UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

“POWER BIT”

DESARROLLO DE UN CASO PRÁCTICO

P R E S E N T A:

ALONSO ZÚÑIGA AARÓN DANIEL

ASESOR: MAESTRO. JUAN GASTALDI PÉREZ

MÉXICO 2019

INDICE

INTRODUCCIÓN

Hoy en día en la sociedad en la que vivimos está pasando por varios acontecimientos de inseguridad (robos, secuestros, asesinatos, etc.) Esto supone un gran problema para las familias que habitamos en ella, sabemos que esto ya tiene mucho tiempo afectándonos motivo por el cual se tratará de dar solución a alguno de estos problemas implementando la tecnología que tenemos disponible.

Las instituciones de seguridad con las que cuenta el gobierno de México se han visto envueltas en varios temas de corrupción, otro factor es causado por el abuso de las autoridades que están encargas de nuestra seguridad por lo cual no es sorpresa que los ciudadanos tengan cada vez menos confianza en dichas instituciones.

Estos factores son síntomas que padecen cada una de estas debido a las condiciones creadas desde el poder político para que sus intereses no se vean afectados, así que el problema de inseguridad no solo está en las calles por la cuales transcurrimos diariamente si no que el problema viene desde los poderes políticos.

Uno de los principales problemas que en gran medida nos afecta es el robo de vehículos, ya que esto supone un gran riesgo para la vida si uno como persona se resiste al asalto, nunca se sabe cómo reaccionará el asaltante tampoco se puede saber si esta armado, de tal manera que nuestra mejor opción es dejar que se lleve el vehículo antes que nuestra vida.

2017 fue el año record en el robo de vehículos en México al tener un índice mayor a los últimos 6 años según el reporte de la Asociación Mexicana de Instituciones de seguros (AMIS).

Este proyecto llamado POWER BIT está enfocado en una posible solución para el robo de vehículos.

Es un dispositivo que se puede esconder dentro del vehículo, la fuente de energía de este es la misma alimentación del vehículo, sus funciones principales son las siguientes:

1. Conectarse a una red de telefonía celular.
2. Activar una alarma atreves de un teléfono.
3. Apagar el vehículo después de cierto tiempo atreves de un comando.
4. Mandar un SMS a un teléfono ya programado con una liga de Google Maps de la ubicación del vehículo.
5. Tiene un inhibidor para el asaltante (gas pimienta, humo).

El último punto aún está en una fase de prueba para no dañar a terceros. Cuando un delincuente roba un vehículo generalmente también se llevan el teléfono de la víctima, el dispositivo está configurado para que reciba el comando desde cualquier celular, y así poder mandar la ubicación al número o números ya pres programados o bien inhabilitar el vehículo activando la alarma e iniciando el inhibidor.

Propósito:

El propósito de este proyecto es combatir el robo de vehículos implementando las tecnologías y conocimientos que tenemos disponibles para dar solución esta situación

Objetivo:

El objetivo de este proyecto es reducir el robo de vehículos

Por eso hemos decidido poner a la mano de los ciudadanos un aparato de seguridad que dependerá de ellos. En este documento se detallará el desarrollo del proyecto de seguridad llamado “POWER BIT “

Este dispositivo será descrito en 5 etapas las cuales serán.

CAPITULO 1: SEÑAL GPS

Se describirá el funcionamiento de el modulo NEO-6M UBLOX, hablaremos de sus principales características, comunicación con y su principal aplicación con el proyecto POWER BIT.

CAPITULO 2: USO DE LA RED DE TELEFONIA CELULAR

En este capítulo se describirá el funcionamiento del módulo SIM800L y su aplicación con Arduino describiendo a su vez el reconocimiento de los SMS.

CAPITULO 3: ALARMA

Se programa el funcionamiento de la alarma en este caso usaremos una bocina de seguridad.

CAPITULO 4: INHABILITAR VEHÍCULO

Se describe brevemente como se diseñó el circuito para parar el vehículo
Se describe también una parte del código para el funcionamiento del inhibidor.

CAPITULO 5: SMS DE LOCALIZACIÓN

Se pone en funcionamiento todos los módulos ya mencionados en los capítulos anteriores explicando el SMS de localización.

CAPITULO 6: ANALISIS FINANCIERO

Describiremos los costos e inversiones totales de cada uno de los componentes que se utilizaron en la creación del dispositivo

CAPITULO 1: SEÑAL GPS

1.1 Conociendo el modulo GPS UBLOX NEO6MV2

Es un módulo que está basado en el chip receptor NEO 6M de U-BLOX este módulo puede ser controlado por Arduino o cualquier otro micro controlador que pueda manejar dicho chip, interfaz de comunicación que maneja (Transmisor Receptor Asíncrono universal (UART), Interfaz Periférica Serial (SPI), i2c).

EL módulo GPS será una parte importante del proyecto ya que este se encargara de obtener las coordenadas de nuestro vehículo

IMAGEN 1.1 GPS UBLOX6M

CARACTERISTICAS

"La familia de receptores GPS NEO-6 están diseñados para tener un pequeño tamaño, pequeño coste, y pequeño consumo. La intensidad de corriente necesaria es de unos 37mA en modo de medición continuo.

La tensión de alimentación es de 2.7 a 3.6V para los modelos NEO-6Q/6M, y 1.75-2.0V para los modelos NEO-6G.

Los datos obtenidos del NEO-6 están en formato $GPRMC, uno de las secuencias disponibles en el protocolo NMEA (National Marine Electronics Association)

La secuencia tiene la forma

*$GPRMC,hhmmss.ss,A,llll.ll,a,yyyyy.yy,a,x.x,x.x,ddmmyy,x.x,a*hh"*[1]

[11] ***(Consulta echa el 21/08/2018*** **https://www.luisllamas.es/localizacion-gps-con-arduino-y-los-modulos-gps-neo-6/**)

1.2Aplicación en Arduino

Dado que la mayoría de los módulos que se ocuparon para este proyecto son compatibles con Arduino el entorno de desarrollo que se implemento fue ARDUINO IDE 1.8.8 Donde se programó el funcionamiento de cada uno de los módulos.

El dispositivo GPS NEO6M como ya vimos usa el método de comunicación UART (Receptores/transmisores asíncronos universales) el GPS se comunica usando el Protocolo serial, UART transmite los datos en la línea “TX” y recibe los datos por la línea “RX”.

IMAGEN 1.2. MODULO GPS NEO6M FUENTE :https://bit.ly/2N8QmbW

```
#include <SoftwareSerial.h>
#include <TinyGPS.h>
#include<string.h>
```

Lo primero que se debe hacer para poder usar el modulo GPS Neo 6m con Arduino es importar la librería TinyGPS.h

1.3 Ejemplo de aplicación

```
#include <SoftwareSerial.h>
#include <TinyGPS.h>

TinyGPS gps;
SoftwareSerial softSerial(4, 3);

void setup()
{
   Serial.begin(115200);
   softSerial.begin(9600);
}

void loop()
{
   bool newData = false;
   unsigned long chars;
   unsigned short sentences, failed;

   // Intentar recibir secuencia durante un segundo
   for (unsigned long start = millis(); millis() - start < 1000;)
   {
      while (softSerial.available())
      {
         char c = softSerial.read();
         if (gps.encode(c)) // Nueva secuencia recibida
            newData = true;
      }
   }

   if (newData)
   {
      float flat, flon;
      unsigned long age;
      gps.f_get_position(&flat, &flon, &age);
      Serial.print("LAT=");
      Serial.print(flat == TinyGPS::GPS_INVALID_F_ANGLE ? 0.0 : flat, 6);
      Serial.print(" LON=");
      Serial.print(flon == TinyGPS::GPS_INVALID_F_ANGLE ? 0.0 : flon, 6);
      Serial.print(" SAT=");
      Serial.print(gps.satellites() == TinyGPS::GPS_INVALID_SATELLITES ? 0 : gps.
tellites());
      Serial.print(" PREC=");
      Serial.print(gps.hdop() == TinyGPS::GPS_INVALID_HDOP ? 0 : gps.hdop());
   }

   gps.stats(&chars, &sentences, &failed);
   Serial.print(" CHARS=");
   Serial.print(chars);
   Serial.print(" SENTENCES=");
   Serial.print(sentences);
   Serial.print(" CSUM ERR=");
   Serial.println(failed);
}
```

2

[2] Imagen 1.3 Código ejemplo fuente: https://bit.ly/2N8QmbW[2]

1.4 Aplicación en POWER BIT

La aplicación del módulo GPS NEO6M en nuestro dispositivo es una parte muy esencial ya que este se encarga de enviar en tiempo real la ubicación del vehículo el cual muestra el siguiente funcionamiento.

Lo primero que hicimos fue configurar el puerto serial a una velocidad de 9600 bauidios, por lo tanto nos hace preguntarnos ¿Por qué tenemos que configurar el puerto a 9600 baudios?

```
Serial.begin(9600);
```

La respuesta a esa pregunta es: porque es una medida estándar de transmisión de baudios que Arduino maneja para poderse sincronizar con este módulo, pero hay que tener cuidado no todos los módulos que Arduino utiliza funcionan con esa velocidad de baudios, eso ya depende de cada uno de los módulos que se vallan a usar.
Por lo cual se recomienda ver primero el datasheet del módulo que se vaya a ocupar.

- CONEXIÓN CON ARDUINO FISICA

La conexión de Arduino con el modulo

Arduino	GPS	
D1 (TX)	RX	TX= TRANSMISIÓN
D0 (RX)	TX	RX= RECEPCIÓN

CONEXIÓN

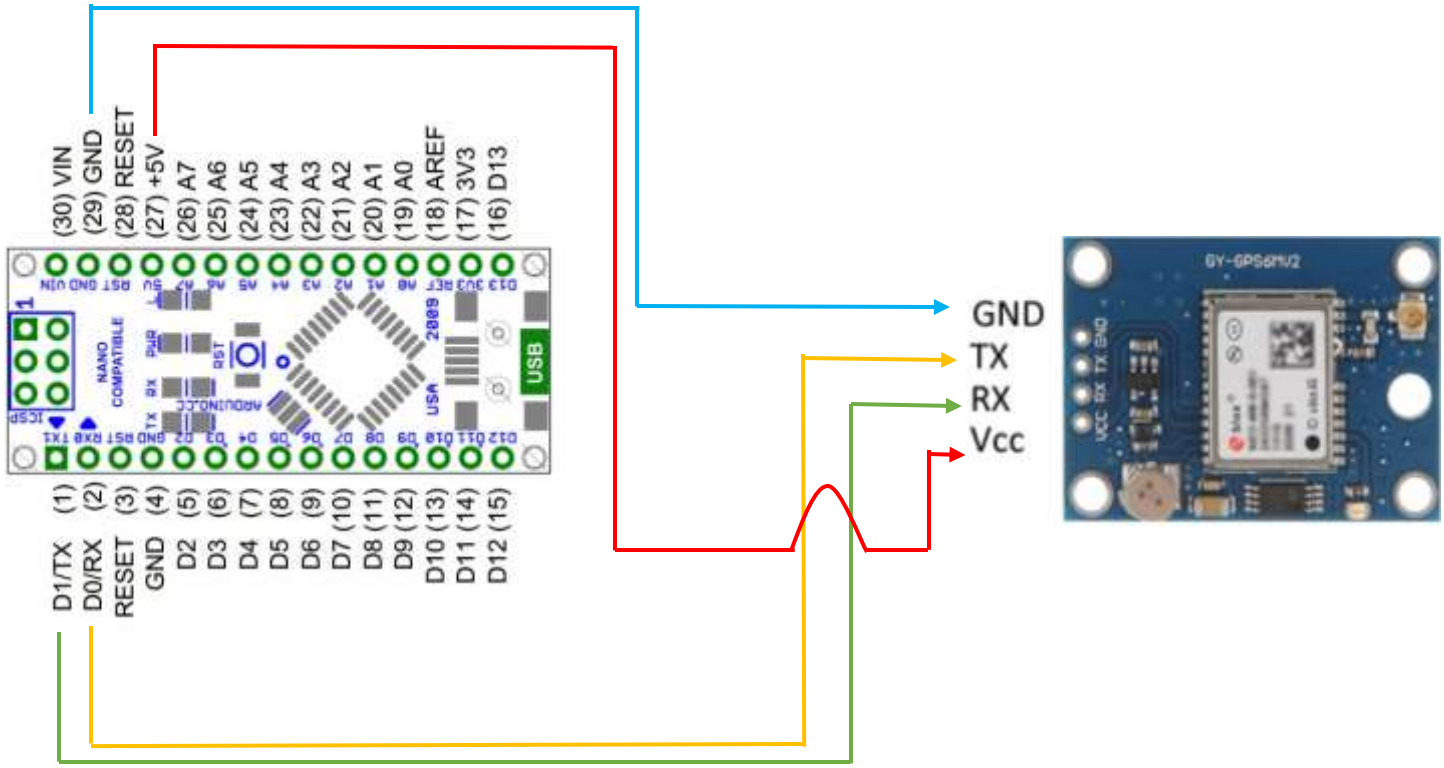


IMAGEN 1.3
CONEXIÓN FÍSICA DEL MÓDULO GPS CON ARDUINO

- CODIGO

```
void setup() {
  Serial.begin(9600);

  Serial.println("");
  Serial.println("GPS****POWER BIT****TBN");
  Serial.println(" ---BUSCANDO SATELITES--- ");
  Serial.println("");
                    }
```

Para poder visualizar los mensajes en el monitor serial primero se debe configurarlo a una velocidad de 9600 baudios.

Cuando compilamos y ejecutamos el código, Arduino nos muestra en el monitor serial el siguiente mensaje.

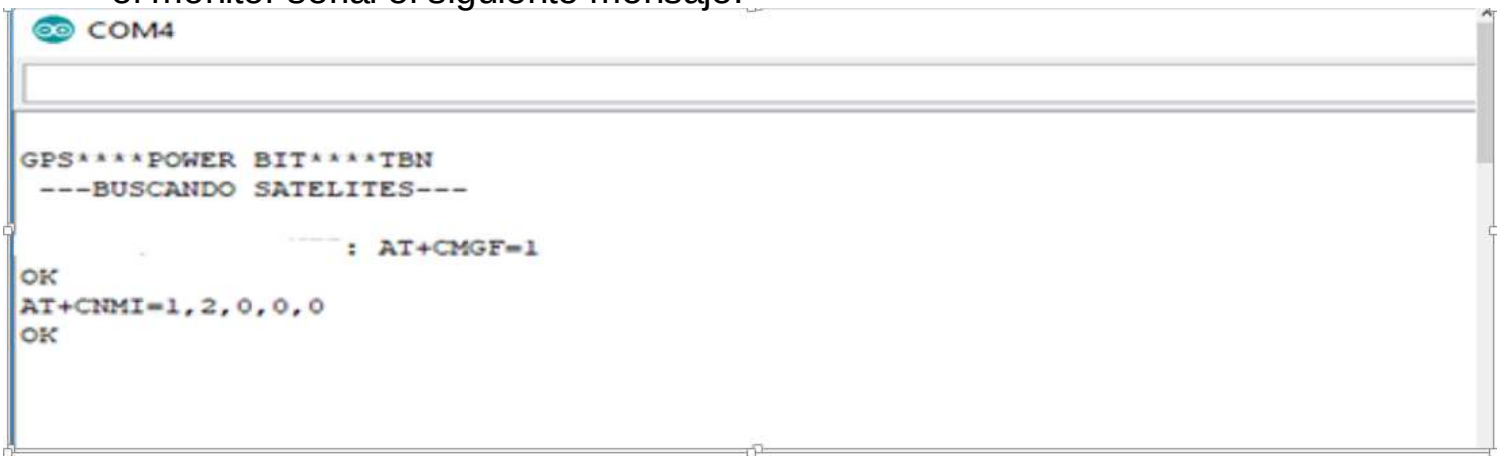


IMAGEN 1.4
MENSAJE QUE MUESTRA EL MONITOR SERIAL SI EL CÓDIGO SE EJECUTÓ BIEN

Observamos en el monitor serial el mensaje de que nuestro GPS ya inicio la búsqueda de los satélites disponibles para la conexión, otra forma más sencilla de saber si el GPS cuenta con conexión satelital es por medio del led que tiene integrado el módulo GPS, una vez que encuentra conexión con algún satélite dicho led empieza a encender y apagar una vez por segundo.

CAPÍTULO 2: USO DE LA RED TELEFONICA CELULAR

Como ya se describió en el capítulo anterior los módulos que usamos son una parte esencial del proyecto, para la conexión de la red se utilizó el modulo sim 800L cuyas especificaciones son las siguientes.

2.1 Especificaciones sim 800L

- Voltaje de Operación: 3.4V - 4.4V DC
- Nivel Lógico de 3V a 5V
- Consumo de corriente (máx): 500 mA
- Consumo de corriente (modo de reposo): 0.7 mA
- Interfaz: Serial UART
- Quad-band 850/900/1800/1900MHz – se conectan a cualquier red mundial GSM con cualquier SIM 2G
- Trabaja solo con tecnología 2G
- Hacer y recibir llamadas de voz usando un auricular o un altavoz de 8Ω externo + micrófono electret.
- Enviar y recibir mensajes SMS
- Enviar y recibir datos GPRS (TCP/IP, HTTP, etc)
- Escanear y recibir emisiones de radio FM
- Controlado por Comandos AT
- Interfaz de comandos AT con detección "automática" de velocidad de transmisión
- Soporta A-GPS
- Datos GPRS:
    - Velocidad máxima de transmisión 85.6 Kbps
    - Protocolo TCP/IP en chip
    - Codificación: CS-1, CS-2, CS-3 y CS-4
    - Soporta USSD
- Soporta reloj en tiempo real (RTC)
- Velocidades de transmisión serial desde 1200bps hasta 115 200 bps
- Tamaño de la SIM: Micro SIM

3

[3] Fuente: https://bit.ly/2wWwBxS

2.2 Funcionamiento

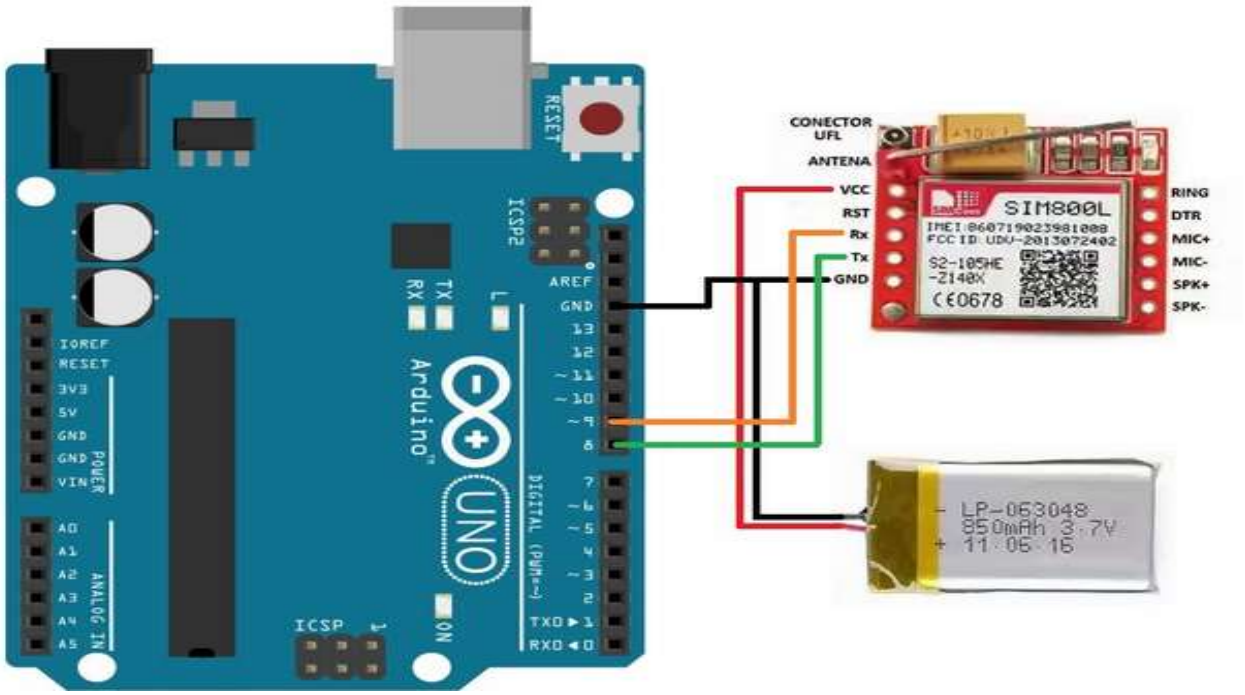


IMAGEN 2.1 [4]
CONEXIÓN FÍSICA ARDUINO-MODULO SIM 800L JUNTO CON UNA FUENTE DE ALIMENTACIÓN EXTERNA

Para saber si el modulo SIM 800L está funcionando bien haremos la siguiente conexión y cargaremos el siguiente código, este código solo muestra si la SIM 800l trabaja correctamente.

[4] Imagen obtenida https://bit.ly/2AUmaNV

```
sketch_sep09a §
#include <SoftwareSerial.h>
SoftwareSerial SIM800L(8, 9);
String Comando;
void setup() {
  Serial.begin(9600);
  SIM800L.begin(9600);
  SIM800L.println("AT+CMGF=1");
  delay(1000);
  SIM800L.println("AT+CNMI=1,2,0,0,0");
  delay(1000);

}

void loop() {
  // ACCIONES A TOMAR "POWER BIT"
  if (SIM800L.available()) {
    Comando = SIM800L.readString();
    Serial.println("NUEVO SMS ENTRANTE: " + Comando);
  }

}
```

IMAGEN 2.2
CÓDIGO PARA VERIFICAR EL CORRECTO FUNCIONAMIENTO DEL MÓDULO SIM 800L

SIM800L.println ("AT+CMGF=1"); configura al módulo para trabajar con SMS
SIM800L.println ("AT+CNMI=1,2,0,0,0"); configura al módulo para obtener los SMS del monitor serial
if (SIM800L.available ()) { verifica si hay datos disponibles

Una vez cargado el código y ejecutado nos muestra el siguiente mensaje

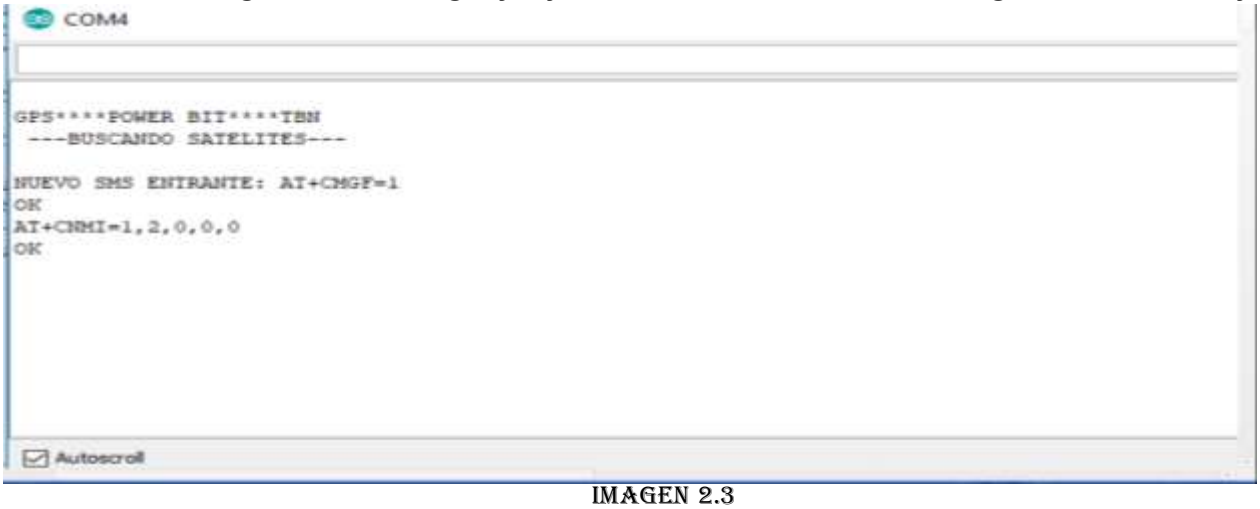


IMAGEN 2.3
MENSAJE DEL MONITOR SERIAL SI EL CÓDIGO SE EJECUTÓ CORRECTAMENTE

En el monitor serial indica que el modulo ya funciona correctamente y está listo para trabajar, pero ¿Cómo podemos saber si el modulo funciona sin ver el monitor serial?

La respuesta a esa pregunta es más fácil de contestar, el modulo cuenta con un led que ya viene integrado.

Cuando el led parpadea una vez cada 5 segundos significa que el módulo está funcionando bien y que a su vez ya está conectado a una red de telefonía celular.

Si el led parpadea 1 vez cada 0.5 seg quiere decir que el modulo funciona pero todavía no ha encontrado una red de telefonía celular a la cual conectarse.

2.3 Problemas con la sim 800l

El modulo sim 800L es un poco difícil de usar, conforme se fue avanzando en el proyecto nos fuimos encontrando con problemas de funcionamiento de dicho modulo.

a) El modulo como se puede ver el apartado de especificaciones menciona que funciona de 3 a 5 V, sabiendo esto aquí es donde se encuentro nuestro principal problema la Energía, al conectar el modulo nos dimos cuenta que nunca encontraba la señal de la red, el funcionamiento que presentaba era el siguiente.

- Al conectar el modulo con Arduino y el circuito el comportamiento de la sim 800L era que su led parpadeaba 7 veces seguidas y después hacia una pausa de 2 seg y volvía a repetir el ciclo marcándonos en el monitor serial ERROR.
- Al conectar el modulo solo con Arduino su comportamiento fue distinto ahora, el led del módulo empezó a parpadear 3 veces seguidas con una pausa de 2 seg iniciando el ciclo de nuevo marcándonos ERROR en el monitor serial.

Entonces aquí surgió una pregunta ¿Por qué no funciona el módulo? decidimos comprar otro modulo pensando que el que teníamos estaba dañado de fábrica, después de comprar el nuevo módulo el comportamiento de este fue exactamente el mismo.

Como el comportamiento en los 2 casos fue el mismo decidimos verificar la energía que alimentaba al módulo cuando este se encuentra conectado al circuito y cuando no lo está.

Conectado al circuito y Arduino el voltaje era de 3.84v
Conectado solo a Arduino el voltaje era de 4.86v

En cualquiera de los 2 casos entraban en el rango de voltaje establecido por el fabricante del módulo, pero sabíamos que el problema era con energía así que empezamos a probar el modulo con 5v e ir bajando el voltaje para ver el comportamiento del modulo

Nuestro modulo solo funciono bien cuando le conectamos de 4 a 4.25 volts
Entonces nuestra observación fue la siguiente cuando el modulo está conectado al circuito y con Arduino el voltaje baja por el consumo de todo el circuito por eso nos marcaba 3.84v

La segunda observación fue cuando el módulo solo está conectado a Arduino el voltaje que nos marcaba era de 4.86v por lo cual tenía sobre voltaje

La única solución que pudimos encontrar fue agregarle al circuito un regulador de 5 volts conectado a 12v puesto que el regulador nos baja a 5 volts más el consumo de todo el circuito nos bajaba a los 4.25 volts que necesitábamos para que el modulo funcione

b) El segundo problema con el cual nos encontramos fue que no todos los chips de telefonía celular acepta

CAPÍTULO 3: ALARMA

3.1 Descripción de la alarma

La alarma que usamos en el proyecto fue una sirena piezoeléctrica pequeña, esta sirena se escogió por 2 grandes ventajas que poseía a comparación de otras sirenas que encontramos.

1) Es bastante pequeña para poder ocultarla dentro de cualquier vehículo.
2) Provoca un sonido muy fuerte que llega a ser demasiado incomodo incluso puede lastimar el oído si se permanece en el lugar donde ha sido activada.

IMAGEN 3.1 SIRENA PIEZOELÉCTRICA

Características

- sirena con cables
- alimentación: 6-15Vdc
- consumo (a 12Vdc con sirena): 200mA
- frecuencia de oscilación: 1.5-3.5kHz
- nivel de presión sonora (a 1m): 107dBA
- dimensiones: 40 x 45 x 60mm
- peso: 47g
- agujeros de montaje: 22mm “

5

[5] FUENTE https://bit.ly/2x7iV3m

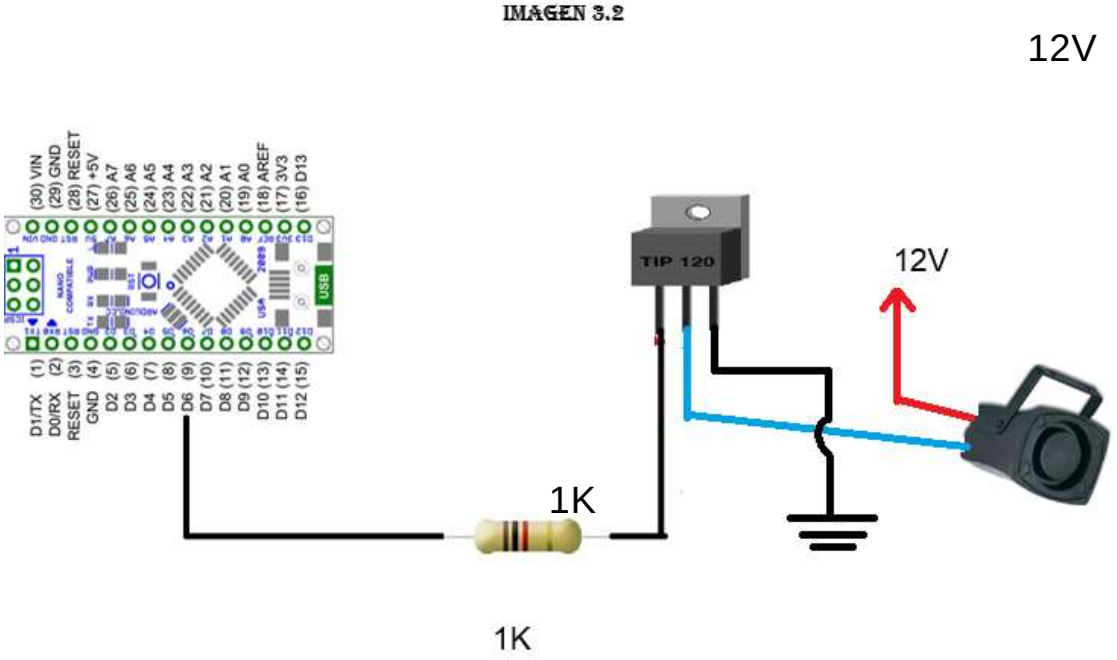
3.2 Aplicación en el proyecto

La principal aplicación de la alarma es hacer entrar en pánico al ladrón y aturdirlo, la alarma trabaja junto con la sim 800l para poder funcionar.

Cuando el modulo sim recibe un comando de activación entra en ejecución la alarma y el apagado del motor que se verá más adelante.

Conexión

IMAGEN 3.2



CONEXIÓN FÍSICA ARDUINO-BOCINA

La sirena no puede ir conectada directamente a Arduino porque este no puede suministrar los 12 volts que necesita la sirena para funcionar, para resolver este problema se usó un TIP 120 su principal función es trabajar como un switch.

Para recibir el mensaje la sim 800l tiene que estar en un modo de espera

```
void setup() {
  Serial.begin(9600);
  SIM800L.begin(9600);   //antes 19200
  while (!Serial);
  SIM800L.println("AT+CMGF=1"); //1
  delay(1000);
  SIM800L.println("AT+CNMI=1,2,0,0,0");

}

void loop() {

if (SIM800L.available()) {
    Comando = SIM800L.readString();
    Serial.println("NUEVO SMS ENTRANTE: " + Comando);
  }

  if (Comando.indexOf("rzc") >= 0) {
    Serial.println("ALARMA ACTIVADA");
    digitalWrite(ALARMA, HIGH);

    digitalWrite(6, LOW);
    delay(200);
    if (Contador3 == 0) {
      APAGANDOMOTOR();
    }
    Comando = "";
  }
}
```

Esta parte del código se encarga de poner a la sim 800l en un modo de espera y en cuanto recibe un SMS entra en ejecución el proceso de reconocimiento del SMS, si este es correcto el programa ejecuta la siguiente fase .

"Comando" es una variable que se declaró de tipo String esta variable se encarga de guardar el SMS que llega al módulo sim 800l, aquí no importa lo que el SMS contenga simplemente lo guarda en la variable.
Convirtiendo así el SMS en una cadena de caracteres para facilitar el reconocimiento del mensaje.

```
void loop() {

1) if (SIM800L.available()) {
     Comando = SIM800L.readString();
     Serial.println("NUEVO SMS ENTRANTE: " + Comando);
   }

2) if (Comando.indexOf("rzc") >= 0) {
     Serial.println("ALARMA ACTIVADA");
     digitalWrite(ALARMA, HIGH);
```

1) En esta parte del código la SIM 800L ya se encuentra en un estado de espera y listo para recibir SMS.
   En cuanto reciba un SMS guardara este en la variable que ya declaramos llamada "Comando", no importa lo que diga el SMS en esta parte solo lo guarda para después ejecutar el siguiente comando.

2) Una vez guardado el SMS en la variable "Comando" se ejecuta el siguiente "if" que se encarga de comparar el SMS con una palabra previamente programada en este caso "rzc".

Para esto se usó el método indexOF que se encarga de comparar la cadena que contiene la variable “Comando” con la palabra a la que ya se programo Si “Comando” contiene la palabra reservada rzc activara la alarma de lo contrario solo mostrara en el Serial el SMS que contiene “Comando” en nuestro ejemplo nosotros le enviamos un SMS con las palabras “Este es un sms erróneo”.

IMAGEN 3.3
EL MONITOR SERIAL MUESTRA EL MENSAJE CON EL SMS QUE RECIBIÓ EL MÓDULO SIM800L

CAPÍTULO4: INHABILITAR VEHÍCULO

4.1 Descripción general

Para inhabilitar el vehículo se requirieron conocimientos de mecánica automotriz, para que el vehículo deje de avanzar tuvimos que realizar unos cambios a la bomba de gasolina, ya que nosotros contábamos con muy pocos conocimientos de mecánica pedimos ayuda a un eléctrico de vehículos, él nos indicó el punto donde debíamos realizar cambios para que el vehículo se pudiera conectar con Arduino y poder controlar su apagado desde este.

Arduino se encargará de desactivar al vehículo mandando una señal a un módulo relé que a su vez este estará conectado a la alimentación de la bomba de gasolina

Lo que haremos será interrumpir la inyección de gasolina a las bujías provocando así que el vehículo se detenga,

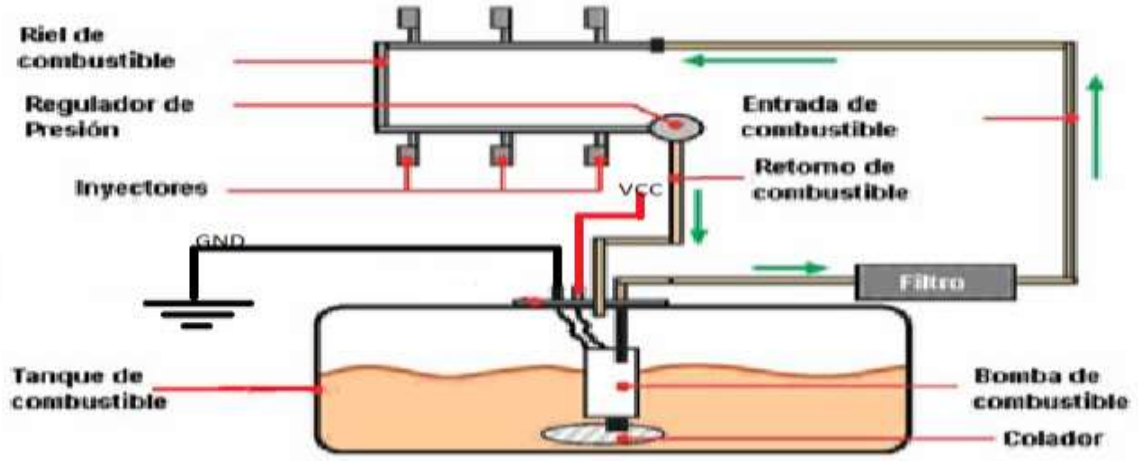


IMAGEN 4.1 RETORNO DE GASOLINA

4.2 Funcionamiento

6 Fuente https://bit.ly/2O1ldLc

Como vimos antes el pagado del vehículo será controlado desde un celular atreves de un comando SMS

```
void setup() {
  Serial.begin(9600);
  SIM800L.begin(9600);   //antes 19200
  while (!Serial);
  SIM800L.println("AT+CMGF=1"); //1
  delay(1000);
  SIM800L.println("AT+CNMI=1,2,0,0,0");

}

void loop() {

if (SIM800L.available()) {
     Comando = SIM800L.readString();
     Serial.println("NUEVO SMS ENTRANTE: " + Comando);
   }

   if (Comando.indexOf("rzc") >= 0) {
     Serial.println("ALARMA ACTIVADA");
     digitalWrite(ALARMA, HIGH);

     digitalWrite(6, LOW);
     delay(200);
     if (Contador3 == 0) {
       APAGANDOMOTOR();
     }
     Comando = "";
   }
}
```

Ahora bien el Comando que se usó para activar la alarma será el mismo para mandar a llamar a un método que se encargara de la desactivación de la bomba de gasolina.

Una vez que a nuestra sim 800l reconoce el comando que en nuestro caso es “rzc” lo primero que hace es activar la alarma si la comparación de “Comando” es la misma con la palabra rzc.

Ya activada nuestra alarma lo siguiente que hace el código es llamar a un método llamado APAGANDOMOTOR(); este método contiene las instrucciones para desactivar a la bomba de gasolina.

```
  if (Comando.indexOf("rzc") >= 0) {
    Serial.println("ALARMA ACTIVADA");
    digitalWrite(ALARMA, HIGH);

    digitalWrite(6, LOW);
    delay(200);
    if (Contador3 == 0) {
      APAGANDOMOTOR();
    }
    Comando = "";
  }
}
```

Como se observa tenemos declarada una bandera llamada contador3 este contador se encarga de no mandar a llamar al método APAGANDOMOTOR dos veces, así nuestro vehículo al reconocer el comando “rzc” iniciara la alarma y detendrá al vehículo que probablemente hará que el ladrón entre en pánico puesto que no podrá encender el vehículo de nuevo.

Comando =” ”; esta parte del código es importante ya que limpia a nuestra variable llamada “Comando” haciendo que este vacía otra vez poniendo en espera a la sim de nuevo.

```
void APAGANDOMOTOR() {
  Serial.print("INICIANDO APAGADO DE MOTOR");
  digitalWrite(GASOLINA, HIGH);
  delay(2000);
  Contador3++;
}
```

GASOLINA es una variable que declaramos como una salida que al ser llamado el método de APAGANDOMOTOR entra en un estado de 1, esta señal la recibirá nuestro módulo relé cortando la alimentación de la bomba de gasolina haciendo que esta deje de funcionar.

Si es la primera vez que se activa contador3 ahora adquiere un valor de 1, Se pensó así porque si el propietario del vehículo, acompañantes o familiares activan la alarma al mismo tiempo Arduino no pierda tiempo llamado al método todas las veces que le llegue el comando rzc, simplemente el primer SMS con el comando correcto que reciba activara la alarma desactivando al vehículo al mismo tiempo.

Si recibe otro SMS correcto ya no mandará a llamar al método puesto que su valor ya no es 0 ahora es 1 tras el primer SMS correcto que recibió haciendo que este no mande a llamar de nuevo al método puesto que ya se está ejecutando, mandarlo a llamar otra vez solo estaríamos desperdiciando memoria.

Ahora solo hay 2 formas de que nuestro vehículo vuelva a encender

1) Reiniciando Arduino:

Esta sería la forma más difícil puesto que se tendría que desarmar el aparato y presionar el botón de reinicio que ya viene integrado con Arduino

2) Que reconozca un comando de encendido

```
if (Comando.indexOf("OFF") >= 0) {
  Serial.println("ALARMA APAGADA");
  digitalWrite(LED, LOW);
  Comando = "";
  ENCENDIENDOMOTOR();
}
```

Al activar la alarma en el código se limpió la variable Comando haciendo posible que nuevamente puede recibir un SMS nuestro Arduino para ejecutar ahora en comando diferente.

Ahora en este caso apagaremos nuestra alarma y reactivaremos a nuestra bomba de gasolina para que nuestro vehículo pueda arrancar de nuevo.

Recibimos ahora un comando llamado OFF y al compáralo que sea el correcto ahora apagaremos la alarma y mandaremos a llamar a un método llamado ENCENDIENDOMOTOR.

El método ENCENDIENDOMOTOR contiene una instrucción que se encargara de activar a nuestra bomba y reiniciando el contador 3 igualándolo a 0 para que pueda ser usado otra vez en el método APAGANDOMOTOR.

```
}
void ENCENDIENDOMOTOR() {
  Serial.print("INICIANDO ENCENDIDO DE MOTOR");
  digitalWrite(GASOLINA, LOW);
  delay(2000);
  Contador3 = 0;

}
```

En esta parte del código ya es posible que nuestro vehículo encienda de nuevo si se ejecutó correctamente.

4.3 Conexión física

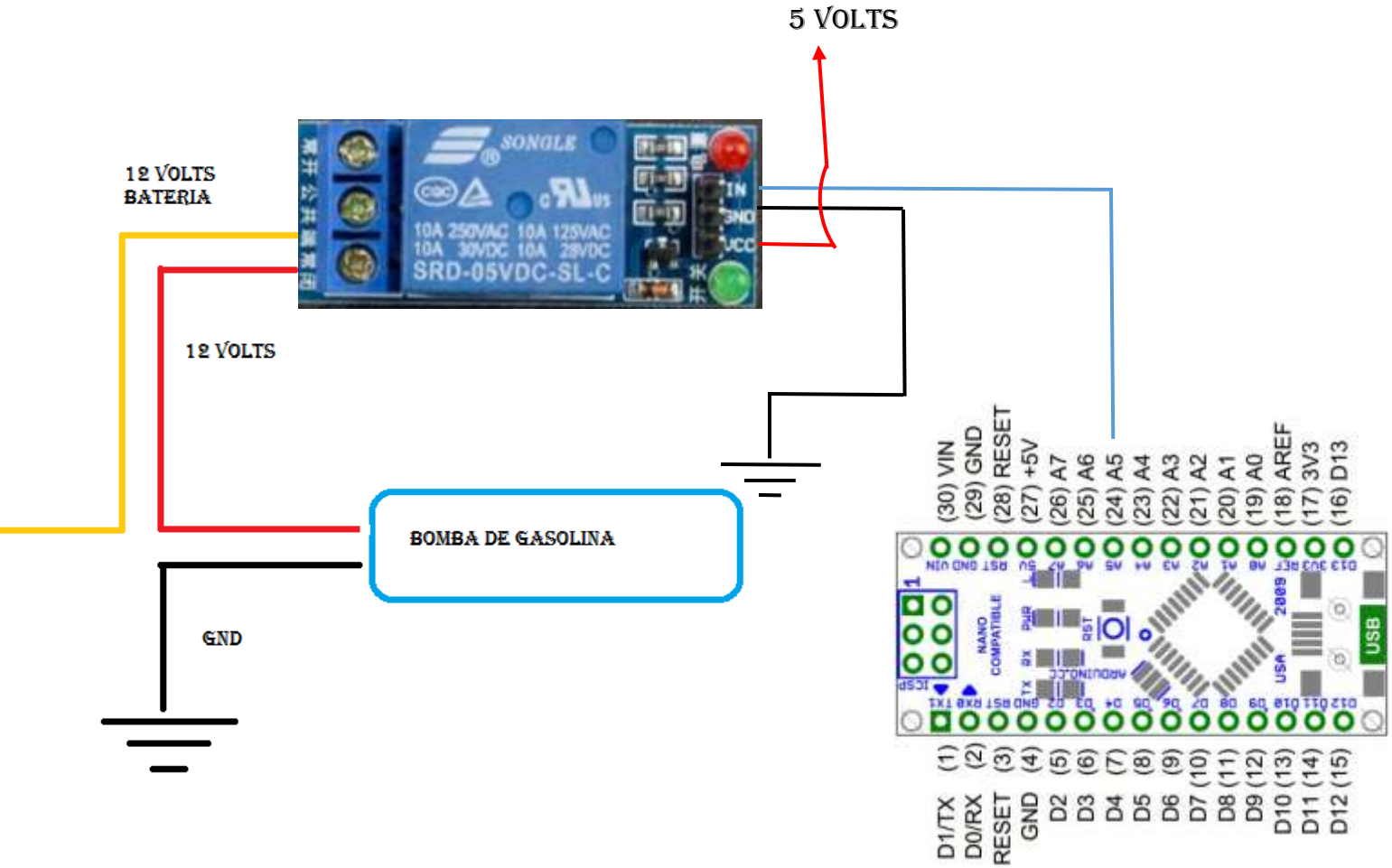


IMAGEN 4.3 CONEXIÓN FÍSICA BOMBA

Para conectar la bomba solo hay que intervenir el cable que le da energía a esta, como se puede observar en la imagen cortamos el cable de vcc de la bomba y colocamos en el módulo relé que este se encargara de desactivar o activar la bomba físicamente a la espera de una señal mandada desde Arduino.

CAPÍTULO 5: SMS DE LOCALIZACIÓN

5.1 Google Maps

Para localizar a nuestro vehículo hicimos uso de la herramienta “Google Maps”, POWER BIT se encargará de mandar una liga de Google Maps a un teléfono o teléfonos ya programados en este en tiempo real.

“Lanzar Google Maps y realizar una acción específica

Para iniciar Google Maps y, opcionalmente, realizar una de las funciones admitidas, utilice un esquema de URL de uno de los siguientes formularios, según la acción solicitada:

- ***Buscar** : abra un mapa de Google que muestre un marcador para un lugar específico, o realice una búsqueda general y ejecute un mapa para mostrar los resultados: https://www.google.com/maps/search/?api=1 & parameters*
- ***Indicaciones** : solicite indicaciones e inicie Google Maps con los resultados: https://www.google.com/maps/dir/?api=1 & parameters*
- ***Mostrar un mapa** : inicie Google Maps sin marcadores ni direcciones: https://www.google.com/maps/@?api=1&map_action=map & parameters*
- ***Mostrar un panorama de Street View** : iniciar una imagen panorámica interactiva: https://www.google.com/maps/@?api=1&map_action=pano & parameters*

***Importante**: el parámetro api=1 identifica la versión de las URL de Maps a las que está destinada esta URL. Este parámetro es requerido en cada solicitud. El único valor válido es 1. Si api=1 NO está presente en la URL, todos los parámetros se ignoran y se iniciará la aplicación predeterminada de Google Maps, ya sea en un navegador o en la aplicación móvil Google Maps, dependiendo de la plataforma en uso (para ejemplo, https://www.google.com/maps).”* [7]

Como ya se habló en los capítulos anteriores el modo de localización será habilitado por un comando SMS ya programado que POWER BIT reconocerá iniciando así nuestra secuencia de localización del vehículo.

[7] Fuente https://developers.google.com/maps/documentation/urls/guide

5.2 Creando liga de Google Maps en Arduino

Dado que Arduino no puede conectarse directamente a internet sin módulos que realicen esta acción nos dimos a la tarea de que Arduino pueda crear una liga de Google Maps obteniendo las coordenadas de nuestro modulo GPS UBLOX NEO6M y enviando la liga a los números telefónicos programados.

```
SIM800L.write("AT+CMGS=\"+52********\"\r\n");
delay(100);


//Enviar contenido del SMS
char lati[32];
char lon[32];

dtostrf(latitud, sizeof(latitud), 5, lati);
dtostrf(longitud, sizeof(longitud), 5, lon);
Serial.println("tel 1");
url = "https://www.google.com.mx/maps/search/?api=1&query=";

url.concat(lati);
url.concat(',');
url.concat(lon);
Serial.println(url);
SIM800L.println(url);
```

IMAGEN 5.1 CÓDIGO PARA CREAR LIGA DE GOOGLE MAPS

En el código es importante definir a que numero celular se le enviara nuestra liga de Google Maps para ello se configura el módulo Sim 800l con el comando AT+CMGS para que pueda mandar nuestro SMS a ese número programado.

El modulo Sim 800l está diseñado para trabajar con cualquier chip de telefonía celular siempre y cuando estas manejen la tecnología 2G en cualquier país.
Una vez conociendo esto surge una pregunta.

¿Cómo nuestro modulo reconoce los números celulares de México?

La respuesta a esa pregunta es muy fácil, [8]

Como observamos en la *Imagen 5.1 creación de liga Maps* el código nos muestra una configuración AT+CMGS seguido de un +52 más asteriscos el +52 es el prefijo telefónico en este caso de México los asteriscos son el número a quien nosotros le enviamos el SMS
EJEMPLO:

```
SIM800L.write("AT+CMGS=\"+525522222222\"\r\n");
delay(100);
```

[8] Visitar http://www.countryareacode.net/es/mexico

Para la creación de la liga se declaró una variable llamada URL de tipo String, la cual se igualo a una dirección de Google Maps, lo único que le falta a esta liga juntarla con las coordenadas de latitud y longitud que proporciona el GPS.

Las variables latitud y longitud son de tipo float como el modulo sim no puede mandar por SMS este tipo de variables se tienen que convertir antes a un tipo de variables que pueda manejar texto en este caso fueron declaradas como char asignándoles un nuevo nombre lati y lon.

Ahora en la siguiente parte del código se concatenan estas variables a la liga que ya tiene nuestra variable llamada url.

```
SIM800L.println(url);
```

Después de concatenar las coordenadas del GPS a nuestra liga de Google mandamos nuestra variable url a la sim 800l que se encargara de mandar nuestra liga con las coordenadas al número que ya se programó con anterioridad.

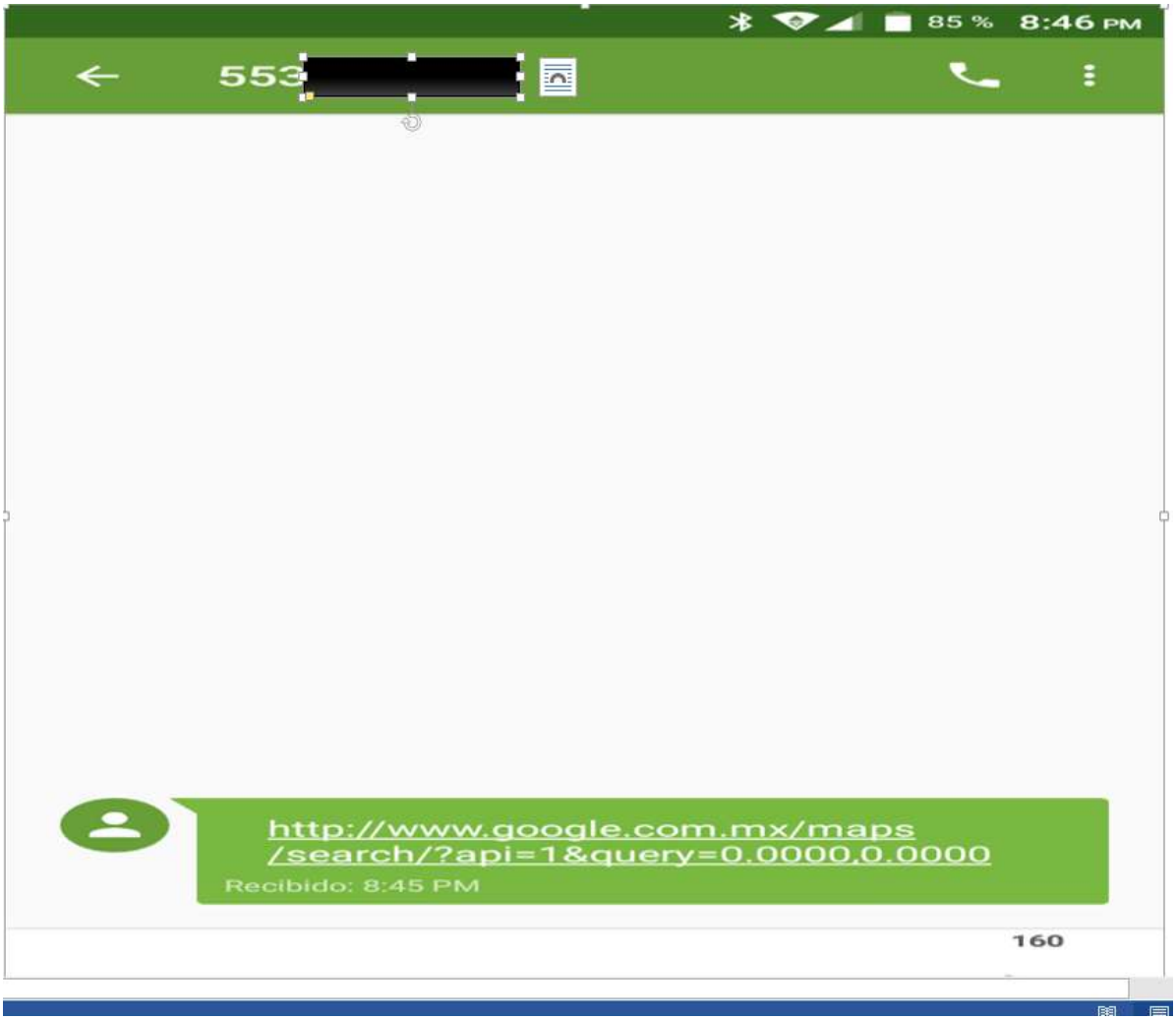


IMAGEN 5.2
VISTA DE LA LIGA DE GOOGLE JUNTO CON LAS COORDENADAS EN UN TELÉFONO CELULAR

Y finalmente así es como recibimos el SMS en nuestro teléfono con la liga de Google Maps y sus coordenadas dando inicio a la localización de nuestro vehículo cabe mencionar que esta liga entra en un bucle mandando la liga a los teléfonos programados cada 20 segundos hasta que se mande el SMS de desactivación.

CAPITULO 6: ANALISIS FINANCIERO

6.1 Costos e inversiones

COSTOS FIJOS	
CONCEPTO	**COSTO AL MES**
Plan celular	$200 pesos
Total	$200 pesos

COSTOS VARIABLES	
CONCEPTO	**COSTO POR UNIDAD**
Modulo GPS neo 6m	$200 pesos
Arduino nano	$80 pesos
Tarjeta sim	$100 pesos
Led	$1 pesos
Botones	$2 pesos
Placa fenólica	$200 pesos
Bocina	$150 pesos
Impresión del circuito	$100 pesos
Total	$833 pesos

INVERSION INICIAL		
CONCEPTO	**COSTO POR UNIDAD**	
computadora	**$3000**	pesos
Total	$3000	pesos

Costo total = costo f + costo v

Costo total =200+833 =1033 pesos

El artículo 34 de la ley isr de impuesto sobre renta dice que la depreciación de equipos de cómputo (pc, computadoras personales, impresoras) es del 30%[9]

Realizando el cálculo

3000 * 0.30 = 900 pesos

Por lo cual la computadora se deprecia 900 pesos por año

[9] Fuente http://xurl.es/uixwb

6.2 Reduciendo costos

COSTOS VARIABLES		
CONCEPTO	**COSTO POR UNIDAD**	**Costo x 10 unidades**
Modulo GPS neo 6m	$200 pesos	$180 pesos
Arduino nano	$80 pesos	$50 pesos
Tarjeta sim	$100 pesos	$50 pesos
Led	$1 pesos	$1 pesos
Botones 2 polos	$2 pesos	$1 pesos
Placa fenólica	$200 pesos	$150 pesos
Bocina	$150 pesos	$140 pesos
Impresión del circuito	$100 pesos	$ 80 pesos
Total	$833 pesos	652 pesos

INVERSION INICIAL	
CONCEPTO	**COSTO POR UNIDAD**
computadora	**$3000** pesos
Total	$3000 pesos

Costo real = costo f + costo v + ganancia

Suponiendo que se vendan 10 aparatos por mes nos da un total de 120 aparatos por año

Depreciación / aparatos por año

900/120= $7.5

Costo real por unidad = 200 + 652 + 7.5 = $ 859.5

859.5 pesos es lo que saldría en crear un prototipo en mayoreo contemplado el uso de la computadora

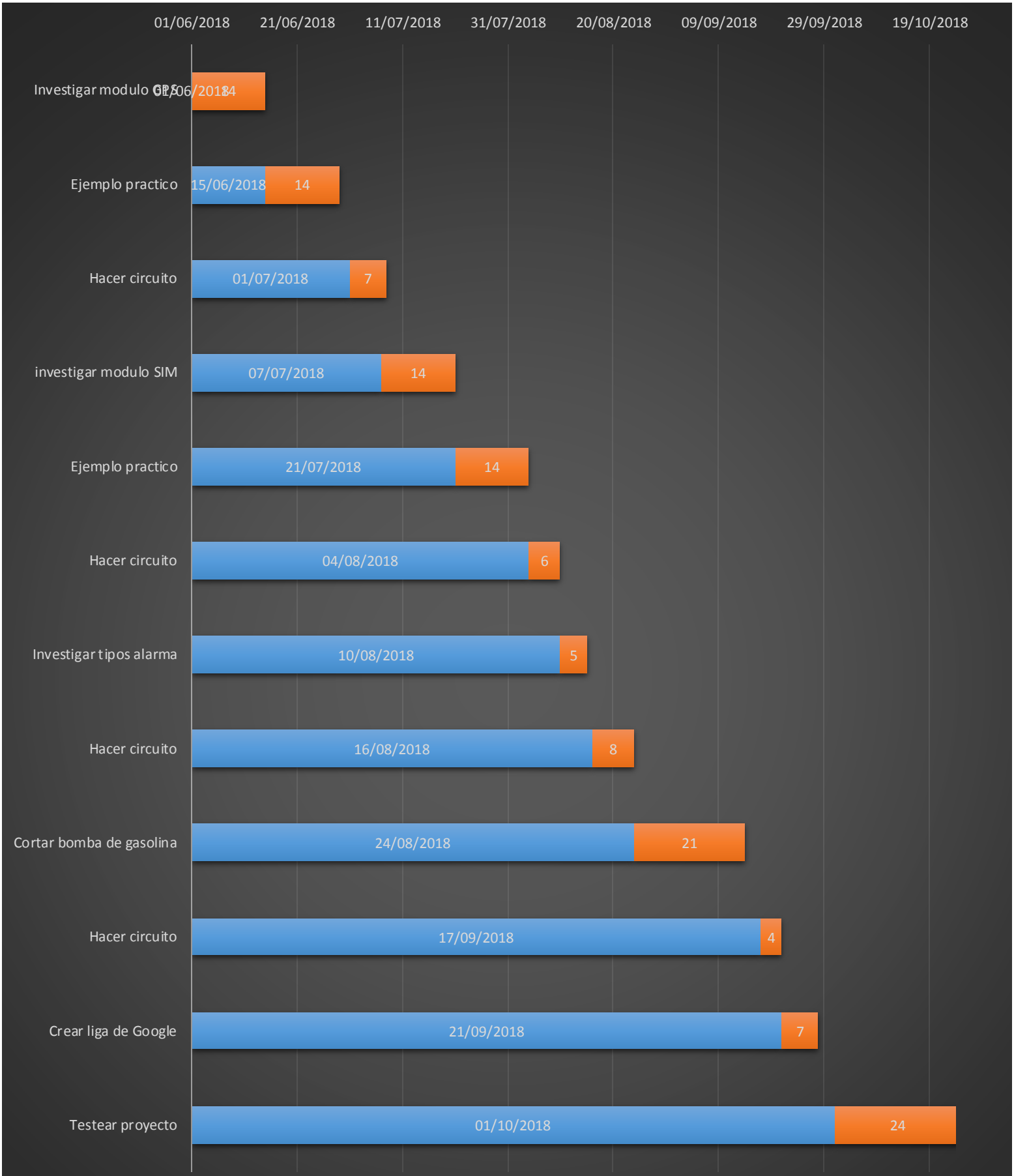
Se pretende ganar más el 15% del valor del producto

859.5*0.15= $128.5

Costo real = costo f + costo v + ganancia

Costo real = 200 + 659.5 + 128.9 = $988.4 pesos

DIAGRAMA DE GANTT



	Testear proyecto	Crear liga de Google	Hacer circuito	Cortar bomba de gasolina	Hacer circuito	Investigar tipos alarma	Hacer circuito	Ejemplo practico	investigar modulo SIM	Hacer circuito	Ejemplo practico	Investigar modulo GPS
Fecha de inicio	01/10/2018	21/09/2018	17/09/2018	24/08/2018	16/08/2018	10/08/2018	04/08/2018	21/07/2018	07/07/2018	01/07/2018	15/06/2018	01/06/2018
Duracion en dias	24	7	4	21	8	5	6	14	14	7	14	14

Fecha de inicio Duracion en dias

CONCLUSIÓN

Finalmente los conocimientos adquiridos de la carrera de ingeniería en computación fueron aplicados para aportar un servicio a la sociedad, estos conocimientos fueron de mucha utilidad para el desarrollo de este proyecto y de esta manera poderlo implementar de manera satisfactoria.

Este proyecto fue desarrollado en Arduino dado que es una plataforma de código abierto teniendo como base el Hardware y Software que son sencillos de usar, existen otros tipos de micro controladores pero se escogió Arduino por tener estas ventajas[10]

- MULTIPLATAFORMA
- ENTORNO DE PROGRAMACION SIMPLE
- BARATO

Por las razones ya mencionadas POWER BIT es un proyecto de seguridad que no solo se pone al alcance de cualquier persona si no que ayuda a resolver uno de los tantos problemas que nos afectan como comunidad, debido a que ya existe proyectos así en el mercado POWER BIT busca ser más accesible para las personas que no dispongan de mucho capital

Es un aparato pequeño que debido a su tamaño puede ser escondido dentro de cualquier vehículo no importando el tamaño de este, por lo cual lo hace muy discreto y sencillo de conectar ya que su principal fuente de energía es la misma que alimenta a los componentes del vehículo (batería del vehículo)

[10] https://arduino.cl/que-es-arduino/

GLOSARIO

BAUDIOS

Unidad de medida utilizada en comunicaciones. Hace referencia al número de intervalos elementales por segundo que supone una señal. Velocidad con que se mide un módem.

Velocidad de señalización de una línea. Es la velocidad de conmutación, o el número de transiciones (cambios de voltaje o de frecuencia) que se realiza por segundo. Sólo a velocidades bajas, los baudios son iguales a los bits por segundo; por ejemplo, 300 baudios equivalen a 300 bps. Sin embargo, puede hacerse que un baudio represente más de un bit por segundo. Por ejemplo, el modem V.22bis genera 1,200 bps a 600 baudios.

Es una medida de la velocidad de modulación, correspondiente al número de cambios en una señal por segundo. Se suele hablar indistintamente de 'bits por segundo' y de 'baudios' habiéndose convertido de hecho en falsos sinónimos. El número de bits por segundo dividido por el número de bits de datos por señal da como resultado el número de baudios.

... Fuente https://sistemas.com/baudio.php

MÉTODO

Es la implementación de un algoritmo que representa una operación o función que un objeto realiza. El conjunto de los métodos de un objeto determinan el comportamiento del objeto.

…Fuente https://msdn.microsoft.com/es-es/library/bb972232.aspx

URL

Es una sigla del idioma inglés correspondiente a **Uniform Resource Locator** (**Localizador Uniforme de Recursos**). Se trata de la secuencia de caracteres que sigue un estándar y que permite denominar recursos dentro del entorno de **Internet** para que puedan ser localizados.

Fuente https://definicion.de/url/

TECNOLOGÍA 2G

Se conoce como telefonía móvil 2G a la segunda generación de telefonía móvil.
La telefonía móvil 2G no es un estándar o un protocolo sino que es una forma de marcar el cambio de protocolos de telefonía móvil analógica a digital.

Fuente https://bit.ly/2Sg9Im3

HARDWARE

La **Real Academia Española** define al **hardware** como el **conjunto de los componentes** que conforman la **parte material** (**física**) de una computadora, a diferencia del **software** que refiere a los **componentes lógicos** (**intangibles**). Sin embargo, el concepto suele ser entendido de manera más amplia y se utiliza para denominar a todos los componentes físicos de una tecnología.

Fuente https://definicion.de/software/

SOFTWARE

Se considera que el software es el equipamiento lógico e intangible **de un ordenador. En otras palabras, el concepto de software abarca a todas las** aplicaciones informáticas**, como los procesadores de textos, las planillas de cálculo y los editores de imágenes.**

Fuente https://definicion.de/software/

REFERENCIAS

***referencias de internet**

Animal Político. (2016). Obtenido de https://www.animalpolitico.com/blogueros-causa-en-comun/2017/01/12/corrupcion-abuso-policial-apuntes/

Geek Factory. (s.f.). Obtenido de https://www.geekfactory.mx/tienda/radiofrecuencia/sim800l-modulo-gsm-gprs/

Google Maps Platform. (s.f.). Obtenido de https://developers.google.com/maps/documentation/urls/guide

Leantec robotics & electronics. (s.f.). Obtenido de https://leantec.es/blog/54_Tutorial-Arduino--Modulo-GPS-GPS6MV2.html

Llamas, L. (27 de Septiembre de 2016). *Tutoriales arduino*. Obtenido de https://www.luisllamas.es/localizacion-gps-con-arduino-y-los-modulos-gps-neo-6/

Planeta Electrónico. (s.f.). Obtenido de https://www.planetaelectronico.com/sirena-piezoelectrica-pequena-12vdc-p-6760.html

***Bibliografía**

OSCAR TORRENTE ARTERO (2016). GENUINO-ARDUINO (curso práctico de formación) / / ALFAOMEGA

AGRADECIMIENTO

Son muchas las personas a las que les debo decir "GRACIAS" principalmente a mis padres por ser los promotores de mis sueños y guías en mi vida, por invertir cada minuto de su tiempo, por cada consejo, por estar presentes no solo en esta etapa si no en cada momento de mi vida en el que solo me ofrecieron lo mejor de ustedes junto con mis hermanos y por permitirme conocer el infinito amor de DIOS que los puso en mi camino.

A mis maestros que a pesar de que a veces llegaba tarde nunca me negaron la entrada para seguir aprendiendo, principalmente a.

MTRO. Juan Gastaldi Pérez.

LIC. María Del Pilar García Villanueva.

M. EN C. Carlos Oliver Morales.

M. EN T.I Omar Mendoza Gonzáles.

Por creer en este proyecto e invertir su tiempo en revisarlo hasta verlo ya concluido solo me queda decirles gracias a cada uno de ellos y a los muchos más que tuve, mi generación llego a su final pero me siento orgulloso de saber que las futuras generaciones tendrán a estos excelentes maestros

"GRACIAS"

"Por mi raza hablara el espíritu"