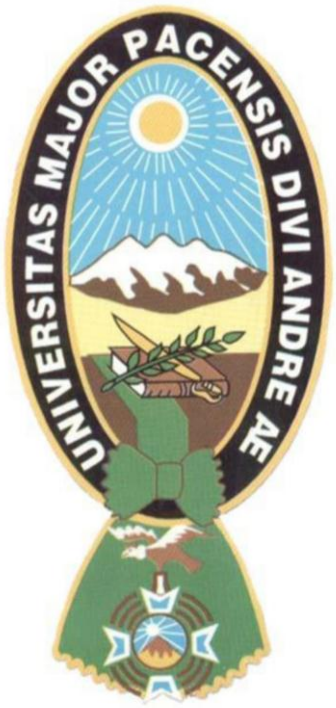
UNIVERSIDAD MAYOR DE SAN ANDRES
FACULTAD DE TECNOLOGIA
CARRERA ELECTRONICA Y TELECOMUNICACIONES



“Diseño de un prototipo de control y seguridad, a través una red inalámbrica para la sala de servidores del Ministerio de Obras Públicas, Servicios y Vivienda”

Proyecto de Grado presentado para obtener el Grado de Licenciatura

Por: Andrés Richard Mamani Pérez

Tutor: M.Sc. Edwin Jesus Alave Alavi

La Paz - Bolivia
Gestión 2018

UNIVERSIDAD MAYOR DE SAN ANDRES

FACULTAD DE TECNOLOGIA

CARRERA ELECTRONICA Y TELECOMUNICACIONES

Proyecto de Grado:

"Diseño de un prototipo de control y seguridad, a través una red inalámbrica para la sala de servidores del Ministerio de Obras Públicas, Servicios y Vivienda"

Presentado por: Andrés Richard Mamani Pérez

Para optar el grado académico de Licenciado en Electrónica y Telecomunicaciones.

Nota Numeral ……..

Nota Literal ……..

Ha sido ……..

M.Sc. Luis Richard Marquez Gonzales

Director de la Carrera de Electrónica y Telecomunicaciones

Tutor: M.Sc. Edwin Jesus Alave Alavi

Tribunal: M.Sc. Ing. Fabian Amado Tito Luque

Tribunal: Lic. Julia Torrez Soria

DEDICATORIA

Dedico este trabajo a mis padres Macario Villa Mamani Laura y Cristina Asunta Pérez Flores por su incansable labor en apoyarme y alentarme a culminar mis estudios, y a mi querida Facultad donde adquirí el conocimiento necesario para el desarrollo de nuestra vida.

AGREDICIMIENTO

Quiero agradecer a los docentes de Carrera, quienes me introdujeron en el apasionante campo de la Electrónica y las Telecomunicaciones; a mi tutor de proyecto, Lic. Edwin Alave A. quien apoyó el desarrollo de este trabajo, dedicando su valioso tiempo supervisando en la elaboración del mismo; al Ministerio de Obras Públicas, Servicios y Vivienda por darme la oportunidad de realizar el presente proyecto en sus ambientes.

RESUMEN

El objetivo del proyecto consiste en diseñar e implementar una solución a partir de un sistema automatizado aplicable a la sala de servidores del Ministerio de Obras Publicas Servicio y Viviendas.
El reto ha consistido en crear un sistema que, basándose en la nueva tecnología de programación y de hardware pueda controlar y supervisar los equipos que se encuentran en dicha área.
El proyecto se divide en dos secciones: En una primera, se ponen en claro las necesidades que el sistema de automatización puede satisfacer. Además, se estudia el mercado de este tipo de sistema y se especifica el tipo de sistema en el que se va a centrar el proyecto.
Posteriormente y como resultado de la fase inicial se implementa una solución técnica que complemente con las conclusiones obtenidas en el estudio inicial.
Ésta sección consiste en el diseño puramente conceptual del sistema, teniendo claras las especificaciones que se debe cumplir para tomar las decisiones oportunas en la elección del protocolo, los componentes y los servicios que integrará el sistema.
La solución técnica consiste en un servicio, producto de un sistema de alerta temprana que se enfoque en el caso que se presente el estado de la sala de servidores con sus distintos dispositivos y equipos.
Conjuntamente a las dos secciones principales ha sido precisa la inclusión de distintos anexos para que la solución técnica sea completa.
En los anexos se encuentra un completo "tutorial" de configuración y programación del sistema de automatización y supervisión vía web. Gracias a este tutorial, el usuario podrá configurar cada uno de los dispositivos que integran el sistema, programar el comportamiento de la sala de servidores en función de diferentes horarios o situaciones.

INDICE GENERAL Pág.

INDICE DE FIGURAS

INDICE DE TABLAS

CAPITULO I

INTRODUCCION

1.1 GENERALIDADES

En el Ministerio de Obras Públicas, Servicios y Vivienda hay un área muy vulnerable que se deja de lado, esta es la sala de servidores, los cuales no tienen un sistema de climatización adecuado para que los equipos instalados en el lugar funcionen de manera adecuada. El presente proyecto plantea implementar cámaras de seguridad, sensores (de temperatura, humo y de corriente), candado electrónico con ingreso de personal VIP y tarjetas controladoras (Arduino y Raspberry pi 2).

Es así que se implementara una cámara de seguridad en un punto estratégico de la sala de servidores que permitirá visualizar todo el ambiente y la entrada a la misma, también se hará el colocado de sensores de distintos tipos para que se pueda hacer un testeo exacto de toda la sala para así tener datos e información correcta de la sala y se realice la acción correspondiente, el control total de la sala de servidores lo podremos ver vía web, es decir se realizara una página web para el control de la temperatura que se guardara en una base de datos de un servidor, sobre la seguridad del lugar se debe indicar que también se guardara los videos e imágenes de las personas que ingresan al lugar.

Este sistema no solo servirá para controlar el lugar, sino también para administrar sus servicios, controlar la luz, temperatura, humo, se construirá un sistema de seguridad de ingreso, entre otros. Y así también se pueda realizar un ahorro de energía.

1.2 ANTECEDENTES

La empresa Gigas, multinacional española de Cloudcomputing (Computación en la nube), anuncio la apertura de su primer data center en la región, el cuarto a nivel mundial, ubicado en Santiago de Chile. Hasta ahora, la compañía cuenta con tres

Data Centers (dos en España -Madrid- y uno en EE.UU -Miami-). El centro de datos permite una mejor conectividad y los niveles más óptimos de eficiencia para todos los clientes, especialmente ubicados en el Cono Sur.

La puesta en marcha de este centro de datos es una muestra de la importancia que tiene el mercado latinoamericano para las compañías. Así, según José Antonio Arribas, COO (Chief Operations Office - Jefe de Operaciones) y co-fundador de Gigas, "guiados por nuestra máxima de cercanía al cliente, ubicar el data center en la región responde a una decisión estratégica para nosotros. Podemos acompañar a las compañías latinoamericanas en todos los procesos de su migración a la nube y a todos los niveles: comercial, soporte, facturación y a partir de ahora también con infraestructura local y regional".

Las nuevas datas centers, categorizados como TIer III, se encuentra dentro de una estructura de 1.200m2 y ha sido construido bajo las más altas exigencias de diseño y construcción. Con una inversión en implantación y operación superior al millón de dólares en los próximos dos años, este centro de datos albergará la plataforma cloud que Gigas ofrece a sus clientes, la cual destaca por su total flexibilidad, escalabilidad y accesibilidad.

Asimismo, afirmo el gerente de la empresa Gigas: "este nuevo hito nos convierte en el operador cloud de referencia para toda la Región. Ofrecemos un servicio flexible, en tiempo real, ágil, que ayuda a las empresas a ser más competitivas, también una alta seguridad y administración vía web, con un sistema automatizado, para la climatización del área".

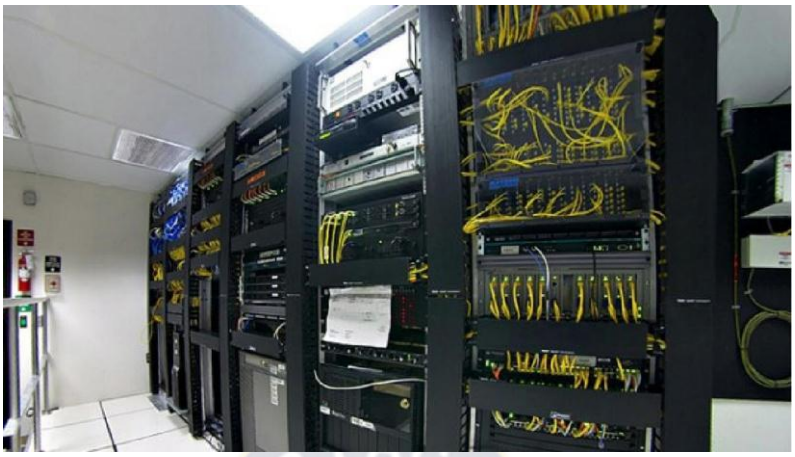


Figura 1 Data Center de la empresa Gigas.

Fuente: https://gigas.com/

1.2.1. Construcción de nuevos y modernos Data Centers en Brasil.

La empresa "DimensiónData" ha añadido cuatro nuevos data center cloud a su portafolio, siendo ahora 11 el número total de data centers que opera. Los centros de datos están en Ashburn (Virginia), Melbourne (Australia), Londres (Reino Unido) y Sao Paulo (Brasil).

Ashburn ya está operativo, Melbourne entro en funcionamiento a finales del mes de diciembre del 2014 en Londres y São Paulo y dio servicio de modernos Data Centers en enero 2015. Estas nuevas funciones se añaden a los data centers modernos que ya ofrecen la Plataforma MCP (Managed Cloud Platform) de DimensiónData en San Jose (California), Amsterdam, Sidney, Johannesburgo, Tokio y Hong Kong, y aquí entra los nuevos servidores de Brasil, que con una gran tecnología en seguridad y administración web, están en la punta de la tecnología. Sao Paulo, primero de los MCP de "*DimensionData*" en América del Sur, proporcionará disponibilidad a las empresas globales que quieran desarrollar su negocio en esta región. El directivo de la empresa Gigas afirmó que las nuevas instalaciones ayudarán también a ofrecer servicios que cumplan con los requisitos regulatorios relacionados con la seguridad y soberanía de los datos.

1.3. PLANTEAMIENTO DEL PROBLEMA

1.3.1. Identificación del Problema

El problema consiste en evitar el gasto de energía innecesario, que también es una fuente de calor para la sala de servidores que interviene en el buen funcionamiento de los equipos que se encuentran en la sala.

Los directos involucrados son los equipos de la sala de servidores, si no tienen una buena climatización, no tendrán un buen funcionamiento, la temperatura deseada para una sala de servidores, es la temperatura ambiente de 29°C. También dentro del grupo de los involucrados estarán los usuarios finales en general, porque es la parte que exigen un buen rendimiento de la red para la navegación de internet y usos de los diferentes sistemas de la institución

Los estudios que se realizaron indican que la buena aclimatación influye de gran manera en una sala de servidores y en el ahorro de energía.

1.3.2. Formulación del Problema

Los problemas que atraviesa la sala de servidores se deben a la falta de control, tanto en términos de seguridad y climatización, actualmente en el área no tiene el clima adecuado para que los dispositivos instalados en el lugar tengan un buen funcionamiento, ya que no hay un sistema que pueda controlar la temperatura del ambiente, otro factor problemático se refiere a la inexistencia de un sistema de alerta temprana, para que en un futuro si habría un desperfecto o corto circuito en la sala de servidores se tenga la información oportuna.

En las condiciones actuales, el personal del área de sistemas no sabría a tiempo las averías que se pueden suceder, mucho menos detectar el posible fallo o desperfecto a nivel hardware, esto debido a la inexistencia del sistema de alerta temprano antes mencionado.

1.4. OBJETIVOS

1.4.1. Objetivo General

Diseñar un prototipo de control y seguridad, a través una red inalámbrica para la sala de servidores del Ministerio de Obras Públicas, Servicios y Vivienda

1.4.2. Objetivos Específicos

- Realizar el estudio estructurado de la sala de servidores para tener un montaje óptimo de los dispositivos (sensor de temperatura, humedad, humo y gas).
- Implementar el hardware para controlar la climatización de la sala de servidores.
- Implementar el hardware para la seguridad del lugar.
- Desarrollar el sistema de alerta temprana.
- Desarrollar una página web para que el personal pueda interactuar con el dispositivo que hay en el área, y tener el control y supervisión del lugar

1.5 JUSTIFICACIÓN

1.5.1. Justificación Técnica

En la actualidad la tecnología va aumentando significativamente, transcurriendo los años también se incrementa la demanda de los usuarios para un mejor servicio. Por esta razón se

debe realizar el diseño de un sistema de climatización y control vía web para poder solucionar los posibles inconvenientes que se hayan suscitado anteriormente y para futuras demandas de servicios.

1.5.2. Justificación Económica

El proyecto manifiesta una determinada importancia que representa el control automatizado en otros países, sin ir muy lejos en nuestro mismo país de nuestra región ya se están implementando las salas de servidores autónomas, eso quiere decir que la sala de servidores tendrán dispositivos que climatice el ambiente para su mejor funcionamiento y sin la intervención de la mano del hombre, lo que representa un avance significativo.

Pero también tendrá la posibilidad de controlar al personal designado, conocer lo que pasa en esa y este control se lo realizara vía web. De esa forma se tendrá un mejor servicio y rendimiento de los equipos que se alojan en la sala de servidores que a la larga representa una reducción de costos para la institución.

1.5.3. Justificación Ambiental

La comunicación se realizará por medio inalámbrico, por esa parte se lograra el ahorro de materiales que en un futuro serán remplazados.

También se puede resaltar el ahorro de energía donde uno de los puntos dentro del diseño del proyecto es la optimización de la energía de la sala de servidores, que actualmente presenta una luminaria siempre encendida, no hay un sistema que regule la energía, todo esto se mejorara a partir del proyecto y su implementación.

1.6 DELIMITACIÓN

1.6.1. Delimitación Temática

El proyecto de grado se enmarcara en el análisis y diseño de la automatización de la sala de servidores, donde uno de los puntos principales a resolver es la falta de climatización y la falta de seguridad del área.

Eliminar todos los problemas que sufre la sala de servidores, para mejorar el funcionamiento de los equipos del lugar que actualmente no logran rendir adecuadamente y por ende los servicios del área de sistemas se ven lentos.

El proyecto se desarrollará en referencia a los fundamentos físicos y estructurales de la red, ofreciendo servicio control y seguridad vía web. Consecuentemente se verá el procedimiento de realización de pruebas en construcción, activación y los parámetros físicos que afectan a la red.

1.6.2. Delimitación Temporal

En correspondencia con los objetivos planteados, se estima que el trabajo de diseño del proyecto se desarrollará en un lapso aproximado de 3 meses.

1.6.3. Delimitación Espacial

El diseño del proyecto se desarrollará en la sala de servidores que está ubicada en el Centro de Comunicaciones La Paz, 5to piso del Ministerio de Obras Públicas, Servicios y Viviendas.

CAPITULO II

FUNDAMENTACIÓN TEÓRICA

2.1 INTRODUCCIÓN

La amplitud de una solución de automatización puede variar desde un único dispositivo, que realiza una sola acción, hasta amplios sistemas que controlan prácticamente todas las instalaciones dentro de un área.

2.2. ESTUDIO DE LA AUTOMATIZACION

2.2.1. Clasificación de dispositivos

2.2.1.1. Controlador: Los controladores son los dispositivos que gestionan el sistema según la programación y la información que reciben. Puede haber un controlador solo, o varios distribuidos por el sistema.

2.2.1.2. Actuador: El actuador es un dispositivo capaz de ejecutar y recibir una orden del controlador y realizar una acción sobre un aparato o sistema (encendido/apagado, subida/bajada, apertura/cierre, etc.).

2.2.1.3. Sensor: El sensor es el dispositivo que monitoriza el entorno captando información que transmite al sistema (sensores de agua, gas, humo, temperatura, humedad, iluminación, etc.).

2.2.1.4. Bus: El bus es el medio de transmisión que transporta la información entre los distintos dispositivos por un cableado propio, por la redes de otros sistemas (red eléctrica y red de datos) o de forma inalámbrica.

2.2.1.5. Interface: Los interfaces son los dispositivos (pantallas, móvil, Internet, conectores) y los formatos (binario, audio) en que se muestra la información del sistema para los usuarios (u otros sistemas) y donde los mismos pueden interactuar con el sistema.

2.2.2. La automatización como tecnología moderna

El término automatización se refiere a una amplia variedad de sistemas y procesos que operan con mínima o sin intervención del ser humano.

En los más modernos sistemas de automatización, el control de las máquinas es realizado por ellas mismas gracias a sensores de control que les permiten percibir cambios en sus alrededores de ciertas condiciones tales como temperatura, volumen y fluidez de la corriente eléctrica y otros sensores los cuales le permiten a un sistema realizar los ajustes necesarios para poder compensar estos cambios.

Y una gran mayoría de las operaciones industriales de hoy son realizadas por enormes máquinas de este tipo.

Donde la automatización es el uso de sistemas o elementos computarizados y electromecánicos para controlar maquinarias o procesos industriales. Como una disciplina de la ingeniería más amplia que un sistema de control, abarca la instrumentación industrial, que incluye los sensores, los transmisores de campo, los sistemas de control y supervisión, los sistemas de transmisión y recolección de datos y las aplicaciones de software en tiempo real para supervisar y controlar las operaciones de plantas o procesos industriales.

Los principales beneficios de la automatización en la actualidad son las siguientes:

- Fomenta la accesibilidad: Facilita el manejo de los elementos de un área a las personas con discapacidades de la forma que más se ajuste a sus necesidades, además de ofrecer servicio de tele-asistencias para aquellos que lo necesiten.

Aportando seguridad de personas y bienes: controles de instrucción y alarmas técnicas que permiten detectar incendios, fugas de gas o inundaciones de agua, etc.

- Convirtiendo un lugar más confortable: Gestión de equipos electrónicos, climatización ventilación, iluminación natural y Artificial. Garantizando las comunicaciones, recepción de avisos de anomalías e información del funcionamiento de equipos e instalaciones gestión remota del hogar, etc...

- Comunicaciones: Transmisión de voz y datos, incluyendo textos imágenes, sonidos (multimedia) con redes locales (LAN) compartiendo acceso a Internet recursos e intercambio entre todos los dispositivos acceso a nuevos servicios de telefonía sobre IP, televisión digital, televisión por cable diagnostico remoto, videoconferencias etc.

- Mantenimiento: Con capacidad de mantenimiento de los equipos vía remota.

2.2.3. Características de la automatización

La automatización tiene como característica principal el hacer funcionar independientemente un objeto o bien de forma semi-independiente del control humano; se dice "semi-independientes" porque aunque sean los dispositivos los que realicen la mayor parte del trabajo, para su correcto desempeño se necesita una supervisión humana. En comunicaciones, aviación, equipos de conmutación telefónica, astronáutica, pilotos automáticos y demás sistemas, todos los elementos se han automatizado para alcanzar una mayor rapidez y eficiencia en las diversas tareas. Si se desea una definición más técnica de lo que es un sistema automático decimos que éstos son mecanismos que funcionan en todo o parte por sí solos; pero como toda modalidad tuvo un origen el cual se puede ubicar en la segunda mitad del siglo XVIII.

Sin embargo, cuando más patente se hizo la automatización fue con la Revolución industrial

en 1750. La fabricación automatizada surgió de la íntima relación entre fuerzas económicas e innovaciones técnicas como la división del trabajo, la transferencia de energía y la mecanización de las fábricas. La división del trabajo permitió incrementar la producción y reducir el nivel de especialización de los obreros. La mecanización fue la siguiente etapa necesaria para la evolución hacia la automatización, facilitando la construcción de máquinas que reproducían los movimientos del trabajador. Además, el desarrollo de la tecnología energética también permitió la evolución del sistema industrial de producción puesto que la maquinaria no funciona sin la suficiente fuente de energía. En 1801, la patente de un telar automático utilizando tarjetas perforadas fue dada a Joseph Marie Jacquard, quien revolucionó la industria del textil.

Otros sistemas de producción antiguos son las primeras máquinas especiales para corte de metal construidas entre 1817 y 1870; el primer torno automático, inventado por Christopher Spencer; o los primeros controles hidráulicos, neumáticos y electrónicos para máquinas de corte automáticas en 1940.

2.2.4. Funciones del sistema de automatización

Las funciones principales de un sistema automatizado (AS) se basan en el tratamiento de los datos correspondientes a los diferentes sistemas de medida, organización de las bases de datos y obtención de los parámetros de interés.
En resumen, las funciones principales de un AS son:

- Señales de medida procedentes de los convertidores iniciales (sensores, calibradores) y su correspondiente conversión.
- Adquisición, transmisión e introducción de los datos en el sistema de ordenadores.
- Procesamiento inicial de los datos.
- Visualización en tablas, gráficos, histogramas.
- Agrupamiento de los datos (codificación, embalaje, etc.).
- Salida, acumulación y mantenimiento de las bases de datos (DB).
- Regulación automática de parámetros como la temperatura, presión,

composición gaseosa, etc.

- Programación y lógica de control, de tanto los objetos investigados como de la condiciones ambientales con el propósito de modelar físicamente el modo de investigación proporcionado.

Además, son necesarias ciertas directrices para un correcto desarrollo de los AS. En primer lugar, las acciones a ejercer deben ser lo más rápidamente posible de modo que la producción se eleve. En segundo lugar, no debe olvidarse mantener la calidad de este producto, de modo que el hecho de haber reducido el tiempo de producción consiga mejorar la precisión del producto final. Además, y en tercer lugar, las personas deben estar satisfecho de modo que otra de las directrices es conseguir aumentar el grado de fiabilidad y confianza de las soluciones obtenidas. Para ello, deben diseñarse los algoritmos de operación corres pendientes para ser utilizados el equipamiento específico así como para la protección matemática del sistema de control (CS). En cuarto lugar, y para obtener una elevada calidad, deberán crearse de nuevas medidas que consigan convertir los equipos básicos en las nuevas unidades más sofisticadas.

Finalmente, es que también debe poder asegurarse una máxima flexibilidad estructural a la hora de elaborar la arquitectura de producción con el fin de poder adaptar el sistema ante cualquier cambio.

2.3. RASPBERRY PI 2

2.3.1. Introducción a su historia

En 2006, los primeros diseños de Raspberry Pi se basaban en el microcontrolador Atmel ATmega644. Sus esquemas y el diseño del circuito impreso están disponibles para su descarga pública.

En mayo de 2009, la Fundación Raspberry Pi fue fundada en Caldecote, South Cambridgeshire, Reino Unido como una asociación caritativa que es regulada por la

Comisión de Caridad de Inglaterra y Gales.

El administrador de la fundación, Eben Upton, se puso en contacto con un grupo de profesores, académicos y entusiastas de la informática para crear un ordenador con la intención de animar a los niños a aprender informática como lo hizo en 1981 el ordenador Acorn BBC Micro.1617. El primer prototipo basado en ARM se montó en un módulo del mismo tamaño que una memoria USB. Tenía un puerto USB en un extremo y un puerto HDMI en el otro.

Fig. 2. Eben Upton, fundador Raspberry Pi

Fuente: https://www.gov.uk/government/world-location-news/eben-upton-of-the-raspberry-pi-foundation-visits-japan

El pre-lanzamiento en agosto de 2011, se fabricaron cincuenta placas Alpha, que tenían las mismas características que el modelo B, pero eran un poco más grandes para integrar bien unas interfaces para depuración. En algunas demostraciones se podía ver la placa ejecutando el escritorio LXDE en Debian, Quake3 a 1080p y vídeo Full HD H.264 a través de la salida HDMI.

En octubre de 2011, el logotipo se seleccionó entre varios diseños enviados por miembros de la comunidad. Durante el mismo mes, se trabajó en una versión de desarrollo de RISC OS y se hizo una demostración en público.

En diciembre de 2011, 25 placas Beta del modelo B fueron ensambladas y probadas de un

total de 100 placas vacías. El diagrama de componentes de las placas finales sería el mismo que el de esas placas Beta. Durante las pruebas a las placas beta se encontró un error de diseño en los pines que suministraban alimentación a la CPU que sería arreglado en la versión final. Se hizo una demostración de la placa beta arrancando Linux, reproduciendo un tráiler de una película a 1080p y ejecutando el benchmark Rightware Samurai OpenGL ES.

Durante la primera semana de diciembre de 2011, se pusieron a subasta diez placas en eBay. Una de ellas fue comprada por una persona anónima y se donó al Centro para la Historia de la informática en Suffolk, Inglaterra. En total se consiguieron 21.269 $us. La última placa, con número de serie No. 01 se vendió por 4.557 $us.

Debido al anticipado anuncio de puesta a la venta a final de febrero de 2012, la fundación sufrió colapso en sus servidores web debido a las actualizaciones de páginas desde los navegadores de gente interesada en la compra de la placa. El primer lote de 10.000 placas se fabricó en Taiwán y China, en vez de Reino Unido. Esto fue en parte porque los impuestos de importación se pagan para los componentes individuales pero no para productos acabados, y porque los fabricantes chinos ofrecían un plazo de entrega de 4 semanas y en el Reino Unido de 12. Con este ahorro conseguido, la fundación podía invertir más dinero en investigación y desarrollo.

Las primeras ventas comenzaron el 29 de febrero de 2012 a las 06:00 UTC; al mismo tiempo se anunció que el modelo A, que originalmente iba a tener 128 MB de RAM, tendría 256 MB. La página de la fundación también anunció que : "Seis años después del origen del proyecto, estamos cerca de finalizar el primer arranque del proyecto - aunque esto es solo el principio de la historia de Raspberry Pi". Por otro lado las dos tiendas que vendían las placas, Premier Farnell y RS Components, tuvieron una gran carga en sus servidores inmediatamente después del lanzamiento. La cuenta oficial de Raspberry Pi en Twitter informó que Premier Farnell vendió toda su existencia de inventario a los pocos minutos del momento de lanzamiento, mientras que RS Components tuvo 100.000 peticiones de interés el primer día. En los seis meses siguientes llegarían a vender 500.000 unidades.

2.3.2. Hardware

Las ventas iniciales fueron del modelo B. El modelo A solo tiene un puerto USB, carece de controlador Ethernet y cuesta menos que el modelo B, el cual tiene dos puertos USB y controlador Ethernet 10/100.53 En 2014 se lanzó el modelo Raspberry Pi 2 B. El último modelo lanzado en 2015 es el Raspberry Pi Zero.

A pesar que el Modelo A no tiene un puerto RJ45, se puede conectar a una red usando un adaptador USB-Ethernet suministrado por el usuario. Por otro lado, a ambos modelos se puede conectar un adaptador Wi-Fi por USB, para tener acceso a redes inalámbricas o internet. El sistema cuenta con 256 MB de memoria RAM en su modelo A, y con 512 MB de memoria RAM en su modelo B. Como es típico en los ordenadores modernos, se pueden usar teclados y ratones con conexión USB compatible con Raspberry Pi.

El Raspberry Pi no viene con reloj en tiempo real, por lo que el sistema operativo debe usar un servidor de hora en red, o pedir al usuario la hora en el momento de arrancar el ordenador. Sin embargo se podría añadir un reloj en tiempo real (como el DS1307) con una batería mediante el uso de la interfaz I²C.

Los esquemas del modelo A y el modelo B fueron lanzados el 20 de abril de 2012 por la fundación. La aceleración por hardware para la codificación de vídeo (H.264) se hizo disponible el 24 de agosto de 2012, cuando se informó que la licencia permitiría su uso gratuitamente; se pensó en anunciarlo cuando se lanzara el módulo de cámara. También se puso a la venta la capacidad para poder usar el codificación-decodificación de MPEG-2 y Microsoft VC-1. Por otro lado, se hizo saber que el ordenador soportaría CEC, permitiendo que pudiera ser controlado mediante un mando a distancia de televisión.

El 5 de septiembre de 2012, se anunció una revisión 2.0 de la placa, que ofrecía un pequeño número de correcciones y mejoras, como unos agujeros de montaje, un circuito para hacer reset, soporte para depuración JTAG, etc.

El 15 de octubre de 2012, la fundación anunció que todos los Raspberry Pi Modelo B serían enviados a partir de ese momento con 512 MB de RAM en vez de 256 MB.

2.3.3. Sistemas operativos

A continuación se muestra una lista de sistemas operativos que pueden alojarse en el RBP o están en proceso de ser lanzados:

- Sistemas operativos
  - AROS
  - Linux
    - Android99
    - Arch Linux ARM
    - Debian Whezzy Soft-Float, versión de Debian sin soporte para coma flotante por hardware
    - DietPi, distribución ligera basada en Raspbian y de sencilla configuración mediante menús
    - Firefox OS
    - Gentoo Linux100
    - Google Chromium OS
    - Kali Linux
    - Open webOS101
    - PiBang Linux,102 distribución Linux derivada de Raspbian con diferente escritorio y aplicaciones
    - Pidora, versión Fedora Remix optimizada103
    - QtonPi, distribución linux con un framework de aplicaciones multiplataforma basado en Qt framework
    - Raspbian,104 versión de Debian Wheezy para ARMv6 con soporte para coma flotante por hardware
    - Slackware ARM, también conocida como ARMedslack
    - Ubuntu MATE

- Plan 9 from Bell Labs105 106
- RISC OS 52
- Unix
  - FreeBSD107
  - NetBSD108 109
- Windows 10
  - Windows CE

Distribuciones ligeras multipropósito:

- Minibian, distribución ligera basada en Raspbian.
- Moebius, distribución ligera ARM HF basada en Debian que usa el repositorio de Raspbian y que cabe en una tarjeta SD de 1GB, usa pocos servicios y está optimizada para usar poca memoria
- Squeezed Arm Puppy, una versión de Puppy Linux (Puppi) para ARMv6 (sap6) específicamente para Raspberry Pi110

2.3.4. Pines GPIO

GPIO (General Purpose Input/Output) es, como su propio nombre indica, un sistema de E/S (Entrada/Salida) de propósito general, es decir, una serie de conexiones que se pueden usar como entradas o salidas para usos múltiples. Estos pines están incluidos en todos los modelos de Raspberry Pi, para que puedas realizar proyectos interesantes como lo harías con Arduino.

Los GPIO representan la interfaz entre la Raspberry Pi y el mundo exterior. Y con ello se aprovechara al máximo, desde hacer titilar un LED hasta otros mucho más sofisticados. Pero para se debe saber las características y como se debe programar. Lo primero variará en función de la revisión de placa que tengas o del modelo.

Todos los pines son de tipo “unbuffered”, es decir, no disponen de buffers de protección, así que hay que tener cuidado con las magnitudes (voltajes, intensidad,…) cuando conectes

componentes a ellos para no dañar la placa. Como podrás apreciar en las imágenes posteriores, no todos los pines tienen la misma función:

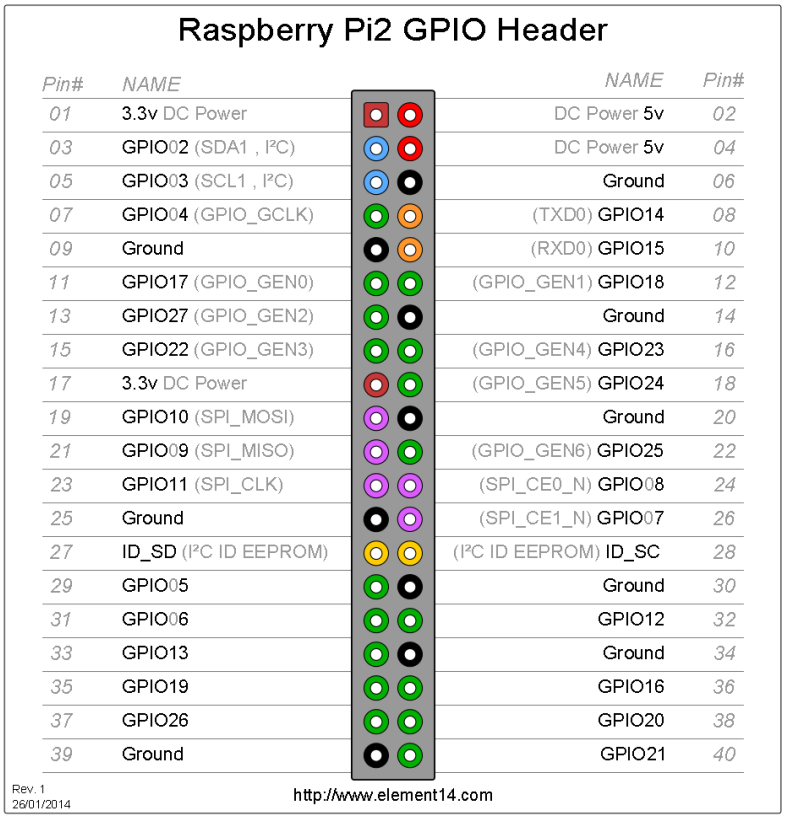


Fig.3 Pines GPIO del Raspberry Pi 2

Fuente: https://es.pinout.xyz/

- Pines de alimentación: se puede apreciar pines de 5v, 3v3 (limitados a 50mA) y tierra (GND o Ground), que aportan alimentación a estos voltajes para los circuitos. Pueden servir como una fuente de alimentación, aunque también puedes utilizar otras fuentes (pilas, fuentes de alimentación externas, etc). Recordemos que son unbuffered y hay que tener cuidado para no dañar la placa.

- DNC (Do Not Connect): son pines que por el momento no tienen función, pero en futuras implementaciones son utilizados para otros fines. Por eso solo se va a encontrar

en modelos más primitivos de la Raspberry Pi. En las actuales placas han sido marcados como GND.

- GPIO normales: son conexiones configurables que se puede programar para el proyecto, tal como veremos más adelante.

- GPIO especiales: dentro de éstos se encuentran algunos pines destinados a una interfaz UART, con conexiones TXD y RXD que sirven para comunicaciones en serie, como por ejemplo, conectar con una placa Arduino. También podemos ver otros como SDA, SCL, MOSI, MISO, SCLK, CE0, CE1, etc…, los cuales explicaremos su funcionamiento más adelante.

2.3.5. Almacenamiento

La Raspberry Pi B+ no dispone de un disco duro, para ello trae un lector/ranura para memorias microSD, un sistema de almacenamiento en estado sólido. El arranque del sistema se hará desde la propia tarjeta microSD, con lo que debido a que tiene que albergar todo el sistema operativo, es necesario que la tarjeta sea de al menos 2 GB de capacidad para almacenar todos los archivos requeridos.

La placa carece de botones de encendido y apagado. Para su alimentación dispone de un conector micro USB tipo B que proporciona 5 V de tensión. La mayoría de los cargadores para smartphones (que suministren más de 700 mA) son compatibles para dar tensión a la Raspberry Pi.

2.3.5. Puerto Ethernet

Se dispone de un conector RJ-45 conectado al integrado LAN9514 de SMSC, nombrado en el apartado, que proporciona conectividad a 10/100 Mbps.

Es posible conectar la Raspberry directamente a un PC sin pasar por un router conectando ambos equipos de manera directa con un cable RJ45.

Los modelos actuales de la Raspberry Pi no cuentan con un componente para poder acceder a redes inalámbricas, pero es posible añadir soporte Wi-Fi a la Raspberry utilizando un adaptador USB para red inalámbrica.

2.3.6. Instalación del Sistema Operativo

Dado que se va a trabajar con Python; el sistema operativo que se va a instalar en este caso es Raspbian.

Para poder instalar la distribución elegida es necesario usar un ordenador con lector de tarjetas SD. Usando un ordenador con Windows, el primer paso que hay que realizar es obtener el sistema operativo desde el siguiente enlace:

https://www.raspberrypi.org/downloads/

Posteriormente, se necesita descargar el software gratuito "Win32 Disk Imager". En Image File se escoge el fichero que contenga la imagen del Sistema Operativo, en Device hay que asegurarse que se elige la unidad correcta asociada a la tarjeta SD y mediante el botón Write se pasa el sistema a la tarjeta.

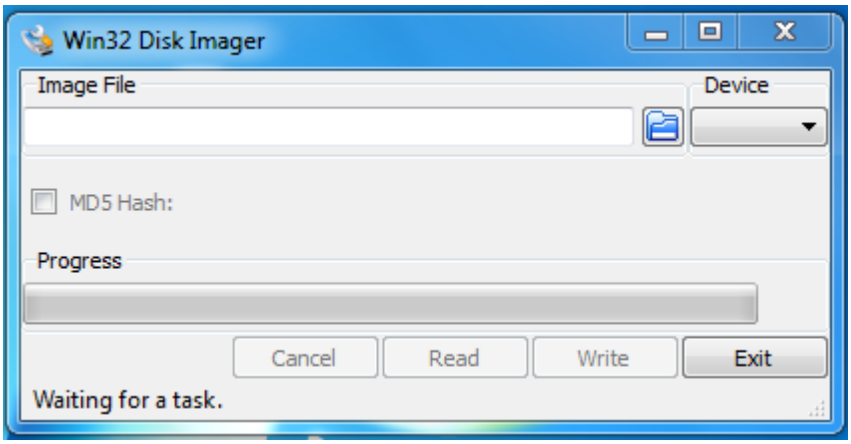


Fig. 4 Captura programa Win32 Disk Imager

Fuente: Elaboracion propia

Una vez finalizado este proceso, ya se puede insertar la tarjeta en Raspberry Pi y comenzar a trabajar.

2.4. ARDUINO

2.4.1 Introducción a su historia

Arduino se inició en el año 2006 como un proyecto para estudiantes en el Instituto IVREA, en Ivrea (Italia). En ese tiempo, los estudiantes usaban el microcontrolador BASIC Stamp, cuyo coste era de 100 dólares estadounidenses, lo que se consideraba demasiado costoso para ellos.

El nombre del proyecto viene del nombre del Bar di Re Arduino (Bar del Rey Arduino) donde Massimo Banzi pasaba algunas horas. El rey Arduino fue rey de Italia entre los años 1002 y 1014. En la creación de este proyecto contribuyó el estudiante colombiano Hernando Barragán, quien desarrolló la tarjeta electrónica Wiring, el lenguaje de programación y la plataforma de desarrollo. Una vez concluida dicha plataforma, los investigadores trabajaron para hacerlo más ligero, más económico y disponible para la comunidad de código abierto (hardware y código abierto). El instituto finalmente cerró sus puertas, así que los

investigadores, entre ellos el español David Cuartielles, promovieron la idea. Banzi afirmaría años más tarde, que el proyecto nunca surgió como una idea de negocio, sino como una necesidad de subsistir ante el inminente cierre del Instituto de diseño Interactivo IVREA. Es decir, que al crear un producto de hardware abierto, este no podría ser embargado.

Posteriormente, Google colaboró en el desarrollo del Kit Android ADK (Accesory Development Kit), una placa Arduino capaz de comunicarse directamente con teléfonos móviles inteligentes bajo el sistema operativo Android para que el teléfono controle luces, motores y sensores conectados de Arduino.

Para la producción en serie de la primera versión se tomó en cuenta que el coste no fuera mayor de 30 euros, que fuera ensamblado en una placa de color azul, debía ser Plug and Play y que trabajara con todas las plataformas informáticas tales como MacOSX, Windows y GNU/Linux. Las primeras 300 unidades se las dieron a los alumnos del Instituto IVREA, con el fin de que las probaran y empezaran a diseñar sus primeros prototipos.

En el año 2005, se incorporó al equipo el profesor Tom Igoe, que había trabajado en computación física, después de que se enterara del mismo a través de Internet. Él ofreció su apoyo para desarrollar el proyecto a gran escala y hacer los contactos para distribuir las tarjetas en territorio estadounidense. En la feria Maker Fair de 2011 se presentó la primera placa Arduino 32 bit para realizar tareas más pesadas.

2.4.2. Hardware del Arduino

Hay infinidad de placas basadas en Arduino. Como ya es sabido es "Open-source", así que cualquiera que quiera hacer una placa puede hacerlo. Y por ello tenemos Arduino de todos los colores, tamaños y con funciones propietarias de lo más diverso, y también productos que sin ningún pudor están basados en Arduino para controlar a su vez distintos dispositivos, integrados en el producto o no.

El hardware Arduino más sencillo consiste en una placa con un microcontrolador y una serie

de puertos de entrada y salida. Los Microcontroladores AVR más usados son el Atmega168, Atmega328, Atmega1280, y Atmega8 por su sencillez y bajo coste que permiten el desarrollo de múltiples diseños, aunque también nos encontramos Microcontroladores CortexM3 de ARM de 32 bits, que coexistirán con las más limitadas, pero también económicas AVR de 8 bits. ARM y AVR son plataformas diferentes, pero gracias al IDE de Arduino los programas se compilan y luego se ejecutan sin cambios en cualquiera de las plataformas. Hay algunos casos que hay ciertos problemas de compatibilidad de librerías entre plataformas, así que habrá que tenerlo en cuenta a la hora de elegir placa Arduino el sistema operativo que se maneja.

La diferencia entre los distintos Arduino se la encuentra por un lado en la tensión utilizada en las placas. Generalmente las Microcontroladores con CortexM3 tienen un voltaje de 3,3 voltios, mientras que la mayor parte de las placas con AVR utilizan una tensión de 5 voltios. Esto luego es fundamental para utilizar lógica TTL (frente a lógica CMOS) por ejemplo, lo que abre la posibilidad de utilizar chips baratos y complementar el Arduino con alguna funcionalidad externa. También hay placas que pueden conmutar el voltaje, así que tampoco es un factor determinante para seleccionar una placa u otra. Y, por otra parte, el número de conexiones, procesador utilizado, memoria y, sobre todo, el número de entradas y salidas y la posibilidad de alimentar distintos elementos desde la propia placa Arduino.

Fig. 5. Arduino MEGA, Atmega328

Fuente: http://www.xataka.com/especiales/guia-del-arduinomaniaco-todo-lo-que-necesitas-saber-sobre-arduino

Hay placas que incluso no necesitan drivers para Linux o para Mac, como las basadas en el chip ATmega8U2 (un ejemplo es el Tosduino Uno R3), siendo detectado por dichos ordenadores como un periférico más.

2.4.3. ¿Por qué Arduino?

Hay muchos otros Microcontroladores y plataformas microcontroladoras disponibles para computación física. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, y muchas otras ofertas de funcionalidad similar. Todas estas herramientas toman los desordenados detalles de la programación de Microcontroladores y la encierran en un paquete fácil de usar. Arduino también simplifica el proceso de trabajo con Microcontroladores, pero ofrece algunas ventajas para profesores, estudiantes y aficionados interesados sobre otros sistemas:

- **Barato:** Las placas Arduino son relativamente baratas comparadas con otras plataformas Microcontroladores. La versión menos cara del módulo Arduino puede ser ensamblada a mano, e incluso los módulos de Arduino pre-ensamblados cuestan menos de 50$ (450Bs).

- **Multiplataforma:** El software de Arduino se ejecuta en sistemas operativos Windows, Macintosh OSX y GNU/Linux. La mayoría de los sistemas Microcontroladores están limitados a Windows.

- **Entorno de programación simple y clara**: El entorno de programación de Arduino es fácil de usar para principiantes, pero suficientemente flexible para que usuarios avanzados puedan aprovecharlo también. Para profesores, está convenientemente basado en el entorno de programación Processing, de manera que estudiantes aprendiendo a programar en ese entorno estarán familiarizados con el aspecto y la imagen de Arduino.

- **Código abierto y software extensible:** El software Arduino está publicado como herramientas de código abierto, disponible para extensión por programadores

experimentados. El lenguaje puede ser expandido mediante librerias C++, y la gente que quiera entender los detalles técnicos pueden hacer el salto desde Arduino a la programación en lenguaje AVR C en el cual está basado. De forma similar, puedes añadir código AVR-C directamente en tus programas Arduino.

- **Código abierto y hardware extensible:** El Arduino está basado en microcontroladores ATMEGA8 y ATMEGA168 de Atmel. Los planos para los módulos están publicados bajo licencia Creative Commons, por lo que diseñadores experimentados de circuitos pueden hacer su propia versión del módulo, extendiéndolo y mejorándolo. Incluso usuarios relativamente inexpertos pueden construir la versión de la placa del módulo para entender cómo funciona y ahorrar dinero.

2.4.4. IDE ARDUINO EN UBUNTU / LINUX

La instalación del IDE de Arduino en Linux es incluso más sencilla. Debemos de abrir una consola de comandos y escribir:

- *sudo apt-get install arduino*

Ya teniendo instalada la última versión estable de Arduino. Si se quiere la última versión Beta o Nightly se debe bajarla de la siguiente forma:

- *sudo rm -r /usr/share/Arduino*
- *wget http://http://downloads.arduino.cc/arduino-1.5.4-linux64.tgz*
- *tar zxvf arduino-1.5.4-linux64.tgz*
- *sudo mv ./arduino-1.5.4 /usr/share/arduino*

Con esto se habrá sustituido la versión de Arduino por la más actual. Recordar cambiar el nombre del archivo por la última versión disponible en el momento.

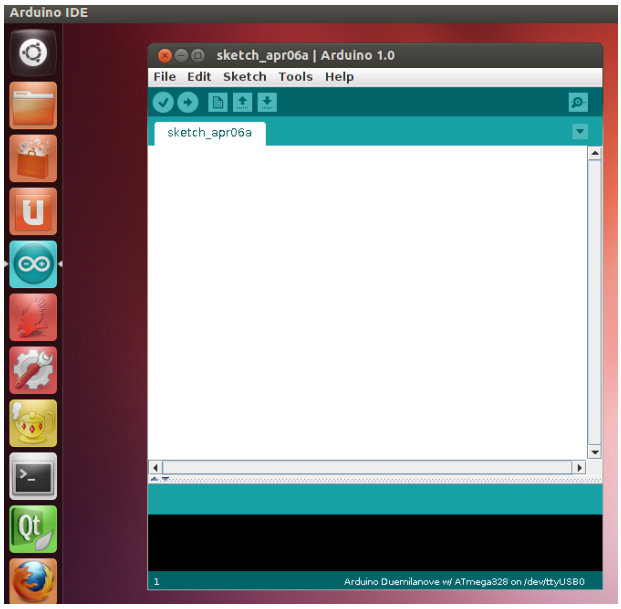


Fig. 6 Captura de pantalla, IDE Arduino

Fuente: Elaboracion propia

2.5. ARDUINO ETHERNET SHIELD

La Arduino Ethernet Shield permite a una placa Arduino conectarse a internet. Está basada en el chip ethernet Wiznet W5100 (datasheet). El Wiznet W5100 provee de una pila de red IP capaz de TCP y UDP. Soporta hasta cuatro conexiones de sockets simultáneas. Usa la librería Ethernet para escribir programas que se conecten a internet usando la shield.

La ethernet shield dispone de unos conectores que permiten conectar a su vez otras placas encima y apilarlas sobre la placa Arduino el cual usa los pines digitales 10, 11, 12, y 13 (SPI) para comunicarse con el W5100 en la ethernet shield. Estos pines no pueden ser usados para e/s genéricas.

El botón de reset en la shield resetea ambos, el W5100 y la placa Arduino.

La shield contiene un número de LEDs para información:

- PWR: indica que la placa y la shield están alimentadas

- LINK: indica la presencia de un enlace de red y parpadea cuando la shield envía o recibe datos

- FULLD: indica que la conexión de red es full dúplex 100M: indica la presencia de una conexión de red de 100 Mb/s (de forma opuesta a una de 10Mb/s)

- RX: parpadea cuando la shield recibe datos

- TX: parpadea cuando la shield envía datos

- COLL: parpadea cuando se detectan colisiones en la red

Fig. 7 Arduino Shield Ethernet

Fuente: www.prometec.com

El jumper soldado marcado como "INT" puede ser conectado para permitir a la placa Arduino recibir notificaciones de eventos por interrupción desde el W5100, pero esto no está soportado por la librería Ethernet. El jumper conecta el pin INT del W5100 al pin digital 2 de Arduino.

Para usar la Ethernet Shield solo hay que montarla sobre la placa Arduino. Para cargar los sketches a la placa conectarla al ordenador mediante el cable USB como se hace normalmente. Una vez que el sketch ha sido cargado se puede desconectar la placa del

ordenador y alimentarla desde una fuente externa.

Conectar la Ethernet Shield a un ordenador, a un switch o a un router utilizando un cable ethernet standard (CAT5 oCAT6 con conectores RJ45). La conexión al ordenador puede requerir el uso de un cable cruzado (aunque muchos ordenadores actuales, incluyendo los últimos modelos Mac pueden hacer el cruce de forma interna).

Al shield se debe asignar una dirección MAC y una IP fija utilizando la función Ethernet.begin(). Una dirección MAC es un identificador global único para cada dispositivo en particular; asignar una al azar suele funcionar, pero no es usual usar la misma para más de una placa.

2.5.1. Característica del Shield Ethernet

Este Shield es 100% compatible con Arduino Duemilanove, UNO y MEGA. La placa Arduino se comunica con el módulo W5100 y la micro-SD utilizando el bus SPI(mediante el conector ICSP). Esto se encuentra en los pines digitales 11, 12 y 13 en el modelo Duemilanove y en los pines 50, 51 y 52 del modelo MEGA. En ambas placas, el pin 10 es utilizado para seleccionar el W5100 y el pin 4 para la micro-SD. Estos pines no pueden ser utilizados para otros fines mientras la Ethernet Shield esté conectada. El en MEGA, el pin SS no es utilizado pero debe dejarse como salida para que el bus SPI funcione correctamente.

Se debe tener en cuenta que el W5100 y la micro-SD comparten el bus SPI, por lo que sólo uno de ellos puede ser utilizado a la vez.

2.6. COMPONENTES

2.6.1. Modulo Bluetooth HC-05

2.6.1.1. Las redes Bluetooth

Muchos usuarios en el mundo poseen un teléfono móvil en el bolsillo, y muy probablemente será del tipo Smartphone, que incluirán un sistema operativo como Android o Apple IOS equipados con Wifi y Bluetooth.

Por eso la posibilidad de integrar conexión WIFI o Bluetooth en el actual proyecto, nos abre unas posibilidades inmensas. Poder controlar nuestro proyecto desde el propio móvil, bien con WIFI o bien con Bluetooth, es algo muy interesante para controlar montajes de diferentes tipos.

Fig. 8 Logotipo Bluetooth
Fuente: http://www.prometec.net/wp-content/uploads/2014
/11/bluetooth_logo.jpg

En el anterior punto, se vio las partes del módulo Shield Ethernet para que se pueda conectar a la red local para gobernar múltiples salidas digitales.

Ahora se verá un poco de las características del Bluetooth, para ver de qué posibilidades se dispone, como configurarlos y demás. De modo que se pueda integrar el Bluetooth con los Arduinos.

Y para ello se hablara un poco, de que es el Bluetooth y de cómo funciona, así como unos

pocos conceptos básicos claves, para poderlo usar con garantías de éxito.

Para empezar hay que decir que los dispositivos Bluetooth pueden actuar como Masters o como Slaves (maestros o esclavos).

La diferencia es que un Bluetooth Slave solo puede conectarse a un master y a nadie más, en cambio un master Bluetooth, puede conectarse a varios Slaves o permitir que ellos se conecten y recibir y solicitar información de todos ellos, arbitrando las transferencias de información (Hasta un máximo de 7 Slaves)

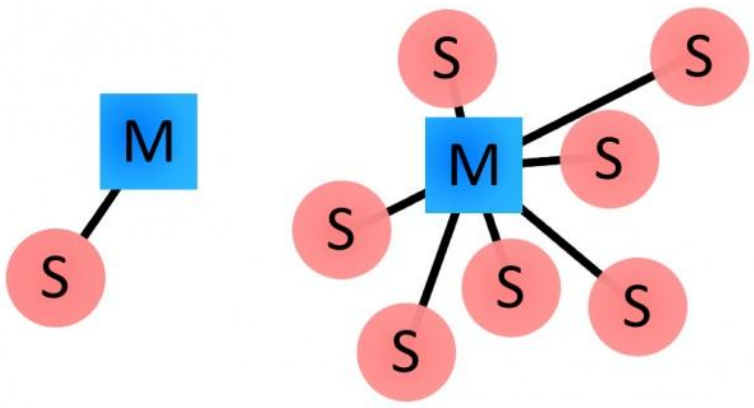


Fig. 9 Masters/Slaves en Bluetooth
Fuente: http://www.prometec.net/wp-content/uploads/2014
/11/BT-masterslave.jpg

Cada uno de los dispositivos que se identifican vía Bluetooth presenta una dirección única de 48 bits y además un nombre de dispositivo que sirva para identificarlo cómodament. Por eso cuando se configura el móvil se puede especificar nombre propio que será el que mostrara a los demás cuando busquen tu teléfono en la inmediaciones.

La dirección propia también se puede identificar pero lógicamente, es un poco menos cómoda y tiene menos utilidad. Tampoco es raro establecer un protocolo IP sobre transporte Bluetooth, con lo que además de su identificación interna Bluetooth

(Equivalente al MAC Ethernet) dispondrá de una dirección IP para conectarse a Internet.

Por se puede conectar vía Bluetooth a tu PC, por ejemplo, y a través del conectar a internet. Así un nodo Bluetooth puede ser Master o Slave y dispone de una dirección única, así como de un nombre para identificarse y muy habitualmente también incluye un PIN de conexión o número de identificación que debe teclearse para ganar acceso al mismo.

Como el Bluetooth lo desarrolló Nokia para conectar teléfonos móviles, a otros dispositivos como auriculares, micrófonos o conexiones al audio del coche, existe un procedimiento definido que se llama Pairing (o emparejamiento) que vincula a dos dispositivos Bluetooth. Cuando vinculas dos dispositivos BT, se inicia un proceso en el que ellos se identifican por nombre y dirección interna y se solicitan la clave PIN para autorizar la conexión.

Si el emparejamiento se realiza con éxito, ambos nodos suelen guardar la identificación del otro y cuando se encuentran cerca se vuelven a vincular sin necesidad de intervención manual. Por eso el reproductor de un vehículo reconoce un móvil cualquiera y se puede reproducir la música que se tiene en un Smartphone.

Aunque un dispositivo Bluetooth pueda enviar o recibir música, debe aceptar otra norma posterior llamada Advanced Audio Distribution Profile (A2DP) y que en caso de ser algún sistema antiguo impedirá la reproducción. Naturalmente, a lo largo de los años la norma ha ido variando y existen varias versiones de la misma, con compatibilidad siempre con las versiones anteriores que se diferencian en la distancia que pueden alcanzar (entre 50 y 100 metros, teóricamente y sin obstáculos) además de la velocidad de transferencia.

Una de las ventajas principales del módulo HC-06, además de su pequeño tamaño y sus buenas características de transmisión y recepción que le brindan un alcance muy amplio (por tratarse de un sistema local Bluetooth), es el bajo consumo de corriente que posee tanto en funcionamiento, como en modo de espera, es decir, alimentado con energía, pero

sin conexión o enlace a otro dispositivo, por ejemplo, un móvil con SO Android. Otra característica interesante de este módulo es que una vez que ha realizado un enlace con otro dispositivo es capaz de recordarlo en su memoria y no solicita validación alguna ("1234" por defecto), pero si se activa el pin 26 (KEY) hacia la tensión de alimentación, esta información se elimina y el módulo HC-06 solicitará nuevamente la validación del enlace. Otro detalle particular es que su tensión de alimentación de 3,3Volts y su bajo consumo (8mA en transmisión/recepción activa) lo transforman en un dispositivo ideal para trabajar con Microcontroladores de la misma tensión de alimentación, logrando de este modo equipos portátiles que pueden ser alimentados durante muchas horas por baterías recargables o alcalinas AA, demostrando características excepcionales en aplicaciones médicas, o para actividades recreativas donde la fuente energética debe ser liviana y portátil.

2.6.1.2. Módulos Bluetooth disponibles para Arduino

Hace ya un tiempo que se dispone de módulos Bluetooth sencillos y económicos, que resultan muy prácticos para todo esto, y en esta parte, se puede empezar viendo cuales están disponibles y como trabajar con ellos.

Los más frecuentes en el mercado son los módulos HC-06 y HC-05 que si se hace una búsqueda por eBay o similares se verá que se consiguen por poco dinero, y están disponibles independientes o en modo SHIELD.

Hay bastante confusión en la red acerca de cómo diferenciar uno de otro y en muchas ocasiones sobre las prestaciones de unos y otros.

Lo primero es que el hardware de ambos módulos es el mismo. No hay diferencia en hardware, pero el software que incorporan es diferente.

Además el módulo de conexión se monta sobre un soporte que a su vez puede presentar diferencias notables según el fabricante y las conexiones que realice, pero por lo que se ha

podido comprobar con un par de módulos de los que se dispone, hay una diferencia obvia, el número de pines del módulo montado.

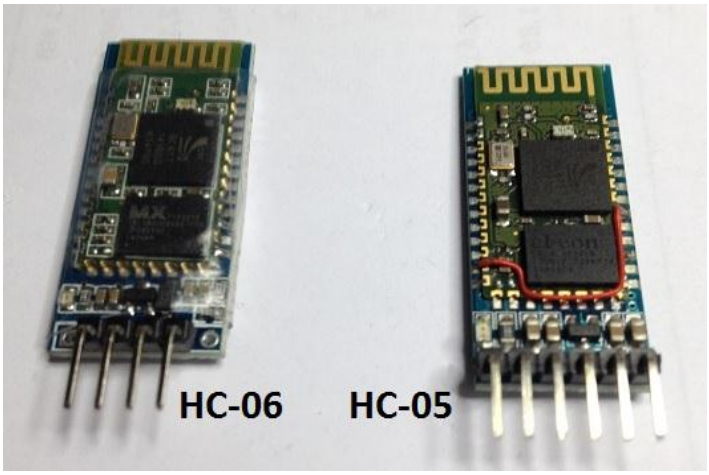


Fig. 10 Tipos de Módulos Bluetooth
Fuente: http://www.prometec.net/wp-content/uploads/2014
/11/BT-masterslave_2.jpg

El modelo HC-06 dispone de 4 pines, en lugar de los 6 que incluye el modelo HC-05, pero hay además importantes diferencias de funcionalidad y de manejo que hace que merezca la pena investigar para cada modelo.

Básicamente el modelo HC-06 solo puede actuar como esclavo y además dispone de un juego reducido de instrucciones a las que atiende, mientras que el modelo HC-05 puede actuar como master o como Slave y acepta un número mayor de órdenes de configuración.

En el actual proyecto se va a usar el modelo HC-06, pero antes se debe hablar los comandos AT.

2.6.1.3. Los comandos AT

En épocas pasadas, para enviar mensajes (de texto exclusivamente) de un ordenador a otro,

se usaba las líneas o puertas serie. Muy al estilo de la comunicación que hoy el Arduino lo hace con una PC a través del USB.

El interface no era USB sino RS232, que aunque físicamente diferente, comunicaban vía serie dos puntos próximos hasta un máximo, de digamos, unos 100 metros.

Pero cuando se quería enviar información a otro equipo situado en una oficina remota de la empresa, la única posibilidad era utilizar las líneas telefónicas, mediante un adaptador, que se llamaba Modem. La idea básica, era codificar el 0 binario con un tono grave de audio (que se pudiera enviar por la línea de teléfono) y los unos binarios como un tono agudo.

Así con la sucesión más o menos rápida de tonos graves y agudos por la línea telefónica se podían enviar un mensaje binario codificado en frecuencia de audio, de un punto a otro.

La palabra Modem, que tal vez se ha oído, deriva del apocope de modulador, y según las películas de los años 80 y primeros 90 eran esos chismes que hacían los ruiditos típicos de audio, asociados a la informática.

Naturalmente, las líneas telefónicas eran analógicas, lo que implicaba una cantidad indecente de ruido térmico, que aumentaba exponencialmente con la distancia, obligando a disminuir la velocidad en función del ruido y a retransmitir una y otra vez el mensaje hasta que se consiga que llegara correctamente (algo que rara vez pasaba a la primera)

La velocidad de los módems empezaron sobre los 300 baudios o bits por segundos y fueron aumentando con la revolución tecnológica hasta 1.200, 2.400, 9.600 y los últimos que se comercializaron con ese nombre fueron de 56.000 baudios. (Se puede comparar ese 56 k de máximo con las redes de cable modernas de 100 Mbps, casi 1.800 veces más rápidas)

Por eso, los módems necesitaban una especie de comandos, que permitieran modificar la velocidad según se necesitara, cuando el ruido impedía una comunicación fiable y ya de

paso modificar algún que otro parámetro sobre la marcha.

Y como el RS232 solo disponía de dos hilos de comunicaciones, no había más remedio que incorporar una orden de atención que significara que a continuación venia una orden de programación del modem, que no debía ser transmitida al otro extremo.

Es decir que las ordenes eran del tipo "AT+Orden", donde AT era el comando especificado de atención. De inmediato todos los módems y demás máquinas de comunicación serie empezaron a aceptar este tipo de órdenes, y al conjunto de ellas se llamó comandos AT.

Con el tiempo, se vio que el hardware sigue siendo sencillo y se comunica con otros equipos vía puerta serie, siguen aceptando ordenes AT para configurarlos y este es el caso de los módulos Bluetooth HC-05, HC-06 y otros pequeños dispositivos que aún se siguen utilizando en los computadores.

2.6.1.4. Comando AT para configurar

El primero y más básico es enviar (siempre en mayúsculas) un simple AT desde la consola. Se supone que debería responder con OK, pero en este caso y con el modulo no es así. Y si es este caso es el mismo se debe probar con más comandos.

AT+VERSION, Requiere la versión del Firmware
AT+NAMEXXX

Cambiar el nombre que se quiere presentar cuando otros dispositivos busquen:

AT+NAMEDATACENTER
AT+BAUDX

Fija la velocidad de comunicación entre el modulo y la consola según la siguiente tabla:

1 configura	1200bps
2 configura	2400bps
3 configura	4800bps
4 configura	9600bps (Por defecto)
5 configura	19200bps
6 configura	38400bps
7 configura	57600bps
8 configura	115200bps

Ejemplo: AT+BAUD7 configura la comunicación a 57600 baudios

AT+PINXXXX, configura el número de identificación personal, que se requerirá para establecer la vinculación

AT+PIN4516, establece 4516 como PIN

En caso de que probando varios de estos comandos, no se obtuviera una respuesta, o se vean caracteres extraños en la consola, se debe de probar con otras velocidades hasta conseguir una que funcione. Se configura con la velocidad recomendada en la línea de consola:

```
BT1.begin(9600);
```

En este caso son todos los comandos que se dispone para un módulo HC-06 y desde luego no dispone de los comandos de interrogación de los que si dispone su hermano mayor HC-05.

Fig. 11 Ingresando los comandos AT

Fuente: Elaboracion propia

2.6.1.5. Probando la conexión con el módulo HC-05

Una vez cargado el programa de emparejamiento al Arduino, con una velocidad de comunicación correcta con el modulo se va a probar a enviar y recibir información desde y hacia, un teléfono móvil.

Para ello se va a utilizar un teléfono Android y un programa Android llamado Bluetooth SPP, pero en realidad servirá cualquier terminal BT.

Se aclara que no se puede vincular este módulo a un Iphone 4S, porque este dispositivo dispone de permisos para el emparejamiento de los bluetooth pero en cambio se puede vincular rápidamente con un Smartphone con S.O. Android.

Ahora se debe confirmar que el LED del módulo BT parpadea, indicando que está en modo AT o esperando vinculación con otro dispositivo. Después ver que el Bluetooth del teléfono está activo y por último, dependiendo del programa que se usa como terminal,

ahora se debe pedir que busque dispositivos BT en los alrededores y vincúlate al que encuentres.

Si todo va bien ya se podrá enviar y recibir mensajes de texto entre el móvil y Arduino en ambas direcciones.

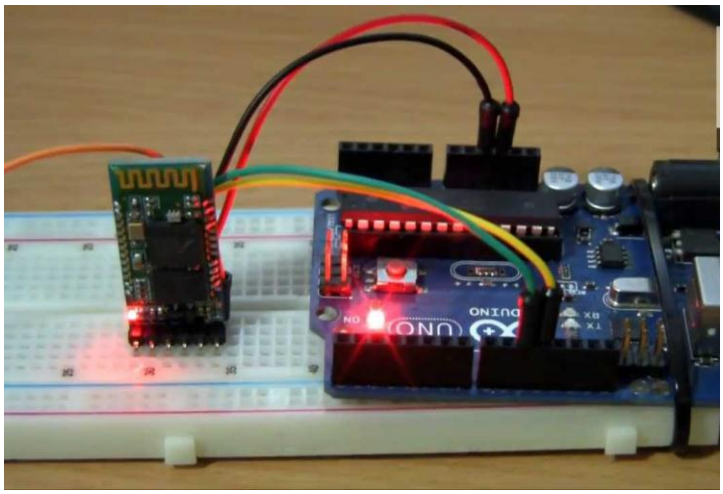


Fig. 12 Conexión del HC-05 y el Arduino UNO

Fuente: Fotografia

2.6.2. SENSORES DE TEMPERATURA Y HUMEDAD

2.6.2.1. El sensor DHT11

En el diseño del proyecto se utilizará el sensor con el código DHT11 que es un sensor de temperatura y humedad digital de bajo costo. Utiliza un sensor capacitivo de humedad y un termistor para medir el aire circundante, y muestra los datos mediante una señal digital en el pin de datos (no hay pines de entrada analógica). Es bastante simple de usar, pero requiere sincronización cuidadosa para tomar datos. El único inconveniente de este sensor es que sólo se puede obtener nuevos datos una vez cada 2 segundos, así que las lecturas

que se pueden realizar serán mínimas cada 2 segundos.

En comparación con el DHT22, este sensor es menos preciso, mas exacto y funciona en un rango más pequeño de temperatura / humedad, pero su empaque es más pequeño y menos caro. Pero en muchas ocasiones, y especialmente en la industria alimentaria, no basta con medir la temperatura, sino que la humedad relativa es también un factor importante a tener en cuenta. Por eso se desarrollaron los sensores de la familia DHT. Nos proporcionan de forma digital la temperatura y la humedad, con diferente precisión según el modelo. Básicamente hay dos variantes DHT11 y DHT22

Fig. 13 Sensor de Humedad y Temperatura
Fuente: http://www.prometec.net/sensores-dht11/

2.6.2.2. Características y funciones

Este sensor de temperatura y humedad DHT11 dispone de una salida calibrada de señal digital con la temperatura y el sensor de humedad. Su tecnología garantiza la alta fiabilidad y una excelente estabilidad a largo plazo.

Un alto rendimiento de 8-bits; Este sensor además incluye un elemento resistivo y una sensación de mojado NTC dispositivos de medición de temperatura. Tiene una excelente

calidad, rapidez de respuesta, la capacidad anti-interferencia y ventajas de rendimiento.

Estos sensores cuentan con calibración extremadamente precisa de la cámara de humedad de calibración. Los coeficientes ya se encuentran en su estado ideal y no es necesario de una nueva calibración.

Sus características del sensor son las siguientes:

- Alimentación: 3Vdc ≤ Vcc ≤ 5Vdc
- Rango de medición de temperatura: 0 a 50 °C
- Precisión de medición de temperatura: ±2.0 °C .
- Resolución Temperatura: 0.1°C
- Rango de medición de humedad: 20% a 90% RH.
- Precisión de medición de humedad: 4% RH.
- Resolución Humedad: 1% RH
- Tiempo de sensado: 1 seg.

2.6.2.3. Diagrama del circuito de conexión

La conexión es sencilla, pero cabe destacar que se vende en dos encapsulados, uno de tres pines que son GND, Data y Vcc, y el otro cuarto pin de la otra versión del sensor sencillamente sobra y no se conecta. Normalmente viene rotulado en el sensor el nombre de cada pin.

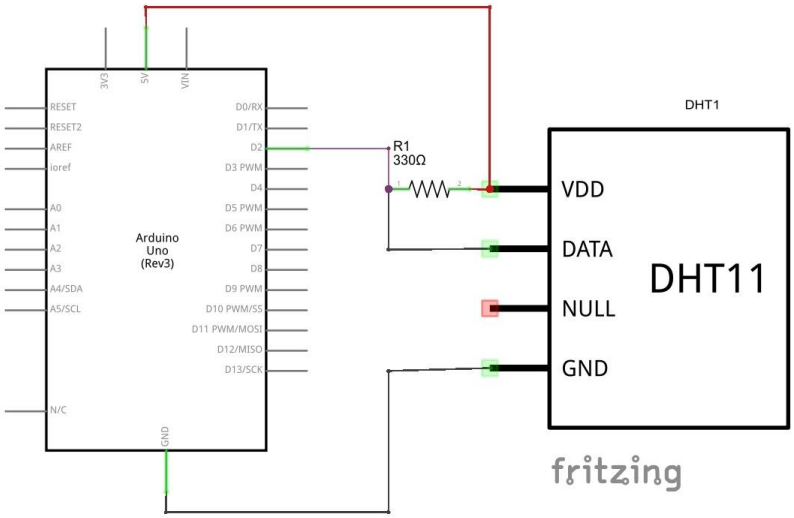


Fig. 14 Diagrama de Conexión

Fuente: Elaboracion propia

La biblioteca de lectura del sensor DHT11 funciona declarando un objeto global del tipo dht (en el ejemplo la variable se llama DHT). El método read11() recibe como argumento el número de pin (terminal) de Arduino donde se encuentra conectado el sensor. Devuelve el resultado de la operación en forma de un entero. Si la conexión fue exitosa, las propiedades “humidity” y “temperature” contendrán los valores medidos de humedad y temperatura respectivamente.

2.6.3. Sensor de Humo y Gas

2.6.3.1. Introducción

El sensor de gas analógico y humo (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 GLP, i-butano, propano, metano, alcohol, hidrógeno, tiene una alta sensibilidad, un tiempo de respuesta rápido 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 Microcontroladores. 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.

La estructura y configuración de MQ-2 sensor de gas, donde este sensor está 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.

Este es un sensor muy sencillo de usar, ideal para medir concentraciones de gas natural en el aire.

El módulo posee una salida analógica que proviene del divisor de voltaje que forma el sensor y una resistencia de carga. También tiene una salida digital que se calibra con un potenciómetro, esta salida tiene un Led indicador.

2.6.3.2. Sus características

Este sensor detecta las concentraciones de gas combustible en el aire y también de humo, ofrece como lectura una tensión analógica. El sensor puede medir concentraciones de gas inflamable de 300 a 10.000 sensor ppm. Puede operar a temperaturas de 20 a 50 ° C y consume menos de 150 mA a 5 V.

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

A continuación se muestra las típicas características de sensibilidad del MQ-2 durante varios gases y su lectura:

- Temperatura: 20 ℃,
- Humedad: 65%,
- Concentración de O2 21%
- RL : 5kΩ
- Ro: resistencia del sensor a 1000 ppm
- H2: en el aire limpio.
- Rs: resistencia del sensor en varias concentraciones de gases.

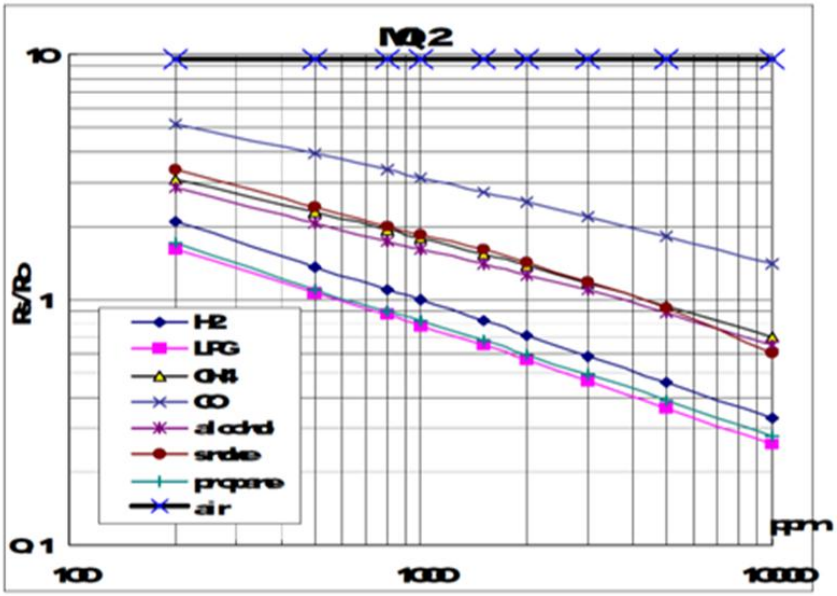


Fig. 15 MQ-2 durante varios gases

Fuente: http://hetpro-store.com/TUTORIALES/wp-content/uploads/2014/08/gas_2.png?c1db43

2.7. INSTALACION DEL SERVIDOR WEB Y FRAMEWORK

2.7.1. Instalar Nginx en Raspbian jessie

Nginx es uno de los servidores web más populares del mundo y es usado por la mayoría de los sitios web con más tráfico de internet. Usa menos recursos que Apache en la mayoría de los casos, y puede ser usado como servidor web o proxy inverso. Lo primero es instalar Nginx en un servidor con Raspbian Jessie, donde puede ser tu propia máquina virtual.

Antes de comenzar, se debe tener en el Raspbian Jessie un usuario normal, un usuario que no sea root y que tenga configurado privilegios de sudo. El siguiente paso es darle al usuario privilegios de sudo. Para ello se va a utilizar el comando "**sudo**".

Ahora se procederá a instalar Nginx de forma fácil ya que el paquete se encuentra disponible en los repositorios oficiales de Raspbian. Como esta es la primera vez que se va a usar el sistema de paquetes "***apt"*** en este, se debe actualizar el índice local de paquetes para tener la información más actualizada posible, empleando lo siguiente:

- *sudo apt-get update*
- *sudo apt-get install nginx*

Seguramente pedirá que se ingrese tu contraseña. Todo el software y sus dependencias serán descargados al servidor e instalado automáticamente.

En Raspbian Jessie, por defecto, Nginx arranca al ser instalado. Se puede acceder a la página por defecto de Nginx para confirmar que el servidor está funcionando de forma correcta visitando el dominio del servidor o IP pública (o local si es una máquina local). Deberías ver una página parecida a:

Fig. 16 Bienvenida en Nginx

Fuente: http://openwebinars.net/media/2014/05/nginx-defecto.png

2.7.2. Instalar Django 1.9 en Raspbian Jessie

La plataforma donde se trabajara en la totalidad del sistema web Django, es un frameworks casi por excelencia para el desarrollo de web-app escrito en Python, se trata de una herramienta gratuita que permitirá mejorar el tiempo de desarrollo que empleamos en la creación del proyecto.

Django es un frameworks de desarrollo web de código abierto escrito en Python, que respeta el patrón de diseño conocida como Modelo – Vista – Controlador (MVC), fue desarrollado en origen para gestionar varias páginas orientadas a noticias de la World Company de Laurense Kansas que se liberó al público con una licencia BCD en Julio 2015.

Para instalar Django Framework y los demás paquetes que son necesarios debemos ingresar los siguientes comandos en nuestra terminal:

- *sudo su*
- *apt-get update*
- *apt-get install python3-pip*
- *pip3 install django==1.9 (en Repositorio se dará todos los paquetes instalados)*

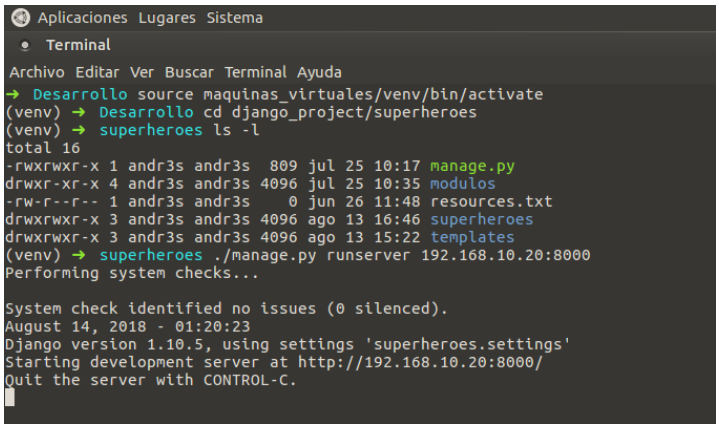


Fig. 17 Versión de Django1.9

Fuente: Elaboracion propia

2.7.3. Instalar Gulp-Angular

Gulp es una herramienta de construcción utilizada por el equipo Angular y muchos desarrolladores mas. La velocidad y la simplicidad se promocionan como beneficios sobre Grunt. Configurar un entorno de desarrollo es muy cómodo con estas herramientas, afortunadamente Gulp resuelve todas las tareas arduas que se realizaban antes.

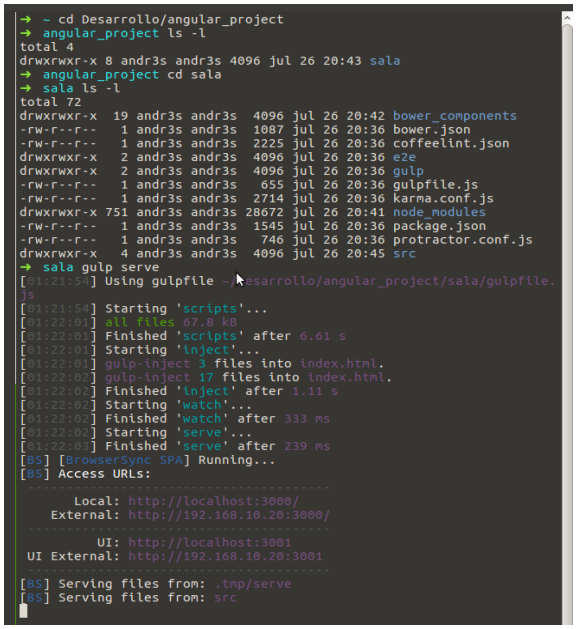


Fig. 18 Correr el entorno en Angular

Fuente: Elaboracion propia

CAPITULO III

INGENIERIA DE PROYECTO

3.1. INTRODUCCION

En esta parte del proyecto se desarrolla una descripción del diseño del sistema de control para la sala de servidores del Ministerio de Obras Públicas Servicios y Viviendas, así como el control de algunos otros artefactos propios del lugar y cámaras de seguridad.

3.2. DESCRIPCIÓN GENERAL DEL SISTEMA

Cumpliendo con el objetivo general del proyecto se realizó una aplicación web que interactúa con los procesos ligados a los Microcontroladores y Raspberry Pi por medio de conexión ethernet y conexión Bluetooth. Para ello se realizó una aplicación web en Python sobre la base del Framework Django REST y Gulp-Angular que actúa como cliente, el cual está encargado de interactuar por medio de comunicación vía web, como servidor.

3.2.1. Raspberry Pi 2

Raspberry Pi 2 es un ordenador de placa reducida, en el cual se alojara las aplicaciones para convertir a este un servidor Web, donde tendrá las aplicaciones que se vayan a crear, plantillas HTML, imágenes, etc.

3.2.2. Arduino

Es una tarjeta de prototipos electrónicos de código abierto (open-source) basada en hardware y software flexibles y fáciles de usar. Es por eso que en esta placa ira conectada el modulo Bluetooth para que se comunique con los distintos sensores, donde se transmitirá los datos y se los enviara vía web a la tarjeta Raspberry Pi2.

3.2.3. Módulo Shield Ethernet

Permite a una placa Arduino conectarse a internet. Está basada en el chip ethernet Wiznet W5100. El Wiznet W5100 provee de una pila de red capaz de comunicarse por los protocolos TCP/UDP. Soporta hasta cuatro conexiones de sockets simultáneas. Este Shield será el interfaz para que se puedan comunicar el Arduino y el Raspberry pi.

3.2.4. Fuente de Alimentación

En la parte de la fuente de alimentación se contara con dos cargadores normales con entrada de 220voltios AC y con dos salidas de 5 voltios DC con una corriente de 5 A (amperios), se distribuirá uno para la tarjeta Arduino y el otro será para el pequeño servidor web Raspberry Pi.

Fig. 19 Cargador para la placa

Fuente: Elaboracion propia

3.3. APLICABILIDAD DEL SISTEMA A LA SALA DE SERVIDORES DEL MINISTERIO.

El diseño y prototipo del sistema se aplicara en la sala de Servidores del Ministerio de Obras Publicas Servicios y Vivienda.

El Ministerio se encuentra ubicado en la Av. Mariscal Santa Cruz, esq. Calle Oruro, Edif. Centro de Comunicaciones N° 1260. El área que pertenece a los ambientes interiores del Ministerio, piso 6.

Fig. 20 Ambientes exteriores e interiores del ministerio

Fuente: Elaboración propia

3.4. SOFTWARE DE CONFUGURACION DEL SISTEMA

En la configuración del sistema se utiliza el frameworks Django REST, una herramienta de desarrollo web para hacer una aplicación basado en lenguaje Python con la ayuda de la herramienta Gulp-Angular, todas estas herramientas estarán configuradas e instaladas en el Raspberry Pi 2, con sistema operativo Raspbian de debian.

3.4.1. Framework Django REST como herramienta de desarrollo web

Django REST es un frameworks diseñado principalmente para crear aplicaciones web puramente diseñado en el lenguaje de Python que es lo mejor que puede tener Django, es un frameworks open-source, que quiere decir que su código raíz esta accesible para todos que nos servirá como un servidor REST.

3.4.2. Proceso de configuración y creación de una aplicación web

Para poder trabajar con Framework Django REST lo primero que se debe realizar es instalar la herramienta, para ello se tiene que abrir la terminal (CTR+ALT+t) para los usuarios GNU/Linux aparecerá una consola donde debemos de ingresar los siguientes comandos:

- *sudo apt-get update*
- *sudo apt-get install pip*
- *sudo apt-get install django*
- *sudo apt-get install unipath*
- *sudo apt-get install nginx*
- *sudo service restart nginx*

Con los anteriores comandos ya ingresados se prepara el entorno de trabajo. Lo siguiente que se hace es activar Django con los comandos, que se muestra en la figura 21, el cual se está activando el entorno virtual para que corra Django REST.

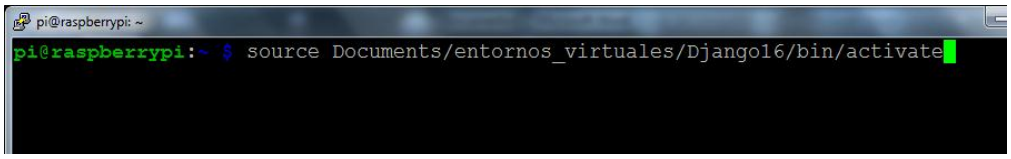


Fig. 21. Activar en torno virtual Django1.9

Fuente: Elaboracion propia

Ahora se creará una aplicación donde se direcciono, donde la dirección de la carpeta será: Documentos/proyectos_django:

- *cd Document/proyectos_django*
- *django-admin startproject primero*
- *cd primero*

Con el comando anterior ya ingresados se debe ver si se creó correctamente el proyecto en la figura 22:

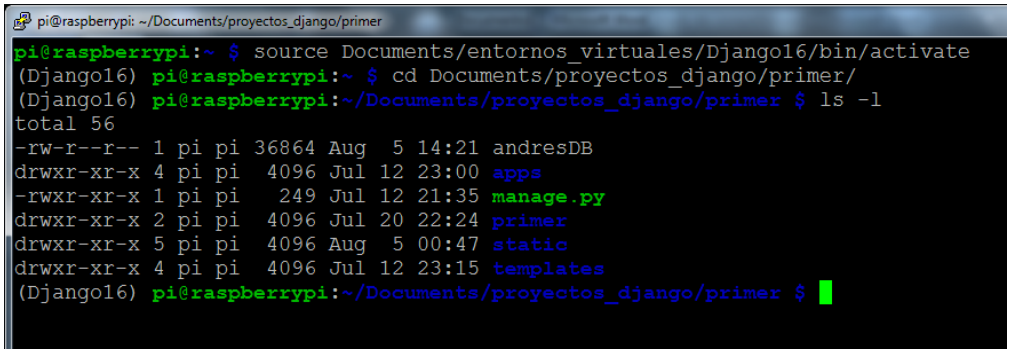


Fig. 22. Comprobar el proyecto creado

Fuente: Elaboracion propia

Si se observa en la figura 22, se notara que la herramienta de Django creo varias carpetas que utilizará el proyecto. El siguiente paso y el último es crear una aplicación dentro de la carpeta ***app***, ingresando los siguientes comandos:

- *cd app/*

- *django-admin startapp inicio*
- *cd ..*

Se debe probar si las instalaciones y creación del proyecto está correctamente hechas, se debe de ingresar a la carpeta principal e ingresar los comandos siguientes indicando el número IP del Raspberry y el puerto por donde se comunicara por el mundo exterior por defecto ya vienen en el puerto ***8000***, que se muestran en la figura 23.

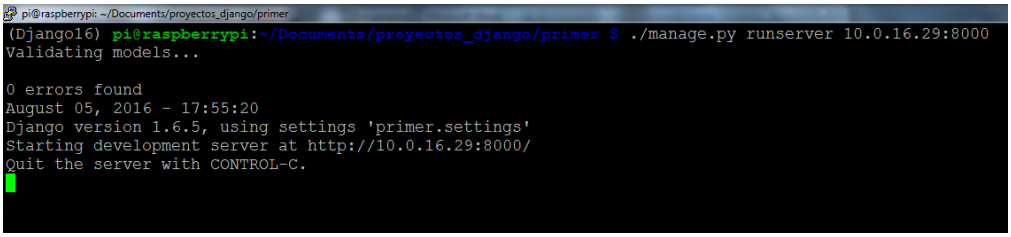


Fig. 23. Arrancando el proyecto

Entorno de desarrollo Django

3.5. FIRMWARE DEL SISTEMA

El Raspberry pi 2, tiene un sistema de seguridad junto con la herramienta de Django, en el cual solo los usuarios creados en una base de datos podrán acceder al sistema, se le llama ***USER_AUTHENTICATION.*** Mientras que en la tarjeta de Arduino ocurre otra cosa, donde los dispositivos se comunican de forma inalámbricamente mediante comunicación Bluetooth entre ellos y la tarjeta Raspberry. Realizara estas tareas para cumplir con los objetivos planteados:

- El Raspberry Pi2, se debe comunicar con el usuario final de forma segura mediante la red de internet de forma local.

- Una vez establecida la comunicación entre los dos dispositivos, el Arduino debe programar al módulo mediante comandos AT para la sincronización con los módulos especificados.

3.6. HARDWARE DEL SISTEMA

El hardware del sistema contiene el Raspberry Pi 2, Arduino UNO, salidas digitales, entradas analógicas, módulos Bluetooth, distintos sensores, fuente de alimentación del sistema (cargadores).

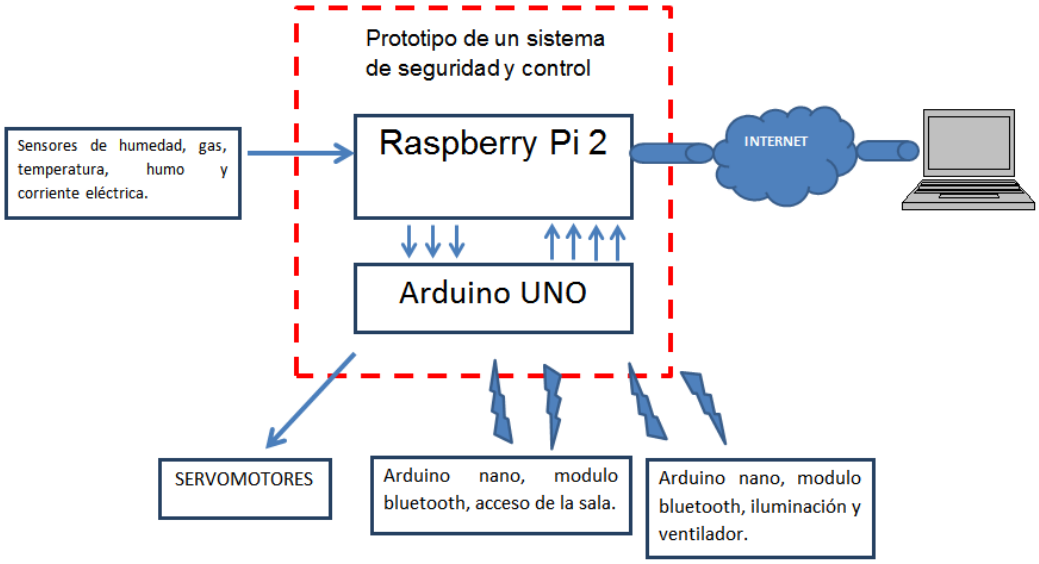


Fig. 24. Diagrama de Bloques del Hardware

Fuente: Elaboracion propia

3.6.1. Conexión del Raspbery Pi 2 y el Arduino UNO

Las dos tarjetas estaran dentro de una caja comunicandose entre si; mandandose informacion, donde la placa del Arduino junto conel Shield Ethernet le mandara los datos de los sensores en formato JSON via web donde el RPi lo gusrdsrs en una base de datos, como se le ve en la figura 25.

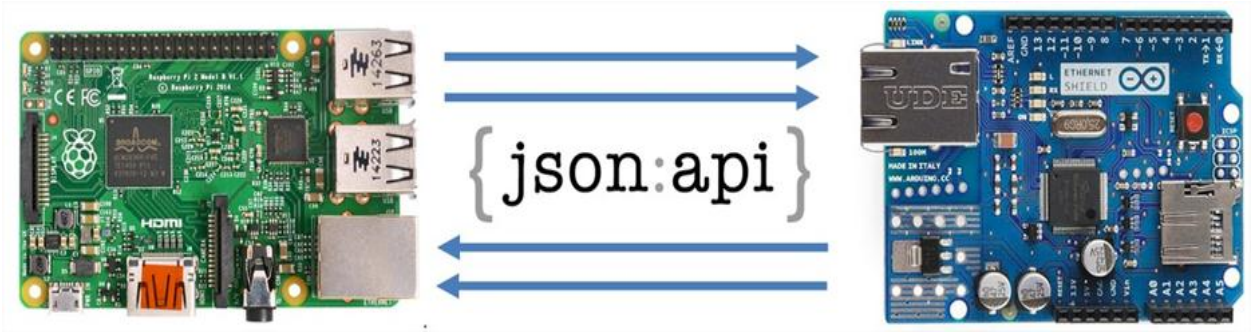


Figura 25. Interfaz de cumunicacion entre placas

Fuente: Adaptacion de esquema

3.6.2. Salidas Digitales/Analogicas del Arduino

En la figura 26 se ve el esquema de circuiteria del arduino, con sus respectivos sensores y sevomotores que controlara el movimiento de la camara, tambien se puede observar que solo se esta utilizando los puertos libres que se tiene, ya que los demas puertos se estan utilizando con la comunicación del Raspberry Pi2.

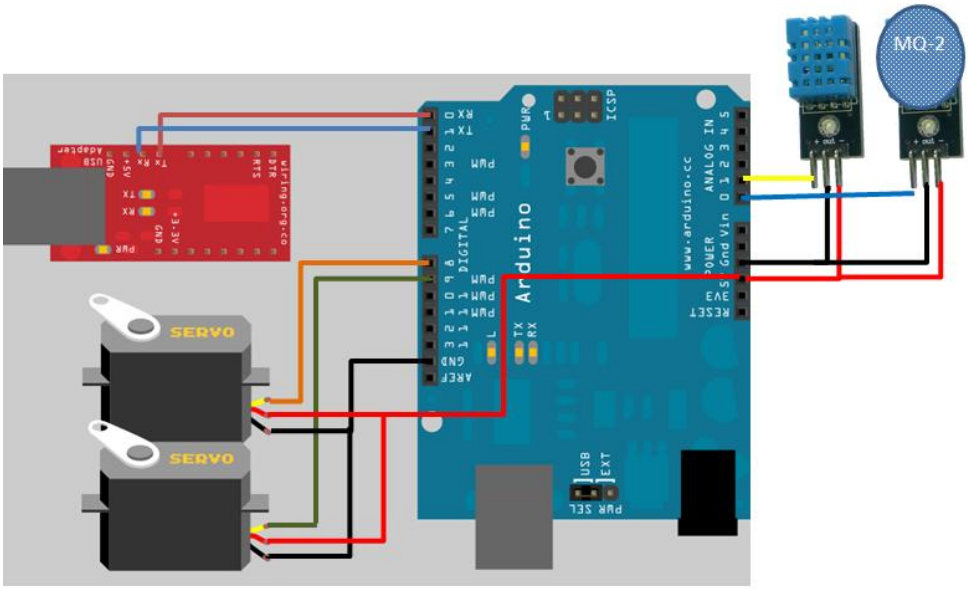


Fig. 26 Circuito del Arduino UNO y sus dispositivos

Fuente: Adaptacion de esquema.

En la figura 26, tambien podemos observar conectados los sensores de tempertatura, humo, humedad y gas para que desde ahí le pase la informacion al raspberry el mismo que lo muestre via web la informaciòn y el estado de los sensores.

3.6.3. Salidas Digitales del Arduino Nano y Bluetooth

Aquí se vera el diagrama de circuito que se hace para el arduino nano, que estarà comunicado inalambricamente con el otro arduino que esta al lado de la tarjeta Raspberry. Este circuito hara caso al arduino, para que pueda mover los sevomotores y tabien a los Relay que estan conectados, para que conmuten. La tarea que realiza es encender/apagar la iluminacion, el ventilador y controlar el acceso de la entrada de la sala de servidores.

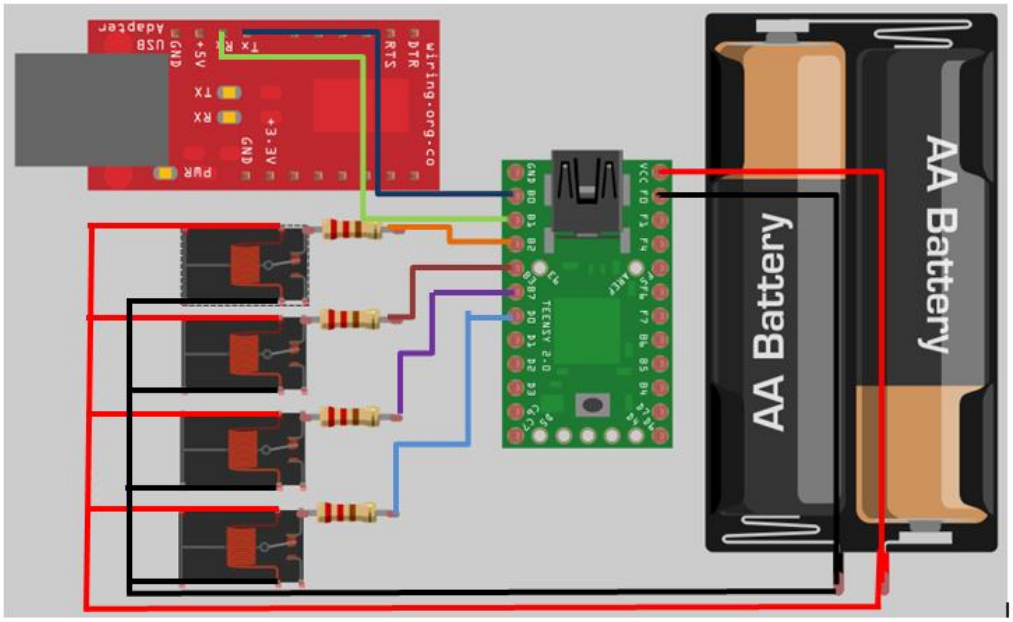


Fig 27. Circuito remoto con el modulo Bluetooth

Fuente: Adaptaciòn de esquema.

Si se observa la figura 27, se nota que se encuentra conectado a una bateria externa que se recomienda que suministre una alimentacion de 5 Voltios para adelante con un valor de corriente de 1A como minimo.

3.7. DIAGRAMA COMPLETO DEL SISTEMA

3.7.1. Circuito controlador

En el diagrama se observa la parte central del circuito completo. Las conexiones son del Raspberry y el Arduino Uno.

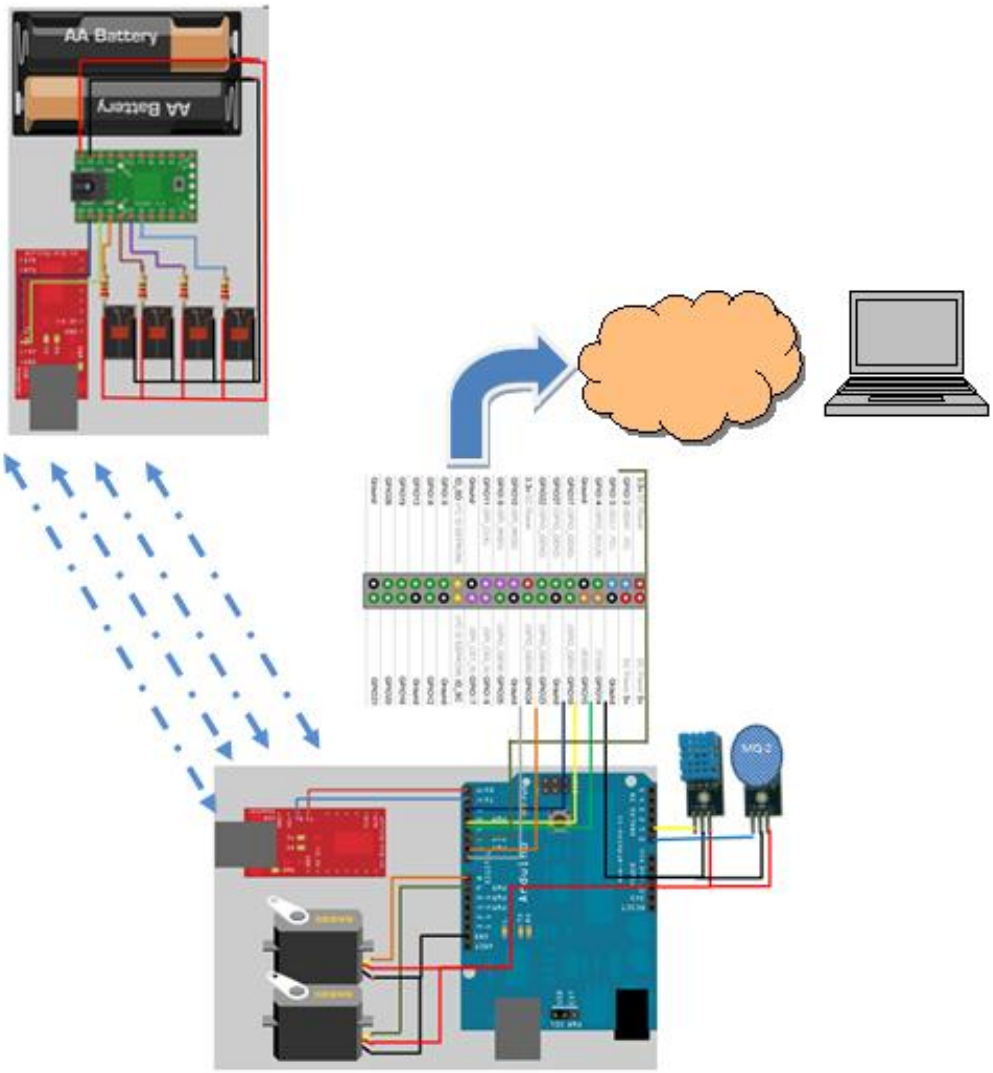


Fig 28. Circuito controlador

Fuente: Adaptaciòn de esquema.

El circuito en resumen se puede ver en la figura 28, donde el usuario que controlarà la sala

de sevidores sera la persona que estara ubicado remotamente en una red interna o externa, por lo que se comunicarà con el Raspberry Pi2, el cual a su vez estarà comunicado con el Arduino Uno testeando los GPIOS digitales, el Arduino Uno tambien tendra la tarea de controlar de forma inalambrica al otro Arduino Nano, el codigo del Rapsberry Pi se muestra en el Anexo C y el codigo de las tarjetas arduinos se encuentra en el Anexo D.

3.8. PRUEBAS Y RESULTADOS DE FUNCIONAMIENTO DEL SISTEMA

En esta sección se presentan las imágenes de la implementación física del sistema y algunos detalles relevantes al respecto. En el documento se encuentra el análisis de los resultados en base a pruebas de funcionamiento del sistema.

3.8.1. Implementacion fisica del sistema

Se observa la implementación física del sistema y el módulo de control, tanbien el sistema de alimentación y las 8 salidas digitales del Raspberry Pi2 están implementados dentro del paquete y unidos por al Arduino UNO.

Fig. 29 Diseño del Case de las placas

Fuente: Elaboracion propia

En la figura 29, se diseño una caja de modo que proteja y oculte los cables que se

colocaran entre el Raspberry Pi 2 y el Arduino.

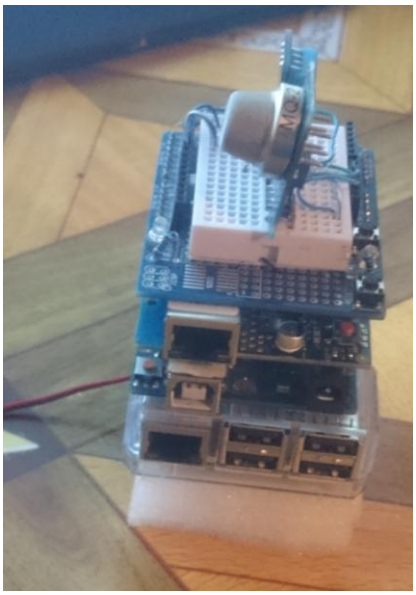


Fig 30. Conexión de Raspberry Pi2 y el Arduino

Fuente: Elaboracion Propia

La integracion de los dos dispositivos para que se comuniquen entre si junto con el modulo Bluetooth que tambien se comunicara de forma inalambricamente con otro arduino nano que estara colocado en un lugar estrategico.

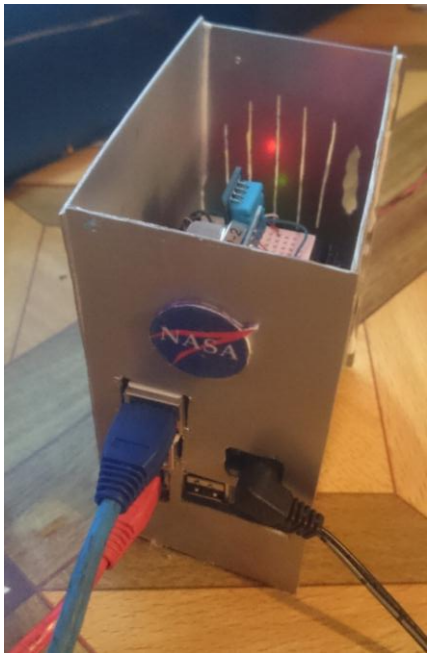


Fig 31. Sistema completo de prueba

Fuente: Elaboracion Propia

Fig 32. Vista superior e inferior del Hardware

Fuente: Elaboracion propia

Se aconseja que en la parte inferior del Raspberry Pi 2 se coloque un ventilador para que el dispositivo no caliente y asi funciones correctamente, recordando que el Raspberry es una pequeña PC.

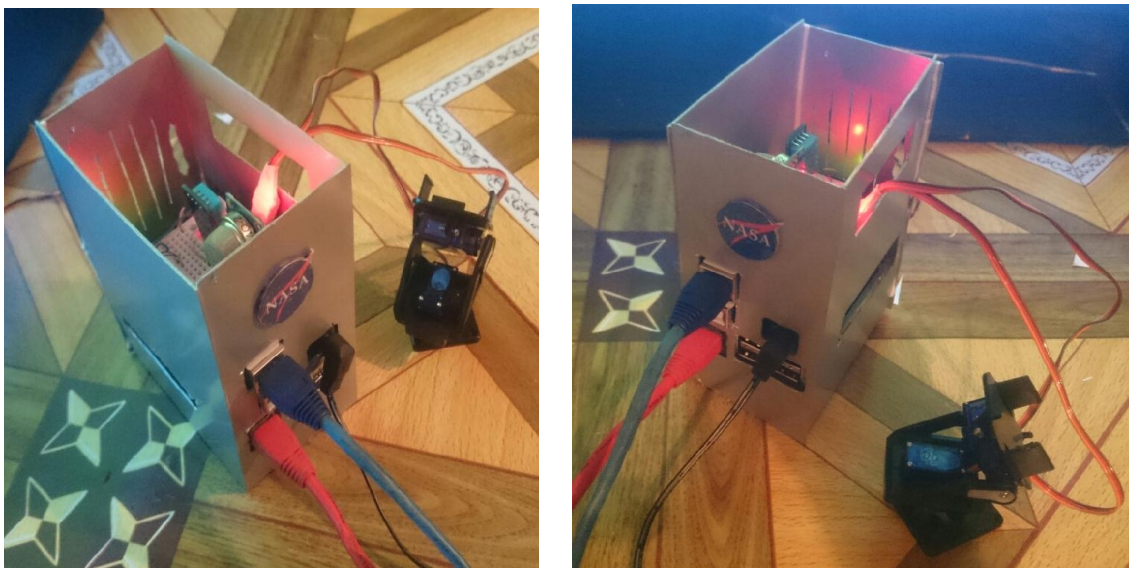


Fig 33. Conexión de los servomotores

Fuente: Elaboracion propia

La conexión de los servomotores tiene la finalidad de que una camara de seguridad pueda moverse en su eje X y en su eje Y, controlado via web.

3.8.2. Prueba del sistema via web

Se probarà el producto final via web, donde lo primero que se debe hacer es; ingresar a al url : ***[ip del raspberry]:8000,*** este es la direccion del Raspberry PI:

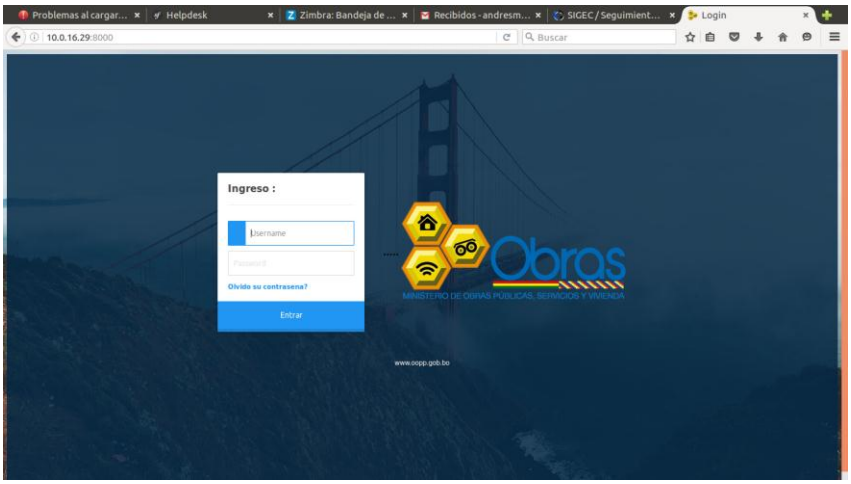


Fig. 34 plantilla de logeo del sistema

Fuente: elaboracion propia

Luego de ingresar el nombre de usuario y la contraseña en los campos de texto, se va a "Ingresar", de donde se dirige a la pagina de inicio y control:

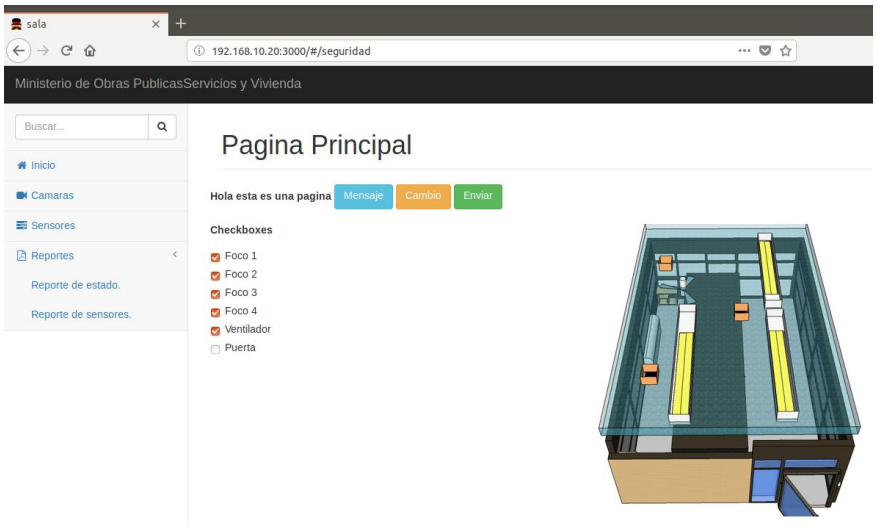


Fig 35. Pagina de Inicio y control del sistema

Fuente: Elaboracion propia

En la figura 35 se puede apreciar la pagina principal donde se podra controlar toda la sala de servidores y testear sus respectivos sensores.

Fig 36. Imagen de la sala de servidores y los diferentes botones de control

Fuente: Elaboracion propia

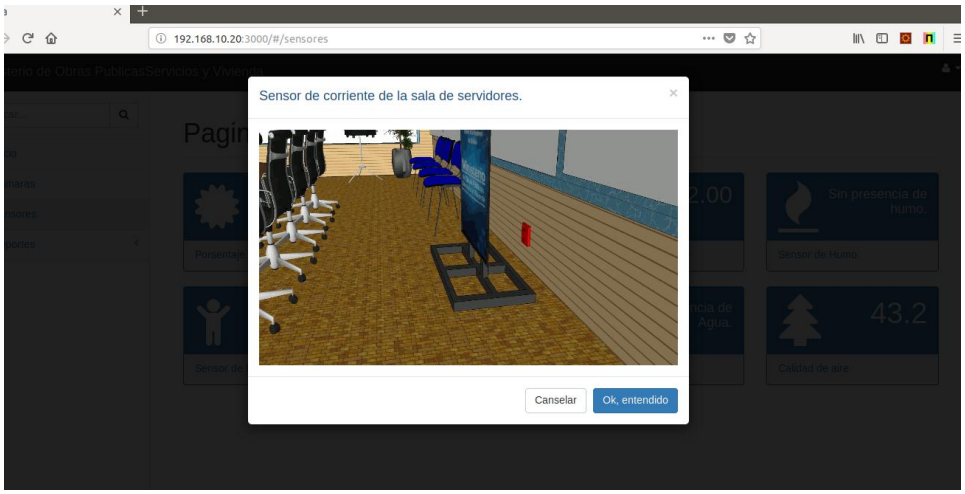


Fig 37. Aviso de falla en la sala de servidores

Fuente: Elaboracion propia

En la figura 37, se onserva el mensaje que muestra la pagina, si los sensores presentan una falla, la pagina web tambièn mostrara una imagen animada donde especifique la falla del sistema.

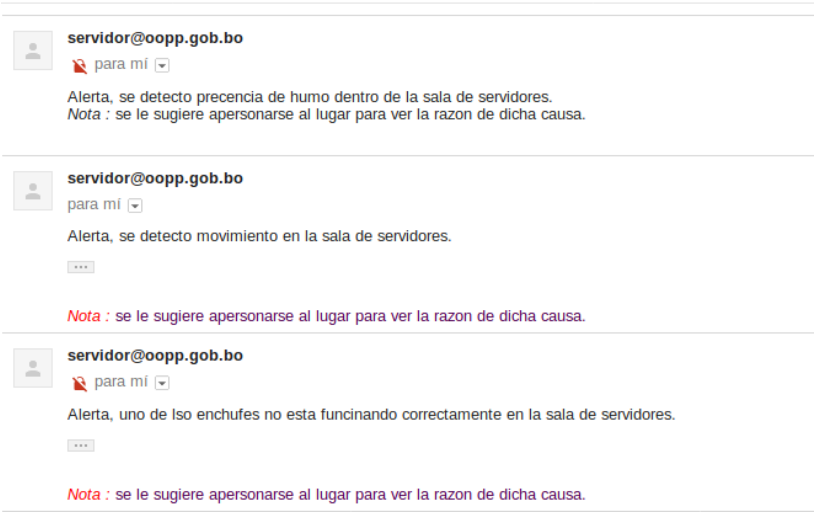


Fig 38. Correo de alerta

Fuente: Elaboracion propia

El sistema tambien envia correos electronicos de alerta a todo el area de sistema del Ministerio, como se ve la figura 38, el sistema envio un correo electronico al personal

alertando la elevacion de temperatura de la sala de servidores y recomendando una supervicion inmediata.

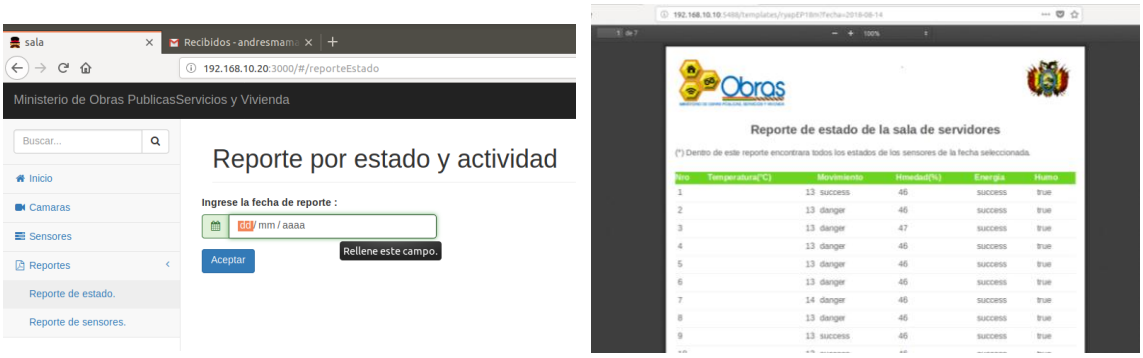


Fig 39. Reportes

Fuente: Elaboracion propia

En la Figura 39 se puede ver el reporte que es posible imprimir el estado del circuito actuador como el de sensores por fecha, gracias al servidor Jsreport que se encuentra alojado en otro servidor.

CAPITULO IV

ANALISIS DE COSTOS

4.1 ANALISIS DE COSTOS

En esta sección se encuentra el análisis de costos del proyecto. Este se divide en el costo del desarrollo de software y el costo de construcción del hardware del prototipo.

4.1.1. Costo de Diseño Software

El análisis de costo de diseño del software se emplea el modelo constructivo de coste empírico. Según Boehm, quien es el desarrollador y está definido para tres tipos de proyectos: Orgánico, semi-acoplado y empotrado.

Las ecuaciones son las siguientes:

$$E = a_b * KLDC^{b}_{b} * FAE$$

$$D = C_b * E^{d}_{b}$$

$$N = E / D$$

Dónde:

E= Esfuerzo aplicado en personas-mes

KLDC=Número de líneas de código (en miles)

FAE=Factor de ajuste del esfuerzo

D=Tiempo de desarrollo (en meses)

N=Número de personas necesarias para el desarrollo del software

ab, bb , cb , db = Son coeficientes extraídos de la tabla 1

Proyecto de Software	ab	bb	cb	db
Orgánico	3,2	1,05	2,5	0,38
Semi-orgánico	3	1,12	2,5	0,35
Empotrado	2,8	1,2	2,5	0,32

Tabla. 1 coeficientes ab, bb , cb , db según el tipo de proyecto

El Factor de ajuste del esfuerzo FAE se determina multiplicando 15 factores extraídos de la tabla 2.

FAE = Fr * Tbd * Cp * Rt * Ra * Vmv * Tr * Ca * Ea * Cpr * Eso * El * Pm * Us * L

Conductores de coste	Valoración					
	Muy bajo	Bajo	Nominal	Alto	Muy alto	Extr. Alto
Fiabilidad requerida del software(Fr)	0,75	0,88	1,00	1,15	1,40	-
Tamaño de la base de datos(Tbd)	-	0,94	1,00	1,08	1,16	-
Complejidad del producto (Cp)	0,70	0,85	1,00	1,15	1,30	1,65
Restricciones del tiempo de ejecución (Rt)	-	-	1,00	1,11	1,30	1,66
Restricciones del almacenamiento principal(Ra)	-	-	1,00	1,06	1,21	1,56
Volatilidad de la maquina virtual(Vmv)	-	0,87	1,00	1,15	1,30	-
Tiempo de respuesta del ordenador(Tr)	-	0,87	1,00	1,07	1,15	-
Capacidad del analista(Ca)	1,46	1,19	1,00	0,86	0,71	-
Experiencia en la aplicación(Ea)	1,29	1,13	1,00	0,91	0,82	-
Capacidad de los programadores(Cpr)	1,42	1,17	1,00	0,86	0,70	-
Experiencia en S.O. utilizado(Eso)	1,21	1,10	1,00	0,90	-	-
Experiencia en el lenguaje de programación(El)	1,14	1,07	1,00	0,95	-	-
Prácticas de programación modernas(Pm)	1,24	1,10	1,00	0,91	0,82	-
Utilización de herramientas software(Us)	1,24	1,10	1,00	0,91	0,83	-
Limitaciones de planificación del proyecto(L)	1,23	1,08	1,00	1,04	1,10	-

Tabla 2. Valoración de los conductores de coste para determinar la FAE.

4.1.1.1. Análisis de costo de diseño de Software

Se considera como un proyecto de software semi-acoplado para el análisis de costo.

FAE = 1 * 0.94 * 1 * 1 * 1 * 1 * 1.15 * 1 *0.91 * 1 * 1 * 1 * 1.1 * 1 * 1.08

FAE = 1.16

$$KLCD = \frac{420}{1000}$$
$$KLCD = 0.420$$

$$\mathrm{E} = 3 * 0.420^{1.12} * 1.16$$
$$\mathrm{E} = 1.31\ [personas - mes]$$

$$\mathrm{D} = 2.5 * 1.31^{0.35}$$
$$\mathrm{D} = 2.74\ [meses]$$
$$\mathrm{N} = \frac{1.31}{2.74} = 0.47 \approx 1[personas]$$

Para determinar el costo del software:

$$C_s = E * I_p$$

Dónde:

E=Proviene de las ecuaciones de Cocomo

Ip= Remuneración mensual de un profesional inicial en el área.

$$Cs = 1.31 * 4500 = 5895\ Bs$$

4.1.2. Costo en la Construcción del Hardware

En las tablas se muestra la lista de componentes empleados, precios unitarios y cantidades, además de elementos adicionales en la construcción.

Nro.	Componente	Valor / Modelo	Precio unitario(Bs)	Cantidad	Precio Total(Bs)
1	Raspberry Pi 2	Pi 2	400	1	400
2	Arduino UNO	UNO	70	1	70
3	Arduino Nano	Nano	35	2	70
4	Ethernet Shield	W5100	50	1	50
5	Sensor de humo y Gas	MQ2	20	1	20
6	Sensor de Temperatura y Humedad	DHT11	20	1	20
7	Relay de 2 canales	Relay 5V	20	.4	80
8	Modulo Bluetooth	HC-05	50	2	100
9	Sensor de Movimiento	Sensor PIR	25	2	100
10	Servomotores	Motor Micro Servo SG90	25	2	50
11	Resistores	330Ω / 1/4W	0.30	10	3
12	Resistores	1kΩ / 1/4W	0.30	5	1.50
13	Resistores	10kΩ / 1/4W	0.30	5	1.50
14	Cargadores	5v, 2 A DC	30	2	60
15	Otros	Cables, pasta para soldar, estaño	30	1	30
TOTAL					1056

Tabla 3. Costo total de los componentes del Proyecto

El costo total de los componentes y elementos que intervienen en la construcción del prototipo, más los módulos externos para realizar pruebas es de 1056 Bs.

4.1.3. Costo de licencias de Programas empleados

En esta sección de costo de licencias empleadas, no se considera un costo por que todos los programas, frameworks, lenguajes, base de datos, sistema operativo y dispositivos son Open Source y no se paga licencia.

4.1.4. Costo Final del Proyecto

El costo final se determinara mediante la siguiente ecuación:

CFp = Costo total diseño del software + Costo total de los componentes + Costo total de licencias.

$$CFp = 5895 + 1056 + 0$$

$$CFp = 6951 \text{ Bs}$$

Es posible aclarar que el costo final del proyecto es para un prototipo con fines académicos.

CAPITULO V

CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

Luego de realizar el diseño, construcción y pruebas de funcionamiento con el prototipo del sistema y el análisis económico, se destacan las siguientes conclusiones:

- Es posible acceder a la tarjeta Raspberry Pi 2, mediante el protocolo de seguridad SCP para poder administrarlo remotamente.
- La tarjeta Raspberry Pi 2, puede lograr una comunicación con la tarjeta Arduino, utilizando los puertos digitales del Arduino y los GPIOS del Raspberry Pi 2.
- Si bien el costo de desarrollo del sistema como prototipo es elevado, la reproducción del mismo demanda un esfuerzo menor muy significativo y proporcional a la disminución del costo final planteado.

5.2 RECOMENDACIONES

Al momento de conectar la tarjeta Raspberry pi2 con la otra tarjeta Arduino UNO, se debe tener en cuenta que la tierra va conectada a ambas en común, solo en el caso de que las placas Arduinos se comunique mediante Bluetooth, no va conectado la tierra en común, ya que es una comunicación inalámbrica requiere dos líneas Tx (transmisión) y Rx (Recepción).

Se debe tomar en cuenta que la tarjeta del RBP trabaja con un mínimo de 5 (Voltios) y máximo 5.2 (Voltios), con una corriente mínima de 1(A) a medida que se coloca más

dispositivos en sus puertos USB se debe de aumentar la fuente de corriente.

Se recomienda colocar un pequeño ventilador a la placa Raspberry Pi 2, ya que con varios procesos que realizara dicha tarjeta los microprocesadores que tienen en su placa se calienta, de esta forma es muy recomendable colocar ventiladores o enfriadores a la tarjeta, de igual forma al Shield Ethernet que esta conectadlo al Arduino UNO.

REFERENCIAS BIBLIOGRAFICA

- Edgar Leytn Q. (2007). Diseño de una instalación domótica en un condominio para el control de seguridad e iluminación mediante la tecnología LonWOrks.
- Mauricio Ramiro Heriquez Schott. (2005). Investigación y desarrollo sistema prototipo de asistencia domótica para personas con movilidad limitada.
- Jaime Boal Martín-Larrauri. (2013). Sistema domótica para una casa inteligente.
- Víctor Alberto Gómez Flores. (2014). Sistemas de control de Control de iluminación con protocolo de control domótica estandarizado**.**
- Liliana Raquel Castillo (2008), Diseño de infraestructura de telecomunicaciones para un data center.
- Brian W. Evans (2011), Arduino Programming Notebook
- Emilio Lledo Sanchez (2012), Diseño de un sistema de control domótica basado en la plataforma de Arduino.
- Ing. Rene Alfonso Carrillo Flores (2015), Diseño e implementación de un prototipo (DOMSYSTEM) de seguridad y control para mantener el resguardo de bienes y el confort mediante una red de sensores utilizando comunicación Wireless Bluethooth.

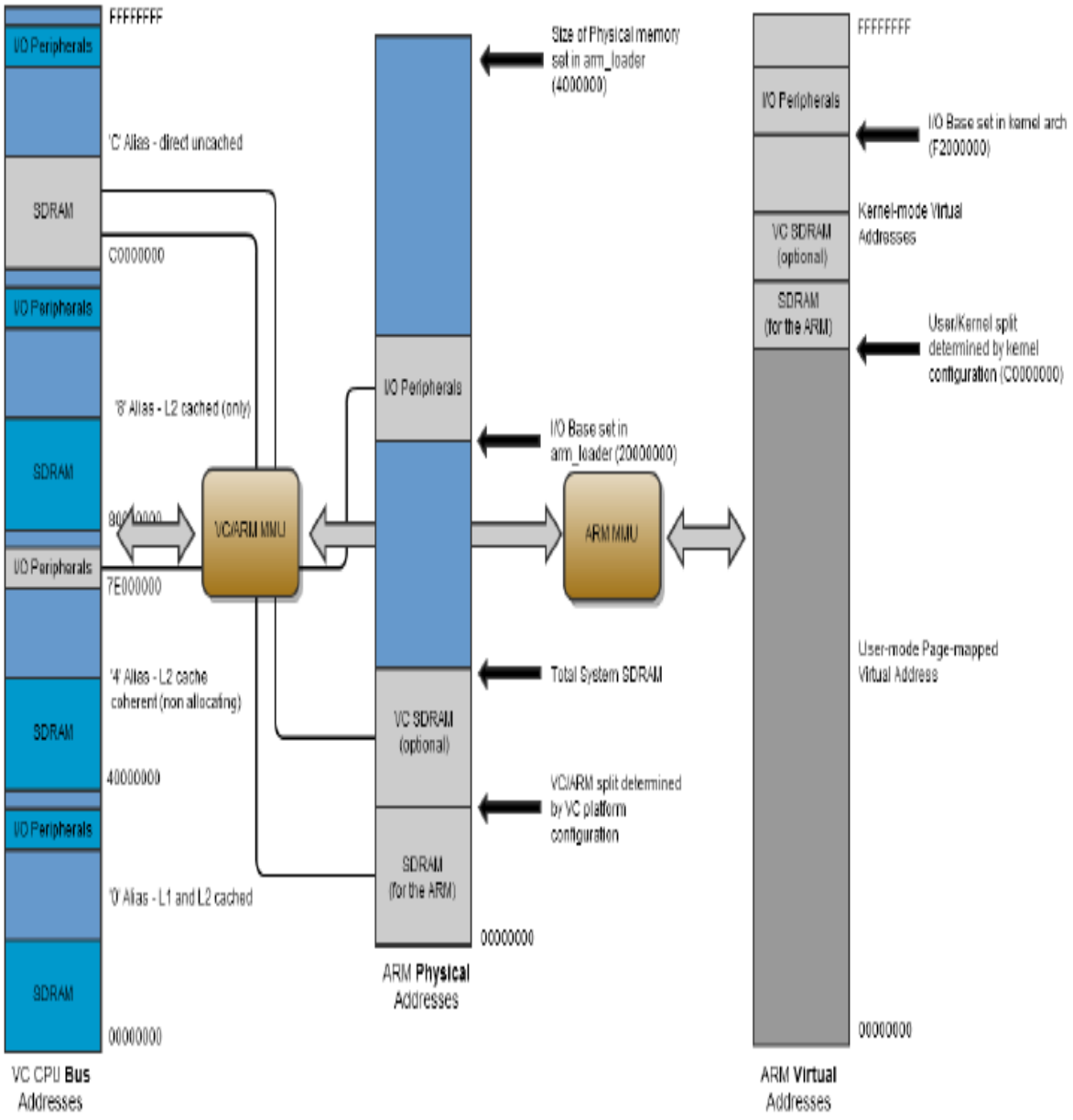
Páginas Web:

http://blog.bricogeek.com/noticias/arduino/descargar-manual-en-pdf-de-arduino-en-espanol/
http://www.prometec.net/modulo-1/ http://www.cienciaaldia.files.wordpress.com.
http://www.quiminet.com/articulos/que-es-la-automatizacion-27058.htm
http://www.prometec.net/shield-ethernet/ http://www.prometec.net/bt-hc06/
http://www.guillenxt.com/2013/01/coleccion-de-libros-arduino-y-processing.html
http://www.dell.com/bo/empresas/p/servers http://www.dell.com/bo/empresas/p/enterprise-products http://www.lr21.com.uy/comunidad/1288818-antel-inauguro-el-data-center-mas-moderno-de-america-latina-donde-google-y-netflix-alojaran-sus-datos
http://www.trucosnoticia.com/antel-inauguro-el-data-center-mas-moderno-de-america-latina-donde-google-y-netflix-alojaran-sus-datos/
http://www.monografias.com/trabajos76/automatizacion/automatizacion.shtml

ANEXO A

Especificaciones Broadcom BCM2835 SoC (Raspberry Pi 2)

BCM2835 ARM Peripherals



8.1 Block Diagram

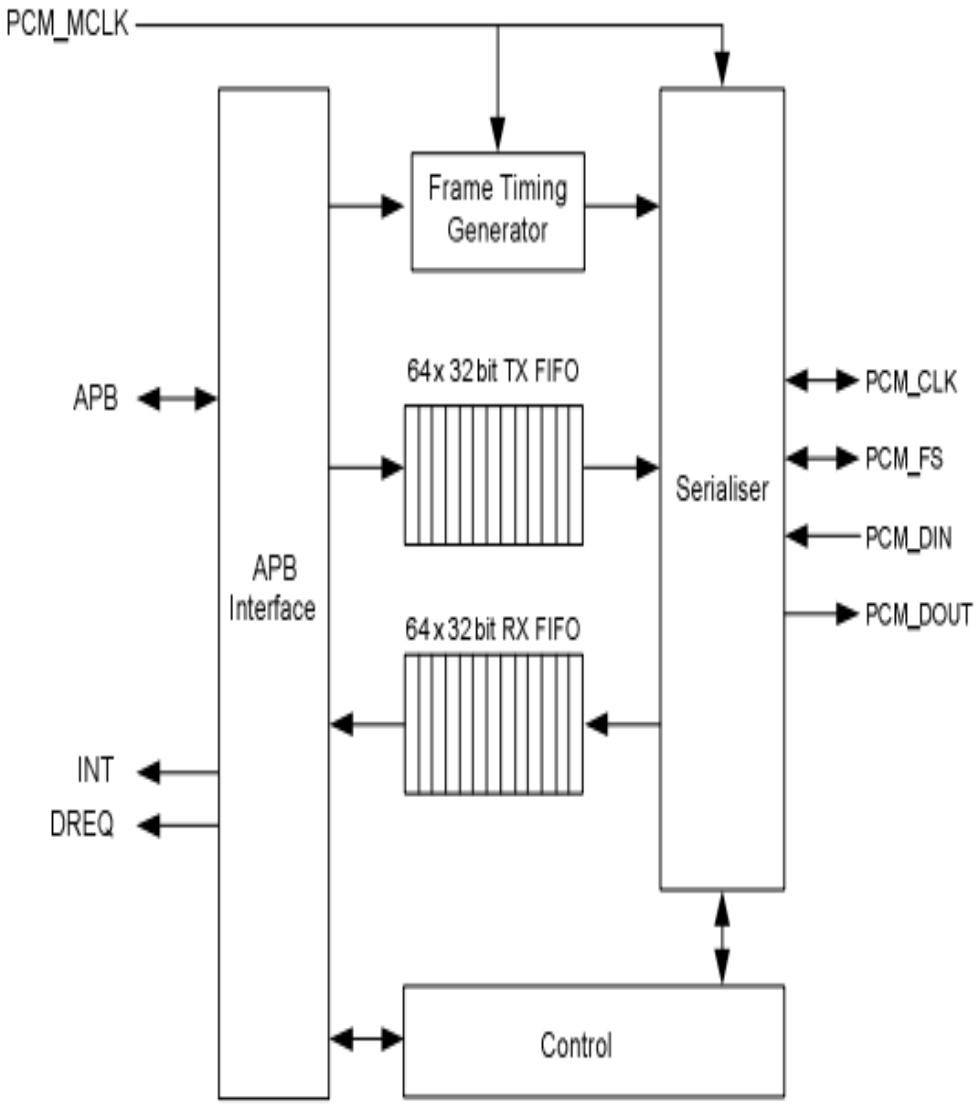
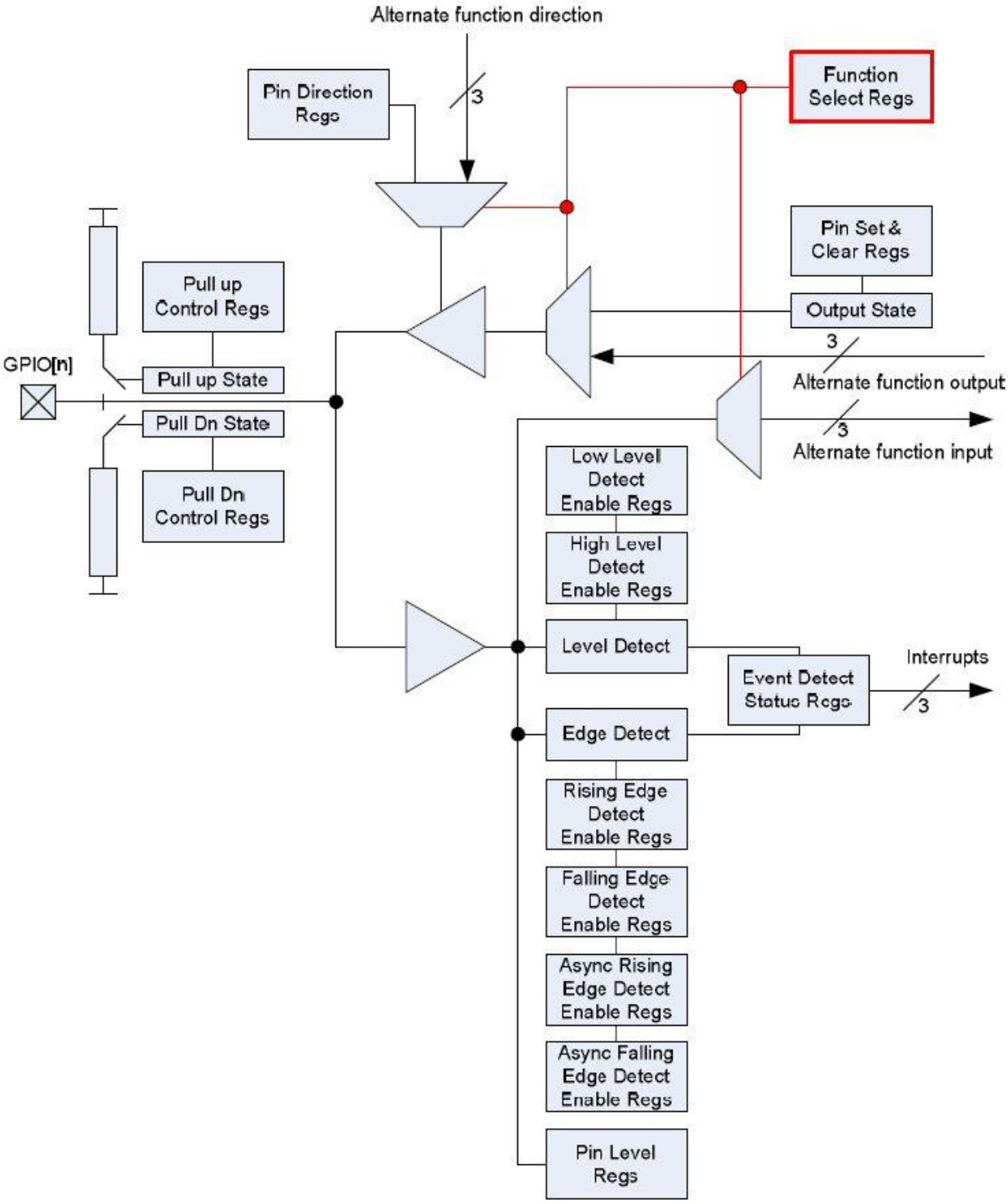


Figure 8-2 PCM Audio Interface Block Diagram


Alternate function direction
Pin Direction Regs
3
Function Select Regs
Pin Set & Clear Regs
Output State
3
Alternate function output
3
Alternate function input
Pull up Control Regs
GPIO[n]
Pull up State
Pull Dn State
Pull Dn Control Regs
Low Level Detect Enable Regs
High Level Detect Enable Regs
Level Detect
Event Detect Status Regs
Interrupts
3
Edge Detect
Rising Edge Detect Enable Regs
Falling Edge Detect Enable Regs
Async Rising Edge Detect Enable Regs
Async Falling Edge Detect Enable Regs
Pin Level Regs

2 Auxiliaries: UART1 & SPI1, SPI2

2.1 Overview

The Device has three Auxiliary peripherals: One mini UART and two SPI masters. These three peripheral are grouped together as they share the same area in the peripheral register map and they share a common interrupt. Also all three are controlled by the auxiliary enable register.

Auxiliary peripherals Register Map (offset = 0x7E21 5000)

Address	Register Name[3]	Description	Size
0x7E21 5000	AUX_IRQ	Auxiliary Interrupt status	3
0x7E21 5004	AUX_ENABLES	Auxiliary enables	3
0x7E21 5040	AUX_MU_IO_REG	Mini Uart I/O Data	8
0x7E21 5044	AUX_MU_IER_REG	Mini Uart Interrupt Enable	8
0x7E21 5048	AUX_MU_IIR_REG	Mini Uart Interrupt Identify	8
0x7E21 504C	AUX_MU_LCR_REG	Mini Uart Line Control	8
0x7E21 5050	AUX_MU_MCR_REG	Mini Uart Modem Control	8
0x7E21 5054	AUX_MU_LSR_REG	Mini Uart Line Status	8
0x7E21 5058	AUX_MU_MSR_REG	Mini Uart Modem Status	8
0x7E21 505C	AUX_MU_SCRATCH	Mini Uart Scratch	8
0x7E21 5060	AUX_MU_CNTL_REG	Mini Uart Extra Control	8
0x7E21 5064	AUX_MU_STAT_REG	Mini Uart Extra Status	32
0x7E21 5068	AUX_MU_BAUD_REG	Mini Uart Baudrate	16
0x7E21 5080	AUX_SPI0_CNTL0_REG	SPI 1 Control register 0	32
0x7E21 5084	AUX_SPI0_CNTL1_REG	SPI 1 Control register 1	8
0x7E21 5088	AUX_SPI0_STAT_REG	SPI 1 Status	32
0x7E21 5090	AUX_SPI0_IO_REG	SPI 1 Data	32
0x7E21 5094	AUX_SPI0_PEEK_REG	SPI 1 Peek	16
0x7E21 50C0	AUX_SPI1_CNTL0_REG	SPI 2 Control register 0	32
0x7E21 50C4	AUX_SPI1_CNTL1_REG	SPI 2 Control register 1	8

[3] These register names are identical to the defines in the AUX_IO header file. For programming purposes these names should be used wherever possible.

0x7E21 50C8	AUX_SPI1_STAT_REG	SPI 2 Status	32
0x7E21 50D0	AUX_SPI1_IO_REG	SPI 2 Data	32
0x7E21 50D4	AUX_SPI1_PEEK_REG	SPI 2 Peek	16

2.1.1 AUX registers

There are two Auxiliary registers which control all three devices. One is the interrupt status register, the second is the Auxiliary enable register. The Auxiliary IRQ status register can help to hierarchically determine the source of an interrupt.

AUXIRQ Register (0x7E21 5000)

SYNOPSIS The AUXIRQ register is used to check any pending interrupts which may be asserted by the three Auxiliary sub blocks.

Bit(s)	Field Name	Description	Type	Reset
31:3		Reserved, write zero, read as don't care		
2	SPI 2 IRQ	If set the SPI 2 module has an interrupt pending.	R	0
1	SPI 1 IRQ	If set the SPI1 module has an interrupt pending.	R	0
0	Mini UART IRQ	If set the mini UART has an interrupt pending.	R	0

AUXENB Register (0x7E21 5004)

SYNOPSIS The AUXENB register is used to enable the three modules; UART, SPI1, SPI2.

Bit(s)	Field Name	Description	Type	Reset
31:3		Reserved, write zero, read as don't care		
2	SPI2 enable	If set the SPI 2 module is enabled. If clear the SPI 2 module is disabled. That also disables any SPI 2 module register access	R/W	0
1	SPI 1 enable	If set the SPI 1 module is enabled. If clear the SPI 1 module is disabled. That also disables any SPI 1 module register access	R/W	0
0	Mini UART enable	If set the mini UART is enabled. The UART will immediately start receiving data, especially if the UART1_RX line is *low*. If clear the mini UART is disabled. That also disables any mini UART register access	R/W	0

If the enable bits are clear you will have ***no access*** to a peripheral. You can not even read or write the registers!

GPIO pins should be set up first the before enabling the UART. The UART core is build to emulate 16550 behaviour. So when it is enabled any data at the inputs will immediately be received . If the UART1_RX line is low (because the GPIO pins have not been set-up yet) that will be seen as a start bit and the UART will start receiving 0x00-characters.

Valid stops bits are not required for the UART. (See also Implementation details). Hence any bit status is acceptable as stop bit and is only used so there is clean timing start for the next bit.

Looking after a reset: the baudrate will be zero and the system clock will be 250 MHz. So only 2.5 µseconds suffice to fill the receive FIFO. The result will be that the FIFO is full and overflowing in no time flat.

2.2 Mini UART

The mini UART is a secondary low throughput[4] UART intended to be used as a console. It needs to be enabled before it can be used. It is also recommended that the correct GPIO function mode is selected before enabling the mini Uart.

The mini Uart has the following features:

- 7 or 8 bit operation.
- 1 start and 1 stop bit.
- No parities.
- Break generation.
- 8 symbols deep FIFOs for receive and transmit.
- SW controlled RTS, SW readable CTS.
- Auto flow control with programmable FIFO level.
- 16550 *like* registers.
- Baudrate derived from system clock.

This is a ***mini*** UART and it does NOT have the following capabilities:

- Break detection
- Framing errors detection.
- Parity bit
- Receive Time-out interrupt
- DCD, DSR, DTR or RI signals.

The implemented UART is ***not*** a 16650 compatible UART However as far as possible the first 8 control and status registers are laid out ***like*** a 16550 UART. Al 16550 register bits which are not supported can be written but will be ignored and read back as 0. All control bits for simple UART receive/transmit operations are available.

[4] The UART itself has no throughput limitations in fact it can run up to 32 Mega baud. But doing so requires significant CPU involvement as it has shallow FIFOs and no DMA support.

AUX_MU_LCR_REG Register (0x7E21 504C)

SYNOPSIS The AUX_MU_LCR_REG register controls the line data format and gives access to the baudrate register

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7	DLAB access	If set the first to Mini UART register give access the the Baudrate register. During operation this bit must be cleared.	R/W	0
6	Break	If set high the UART1_TX line is pulled low continuously. If held for at least 12 bits times that will indicate a break condition.	R/W	0
5:1		Reserved, write zero, read as don't care *Some of these bits have functions in a 16550 compatible UART but are ignored here*		0
0	data size	If clear the UART works in 7-bit mode If set the UART works in 8-bit mode	R/W	0

AUX_MU_MCR_REG Register (0x7E21 5050)

SYNOPSIS The AUX_MU_MCR_REG register controls the 'modem' signals.

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7:2		Reserved, write zero, read as don't care *Some of these bits have functions in a 16550 compatible UART but are ignored here*		0
1	RTS	If clear the UART1_RTS line is high If set the UART1_RTS line is low This bit is ignored if the RTS is used for auto-flow control. See the Mini Uart Extra Control register description)	R/W	0
0		Reserved, write zero, read as don't care *This bit has a function in a 16550 compatible UART but is ignored here*		0

AUX_MU_LSR_REG Register (0x7E21 5054)

SYNOPSIS The AUX_MU_LSR_REG register shows the data status.

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7		Reserved, write zero, read as don't care *This bit has a function in a 16550 compatible UART but is ignored here*		0
6	Transmitter idle	This bit is set if the transmit FIFO is empty and the transmitter is idle. (Finished shifting out the last bit).	R	1
5	Transmitter empty	This bit is set if the transmit FIFO can accept at least one byte.	R	0
4:2		Reserved, write zero, read as don't care *Some of these bits have functions in a 16550 compatible UART but are ignored here*		0
1	Receiver Overrun	This bit is set if there was a receiver overrun. That is: one or more characters arrived whilst the receive FIFO was full. The newly arrived charters have been discarded. This bit is cleared each time this register is read. To do a non-destructive read of this overrun bit use the Mini Uart Extra Status register.	R/C	0
0	Data ready	This bit is set if the receive FIFO holds at least 1 symbol.	R	0

AUX_MU_MSR_REG Register (0x7E21 5058)

SYNOPSIS The AUX_MU_MSR_REG register shows the 'modem' status.

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7:6		Reserved, write zero, read as don't care *Some of these bits have functions in a 16550 compatible UART but are ignored here*		0
5	CTS status	This bit is the inverse of the UART1_CTS input Thus : If set the UART1_CTS pin is low If clear the UART1_CTS pin is high	R	1
3:0		Reserved, write zero, read as don't care *Some of these bits have functions in a 16550 compatible UART but are ignored here*		0

AUX_MU_SCRATCH Register (0x7E21 505C)

SYNOPSIS The AUX_MU_SCRATCH is a single byte storage.

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7:0	Scratch	One whole byte extra on top of the 134217728 provided by the SDC	R/W	0

AUX_MU_CNTL_REG Register (0x7E21 5060)

SYNOPSIS The AUX_MU_CNTL_REG provides access to some extra useful and nice features not found on a normal 16550 UART .

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7	CTS assert level	This bit allows one to invert the CTS auto flow operation polarity. If set the CTS auto flow assert level is low* If clear the CTS auto flow assert level is high*	R/W	0
6	RTS assert level	This bit allows one to invert the RTS auto flow operation polarity. If set the RTS auto flow assert level is low* If clear the RTS auto flow assert level is high*	R/W	0
5:4	RTS AUTO flow level	These two bits specify at what receiver FIFO level the RTS line is de-asserted in auto-flow mode. 00 : De-assert RTS when the receive FIFO has 3 empty spaces left. 01 : De-assert RTS when the receive FIFO has 2 empty spaces left. 10 : De-assert RTS when the receive FIFO has 1 empty space left. 11 : De-assert RTS when the receive FIFO has 4 empty spaces left.	R/W	0
3	Enable transmit Auto flow-control using CTS	If this bit is set the transmitter will stop if the CTS line is de-asserted. If this bit is clear the transmitter will ignore the status of the CTS line	R/W	0

ANEXO B

El TUTORIAL DE Python

(Autor original: Guido van Rossom
Editor original: Fred L. Drake, Jr)

1. Introducción

Python es un lenguaje de programación poderoso y fácil de aprender. Cuenta con estructuras de datos eficientes y de alto nivel y un enfoque simple pero efectivo a la programación orientada a objetos. La elegante sintaxis de Python y su tipado dinámico, junto con su naturaleza interpretada, hacen de éste un
lenguaje ideal para scripting y desarrollo rápido de aplicaciones en diversas áreas y sobre la mayoría de las plataformas.

El intérprete de Python y la extensa biblioteca estándar están a libre disposición en forma binaria y de código fuente para las principales plataformas desde el sitio web de Python, http://www.python.org/, y puede distribuirse libremente. El mismo sitio contiene también distribuciones y enlaces de muchos módulos libres de Python de terceros, programas y herramientas, y documentación adicional.

El intérprete de Python puede extenderse fácilmente con nuevas funcionalidades y tipos de datos implementados en C o C++ (u otros lenguajes accesibles desde C). Python también puede usarse como un lenguaje de extensiones para aplicaciones personalizables.

Este tutorial introduce de manera informal al lector a los conceptos y características básicas del lenguaje y el sistema de Python. Es bueno tener un intérprete de Python a mano para experimentar, sin embargo todos los ejemplos están aislados, por lo tanto el tutorial puede leerse estando desconectado.

Para una descripción de los objetos y módulos estándar, mira la Referencia de la Biblioteca de Python. El Manual de Referencia de Python provee una definición más formal del lenguaje. Para escribir extensiones en C o C++, lee Extendiendo e Integrando el Intérprete de Python y la Referencia de la API Python/C. Hay
también numerosos libros que tratan a Python en profundidad.

Este tutorial no pretende ser exhaustivo ni tratar cada una de las características, o siquiera las características más usadas. En cambio, introduce la mayoría de las características más notables de Python, y te dará una buena idea del gusto y estilo del lenguaje. Luego de leerlo, serás capaz de leer y escribir módulos y programas en Python, y estarás listo para aprender más de los variados módulos de la
Biblioteca de Python descriptos en la Referencia de la Biblioteca de Python.

2. Números

El intérprete actúa como una simple calculadora; podes ingresar una expresión y este escribirá los valores.
La sintaxis es sencilla: los operadores +, -, * y / funcionan como en la mayoría de los lenguajes (por ejemplo, Pascal o C); los paréntesis pueden ser usados para agrupar. Por ejemplo:

```
>>> 2+2
4
>>> # Este es un comentario
... 2+2
4
>>> 2+2
# y un comentario en la misma línea que el código
4
>>> (50-5*6)/4
5
>>>
# La división entera retorna redondeado al piso:
... 7/3
2
```

```
>>> 7/-3
-3
```

El signo igual (=) es usado para asignar un valor a una variable. Luego, ningún resultado es mostrado antes del próximo prompt:

```
>>> ancho = 20
>>> largo = 5*9
>>> ancho * largo
900
```

Un valor puede ser asignado a varias variables simultáneamente:

3. Cadena de caracteres

Cadenas de caracteres Además de números, Python puede manipular cadenas de texto, las cuales pueden ser expresadas de distintas formas. Pueden estar encerradas en comillas simples o dobles:

```
>>> 'huevos y pan'
'huevos y pan'
>>> 'doesn\t'
"doesn't"
>>> "doesn't"
"doesn't"
>>> '"Si," le dijo.'
'"Si," le dijo.'
>>> "\"Si,\" le dijo."
'"Si," le dijo.'
>>> '"Isn\'t," she said.'
'"Isn\'t," she said.'
```

Las cadenas de texto literales pueden contener múltiples líneas de distintas

formas. Las líneas continuas se pueden usar, con una barra invertida como el último carácter de la línea para indicar que la siguiente línea es la continuación lógica de la línea

```
hola = "Esta es una larga cadena que contiene \n\ varias líneas de texto, tal y como se hace en C. \n\     Notar que los espacios en blanco al principio de la línea \  son significantes."
>>> print hola
```

ANEXO C

CODIGO DEL SERVIDOR WEB EN

EL RASPBERRY PI 2

1. EL ARCHIVO MAS IMPORTANTE ES EL “SETTINGS.PY

```
import os
from unipath import Path
BASE_DIR = Path(__file__).ancestor(2)

SECRET_KEY = 'xyfpnufe6+0p6eq9%^rpdas5p+ne)%@d=nvrtmnde52%86_v3&'
DEBUG = True
TEMPLATE_DEBUG = True
ALLOWED_HOSTS = []

INSTALLED_APPS = (
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'apps.inicio',
    'apps.autores',
)

from django.core.urlresolvers import reverse_lazy
LOGIN_URL = reverse_lazy('login')
LOGIN_REDIRECT_URL = reverse_lazy('pagina')
LOGOUT_URL = reverse_lazy('logout')

MIDDLEWARE_CLASSES = (
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
)

ROOT_URLCONF = 'primer.urls'
WSGI_APPLICATION = 'primer.wsgi.application'

TEMPLATE_DIRS = (
    BASE_DIR.child('templates'),
)

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'andresDB'),
    }
}

LANGUAGE_CODE = 'en-us'
TIME_ZONE = 'UTC'
USE_I18N = True
USE_L10N = True

USE_TZ = True
```

```
STATIC_URL = '/static/'

STATICFILES_DIRS = (
    BASE_DIR.child('static'),
)
```

2. ARCHIVO PRINCIPAL "URL.PY"

```
from django.conf.urls import patterns, include, url

from django.contrib import admin
admin.autodiscover()

urlpatterns = patterns('',
    url(r'^admin/', include(admin.site.urls)),
    url(r'^', include('apps.inicio.urls')),
)
```

3. ARCHIVOS DE LA APLICACIÓN

3.1. Archivo "URLS.PY" de la aplicación "Inicio"

```
from django.conf.urls import patterns, include, url
from .views import index, calendario

urlpatterns = patterns('',
    url(r'^$', 'django.contrib.auth.views.login', {'template_name':'inicio/index.html'}, name='login'),
    url(r'^cerrar/$', 'django.contrib.auth.views.logout_then_login', name='logout'),
    url(r'^inicio/$', 'apps.inicio.views.led', name="pagina"),
    url(r'^calendario/$', 'apps.inicio.views.calendario', name="calendario"),

)
```

3.2. Archivo "VIEWS.PY" de la aplicación "Inicio"

```
from django.shortcuts import render, render_to_response
from django.template.loader import get_template
from django.template import Context
from django.http import HttpResponse
from django.views.generic import TemplateView
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(2, GPIO.OUT)
GPIO.setup(14, GPIO.OUT)
GPIO.setup(15, GPIO.OUT)
GPIO.setup(18, GPIO.OUT)
GPIO.setup(23, GPIO.OUT)
GPIO.setup(24, GPIO.OUT)
GPIO.setup(5, GPIO.IN)
GPIO.setup(25, GPIO.IN)
GPIO.setup(21, GPIO.IN)
```

```
def index(request):
    return render('inicio/control.html')

def led(request):
    op1 = 20
    op2 = 20


    if 'result_1' in request.POST:
        estado1 = request.POST['result_1']
    else:
        estado1 = 'off'
    if 'on1' in request.POST:
        if estado1 == 'off':
            estado1 = 'on'
        else:
            estado1 = 'off'

    if 'result_2' in request.POST:
        estado2 = request.POST['result_2']
    else:
        estado2 = 'off'
    if 'on2' in request.POST:
        if estado2 == 'off':
            estado2 = 'on'
        else:
            estado2 = 'off'

    if 'result_3' in request.POST:
        estado3 = request.POST['result_3']
    else:
        estado3 = 'off'
    if 'on3' in request.POST:
        if estado3 == 'off':
            estado3 = 'on'
        else:
            estado3 = 'off'

    if 'result_4' in request.POST:
        estado4 = request.POST['result_4']
    else:
        estado4 = 'off'
    if 'on4' in request.POST:
        if estado4 == 'off':
            estado4 = 'on'
        else:
            estado4 = 'off'
    ctx = {'result_1': estado1, 'result_2': estado2, 'result_3': estado3, 'result_4': estado4}

    if 'on1' in request.POST:
                GPIO.output(23,GPIO.HIGH)
    elif 'off1' in request.POST:
                GPIO.output(23,GPIO.LOW)
    elif 'on2' in request.POST:
        GPIO.output(24,GPIO.HIGH)
    elif 'off2' in request.POST:
```

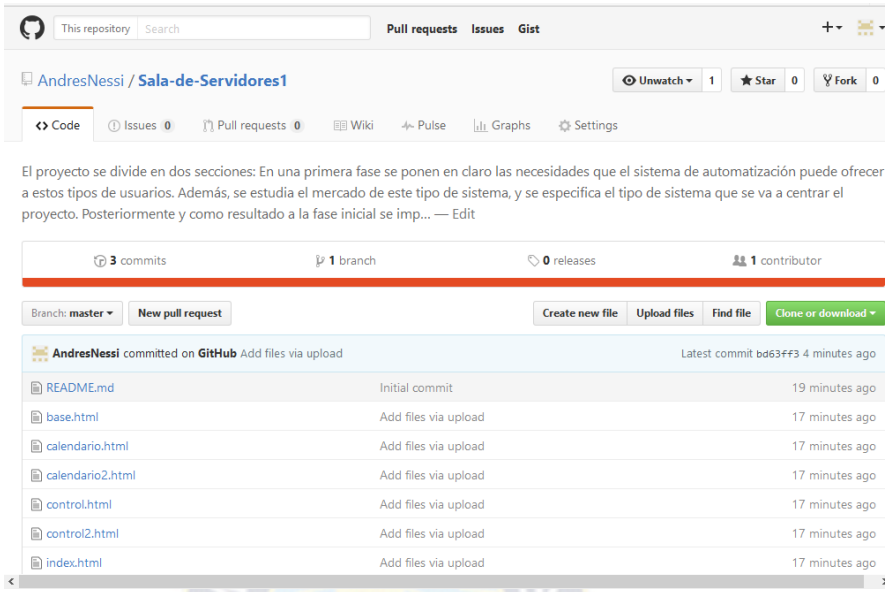
```
            GPIO.output(24,GPIO.LOW)
        elif 'up' in request.POST:
            GPIO.output(18,GPIO.HIGH)
            time.sleep(1)
            GPIO.output(18,GPIO.LOW)
        elif 'left' in request.POST:
            GPIO.output(14,GPIO.HIGH)
            time.sleep(1)
            GPIO.output(14,GPIO.LOW)
        elif 'rigth' in request.POST:
            GPIO.output(15,GPIO.HIGH)
            time.sleep(1)
            GPIO.output(15,GPIO.LOW)
        elif 'down' in request.POST:
            GPIO.output(2,GPIO.HIGH)
            time.sleep(1)
            GPIO.output(2,GPIO.LOW)
        elif 'sitio' in request.POST:
            return render(request, 'inicio/index4.html', ctx)
        elif 'alerta' in request.POST:
              return render(request, 'inicio/control.html', ctx)
        elif ( GPIO.input(25) == False ):
            return render(request, 'inicio/index2.html', ctx)
        elif ( GPIO.input(21) == False ):
            return render(request, 'inicio/index3.html', ctx)

        response = render(request, 'inicio/control.html', ctx)
        return response


    def calendario(request):
        return render(request, 'inicio/calendario.html')
```

Nota: todo el código completo se encuentra en mi repositorio que está abierto al público en mi cuenta de GitHub, aquí les dejo el link:

- https://github.com/AndresNessi/Sala-de-Servidores1


This repository
Search
Pull requests
Issues
Gist
AndresNessi / Sala-de-Servidores1
Unwatch 1
Star 0
Fork 0
Code
Issues 0
Pull requests 0
Wiki
Pulse
Graphs
Settings
El proyecto se divide en dos secciones: En una primera fase se ponen en claro las necesidades que el sistema de automatización puede ofrecer a estos tipos de usuarios. Además, se estudia el mercado de este tipo de sistema, y se especifica el tipo de sistema que se va a centrar el proyecto. Posteriormente y como resultado a la fase inicial se imp... — Edit
3 commits
1 branch
0 releases
1 contributor
Branch: master
New pull request
Create new file
Upload files
Find file
Clone or download
AndresNessi committed on GitHub Add files via upload
Latest commit bd63ff3 4 minutes ago
README.md Initial commit 19 minutes ago
base.html Add files via upload 17 minutes ago
calendario.html Add files via upload 17 minutes ago
calendario2.html Add files via upload 17 minutes ago
control.html Add files via upload 17 minutes ago
control2.html Add files via upload 17 minutes ago
index.html Add files via upload 17 minutes ago

ANEXO D

CODIGO EN EL ARDUINO UNO/NANO

```
#include  <SPI.h>
#include  <Ethernet.h>
#include  <DHT11.h>

int pin=2;
int movimiento=3;
int corriente1=4;
int humo=5;
int agua=6;

int pinState1 = 0;
int pinState2 = 0;
int pinState3 = 0;
int pinState4 = 0;
DHT11 dht11(pin);

byte mac[] = {
  // first byte DE, remaining bytes decided by d20 roll
    0xDE,  0xB2,  0x77,  0xA5,  0x21,  0x48
};

IPAddress ip(192, 168, 10, 200);

EthernetServer server(80);

void setup() {
    Ethernet.begin(mac, ip);
    server.begin();
    pinMode(movimiento, INPUT);
    pinMode(corriente1, INPUT);
    pinMode(humo, INPUT);
    pinMode(agua, INPUT);
}
```

```
const int MAXLN = 256;
char ln[MAXLN + 1];
int contentLength;

boolean readLine(EthernetClient &client) {
   int i = 0;
   boolean cr = false;

   for (;;) {
      if (!client.connected()) return false;
      if (!client.available()) continue;
      char c = client.read();
      switch (c) {
         case '\r':
            if (cr) {
               // should never happen
               if (i < MAXLN) ln[i++] = '\r';
            }
            cr  =  true;
            break;

         case '\n':
            if (cr) {
               // there's always room for the trailing NUL
               ln[i++]  =  '\0';
               return true;
            }
            // else fall through, treat \n like a regular character
            // should never happen

         default:
            if (cr) {
               // should never happen
               if (i < MAXLN) ln[i++] = '\r';
```

```
            cr  =  false;
          }
          if (i < MAXLN) ln[i++] = c;
          break;
      }
    }
}

boolean readRequestHeader(EthernetClient &client) {
    if(!readLine(client)) return false; // read the http request line

    contentLength  =  -1;  // initialize

    // we are assuming http 1.1. Read lines until we get a blank line
    do {
      if(!readLine(client)) return false; // read a header

      // pull out Content-Length, for example
      if(strncmp(ln, "Content-Length: ", 16) == 0) {
        contentLength  =  atoi(ln+16);
      }
    }
    while (strlen(ln) > 0);
    return true;
}


void loop() {
    // is someone making a request?
    EthernetClient client = server.available();
    if (!client) return;

    if(!readRequestHeader(client)) {
      client.stop();
```

```
    return;
}

int err;
float temp, hum;
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: application/json");
client.println("Connection: close");
// this is the CORS header. Yery important. Google CORS.
client.println("Access-Control-Allow-Origin: *");
client.println();
client.print("{\"alcohol\":");
int sensorReading = analogRead(0);
client.print(sensorReading);
client.print(',');
if((err = dht11.read(hum, temp)) == 0){
  client.print("\"temperatura\":");
  client.print(temp);
  client.print(',');
  client.print("\"humedad\":");
  client.print(hum);
}
client.print(',');
client.print("\"movimiento\":");
pinState1 = digitalRead(movimiento);
if (pinState1 == HIGH) {
  client.print("\"success\"");
} else {
  client.print("\"danger\"");
}
client.print(',');
client.print("\"humo\":");
pinState3 = digitalRead(humo);
if (pinState3 == HIGH) {
```

```
  client.print(true);
} else {
  client.print(false);
}
client.print(',');
client.print("\"agua\":");
pinState4 = digitalRead(agua);
if (pinState4 == HIGH) {
  client.print(true);
} else {
  client.print(false);
}
client.print(',');
client.print("\"corriente1\":");
pinState2 = digitalRead(corriente1);
if (pinState2 == HIGH) {
  client.print("\"success\"");
  client.print(',');
  client.print("\"estado1\":");
  client.print(false);
} else {
  client.print("\"danger\"");
  client.print(',');
  client.print("\"estado1\":");
  client.print(true);
}
client.print(',');
client.print("\"mensaje\":");
client.print("\"Estado de sensores\"");
client.println("}");

// give the web browser time to receive the data
delay(1);
// close the connection:
```

```
    client.stop();
}
```