UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR

POSGRADOS

MAESTRÍA EN

ELECTRÓNICA Y AUTOMATIZACIÓN

RPC-SO-30-No.507-2019

OPCIÓN DE TITULACION:

PROYECTO DE DESARROLLO

TEMA:

DESARROLLO DE UNA PLATAFORMA DIGITAL BASADA EN TELEMETRÍA E IOT PARA EL SEGUIMIENTO Y CONTROL REMOTO DE EQUIPOS IMPRESORES INDUSTRIALES.

AUTOR:

MILLER ALFREDO VILLAMAR BRAVO
GEOVANNY XAVIER GARCÍA FLOR

DIRECTOR:

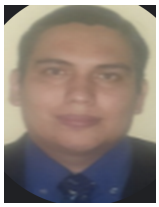
LUIS SILVIO CORDOVA RIVADENEIRA

GUAYAQUIL-ECUADOR
2022

COHORTE
2019- 2020

Autores:

Miller Alfredo Villamar Bravo.
Ingeniero Electrónico
Candidato a Magíster en Electrónica y Automatización, Mención en Informática Industrial por la Universidad Politécnica Salesiana - Sede Guayaquil.
mvillamar@est.ups.edu.ec



Geovanny Xavier García Flor.
Ingeniero Electrónico
Candidato a Magíster en Electrónica y Automatización, Mención en Informática Industrial por la Universidad Politécnica Salesiana - Sede Guayaquil.
ggarciaf@est.ups.edu.ec

Dirigido por:

Luis Silvio Córdova Rivadeneira.
Ingeniero en Electricidad con espcialización en Electrónica.
Magister en Telecomunicaciones.
lcordova@ups.edu.ec

DESARROLLO DE UNA PLATAFORMA DIGITAL BASADA EN TELEMETRÍA E IoT PARA EL SEGUIMIENTO Y CONTROL REMOTO DE EQUIPOS IMPRESORES INDUSTRIALES.

Índice general

Índice de Figuras

Índice de Tablas

Dedicatoria

Miller Alfredo Villamar Bravo

Este trabajo se lo dedico a mis hijos y esposa, quienes presenciaron las múltiples jornadas de trabajo que efectué y a quienes les quité tiempo para poderlo culminar.

Agradezco a mi esposa y padres quienes estuvieron siempre motivándome y apoyandome.

A Dios por permitírmelo y a una buena amiga por colaborarme CF.

Geovanny Xavier García Flor

Dedico este trabajo a Dios por todo lo que me ha prestado y este nuevo paso en mi vida es gracias a su voluntad. Se lo dedico a mi amada esposa Karen quien me impulso a continuar con mis estudios académicos. A mi madre quien siempre ha creído en mi, dándome ejemplo de superación, humildad y sacrificio, gracias por todo madre mía.

Agradezco a mi amigo y compañero Miller, por su dedicación y conocimiento en este trabajo.

Agradezco al Msc. Luis Córdova por su aporte y colaboración.

Encomienda a Jehová tus obras, y tus pensamientos serán afirmados.(Proverbios 16:3)

Resumen

En la actualidad existen muchas alternativas de telemetría, sin embargo, se propone desarrollar una plataforma para el seguimiento de equipos impresores con el propósito de contar con nuevas alternativas para sistemas de Industria4.0, mediante la implementación de hardware y software diseñado a la medida, para la recolección y presentación de parámetros de funcionamiento y operación; de tal manera que esta información se pueda monitorear y controlar con el objetivo de ejecutar acciones preventivas y correctivas oportunas en el proceso.

Se diseñaron tarjetas electrónicas para la recolección y envío de información correspondiente a datos de operación y funcionamiento de los equipos impresores en planta, seleccionando módulos de comunicación LoRa que realizan la comunicación por radio frecuencia. Los datos obtenidos fueron procesados a través de una plataforma digital realizada en el software Labview, donde se puede realizar el control de trazabilidad, el envío de datos a la plataforma web realizada en Thingsboard. Además, se diseñó el hardware para el control del sistema redundante que permite mantener en operación dos equipos impresores industriales en una misma línea de producción para reducir tiempos de parada no planeados de línea.

Se realizaron pruebas experimentales para determinar el alcance óptimo de los módulos de frecuencia sin tener perdida de información. En esta prueba concluimos que el tipo de antena tiene una participación importante en lo que respecta al alcance así como también la frecuencia de trabajo de los módulos lora.

Palabras clave: Industria 4.0 , IoT, LoRa, Diseño, Thingsboard.

Abstract

Many telemetry alternatives are currently available. However, it is proposed to develop a platform for tracking printing equipment with the purpose of having new alternatives for Industry4.0 systems, through the implementation of hardware and software designed to measure, for the collection and presentation of operating and operation parameters; so that this information can be monitored and controlled with the objective of executing preventive and corrective actions in a timely manner in the process.

Electronic cards were designed to collect and send information corresponding to the operation and functioning data of the printing equipment in the plant, selecting LoRa communication modules that communicate by radio frequency. The data obtained were processed through a digital platform made in Labview software, where traceability control can be performed, sending data to the web platform made in Thingsboard. In addition, the hardware was designed for the control of the redundant system that allows to maintain in operation two industrial printing equipment in the same production line to reduce unplanned downtime of the line.

Experimental tests were carried out to determine the optimal range of the frequency modules without any loss of information. In this test we concluded that the type of antenna has an important role in terms of range as well as the working frequency of the lora modules.

Key words: Industry 4.0, IoT, LoRa, Design, Thingsboard.

Capítulo 1

Introducción

En la actualidad, con lo que se ha denominado la cuarta revolución industrial o también conocida como industria 4.0 [Kalor et al., 2018, Bassi, 2017], aparecen nuevas tecnologías como la analítica de datos, inteligencia artificial e internet de las cosas (IoT). La tecnología IoT en la industria [Bauer et al., 2018] aprovecha el uso del Internet para conectar y recopilar datos de los procesos con distintos tipos de dispositivos como sensores, actuadores y controladores. [Zhou et al., 2021, Kada et al., 2019].

La implementación de la tecnología IoT ha permitido realizar mejoras en procesos de fabricación [Luque et al., 2017, Nakagawa et al., 2021] debido a la obtención de información para su análisis [Gulati et al., 2021].

En la industria alimenticia The Tesalia Springs Company como parte del proceso de fabricación, se encuentra el proceso de impresión en el producto, que consiste en imprimir la fecha de fabricación, fecha de caducidad, precio de venta al público y el lote de producción. La información sirve de trazabilidad para el fabricante e indica al consumidor final si el producto está vigente para su consumo. La información de trazabilidad debe ser impresa de forma legible sobre el producto, de lo contrario no puede ser comercializado según normativa INEN [INEN NTE INEN 1334-1:2011 TERCERA REVISION, 2011]. Por esta razón, la empresa presta mucha atención a este proceso de impresión y vela porque su producto cuente con la información impresa bajo la normativa. La falta o errores de trazabilidad de información en cada línea de producción conlleva al bloqueo o devolución del producto, siendo más crítico en las líneas denominadas de alta velocidad de producción [Garg et al., 2021]. La falta de un sistema centralizado de información de trazabilidad impresa, en el que se podría monitorear el estatus y controlar de forma remota la información que imprimirían.

1.1. Descripción general del problema

El proceso de impresión del producto por lo general está al final de la cadena de producción previo al paletizado (véase figura 1.1). Cada línea de envasado tiene al menos 1 equipo impresor al final del proceso. En el caso de la industria mencionada tiene 7 líneas de envasado por lo tanto cuenta con 9 equipos impresores incluidos los equipos de respaldo para sus líneas críticas.

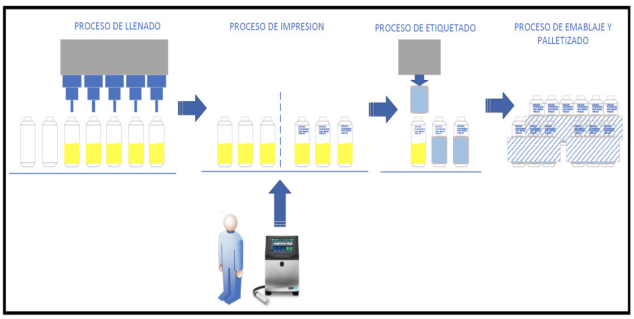


Figura 1.1: Línea de producción
Fuente: Los autores

En este proceso de impresión, existen errores frecuentes de digitación operacional el cual radica en que si el operador ingresa en el equipo impresor información de trazabilidad errónea entonces todos los productos fabricados tendrán error de trazabilidad. A esto se agrega el tiempo que implica la puesta en marcha del equipo de respaldo cuando se lo requiere, estos eventos representan para el fabricante reprocesos, pago de horas extras, desperdicio energético, retraso en la cadena de producción, etc. Esta maniobra implica tiempo que impacta de forma negativa el rendimiento de la línea, esto es debido a que las líneas de producción críticas de la planta no cuenta con un sistema de redundancia de equipos impresores, que consistiría en dos equipos impresores trabajando al mismo tiempo sobre la misma línea de producción (véase la Figura 1.2), es decir cada equipo trabajaría sobre el 50 % de la producción en ejecución y en conjunto harían el 100 %, con lo que se pudiera elevar el rendimiento de la línea, si uno de los equipos fallase el otro equipo tomaría su 50 % de trabajo y así la línea no es detenida mientras se revisaría el equipo averiado.

Mediante este proyecto de titulación se resolvería la falta de seguimiento y control de información como precio de venta al público, fecha de caducidad, fecha de fabricación, lote impreso en el producto, tomando en consideración

Figura 1.2: Sistema de redundancia propuestos usando equipos VideoJet
Fuente: Los autores

que al momento la empresa ya cuenta con los equipos impresores marca VideoJet modelo VJ1580 (véase la Figura 1.3) funcionando en sus líneas de producción, pero no cuenta con ninguna red inalámbrica para la interconexión de los equipos ni sistemas de redundancia. Las líneas de producción críticas para el cliente cuentan con su respectivo equipo de impresor de respaldo a un lado de la línea, permanece apagado hasta que el operador lo prepare para su uso en caso de que el equipo principal falle.

Figura 1.3: Equipo impresor basado en tecnología de tinta continua marca VideoJet
Fuente: Los autores

Con el desarrollo propuesto se pretendería lograr incrementar la productividad de las líneas de producción del cliente The Tesalia Springs Company reduciendo la probabilidad de errores operativos en lo concerniente al uso de los equipos impresores industriales marca Videojet, se desarrollaría

una plataforma centralizada basada en telemetría e IoT para el monitoreo y control básico de los equipos impresores, se conocería en tiempo quasi real el estatus actual de los equipos, así como también se podría incrementar la eficiencia de las líneas de producción críticas mediante el desarrollo del sistema redundante propuesto, el cual sería monitoreado desde la plataforma propuesta. Mediante esta redundancia de equipos se reduciría los tiempos de parada de las líneas de producción críticas, asegurando que la información impresa sea la correcta aplicando sistemas embebidos para la telemetría que trabajaran en conjunto con la plataforma de monitoreo propuesta, en conclusión contribuiría positivamente con la eficiencia de las líneas de producción.

1.2. Objetivos

1.2.1. Objetivo general

Desarrollar una plataforma digital para el seguimiento y control remoto de equipos impresores industriales mediante el uso sistemas embebidos y dispositivos de comunicación inalámbrica e IoT con aplicación de software.

1.2.2. Objetivos específicos

- Definir los requerimientos de hardware, software y de la plataforma digital, mediante el análisis de funcionamiento de los equipos impresores industriales para la recepción y envío de información mediante redes inalámbricas.
- Diseñar la interfaz gráfica basada en los requerimientos de la plataforma digital mediante la herramienta de software Labview para el seguimiento y control remoto de los equipos impresores.
- Implementar tarjetas electrónicas de control mediante sistemas embebidos y dispositivos de comunicación inalámbrica Lora, para la gestión de operación del sistema redundante y el envío / recepción de información de los equipos impresores industriales.
- Desarrollar una plataforma IoT para el seguimiento de estatus de los equipos impresores.

1.3. Métodos y técnicas

Para el desarrollo de la investigación se emplearan los siguientes métodos y técnicas:

1.3.1. Método experimental

Se empleará el método experimental para el desarrollo de prototipos de tarjetas electrónicas con los elementos de hardware seleccionados (dispositivos electrónicos, terminales, conectores, carcasas), que permiten la adquisición y envío de datos obtenidos de los equipos impresores industriales por telemetría a las plataformas digitales a desarrollar.

1.3.2. Técnica Documental

Se empleará para la adquisición de información en documentación, aplicaciones , teorías de cada uno de los elementos a emplear en el desarrollo del proyecto final y para el desarrollo del marco teórico.

1.3.3. Técnica Observación

Se empleará para la ayuda en el desarrollo de la investigación mediante la visualización de hechos, acciones, objetos.

1.3.4. Técnica Experimental

Se empleará para la verificación de nuevos conocimientos adquiridos con la técnica documental.

1.3.5. Técnica de Test

Se empleará para la comprobación de los resultados de la experimentación a realizar.

1.3.6. Técnica de diseño

Se utilizará para la aplicación de conocimientos que se hayan adquiridos durante la ejecución de la investigación en el desarrollo del proyecto.

1.4. Metodología

A continuación, se describen cada uno de los objetivos, así como el conjunto de procedimientos a realizar para su cumplimiento:

- **OE1:** Definir los requerimientos de hardware y software de la plataforma digital, mediante el análisis de funcionamiento de los equipos impresores industriales para la recepción y envío de información mediante redes inalámbricas. Para el cumplimiento de este objetivo en primera instancia se recolectaría información sobre los diferentes productos que se fabrican en la línea de más alta producción. Los equipos impresores industriales cuentan con protocolos de comunicación serial como es el RS232, el cual se usaría para comunicarse con el hardware a implementar. Este hardware para cada equipo se programaría para la comunicación con los equipos impresores para enviar y recibir información desde y hacia la plataforma digital.

- **OE2:** Diseñar la interfaz gráfica basada en los requerimientos de la plataforma digital mediante la herramienta de software Labview para el seguimiento y control remoto de los equipos impresores. La plataforma digital se diseñaría mediante el software Labview, tendría la capacidad de enviar y recibir información de trazabilidad mediante el hardware implementado para comunicación inalámbrica. También mostraría el estatus de cada equipo impresor conectado a la red.

- **OE3:** Implementar tarjetas electrónicas de control mediante sistemas embebidos y dispositivos de comunicación inalámbrica Lora, para la gestión de operación del sistema redundante y el envío / recepción de información de los equipos impresores industriales. Mediante la implementación de la comunicación inalámbrica basada en sistemas embebidos y módulos de comunicación Lora para el control y seguimiento del sistema redundante en las líneas de producción críticas de la empresa. Así también se emplearía las tarjetas de control electrónicas para el envió y recepción de información a la plataforma digital diseñada en el software Labview, serían colocadas en cada uno de los equipos impresores industriales para formar la red de comunicación Lora. Así como una tarjeta electrónica de control central para la recepción de información de cada uno de los equipos que conforman la red como se muestra a continuación en la 1.4:

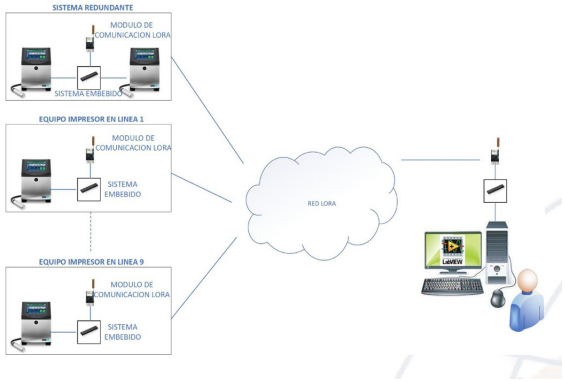


Figura 1.4: Red de comunicación LoRa
Fuente: Los autores

- **OE4:** Desarrollar una plataforma IoT para el seguimiento de estatus de los equipos impresores. Mediante la recolección de información proveniente de la red inalámbrica (véase figura 5), se desarrollaría una plataforma IoT con la cual se daría seguimiento o visualizaría el estatus de los equipos impresores comunicados a la plataforma digital desarrollado en el software Labview, el cual se encargaría del envió de información hacia la plataforma IoT.

  Con la ayuda del hardware de la tarjeta electrónica máster a implemnetarse, la plataforma digital podría consultar equipo por equipo la información y la presentaría en la plataforma digital, en donde se podría visualizar y controlar la información de trazabilidad. Debido a que la información de trazabilidad es sensible, no debería ser controlada desde la plataforma web, es por esta razón que solo la plataforma digital podría realizar cambios en la información de trazabilidad si el cliente así lo requiere y a la plataforma web serían enviados datos únicamente para el monitoreo. El acceso al cambio de información de trazabilidad estaría restringido por contraseña para evitar que por error se envíe información incorrecta, a continuación en la 1.5 se muestra el esquema general del sistema:

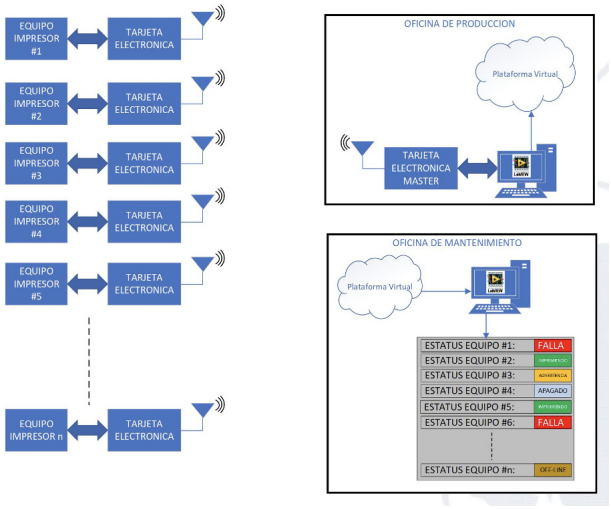


Figura 1.5: Esquema general del sistema
Fuente: Los autores

La interfaz a realizar en Labview mostraría la información de trazabilidad que se está imprimiendo y además permitirá conocer el nombre de la receta cargada, así como también el estatus del equipo, como se muestra a continuación en la 1.6:

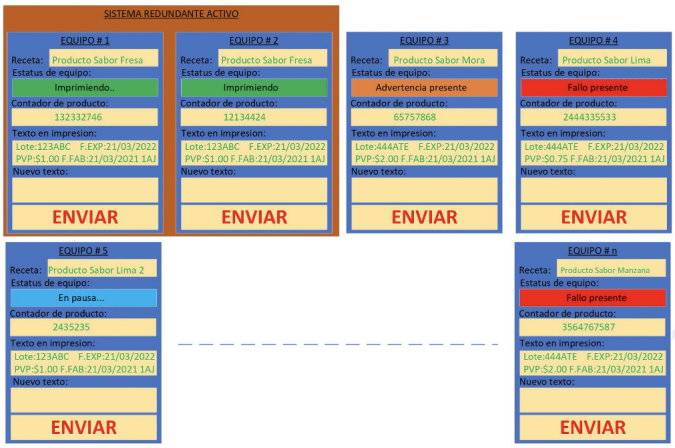


Figura 1.6: Interfaz estimada a realizar en Labview
Fuente: Los autores

Además, permitiría digitar la información de trazabilidad que se desea que el o los equipos impriman. El software gestionaría el envió de información hacia cada uno de los equipos impresores. Parte de la información que recolectaría el software, sería subida a la plataforma web para que sea presentada, con el fin de que el departamento correspondiente tenga acceso a esa información y pueda actuar de manera preventiva ante los posibles eventos que se presenten.

La plataforma web mostraría información concerniente a la producción, tal como el nombre de la receta cargada, estatus del equipo, productos impresos (batch) con la receta escogida, contador total de impresiones, y la cantidad de advertencias y/o fallas junto con su respectiva descripción. Podría también advertir al usuario sobre el próximo abastecimiento de fluidos necesario para que sea realizado de manera oportuna. Adicional, se añadiría un historial de fallos mensual en la cual el técnico podría evaluar el rendimiento de los equipos. Finalmente, podría visualizarse el comportamiento del sistema redundante y observar que equipo es el que esta trabajando más en un determinado tiempo, como se muestra a continuación en la 1.7:

Figura 1.7: Interfaz estimada de plataforma web
Fuente: Los autores

1.5. Organización del manuscrito

En el **Capítulo 1** se describen los hechos preliminares como introducción, descripción general del problema, objetivos general y específicos, metodología y técnicas empleadas para el desarrollo del proyecto.

En el **Capítulo 2** se realiza el marco teórica sobre los temas principales del proyecto, donde se describen los elementos utilizados que forman parte del desarrollo del proyecto.

En el **Capítulo 3** se utilizan las técnicas y procedimientos para el desarrollo del proyecto, describiendo los circuitos electrónicos utilizados, diagramas y diseños de las PCB, realización de algoritmos para el control del sistema redundante y equipos remotos,desarrollo de las plataformas digitales y web, resultados y conclusiones.

Capítulo 2

Marco teórico

2.1. Antecedentes

En la industria alimenticia The Tesalia Springs Company como parte del proceso de fabricación, se encuentra el proceso de impresión en el producto, que consiste en imprimir la fecha de fabricación, fecha de caducidad, precio de venta al público y el lote de producción. La información sirve de trazabilidad para el fabricante e indica al consumidor final si el producto está vigente para su consumo. La información de trazabilidad debe ser impresa de forma legible sobre el producto, de lo contrario no puede ser comercializado según normativa INEN. Por esta razón, la empresa presta mucha atención a este proceso de impresión y vela porque su producto cuente con la información impresa bajo la normativa. La falta o errores de trazabilidad de información en cada línea de producción conlleva al bloqueo o devolución del producto, siendo más crítico en las líneas denominadas de alta velocidad de producción. Este problema se ocasiona por la falta de un sistema centralizado de información de trazabilidad impresa por los equipos, en el que se podría monitorear el estatus y controlar de forma remota la información que estarían imprimiendo.

2.2. Telemetría

2.2.1. Definición

Proceso de recolectar , procesar , enviar datos de distintos puntos ya sean estos del tipo discreto (alarmas) , puntos analógicos (señales de voltaje, corriente) hacia el punto de recepción de datos centralizado para el análisis,

tratamiento , control, etc.

2.2.2. Telemetría de radio

En aplicaciones de telemetría con grandes distancias, normalmente se selecciona la transmisión de la señal por radio frecuencia. Según la ubicación geográfica existen regulaciones para el uso del espectro radioeléctrico, por lo que los equipos y aplicaciones deben estar adecuadas para operar en una banda de telemetría, como 890 MHz A 960MHZ (UHF).

2.2.3. Telemetría digital

La comunicación de datos mediante telemetría no solamente se puede realizar con métodos analógicos de codificación, también puede ser usado en un enfoque digital. La diferencia entre ambos métodos de transmisión,con el método digital es necesario enviar una tren de pulsos para la transmisión del valor de una variable medida. La ventaja de la codificación digital posee una mejor inmunidad al ruido o interferencias.

2.3. Comunicación inalámbrica

2.3.1. Definición

La comunicación inalámbrica no emplea medio de propagación físico (cables o conductores),para la transmisión emplea la modulación ondas electromagnéticas como medio de comunicación. Comúnmente, la comunicación inalámbrica se realiza en ondas de radiofrecuencia de baja potencia y con un ancho de banda específico. [Blázquez, 2015]

2.3.2. Clasificación

La clasificación de las comunicaciones inalámbricas según su alcance y acceso en la red se pueden agrupar por:

- Redes de área personal inalámbrica (WPAN).
- Redes de área local inalámbrica (WLAN)
- Redes de área extendida inalámbrica (WWAN)

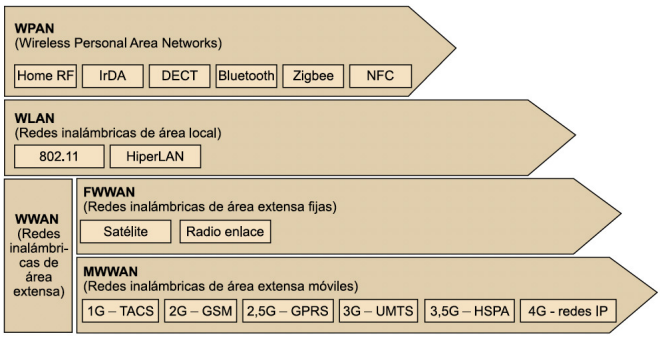


Figura 2.1: Redes inalámbricas
Fuente: [Blázquez, 2015]

2.4. Industria 4.0

2.4.1. Introducción

En la actualidad con la evolución de las tecnologías de la información (TI), la digitalización que se a experimentado en las comunicaciones y el conocimiento que han conllevado a grandes avances en la mayor partes de la actividades humanas, lo que se la a denominado "sociedad digitalz su conectividad a propiciado la cuarta revolución industrial(Industria 4.0).[Garrell]

2.4.2. Definición

El concepto de industria 4.0 hace referencia a una nueva manera de obtener información de los procesos o ciclos del producto a lo largo de su fabricación que permita realizar un modelo mejorado de control y organización, apoyado por las tecnologías de la información.[del Val Román, 2016]

En la industria 4.0 algunas de las tecnologías presentes son:

- Sistemas ciberfísicos (CPS).
- Ciber seguridad
- Blockchain
- Inteligencia artificial

- Machine learning (aprendizaje autónomo)
- Deep Learning
- Big data
- Internet de las cosas (IoT)

2.4.3. Sistema Ciberfísicos (CPS)

Los sistemas CPS se fundamentan en modelado matemático e informático para el diseño y simulación de sistemas integrados en tiempo real, constituyen un nuevo campo de aplicación en la ingeniería.[Rozo-García, 2020]

2.4.4. Ciberseguridad

Con la digitalización de procesos en la industria para el envío de datos de control e información de la cadena de producción de las capacidades digitales de la industria 4.0 y las ventajas que representan también trae consigo nuevos riesgos cibernéticos para lo que se realizan conjuntos de tecnologías, procesos, practicas, implementadas y diseñadas para el riesgo que representa en el ciberespacio el almacenar, enviar, procesar información de organizaciones y personas en la campo tecnológico y procesos. La ciberseguridad es un punto crítico que debe de ser considerado en la planificación y estrategias en el empleo de las tecnologías de la industria 4.0.[Carrasco and Puerta, 2013]

2.4.5. Blockchain

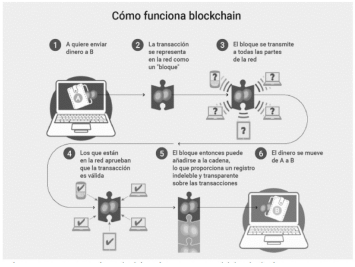


Figura 2.2: Funcionamiento de Blockchain
Fuente: [Nofer et al., 2017]

Blockcahin o cadena de bloques se la denomina como la segunda era del internet. Mediante esta tecnología la infomración obentida es codificada a traves de procesos llamados criptografía [Nofer et al., 2017].

2.4.6. Inteligencia artificial

La inteligencia artificial contiene campos como la lógica , matemática, ciencias de la computación e información, estadística, biología, psicología y mas áreas, se la denomina como una rama de conocimiento multidisciplinar. Sus aplicaciones son diversas y de gran alcance por lo que en la actualidad esta siendo empleada para plantear y resolver problemas en áreas de investigación científica e industrial.[Rouhiainen, 2018]

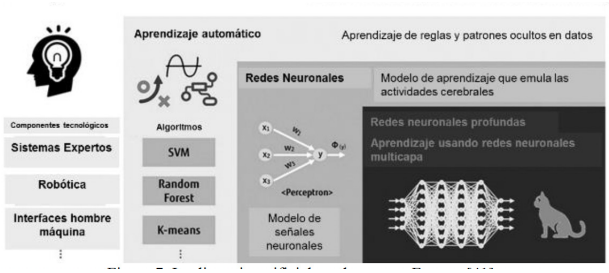


Figura 2.3: Inteligencia artificial
Fuente: https://journal.jp.fujitsu.com/en/2018/11/29/01/

2.4.7. Machine learning

Machine learning o aprendizaje autónomo es una rama de la inteligencia artificial , emplea los datos para ingresar a algoritmos que puede ser capaz de interpretar la relación entre la entrada y salida del sistema que se encuentra bajo análisis para poder predecir, clasificar, generar conocimiento. En el aprendizaje autónomo se aprovecha los avances tecnológicos de las ciencias computacionales, capacidad de computo, almacenamiento y envió de información.[Mitchell et al., 1990]

2.5. Internet de las cosas (IoT)

2.5.1. Introducción

Los avances tecnológicos en la arquitectura de la información basa en la internet, da acceso al intercambio de servicios, bienes entre los elementos , objetos , equipos enlazados a la red. En el internet de la cosas se encuentran desarrollando aplicaciones para la comunicación entre dispositivos que se comunican electrónicamente para la adquisición y envío de la información adquirida hacia el internet.

2.5.2. Definición

El internet de las cosas en general es la conexión a la red de objetos , dispositivos , sensores que se encuentren equipados para poder comunicarse al internet.[Salazar and Silvestre, 2016] Con esta nueva tecnología también surgen nuevas necesidades como es el protocolo de internet versión 6 IPv6.

2.5.3. Protocolo de internet versión 6

En la comunicación entre la red y ordenadores , dispositivos inteligentes se realizan mediante de protocolos de internet , para el IoT es necesario el uso de la última versión de IP. EL IPv6 otorga protocolos de comunicación mediante sistemas de identificación, rutas de tráfico, ubicación de equipos en la red. IPv6 emplea un formato de dirección de 128 bits. La cabecera IPv6 tiene el siguiente formato que se muestra en la figura 2.4:

Figura 2.4: Cabecera IPv6
Fuente: [Salazar and Silvestre, 2016]

2.6. Sistemas Embebidos

2.6.1. Definición

Los sistemas embebidos son controlados por microprocesadores o microcontroladores en los cuales se crean sistemas operativos para cubrir necesidades específicas y no generales como un computador, en la figura 2.5 se muestra un ejemplo de sistema embebido.

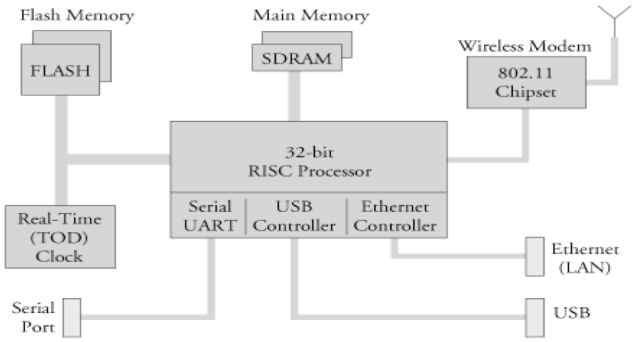


Figura 2.5: Sistema embebido
Fuente: [Pérez, 2009]

Su funcionamiento en términos generales esta constituido por:

- Entrada (sensores, periféricos)
- Proceso (tiempo real)
- Salida (respuestas, resultados, periféricos)

2.6.2. Diseño

Los sistemas embebidos en general están constituidos por una parte de software y hardware.

- Hardware: es la parte física del sistema como tarjeta de control, sensores, etapa de potencia y otros elementos electrónicos que lo conformen.
- Software: es la capa principal de código que se ejecuta en el controlador

El diseño de un sistema embebido normalmente esta orientado a:

- Realizar la mayor cantidad de tareas posibles.
- Aumentar el rendimiento y confiabilidad.
- Garantizar determinismo y tiempo de respuesta.
- Reducción del tamaño, consumo y costo.

2.7. ESP32

2.7.1. Definición

El ESP32 es un microcontrolador fabricado por la empresa ESPRESSIF, en la actualidad son uno de los chips de mayor uso para el desarrollo de prototipos de aplicación en IoT debido a las características del dispositivo embebido que incluyen los siguientes periféricos y módulos de comunicación:

- CPU y memoria : Xtensa® single-/dual-core 32-bit LX6 microprocesador.
- Wi-Fi: 802.11 n (2.4 GHz), up to 150 Mbps
- Bluetooth: Bluetooth v4.2 BR/EDR y Bluetooth LE
- Periféricos: SPI, I2C, ETHERNET MAC, PWM, Hall sensor, UART, I2S.

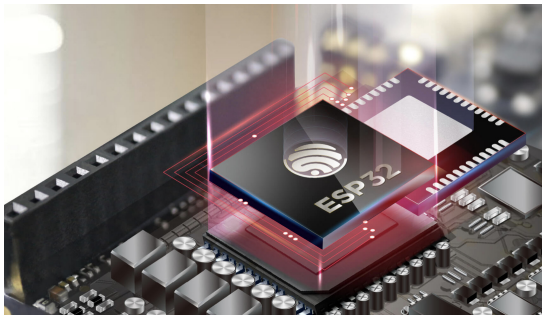


Figura 2.6: Microcontrolador ESP32
Fuente: https://www.espressif.com/en/products/socs/esp32

2.7.2. Diagrama de bloque

Como se puede observar en la figura 2.7, detalla la distribución de los diferentes periféricos, memorias, transmisores de radio frecuencia, módulo Wi-Fi, Bluetooth, donde destaca el CPU con microprocesadores de 32 bits dual core.

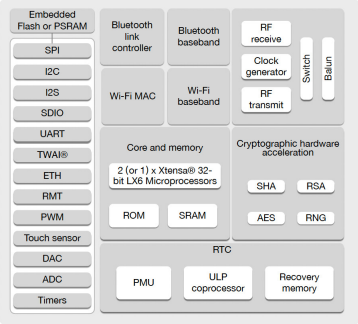


Figura 2.7: Diagrama de bloque ESP32
Fuente: [ESP, 2021]

2.7.3. Disposición de pines ESP32

En la figura 2.8 se detalla la distribución de los pines en el chip ESP32 en vista superior.

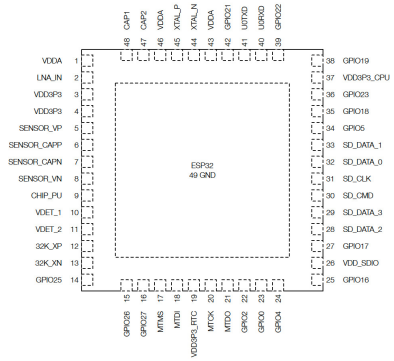


Figura 2.8: Disposiciones de pines ESP32
Fuente: [ESP, 2021]

2.7.4. Aplicaciones

A continuación se detallan algunas de la aplicaciones donde se puede implementar el ESP32:

- Sensores IoT de bajo consumo.
- Data loggers IoT de bajo consumo.
- Reconocimiento de imágenes.
- Automatización del hogar: control de luces, conectores inteligentes.
- Edificaciones inteligentes: monitoreo de energía, iluminación inteligente.
- Automatización industrial: control industrial inalámbrico, robótica industrial.

2.8. Comunicación inalámbrica LoRa

2.8.1. Definición

LoRa proviene de la abreviación Long Range (largo alcance), es una tecnología empleada para la comunicación inalámbrica entre dispositivos de usos específicos, como los dispositivos utilizados en IoT. La tecnología LoRa permite la transmisión y recepción de datos punto a punto, una de las principales características de un dispositivo con comunicación LORA es el largo alcance.

2.8.2. Características

Lora utiliza la técnica del espectro ensanchado, lo que significa es que la señal a enviar más ancho de banda de lo que se requiere en la comunicación, lo cual da acceso a recepción de múltiples señales a la vez que tengas distintas velocidades.

Entre las principales características de la comunicación LoRa es su largo alcance y en condiciones favorables pueden llegar a los 20 kilómetros.

2.8.3. Frecuencia de trabajo

Las frecuencias de trabajo que se emplea en la comunicación LoRa son las bandas ISM, la cuales son bandas de uso no comercial para las áreas industriales, científicas y médicas, sin embargo la tecnología puede trabajar en cualquier frecuencia por debajo de 1Ghz.

Las bandas de frecuencia que emplea LoRa son las 433 MHz, 866 MHz y 915 MHz la cual solo se emplea en América.

2.8.4. Funcionamiento

Los canales que posee un canal de comunicación LoRa son:

- Canal: Identifica los canales que se emplearan dentro de la banda de frecuencia.
- Spreading factor (SF): determina el numero de bits empleados para codificar un símbolo.
- Coding rate (CR): Indica la forma de codificar para la corrección de errores.
- Bandwith (BW): Indica el ancho de banda que se va a utilizar en la comunicación.

2.9. Convertidor UART a ETHERNET USR-TCP232-T2

Figura 2.9: USR-TCP232-T2
Fuente:
https://www.pusr.com/download/M0/USR-TCP232-T2-User-Manual-V1.1.pdf

El USR-TCP232-T2 es un módulo que realiza la transmisión transparente bidireccional de datos entre el puerto TTL y el puerto RJ45, también se puede usar en RT232/ RS485 por circuito de cambio de nivel.Está equipado con núcleo Cortex-M0. Tiene caracteres de baja potencia, alta velocidad, alta eficiencia, fuerte compatibilidad.

2.9.1. Características

- Soporta DHCP.
- Soporta DNS.
- Configuración de parámetros vía web.
- Actualización de firmware vía red.
- Modo de trabajo: Servidor TCP, cliente TCP, servidor UDP, cliente UDP, cliente HTTPD.
- Los parámetros relacionados con el modelo de trabajo se pueden configurar a través de un puerto serie o una red.
- Admite dirección MAC definida por el usuario.
- Restauración a valores de fábrica.
- Admite puerto serie virtual, software USR-VCOM.

2.9.2. Parámetros

Tabla 2.1: USR-TCP232-T2 parámetros

Parámetro	***Valor***
Voltaje	VCC: 3.3 VDC - 5 VDC
Corriente	130 mA
Corriente	<1W
Nivel serial	TTL
Puerto LAN	RJ45
Empaquetado	DIP
Temperatura	Trabajo : -25 75 C
Humedad	5 % 95 % Hr

2.9.3. Definición de pines

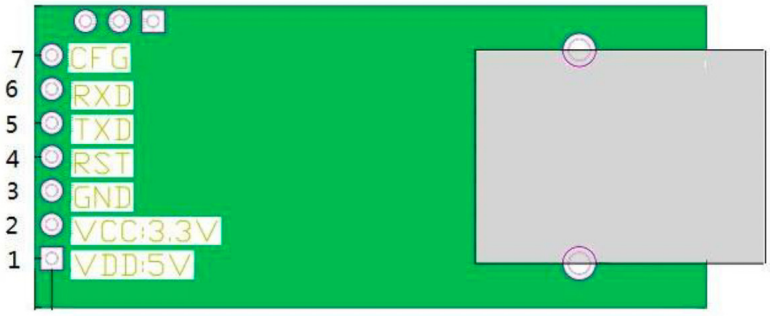


Figura 2.10: USR-TCP232-T2 pines
Fuente:
https://www.pusr.com/download/M0/USR-TCP232-T2-User-Manual-V1.1.pdf

Tabla 2.2: USR-TCP232-T2 pines

No	***Pin***	***Funcion***
1	VDD	Voltaje de fuente
2	VCC	Voltaje de fuente
3	GND	Tierra
4	RST	Reset
5	TXD	Transmisión de datos
6	RXD	Recepción de datos
7	CFG	Pin de configuración de la módulo

El consumo de corriente del módulo normalmente esta en 200 mA, la comunicación serial maneja voltaje TTL 3.3 vdc, tanto en el transmisor como en el receptor.

2.10. Módulo Heltec Wi-Fi LoRa 32

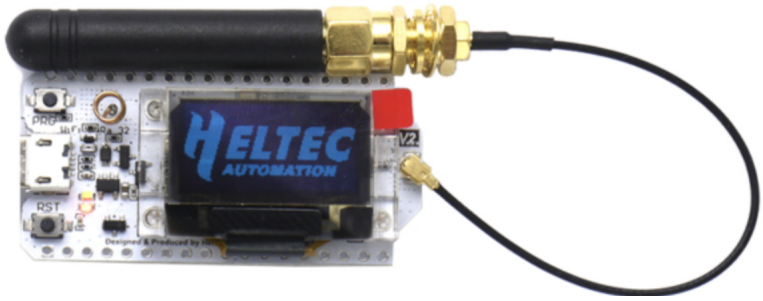


Figura 2.11: Heltec Wi-Fi LoRa 32
Fuente: https://heltec.org/project/wifi-lora-32/

2.10.1. Características

A continuación se detalla algunas de las características del módulo Heltec:

- Microprocesador: ESP32 (MCU de 32 bits de doble núcleo + núcleo ULP), con chip de nodo LoRa SX1276/SX1278.
- Interfaz micro USB con regulador de voltaje completo.
- Interfaz de batería SH1.25-2 integrada.
- Sistema de gestión de batería de litio integrado (gestión de carga y descarga.
- Wi-Fi integrado, LoRa, tres conexiones de red Bluetooth.
- Wi-Fi integrado, antena 3D de metal de 2,4 GHz dedicada a Bluetooth.
- Pantalla OLED de matriz de puntos de 128 * 64 de 0,96 pulgadas integrada.
- Chip integrado CP2102 USB a puerto serie.
- LoRaWAN estándar que puede comunicarse con cualquier puerta de enlace LoRa que ejecute el protocolo LoRaWAN.

- Diseño de circuito de RF y un diseño básico de baja potencia (corriente de reposo menor a 800uA), es conveniente para los proveedores de aplicaciones de IoT verificar soluciones e implementar aplicaciones rápidamente.

2.10.2. Definición de pines

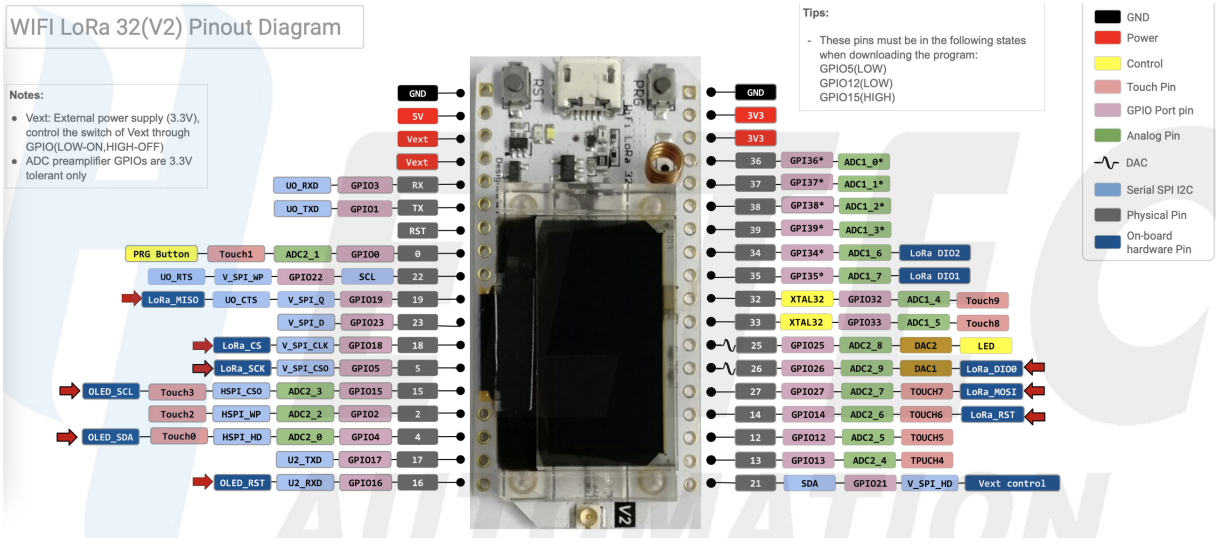


Figura 2.12: Pines Heltec Wi-Fi LoRa 32
Fuente: https://heltec.org/project/wifi-lora-32/

2.11. Módulo Mega 2560 Pro

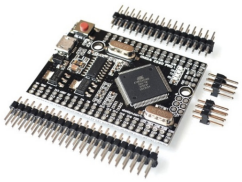


Figura 2.13: Mega 2560 Pro
Fuente: https://sandorobotics.com/producto/hs0993/

La placa Mega 2560 Pro es una tarjeta de desarrollo integrada, pequeña emplea el microcontrolador ATmega2560 (16 MHz) y el chip de interfaz USB-UART CH340G.

2.11.1. Características

- Microcontrolador: ATmega2560
- Chip USB: CH340G
- Voltaje de Operación: 5V
- Voltaje de alimentación: 7-9V (Mediante pin VIn y GND)
- Pines digitales I/O: 54 (15 salidas PWM)
- Entradas analógicas: 16
- Corriente máxima entrada/salida: 40mA
- Memoria FLASH: 256K

2.11.2. Definición de pines

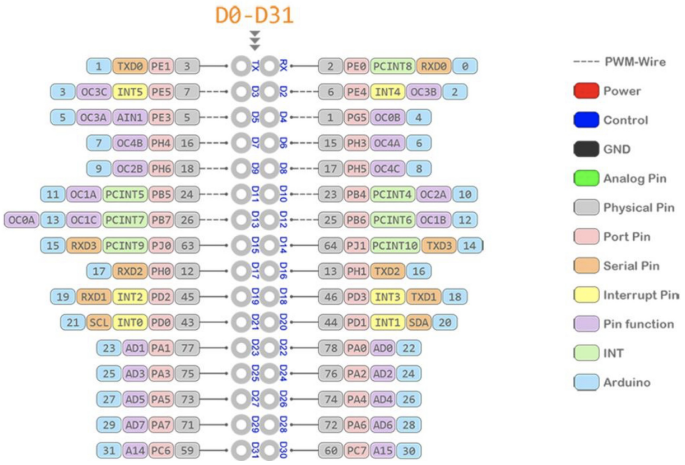


Figura 2.14: Pines Mega 2560 PRO
Fuente: https://www.enmindustry.de

2.12. Protocolo de comunicación Zipher

El protocolo Zipher desarrollado por el fabricante de impresoras industriales Videojet, determina la estructura y forma en la que los equipos impresores se comunican con otros dispositivos mediante los medios físicos RS232 o TCP/IP. Este protocolo abarca un conjunto de comandos definidos y específicos para la requisición y envío de información hacia el equipo. La información que se puede adquirir desde los equipos impresores, corresponden a datos de producción, estados, fallos y advertencias presentes en el equipo. Los comandos están formados por conjunto de caracteres o cadenas en formato ANSI (1 byte por carácter), al final de cada comando debe incluirse el carácter de retorno de carro (ANSI caracter codigo 13 decimal / 0x0D hexadecimal), también representado como <CR>. El carácter de separación de campos es 127 en ASCII correspondiente al carácter "|". A cada comando enviado correctamente, el protocolo determina la cadena ACK<CR>que indica que el comando se ha recibido con éxito y responde ERR<CR>ante un comando enviado de forma incorrecta. En la siguiente tabla se muestran varios de los comandos disponibles con su respectiva descripción.

Tabla 2.3: Comandos

Comando	***Descripcin***
SEL	SELECCIÓN DE TRABAJO
CAW	BORRAR ADVERTENCIAS
JDU	ACTUALIZACIÓN DE CAMPOS
GFT	OBTENER LAS FALLAS
PRN	IMPRIMIR
SST	DEFINE ESTADO IMPRESIÓN
GST	OBTENER ESTADO IMPRESORA
CAF	BORRA FALLAS
GWN	OBTENER ADVERTENCIAS
GTD	OBTENER TIEMPO Y FECHA
GPC	OBTENER CONTADOR
GJD	OBTENER CAMPOS DE TEXTO

Capítulo 3

Desarrollo

3.1. Introducción

En este capítulo se describirá el desarrollo de la plataforma digital basa en telemetría e Iot donde se a desarrollado el hardware y la aplicación del software requerido para la adquisición y envió de datos de los equipos impresores industriales.

3.2. Diseño del experimento

3.2.1. Esquema general

En la figura 3.1 se muestra el esquema de la plataforma digital para el procesamiento de datos y envió de manera inalámbrica de cada uno de los equipos impresores industriales de la s diferentes lineas de producción, para su posterior recepción y envío a la nube.

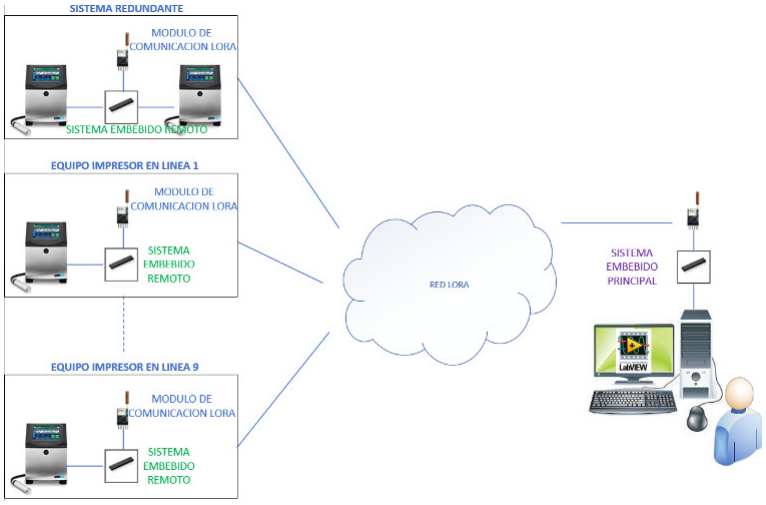


Figura 3.1: Esquema general
Fuente: Los autores

3.3. Desarrollo

3.3.1. Requerimiento equipos impresores industriales

El lenguaje de comunicación de los equipos impresores industriales de la marca Videojet es denominado Zipher, el cual indica los parámetros de comunicación que se deben seguir para una comunicación efectiva con los equipos impresores. Este protocolo consta de varias directivas y define los comandos a emplearse para la solicitud de información hacia los equipos impresores.

3.3.2. Comandos del protocolo Zipher

La primera regla para la comunicación indica que cada comando a enviarse debe terminar con un carácter Carrier return el cual indica al equipo el fin del comando enviado. En la siguiente tabla se describen los comandos usados extraídos de la extensa lista de comandos preestablecidos por el protocolo.

COMANDO	NOMBRE	R/W	DESCRIPCION
GST	GET PRINTER STATE	R	Obtiene el estado general del equipo. La sintaxis es GST<GR> y la respuesta STS contiene: estado general, estado de errores, tarea actual, contador de batch, contador global.
GWN	GET CURRENT WARNINGS	R	Enlista todas las advertencias presentes en la impresora. La sintaxis es GWN<GR> y la respuesta WRN contiene: cantidad total de advertencias, código de advertencia, borrable o no, descripción de la advertencia.
GFT	GET CURRENT FAULTS	R	Enlista todos los fallos presentes en la impresora. La sintaxis es GFT<CR> y la respuesta FLT contiene: cantidad total de fallos, código de fallos, borrable o no, descripción del fallo.
GJD	GET CURRENT JOB DATA	R	Obtiene la cantidad total de campos presentes en la receta actual. La sintaxis es GJD<CR> y la respuesta JDL contiene: cantidad de campos, nombre[s] de campo[s] concatenados.
SLA	JOB SELECT	W	Selecciona una receta rellena cada campo con información. La sintaxis es SLA<GR>. La respuesta es un ACK.

Figura 3.2: Comandos GST
Fuente: Los autores

El comando Get printer state (GST) solicita al equipo impresor el estado del mismo. La información que se obtiene como respuesta ante cada comando, debe ser tratada y acoplada a los requisitos establecidos. A continuación, se detalla el tratamiento aplicado a cada respuesta recibida:

- **Respuesta del comando GST:** la información recibida ante este comando tiene la siguiente estructura: STS|<estadogeneral>|<estadoerror>|<tarealactual><contadorbatch>|<contadorglobal>|<CR>

  De la respuesta recibida, se almacena únicamente los siguientes campos:

  - Estadogeneral: indica el estado del equipo el cual puede ser: 0 APAGADO, 1 ENCENDIENDO, 2 APAGANDO, 3 ONLINE y 4 OFFLINE.
  - Tarealactual: indica el nombre de la receta/trabajo que se esta ejecuntando.

  - Contadorbatch.: Indica el contador del batch correspondiente a la cantidad de productos impresos con la receta actual.
  - Contadorglobal.- Indica el contador histórico del total de impresiones.

- **Repuesta del comando GWN:** la información de advertencias tiene la siguiente estructura: WRN|<totalwarnings>|[<codigo>|<borrable>|<descripcion>|]<CR>De la respuesta recibida, se almacena únicamente el siguiente campo: Codigo.- este codigo de 6 digitos representa la advertencia actual en el equipo y si existiese mas de una advertencia, entonces los códigos y descripciones llegaran concatenados.

- **Respuesta del comando GFT:** la información de los fallos tiene la siguiente estructura: FLT|<totalfallos>|[<codigo>|<borrable>|<descripcion>|]<CR>De la respuesta recibida, se almacena únicamente el siguiente campo: Código.- este codigo de 6 digitos representa el fallo actual en el equipo y si existiese mas de un fallo, entonces los códigos y descripciones llegaran concatenados.

- **Respuesta del comando GJD:** la información de los campos tiene la siguiente estructura:

  JDL|<cantidadcampos>|[<camponombre>=<valor>|]<CR>De la respuesta recibida, se almacena únicamente el siguiente campo: <camponombre>=<valor>.- indica el nombre del campo y también su valor. Esta información representa la trazabilidad que se esta imprimiendo en ese instante.

- **El comando SLA:** es un comando de escritura solamente, es decir, mediante este comando se enviara hacia los equipos la información de trazabilidad que deseamos el equipo imprima. Para lo cual, los campos de fecha de fabricación y caducidad también están codificados.

3.4. Hardware del sistema embebido

La tarjeta electrónica desarrollada e instalada junto a cada dispositivo impresor, consta de la siguiente estructura:

- **Power supply +5Vcc:** módulo step down para regulación y filtrado de voltaje dc, en su entrada se provee +9vcc 2A provenientes de

un adaptador de energía, posteriormente el módulo step down lo reduce a +5 y mantiene estabilizado, este voltaje alimenta a los demás dispositivos instalados.

- **ESP32 LORA V2:** módulo de comunicación inalámbrica LoRa ajustado a 868MHZ el cual tiene grabado una dirección única, se lo usa para transmitir y recibir información vía RF hacia y desde la tarjeta principal. Este módulo mantiene almacenado la información (del equipo impresor) empaquetada y codificada lista para ser enviada cuando se le haga la consulta respectiva.

- **Mega 2560:** módulo encargado de la comunicación con el equipo impresor. Este módulo tiene grabado los comandos requeridos para la comunicación, dichos comandos son enviados periódicamente hacia el equipo impresor para así obtener las respectivas respuestas. Una vez que se han completado las consultas y se han recibido las respuestas, el módulo procesa y empaqueta la información y la transmite hacia el módulo ESP32 LoRa.

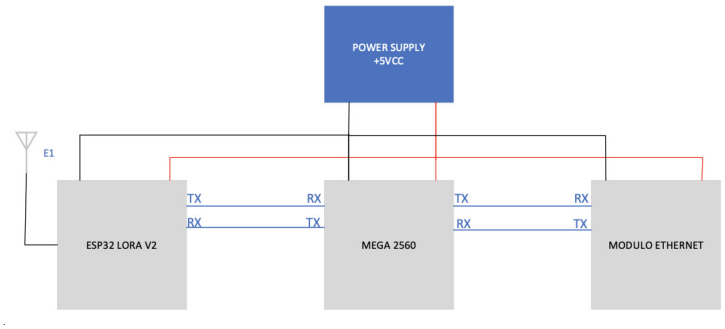


Figura 3.3: Sistema embebido
Fuente: Los autores

- **Módulo Ethernet:** Este módulo adapta la conexión Ethernet del equipo impresor de manera que el módulo Mega2560 mediante uno de sus puerto serie se pueda comunicarse con el equipo impresor.

Tanto el módulo ESP32 como el MEGA 2560 tienen internamente grabado su firmware, el cual esta orientado al manejo de información y procesamiento.

Figura 3.4: Sistema embebido
Fuente: Los autores

3.4.1. Hardware del sistema redundante

A continuación se muestra los diagramas electrónicos realizados en el software PROTEUS 8.11 de cada etapa del hardware:

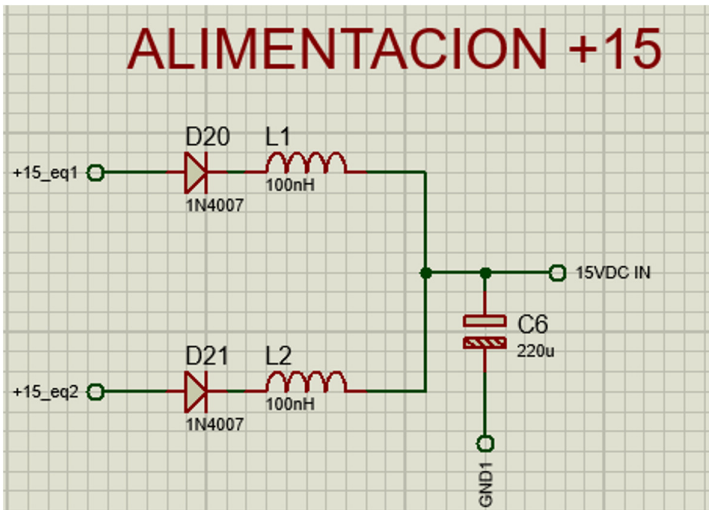


Figura 3.5: Alimentación 15 VDC
Fuente: Los autores

Cada equipo, alimenta al sistema redundante con +15Vcc. Este voltaje es usado para: alimentación del sensor de producto, alimentación del sensor redundante, orden de trabajo para equipo 1, orden de trabajo para equipo 2.

El hardware implementado permite la selección del tipo de señal de sensor a usarse , es decir si la señal es positiva PNP o negativa PNP Esto se logra cambiando los jumpers a la estrada de cada optoacopldor de manera que se puede tener una señal de entrada tipo PNP y otra NPN si se quisiera. El optoacoplador PC817 implementado fue seleccionado en base

a sus características de eléctricas de tiempo de respuesta,la velocidad de conmutación debe ser lo suficientemente rápida para evitar perder pulsos entrantes. La velocidad de conmutación del PC817 es de alrededor de 18us max, ver diagrama esquemático en la figura 3.6.

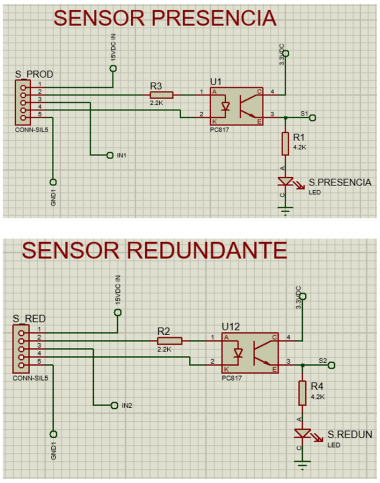


Figura 3.6: Sensor de presencia y redundante
Fuente: Los autores

Para la lectura de estados de cada equipo, hacemos uso del mismo tipo de optoacoplador. En este caso, los estados de cada máquina se determinan por la presencia / ausencia del 0v de cada equipo impresor,ver diagrama esquemático en la figura 3.7.

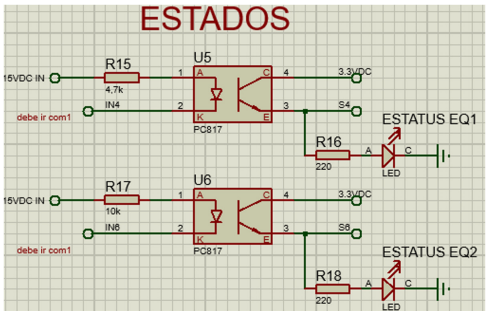


Figura 3.7: Lectura de estados
Fuente: Los autores

Al igual que la detección de los estados, los pulsos de confirmación son detectados por la presencia de 0v en el cátodo del diodo del optoacoplador. El estado de cada equipo se visualiza físicamente en la tarjeta electrónica por medio de los diodos led llamados: P-CONF EQ1 y P-CONF EQ2,ver diagrama esquemático en la figura 3.8.

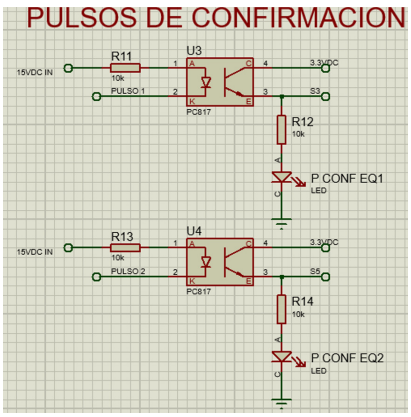


Figura 3.8: Pulsos de confirmación
Fuente: Los autores

ESP32 encargado de gestionar la comunicación inalámbrica , las señales electrónicas provenientes de los sensores, estados, pulsos de confirmación y las ordenes de impresión, ver diagrama esquemático en la figura 3.9.

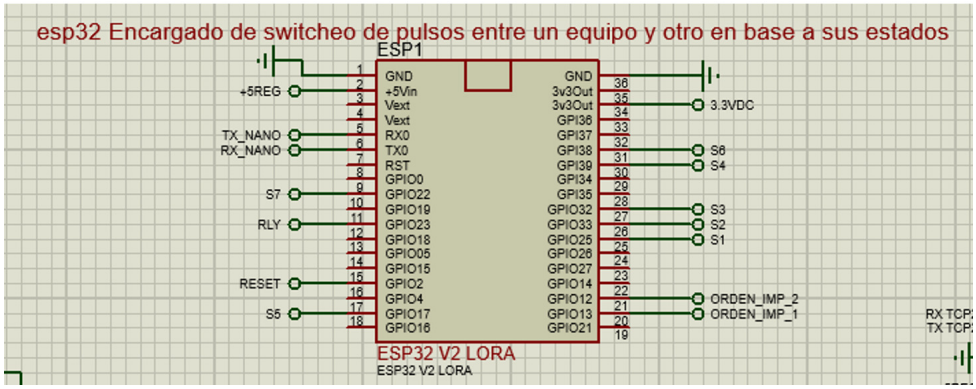


Figura 3.9: ESP32 V2 LoRa
Fuente: Los autores

Mega2560 encargado de la comunicación con los 2 equipos impresores. Se encarga de obtener la información de cada máquina para posteriormente enviarla al ESP32. Este uC se comunica con los equipos impresores mediante los módulos de comunicación TCP-Serial, ver diagrama esquemático en la figura 3.10.

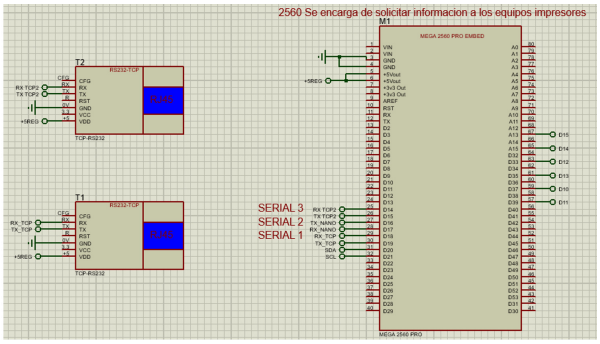


Figura 3.10: Mega 2560 Pro
Fuente: Los autores

3.4.2. Diseño de PCB del sistema redundante

El diseño de la PCB del sistema redundante fue realizado en el software de diseño ARES-PROTEUS, en la figura 3.11 se muestra el diseño.

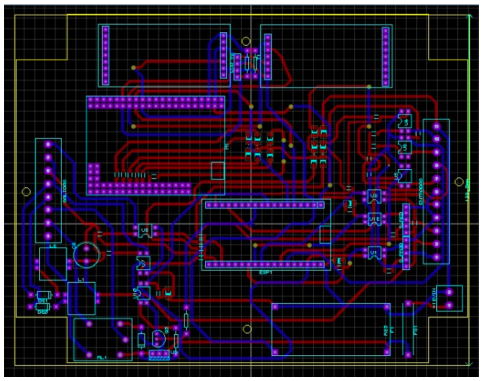


Figura 3.11: Diseño PCB sistema redundante
Fuente: Los autores

3.4.3. Diseño de PCB equipo remotos

El diseño de la PCB de equipos remotos fue realizado en el software de diseño ARES-PROTEUS, en la figura 3.12 se muestra el diseño.

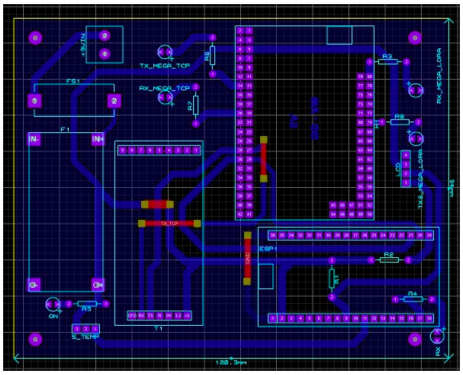


Figura 3.12: Diseño PCB equipos remotos
Fuente: Los autores

3.4.4. Fabricación de PCB sistema redundante

En la figura 3.13, se describe la disposición de cada uno de los elementos empleados para el sistema de control redundante.

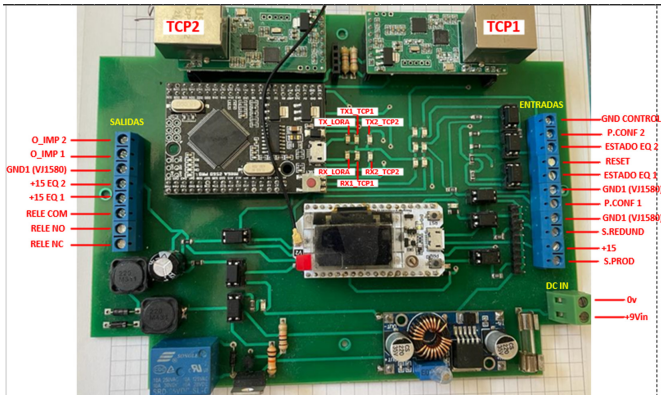


Figura 3.13: PCB sistema redundante
Fuente: Los autores

3.4.5. Fabricación de PCB equipos remotos

La tarjeta electrónica fue diseñada para la adquisición de información y envío de datos remotos de cada uno de los equipos impresores industriales instalados en las lineas de producción, ver PCB en la figura 3.14.

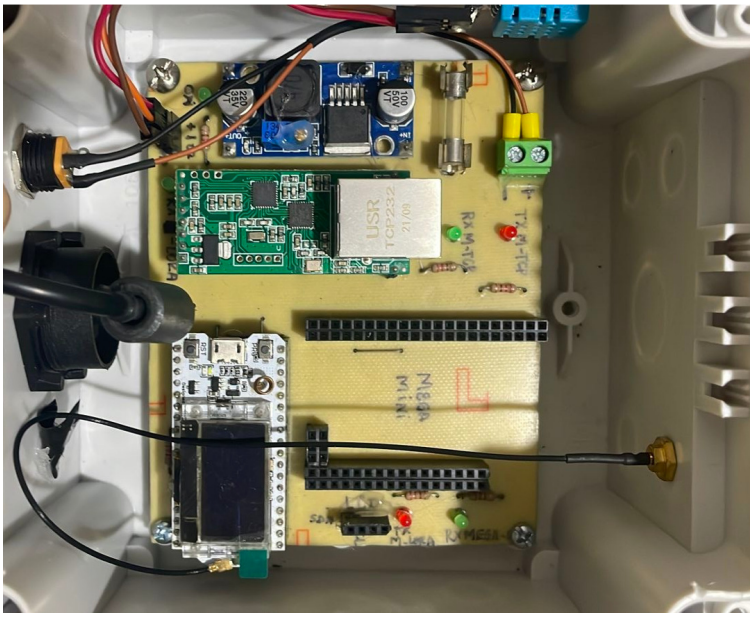


Figura 3.14: PCB equipos remotos
Fuente: Los autores

3.5. Desarrollo del código de programación

3.5.1. Programación Mega 2560

El microcontrolador Atmega2560 fue escogido debido a su característica puntual de contar con varios puertos UART necesarios en este proyecto para la comunicación con el módulo LoRa y módulo Ethernet. El firmware esta orientado a realizar una serie de consultas al equipo impresor, esperar la respuesta, codificar los datos y concatenarlos en un solo string el cual es retransmitidos al módulo lora frecuentemente. A continuación, en las líneas de código empleadas para las consultas al equipo impresor usando los distintos comandos indicados previamente.

```
1
2  ////////////////////------CONSULTAS--------//////////////////////
3  //Consulta del ESTADO
4  if (((tiempo_actual - previous_gst) > intervalo1 && llave1 ))
5  {
6    digitalWrite(TX_MEGA_LED, HIGH);
7    llave1 = false;
8    Serial2.print('\r');
9    Serial2.print("GST\r");
10   digitalWrite(TX_MEGA_LED, LOW);
11 }
12
13 //CONSULTA DEL WARNINGS
14 if (((tiempo_actual - previous_wrn) > intervalo2 && llave2 ))
15 {
16   digitalWrite(TX_MEGA_LED, HIGH);
17   llave2 = false;
18   Serial2.print('\r');
19   Serial2.print("GWN\r");
20   digitalWrite(TX_MEGA_LED, LOW);
21 }
22
23 //COMSULTA DE LOS CAMPOS
24 if (((tiempo_actual - previous_gjd) > intervalo3 && llave3 ))
25 {
26   digitalWrite(TX_MEGA_LED, HIGH);
27   llave3 = false;
28   Serial2.print('\r');
29   Serial2.print("GJD\r");
30   digitalWrite(TX_MEGA_LED, LOW);
31 }
32
33 //CONSULTA DE FALLAS
34 if (((tiempo_actual - previous_gft) > intervalo4 && llave4 ))
35 {
36   digitalWrite(TX_MEGA_LED, HIGH);
37   llave4 = false;
38   Serial2.print('\r');
39   Serial2.print("GFT\r");
40   digitalWrite(TX_MEGA_LED, LOW);
41 }
```

Todas las consultas son realizadas mediante el uso de interrupciones por software, mediante este método se logra fluidez en el código. Para esto se hace uso de variables booleanas (entre otro tipo de variables) para la activación y desactivación de las consultas.

La “variables tiempo actual”, indica el tiempo recorrido desde el inicio de la ejecución del código. Las variables llamadas previus-gst, previus-wrn,

previous-gjd, etc) registran el tiempo de la última consulta efectuada. Las variables "intervalo1, intervalo2, intervalo3, intervalo4, contienen el tiempo que debe pasar antes de que se ejecútela consulta. Cada una de ellas tiene un desfase de 500ms. Las variables "llave1, llave2, llave3, llave4", son usadas para activar las consultas que se van a realizar. El tiempo total de consultas es de alrededor de 2.5 s.

Al cabo de cada consulta enviada, se espera la llegada de los bytes de respuesta provenientes del equipo impresor, esta lectura de datos se efectúa con el siguiente bloque de código haciendo uso de los eventos.

```
  void serialEvent2() { //Se ejecuata al finalizar el loop.
  //TCP ENVIA INFO
  while (Serial2.available()) {
    digitalWrite(RX_MEGA_LED, HIGH);
    char inChar = (char)Serial2.read();// obtiene un nuevo byte
    inputString += inChar;  // se agrega a inputString:

    if (inChar == '\r') {
      stringComplete = true;
    }
    digitalWrite(RX_MEGA_LED, LOW);
  }
}
```

Los datos recibidos, son alojados en la variable tipo string llamada "inputString". Esta variable contiene la respuesta a la consulta previamente enviada para lo cual la cabecera de esta variable contendrá los bytes respectivos definidos en el protocolo zipher. En base a la cabecera recibida, se procede a segmentar y codificar los datos para así poderlos empaquetar y concatenar posteriormente.

Si la cabecera empieza con "STS" significa que es una respuesta correspondiente al estado del equipo. Esta cadena nos entrega los valores de las variables estado (0,4) , receta, contador y batch, las cuales serán usadas más adelante. A continuación se muestra el procesamiento de los datos en base a la cabecera recibida.

```
////////////// ////-------RESPUESTAS-------//////////////////
// print the string when a newline arrives:
if (stringComplete) {
  //Serial.println(inputString);

  //Analiza info del equipo SI la respuesta es de solo 7
  //bytes =>  es una respuesta de cambio de estado del equipo.
  if (inputString.startsWith("STS|")
  && inputString.length() == 7)
  {
    estado = " ";
    estado = inputString.substring(4, 5);
  }

  //SI la respuesta es de MAS de 7 bytes =>
  //es una respuesta de estado general del equipo.
  if (inputString.startsWith("STS|")
  && inputString.length() > 8) {
    estado = " ";
    estado = inputString.substring(4, 5);
    //if (estado == "0") estado = "APAGADO";
    //if (estado == "1") estado = "ENCENDIENDO";
    //if (estado == "2") estado = "APAGANDO";
    //if (estado == "3") estado = "EN FUNCIONAMIENTO";
    //if (estado == "4") estado = "FUERA DE LINEA";
    String x = inputString.substring(8);
    int y = x.indexOf("|");
    receta = " ";
    receta = x.substring(0, y);
    String a = x.substring(y + 1);
    int b = a.indexOf("|");
    batch = a.substring(0, b);
    contador = " ";
    contador = a.substring(b + 1, a.length() - 2);
    RX1 = true ; //confirmo recepcion de datos
    //(estados, receta, batch, contador)
  }
```

Si la cabecera empieza con “WRN” significa que es una respuesta correspondiente a las advertencias presentes en el equipo. De esta cadena, únicamente guardamos los códigos de advertencias presentes, de manera que se los pueda concatenar como (código1)(código2)...etc. A continuación se muestra el código para la cabecera WRN.

```
  //Si la respuesta es de warnings disponibles.
  //Obtengo cantidad de warnings y el codigo
    if (inputString.startsWith("WRN|")) {
      qty_Warning = " ";
      qty_Warning = inputString.substring(4, 5);
      warning_cod = " ";
      if (qty_Warning != "0")
      {//almaceno solo los codigos del warning
      //y no su descripcion.
        for (int d = 0; d <= inputString.length(); d++) {
          if (inputString.substring(d, d + 1) == "(") {
        warning_cod = warning_cod +
     inputString.substring(d + 1, inputString.indexOf(")", d));
          }
        }
        //Serial2.println(warning_cod);
      }
      RX2=true ;  //confirmo recepcion de datos
      //(qty adver, cod adv)
    }
```

Si la cabecera empieza con "FLT" significa que es una respuesta correspondiente a las FALLAS presentes en el equipo. De esta cadena, únicamente guardamos los códigos de fallas presentes, de manera que se los pueda concatenar como (código1)(código2)...etc. A continuación se muestra el código para la cabecera FLT.

```
//Si la respuesta es de FAULTS disponibles.
//Obtengo cantidad de faults y el codigo
    if (inputString.startsWith("FLT|")) {
      qty_faults = " ";
      qty_faults = inputString.substring(4, 5);
      faults_cod = " ";
      if (qty_faults != "0") {
        for (int c = 0; c <= inputString.length(); c++) {
          if (inputString.substring(c, c + 1) == "(") {
            faults_cod = faults_cod +
    inputString.substring(c + 1, inputString.indexOf(")", c));
          }
        }
        //Serial2.println(faults_cod);
      }
      RX3=true; //confirmo recepcion de datos
      //(qty fallos, cod fallos)
    }
```

Si la cabecera empieza con "JDL" significa que es una respuesta correspondiente a los campos presentes en la receta del equipo, es decir, es la información que se esta imprimiendo en ese instante. Este string es el mas largo de todos, contiene tanto los labels como los contenidos de cada campo presente en la receta. En esta porción de código se aplica la codificación presentada en tablas anteriores con el fin de reducir la cantidad de bytes en los campos correspondientes a las fechas de fabricación y de vencimiento. Al aplicar esta codificaciones, se logra reducir de 10 a 3 bytes, acortando así la longitud final de caracteres a ser enviados. A continuación se muestra el código para la cabecera JDL.

```
//Si la respuesta es de los campos disponibles.
//Obtengo cantidad de campos y campos del mensaje
    if (inputString.startsWith("JDL|")) {
      qty_campos = " ";
      qty_campos = inputString.substring(4, 5);
      campos = " ";
      if (qty_campos != "0") {
        campos = inputString.substring(6); //JDL|7|...
        campos = campos.substring(10);
        fabricacion = "";
        hora="";
        lote="";
        precio="";
        vence="";
        fabricacion = campos.substring(0, campos.indexOf("|"));
        //2022/09/30
        if (fabricacion.length()==10){

        String mm=fabricacion.substring(5,7); //extraigo el mes
            Mm= mm.toInt();
        String dd=fabricacion.substring(8); //extraigo los dias
            Dd= dd.toInt(); //contiene los dias en entero
            switch (Mm){ //Codificacion del mes
              case 1:
              m="A";
              break;

              case 2:
              m="B";
              break;

              case 3:
              m="C";
              break;

```

```
36                case 4:
37                m="D";
38                break;
39
40                case 5:
41                m="E";
42                break;
43
44                case 6:
45                m="F";
46
47                case 7:
48                m="G";
49                break;
50
51                case 8:
52                m="H";
53                break;
54
55                case 9:
56                m="I";
57                break;
58
59                case 10:
60                m="J";
61                break;
62
63                case 11:
64                m="K";
65                break;
66
67                case 12:
68                m="L";
69                break;
70
71                default:
72                m="0";
73                break;
74              }
```

Al finalizar de segmentar y codificar las respuestas, se concatena en un solo string "datoA" sus contenidos para posteriormente ser enviado hacia el módulo de comunicación ESP32 LORA. A continuación se muestra el código donde se concatena los strings enviados:

```

// SI se recibieron las repuestas, se concatena.
  if (RX1 == true && RX2==true && RX3==true && RX4==true){
  // CADENA QUE INCLUYE CANTIDAD DE FALLAS Y
  //CANTIDAD DE ADVERTENCIAS
  //dato_A = "#" + receta + "*" + estado + "@" + contador + "$"
  //+ batch   + "&" + qty_Warning + "<" + qty_faults  + "/"
  //+ faults_cod + ">" + warning_cod + "?" + fabricacion + "^"
  //+ hora + "!" + lote + "%" + precio + "{" + vence;
  dato_A = "#" + receta + "*" + estado + "@" + contador + "$"
  + batch   + "&" + faults_cod + ">" + warning_cod + "?" +
  fabricacion + "^" + hora + "!" + lote + "%" + precio +
  "{" + vence;
  }
```

3.5.2. Comunicación con el ESP32 LoRa

Envío al ESP32 LoRA, a continuación se muestra el código.

```
//tiempo de espera para respuesta del comando anterior
  if (((tiempo_actual - previous_delay) > intervalo5  ))
  {
    previous_gst = tiempo_actual;
    previous_wrn = tiempo_actual;
    previous_gjd = tiempo_actual;
    previous_gft = tiempo_actual;
    previous_delay = tiempo_actual;
    llave1 = true;
    llave2 = true;
    llave3 = true;
    llave4 = true;

    //envia al ESP-lora la informacion recolectada cada 2.5g
    digitalWrite(TX_LORA_LED, HIGH);
    Serial1.print(dato_A);
    //libero las llaves para que se vuelva a concatenar
    //la variable dato.
    RX1 = false;
    RX2 = false;
    RX3 = false;
    RX4 = false;
    //MONITOREO POR EL TCP
    //Serial2.println(dato_A);
    ////////////////////////7
    digitalWrite(TX_LORA_LED, LOW);
    dato_A = " ";
  }
```

El tránsito de información entre los módulos ESP32 LORA – MEGA 2560 – MÓDULO ETHERNET, es visualizado mediante diodos leds soldados en la tarjeta electrónica. Una vez enviada al ESP32 LORA la trama "datoA" que contiene toda la información del equipo impresor, el proceso se repite indefinidamente. Otro bloque de código, se encarga de la recepción de información (proveniente del ESP32) , procesarla y retransmitirla hacia el equipo impresor. Se trata de la información de trazabilidad que el usuario desea que el equipo imprima a continuación se muestra el código.

```
//Actualizacion de campos. Envia a la codificadora la trama
//decodificada
//SLA|RECETA REMOTO|F=2022/08/01|E=2023/08/01|P=0.50|L=LGG|<CR>
  if (stringESP) {//  >2H13H1050LGG<CR> .
  //la informacion que recibe el mega esta codificada,
  //apuntespara ver las equivalencias de las letras asignadas.
  //2H1=FABRICACION . 3H1=VENCIMIENTO . 050=PRECIO . LGG=LOTE
    if (inputESP.startsWith(">")) {
      //segmento los campos
      traza_fab=inputESP.substring(1,4);      //2H1
      traza_vence=inputESP.substring(4,7);    //3H1
      traza_price=inputESP.substring(7,10);   //050
      traza_lote=inputESP.substring(10);      //LGG
```

Al recibir la trama codificada, el algoritmo debe decodificarla para empaquetarla de forma que el equipo impresor pueda entenderla. A continuación se muestra el código.

```
//identifico campos para FECHA DE FABRICACION
      String aF = traza_fab.substring(0,1);

      String mF = traza_fab.substring(1,2); //seleccion del mes
      if (mF=="A") {mes_fab= "01" ;}
      if (mF=="B") {mes_fab= "02" ;}
      if (mF=="C") {mes_fab= "03" ;}
      if (mF=="D") {mes_fab= "04" ;}
      if (mF=="E") {mes_fab= "05" ;}
      if (mF=="F") {mes_fab= "06" ;}
      if (mF=="G") {mes_fab= "07" ;}
      if (mF=="H") {mes_fab= "08" ;}
      if (mF=="I") {mes_fab= "09" ;}
      if (mF=="J") {mes_fab= "10" ;}
      if (mF=="K") {mes_fab= "11" ;}
      if (mF=="L") {mes_fab= "12" ;}
      String dF =traza_fab.substring(2,3);  //seleccion del dia
```

```
      int fday = dF.toInt();
      if (fday<10 && fday!=0){ // si el digito es <10,
      //le asigno un 0 para que se represente como 01, 02, 03..
       // dia_fab="0"+dF;
      }
      if (fday==0){ // si la conversion no es posible
      //debido a que fday NO ES un numero sino letra
      //=> busco que letra es y obtengo la posicion-.
        for (int x= 0; x<=21 ; x++){
          if(dF == dias[x]){
             dia_fab=(String) (x+10);
          }
        }
      }
      traza_fab=ano_fab + "/" + mes_fab + "/" + dia_fab;
      //fecha de fabircacion en formato yyyy/mm/dd
```

```
//identifico campos para FECHA DE VENCIMIENTO
      String aV = traza_vence.substring(0,1);
      ano_vence="202"+aV;
      String mV=traza_vence.substring(1,2);
      if (mV=="A") {mes_vence= "01" ;}
      if (mV=="B") {mes_vence= "02" ;}
      if (mV=="C") {mes_vence= "03" ;}
      if (mV=="D") {mes_vence= "04" ;}
      if (mV=="E") {mes_vence= "05" ;}
      if (mV=="F") {mes_vence= "06" ;}
      if (mV=="G") {mes_vence= "07" ;}
      if (mV=="H") {mes_vence= "08" ;}
      if (mV=="I") {mes_vence= "09" ;}
      if (mV=="J") {mes_vence= "10" ;}
      if (mV=="K") {mes_vence= "11" ;}
      if (mV=="L") {mes_vence= "12" ;}
      String dV = traza_vence.substring(2,3);
      int vday = dV.toInt();
      if (vday<10 && vday!=0){ // si el digito es <10,
        dia_vence = "0"+dV;
      }
      if (vday==0){ // si la conversion no es posible debido a
      //que vday NO ES un numero sino letra
      //=> busco que letra es y obtengo la posicion-.
        for (int y= 0; y<=21 ; y++){
          if(dV == dias[y]){
             dia_vence=(String) (y+10);
          }
        }
      }
```

Finalmente, la información de trazabilidad decodificada se empaqueta en un string "receta" y se envía al módulo Ethernet quien a su vez lo trasmite al equipo impresor. A continuación se muestra el código .

```
1
2  //quedando asi =>0.50
3        //traza_lote //=traza_lote.substring(0,3)+"|"
4        //+traza_lote.substring(3);
5
6
7  receta="SLA|"+ trabajo + "|F=" + traza_fab +
8  "|E=" + traza_vence + "|P=" + traza_price +"|L=" + traza_lote ;
9        Serial2.print(receta);
10       // ENVIO AL CODIFICADOR EL STRING CORRECTO
11       inputESP = "";
12       stringESP = false;
13       traza_fab ="";
14       traza_vence ="";
15       traza_price="";
16       traza_lote ="";
17     }
18   }
19
20 }
```

3.5.3. Configuración de módulo ETHERNET

El módulo convertidor TCP – RS232 se configura desde el propio software provisto por el fabricante. El software usado de USR-TCP232-T2 en su versión v2.2.3.286. A continuación la pantalla de inicio del software.

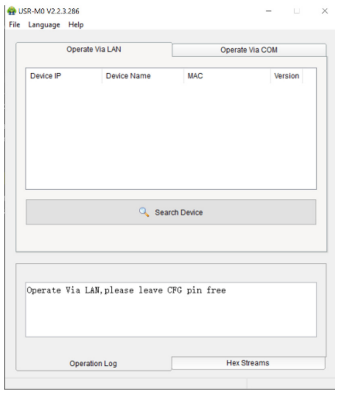


Figura 3.15: Inicio software USR-TCP232-T2
Fuente: Los autores

Para la configuración del módulo Ethernet, debemos cambiar la IP del computador, mascara de subred y puerta de enlace tal como se muestra en la siguiente figura 3.16.

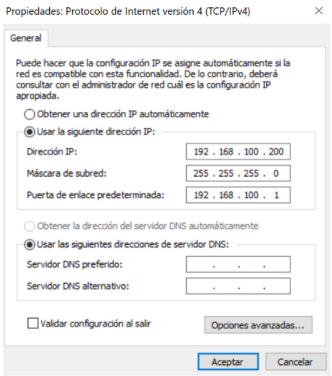


Figura 3.16: Configuración del módulo
Fuente: Los autores

El módulo Ethernet tiene designada de fabrica la siguiente IP:

192.168.100.7, por lo que nuestra PC deberá estar dentro de esa misma red.

Una vez cambiado las propiedades del protocolo, procedemos a conectar nuestro módulo al PC mediante un cable de red. El módulo Ethernet debe estar alimentado con +5vcc. Luego, en el software de USR-TCP232-T2 procedemos a presionar el botón "Search Device". Ver figura 3.17.

Figura 3.17: Búsqueda de dispositivos
Fuente: Los autores

Una vez que el dispositivo es detectado, mostrara los detalles de su configuración que tiene cargados como se puede observar en la figura 3.18.

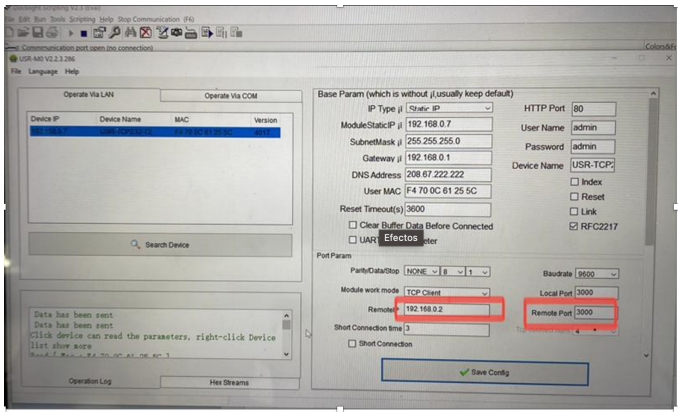


Figura 3.18: Búsqueda de dispositivos
Fuente: Los autores

En esta pantalla, procedemos a cambiar en el campo "MODULE WORK

MODE" a TCP Client. Inmediatamente después en "RemoteIP" ingresamos la dirección IP del equipo impresor, la cual es 192.168.100.2 de fabrica. Finalmente, ingresamos el puerto de comunicación en "Remote Port" el cual es 3000, mismo puerto de comunicación del equipo impresor. Al terminar los ajustes, debemos dar click sobre "Save config" y esperamos a que termine de guardarse los cambios en el módulo. Una vez guardados los cambios, procedemos a la desconexión del cable de red del pc y a conectarlo a l puerto Ethernet del equipo impresor.

3.5.4. Programa del ESP32 LoRa

El módulo de comunicación ESP32 LORA hace uso de un microcontrolador dual-core 32-bit y del módulo lora SX1276. Incluye además un display oled 128*64. Este dispositivo es el encargado de responder ante solicitudes de información asegurando el envió de la misma haciendo uso algoritmos especializados en la detección de fallos de recepción de información.

El programa cargado en el dispositivo tiene la capacidad de receptar todo el tránsito de información existente y solo responde si la información es enviada a la dirección única pre establecida, de manera que la información recibida por parte del Mega2560 es retransmitida hacia el dispositivo principal Únicamente cuando se le haga la consulta. A continuación, se muestra la tabla de direcciones establecidas y asignadas.

Tabla 3.1: Direcciones

Dispositivo	***Direccion***
Equipo1	0xE6
Equipo2	0xDC
Equipo3	0xD2
Equipo4	0xC8
Equipo5	0xBE
Equipo6	0xD3
Equipo7	0xF0
Equipo8	0xFD
Equipo9	0xFD

El equipo 8 y 9 comparten la misma dirección, y esto es debido a que estos equipos trabajan con el sistema redundante el cual solo maneja un módulo

de comunicación lora. Este dispositivo se vera a detalle mas adelante. A continuación se muestra el programa cargado en el ESP32.

```
if (Impresor.available() > 0) {
    inputESP = " ";
    receta="";
    estado="";
    contador="";
    batch="";
    qty_war="";
    qty_flt="";
    fabricacion="";
    hora="";
    lote="";
    precio="";
    vencimiento="";

    while (Impresor.available() > 0) {
      String stringEsp = Impresor.readStringUntil('\r');
      inputESP += stringEsp;
    }
    //"#" + receta + "*" + estado + "@" + contador+
    //"$" + batch   + "&" + qty_Warning + "<" + qty_faults
    //+ "/"  + faults_cod + ">" + warning_cod + "?"
    //+ fabricacion + "^" + hora + "!" + lote + "%"
    //+ precio + "{" + vence;
    dato_mega = inputESP;
    //Serial.println(dato_mega);
    receta      = dato_mega.substring(dato_mega.indexOf('#')
    + 1,dato_mega.indexOf('*'));
    estado      = dato_mega.substring(dato_mega.indexOf('*')
    + 1, dato_mega.indexOf('@'));
    est=0;
    est_rf="";
    est=estado.toInt();
    switch (est){
      case 0:
      estado="APAGADO";
      est_rf="0";
      break;

      case 1:
      estado="Encendiendo";
      est_rf="1";
      break;
```

El ESP32 siempre monitorea si hay datos en el buffer del puerto serie provenientes del Mega2560 y en caso de existir información se recibe, procesa y empaqueta. La información proveniente del Mega2560, contiene los siguientes campos: Receta, estado del equipo, contador general, contador del batch, códigos de advertencias, códigos de fallas, fecha de fabricación, fecha de vencimiento , precio, lote y hora. Una vez completados los campos, se concatenan en una sola variable para medir la longitud total de caracteres a ser enviados.

La verificación de solicitud de información por RF se logra consultando constantemente si existen datos provenientes del chip lora. En caso de existir y si la dirección de consulta es igual a la dirección local, entonces se procede a responde ante el requerimiento. El primer comando a enviar es la cantidad total de bytes a ser enviados y posteriormente cada campo por separado. A continuación se muestra código.

```
onReceive(LoRa.parsePacket());

  if (disp_destinatario == dir_local && incoming ==
  (String)dir_local ){ // si se nos hace una consulta
  //y el contenido es = a la direccion local => se responde.
     sendMessage(longitud);//primero envio la longitud de
     //los datos  a enviarse. *! + longitud
      sendMessage(receta);
      sendMessage(est_rf);  //est contiene el estado   0 al 3.
      sendMessage(contador);
      sendMessage(batch);
      sendMessage(war_cod);
      sendMessage(flt_cod);
      sendMessage(fabricacion);
      sendMessage(vencimiento);
      sendMessage(hora);
      sendMessage(precio);
      sendMessage(lote);
      disp_destinatario=0;
  }
```

Otra tarea del ESP32, es recibir y retransmitir al Mega2560 la información de trazabilidad que el usuario desea que el equipo terminal imprima, como se muestra código.

```
if (incoming.startsWith(">") && t==true){
    Impresor.print(incoming); //envio al mega la trama
    //tal cual llega.  >2H13H1050JGG<CR>
    Heltec.display->clear();
    Heltec.display->setTextAlignment(TEXT_ALIGN_LEFT);
    Heltec.display->setFont(ArialMT_Plain_10);
    Heltec.display->drawString(0, 0, "Dato tx a PC:" );
    Heltec.display->drawString(0, 10, incoming);
    Heltec.display->display();
    Serial.println("Se ha enviado al mega:");
    Serial.println(incoming);
   // long_traza=0;
    //trazabilidad="";
    //l , t = false;
    t = false;
     datoRxLora = '\0'; //borro el contenido
   }
```

Subrutina para el envió de información por RF. Controla las direcciones del dispositivo destino , si la dirección del dispositivo destino es diferente a la dirección del dispositivo local, entonces la información es desechada.A continuación se muestra el código.

```
void sendMessage(String orden)
{
  LoRa.beginPacket();            // start packet
  LoRa.write(dir_destino);      // add dir_destino address
  LoRa.write(dir_local);        // add disp_emisor address
  LoRa.write(msgCount);         // add message ID
  LoRa.write(orden.length());   // add payload length
  LoRa.print(orden);           // add payload
  LoRa.endPacket();           // finish packet and send it
  msgCount++;                 // increment message ID
}
```

3.5.5. Sistema redundante

El principio del sistema redundante consiste en mantener siempre dos equipos impresores trabajando todo el tiempo de manera que, si al fallar uno de ellos, la línea de producción no se detenga el equipo restante quedaría trabajando. Esta asignación de carga de trabajo se logra automáticamente y es gestionada por un microcontrolador ESP32. Para lograrlo, es necesario monitorear el estado actual de cada equipo en todo momento y en base a los estados de cada equipo tomar las decisiones de asignación. Además, se

requiere de dos sensores fotoeléctricos para la detección de los productos Un sensor para detectar el paso del producto (orden de trabajo). y el otro sensor de seguridad para avisar del bloqueo del primer sensor. El esquema de asignación de tarea en base a estados se aprecia en la siguiente figura 3.19.

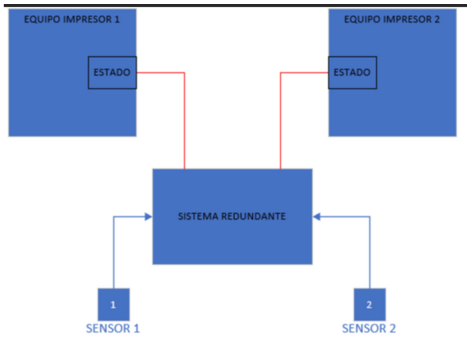


Figura 3.19: Estados
Fuente: Los autores

La lógica aplicada para la asignación de trabajo, se la presenta en el siguiente diagrama de flujo, ver figura 3.20.

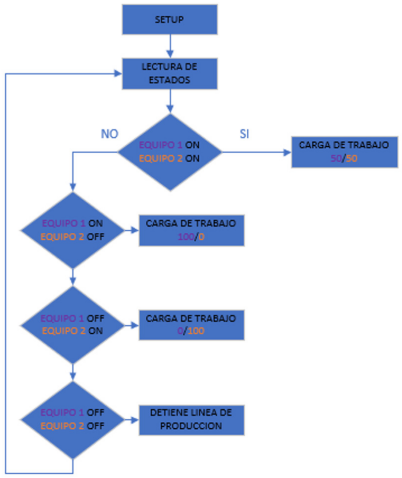


Figura 3.20: Estados
Fuente: Los autores

Las ordenes de impresión son gestionadas en el ESP32 dependiendo de la distribución de la carga de trabajo, es decir, si solo hay un solo equipo operativo, el sistema enviara todas las ordenes de trabajo entrantes hacia ese único dispositivo. Por el contrario, si ambos equipos estuviesen activos, entonces las órdenes de trabajo se enviaran de forma alternada hacia cada uno de los equipos. Cada orden de trabajo enviada hacia los equipos es verificada mediante un pulso de confirmación. Este pulso de confirmación proviene de cada equipo impresor y es emitido una vez que la orden de trabajo se ha completado con éxito. Los pulsos de confirmación indican al sistema que el equipo impresor esta "activo" y que se le puede seguir enviando pulsos de ordenes de trabajo. Si por alguna razón, el equipo impresor dejara de emitir este pulso de confirmación luego de haber recibido una orden de trabajo, entonces el sistema determina que el equipo impresor esta inhibido, lo cual suele suceder cuando hay temperaturas de trabajo excesivamente altas.

3.5.6. Código sistema redundante

Para la captura de los pulsos provenientes del sensor de producto, sensor redundante, pulsos de confirmación 1 y pulsos de confirmación 2, es necesario hacer uso de interrupciones, las cuales ayudan enormemente al procesamiento de cada señal e impiden la perdida de pulsos. Cada interrupción se encarga de incrementar o decrementar variables y activar banderas que son usadas en el bloque principal, a continuación se muestra el código.

```
//interrupcion//
void IRAM_ATTR PULSO1() {
    orden_imp1 = 0; // si se recibe un pulso de confirmacion,
    //sobra y basta. si no hay respuesta en mas de 20 pulsos
}

void IRAM_ATTR PULSO2() {
   orden_imp2 = 0;
}

void IRAM_ATTR SPRODU() {
//Pulsos entrantes del sensor de producto.
   sp = true;
   produ_in++; //pulsos del sp entrantes
   prev_Sprodu_time=millis();
}

void IRAM_ATTR SREDU() {
```

```
   produ_in--;
}

void IRAM_ATTR RESET() {
      produ_in = 0;
      orden_imp1=0;
      orden_imp2=0;
      digitalWrite(rele_fallo, LOW);
      rele_llave = false;
      paro_linea = "";
}
```

A continuación se muestra el código de la lectura de los estados de los equipos impresores.

```
void loop() {
    tiempo_actual=millis();
    val_estado_eq1 = digitalRead(estado_eq1);
    val_estado_eq2 = digitalRead(estado_eq2);
    if (val_estado_eq1 == LOW && val_estado_eq2 == LOW )
    { estado_actual = 0;}// Si ambos equipos  en fallo/apagados
    //=> se activa el rele de fallo.
    if (val_estado_eq1 == HIGH && val_estado_eq2 == LOW)
    { estado_actual = 1;}// Si eq1 OK y eq2 FALLO
    if (val_estado_eq1 == LOW && val_estado_eq2 == HIGH )
    { estado_actual = 2;} // Si eq1 FALLO y eq2 OK
    if (val_estado_eq1 == HIGH && val_estado_eq2 == HIGH )
    { estado_actual = 3;} // Si eq1 OK y eq2 OK
```

Si ambos equipos están fuera de servicio, se activa el relé de fallo el cual hace detener la línea de producción, a continuación se muestra el código.

```
case 0: //cuando ambos equipos no estan listos
        estado="#EQ1=OFF / EQ2=OFF";
        digitalWrite (rele_fallo, HIGH);
        // ACTIVA RELE PARA DETENER LA LINEA DE PRODUCCIO
        Heltec.display->clear();
        Heltec.display->setTextAlignment(TEXT_ALIGN_LEFT);
        Heltec.display->setFont(ArialMT_Plain_10);
        Heltec.display->drawString(0, 0,"EQ1 FAULT/EQ2 FAULT");
        Heltec.display->drawString(0, 54, "Umbral: "
        + String(produ_in));
        Heltec.display->setTextAlignment(TEXT_ALIGN_CENTER);
        Heltec.display->drawString(64, 22, "RELE ACTIVADO");
        Heltec.display->display();
        break;
```

A continuación se muestra el código de las ordenes de impresión para el equipo 2 y se muestra estado en la pantalla oled.

```
case 2://2->eq1 fallo y eq2 OK
        estado="#EQ1=OFF / EQ2=RUN";
       if (rele_llave == false){
          digitalWrite (rele_fallo, LOW);
          // DESACTIVA RELE DE FALLA
        } //IMPRIME
       if(sp){
       //si se recibe un pulso del sensor de producto,
       //se envia a imprimir
         digitalWrite (orden_imp_eq2, HIGH);
    //activo opto para orden de impresion eq1 queda en alto
    if(tiempo_actual - prev_Sprodu_time >= ancho_pulso_imp)
         { //ancho pulso es de 30
            prev_Sprodu_time = millis();
            digitalWrite (orden_imp_eq2, LOW);
            orden_imp2++;
            sp=false;
         }
       }
        Heltec.display->clear();
        Heltec.display->setTextAlignment(TEXT_ALIGN_LEFT);
        Heltec.display->setFont(ArialMT_Plain_10);
        Heltec.display->drawString(0, 0, "EQ1 FAULT / EQ2 RUN");
        Heltec.display->drawString(0, 10, String(paro_linea));
        Heltec.display->drawString(0, 45,"Orden Impresion 2: "
        + String(orden_imp2));
        Heltec.display->drawString(0, 54, "Umbral: "
        + String(produ_in));
        Heltec.display->display();
        break;
```

Alternancia de ordenes de impresión entre equipo 1 y equipo 2. Se actualiza el estado en la pantalla oled, a continuación se muestra el código.

```
case 3://, 3->eq1 OK y eq2 OK. IMPRIME ALTERNADAMENTE//
        estado="#EQ1=RUN / EQ2=RUN";
        if (rele_llave == false){
          digitalWrite (rele_fallo, LOW);
        }// DESACTIVA RELE DE FALLA
       //IMPRIME eq 1
       if(sp==true && anterior == false ){
         digitalWrite (orden_imp_eq1, HIGH);
        if(tiempo_actual - prev_Sprodu_time >= ancho_pulso_imp)
         { //ancho pulso es de 30
            prev_Sprodu_time = millis();
```

```
            digitalWrite (orden_imp_eq1, LOW);
            orden_imp1++;  //incrementa en 1 la var
            sp=false;
            anterior = true;
          }
        }

        //IMPRIME  EQ2
        if(sp==true && anterior == true){
          digitalWrite (orden_imp_eq2, HIGH);
    if(tiempo_actual - prev_Sprodu_time >= ancho_pulso_imp){
        prev_Sprodu_time = millis(); //actualizo la variable.
            digitalWrite (orden_imp_eq2, LOW);
            orden_imp2++;
     //incrementa en 1 la var de orden de impresion al eq 1.
            sp=false;
            anterior = false;
          }
        }
         Heltec.display->clear();
         Heltec.display->setTextAlignment(TEXT_ALIGN_LEFT);
         Heltec.display->setFont(ArialMT_Plain_10);
         Heltec.display->drawString(0, 0, "EQ1 RUN / EQ2 RUN");
         Heltec.display->drawString(0, 10, String(paro_linea));
         Heltec.display->drawString(0, 35, "Orden Imp1: "
         + String(orden_imp1));
         Heltec.display->drawString(0, 45,  "Orden Imp2: "
         + String(orden_imp2));
         Heltec.display->drawString(0, 54, "Umbral: "
         + String(produ_in));
         Heltec.display->display();
         break;
```

Activación de relé de fallo si las ordenes de impresión no han sido confirmadas luego de un valor estableció o si el sensor de presencia 1 esta bloqueado, a continuación se muestra el código.

```
if( produ_in <=margen_prod_perdido
|| orden_imp1 >margen_p_c_perdidos
|| orden_imp2 > margen_p_c_perdidos){
      digitalWrite(rele_fallo, HIGH);
      rele_llave = true ;
// desactivo control del rele en el swtich case de arriba.
      paro_linea="Rele Fallo ACTIVADO";
    }
```

Repuestas vía RF si existiese alguna consulta de información, a continuación se muestra el código.

```
//Lectura del buffer  RF
    onReceive(LoRa.parsePacket());
    //Evaluo si llego algun requerimiento de informacion
    if (t){ // ESP MAIN envia 1 o 2
      switch (req0){
        case 1: // respuesta ante solicitud de info al eq1
          sendMessage(longitud1);// envio la longitud datos
          sendMessage(receta1);
          sendMessage(est_rf1);
          //est contiene el estado en nuemro del 0 al 3.
          sendMessage(contador1);
          sendMessage(batch1);
          sendMessage(war_cod1);
          sendMessage(flt_cod1);
          sendMessage(fabricacion1);
          sendMessage(vencimiento1);
          sendMessage(hora1);
          sendMessage(precio1);
          sendMessage(lote1); //lote contiene un > al final
          t=false;
          req0=0;
        break;
        case 2: // respuesta ante solicitud de info al eq1
          sendMessage(longitud2);
          //primero envio la longitud de los datos  a enviarse.
          //*!+ longitud
          sendMessage(receta2);
          sendMessage(est_rf2);
    //est contiene el estado en nuemro del 0 al 3.
          sendMessage(contador2);
          sendMessage(batch2);
          sendMessage(war_cod2);
          sendMessage(flt_cod2);
          sendMessage(fabricacion2);
          sendMessage(vencimiento2);
          sendMessage(hora2);
          sendMessage(precio2);
          sendMessage(lote2); //lote contiene un > al final
          t=false;
          req0=0;
        break;
        default:
        break;
      }
    }
  }
```

Función para envió de datos vía RF,a continuación se muestra el código.

```
void sendMessage(String orden)
{
  LoRa.beginPacket();           // start packet
  LoRa.write(dir_destino);      // add dir_destino address
  LoRa.write(dir_local);        // add disp_emisor address
  LoRa.write(msgCount);         // add message ID
  LoRa.write(orden.length());   // add payload length
  LoRa.print(orden);            // add payload
  LoRa.endPacket();             // finish packet and send it
  msgCount++;                   // increment message ID
}
```

Función para lectura de datos vía RF, a continuación se muestra el código. vskip 0.5cm

```
void onReceive(int packetSize)
{
  incoming="";
  if (packetSize == 0) return; // if there's no packet, return
  // read packet header bytes:
  int destinatario = LoRa.read();// destinatario address
  byte disp_emisor = LoRa.read(); // disp_emisor address
  byte incomingMsgId = LoRa.read();// incoming msg ID
  byte incomingLength = LoRa.read();// incoming msg length
  disp_destinatario=destinatario;
  Emisor_dir = disp_emisor; // direccion del emisor en  byte

  if (destinatario != dir_local ) {
    return;                // skip rest of function
  }

  //LECTURA DEL BUFFER
  while (LoRa.available())
  {
    datoRxLora = LoRa.read(); // dato  a evaluar
    incoming += (char)datoRxLora;
  }
```

3.6. Desarrollo de la plataforma digital en Labview

3.6.1. Programación de la tarjeta de adquisición principal

La tarjeta de adquisición principal se encarga de realizar las consultas a cada equipo terminal de forma cíclica. En esta tarjeta ya están guardadas todas las direcciones a la cual debe realizar las consultas, así como también el orden a seguir. El primer dato de las repuesta recibida, indica la longitud de la trama total que debe recibirse, el algoritmo recibe esta información y posteriormente la comprara con la longitud total de datos recibidos. Si el algoritmo detecta que la longitud de datos recibidos es menor que lo que debe ser, entonces envía una nueva consulta al mismo dispositivo y si la longitud de datos ahora es igual a lo esperado entonces cambia de equipo para una nueva consulta. A continuación en la figura 3.21, un esquema del tránsito de información.

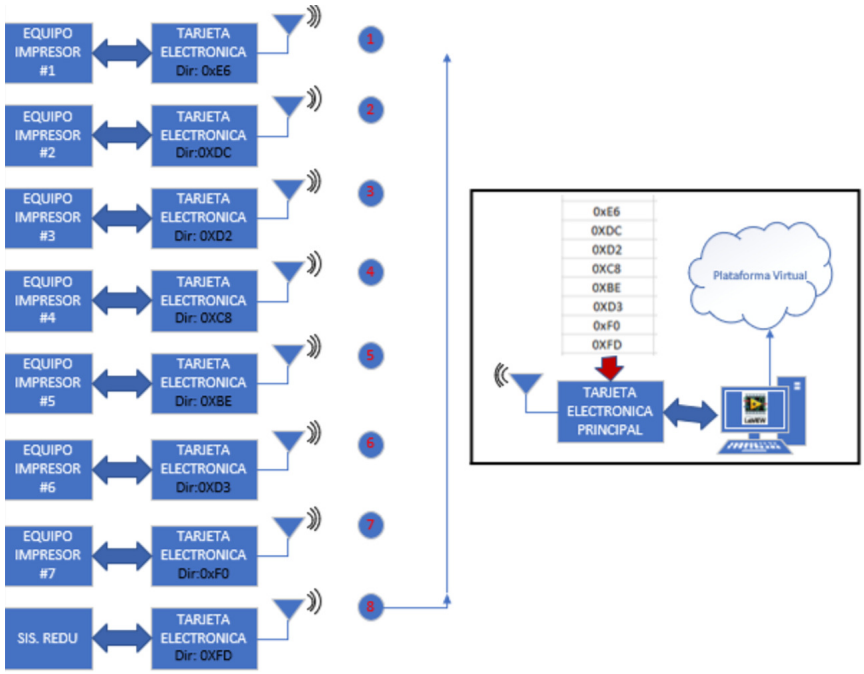


Figura 3.21: Esquema del tránsito de información
Fuente: Los autores

3.6.2. Interfaz gráfica

La interfaz grafica esta compuesta por varios subVi para el tratamiento de la información recibida proveniente de la tarjeta electrónica principal. Con la finalidad de compactar y reducir la cantidad de bytes a enviarse en la comunicación inalámbrica se codificaron los textos correspondientes a campos de fechas (elaboración y caducidad). Esos campos llegan codificados hasta la interfaz para ser decodificados y presentados correctamente. A medida que la información de cada equipo llegue, se va presentando y se actualiza en cada barrido de consultas. Una vez que la información ha llegado por completo es enviada también hacia la plataforma en la nube de thingsboard. A continuacion en la figura 3.22 se muestra la interfaz grafica implementada.

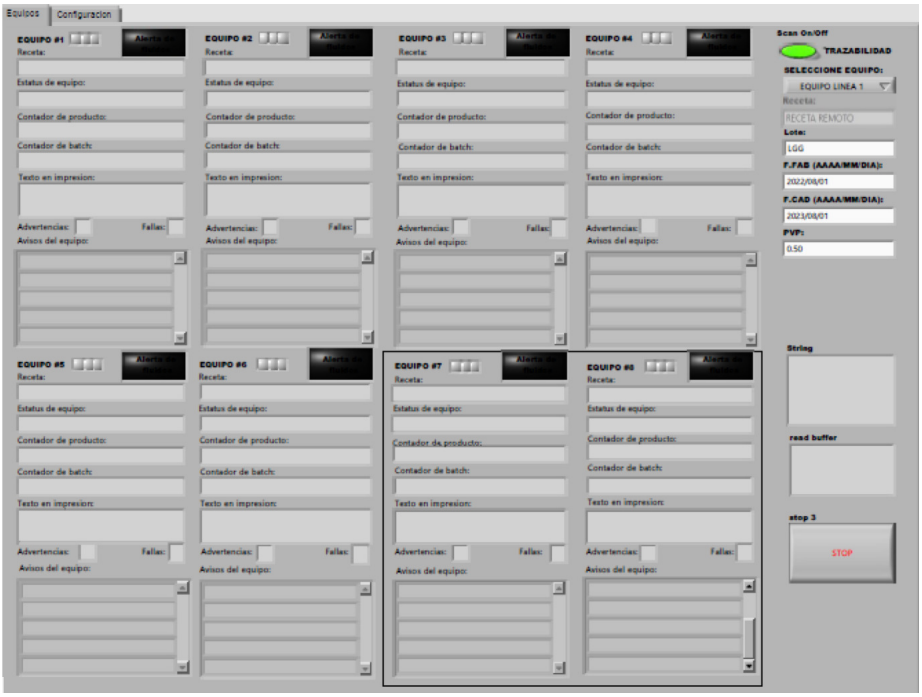


Figura 3.22: Interfaz gráfica
Fuente: Los autores

Cada segmento este compuesto de la siguiente estructura.

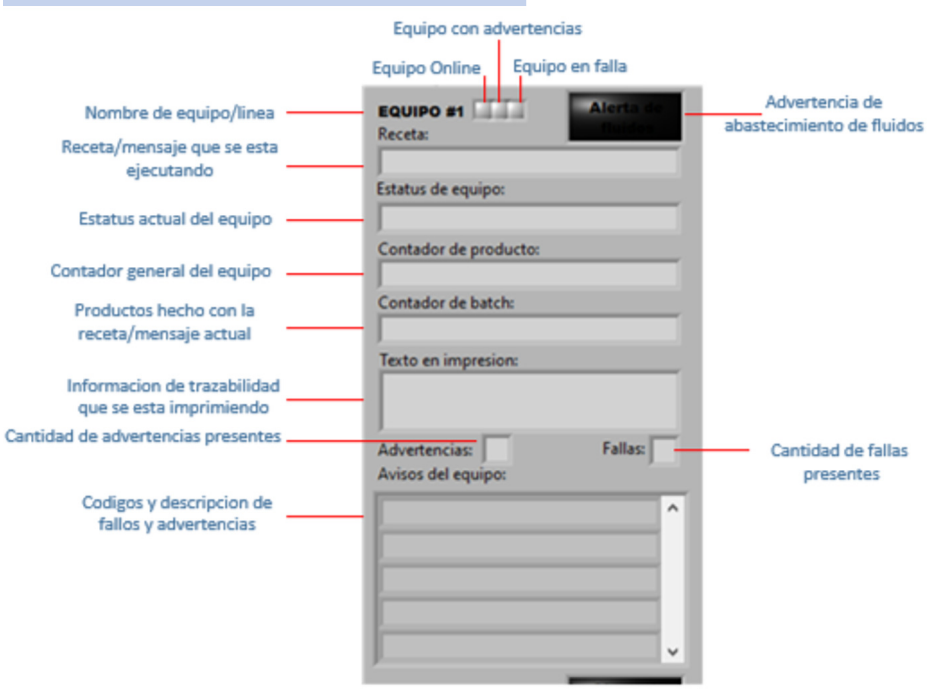


Figura 3.23: Segmento interfaz gráfica
Fuente: Los autores

- **Campo "Nombre de equipo/linea":** indica el nombre del equipo o de la línea donde esta instalado el equipo.
- **Campo "Online":** indica que el equipo esta en modo de impresión/activo.
- **Campo "Equipo con advertencias":**indica que el equipo presente una o mas advertencias.
- **Campo "Equipo en falla":** indica que el equipo esta en falla y por ende no esta trabajando.
- **Campo "advertencia de abastecimiento de fluidos":** indica que existe una o mas advertencias concernientes al abastecimiento de fluidos. Esta advertencia se ilumina en rojo para que le cliente le de la importación del caso de omitirla entonces el equipo va a detenerse.

- **Campo "Receta":** indica el nombre de la receta o mensaje que esta siendo codificado por el equipo impresor, esta información es relevante para el cliente debe coincidir con la producción en marcha.
- **Campo "Estatus de equipo":** indica el estado actual del equipo, estos estados pueden ser apagado, encendiendo, en línea, fuera de línea, apagado.
- **Campo "Contador de producto":** indica el contador total general del equipo. El cliente toma constantes lecturas de este valor para cotejar con los productos fabricados.
- **Campo "Contado de batch":**indica cuantas unidades han sido impresas con la receta actual. Si la receta cambia, este contador vuevle a 0.
- **Campo "Texto en impresión":** indica al cliente que es lo que se esta imprimiendo (información de trazabilidad) actualmente, esta información esta relacionada con el campo "receta". Para recetas diferentes la información de trazabilidad es diferente también.
- **Campo "Advertencias":** indica al cliente la cantidad de advertencia que están presentes en el equipo en ese instante.
- **Campo "Fallas":** indica a cliente la cantidad de fallas presentes en el equipo.
- **Campo "Avisos del equipo":** indica los códigos y descricon de la advertencia y/o fallo presente en el equipo.

La descomposición de esta trama se lleva acabo en el subvi denominado "DESENCADENANDO", en la siguiente figura 3.24 se muestra la estructura de programa.

La cadena de entrada se va descomponiendo a medida que el bloque "Match pattern" va segmentando los campos y alojando los resultados en las variables correspondientes. Los campos **receta, contador, batch, cantidad de fallos y cantidad de advertencias, hora, precio y lote**, son variables cuyos contenidos se reciben y guardan sin ningún tratamiento.

Por el contrario, las variables **estado, avisos, fecha de elaboración y caducidad** son variables que deben ser tratadas antes de ser alojadas en la interfaz

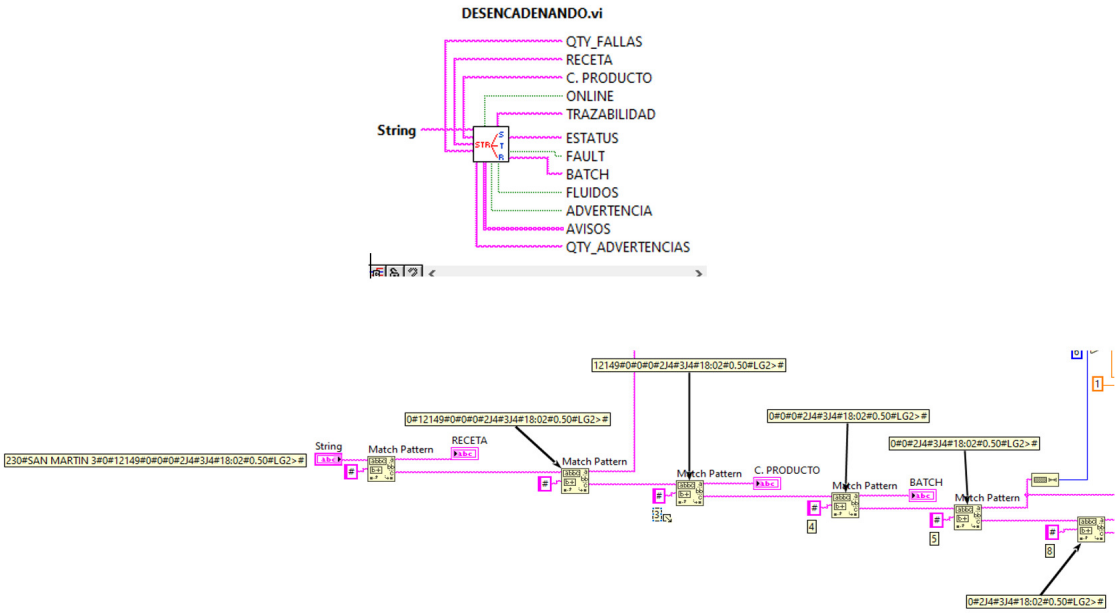


Figura 3.24: Segmento interfaz gráfica
Fuente: Los autores

3.6.3. Tratamiento de variables de estado

Esta variable de tipo string puede contener los caracteres 0,1,2,3 o 4. Cada uno de ellos representa el estado actual de la maquina por lo tanto se hizo uso del bloque "case structure" para interpretarlos y asignarles su contenido equivalente en base a la siguiente tabla.

Tabla 3.2: Variables de estado

valores	***Equivalencias***
Valor 0	Equivale a: Apagado
Valor 1	Equivale a: Encendiendo
Valor 2	Equivale a: Apagando
Valor 3	Equivale a: Online
Valor 4	Equivale a: Offline

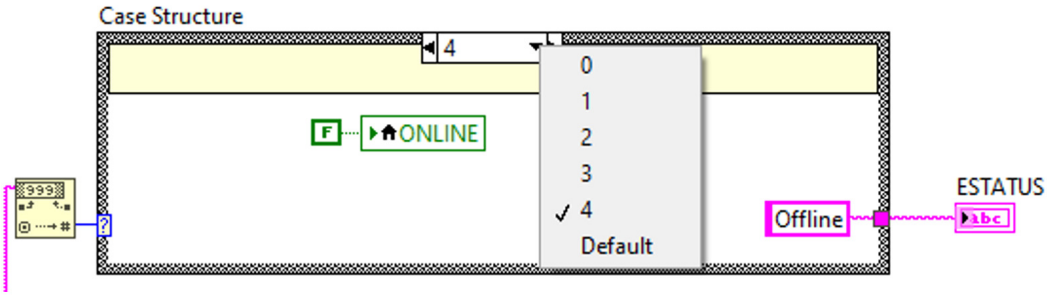


Figura 3.25: Estructura de caso
Fuente: Los autores

3.6.4. Tratamiento de variable Avisos

Esta variable de tipo string, contiene los códigos de las advertencias y/o fallas que pudiese presentar el equipo. Esta variable contiene los códigos presentes de forma concatenada y al recibirlos en la interfaz un subVi llamado "fallos y advertencias.vi" se encarga de separarlos y a su ves asignarles su respectiva descripción para finalmente presentar el resultado en un array llamado "Avisos". Este subvi también posee una salida de tipo boolena para activar el campo "Alerta de fluidos" si es que fuese necesario. Los códigos que llegan tienen el siguiente formato E10601E10602, etc y cada código consta de 6 digitos, como se muestra en la figura 3.26.

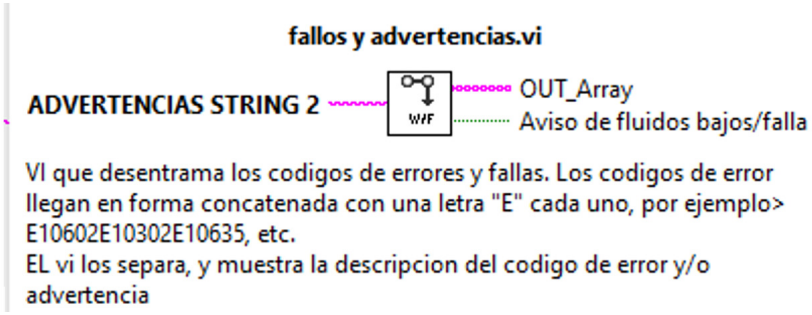


Figura 3.26: Fallos y advertencias
Fuente: Los autores

Este subvi tiene almacenado los códigos de error del equipo impresor, de manera que una vez que separa los códigos busca uno a uno su respectiva

descripción para así finalmente alojar el resultado en un array de salida, ver figura 3.27

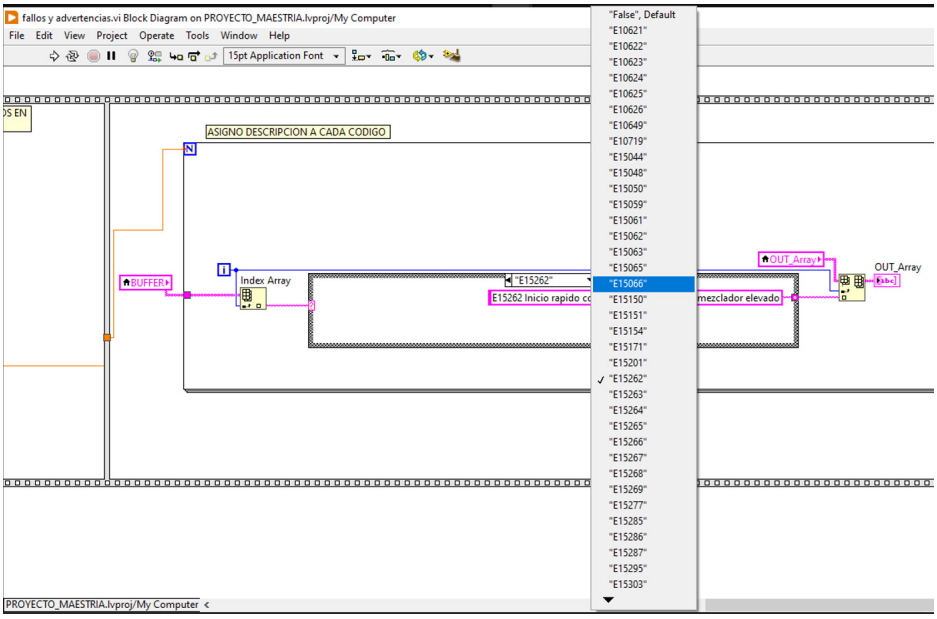


Figura 3.27: Códigos de error
Fuente: Los autores

3.6.5. Tratamiento de variable Fecha de elaboración y caducidad

Las variables tipo string correspondientes al a fecha de elaboración y fecha de caducidad constan de 3 dígitos. Estos dígitos representan el año, mes y día. Ejemplo “2AJ” donde 2 equivale al año 2022, “A” equivale al mes y “J” equivale al día del mes. El subVi llamado “DECODIFICANDO FECHAS.vi” es el encargado de interpretar estos códigos y de asignarle su respectivo contenido, como se muestra en la figura 3.28:

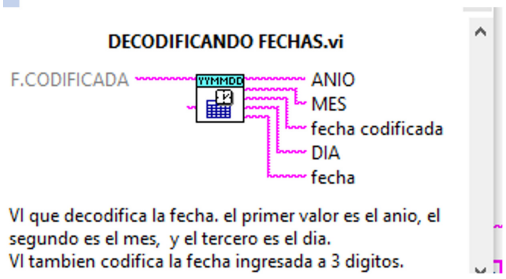


Figura 3.28: Decodificador de fechas
Fuente: Los autores

Otra función de este subVi es la de codificar fechas en 3 digitos. Función utilizada cuando se envia fecha de trazabilidad hacia los equipos impresores. La interpretación de códigos se lleva a cabo mediante el uso de bloques "case strucure" como se muestra en la siguiente figura 3.29.

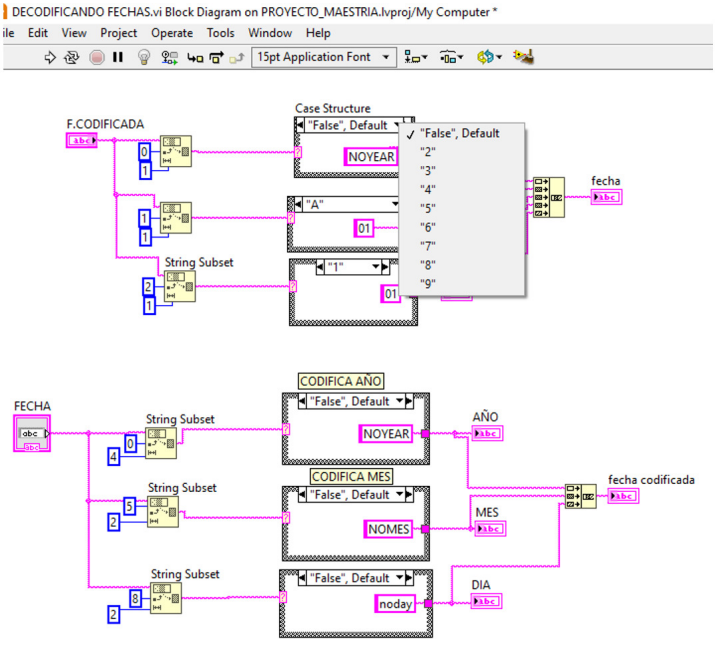


Figura 3.29: Decodificador de fechas
Fuente: Los autores

Una vez que toda la información ha sido procesada, se aloja los resultados

en los respectivos contenedores, ver figura 3.30

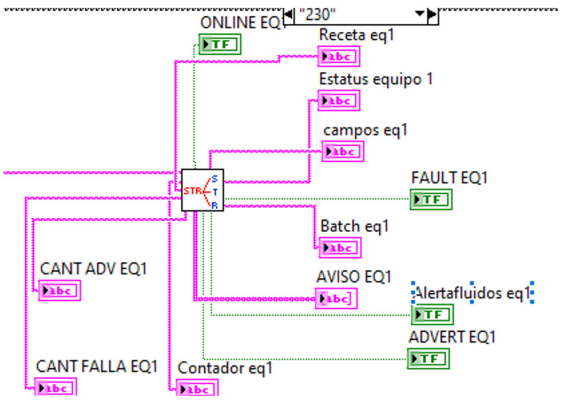


Figura 3.30: Contenedores
Fuente: Los autores

Seguidamente, la información recopilada es enviada a otro subvi llamado "Desencadena Uploadthingsboard.vi", el cual es el encargado de transmitir a la plataforma web thingsboard los datos que se necesiten mostrar, ver figura 3.31.

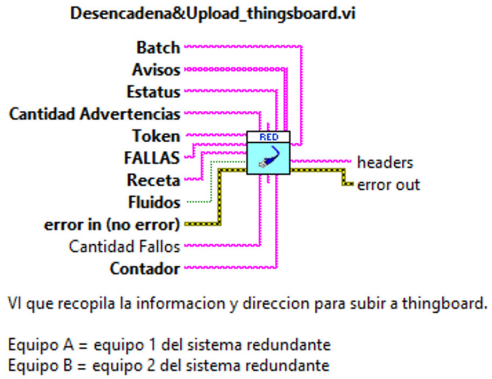


Figura 3.31: Desencadena- Upload Thingsboard
Fuente: Los autores

Este subvi, requiere que los campos de entrada estén siempre con informacion para poder subir los datos a la nube. Uno de los campos importantes de este subvi, es el token el cual debe ser previamente

establecido. Este token se lo obtiene de la plataforma web Thingsboard. El token es la dirección donde se van a enviar las variables y sus respectivos contenidos para ser presentados y cabe recalcar que el Token es un código único por dispositivo. A continuación, la tabla de los token por equipo.

Tabla 3.3: Tokens

Equipo	***Token***
Equipo 1	wymkUP3uXbgAAk9NxlMZ
Equipo 2	9gB6Yb15zhuAQlXLwxw8
Equipo 3	BwHYrI1l6ULLNXcbOwIt
Equipo 4	9qBMzzNtJRdR85RUI9ta
Equipo 5	ypDyOQC4KHXiXoPEg18K
Equipo 6	2DSrKK0DlTD5a124cIzw
Equipo 7	y2ARxgorQSMfyfc0ap26
Equipo 8	VUvm6mQg4b4hPdoPi59q
Equipo 9	gO3DOnMk6WHOdGa1Jdbd

A cada token se le envia un grupo de variables más su respectivo contenido, estas variables representan la información relevante que se quiere proyectar y compartir con otros usuarios y son solo variables de lectura. La actualización de datos en la nube se da inmediatamente después de que los datos han arribado a la intefaz. A continuación la lista de variables que se suben a la nube.

Tabla 3.4: Variable a la nube

Variables a la nube
Receta
Contador
Estado
Batch
Advertencias
Fallos
Avisos
Fluidos
Equipo A
Equipo B
Historial de fallos

La estructura del subvi en cargo de subir los datos a la nube se muestra en la siguiente figura 3.32.

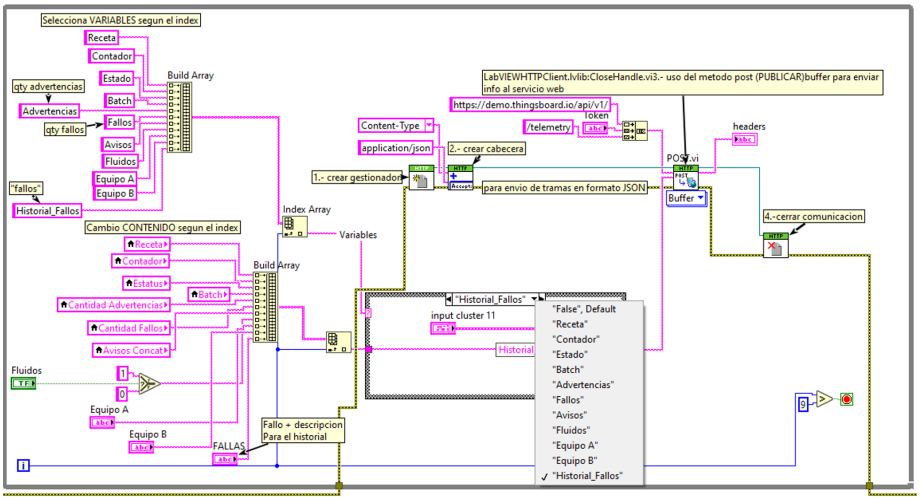


Figura 3.32: Envío de datos a la nube
Fuente: Los autores

Los campos de la interfaz se van completando a medida que la información va llegando en el primer levantamiento de información, posteriormente los datos se actualizan en las siguientes consultas y no desaparecen de la interfaz de manera que el usuario siempre tiene a su vista los datos que se están recopilando con el fin de que tome las acciones preventivas que sean necesarias. De igual forma, en la plataforma web los datos permanecen presentes y son actualizados a medida que se van realizando las respectivas consultas.

A continuación en la figura 3.33 muestra la interfaz con información recopilada tras varios consultas.

Figura 3.33: Envío de datos a la nube
Fuente: Los autores

3.6.6. Envío de trazabilidad

La interfaz gráfica tiene la capacitad de enviar información de trazabilidad al equipo que se escoja (excepto los equipos que trabajan con el sistema redundante). Para efectuar el envío, basta con seguir los siguientes pasos en la interfaz.

1. Apagar las consultas desde el switch. En este modo, el sistema no presentara ninguna actualización de datos.

2. Seleccionar el equipo al que se desea enviar la información

Figura 3.34: Envío de datos a la nube
Fuente: Los autores

3.7. Desarrollo de plataforma web

La plataforma web fue diseñada en el portal de Thingsboard que debido a sus prestaciones se acopla a las necesidades de la presente solución. Para el acceso al diseño de la plataforma web, basta con registrarse y definir un correo y contraseña. El primer paso en la plataforma es la creación de los dispositivos, ver figura 3.35.

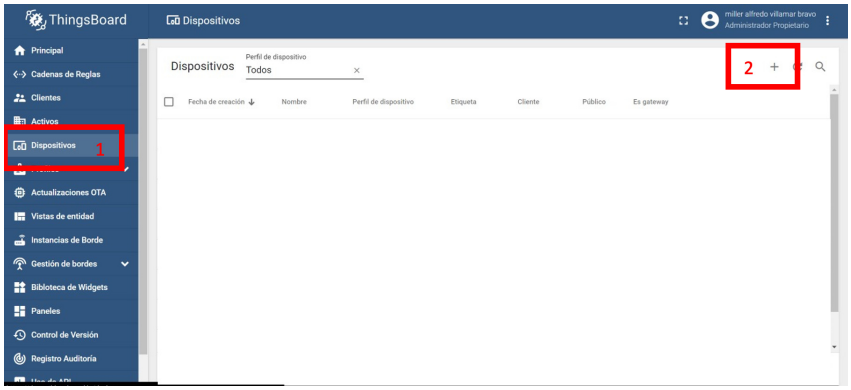


Figura 3.35: Creación de dispositivos
Fuente: Los autores

Una vez creado e dispositivo, podemos tener acceso al token, ver figura 3.36.

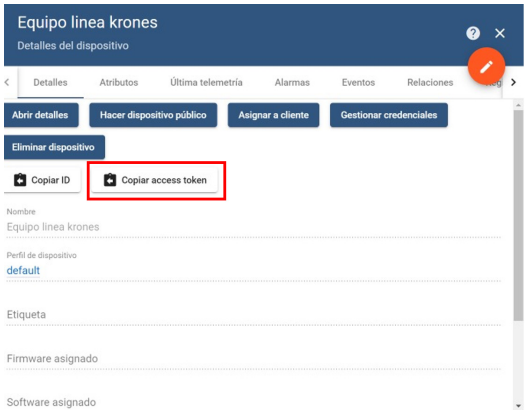


Figura 3.36: Generación de Token
Fuente: Los autores

El token nos permite enviar información hacia el dispositivo para que sea mostrado. Es necesario enviar todas las variables y contenidos antes de continuar. Eso es debido a que se necesitan tener las variables de programa ya registradas en la plataforma.

Una vez que la información se ha enviado a la plataforma, se puede definir los widgets a usarse para mostrar la formación, para esto debemos crear previamente un panel, ver figura 3.37.

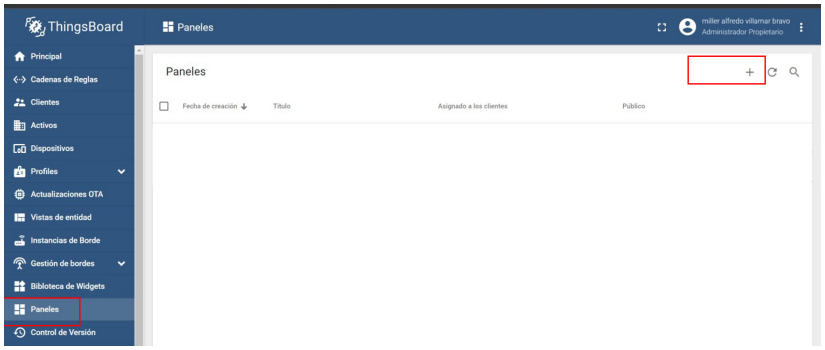


Figura 3.37: Creación de paneles
Fuente: Los autores

Luego seleccionamos y editamos el tablero,ver figura 3.38.

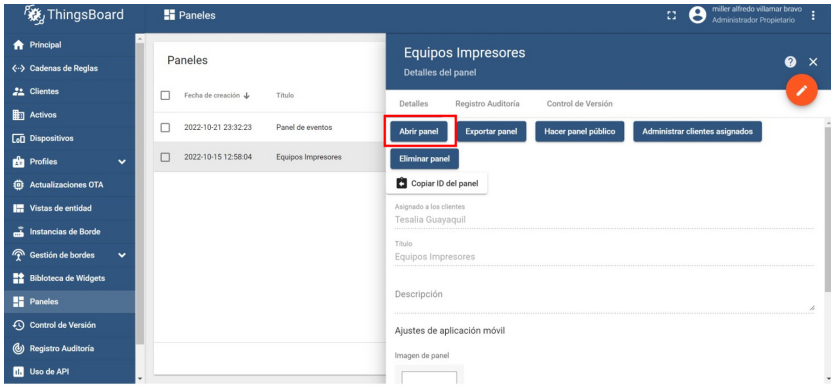


Figura 3.38: Edición de tableros
Fuente: Los autores

Una vez dentro del panel, ya podemos ingresar los widgets requeridos enlazándolos con las variables que se van a presentar, ver figura 3.39.

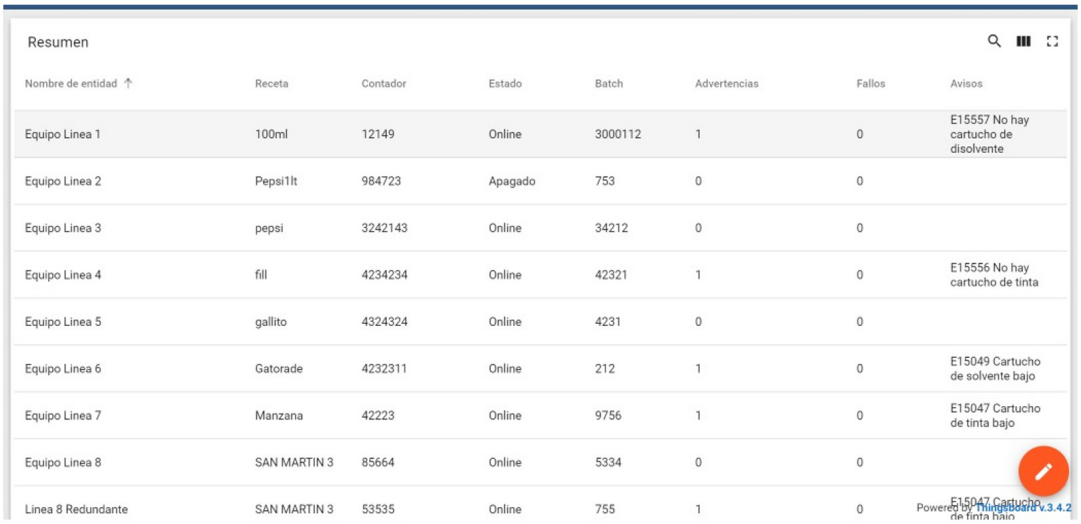


Figura 3.39: Ingreso de widgets
Fuente: Los autores

3.7.1. Widgets

Widgets para presentación de informacion del sistema redundante, ver figura 3.40.

Figura 3.40: Widgets sistema redundante
Fuente: Los autores

Widget para presentación de avisos por consumibles, ver figura 3.41.

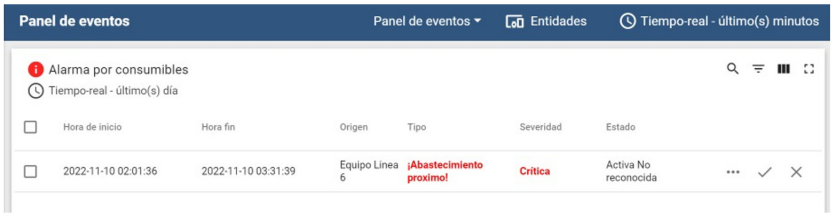


Figura 3.41: Widget por consumibles
Fuente: Los autores

Widget para presentación de ihistoria de fallos, ver figura 3.42.

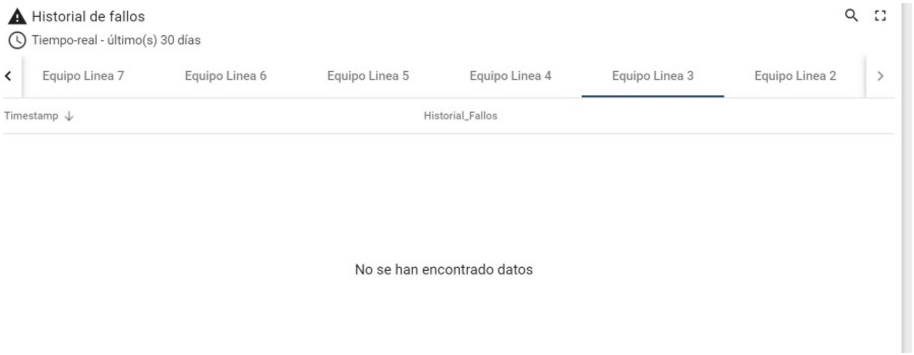


Figura 3.42: Widget historial de fallos
Fuente: Los autores

3.8. Implementación

En esta sección se muestra el montaje en fabrica de los dispositivos electrónicos modulares desarrollados en el proyecto, en la figura 3.43 se muestra la información que muestra en la pantalla el equipo impresor industrial marca Videojet.

Figura 3.43: Impresora Videojet
Fuente: Los autores

En la figura 3.44 se muestra el módulo de equipo remoto para la adquisición y envío de datos obtenidos del equipo impresor,por medio de comunicación LoRa hacia el módulo central conectado al ordenador.

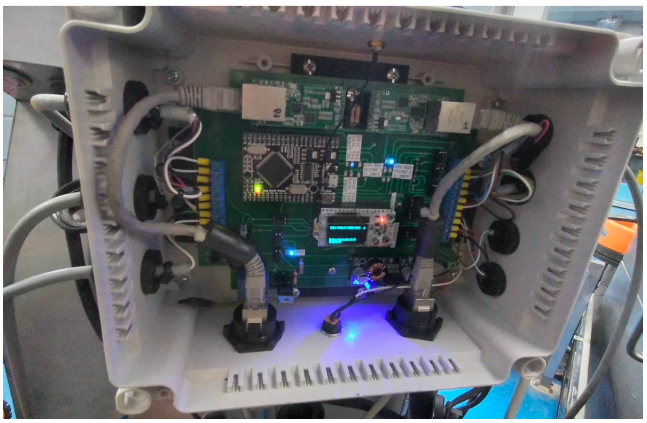


Figura 3.44: Módulo equipo remoto
Fuente: Los autores

En la figura 3.45 se muestra el módulo de sistema de control redundante instalado en línea de producción principal de la fábrica, para el mejoramiento del rendimiento ante fallos y paradas de líneas.

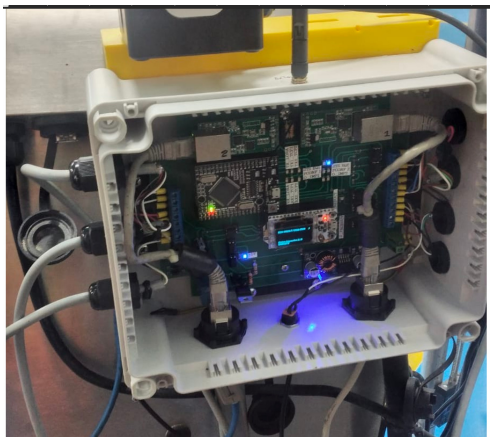


Figura 3.45: Módulo equipo remoto
Fuente: Los autores

3.9. Resultados y conclusiones

3.9.1. Resultados

Uno de los principales resultados positivos que se obtuvieron en la implementación del presente proyecto, en base a la información de los avisos recopilada, se efectúa de manera oportuna los abastecimientos de insumos a los equipos impresores y no de manera correctiva. Esto gracias al uso de tecnología IOT y a los sistemas embebidos que dieron lugar a obtener la información del estado de los equipos en intervalos cortos de tiempo sin la necesidad de estar frente al equipo. El abastecimiento oportuno a los equipos contribuyó a mermar la cantidad de incidencias en la eficiencia de las líneas de producción entre un 70 % y 80 %, pues antes las paradas por faltas de insumos eran de 2 incidencias al día mientras que con la implementación del proyecto se logró que la incidencia por falta de abastecimiento sea de 1 cada 3 días aproximadamente. Contribuyendo al incremento de eficiencia, esta el sistema redundante instalado en la línea de mayor producción en la planta. El sistema redundante incrementó la eficiencia de la línea se redujo a 0 el tiempo de instalación y ajuste del equipo de respaldo puesto que ahora la maquina de respaldo pasa todo el tiempo operando en conjunto con el equipo titular. Datos históricos antes del montaje:

Datos históricos antes de la implementación						
Histórico	Tiempo parada por cambio de máquina en línea de alta producción	Performance p/min	Productos perdidos	Objetivo /hora	Alcanzado /hora	% Eficiencia /hora
Semana 1	15	220	3300	13200	9900	75
Semana 2	20	220	4400	13200	8800	66,7
Semana 3	22	220	4840	13200	8360	63,3
Semana 4	21	220	4620	13200	8580	65
Semana 5	18	220	3960	13200	9240	70

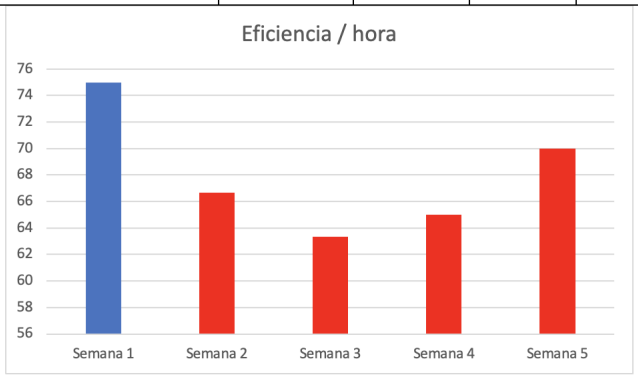


Figura 3.46: Datos históricos antes de la implementación
Fuente: Los autores

Datos obtenidos luego del montaje.

Datos históricos despues de la implementación						
Histórico	Tiempo parada por cambio de máquina en línea de alta producción	Performance p/min	Productos perdidos	Objetivo /hora	Alcanzado /hora	% Eficiencia /hora
Semana 1	0	220	0	13200	13200	100
Semana 2	0	220	0	13200	13200	100
Semana 3	0	220	0	13200	13200	100
Semana 4	0	220	0	13200	13200	100

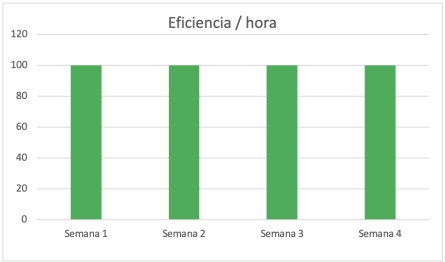


Figura 3.47: Datos obtenidos despues de la implementación
Fuente: Los autores

Con el acceso a la información de trazabilidad y datos de producción (contadores globales y de producción) que otorga la plataforma web se obtiene una lectura mas rápida y exacta de información lo cual es primordial para el personal de producción que pude acceder a la plataforma, ahora se pueden cerrar ordenes de trabajo en menor tiempo.

3.9.2. Conclusiones

- Se concluye que la implementación de tecnología de sistemas embebidos en procesos de lectura de datos, otorga rapidez y confiabilidad en la información. Elimina de necesidad de la intervención operacional para la lectura de datos donde pueden existir errores de lecturas que conllevan a retrasos en el envío de información.

- Hacer uso de IOT a nivel industrial, abre un abanico de posibilidades de implementaciones de mejoras en procesos tanto básicos como complejos. Permite tener en tiempo quasi-real información precisa y critica sin la necesidad de acercarse personalmente al dispositivo terminal .

- Sistemas embebidos y tecnología IOT por si solos ya son herramientas poderosas y útiles, y si a esto se le añade una interfaz para el

procesamiento y gestión de información se logra un sistema robusto con capacidad a cubrir diversas aplicaciones de la industria.

Sistemas embebidos y tecnología IOT por si solos ya son herramientas poderosas y útiles, y si a esto se le añade una interfaz para el procesamiento y gestión de información se logra un sistema robusto con capacidad a cubrir diversas aplicaciones de la industria.

Bibliografía

L. Bassi. Industry 4.0: Hope, hype or revolution? In *2017 IEEE 3rd International Forum on Research and Technologies for Society and Industry (RTSI)*, pages 1–6. IEEE, 2017.

H. Bauer, F. Brandl, C. Lock, and G. Reinhart. Integration of Industrie 4.0 in Lean Manufacturing Learning Factories. *Procedia Manufacturing*, 23 (2017):147–152, 2018. ISSN 23519789. doi: 10.1016/j.promfg.2018.04.008. URL https://doi.org/10.1016/j.promfg.2018.04.008.

J. P. Blázquez. Introducción a los sistemas de comunicación inalámbricos. *Universitat Oberta de Catalunya*, 2015.

Ó. N. Carrasco and A. V. Puerta. Una visión global de la ciberseguridad de los sistemas de control. *Revista SIC: ciberseguridad, seguridad de la información y privacidad*, (106):52–55, 2013.

J. L. del Val Román. Industria 4.0: la transformación digital de la industria. In *Valencia: Conferencia de Directores y Decanos de Ingeniería Informática, Informes CODDII*, 2016.

ESP. ESP32 Series Datasheet. *Espressif Systems*, pages 1–65, 2021. URL https://www.espressif.com/en/support/download/documents.{%}0Ahttps://www.espressif.com/sites/default/files/documentation/esp32{_}datasheet{_}en.pdf.

K. Garg, C. Goswami, R. Chhatrawat, S. Kumar Dhakar, and G. Kumar. Internet of things in manufacturing: A review. *Materials Today: Proceedings*, (xxxx):1–3, 2021. ISSN 22147853. doi: 10.1016/j.matpr.2021.05.321. URL https://doi.org/10.1016/j.matpr.2021.05.321.

A. Garrell. *La Industria 4.0*. ISBN 9788417313852.

K. Gulati, R. S. Kumar Boddu, D. Kapila, S. L. Bangare, N. Chandnani, and G. Saravanan. A review paper on wireless sensor network techniques in Internet of Things (IoT). *Materials Today: Proceedings*, (xxxx), 2021. ISSN 22147853. doi: 10.1016/j.matpr.2021.05.067. URL https://doi.org/10.1016/j.matpr.2021.05.067.

INEN NTE INEN 1334-1:2011 TERCERA REVISION. e p u b l i c o f c u a d o r, 2011.

B. Kada, A. Alzubairi, and A. Tameem. Industrial communication networks and the future of industrial automation. In *2019 Industrial & Systems Engineering Conference (ISEC)*, pages 1–5. IEEE, 2019.

A. E. Kalor, R. Guillaume, J. J. Nielsen, A. Mueller, and P. Popovski. Network slicing in industry 4.0 applications: Abstraction methods and end-to-end analysis. *IEEE Transactions on Industrial Informatics*, 14(12): 5419–5427, 2018. ISSN 15513203. doi: 10.1109/TII.2018.2839721.

A. Luque, M. E. Peralta, A. de las Heras, and A. Córdoba. State of the Industry 4.0 in the Andalusian food sector. *Procedia Manufacturing*, 13: 1199–1205, 2017. ISSN 23519789. doi: 10.1016/j.promfg.2017.09.195. URL https://doi.org/10.1016/j.promfg.2017.09.195.

T. Mitchell, B. Buchanan, G. DeJong, T. Dietterich, P. Rosenbloom, and A. Waibel. Machine learning. *Annual review of computer science*, 4(1): 417–433, 1990.

E. Y. Nakagawa, P. O. Antonino, F. Schnicke, R. Capilla, T. Kuhn, and P. Liggesmeyer. Industry 4.0 reference architectures: State of the art and future trends. *Computers and Industrial Engineering*, 156(September 2020):107241, 2021. ISSN 03608352. doi: 10.1016/j.cie.2021.107241. URL https://doi.org/10.1016/j.cie.2021.107241.

M. Nofer, P. Gomber, O. Hinz, and D. Schiereck. Blockchain. *Business & Information Systems Engineering*, 59(3):183–187, 2017.

D. Pérez. Sistemas embebidos y sistemas operativos embedidos. *Lecturas en ciencias de la computación. Universidad Central de Venezuela, Vols. % i de % 2lSSN*, pages 1316–6239, 2009.

L. Rouhiainen. Inteligencia artificial. *Madrid: Alienta Editorial*, 2018.

F. Rozo-García. Revisión de las tecnologías presentes en la industria 4.0. *Revista UIS Ingenierías*, 19(2):177–191, 2020.

J. Salazar and S. Silvestre. Internet de las cosas. *Techpedia. České vysoké učení technické v Praze Fakulta elektrotechnická*, 2016.

I. Zhou, I. Makhdoom, N. Shariati, M. A. Raza, R. Keshavarz, J. Lipman, M. Abolhasan, and A. Jamalipour. Internet of Things 2.0: Concepts, Applications, and Future Directions. *IEEE Access*, 9:70961–71012, 2021. ISSN 21693536. doi: 10.1109/access.2021.3078549.