



INSTITUTO POLITÉCNICO NACIONAL

**ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA Y
ELÉCTRICA**

UNIDAD CULHUACAN

SISTEMA DE LOCALIZACIÓN Y BLOQUEO DE MAQUINARIA AGRÍCOLA VÍA GSM/GPS

TESIS

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMUNICACIONES Y ELECTRONICA

P R E S E N T A

MARIO ALBERTO BELLACETIN VILLEGAS

A S E S O R E S

ING. CARLOS AQUINO RUIZ

ING. CELEDONIO ENRIQUE AGUILAR MEZA



**MEXICO, D.F.
2013**

**INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA Y ELÉCTRICA
UNIDAD CULHUACAN**

TESIS INDIVIDUAL

Que como prueba escrita de su Examen Profesional para obtener el Título de Ingeniero en Comunicaciones y Electrónica, deberán desarrollar el C:

MARIO ALBERTO BELLACETIN VILLEGAS

“SISTEMA DE LOCALIZACION Y BLOQUEO DE MAQUINARIA AGRICOLA VIA GSM/GPS”

Con este sistema no se podrá evitar el robo de la maquinaria agrícola, pero si podemos evitar una pérdida total de esta misma ya que con el sistema de localización podremos encontrar la maquinaria que fue robada vía internet en base a un mapa la cual se basa en tecnología GPS y en base a la tecnología GSM se mandará un mensaje con el celular para detener la maquinaria agrícola, esto se conseguirá que al momento de recibir el mensaje el módulo GSM/GPS mandará un pulso el cual corte el paso de combustible al motor de la máquina y así éste pierda movimiento y sea más fácil su localización.

También en ese mismo instante el módulo GSM/GPS nos devolverá con un mensaje la ubicación de nuestra maquinaria robada y de esta manera podremos evitar la pérdida total de este bien, el cual además de valer demasiado dinero tiene alto valor útil tanto para una persona como para la sociedad ya que la función que desarrollan estas máquinas son muy importantes.

CAPITULADO

**CAPITULO 1 ESTADO DEL ARTE
CAPITULO 2 MARCO TEORICO
CAPITULO 3 DISEÑO E IMPLEMENTACION
CAPITULO 4 PRUEBAS Y RESULTADOS**

México D. F., a 21 de noviembre del 2013

PRIMER ASESOR:

SEGUNDO ASESOR:

ING. CARLOS AQUINO RUIZ

ING. CELEDONIO ENRIQUE AGUILAR MEZA

Vo. Bo.

APROBADO

M. en C. ANTONIO ROMERO ROJANO
JEFE DE LA CARRERA DE I.C.E.

M. en C. HECTOR BECERRIL MENDOZA
SUBDIRECTOR ACADÉMICO

Índice

Planteamiento del problema	5
Justificación	9
Objetivos	10
Capítulo 1. Estado del arte	11
Capítulo 2. Marco teórico	17
2.1 Introducción	18
2.2 Tecnología GPS	18
2.2.1 Arquitectura del sistema GPS	19
2.2.2 Principios de funcionamiento de los GPS	19
2.2.3 Cómo ubica la posición el receptor GPS	19
2.2.4 Fuente de errores de los GPS	21
2.3 Tecnología GSM	21
2.3.1 Arquitectura de la red GSM	22
2.4 Arduino	24
2.5 Arduino Uno	26
2.5.1 Resumen	28
2.5.2 Alimentación	30
2.5.3 Memoria Atmega328	30
2.5.4 Entrada y salida	31
2.5.5 Comunicación	32
2.5.6 Programación	33
2.5.7 Características físicas	34
2.6 Microprocesadores	34
2.6.1 Tipos de microprocesadores	35
2.7 Tarjeta SIM	35
2.8 Arduino Shield Cellular SM5100 B	37
2.9 Arduino Shield GPS	39
2.10 Bomba de combustible eléctrica	41
2.11 Estado financiero	44
Capítulo 3 Diseño e implementación	45
3.1 Esquema de funcionamiento de SLAyB de maquinaria Agric.	46
3.2 Fases de función	47
3.3 Diagrama a bloques de SLAyB de maquinaria Agric.	48
3.4 Diagrama de flujo del proceso del proyecto	50
3.5 Control de las salidas digitales del arduino mediante un mensaje de texto SMS	51
3.5.1 Conexión de hardware	52
3.5.2 Diseño de software	53
3.6 Circuito de activación y desactivación de una bomba de combustible	58
3.7 Ubicación y localización de coordenadas geográficas mediante el shield GPS	60

3.7.1 Instalación del shield GPS con el Arduino Uno	60
3.7 .2 Programación para el modulo GPS	61
3.8 Envío de un mensaje mediante el shield GSM con las coordenadas geográficas	65
3.8 .1 Envío de un mensaje de texto SMS con el Shield GSM	66
3.9 circuito de conmutación para la alimentación del sistema de localización y bloqueo de maquinaria agrícola	68
Capítulo 4 Pruebas y resultados	70
Resumen	71
4.1 Envío de un mensaje SMS con las coordenadas geográficas en base a un evento	71
4.2 Funcionamiento y problemas del shield GPS	76
4.3 Desarrollo y pruebas del programa general	79
Conclusiones	85
Apéndice	87
Referencias	102
Referencia de imágenes	105

Índice de figuras

Figura 2.1 Esquema de funcionamiento de un sistema GSM	23
Figura 2.2 Vista frontal del Arduino Uno	26
Figura 2.3 Vista trasera del Arduino Uno	27
Figura 2.4 Diagrama eléctrico del Arduino Uno	29
Figura 2.5 Configuración de pines del ATMEGA 328	31
Figura 2.6 Diagrama a bloques de un CPU	35
Figura 2.7 Tarjeta SIM versión pequeña: 25 x 15 x 0,76 mm	36
Figura 2.8 Shield Cellular SM5100 B	38
Figura 2.9 Diagrama eléctrico de Arduino Shield Cellular SM5100 B	38
Figura 2.10. a) Arduino Shield GPS montado a Arduino UNO. B) Parte trasera de Arduino Shield GPS	39
Figura 2.11. Mediciones de la parte trasera con medidas del Arduino shield GPS	40
Figura 2.12. Diagrama esquemático del Arduino Shield GPS	40
Figura 2.13. Esquema de una bomba de combustible	41
Figura 2.14. Esquema de una bomba de combustible eléctrica	42
Figura 3.1 Esquema de funcionamiento del sistema de localización alarma y bloqueo de maquinaria agrícola.	43
Figura 3.2 Descripción del proceso a realizar mediante un diagrama de flujo.	50
Figura 3.3 Conexión del shield GSM con el Arduino	52
Figura 3.4 Antena para celular de cuatro bandas y conexión estándar SMA	52
Figura 3.5 Adaptación de una antena de mayor potencia en el shield GSM	53
Figura 3.6 Diagrama de flujo del control de salidas digitales en base a un mensaje SMS	56
Figura 3.7 Circuito para la comprobación del control de salidas del Arduino mediante un mensaje SMS	57
Figura 3.8 diagrama eléctrico de circuito activador desactivador	58
Figura 3.9 diagrama eléctrico de un optoacoplador indicando entradas y salidas	59
Figura 3.10 shield GPS montado en el Arduino con el modulo GPS EM406A	61

Figura 3.11 Diagrama de flujo para la obtención de coordenadas geográficas	62
Figura 3.12 pantalla serial que muestra las coordenadas geográficas	64
Figura 3.13 ubicación donde fue probado el modulo GPS	65
Figura 3.14 diagrama de flujo del envío de mensaje mediante Arduino	67
Figura 3.15 Circuito de conmutación	68
Figura 4.1. Diagrama de flujo del funcionamiento de la primera prueba.	71
Figura 4.2. Diseño de circuito de activación en base a un evento	74
Figura 4.3. Muestras del mensaje enviado por el celular	75
Figura 4.4. Papa que muestra la localización del lugar en donde está el dispositivo situado	75
Figura 4.5. Localización del sketch para la prueba del GPS	76
Figura 4.6. Respuesta enviada por la pantalla serial.	77
Figura 4.7. Muestra de mensaje recibido por el shield GSM en un celular	79
Figura 4.8. Diagrama de flujo general de funcionamiento del programa final	80
Figura 4.9. Circuito activador desactivador y pistas del mismo circuito	82
Figura 4.10. Conector Alimentación Jack 5.5x2.1mm Arduino	83
Figura 4.11. Circuito de conmutación con el regulador 7805	84

Índice de tablas

Tabla 1.- índice de robos de autos del 2004-2011 en la república mexicana	6
---	---

Índice de Gráficas

Gráfica 1.- Indicador del índice de seguridad de la zona Tenango del aire y Amecameca	7
Gráfica 2.- Indicador de victimas que han sufrido robo de su maquinaria agrícola	8
Gráfica 3.- Personas que han recuperado su maquinaria agrícola después de un robo	9

Planteamiento del problema.

La seguridad ciudadana se ha convertido en un asunto prominente en la agenda de la consolidación democrática y el desarrollo de México. Impone elevados costes económicos y se introduce en la vida política. Se examinan los incentivos a la violencia que supone la impunidad generalizada que es consecuencia de la debilidad de las fuerzas de orden público, ineficiencia política, gran crecimiento de pobreza y el sistema judicial. La seguridad, como bien público, es responsabilidad primaria del Estado, pero también compete a las autoridades locales y la sociedad civil.

Una problemática a resaltar en el tema de la inseguridad en México es el tema del robo de autos, presentándose casos más a menudo. Como no los muestra la Asociación Mexicana de Instituciones de Seguros, son alarmantes las cifras que presentan en torno al robo de vehículos en todo el país.

Según los datos de la AMIS (Asociación Mexicana de Instituciones de Seguros), el número de robos de autos asegurados a nivel nacional en el periodo Enero-Abril 2011 ha alcanzado la cifra de 26,604 unidades, lo que representa un incremento de casi el doble con respecto a la misma cifra del 2004 cuando ésta alcanzó las 15,547 unidades.

Dentro de dicha cifra las entidades que presentan un mayor índice son el Estado de México, Nuevo León, Jalisco, Sinaloa, Veracruz, Tamaulipas, Coahuila y Durango como se observa en la tabla 1. Mientras que el Distrito Federal, Chihuahua, Baja California y Puebla muestran un ligero decremento.

Tabla 1.- Índice de robos de autos del 2004-2011 en la republica mexicana

Entidad	2004	2005	2006	2007	2008	2009	2010	2011
DF	6,074	5,293	4,551	4,536	5,247	5,321	4,833	3,690
Edo. México	4,031	3,525	3,229	3,995	3,983	4,748	5,302	5,499
Nuevo León	362	370	523	1,156	1,580	2,058	2,676	4,607
Jalisco	1,336	1,188	1,001	1,062	1,367	1,513	1,853	2,017
Chihuahua	299	411	574	668	883	1,644	1,985	1,733
Baja California	847	885	908	885	937	725	592	483
Sinaloa	325	429	348	391	498	1,057	1,325	1,394
Puebla	297	344	347	325	448	493	484	449
Veracruz	136	216	232	247	465	471	500	818
Tamaulipas	154	193	253	278	368	348	616	944
Coahuila	48	64	92	101	148	312	456	655
Durango	69	49	32	49	68	146	283	340
TOTAL	15,547	14,753	13,796	16,044	18,616	21,752	24,206	26,604

Fuente: Asociación Mexicana de Instituciones de Seguros (AMIS).

Debido a este problema la gente opta por la opción de asegurar su carro, lo cual permite que al momento de sufrir un incidente como el robo de un automóvil esta persona no se preocupe por una pérdida total de su automóvil. Este fenómeno del robo de autos ha hecho que las empresas aseguradoras de autos crezcan y que la mayoría de autos ya estén asegurados, en especial los que salen de agencia, ya que si en caso de que no esté asegurado este no podrá circular.

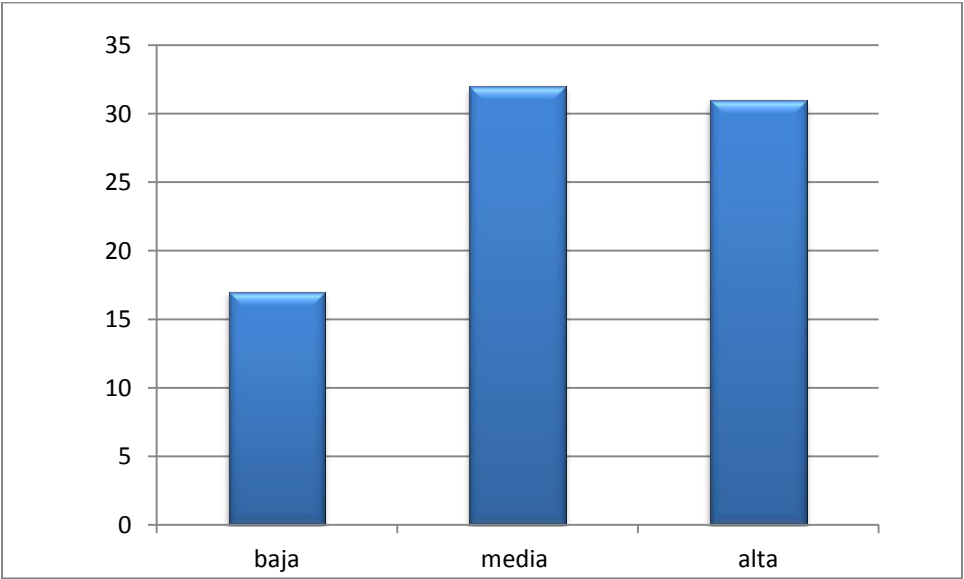
Un sector olvidado por las aseguradoras es el sector de la agricultura con toda su maquinaria agrícola que existe en este sector, como son tractores, trilladoras, sembradoras y aspersores. Son máquinas que rondan alrededor de los \$300,000 como valor mínimo hasta los \$3,000, 000 como valor máximo.

Este sector agrícola también es víctima de la inseguridad, ya que también se encuentra el caso constante de robo de maquinaria agrícola en este sector antes mencionado, problema que es de llamar la atención ya que como lo mencionamos es un sector olvidado por las aseguradoras y al momento de ser robados pues se da el caso que tenemos una pérdida total, y ésta es mucho mayor que una pérdida de automóvil pues la diferencia de precios que existe es de

considerarse, y en el proceso de compra y venta de la maquinaria agrícola no existe de por medio ninguna aseguradora, y los sistemas de seguridad en caso de robo o extravío son mínimos los casos.

A continuación se presentan algunas gráficas, que se diseñaron por medio de una encuesta realizada a las personas del sector campo y que tienen contacto a diario con el sector y maquinaria agrícola, la cual nos muestra los niveles de inseguridad que existen en este sector y nos muestran un índice de robo de maquinaria agrícola y datos sobresalientes sobre los niveles de seguridad en la zona de Tenango del aire y Amecameca:

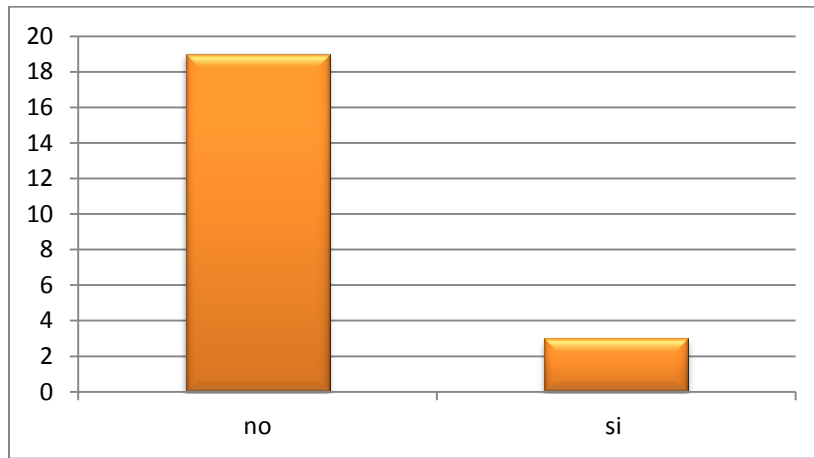
Gráfica 1.- indicador del índice de seguridad en la zona de Tenango del aire y Amecameca.



Fuente: Encuesta hecha a 100 personas en la zona de Tenango del aire y Amecameca.

Tabla que nos indica la opinión de 100 personas encuestadas en la zona de Tenango del aire sobre su opinión de que tan alto es el índice de seguridad en su región.

Gráfica 2.- indicador de víctimas que han sufrido robo de su maquinaria agrícola.

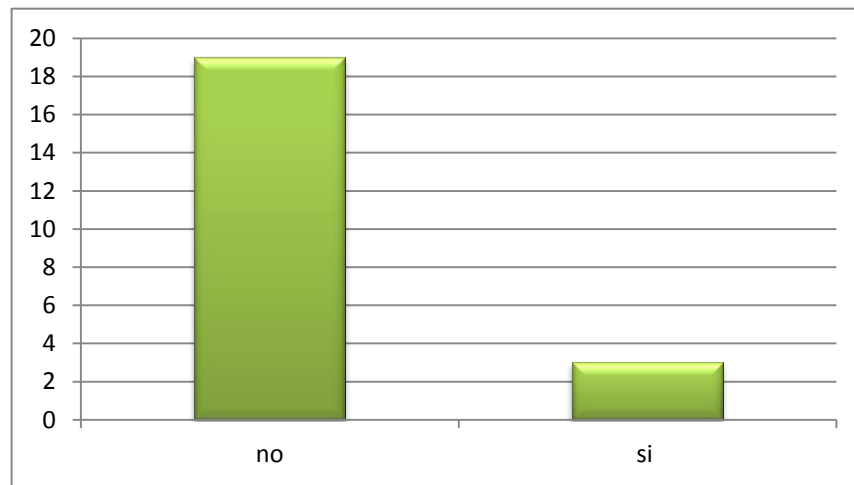


Fuente: Encuesta hecha a 100 personas en la zona de Tenango del aire y Amecameca.

Como lo indica la tabla 3 al menos el 22% de las personas que radican en este lugar han sufrido de robo de maquinaria agrícola, mientras el otro 78% no ha sufrido de este incidente, pero de este 78% al menos el 90% ha sabido de casos de robo de maquinaria agrícola a sus alrededores.

Del 22% que si ha sufrido el robo de maquinaria agrícola al menos el 68% ha sido a mano armada, mientras que el otro 32% fue desaparición o robo de mientras la maquinaria estaba sola.

Gráfica 3.- Personas que han recuperado su maquinaria agrícola después de un robo.



Fuente: Encuesta Hecha a 100 personas en la zona de Tenango del aire y Amecameca.

Como es mostrado en la tabla la mayoría de las personas no recuperan sus pertenencias por falta de una aseguradora, o un dispositivo de seguridad que les permita saber la localización de su maquinaria agrícola. Muchas de las personas encuestadas adquirirían un dispositivo el cual les permita localizar su maquinaria agrícola cuando esta sea robada o desaparecida.

Justificación:

Debido al alto índice que existe en el robo de maquinaria agrícola, y que además es un problema que no se toma con la seriedad que se debe, ya que el sector agrícola es un sector olvidado por el gobierno y a la vez por las aseguradoras para la protección ante robo. Se diseñará un sistema de localización y bloqueo de maquinaria agrícola vía GSM/GPS.

Con este sistema no se podrá evitar el robo de la maquinaria agrícola, pero si podemos evitar una pérdida total de esta misma, ya que con el sistema de localización podremos encontrar la maquinaria que fue robada vía internet en base a un mapa la cual se basa en tecnología GPS y en base a la tecnología GSM se

mandará un mensaje con el celular para detener la maquinaria agrícola, esto se conseguirá que al momento de recibir el mensaje el módulo GSM/GPS mandará un pulso el cual corte el paso de combustible al motor de la máquina y así éste pierda movimiento y sea mas fácil su localización, también en ese mismo instante el módulo GSM/GPS nos devolverá con un mensaje la ubicación de nuestra maquinaria robada y de esta manera podremos evitar la pérdida total de este bien, el cual además de valer demasiado dinero tiene un alto valor útil tanto para una persona como para la sociedad ya que la función que desarrollan estas máquinas son muy importantes.

Con esto promoveremos mayor seguridad a las personas que tengan u obtengan alguna maquinaria agrícola, y que las personas tengan mayor seguridad en el momento que llegue a suceder algún robo o desaparición de su maquinaria agrícola, tenga la confianza de que recuperara su bien de una manera a la parcial, pero esto no afectaría tanto a las personas a comparación de una pérdida total la cual representa un gasto y una pérdida muy grande al dueño de la maquinaria agrícola.

Objetivo general:

Diseñar e implementar un sistema de localización, bloqueo y alarma para maquinaria agrícola mediante un módulo GSM/GPS.

Objetivos particulares:

- Analizar los módulos de tecnología GSM/GPS para obtener las señales de control del sistema.
- Diseñar un sistema de bloqueo mediante el corte de combustible, para evitar que la maquinaria agrícola se mueva y así prevenir una pérdida total en un robo.
- Implementar un mensaje de alerta al celular del dueño de la maquinaria agrícola.

Capítulo 1

Estado del arte

Capítulo 1. Estado del arte

En este apartado se mencionarán algunos proyectos relacionados con el proyecto de sistema de localización y bloqueo de maquinaria agrícola vía GSM/GPS, mostrando algunos avances tecnológicos los cuales sirven como base para así poder aplicar esos avances tecnológicos en el proyecto antes mencionado, tomando en cuenta cuales son las ventajas y desventajas que proporcionan o documentan estos proyectos para así poder retomar lo mejor de las tecnologías utilizadas.

En la implementación del módulos GSM/GPS, usando un teléfono móvil viejo se ha implementado un sistema de alarma que comprende de un circuito muy sencillo mediante un PIC 12F629, el sistema es de pequeñas dimensiones además de ser muy económico en el sentido de alimentación pues funciona con una tensión entre 3V hasta 18V debido a un estabilizador de tensión (diodo zener) lo cual lo permite ser compatible con una gran variedad de celulares en desuso.

La ventaja sobre otros sistemas es que envían un SMS de que no genera ningún tipo de consumo, ya que cuando nosotros recibimos la llamada y ver que el número pertenece a nuestro sistema de alarma, no tenemos la necesidad de descolgar. Lo cual nos da la posibilidad de usar tarjetas de prepago, esto ya dependiendo de la compañía celular.

Este sistema cuenta con tres versiones:

- La versión 1.1 provoca la activación de la alarma al abrir la puerta, repitiendo la llamada cada 5 minutos en el caso de que se mantenga la puerta abierta.
- La versión 1.2 está especialmente indicada para controlar la apertura y cierre de tiendas y comercios, ya que solamente llamara cuando se abra la puerta y cuando se vuelva a cerrar.
- La versión 1.3 se puede configurar a las necesidades del usuario.

La utilización de este circuito o sistema es alimentada por la misma batería del celular al cual es instalado, lo cual lo hace de una cierta forma un sistema cómodo al no necesitar una fuente de energía externa.

Entre sus desventajas es que este sistema no es compatible para todas las marcas o modelos de celular, un ejemplo es con los celulares NOKIA la cual necesita una adaptación de otro circuito a base de relevadores; lo que significa otro gasto más.

También se encontró con una alarma creada por estudiantes de la comunidad estudiantil del tecnológico de estudios superiores de Ecatepec, la cual tiene como función disminuir o anular el robo de autos (Tsuru), ya que cuando se cometa este acto el sistema tenga la función de activar una sirena y anular completamente la corriente del auto.

Para la configuración de este sistema de alarma utilizaron dos circuitos integrados 555, los cuales están conectados en configuración monoestable; el primer 555 provee el tiempo necesario de retardo para poder salir del auto, y se activa por el conductor antes de salir del mismo presionando un interruptor de salida. Pasando el tiempo de retardo la salida del primer circuito integrado 555 pasa a un nivel bajo y la alarma queda activada para detectar intrusos.

Después de este proceso la alarma como se dijo queda activada ya que mantiene un capacitor cargado con 12V, en el siguiente paso si el dueño del auto regresa al automóvil tendrá que desconectar la alarma, esto lo hace cuando al abrir la puerta se activa un temporizador que dará tiempo para desactivar la alarma y así poder realizar sus funciones con normalidad el carro, en caso de que este sea un ladrón o una persona ajena al automóvil no tendrá conocimiento de esta alarma o no sabrá como desconectar la alarma, y ya pasado el tiempo de retardo del primer 555 este pasara a un nivel bajo, en ese momento el 555 en una de sus terminales manda una señal al segundo 555 el cual activará una sirena por el tiempo establecido en la configuración del 555.

Al final del análisis de este proyecto de alarma para automóviles tuvieron las complicaciones de no poder llevar todas las funciones preestablecidas en sus objetivos pues solo activaron la alarma, pero no pudieron desactivar las funciones del automóvil, y además la alarma no aplicaron amplificadores para que el sonido fuera más intenso.

En una página de internet cuyo autor no es conocido, nos encontramos con el proyecto llamado “nano phone”, este es un sistema capaz de leer un sensor de temperatura y enviar dichos datos por SMS. Para que éste pareciera un teléfono móvil se le añadió una pantalla LCD a color, un reloj de tiempo real y así ser bautizado como “Nano Phone”.

De este modo además del envío del SMS, añade la posibilidad de visualizar los datos obtenidos tanto del sensor de temperatura, como del reloj, y poder ver la fecha y hora.

Consiste en una placa Arduino UNO y una placa Cellular shield Spark Fun acompañada de complementos como una pantalla LCD, un reloj RTC DS 1307 y un sensor de temperatura LM35.

La placa Cellular shield Spark Fun es una parte importante para este proyecto ya que sin ella no se podrá enviar mensajes vía SMS. Para que así lleve el objetivo principal que es que en un determinado tiempo se detecte la temperatura y esta sea mandada vía GSM al usuario para que este sepa la temperatura de alguna cierta zona.

En base a todas estas tecnologías mencionadas en este apartado se basará el proyecto de localización y bloqueo de maquinaria agrícola, como lo son las tecnologías en alarmas vía GPS y GSM tomando lo mejor de estos proyectos como es el uso del corte de combustible y corte de energía eléctrica a la máquina agrícola. La utilización del arduino y su placa adaptadora para poder trabajar vía GSM/GPS además de ser muy barato y que da muchas libertades de trabajo y diseño de proyectos el cual solo como límite nos pone la imaginación.

Y por último nos encontramos con una aplicación llamada AG1, es una aplicación embebida para el Modem GSM/GPS "m-trac 25" que incorpora un módulo GSM/GPS Wavecom Q2501B. Esta realiza las funciones de alarma con la posibilidad de conocer la ubicación geográfica del mismo así como de la velocidad a la que pudiera estar desplazándose el mismo.

Permite la llamada o el envío de un mensaje SMS a un teléfono móvil o fijo, previamente programado, al activarse una señal en su canal de entrada.

Dentro de las aplicaciones que se le asignan a este sistema podemos encontrar las siguientes:

- Sistemas de alarmas para vehículos e inmuebles.
- Localización de vehículos.
- Aviso de disparo de sensores: termostatos, presostatos, detectores de nivel, etc.

Dentro de sus principales características encontramos que dispone de un canal de entrada, es fácil de programa, posee 5 modos de funcionamiento, los datos de configuración están almacenados en la memoria EEPROM del modem y el AG1 se puede conectar fácilmente a múltiples sistemas de alarma existentes en el mercado para su desactivación remota.

Como se mencionó antes tiene 5 modos de funcionamiento los cuales son los siguientes:

- **Modo 1.** Realiza una llamada de 5 segundos al número pre programado.
- **Modo 2.** Inmediatamente envía un SMS al numero pre programado indicando que la entrada se ha activado, con la finalidad de que no se envíen SMS sucesivos en caso de que persista la señal de alarma.

- **Modo 3.** Inmediatamente envía un SMS indicando que la entera se ha activado. En dicho reporte indicará las coordenadas geográficas en que se encuentra el dispositivo así como la velocidad a la que se desplaza.
- **Modo 4.** Realiza una llamada de aproximadamente 5 segundos al número de teléfono pre programado. Y así durante tres veces, la entrada del AG1 quedará deshabilitada.
- **Modo 5.** Este modo es muy parecido al modo 1 solo que la llamada durará 20 segundos a los primeros tres números pre programados.

Como se mencionó anteriormente estos trabajos son las bases en ideas para la realización del proyecto de sistema de localización y bloqueo de maquinaria agrícola vía GSM/GPS, además de que nos sirve como base en la tecnología a utilizar como son las tecnologías GPS/GSM y el arduino los cuales serán las aplicaciones que nos servirán para echar andar el dispositivo antes mencionado, corrigiendo errores de estos pasados proyectos y tomando las mejores aportaciones, para que el dispositivo tenga un funcionamiento óptimo y sea totalmente comercial.

Capítulo 2

Marco teórico

Capítulo 2. Marco teórico

2.1 Introducción

En el siguiente capítulo, se describirá tanto las tecnologías como las herramientas e instrumentos que utilizaremos para el desarrollo del sistema de localización de alarma y bloqueo de maquinaria agrícola.

Los temas de las tecnologías a desarrollar son:

- Tecnología GPS
- Tecnología GSM.
- Arduino.
- Microprocesadores.

Los instrumentos que se utilizarán y se describirán a detalle en este trabajo son los siguientes:

- Arduino UNO.
- Arduino Shield Cellular SM5100 B.
- Arduino Shield GPS.
- Bomba de combustible eléctrica.
- Tarjeta SIM.

2.2 Tecnología GPS

El Sistema de Posicionamiento Global (GPS) es un sistema de localización, diseñado por el Departamento de Defensa de los Estados Unidos con fines militares para proporcionar estimaciones precisas de posición, velocidad y tiempo; operativo desde 1995 utiliza conjuntamente una red de ordenadores y una constelación de 24 satélites para determinar por triangulación, la altitud, longitud y latitud de cualquier objeto en la superficie Terrestre.

2.2.1 Arquitectura del sistema GPS

El sistema se descompone en tres segmentos básicos:

- **Segmento espacio:** Formado por 24 satélites GPS con una órbita de 26560 Km. De radio y un periodo de 12h.
- **Segmento control:** Consta de cinco estaciones monitoras encargadas de mantener en órbita los satélites y supervisar su correcto funcionamiento, tres antenas terrestres que envían a los satélites las señales que deben transmitir y una estación experta de supervisión de todas las operaciones.
- **Segmento usuario:** Formado por las antenas y los receptores pasivos situados en tierra.

2.2.2 Principios de Funcionamiento de los GPS

Los receptores GPS más sencillos están preparados para determinar con un margen mínimo de error la latitud, longitud y altura desde cualquier punto de la tierra donde nos encontremos situados. Otros más completos muestran también el punto donde hemos estado e incluso trazan de forma visual sobre un mapa la trayectoria seguida o la que vamos siguiendo en esos momentos.

El funcionamiento del sistema GPS se basa también, al igual que los sistemas electrónicos antiguos de navegación, en el principio matemático de la triangulación.

2.2.3 Cómo ubica la posición el receptor GPS

Para ubicar la posición exacta donde nos encontramos situados, el receptor GPS tiene que localizar por lo menos 3 satélites que le sirvan de puntos de referencia. En realidad eso no constituye ningún problema porque normalmente siempre hay 8 satélites dentro del campo visual de cualquier receptor GPS. Para

determinar el lugar exacto de la órbita donde deben encontrarse los satélites en un momento dado, el receptor tiene en su memoria un almanaque electrónico que contiene esos datos.

Tanto los receptores GPS de mano, como los instalados en vehículos con antena exterior fija, necesitan abarcar el campo visual de los satélites. Generalmente esos dispositivos no funcionan bajo techo ni debajo de las copas de los árboles, por lo que trabajan con precisión hay que situarlos en el exterior, preferiblemente donde no existan obstáculos que impidan la visibilidad y reduzcan su capacidad de captar las señales que envían a la tierra los satélites.

El principio de funcionamiento de los receptores GPS es el siguiente:

Primero: Cuando el receptor detecta el primer satélite se genera una esfera virtual o imaginaria, cuyo centro es el propio satélite. El radio de la esfera, es decir, la distancia que existe desde su centro hasta la superficie, será la misma que separa al satélite del receptor. Éste último asume entonces que se encuentra situado en un punto cualquiera de la superficie de la esfera, que aún no puede precisar.

Segundo: Al calcular la distancia hasta un segundo satélite, se genera otra esfera virtual. La esfera anteriormente creada se superpone a esta otra y se crea un anillo imaginario que pasa por los dos puntos donde se interceptan ambas esferas. En ese instante ya el receptor reconoce que sólo se puede encontrar situado.

Tercero: El receptor calcula la distancia a un tercer satélite y se genera una tercera esfera virtual. Esa esfera se corta con un extremo del anillo anteriormente creado en un punto en el espacio y con el otro extremo en la superficie de la Tierra. El receptor discrimina como ubicación el punto situado en el espacio utilizando sus recursos matemáticos de posicionamiento y toma como posición correcta.

Cuarto: Una vez que el receptor ejecuta los tres pasos anteriores ya puede mostrar en su pantalla los valores correspondientes a las coordenadas de su

posición.

Quinto: Para detectar también la altura a la que se encuentra situado el receptor GPS sobre el nivel del mar, tendrá que medir adicionalmente la distancia que lo separa de un cuarto satélite y generar otra esfera virtual que permitirá determinar esa medición.

2.2.4 Fuente de errores de los GPS

A continuación se mencionan las fuentes de error que en la actualidad afectan de forma significativa a las medidas realizadas con el GPS:

- Perturbación de la ionosfera.
- Fenómenos meteorológicos.
- Imprecisión en los relojes.
- Interferencias eléctricas imprevistas.
- Error multisenda.
- Interferencia "Disponibilidad Selectiva S/A
- Topología receptor-satélite.

2.3 Tecnología GSM

La red GSM (Sistema global de comunicaciones móviles), a comienzos del siglo XXI, es el estándar más usado de Europa. Se denomina estándar "de segunda generación" (2G) porque, a diferencia de la primera generación de teléfonos portátiles, las comunicaciones se producen de un modo completamente digital.

En Europa, el estándar GSM usa las bandas de frecuencia de 900MHz y 1800 MHz. Sin embargo, en los Estados Unidos se usa la banda de frecuencia de 1900 MHz.

El estándar GSM permite un rendimiento máximo de 9,6 kbps, que permite transmisiones de voz y de datos digitales de volumen bajo, por ejemplo, mensajes de texto (SMS, Servicio de mensajes cortos) o mensajes multimedia (MMS, Servicio de mensajes multimedia).

2.3.1 Arquitectura de la red GSM

En una red GSM, la terminal del usuario se llama estación móvil. Una estación móvil está constituida por una tarjeta SIM (Módulo de identificación de abonado), que permite identificar de manera única al usuario y a la terminal móvil, o sea, al dispositivo del usuario (normalmente un teléfono portátil).

Las terminales (dispositivos) se identifican por medio de un número único de identificación de 15 dígitos denominado IMEI (Identificador internacional de equipos móviles). Cada tarjeta SIM posee un número de identificación único (y secreto) denominado IMSI (Identificador internacional de abonados móviles). Este código se puede proteger con una clave de 4 dígitos llamada código PIN.

Por lo tanto, la tarjeta SIM permite identificar a cada usuario independientemente de la terminal utilizada durante la comunicación con la estación base. Las comunicaciones entre una estación móvil y una estación base se producen a través de un vínculo de radio, por lo general denominado interfaz de aire.

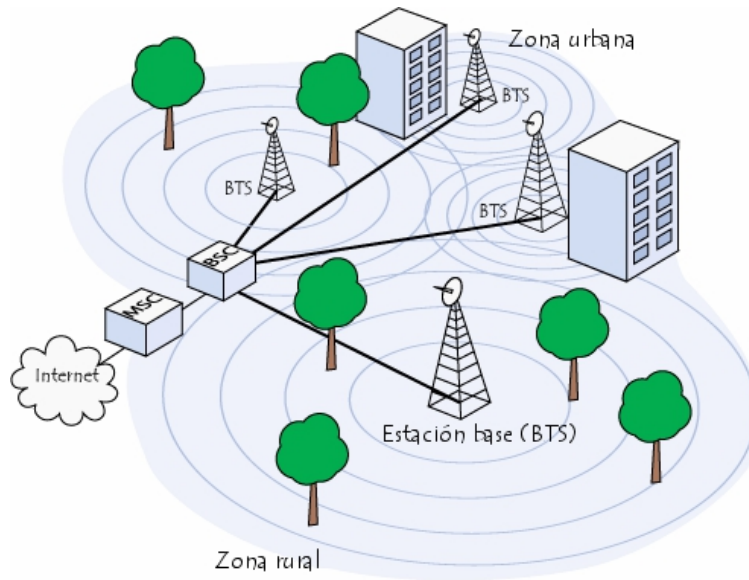


Figura 2.1 Esquema de funcionamiento de un sistema GSM

Todas las estaciones base de una red celular están conectadas a un controlador de estaciones base (o BSC), que administra la distribución de los recursos. El sistema compuesto del controlador de estaciones base y sus estaciones base conectadas es el Subsistema de estaciones base (o BSS).

Por último, los controladores de estaciones base están físicamente conectados al Centro de conmutación móvil (MSC) que los conecta con la red de telefonía pública y con Internet; lo administra el operador de la red telefónica. El MSC pertenece a un Subsistema de conmutación de red (NSS) que gestiona las identidades de los usuarios, su ubicación y el establecimiento de comunicaciones con otros usuarios.

Generalmente, el MSC se conecta a bases de datos que proporcionan funciones adicionales:

- **El Registro de ubicación de origen (HLR):** es una base de datos que contiene información (posición geográfica, información administrativa, etc.) de los abonados registrados dentro de la zona del conmutador (MSC).

- **El Registro de ubicación de visitante (VLR):** es una base de datos que contiene información de usuarios que no son abonados locales. El VLR recupera los datos de un usuario nuevo del HLR de la zona de abonado del usuario. Los datos se conservan mientras el usuario está dentro de la zona y se eliminan en cuanto abandona la zona o después de un período de inactividad prolongado (terminal apagada).
- **El Registro de identificación del equipo (EIR):** es una base de datos que contiene la lista de terminales móviles.
- **El Centro de autenticación (AUC):** verifica las identidades de los usuarios.

La red celular compuesta de esta manera está diseñada para admitir movilidad a través de la gestión de traspasos (movimientos que se realizan de una celda a otra).

Finalmente, las redes GSM admiten el concepto de roaming: el movimiento desde la red de un operador a otra.

2.4 Arduino

Arduino es una plataforma de electrónica abierta para la creación de prototipos basada en software y hardware flexibles y fáciles de usar. Se creó para artistas, diseñadores, aficionados y cualquiera interesado en crear entornos u objetos interactivos.

Arduino puede tomar información del entorno a través de sus pines de entrada de toda una gama de sensores y puede controlar todo aquello que le rodee como luces, motores y otros actuadores. El microcontrolador en la placa Arduino se programa mediante el lenguaje de programación Arduino y el entorno de desarrollo Arduino. Los proyectos hechos con Arduino pueden ejecutarse sin necesidad de conectarse a una computadora, si bien tienen la posibilidad de hacerlo y comunicar con diferentes tipos de software.

El software de Arduino consiste en un entorno de desarrollo (IDE) y las bibliotecas del núcleo. El IDE está escrito en Java y basado en un entorno de desarrollo de procesamiento. Las bibliotecas del núcleo están escritas en C y C + + y compilado usando AVR-GCC y LIBC AVR. El código fuente para Arduino está ahora alojado en GitHub.

2.5 Arduino Uno

El Arduino Uno es una placa electrónica basada en el ATmega328. Cuenta con:

1. 14 entradas/salidas digitales en forma de pines.
2. 6 entradas analógicas.
3. Un oscilador de cristal de 16 MHz
4. Una conexión USB
5. Un conector de alimentación
6. Una cabecera de ICSP
7. Un botón de reinicio.

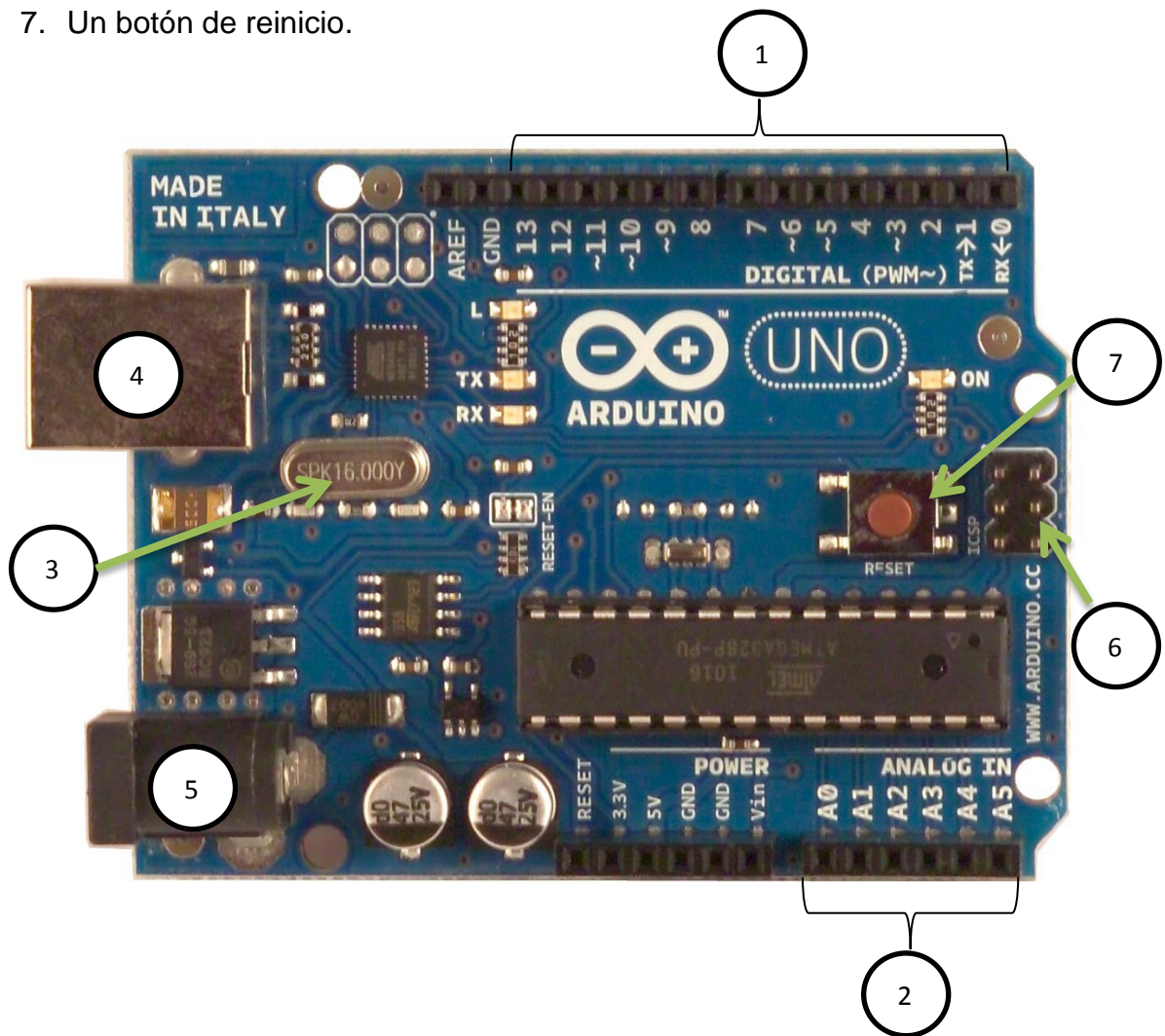


Figura 2.2 Vista frontal del Arduino Uno.

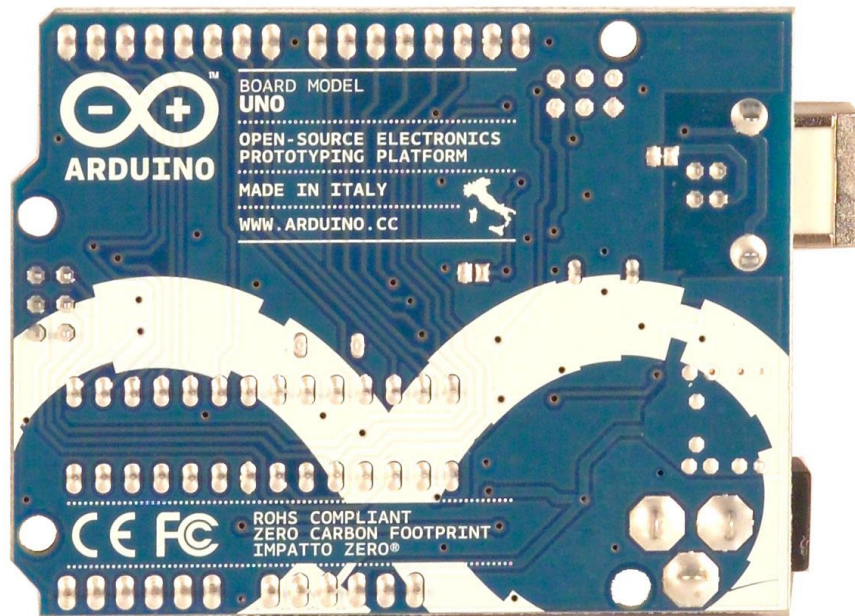


Figura 2.3 Vista trasera del Arduino Uno

Contiene todo lo necesario para apoyar al microcontrolador, sólo tiene que conectarlo a una computadora con un cable USB o energizarlo con un adaptador AC-DC o una batería para empezar su uso.

La placa tiene las siguientes características:

- 1º pin de salida: añade pines SDA y SCL que se encuentran cerca del pin AREF y dos pasadores de otros nuevos que se pongan cerca del pin de RESET, el IOREF que permite a los escudos de adaptarse a la tensión proporcionada por la placa. El segundo es un pin que no está conectado.

- Una función de reinicio con mayor estabilidad del circuito Arduino Uno.
- Atmega 16U2 sustituye al 8U2, programado como un convertidor de USB a serie.
- La tarjeta Uno tiene una resistencia tirando de la línea HWB 8U2 a tierra, por lo que es más fácil poner en modo DFU.

2.5.1 Resumen

Microcontrolador	ATmega328
Voltaje de apertura	5V
Voltaje de entrada	7-12V
Voltaje de salida	6-20V
Pines digitales E/S	14 (of which 6 provide PWM output)
Pines de salida análogos	6
Corriente DC de pin de E/S	40 mA
Corriente DC para pin 3.3V	50 mA
Memoria flash	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Reloj de aceleración.	16 MHz

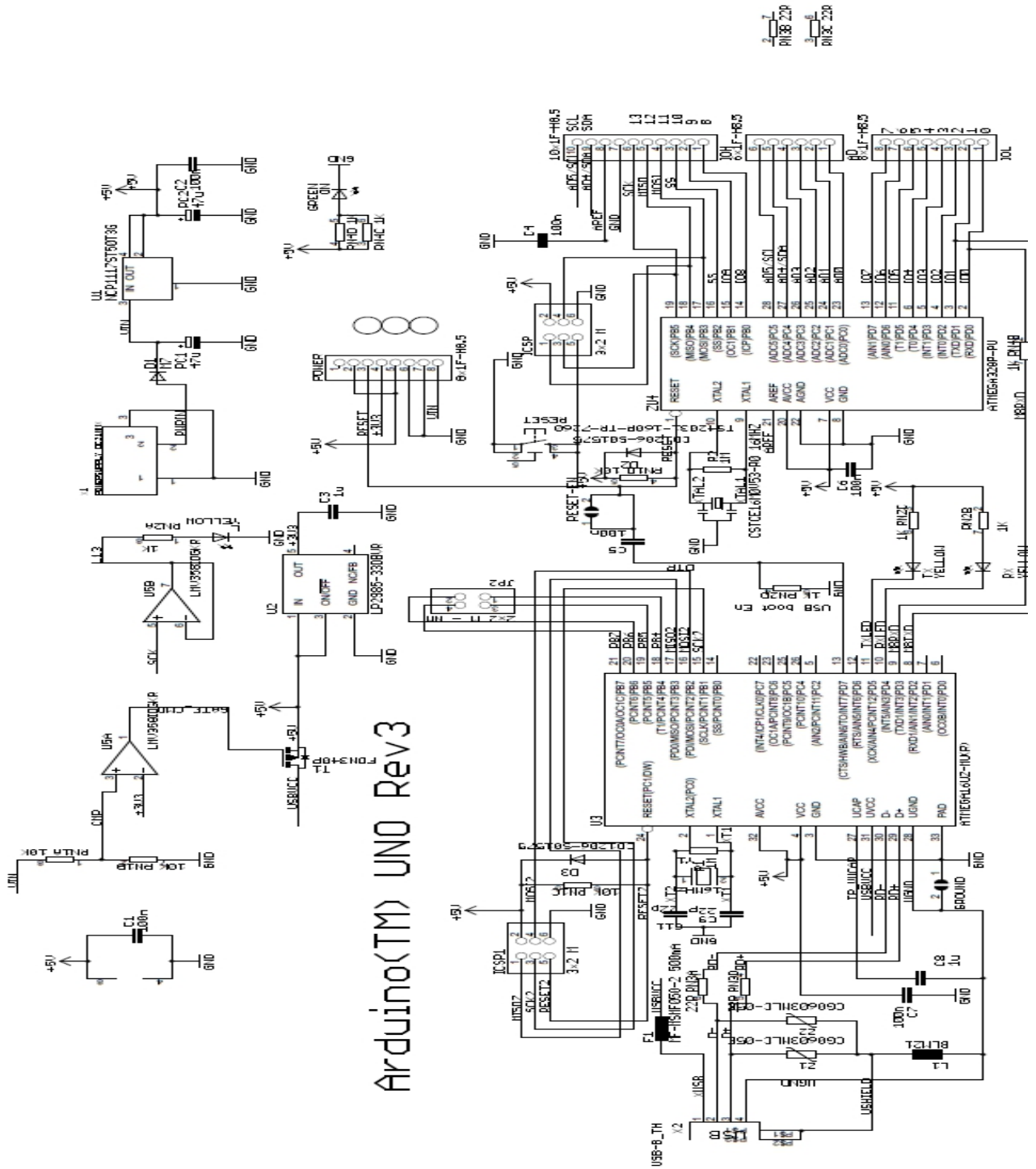


Figura 2.4 Diagrama eléctrico del Arduino Uno.

2.5.2 Alimentación

El Uno Arduino puede alimentarse a través de la conexión USB o con una fuente de alimentación externa.

La alimentación externa (no USB), puede ser alimentada con un adaptador de CA a CD o con una batería. El adaptador se puede conectar al enchufe de 2.1mm-positivo. En caso de usar una batería, se puede insertar en los encabezados del pin GND y Vin del conector de alimentación.

La placa puede operar con un suministro externo de 6 a 20 voltios. Si se suministran con menos de 7V, la tarjeta puede ser inestable. Si utiliza más de 12V, el regulador de voltaje se puede sobrecalentar y dañar la placa. El rango recomendado es de 7 a 12 voltios.

Los pines de alimentación son los siguientes:

- VIN. El voltaje de entrada a la placa Arduino UNO cuando se utiliza una fuente de alimentación externa (a diferencia de 5 voltios de la conexión USB o de otra fuente de alimentación regulada).
- 5V. La fuente de alimentación regulada, se utilizada para alimentar el microcontrolador y otros componentes en el tablero.
- 3.3V. Un suministro de 3.3 voltios generada por el regulador a bordo.
- Consumo de corriente máxima es de 50 mA.
- GND. pin de tierra.

2.5.3 Memoria Atmega328

El Atmega328 tiene 32 MB (con 0,5 KB utilizados para el gestor de arranque). También dispone de 2 KB de SRAM y 1 KB de memoria EEPROM (que puede ser leído y escrito con la librería EEPROM).

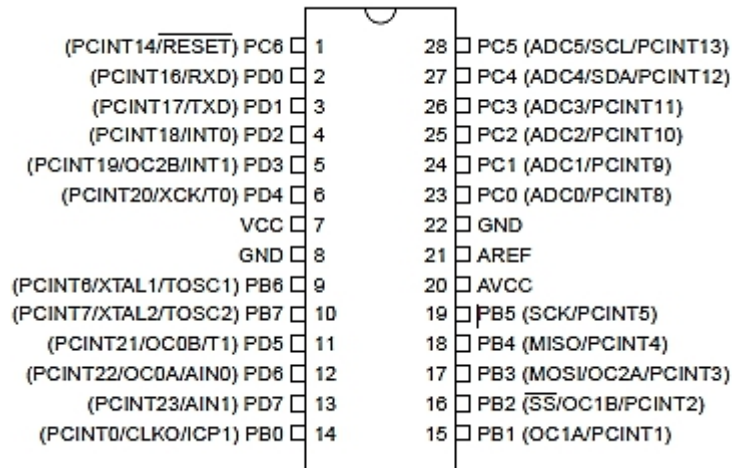


Figura 2.5 Configuración de pines del ATMEGA 328.

2.5.4 Entrada y salida

Cada uno de los 14 pines digitales en el Arduino Uno se puede utilizar como pin de entrada o de salida, usando las funciones `pinMode`, `digitalWrite`, y las funciones de `digitalRead`. Ellos operan a 5 voltios. Cada pin puede proporcionar o recibir un máximo de 40 mA y tiene una resistencia interna pull-up de 20 a 50 kOhm. Además, algunos pines tienen funciones especializadas:

- **Serie: pin 0 (RX) y pin 1 (TX).** Se utiliza para recibir (RX) y transmitir (TX) datos serie TTL. Estos se encuentran conectadas a los pines correspondientes de la ATmega8U2 USB-a-ttl de serie chip.
- **Interruptor externo: pin 2 y pin 3.** Estos pines se pueden configurar para desencadenar una interrupción en un valor bajo, un borde ascendente o descendente, o un cambio en el valor.
- **PWM: pines 3, 5, 6, 9, 10 y 11.** Proporcionar 8-bit de salida PWM con la función `analogWrite` (escritura análoga).

- **SPI: pin 10 (SS), pin 11 (MOSI), pin 12 (MISO), pin 13 (SCK).** Estos pines dan apoyo a la comunicación SPI utilizando la biblioteca de SPI.
- **LED: pin 13.** Hay un built-in LED conectado al pin digital 13. Cuando el pin es de alto valor, el LED está encendido, cuando el pin pasa a valor bajo, está apagado.

El Uno tiene 6 entradas analógicas, etiquetados A0 a A5, cada una proporcionan 10 bits de resolución (es decir 1024 valores diferentes). Por defecto miden desde tierra hasta 5 voltios, aunque es posible cambiar el extremo superior de su rango con el pasador y el AREF función análoga de referencia. Además, algunos pines tienen funciones especializadas:

- **IST: A4 o A5 y SDA pin o pines SCL.** Apoyar la comunicación TWI utilizando la librería Wire.

Hay un par de pines en el tablero:

- **AREF.** Tensión de referencia para las entradas analógicas. Se utiliza con analogReference (referencia analogica).
- **Restablecer.** Lleve esta línea LOW para reiniciar el microcontrolador. Normalmente se utiliza para añadir un botón de reinicio para escudos que bloquean el Arduino UNO en el tablero.

2.5.5 Comunicación

El Arduino Uno tiene una serie de facilidades para comunicarse con un ordenador, otro Arduino, u otros microcontroladores. El ATmega328 ofrece UART TTL (5V) de comunicación en serie, que está disponible en los pines digitales 0 (RX) y 1 (TX). El ATmega16U2 en los canales de comunicación a bordo esta serie a través de USB y aparece como un puerto COM virtual con el software en el ordenador. El firmware '16U2 utiliza el estándar de los controladores USB, COM, y no hay ningún controlador externo. Sin embargo, en Windows, se requiere un

archivo .Inf. El software de Arduino incluye un monitor en serie que permite que se envíen los datos de texto hacia la placa Arduino. Los LEDs RX y TX en el tablero parpadea cuando se están transmitiendo datos a través de la placa vía USB en serie y la conexión USB a la computadora (pero no para la comunicación de serie en los pines 0 y 1).

2.5.6 Programación

El Arduino UNO puede ser programado con el software de Arduino. El ATmega328 en la Arduino UNO viene pre-quemado con un gestor de arranque que le permite cargar un nuevo código a la misma sin el uso de un hardware programador externo. Se comunica utilizando el protocolo original STK500 (de referencia, archivos de cabecera C).

También puede pasar por alto el gestor de arranque y el programa del microcontrolador a través de la ICSP (programación In-Circuit Serial) cabecera.

Para el ATmega16U2 el código fuente está disponible el firmware. El ATmega16U2/8U2 está cargado con un cargador de arranque DFU, que puede ser activado por:

- **En Rev1 juntas:** conectar el puente de soldadura en la parte posterior de la placa y luego reiniciar el 8U2.
- **En Rev2 o tablas posteriores:** hay una resistencia que tirando de la línea de 8U2/16U2 HWB a tierra, por lo que es más fácil poner en modo DFU.

Puede utilizar el software de la FLIP de Atmel (Windows) o el programador DFU (Mac OS X y Linux) para cargar un nuevo firmware. O puede utilizar el encabezado de ISP con un programador externo (sobrescribir el gestor de arranque DFU).

2.5.7 Características físicas

La longitud máxima y la anchura de la placa de Arduino UNO son 6.858 cm y 5.334cm respectivamente, con el conector USB y conector de alimentación se extiende más allá de la dimensión anterior. Cuatro orificios de los tornillos permite que la tarjeta sea sujeto a una superficie o caja. Obsérvese que la distancia entre los pines digitales 7 y 8 es de 0.4064cm no un múltiplo par de la separación 0.254cm de los pasadores de otros.

2.6 Microprocesadores

Una computadora digital es una combinación de dispositivos y circuitos digitales que pueden realizar una secuencia en operaciones con una mínima intervención humana. A la secuencia de operaciones se le denomina programa. El programa es un conjunto de instrucciones codificadas que se almacenan en la memoria interna de la computadora con todos los datos que el programa requiere. Cuando a la computadora se le ordena ejecutar el programa, ésta lleva a cabo las instrucciones en el orden en que están almacenadas en la memoria hasta que el programa se completa.

El microprocesador es la parte más importante de una microcomputadora, el microprocesador está contenido en un circuito integrado que contiene todos los circuitos de la unidad de control y lógica aritmética, en otras palabras la CPU. MICROPROCESADOR ó CPU (Unidad de Procesamiento Central). Es un paquete de circuitos integrados que contiene una unidad procesadora (ALU) y una unidad de control.

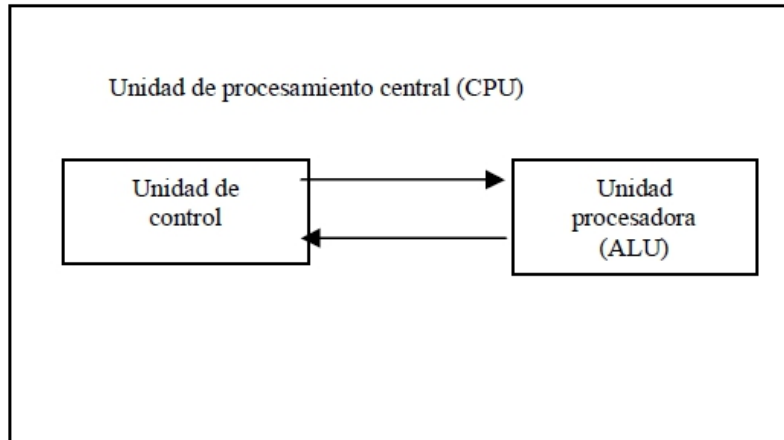


Figura 2.6 Diagrama a bloques de un CPU.

2.6.1 Tipos de microprocesadores

Los microprocesadores se dividen por el tamaño de la palabra de datos, la palabra de datos es el número de bits que puede el microprocesador manejar a la vez, los tamaños más usados de palabra son: 8,16 y 32 bits. Por ejemplo: sumar dos números de ocho bits para un micro procesador de 8 bits tomaría un tiempo dado "To"; En contra parte un MICRO PROCESADOR de 32 bits la suma de entre dos números la efectuaría en un tiempo "To", por lo anterior el micro procesador de 32 bits tiene que efectuar más operaciones en tiempo "To". Otra clasificación valida es por la cantidad de partes electrónicas que puede tener un CI o CHIP, esta clasificación es válida en la cantidad de compuertas integradas, pero todos los microprocesadores usan integración a gran escala (LSI).

2.7 Tarjeta SIM

Una tarjeta SIM es una tarjeta inteligente desmontable usada en teléfonos móviles y módems USB. Las tarjetas SIM almacenan de forma segura la clave de

servicio del suscriptor usada para identificarse ante la red, de forma que sea posible cambiar la línea de un terminal a otro simplemente cambiando la tarjeta.

El uso de las tarjetas SIM es obligatorio en las redes GSM y, fue a partir de este momento, cuando empezó a forjarse el éxito de estas tarjetas, ya que de la mano de la tecnología GSM, evolucionó y se volvió más económica, más eficaz y cómoda por su tamaño.

Las tarjetas SIM están disponibles en dos tamaños: el primero es similar al de una tarjeta de crédito: 85,60 x 53,98 x 0,76 mm, y el segundo y más popular es la versión pequeña: 25 x 15 x 0,76 mm. En cuanto a la capacidad de almacenamiento, la típica tarjeta SIM de bajo coste tiene poca memoria, 2-3 kB, según describe la especificación (directorio telefónico y poco más). Este espacio de almacenamiento es usado directamente por el teléfono. El segmento de las tarjetas de bajo coste está en declive.

Las tarjetas SIM con aplicaciones adicionales están disponibles con muchas capacidades de almacenamiento diferentes, siendo la mayor 512 kB. Las tarjetas de 32 y 16 kB son las dominantes en zonas con redes GSM menos desarrolladas. También existen las tarjetas Large Memory SIM, con capacidades del orden de 128 a 512 kB.



Figura 2.7 Tarjeta SIM versión pequeña: 25 x 15 x 0,76 mm.

Los sistemas operativos para las tarjetas SIM son principalmente dos: nativos, con software propietario y específico del vendedor, correspondiendo típicamente con el segmento de mercado de bajo coste, y basados en Java, que tienen la ventaja de ser independientes del hardware e interoperables. Las tarjetas SIM poseen diversos recursos que le permiten actuar dentro del dispositivo móvil como: recursos de seguridad e identificación (posibilidad de brindarle una identificación única al usuario en la red móvil), la posibilidad de tener en el teléfono, y con interfaces muy sencillas de usar, servicios de valor agregado como servicio de mensajería instantánea (chat), televisión móvil, juegos, tonos, etc. o la posibilidad de acceder a redes sociales y generar respaldos de agenda que pueden ser alojados remotamente permitiendo así agilidad, economía y seguridad. También nos permite acceder a servicios financieros seguros desde el teléfono móvil.

2.8 Arduino Shield Cellular SM5100 B

El shield de telefonía celular para Arduino incluye todos los componentes necesarios para emplear el Arduino con un módulo celular SM5100B. Esto le permitirá agregar fácilmente funcionalidades de SMS, GSM/GPRS y TCP/IP a tus proyectos. Lo único adicional que se requiere es una tarjeta SIM para que se pueda empezar a enviar comandos seriales y de esta manera realizar llamadas, enviar mensajes de texto e incluso funcionar como servidor web.

Los principales componentes de este shield son un conector de 60 pines SM5100B, un socket de tarjeta SIM y un regulador de voltaje SPX29302 configurado para regular el voltaje del Arduino a 3.8V. Así mismo posee un LED rojo para indicar que está conectada a la alimentación. El botón de reset del Arduino se encuentra disponible también en este shield. Mediante los dos jumpers en este shield se puede seleccionar cuales pines seriales del Arduino se conectarán al módulo celular: software (D2, D3) o hardware (D0, D1). También

2.9 Arduino Shield GPS

Con el Shield de GPS se puede agregar la funcionalidad de GPS a su Arduino. Un conector para el popular EM-406 receptor GPS que está acoplado en el tablero, y las pistas de EM-408 y los conectores de EB-85A también están disponibles. Los pines regulares GPS (RX, TX, PPS, etc) también se rompe a una de 10 pines 0,1 "cabecera terreno de juego", y un área pequeña de prototipos. El interruptor de DLINE / UART cambia la entrada del módulo GPS / salida entre el estándar de pines del Arduino TX / RX y pines digitales 2 y 3. DLINE deben ser seleccionados con el fin de subir el código a través de la IDE de Arduino.

Un interruptor ON / OFF está incluido, el cual controla la potencia del módulo GPS. Por último, el botón de reinicio Arduino también es compatible con el Shield.

.Características:

EM-406 conector poblado y conectado al Arduino pines UART (D0/D1) EM-408 y EB-85A pistas de conexión siempre y conectado para su uso opcional Célula de la batería toma la pista siempre y conectado para copia de seguridad de la batería opcional de EB-85A módulo GPS Arduino estándar de tamaño regular. Prototipos área de GPS de serie y señales PPS estallado a un 0,1 "de cabecera Arduino botón de reinicio Interruptor ON / OFF controla la energía de un módulo GPS.

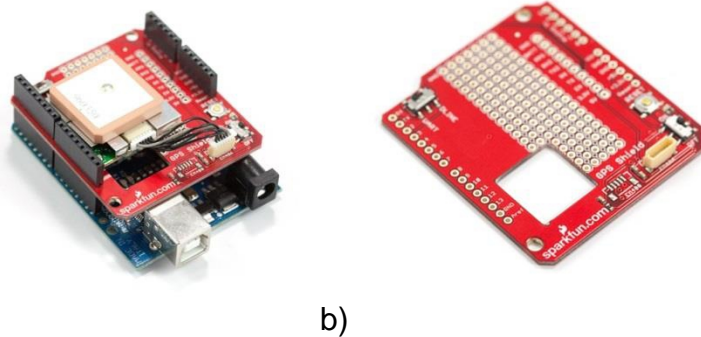


Figura 2.10. a) Arduino Shield GPS montado a Arduino UNO. B) Parte trasera de Arduino Shield GPS

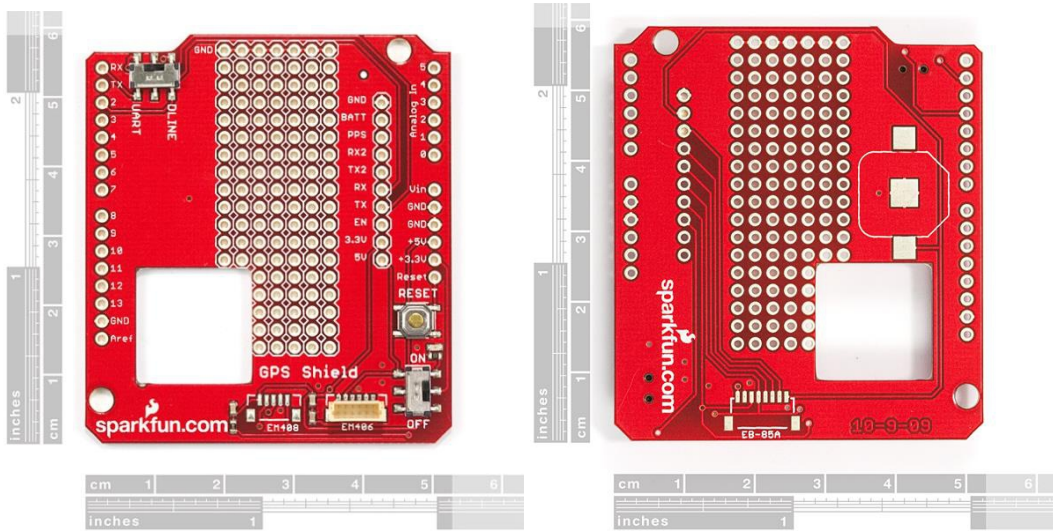


Figura 2.11. Mediciones de la parte trasera con medidas del Arduino shield GPS.

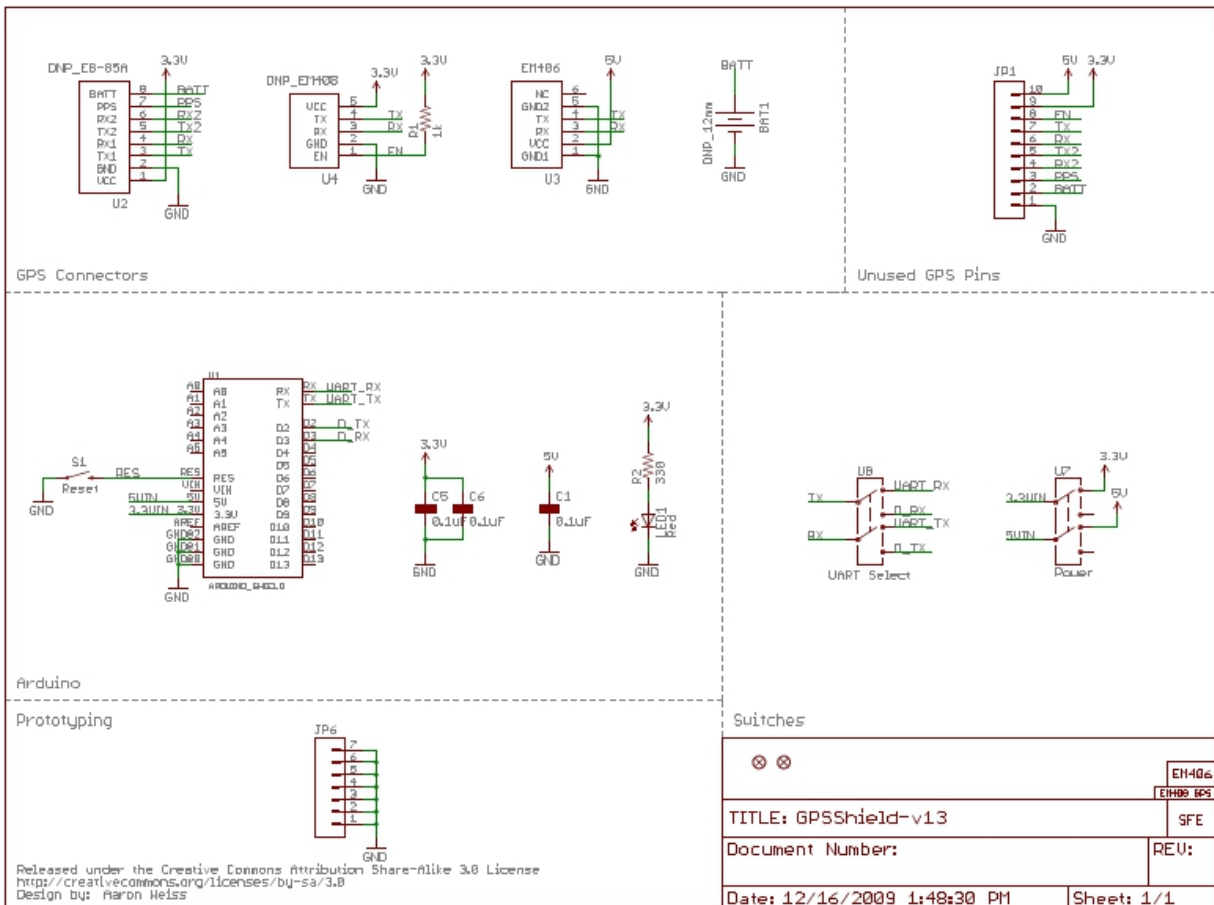


Figura 2.12. Diagrama esquemático del Arduino Shield GPS.

2.10 Bomba de combustible eléctrica

La bomba eléctrica de combustible forma parte del sistema de alimentación del automóvil y puede encontrarse tanto dentro del tanque del combustible como fuera de él, denominados según el caso IN TANK o IN LINE respectivamente.

Su función consiste en suministrar el combustible necesario para el funcionamiento del motor.

Dado que la presión de la alimentación de combustible debe permanecer constante cualquiera sea el régimen del motor el combustible es entonces suministrado en un caudal mayor de lo realmente necesario, volviendo el excedente nuevamente al tanque.

En la imagen 2.13 puede apreciarse un esquema de este tipo de bombas, con sus conductos, inducido, admisión y salida.

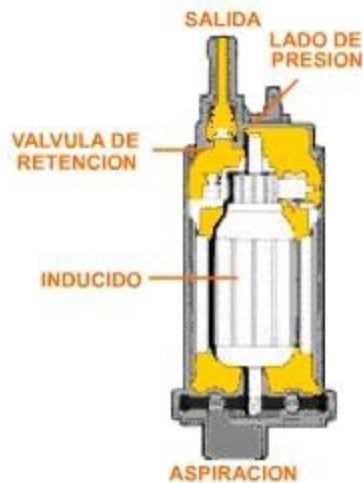


Figura 2.13. Esquema de una bomba de combustible

Estas bombas eléctricas entregan un flujo constante de combustible al motor y el combustible que no es usado, es retornado al tanque y esto también reduce la posibilidad de calentar al mismo dado que no se acerca a partes calientes del motor por mucho tiempo. La llave de ignición no acciona directamente a la bomba eléctrica, sino que lo hace a través de un relay, dado que este puede manejar grandes corrientes. Por otro lado, es muy común que este relay se oxide y deje de funcionar y es lo más común en las fallas de las bombas.

Por este motivo, muchos motores modernos utilizan controles de estado sólido que permiten controlar la presión vía una modulación por ancho de pulso del voltaje de la bomba. Esto permite el incremento de la vida útil de las bombas, que los dispositivos sean más pequeños y livianos y permite también reducir la carga eléctrica. Algunos automóviles con ECU tienen una lógica de seguridad que detiene a la bomba aún con la ignición presente, si no hay combustible, si existe algún daño de rodamientos en la misma o por algún accidente. Para estos últimos casos, también previene la pérdida de combustible ante la rotura de alguna línea. El armado de una bomba puede ser una combinación de la bomba propiamente dicha, filtro y algún dispositivo electrónico usado para medir la cantidad de combustible que existe en el tanque vía un flotante o sensor que envía datos a una aguja en el tablero de instrumentos.



Figura 2.14. Esquema de una bomba de combustible eléctrica

2.10.1 Especificaciones generales de una bomba de combustible eléctrica

Aplicación

Diseñado para la transferencia segura de los combustibles de baja viscosidad derivados del petróleo, como la gasolina (hasta de 15% mezclas de alcohol, tales como E15), el combustible diesel (las mezclas de hasta un 20% de biodiesel como B20) y el queroseno. La bomba está diseñada para un montaje permanente en los tanques de almacenamiento ventilados.

Carcasa de la bomba: Hierro fundido

Rendimiento: Índice de la bomba: Hasta 25 GPM (94 LPM) Hasta 25 GPM (94 LPM)

Ciclo de funcionamiento: 30 min. encendido, 30 min. apagado 30 min. encendido, 30 min. apagado

Máxima succión: Hasta 15 feet (4,6 metros) Hasta 15 feet (4,6 metros)

Máxima descarga: Hasta 10 feet (3 metros) Hasta 10 feet (3 metros)

Temperatura de funcionamiento: -20° F a +125° F (-29° C a +52° C)

Presión de trabajo: 20 PSI

Especificaciones eléctricas:

- Entrada: 12 voltios DC 24 voltios DC
- Consumo de corriente: 35 amperios 20 amperios
- Motor: 2000 RPM, 4/10 CV (300 vatios) 2000 RPM, 4/10 CV (300 vatios)
- Protección del motor: Disyuntor de 40 amperios Disyuntor de 20 amperios
- Cable: 15 pies de calibre 10 (4,6 metros) 15 pies de calibre 10 (4,6 metros)
- Fusible: 40 amperios 20 amperios

Conexión mecánica:

- Tapón: 2 pulgadas NPT 2 pulgadas NPT
- Entrada: 1 pulgada NPT 1 pulgada NPT
- Salida: 1 pulgada NPT 1 pulgada NPT

2.10 Estado financiero

Insumos.	Programado.	Costo por unidad.	Abril.	Mayo.	Junio.	Julio.	Subtotal.
Arduino UNO.	1	\$464.00	1	0	0	0	\$464.00
Atmega 328	1	\$90.00	1	0	0	0	\$90.00
Bomba de combustible eléctrica	1	\$1500.00	1	0	0	0	\$1500.00
Tarjeta SIM	1	\$150.00	1	0	0	0	\$150.00
Arduino celular shield SM5100B GSM	1	\$1600.00	1				\$1600.00
Celular	1	\$500.00	1	0	0	0	\$500.00
GPS shield	1	\$1700.00	1	0	0	0	\$1700.00
Batería 12V	1	\$700.00	1	0	0	0	\$700.00
Alambre	50m	\$30.00	1	0	0	0	\$30.00
							Total: \$6734.00

Capítulo 3

Diseño e implementación.

Capítulo 3. Diseño e implementación.

En este capítulo se describe el diseño que se ha aplicado para el funcionamiento del sistema de localización y bloqueo de maquinaria agrícola vía GSM/GPS, mostrando una descripción detallada del diseño de los circuitos y los algoritmos implementados para el óptimo funcionamiento de este prototipo.

3.1 Esquema de funcionamiento de sistema de localización, alarma y bloqueo de una maquinaria agrícola.

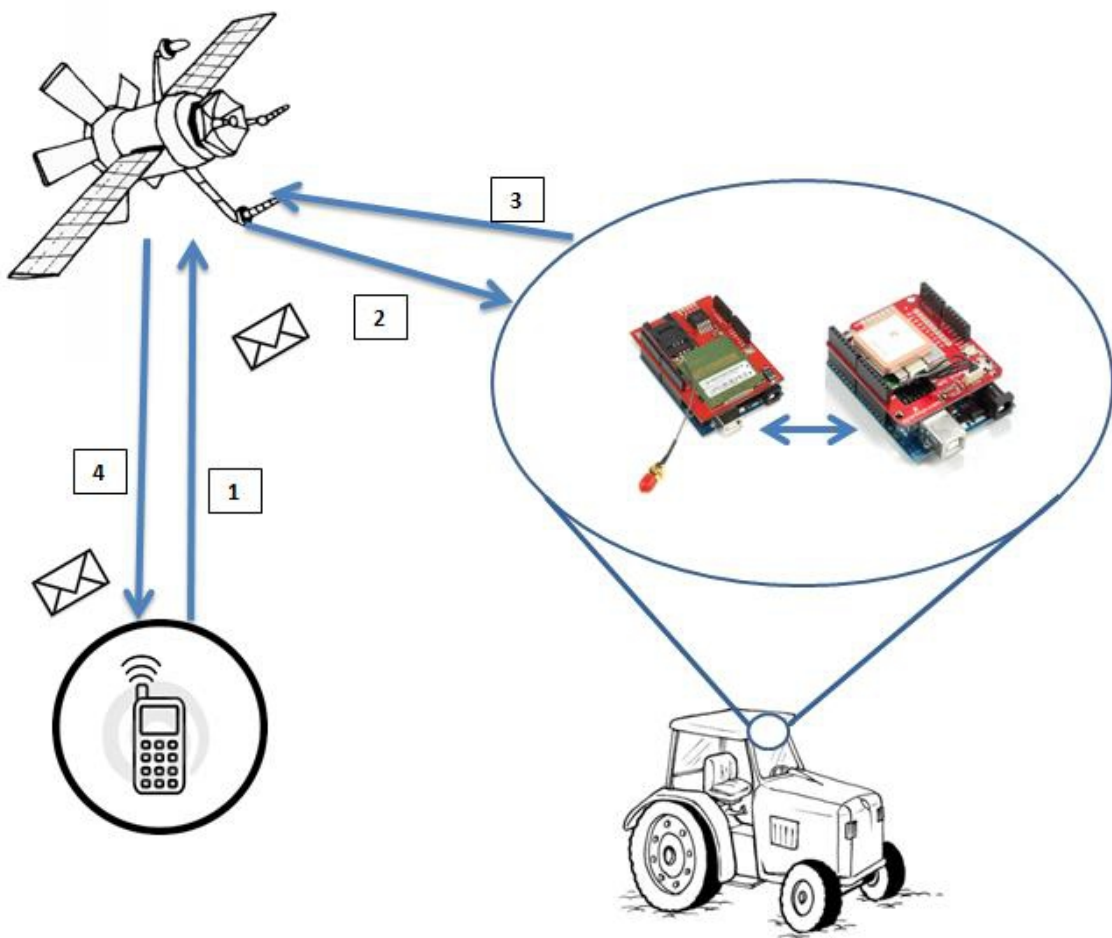
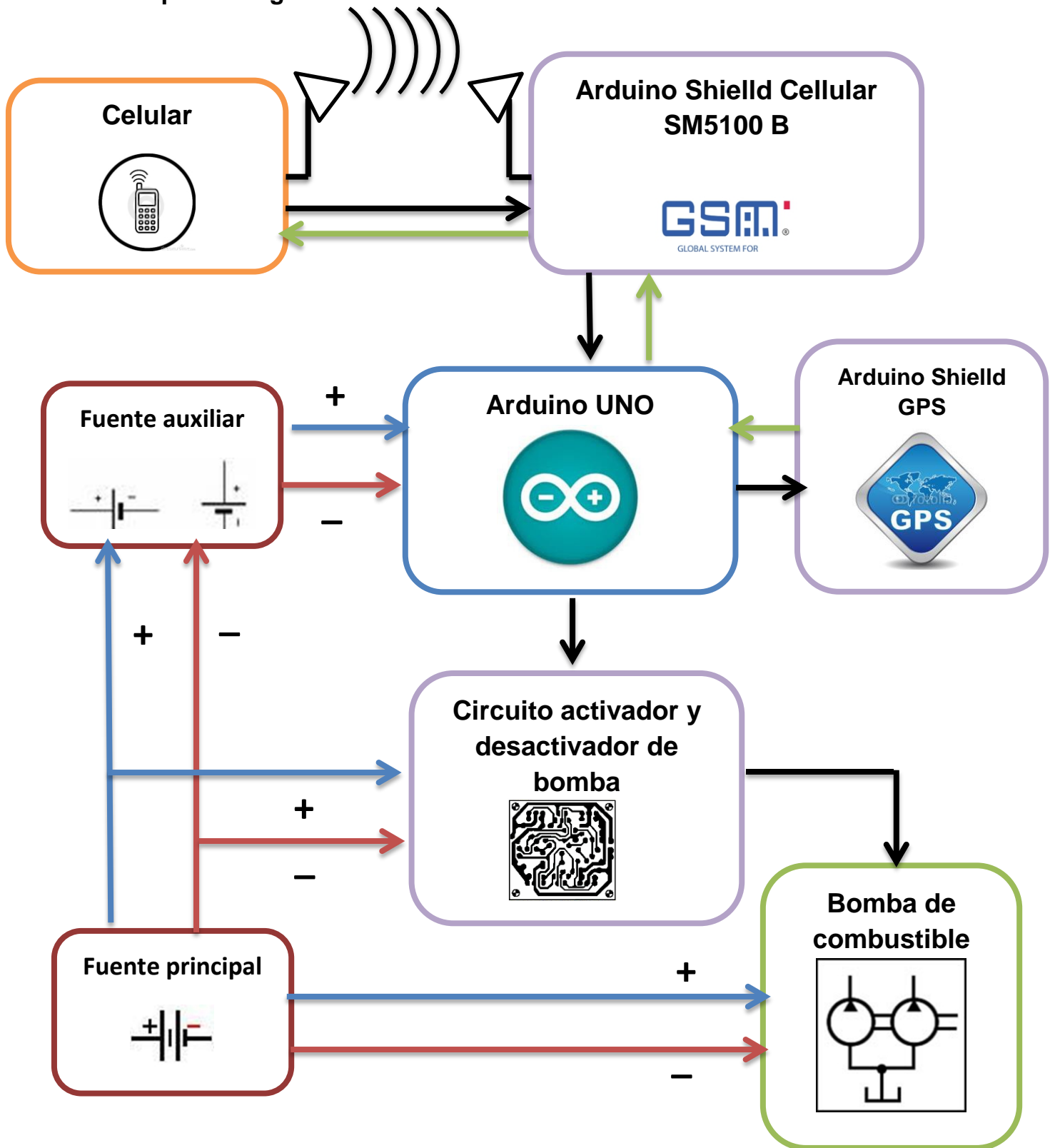


Figura 3.1 Esquema de funcionamiento del sistema de localización alarma y bloqueo de maquinaria agrícola.

3.2 Fases del funcionamiento

1. Al no encontrar nuestra maquinaria agrícola o en caso de ser robado, desde nuestro celular se enviará un mensaje.
2. Este llegará al dispositivo instalado en la maquinaria agrícola y se encargara de detener y activar una alarma que bloquee las funciones de la maquinaria agrícola.
3. El dispositivo regresará un mensaje con los datos en donde esta ubicado.
4. El celular del cual enviamos el mensaje recibirá el mensaje con las coordenadas geográficas (longitud, latitud y altitud) de donde esta ubicado la maquinaria agrícola desaparecida.

3.3 Diagrama a bloques del sistema de localización alarma y bloqueo de maquinaria agrícola



1. Se envía un mensaje vía SMS por medio del celular al momento de que la maquinaria agrícola desaparece, este mensaje se envía al Arduino.
2. El mensaje es recibido por el Shield GSM el cual envía una señal que activa al Arduino UNO para que este mande la señal al Shield GPS y a un circuito desactivador de bomba de combustible.
3. El Shield regresara un mensaje al Arduino UNO el cual con el apoyo del Shield GPS regresara un mensaje al celular indicando las coordenadas de localización de la máquina.

3.4 Diagrama de flujo del proceso del proyecto

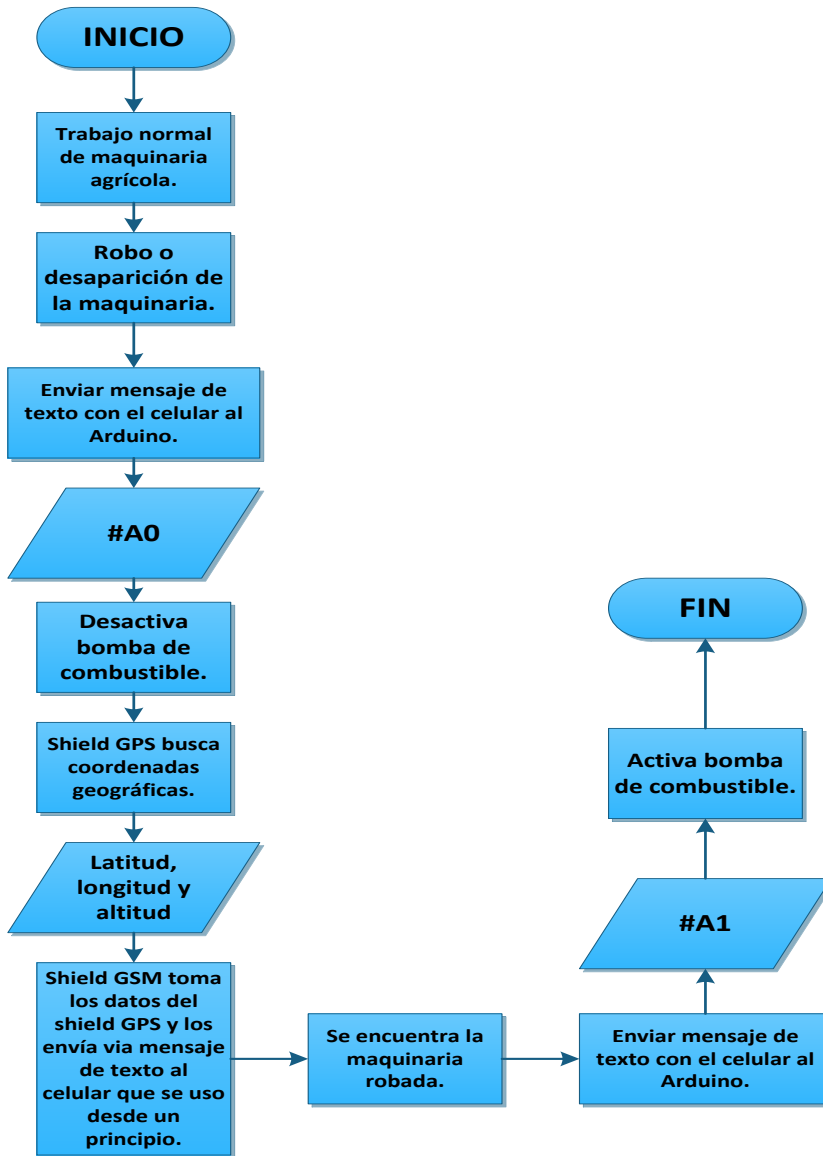


Figura 3.2 Descripción del proceso a realizar mediante un diagrama de flujo.

Basándose en el diagrama de flujo que tenemos, podemos hacer una clasificación para poder explicar el proceso de funcionamiento del sistema en desarrollo de una manera más concreta y entendible, para así obtener la siguiente clasificación:

- 3.5 Control de las salidas digitales del Arduino mediante un mensaje de texto SMS.
- 3.6 Circuito de activación y desactivación de una bomba de combustible.
- 3.7 Ubicación y localización de coordenadas geográficas mediante el shield GPS.
- 3.8 Envío de un mensaje mediante el shield GSM con las coordenadas geográficas.
- 3.9 Circuito de conmutación para la alimentación del sistema de localización y bloqueó de maquinaria agrícola.

3.5 Control de las salidas digitales del Arduino mediante un mensaje de texto SMS

Uno de los objetivos de este trabajo es lograr la desactivación de una bomba de combustible, para cortar el paso del flujo del diesel y así prevenir la pérdida total de la maquinaria agrícola.

Para esto es necesario desarrollar un circuito desactivador (del cual se explicara su diseño en el siguiente subtítulo) el cual funciona en compañía del Arduino y el shield GSM, los dos funcionan mediante el desarrollo de un código o sketch que se programa en el Arduino, para que así el circuito antes mencionado realice la función de desactivar la bomba de combustible.

Para lograr lo antes mencionado es necesario controlar las salidas digitales del Arduino, las cuales controlan, el encendido o apagado de algún dispositivo conectado a estas salidas digitales, ya sea mandando pulsos altos (5V) o pulsos bajos (0V), y con ayuda del shield GSM podemos lograr el control de funcionamiento del Arduino mediante un mensaje de texto SMS, para que este primer paso funcione de manera óptima es necesario conocer el diseño de conexión de hardware y el diseño del software.

3.5.1 Conexión de hardware

Para que el Arduino y el shield GSM funcionen se necesitaron una alimentación mínima de 5V la cual es suministrada por la entrada de alimentación del Arduino, la unión o conexión de estas dos placas es muy sencilla basta montar el shield GSM encima del Arduino y conectar los pines de salida y entrada digital y salida analógica.



Figura 3.3 Conexión del shield GSM con el Arduino

Ya teniendo conectado el Arduino con el shield GSM, es necesario tomar en cuenta los siguientes puntos, para el óptimo funcionamiento del shield GSM y el Arduino:

- Utilizar un teléfono para desactivar la solicitud de PIN de su tarjeta SIM (en caso de que este lo tenga).
- Utilizar una antena de celular de cuatro bandas con conexión estándar SMA, ya que la sonda que viene adherida al shield GSM no es una antena.



Figura 3.4 Antena para celular de cuatro bandas y conexión estándar SMA

- Si se está en un área de recepción débil, utilizar una antena externa como la que se utiliza en un vehículo de motor. Si está utilizando el vehículo más grande de estilo aéreo, puede encontrarse con que el enchufe no encaja en el conector del escudo. Por ejemplo, considere lo siguiente:

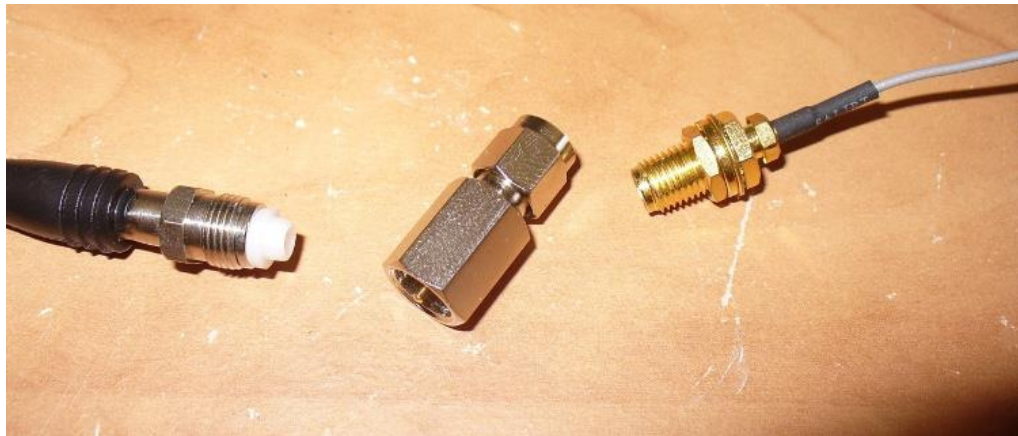


Figura 3.5 Adaptación de una antena de mayor potencia en el shield GSM

A la izquierda está el extremo del cable de la antena de Carphone, el derecho es el conductor del escudo de GSM. La solución está en el centro: un varón al adaptador FME SMA macho.

- La antena del shield GSM es muy fácil por lo cual es muy recomendable tener precaución al utilizar el shield, ya que esta solo viene protegida con una pequeña soldadura.

Teniendo ya esta parte de hardware bien armada descartamos ya los fallos de conexión, y se procede a la parte de software o de programación, para así ya poder realizar la función indicada a cabo.

3.5.2 Diseño de software

Para el uso del shield GSM es necesario contar con la librería NewSoftSerial (en el caso de usar una versión antecesora de Arduino 1.0) la cual se proporciona

una página web, ya teniendo este archivo sólo se necesita pegar la carpeta de NewSoftSerial dentro de la carpeta libraries del software Arduino.

El siguiente paso es el desarrollo del sketch para poder subirlo al Arduino y así poder realizar la función que deseamos hacer, en este caso como ya antes se mencionó la función a desarrollar es el control de las salidas digitales del Arduino.

Los comandos que más afectan en este programa son los siguientes:

- **AT+CMGF=1:** Establece el modo SMS al texto.
- **AT+CNMI=3,3,0,0:** Este comando le dice al módulo GSM para enviar de inmediato los nuevos datos de SMS a la salida serial.
- **AT+CMGD=1,4:** Esto elimina todos los mensajes de texto desde la tarjeta SIM. Como hay una cantidad finita de espacio de almacenamiento en la tarjeta SIM, es prudente eliminar el mensaje de entrada después de haber seguido las instrucciones en su interior.

Al tener conocimiento de estos comandos, se procede al desarrollo de un algoritmo donde se utilizan los comandos antes mencionados, que nos ayudaran a tener el control de los pines de salida del Arduino, para así también tener el control de algún otro artefacto.

Algoritmo y diagrama a flujo del programa

Algoritmo

Para el desarrollo de un sketch, es necesario tener claro qué es lo que vamos hacer, lo ideal es empezar explicando la función del sketch en base a un algoritmo:

1. Incluir la biblioteca NewSoftSerial “**#include <NewSoftSerial>**”.
2. Mantener nuestro carácter desde el puerto serie “**char inchar;**”.
3. Crear un puerto serie falso donde el pin 2 es Rx y el pin 3 es Tx “**NewSoftSerial cell(2,3)**”.

4. Meter la variable led1 en el pin 9 **"int led1=9"**.
5. Preparar el pin 9 en modo salida **"pinMode(led1, OUTPUT)"**.
6. Inicializamos el pin 9 en modo apagado **"digitalWrite(led1, LOW)"**.
7. Abrimos puerto serie del shield GSM y fijamos una velocidad de 9600 baudios **"cell.begin(9600)"**.
8. Damos un tiempo de 30 segundos de retardo para inicializar el módulo GSM **"delay(30000);"**.
9. Establecemos el módulo GSM en modo de texto **"cell.println("AT+CMGF=1")**.
10. Metemos los conjuntos de módulos para enviar los datos SMS a cabo en serie a la recepción **"cell.println("AT+CNMI=3,3,0,0")**.
11. Si entra un carácter al modulo celular **"if(cell.available(>0))"**.
 - ✓ #AX. **"if(inchar==#)"** , **"if(inchar==A)"**.
 - Si X=0 manda 0V pin9 **"digitalWrite(led1, LOW)"**.
 - Si X=1 manda 5V pin9 **"digitalWrite(led1, HIGH)"**.
12. Borrar el SMS **"AT+CMGD=1,4:"**.

Para el mejor entendimiento también es necesario desarrollar un diagrama de flujo, que se formará con base en al algoritmo anteriormente desarrollado.

Diagrama de flujo

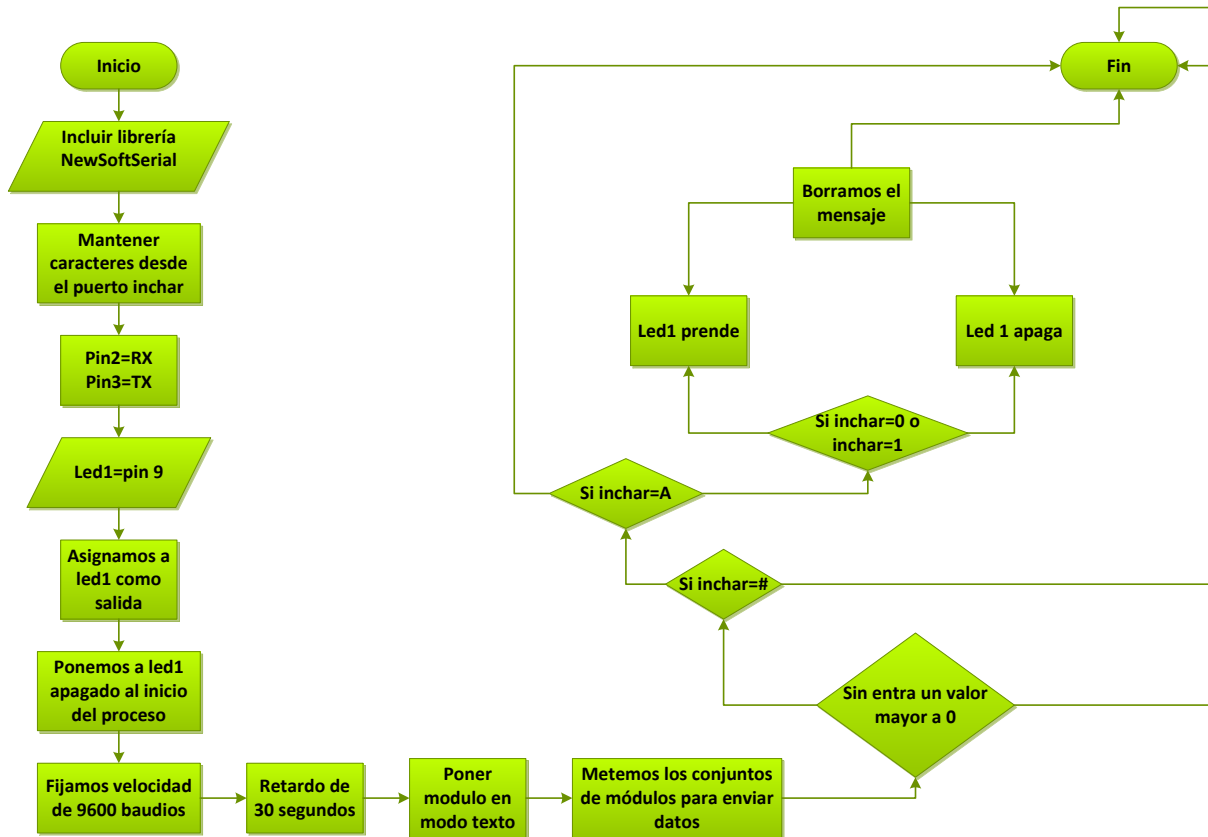


Figura 3.6 Diagrama de flujo del control de salidas digitales en base a un mensaje SMS

Ya desarrollado el diagrama de flujo que surge del algoritmo explicado anteriormente, es más fácil desarrollar el sketch, para así ser implementado en el hardware y realizar la función indicada.

Circuito de comprobación.

Para comprobar que el sketch generado anteriormente funciona, es necesario armar el Arduino de la siguiente manera:

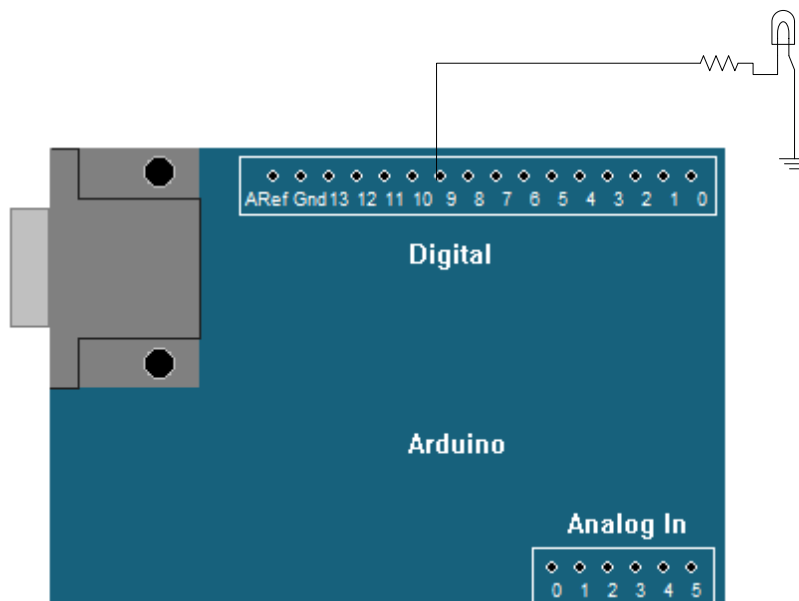


Figura 3.7 Circuito para la comprobación del control de salidas del Arduino mediante un mensaje SMS

Ya armado el Arduino como en la figura 3.7 y montado el sketch sobre este mismo, obtenemos que al enviarle un mensaje al Arduino con el código requerido, debe de prender o apagar el led que está conectado al pin 9, todo esto dependerá que si el código que enviamos es para activación o desactivación.

El led ésta conectado a una resistencia de 220 ohms para la protección del led que se encuentra conectado el Arduino, el cual está conectado a la salida digital 9 del Arduino, salida la cual fue programa para encender el led o apagarlo, según la orden generada en el mensaje de texto SMS.

Nota: En la fig 4. No se muestra el shield GSM, esto no muestra ningún inconveniente ya que al montar el shield este comparte las mismas salidas y entradas con el Arduino.

3.6 Circuito de activación y desactivación de una bomba de combustible.

Una de las partes más importantes de este proyecto, es el circuito de activación y desactivación de una bomba de combustible, la cual en base a un pulso mandado por el Arduino deberá desactivar o activar el funcionamiento de una bomba de combustible de diesel, según sea el caso.

Los materiales que se usaran para el armado de este circuito son los siguientes:

- 1 relevador de 12 V DC y 40 Amp.
- Optoacoplador MOC3010.
- Resistencia de 39 ohms.
- Diodo 1N4007.
- Batería de 12 V DC (batería del tractor).

El circuito se debe armar de la como se muestra en el siguiente diagrama eléctrico.

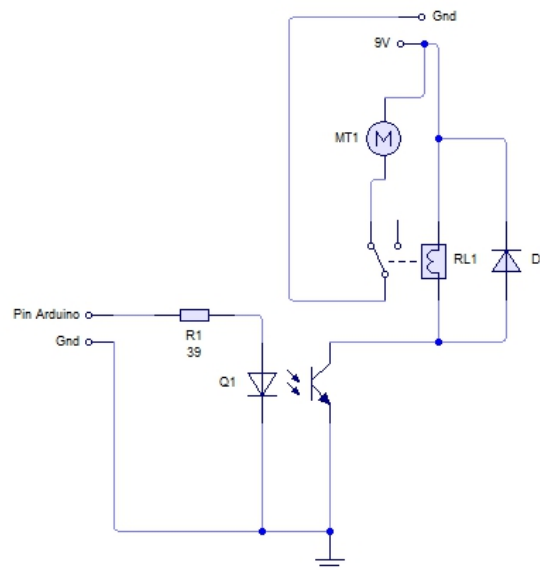


Figura 3.8 diagrama eléctrico de circuito activador-desactivador

El funcionamiento de este circuito es que en base a el pulso del pin de control que será el pin del Arduino que indique, si está en estado alto (5v) o en estado bajo (0V), se activara o se desactivara nuestra bomba de combustible, esto es posible a que el relevador del circuito funciona como un interruptor operado magnéticamente, esto significa que al momento de mandar el Arduino un pulso alto, el relevador desconectara la terminal a la cual está conectada la bomba de combustible, y cuando el pin del Arduino mande un pulso bajo la terminal de la bomba de combustible será activada de nuevo.

El optoacoplador tiene la función de proteger el Arduino, ya que la el relevador tiene una bobina la cual cuando trabaja en corriente directa, trabaja en modo de corto circuito, cosa que al paso del tiempo dañara al pin del Arduino. El optoacoplador al momento de recibir un estado alto por el pin del Arduino este prendera el diodo led interno que tiene, mandando la señal al transistor interno del optoacoplador el mandara una señal de 2 a 3 voltios con una corriente de 100 mAmp a la bobina del relevador.

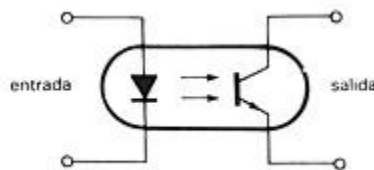


Figura 3.9 diagrama eléctrico de un optoacoplador indicando entradas y salidas

Entre el optoacoplador y el pin del Arduino se encuentra una resistencia esta resistencia sirve como protección de la corriente de salida que nos da el Arduino, para poder definir el valor de la resistencia, se debe calcular su valor con la siguiente formula:

$$R = \frac{V_t - V_r}{I_r}$$

Sustituyendo esta fórmula con los valores reales de los componentes obtenemos el siguiente resultado:

$$V_t = 5V$$

$$V_r = 1.5V$$

$$I_r = 100mAamp$$

$$R = \frac{5 - 1.5}{100 \times 10^{-3}} = 35 \text{ ohms}$$

El valor obtenido de la resistencia es de 35 ohms, lo que nos dice que en valores comerciales se puede usar una resistencia de 33 ohms o 39 ohms.

El diodo que va conectado a las terminales de la bobina sirve como protección de las altas corrientes que puedan llegar al momento del encendido del tractor, así se garantiza un óptimo funcionamiento del relevador y se le da más tiempo de funcionalidad a este.

3.7 Ubicación y localización de coordenadas geográficas mediante el shield GPS.

Para realizar la ubicación y localización de algún objeto, es necesario el uso del shield GPS, el cual está acompañado de un módulo GPS EM406A. Para el uso de este de este módulo GPS y su shield es necesario primero saber, su instalación con el Arduino, su funcionamiento, y el programa el cual hace la función de localización mediante coordenadas geográficas.

3.7.1 Instalación del shield GPS con el Arduino Uno

El modulo GPS EM406A es un módulo muy utilizado, pequeño, potente y con la antena integrad, consigue datos cada segundo, se alimenta con 5V y se comunica por serie. Es compatible con el sistema WAAS (sistema de aumentación basado en satélites) que permite incluso mayor precisión.

Para conectar el shield en el Arduino Uno, posee un microconector. Con el GPS shield, solo debemos enchufarlo y montarlo con el Arduino Uno. Para esta prueba

se configura los dos mini interruptores de la shield como ON y DLINE, esto mantendrá el modulo encendido y enviara los datos por los pines digitales 2 y 3.

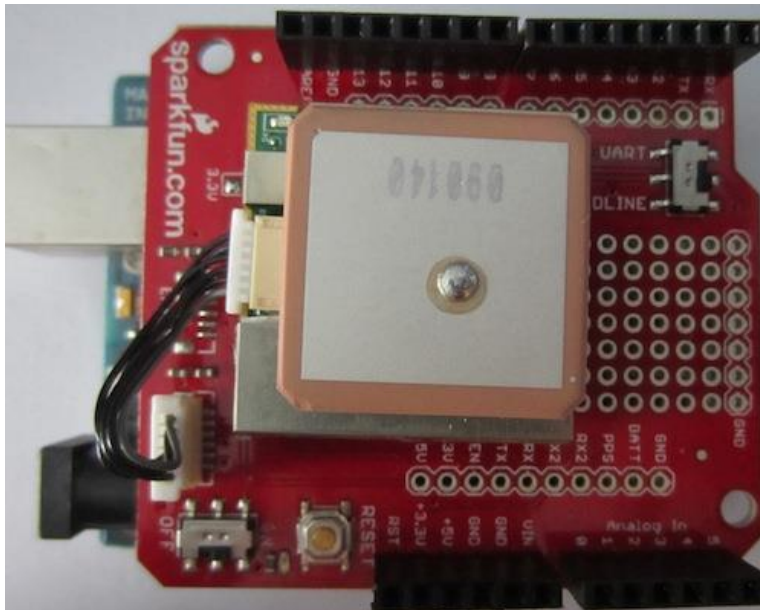


Figura 3.10 shield GPS montado en el Arduino con el modulo GPS EM406A

3.7.2 Programación para el modulo GPS

Es necesario incluir un par de librerías en el IDE del Arduino, estas dos librerías son la TinyGPS y la NewSoftSerial. Para incluir estas dos librerías es necesario que desde nuestra computadora descarguemos las dos bibliotecas, y ponerlas en la carpeta “libraries”.

Al ya tener montado el shield GPS con el modulo GPS en el Arduino, es necesario desarrollar un sketch que analiza las sentencias NMEA desde un módulo GPS EM406A funcionando a 4800 bps en lectura de valores de latitud, longitud, altitud, fecha, hora, curso y velocidad.

Como fue antes mencionando es necesario asegurarse de que el interruptor este en Dline. Se utiliza la biblioteca NewSoftserial, para la comunicación serial con el GPS, así se puede conectar el GPS TX y RX pines a cualquier pin digital en el Arduino.

Generación del sketch

Diagrama de flujo

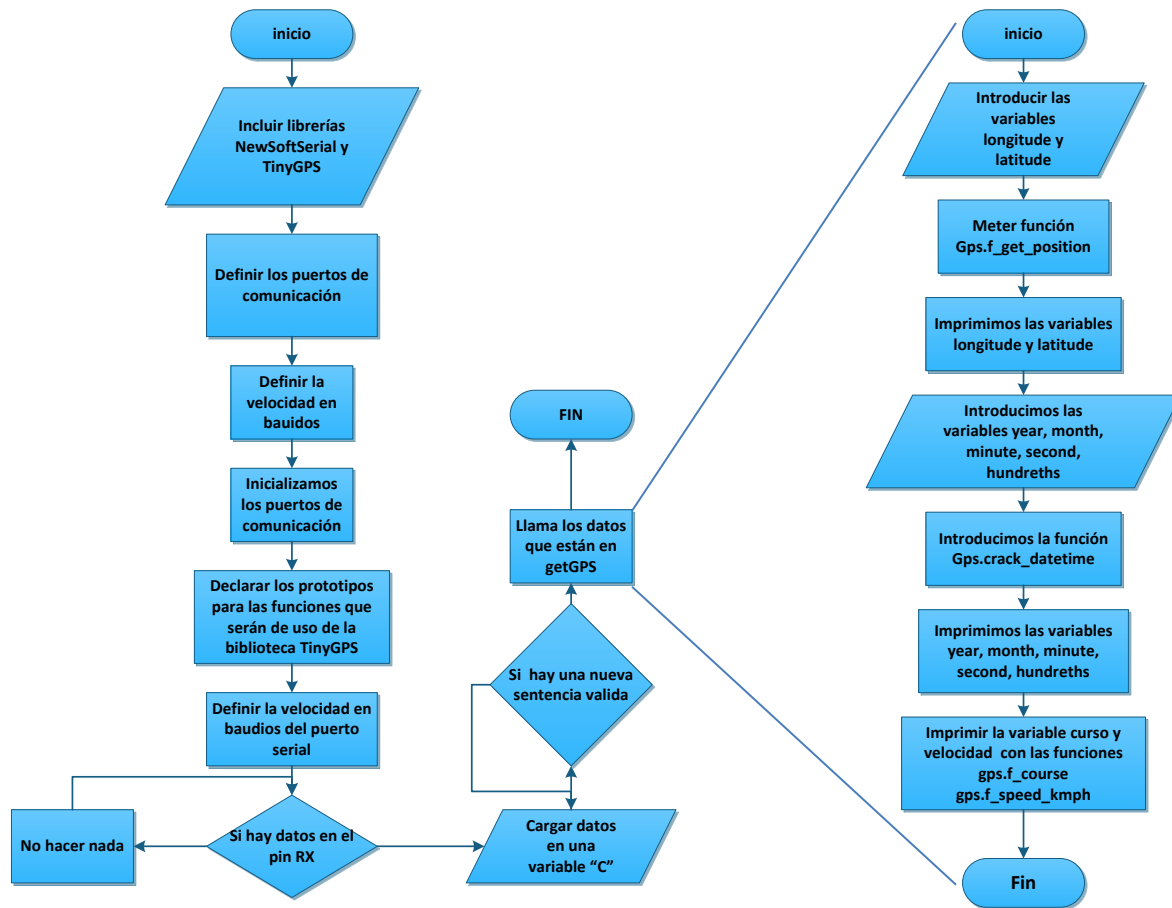


Figura 3.11 Diagrama de flujo para la obtención de coordenadas geográficas

Algoritmo

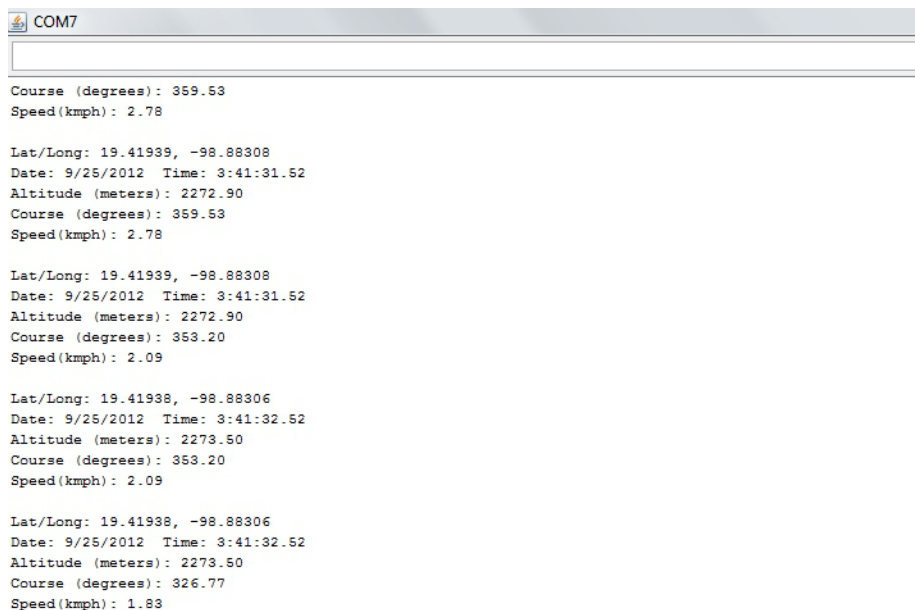
1. Incluir las bibliotecas:
 - **#include <NewSoftSerial.h>**
 - **#include <TinyGPS.h>**
2. Definir los pines de comunicación del Arduino y el GPS:
 - **#define RXPIN 2**
 - **#define TXPIN 3**
3. Establecer el valor de velocidad de transmisión del GPS:
 - **#define GPSBAUD 4800**

4. Crear una instancia del objeto TinyGPS:
 - **TinyGPS gps;**
5. Inicializamos la biblioteca NewSoftSerial a los pines que fueron definidos anteriormente:
 - **NewSoftSerial uart_gps(RXPIN, TXPIN)**
6. Declaramos los prototipos para las funciones que serán uso de la biblioteca TinyGPS:
 - **void getgps(TinyGPS &gps);**
7. Si hay datos en el pin RX
 - **while(uart_gps.available())**
8. cargamos los datos en una variable:
 - **int c = uart_gps.read();**
9. Si hay una nueva sentencia valida:
 - **if(gps.encode(c))**
10. Tomar los datos que hay en getgps(gps):
 - 10.1. Definir las variables de altitud y latitud:
 - **float latitude, longitude;**
 - 10.2. Introducir función para imprimir las variables:
 - **gps.f_get_position(&latitude, &longitude);**
 - **Serial.print("Lat/Long: ");**
 - **Serial.print(latitude,5);**
 - **Serial.print(", ");**
 - **Serial.println(longitude,5)**
 - 10.3. Introducir las variables de año, día, mes, hora, minutos, segundos y centésimas, velocidad y curso:
 - **int year;**
 - **byte month, day, hour, minute, second, hundredths;**
 - **gps.crack_datetime(&year,&month,&day,&hour,&minute, &second,&hundredths);**
 - **Serial.print("Date: "); Serial.print(month, DEC);**
 - **Serial.print("/");**

- **Serial.print(day, DEC); Serial.print("/"); Serial.print(year);**
- **Serial.print(" Time: "); Serial.print(hour, DEC);**
- **Serial.print(":");**
- **Serial.print(minute, DEC); Serial.print(":");**
- **Serial.print(second, DEC);**
- **Serial.print("."); Serial.println(hundredths, DEC);**
- **Serial.print("Altitude (meters): ");**
- **Serial.println(gps.f_altitude());**
- **Serial.print("Course (degrees): ");**
- **Serial.println(gps.f_course());**
- **Serial.print("Speed(kmph): ");**
- **Serial.println(gps.f_speed_kmph());**
- **Serial.println();**

11. Fin

Desarrollado el algoritmo y programar el Arduino con el sketch generado para realizar la función ya indicada, se procede para abrir al pantalla serial para obtener el siguiente resultado.



```

COM7
Course (degrees): 359.53
Speed(kmph): 2.78

Lat/Long: 19.41939, -98.88308
Date: 9/25/2012 Time: 3:41:31.52
Altitude (meters): 2272.90
Course (degrees): 359.53
Speed(kmph): 2.78

Lat/Long: 19.41939, -98.88308
Date: 9/25/2012 Time: 3:41:31.52
Altitude (meters): 2272.90
Course (degrees): 353.20
Speed(kmph): 2.09

Lat/Long: 19.41938, -98.88306
Date: 9/25/2012 Time: 3:41:32.52
Altitude (meters): 2273.50
Course (degrees): 353.20
Speed(kmph): 2.09

Lat/Long: 19.41938, -98.88306
Date: 9/25/2012 Time: 3:41:32.52
Altitude (meters): 2273.50
Course (degrees): 326.77
Speed(kmph): 1.83

```

Figura 3.12 pantalla serial que muestra las coordenadas geográficas

En la pantalla serial obtenemos los datos donde se esta localizado el modulo GPS esto es posible de comprobar al usar el siguiente link.

<http://maps.google.com/maps?q=19.41938,-98.88306>

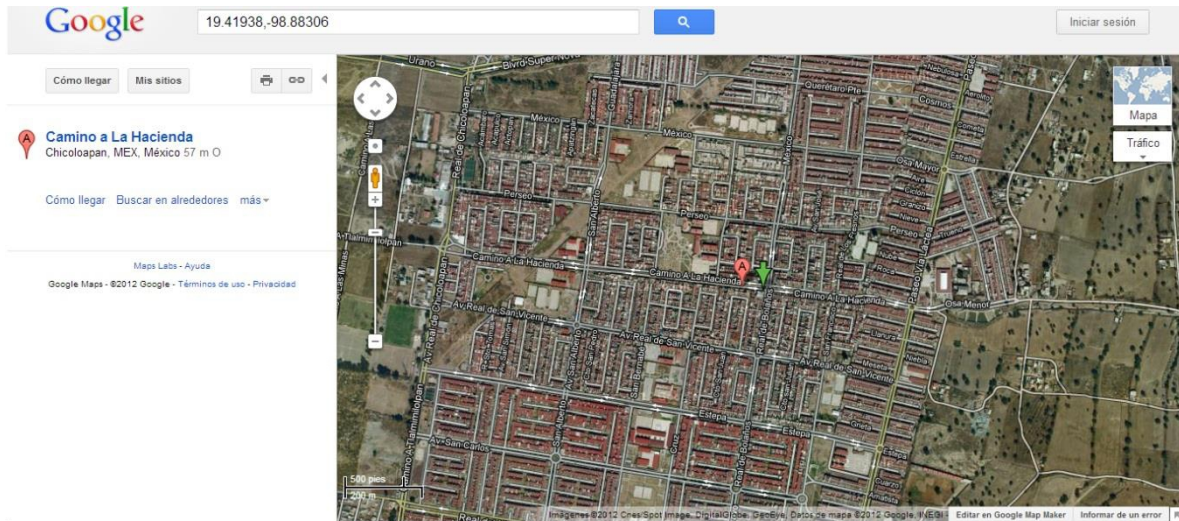


Figura 3.13 ubicación donde fue probado el modulo GPS

Sustituyendo la y última parte para así poder obtener el mapa la ubicación en donde estamos por medio de la página web antes mencionada.

3.8 Envío de un mensaje mediante el shield GSM con las coordenadas geográficas

Uno de las partes más importantes de este proyecto, es el envío de mensajes a un celular mediante el uso del Shield GSM acompañado de un shield GPS para que este nos proporcione la información necesaria para la localización de la maquinaria agrícola, para esto es indispensable analizar los siguientes puntos:

- Envío de un mensaje de texto SMS con el shield GSM
- Instalación del shield GPS en el Arduino.
- Instalación del shield GSM y GPS en el Arduino al mismo tiempo.

- Sketch para el envío de las coordenadas geográficas mediante un mensaje de texto SMS.

3.8.1 Envío de un mensaje de texto SMS con el shield GSM

Como primer punto es conveniente acabar de analizar y diseñar un programa, que nos muestre como es el envío de un mensaje SMS a un celular desde Arduino con la ayuda del shield GSM.

La instalación del software para el envío de un mensaje desde el Arduino con el shield GSM es exactamente el mismo que se vio en el subtema de “Control de las salidas digitales del Arduino con un mensaje SMS, lo que es más conveniente tomar como tema central es la generación del sketch, ya que mediante él se puede generar este propósito.

El comando AT nuevo que se ve en este sketch es el comando "AT+CMGS=", mediante el cual podemos meter el texto a un mensaje con un máximo de 160 caracteres.

Generación del sketch.

Algoritmo

1. Incluir la biblioteca NewSoftSerial **"#include <NewSoftSerial>"**.
2. Crear un puerto serie falso donde el pin 2 es Rx y el pin 3 es Tx **"NewSoftSerial cell(2,3)"**.
3. Meter variable con el número telefónico al cual enviar el mensaje "mobilenumber" **"char mobilenumber[]"="xxxxxxxxxx";"**.
4. Abrimos puerto serie del shield GSM y fijamos una velocidad de 9600 baudios **"cell.begin(9600)"**.
5. Damos un tiempo de 35 segundos de retardo para inicializar el módulo GSM **"delay(35000);"**.
6. Establecemos el módulo GSM en modo de texto **"cell.println("AT+CMGF=1")**.
7. Meter código para enviar mensaje **"cell.print("AT+CMGS=")"**.

8. Metemos el símbolo “ en código ASCII (34) “**cell.print(34,BYTE);**”.
9. Metemos la variable mobilenumber “**cell.print(mobilenumber);**”.
10. Metemos el símbolo “ en código ASCII (34) “**cell.print(34,BYTE);**”.
11. Introducimos el mensaje que queremos enviar.
12. Metemos ctrl-z en ASCII (26) “**cell.print(26,BYTE);**”.
13. Damos 15 segundos de retardo para volver a estado correcto “**delay(15000);**”.
14. Usar función do while para que el mensaje sea sólo enviado una vez.

Diagrama de flujo

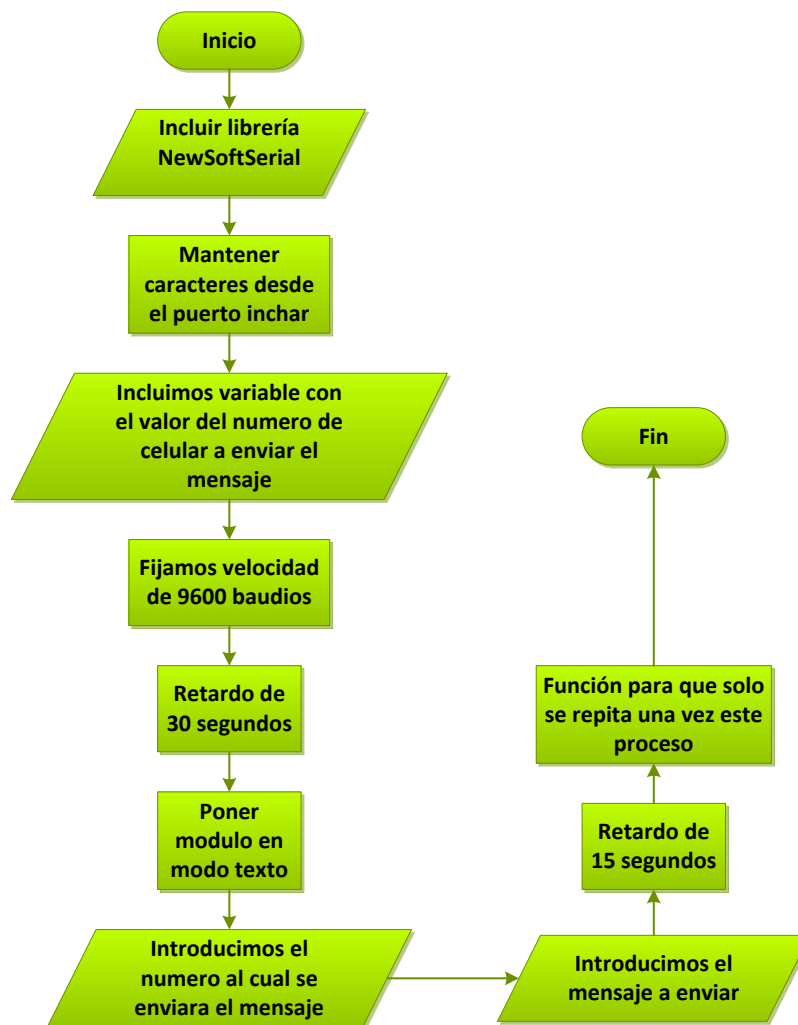


Figura 3.14 diagrama de flujo del envío de mensaje mediante Arduino

En base a que el algoritmo esta ya desarrollado y tenemos una mejor visión con el diagrama de flujo se ya es más simple ahora poder desarrollar el sketch.

Después de subir el programa al Arduino, se tiene que esperar un tiempo promedio de 35 a 50 segundos, y el Arduino generar un mensaje que recibirá el celular al cual fue destinado con el numero indicado en el sktech, el mensaje recibido también será controlado por el mensaje compuesto en el sketch.

3.9 Circuito de conmutación para la alimentación del sistema de localización y bloqueó de maquinaria agrícola

Es necesario que este sistema, tenga o cuente con al menos una fuente auxiliar de energía, es por eso que también es necesario un sistema de conmutación entre dos o más fuentes de energía.

El diseño del sistema de conmutación está representado en la siguiente figura:

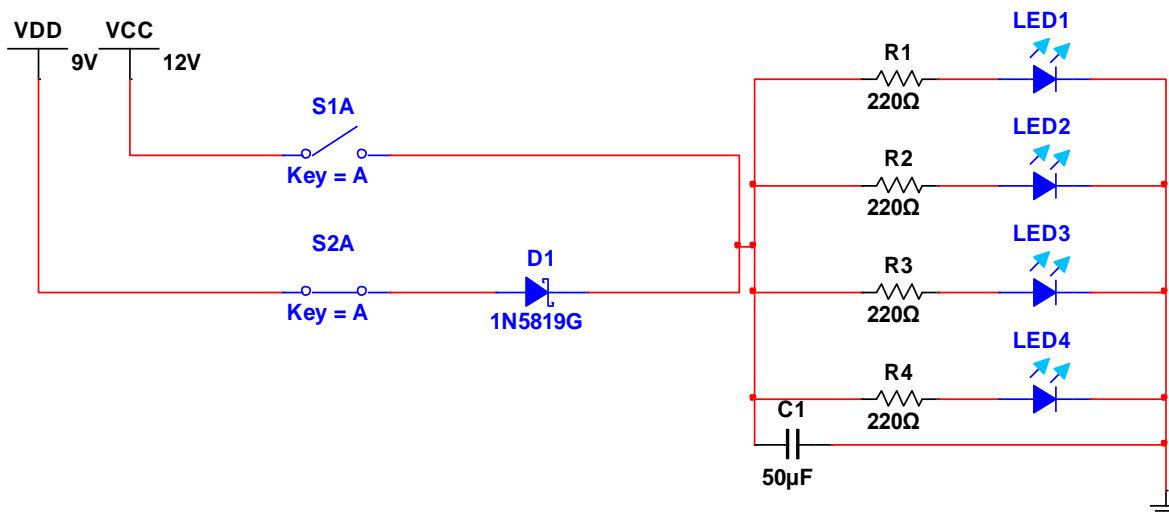


Figura 3.15 Circuito de conmutación

Nota: En la versión 1.0 de Arduino, las variables tipo BYTE han sido eliminadas, por lo que se deberá utilizar la versión 0023 o 0022 para poder utilizar el celular shield o buscar alguna librería softwareSerial que sea compatible con el IDE 1.0 de Arduino

El diodo 1N5819 es un diodo de barrera o también conocido como diodo Schottky, este diodo es usado por su rápida conmutación, el funcionamiento general de este circuito, consiste en que cuando la fuente de alimentación de 12V que es la fuente general, este conectada el diodo 1N5819 se polarice inversamente y no permita que la batería de 9V (fuente auxiliar) funcione, al momento de que sea desconectada la fuente, el diodo antes mencionado es activado y permite el paso de la fuente de alimentación de 9V, así previniendo que el sistema alimentado se apague y no funcione.

El capacitor que tenemos conectado a la salida de los diodos, es para quitar ruido del sistema y halla alguna variación de voltaje que provoque que el sistema se sobrecargue y se queme o haga un corto circuito.

Capítulo 4

Pruebas y resultados

Capítulo 4. Pruebas y resultados

Resumen

En este capítulo se presenta, el desarrollo de pruebas que se realizaron para que el sistema de localización alarma y bloqueo vía GSM/GPS, funcionara de manera óptima.

Las pruebas que se realizaron van desde lo más básico hasta lo más sofisticado, esto para entender tanto el funcionamiento interno (generación de sketch), hasta el cómo este prototipo facilita la interacción con el usuario, para su mejor uso, mostrando las pruebas que se llevaron con éxito y cuáles son esas condiciones de trabajo para que no exista ninguna falla.

4.1. Envió de un mensaje SMS con las coordenadas geográficas en base a un evento ocurrido

El primer paso que se debe de tomar en cuenta es el envío de un mensaje en base a un evento ocurrido, el primer paso para el desarrollo de este proyecto es realizar un programa que detecte el cambio de estado de un pin del Arduino (de estado bajo a estado alto), y en base a este cambio de estado active un pin encendiendo un led y mande un mensaje con las coordenadas geográficas.

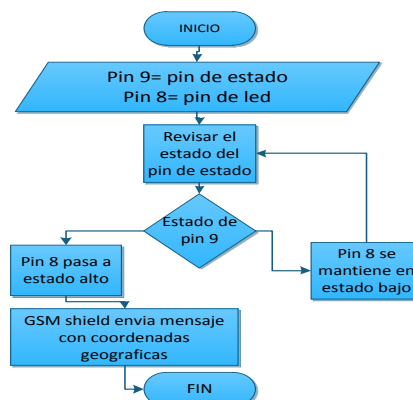


Figura 4.1. Diagrama de flujo del funcionamiento de la primera prueba

Este programa está hecho en base a los programas desarrollados en los subcapítulos 3.5 y 3.8 del capítulo 3, de los cuales se tomarán los métodos de programación para la activación de un pin y el envío de un mensaje con las coordenadas geográficas.

Los pasos que se deben seguir para el armado del programa son los siguientes:

- Lo primero que hay que hacer es la introducción de constantes en el programa para la identificación de las variables que vamos a usar.

```
char mobilenumber[] = "XXXXXXXXXXXX"; //Asigna un número telefónico con el cual nos comunicaremos.
int led1=8; //Asignamos el pin 8 como led1.
void getgps(TinyGPS &gps); //Declara prototipos para las funciones que serán uso de la biblioteca TinyGPS.
const int buttonPin = 4;
const int ledPin = 5;
int buttonState = 0;
```

Esta parte del programa es escrita al principio para que no tenga problemas el programa al momento de querer detectarlos.

- La creación de los puertos de comunicación es una de las partes más importantes del programa y del funcionamiento del circuito, ya que en base a ellos se evita que las entradas de comunicación de los shields.

```
NewSoftSerial uart_gps(0,1); // Asigna puertos series falsos para comunicación con el Shield GPS.
NewSoftSerial cell(2,3); //asigna puertos series falsos para comunicación con el shield Cellular 2=RX y 3=TX.
```

- Lo siguiente que hay que hacer es definir en qué modo funcionaran los pines y sus estados iniciales. Además de definir las velocidades de

comunicaciones en baudios para el funcionamiento del correcto del Arduino con los shields.

```
pinMode(ledPin, OUTPUT); //Ponemos a led1 como pin de salida
pinMode(buttonPin, INPUT); //Ponemos a led1 como pin de entrada
pinMode (led1, OUTPUT); //Ponemos a led1 como pin de salida.
digitalWrite(led1, HIGH); //Ponemos a led1 en estado alto como estado inicial.
cell.begin(9600); //establece la velocidad de comunicación en baudios para el
celullar shield.

Serial.begin(115200); //estable la velocidad de comunicación en baudios para el
puerto serie.

uart_gps.begin(4800); //establece la velocidad de comunicación en baudios para
el GPS Shield.
```

Esta parte es escrita en la sección de setup que inicializa el modo de trabajo de los pines o el puerto serie.

- Ya en la parte de bucle o loop definiremos bien el funcionamiento del programa. La primero que hay que definir es el estado del pin de entrada esto lo hacemos con el siguiente comando.

```
buttonState = digitalRead(buttonPin);
if (buttonState == HIGH)
```

Ahora también hay que especificar las condiciones dependiendo el estado del pin 9. Si está en estado alto debe de encender un led que está conectado al pin 8 y además mandar un mensaje con las coordenadas geográficas.

```
digitalWrite(ledPin, HIGH); //pone al led en modo alto.
while(uart_gps.available()) //si el GPS detecta algo.
{
int c = uart_gps.read(); //inserta lo leído por el GPS en la constante c.
if(gps.encode(c)) //si hay una nuvea sentencia valida.{
    getgps(gps); //agarrar los datos
```

La parte de getgps(gps) esta explicada de una mejor manera en el capítulo 3 en el apartado 3.8 envío de un mensaje mediante el shield GSM con las coordenadas geográficas. Cuando el pin 9 este en estado bajo el programa debe de mantener apagado el led conectado al pin 8 y no mandar ningún tipo de mensaje

Ya realizados estos pasos basta con cargar el sketch en el Arduino con los shields GPS y GSM en el Arduino, y los pines conectados como se muestra en la siguientes figura 4.4 y así ya poder probar el funcionamiento del programa echo.

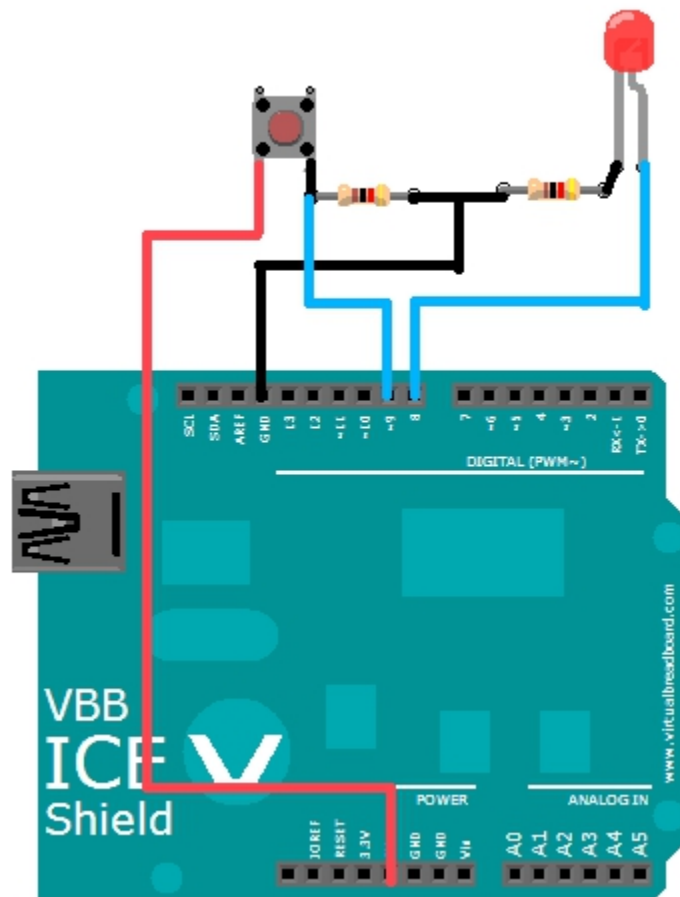


Figura 4.2. Diseño de circuito de activación en base a un evento

Las resistencias deben tener valores menores a los 300 ohms ya que solo sirven como protección y la corriente que se maneja en este circuito no es tan alta, así que no requiere una protección mayor.



Figura 4.3. Muestras del mensaje enviado por el celular

Los resultados obtenidos son, que debe de prender el led conectado en el pin 8, y debe recibir un mensaje el celular con las coordenadas geográficas del lugar en donde se encuentra el dispositivo, el celular al cual fue asignado el programa para enviar el mensaje.

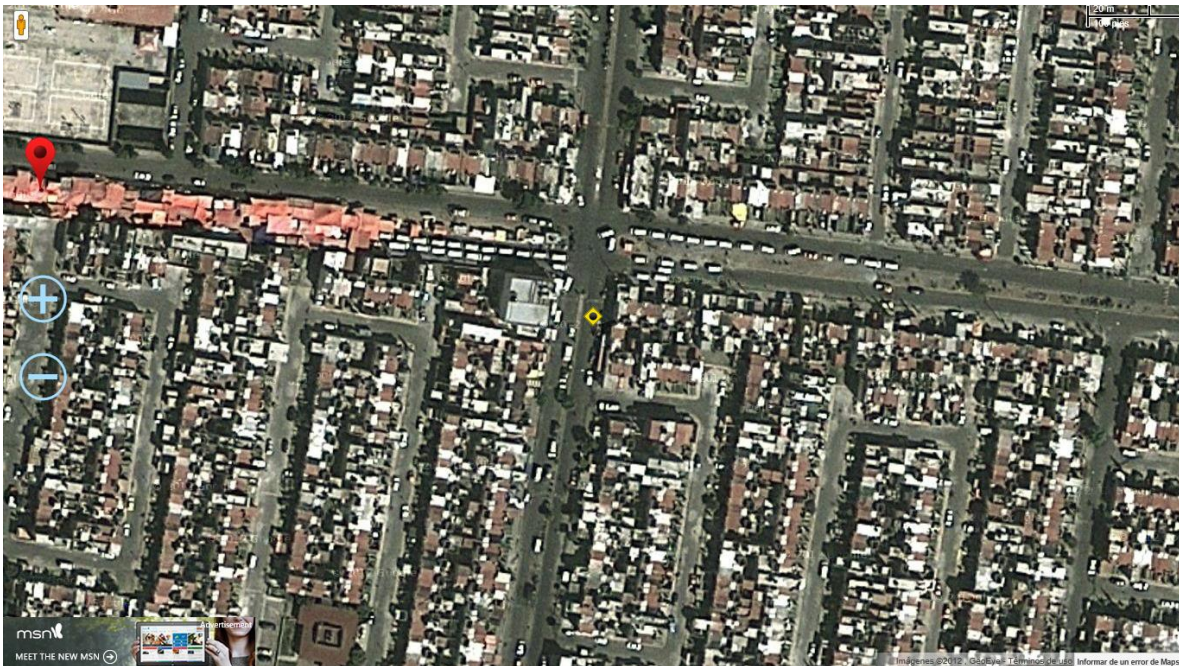


Figura 4.4. Mapa que muestra la localización del lugar en donde está el dispositivo situado

Como comprobación del mensaje podemos adherir que el mensaje llegue en forma de link, para que este sea copiado y así poder comprobar en algún buscador de internet el lugar donde el dispositivo está situado, esto se observa en la figura 4.5.

4.2. Funcionamiento y problemas del shield GPS

El shield GPS es una herramienta muy útil para lo que son proyectos escolares, tiene una amplia gama de localización a nivel mundial, y su velocidad de respuesta es muy rápida. En este apartado se revisaran las pruebas de inicio de cómo trabajar con el shield GPS solo con el Arduino y las recomendaciones que se deben seguir al momento de trabajar en conjunto del shield GSM.

La primera prueba consiste en probar su funcionamiento en si solo con el Arduino para ver la localización del dispositivo con coordenadas geográficas. El programa de prueba para esta prueba está localizado en la biblioteca TinyGPS, localizada en el IDE de Arduino.

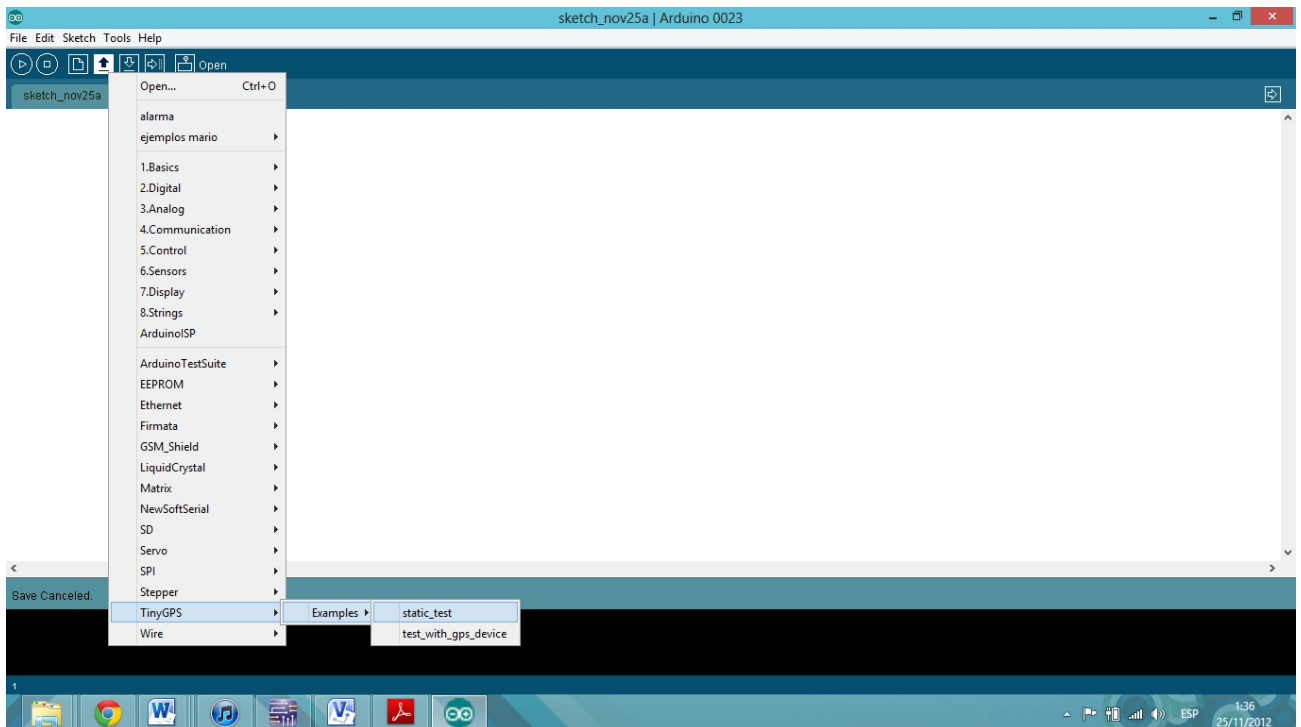


Figura 4.5. Localización del sketch para la prueba del GPS

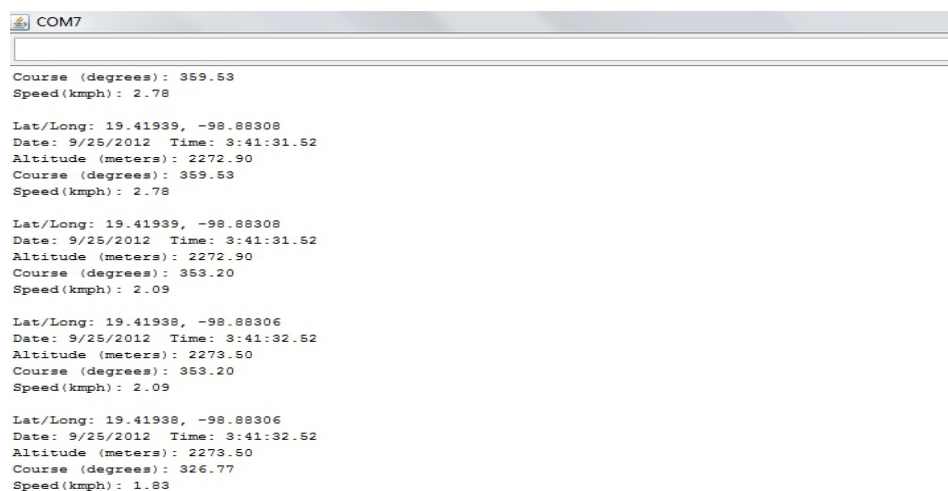
Al cargar este programa en el Arduino es necesario que el shield GPS no este montado en el Arduino o en el caso de que este montado que este esté apagado, recordando que este shield tiene un switch de encendido y apagado.

Las condiciones de hardware para que este programa funcione; son las siguientes:

- El shield debe estar al aire libre ya que en condiciones de estar cubierto por algún techo grueso, ya que en esas condiciones tarda mucho en detectar la señal de los satélites. El modo en el que se puede saber si el shield GPS está detectando la señal es al ver el led en el EM406A parpadear, en ese momento nos está indicando que el shield ya detecto la señal de los satélites y es posible que nos entregue la posición.
- El shield GPS contiene un switch que cambia el modo de comunicación de este con el Arduino este switch cambia de modo UART (comunicación vía serial) y el modo DLINE (comunicación vía pines digitales) para trabajar el programa de muestra se trabaja con los pines 2 y 3, el pin 2 como RX y el pin 3 como TX, pero también puede trabajar como modo UART puede funcionar el programa solo basta con cambiar los pines 2 y 3 a 1 y 0 como se muestra en el siguiente ejemplo:

```
NewSoftSerial uart_gps(2,3); //modo DLINE  
NewSoftSerial uart_gps(0,1); //modo UART
```

Al cumplir con estas condiciones podemos ver en la ventana serial del IDE de Arduino lo siguiente:



```
COM7  
  
Course (degrees): 359.53  
Speed(kmph): 2.78  
  
Lat/Long: 19.41939, -98.88308  
Date: 9/25/2012 Time: 3:41:31.52  
Altitude (meters): 2272.90  
Course (degrees): 359.53  
Speed(kmph): 2.78  
  
Lat/Long: 19.41939, -98.88308  
Date: 9/25/2012 Time: 3:41:31.52  
Altitude (meters): 2272.90  
Course (degrees): 353.20  
Speed(kmph): 2.09  
  
Lat/Long: 19.41938, -98.88306  
Date: 9/25/2012 Time: 3:41:32.52  
Altitude (meters): 2273.50  
Course (degrees): 353.20  
Speed(kmph): 2.09  
  
Lat/Long: 19.41938, -98.88306  
Date: 9/25/2012 Time: 3:41:32.52  
Altitude (meters): 2273.50  
Course (degrees): 326.77  
Speed(kmph): 1.83
```

Figura 4.6. Respuesta enviada por la pantalla serial

El segundo modo de trabajo es en compañía del shield GSM, en esta prueba los dos shield estarán montados sobre el Arduino. El modo de trabajo es casi el mismo solamente que el shield GPS debe de trabajar en modo UART. El programa se explica en el capítulo 3 en el apartado 3.8 y en este mismo capítulo en el apartado anterior.

Las opciones de la información que queremos que nos entregue el shield GPS dependen mucho de la programación, esta parte la podemos controlar con el siguiente código:

```
float latitude, longitude;
    gps.f_get_position(&latitude, &longitude);
    Serial.print("Lat/Long: ");
    Serial.print(latitude,5);
    Serial.print(", ");
    Serial.println(longitude,5);
    int year;
    byte month, day, hour, minute, second, hundredths;
gps.crack_datetime(&year,&month,&day,&hour,&minute,&second,&hundredths);
    Serial.print("Date: "); Serial.print(month, DEC); Serial.print("/");
    Serial.print(day, DEC); Serial.print("/"); Serial.print(year);
    Serial.print("  Time: "); Serial.print(hour, DEC); Serial.print(":");
    Serial.print(minute, DEC); Serial.print(":"); Serial.print(second, DEC);
    Serial.print("."); Serial.println(hundredths, DEC);
    Serial.print("Altitude (meters): "); Serial.println(gps.f_altitude());
    Serial.print("Course (degrees): "); Serial.println(gps.f_course());
    Serial.print("Speed(kmph): "); Serial.println(gps.f_speed_kmph());
    Serial.println();
    unsigned long chars;
    unsigned short sentences, failed_checksum;
    gps.stats(&chars, &sentences, &failed_checksum);
```

Este código nos muestra desde longitud, latitud, velocidad hasta fecha, y solo basta con quitar alguna opción para que no se vean en la pantalla serial. En el caso de que se quiera mostrar lo del código mostrado anteriormente en la pantalla del celular vía mensaje basta con cambiar lo siguiente:

```
Serial.println();
```

Por lo siguiente:

```
cell.println();
```

Esto hará que lo que se ve en nuestra pantalla serial también salga en la pantalla del celular vía mensaje SMS.



Figura 4.7. Muestra de mensaje recibido por el shield GSM en un celular

En la figura 4.7 se mira el ejemplo de un mensaje enviado por el Arduino con ayuda de los shield GSM y GPS marcando las coordenadas geográficas.

4.3. Desarrollo y pruebas del programa general.

Con el desarrollo de las pruebas anteriores y el armado de los circuitos mencionados en capítulos anteriores, se llega como resultado final a la construcción del siguiente prototipo que consiste en lo siguiente:

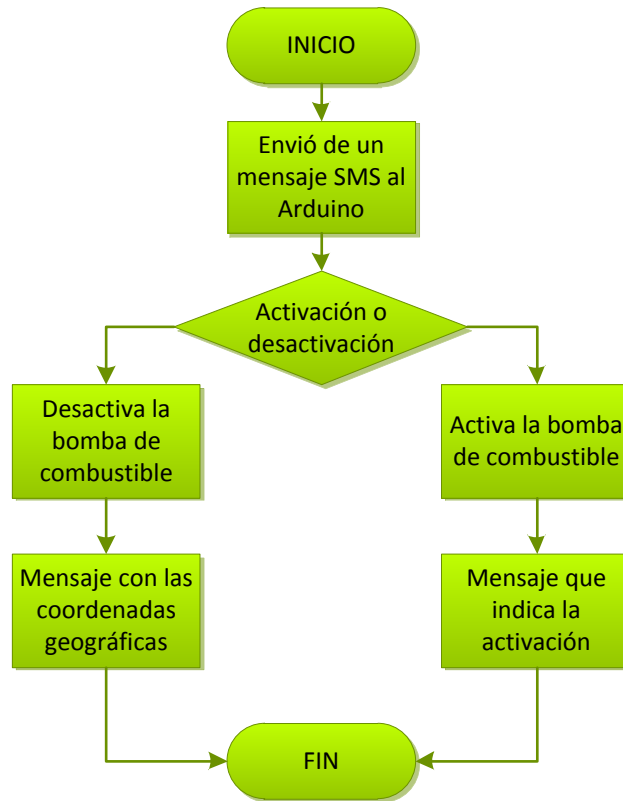


Figura 4.8. Diagrama de flujo general de funcionamiento del programa final

En esencia el sketch desarrollado para esta última prueba es muy semejante al mencionado en el apartado 4.1, solamente que el cambio que ahora se le hará que el evento ocurrido no será un cambio de estado de un pin, si no que ahora el evento ocurrido será la entrada de un mensaje al Arduino, esto será detectado por el shield GSM, esta parte se la vemos reflejada en el programa en la siguiente parte.

```

if(cell.available()>0)
{
  inchar=cell.read();
  if(inchar=='#')
  {
    delay(10);
    inchar=cell.read();
    if(inchar=='A')
  
```

```

{
  delay(10);
  inchar=cell.read();
  if (inchar=='0')
  {
    digitalWrite(led1,LOW);
    cell.print("BOMBA DE COMBUSTIBLE ACTIVADA");
  }
  else if (inchar=='1')
  {
    digitalWrite(led1, HIGH);
    startSMS();
    cell.print("LATITUD=XXX-LONGITUD=XXXX");
    endSMS();
    do
    {

```

Como lo vemos en el código tenemos una sentencia “If” que nos indica que si el shield GPS detecta un mensaje, realizara las siguientes funciones:

1. Detectar el primer carácter del mensaje entrante el cual será #.
2. Después detectara un segundo carácter en este caso será A.
3. Ya detectado el segundo carácter, ahora deberá detectar un carácter que definirá la función que realice el Arduino.
 - Si recibe un uno el Arduino encenderá el pin del led, y además mandara un mensaje con las coordenadas geográficas detectadas por el shield GPS.
 - Si recibe un cero el Arduino mantendrá el pin del el led en estado bajo o apagara este pin en caso de que se encuentre encendido, y mandara un mensaje de activación.

Este pin que se prende y apaga, o que pasa de estado bajo a alto y viceversa es el pin que estará conectado al circuito de activación; que se muestra en la siguiente figura:

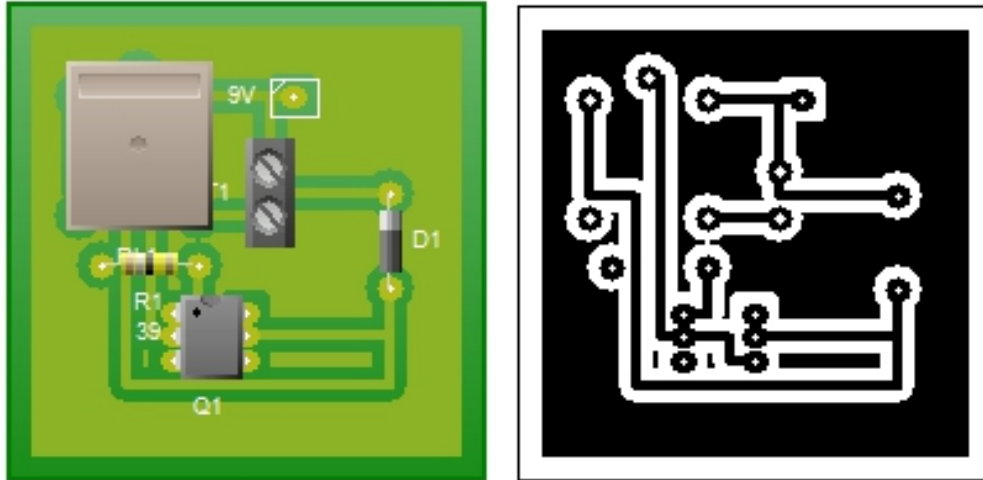


Figura 4.9. Circuito activador desactivador y pistas del mismo circuito

Al momento que el pin de activación y desactivación pasa a estado alto, manda una señal que hace que el relevador conectado a este pin, cambie de contacto y abra el circuito que está conectado con la bomba de combustible, lo que hace que la bomba de combustible deje de recibir alimentación y esto hace que no funcione y no permita el flujo de diesel. Pero en el momento que el pin está en estado bajo, el conector del relevador cierra el circuito, y permite que la bomba de combustible sea alimentada con los respectivos 12V.

Otra parte muy importante que mencionar es que si el shield GPS no detecta señal el sistema implementado no mandara ningún mensaje, este mensaje será realizado hasta que el shield GPS detecte alguna señal y pueda otorgarnos las coordenadas, en pocas palabras lo antes explicado se puede resumir en que si el GPS no detecta alguna señal la bomba de combustible será desactivada, pero no obtendremos algún mensaje de regreso a el celular deseado.

Lo mencionado anteriormente queda ejemplificado en el siguiente código:

```

while(uart_gps.available())
{
  int c = uart_gps.read();
  if(gps.encode(c))
  {
    getgps(gps);
  }
}

```

Otro factor de pruebas también es la parte de alimentación del Arduino y sus demás componentes, ya que este sistema no estará ya conectado a una computadora que forme parte de su alimentación, para esto el Arduino cuenta con una entrada de alimentación tipo Jack 5.5x2.1mm.



Figura 4.10. Conector Alimentación Jack 5.5x2.1mm Arduino

Este conector ira conectado a una pila de 9V la cual alimentara al Arduino y sus demás componentes, pera pila de 9 voltios será la fuente de alimentación auxiliar ya que la fuente de alimentación general será la batería del tractor en donde esté instalado el Arduino, como se muestra en la figura 3.15, pero en lugar de ir conectada a los leds este estará conectado al Arduino, es necesario decir que en la parte de alimentación de la batería es recomendable adaptarle un regulador 7805 o 7809 ya que este regulador bajara el volatje de entrada de 12V a 5V o 9V según sea el caso del regulador utilizado.

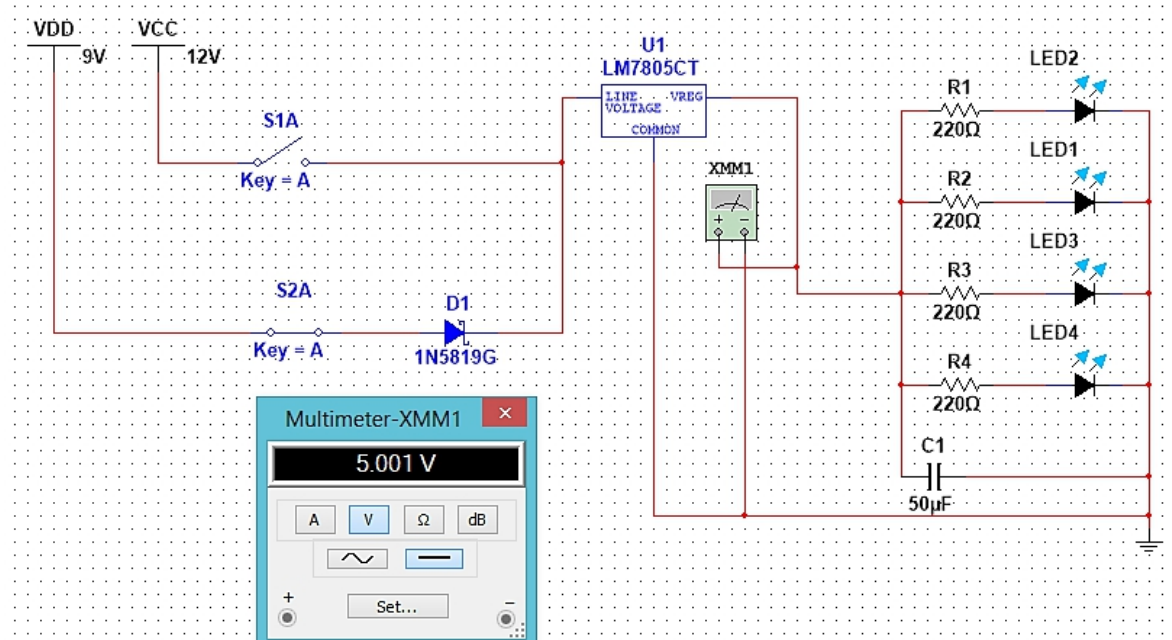


Figura 4.11. Circuito de conmutación con el regulador 7805

Como se observa en la figura 4.11 el circuito tiene ya adaptado el regulador de voltaje 7805 el cual podemos ver en el multímetro que nos entrega un voltaje de 5V aproximadamente.

Cumpliendo lo antes mencionado, y lo mencionado en los capítulos anteriores para el desarrollo del sistema de localización alarma y bloqueo de maquinaria agrícola vía GSM/GPS, se puede alcanzar el óptimo funcionamiento de este proyecto, así cumpliendo los requerimientos en el objetivo marcado al principio de este mismo trabajo.

Conclusiones

Conclusiones

Los objetivos marcados en el inicio de este trabajo fueron alcanzados, al desarrollo de este proyecto, pudiendo hacer una aplicación que va desde un software hasta la combinación de una serie de dispositivos, pueden llevar acabo la función de un sistema de localización alarma y bloqueo, teniendo como condición el buen manejo en el a creación del sketch con el que será programado el Arduino, y el dominio de las fuentes de alimentación para evitar un sobrecalentamiento, corto circuito o mal funcionamiento de este dispositivo.

El sistema mínimo conocido como Arduino que en compañía de los shields GSM y GPS, puede llegar a ser un dispositivo muy cómodo de trabajo, ya que solo hay que tener en cuenta la parte de la via de comunicación serial para así poder explotar estos tres elementos, al máximo. Ya que la parte de software es muy cómoda de trabajar, además de que cuentan con salidas que se prestan mucho al armado de diferentes tipos de proyectos. Concluyendo que Arduino es uno de los sistemas que se presta mejor para el desarrollo de proyectos tanto escolares como de un nivel más alto.

El único inconveniente que se presentó en este trabajo es que el GPS shield no detecta señales dentro de cuartos cerrados ya que las paredes son gruesas y no permiten que se detecte la señal para que el GPS funcione bien, lo que nos lleva como conclusión buscar otro sistema GPS que sea compatible con el sistema Arduino para así poder superar este único inconveniente.

Apéndice

Apéndice A. Glosario

Arduino: es una plataforma de hardware libre, basada en una placa con un microcontrolador y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica en proyectos multidisciplinarios.

Baudios: es una medida de la cantidad de ancho de banda con que cuenta una transmisión.

Comandos AT: los comandos AT son instrucciones codificadas que conforman un lenguaje de comunicación entre el hombre y un terminal modem

Entrada y salida digital: son entradas que funcionan a base de 1 y 0, lo que nos quiere decir que 1 funciona como un nivel alto y 0 como un nivel bajo.

Estación base: estación de transmisión y recepción situada en un lugar fijo, compuesta de una o más antenas de recepción/transmisión, una antena de microondas, y un conjunto de circuitos electrónicos, y utilizada para manejar el tráfico telefónico.

GitHub: es una forja para alojar proyectos utilizando el sistema de control de versiones Git. Utiliza el framework Ruby on Rails por GitHub, Inc.(anteriormente conocida como Logical Awesome).

Latitud: la latitud mide el ángulo entre cualquier punto y el ecuador. Las líneas de latitud se llaman paralelos y son círculos paralelos al ecuador en la superficie de la Tierra. La latitud es la distancia que existe entre un punto cualquiera y el Ecuador, medida sobre el meridiano que pasa por dicho punto.

Longitud: la longitud mide el ángulo a lo largo del ecuador desde cualquier punto de la Tierra. Se acepta que Greenwich en Londres es la longitud 0 en la mayoría de las sociedades modernas. Las líneas de longitud son círculos máximos que pasan por los polos y se llaman meridianos.

Loop(): la función loop() hace precisamente lo que sugiere su nombre, se ejecuta de forma cíclica, lo que posibilita que el programa este respondiendo continuamente ante los eventos que se produzcan en la tarjeta

Memoria EEPROM: es un tipo de memoria ROM que puede ser programada, borrada y reprogramada eléctricamente, a diferencia de la EPROM que ha de borrarse mediante un aparato que emite rayos ultravioleta. Son memorias no volátiles.

Memoria flash: es un tipo de memoria informática basada en semiconductores, no volátil y re escribible Esto significa que posee muchas de las características de la memoria RAM, excepto que sus datos no se eliminan al apagarse el ordenador. La memoria Flash almacena porciones de datos en las celdas de memoria, pero esos datos permanecen almacenados aunque se produzca un corte de energía.

MSC (Mobile Switching Centre, o Centro de Conmutación de Servicios Móviles): es un elemento de las redes de comunicaciones móviles GSM que tiene como función interconectar usuarios de la red fija con la red móvil, o usuarios de la red móvil entre sí. Al mismo tiempo mantiene las bases de datos para tratar las peticiones de llamada de los clientes.

Pin Rx: pin de recepción (entrada de datos).

Pin Tx: pin de transmisión (salida de datos).

Regulador de tensión: es un dispositivo electrónico diseñado para mantener un nivel de voltaje constante.

Setup(): la función setup() se invoca una sola vez cuando el programa empieza. Se utiliza para inicializar los modos de trabajo de los pins, o el puerto serie. Debe ser incluido en un programa aunque no haya declaración que ejecutar.

SMS: son las siglas de Servicio de Mensaje Corto. Disponible en redes digitales GSM permitiendo enviar y recibir mensajes de texto de hasta 160 caracteres a teléfonos móviles vía el centro de mensajes de un operador de red.

Variables: Una variable es una manera de nombrar y almacenar un valor numérico para su uso posterior por el programa. Como su nombre indica, las variables son números que se pueden variar continuamente en contra de lo que ocurre con las constantes cuyo valor nunca cambia.

Apéndice B. Instrucciones en C++ para la programación de Arduino

Estructura de un programa

La estructura básica del lenguaje de programación de Arduino es bastante simple y se compone de al menos dos partes. Estas dos partes necesarias, o funciones, encierran bloques que contienen declaraciones, estamentos o instrucciones.

```
Void setup()
{
    Estamentos;
}

Void loop()
{
    Estamentos;
```

Setup()

La función setup() se invoca una sola vez cuando el programa empieza. Se utiliza para inicializar los modos de trabajo de los pins, o el puerto serie. Debe ser incluido en un programa aunque no haya declaración que ejecutar.

```
Void setup()
{
    pinMode(pin, INPUT); //configura el pin como entrada
}
```

Loop()

Después de llamar a setup(), la función loop() hace precisamente lo que sugiere su nombre, se ejecuta de forma cíclica, lo que posibilita que el programa este respondiendo continuamente ante los eventos que se produzcan en la tarjeta.

```
Void loop()
{
  digitalWrite(pin, HIGH); //pone en uno (on, 5v) el ´pin´
  delay(1000); //espera un segundo (1000 ms)

  digitalWrite(pin, LOW); //pone en cero (off, 0v.) el
´pin´

  delay(1000);
}
```

Funciones

Una función es un bloque de código que tiene un nombre y un conjunto de estamentos que son ejecutados cuando se llama a la función. Son funciones setup() y loop() de las que ya se ha hablado. Las funciones de usuario pueden ser escritas para realizar tareas repetitivas y para reducir el tamaño de un programa. Las funciones se declaran asociadas a un tipo de valor “type”. Este valor será el que devolverá la función, por ejemplo 'int' se utilizará cuando la función devuelva un dato numérico de tipo entero. Si la función no devuelve ningún valor entonces se colocará delante la palabra “void”, que significa “función vacía”. Después de declarar el tipo de dato que devuelve la función se debe escribir el nombre de la función y entre paréntesis se escribirán, si es necesario, los parámetros que se deben pasar a la función para que se ejecute.

```
type nombreFuncion(parámetros)
```

```
{
```

```
Estamentos;
```

```
}
```

{ } Entre llaves

Las llaves sirven para definir el principio y el final de un bloque de instrucciones. Se utilizan para los bloques de programación `setup()`, `loop()`, `if...`, etc.

```
type funcion ()
```

```
{
```

```
Estamentos;
```

```
}
```

Una llave de apertura “{” siempre debe ir seguida de una llave de cierre “}”, si no es así el programa dará errores.

; Punto y coma

El punto y coma “;” se utiliza para separar instrucciones en el lenguaje de programación de Arduino. También se utiliza para separar elementos en una instrucción de tipo “bucle for”.

```
int x = 13; //declara la variable 'x' como tipo entero de valor 13
```

Declaración de variables

Todas las variables tienen que declararse antes de que puedan ser utilizadas. Para declarar una variable se comienza por definir su tipo como `int` (entero), `long` (largo), `float` (coma flotante), etc, asignándoles siempre un nombre, y, opcionalmente, un valor inicial. Esto sólo debe hacerse una vez en un programa,

pero el valor se puede cambiar en cualquier momento usando aritmética y reasignaciones diversas.

El siguiente ejemplo declara la variable `entradaVariable` como una variable de tipo entero “int”, y asignándole un valor inicial igual a cero. Esto se llama una asignación.

```
Int entradaVariable=0;
```

Byte

Byte almacena un valor numérico de 8 bits sin decimales. Tienen un rango entre 0 y 255.

```
Byte unaVariable = 150; //declara 'unaVariable' como tipo byte
```

Int

Enteros son un tipo de datos primarios que almacenan valores numéricos de 16 bits sin decimales comprendidos en el rango 32,767 to -32,768.

```
Int unaVariable = 1600; //declara 'unaVariable' como una variable  
tipo entero
```

Long

El formato de variable numérica de tipo extendido “long” se refiere a números enteros (tipo 32 bits) sin decimales que se encuentran dentro del rango -2147483648 a 2147483647.

```
long unaVariable = 90000; // declara 'unaVariable' como tipo long
```

float

El formato de dato del tipo “punto flotante” “float” se aplica a los números con decimales. Los números de punto flotante tienen una mayor resolución que los de 32 bits con un rango comprendido 3.4028235E +38 a +38-3.4028235E.

```
float unaVariable = 3.14; // declara 'unaVariable' como tipo
flotante
```

Operadores de comparación

```
x == y // x es igual a y
x != y // x no es igual a y
x < y // x es menor que y
x > y // x es mayor que y
```

digitalRead(pin)

Lee el valor de un pin (definido como digital) dando un resultado HIGH (alto) o LOW (bajo). El pin se puede especificar ya sea como una variable o una constante (0-13).

```
valor = digitalRead(Pin); // hace que 'valor' sea igual al estado
leído en 'Pin'
```

digitalWrite(pin)

Envía al 'pin' definido previamente como OUTPUT el valor HIGH o LOW (poniendo en 1 o 0 la salida). El pin se puede especificar ya sea como una variable o como una constante (0-13).

```
digitalWrite(pin, HIGH); // deposita en el 'pin' un valor HIGH
(alto o 1)
```

Serial.println(data, datatype)

Vuelca o envía un número o una cadena de caracteres al puerto serie, seguido de un carácter de retorno de carro "CR" (ASCII 13, or '\r') y un carácter de salto de línea "LF"(ASCII 10, or '\n'). Toma la misma forma que el comando Serial.print()

```
Serial.println(b) vuelca o envía el valor de b como un número
decimal en caracteres ASCII seguido de "CR" y "LF".
```


`Serial.println(b, DEC)` vuelca o envía el valor de `b` como un número decimal en caracteres ASCII seguido de "CR" y "LF".

`Serial.println(b, HEX)` vuelca o envía el valor de `b` como un número hexadecimal en caracteres ASCII seguido de "CR" y "LF".

`Serial.println(b, OCT)` vuelca o envía el valor de `b` como un número Octal en caracteres ASCII seguido de "CR" y "LF".

`Serial.println(b, BIN)` vuelca o envía el valor de `b` como un número binario en caracteres ASCII seguido de "CR" y "LF".

`Serial.print(b, BYTE)` vuelca o envía el valor de `b` como un byte seguido de "CR" y "LF".

`Serial.println(str)` vuelca o envía la cadena de caracteres como una cadena ASCII seguido de "CR" y "LF".

`Serial.println()` sólo vuelca o envía "CR" y "LF". Equivaldría a `println()`.

Serial.Read()

Lee o captura un byte (un caracter) desde el puerto serie. Equivaldría a la función `serialRead()`.

Biblioteca NewSoftSerial

NewSoftSerial es la última de las tres bibliotecas que prestan Arduino para apoyo del puerto serie. Es el descendiente directo de ladyada's AFSoftSerial, que introdujo por interrupciones una importante mejora de la votación requerida por el nativo de SoftwareSerial.

Sin interrupciones, el diseño de su programa está bastante restringido, ya que continuamente debe sondear el puerto serie en muy breves intervalos regulares.

Esto hace que sea casi imposible que, por ejemplo, para utilizar SoftwareSerial para recibir los datos del GPS y analizarlo en una forma utilizable. Su programa está demasiado ocupado tratando de mantenerse al día con caracteres de NMEA a medida que llegan a pasar realmente tiempo de montaje en algo significativo. Aquí es donde (y de NewSoftSerial), la arquitectura de interrupción AFSoftSerial es un regalo del cielo. Usando por interrupciones RX, el programa se llena el buffer detrás de las escenas durante el procesamiento de los datos previamente recibidos.

Mejoras

NewSoftSerial ofrece una serie de mejoras con respecto a SoftwareSerial:

- Se hereda de built-in de impresión de clase, la eliminación de algunos 4-600 bytes de código duplicado
- Se implementa sistema de almacenamiento en búfer circular para hacer más eficiente el procesamiento de RX
- Se extiende el apoyo a todos los pines de Arduino 0-19 (0-21 en el Arduino Mini), no sólo 0-13
- Es compatible con varios dispositivos en serie simultáneos suaves. *
- Es compatible con una gama mucho más amplia de velocidades de transmisión. **
- Proporciona un desbordamiento de boolean () método para detectar desbordamiento de búfer.
- Mayores velocidades de transmisión han sido afinados para una mayor precisión.
- Es compatible con el ATmega328 y 168.
- Es compatible con los procesadores de 8MHz.
- Se utiliza el puerto E / S directa de la operación más rápida y más precisa.
- (Nueva con la versión 10). Es compatible con la inversión de señal de software.
- (Nuevo) Es compatible con los procesadores de 20 Mhz.

- (Nuevo) Se ejecuta en la Teensy y Teensy ++.
- (Nuevo) Es compatible con el método `fin ()` como un complemento para `empezar ()`.

* Sin embargo, ver más abajo para hacer una advertencia importante en varias instancias.

** Ser prudentes sobre el uso de 300 y 1200 baudios sin embargo. El manejador de interrupciones en estas tasa llega a ser tan largo que las interrupciones del temporizador de garrapatas pueden ser muerto de hambre, causando `millis ()` para dejar de trabajar durante la recibe.

Uso de varias instancias

Ha habido un apoyo considerable para una biblioteca que permita a múltiples dispositivos serie suaves. Sin embargo, el manejo de datos de forma asíncrona recibidas de dos, tres, o cuatro o más dispositivos de serie resulta ser un problema extremadamente difícil, si no es intratable. Imagínese cuatro dispositivos serie conectados a un Arduino, cada uno transmite a 38.400 baudios. Como los bits llegan, pobre procesador de poca Arduino debe muestrear y procesar cada uno de los 4 bits de entrada dentro de los 26 microsegundos o de lo contrario perderá para siempre.

Se me ocurrió, sin embargo, que varias instancias aún podría ser posible si el usuario de la biblioteca estaba dispuesto a hacer una pequeña concesión. `NewSoftSerial` está escrito en el principio de que usted puede tener tantos dispositivos conectados como las limitaciones de recursos lo permitan, siempre y cuando sólo vas a usar uno de ellos a la vez. Si usted puede organizar el código de programa en torno a esta restricción, entonces `NewSoftSerial` puede trabajar para usted.

¿Qué significa esto exactamente? Bueno, usted tiene que utilizar sus dispositivos de serie en serie, de esta manera:

```
#include <NewSoftSerial.h>
```

```

// Esto es un dispositivo GPS conectarse a los pines 3 y 4
NewSoftSerial gps(4,3);

// Un termómetro serie conectado a 5 y 6
NewSoftSerial therm(6,5);

// Una pantalla LCD conectado a 7 y 8
NewSoftSerial LCD(8,7); // serial LCD

void loop()
{
    ...
    // Recoger los datos de la unidad GPS durante unos
segundos
    gps.listen();
    read_gps_data(); // Usar el GPS como dispositivo activo
    // Recopilar datos de temperatura del termómetro
    therm.listen();
    read_thermometer_data(); // Ahora use therm
    // LCD se convierte en el dispositivo activo aquí
    LCD.listen();
    LCD.print("Data gathered...");
    ...
}

```

En este ejemplo, se supone que `read_gps_data()` utiliza el objeto de GPS y `read_thermometer_data()` utiliza el objeto `therm`. Cada vez que usted llame a la escucha (`listen()`) método, se convierte en el "activo" el objeto y el objeto anteriormente activo se desactiva y se desecha el buffer RX. Un punto importante aquí es que `object.available()` siempre devuelve 0 a menos que objeto ya está activo. Esto significa que no se puede escribir código como este:

```

void loop()
{
    device1.listen();
    if (device1.available() > 0)
    {
        int c = device1.read();
        ...
    }
}

```

```
device2.listen();
if (device2.available() > 0)
{
    int c = device2.read();
    ...
}
}
```

Este código no se hace otra cosa que activar un dispositivo después de la otra.

Inversión de señal.

"Normal" la señalización de serie TTL define un bit de inicio como una transición de "alto" a "bajo" la lógica. Lógica 1 es "alta", 0 es "baja". Sin embargo, algunos dispositivos serie activar esta lógica al revés, con lo que se llama "señalización invertida". A partir de la versión 10, NewSoftSerial compatible con estos dispositivos de forma nativa con un tercer parámetro en el constructor.

```
NewSoftSerial myInvertedConn(7, 5, true); // Este dispositivo
utiliza una señalización invertida.
NewSoftSerial myGPS(3, 2); // esto no se hace.
```

Versión de la biblioteca

Puede obtener la versión de la biblioteca NewSoftSerial llamando al miembro estático.

```
library_version().

int ver = NewSoftSerial::library_version();
```

Referencias

Referencias

- Sin autor “Incrementa el robo de autos en todo el país”
<http://www.noticias.autoplaza.com.mx/nota-11514--incrementa-el-robo-de-autos-en-todo-el-pais>
Consultado el 12 de febrero del 2012.
- CNN “Los 10 autos mas robado en Mexico”
<http://www.cnnexpansion.com/mi-dinero/2011/08/23/los-10-autos-mas-robados-en-mexico>
Consultado el 13 de febrero del 2012.
- Sin autor “Seguridad ciudadana en América latina un bien público cada vez más escaso”
<http://www.pensamientoiberoamericano.org/articulos/0/26/0/seguridad-ciudadana-en-america-latina-un-bien-publico-cada-vez-mas-escaso.html>
Consultado el 13 de abril del 2012.
- José Antonio E. García Álvarez “así funciona el GPS”
http://www.asifunciona.com/electronica/af_gps/af_gps_10.htm
Consultado el 6 de abril del 2012
- [T. A. Herring, “The Global Positioning System”, *Scientific American*, 1996, pág. 32-38.
J.A. Fernández Rubio, G.Seco Granados, “Sistemas de posicionamiento: de GPS a GNSS”, *Mundo Electrónico*, 1997, 280, pp.46- 52.
G. J. Sonnenberg, *The Global Positioning System, Radar and Electronic Navigation*, Butterworths, 1988.
Understanding GPS: Principles and Applications, Editor Elliot D. Kaplan, Artech House, 1996.
R. Ware, S. Businger, “Global Positioning Finds Application in Geosciences Research”, Universidad Navstar Consortium, Boulder, Universidad de Hawaii, Honolulu, 1995.
http://www.cs.cmu.edu/afs/cs/project/lri-13/www/atacamatrek/mad_4_kids/kids_instruments.html
www.wikipedia.com/dpf-7/GPS/course/2html
www.verychanel.com/gps/edu/resources.html
<http://html.rincondelvago.com/sistemas-gps.html>
Consultado el 6 de abril del 2012.

- Bruj0 de la 7a69 EziNe " Tecnología GSM"<http://www.todo-cel.com.ar/info/gsm.html>.
Consultado el 7 de abril del 2012.
- Sin autor "Estándar GSM (Sistema global de comunicaciones móviles)"
<http://es.kioskea.net/contents/telephonie-mobile/gsm.php3>
Consultado el 7 de abril del 2007.
- Sin autor "Arduino"
<http://www.arduino.cc/es/>
Consultado el 7 de abril del 2012.
- Sin autor "Arduino UNO"
<http://arduino.cc/en/Main/arduinoBoardUno>
Consultado el 7 de abril del 2012.
- Sin autor "Tarjeta SIM"
<http://ingeniatic.net/index.php/tecnologias/item/612-tarjeta-sim>
Consultado el 8 de abril del 2012.
- Sin autor "implementación del modulo GSM-GPS"
<http://dSPACE.upse.edu.ec/bitstream/123456789/70/8/Capitulo2.pdf>
Consultado el 29 de marzo del 2012.
- Sin autor "Nano phone: Enviando SMS con arduino"
<http://electronicavm.wordpress.com/2011/12/21/nanophone-enviando-sms-con-arduino/>
Consultado el 1 de abril del 2012.
- Sin autor "proyecto alarma de seguridad para autos"
<http://www.slideshare.net/pedrobriones/proyecto-alarma-de-seguridad-para-autos>
Consultado el 1 de abril del 2012.
- Luis M. serrano "Alarma GPS".
<http://perso.wanadoo.es/luism..serrano/AlarmaGPS/ManGPSAG1.pdf>
Consultado el 2 de abril del 2012.
- Sin autor "Shield celular (SM5100B) para Arduino"
www.5hz-electronica.com/shieldcelularsm5100bparaarduino.aspx
Consultado el 22 de mayo del 2012.
- Sin autor "Arduino Shield GPS"

<http://www.sparkfun.com/products/9487>

Consultado el 22 de mayo del 2012.

- Sin autor “GitHub”

<http://es.wikipedia.org/wiki/GitHub>

Consultado el 24 de noviembre del 2012

- Sin autor “bomba electrica de combustible”

<http://www.mecanicafacil.info/mecanica.php?id=bombaElectricaCombustible>

Consultado el 24 de noviembre del 2012

- Sin autor “definición de baudios”

<http://jamesronco11.tripod.com/baudios.htm>

Consultado el 24 de noviembre del 2012

- Biblioteca técnica Nosso “bomba de combustible”

http://www.nosso.com.ar/spanish/tech_topics/bombas_comb01.php

Consultado el 24 de noviembre del 2012

- GREAT PLAINS INDUSTRIES, INC., Wichita, KS “bomba de combustible eléctrica de alto rendimiento”

<http://catalog.gpi.net/Asset/921928-07-M-3025-3425-Espanol.pdf>

Consultado el 24 de noviembre del 2012

- Sin autor “latitud y longitud”

<http://www.educaplus.org/play-280-Latitud-y-longitud.html>

Consultado el 25 de noviembre del 2012

- Sin autor “estación base”

<http://ec.europa.eu/health/opinions2/es/campos>

[electromagneticos/glosario/def/estacionbase.htm](http://ec.europa.eu/health/opinions2/es/campos/electromagneticos/glosario/def/estacionbase.htm)

Consultado el 25 de noviembre del 2012

- Sin autor “MSC”

<http://es.wikitel.info/wiki/MSC>

Consultado el 25 de noviembre del 2012

- Sin autor “memoria FLASH”

<http://es.kioskea.net/contents/pc/cartes-memoire-flash.php3>

Consultado el 25 de noviembre del 2012

Referencia de imágenes.

- <http://cibergeek.com/rastrea-un-celular-telefono-movil/> (celular)
Consultado el 12 de abril.
- <http://www.cuentocuentos.net/images/colorear/dibujos/Tractor-03.jpg>
(tractor)
Consultado el 12 de abril.
- <http://satelitecancho.blogspot.mx/> (satélite)
Consultado el 12 de abril.
- <http://www.sparkfun.com/products/9607> (shield GSM)
Consultado el 12 de abril y 22 de mayo.
- <http://www.sparkfun.com/products/9487> (shield GPS)
Consultado el 12 de abril y 22 de mayo.