UNIVERSIDAD MAYOR DE SAN ANDRÉS
FACULTAD DE CIENCIAS PURAS Y NATURALES
CARRERA DE INFORMÁTICA

TESIS DE GRADO

TÍTULO: "SISTEMA DE CONTROL DE TRÁFICO VEHICULAR EN BASE A INTERNET DE LAS COSAS"

PARA OPTAR AL TÍTULO DE LICENCIATURA EN INFORMÁTICA
MENCIÓN: INGENIERÍA DE SISTEMAS INFORMÁTICOS

POSTULANTE: Ricardo Esteban Pari Cahuna

TUTOR METODOLÓGICO: M.Sc. Franz Cuevas Quiroz

ASESOR: Lic. German Huanca Ticona

NUESTRA SEÑORA DE LA PAZ – BOLIVIA

2017

UNIVERSIDAD MAYOR DE SAN ANDRÉS
FACULTAD DE CIENCIAS PURAS Y NATURALES
CARRERA DE INFORMÁTICA

LA CARRERA DE INFORMÁTICA DE LA FACULTAD DE CIENCIAS PURAS Y NATURALES PERTENECIENTE A LA UNIVERSIDAD MAYOR DE SAN ANDRÉS AUTORIZA EL USO DE LA INFORMACIÓN CONTENIDA EN ESTE DOCUMENTO SI LOS PROPÓSITOS SON ESTRICTAMENTE ACADÉMICOS.

LICENCIA DE USO

El usuario está autorizado a:

a) visualizar el documento mediante el uso de un ordenador o dispositivo móvil.
b) copiar, almacenar o imprimir si ha de ser de uso exclusivamente personal y privado.
c) copiar textualmente parte(s) de su contenido mencionando la fuente y/o haciendo la referencia correspondiente respetando normas de redacción e investigación.

El usuario no puede publicar, distribuir o realizar emisión o exhibición alguna de este material, sin la autorización correspondiente.

DEDICATORIA

A mi papa Donato Pari, mi mama Trinidad Cahuna, mis hermanos y hermana, porque siempre estuvieron a mi lado incondicionalmente y me brindaron la fortaleza para poder culminar con éxito este proyecto.

AGRADECIMIENTO

A mi tutor M.Sc. Franz Cuevas Quiroz que me ha colaborado con su capacidad y modesto conocimiento en realización de la presente tesis.

Le doy un profundo agradecimiento al Lic. German Huanca Ticona, mi revisor por sus consejos tiempo y paciencia al revisar cada uno de los avances de la tesis.

Agradecer a toda la comunidad de software y hardware libre por compartir sus conocimientos con el resto del mundo

Finalmente agradecer a todos los docentes y compañeros estudiantes de la carrera de Informática y a la Universidad Mayor de San Andrés por permitir mi formación como profesional.

Ricardo Esteban Pari Cahuna

RESUMEN

La congestión de los vehículos es quizás el mayor problema relacionado con el transporte que los habitantes de las grandes metrópolis deben afrontar día a día, generando problemas como ser contaminación ambiental, contaminación acústica, estrés en los conductores, etc.

El trabajo presentado pretende atacar este problema, que consiste en proporcionar datos exactos sobre el flujo vehicular para su posterior utilización en los tiempos de los semáforos en una intersección. Esto se logra aprovechando las nuevas tecnologías de internet de las cosas IoT mediante el monitoreo y el intercambio en tiempo real de la densidad de tráfico en cada una de las calles que convergen en la intersección y con base en estos datos, brindar una información precisa para la toma de decisiones de los semáforos.

El sistema propuesto se basa en un enfoque cliente servidor donde cada grupo de sensores ubicados en una determinada calle envían información exacta al servidor de los vehículos que están en la calle como ser, número de vehículos, tipos de vehículos, prioridad de cada vehículo, datos del propietario, datos del vehículo como placa, etc.

El sistema de control de tráfico vehicular fue implementado utilizando placas Arduino uno, sensores, actuadores, nodeMCU, módulos RFID, modulo Wifi ESP 8266 complementado por un sistema web con arquitectura MEAN stack, mongoDB, Express, Angular 4 y NodeJS.

El prototipo se probó usando una maqueta y un modelo grafico representativo del tráfico vehicular con el software VISSIM lo cual demostró que el prototipo devuelve datos mucho más detallados por nodo al 95%, del flujo vehicular en comparación del software que devuelve datos más generales y menos detallados como ser cantidad de vehículos, tipos de vehículos, etc.

SUMMARY

The congestion of vehicles is perhaps the biggest problem related to the transportation that the inhabitants of the big metropolis have to face day to day, generating problems such as environmental pollution, noise pollution, driver stress, and so on.

The work presented aims to tackle this problem, which is to provide accurate data on the vehicular flow for later use in traffic light times at an intersection. This is achieved by taking advantage of the new internet technologies of IoT things by monitoring and real-time exchange of traffic density in each of the streets that converge at the intersection and based on this data, provide accurate information for The decision making of the traffic lights.

The proposed system is based on a client server approach where each group of sensors located in a certain street send exact information to the server of the vehicles that are in the street as being, number of vehicles, types of vehicles, priority of each vehicle, data Of the owner, vehicle data as a plate, etc.

The vehicle traffic control system was implemented using Arduino 1, sensors, actuators, nodeMCU, RFID modules, Wifi ESP 8266 module complemented by a web system with architecture MEAN stack, mongoDB, Express, Angular 4 and NodeJS.

The prototype was tested using a model and graphic model representative of vehicular traffic with the VISSIM software which demonstrated that the prototype returns much more detailed data per node to 95% of the vehicular flow in comparison to the software that returns more general data and less Such as quantity of vehicles, types of vehicles, etc.

INDICE GENERAL

CAPITULO I **Pag.**

CAPITULO II

CAPITULO V

ÍNDICE DE FIGURAS

Pag.

INDICE DE TABLAS

Pag.

CAPITULO I

INTRODUCCIÓN

La congestión de tránsito ha ido en aumento en gran parte del mundo, desarrollado o no, y todo indica que seguirá agravándose, constituyendo un peligro cierto que se cierne sobre la calidad de vida urbana. El explosivo aumento del parque de automóviles y el indiscriminado deseo de usarlos, por razones de comodidad o estatus, especialmente en los países en desarrollo, ejercen una gran y creciente presión sobre la capacidad de las vías públicas existentes.

Los fuertes impactos negativos de la congestión, tanto inmediatos como de largo plazo, exigen esfuerzos multidisciplinarios para mantenerla bajo control, mediante el diseño de políticas y medidas apropiadas, no siendo sencillo encontrar las soluciones más indicadas. Todo señala que debe intentarse un conjunto de acciones sobre la oferta de transporte, así como sobre la demanda, a fin de racionalizar el uso de las vías públicas.

El control de la congestión forma parte de la elaboración de una visión estratégica de largo plazo del desarrollo de una ciudad, que permita compatibilizar la movilidad, el crecimiento y la competitividad, tan necesarias actualmente, con la sostenibilidad de la urbe y su calidad de vida. El tema es complicado y exige una alta capacidad profesional y de liderazgo de parte de las autoridades urbanas y de transporte. El trabajo ha de ser continuo y permanente.

En los últimos tiempos han aparecido diversas plataformas de desarrollo tanto hardware como software con un coste muy reducido y con gran documentación y una comunidad muy activa que permiten la realización de proyectos de diverso tipo que hace tiempo serían impensables su realización a pequeña escala.

Tanto la plataforma Arduino como Raspberry Pi disponen de una gran y activa comunidad donde se puede encontrar mucha documentación y ayuda del resto de usuarios, facilitando de esta forma la realización de los proyectos deseados. De la misma forma el módulo wifi esp8266 también ha generado una gran comunidad en especial por su bajo coste comparado con el resto de alternativas existentes.

Se realizará un prototipo funcional de un sistema web basado en internet de las cosas IoT (Internet of Things) que pueda regular el tráfico vehicular controlando y brindando información exacta de los vehículos que están en un determinado flujo pensando en su posible posterior ampliación con nuevas características y funciones o a su unión con otros proyectos más específicos.

1.1. ANTECEDENTES

1.1.1. Compania Sic TransCore - SicFlotas

El seguimiento de vehículos y cargas es decisivo a la hora de brindar un buen servicio al cliente y operar eficientemente con el transporte.

SicFlotas es un sistema de identificación vehicular basado en el uso de tecnología inalámbrica (lectores de radio frecuencia (RFID) y etiquetas), que permiten a las compañías con flotas de vehículos realizar una identificación de la unidad o el conductor.

Luego, se ubican las antenas lectoras en puntos estratégicos. Ejemplos: Accesos, estaciones, puertas, sitios de carga de combustible o de mantenimiento. Cuando la etiqueta pasa por la zona de lectura identifica al móvil y el lector retransmite la información programada

al centro de control. La etiqueta está especialmente diseñada para aplicaciones en las que se opera con rangos largos y admite la exposición a ambientes severos.

Figura 1.1: Detección de camiones de SicFlotas

Fuente: www.smartpass.com

De las características más relevantes que se pueden apreciar en esta Figura destacan:

- El uso de dos frecuencias 902 a 904 Mhz ó 909.75 a 921.75 Mhz.
- Alcance regulable para etiquetas con batería.
- Uso de etiquetas sin batería (opcional).
- Superficie de montaje no metálica.

Figura 1.2: Especificaciones Técnicas del lector SicFlotas

Fuente: www.smartpass.com

1.1.2. Compañía GIGA–TMS

Otra empresa que se encarga también de crear lectores para la detección de vehículos, es GIGA–TMS con un modelo GP90-A, controla el acceso, con detección de vehículos a una distancia de 90 centímetros. Utiliza etiquetas RFID de 64 bits que posee un led indicador de encendido y lectura, posee un diseño capaz de soportar cambios de temperatura.

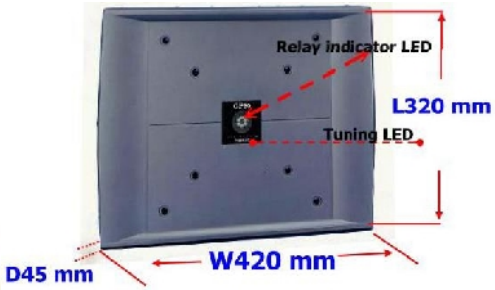

Figura 1.3: Lector GP90-A de la compañía GIGA-TMS

Fuente: www.giga-tms.com

Tabla 1.1: Especificaciones Técnicas del lector GP90-A

Especificaciones:	
Interfaces	MSR ABA Track2 RS232 RS485 O especificaciones especiales para los clientes.
Rango de lectura	Más de 90 cm. con tarjeta ISO en condiciones ideales o más de 130 cm. con tarjeta clamshell especial.
Tarjetas de RFID aceptadas	125 Khz., 64 bits, codificación Manchester.
Nivel de energía	Más de 24 V / 2 A
Temperatura de trabajo	-10 a +60°C
Indicaciones Audio/Visuales	Luz azul brillante y timbre
Dimensiones	420 (W) x 320 (L) x 45 (D) mm.

Fuente: www.giga-tms.com

1.1.3. Limitaciones de los trabajos relacionados

Dentro de las limitaciones que se detectan en las dos empresas es que están diseñadas para montarse en camiones de carga, con la ayuda de placas (que no es muy práctico ya que debe ser portable para aquellos vehículos que no requieren de una etiqueta fija).

Para el caso de la compañía Sic TransCore – SicFlotas:

El principal problema que se encontró fue la estandarización, debido a que Sic TransCore no sigue un estándar (por ejemplo: EPC), con ello lograr que el software, sea aceptado fácilmente por parte de los usuarios, debido a que podrá hacer uso de etiquetas o lectores debidamente documentados, probados por empresas que trabajen con el mismo estándar.

Para el caso de la compañía GIGA–TMS:

El principal problema que se observó fue su rango de lectura, debido a que trabaja con etiquetas RFID de alta frecuencia (125Khz), esto hace que la distancia de lectura entre la etiqueta y el lector sea muy reducida aproximadamente un metro, aunque esta empresa asegura que su rango de lectura es de 130 centímetros., no es suficiente.

1.2. PLANTEAMIENTO DEL PROBLEMA

Uno de los grandes problemas de los países desarrollados o no, es la congestión vehicular, tomando el caso de la cuidad de La Paz informes emitidos anualmente indican que la cantidad de vehículos aumenta un 10% por año, aumentando el tiempo de viaje de un punto a otro se identificaron 7 cuellos de botellas principales de los cuales el 85% de las 17 vías conflictivas están saturadas por los vehículos de transporte público, según el ejecutivo de la organización sindical recorrer en horas pico desde la Pérez Velasco hasta la Plaza del Estudiante (siete cuadras) toma al menos 25 minutos.

A esto se suman los siguientes problemas

- Algunos conductores no respetan la luz roja de los semáforos
- No se respeta la restricción vehicular por parte de algunos conductores
- Algunos conductores de transporte público se dedican al trameaje
- Existe falta de información de la cantidad de vehículos que están en el flujo vehicular
- Existe falta de información de los vehículos que se encuentran en un flujo vehicular como ser matricula, tipo de vehículo, datos del propietario, prioridad, etc.
- No se tienen datos estadísticos de la contaminación ambiental y acústica generada en un flujo vehicular en tiempo real

Por lo tanto, el problema de investigación es el siguiente:

¿Un sistema web basado en internet de las cosas permitirá obtener datos óptimos del flujo vehicular para su posterior uso en la administración de tiempos en los semáforos?

1.3. JUSTIFICACIÓN

El uso del internet de las cosas IoT actualmente es utilizado en distintos campos. El campo del tráfico vehicular no es una excepción, en este trabajo se hace uso de actuadores y sensores para dar la información precisa y correcta para un manejo óptimo de los semáforos.

El gasto económico que implica que personas y vehículos no lleguen a su destino a tiempo tiene altos costos, es por eso que la mayoría de las personas optan por un transporte que llegue más rápido a su destino, aunque tenga mayor costo.

La implementación del internet de las cosas aplicada al campo del tráfico vehicular en una ciudad determinada reducirá estos costos haciendo que la gente reduzca gastos y tiempo.

El sistema de tráfico vehicular en base a internet de las cosas eliminara en gran medida lo que hoy se denomina violencia vial, estrés en los conductores, etc.

1.4. HIPÓTESIS

El sistema web de control de tráfico vehicular en base a internet de las cosas brinda información precisa y exacta al 95% sobre el flujo vehicular

1.5. OBJETIVOS

1.5.1. Objetivo general

Implementar un sistema web de control de tráfico vehicular utilizando internet de las cosas IoT proporcionara un beneficio en el tránsito vehicular en una ciudad.

1.5.2. Objetivos específicos

- Diseñar el registro automático de infracciones por pasarse la luz roja de los semáforos
- Implementar la restricción vehicular mediante RFID
- Diseñar el control de rutas del transporte público mediante RFID
- Implementar un circuito electrónico que permita obtener datos de la cantidad de vehículos en un flujo vehicular
- Implementar encriptación de datos en los identificadores RFID de los vehículos para obtener todos los datos correspondientes.
- Implementar un circuito electrónico que permita monitorear en tiempo real la contaminación ambiental y acústica generada en un flujo vehicular

1.6. ALCANCES

El alcance del presente trabajo de investigación comprende la recolección y manipulación de datos de los sensores y actuadores ubicados en los flujos vehiculares para brindar información para su posterior uso en un manejo adecuado de los tiempos en los semáforos.

El trabajo implica la investigación y el desarrollo del sistema web de control de tráfico vehicular basado en internet de las cosas como prototipo esto hará uso de la tecnología informática y electrónica, el resultado será un prototipo de interacción de software y electrónica mediante el cual se manipulará información sobre el flujo vehicular que los componentes electrónicos brinden al software.

1.7. APORTES

El trabajo muestra como el sistema web de tráfico vehicular basado en internet de las cosas puede colaborar de una forma importante a la solución del problema de la congestión vehicular, se espera brindar datos confiables del flujo vehicular para su posterior uso en la administración de tiempos de los semáforos y de esta manera ayudar a reducir la congestión vehicular.

Hoy en día el uso de la tecnología hace posible que muchas de las tareas que el hombre realiza manualmente las realicen la computadora automáticamente.

El alcance del internet de las cosas alcanza los más diversos campos del saber humano, el tráfico vehicular no es una excepción. El uso especialmente en este terreno es uno de los que más debe ser explotado, el proyecto surge de un problema real que se vive diariamente, su aporte se basa en la obtención de datos confiables sobre el tráfico vehicular para un posterior uso en el manejo de tiempos de los semáforos, esto involucra la reducción de la congestión vehicular que se vive diariamente.

1.8. LIMITES

Las limitaciones del presente estudio se enumeran de la siguiente manera:

- Los problemas de acontecimientos sociales marchas, bloqueos, etc. quedan fuera del alcance de la tesis
- Los eventos climáticos quedan fuera del alcance de la tesis

- La tesis no soluciona el problema de vehículos en mal estado
- Las motocicletas y otros tipos de motorizados quedan fuera del alcance de la tesis
- Las velocidades de los vehículos del flujo vehicular quedan fuera del alcance de la tesis

CAPITULO II

MARCO TEÓRICO

2.1. INTERNET DE LAS COSAS

Internet de las cosas (Internet of things, IoT) es un concepto que se refiere a la interconexión digital de objetos cotidianos con internet. Alternativamente, Internet de las cosas es el punto en el tiempo en el que se conectarían a internet más “cosas u objetos” que personas. También suele referirse como el internet de todas las cosas o internet en las cosas. Si los objetos de la vida cotidiana tuvieran incorporadas etiquetas de radio, podrían ser identificados y gestionados por otros equipos, de la misma manera que si lo fuesen por seres humanos.

El concepto de internet de las cosas lo propuso Kevin Ashton en el Auto-ID Center del MIT en 1999, donde se realizaban investigaciones en el campo de la identificación por radiofrecuencia en red (RFID) y tecnologías de sensores.

Por ejemplo, si los libros, termostatos, refrigeradores, la paquetería, lámparas, botiquines, partes automotrices, etc. estuvieran conectados a Internet y equipados con dispositivos de identificación, no existirían, en teoría, artículos fuera de stock o medicinas caducadas; sabríamos exactamente la ubicación, cómo se consumen y se compran productos en todo el mundo; el extravío sería cosa del pasado y sabríamos qué está encendido o apagado en todo momento.

El internet de las cosas debería codificar de 50 a 100.000 billones de objetos y seguir el movimiento de estos; se calcula que todo ser humano está rodeado de por lo menos 1000 a 5000 objetos. Según la empresa Gartner, en 2020 habrá en el mundo aproximadamente 26 mil millones de dispositivos con un sistema de adaptación al internet de las cosas. Abi Research, por otro lado, asegura que para el mismo año existirán 30 mil millones de dispositivos inalámbricos conectados al Internet. Con la próxima generación de aplicaciones de Internet (protocolo IPv6) se podrían identificar todos los objetos, algo que no se podía hacer con IPv4. Este sistema sería capaz de identificar instantáneamente por medio de un código a cualquier tipo de objeto.

2.1.1. Características

a) Inteligencia

El Internet de las cosas es "no determinista" y de red abierta, en la que entidades inteligentes auto-organizadas (servicio Web, componentes SOA) u objetos virtuales (avatares) son interoperables y capaces de actuar de forma independiente (que persiguen objetivos propios o compartidos), en función del contexto, las circunstancias o el ambiente. Generando una Inteligencia Ambiental.

La versión industrial del IoT se conoce como IIoT, Industrial Internet of Things, de sus siglas en inglés. Incluye determinismo, fiabilidad y sincronismo.

b) Arquitectura

El sistema es un ejemplo de "arquitectura orientada a eventos, construida de abajo hacia arriba (basada en el contexto de procesos y operaciones, en tiempo real) y tiene en consideración cualquier nivel adicional. Por lo tanto, el modelo orientado a eventos y el enfoque funcional coexisten con nuevos modelos capaces de tratar excepciones y la evolución insólita de procesos.

El significado de un evento no está necesariamente basado en modelos determinísticos o sintácticos. Este se basa en el contexto del propio evento: así, también una Web Semántica. En consecuencia, no son estrictamente necesarias normas comunes que no serían capaces de manejar todos los contextos o usos: algunos actores (servicios, componentes, avatares) están auto referenciados de forma coordinada y, si fuera necesario, se adaptan a normas comunes (para predecir algo solo sería necesario definir una "finalidad global", algo que no es posible con ninguno de los actuales enfoques y normas).

c) Seguridad

La empresa Hewlett Packard realizó un estudio en 2015, reportando que entre otros hallazgos respecto a los dispositivos IoT, 70% de ellos tiene vulnerabilidades de seguridad en sus contraseñas, hay problema con el cifrado de los datos o los permisos de acceso, y el 50% de las aplicaciones de dispositivos móviles no encriptan las comunicaciones. La firma de seguridad Kaspersky Lab realizó pruebas en objetos conectados al IoT y encontró que una cámara monitor para bebé podía hackearse para interceptar el video y en una cafetera que transmitía información sin encriptar, se podía conocer la contraseña de la red WiFi en donde estuviera conectada.

Dado esto, la seguridad se ha vuelto una situación de mucha importancia, porque pareciera que estos dispositivos no tienen riesgo para los datos que transmiten y almacenan, pareciera que no son vulnerables, pero la realidad es que estos dispositivos junto con otros que no tienen una debida protección, hicieron posible los ataques del 21 de octubre de 2016, donde los atacantes usaron miles de estos dispositivos, los cuales habían sido previamente infectados con código malicioso para formar una botnet.

Los datos que guardan los dispositivos IoT son altamente codiciados debido a que son de uso cotidiano y almacenan información acerca de los hábitos de los usuarios, por lo que contar con esas bases de datos es valiosa para varias empresas, que pueden dirigir sus esfuerzos en productos y servicios enfocados en los hábitos y preferencias de las masas. Lo que podrá ayudar a aminorar el problema, será el cifrado y la encriptación de datos, para poder subir los datos a la nube.

2.1.2. Internet de las cosas vivas

Este es un nuevo concepto en el que se hackea a la naturaleza para hacerla más social. El internet que fue originalmente concebida como una solución extrema en caso de una guerra mundial que permitiría que las comunicaciones fluyeran aun cuando otros sistemas estuvieran caídos hoy en día se está transfiriendo internet a la naturaleza para que en un futuro no muy lejano diferentes especies se puedan comunicar entre sí, sin importar si se trata de delfines, zorros, ballenas, gatos o perros.

Internet funciona bajo el mismo principio: la interconexión de distintos dispositivos sin importar su color o tamaño, país de procedencia, sistema operativo, lenguaje o patrón de conducta. Sin importar sus diferencias, todos los dispositivos se conectan a la red de redes.

Todos tenemos las mismas necesidades, el mismo código de programación. Comemos, nos reproducimos y nos comunicamos. Encontrar un lenguaje común que todos podamos hablar no es una idea tan descabellada. Después de todo si las computadoras lo hicieron, ¿por qué los seres vivos no podríamos?

La biología, por ejemplo, podría ser la madre de la informática. En las entrañas de su naturaleza es posible encontrar el código más complejo, el lenguaje de programación más abundante. Los seres vivos tenemos ADN y la información genética de cada uno es una pieza de código que puede dejar sorprendido a más de un incrédulo. Parece que, después de todo, la naturaleza también cuenta con su propia red y con protocolos que le permiten comunicarse entre evoluciones y generaciones. La naturaleza cambia constantemente, se adapta, se hackea para probar nuevas especies y los seres humanos podemos aprender de ella.

La internet de las cosas vivas ya existe y nos rodea. Es una red de redes compuesta de diminutos organismos, llámense células o bacterias y viven en diferentes especies, de distintos tamaños.

2.2. ARDUINO

2.2.1 Definición

Arduino es una plataforma de electrónica abierta con software y hardware libre y flexible para poder realizar prototipos y desarrollos de forma sencilla y fácil de utilizar.

Se inició en el año 2005 como un proyecto educativo en el instituto IVREA, en Ivrea (Italia), donde uno de sus fundadores Massimo Banzi, daba clases. El nombre viene del Bar di Re Arduino donde MaSSimo Banzi pasaba algunas horas. En su creación también contribuyeron tanto estudiantes como Hernando Barragán (Colombia) y profesores como David Cuartielles (España).

El instituto se vio obligado a cerrar en 2005 y se decidió abrir todo el proyecto a la comunidad para que no se pierda el proyecto y que pudiera evolucionar y mejorar gracias a la colaboración de muchísima gente a nivel mundial.

Gracias al modelo abierto de Arduino han surgido una infinidad de versiones y clones del mismo, ofreciendo un gran abanico de opciones y prestaciones disponibles, incluso pudiendo desarrollar nuestro propio modelo de Arduino para adaptarlo a nuestras necesidades

Se puede decir que Arduino en realidad está formada por 3 cosas:

- Una placa hardware.
- Un software de desarrollo.
- Lenguaje de programación.

2.2.1.1. Una placa hardware

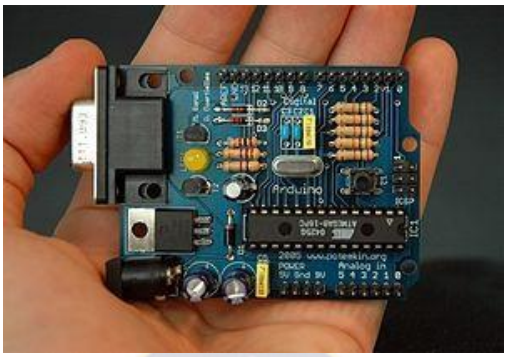

Figura 2.1: Placa Arduino RS232
Fuente: www.arduino.com

Es una placa de circuito impreso que incorpora un microcontrolador y los pines o puertos necesarios para su utilización, ya sean pines Entrada/Salida para su uso con los sensores como de comunicación RS232 o USB e incluso los de alimentación para su correcto funcionamiento.

El diseño hardware de la placa estaba inspirado en un principio en otra placa de Hardware libre existente, la placa Wiring (http://wiring.org.co)

Los microcontroladores que utilizan las placas arduino son modelos estándar de fabricante Atmel, los cuales vienen preprogramados con un bootloader o gestor de arranque para que podamos utilizarlos de forma sencilla con la plataforma Arduino. Cada modelo de Arduino dispone de unas características diferentes, ya que pueden tener distintos microcontroladores y distintas características según el uso para el que fueron diseñados.

2.2.1.2. Un software de desarrollo

Es un entorno de desarrollo gratuito, libre y multiplataforma, funciona en entornos Windows, Linux y MacOS, en él se especificará la placa que estamos utilizando, el puerto al

que la hemos conectado y se podrán añadir las librerías que deseemos utilizar en nuestro proyecto.

Con este software podemos escribir, compilar y programar la placa Arduino conectando la placa a nuestro ordenador mediante un cable usb que a su vez nos permite alimentar la placa si no queremos utilizar alimentación externa. También dispone de un monitor serie para poder comunicar nuestro ordenador y la placa con nuestro programa de una forma sencilla.

A pesar de funcionar correctamente, no es un editor de código muy completo no dispone de características consideradas básicas por muchos desarrolladores como el autocompletado entre otras y por ello permite la opción de poder utilizar nuestro editor preferido (CodeBlocks, Notepad++, Sublime, etc.) y utilizar el IDE solo para cargar el programa a la placa.

2.2.1.3. Un lenguaje de programación

Es un lenguaje de programación libre basado en otro lenguaje libre ya existente Processing (https://processing.org), el IDE de Arduino también está basado en él ya que era el lenguaje que utilizaban en el instituto Ivrea.

El lenguaje de Arduino tiene una gran similitud con C++, compartiendo las mismas estructuras de control, sintaxis, operadores, etc. Por lo que si se conoce dicho lenguaje es muy fácil adaptarse a él.

2.2.2. Versiones de Arduino

En este apartado se enumerarán algunos de los modelos oficiales de la placa Arduino, así como sus características, los modelos no oficiales y clones son muchísimo, muchos de ellos con características muy destacables y normalmente con un precio más reducido.

a) Arduino UNO Rev3

Placa basada en el microcontrolador ATmega328, se puede decir que es la placa base de la plataforma arduino. Dispone de 14 pines digitales y 6 analógicos

Figura 2.2: Arduino UNO Rev3
Fuente: www.arduino.com

b) Arduino YÚN

Esta placa dispone de conexión WIFI, se basa en el microcontrolador Atmega32u$ y en el chip Atheros AR9331, dispone de conector microsd y soporta una versión de la distribución Linux OpenWrt.

Figura 2.3: Arduino YÚN
Fuente: www.arduino.com

c) Arduino Nano 3.x

A pesar de tener un tamaño mucho menor dispone de las mismas características del Arduino UNO, pero utilizando un conector miniUSB y sin conector para alimentación, al utilizar pines machos se puede insertar perfectamente en una protoboard facilitando las pruebas. Dado su pequeño tamaño y compatibilidad con el Arduino Uno, será esta versión la que utilice para los Nodos del proyecto.

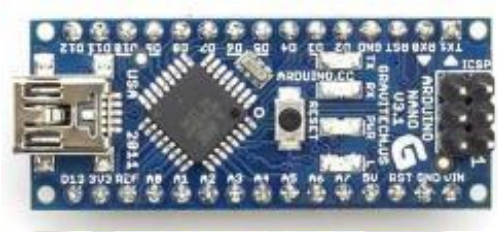

Figura 2.4: Arduino Nano 3.x
Fuente: www.arduino.com

d) Arduino DUE

El Arduino DUE es la evolución del Arduino Mega. Es la placa más potente de todas a la vez que nos proporciona una grandísima cantidad de pines, 54 de E/S, por si nuestro proyecto los necesitara y los anteriores modelos no fueran suficiente, el microcontrolador de esta placa dispone de registros de 32 bits, al contrario que el resto de microcontroladores utilizados en Arduino, los cuales tienen registros de 8 bits.

Figura 2.5: Arduino DUE
Fuente: www.arduino.com

Tabla 2.1: Revisión de características de los Arduinos más empleados

	UNO	YUN	NANO	DUE
Microcontrolador	ATmega328	ATmega32u4	ATmega328	AT91SAM3X8E
Voltaje	5V	5V	5 V	3.3V
Voltaje Vin	7-12V	5V	7-12 V	7-12V
Voltaje (limite)	6-20V	5V	6-20 V	6-20V
Pines E/S Digitales	14	20	14	54
Pines PWM	6	7	6	12
Pines E analógicos	6	12	8	12
Corriente pin E/S	40 mA	40 mA	40 mA	130 mA
Corriente pin 3.3V	50 mA	50 mA	40 mA	800 mA
Memoria Flash	32 KB	32 kB	32 KB	512 KB
Tamaño Bootloader	0.5 KB	4 kB	2 kB	-
SRAM	2 KB	2.5 kB	ATmega328	96 KB
EEPROM	1 KB	1 kB	1 KB	-
Velocidad reloj	16 MHz	16 MHz	16 MHz	84 MHz
Alto	68.6 mm	53 mm	45 mm	101.52 mm
Ancho	53.4 mm	73 mm	18 mm	53.3 mm
Peso	25 g	32 g	5 g	36 g

2.2.3. Hardware

El estudio del Hardware se realizará de la placa Arduino UNO dado que es la Básica y compatible en número de pines y microcontrolador con la placa nano, la placa UNO utiliza un formato DIP (Dual In-line Package) mientras la nano utiliza SMD (Surface Mount Device)

Esta placa hace uso del microcontrolador ATmega328P, la “P” significa que dispone de la tecnología picopower que permite reducir el consumo, tiene una arquitectura AVR desarrollada por Atmel y competencia de otras arquitecturas como puede ser PIC de Microchip

El Pinout completo tanto de la placa UnoR3 como del ATmega328:

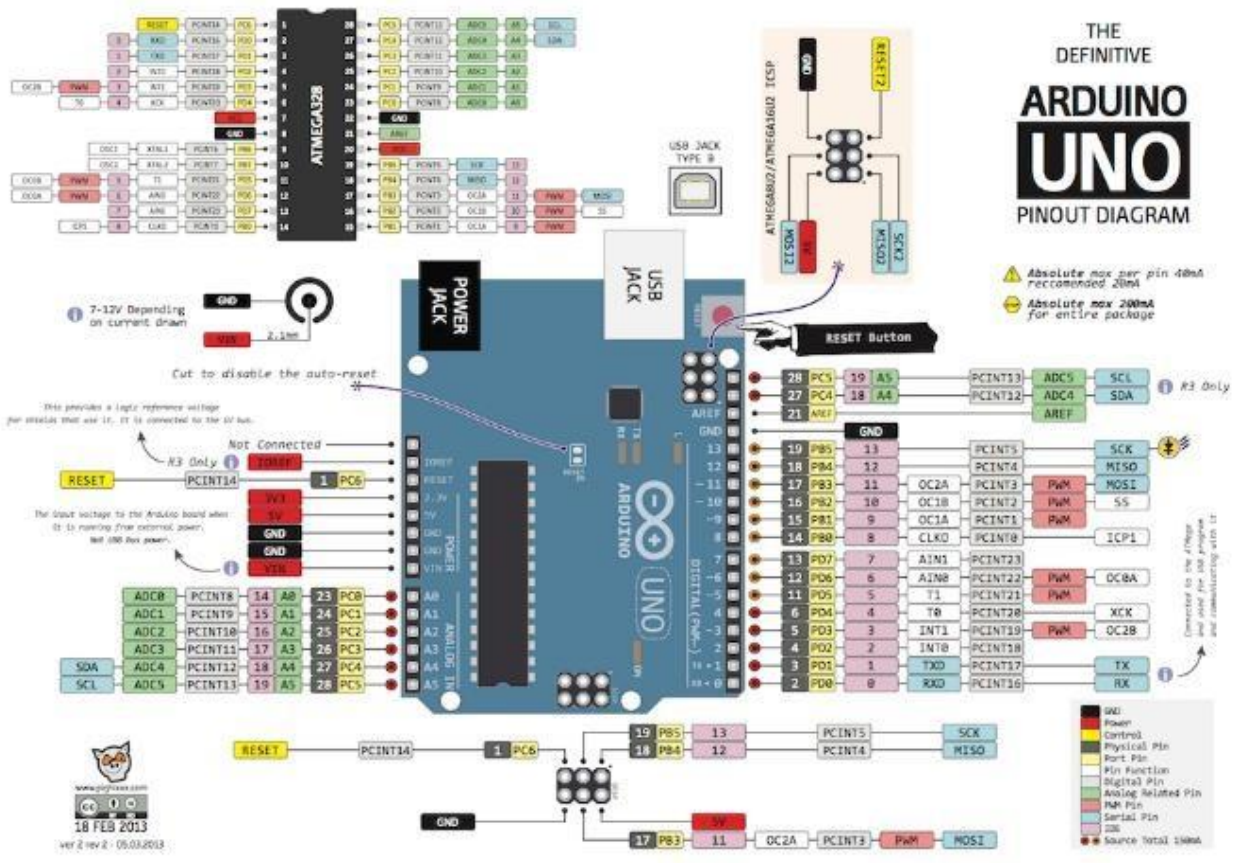

Figura 2.6: Diagrama de pines Arduino UNO

Fuente: www.arduino.com

a) Memorias

Memoria Flash: Es una memoria persistente en la cual se carga de forma permanente el programa que se va a ejecutar, hasta que sea sustituido por otro. Dispondremos de 32KB, pero hay que tener en cuenta que no se pueden utilizar por completo ya que 512 bytes son utilizados por el "bootloader block" para cargar el "bootloader".

Memoria SRAM: Esta es una memoria volátil donde se almacenarán los datos y variables que el programa utiliza en tiempo de ejecución. Su valor será eliminado cuando se pierda la alimentación o se reinicie la placa. Disponemos de 2KB.

Memoria EEPROM: Es una memoria persistente donde podemos almacenar los datos que deseamos conservar, aunque se pierda la alimentación de la placa y así poder utilizar en siguientes ejecuciones. Se dispone de 1KB.

Se puede ampliar la cantidad de memoria tanto SRAM como EEPROM conectando módulos de las mismas y comunicándolas con el microcontrolador mediante algún protocolo de comunicación, como pueda ser I^2C por ejemplo. En el caso de la memoria persistente, también se pueden incluir otro tipo de memoria como pueda ser una tarjeta SD (Secure Digital) y mediante el correspondiente módulo comunicarla con el microcontrolador.

b) Protocolos de comunicación

En la placa Arduino UNO disponemos de dos protocolos de comunicación:

I2C (Inter-Integrated Circuit): Es un estándar de comunicación, se caracteriza por utilizar solo dos líneas para realizar la comunicación, una línea SDA para transferir los datos y otra línea SCL mediante la cual se envía una señal de reloj para coordinar la comunicación, también hacen falta don líneas más la de alimentación y el común, pero se presuponen en el circuito. El pin de la señal SDA es el 27 y el de la señal SCL el 28.

SPI (Serial Peripheral Interface): También es un estándar, un maestro se puede comunicar con varios esclavos, éste protocolo necesita 4 líneas para realizar la comunicación. SCK es la señal de reloj. SS una línea diferente para cada esclavo con la que se decide que esclavo activar MOSI y MISO son líneas de datos que son compartidas por todos los esclavos al igual que SCK, solo hará uso de ellas el seleccionado por la línea SS. Los pines correspondientes son SS el 16, MOSI el 17, MISO el 18 y el SDK es el 19, para controlar más esclavos solo hay que utilizar cualquier pin digital como SS, uno para cada esclavo.

c) Alimentación

El voltaje de funcionamiento de la placa es de 5v DC, la placa se puede alimentar de varias maneras:

- Conectando la placa mediante el conector de tipo Jack de 2,1mm a una fuente externa de entre 7v y 12v, aunque en teoría estaría preparada para soportar tensiones de entre 6v y 20v. La polaridad tiene que ser positivo en el centro. El regulador que dispone la placa transformará el voltaje a los 5v necesarios para su funcionamiento.

- También se puede conectar la placa al USB del ordenador, la placa se alimentará de las 5v que proporciona el USB, nos dará 500mA, hay que tener en cuenta que el USB puede no ser capaz de proporcionar la corriente necesaria para el circuito, por ejemplo, hay que alimentar el módulo esp8266 de forma externa ya que no es capaz de proporcionar la corriente necesaria para su uso.

- Pines de alimentación de la placa:

  - **GND**: Pin común conectado a tierra.

  - **Vin**: Hace la misma función que el pin central del conector Jack pudiendo alimentar la placa de la misma manera que con dicho conector.

  - **5v**: Se puede utilizar para alimentar la placa mediante una fuente regulada de 5v o también si se alimenta la placa por otro de los medios disponibles ofrecer 5v regulados con 40mA para conectar cualquier elemento.

  - **3.3v**: Este pin ofrece un voltaje regulado de 3.3v y 50mA sea cual sea el método de alimentación de la placa

d) Comunicación USB-Serie

Para la comunicación con el ordenador el Arduino UNO R3 hace uso de otro microcontrolador, el ATmega16U2 para que realice la conversión de USB a TTL-UART serie que incorpora el ATmega328P, de esta forma se nos crea una especie de puerto serie virtual para la comunicación USB-Arduino.

Los pines digitales 0 (RX) y 1 (TX) se utilizan también para realizar la comunicación serie por lo que no podrán ser utilizados como E/S digitales si se está realizando comunicación serie.

En modelos anteriores de Arduino se utilizaba el integrado FT232RL para realizar dicha conversión. En algunos clones del modelo nano, en especial modelos chinos, se incorpora el integrado CH340G para realizar dicha función, este modelo no me ha dado ningún problema utilizando el IDE desde Linux, pero en Windows no reconoce el dispositivo por lo que hay que instalar los drivers necesarios para su correcto funcionamiento:

http://wch.cn/download/CH340PCB_ZIP.html

e) Entradas y salidas digitales

Se dispone de 14 entradas y salidas digitales (pines del 0 al 13), funcionan a 5v y pueden trabajar con un máximo de 40mA, también disponen estos pines de una resistencia tipo “pull- up” interna pero que está desconectada, hay que activarla en el código del programa.

f) Entradas analógicas

Disponemos de 6 entradas analógicas (pines del A0 al A5) que son capaces de recibir voltajes entre 0v y 5v, estos pines van a un conversor analógico digital en la placa de 6 canales y 10 bits de resolución por canal por lo que dispondremos de una precisión de 2^{10} lo que es lo mismo 1024 que corresponderán 0 a 0v y 1023 a 5v.

g) Salidas analógicas PWM

La placa Arduino no dispone de salidas analógicas propiamente dichas, pero lo simula utilizando unas cuantas salidas digitales, estas salidas están etiquetadas con PWM (Pulse Width Modulation) y son las que corresponden a los pines 3, 5, 6, 9, 10 y 11.

Esto se consigue emitiendo una señal de pulsos cuadrada con una frecuencia constante, sobre 490Hz, al variar la duración de los pulsos se varía también la tensión promedio resultante, a pulsos más cortos, menor tensión promedio y viceversa.

Los pines PWM tienen una resolución de 8 bits por lo que con valor 0 tendríamos 0v y con valor 255 los 5v deseados.

También hay que tener los pines PWM están controlados por 3 temporizadores diferentes:

- Pines 3 y 11 controlados por Timer1
- Pines 5 y 6 controlados por Timer2
- Pines 9 y 10 controlados por Timer3

Hay que tener en cuenta los Timers ya que en ocasiones distintas librerías hacen uso del mismo Timer produciendo efectos no deseados en el funcionamiento del circuito, como me ocurrió al utilizar el servo y la librería SoftwareSerial.h, se solucionó utilizando la librería PWMServo.h (http://arduiniana.org/libraries/pwmservo/) dado que utiliza otro Timer distinto.

h) Interrupciones

Se puede hacer uso de interrupciones hardware con Arduino UNO, para ello se utilizan los pines 2 y 3 pero también se pueden utilizar el resto de pines digitales usando interrupciones por software mediante el uso de librerías como, por ejemplo:

https://github.com/GreyGnome/EnableInterrupt

i) Pines varios

- AEREF: Voltaje de referencia externo para aumentar la precisión de las entradas analógicas.

- IOREF: indica el voltaje al que trabajan los pines de entrada/salida

- RESET: Si se pone este pin a BAJO, el microcontrolador se reiniciará de la misma forma que si se pulsara el botón de RESET.

2.2.4. Lenguaje de programación

El lenguaje de programación de Arduino, como se ha indicado anteriormente es una versión simplificada de C/C++, los programas de Arduino de denominan Sketch tiene extensión ".ino" y una vez compilados se convierten a código binario AVR.

El programa se verifica y se carga en el Arduino utilizando los botones del IDE:

2.2.4.1. Características:

- Es un lenguaje case-sensitive (diferencia entre mayúsculas y minúsculas)
- Todas las instrucciones terminan con un punto y coma.
- Un programa o sketch de Arduino se compone de tres secciones:

- **Sección de declaraciones**
  En esta sección se declararán las variables globales del programa y también se incluirán las librerías y ficheros que utilicemos en el mismo.

- **Sección “void setup ( ) {…}”**

  Esta sección solo se ejecutará una vez cuando se alimente la placa o cuando se reinicie la misma. Se ejecutarán una sola vez todas las instrucciones dentro de la función setup () estas instrucciones incluirán las inicializaciones de las variables y librerías necesarias, así como también se establecerán el estado de los Pines de la placa.

- **Sección “void loop ( ) {…}”**

  Justo tras terminar la sección setup () se ejecutarán las instrucciones contenidas dentro de la función loop () de forma continua e infinitas veces hasta que la placa sea desconectada o reseteada.

El lenguaje de programación de la plataforma Arduino es sencillo y prácticamente igual a C/C++ por lo que cualquiera que tenga conocimientos de programación no tendrá excesivas dificultades en empezar a desarrollar programas para esta plataforma, se ha incluido un anexo con la estructura de dicho lenguaje para su mayor comprensión.

2.3. SENSORES Y ACTUADORES

En esta sección se estudiarán y se implementarán los programas y circuitos necesarios para el correcto uso de una serie de sensores y actuadores, de forma que más tarde se puedan integrar en los nodos, se intentará que los circuitos y programas comiencen siendo sencillos y se incremente su complejidad, conforme aumente el conocimiento y la práctica con Arduino y los sensores mismos. Tanto el código de los programas como los circuitos en formato ‘. fzz’ se incluirán en los archivos adjuntos. La plataforma Arduino y su comunidad nos proporciona una gran cantidad de ejemplos y documentación para poder controlar dichos sensores, el código correspondiente a los siguientes ejemplos se puede encontrar en el Anexo.

Para representar los circuitos de forma sencilla y reproducible se utilizará el programa ‘Fritzing’ que nos permite diseñar los circuitos de una forma visual e intuitiva. Fritzing es una iniciativa de software libre creada bajo los principios de Processing y Arduino, también es una herramienta multiplataforma y dispone de una gran comunidad de usuarios que ayuda a su evolución y mejora continua.

2.3.1. Leds

Este puede ser el circuito y programa más simple, consiste en conectar un led a una salida digital y cambiar el estado de dicha salida para que se encienda o se apague dicho led. Hay que recordar incluir una resistencia o de otra forma podríamos dañar la placa Arduino.

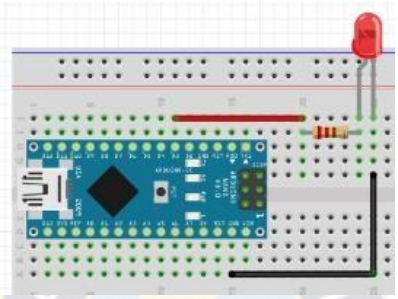

Figura 2.7: Circuito led

2.3.2. Buzzer

Este es un circuito similar al anterior, pero en este caso vamos a hacer uso de las salidas analógicas, es decir de las salidas PWM, para hacer vibrar a cierta frecuencia el buzzer piezoeléctrico y así conseguir un sonido audible, de esta sencilla forma podemos disponer de una alarma para nuestros nodos.

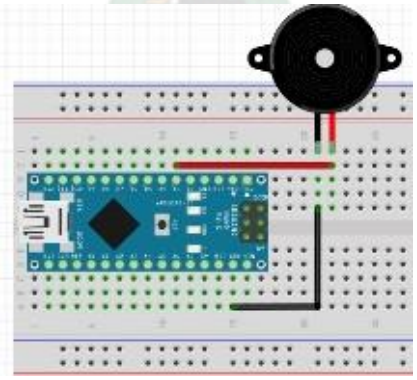

Figura 2.8. Circuito buzzer

2.3.3. Sensor de movimiento

Para este circuito se utilizará un sensor de movimiento PIR (Passive Infrared), este sensor dispone de componente que detecta los cambios de radiación infrarroja recibida como pueda ser el calor que generamos los seres humanos o los animales, el circuito de este módulo activa uno de los pines a 5v en el momento que detecte un cambio.

Este módulo hay que alimentarlo a 5v por lo que podemos utilizar los pines del mismo arduino para realizar esto, también dispone de 2 potenciómetros para calibrar su sensibilidad.

En el programa mostraremos por el puerto serie un mensaje alertando del movimiento detectado, así hacemos uso de él, es una buena forma para hacer debug a nuestro programa, para poder verlo hay que seleccionar en el IDE de Arduino 'Herramientas > Monitor Serie', hay que comprobar que la velocidad del puerto corresponda a la que hemos definido en el programa o lo se recibirán los datos correctos.

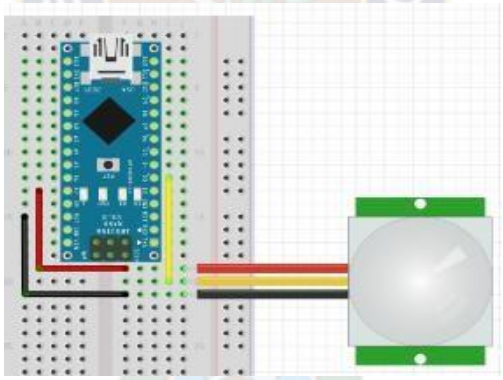

Figura 2.9. Circuito sensor de movimiento

2.3.4. Servo

El siguiente elemento a probar será un servo, el servo necesita de las señales PWM para poder cambiar el ángulo del brazo, entre 0º y 180º, este elemento que nos brinda una gran cantidad de posibilidades para actuar en el mundo real, podríamos por ejemplo mover una cerradura o la orientación de una cámara.

El circuito es muy sencillo como los anteriores, se alimenta el servo a 5v y mediante la señal PWM del pin 3 se controla, esta vez en lugar de escribir el Arduino en el puerto serie, seremos nosotros los que escribiremos en dicho puerto el ángulo que deseamos que tenga el brazo del servo, se hará uso de la librería servo.h.

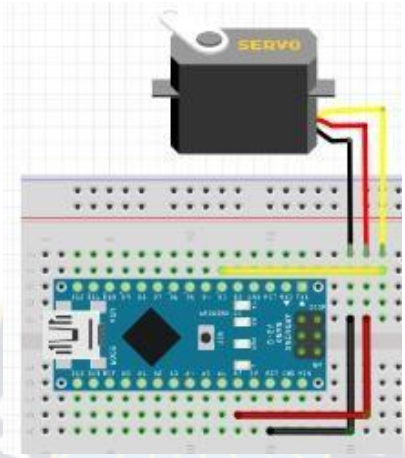

Figura 2.10. Circuito servo

2.3.5. Relé

Otro componente de gran utilidad para actuar con el mundo real es el relé el cual nos permite tener control de otros circuitos de Voltaje superior al que admite Arduino como puede ser el control de las luces de nuestra casa, podríamos utilizarlo de interruptor automático si lo unimos al módulo de sensor de movimiento, por ejemplo.

El relé se alimenta también a 5v, y se controla con una sola señal en este caso conectada al pin 3, dispone de un conector común y otros dos conectores que son normalmente abierto y normalmente cerrado respectivamente, cuando cambiemos el valor del pin 3 se cambiarán el valor de estas 2 salidas

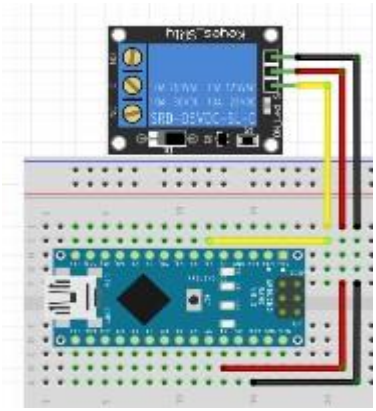

Figura 2.11. Circuito relé

2.3.6. Temperatura y humedad

Existen sensores específicos para conocer la temperatura y otros para conocer la humedad respectivamente, en este caso se utilizará un sensor que nos proporcionará estos dos valores él solo, este sensor es el DHT11, es un sensor muy barato con el que podemos medir la temperatura entre 0º y 50º C con una precisión de ±2°C y la humedad entre 20% y 80% con una precisión de 5%.

También se alimenta a 5v este sensor, el pin de señal será el 3 que se conecta al 2 del sensor y hace falta una resistencia pull up de 10k entre el pin 3 y el pin 2 del sensor

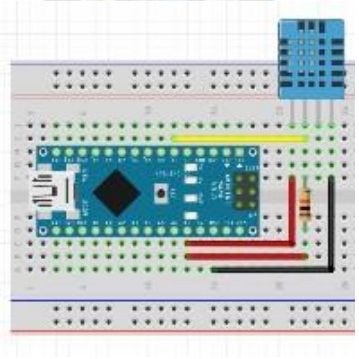

Figura 2.12. Circuito temperatura y humedad

2.3.7. Sensor de gas y humo (MQ2)

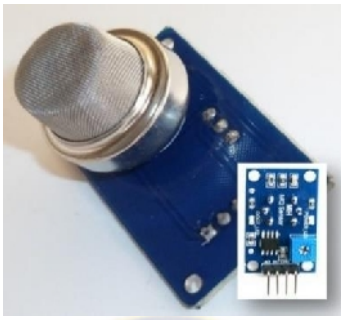

Figura 2.13. Sensor de gas y de huno arduino
Fuente: www.naylampmechatronics.com

El sensor de gas analógico (MQ2) se utiliza en la detección de fugas de gas de equipos en los mercados de consumo y la industria, este sensor es adecuado para la detección de gas LP, i-butano, propano, metano, alcohol, hidrógeno, tiene una alta sensibilidad, un tiempo de respuesta rápido y dicha sensibilidad puede ser ajustada por el potenciómetro.

Este pequeño sensor de gas detecta la presencia de gas combustible y humo en concentraciones de 300 a 10.000 ppm. Incorpora una sencilla interfaz de tensión analógica que únicamente requiere un pin de entrada analógica del microcontrolador. Con la conexión de cinco voltios en los pines el sensor se mantiene lo suficientemente caliente para que funcione correctamente. Solo tiene que conectar 5V a cualquiera de los pines (A o B) para que el sensor emita tensión. La sensibilidad del detector se ajusta con una carga resistiva entre los pines de salida y tierra.

Estructura y configuración de MQ-2 sensor de gas, el sensor compuesto por micro tubo de cerámica Al2O3, capa sensible de Dióxido de Estaño (SnO2), el electrodo de medida y el calentador se fija en una corteza hecha por el plástico y red de acero inoxidable. El calentador proporciona las condiciones de trabajo necesarias para el trabajo de componentes sensibles. La envoltura MQ-2 tienen 6 pines, 4 de ellos se utilizan para recoger las señales, y otros se utilizan 2 para proporcionar corriente de calentamiento.

- Condiciones de trabajo
- Voltaje de circuito: 5V
- Voltaje de calentamiento: 5v
- Resistencia de carga: puede ser ajustable
- Resistencia del calentador: 33Ω ±5%
- Consumo: menos de 800mW
- Ro: resistencia del sensor a 1000 ppm de H_2 en el aire a 33% HR y 20 grados.
- Rs: resistencia del sensor a 1000 ppm de H_2 a diferentes temperaturas y humedades

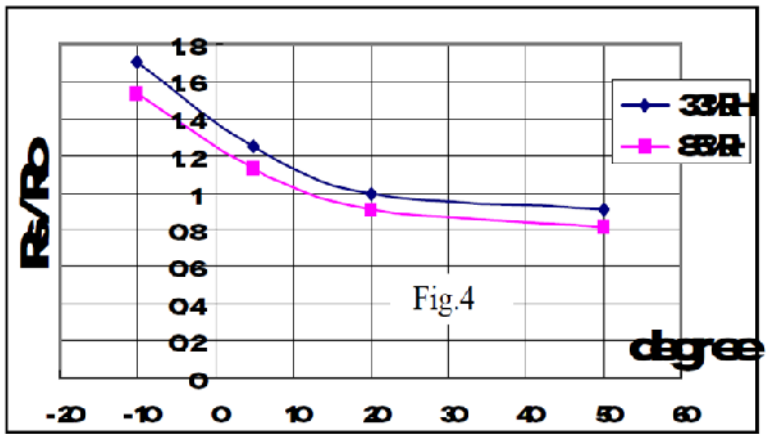

Figura 2.14. Se muestra la dependencia típica del MQ-2 de la temperatura y la humedad

Fuente: www.naylampmechatronics.com

2.3.8. Sensor de sonido

Mediante este sensor, si en un momento determinado se escucha un sonido, este montaje lo detecta. Se puede seleccionar la sensibilidad del sensor, para en función del volumen del sonido sea recogido o no. El sensor empleado en este tutorial tiene un LED verde incorporado que te indica cuando percibe un sonido y cuando no. Si está encendido significa que está recibiendo sonido, si está apagado lo contrario.

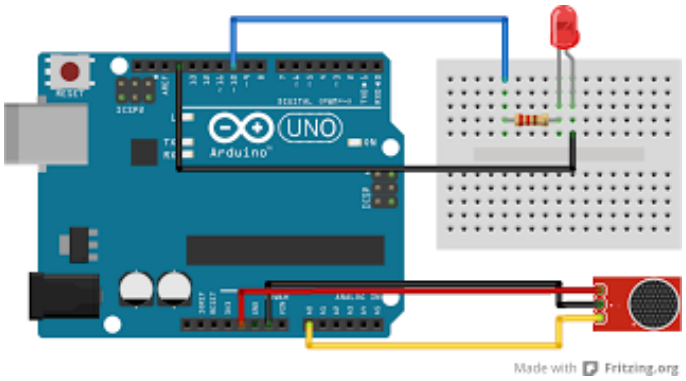

Figura 2.15. Circuito básico para el sensor de sonido

Seguidamente se debe de conectar el sensor de sonido a la placa Arduino, vemos que hay 3 pines macho en el sensor: Vcc, GND y OUT. Vcc es el pin que se debe de conectar a la fuente de tensión (5V), GND es la toma a tierra y por último OUT es el pin de salida al que queremos conectar el sensor (en este caso se ha conectado a A0). Cabe destacar que este sensor recoge datos analógicos, por lo que debe de conectarse a los pines hembra de la parte analógica, no en la digital (cosa fácilmente confundible, pues se puede pensar que simplemente recoge sonido (1) o no recoge sonido (0), pero no es así).

Por último, se debe de calibrar la sensibilidad del sensor, para ello en la parte delantera se ve un regulador que mediante un tornillo de punta de estrella puede calibrarse al gusto.

2.4. RFID

Es un sistema de almacenamiento y recuperación de datos remoto que usa dispositivos denominados etiquetas, tarjetas, transpondedores o tags RFID. El propósito fundamental de la tecnología RFID es transmitir la identidad de un objeto (similar a un número de serie único) mediante ondas de radio. Las tecnologías RFID se agrupan dentro de las denominadas Auto ID (automatic identification, o identificación automática).

Las etiquetas RFID (RFID Tag, en inglés) son unos dispositivos pequeños, similares a una pegatina, que pueden ser adheridas o incorporadas a un producto, un animal o una persona. Contienen antenas para permitirles recibir y responder a peticiones por radiofrecuencia desde

un emisor-receptor RFID. Las etiquetas pasivas no necesitan alimentación eléctrica interna, mientras que las activas sí lo requieren. Una de las ventajas del uso de radiofrecuencia (en lugar, por ejemplo, de infrarrojos) es que no se requiere visión directa entre emisor y receptor.

2.4.1. Arquitectura

El modo de funcionamiento de los sistemas RFID es simple. La etiqueta RFID, que contiene los datos de identificación del objeto al que se encuentra adherida, genera una señal de radiofrecuencia con dichos datos. Esta señal puede ser captada por un lector RFID, el cual se encarga de leer la información y pasarla en formato digital a la aplicación específica que utiliza RFID. Un sistema RFID consta de los siguientes tres componentes:

Etiqueta RFID o transpondedor: compuesta por una antena, un transductor radio y un material encapsulado o chip. El propósito de la antena es permitirle al chip, el cual contiene la información, transmitir la información de identificación de la etiqueta. Existen varios tipos de etiquetas. El chip posee una memoria interna con una capacidad que depende del modelo y varía de una decena a millares de bytes. Existen varios tipos de memoria:

- **Solo lectura**: el código de identificación que contiene es único y es personalizado durante la fabricación de la etiqueta.

- **De lectura y escritura**: la información de identificación puede ser modificada por el lector.

- **Anticolisión**. Se trata de etiquetas especiales que permiten que un lector identifique varias al mismo tiempo (habitualmente las etiquetas deben entrar una a una en la zona de cobertura del lector).

- **Lector de RFID** o **transceptor**: compuesto por una antena, un transceptor y un decodificador. El lector envía periódicamente señales para ver si hay alguna etiqueta en sus inmediaciones. Cuando capta una señal de una etiqueta (la cual contiene la

información de identificación de esta), extrae la información y se la pasa al subsistema de procesamiento de datos.

- **Subsistema de procesamiento de datos o Middleware RFID**: proporciona los medios de proceso y almacenamiento de datos.

2.4.2. Tipos de etiquetas RFID

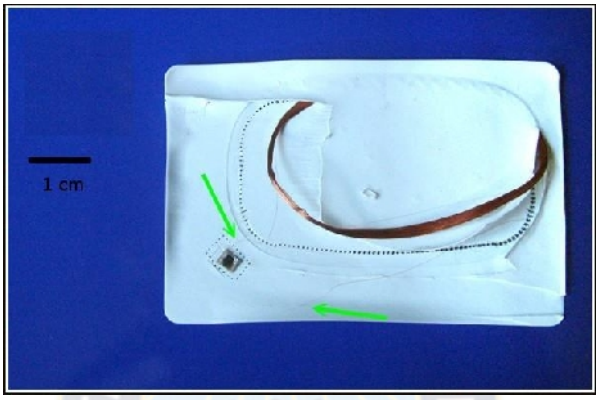

Figura 2.16. Tamaño de un chip RFID con antena
Fuente: https://es.wikipedia.org/wiki/RFID

Las etiquetas RFID pueden ser activas, semipasivas (también conocidos como semiactivos o asistidos por batería) o pasivos. Las etiquetas pasivas no requieren ninguna fuente de alimentación interna y son dispositivos puramente pasivos (sólo se activan cuando un lector se encuentra cerca para suministrarles la energía necesaria). Los otros dos tipos necesitan alimentación, típicamente una pila pequeña.

La gran mayoría de las etiquetas RFID son pasivas, que son mucho más baratas de fabricar y no necesitan batería. En 2004, estas etiquetas tenían un precio desde 0.40$, en grandes pedidos, para etiquetas inteligentes, según el formato, y de 0.95$ para etiquetas rígidas usados frecuentemente en el sector textil encapsulados en PPs o epoxi. El mercado de RFID universal de productos individuales será comercialmente viable con volúmenes muy grandes de 10 000 millones de unidades al año, llevando el coste de producción a menos de 0.05$ según un fabricante. La demanda actual de chips de circuitos integrados con RFID no está cerca de soportar ese coste. Los analistas de las compañías independientes de investigación como Gartner

and Forrester Research convienen en que un nivel de precio de menos de 0.10$ (con un volumen de producción de 1 000 millones de unidades) sólo se puede lograr en unos 6 u 8 años, lo que limita los planes a corto plazo para una adopción extensa de las etiquetas RFID pasivas. Otros analistas creen que esos precios serían alcanzables dentro de 10-15 años.

A pesar de que las ventajas en cuanto al coste de las etiquetas RFID pasivas con respecto a las activas son significativas, otros factores, incluyendo exactitud, funcionamiento en ciertos ambientes como cerca del agua o metal, y fiabilidad, hacen que el uso de etiquetas activas sea muy común hoy en día.

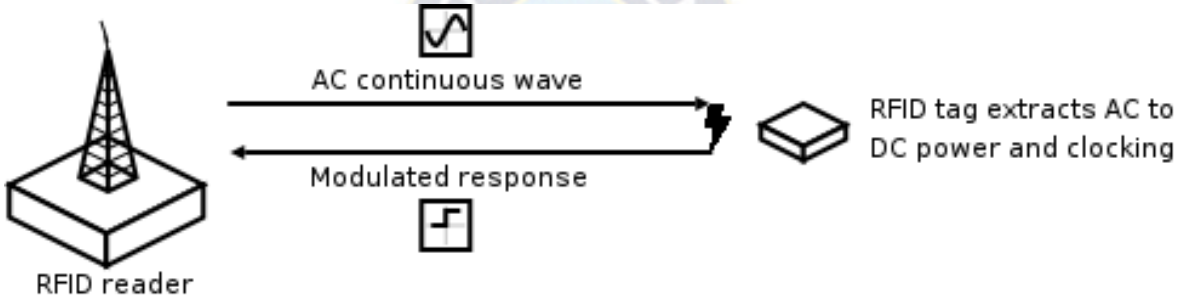

Figura 2.17. Backscatter en RFID
Fuente: https://es.wikipedia.org/wiki/RFID

Para comunicarse, las etiquetas responden a peticiones o preguntas generando señales que a su vez no deben interferir con las transmisiones del lector, ya que las señales que llegan de las etiquetas pueden ser muy débiles y han de poder distinguirse. Además de la reflexión o backscatter, puede manipularse el campo magnético del lector por medio de técnicas de modulación de carga. El backscatter se usa típicamente en el campo lejano y la modulación de carga en el campo próximo (a distancias de unas pocas veces la longitud de onda del lector).

a) Etiquetas pasivas

Las etiquetas pasivas no poseen alimentación eléctrica. La señal que les llega de los lectores induce una corriente eléctrica pequeña y suficiente para operar el circuito integrado CMOS de la etiqueta, de forma que puede generar y transmitir una respuesta. La mayoría de las

etiquetas pasivas utiliza backscatter sobre la portadora recibida; esto es, la antena ha de estar diseñada para obtener la energía necesaria para funcionar a la vez que para transmitir la respuesta por backscatter. Esta respuesta puede ser cualquier tipo de información, no sólo un código identificador. Una etiqueta puede incluir memoria no volátil, posiblemente escribible (por ejemplo (EEPROM).

Las etiquetas pasivas suelen tener distancias de uso práctico comprendidas entre los 10 cm (ISO 14443) y llegando hasta unos pocos metros (EPC e ISO 18000-6), según la frecuencia de funcionamiento y el diseño y tamaño de la antena. Por su sencillez conceptual, son obtenibles por medio de un proceso de impresión de las antenas. Como no precisan de alimentación energética, el dispositivo puede resultar muy pequeño: pueden incluirse en una pegatina o insertarse bajo la piel (etiquetas de baja frecuencia).

En 2006, Hitachi desarrolló un dispositivo pasivo denominado μ-Chip con un tamaño de 0.15×0.15 mm sin antena, más delgado que una hoja de papel (7,5 μm). Se utiliza SOI (Silicon-on-Insulator) para lograr esta integración. Este chip puede transmitir un identificador único de 128 bits fijado a él en su fabricación, que no puede modificarse y confiere autenticidad al mismo. Tiene un rango máximo de lectura de 30 cm. En febrero de 2007 Hitachi presentó un dispositivo aún menor de 0.05×0.05 mm y lo suficientemente delgado como para poder estar integrado en una hoja de papel.[6] Estos chips tienen capacidad de almacenamiento y pueden funcionar en distancias de hasta unos pocos cientos de metros. Su principal inconveniente es que su antena debe ser como mínimo 80 veces más grande que el chip.

Alien Technology (Fluidic Self Assembly), SmartCode (Flexible Area Synchronized Transfer) y Symbol Technologies (PICA) declaran disponer de procesos en diversas etapas de desarrollo que pueden reducir aún más los costes por medio de procesos de fabricación paralela. Estos medios de producción podrían reducir mucho más los costes y dirigir los modelos de economía de escala de un sector importante de la manufactura del silicio. Esto podría llevar a una expansión mayor de la tecnología de etiquetas pasivas.

Existen etiquetas fabricadas con semiconductores basados en polímeros desarrollados por compañías de todo el mundo. En 2005 PolyIC y Philips presentaron etiquetas sencillas en el rango de 13,56 MHz que utilizaban esta tecnología. Si se introducen en el mercado con éxito, estas etiquetas serían producibles en imprenta como una revista, con costes de producción mucho menores que las de silicio, sirviendo como alternativa totalmente impresa, como los actuales códigos de barras. Sin embargo, para ello es necesario que superen aspectos técnicos y económicos, teniendo en cuenta que el silicio es una tecnología que lleva décadas disfrutando de inversiones de desarrollo multimillonarias que han resultado en un coste menor que el de la impresión convencional.

Debido a las preocupaciones por la energía y el coste, la respuesta de una etiqueta pasiva RFID es necesariamente breve, normalmente apenas un número de identificación (GUID). La falta de una fuente de alimentación propia hace que el dispositivo pueda ser bastante pequeño: existen productos disponibles de forma comercial que pueden ser insertados bajo la piel. En la práctica, las etiquetas pasivas tienen distancias de lectura que varían entre unos 10 milímetros hasta cerca de 6 metros, dependiendo del tamaño de la antena de la etiqueta y de la potencia y frecuencia en la que opera el lector. En 2007, el dispositivo disponible comercialmente más pequeño de este tipo medía 0.05 milímetros × 0.05 milímetros, y más fino que una hoja de papel; estos dispositivos son prácticamente invisibles.

b) Etiquetas activas

A diferencia de las etiquetas pasivas, las activas poseen su propia fuente autónoma de energía, que utilizan para dar corriente a sus circuitos integrados y propagar su señal al lector. Estas son mucho más fiables (tienen menos errores) que las pasivas debido a su capacidad de establecer sesiones con el lector. Gracias a su fuente de energía son capaces de transmitir señales más potentes que las de las pasivas, lo que les lleva a ser más eficientes en entornos dificultosos para la radiofrecuencia como el agua (incluyendo humanos y ganado, formados en su mayoría por agua), metal (contenedores, vehículos). También son efectivas a distancias mayores pudiendo generar respuestas claras a partir de recepciones débiles (al contrario que las pasivas). Por el contrario, suelen ser mayores y más caras, y su vida útil es en general mucho más corta.

Muchas etiquetas activas tienen rangos efectivos de cientos de metros y una vida útil de sus baterías de hasta 10 años. Algunas de ellas integran sensores de registro de temperatura y otras variables que pueden usarse para monitorizar entornos de alimentación o productos farmacéuticos. Otros sensores asociados con RFID incluyen humedad, vibración, luz, radiación, temperatura y componentes atmosféricos como el etileno. Además de mucho más rango (500 m), tienen capacidades de almacenamiento mayores y la habilidad de guardar información adicional enviada por el transceptor.

Actualmente, las etiquetas activas más pequeñas tienen un tamaño aproximado de una moneda. Muchas etiquetas activas tienen rangos prácticos de diez metros, y una duración de batería de hasta varios años.

Características

- Fuente de alimentación propia mediante batería de larga duración (generalmente baterías de litio / dióxido de manganeso).
- Distancias de lectura escritura mayor de 10m a 100m generalmente.
- Diversas tecnologías y frecuencias.
    - Hasta 868 MHz (UHF) o según estándares aplicados.
    - 2.4 GHz muy utilizada (banda ISM, Industrial Scientific and Medical), la misma que para dispositivos wireless LAN 802.11b.
- Memoria generalmente entre 4 y 32 kB.
- Principales fabricantes: TagMaster, Identec Solutions, Siemens, Nedap, WhereNet, Bluesoft, Syris RFID.
- Precio de la etiqueta: 30 a 90 €.

La principal ventaja de las etiquetas RFID activas respecto a las pasivas es el elevado rango de lectura, del orden de decenas de metros. Como desventajas, cabe destacar el precio, que es muy superior que las pasivas y la dependencia de alimentación por baterías. El tiempo de vida de las baterías depende de cada modelo de etiqueta y también de la actividad de este,

normalmente es del orden de años. Para facilitar la gestión de las baterías, es habitual que las etiquetas RFID activas envíen al lector información del nivel de batería, lo que permite sustituir con antelación aquellas que están a punto de agotarse.

Baterías de larga duración utilizadas en etiquetas RFID activas

Estas baterías proporcionan a las etiquetas una alimentación en modo reposo en el cual la corriente consumida es muy pequeña (3uA generalmente) y en modo de funcionamiento (donde se consume 24mA) estas baterías pueden durar desde 1 a 10 años, lo que los hace más robustos. Los más utilizados son los de litio y dióxido de manganeso como el CR2032 y el CR2320. A continuación se tiene sus características técnicas:

- Sistema químico: Li /MnO2
- Voltaje nominal: 3 V
- Capacidad nominal: 235 mAh
- Descarga de corriente estándar: 0.4 mA
- Máxima corriente de descarga: 3.0 mA
- Peso promedio: 2.8 g
- Rango de temperatura: de -30 a 70 °C
- Descarga pasiva a 23 °C: < 1 %/al año

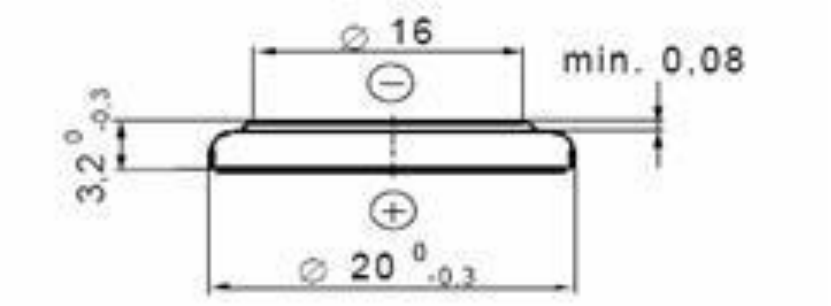

Figura 2.18. Dimensiones de acuerdo al IEC 60086

Fuente: https://es.wikipedia.org/wiki/RFID

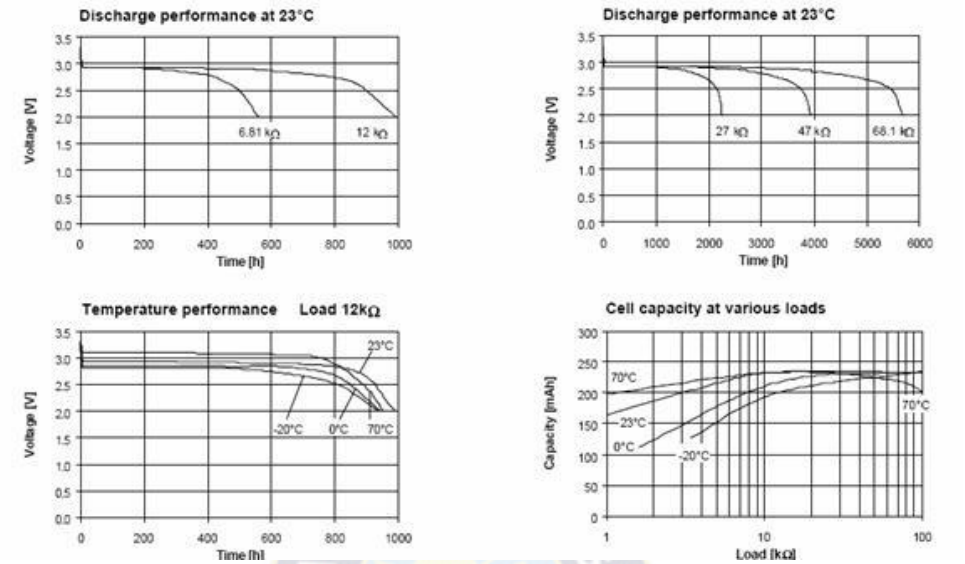

Figura 2.19. Descarga en función a la temperatura y a la resistencia
Fuente: https://es.wikipedia.org/wiki/RFID

También hay baterías impresas ultra-finas para el diseño de empaquetado activo. Estas baterías son flexibles, de gran alcance y tienen menos de un milímetro de grosor, lo que las hacen ideales para las etiquetas activas de los sistemas RFID.

Otra alternativa son las baterías de papel, que tienen aplicaciones en dispositivos RFID, smart cards y LED en papel, entre otros. Se trata de una batería que está formada por laminas finas de compuestos químicos incrustados en papel obteniéndose energía eléctrica a partir de reacciones de oxidación-reducción, produciendo en los bornes un voltaje nominal de 1.5 V y una carga de 1.5 mAh aproximadamente.

c) Etiquetas semipasivas

Las etiquetas semipasivas se parecen a las activas en que poseen una fuente de alimentación propia, aunque en este caso se utiliza principalmente para alimentar el microchip y no para transmitir una señal. La energía contenida en la radiofrecuencia se refleja hacia el lector como en una etiqueta pasiva. Un uso alternativo para la batería es almacenar información

propagada desde el lector para emitir una respuesta en el futuro, típicamente usando backscatter. Las etiquetas sin batería deben responder reflejando energía de la portadora del lector al vuelo.

La batería puede permitir al circuito integrado de la etiqueta estar constantemente alimentado y eliminar la necesidad de diseñar una antena para recoger potencia de una señal entrante. Por ello, las antenas pueden ser optimizadas para utilizar métodos de backscattering. Las etiquetas RFID semipasivas responden más rápidamente, por lo que son más fuertes en el ratio de lectura que las pasivas.

Este tipo de etiqueta tiene una fiabilidad comparable a la de las activas, a la vez que pueden mantener el rango operativo de una pasiva. También suelen durar más tiempo que las activas.

2.4.3. Tipos de antena

El tipo de antena utilizado en una etiqueta depende de la aplicación para la que está diseñado y de la frecuencia de operación. Las etiquetas de baja frecuencia (LF, del inglés low frequency) normalmente se sirven de la inducción electromagnética. Como el voltaje inducido es proporcional a la frecuencia, se puede producir el necesario para alimentar un circuito integrado utilizando un número suficiente de espiras. Existen etiquetas LF compactas (como las encapsuladas en vidrio, utilizadas para identificación humana y animal) que utilizan una antena en varios niveles (tres de 100-150 espiras cada uno) alrededor de un núcleo de ferrita.

En alta frecuencia (HF, 13.56 MHz) se utiliza una espiral plana con 5-7 vueltas y un factor de forma parecido al de una tarjeta de crédito para lograr distancias de decenas de centímetros. Estas antenas son más baratas que las LF ya que pueden producirse por medio de litografía en lugar de espiración, aunque son necesarias dos superficies de metal y una aislante para realizar la conexión cruzada del nivel exterior al interior de la espiral, donde se encuentran el condensador de resonancia y el circuito integrado.

Las etiquetas pasivas en frecuencias ultra alta (UHF) y de microondas suelen acoplarse por radio a la antena del lector y utilizar antenas clásicas de dipolo. Sólo es necesaria una capa de metal, lo que reduce el coste. Las antenas de dipolo, no obstante, no se ajustan muy bien a las características de los circuitos integrados típicos (con alta impedancia de entrada, ligeramente capacitiva). Se pueden utilizar dipolos plegados o bucles cortos como estructuras inductivas complementarias para mejorar la alimentación. Los dipolos de media onda (16 cm a 900 MHz) son demasiado grandes para la mayoría de aplicaciones (por ejemplo, las etiquetas RFID para uso en etiquetas no pueden medir más de 10 cm), por lo que hay que doblar las antenas para satisfacer las necesidades de tamaño. También pueden usarse estructuras de banda ancha. La ganancia de las antenas compactas suele ser menor que la de un dipolo (menos de 2 dB) y pueden considerarse isótropas en el plano perpendicular a su eje.

Los dipolos experimentan acoplamiento con la radiación que se polariza en sus ejes, por lo que la visibilidad de una etiqueta con una antena de dipolo simple depende de su orientación. Las etiquetas con dos antenas octogonales (etiquetas de doble dipolo) dependen mucho menos de ella y de la polarización de la antena del lector, pero suelen ser más grandes y caras que sus contrapartidas simples.

Pueden usarse antenas de parche (patch) para dar servicio en las cercanías de superficies metálicas, aunque es necesario un grosor de 3 a 6 mm para lograr un buen ancho de banda, además de que es necesario tener una conexión a tierra que incrementa el coste comparado con estructuras de una capa más sencillas. Las antenas HF y UHF suelen ser de cobre o aluminio. Se han probado tintas conductoras en algunas antenas encontrando problemas con la adhesión al circuito integrado y la estabilidad del entorno.

2.4.4. Asociación de etiquetas

Existen tres tipos básicos de etiquetas por su relación con los objetos que identifican: asociable, implantable e insertable (attachable, implantable, insertion). Además de estos tipos de etiquetas, Eastman Kodak ha presentado dos solicitudes de patente que tratan de la monitorización del consumo de medicina en forma de una etiqueta “digerible”.

2.4.5. Posicionamiento de las etiquetas

La orientación puede afectar al desempeño de etiquetas UHF a través del aire. En general, no es necesaria una recepción óptima de la energía del lector para operar sobre las etiquetas pasivas. No obstante, puede haber casos en los que se fija la distancia entre ambas partes, así como la potencia efectiva emitida. En este caso, es necesario saber en qué casos se puede trabajar de forma óptima con ellos.

Se definen los puntos denominados R (de resonancia, resonance spot), L (vivo, live spot) y D (muerto, dead spot) para especificar la localización de las etiquetas en un objeto marcado, de forma que estas aún puedan recibir la energía necesaria con base a unos niveles determinados de potencia emitida y distancia.

2.4.7. Entornos de etiquetas

El concepto de etiqueta RFID va asociado al de su ubicuidad. Esto supone que los lectores pueden requerir la selección de etiquetas a explorar de entre muchos candidatos posibles. También podrían desear realizar una exploración de las etiquetas de su entorno para realizar inventarios o, si las etiquetas se asocian a sensores y pueden mantener sus valores, identificar condiciones del entorno. Si un lector RFID intenta trabajar con un conjunto de etiquetas, debe conocer los dispositivos que se encuentran en su área de acción para después recorrerlos uno a uno, o bien hacer uso de protocolos de evitación de colisiones.

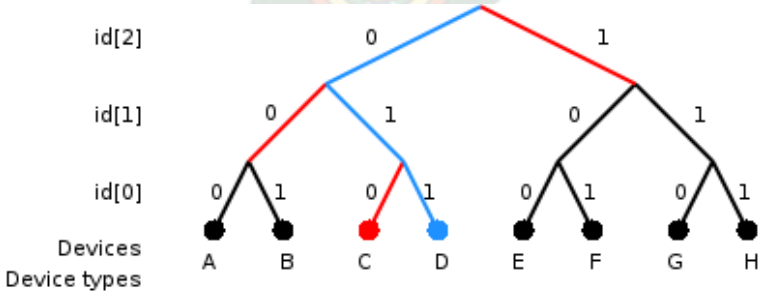

Figura 2.20. Identificación de etiquetas en un entorno de búsqueda

Fuente: https://es.wikipedia.org/wiki/RFID

Para leer los datos de las etiquetas, los lectores utilizan un algoritmo de singulación basado en el recorrido de árboles, resolviendo las colisiones que puedan darse y procesando secuencialmente las respuestas. Existen etiquetas bloqueantes (blocker tags) que pueden usarse para evitar que haya lectores que accedan a las etiquetas de un área sin necesidad de recurrir a comandos de suicidio para inhabilitarlas. Estas se hacen pasar por etiquetas normales, pero poseen ciertas características específicas; en concreto, pueden tomar cualquier código de identificación como propio, y pueden responder a toda pregunta que escuchen, asegurando el entorno al anular la utilidad de estas preguntas.

En general, puede emitirse una señal espuria si se detecta actividad de etiquetas para bloquear las transmisiones débiles producidas por estas. En caso de que las etiquetas sean prescindibles o no sean necesarias de nuevo, pueden inutilizarse induciendo en ellos corrientes elevadas que inutilicen sus circuitos.

Aparte de esto, una etiqueta puede ser promiscua, si responde a todas las peticiones sin excepción, o segura, si requiere autentificación (esto conlleva los aspectos típicos de gestión de claves criptográficas y de acceso). Una etiqueta puede estar preparada para activarse o desactivarse como respuesta a comandos del lector.

Los lectores encargados de un grupo de etiquetas en un área pueden operar en modo autónomo, en contraposición al modo interactivo. Si trabajan de esta forma, realizan una identificación periódica de todas las etiquetas en su entorno y mantienen una lista de presencia con tiempos de persistencia (timeouts) e información de control. Si una entrada expira, se elimina de la tabla.

Con frecuencia una aplicación distribuida requiere el uso de ambos tipos extremos de etiquetas. Las pasivas no pueden realizar labores de monitorización continua, sino que realizan tareas bajo demanda cuando los lectores se las solicitan. Son útiles para realizar actividades regulares y bien definidas con necesidades de almacenamiento y seguridad acotadas. Si hay accesos frecuentes, continuos o impredecibles, o bien existen requerimientos de tiempo real o

procesamiento de datos (como búsqueda en tablas internas) suele ser conveniente utilizar etiquetas activas.

2.4.8. Clasificación

Los sistemas RFID se clasifican dependiendo del rango de frecuencias que usan. Existen cuatro tipos de sistemas: de frecuencia baja (LF: 125 ó 134.2 KHz); de alta frecuencia (HF: 13.56 MHz); de frecuencia ultraelevada (UHF: 868 a 956 MHz); y de microondas (2.45 gigahercios). Los sistemas UHF no pueden ser utilizados en todo el mundo porque no existe una única regulación global para su uso.

2.4.9. Estandarización

Los estándares de RFID abordan cuatro áreas fundamentales:

- Protocolo en la interfaz aéreo: especifica el modo en el que etiquetas RFID y lectores se comunican mediante radiofrecuencia.
- Contenido de los datos: especifica el formato y semántica de los datos que se comunican entre etiquetas y lectores.
- Certificación: pruebas que los productos deben cumplir para garantizar el desarrollo de los estándares y pueden interoperar con otros dispositivos de distintos fabricantes.
- Aplicaciones: usos de los sistemas RFID.

Como en otras áreas tecnológicas, la estandarización en el campo de RFID se caracteriza por la existencia de varios grupos de especificaciones competidoras. Por una parte está ISO, y por otra Auto-ID Centre (conocida desde octubre de 2003 como EPCglobal, de EPC, Electronic Product Code). Ambas comparten el objetivo de conseguir etiquetas de bajo coste que operen en UHF.

Los estándares EPC para etiquetas son de dos clases:

- Clase 1: etiqueta simple, pasiva, de sólo lectura con una memoria no volátil programable una sola vez.
- Clase 2: etiqueta de sólo lectura que se programa en el momento de fabricación del chip (no reprogramable posteriormente).

Las clases no son ínteroperables y además son incompatibles con los estándares de ISO. Aunque EPCglobal está desarrollando una nueva generación de estándares EPC está (denominada Gen2), con el objetivo de conseguir interoperabilidad con los estándares de ISO, aún se está en discusión sobre el AFI (Application Family Identifier) de 8 bits.

Por su parte, ISO ha desarrollado estándares de RFID para la identificación automática y la gestión de objetos. Existen varios estándares relacionados, como ISO 10536, ISO 14443 e ISO 15693, pero la serie de estándares estrictamente relacionada con las RFID y las frecuencias empleadas en dichos sistemas es la serie 18000.

2.5. MÓDULO LECTOR RFID-RC522 RF

Figura 2.21. Lector RFID y tags
Fuente: www.prometec.net/arduino-rfid

El Módulo Lector RFID-RC522 RF utiliza 3.3V como voltaje de alimentación y se controla a través del protocolo SPI, así como el protocolo UART, por lo que es compatible con casi cualquier micro controlador, Arduino o tarjeta de desarrollo. El RC522 también utiliza un sistema avanzado de modulación y demodulación para todo tipo de dispositivos pasivos de 13.56Mhz. Puesto que se hará una lectura y escritura de la tarjeta, es necesario conocer las características de los bloques de memoria una tarjeta: La tarjeta que viene con el módulo RFID cuenta con 64 bloques de memoria (0-63) donde se hace lectura y/o escritura. Cada bloque de memoria tiene la capacidad de almacenar sobre todo hasta 16 Bytes. El número de serie consiste de 5 valores hexadecimales, se podría utilizar esto para hacer una operación dependiendo del número de serie.

2.5.1. Características

- Modelo: MF522-ED
- Corriente de operación: 13-26mA a 3.3V
- Isb de stand by: 10-13mA a 3.3V
- Ism de sleep-mode: < 80uA
- Im máxima: 30mA
- Frecuencia de operación: 13.56Mhz
- Distancia de lectura: 0 a 60mm
- Protocolo de comunicación: SPI
- Velocidad de datos máxima: 10Mbit/s
- Dimensiones: 40 x 60 mm
- Temperatura de operación: -20 a 80ºC
- Humedad de operación: 5%-95%
- Máxima velocidad de SPI: 10Mbit/s
- Incluye pines, llavero y tarjeta

2.5.2. Conexión con Arduino UNO

A continuación, se muestra una tabla con los pines del Módulo Lector RFID-RC522 RF, así como la conexión que tendrá con el Arduino UNO

.Tabla 2.2. RFID conexión RC522 con Arduino UNO

ARDUINO UNO	RFID RC552
DIGITAL PIN #10	SDA
DIGITAL PIN #13	SCK
DIGITAL PIN #11	MOSI
DIGITAL PIN #12	MISO
N/A	IRQ
POWER GND	GND
DIGITAL PIN #5	RST
POWER 3.3 V	3.3 V

Fuente: www.prometec.net/arduino-rfid

2.6. ESP8266

2.6.1. Definición

Para poder conectar las placas Arduino a Internet existen muchas posibilidades, incluidas placas oficiales de Arduino para realizar esta tarea, tanto por wifi como por cable Rj45, con un funcionamiento muy correcto como por ejemplo las siguientes dos placas:

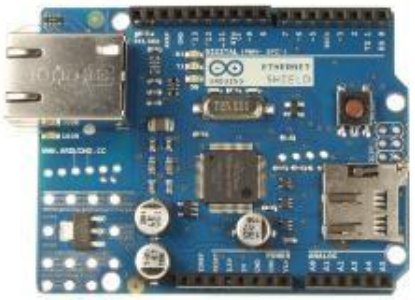

Figura 2.22. Arduino Ethernet Shield

Fuente: www.prometec.net/arduino-rfid

Figura 2.23. Arduino Wifi Shield

Fuente: www.prometec.net/arduino-rfid

El único pero que pueden tener estos módulos es que tienen un precio un poco elevado 69€ la placa wifi y 29€ la placa Ethernet, en este apartado de estudiará y se utilizará un módulo de reciente aparición y un bajísimo precio, el esp8266 que con un precio cercano a los 3€ es una opción a tener muy en cuenta.

Figura 2.24. Esp8266 ESP-01

Fuente: www.prometec.net/arduino-rfid

Hay distintas versiones del módulo, pero se utilizará en este proyecto la versión ESP-01 dado que es la que se dispone.

El módulo incluye todo lo necesario para conectarse por WIFI mediante comandos AT utilizando el puerto serie, en verdad es una placa que funciona de forma independiente del Arduino, pero mediante comandos AT podemos utilizarla como si el propio Arduino estuviera conectado al WIFI.

2.6.2. Características técnicas

- Protocolos soportados: 802.11 b/g/n
- Wi-Fi Direct (P2p), Soft Access Point
- Pila TCP/IP integrado
- PLL, reguladores y unidades de manejo de energía integrados
- Potencia de salida: +19.5dBm en modo 802.11b
- Sensor de temperatura integrado
- Consumo en modo de baja energía: <10 uA
- Procesador integrado de 32 bits, puede ser utilizado como procesador de aplicaciones
- Wi-Fi 2.4 GHz, soporta WPA/WPA2
- Tamaño ultra reducido (11.5mm x 11.5mm)
- Conversor analógico a digital de 10-bit
- Soporta variedad de antenas
- SDIO 2.0, SPI, UART, I2C
- Encendido y transmisión de datos en menos de 2ms
- Rango de operación -40C° ~ 125C°

Hay que tener en cuenta que, dependiendo de la versión de firmware del módulo, los comandos AT del mismo pueden variar. Hay muchas versiones de firmware, incluso hay alguna para poder utilizar el lenguaje Lua y así facilitar su uso (https://nodelua.org/). Pero en este Proyecto se utilizará el firmware que viene en el módulo ya preinstalado.

2.6.4. Circuito

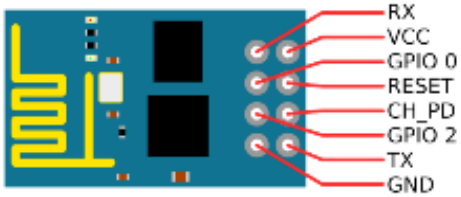

Figura 2.25. Pines Esp8266

Fuente: www.prometec.net/arduino-rfid

Para realizar circuito básico para poder utilizar el módulo hay que tener cuidado con la alimentación del módulo ya que es a 3,3v y hay que utilizar una alimentación externa al Arduino dado que éste no parece ser suficiente para alimentar el módulo, por lo que tendremos que unir ambas GNDs para el correcto funcionamiento del mismo.

Para la comunicación con el módulo se utilizará la librería SoftWareSerial.h para poder utilizar otros pines como puerto serie, es este caso utilizaremos los pines 2 y 3 del Arduino y de esta forma poder seguir utilizando el puerto serie del Arduino con nuestro ordenador, que utiliza los pines 0 y 1.

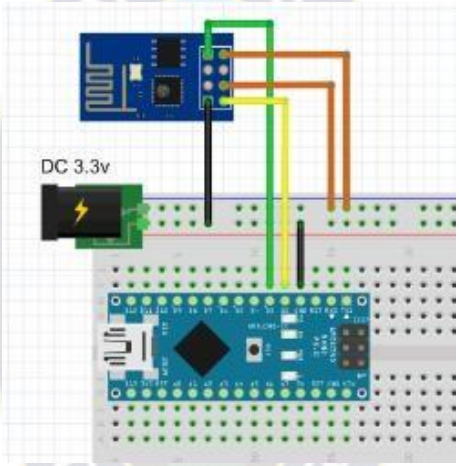

Figura 2.26. Circuito Esp8266 Básico

2.6.5. Pruebas

Con el programa basicoEsp8166.ino leemos los caracteres del puerto serie del ordenador y lo escribimos en el del módulo esp8266 y viceversa. Cargamos el programa y abrimos el Monitor Serie.

2.6.6. Conexión básica:

Comprobamos que funciona el circuito y el módulo escribiendo:

AT

Lo que nos responderá:

OK

Esto es que se comunican de forma correcta, si aparecieran caracteres extraños u otra cosa probaremos a seleccionar otra velocidad en el Monitor Serie.

AT+GMR

Nos devolverá la versión del firmware en mi caso:

0018000902-AI03

Escribimos:

AT+CIOBAUD?

Y nos devuelve la velocidad de comunicación del puerto, se puede cambiar con AT+CIOBAUD=9600 por ejemplo, ahora escribimos:

AT+CWMODE?

Que devuelve el modo en el que trabaja el esp8266, para cambiar el modo de funcionamiento escribiremos AT+CWMODE=1 por ejemplo. Seguimos escribiendo comandos:

AT+CWLAP

Nos devolverá una lista de redes que están dentro del alcance del esp8266 con el siguiente formato: +CWLAP: ecn,ssid,rssi,mac

Donde:

- ecn: 0 OPEN , 1 WEP , 2 WPA_PSK , 3 WPA2_PSK , 4 WPA_WPA2_PSK
- ssid: string, SSID del punto de acceso
- rssi: fuerza de la señal
- mac: string, dirección MAC

Para conectarnos a una de las redes listadas anteriormente escribiremos:

AT+CWJAP="SSID","password"

y con:

AT+CIFSR

Nos devolverá la IP que nos ha sido asignada.

Una vez que nos hemos conectado a una red el módulo por defecto guarda la última conexión y a no ser que indiquemos lo contrario tratará de conectarse de forma automática otra vez a la última red a la que se conectó.

2.6.7. Probando modo servidor

Asignamos el modo deseado

AT+CWMODE=1

Comprobamos que se realizó correctamente:

AT+CWMODE?

Permitiremos conexiones múltiples:

AT+CIPMUX=1

Y activaremos el modo servidor en el puerto 80:

AT+CIPSERVER=1,80

Desde un navegador del ordenador escribimos la IP que nos ha sido asignada y el puerto:

192.168.1.128:80

Con lo que obtenemos la petición del navegador por el Monitor Serie:

Link

+IPD,0,308:GET / HTTP/1.1

Host: 192.168.1.128

User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:40.0) Gecko/20100101 Firefox/40.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

Accept-Language: es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3

Accept-Encoding: gzip, deflate

Connection: keep-alive

OK

Y le mandamos la respuesta:

AT+CIPSEND=0,13

Le vamos a mandar 13 caracteres por el canal 0, en el monitor serie aparecerá el símbolo > esperando esos 13 caracteres que le hemos indicado, escribimos, por ejemplo

Hola Mundo!!! y tras esto cerramos el canal:

AT+CIPCLOSE=0

En el navegador nos aparecerá la respuesta:

Hola Mundo!!!

2.7. CONECTANDO CON THINGSPEAK.COM

ThingSpeak es una plataforma web para el internet de las cosas, en la cual podremos almacenar los datos que consigamos con nuestros sensores como pueda ser la temperatura de una forma muy sencilla y poder procesar esos datos mediante gráficas, por ejemplo.

Es Software libre y su API para procesar peticiones HTTP está disponible en un repositorio de GitHub (https://github.com/iobridge/thingspeak)

Los pasos para utilizarlo consisten en:

1. Nos creamos una cuenta.
2. Creamos un canal, por ejemplo, Temperatura.
3. Añadimos un field al canal.
4. Nos darán un Write API Key del canal que utilizaremos para identificar dicho canal, en nuestro caso nos dio: 5G87A0JO4U5WCS2S

Ahora utilizaremos el módulo esp8266 para actualizar el valor del field que hemos creado antes por lo que seguiremos utilizando el circuito anterior y el Monitor Serie como hasta ahora:

Establecemos una conexión con la IP de ThingSpeak mediante el puerto 80:

AT+CIPSTART=4,"TCP","184.106.153.149",80

Nos responderá:

OK Linked

Indicamos el número de caracteres que vamos a enviar:

AT+CIPSEND=4,43

Y escribimos cuando aparezca el símbolo >

GET /update?key=5G87A0JO4U5WCS2S&field1=5

Nos responderá:

SEND OK

+IPD,4,1:5

OK Unlink

Ahora solo nos falta cerrar la conexión:

AT+CIPCLOSE

2.8. SERVICIOS WEB REST

REST (Representational State Transfer) o Transferencia de Estado Representacional es un estilo de programación de servicios web centrado en los recursos y en sus representaciones sin las abstracciones de los protocolos basados en patrones de intercambio de mensajes, como por ejemplo SOAP.

Una de las características es que es un protocolo cliente/servidor sin estado, toda información necesaria tiene que estar en la propia consulta http

El acceso a los recursos se hace mediante los métodos estándar de http:

- PUT – Modificar un recurso
- GET – Leer un recurso
- POST – Añadir un recurso
- DELETE – Borrar un recurso

Los recursos se identifican mediante identificadores de recursos uniformes URI (Uniform Resource Identifier) y un recurso puede tener diferentes representaciones (texto, xml, json…)

2.9. NodeMCU

Figura 2.27. NodeMCU DEVKIT 1.0

Fuente: www.nodemcu.com

NodeMCU es una plataforma IoT de código abierto. Incluye firmware que se ejecuta en el ESP8266 Wi-Fi SoC de Espressif Systems, y el hardware, que se basa en el módulo ESP-12. El término "NodeMCU" por defecto se refiere al firmware en lugar de los kits de desarrollo. El firmware utiliza el lenguaje de secuencias de comandos Lua. Se basa en el proyecto eLua, y se basa en el SDK Espressif no OS para ESP8266. Utiliza muchos proyectos de código abierto, como lua-cjson, y spiffs.

NodeMCU se creó poco después de la salida del ESP8266. El 30 de diciembre de 2013, Espressif Systems comenzó la producción de la ESP8266. El ESP8266 es un Wi-Fi SoC integrado con un núcleo de Tensilica Xtensa LX106, ampliamente utilizado en aplicaciones de IoT. NodeMCU comenzó el 13 de octubre de 2014, cuando Hong comprometió el primer archivo de nodemcu-firmware a GitHub. Dos meses más tarde, el proyecto se expandió para incluir una plataforma de hardware abierto cuando el desarrollador Huang R comprometió el archivo gerber de una placa ESP8266, llamada devkit v0.9. Más tarde ese mes, Tuan PM portó la biblioteca cliente MQTT de Contiki a la plataforma SoC ESP8266, y comprometido con el proyecto NodeMCU, entonces NodeMCU fue capaz de soportar el protocolo MQTT IoT, utilizando Lua para acceder al agente MQTT. Otra actualización importante se hizo el 30 de enero de 2015, cuando Devsaurus portó el u8glib a NodeMCU proyecto, permitiendo NodeMCU para conducir fácilmente LCD, pantalla, OLED, incluso VGA muestra.

En el verano de 2015 los creadores abandonaron el proyecto de firmware y un grupo de colaboradores independientes pero dedicados tomó el relevo. Para el verano de 2016 el NodeMCU incluyó más de 40 módulos diferentes. Debido a limitaciones de recursos, los usuarios deben seleccionar los módulos relevantes para su proyecto y construir un firmware adaptado a sus necesidades.

2.10. MEAN STACK

2.10.1. ¿Qué es MEAN STACK?

MEAN es el acrónimo que referencia Arquitecturas desarrolladas con MongoDB, Express.js , Angular.js y Node.js. Los cuatro son nuevos productos fuertemente ligados al mundo Javascript.

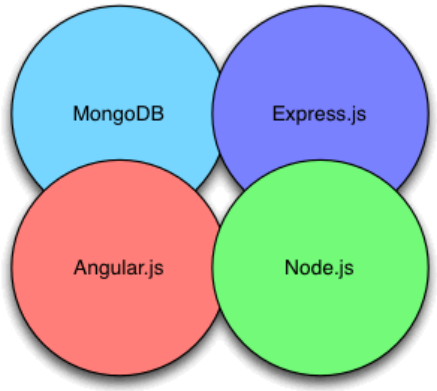

Figura 2.28. Arquitectura MEAN
Fuente: www.udemy.com/angular-2

Cada una de estas tecnologías aporta una funcionalidad dentro del Stack:

- MongoDB: Es la base de datos NoSQL que nos permitirá almacenar información.
- Node.js : Permite ejecutar Javascript del lado del servidor. Es nuestro nuevo servidor de aplicaciones
- Express.js : Es el framework JavaScript Web del Servidor
- Angular.js: Es el framework JavaScript del Cliente.

Las cuatro tecnologías unidas nos permiten desarrollar aplicaciones escalables, cercanas al mundo móvil y a la necesidad de tiempo real.

En los últimos años se han puesto muy de moda tecnologías como MongoDB, Node.js y AngularJS, lo que ha hecho que haya surgido una nueva corriente para la construcción de aplicaciones web rápidas, robustas y mantenibles usando las tecnologías de MongoDB, Express, AngularJS y Node.js. El uso de estas 4 tecnologías juntas conforma el llamado "stack MEAN" (MEAN = **M**ongo-**E**xpress-**A**ngular-**N**ode) que tienen como nexo común la utilización del mismo lenguaje de programación (JavaScript), lo cual supone una gran ventaja para los que sepan Javascript, que por otro lado no es un lenguaje muy complejo. Por tanto, con estas 4 tecnologías tenemos todo lo necesario para desarrollar aplicaciones web: Frontend, Backend y Base de Datos.

2.10.2. ¿Cómo funciona MEAN?

Como ya se ha dicho, con el uso de estas 4 tecnologías tenemos todo lo necesario (Frontend, Backend y Base de Datos) para desarrollar una aplicación web, pero os preguntareis cual es la finalidad y la funcionalidad de cada una de estas tecnologías. En la siguiente imagen podeis ver cuál es el flujo de las aplicaciones web realizadas con MEAN.

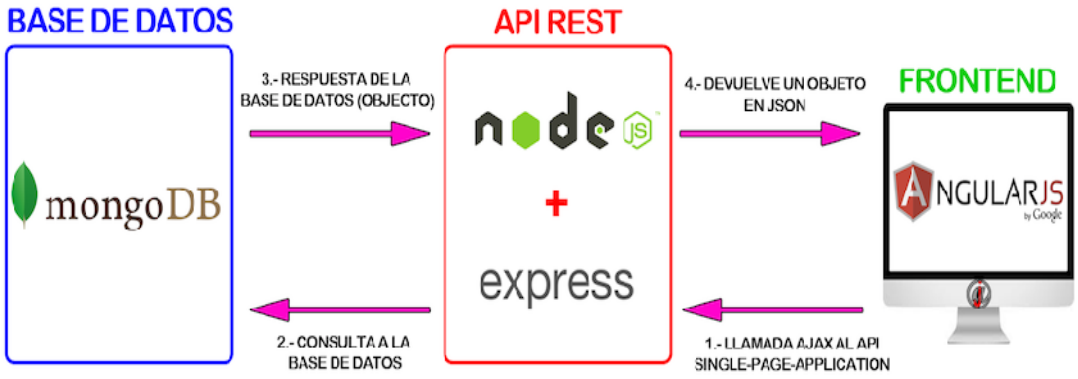

Figura 2.29. Funcionamiento de la arquitectura MEAN

Fuente www.udemy.com/angular-2

Como se observa en la imagen, el Frontend lo desarrollamos con Angular, que es el encargado de hacer llamadas al API REST (Post, Put, Get y Delete) desarrollado en NodeJS que utiliza el framework de Express. El API podrá hacer un CRUD (Create-Read-Update-Delete) a

la base de datos MongoDB y cuando el API tenga los datos que se le han pedido en la llamada los devolverá a Angular (es decir al Frontend) en formato JSON y este los mostrará en pantalla sin necesidad de recargar la página ya que Angular mantiene el modelo de datos actualizado sin necesidad de recargar la página.

Como vemos esta es la arquitectura y el flujo de datos que hay en las aplicaciones desarrolladas con estas tecnologías. Dependiendo del proyecto podremos meter más lógica en la parte de Frontend o en la de Backend, pero eso ya son cosas que hay que valorar según las características del proyecto. Por último, comentar una cosa que me ha sorprendido gratamente de Angular y es que desde el propio HTML nos permite validar formularios con las etiquetas que nos proporciona, por tanto, no será necesario validar los formularios en Javacript como antaño.

a) Node JS

Figura 2.30. Logotipo de nodeJS
Fuente: www.nodejs.com

Esto no lo sabe todo el mundo pero, de hecho, JavaScript nació en el lado servidor casi al mismo tiempo que en el navegador. La propia empresa que creó el lenguaje, Netscape, lo incluyó también en su servidor web llamado Netscape Enterprise Server, aunque con poco éxito. Poco después Microsoft lo incorporó a su ASP clásico, donde todavía funciona.

No es sin embargo hasta hace relativamente poco, en 2009, cuando un proyecto empezó a poner en el mapa realmente a JavaScript en el lado servidor. Se trata de Node.js, un entorno de ejecución de aplicaciones multiplataforma y de código abierto.

Node.js utiliza por debajo el motor de JavaScript de Google, denominado V8, y provee de una arquitectura orientada a eventos (como la de los navegadores) así como una serie de APIs no-bloqueantes (asíncronas) que le proporcionan un rendimiento y una escalabilidad muy elevadas. Se puede utilizar para crear cualquier tipo de lógica de aplicación, pero dado que incorpora un módulo para poder actuar como un servidor web, es especialmente popular para crear aplicaciones web. Actualmente lo emplean para sus aplicaciones multitud de empresas de todos los ámbitos, pero especialmente de Internet: PayPal, SAP, Groupon, Microsoft, que ayuda a crear la versión para Windows (bajo IIS funciona de maravilla, con grandes ventajas frente a hostearlo solo) y lo soporta en Azure y en sus herramientas de desarrollo.

b) Express

Node.js por sí mismo está muy bien para crear la lógica de las aplicaciones, y como dispone de un módulo para el protocolo HTTP es posible crear aplicaciones web con lo que trae por defecto. Sin embargo, es complejo, costoso y tenemos que hacer todo el trabajo a bajo nivel nosotros mismos.

Para ayudar a crear aplicaciones web más fácilmente nació Express. Este framework está escrito en JavaScript para Node.js. Su objetivo es que no tengamos que reinventar la rueda cada vez que queramos crear una aplicación web, ofreciéndonos soporte para las principales necesidades en este tipo de aplicaciones: gestión de peticiones y respuestas, cabeceras, rutas, vistas, etc.

Digamos que Node.js sin Express.js sería mucho menos de lo que es hoy. Y costaría 10 veces más construir aplicaciones web con JavaScript. Así que van unidos felizmente en el servidor para permitirnos crear aplicaciones web de manera sencilla, por eso los he puesto juntos en la capa del medio de la figura anterior.

c) Angular 4.0

Aunque evidentemente en el navegador podríamos desarrollar todo el código usando solamente HTML, CSS y JavaScript puro, qué duda cabe que disponer de alguna biblioteca que nos proporcione muchas funcionalidades ya hechas, facilita mucho las cosas. Es algo parecido a lo que pasa con Node.js y Express.js: el primero puede funcionar sin el segundo, pero seríamos tontos si no utilizásemos este último.

En el navegador desde siempre han existido bibliotecas de funciones que nos facilitan mucho la vida. La más conocida y utilizada es sin duda jQuery, que también fue una de las primeras en ser adoptada universalmente. En los últimos años se ha trasladado al navegador el patrón de diseño denominado MVC (Modelo-Vista-Controlador) y han surgido cientos de bibliotecas especializadas en facilitarnos su uso. De entre todas ellas destaca Angular que ha tomado un especial protagonismo en los últimos tiempos, entre otras muchas cosas porque está creada y soportada por Google, es gratuita y de código abierto.

Algunos han definido a AngularJS como "lo que HTML debería haber sido si se hubiese diseñado para crear aplicaciones". Ahí queda eso.

Angular va mucho más allá de ser una simple biblioteca de funciones: es un completo framework que nos brinda todo tipo de funcionalidades avanzadas, extendiendo de hecho HTML.

d) MongoDB

En el lado del almacenamiento se han utilizado tradicionalmente bases de datos relacionales. Sin embargo, actualmente, los tipos de información que suelen requerir las aplicaciones web demandan mayor flexibilidad, menos coherencia y sobre todo mayor capacidad de escalar. Para dar respuesta a todo esto surge la tendencia tecnológica en almacenes de datos que se denomina NoSQL.

Estos almacenes de datos NoSQL pueden ser de diversos tipos, pero en muchos casos utilizan JavaScript para representar la información**,** recibiendo, enviando y almacenando datos usando la notación JSON.

De entre todos estos almacenes de datos no-relacionales hay uno que destaca especialmente: MongoDB, y que es la "M" del stack MEAN.

e) JSON

Esto no forma parte del acrónimo MEAN, pero es una parte integral del mismo. Es también a su vez una palabra formada por iniciales de palabras, y significa: JavaScript Simple Object Notation**.**

Como su propio nombre indica JSON permite representar objetos en forma de código JavaScript que luego podemos evaluar. Su padre es Douglas Crockford.

Este formato es ligero, fácilmente legible por un humano, pero también por un ordenador, está basado en JavaScript y además es muy fácil de procesar por parte de cualquier intérprete de este lenguaje.

Por ejemplo, este es un objeto JSON que representa un contacto:

```
{"nombre" : "José Manuel",
 "apellidos”: "Alarcón Aguín",
 "empresa”: "campusMVP",
 "teléfono”: "986 165 802",
 "edad”: 42}
```

Como vemos no es más que un objeto JavaScript normal y corriente. Crearlo es muy fácil pues es sintaxis JavaScript convencional. En www.json.org es posible encontrar una explicación completa de esta notación.

JSON es el pegamento de todas las capas, el formato en el que se transfieren los datos entre todos los niveles de la aplicación: navegador, servidor web y servidor de datos. Aunque no forme parte del nombre es una parte esencial del stack MEAN.

2.11. PTV VISSIM

Es una herramienta de software para la simulación microscópica y multimodal del tránsito, desarrollada por la empresa PTV -- Planung Transport Verkehr AG en Karlsruhe, Alemania. El acrónimo deriva del alemán “Verkehr In Städten - SIMulation” (en castellano “Simulación de tránsito en ciudades"). El fundamento teórico de VISSIM se sitúa en la universidad de Karlsruhe en los años 80 y su primera aparición como herramienta comercial en entorno Windows de Microsoft fue en el año 1992, con la versión 2.03. Actualmente lidera el mercado mundial.

Con PTV Vissim, puede simularse la situación del tráfico a la perfección, tanto la comparación de operar con distintos tipos de intersecciones como el análisis de implementar medidas de prioridad al transporte público o el impacto de un distinto plan de semaforización. PTV Vissim, como software líder mundial para la simulación microscópica del tráfico, en un solo modelo permite representar a todos los usuarios de la vía pública y estudiar sus interacciones: autos, transporte de carga y cualquier tipo de transporte público, ya sea ferroviario o convencional. Para ello, los modelos de comportamiento vehicular, científicamente desarrollados y validados, proporcionan una simulación realista de todos los agentes.

El software ofrece una gran flexibilidad en múltiples aspectos: el concepto de arcos y conectores permite que los usuarios modelen geometrías de cualquier tipo, por muy complejas que sean. Las características de conductores y vehículos permiten una parametrización individual. Además, gracias a la gran variedad de interfaces se pueden integrar sin dificultades otros sistemas de control semafórico, gestión del tráfico, o modelos de emisiones.

Las amplias posibilidades de análisis hacen de PTV Vissim una herramienta potente para evaluar y planificar la infraestructura vial tanto urbana como inter-urbana. Con este software se

pueden obtener tanto resultados numéricos detallados como impresionantes animaciones en 3D representando diversos escenarios. Resulta un recurso ideal para presentar propuestas de infraestructura tanto ante los agentes responsables de la toma de decisiones, como a la opinión pública, de forma comprensible y convincente.

a) Simulación microscópica

En un modelo de simulación microscópica o modelo de microsimulación los individuos que componen los flujos de tránsito (vehículos, bicicletas, peatones, etc.) son el elemento mínimo. Sus características (físicas y psicológicas) y su interacción mutua y con elementos viales son modelizadas con reglas, algoritmos y modelos de comportamiento. Un modelo de microsimulación del tránsito es dinámico (que evoluciona en el tiempo), discreto (el estado de las variables cambia instantáneamente en tiempos puntuales, normalmente fijos) y estocástico (con resultados aleatorios).

b) Simulación multimodal

La simulación multimodal se particulariza por modelar más de un tipo de tránsito y las interacciones entre estos. En VISSIM pueden simularse los siguientes tipos de tránsito:

- Vehículos (coches, buses, camiones, motocicletas, etc.)
- Transporte público (tranvías, buses, etc.)
- Bicicletas
- Peatones
- Rickshaws

c) Aplicación

El ámbito de aplicación de VISSIM comprende desde la ingeniería del tránsito (sincronización y planificación de planes semafóricos, experimentación con sistemas inteligentes de transporte y sistemas de control y gestión del tránsito) pasando por la planificación del transporte, estudios de movilidad hasta visualizaciones en 3D para documentación ilustrativa y presentaciones.

d) Fundamento científico

El modelo de movimiento básico de VISSIM fue desarrollado por Rainer Wiedemann en el 1974 en la Universidad de Karlsruhe (Universität Karlsruhe (TH)). Se trata de un modelo de seguimiento entre vehículos que considera los aspectos físicos y psicológicos de los conductores.

CAPITULO III

MODELADO Y DESARROLLO DEL PROTOTIPO

El prototipo del sistema web basado en internet de las cosas se dividió en tres partes las cuales son:

- Cliente
- Servidor
- Electrónica

El sistema web se realizó utilizando el formato MEAN SATCK mencionado anteriormente por lo cual el almacenamiento de datos de las lecturas RFID, actuadores y sensores que nos envíen los nodos, se las guardaran en la base de datos MONGODB , se ha seleccionado este sistema de gestión de base de datos porque la comunicación debe realizarse en tiempo real y mongoDB está orientado al manejo de documentos esto hace posible que el acceso a los datos sea de una manera muy rápida, también es uno de los sistemas más extendidos y está muy documentado a la vez de que dispone de herramientas tales como por ejemplo ROBOMONGO, la cual nos facilita mucho la gestión de la base de datos, pero podríamos haber utilizado cualquier otro sistema de base de datos.

Las herramientas utilizadas para la implementación del sistema web se muestran en la siguiente tabla

Tabla 3.1: Herramientas utilizadas para el sistema web

Nº	HERRAMIENTA	VERSION
1	HTML	5.0
2	CSS	3.0
3	BOOTSTRAP	4.0
4	ANGULAR	4.0
5	NODE JS	7.7.4
6	EXPRESS	4.15.2
7	MONGO DB	3.4.3
8	ROBOMONGO	1.0

3.1. SERVICIOS REST

En este apartado definiremos los servicios REST a utilizar por parte de los módulos WIFI utilizados y su implementación.

Tabla 3.2: Servicios REST implementados finalmente

Nº	HTTP	URL	ACCION
1	**GET**	http://192.168.1.25 / api / entrada / **:**grupo / **:**flujo / **:**rfid / **:**codigo / **:**altura_vehiculo	Registra el número de grupo, el número de flujo, el rfid, el encriptado y la altura del vehículo
2	**GET**	http://192.168.1.25 / api / salida / :grupo / :flujo / :rfid / :codigo / **:**estado_semaforo	Registra el número de grupo, el número de flujo, el rfid, el encriptado y el estado del semáforo
3	**GET**	http://192.168.1.25 / api / **:**humo / **:**sonido / **:**temperatura	Registra los datos analógicos tomados del sensor de humo, sonido y temperatura
4	**GET**	http://192.168.1.25 / api / **:**id_semaforo	Obtiene los tiempos asignados a las luces del semáforo con código igual a ID

Se detectó que existen ciertos problemas a la hora de realizar PUTs utilizando el módulo esp8266 con el firmware original y recomiendan actualizar los datos mediante GETs y pasando los datos como argumento en la misma petición, por lo que se creará también ese servicio, dado lo anterior nuestros servicios REST quedarán finalmente como en la tabla.

Los servicios REST que necesitamos los crearemos en un archivo llamado ‘api.js’ el cual lo pondremos dentro de la ruta “/ backend / routes /”

3.2. NODOS

Se implementará uno de los nodos el cual está compuesto por cuatro placas Arduinos, las cuales tienen distinta funcionalidad dependiendo de la ubicación del mismo los cuales se muestran en la siguiente figura.

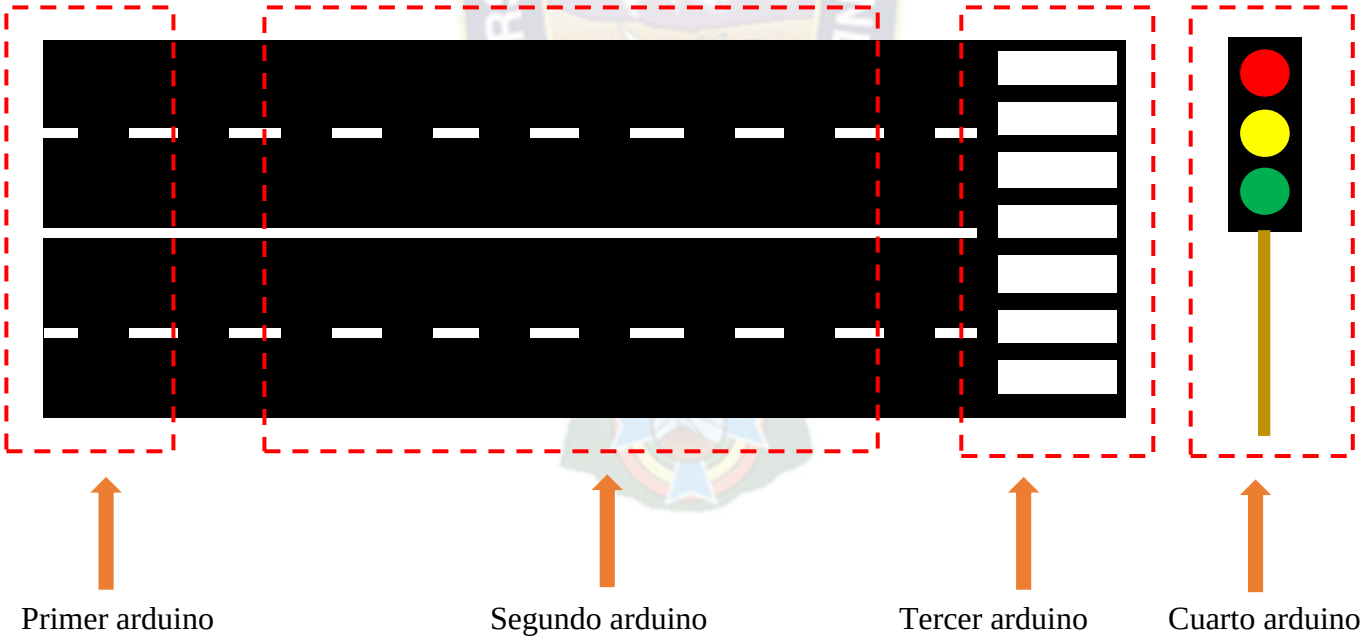

Figura 3.1: Ubicación de las placas Arduino Uno

3.2.1. Circuito

La implementación de cada uno de los circuitos correspondientes a las cuatro placas Arduino se las mencionara de forma separada para un mayor entendimiento, las cuales en si son la unión de los circuitos vistos en el capítulo dedicado a los sensores y actuadores.

Para que cada arduino funcione de forma correcta, necesitamos dos fuentes de tensión regulada, una de 5v para alimentar al Arduino y al resto de componentes y otra de 3,3v para alimentar al módulo esp8266, esto lo podemos conseguir utilizando reguladores como pueden ser la familia de reguladores AMS1117 que nos ofrece variedad de reguladores para distintos voltajes.

Siguiendo con la línea de implementar un circuito se encontró un módulo para poder alimentar los circuitos directamente en la protoboard y a un bajísimo precio, por lo que se decidió utilizar dicho módulo para tal fin. El módulo en cuestión es el "YwRobot Breadboard Power Supply", este módulo utiliza dos reguladores de la familia AMS1117 para ofrecernos los 3,3v y los 5v que necesitamos a la vez que nos proporciona conector de alimentación, interruptor y led de estado.

Figura 3.2: Módulo YwRobot Breadboard Power Supply
Fuente: www.petervis.com

También se ha incluido un jumper para aislar al Arduino de los 5v del regulador por si fuera necesario cuando tenemos al Arduino conectado por USB.

a) Primer Circuito

Este circuito corresponde a la unión del módulo RFID, Placa Arduino UNO, Sensor Ultrasónico y el módulo esp8266. Utilizando los pines digitales 9,10,11,12 y 13 del Arduino para hacer la conexión con los pines RESET, SDA, SCK, MOSI y MISO del módulo RFID respectivamente, los pines A0 y 7 para el sensor ultrasónico y finalmente la conexión del módulo esp8266 al arduino mencionado en el anterior capitulo.

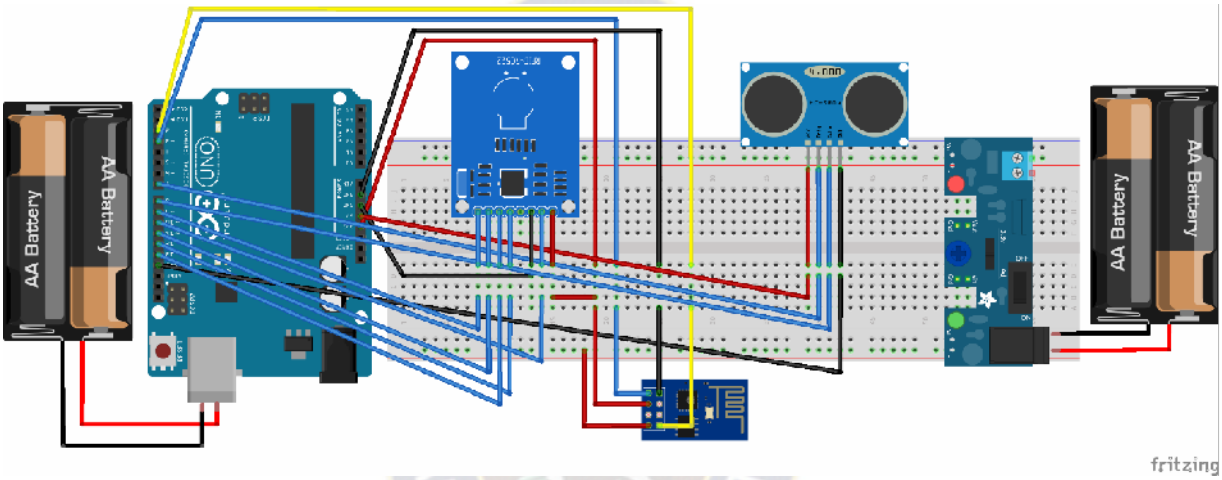

Figura 3.3: Muestra grafica del armado del primer circuito

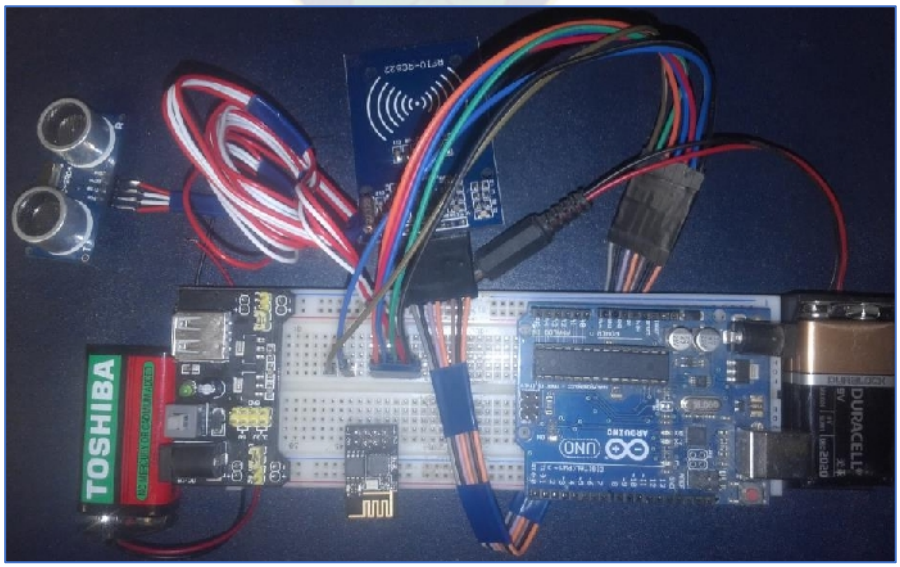

Figura 3.4: Vista superior del armado del primer circuito

b) Segundo Circuito

Este circuito es muy similar al anterior con la diferencia que no cuenta con el sensor ultrasónico, corresponde también a la unión del módulo RFID, Placa Arduino UNO y el módulo esp8266. Utilizando los pines 9,10,11,12 y 13 del Arduino para hacer la conexión con los pines RESET, SDA, SCK, MOSI y MISO del módulo RFID respectivamente y finalmente la conexión entre la placa arduino UNO y el módulo esp8266 mencionado en el anterior capitulo

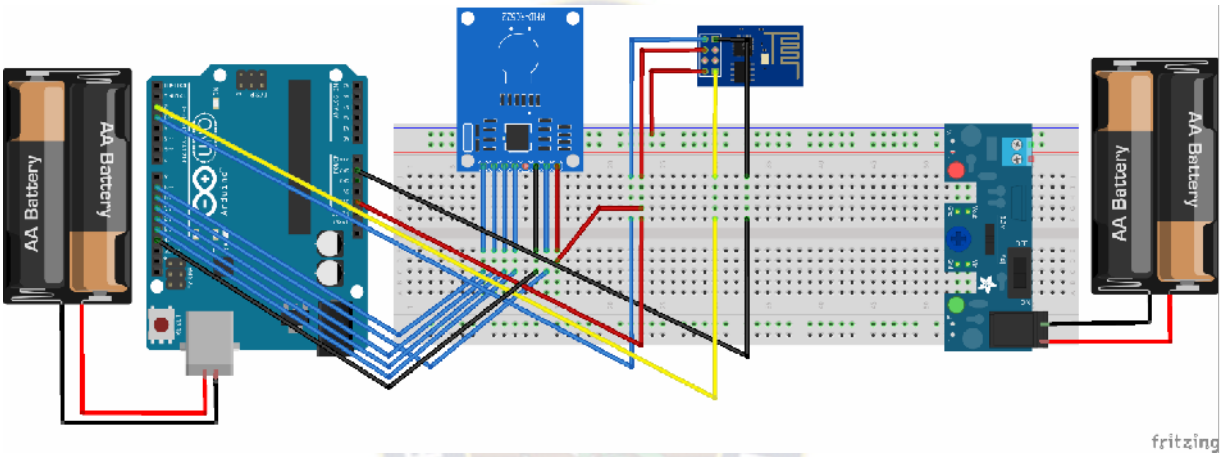

Figura 3.5: Muestra grafica del armado del segundo circuito

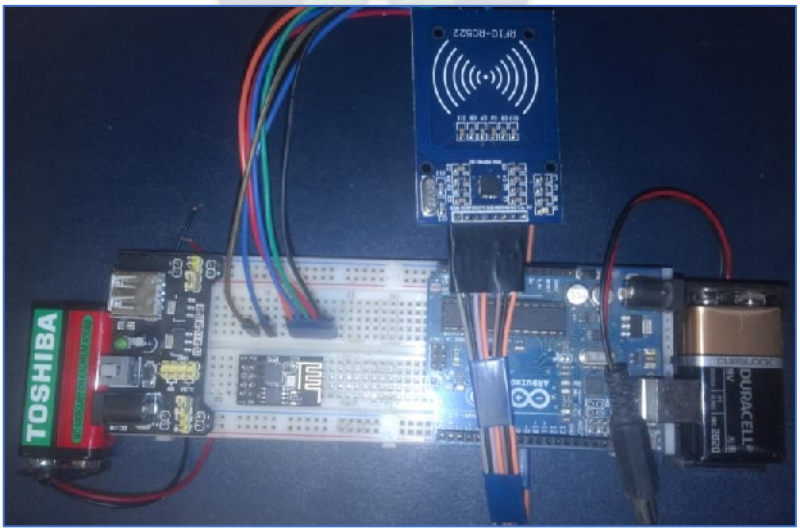

Figura 3.6: Vista superior del armado del segundo circuito

c) Tercer Circuito

En este circuito se hace la unión de la placa Arduino UNO, sensor de sonido, sensor humo, sensor de temperatura y el módulo esp8266. Utilizando los pines analógicos A0, A1 y A2 del arduino para hacer la conexión con los sensores a los pines A0 de los sensores respectivamente y finalmente la conexión entre la placa arduino UNO y el módulo esp8266 mencionado en el capítulo anterior.

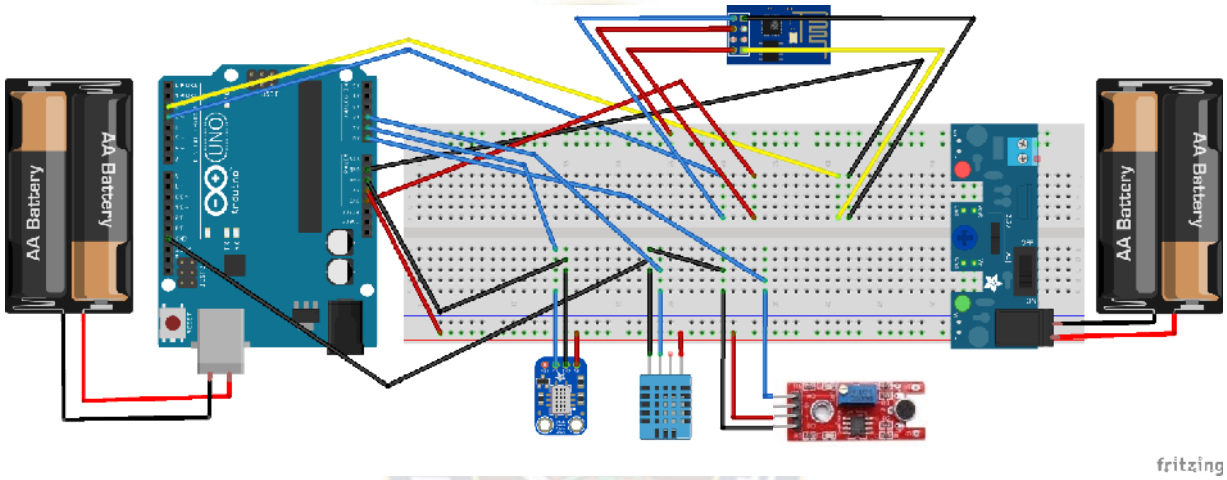

Figura 3.7: Muestra grafica del armado del tercer circuito

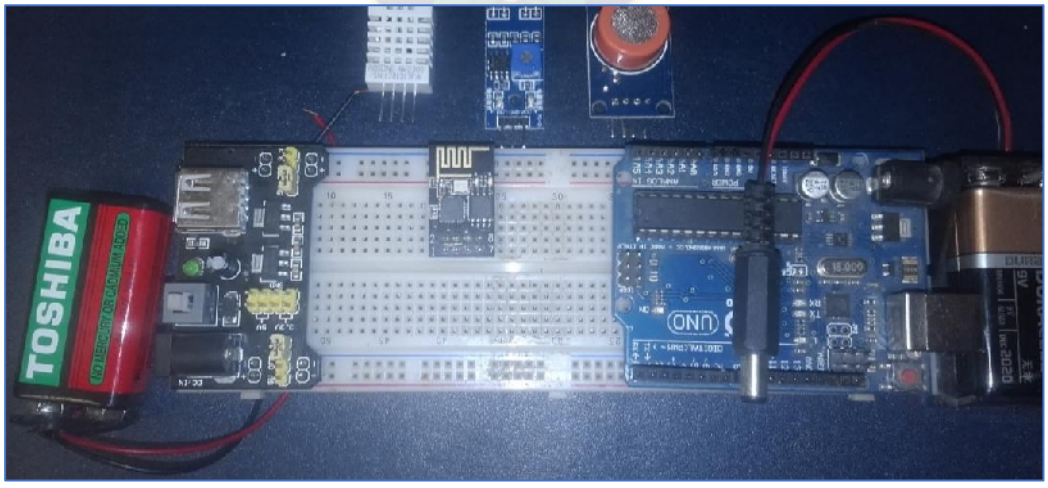

Figura 3.8: Vista superior del armado del tercer circuito

d) Cuarto Circuito

Finalmente, este circuito corresponde a la unión de la placa arduino UNO, leds y el módulo esp8266. Utilizando los pines 9,10 y 11 del arduino para hacer la conexión con los catodos correspondientes a los leds rojo, amarillo y verde respectivamente y finalmente la conexión al módulo esp8266 ya mencionado.

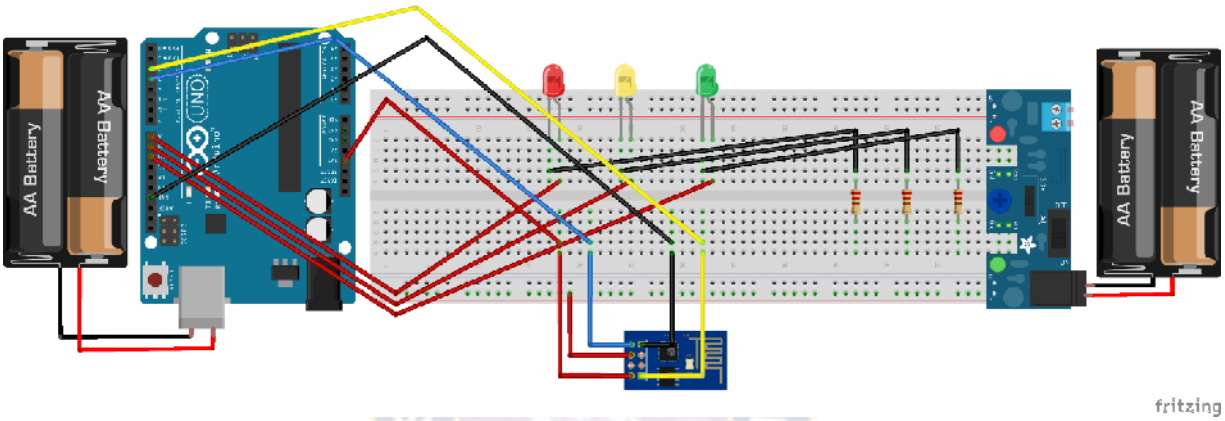

Figura 3.9: Muestra grafica del armado del cuarto circuito

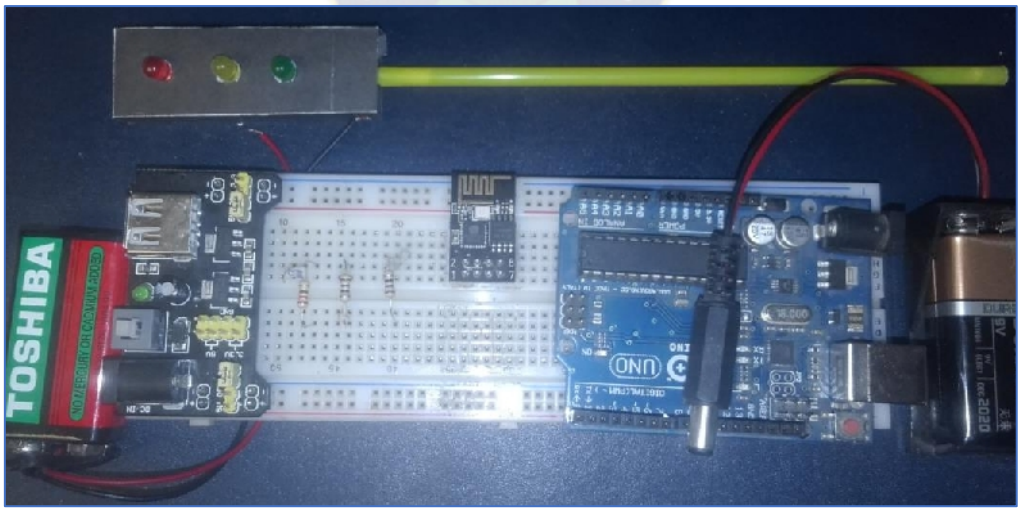

Figura 3.10: Vista superior del armado del cuarto circuito

3.2.2. Programación

Como se mencionó cada nodo contiene cuatro Arduinos con una funcionalidad distinta, es por ello que los programas correspondientes a cada uno se describirán de forma separada para una mejor comprensión, mencionando que cada programa contendrá una función en común "connectWiFi( )" la cual está ubicada dentro de la función "void setup( )" y se encarga primordialmente en que el Arduino se conecte correctamente a la red a través del módulo esp8266.

a) Primer Programa

Enviara las lecturas que realice el módulo RFID ubicado en la entrada del flujo vehicular junto con la altura del vehículo registrado con el sensor ultrasónico ubicado en la misma posición que el módulo RFID para ello se utilizara el servicio REST 1 de la tabla 3.1.

El programa correspondiente es el fichero **"entradaSemaforo.ino"** en el cual se pueden observar las siguientes funciones:

- **void setup( )**: En esta función se inicializan todas las variables y pines necesarios del Arduino, se definen las comunicaciones serie utilizadas y se conecta a la red llamando a la función 'connectWiFi( )'

- **void loop( )**: En esta función se está escuchando constantemente si existe lectura del módulo RFID, en caso afirmativo se toman los datos del TAG leído junto con los datos encriptados en él y el valor actual del sensor ultrasónico para enviar una cadena en el servicio REST

- **void calibración( )**: Esta encargada de la calibración del sensor ultrasónico antes del envió de datos

b) Segundo Programa

Enviara las lecturas que realice el módulo RFID ubicado en la salida del flujo vehicular junto con el valor del estado actual del semáforo correspondiente a ese flujo para ello se utilizara el servicio REST 2 mencionado en la tabla 3.1.

El programa correspondiente al arduino es el fichero **"salidaSemaforo.ino"** en el cual se pueden observar las siguientes funciones:

- **void setup( )**: En esta función se inicializan todas las variables y pines necesarios del Arduino, se definen las comunicaciones serie utilizadas y se conecta a la red llamando a la función 'connectWiFi( )'

- **void loop( )**: En esta función se está escuchando constantemente si existe lectura del módulo RFID, en caso afirmativo se toman los datos del TAG leído junto con el valor actual de la entrada digital correspondiente al semáforo para enviar una cadena en el servicio REST :

c) Tercer Programa

Enviara los datos que los sensores de humo, sonido y temperatura registren cada 5 segundos, ubicados en el flujo vehicular al servidor implementado con NodeJS utilizando en servicio REST 3 mencionado en la tabla 3.1.

El programa correspondiente al tercer arduino es el fichero **"sensores.ino"** en el cual se pueden observar las siguientes funciones:

- **void setup( )**: En esta función se inicializan todas las variables y pines necesarios del Arduino, se definen las comunicaciones serie utilizadas y se conecta a la red llamando a la función 'connectWiFi()'

- **void loop( )**: En esta función se envía los datos de lectura de los sensores cada 5 segundos mediante el servicio REST mencionado

d) Cuarto Programa

Finalmente, este programa se encargará de solicitar los tiempos asignados a cada luz del semáforo correspondiente a ese flujo vehicular para ello utilizará el servicio REST 4 mencionado en la tabla 3.1.

El programa correspondiente al cuarto arduino es el fichero **"semaforo.ino"** en el cual se pueden observar las siguientes funciones:

- **void setup( )**: En esta función se inicializan todas las variables y pines necesarios del Arduino, se definen las comunicaciones serie utilizadas y se conecta a la red llamando a la función 'connectWiFi()'

- **void loop( )**: En esta función se envía los datos de lectura de los sensores cada 5 segundos mediante el servicio REST mencionado

- **int obtenerTiempo(String color)**: Esta función realiza una búsqueda de la cadena color en la respuesta del servicio REST donde devuelve el tiempo de encendido del led correspondiente al color

3.3. ALMACENADO DE DATOS RFID

El almacenado de datos en los lectores RFID se realizó mediante la escritura con el módulo RFID-RC522 RF mencionado en el capítulo anterior, los tags soportan 16 bits de almacenamiento por lo cual se decidió grabar la información más relevante sobre el vehículo.

Se tomó en cuenta la placa del vehiculó, tipo de vehículo y el número de carnet del propietario para en base a estos datos tener un manejo más rápido de consulta con la base de datos

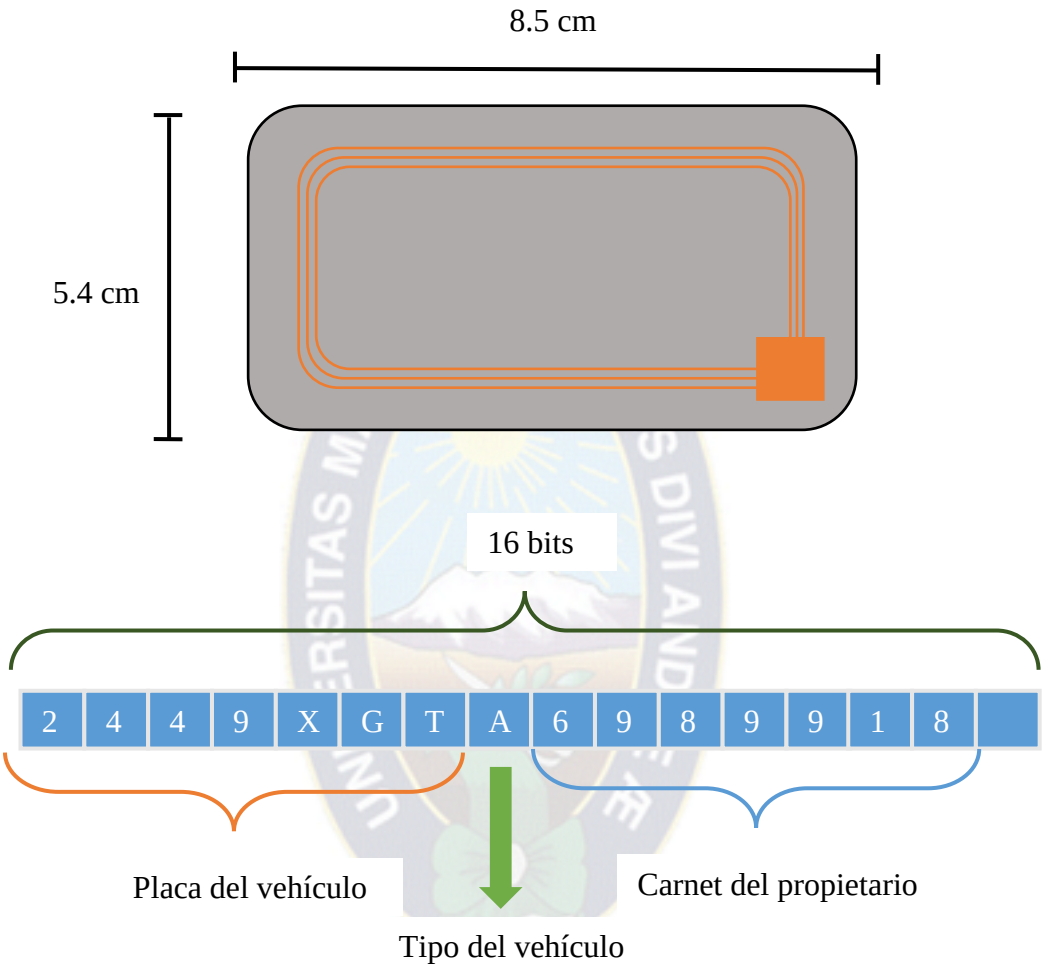

Figura 3.11: Almacenamiento de datos en el tag RFID

Por último, la implementación del sistema web nos permite interactuar con todos los circuitos electrónicos mencionados de una forma muy sencilla y fácil ya sea desde la computadora o una aplicación.

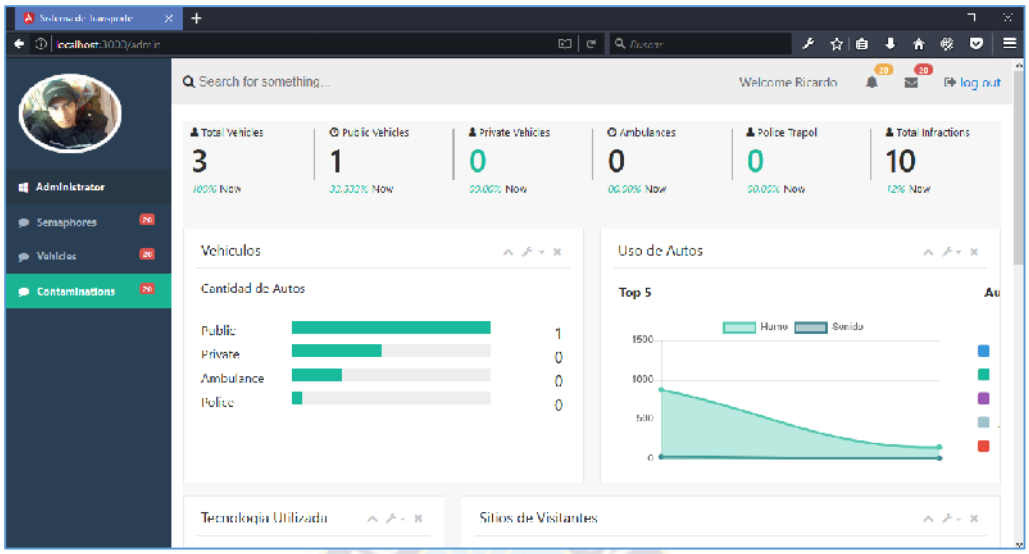

Figura 3.12: Vista de la interfaz gráfica del sistema web

3.4. MODELO FISICO

Para la representación del prototipo se realizó una maqueta en el cual se toman la unión de todos los circuitos por nodo, como se mencionó anteriormente solo se implementará uno de los nodos para demostrar su funcionalidad ya que los demás cumplen la misma función en este caso es el nodo 8 de la figura 3.12. Se organizaron los nodos por grupos para tener un manejo ordenado ya que estos son los que afectan directamente en el flujo vehicular en ese punto:

Grupo 1:

Nodo 8 contiene Flujo 1, Semáforo 1
Nodo 2 contiene Flujo 2, Semáforo 2
Nodo 4 contiene Flujo 3, Semáforo 3
Nodo 6 contiene Flujo 4, Semáforo 4

Grupo 2:

Nodo 13 contiene Flujo 1, Semáforo 5
Nodo 9 contiene Flujo 2, Semáforo 6
Nodo 7 contiene Flujo 3, Semáforo 7
Nodo 11 contiene Flujo 4, Semáforo 8

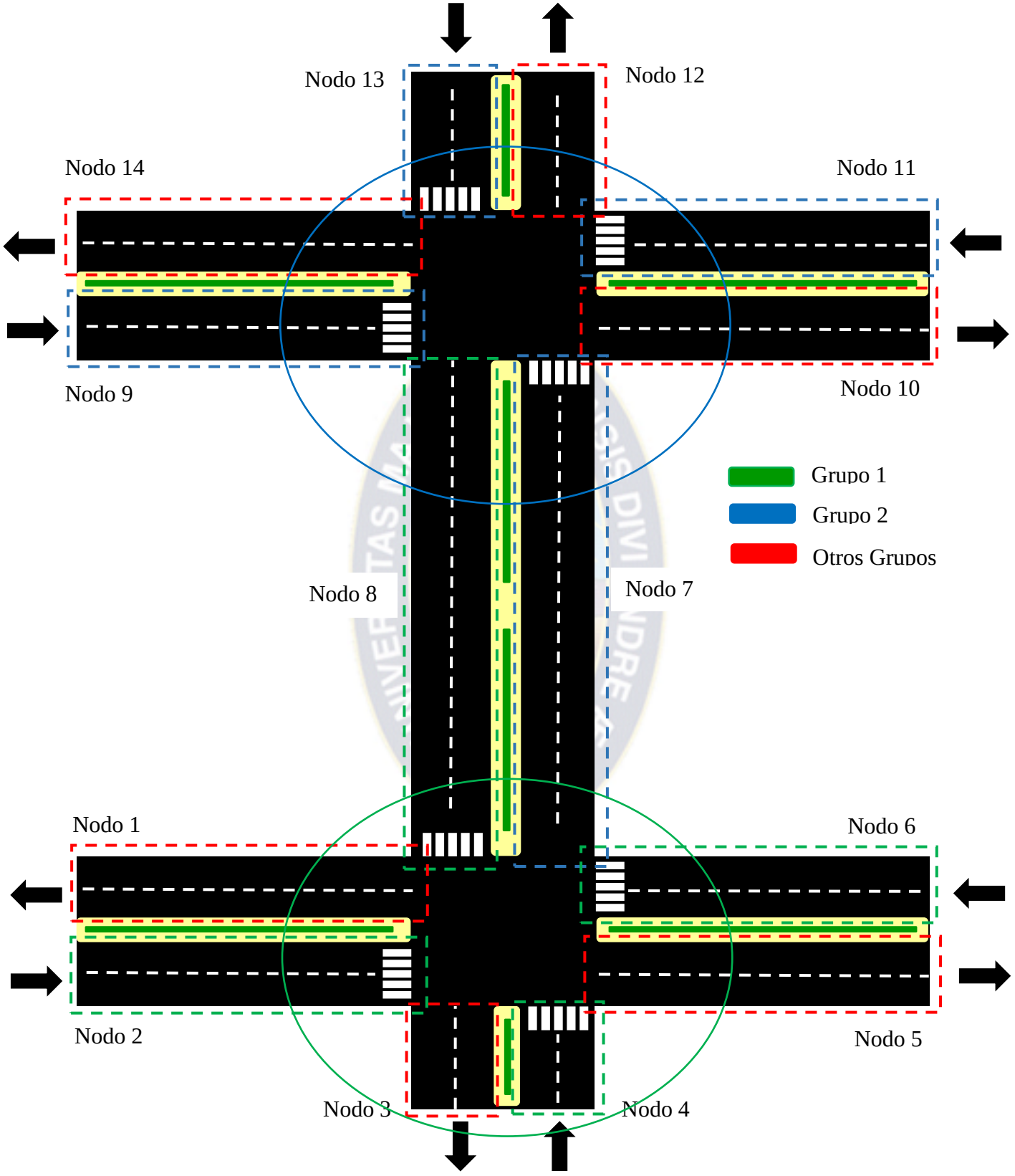

Figura 3.13: Representación de ubicación de los nodos

Los cableados de los cuatro circuitos mencionados para el nodo 8 quedan totalmente ocultos para mostrarlo de una forma más entendible

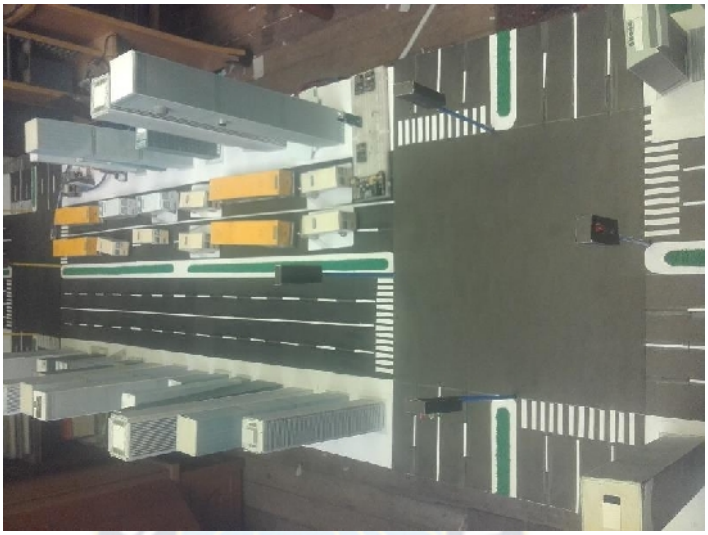

Figura 3.14: Vista superior de los sensores y actuadores ubicados en la maqueta

Figura 3.15: Vista lateral de maqueta

CAPITULO IV

VALIDACIÓN Y EVALUACIÓN

Para representar el tráfico vehicular del prototipo se optó por el software VISSIM, el cual incorpora una gran cantidad de elementos como detectores, semáforos, reguladores, etc. Esto permite aplicar sofisticados métodos de control como el que ofrece la tesis, son capaces de modificar, en tiempo real, las temporizaciones actuales de los cruces en función de las características actuales del tráfico.

Dicho esto, cabe mencionar que la tesis no abarca todas las funcionalidades que este software representa, por este motivo solo se pretende representar y tomar en cuenta los datos respectivos al flujo vehicular que el software devuelva y en base a estos realizar comparaciones con las funcionalidades que el prototipo ofrece.

Se eligió configurar con la plataforma de simulación Vissim, por ser este, dentro de la gama de modelos microscópicos, un modelo estocástico (Las leyes conocidas de causa-efecto están descritas por distribuciones de probabilidades, procesos al azar con indeterminancias en su evolución futura, es sabida la condición actual, pero al final del proceso son muchas las posibilidades, sin embargo, algunas trayectorias son más probables y otras menos). En contraposición con el modelo macroscópico del software Transyt 7F, desarrollado en Estados Unidos, que es determinístico (La relación causa-efecto se conoce en su totalidad, por ejemplo, representa todos los fenómenos que siguen las leyes de la física clásica).

4.1. LOCALIZACIÓN

Al realizar un diagnóstico de las condiciones de movilidad en la ciudad de La Paz-Bolivia se identificaron dos puntos muy importantes, los cuales generan flujos vehiculares determinantes estos son:

- La avenida Mariscal Santa Cruz ubicado en la ciudad de La Paz se eligió el tramo entre las calles Oruro y Colon para evaluar el escenario base a la luz de un modelo microscópico en el software VISSIM, ya que este tramo genera un flujo vehicular importante.

- La avenida 6 de Marzo ubicado en la ciudad de El Alto, se eligió el tramo entre las calles cuatro y cinco, ya que este tramo genera un flujo vehicular determinante y caótico.

Por consiguiente, cada punto está conformado por dos intersecciones los cuales están operados mediante dispositivos semafóricos.

Figura 4.1: Ubicación de la calle Oruro y Colon en la representación grafica

4.2. MODELO GRAFICO REPRESENTATIVO

Los datos estadísticos de cada punto se obtuvieron mediante consultas a la Alcaldía de La Paz y de El Alto respectivamente en horas pico por punto para posteriores comparaciones, se debe mencionar que solo se representó el punto ubicado en la Ciudad de La Paz con el software VISSIM, tomando como referencia los datos del punto de la ciudad de El Alto.

Tabla 4.1: Datos físicos de las avenidas en VISSIM

Nº	NOMBRE	TIPO DE COMPORTAMIENTO	ANCHURA	LONGITUD
1	Mariscal Santa Cruz carril 1	50: 50 Km/h	3.50 m	216.738 m
2	Mariscal Santa Cruz carril 2	50: 50 Km/h	3.50 m	215.404 m
3	Calle Oruro carril 1	50: 50 Km/h	3.50 m	89.877 m
4	Calle Oruro carril 2	50: 50 Km/h	3.50 m	89.879 m
5	Calle Colon carril 1	50: 50 Km/h	3.50 m	80.157 m
6	Calle Colon carril 2	50: 50 Km/h	3.50 m	79.562 m

Tabla 4.2: Composición de vehículos para los flujos en VISSIM

COMPOSICION PERSONALIZADA			
Nº	TIPO DE VEHICULO	DESSPEEDDISTR	FLUJO REAL
1	100: Auto Particular	50: 50 Km/h	0.480
2	200: Vehículo Pesado	50: 50 Km/h	0.010
3	300: Bus	50: 50 Km/h	0.050
4	400: Ambulancia	50: 50 Km/h	0.050
5	500: Policía	50: 50 Km/h	0.050
6	600: Minibús	50: 50 Km/h	0.890

Tabla 4.3: Datos de entrada por avenida en VISSIM

Nº	NOMBRE	VOLUME	COMPOSICION VEHICULOS
1	Mariscal Santa Cruz carril 1	5000	Composición personalizada
2	Mariscal Santa Cruz carril 2	5000	Composición personalizada
3	Calle Oruro carril 1	5000	Composición personalizada
4	Calle Oruro carril 2	5000	Composición personalizada
5	Calle Colon carril 1	5000	Composición personalizada
6	Calle Colon carril 2	5000	Composición personalizada

Como se observa la simulación obtiene datos generales por tipo de vehículo, pero el prototipo nos ofrece datos por tipo de vehículo en tiempo real, pero por nodo como se muestra a continuación

Tabla 4.4: Datos por tipo de vehículo obtenidos del nodo 8 en un ciclo semafórico

N.º	TIPO VEHICULO	SUMA PRIORIDAD	CANTIDAD	CONT. ACUSTICA	CONT. AMBIENTAL	TEMP
1	Auto Particular	2	2	**20%**	**30%**	**15º**
2	Vehículo Pesado	1	1			
3	Bus	6	3			
4	Ambulancia	6	1			
5	Policía	5	1			
6	Minibús	10	5			

A continuación, se desglosa la toma de datos del primer arduino ubicado en la entrada del flujo vehicular del nodo 8

Tabla 4.5: Datos del primer arduino del nodo 8 en un ciclo semafórico

N.º	TIPO DE VEHICULO	PRIORIDAD	PLACA	RFID	ALTURA	HORA	FECHA
1	Auto Particular	1	ABC1234	123456	3.5 cm	12:00	05/06/2017
2	Auto Particular	1	DEF1234	724456	3.5 cm	12:00	05/06/2017
3	Vehículo Pesado	1	GHI1234	453956	4 cm	12:01	05/06/2017
4	Bus	2	JKL1234	522456	4 cm	12:02	05/06/2017
5	Bus	2	LNO1234	123456	4 cm	12:02	05/06/2017
6	Bus	2	PQR1234	823456	4 cm	12:04	05/06/2017
7	Ambulancia	6	STU1234	128256	4.3 cm	12:04	05/06/2017
8	Policía	5	VWX1234	323456	3.6 cm	12:05	05/06/2017
9	Minibús	2	YZA1234	127458	3.8 cm	12:05	05/06/2017
10	Minibús	2	DCB1234	866456	3.8 cm	12:07	05/06/2017
11	Minibús	2	ERT1234	743456	3.8 cm	12:08	05/06/2017
12	Minibús	2	ERT1234	123456	3.8 cm	12:09	05/06/2017
13	Minibús	2	ERT1234	993456	3.8 cm	12:10	05/06/2017

Continuando, se desglosa la toma de datos del segundo arduino ubicado en la salida del flujo vehicular del nodo 8

Tabla 4.6: Datos del segundo arduino del nodo 8 terminado un ciclo semafórico

N.º	TIPO DE VEHICULO	PRIORIDAD	PLACA	RFID	HORA	FECHA	ESTADO SEMAFORO
1	Auto Particular	1	ABC1234	123056	12:11	05/06/2017	Rojo
2	Auto Particular	1	DEF1234	724456	12:11	05/06/2017	Rojo
3	Vehículo Pesado	1	GHI1234	453956	12:12	05/06/2017	Rojo
4	Bus	2	JKL1234	522456	12:12	05/06/2017	Amarillo
5	Bus	2	LNO1234	123450	12:13	05/06/2017	Verde
6	Bus	2	PQR1234	823456	12:13	05/06/2017	Verde
7	Ambulancia	6	STU1234	128256	12:13	05/06/2017	Verde
8	Policía	5	VWX1234	323456	12:14	05/06/2017	Verde
9	Minibús	2	YZA1234	127458	12:14	05/06/2017	Verde
10	Minibús	2	DCB1234	866456	12:15	05/06/2017	Verde
11	Minibús	2	ERT1234	743456	12:16	05/06/2017	Verde
12	Minibús	2	ERT1234	123456	12:17	05/06/2017	Verde
13	Minibús	2	ERT1234	993456	12:17	05/06/2017	Verde

Como se observa el prototipo ofrece datos muy detallados del flujo vehicular en comparación con el software VISSIM, lo que beneficia en mayor importancia para consecuentes estudios para el problema de la congestión vehicular.

Tabla 4.7: Datos detallados en un ciclo semafórico del nodo 8

N.º	TIPO DE VEHICULO	PLACA	RFID	HORA ENTRADA	HORA SALIDA	TIEMPO ESPERA	FECHA
1	Auto Particular	ABC1234	123056	12:00	12:11	11 min	05/06/2017
2	Auto Particular	DEF1234	724456	12:00	12:11	11 min	05/06/2017
3	Vehículo Pesado	GHI1234	453956	12:01	12:12	11 min	05/06/2017
4	Bus	JKL1234	522456	12:02	12:12	10 min	05/06/2017
5	Bus	LNO1234	123450	12:02	12:13	11 min	05/06/2017
6	Bus	PQR1234	823456	12:04	12:13	9 min	05/06/2017
7	Ambulancia	STU1234	128256	12:04	12:13	9 min	05/06/2017
8	Policía	VWX1234	323456	12:05	12:14	9 min	05/06/2017
9	Minibús	YZA1234	127458	12:05	12:14	9 min	05/06/2017
10	Minibús	DCB1234	866456	12:07	12:15	8 min	05/06/2017
11	Minibús	ERT1234	743456	12:08	12:16	8 min	05/06/2017
12	Minibús	ERT1234	123456	12:09	12:17	8 min	05/06/2017
13	Minibús	ERT1234	993456	12:10	12:17	7 min	05/06/2017

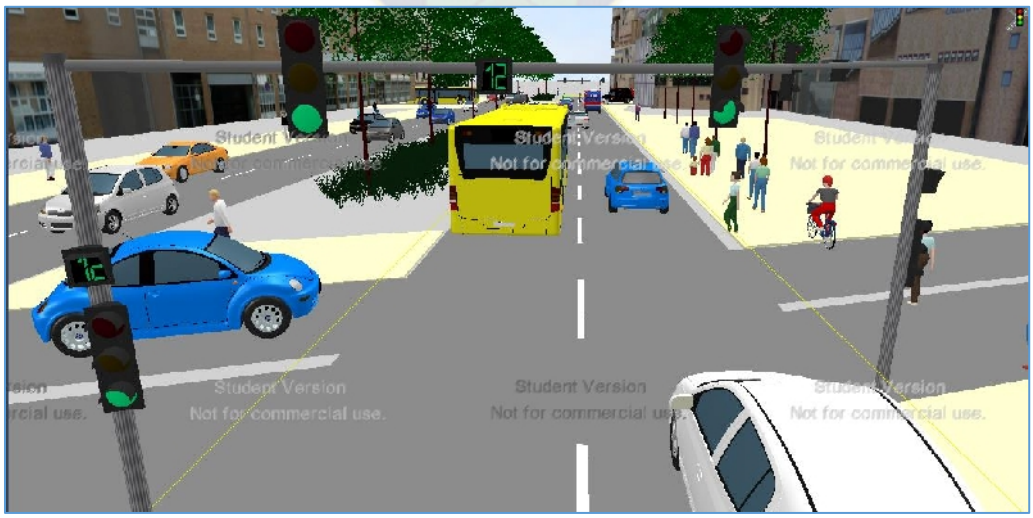

Figura 4.2: Vista 3d en VISSIM

CAPITULO V

CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES GENERALES

La implementación de un prototipo funcional, modular y fácilmente ampliable se logró ya que la mayoría de los sensores y actuadores tienen un funcionamiento muy similar. Las posibilidades que nos brindan los servicios web REST son muy grandes, y gracias a la sencillez de las herramientas utilizadas se pueden adaptar estos servicios sin una elevada complicación a distintos proyectos.

Desarrollar con Arduino ha sido muy gratificante al ver como poco a poco se conseguía mejoras con los circuitos, los sensores y las posibilidades que nos brindan, tanto es así que adquirí un kit de sensores y actuadores (kit 37 in 1) para continuar probando con otros modelos. Por el contrario, hubo bastantes problemas utilizando el módulo esp8266 dado la escasez de documentación oficial y ciertos problemas que según parece deberían solucionarse al utilizar otro firmware, aunque gracias a la comunidad de usuarios de este módulo se consiguió solucionar la mayoría.

Los resultados devueltos del prototipo fueron mucho más detallados por nodo al 95%, del flujo vehicular en comparación del software VISSIM que devuelve datos más generales y menos detallados como ser cantidad de vehículos, tipos de vehículos, etc.

5.2. RECOMENDACIONES

- Mejorar la conexión de los objetos a la web mediante el uso de sockets para no tener segundos de latencia en el momento de hacer las peticiones

- El uso de Raspberry py cuando el sistema sea más complejo y se den más funcionalidades a los objetos conectados a la web

- Mejorar la forma de asignar las direcciones IP y los nombres a los nodos

- Probar otras versiones del firmware del módulo esp8266

- Con una batería recargable, el nodo podría ser completamente inalámbrico

- Mejorar la implementación del circuito para dar un aspecto más profesional e incluso se podría realizar una carcasa para contener al nodo con todos sus componentes

- Implementar dirección IP v6 ya que las direcciones IP v4 están obsoletas para el internet de las cosas

REFERENCIAS BIBLIOGRAFÍCAS

FERNANDO REYES (2001). Aplicaciones en robótica, mecatrónica e ingenierías, Editorial: Estribor

ERIK SAVASGARD (2000). Arduino 101 beginners guide, Editorial: Estribor

SIMON MONK (2003). 12 proyectos arduino + android, Editorial: Estribor

ÓSCAR TORRENTE ARTERO (2005). Arduino curso práctico de formación, Editorial: Estribor

MASSIMO BANZI ANAYA (2001). Introducción a arduino, Editorial: Estribor

SERGIO I. MELDRICK Y JOSE MIGUEL BARRAZA (2001). Internet de las cosas web 3.0 y la revolución móvil, Editorial: Estribor

MARCO SCHWARTZ (2001). Internet of things with ESP8266, Editorial: Packt

CHARALAMPOS DOUKAS (2009). Building internet of things with the arduino, Editorial: Morris

INSTITUTO DE LA CONSTRUCCION Y GERENCIA (1995). Ingeniería de tránsito y seguridad vial, Editorial: ICG

VAN VALKEMBURGH (2000). Electrónica básica, Editorial: Bell

ADAM JUNIPER (2001). La guía completa de drones, Editorial: Bell

AGUS KURNIAWAN (2010). NodeMCU development workshop, Editorial: Wan

CARLOS AZAUSTRE (2000). Desarrollo web ágil con angularJS, Editorial: Mobi

LENGUAJE ARDUINO. Fecha consulta marzo 2016,

Sitio web: https://www.arduino.cc/

COMO USAR FLASK. Fecha consulta marzo 2016,

Sitio web: http://flask.pocoo.org/

QUE ES FRITZING. Fecha consulta marzo 2016,

Sitio web: http://fritzing.org

LO MEJOR THINGSPEAK. Fecha consulta marzo 2016,

Sitio web: https://thingspeak.com/

TUTORIALES ARDUINO. Fecha consulta abril 2016,

Sitio web: http://www.prometec.net/

INTERNET DE LAS COSAS. Fecha consulta junio 2016,

Sitio web: http://www.internetdelascosas.cl

ESP8266 COMMUNITY FORUM. Fecha consulta Julio 2016,

Sitio web: http://www.esp8266.com/

MANUAL PROGRAMACIÓN ARDUINO. Fecha consulta junio 2016,

Sitio web: http://arduinobot.pbworks.com/f/Manual+Programacion+Arduino.pdf

FLASK WEB DEVELOPMENT. Fecha consulta junio 2016,

Sitio web: http://blog.miguelgrinberg.com/

INTRODUCCIÓN ARDUINO. Fecha consulta julio 2016,

Sitio web: http://recursos.cepindalo.es/course/view.php?id=35

ANÁLISIS COMPARATIVO DE LAS PLACAS ARDUINO. Fecha consulta mayo 2016,

Sitio web:http://comohacer.eu/analisis-comparativo-placas-arduino-compatibles/

ALSELECTRO ESP8266. Fecha consulta agosto 2016,

Sitio web: https://alselectro.wordpress.com/

ARDUINO, ESP8266. Fecha consulta agosto 2016,

Sitio web: http://electronut.in

CHILDOFCODE. Fecha consulta agosto 2016,

Sitio web: http://childofcode.com/

ARDUINO CON ESP8266. Fecha consulta julio 2016,

Sitio web: http://www.jopapa.me/

ELECTRÓNICA, ARDUINO Y ESP8266. fecha Consulta agosto 2016

Sitio web: http://polaridad.es/

DETECTOR DE LLAMA. Fecha consulta abril 2016,

Sitio web: https://prometec.com/sensor_humo

PRIMEROS PASOS CON RFID. Fecha consulta marzo 2016,

Sitio web: https://prometec.com/arduino/rfid_arduino

RFID. Fecha consulta abril 2016,

Sitio web: https://wikipedia.com/rfid

SENSOR DE DISTANCIA. Fecha consulta marzo 2016,

Sitio web: https://prometec.com/arduino/sensor_distancia

SENSOR DE SONIDO ARDUINO. Fecha consulta marzo 2016,

Sitio web: https://prometec.com/arduino/sensor_sonido

TAGS RFID ARDUINO. Fecha consulta abril 2016,

Sitio web: https://prometec.com/arduino/tags_rfid

ANEXOS

ANEXO A GLOSARIO

ASP	Active Ser Active Server Pages = Activación de Páginas de Servicio.
CRM	Customer Relationship Mangagement = Gestión de relación con los clientes.
ERP	Enterprise Enterprise Resource Planning = Planificación de Recursos Empresariales.
IaaS	Infrastructure as a Service = Infraestructura como Servicio.
IIS	Internet Información de Servicios.
PaaS	Platform as a Service = Plataforma como servicio.
Pyme	Pequeña y mediana empresa.
SaaS	Software as a Service = Software como servicio.
TIC	Tecnología de información y comunicación.
IEEE	Institute of Electrical and Electronics Engineers = Instituto de Ingenieros Eléctricos y Electrónicos.
IOT	Internet of things = Internet de las cosas

ANEXO B ESTRUCTURA LENGUAJE ARDUINO

Estructuras de control

- if (comparador si-entonces)
- if...else (comparador si...sino
- for (bucle con contador)
- switch case (comparador múltiple)
- while (bucle por comparación booleana)
- do... while (bucle por comparación booleana)
- break (salida de bloque de código)
- continue (continuación en bloque de código)
- return (devuelve valor a programa)

Sintaxis

- ; (punto y coma)
- {} (llaves)
- // (comentarios en una línea)
- /* */ (comentarios en múltiples líneas)

Operadores Aritméticos

- = (asignación)
- + (suma)
- - (resta)
- * (multiplicación)
- / (división)
- % (modulo)

Operadores Comparativos

- == (igual a)
- != (distinto de)
- < (menor que)

- > (mayor que)
- <= (menor o igual que)
- >= (mayor o igual que)

Operadores Booleanos

- && (y)
- || (o)
- ! (negación)

Operadores compuestos

- ++ (incremento)
- -- (decremento)
- += (suma compuesta)
- -= (resta compuesta)
- *= (multiplicación compuesta)
- /= (división compuesta)

Constantes

- HIGH | LOW
- INPUT | OUTPUT
- true | false
- Constantes Numéricas

Tipos de Datos

- boolean (1,0,true,false)
- char (carácter ej. ‘a’ o de -128 a 127)
- byte (de 0 a 255)
- int (entero de -32768 a 32767)
- unsigned int (entero sin signo de 0 a 65535)
- long (entero 32b, de -2147483648 a 2147483647)
- unsigned long (entero 32b sin signo de 0 a 4294967295)

- float (en coma flotante de -3.4028235E +38 a 3.4028235E +38)
- double (en coma flotante de 32b de -3.4028235E +38 a 3.4028235E +38)
- string (cadena de caracteres)
- array (cadena)
- void (vacío)

E/S Digitales

- pinMode(pin,[INPUT, OUTPUT])
- digitalWrite(pin, valor)
- digitalRead(pin)

E/S Analógicas

- analogRead(pin)
- analogWrite(pin, valor) - PWM (modulación por ancho de pulso)

E/S Avanzadas

- tone(pin, freqhz)
- tone(pin, freqhz, duración_ms)
- noTone(pin)
- shiftOut(pinDatos, pinReloj,[MSBFIRST,LSBFIRST], valor)
- pulseIn(pin, [HIGH, LOW])

Tiempo

- millis() - Se desborda el contador en 50 dias
- micros() – Se desborda el contador en 70 minutos
- delay(ms)
- delayMicroseconds(us)

Matemáticas

- min(x,y) (mínimo)
- max(x,y) (máximo)

- abs(x) (valor absoluto)
- constrain(x, valMin, valMax) (limita)
- map(val, deBAJO, deALTO, aBAJO, aALTO) (cambia valor de rango)
- pow(base, exponente) (eleva a un número)
- sq(x) (eleva al cuadrado)
- sqrt(x) (raíz cuadrada)

Trigonometría

- sin(x) (seno)
- cos(x) (coseno)
- tan(x) (tangente)

Números Aleatorios

- randomSeed(semilla) – long o int
- random(max)
- random(min, max)

Comunicación Serial

- begin(velocidad) – abre puerto serie y establece velocidad
- end() – desactiva puerto serie
- available() – rebela si hay datos en el Puerto
- read() – lee un único carácter del buffer
- flush() – vacía el buffer
- print(datos) - imprime los datos en el puerto serie
- println(datos) - imprime los datos en el puerto serie con retorno de carro
- write(bytes) – escribe caracteres en el puerto serie

ANEXO C **COMANDOS AT**

Los comandos utilizables con la placa Esp8266 son:

ORDEN AT	DESCRIPCIÓN
AT	Verifica si funciona devolviendo ‘OK’
AT+GMA	Información firmware
AT+RST	Reinicia el módulo
ATE1 , ATE0	Activa o desactiva el eco de los comando AT
AT+GSLP	Modo reposo
AT+CWMODE	Modo de funcionamiento. (1)estación, (2)punto Acceso,
AT+CIOBAUD	Velocidad comunicación en baudios
AT+CSYSWDTENABLE	Activa Watchdog
AT+CSYSWDTDISABLE	Desactiva Watchdog
AT+CIPSTAMAC	Establecer o consultar MAC del Punto de acceso
AT+CIPAPAMAC	Establecer o consultar MAC de la estación
AT+CWLAP	Consultar redes disponibles y establecer conexión
AT+CWQAP	Desconectar punto de acceso
AT+CWSAP	Configura punto de acceso
AT+CWDHCP	Activa/Desactiva DHCP en el punto de acceso
AT+CIPSTA	Establece IP de la estación
AT+CIPAP	Establece IP del punto de acceso
AT+CWLIF	IPs conectadas al punto de acceso
AT+CIFSR	Ip actual del módulo
AT+CIPMUX	Modo de conexión (0) simple, (1) múltiple
AT+CIPSERVER	Activa/Desactiva el modo servidor
AT+CIPSTO	Configura timeout
AT+CIPSTART	Inicia una conexión
AT+CIPSTATUS	Muestra estado de una conexión
AT+CIPCLOSE	Cierra una conexión
AT+CIPSEND	Prepara el envío de datos
AT+CIPUPDATE	Se usa para actualizar el módulo