenta las variables de entrada y las condiciones necesarias de operación. La dirección del movimiento del cabezal está definida por en sentido de rotación del eje del motor.

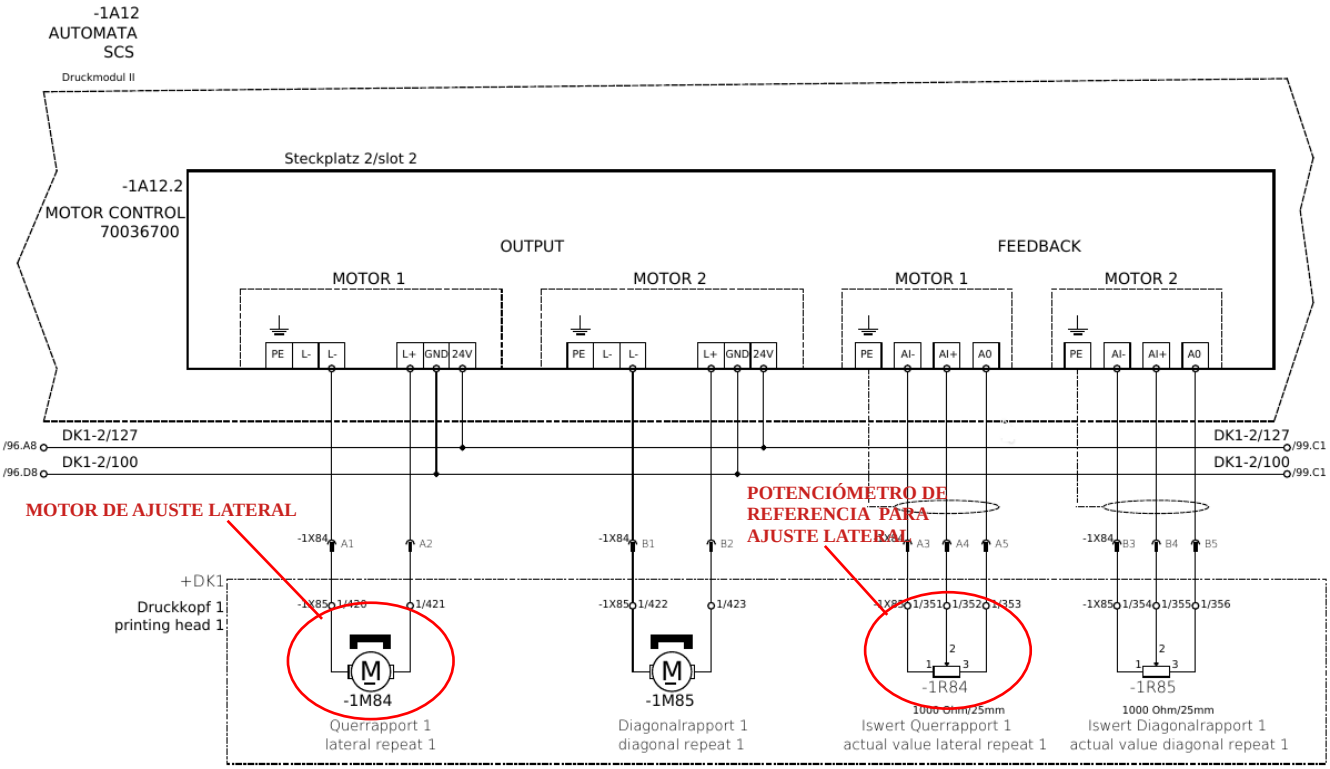


Figura 32: Motor y potenciómetro para ajuste lateral.

La placa electrónica designada como -1A12.2 de acuerdo a la nomenclatura mostrada en el diagrama eléctrico, corresponde al cabezal de estampación numero uno. El cual a su vez pertenece a la estación de control -1A12. Para cada cabezal existe una placa electrónica similar. Existe una estación de control para cada dos cabezales en el que se pueden encontrar cuatro slots para empotran las placas, pero el fabricante ubica solamente tres placas por estación, para el caso mostrado en la figura 32, se puede encontrar que el motor -1M84 es el encargado del ajuste lateral, este ultimo en conjunto con la señal de referencia provista por el potenciómetro llamado -1R84, y el pulsador para entrada de la dirección, conforman el hardware para el ajuste lateral.

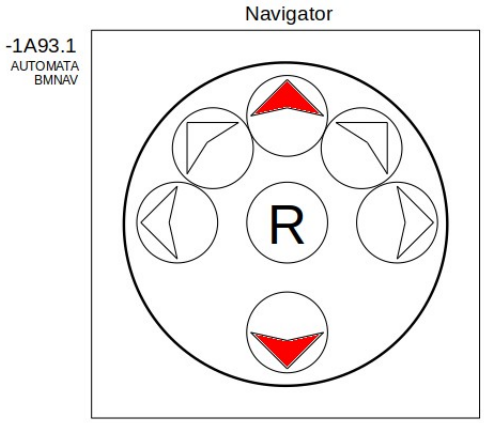


Figura 33: Teclas de control de ajuste

Hasta este punto el análisis parece un procedimiento de cambio del sentido de la rotación del eje de un motor común y corriente, pero se complica con el hecho de que este ajuste debe hacerse de manera rápida y precisa, ya que la maquinaria no solamente se trabaja con un solo tipo de diseño, de hecho en un solo día pueden llegar a cambiar de diseño hasta cinco o más veces, ya que la ROTA-TG/116 puede llegar a trabajar a velocidades de hasta 120 m/s[1], esta velocidad le da la capacidad de estampar miles de kilogramos de tejido diariamente sin contar que es una maquinaria que funciona las veinticuatro

horas del día, y los siete días de la semana, por lo que la rapidez es una característica que no se puede pasar por alto en este proceso.

Para poder explicar un poco acerca del modo de operación del ajuste lateral, se puede mencionar que es un acto simple, tanto como pulsar un botón que apunta en la dirección deseada del ajuste, mientras otra persona observa que la alineación sea efectiva. Cabe mencionar que con la nueva placa desarrollada, la operación será siempre de una manera sencilla, pero esta vez se hará a través de un selector de tres posiciones, ya que la forma original de operar estas funciones se hacia anteriormente con un pequeño teclado como aparece en el diagrama eléctrico de la maquinaria como el que se muestra en la figura 33, en donde las teclas con flechas de color rojo son las encargadas del ajuste lateral.

Se sugirió que era posible tomar el teclado original y desarmarlo para intervenir las líneas necesarias para que el procedimiento no tuviese que cambiar en nada para el operador de la maquinaria, pero debido a ordenes del personal técnico de mantenimiento del lugar, las cuales indicaban que no se comprometiera la integridad del teclado, no fue posible llevar a cabo la sugerencia hecha, debido a que este integra tres funciones más las cuales no son objeto de estudio en este trabajo de graduación, y las cuales se encuentran trabajando perfectamente.

La nueva forma de ajuste será posible a través de un selector rotativo de tres posiciones, en las que la posición 0 significa sistema en reposo o no ejecutar ninguna acción, la posición 1 ajuste lateral manual a la izquierda, y la posición 2 ajuste lateral manual a la derecha.

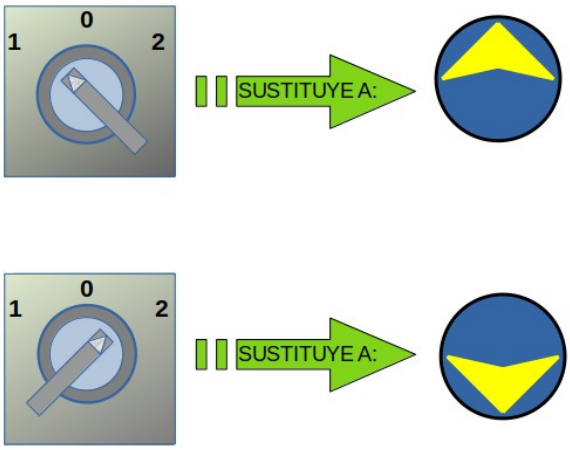


Figura 34: Nuevo modo de operación del sistema de ajuste lateral.

El detalle de las conexiones eléctricas a realizar con las nuevas modificaciones se muestra a continuación en la figura 35

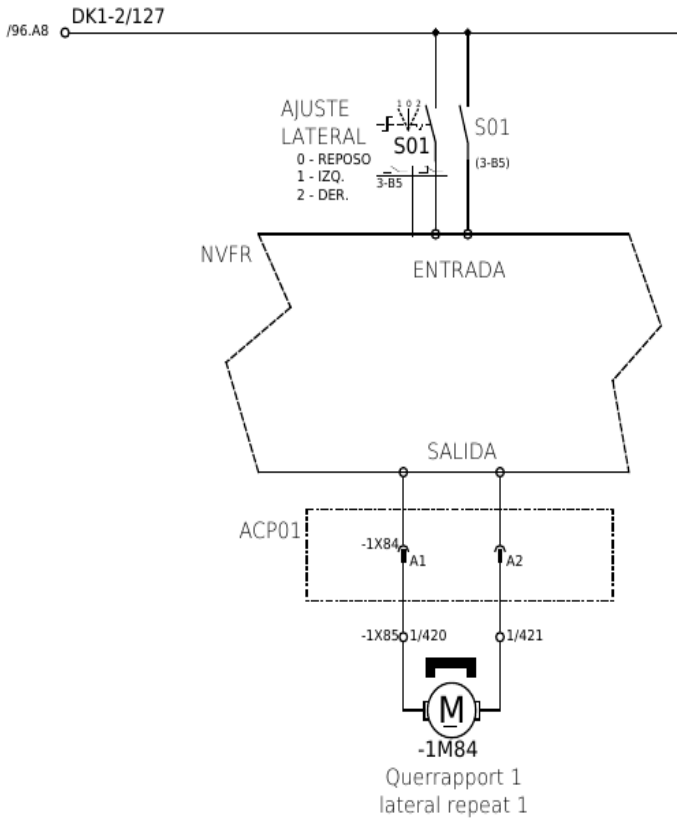


Figura 35: Control del motor de ajuste lateral con nueva propuesta.

Se puede mencionar que en el diagrama de la figura 35, se ha representado una parte de lo que es la placa electrónica desarrollada, la cual tendrá el fin de elegir la dirección de ajuste y luego la salida que es el motor. La linea DK1-2/127 corresponde a la tensión de suministro con una magnitud de +24VDC.

Tal parece que el procedimiento es muy sencillo, esto es lógico debido a que se da la impresión de ser una simple acción que la persona realiza como presionar un pulsador o girar un selector. Sin embargo, dicha acción simple desencadena todo el flujo de decisiones y procesos de un algoritmo que se ejecuta de manera lógica, sistemática y ordenada para poder llegar a los resultados esperados.

2.5.2 LÓGICA DE FUNCIONAMIENTO DEL AJUSTE LATERAL

Con el análisis anteriormente realizado en la sección 2.5.1 de este trabajo de graduación, es suficiente para poder concluir que el ajuste lateral hace trabajar al motor -1M84 tanto en dirección horaria como anti horaria dependiendo si se desea mover a derecha o a izquierda el cilindro de estampación, pero surge el siguiente problema, no existe un dispositivo tal como un final de carrera como en la mayoría de sistemas convencionales donde estos se utilizan cortando el suministro de energía al motor rotando

en un sentido y a la vez activando el sentido de giro contrario o desactivando el sistema por completo a manera de protección de los sistemas mecánicos y eléctricos. En este sistema no existen esos dispositivos que limiten el movimiento en ambas direcciones cuando el cilindro ha alcanzado la posición más a la izquierda o la posición más a la derecha, puntos que por convención se han nombrado lateral mínimo y lateral máximo respectivamente, pero por alguna razón, al alcanzar el valor lateral mínimo(-10.10mm) o el valor lateral máximo(10.00mm), se detiene súbitamente el motor protegiendo así la estructura del cabezal, el cilindro de estampación, el motor, los componentes de la placa AUTOMATA 70036700 220L3670 REV.01, así como las partes mecánicas que conforman el sistema.

Por lo tanto se ha supuesto que el algoritmo que ejecuta el microcontrolador del sistema *Navigator Automata*, se está guiando por los valores leídos del potenciómetro de referencia, de manera que al leer un valor que se encuentra debajo de 1.608V, es decir, totalmente a la izquierda, interpreta que es un valor más allá del valor lateral mínimo permitido y por lo tanto ejecuta una especie de final de carrera por software basado en el valor leído el cual debe de cortar el paso de energía al motor para proteger el sistema, luego se bloquea la opción de continuar moviendo hacia la izquierda y solo se permite mover manualmente hacia la derecha. Algo similar sucede cuando el cilindro alcanza la posición lateral máxima, es decir, totalmente a la derecha. En este punto el valor de tensión en el potenciómetro estaría mas allá de 9.170V y el motor deja de trabajar, se bloquea el ajuste lateral hacia la derecha y solo es permitida la manipulación manual hacia la izquierda, de otra forma el motor deja de trabajar.

Se puede observar gráficamente el algoritmo basado en el supuesto del comportamiento observado en la figura 36 a continuación.

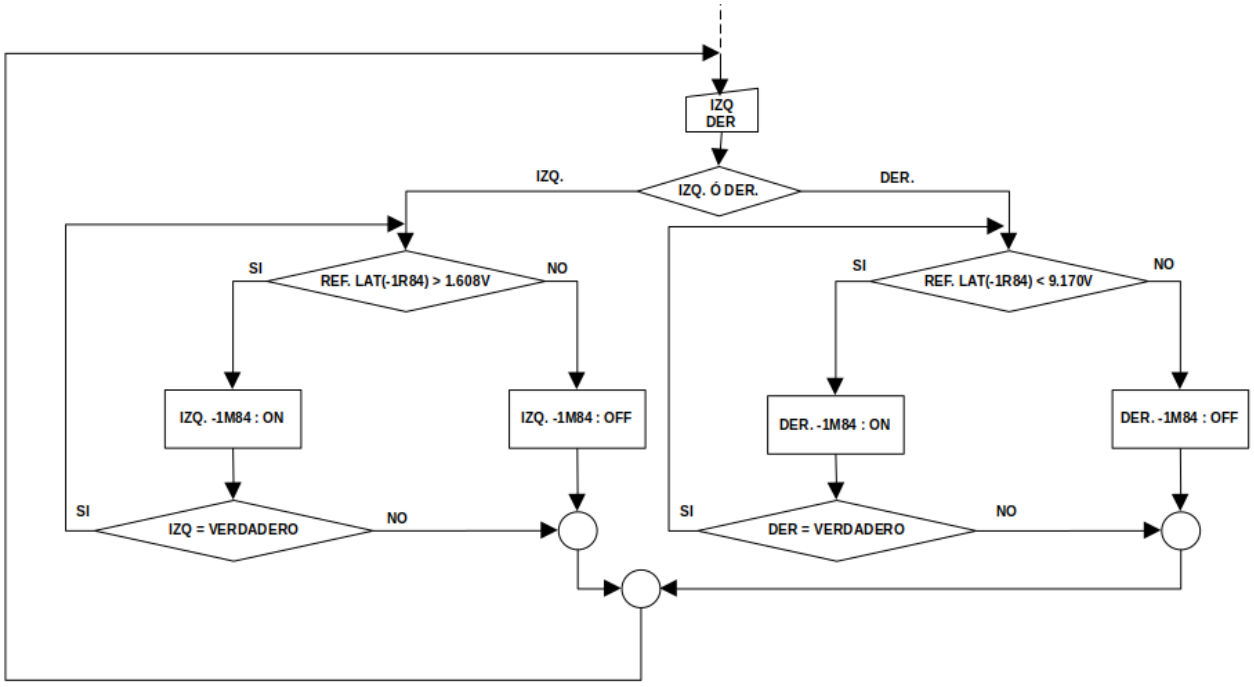


Figura 36: Diagrama de comportamiento observado en ajuste lateral.

Luego de realizar numerosas observaciones, se logró entender que en efecto, el comportamiento del sistema de ajuste lateral, se lleva a cabo siguiendo la lógica según el flujo del diagrama mostrado en la figura 36. Todo parece que debería de funcionar a la perfección, de no ser porque se han obviado algunos efectos físicos adversos que se describen en la sección ocho de este capítulo, los cuales son los causantes de que el sistema falle constantemente.

Uno de los objetivos mas importantes descritos en esta trabajo de graduación, es el de encontrar las causas de las fallas recurrentes en este sistema, por lo que del algoritmo que se muestra gráficamente en la figura 36, se utilizará la parte mas básica en la fase de programación, pero con una mejora diseñada para evitar que la mala operación, o los efectos físicos que causan los problemas actuales vuelvan a estropear alguna parte del sistema, logrando dar cumplimiento al objetivo señalado. En la figura 37 se puede apreciar la mejora que se aplicará en la lógica de funcionamiento.

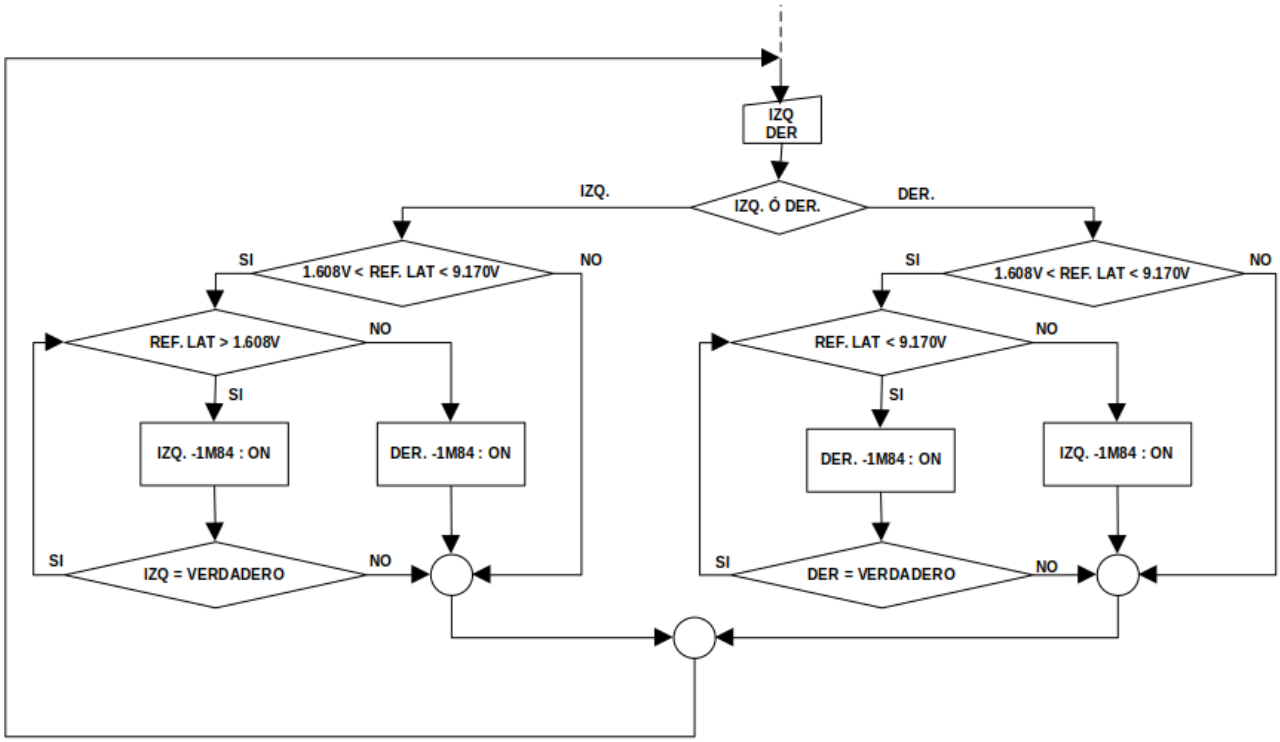


Figura 37: Diagrama con la lógica mejorada para ajuste lateral.

Aunque al hacer la comparación entre los diagramas de flujo del proceso de ajuste lateral entre la figura 36 y la figura 37, pareciera no mostrar una diferencia relevante, se han introducido unos cambios realmente simples pero con excelentes resultados en la operación de sistema, incluso si el operador llegase a operar de manera errónea, ahora el sistema permite hacer un autoajuste lateral que evita que por mando del operador el sistema falle por errores en la señal del potenciómetro de referencia o por una señal de ajuste forzado.

La lógica de funcionamiento con estas mejoras se describen de la siguiente manera:

1. El sistema se encuentra a la espera de recibir la orden de un ajuste lateral ya sea a la izquierda o a la derecha.
2. Se introduce la orden de ajuste deseado.
3. Se llega a la bifurcación de acuerdo a la dirección de ajuste que se ha decidido.
4. Se continua el flujo del proceso de acuerdo a la dirección que se desee y se lee el potenciómetro de referencia de ajuste lateral, el cual como ya se ha mencionado, regresa un valor de tensión que se traduce en la posición del cilindro de estampación, si el cilindro se encuentra en una posición menor al valor mínimo o mayor al valor máximo de distancia longitudinal permitida, que por lo general se da en caso que el potenciómetro haya llegado al final de su vida útil, o realmente el cilindro haya sobrepasado los valores extremos, entonces simplemente el sistema no responde a la petición manual o automática de ajuste, y permanece así hasta encontrarse en los parámetros correctos. Pero si es leído un valor que se encuentre dentro del rango fijado de valores aceptados para ejecutar un ajuste, entonces se procede a ejecutar el siguiente paso.
5. Se procede nuevamente a leer el potenciómetro como una medición mucho mas especifica que la primera lectura, esta ultima con el objetivo de seguridad de no sobrepasar la posición limite izquierdo o derecho según sea el caso. Si el valor de lectura cumple la condición, entonces se procede exitosamente a energizar el motor en la dirección que corresponda siempre y cuando se encuentre activa la entrada correspondiente a la petición de ajuste inicial, es decir, ajuste a la izquierda, o ajuste a la derecha. En caso de que la entrada permanezca activa, el desplazamiento del cilindro continuará hasta encontrar una condición de no operación como medida de seguridad. Pero si aun estando la entrada activa se encuentra la condición que excede los limites ya sea izquierdo o derecho, en lugar de simplemente desactivar el movimiento en el sentido actual, se cambia la polaridad del motor y por ende su dirección de rotación, lo que provoca que el cilindro vaya hacia el lado opuesto hasta quedar en el limite seguro en donde no exija al motor mover una carga que ya encontró su limite mecánico.
6. Al desactivar la entrada para ajuste, el sistema simplemente vuelve a quedar en reposo hasta que se le pida nuevamente otro ajuste.

2.5.3 DETERMINACIÓN DE LA POSICIÓN LATERAL DEL CABEZAL

Durante la toma de lecturas del sistema *Navigator Automata*, en el contexto del ajuste lateral se obtuvieron mediciones de una señal de cero a diez voltios de corriente directa, esta señal es utilizada por la placa electrónica AUTOMATA 70036700 220L3670 REV.01, para determinar la posición del cabezal en todo momento. Cabe destacar que, para la determinación de la posición del cabezal, el microprocesador que se encuentra en dicha placa se vale de la señal regulada por un potenciómetro, dicha señal es luego interpretada y los valores de tensión son traducidos a milímetros.

Como se ha determinado por medio de las mediciones, el rango de desplazamiento del cabezal el cual va de -10.00mm a +10.00mm, está comprendido entre los 1.608V y los 9.170V, en donde 1.608V corresponde a la posición mas a la izquierda, es decir -10.00mm, que significa todo el cilindro se encuentra desplazado a la izquierda, y 9.170V corresponde a +10.00mm o todo el cilindro desplazado a la derecha.

Para este caso se tomaron las siguientes mediciones, ajustando el valor lateral que había recorrido el cilindro a valores fácilmente conocidos y luego midiendo la tensión, los resultados se presentan en la tabla 1.

LONGITUD(mm)	**VOLTAJE(V)**	**LONGITUD(mm)**	**VOLTAJE(V)**
-10.10	1.608	1.20	5.843
-9.00	2.014	2.10	6.169
-8.00	2.383	3.00	6.486
-7.00	2.761	4.00	6.890
-6.00	3.136	5.10	7.290
-5.10	3.471	6.00	7.620
-4.00	3.889	7.20	8.090
-3.00	4.267	8.10	8.410
-2.10	4.587	9.00	8.750
-1.0	4.985	10.10	9.170
0.00	5.360	-	-

Tabla 1: Valores de longitud vrs voltaje de ajuste lateral.

De acuerdo al comportamiento observado fue sencillo establecer que se trataba de una relación lineal, para comprobarlo se puede apreciar el grafico de la figura 38. A partir de esta grafica y de los datos obtenidos, se puede obtener la ecuación que describe la longitud recorrida por el cilindro de estampación y esta puede ser utilizada en el fase de programación para mostrar la posición del cilindro en todo momento.

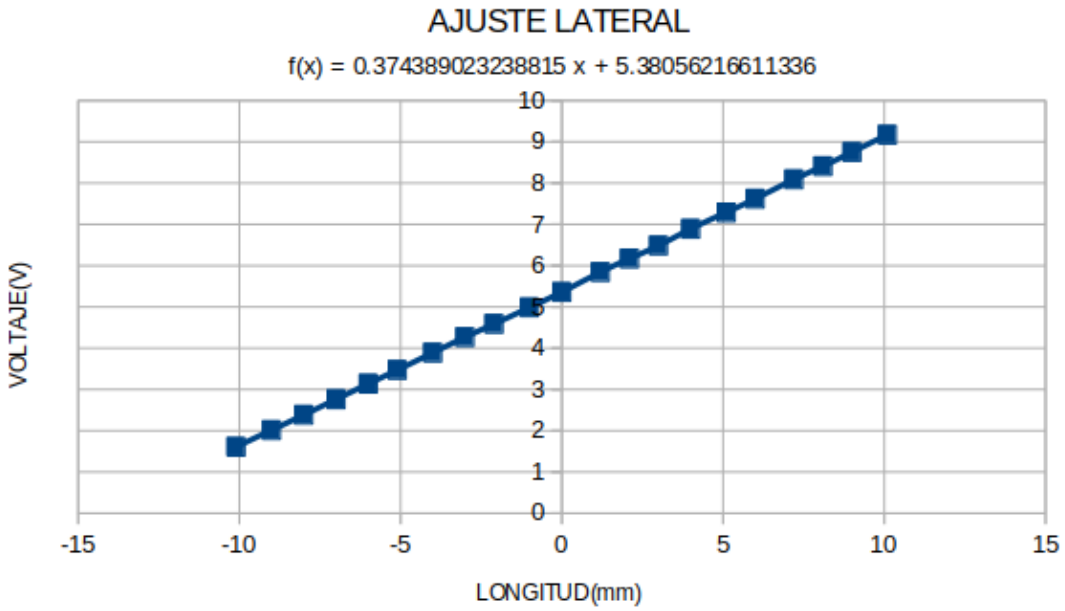


Figura 38: Gráfico de Longitud vrs. Voltaje en ajuste lateral.

2.6 ANÁLISIS DEL AJUSTE DIAGONAL

2.6.1 HARDWARE PARA LA OPERACIÓN DEL AJUSTE DIAGONAL

Como se hizo mención en la sección 2.5.1 de este trabajo de graduación, el ajuste diagonal, para este contexto, es simplemente una extensión del ajuste lateral, ya que la operación es bastante similar, pero con resultados diferentes, ya que los defectos por alineación lateral no pueden resolverse con el ajuste diagonal, y tampoco puede hacerse lo contrario. Sin embargo ambos ajustes también son controlado por la placa electrónica AUTOMATA 70036700 200L3670 Rev0.1

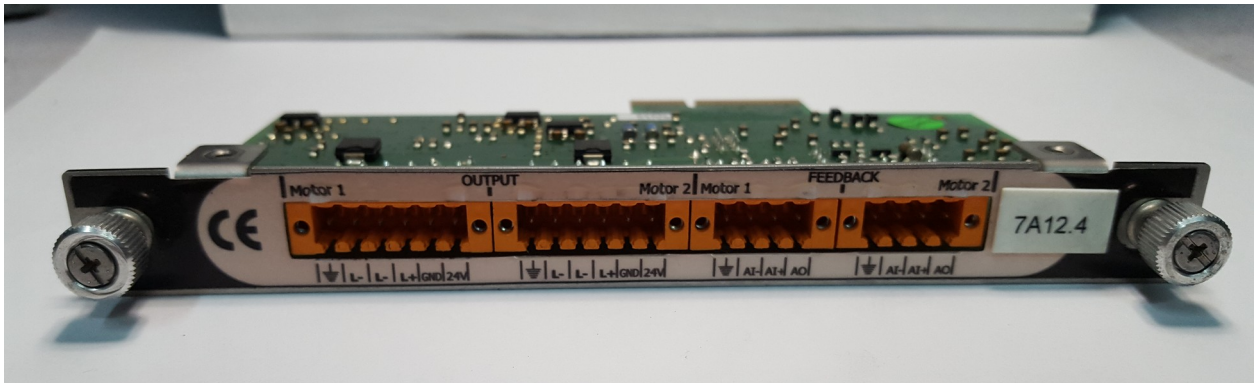


Figura 39: Vista frontal de la placa AUTOMATA 70036700 200L3670 Rev0.1

Como puede observarse en el circuito del diagrama eléctrico que se muestra en la figura 40, el ajuste diagonal es procesado mediante el uso de la placa electrónica AUTOMATA 70036700 220L3670 REV.01, ya que es esta placa la responsable del análisis de las entradas y accionamiento de las salidas para los procesos de ajuste lateral y diagonal al ser solicitados por el operador de la maquina.

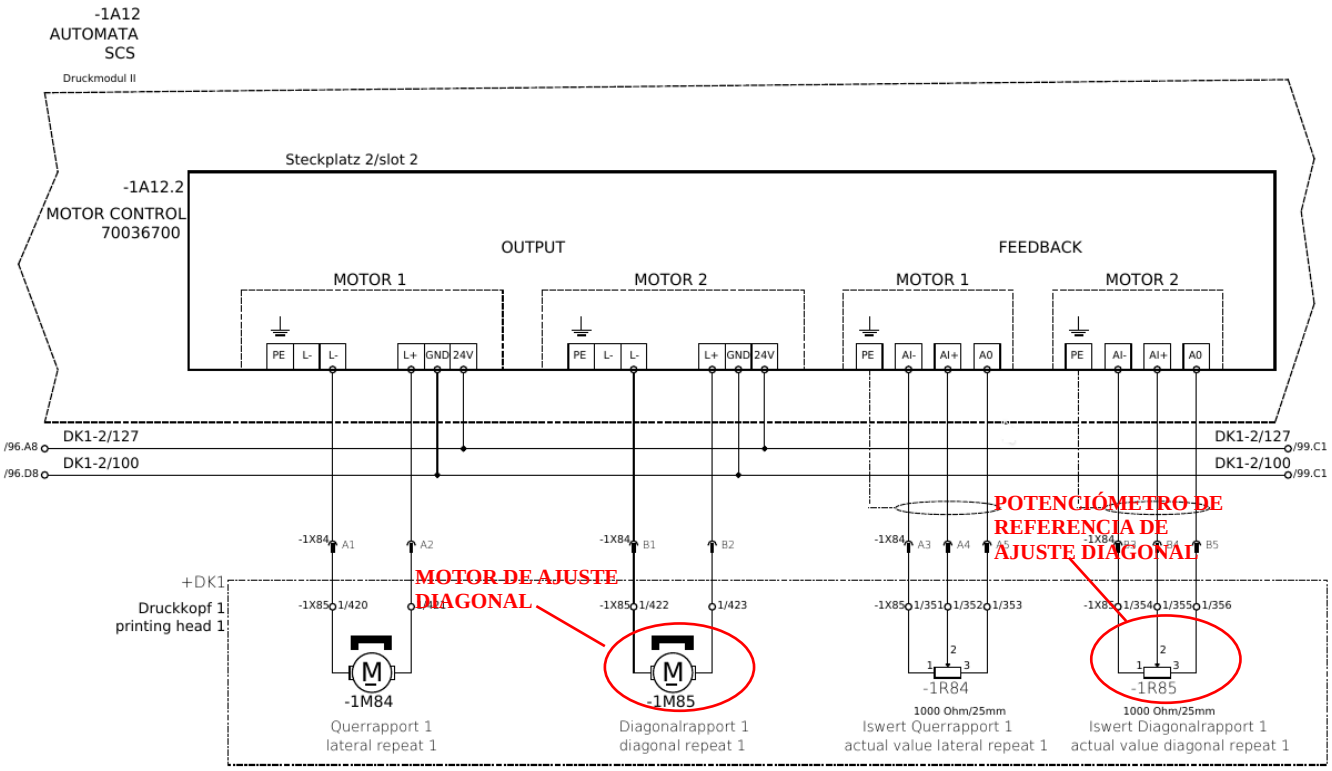


Figura 40: Motor y potenciómetros para ajuste diagonal.

Se puede apreciar que para este caso, es el motor conectado en el segundo puerto de la tarjeta el que se encarga del ajuste diagonal. En este caso se dice que los puntos mínimo y máximo de ajuste son llamados diagonal inferior, y diagonal superior respectivamente. A diferencia del ajuste lateral, el cual se mueve de izquierda a derecha, el ajuste diagonal hace su movimiento de forma perpendicular al movimiento lateral.

El motor -1M85 al ser energizado pone en movimiento los mecanismos necesarios para ajustar el cilindro de estampación hasta que el diseño en el que se está trabajando logre quedar alineado perfectamente. El potenciómetro -1R85 es quien provee la señal de referencia a la placa electrónica de manera que esta sepa siempre la posición del cilindro de estampación a fin de evitar fallas en el cilindro mismo y en los demás componente que conforman el cabezal de estampación.

La manera de introducir las señales necesarias a la placa controladora al requerir un ajuste diagonal, es mediante el teclado incluido en la maquinaria, las teclas resaltadas en color rojo en la figura 41 son las asignadas para esta función.

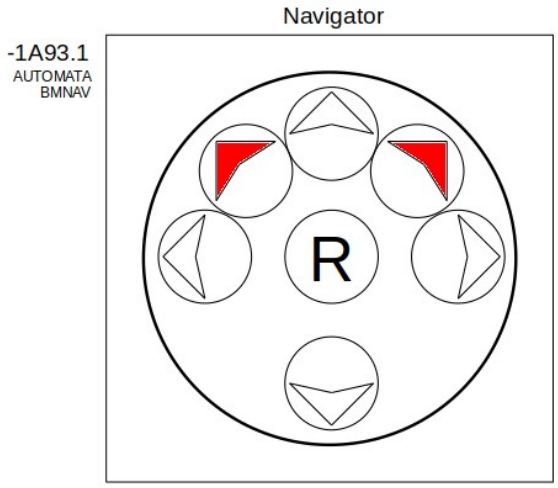


Figura 41: Teclas de ajuste diagonal.

Como se mencionó en la forma de operación del ajuste lateral, la nueva forma de hacer el ajuste será un selector de tres posiciones, para el caso del ajuste diagonal también se utilizara el mismo dispositivo de selección, en donde la posición cero mantiene el sistema en reposo, la posición uno, enviá una señal para que se ejecute el moviendo de ajuste hacia la posición de la diagonal inferior, y la posición dos se encarga de habilitar la entrada que ejecuta el ajuste diagonal hacia la diagonal superior como se muestra en la figura 42.

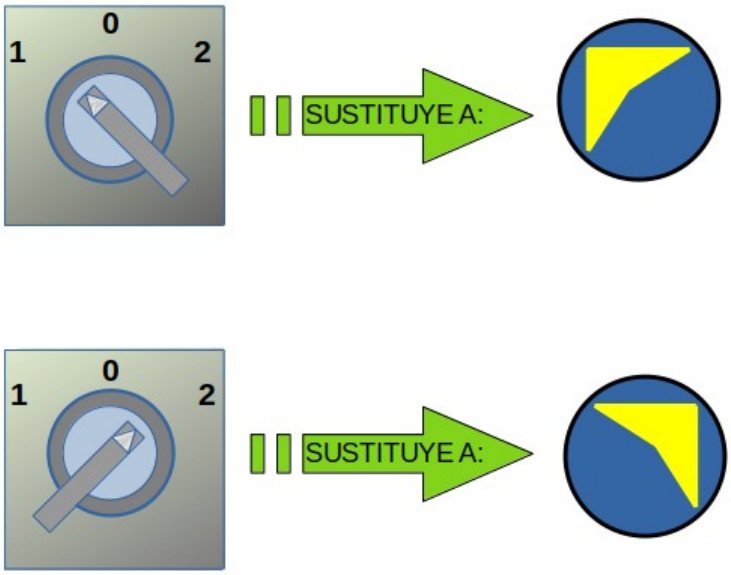


Figura 42: Nueva forma de operación de ajuste diagonal.

En la figura 43 se muestra una representación del diagrama eléctrico con la nueva placa diseñada para el control del ajuste diagonal.

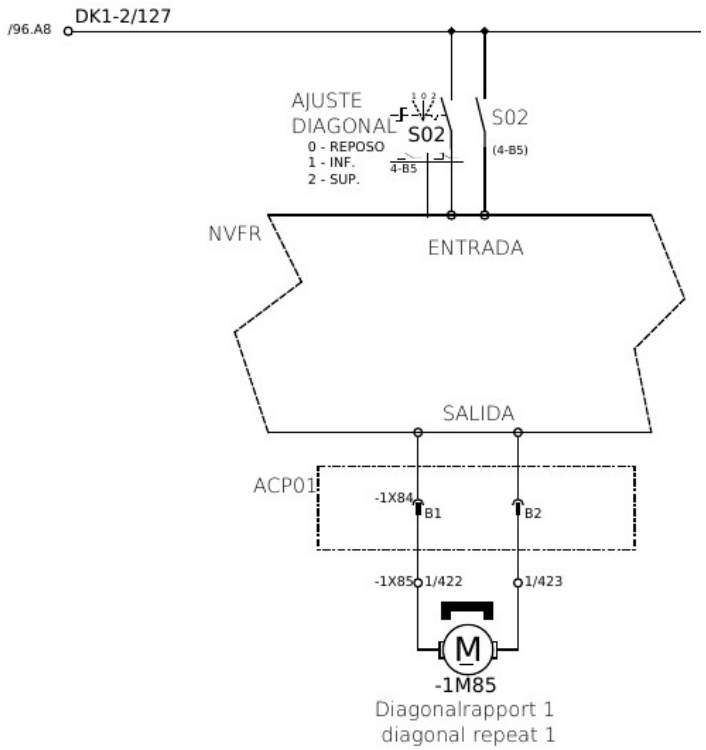


Figura 43: Control de motor de ajuste diagonal con nueva propuesta.

2.6.2 LÓGICA DE FUNCIONAMIENTO DEL AJUSTE DIAGONAL

Basado en las observaciones que se realizaron para el análisis del ajuste lateral, se pudo observar un comportamiento similar en el ajuste diagonal, con las únicas variantes en los niveles de tensión que representan los valores mínimo y máximo diagonal. Por lo demás, la lógica es la misma, tal como se muestra en el diagrama de la figura 44.

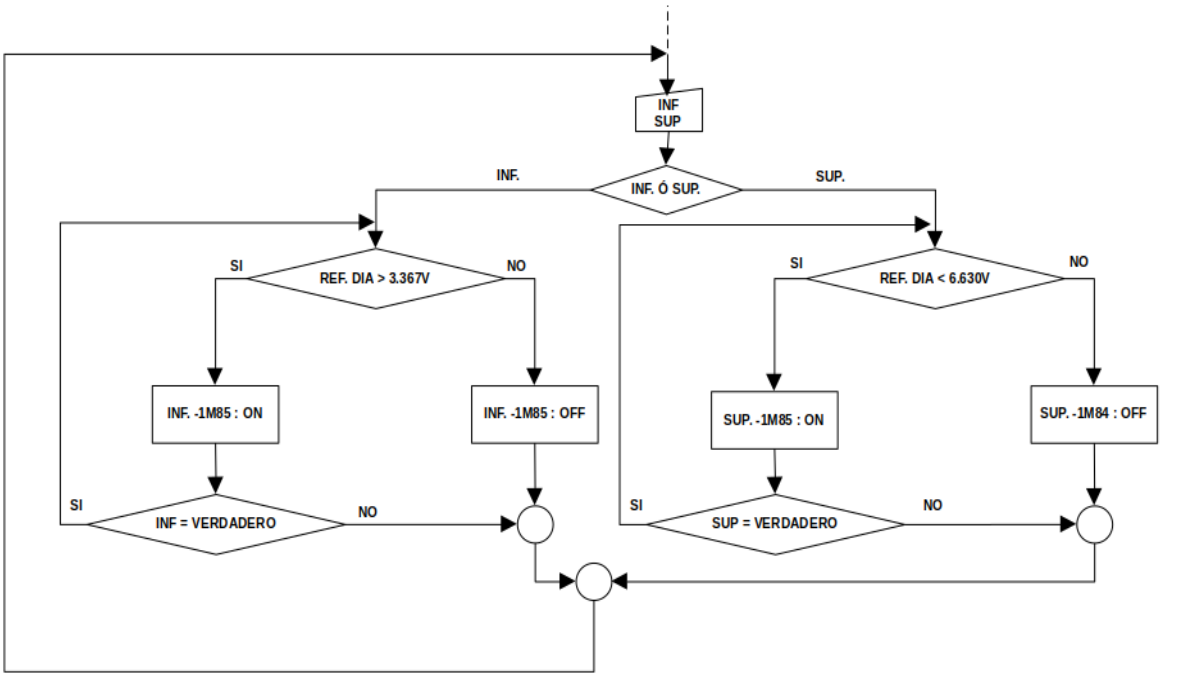


Figura 44: Lógica de control de ajuste diagonal.

También es posible aplicar las mismas mejoras que se aplicaron en el algoritmo de control del ajuste lateral en el ajuste diagonal. Estas se muestran en el diagrama de la figura 45.

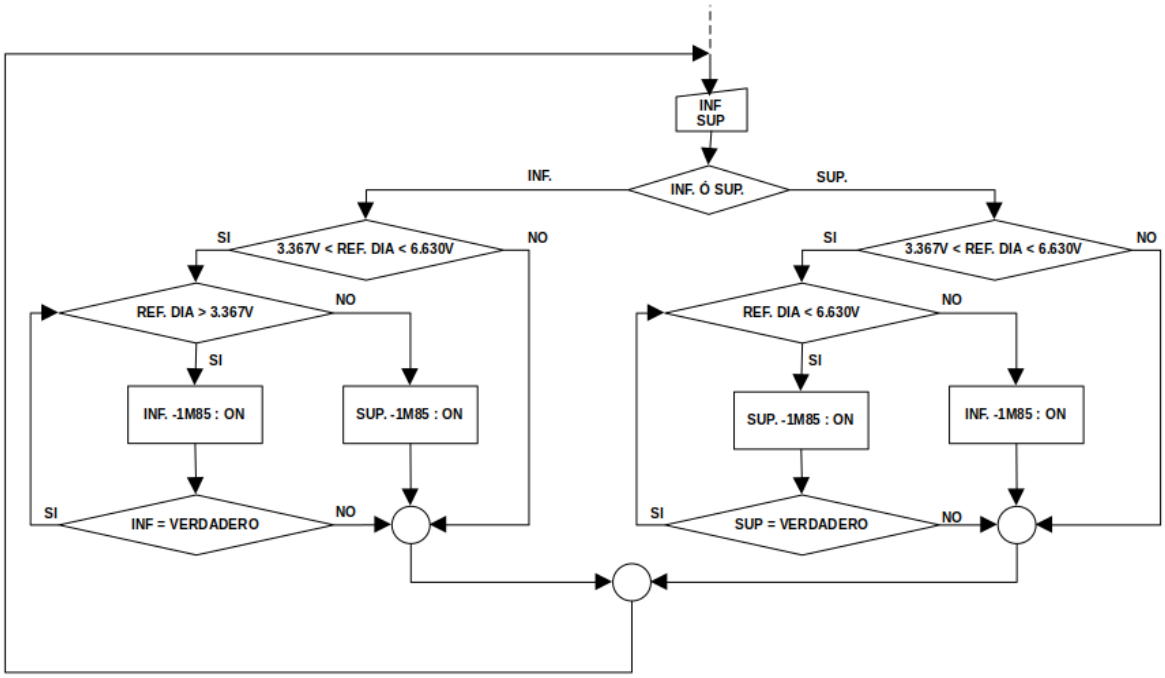


Figura 45: Lógica de control mejorada para ajuste diagonal.

2.6.3 DETERMINACIÓN DE LA POSICIÓN DIAGONAL DEL CABEZAL

Al principio del análisis del problema, se inició con el ajuste lateral, ya que por recomendación del personal técnico de mantenimiento, se sugirió que ese tipo de ajuste es mas sencillo de entender que el ajuste diagonal. Al haber tomado un método de análisis para el movimiento lateral, se tiene la base para poder realizar el mismo método para el ajuste diagonal, por lo tanto se tomaron las lecturas correspondientes del potenciómetro de referencia las cuales se muestran en la tabla 2.

LONGITUD(mm)	VOLTAJE(V)	LONGITUD(mm)	VOLTAJE(V)
-4.00	3.367	1.00	5.403
-3.50	3.592	1.50	5.619
-2.00	4.175	2.00	5.817
-1.60	4.365	2.50	6.009
-1.00	4.593	3.00	6.214
-0.50	4.779	3.50	6.420
0.00	4.993	4.0	6.630
0.50	5.189	-	-

Tabla 2: Valores de longitud vrs voltaje de ajuste diagonal.

Con los datos obtenidos de la tabla 2, se llegó a la conclusión, tal y como se esperaba, de nuevamente encontrar la relación lineal que presentaron los datos recolectado en el análisis del ajuste lateral, pero con la diferencia muy marcada, que mientras en el ajuste lateral el rango va desde los 1.608V hasta los 9.170V, en el ajuste diagonal se obtuvo un rango que va desde los 3.367V hasta los 6.630V, se puede notar que el rango en que el ajuste tiene lugar es mucho mas reducido que para el caso del ajuste lateral. La figura 46 muestra el gráfico con su respectiva ecuación de los datos de la tabla 2.

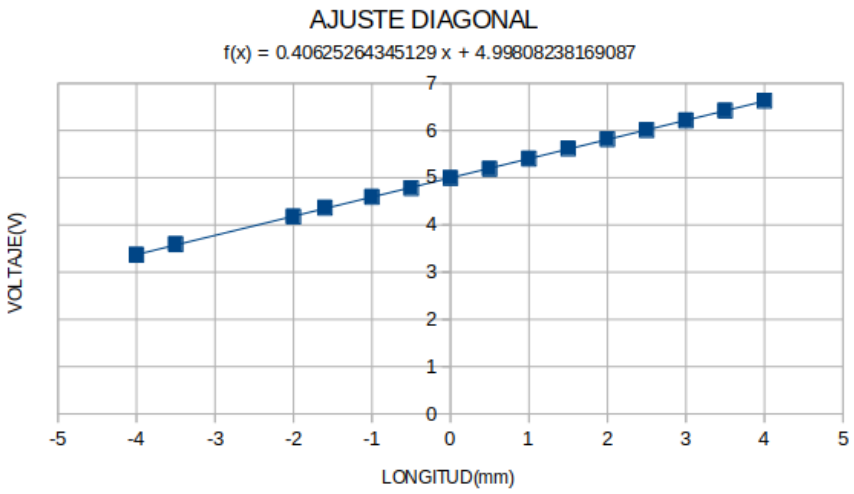


Figura 46: Gráfico Longitud vrs. Voltaje de ajuste diagonal.

2.7 ANÁLISIS DEL CONTROL DE SUMINISTRO DE TINTAS

2.7.1 HARDWARE DEL CONTROL DE SUMINISTRO DE TINTAS.

A diferencia de los ajustes lateral y diagonal de los cuales se encarga la placa electrónica AUTOMATA 70036700 220L3670 REV0.1, el control del suministro de la tinta o pasta de color se encuentra en la placa AUTOMATA 70036800 220L3670 REV 0.1, dicha placa se muestra en la figura 47.

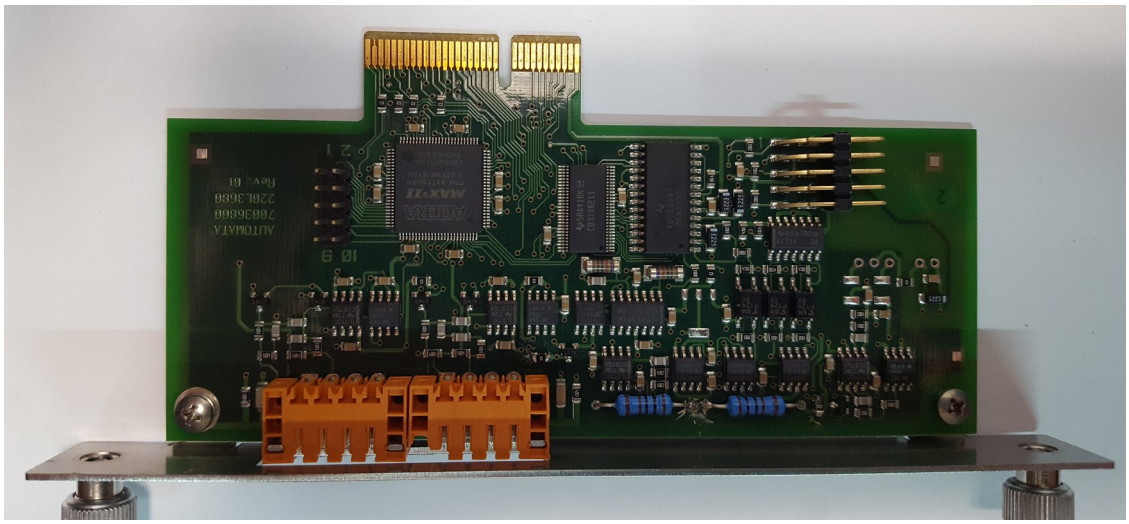


Figura 47: Placa AUTOMATA 70036800 220L3670 REV 0.1

A primera vista se puede notar que la placa posee dos puertos, estos dos puertos sirven para conectar los sensores de nivel de tinta. Toda la lógica de control respectiva se ejecuta en esta placa. La activación de la bomba de tinta no se encuentra directamente en ella, en lugar de eso mediante observación y toma de mediciones, se determino que la placa solo habilita una salida lógica y la envía a la placa base AUTOMATA 70036600 220L3660 REV. 0.3, en donde se activa un pequeño relé interno que energiza y desenergiza la bobina de una electroválvula que realiza el suministro de tinta según sea necesario.

La placa controladora del suministro de tinta en el interior del cilindro de estampación puede controlar a su vez dos cabezales, para el caso estudiado, corresponde a las electroválvulas que según diagrama eléctrico son -1Y91 y -2Y91 como se muestra en la figura 48.

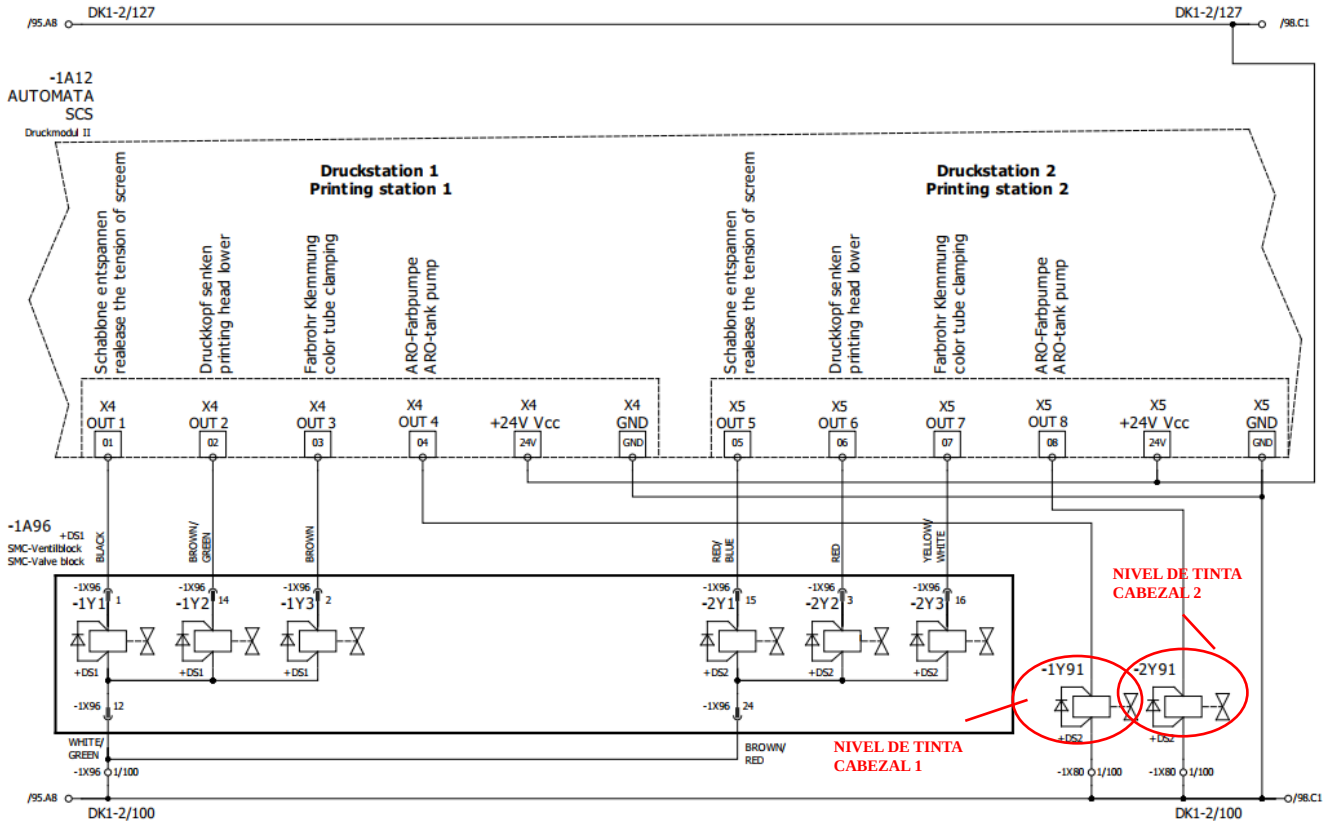


Figura 48: Control de electroválvulas de bombas de color.

2.7.2 LÓGICA DE FUNCIONAMIENTO DEL SUMINISTRO DE TINTA.

El funcionamiento del suministro de tinta o pasta de color se basa en el uso de un electrodo en forma de “L” que se fija en un soporte del cabezal de estampación como el mostrado en la figura 23, el cual esta conectado directamente al puerto correspondiente de la placa de control AUTOMATA 70036800 220L3670 REV 0.1.

La figura 49 muestra la sección del diagrama eléctrico que corresponde a la placa controladora del suministro de tinta, en ella se puede apreciar los dos sensores -1B91 y -2B91, cada uno para un cabezal.

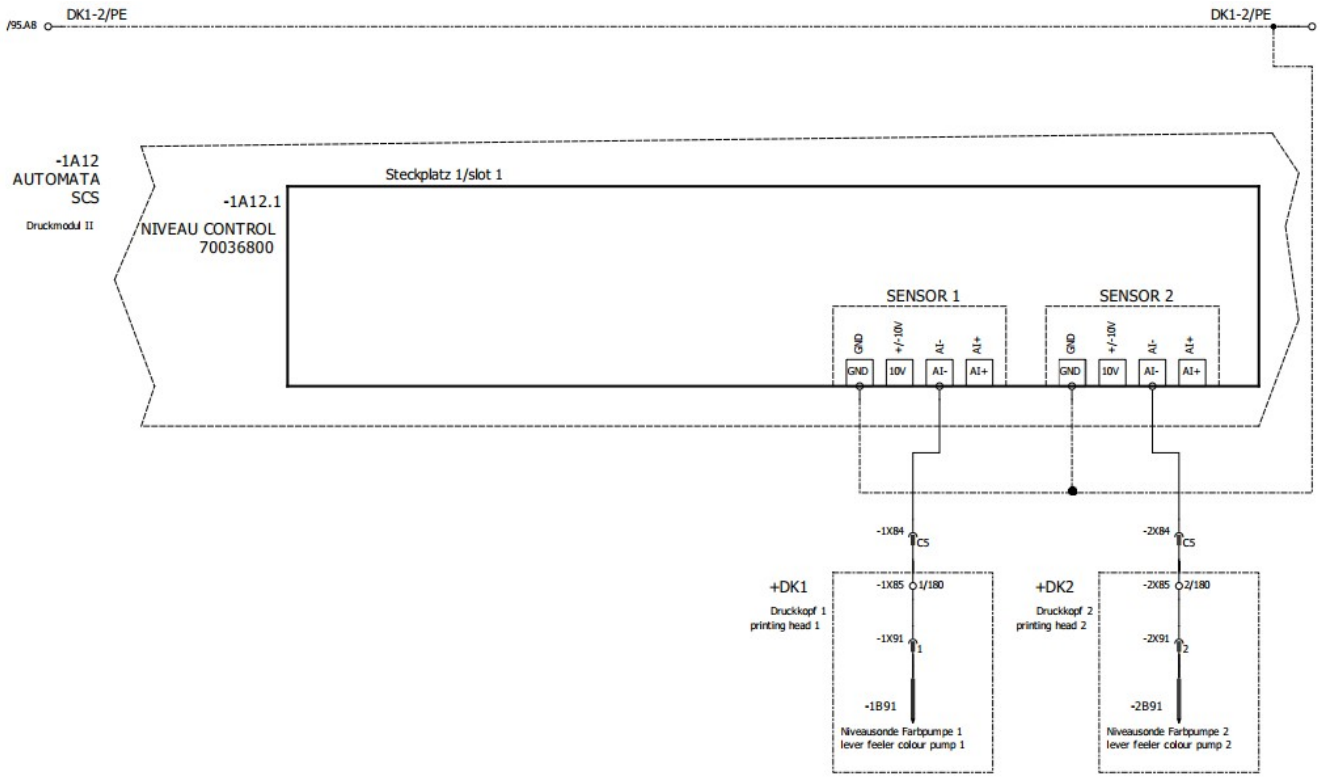


Figura 49: Placa AUTOMATA 70036800 220L3670 REV 0.1.

Existen dos formas de poder operar el control del suministro de tinta, estas dos formas son manual y automático. En el modo manual el operador se encarga de presionar un pulsador el cual directamente energiza la electroválvula correspondiente al cabezal elegido, y de esta forma se logra llenar con la tinta el interior del cilindro de estampación cuando a la percepción del operario, hace falta intensidad de color por alguna obstrucción en la racleta distribuidora del color. Este modo es poco usado y solo se utiliza en casos de emergencia como el mencionado con el problema de la racleta.

El modo automático, es el mas usado y utiliza el electrodo antes mencionado para determinar si el nivel de tinta en el interior del cilindro es el adecuado para continuar el proceso de estampación. El electrodo es introducido en el interior del cilindro de estampación y mientras este se encuentre sumergido en la tinta, la bomba que inyecta la tinta en el interior del cilindro de estampación a través de la racleta, permanece apagada. Según las mediciones realizadas, análisis del proceso y análisis del diagrama eléctrico, se determino que el electrodo sumergido en tinta cierra un circuito interno en la placa AUTOMATA 70036800 220L3670 REV 0.1, haciendo que se active la lógica del algoritmo que controla este proceso. El diagrama de la figura 50 explica de manera gráfica el proceso antes mencionado.

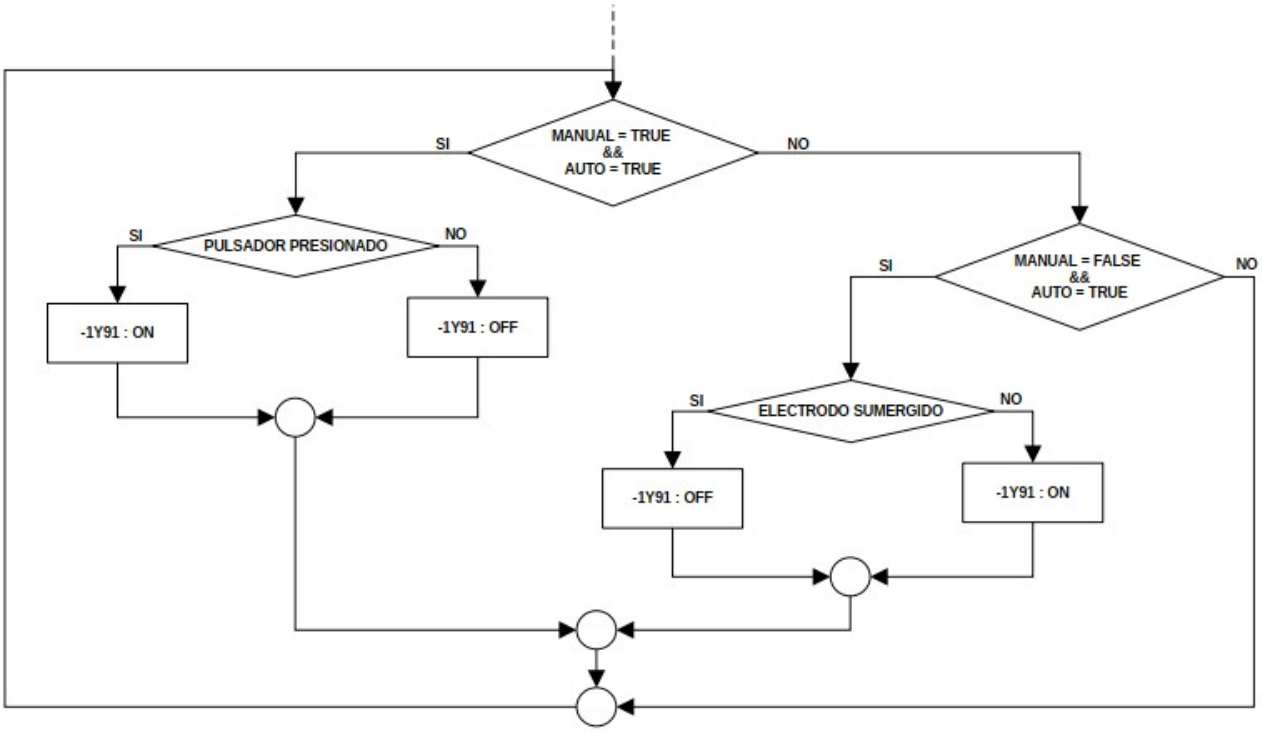


Figura 50: Lógica de control de nivel de tinta.

Un dato interesante en este proceso es que el modo manual es predominante sobre el modo automático, lo que permite que si el sensor llegase a fallar en pleno estampado, el operador tiene la opción de inyectar tinta manualmente para compensar la falla presentada mientras se hace el cambio de sensor.

2.8 CONCLUSIONES DE ANÁLISIS DEL PROBLEMA

Uno de los objetivos planteados al inicio en este trabajo de graduación, es encontrar la causa de las fallas recurrentes en la placa AUTOMATA 70036700 220L3670 REV.01, lo cual con la observación del proceso, análisis de diagramas eléctricos, toma de mediciones, análisis del sistema físicamente, análisis de los datos obtenido y observación de la forma en la cual el operador trabaja la máquina se ha llegado a la conclusión que se ha dado cumplimiento al objetivo al cual se hace referencia.

La falla radica en que debido a las vibraciones en la máquina, a la dilatación y contracción en el soporte que sostiene el potenciómetro debido a los cambios de temperatura, provocan que el potenciómetro que da la referencia para ubicar la posición del cilindro, quede sin el ajuste óptimo para permanecer en el lugar en el que la calibración cumple los valores detallados en la tabla 1 para el ajuste lateral y tabla 2 para el diagonal. El potenciómetro se sitúa en un lugar oculto en donde no es posible observarlo con facilidad y enterarse de que se encuentra mal fijado y que inminentemente provocará la falla.

La mala posición del potenciómetro de referencia “engaña” al algoritmo del sistema *Navigator automata*, enviándole valores de señal que le permiten mover al cilindro mucho más allá de las posiciones físicamente permitidas, lo que significa que al llegar al límite físico en donde ya no es posible continuar moviendo el cilindro, el motor se ve forzado a continuar trabajando con una carga que no puede mover, esto implica calentamiento excesivo en el motor, alta demanda de corriente que sobrepasa los valores de tolerancia de los componentes electrónicos de la placa AUTOMATA 70036700 220L3670 REV.01 por donde se alimenta el motor. Esta falla deriva en otras consecuencias de tipo mecánicas que consisten en deformar piezas, quebrar soportes y hasta doblar el cilindro de estampación.

Esta falla se ha presentado en los doce cabezales de manera recurrente desde el año 2013 hasta la fecha, de acuerdo a testimonios del personal técnico y de los operadores de la maquinaria, lo que implica un gasto enorme en la adquisición de nuevas placas electrónicas, paro indefinido de maquinaria, largos tiempos de entrega y por ende perdida económicas.

Otra de las causas encontradas que provocan fallas es que el potenciómetro de referencia ya sea lateral o diagonal, al llegar al fin de su vida útil envían valores erróneos a la placa de control permitiendo que como consecuencia el operador intente forzar el ajuste exponiendo el motor, placa electrónica y demás componentes mecánicos a condiciones anómalas que terminan por estropear todos o algunos de los componentes antes mencionados.

CAPITULO 3: IMPLEMENTACIÓN DE PRUEBA PILOTO

Antes de poner en marcha la construcción de la placa final, dado que es una maquinaria de alto costo, es necesario realizar algunas pruebas preliminares que permitan registrar si el diseño propuesto para solucionar las fallas encontradas durante el análisis presenta algún inconveniente que pueda ser tratado a tiempo. En este capítulo se pone en marcha el proceso de diseño y realización de pruebas a manera de simulación, haciendo uso de software y hardware de libre distribución.

Todo el análisis que se describe tanto de hardware como de software en el capítulo 2, se implementó primeramente en una placa de entrenamiento Arduino UNO R3, para simular las diversas situaciones que podrían surgir una vez instalada la nueva placa de control y evitar proceder a la construcción de dicha placa sin antes haber corregido los errores o mal funcionamiento de alguna de las tareas a realizar.

En este trabajo de graduación se han analizado tres placas las cuales conforman el sistema *Navigator Automata*, estas placas electrónicas son:

- Placa base: AUTOMATA 70036600 220L3660 Rev. 0.3
- Placa para control de motores: AUTOMATA 70036700 220L3670 Rev 0.1
- Placa para control de suministro de tinta: AUTOMATA 70036800 220L3670 Rev 0.1

A diferencia de la segmentación que presenta el sistema *Navigator Automata*, el cual esta conformado por tres placas electrónicas, la solución propuesta consiste en una sola placa que controle los motores y el suministro de tinta. Toda la lógica de control se llevara a cabo por un microcontrolador ATMEGA328P. Para las pruebas piloto se hará uso de una placa Arduino UNO R3, ya que es una placa de desarrollo basada en software y hardware de libre distribución, ademas de la facilidad de programación es una buena opción para las pruebas, pero cabe resaltar que esta no es una placa que deba de usarse en el ambiente industrial, ya que como este dispositivo se describe es una placa de desarrollo.

3.1 ENTRADAS A LA PLACA ELECTRÓNICA.

La placa electrónica desarrollada debe de ser capaz de controlar las mismas entradas y salidas que el sistema original. De acuerdo a las partes del diagrama eléctrico que se han mostrado y analizado hasta este punto, se tiene el siguiente listado de entradas en la tabla 3:

ENTRADAS				
No.	**NOMBRE**	**TIPO**	**NAVIGATOR**	**ARDUINO**
1	GND_24VDC	ALIMENTACIÓN	GND	-
2	+24VDC	ALIMENTACIÓN	+24V	-
3	COLOR_MAN	DIGITAL	AI-	2
4	COLOR_AUTO	DIGITAL	-1A93.14	3
5	LAT_IZQ	DIGITAL	-1A93.1.1	4
6	LAT_DER	DIGITAL	-1A93.1.4	5
7	DIA_INF	DIGITAL	-1A93.1.5	6
8	DIA_SUP	DIGITAL	-1A93.1.3	7
9	LAT_REF	ANALÓGICA	-1A12.2.A4	A2
10	DIA_REF	ANALÓGICA	-1A12.2B4	A3
11	TECLADO	ANALÓGICA	-	A0

Tabla 3: Entradas a la placa electrónica.

La segunda columna de la tabla anterior contiene los nombres que de le han asignado a cada entrada, la tercera contiene el tipo. En la tercera columna se puede encontrar el nombre o identificador de la entrada en el diagrama eléctrico de la máquina, mientras que la cuarta columna contiene el número de pin correspondiente en la placa Arduino UNO R3. Para cada entrada según el nombre asignado, se explica su función a continuación:

- GND, +24VDC: Entradas donde se encuentra conectada la tensión de suministro de energía a la placa. Para el caso de la placa desarrollada se implementará una fuente con regulación de tensión para utilizar 5VDC como tensión de alimentación para el microcontrolador y los demás elementos que constituyen la placa.
- COLOR_AUTO: Entrada digital la cual de acuerdo a la lógica de funcionamiento en la sección 2.7.2 se activa cuando el electrodo sensor de nivel no esta sumergido en la tinta.
- COLOR_MAN: Entrada digital para activar la bomba de color de forma manual, esta se toma como prioridad sobre el modo automático como se describió en la sección 2.7.2
- LAT_IZQ: Entrada digital para realizar un ajuste lateral hacia la izquierda.
- LAT_DER: Entrada digital para realizar un ajuste lateral hacia la derecha.
- DIA_INF: Entrada digital para realizar un ajuste diagonal hacia la diagonal inferior.
- DIA_SUP: Entrada digital para realizar un ajuste diagonal hacia la diagonal superior.

- LAT_REF: Entrada analógica para conectar el potenciómetro de referencia del ajuste lateral. En la placa original se maneja un rango de 0-10VDC para la determinación de la posición del cilindro de estampación, pero para la placa desarrollada este rango se sustituirá por uno de 0-5VDC ya que es el rango con el que trabaja el microcontrolador ATMEGA328P.[13]
- LAT_REF: Entrada analógica para conectar el potenciómetro de referencia del ajuste lateral. Al igual que la entrada analógica de referencia diagonal, esta entrada también trabajara con un rango de tensiones de 0-5VDC.
- TECLADO: Esta entrada analógica se ha agregado para poder leer los valores que arroja un pequeño teclado adjunto a la placa electrónica, el propósito de este dispositivo es controlar las entradas de ajuste lateral y diagonal como de la activación manual de la bomba de color.

Una vez que se tiene claro el número y propósito de las entradas lo siguiente es la escritura del código fuente del software controlador, el cual se muestra a continuación:

```
//ENTRADAS DIGITALES.
const int entradaColMan = 2; // Bomba de tinta - Manual.
const int entradaColAut = 3; // Bomba de tinta - Automático.
const int entradaLatIzq = 4; // Entrada para mover a la izquierda.
const int entradaLatDer = 5; // Entrada para mover a la derecha.
const int entradaDiaInf = 6; // Entrada para mover a inferior.
const int entradaDiaSup = 7; // Entrada para mover a superior.

// ENTRADAS ANÁLOGAS.
const int tecladoAnalogo = A0;      // Teclado análogo.
const int referenciaLateral = A2;   // Potenciómetro lateral.
const int referenciaDiagonal = A3;  // Potenciómetro diagonal.
```

Las primeras declaraciones corresponden a entradas digitales para los comandos de activación de la bomba de tinta de forma manual y automática, seguido por los comandos para ajuste lateral y diagonal y terminando con la declaración de las entradas analógicas correspondientes a un teclado analógico y los potenciómetros de referencia de ajuste lateral y diagonal.

Las entradas análogas se manejaran de una forma un poco diferente, ya que mientras el sistema anterior utilizaba un rango de tensiones desde 1.608V hasta 9.170V para el ajuste lateral, y desde 3.367V hasta 6.630V para el diagonal, se utilizará el rango de tensión máximo con el que el ATMEGA328P puede trabajar sin sufrir daños, es cual es de 5VDC [13] ,lo que significa que los valores encontrados en la tabla 1 tendrán que cambiar, pero dado que se trata de una relación de tipo lineal no habría ningún problema en adaptarse al nuevo rango de tensiones para determinar la posición del cilindro de estampación.

3.1.1 TECLADO ANALÓGICO.

Como se pudo observar en la tabla 3 correspondiente a las entradas a la placa electrónica, la única entrada que no se encuentra originalmente en el sistema *Navigator Automata* es la entrada del teclado, pero esta se ha agregado con el fin de hacer pruebas y calibración en el lugar de instalación de la placa con fines de mantenimiento, ya que los selectores disponible para el operador se encuentran a una distancia la cual no es muy práctico estarse desplazando para realizar pruebas. El circuito asociado al teclado mencionado se muestra en la figura 51.

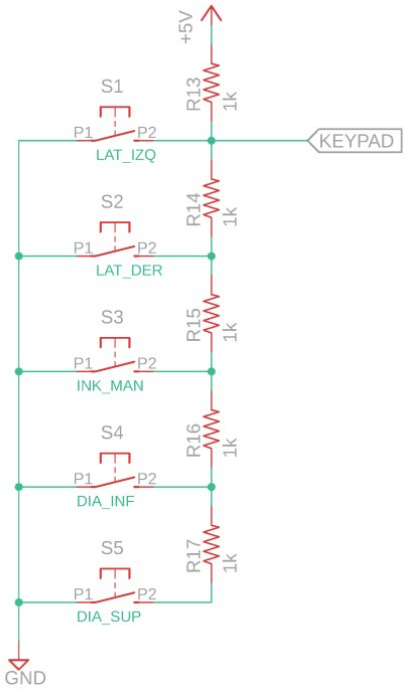


Figura 51: Esquema electrónico de teclado analógico.

El teclado consiste en cinco pulsadores y cinco resistencias que por facilidad se eligieron con valor de 1kOhm cada una, la resistencia R13 se encuentra conectada a Vcc (5VDC) y entre R13 y R14 se encuentra la salida que va hacia una entrada del microcontrolador.

Al no tener presionada ninguna tecla, la tensión en el nodo entre R13 y R14 registra un valor de tensión de 5VDC y la resistencia R13 actúa como una resistencia *pull-up* ya que no se tiene un circuito cerrado. Es fácil deducir que entre la R13 y las demás resistencias hay un divisor de tensión cada vez que se presiona un pulsador.

Ningún pulsador presionado:

$$V_0=\left(\frac{R_{13}}{R_{13}+0}\right)*V_{CC}=\left(\frac{1\,k\Omega}{1\,k\Omega+0}\right)*5\,VDC=(1)*(5\,VDC)=5\,VDC$$

Ec. 1

Pulsador *S1* presionado:

$$V_{S1}=\left(\frac{0}{R_{13}+0}\right)*V_{CC}=\left(\frac{0}{1\,k\Omega+0}\right)*5\,VDC=(0)*(5\,VDC)=0\,VDC$$

Ec. 2

Pulsador *S2* presionado:

$$V_{S2}=\left(\frac{R_{14}}{R_{13}+R_{14}}\right)*V_{CC}=\left(\frac{1\,k\Omega}{1\,k\Omega+1\,k\Omega}\right)*5\,VDC=\left(\frac{1}{2}\right)*(5\,VDC)=2.5\,VDC$$

Ec. 3

Pulsador *S3* presionado:

$$V_{S3}=\left(\frac{R_{14}+R_{15}}{R_{13}+(R_{14}+R_{15})}\right)*V_{CC}=\left(\frac{1k\Omega+1k\Omega}{1k\Omega+(1k\Omega+1k\Omega)}\right)*5VDC=\left(\frac{2}{3}\right)*(5VDC)=3.33VDC$$

Ec. 4

Pulsador *S4* presionado:

$$V_{S4}=\left(\frac{R_{14}+R_{15}+R_{16}}{R_{13}+(R_{14}+R_{15}+R_{16})}\right)*V_{CC}=\left(\frac{1k\Omega+1k\Omega+1k\Omega}{1k\Omega+(1k\Omega+1k\Omega+1k\Omega)}\right)*5VDC=\left(\frac{3}{4}\right)*(5VDC)=3.75VDC$$

Ec. 5

Pulsador *S5* presionado:

$$V_{S5}=\left(\frac{R_{14}+R_{15}+R_{16}+R_{17}}{R_{13}+(R_{14}+R_{15}+R_{16}+R_{17})}\right)*V_{CC}=\left(\frac{1k\Omega+1k\Omega+1k\Omega+1k\Omega}{1k\Omega+(1k\Omega+1k\Omega+1k\Omega+1k\Omega)}\right)*5VDC=\left(\frac{4}{5}\right)*(5VDC)=4VDC$$

Ec. 6

Estos valores de tensión resultantes no son valores que puedan permanecer fijos cada vez que se presiona un pulsador, de hecho oscilan hacia arriba o abajo constantemente, esto provoca que la función asignada a cada pulsador se traslape por no tener un valor fijo. Este problema se puede resolver trabajando con el valor que devuelve el ADC del microcontrolador al presionar cada uno de los pulsadores. De acuerdo a la hoja de datos técnicos del microcontrolador ATMEGA328P, el resultado de la conversión analógica-digital se puede calcular con la ecuación 7 [13]:

$$ADC=\frac{(V_{IN}X\,1024)}{V_{REF}}$$

Ec. 7

Donde: V_{IN}: Tensión de entrada en pin analógico.

V_{REF}: 5VDC

Para cada valor de tensión obtenido anteriormente, al aplicar la formula de la conversión ADC se tiene:

- Para 5V:

$$\mathrm{ADC}_{5V}=\left(\frac{5VDC\,X\,1024}{5VDC}\right)=1024$$

Ec. 8

- Para 0V:

$$\mathrm{ADC}_{0V} = \left(\frac{0\,VDC\ X\ 1024}{5\,VDC} \right) = 0$$

Ec. 9

- Para 2.5V:

$$\mathrm{ADC}_{2.5V} = \left(\frac{2.5\,VDC\ X\ 1024}{5\,VDC} \right) = 512$$

Ec. 10

- Para 3.33V:

$$\mathrm{ADC}_{3.33V} = \left(\frac{3.33\,VDC\ X\ 1024}{5\,VDC} \right) = 681.984 \approx 682$$

Ec. 11

- Para 3.75V:

$$\mathrm{ADC}_{3.75V} = \left(\frac{3.75\,VDC\ X\ 1024}{5\,VDC} \right) = 768$$

Ec. 12

- Para 4V:

$$\mathrm{ADC}_{4V} = \left(\frac{4\,VDC\ X\ 1024}{5\,VDC} \right) = 819.2 \approx 819$$

Ec. 13

Con estos valores obtenidos después de la conversión ADC se puede asignar un rango para cada pulsador de teclado y así evitar el traslape de funciones. Los resultados se muestran a continuación en la tabla 4.

PULSADOR	NOMBRE	VOLTAJE(V)	VALOR ADC	INTERVALO ADC
S1	LAT_IZQ	0.00	0	0-100
S2	LAT_DER	2.50	512	450-550
S3	INK_MAN	3.33	682	598-698
S4	DIA_INF	3.75	768	699-789
S5	DIA_SUP	4.00	819	790-900

Tabla 4: Valores ADC para teclado analógico.

Con el análisis realizado para el teclado analógico se puede proceder a hacer las pruebas correspondientes y crear una rutina especial para lectura de la entrada analógica correspondiente al teclado. Dicha rutina se explica con detalle en la sección 3.7.7

3.2 SALIDAS DE LA PLACA ELECTRÓNICA

Las salidas que se contabilizaron en el sistema *Navigator Automata* fueron las que se muestran en la tabla 5 a continuación:

SALIDAS			
No.	**NOMBRE**	**TIPO**	**NAVIGATOR**
1	GND_18VDC_M1	ALIMENTACIÓN	-1A12.2.A1
2	+18VDC_M1	ALIMENTACIÓN	-1A12.2.A2
3	GND_18VDC_M2	ALIMENTACIÓN	-1A12.2.B1
4	+18VDC_M2	ALIMENTACIÓN	-1A12.2.B2
5	COLOR	DIGITAL	-1A12.X4

Tabla 5: Salidas de la placa del sistema Navigator Automata.

Cada una de las salidas se describen a continuación:

- GND_18VDC_M1 y +18VDC_M1: Estas salidas corresponden a los bornes de conexión según diagrama eléctrico en la tarjeta AUTOMATA 70036700 220L3670 Rev 0.1 para “L-” y “L+” del MOTOR 1 el cual es el ajuste lateral.

- GND_18VDC_M2 y +18VDC_M2: Estas salidas corresponden a los bornes de conexión según diagrama eléctrico en la tarjeta AUTOMATA 70036700 220L3670 Rev 0.1 para “L-” y “L+” del MOTOR 2 el cual es el ajuste diagonal.

- COLOR: Esta salida se encuentra en la placa base AUTOMATA 70036600 220L3660 Rev. 0.3, corresponde a las salidas “X4 OUT4” y ”X5 OUT8”, su función es activar y desactivar la bobina que controla la bomba de tinta. Para esta salidas es muy importante resaltar que tanto para la prueba piloto, como para la nueva placa desarrollada, solo se incluirá una sola salida “COLOR”, esto es debido a que como ya se mencionó, se integraran las funciones de las placas de control de motores y la función del control de suministro de tinta en una sola placa.

Una de las observaciones importantes en este capítulo, es que las pruebas piloto se realizaron utilizando una placa de relés a 5VDC, cuyas bobinas son activadas a través de un dispositivo optocoplador, con esta medida se puede emular las funciones de la placa base AUTOMATA 70036600 220L3660 Rev. 0.3. Esto debido a que las salidas digitales del microcontrolador ATMEGA328P en la placa Arduino UNO R3, son incapaces de manejar grandes cantidades de corriente como para controlar directamente las terminales del motor, por no decir que tampoco es recomendable controlar la bobina del relé directamente ya que de acuerdo a las especificaciones técnicas del microcontrolador sus pines manejan como máximo 40mA, pero lo recomendable son 20mA [13].

A diferencia del caso de las entradas, en el que las mismas entradas del sistema original serán siempre las mismas entradas del sistema nuevo, las salidas se trabajarán con una lista de salidas diferente, debido a que hay que realizar algunas pequeñas adaptaciones para que funcione con la placa Arduino UNO R3, ya que como se ha mencionado, en esta prueba la placa de entrenamiento simplemente corre la lógica de funcionamiento y las salidas se encarga de activar las bobinas de un modulo de relés según sea necesario. La tabla 6 contiene la lista de salidas en la placa Arduino UNO R3.

SALIDAS EN ARDUINO			
No.	**PIN ARDUINO**	**TIPO**	**NOMBRE**
1	8	DIGITAL	SAL_LAT_IZQ
2	9	DIGITAL	SAL_LAT_DER
3	10	DIGITAL	SAL_DIA_INF
4	11	DIGITAL	SAL_DIA_SUP
5	12	DIGITAL	SAL_COLOR

Tabla 6: Salidas de la placa Arduino en prueba piloto.

De la tabla anterior, se puede continuar con la declaración para los pines de la placa Arduino UNO R3 que actuaran como salidas. A continuación se muestra el segmento de código respectivo:

```
// SALIDAS DIGITALES.
//Para ajuste lateral.
const int salidaLatIzq = 8; // Ajuste lateral hacia la izquierda.
const int salidaLatDer = 9; // Ajuste lateral hacia la derecha.

//Para ajuste diagonal.
const int salidaDiaInf = 10; // Ajuste diagonal hacia superior.
const int salidaDiaSup = 11; // Ajuste diagonal hacia inferior.

//Para bomba de color.
const int salidaColor = 12; //Activación de bomba de color
```

Las salidas declaradas en los pines 8 y 9 corresponden al control del ajuste lateral, si se desea girar el eje del motor en una dirección se activara las salida 8, y si se desea la dirección de giro contraria, se activaran las salidas 9. De la misma manera sucede para el caso del ajuste diagonal, la salida 10 activada hace girar el eje del motor en un sentido, y las salidas 11 hace girar el motor en sentido contrario. Por último se tiene la salida 12 que activara el relé que permite el paso de energía para activar la bomba de color.

3.3 CONSTANTES

Como su nombre lo indica son valores que no cambiaran durante toda la ejecución del programa. Son de gran importancia y el propósito del porque se definieron como constantes radica en que son los valores limite para cada ajuste. Si se diera el caso que uno de estos valores cambiara durante la ejecución del programa, resultaría en defectos por ajuste lateral o diagonal, esto causaría perdida de la calidad del tejido estampado y muy posiblemente en destrucción de algún elemento del mecanismos que mueve el cilindro de estampación.

```
#define POS_LAT_MIN  0.50    // Valor mínimo de voltaje lateral.
#define POS_LAT_MAX  3.50    // Valor máximo de voltaje lateral.
#define POS_DIA_MIN  0.50    // Valor mínimo de voltaje diagonal.
#define POS_DIA_MAX  1.00    // Valor máximo de voltaje diagonal.
#define TIEMPO_PULSO  100    // Pulso de tiempo para funcionamiento del motor.
```

3.4 PROGRAMACIÓN DE FUNCIÓN PARA AJUSTE LATERAL

Tal y como se pudo observar en el diagrama con la lógica para el ajuste lateral de la figura 36 existen una serie de pasos que al parecer deberían de ejecutarse y generar resultados satisfactorios, pero ya que no es ese el caso, se añadieron unas pequeñas mejoras que permitirán que el resultado del proceso de ajuste lateral sea satisfactorio tal y como se mostró en el diagrama de la figura 37.

Se comienza por hacer la comprobación de la condición para poder activar el motor.

```
if(voltPosLat > POS_LAT_MIN && voltPosLat < POS_LAT_MAX )
      activarMotor(salidaLatIzq , TIEMPO_PULSO);
```

Se lee el potenciómetro de ajuste lateral, y de acuerdo al valor leído, si se encuentra entre el mínimo y el máximo permitido para este ajuste, se procede a activar el motor de manera que el cilindro de estampación se mueva hacia la izquierda para el caso del segmento de código mostrado. La rutina *activarMotor* se explica con detalle en el apartado de RUTINAS ADICIONALES en la sección siete de este capítulo.

Para el caso de un ajuste lateral hacia la derecha el segmento de código es el siguiente:

```
if(voltPosLat > POS_LAT_MIN && voltPosLat < POS_LAT_MAX )
      activarMotor(salidaLatDer, TIEMPO_PULSO);
```

Como se puede observar en el diagrama de la figura 37, existe una segunda comprobación mas específica para el movimiento de ajuste que se desea realizar, pero se ha decidido que es mucho mas funcional realizar esas comprobaciones en una sola rutina para que la operación se desarrolle de

manera segura, estos detalles se ampliaran en la rutina *autoAjusteSeguro* en la sección de RUTINAS ADICIONALES.

3.5 PROGRAMACIÓN DE FUNCIÓN PARA AJUSTE DIAGONAL

Las instrucciones para el control del ajuste diagonal en realidad no son muy distintas de las correspondientes al lateral como se ha podido observar hasta este punto. El comportamiento de ambos ajustes es similar, con algunas ligeras variaciones, pero el efecto que cada uno produce en el tejido estampado es muy distinto uno del otro. Como se puede observar en el siguiente segmento de código el cual corresponde al movimiento diagonal hacia el lado inferior, se hacen las comprobaciones necesarias leyendo el potenciómetro de referencia diagonal y luego hace el llamado a la rutina *activarMotor* para llevar a cabo el ajuste solicitado.

```
if(voltPosDia > POS_DIA_MIN && voltPosDia < POS_DIA_MAX )
      activarMotor(salidaDiaInf, TIEMPO_PULSO);
```

De igual manera para el ajuste hacia la diagonal superior, como se puede observar a continuación:

```
if(voltPosDia > POS_DIA_MIN && voltPosDia < POS_DIA_MAX )
      activarMotor(salidaDiaSup, TIEMPO_PULSO);
```

3.6 PROGRAMACIÓN DE FUNCIÓN DE SUMINISTRO DE TINTAS.

De acuerdo al diagrama de la lógica del suministro de tinta mostrado en la figura 50, la activación/desactivación de la bomba de tinta es un procedimiento muy sencillo, y se da según las dos formas que se tienen para hacerlo, es decir, de forma manual y automática. Basta entender que si el electrodo se encuentra sumergido en la tinta la bomba de suministro de encontrará apagada, y si es el caso contrario, la bomba se activará llenando de tinta el interior del cilindro. El modo manual simplemente se trata de activar la salida correspondiente a la bomba de color en cualquier momento independientemente del estado del electrodo.

Se comienza leyendo las entradas del microcontrolador. En el caso del accionamiento manual se tienen dos opciones, por el pulsador disponible para el operador, o por el teclado analógico instalado en la placa de control como se muestra a continuación:

```
...if (option_key > 598 && option_key < 698)
            return button_ink_pump;…
```

```
...else if(digitalRead(entradaColMan)==LOW || \
           leerTeclado()==5)
           return ink_pump_man;...
```

O bien por la entrada activada por el electrodo en el interior del cilindro de estampación.

```
... else if(digitalRead(entradaColAut)==LOW)
           return ink_pump_aut; ...
```

3.7 RUTINAS ADICIONALES PARA SOFTWARE DE PRUEBA PILOTO.

Con el fin de disminuir el número de líneas de código y de presentar un algoritmo entendible y ordenado para futuras mejoras o revisiones, se han realizado algunas rutinas que reúnen todas las instrucciones que resultan repetitivas o que cumplen funciones especiales dentro del algoritmo de control de ajuste lateral/diagonal y control de la bomba de color.

3.7.1 leerVoltaje

Debido a que el microcontrolador ATMEGA328P en sus entradas analógicas tiene un ADC con una resolución de 10 bits, es decir 2^{10}, se tiene un resultado de 1024 valores numéricos que corresponden a la señal que se está leyendo [13]. Por ejemplo la señal de entrada para el caso de cualquiera de las dos referencias, ya sea lateral o diagonal, varia desde 0 a 5V, que corresponde a los valores de 0 y 1023 respectivamente. Se puede observar que es mucho más sencillo trabajar con un valor del cual se puede tener la noción como lo es el valor de tensión, el cual se estará utilizando muy a menudo en el algoritmo de control, en lugar de el rango de valores de 0 a 1023 del cual no se esta familiarizado.

Por lo tanto al aplicar la ecuación 7: $$ADC = \frac{(V_{IN}\, X\, 1024)}{V_{REF}}$$ [13]

Y despejando para V_{IN}: $$V_{IN} = \left(\frac{ADC\, x\, V_{REF}}{1024} \right)$$ *Ec. 14*

Donde: $V_{REF} = V_{CC} = 5VDC$

Argumentos de la función:

- entradaAnaloga : Se trata de una variable de tipo entero, haciendo referencia al numero de pin analógico del microcontrolador que se desea leer y mostrar su valor en una magnitud conocida, para este caso tensión eléctrica en voltios.

Valor de retorno: La función retorna un número de coma flotante, el cual se interpreta como la tensión eléctrica leída por esa entrada.

```
float leerVoltaje(int entradaAnaloga){

    int valorLeido = analogRead(entradaAnaloga);
    float voltaje = valorLeido * (5.0 / 1023);
    return voltaje;


}
```

3.7.2 leerEntrada

Durante todo el proceso de ejecución de la lógica de trabajo del sistema para ajuste del cilindro de estampación, el algoritmo se encuentra a la espera de una entrada la cual represente la dirección deseada de ajuste o bien la activación/desactivación de la bomba de color. La función *leerEntrada* se encarga de esta tarea repetitiva y devuelve como resultado un número entero el cual denota la petición realizada por el operario.

Argumentos de la función: Sin argumentos

Valor de retorno: Retorna un numero entero entre 0 y 10 cuyo significado se muestra en la tabla 7:

VALOR DE RETORNO	SIGNIFICADO
0	Ninguna entrada digital activa.
1	Petición de ajuste lateral hacia la izquierda.
2	Petición de ajuste lateral hacia la derecha.
3	Petición de ajuste diagonal hacia inferior.
4	Petición de ajuste diagonal hacia superior.
5	Activar bomba manual.
6	Activar bomba automático.
7	El valor de tensión leído esta por abajo del limite lateral izquierdo permitido.
8	El valor de tensión leído esta por arriba del limite lateral derecho permitido.
9	El valor de tensión leído esta por abajo del limite diagonal inferior permitido.
10	El valor de tensión leído esta por arriba del limite diagonal superior permitido.

Tabla 7: Valores de retorno de la rutina leerEntrada.

```
int leerEntrada(void){


    int move_to_left = 1;
```

```
int move_to_right = 2;
int move_to_down = 3;
int move_to_up = 4;
int ink_pump_man = 5;
int ink_pump_aut = 6;

if(digitalRead(entradaLatDer)==HIGH &&  \
    digitalRead(entradaLatIzq)==LOW ||   \
    leerTeclado()==1)
    return move_to_left;
else if(digitalRead(entradaLatDer)==LOW && \
    digitalRead(entradaLatIzq)==HIGH ||     \
    leerTeclado()==2)
    return move_to_right;
else if(digitalRead(entradaDiaInf)==LOW && \
    digitalRead(entradaDiaSup)==HIGH ||      \
    leerTeclado()==3)
    return move_to_down;
else if(digitalRead(entradaDiaInf)==HIGH &&  \
    digitalRead(entradaDiaSup)==LOW ||        \
    leerTeclado()==4)
    return move_to_up;
else if(digitalRead(entradaColMan)==LOW || \
    leerTeclado()==5)
    return ink_pump_man;
else if(digitalRead(entradaColAut)==LOW)
    return ink_pump_aut;
else if(leerVoltaje(referenciaLateral) < POS_LAT_MIN)
    return 7;
else if(leerVoltaje(referenciaLateral) > POS_LAT_MAX)
    return 8;
else if(leerVoltaje(referenciaDiagonal) < POS_DIA_MIN)
    return 9;
else if(leerVoltaje(referenciaDiagonal) > POS_DIA_MAX)
```

```
        return 10;
    else
        return 0;
}
```

Las primeras cuatro comprobaciones tienen el objetivo de leer la dirección de ajuste deseada pero a la vez se debe de cumplir la condición de que no se pidan ambas direcciones del mismo ajuste simultáneamente, esto evitara que debido a una mala operación o deterioro del selector de ajuste se obtengan resultados inesperados que terminen en destrucción de componentes o mala calidad del producto final. Las siguientes condicionales las cuales retornan los valores 5 y 6 corresponden a la activación de la bomba de color de manera manual y automática respectivamente. Las siguientes cuatro comprobaciones de la 7 a la 10 verifican si se ha alcanzado alguno de los valores mínimo o máximo en el ajuste lateral y diagonal, esto es de utilidad para lanzar un aviso al operador aunque llegada a una de estas condiciones es la función *autoAjusteSeguro* quien se encarga de la corrección del problema.

3.7.3 activarMotor

Esta función tiene como objetivo activar cualquiera de los dos motores dependiendo del ajuste solicitado. De acuerdo al comportamiento observado en el sistema *Navigator Automata*, el motor no trabaja continuamente, sino con pequeñas pausas entre conexión y desconexión de la tensión en los terminales del motor. La duración de las pausas no pudieron ser calculadas por la rapidez con que suceden, pero la función *activarMotor* permite ingresar un valor en milisegundos con el que se puede igualar el comportamiento del sistema original con prueba y error.

Argumentos de la función:

- salida: Variable de tipo entero, representa el numero de pin del microcontrolador a ser activado el cual al energizar el relé dejara pasar la energía al motor.
- msec: Variable de tipo entero que representa el valor en milisegundos que durará la activación del motor.

Valor de retorno: Sin valor de retorno.

```
void activarMotor(int salida ,  int msec){

    monitor();
    apagarSalidas();
    digitalWrite(salida, HIGH);
    delay(msec);

}
```

3.7.4 autoAjusteSeguro

Esta rutina es una de las soluciones mas acertadas que surgieron en este trabajo de graduación, debido a que es la solución que evita que por la mala operación o ya sea por problemas en el selector de ajuste o

en el potenciómetro de referencia lateral o diagonal resulten las fallas que presentaba el sistema *Navigator Automata.* Su trabajo consiste en leer continuamente las entradas analógicas correspondientes a los potenciómetros de referencia lateral y diagonal, y evaluar si se encuentran dentro de sus limites mínimo y máximo para ser operados manualmente. En el caso de que el operador no preste atención a la posición del cilindro de estampación que esta alcanzando sus limites, o surja otra situación relacionada con las causantes de falla descritas en la sección 2.8, la función *autoAjusteSeguro* toma el control y se encarga de mover el cilindro de forma automática hasta encontrar la posición mas cercana en donde la operación manual sea segura y retorna el control al operador de la maquina.

Argumentos de la función:

- alp: Variable de tipo coma flotante el cual representa el valor de la tensión que el potenciómetro de referencia lateral se encuentra leyendo.
- adp: Variable de tipo coma flotante el cual representa el valor de la tensión que el potenciómetro de referencia diagonal se encuentra leyendo.
- Valor de retorno: Sin valor de retorno.

```
void autoAjusteSeguro(float alp, float adp){

    if(alp < POS_LAT_MIN){//posición=limite izquierda
        activarMotor(salidaLatDer, TIEMPO_PULSO);
    }
    else if(alp > POS_LAT_MAX){//posición=limite derecha
        activarMotor(salidaLatIzq, TIEMPO_PULSO);
    }

    if(adp < POS_DIA_MIN){//posición=limite abajo
        activarMotor(salidaDiaSup, TIEMPO_PULSO);
    }
    else if(adp > POS_DIA_MAX){//posición=limite arriba
        activarMotor(salidaDiaInf, TIEMPO_PULSO);
    }

}
```

3.7.5 monitor

Esta función es de una gran utilidad para observar los eventos que están sucediendo durante la ejecución del programa. Puede mostrar mensajes mas completos y explícitos de los que podría mostrarse en una pequeña pantalla LCD de 2x16. Con la ayuda del monitor serial incluido en el IDE de

Arduino se puede establecer comunicación con el microcontrolador y realizar las tareas de control y monitoreo que sean necesarias. Esta rutina hace uso del valor de retorno de la función *leerEntrada* a fin de mostrar los eventos tanto en el monitor serial con información detallada, como en una pequeña pantalla LCD de 2x16 integrado en el hardware con información mas escueta.

Argumentos de la función: Sin argumentos.

Valor de retorno: Sin valor de retorno.

```
void monitor(void){

    float lv = leerVoltaje(referenciaLateral);
    float dv = leerVoltaje(referenciaDiagonal);
    int direccionAjuste = leerEntrada();
    int option_key = analogRead(tecladoAnalogo);

    Serial.print("   ");
    Serial.print(direccionAjuste);
    Serial.print("   ");
    Serial.print(option_key);
    Serial.print("   ");
    Serial.print(lv);
    Serial.print("V   ");
    Serial.print(dv);
    Serial.print("V");

    switch(direccionAjuste){

        case 0:
            Serial.println("  Status: Reading . . .");
            lcd.setCursor(0,0);
            lcd.print("Estado: Leyendo.");
            lcd.setCursor(0,1);
            lcd.print("VL:");
            lcd.setCursor(3,1);
            lcd.print(lv);
            lcd.setCursor(8,1);
            lcd.print("VD:");
            lcd.setCursor(11,1);
            lcd.print(dv);
            break;

        case 1:
            Serial.println("  Status: Running left");
            lcd.setCursor(0,0);
            lcd.print("Estado: Lat.izq");
            lcd.setCursor(0,1);
            lcd.print("VL:");
            lcd.setCursor(3,1);
            lcd.print(lv);
            lcd.setCursor(8,1);
            lcd.print("VD:");
            lcd.setCursor(11,1);
```

```
    lcd.print(dv);
    break;

case 2:
    Serial.println("  Status: Running right");
    lcd.setCursor(0,0);
    lcd.print("Estado: Lat.der");
    lcd.setCursor(0,1);
    lcd.print("VL:");
    lcd.setCursor(3,1);
    lcd.print(lv);
    lcd.setCursor(8,1);
    lcd.print("VD:");
    lcd.setCursor(11,1);
    lcd.print(dv);
    break;

case 3:
    Serial.println("  Status: Running down");
    lcd.setCursor(0,0);
    lcd.print("Estado: Dia.inf");
    lcd.setCursor(0,1);
    lcd.print("VL:");
    lcd.setCursor(3,1);
    lcd.print(lv);
    lcd.setCursor(8,1);
    lcd.print("VD:");
    lcd.setCursor(11,1);
    lcd.print(dv);
    break;

case 4:
    Serial.println("  Status: Running up");
    lcd.setCursor(0,0);
    lcd.print("Estado: Dia.sup");
    lcd.setCursor(0,1);
    lcd.print("VL:");
    lcd.setCursor(3,1);
    lcd.print(lv);
    lcd.setCursor(8,1);
    lcd.print("VD:");
    lcd.setCursor(11,1);
    lcd.print(dv);
    break;

case 5:
    Serial.println("  Status: Ink pump -> Manual!");
    lcd.setCursor(0,0);
    lcd.print("Estado:Color man");
    lcd.setCursor(0,1);
    lcd.print("VL:");
    lcd.setCursor(3,1);
    lcd.print(lv);
    lcd.setCursor(8,1);
```

```
        lcd.print("VD:");
        lcd.setCursor(11,1);
        lcd.print(dv);
        break;
        break;

    case 6:
        Serial.println("  Status: Ink pump -> Auto!");
        lcd.setCursor(0,0);
        lcd.print("Estado:Color Aut");
        lcd.setCursor(0,1);
        lcd.print("VL:");
        lcd.setCursor(3,1);
        lcd.print(lv);
        lcd.setCursor(8,1);
        lcd.print("VD:");
        lcd.setCursor(11,1);
        lcd.print(dv);
        break;

    case 7:
        Serial.println("  Warning: Lateral lower limit reached!");
        lcd.setCursor(0,0);
        lcd.print("Estado: Auto-der");
        lcd.setCursor(0,1);
        lcd.print("VL:");
        lcd.setCursor(3,1);
        lcd.print(lv);
        lcd.setCursor(8,1);
        lcd.print("VD:");
        lcd.setCursor(11,1);
        lcd.print(dv);
        break;

    case 8:
        Serial.println("  Warning: Lateral higher limit reached!");
        lcd.setCursor(0,0);
        lcd.print("Estado: Auto-izq");
        lcd.setCursor(0,1);
        lcd.print("VL:");
        lcd.setCursor(3,1);
        lcd.print(lv);
        lcd.setCursor(8,1);
        lcd.print("VD:");
        lcd.setCursor(11,1);
        lcd.print(dv);
        break;

    case 9:
        Serial.println("  Warning: Diagonal lower limit reached!");
        lcd.setCursor(0,0);
        lcd.print("Estado: Auto-sup");
        lcd.setCursor(0,1);
        lcd.print("VL:");
```

```
            lcd.setCursor(3,1);
            lcd.print(lv);
            lcd.setCursor(8,1);
            lcd.print("VD:");
            lcd.setCursor(11,1);
            lcd.print(dv);
            break;

        case 10:
            Serial.println("  Warning: Diagonal higher limit reached!");
            lcd.setCursor(0,0);
            lcd.print("Estado: Auto-inf");
            lcd.setCursor(0,1);
            lcd.print("VL:");
            lcd.setCursor(3,1);
            lcd.print(lv);
            lcd.setCursor(8,1);
            lcd.print("VD:");
            lcd.setCursor(11,1);
            lcd.print(dv);
            break;

        default:
            Serial.println("  Warning: No recognized status!");
            lcd.setCursor(0,0);
            lcd.print("ERROR: ****** ");
            lcd.setCursor(0,1);
            lcd.print("VL:");
            lcd.setCursor(3,1);
            lcd.print(lv);
            lcd.setCursor(8,1);
            lcd.print("VD:");
            lcd.setCursor(11,1);
            lcd.print(dv);
            break;
    }

}
```

3.7.6 apagarSalidas

Esta rutina se ejecuta por seguridad, bien podría parecer que es innecesaria, pero actúa como un tipo de redundancia apagando todas las salidas que activan los relés para mantenerlos en ese estado mientras se lee una entrada en la que se necesite un ajuste lateral o diagonal, luego de haber realizado el ajuste solicitado vuelve a ejecutarse.

Argumentos de la función: Sin argumentos.

Valor de retorno: Sin valor de retorno.

```
void apagarSalidas(void){

    monitor();
    digitalWrite(salidaLatIzq, LOW);
    digitalWrite(salidaLatDer, LOW);
    digitalWrite(salidaDiaInf, LOW);
    digitalWrite(salidaDiaSup, LOW);
    digitalWrite(salidaColor, LOW);

}
```

3.7.7 leerTeclado

Esta rutina se encarga de leer la entrada correspondiente a un pequeño teclado analógico creado para introducir todos los comando manuales del sistema, es decir, activación de la bomba, ajuste lateral a la izquierda y derecha, ajuste diagonal inferior y superior, estas cinco funciones se integran en este pequeño teclado a fin de que se puedan ejecutar desde la placa desarrollada con fines de mantenimiento.

El teclado integrado en la placa electrónica desarrollada posee un total de seis pulsadores, de los cuales cinco son configurables y uno es el reset del microcontrolador.

Argumentos: Sin argumentos.

Valor de retorno: Retorna un entero entre 1 y 5, los cuales representan las funciones de acuerdo a la tabla 8:

VALOR DE RETORNO	SIGNIFICADO
1	Petición de ajuste lateral hacia la izquierda.
2	Petición de ajuste lateral hacia la derecha.
3	Petición de ajuste diagonal hacia inferior.
4	Petición de ajuste diagonal hacia superior.
5	Activar bomba manual.

Tabla 8: Valores de retorno de la rutina leerTeclado.

```
int leerTeclado(void){

    int button_lateral_left = 1;
    int button_lateral_right = 2;
    int button_diagonal_down = 3;
    int button_diagonal_up = 4;
    int button_ink_pump = 5;
    int option_key = analogRead(tecladoAnalogo);

    if (option_key > 1000)
        return 0;
```

```
    if (option_key > 0 && option_key < 100)
        return button_lateral_left;

    if (option_key > 450 && option_key < 550)
        return button_lateral_right;

    if (option_key > 598 && option_key < 698)
    return button_ink_pump;

    if (option_key > 699 && option_key < 789)
        return button_diagonal_down;

    if (option_key > 790 && option_key < 900)
        return button_diagonal_up;

    return 0;

}
```

3.7.8 intro

Esta rutina se encarga de mostrar información acerca del autor, y versión del software al energizar el sistema o presionar el pulsador de RESET del microcontrolador, tiene una duración de unos pocos segundos y solo se muestra en la pantalla LCD 2x16. En la porción de código mostrada a continuación se puede apreciar que es la versión “NVFR v1_5”.

```
void intro(void){

    String stringTemp, stringDot;
    int i=0;
    stringTemp = String(".");
    stringDot = String(".");

    for(i=0;i<15;i++){
        lcd.setCursor(0,0);
        lcd.print("Navigator free");
        lcd.setCursor(0,1);
        stringTemp += stringDot;
        lcd.print(stringTemp);
        delay(100);
        lcd.clear();
    }
    lcd.setCursor(0,0);
    lcd.print("Elaborado por:");
    lcd.setCursor(0,1);
    lcd.print("Edgardo Gálvez");
    delay(1500);

    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Versión software");
```

```
    lcd.setCursor(4,1);
    lcd.print("NVFR v1_5");
    delay(1500);
    lcd.clear();

}
```

3.8 RESULTADOS DE LA PRUEBA PILOTO

Inicialmente la prueba se desarrolló como una simulación, en donde se hizo uso de la placa Arduino UNO R3 la cual posee el microcontrolador ATMEGA328P responsable de la lógica de control del nuevo sistema. Para dicho controlador se desarrollaron varias versiones del software controlador, de manera que cada versión va de la mas sencilla a la mas compleja, agregando en cada una funciones que van mejorando la operación. Cada una fue puesta a prueba tanto fuera de la máquina a manera de simulación, como dentro de la misma para realizar las observaciones respectivas y obtener información en el sitio de las fallas presentadas, o posibles mejoras. Al software controlador se le dio el nombre de NVFR en alusión al nombre "*Navigator Free*", la alternativa libre al sistema propietario *Navigator Automata*. El historial de las versiones se presenta a continuación en la tabla 9.

HISTORIAL DE VERSIONES	
VERSIÓN	OBSERVACIONES
NVFR V1_0	• Solo para ajuste lateral. Prueba en placa Arduino con resultados satisfactorios.
NVFR V1_1	• Se agregó función de ajuste diagonal. Prueba en placa Arduino, resultados satisfactorios.
NVFR V1_2	• Pruebas en la máquina para ajuste lateral y diagonal, las entradas presentan rebotes. • Se agregó rutina para apagar relés en condiciones de reposo del sistema, ya que por el problema de los rebotes se activan las salidas por si solas. • Se agregó la rutina *leerEntrada* para ordenar código fuente. • Se agregó rutina *monitor* para darle seguimiento a los distintos problemas que podría presentar el sistema. • Se agregó rutina *intro* para tener el control de la versión de software subida en el microcontrolador.
NVFR V1_3	• Se integró entradas para bomba de color, así como la salida correspondiente para esta función. • se resolvió problema de rebotes en las entradas utilizando las resistencias PULLUP del microcontrolador. Este problema se tratará con mejores resultados en la placa NVFR con los diferentes dispositivos que integran el hardware de las entradas.
NVFR V1_4	• Se integró rutina de lectura del teclado analógico, lectura y conversión de entradas analógicas y activar motores. • Se escribió rutina de *autoAjusteSeguro*,debido a las observaciones realizadas en las pruebas anteriores, ya que los operarios insistían en querer mover el cilindro

	de estampación mas allá de los valores mínimo y máximo permitido en los ajustes lateral y diagonal.
NVFR V1_5	• Se borraron muchas lineas y se reescribieron para acortar el código fuente a fin de mostrar un mayor orden y comprensión del mismo. • Pruebas en maquinaria, la rutina autoAjusteSeguro trabajo exitosamente. • El teclado analógico presentó traslape de las funciones asignadas para LAT_IZQ y LAT_DER.

Tabla 9: Historial de versiones del software controlador de la placa NVFR.

Basándose en la tabla anterior, se puede ver que algunas de las funciones que integran el nuevo sistema *Navigator Free*, funcionaron satisfactoriamente en la primera vez, pero otras de las mismas no lo hicieron así. Algunos de los resultados no deseados fueron observados hasta que el sistema fue probado en la maquinaria, y corregidos en las versiones siguientes.

Hasta este punto se tiene pendiente la mejora del diseño del teclado, el cual en la ultima versión del software presentó efectos indeseados. El diseño mas estable del software y el hardware serán presentados en el capítulo cuatro de este trabajo de graduación.

CAPITULO 4: PROPUESTA FINAL Y PUESTA EN MARCHA

El análisis realizado durante el proceso de estampado respecto a operación y funcionamiento de los distintos elementos que componen el sistema *Navigator Automata* se puede considerar acertado ya que la lógica de trabajo observada replicada junto a las mejoras añadidas, han dado como resultado que las pruebas realizadas con la placa Arduino UNO R3 sean satisfactorias. Sin embargo, hasta este punto aunque la mayoría de los resultados sean los esperados, no es una opción dejar en operación continua a la placa Arduino UNO R3 utilizada, ya que los elementos observados en la maquinaria estampadora ROTA-TG/116, en su mayoría, tales como contactores, relés de control, sensores, placas electrónicas y demás son elementos que se alimentan al nivel de tensión de control de la maquinaria, es decir, 24VDC. Los pulsadores, selectores y demás dispositivos de entrada también hacen uso de la tensión de control de la máquina. Por lo tanto hay que adaptarse a esta situación y proveer una solución mucho mas robusta que se apegue a las condiciones de trabajo que se tienen en el lugar y que pueda trabajar de manera permanente sustituyendo las placas del sistema *Navigator Automata*. Es aquí en este punto en donde se presenta la propuesta final que sustituya las tres placas que conforman el sistema anterior, teniendo las siguientes consideraciones:

- Debe ser mas robusta que el sistema original, es decir, debe de tener la capacidad de durar muchas mas horas de trabajo, ya que de acuerdo al historial de cambios de las placas del sistema Navigator Automata, estas apenas suelen durar entre uno y tres meses.
- Debe de ser de bajo costo. La placa desarrollada debe de cumplir la condición de ser de bajo costo en comparación con las placas del sistema original, es decir, hasta un 30% del costo del hardware propietario.
- Debe de cumplir con todas las tareas que realiza el sistema Navigator Automata.
- El manejo de dicho sistema debe de ser sencillo, de manera que no haya complicaciones al trabajarlo para evitar confusiones, la operación debe de ser bastante lógica para el usuario.

Bajo dichas consideraciones se presenta la propuesta nombrada “Navigator Free”, abreviada “NVFR”, controlada por el software del mismo nombre, desarrollados para el control de ajustes lateral y diagonal, ademas de integrar la funcionalidad del control de la bomba de color.

4.1 NAVIGATOR FREE – HARDWARE

La placa *Navigator Free* es una placa desarrollada para la industria textil, específicamente para la maquinaria estampadora modelo ROTA-TG/116, del fabricante Zimmer Maschinenbau GmbH,la cual utiliza la técnica de serigrafía rotativa. La placa *NVFR* se encarga tanto del control de ajuste lateral y diagonal como del control de bomba de color de estaciones de trabajo conocidas como cabezales de estampación.

Dicha placa se ha desarrollado como sustituto de su similar comercial *Navigator Automata*, del fabricante CANNON-AUTOMATA, especialistas en fabricación de dispositivos dedicados al

automatismo industrial. El sistema *Navigator Automata* del cual, no se duda de sus capacidades, presenta fallas recurrentes las cuales en un corto período de tiempo dejan la placa inutilizable ya sea por mala operación o por condiciones talvéz imprevistas por los desarrolladores.

Mientras que en la estación de control original de la maquinaria se hace uso de tres placas electrónicas para dicho fin, la NVFR hace uso de una sola placa electrónica capaz de realizar las tres funciones. Toda la lógica de control se encuentra gobernada por un microcontrolador ATMEGA328P.

El hardware NVFR cumple con las siguientes especificaciones técnicas:

- Tensión de alimentación: 5VDC
- Consumo de corriente: 76mA en modo lectura
- Numero de entradas digitales: 6 (24VDC)
- Numero de entradas analógicas: 3 (5VDC)
- Resolución de entradas analógicas: 10bits
- Numero de salidas digitales: 7 (5VDC, 500mA)
- Microcontrolador: ATMEGA328PU
- Pantalla LCD 2X16 para visualización de parámetros de interés.
- Teclado de 6 teclas. (5 configurables + 1 Reset del controlador)
- Temperatura de trabajo: Ha trabajado en ambientes desde 20°C hasta 80°C
- Dimensiones:150mmX100mm
- Montaje: Riel DIN

4.1.1 MICROCONTROLADOR ATMEGA328P.

El dispositivo electrónico ATMEGA328P es un microcontrolador de 8 bits con 32Kbytes de memoria flash, 1Kbyte de EEPROM, 2Kbytes de memoria SRAM, con 23 lineas programables de entrada/salida (Figura 52), de las cuales 6 pueden ser utilizados como pines analógicos y que puede operar en el rango de 2.7 a 5.5VDC. Este microcontrolador es un producto del fabricante y diseñador de dispositivos semiconductores Atmel Corporation, el cual tiene a disposición una gran variedad de microcontroladores disponibles de acuerdo a la necesidad que se tenga.

El ATMEGA328P es un microcontrolador que puede encontrarse en gran variedad de dispositivos electrónicos que se han desarrollado en los últimos años gracias al incremento masivo en la utilización del mismo en las placas de entrenamiento Arduino.

Suele existir las confusión de pensar que el microcontrolador en cuestión es hardware libre, pero en realidad esta afirmación esta muy lejos de la realidad, ya que simplemente es un elemento comercializado y utilizado en las placas de entrenamiento y bien pudiera ser sustituido por cualquier otro tipo de microcontrolador de otro fabricante, aunque cada microcontrolador de acuerdo a su fabricante tiene un conjunto único de instrucciones, y por esa razón los programas escritos para una marca de microcontrolador en especifico, no son compatibles para otro microcontrolador de otro fabricante.[12]

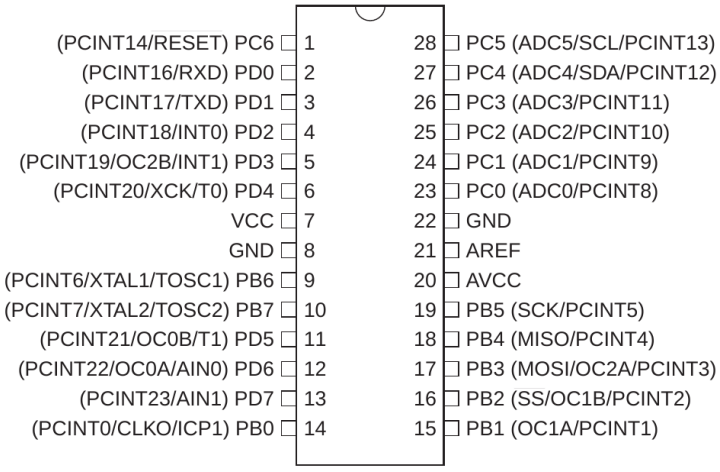


Figura 52: Pinout de microcontrolador ATmega328PU

Los criterios por los cuales se ha hecho uso del ATmega328P en la placa NVFR son debido a que no es una aplicación que requiera un gran número de entradas y salidas, de hecho como se podrá ver en el diseño final, no todos los pines están siendo utilizados, ademas, el proceso para el cual este microcontrolador sera el que tome las decisiones, se debe a que no es una aplicación que requiera altas velocidades en el procesamiento.[12]

También cabe mencionar que otro criterio para la utilización de este microcontrolador es que la programación resulta relativamente sencilla y posee todas las instrucciones de control del flujo que poseen otros lenguajes de programación de amplia utilización, por lo que es una buena opción a la hora de implementar la lógica de gran número de procesos.[12]

Por último en el criterio de los costos y la adquisición del microcontrolador, el ATmega328P resulta apropiado porque se trata de un dispositivo accesible tanto en costo como en disponibilidad.[12]

4.1.2 OPTOCOPLADOR PC817

Debido a la necesidad de aislar la etapa inteligente de la placa NVFR de la tensión de control que la maquinaria ROTA-TG/116 posee, la cual es de 24VDC, se hace necesario el uso de un dispositivo que permita hacer dicho aislamiento, por lo que se ha hecho uso del optocoplador PC817 [20], el cual permite aislar de manera óptica las entradas del microcontrolador.

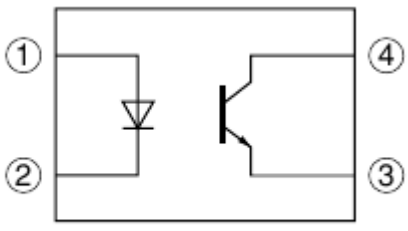


Figura 53: Esquema de un optocoplador.

El funcionamiento de este dispositivo se da cuando al hacer pasar una corriente apropiada en el emisor de luz(pines 1 y

2) el cual es un fotodiodo, ilumina al dispositivo receptor, el cual es un fototransistor (pines 3 y 4) que al recibir el nivel adecuado de luz conduce de colector a emisor, de esta manera queda protegido el microcontrolador de cualquier evento externo que lo pueda dañar, como un cortocircuito que es de los mas comunes.

El optocoplador PC817 se ha utilizado en cada una de las entradas de la placa NVFR, con su configuración respectiva la cual se explica en la sección 4.1.5 ENTRADAS DIGITALES. Uno de los criterios por los cuales se ha elegido este tipo de optocoplador dentro de la gran gama que existe, es debido a su disponibilidad y a que por su encapsulado el cual solo contiene cuatro pines(figura 53), ahorra espacio en la placa, lo que es muy conveniente debido a que el espacio debe de utilizarse optimamente a fin de que en el proceso de ruteo de las pistas por donde se conducen las diversas señales no encuentre ningún problema y pueda llevarse a cabo de manera satisfactoria.

4.1.3 INVERSOR SCHMITT-TRIGGER.

Como ya es sabido, la placa electrónica NVFR debe de operar en un ambiente industrial, en el cual se ven involucrados diversos tipos de cargas conectadas a la red eléctrica, algunas de estas cargas provocan niveles de ruido que afectan los demás dispositivos. Las altas temperaturas también son causantes de variaciones entre otros factores[21]. En las pruebas piloto realizadas en el capítulo tres de este trabajo de graduación, se pudo observar que al activar una entrada hacia el microcontrolador esta no era una señal estable, por lo tanto provocada resultados no esperados y esa es una de las razones por las cuales no se debe de dejar en operación la placa de entrenamiento.

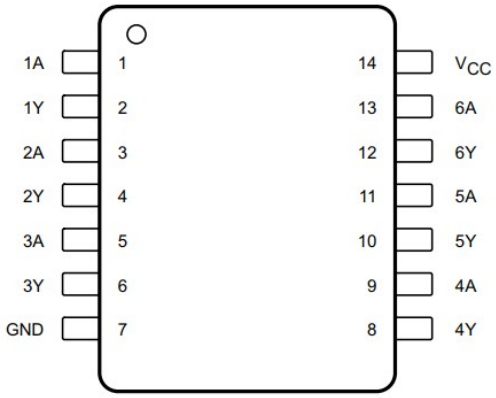


Figura 54: Pinout de IC7414

Los niveles lógicos que se manejan en el microcontrolador son muy importantes debido a que son los niveles con valores inapropiados los que causan los fallos inesperados en la ejecución del proceso. Para las muchas causas del ruido también existen igual numero de soluciones, pero se ha elegido hacer uso de una muy popular y accesible, la cual es utilizar una compuerta inversora Schmitt-trigger, que no es mas que una compuerta que al recibir un valor lógico en su entrada, genera en su salida el nivel lógico opuesto, es decir, para un cero lógico en la entrada, se obtendrá un uno lógico en la salida y viceversa, con la diferencia de otras compuertas inversoras(Figura 54), de que esta tiene la peculiaridad de diferenciar muy precisamente los niveles lógicos.

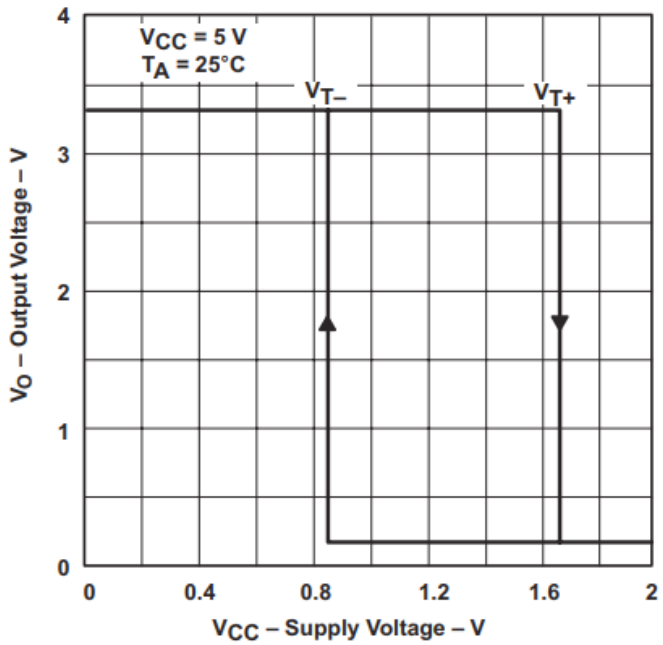


Figura 55: Histéresis de compuertas Schmitt trigger

Esta capacidad es gracias a que este circuito integrado opera en base a histeresis(Figura 55), es decir dos umbrales o niveles como se muestra en la figura[22], de manera que si la entrada esta abajo del umbral mas bajo, la salida sera un uno lógico y si la entrada esta arriba del umbral mas alto, la salida será un cero lógico para el caso del Schmitt-Trigger con salida inversora, de esta manera se evita que los valores intermedios se confundan con los dos valores lógicos aceptados por el microcontrolador.

4.1.4 ARREGLO DE TRANSISTORES DE ALTA CORRIENTE ULN2003A.

Las especificaciones técnicas del microcontrolador ATmega328P son claras en advertir que el valor máximo de corriente por pin es de 40mA@5VDC, y que no se debe de sobrepasar los 100mA por puerto [13], estos datos bastan para saber que no es posible conectar una carga tal como un relé o un pequeño motor directamente a las salidas del microcontrolador, es por esa razón que se hará uso del circuito integrado ULN2003A. Este circuito integrado, de acuerdo a su hoja de datos técnicos, posee dos canales,uno que consta de siete entradas, y otro con mismo número de salidas, una para su respectiva entrada de acuerdo a la figura 56 [23].

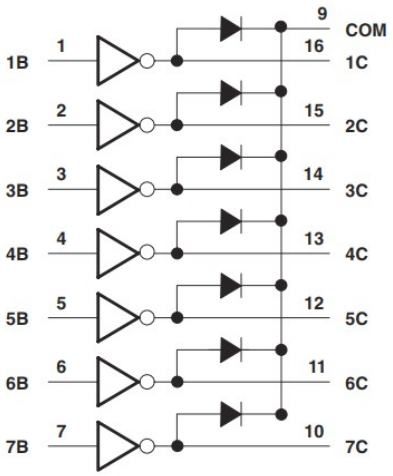


Figura 56: Pinout IC ULN2003A.

Se trata de un circuito integrado con un arreglo de siete transistores en configuración Darlington, esta configuración es especialmente útil cuando se necesitan altas corrientes[24].

Las entradas están marcadas con la letra “B”, mientras que las salidas con la letra “C” (Figura 56), cada entrada posee un diodo con el que se garantiza la operación segura, ya que cada uno de los cátodos se encuentran conectados en una sola linea común, en donde se conecta la polaridad positiva de las cargas que este circuito integrado va a alimentar, mientras que las salidas se encargan de conducir la polaridad

negativa cerrando el circuito y operándolo de manera segura para que no hayan retornos a través de la carga.

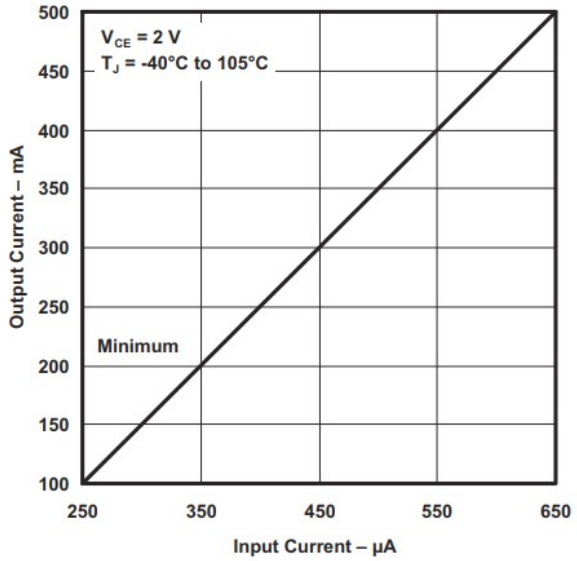


Figura 57: Corrientes I_{in} vrs I_{out} de IC ULN2003A.

Otro dato interesante de este circuito integrado es que si se desea controlar mas corriente de la que una de las salidas puede controlar, se pueden poner tantas salidas en paralelo como sea necesarias para la carga que se tenga, siempre y cuando también igual número de entradas respectivas se configuren de la misma manera. La corriente que puede controlar cada salida del integrado ULN2003A es de 500mA, suficiente para controlar la bobina de un pequeño relé [23].

Como se puede observar en el gráfico de la figura 57, con una pequeña corriente en la entrada, es posible controlar una corriente mucho mayor.

4.1.5 ENTRADAS DIGITALES

La placa NVFR posee un total de seis entradas digitales con el propósito de manejar las señales de control de bomba de color en manual y automático, control de ajuste lateral hacia la izquierda y hacia la derecha, y control de ajuste diagonal hacia diagonal inferior y superior.

Dado que el microcontrolador es incapaz de soportar un nivel de tensión de 24VDC en sus entradas, se ha hecho uso de un arreglo muy utilizado en estos casos, utilizando un dispositivo optocoplador que permite habilitar las entradas del microcontrolador de manera segura. En el circuito mostrado a continuación en la figura 59, se puede apreciar la configuración para todas las entradas digitales.

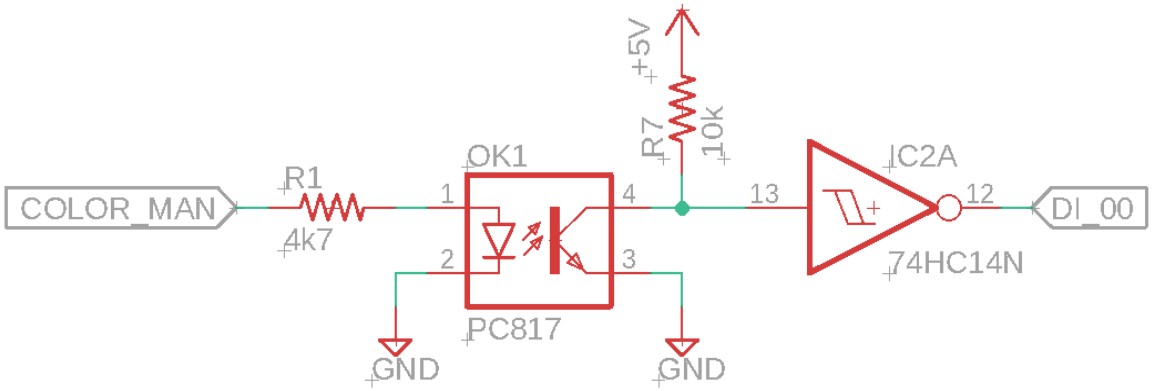


Figura 58: Configuración de entradas aisladas.

La figura anterior hace referencia al circuito correspondiente a la entrada de control manual de la bomba de color. La entrada COLOR_MAN posee un nivel de tensión de 24VDC, la cual es la tensión de control de la maquinaria, pero a través de la resistencia R1 se produce una caída de tensión que permite alimentar el emisor del optocoplador, cuando este es energizado, ilumina el receptor, el cual es un fototransistor, que al recibir la luz emitida alcanza su estado de saturación. La resistencia R7 que se encuentra en el lado del fototransistor conectado a su colector, se le conoce como resistencia *pull-up*, y su función, es la de evitar que mientras no haya energía en la entrada del optocoplador, se mantenga un valor estable en la entrada del microcontrolador.

Se puede observar que en el colector del fototransistor también se encuentra un inversor tipo *Schmitt trigger*, este último junto a la resistencia *pull-up* tienen la función de garantizar que la señal que llegue al terminal de entrada del microcontrolador sea una señal con un valor lógico estable de manera que se eviten los llamados rebotes, que no son mas que valores de tensión entre cero y cinco voltios que no están contemplados para ser niveles lógicos adecuados para el microcontrolador [26]. Dichos valores oscilan y “engañan” al microcontrolador produciendo resultados erróneos en el proceso.

Todas las configuraciones utilizadas para las entradas digitales de la placa NVFR se deben a que el ambiente donde se pondrá a prueba dicha placa, es un ambiente industrial, y por lo tanto se debe de evitar cualquier tipo de ruido que cause una mala interpretación de las señales de entradas [21] y por ende se traduzca no solo en fallas en los componentes del sistema, sino también en la calidad del producto final.

4.1.6 ENTRADAS ANALÓGICAS

Ademas de las seis entradas digitales que la placa NVFR posee, también se han puesto a disposición tres entradas analógicas. Dos de estas tres entradas tienen el fin de leer el valor de tensión de los potenciómetros para el ajuste lateral y diagonal mientras que una tercera entrada permanece de reserva para necesidades futuras. A diferencia de las entradas digitales la cuales tienen una etapa en la que se hace el aislamiento entre la tensión de alimentación y la tensión de entrada a los pines digitales del microcontrolador, las entradas analógicas simplemente se dirigen directamente a los pines analógicos del microcontrolador. Las dos entradas de ajuste se encuentran conectadas en los pines 25 y 26 del microcontrolador ATMEGA328P respectivamente como se puede observar en la siguiente figura.

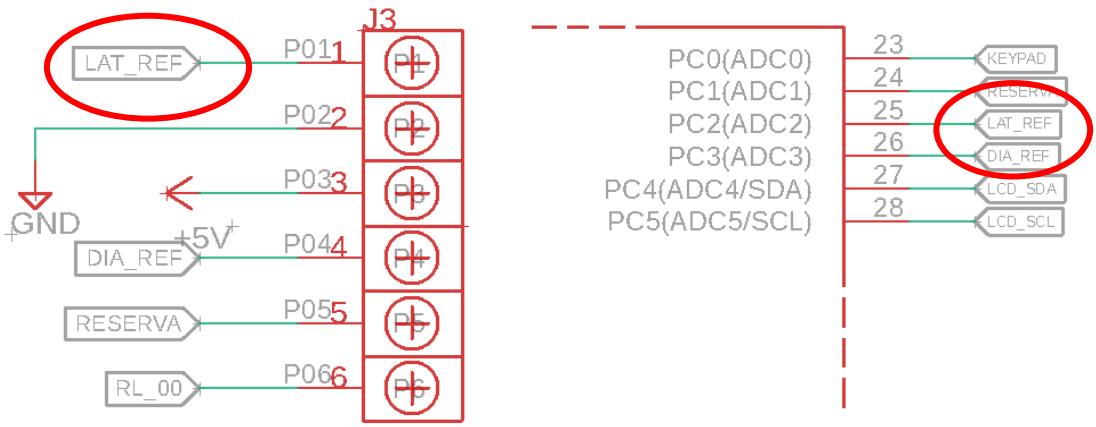


Figura 59: Entradas analógicas de la placa NVFR.

En la figura 59 se puede observar en el lado izquierdo el segmento correspondiente a los bornes de entradas en donde se realiza la conexión de la una de las entradas analógicas, y en la parte derecha, se tiene el detalle de los pines analógicos del microcontrolador. La tensión aplicada en los extremos de los potenciómetros es de 5VDC como se mencionó en secciones anteriores, ya que de acuerdo a la hoja de datos del microcontrolador ATMEGA328P este es el rango permitido para este tipo de entradas [13].

4.1.7 SALIDAS

Las salidas de la placa desarrollada son todas de tipo digital, siete en total. Cada una de ellas al ser activada suministra un nivel de tensión de 5VDC y con capacidad de controlar cargas hasta de 500mA, lo que significa que puede energizar pequeños relés cuyas bobinas estén dentro de estos parámetros, esto gracias al circuito integrado ULN2003A, el cual posee siete entradas las cuales se conectan a las salidas del microcontrolador ATMEGA328P, ampliando la capacidad de corriente que las salidas del microcontrolador pueden controlar por si mismas, las cuales esta por debajo de la capacidad del ULN2003A, siendo la corriente para cada pin de E/S 40mA en el ATMEGA328P [13] y de 500mA en el ULN2003A [23].

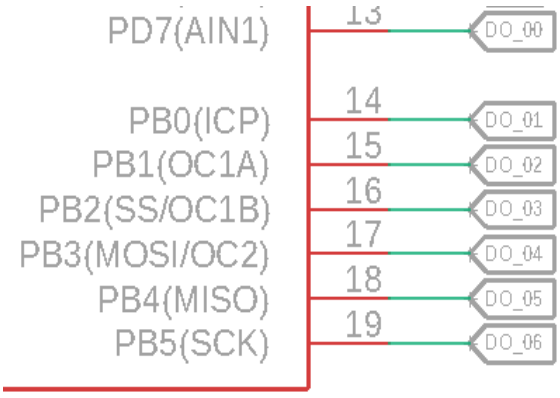


Figura 60: Salidas digitales de la placa NVFR.

De estas siete salidas disponibles, cuatro tienen el propósito del control de giro de los motores de ajuste lateral y diagonal, y una salida controla la bomba de color cuando la entrada manual o automática es

activada. Las dos salidas restantes sirven de reserva para necesidades futuras, lo cual se puede observar en la figura 61

Cada uno de los pines de salida del microcontrolador se conectan al puerto de entradas del ULN2003A, como se muestra en la figura 62, y cada vez que una de estas es activada, la señal viaja por el puerto de salida del ULN2003A permitiendo el manejo de los dispositivos externos a la placa NVFR que se encuentran conectados en sus terminales de salida.

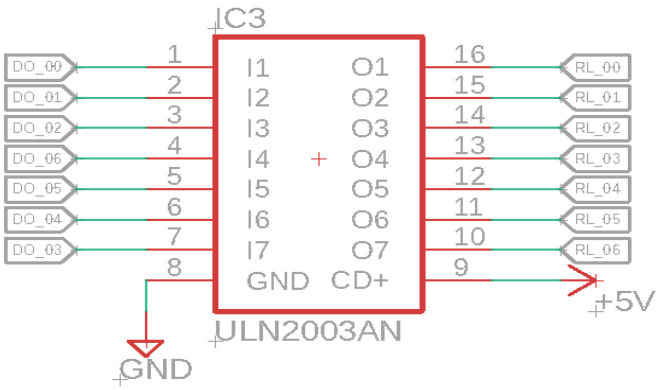


Figura 61: Salidas con IC ULN2003A.

4.2 NAVIGATOR FREE – SOFTWARE

Como se ha podido observar en las pruebas piloto, se inició el desarrollo del software encargado de controlar los procesos de ajuste lateral, ajuste diagonal y control de bomba de color. Dicho software al igual que la parte tangible del proyecto, es decir, el hardware llevan el nombre de *Navigator Free*, que se abrevia como "*NVFR*". Las primeras dos versiones del software (NVFR v1_0 y v1_1) como se puede encontrar en los resultados de las pruebas piloto en el capítulo tres de este trabajo de graduación, solamente fueron creadas para simulación en la placa de entrenamiento y realización de prototipos. Una vez que estas dos versiones fueron simuladas se procedió a hacer las adaptaciones necesarias en la maquinaria para poder pasar de las simulaciones a las pruebas reales que a partir de la versión NVFR v1.3, se comenzaron a poner en marcha en la maquinaria y se iniciaron las observaciones tanto del comportamiento como de los resultados.

Las versiones v1_3 y v1_4 mostraron muy buenos resultados, aunque no estaban exentas de algunos detalles como el tiempo de activación/desactivación del motor de ajuste lateral/diagonal, en los que se hicieron algunas pruebas extras para lograr igualar el comportamiento de los demás cabezales de estampación.

La versión del software más estable y la cual iguala en muchas maneras el comportamiento del sistema *Navigator Automata* es la NVFRv1_5, en la que se hizo uso por primera vez de la rutina *autoAjusteSeguro*, de la cual se obtuvieron excelentes resultados, ya que como se planteo en el análisis de ajuste lateral y diagonal, las mejoras propuestas para darle solución a los problemas con los que se inició dieron el resultado esperado.

La versión NVFRv1_5 a pesar de ser la mas estable, aun necesita la mejora y adición de pequeños detalles para poder llegar a ser la versión final que quedará en uso al final de la puesta en marcha de este proyecto.

Cada una de las versiones escritas del software NVFR han ido incluyendo nuevas características que le han permitido ampliar las opciones de control del sistema Navigator Automata como se pudo observar en la tabla 9, aun así la versión v1_5 no es suficiente para responder a las necesidades que se deben de suplir. Por lo tanto se ha desarrollado una nueva versión la cual será la encargada de controlar el hardware desarrollado, se trata de la versión NVFRv1_6. En esta versión se incluyen pequeñas mejoras en algunas de las rutinas, así como una rutina que permite la calibración del sistemas antes de su puesta en marcha.

4.2.1 RUTINA PARA CALIBRACIÓN.

La versión final del software que controla la placa desarrollada debe de ser capaz de permitir a los usuarios de la misma poder realizar los ajustes iniciales para un correcto funcionamiento. Por dicha razón se ha creado una rutina que interviene a la hora de realizar dicho ajustes, se trata de la rutina para calibración, que no es mas que permitir al usuario que instale la placa NVFR v3.1, mover libremente la posición lateral y diagonal del cabezal sin que la rutina *autoAjusteSeguro* intervenga hasta lograr las condiciones iniciales idóneas para una correcta operación.

Argumentos: Sin argumentos.

Valor de retorno: Sin valor de retorno.

El código fuente de la rutina mencionada se muestra a continuación:

```
void calibracion(void){

    int opcion = leerEntrada();

    monitor();

    if(opcion == 1 ){
        activarMotor(salidaLatIzq , TIEMPO_PULSO);
    }
        else if(opcion == 2 ){
        activarMotor(salidaLatDer , TIEMPO_PULSO);
    }
        else if(opcion == 3 ){
        activarMotor(salidaDiaInf , TIEMPO_PULSO);
    }
        else if(opcion == 4 ){
        activarMotor(salidaDiaSup , TIEMPO_PULSO);
    }
    apagarSalidas();

}
```

La rutina anterior es llamada en el programa cuando la entrada correspondiente de calibración es activada con un cero lógico. Cuando es activada esta función es necesario seguir correctamente los pasos de calibración para lograr resultados satisfactorios.

4.3 MÉTODO DE AJUSTE INICIAL Y CALIBRACIÓN.

En el sistema *Navigator Automata* se observó que al momento del fallo en la placa electrónica de control de motores AUTOMATA 70036700 220L3670 Rev 0.1 el cabezal queda inmóvil en la posición en la que se le practicó el último ajuste lateral y diagonal, por lo que ya no es funcional a menos que solo se utilice para el diseño en el que por última fue vez usado con éxito. Pero no es este el caso, ya que son muchos diseños los que pasan por este tipo de maquinaria todos los días.

Al hacer el cambio de la placa dañada por una nueva, se observó que la nueva placa simplemente se ajusta a los valores de referencia que en ese momento se encuentra leyendo de los potenciómetros de ajuste lateral y diagonal, lo que le permite continuar el trabajo sabiendo la posición del cilindro de estampación en todo momento. Este método sería eficaz si se tomara en consideración revisar la posición del potenciómetro y se tomara la lectura del valor de referencia que en ese momento le enviá a la entrada analógica correspondiente en la placa electrónica antes dejarla en trabajo continuó. Ya que como se mencionó, en el análisis del proceso y causa de fallas, el mal ajuste del potenciómetro en cuestión, es responsable de causar las fallas encontradas. Por lo tanto se ha planteado un método y una rutina dentro del programa con los cuales se harán los ajustes iniciales para la puesta en marcha de la placa NVFR el cual se describe a continuación.

1. Antes de energizar la placa por primera vez, se hacen las conexiones necesarias de potenciómetros de referencia, entradas para ajuste lateral y/o diagonal, bomba de color.
2. Remover el puente que une los pines 1 y 2 en JP2 y ubicar en los pines 2 y 3. Esta acción evitará que la función *autoAjusteSeguro* intervenga en caso de que los potenciómetros de referencia de ajuste lateral/diagonal se encuentren ubicados de manera incorrecta.
3. Se energiza la placa.
4. Con el selector de ajuste lateral o con el teclado en la placa, mover el cilindro de estampación hasta las marcas de posición de limite lateral izquierdo.
5. Verificar potenciómetro de referencia de ajuste lateral y con la ayuda de la pantalla LCD 2X16 en la placa NVFR verificar que la tensión sea de 0.5VDC, este valor se muestra en la pantalla como “VL”, sino fuera este el caso, hacer los ajustes necesarios en el potenciómetro hasta alcanzar el valor de 0.5V en la posición lateral mínima.
6. Con el selector de ajuste lateral o con el teclado en la placa, mover el cilindro de estampación hasta las marcas de posición de limite lateral derecho hasta alcanzar el valor de VL=3.5V.

7. Con el selector de ajuste diagonal o con el teclado en la placa,mover el cilindro de estampación hasta las marcas de posición de limite diagonal inferior.
8. Verificar potenciómetro de referencia de ajuste diagonal y con la ayuda de la pantalla LCD 2X16 en la placa NVFR verificar que la tensión sea de 0.5VDC, este valor se muestra en la pantalla como “VD”, sino fuera este el caso, hacer los ajustes necesarios en el potenciómetro hasta alcanzar el valor de 0.5V.
9. Con el selector de ajuste diagonal o con el teclado en la placa, mover el cilindro de estampación hasta las marcas de posición de limite diagonal superior hasta alcanzar el valor de VD=1.0V.
10. Remover el puente que une los pines 2 y 3 en JP2 y ubicarlos nuevamente en los pines 1 y 2. Con esta acción la función *autoAjusteSeguro* tomará el control en caso de una mala operación. De no realizar este paso se corre el riesgo de que el cabezal de estampación sufra los mismos daños mecánicos producidos por el fallo del sistema *Navigator Automata*.

4.4 COSTOS DE LA PLACA NVRF VRS NAVIGATOR AUTOMATA.

Durante la realización de este trabajo de graduación se investigó internamente en la empresa los costos del hardware original de la maquinaria, es decir, la placa base (AUTOMATA 70036600 220L3660 Rev. 0.3), la placa de control de motores (AUTOMATA 70036700 220L3670 Rev 0.1) y la placa de control de bomba de color (AUTOMATA 70036800 220L3670 Rev 0.1) y es evidente que los costos son muy altos sin contar que los tiempos de entrega también son muy prolongados y para agilizarlos es necesario una inversión más alta para lograr obtener los dispositivos solicitados en el menor tiempo posible.

De acuerdo al sistema de compras al exterior de país de la empresa se encontró que los precios para las placas electrónicas son los siguientes:

Nombre del artículo	Cantidad	Proveedores	Maquina	Precio	Moneda
DRUCKMODUL II 24VDC (AUTOMATA SCS) CARD MODULE FROM PRINTING HEADS STATION, CODIGO: 1A12	3	J.ZIMMER MASCHINENBAU ...	--	$1,012.00000	(€) EUR
DRUCKMODUL II (-1A12 AUTOMATA SCS) TERMINAL BOX PRINTING STATION D-DRUCKSTATION/HEADS, CODIGO: -1A12, PAGINA: 95	2	J.ZIMMER MASCHINENBAU ...	ESTAMPADORA-J...	$1,012.00000	(€) EUR
NIVEAU CONTROL 70036800 FOR DRUCKSTATIONS HEADS, PAG. 97, ROTA-TG116, CODIGO: 1A12-1	2		ESTAMPADORA-J...	$248.00000	(€) EUR
MOTOR CONTROL 70036700 FOR DRUCKSTATIONS HEADS, PAG. 98, CATALOGO: ROTA-TG116, CODIGO: 1A12.2	6		ESTAMPADORA-J...	$449.00000	(€) EUR
DRUCKMODUL II (-1A12 AUTOMATA SCS) TERMINAL BOX PRINTING STATION D-DRUCKSTATION/HEADS, CODIGO: -1A12, PAGINA: 95	2	J.ZIMMER MASCHINENBAU ...	ESTAMPADORA R...		
NIVEAU CONTROL 70036800 FOR DRUCKSTATIONS HEADS, PAG. 97, ROTA-TG116, CODIGO: 1A12-1	2		ESTAMPADORA R...		

Figura 62: Costos de las placas del fabricante CANNON AUTOMATA.

Como se puede observar los precios para cada elemento que conforma el sistema *Navigator Automata* son bastante elevados y a esto se le suma que cada precio esta en euros.

DESCRIPCIÓN	MODELO	PRECIO(€)	PRECIO($)
Placa base	AUTOMATA 70036600 220L3660 Rev. 0.3	1012.00	1188.31
Placa control de motores	AUTOMATA 70036700 220L3670 Rev 0.1	449.00	527.22
Placa control de bomba	AUTOMATA 70036800 220L3670 Rev 0.1	248.00	291.21

Tabla 10: Precios en dólares de los Estados Unidos de Norte America.

Debido a que la placa electrónica NVFR v3.1 solamente se esta utilizando en su mayoría para sustituir la placa de control de motores y la placa de control de bomba en la cual simplemente se hace la lectura del sensor de nivel de tinta y ya que la salida para activar la bomba se encuentra en la placa base la cual realiza muchas mas funciones, se asumirá que solo se están considerando los precios de las placas de control de motores y control de nivel de tinta, es decir:

Costo total = Costo de placa de control de motores + costo de placa de control de nivel de tinta

Costo total = $527.22 + $291.21

Costo total = $818.43

El costo de estas dos placas es de $818.43 dólares de los Estados Unidos, sin contar los gastos de embalaje.

Uno de los objetivos planteados al inicio de este trabajo de graduación es que la solución propuesta sea de bajo costo, es decir, que sea un 30% del costo total del hardware original.

En la tabla 11 se puede observar que se tiene un costo por placa de $1.66 esto es debido a que dichas placas se enviaron a elaborar en una empresa china llamada JLCPCB, para que tuvieran un acabado profesional , teniendo un costo total por cinco placas de $8.30. Es importante mencionar que los costos de envío, son incluso mucho mas altos que los costos de fabricación de la PCB en si, ya que estos ascienden a un total de $94.70. El costo total de las cinco placas mas los costos de envío fue de $103.00, aun con ese precio tanto el desarrollo como la fabricación de las placas esta muy por debajo del costo del hardware original de la máquina.

Se puede decir entonces que se dio cumplimiento al objetivo de que la placa sea de bajo costo ya que incluso el valor total de la placa esta muy por debajo del 30% que se tenía como objetivo.

CANTIDAD	ELEMENTO	PRECIO UNITARIO($)	SUBTOTAL($)
5	Resistencia 1kΩ	0.23	1.15
6	Resistencia 4.7kΩ	0.23	1.38
7	Resistencia 10kΩ	0.23	1.67
1	Diodo	0.20	0.20
1	Cristal 16MHz	0.79	0.79
2	Capacitor 22pF	0.16	0.32
1	Portafusible 4x20mm	0.80	0.80
1	Fusible 4x20mm, 1A	0.15	0.15
6	Pulsadores NO	0.50	3.00
1	IC 74LS14	0.53	0.53
1	IC ULN2003A	0.70	0.70
1	µC ATMEGA328PU	5.00	5.00
1	LCD Display 2x16 I2C	8.00	8.00
1	Pinheader 40 pines	1.00	1.00
6	Terminal block 3 posic.	0.40	2.40
1	PCB manufacturado JLCPCB	1.66	1.66
1	Costo de envío PCB	18.94	18.94
1	Placa de relés de 8 canales	10.00	10.00
1	Arduino UNO	10.00	10.00
TOTAL			67.69

Tabla 11: Precios de elementos de placa NVFR v3.1

Teniendo un total de $67.69 en donde se incluyen los gastos de desarrollo, es decir, la placa Arduino UNO para las pruebas, se puede hacer el siguiente cálculo.

$$\frac{Costo\ hardware\ original}{100\,\%} = \frac{Costo\ NVFR}{\%Costo\ NVFR}$$

$$\%Costo\ NVFR = \frac{Costo\ NVFR}{Costo\ hardware\ orignal}\,100\,\%$$

$$\%Costo\ NVFR = \frac{\$\,67.69}{\$\,818.43}\,100\,\%$$

$$\%Costo\ NVFR = 8.27\,\%$$

4.5 RESULTADOS DE LA PLACA NVFR.

Una vez ensamblada la placa NVFR v3.1 con todos sus componentes fue necesario realizar las pruebas fuera de la máquina para comprobar el correcto funcionamiento y detectarlos a tiempo. Como se puede observar en la figura 63, se ha montado la placa con su respectivo módulo de relés para las salidas en una base acrílica, ya que el conjunto posee un sujetador para el montaje en riel DIN.

También se puede ver un pequeño teclado de seis teclas, las cuales tienen el objetivo de realizar las funciones de ajuste lateral y diagonal, así como el reset del controlador. El controlador también posee una pequeña pantalla LCD 2X16 como dispositivo de salida de datos para que el personal de mantenimiento pueda ver en tiempo real los valores de referencia que se encuentra leyendo el controlador de los potenciómetros ubicados en las posiciones lateral y diagonal.

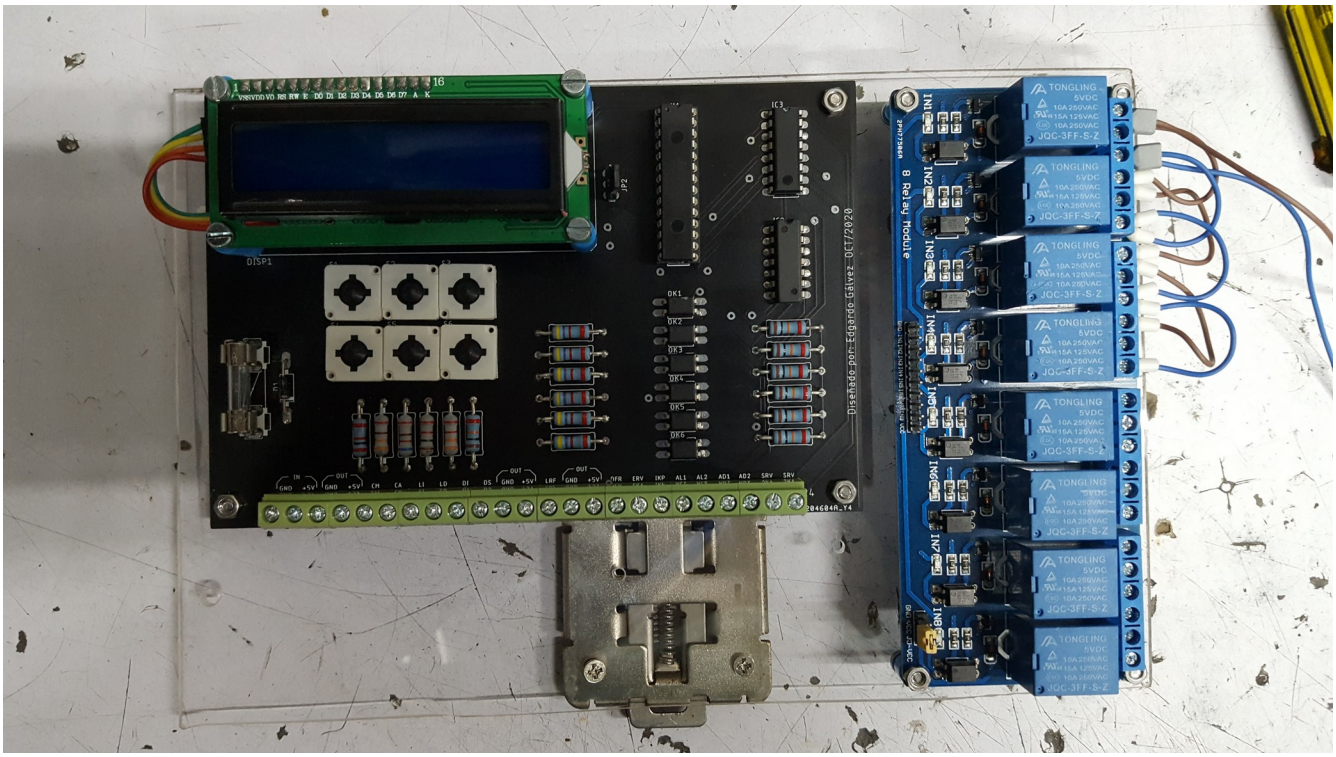


Figure 63: Ensamble final de la placa NVFR v3.1 con su módulo de salidas a relé.

Se simularon distintos escenarios con la placa fuera de la máquina, se produjeron fallas intencionales a través de las entradas digitales, se conectó un potenciómetro en las entradas analógicas de manera que simularan posiciones erróneas del cabezal. También se pusieron a prueba las funciones de auto ajuste seguro así como también la función de calibración del sistema para trabajar en conjunto con el controlador.

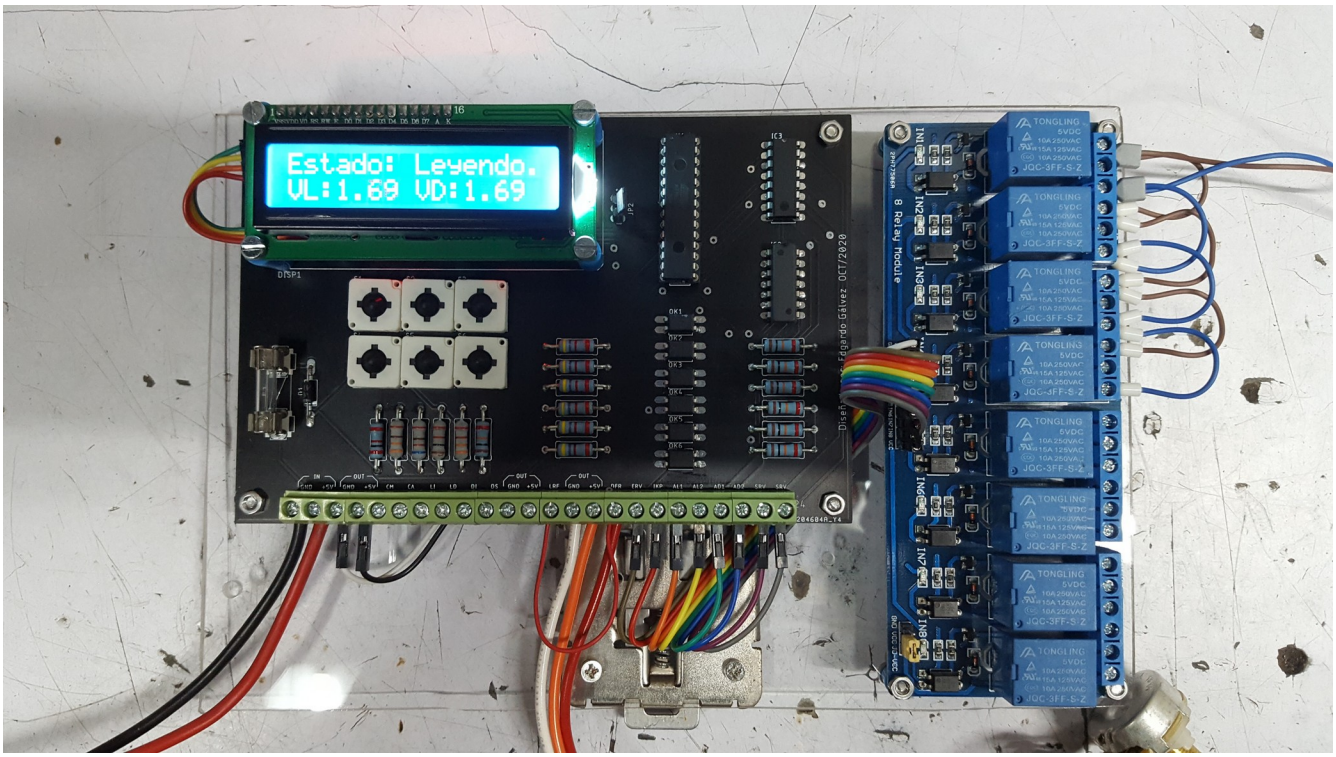


Figure 64: Pruebas en el hardware NVFR v3.1 antes de ser montado en la maquinaria.

Todas las pruebas fueron superadas satisfactoriamente por lo que se procedió al montaje de la placa en el sitio donde permanecerá operando y como último paso para poner en marcha la versión final del hardware NVFR v3.1, fue necesario remover el ensamble realizado para las pruebas piloto, el cual consta de una caja plástica en la cual se habían acomodado los elementos antes mencionados para este tipo de pruebas, y se procedió a montar en el panel eléctrico de la máquina como se muestra en la figura 65.

Al ser montada la placa en el panel eléctrico de la máquina se procedió a realizar las conexiones necesarias como se muestra en el diagrama eléctrico mostrado en los anexos de este trabajo de graduación, para luego seguir los pasos detallados en el apartado de la calibración y puesta en marcha.

El hardware NVFR v3.1 se desempeña de manera satisfactoria en la maquina estampadora ROTA-TG/116, exactamente en el cabezal número 1 con la posibilidad de instalar nuevas placas NVFR en los cabezales que así lo requieran al llegar a su vida útil el hardware propietario.

RECOMENDACIONES

Para la placa electrónica desarrollada y debido a los problemas encontrados los cuales son causas del mal funcionamiento del sistema se sugieren las siguientes recomendaciones:

- Antes de la puesta en marcha es necesario seguir los pasos de calibración los cuales han sido detallados claramente para lograr que el sistema funcione dentro de las condiciones normales de operación sin forzar las partes mecánicas del sistema.
- Por ningún motivo trabajar el sistema en modo de calibración en tiempo de producción, el modo de calibración es exclusivo para los ajustes iniciales del cabezal. Siempre mantener el puente del pin header JP2 en los pines 1 y 2, de hacer caso omiso se corre el riesgo de dañar seriamente las partes mecánicas del sistema, así como motores e incluso los relés de control.
- La placa electrónica NVFR v3.1 tiene la capacidad de manejar entre 12 y 24VDC en sus entradas, abstenerse de superar estos límites ya que esto podrían resultar en la destrucción de alguno de sus componentes.
- Dar mantenimiento al sistema al menos una vez cada tres meses, el cual debe de incluir revisión del ajuste de los potenciómetros para ajuste lateral y diagonal y revisar que el tornillo que se encuentra acoplado al motor para el movimiento de los cabezales este correctamente lubricado.
- Cada vez que se haga un ajuste o reemplazo de alguno de los potenciómetros, realizar el respectivo procedimiento de calibración.
- En próximas versiones del hardware desarrollado, sería de mucha utilidad agregar un puerto para programación del microcontrolador sin tener necesidad de removerlo de la placa.

CONCLUSIONES

- El uso del software y del hardware de libre distribución para el desarrollo de una solución efectiva para un problema real de una maquinaria textil de la industria Salvadoreña fue una excelente opción ya que permitió reducir los costos y se evita caer en consecuencias legales debido al uso de software privativo sin las licencias necesarias, lo cual incurre en costos adicionales. Sin embargo se debe de tomar en cuenta que muchos de los programas de software especializados para desarrollo de aplicaciones de control industrial son de naturaleza privativa y por ende para hacer uso completo de todas sus funcionalidades es necesario hacer las respectivas inversiones económicas.
- Tanto en las pruebas piloto como en el uso continuo de la placa desarrollada se obtuvieron excelentes resultados, ademas de que el software creado para dicha placa permite que aun bajo condiciones de mala operación se corrijan las situaciones que anteriormente producían fallas que resultaban en paros indefinidos de maquinaria. Por lo tanto se puede concluir que la solución desarrollada es mucho mas robusta que la equivalente comercial.
- Los costos de fabricación, sumados a los costos de desarrollo de la solución basada en software y hardware de libre distribución son mucho más bajos que la opción comercial, por lo que es evidente que se pueden desarrollar soluciones de bajo costo con resultados de la misma calidad que los dispositivos comerciales. Cabe destacar que aunque las placas electrónicas se manufacturaron fuera del país, elevando el coste de estas, se obtuvo una mejor calidad que eligiendo la opción del acabado artesanal, y sin embargo aun con los costos de fabricación el precio de una placa terminada es mucho mas bajo que el precio del hardware comercial.
- Es de vital importancia reconocer que no todas las aplicaciones de la industria pueden sustituirse por soluciones con poco tiempo de desarrollo. Debe de evaluarse el tipo de problema para darle una solución como la desarrollada en este trabajo de graduación o recurrir a una solución de un fabricante comercial, las cuales cuentan con equipos especializados de desarrollo en los cuales invierten grandes presupuestos para investigación y de esa manera obtener un producto de alta calidad.

REFERENCIAS BIBLIOGRÁFICAS.

[1] Stallman, R. (2019). *Free Software, philosophy/History*. What is free software? https://www.gnu.org/philosophy/free-sw.en.html#History

[2] Stallman, R. (2015). *Free Sofware philosophy/Selling*. Selling Free Software. https://www.gnu.org/philosophy/selling.en.html

[3] Stallman, R. (2019). *Log of /www/philosophy/free-sw.html*. http://web.cvs.savannah.gnu.org/viewvc/www/www/philosophy/free-sw.html?view=log

[4] Stallman, R. (2019). *Philosophy/Free hardware designs*. Free Software and Free Hardware Designs. https://www.gnu.org/philosophy/free-hardware-designs.en.html

[5] Stallman, R. (2019). *Log of /www/philosophy/free-hardware-designs.html*. http://web.cvs.savannah.gnu.org/viewvc/www/www/philosophy/free-hardware-designs.html?view=log

[6] *Various Licenses and Comments about Them*. (2020). https://www.gnu.org/licenses/license-list.html#GPLv2

[7] Stallman, R. (1989). *GNU General Public License, version 1*. https://www.gnu.org/licenses/old-licenses/gpl-1.0.html

[8] Stallman, R. (1991). *GNU General Public License, version 2*. https://www.gnu.org/licenses/old-licenses/gpl-2.0.html

[9] Stallman, R. (2007). *GNU GENERAL PUBLIC LICENSE, Version 3*. https://www.gnu.org/licenses/gpl-3.0.html

[10] Repository, O. hardware. (n.d.). *Licences*. https://ohwr.org/licences

[11] Stallman, R. (2019). *Philosophy/Free hardware designs*. Licenses and Copyright for Free Hardware Designs. https://www.gnu.org/philosophy/free-hardware-designs.en.html

[12] Mazidi, M. A., Naimi, S., & Naimi, S. (2011). The AVR microcontroller and embedded systems: Using Assembly and C. Prentice Hall.

[13] Corporation, A. (2015). *ATmega328P | 8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash*. http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

[14] cc, A. (2020). *Arduino Uno Rev3*. https://store.arduino.cc/usa/arduino-uno-rev3

[15] Wiring (development platform). (2020). In *Wikipedia*. https://en.wikipedia.org/w/index.php?title=Wiring_(development_platform)&oldid=967088213

[16] Stallman, R. (2019). *Linux and the GNU System*. https://www.gnu.org/gnu/linux-and-gnu.en.html

[17] Contributors, openSuSE. (2019). *OpenSUSE Leap 15.2*. https://software.opensuse.org/distributions/leap

[18] Austria, Z. (2019). *ZIMMER AUSTRIA*. https://www.zimmer-austria.com/en/content/company-0

[19] *ROTASCREEN.TG/TU |ROTARY SCREEN PRINTING*. (2017). J. Zimmer Maschinenbau GmbH Screen & Coating Systems. https://www.zimmer-klagenfurt.com/en/content/downloads-0

[20] SHARP. (2003). *PC817X Series DIP 4pin General Purpose Photocoupler*. https://www.farnell.com/datasheets/73758.pdf

[21] Gajski, D. D., & García Puntonet, C. (2004). *Principios de diseño digital*. Prentice-Hall.

[22] Instrument, T. (2016). *SNx414 and SNx4LS14 Hex Schmitt-Trigger Inverters*. https://www.ti.com/lit/ds/symlink/sn74ls14.pdf

[23] Instrument, T. (2019). *ULN200x, ULQ200x High-Voltage, High-CurrentDarlington Transistor Arrays*. https://www.ti.com/lit/ds/symlink/uln2003a.pdf

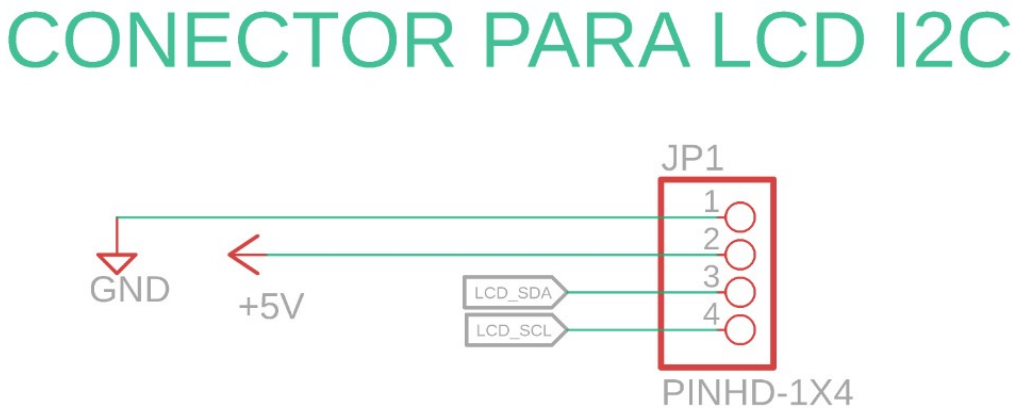
[24] Paul Horowitz. (2015). *The art of electronics* (Third edition). Cambridge University Press.

[25] Instrument, T. (2003). *MA7800 SERIES POSITIVE-VOLTAGE REGULATORS*. https://www.sparkfun.com/datasheets/Components/LM7805.pdf

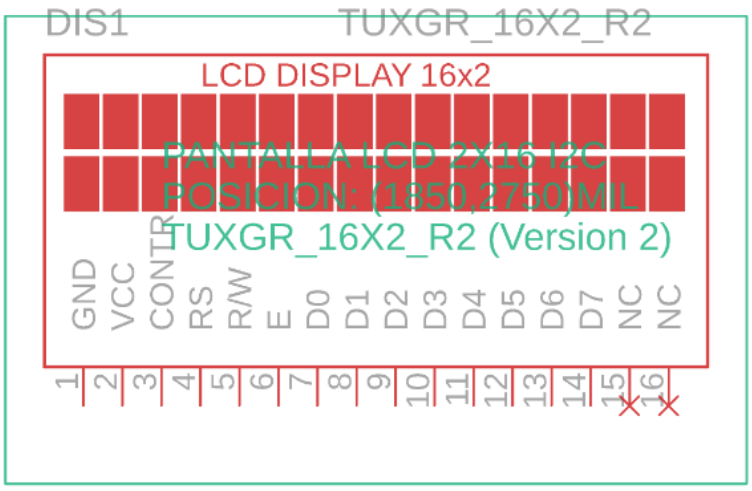
[26] Nelson, V. (1996). *Análisis y diseño de circuitos lógicos digitales* (1a ed.). Prentice Hall.

ANEXOS

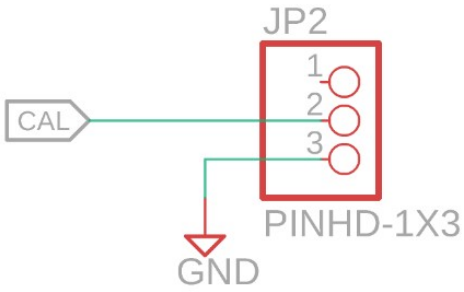
CIRCUITO ESQUEMÁTICO



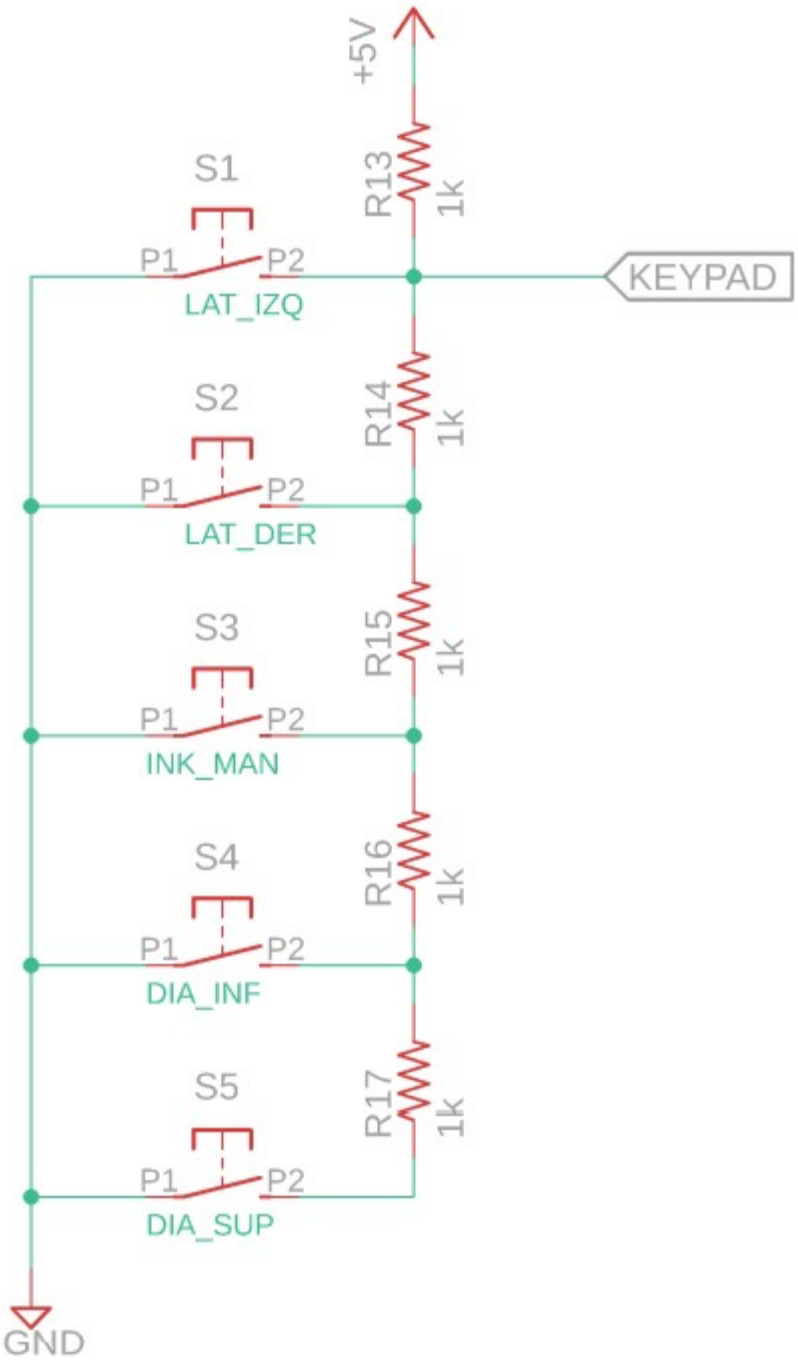
PANTALLA LCD 2X16 I2C



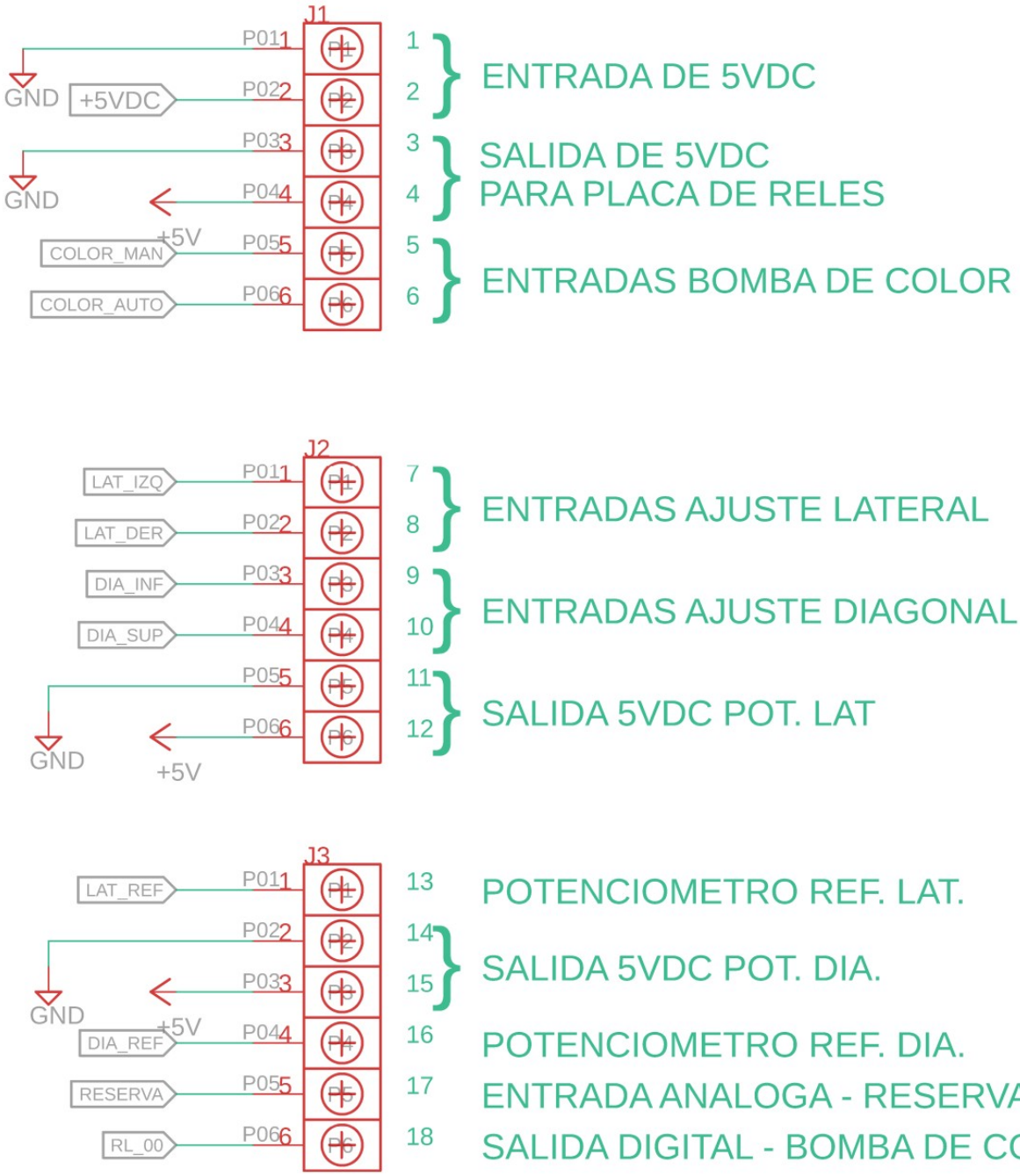
PUENTE PARA AJUSTE INICIAL.



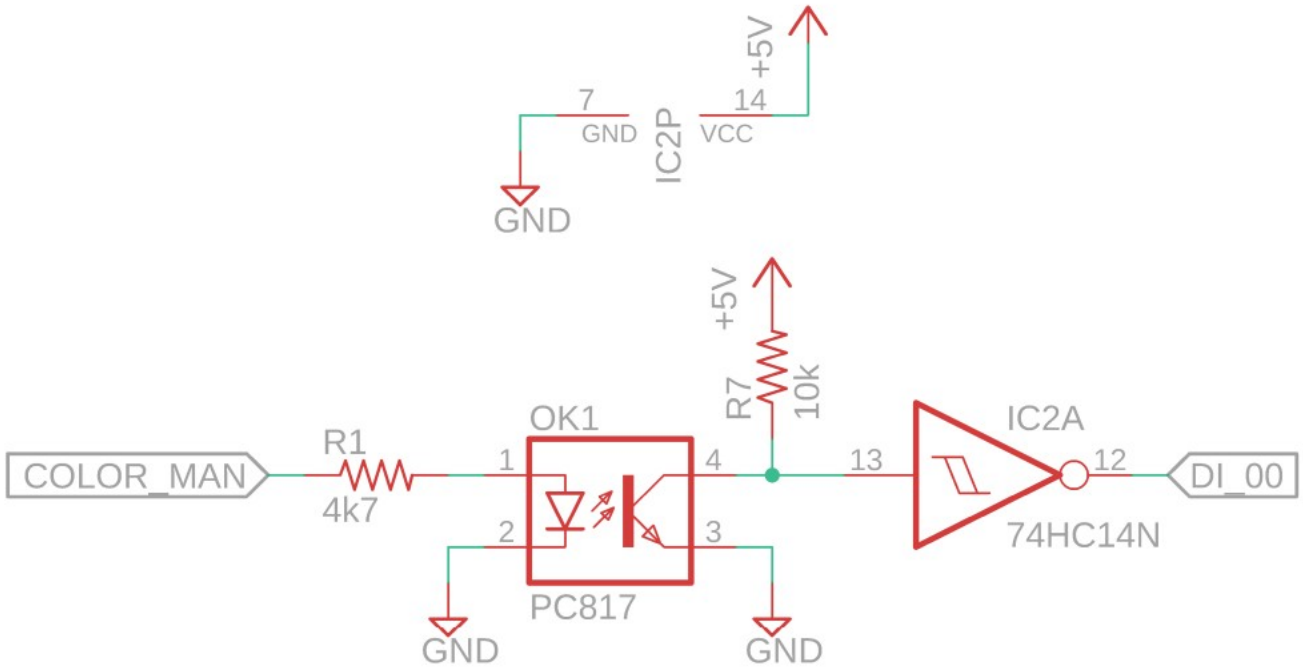
TECLADO

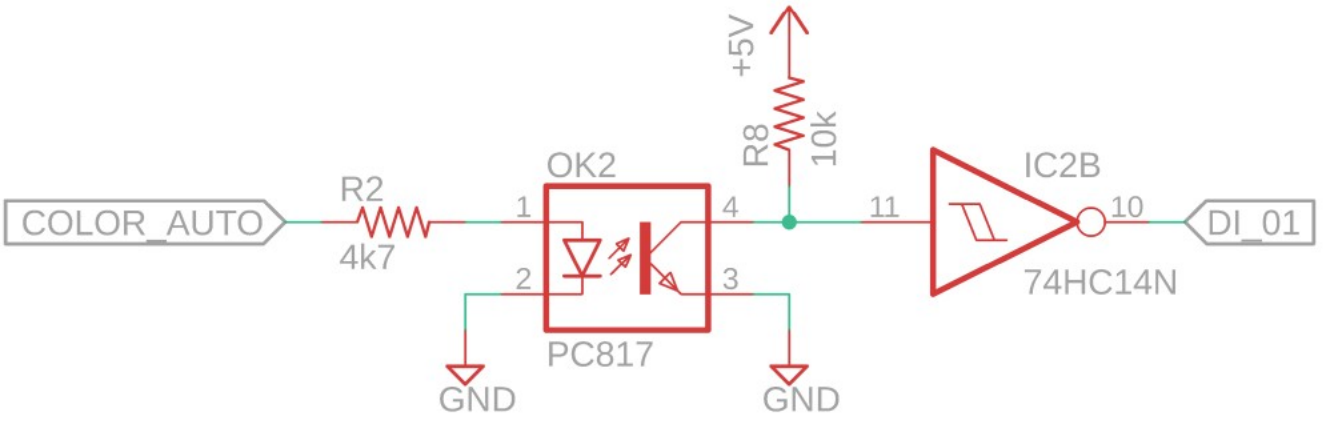

+5V
R13
1k
S1
P1
P2
LAT_IZQ
KEYPAD
R14
1k
S2
P1
P2
LAT_DER
R15
1k
S3
P1
P2
INK_MAN
R16
1k
S4
P1
P2
DIA_INF
R17
1k
S5
P1
P2
DIA_SUP
GND

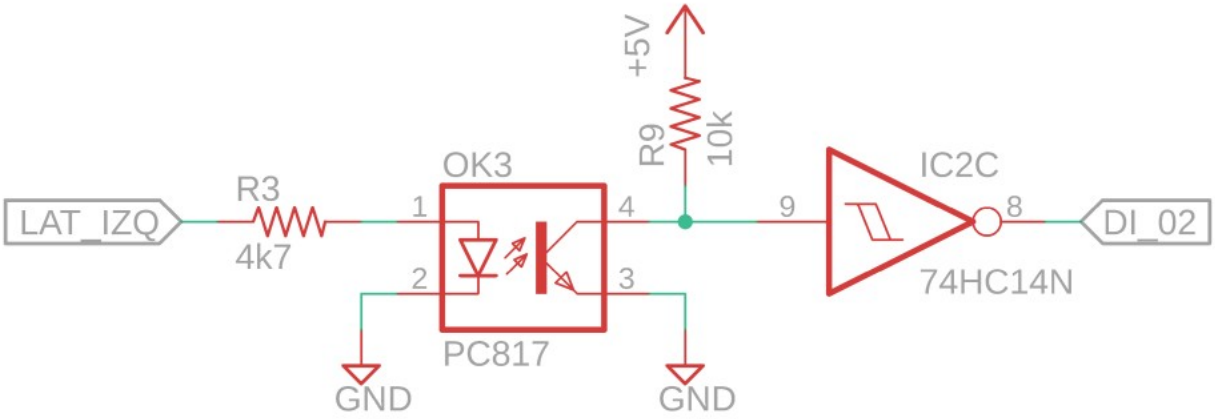
BORNERAS DE ENTRADA/SALIDA

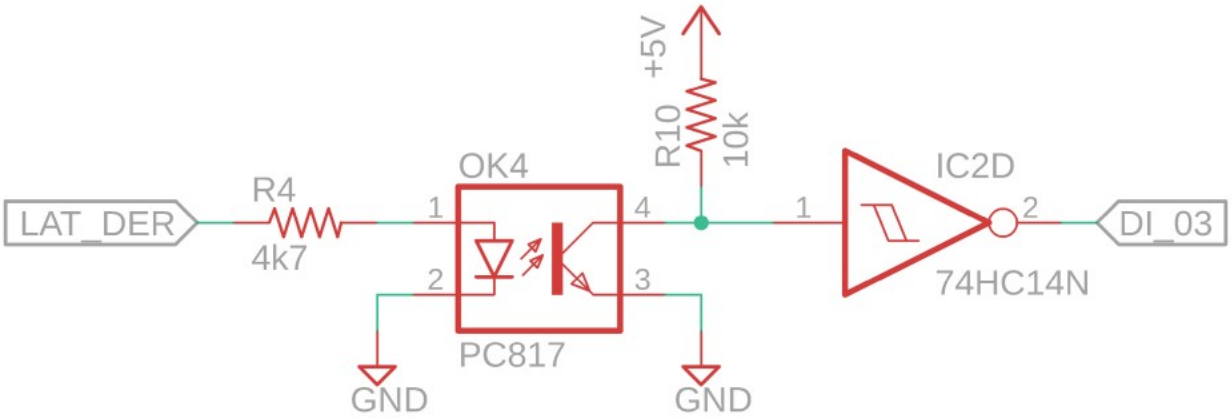


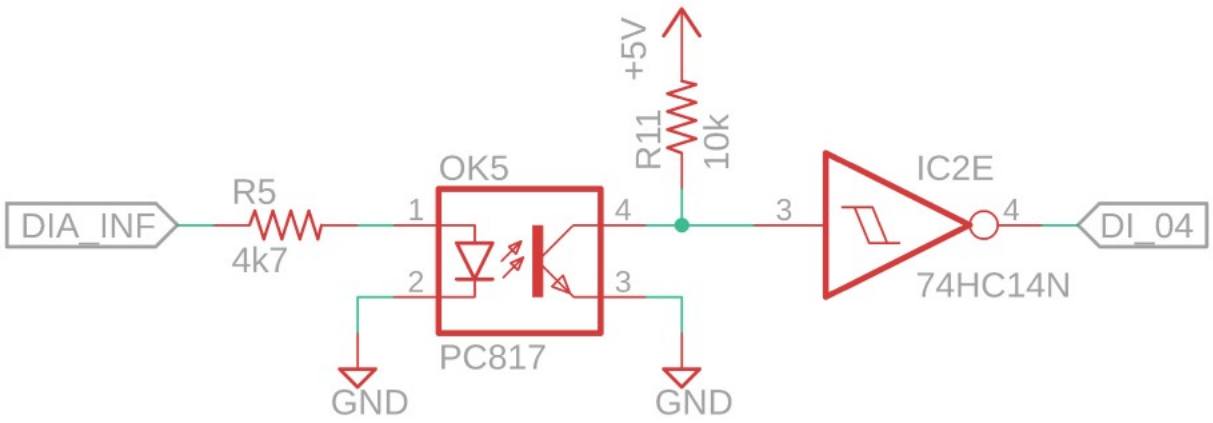
AISLAMIENTO DE ENTRADAS

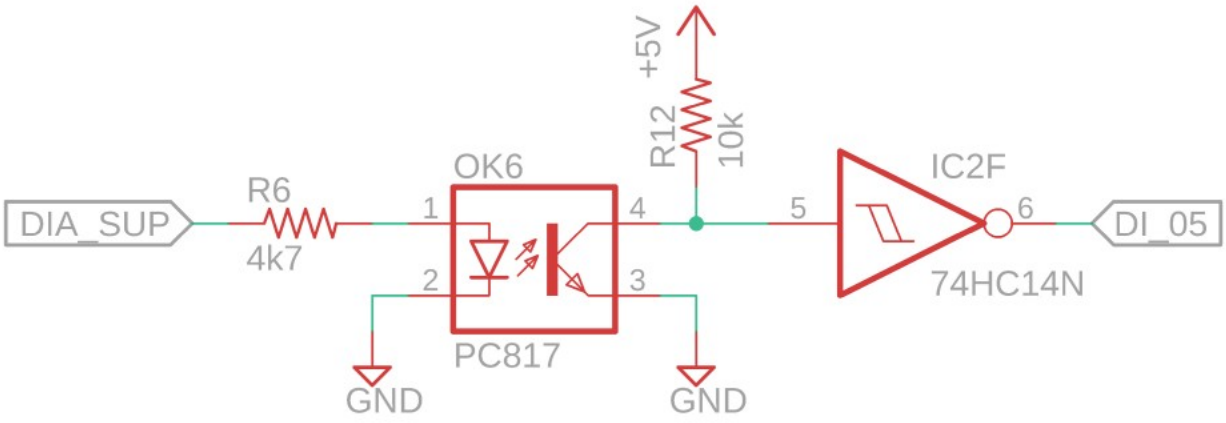



+5V
R8
10k
OK2
R2
COLOR_AUTO
4k7
1
2
4
3
11
IC2B
10
DI_01
74HC14N
PC817
GND
GND

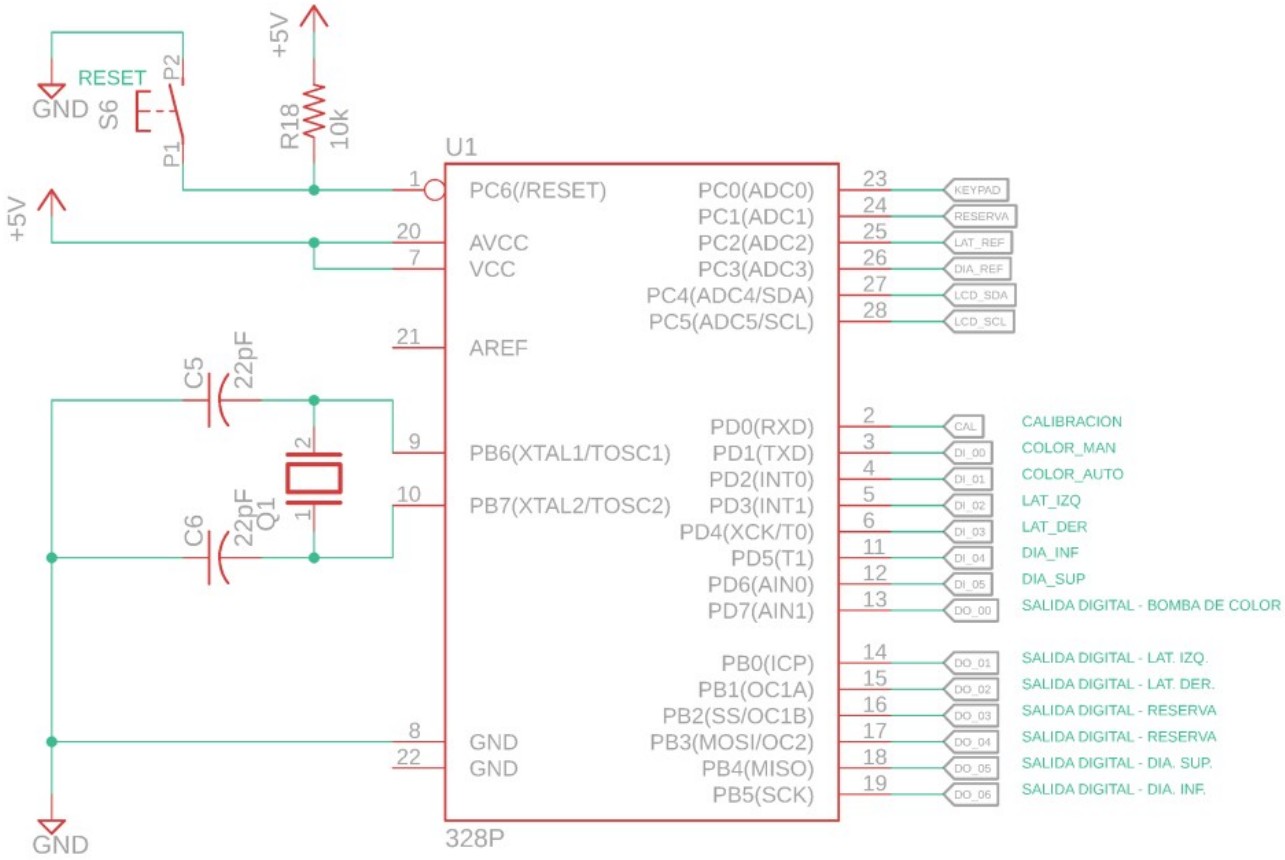

+5V
R9
10k
OK3
R3
LAT_IZQ
4k7
1
2
4
3
9
IC2C
8
DI_02
74HC14N
PC817
GND
GND


+5V
R10
10k
OK4
R4
LAT_DER
4k7
1
2
4
3
1
IC2D
2
DI_03
74HC14N
PC817
GND
GND

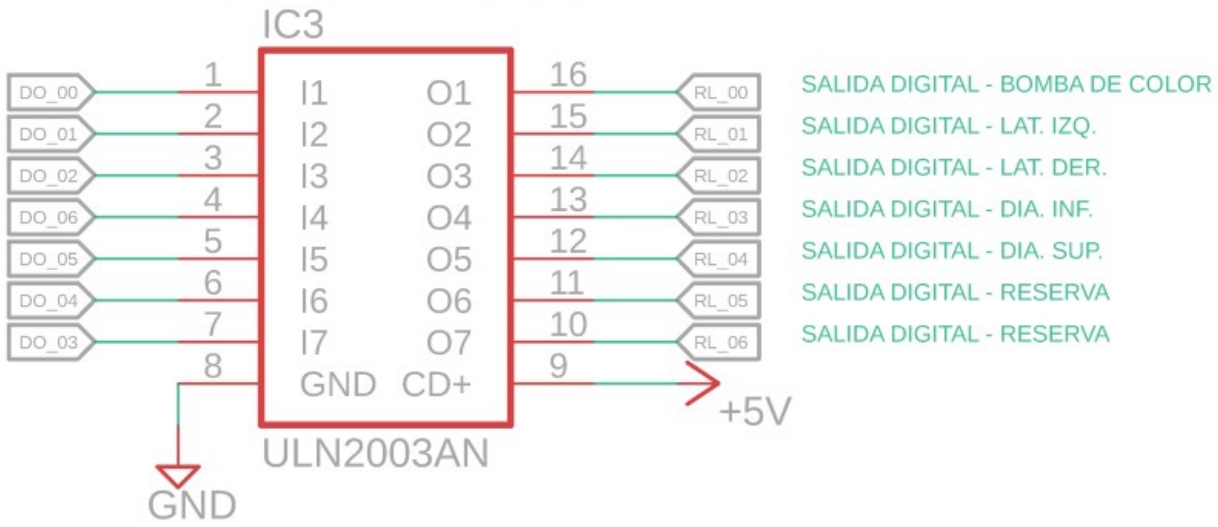

+5V
R11
10k
OK5
R5
DIA_INF
4k7
1
2
4
3
IC2E
4
DI_04
74HC14N
PC817
GND
GND


+5V
R12
10k
OK6
R6
DIA_SUP
4k7
1
2
4
3
5
IC2F
6
DI_05
74HC14N
PC817
GND
GND

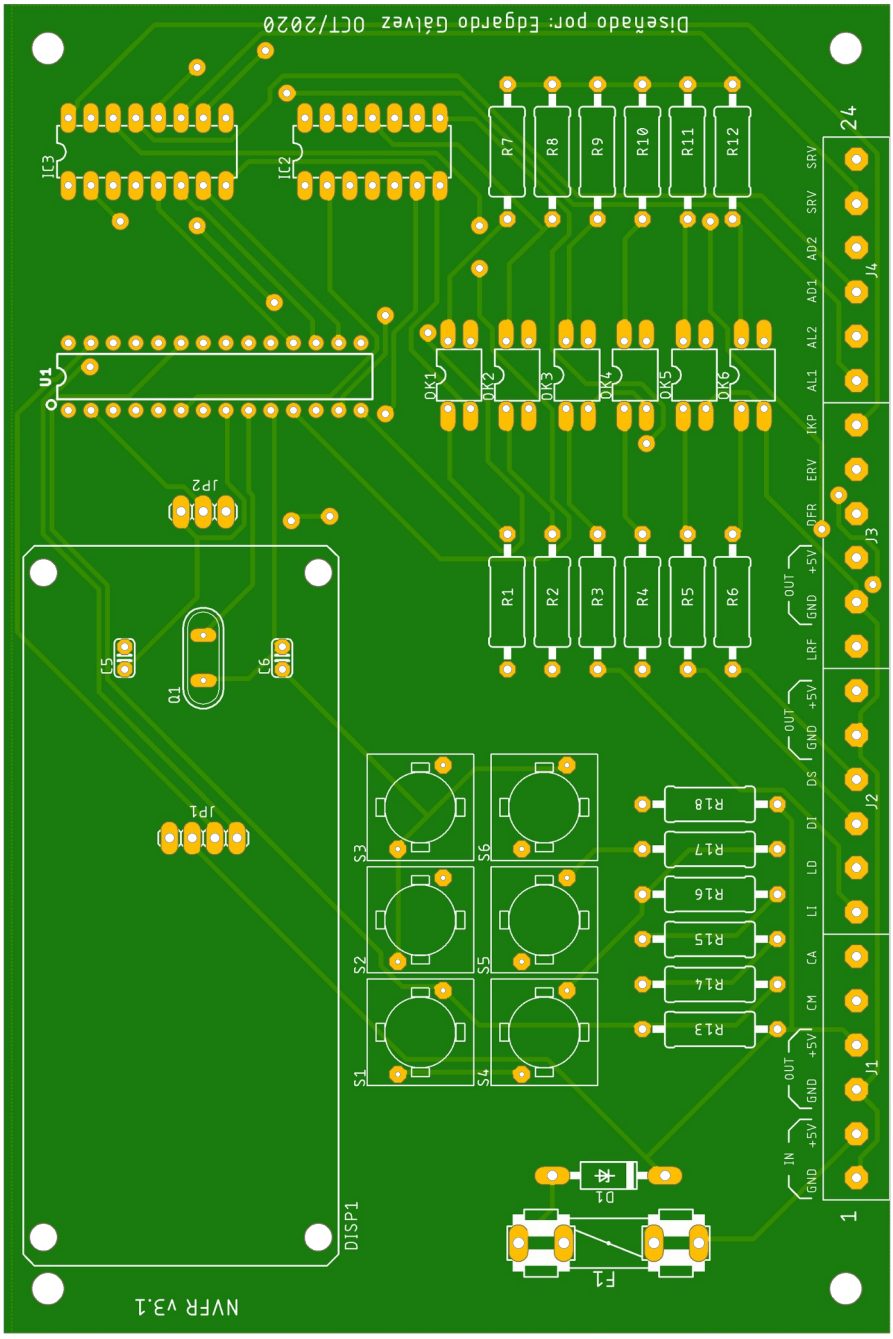
uC ATMEGA 328P-PU



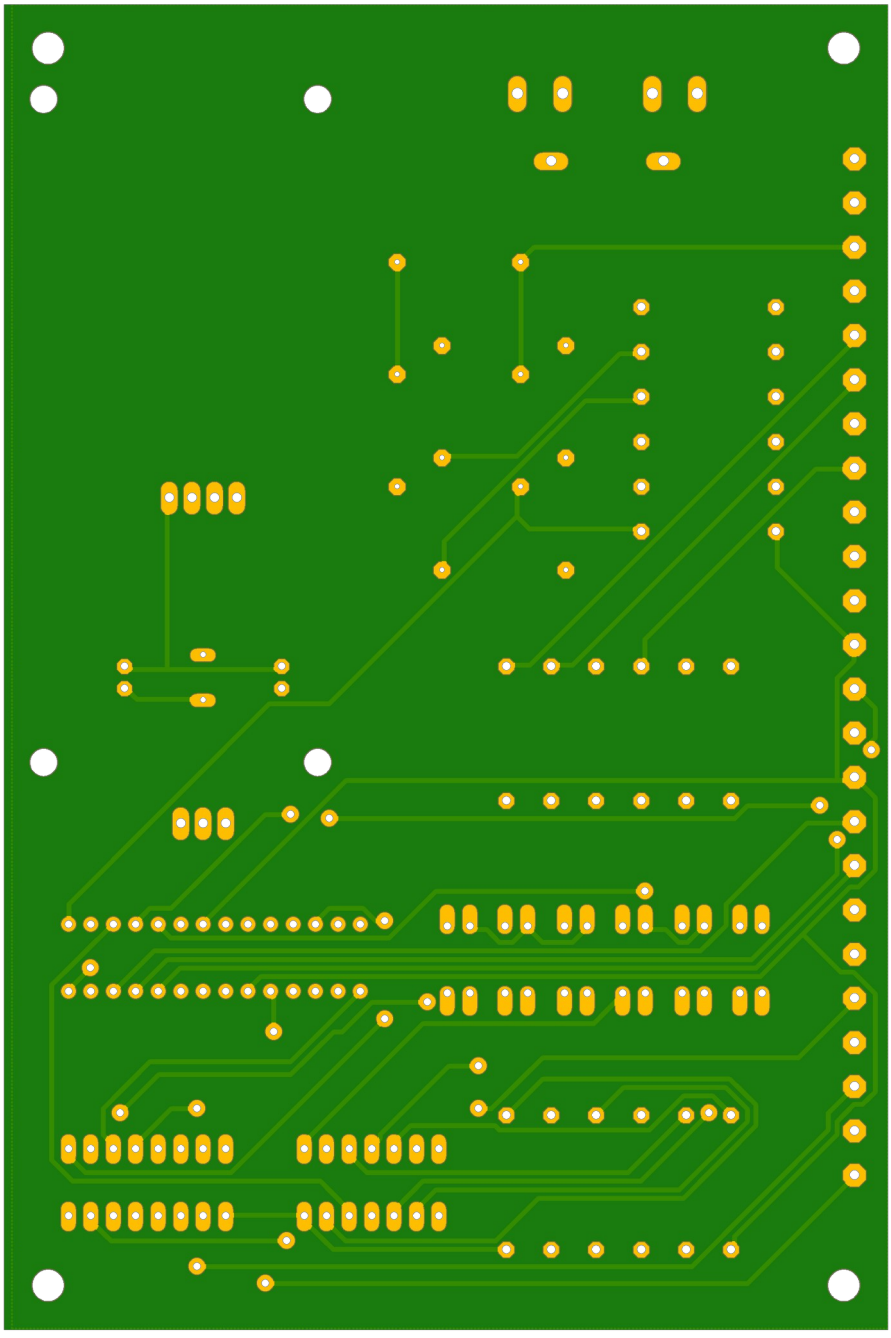
SALIDAS



PCB CAPA SUPERIOR



PCB CAPA INFERIOR



PCB MANUFACTURADA SIN ELEMENTOS MONTADOS.

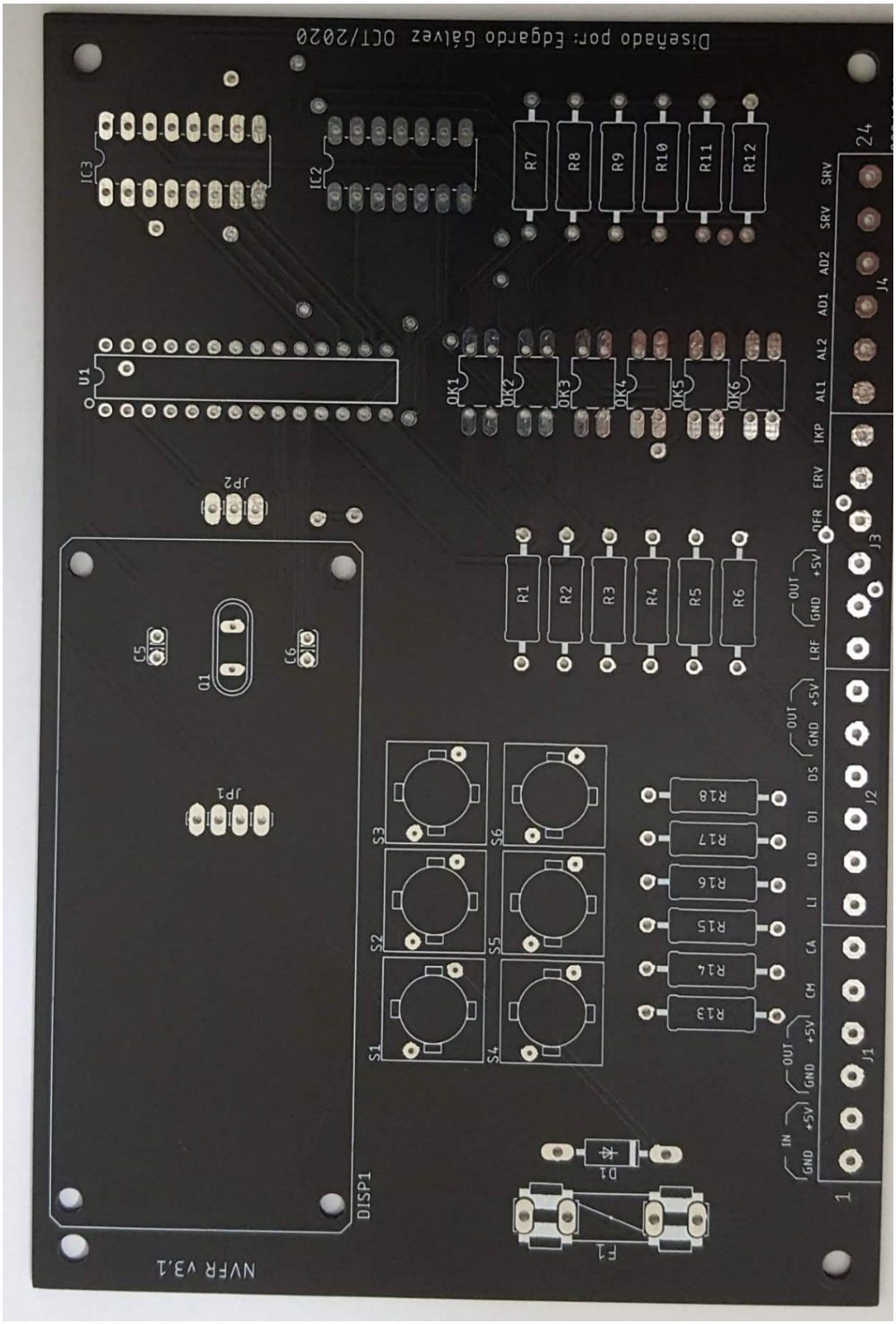
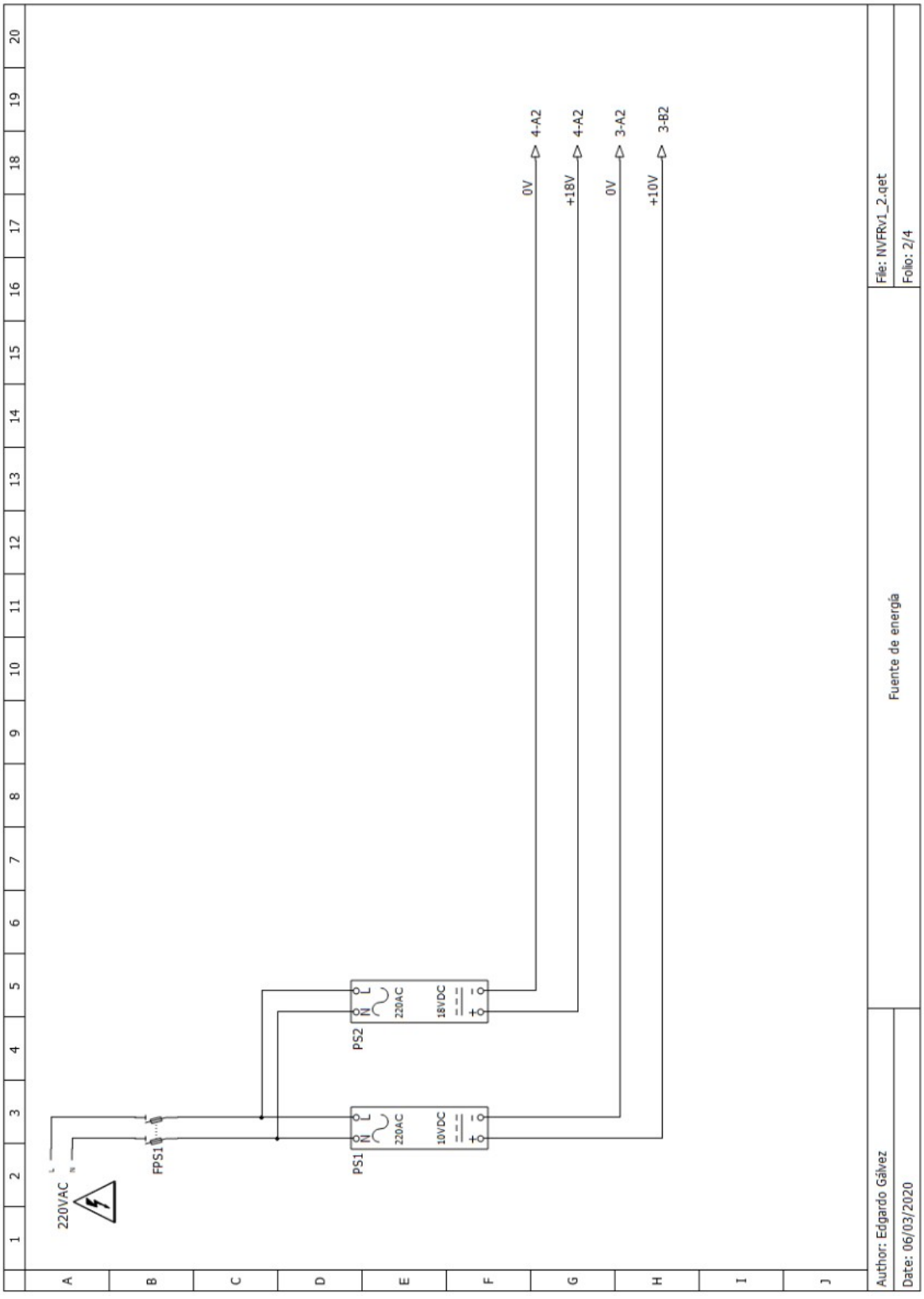
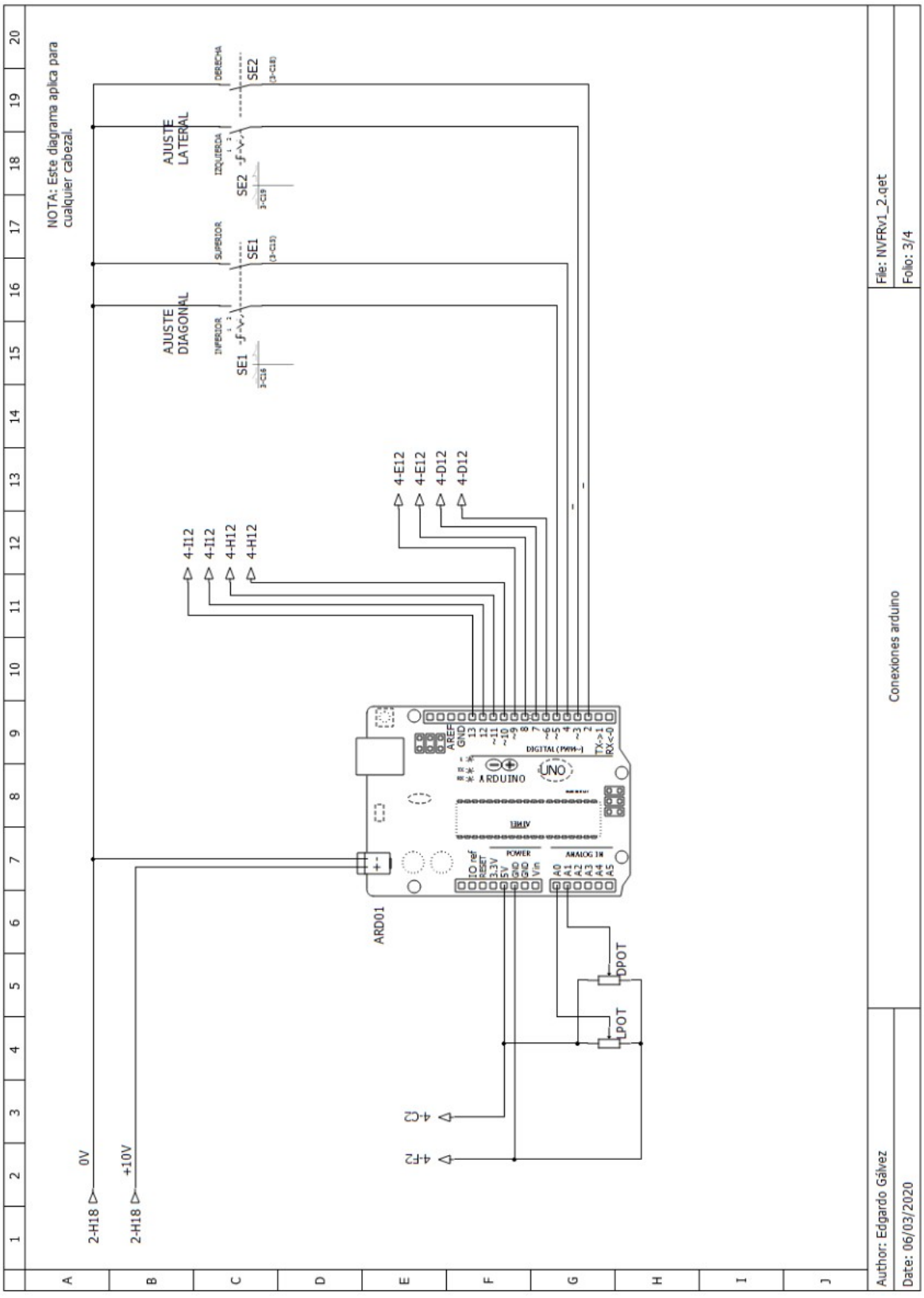
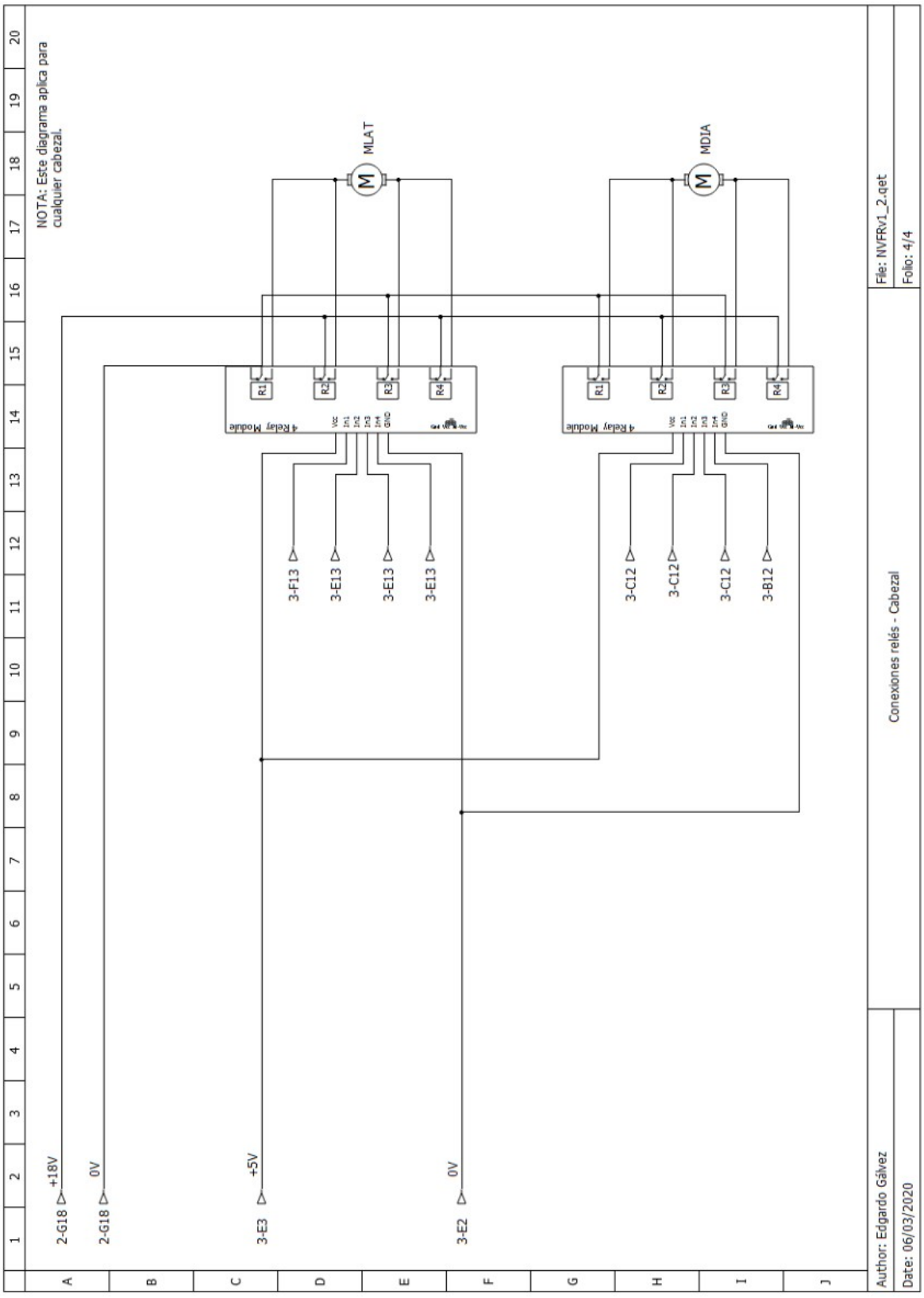


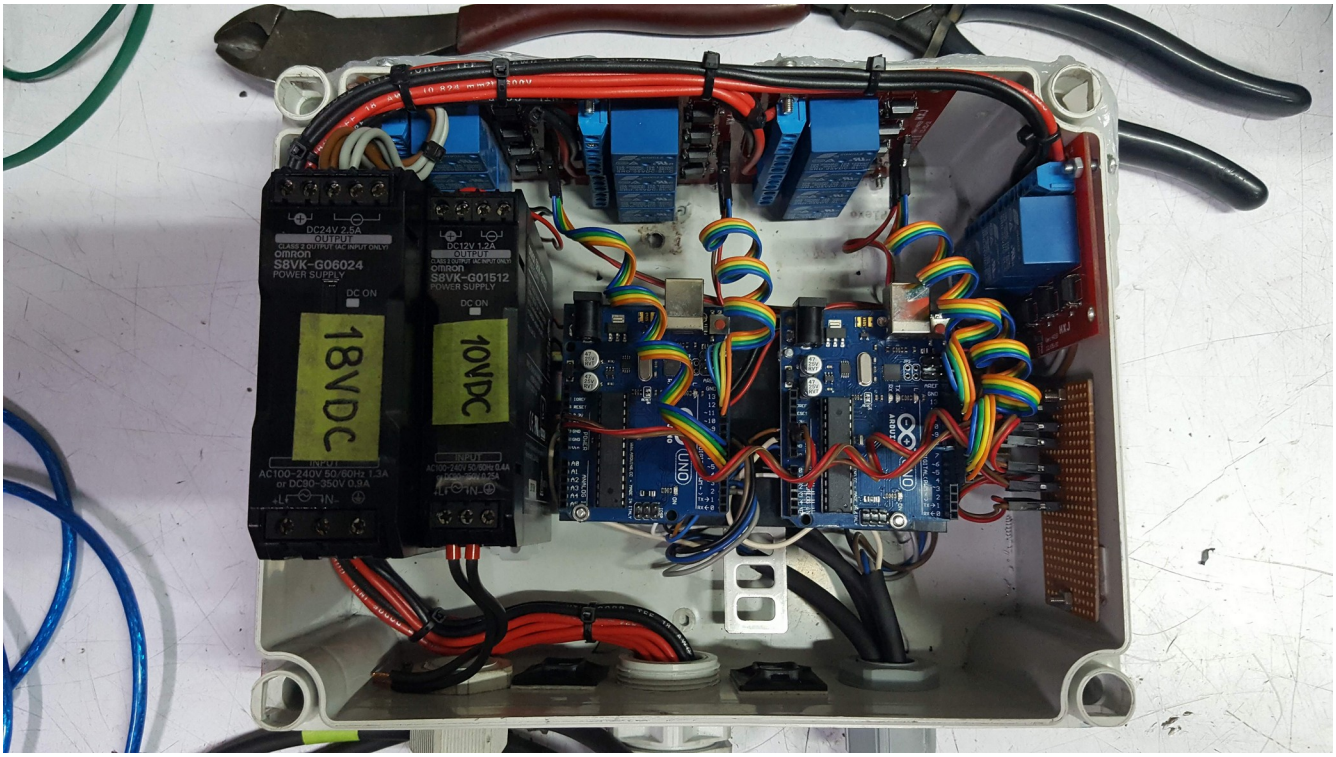
DIAGRAMA ELÉCTRICO PARA PRUEBA PILOTO


220VAC
FPS1
PS1
PS2
220AC
10VDC
18VDC
0V
+10V
+18V
0V
4-A2
4-A2
3-A2
3-B2
Fuente de energía
Author: Edgardo Gálvez
Date: 06/03/2020
File: NVFRv1_2.qet
Folio: 2/4


NOTA: Este diagrama aplica para cualquier cabezal.
AJUSTE LATERAL
AJUSTE DIAGONAL
DERECHA
IZQUIERDA
SUPERIOR
INFERIOR
SE1
SE2
4-E12
4-E12
4-D12
4-D12
4-I12
4-I12
4-H12
4-H12
ARD01
DPOT
LPOT
0V
+10V
2-H18
2-H18
4-C2
4-F2
Conexiones arduino
File: NVFRv1_2.qet
Folio: 3/4
Author: Edgardo Gálvez
Date: 06/03/2020


NOTA: Este diagrama aplica para cualquier cabezal.
MLAT
MDIA
4 Relay Module
R1
R2
R3
R4
Vcc
In1
In2
In3
In4
GND
3-F13
3-E13
3-E13
3-E13
3-C12
3-C12
3-C12
3-B12
+18V
2-G18
0V
2-G18
+5V
3-E3
0V
3-E2
Author: Edgardo Gálvez
Date: 06/03/2020
Conexiones relés - Cabezal
File: NVFRv1_2.qet
Folio: 4/4

HARDWARE DE LA PRUEBA PILOTO.



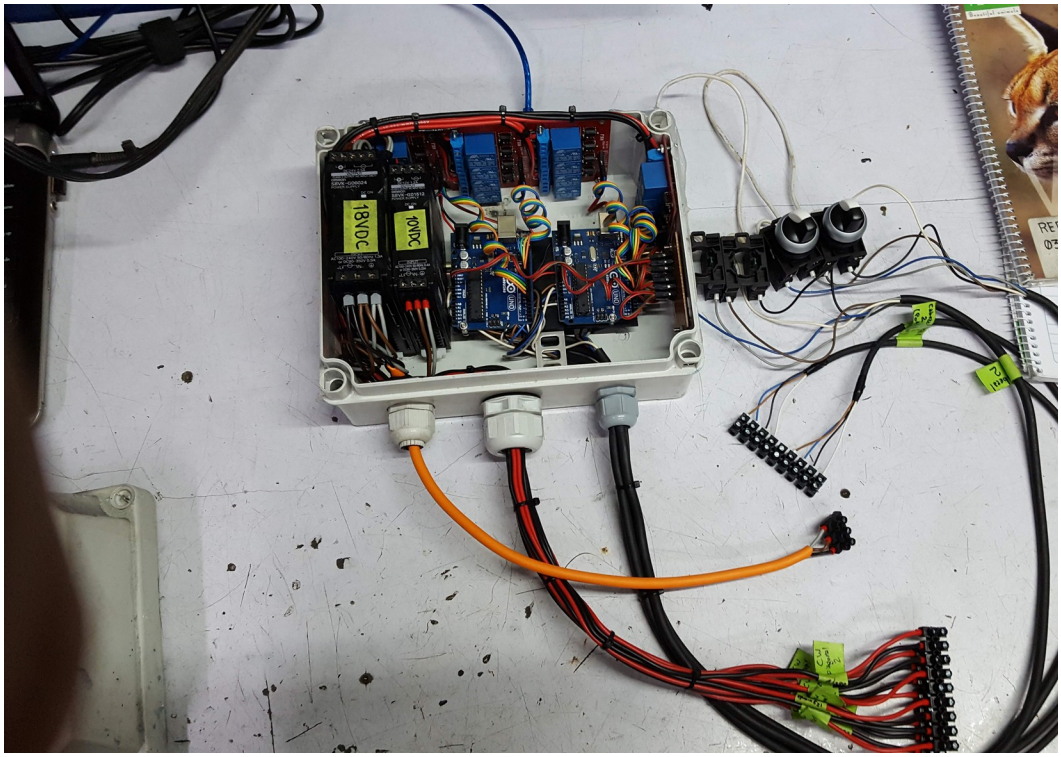
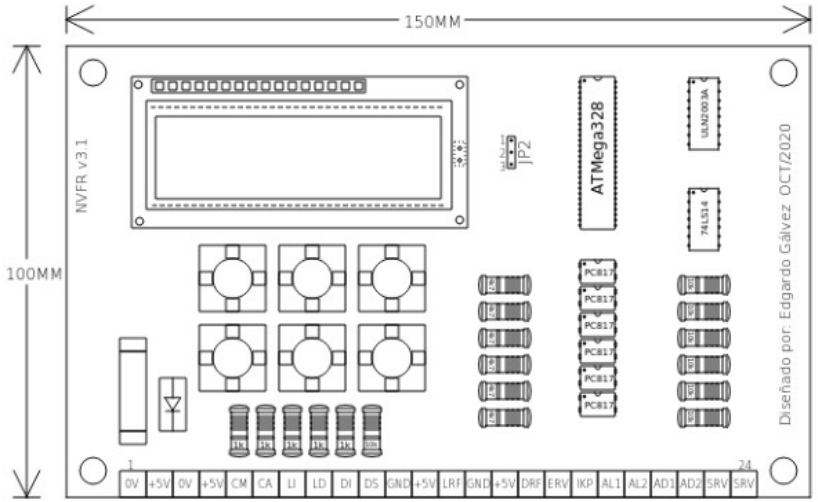
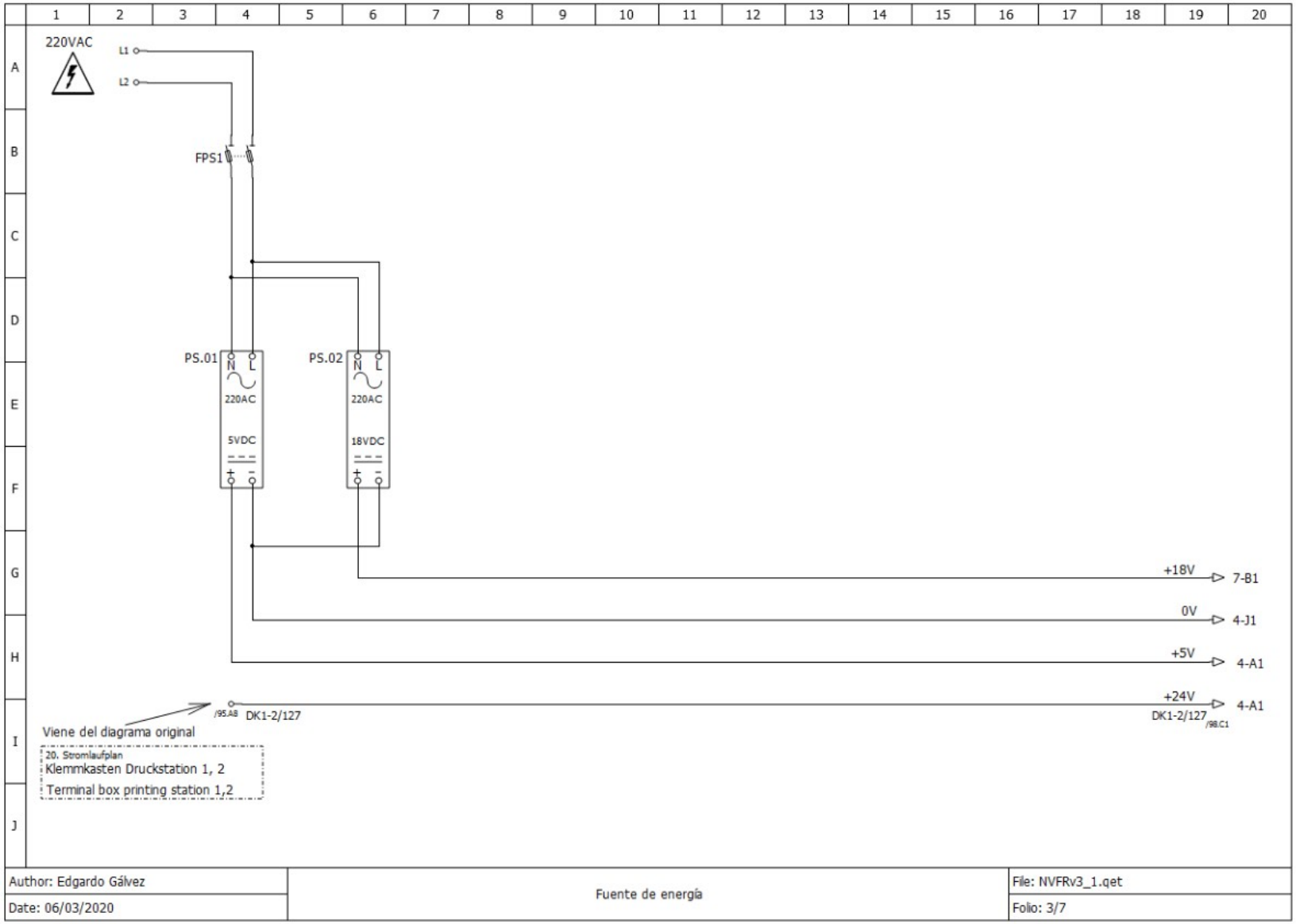


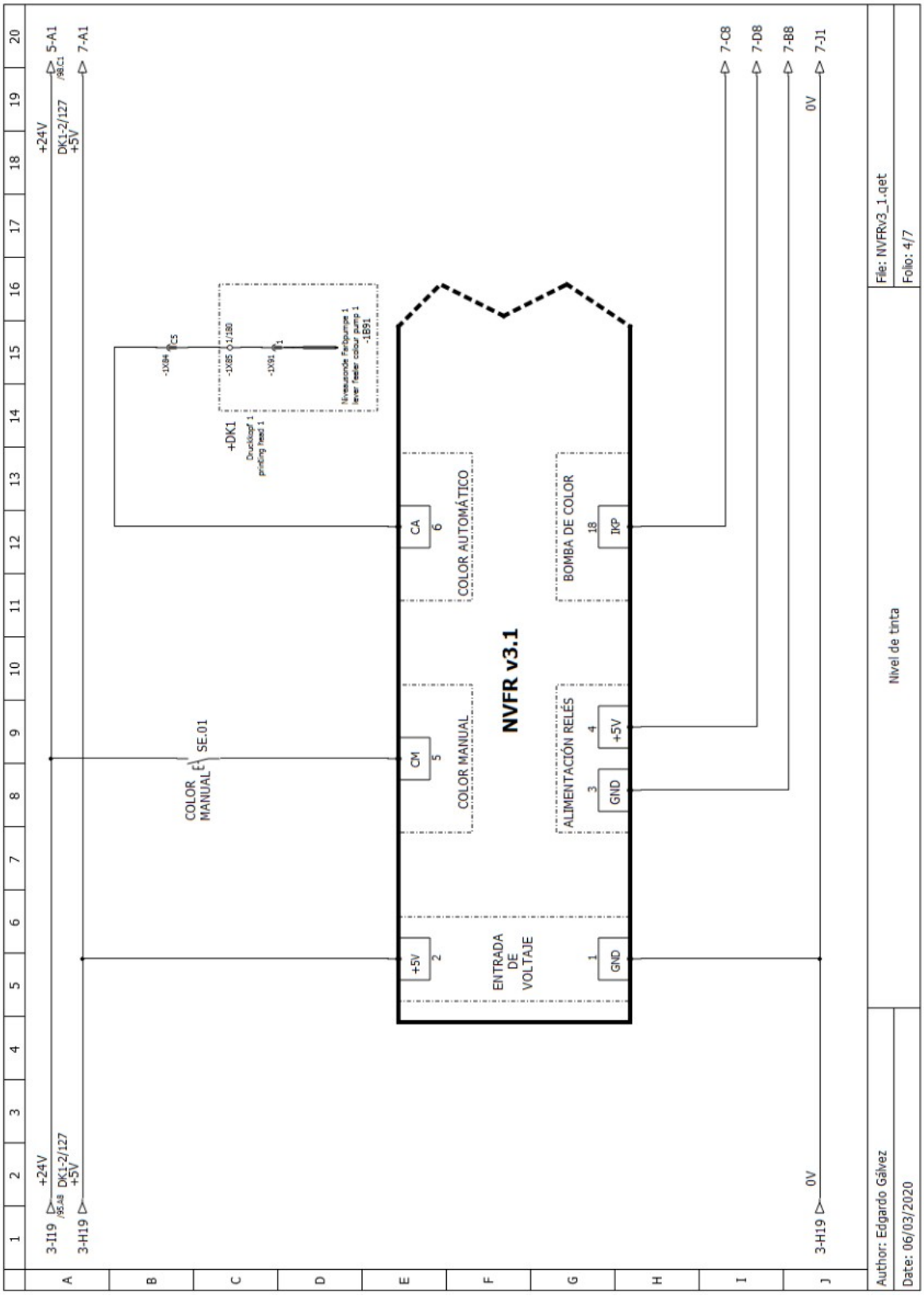
DIAGRAMA ELÉCTRICO USANDO PLACA NVFR v3.1

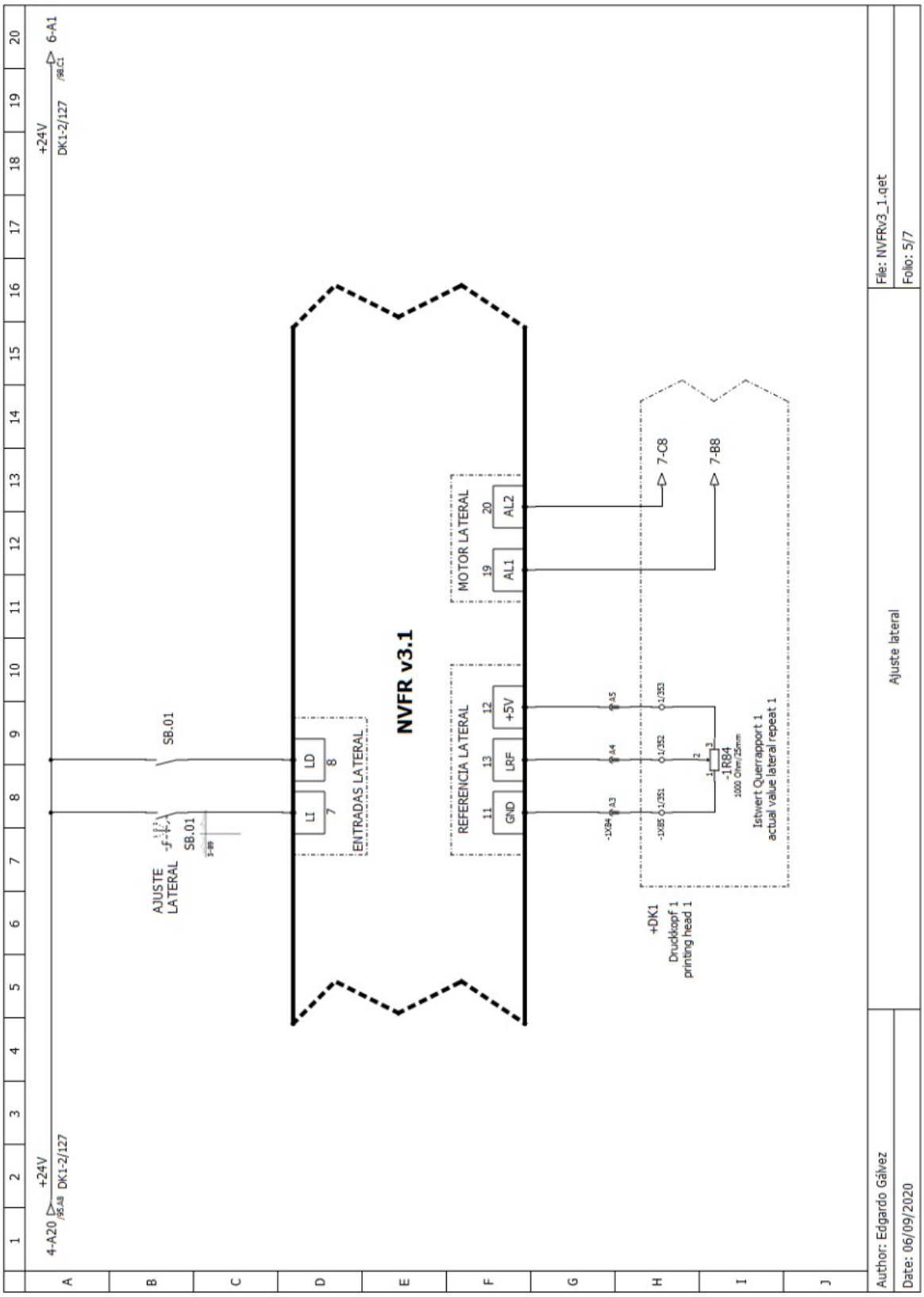


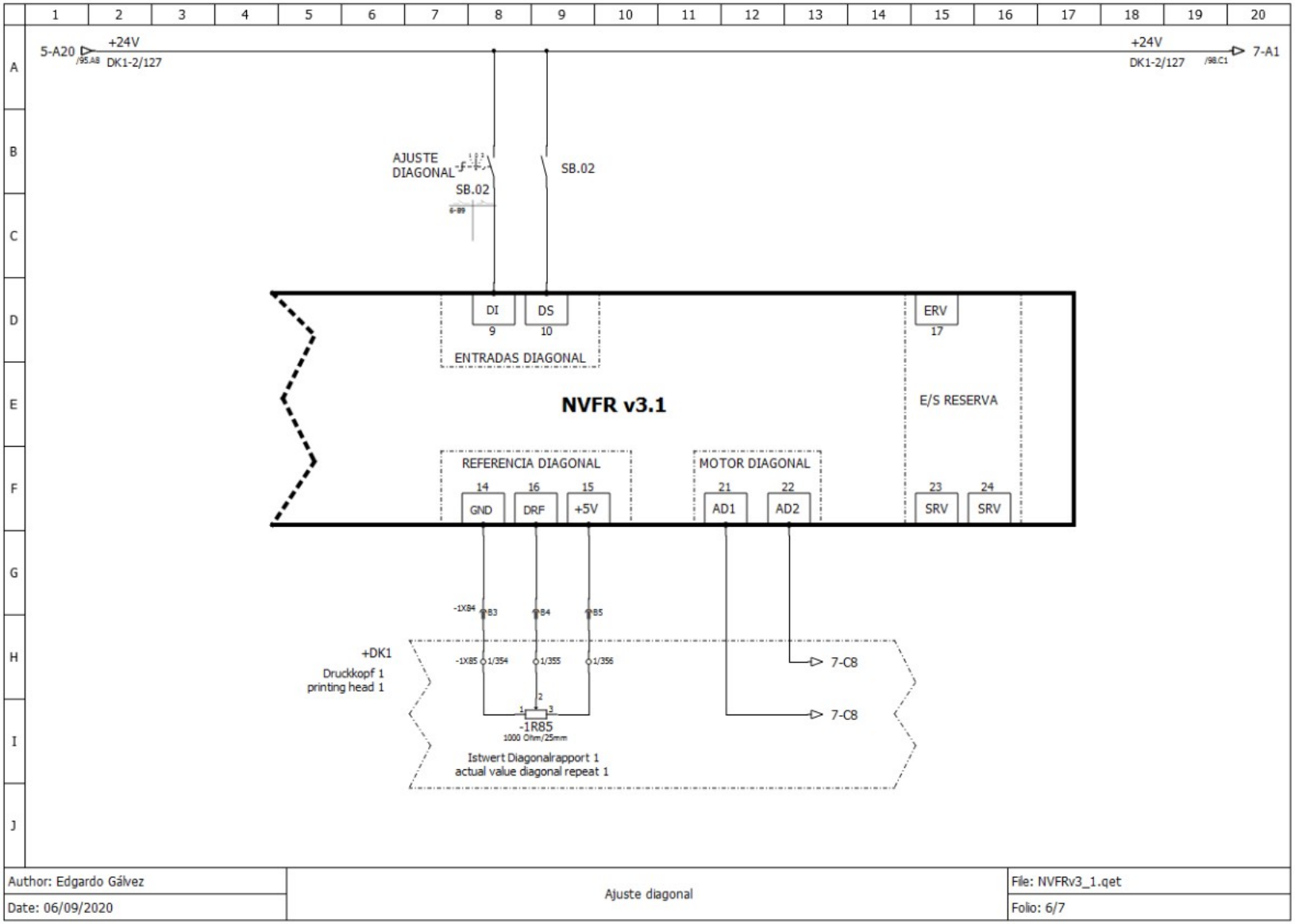
No. PIN	NOMBRE	DESCRIPCIÓN
01	GND	Entrada de alimentación 0V
02	+5V	Entrada de alimentación +5VDC
03	GND	Salida 0V
04	+5V	Salida +5VDC
05	CM	Entrada color manual
06	CA	Entrada color automático
07	LI	Entrada lateral izquierda
08	LD	Entrada lateral derecha
09	DI	Entrada diagonal inferior
10	DS	Entrada diagonal superior
11	GND	Salida 0V para pot. ref. lateral
12	+5V	Salida +5VDC para pot. ref. lateral
13	LRF	Referencia lateral
14	GND	Salida 0V para pot. ref. diagonal
15	+5V	Salida +5VDC para pot. ref. diagonal
16	DRF	Referencia diagonal
17	ERV	Entrada de reserva
18	IKP	Salida bomba de color
19	AL1	Salida hacia motor ajuste lateral
20	AL2	Salida hacia motor ajuste lateral
21	DL1	Salida hacia motor ajuste diagonal
22	DL2	Salida hacia motor ajuste diagonal
23	SRV	Salida de reserva
24	SRV	Salida de reserva
-	JP2(1,2)	Controlador listo
-	JP2(2,3)	Calibración.

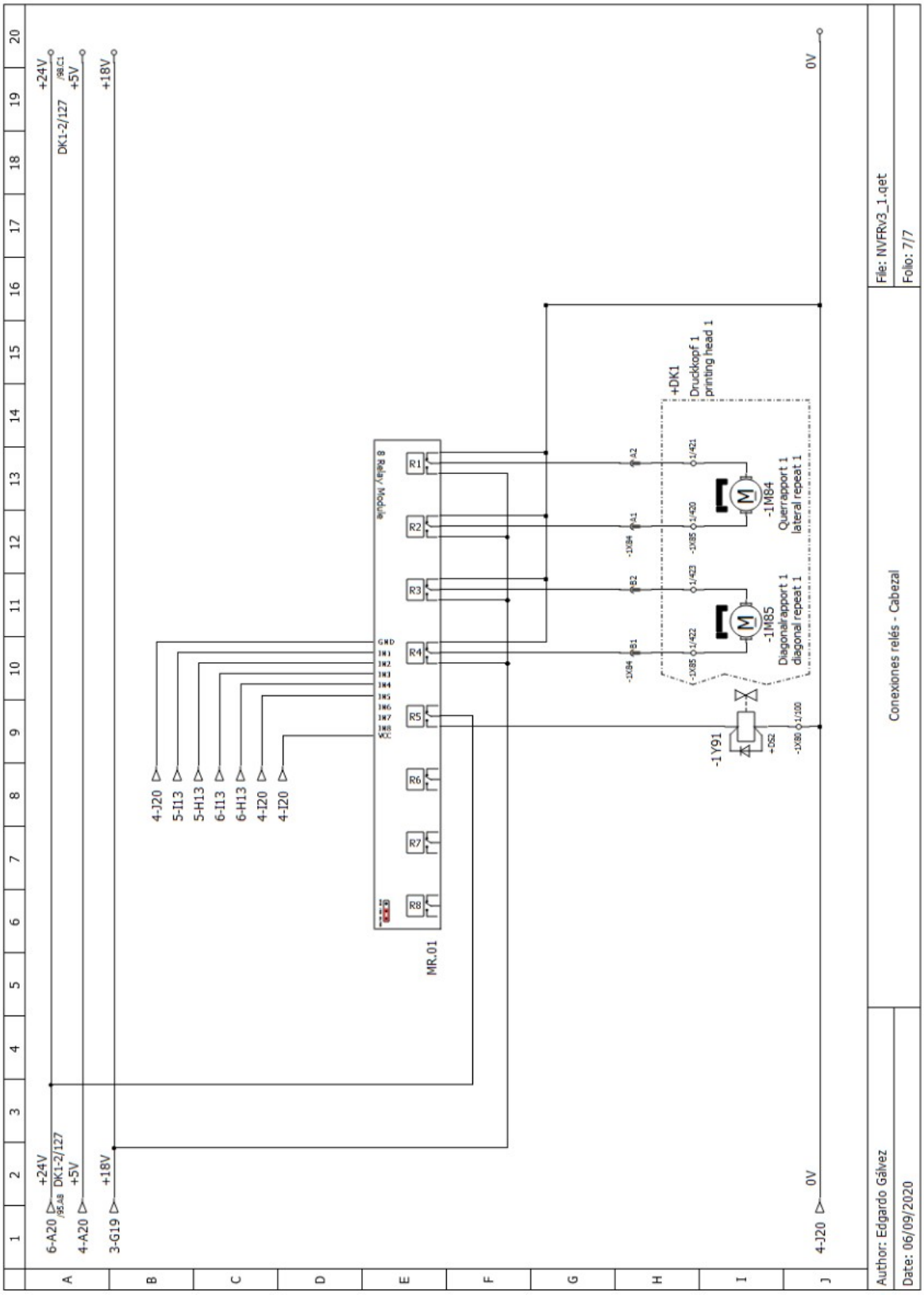
Author: Edgardo Gálvez	Vista general	File: NVFRv3_1.qet
Date: 3/6/20		Folio: 2/7


220VAC
L1
L2
FPS1
PS.01
N L
220AC
5VDC
+ -
PS.02
N L
220AC
18VDC
+ -
+18V
7-B1
0V
4-J1
+5V
4-A1
+24V
4-A1
/95.A8
DK1-2/127
DK1-2/127
/98.C1
Viene del diagrama original
20. Stromlaufplan
Klemmkasten Druckstation 1, 2
Terminal box printing station 1,2
Author: Edgardo Gálvez
Date: 06/03/2020
Fuente de energía
File: NVFRv3_1.qet
Folio: 3/7

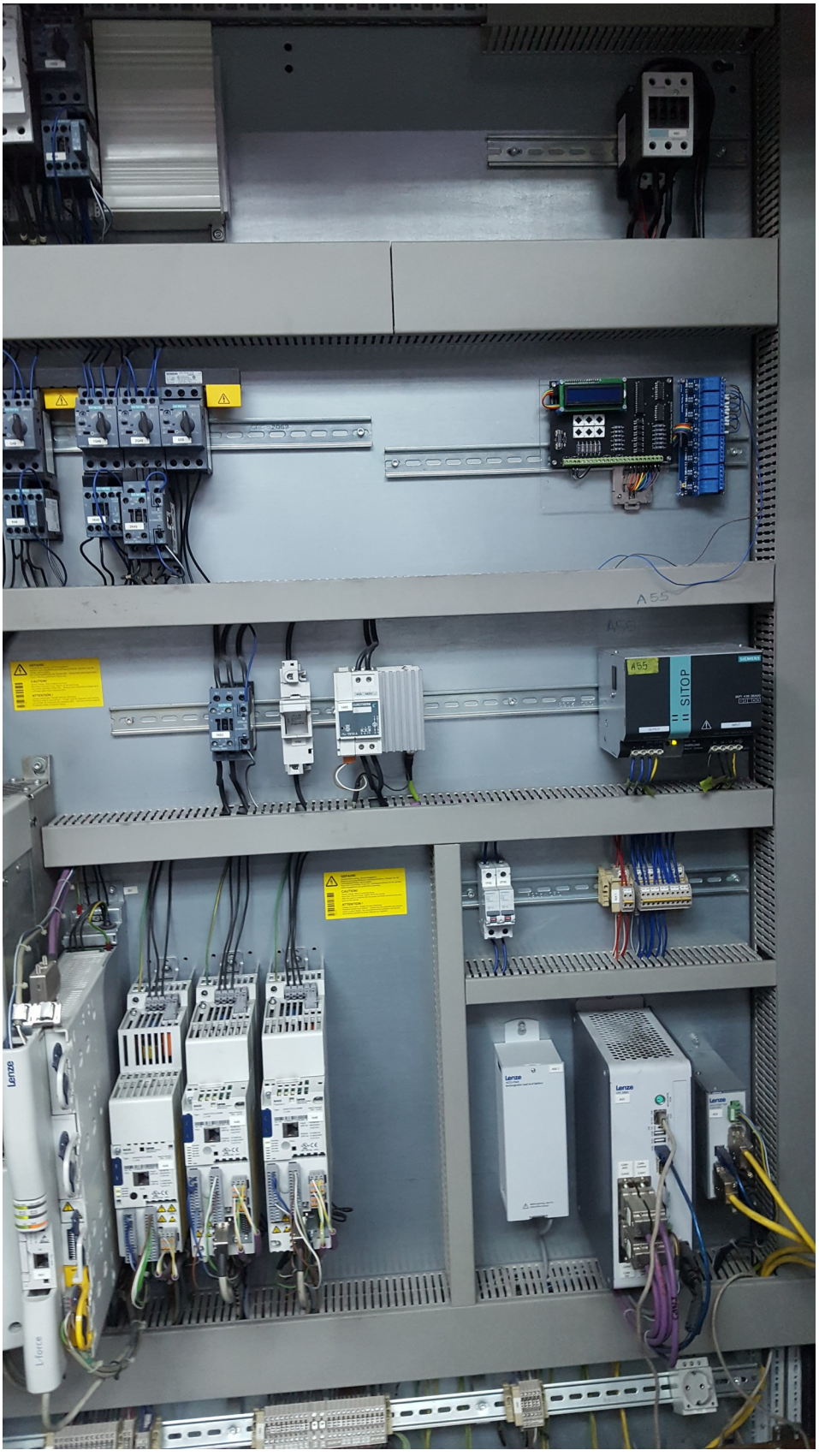

NVFR v3.1
ENTRADA DE VOLTAJE
+5V
2
GND
1
COLOR MANUAL
CM
5
COLOR AUTOMÁTICO
CA
6
ALIMENTACIÓN RELÉS
+5V
4
GND
3
BOMBA DE COLOR
IKP
18
COLOR MANUAL
SE.01
+DK1
+24V
DK1-2/127
+5V
0V
7-A1
5-A1
7-C8
7-D8
7-B8
7-J1
3-I19
3-H19
Nivel de tinta
Author: Edgardo Gálvez
Date: 06/03/2020
File: NVFRv3_1.qet
Folio: 4/7


NVFR v3.1
+24V
4-A20
DK1-2/127
+24V
DK1-2/127
6-A1
SB.01
SB.01
AJUSTE LATERAL
LI
LD
ENTRADAS LATERAL
REFERENCIA LATERAL
GND
LRF
+5V
MOTOR LATERAL
AL1
AL2
7-C8
7-B8
+DK1
Druckkopf 1
printing head 1
-1R84
1000 Ohm/25mm
Istwert Querrapport 1
actual value lateral repeat 1
Ajuste lateral
Author: Edgardo Gálvez
Date: 06/09/2020
File: NVFRv3_1.qet
Folio: 5/7


5-A20
+24V
DK1-2/127
+24V
DK1-2/127
7-A1
AJUSTE DIAGONAL
SB.02
SB.02
DI
9
DS
10
ENTRADAS DIAGONAL
NVFR v3.1
ERV
17
E/S RESERVA
REFERENCIA DIAGONAL
14
GND
16
DRF
15
+5V
MOTOR DIAGONAL
21
AD1
22
AD2
23
SRV
24
SRV
-1X84
B3
B4
B5
+DK1
Druckkopf 1
printing head 1
-1X85
1/354
1/355
1/356
7-C8
7-C8
-1R85
1000 Ohm/25mm
Istwert Diagonalrapport 1
actual value diagonal repeat 1
Author: Edgardo Gálvez
Date: 06/09/2020
Ajuste diagonal
File: NVFRv3_1.qet
Folio: 6/7


Conexiones relés - Cabezal
Author: Edgardo Gálvez
Date: 06/09/2020
File: NVFRv3_1.qet
Folio: 7/7
+24V
+5V
+18V
0V
DK1-2/127
6-A20
4-A20
3-G19
4-J20
8 Relay Module
MR.01
R1
R2
R3
R4
R5
R6
R7
R8
4-J20
5-I13
5-H13
6-I13
6-H13
4-I20
4-I20
+DK1
Druckkopf 1
printing head 1
-1M84
Querrapport 1
lateral repeat 1
-1M85
Diagonalrapport 1
diagonal repeat 1
-1Y91

PLACA NVFR v3.1 MONTADA EN PANEL ELÉCTRICO DE LA MÁQUINA.




A55
A55
SIEMENS
SITOP

COTIZACIONES

We are closed for holidays from 21.12.2019 - 06.01.2020.
We will be back at your service on Tuesday 7th of January 2020.

Messrs.
Textufil S.A de C.V.
12 Avenida Sur Contiguo a Fabrica Diana
Soyapango
San Salvador
El Salvador

Date:	23.10.2019
Contact:	Josef Koschier
Phone:	+43-463-3848-237
Fax:	+43-463-319203
E-Mail:	spareparts@zimmer-austria.com
Customer-internal:	487 - Textufil (SLV)
Validity:	3 months

Delivery address
Messrs.
Textufil S.A de C.V.
12 Avenida Sur Contiguo a Fabrica Diana
Soyapango
San Salvador
El Salvador

PROFORMA-INVOICE - 17033-19

Rota-TG/116
According to your enquiry we are pleased to offer you as follows
_Ref.: Req. T021019E / 02.10.2019

Item	Part Drawing / Description	Quantity	Unit	Unit price €	Price/Item €	Price/Item net €
10	Our Part: A-89688 **SCS-printing module II** Base module Code: 70063800 Customs tariff code: 85389099 • With loaded program • 1A12, ... in circuit diagram Machine: Rota-TG/116	2	pcs	1.012,00	2.024,00	2.024,00
20	Our Part: A-89689 **SCS-printing module II** Plug in card motor control Code: 70036700 Customs tariff code: 85389099 • 1A12.2, ... in circuit diagram Machine: Rota-TG/116	6	pcs	449,00	2.694,00	2.694,00
30	Our Part: A-89690 **SCS-printing module II** Plug in card level control Code: 70036800 Customs tariff code: 85362090 • 1A12.1, ... in circuit diagram Machine: Rota-TG/116	2	pcs	248,00	496,00	496,00

PROFORMA-INVOICE - 17033-19

Rota-TG/116

Item	Part Drawing / Description	Quantity Unit	Unit price €	Price/Item €	Price/Item net €
Net price				€	5.214,00
Packing				€	12,00
Total price EXW Klagenfurt				**€**	**5.226,00**

Terms of payment:
100% prepayment with the order

Delivery time:
approx. 3 working weeks after receipt of order and any agreed prepayment

Packing/Weight:
1 carton approx. 42 x 42 x 22 cm net: approx. 4,00 kg / gross: approx. 5,00 kg

Freight-, insurance-, customs- and import duties, installation and commissioning at customer's side are not included in above mentioned price, besides mentioned in a quotation position. Shipping date applies after receipt of order and possible down payment.
Subject to prior sale. For repair: Subject to actability.
The delivery will be effected in accordance with the General Terms of Delivery of the Association of the Austrian Machinery and Steel Construction Industries.
We refer to http://www.zimmer-klagenfurt.com/en/content/general-terms-0

J. Zimmer Maschinenbau
GmbH Klagenfurt
(This is an automatically generated quotation and therefore valid without signature)

J.Zimmer Maschinenbau GmbH Klagenfurt
Ebentaler Straße 133, 9020 Klagenfurt, Austria
UID: ATU 25267804 Landesgericht Klagenfurt FN 92181k
T: +43-463-3848-0 F: +43-463-319203 I: www.zimmer-austria.com

BKS Bank AG, 9020 Klagenfurt, Austria
IBAN: AT06 1700 0001 0039 5770 BIC: BFKK AT2K
UniCredit Bank Austria AG, 9020 Klagenfurt, Austria
IBAN: AT37 1100 0098 1344 3000 BIC: BKAU ATWW

Messrs.
Textufil S.A de C.V.
12 Avenida Sur Contiguo a Fabrica Diana
Soyapango
San Salvador
El Salvador

Date:	14.02.2020
Contact:	Josef Koschier
Phone:	+43-463-3848-237
Fax:	+43-463-319203
E-Mail:	spareparts@zimmer-austria.com
Customer-internal:	487 - Textufil (SLV)
Validity:	3 months

Delivery address
Messrs.
Textufil S.A de C.V.
12 Avenida Sur Contiguo a Fabrica Diana
Soyapango
San Salvador
El Salvador

PROFORMA-INVOICE - 17424-20

Rota-TG/116
According to your enquiry we are pleased to offer you as follows
_Ref.: Req. T0700220E / 12.02.2020

Item	Part Drawing / Description	Quantity	Unit	Unit price €	Price/Item €	Price/Item net €
10	Our Part: A-89688 **SCS-printing module II** Base module Code: 70063800 Customs tariff code: 85389099 • With loaded program • 1A12, ... in circuit diagram Machine: Rota-TG/116	2	pcs	1.012,00	2.024,00	2.024,00
20	Our Part: A-89689 **SCS-printing module II** Plug in card motor control Code: 70036700 Customs tariff code: 85389099 • 1A12.2, ... in circuit diagram Machine: Rota-TG/116	2	pcs	449,00	898,00	898,00
30	Our Part: A-89690 **SCS-printing module II** Plug in card level control Code: 70036800 Customs tariff code: 85362090 • 1A12.1, ... in circuit diagram Machine: Rota-TG/116	2	pcs	248,00	496,00	496,00

PROFORMA-INVOICE - 17424-20

Rota-TG/116

Item	Part Drawing / Description	Quantity Unit	Unit price €	Price/Item €	Price/Item net €
Net price				€	3.418,00
Packing				€	12,00
Total price EXW Klagenfurt				**€**	**3.430,00**

Terms of payment:
100% prepayment with the order

Delivery time:
approx. 3 working weeks after receipt of order and any agreed prepayment

Packing/Weight:
1 carton approx. 42 x 42 x 22 cm net: approx. 3,00 kg / gross: approx. 4,00 kg

Freight-, insurance-, customs- and import duties, installation and commissioning at customer's side are not included in above mentioned price, besides mentioned in a quotation position. Shipping date applies after receipt of order and possible down payment.
Subject to prior sale. For repair: Subject to actability.
The delivery will be effected in accordance with the General Terms of Delivery of the Association of the Austrian Machinery and Steel Construction Industries.
We refer to http://www.zimmer-klagenfurt.com/en/content/general-terms-0

J. Zimmer Maschinenbau
GmbH Klagenfurt
(This is an automatically generated quotation and therefore valid without signature)

J.Zimmer Maschinenbau GmbH Klagenfurt
Ebentaler Straße 133, 9020 Klagenfurt, Austria
UID: ATU 25267804 Landesgericht Klagenfurt FN 92181k
T: +43-463-3848-0 F: +43-463-319203 I: www.zimmer-austria.com

BKS Bank AG, 9020 Klagenfurt, Austria
IBAN: AT06 1700 0001 0039 5770 BIC: BFKK AT2K
UniCredit Bank Austria AG, 9020 Klagenfurt, Austria
IBAN: AT37 1100 0098 1344 3000 BIC: BKAU ATWW

CÓDIGO FUENTE DEL SOFTWARE NVFRv1_6

```
/*****************************************************************
*    Autor: Edgardo Gálvez                                       *
*    Fecha: 15.oct.2020                                          *
*    Proyecto: DESARROLLO DE UNA PLACA ELECTRÓNICA APLICADA EN LA *
*              INDUSTRIA TEXTIL BASADA EN EL HARDWARE Y SOFTWARE  *
*              DE LIBRE UTILIZACIÓN.                              *
*                                                                 *
*    Descripción: Alineación lateral/diagonal del cilindro        *
*                 del cabezal y control de bomba de color.        *
*                                                                 *
*                        ***CALIBRACIÓN***                        *
*    LATERAL:                                                     *
*    Colocar el cabezal en posición LATERAL mínima, el voltaje    *
*    en el potenciómetro debe ser:0.50V                           *
*    Colocar el cabezal en posición LATERAL máxima, el voltaje    *
*    en el potenciómetro debe ser:3.50V                           *
*                                                                 *
*    DIAGONAL:                                                    *
*    Colocar el cabezal en posición DIAGONAL mínima, el voltaje   *
*    en el potenciómetro debe ser:0.50V                           *
*    Colocar el cabezal en posición DIAGONAL máxima, el voltaje   *
*    en el potenciómetro debe ser:1.00V                           *
*****************************************************************
*/

#include <Wire.h>
#include <LiquidCrystal_I2C.h>

//Prototipos de rutinas utilizadas.
void  intro(void);
int   leerEntrada(void);
float leerVoltaje(int analogInput);
int   leerTeclado(void);
void  autoAjusteSeguro(float alp, float adp);
void  calibracion(void);
void  activarMotor(int salida ,  int msec);
void  apagarSalidas(void);
void  monitor(void);

//ENTRADAS DIGITALES.
const int entradaColMan = 2; // Bomba de tinta - Manual.
const int entradaColAut = 3; // Bomba de tinta - Automático.
const int entradaLatIzq = 4; // Entrada para mover a la izquierda.
const int entradaLatDer = 5; // Entrada para mover a la derecha.
const int entradaDiaInf = 6; // Entrada para mover a inferior.
const int entradaDiaSup = 7; // Entrada para mover a superior.
const int entradaCalib  = 13;// Calibración.
// ENTRADAS ANÁLOGAS.
```

```
49.      const int tecladoAnalogo = A0;      // Teclado análogo.
50.      const int referenciaLateral = A2;   // Potenciómetro lateral.
51.      const int referenciaDiagonal = A3;  // Potenciómetro diagonal.
52.
53.      //COMUNICACION I2C;
54.      const int lcd_sda = A4; // Línea de datos pantalla lcd.
55.      const int lcd_scl = A5; // Línea de reloj pantalla lcd.
56.
57.      // SALIDAS DIGITALES.
58.      //Para ajuste lateral.
59.      const int salidaLatIzq = 8; // Ajuste lateral hacia la izquierda.
60.      const int salidaLatDer = 9; // Ajuste lateral hacia la derecha.
61.
62.      //Para ajuste diagonal.
63.      const int salidaDiaInf = 10; // Ajuste diagonal hacia superior.
64.      const int salidaDiaSup = 11; // Ajuste diagonal hacia inferior.
65.
66.      //Para bomba de color.
67.      const int salidaColor = 12; //Activación de bomba de color.
68.
69.      //Constantes para valores mínimos, máximos y de tiempo.
70.      #define POS_LAT_MIN     0.50    // Valor mínimo de voltaje lateral.
71.      #define POS_LAT_MAX     3.50    // Valor máximo de voltaje lateral.
72.      #define POS_DIA_MIN     0.50    // Valor mínimo de voltaje diagonal.
73.      #define POS_DIA_MAX     1.00    // Valor máximo de voltaje diagonal.
74.      #define TIEMPO_PULSO    100     // Pulso de tiempo para motor.
75.
76.      LiquidCrystal_I2C lcd(0x27,16,2);
77.
78.      void setup(){
79.
80.          Serial.begin(9600);
81.          lcd.init();
82.          lcd.backlight();
83.
84.          intro();  //Pantalla de inicio.
85.
86.          //ENTRADAS DIGITALES.
87.          pinMode(entradaCalib, INPUT_PULLUP);
88.          pinMode(entradaColMan, INPUT);
89.          pinMode(entradaColAut, INPUT);
90.          pinMode(entradaLatIzq, INPUT);
91.          pinMode(entradaLatDer, INPUT);
92.          pinMode(entradaDiaInf, INPUT);
93.          pinMode(entradaDiaSup, INPUT);
94.
95.          //ENTRADAS ANÁLOGAS.
96.          pinMode(tecladoAnalogo, INPUT);
97.          pinMode(referenciaLateral,INPUT);
```

```
98.           pinMode(referenciaDiagonal,INPUT);
99.
100.          //SALIDAS DIGITALES.
101.          pinMode(salidaLatIzq, OUTPUT);
102.          pinMode(salidaLatDer, OUTPUT);
103.          pinMode(salidaDiaInf, OUTPUT);
104.          pinMode(salidaDiaSup, OUTPUT);
105.          pinMode(salidaColor, OUTPUT);
106.
107.      }
108.
109.      void loop() {
110.
111.          float voltPosLat = leerVoltaje(referenciaLateral);
112.          float voltPosDia = leerVoltaje(referenciaDiagonal);
113.          int direccionAjuste = leerEntrada();
114.
115.          autoAjusteSeguro(voltPosLat, voltPosDia);
116.          monitor();
117.
118.          switch(direccionAjuste){
119.
120.              case 1:
121.                  if(voltPosLat > POS_LAT_MIN && voltPosLat < POS_LAT_MAX )
122.                      activarMotor(salidaLatIzq , TIEMPO_PULSO);
123.                      break;
124.
125.              case 2:
126.                  if(voltPosLat > POS_LAT_MIN && voltPosLat < POS_LAT_MAX )
127.                      activarMotor(salidaLatDer, TIEMPO_PULSO);
128.                      break;
129.
130.              case 3:
131.                  if(voltPosDia > POS_DIA_MIN && voltPosDia < POS_DIA_MAX )
132.                      activarMotor(salidaDiaInf, TIEMPO_PULSO);
133.                      break;
134.
135.              case 4:
136.                  if(voltPosDia > POS_DIA_MIN && voltPosDia < POS_DIA_MAX )
137.                      activarMotor(salidaDiaSup, TIEMPO_PULSO);
138.                      break;
139.
140.              case 5: //COLOR: Para la bomba de color manual.
141.                  digitalWrite(salidaColor,HIGH);
142.                  break;
143.
144.              case 6: //CALIBRACIÓN.
145.                  while(digitalRead(entradaCalib) == LOW){
146.                      calibracion();
```

```
                }
                break;

            case 7: //COLOR: Para la bomba de color auto.
                digitalWrite(salidaColor,HIGH);
                break;

            default: //Apaga todos los relés en condiciones no previstas.
                autoAjusteSeguro(voltPosLat, voltPosDia);
                apagarSalidas();
                break;

        }

    }

    //PANTALLA DE INICIO.
    void intro(void){

        String stringTemp, stringDot;
        int i=0;

        stringTemp = String(".");
        stringDot = String(".");

        lcd.setCursor(0,0);
        lcd.print(" Navigator free");

        for(i=0;i<15;i++){
            lcd.setCursor(0,1);
            stringTemp += stringDot;
            lcd.print(stringTemp);
            delay(150);
        }

        lcd.clear();
        lcd.setCursor(0,0);
        lcd.print("Elaborado por:");
        lcd.setCursor(0,1);
        lcd.print("Edgardo Galvez");
        delay(1500);

        lcd.clear();
        lcd.setCursor(0,0);
        lcd.print("Version hardware");
        lcd.setCursor(4,1);
        lcd.print("NVFR v3.1");
        delay(1500);
        lcd.clear();
```

```
196.
197.         lcd.setCursor(0,0);
198.         lcd.print("Version software");
199.         lcd.setCursor(4,1);
200.         lcd.print("NVFR v1_6");
201.         delay(1500);
202.         lcd.clear();
203.
204.     }
205.
206.     /*Lee las entradas digitales
207.     +------------+-----------+
208.     |  RETORNO   |   MODO    |
209.     +------------+-----------+
210.     |    0       |  Reading  |
211.     +------------+-----------+
212.     |  1 - 6     |  Manual   |
213.     +------------+-----------+
214.     |  7 - 10    |   AUTO    |
215.     +------------+-----------+
216.     */
217.     int leerEntrada(void){
218.
219.         int move_to_left = 1;
220.         int move_to_right = 2;
221.         int move_to_down = 3;
222.         int move_to_up = 4;
223.         int ink_pump_man = 5;
224.         int calibration = 6;
225.         int ink_pump_aut = 7;
226.
227.         if(digitalRead(entradaLatDer)==LOW &&  \
228.             digitalRead(entradaLatIzq)==HIGH ||   \
229.             leerTeclado()==1)
230.             return move_to_left;
231.
232.         else if(digitalRead(entradaLatDer)==HIGH &&  \
233.             digitalRead(entradaLatIzq)==LOW ||     \
234.             leerTeclado()==2)
235.             return move_to_right;
236.
237.         else if(digitalRead(entradaDiaInf)==HIGH && \
238.             digitalRead(entradaDiaSup)==LOW ||      \
239.             leerTeclado()==3)
240.             return move_to_down;
241.
242.         else if(digitalRead(entradaDiaInf)==LOW &&  \
243.             digitalRead(entradaDiaSup)==HIGH ||     \
244.             leerTeclado()==4)
```

```
245.             return move_to_up;
246.
247.         else if(digitalRead(entradaColMan)==HIGH || \
248.             leerTeclado()==5)
249.             return ink_pump_man;
250.
251.         else if(digitalRead(entradaCalib)==LOW)
252.             return calibration;
253.
254.         else if(digitalRead(entradaColAut)==HIGH)
255.             return ink_pump_aut;
256.
257.         else if(leerVoltaje(referenciaLateral) < POS_LAT_MIN)
258.             return 8;
259.
260.         else if(leerVoltaje(referenciaLateral) > POS_LAT_MAX)
261.             return 9;
262.
263.         else if(leerVoltaje(referenciaDiagonal) < POS_DIA_MIN)
264.             return 10;
265.
266.         else if(leerVoltaje(referenciaDiagonal) > POS_DIA_MAX)
267.             return 11;
268.
269.         else
270.             return 0;
271.
272.     }
273.
274.     //Lectura y conversión de valor leído
275.     float leerVoltaje( int entradaAnaloga){
276.
277.         int valorLeido = analogRead(entradaAnaloga);
278.         float voltaje = valorLeido * (5.0 / 1023);
279.         return voltaje;
280.
281.     }
282.     //Lee los pulsadores del teclado análogo.
283.     int leerTeclado(void){
284.
285.         int button_lateral_left = 1;
286.         int button_lateral_right = 2;
287.         int button_diagonal_down = 3;
288.         int button_diagonal_up = 4;
289.         int button_ink_pump = 5;
290.         int option_key = analogRead(tecladoAnalogo);
291.
292.         if (option_key > 1000)
293.             return 0;
```

```
294.
295.         if (option_key > 0 && option_key < 100)
296.             return button_lateral_left;
297.
298.         if (option_key > 450 && option_key < 550)
299.             return button_lateral_right;
300.
301.         if (option_key > 598 && option_key < 698)
302.             return button_ink_pump;
303.
304.         if (option_key > 699 && option_key < 789)
305.             return button_diagonal_down;
306.
307.         if (option_key > 790 && option_key < 900)
308.             return button_diagonal_up;
309.
310.         return 0;
311.
312.     }
313.
314.     //Ubica en posición correcta el cabezal para una operación segura.
315.     void autoAjusteSeguro(float alp, float adp){
316.
317.         if(alp < POS_LAT_MIN){//posición=limite izquierda
318.             activarMotor(salidaLatDer, TIEMPO_PULSO);
319.         }
320.         else if(alp > POS_LAT_MAX){//posición=limite derecha
321.             activarMotor(salidaLatIzq, TIEMPO_PULSO);
322.         }
323.
324.         if(adp < POS_DIA_MIN){//posición=limite abajo
325.             activarMotor(salidaDiaSup, TIEMPO_PULSO);
326.         }
327.         else if(adp > POS_DIA_MAX){//posición=limite arriba
328.             activarMotor(salidaDiaInf, TIEMPO_PULSO);
329.         }
330.
331.     }
332.
333.     //Función para calibración inicial.
334.     void calibracion(void){
335.
336.         int opcion = leerEntrada();
337.
338.         monitor();
339.
340.         if(opcion == 1 ){
341.             activarMotor(salidaLatIzq , TIEMPO_PULSO);
342.         }
```

```
343.
344.         else if(opcion == 2 ){
345.             activarMotor(salidaLatDer , TIEMPO_PULSO);
346.         }
347.
348.         else if(opcion == 3 ){
349.             activarMotor(salidaDiaInf , TIEMPO_PULSO);
350.         }
351.
352.         else if(opcion == 4 ){
353.             activarMotor(salidaDiaSup , TIEMPO_PULSO);
354.         }
355.
356.         apagarSalidas();
357.
358.     }
359.
360.     //Función para activar salidas de manera pulsante.
361.     void activarMotor( int salida ,  int msec){
362.
363.         monitor();
364.         apagarSalidas();
365.         digitalWrite(salida, HIGH);
366.         delay(msec);
367.
368.     }
369.
370.     //Apaga todos los reles.
371.     void apagarSalidas(void){
372.
373.         monitor();
374.         digitalWrite(salidaLatIzq, LOW);
375.         digitalWrite(salidaLatDer, LOW);
376.         digitalWrite(salidaDiaInf, LOW);
377.         digitalWrite(salidaDiaSup, LOW);
378.         digitalWrite(salidaColor, LOW);
379.
380.     }
381.
382.     //Muestra valores/eventos en monitor serial y pantalla LCD.
383.     void monitor(void){
384.
385.         float lv = leerVoltaje(referenciaLateral);
386.         float dv = leerVoltaje(referenciaDiagonal);
387.         int direccionAjuste = leerEntrada();
388.         int option_key = analogRead(tecladoAnalogo);
389.
390.         Serial.print("   ");
391.         Serial.print(direccionAjuste);
```

```
392.         Serial.print("   ");
393.         Serial.print(option_key);
394.         Serial.print("   ");
395.         Serial.print(lv);
396.         Serial.print("V   ");
397.         Serial.print(dv);
398.         Serial.print("V");
399.
400.         switch(direccionAjuste){
401.
402.             case 0:
403.                 Serial.println("  Status: Reading . . .");
404.                 lcd.setCursor(0,0);
405.                 lcd.print("Estado: Leyendo.");
406.                 lcd.setCursor(0,1);
407.                 lcd.print("VL:");
408.                 lcd.setCursor(3,1);
409.                 lcd.print(lv);
410.                 lcd.setCursor(8,1);
411.                 lcd.print("VD:");
412.                 lcd.setCursor(11,1);
413.                 lcd.print(dv);
414.                 break;
415.
416.             case 1:
417.                 Serial.println("  Status: Running left");
418.                 lcd.setCursor(0,0);
419.                 lcd.print("Estado: Lat.izq");
420.                 lcd.setCursor(0,1);
421.                 lcd.print("VL:");
422.                 lcd.setCursor(3,1);
423.                 lcd.print(lv);
424.                 lcd.setCursor(8,1);
425.                 lcd.print("VD:");
426.                 lcd.setCursor(11,1);
427.                 lcd.print(dv);
428.                 break;
429.
430.             case 2:
431.                 Serial.println("  Status: Running right");
432.                 lcd.setCursor(0,0);
433.                 lcd.print("Estado: Lat.der");
434.                 lcd.setCursor(0,1);
435.                 lcd.print("VL:");
436.                 lcd.setCursor(3,1);
437.                 lcd.print(lv);
438.                 lcd.setCursor(8,1);
439.                 lcd.print("VD:");
440.                 lcd.setCursor(11,1);
```

```
441.                 lcd.print(dv);
442.                 break;
443.
444.             case 3:
445.                 Serial.println("  Status: Running down");
446.                 lcd.setCursor(0,0);
447.                 lcd.print("Estado: Dia.inf");
448.                 lcd.setCursor(0,1);
449.                 lcd.print("VL:");
450.                 lcd.setCursor(3,1);
451.                 lcd.print(lv);
452.                 lcd.setCursor(8,1);
453.                 lcd.print("VD:");
454.                 lcd.setCursor(11,1);
455.                 lcd.print(dv);
456.                 break;
457.
458.             case 4:
459.                 Serial.println("  Status: Running up");
460.                 lcd.setCursor(0,0);
461.                 lcd.print("Estado: Dia.sup");
462.                 lcd.setCursor(0,1);
463.                 lcd.print("VL:");
464.                 lcd.setCursor(3,1);
465.                 lcd.print(lv);
466.                 lcd.setCursor(8,1);
467.                 lcd.print("VD:");
468.                 lcd.setCursor(11,1);
469.                 lcd.print(dv);
470.                 break;
471.
472.             case 5:
473.                 Serial.println("  Status: Ink pump -> Manual!");
474.                 lcd.setCursor(0,0);
475.                 lcd.print("Estado:Color man");
476.                 lcd.setCursor(0,1);
477.                 lcd.print("VL:");
478.                 lcd.setCursor(3,1);
479.                 lcd.print(lv);
480.                 lcd.setCursor(8,1);
481.                 lcd.print("VD:");
482.                 lcd.setCursor(11,1);
483.                 lcd.print(dv);
484.                 break;
485.
486.             case 6:
487.                 Serial.println("  Status: Calibrating . . .");
488.                 lcd.setCursor(0,0);
489.                 lcd.print("Estado: Calibrar");
```

```
                lcd.setCursor(0,1);
                lcd.print("VL:");
                lcd.setCursor(3,1);
                lcd.print(lv);
                lcd.setCursor(8,1);
                lcd.print("VD:");
                lcd.setCursor(11,1);
                lcd.print(dv);
                break;

            case 7:
                Serial.println("  Status: Ink pump -> Auto!");
                lcd.setCursor(0,0);
                lcd.print("Estado:Color Aut");
                lcd.setCursor(0,1);
                lcd.print("VL:");
                lcd.setCursor(3,1);
                lcd.print(lv);
                lcd.setCursor(8,1);
                lcd.print("VD:");
                lcd.setCursor(11,1);
                lcd.print(dv);
                break;

            case 8:
                Serial.println("  Warning: Lateral lower limit reached!");
                lcd.setCursor(0,0);
                lcd.print("Estado: Auto-der");
                lcd.setCursor(0,1);
                lcd.print("VL:");
                lcd.setCursor(3,1);
                lcd.print(lv);
                lcd.setCursor(8,1);
                lcd.print("VD:");
                lcd.setCursor(11,1);
                lcd.print(dv);
                break;

            case 9:
                Serial.println("  Warning: Lateral higher limit reached!");
                lcd.setCursor(0,0);
                lcd.print("Estado: Auto-izq");
                lcd.setCursor(0,1);
                lcd.print("VL:");
                lcd.setCursor(3,1);
                lcd.print(lv);
                lcd.setCursor(8,1);
                lcd.print("VD:");
                lcd.setCursor(11,1);
```

```
539.                 lcd.print(dv);
540.                 break;
541.
542.             case 10:
543.                 Serial.println("  Warning: Diagonal lower limit reached!");
544.                 lcd.setCursor(0,0);
545.                 lcd.print("Estado: Auto-sup");
546.                 lcd.setCursor(0,1);
547.                 lcd.print("VL:");
548.                 lcd.setCursor(3,1);
549.                 lcd.print(lv);
550.                 lcd.setCursor(8,1);
551.                 lcd.print("VD:");
552.                 lcd.setCursor(11,1);
553.                 lcd.print(dv);
554.                 break;
555.
556.             case 11:
557.                 Serial.println(" Warning: Diagonal higher limit reached!");
558.                 lcd.setCursor(0,0);
559.                 lcd.print("Estado: Auto-inf");
560.                 lcd.setCursor(0,1);
561.                 lcd.print("VL:");
562.                 lcd.setCursor(3,1);
563.                 lcd.print(lv);
564.                 lcd.setCursor(8,1);
565.                 lcd.print("VD:");
566.                 lcd.setCursor(11,1);
567.                 lcd.print(dv);
568.                 break;
569.
570.             default:
571.                 Serial.println("  Warning: No recognized status!");
572.                 lcd.setCursor(0,0);
573.                 lcd.print("ERROR: ****** ");
574.                 lcd.setCursor(0,1);
575.                 lcd.print("VL:");
576.                 lcd.setCursor(3,1);
577.                 lcd.print(lv);
578.                 lcd.setCursor(8,1);
579.                 lcd.print("VD:");
580.                 lcd.setCursor(11,1);
581.                 lcd.print(dv);
582.                 break;
583.         }
584.
585.     }
```