

**ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO**

**FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**

**ESCUELA DE INGENIERÍA EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES**

**DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA ELECTRÓNICO PROTOTIPO GEORREFERENCIADO PARA MONITOREO Y CONTROL DE PUERTAS DE BUSES EN PARADAS ESPECÍFICAS DE LA CIUDAD DE RIOBAMBA**

Trabajo de titulación presentado para optar al grado académico de:

**INGENIERO EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES**

**AUTORES:** LUIS ALBERTO UVIDIA ARMIJO

NÉSTOR AUGUSTO ESTRADA BRITO

**TUTOR:** ING. PAÚL ROMERO R.

Riobamba- Ecuador

2015

**ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO**

**FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**

**ESCUELA EN INGENIERÍA ELECTRÓNICA, TELECOMUNICACIONES Y**

**REDES**

El Tribunal de Tesis certifica que: El trabajo de investigación: "DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA ELECTRÓNICO PROTOTIPO GEORREFERENCIADO PARA MONITOREO Y CONTROL DE PUERTAS DE BUSES EN PARADAS ESPECÍFICAS DE LA CIUDAD DE RIOBAMBA"*,* de responsabilidad de los señores Luis Alberto Uvidia Armijo y Néstor Augusto Estrada Brito, ha sido minuciosamente revisado por los Miembros del Tribunal de Tesis, quedando autorizada su presentación.

ING. GONZALO NICOLAY, Ph.D \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Decano de la Facultad de**

**Informática y Electrónica**

ING. FRANKLIN MORENO \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Director de la Escuela** **de**

**Ingeniería Electrónica**

ING. PAÚL ROMERO R. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Director de Tesis**

ING. WILLIAM CALVOPIÑA \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Miembro de Tesis**

ING. JORGE HUILCA \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Vicedecano de la Facultad de**

**Informática y Electrónica**

Nosotros Luis Alberto Uvidia Armijo y Néstor Augusto Estrada Brito somos responsables de las ideas, doctrinas y resultados expuestos en esta Tesis y el patrimonio intelectual de la Tesis de Grado pertenece a la Escuela Superior Politécnica De Chimborazo

Luis Alberto Uvidia Armijo Néstor Augusto Estrada Brito

060423777-6 060341288-3

**DEDICATORIA**

Primero ante todo agradecemos a Dios por darnos las fuerzas y guiarnos en este camino lleno de aprendizaje y sabiduría.

A los ingenieros Paul romero y William Calvopiña por habernos guiado en el desarrollo de este proyecto.

Agradecemos también al ingeniero Eduardo García por habernos brindado sus conocimientos, sugerencias, consejos y colaborado en todo lo necesario parara poder culminar nuestro proyecto de titulación.

De igual manera agradecemos a nuestras familias que siempre nos alentaron a esforzarnos, enseñándonos que todo esfuerzo trae su recompensa.

Y como no a nuestra querida Escuela Superior Politécnica de Chimborazo, por brindarnos la oportunidad de obtener una profesión y ser una ayuda para la sociedad.

Luis, Néstor

**AGRADECIMIENTO**

A Dios por haberme dado la vida y mucha fuerza para enfrentar todas las pruebas y poder culminar mi formación profesional enseñándome que con su guía todo es posible.

A mis padres que me guiaron y apoyaron incondicionalmente que me enseñaron a luchar por lo que quiero, a que es mejor dar que recibir, gracias por siempre estar ahí.

A mis hermanos Joe y Joa por estar siempre a mi lado apoyándome y dándome consejos de bien.

En especial a Gaby por ser mi guía, confidente y la mejor enamorada del mundo que con su apoyo logramos salir adelante juntos y vencemos cualquier obstáculo que la vida nos presente.

Luis

El más sincero agradecimiento a Dios por darme la vida y las fuerzas necesarias para salir a delante ante cualquier obstáculo con su guía y fe.

A mis padres y a mi hermano por los concejos y el apoyo incondicional que me enseñaron a luchar siempre para llegar a la meta que uno se lo propone y siempre conservar la humildad.

A mis tíos, tías y demás familiares por mantener siempre un interés y un apoyo incondicional en lo largo en carrera y mi vida.

A la Escuela Superior Politécnica de Chimborazo, por darme la oportunidad nutrirme de conocimientos valiosos y de obtener una profesión y ser una ayuda para la sociedad.

Néstor

# TABLA DE CONTENIDO

[TABLA DE CONTENIDO vi](#_Toc434999323)

[ÍNDICE DE TABLA viii](#_Toc434999324)

[ÍNDCE DE ILUSTRACIONES ix](#_Toc434999325)

[RESUMEN x](#_Toc434999326)

[SUMARY xi](#_Toc434999327)

[INTRODUCCIÓN 12](#_Toc434999328)

[CAPTÍULO I 13](#_Toc434999329)

[**1** MARCO TEÓRICO 13](#_Toc434999330)

[**1.1** **Sistema de Posicionamiento Global** 13](#_Toc434999331)

[**1.1.1** ***Características técnicas del gps*** 13](#_Toc434999332)

[**1.1.2** ***Generalidades*** 14](#_Toc434999333)

[**1.1.3** ***Segmentos*** 15](#_Toc434999334)

[**1.1.3.1** *Segmento espacial* 15](#_Toc434999335)

[**1.1.3.2** *Segmento de control* 16](#_Toc434999336)

[**1.1.3.3** *Segmento usuario* 17](#_Toc434999337)

[**1.1.4** ***Receptores Gps*** 18](#_Toc434999338)

[**1.1.5** ***Fuentes de error en el Gps*** 18](#_Toc434999339)

[**1.1.6** ***Módulo gps u-blox neo-6m*** 19](#_Toc434999340)

[**1.1.6.1** *Funcionamiento del gps ublox neo-6m* 20](#_Toc434999341)

[**1.1.6.2** *Características modulo gps ublox neo-6m* 21](#_Toc434999342)

[**1.2** **Arduino** 23](#_Toc434999343)

[**1.2.1** ***Hardware*** 23](#_Toc434999344)

[**1.2.1.1** *Arduino mega* 23](#_Toc434999345)

[**1.2.1.2** *Alimentación* 24](#_Toc434999346)

[**1.2.1.3** *Memoria* 25](#_Toc434999347)

[**1.2.1.4** *Entradas y salidas* 25](#_Toc434999348)

[**1.2.1.5** *Comunicaciones* 27](#_Toc434999349)

[**1.2.1.6** *Programación* 27](#_Toc434999350)

[**1.2.2** ***Software*** 28](#_Toc434999351)

[**1.2.2.1** *Instalación del software* 28](#_Toc434999352)

[**1.2.2.1.1** *Obtener una placa Arduino y el cable USB* 28](#_Toc434999353)

[**1.2.2.1.2** *Descargue el software de Arduino (IDE*) 28](#_Toc434999354)

[**1.2.2.1.3** *Instalar Drivers USB* 28](#_Toc434999355)

[**1.2.2.1.4** *Conectar la placa* 29](#_Toc434999356)

[**1.2.2.1.5** *Ejecutar el entorno Arduino* 32](#_Toc434999357)

[**1.3** **lcd con módulo i2c** 32](#_Toc434999358)

[**1.3.1** ***Características***: 32](#_Toc434999359)

[CAPÍTULO II 32](#_Toc434999360)

[**2** MARCO METODOLÓGICO 32](#_Toc434999361)

[**2.1** **Selección de la placa arduino** 32](#_Toc434999362)

[**2.2** **Selección del Gps** 35](#_Toc434999363)

[**2.3** **Conexiones gps con la tarjeta arduino mega** 37](#_Toc434999364)

[**2.4** **Conexiones Modulo de Tarjeta SD con la Tarjeta Arduino** 38](#_Toc434999365)

[**2.5** **Conexiones I2c lcd display con la Tarjeta Arduino** 39](#_Toc434999366)

[**2.6** **Interfaz de potencia** 41](#_Toc434999367)

[CAPÍTULO III 41](#_Toc434999368)

[**3** **MARCO DE RESULTADOS** 41](#_Toc434999369)

[**3.1** **Coordenadas Predefinidas en la Sdcard** 41](#_Toc434999370)

[**3.2** **Lectura de Datos** 42](#_Toc434999371)

[**3.3** **Diferencia de Lecturas** 45](#_Toc434999372)

[**3.4** **Tabla de Resultados** 46](#_Toc434999373)

[CONCLUSIONES 49](#_Toc434999374)

[RECOMENDACIONES 50](#_Toc434999375)

[BIBLIOGRAFÍA](#_Toc434999376)

[ANEXOS](#_Toc434999377)

# ÍNDICE DE TABLA

[**Tabla 1‑1:** Funcionamiento del GPS neo-6m 20](#_Toc434999389)

[**Tabla 2-1:** Características de Arduino mega 24](#_Toc434999390)

[**Tabla 1‑2 :**Tabla comparativa de Arduino. 33](#_Toc434999391)

[**Tabla 2‑2 :**Escala cuantitativa 33](#_Toc434999392)

[**Tabla 3-2 :**Escala cualitativa 33](#_Toc434999393)

[**Tabla 4‑2 :**Tabla cuantitativa de Arduino 34](#_Toc434999394)

[**Tabla 5‑2 :**Interpretación 34](#_Toc434999395)

[**Tabla 6‑2 :**Tabla comparativa de GPS 35](#_Toc434999396)

[**Tabla 7‑2 :**Tabla Cuantitativa de GPS 36](#_Toc434999397)

[**Tabla 8‑2 :**Interpretación 36](#_Toc434999398)

[**Tabla 1-3:** Resultados con las Primeras coordenadas predefinidas en la Sdcard 47](#_Toc434999399)

[**Tabla 2-3**: Resultados con las Segundas coordenadas predefinidas en la Sdcard 47](#_Toc434999400)

[**Tabla 3‑3:** Resultados con las Terceras coordenadas predefinidas en la Sdcard 47](#_Toc434999401)

# ÍNDICE DE ILUSTRACIONES

[**Figura 1-1 :** Sistema de Posicionamiento Global 13](file:///C:\Users\Luz%20America\Music\tesis100xciento.docx#_Toc434999483)

[**Figura 2-1:** Señales GPS para calcular las posiciones en 3 dimensiones 15](file:///C:\Users\Luz%20America\Music\tesis100xciento.docx#_Toc434999484)

[**Figura 3-1:** Constelación de satélites alrededor de la Tierra 15](file:///C:\Users\Luz%20America\Music\tesis100xciento.docx#_Toc434999485)

[**Figura 4-1 :** Número de orbitas alrededor de la Tierra 16](file:///C:\Users\Luz%20America\Music\tesis100xciento.docx#_Toc434999486)

[**Figura 5-1:** Estaciones de seguimiento 17](file:///C:\Users\Luz%20America\Music\tesis100xciento.docx#_Toc434999487)

[**Figura 6-1:** Cuatro satélites calculan las cuatro dimensiones 17](file:///C:\Users\Luz%20America\Music\tesis100xciento.docx#_Toc434999488)

[**Figura 7-1**:Módulo GPS Ublox NEO-6M con antena 19](file:///C:\Users\Luz%20America\Music\tesis100xciento.docx#_Toc434999489)

[**Figura 8-1:** Diagrama esquemático del módulo 22](file:///C:\Users\Luz%20America\Music\tesis100xciento.docx#_Toc434999490)

[**Figura 9-1:**Arduino Mega 2560 23](file:///C:\Users\Luz%20America\Music\tesis100xciento.docx#_Toc434999491)

[**Figura 10-1:** Descripción de componentes de la placa Arduino Mega2560 28](#_Toc434999492)

[**Figura 11-1:** Conexión del cable USB a Arduino 29](file:///C:\Users\Luz%20America\Music\tesis100xciento.docx#_Toc434999493)

[**Figura 13-1:**Asistente para Nuevo hardware paso 2 30](#_Toc434999494)

[**Figura 12-1:**Asistente para el nuevo hardware paso1 30](file:///C:\Users\Luz%20America\Music\tesis100xciento.docx#_Toc434999495)

[**Figura 14-1:**Asistente para Nuevo hardware paso 3 31](#_Toc434999496)

[**Figura 15-1:**Asistente para Nuevo hardware paso 4 31](file:///C:\Users\Luz%20America\Music\tesis100xciento.docx#_Toc434999497)

[**Figura 1-2:** Gráfico porcentajes para la elección de Arduino 35](#_Toc434999498)

[**Figura 2-2:** Gráfico de porcentajes para la elección del GPS 37](#_Toc434999499)

[**Figura 3-2:** Conexión del módulo GPS con el Arduino MEGA 2560 37](#_Toc434999500)

[**Figura 4-2:** Conexión del módulo GPS con el Arduino MEGA 2560 38](#_Toc434999501)

[**Figura 5-2:** Conexión del módulo GPS con el Arduino MEGA 2560 39](#_Toc434999502)

[**Figura 6-2:** Interfaz de Potencia 41](#_Toc434999503)

[**Figura 1-3:** Coordenadas predefinidas en la SDcard 42](#_Toc434999504)

[**Figura 2-3:** Lectura de la Sdcard 43](#_Toc434999505)

[**Figura 3-3:** Lectura de la Sdcard 43](#_Toc434999506)

[**Figura 4-3:** Lectura de las coordenadas 44](#_Toc434999507)

[**Figura 5-3:** Lectura del GPS 44](#_Toc434999508)

[**Figura 6-3:** Lectura del GPS 45](#_Toc434999509)

[**Figura 7-3:** Diferencia entre las coordenadas 46](#_Toc434999510)

# RESUMEN

Se diseñó e implemento un sistema electrónico prototipo georreferenciado para monitoreo y control de puertas de buses en paradas específicas de la ciudad de Riobamba, con la finalidad de prohibir a los conductores estacionarse y abrir puertas en paradas no establecidas y que en la ciudad no se genere congestionamiento de tráfico. Para el diseño se aplicó la plataforma arduino y para su implementación se utilizó el Sistema de Posicionamiento Global (GPS) el cual ayudó a obtener los datos requeridos tales como la hora actual, fecha, latitud (localización) del vehículo monitoreado, lo cual proporcionó información exacta en el momento requerido, los datos proporcionados, se puede observar en un display, mediante la utilización de algunas líneas de programación en arduino, se logró obtener las coordenadas geográficas del GPS para proceder a la comparación con las 3 coordenadas de las paradas anteriormente establecidas en una base de datos en una Sdcard. Una vez implementado el prototipo se pudo analizar y obtener resultados satisfactorios de su funcionamiento, en un 100% de efectividad con un rango del 5% del margen de error. Se concluye que con la implementación del sistema electrónico prototipo georreferenciado, los vehículos podrán abrir sus puertas solo en paradas establecidas, previamente programadas, se recomienda utilizar un arduino con una mayor memoria para poder implementar más líneas de programación.

Palabras Clave: <SISTEMA GEORREFERRENCIAL> <GPS> <ARDUINO> <BASE DE DATOS> <SDCARD>

# SUMARY

An electronic system of dereferenced model was implemented for monitoring and control of doors of buses at specific stops of the city of Riobamba in order to forbid drivers to park and open doors in not established stops and avoiding traffic jams into the city. The Arduino platform was placed for the design and for its implementation was used the Global Positioning System (GPS) which helped to obtain the required data such as the current time, date, latitude (location) of the monitored vehicle, which provided information at the necessary time, the data supplied could be observed in a display, using some lines of Arduino programming, it was possible to obtain the geographical coordinates of the GPS to carry out the comparison with the 3 coordinates of the stops previously established in a database in a Sdcard. Once the model was implemented it was possible to analyze and getting satisfactory results of its operation, in a 100% of effectiveness with a margin of error range of 5%. It is concluded that with the implementation of electronic system of georeferenced model, the vehicles may open their doors at established stops only; it is also recommended an Arduino with more memory to implement more lines of programming.

# INTRODUCCIÓN

La evolución tecnológica, la globalización, la integración de la sociedad y las tecnologías de posicionamiento y localización obligan a todos los países a estar a la par con su evolución, mejorando así la calidad de vida de los habitantes y permitiendo alcanzar nuevos desafíos para el desarrollo de un entorno.

Gracias a la tecnología del Sistema de Posicionamiento Global (GPS) que sirve de apoyo en la ubicación y posicionamiento de los medios de transporte tanto aéreo, terrestre y marítimo el cual nos proporciona su exactitud cronométrica.

El crecimiento de la población y el aumento vehicular en la ciudad de Riobamba obliga a investigar e implementar nuevos sistemas de movilización segura para una mejor calidad de vida de los habitantes.

La ciudad de Riobamba cuenta con distintas cooperativas de buses para el transporte de pasajeros, las cuales tiene su propio recorrido. Existe una actitud, en la mayoría de los casos inconsecuente por parte de los choferes profesionales al momento de conducir un medio de transporte público en este caso un Auto bus que tienden a estacionarse de manera autoritaria e inconsciente en zonas no establecidas como paradas, por ejemplo redondeles, pasos cebras, generando congestionamiento de tráfico.

La implementación de este sistema nació de la necesidad de disminuir la congestión vehicular por parte de los buses para una movilidad peatonal segura, también para controlar y verificar que el tiempo y la ruta sean cumplidas a cabalidad.

Este proyecto de tesis busca aplicar e implementar todos los conocimientos adquiridos en la carrera universitaria para implementar un sistema para el control de puertas de los buses según las coordenadas geográficas de las paradas respectivas en la ruta establecida de la línea de bus.

En el prototipo realizado se programaron 3 paradas importantes que obligara a los choferes y usuarios a optimizar su tiempo y a abrir sus puertas solo en paradas específicas que previamente fueron programadas en el dispositivo, otorgando así un medio de transporte seguro, eficiente y confiable.

# CAPTÍULO I

# MARCO TEÓRICO

## SISTEMA DE POSICIONAMIENTO GLOBAL

La tecnología GPS hoy en día se ha convertido en un recurso muy utilizado a nivel mundial tanto es así que supera los objetivos para los que fue diseñada. El GPS al pasar de los años ha cambiado en el hecho de tener un uso exclusivamente militar a convertirse en un instrumento de uso común que permite hacer un sinfín de trabajos más seguros y productivos. Las aplicaciones principales del sistema son:

* **Localización** o determinación básica de la posición.
* **Navegación** o establecimiento de rutas entre puntos distintos.
* **Seguimiento** o monitorización de los desplazamientos de personas u objetos.
* **Cartografía**. El GPS se emplea en la creación de mapas de precisión.
* **Fuente de sincronismo** con precisiones superiores al microsegundo.

### CARACTERÍSTICAS TÉCNICAS DEL GPS

La principal aplicación del GPS es la determinación de una posición geográfica en cualquier punto del planeta y bajo cualquier tipo de condición atmosférica.

El GPS fue creado principalmente para brindarnos datos sobre el camino que lleva cualquier receptor móvil, ya sea marítimo, aéreo o terrestre, guiándolos para que no se desvíen de su ruta y mejorando la densidad del tráfico.

Figura 1-1: Sistema de Posicionamiento Global

**Fuente:** (Pc-docto, 2014)

Gracias a estas ventajas que nos proporciona el GPS podemos gozar de una mayor seguridad y menor consumo de combustible al momento de conducir de un punto a otro. El empleo de dicho dispositivo está tan avanzado que incluso se lo emplea para el control y monitoreo de flotas de camiones de transporte o de autobuses urbanos.

Sin embargo, gracias a la elevada precisión de las señales brindadas por los satélites GPS, éstas también se emplean como fuente de sincronismo para múltiples redes telemáticas, como la propia Internet, mediante protocolos específicos como el NTP (*Network Timing Protocol*).

El receptor GPS tiene la capacidad de interconectarse con varios sistemas electrónicos o informáticos como odómetros, giróscopos, detectores de movimiento, alarmas, etc.

El Sistema de Información Geográfica (GIS) es el complemento perfecto para un receptor GPS: dicho sistema es un programa de ordenador que detalla bases cartográficas digitales sobre las cuales se puede establecer posiciones y rumbos gracias a los datos brindados por el GPS.

Hoy en día gracias a los avances tecnológicos se va abaratando y reducido significativamente el tamaño de los receptores logrando que esta tecnología se haga cada vez más útil y común en nuestra sociedad.

### GENERALIDADES

* El sistema fue diseñado y operado por el ejército estadounidense conjuntamente con el departamento de Defensa de Estados Unidos (DOD)
* Proporciona una información muy fiable acerca de la posición en 3 dimensiones, la velocidad y la hora exacta en cualquier parte del mundo.
* Es el sistema satelital conformado por señales de radio enviadas por una constelación de 21 a 32 satélites en modo activo que permanecen en órbita alrededor de la tierra a una altura de 20 200 km aproximadamente, las 24 horas del día, desplazándose a una velocidad de 14.500 Km./h.
* Se emplean cuatro señales de los satélites GPS para descifrar las posiciones en tres dimensiones y el desplazamiento en el reloj del receptor tiempo.

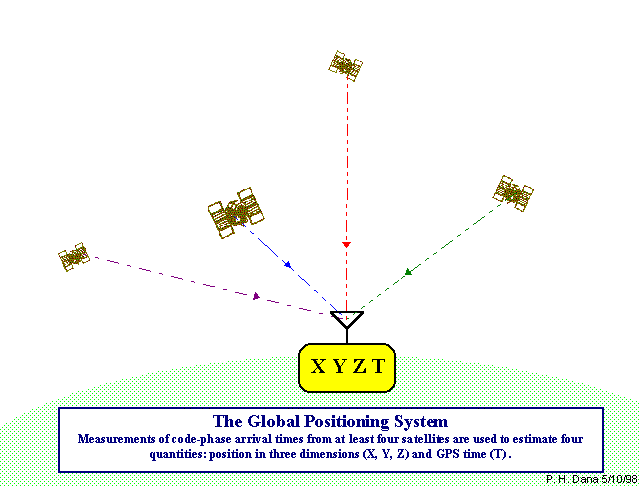


Figura 2-1: Señales GPS para calcular las posiciones en 3 dimensiones

**Fuente**: (Federal Radionavigation Plan, 2000)

### SEGMENTOS

#### SEGMENTO ESPACIAL

Constituido por una constelación de satélites, estos vehículos espaciales (SVS) envían señales de radio desde el espacio.

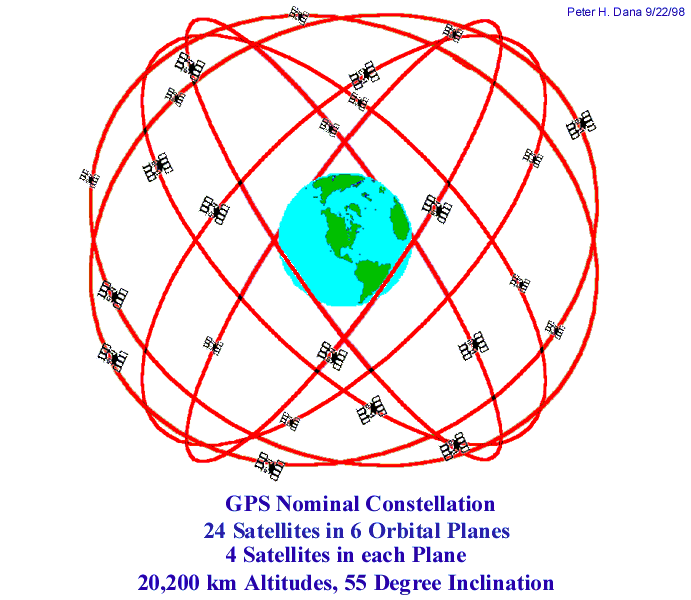


Figura 3-1: Constelación de satélites alrededor de la Tierra

**Fuente:** (Departamento de Geografía de Texas, 2015).

El GPS está constituido por 32 satélites operativos y 4 o más de reserva, ubicados en seis planos orbitales inclinados 55 grados con respecto al plano ecuatorial terrestre, desde una distancia de 20200 km de la Tierra.

Dichos satélites completan una órbita a su alrededor cada 12 horas, permitiendo que puedan ser vistos unos 6 satélites como mínimo a cada instante desde cualquier punto del planeta.

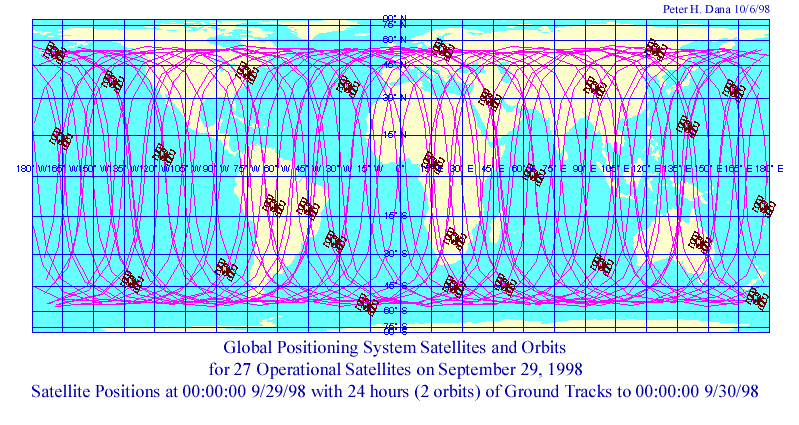


Figura 4-1: Número de orbitas alrededor de la Tierra

**Fuente:** (Departamento de Geografía de la Universidad de Texas)

Transmiten un mensaje modulado en la señal de radio de una forma continua, llevando con sigo datos relacionados a su ubicación en el espacio y la hora actual en formato UTC (Tiempo Universal Coordinado)

#### SEGMENTO DE CONTROL

Transmiten un mensaje modulado en la señal de radio de una forma continua, llevando con sigo datos relacionados a su ubicación en el espacio y la hora actual en formato UTC (Tiempo Universal Coordinado).

El centro de control maestro de GPSselocaliza en la base aérea Falcón, en Colorado (EEUU), en el cuál técnicos especializados se encargan de corregir la información recibida por las estaciones de seguimiento en cuanto a posición y tiempo se refiere de toda la constelación. 

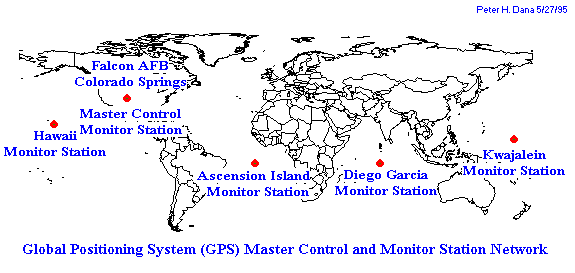


Figura 5-1: Estaciones de seguimiento

**Fuente**: (Departamento de Geografía de la Universidad de Texas, 2015)

#### SEGMENTO USUARIO

El Segmento GPS del usuario se conforma por los receptores GPS y la comunidad de usuarios. Los receptores GPS transforman las señales receptadas en estimaciones acertadas de velocidad, posición y tiempo.

Se necesita de la ayuda de cuatro satélites para determinar las tres dimensiones de X, Y, Z (posición) y la hora. Los receptores GPS se utilizan para la navegación, posicionamiento, difusión tiempo, y otras investigaciones.

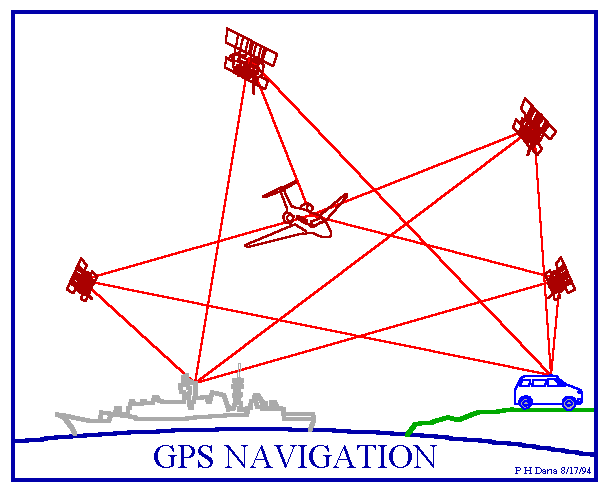


Figura 6-1: Cuatro satélites calculan las cuatro dimensiones

**Fuente**: (Departamento de Geografía de la Universidad de Texas, 2015).

### RECEPTORES GPS

Funcionalmente existen tres tipos de receptores GPS:

* **Receptores secuenciales o monocanal.** Poseen un solo canal radio por esta razón únicamente enganchan un satélite a la vez.
* **Receptores continuos o multicanal.** El receptor está conformado por lo general por 4 canales, que se usan para sincronizar con las emisiones de otros satélites en forma simultánea.
* **Receptores multiplexados.** Se posee un único canal físico sobre el cual montan varios canales lógicos mediante software, con los que se pueden sincronizar datos de todos los satélites visibles en un tiempo no máximo a los 20 milisegundos.

Se encuentra en dos estados distintos el receptor GPS de un usuario cuando entra en funcionamiento los cuales son:

* **Perdido.** Los datos memorizados por el satélite la última vez que estuvo en funcionamiento tienen una fecha muy antigua. El receptor busca al satélite que le ofrezca una mejor relación señal a ruido y demodula su mensaje de navegación, consiguiendo de esta forma el almanaque y la referencia temporal. Esta situación puede llevar varios minutos.
* **Memorizado.** El receptor comprueba que los datos que tenía almacenados en memoria son válidos y utiliza los mismos satélites que la última vez que fue empleado.

### FUENTES DE ERROR EN EL GPS

**Disponibilidad Selectiva:** Es causada por la contaminación de las señales de telemetría que usan los usuarios no privilegiados y sirviendo como mecanismo de defensa ante los elementos que no son autorizados para usar el GPS, supone una degradación de la precisión de entre 30 y 100 metros.

**Retardo de propagación en la atmósfera.**Una señal GPS cruza un gran número de capas no homogéneas de la atmósfera, causando retardos adicionales a la sincronización: la ionosfera está compuesta de partículas cargadas y su distribución en tamaño y densidad de carga es variable en relación de la radiación solar, las fluctuaciones del campo magnético terrestre y otros factores.

**Propagación multicamino.** Existen errores debidos a reflexión de las señales en diversos obstáculos cercanos al receptor. La suma de las señales directa y reflejada degrada la señal digital, lo cual obliga a la utilización de sistemas de corrección de errores.

**Errores de efemérides.**Las posiciones orbitales teóricas de los satélites pueden variar con el tiempo. Estos errores se denominan "errores de efemérides", y se solucionan monitorizando constantemente cada satélite, existen varios algoritmos basados en datos experimentales, cuyos coeficientes se transmiten a través del mensaje de navegación para que el receptor pueda utilizarlos.

**Atenuación de precisión debida a posición.**Estos errores se deben a la propia posición de los satélites: si el receptor sintoniza satélites cercanos entre sí, las esferas interferentes de las que hemos hablado se intersecan en ángulos muy agudos, lo cual hace aumentar el área de error de posición.

Los errores más graves, e incluso varias centenas de kilómetros, se deben a fallos electrónicos en alguna parte del sistema.

### Modulo GPS U-BLOX NEO-6M



**Figura 7-1**: **Módulo GPS Ublox NEO-6M con antena**

Fuente: (hetpro-store 2015)

Estos receptores flexibles y rentables ofrecen numerosas opciones de conectividad en una miniatura, combinan un alto nivel de capacidad de integración con conectividad flexible, y presentación de la información en un paquete pequeño.

Es capaz de realizar búsquedas paralelas masivas en un corto espacio de tiempo, lo que le permite encontrar satélites al instante, su diseño y tecnología innovadora suprime las fuentes de interferencia y mitiga los efectos de trayectoria múltiple, dando a los receptores GPS NEO-6 excelente performance de navegación, incluso en los entornos más difíciles.

Esto lo hace perfectamente adecuado para los productos finales de mercado masivo con tamaño y costo bajos que satisfagan nuestros intereses sin mayor inversión. Todos los módulos neo-6 se fabrican en ISOes de AlimentaciSalidas GPS.ntereses. ototipo por su diseño que permite / TS 16949, los cuales son sitios certificados, cada módulo se prueba y se inspecciona durante la producción.

Los módulos están calificados de acuerdo a la norma ISO 16750; U-blox 6 fue seleccionado para el desarrollo del prototipo por su bajo consumo de energía y bajos costos. La gestión inteligente de la energía es un desglose a través de aplicaciones de baja potencia.

#### FUNCIONAMIENTO DEL GPS ****UBLOX NEO-6M****

Tabla 1‑1: Funcionamiento del GPS neo-6m

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PARAMETRO** | **ESPECIFICACION** | | | | |
| Tipo de Receptor | 50 canales  GPS frecuencia L1, C/A code  SBAS: WAAS, EGNOS, MSAS | | | | | |
| Hora de primera revisión | arranque en frío arranque en caliente arranque en caliente comienza ayudados | NEO-6G/Q/T  26s  26s  1s  1s | NEO-6M/V  27s  27s  1s  <3 s | | NEO-6P  32s  32s  1s  <3 s | |
| sensibilidad | Seguimiento y navegación readquisición Arranque en frío Inicio caliente | NEO-6G/Q/T  -162dBm  -160dBm  -148dBm  -157dBm | NEO-6M/V  -161dBm  -160dBm  -147dBm  -156dBm | | NEO-6P  -160dBm  -160dBm  -146dBm  -155dBm | |
| Tipo de actualización de navegación máxima |  | NEO-6G/Q/MT  5Hz |  | |  | |
| Precisión horizontal | GPS  SBAS  SBAS + PPP7  SBAS + PPP | 2.5m  2.0m  < 1m (2D, R50)  < 2m (3D, R50)) | | | | |
| gama de frecuencia configurable |  | NEO-6G/Q/M/P/V  0.25 Hz to 1 kHz | | NEO-6T  0.25 Hz to 10 MHz | | |
| Precisión de señal | RMS  99%  Granularity  Compensated | 30 ns  <60 ns  21 ns  15ns | | | | |
| Exactitud de velocidad |  | 0.1m/s | | | | |
| Precisión de partida |  | 0.5 degrees | | | | |
| Limites operativos | Dinámica  Altitud  Velocidad | 4g  50,000 m  500 m/s | | | | |

**Realizado por**: NEO-6M

**Fuente:** (datasheet ublox, 2013)

#### CARACTERÍSTICAS MODULO GPS UBLOX NEO-6M

* Comunicación serial
* Voltaje de alimentación:  (3.5 – 5 )VDC
* Incluye Antena cerámica
* LED indicador de señal
* Tamaño de antena 22x22mm
* Tamaño de módulo 23x30mm
* Batería incluida
* BAUDRATE: 9600
* EEPROM
* WGS-84
* Sensibilidad de captura -148dBm
* Sensibilidad de rastreo: -161 dBm
* Máxima altura medible: 18000
* Máxima velocidad 515 m/s
* Exactitud: 1micro segundo
* Frecuencia receptora: L1 (1575.42 Mhz)
* Código C/A 1.023 Mhz
* Tiempo de inicio primera vez: 38s en promedio
* Tiempo de inicio : 35s en promedio
* A continuación en la figura se muestra el diagrama esquemático de la estructura interna del GPS U-blox NE=6M.

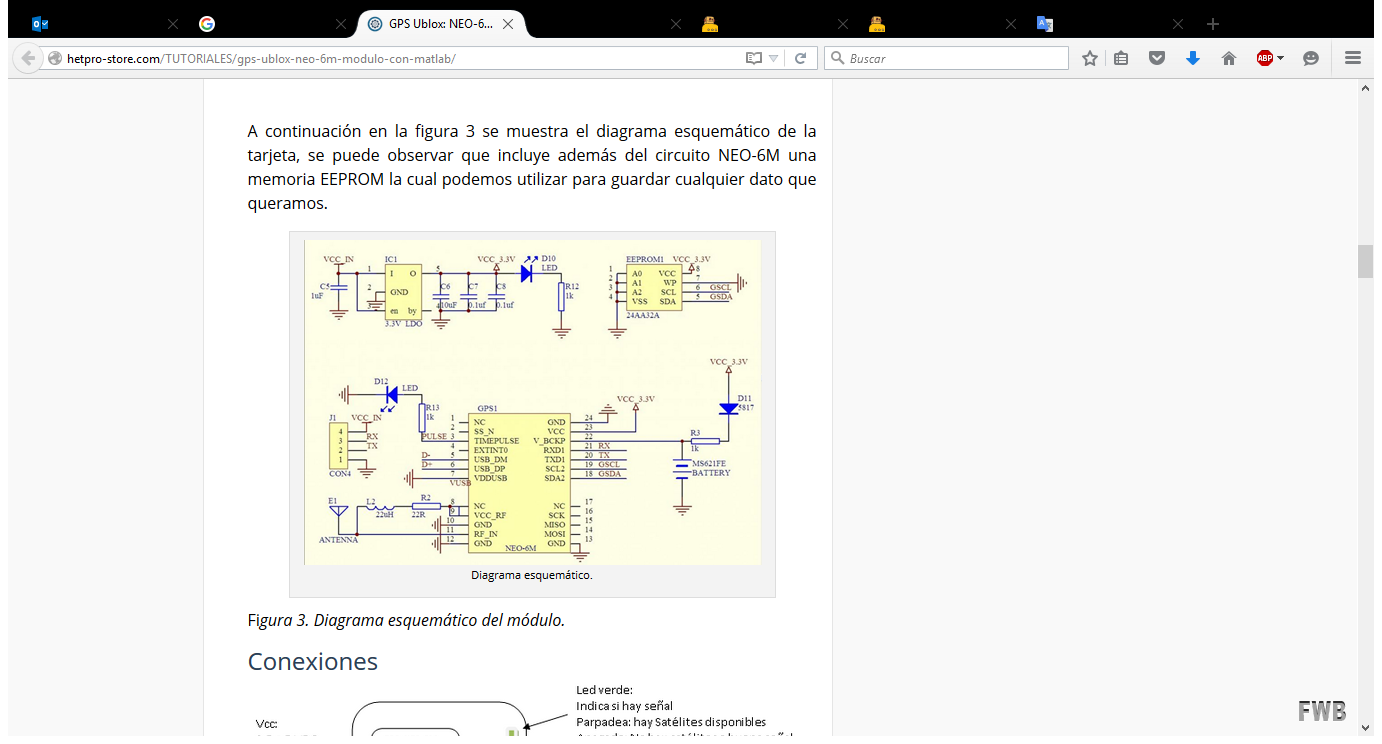


Figura 8-1: Diagrama esquemático del módulo

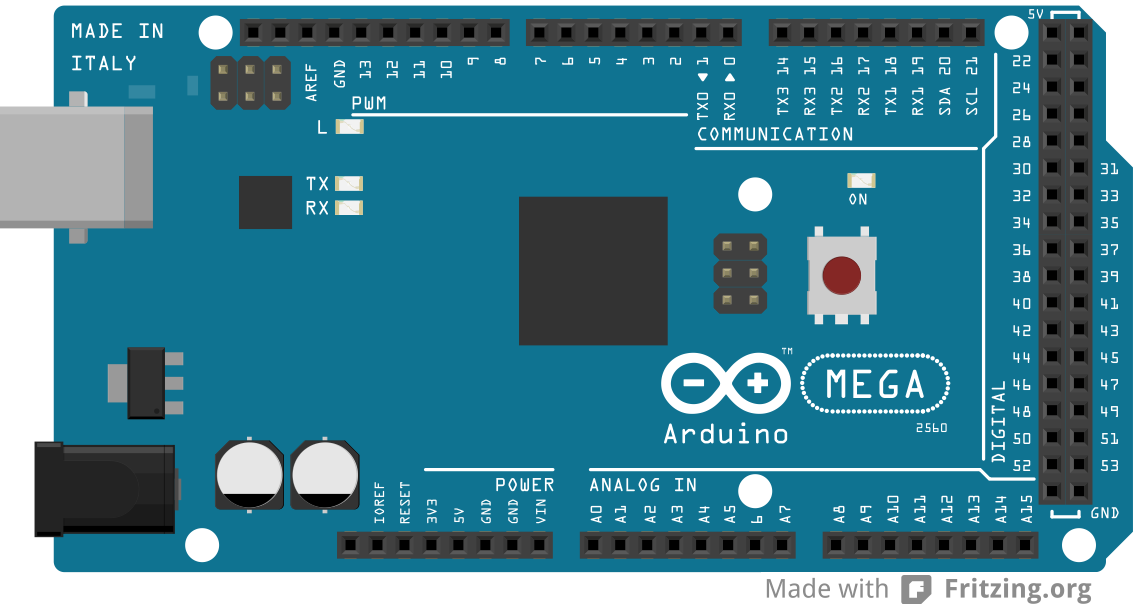
**Fuente:** (Hetpro-store, 2015)

## ARDUINO

### HARDWARE

#### ARDUINO MEGA

El Arduino Mega 2560 es una nueva versión extendida de la placa electrónica original de Arduino ya que también está basada en el microcontrolador Atmega2560.



**Figura 9-1:** Arduino Mega 2560

**Fuente:** (Datasheet Arduino, 2013)

Posee 54 pines de entradas/salidas que son digitales, de las cuales 15 se puede utilizar como salidas PWM (modulación de anchura de pulso, 16 entradas analógicas, 4 UARTs (puertas series), un oscilador de 16MHz, una conexión USB, un conector de alimentación, un conector ICSP y un pulsador para el reset. Para utilizar la placa es necesario conectarla al ordenador a través de un cable USB, o alimentarla con un adaptador de corriente AC/DC. También se lo puede alimentar mediante una batería. (Arduino, 2013)

La principal diferencia de la tarjeta Arduino MEGA 2560 es que no utiliza el convertidor USB-serie de la firma FTDI, emplea un microcontrolador Atmega8U2 programado como convertidor USB a serie. El Arduino MEGA2560 es compatible con la mayoría de los shield o tarjetas de aplicación/ampliación disponibles para las tarjetas Arduino UNO original. (Arduino, 2013)

Sus características principales son:

Tabla 2-1: Características de Arduino mega

|  |  |
| --- | --- |
| Microcontrolador | [ATmega2560](http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf) |
| Voltaje de Operación | 5V |
| Tensión de alimentación (recomendado) | 7-12V |
| Tensión de alimentación (limite) | 6-20V |
| Pines de E/S digitales | 54 (15 se utiliza como salida PWM ) |
| Pines de entrada analógica | 16 |
| Corriente continua para E/S | 20 mA |
| Salida de alimentación a 3.3V | 50 mA |
| Memoria de programación | 256 KB (8KB usado por bootloader) |
| Memoria SRAM | 8 KB |
| Memoria EEPROM para datos y variables | 4 KB |
| Velocidad de reloj | 16 MHz |
| Alto | 101.52 mm |
| Ancho | 53.3 mm |
| Peso | 37 g |

Realizado por: Arduino

**Fuente:** (Data sheet, 2013)

#### ALIMENTACIÓN

Puede ser alimentado por conexión USB o con una fuente de alimentación externa. La selección del origen de alimentación es de manera automática.

La alimentación externa pueden ser tanto un transformador o una batería. El transformador puede ser conectado mediante conector macho de 2.1mm con centro positivo en el conector hembra de la placa. Los cables de la batería pueden ser conectaos a los pines Gnd y Vin en los conectores de alimentación. (Enríquez Herrador R, 2009, p.12.).

La placa puede trabajar con una alimentación externa de entre 6 a 20 voltios. Si el voltaje suministrado es inferior a 7V, el pin de 5V puede proporcionar menos de 5 Voltios y la placa puede volverse inestable; si se usan más de 12V los reguladores de voltaje se pueden sobrecalentar y dañar la placa. El rango recomendado es de 7 a 12 voltios (Rafael Enríquez Herrador, 2009, p.12.).

Los pines de alimentación son:

* **VIN.** La entrada de voltaje a la placa Arduino cando se está usando una fuente externa de alimentación (en opuesto a los 5 voltios de la conexión USB). Se puede proporcionar voltaje a través de este pin, o, si se está alimentando a través de la conexión de 21mm, acceder a ella a través de este pin. (Enríquez Herrador R, 2009, p.12.).
* **5V.** Fuente estabilizado de voltaje que se usa para alimentar el microcontrolador y otros componentes de la placa. Esta puede provenir de VIN a través de un regulador integrado en la placa, o por el USB u otra fuente estabilizada de 5V. (Enríquez Herrador R, 2009, p.12.).
* **3V3.** Fuente de voltaje de 3.3 voltios generada por un regulador integrado en la placa. La corriente máxima soportada 50mA. (Enríquez Herrador R, 2009, p.12.).
* **GND.** Pines de toma de tierra. (Enríquez Herrador R, 2009, p.12.).

#### MEMORIA

Posee 256KB de memoria flash para almacenar código, 8KB son usados para el arranque del sistema (bootloader). Tiene 8 KB de memoria SRAM y 4KB de EEPROM, a la cual se puede acceder para leer o escribir con la [librería EEPROM](http://www.arduino.cc/en/Reference/EEPROM). (Enríquez Herrador R, 2009, p.13.).

#### ENTRADAS Y SALIDAS

Tiene 54 pines digitales los cuales pueden utilizarse como entradas o como salidas usando las funciones [**pinole ()**](http://arduino.cc/es/Reference/PinMode)**, [digitalWrite ()](http://arduino.cc/es/Reference/DigitalWrite), y [digitalRead ()](http://arduino.cc/es/Reference/DigitalRead)**. Las E/S operan a 5 voltios. Cada pin puede proporcionar o recibir una intensidad máxima de 40mA y tiene una resistencia interna de puy-up de 20-50kOhms. Que esta desconectada por defecto. (Enríquez Herrador R, 2009, p.13.).

Algunos pines tienen funciones especializadas:

* **Serie: 0 (RX) y 1 (TX), Serie 1: 19 (RX) y 18 (TX); Serie 2: 17 (RX) y 16 (TX); Serie 3: 15 (RX) y 14 (TX).** Usados para recibir (RX) transmitir (TX) datos a través de puerto serie TTL. Los pines Serie: 0 (RX) y 1 (TX) están conectados a los pines correspondientes del chip FTDI USB-te-TTL (Enríquez Herrador R, 2009, p.13.).
* **Interrupciones Externas: 2 (interrupción 0), 3 (interrupción 1), 18 (interrupción 5), 19 (interrupción 4), 20 (interrupción 3), y 21 (interrupción 2).**Estos pines se pueden configurar para lanzar una interrupción en un valor LOW (0V), en flancos de subida o bajada (cambio de LOW a HIGH (5V) o viceversa), o en cambios de valor. Ver la función [attachInterrupt ()](http://arduino.cc/es/Reference/AttachInterrupt) para más detalles (Enríquez Herrador R, 2009, p.13.).
* **PWM: de 0 a 13.**Proporciona una salida PWM (Pulse Wave Modulación, modulación de onda por pulsos) de 8 bits de resolución (valores de 0 a 255) a través de la función [analogWrite ()](http://arduino.cc/es/Reference/AnalogWrite) (Enríquez Herrador R, 2009, p.13.).
* **SPI: 50 (SS), 51 (MOSI), 52 (MISO), 53 (SCK).** Estos pines proporcionan comunicación SPI, usando la [librería SPI](http://arduino.cc/en/Reference/SPI) (Enríquez Herrador R, 2009, p.13.).
* **LED: 13.**Hay un LED integrado en la placa conectado al pin digital 13, cuando este pin tiene un valor HIGH(5V) el LED se enciende y cuando este tiene un valor LOW(0V) este se apaga (Enríquez Herrador R, 2009, p.13.).

El Mega posee 16 entradas analógicas, y cada una de ellas proporciona 10bits de resolución (1024 valores). Se mide desde 0V a 5V por defecto, es posible cambiar la cota superior usando el pin **AREF** y la función **[analogReference ()](http://arduino.cc/es/Reference/AnalogReference).** (Enríquez Herrador R, 2009, p.13.).

Algunos pines tienen funciones especializadas:

* **I2C: 21 (SCL y 20 (SDA)).**Se usa la librería Wire para el protocolo de comunicaciones I2C (TWI)
* **AREF.** Se usa  [analogReference ()](http://arduino.cc/es/Reference/AnalogReference) para voltaje de referencia para las entradas analógicas.
* **Reset.** Para el reinicio del microcontrolador se suministrar un valor LOW (0V).

#### COMUNICACIONES

Tiene una serie de instalaciones para la comunicación con el ordenador, la otra tabla, u otros microcontroladores, ofrece cuatro UART para TTL de 5V de comunicación serie, un ATmega16U2 para canales de mesa uno de los cuales es a través de USB y proporciona un puerto COM virtual para el software en el equipo, para máquinas de Windows necesitarán un archivo .inf, pero para las máquinas OSX y Linux reconoce la junta como un puerto COM de forma automática.

El software de Arduino (IDE) incluye un monitor de serie que permite que los datos de texto simples puedan ser enviados desde y hacia el tablero. Los LEDs TX y RX de la placa parpadean al momento de transmitir datos a través de la ATmega8U2 / ATmega16U2 chip y conexión USB al ordenador, pero no parpadea cuando en los pines 0 y 1 se realiza una comunicación en serie.

Lo que permite una comunicación en serie en cualquiera de los pines que son digitales, es la biblioteca SoftwareSerial.

El Mega 2560 puede soportar una comunicación SPI y TWI. El software de Arduino (IDE) incluye la librería Wire para poder simplificar a utilización del bus TWI.

#### PROGRAMACIÓN

El Arduino Mega puede ser programado con el software Arduino (IDE), viene precargado [un gestor de arranque (bootloader)](http://arduino.cc/es/Tutorial/Bootloader) el cual permite que se pueda cargar nuevo código sin la necesidad de un programador por hardware externo. La utilización del protocolo STK500 original sirve para su comunicación.

Para evitar el gestor de arranque se puede programar directamente el microcontrolador a través del puerto ICSP (In Circuit Serial Programming)

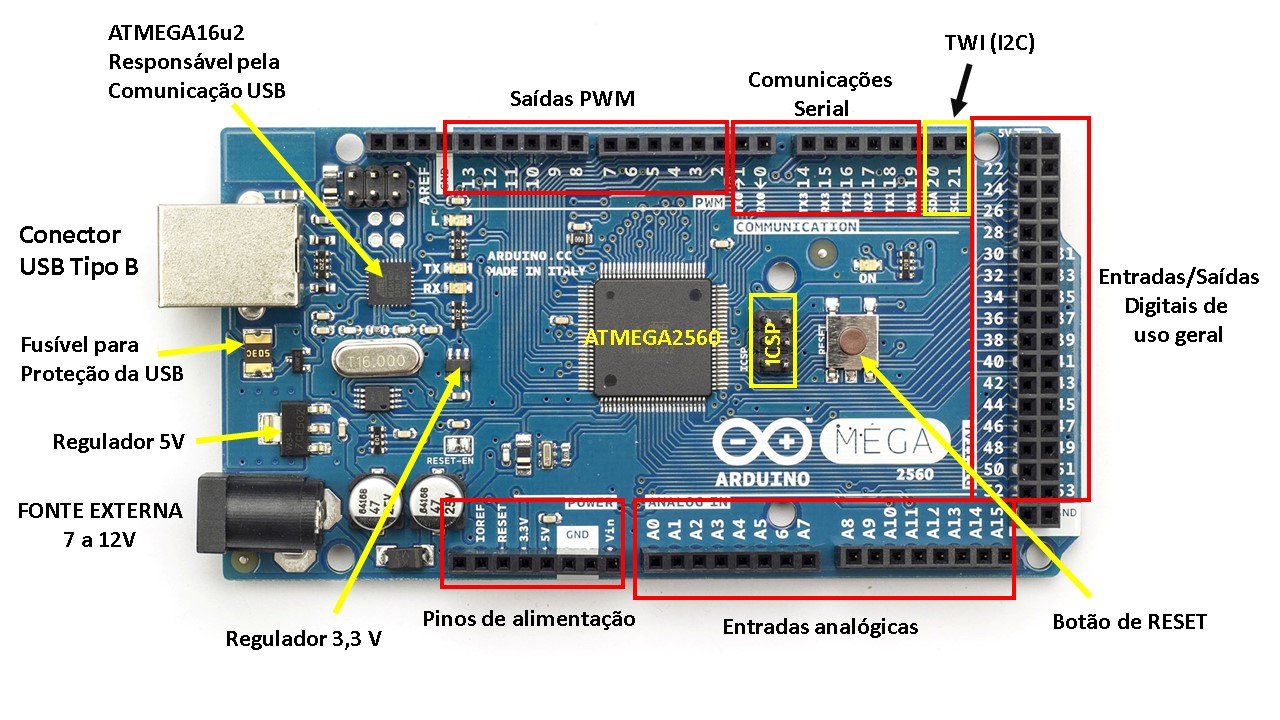
### SOFTWARE

#### INSTALACIÓN DEL SOFTWARE

En esta sección se describe como instalar el software Arduino en un ordenador con un sistema operativo Windows y la conexión de la placa al ordenador.

##### [**Obtener una placa Arduino y el cable USB**](https://translate.googleusercontent.com/translate_c?act=url&depth=1&hl=es&ie=UTF8&prev=_t&rurl=translate.google.com.ec&sl=en&tl=es&u=https://www.arduino.cc/en/Guide/Windows&usg=ALkJrhgOIY61-xcJdVjxyeDZ8aKfuVKfqw#toc1)

En este caso conseguir la placa Arduino mega 2560 y su respectivo cable USB



**Figura 10-1:** Descripción de componentes de la placa Arduino Mega2560

**Fuente**: (Datasheet Arduino)

##### [**Descargue el software de Arduino (IDE)**](https://translate.googleusercontent.com/translate_c?act=url&depth=1&hl=es&ie=UTF8&prev=_t&rurl=translate.google.com.ec&sl=en&tl=es&u=https://www.arduino.cc/en/Guide/Windows&usg=ALkJrhgOIY61-xcJdVjxyeDZ8aKfuVKfqw#toc2)

Se puede descargar el programa de la página oficial de Arduino, en la siguiente página web: https://www.arduino.cc/en/Main/Software

##### [**Instalar**](https://translate.googleusercontent.com/translate_c?act=url&depth=1&hl=es&ie=UTF8&prev=_t&rurl=translate.google.com.ec&sl=en&tl=es&u=https://www.arduino.cc/en/Guide/Windows&usg=ALkJrhgOIY61-xcJdVjxyeDZ8aKfuVKfqw#toc3) **Drivers USB**

Si se utiliza Arduino USB se necesita descargar los drivers para el chip FTDI de la placa, y se pueden encontrar en el directorio drivers/FTDI.

Después conecte la placa y se mostrará el asistente para añadir un nuevo hardware.

##### [**Conectar**](https://translate.googleusercontent.com/translate_c?act=url&depth=1&hl=es&ie=UTF8&prev=_t&rurl=translate.google.com.ec&sl=en&tl=es&u=https://www.arduino.cc/en/Guide/Windows&usg=ALkJrhgOIY61-xcJdVjxyeDZ8aKfuVKfqw#toc4) **la placa**

La fuente de alimentación se logra seleccionar mediante el jumper entre conectores del USB.

Para la alimentación desde el puerto USB se coloca el jumper en los 2 pines más cercanos al conector USB.

Para la alimentación desde una fuente externa con un voltaje de 6 a 12v se coloca el jumper en los 2 pines cercanos al conector de alimentación.

Conectar la placa al puerto USB del ordenador.



Figura 11-1: Conexión del cable USB a Arduino

**Fuente:** (Arduino User Guide, 2009)

El asistente para añadir el nuevo hardware, se hace click en: no this time y se procede al siguiente paso.

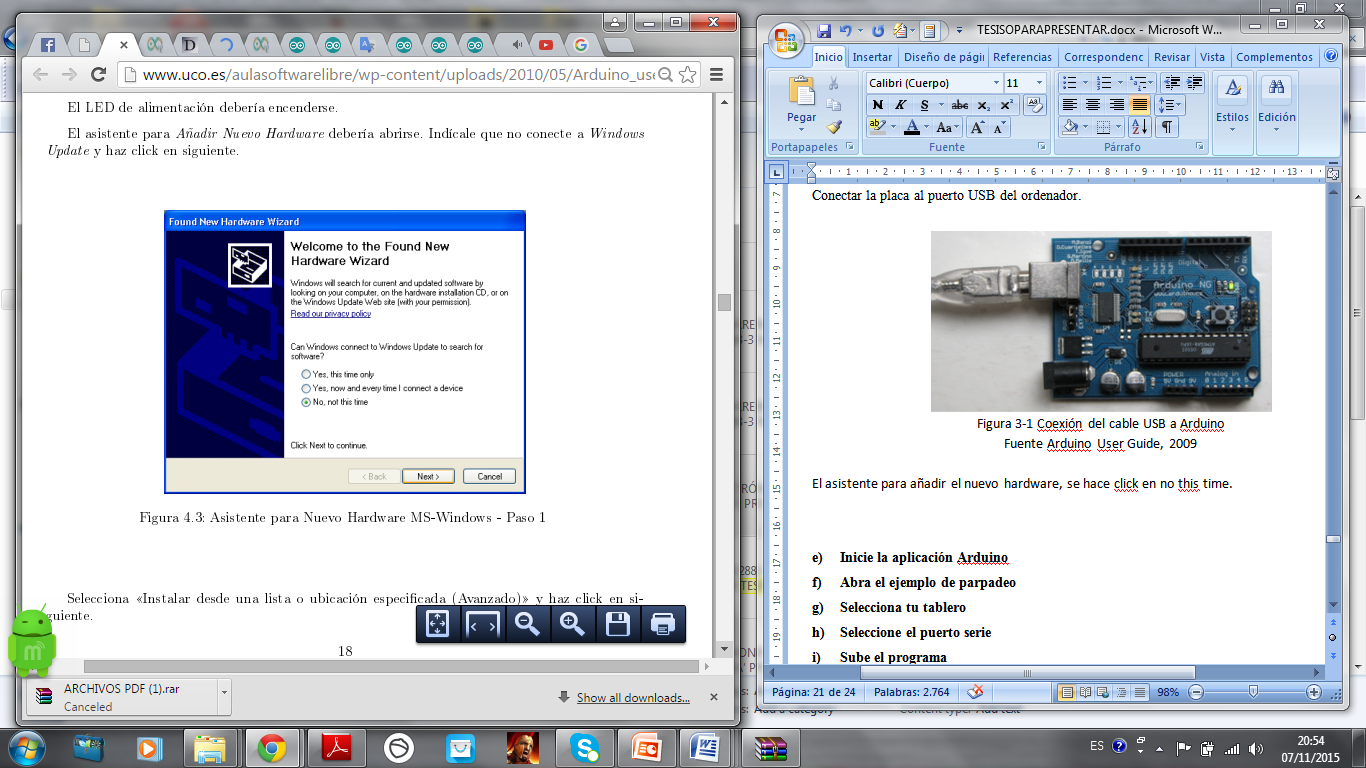
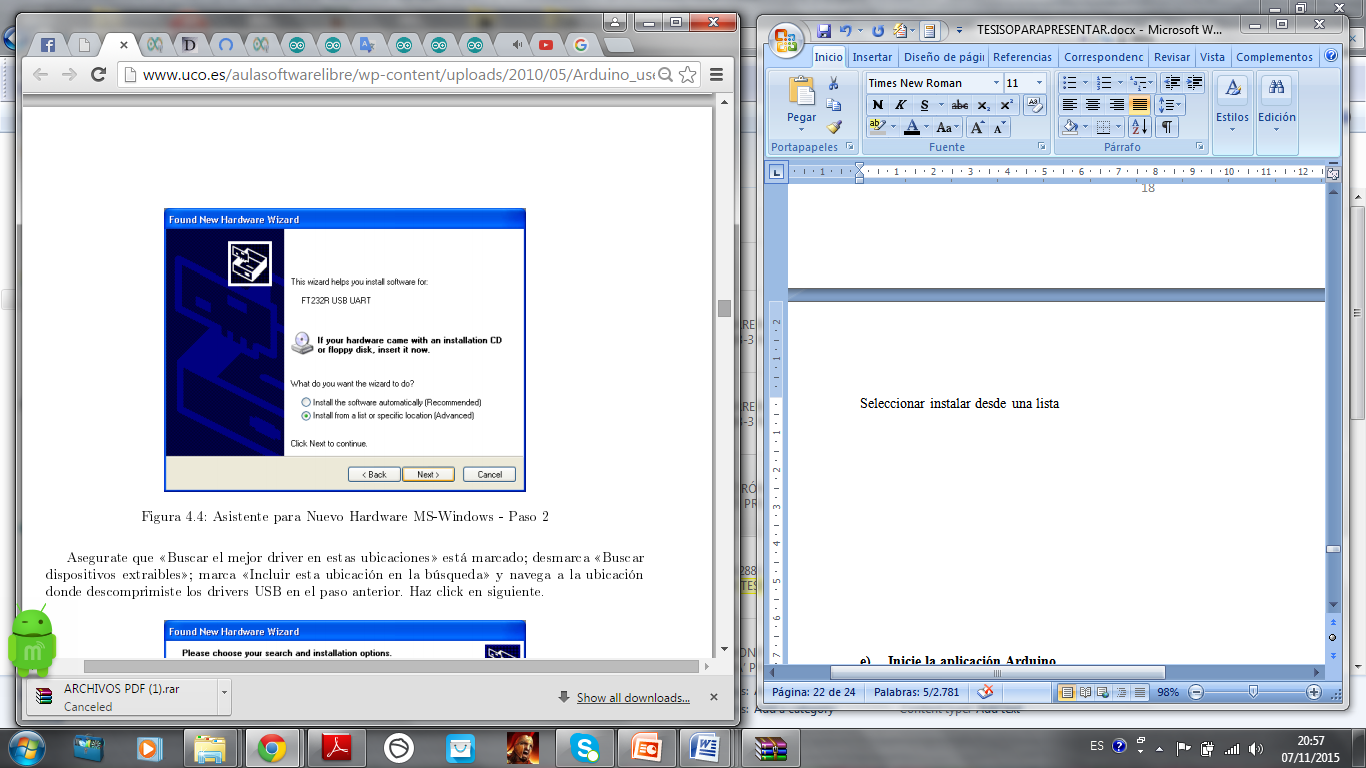


Figura 12-1: Asistente para el nuevo hardware paso1

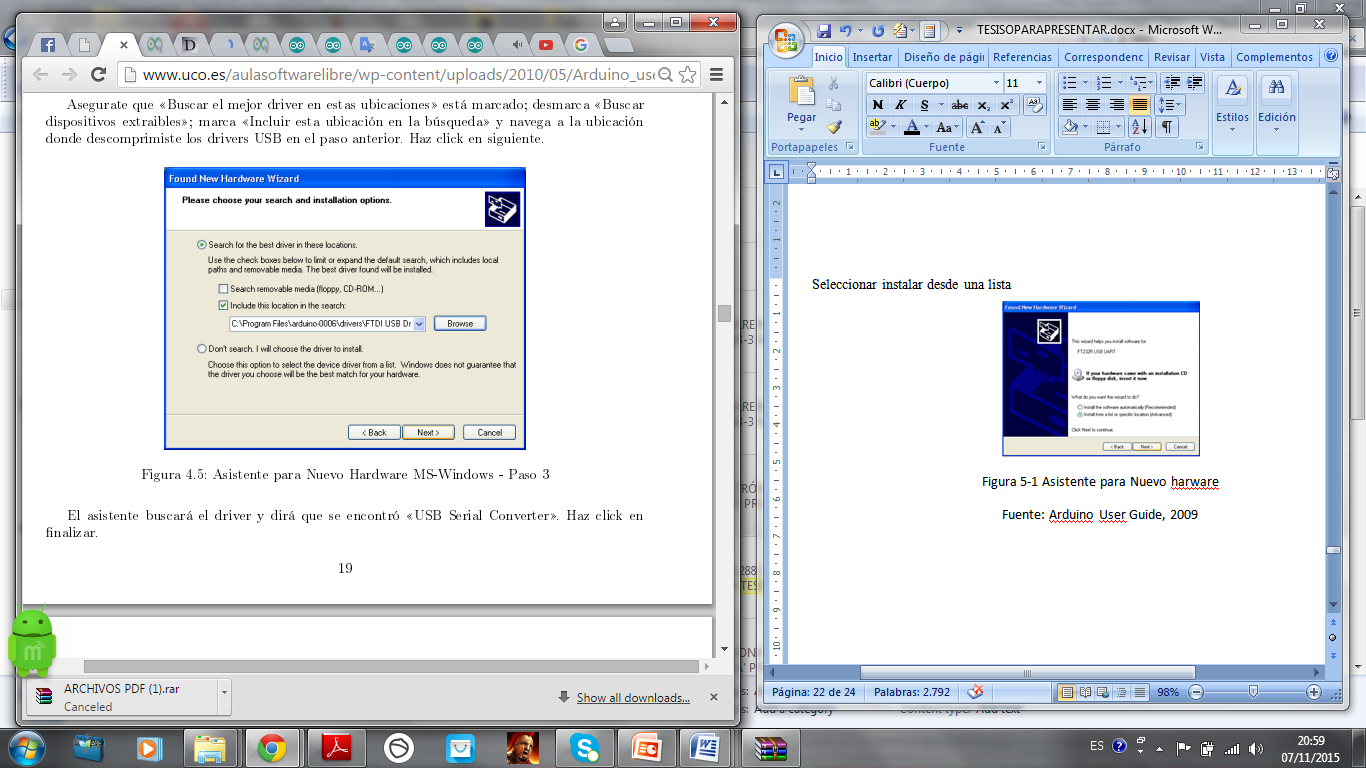
Fuente: (Arduino User Guide , 2009)

Seleccionar instalar desde una lista



**Figura 13-1:** Asistente para Nuevo hardware paso 2

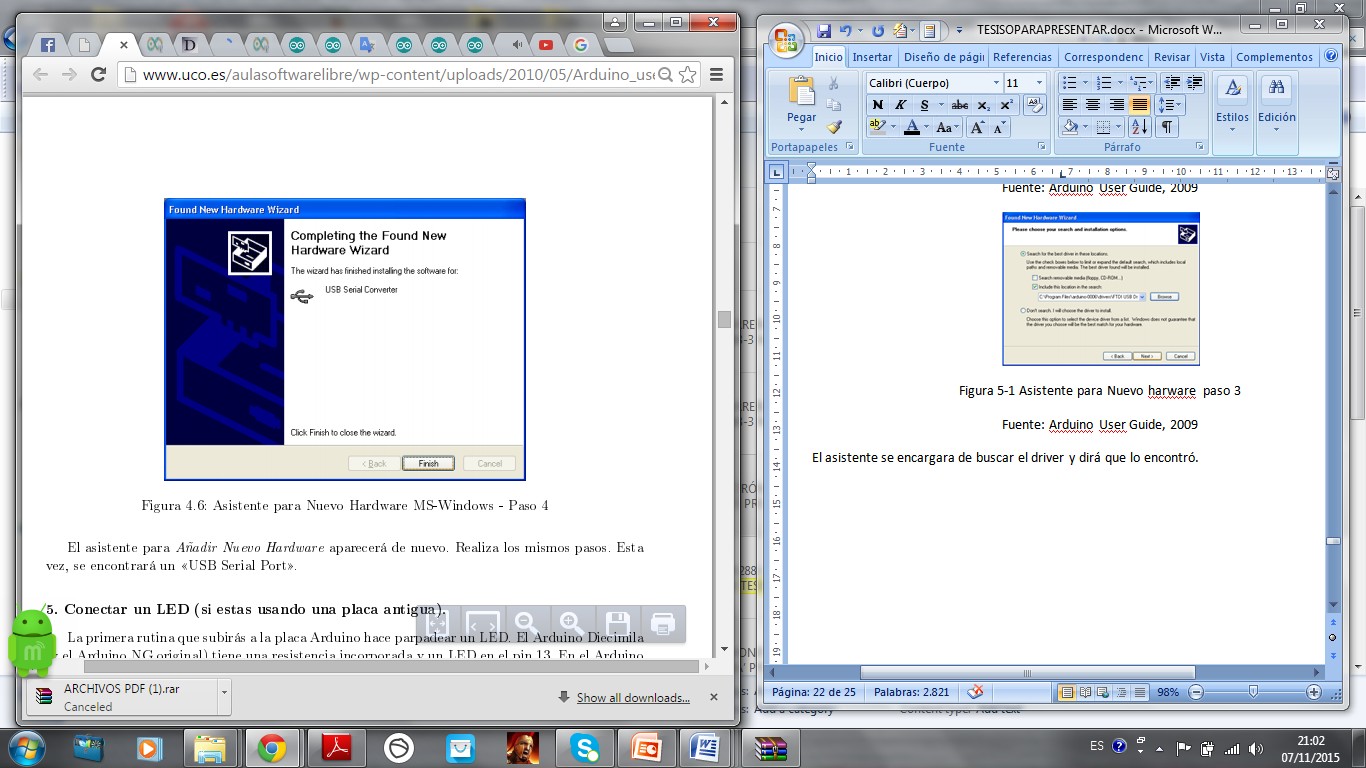
**Fuente:** (Arduino User Guide, 2009)



**Figura 14-1:** Asistente para Nuevo hardware paso 3

**Fuente:(** Arduino User Guide, 2009)

El asistente se encargara de buscar el driver y dirá que lo encontró.



**Figura 15-1:** Asistente para Nuevo hardware paso 4

**Fuente**: (Arduino User Guide, 2009)

##### [**Ejecutar**](https://translate.googleusercontent.com/translate_c?act=url&depth=1&hl=es&ie=UTF8&prev=_t&rurl=translate.google.com.ec&sl=en&tl=es&u=https://www.arduino.cc/en/Guide/Windows&usg=ALkJrhgOIY61-xcJdVjxyeDZ8aKfuVKfqw#toc6) **el entorno Arduino**

Abrir la carpeta Arduino y abrir la aplicación.

## LCD CON MÓDULO I2C

El Módulo serial IIC-I2C-TWI-SPI es una manera más sencilla de controlar un display LCD desde cualquier Arduino.

El lado izquierdo tiene 4 pines de las cuales dos son para la alimentación (Vcc y GND), y las otras dos son la interfaz I2C (SDA y SCL).

El potenciómetro de la placa es para el ajuste de contraste de la pantalla, y el puente en el lado opuesto permite que la luz de fondo sea controlado por el programa o permanecer apagados.

### Características:

**Voltaje de operación: ≈5v**

**Peso: 6g**

**Tamaño: 54mm/19 mm (largo/ancho)**

**Pines: 16**

**Dirección I2C: 0X20~0X27**

**Compatible con**[Display 16x2](http://electrocrea.com/collections/displays-pantallas/products/display-16x2-fondo-azul-o-verde?variant=7505243843" \t "_blank" \o "Display 16x2)***y***[Display 20x4](http://electrocrea.com/collections/displays-pantallas/products/display-20x4-fondo-azul-o-verde?variant=7507682627)

# CAPÍTULO II

# MARCO METODOLÓGICO

## SELECCIÓN DE LA PLACA ARDUINO

Se utilizó el método comparativo con el cual se puede observar las diferencias entre dos o más placas de Arduino para poder escoger la mejor placa que servirá con mayor eficacia en el desarrollo del prototipo.

Para esto se procede a comparar las placas de Arduino que serían aptas para la implementación en la siguiente tabla comparativa:

Tabla 1‑2: Tabla comparativa de Arduino.

|  |  |  |  |
| --- | --- | --- | --- |
| **Característica de Arduino** | **UNO** | **Mega 2560** | **Leonardo** |
| Tipo de microcontrolador | Atmega 328 | Atmega 2560 | Atmega 32U4 |
| Velocidad de reloj | 16 MHz | 16 MHz | 16 MHz |
| Pines digitales de E/S | 14 | 54 | 20 |
| Entradas analógicas | 6 | 16 | 12 |
| Salidas analógicas | 0 | 0 | 0 |
| Memoria de programa (Flash) | 32 Kb | 256 Kb | 32 Kb |
| Memoria de datos (SRAM) | 2 Kb | 8 Kb | 2.5 Kb |
| Memoria auxiliar (EEPROM) | 1 Kb | 4 Kb | 1 Kb |

**Realizado por:** Solo Arduino

**Fuente**: (Arduino Blog, 2014)

Para realizar a comparación asignamos valores en porcentaje siendo:

Tabla 2-2: Escala cuantitativa

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| VALORACIÓN | | | | |
| 0 | 1 | 2 | 3 | 4 |
| 0 | 25% | 50% | 75% | 100% |

Realizado por: Luis U, Néstor E.

Estas valoraciones se las califica en la siguiente tabla:

Tabla 3-2: Escala cualitativa

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | 3 | 2 | 1 | 0 |
| Muy adecuado | Adecuado | Poco adecuado | Inadecuado | Nada adecuado |

Realizado por: Luis U, Néstor E.

Teniendo estas tablas procedemos a dar valores cuantitativos a la tabla de comparación de los diferentes tipos de Arduinos para poder observar cual es el mejor y más apto para la utilización en el prototipo

Tabla 4‑2: Tabla cuantitativa de Arduino

|  |  |  |  |
| --- | --- | --- | --- |
| **Característica de Arduino** | **UNO** | **Mega 2560** | **Leonardo** |
| Tipo de microcontrolador | 4 | 3 | 3 |
| Velocidad de reloj | 4 | 4 | 4 |
| Pines digitales de E/S | 2 | 4 | 3 |
| Entradas analógicas | 2 | 4 | 3 |
| Salidas analógicas | 0 | 0 | 0 |
| Memoria de programa (Flash) | 1 | 4 | 1 |
| Memoria de datos (SRAM) | 1 | 4 | 2 |
| Memoria auxiliar (EEPROM) | 1 | 4 | 1 |
| TOTAL | 15 | 27 | 17 |

Realizado por: Luis U, Néstor E.

Obteniendo estos datos podemos interpretarlos tomando un valor mayor como referencia para realizar la regla de 3 y sacar sus respectivos porcentajes.

**Interpretación:**

Tabla 5‑2: Interpretación

|  |  |  |  |
| --- | --- | --- | --- |
| Puntos máximos | UNO | MEGA 2560 | LEONARDO |
| 30 | 15 | 27 | 17 |
| 100% | 50% | 90% | 56,67% |

Realizado por: Luis U, Néstor E.

Para realizar la comparación se ha tomado tres elementos que son desarrollados por la empresa Arduino Software, cada uno de ellos tiene características similares de entre los tres se ha escogido el Arduino Mega 2560 ya que haciendo el análisis de sus características cuenta con un 90% y es el que más porcentaje tiene.

Figura 1-2: Gráfico porcentajes para la elección de Arduino

Figura: Realizado por: Luis U, Néstor E.

Dentro de la puntuación cuenta con 27 puntos y su promedio entre las 9 características dio un total de 3 el cual se encuentra entre el rango de adecuado - muy adecuado.

También debemos tener en cuenta que Arduino Mega 2560 tiene un bajo consumo de energía y una memoria mayor al resto y eso nos ayuda al momento de realizar más líneas de programación.

## SELECCIÓN DEL GPS

Para la selección del GPS de la misma forma se utilizó el método comparativo que ayudara a ver las diferencias de cada módulo GPS para optar por escoger el mejor y más adecuado para el prototipo.

Tabla 6‑2: Tabla comparativa de GPS

|  |  |  |
| --- | --- | --- |
| **Características** | **GPS NEO6M** | **GPS 18 LVC** |
| Canales de lectura | 66 | 12 |
| Precisión tiempo de velocidad | 0,1 m/s | 0,05 m/s |
| tiempo de lectura en frio | 36 seg | 45seg |
| tiempo de lectura en caliente | 1seg | 2seg |
| consumo de energía | poco | poco |

Realizado por: Luis U, Néstor E.

Teniendo en cuenta los valores de la tabla 2-2 se procede a dar valores cuantitativos a la tabla de comparación de los GPS para poder observar cual es el mejor y más apto para la utilización en el prototipo.

Tabla 7‑2: Tabla Cuantitativa de GPS

|  |  |  |
| --- | --- | --- |
| CARACTERÍSTICAS | GPS NEO6M | GPS 18 LVC |
| Canales de lectura | 4 | 1 |
| Precisión tiempo de velocidad | 4 | 2 |
| tiempo de lectura en frio | 4 | 3 |
| tiempo de lectura en caliente | 4 | 3 |
| consumo de energía | 4 | 4 |
| TOTAL | 20 | 13 |

Realizado por: Luis U, Néstor E.

Obteniendo estos datos se procede a interpretarlos tomando un valor mayor como referencia para realizar la regla de 3 y sacar sus respectivos porcentajes.

**Interpretación:**

Tabla 8‑2: Interpretación

|  |  |  |
| --- | --- | --- |
| Puntos máximos | NEO 6M | GPS 18 LVC |
| 25 | 20 | 13 |
| 100% | 80% | 52% |

Realizado por: Luis U, Néstor E.

Para realizar la comparación se ha tomado 2 elementos, cada uno de ellos tiene características similares de entre los tres se ha escogido el GPS neo6m ya que haciendo el análisis de sus características cuenta con un 80% y es el que más porcentaje tiene.

Dentro de la puntuación cuenta con 20 puntos y su promedio entre las 5 características dio un total de 4 el cual se encuentra muy adecuado para escogerlo.

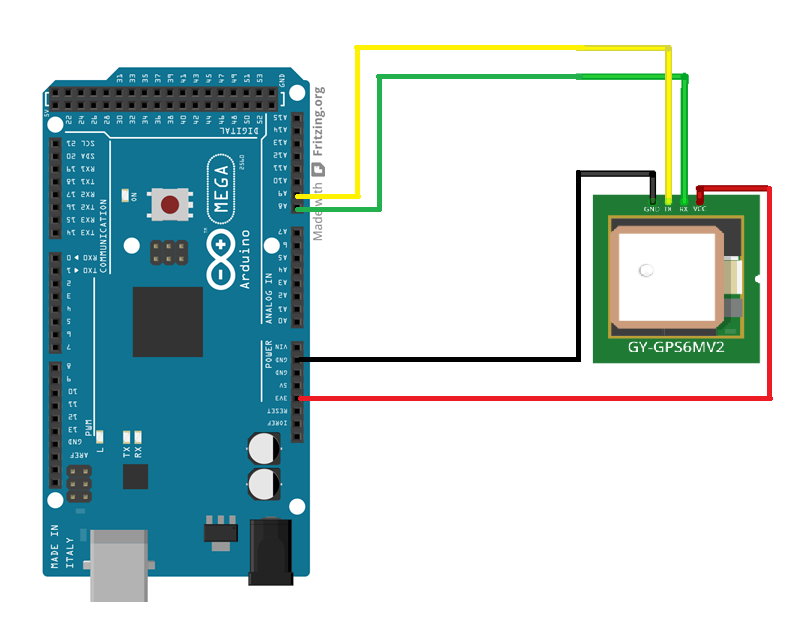
También se debes tener en cuenta que GPS NEO6m es adaptable para Arduino mega 2560 y lectura de datos no es tan compleja permitiéndonos observar las coordenadas. .

Figura 2-2: Gráfico de porcentajes para la elección del GPS

**Fuente:** Realizado por Luis U, Néstor E.

## CONEXIONES GPS CON LA TARJETA ARDUINO MEGA

Una vez ya escogido los materiales procedemos a su conexión.

****

**Figura 3-2:** Conexión del módulo GPS con el Arduino MEGA 2560

**Fuente:** (Electrotec electrónica y robótica)

Una vez ya realizada la elección de la placa Arduino y el GPS, en esta sección se procedió a conectar el GPS Neo-6m al Arduino .

Conectar:

Tx del GPS al pin A9 del Arduino

Rx del GPS al pin A8 del Arduino

GND del GPS al GND del Arduino

Vcc del GPS al 3v3 del Arduino

## Conexiones Modulo de Tarjeta SD con la Tarjeta Arduino

Se escogió el modulo Sd card para predefinir las coordenadas de las paradas específicas para que no ocupe la memoria de la atarjeta arduino y que luego por medio del programa arduino IDE lo compare con las coordenadas recividas por el gps.

****

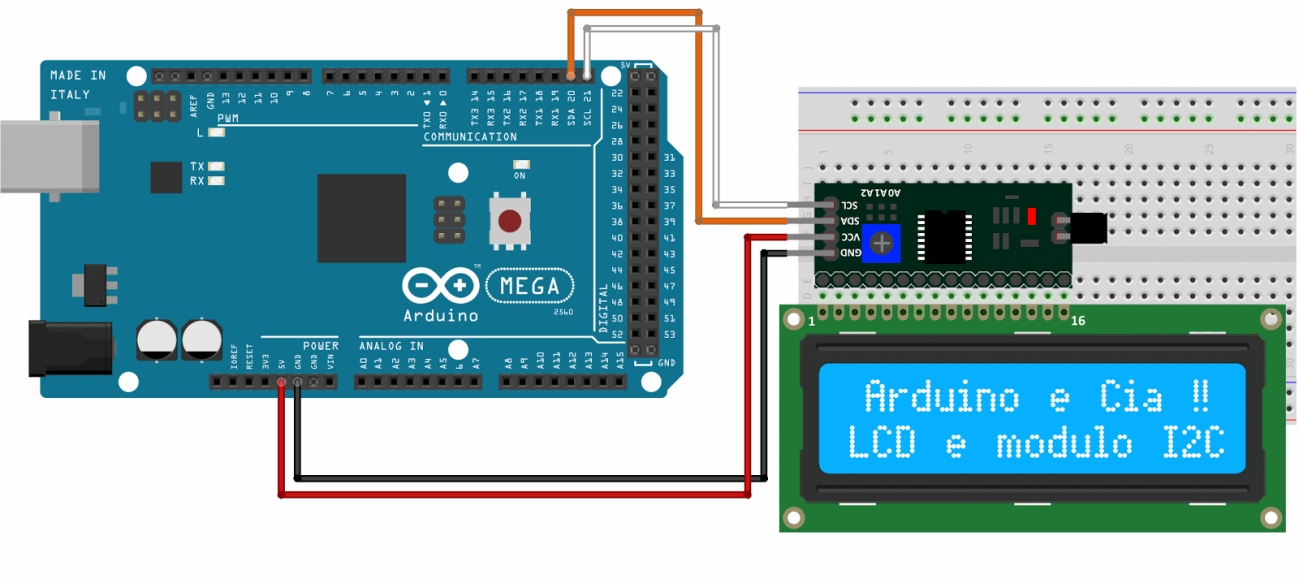
**Figura 4-2:** Conexión del módulo GPS con el Arduino MEGA 2560

**Fuente:** (Electrotec electrónica y robótica)

En este caso del módulo MICRO SD CARD Adapter es necesario hacer las siguientes conexiones:

- MOSI –> Pin 11 (La mayoría de las placas Arduino), para Arduino Mega: 51  
- MISO –> Pin 12 (La mayoría de las placas Arduino), para Arduino Mega: 50  
- CLK —-> Pin 13 (La mayoría de las placas Arduino), para Arduino Mega: 52  
- CS ——> Pin 4  
- 5V ——> 5V  
- GND –> GND

## Conexiones I2c lcd display con la Tarjeta Arduino

****

**Figura 5-2:** Conexión del módulo GPS con el Arduino MEGA 2560

**Fuente:** (Electrotec electrónica y robótica )

En este caso conectar:

SCL del lcd al pin SCL 21 del Arduino.

GND del lcd al pin GND del Arduino.

VCC del lcd al pin 5v del Arduino.

SDA del lcd al pin SDA 20 del Arduino.

Después de la conexión lo que se debe hacer es descargar una librería nueva para el manejo del display con I2C, el cual se llama LiquidCrystal\_I2C.

Descarga la librería[LiquidCrystal\_I2C](http://www.prometec.net/wp-content/uploads/2014/10/LiquidCrystal_I2C.zip) , y vamos a instalarla lo primero.

\\Programa\ImportarLibreria\Añadir \Librería

Buscar en el directorio de descargas, la librería LiquidCrystal\_I2C .zip y hacer doble click, para que se instale en el IDE.

Ahora se debe incluir la librería I2c que en Arduino se llama Wire, y como es estándar se incluye con:

\\Programa\ImportarLibreria\Wire

Ahora hay que incluir la librería LiquidCrystal normal y después la de LiquidCrystal\_I2C.

Estas 3 líneas se deben poner:

#include <Wire.h>

#include <LCD.h>

#include <LiquidCrystal\_I2C.h>

Ahora se define una variable con la dirección del display.

Después creamos una instancia del objeto LiquidCrystal\_I2C:

LiquidCrystal\_I2C             lcd (dir, 2, 1, 0, 4, 5, 6, 7);

Al que le pasa la dirección dir del display y varios números que indican que pines debe usar la librería para el display.

El resto queda así:

#include <Wire.h>

#include <LCD.h>

#include <LiquidCrystal\_I2C.h>

#define I2C\_ADDR    0x27

LiquidCrystal\_I2C             lcd(I2C\_ADDR,2, 1, 0, 4, 5, 6, 7);

void setup()

{

lcd.begin (16,2);    // Inicializa el display con 16 caracteres 2 líneas

lcd.setBacklightPin (3, POSITIVE);

lcd.setBacklight (HIGH);

lcd.home ();                   // ir a inicio

lcd.print ("Prometec.net");

lcd.setCursor (0, 1);        // ir a la segunda línea

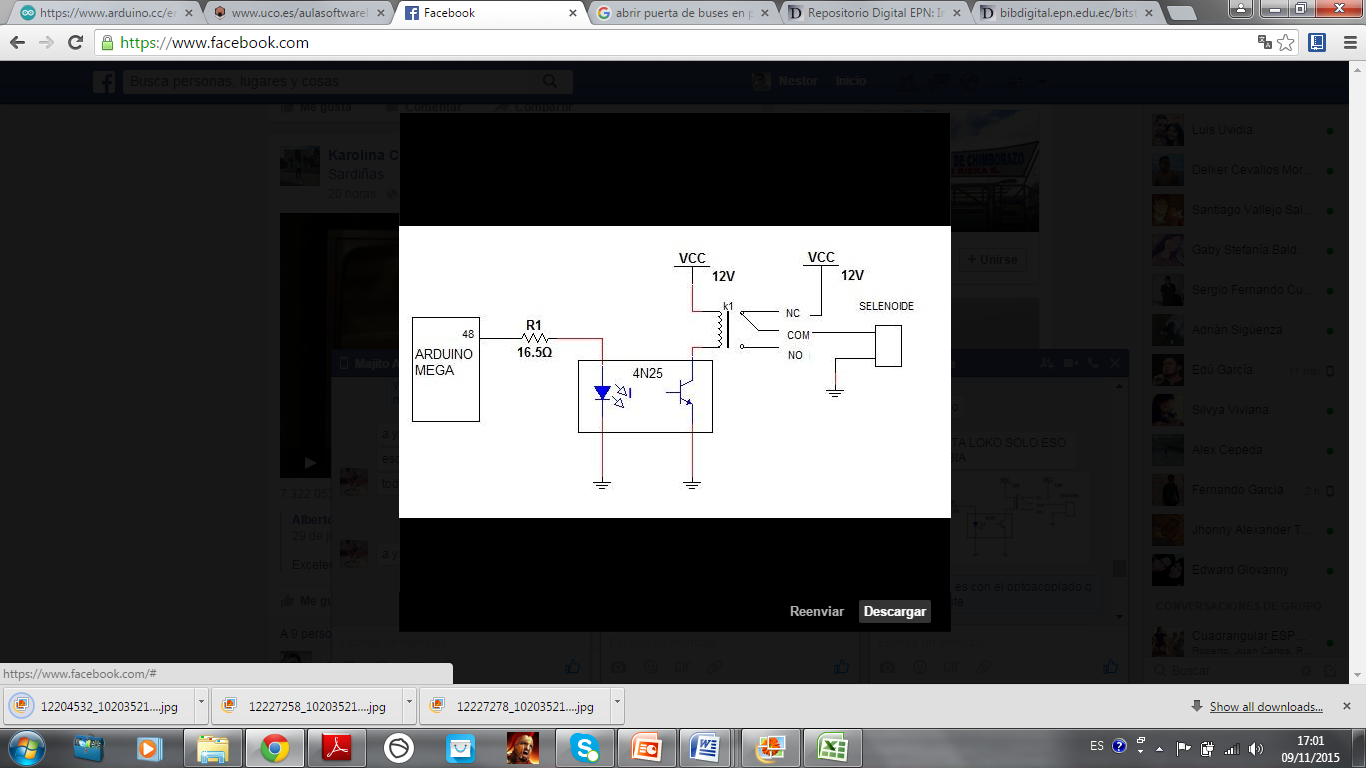
lcd.print ("Malpartida lib");

}

void loop ()

{}

## INTERFAZ DE POTENCIA



**Figura 6-2:** Interfaz de Potencia

**Fuente:** electrotec electrónica

Del pin 48 de salida de voltaje de 5v conectamos a una resistencia y procedemos a conectar a la válvula que permitirá abrir las puertas, y terminado eso podríamos hacer las pruebas y los análisis.

# CAPÍTULO III

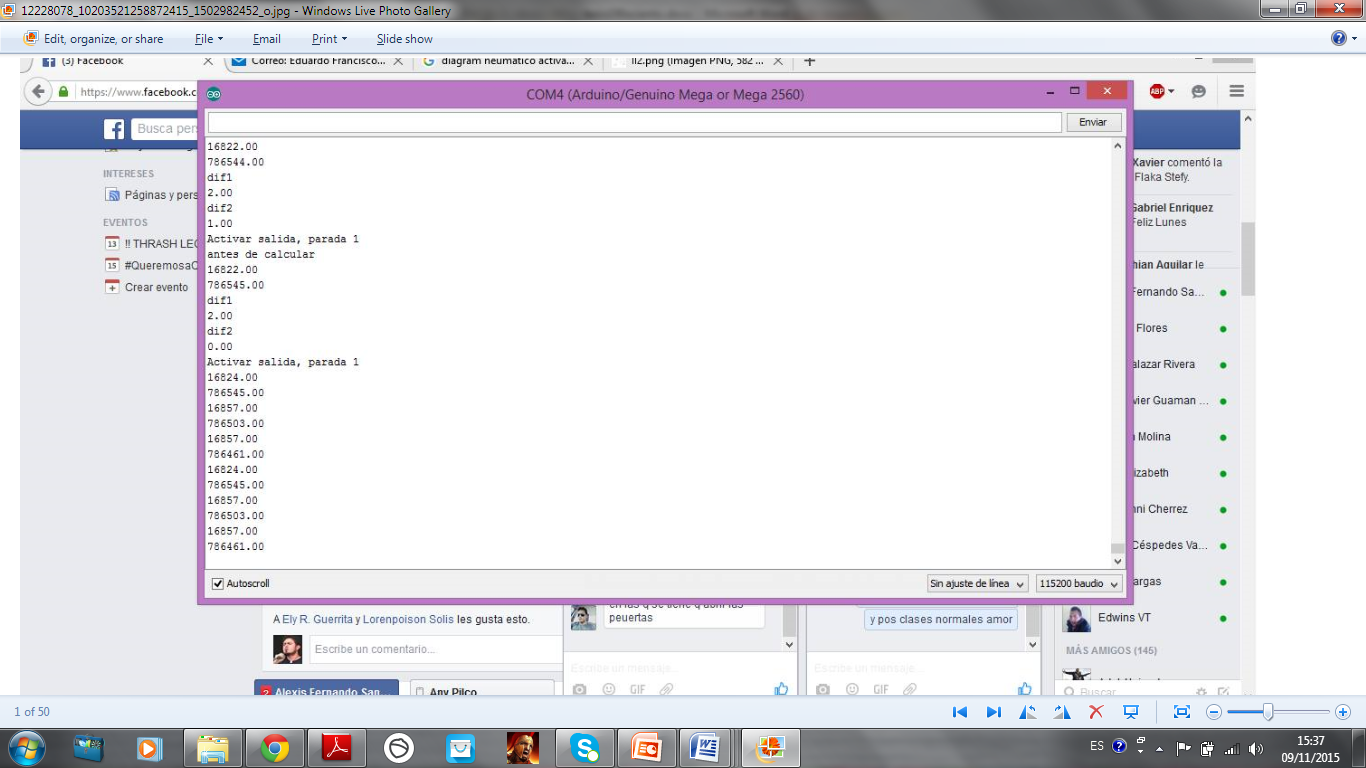
# MARCO DE RESULTADOS

## Coordenadas Predefinidas en la Sdcard

Este prototipo se empezó con Arduino uno, pero en el momento de arrancar el programa y poner las líneas de programación requeridas se pudo dar cuenta que contaba con poca memoria para comparar las coordenadas pero tenía poca memoria para las líneas de programación y se optó por cambiar de placa y adquirir la tarjeta Arduino Mega 2560.

Se realizó nuevamente la programación y se pudo dar cuenta que con esta tarjeta se podría poner más líneas de programación y no dio más inconvenientes.

Después se procedió a guardar 3 coordenadas de ejemplos en la sdcard para compararlas con las que de lectura el GPS, los cuales son [16824.00: 786545.00] [16857.00: 786503.00] [16857.00:786461.00]



**Figura 1-3:** Coordenadas predefinidas en la SDcard

**Fuente:** Realizado por Luis U, Néstor E.

En la SDcard se debe asegurar que al momento de guardar las coordenadas no guardar con espacios en blanco a lado de las coordenadas porque el programa lo lee como carácter y puede salir errores que no se entendería el porqué del error, así que se debe prestar mucha atención al momento de guardar las coordenadas y al momento de realizar la programación.

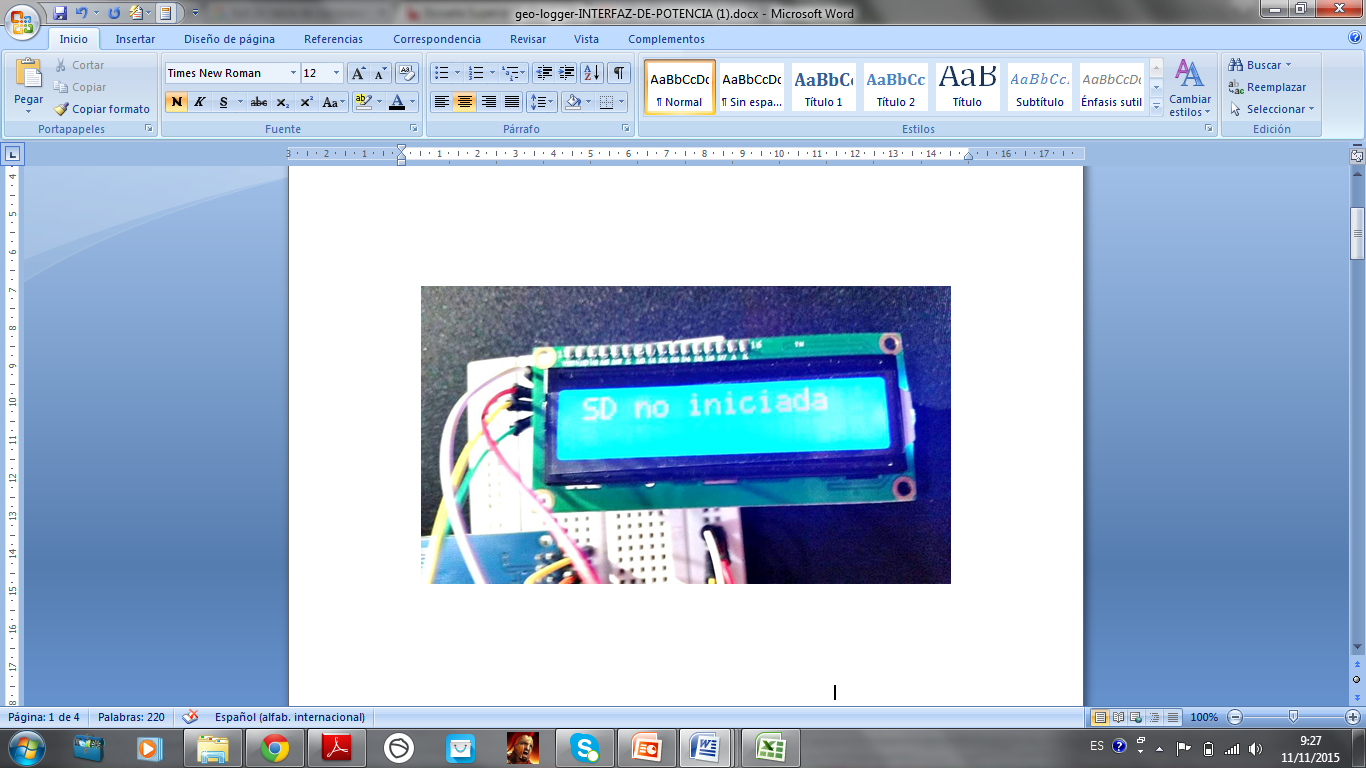
## Lectura de Datos

El GPS manda las coordenadas de ubicación del automóvil, en el programa IDE da lectura a estas coordenadas, las cuales se podrán visualizar en la Lcd i2c.

Estas coordenadas permiten verificar si el automóvil se encuentra en la parada específica para que se puedan abrir las puertas.

La siguiente pantalla muestra el resultado cuando no está bien ubicada la SD card en el Module SD Card de Arduino.

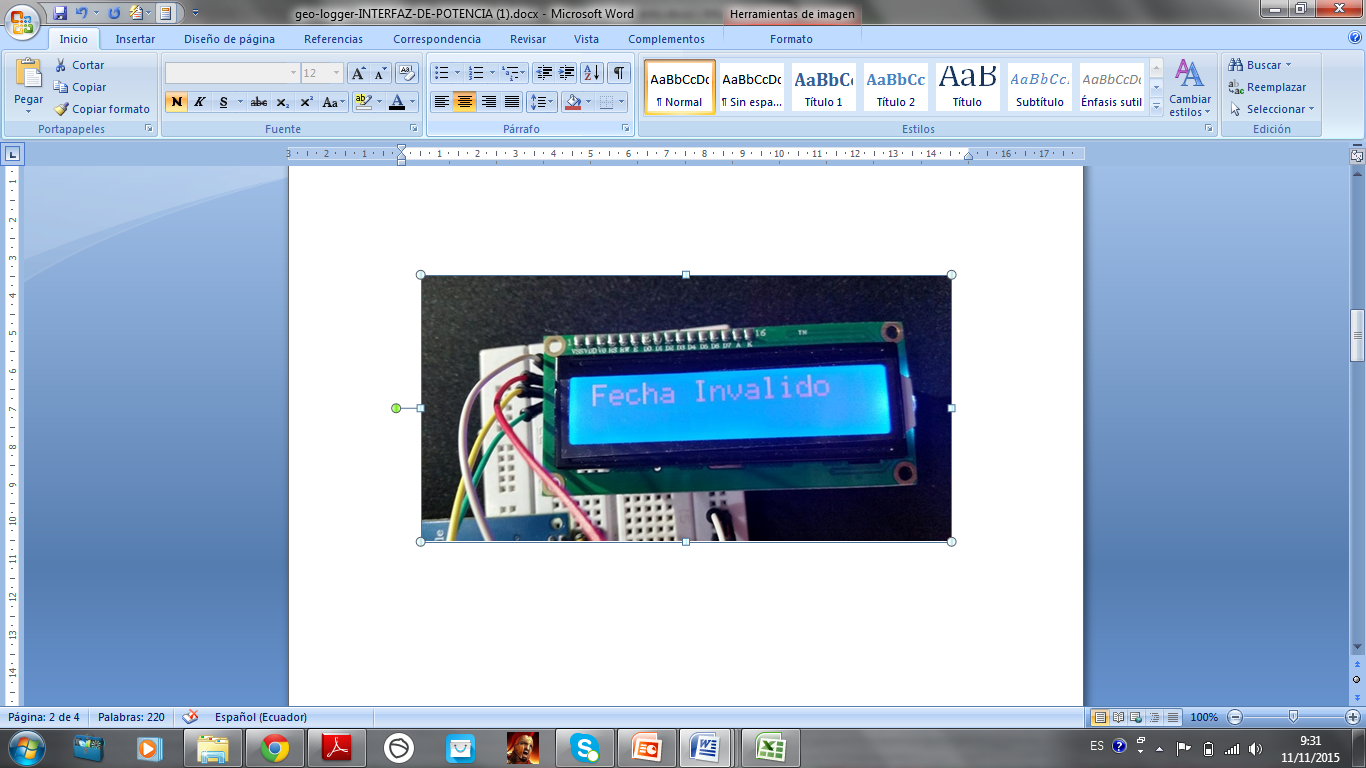
El Arduino no posee la base de datos de longitud y latitud establecidas de las paradas predefinidas para empezar a comparar coordenadas emitidas por el GPS. Por esa razón se debe asgurar de poner bien la SDcard.



**Figura 2-3:** Lectura de la Sdcard

**Fuente:** Realizado por Luis U, Néstor E

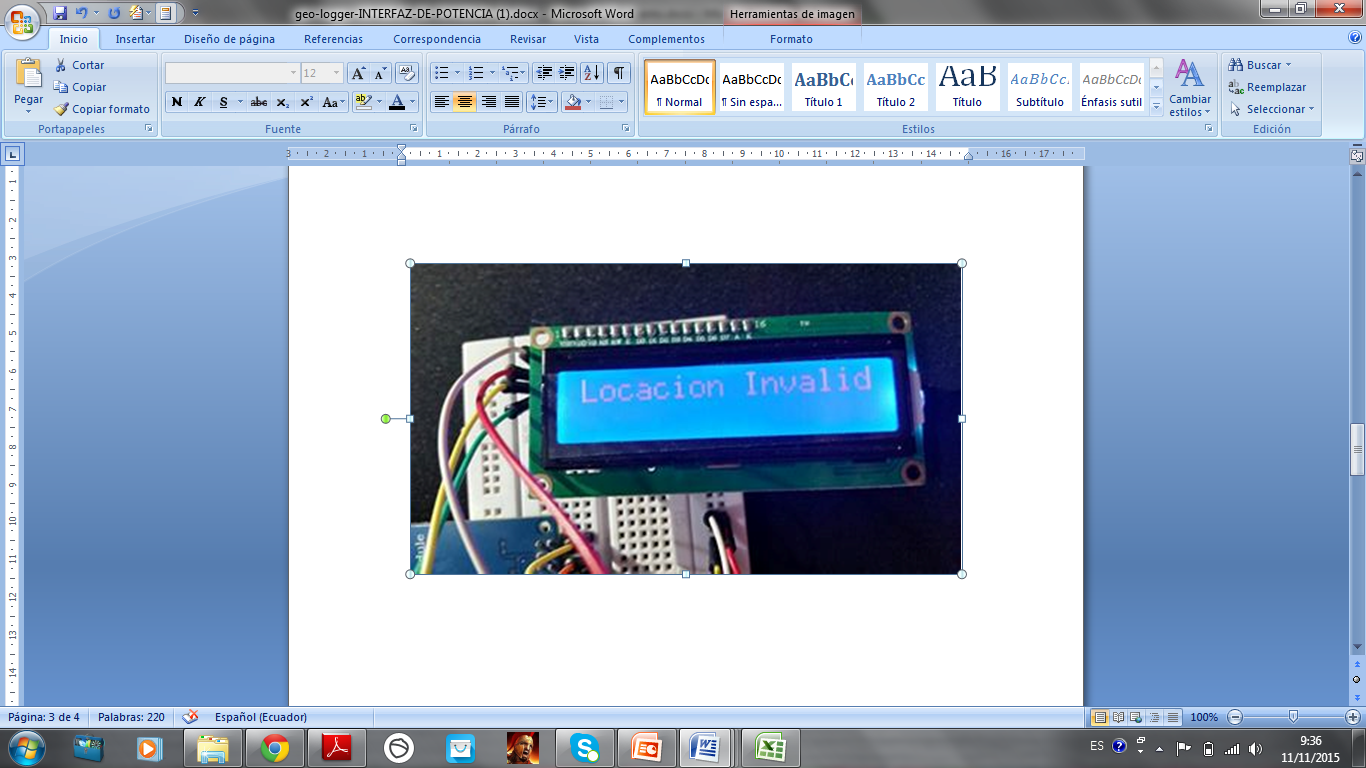
La siguiente pantalla aparecerá si el GPS no está entregando datos o está mal conectado los terminales Rx y Tx del GPS con los terminales declarados Rx y Tx del Arduino Mega. Por esta razón se debe asegurar de conectar bien el gps.



**Figura 3-3:** Lectura de la Sdcard

**Fuente:** Realizado por Luis U, Néstor E

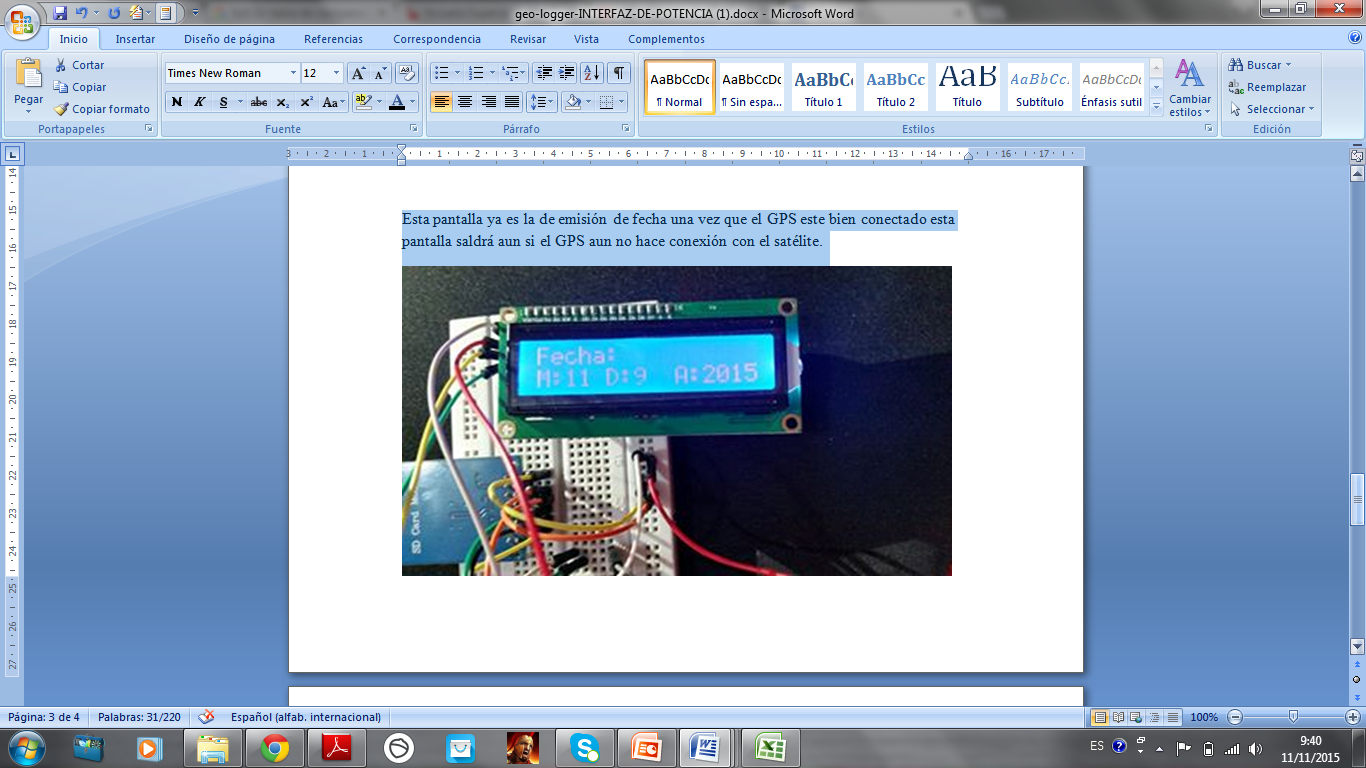
Esta pantalla indica que el GPS aún no ha realizado conexión con el satélite y no está emitiendo coordenadas de latitud y longitud.



**Figura 4-3:** Lectura de las coordenadas

**Fuente:** Realizado por Luis U, Néstor E

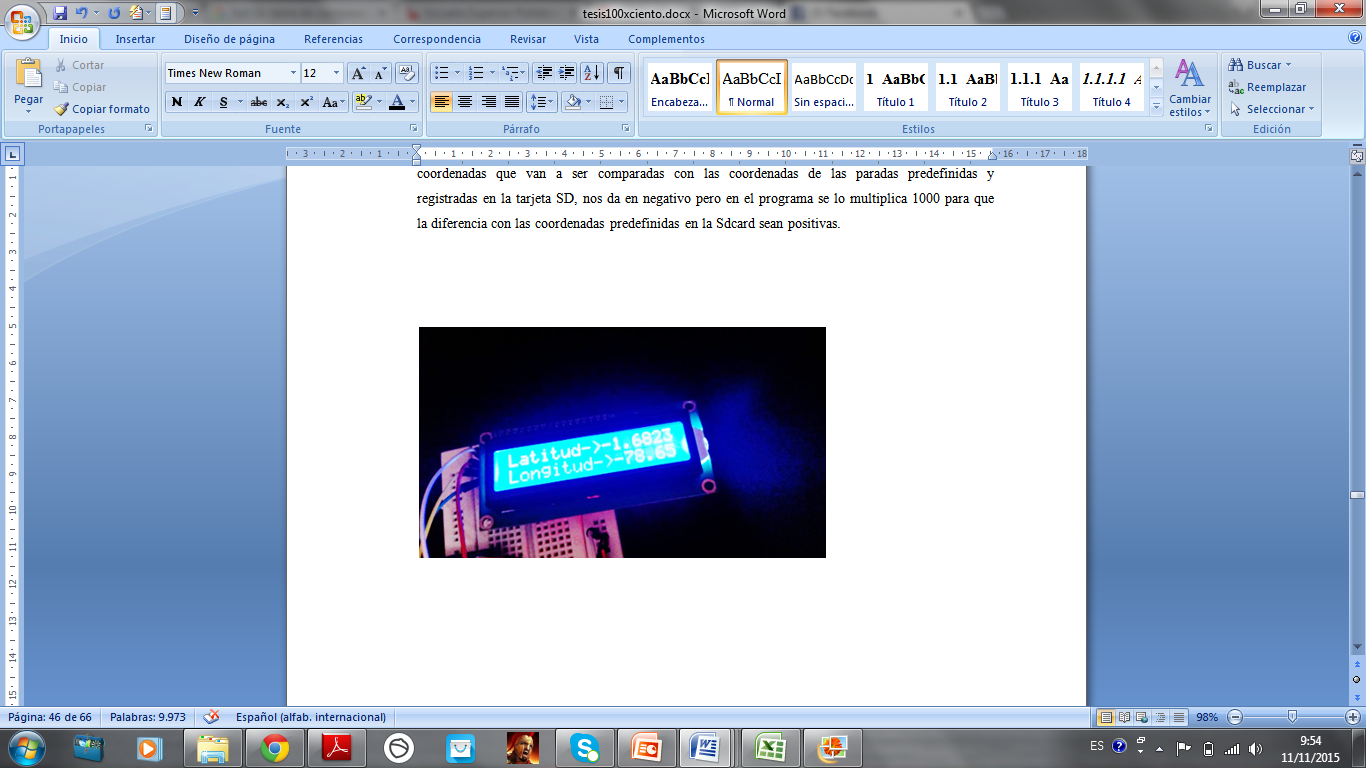
Esta pantalla ya es la de emisión de fecha una vez que el GPS este bien conectado esta pantalla saldrá aun si el GPS aun no hace conexión con el satélite.



**Figura 5-3:** Lectura del GPS

**Fuente:** Realizado por Luis U, Néstor E

Esta pantalla nos da ya las coordenadas de latitud y longitud emitidas por el GPS son las coordenadas que van a ser comparadas con las coordenadas de las paradas predefinidas y registradas en la tarjeta SD, da en negativo pero en el programa se lo multiplica 10000 y por -1 para que la diferencia con las coordenadas predefinidas en la Sdcard sean positivas.



**Figura 6-3:** Lectura del GPS

**Fuente:** Realizado por Luis U, Néstor E

## Diferencia de Lecturas

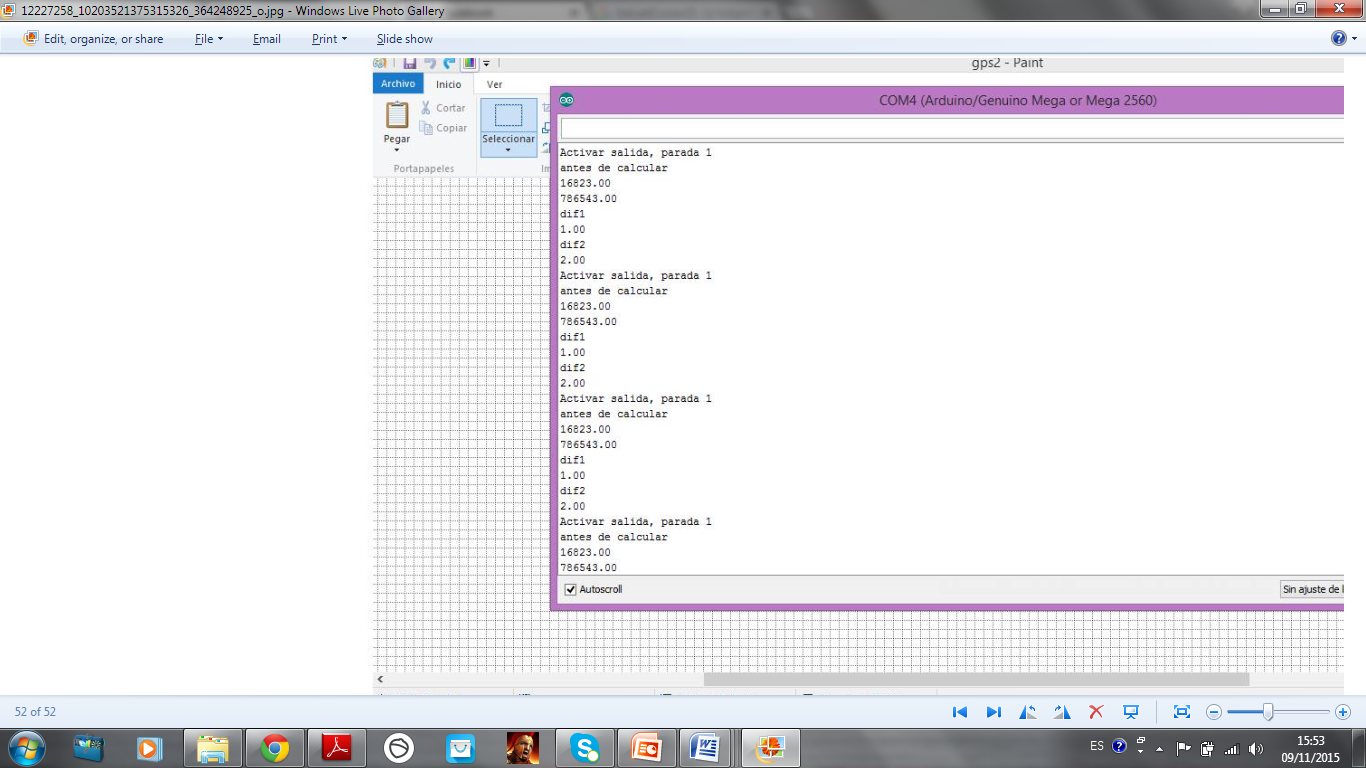
Al inicio el GPS manda las coordenadas de ubicación del automóvil, en el programa IDE da lectura a estas coordenadas pero en valores negativos, se debe multiplicarlos 10000 y por -1 para que la resta con las coordenadas seteadas en la sdcard sean más fáciles y su resultado sea positivo.

El GPS deberá estar en un lugar que no sea cerrado para que no se tarde mucho en enviar las coordenadas y no tenga cualquier clase de interferencia.

Cuando Arduino obtiene las coordenadas enviadas por el GPS procede a realizar la diferencia entre los valores del GPS con los valores predefinidos en la sdcard, si la diferencia es menor que 3 se activa el botón para abrir las puertas.

Se pone la diferencia de 3 porque el margen de error es de 5 a 9 metros.

Esta diferencia la realiza las variables dif1 y dif2 para latitud y longitud correspondientemente., como se muestra en la siguiente figura.



**Figura 7-3:** Diferencia entre las coordenadas

**Fuente:** Realizado por Luis U, Néstor E

## Tabla de Resultados

En las siguientes tablas se muestra latitud, longitud que transmite el GPS y hace la diferencia con la latitud y longitud de las coordenadas ya definidas en la Sdcard, si la diferencia es menor que tres procederá activar el seguro para que pueda en chofer abrir las puertas, también se observa la hora que llega a la parada y la hora de salida en el momento que deja la parada, no se estableció un tiempo límite por distintas razones como por ejemplo que en la parada existan bastantes pasajeros que quieran abordar el vehículo no abra un límite de tiempo para que las mismas aborden el vehículo.

Tabla 1-3: Resultados con las Primeras coordenadas predefinidas en la Sdcard

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | GPS | | Sdcard | | Diferencia | | Abrir Puertas | | Hora | |
| paradas | Latitud | longitud | Latitud | longitud | Latitud | longitud | SI | NO | Parada | Salida |
| 1 | 16824,00 | 786542,00 | 16824.00 | 786545.00 | 0 | 3 |  | x | 10:50 | 10:52 |
| 2 | 16824,00 | 786544,00 | 16824.00 | 786545.00 | 0 | 1 | x |  | 13:15 | 13:19 |
| 3 | 16822.00 | 786545.00 | 16824.00 | 786545.00 | 2 | 0 | x |  | 16:22 | 16:23 |
| 4 | 16824,00 | 786543,00 | 16824.00 | 786545.00 | 0 | 2 | x |  | 19:30 | 19:31 |

Realizado por: Uvidia Luis, Néstor Estrada

**Tabla 2-3**: Resultados con las Segundas coordenadas predefinidas en la Sdcard

Realizado por: Uvidia Luis, Néstor Estrada

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | GPS | | Sdcard | | Diferencia | | Abrir Puertas | | Hora | |
| Paradas | Latitud | longitud | Latitud | longitud | Latitud | longitud | SI | NO | Parada | Salida |
| 1 | 16857,00 | 786500,00 | 16857.00 | 786503.00 | 0 | 3 |  | x | 10:50 | 10:52 |
| 2 | 16856,00 | 786503,00 | 16857.00 | 786503.00 | 1 | 0 | x |  | 13:15 | 13:19 |
| 3 | 16855,00 | 786500,00 | 16857.00 | 786503.00 | 2 | 0 | x |  | 16:22 | 16:23 |
| 4 | 16857,00 | 786500,00 | 16857.00 | 786503.00 | 0 | 4 |  | x | 19:30 | 19:31 |

Tabla 3-3: Resultados con las Terceras coordenadas predefinidas en la Sdcard

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | GPS | | Sdcard | | Diferencia | | Abrir Puertas | | Hora | |
| paradas | Latitud | longitud | Latitud | longitud | Latitud | longitud | SI | NO | Parada | Salida |
| 1 | 16857.00 | 786461.00 | 16857.00 | 786461.00 | 0 | 0 | x |  | 10:50 | 10:52 |
| 2 | 16857.00 | 786458.00 | 16857.00 | 786461.00 | 0 | 3 |  | x | 13:15 | 13:19 |
| 3 | 16857.00 | 786456.00 | 16857.00 | 786461.00 | 0 | 5 |  | x | 16:22 | 16:23 |
| 4 | 16856.00 | 786461.00 | 16857.00 | 786461.00 | 1 | 0 | x |  | 19:30 | 19:31 |

Realizado por: Uvidia Luis, Néstor Estrada

En este prototipo se utilizó un módulo i2c que permite la incorporación del LCD de 16x2 y su fácil conexión ya que se utiliza tan solo 4 pines de la placa de Arduino.

Se implementó un botón al que se lo llama botón de pánico que estaría ubicado en serie al contacto del relé, el cual ayudará a que el chofer del vehículo que pueda abrir la puerta en un caso de emergencia, pero este botón solo se puede utilizarlo 5 veces al día, se puso ese valor porque el porcentaje para que ocurra un accidente en el vehículo es bajo.

Después de pasar 24 horas se resetea el contador y se iniciara nuevamente en cero para su uso.

La implementación de este prototipo en climas lluviosos puede causar errores ya que al momento que el GPS proceda a recibir coordenadas de los satélites puede causar interferencia.

Se utilizó un margen de error del cinco por ciento, esto es que el bus pueda abrir sus puertas cuando este a cinco metros a la redonda de las coordenadas correspondientes, ya que al momento de realizar las pruebas hubo un instante que en una parada estuvo estacionado un carro y cuando se estaciono el automóvil a un costado no se pudo abrir la puerta, así que se optó por poner ese margen de error y se logró abrir las puertas

No se consideró poner un límite de tiempo para mantener las puertas abiertas porque en la parada al momento de subir los pasajeros pueden demorarse un considerable tiempo, como también no puede haber pasajero en otra parada respectiva entonces se podría cerrar las puertas inmediatamente.

Este prototipo no es necesario que esté conectado a la batería del bus, ya que el Arduino consume poca batería lo más conveniente sería poner unas baterías de nueve voltios y una resistencia para evitar que se sobrecaliente y en peores casos que se queme la placa.

A parte de lo antes mencionado, se puede decir que se obtuvo resultados satisfactorios ya que las puertas solo se pudieron abrir en las paradas específicas en todas las condiciones normales en las que se podría demostrar.

# CONCLUSIONES

Según los análisis realizados durante el presente trabajo, finalizamos con la presentación de las siguientes conclusiones referentes a la elaboración de dicho proyecto.

* Se implementó un prototipo de control electrónico para abrir y cerrar automáticamente los seguros de las puertas de vehículo únicamente en las paradas programadas, mediante el uso de la plataforma Arduino, dispositivo de georreferencia (GPS), dispositivos eléctricos, entre otros.
* Se investigó que tipo de tarjeta Arduino seria capas de programar datos necesarios para enviar y recibir información en el momento de la llegada del vehículo a su respectiva parada, entonces el haber escogido la tarjeta de Arduino Mega 2560 nos permitió un excelente manejo y transportación de datos, ya que dicho transporte de datos es confiable entre dispositivos remotos, las características que esta tarjeta nos ofrece son muy importantes; bajo costo para implementaciones en gran escala, bajo consumo de energía, mas memoria para programar, entre otras.
* Se utilizó un módulo GPS U-BLOX NEO-6M ya que es adaptable para la tarjeta Arduino para enviar y recibir información referente a la posición del vehículo en el cual se integraron elementos tales como; protoboard, display LCD, tarjeta de almacenamiento SDcard, siendo primordiales en la construcción del circuito para su correcto funcionamiento y posterior visualización al usuario, es de bajo consumo de energía y d alta precisión, además la trama de datos q entrega este GPS es corta y fácilmente manipulable
* Se implementó un circuito actuador que se encarga del sistema de apertura y cerradura del seguro de las puertas del vehículo mediante el uso de un relé, el cual permitirá que se abran las puertas solo en las paradas específicas.
* Se utilizó el programa Arduino IDE que sirvió para la configuración de la tarjeta Arduino y del módulo GPS, que son la base para el funcionamiento global de nuestro proyecto.

# RECOMENDACIONES

Finalmente las recomendaciones que vamos a citar servirán para ampliar el campo aplicativo de nuestro proyecto, así como errores que deben evitarse:

* El proyecto que se desarrolló permitirá múltiples aplicaciones afines, como por ejemplo la posibilidad de llevar un control de tiempos de recorridos y tiempos de llegada a cada parada, así como la visualización de la ruta que lleva el vehículo mediante la cooperación de otros programas de rutas visuales.
* Al momento de configurar la tarjeta Arduino dentro de nuestro sistema, no se debe olvidar que uno siempre tiene que ser el coordinador, ya que se debe realizar una programación de las paradas deseadas con anticipación, esto se lo realiza mediante el Arduino IDE, teniendo en cuenta que una implementación posterior puede ser a niveles más grandes como a nivel regional e incluso nacional, se puede crear posteriormente una aplicaciones para Android, IOS, Mac o Windows para facilitar el acceso al control de nuestro sistema en red a través de internet.
* Como se está trabajando con el modulo GPS, se debe tomar en cuenta que puede ser susceptible a interferencias o a demoras en la recepción de la señal satelital, para ello se recomienda ubicar el dispositivo en un lugar visible del vehículo, no en lugares cerrados o escondidos.
* Se recomienda utilizar un Arduino con mayor memoria para poder realizar más líneas de programación.
* Se recomienda utilizar este prototipo en todas las líneas de bus para poder reducir aunque sea en un poco porcentaje el tráfico ocasionado por los mismos.

# BIBLIOGRAFÍA

**Arduino.c***. Arduino - Windows* [en línea].mayo 2015. [Consulta 15 octubre 2015]. Disponible en:

https://www.arduino.cc/en/Guide/Windows

**Arduinoecia**, *modulo-i2c-display-16x2-arduino* [en línea], diciembre 2014. [Consulta 15 de Octubre 2015]. Disponible en:

http://www.arduinoecia.com.br/2014/12/modulo-i2c-display-16x2-arduino.html

**BANZI, Massimo**. *Primeiros passos com o Arduino* [en línea]*.* *São Paulo: Novatec*, 2011, [Consulta 14 octubre 2015]. Disponible en :

http://novatec.com.br/livros/primeiros-passos-arduino/capitulo9788575222904.pdf

**Bayle, Julien. C**. Programming for Arduino. Olton, Birmingham, GBR: Packt Publishing Ltd, 2013. [Consulta: 9 Noviembre 2015]. Disponible en:

<http://site.ebrary.com/lib/espoch/reader.action?docID=10714267&ppg=1>

**Bevly, David M., and Cobb, Stewart**. GNSS for Vehicle Control. Norwood, MA, USA: Artech House, 2010. [Consulta: 9 Noviembre 2015]. Disponible en:

<http://site.ebrary.com/lib/espoch/reader.action?docID=10421845&ppg=5>

**Blum, Jeremy.** Exploring Arduino. Tools and Techniques for Engineering Wizardry. Somerset, NJ, USA: John Wiley & Sons, 2013. [Consulta: 9 Noviembre 2015]. Disponible en:

<http://site.ebrary.com/lib/espoch/reader.action?docID=10735206&ppg=1>

**Diggelen, Frank van. A-GPS**: Assisted GPS, GNSS, and SBAS. Norwood, MA, USA: Artech House, 2009. pp. 9-12. [Consulta: 9 Noviembre 2015]. Disponible en:

<http://site.ebrary.com/lib/espoch/reader.action?docID=10312937&ppg=1>

**Griffey, Jason. Gadgets and Gizmos**. Personal Electronics and the Library: A Library Technology Report. Chicago, IL, USA: ALA Editions, 2010. [Consulta: 9 Noviembre 2015].

Disponible en:

<http://site.ebrary.com/lib/espoch/reader.action?docID=10486809&ppg=1>

**HERRADOR, Rafael Enríquez**. *Guía de Usuario de Arduino* [en línea]. Universidad de Córdoba, 2009, [Consulta 6 noviembre 2015]. Disponible en :

http://www.uco.es/aulasoftwarelibre/wp-content/uploads/2010/05/Arduino\_user\_manual\_es.pdf

**HOME**, *Arduino. Portal oficial del hardware libre Arduino* [en línea]. *mayo 2015,* [consulta 15 octubre 2015]. Disponible en:

https://www.arduino.cc/en/Guide/Windows

**Lo, Burt**. GPS and Geocaching in Education. Eugene, OR, USA: ISTE, 2010. pp. 9-17. [Consulta: 9 Noviembre 2015]. Disponible en:

<http://site.ebrary.com/lib/espoch/reader.action?docID=10759763&ppg=7>

**Milette, Greg, and Stroud, Adam.** Professional Android Sensor Programming. Somerset, NJ, USA: John Wiley & Sons, 2012. pp. 7-9. [Consulta: 9 Noviembre 2015]. Disponible en:

<http://site.ebrary.com/lib/espoch/reader.action?docID=10565136&ppg=1>

**National Academy of Engineering**. Global Navigation Satellite Systems: Report of a Joint Workshop of the National Academy of Engineering and the Chinese Academy of Engineering. Washington, DC, USA: National Academies Press, 2012. [Consulta: 9 Noviembre 2015].

Disponible en:

<http://site.ebrary.com/lib/espoch/reader.action?docID=10557784&ppg=13>

**Nussey, John.** Arduino for Dummies. Somerset, NJ, USA: John Wiley & Sons, 2013. [Consulta: 9 Noviembre 2015]. Disponible en:

<http://site.ebrary.com/lib/espoch/reader.action?docID=10695401&ppg=1>

**Parkinson, Bradford W, and Spilker, James J, eds**. Global Positioning System: Theory and Applications, Volume I. Reston, VA, USA: American Institute of Aeronautics and Astronautics, 1996. [Consulta: 9 Noviembre 2015]. Disponible en:

<http://site.ebrary.com/lib/espoch/reader.action?docID=10516768&ppg=34>

**PATAGONIATECNOLOGY.** *arduino mega 2560* [en línea]. Leandro [consulta 29 de octubre 2015] Disponible en:

http://saber.patagoniatecnology.com/arduino-mega-2560-atmega-mega-arduino-clon-compatible-argentina-tutorial-basico-informacion-arduino-argentina-ptec/

**Prasad, Ramjee, and Ruggieri**. Marina Applied Satellite Navigation Using GPS, and Augmentation Systems. Norwood, MA, USA: Artech House, 2005. [Consulta: 9 Noviembre 2015].

Disponible en:

<http://site.ebrary.com/lib/espoch/reader.action?docID=10081977&ppg=15>

**Rao, B. Rama, Kunysz, W, and Fante, R**. GPS/GNSS Antennas. Norwood, MA, USA: Artech House, 2012. [Consulta: 9 Noviembre 2015]. Disponible en:

<http://site.ebrary.com/lib/espoch/reader.action?docID=10818326&ppg=1>

**TIENDA ARDUINO.** *datasheet arduino mega* [en línea]. [consulta 15 octubre 2015]

www.tiendaarduino.com/datasheet/arduino-mega\_2560.htm#.VjyhbrerTIU

**Varnum, Kenneth J. Top Technologies Every Librarian Needs to Know:** A LITA Guide. Chicago, IL, USA: American Library Association, 2014. [Consulta: 9 Noviembre 2015].

Disponible en:

<http://site.ebrary.com/lib/espoch/reader.action?docID=10921672&ppg=54>

**Villasenor, Ignacio.** **“**Reforma”. Cómo Funciona? Más Automatica que Manual, 2001, Mexico, volume 20, pp. 345-354. [Consulta: 9 Noviembre 2015]. Disponible en:

http://search.proquest.com/docview/310685946/abstract/EE7CF556866A4348PQ/9?accountid=36724

**Yang, Dehuai, ed**. Applied Mechanics and Materials, Volume 431: Mechanical Engineering, Industrial Materials and Industrial Electronics. Durnten, Zurich, CHE: Trans Tech Publications, 2013. [Consulta: 9 Noviembre 2015]. Disponible en:

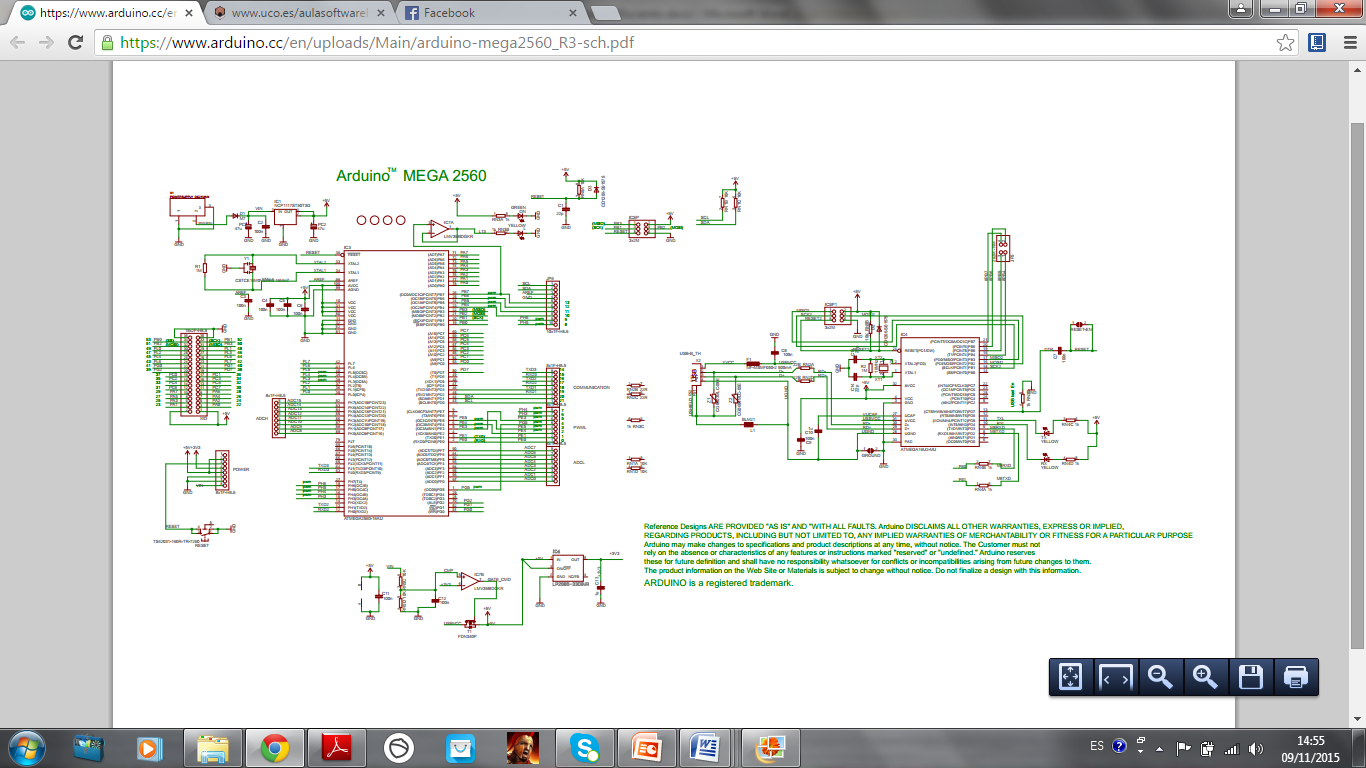
<http://site.ebrary.com/lib/espoch/reader.action?docID=10803706&ppg=68>

**Mar Domínguez**, *solo arduino*  [blog]. [consulta 5 Noviembre 2015]

Disponible en: http://soloarduino.blogspot.com/

# ANEXOS

**Anexo A:** Datasheet de Arduino Mega 2560



**Anexo B :** Programación de Arduino

En esta sección se declaran o se hace el llamado a las bibliotecas del Sistema que nos serán útiles para el manejo de los dispositivos que componen el prototipo.

#include <TinyGPS++.h>

Biblioteca TinyGPS, convierte de formato NEMA datos de posicionamiento global en variables fáciles de usar para Latitud, Longitud, Tiempo y otra información.

#include <SoftwareSerial.h>

Biblioteca SoftwareSerial.h, crea un puerto Serie que trabaje con el GPS. (Rx, Tx)

#include <LiquidCrystal\_I2C.h> //Para usar un display I2C

Biblioteca LiquidCrystal\_I2C.h, nos proporciona las instrucciones para manejar el I2C LCD Display.

#include <Wire.h> //La necesita la libreria LiquidCrystal\_I2C.h

Biblioteca Wire.h, complementa a la biblioteca LiquidCrystal\_I2C.h para manejar el I2C LCD Display.

#include <SPI.h>

#include <SD.h>

Biblioteca SPI.h y SD.h, proporciona el grupo de instrucciones para leer y escribir datos desde una SD card.

Una vez definidas las bibliotecas como en la programación básica se definen las variables y constantes q serán empleadas en el programa.

static const int RXPin = 63, TXPin = 62;

Definición de pines para comunicación del GPS

static const uint32\_t GPSBaud = 9600;

Definición de la velocidad de comunicación GPS

LiquidCrystal\_I2C lcd(0x27,2,1,0,4,5,6,7,3, POSITIVE);

Definición de pines para display

TinyGPSPlus gps;

SoftwareSerial ss(RXPin, TXPin);

Habilitamos los pines de RX y TX como puerto serial para poder comunicar el GPS con la tarjeta de Arduino.

const int chipSelect = 53;

Declaración del pin para control de la tarjeta.

File Archivo;

Asignacion del nombre al handler del archivo de la tarjeta.

double lat1,lat2,lat3,lon1,lon2,lon3;

float lagps,logps,dif1,dif2;

char trama[100];

byte aux;

String cadena="";

int i,j,cp,salida;

boolean bandera;

Definición variables para ejecución del programa.

void setup()

{

Serial.begin(115200);

ss.begin(GPSBaud);

pinMode(salida,OUTPUT); //declarar pin 48 como vara=iable salida para el resto del programa

lat1=0; lat2=0; lat3=0;lon1=0,lon2=0,lon3=0;

i=0; j=1; cp=0;

lcd.begin(16,2); // Inicializamos el LCD

lcd.backlight();

if (!SD.begin(chipSelect)) {

lcd.print("SD no iniciada");

delay(500);

// don't do anything more:

return;

}

lcd.print("SD iniciada");

delay(500);

//leer datos de la tarjeta previamente guardados

Archivo = SD.open("DATOS.txt");

if (Archivo){ ooioioiii

while (Archivo.available()){

//mientras exista informacion en el txt

i=0; //puntero para leer las lineas del txt

//j=1; //puntero para saber que linea del txt estoy leyendo

bandera=true;

while(bandera==true)

{

aux=Archivo.read();

trama[i]=aux;

cadena.concat(trama[i]);

i++;

if (aux=='\n'){

bandera=false;

switch (j){

case 1:

lat1=cadena.toFloat();

Serial.println(lat1);

j++;

break;

case 2:

lon1=cadena.toFloat();

Serial.println(lon1);

j++;

break;

case 3:

lat2=cadena.toFloat();

Serial.println(lat2);

j++;

break;

case 4:

lon2=cadena.toFloat();

Serial.println(lon2);

j++;

break;

case 5:

lat3=cadena.toFloat();

Serial.println(lat3);

j++;

break;

case 6:

lon3=cadena.toFloat();

Serial.println(lon3);

j++;

break;

}

cadena="";

}

else{

bandera=true;

};

};

}

};

Archivo.close(); //cerrar el archivo de txt

}

void loop()

{

// esta porcion muestra informacion del gps cuando es obtenida correctamente

while (ss.available() > 0)

if (gps.encode(ss.read()))

displayInfo();

if (millis() > 5000 && gps.charsProcessed() < 10) //si pasa del tiempo muerto, muestra el mensaje no hay GPS

{

lcd.clear();

lcd.print("No GPS");

while(true);

}

}

void displayInfo()

{

if (gps.location.isValid())

{

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Latitud->"); // imprimirmos la cadena

lcd.print(gps.location.lat(),6); // imprimimos la latitud

lcd.setCursor(0, 1);

lcd.print("Longitud->");

lcd.print(gps.location.lng(),6); // imprimimos la longitud

//preguntar por la direccion del gps

lagps=gps.location.lat()\*10000\*(-1); //para redondear la locacion de latitud

lagps=round(lagps); //redondear latitud

logps=gps.location.lng()\*10000\*(-1); //para redondear la locacion de longitud

logps=round(logps);

Serial.println("antes de calcular");

Serial.println(lagps);

Serial.println(logps);

dif1=lat1-lagps; //encontrar la diferencia de latitud para parada 1

dif2=lon1-logps; //encontrar la diferencia de longitud para parada 1

Serial.println("dif1");

Serial.println(dif1);

Serial.println("dif2");

Serial.println(dif2);

if((dif1<3)&(dif2<3)&(dif1>(-3))&(dif2>(-3))){

Serial.println("Activar salida, parada 1");

digitalWrite(salida,HIGH); //activo la salida para el rele

delay(1000); //riempo que permanecera encendida

digitalWrite(salida,LOW);

}

else{

dif1=lat2-lagps; //diferencia latitud parada 2

dif2=lon2-logps; //diferencia longitud parada 2

if((dif1<3)&(dif2<3)&(dif1>(-3))&(dif2>(-3))){

Serial.println("Activar salida, parada 2");

digitalWrite(salida,HIGH); //activo la salida para el rele

delay(1000); //riempo que permanecera encendida

digitalWrite(salida,LOW);

}

else{

dif1=lat3-lagps; //diferencia latitud parada 3

dif2=lon3-logps; //diferencia longitud parada 3

if ((dif1<3)&(dif2<3)&(dif1>(-3))&(dif2>(-3))){

Serial.println("Activar salida, parada 3");

digitalWrite(salida,HIGH); //activo la salida para el rele

delay(1000); //riempo que permanecera encendida

digitalWrite(salida,LOW);

Serial.println("recorrido completo");

Serial.println("numero de recorridos");

cp++;

Serial.println(cp);

delay(200);

}

}

}

delay(200);

}

else

{

lcd.clear();

lcd.print("Locacion Invalido");

delay(1000);

}

if (gps.date.isValid())

{

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Fecha:");

lcd.setCursor(0, 1);

lcd.print("M:"); // imprimirmos la cadena

lcd.print(gps.date.month()); // imprimimos la variable

lcd.setCursor(5, 1);

lcd.print("D:");

lcd.print(gps.date.day());

lcd.setCursor(10, 1);

lcd.print("A:");

lcd.print(gps.date.year());

delay(1000);

}

else

{

lcd.clear();

lcd.print("Fecha Invalido");

delay(2000);

}

}