ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERIA ELECTRONICA EN TELECOMUNICACIONES Y REDES

"DISEÑO DE UN PROTOTIPO DE SISTEMA DE PARQUEO INTELIGENTE PARA EL EDIFICIO DE LA FIE UTILIZANDO TECNOLOGÍAS BASADO EN EL INTERNET DE LAS COSAS"

TRABAJO DE TITULACIÓN

Tipo: DISPOSITIVO TECNOLÓGICO

Presentado para optar al grado académico de:

INGENIERO EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES

AUTOR: ESTALIN JAVIER SALTOS TAIPE

TUTOR: ING. MARCO VINICIO RAMOS VALENCIA MsC.

Riobamba-Ecuador

2018

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERÍA ELECTRÓNICA EN TELECOMUNICACIONES Y REDES

El Tribunal de trabajo de titulación certifica que: El trabajo de titulación: **DISEÑO DE UN PROTOTIPO DE SISTEMA DE PARQUEO INTELIGENTE PARA EL EDIFICIO DE LA FIE UTILIZANDO TECNOLOGIAS BASADO EN EL INTERNET DE LAS COSAS,** de responsabilidad del señor Estalin Javier Saltos Taipe, ha sido minuciosamente revisado por los Miembros del Tribunal del trabajo de titulación quedando autorizado su presentación.

NOMBRE	**FIRMA**	**FECHA**
Dr. Julio Santillán **VICEDECANO DE LA FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**	_____	_____
Ing. Patricio Romero **DIRECTOR DE LA ESCUELA DE INGENIERÍA ELECTRÓNICA, TELECOMUNICACIONES Y REDES**	_____	_____
Ing. Vinicio Ramos MsC. **DIRECTOR DE TRABAJO DE TITULACIO**	_____	_____
Ing. Jorge Yuquilema **MIEMBRO DEL TRIBUNAL**	_____	_____

Yo, Estalin Javier Saltos Taipe soy responsable de las ideas, doctrinas y resultados expuestos en esta Tesis y el patrimonio intelectual de la Tesis de Grado pertenece a la Escuela Superior Politécnica de Chimborazo.

Estalin Javier Saltos Taipe

DEDICATORIA

A Dios por ser él mi principal guía durante toda la carrera y no permitir que me rinda en los momentos más difíciles, por su infinito amor y bendiciones derramadas sobre mí a lo largo de mi vida académica.

A mis padres Martha y Juan por ser mi ejemplo de superación, por brindarme su confianza y formarme con valores, por su constante apoyo y sacrificio para que pudiera cumplir mi tan anhelada meta. A mis hermanas Estefanía y Nardelia compañeras de vida, por ser quienes me apoyaron y motivaron en todo momento, por ser mis consejeras y por brindarme todo su cariño.

Estalin

AGRADECIMIENTO

A Dios por permitirme gozar de buena salud, por darme inteligencia, sabiduría y paciencia para que con madurez supere los momentos más difíciles de la carrera y culminar satisfactoriamente esta meta planteada.

A mis padres quienes son el pilar fundamental de mi vida, por brindarme su apoyo moral e incondicional y por acompañarme durante mi preparación académica. A mis hermanas quienes con su paciencia y locuras estuvieron siempre apoyándome. A mis familiares que de una u otra manera estuvieron pendiente de mi carrera, por sus buenos deseos y consejos brindados.

A mis amigos y compañeros, especialmente a TELMEL grupo de hermanos politécnicos, por estar siempre en los buenos y malos momentos, por su amistad desinteresada, por haber compartido vivencias inolvidables y por haber alcanzado nuestras metas.

A mis estimados docentes por compartirme sus conocimientos en especial a mi director del trabajo de titulación Ingeniero Vinicio Ramos por su tiempo y asesoramiento durante el desarrollo de este trabajo, y con ello concluir satisfactoriamente el trabajo de titulación.

Estalin

TABLA DE CONTENIDOS

INDICE DE TABLAS

INDICE DE FIGURAS

INDICE DE GRÁFICOS

INDICE DE ABREVIATURAS

ESPOCH	Escuela Superior Politécnica de Chimborazo
FIE	Facultad de Informática y Electrónica
IoT	Internet de las Cosas
TIC	Tecnologías de la información y comunicación
RFID	Identificador por Radiofrecuencia
LAN	Red de área local
WPAN	Redes de área personal inalámbricas
WLAN	Redes inalámbricas de área local
Wi-Fi	Wireless Fidelity (Fidelidad Inalámbrica)
WEP	Wired Equivalent Privacy
WPA	Acceso Inalámbrico Protegido
TKIP	Emporal Key Integrity Protoco
AES	Advanced Encryption Standard
IEEE	Instituto de Ingeniería Eléctrica y Electrónica
IETF	Internet Engineering Task Force (Grupo de Trabajo de Ingeniería de Internet)
DANE	Departamento Administrativo Nacional de Estadística

FHSS	Espectro Expandido por Salto de Frecuencia
DSSS	Espectro Ensanchado por Secuencia Directa
OFDM	Multiplexación por División de Frecuencias Ortogonales
MIMO	Multiple Input Multiple Output
CoAP	Constrained Application Protocol
HTTP	Hypertext Transfer Protocol
TCP/IP	Protocolo de Control de Transmisión/Protocolo de Internet**.**
UDP	User Datagram Protocol
OSI	Open System Interconnection
IBM	International Business Machines
MQTT	Message Queuing Telemetry Transport
M2M	Machine to Machine
XMPP	Protocolo Extensible de Mensajería y Comunicación de Presencia
XML	Lenguaje de Marcado Extensible
FTP	File Transfer Protocol
SMTP	Protocolo para Transferencia Simple de Correo
JMS	Servicio de Mensajes Java

PSTN	Red Telefónica Pública Conmutada
DSL	Línea de Suscriptor Digital
3G	Tercera Generación de Transmisión de Voz y Datos
LTE	Long Term Evolution
IAB	Comité de Arquitectura de Internet
API	Interfaz de Programación de Aplicaciones
TLS	Seguridad de la Capa de Transporte
CSMA/CD	Acceso múltiple con escucha de portadora y detección de colisiones
GPIO	Entrada/Salida de Propósito General
PHP	Preprocesador de Hipertexto
IDE	Integrated Development Environment
SDD	Software Design Description

RESUMEN

El objetivo del trabajo de titulación fue diseñar un prototipo de sistema de parqueo inteligente para el edificio de la Facultad de Informática y Electrónica (FIE) de la Escuela Superior Politécnica de Chimborazo (ESPOCH) utilizando tecnologías basadas en el internet de las cosas. El prototipo del sistema consiste en una red inalámbrica que utiliza las tarjetas de desarrollo Arduino y Nodemcu. Está conformado por dos nodos: el sensor y recolector de datos, que monitorean en tiempo real el estado de las plazas de aparcamiento del parqueadero de la FIE. En el prototipo se utilizaron sensores ultrasónicos para detectar la presencia de vehículos, los cuales enviaran señales de ondas ultrasónicas al Arduino ubicados en el nodo sensor. Se envía los datos hacia el nodo recolector mediante la tecnología bluetooth utilizando la topología maestro-esclavo. La información se transmite al servidor K2S01(FL-US) de forma digital a través de la comunicación inalámbrica Wi-Fi, donde los datos son visualizados gráficamente en tiempo real acerca de la disponibilidad de parqueos, permitiendo a los usuarios reservar una plaza de aparcamiento durante un tiempo estimado en la aplicación web. De las pruebas desarrolladas se evidenció que el sistema implementado admite un error absoluto de +/- 1 cm en las mediciones de distancia y la repetividad de datos no mayor al 1%, además la interconexión con la Plataforma IoT no presenta perdida de paquetes. De las pruebas realizadas se puede concluir que el prototipo implementado se puede convertir en una herramienta de ayuda para mejorar el flujo de tráfico y el tiempo de los docentes y estudiantes de la FIE. Se recomienda incorporar un módulo de control de acceso vehicular para identificar a los usuarios que anticipadamente hicieron uso de la reservación de una plaza de aparcamiento a través de la aplicación web del sistema.

Palabras clave: <TECNOLOGÍA Y CIENCIAS DE LA INGENIERÍA>, <TELECOMUNICACIONES>, <COMUNICACIONES INALÁMBRICAS>, <PARQUEADEROS INTELIGENTES>, <PLAZAS DE APARCAMIENTO>, <PLATAFORMA IOT>, <INTERNET DE LAS COSAS (IOT)>, <TRASMISIÓN DE DATOS >.

ABSTRACT

The aim of the degree work was to design a prototype of Smart Car Parking System for the building of the Faculty of Informatics and Electronics (FIE) of Escuela Superior Politecnica de Chimborazo (ESPOCH) using technologies based on the internet of things. The prototype of the system consists of a wireless network that uses the Arduino and Nodemcu development cards. It is formed by two nodes: the sensor and data collector, which monitor in real time the status of parking spaces in the FIE parking lot. Ultrasonic sensors were used in the prototype to detect the presence of vehicles, which will send ultrasonic wave signals to the Arduino located in the sensor node. The data is sent to the collector node through bluetooth technology using the master-slave topology. The information is transmitted to the server K2S01 (FL-US) digitally through Wi-Fi wireless communication, where the data are graphically displayed in real time about the availability of parking lots, allowing users to reserve a parking space for an estimated time in the web application. Of the tests carried out it was evidenced that the implemented system admits an absolute error of +/- 1 cm in the distance measurements and the data repeatability not greater than 1%, in addition the interconnection with the IoT Platform does not present packet loss. Of the tests carried out one may conclude that the prototype implemented can become in a support tool to improve the traffic flow and time of teachers and students of the FIE. It is recommended to incorporate a vehicular access control module to identify the users that previously made use of the reservation of a parking space through the web application of the system.

Key Words: <TECHNOLOGY AND ENGINEERING SCIENCES> <TELECOMMUNICATIONS>, <WIRELESS COMMUNICATIONS >, <SMART CAR PARKING >, <PARKING SPACE>, <IOT PLATFORM >, <INTERNET OF THINGS (IOT)>, < DATA TRANSMISSION >.

INTRODUCCIÓN

ANTECEDENTES

En la actualidad el parque automotor del Ecuador ha tenido un crecimiento muy rápido, generando la falta de espacios donde se pueda estacionar los vehículos sin ningún inconveniente. Razón por lo cual el sector público y privado se ven en la necesidad de mejorar y generar parqueaderos en la ciudad.

En el año 2015 el Instituto Nacional de Estadísticas y Censos (INEC) informo los últimos datos del anuario que se matricularon 1’925.368 vehículos a motor en el Ecuador, incrementándose el parque automotor en un 57% de lo que se registró en el año 2010 con una cifra de 1’226.349 vehículos matriculados. Teniendo en consideración que en la provincia de Pichincha se realizaron la mayor cantidad de vehículos matriculados con 492.568, seguido Guayas con 362.857 y Manabí con 152.231. La empresa Chevrolet fue la marca que lidero en las principales provincias con 554.042 vehículos matriculados.

Según un reporte realizado por el diario El Universo sobre el cobro de uso del espacio en el parqueadero del Mall del Sol, se debe a que según estudios realizados existen personas que dejan sus vehículos aparcados por largas horas sin ser clientes, es decir que se está haciendo el uso indebido de los espacios que tiene el parqueadero, ocasionando la falta de disponibilidad al momento de parquear los clientes sus vehículos.

La escasez de plazas de aparcamientos disponibles dentro de parqueaderos de un Mall produce mayor congestión de vehículos, molestia a los conductores por la pérdida de tiempo, además causando afectaciones ambientales, consumo de combustible y aumentando la emisión de diversos gases contaminantes.

En diferentes parqueaderos de los centros comerciales existen altas demandas en las entradas de los estacionamientos por la espera que deben tener los conductores para ingresar y que después de un tiempo no puedan acceder al parqueadero porque ya no cuenta con plazas de aparcamiento libres.

Este proyecto muestra una propuesta al problema desarrollando una base de datos en un servidor de todas las plazas de aparcamiento que tiene el parqueadero de un Mall mediante la creación de una

aplicación en un Smartphone que permita consultar y reservar un espacio para estacionar sus vehículos sin ningún inconveniente.

FORMULACIÓN DEL PROBLEMA

¿Cómo diseñar un prototipo de sistema de parqueo inteligente para el edificio de la FIE utilizando tecnologías basado en el internet de las cosas?

SISTEMATIZACIÓN DEL PROBLEMA

- ¿Cuál es el estudio del arte para el sistema de parqueo inteligente basado en internet de las cosas?
- ¿Qué topología de red de comunicación es adecuada para el sistema de parqueo inteligente que mejor se adapte a los requisitos planteados?
- ¿Cuáles son los requerimientos (hardware y software) que debe cumplir el prototipo del sistema de parqueo inteligente?
- ¿El sistema de parqueo inteligente implementado cumple con las exigencias planteadas al inicio de la investigación?

JUSTIFICACIÓN TEÓRICA

Entre las investigaciones realizadas en el país se encuentra:

- En la Universidad Politécnica Salesiana del Ecuador, que se trata de la implementación de un parqueo inteligente, el cual está basado en la utilización de sensores ultrasónicos que servirán para detectar objetos, en nuestro caso vehículos medianos los cuales enviaran señales a nuestra tarjeta receptora de datos y esta a su vez transmitir datos de manera digital a nuestro Arduino, que con la ayuda de la plataforma Tembo la cual va a ser la encargada de administrar y enviar la información por medio de internet, para poder visualizar la disponibilidad de parqueos disponibles mediante una cuenta Twitter.

- En la Universidad de Guayaquil, que se trata de una plataforma de estacionamiento inteligente con sistema de información en tiempo real usando aplicación móvil para Shopping Center de Quevedo, el propósito es analizar la factibilidad de aplicación de un sistema de parqueadero

inteligente con redes de sensores inalámbricos para reducir el tiempo de búsqueda en un lugar de parqueo y ahorro de recursos.

Aumentar la eficiencia del estacionamiento vehicular en un Mall a través de una señalización acertada que permita al cliente ubicar en el menor tiempo posible los espacios disponibles, mejorando el tráfico en el interior del parqueadero y reduciendo la permanencia de los conductores dentro de su vehículo, impactando directamente en la disminución de las emisiones de gases contaminantes. Además, disminuye el estrés a los conductores a través del ahorro en tiempos de entrada y salida, al mismo tiempo que se incrementa la percepción de innovación y adopción de nuevas tecnologías por parte de los Malls, especialmente para usuarios adaptados a realizar un gran número de actividades desde sus Smartphones, Tablets.

En el Mall de los Andes de la ciudad de Ambato no existe un sistema de parqueo inteligente que ayude a mejorar el tránsito vehicular permitiendo aparcar sus vehículos rápidamente. Por tal motivo el proyecto busca desarrollar un sistema de parqueo que verifique, almacene e informe en tiempo real la disponibilidad o no de lugares de aparcamiento, mediante una aplicación que permita a los usuarios realizar la reservación de la misma, para posterior dirigirse al espacio de forma rápida y eficaz a través de señalética.

JUSTIFICACIÓN APLICATIVA

En la presente investigación se busca realizar un prototipo para el sistema de parqueo inteligente en un Mall, mediante una red de sensores que será ubicada en tres áreas diferentes dentro del parqueadero para conocer si dicho lugar se encuentra disponible, ocupado o reservado, toda esta información que proporcionen dichos sensores será enviada a un equipo electrónico el cual se encargara del procesamiento de la misma para luego ser almacenada en una base de datos, que a través de una aplicación web desarrollada permita la visualización de modo gráfico y sencillo para los clientes sobre el estado actual del parqueadero.

La aplicación web mostrará a los clientes cuantos espacios hay disponibles en el parqueadero del Mall para realizar la reservación, misma que enviara una señal al sensor para que por medio de un dispositivo led se indique que el aparcamiento no se encuentra disponible el cual tendrá un tiempo estimado para que el conductor llegue al lugar.

OBJETIVOS

OBJETIVO GENERAL

Diseñar un prototipo de sistema de parqueo inteligente para el edificio de la FIE utilizando tecnologías basado en el internet de las cosas.

OBJETIVOS ESPECÍFICOS

- Realizar el estudio del arte para el sistema de parqueo inteligente basado en internet de las cosas.
- Establecer la arquitectura de comunicación del sistema de parqueo inteligente.
- Determinar los requerimientos (hardware y software) que debe cumplir el prototipo del sistema de parqueo inteligente.
- Evaluar si el sistema de parqueo inteligente implementado cumple con los requisitos planteados.

CAPÍTULO I

1. MARCO TEÓRICO

Este capítulo presenta lo que es un sistema de parqueo inteligente y sus beneficios, además se describe el concepto de Internet de las cosas, sus protocolos, las tecnologías, la comunicación, plataformas y aplicaciones de las mismas, se define lo que es una tarjeta de desarrollo, las características de algunas y los tipos de topologías de red.

1.1. Sistemas de Parqueo Inteligente

Un sistema de parqueo inteligente ayuda a conductores encontrar una plaza de aparcamiento de manera eficiente y con mayor rapidez mediante el uso de tecnologías de comunicación e información (TIC). Según cómo ha evolucionado la tecnología, en la actualidad es fácil adquirir un smartphone el cual le permita al conductor tener acceso a internet y así encontrar numerosos parqueaderos que se encuentren disponibles y cercanos al lugar que se requiera. El mayor inconveniente que existe en las grandes ciudades es hallar una plaza de aparcamiento disponible para estacionar un vehículo. Durante el recorrido hasta encontrar un parqueadero con plazas desocupadas se presentan diversos factores como: el tiempo perdido, la contaminación ambiental, el consumo de gasolina, el tráfico vehicular, etc. Es por ello que nace el Smart Parking o estacionamientos inteligentes, como solución al problema de circulación vehicular (Godoy, 2015).

La implementación de sistemas de parqueo inteligente consiste en reducir la congestión vehicular que cada vez es mayor. De acuerdo con diversos investigadores y ciudades, se ha realizado el despliegue de múltiples servicios considerando las diferentes circunstancias que presentan. Uno de los servicios de parqueo inteligente es brindar información de los lugares de estacionamientos e información del flujo de tráfico vehicular por donde el conductor este transitando en busca de un parqueadero disponible. Este servicio permite que los conductores reciban la información necesaria acerca de la disponibilidad de estacionamiento, posterior dirigirse al área deseada y finalmente parquear su vehículo (Rosales, 2016, p. 14-15).

Con el avance de la tecnología existen muchas formas de obtener información acerca del estado en que se encuentran las plazas de aparcamiento, lo más óptimo para detectar la presencia de vehículos

es automáticamente. Para ello se presentan diferentes tecnologías tales como: procesamiento de imágenes, RFID, parquímetros, sensores ultrasónicos, sensores magnéticos, etc. Por otra parte, la información adquirida por parte de las tecnologías se puede presenciar a través de plataformas IoT, aplicaciones web, aplicaciones para Smartphone, pantallas digitales en lugares apropiados donde exista mayor conflicto de parqueaderos y cualquier otro dispositivo digital o electrónico que permita visualizar la información a los conductores (Boccalari y Gonzalez, 2016).

1.1.1. Beneficios de un sistema de parqueo inteligente

Un sistema de parquero inteligente presenta diversos beneficios no solo a conductores, si no a propietarios de estacionamientos, al medio ambiente, y el público en general. A continuación, se presenta tres beneficios más significativos.

Como primer beneficio es la reducción del tiempo de los conductores en búsqueda de plazas disponibles en un lugar determinado. Al no existir un sistema de parqueo inteligente el tiempo se prolonga mayoritariamente, debido que el conductor tendrá que transportar el vehículo alrededor de su destino con la esperanza de hallar un estacionamiento libre. Un sistema de parqueo inteligente no solo proporciona información acerca del parqueadero, si no, que permite hacer una reservación anticipada con un tiempo límite para que el conductor llegue al estacionamiento. Además, evitar accidentes de tránsito por distracciones en búsqueda de plazas. (Rosales, 2016).

Por otra parte, un sistema de parqueo inteligente contribuye con el medio ambiente, debido que conductores manejaran directamente hacia el estacionamiento disponible y no estar buscando alrededor plazas de aparcamiento lo cual incrementa la congestión de tráfico y la contaminación ambiental. Así mismo, se reducirá el ruido en las vías y se tendrá una mejor seguridad. (Rosales, 2016)

Además, un beneficio del sistema de parqueo inteligente es para los propietarios de los estacionamientos, porque mediante un registro que es almacenado en una base de datos se puede determinar un análisis del estacionamiento, por ejemplo, la media de la ocupación de cada aparcamiento, el tiempo que tienen más disponibilidad de plazas, etc. Estos datos pueden ser utilizados por parte de los propietarios para beneficio propio de la empresa (Rosales, 2016).

1.1.2. Aparcamiento inteligente

Un aparcamiento inteligente es considerado como un servicio del Smart city en el desarrollo del Internet de las Cosas, el cual utiliza herramientas de la tecnología de información y comunicación (TIC). Su principal funcionalidad es comunicar en tiempo real a los ciudadanos acerca de las plazas de estacionamientos que se encuentren libres para luego estacionarse sin problemas en el menor tiempo estimado. Además, reducir el consumo de combustible, la emisión de gases y primordialmente la congestión vehicular. En la actualidad en las ciudades modernas existen múltiples plazas de aparcamientos inteligentes que detectan los espacios libres e informan a través de paneles a los conductores (Valentín, 2016, p. 50).

Hasta la actualidad se han elaborado diversas aplicaciones que dan soluciones a un aparcamiento inteligente permitiendo al usuario conocer su disponibilidad. Para ello se detalla en la tabla 1-1 los diferentes tipos de aparcamientos.

Tabla 1-1: Diversos tipos de aplicaciones de aparcamientos

APARCAMIENTOS	DESCRIPCIÓN
Robotizado	Es un sistema de aparcamiento donde se monta el vehículo en una base para luego ser colocada en una infraestructura específica a través de máquinas robotizadas sin necesidad del conductor debido que todo se efectúa de manera automatizada.
Colaborativo	Consiste de una comunidad de usuarios que informan acerca del estado actual de un parqueadero, cuando es liberado u ocupado, dicha información será visualizada en teléfonos inteligentes para que los clientes puedan observar la disponibilidad del parqueadero para luego hacer uso del mismo con respecto a su ubicación.
Sensorizado	Es un sistema cuyos datos se obtiene a través de sensores específicos que detectan la presencia de un objeto y se pueden instalar en el plano horizontal y vertical. La información se presentará en paneles o a través de aplicaciones web.
Video vigilado	Es un sistema que se aplica generalmente en lugares amplios con dispositivos como son las cámaras que además de detectar el estado de ocupación de las zonas de parqueo, graban todo lo que sucede de acuerdo a su limitado espacio, brindando seguridad a sus clientes y vehículos en todo el estacionamiento

Fuente: (Valentín, 2016, p. 21)

Realizado por: SALTOS, Estalin, 2018

1.1.3. Plazas de Estacionamiento reservadas

Teniendo en consideración la definición de parqueaderos inteligentes aplicado al IoT y la presencia de plazas de aparcamiento reservados exclusivamente para usuarios como, por ejemplo, autoridades, personas discapacitadas, embarazadas y de tercera edad, etc. Es necesario realizar gestiones de dichas plazas a través de tecnologías que detecten el estado de las plazas automáticamente. Que permita identificar a los vehículos autorizados para hacer uso de los estacionamientos prioritarios. Así mismo, que en horas establecidas donde gran parte de estos usuarios no hacen uso de las plazas, habilitar a cualquier otro conductor para así incrementar la disponibilidad de los parqueaderos en la ciudad (Boccalari y Gonzalez, 2016).

1.2. Internet de las cosas

Internet ha tenido un crecimiento muy vertiginoso con respecto a su tecnología, debido a que en sus principios se podía compartir solamente información, posterior a esto y con el transcurrir de los años se ha desarrollado múltiples servicios que permiten a los usuarios hacer uso de sus requerimientos como pueden ser prestar el servicio de streaming de voz, transiciones bancarias, etc. (Coronel y Tenelanda, 2016).

Internet de las cosas denominado IoT por sus siglas en inglés (internet of things), se define como una infraestructura de red de trascendencia mundial que enlaza los objetos físicos y los objetos virtuales a través de la inter-operatividad de la TIC, como se muestra en la Figura 1-1, para la obtención de datos y estos ser procesados mediante la comunicación a internet (Casagras, s.f).

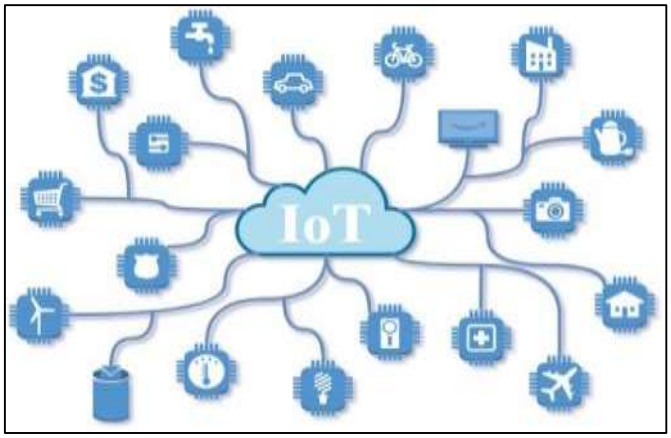


Figura 1-1: Internet de las Cosas

Realizado por: SALTOS, Estalin,2018

Las principales características de IoT consiste en: la capacidad de identificación, la obtención de información, el procesamiento y la comunicación de datos, es por aquello que IoT se encarga completamente del manejo de los objetos con la finalidad de brindar servicios a cada una de las aplicaciones, teniendo en consideración el acatamiento de las exigencias de privacidad y seguridad que se debe tener (ITU, 2012, p. 8)

1.2.1. Tecnologías IoT

1.2.1.1. Bluetooth

Tecnología de redes inalámbricas utilizada en un entorno cuya distancia de conectividad es menor en comparación a otras tecnologías. Es por lo que Bluetooth permite únicamente la comunicación entre un ordenador o dispositivo con sus respectivos periféricos. La primordial característica consiste en la comunicación, misma que se realiza a través del modelo maestro-esclavo o también denominado piconet. Un maestro tiene la capacidad de comunicarse hasta con siete esclavos. De tal forma que, el maestro puede interrumpir la comunicación que tenga con cualesquiera de sus esclavos a través del parking y posterior a esta activar la comunicación a otro esclavo. La técnica que se utiliza en esta tecnología es FHSS (Espectro expandido por salto de frecuencia), el cual se opera en la banda de frecuencia de 2.4 GHz con un intervalo de distancia máxima de 10 m. Además, se puede comunicar con otros dispositivos a una velocidad máxima de 721 Kbps como única dirección y 57.6 Kbps en dirección opuesta, es decir es un canal asíncrono y para canales síncronos es de 432.6 Kbps en ambas direcciones considerando que es simétrico (Gómez y Etc, 2016, p. 20).

A través de la tecnología Bluetooth se logra comunicar entre diferentes dispositivos, para así facilitar su comunicación disminuyendo el uso de cables, espacio, etc. En la figura 2-1 muestra la comunicación que puede existir con dicha tecnología tales como teléfonos móviles, periféricos de entrada y salida, equipos de sonido, entre otros.

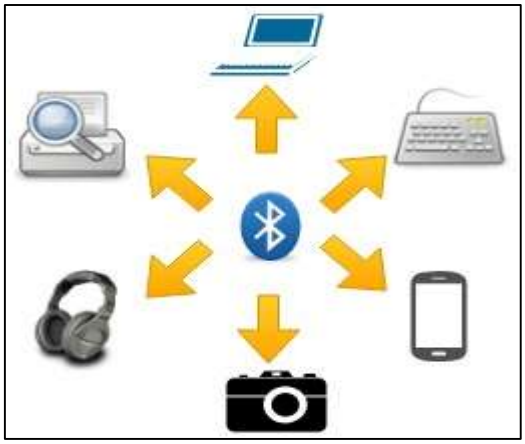


Figura 2-1: Comunicación de Bluetooth

Realizado por: SALTOS, Estalin,2018

1.2.1.2. Wi-fi

Es una tecnología inalámbrica de redes de área local (LAN) basado en el estándar IEEE 802.11 que opera en las bandas de frecuencias de 2.4 GHz y 5GHz, el rango de distancia utilizando medios convencionales en espacios abiertos es hasta 100 metros y en edificios hasta 20 metros. La principal característica de la tecnología Wi-Fi consiste en el intercambio de datos de manera inalámbrica (Bliznakoff, 2014, p. 24).

Esta tecnología contiene distintos mecanismos de cifrado de datos que permite brindar mayor seguridad a la información que se transporta en el medio. Para ello, se han desarrollado dos tipos de cifrado, WEP y WPA siendo esta la versión mejorada de su antecesora. WPA utiliza un servidor de autenticación que le permiten generar una variedad de claves y certificados, además presenta el protocolo TKIP para encriptar datos. Asimismo, se ha desarrollado WPA 2, versión actual que fue modificada de WPA proponiendo mayor velocidad de cifrado de datos y el protocolo de encriptación AES. Todas las mejoras se han realizado con la finalidad de evitar ataques que puedan revelar claves (Rodas, 2013, p. 10).

La Tabla 2-1 muestra las principales características de los protocolos de la tecnología IEEE 802.11 de acuerdo con la evolución que se ha desarrollado hasta la actualidad.

Tabla 2-1: Protocolos para el estándar de la comunicación WI-FI

Protocolo	Año	Frecuencia	Velocidad de datos	Modulación
802.11	1997	2,4 GHz	1-2 Mbps	FHSS/DSSS
802.11 b	1999	2,4 GHz	1-11 Mbps	DSSS
802.11 a	1999	5 GHz	6-54 Mbps	OFDM
802.11 g	2003	2,4 GHz	6-54 Mbps	OFDM
802.11 n	2008	2,5 GHz 5 GHz	450 Mbps	OFDM
802.11 ac w1	2014	5 GHz	867 Mbps	256 QAM
802.11 ac w2	2016	5 GHz	1,73 Gbps	MU-MIMO

Fuente: (INTEL, 2017, p. 1)

Realizado por: SALTOS, Estalin, 2018

1.2.1.3. Zigbee

Es una tecnología de comunicaciones inalámbricas diseñado para realizar redes de control y para usar sensores con bajo consumo. Zigbee permite suministrar conectividad de bajo costo, escalable y baja potencia debido que existen dispositivos que no necesitan mayor tasa de transferencia de datos. Es un conjunto determinado para redes inalámbricas de área personal (WPAN) basado en el estándar IEEE 802.15.4, la velocidad de transmisión máxima de datos es 250 Kbps y su banda de operación de frecuencia es de 868 MHz, 915 MHz y 2.4 GHz (Rao y Etc, 2015).

Figura 3-1: Aplicaciones de la tecnología Zigbee

Realizado por: SALTOS, Estalin,2018

Dentro de la tecnología Zigbee se han desarrollado múltiples aplicaciones tales como: automatización en el hogar, automatización industrial, automatización en hospitales, monitoreo ambiental, control de acceso en hoteles, entre otros. cómo se observa en la figura 3-1.

1.2.1.4. Diferencias entre las tecnologías Bluetooth, Wi-Fi y Zigbee.

La tabla 3-1, indica que la tecnología Wi-Fi y Bluetooth tiene una cantidad moderada de nodos maestro en la red, un alcance promedio a su funcionalidad, además entre las ventajas con respecto a las otras tecnologías esta la velocidad que tiene para transmitir datos, acorde a estos parámetros se optó a utilizar la tecnología Wi-Fi para la comunicación con la plataforma IoT y Bluetooth para la comunicación entre las tarjetas de desarrollo.

Tabla 3-1: Diferencias entre las tecnologías Bluetooth, Wi-Fi y Zigbee

Parámetros	**Bluetooth**	**Wi-Fi**	**Zigbee**
Ancho de banda	1 Mbps	54 Mbps	250 Kbps
Distancia	10 m	100 m	1000 m
Modulación	FHSS	DSSS	DSSS
Consumo de energía	Transmisión 40 mA Reposo 0.2 mA	Transmisión 400 mA Reposo 20 mA	Transmisión 30 mA Reposo 3uA
Cantidad de nodos maestro	7	32	64000
Aplicaciones	Solución de cables, WPAN y móviles	Web, WLAN, datos y e-mail	Control y monitorización
Ventajas	Interoperabilidad, Sustitución de cable	Flexibilidad y velocidad	Bajo costo y consumo, fiabilidad

Fuente:(Rodas, 2013, p. 13)

Realizado por: SALTOS, Estalin, 2018

1.2.2. Protocolos IoT

1.2.2.1. MQTT

MQTT es un protocolo de comunicación IoT que se ejecuta sobre el protocolo TCP/IP. Fue inventado en el año 1999 por el Dr. Andy Stanford-Clark desarrollador de IBM y Arlen Nipper desarrollador de Eurotech. La principal funcionalidad del protocolo MQTT es trasportar mensajes basado en el modelo publish/subscribe, además fue diseñado específicamente para dispositivos que contienen entornos con recursos limitados, restricciones, elevada latencia, poca confiabilidad y limitado ancho de banda. Así

mismo, MQTT permite la comunicación maquina a máquina (M2M) y el uso ideal en aplicaciones móviles debido que posee mejores recursos como es un mejor ancho de banda y mínimo consumo de batería (Mqtt, 2018).

En el modelo publish/subscribe se determinan dos tipos de entidades en la red, tales como: un nodo central mismo que se encarga de realizar la función de un servidor o bróker y un número de clientes. Un bróker es aquel donde se gestiona la red y se retransmiten mensajes, es decir, hace de intermediario entre los clientes de manera que al servidor le envíen todos los mensajes para luego direccionarlos a cada cliente destino que le pertenece dicho recado. En cambio, un cliente es aquel que emite datos recolectados ya sea por sensores o a través de una aplicación de procesamiento de datos que permite publicar en el bróker un mensaje con un tema y por otra parte un mismo cliente puede suscribirse en varios temas. En la figura 4-1 se muestra el modelo publish/subscribe del protocolo MQTT (Yuan, 2017).

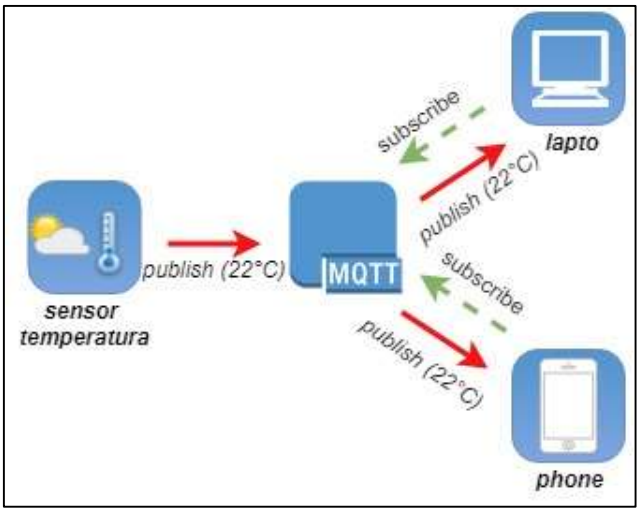


Figura 4-1: Modelo Publish/Subscribe

Realizado por: SALTOS, Estalin,2018

1.2.2.2. CoAP

El protocolo de aplicación restringida (CoAP) fue desarrollado por un grupo de trabajo de ingeniería (IETF). Es un protocolo de comunicación IoT y se ejecuta en la capa de aplicación que corresponde al modelo OSI, además se halla sobre el protocolo UDP debido que TCP genera mayor carga de tráfico. CoAP se basa fundamentalmente en el modelo request/response para hacer su utilización más ligera presentando poco requerimiento en su implementación para ayudar a mejorar los dispositivos que abarcan recursos limitados (Moreno, 2018, p. 19).

Para el diseño de CoAP los desarrolladores tomaron como referencia el protocolo HTTP debido que es el más utilizado a nivel de comunicación. Es por ello, que en su mayoría utiliza semejantes métodos de HTTP como es GET, PUT, POST y DELETE. Así mismo, el protocolo CoAP puede trasladar diversos tipos de carga útiles e identificar el tipo de carga útil que se esté utilizando en el instante. Por otra parte, la principal característica de CoAP es el soporte de multicast que permite tener varios dispositivos conectados para el desarrollo de IoT. En la figura 5-1 se muestra el entorno del protocolo CoAP (Moreno, 2018, p. 20).

Figura 5-1: Entorno protocolo CoAP

Realizado por: SALTOS, Estalin,2018

1.2.2.3. XMPP

El protocolo extensible de mensajería y presencia (XMPP), fue establecido por un grupo de trabajo de ingeniería (IETF) en el año 2002-2003, posterior a un año se estandarizo bajo la RFC 3920-3921. El objetivo del desarrollo de XMPP fue hacer un protocolo base que permita tener una comunicación en tiempo real, primordialmente en aplicaciones que se fundamentan en el estándar XML, esto hace que sea un protocolo extensivo e interoperable en cualquier red (Mendoza, 2008, p. 18).

XMPP trabaja sobre el protocolo TCP de la capa de transporte en el modelo OSI. Además, se basa en la extensión XML beneficiando a ciertos sistemas del IoT. Su primordial característica consiste en la capacidad que tiene para trabajar con dos tipos de modelos, tales como: el modelo request/response y el modelo publish/subscribe. En la figura 6-1 se indica el funcionamiento del protocolo XMPP. (Moreno, 2018, p. 20)

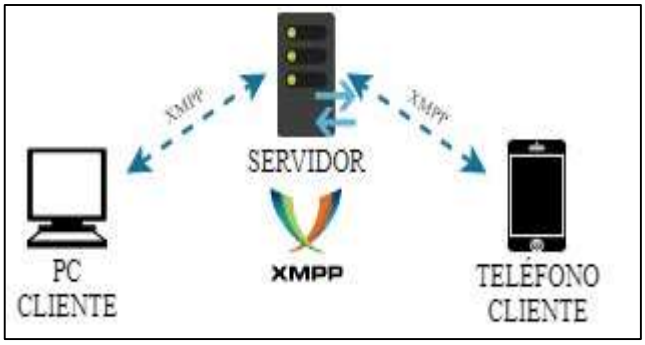


Figura 6-1: Protocolo XMPP

Realizado por: SALTOS, Estalin,2018

1.2.3. Descripción técnica del IoT

En el mundo de la información, un objeto virtual puede existir sin la necesidad de relacionarse con algún objeto físico, en cambio un objeto físico puede ser representado por uno o diversos objetos virtuales. Además, un dispositivo tiene la capacidad para realizar la comunicación entre objetos y la capacidad para detectar, accionar, obtener, almacenar y procesar la información.

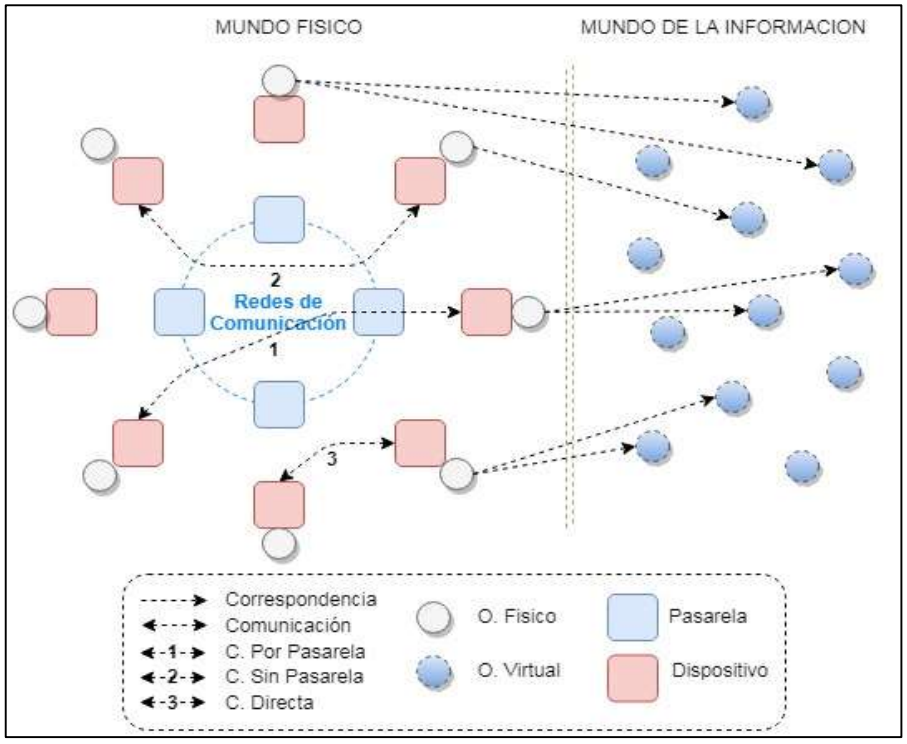


Figura 7-1: Descripción técnica IoT

Realizado por: SALTOS, Estalin,2018

La comunicación entre un dispositivo con otro se puede realizar mediante una red de comunicaciones a través de una red con pasarela (caso 1), por medio de una red sin pasarela (caso 2), o de forma directa es decir fuera del alcance de la red de comunicación (caso 3). Además, se pueden realizar combinaciones entre el caso tres conjuntamente con cualquiera de los otros dos casos (ITU, 2012, p. 10). En la Figura 7-1 se indica la descripción técnica de IoT.

1.2.4. Arquitectura IoT

Debido a diferentes investigaciones que se han realizado, no existe un consenso general acerca de la arquitectura IoT, por lo que se ha propuesto modelos de referencia que abarcan una variedad de detalles a los distintos aspectos de IoT (González, 2017, p. 13). En la Figura 8-1 se indica la arquitectura de IoT.

Figura 8-1: Aplicaciones Zigbee

Realizado por: SALTOS, Estalin, 2018

1.2.4.1. Capa de aplicación

Es donde se pueden extender distintas áreas de aplicaciones realizadas para IoT. Esta capa de aplicación logra ejecutarse a través de los siguientes protocolos de red, tales como HTTP, CoAP y MQTT. De igual manera se pueden utilizar los conocidos protocolos FTP, SMTP y JMS (Rodríguez, 2013, p. 4)

1.2.4.2. Capa de proceso

Se encarga de almacenar, analizar y procesar datos que son adquiridos a través de dispositivos físicos. La recopilación de todos estos datos es muy importante en el sistema de IoT que pueden ser utilizados por distintas aplicaciones. Además, se puede suministrar y gestionar servicios mediante el uso de plataformas en la nube, tales como bases de datos, big data y cloud computing (González, 2017, p. 14)

1.2.4.3. Capa de red

La capa de red se encuentra establecida por dos tipos de niveles:

- **Nivel de red**

Brindan diversas gestiones para el control de la comunicación en la red, como son: gestión de control de acceso íntegro y de recurso apropiado para su transporte, gestión de autorización y autenticación (ITU, 2012, p. 14).

- **Nivel de transporte**

Se encarga de proveer conectividad para transferir correctamente los datos captados por sensores, los datos determinados de servicios y aplicaciones IoT, que se vinculan a través de distintas redes de comunicación.

Por otra parte, en este nivel de transporte se puede elegir los protocolos UDP o TCP. UDP únicamente cuando se utilice el protocolo de aplicación CoAP, mientras que TCP para los demás, principalmente HTTP por personalizar en mayor cantidad a sus aplicaciones web, además se puede emplear en el protocolo MQTT que representa primordial a las comunicaciones de IoT (Rodríguez, 2013, p. 4).

1.2.4.4. Capa de dispositivo

Es la capa inferior jerárquicamente del modelo de arquitectura, compuesta principalmente por las capacidades de dispositivo y de pasarela.

- **Capacidades de dispositivos**

En esta capa, existen una gran variedad de dispositivos que consiguen conectarse al internet, para ello se clasifican en dos tipos de comunicación a la red, la conexión directa y la conexión indirecta.

Conexión directa: son dispositivos que solicitan y cargan toda información en la red de forma indirecta, de igual manera recogen toda información de la red indirectamente. Es decir, en este tipo de conexión a la red no hay necesidad de acudir a capacidades de pasarela.

Conexión indirecta: son dispositivos que solicitan y cargan toda información en la red de forma indirecta, de igual manera recogen toda información de la red indirectamente. Es decir, en este tipo de conexión a la red se realiza la comunicación por medio de capacidades de pasarela.

- **Capacidades de pasarela**

Es aquel que tiene la capacidad de soportar dispositivos enlazados de varias maneras. Como primera instancia en la capa de red puede existir comunicación por medio de diferentes tecnologías, estas pueden ser: PSTN, DSL, 3G, LTE, Ethernet. En cambio, en la capa de dispositivo se encuentran las tecnologías de tipo inalámbricas y alámbricas, tales como Wi-Fi, Bluetooth, ZigBee, RFID o CAN. (ITU, 2012, p. 14)

1.2.5. Modelos de comunicación del IoT

Los modelos de comunicación permiten analizar el comportamiento de los dispositivos conjuntamente con objetos inteligentes que se manejan en IoT, de manera que se logre establecer una comunicación según su requerimiento. Para ello, el Comité de Arquitectura de Internet (IAB) ha establecido cuatro modelos de comunicación para ser utilizados por los dispositivos, los modelos se presentan a continuación:

- Comunicación de dispositivo a dispositivo
- Comunicación de dispositivo a internet
- Comunicación de dispositivo a puerta de enlace
- Intercambio de datos a través de Back-End

1.2.5.1. Comunicación de dispositivo a dispositivo

Es un modelo de comunicación de acuerdo con la figura 9-1, conformado por dos o más dispositivos que se enlazan entre sí y no es necesario la presencia de un intermediario como un servidor. La comunicación entre dispositivos se realiza a través del protocolo de internet IP o igualmente mediante los protocolos ZigBee, Bluetooth o Z-Ware. Este modelo es muy útil en automatización principalmente para sistemas del hogar como habilitar o bloquear cerraduras, encender o apagar bombillos, entre otros (Rose y Etc, 2015, p. 18)

Figura 9-1: Modelo de comunicación dispositivo a dispositivo

Realizado por: SALTOS, Estalin, 2018

1.2.5.2. Comunicación de dispositivo a internet

En este modelo de comunicación todos los datos adquiridos por sensores son cargados directamente a un proveedor de servicios de aplicaciones. Además, la comunicación se efectúa internamente entre dispositivo-nube sin necesidad de interoperabilidad. Las conexiones se pueden realizar a través de la utilización de Wi-Fi o Ethernet, basado en el modelo TCP/IP como se observa en la figura 10-1 (Tschofenig y Etc, 2015, p. 6).

Figura 10-1: Modelo de comunicación dispositivo a internet

Realizado por: SALTOS, Estalin, 2018

1.2.5.3. Comunicación de dispositivo a puerta de enlace

La figura 11-1 indica el modelo de comunicación dispositivo-internet descrito en la sección 1.2.5.2, recomienda tener objetos inteligentes con su propio proveedor de servicios y aplicaciones, el cual funciona con una tecnología de mayor uso como es Wi-Fi, basado en el estándar 802.11. Además, se utilizan tecnologías de radio inferior como 802.15.4 lo que conlleva a introducir una forma distinta de interoperabilidad, cambiando su estructura y función de red (Tschofenig y Etc, 2015, p. 7).

La conexión de los dispositivos a la red se realiza mediante el servicio de aplicación ALG que permite alcanzar un servicio en la nube. Un software de aplicación será ejecutado en el dispositivo que efectúa como puerta de enlace local para establecer la comunicación entre los dispositivos y la nube (Rose y Etc, 2015, p. 15).

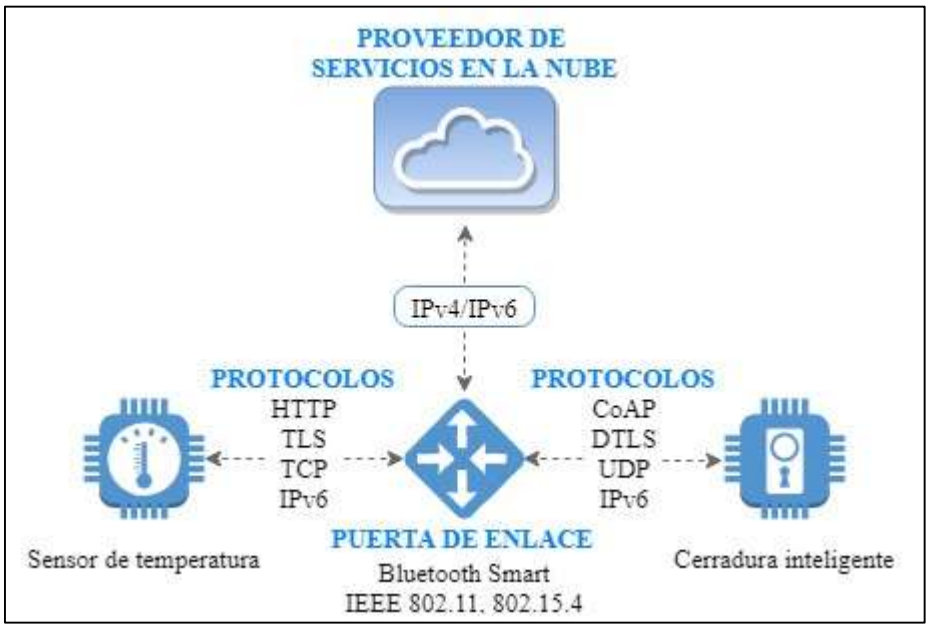


Figura 11-1: Modelo de comunicación dispositivo a puerto de enlace

Realizado por: SALTOS, Estalin, 2018

1.2.5.4. Intercambio de datos a través de Back-End

Dentro de este modelo se analizan todos los datos recopilados por objetos inteligentes, que son cargados a un servicio de aplicación en la nube para realizar una combinación con diferentes datos propuestos por otros servicios. El intercambio de datos a través de Back-End según la figura 12-1,

permite el acceso de terceros a toda la información subida al proveedor de servicios en la nube (Rose y Etc, 2015, p. 23)\

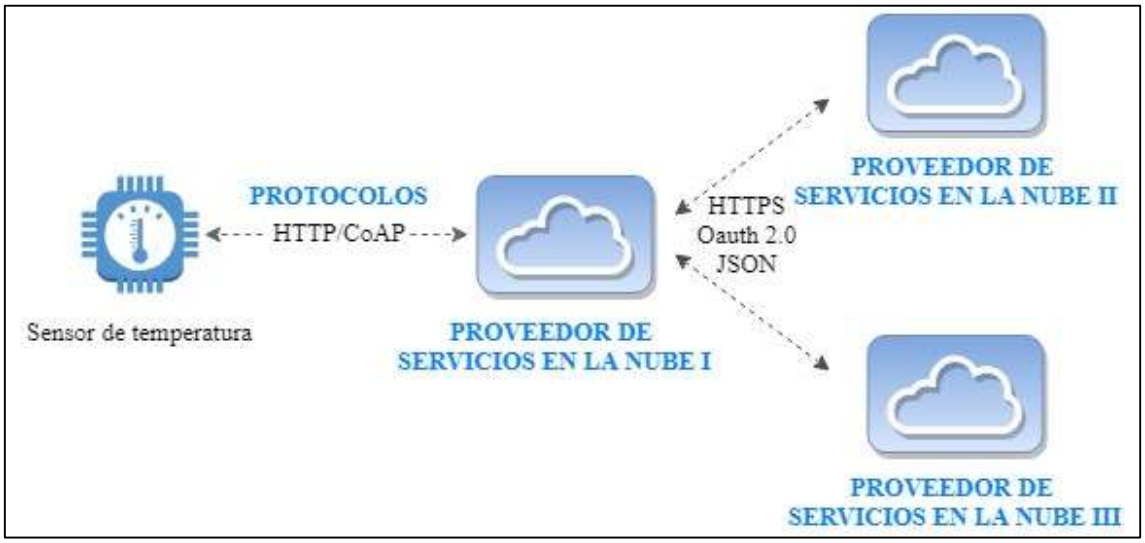


Figura 12-1: Intercambio de datos a través de Back-End

Realizado por: SALTOS, Estalin, 2018

1.2.6. Seguridad de IoT

La seguridad es muy fundamental en IoT debido que todos los objetos y sistemas se encuentran interconectados en la web. Donde, el administrador podrá conocer el estado actual que estos presentan, la información que ha sido recopilada y principalmente verificar su total funcionamiento. En las plataformas IoT se encuentra numerosa información que al ser inseguras se expone a ataques maliciosos realizadas por terceras personas o programas que son creadas para sustraer información y originar daño. Para ello es importante verificar constantemente todo el sistema IoT, proteger aplicaciones y dispositivos, corregir errores que se presentan habitualmente y así evitar ataques que perjudican al desarrollo del sistema.

1.2.7. Plataformas de desarrollo IoT

La principal característica del IoT radica en el procesamiento de la información que es capturada por diversos dispositivos, cada dispositivo deberá estar acoplado a tarjetas de desarrollo con la finalidad de tener conexión a internet donde se almacena toda información dentro de un servidor de aplicaciones web. Para ello existen tipologías de plataformas IoT (Peña y Suquillo, 2016, p. 41).

- Plataformas de desarrollo de software
- Plataformas de desarrollo de hardware

Según los parámetros requeridos para el desarrollo del IoT existen plataformas locales o externas de acuerdo al entorno en el que va a ser utilizado. Conjuntamente, las plataformas ofrecen diversas formas de soporte para aplicaciones de desarrollo web (Ramirez y Rodríguez, 2016, p. 57).

1.2.7.1. Plataforma de desarrollo de software

Las plataformas de desarrollo de software permiten crear servidores de aplicaciones y almacenamiento web donde se pueda almacenar información. Para ello se han desarrollado diferentes tipos de software que brindan un servicio de acuerdo con las necesidades del usuario. Además, a través de la web se puede configurar y controlar los dispositivos que se encargan de recopilar la información de los objetos.

- **Thingspeak**

Es una plataforma web se servicio Open Source y API el cual permite almacenar y redimir datos de objetos que a través de una red LAN utilizan el protocolo HTTP de acuerdo con la figura 13-1. Por otra parte, la plataforma de desarrollo Thingspeak permite la creación de distintas áreas de aplicación de desarrollo IoT, tales como gestionar dispositivos, rastreo de sitios, etc. Thingspeak está basado de un framework en RoR versión 3.0, en el que se puede desarrollar aplicaciones web cuya característica principal consiste en su programación, puesto que utiliza menos código y es más sencilla al realizar aplicaciones de desarrollo web (Loureiro, 2015, p. 5).

Figura 13-1: Plataforma de desarrollo Thingspeak

Realizado por: (Softpowergroup, 2018)

- **Ubidots**

Ubidots es un servicio de aplicación web de código abierto como se indica la figura 14-1. Además, por su evolución en la tecnología maneja Ubidots API, mismo que se encarga de efectuar numerosos dispositivos basado en un código único, donde las variables de Ubidots serán asignados automáticamente a cada uno de los dispositivos. Dentro de la plataforma Ubidots se ha desarrollado diferentes aplicaciones y servicios para el IoT, conjuntamente con herramientas que permiten recopilar, indagar y visualizar información capturada por múltiples dispositivos. Para ello se deberá configurar y desarrollar un servicio en la nube, el cual presenta un inconveniente por su dificultad para realizarlo debido que se debe crear varios parámetros (Ubidots, 2018).

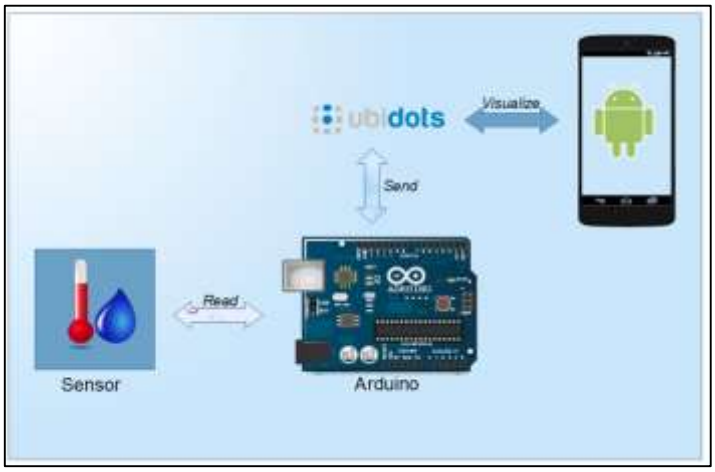


Figura 14-1: Plataforma de desarrollo Tembo

Realizado por:(Pinterest, 2018)

- **Tembo**

La plataforma de desarrollo tembo consiste en múltiples API, que permite enlazar tarjetas de desarrollo de hardware al internet. Además, posee una interfaz muy elemental para realizar aplicaciones de IoT, en el que se utilizara pocas líneas de código. Esto hace que sea un lenguaje con estructura estable donde se pueda utilizar diversas técnicas (Flores y Etc, 2016, p. 24-25)

Debido a la escalabilidad que presenta la plataforma temba, su arquitectura es completamente independiente para usar el sistema. De igual manera, para mayor seguridad dentro de la plataforma se encriptará la información a través de AES a datos que se hallan en reposo y TLS a los datos que estén en constante movimiento (Tembo, 2018). La figura 15-1 muestra a esta plataforma.

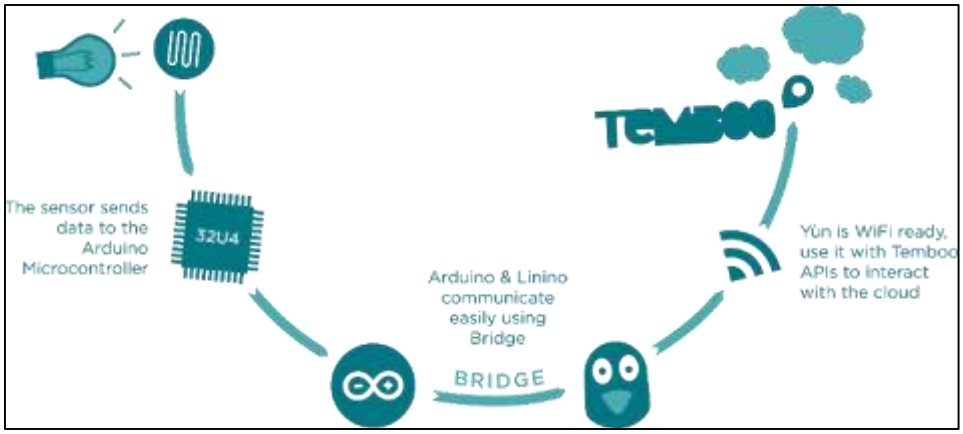


Figura 15-1: Plataforma de desarrollo Tembo

Realizado por: (Arduino Tec, 2010)

1.2.7.2. Plataforma de desarrollo de hardware

Conocidas como placas electrónicas que en su interior comprenden de circuitos impresos y componentes para realizar distintas funciones según como fueron diseñadas por los fabricantes. Estas tarjetas pueden ser compatibles con varios dispositivos de diferentes tecnologías como: sensores, tarjetas microSD, módulos PIC, módulos inalámbricos, etc. Su principal característica consiste en el Software propio que tiene cada tarjeta de desarrollo para realizar todo tipo de programación que se requiera. Además, se debe considerar que en la actualidad existen una gran cantidad de tarjetas de desarrollo, que al pasar el tiempo han ido perfeccionando sus versiones y diseño para realizar así aplicaciones acordes a las necesidades que presentan a los usuarios (Paredes, 2017, p. 22).

- **Raspberry Pi**

Raspberry Pi es un mini procesador diseñado para ser aplicado en distintos proyectos tecnológicos. Compatible con todas sus versiones antiguas aplicadas a sistemas operativos de Linux, Ubuntu y Debian, en la actual es adecuada para el sistema operativo de Windows 10 que en su mayor aplicación se realiza para el Internet de las Cosas. Además, su avance tecnológico hace que sea una tarjeta de desarrollo más eficaz y robusta que sus precedentes (Cobos y Ortiz, 2017, p. 15).

En la Figura 16-1 se muestra un ejemplo de la tarjeta de desarrollo de Raspberry Pi 2. En cambio, en la Tabla 4-1 se presentan las características principales que contiene la plataforma.

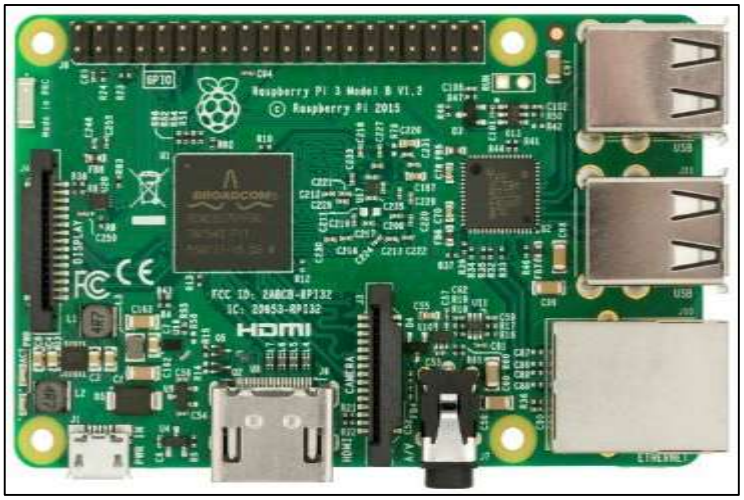


Figura 16-1: Raspberry Pi

Realizado por: (Perles, 2017)

Tabla 4-1: Características de la tarjeta Raspberry Pi 2

Memoria RAM	1 GB
Salida de video	1080p
Socket MicroSD	1
Puertos USB 2.0	4
Pines GPIO	40
Voltaje de operación	5V – 2ª
Dimensiones de placa	86 x 56 x 20 mm

Realizado por: SALTOS, Estalin, 2018

- **Arduino**

Es una placa muy sencilla con un microcontrolador en su interior fabricada por la marca ATMEL, toda esta plataforma se compone de entradas y salidas tanto analógicas como digitales. Además, es una tarjeta de desarrollo con hardware de código abierto que contiene un lenguaje de programación processing. La principal característica de este dispositivo es conectar todos los componentes físicos y hacerlos virtual, o la parte analógica controlarlo de forma digital. Por otra parte, el software de esta plataforma es de licencia libre misma que está diseñada para ser extendida mediante librerías de C++, puede ser modificado a través del lenguaje de programación en el que fue diseñado anteriormente como lo es AVR C, esto lo pueden realizar desarrolladores y programadores con un nivel muy alto de experiencia en estos lenguajes (Tapia y Manzano, 2013, p. 25). En la Figura 17-1 se indica este tipo de tarjeta.

Figura 17-1: Arduino Uno

Realizado por: (Diaz, 2016)

- **Node MCU**

Es una tarjeta de desarrollo similar a la plataforma de Arduino, en el mercado se le puede adquirir a un precio reducido. Su diseño permite realizar aplicaciones tecnológicas enfocado al internet de las cosas, se enlaza al internet a través de Wi-Fi. La principal característica de esta tarjeta consiste en el procesador que contiene una arquitectura de 32 bits haciendo más eficaz potente al momento de trabajar. Por otra parte, se puede programar con el lenguaje interpretado LUA y el lenguaje que se trabaja Arduino (Gusqui, 2017, p. 13-14). En la Figura 18-1 se observa este tipo de tarjeta.

Figura 18-1: Node MCU

Realizado por: (Del Valle, 2018)

- **Comparación de las plataformas de desarrollo**

En la tabla 5-1 se presenta las principales especificaciones de las plataformas de desarrollo de hardware anteriormente descritas. Posterior a un análisis se opta por utilizar las tarjetas Arduino y Node MCU, debido que los dos presentan reducidos costos, compatibilidad, terminales analógicos y digitales, además, velocidad provechosa según los requerimientos del usuario.

Tabla 5-1: Diferencias entre plataformas: Raspberry Pi 2, Arduino y Node MCU

CARACTERISITCAS	RASPBERRY PI	ARDUINO	NODE MCU
Procesador	ARM 11	ATMega 328	Tensilica Xtensa LX3
Memoria RAM	1 GB	2 KB	96 KB
Memoria Flash	Tarjeta SD	2 KB	4 Mb
Velocidad	700 MHz	16 MHz	80 MHz
Voltaje de operación	5 V	5 V	3.6 V
Ethernet	10/100	N/A	N/A
USB	4	N/A	N/A
Sistema operativo	Windows 10, Distribuciones de Linux	N/A	N/A
WLAN	802.11 b/g/n	N/A	802.11 b/g/n
Precio	$85	$18	$10

Realizado por: SALTOS, Estalin, 2018

1.2.8. Áreas de aplicaciones del IoT

En la actualidad existen numerosas áreas de aplicaciones del IoT, comprendido en cada uno de los ámbitos que realizan las personas en la vida cotidiana, es por ello por lo que ha incrementado el desarrollo de la tecnología IoT, conectando objetos a diversos dispositivos electrónicos con el propósito de obtener datos, interpretar toda la información y transmitirla. Entre las principales aplicaciones del Internet de las Cosas se tiene: el hogar, la industria, salud inteligente, energía inteligente, edificios inteligentes, transporte inteligente, ciudades inteligentes, como se indica en la figura 19-1, siendo el campo de ciudades inteligentes donde se desarrolla el presente trabajo de titulación (Toazo y Vega, 2017, p. 7).

Figura 19-1: Áreas de aplicaciones IoT

Realizado por: SALTOS, Estalin, 2018

1.3. Sensores

Los sensores son la parte primordial para el desarrollo del Internet de las Cosas, debido que a través de estos dispositivos se logra obtener información dentro de un entorno donde se encuentran los objetos, para luego ser enviados y procesados por un sistema de control. Los datos recopilados por los sensores son adquiridos en tiempo real, permitiendo el acceso a dicha información de manera remota, verificando su funcionalidad para el que fue diseñado (Luis, 2015, p. 21).

Por otra parte, los sensores deben ser elegidos considerando los parámetros que se quieren analizar, para ellos existen diferentes tipos de dispositivos que permiten obtener datos de los objetos tales como proximidad, temperatura, movimiento, humedad, etc. como se muestra en la figura 20-1. La mayor parte de sensores son conectados a una fuente de energía conjuntamente a una tarjeta de desarrollo donde se realizará el procesamiento de datos (Ramirez y Rodríguez, 2016, p. 47).

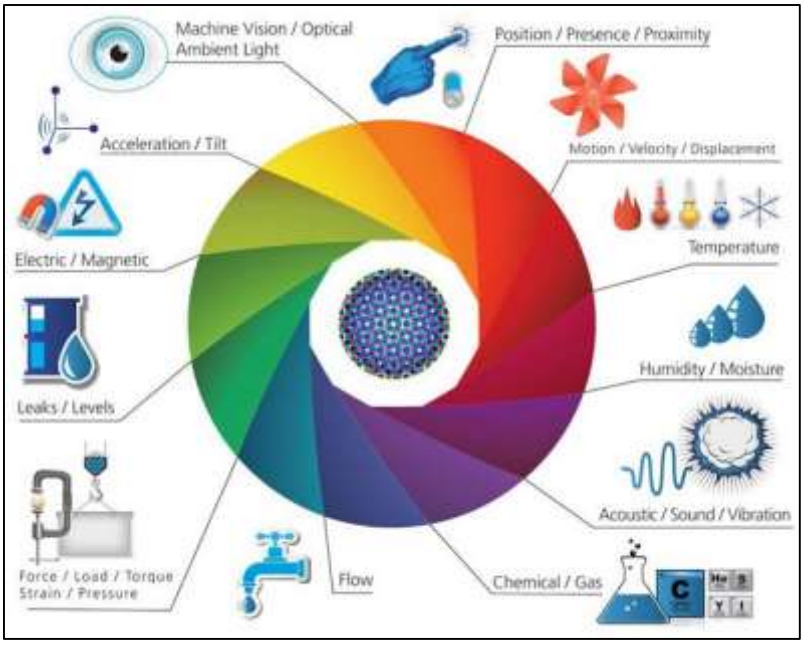


Figura 20-1: Aplicación de sensores para el desarrollo IoT

Realizado por: (López, 2015)

1.4. Smartphone

Se denomina smartphone a los teléfonos móviles inteligentes que permite al usuario realizar múltiples funcionalidades de acuerdo con las características y capacidades que esté presente, a más de las que se efectúan normalmente como es llamadas y mensajerías de texto. Cada smartphone contiene instalado un sistema operativo propio según sus desarrolladores, comúnmente se maneja con los siguientes sistemas operativos: Android, Windows Phone, iOS. Además, en la actualidad se pueden efectuar varias funciones que normalmente se hace en un ordenador fijo o portátil. Es por ello, que un teléfono inteligente es considerado como una gran evolución en la tecnología (WordReference, 2009).

1.5. Topología de red

Topología de red se refiere al funcionamiento y rendimiento de los dispositivos que estén acoplados a una red. Esta se puede componer en dos partes; la conexión física y la conexión lógica. Para seleccionar la topología correcta se debe considerar la cantidad de nodos que integraran la red, la capacidad de expansión, el retardo que tendrá para trasmitir datos y detectar errores en la comunicación. Entre las topologías de red que se utilizan con mayor frecuencia son las siguientes:

1.5.1. Topología estrella

Es una red donde a un nodo o punto central se conectan directamente varias estaciones como se indica en la Figura 21-1. Para que todos puedan comunicarse entre ellos necesariamente el nodo central debe estar activo porque a través de éste se transmitirá toda la información necesaria entre los dispositivos, debido que estos no se encuentran conectados directamente entre sí.

Además, esta topología depende exclusivamente del nodo central el cual siempre debe permanecer activo en la red para que no exista perdida de comunicación y problemas entre los distintos nodos (Castells y Martínez, 1997).

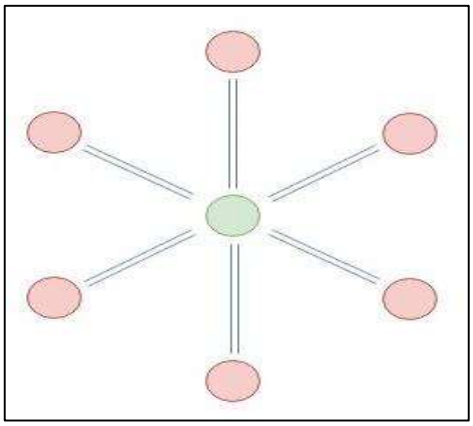


Figura 21-1: Topología estrella

Realizado por: SALTOS, Estalin, 2018

1.5.2. Topología malla

En esta topología cualquier nodo que pertenezca a la red puede transferir a otros nodos siempre que estos se hallen dentro de un rango determinado para la comunicación. Principalmente se encuentra formada por nodos multi-trayecto y de consumo de energía relativamente baja. Los nodos iniciales deben retransmitir los diferentes mensajes que emiten los nodos que contienen bajo consumo de energía con destino a otros nodos que conforman la red.

Además, si se desea enviar un mensaje desde un nodo hacia otro que no se encuentre dentro del rango determinado de comunicación se utilizara un tercer nodo intermedio que le permita retransmitir el mensaje al nodo destino.

Una de las ventajas importantes de esta topología es ser escalable y redundante. En cambio, su primordial desventaja es tener nodos que ocupan un mayor consumo de energía debido a que son multi-trayecto el cual genera un tiempo de vida muy finito de la batería. (Bravo y Belduma, 2017, p. 3). La Figura 22-1 indica la topología de malla.

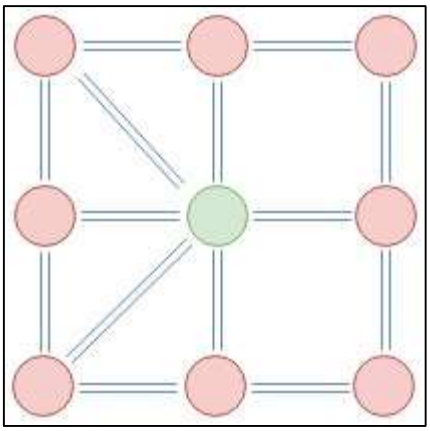


Figura 22-1: Topología malla
Realizado por: SALTOS, Estalin, 2018

1.5.3. Topología bus

Todos los nodos que conforman esta red se colocan de forma lineal interconectados a un medio de comunicación común a través de un cable, a esto se le denomina bus. Los mensajes emitidos por un nodo cualquiera se propagan alrededor de todo el bus, llegando a los demás nodos que son parte de la red. Todos los nodos deberán de reconocer la información que fue emitida por el nodo origen para así establecer a que nodo destino le corresponde el mensaje.

Además, el método de acceso al bus que se usa es el CSMA/CD este es el encargado de la gestión para que los nodos puedan tener acceso y mediante un algoritmo resolver posibles conflictos dentro de esta .la principal ventaja de esta topología es que un fallo cualquiera de un nodo no afecta a la red por lo que los demás nodos podrán trabajar normalmente (Mendoza, 2012, p. 19). El modelo de esta topología se indica en la Figura 23-1.

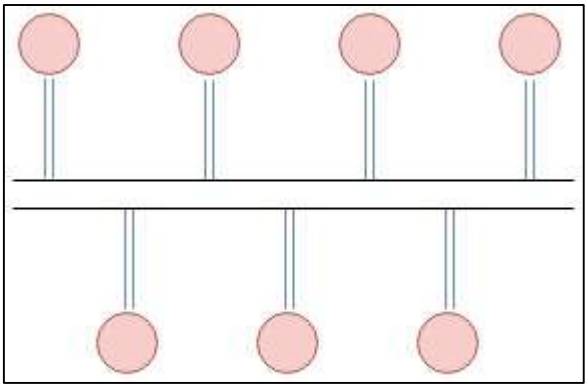


Figura 23-1: Topología Bus

Realizado por: SALTOS, Estalin, 2018

1.5.4. Topología piconet

Son redes que presentan la posibilidad de extenderse hasta un máximo de 8 enlaces punto a punto. Además, existe otra forma para que este tipo de red aumente es a través de scatternets. En este tipo de red un dispositivo actúa como maestro y los demás como esclavos. (Cuevas y Carrillo, s.f).

1.5.5. Topología scatternets

Son redes originadas cuando dos elementos pertenecientes a dos o más redes piconets distintas. Las Scatternets se crean cuando un dispositivo en una red piconet, sea un maestro o esclavo, desea ser el esclavo del maestro de otra red piconet. Es decir que el dispositivo se transforma en el puente entre las dos redes piconets. (Cuevas y Carrillo, s.f).

CAPITULO II

2. DISEÑO DEL PROTOTIPO DEL SISTEMA DE PARQUEO INTELIGENTE

Este capítulo detalla el diseño del prototipo del sistema de parqueo inteligente para el edificio de la FIE, los requerimientos del sistema, la concepción general, los diagramas de bloques de los nodos, la selección de los elementos que componen el hardware. Además, muestra las conexiones de los nodos del cual está compuesto el sistema.

2.1. Requerimientos del sistema de parqueo

Definiendo los conceptos teóricos correspondientes al tema y analizando la encuesta realizada, se establecen los requerimientos del diseño para el prototipo del sistema de parqueo inteligente. A continuación, se indica los requerimientos del sistema.

- El sistema debe estar diseñado para detectar la presencia de vehículos en el interior del parqueadero del edificio de la FIE
- El sistema de parqueo debe ser escalable y de bajo costo.
- La información obtenida para el sistema es en tiempo real, a su vez es almacenada en la red.
- Supervisar toda el área del estacionamiento utilizando sensores y un nodo recolector.
- El sistema debe permitir al usuario reservar el estacionamiento para su vehículo durante un tiempo establecido.

2.1.1. Concepción general del sistema

El sistema de parqueo para el estacionamiento del edificio de la FIE consta de una red de sensores que será ubicada dentro del área del parqueadero para conocer si dicho lugar se encuentra disponible, ocupado o reservado, toda esta información que proporcionen dichos sensores será enviada a un equipo electrónico el cual se encargara del procesamiento de la misma a través de la tecnología Wi-

Fi para ser almacenada en una base de datos, que a través de una aplicación desarrollada en un dispositivo móvil con sistema operativo Android permita la visualización de modo gráfico y sencillo para los clientes sobre el estado actual del parqueadero. La red está compuesta por dos nodos: nodo lector y nodo recolector de acuerdo con la Figura 1-2.

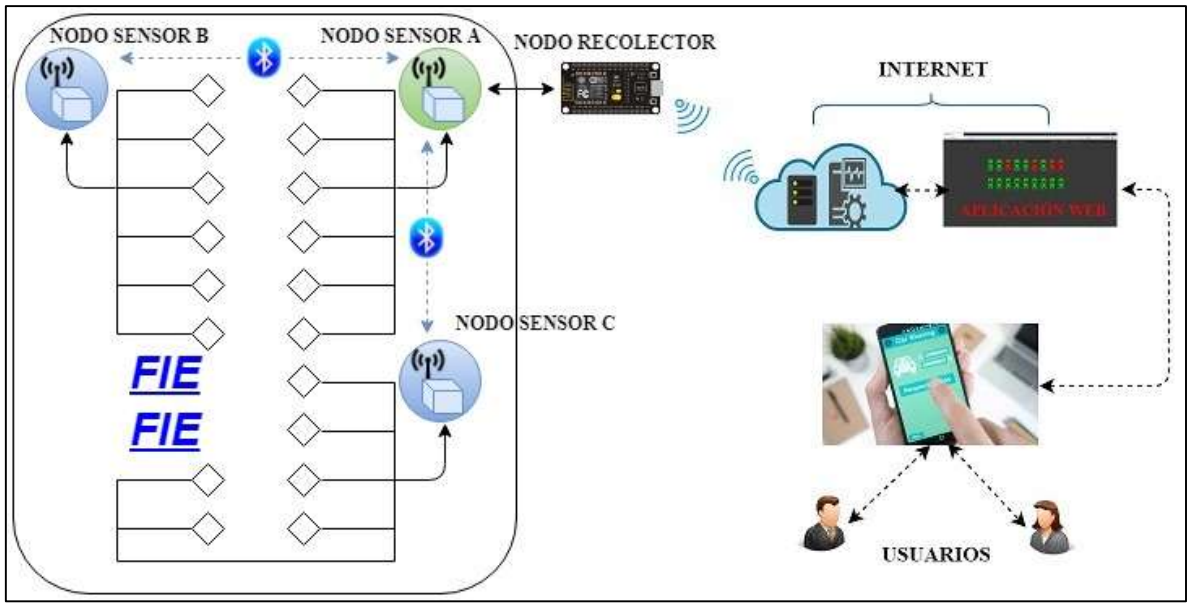


Figura 1-2: Esquema de conexión del sistema

Realizado por: SALTOS, Estalin, 2018

Nodo sensor: Es el encargado de verificar el estado en que se encuentra una plaza de aparcamiento y proveen toda información a un nodo recolector. Así mismo, disponen de sensores de proximidad y una tarjeta de desarrollo. Dichos dispositivos electrónicos obtendrán los datos que se enviarán al nodo recolector, la zona del parqueadero está dividida en tres áreas que abarca seis sensores en cada uno de ellos.

Nodo recolector: En esta sección se realiza el procesamiento de toda información recibida por parte del nodo lector, conformado por una tarjeta de desarrollo mismo que emitirá información a través de la tecnología wifi hacia una base de datos montado sobre una plataforma en la nube. La información se podrá visualizar en una página web o una aplicación web en un dispositivo móvil.

La aplicación móvil mostrara a los clientes cuantos espacios hay disponibles en el estacionamiento del Edificio de la FIE para realizar la reservación, misma que enviara una señal al sensor para que

por medio de un dispositivo led se indique que el aparcamiento no se encuentra disponible el cual tendrá un tiempo estimado para que el conductor llegue al lugar establecido.

2.1.2. Arquitectura del sistema

Se forma por dos diagramas de bloques de cada nodo del sistema, cada uno se muestra a continuación.

2.1.2.1. Diagrama de bloque del nodo lector

El diagrama de bloque del nodo lector que se indica en la Figura 2-2 está compuesto por cuatro bloques, inicialmente el elemento sensor de distancia que se encarga de la adquisición de los datos detectando la presencia o ausencia del objeto, posterior se encuentra el elemento de procesamiento de datos que se abastece por una fuente independiente y finalmente está el bloque de transmisión de datos que usa comunicación serial.

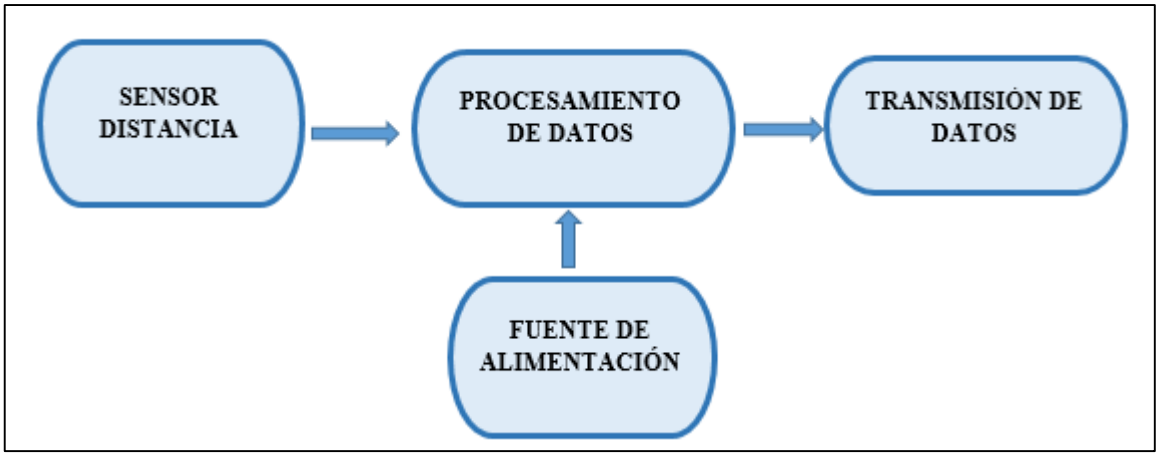


Figura 2-2: Diagrama de bloque del nodo lector.

Realizado por: SALTOS, Estalin, 2018

2.1.2.2. Diagrama de bloque del nodo recolector

El diagrama de bloque del nodo recolector que se indica en la Figura 3-2 está comprendido de cuatro bloques, inicialmente el elemento recepción de datos que su función es receptar la información enviada por el nodo lector para inmediatamente ser procesada; el componente de la fuente de alimentación se encarga de suministrar energía al sistema del nodo recolector; finalmente el bloque de transmisión Wi-Fi, a través de la comunicación wifi es viable enviar los datos recolectados en este

nodo hacia un servidor donde se almacena toda la información para luego ser visualizada en una página web. Antes de acceder a la página web es necesario que el usuario se registre con un correo y una contraseña.

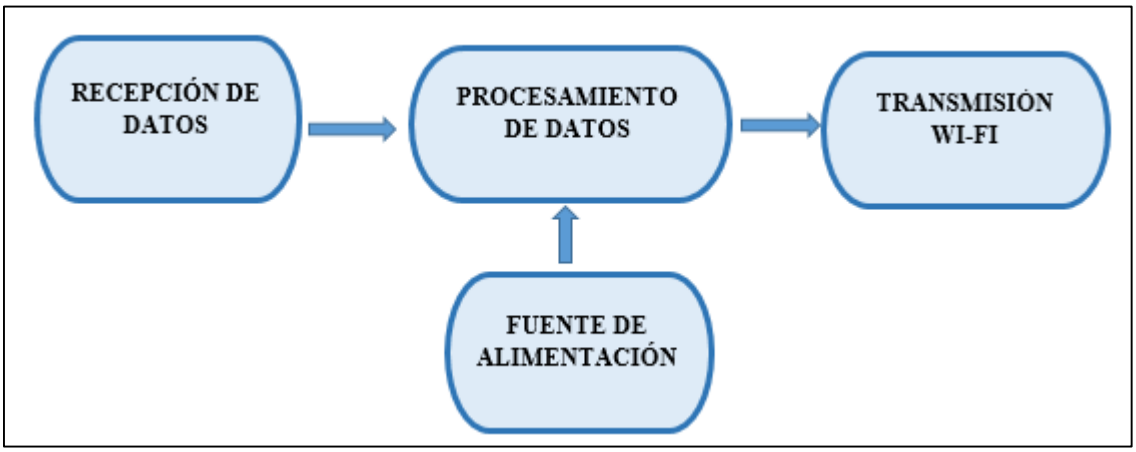


Figura 3-2: Diagrama de bloque del nodo recolector.

Realizado por: SALTOS, Estalin, 2018

2.2. Selección del hardware del sistema de parqueo

Los dispositivos que se utilizaron para el diseño del prototipo del sistema de parqueo se enumeran a continuación.

2.2.1. Tarjetas de desarrollo

Para la selección de las tarjetas de desarrollo se analizaron las características que estén acorde y cumplan con los beneficios para el sistema. Para el diseño del prototipo se utilizó la siguiente plataforma.

- **Arduino**

En la Tabla 1-2, se analizan las diferentes características entre las diversas tarjetas de desarrollo arduino: Uno, Nano, Mega. Se opta por la utilización del Arduino Mega 2560, debido que contiene mayor cantidad de pines digitales y analógicas de entrada y salida, requerimiento que necesita el usuario para el desarrollo del sistema de parqueadero.

Tabla 1-2: Diferencias entre tarjetas de desarrollo Arduino

ESPECIFICACIONES	UNO	NANO	MEGA
Microcontrolador	Atmega328	Atmega168-20UA	Atmega2560
Voltaje de entrada	5-12 V	5-12 V	5-12 V
Voltaje de salida	3,5-5 V	3,3-5 V	3,3-5 V
Frecuencia de reloj	16 MHz	16 MHz	16 MHz
Memoria Flash	32 KB	32 KB	256 KB
Pines I/O Digitales	14	14	54
Pines entradas analógicas	6	8	16
Precio	$15,00	$10,00	$25,00

Realizado por: SALTOS, Estalin, 2018

- **Arduino Mega**

Es un microcontrolador basado en un procesador ATmega2560. La Figura 4-2 indica cada uno de los componentes que se mencionan a continuación: está compuesto por 54 terminales digitales para entradas y salidas, 16 entradas analógicas, 4 puertos seriales, un oscilador de cristal, un conector para la alimentación y una conexión USB, etc. En la tabla 2-2 se presenta las características fundamentales del Arduino Mega.

Figura 4-2: Tarjeta de desarrollo Arduino Mega2560

Realizado por: SALTOS, Estalin, 2018

Tabla 2-2: Características principales del Arduino Mega.

Especificaciones	Características
Procesador	ATmega 2560
Voltaje de operación	5 V
Corriente DC	20 mA
Corriente DC para 3.3V	50 mA
Memoria flash	256 KB
SRAM	8 KB
EEPROM	4 KB
Velocidad de reloj	16 MHz

Realizado por: SALTOS, Estalin, 2018

2.2.2. NodeMCU ESP8266

El NodeMCU, es una plataforma que presta soluciones para conectividad Wi-Fi de acuerdo con la Figura 5-2, primordialmente se utiliza en IoT por su alto nivel de integración, por ser eficiente y por bajo consumo de energía. Posee un procesador de 32 bits. Para la programación de esta tarjeta se puede usar el lenguaje Lua y Arduino. Tiene integrado 17 terminales digitales GPIO y un terminal analógico ADC.

La Tabla 3-2 muestra algunas de las especificaciones técnicas de esta tarjeta de desarrollo.

Tabla 3-2: Características de NodeMCU ESP8266

Parámetros	Características
Voltaje de alimentación	5V
Voltaje de entradas y salidas	3.3V
Procesador	Tensilica Xtensa
Frecuencia de procesador	80 MHz/160MHz
RAM de datos:	96 KB
Memoria Flash Externa	4MB
Protocolos	TCP/IP
Potencia	1mW
UART	Tx y Rx
Estándar	802.11 b/g/n

Realizado por: SALTOS, Estalin, 2018

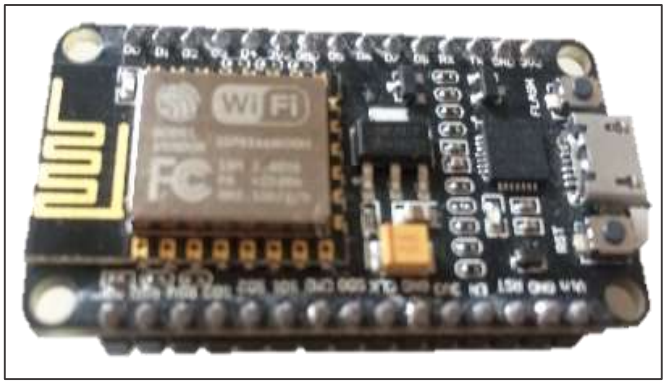


Figura 5-2: NodeMCU ESP8266

Realizado por: SALTOS, Estalin, 2018

2.2.3. Sensor HC-SR04

El sensor ultrasónico HC-SR04 está compuesto por un circuito de control, un transmisor y un receptor ultrasónico como muestra la Figura 6-2. Además, contiene cuatro pines de conexión: VCC, Trig (emite ultrasonido), Echo (recepta ultrasonido) y GND. Su principal funcionalidad es medir distancias a través de ondas ultrasónicas. En la Tabla 3-2 se describen las especificaciones técnicas según (Elecfreaks, 2016).

Tabla 4-2: Especificaciones de sensor HC-SR04

Especificaciones	**Características**
Voltaje de alimentación	5 V
Frecuencia de trabajo	40 KHz
Corriente de trabajo	15 mA
Rango mínimo	2 cm
Rango máximo	4 cm
Angulo de medición	15°
Duración del pulso de entrada (TTL)	10 uS
Duración del pulso eco de salida (TTL)	100-2500 uS

Realizado por: SALTOS, Estalin, 2018

Figura 6-2: Sensor HC-SR04

Realizado por: SALTOS, Estalin, 2018

2.2.4. Modulo Bluetooth HC-05

Es una tarjeta electrónica como indica la Figura 7-2, Permite una comunicación con otros dispositivos, mediante el protocolo bluetooth. Puede trabajar como maestro o esclavo. Para la configuración de este módulo se utiliza comandos AT. La Tabla 5-2 indica las especificaciones de este dispositivo.

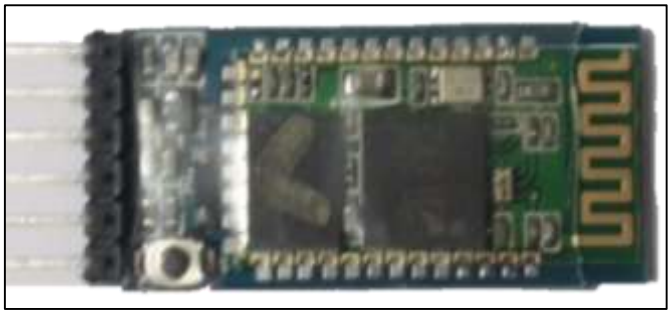


Figura 7-2: Modulo Bluetooth HC-05

Realizado por: SALTOS, Estalin, 2018

Tabla 5-2: Especificaciones del Módulo Bluetooth HC-05

Especificaciones	**Características**
Voltaje de operación	3.3 -5 V
Corriente de operación	50mA
Alcance	10 metros
Frecuencia de trabajo	2.4 Ghz
Velocidad de transmisión	3 Mbps
Modulación	GFSK
Sensibilidad	80 dBm

Realizado por: SALTOS, Estalin, 2018

2.2.5. Fuente de alimentación

Para el funcionamiento del sistema se utilizará dos fuentes de alimentación de 9 voltios como se observa en la Figura 8-2. Las características técnicas se presentan en la Tabla 6-2.

Figura 8-2: Fuente de alimentación

Realizado por: SALTOS, Estalin, 2018

Tabla 6-2: Especificaciones de la fuente de alimentación

Especificaciones	Características
Voltaje de entrada	100-240 VAC
Voltaje de salida	9 V
Corriente de operación	2 A
Frecuencia de entrada	50/60 Hz
Distancia cable	1 m

Realizado por: SALTOS, Estalin, 2018

2.2.6. Esquema de conexión del sistema de parqueadero

En esta sección se detalla la forma en el cual van conectados los elementos y nodos que conforman el sistema de parqueadero.

2.2.6.1. Esquema de conexión del nodo sensor

En la Figura 9-2, se observa el esquema de conexión para el funcionamiento del nodo lector, el diagrama se conforma de tres dispositivos principales: Sensor HC-SR04, Modulo Bluetooth HC-06 y Arduino Mega 2560. La función primordial del nodo sensor es realizado por el Módulo Bluetooth, porque establece la comunicación del sistema. Sus elementos están conectados de la siguiente forma:

- El sensor ultrasónico es suministrado con una fuente de alimentación de 5 voltios.

- Los terminales Trig y Echo del sensor HC-SR04 están conectados con los terminales analógicos 2 y 3 respectivamente del Arduino Mega.

- El módulo Bluetooth se alimenta con 3,3 Voltios y GND que es proporcionado a través del Arduino Mega.

- El arduino Mega 2560 se conecta mediante los terminales 18(TX1) y 19(RX1) con los terminales RX y TX del módulo Bluetooth.

- Para el funcionamiento del arduino Mega se requiere de una fuente de alimentación que suministre 5 voltios.

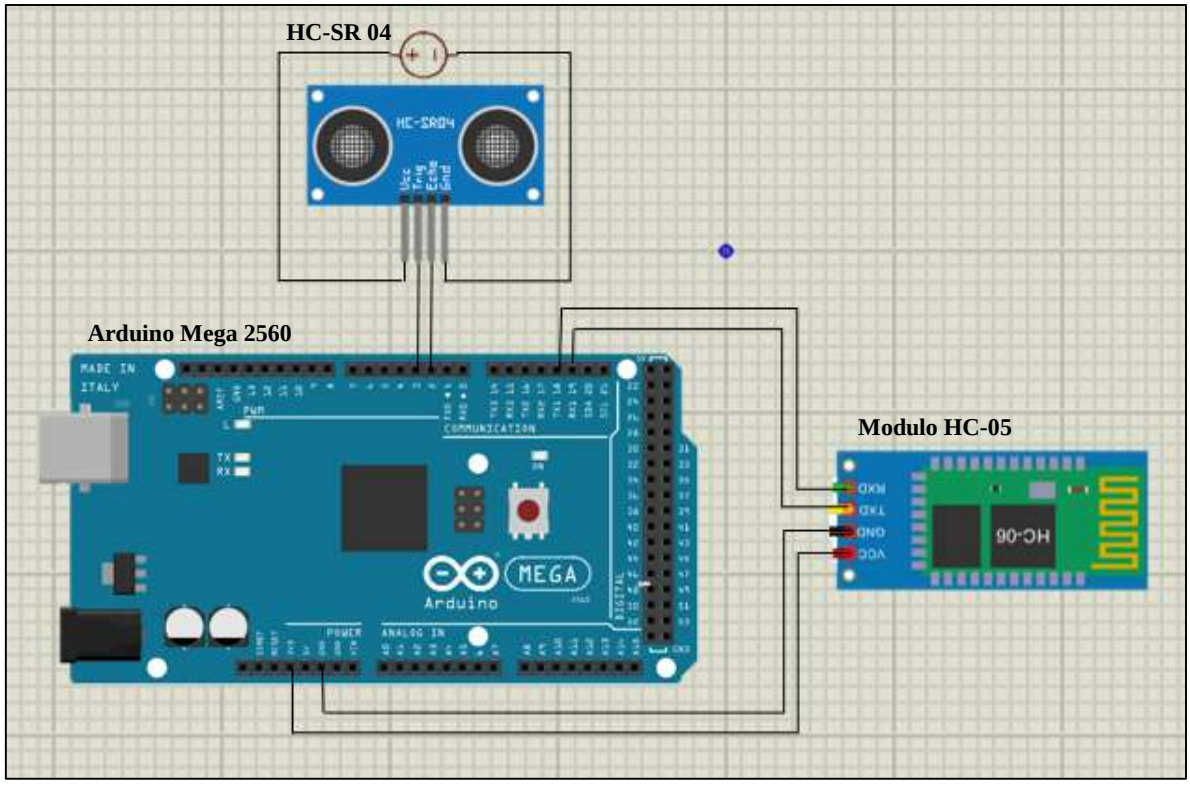


Figura 9-2: Esquema de conexión del nodo sensor

Realizado por: SALTOS, Estalin, 2018

2.2.6.2. Esquema de conexión del nodo recolector

En la figura 10-2, se observa el esquema de conexión para el funcionamiento del nodo recolector, el diagrama se conforma de tres dispositivos principales: Modulo Bluetooth HC-06, Arduino Mega 2560 y NodeMCU ESP8266. La función primordial del nodo recolector es realizada por el NodeMCU ESP8266, porque permite la conectividad a la red utilizando la tecnología WI-FI. Sus elementos están conectados de la siguiente forma:

- El voltaje suministrado para la tarjeta de desarrollo ESP8266 es mediante el Arduino Mega conectándose a los terminales de 5 Voltios y GND.

- La conexión entre el Arduino Mega y el EPS8266 es a través de los terminales 14(TX) y 15(RX) de la tarjeta de desarrollo y los terminales RX y TX del módulo Wi-Fi respectivamente.

- El módulo Bluetooth es alimentado a través de los terminales 3,3 Voltios y GND que es proporcionado por el Arduino Mega.

- El arduino Mega 2560 se conecta mediante los terminales 18(TX1) y 19(RX1) con los terminales RX y TX del módulo Bluetooth.

- Para el funcionamiento del arduino Mega se requiere de una fuente de alimentación que suministre 5 voltios.

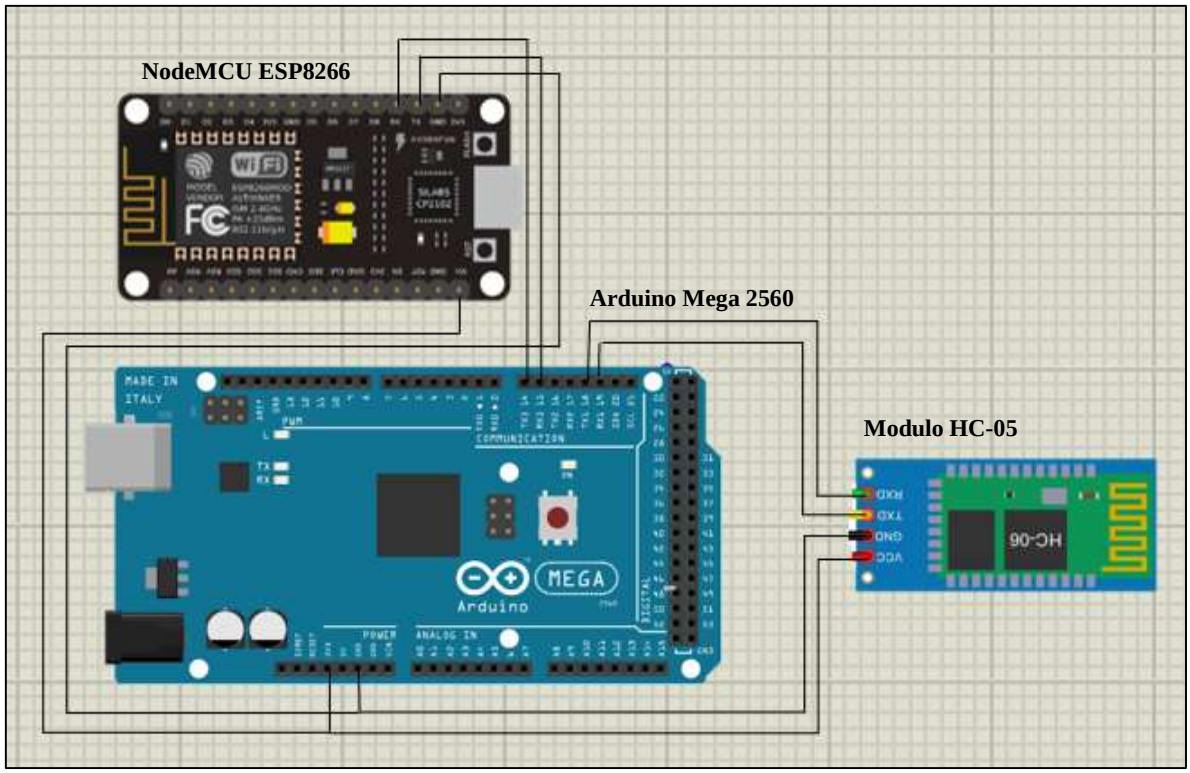


Figura 10-2: Esquema de conexión del nodo recolector

Realizado por: SALTOS, Estalin, 2018

2.3. Selección del software de desarrollo para el sistema de parqueadero

2.3.1. Xampp

Es un software libre de Apache, tiene incluido MySQL, PHP y una variedad de herramientas para realizar distintas aplicaciones web. XAMPP es diseñado para varios sistemas operativos. Con la instalación de esta distribución se realizó la programación para mostrar los datos para el servidor en la red (Gonzalez, s.f)

2.3.2. Netbeans

Diseñado para código abierto y de libre distribución para desarrollar aplicaciones web, móviles, y de escritorio en Java. El IDE consta de editores y herramientas para PHP, HTML, Groovy, JavaScript y C / C ++, etc. NetBeans IDE es ejecutable en los sistemas operativos Windows, Linux hasta sistemas como Mac OS X. Write Once, Run Anywhere. Este software se utilizó para la programación de crear la base de datos (Oracle Corporation, 2013).

2.3.3. Arduino IDE

Arduino IDE (Entorno de Desarrollo Integrado), es un software libre que trabaja con el lenguaje de programación C++ basado en AVR-GCC, Processing. Para el desarrollo de la programación se realizó con la versión IDE 1.5.8, en un editor de texto que permite crear y ejecutar archivos con extensión. Ino, así mismo, contiene gestores de librerías y placas que complementan la programación. Además, su entorno le permite desarrollarse en múltiples sistemas operativos tales como: distribuciones de Linux, Windows y Mac (Delgado, 2018, p. 63)

2.3.4. K2s01(FL - US)

Es un hosting de dominio pagado, mismo que proporciona diferentes planes de servicio anual o mensual de acuerdo con los requerimientos del usuario. Para el sistema de parqueo se contrató un hosting con las siguientes características: 1GB de almacenamiento de disco SSD, 8 GB de ancho de banda, compatibilidad del servidor con versiones estables de Apache, Linux, PHP y MySQL. Además, permite la transmisión de datos mediante el protocolo FTP y se puede efectuar copias de seguridad. La Tabla 7-2, indica las principales características del servidor K2s01 (FL-US).

Tabla 7-2: Principales características del servidor k2s01 (FL-US)

Especificaciones	**Características**
Nombre del servidor	k2s01
Sistema Operativo	Linux
Dirección IP compartida	155.254.28.156
cPanel Versión	74.0
Versión Apache	2.4.35
Versión PHP	7.1.22

Versión MySQL	10.2.18-MariaDB
Arquitectura	X86_64
Paquete de alojamiento	Plan Iniciate

Realizado por: SALTOS, Estalin, 2018

La Tabla 8-2, indica las especificaciones del servidor que se asignado para el sistema de parqueo, mismo que se almacenara la aplicación web y almacenamiento de la información adquirida en tiempo real a través de la red de sensores.

Tabla 8-2: Especificaciones asignado para el sistema de parqueo

Especificaciones	**Características**
Usuario	Netwolfc
Dominio Principal	netwolf-company.com
Shared IP Address	155.254.28.156
Directorio Principal	/home/netwolfc

Realizado por: SALTOS, Estalin, 2018

2.3.5. Requerimientos del software del sistema

A continuación, se describe los requerimientos del software para los nodos que conforman el sistema de parqueo.

2.3.5.1. Requerimiento del nodo sensor

- Adquirir los datos emitidos por los sensores ultrasónicos a través de pulsos eléctricos en tiempo real, convirtiendo estos pulsos en información del estado actual de las plazas de aparcamiento.

- Transmitir los valores obtenidos de los sensores hacia el nodo recolector a través de la comunicación bluetooth.

2.3.5.2. Requerimiento del nodo recolector

- Receptar las tramas bluetooth emitidas por los nodos sensor mediante la comunicación Bluetooth.

- Enviar los datos del estado actual del aparcamiento hacia el servidor para proporcionar la información de disponibilidad al usuario acerca de las plazas de estacionamiento en el parqueadero de la FIE, permitiéndoles reversar una de las mismas.

2.3.6. Diagramas de flujo del sistema de parqueo

A continuación, se ilustra el flujograma del software desarrollado para el nodo sensor y el nodo recolector, detallando las funciones y librerías que se emplean en el prototipo.

2.3.6.1. Diagrama de flujo del nodo sensor

La figura 11-2 muestra el algoritmo del mecanismo del nodo sensor de datos. Los detalles de la programación se presentan a continuación:

Inicialización

- Declaración de las variables y constantes para detectar la presencia de un objeto: rd [], wr [], ec [], tr [].
- Establecer la comunicación del puerto serial mediante la variable Serial1.begin [] que se efectúa con una velocidad de datos de 115200.
- Declaración de los pines de entradas y salidas del pulso generado por los sensores de ultrasonido: pinMode (ec, INPUT), pinMode (tr, OUTPUT).

Bucle repetitivo

- Leer el puerto serial: rd=Serial1.read ()
- Si la lectura del puerto serial es igual al número del sensor que se esté ejecutando, se realiza la función de lectura del sensor.
- Se enviará una respuesta codificado de números binarios que representaran si el aparcamiento está libre u ocupado.
- Si la lectura del puerto serial es distinto al número del sensor que se esté ejecutando, procede a la lectura del siguiente sensor que le corresponda.

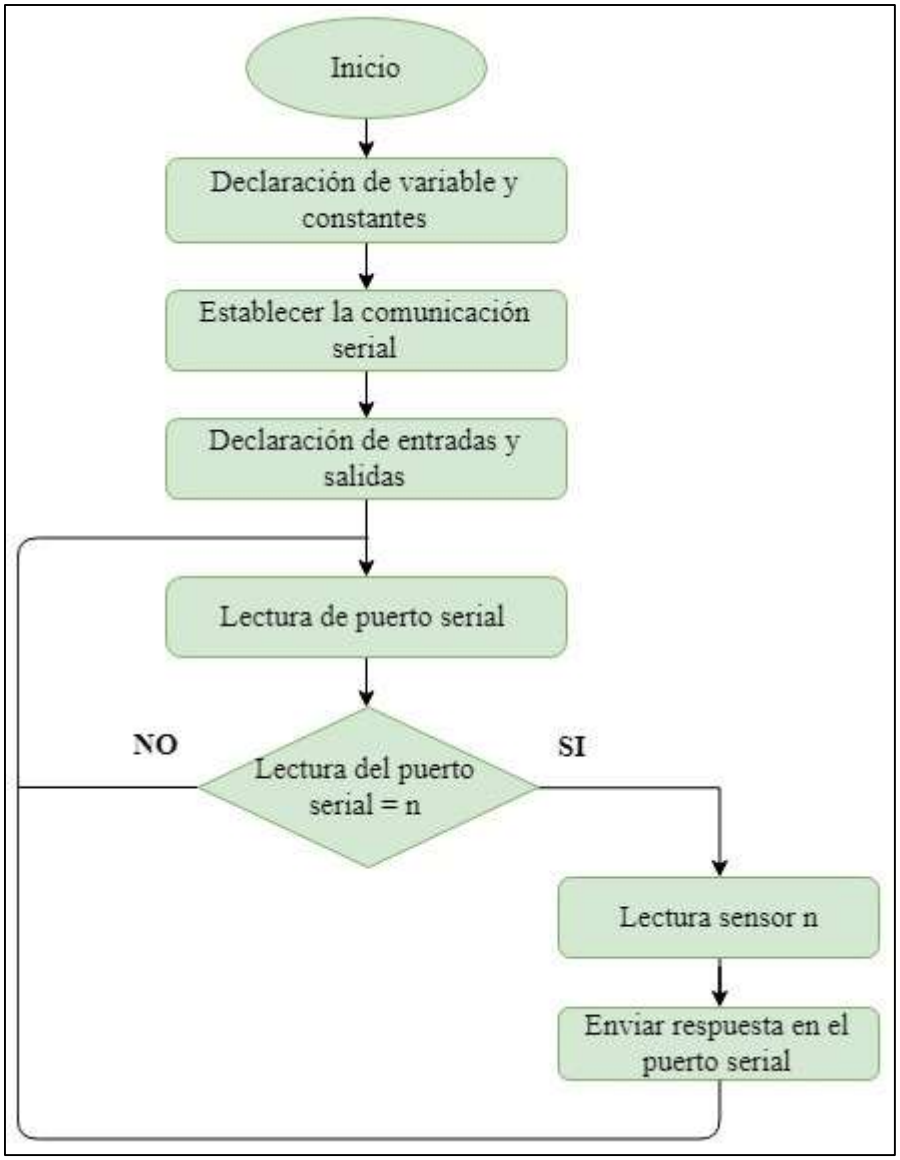


Figura 11-2: Diagrama de flujo del nodo sensor o lector

Realizado por: SALTOS, Estalin, 2018

2.3.6.2. Diagrama de flujo del nodo recolector

La figura 12-2 muestra el algoritmo del mecanismo del nodo recolector de datos. Los detalles de la programación se realizan de la siguiente manera:

Inicialización

- Definir la librería SoftwareSerial.h, la función principal de esta librería consiste en establecer una comunicación serial con el nodo sensor.
- Definir la librería ESP8266WiFi.h, esta librería permite tener comunicación del NodeMCU al Internet a través de la tecnología Wi-Fi
- Definir la librería WiFiClient.h, esta librería permite realizar al NodeMCU las funciones de un cliente.
- Declarar las variables: int chipid, id, st, val
- Declarar las contantes: char* ssid, char* pass, char* host, para ingresar el nombre, contraseña y dirección IP de la red a la cual se va a conectar el NodeMCU.
- Establecer la comunicación del puerto serial mediante la variable SerialX.begin [] que se efectúa con una velocidad de datos de 115200.
- Declarar Inicializar la comunicación Wi-Fi a través de la línea de código: begin (ssid, pass)
- Declaración de los pines de entradas y salidas para la comunicación serial: pinMode (D2, INPUT), pinMode (D3, OUTPUT).

Bucle repetitivo

- Si el contador es menor igual a seis se procede a leer el sensor y se envía la información a través del puerto serial 1.
- Si el contador es mayor a seis y menor igual a doce, se procesa los datos en el puerto serial 2, posterior se lee el puerto serial 1.
- De acuerdo con la lectura del puerto serial 1, si es igual a un número n se procede a leer el sensor y se envía la información a través del puerto serial 1.
- Si el contador es mayor a doce, se procesa los datos en el puerto serial 3, posterior se lee el puerto serial 1.
- De acuerdo con la lectura del puerto serial 1, si es igual a un número n se procede a leer el sensor y se envía la información a través del puerto serial 1.
- Todos los datos se enviarán hacia el servidor a través de una conexión TCP, posterior se procede a cerrar la conexión y pasa al siguiente sensor para ser procesado la información que adquiere.

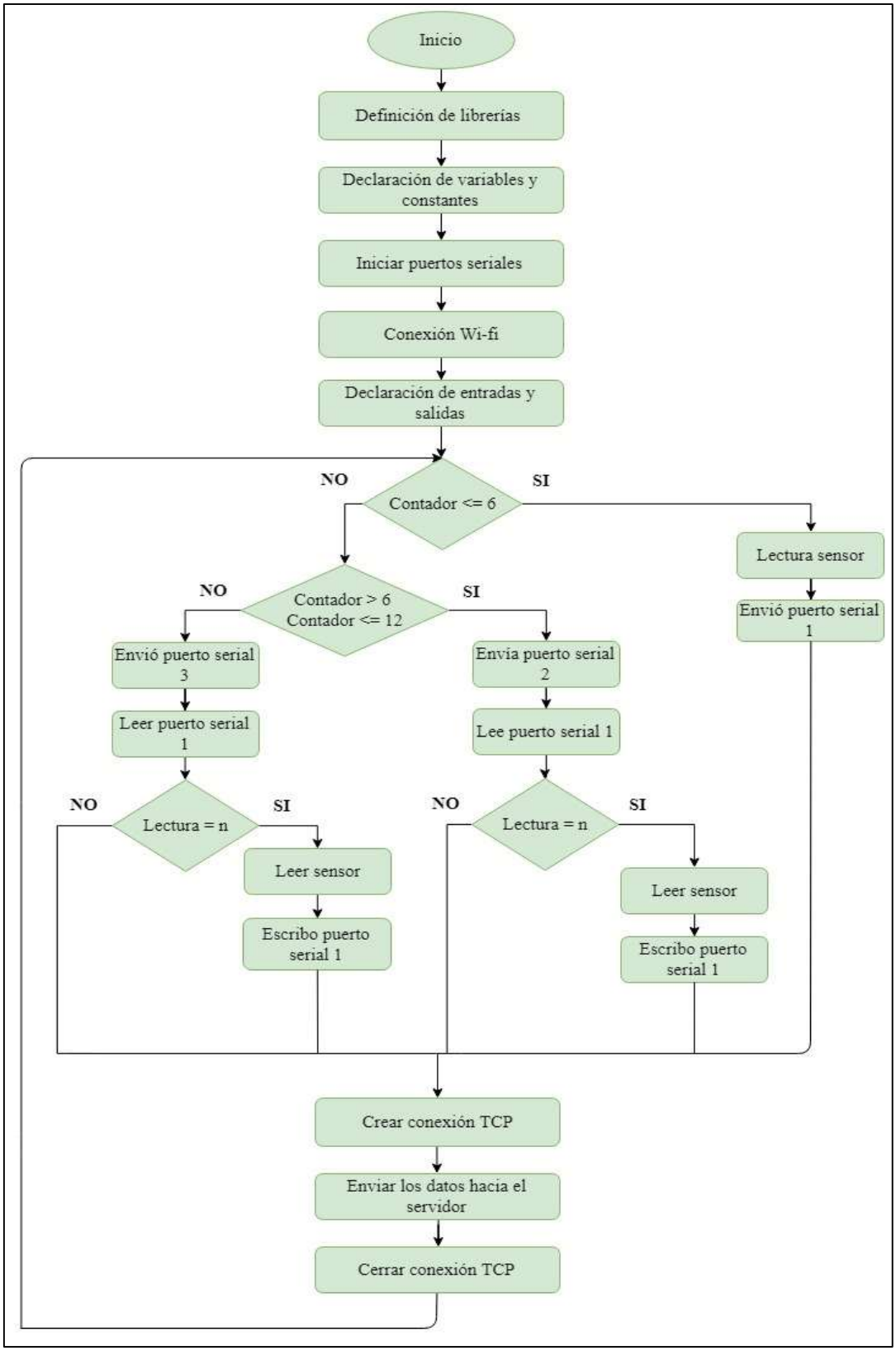


Figura 12-2: Diagrama de flujo del nodo recolector

Realizado por: SALTOS, Estalin, 2018

2.3.7. Visualización de la interfaz web

En la Figura 13-2 se muestra la página de inicio visualizada en la web del sistema de parqueo. Mediante esta interfaz web se puede obtener la información acerca del estado del parqueadero del edificio de la FIE. Consta de diferentes opciones por ejemplo el campo registro es de gran importancia para poder acceder al sistema.

Figura 13-2: Visualización de la aplicación web

Realizado por: SALTOS, Estalin, 2018

Para tener acceso a esta plataforma se debe iniciar sesión, ya establecida la sesión se puede reservar un aparcamiento del estacionamiento durante un tiempo estimado de espera. En caso de que el usuario no cumpla con el tiempo estimado para la reservación tendrá un número máximo de intentos estipulados por el administrador. La figura 14-2, muestra cómo se encuentra el parqueadero del edificio en tiempo real, mismo que se compone de tres colores el cual representa lo siguiente: el color amarillo indica que el aparcamiento está reservado, el color rojo que el aparcamiento está ocupado y el color verde que el aparcamiento esta libre. Se puede tener acceso a esta interfaz web en cualquier lugar a través de Internet.

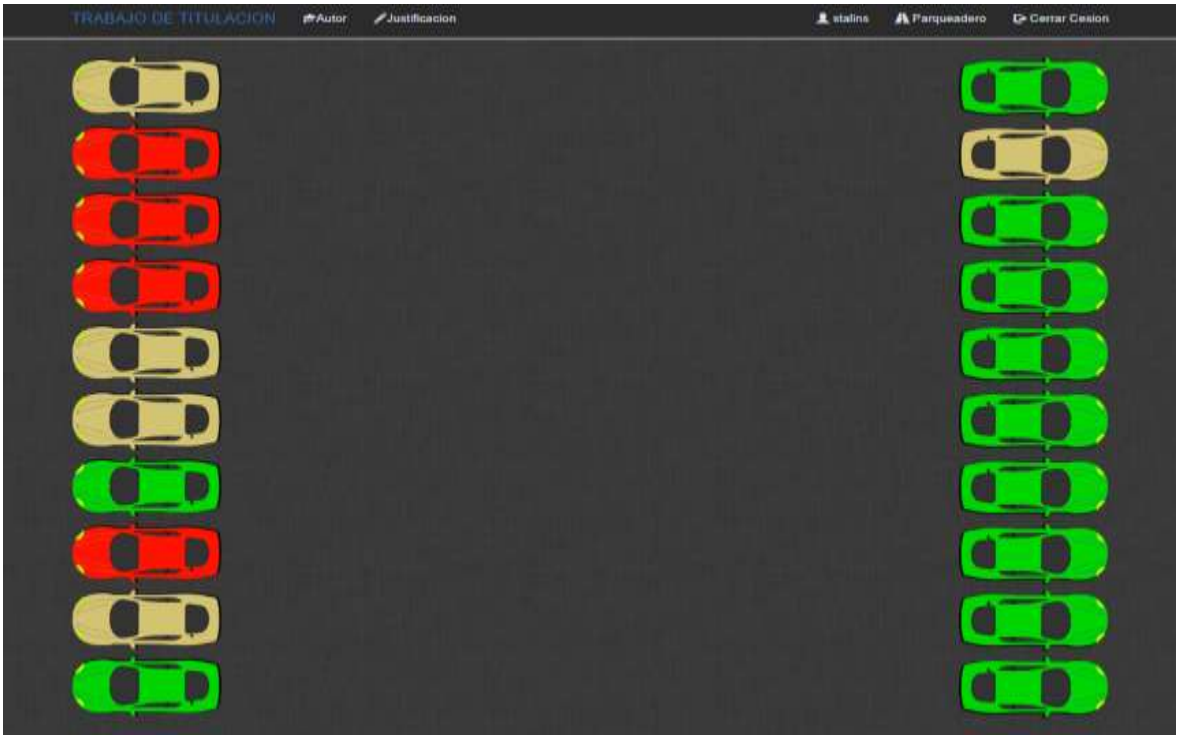


Figura 14-2: Visualización web del estado actual del parqueadero

Realizado por: SALTOS, Estalin, 2018

CAPITULO III

3. PRUEBAS Y RESULTADOS

En este capítulo se detalla los resultados de acuerdo con las pruebas que se ejecutaron con el sistema de parqueo. Se encontró el error del sistema comparando con un equipo patrón y los sensores del prototipo. Se realizó una encuesta para comprobar los requerimientos del sistema y finalmente se presenta el análisis económico del sistema

3.1. Localización del prototipo del sistema de parqueo inteligente implementado

En la Tabla 1-3 se indica los datos exactos donde se encuentra ubicado el sistema de parqueo inteligente. Así mismo, en la Figura 1-3 se presenta una imagen adquirida por la aplicación de Google Earth, acerca del lugar en donde se implementó el prototipo.

Tabla 1-3: Ubicación del sistema

Provincia	Chimborazo
Cantón	Riobamba
Sector	Facultad de Informática y Electrónica de la ESPOCH
Coordenadas	1°39’23.67’’ S 78°40’33.64’’ O
Referencia	Parqueadero del edificio principal de la FIE, Frente a la Escuela de Ingeniería Automotriz.

Realizado por: SALTOS, Estalin, 2018

Figura 1- 3: Ubicación de la implementación del prototipo.

Realizado por: SALTOS, Estalin, 2018

En el interior del parqueadero de la Facultad de Informática y Electrónica de la Escuela Superior Politécnica de Chimborazo, existe 20 plazas de aparcamiento de los cuales 2 estacionamientos son reservados exclusivamente para las autoridades de la facultad, decano y vicedecano.

3.2. Tiempo de subida de datos entre el sistema y el servidor

Para obtener resultados de esta prueba se enviaron datos desde el sistema a la nube, para la cual se tomaron en cuenta 10 pruebas de acuerdo con la Tabla 1-3.

Tabla 2-3: Tiempo de subida de datos entre el sistema y el servidor

N° de prueba	Tiempo	N° de prueba	Tiempo	N° de prueba	Tiempo
1	29500	11	26700	21	23600
2	27000	12	23500	22	23640
3	26300	13	26530	23	25000
4	25100	14	29800	24	21500
5	27900	15	29600	25	27000
6	28600	16	28960	26	28650
7	26400	17	23560	27	29500
8	22000	18	24660	28	23600
9	21400	19	28900	29	21400
10	25900	20	23900	30	29900
Promedio					26000

Realizado por: SALTOS, Estalin, 2018

Se determinó que el tiempo promedio subida de datos entre el sistema y el servidor K2s01(FL-US) es de 26000 ms, por lo tanto, es un tiempo aceptable para el correcto funcionamiento del sistema de parqueo.

3.3. Análisis para el requerimiento del sistema

El diseño del prototipo de sistema de parqueo está orientado para los docentes y estudiantes de la facultad de informática y electrónica con la finalidad de verificar la disponibilidad de plazas de aparcamiento dentro del parqueadero. Se estructuro una encuesta de cinco preguntas para determinar los requerimientos y aceptación del sistema.

Para determinar la cantidad de encuestas a realizar se utilizó la ecuación-1-3.

Ecuación 1-3: Formula para calcular el tamaño de la muestra

$$n = \frac{z^2 * \mathrm{p} * \mathrm{q} * \mathrm{N}}{(\mathrm{N}-1) * e^2 + z^2 * p * q}$$

Donde:

z: nivel de confianza

N: tamaño de población estudiada para determinar la muestra

p: probabilidad de que ocurra algo

q: probabilidad de que no ocurra algo

e: margen de error del muestreo

Para determinar la población se consideró el número de docentes y estudiantes que pertenecen a la facultad de informática y electrónica. La población de docentes es 96 y de estudiantes es 1855, de los cuales el 85% de docentes y el 5% de estudiantes tienen vehículo, es decir 82 docentes y 93 estudiantes aproximadamente. Finalmente, la población total para realizar la encuesta es de 175.

$$n = \frac{(2)^2 * (0.5) * (0.5) * (175)}{(174) * (0.1)^2 + (2)^2 * (0.5) * (0.5)}$$

$$n = \frac{175}{2.74}$$

$$n = 63.8686 \cong 64$$

De acuerdo con el resultado obtenido, aplicando la ecuación 1-3 se determinó que la muestra es de 64 número total de encuestas que a realizarse.

3.4. Resultados y análisis de las encuestas

Se realizaron 64 encuestas entre docentes y estudiantes que poseen vehículos. A continuación, se detallan los resultados y análisis de cada pregunta.

La tabla 3-3, indica los resultados correspondientes de la pregunta 1

Tabla 3-3: Resultados de la pregunta 1

Pregunta 1		
Conocimiento del sistema	**Número de respuestas**	**Porcentaje %**
Si	31	48.4
No	33	51.6

Realizado por: SALTOS, Estalin, 2018

De acuerdo con los resultados obtenidos, se determinó que el 48.4% de las personas encuestadas tienen conocimiento de sistemas de parqueos inteligentes, mientras que el 51.6% de las personas desconocen sobre este sistema. Para tener un mejor contraste acerca de los resultados se muestra en el gráfico 1-3.

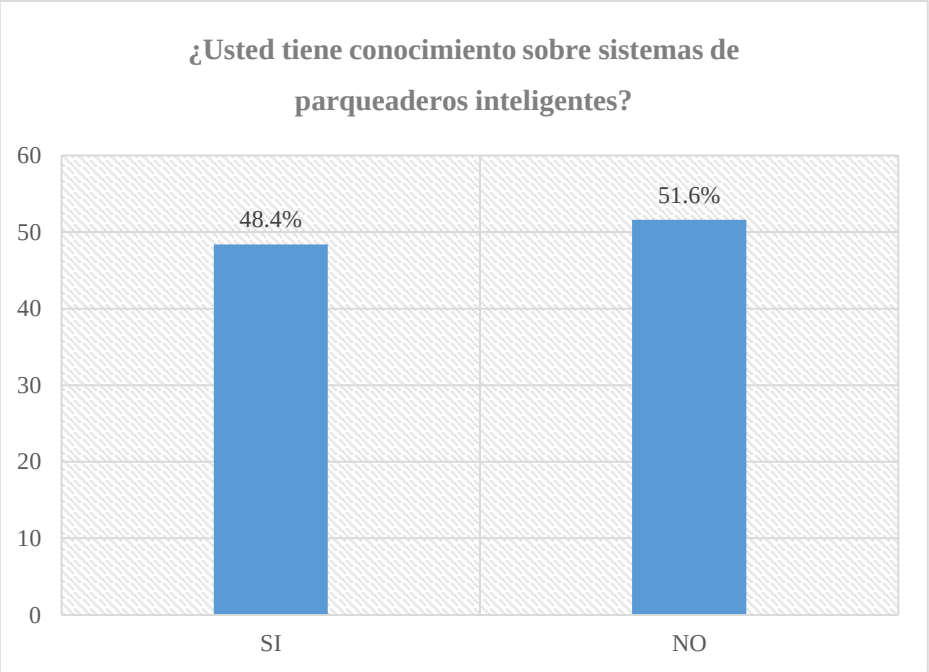


Gráfico 1-3: Resultados de la pregunta 1

Realizado por: SALTOS, Estalin, 2018

En la tabla 4-3, se observa los resultados de la pregunta 2.

Tabla 4-3: Resultados de la pregunta 2

Pregunta 2		
Diseño de un sistema para la FIE	**Número de respuestas**	**Porcentaje %**
Si	54	84.4
No	10	15.6

Realizado por: SALTOS, Estalin, 2018

Con los resultados adquiridos, se determinó que el 84.4% de las personas piensan que se debe diseñar un sistema de parqueo para la Facultad de Informática y Electrónica, mientras que únicamente el

15.6% de las personas consideran que no es necesario. En el gráfico 2-3, se observa los porcentajes anteriormente descritos.

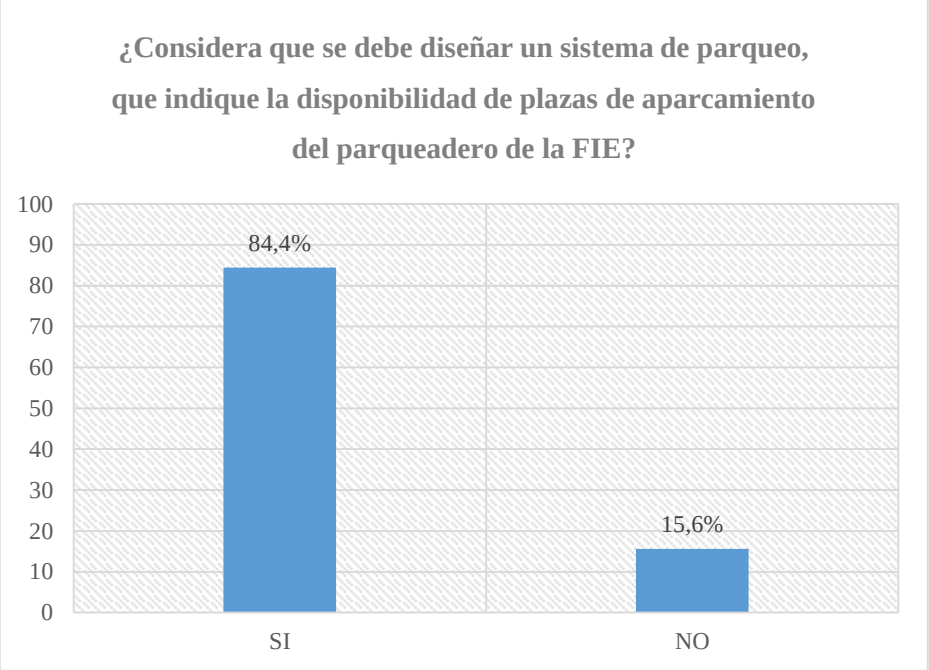


Gráfico 2-3: Resultados de la pregunta 2

Realizado por: SALTOS, Estalin, 2018

En la tabla 5-3, se aprecia los resultados de la pregunta 3.

Tabla 5-3: Resultados de la pregunta 3

Pregunta 3		
Aprovechamiento de tiempo	**Número de respuestas**	**Porcentaje %**
Si	55	85.9
No	9	14.1

Realizado por: SALTOS, Estalin, 2018

Se evidencio que el 85.9% de usuarios consideran que el sistema de parqueo ayudará a reducir el tiempo perdido por buscar una plaza de aparcamiento, mientras que el solamente el 14.1% de usuarios creen que no tendrá ningún beneficio. El gráfico 3-3 indica los porcentajes obtenidos.

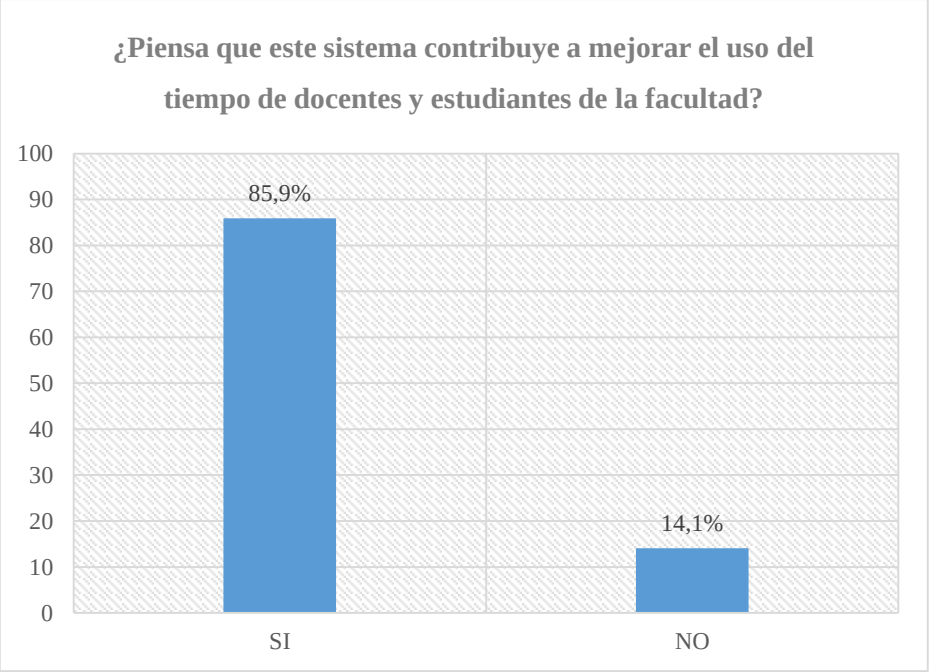


Gráfico 3-3: Resultados de la pregunta 3

Realizado por: SALTOS, Estalin, 2018

En la tabla 6-3, se observa los resultados de la pregunta 4.

Tabla 6-3: Análisis de la pregunta 4

Pregunta 4		
Utilización del sistema	**Número de respuestas**	**Porcentaje %**
Si	52	81.3
No	12	18.8

Realizado por: SALTOS, Estalin, 2018

Con los resultados adquiridos, se determinó que el 81.3% de las personas harían uso del sistema de parqueo para reservar una plaza de aparcamiento, mientras que el 18.8% de las personas no utilizarían este sistema. En el gráfico 4-3 se evidencia los resultados en porcentajes.

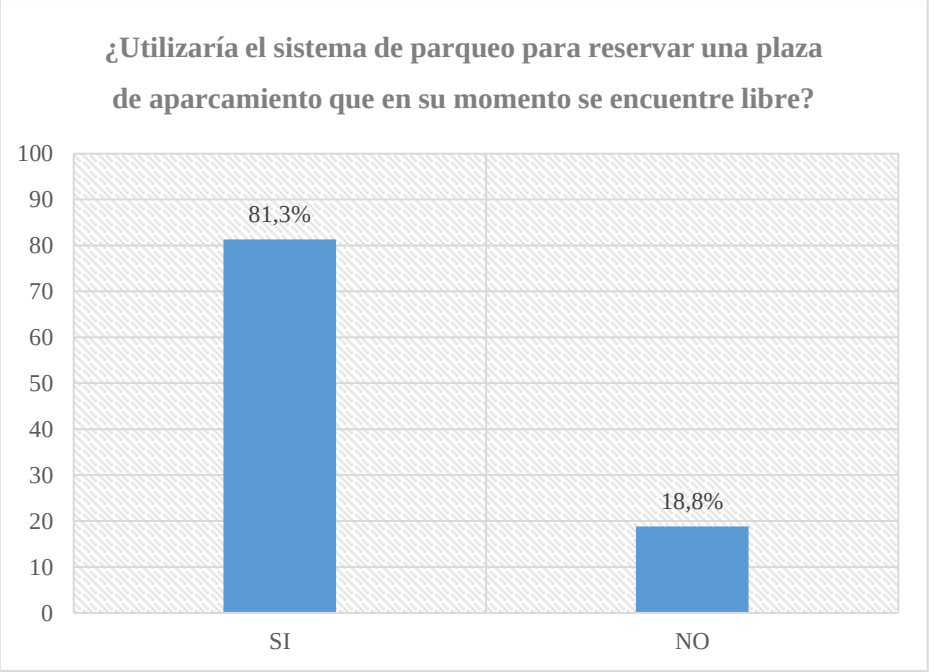


Gráfico 4-3: Resultados de la pregunta 4

Realizado por: SALTOS, Estalin, 2018

En la tabla 7-3, se observa los resultados de la pregunta 5.

Tabla 7-3: Análisis de la pregunta 5

Pregunta 5		
Utilización de la página web	**Número de respuestas**	**Porcentaje %**
Si	46	71.9
No	18	28.1

Realizado por: SALTOS, Estalin, 2018

Con los resultados adquiridos, se determinó que el 71.9% de las personas encuestadas reservarían con mayor facilidad una plaza de aparcamiento a través de una página web, mientras que el 28.1% de personas no harían uso de la página web para reservar un estacionamiento. En el grafico 5-3 se presencia los porcentajes descritos.

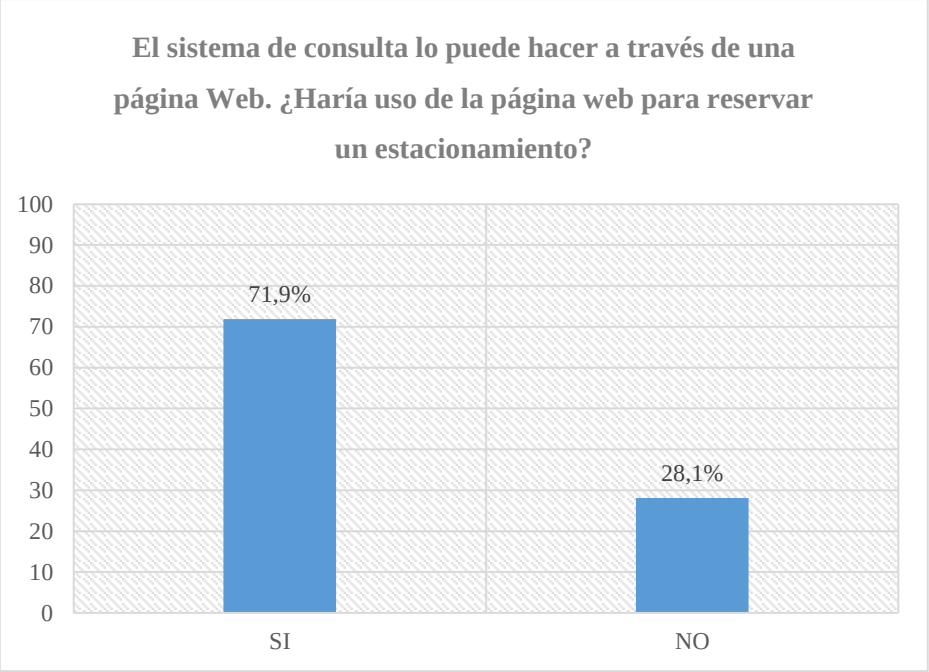


Gráfico 5-3: Resultados de la pregunta 5

Realizado por: SALTOS, Estalin, 2018

3.5. Caracterización del nodo sensor

Para poner en funcionamiento el nodo sensor se realizaron pruebas de comunicación entre los dispositivos que conforman a este.

3.5.1. Mediciones del nodo sensor

El objetivo fundamental es definir si el sistema de parqueo incorpora error al censar, para ello se relacionó los valores del sensor ultrasónico del sistema con un equipo patrón (Flexómetro) como se muestra en la figura 2-3.

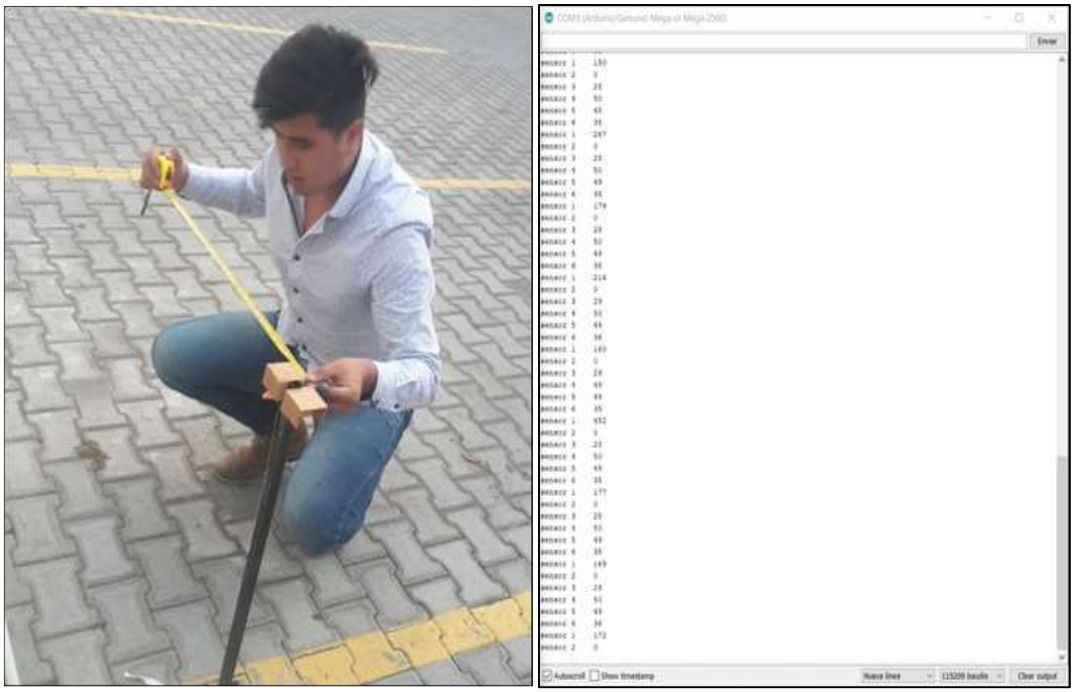


Figura 2-3: Mediciones de la distancia entre un flexómetro(izquierda) y los sensores(derecha)

Realizado por: SALTOS, Estalin, 2018

La tabla 8-3 indica los datos adquiridos entre el sensor ultrasónico del sistema y el flexómetro. Estos datos fueron recopilados durante cinco días tomando 30 muestras, se determina que el sensor del sistema presenta un error de +/- 1 cm.

Tabla 8-3: Comparación de datos entre un flexómetro y el sensor HC-SR04

N° de muestra	Distancia		Error absoluto de la distancia (cm)
	Distancia referencial Flexómetro (cm)	Sensor de distancia HC-SR04 (cm)	
1	45	44	1
2	60	60	0
3	15	16	-1
4	45	45	0
5	45	44	1
6	70	69	1
7	70	70	0
8	60	59	1

9	60	60	0
10	45	44	1
11	15	15	0
12	15	14	1
13	70	70	0
14	15	15	0
15	45	46	-1
16	60	61	-1
17	45	45	0
18	70	70	0
19	45	44	1
20	45	45	0

Realizado por: SALTOS, Estalin, 2018

Se determina que el sistema de parqueo no admite errores, por lo tanto, es capaz de medir la distancia para satisfacer las exigencias del sistema.

3.5.2. Repetividad de datos

El objetivo principal de esta prueba es determinar la estabilidad de los sensores implementado en el parqueadero de la FIE. Para ello se consideró 15 muestras del sensor durante 5 días laborables realizando las mediciones 3 veces al día, como se indica en la tabla 9-3.

Tabla 9-3: Repetividad de datos de la distancia del sensor

N° de muestra	Hora	Distancia
1	29/10/2018	45
2	29/10/2018	44
3	29/10/2018	45
4	30/10/2018	44
5	30/10/2018	45
6	30/10/2018	44
7	31/10/2018	45

8	31/10/2018	45
9	31/10/2018	45
10	5/11/2018	44
11	5/11/2018	45
12	5/11/2018	45
13	6/11/2018	44
14	6/11/2018	45
15	6/11/2018	44
Media		44,6
Desviación estándar		0,51
Coeficiente de variación		1%

Realizado por: SALTOS, Estalin, 2018

Los datos del sensor HC-SR 04 tienen un coeficiente de variación del 1%, de acuerdo con los resultados que indica la tabla 1-3. Según DANE para tener una variabilidad mínima el coeficiente de variación debe ser menor a 7%, por lo tanto, se concluye que el sistema de parqueo es estable.

3.5.3. Pruebas de adquisición de datos

Esta prueba se realizó con el objetivo de verificar que datos son adquiridos y enviados a las tarjetas de desarrollo. Comprobando la integridad de los datos y el correcto funcionamiento de los sensores que conforman el sistema de parqueo de acuerdo con la figura 3-3.

Figura 3-3: Adquisición de datos mediante el sensor

Realizado por: SALTOS, Estalin, 2018

3.6. Caracterización del nodo recolector

El nodo recolector se muestra en la figura. Se realizaron pruebas de comunicación y almacenamiento de datos.

3.6.1. Comunicación entre las tarjetas de desarrollo

Tiene como objetivo principal establecer la comunicación entre las tarjetas de desarrollo del sistema de parqueo, para ello se utiliza la tecnología bluetooth como un medio de transmisión de datos. La figura 4-3 indica todos los datos capturados por cada uno de los sensores en el entorno Arduino IDE.

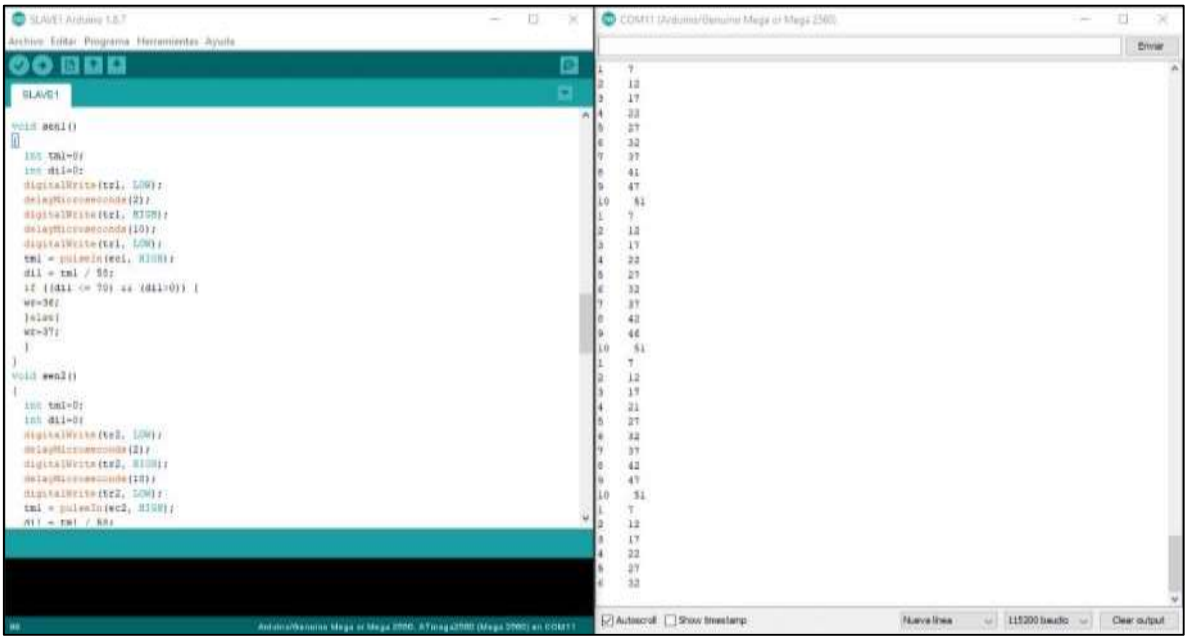


Figura 4-3: Captura de datos en el entorno arduino IDE

Realizado por: SALTOS, Estalin, 2018

Se demostró que en la comunicación entre las tarjetas de desarrollo no existe perdidas de información, la misma que es aceptable para la transferencia de datos.

3.6.2. Comunicación entre ESP8266 y la plataforma IoT

El nodo recolector está conformado por un módulo ESP8266 que se transmite por medio de comunicación inalámbrica El objetivo elemental de esta prueba consiste en establecer la

comunicación inalámbrica y comprobar la integridad de los datos emitidos hacia el servidor. En la figura 5-3, se observa la transmisión de 18 datos hacia la nube, donde son almacenados y presentados de manera gráfica en la aplicación web.

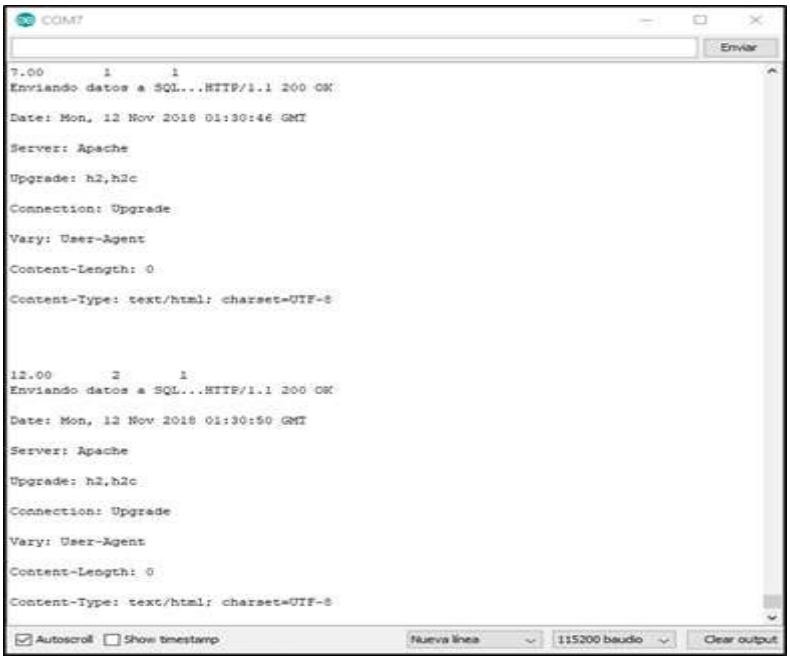


Figura 5-3: Comunicación entre el ESP 8266 y el servidor

Realizado por: SALTOS, Estalin, 2018

Se verifico que las lecturas de los sensores son en tiempo real, indicando los valores actuales de forma gráfica en la plataforma IoT.

3.6.3. Comunicación entre servidor - cliente

Esta prueba se realiza para comprobar la conexión entre el servidor y un ordenador remoto. Para ello, desde el ordenador se accede a la ventana de símbolo del sistema (cmd), posterior se ejecuta el comando ping y la dirección IP asignada del servidor k2s02 (FL-US) 155.254.28.156, como se indica en la figura 6-3.

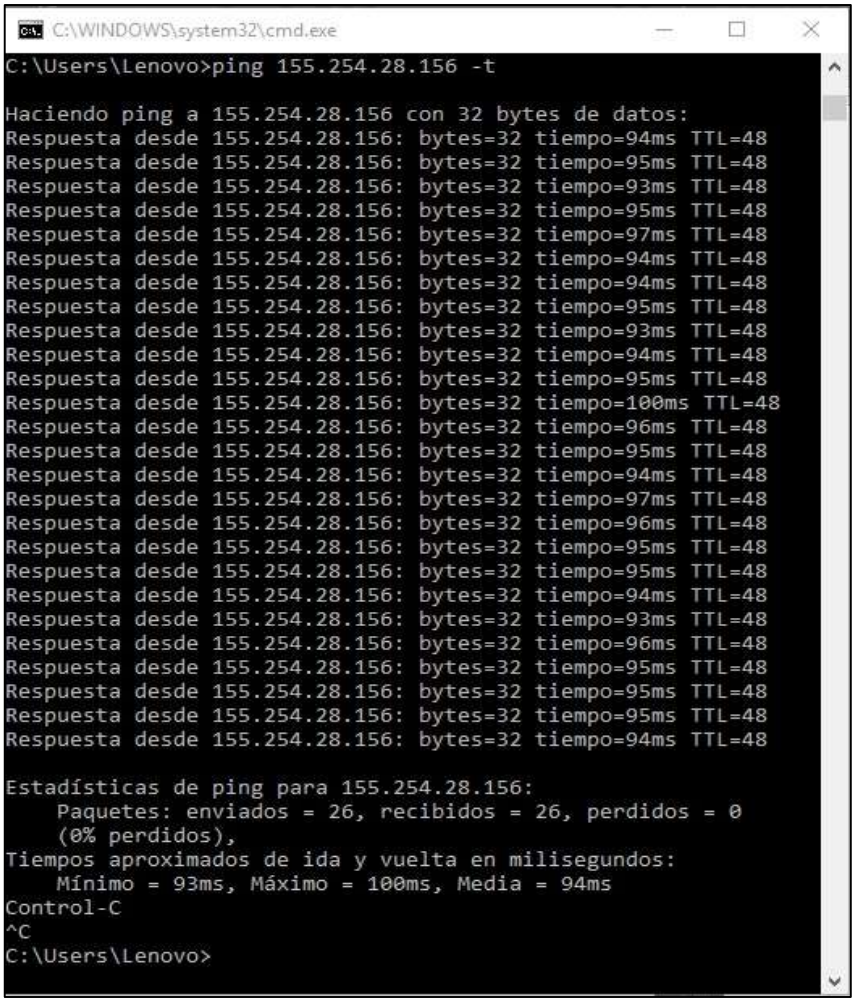


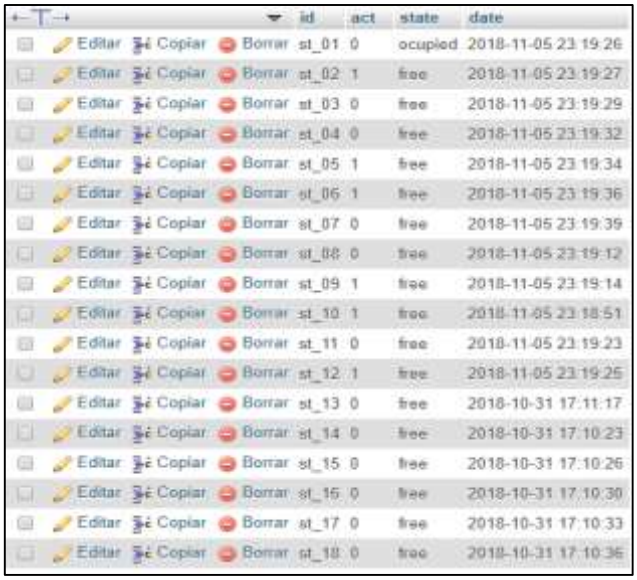
Figura 6-3: Prueba de comunicación cliente – servidor

Realizado por: SALTOS, Estalin, 2018

De la prueba realizada se verificó que existe conexión entre el servidor y el cliente. El cual, se enviaron 26 paquetes obteniendo los mismos de recepción, determinando el 0% de perdidas de datos con un tiempo de respuesta media aproximado de 94 ms de ida y vuelta de paquetes.

3.6.4. Almacenamiento de datos

Para la verificación del almacenamiento de los datos adquiridos por los sensores en el servidor K2s01(FL-US), se utilizó la herramienta PHP que permite la visualización de la información de forma detallada del estado del parqueadero de la FIE, según la figura 7-3.

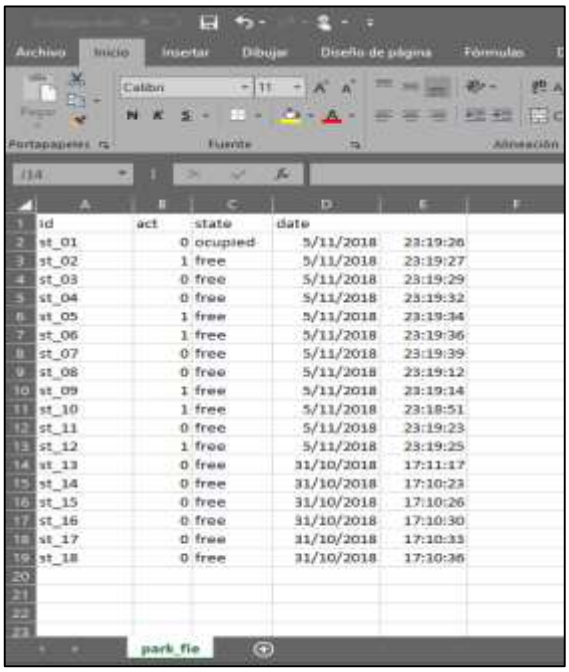


	id	act	state	date
Editar Copiar Borrar	st_01	0	ocupied	2018-11-05 23:19:26
Editar Copiar Borrar	st_02	1	free	2018-11-05 23:19:27
Editar Copiar Borrar	st_03	0	free	2018-11-05 23:19:29
Editar Copiar Borrar	st_04	0	free	2018-11-05 23:19:32
Editar Copiar Borrar	st_05	1	free	2018-11-05 23:19:34
Editar Copiar Borrar	st_06	1	free	2018-11-05 23:19:36
Editar Copiar Borrar	st_07	0	free	2018-11-05 23:19:39
Editar Copiar Borrar	st_08	0	free	2018-11-05 23:19:12
Editar Copiar Borrar	st_09	1	free	2018-11-05 23:19:14
Editar Copiar Borrar	st_10	1	free	2018-11-05 23:18:51
Editar Copiar Borrar	st_11	0	free	2018-11-05 23:19:23
Editar Copiar Borrar	st_12	1	free	2018-11-05 23:19:25
Editar Copiar Borrar	st_13	0	free	2018-10-31 17:11:17
Editar Copiar Borrar	st_14	0	free	2018-10-31 17:10:23
Editar Copiar Borrar	st_15	0	free	2018-10-31 17:10:26
Editar Copiar Borrar	st_16	0	free	2018-10-31 17:10:30
Editar Copiar Borrar	st_17	0	free	2018-10-31 17:10:33
Editar Copiar Borrar	st_18	0	free	2018-10-31 17:10:36

Figura 7-3: Almacenamiento de la información en el servidor

Realizado por: SALTOS, Estalin, 2018

Se demostró que los datos se almacenan de forma correcta en la nube, esta información puede ser utilizada para realizar un análisis a futuro, además estos datos pueden ser descargados del servidor en archivos Excel, de acuerdo con la Figura 7-3.



	A	B	C	D	E	F
1	id	act	state	date		
2	st_01	0	ocupied	5/11/2018	23:19:26	
3	st_02	1	free	5/11/2018	23:19:27	
4	st_03	0	free	5/11/2018	23:19:29	
5	st_04	0	free	5/11/2018	23:19:32	
6	st_05	1	free	5/11/2018	23:19:34	
7	st_06	1	free	5/11/2018	23:19:36	
8	st_07	0	free	5/11/2018	23:19:39	
9	st_08	0	free	5/11/2018	23:19:12	
10	st_09	1	free	5/11/2018	23:19:14	
11	st_10	1	free	5/11/2018	23:18:51	
12	st_11	0	free	5/11/2018	23:19:23	
13	st_12	1	free	5/11/2018	23:19:25	
14	st_13	0	free	31/10/2018	17:11:17	
15	st_14	0	free	31/10/2018	17:10:23	
16	st_15	0	free	31/10/2018	17:10:26	
17	st_16	0	free	31/10/2018	17:10:30	
18	st_17	0	free	31/10/2018	17:10:33	
19	st_18	0	free	31/10/2018	17:10:36	
20						
21						
22						
23						

Figura 8-3: Archivo Excel de los datos almacenados en el servidor

Realizado por: SALTOS, Estalin, 2018

3.7. Consumo de energía del sistema

Esta prueba se realizó para determinar el consumo de voltaje y corriente de cada dispositivo electrónico que compone el sistema de parqueo. Apoyados de un multímetro, en la Tabla 10-3, se presentan valores reales del voltaje y corriente, y valores teóricos establecidos.

Tabla 10-3: Consumo de corriente y voltaje del sistema

Dispositivo	**Consumo de corriente**		**Voltaje de entrada**	
	Medidos (mA)	**Teóricos (mA)**	**Medidos (V)**	**Teóricos (V)**
Arduino Mega 2560	38.17	20	5.37	5
Sensor ultrasónico	18.25	15	5.28	5
Módulo Bluetooth	62.49	50	3.36	3.3
Node MCU	125.32	80	3.39	3.3
Consumo del sistema	755.8			

Realizado por: SALTOS, Estalin, 2018

De las pruebas realizadas, se determina que los valores medidos son aproximados de los valores teóricos tanto en el consumo de voltaje de operación y de corriente. El consumo de corriente total aproximado del sistema es de 755.8 mA.

3.8. Funcionamiento general del sistema

Luego de validar cada uno de los componentes que conforman el sistema de parqueo e implementar en el parqueadero de la Facultad de Informática y Electrónica de la Escuela Superior Politécnica de Chimborazo, se evaluó el funcionamiento total del sistema determinando la comunicación inalámbrica, visualización del estado actual del parqueadero, además permite al usuario reservar una plaza de aparcamiento y el almacenamiento de los datos en el servidor, cumpliendo con los requerimientos establecidos en la investigación. En la figura 9-3 se indica el nodo sensor y recolector, mientras que en la figura 10-3 se observa el prototipo funcional del parqueadero.

Figura 9-3: Nodo sensor (Izquierdo), Nodo Recolector (Derecho)

Realizado por: SALTOS, Estalin, 2018

Figura 10-3: Sistema de parqueo inteligente implementado

Realizado por: SALTOS, Estalin, 2018

3.9. Análisis económico del sistema de parqueo

La tabla 10-3 muestra el presupuesto económico de los dispositivos que compone el sistema de parqueo inteligente. Para ello se realizó el analizado de costo de cada uno de los dispositivos.:

Tabla 11-3: Detalle del presupuesto económico del sistema de parqueo

Nodo del sistema de parqueo	**Dispositivo**	**Cantidad**	**Costo**	**Costo Total**
Nodo Sensor	Tarjeta de desarrollo Arduino Mega	2	$ 35,00	$ 70,00
	Sensor ultrasónico HC-SR04	18	$ 7,00	$ 126,00
	Fuente de alimentación	2	$ 10,00	$ 20,00
	Modulo Bluetooth	2	$ 15,00	$ 30,00
	Cables de conexión	1	$ 35,00	$ 35,00
	Ensamblaje	1	$ 100,00	$ 100,00
	Bases metálicas	9	$ 10,00	$ 90,00
	Total del Nodo sensor			$ 471,00
Nodo Recolector	Tarjeta de desarrollo Arduino Mega	1	$ 35,00	$ 35,00
	NodeMCU Esp 8266	1	$ 15,00	$ 15,00
	Modulo Bluetooth	1	$ 15,00	$ 15,00
	Cables de conexión	1	$ 35,00	$ 35,00
	Ensamblaje	1	$ 50,00	$ 50,00
	Total del Nodo Recolector			$ 150,00
COSTO TOTAL DEL SISTEMA DE PARQUEO				$ 621,00

Realizado por: SALTOS, Estalin, 2018

CONCLUSIONES

- Por medio del estudio de sistemas de parqueo inteligente, se logró diseñar el prototipo de un sistema de aparcamiento específicamente con el uso de sensores que detectan automáticamente la presencia de un objeto. Además, aportan una variedad de beneficios como: disminución de la contaminación ambiental, mejorar el rendimiento en las actividades a través del tiempo de los conductores disminuyendo la congestión vehicular.

- De acuerdo con las comparaciones realizadas entre las características de tecnologías inalámbricas, se definió que las tecnologías apropiadas para el sistema de parqueo inteligente son, Bluetooth para establecer la comunicación entre nodos y Wi-Fi para la transferencia de datos a la Plataforma IoT.

- Mediante la encuesta realizada a docentes y estudiantes de la FIE, se pudo determinar los requerimientos del sistema y posterior se realizó un análisis para determinar el porcentaje de aceptación del sistema el cual fue favorable con un 81.3%.

- El sistema implementado cumple con los requerimientos establecidos, es decir que el prototipo es seguro, escalable y de bajo costo con un valor aproximado de $621.00, permitiéndole insertar otros dispositivos para el mejoramiento de este en un fututo.

- Mediante las pruebas realizadas se determinó que el sistema implementado posee errores de +/- 1 centímetro de distancia al censar, el tiempo promedio de subida de datos entre el sistema y el servidor es de 26000 ms y el coeficiente de variación correspondiente a la repetividad de datos es 1% valor admitido según DANE.

- Se comprobó la conexión entre el servidor - cliente, determinando que el desarrollo de la Plataforma IoT es aceptable, con un tiempo media de 94 ms de ida y vuelta y 0% de perdidas de paquetes enviados hacia el servidor k2s01(FL-US) a través del protocolo HTTP, el modelo de comunicación dispositivo a internet y en tiempo real.

RECOMENDACIONES

- Incorporar un módulo de control de acceso vehicular al ingreso de los parqueaderos para identificar a los usuarios que anticipadamente hicieron uso de la reservación de una plaza de aparcamiento a través de la aplicación web del sistema.

- Se recomienda que este sistema se utilice en parqueaderos cubiertos, como seguridad para los dispositivos, colocándose en lugares donde no obstruya a los conductores al momento estacionarse.

- Usar un regulador de 5v para garantizar el voltaje adecuado de alimentación del sistema, con la finalidad de evitar daños en los dispositivos.

- Integrar dispositivos con mayor capacidad para mejorar la transmisión de datos y reducir el retardo existente en el sistema de parqueo al momento de enlazar los datos hacia el servidor.

- Se recomienda que, en el sistema de parqueo inteligente, se utilice cajas impermeables para proteger a los dispositivos electrónicos y no sean afectados por condiciones climáticas.

BIBLIOGRAFIA

ARDUINO TEC, *Multi proyectos 0006.* [en línea]. 2010. [Consulta: 5 noviembre 2018]. Disponible en: http://arduinoteco.blogspot.com/2013/12/multi-proyectos-0006.html.

BLIZNAKOFF, D., *IoT : TECNOLOGÍAS , usos, tendencias y desarrollo futuro* [en línea]. 2014. [Consulta: 13 noviembre 2018]. 2014. Disponible en: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/40044/6/dbliznakoffTFM0115memoria.pdf.

BOCCALARI, E. y GONZALEZ, F., *rParking (Sistema de plazas de estacionamiento reservadas)* [en línea]. (tesis) (pregrado). UNIVERSIDAD NACIONAL DE LA PLATA. 2016. [Consulta: 14 noviembre 2018]. Disponible en: http://sedici.unlp.edu.ar/bitstream/handle/10915/59485/Documento_completo.pdf-PDFA.pdf?sequence=1.

BRAVO, B. y BELDUMA, L., *DISEÑO DE UNA RED DE SENSORES INALAMBRICOS PARA EL MONITOREO DEL TRANSITO VEHICULAR Y LA CONTAMINACION CO2 DENTRO DE UN SECTOR URBANO* [en línea]. (tesis) (pregrado). UNIVERSIDAD POLITECNICA SALESIANA. 2017. [Consulta: 5 octubre 2018]. Disponible en: https://dspace.ups.edu.ec/bitstream/123456789/14153/1/UPS-CT006969.pdf.

CASAGRAS, *RFID and the Inclusive Model for the Internet of Things.* [en línea]. [Consulta: 25 octubre 2018]. Disponible en: http://rwi.future-internet.eu/index.php/Main_Page.

CASTELLS, M. y MARTÍNEZ, C., *La sociedad red* [en línea]. Alianza. 1997. [Consulta: 5 septiembre 2018]. ISBN 8420642479. Disponible en: http://estudiantes.iems.edu.mx/cired/html/articulos/politicainformactica/topologias.html.

COBOS, M. y ORTIZ, M., *Implementación de un prototipo de una red inalambrica de sensores para la identificación de personas y acceso a historias clínicas basadas en tarjetas de desarrollo.* [en línea]. (tesis) (pregrado). ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO. 2017. [Consulta: 5 noviembre 2018]. Disponible en: http://dspace.espoch.edu.ec/bitstream/123456789/6869/1/98T00148.pdf.

CORONEL, V. y TENELANDA, D., *Análisis de interoperabilidad de plataformas IoT aplicado al desarrollo de un sistema de monitoreo de polución de aire para la ESPOCH* [en línea]. (tesis) (pregrado). ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO. 2016. [Consulta: 6 julio 2018]. Disponible en: http://dspace.espoch.edu.ec/bitstream/123456789/5440/1/98T00093.pdf.

CUEVAS, M. y CARRILLO, I., *Bluetooth* [en línea]. [Consulta: 5 octubre 2018]. Disponible en: http://trajano.us.es/~fornes/RSR/2010/presentacion_bluetooth.pdf?fbclid=IwAR0y_ML8TPrNqc9h1KwinjLge5qxorfeCjBqHsqoZCrYLqqAJxBUN9ZTTMU.

DEL VALLE, L., *ESP8266 todo lo que necesitas saber del módulo WiFi para Arduino.* [en línea]. 2018. [Consulta: 12 noviembre 2018]. Disponible en: https://programarfacil.com/podcast/esp8266-wifi-coste-arduino/.

DELGADO, O., *PROPUESTA DE UN SISTEMA AUTOMATIZADO PARA CONTROLAR EL ACCESO VEHICULAR EN LA ESPOCH MEDIANTE EL USO DE TECNOLOGÍAS INALÁMBRICAS* [en línea]. (tesis) (pregrado). ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO FACULTAD. 2018. [Consulta: 12 noviembre 2018]. Disponible en: http://dspace.espoch.edu.ec/bitstream/123456789/8446/1/98T00192.pdf.

DIAZ, J., *Placa Arduino UNO.* [en línea]. 2016. [Consulta: 12 noviembre 2018]. Disponible en: http://www.iescamp.es/miarduino/2016/01/21/placa-arduino-uno/.

ELECFREAKS, *Ultrasonic Ranging Module HC-SR04* [en línea]. 2016. Disponible en: www.Elecfreaks.com.

FLORES, O., et al, *Implementación de un entorno de aprendizaje virtual integrando herramientas de E-learning y CMS. Recopilación de Investigaciones en Tecnología 2016* [en línea]. 2016. [Consulta: 8 noviembre 2018]. ISBN 978-99961-48-62-0. Disponible en: http://www.utec.edu.sv/media/publicaciones/flips/coleccionInvestigaciones/2016/investigaciones_tecnologia/files/libro_59_investigaciones.pdf?fbclid=IwAR3cyU-0npPTwfK3DKRkjEjWQPRH6akj7AB6YRlaACm0Lv0afEMD2NlYHUY.

GODOY, C., *DESARROLLO DE UN PROTOTIPO DE PARQUEADERO INTELIGENTE PARA LA*

AUTOMATIZACIÓN DEL SISTEMA DE APARCAMIENTO SIMERT EN LA CIUDAD DE LOJA [en línea]. (tesis) (pregrado). UNIVERSIDAD NACIONAL DE LOJA. 2015. [Consulta: 14 noviembre 2018]. Disponible en: http://dspace.unl.edu.ec/jspui/bitstream/123456789/11405/1/Godoy Ramón%2C Cristina Stefanía.pdf.

GÓMEZ, E., et al, *SISTEMA DE COMUNICACIÓN CON EQUIPOS MOVILES PARA MUSEOS UTILIZANDO TECNOLOGIA NFC* [en línea]. (tesis) (pregrado). INSTITUTO POLITÉCNICO NACIONAL. 2016. [Consulta: 10 junio 2018]. Disponible en: https://tesis.ipn.mx/bitstream/handle/123456789/21629/TesisSistemadeComunicacionconequiposMovilesRevisada ok.pdf?sequence=1&isAllowed=y.

GONZÁLEZ, A., *IoT : Dispositivos , tecnologías de transporte y aplicaciones* [en línea]. (tesis) (pregrado). Universitat Oberta de Catalunya. 2017. [Consulta: 23 agosto 2018]. Disponible en: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/64286/3/agonzalezgarcia0TFM0617memoria.pdf.

GONZALEZ, J., *Desarrollo de sitios web con PHP y MySQL* [en línea]. [Consulta: 12 noviembre 2018]. Disponible en: http://www.lsi.us.es/cursos/cursophp/apuntes/tema1.pdf.

GUSQUI, Y., *DISEÑO DE UN PROTOTIPO DE RED WSN PARA EL MONITOREO DEL NIVEL DE CONTAMINACIÓN DE CO2 EXISTENTE EN EL CENTRO DE LA CIUDAD DE RIOBAMBA*. (tesis) (pregrado). ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO. 2017.

INTEL, *WiFi diferentes protocolos y las velocidades de datos.* [en línea]. 2017. [Consulta: 2 septiembre 2018]. Disponible en: https://www.intel.la/content/www/xl/es/support/articles/000005725/network-and-i-o/wireless-networking.html.

UNION UNIVERSAL DE TELECOMUNICACIONES, *Visión general de la Internet de las cosas (ITU-T Y.4000/Y.2060 (06/2012)).* [en línea], 2012. Disponible en: http://handle.itu.int/11.1002/1000/11559.

LÓPEZ, J.A., *Sensores (Sensors) vs Actuadores (Actuators).* [en línea]. 2015. [Consulta: 12 noviembre 2018]. Disponible en: http://www.tuataratech.com/2015/06/sensores-sensors-vs-

actuadores-actuators_8.html?fbclid=IwAR2LCfZO-1XbiULNBmvO_fXL4LG8CDqkr1tPoZGyJhkET-Iif4kHJhpLQ1Q.

LOUREIRO, R., *Estudio Plataformas IoT* [en línea]. 2015. [Consulta: 5 noviembre 2018]. 2015. ISBN 0197-7261. Disponible en: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/42812/6/rloureiroTFC0615memoria.pdf.

LUIS, L., *Estudio del impacto técnico y económico de la transición de internet al internet de las cosas (IoT) para el caso Colombiano* [en línea]. Universidad Nacional de Colombia. 2015. [Consulta: 5 noviembre 2018]. Disponible en: http://bdigital.unal.edu.co/50458/1/Estudio Técnico y Económico de la transición de Internet al Internet de las Cosas %28IoT%29 en el caso colombiano.pdf.

MENDOZA, E., *DISEÑO Y CONSTRUCCION DE UNA RED DE COMPUTO BAJO NORMAS INTERNACIONALES, APLICADAS PARA UN LABORATORIO DE REDES DE COMPUTADORAS* [en línea]. ISTITUTO POLITECNICO NACIONAL. 2012. [Consulta: 5 noviembre 2018]. Disponible en: https://tesis.ipn.mx/xmlui/bitstream/handle/123456789/10768/19.pdf?sequence=1.

MENDOZA, J.C., *Implementación de una plataforma de mensajería unificada integrada a una aplicación B2B* [en línea]. (tesis) (pregrado). PONTIFICIA UNIVERSIDAD CATOLICA DEL PERU. 2008. [Consulta: 16 agosto 2018]. Disponible en: http://tesis.pucp.edu.pe/repositorio/bitstream/handle/123456789/211/MENDOZA_JAMPIER_IMPLEMENTACION_PLATAFORMA_MENSAJERIA_INTEGRADA_APLICACION_B2B.pdf?sequence=2.

MORENO, F., *Demostrador arquitectura publish/subscribe con MQTT.* [en línea]. (tesis) (pregrado). 2018. [Consulta: 15 agosto 2018]. Disponible en: https://upcommons.upc.edu/bitstream/handle/2117/117782/MQTT_MEMORIA.pdf?sequence=1&isAllowed=y.

MQTT, *Frequently Asked Questions.* [en línea]. 2018. [Consulta: 3 agosto 2018]. Disponible en: http://mqtt.org/faq.

ORACLE CORPORATION, *NetBeans IDE - Overview.* [en línea]. 2013. [Consulta: 12 noviembre 2018]. Disponible en: https://netbeans.org/features/index.html.

PAREDES, M., *Implementación De Un Prototipo De Wsn Con Nodos Inteligentes Para El Sistema De Riego Aplicado a La Agricultura De Precisión Para El Cer – Espoch* [en línea]. (tesis) (pregrado). ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO. 2017. Disponible en: http://dspace.espoch.edu.ec/bitstream/123456789/7956/1/98T00173.pdf.

PEÑA, J. y SUQUILLO, G., *Estudio del modelo de referencia del internet de las cosas, con la implementacion de un prototipo domotico* [en línea]. (tesis) (pregrado). ESCUELA POLITÉCNICA NACIONAL FACULTAD. 2016. [Consulta: 5 noviembre 2018]. Disponible en: http://bibdigital.epn.edu.ec/bitstream/15000/15096/1/CD-6908.pdf.

PERLES, À., *Empezar con la Raspberry Pi (RPi)* [en línea]. 2017. [Consulta: 12 noviembre 2018]. 2017. Disponible en: http://www.disca.upv.es/aperles/asignatures/oses/Empezar-Raspberry-Pi.pdf.

PINTEREST, *Ubidots.* [en línea]. 2018. [Consulta: 5 noviembre 2018]. Disponible en: https://www.pinterest.es/pin/664703226223742446/?lp=true.

RAMIREZ, D. y RODRÍGUEZ, E., *Diseño de un método para identificar necesidades y oportunidades para la implementación de Internet de las cosas (IoT) aplicable a oficinas de trabajo donde permanezcan entre 30 y 70 personas y planteamiento de un caso práctico de solución en las oficinas* [en línea]. UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS. 2016. Disponible en: http://repository.udistrital.edu.co/bitstream/11349/5343/1/RamirezMadridDavidAndres2017.pdf.

RAO, J., et al, *Zigbee wireless technology.* [en línea], 2015. [Consulta: 15 octubre 2018]. Disponible en: https://www.academia.edu/12329792/zigbee_wireless_technology?auto=download.

RODAS, U., *Diseño de una red inalámbrica de sensores para el monitoreo de una empresa en Lima-Perú* [en línea]. (tesis) (pregrado). PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ FACULTAD. 2013. [Consulta: 22 julio 2018]. Disponible en: http://tesis.pucp.edu.pe/repositorio/bitstream/handle/123456789/5425/RODAS_URPI_DISEÑO_RED_INALAMBRICA_SENSORES_EMPRESA_PERU.pdf?sequence=1&isAllowed=y.

RODRÍGUEZ, D., *Arquitectura y Gestión de la IoT IoT Network Management / Abstract. Revista*

Telem@tica [en línea], 2013. [Consulta: 27 agosto 2018]. ISSN 1729-3804. DOI 10.1063/1.442204. Disponible en: http://revistatelematica.cujae.edu.cu/index.php/tele.

ROSALES, L., *Diseño e implementación de un parqueo inteligente utilizando arduino yun basado en internet de las cosas (IoT)* [en línea]. (tesis) (pregrado). UNIVERSIDAD POLITECNICA SALESIANA. 2016. [Consulta: 14 noviembre 2018]. Disponible en: https://dspace.ups.edu.ec/bitstream/123456789/13461/1/UPS-GT001798.pdf.

ROSE, K., et al, *La Internet De Las Cosas — Una Breve Reseña. Internet Society-ISOC* [en línea], 2015. [Consulta: 30 septiembre 2018]. ISSN 20292341. DOI http://dx.doi.org/10.3846/mla.2010.041. Disponible en: https://www.internetsociety.org/wp-content/uploads/2017/09/report-InternetOfThings-20160817-es-1.pdf.

SOFTPOWERGROUP, *Android Arduino Control: Android and Arduino IoT Control Devices with ThingSpeak.* [en línea]. 2018. [Consulta: 5 noviembre 2018]. Disponible en: http://androidcontrol.blogspot.com/2015/06/android-iot-control-thingspeak.html.

TAPIA, C. y MANZANO, H., *Evaluacion de la plataforma arduino e implementacion de un sistema de control de posicion horizontal* [en línea]. (tesis) (pregrado). UNIVERSIDAD POLITECNICA SALESIANA. 2013. [Consulta: 5 noviembre 2018]. Disponible en: https://dspace.ups.edu.ec/bitstream/123456789/5522/1/UPS-GT000511.pdf.

TEMBO, *What is Temboo? - Platform and Features.* [en línea]. 2018. [Consulta: 5 noviembre 2018]. Disponible en: https://temboo.com/platform.

TOAZO, G. y VEGA, M., *Prototipo de una gaveta experimental basada en Internet de las Cosas* [en línea]. (tesis) (pregrado). ESCUELA POLITECNICA NACIONAL. 2017. [Consulta: 8 noviembre 2018]. Disponible en: http://bibdigital.epn.edu.ec/bitstream/15000/17519/1/CD-8024.pdf.

TSCHOFENIG, H. et al., *Architectural Considerations in Smart Object Networking* [en línea]. 2015. [Consulta: 4 octubre 2018]. 2015. Disponible en: http://trustee.ietf.org/license-info.

UBIDOTS, *IoT platform | Internet of Things | Ubidots.* [en línea]. 2018. [Consulta: 5 noviembre 2018]. Disponible en: https://ubidots.com/.

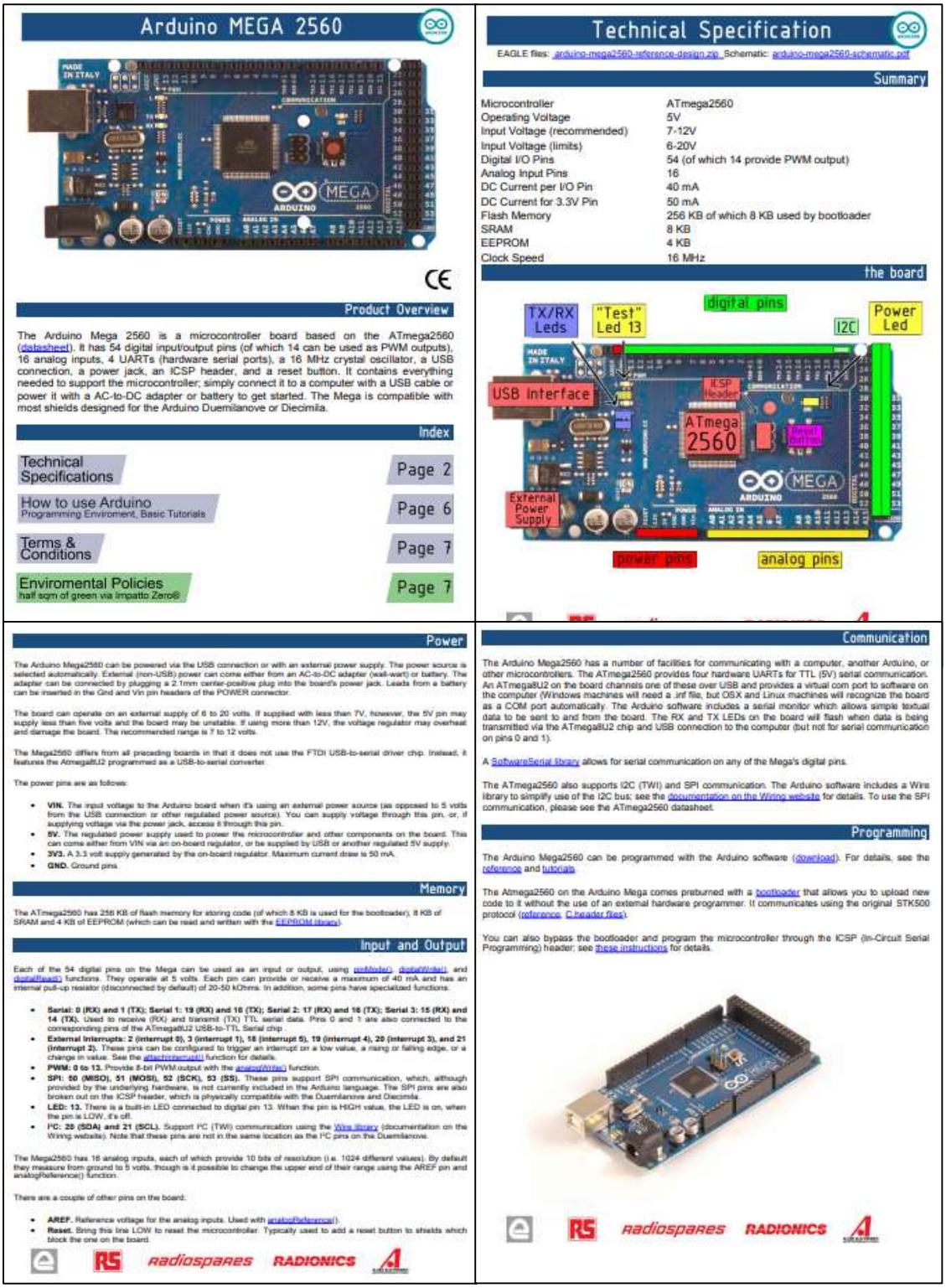
VALENTÍN, M., *Sistema de Gestión Urbana Inteligente del Aparcamiento (Si-GUIA).* [en línea], (tesis) (pregrado). 2016. [Consulta: 5 julio 2018]. Disponible en: https://reunir.unir.net/bitstream/handle/123456789/4373/VALENTIN ZAERA%2C MERCEDES.pdf?sequence=1.

WORDREFERENCE, *Que es un smartphone - Definición de smartphone.* [en línea]. 2009. [Consulta: 5 noviembre 2018]. Disponible en: https://www.quees.info/que-es-un-smartphone.html.

YUAN, M., *¿Por qué MQTT es uno de los mejores protocolos de red para Internet de las Cosas?* [en línea]. 2017. [Consulta: 4 agosto 2018]. 2017. Disponible en: https://www.ibm.com/developerworks/ssa/library/iot-mqtt-why-good-for-iot/iot-mqtt-why-good-for-iot-pdf.pdf.

ANEXOS

Anexos A: Hojas técnicas de los dispositivos del sistema de parqueo



Arduino MEGA 2560

Product Overview

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

Index

Technical Specification

EAGLE files: arduino-mega2560-reference-design.zip Schematic: arduino-mega2560-schematic.pdf

Summary

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

the board

Power

The Arduino Mega2560 can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions.

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 0 to 13.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Duemilanove and Diecimila.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- **I²C: 20 (SDA) and 21 (SCL).** Support I²C (TWI) communication using the Wire library (documentation on the Wiring website). Note that these pins are not in the same location as the I²C pins on the Duemilanove.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Mega's digital pins.

The ATmega2560 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation on the Wiring website for details. To use the SPI communication, please see the ATmega2560 datasheet.

Programming

The Arduino Mega2560 can be programmed with the Arduino software (download). For details, see the reference and tutorials.

The Atmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

Automatic (Software) Reset

Rather then requiring a physical press of the reset button before an upload, the Arduino Mega2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Mega2560 is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Mega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Mega contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Mega has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics and Shield Compatibility

The maximum length and width of the Mega PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

The Mega is designed to be compatible with most shields designed for the Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega and Duemilanove / Diecimila. **Please note that I²C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).**

How to use Arduino

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, MaxMSP).

Arduino is a cross-platoform program. You'll have to follow different instructions for your personal OS. Check on the Arduino site for the latest instructions. *http://arduino.cc/en/Guide/HomePage*

Linux Install **Windows Install** **Mac Install**

Once you have downloaded/unzipped the arduino IDE, you can Plug the Arduino to your PC via USB cable.

Blink led

Now you're actually ready to "burn" your first program on the arduino board. To select "blink led", the physical translation of the well known programming "hello world", select

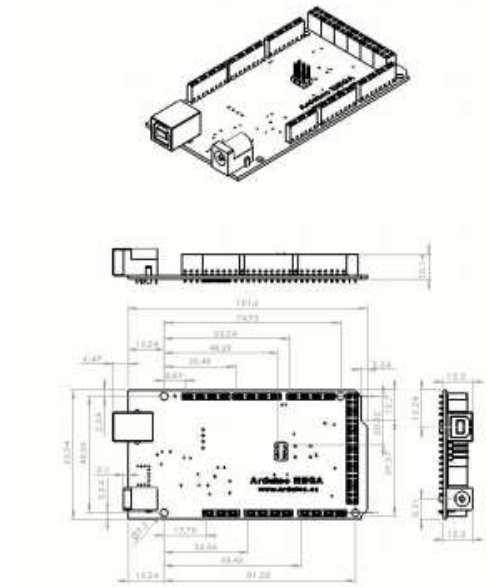
File>Sketchbook>
Arduino-0017>Examples>
Digital>Blink

Once you have your skecth you'll see something very close to the screenshot on the right.

In **Tools>Board** select MEGA

Now you have to go to
Tools>SerialPort
and select the right serial port, the one arduino is attached to.

Dimensioned Drawing



Terms & Conditions

1. Warranties

1.1 The producer warrants that its products will conform to the Specifications. This warranty lasts for one (1) years from the date of the sale. The producer shall not be liable for any defects that are caused by neglect, misuse or mistreatment by the Customer, including improper installation or testing, or for any products that have been altered or modified in any way by a Customer. Moreover, The producer shall not be liable for any defects that result from Customer's design, specifications or instructions for such products. Testing and other quality control techniques are used to the extent the producer deems necessary.

1.2 If any products fail to conform to the warranty set forth above, the producer's sole liability shall be to replace such products. The producer's liability shall be limited to products that are determined by the producer not to conform to such warranty. If the producer elects to replace such products, the producer shall have a reasonable time to replacements. Replaced products shall be warranted for a new full warranty period.

1.3 EXCEPT AS SET FORTH ABOVE, PRODUCTS ARE PROVIDED "AS IS" AND "WITH ALL FAULTS." THE PRODUCER DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

1.4 Customer agrees that prior to using any systems that include the producer products, Customer will test such systems and the functionality of the products as used in such systems. The producer may provide technical, applications or design advice, quality characterization, reliability data or other services. Customer acknowledges and agrees that providing these services shall not expand or otherwise alter the producer's warranties, as set forth above, and no additional obligations or liabilities shall arise from the producer providing such services.

1.5 The Arduino™ products are not authorized for use in safety-critical applications where a failure of the product would reasonably be expected to cause severe personal injury or death. Safety-Critical Applications include, without limitation, life support devices and systems, equipment or systems for the operation of nuclear facilities and weapons systems. Arduino™ products are neither designed nor intended for use in military or aerospace applications or environments and for automotive applications or environment. Customer acknowledges and agrees that any such use of Arduino™ products which is solely at the Customer's risk, and that Customer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

1.6 Customer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products and any use of Arduino™ products in Customer's applications, notwithstanding any applications-related information or support that may be provided by the producer.

2. Indemnification

The Customer acknowledges and agrees to defend, indemnify and hold harmless the producer from and against any and all third-party losses, damages, liabilities and expenses it incurs to the extent directly caused by: (i) an actual breach by a Customer of the representation and warranties made under this terms and conditions or (ii) the gross negligence or willful misconduct by the Customer.

3. Consequential Damages Waiver

In no event the producer shall be liable to the Customer or any third parties for any special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of the products provided hereunder, regardless of whether the producer has been advised of the possibility of such damages. This section will survive the termination of the warranty period.

4. Changes to specifications

The producer may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." The producer reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.

Enviromental Policies

The producer of Arduino™ has joined the Impatto Zero® policy of LifeGate.it. For each Arduino board produced is created / looked after half squared Km of Costa Rica's forest's.

ESPRESSIF SMART CONNECTIVITY PLATFORM: ESP8266

Table of Contents

1 Introduction

Espressif Systems' Smart Connectivity Platform (ESCP) of high performance wireless SOCs, for mobile platform designers, provides unsurpassed ability to embed Wi-Fi capabilities within other systems, at the lowest cost with the greatest functionality.

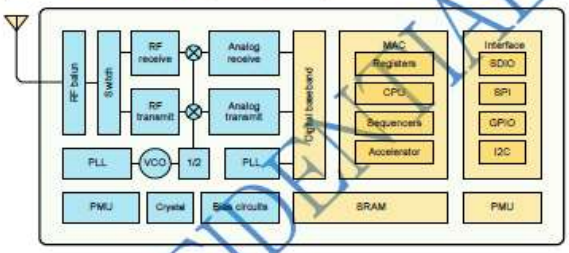


Figure 1: ESP8266 Block Diagram

2 Technology Overview

ESP8266 offers a complete and self-contained Wi-Fi networking solution, allowing it to either host the application or to offload all Wi-Fi networking functions from another application processor.

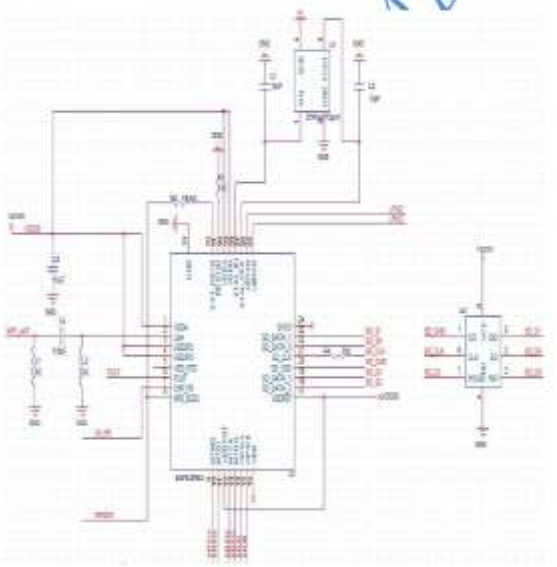
When ESP8266 hosts the application, and when it is the only application processor in the device, it is able to boot up directly from an external flash. It has integrated cache to improve the performance of the system in such applications, and to minimize the memory requirements.

Alternately, serving as a Wi-Fi adapter, wireless internet access can be added to any microcontroller-based design with simple connectivity through UART interface or the CPU AHB bridge interface.

ESP8266 on-board processing and storage capabilities allow it to be integrated with the sensors and other application specific devices through its GPIOs with minimal development up-front and minimal loading during runtime. With its high degree of on-chip integration, which includes the antenna switch balun, power management converters, it requires minimal external circuitry, and the entire solution, including front-end module, is designed to occupy minimal PCB area.

Sophisticated system-level features include fast sleep/wake context switching for energy-efficient VoIP, adaptive radio biasing for low-power operation, advance signal processing, and spur cancellation and radio co-existence features for common cellular, Bluetooth, DDR, LVDS, LCD interference mitigation.

4 Application Diagram



5 Ultra Low Power Technology

ESP8266 has been designed for mobile, wearable electronics and Internet of Things applications with the aim of achieving the lowest power consumption with a combination of several proprietary techniques. The power saving architecture operates in 3 modes: active mode, sleep mode and deep sleep mode.

By using advance power management techniques and logic to power-down functions not required and to control switching between sleep and active modes, ESP8266 consumes less than 12uA in sleep mode and less than 1.0mW (DTIM=3) or less than 0.5mW (DTIM=10) to stay connected to the access point.

When in sleep mode, only the calibrated real-time clock and watchdog remains active. The real-time clock can be programmed to wake up the ESP8266 at any required interval.

The ESP8266 can be programmed to wake up when a specified condition is detected. This minimal wake-up time feature of the ESP8266 can be utilized by mobile device SOCs, allowing them to remain in the low-power standby mode until Wi-Fi is needed.

In order to satisfy the power demand of mobile and wearable electronics, ESP8266 can be programmed to reduce the output power of the PA to fit various application profiles, by trading off range for power consumption.

5.1 Highest Level of Integration

By integrating the costliest components such as power management unit, TR switch, RF balun, high power PA capable of delivering +25dBm (peak), ESP8266 ensures that the BOM cost is the lowest possible, and ease of integration into any system.

With ESP8266, the only external BOM are resistors, capacitors, and crystal.

6 ESP8266 Applications

- Smart power plugs
- Home automation
- Mesh network
- Industrial wireless control
- Baby monitors
- IP Cameras
- Sensor networks
- Wearable electronics
- Wi-Fi location-aware devices
- Security ID tags
- Wi-Fi position system beacons

7 Specifications

7.1 Current Consumption

The following current consumption is based on 3.3V supply, and 25°C ambient, using internal regulators. Measurements are done at antenna port without SAW filter. All the transmitter's measurements are based on 90% duty cycle, continuous transmit mode.

Mode	Min	Typ	Max	Unit
Transmit 802.11b, CCK 1Mbps, P_{OUT}=+19.5dBm		215		mA
Transmit 802.11b, CCK 11Mbps, P_{OUT}=+18.5dBm		197		mA
Transmit 802.11g, OFDM 54Mbps, P_{OUT} =+16dBm		145		mA
Transmit 802.11n, MCS7, P_{OUT}=+14dBm		135		mA
Receive 802.11b, packet length=1024 byte, -80dBm		60		mA
Receive 802.11g, packet length=1024 byte, -70dBm		60		mA
Receive 802.11n, packet length=1024 byte, -65dBm		62		mA
Standby		0.9		mA
Deep sleep		10		uA
Power save mode DTIM 1		1.2		mA
Power save mode DTIM 3		0.86		mA
Total shutdown		0.5		uA

7.2 RF Performance

The following are measured under room temperature conditions with 3.3V and 1.1V power supplies.

Description	Min	Typical	Max	Unit
Input frequency	2412		2484	MHz
Input impedance		50		Ω
Input reflection			-10	dB
Output power of PA for 72.2Mbps	14	15	16	dBm
Output power of PA for 11b mode	17.5	18.5	19.5	dBm
Sensitivity				
CCK, 1Mbps		-98		dBm
CCK, 11Mbps		-91		dBm
6Mbps (1/2 BPSK)		-93		dBm
54Mbps (3/4 64-QAM)		-75		dBm
HT20, MCS7 (65Mbps, 72.2Mbps)		-71		dBm
Adjacent Channel Rejection				
OFDM, 6Mbps		37		dB
OFDM, 54Mbps		21		dB
HT20, MCS0		37		dB
HT20, MCS7		20		dB

8 CPU, Memory and Interfaces

8.1 CPU

This chip embeds an ultra low power Micro 32-bit CPU, with 16-bit thumb mode. This CPU can be interfaced using:

- code RAM/ROM interface (iBus) that goes to the memory controller, that can also be used to access external flash memory,
- data RAM interface (dBus), that also goes to the memory controller
- AHB interface, for register access, and
- JTAG interface for debugging

8.2 Memory Controller

The memory controller contains ROM, and SRAM. It is accessed by the CPU using the iBus, dBus and AHB interface. Any of these interfaces can request access to the ROM or RAM modules, and the memory controller arbiters serve these 3 interfaces on a first-come-first-serve basis.

8.3 AHB and AHB Blocks

The AHB blocks performs the function of an arbiter, controls the AHB interfaces from the MAC, SDIO (host) and CPU. Depending on the address, the AHB data requests can go into one of the two slaves:

- APB block, or
- flash controller (usually for standalone applications).

Data requests to the memory controller are usually high speed requests, and requests to the APB block are usually register access.

The APB block acts as a decoder. It is meant only for access to programmable registers within ESP8266's main blocks. Depending on the address, the APB request can go to the radio, SI/SPI, SDIO (host), GPIO, UART, real-time clock (RTC), MAC or digital baseband.

8.4 Interfaces

The ESP8266 contains several analog and digital interfaces described in the following sections.

8.4.1 Master SI / SPI Control (Optional)

The master serial interface (SI) can operate in two, three or four-wire bus configurations to control the EEPROM or other I2C/SPI devices. Multiple I2C devices with different device addresses are supported by sharing the 2-wire bus.

Multiple SPI devices are supported by sharing the clock and data signals, using separate software controlled GPIO pins as chip selects.

The SPI can be used for controlling external devices such as serial flash memories, audio CODECs, or other slave devices. It is set up as a standard master SPI device with 3 different enable pins:

- SPI_EN0,
- SPI_EN1,
- SPI_EN2.

Both SPI master and SPI slave are supported with the latter being used as a host interface.

SPI_EN0 is used as an enable signal to an external serial flash memory for downloading patch code and/or MIB-data to the baseband in an embedded application. In a host based application, patch code and MIB-data can alternatively be downloaded via the host interface. This pin is active low and should be left open if not used.

SPI_EN1 is usually used for a user application, e.g. to control an external audio codec or sensor ADC, in an embedded application. This pin is active low and should be left open if not used.

SPI_EN2 usually controls an EEPROM to store individual data, such as MIB information, MAC address, and calibration data, or for general use. This pin is active low and should be left open if not used.

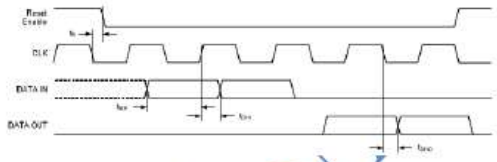


Figure 2: SPI timing characteristics

8.4.2 General Purpose IO

There are up to 16 GPIO pins. They can be assigned to various functions by the firmware. Each GPIO can be configured with internal pull-up/down, input available for sampling by a software register, input triggering an edge or level CPU interrupt, input triggering a level wakeup interrupt, open-drain or push-pull output driver, or output source from a software register, or a sigma-delta PWM DAC.

These pins are multiplexed with other functions such as host interface, UART, SI, Bluetooth coexistence, etc.

8.4.3 Digital IO Pads

The digital IO pads are bidirectional, non-inverting and tri-state. It includes input and an output buffer with tristate control inputs. Besides this, for low power operations, the IO can also be set to hold. For instance, when we power down the chip, all output enable signals can be set to hold low.

Optional hold functionality can be built into the IO if requested. When the IO is not driven by the internal or external circuitry, the hold functionality can be used to hold the state to the last used state.

The hold functionality introduces some positive feedback into the pad. Hence, the external driver that drives the pad must be stronger than the positive feedback. The required drive strength is however small – in the range of 5uA.

Parameter	Symbol	Min	Max	Unit
Input low voltage	V_{IL}	-0.3	$0.25 \times V_{IO}$	V
Input high voltage	V_{IH}	0.75×VIO	3.6	V
Input leakage current	I_{IL}		50	nA
Output low voltage	V_{OL}		$0.1 \times V_{IO}$	V
Ourput high voltage	V_{OH}	0.8×VIO		V
Input pin capacitance	C_{pad}		2	pF
VDDIO	V_{IO}	1.7	3.6	V
Maximum drive capability	I_{MAX}		12	mA
Temperature	T_{amb}	-20	100	°C

All digital IO pins are protected from over-voltage with a snap-back circuit connected between the pad and ground. The snap back voltage is typically about 6V, and the holding voltage is 5.8V. This provides protection from over-voltages and ESD. The output devices are also protected from reversed voltages with diodes.

9 Firmware & Software Development Kit

The application and firmware is executed in on-chip ROM and SRAM, which loads the instructions during wake-up, through the SDIO interface, from the external flash.

The firmware implements TCP/IP, the full 802.11 b/g/n/e/i WLAN MAC protocol and Wi-Fi Direct specification. It supports not only basic service set (BSS) operations under the distributed control function (DCF) but also P2P group operation compliant with the latest Wi-Fi P2P protocol. Low level protocol functions are handled automatically by ESP8266:

- RTS/CTS,
- acknowledgement,
- fragmentation and defragmentation,
- aggregation,
- frame encapsulation (802.11h/RFC 1042),
- automatic beacon monitoring / scanning, and
- P2P Wi-Fi direct,

Passive or active scanning, as well as P2P discovery procedure is performed autonomously once initiated by the appropriate command. Power management is handled with minimum host interaction to minimize active duty period.

9.1 Features

The SDK includes the following library functions:

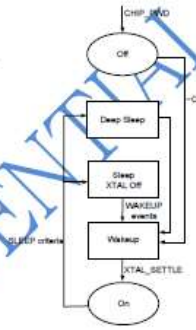
- 802.11 b/g/n/d/e/i/k/r support;
- Wi-Fi Direct (P2P) support:
- P2P Discovery, P2P Group Owner mode, P2P Power Management
- Infrastructure BSS Station mode / P2P mode / softAP mode support;
- Hardware accelerators for CCMP (CBC-MAC, counter mode), TKIP (MIC, RC4), WAPI (SMS4), WEP (RC4), CRC;

- WPA/WPA2 PSK, and WPS driver;
- Additional 802.11i security features such as pre-authentication, and TSN;
- Open Interface for various upper layer authentication schemes over EAP such as TLS, PEAP, LEAP, SIM, AKA, or customer specific;
- 802.11n support (2.4GHz / 5GHz);
- Supports MIMO 1×1 and 2×1, STBC, A-MPDU and A-MSDU aggregation and 0.4μs guard interval;
- WMM power save U-APSD;
- Multiple queue management to fully utilize traffic prioritization defined by 802.11e standard;
- UMA compliant and certified;
- 802.1h/RFC1042 frame encapsulation;
- Scattered DMA for optimal CPU off load on Zero Copy data transfer operations;
- Antenna diversity and selection (software managed hardware);
- Clock/power gating combined with 802.11-compliant power management dynamically adapted to curent connection condition providing minimal power consumption;
- Adaptive rate fallback algorithm sets the optimium transmission rate and Tx power based on actual SNR and packet loss information;
- Automatic retransmission and response on MAC to avoid packet discarding on slow host environment;
- Seamless roaming support;
- Configurable packet traffic arbitration (PTA) with dedicated slave processor based design provides flexible and exact timing Bluetooth co-existence supoport for a wide range of Bluetooth Chip vendors;
- Dual and single antenna Bluetooth co-existence support with optional simultaneous receive (Wi-Fi/Bluetooth) capability.

10 Power Management

The chip can be put into the following states:

- **OFF**: CHIP_PD pin is low. The RTC is disabled. All registers are cleared.
- **DEEP_SLEEP**: Only RTC is powered on – the rest of the chip is powered off. Recovery memory of RTC can keep basic Wi-Fi connecting information.
- **SLEEP**: Only the RTC is operating. The crystal oscillator is disabled. Any wakeup events (MAC, host, RTC timer, external interrupts) will put the chip into the WAKEUP state.
- **WAKEUP**: In this state, the system goes from the sleep states to the PWR state. The crystal oscillator and PLLs are enabled.
- **ON state**: the high speed clock is operational and sent to each block enabled by the clock control register. Lower level clock gating is implemented at the block level, including the CPU, which can be gated off using the WAITI instruction, while the system is on.



11 Clock Management

11.1 High Frequency Clock

The high frequency clock on ESP8266 is used to drive both the Tx and Rx mixers. This clock is generated from the internal crystal oscillator and an external crystal. The crystal frequency can range from 26MHz to 52MHz.

While internal calibration of the crystal oscillator ensures that a wide range of crystals can be used, in general, the quality of the crystal is still a factor to consider, to obtain reasonable phase noise. When the crystal selected is sub-optimal due to large frequency drifts or poor Q-factor, the maximum throughput and sensitivity of the Wi-Fi system is degraded. Please refer to the application notes on how the frequency offset can be measured.

Parameter	Symbol	Min	Max	Unit
Frequency	F_{XO}	26	52	MHz
Loading capacitance	C_L		32	pF
Motional capacitance	C_M	2	5	pF
Series resistance	R_S	0	65	Ω
Frequency tolerance	ΔF_{XO}	-15	15	ppm
Frequency vs temperature (-25°C ~ 75°C)	ΔF_{XO_Temp}	-15	15	ppm

11.2 External Reference Requirements

For an externally generated clock, the frequency can range from 26MHz to 52MHz can be used. For good performance of the radio, the following characteristics are expected of the clock:

Parameter	Symbol	Min	Max	Unit
Clock amplitude	V_{XO}	0.2	1	Vpp
External clock accuracy	ΔF_{XO_EXT}	-15	15	ppm
Phase noise @1kHz offset, 40MHz clock			-129	dBc/Hz
Phase noise @10kHz offset, 40MHz clock			-130	dBc/Hz
Phase noise @100kHz offset, 40MHz clock			-138	dBc/Hz

12 Radio

The ESP8266 radio consists of the following main blocks:

- 2.4GHz receiver
- 2.4GHz transmitter
- High speed clock generators and crystal oscillator
- Real time clock
- Bias and regulators
- Power management

12.1 Channel Frequencies

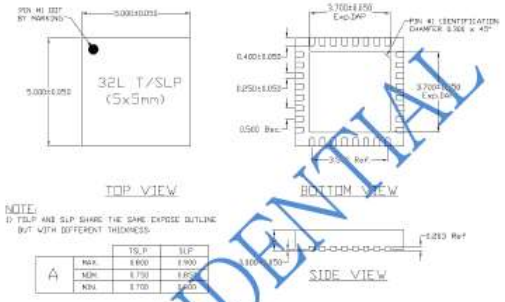
The RF transceiver supports the following channels according to the IEEE802.11bgn standards.

Channel No	Frequency (MHz)	Channel No	Frequency (MHz)
1	2412	8	2447
2	2417	9	2452
3	2422	10	2457
4	2427	11	2462
5	2432	12	2467
6	2437	13	2472
7	2442	14	2484

12.2 2.4GHz Receiver

The 2.4GHz receiver downconverts the RF signal to quadrature baseband signals and converts them to the digital domain with 2 high resolution high speed ADCs. To adapt to varying signal channel conditions, RF filters, automatic gain control, DC offset cancelation circuits and baseband filters are integrated within the radio.

App. QFN32 Package Drawing



12.3 2.4GHz Transmitter

The 2.4GHz transmitter upconverts the quadrature baseband signals to 2.4GHz, and drives the antenna with a high powered CMOS power amplifier. The use of digital calibration further improves the linearity of the power amplifier, enabling a state of art performance of delivering +19dBm average power for 802.11b transmission and +16dBm for 802.11n transmission.

Additional calibrations are integrated to cancel any imperfections of the radio, such as:

- carrier leakage,
- I/Q phase matching, and
- baseband nonlinearities

This reduces the amount of time required and test equipment required for production testing.

12.4 Clock Generator

The clock generator generates quadrature 2.4GHz clock signals for the receiver and transmitter. All components of the clock generator are integrated on-chip, including:

- inductor,
- varactor, and
- loop filter.

The clock generator has built-in calibration and self test circuits. Quadrature clock phases and phase noise are optimized on-chip with patented calibration algorithms to ensure the best receiver and transmitter performance.

HC-SR04 Ultrasonic Sensor

Elijah J. Morgan

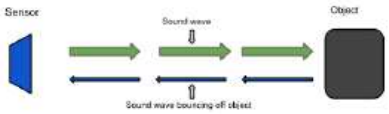
Nov. 16 2014

The purpose of this file is to explain how the HC-SR04 works. It will give a brief explanation of how ultrasonic sensors work in general. It will also explain how to wire the sensor up to a microcontroller and how to take/interpret readings. It will also discuss some sources of errors and bad readings.

1. How Ultrasonic Sensors Work
2. HC-SR04 Specifications
3. Timing chart, Pin explanations and Taking Distance Measurements
4. Wiring HC-SR04 with a microcontroller
5. Errors and Bad Readings

1. How Ultrasonic Sensors Work

Ultrasonic sensors use sound to determine the distance between the sensor and the closest object in its path. How do ultrasonic sensors do this? Ultrasonic sensors are essentially sound sensors, but they operate at a frequency above human hearing.



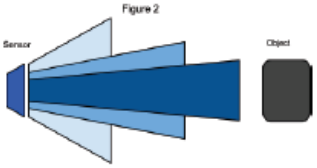
The sensor sends out a sound wave at a specific frequency. It then listens for that specific sound wave to bounce off of an object and come back (Figure 1). The sensor keeps track of the time between sending the sound wave and the sound wave returning. If you know how fast something is going and how long it is traveling you can find the distance traveled with equation 1.

Equation 1. $d = v \times t$

The speed of sound can be calculated based on the a variety of atmospheric conditions, including temperature, humidity and pressure. Actually calculating the distance will be shown later on in this document.

It should be noted that ultrasonic sensors have a cone of detection, the angle of this cone varies with distance, Figure 2 show this relation. The ability of a sensor to detect an object also depends on the objects orientation to the sensor. If an object doesn't present a flat surface to the sensor then it is possible the sound wave will bounce off the object in a way that it does not return to the sensor.

Figure 2



2. HC-SR04 Specifications

The sensor chosen for the Firefighting Drone Project was the HC-SR04. This section contains the specifications and why they are important to the sensor module. The sensor modules requirements are as follows.

- Cost
- Weight
- Community of hobbyists and support
- Accuracy of object detection
- Probability of working in a smoky environment
- Ease of use

The HC-SR04 Specifications are listed below. These specifications are from the Cytron Technologies HC-SR04 User's Manual (source 1).

- Power Supply: +5V DC
- Quiescent Current: <2mA
- Working current: 15mA
- Effectual Angle: <15°
- Ranging Distance: 2-400 cm
- Resolution: 0.3 cm
- Measuring Angle: 30°
- Trigger Input Pulse width: 10uS
- Dimension: 45mm x 20mm x 15mm
- Weight: approx. 10 g

The HC-SR04's best selling point is its price; it can be purchased at around $2 per unit.

3. Timing Chart and Pin Explanations

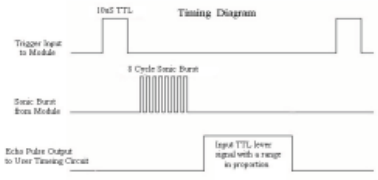
The HC-SR04 has four pins, VCC, GND, TRIG and ECHO; these pins all have different functions. The VCC and GND pins are the simplest -- they power the HC-SR04. These pins need to be attached to a +5 volt source and ground respectively. There is a single control pin: the TRIG pin. The TRIG pin is responsible for sending the ultrasonic burst. This pin should be set to HIGH for 10 µs, at which point the HC-SR04 will send out an eight cycle sonic burst at 40 kHZ. After a sonic burst has been sent the ECHO pin will go HIGH. The ECHO pin is the data pin -- it is used in taking distance measurements. After an ultrasonic burst is sent the pin will go HIGH, it will stay high until an ultrasonic burst is detected back, at which point it will go LOW.

Taking Distance Measurements

The HC-SR04 can be triggered to send out an ultrasonic burst by setting the TRIG pin to HIGH. Once the burst is sent the ECHO pin will automatically go HIGH. This pin will remain HIGH until the the burst hits the sensor again. You can calculate the distance to the object by keeping track of how long the ECHO pin stays HIGH. The time ECHO stays HIGH is the time the burst spent traveling. Using this measurement in equation 1 along with the speed of sound will yield the distance travelled. A summary of this is listed below, along with a visual representation in Figure 2.

1. Set TRIG to HIGH
2. Set a timer when ECHO goes to HIGH
3. Keep the timer running until ECHO goes to LOW
4. Save that time
5. Use equation 1 to determine the distance travelled

Figure 3
Source 2



Source 2

To interpret the time reading into a distance you need to change equation 1. The clock on the device you are using will probably count in microseconds or smaller. To use equation 1 the speed of sound needs to determined,which is 343 meters per second at standard temperature and pressure. To convert this into more useful form use equation 2 to change from meters per second to microseconds per centimeter. Then equation 3 can be used to easily compute the distance in centimeters.

Equation 2. $Distance = \frac{Speed}{170.15\,m} \times \frac{Meters}{100\,cm} \times \frac{1e6\,\mu S}{170.15\,m} \times \frac{58.772\,\mu S}{cm}$

Equation 3. $Distance = \frac{time}{58} = \frac{\mu s}{\mu s/cm} = cm$

4. Wiring the HC-SR04 to a Microcontroller

This section only covers the hardware side. For information on how to integrate the software side, look at one of the links below or look into the specific microcontroller you are using.

The HC-SR04 has 4 pins: VCC, GND, TRIG and ECHO.

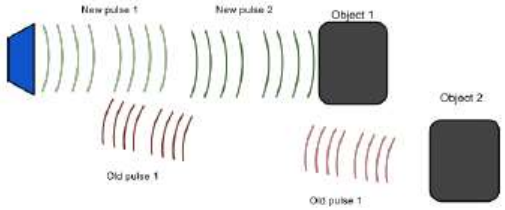
1. VCC is a 5v power supply. This should come from the microcontroller
2. GND is a ground pin. Attach to ground on the microcontroller.
3. TRIG should be attached to a GPIO pin that can be set to HIGH
4. ECHO is a little more difficult. The HC-SR04 outputs 5v, which could destroy many microcontroller GPIO pins (the maximum allowed voltage varies). In order to step down the voltage use a single resistor or a voltage divider circuit. Once again this depends on the specific microcontroller you are using, you will need to find out its GPIO maximum voltage and make sure you are below that.

Figure 4

5. Errors and Bad Readings

Ultrasonic sensors are great sensors -- they work well for many applications where other types of sensors fall short. Unfortunately, they do have weaknesses. These weaknesses can be mitigated and worked around, but first they must be understood. The

first weakness is that they use sound. There is a limit to how fast ultrasonic sensors can get distance measurements. The longer the distance, the slower they are at reporting the distance. The second weakness comes from the way sound bounces off of objects. In enclosed spaces it is possible, if not probable that there will be unintended echos. The echos can very easily cause false short readings. In Figure 2 a pulse was sent out. It bounced off of object 1 and returned to the sensor. The distance was recorded and then a new pulse was sent. There was another object farther away, so that when the new pulse reaches object 1, the first signal will reach the sensor. This will cause the sensor to think that there is an object closer than is actually true. The old pulse is smaller than the new pulse because it has grown weaker. The longer the pulse exists the weaker it grows until it is negligible. If multiple sensors are being used, the number of echos will increase along with the number of errors. There are two main ways to reduce the number of errors. The first is to provide shielding around the sensor. This prevents echos coming in from angle outside what the sensor should actually pick up. The second is to reduce the frequency at which pulses are sent out. This gives more time for the echos to dissipate.



Works Cited

Source 1.
"HC-SR04 User's_Manual." *docs.google*. Cytron Technologies, May 2013 Web. 5 Dec. 2009.
<https://docs.google.com/document/d/1Y-yZnNhMYy7rwhAgyL_pfa39RsB-x2qR4vP8saG73rE/edit>

Source 2.
"Attiny2313 Ultrasonic distance (HR-SR04) example." *CircuitDB*. n.a. 7 Sept. 2014 Web. 5 Dec. 2014. <http://www.circuitdb.com/?p=1162>

Links

These are not formatted; you will need to copy and paste them into your web browser.

Want to learn about Ultrasonic Sensors in general?
http://www.sensorsmag.com/sensors/acoustic-ultrasound/choosing-ultrasonic-sensor-proximity-or-distance-measurement-825

All about the HC-SR04

- http://www.circuitdb.com/?p=1162
- http://www.micropik.com/PDF/HCSR04.pdf
- http://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/
- http://www.ezdenki.com/ultrasonic.php
  (^fantastic tutorial, explains a lot of stuff)
- http://www.elecrow.com/hcsr04-ultrasonic-ranging-sensor-p-316.html
  (^ this one has some cool charts)

HC-05

-Bluetooth to Serial Port Module

Overview

HC-05 module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup.
Serial port Bluetooth module is fully qualified Bluetooth V2.0+EDR (Enhanced Data Rate) 3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR Bluecore 04-External single chip Bluetooth system with CMOS technology and with AFH(Adaptive Frequency Hopping Feature). It has the footprint as small as 12.7mmx27mm. Hope it will simplify your overall design/development cycle.

Specifications

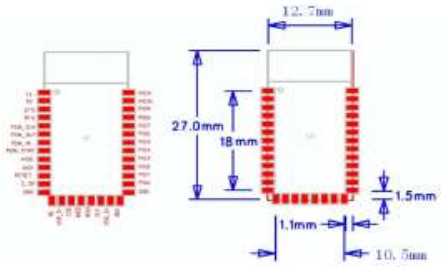
Hardware features

- Typical -80dBm sensitivity
- Up to +4dBm RF transmit power
- Low Power 1.8V Operation ,1.8 to 3.6V I/O
- PIO control
- UART interface with programmable baud rate
- With integrated antenna
- With edge connector

Software features

- Default Baud rate: 38400, Data bits:8, Stop bit:1,Parity:No parity, Data control: has. Supported baud rate: 9600,19200,38400,57600,115200,230400,460800.
- Given a rising pulse in PIO0, device will be disconnected.
- Status instruction port PIO1: low-disconnected, high-connected;
- PIO10 and PIO11 can be connected to red and blue led separately. When master and slave are paired, red and blue led blinks 1time/2s in interval, while disconnected only blue led blinks 2times/s.
- Auto-connect to the last device on power as default.
- Permit pairing device to connect as default.
- Auto-pairing PINCODE:"0000" as default
- Auto-reconnect in 30 min when disconnected as a result of beyond the range of connection.

Hardware



PIN Name	PIN #	Pad type	Description	Note
GND	13 21 22	VSS	Ground pot	
3.3 VCC	12	3.3V	Integrated 3.3V (+) supply with On-chip linear regulator output within 3.15-3.3V	
AIO0	9	Bi-Directional	Programmable input/output line	
AIO1	10	Bi-Directional	Programmable input/output line	
PIO0	23	Bi-Directional RX EN	Programmable input/output line, control output for LNA(if fitted)	
PIO1	24	Bi-Directional TX EN	Programmable input/output line, control output for PA(if fitted)	
PIO2	25	Bi-Directional	Programmable input/output line	
PIO3	26	Bi-Directional	Programmable input/output line	
PIO4	27	Bi-Directional	Programmable input/output line	
PIO5	28	Bi-Directional	Programmable input/output line	
PIO6	29	Bi-Directional	Programmable input/output line	
PIO7	30	Bi-Directional	Programmable input/output line	
PIO8	31	Bi-Directional	Programmable input/output line	
PIO9	32	Bi-Directional	Programmable input/output line	
PIO10	33	Bi-Directional	Programmable input/output line	
PIO11	34	Bi-Directional	Programmable input/output line	

RESETB	11	CMOS input with weak internal pull-up	Reset if low .input debounced so must be low for >5MS to cause a reset	
UART_RTS	4	CMOS output, tri-stable with weak internal pull-up	UART request to send, active low	
UART_CTS	3	CMOS input with weak internal pull-down	UART clear to send, active low	
UART_RX	2	CMOS input with weak internal pull-down	UART Data input	
UART_TX	1	CMOS output, Tri-stable with weak internal pull-up	UART Data output	
SPI_MOSI	17	CMOS input with weak internal pull-down	Serial peripheral interface data input	
SPI_CSB	16	CMOS input with weak internal pull-up	Chip select for serial peripheral interface, active low	
SPI_CLK	19	CMOS input with weak internal pull-down	Serial peripheral interface clock	
SPI_MISO	18	CMOS input with weak internal pull-down	Serial peripheral interface data Output	
USB_-	15	Bi-Directional		

USB_+	20	Bi-Directional		
NC	14			
PCM_CLK	5	Bi-Directional	Synchronous PCM data clock	
PCM_OUT	6	CMOS output	Synchronous PCM data output	
PCM_IN	7	CMOS Input	Synchronous PCM data input	
PCM_SYNC	8	Bi-Directional	Synchronous PCM data strobe	

AT command Default:

How to set the mode to server (master):
1. Connect PIO11 to high level.
2. Power on, module into command state.
3. Using baud rate 38400, sent the "AT+ROLE=1\r\n" to module, with "OK\r\n" means setting successes.
4. Connect the PIO11 to low level, repower the module, the module work as server (master).

AT commands: (all end with \r\n)

1. Test command:

Command	Respond	Parameter
AT	OK	-

2. Reset

Command	Respond	Parameter
AT+RESET	OK	-

3. Get firmware version

Command	Respond	Parameter
AT+VERSION?	+VERSION:<Param> OK	Param : firmware version

Example:
AT+VERSION?\r\n
+VERSION:2.0-20100601
OK

4. Restore default

Command	Respond	Parameter
AT+ORGL	OK	-

Default state:
Slave mode, pin code :1234, device name: H-C-2010-06-01 ,Baud 38400bits/s.

5. Get module address

Command	Respond	Parameter
AT+ADDR?	+ADDR:<Param> OK	Param: address of Bluetooth module

Bluetooth address: NAP: UAP : LAP
Example:
AT+ADDR?\r\n
+ADDR:1234:56:abcdef
OK

6. Set/Check module name:

Command	Respond	Parameter
AT+NAME=<Param>	OK	Param: Bluetooth module name (Default :HC-05)
AT+NAME?	+NAME:<Param> OK (/FAIL)	

Example:
AT+NAME=HC-05\r\n set the module name to "HC-05"
OK
AT+NAME=ITeadStudio\r\n
OK
AT+NAME?\r\n
+NAME: ITeadStudio
OK

7. Get the Bluetooth device name:

Command	Respond	Parameter
AT+RNAME?<Param1>	1. +NAME:<Param2> OK 2. FAIL	Param1,Param 2 : the address of Bluetooth device

Example: (Device address 00:02:72:cd:22:24, name：ITead)
AT+RNAME? 0002，72，cd2224\r\n
+RNAME:ITead
OK

8. Set/Check module mode:

Command	Respond	Parameter
AT+ROLE=<Param>	OK	Param: 0- Slave
AT+ ROLE?	+ROLE:<Param>	

	OK	1-Master 2-Slave-Loop

9. Set/Check device class

Command	Respond	Parameter
AT+CLASS=<Param>	OK	Param: Device Class
AT+ CLASS?	1. +CLASS:<Param> OK 2. FAIL	

10. Set/Check GIAC (General Inquire Access Code)

Command	Respond	Parameter
AT+IAC=<Param>	1.OK 2. FAIL	Param: GIAC (Default : 9e8b33)
AT+IAC	+IAC:<Param> OK	

Example:
AT+IAC=9e8b3f\r\n
OK
AT+IAC?\r\n
+IAC: 9e8b3f
OK

11. Set/Check – Query access patterns

Command	Respond	Parameter
AT+INQM=<Param>,<Param2>,<Param3>	1.OK 2. FAIL	Param: 0——inquiry_mode_standard 1——inquiry_mode_rssi Param2: Maximum number of Bluetooth devices to respond to Param3: Timeout (1-48 : 1.28s to 61.44s)
AT+ INQM?	+INQM： <Param>,<Param2>,<Param3> OK	

Example:
AT+INQM=1,9,48\r\n
OK
AT+INQM\r\n
+INQM:1, 9, 48
OK

12. Set/Check PIN code:

Command	Respond	Parameter
AT+PSWD=<Param>	OK	Param: PIN code (Default 1234)
AT+ PSWD?	+ PSWD ： <Param> OK	

13. Set/Check serial parameter:

Command	Respond	Parameter
AT+UART=<Param>,<Param2>,<Param3>	OK	Param1: Baud Param2: Stop bit Param3: Parity
AT+ UART?	+UART=<Param>,<Param2>,<Param3> OK	

Example:
AT+UART=115200，1,2,\r\n
OK
AT+UART?
+UART:115200,1,2
OK

14. Set/Check connect mode:

Command	Respond	Parameter
AT+CMODE=<Param>	OK	Param: 0 - connect fixed address 1 - connect any address 2 - slave-Loop
AT+ CMODE?	+ CMODE:<Param> OK	

15. Set/Check fixed address:

Command	Respond	Parameter
AT+BIND=<Param>	OK	Param: Fixed address (Default 00:00:00:00:00:00)
AT+ BIND?	+ BIND:<Param> OK	

Example:
AT+BIND=1234，56，abcdef\r\n
OK
AT+BIND?\r\n
+BIND:1234:56:abcdef
OK

16. Set/Check LED I/O

Command	Respond	Parameter
AT+POLAR=<Param1,<Param2>	OK	Param1: 0- PIO8 low drive LED 1- PIO8 high drive LED
AT+ POLAR?	+ POLAR=<Param1>,<Param2> OK	

		Param2: 0- PIO9 low drive LED 1- PIO9 high drive LED

17. Set PIO output

Command	Respond	Parameter
AT+PIO=<Param1>,<Param2>	OK	Param1: PIO number Param2: PIO level 0- low 1- high

Example:
1. PIO10 output high level
AT+PIO=10, 1\r\n
OK

18. Set/Check – scan parameter

Command	Respond	Parameter
AT+HPSCAN=<Param1>,<Param2>,<Param3>,<Param4>	OK	Param1: Query time interval Param2: Query duration Param3: Paging interval Param4: Call duration
AT+HPSCAN?	+HPSCAN:<Param1>,<Param2>,<Param3>,<Param4> OK	

Example:
AT+HPSCAN =1234,500,1200,250\r\n
OK
AT+HPSCAN?
+HPSCAN:1234,500,1200,250

19. Set/Check – SHIFF parameter

Command	Respond	Parameter
AT+SNIFF=<Param1>,<Param2>,<Param3>,<Param4>	OK	Param1: Max time Param2: Min time Param3: Retry time Param4: Time out
AT+ SNIFF?	+SNIFF:<Param1>,<Param2>,<Param3>,<Param4> OK	

20. Set/Check security mode

Command	Respond	Parameter
AT+SENM=<Param1>,<Param2>	1. OK 2. FAIL	Param1: 0——sec_mode0+off 1——sec_mode1+non_se
AT+ SENM?	+ SENM:<Param1>,<Param2>	

	OK	cure 2——sec_mode2_service 3——sec_mode3_link 4——sec_mode_unknown Param2: 0——hci_enc_mode_off 1——hci_enc_mode_pt_to_pt 2——hci_enc_mode_pt_to_pt_and_bcast

21. Delete Authenticated Device

Command	Respond	Parameter
AT+PMSAD=<Param>	OK	Param: Authenticated Device Address

Example:
AT+PMSAD =1234,56,abcdef\r\n
OK

22. Delete All Authenticated Device

Command	Respond	Parameter
AT+ RMAAD	OK	-

23. Search Authenticated Device

Command	Respond	Parameter
AT+FSAD=<Param>	1. OK 2. FAIL	Param: Device address

24. Get Authenticated Device Count

Command	Respond	Parameter
AT+ADCN?	+ADCN：<Param> OK	Param: Device Count

25. Most Recently Used Authenticated Device

Command	Respond	Parameter
AT+MRAD?	+ MRAD：<Param> OK	Param: Recently Authenticated Device Address

26. Get the module working state

Command	Respond	Parameter

AT+ STATE?	+ STATE：<Param> OK	Param: "INITIALIZED" "READY" "PAIRABLE" "PAIRED" "INQUIRING" "CONNECTING" "CONNECTED" "DISCONNECTED" "NUKNOW"

27. Initialize the SPP profile lib

Command	Respond	Parameter
AT+INIT	1. OK 2. FAIL	-

28. Inquiry Bluetooth Device

Command	Respond	Parameter
AT+INQ	+INQ：<Param1>，<Param2>，<Param3> .... OK	Param1：Address Param2：Device Class Param3：RSSI Signal strength

Example:
AT+INIT\r\n
OK
AT+IAC=9e8b33\r\n
OK
AT+CLASS=0\r\n
AT+INQM=1,9,48\r\n
At+INQ\r\n
+INQ:2:72:D2224,3E0104,FFBC
+INQ:1234:56:0,1F1F,FFC1
+INQ:1234:56:0,1F1F,FFC0
+INQ:1234:56:0,1F1F,FFC1
+INQ:2:72:D2224,3F0104,FFAD
+INQ:1234:56:0,1F1F,FFBE
+INQ:1234:56:0,1F1F,FFC2
+INQ:1234:56:0,1F1F,FFBE
+INQ:2:72:D2224,3F0104,FFBC
OK

28. Cancel Inquiring Bluetooth Device

Command	Respond	Parameter
AT+ INQC	OK	-

29. Equipment Matching

Command	Respond	Parameter
AT+PAIR=<Param1>,<Param2>	1. OK 2. FAIL	Param1：Device Address Param2：Time out

30. Connect Device

Command	Respond	Parameter
AT+LINK=<Param>	1. OK 2. FAIL	Param：Device Address

Example:
AT+FSAD=1234,56,abcdef\r\n
OK
AT+LINK=1234,56,abcdef\r\n
OK

31. Disconnect

Command	Respond	Parameter
AT+DISC	1. +DISC:SUCCESS OK 2. +DISC:LINK_LOSS OK 3. +DISC:NO_SLC OK 4. +DISC:TIMEOUT OK 5. +DISC:ERROR OK	Param：Device Address

32. Energy-saving mode

Command	Respond	Parameter
AT+ENSNIFF=<Param>	OK	Param：Device Address

33. Exerts Energy-saving mode

Command	Respond	Parameter
AT+ EXSNIFF =<Param>	OK	Param：Device Address

ITead Studio

Tech Support: info@iteadstudio.com

Revision History

Rev.	Description	Release date
v1.0	Initial version	7/18/2010

Anexo B. Código del nodo sensor

```
int ec1=2;
int tr1=3;
int ec2=4;
int tr2=5;
int ec3=6;
int tr3=7;
int ec4=8;
int tr4=9;
int ec5=10;
int tr5=11;
int ec6=12;
int tr6=13;
int rd=0, rm=0;
int wr=0;
int led1 = 46;
int led2 = 47;
int led3 = 50;
int led4 = 51;
void setup() {
Serial.begin(115200);
Serial1.begin(115200);
pinMode(led1, OUTPUT);
pinMode(led2, OUTPUT);
pinMode(led3, OUTPUT);
pinMode(led4, OUTPUT);
pinMode(ec1,INPUT);
pinMode(tr1,OUTPUT);
pinMode(ec2,INPUT);
pinMode(tr2,OUTPUT);
pinMode(ec3,INPUT);
pinMode(tr3,OUTPUT);
pinMode(ec4,INPUT);
pinMode(tr4,OUTPUT);
pinMode(ec5,INPUT);
pinMode(tr5,OUTPUT);
pinMode(ec6,INPUT);
pinMode(tr6,OUTPUT);
}
void loop() {
 if(Serial1.available()){
 rd=Serial1.read();
 if(rd>4){
 if(rd==5){
  digitalWrite(led1, LOW);
 }if(rd==8){
  digitalWrite(led1, HIGH);
 }if(rd==9){
  digitalWrite(led2, LOW);
}if(rd==10){
  digitalWrite(led2, HIGH);
}if(rd==14){
  digitalWrite(led3, LOW);
}if(rd==15){
  digitalWrite(led3, HIGH);
}if(rd==19){
  digitalWrite(led4, LOW);
}if(rd==20){
  digitalWrite(led4, HIGH);
}}else{
if(rd==1){
sen3();
//wr=7;
Serial.print("sensor 1:    ");
//delay(1000);
}if(rd==2){
sen4();
//wr=12;
Serial.print("sensor 2:    ");
//delay(1000);
}if(rd==3){
sen5();
//wr=17;
Serial.print("sensor 3:    ");
//delay(1000);
}if(rd==4){
//wr=22;
sen6();
Serial.print("sensor 4:    ");
//delay(1000);
}if(rd==11){
sen5();
Serial.print("sensor 5:    ");
Serial.println(wr);
Serial1.write(wr);
Serial.println();
//delay(1000);
}if(rd==12){
sen6();
Serial.print("sensor 6:    ");
Serial.println(wr);
Serial1.write(wr);
Serial.println();
//delay(1000);
}
Serial.println(wr);
Serial1.write(wr);
```

```
  Serial.println();
  }
}
}
void sen1()
{
  int tm1=0;
  int di1=0;
  digitalWrite(tr1, LOW);
  delayMicroseconds(2);
  digitalWrite(tr1, HIGH);
  delayMicroseconds(10);
  digitalWrite(tr1, LOW);
  tm1 = pulseIn(ec1, HIGH);
  di1 = tm1 / 58;
  if ((di1 <= 70) && (di1>0)) {
  wr=36;
  }else{
  wr=37;
  }
}
void sen2()
{
  int tm1=0;
  int di1=0;
  digitalWrite(tr2, LOW);
  delayMicroseconds(2);
  digitalWrite(tr2, HIGH);
  delayMicroseconds(10);
  digitalWrite(tr2, LOW);
  tm1 = pulseIn(ec2, HIGH);
  di1 = tm1 / 58;
  if ((di1 <= 70) && (di1>0)) {
  wr=41;
}else{
  wr=42;
  }
}
void sen3()
{
  int tm1=0;
  int di1=0;
  digitalWrite(tr3, LOW);
  delayMicroseconds(2);
  digitalWrite(tr3, HIGH);
  delayMicroseconds(10);
  digitalWrite(tr3, LOW);
  tm1 = pulseIn(ec3, HIGH);
  di1 = tm1 / 58;
  if ((di1 <= 70) && (di1>0)) {
  wr=6;
  }else{
  wr=7;
  }
}
void sen4()
{
  int tm1=0;
  int di1=0;
  digitalWrite(tr4, LOW);
  delayMicroseconds(2);
  digitalWrite(tr4, HIGH);
  delayMicroseconds(10);
  digitalWrite(tr4, LOW);
  tm1 = pulseIn(ec4, HIGH);
  di1 = tm1 / 58;
  if ((di1 <= 70) && (di1>0)) {
  wr=11;
  }else{
  wr=12;
  }
}
void sen5()
{
  int tm1=0;
  int di1=0;
  digitalWrite(tr5, LOW);
  delayMicroseconds(2);
  digitalWrite(tr5, HIGH);
  delayMicroseconds(10);
  digitalWrite(tr5, LOW);
  tm1 = pulseIn(ec5, HIGH);
  di1 = tm1 / 58;
  if ((di1 <= 70) && (di1>0)) {
  wr=16;
  }else{
  wr=17;
  }
}
void sen6()
{
  int tm1=0;
  int di1=0;
  digitalWrite(tr6, LOW);
  delayMicroseconds(2);
  digitalWrite(tr6, HIGH);
```

```
  delayMicroseconds(10);
  digitalWrite(tr6, LOW);
  tm1 = pulseIn(ec6, HIGH);
  di1 = tm1 / 58;
  if ((di1 <= 70) && (di1>0)) {
  wr=21;
  }else{
  wr=22;
  }
}
```

Anexo D: Código del nodo recolector

```
int ec1=2;
int tr1=3;
int ec2=4;
int tr2=5;
int ec3=6;
int tr3=7;
int ec4=8;
int tr4=9;
int ec5=10;
int tr5=11;
int ec6=12;
int tr6=13;
int rd=0, rm=0;
int wr=0;
int i=0;
int led1 = 42;
int led2 = 43;
int led3 = 46;
int led4 = 47;
int led5 = 50;
int led6 = 51;
void setup() {
Serial.begin(115200);
Serial1.begin(115200);
Serial2.begin(115200);
Serial3.begin(115200);
pinMode(led1,OUTPUT);
pinMode(led2,OUTPUT);
pinMode(led3,OUTPUT);
pinMode(led4,OUTPUT);
pinMode(led5,OUTPUT);
pinMode(led6,OUTPUT);
pinMode(ec1,INPUT);
pinMode(tr1,OUTPUT);
pinMode(ec2,INPUT);
pinMode(tr2,OUTPUT);
pinMode(ec3,INPUT);
pinMode(tr3,OUTPUT);
pinMode(ec4,INPUT);
pinMode(tr4,OUTPUT);
pinMode(ec5,INPUT);
pinMode(tr5,OUTPUT);
pinMode(ec6,INPUT);
pinMode(tr6,OUTPUT);
}
void loop() {
  if(Serial1.available()){
    rm = Serial1.read();
    Serial.println(rm);
    if(rm<23){
      Serial2.write(rm);
    }
    if(rm==24){
      digitalWrite(led1, LOW);
    }if(rm==25){
      digitalWrite(led1, HIGH);
    }if(rm==29){
      digitalWrite(led2, LOW);
    }if(rm==30){
      digitalWrite(led2, HIGH);
    }if(rm==34){
      digitalWrite(led3, LOW);
    }if(rm==35){
      digitalWrite(led3, HIGH);
    }if(rm==39){
      digitalWrite(led4, LOW);
    }if(rm==40){
      digitalWrite(led4, HIGH);
    }if(rm==44){
      digitalWrite(led5, LOW);
    }if(rm==45){
      digitalWrite(led5, HIGH);
    }if(rm==49){
      digitalWrite(led6, LOW);
    }if(rm==50){
      digitalWrite(led6, HIGH);
    }
  }
  i=i+1;
  Serial.print(i);
  Serial.print("    ");
  if(i<=4){
    Serial2.write(i);
    delay(700);
    if(Serial2.available()){
    rd=Serial2.read();
    wr=rd;
    }
   }if((i>4)&&(i<=10)){
  if(i==5){
    //wr=27;
    sen1();
  }if(i==6){
    sen2();
    //wr=32;
  }if(i==7){
```

```
  sen3();
  //wr=37;
 }if(i==8){
  sen4();
  //wr=42;
 }if(i==9){
  sen5();
  //wr=47;
 }if(i==10){
  sen6();
  //wr=52;
 }
  Serial.print(wr);
 Serial1.write(wr);
 Serial.println();
 delay(4000);
 if(i==10){
  i=0;
 }
}

void sen1()
{
 int tm1=0;
 int di1=0;
 digitalWrite(tr1, LOW);
 delayMicroseconds(2);
 digitalWrite(tr1, HIGH);
 delayMicroseconds(10);
 digitalWrite(tr1, LOW);
 tm1 = pulseIn(ec1, HIGH);
 di1 = tm1 / 58;
 if ((di1 <= 70) && (di1>0)) {
 wr=26;
 }else{
 wr=27;
 }
}
void sen2()
{
 int tm1=0;
 int di1=0;
 digitalWrite(tr2, LOW);
 delayMicroseconds(2);
 digitalWrite(tr2, HIGH);
 delayMicroseconds(10);
 digitalWrite(tr2, LOW);
 tm1 = pulseIn(ec2, HIGH);
 di1 = tm1 / 58;
 if ((di1 <= 70) && (di1>0)) {
 wr=31;
 }else{
 wr=32;
 }
}
void sen3()
{
 int tm1=0;
 int di1=0;
 digitalWrite(tr3, LOW);
 delayMicroseconds(2);
 digitalWrite(tr3, HIGH);
 delayMicroseconds(10);
 digitalWrite(tr3, LOW);
 tm1 = pulseIn(ec3, HIGH);
 di1 = tm1 / 58;
 if ((di1 <= 70) && (di1>0)) {
 wr=36;
 }else{
 wr=
 37;
 }
}
void sen4()
{
 int tm1=0;
 int di1=0;
 digitalWrite(tr4, LOW);
 delayMicroseconds(2);
 digitalWrite(tr4, HIGH);
 delayMicroseconds(10);
 digitalWrite(tr4, LOW);
 tm1 = pulseIn(ec4, HIGH);
 di1 = tm1 / 58;
 if ((di1 <= 70) && (di1>0)) {
 wr=41;
 }else{
 wr=42;
 }
}
void sen5()
{
 int tm1=0;
 int di1=0;
 digitalWrite(tr5, LOW);
 delayMicroseconds(2);
```

```
 digitalWrite(tr5, HIGH);
 delayMicroseconds(10);
 digitalWrite(tr5, LOW);
 tm1 = pulseIn(ec5, HIGH);
 di1 = tm1 / 58;
 if ((di1 <= 70) && (di1>0)) {
 wr=46;
 }else{
 wr=47;
 }
}
void sen6()
{
 int tm1=0;
 int di1=0;
 digitalWrite(tr6, LOW);
 delayMicroseconds(2);
 digitalWrite(tr6, HIGH);
 delayMicroseconds(10);
 digitalWrite(tr6, LOW);
 tm1 = pulseIn(ec6, HIGH);
 di1 = tm1 / 58;
 if ((di1 <= 70) && (di1>0)) {
 wr=51;
 }else{
 wr=52;
 }
}
```

```
#include <SoftwareSerial.h>
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
int contconexion = 0;
int  id=0, st=0, act=0;
int b1=0, b2=0, b3=0, b4=0, b5=0;
int c1=0, c2=0, c3=0, c4=0, c5=0;
int d1=0, d2=0, d3=0, d4=0, d5=0;
int e1=0, e2=0, e3=0;
float rd=2;
float wr;
int res=0;
const char *ssid = "LAPTOP-JAVIER 5";
const char *password = "12345678JS";
unsigned long previousMillis = 0;
char host[48];
String strhost = "www.netwolf-
company.com";
String strurl = "/app/ingDatos.php";
String strurl2 = "/app/actRes.php";
String chipid = "";
SoftwareSerial SerialX(D2, D3);
String enviardatos(String datos) {
  String linea = "error";
  WiFiClient client;
  strhost.toCharArray(host, 49);
  if (!client.connect(host, 80)) {
    Serial.println("Fallo de conexion");
    return linea;
  }
  client.print(String("POST ") + strurl + "
HTTP/1.1" + "\r\n" +
               "Host: " + strhost + "\r\n" +
               "Accept: */*" + "*\r\n" +
               "Content-Length: " + datos.length() +
"\r\n" +
               "Content-Type: application/x-www-
form-urlencoded" + "\r\n" +
               "\r\n" + datos);
  delay(10);
  Serial.print("Enviando datos a SQL...");
  unsigned long timeout = millis();
  while (client.available() == 0) {
    if (millis() - timeout > 5000) {
      Serial.println("Cliente fuera de tiempo!");
      client.stop();
      return linea;
    }
  }
  while(client.available()){
    linea = client.readStringUntil('\r');
    if(linea.indexOf("free") != -1){
      res=id+20;
      if(res==21){
      temp1();
      }if(res==22){
      temp2();
      }if(res==23){
      temp3();
      }if(res==24){
      temp4();
      }if(res==25){
      temp5();
      }if(res==26){
      temp6();
      }if(res==27){
      temp7();
      }if(res==28){
      temp8();
      }if(res==29){
      temp9();
      }if(res==30){
      temp10();
      }if(res==31){
      temp11();
      }if(res==32){
      temp12();
      }if(res==33){
      temp13();
      }if(res==34){
      temp14();
      }if(res==35){
      temp15();
      }if(res==36){
      temp16();
      }if(res==37){
      temp17();
      }if(res==38){
      temp18();
      }
    }
    Serial.println(linea);
  }
  return linea;
}
String senda(String dat) {
```

```
 String lin = "error";
 WiFiClient client;
 strhost.toCharArray(host, 49);
 if (!client.connect(host, 80)) {
  Serial.println("Fallo de conexion");
  return lin;
 }
 client.print(String("POST ") + strurl2 + "
HTTP/1.1" + "\r\n" +
          "Host: " + strhost + "\r\n" +
          "Accept: */*" + "*\r\n" +
          "Content-Length: " + dat.length() +
"\r\n" +
          "Content-Type: application/x-www-
form-urlencoded" + "\r\n" +
          "\r\n" + dat);
 delay(10);
 Serial.print("Enviando datos a SQL...");
 unsigned long timeout = millis();
 while (client.available() == 0) {
  if (millis() - timeout > 5000) {
   Serial.println("Cliente fuera de tiempo!");
   client.stop();
   return lin;
  }
 }
 while(client.available()){
  lin = client.readString();
  Serial.println(lin);
 }
 return lin;
}
void setup() {
 pinMode(D2, INPUT);
 pinMode(D3, OUTPUT);
 Serial.begin(115200);
 Serial.println("");
 SerialX.begin(115200);
 Serial.print("chipId: ");
 chipid = String(ESP.getChipId());
 Serial.println(chipid);
 WiFi.begin(ssid, password);
 while (WiFi.status() != WL_CONNECTED and
contconexion <50) { //Cuenta hasta 50 si no
se puede conectar lo cancela
  ++contconexion;
  delay(500);
  Serial.print(".");
 }
 if (contconexion <50) {
   //para usar con ip fija
   IPAddress ip(192,168,137,100);
   IPAddress gateway(192,168,137,1);
   IPAddress subnet(255,255,255,0);
   WiFi.config(ip, gateway, subnet);
   Serial.println("");
   Serial.println("WiFi conectado");
   Serial.println(WiFi.localIP());
 }
 else {
   Serial.println("");
   Serial.println("Error de conexion");
 }
}
void loop() {
 delay(2000);
if(SerialX.available()){
 rd=SerialX.read();
 Serial.print(rd);
 wr=rd/5;
 id=wr;
 st=(wr-int(wr))*10;
 if(st>2){
  st=1;
 }else{
  st=2;
 }
 Serial.print("      ");
 Serial.print(id);
 Serial.print("      ");
 Serial.println(st);
 enviardatos("id=" + String(id) + "&state=" +
String(st));
}
}
void temp1(){
  b1=b1+1;
  if(b1=3){
  String ide = "st_01";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(5);
  b1=0;
  }else{
  SerialX.write(8);
  }
}
```

```
void temp2(){
  b2=b2+1;
  if(b2==3){
  String ide = "st_02";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(9);
  b2=0;
  }else{
  SerialX.write(10);
  }
}
void temp3(){
  b3=b3+1;
  if(b3==3){
  String ide = "st_03";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(14);
  b3=0;
  }else{
  SerialX.write(15); }
}
void temp4(){
  b4=b4+1;
  if(b4==3){
  String ide = "st_04";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(19);
  b4=0;
  }else{
  SerialX.write(20);}
}
void temp5(){
  b5=b5+1;
  if(b5==3){
  String ide = "st_05";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(24);
  b5=0;
  }else{
  SerialX.write(25);}
}
void temp6(){
  c1=c1+1;
  if(c1==3){
  String ide = "st_06";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(29);
  c1=0;
  }else{
  SerialX.write(30);}
}
void temp7(){
  c2=c2+1;
  if(c2==3){
  String ide = "st_07";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(34);
  c2=0;
  }else{
  SerialX.write(35);}
}
void temp8(){
  c3=c3+1;
  if(c3==3){
  String ide = "st_08";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(39);
  c3=0;
  }else{
  SerialX.write(40);}
}
void temp9(){
  c4=c4+1;
  if(c4==3){
  String ide = "st_09";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(44);
  c4=0;
  }else{
  SerialX.write(45);}
}
void temp10(){
  c5=c5+1;
  if(c5==3){
  String ide = "st_10";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(49);
  c5=0;
  }else{
  SerialX.write(50);}
}
void temp11(){
  b1=b1+1;
  if(b1==3){
  String ide = "st_11";
  senda("ide=" + ide + "&act=" + String(act));
```

```
  SerialX.write(54);
  b1=0;
  }else{
  SerialX.write(55);}
}
void temp12(){
  b2=b2+1;
  if(b2==3){
  String ide = "st_12";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(59);
  b2=0;
  }else{
  SerialX.write(60);}
}
void temp13(){
  b3=b3+1;
  if(b3==3){
  String ide = "st_13";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(64);
  b3=0;
  }else{
  SerialX.write(65);}
}
void temp14(){
  b4=b4+1;
  if(b4==3){
  String ide = "st_14";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(69);
  b4=0;
  }else{
  SerialX.write(70);}
}
void temp15(){
  b5=b5+1;
  if(b5==3){
  String ide = "st_15";
  senda("ide=" + ide + "&act=" + String(act));
SerialX.write(74);
  b5=0;
}else{
  SerialX.write(75);}
}
void temp16(){
  e1=e1+1;
  if(e1==3){
  String ide = "st_16";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(79);
  e1=0;
  }else{
  SerialX.write(80);}
}
void temp17(){
  e2=e2+1;
  if(e2==3){
  String ide = "st_17";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(84);
  e2=0;
  }else{
  SerialX.write(85);}
}
void temp18(){
  e3=e3+1;
  if(e3==3){
  String ide = "st_18";
  senda("ide=" + ide + "&act=" + String(act));
  SerialX.write(89);
  e3=0;
  }else{
  SerialX.write(90);}
}
```

Anexo D. Modelo de encuesta

Encuesta para Trabajo de Titulación

Información de plazas de aparcamiento disponible del parqueadero de la FIE

Objetivo: Evaluar los beneficios y nivel de conocimiento acerca de sistemas de parqueos utilizando el IoT

Lea detenidamente las preguntas y seleccione según su criterio.

1. ¿Usted tiene conocimiento sobre sistemas de parqueaderos inteligentes?
   - Si
   - No
2. ¿Considera que se debe diseñar un sistema de parqueo, que indiqué la disponibilidad de plazas de aparcamiento del parqueadero de la FIE?
   - Si
   - No
3. ¿Piensa que este sistema contribuye a mejorar el uso del tiempo de docentes y estudiantes de la facultad?
   - Si
   - No
4. ¿Utilizaría el sistema de parqueo para reservar una plaza de aparcamiento que en su momento se encuentre libre?
   - Si
   - No
5. 5. El sistema de consulta lo puede hacer a través de una página Web. ¿Haría uso de la página web para reservar un estacionamiento?
   - Si
   - No