Congreso Internacional en Ingeniería Electrónica. Mem. Electro 2015, Vol. 37, pp 114-118, Chihuahua, Chih. México
http://depi.itchihuahua.edu.mx/display/memorias_electro/MemoriaElectro2015
ISSN 1405-2172

MONITOREO DEL BUS CAN DE UN AUTOMOVIL CON ARDUINO Y UN CELULAR, UTILIZANDO BLUETOOTH.

Hernández R. Sergio I., Ortiz N. Bertha L., Meranza C. Manuel O.,
Ceballos G. Francisco J., Flores M. Gonzalo, Cruz Dario.
Instituto Tecnológico de Nogales
Ave. Instituto Tecnológico Núm. 911, Nogales, Sonora. C.P. 84065
Teléfono (631) 123- 9703.
sergio.hernandez@cicsa-net.com.mx

RESUMEN.
El presente trabajo es el resultado de la investigación realizada en el Instituto Tecnológico de Nogales, el cual contempla el diseño e implementación de un sistema de comunicación automotriz basado en el protocolo CAN (Controller Area Network) y la plataforma Arduino, con el fin de conocer el estado actual que guarda el automóvil (puertas, neumáticos, luces, asientos, entre otros). Se empleó tecnologías actuales como: Bluetooth (BT) para establecer la comunicación con un celular, habilitando al sistema para poder enviar mensajes y alertas, con los cuales se conoce las condiciones y el estado actual del automóvil.
Palabras Clave: CAN, Arduino, LabView, Bluetooth.

ABSTRACT.
This work is the result of research conducted at the Instituto Tecnológico de Nogales, which includes the design and implementation of a system of automotive communication protocol based on CAN (Controller Area Network) and the Arduino platform. For know the current status of the car (doors, tires, lights, seats, etc.). Current technologies such as was used: Bluetooth (BT) to communicate with a cell to send messages and alerts, with which you can know the conditions and the current state of the car.
Keywords: CAN, Arduino, LabView, Bluetooth.

1. INTRODUCCIÓN
Hoy en día es cada vez más demandante el uso de tecnología inalámbrica (BT) para conectar dispositivos electrónicos, en especial estudiemos el caso de los teléfonos celulares [1][2], los cuales es posible enlazarlos a la red del automóvil, utilizando las bondades de flexibilidad de esta tecnología y la cual condujo al objetivo del presente trabajo, que es mostrar la viabilidad del uso de tales protocolos dentro de un ambiente de comunicación automotriz a través de CAN y poder transmitir información que guardan los sensores y actuadores conectados a la red del automóvil al teléfono celular.

Se utilizaron módulos BT con un alcance de hasta 10m. Permitiendo de esta manera la comunicación inalámbrica con el celular, explotando al máximo los beneficios de BT y el estándar de comunicación CAN.

Los beneficios esperados a corto plazo, es el de tener a la mano la información necesaria proveniente de los sensores y actuadores del automóvil, para tomar decisiones del mantenimiento y servicios del vehículo a tiempo, además de contar con un diagnóstico y así poder realizar reparaciones y pruebas, reduciendo los costos. A largo plazo se pretende que el sistema sea una herramienta fundamental para reducir los índices de robos de automóviles, al tener la capacidad de enviar alertas al celular, estas alertas pueden ser por ejemplo si las puertas son abiertas, avisar al dueño del automóvil y evitar de esta manera el robo. Utilizando módulos BT de hasta 1 km de alcance.

2. ESTADO DEL ARTE
2.1. Arduino.
Arduino es una plataforma de desarrollo de computación física (physical computing) de código abierto, basada en una tarjeta con un microcontrolador y un entorno de desarrollo para crear software (programas). Arduino se utiliza para crear aplicaciones interactivas, obteniendo datos de una gran variedad de interruptores y sensores, para el control de diferentes dispositivos como: luces, motores y otros actuadores [3]. Los proyectos con Arduino pueden ser autónomos o comunicarse con un programa (software) que se ejecute en una computadora. El hardware está disponible para realizarlo uno mismo o se puede comprar ya listo para su uso, el software de desarrollo es abierto. El lenguaje de programación de Arduino es una implementación de Wiring, una plataforma de computación física, que a su vez se basa en Processing, un entorno de programación multimedia.

Figura 1. Tarjeta Arduino Uno.

Para la conexión con el automóvil se utilizó el dispositivo CAN-BUS, el cual permite conectar la tarjeta Arduino a la red CAN.

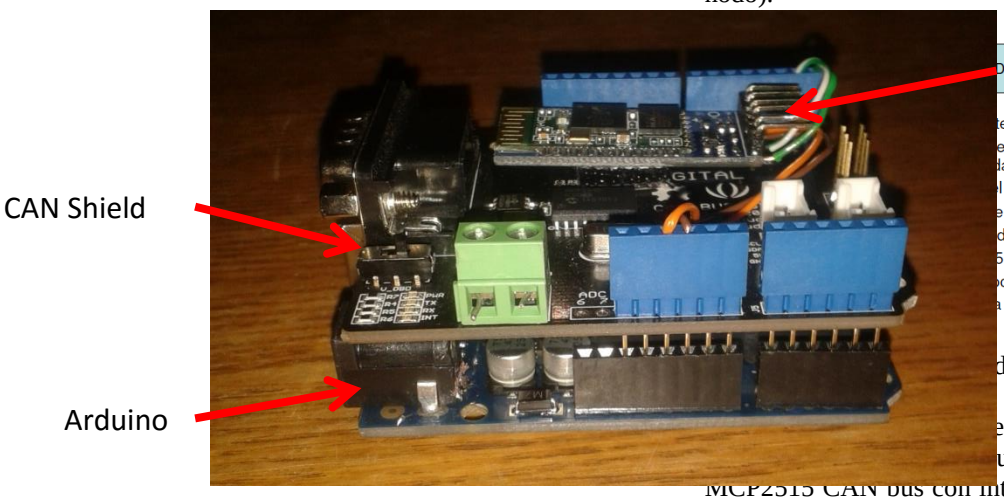


Figura 2. Módulo CAN-BUS.

2.2. CAN.

CAN es un protocolo de comunicación, desarrollado por la firma alemana Robert Bosch [4], basado en una topología de bus para la transmisión de mensajes en ambientes distribuidos, además ofrece una solución a la gestión de la comunicación entre múltiples unidades de proceso.

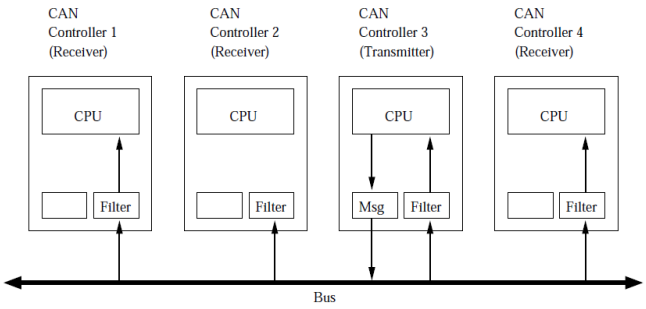
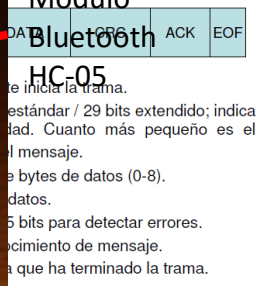


Figura 3. Arquitectura CAN.

El protocolo de comunicación CAN proporciona varios beneficios como por ejemplo que es un protocolo de comunicaciones normalizado, con el que se simplifica y economiza la tarea de comunicar subsistemas de diferentes fabricantes sobre una red común o bus. Las redes CAN están basadas en el mecanismo *broadcast* o difusión, lo que quiere decir, que lo que se emite por un nodo es escuchado por todos, decidiendo cada nodo si le es útil la información o no.

La información es transmitida mediante mensajes (fig. 3), cada mensaje se empaqueta en una trama o "*frame*". Existen cuatro tipos de mensajes distintos definidos por el protocolo: *Data Frame* (mensaje de información de datos), *Remote frame* (mensaje de petición de datos), *Error Frame* (mensaje que indica error) y *Overload Frame* (mensaje de sobrecarga de un nodo).



de un mensaje en un bus CAN.

e utilizó para la comunicación entre
utomóvil cuenta con un regulador
MCP2515 CAN bus con interfaz SPI (fig.5) y MCP2551 CAN transceptor.

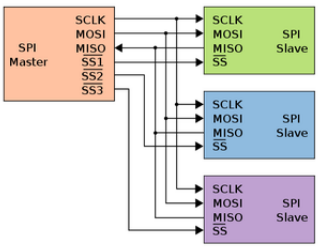


Figura 5. Comunicación SPI.

2.3. OBDII.

El sistema OBD II controla virtualmente todos los sistemas de control de emisiones y componentes que puedan afectar los gases de escape o emisiones evaporativas. Si un sistema o componente ocasiona que se supere el umbral máximo de emisiones o no opera dentro de las especificaciones del fabricante, un DTC (Diagnostic Trouble Code) debe ser almacenado. Un DTC es almacenado en la Memoria de Almacenamiento Activa (PCM Keep Alive Memory - KAM) cuando un mal funcionamiento es inicialmente detectado.

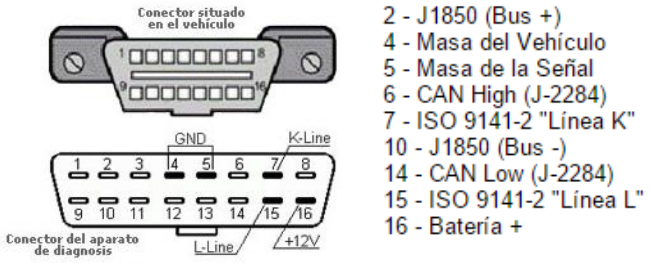


Figura 6. Descripción del conector OBDII.

El estándar SAE J2Q12 define un código de 5 dígitos en el cual cada dígito representa un valor predeterminado. Todos los códigos son presentados de igual forma para facilidad del mecánico. Algunos de estos son definidos por este estándar, y otros son reservados para uso de los fabricantes.

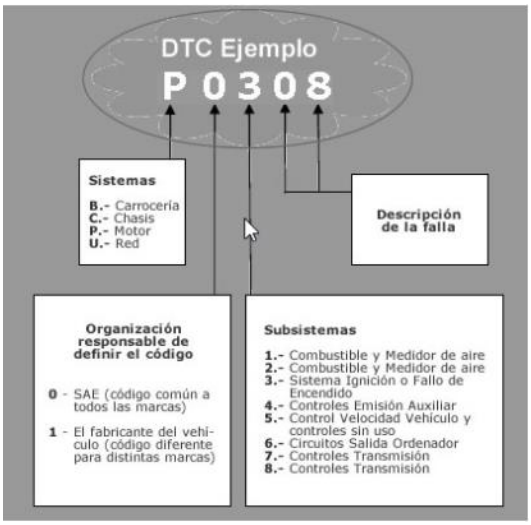


Figura 7. DTC (*Diagnostic Trouble Code*).

3. DESARROLLO.

Se diseñó e implementó un sistema de comunicación automotriz, tomando como base el protocolo de comunicación CAN. Se utiliza la tarjeta electrónica Arduino y la placa CAN-BUS como interface para obtener, decodificar y analizar los mensajes del Bus CAN del automóvil. Una vez que los mensajes son analizados, la información es enviada de manera inalámbrica a un teléfono celular por medio del protocolo de comunicación Bluetooth.

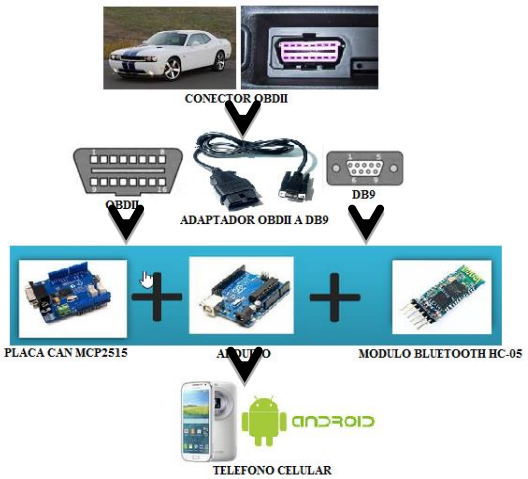


Figura 8. Sistema de monitoreo de un automóvil.

Se utiliza el OBD (*On-Board Diagnostics*) para establecer comunicación entre la red CAN del automóvil y la tarjeta CAN-BUS, esta tarjeta es la interface interpretando los mensajes arrojados por el automóvil (comunicación CAN), los cuales envía al Arduino (comunicación SPI). Los vehículos más actuales contienen el protocolo OBD y OBD-II, que especifica el tipo de conector.

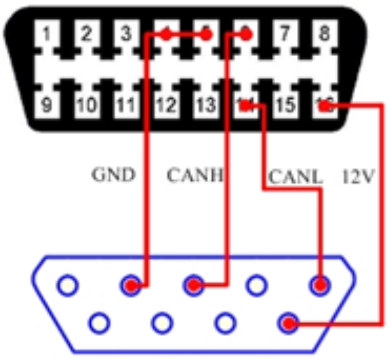


Figura 9. Conexión entre OBDII y la placa CAN-BUS.

Se desarrolló un programa en el lenguaje Arduino para procesar e identificar los mensajes recibidos de los diferentes dispositivos conectados a la red CAN del automóvil, posteriormente decodificarlos y enviar alertas al usuario o cliente del estado actual que guardan estos dispositivos.

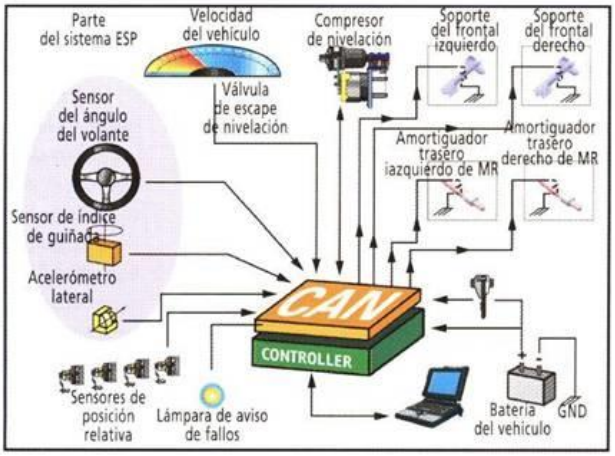


Figura 10. Estructura CAN del un automóvil.

En seguida se muestra un fragmento del código desarrollado para la decodificación de los mensajes:

```
void setup()
{
  CAN.begin(CAN_500KBPS);
  Serial.begin(115200);
  Bluetooth.begin(9600);
}
void MCP2515_ISR()
{
    Flag_Recv = 1;
}
void loop()
{
    if(Flag_Recv)
    {
      Flag_Recv = 0;
      CAN.readMsgBuf(&len, buf);
         if(buf[2]==12)
           {
             RPM = ((tres*256) + cuatro)/4;
              Serial.print("RPM=");
              Serial.println(RPM);
              Serial.print("\n");
              delay(50);
            }
    }
}
```

Figura 11. Segmento de código para interpretar las revoluciones del motor del automóvil.

Se implementó y desarrolló un programa en lenguaje Android para el teléfono celular utilizando la plataforma MIT APP Inventor, es una aplicación desarrollada por el Instituto Tecnológico de Massachusetts, que permite a los usuarios crear sus propias aplicaciones. A continuación se muestra parte del código diseñado para recibir los mensajes en el teléfono celular:

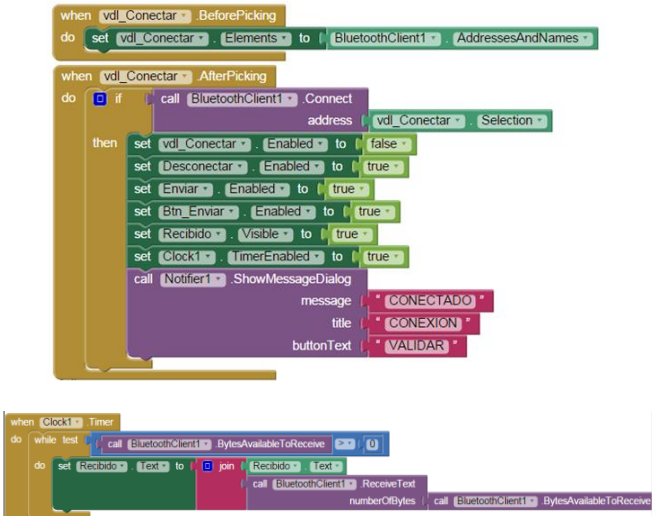


Figura 12. Código para el celular.

4. RESULTADOS

Primeramente se realizó una aplicación en el lenguaje LabView, auxiliándonos de una computadora y la tarjeta de *National Instruments* USB-8473, se pudo simular los datos o mensajes del bus del automóvil. Esto con el fin de validar y probar el sistema desarrollado con la tarjeta arduino.

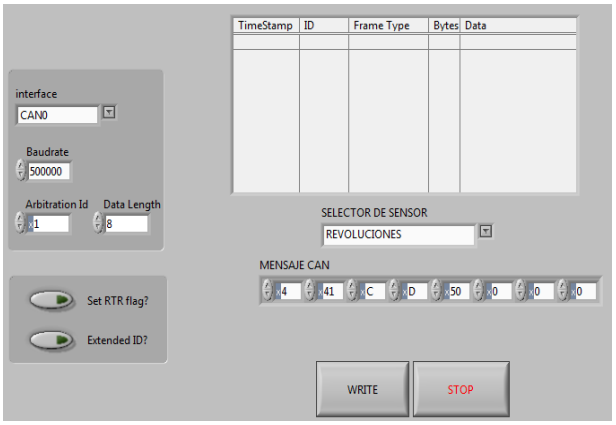


Figura 13.Panel Frontal de la Aplicación en LabView.

Apoyándonos de la aplicación se logró identificar y decodificar algunos mensajes tomados del bus CAN, ya que estos identificadores o mensajes son información reservada de las empresas automotrices, a continuación se muestra la siguiente tabla interpretada y que posterior fue utilizada por el sistema para enviar estos mensajes ya decodificados al celular.

TABLA 1. Identificadores de mensajes CAN.

ID	Descripción
C	Velocidad del Motor (RPM)
45	Posición del Acelerador (%)
10	Flujo de Aire (g/s)
D	Sensor de Velocidad
E	Sincronización de Avance PID (Spark Advance Deg.)
F	Temperatura de Aire (Deg C)
5	Temperatura del Refrigerante del Motor (°C)
4	Carga del Motor (%)
1F	Tiempo Transcurrido desde el Encendido (s)

Una vez que se comprobó el funcionamiento del sistema emulando mensajes con la computadora, se paso a conectar el sistema al automóvil, logrando obtener y decodificar algunos de los datos mencionados en la tabla anterior.

Figura14. Dispositivo conectado al automóvil.

Los datos obtenidos fueron analizados y enviados al teléfono celular para mostrarlos, comprobando el funcionamiento de todo el sistema y el buen rendimiento de la tarjeta Arduino conectada al bus CAN del automóvil.

Figura15. Pantalla del celular.

5. CONCLUSIONES

Uno de los objetivos principales del presente trabajo fue el de validar el comportamiento del la tarjeta Arduino y su interface CAN-BUS, al conectarlas dentro del bus CAN del automóvil, el cual maneja una gran información del estado del automóvil, la cual es obtenida de múltiples sensores y dispositivos conectados a la red. Se presenta de manera general el contexto y los requerimientos necesarios, para llevar a cabo la comunicación entre la red de comunicación CAN y un teléfono celular, utilizando como medio de apoyo la tarjeta electrónica Arduino, comprobando el buen desempeño de la tarjeta.

En resumen la aplicación monitorea la transferencia de datos en el bus CAN del automóvil y una de sus principales características es la de proveer información del estado actual del automóvil.

6. REFERENCIAS

[1] Sergio I. Hernández R., Pedro Ochoa M., and Julio C. Ramírez V. “Bluetooth and OPC (OLE for Process Control) for the Distributed Data Integration”. Fourth Congress of Electronics, Robotics and Automotive Mechanics. IEEE, 2007.

[2] Hernández Sergio, Tautímez Gustavo, Cruz Diego, Cruz Darío, Ramírez Julio, Ortiz Bertha, Macías Ricardo. “Comunicación de una Red de Automóvil CAN(Controller Area Network) con un Celular, mediante el Protocolo de Comunicación Inalámbrica Bluetooth”. Congr. Int. ing. Eltrón. Mem. Electro 2011. Vol. 33, pp. 102-107.

[3] Evans Brian, “Beginning Arduino Programming”, writing code for the most popular microcontroller board in the world, Technology in Action, 2011, ISBN: 978-1-4302-3778-5.

[4] Michiel van Osch, Scott A. Smolka, “Finite-State Analysis of the CAN Bus Protocol”, Proceedings of the 6th IEEE InternationalSymposium on High Assurence Systems Engineering (HASE’01), 2001.

[5] Lajara José, Pelegrí José, “LabVIEW, Entorno Gráfico de Programación”, Ed. Alfaomega, 2007, ISBN: 978-970-15-1133-6

[6] Herrera Enrique, “Tecnologías y Redes de Transmisión de Datos”, Ed. Limusa, 2003, ISBN: 968-18-6383-6.