UNIVERSIDAD TÉCNICA FEDERICO SANTA MARIA
SEDE CONCEPCIÓN – REY BALDUINO DE BÉLGICA

SUPERVISIÓN Y CONTROL DE PLANTA DE LORENZO DL 2314 EN ANDROID CON ARDUINO MEDIANTE ETHERNET TCP

Trabajo de Titulación para optar al Título
de Técnico Universitario en
AUTOMATIZACIÓN Y CONTROL

Alumno:
Jonathan Sanhueza Riffo

Profesor Guía:
Erwin Alarcón I.

2018

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARIA
SEDE CONCEPCIÓN – REY BALDUINO DE BÉLGICA

SUPERVISIÓN Y CONTROL DE PLANTA DE LORENZO DL 2314 EN ANDROID CON ARDUINO MEDIANTE ETHERNET TCP

Trabajo de Titulación para optar al Título
de Técnico Universitario en
AUTOMATIZACIÓN Y CONTROL

Alumno:
Jonathan Sanhueza Riffo

Profesor Guía:
Erwin Alarcón I.

2018

DEDICATORIA

Quiero agradecer en primer lugar a Dios por permitirme cerrar esta etapa, etapa que en muchos momentos estuvo cuesta arriba, sin embargo, gracias a mi FE en él he podido lograrlo. A mi pareja, pilar fundamental en mi vida, hoy por hoy se ha convertido en el motor que mueve cada una de las acciones que realizo, su apoyo incondicional en este ciclo y en cada decisión de mi vida, me ha dado la fortaleza para seguir desarrollándome como persona y profesional.

No puedo dejar de mencionar a mi familia, mis padres, otro pilar fundamental, nunca han dejado de creer en mí, a pesar de lo inalcanzable que se vio en algún minuto, su confianza en mí no decayó y siempre estuvieron ahí con una palabra de aliento. A mi hermano, persona que nunca ha dudado de mis capacidades como estudiante y profesional.

Una mención especial también a mis suegros, quienes se han convertido en mi segunda familia, y su apoyo sin dudas se agradece.

A mi profesor Guía, Don Erwin Alarcón, gracias por sus consejos, por su paciencia y también por creer en mí pese a la demora en la entrega, gracias.

Por último, quiero dedicar este logro a mi mismo, ya que en algún momento dudé de mis capacidades, sin embargo, con el esfuerzo y el apoyo que tuve logré culminar este ciclo, ciclo que significó mucho para mí. Gracias.

Jonathan Rodrigo Sanhueza Riffo.

RESUMEN

El presente documento detalla el procedimiento y las consideraciones utilizadas para llevar a cabo el diseño e implementación de un sistema de control, entre la Planta De Lorenzo DL 2314, ubicada dentro del Laboratorio de Procesos de la USM, y dispositivos inteligentes Android. Para ello se comienza abarcando la teoría relacionada a Ethernet TCP, Hardware Arduino, Ethernet Shield y al Sistema Operativo Android, para posteriormente centrarnos en la explicación e implementación del proyecto.

Dentro de este escrito se exponen las consideraciones de su implementación y funcionamiento. Además, de sus respectivas pruebas de funcionamiento que permitieron su desarrollo. De la misma manera, se detallan las configuraciones previas de los dispositivos, necesarias para la óptima comunicación entre la interfaz de usuario MIT APP Inventor 2 y la Planta De Lorenzo DL 2314.

INDICE

INDICE TABLAS

INDICE FIGURAS

GLOSARIO

Proceso: Conjunto de acciones integradas y dirigidas hacia un fin.

Control: El acto o el poder de dominar, dirigir, regular, comandar, controlar.

Sensor: Dispositivo capacitado para detectar acciones o estímulos externos y responder en consecuencia, pueden transformar las magnitudes físicas o químicas en magnitudes eléctricas.

Actuador: Dispositivo que transforma señales eléctricas en diferentes tipos de energía, produciendo un efecto sobre un proceso (normalmente bajo un esquema de control).

Variable: Magnitud física que puede tomar varios valores distintos en el tiempo.

Sistema: Conjunto de reglas, principios, ideas o cosas, que están unidas por un criterio común y tienen una finalidad determinada.

Programación: Acción de programar una computadora.

Lazo de control: Es el conjunto de dispositivos capaces de realizar un sistema de control automático.

Nivel: Variable de proceso de medición referencial, que mide la línea de la superficie liquida respecto a una referencia en función de las características del líquido o solidos a medir.

Caudal o Flujo: cantidad de fluido que circula a través de una sección de un ducto por unidad de tiempo.

Temperatura: Magnitud física que indica la intensidad de calor o frío de un cuerpo.

Presión: Fuerza que ejerce un gas, un líquido o un sólido sobre una superficie.

Interfaz de usuario: Medio con que el usuario puede comunicarse con una máquina, equipo, computadora o dispositivo.

SIMBOLOGÍA

D.N: Divisor de Nivel.

D.T: Divisor de Temperatura.

D.P: Divisor de Presión.

C.N: Conector de Nivel.

C.F: Conector de Flujo.

C.T: Conector de Temperatura.

C.P: Conector de Presión.

GND: Ground o Tierra de Protección.

V: Voltaje, su unidad de medida es el volt [v].

V_{IN}: Voltaje de Entrada.

V_{OUT}: Voltaje de Salida.

R: Resistencia Eléctrica, su unidad de medida es el ohm [Ω]

I: Corriente eléctrica, unidad de medida Amper [A]/[mA].

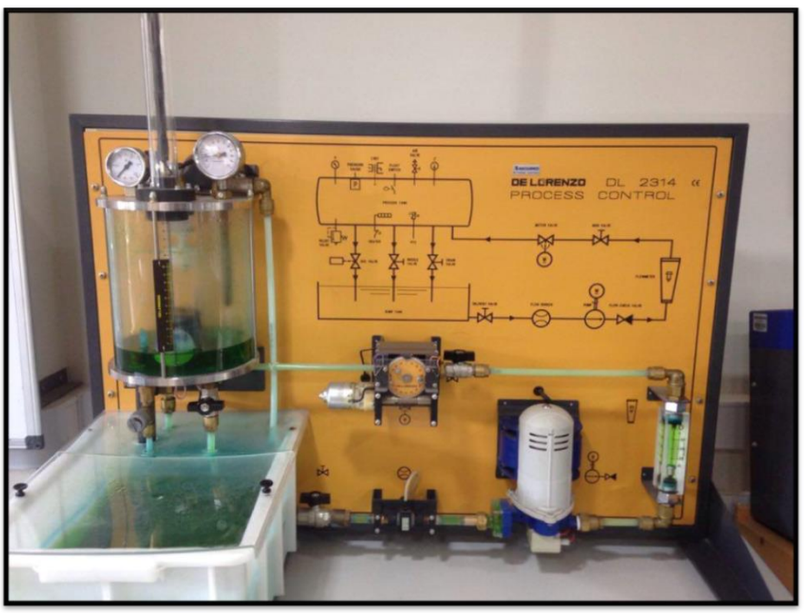
FS: Factor de Seguridad.

ß**:** Ganancia de un transistor.

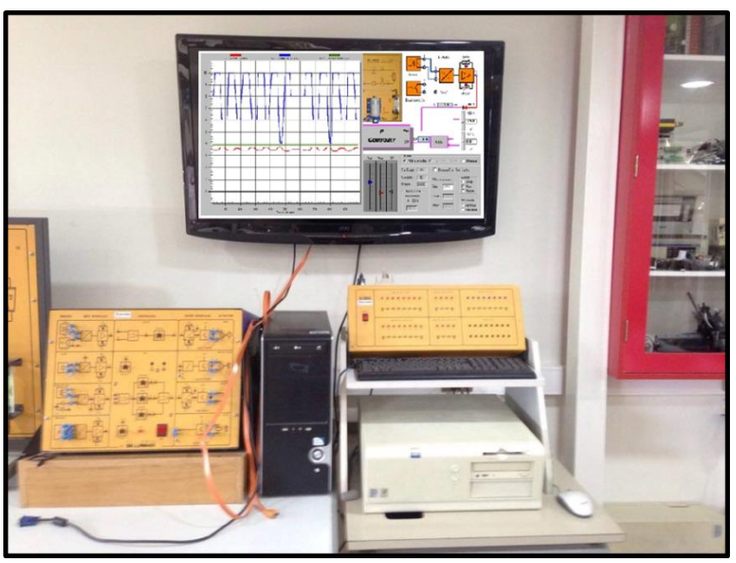
INTRODUCCIÓN

La Planta Piloto De Lorenzo se caracteriza por ser una unidad didáctica y compacta, creada para fines pedagógicos, cuyo objetivo principal es el estudio de variables y lazos de control típicos que intervienen en los procesos industriales. Por ej. Nivel, Flujo, Temperatura y Presión.

Actualmente su operación solo puede ser realizada desde la estación de operación, motivo por el cual, la supervisión y el control se ven limitados.



Planta Didáctica De Lorenzo DL 2314.



Estación de Operación

PROPUESTA SOLUCIÓN

Para lograr una supervisión y control de las variables y lazos de control típicos de la Planta De Lorenzo DL 2314; además de proporcionar otro método de control más amigable, se propone la implementación de un sistema, utilizando la plataforma de hardware Arduino, Ethernet shield, Router Wi-Fi, MIT APP Inventor 2 y dispositivos inteligentes Android.

OBJETIVO GENERAL

Diseñar e Implementar una Aplicación para Dispositivos Inteligentes Android, que permita la Supervisión y el Control Remoto de todas las variables asociadas a los Lazos de Control de la Planta De Lorenzo DL 2314.

OBJETIVOS ESPECÍFICOS

- Levantamiento de las variables de entrada/salida (I/O) disponibles en la Planta De Lorenzo DL 2314.
- Estudio y recopilación de información de Hardware Arduino y Ethernet Shield.
- Estudio y recopilación de información de Amplificadores de Potencia y Transistores (Electrónica Análoga-Digital).
- Estudio de Software MIT APP Inventor 2.
- Diseño, implementación y configuración de la arquitectura de red.

ALCANCE

- Supervisión y control de todas las variables asociadas a los lazos de control de la Planta De Lorenzo DL 2314.
- Lectura de todos los sensores disponibles en la Planta De Lorenzo DL 2314.
- Acciones de comando sobre todos los actuadores disponibles en la Planta De Lorenzo DL 2314.
- Aplicación desarrollada y orientada para dispositivos inteligentes Android.
- Elaboración de Interfaz de Usuario.
- Elaboración de planos:
    - Diagrama de Conexionado de acondicionadores de señal
    - Interconexión de las Señales de Entrada y Salida.

EXCLUSIONES

- La aplicación no incluirá ningún tipo de algoritmo de control automático. Por ej. ON – OFF y PID.
- Se excluye el análisis de tendencias de cada variable medida.
- Se excluye el almacenamiento de históricos y despliegue de alarmas.

CAPITULO 1:

TEORÍA RELACIONADA

1. TEORÍA RELACIONADA

En este capítulo se abarca la teoría relacionada al proyecto, particularmente orientado a los protocolos de transmisión, al hardware y software asociado, bases para el análisis y desarrollo del mismo.

1.2 MODELO OSI

El modelo OSI (Open Systems Interconnection) (ISO/IEC 7498-1). Caracteriza y estandariza las funciones de un sistema de comunicaciones en términos de abstracción de capas. Cada capa se encarga de ejecutar una determinada parte del proceso global.

Eventos que abarca el Modelo OSI:

- Modo en que los datos se traducen de la manera apropiada para la arquitectura de red que se está ocupando.
- Modo en que los dispositivos de red se comunican.
- Modo en que se trasmiten los datos y comprobación de errores entre los distintos dispositivos de la red.
- Modo en que el direccionamiento lógico se convierte en el direccionamiento físico que otorga la red.

1.1.1 Capas del modelo OSI

1. Capa Física.

Define las especificaciones eléctricas y físicas de los dispositivos que se conectan a la red (cables, hubs, repetidores, adaptadores de red).

2. Capa de Enlace de Datos.

Provee los medios funcionales y procedimientos para transferir información entre entidades de red, además detecta y posiblemente corrige errores que puedan ocurrir en la capa física.

Funciones:

- Framing.
- Direccionamiento físico.
- Control de flujo.
- Control de errores.
- Control de acceso.
- Media Access Control (MAC).

3. Capa de Red.

Se encarga de transferir las secuencias de datos desde un servidor de origen de una red, hacia un servidor de destino de otra red. Rol importante cumplen los conocidos router, quienes se hacen presente realizando funciones de ruteo y enviando datos a través de la red.

4. Capa de Transporte.

Encargada de controlar el flujo de datos entre los nodos que establecen una comunicación, ofrece un servicio confiable para las otras capas, además es quien controla la segmentación y control de errores.

5. Capa de Sesión.

Controla los diálogos entre los dispositivos, establece, administra y termina las conexiones entre las aplicaciones locales y remotas. Provee operaciones full-duplex, half-duplex y simplex. Además, es responsable del cierre de sesiones correctas, que es una propiedad del protocolo de control de transmisión (TCP) y comúnmente se implementa en aplicaciones con ambientes que usan llamadas de procedimientos remotos.

6. Capa de Presentación.

Esta capa toma los paquetes de la capa de aplicación y los convierte a un formato genérico que pueden leer todos los dispositivos, además se encarga de reducir el tamaño de estos, ya sea cifrándolos o comprimiéndolos.

7. Capa de Aplicación.

Proporciona la interfaz y servicios que soportan las aplicaciones de usuario, también ofrece los servicios de red relacionados con estas aplicaciones, como la gestión de mensajes, la transferencia de archivos y las consultas a base de datos.
Entre los servicios de intercambio de información que gestiona la capa de aplicación se encuentran los protocolos SMTP, Telnet, ftp y http.

Tabla 1-1: Protocolos relacionados con el Modelo OSI

Capa 1	Nivel Físico	Cable Coaxial, Cable Par Trenzado.
Capa 2	Nivel de Enlace de Datos	Ethernet, Wireless Ethernet, Token Ring
Capa 3	Nivel de Red	Modems, Routers.
Capa 4	Nivel de Transporte	TCP, UDP
Capa 5	Nivel de Sesión	NetBIOS
Capa 6	Nivel de Presentación	ASN.1
Capa 7	Nivel de Aplicación	SMTP, FTP, HTTP

1.2 MODELO TCP/IP

Los niveles del modelo OSI; Aplicación, Presentación y Sesión, son considerados como el nivel de Aplicación en el conjunto TCP/IP.

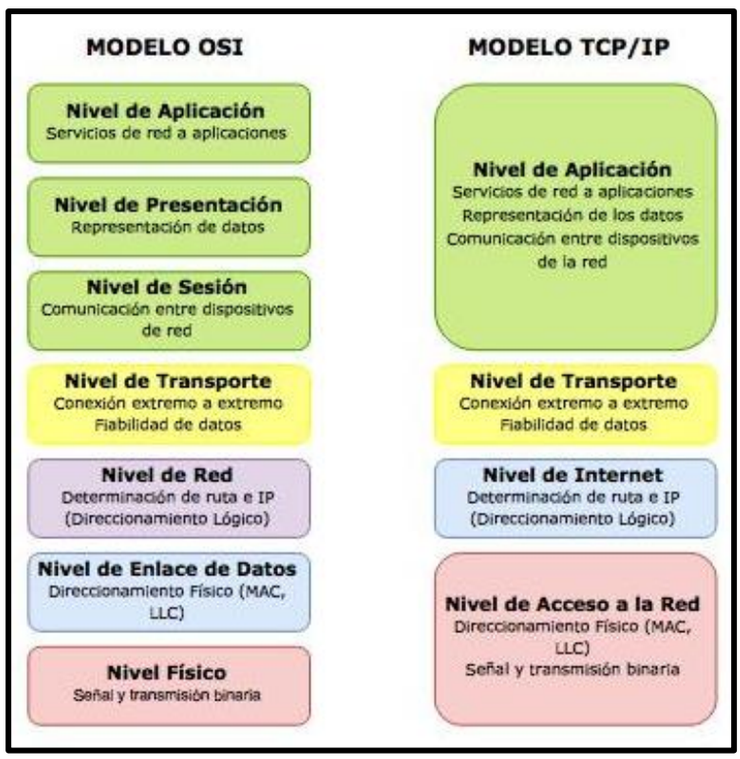


Figura 1-1: Modelo OSI y TCP/IP.

1.2.1 Capas del modelo TCP/IP

1. Capa de Acceso a la red.

Capa que maneja todos los aspectos que un paquete IP requiere para efectuar un enlace físico real con los medios de la red, además incluye los detalles de la tecnología LAN, WAN y de las capas físicas y de enlace de datos del modelo OSI.

Funciones:

- Asignación de direcciones IP a las direcciones físicas.
- Encapsulamiento de paquetes IP en tramas.
- Define la conexión con los medios físicos de la red.

2. Capa de Internet.

Capa que tiene como objetivo seleccionar la mejor ruta para enviar paquetes por la red, siendo su principal protocolo el conocido Protocolo de Internet (IP).

3. Capa de Transporte.

Provee información desde el servidor de origen hacia el servidor de destino (extremo a extremo), regulando el flujo de información y corrigiendo posibles errores, lo que permite que el servicio de transporte sea confiable. Comprende a los protocolos TCP y UDP.

4. Capa de Aplicación.

Nivel que utilizan los programas para comunicarse a través de una red con otros programas. Los procesos que acontecen, son aplicaciones específicas que pasan los datos al nivel de aplicación en el formato que internamente use el programa y es codificado de acuerdo con un protocolo estándar.

Proporcionan servicios que directamente trabajan con las aplicaciones de usuario. Estos programas y sus correspondientes protocolos incluyen a HTTP (HyperText Transfer Protocol), FTP (Transferencia de archivos), SMTP (Correo Electrónico), SSH (Login Remoto Seguro), DNS (Resolución de Nombres de Dominio) y a muchos otros.

1.3 ESTÁNDAR ETHERNET

La especificación IEEE para Ethernet es la 802.3, que define los tipos de cables permitidos y cuáles son las características de la señal que se transporta. Actualmente son 4 los tipos de velocidad definidos para Ethernet con cables de fibra óptica y/o par trenzado:

- Ethernet: Hasta 10 Mb/s.
- Fast Ethernet: Hasta 100 Mb/s.
- Gigabit Ethernet: Hasta 1000 Mb/s.
- 10 Gigabit Ethernet.

Debido a la variedad de implementaciones de este estándar, se utiliza una fórmula general del tipo N Base T, que tiene como características principales:

- La tasa de transferencia de datos en Mb/s.
- Tipo de señalización utilizada.
- La máxima longitud del segmento de acuerdo al tipo de cable utilizado.

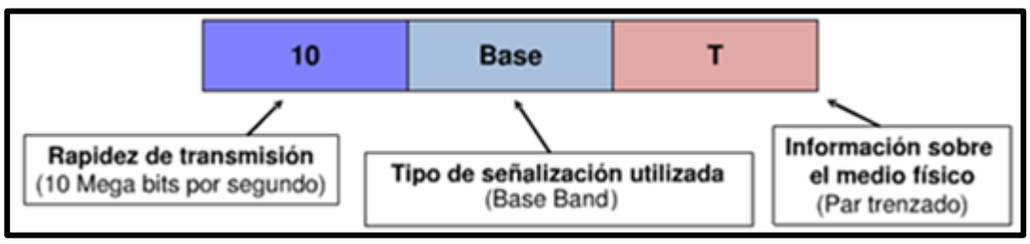


Figura 1-2: Notación Ethernet.

Donde:

- N (10) = Velocidad de transmisión de datos (Mb/s) en el canal.
- Base: Método de modulación empleado (Base-Band Modulation).
- T: Longitud máxima de cada segmento, o tecnología empleada señalada con una letra (Fibra (F), Par trenzado (T)).

Tabla 1-2: Tipos de Ethernet.

Tipo de Ethernet	**Ancho de Banda**	**Tipo de Cable**	**Dúplex**	**Distancia máx.**
10 Base-5	10 Mbps	Coaxial ThickNet	Half	500 m
10 Base-2	10 Mbps	Coaxial Thinnet	Half	185 m
10 Base-Tx	10 Mbps	UTP Cat3/Cat5	Half	100 m
100 Base-Tx	100 Mbps	UTP Cat5	Half	100 m
100 Base-Fx	100 Mbps	Fibra Multimodo	Half	400 m
1000 Base-T	1 Gbps	UTP Cat5e	Full	2 Km
1000 Base-Tx	1 Gbps	UTP Cat5e	Full	100 m
1000 Base-Sx	1 Gbps	Fibra Multimodo	Full	100 m
1000 Base-Lx	1 Gbps	Fibra Monomodo	Full	550 m
10G Base-Cx4	1 Gbps	Twinaxial	Full	2 Km
10G Base-T	10 Gbps	UTP Cat6a/Cat7	Full	100 m
10G Base-Lx4	10 Gbps	Fibra Multimodo	Full	100 m
10G Base-Lx4	10 Gbps	Fibra Monomodo	Full	300 m

1.4 MEDIOS DE TRANSMISIÓN ETHERNET

Para contextualizar, un medio de transmisión es el canal por el cual viajan los paquetes de datos, pudiendo ser cables metálicos, fibra de vidrio o espectro electromagnético, estos se clasifican en:

- Medios de Cobre.
- Medios de Fibra.
- Medios Inalámbricos.

1.4.1 Medio de Cobre

Es el medio más utilizado, se pueden clasificar en distintas categorías que van desde la 1 a la 7, siendo actualmente las categorías 5, 6 y 7 las de uso regular, sin embargo, esta última es utilizada para redes más robustas donde se requiere el mínimo de interferencias posibles. Existen otras derivaciones a raíz de estas, a las cuales se le agrega una letra a la categoría, lo que significa que añade otras características. (Ejemplo 5e).

Tabla 1-3: Categorías más usadas.

Categoría	Distancia [m]	Velocidad Máx. [Mb/s]	Frecuencia [Mhz]
CAT. 5	100	100	100
CAT. 5e	100	1000	100
CAT. 6	100	1000	250
CAT. 6a	100	100000	500

1.4.1.1 Cable Par Trenzado

Es un tipo de cable que en su forma simple posee 2 conductores eléctricos aislados y que se entrelazan para anular las interferencias producidas por fuentes externas, su uso es adecuado para redes del tipo local con pocos nodos y una conectividad simple. Se distinguen 3 tipos de cables de par trenzado, el cable par trenzado apantallado individual STP (Shielded Twisted Pair), el par trenzado apantallado FTP (Fully Twisted Pair) y el no apantallado UTP (Unshielded Twisted Pair), siendo este último el común conocido.

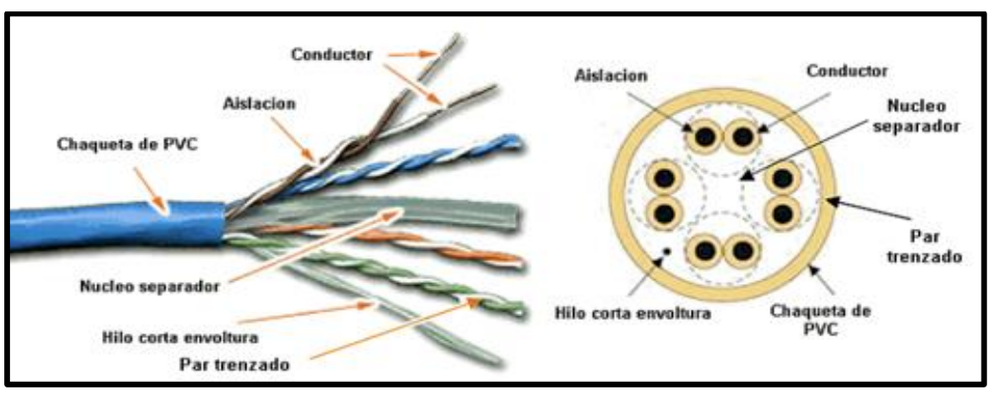


Figura 1-3: Par Trenzado UTP.

El Conector típico y más usado en las redes Ethernet, es el conector RJ45, se clasifica por categoría y existen distintos tipos de acuerdo a su grado de protección requerida. Posee 8 pines y la disposición de estos depende del estándar TIA/EIA-568-B, y sus hilos varían según la norma empleada (A o B).

Figura 1-4: Conector RJ45 para redes locales.

Existen 2 tipos de configuraciones para la conexión con el cable de par trenzado, las cuales son:

- Configuración Directa: Posee la misma distribución de hilos en ambos extremos, se utiliza para interconexión de equipos desiguales.

- Configuración Cruzada: Posee un estándar de distribución para cada extremo, se utiliza para la interconexión de dispositivos electrónicos con comunicación full dúplex.

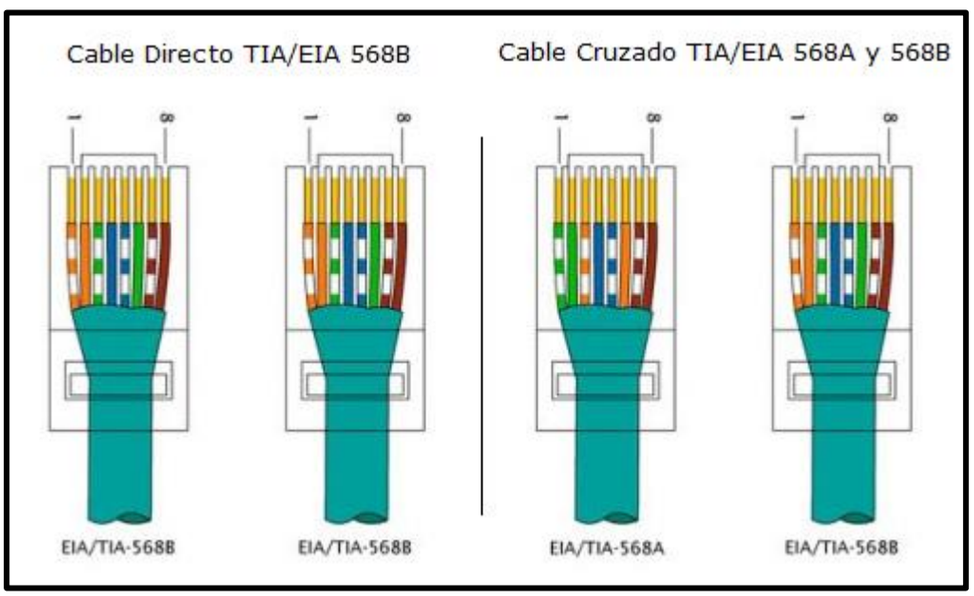


Figura 1-5: Conexión Directa y Cruzada, RJ45 con Cable Par Trenzado.

1.4.2 Medio Inalámbrico

Posee 4 estándares comunes de comunicación de datos, los cuales son:

- IEE estándar 802.11: Tecnología LAN Inalámbrica, comúnmente denominada WI-FI.
- IEE estándar 802.15: Red de área personal inalámbrica WPAN, comúnmente denominada Bluetooth.
- IEE estándar 802.16: comúnmente conocido como WiMAX.
- Sistema Global para Comunicaciones Móviles GSM.

1.5 PROTOCOLOS DE TRANSMISIÓN

Son principalmente las reglas que se deben seguir para poder realizar una correcta comunicación, lo que hacen es reducir el tamaño de los paquetes, separándolos en pequeñas partes y ordenándolas en grupo para su posterior envío. El receptor al recibir estos datos debe enviar un mensaje para corroborar la correcta transmisión, si el paquete llega dañado, el receptor avisa para que envíen nuevamente la información y así sucesivamente hasta lograr completar el envío correctamente.

1.5.1 TCP/IP

Conjunto de protocolos, de siglas TCP (Protocolo de control de transmisión) / IP (Protocolo de internet), que tiene directa relación con el modelo OSI y el modelo TCP/IP. Representan las reglas de comunicación para internet, siendo la base de éste, el concepto de direcciones IP.

Básicamente lo que hace es otorgar una dirección IP a todos los equipos que pertenezcan a la red para poder enrutar paquetes de datos.

Criterios:

- Dividir mensaje en paquetes.
- Usar sistema de direcciones.
- Enrutar datos por la red.
- Detectar errores en la transmisión de datos.

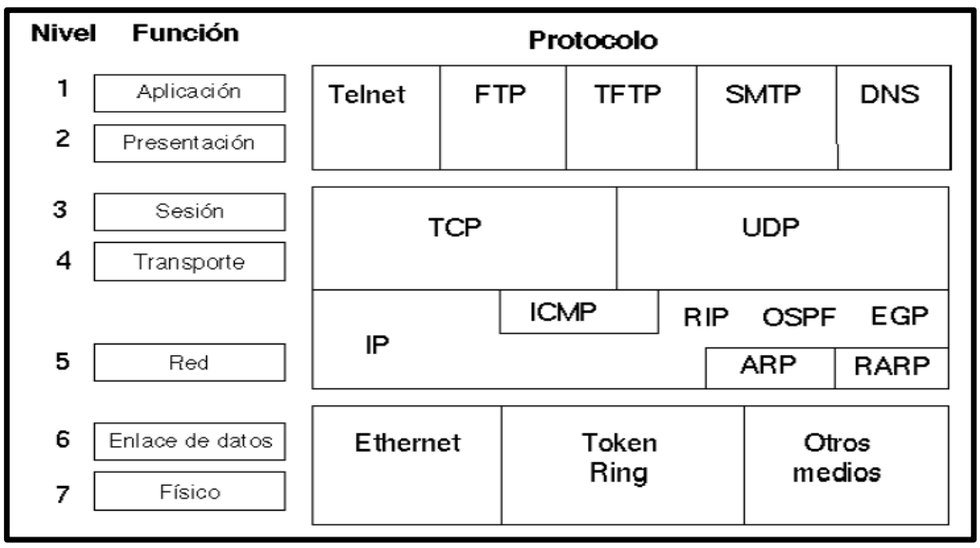


Figura 1-6: Modelo de referencia OSI y las capas de TCP/IP correspondientes.

1.5.1.1 Nivel de transporte

El nivel de transporte del conjunto de protocolos TCP/IP consta de dos protocolos: el Protocolo de datagramas de usuario (UDP) y el Protocolo de control de transmisión (TCP). El protocolo UDP proporciona un servicio de entrega sin conexión y poco fiable para enviar y recibir mensajes y el protocolo TCP incorpora servicios de entrega fiable al servicio de entrega de datagramas IP.

1.6 HARDWARE

1.6.1 Arduino

Arduino es una plataforma de prototipos electrónica de hardware libre y código abierto (OPEN-SOURCE), que está basada en un microcontrolador Atmel AVR, siendo los más utilizados el ATmega168, ATmega328 y ATmega1280, su entorno de desarrollo utiliza el lenguaje Processing/Wiring, lo que proporciona una plataforma amigable y fácil de usar.

Arduino Uno Rev3. posee 14 pines digitales de entrada y salida, 6 de los cuales pueden ser configurados para realizar control por ancho de pulso (PWM), 6 entradas análogas, una conexión USB, un Jack de poder y un botón de reseteo.

La placa puede ser alimentada de varias formas, con un cable USB conectado al computador, o con una fuente externa conectada al Jack de poder utilizando un voltaje recomendado de entre 7 a 12 [v].

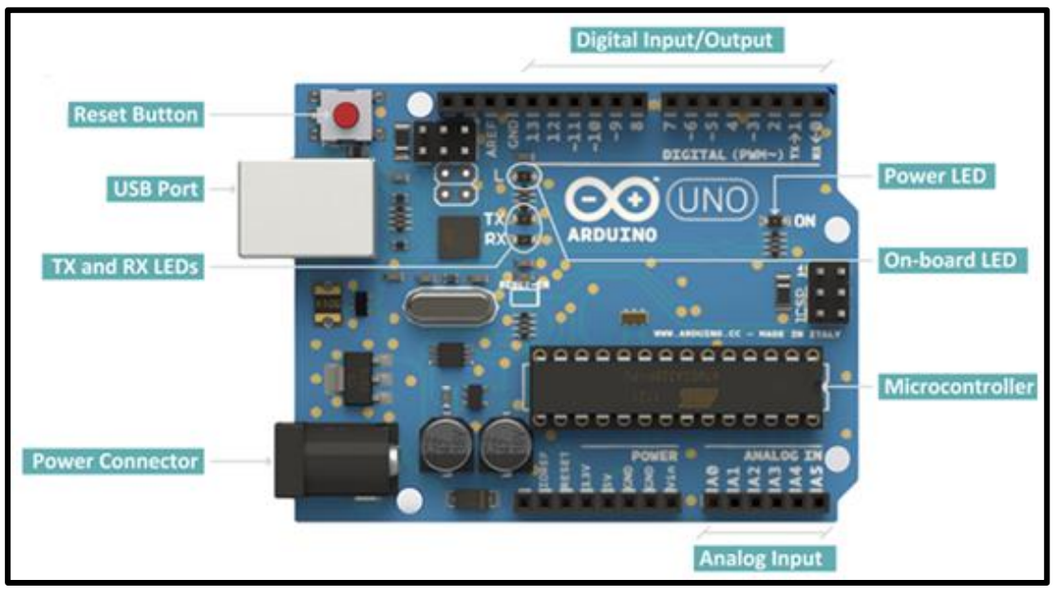


Figura 1-7: Placa Arduino uno Rev3.

1.6.1.1 Pines Arduino uno Rev3

- Entradas y Salidas Digitales: Las entradas y salidas digitales son 14, desde el pin 0 al 13 y ofrecen una tensión de 5[V] y una corriente DC de 20[mA].

- Pin VIN: Si tenemos una fuente de alimentación conectada mediante un adaptador, se puede obtener la alimentación para conectar otro dispositivo, pero hay que tener en cuenta que la placa no regulará la tensión y obtendremos el mismo voltaje que tenga el adaptador. Por otro lado, si tenemos conectado el USB, la tensión será regulada a 5[v]. Si tenemos una fuente de alimentación externa como por ejemplo baterías, el borne positivo de esta irá conectado al pin VIN y el borne negativo al pin GND, si la batería otorga 10[v] la placa regulará la tensión a 5[v]

- Pin GND: Pin GND es la tierra de Arduino.

- Pin 5V: Podemos alimentar la placa mediante este pin, siempre que tengamos la fuente externa regulada a 5[v]. Por otro lado, si tenemos la placa alimentada tanto por el Jack como por USB, se puede alimentar otro componente con una tensión regulada de 5[v]

- Pin 3.3V: Este pin nos otorga un voltaje de 3.3[v], ya sea al estar alimentado por el USB o mediante el Jack de poder. Generalmente se utiliza para alimentar componentes que requieren baja tensión.

- Pines de Entradas Analógicas: Los pines de entrada analógicos son 6 y van desde el pin A0 hasta el A5, estas entradas tienen una resolución de 10 bits, es decir, utiliza valores entre 0 y 1023, y realiza una lectura de voltaje que va de 0[v] hasta 5[v], aunque es posible cambiar este rango utilizando una función con el pin AREF.

- Pin AREF: Pin que ofrece un voltaje de referencia para las entradas análogas.

- Pin IOREF: El pin IOREF es una copia del pin VIN y se utiliza para indicar a los demás dispositivos conectados a la placa que las tensiones de los pines de entrada y salida son 5[v].

- Pin Reset: Pin que cumple la misma función que el botón reset, la cual es reiniciar el microcontrolador.

- Pines A5 SCL y A4 SDA: Se utilizan para conectar dispositivos que realizan comunicación mediante la librería Wire.

- Pin 1 TX y 0 RX: Pines utilizados para recibir y transmitir en serie.

Tabla 1-3: Características Técnicas Arduino Uno Rev3.

Microcontrolador	ATmega328P
Voltaje de Funcionamiento	5 [V]
Voltaje de Entrada Recomendado	7-12 [V]
Voltaje de Entrada Límite	6-20 [V]
Pines Digitales I/O	14
Pines Digitales PWM I/O	6
Pines de Entrada Analógica	6
Corriente DC por Pin I/O	20 [mA]
Corriente DC Pin 3.3 Volts	50 [mA]
Memoria Flash	32 Kb ATmega328P
SRAM	2 Kb ATmega328P
EEPROM	1 Kb ATmega328P
Velocidad de Reloj	16 MHz
Longitud	68.6 [mm]
Anchura	53.4 [mm]
Peso	25 g.

Una de las tantas ventajas de este hardware, es la posibilidad de expandir sus funciones para otros fines requeridos, ya sea dotándolo de conexión Ethernet, WiFi, Bluetooth y/o sensores a distancias, etc. Todo esto gracias a las placas conocidas como shields.

Figura 1-8: Conexión de un Shield a Placa Arduino.

1.6.2 Ethernet Shield

El Ethernet shield es quien posibilita que Arduino Uno Rev3 se pueda conectar a una red Ethernet, gracias al chip Wiznet W5100 que implementa la pila de protocolos TCP/IP, que soporta tanto al protocolo TCP como al UDP. Hay que considerar que mientras están en comunicación, Arduino UNO hace uso de los pines digitales 10, 11, 12 y 13 (Bus SPI), lo que se traduce en una reducción de salidas.

Para que el shield establezca comunicación con la red, se debe conectar a un router para que éste le asigne una dirección IP, esto se logra gracias a que dispone de un puerto Ethernet, donde se conecta un cable Ethernet normalmente de categoría 5 o 6 (CAT.5, CAT.6) con conectores rj45. Otro factor necesario, es el incluir la librería ethernet.h en la programación del IDE, para conseguir una correcta lectura y escritura de datos.

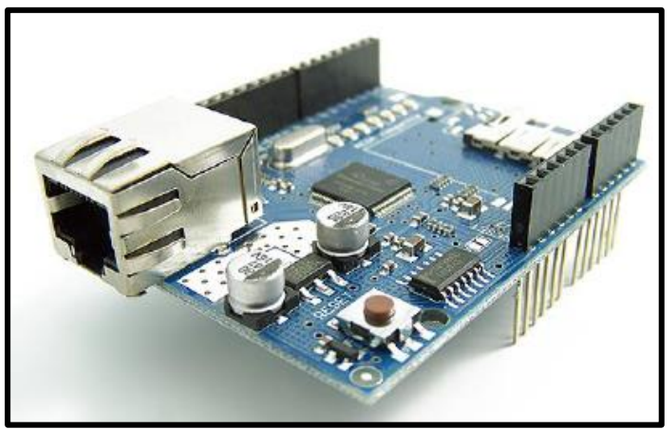


Figura 1-9: Ethernet Shield.

Características importantes:

- Voltaje de operación 5[v].
- El chip W5100 posee 16K de Buffer interno y no consume memoria.
- Incluir librería SPI.h para comunicación con arduino.
- Incluir librería Ethernet.h para manejo del shield.
- Soporta hasta 4 conexiones simultáneas.
- Dispone de un slot para tarjetas de memorias micro-SD.
- Ethernet y SD no pueden trabajar simultáneamente.

1.6.3 Software IDE

Entorno de desarrollo integrado (siglas en inglés de Integrated Development Environment), basado en el entorno de processing y lenguaje de programación basado en Wiring.

El IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación; es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). El microcontrolador se programa por medio de un computador, usando una comunicación serial mediante un convertidor de niveles RS-232 a TTL serial.

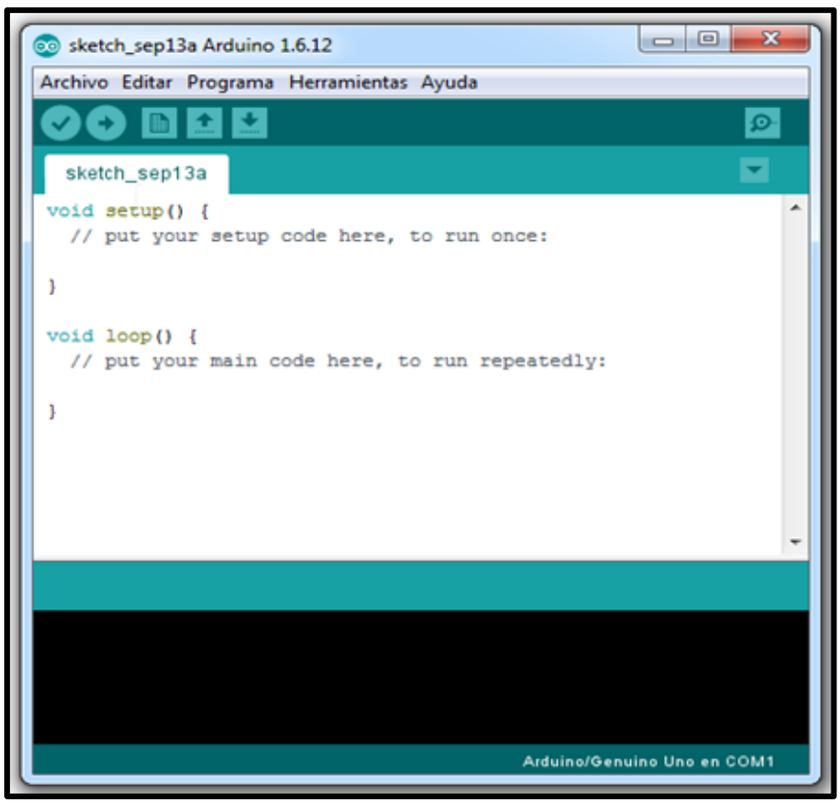


Figura 1-10: Software IDE.

1.7 PLATAFORMA MÓVIL

Las plataformas móviles son la base para el funcionamiento de cualquier dispositivo móvil. Proporcionan el SDK (Software Development Kit), herramientas y el sistema operativo que permiten el desarrollo de aplicaciones para determinada plataforma. Por lo general, éstas tienen su propio modelo de distribución.

Por ejemplo, Google/Android tiene Play Store, Windows Phone tiene Windows Marketplace e iOS posee el AppStore. Las aplicaciones móviles se desarrollan para una diversidad de plataformas móviles como Android, iOS (iPhone), Symbian (Nokia), Windows Mobile (Microsoft), Blackberry (SonyEricsson), etc.

Desde la introducción de iOS (2007) y Android (2008) como plataformas móviles para smartphones, la popularidad ha ido creciendo en torno a estos dos sistemas operativos como base para el desarrollo de aplicaciones móviles. Apple lanzó iOS como plataforma para sus propios dispositivos (iPhone, iPod Touch). En contraste, Android fue liberado como código abierto por la Open Handset Alliance.

Actualmente, Android cuenta con el apoyo de varios fabricantes como Samsung, Sony Ericsson, HTC, Toshiba y LG entre otros.

Este proyecto focaliza el entorno de desarrollo basado en el sistema operativo Android.

1.7.1 Android

Es un conjunto de herramientas de software de código abierto para teléfonos móviles, cuyo kernel está basado en los sistemas Linux. Creado en un principio por Android Inc, más tarde comprado por Google y actualmente por Open Handset Alliance, siendo Google quien ha publicado el mayor porcentaje del código fuente bajo la licencia de Software Apache (software libre y código abierto). Esta característica, favorece la creación de aplicaciones libres o privadas sin la necesidad de que sean aprobadas por Google, pudiendo ser descargadas a través del Play Store antiguamente conocido como Android Market.

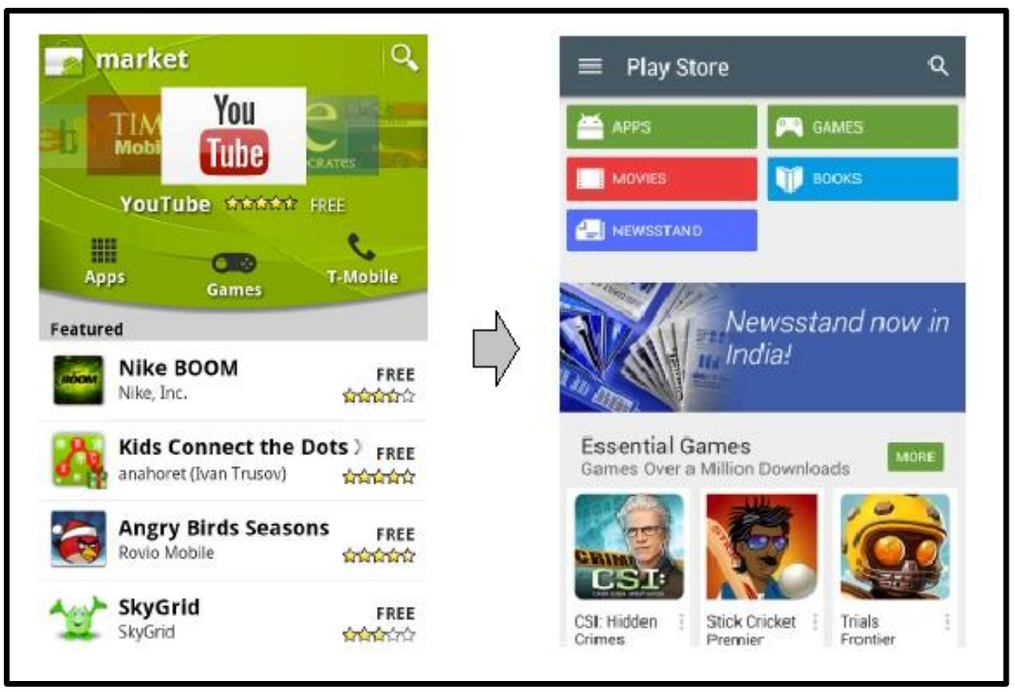


Figura 1-11: Android Market y evolución a Play Store.

Características principales:

- Plataforma abierta.
  - Adaptable a cualquier hardware.
  - Portabilidad.
  - Gran cantidad de servicios.
  - Aceptable nivel de seguridad.
  - Optimizado para baja potencia y poca memoria.
  - Alta calidad de gráficos y sonido.

1.7.1.1 Arquitectura Android

Plataforma formada por 4 capas, que incluye un sistema operativo, un middleware y aplicaciones básicas para el usuario, siendo su principal característica el que cada capa esté basada en software libre, su desarrollo está relacionado con la utilización de la máquina virtual Dalvik (Android Runtime), que permite el uso de Java como lenguaje de programación.

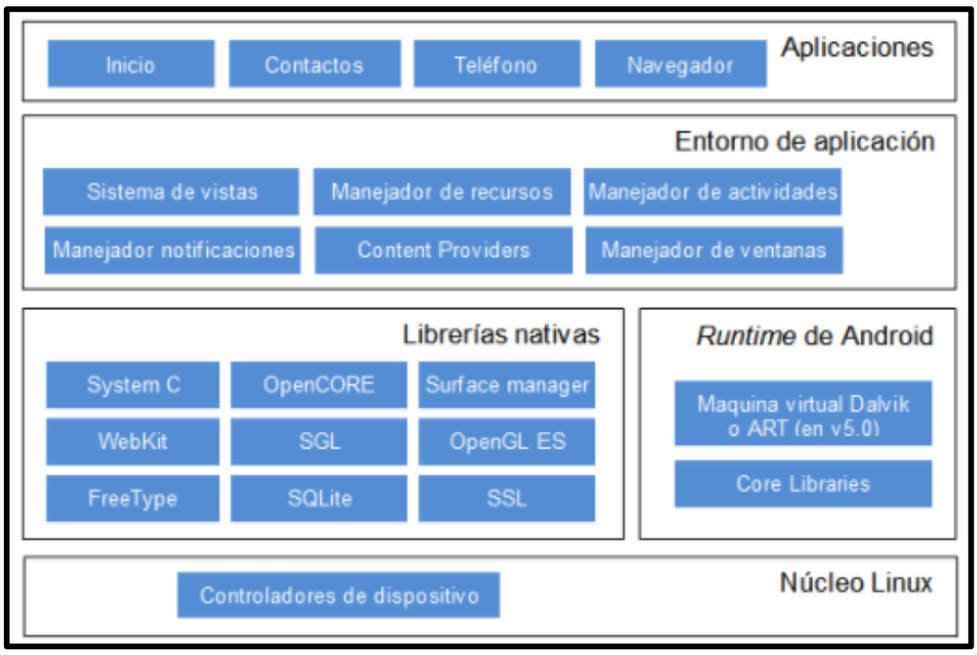


Figura 1-12: Arquitectura Android.

En la figura 1-12, se muestran las capas que constituyen la arquitectura de Android, donde:

- Núcleo Linux: Es la capa que proporciona servicios de seguridad, maneja los protocolos y el soporte de drivers para dispositivos. Basado en el sistema operativo Linux versión 2.6.

- Runtime de Android: Capa basada en el concepto de máquina virtual (máquina virtual dalvik), siendo su principal objetivo optimizar los procesos para ahorrar memoria.

- Librerías Nativas: Capa que incluye un conjunto de librerías en C/C++ compiladas con el código nativo del procesador.

- Entorno de Aplicación: Capa que proporciona una plataforma de desarrollo libre para aplicaciones, diseñada principalmente para simplificar la reutilización de componentes, de tal forma que otras aplicaciones puedan hacer uso de ellas.

- Aplicaciones: Capa que está formada por el conjunto de aplicaciones instaladas en el equipo Android.

1.7.1.2 Software MIT App Inventor 2

MIT App Inventor 2 es un entorno de desarrollo de aplicaciones para dispositivos Android. Permite diseñar, crear y editar aplicaciones directamente en el navegador web (nube), a través de su lenguaje de programación en bloques.

Posee dos herramientas de trabajo para el desarrollo de las aplicaciones, el App Inventor Designer y el App Inventor Blocks Editor.

- Inventor Designer: Es donde se crea la interfaz de usuario, mediante la elección de los elementos y componentes de la aplicación que interactuarán con el usuario.

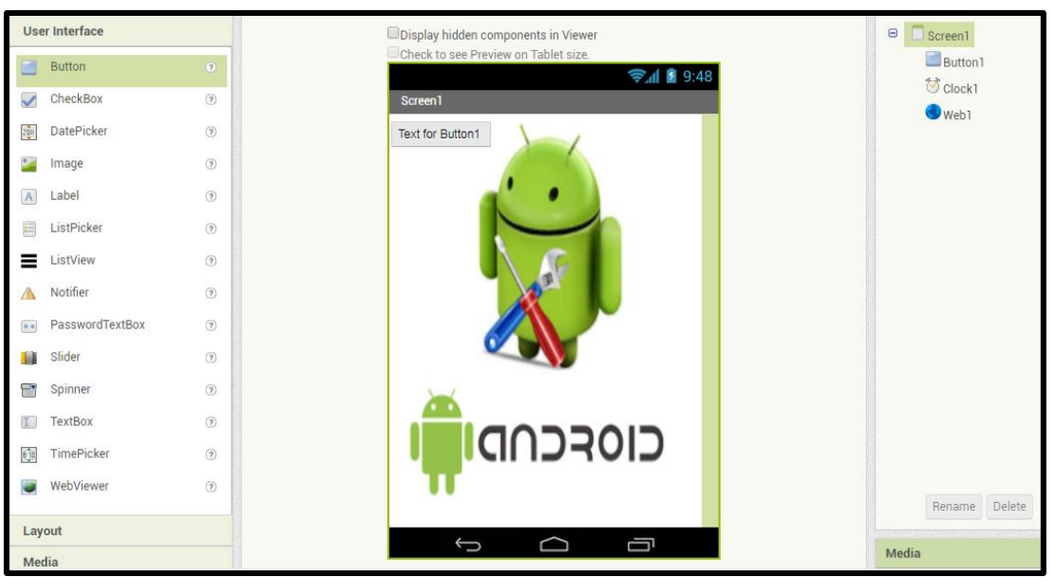


Figura 1-13: Inventor Designer.

- Blocks Editor: Es donde se definen las funciones y el comportamiento de los componentes utilizados en el diseño (programación).

Figura 1-14: Blocks Editor.

CAPITULO 2:

CONSIDERACIONES Y CRITERIOS DEL PROYECTO

2. CONSIDERACIONES Y CRITERIOS DEL PROYECTO

2.1 DEL PROYECTO EN GENERAL

El proyecto en general estará basado en el estándar Ethernet y el conjunto de protocolos TCP/IP, este estándar permitirá la interconexión y comunicación entre la Planta De Lorenzo DL 2314 ubicada en el laboratorio de procesos de la USM y los distintos dispositivos que conformen la red, sin embargo, en una primera instancia, es necesario tener en cuenta algunas consideraciones propias de los equipos y de la misma planta, como por ejemplo, las limitaciones de corriente y voltaje.

2.1.1 Levantamiento de entradas y salidas Planta De Lorenzo DL 2314

La planta De Lorenzo DL 2314 originalmente está conformada, por la Planta propiamente tal y por su Estación de Operación. Uno de los módulos que conforma dicha estación es el panel de control, el cual está constituido por 3 interfaces, siendo el primero la Interfaz de Entrada (Sensores), segundo la Interfaz de Control (Controlador) y por último la Interfaz de Salida (Actuadores), las que en conjunto permiten realizar control ON/OFF O PID de un proceso de manera local.

Sin embargo, este proyecto busca monitorear de manera remota, las variables de entradas (sensores) y realizar acciones de comando sobre los actuadores (salida), sustituyendo la interfaz de control por la plataforma Arduino y Ethernet shield.

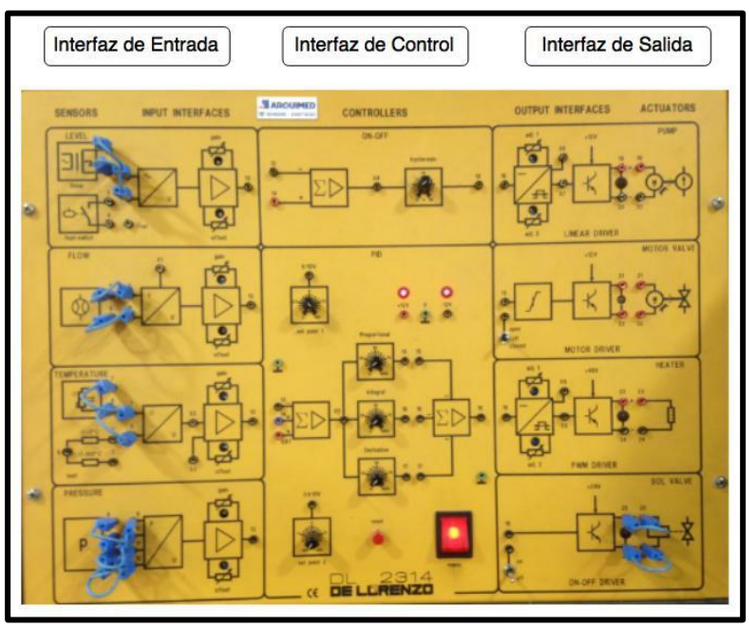


Figura 2-1: Panel de Control y sus Interfaces.

2.1.1.1 Interfaz de entrada

Las variables que maneja esta interfaz son del tipo Análogas, la cuales son:

- Nivel (Level)/Sensor LVDT Serie SX12.
- Flujo (Flow)/ Sensor IR-OPFLOW TYPE 2
- Temperatura (Temperature)/ Termistor PTC
- Presión (Pressure)/Sensor Transductor de Presión HCX.

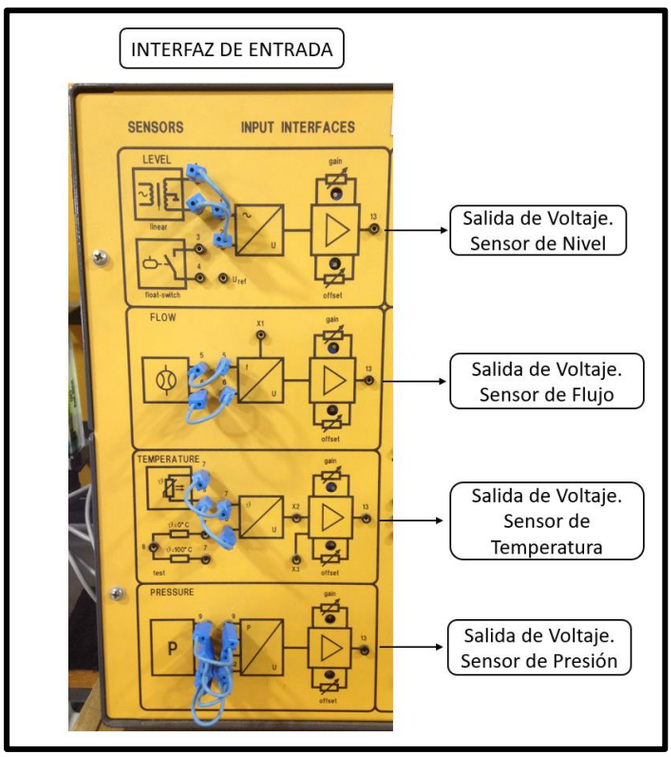


Figura 2-2: Interfaz de Entrada y sus Variables de Proceso.

Los valores de voltaje que entregan los sensores de cada variable, oscilan de 0[v] a 10[v], es por esta razón que se debe realizar un divisor de voltaje y conectarlo a la salida de cada uno, para así obtener tensiones que varíen entre 0[v] y 5[v], ya que Arduino posee esta limitación en su etapa de entrada.

2.1.1.2 Interfaz de salida

Esta interface maneja cuatro actuadores para realizar acciones de operación sobre las variables medidas en la etapa de entrada, las cuales son:

- Bomba de Recirculación(Pump), para el proceso de Nivel.
- Válvula Motorizada (Motor Valve), para el proceso del Flujo.
- Calentador (Heater), para el proceso de Temperatura.
- Válvula Solenoide (Solenoide Valve), para el proceso de presión.

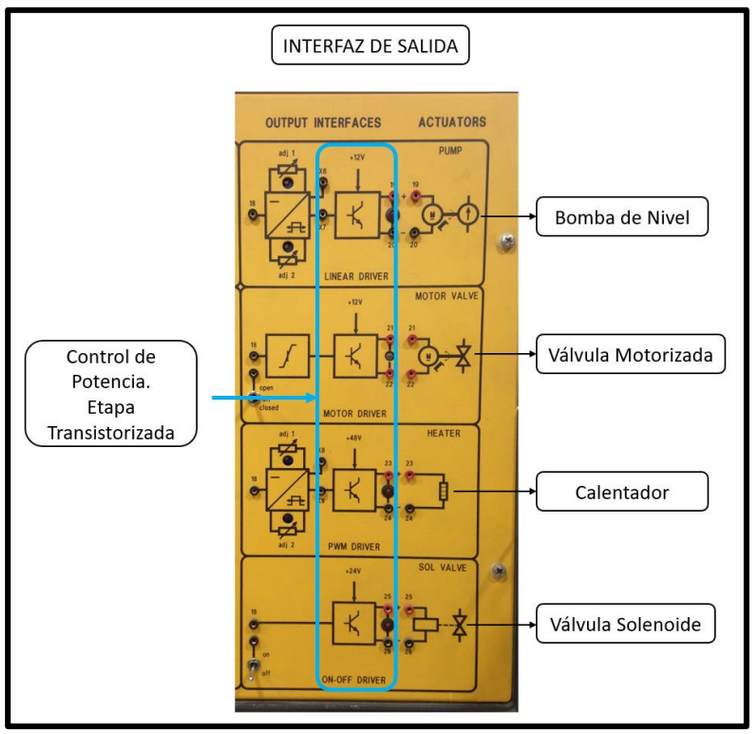


Figura 2-3: Interfaz de Salida y sus Actuadores.

El control de potencia de estos, originalmente es realizado por una etapa transistorizada que pertenece a la propia Planta (ver figura 2-3), sin embargo y por efectos de conveniencia para el proyecto, será sustituida por otra que se ajuste a los parámetros proporcionados por la placa Arduino.

A continuación, en la tabla 2-1, se muestran los valores de corriente y voltaje que consume cada actuador.

Tabla 2-1: Voltajes y Corrientes de Actuadores.

Actuadores	Señal de Control	Voltaje de Operación	Consumo Corriente del Drive	Corriente de Colector (E. Transistorizada)
Bomba	ON = 10.04 [v]	10.93 [v]	0.81 [mA]	1.62 [A]
	OFF = 0 [v]	0 [v]	0.31 [mA]	
Válvula Motorizada	OPEN	10 [v]	0.49 [mA]	0.41 [A]
	OFF	0 [v]	0 [mA]	
	CLOSE	-10 [v]	-0.49 [mA]	
Calentador	ON = 10.04 [v]	45.8 [v]	0.78 [mA]	3.86 [A]
	OFF = 0 [v]	0 [v]	0.26 [mA]	
Válvula Solenoide	OPEN	10 [v]	0.94 [mA]	0.45 [A]
	CLOSE	0 [v]	0 [mA]	

La razón por la cual se necesita cambiar la etapa de potencia de la planta, es por el hecho de que Arduino entrega un voltaje de 5[v] en sus pines de salida, y como lo evidencia la tabla 2-1, cada actuador requiere de 10[v] como señal de control, motivo que justifica una amplificación de tensión.

Otro factor a considerar, es el funcionamiento de la Bomba de Nivel y el Calentador, ya que se requiere variar la velocidad y la resistencia de estos respectivamente. Para ello es necesario realizar un control de ancho de pulso (PWM) sobre el voltaje amplificado, si bien esto es posible, la mayoría de las soluciones que lo permiten (transistores) tienen una configuración que posee 2 salidas, razón que dificultaría la operación si se considera usar el Control de Potencia de la Planta De Lorenzo DL 2314, ya que ésta solo permite una entrada para el drive de cada actuador, la que es denotada con el número 18 de acuerdo a la figura 2-4.

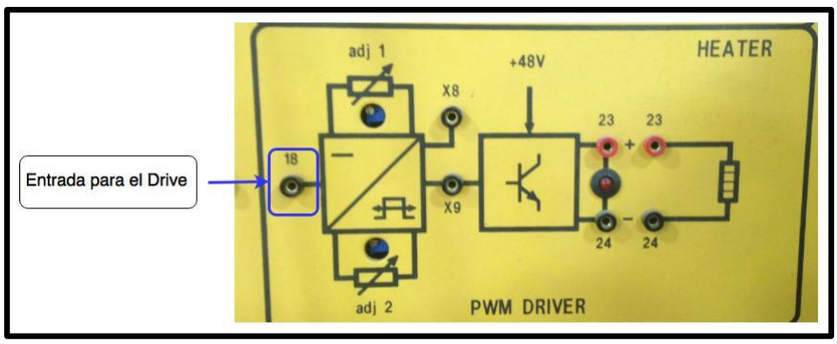


Figura 2-4: Entrada para el Drive del Actuador.

Además, se agrega un tercer factor a considerar, el cual está relacionado al funcionamiento de la válvula motorizada, ya que esta no funcionaría correctamente, porque requiere para su apertura un voltaje positivo y para su cierre un voltaje negativo. Arduino no es capaz de entregar estos valores por sí solo, y si lo fuera, los voltajes se anularían en la entrada del drive indicada.

Lo expuesto, permite determinar que la mejor opción para el proyecto, es sustituir la etapa de potencia de la planta, por otra que se ajuste a las necesidades de la aplicación.

2.2 DE LA RED DE COMUNICACIÓN

2.2.1 Protocolo

Como se ha estudiado, el protocolo más acertado para el proyecto, es el TCP/IP bajo el estándar Ethernet 10/100 Base TX. Sus principales ventajas son el modo en que transmite, la gestión de los paquetes de datos (detección y solución de errores) y su alcance, ya que este protocolo les asigna una IP a todos los equipos conectados a la red, lo que en teoría se traduce en un alcance infinito, a diferencia del bluetooth por ejemplo que posee una limitación de 100 metros entre sus dispositivos.

2.2.2 Capacidad de la Red

La red en general incluye 4 equipos, los cuales son Arduino, Ethernet Shield, un router y un dispositivo Android, pero se debe considerar que Arduino y Ethernet Shield serán solo uno, por lo tanto, en general son 3 equipos. Sin embargo, la red puede crecer a medida que se conecten más dispositivos Android y ésta solo se vería limitada por la cantidad que permita el router.

2.2.3 Cableado de la Red

El cable que se debe utilizar debe ser par trenzado UTP (sin pantalla), de categoría 5e como mínimo, su conectorización debe considerar conectores RJ45 de conexión local con cableado directo bajo el estándar TIA/EIA 568B.

2.3 DEL HARDWARE

La elección del hardware Arduino Uno Rev3. y Ethernet Shield, se ha hecho considerando factores como, costo y fácil implementación, sin embargo, es necesario señalar que la cantidad de entradas y salidas de esta plataforma limita la posibilidad de expansión para futuras aplicaciones, es por esta razón que, si se desean expandir sus funciones, se debe considerar el cambio a versiones posteriores.

2.4 DEL SOFTWARE

El software utilizado para el caso de Arduino Uno Rev3. es su propio IDE, basado en el lenguaje de programación de código abierto, y para el desarrollo del interfaz usuario, se utilizará MIT App Inventor 2.

2.5 DE LA ESTRUCTURA FÍSICA

La estructura física de la aplicación contempla una base de acrílico, en donde se ubicará la Placa Universal que contiene al conjunto Arduino/Ethernet Shield, los divisores de voltaje y la nueva etapa de potencia.

Para efectos de ampliación en el proyecto, se entregará el PCB completo de la etapa de potencia, el cual se ha realizado con el software EasyEDA, teniendo como objetivo el que la creación de ésta a futuro sea más sólida y profesional.

CAPITULO 3:

DESCRIPCIÓN DE LA SOLUCIÓN A IMPLEMENTAR

3. DESCRIPCIÓN DE LA SOLUCIÓN A IMPLEMENTAR

3.1 DESCRIPCIÓN GENERAL DEL PROYECTO

Como se ha mencionado en el objetivo, se requiere diseñar e implementar una aplicación para dispositivos inteligentes Android, que permita la supervisión y el control remoto de todas las variables asociadas a los lazos de control de la Planta De Lorenzo DL 2314. Para lograr esta supervisión, se dispone del conjunto Arduino Uno Rev3. Ethernet Shield, router y/o celulares/tablets Android.

En primer lugar, el conjunto Arduino Uno y Ethernet Shield, se deben conectar al router para que se les asigne una dirección IP y pueda pertenecer a una Red Ethernet.

En segundo lugar, se debe programar al Arduino Uno a través de su software IDE para que sea capaz de establecer comunicación con la red Ethernet, supervise las variables de entrada de la Planta De Lorenzo DL 2314, y a su vez para que realice acciones de operación sobre ellas.

En tercer lugar, se debe crear la interfaz de usuario a través del software MIT App Inventor 2, que permita la supervisión y el control de las variables de manera remota en nuestros dispositivos Android.

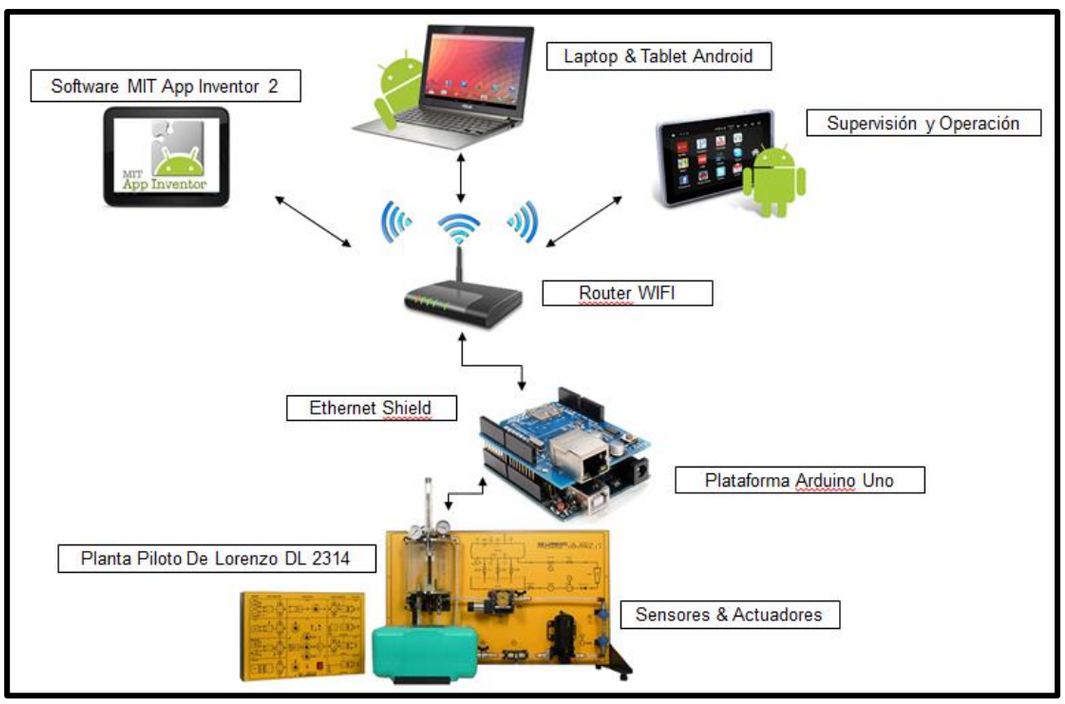


Figura 3-1: Diagrama explicativo del Proyecto.

El principio de funcionamiento básicamente consiste en que el cliente móvil, envía una petición al servidor que es Arduino, y éste cuando la recibe se encarga de ejecutarla.

Para que esto sea posible se deben considerar los siguientes conceptos:

- Diseño del protocolo de transferencia.
- Implementación del servidor en Arduino Uno Rev3.
- Diseño de la aplicación móvil en Android (Interfaz Usuario).
- Implementación del Servidor en Android.

3.1.1 Diseño del Protocolo de Transferencia

Sin duda uno de los pasos más importantes es lograr la transferencia de archivos/datos entre el servidor y el móvil, para ello es fundamental que el programa desarrollado en el IDE se entienda con el programa creado en MIT App Inventor 2, en ese aspecto el protocolo TCP se encarga y garantiza que los datos se envíen correctamente.

En cuanto al modelo de comunicación, TCP es un protocolo peer-to-peer orientado a la conexión donde no existe la relación Maestro/Esclavo, sin embargo, se usa la comunicación Cliente/Servidor, es decir, el cliente (Android) es el solicitante de un servicio (petición) y el servidor (Arduino), es quien ofrece y responde a ese servicio (respuesta a la petición). Se destaca el hecho de que el servidor puede recibir múltiples peticiones, es decir varios dispositivos Android.

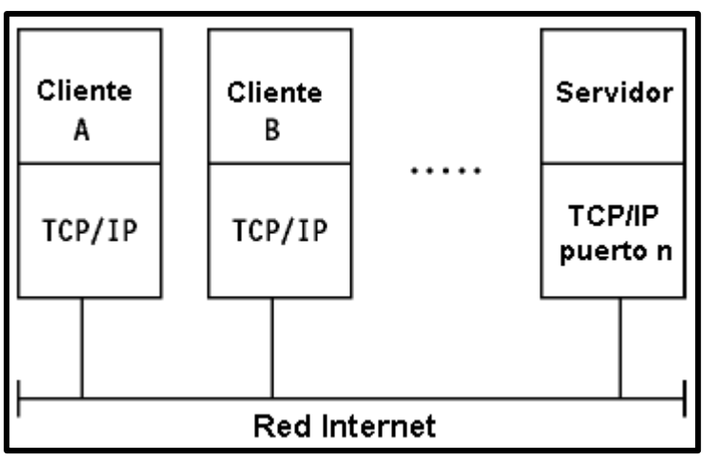


Figura 3-2: Modelo Cliente/Servidor.

Para que la comunicación de datos sea correcta, el protocolo de transferencia debe seguir ciertos pasos, en primer lugar, Arduino y el móvil Android deben establecer una conexión, en segundo lugar, se leen las variables de la Planta y de forma simultánea se puede solicitar realizar acciones sobre ellas, por último, el

servidor (Arduino) responde a estas peticiones y se visualizan en el móvil, en la página servidor y en la propia Planta.

3.1.2 Implementación del Servidor en Arduino Uno Rev3

3.1.2.1 Primeros pasos en Arduino

Para poder utilizar el entorno de desarrollo de Arduino (IDE), es necesario realizar algunas configuraciones entre la placa Arduino y el Computador. Para ello se debe abrir la pestaña "Herramientas", luego click en "Puerto" y dentro de esta se selecciona el puerto al cual está conectado Arduino Uno. En Windows, si desconoce el puerto, puede buscarlo a través del administrador de dispositivos (Puertos COM & LTP/USB Serial Port).

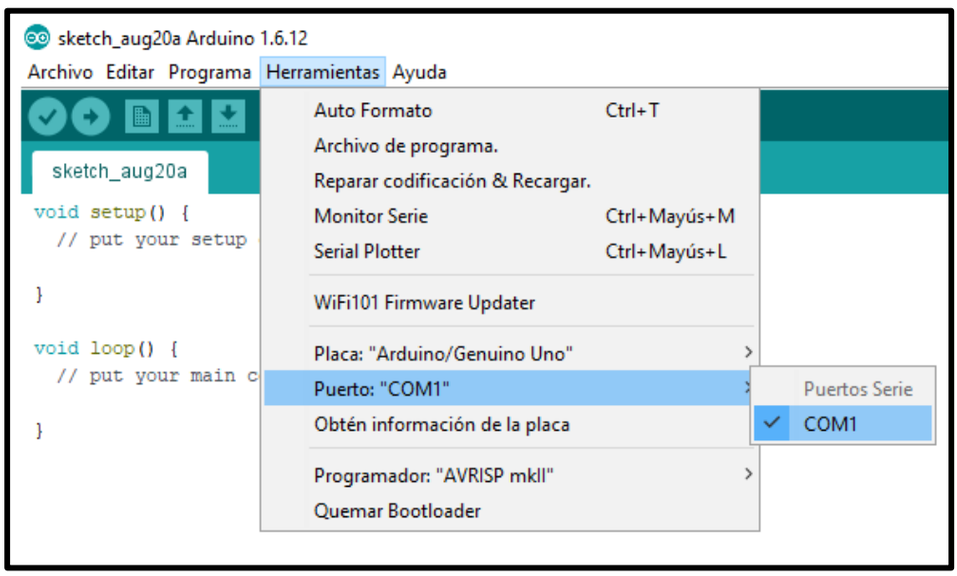


Figura 3-3: Selección Puerto Arduino.

La estructura de programación de Arduino divide el programa en dos partes fundamentales, las cuales son:

- SETUP: Constituye la preparación del programa. Se incluye la declaración de variables y la configuración del PinMode (entrada o salida), siendo la primera función que se ejecuta y por única vez.

- LOOP: Constituye la ejecución del programa. Incluye el código a ser ejecutado de manera continua y la lectura de entradas y salidas de la placa Arduino.

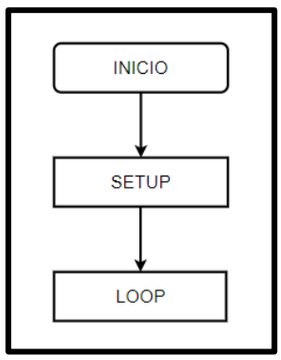


Figura 3-4: Diagrama de Flujo General del Programa.

Las Librerías son muy importantes en el programa, ya que proveen funcionalidades extra a la placa Arduino, en este caso las librerías a utilizar son la SPI.h y Ethernet.h, la primera es necesaria para la comunicación entre Arduino y Ethernet Shield, la segunda para dotarlo de comunicación Ethernet.

3.1.2.2 Declaración del servidor en Arduino (Ethernet)

Para que Arduino actúe como servidor, se debe declarar y configurar con una dirección MAC aleatoria, una IP (Puerta de enlace) y un Puerto de comunicación.

```
byte mac[] = {0xB7, 0x17, 0x07, 0xFF, 0xCB, 0x43};
IPAddress ip (192,168,1,60);

EthernetServer server(80);
```

Figura 3-5: Declaración del Servidor.

3.1.2.3 Declaración de Variables Enteras

Antes de definir el PinMode, es necesario declarar las variables como enteros y a su vez declarar los pines que utilizará en Arduino.

```
// Variables Enteras

int Pump = 5;
int Valve_Close = 2;
int Valve_Open = 3;
int Heater = 6;
int Solenoide = 9;
```

Figura 3-6: Declaración de Variables.

3.1.2.4 Setup

Como se ha dicho, en el Setup se declaran las variables y el PinMode, definiendo si éstas son entradas o salidas según corresponda, además se inicia la comunicación Ethernet declarada anteriormente (MAC, IP, PUERTO).

La figura 3-7, indica la estructura del Setup y las variables declaradas.

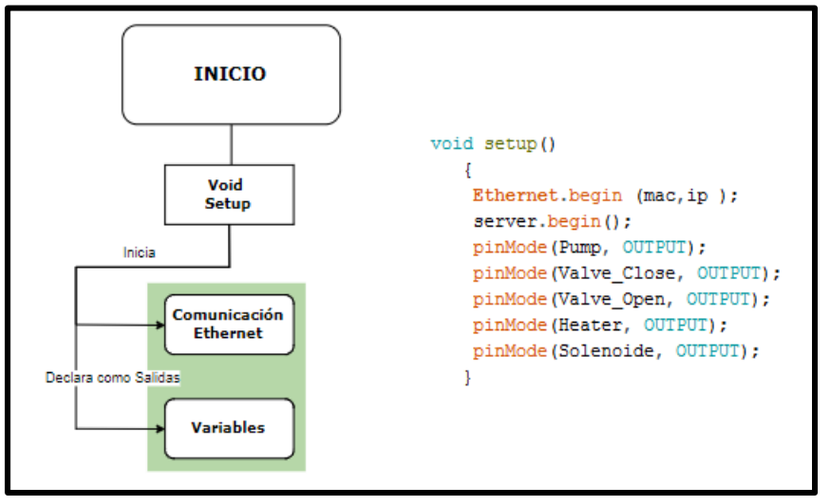


Figura 3-7: Estructura Setup.

3.1.2.5 Loop

Esta etapa es donde se ejecuta la función principal del programa es conocido como bucle, ya que se ejecuta repetitivamente.

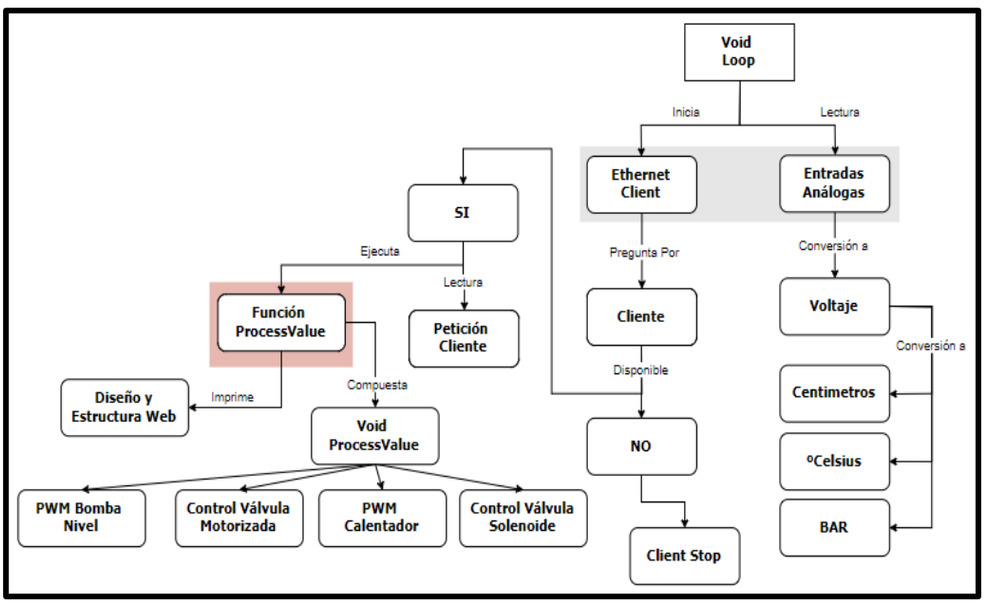


Figura 3-8: Estructura Loop.

La figura 3-8 muestra la estructura del Loop. Aquí se realiza la lectura de entradas análogas de la Planta De Lorenzo DL 2314 (Arduino lee valores de 0 a 1023), y el programa se encarga en primer lugar de convertirlas a una lectura de voltaje y posteriormente en las Unidades de Medición requeridas (Centímetros, Grados Celsius, BAR).

```
// Lectura de Pines Análogos

const int SensorNivel = analogRead(A0);
const int SensorFlujo = analogRead(A1);
const int SensorTemperatura = analogRead(A2);
const int SensorPresion = analogRead(A3);

// Realiza el Escalar de la Lectura Análoga a Voltaje

float VoltajeNivel = SensorNivel*(5.0 / 1023.0);
float VoltajeFlujo = SensorFlujo*(5.0 / 1023.0);
float VoltajeTemperatura = SensorTemperatura*(5.0 / 1023.0);
float VoltajePresion = SensorPresion*(5.0 / 1023.0);

// Realiza el Escalar del Voltaje anterior a la Variable de Proceso Correspondiente

float Nivel = VoltajeNivel * (10.0 / 5.0);
float Temperatura = VoltajeTemperatura * (40.00 / 5.0);
float Presion = VoltajePresion * (2.0 / 5.0);
```

Figura 3-9: Lectura y Conversión Entradas Análogas.

Al mismo tiempo que se leen las entradas análogas, se inicia la comunicación Ethernet, en donde el servidor pregunta por un cliente. Cuando el cliente está disponible, se ejecutan las acciones de comando para los actuadores según sea la petición (accionar/apagar Bomba, abrir/cerrar Válvulas, realizar control PWM) y de forma paralela se imprime el diseño del servidor WEB donde se muestran los valores de la acción realizada.

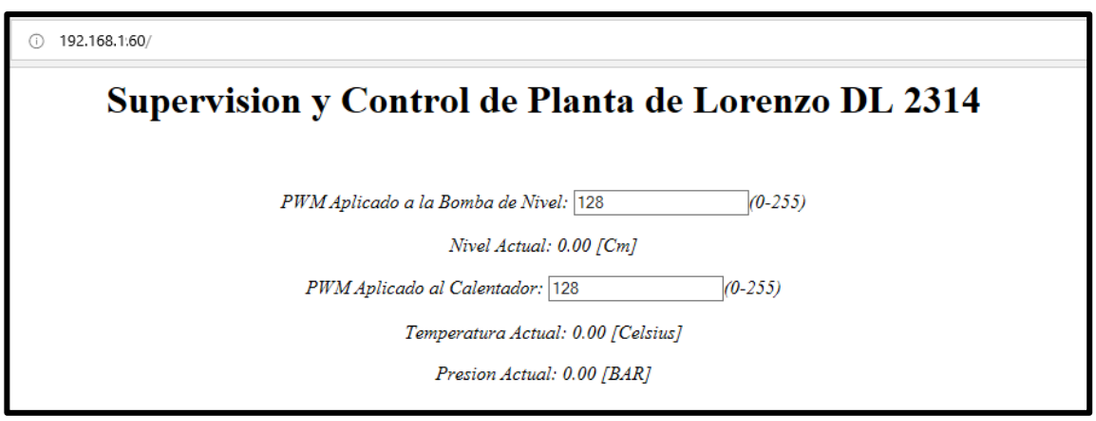


Figura 3-10: Diseño WEB.

Cada acción de comando requiere su propio código en IDE, sin embargo, existe similitud entre ellos debido al principio de funcionamiento. La diferencia recae en dos aspectos principalmente, la primera es el tipo de salida escogida, ya sea digital (High, Low) o análoga (0, 255), y la segunda es la nomenclatura usada en la función "indexOf", función que se utiliza para acceder a la localización del DATO asociado a la petición, comúnmente conocido como "DATA".

Para el proceso de control de nivel, se requiere accionar, apagar y variar la velocidad de la bomba, en este caso el tipo de salida será análoga, por lo tanto, su estado en alto equivale a 255, su estado bajo equivale a 0 y los valores intermedios reflejan la variación de velocidad. Estas solicitudes se guardan (DATA) bajo la nomenclatura **?n=**, siendo este el código utilizado en la programación y que corresponde al texto que se encuentra después de la IP del servidor, particularmente la dirección sería 192.168.1.60/?n=255 para estado alto y 192.168.1.60/?n=0 para estado bajo.

```
start = HTTP_req.indexOf("?n=");

if (start != -1) {
n = start + 3;
while (a != ' ') {
a = HTTP_req.charAt(n);
numasstr += a;
n++;
}
Pump_Out = numasstr.toInt();
 if (Pump_Out == 0) {
 digitalWrite(Pump, LOW);
 }
  else{
      if (Pump_Out == 255) {
      digitalWrite(Pump, HIGH);
    }
        else{
        analogWrite(Pump, Pump_Out);
```

Figura 3-11: Código Control de Nivel.

El código para el proceso de control de presión es similar al de nivel, sin embargo, como se verá en la figura 3-12, la salida es del tipo digital (High = Alto, Low = Bajo), debido a que solo se necesita abrir y cerrar una válvula solenoide. La nomenclatura usada en la función "indexOf" es **?p=**, por lo tanto la dirección sería 192.168.1.60/?p=10 para estado alto y 192.168.1.160/?p=0 para estado bajo.

```
start = HTTP_req.indexOf("?p=");


if (start != -1) {
n = start + 3;
while (a != ' ') {
a = HTTP_req.charAt(n);
numasstr += a;
n++;
}
Solenoide_Out = numasstr.toInt();
  if (Solenoide_Out == 0) {
  digitalWrite(Solenoide, LOW);
  }
    else{
       if (Solenoide_Out == 10) {
       digitalWrite(Solenoide, HIGH);
```

Figura 3-12: Código Control de Presión.

3.1.2.6 Estructura Final de la Programación

La estructura final, es el resultado de la unión de todo lo anterior descrito, las que en conjunto permiten el correcto funcionamiento del Programa.

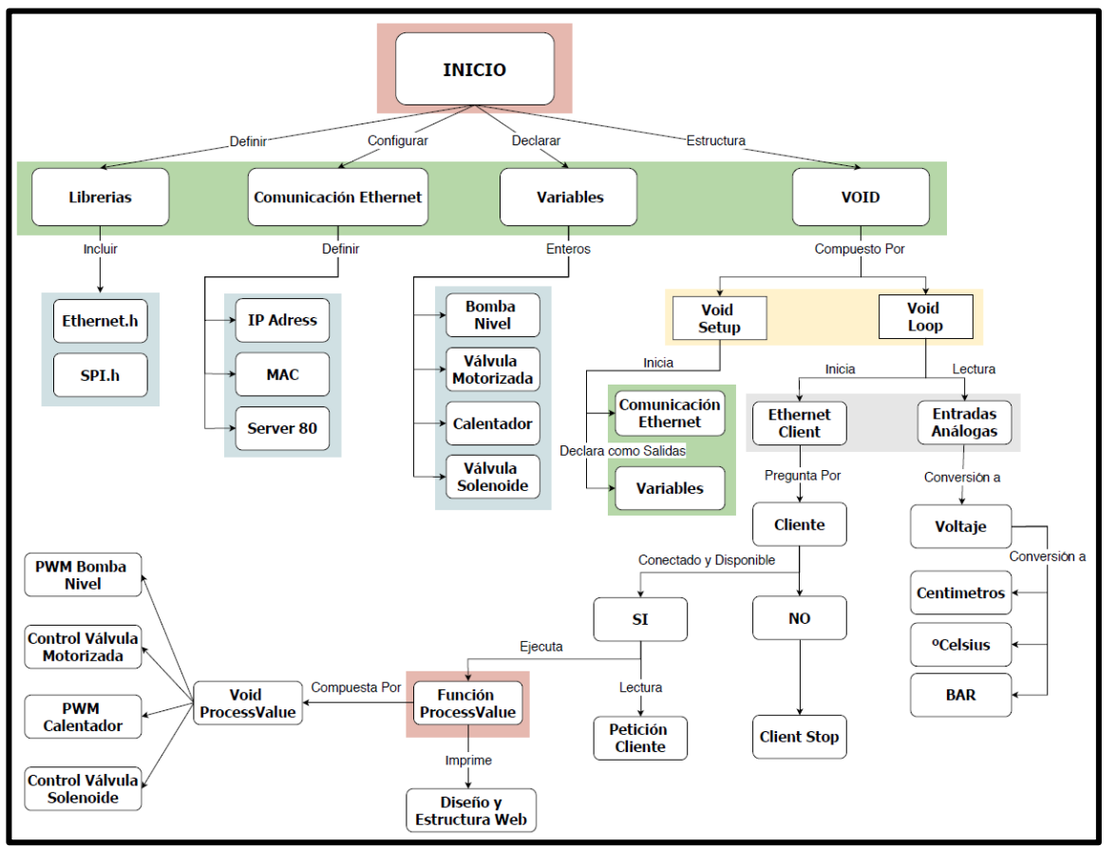


Figura 3-13: Estructura Final Programación.

3.1.3 Diseño de Aplicación en Android.

El diseño de la aplicación será realizado por medio del software ya mencionado MIT App Inventor 2, el cual, a través del Inventor Designer proporciona todos los elementos y componentes que cumplen con los requisitos requeridos.

Para comenzar a desarrollar la aplicación, se debe acceder a la página principal http://ai2.appinventor.mit.edu/, en donde será necesario ingresar con una cuenta de Google, posteriormente seleccionar Nuevo Proyecto y finalmente crear el software de control.

La aplicación creada tiene por nombre "Proyecto de Titulo" y está compuesta por seis pantallas (screens), las cuales son, Portada, Índice de Variables, Supervisión de Nivel, Supervisión de Flujo, Supervisión de Temperatura y Supervisión de Presión.

3.1.3.1 Screen de Portada

Figura 3-14: Screen de Portada.

3.1.3.2 Screen Índice de Variables

Este screen ha sido diseñado utilizando de imagen de fondo la Planta De Lorenzo DL 2314, en donde se muestran los elementos que la componen.

- Nivel: Estanque de Nivel, Bomba de Nivel.
- Flujo: Válvula Motorizada, Flujómetro.
- Temperatura: Calentador, Termómetro.
- Presión: Válvula Solenoide, Barómetro.

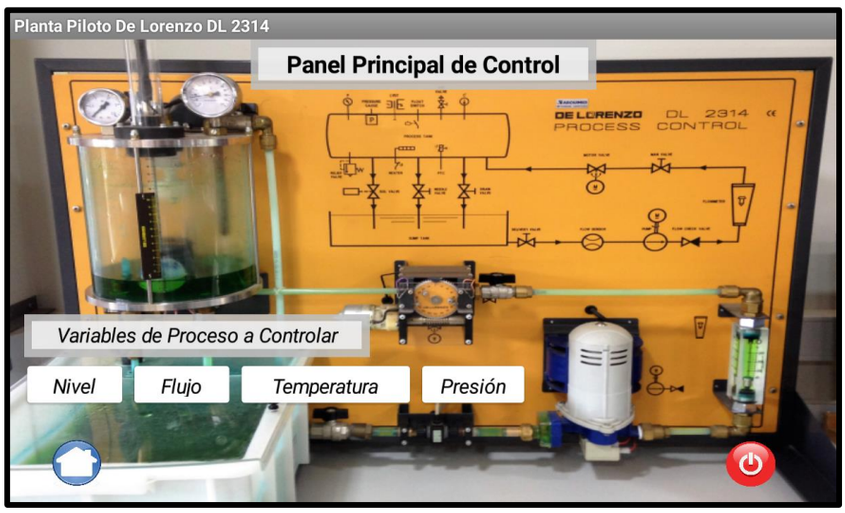


Figura 3-15: Screen Índice de Variables.

3.1.3.3 Screen Supervisión de Nivel

Screen compuesto por los botones que accionan y apagan la Bomba de Nivel, también posee una barra slider la que permitirá realizar control PWM sobre el motor de la Bomba, es decir, variar la velocidad de éste. Además, cuenta con un visor que muestra el nivel en Centímetros con refrescos de 2 segundos.

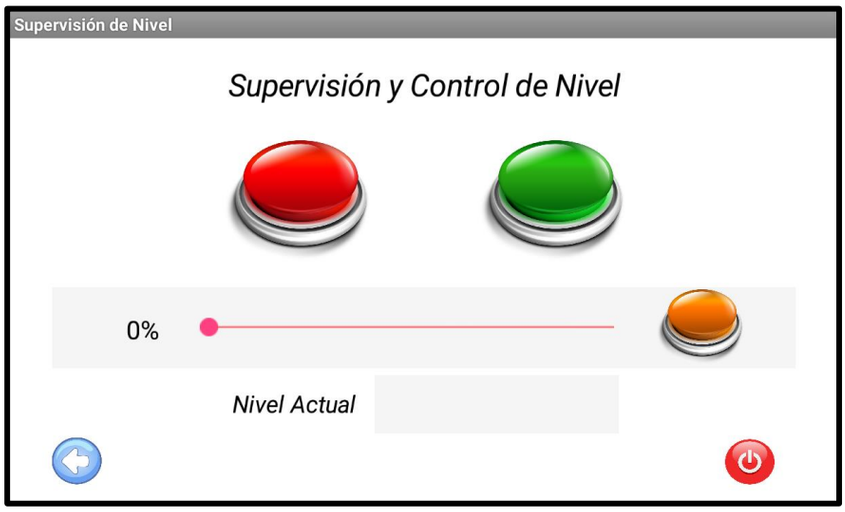


Figura 3-16: Screen Supervisión de Nivel.

3.1.3.4 Screen Supervisión de Flujo

Screen compuesto por tres botones, los cuales permiten, accionar la válvula motorizada (Apertura Total 0º), cerrar la válvula motorizada (Cierre de 40º) y por último apagar la válvula.

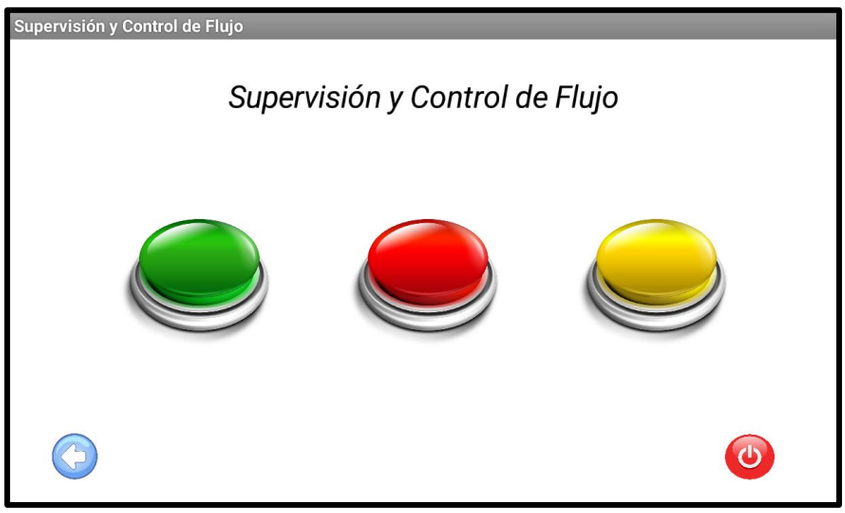


Figura 3-17: Screen Supervisión de Flujo.

3.1.3.5 Screen Supervisión de Temperatura

La funcionalidad de los botones en este diseño, son iguales a los del screen de supervisión de nivel.

El visor muestra la temperatura en Grados Celsius del agua, con refrescos de 2 segundos.

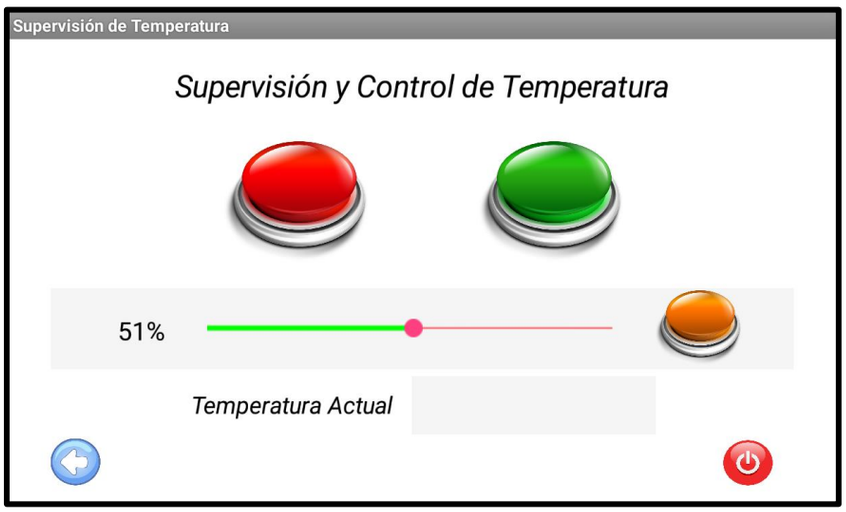


Figura 3-18: Screen Supervisión de Temperatura.

3.1.3.6 Screen Supervisión de Presión

Este screen contiene los botones para apertura y cierre de la válvula solenoide, además de un visor que muestra cada 2 segundos la presión en BAR del estanque de nivel.

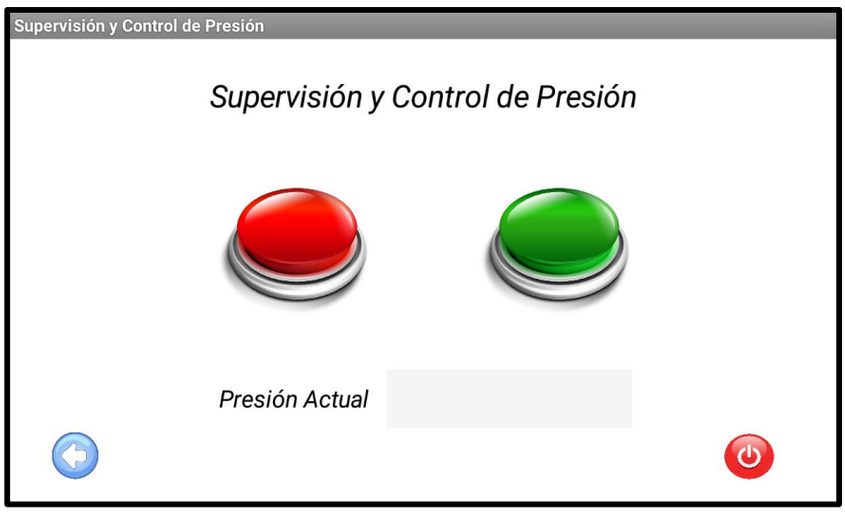


Figura 3-19: Screen Supervisión de Presión.

3.1.4 Implementación del Servidor en Android.

MIT App Inventor 2 no solo permite crear la interfaz de usuario (diseño), sino que además permite crear el código para lograr la conectividad entre Arduino y Android.

El lenguaje de programación en bloques, proporciona un entorno de desarrollo más amigable a la hora de crear el código, sin embargo, el uso de tutoriales será necesario para su correcta implementación.

La herramienta de trabajo necesaria se denomina Blocks Editor, y es quien define las funciones y el comportamiento de todos los componentes que se utilizaron en el diseño. Ejemplo, cambios de pantalla/screen, acciones sobre un botón, etc.

Cada componente utilizado en la etapa de diseño, proporciona funciones específicas para cada uno de ellos, éstas vienen definidas por colores, donde cada color significa una función o proceso en particular, ya sean, peticiones, llamados, lógica de control, definir variables, etc.

Las pantallas/screens implementadas requieren su propio código para poder funcionar, ya que se desean controlar diferentes procesos, a pesar de ello la diferencia entre los códigos es mínima, debido a la similitud de las acciones que se requieren. Es por esta razón que se detallará la programación realizada para la Supervisión de Nivel y para la Supervisión de Presión.

3.1.4.1 Código de Screen Supervisión y Control de Nivel

Para comenzar, lo primero que se debe hacer es declarar como Variables Globales, el Nivel y la dirección IP de acceso a la Página Servidor, la que corresponde a la misma IP utilizada en la programación del IDE.

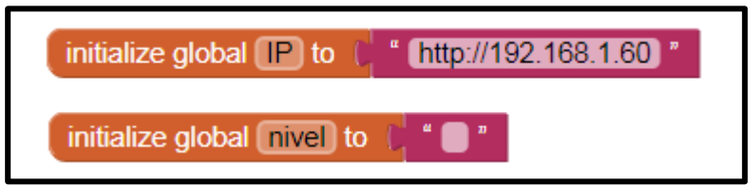


Figura 3-20: Variables Globales, IP, Nivel.

Una vez realizado, se procede a programar las acciones para cada Botón utilizado en la etapa de diseño.

- Apagar Bomba (OFF): Corresponde al "OFF" en el código.

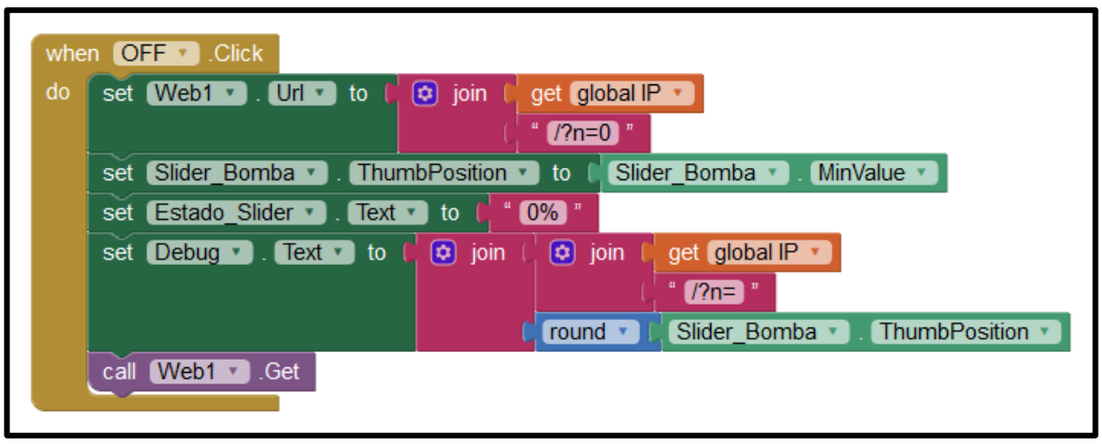


Figura 3-21: Código Apagar Bomba.

Este código debe estar en sincronía con el código realizado en el IDE de Arduino, para ello es necesario recordar que la nomenclatura utilizada para guardar los datos de las peticiones (DATA) en este caso es **?n=0** (Low=Apagar).

Se puede visualizar en la programación de la imagen 3-21, que al hacer click sobre el "OFF" se realizarán los procedimientos que permitirán apagar la bomba de nivel. Primero se establece la dirección IP de la página servidor (192.168.1.60) y su llamado (Call/Get), esto aplica para los botones que tienen relación con el control de la Bomba (OFF, ON, PWM). En el apartado Slider_Bomba lo que se hace es posicionar la barra slider al valor mínimo, que corresponde al cero (apagar).

- Accionar Bomba (ON): Corresponde al "ON" en el código.

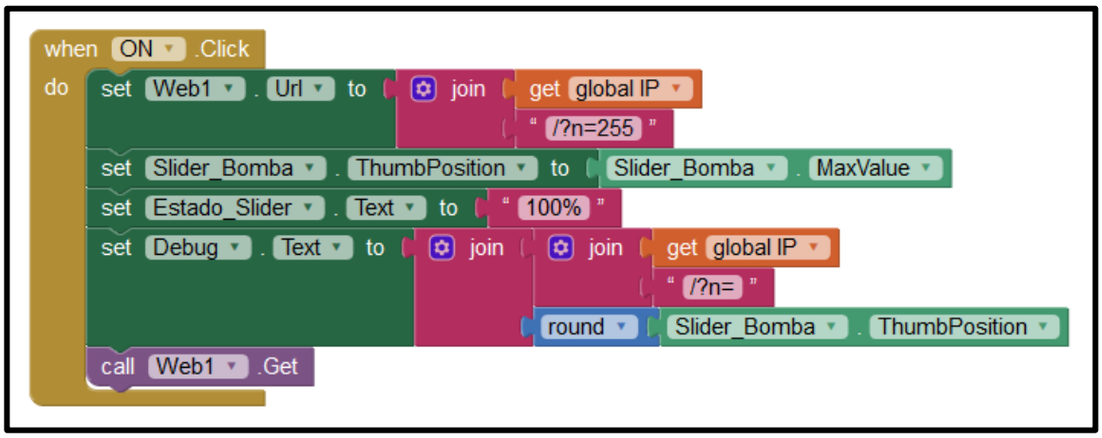


Figura 3-22: Código Accionar Bomba.

Programación similar al botón de apagado, la diferencia recae en la extensión utilizada para guardar el dato de la petición (DATA), la cual, para este caso es **?n=255** (High=Accionar).

La barra slider en esta oportunidad se posiciona en el valor máximo permitido, que corresponde al estado alto.

- Velocidad Bomba (PWM): Corresponde al "PWM" en el código.

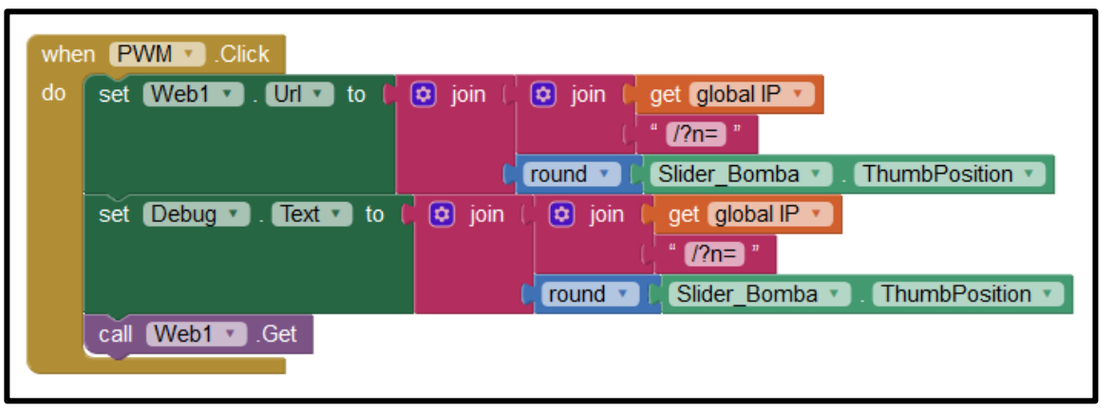


Figura 3-23: Código Control PWM.

La nomenclatura utilizada en esta oportunidad solo es aplica **?n=**, debido a que al mover la barra, ésta adquiere valores intermedios entre 0 y 255.

Básicamente lo que realiza es activar el valor intermedio adquirido producto del desplazamiento de la barra slider.

- Slider: Corresponde a la barra que se desplaza para realizar control PWM sobre el motor de la bomba.

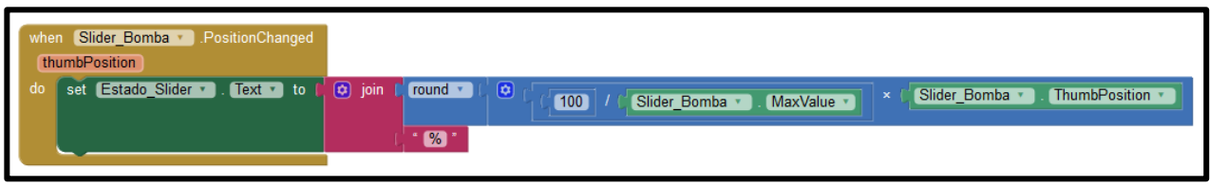


Figura 3-24: Código Barra Slider.

En la figura 3-24, se observa el código para el slider, éste permite visualizar cual es el porcentaje (%) de acuerdo a la posición/desplazamiento de la barra. Esto se logra a través del escalar propuesto en la imagen.

Escalar Propuesto:

$$Slider = \left[\left(\frac{100}{Máx.Slider.Bomba}\right) \times Position.Slider.Bomba\right]$$

- Visor de Nivel: Corresponde a la casilla que permite mostrar el valor actual del Nivel en centímetros.

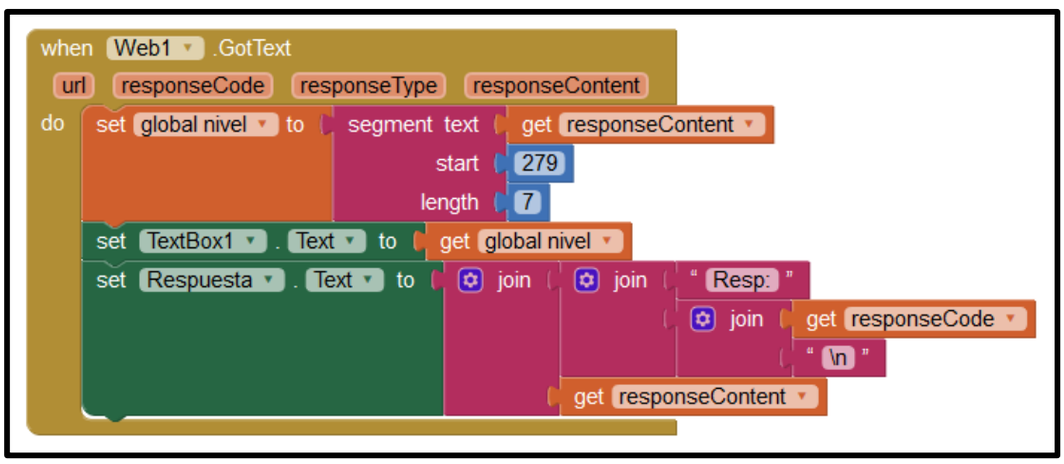


Figura 3-25: Código Visor de Nivel.

El código para poder visualizar el Nivel actual del estanque, es el propuesto en la figura 3-25.

Para entender cómo se logra, es necesario mencionar que el Servidor Web creado a través del IDE, tiene una estructura que utiliza el lenguaje de programación HTML, por lo tanto, cada carácter usado se encuentra en una posición específica dentro del total de caracteres empleados. Dicho esto, lo que hace el código creado en

MIT App Inventor 2, es posicionarse en la ubicación exacta donde aparece el valor del nivel y además indica la cantidad de caracteres que se debe mostrar (largo de la palabra).

Los refrescos de 2 segundos que permiten mantener actualizado el valor, se logran mediante el uso de un reloj que al cabo del tiempo mencionado realiza un llamado al Servidor Web. Figura 3-26.

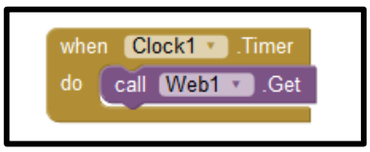


Figura 3-26: Código de Refresco de Datos.

3.1.4.2 Código de Screen Supervisión y Control de Presión

El principal cambio en contraste con el código anterior recae en la nomenclatura usada para guardar los datos de la solicitud (DATA), esto se debe a que el tipo de control que se realiza es digital, en consecuencia de ello se utiliza **?p=0** para estado Low (CLOSE) y **?p=10** para estado High (OPEN). Para el visor, el principio es el mismo, sólo cambia la posición del dato al que se necesita acceder.

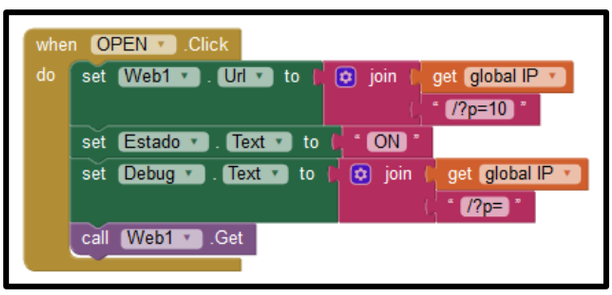


Figura 3-27: Código Abrir Válvula Solenoide.

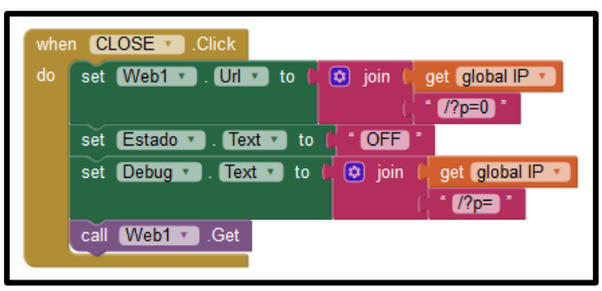


Figura 3-28: Código Cerrar Válvula Solenoide.

CAPITULO 4:

CÁLCULOS PREVIOS A LA IMPLEMENTACIÓN

4. CÁLCULOS PREVIOS A LA IMPLEMENTACIÓN

Este capítulo considera los ajustes y cálculos previos que deben ser desarrollados para una correcta implementación y comunicación entre Arduino, la interfaz de entrada de la Planta De Lorenzo DL 2314 y la nueva interfaz de salida, lo que involucra un estudio y recopilación de información relacionados a Divisores de Voltaje, Amplificadores de Potencia y Transistores (Electrónica Análoga-Digital).

4.1 INTERFAZ DE ENTRADA

La Planta De Lorenzo DL 2314 en su interfaz de entrada maneja variables de instrumentación análogas (Nivel, Flujo, Temperatura, Presión), éstas son medidas por sensores que son los encargados de adaptar el tipo de señal leído a otro tipo de señal para que pueda ser interpretada por un controlador, en este caso en valores de Tensión que oscilan entre 0[v] y 10[v]. Arduino Uno Rev3, para leer estos valores posee Pines analógicos, sin embargo, el límite permitido corresponde a 5[v] por Pin. Es por esta razón que es necesario realizar un divisor de voltaje que reduzca los 10[v] entregados por el sensor, a 5[v] soportados por la entrada análoga del Arduino.

4.1.1 Divisor de Voltaje

Un divisor de voltaje es un circuito que reparte la tensión de una fuente entre una o más resistencias conectadas. La fuente de voltaje V_{IN} corresponde a los 10[v] entregados por el sensor de cada variable y son dos las resistencias que están en serie (R_1 y R_2) que permiten una salida V_{OUT} de 5[v].

La ecuación propuesta es la siguiente:

$$Vout = \left[Vin\ x\ \left(\frac{R2}{R1 + R2}\right)\right]$$

Donde:

- V_{OUT}: Voltaje de Salida.
- $V_{IN:}$ Voltaje de Entrada (Sensor).
- R_1: Resistencia uno.
- R_2: Resistencia dos.

Considerando lo siguiente, si R_1 y R_2 son iguales, entonces el voltaje de salida será la mitad del voltaje de entrada. Esto aplica independientemente de los valores de las resistencias. 10[KΩ] fue el valor escogido para proteger los Pines análogos del controlador Arduino.

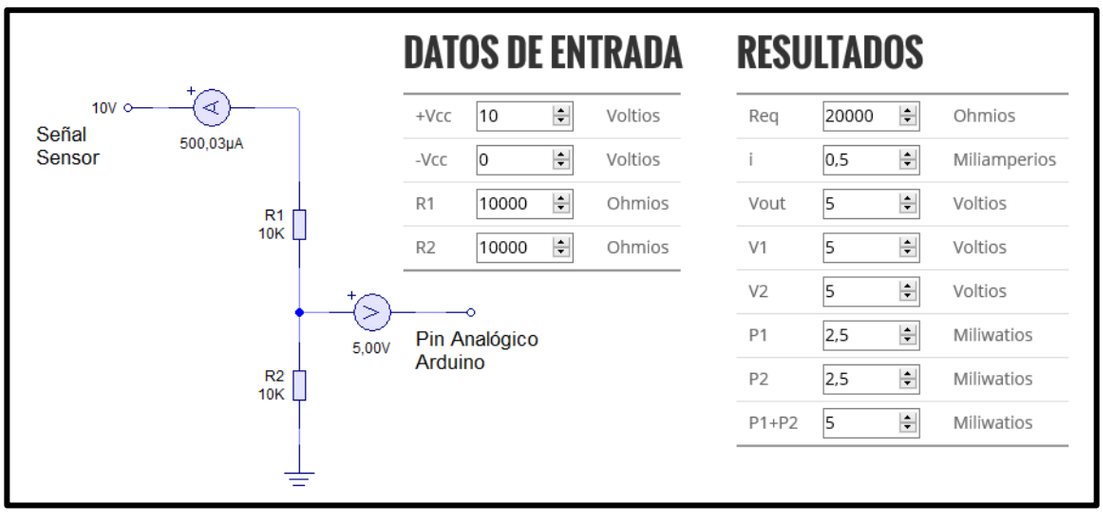


Figura 4-1: Divisor de Voltaje Propuesto.

4.2 INTERFAZ DE SALIDA

La interfaz de salida posee una Bomba de Nivel, una Válvula Motorizada, una Calentador y una Válvula Solenoide, cada uno de ellos requiere un voltaje para operar, además de consumir determinada corriente al estar en funcionamiento. (Ver Tabla 2-1).

A partir de esta tabla, se mencionó la necesidad que existe de sustituir la etapa de potencia de la Planta De Lorenzo DL 2314, por otra que se ajuste a la capacidad del controlador y a la necesidad del proyecto, ya que, de acuerdo a lo estudiado, Arduino es capaz de entregar 5[v] y una corriente recomendada de 20[$_{m}$A] por Pin, valores que no se ajustan a lo que necesita cada actuador.

Debido a esto fue necesario realizar un estudio orientado a los transistores utilizados como Amplificadores de Corriente y Puente H para inversión de giro en motores.

4.2.1 Principio de funcionamiento del Transistor

Los transistores son dispositivos semiconductores que poseen tres terminales de conexión, llamados Base (B), Colector (C), y Emisor (E). Tienen la particularidad de controlar grandes cantidades de corriente a partir de una corriente muy pequeña, siendo éste precisamente su funcionamiento.

La señal de entrada es quien determinará su comportamiento, por lo tanto, el transistor puede comportarse como amplificador de señal, conmutador, oscilador, y rectificador de señales. Para el proyecto se estudiaron los transistores BJT, los cuales se caracterizan por ser del tipo NPN, bipolares y por la cualidad de amplificar corriente, ya que ésta aumenta a medida que aumenta la corriente de Base.

Su configuración difiere de los PNP, debido a que en los NPN se necesita aplicar un voltaje positivo en el Colector para que la corriente fluya desde el Colector hacia el Emisor, esta configuración se conoce como Amplificación con Emisor común.

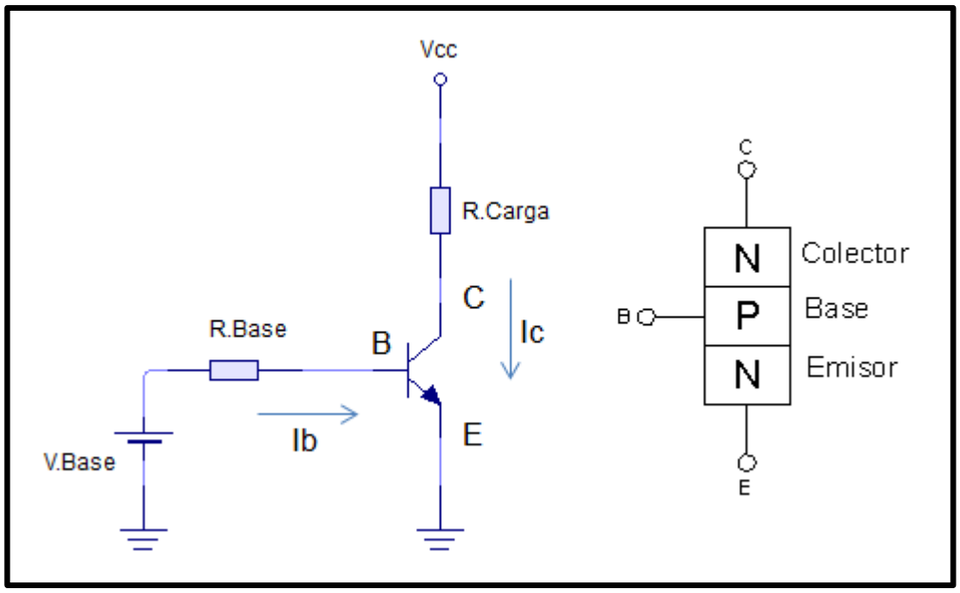


Figura 4-2: Transistor Bipolar NPN, Emisor Común.

Sus zonas de trabajo son:

- **Corte**: No circula intensidad por la Base (I_B=0), por lo tanto, la intensidad de Colector (I_C) y Emisor (I_E) son nulas. La tensión entre Colector y Emisor es la de la fuente ($V_{CE} = V_{CC}$), y éste se comporta como un interruptor abierto.

- **Saturación**: Al circular una intensidad por la Base, se aprecia un aumento de la corriente en el Colector. La tensión entre Colector y Emisor es cero (V_{CE} = 0), por lo tanto, el voltaje de la fuente recae en la carga conectada al Colector, comportándose como un interruptor cerrado.

- **Activa:** Es cuando el transistor funciona de forma normal.

La ganancia de corriente es un parámetro también importante para los transistores, ya que relaciona directamente la variación que sufre la corriente de Colector, producto de los cambios de corriente en la Base. Normalmente se denomina hFE y se expresa como ß = I_C / I_B.

4.2.1.1 TIP120

El transistor escogido para la implementación de la nueva etapa de potencia de la Bomba de Nivel y el Calentador, es el TIP120, su elección se debe a que cumple con los requisitos necesarios para hacer funcionar ambos actuadores, ya que de acuerdo a su Datasheet posee una Ganancia de Corriente (hFE) de 2500 y es capaz de suministrar una Intensidad (I_C) de 5[A].

Hay que recordar que la bomba de nivel es un motor que requiere 12[v] y consume 1.62[A] en funcionamiento. Por otro lado, el calentador es una resistencia que convierte la energía eléctrica en calor y que por defecto utiliza 48[v] y consume 3.86[A], pero para nuestro caso se alimentará con 12[v].

Volviendo a las características del TIP120, el factor determinante a la hora de escoger éste por sobre otros, fue su ganancia, ya que a mayor hFE, mayor será la corriente de Colector suministrada y menor será la corriente de Base (I_B) requerida, punto muy importante si se consideran los 20[$_m$A] entregados por los Pines de Arduino. Para determinar la corriente de Base (I_B) que necesitará de Arduino, se debe identificar la Corriente de Colector (I_C), la cual corresponde a la corriente de la resistencia de carga, posteriormente identificar el hFE de acuerdo al Datasheet y finalmente multiplicar el valor obtenido por un factor de seguridad.

La ecuación propuesta es la siguiente:

$$Ib = \left[FS\ x\ \left(\frac{Ic}{ß}\right)\right]$$

Donde:

- I_B: Corriente de Base.
- FS: Factor de Seguridad.
- I_C: Corriente de Colector (Motor).
- ß: Ganancia del transistor.

4.2.1.2 Puente H, L293D

El Puente H L293D es un circuito integrado que posee cuatro canales capaces de suministrar 600[mA] cada uno, y soporta corrientes peak de hasta 1.2[A]. Cada canal es controlado por señales TTL (Lógica de Compuertas), siendo esta una característica fundamental a la hora de operar, ya que los 5[v] que entrega Arduino son suficientes para activar el canal.

Su estructura interna facilita la posibilidad de usar dos voltajes diferentes, es decir, uno para la propia alimentación del Circuito Integrado (normalmente se utiliza el pin +5[v] de Arduino) y el otro para la alimentación del motor, el cual puede ser una batería o fuente externa.

Tiene la capacidad de controlar 2 motores a la vez, con la particularidad de invertir su giro y regular su velocidad, sin embargo, para el proyecto será utilizado para invertir el giro de la válvula motorizada y para activar la válvula solenoide.

Tabla 4-1: Características Técnicas Puente H L293D.

Voltaje de Alimentación (Motores) Vcc2: 4.5[v] a 36[v]
Voltaje de Alimentación Puente H Vcc1: Máx. 7[v]
Corriente de salida: 600 [mA] por canal.
Corriente Peak de salida: 1.2[A] por canal (no repetitiva).
Fuentes de alimentación separadas.
Protección contra exceso de temperatura.
Diodos de protección incorporados.
Alta inmunidad al ruido

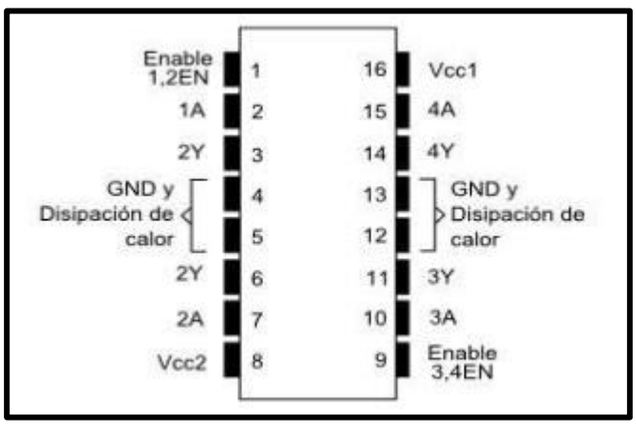


Figura 4-3: Distribución de Pines Puente H L293D.

4.2.2 Nueva etapa de potencia para Bomba de Nivel

En este punto, se desarrollarán los cálculos para obtener la corriente de base (I_B), así como también el cálculo de la resistencia de base (R_B). Los valores obtenidos son los que definen el comportamiento del transistor (Corte/Saturación), y se utilizarán en la implementación de la nueva etapa de potencia.

Datos:

- Corriente de Motor (I_C): 1.62[A]
- Voltaje de Operación Teórico (Vcc): 12[v]
- Ganancia del transistor ß: 2500
- Voltaje de Base (V_B): 4.3[v]
- Factor de Seguridad FS: 4

Nota1: El voltaje de Base (V_B) Corresponde a la diferencia entre el voltaje del Pin Arduino y la caída de tensión del Transistor (5[v] – 0.7[v]).

Nota2: El factor de seguridad es un valor que se utiliza para asegurar la saturación del Transistor, normalmente se utilizan los dígitos 2,3 y 4.

Cálculo I_B:

$$Ib = \left[4\ x\ \left(\frac{1.62\ [A]}{2500}\right)\right]$$

$$Ib = 2.592\ [mA]$$

Cálculo R_B:

Por Ley de Ohm se tiene la siguiente ecuación:

$$Vb = [Ib\ x\ Rb]$$

Despejando R_B:

$$Rb = \left[\left(\frac{4.3[v]}{2.595\ [\mathrm{mA}]}\right)\right]$$

$$Rb = 1658\ [\Omega]$$

$$Valor\ de\ Comercilaización$$

$$Rb \approx 1500\ [\Omega]$$

Tabla 4-2: Resumen de valores, nueva etapa de Potencia de Nivel.

Valores TIP120 Saturado				
Voltaje de Base [V_B]	Resistencia de Base [R_B]	Corriente de Base [I_B]	Voltaje de Operación [Vcc]	Corriente de Carga y/o Colector [I_C]
4.3[v]	1500[Ω]	2.595[mA]	12[v]	1.62[A]

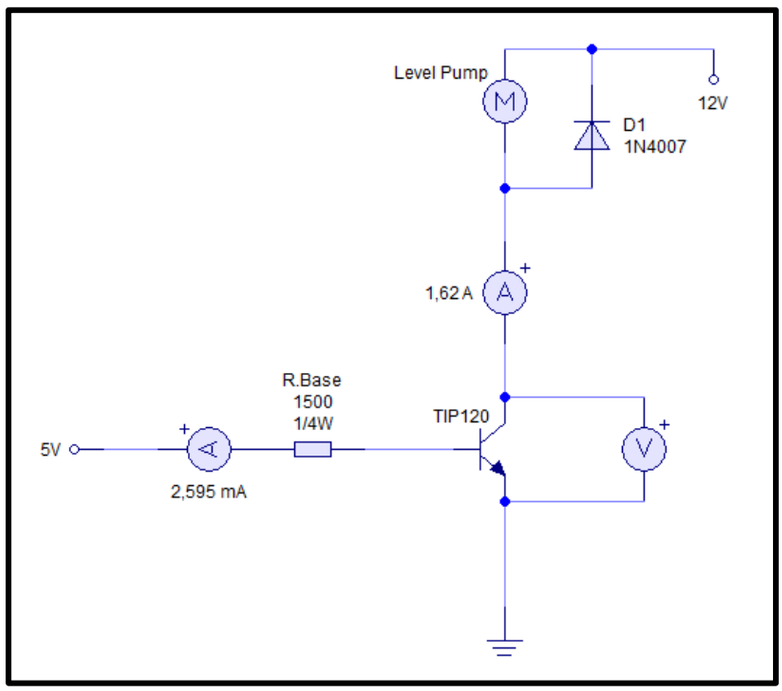


Figura 4-4: Diagrama de Conexionado Propuesto, Nueva Etapa de Potencia para Bomba de Nivel.

4.2.3 Nueva etapa de potencia para el Calentador de Temperatura

Se mantiene el mismo principio de funcionamiento que fue considerado en la etapa de potencia para la bomba de nivel.

Datos:

- Corriente del Calentador (I_C): 3.86[A]
- Voltaje de Operación Teórico (Vcc): 12[v]
- Ganancia del transistor ß: 2500
- Voltaje de Base (V_B): 4.3[v]
- Factor de Seguridad FS: 4

Cálculo I_B:

$$Ib = \left[4\ x\ \left(\frac{3.86\ [A]}{2500}\right)\right]$$

$$Ib = 6.176\ [mA]$$

Cálculo R_B:

Por Ley de Ohm se tiene la siguiente ecuación:

$$Vb = [Ib\ x\ Rb]$$

Despejando R_B:

$$Rb = \left[\left(\frac{4.3[v]}{6.176\ [\mathrm{mA}]}\right)\right]$$

$$Rb = 696.24\ [\Omega]$$

$$Valor\ de\ Comercilaización$$

$$Rb \approx 680\ [\Omega]$$

Tabla 4-3: Resumen de valores, nueva etapa de Potencia Calentador de Temperatura.

Valores TIP120 Saturado				
Voltaje de Base [V_B]	Resistencia de Base [R_B]	Corriente de Base [I_B]	Voltaje de Operación [Vcc]	Corriente de Carga y/o Colector [I_C]
4.3[v]	680[Ω]	6.176[mA]	12[v]	3.86[A]

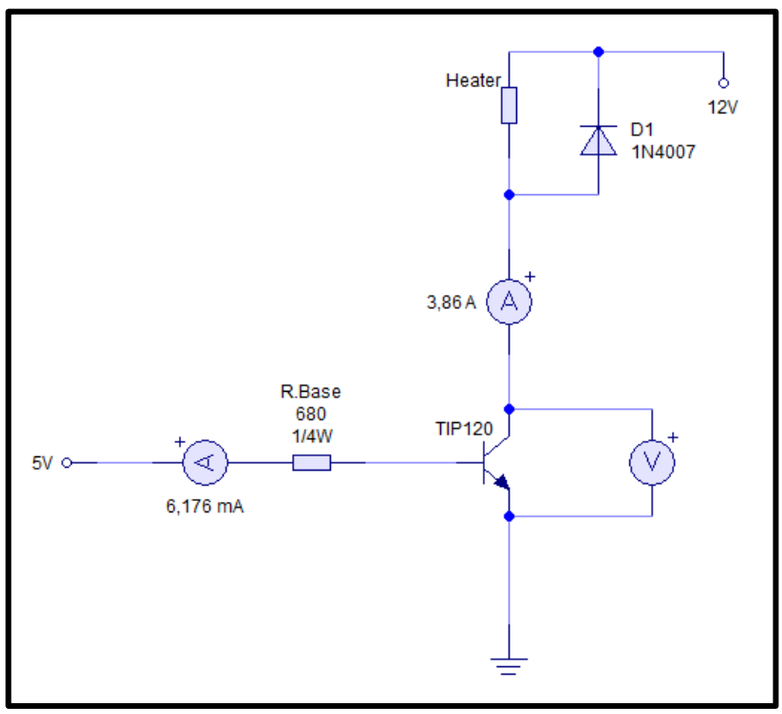


Figura 4-5: Diagrama de Conexionado Propuesto, Nueva Etapa de Potencia para el Calentador.

4.2.4 Nueva etapa de potencia para Válvula Motorizada y Válvula Solenoide

La etapa de potencia para estos actuadores será realizada por el puente H L293D, quien será el encargado de cambiar el giro de la Válvula Motorizada lo que se traduce en la apertura y el cierre de ésta, y también será el encargado de abrir y cerrar la válvula solenoide para efectos de liberar o mantener una presión dentro del estanque de la Planta De Lorenzo DL 2314.

A continuación, se muestra la imagen del diagrama de conexionado propuesto para que esto sea posible.

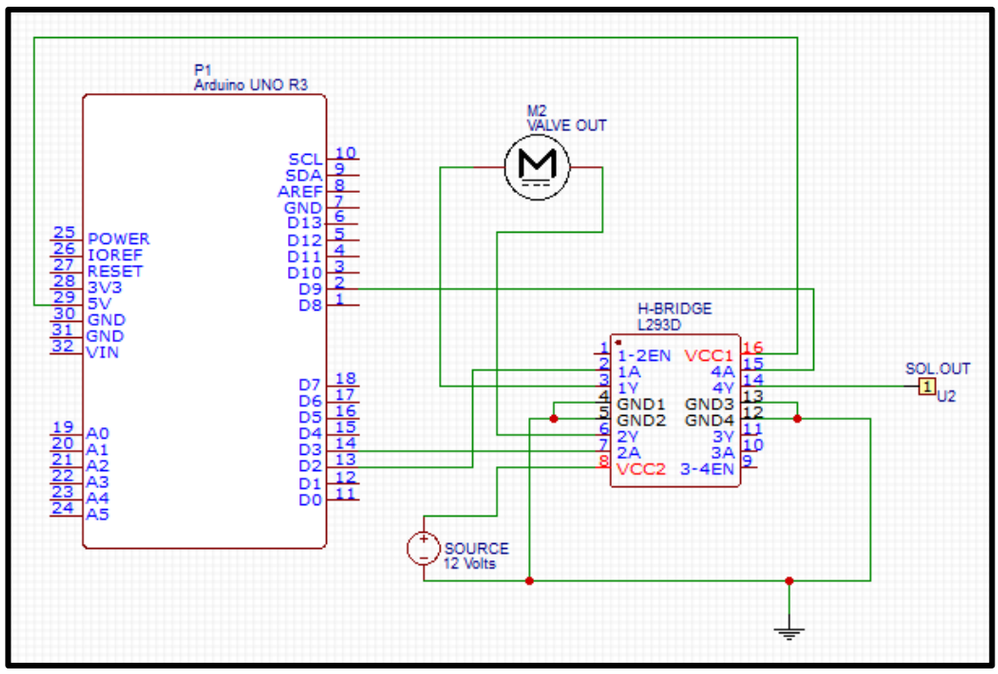


Figura 4-6: Diagrama de Conexionado Propuesto, Nueva Etapa de Potencia para Válvula Motorizada y Válvula Solenoide.

El Drive L293D utiliza dos de sus canales para realizar la inversión de giro sobre la válvula motorizada, su funcionamiento se logra gracias a los pines 1A y 2A, los cuales reciben la señal de control de Arduino y activan las compuertas lógicas internas según la solicitud. Las salidas de estos canales son 1Y y 2Y respectivamente, las que se deben conectar a la válvula motorizada para evidenciar el cambio de giro.

Un tercer canal (4A/4Y) es utilizado para abrir y/o cerrar la válvula solenoide, su funcionamiento es simple, ya que se activa o apaga según la orden de Arduino.

Como se observa en la imagen 4-6, es necesario alimentar el Puente H con un voltaje de 5[v], para este caso se utiliza el Pin +5[v] de Arduino conectado directamente al Pin 16 (Vcc1) del Drive L293D, el valor máximo permitido es 7[v], valor que se debe respetar, de lo contrario se quemarán las compuertas lógicas internas que componen al Drive.

Por otro lado, el Pin que permite la alimentación para el motor y/o las cargas asociadas es el Pin 8 (Vcc2), el cual soporta valores desde 4.5[v] hasta 36[v]. La particularidad de tener voltajes de alimentación independientes, es sin duda una gran ventaja ya que facilita el uso de motores de mayor consumo de tensión.

CAPITULO 5:

IMPLEMENTACIÓN Y MONTAJE DE DISPOSITIVOS

5. IMPLEMENTACIÓN Y MONTAJE DE DISPOSITIVOS.

Este capítulo considera la implementación de cada uno de los divisores de voltaje que serán utilizados por cada variable de control, además de la construcción de la nueva etapa de potencia que permitirá realizar acciones de operación sobre los actuadores, las que en conjunto componen el nuevo Sistema de Control y que facultaran el desarrollo de ensayos.

5.1 IMPLEMENTACIÓN GENERAL DEL SISTEMA DE CONTROL

La distribución general del nuevo sistema de control fue realizada considerando la estructura que posee el Panel de Control de la Planta De Lorenzo DL 2314, lo que significa que por el lado izquierdo se tienen los Divisores de Voltaje y por el lado derecho se tiene la nueva Etapa de Potencia, las cuales se deberán conectar con la Interfaz de Entrada y Salida de la Planta respectivamente.

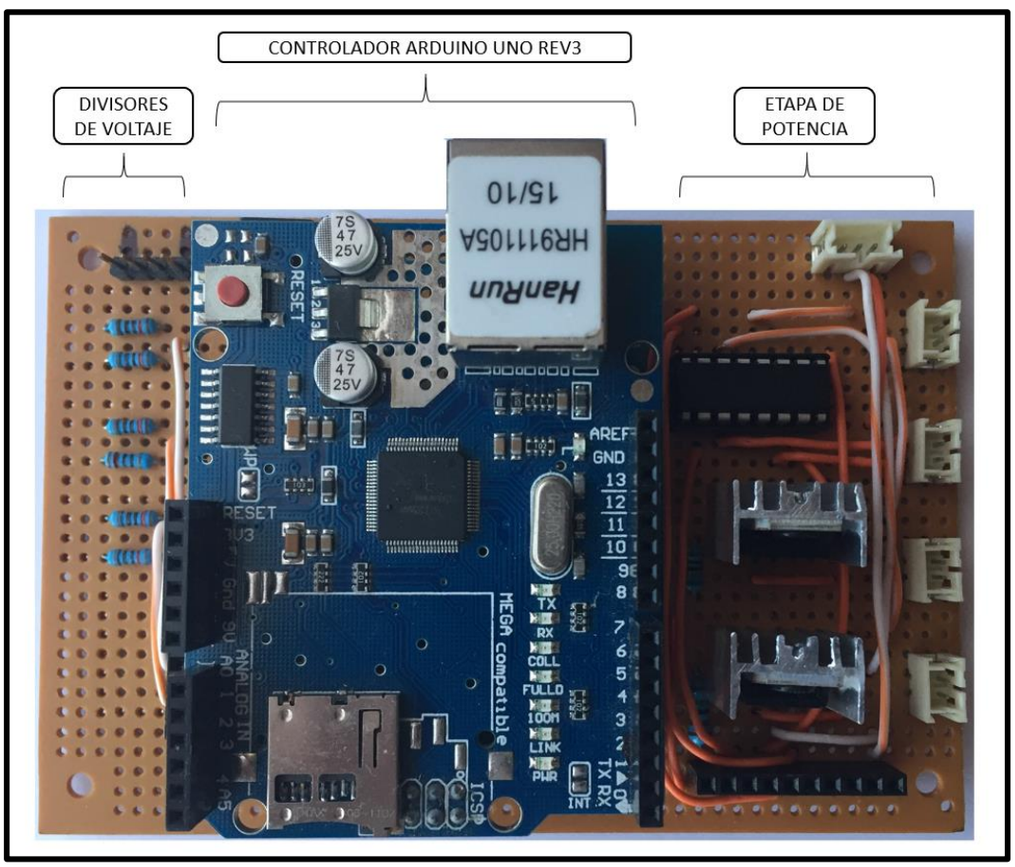


Figura 5-1: Distribución Final de Dispositivos en Placa Universal.

5.2 IMPLEMENTACIÓN DE DIVISORES DE VOLTAJE

En la práctica solo se construyeron 3 divisores de voltaje, los cuales corresponden a los que serán conectados con la interfaz de entrada de la Planta De Lorenzo DL 2314, es decir a la salida de los sensores de nivel, temperatura y presión. La razón por la cual no se construyó el divisor para el flujo, es debido a que el sensor está descompuesto.

Materiales:

- 6 resistencias de 10 [KΩ]. ¼ Watts.

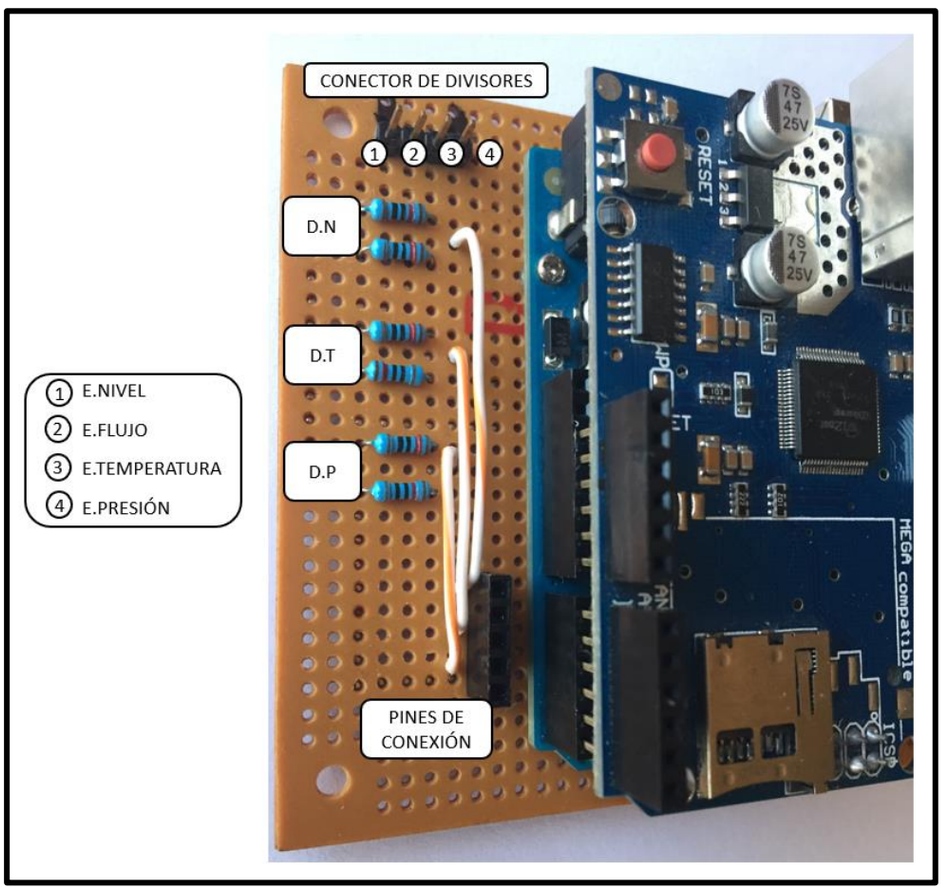


Figura 5-2: Construcción de Divisores de Voltajes.

En la figura 5-2, se muestra la construcción de los divisores propuestos,

En Donde:

- CONECTOR DE DIVISORES: Corresponde al conector que permitirá la conexión con la salida de cada sensor de la Planta De Lorenzo DL 2314.

- D.N: Corresponde al Divisor de Voltaje para el Nivel.
- D.T: Corresponde al Divisor de Voltaje para la temperatura.
- D.P: Corresponde al Divisor de Voltaje para la Presión.
- PINES DE CONEXIÓN: Permitirá la interconexión de la salida de los divisores con los Pines análogos de Arduino para su posterior lectura.

5.3 IMPLEMENTACIÓN NUEVA ETAPA DE POTENCIA

La nueva Etapa de Potencia, fue construida considerando los cálculos teóricos anteriormente expuestos, ésta se deberá conectar con la interfaz de salida de la Planta De Lorenzo DL 2314, es decir con cada actuador disponible.

Materiales:

- 1 resistencia de 1.5 [KΩ]. ¼ Watts.
- 1 resistencia de 680 [Ω]. ¼ Watts.
- 2 diodos rectificadores 1N4007.
- 2 transistores TIP 120.
- 1 Puente H L293D.

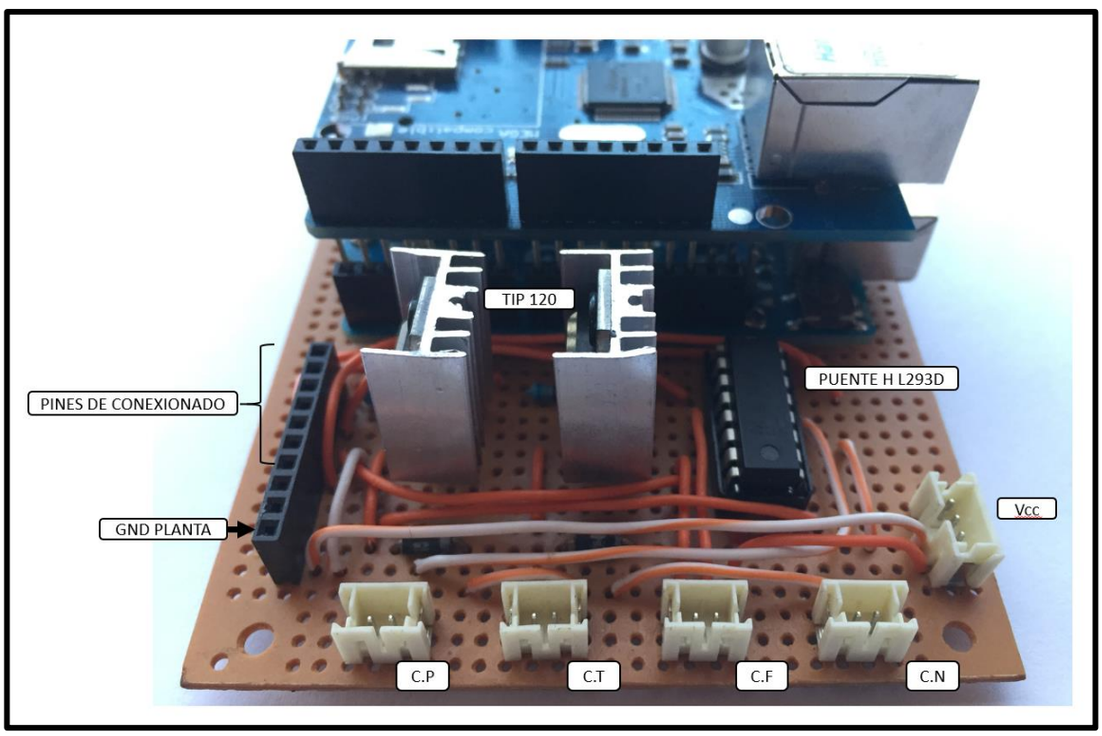


Figura 5-3: Construcción Etapa de Potencia.

En la figura 5-3, se muestra la construcción de la etapa de potencia.

En Donde:

- PUENTE H L293D: Permitirá el cambio de giro de la válvula motorizada.
- TIP120: Amplía la corriente suministrada por Arduino.
- PINES DE CONEXIONADO: Permitirá la interconexión de las salidas de Arduino con los elementos utilizados en la construcción de la etapa de potencia.
- GND PLANTA: Conector que permitirá la conexión con la Tierra (GND) de la Planta De Lorenzo DL 2314.
- Vcc: Corresponde al conector que permitirá la alimentación de las etapas de potencia.
- C.N: Corresponde al conector que permitirá la conexión con la bomba de nivel de la Planta.
- C.F: Corresponde al conector que permitirá la conexión con la válvula motorizada de flujo de la Planta.
- C.T: Corresponde al conector que permitirá la conexión con el calentador de la Planta.
- C.P: Corresponde al conector que permitirá la conexión con la válvula solenoide de la Planta.

5.3.1 Conexión del Sistema de Control con la Planta De Lorenzo DL 2314

Para lograr una correcta lectura de datos y ejecución de solicitudes, es necesario conectar todos los equipos y dispositivos de manera correcta, las instrucciones son las siguientes:

A. Conectar la salida de cada sensor ubicado en la Planta (PIN 13), con el conector de los divisores de voltaje.
B. Conectar cada sócalo de la nueva etapa de potencia con los actuadores de la Planta.
C. Conectar sócalo "GND Planta" con la "Tierra (GND)" de la Planta.
D. Conectar Fuente de Poder en el sócalo Vcc.
E. Conectar router al Ethernet Shield.
F. Energizar Arduino, a través del cable USB o el Jack de poder.
G. Accionar botón de encendido de la Fuente de Poder.
H. Accionar botón de encendido de la Planta De Lorenzo DL 2314.
I. Encender dispositivo Android, acceder a la red Wifi "Planta De Lorenzo DL 2314" y ejecutar Aplicación "Proyecto de Titulo".
J. Realizar ensayos.

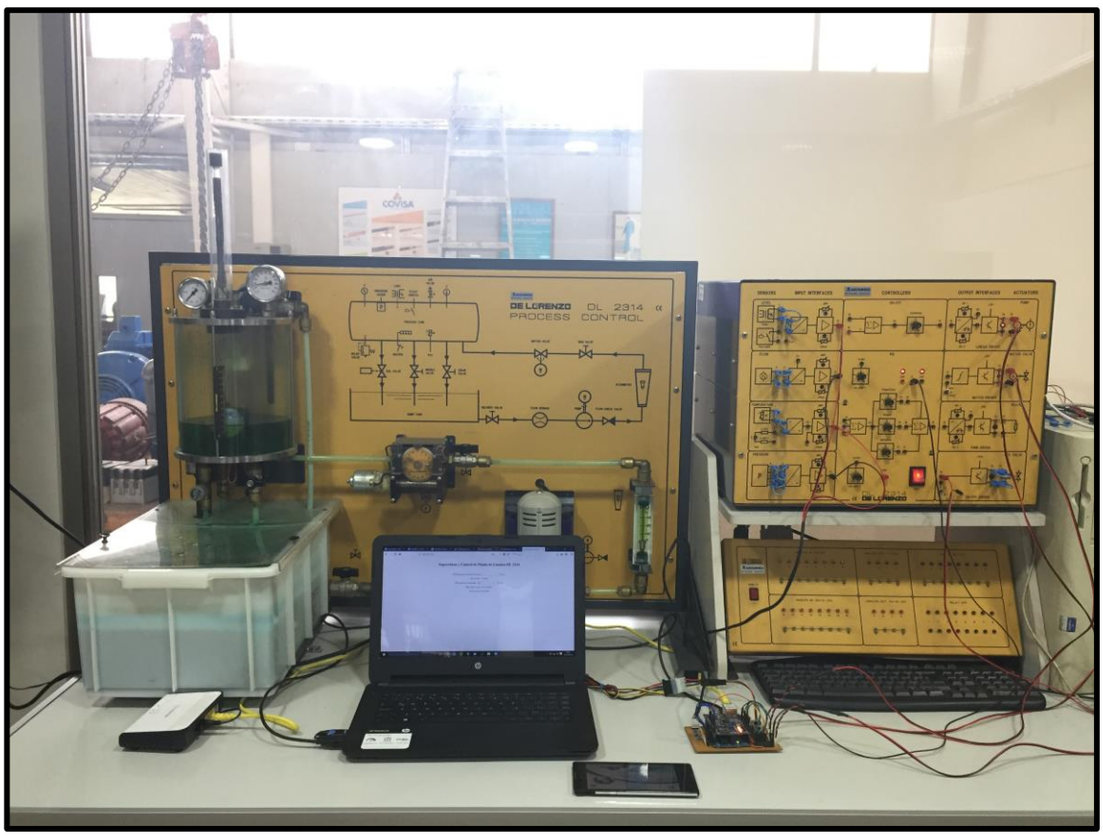


Figura 5-4: Montaje Final de Equipos y Dispositivos.

CAPITULO 6:

ENSAYOS Y RESULTADOS

6. ENSAYOS Y RESULTADOS

Para verificar el correcto funcionamiento del Montaje y la Aplicación desarrollada para Android, es necesario realizar ensayos que permitan visualizar la respuesta por parte de la Planta De Lorenzo DL 2314, esto se traduce en correctas lecturas de las variables, así como también en un correcto comportamiento de los actuadores.

6.1 VARIABLES A VISUALIZAR

Las variables a considerar en estos ensayos son los que comúnmente están presente en la mayoría de los procesos industriales.
Los cuales son:

- Lectura en tiempo real de Nivel
- Lectura en tiempo real de Temperatura.
- Lectura en tiempo real de presión.
- Estado de la Bomba de Nivel (Run/Stop/PWM).
- Estado de la Válvula Motorizada (Open/Close/OFF).
- Estado del Calentador (ON/OFF).
- Estado de la Válvula Solenoide (Open/Close).

6.2 ENSAYOS

Antes de comenzar con los ensayos, fue necesario ajustar la Ganancia y el OFFSET que tiene cada sensor de control que posee la Planta, esto para lograr una correcta lectura de datos de acuerdo al escalar propuesto en la programación de Arduino. Los valores deben ser los siguientes.

- Sensor de Nivel: 10[v] = 10 [cm].
- Sensor de Temperatura: 10[v] = 40 [ºC].
- Sensor de Presión: 10[v] = 2 [BAR].

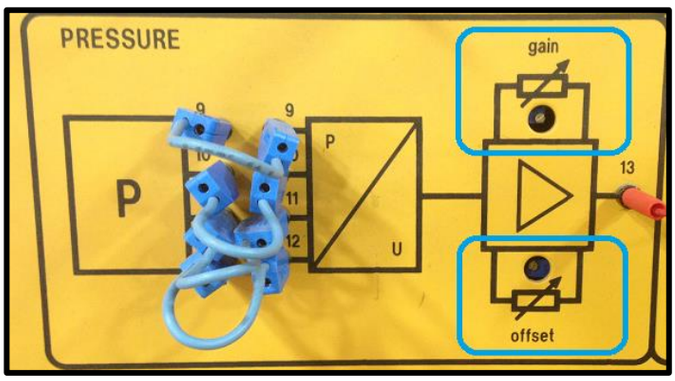


Figura 6-1: Ganancia y OFFSET.

La forma correcta de interpretar los valores expuestos es simple, básicamente se indica que para una salida de voltaje de 10[v] en cada sensor, el nivel, la temperatura y la presión deben ser 10[cm], 40[ºC] y 2[BAR] respectivamente. Para obtener la lectura del voltaje se debe conectar el terminal positivo del multímetro al PIN13 y el COM al GND de la Planta, finalmente mientras se leen los valores se manipulan los tornillos de Ganancia y OFFSET que se muestran en la imagen 6-1.

6.2.1 Accionamiento de Bomba de Nivel y Lectura de Nivel

Una vez conectados todos los dispositivos se procede a accionar la bomba de nivel con el botón de la aplicación en Android, su comportamiento es correcto y se traduce en el accionar de esta, posteriormente se procede a realizar el control PWM, para ello se desplaza la barra slider, como resultado se tiene una variación en la velocidad, por último, se apaga la bomba con el botón indicado.

La lectura del nivel está dada en centímetros, y esta se visualiza y actualiza cada 2 segundos, sus valores en la medida que asciende el nivel es correcto.

A continuación, se adjunta una de las imágenes que lo evidencia.

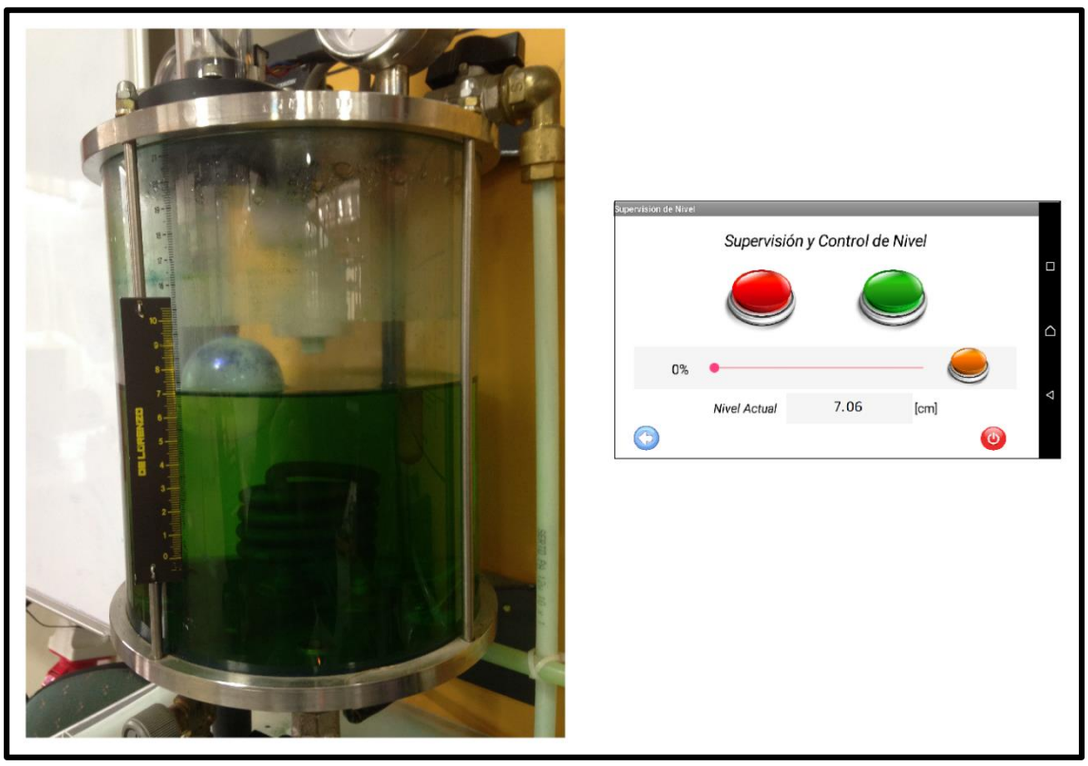


Figura 6-2: Lectura de Nivel.

6.2.2 Accionamiento de Válvula Motorizada.

El ensayo realizado para la válvula motorizada fue exitoso, esto se traduce en una disminución o aumento del flujo que llega al estanque de nivel, el motivo se debe a que al generar un cierre de 40º se le obstruye el paso al caudal, no así al generar la apertura total (0º).

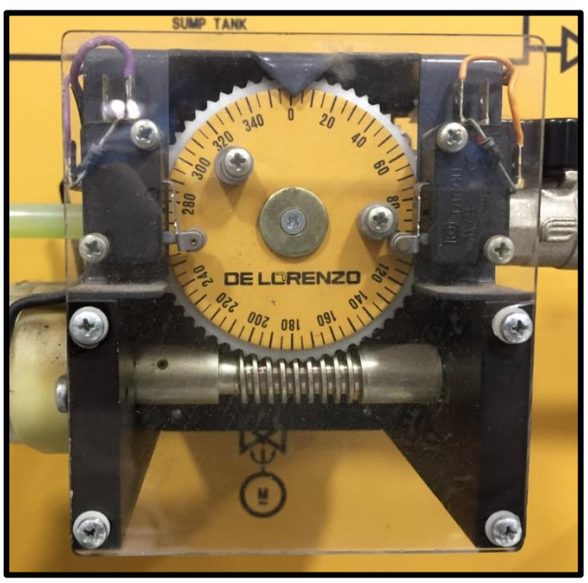


Figura 6-3: Válvula Motorizada en Apertura Total.

6.2.3 Lectura de Temperatura

La lectura se realizó considerando la temperatura ambiente, ya que el proceso para calentar el líquido es lento. Debido a esto, el escalar propuesto para generar la lectura puede evidenciar leves variaciones con respecto al valor mostrado por el Termómetro.

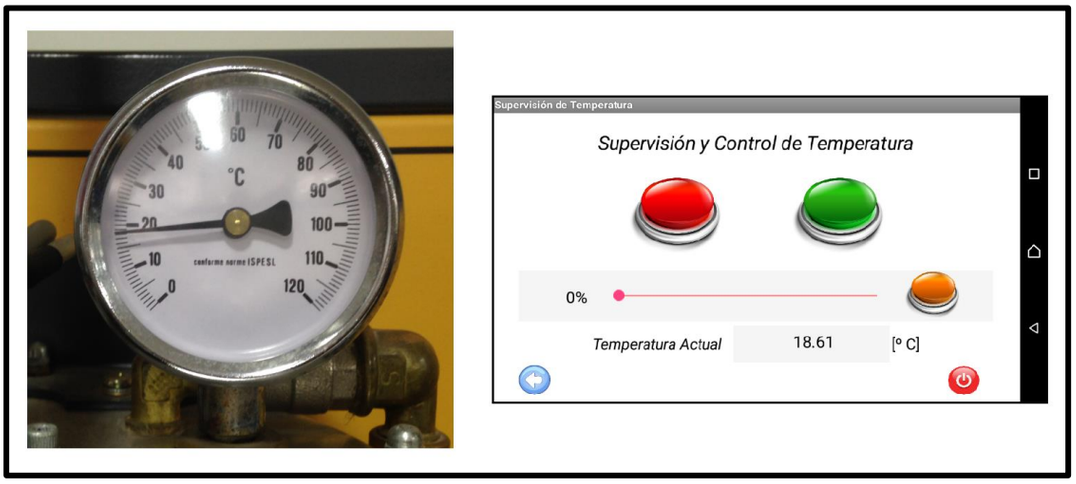


Figura 6-4: Lectura de Temperatura.

6.2.3 Lectura de Presión y Accionamiento de Válvula Solenoide.

Para generar ensayos entorno a esta variable, es necesario cerrar de forma manual la válvula de Alivio que está por detrás del termómetro, esto con el fin de poder leer la presión interna del estanque.

En la medida que asciende el nivel también lo hace la presión, para controlarla se procede a accionar la válvula solenoide de despiche, la que finalmente libera nivel y permite variar la presión.

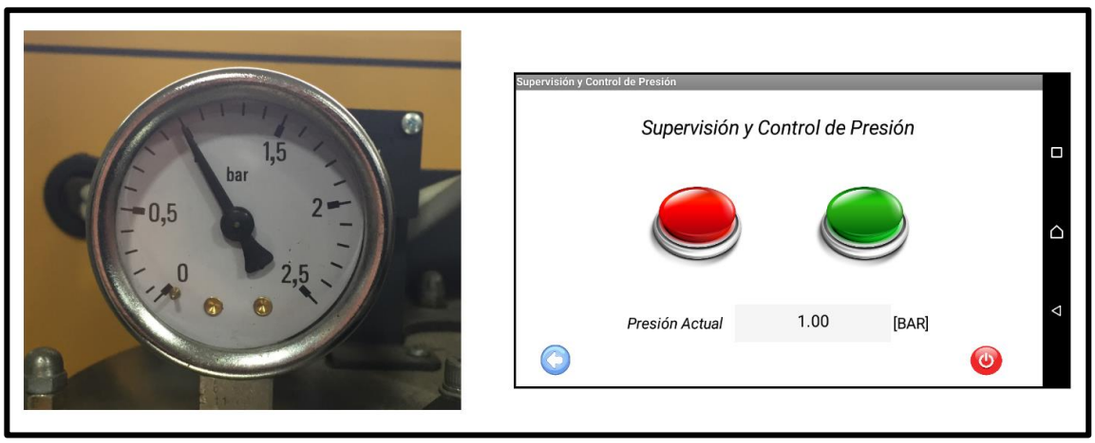


Figura 6-5: Lectura de Presión.

CONCLUSIÓN

La implementación del sistema de control desarrollado en este documento, cumple con los requerimientos mencionados en la introducción del mismo. El objetivo principal que consistía en crear una aplicación para dispositivos Android que permitiera supervisar y controlar todas las variables asociadas a los lazos de control de la Planta De Lorenzo DL 2314 de forma remota, se cumplió en su totalidad.

La aplicación desarrollada, permite una conexión directa entre el/los dispositivos Android y la Planta De Lorenzo DL 2314, lo que se traduce en la correcta lectura de todas las variables asociadas a los lazos de control disponible, así como también en una correcta ejecución de las acciones de operación sobre los actuadores.

A lo largo del proceso que implicó la creación del sistema de supervisión y control, se utilizaron diversas técnicas relacionadas al acondicionamiento de señales, como por ejemplo, amplificadores operacionales LM 741 en modo no inversor e inversor para la obtención de voltajes positivos y negativos, transistores 2n2222 como amplificadores de corriente y por último módulos relés para la conmutación de señales, sin embargo su funcionamiento no fue del todo favorable, ya que el amplificador era inestable y las corrientes suministradas por los transistores no cumplían con las condiciones para ser utilizados, debido a eso la temperatura de los mismos era un riesgo constante. Es por estas razones que finalmente se decantó por los dispositivos descritos en este informe, los que resultaron ser una opción viable y confiable a la hora de su funcionamiento, pese a ello no significa que sean la mejor opción, dejando abierta la posibilidad de búsqueda para la mejor alternativa.

Respecto de la aplicación desarrollada para los dispositivos inteligentes Android, ésta queda disponible para futuras mejoras, relacionadas por ejemplo al análisis gráfico de tendencias para cada variable, almacenamiento de históricos, gestión de alarmas y por último el mejoramiento de las estrategias de control empleadas.

En relación al Hardware, las líneas de trabajo que se pueden abordar son el diseño y construcción de Cables Plug and Play, diseño de una PCB con acabado profesional a partir del prototipo expuesto y finalmente mejorar el diseño de los acondicionadores de señal para una reducción de tamaño.

Algunas recomendaciones con respecto al trabajo realizado, son por ejemplo la confección de los cables y los tiempos empleados en su construcción, se deben considerar además los tiempos de espera a la hora de solicitar algún material que no cuente con stock en la zona y, por último, poner especial énfasis en los equipos escogidos y la configuración utilizada.

LINKOGRAFÍA

- Aprendiendo Arduino. [en línea] < https://aprendiendoarduino.wordpress.com/> [consulta: 1 de noviembre 2016]
- Arduino Uno R3. [en línea] <https://www.infootec.net/arduino/> [consulta: 1 noviembre 2016]
- Modelo OSI. [en línea] < http://belarmino.galeon.com/> [consulta: 17 noviembre 2016].
- Estructura Modelo OSI. [en línea] <http://aprendoconcalle.blogspot.com/2015/08/modelo-de-referencia.html> [consulta: 17 noviembre 2016].
- Redes Locales y Globales. [en línea] <https://sites.google.com/site/redeslocalesyglobales/home> [consulta: 25 noviembre 2016].
- SDK Android. [en línea] < https://www.androidpit.es/sdk-android> [consulta: 4 abril 2017].
- Software de Comunicaciones, Android. [en línea] <https://sites.google.com/site/swcuc3m/home/android> [consulta: 9 junio 2017].
- Cliente- Servidor. [en línea] < https://www.ecured.cu/Cliente-Servidor> [consulta: 17 julio 2017].
- Transistor BJT. [en línea] < https://www.luisllamas.es/salidas-mayor-potencia-arduino-transistor-bjt/> [consulta: 21 agosto 2017].
- Corte y Saturación, Transistor BJT. [en línea] <http://mrelbernitutoriales.com/corte-y-saturacion-bjt/> [consulta: 22 agosto 2017].
- Puente H. L293D. [en línea] <https://www.prometec.net/hbridge/#> [consulta: 24 agosto 2017].
- Puente H, Inversor de Giro en Motores. [en línea] <http://panamahitek.com/el-puente-h-invirtiendo-el-sentido-de-giro-de-un-motor-con-arduino/> [consulta: 24 agosto 2017].

ANEXOS

ANEXOS

- A1 - DataSheets
    - Arduino Esquemático.
    - DataSheet 1N4007.
    - DataSheet Puente H L293.
    - DataSheet TIP120-D.
    - DataSheet W5100.
    - Ethernet Shield Esquemático.

- A2 - Diagramas
    - Diagrama de Conexionado Esquemático Etapa de Potencia.
    - Diagrama de Flujo Programación Arduino.
    - PCB Sistema de Supervisión y Control.

- A3 - Guía Rápida de Configuración.
- A4 - Programación de la Aplicación en Android.
- A5 - Programación de la Aplicación en Arduino.