UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

TEMA:

Estudio y diseño de un parqueo inteligente utilizando Arduino a través del Internet de las cosas (IoT)

AUTOR

Méndez Ugarte, Erick Fernando

Trabajo de titulación previo a la obtención del título de

INGENIERO EN TELECOMUNICACIONES

TUTOR

M. Sc. Romero Paz, Manuel de Jesús

Guayaquil, Ecuador

15 de Septiembre del 2021

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

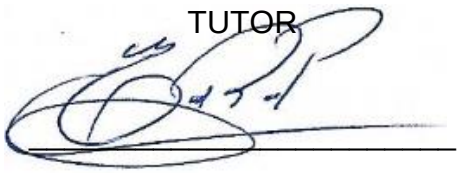
FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

CERTIFICACIÓN

Certificamos que el presente trabajo fue realizado en su totalidad por el Sr. **Méndez Ugarte, Erick Fernando** como requerimiento para la obtención del título de **INGENIERO EN TELECOMUNICACIONES**.

TUTOR



M. Sc. Romero Paz, Manuel de Jesús

DIRECTOR DE CARRERA

M. Sc. Heras Sánchez, Miguel Armando

Guayaquil, a los 15 días del mes de septiembre de 2021

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

Yo, **Méndez Ugarte, Erick Fernando**

DECLARO QUE:

El trabajo de titulación **"Estudio y diseño de un parqueo inteligente utilizando Arduino a través del Internet de las cosas (IoT)"** previo a la obtención del Título de **Ingeniero en Telecomunicaciones**, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

Guayaquil, a los 15 días del mes de septiembre de 2021

EL AUTOR

MENDEZ UGARTE, ERICK FERNANDO

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

AUTORIZACIÓN

Yo, **Méndez Ugarte, Erick Fernando**

Autorizó a la Universidad Católica de Santiago de Guayaquil, la publicación, en la biblioteca de la institución del Trabajo de Titulación: **"Estudio y diseño de un parqueo inteligente utilizando Arduino a través del Internet de las cosas (IoT)"**, cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y total autoría.

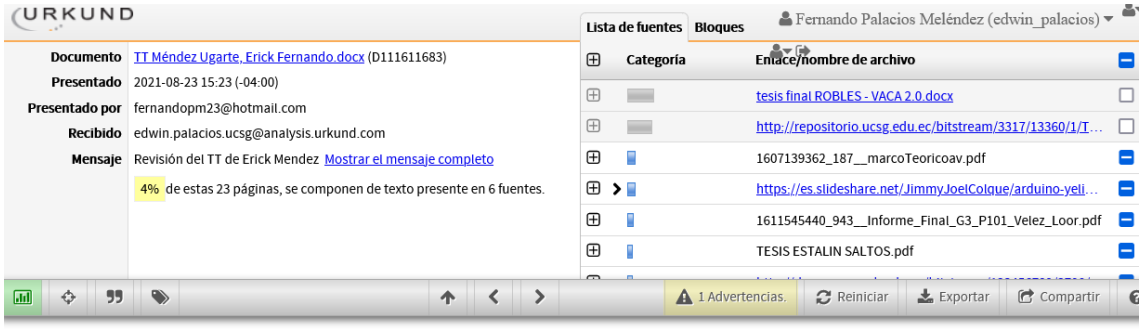
Guayaquil, a los 15 días del mes de septiembre de 2021

EL AUTOR

_____

MENDEZ UGARTE, ERICK FERNANDO

REPORTE URKUND



CARRERA INGENIERIA EN TELECOMUNICACIONES

TEMA Estudio y diseño de un parqueo inteligente utilizando Arduino a través del Internet de las cosas (IoT)

AUTOR Méndez Ugarte, Erick Fernando

Trabajo de titulación

previo a la obtención del título de INGENIERO EN TELECOMUNICACIONES

TUTOR M. Sc. Manuel Romero Paz

Guayaquil, 28 de agosto de 2021 2021

UNIVERSIDAD CATÓLICA DE SANTIAGO DE GUAYAQUIL FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

CERTIFICACIÓN

Certificamos que el presente trabajo fue realizado en su totalidad por

el Sr. Méndez Ugarte, Erick Fernando

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

TRIBUNAL DE SUSTENTACIÓN

f. _____

M. Sc. ROMERO PAZ, MANUEL DE JESÚS

DECANO

f. _____

M. Sc. HERAS SÁNCHEZ, MIGUEL ARMANDO

DIRECTOR DE CARRERA

f. _____

M. Sc. PALACIOS MELÉNDEZ, EDWIN FERNANDO

OPONENTE

ÍNDICE GENERAL

Índice de Figuras

Capítulo 2

Capítulo 3

Índice de Tablas

Capítulo 2

Capítulo 3

RESUMEN

En el presente trabajo de titulación, se diseñó un sistema de parqueo inteligente con Arduino a través del internet de las cosas (IoT) para el futuro control de espacios disponibles u ocupados en el parqueadero de la Universidad Católica Santiago de Guayaquil. En el segundo capítulo se dio a conocer la fundamentación teórica de la tecnología IoT además de los sensores, placas de Arduino, y la comunicación que se puede dar entre ambas placas, por lo cual en el tercer capítulo se estableció un medio físico y un protocolo de comunicación, el cual fue el RS-485 con tipo de comunicación Half-Duplex para ambos Arduinos, bajo el concepto de maestro y esclavos para dar el funcionamiento del sistema. En el diseño de los esclavos se estableció que estos contienen un sensor ultrasónico para detectar la presencia del vehículo conectados a la placa de Arduino Pro mini, mientras que el maestro es una placa de Arduino Yun que recibe la información de los esclavos y la envía a una página web, por lo cual, se creó un servidor web local para almacenar la base de datos y enviar dicha información por medio de una red LAN a la página web. Con este diseño de sistema se pudo prever que su implementación podría reducir la huella de carbono de los vehículos ya que se optimizaría el tiempo de los estudiantes al estacionarse, además se elaboró un presupuesto de los elementos necesarios para el diseño de este sistema.

Palabras clave: IoT, Arduino, RS-485, Microcontroladores, Protocolos, Proteus

ABSTRACT

In the present degree work, an intelligent parking system with Arduino was designed through the internet of things (IoT) for the future control of available or occupied spaces in the parking lot of the Universidad Católica Santiago de Guayaquil. In the second chapter, the theoretical foundation of IoT technology was disclosed in addition to the sensors, Arduino boards, and the communication that can occur between both boards, for which in the third chapter a physical medium and a protocol were established of communication, which was the RS-485 with Half-Duplex communication type for both Arduinos, under the concept of master and slaves to give the operation of the system. In the design of the slaves it was established that they contain an ultrasonic sensor to detect the presence of the vehicle connected to the Arduino Pro mini board, while the master is an Arduino Yun board that receives the information from the slaves and sends it to a web page, therefore, a local web server was created to store the database and send this information through a LAN network to the web page. With this system design, it was possible to foresee that its implementation could reduce the carbon footprint of the vehicles since it would optimize the time of the students when parking, in addition, a budget was prepared for the elements necessary for the design of this system.

Keywords: IoT, Arduino, RS-485, Microcontrollers, Protocols, Proteus

CAPITULO 1: GENERALIDADES DEL TRABAJO DE INVESTIGACIÓN

En este capítulo se presenta de forma detallada la introducción y los antecedentes del presente trabajo de investigación a desarrollar, el problema, los objetivos y la hipótesis, como fundamentos del trabajo que se va a realizar.

1.1 Introducción

La Universidad Católica Santiago de Guayaquil (UCSG) como institución, demanda tráfico cuando comienza su ciclo estudiantil por lo que, ha contado con un espacio de estacionamiento adecuado para recibir y acomodar los vehículos de sus alumnos, sin embargo, dicho espacio suele saturarse. Durante los últimos años de actividades presenciales, la tasa de motorización en el país se ha incrementado, lo que genera problemas de estacionamiento tanto en centros comerciales, como empresas y dentro de ellas, instituciones educativas, sin dejar atrás al parqueadero de la universidad la cual, tiene la necesidad de proveer suficientes lugares para estacionarse.

El crecimiento de los usuarios de internet durante los últimos años ha demandado que la tecnología avance rápidamente, logrando hoy en día obtener el control de las cosas a través de un dispositivo móvil o un dispositivo que le resulte cómodo al usuario, todo gracias a lo que se denomina *el internet de las cosas (IoT)*, cambiando e innovando la forma tradicional en la que se operan ciertas acciones, como lo hacen las Ciudades inteligentes que optimizan recursos y tiempo del habitante. El IoT permite varias aplicaciones, una de ellas son los parqueaderos inteligentes que le facilita al usuario conocer si existe disponibilidad de parqueo a través del uso de internet.

El presente trabajo de investigación propondrá el diseño de un parqueadero inteligente en el campus de la UCSG, que monitoree de forma remota el espacio e informe automáticamente a través de una página web la disponibilidad de lugares para estacionarse, se usará la placa Arduino mini Pro y sensores ultrasónicos como un subsistema detector de espacios disponibles y un Arduino Yun que receptará y enviará información a una base de datos del servidor para mostrar los espacios disponibles en la página web

la cual permitirá a los usuarios revisar los espacios disponibles del parqueadero, aportando un orden y control del parqueadero. Cuando se retomen las clases presenciales, el abastecimiento del parqueadero se estima que será limitado, especialmente en horas pico lo que, generaría un descontrol al momento de estacionarse.

Existiendo un proyecto denominado “Diseño de edificio de parqueos en Campus de la Universidad Católica Santiago De Guayaquil, costos y prefactibilidad del proyecto” (Manrique, 2011), se usará como referencia los datos estadísticos obtenidos de su estudio de la cantidad de incremento anual de estudiantes y sus preferencias de parqueo para proponer una solución al problema de congestionamiento que se atravesaba diariamente y que puede continuar al retomar las clases presenciales.

1.2 Antecedentes

El número de vehículos matriculados durante los últimos 10 años fue incrementando según lo indica el Instituto Nacional de Estadística y Censos, en su reporte del 2020, por lo que se estima que la cifra aumentará cuando se retome la modalidad presencial (INEC, 2020).

La UCSG ha implementado parqueaderos para dar seguridad a los estudiantes, hace 3 años se construyó un parqueadero principal de 4 pisos para liberar congestionamientos, sin embargo, debido a la gran demanda de estudiantes propietarios de vehículos, el control de aparcamiento se suele encontrar saturado en las horas pico y no satisface las necesidades de los estudiantes.

El congestionamiento que se forma en el parqueadero de la universidad se debe a que los usuarios desconocen donde encontrar un espacio disponible, lo que genera pérdida del tiempo en el estudiante, el cual muchas veces pierde su seguridad y prefiere buscar un lugar fuera de la institución educativa.

Otro factor relevante en el congestionamiento es la contaminación, los vehículos emiten CO_2, el cual es nocivo para el medio ambiente, por lo que, al circular por mayor tiempo, este gas se incrementa y si esto sucede en varios vehículos, la contaminación también aumentará considerablemente.

El tiempo promedio que gasta una persona en buscar un espacio de parqueo disponible, supera los 8 minutos donde muchas veces no suele encontrar y debe marcharse, logrando aportar negativamente hasta un 30% de congestionamiento por falta de control en el parqueadero (Shoup, 2006).

La solución a los problemas mencionados anteriormente debe prever al usuario ahorro de tiempo y energía, además de aportar a la sociedad en la reducción de contaminación, sin olvidar otorgarle seguridad dentro de la institución.

1.3 Definición del Problema

Se presenta la falta de control de parqueo y que exista un sistema de sensores y de un mecanismo que ayude y ahorre tiempo a los estudiantes en el momento de querer parquearse dentro de la Universidad. El problema de investigación es el siguiente: “La necesidad de contar con un diseño de un sistema de parqueo inteligente basado en la tecnología IoT para los estudiantes de la Universidad Católica de Santiago de Guayaquil que los beneficiaría en tiempo y seguridad”

1.4 Justificación del Problema

Actualmente los estudiantes de la UCSG tienen dificultades para encontrar parqueo en la Universidad y esto les quita tiempo al momento que tienen que asistir a sus respectivas clases, la institución no cuenta con un sistema que facilite a los estudiantes encontrar parqueo de manera más rápida y segura.

De acuerdo con el estudio realizado por (Manrique, 2011), existía un déficit de 200 espacios dentro del parqueadero del campus, teniendo una

proyección de una tasa de crecimiento anual del 5.3% indicando que, esta cifra se incrementaría y a su vez el déficit de espacios. Esta proyección la pudo confirmar (Valarezo, 2018), quien señaló que en el año 2016 la universidad presentó un total de 15.771 estudiantes matriculados, lo cual se encuentra dentro del rango presentado por (Manrique, 2011), dando como indicio que la tasa seguirá incrementándose y los estudiantes no podrán gozar de seguridad y control al momento de necesitar estacionarse dentro del campus.

La falta de un sistema inteligente de parqueo dentro de la universidad preverá que las autoridades tomen otras posibles soluciones al déficit de espacio, que demanden gastos elevados, como construir o ampliar otro parqueadero para querer satisfacer a la mayoría de los estudiantes, sin notar que el problema es la desorganización que estos espacios pueden ocasionar al no tener un control de monitoreo continuo de los mismos.

Esta investigación es muy necesaria ya que ofrece más seguridad y mejor control en los estacionamientos de la Universidad y fomenta a que haya más orden, ya que esto evitaría que se aglomeren muchos carros en el momento de querer buscar un parqueo y a su vez ayudaría a evitar el congestionamiento de vehículos en los parqueaderos de la Universidad.

1.5 Objetivo General

Diseñar un sistema de parqueadero inteligente utilizando Arduino y el Internet de las cosas (IoT) para la Universidad Católica de Santiago de Guayaquil.

1.6 Objetivos específicos

- Determinar las especificaciones técnicas de los componentes que se van a utilizar para este sistema.
- Establecer los sitios adecuados en donde se deben implementar los sensores.
- Diseñar el sistema de parqueo inteligente a través del software Proteus.

- Programar el microcontrolador a para que recepte la señal de los sensores.
- Seleccionar la plataforma de IoT que permitirá monitorear la disponibilidad de espacio en el parqueadero.
- Elaborar un presupuesto de la implementación de este proyecto de sensores para un estacionamiento inteligente.

1.7 Hipótesis

Con el presente trabajo de titulación para diseñar un parqueo inteligente con Arduino a través de IoT para la Universidad Católica Santiago de Guayaquil, se propondrá brindar seguridad y ahorro de tiempo a los estudiantes al momento de estacionarse.

1.8 Metodología de la investigación

El presente trabajo de titulación a desarrollar es de tipo descriptivo, analítico y explicativo ya que, se va a estudiar y diseñar un sistema que convierta un parqueadero de la universidad en uno inteligente, capaz de proporcionar monitoreo y control de los espacios disponibles para brindar comodidad y seguridad en los estudiantes.

CAPÍTULO 2: FUNDAMENTACIÓN TEÓRICA

En este capítulo se define y fundamenta el concepto teórico de los elementos a nivel de hardware y software que se aplicará para el diseño del sistema inteligente de parqueo para la UCSG.

2.1. Internet de las Cosas (IoT)

La expansión de Internet y la gestión constante de nuevas tecnologías, servicios y plataformas ha permitido el surgimiento de un fenómeno conocido como Internet de las cosas (IoT, Internet of Things), esto incluye la evolución de la conexión a Internet desde una red de objetos interconectados desde una computadora. IoT, junto con la inteligencia artificial, la robótica, la impresión 3D, 4D, la nanotecnología, la biotecnología o la ciencia de los materiales (Barrio, 2018).

El término Internet de las cosas (IoT) fue acuñado por primera vez por Kevin Ahston en 1999. Describe un sistema que puede automatizar la recopilación de datos conectando objetos en el mundo físico a Internet a través de sensores. Internet de las cosas existe desde hace varios años en las sociedades de alta tecnología, incluidos los primeros dispositivos, como teléfonos inteligentes y tabletas, ahora interactúa no solo con computadoras, sino con otros tipos de artículos y objetos, como ropa técnica y usables (relojes, pulseras inteligentes, gafas de realidad aumentada, etc.), electrodomésticos (refrigeradores, aspiradoras de polvo, etc.), televisores, consolas de videojuegos, automóviles, artículos de construcción, entre otros. El acceso a la infraestructura pública más importante, como cámaras, control de acceso, sensores de temperatura, puentes, carreteras y ciudades, lo que permite abrir la puerta a la interacción máquina a máquina (M2M). También puede conectar personas con objetos y conectar animales como ha sucedido con algunas granjas y algunos programas de conservación de la biodiversidad (Barrio, 2018).

Siendo así que el Internet se desarrolló originalmente con el único propósito de compartir información, el cual, ha evolucionado a lo largo de los

años para mejorar el servicio que brinda al usuario, mediante la prestación de servicios de voz, transmisiones de video, operaciones bancarias y confort en casa en general (Coronel & Tenelanda, 2016).

2.1.1. Definición de Internet de las Cosas (IoT)

Se denomina internet de las cosas a la interacción digital que permite el trabajo conjunto entre la interconexión de lo instrumental con la tecnología, lo que hace que los objetos cotidianos sean capaces de introducir y recibir información en la web sin la ayuda del ser humano y supliendo sus necesidades de forma automatizada (Rodríguez, 2019).

Dado que el IoT involucra dispositivos, cuando estos están conectados con sensores, software y otras tecnologías, permiten transferir y recibir datos, desde y hacia otras cosas.

2.1.2. Funcionamiento de IoT

El sistema IoT contiene sensores, los cuales son capaces de capturar datos que el ser humano puede ver, oír o percibir en su entorno, esta información es recibida y se comparte usando las conexiones de redes disponibles a través de los dispositivos IoT, siendo accesible a través de la nube ya sea esta, pública o privada; los datos son procesados y recibidos en el software programado para que realice la actividad requerida, por último, se recopilan y analizan los datos de todos los dispositivos de la red IoT, lo cual brinda información que permitirá automatizar y fundamentar las futuras decisiones (SAP, 2020).

2.1.3. Beneficios del IoT

El IoT genera amplias aplicaciones gracias a su interconexión accesible, lo que genera beneficios no solo personales sino también industriales tales como:

- Mejora de la conectividad, gracias a la capacidad de operar varias cosas desde un dispositivo (Mok, 2017).

- Eficiencia, debido a la conectividad el tiempo de operación es menor (Lang, 2017).
- Comodidad al realizar tareas, por ejemplo, los electrodomésticos inteligentes que ofrecen ahorro de tiempo a través de una sola acción.
- Bienestar, ya que se puede monitorear la salud en tiempo real (Lindsay, 2017).
- Conservación, como por ejemplo en las ciudades inteligentes donde se puede monitorear las condiciones de la ciudad ya sea el tráfico, calidad de aire, uso del agua, factores ambientales y, así distribuir de forma eficiente la energía (Lee, Kwon, Cho, Kim, & Lee, 2016).
- Personalización, al momento de realizar conexiones que se adapten a las necesidades (Lindsay, 2017).

2.1.4. Tecnologías IoT

- **Wi-fi**

Es una tecnología de red inalámbrica que permite que dispositivos como computadoras, móviles y otros dispositivos (impresoras y cámaras) interactúen con WiFi-Internet, permitiendo que estos y muchos otros dispositivos intercambien información entre sí y creen redes (CISCO, 2021). Técnicamente, el estándar IEEE 802.11 define los protocolos que permiten la comunicación con los dispositivos inalámbricos habilitados para WiFi de hoy en día, como enrutadores inalámbricos y puntos de acceso inalámbricos. Los puntos de acceso inalámbricos admiten varios estándares IEEE (CISCO, 2021). La forma más común para que los usuarios se conecten a Internet de forma inalámbrica es utilizar un enrutador inalámbrico (WiFi) en una computadora de escritorio, que ayuda a difundir la señal por toda la casa o el lugar de trabajo. Cuanto más lejos esté un usuario del enrutador WiFi base, más débil será la señal (CISCO, 2021).

- **Bluetooth**

Bluetooth es un estándar desarrollado para la comunicación de datos inalámbrica de corto alcance. Sus principales características son robustez, baja complejidad, bajo consumo y bajo costo. La tecnología Bluetooth es

pequeña y de barata. Esta tecnología opera en la banda de 2.4Ghz y tiene penetración de pares, lo que la hace ideal para trabajos móviles y de oficina (BIBING, 2020). Esta tecnología se clasifica en diferentes clases que definen la potencia de transmisión del dispositivo, tal como lo describe la siguiente tabla:

Tabla 0.1 Clases Bluetooth

Clase	Potencia máxima permitida (mW)	Potencia máxima permitida (dBm)	Rango aproximado
Clase 1	100 mW	20 dBm	~100 metros
Clase 2	2,5 mW	4 dBm	~20 metros
Clase 3	1 mW	0 dBm	~1 metro

Fuente: (BIBING, 2020)

La tecnología Bluetooth permite aumentar la eficiencia de las comunicaciones de corta distancia, su topología puede ser de punto a punto o de punto a multipunto, tal como lo describe la siguiente imagen.

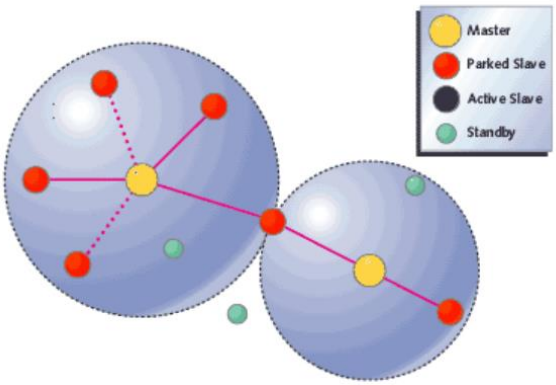


Figura 0-1 Topología de red Bluettoth
Fuente: (BIBING, 2020)

Gracias a esta tecnología, los dispositivos se comunican en redes piconects, las cuales permiten crecer hasta 8 conexiones de punto a punto, donde un dispositivo actúa como máster y envía información del reloj de saltos de frecuencia, donde, los demás dispositivos actúan como esclavos.

- **Zigbee**

Zigbee es el nombre comercial dado a la tecnología de redes personales inalámbricas de corto alcance según el estándar utilizado por Zigbee Alliance

y el desarrollo de redes de sensores inalámbricos, IEEE802.15. Zigbee se basa en la capa física y la capa de enlace de datos definidas en el estándar IEEE 802.15 y define las capas superiores como la capa de red y la capa de aplicación y opera a banda libre de 2.4Ghz, 885Mhz para Europa y 915Mhz para Estados Unidos (Coronel & Tenelanda, 2016).

Esta tecnología se estableció como solución inalámbrica alternativa de baja capacidad para aplicaciones específicas en el hogar, dando paso a la seguridad y automatización aplicable a la domótica, automatización industrial, reconocimiento remoto, medicina, entre otros (Moreno & Ruiz, 2007).

La arquitectura de esta tecnología contiene varias capas como IEE 802.15.4.2003 en la capa de control de acceso al medio (MAC), la capa física (PHY) y la capa de red Zigbee (NWK) (Moreno & Ruiz, 2007), tal como lo describe la siguiente imagen.

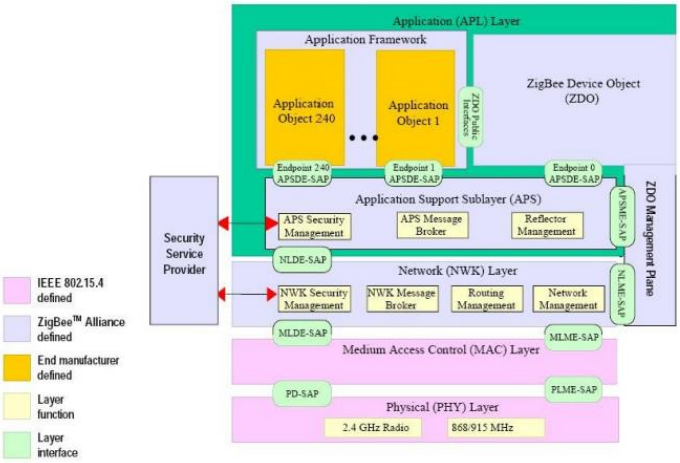


Figura 0-2 Pila de protocolo Zigbee
Fuente: (Moreno & Ruiz, 2007)

En este proyecto de titulación se utilizará la tecnología Wi-Fi para enviar la información recibida por los sensores a través de un servidor en una página web.

2.1.5. Plataforma IoT

Las plataformas IoT son aquellas que permiten establecer una conexión de red local a la nube, a través de aplicaciones que ofrecen acceso a los usuarios desde cualquier parte del mundo. Estas plataformas se integran a

tecnologías como Data Analytic, Machine Learning, Inteligencia Artificial, entre otros (Quiñonez, 2019).

En la arquitectura de IoT, las plataformas involucran un papel importante ya que estas permiten la conectividad de dispositivos IoT tales como sensores y redes lo que la hace capaz de administrar datos en la nube, gestionar la seguridad y modularse a la necesidad de cada cliente de forma independiente (Quiñonez, 2019).

Actualmente existen en el mercado muchas plataformas de IoT, compañías que ofrecen su servicio de forma libre u otras que son de paga. En este trabajo de titulación se utilizará una plataforma de libre acceso para el usuario.

2.2. Hardware Libre

Es aquel hardware público, que permite el acceso completo al diseño sin licencia de propiedad, permitiendo analizar, modificar, para mejorar o distribuir el hardware en cuestión (Novillo, Hernández, Mazón, Molina, & Cárdenas, 2018).

2.3. Software Libre

Es aquel Software público, que permite el acceso completo al código de programación, permitiendo a otro usuario o a una comunidad, analizar, modificar e implementar dicho código para la implementación de futuros proyectos.

2.4. Microcontroladores

Un microcontrolador es un sistema completo de microcomputadoras. Esto significa que contiene un microprocesador, memoria de datos (SRAM), memoria de programa (FLASH) y la unidad de entradas y salidas (E/S) los cuales, están contenidos en un circuito integrado, lo que lo hace extremadamente pequeño, a su vez económico, fácil de manejar e ideal para

muchas aplicaciones que requieran un trabajo en específico, permitiendo ser parte de un sistema embebido (Rossano, 2009).

Figura 0-3 Estructura de un microcontrolador
Fuente: (Rossano, 2009)

Actualmente, existe una gran variedad de microcontroladores diseñados y fabricados por empresas como: Microchip, Freescale, Intel, Atmel; todos variados dependiendo de la necesidad que requiera el usuario, lo que los hace útiles a innumerables aplicaciones (Rossano, 2009).

Para este trabajo de titulación se trabajará con microcontroladores Atmega de la familia Atmel.

2.5. Comunicación entre microcontroladores

Se sabe que para establecer una comunicación se necesitan dos partes involucradas, es decir, un emisor y un receptor, siendo el emisor aquel que envía un mensaje en un código adecuado para ser recibido y comprendido por el receptor, las telecomunicaciones se basan en este concepto ya que permiten la comunicación a distancia, a través de dispositivos eléctricos y/o tecnológicos, que incluyen microcontroladores, los cuales donde pueden actuar como emisores y receptores.

Además, se necesita de un medio para establecer la transmisión de datos entre los dispositivos involucrados, el cual puede ser de tipo guiado y no guiado. Los medios guiados conducen ondas electromagnéticas a través de un camino físico por lo que entre ellos se encuentran los pares trenzados, cables coaxiales, fibras ópticas, etc., y por otra parte los no guiados brindan

un soporte para que las ondas se transmitan mas no las dirigen con el fin de que no se confinen, entre ellos se encuentran el aire y vacío, es decir que la transmisión se puede dar vía radio, vía satélite, entre otros (Perez, 2016).

2.5.1. Medio de transmisión

De acuerdo con lo dicho anteriormente, los datos que se transmiten en la pueden circular en tres medios:

- **Simplex:** También llamado unidireccional, la transmisión se da en una sola dirección por lo cual se considera única, un ejemplo de esta es la señal que se envía de una estación de radio hacia los radios en casa (CISCO, 2021).

- **Half- Duplex:** Los datos se transmiten en dos direcciones, pero no de forma simultánea, por lo que los datos circulan en una sola dirección por vez de transmisión, un ejemplo de esto es el radio comunicador (CISCO, 2021).

- **Full- Duplex:** Se establece cuando los datos circulan en ambas direcciones a la vez, es decir que es bidireccional, sin embargo, el ancho de banda se mide en una sola dirección de transmisión, un ejemplo de esto es el teléfono (CISCO, 2021).

2.5.2. Protocolos

Para establecer comunicación entre mínimo dos microcontroladores se necesita utilizar un protocolo, el cual se puede definir como un conjunto de reglas que indique como establecer dicha comunicación. Actualmente existen dos tipos de protocolos, en serie o serial y en paralelo; la comunicación en paralelo transmite todos los bits de un carácter de máquina a través de varias líneas, se utiliza en distancias cortas, en la comunicación en serie los datos son transmitidos de bit en bit a través de una sola línea, lo que lo hace ideal para distancias cortas, medianas y largas (Cobo, 2009). Siendo esta ultima la

más utilizada para comunicar microcontroladores debido a sus amplias aplicaciones.

De la comunicación en serie o serial se encuentran:

- **RS-485:** Es un sistema de bus de transmisión multipunto diferencial (equilibrada), consta de transceptores conectados por un cable de par trenzado, es decir que la señal es transportada por dos cables, uno transmite la señal y otro transporta su copia inversa, lo que lo hace ideal para transmitir datos a largas distancias con altas velocidades, 100kbits/s a 1200 metros. Entre sus ventajas se encuentra que se pueden conectar hasta 256 dispositivos, además, que reduce los ruidos que aparecen en los voltajes producidos en la línea de transmisión, lo que lo hace útil en el área industrial o en ambientes sensibles (Perez, 2016).

- **I2C:** Se deriva del acrónimo en inglés *Inter-IC*, es un bus de comunicación half- duplex, con una línea para la señal de reloj y la otra para datos llamadas SCL Y SDA respectivamente. Permite velocidades desde 100kb/s hasta 3.4 Mb/s. Es fácil de usar y pueden añadirse varios dispositivos al bus, hasta 128 sin embargo, su distancia de transmisión es corta (Valderrama, 1999).

- **SPI:** El bus SPI (Serial Peripheral Interface) es un protocolo síncrono de modo full duplex lo que permite la comunicación entre dispositivos para recibir y transmitir información al mismo tiempo utilizando canales diferentes a través del mismo cable, es decir que funciona bajo el paradigma maestro-esclavo, transmite datos a velocidades altas hasta 10Mbps, pero a distancias cortas de 10cm a 20 cm (Perez, 2016).

Ya que se requiere una red de sensores amplia debido a la capacidad del parqueadero de la universidad, se utilizará el protocolo de RS-485.

2.6. Topología de Red del RS-485

Una topología de red es una estructura de intercomunicación que permite organizar las redes de transmisión de datos entre dispositivos (García & Castillo, 2007). La manera en que se distribuyen los cables del protocolo RS-485, permite definir diferentes topologías de red, las cuales solo deben ser consideradas cuando la distancia entre maestros y esclavos sea grande ya que se involucran cables largos. Estas topologías se mencionan a continuación.

2.6.1. Bus con derivaciones

Está formada por un bus principal y varias ramificaciones, es decir que comparten el mismo canal de comunicación, las ramificaciones deben ser menores que el bus principal ya que varios dispositivos conectados a largas distancias pueden traer problemas de comunicación. El mensaje se envía por el bus y todos los nodos receptan esta información (García & Castillo, 2007).

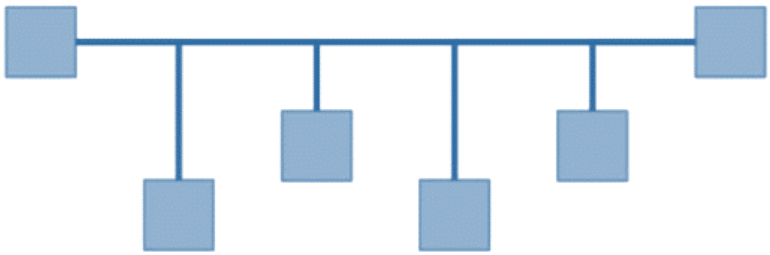


Figura 0-4 Topología en Bus
Fuente: (Guerra, 2021)

2.6.2. Estrella o bus con estrellas

Este tipo de topología tiene la estructura en forma de estrella y suele ser usada para redes pequeñas que usen este protocolo, lo que quiere decir que, involucren pocos dispositivos y distancias cortas.

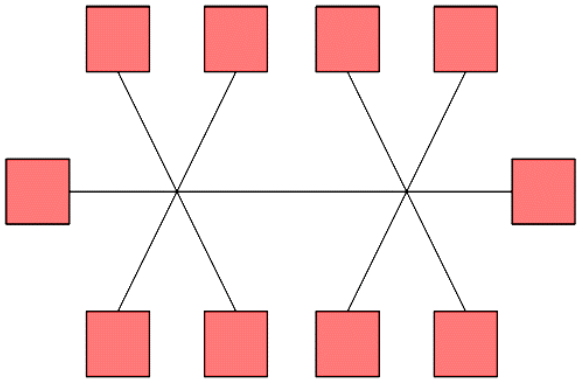


Figura 0-5 Topología en estrella
Fuente: (Guerra, 2021)

2.6.3. Daisy Chain

También llamada conexión en cadena se utiliza en la mayoría de las instalaciones de redes RS-485, ya que los dispositivos están directamente conectados al bus principal sin tener ramales, lo cual hace que las reflexiones dadas por los conductores se minimicen, siendo la más recomendable para este protocolo.

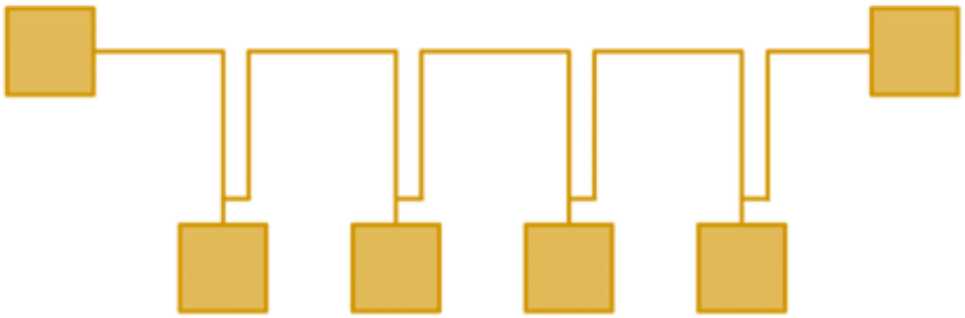


Figura 0-6 Topología de red en cadena
Fuente: (Guerra, 2021)

2.7. Arduino

Arduino es una compañía de desarrollo de hardware, que diseña y crea circuitos electrónicos en circuitos impresos que incluyen un microcontrolador, permitiendo a los usuarios construir dispositivos electrónicos y/o sistemas que sean fáciles de programar escogiendo la placa que se ajuste a sus necesidades específicas (Novillo, Hernández, Mazón, Molina, & Cárdenas, 2018).

Existen varios modelos de las placas de Arduino, entre ellas se encuentran Arduino Uno, Arduino nano, Arduino Pro Mini, Arduino Yun,

Arduino ADK, Arduino Duemilanove, Arduino Leonardo, entre otros. En este trabajo de titulación se resaltarán los microcontroladores: Arduino Yun y Arduino Pro mini.

2.7.1. Arduino Uno

Es la placa de microcontrolador más utilizada de la familia Arduino puesto que, es usada en fines académicos y en la elaboración de proyectos personales, siendo factible para iniciar en el desarrollo de hardware gracias a que se encuentra ampliamente documentada (Novillo, Hernández, Mazón, Molina, & Cárdenas, 2018).

Su nombre “Uno” se asignó ya que fue la primera placa de la serie de Arduino que contiene USB (Arduino CC, 2021).

2.7.2. Arduino Yun

Esta placa de microcontrolador de desarrollo está basada en el sistema Linux, que permite la conexión avanzada a redes y aplicaciones a través de su placa de red Wi-Fi o por su puerto ethernet. Su comunicación con LunixDistribution on board permite utilizarlo como una computadora en red con fácil implementación de Arduino (Novillo, Hernández, Mazón, Molina, & Cárdenas, 2018).

2.7.3. Arduino Pro Mini

Es una placa de la familia Arduino que se basa en el microcontrolador Atmega328, es más pequeña comparada con otras placas como Arduino Nano o Arduino Micro, es más económico que la placa Arduino Uno y está diseñado para usarlo en prototipos y proyectos finales que no requieran cambios grandes y que se fijen en lugares semipermanentes con conexiones establecidas. Esta placa viene sin cabezales premontados, es decir que permite el uso de varios tipos de conectores o soldaduras directas de cables (Arduino CC, 2021).

2.8. Sensores

Los sensores son fundamentales para el desarrollo del Internet de las Cosas, ya que, son dispositivos electrónicos que miden alguna variable del entorno y la traducen en una señal medible, contiene un transductor que transforma la energía que se desea medir en otro tipo de energía. (Escolano, Cazorla, Alfonso, Colomina, & Lozano, 2003).

La configuración de un sensor se la puede observar en el siguiente esquema:

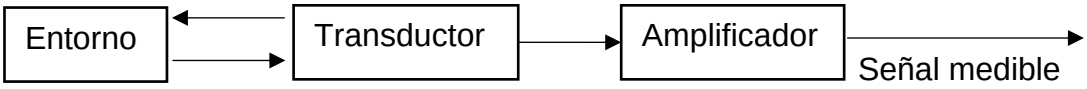


Figura 0-7 Esquema de la configuración de un sensor

Fuente: (Escolano, Cazorla, Alfonso, Colomina, & Lozano, 2003)

Existen dos tipos de sensores, activos y pasivos, los primeros son aquellos que interactúan con el entorno y emiten energía para poder realizar la medición, al contrario, los pasivos no emiten la energía que reciben (Domínguez & Ferrer, 2012).

La aplicación de sensores es amplia debido a que existen sensores para cada tipo de variable necesaria a medir, tales como, de temperatura, luz, distancia, proximidad, posición, color, humedad, velocidad, los cuales pueden ser medidos con sensores capacitivos, inductivos, fotoeléctricos, conductivos, o de ultrasonido.

De acuerdo con este trabajo de titulación, los tipos de sensores se reducen a ser seleccionados entre capacitivos, inductivos o ultrasónicos, por lo que a continuación, se plantea la siguiente tabla que describe algunas características en las que se diferencian estos tipos de sensores.

Tabla 0.2 Características entre sensores

Sensor	Rango máximo de medición [m]	Tipo de material que puede medir
Capacitivo	0,001-0,015	Cualquiera
Inductivo	0,001-0,5	Metal
Ultrasónico	0,01-6	Cualquiera

Fuente: Autor

2.8.1. Sensor Ultrasónico

Este sensor se utiliza para medir la distancia a la que se encuentran posibles obstáculos y para vigilar un espacio, con respecto a vehículos, suelen colocarse dentro de los parachoques puesto que facilita la entrada y salida de aparcaderos y maniobras de estacionamiento (BOSCH, 2002).

2.9. Resistor

El resistor, también llamado resistencia es un componente electrónico compuesto por carbón que se encarga de introducir una resistencia eléctrica en un circuito eléctrico, con el fin de disminuir la corriente que pasa a través de él, evitando que se quemen otros dispositivos (como los diodos led), involucrados en el circuito. Su unidad de medida es el Ohmio, atribuido así por el físico y matemático Georg Simon Ohm (EETech Media, LLC. , 2021).

Existen dos tipos de resistencias, las fijas y las variables, Las resistencias fijas son aquellas que tiene un valor óhmico que no se puede modificar, es compacta y duradera (Fowler, 1994). Las resistencias variables son aquellas que permiten la modificación del valor óhmico a voluntad del usuario, tal como el potenciómetro (Bragós, 1999).

El tipo de resistencia que se usará en este trabajo de titulación son las resistencias fijas de carbono con determinado valor óhmico y potencia.

2.10. Diodo Led

El emisor de Luz o diodo LED (Light Emiting Diode), es un elemento electrónico llamado así, porque emite fotones cuando recibe corriente eléctrica de baja intensidad, se lo encierra en un material de plástico de color, lo que permite acentuar la longitud de onda que se genera por el diodo (Mecatrónica Latam, 2021).

Contiene dos terminales, el ánodo y el cátodo, siendo el ánodo el más largo; cuando el diodo LED se polariza comienza la producción de fotones, gracias a los electrones de la banda que se conducen y combinan con agujeros de la banda de valencia. Su intensidad de luz está relacionada con la cantidad de corriente que fluye a través de él.

2.11. Proteus

Proteus es una aplicación dedicada a la ejecución de proyectos electrónicos que involucra el diseño esquemático, programación del software, construcción de la placa PCB (Printed Circuit Board), simulación de todo el conjunto, depuración de errores, documentación y construcción (HUBOR, 2015).

CAPÍTULO 3: DESARROLLO DE LA INVESTIGACIÓN

En este capítulo se desarrollarán los objetivos específicos de este trabajo de titulación, por lo que se establece la arquitectura del sistema de parqueo inteligente, se describen las características técnicas de los componentes que involucran el hardware, se plantea el diseño esquemático de la conexión en el software Proteus, se presenta el código de programación en las placas Arduino que contienen los microcontroladores, el lugar donde podría aplicarse este proyecto y su precio total estimado de implementación.

3.1. Diseño del sistema de parque inteligente

En esta sección se detalla la propuesta de diseño de la arquitectura del sistema de parqueo inteligente para la recepción de señales por cada sensor ultrasónico con el Arduino Pro Mini, involucrados como nodos sensores o esclavos, y la composición de Arduino Yun como maestro, a través de la correcta conexión eléctrica del circuito y de su programación. En el software Proteus se diseñan los subsistemas nodo sensor y nodo recolector.

3.1.1. Arquitectura del sistema

La arquitectura del sistema de parqueo inteligente se representa en la Figura 0-1, el cual pretende almacenar la información sobre el estacionamiento disponible y desocupado, a través de la página web como plataforma IoT, la cual es de fácil acceso a través del teléfono inteligente o una computadora, permitiendo el usuario podrá obtener información de los espacios disponibles.

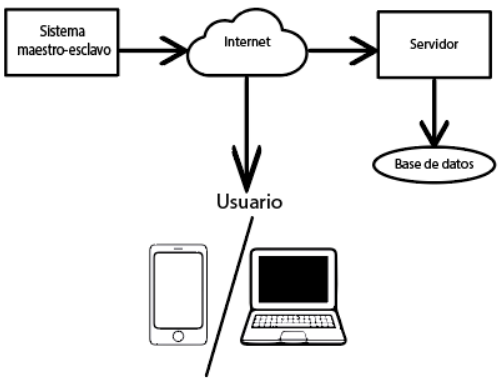


Figura 0-1 Arquitectura del sistema de parqueo inteligente propuesto.
Fuente: Autor

Este sistema de estacionamiento inteligente que, consta de varios componentes y su funcionalidad, incluye:

- **Sistema maestro-esclavo:** Está compuesto por redes de sensores establecidas como nodos sensores (esclavos), y un nodo recolector que actúa maestro y envía la adquisición de datos a través de internet a un servidor web.
- **Servidor:** Mantiene información sobre las plazas de aparcamiento presentes en el parqueadero. Un servidor HTTP, comúnmente llamado servidor web, se define como una tecnología de la información que procesa solicitudes a través del protocolo de red HTTP utilizado para difundir y compartir información en la World Wide Web.
- **Acceso a la web:** el usuario puede conectarse con el sistema de estacionamiento inteligente a través de sus teléfonos inteligentes o computadoras a la página web.

A continuación, se establece la estructura del sistema maestro-esclavo:

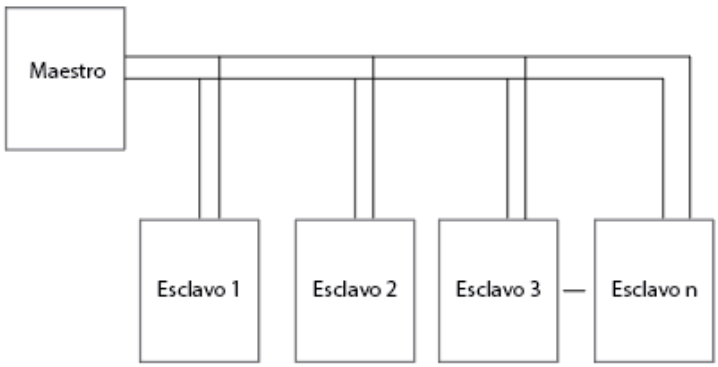


Figura 0-2 Estructura del sistema maestro-esclavo
Fuente: Autor

Cada esclavo se denominará nodo sensor ya que contendrá un sensor de distancia y estará conectado al maestro o nodo recolector el cual recibe información enviada por los esclavos.

- **Estructura del nodo sensor**

Este se encarga de verificar el estado del parqueadero, a través de un sensor de distancia, con una placa de Arduino se establecerá la adquisición

de datos, y se procesará la información, la cual se transmite usando comunicación en serie al nodo recolector. El siguiente diagrama describe el principio de funcionamiento de este nodo:

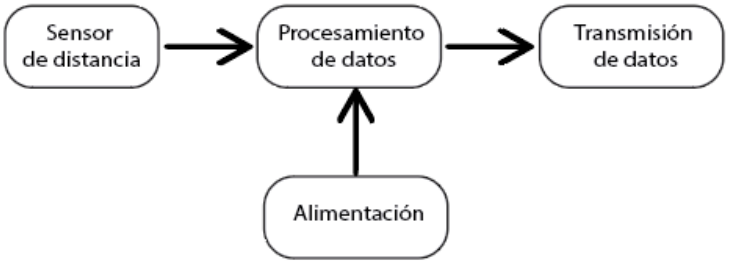


Figura 0-3 Diagrama de bloques de la estructura del nodo sensor
Fuente: Autor

- **Estructura del nodo recolector**

Este nodo recibirá los datos del nodo sensor a través de la comunicación serial de una placa de Arduino como CPU, la misma que actuará como maestro y enviará la información por Wifi a un servidor y página web.

El siguiente diagrama describe el principio de funcionamiento de este nodo:

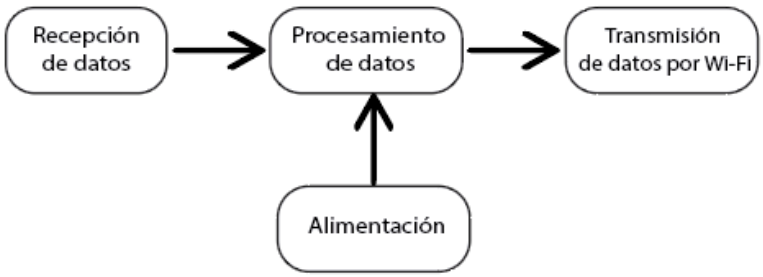


Figura 0-4 Diagrama de bloques de la estructura del nodo recolector
Fuente: Autor

3.2. Selección y especificaciones técnicas de los componentes que se van a utilizar para el sistema.

Los dispositivos y/o elementos seleccionados para el diseño del sistema de parqueo inteligente se enumeran a continuación con sus respectivas características técnicas.

3.2.1. Nodo sensor

- **Arduino Pro Mini**

Se seleccionó el Arduino Pro Mini como placa que administrará las señales detectadas por el sensor ya que, es económico, de uso compacto y estará en un lugar fijo. En la tabla 3.1 se muestran las especificaciones técnicas más relevantes del dispositivo Arduino Pro mini.

Tabla 0.1 Especificaciones técnicas del Arduino Pro mini.

Arduino Pro mini		
	Microcontrolador	ATmega328P
	Fuente de alimentación de la placa	5-12 V
	Voltaje de funcionamiento del circuito	5 V
	Pines de E / S digitales	14
	Pines PWM	6
	UART	1
	SPI	1
	I2C	1
	Pines de entrada analógica	6
	Interrupciones externas	2
	Corriente CC por pin de E / S	40 mA

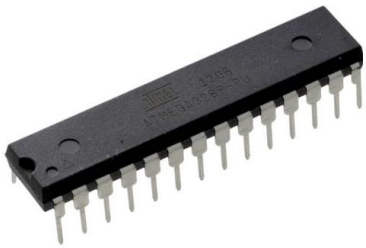
Fuente: (Arduino CC, 2021)

Microcontrolador Atmega 328P

A continuación, en la tabla 3.2 se presentan las características técnicas del microcontrolador Atmega328P contenido en el Arduino Pro mini.

Tabla 0.2 Especificaciones técnicas del microcontrolador Atmega328P

Microcontrolador Atmega 328P	
Fabricante	Atmel
Voltaje de Operación	1.8 a 5.5 VDC
Arquitectura de CPU	8 bit AVR
Memoria Flash	32kB
Memoria RAM	2 kB
Memoria EEPROM	2 kB

	Frecuencia de operación	20 Mhz
	Pines de E/S	23
	Canales ADC	10
	Interfaces	UART, TWI, SPI
	Temperatura de operación	-40° a 85°C

Fuente: (Geek Factory, 2021)

- **Sensor HC-SR04**

De acuerdo con la tabla 3.3, la mejor opción de sensores para este tipo de aplicación son los sensores ultrasónicos ya que, tienen mayor rango de alcance y detectan cualquier material, por esta razón se escoge el sensor HC-SR04, el cual tiene un rango de medida desde 2cm hasta 450cm y no se ve afectado por la luz solar ni por el color, lo cual permite detectar vehículos sin problema.

Tabla 0.3 Especificaciones técnicas del Sensor HC-SR04

Sensor HC-SR04		
	Voltaje de Operación	5V DC
	Corriente de reposo	2mA
	Corriente de trabajo	15mA
	Rango de medición	2cm a 450cm
	Precisión	+- 3mm
	Ángulo de apertura	15°
	Frecuencia de ultrasonido	40KHz
Duración mínima del pulso de disparo TRIG (nivel TTL)		10 µS
Duración del pulso ECO de salida (nivel TTL)		100-25000 µS
Dimensiones		45mm x 20mm x 15mm

Fuente: (Naylamp Mechatronics SAC, 2021)

- **Diodo LED**

Dado que el sistema de parque necesita un indicador visual que muestre al usuario el aparcamiento disponible u ocupado, se usan diodos led de color rojo que indicará que el espacio está ocupado y verde que el espacio está disponible.

Tabla 0.4 Especificaciones técnicas del diodo LED color rojo

Diodo LED		
	Color	Rojo
	Longitud de onda	660 nm
	Tensión	1.8V a 2.2V

Fuente: (Mecatrónica Latam, 2021)

Tabla 0.5 Especificaciones técnicas del diodo LED color verde

Diodo LED		
	Color	Verde
	Longitud de onda	565 nm
	Tensión	2.0 V a 3.5V

Fuente: (Mecatrónica Latam, 2021)

- **Resistor**

Se selecciona un resistor de cerámica debido a que los leds necesitan una resistencia eléctrica para no dañarse por la cantidad de corriente recibida.

Tabla 0.6 Especificaciones técnicas del resistor a utilizar

Resistor		
	Tipo	Película de carbono
	Resistencia	220 ohm
	Potencia N.	0.5 W
	Terminación	axial

Fuente: (Ecobadajoz, 2021)

3.2.2. CPU

- **Arduino Yun**

Se seleccionó el Arduino Yun como placa máster debido a que puede usarse como una CPU ya que contiene su microprocesador con un almacenamiento por microSD, wifi y puerto ethernet para establecer conexión por internet.

Tabla 0.7 Especificaciones técnicas del Arduino Yun.

Arduino YUN		
	Microcontrolador	ATmega32U4
	Voltaje de Operación	5V
	Pines digitales	20
	Pines PWM	7
	Pines de entradas analógicas	12
	Corriente DC por cada pin I/O	40 mA
	Corriente DC en el pin de 3.5 V	50mA
	Memoria Flash	32 kB
	Memoria SRAM	2.5 kB
	Memoria EEPROM	1 kB
	Velocidad de reloj	16 MHz

Fuente: (Arduino CC, 2021)

A continuación, se presenta la información técnica del microprocesador incluido en la placa de Arduino Yun.

Microprocesador Atheros AR9331

Tabla 0.8 Especificaciones técnicas del Microprocesador Atheros AR9331

Procesador	**Atheros AR9331**
Arquitectura	MIPS
Tensión de funcionamiento	3,3 V
Ethernet	802,3 10/100 Mbit / s
Wifi	802.11b / g / n 2,4 GHz
Tipo de USB	2.0 anfitrión
Puerto de memoria	Micro-SD
RAM	DDR2 de 64 MB
Memoria flash	16 MB
SRAM	2,5 KB
EEPROM	1 KB
Velocidad de reloj	400 MHz

Fuente: (Arduino CC, 2021)

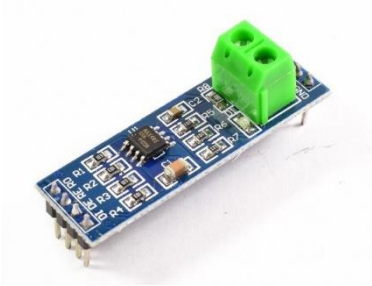
3.2.3. Comunicación

El medio por el que se establecerá comunicación entre el Arduino del nodo sensor y el Arduino del nodo recolector será por cable utilizando el protocolo RS-485, con topología de Red Bus Daysi Chain y comunicación tipo Half-Duplex, esto se seleccionó debido a que se requiere conectar una red grande de sensores en un ambiente ruidoso.

Para conectar esta comunicación entre las placas de Arduino Yun y Arduino Pro mini, se utilizará un módulo del chip RS-485 ya que es de fácil conexión e instalación, a continuación, se presenta sus respectivas especificaciones técnicas.

- **Módulo RS-485**

Tabla 0.9 Especificaciones técnicas del módulo RS-485

Módulo conversor RS-485		
	Voltaje de alimentación	5V DC
	Corriente de operación	500uA máx
	Chip principal	MAX-485
	Tipo de comunicación	Half Dúplex
	Corriente DC por cada pin I/O	40 mA
	Velocidad máxima	10 Mbit/s a 12 metros
	Distancia máxima de alcance	1200 metros a 100kbit/s
	Dimensiones	44*14*20 mm
	Peso	4 gr.

Fuente: (Naylamp Mechatronics SAC, 2021)

También se añadirá resistencia de 120 ohm en el módulo RS485 para reducir problemas de interferencia, perdidas debido a la larga distancia y problemas de desacople.

3.2.4. Base de datos

Para lograr cumplir con el objetivo de acceder a la disponibilidad del proyecto, se diseña un servidor web de forma local para tener la información que será almacenada en la nube, para que posteriormente sea visualizada por los usuarios en una página web, por lo que se optó por crear un servidor web local en php utilizando nginx, en caso de implementación se pueden utilizar servidores webs remotos profesionales de paga.

La base de datos se creará utilizando el servicio de MySQL ya que es gratuito y permite cumplir con la base que se requiere en este proyecto. Se crea una tabla como tal para almacenar datos como id del espacio, número del espacio, estado y hora con el programa MySQL Workbench.

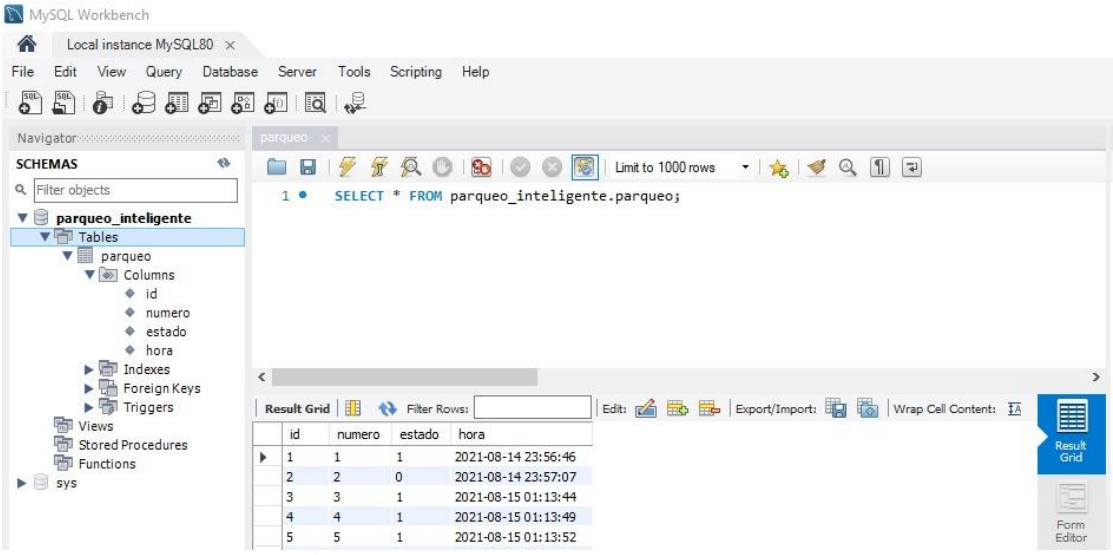


Figura 0-5 Creación de la base de datos en el programa MySQL Workbench
Fuente: Autor

El servidor web está en php usando nginx y se lo configura:

nginx.conf 1

conf > nginx.conf

```
32      #gzip  on;
33
34      server {
35          listen       80;
36          server_name  localhost;
37
38          #charset koi8-r;
39
40          #access_log  logs/host.access.log  main;
41          root www;
42
43          location / {
44              index  index.html index.htm;
45          }
46
47          #error_page  404              /404.html;
48
49          # redirect server error pages to the static page /50x.html
50          #
51          error_page   500 502 503 504  /50x.html;
52          location = /50x.html {
53              root   html;
54          }
55
56          # proxy the PHP scripts to Apache listening on 127.0.0.1:80
57          #
58          #location ~ \.php$ {
59          #    proxy_pass   http://127.0.0.1;
60          #}
61
62          # pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000
63          #
64          location ~ \.php$ {
65             fastcgi_pass   127.0.0.1:9999;
66              fastcgi_index  index.php;
67              fastcgi_param  SCRIPT_FILENAME  $document_root$fastcgi_script_name;
68              include        fastcgi_params;
69          }
70
```

Figura 0-6 Programación del servidor web
Fuente: Autor

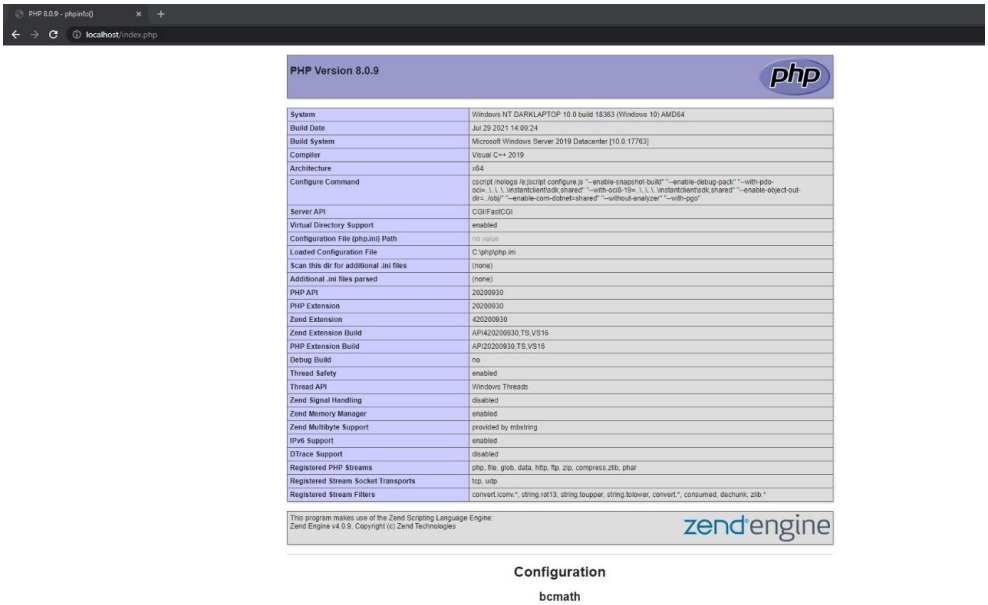
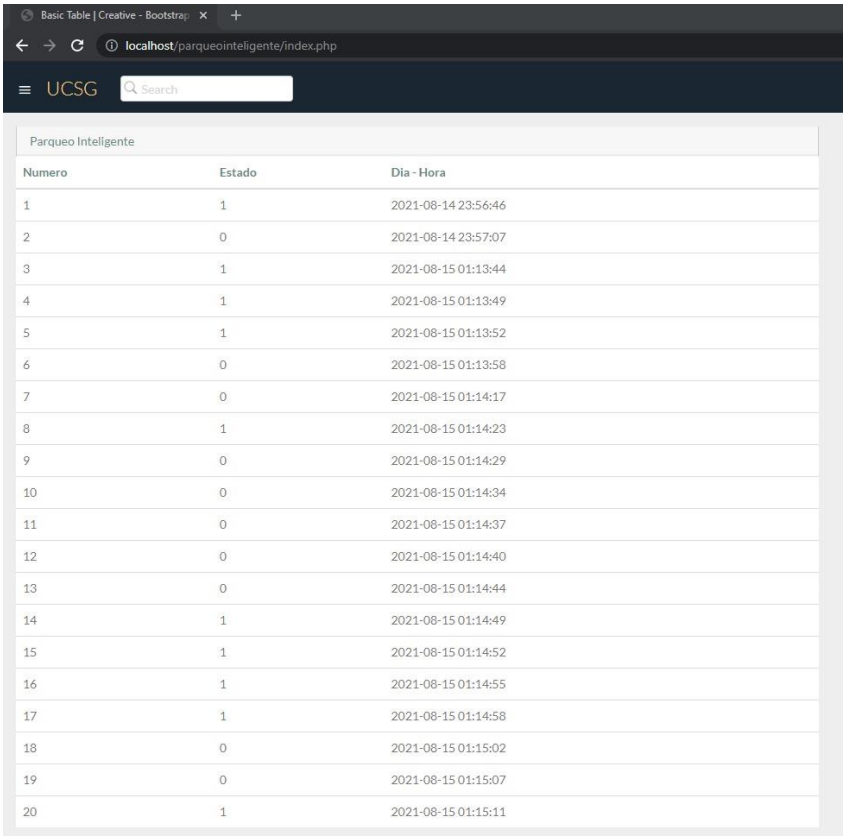


Figura 0-7 Configuración PHP
Fuente: Autor

La página web se presenta en el servidor web y este se conecta a la base de datos para obtener y presentar en forma de tabla los datos de los sensores.



Numero	Estado	Dia - Hora
1	1	2021-08-14 23:56:46
2	0	2021-08-14 23:57:07
3	1	2021-08-15 01:13:44
4	1	2021-08-15 01:13:49
5	1	2021-08-15 01:13:52
6	0	2021-08-15 01:13:58
7	0	2021-08-15 01:14:17
8	1	2021-08-15 01:14:23
9	0	2021-08-15 01:14:29
10	0	2021-08-15 01:14:34
11	0	2021-08-15 01:14:37
12	0	2021-08-15 01:14:40
13	0	2021-08-15 01:14:44
14	1	2021-08-15 01:14:49
15	1	2021-08-15 01:14:52
16	1	2021-08-15 01:14:55
17	1	2021-08-15 01:14:58
18	0	2021-08-15 01:15:02
19	0	2021-08-15 01:15:07
20	1	2021-08-15 01:15:11

Figura 0-8 Página web que indicaría los espacios disponibles y ocupados
Fuente: Autor

3.3. Diagrama de bloques

El siguiente diagrama ilustra los diferentes componentes del proyecto. Consta de dos partes, la parte principal está compuesta por una placa de Arduino YUN, que recibe datos de almacenamiento y se conecta a los diferentes nodos sensores. La segunda parte es las composiciones los nodos sensores los cuales contienen a la placa Arduino Pro mini, y estos administran la data recibida por el sensor HC-SR04 como detector de vehículos. El Arduino Yun se conecta al Wi-fi que envía datos del lugar de estacionamiento al servidor web local a través de una red lan y este a la página web.

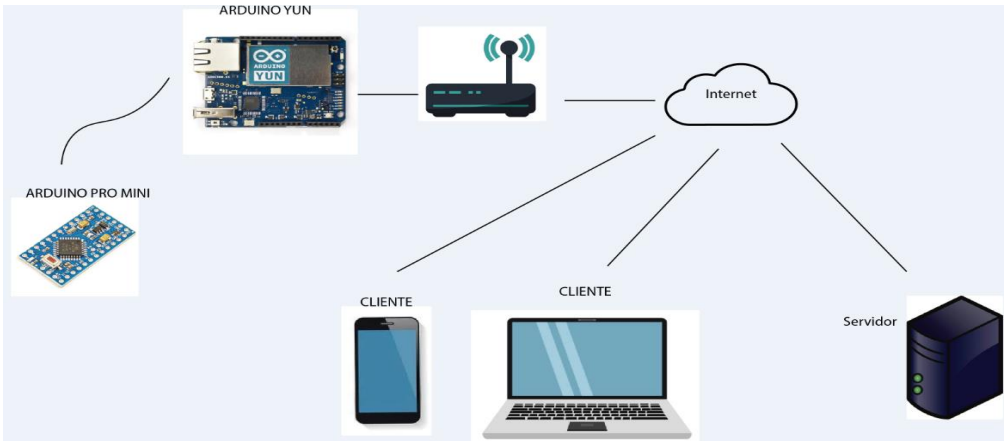


Figura 0-9 Diagrama de bloque del sistema de parqueo inteligente
Fuente: Autor

3.4. Principio de funcionamiento

El nodo sensor estará inicialmente en reposo, el cual se activará cuando empiece a recibir señal a través del sensor ultrasónico lo que hará que empiece a registrar y transmitir los valores entregados por cada sensor al nodo recolector quien contiene al Arduino Yun como CPU, el cual receptara esta información y la enviará a un servidor incluido para posterior ser enviado a la nube para visualizar los datos en la página web.

El siguiente esquema representa gráficamente la posible implementación de los sensores en el parqueadero de la UCSG.

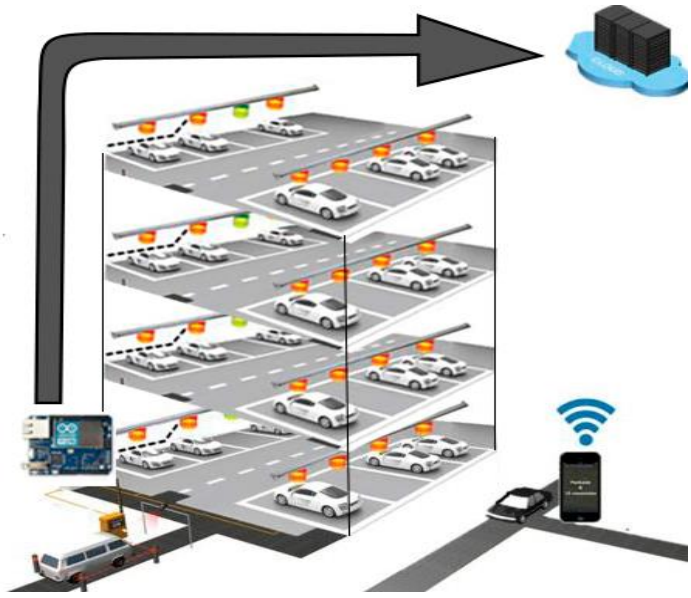


Figura 0-10 Sistema de parqueo inteligente
Fuente: Autor

3.5. Diseño esquemático en Proteus.

Los siguientes esquemas representan el diseño de la conexión del circuito del nodo sensor y nodo recolector en el software Proteus.

- **Nodo sensor**

Este sistema de nodo sensor consta de una placa Arduino Pro mini, en la cual se conectan, el sensor ultrasónico HC-SR04, Los leds de color rojo y verde y el módulo RS-485 el cual funcionará como transmisor. Las alimentaciones de VCC y GND del sensor y el módulo (línea A y línea B) deben ir conectadas al VCC y GND del Arduino Pro Mini.

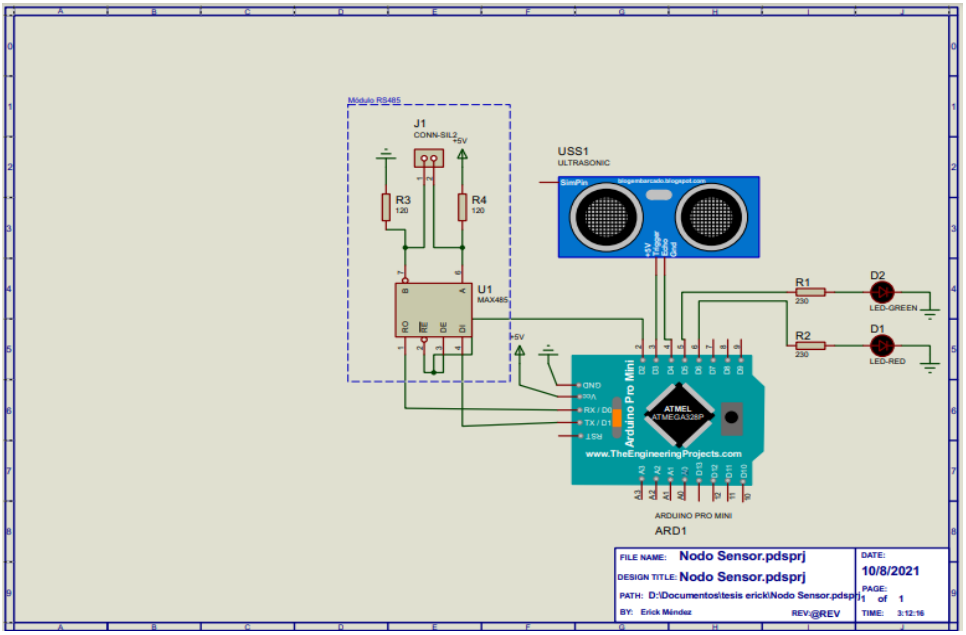


Figura 0-11 Diagrama esquemático del nodo sensor.
Fuente: Autor

- **Nodo Recolector**

Este sistema de nodo recolector estará conformado por la placa Arduino Yun, en la cual se conecta el módulo RS-485. Las alimentaciones de VCC y GND del sensor y el módulo (línea A y línea B) deben ir conectadas al VCC y GND del Arduino Pro Mini.

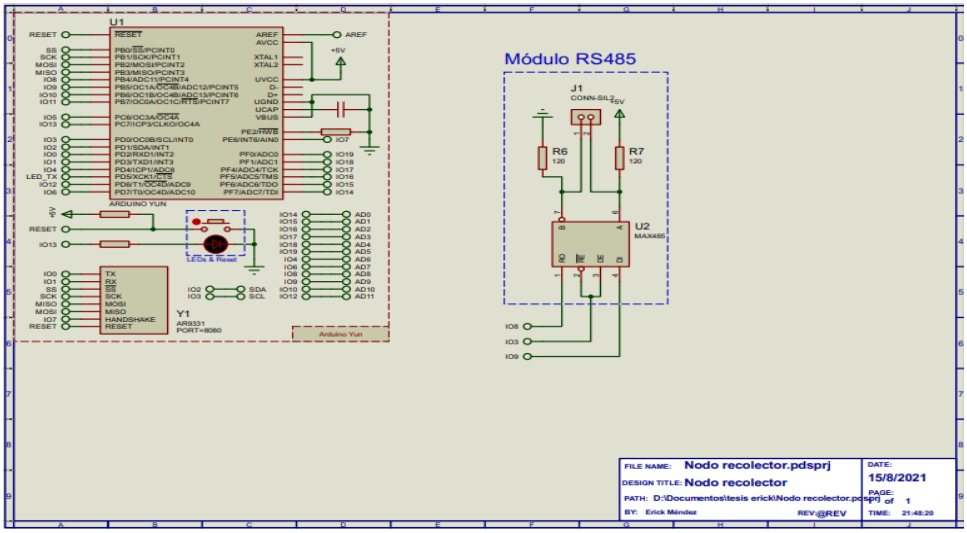


Figura 0-12 Diagrama esquemático del nodo recolector.
Fuente: Autor

- **Sistema de maestro-esclavo**

Este sistema denomina maestro al nodo recolector compuesto por el Arduino YUN, del cual habrá solo uno, y denomina esclavos a los diferentes nodos sensores que requiera el parqueadero. Cada esclavo deberá estar conectado a las líneas A y B con su respectivo modulo RS-485.

En el siguiente esquema ejemplifica la comunicación entre dos esclavos o nodos sensores conectados al mismo Bus del módulo RS-485 del maestro, pero que se puede implementar tanto esclavos como requiera el parqueadero, soportando un máximo de 256 dispositivos.

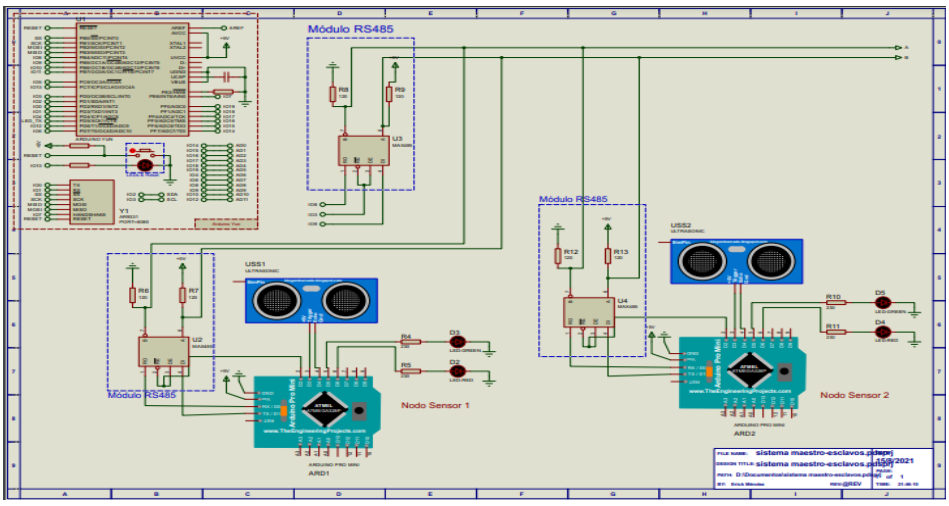


Figura 0-13 Diagrama esquemático del sistema maestro-esclavo
Fuente: Autor

3.6. Programación del microcontrolador para la recepción de la señal de los sensores, procesamiento de datos y envío de información a la página web.

Se detalla el código que puede implementarse en los esclavos y el maestro, a través del IDE de Arduino.

3.6.1. Esclavo

esclavo Arduino 1.8.9

Archivo Editar Programa Herramientas Ayuda

esclavo

```
/**
 * Tesis Erick Méndez
 * Comunicación RS-485 con MAX485 y Arduino
 * Comunicacion: Half-duplex
 * Roll: Sclavo
 * Placa: Arduino Pro Mini
 * Comandos: 0x01 - Encender led
 *           0x02 - Apagar led
 *           0x03 - leer distancia
 */

// Trama maestro-esclavo
#define HEAD 0xAA          // Inicio de trama
#define MY_SLAVE_ID 0x23   // ID del esclavo al que va dirigido el mensaje
#define TAIL 0xFE          // fin de trama

//Comandos soportados
#define CMD_LED_ON           0x01
#define CMD_LED_OFF          0x02
#define CMD_READ_DISTANCE    0x03

// Formato de Trama: <HEAD> <SLAVE_ID> <CMD> <TAIL>
// Ejemplo: 0xFF 0x34 0x01 0xFE -> Indica que el esclavo 0x34 debe ejecutar la orden 0x01

const int Trigger =3;          // Activación del pin 6 como salida para el pulso ultrasonico
const int Echo =4;             // Activación del pin 5 como entrada para el tiempo de rebote del ultrasonido
const int RS485_PIN_MODE =2;   // HIGH -> Transmisión(TX); LOW-> recepción(RX)
const int LED_Verde=5;
const int LED_Rojo=6;
byte buff[5], idx;
long distancia, tiempo;

void enviarRespuesta( float x ){
  buff[0] = HEAD;
  buff[1] = MY_SLAVE_ID;
  buff[2] = (byte)x;
  buff[3] = TAIL;
  digitalWrite( RS485_PIN_MODE, HIGH ); // poner en modo Tx
  RS485.write( buff, 4 );               // transmitir mensaje
  RS485.flush();
  digitalWrite( RS485_PIN_MODE, LOW);   // poner en modo Rx
}
```

```
void ejecutarComando(){
  Serial.println("Ejectutando comando!");
  if ( buff[1] != MY_SLAVE_ID ) // El mensaje es para otro esclavo
    return;

  switch( buff[2] ){                    // ejecutar comando

    case CMD_LED_ON:                    // Encender Led
      digitalWrite( LED_BUILTIN, HIGH );
      break;

    case CMD_LED_OFF:                   // Apagar Led
      digitalWrite( LED_BUILTIN, LOW );
      break;

    case CMD_READ_DISTANCE:
      digitalWrite(Trigger,LOW); //Para estabilizar el sensor//
      delayMicroseconds(10);
      digitalWrite(Trigger, HIGH); //envío del pulso ultrasónico//
      delayMicroseconds(10);
      tiempo= pulseIn(Echo,HIGH); //Se obtiene el ancho de pulso
      distancia= tiempo/59;       //Se escala el tiempo a una distancia en cm

       if (distancia<15){
         digitalWrite(LED_Rojo, HIGH);
         digitalWrite(LED_Verde, LOW);
         enviarRespuesta(1);        // espacio ocupado
         }
         else {
           digitalWrite(LED_Rojo, LOW);
           digitalWrite(LED_Verde, HIGH);
           enviarRespuesta(0);      // espacio disponible
         }
       break;

     default:                                 // Comando Inválido
       break;
  }
}

 void setup() {
  //Configurar Serial a 19200 baudios (para el monitor serie)
  Serial.begin(19200);

  //Configurar para utilizar el bus 485 a 9600
  RS485.begin(9600);

  //Configrar pines
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(RS485_PIN_MODE, OUTPUT);   //Pin REDE como salida
  pinMode(Echo, INPUT);
  pinMode(Trigger, OUTPUT);
  pinMode(LED_Verde, OUTPUT);        //LED Verde como salida
  pinMode(LED_Rojo, OUTPUT);         //LED Rojo como salida

  digitalWrite(Trigger, LOW);        // Se inicializa el pin con 0
  digitalWrite(LED_BUILTIN, LOW);    // Apagar led
  digitalWrite(RS485_PIN_MODE, LOW); // Poner en modo de recepción

  idx = 0;
}
```

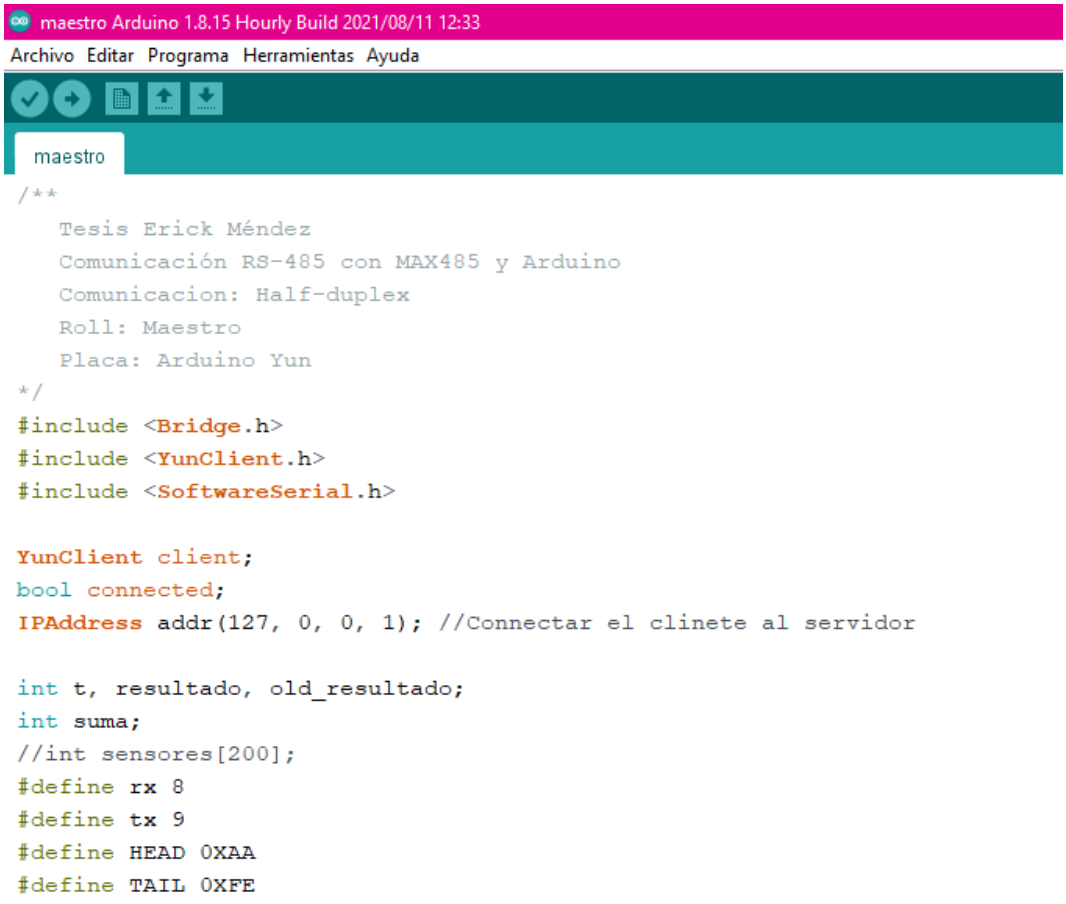
```
void loop() {
  if( !RS485.available() )
    return;
  byte incoming = RS485.read();
  Serial.print("Recibido: ");
  Serial.println(incoming);

  if( idx == 0 ){           // principio de trama
    if( incoming != HEAD ) // trama incorrecta
      return;

    buff[idx] = incoming;
    idx++;
  }
  else if ( idx > 0 && idx < 4 ){ //
    buff[idx++] = incoming;      //

    if ( idx == 4 ){                // fin de trama
      if( buff[3] == TAIL )         // verificar que termine bien
        ejecutarComando();
      idx = 0;
    } }}
```

3.6.2. Maestro




```
/**
   Tesis Erick Méndez
   Comunicación RS-485 con MAX485 y Arduino
   Comunicacion: Half-duplex
   Roll: Maestro
   Placa: Arduino Yun
*/
#include <Bridge.h>
#include <YunClient.h>
#include <SoftwareSerial.h>

YunClient client;
bool connected;
IPAddress addr(127, 0, 0, 1); //Connectar el clinete al servidor

int t, resultado, old_resultado;
int suma;
//int sensores[200];
#define rx 8
#define tx 9
#define HEAD 0XAA
#define TAIL 0XFE
```

```
//Comandos soportados
#define CMD_LED_ON    0X01
#define CMD_LED_OFF   0X02
#define CMD_READ_DISTANCE 0X03

#define RS485_PIN_MODE 3  //HIGH -> Transmisión; LOW-> recepción
//Se añade el número de esclavos o dispositivos, max 256, en esta ocasión se utiliza de ejemplo el esclavo 23
#define SLAVE 0x23
SoftwareSerial miSerial(rx, tx); //asigna los puertos seriales

byte trama[5], idx;

void enviarComando(byte esclavo, byte cmd) {
  trama[0] = HEAD;
  trama[1] = esclavo;
  trama[2] = cmd;
  trama[3] = TAIL;
  digitalWrite(RS485_PIN_MODE, HIGH); // modo tx
  Serial.write(trama, 4);
  Serial.flush();
  digitalWrite(RS485_PIN_MODE, LOW); // modo rx
}
```

```
int recibirRespuesta( byte esclavo ) {
   digitalWrite(RS485_PIN_MODE, LOW); // modo rx
   delay(2000);
   Serial.readBytes( trama, 4 );

   if ( trama[0] != HEAD )    // error en la trama
     return -1;

   if ( trama[1] != esclavo ) // respuesta de otro esclavo
     return -1;

   if ( trama[3] != TAIL )    // error en trama
     return -1;

   return trama[2];
}

void setup() {
  Bridge.begin();
  // Configurar Serial para utilizarlo como monitor
  Serial.begin(19200);

  while (!Serial) {
    digitalWrite(13, HIGH);
    delay(125);
    digitalWrite(13, LOW);
    delay(125);
  }
  Serial.println("Cliente conectado en el puerto 255");
  Connected = false;

  pinMode( RS485_PIN_MODE, OUTPUT );
  digitalWrite(RS485_PIN_MODE, LOW); // modo recepcion
  //Configurar serial para el BUS RS-485
  miSerial.begin(9600);
  old_resultado = 0;
}
```

```
void loop() {
  if (!Connected)
  {
    // No conectado actualmente, intentando estabilizar una conexion
    client.connect(addr, PORT);
    if (client.connected())
    {
      Serial.println("Conectado al server.");
      if ( Serial.available() ) {
        char c = Serial.read();
        int h;

        switch (c) {
          case '0':
            enviarComando(SLAVE, CMD_LED_ON);
            break;

          case '1':
            enviarComando(SLAVE, CMD_LED_OFF);
            break;

        case '2':
          enviarComando(SLAVE, CMD_READ_DISTANCE);
          Serial.print("Estado: ");
          t = recibirRespuesta(SLAVE);
          if ( t == -1 ) {
            Serial.println( "No se recibió' respuesta" );
          } else {
            for (int i = 2; i < 202; i++)
              // sensores[i-2]=t;
              // resultado= sensores[0]+sensores[1]+sensores[2]+sensores[3]+sensores[4]+
              // sensores[5]+sensores[6]+sensores[7]+sensores[8]+.....];
              for (int j = 0; j < 200; j++)
                Serial.println(sensores[j]);
            if (resultado = !old_resultado[j]) {
              client.println("Los parqueos disponibles son" + String(200 - resultado) + ",
                             sistema activado" + "hace" + String(time / 60000) + " minutos");
            }
            old_resultado = resultado;
            client.print("La cantidad de parqueos es:");
            client.print(200 - resultado);
            client.println("disponible");
            client.println();
            Serial.println(t);
            client.println("Conexion: cerrada");
          }
          else {
            Serial.println("Error de conexión");
            digitalWrite(13, HIGH);
            delay(3000);
          }
          if (client.connected()) {
            client.stop(); //desconectar después del envío
            Serial.println("desconnectado");
          }

          break;

        default:
          break;
      }
    }
    Connected = true;
  }
```

```
  else {
    // El intento de conexión ha fallado.
    Serial.println("No se puede conectar con el servidor.");
    delay(5000);
  }
}

if (Connected)
{
  if (client.connected())
  {
     if ( Serial.available() ) {
      char c = Serial.read();
      int h;

      switch (c) {
        case '0':
          enviarComando(SLAVE, CMD_LED_ON);
          break;

        case '1':
          enviarComando(SLAVE, CMD_LED_OFF);
          break;

       case '2':
         enviarComando(SLAVE, CMD_READ_DISTANCE);
         Serial.print("Estado: ");
         t = recibirRespuesta(SLAVE);
         if ( t == -1 ) {
           Serial.println( "No se recibió' respuesta" );
         } else {
           for (int i = 2; i < 202; i++)
             // sensores[i-2]=t;
             // resultado= sensores[0]+sensores[1]+sensores[2]+sensores[3]+sensores[4]+
             // sensores[5]+sensores[6]+sensores[7]+sensores[8]+.....];
             for (int j = 0; j < 200; j++)
               Serial.println(sensores[j]);
           if (resultado = !old_resultado[j]) {
             client.println("Los parqueos disponibles son" + String(200 - resultado) + ",
                            sistema activado" + "hace" + String(time / 60000) + " minutos");
           }
           old_resultado = resultado;
           client.print("La cantidad de parqueos es:");
           client.print(200 - resultado);
           client.println("disponible");
           client.println();

             Serial.println(t);
             client.println("Conexion: cerrada");
           }
           else {
             Serial.println("Error de conexión");
             digitalWrite(13, HIGH);
             delay(3000);
           }
           if (client.connected()) {
             client.stop(); //desconectar después del envío
             Serial.println("desconnectado");
           }
           break;

         default:
           break;
       }
     }
     delay(1000);
     while (client.available())
     {
```

```
        char c = client.read();
        Serial.print(c);
      }
      Serial.flush();
    }
    else
    {
      Serial.println("Conexión del servidor cerrada!");
      // Limpiar la conexión del cliente
      client.stop();
      Connected = false;
    }
  }
}
```

3.7. Localización de los sitios adecuados del parqueadero donde se implementarán los sensores.

De acuerdo con el segundo objetivo específico, el parqueadero de la UCSG se encuentra ubicado geográficamente dentro de la institución en la Av. Carlos Julio Arosemena km 1 1/2 Vía Daule.

Figura 0-14 Localización geográfica del parqueadero de la UCSG
Fuente: (Google, 2021)

Figura 0-15 Parqueadero de la UCSG
Fuente: (WAZE, 2021)

En la Tabla 0.10 se detallan los datos donde se encontraría ubicado el sistema de parqueo inteligente para la UCSG.

Tabla 0.10 Ubicación del sistema de parqueo inteligente para la UCSG

Provincia	Guayas
Cantón	Guayaquil
Parroquia	Tarqui
Sector	Parqueadero de la UCSG
Coordenadas	-2.182114438776797, -79.90505689992571
Referencia	Al frente de la facultad de Economía, entre la facultad de Medicina y la de Ingeniería.

Fuente: Autor, 2021.

3.8. Elaborar un presupuesto de la implementación de este proyecto de sensores para un estacionamiento inteligente.

En busca de una solución efectiva y con el propósito de implementar un sistema de parqueo inteligente para la Universidad Católica Santiago de Guayaquil, se elabora un presupuesto de costos de los elementos necesarios, sin incluir la mano de obra.

La siguiente tabla, muestra el valor unitario y total de los elementos requeridos para este sistema de parqueo inteligente con unidades de valor en dólares estadounidenses. Las cantidades de los dispositivos que requerimos puede variar en el momento que se desee implementar a más parqueadores de la Universidad.

Tabla 0.11 Presupuesto general para el diseño del sistema de parqueo inteligente para la UCSG.

Detalle	Cantidad	Valor Unitario ($)	Valor total ($)
Arduino Yun	1	70,86	70,86
Arduino Pro Mini	200	1,60	320,00
Sensor Ultrasónico HCSR04	200	2,25	450,00
Resistor 230 Ohm	400	0,05	20,00
Resistor 120 ohm	400	0,05	20,10
LEDs color rojo y verde	400	0,05	20,00
Módulo RS-485	400	1,50	600,00
Total Aproximado			1500,96

Fuente: Autor

Conclusiones

Se seleccionaron los elementos necesarios tanto en software como Hardware para llevar a cabo el sistema de parqueo inteligente de las cuales se especificaron las características técnicas de cada elemento.

La plataforma IoT selecciona fue crear un servidor web que almacene la base de datos en la nube y diseñar una página web llamativa que indique a los usuarios

Se determinó que el sitio adecuado para implementar los sensores ultrasónicos es colocarlos en el techo, protegidos en cajas de paso metálicas y plásticas y mediante cableado con tuberías galvanizadas, y para proteger los leds colocar cajas reflectoras que además servirán para una mejor visualización.

El software Proteus fue una herramienta ingenieril útil para plantear la conexión del circuito, ya que contiene librerías de Arduino que la hacen amigable de entender la conexión de pines para que se establezca la comunicación entre microcontroladores.

La programación en ambos microcontroladores se la realizó con el IDE de Arduino la cual es accesible de entender y permite al usuario tener una idea de la lógica implementada para el diseño del sistema de parqueo inteligente.

La implementación de este proyecto puede contribuir a la reducción la huella de carbono de los automóviles y motocicletas que buscan estacionarse en el parqueadero principal de la UCG, ya que el usuario ahorra tiempo de búsqueda para estacionarse.

Se realizó una tabla de presupuesto para el diseño de un sistema inteligente para el parqueadero de la UCSG a través de Arduino y el Internet de las Cosas (IoT) para 4 niveles del edificio, por lo que, si se requiere ampliar el parqueadero se debe considerar que el presupuesto aumentará y posiblemente se necesite otro tipo de comunicación o que la CPU del Arduino Yun se reemplace por una Raspberry.

Recomendaciones

Si este proyecto se desea implementar para un parqueadero más pequeño, se pueden utilizar otros elementos más económicos como sensores IR que poseen un rango de medición menor.

Para el diseño e implementación de la página web se requiere de ingenieros en sistemas que manejen programación en HTML, JS Y CSS que conecte el servidor con la base de datos almacenada.

Para el mantenimiento de comunicación, se podría utilizar un software dedicado para rastrear y analizar la actividad del puerto serial y así detectar error y comparar los datos recibidos.

Dado que el Arduino Yun tiene una memoria no volátil que soporta una cantidad limitada de escrituras, se recomienda usar una tarjeta microSD para guardar los datos.

Si este sistema de parqueo inteligente se desea implementar en estacionamientos abiertos y de igual o mayor capacidad, es aconsejable implementar la tecnología Zigbee ya que esta no requiere de utilizar cableado.

Referencias

INEC. (2020).

Arduino CC. (2021). *Arduino Pro Mini.* Obtenido de Arduino: https://www.arduino.cc/en/pmwiki.php?n=Main/ArduinoBoardProMini

Arduino CC. (2021). *Arduino YUN*. Obtenido de https://store.arduino.cc/usa/arduino-yun

Barrio, A. (2018). *Internet de las Cosas.* Madrid: REUS.

BIBING. (2020). Obtenido de BIBING: http://bibing.us.es/proyectos/abreproy/40048/fichero/VOLUMEN+1.+MEMORIA%252F4.+Tecnolog%C3%ADa+Bluetooth.pdf

BOSCH. (2002). *Los sensores en el automóvil.*

Bragós, R. (1999). *Circuitos y dispositivos electrónicos. Fundamentos de electrónica (PT).* Universidad Politécnica de Catalunya.

CISCO. (2021). *Transmisión de datos en la red.* Obtenido de CISCO: http://www.cca.org.mx/profesores/abc/pdfs/cisco/cisco_0.pdf

CISCO. (2021). *What Is Wi-Fi?* Obtenido de CISCO: https://www.cisco.com/c/en/us/products/wireless/what-is-wifi.html

Cobo, A. (2009). *Estudio científico de las redes de ordenadores.* Madrid: Visión Libros.

Coronel, V., & Tenelanda, D. (2016). *Análisis de interoperabilidad de plataformas IoT aplicado al desarrollo de un sistema de monitoreo de polución de aire para la ESPOCH.* Obtenido de Repositorio de la Escuela Superior Politécnica de Chimborazo: http://dspace.espoch.edu.ec/handle/123456789/5440

Domínguez, E., & Ferrer, J. (2012). *Circuitos eléctricos auxiliares del vehículo.* Editex. Obtenido de https://books.google.com.ec/books?id=WfXEAwAAQBAJ&pg=PA79&dq=sensores+activos+y+pasivos&hl=es-419&sa=X&ved=2ahUKEwjzvYmKoJ7yAhVpQzABHUgBDhsQ6AEwCHoECAgQAg#v=onepage&q=sensores%20activos%20y%20pasivos&f=false

Ecobadajoz. (2021). *Ecobadajoz*. Obtenido de https://www.ecobadajoz.es/resistencias-y-potenciometros/resistencia-de-carbon-05w-220-ohm.html

EETech Media, LLC. . (2021). *Carbon film resistor*. Obtenido de EEPower: https://eepower.com/resistor-guide/resistor-materials/carbon-film-resistor/

Escolano, F., Cazorla, M., Alfonso, M., Colomina, O., & Lozano, M. (2003). *Inteligencia artificial: modelos, técnicas y áreas de aplicación.* Paraninfo.

Fowler, R. (1994). *Electricidad principios y aplicaciones.* Reverte.

García, A., & Castillo, F. (2007). *Cim: El computador en la automatización de la producción.* Cuenca: Universidad de Castilla- La Mancha.

Geek Factory. (2021). *Microcontrolador Atmega328P* . Obtenido de https://www.geekfactory.mx/tienda/componentes-electronicos/circuitos-integrados/atmega328p-microcontrolador-avr/

Geek Factory. (2021). *Pinout diagram Arduino Yun*. Obtenido de https://www.geekfactory.mx/tutoriales/tutoriales-arduino/diagrama-de-pines-arduino-pinout-arduino/

Google. (2021). *Google Maps*. Obtenido de https://www.google.com/maps/place/Parqueo+de+la+UCSG/@-2.1821869,-79.9057361,373m/data=!3m2!1e3!4b1!4m5!3m4!1s0x902d6de4545bacf3:0x8d67354892d234d9!8m2!3d-2.1821886!4d-79.9050467

Guerra, J. (2021). *RS-485 y MAX485 comunicación serie entre Arduinos*. Obtenido de https://programarfacil.com/blog/arduino-blog/rs-485-max485-arduino/

HUBOR. (2015). Proteus.

Izan, L. (2015). *LEDs bicolor* . Obtenido de IZANTECH: http://blog.izantech.com/2015/10/leds-bicolores.html

Lang, T. (2017). *Why Amazon's Alexa could be the death of meetings.* Obtenido de Fortune: https://fortune.com/2017/06/11/amazon-alexa-echo-personal-assistant-skills-meetings/

Lee, S. K., Kwon, H. R., Cho, H., Kim, J., & Lee, D. (Junio de 2016). *International Case Studies of Smart Cities. Orlando, United States of America.* Obtenido de https://publications.iadb.org/publications/english/document/International-Case-Studies-of-Smart-Cities-Songdo-Republic-of-Korea.pdf

Lindsay, K. (2017). *Why IoT and personalization are made for each other.* Obtenido de CMSWire: https://www.cmswire.com/digital-experience/why-iot-and-personalization-are-made-for-each-other/

Llamas, L. (2016). *Diagrama de pin del módulo RS-85*. Obtenido de https://www.luisllamas.es/arduino-rs485-max485/

Manrique, J. (2011). *Diseño de edificio de parqueos en campus de la Universidad Católica de Santiago de Guayaquil, costos y prefactibilidad del proyecto.* Obtenido de Repositorio UCSG: http://repositorio.ucsg.edu.ec/bitstream/3317/1197/1/T-UCSG-PRE-ING-IC-36.pdf

Mecatrónica Latam. (2021). *Mecatrónica Latam*. Obtenido de https://www.mecatronicalatam.com/es/tutoriales/electronica/componentes-electronicos/diodo/diodo-led/

Mok, K. (2017). *Prototype: A smartphone universal remote for the Internet of Things.* Obtenido de Thenewstack: https://thenewstack.io/smartphone-prototype-acts-like-universal-remote-internet-things/

Moreno, J., & Ruiz, D. (2007). *Informe Técnico: Protocolo ZigBee (IEEE 802.15.4).*

Naylamp Mechatronics SAC. (2021). *DFROBOT*. Obtenido de https://naylampmechatronics.com/sensores-proximidad/10-sensor-ultrasonido-hc-sr04.html

Naylamp Mechatronics SAC. (2021). *Módulo Conversor RS485 a Serial TTL*. Obtenido de Naylamp Mechatronics: https://naylampmechatronics.com/comunicacion/62-conversor-rs485-a-serial-ttl.html

Novillo, J., Hernández, D., Mazón, B., Molina, J., & Cárdenas, O. (2018). *Arduino y el Internet de las cosas.* Área de Innovación y Desarrollo,S.L.

Perez, A. D. (2016). *Protocolos de comunicación.* Universidad Nacional de la Plata. Obtenido de file:///D:/Documentos/tesis%20erick/Tesis.%20Protocolos%20de%20comunicaci%C3%B3n%20entre%20microcontroladores.pdf-PDFA.pdf

Quiñonez, O. (2019). *Internet de las Cosas.* IBUKKU.

Rodríguez, R. (2019). *Internet de las cosas: Futuro y desafío para la epidemiología y la salud pública.* doi:https://doi.org/10.22267/rus.192103.162

Rossano, V. (2009). *Electrónica y microcontroladores PIC.* USERSHOP.

SAP. (2020). *¿Qué es internet de las cosas (IoT)?* Obtenido de SAP Insights: https://www.sap.com/latinamerica/insights/what-is-iot-internet-of-things.html

Shoup, D. (2006). *Cruising for parking.* doi:10.1016/j.tranpol.2006.05.005

The Engineering Projects. (2021). *Diagrama de Pin del Arduino Pro mini*. Obtenido de https://www.theengineeringprojects.com/2018/06/introduction-to-arduino-pro-mini.html

Unit Electronics. (2021). *Unit Electronics*. Obtenido de https://uelectronics.com/producto/oscilador-cristal-cuarzo-hc-49s/

Valarezo, L. (2018). *Modelo de gestión de la vinculación universitaria de la carrera de Ingeniería Civil de la Universidad Católica de Santiago de Guayaquil.* Obtenido de Revista Espacios: http://www.revistaespacios.com/a18v39n52/a18v39n52p22.pdf

Valderrama, J. O. (1999). Información Tecnológica. *Centro de Información Tecnológica*, 390.

WAZE. (2021). *Waze: Navegación y tráfico real*. Obtenido de https://venue-image.waze.com/thumbs/thumb700_feaa5074-3268-4127-b54e-2fcd912ae28b

Anexos

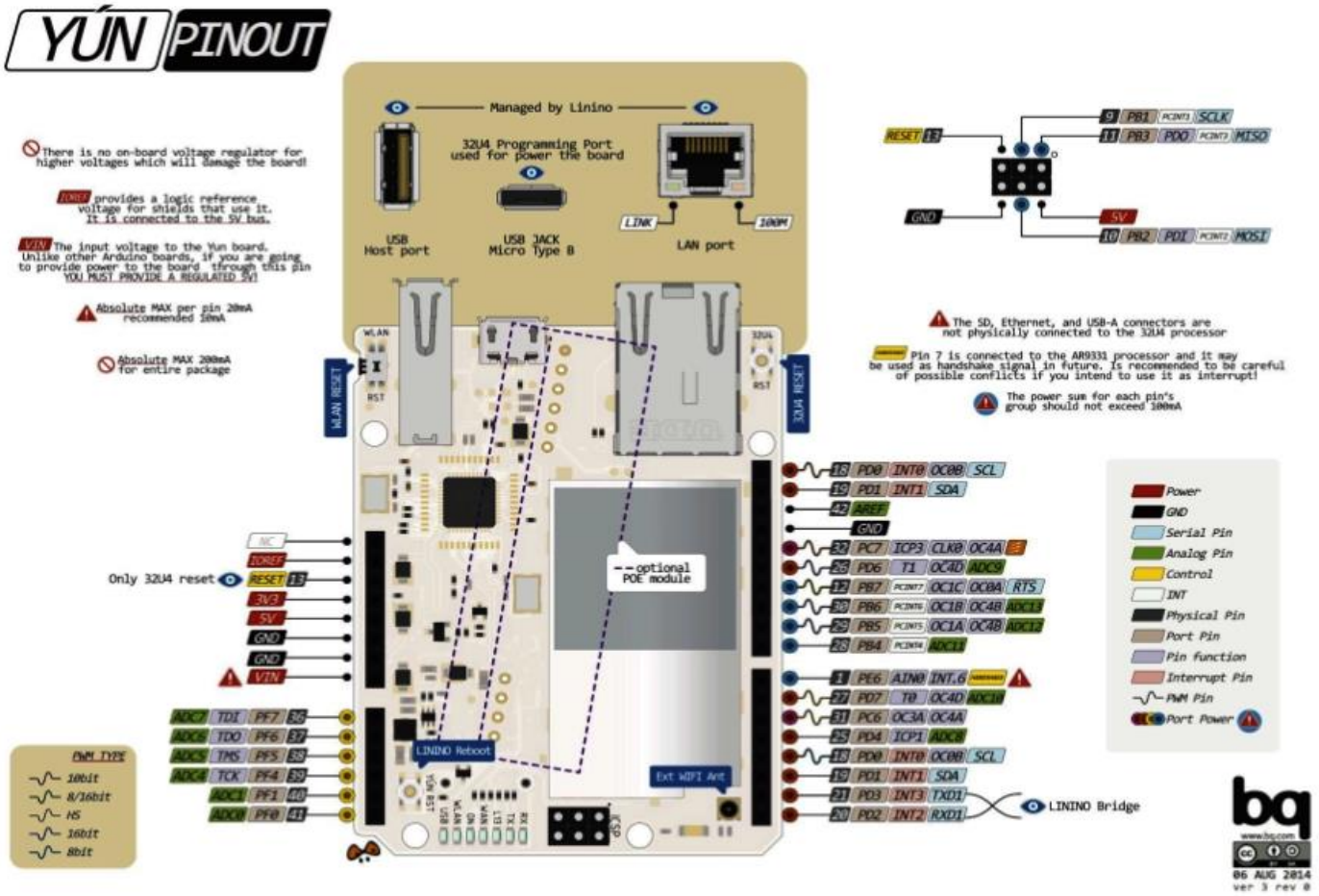


Figura 0-1 Diagrama de Pin del Arduino Yun
Fuente: (Geek Factory, 2021)

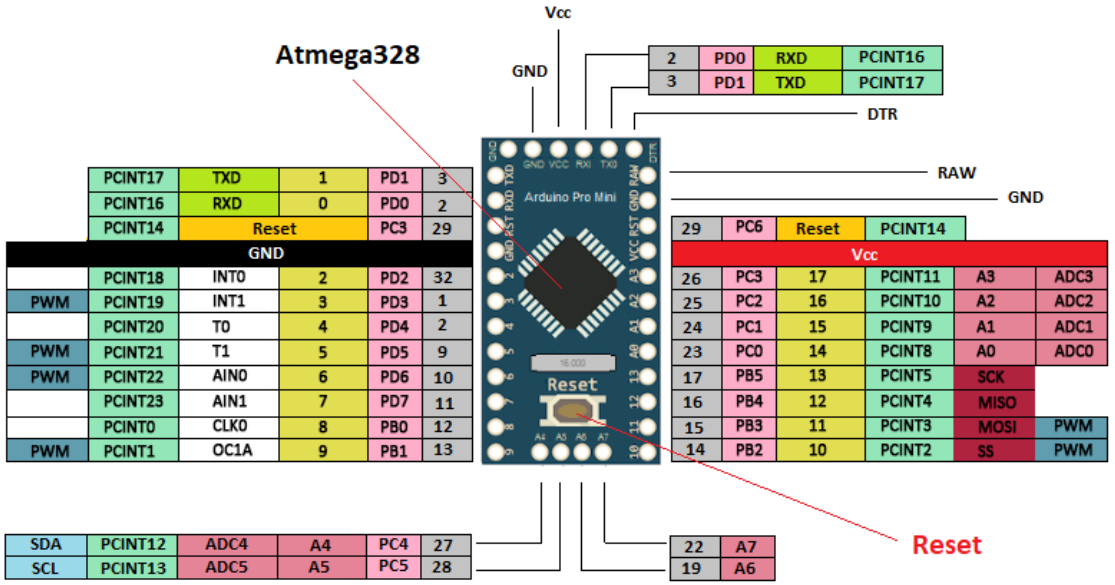


Figura 0-2 Diagrama de Pin del Arduino Pro mini
Fuente: (The Engineering Projects, 2021)

Figura 0-3 Pines del Módulo RS-485
Fuente: (Llamas, 2016)

DECLARACIÓN Y AUTORIZACIÓN

Yo, **Méndez Ugarte, Erick Fernando** con C.C: # 0927599050 autor del Trabajo de Titulación: **Estudio y diseño de un parqueo inteligente utilizando Arduino a través del Internet de las cosas (IoT),** previo a la obtención del título de **INGENIERO EN TELECOMUNICACIONES** en la Universidad Católica de Santiago de Guayaquil.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Guayaquil, 15 de septiembre del 2021

f. _____

Nombre: Méndez Ugarte, Erick Fernando

C.C: 0927599050

REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA			
FICHA DE REGISTRO DE TESIS/TRABAJO DE TITULACIÓN			
TÍTULO Y SUBTÍTULO:	Estudio y diseño de un parqueo inteligente utilizando Arduino a través del Internet de las cosas (IoT)		
AUTOR(ES)	Méndez Ugarte, Erick Fernando		
REVISOR(ES)/TUTOR(ES)	M. Sc. Romero Paz, Manuel De Jesús		
INSTITUCIÓN:	Universidad Católica de Santiago de Guayaquil		
FACULTAD:	Facultad de Educación Técnica para el Desarrollo		
CARRERA:	Ingeniería en Telecomunicaciones		
TITULO OBTENIDO:	Ingeniero en Telecomunicaciones		
FECHA DE PUBLICACIÓN:	15 de septiembre del 2021	**No. DE PÁGINAS:**	65
ÁREAS TEMÁTICAS:	Internet de las Cosas, Hardware Libre, Software Libre, Microcontroladores		
PALABRAS CLAVES/KEYWORDS:	IoT, Arduino, RS-485, Microcontroladores, Protocolos, Proteus		

RESUMEN/ABSTRACT:
En el presente trabajo de titulación, se diseñó un sistema de parqueo inteligente con Arduino a través del internet de las cosas (IoT) para el futuro control de espacios disponibles u ocupados en el parqueadero de la Universidad Católica Santiago de Guayaquil. En el segundo capítulo se dio a conocer la fundamentación teórica de la tecnología IoT además de los sensores, placas de Arduino, y la comunicación que se puede dar entre ambas placas, por lo cual en el tercer capítulo se estableció un medio físico y un protocolo de comunicación, el cual fue el RS-485 con tipo de comunicación Half-Duplex para ambos Arduinos, bajo el concepto de maestro y esclavos para dar el funcionamiento del sistema. En el diseño de los esclavos se estableció que estos contienen un sensor ultrasónico para detectar la presencia del vehículo conectados a la placa de Arduino Pro mini, mientras que el maestro es una placa de Arduino Yun que recibe la información de los esclavos y la envía a una página web, por lo cual, se creó un servidor web local para almacenar la base de datos y enviar dicha información por medio de una red LAN a la página web. Con este diseño de sistema se pudo prever que su implementación podría reducir la huella de carbono de los vehículos ya que se optimizaría el tiempo de los estudiantes al estacionarse, además se elaboró un presupuesto de los elementos necesarios para el diseño de este sistema.

ADJUNTO PDF:	☒ SI	☐ NO
CONTACTO CON AUTOR/ES:	**Teléfono:** +593988608253	E-mail: nericknm@hotmail.com
CONTACTO CON LA INSTITUCIÓN:	**Nombre:** Palacios Meléndez, Edwin Fernando	
COORDINADOR DEL PROCESO DE UTE	**Teléfono:** 593-967608298	
	E-mail: edwin.palacios@cu.ucsg.edu.ec	

SECCIÓN PARA USO DE BIBLIOTECA	
Nº. DE REGISTRO (en base a datos):	
Nº. DE CLASIFICACIÓN:	
DIRECCIÓN URL (tesis en la web):	