UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

INGENIERÍA EN MANTENIMIENTO AUTOMOTRIZ

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO AUTOMOTRIZ

TEMA: DESARROLLO DE UN DISPOSITIVO BASADO EN MICROCONTROLADOR ARDUINO PARA LA ADQUISICIÓN DE DATOS A TRAVÉS DE LA RED CAN PARA CONOCER LAS CONDICIONES QUE SUSCITAN ACCIDENTES VEHICULARES.

AUTORES:

PUGA GUTIERREZ ANTHONY MOISES

MORALES RECALDE JOSELYN STEFANY

DIRECTOR: ING. ANDRÉS FELIPE CEVALLOS GONZALEZ, MSc.

Ibarra, 2022

CERTIFICADO

ACEPTACIÓN DEL DIRECTOR

En mi calidad de director del plan de trabajo de grado, previo a la obtención del título de Ingeniería Automotriz, nombrado por el Honorable Consejo Directivo de la Facultad de Ingeniería en Ciencias Aplicadas.

CERTIFICO:

Que una vez analizado el plan de grado cuyo título es “DESARROLLO DE UN DISPOSITIVO BASADO EN MICROCONTROLADOR ARDUINO PARA LA ADQUISICIÓN DE DATOS A TRAVÉS DE LA RED CAN PARA CONOCER LAS CONDICIONES QUE SUSCITAN ACCIDENTES VEHICULARES”
presentado por el señor: Puga Gutiérrez Anthony Moisés con número de cédula 0503782880 y la señorita: Morales Recalde Joselyn Stefany con número de cédula 1004656250, doy fe que dicho trabajo reúne los requisitos y méritos suficientes para ser sometido a presentación pública y evaluación por parte de los señores integrantes del jurado examinador que se designe.

En la ciudad de Ibarra, a los 22 días del mes de marzo del 2022.

Atentamente

Firmado electrónicamente por:

ANDRES FELIPE CEVALLOS GONZALEZ

Ing. Andrés Felipe Cevallos González,MSc.

DIRECTOR DEL TRABAJO DE GRADO

UNIVERSIDAD TÉCNICA DEL NORTE
BIBLIOTECA UNIVERSITARIA
AUTORIZACIÓN DE USO Y PUBLICACIÓN
A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO	
CEDULA DE IDENTIDAD:	050378288-0
APELLIDOS Y NOMBRES:	Puga Gutiérrez Anthony Moisés
DIRECCIÓN:	Ibarra, Av. Virginia Pérez y Marco Nicolalde, Edificio 8-121
EMAIL:	ampugag@utn.edu.ec
TELÉFONO MÓVIL	0983724159

DATOS DE CONTACTO	
CEDULA DE IDENTIDAD:	100465625-0
APELLIDOS Y NOMBRES:	Morales Recalde Joselyn Stefany
DIRECCIÓN:	Ibarra, Av. Eugenio Espejo (Puente Amarillo)
EMAIL:	jsmoralesr@utn.edu.ec
TELÉFONO MÓVIL	0981437690

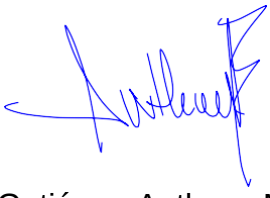
DATOS DE LA OBRA	
TÍTULO:	**TEMA: DESARROLLO DE UN DISPOSITIVO BASADO EN MICROCONTROLADOR ARDUINO PARA LA ADQUISICIÓN DE DATOS A TRAVÉS DE LA RED CAN PARA CONOCER LAS CONDICIONES QUE SUSCITAN ACCIDENTES VEHICULARES.**
AUTORES:	Puga Gutiérrez Anthony Moisés Morales Recalde Joselyn Stefany
FECHA:	22/03/2022
SOLO PARA TRABAJOS DE GRADO	
PROGRAMA:	PREGRADO
TITULO POR EL QUE OPTA	INGENIERÍA AUTOMOTRIZ
ASESOR/DIRECTOR	Ing. Andrés Felipe Cevallos González, MSc.

2. CONSTANCIAS

Los autores manifiestan que la obra objeto de la presente autorización es original y se la desarrollo, sin violar derechos del autor de terceros, por lo tanto, la obra es original y que son las titulares de los derechos patrimoniales, por lo que asumen la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 22 días del mes de marzo del 2021

AUTORES:



Puga Gutiérrez Anthony Moisés
050378288-0

Morales Recalde Joselyn Stefany
100465625-0

DEDICATORIA

Esta tesis está dedicada a mi mamá quién me animó en este campo de estudio y a la cual admiro mucho por su inteligencia y astucia.

A mi papá quien es mi ejemplo para seguir como profesional, pues realizarme como tal siempre fue su objetivo, él es sin duda para mí la verdadera definición de ser un ingeniero.

A mi compañera de proyecto y mejor amiga Stefy ya que sin su ayuda y trabajo en equipo no hubiese sido posible.

A mi buen amigo Albaro el cual me dio la carrera y siempre compartió conmigo tristezas y alegrías.

A mis hermanos que son las personas que más quiero.

Sobre todo, a esa personita que hace que mis días sean más alegres, por ser mi cómplice, mi confidente, por ayudarme a crecer, por amarme y siempre motivarme a cumplir mis sueños, mi corazón te pertenece.

Puga Gutiérrez Anthony Moisés

Este trabajo de grado es dedicado a mi madre quién siempre me cuido y apoyo durante este transcurso de mi vida, por inculcarme valores de respeto, humildad y principalmente de responsabilidad con la finalidad de cumplir mis metas planteadas, así mismo a mis abuelitos quienes con su amor y cariño me incentivaban a culminar esta formación académica y finalmente a mi pequeño hermano José Andrés quien me mira con admiración y un ejemplo a seguir.

Morales Recalde Joselyn Stefany

AGRADECIMIENTOS

En primer lugar y de manera especial agradezco a mi papá William, por todo el esfuerzo y sacrificio que hizo para educarme como profesional, por todo el apoyo, consejos y tiempo, que dedicó para escucharme y enseñarme cada día. Por ser un buen padre conmigo y siempre estar dispuesto a ayudarme sin dudarlo y por su amor infinito.

A mi mamá María y a Luis por enseñarme de la vida y hacer de mí la persona que soy ahora, por los valores y principios que me inculcaron desde niño y por todo el esfuerzo que mi mamá invirtió en mi educación para prepararme para esta etapa universitaria.

A mi querida abuelita Piedad porque siempre estuvo dispuesta a ayudarme y aconsejarme.

A mis hermanos William, Renata y Luis por todo su apoyo en cada momento de mi vida.

A mi director de tesis el ingeniero Andrés por todo lo que me enseñó en cada etapa de mi carrera universitaria, además del tiempo y paciencia que dedicó durante el desarrollo de este proyecto de grado.

Finalmente, a todos mis profesores que contribuyeron en mi formación académica como profesional, por ser unos excelentes seres humanos, amigos y profesionales.

Puga Gutiérrez Anthony Moisés

En primer lugar, quiero agradecer a mi madre Jhovana, quién con mucho esfuerzo y sacrificio logro brindarme la educación necesaria para convertirme en una profesional, así mismo agradezco a mi padrastro Andrés quien a pesar de no ser mi padre siempre me apoyo económica y moralmente.

A mis abuelitos Flora y Luis, por siempre estar pendientes de mí, aconsejarme y ayudarme a resolver los dilemas que se suscitan en el transcurso de la vida.

A mi mejor amigo y compañero Moisés, por su apoyo moral durante esta etapa universitaria y el trabajo en equipo que hicimos para culminar este trabajo de grado. Con el objetivo de graduarnos juntos.

A mis profesores y mi director de tesis el Ing. Andrés, quienes con sus conocimientos brindados ayudaron a formarme en esta rama profesional.

Y finalmente me agradezco a mí, por el esfuerzo y sacrificio que llevo formarme como profesional.

Morales Recalde Joselyn Stefany

INDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ANEXOS

ANEXO 1

ÍNDICE DE ECUACIONES

RESUMEN

En el presente trabajo de grado se plantea el desarrollo de un dispositivo basado en Arduino, con la finalidad de adquirir los datos de la red CAN para conocer las condiciones que suscitan accidentes vehiculares, mediante una lectura inicial del flujo de datos a los módulos que conforman la Unidad Electrónica de Control (ECU) por medio de un escáner automotriz, siendo los módulos ENGINE Y CHASIS CONTROL los que contengan variables referente a la seguridad del vehículo, haciendo una previa selección de las variables que cumplan con los requerimientos del estudio, como son: WHEEL SPEED (Velocidad de las ruedas), CAL/LD VALUE (Valor calculado de la carga), BRAKE SWITCH (Switch del freno), ABS MALF (Estado del sistema ABS), SIDE AND DECEL G SENSOR (Aceleraciones transversales y longitudinales), YAW RATE SENSOR (Sensor giro o guiñada), VEHICLE SPEED (Sensor VSS), STEERING ANG SENSOR (Ángulo de dirección) y ENGINE SPEED (Revoluciones del motor). De tal manera por medio del CAN Analyzer y juntamente con el escáner se extraen los datos denominados tramas de información y dentro de cada trama se encuentran los PIDs, que son las solicitudes y respuestas que se usa para la comunicación del microcontrolador con la red CAN del vehículo, de cada una de las variables sea estándar o fabricante.

Para la extracción de los PIDs estándar se genera una comunicación entre la red CAN del vehículo y las herramientas: un software denominado Hyperterminal, el analizador CAN y el escáner. Donde se examinan las tramas de información, analizando cuales son las solicitudes y respuestas de cada una de las variables seleccionadas netamente estándar, mientras que la extracción de los PIDs fabricante se usa un grabador de datos “DATA LOG”, que guarda un flujo de datos en formato CSV siendo estos filtrados para posteriormente ser analizados, determinando las solicitudes para ingresar a cada una de las variables requeridas y a su vez adquirir las respuestas de estas variables. Estos datos muestran un formato hexadecimal, siendo difícil ser interpretados, de tal manera de realiza un proceso de decodificación. Mediante la identificación de valores picos de las gráficas que ofrece el escáner capturando datos en tiempo real, donde se desarrolla fórmulas que conviertan estos datos hexadecimales a decimales, permitiendo que la deducción e interpretación de los datos sea sencilla.

El desarrollo de una interfaz permite la comunicación entre el microcontrolador y la red CAN por medio de la interconexión de módulos como: el CAN Bus Shield, Arduino Mega

y entre otros. Con la finalidad de generar la lectura, escritura y almacenamiento de los datos de la red CAN en una tarjeta SD, el proceso de registro y almacenamiento de datos se lleva a cabo siempre y cuando se supere un umbral de valores referente a las aceleraciones trasversales y longitudinales, cuando el vehículo entre en riesgo de una posible colisión los datos se almacenaran para su posterior lectura.

ABSTRACT

In this degree work we propose the development of a device based on Arduino, in order to acquire data from the CAN network to know the conditions that cause vehicle accidents, through an initial reading of the data flow to the modules that make up the Electronic Control Unit (ECU) through an automotive scanner, being the ENGINE and CHASSIS CONTROL modules that contain variables relating to vehicle safety, making a previous selection of the variables that meet the requirements of the study, such as: WHEEL SPEED (wheel speed), CAL/LD VALUE (calculated load value), BRAKE SWITCH (brake switch), ABS MALF (ABS system status), SIDE AND DECEL G SENSOR (transverse and longitudinal accelerations), YAW RATE SENSOR (yaw sensor), VEHICLE SPEED (VSS sensor), STEERING ANG SENSOR (steering angle) and ENGINE SPEED (engine speed). In this way, by means of the CAN Analyzer and together with the scanner, the data called information frames are extracted and within each frame there are the PIDs, which are the requests and responses used for the communication of the microcontroller with the CAN network of the vehicle, of each of the variables, whether standard or manufacturer.

For the extraction of the standard PIDs, a communication is generated between the CAN network of the vehicle and the tools: a software called Hyperterminal, the CAN analyzer and the scanner. Where the information frames are examined, analyzing which are the requests and responses of each of the selected variables purely standard, while the extraction of the PIDs manufacturer uses a data recorder "DATA LOG", which saves a data stream in CSV format being these filtered for later analysis, determining the requests to enter each of the required variables and in turn acquire the responses of these variables. These data show a hexadecimal format, being difficult to be interpreted, so a decoding process is performed. Through the identification of peak values of the graphs offered by the scanner capturing data in real time, where formulas are developed to convert these hexadecimal data to decimals, allowing the deduction and interpretation of the data to be simple.

The development of an interface allows communication between the microcontroller and the CAN network through the interconnection of modules such as: CAN Bus Shield, Arduino Mega and others. In order to generate the reading, writing and storage of data from the CAN network on an SD card, the process of recording and storing data is carried out as long as a threshold of values relating to transverse and longitudinal accelerations is exceeded, when the vehicle is at risk of a possible collision the data will be stored for later reading.

INTRODUCCIÓN

La presente investigación aborda temas referentes a seguridad vehicular, aprovechando las características propias de la red multiplexada CAN misma que llevan integrada los vehículos más recientes. Se pretende solicitar, recibir y almacenar variables de interés disponibles en la red que podrían resultar útiles para un análisis en caso de un accidente vehicular. Para ello, se proponen métodos innovadores para la adquisición de estas variables utilizando técnicas como el CAN hacking y el análisis del flujo de datos propios de cada computadora que forme parte de la red. Mediante un microcontrolador programable interconectado a la red.

Las variables consideradas importantes para adquirirlas se seleccionan basándose en las características y funcionamiento de los sistemas de seguridad activa y pasiva propios del automóvil. Es importante hoy en día disponer de varios métodos para el análisis de accidentes vehiculares, a veces la información que se recaba después de un accidente vehicular es limitada, resulta útil un dispositivo que guarde la información de los valores de los distintos sensores que dispone el vehículo como velocidad, aceleración longitudinal, aceleración transversal, entre otros.

CAPÍTULO I

1 REVISIÓN BIBLIOGRÁFICA

1.1 OBJETIVOS

1.1.1 Objetivo general

Desarrollar un dispositivo basado en microcontrolador Arduino para la adquisición de datos a través de la red CAN para conocer las condiciones que suscitan accidentes vehiculares.

1.1.2 Objetivos específicos

- Evaluar los parámetros que intervienen en los sistemas de seguridad activa y pasiva del vehículo, con el objetivo de establecer variables de interés.
- Analizar por medio de un dispositivo de diagnóstico automotriz las variables de interés del flujo de datos de los módulos automotrices.
- Identificar los códigos de requerimiento para la extracción de las variables analizadas a través de la red CAN.
- Desarrollar una interfaz de comunicación con el microcontrolador Arduino utilizando valores críticos que pueden intervenir en una posible colisión, analizando del comportamiento de la red CAN, mediante la ejecución de pruebas dinámicas.
- Ejecutar pruebas de funcionamiento para evaluar la fiabilidad del prototipo.

1.2 JUSTIFICACIÓN

El ser humano generalmente es el responsable de controlar la actividad del vehículo durante la conducción, pues es él, quien decide optar por el cumplimiento de las normas o leyes de tránsito (García & Coello, 2017). Evidentemente, cuando estas normas se pasan por alto, ocurren accidentes de tránsito y es necesario determinar las causas mediante peritajes. En un peritaje de tránsito se parte de la idea de que un accidente de tránsito es un suceso no

planeado y no deseado (Perez, 2017). Sin embargo, en ocasiones resulta complicado determinar esta afirmación.

Con un dispositivo que guarde y provea un registro interpretable de las variables de interés del flujo de datos de las computadoras del vehículo antes, durante y después de la colisión, se puede obtener de manera precisa parámetros de importancia para un informe pericial como: velocidad del vehículo, velocidad del motor, posición del pedal de acelerador, tiempo de frenado, aceleración longitudinal y transversal del vehículo, datos del sistema EPS (Dirección Asistida Eléctricamente), entre otros.

Esta información sería precisa y útil, además ayudaría en el desarrollo del informe pericial tipo “C” y “B” del SIAT, infiriendo de una manera más sencilla en las causas del accidente. Adicionalmente, los peritos de tránsito podrían sustentar en base a evidencia contundente los resultados de sus peritajes y así contribuir a solventar las dudas generadas con respecto a sus informes como lo establece el artículo 505 del Código Orgánico Integral Penal (COIP) el cual plantea que “Los peritos sustentarán oralmente los resultados de sus peritajes y responderán al interrogatorio y al contrainterrogatorio de los sujetos procesales.”

El factor psicológico, también puede inferir en la causa de un accidente vehicular, al momento que un conductor es consiente de ser observado modifica su comportamiento. (Wouters & Bos, 2000).Una vez implementando dispositivos EDR en los vehículos, se puede hacer uso de este fenómeno psicológico para controlar al conductor y que su conducción sea más segura, con la finalidad de minimizar el porcentaje de accidentabilidad.

1.3 ALCANCE

El presento proyecto propone la realización de un dispositivo de monitoreo basado en microcontrolador Arduino para la adquisición de datos a través de la red CAN para conocer las condiciones que suscitan accidentes vehiculares. La primera instancia del proyecto es analizar las condiciones en las que se considera que un vehículo está en riesgo de colisión como son: las fuerzas g en el momento de desaceleración brusca y la detección de impacto con sensores especializados.

Una vez estimada esta información, el siguiente punto consiste en inferir en la red multiplexada de un vehículo para extraer los códigos referentes a parámetros de funcionamiento de este y especialmente relacionados a causas de accidentabilidad, por

ejemplo: velocidad, aceleración longitudinal, aceleración transversal, torque de giro, posición del volante, activación de freno, estado de sistema de tracción, entre otros; dependiendo de la tecnología existente en el vehículo de prueba.

La parte final del proyecto consiste en desarrollar un dispositivo en base al microcontrolador Arduino que sea capaz de detectar las condiciones de un accidente, comunicarse a través de la red CAN con el vehículo, para extraer los datos que se consideran importantes en un siniestro y almacenarlos para posterior peritaje.

1.4 ANTECEDENTES

De acuerdo con European Commission of Mobility and Trasport, los registradores de datos en el momento de suscitarse un accidente han sido utilizados durante varios años en vehículos de transporte comercial. (*Black boxes/ in-vehicle data recorders | Mobility and transport*, s/f). Los dispositivos se basan en el módulo de control del airbag y almacenan información hasta que las bolsas de aire detonen. Estos dispositivos ya han sido empleados anteriormente, no son de carácter obligatorio.

A partir de 1970, el fabricante de automóviles GM ha hecho uso de registradores de datos de accidentes vehiculares, con el fin de evaluar el rendimiento de las bolsas de aire al momento de suscitarse un choque. En el caso de Ford y Volvo desde la década de los 90 se han empleado registradores de datos de accidentes, pero solo en automóviles tecnológicamente avanzados. (Lie et al., 2006)

Según un estudio realizado por el Instituto SWOV para la investigación de seguridad vial, las personas que son observadas tienden a cambiar su comportamiento ante diversas situaciones. (Wouters & Bos, 2000) concluye que, los registradores de datos de accidentes instalados en camiones y furgonetas conducen a una reducción media del 20% en el número de choques y daños. Esto se debe a que el conductor de cierta manera está consciente de que el vehículo está monitorizado y se puede determinar la causa del accidente por medio del dispositivo.

Generalmente existen métodos para la reconstrucción y análisis de escenarios que suscitaron un accidente de tránsito y uno de ellos es calcular la velocidad de energía tradicional mediante una estimación de la desaceleración debido al frenado producido en el vehículo. Además, se considera el 50 % del factor de resistencia total de la carretera por la

desaceleración producida, mediante el uso del frenado y la velocidad antepuesta al impacto por hora, con la finalidad de aplicarla en una fórmula que reúne las velocidades calculadas antes y después del impacto. De tal manera con base a esos cálculos se determina si la velocidad es o no, un factor importante para que se suscite un accidente. (Andrews & Limpert, 2013).

En base a las estadísticas de accidentes vehiculares de la Agencial Nacional de Transito, 4006 fueron el número de siniestros por exceso de velocidad, 370 sucesos por daños mecánicos y 11823 por imprudencia e impericia del conductor. Las causas de estos eventos son deducidas en base a peritajes con métodos tradicionales en función de la reconstrucción de los hechos. (*Estadísticas siniestros de tránsito – Agencia Nacional de Tránsito del Ecuador – ANT*, 2021)

1.5 PLANTEAMIENTO DEL PROBLEMA

En Ecuador actualmente, no existe una normativa que obligue a los fabricantes automotrices a emplear dispositivos EDR (Registrador de datos de eventos) para analizar las causas de los accidentes vehiculares lo cual es un problema, ya que, en ocasiones las causas que suscitan un accidente vehicular son difíciles de estudiar y determinar, dada la naturaleza misma del suceso. Aunque en algunos países se ha estado trabajando en normativas para la implementación de este tipo de dispositivos, aún no existe un avance significativo al respecto. Por otro lado, está el hecho de que los fabricantes automotrices solo incluyen los EDR en vehículos de alta gama y de manera totalmente voluntaria, además los métodos de extracción de los datos es una tarea complicada, ya que, se requiere hacer uso de dispositivos CDR (Recuperador de datos de accidentes) o simplemente los dispositivos no están universalizados.

Sin embargo, recientemente se implementó el uso obligatorio de tacógrafos encargados de registrar datos acerca de la marcha del vehículo y determinadas actividades del conductor como: kilómetros recorridos, velocidad, la hora y los periodos de trabajo y descanso. Pero este tipo de dispositivos son instalados solo en vehículos que superen los 3.500 kg o que pueden transportar a más de 9 personas. Además, no almacenan datos que se consideren importantes para determinar las causas que conllevaron a un accidente de tránsito.

En el país en materia de tránsito, mediante informes periciales se puede entender las circunstancias y consecuencias del accidente (Perez, 2017). Un informe pericial es un conjunto de enunciados verificados, partiendo previamente de las hipótesis de los peritos, que por medio de información colectada del entorno donde tuvo lugar el evento, así como de elementos técnicos empleados por especialistas, son verificadas o desmentidas (Racines & López, 2016). Mientras mayor sea la información obtenida en el lugar de los hechos, mayor será el aporte para la determinación de causas del accidente, sin embargo, a veces la información es escasa o indeterminable aumentando la dificultad para llegar a una conclusión concreta de lo que ocurrió. Además, hay que tomar en cuenta que un peritaje es una investigación humana, misma que puede estar sujeta errores por su naturaleza.

Por medio del presente proyecto se pretende resolver la problemática mencionada anteriormente, mediante el desarrollo de un dispositivo basado en microcontrolador Arduino, para adquirir los datos de la red CAN como velocidad, aceleración, posición del volante, estado del freno y sistemas de tracción. Con la finalidad de conocer las condiciones que provocaron el accidente vehicular.

1.6 SITUACIÓN ACTUAL

Según (Andrews & Limpert, 2013), existen empresas y marcas vehiculares que implementan dispositivos para el almacenamiento de datos como Detroit, Cummins o Mack, sin embargo, estos dispositivos solo son aplicados para vehículos de clase N, cuya masa sea superior a 3.5 toneladas e inferior a 12 toneladas, fabricados para el transporte de mercancias. Mientras que no se evidencia la aplicación de estos dispositivos para vehiculos de clase M1.

Los flujos de datos que se encuentran en un dispositivo de registro de datos, se logra recuperar mediante una herramienta de recuperación CDR (Crash Data Retrieval), siendo compleja la extracción de datos. Especialmente en vehículos antiguos la recuperación de estos datos es difícil de interpretar sin un software adecuado, mientras que en vehículos nuevos se requiere el uso de códigos de acceso. Además, en países como Estados Unidos sin una orden judicial, no se permite el acceso al flujo de datos del vehículo. (*What is an Automobile Black Box | ExpertLaw*, s/f)

Según el sitio web de la consultora (*EU funded projects - Veronica I+II - Squaris s.p.r.l.*, s/f), en 2004 se creó un proyecto denominado VERONICA I (Registro de eventos de

vehículos basado en evaluación inteligente de accidentes) el cual era financiado con fondos comunitarios donde se buscaba definir aspectos técnicos, legales y preventivos para la posible introducción obligatoria de dispositivos de grabación de datos, de esta forma se establecerían normas necesarias, requisitos técnicos y legales para el uso de dispositivos registradores de datos de eventos (EDR). El proyecto reunió varios fabricantes automotrices, así como, compañías de seguros, expertos en peritajes de accidentes, entre otros.

1.7 SISTEMA DE DIAGNOSTICO A BORDO OBDII

1.7.1 Historia

El sistema de diagnóstico a bordo OBD, se desarrolla con la finalidad de controlar las emisiones de gases de escape emitidas por vehículos de combustión, tratándose de un sistema de diagnóstico integrado en la gestión del motor, con la finalidad de vigilar contantemente cada uno de los componentes que intervienen en las emisiones de gases de escape. Denominada una estandarización, fue trabajada conjuntamente por gobiernos y entidades gubernamentales preocupadas por reducir las emisiones de CO2. Tras varios años de investigación entre SAE (Society of Automotive Engineer), CARB (The California Air Resource Board) y EPA (Eviromental Protection Agency) en la década de los 90, se estableció la nueva generación del sistema de diagnóstico a bordo OBD II y a partir del año de 1996 la norma fue adoptada por Estados Unidos de América y a partir de ese año todos los vehículos vendidos tenían que estar equipados con OBDII. (César & López, 2014).

La nueva generación del sistema de diagnóstico a bordo OBDII incorporo un conector de diagnóstico y la ubicación de este, dentro del habitáculo del vehículo. Este estándar definió cuales son las partes del motor que deben ser monitoreadas constantemente y bajo que parámetros. En el caso de detectarse una falla en uno de los sensores, por medio de una luz indicadora de un funcionamiento anormal denomina MIL (Malfunction Indicator Light) ubicada en el tablero del vehículo, serviría como medio informativo sobre la existencia de fallas en uno de los sensores. Permitiendo identificar y solucionar el problema en el menor tiempo posible. (César & López, 2014). Así mismo, implemento un protocolo de comunicación entre la ECU (Unidad de Control del Motor o Engine Control Unit), por medio de la estandarización de este protocolo de comunicación se puede diagnosticar e identificar fallas en el vehículo sin necesidad de que la herramienta sea propia del fabricante, controlando rigurosamente los niveles emitidos por los gases de escape cumpliendo con las reglas establecidas por EPA. Todos los vehículos que se encuentran equipados con sistema

OBDII, poseen una calcomanía que indica la información del control de emisiones y esta se encuentra ubicada en la parte inferior del capó del automóvil.

1.7.2 Funcionamiento y detección de fallas del sistema de diagnóstico a bordo OBDII.

El sistema OBDII permite monitorear condiciones o posibles fallas que se suscitan en los sensores del vehículo creando una secuencia rutinaria mientras este es conducido. En el caso de arrojar una falla en los resultados de la secuencia, se visualizará la luz MIL en el tablero del automóvil e inmediatamente se generará un código de falla y se almacenará en la memoria de la computadora (ECU). Además, el código de falla manteniente un formato donde indica en que parte del vehículo ocurrió la falla detectada y en qué condiciones se suscitó. (Villamar Aguirre, 2008). Todo este proceso se lleva a cabo mediante un conector estándar que cuenta el sistema OBDII, denominado como DLC (Data Link Connector o J1962), siendo un conector con 16 pines y cada uno de ellos son asignados en función específica para cada protocolo, así mismo los códigos de falla se pueden visualizar y diagnosticar por medio de un scanner automotriz.

Por medio de la conexión de la ECU y un scanner automotriz podemos verificar una serie de parámetros que determinan el funcionamiento del motor: velocidad, carga, temperatura del motor, consumo de combustible, temperatura ambiente. Caudal de aire y emisiones.

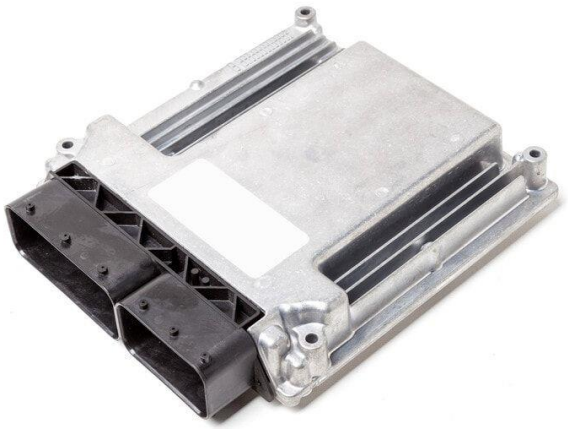


Figura 1.1 Unidad de Control Electrónico del Motor

(S.A, 2021)

En el caso de detectar funcionamientos erróneos en los sistemas de control de emisiones y componentes, por parte del sistema OBD II. El DTC es almacenado en la memoria KAM (Memoria de Almacenamiento Activa) y en el mayor de los casos la luz MIL es iluminada luego de dos ciclos consecutivos en los que estuvo presente la posible falla, una vez realizado este proceso debe transcurrir tres ciclos para que la luz MIL se apague. El DTC será borrado de la memoria KAM, al momento de finalizar los 40 ciclos de arranque y calentamiento del motor. (Villamar Aguirre, 2008).

1.7.3 Modos de medición OBDII

El sistema de detección a bordo OBDII tiene varios modos de medición que ejecutan diversas funciones del sistema y estos vienen implementados por diversos protocolos según sea la información que deseamos acceder, provocando una mayor estructuración y orden del sistema. Dentro de cada uno de estos modos de medición se puede usar varios parámetros permitiendo registrar datos para la verificación de este, extraer códigos de averías, borrar los mismos códigos y realizar pruebas dinámicas de actuadores. (Jorge Sánchez Carrizo, 2017).

Los modos de medición aparte de encargarse del registro de datos; envían la información registrada al software de diagnóstico correspondiente, con el fin de interpretar y representar estos datos correctamente, permitiendo facilitar el diagnóstico de la falla ocurrida en el vehículo. Estos modos de medición están compuestos de un formato de 7 bytes, donde cada byte registra diferente contenido dependiendo si el concepto sea de petición o respuesta y en cual modo se está trabajando. (Jorge Sánchez Carrizo, 2017).Con una excepción del byte 0, el cual indica el número de bytes que contiene información del sistema OBDII, además la existencia de este byte dependerá del protocolo en donde se envié el mensaje. En el caso del protocolo CAN es el único con la capacidad de sobrellevar 8 bytes de información a diferencia de estos protocolos que solo soportan 7 bytes.

1.7.3.1 Modo 01: Obtención de datos de diagnóstico actualizados

El modo 1 también denominado flujo de datos, se encarga de acceder a los datos del sistema en tiempo real, considerando valores tanto analógicos como digitales ya sea de salida o entrada de la unidad de control electrónica del motor (ECU). Permitiendo monitorizar la

temperatura del motor, los voltajes generados, sensores y actuadores. (Dimaté Cáceres & González Castillo, 2013).

Cuando se trabaja por el modo 1, se tiene la capacidad de solicitar peticiones de diversos PID es una sola petición, estos son denominado como (Parámetros ID), encargados de indicar al sistema de diagnóstico a bordo OBDII la información que se necesita, los PIDs son códigos hexadecimales interpretados por las Unidades Electrónicas de Control mismas que estas responden otros códigos hexadecimales que posteriormente son convertidos a un valor decimal utilizando una formula por la herramienta de diagnóstico. En el caso de tener un PID $20, indica los PIDs soportados desde el PID $01 al $20, mientras que si tenemos un PID $20 muestra los PIDS soportados entre el PID $21 y el $40. Al presente se tiene una gran variada de PIDs pero se pueden abreviar según el protocolo en el cual se esté trabajando. En el Protocolo SAE J1970 establece que todos los PIDs y sus modificaciones deben estar aglomerados en la norma OBDII ya sean PIDs genéricos o de fabricante. En la siguiente tabla se presenta los PIDS principales para OBDII (Jorge Sánchez Carrizo, 2017).

PID	Bytes	Descripción	Mínimo	Máximo	Unidades	Fórmula
00	4	PIDs soportados [01-20].	-	-	-	32 Bits: BitX=0>PIDX Sop.
01	4	Estado de MIL y número de DTCs almacenados.	-	-	-	Bits codificados
03	2	Estado del sistema de combustible.	-	-	-	Bits codificados
04	1	Valor de carga del motor.	0	100	%	A*100/255
05	1	T° del refrigerante.	-40	215	°C	A-40
0A	1	Presión del combustible.	0	765	kPa	A*3

0C	2	Revoluciones por minuto del motor.	0	16.383	Rpm	((A*256)+B)/4
OD	1	Temperatura del aire de entrada.	0	255	Km/h	A
OF	1	Temperatura del aire de entrada	-40	215	°C	A-40
11	1	Posición del acelerador.	0	100	%	A*100/255
13-1B	2	Voltajes de distintos sensores de oxígeno	0	1.275	V	A/200(b-128)*100/128
1C	1	Estándar OBD conforme al vehículo.	-	-	-	Bits codificados
1F	2	Tiempo trascurrido desde el arranque.	0	65.535	Seg	(A*256)+B
20	4	PIDs soportados [21-40].	-	-	-	32 Bits: BitX=0>PIDX Sop. Bit=1 > PIDX No sop.
21	2	Distancia recorrida con el indicador MIL encendidos.	0	55.535	Km	(A*256)+B
21 – 2B	4	Voltaje equivalente de sonda lambda.	0	8	V	((A*256)+B)*R ((C*256)+D)*R Con R=2/65535

2C	1	Recirculación de gases de escape o EGR.	0			A*100/255
2E	1	Depuración por evaporación.	0			A*100/255
2F	1	Niveles de combustible.	0			A*100/255
31	2	Distancia recorrida desde la última limpieza de errores.	0			(A*256)+B
34 - 3B	4	Corriente equivalente de la sonda lambda.	-128			((A*256)+B)/32.768 ((C*256)+D)/128
3C – 3F	2	Temperatura del catalizador.	-40			((A*256)+B)/10-40
40	4	PIDs soportados [41-60].	-			32Bits: BitX=0>PIDX Sop. BitX=1 > PIDX No sop.
42	2	Modulo controlador del voltaje.	0			(A*256)+B
46	1	T° Ambiente	-40	215	°C	A-40
4D	2	Tiempo en marcha mientras el MIL permanece encendido.	0	65.535	Min	(A*256)+B
51	1	Tipo de combustible.	-	-	-	Bits codificados

5B	1	Nivel de batería híbrida.	0	100	%	A*100/255

Tabla 1.1 PIDs estándar para OBDII modo 01.

(Jorge Sánchez Carrizo, 2017)

1.7.3.2 Modo 02: Acceso a datos congelados.

El modo de acceso al cuadro de datos congelados es considerado como el más importante, debido a que la ECU toma una muestra de todos los valores que sen encuentra relacionados con las emisiones, cuando se produce un fallo en el sistema. Permitiendo recoger los datos y analizar las condiciones exactas que provocaron el fallo. El modo 01 y modo 01 se manejan con el mismo funcionamiento de PIDs, con la única diferencia que el modo 01 registra y controla los datos en tiempo real mientras que el modo 01 solo los datos almacenados.

Modo (hex)	**PIDs**	**Bytes**	**Descripción**	**Mínimo**	**Máximo**	**Unidades**	**Formula**
2	2	2	Freeze frame trouble code	-	-	-	BCD

Tabla 1.2 PIDs para interpretar datos del Freeze Frame del modo 02.

(César & López, 2014)

1.7.3.3 Modo 03: Códigos de diagnóstico almacenados

Para la detección de códigos de falla en los vehículos, lo fabricantes asignaron códigos denominados DTC, con la finalidad de guiar a los técnicos automotrices sobre el tipo de falla producida en el vehículo y su procedencia. La Sociedad Americana de Ingenieros, estableció la norma J2012 para estandarizar un formato para los códigos de diagnóstico, permitiendo que los scanners accedan a diversos sistemas de los vehículos sin considerar la marca de este. (Villamar Aguirre, 2008). El formato de los DTC establece valores alfanuméricos, consistiendo en un código con 5 dígitos, siendo el primer digito el lugar donde se produce la falla del sistema.

- P – POWERTRAIN (Tren motriz, proviene de la transmisión automática del motor).
- B – BODY (Cuerpo o carrocería del vehículo).

- U – NETWORK (Red, donde la falla tienen que ver con el sistema de trasmisión de los diferentes módulos).
- C – (Chasis, la falla proviene de diferentes elementos como: bolsas de aire, frenos entre otros).

El segundo dígito es un valor numérico encargado de indicar si el código es genérico o específico del fabricante. Siendo 0 para código genérico y 1 para códigos del propio fabricante del vehículo, el tercer digito de un código de falla OBDII es numérico y establece el subsistema donde se encuentra la falla. (Blasco, s/f).

1 – Detección de falla de un sensor en un subsistema que controla la relación de aire-combustible en el motor.

2 – Detección de falla de un subsistema de alimentación como: inyectores y bomba de combustible.

3 - Falla ubicada en el subsistema de encendido del vehículo.

4 – Falla con el subsistema de emisiones.

5 – Falla en marcha minina y velocidad.

6 – El módulo de control del motor (ECU) y salidas auxiliares, pueden presentar falla en la memoria del procesador o circuitos del procesamiento.

7 y 8 – Detección de falla en el sistema de trasmisión automática o sistema de control con tracción de 4 ruedas.

Mientras que los últimos dos dígitos indican la falla especifica detectada por el sistema OBDII.

1.7.3.4 Modo 04: Borrado de códigos de falla (DTC) y cuadro de almacenados

Este modo permite borrar todos lo DTCs almacenados y los cuadros de datos grabados de la Unidad de Control Electrónica del Motor (PCM), también se encarga de apagar el indicador de luz MIL cuando existen un fallo en sistema OBDII. (Mauricio & Velandia, 2014). El modo 04 a diferencia de los demás, debido a que su respuesta cuando no se puede borrar los datos congelados cambia totalmente. (Jorge Sánchez Carrizo, 2017).

1.7.3.5 Modo 05: Test de los sensores de oxígeno.

El modo 05 muestra los resultados de las pruebas ejecutadas a los sensores de oxígeno, con la finalidad de determinar el funcionamiento de estos y del convertidor catalítico. Además, la cantidad de sensores de oxígeno de penderá del modelo y la marca del vehículo según sea la necesidad de este. (César & López, 2014).

Los resultados de las pruebas realizadas se obtienen mediante el retorno de determinados voltajes, esto depende de la mezcla que puede existe entre aire y combustible. Mostrando valores como: el umbral del voltaje en el sensor de oxígeno de rico a pobre y viceversa, tiempo trascurrido durante el cambio de rico a pobre nuevamente viceversa, los voltajes máximos y mínimos del sensor y finalmente el tiempo de transición entre los diferentes sensores que tiene el sistema. (Jorge Sánchez Carrizo, 2017).

1.7.3.6 Modo 06: Resultados de las pruebas de monitoreo de control de otros traductores (monitoreo del sensor de oxígeno solo CAN).

En el modo 06 los resultados de las pruebas de monitoreo del control de sistemas específicos son datos codificados que se utilizan internamente por el PCM, con la finalidad de determinar el estado de la luz indicadora MIL, este modo no toma los valores en tiempo real. Sin embargo, en algunos vehículos de última generación incorporados con protocolo CAN puede variar en el contenido de este modo. (Dimaté Cáceres & González Castillo, 2013).

1.7.3.7 Modo 07: Mostrar códigos de diagnóstico pendientes, en el último o actual ciclo de manejo completado.

En la memoria de la ECU este modo permite la lectura de los DTCs que se han mostrado más de una vez. Los DTCs que se encuentran relacionados con el sistema de emisiones es reportado en este modo. (Dimaté Cáceres & González Castillo, 2013). En el caso de querer pasar al modo 3 de DTCs confirmandos, es necesario requerir hasta 3 ciclos de manejo continuo con la misma falla.

1.7.3.8 Modo 08: Control de funcionamiento del sistema a bordo.

Por medio del modo 08 se puede llevar a cabo prueba de actuadores, donde el técnico automotriz puede activar o desactivar actuadores como bombas de combustible, válvula de ralentí entre otros. Controlando el funcionamiento del sistema de diagnóstico interno del vehículo. (Mauricio & Velandia, 2014).

1.7.3.9 Modo 09: Información del vehículo.

El modo 09 se encarga de mostrar el número de identificación del vehículo (VIN), además de la identificación de la PCM y las calibraciones del software almacenados en el PCM. (Dimaté Cáceres & González Castillo, 2013).

1.7.4 Data Link Connector

El conector Data Link Connector (DLC) OBDII se encuentra estandarizado bajo la normativa J1962, siendo una herramienta de diagnóstico y los diferentes módulos del vehículo, además de acceso como al sistema de diagnóstico a bordo. (César & López, 2014). El conector tipo hembra fue usando posterior a 1996, se encuentra ubicado en la parte inferior del panel de instrumentos. Este cuenta con 16 pines y cada protocolo de comunicación es asignado a cada uno de los pines dependiendo de la marca del automóvil.

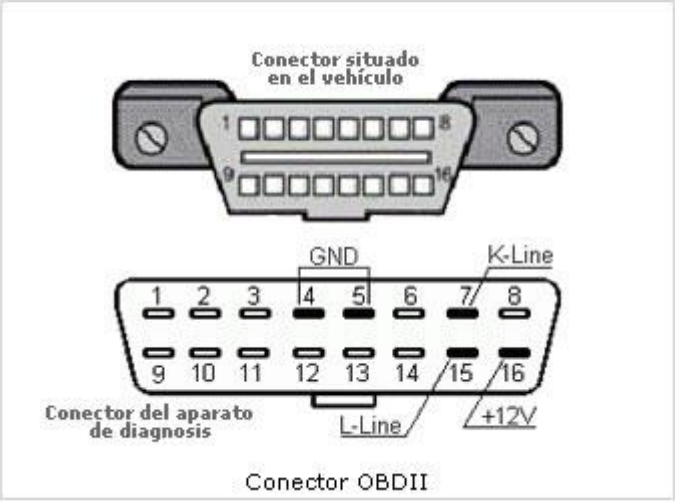


Figura 1.2 *Data Link Connector OBDII, tipo hembra*

(Automotriz, 2020)

El conector DLC – OBDII tipo macho es utilizado para herramientas de diagnóstico, sus pines son asignados tipo espejo al conector DLC tipo hembra.

1.8 REDES MULTIPLEXADAS Y PROTOCOLOS DE COMUNICACIÓN.

1.8.1 Conceptualización

Se conoce como redes multiplexadas a la interconexión que existe entre computadoras o módulos, con el fin de compartir información entre un módulo o una computadora, mediante la red multiplexada se obtiene varias ventajas como reducir el costo referente a espacio y peso de cableados, facilita la confiablidad y funcionalidad debido a la existencia de pocas conexiones plug-in, simplifica el ensamblaje del vehículo durante la producción y el uso múltiple de señales de los sensores. (Ruiz Campoverde, 2013). El uso de redes multiplexadas abre la oportunidad de usar un bus de datos, siendo un solo cable que porta la información de los módulos electrónicos.

1.8.2 Conceptos básicos de redes multiplexadas

- **Bit –** Se define como un digito en el sistema binario, el cual se representa por un periodo completo dentro de una secuencia de bits.
- **Byte** – Se denomina una unidad de información además es múltiplo del siendo su equivalencia igual a 8 bits.
- **Red –** Una red es un grupo de elementos donde se puede intercambiar de información a través de un medio de transportación.
- **Nodo** – Es un módulo electrónico el cual está conectado a una red, también conocidos como subscriptores y estaciones de red.
- **Bus de datos - Es** un canal de transición de información de la ECU, donde los datos son enviados por bytes al mismo tiempo.
- **Niveles de transmisión de datos** - Para la transmisión de datos existen dos niveles lógicos 0 y 1, donde el nivel recesivo está representado por el 1, mientras que el nivel dominante está representado por el 0.
- **Modos Sleep y Wake up -** Para reducir el consumo de energía es una red se implementan los nodos Sleep y Wake up, en el caso del nodo Sleep admite cesar la actividad cuando el nodo al ser afectado por alguna actividad o condiciones del sistema, permitiendo que el

controlador del bus se desconecta de la línea y en ese caso el nodo se convierte en Wake up. (Ruiz Campoverde, 2013).

1.8.3 Tipos de configuración de redes.

La configuración de una red es importante para realizar sistemas electrónicos modernos aplicados en el automóvil, el tipo de configuración dependerá directamente del fabricante ya sea el diseño de su electrónica y la ubicación de los diversos elementos instalados.

1.8.3.1 Configuración punto a punto

La configuración punto a punto es denominada como la más sencilla debido a que solo dispones de dos módulos; la comunicación entre la ECU y el scanner de diagnosticó automotriz, este tipo de red hace uso solo de dos pares de cables trenzados. (Carpio Guartambel, 2013).

Figura 1.3 Conexión de red punto a punto.

(Luna, 2016)

1.8.3.2 Configuración en estrella

La configuración de red tipo estrella es centralizada adquiriendo cierta ventaja a diferencias de las demás configuraciones, en el caso de existir una falla de conexión con un módulo, la configuración deja afuera solo a ese componente.

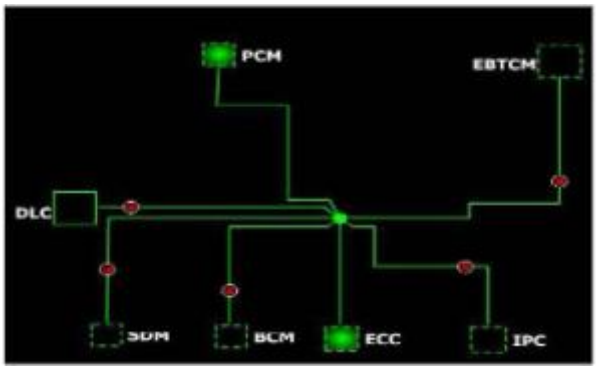


Figura 1.4 Conexión de red tipo estrella.

(Luna, 2016)

1.8.3.3 Configuración en anillo

Este tipo de configuración se considera como la más extensa debido a que se encuentran sumidos de 4 hasta 20 módulos, pero tiene como ventaje que la trasmisión de datos es bidireccional, en otras palabras, si un canal de comunicación se abre la información puede viajar por otra dirección y llegar hasta los módulos (Carpio Guartambel, 2013).

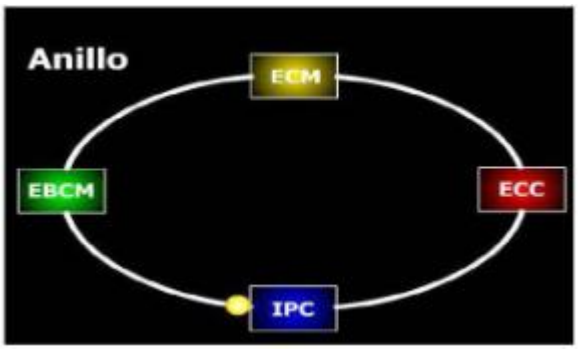


Figura 1.5 Configuración de red tipo anillo.

(Luna, 2016)

1.8.3.4 Configuración Lineal

A diferencia de los demás tipos de configuración de red, la configuración lineal tiene la característica especial de contener la mínima cantidad de cable para su estructura, permitiendo obtener una ruta sencilla de alambrado a lo largo del vehículo, además este tipo de configuración no requiere de un orden en la lectura de los datos, sin embargo, en el caso de la fractura de un cable la comunicación queda deshabilitada totalmente (Luna, 2016).

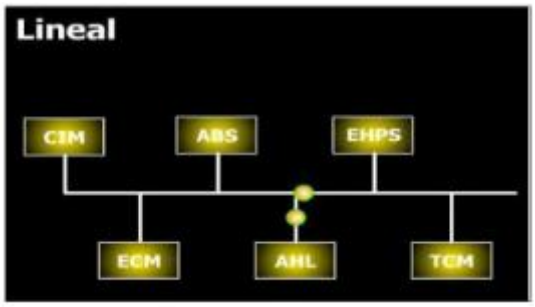


Figura 1.6 Configuración de red lineal.

(Luna, 2016)

1.8.3.5 Configuración Daisy Chain

La configuración Daysi Chain es aplicada al sector automotriz, debido a su mínima cantidad de nodos en su estructura permitiendo tener un alto grado de seguridad al contener dos canales de comunicación con los mismos datos de trasferencia.(Carpio Guartambel, 2013). La desventaja de esta configuración al igual que la lineal, en el momento que se desconecta un módulo queda deshabilitado todo el sistema.

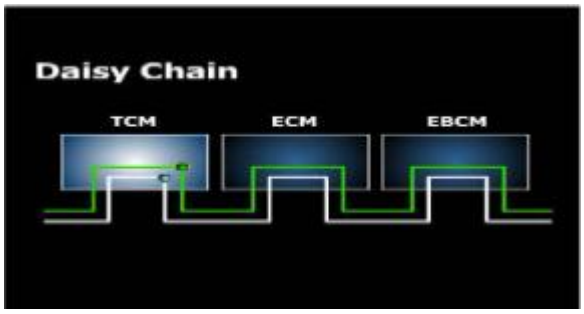


Figura 1.7 Configuración Daysi Chain

(Luna, 2016)

1.8.3.6 Configuración Maestro – Esclavo

La configuración de esta red se basa que la unidad de control debe estar comunicada con la red principal, con la finalidad de comandar bajo sus requerimientos a otros módulos. Se denomina a la unidad de control como maestro y a los módulos o ordenadores como esclavos.

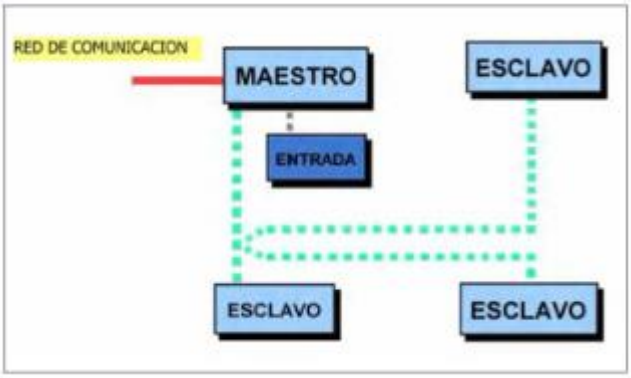


Figura 1.8 Configuración de red Maestro- Esclavo

(Luna, 2016)

1.8.3.7 Configuración Gateway

La configuración Gateway se denomina una compuerta de enlace, la cual realiza una conversión de un protocolo a otro, donde la aplicación se comunique por medio de la compuerta con la menor influencia.

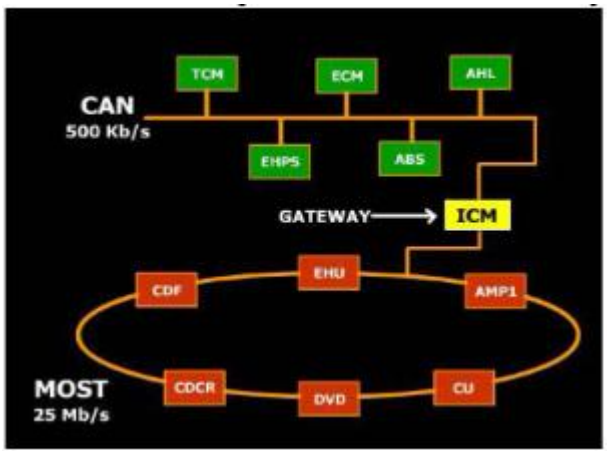


Figura 1.9 Configuración de red Gateway.

(Luna, 2016)

1.8.4 Protocolos de comunicación

Los protocolos de comunicación son un conjunto de normas establecidas, permitiendo una comunicación exitosa entre dos o más dispositivos con la finalidad de realizar un intercambio de información. En el caso de acceder a la ECU para diagnosticar el vehículo es necesario una herramienta de diagnóstico con el protocolo de comunicación que venga incorporado en

el automóvil. Los protocolos que actualmente se usan son ISO y SAE. (César & López, 2014).

1.8.4.1 ISO 9141-2

Este protocolo de comunicación es el más antiguo y fue estandarizado por ISO en 1989, se basa en una comunicación en serie asíncrona siendo el bit 1 con 12 voltios y el 0 con cero voltios. (César & López, 2014). Tiene un rango de tiempo de 96.15 segundos y una taza de bits estándar de 10.4 Kbps, este protocolo es usado generalmente por las marcas Chrysler y automóviles asiáticos y europeos. (Cantos Calderón et al., 2016). Los pines de conexión del DLC son 4,5,7,16 como se muestra en la figura.

Figura 1.10 Conector DLC ISO 9141-2

Figura 1.10 Conector DLC ISO 9141-2

(A.G.P.S, s/f)

1.8.4.2 ISO 14230-4

Se denomina un protocolo en serie asíncrona, donde tiene por lectura el mensaje de diagnóstico con velocidades de trasmisión de información de 1 200 a 10 400 baudios, generalmente este protocolo se utiliza la capa física del modelo OSI específicamente para redes computacionales, además de la capa de sesión en términos de iniciación y finalización de la comunicación. Donde el primer byte determina el modo de dirección ya sea físico o funcional, el segundo byte direcciona la recepción del mensaje y el tercer byte es la dirección

física siendo el remitente del mensaje de diagnóstico.(Cantos Calderón et al., 2016). Los pines de conexión del ISO 14230-4 es igual al ISO 9141-2, se muestra en la Figura 1.3.

1.8.4.3 SAE J1850

Este protocolo fue definido por SAE y es usado solo por General Motors siendo un estándar común en la industria automotriz, es un protocolo que maneja un bus de dos líneas y pulso de ancho variable (VPW), mientras que Ford Motors utiliza el protocolo del pulso de ancho modulado (PWM). Tiene una tasa de transferencia de 41.6 kb/s. (Esteban del Castillo Pérez, 2019).

En el caso del SAE J1850 PWM usa los pines de conexión 2, 4, 5, 10, 16 como se muestra en la figura.

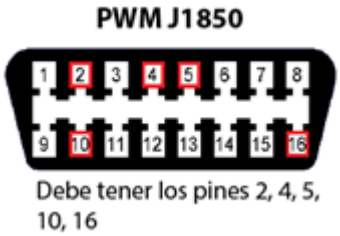


Figura 1.11 Conector DLC SAE J1850 (PWM)

(A.G.P.S, s/f)

Mientras que el protocolo SAE J1850 VPW usa los pines de conexión 2,4,5,10,16 como se muestra en la figura.

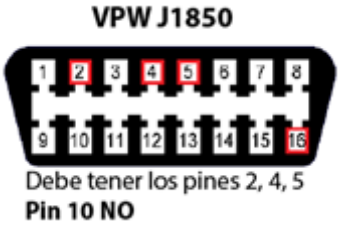


Figura 1.12 Conector DLC SAE J1850 (VPW)

(A.G.P.S, s/f)

1.8.4.4 CAN ISO 15765 – 4

Este es el protocolo más utilizado para vehículos de última generación, siendo su uso obligatorio para todos los automóviles que transitan en Estados Unidos a partir del año 2008. (Cantos Calderón et al., 2016). El protocolo CAN ISO 15765-4 fue desarrollado por la empresa Bosch para la industria automotriz y aeroespacial, generalmente es conocido como Controller Area Network (CAN), permitiendo la comunicación entre dos dispositivos sin la necesidad de usar un host. Tiene una tasa de transferencia de 1Mbit/s 0 1 000 000 baudios, sus pines de conexión DLC son: 6 y 14 como se muestra en la Figura 1.6. (César & López, 2014).

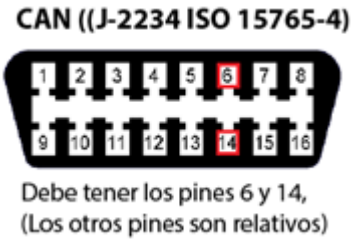


Figura 1.13 Conector DLC CAN ISO 15765-4

(A.G.P.S, s/f)

1.8.4.5 SAE J1939

Es un protocolo basado en CAN es generalmente utilizado en diferentes ares de la industria como: camiones pesados especialmente para motores diésel, vehículos de todo terreno, generación de energía y bombas industriales. Actualmente ante la creciente demanda del control de emisiones vehiculares los fabricantes de motores diésel optan por este protocolo. (César & López, 2014).

1.10 REDES MULTIPLEXADAS CAN BUS

Una red multiplexada CAN (Red de Área Controlada), es una interconexión mediante nodos entre dos o más módulos electrónicos a un BUS de datos, fue propuesta inicialmente para uso automotriz en los años 80´s por la compañía alemana Bosch. Establece estándares para una comunicación rápida, efectiva y en tiempo real entre módulos o unidades electrónicas del vehículo, simplificando considerablemente el uso de cables para interconectarlas

(Johansson et al., 2005). Es ampliamente utilizada en vehículos desde el año 2008, ya que, fue propuesta de manera no obligatoria a los distintos fabricantes automotrices como única red multiplexada de comunicación para vehículos (Instituto Mexicano del Transporte, s/f).

Ciertamente la red CAN BUS presenta varias ventajas como; reducción del cableado y conectores (gracias a su conexión punto a punto entre los dispositivos), facilidad de diagnóstico y no es susceptible a ruido o interferencias. En 1983 debido a su funcionalidad y utilidad la red CAN fue estandarizada como protocolo ISO 11898-1. Su amplio uso en la industria del transporte, provocó un incremento en la producción de microcontroladores con controladores CAN integrados, abaratando costos de adquisición y manufactura, haciendo cada vez más común su aplicación (Johansson et al., 2005).

1.10.1 Comunicación de datos a través de Red CAN

La Red CAN mantiene una estructura de comunicación ordenada entre módulos electrónicos conectados por nodos utilizando el BUS de datos, la cual permite una adecuada interoperabilidad entre los módulos electrónicos que conforman la red, manteniendo así un orden de emisión y recepción de datos (Microchip, 1999). El BUS de datos de la red transporta una gran cantidad de datos en forma de señales eléctricas de voltaje a manera de broadcast, los cuales los distintos módulos envían y reciben utilizando las conexiones por nodos. Estos datos o mensajes son denominados tramas las cuales son un conjunto ordenado de bits y se agrupan en campos dentro de un marco de datos para poder ser interpretados (Gallegos, 2015).

1.10.1.1 Algoritmo de detección CSMA/CA

La red CAN utiliza el algoritmo de detección CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance), donde todos los módulos que componen la red (a través de los nodos) monitorizan constantemente el BUS de datos antes de enviar un mensaje, con la finalidad de encontrar un periodo de inactividad en la red y evitar la colisión de datos (bits) (Microchip, 1999). La velocidad de comunicación de la red depende principalmente de la distancia entre nodos. Para distancias menores a 40 metros, se puede alcanzar velocidades de transmisión de hasta 1 Mbps (Johansson et al., 2005).

Si dos o más nodos de la red transmiten información al mismo tiempo, existe riesgo de colisión de datos. Estas colisiones se evitan utilizando métodos de arbitraje bit a bit no destructivos, manteniendo el mensaje intacto incluso si existieran colisiones. Estos arbitrajes toman lugar sin interrumpir o retrasar los mensajes de mayor prioridad enviados a través del BUS de datos (Johansson et al., 2005).

Existen algunas reglas que utiliza el algoritmo para mantener el mensaje de prioridad (arbitraje bit a bit). La primera regla define el mensaje como recesivo (1) o dominante (0), un bit de estado dominante siempre tendrá prioridad ante uno de estado recesivo, es decir los mensajes con bits dominantes se mantienen mientras que los mensajes con Bits recesivos se descartan. En la segunda regla un nodo transmisor monitorea el estado del BUS de datos para comprobar si el estado lógico que se desea enviar ya se encuentra en el BUS de datos (Microchip, 1999).

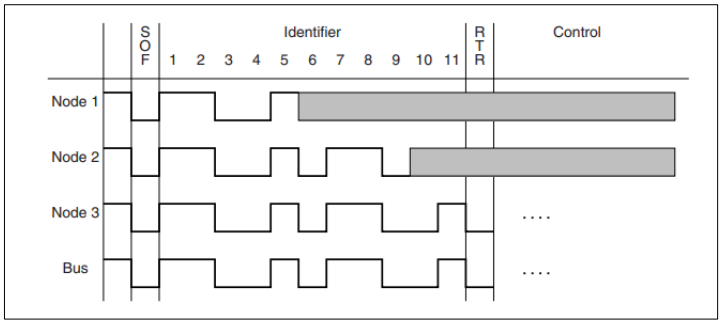


Figura 1.14 Ejemplo de arbitraje Bit a Bit entre tres nodos que transmiten simultáneamente.

(Johansson et al., 2005)

1.10.1.2 Comunicación basada en mensajes

El protocolo CAN es un protocolo basado en mensajes, sin embargo, los mensajes no se transmiten entre nodos, sino que están anexos a un marco de datos que se transmite a través del BUS de datos al cual se enlazan los nodos. Estos mensajes mantienen la prioridad que le otorga cada nodo y el contenido que estos transmiten. Cada nodo de la red recibe el mensaje del BUS de datos y decide si aceptarlo para procesarlo o descartarlo. Todos los nodos del sistema tienen la capacidad de solicitar información de otros nodos por medio del BUS de datos, esta característica se denomina RTR (Solicitud de Transmisión Remota) (Johansson et al., 2005).

1.10.1.3 Marco de datos de la Red CAN

El protocolo CAN utiliza cuatro tipos de campos dentro de su marco de datos, el cual inicia con un SOF (Estado de trama) y termina con una trama de reconocimiento ACK seguido de un EOF (Fin de trama). Estos campos proveen información necesaria dentro del marco de datos que utiliza la red (Microchip, 1999). En la figura a continuación se muestra un marco de datos completo.

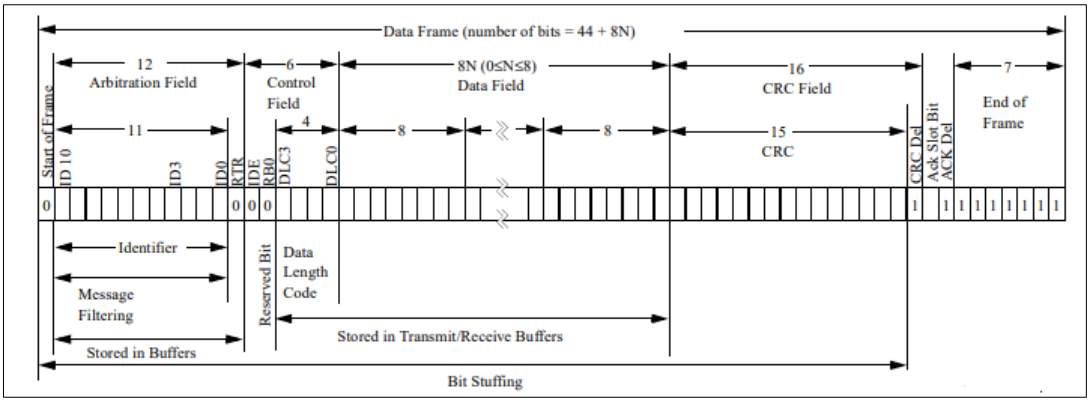


Figura 1.15 Marco de datos de una Red CAN

(Microchip, 1999)

Estado de trama SOF - Indica la prioridad de transmisión del mensaje

Campo de arbitraje - Este campo se utiliza para priorizar los mensajes que llegan al BUS de datos. El protocolo CAN define como 1 al Bit recesivo y 0 al Bit dominante. Tiene una longitud de 12 Bits en tramas estándar de los cuales 11 son identificadores de prioridad y el último es un Bit para RTR (Solicitud de Transmisión Remota) (Corrigan, 2002).

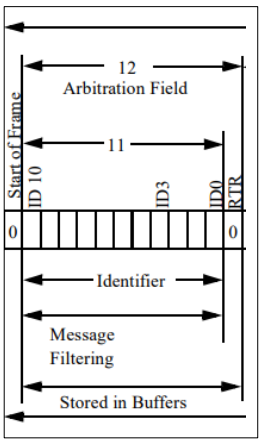


Figura 1.16 Campo de arbitraje estándar de 12 Bits de longitud, 11 Bits identificadores y un Bit RTR.

(Microchip, 1999)

En algunos casos la trama del campo de arbitraje puede estar compuesta de 32 Bits de los cuales, 29 son identificadores, 1 es un Bit SRR (Solicitud Remota de Sustitución) el cual siempre se transmite como recesivo, 1 es un Bit que define al campo como extendido (es decir que tiene una longitud de 29 Bits) y el último es un bit para RTR (Solicitud de Transmisión Remota) como en los casos de tramas estándar. Por regla general un campo de arbitraje de 12 Bits tiene mayor prioridad que uno de 32 Bits (Corrigan, 2002).

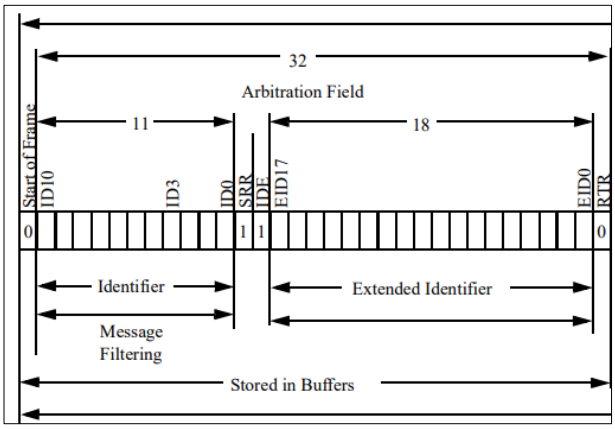


Figura 1.17 Campo de arbitraje estándar de 32 Bits de longitud, 11 Bits identificadores y un Bit RTR.

(Microchip, 1999)

Campo de control - Este campo está compuesto por 6 Bits, el Bit más significativo es el IDE Bit el cual indica que la trama está extendida y es dominante para tramas de datos estándar (Johansson et al., 2005). En tramas extendidas, este Bit es denominado RB1 y está reservado. El siguiente Bit se denomina RB0 y de igual manera está reservado. Los cuatro Bits menos significativos, muestran cuantos grupos de Bytes están incluidos en el mensaje; es decir, la longitud del Código de Datos (Microchip, 1999).

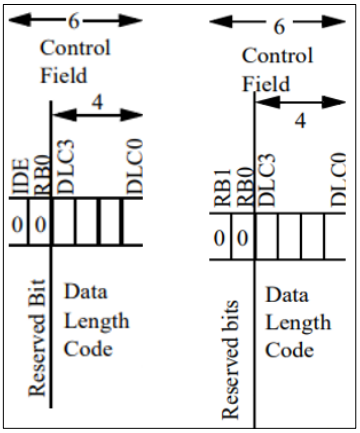


Figura 1.18 Campos de control para tramas de datos estándar (izquierda) y extendidas (derecha).

(Microchip, 1999)

Campo de datos - Está compuesto del número de grupos de Bytes indicados en el campo de control (Microchip, 1999).

Campo CRC - Es un campo indispensable en el rendimiento de la red, se denomina campo de redundancia cíclica y consta de 15 Bits y un Bit delimitador (Microchip, 1999). Este campo contiene la suma de comprobación de Bits o números de Bits transmitidos en el Campo de datos (Corrigan, 2002). Es usado por nodos receptores para comprobar errores de transmisión de datos, gracias a este campo la probabilidad de tener errores de transmisión de datos es muy baja (Johansson et al., 2005).

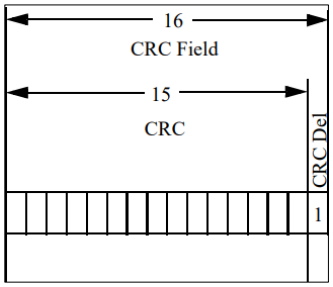


Figura 1.19 Campos CRC

(Microchip, 1999)

Trama de reconocimiento ACK - Esta trama consta de dos Bits, uno denominado ACK slot Bit el cual es utilizado para indicar si el mensaje fue recibido correctamente, aquí cada nodo que ha recibido el mensaje de manera correcta añade un Bit dominante en este espacio

reemplazando al Bit recesivo. El otro es un bit delimitador que indica que el ACK finaliza (Corrigan, 2002).

Fin de trama - Está compuesto de 7 Bits recesivos e indica que la trama finalizó. (Johansson et al., 2005)

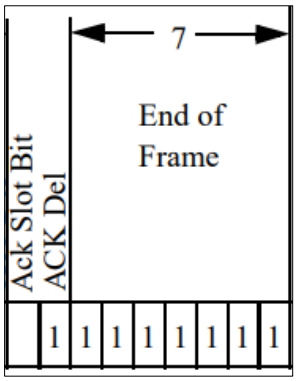


Figura 1.20 Trama de reconocimiento y fin de trama.

(Microchip, 1999)

1.10.1.4 Manejo de errores en la Red CAN

Existen 5 formas diferentes de detectar errores de transmisión de datos en el protocolo CAN como: el monitoreo de Bits, el relleno de Bits o Bits Stuffing, el Bit de reconocimiento de tramas y la comprobación de campos CRC (Johansson et al., 2005).

- **Monitoreo de Bits -** El monitoreo de Bits es un suceso donde cada transmisor monitorea el nivel del BUS de datos, si no coincide el nivel con la señal que se transmite el transmisor señala un error. Es importante mencionar que la monitorización de Bits no ocurre en el campo de arbitraje) (Johansson et al., 2005).

- **Bit Sttufing -** Cuando un nodo transmite 5 Bits idénticos envía un Bit opuesto a esos Bits idénticos, a este proceso se le denomina Bit Stuffing (relleno de Bits) y también es una forma de detectar errores en la transmisión de los datos. Este Bit opuesto es descartado por el nodo receptor de la transmisión. El Bit Stuffing tiene lugar únicamente desde el inicio del marco de datos, hasta el delimitador del campo CRC (Johansson et al., 2005).

- **Chequeo de tramas -** Es un proceso que verifica que los Bits de la trama tengan los valores que se supone que deben tener, por ejemplo, los siete Bits del final de la trama deben ser recesivos, si uno es dominante se marcará un error (Corrigan, 2002).
- **Bit de reconocimiento ACK -** Tiene lugar en el campo de reconocimiento ACK.
- **Comprobación CRC -** Tiene lugar en el campo CRC.

1.10.2 Estructura física de una Red CAN

1.10.2.1 Bus de datos

Consta de un par de cables trenzados denominados CAN High y CAN Low respectivamente, los cuales transportan señales eléctricas (datos) de cada nodo de la Red, estos datos son transmitidos por diferencia de tensión entre ambos cables. Si uno de los dos cables falla, el otro hace diferencia con masa y continúa operando la Red (Instituto Mexicano del Transporte, s/f, p.42).

En una línea CAN High la velocidad de transmisión de datos de 0,25 milisegundos. Cada Unidad Electrónica de Control del vehículo transmite datos a distinta velocidad, por ejemplo, el módulo ABS transmite cada siete milisegundos, el módulo de la gestión electrónica del MCI cada 10. El CAN High tiene un valor de tensión de entre 2,5 y 3,5 V, mientras que el CAN Low opera alrededor de 1,5 y 2,5 V (Gallegos, 2015).

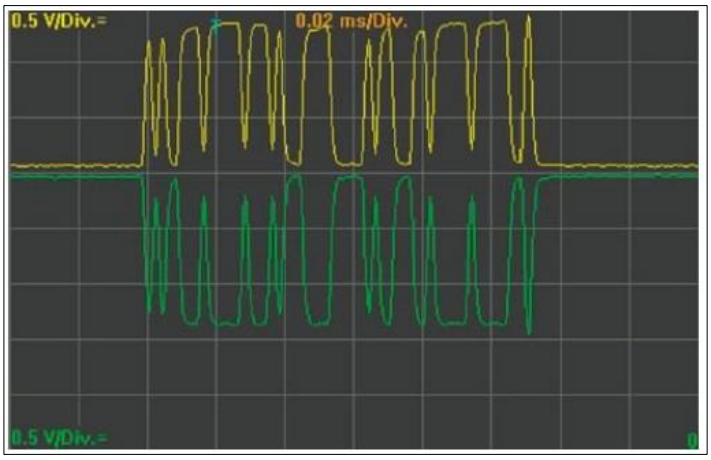


Figura 1.21 Oscilograma de las señales del BUS de datos, se puede observar la señal CAN High en amarillo y la señal CAN Low en verde.

(Gallegos, 2015)

Cuando la señal CAN High y CAN Low se separan existe un bit dominante (0), cuando las señales se juntan existe un Bit Recesivo 1, generalmente las señales CAN High y CAN Low son simétricamente opuestas en el oscilograma.

1.10.2.2 Resistencias de terminación

Son un par de resistencias que se conectan al extremo del BUS de datos (una en cada extremo) tienen un valor aproximado de 120 ohm cada una y su finalidad es evitar el reflejo de las señales (Gallegos, 2015).

1.10.2.3 Unidad Electrónica de Control

Cada vehículo tiene un número diferente de Unidades Electrónicas de Control, cuentan con uno o más microprocesadores o microcontroladores conectados a un nodo que a su vez se conecta al BUS de datos. Este dispositivo es capaz de procesar la información de los diferentes sensores y actuadores que conforman el sistema vehicular al que pertenecen (Instituto Mexicano del Transporte, s/f, p.9).

1.10.2.4 Nodo CAN

Un nodo de Red CAN tiene principalmente 3 componentes:

- **Host Processor-** Es un controlador que se encarga de comprender los mensajes que llegan al nodo y elige los mensajes que se requiere transmitir desde el microprocesador de la unidad electrónica del control del vehículo. (Pacheco, 2011).
- **CAN Controller -** Es un controlador encargado de almacenar los datos de la trama que requiere transmitir el Host Processor. Además, puede enviar los bits de la trama uno por uno (Pacheco, 2011).
- **Transceptor -** Es anexo al CAN controller y se encarga de ajustar los niveles lógicos entre el CAN controller y el BUS de datos, es decir transforma las señales digitales en señales eléctricas (Pacheco, 2011). Utilizan un voltaje de alimentación de 5V DC, sin embargo, existen transceptores que utilizan 3.3V únicamente. Estos dispositivos son capaces de soportar corto circuitos accidentales, picos de voltaje, estática y ruido eléctrico (Corrigan, 2002).

1.10.3 Tipos de redes multiplexadas CAN

Las redes multiplexadas CAN en un vehículo se clasifican de acuerdo a la velocidad en que transmiten los datos la cual puede ir desde 10Kbps a 1Mbps, la señal que generan (eléctrica, luminosa o inalámbrica) y el protocolo que emplean (Gallegos, 2015).

1.10.3.1 Protocolos de comunicación basados en CAN

- **CAN bus -** Tiene una buena velocidad de transmisión de datos y se utiliza en sistemas que requieren transmitir información con muchos datos de prioridad y a una buena velocidad.
- **Lin bus -** Es una extensión de CAN, se utiliza para sistemas que no requieren emitir mucha información y su velocidad es inferior, por ejemplo: vidrios eléctricos, climatizador, techo corredizo. Utiliza un solo cable, opera a una velocidad aproximada de 20 Kbps y es unidireccional. EL voltaje de trabajo es de 11 y 12 voltios. Mantiene jerarquía de conexión con unidades electrónicas maestras y esclavas a diferencia de la red CAN bus (Gallegos, 2015).
- **Most bus -** Similar a CAN, pero transmite los datos por fibra óptica, por tanto, es mucho más rápida.
- **Van bus -** Es similar a CAN, pero con prestaciones limitadas.
- **Flex Ray -** Presenta una mejora en velocidad de transmisión de datos, aparece en vehículos recientes.

1.10.4 Comunicación entre protocolos CAN

En un vehículo, usualmente se pueden encontrar dos o más protocolos basados en CAN, por ejemplo: CAN bus y Lin bus. En este caso el CAN bus se utilizaría para la comunicación entre el módulo del motor y el módulo de la caja de cambios y el Lin bus se utilizaría para los sistemas de confort del vehículo. Esto se hace con la finalidad de no saturar de nodos la red CAN bus. La información entre protocolos CAN puede ser compartida a través de un módulo Gateway. Los Gateway son puertos de enlace de la red del vehículo y tienen la

finalidad de interconectar diferentes tipos de protocolos CAN con distintas velocidades de transmisión de datos (Ingeniería y mecánica automotriz, s/f).

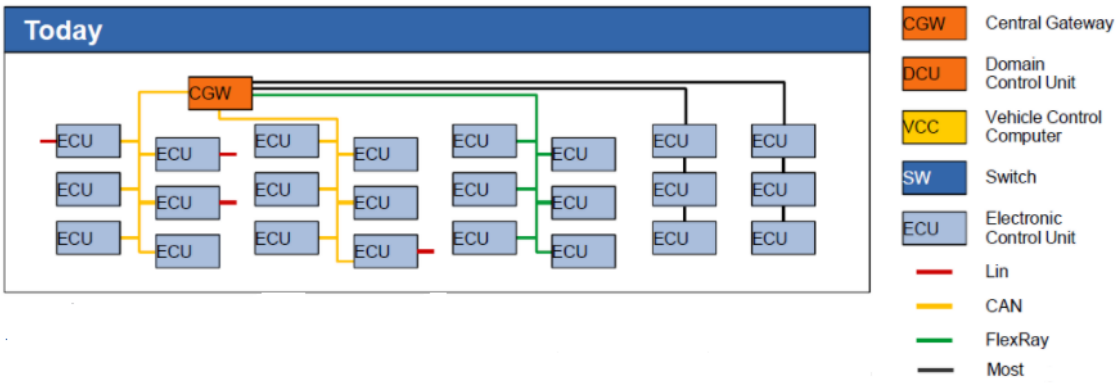


Figura 1.22 Ejemplo de Red CAN con distintos protocolos CAN interconectados por un módulo Gateway.

(Ingeniería y mecánica automotriz, s/f)

1.11 UNIDADES ELECTRÓNICAS DE CONTROL (ECU)

La Unidad Electrónica de Control (ECU) es una computadora que consta de un microcontrolador/microprocesador con un software programado que se encarga de mantener el funcionamiento autónomo de un sistema mecatrónico evaluando señales del exterior por medio de sensores y comandando respuestas usando actuadores. Son ampliamente utilizadas en la industria automotriz desde el año 1980 debido a la introducción de la inyección electrónica del combustible con la finalidad de reducir las emisiones de gases contaminantes de los vehículos. Actualmente las Unidades Electrónicas de Control cumplen algunos propósitos en el vehículo y no solo el de reducir las emisiones. Cada computadora, se destina a un determinado sistema automotriz, por ejemplo: ABS, motor, transmisión, dirección, climatización, confort, entretenimiento, entre otros. Esto con el fin de mantener el correcto funcionamiento y mejorar la eficiencia de los sistemas a los cuales son anexas y reciben su nombre de acuerdo al sistema al que pertenecen (Spiegel, 2015).

1.11.1 PCM

La computadora principal en la mayoría de vehículos se denomina PCM (Módulo de Control del Tren de Potencia), este módulo es una combinación entre una Módulo de Control del Motor (ECM) y Módulo de Control de la Transmisión (TCM) en el caso de transmisiones automáticas o una interconexión de ambos (Spiegel, 2015).

1.11.2 ECM

El Módulo de Control del Motor ECM, es una ECU utilizada para mantener el funcionamiento de los motores de combustión interna, se encarga de la relación aire-combustible, ralentí, avance y tiempo de encendido de las bobinas, apertura y cierre de válvulas de admisión variables, entre otros. Estas funciones son controladas por medio de datos de sensores de posición, temperatura, flujo de aire o presión de vacío. Este módulo tiene comunicación directa con el Módulo de Control de la Transmisión (TCM), el Body Control Module (BCM), el sistema de control de estabilidad (ESP), el sistema anti robo, entre otros (Ilakkiya B & Vanitha V, 2016).

1.11.3 TCM

El Módulo Electrónico de Control de la Transmisión, se encarga de comandar los cambios en los vehículos de transmisión automática, procesando de manera precisa la información que recibe de la ECM y de los diferentes sensores que se encuentran instalados o forman parte de la transmisión automática (Spiegel, 2015).

1.11.4 EBCM

El Módulo Electrónico de Control del Frenado, se encuentra equipado en los vehículos que cuentan con sistema antibloqueo de frenos (ABS), su función es evitar el bloqueo de frenos regulando y distribuyendo adecuadamente la fuerza del frenado (Ilakkiya B & Vanitha V, 2016).

Evalúa 5 señales para la toma de decisiones:

- Si el vehículo está encendido

- Si el pedal del freno está presionado
- La velocidad de giro de cada rueda
- La velocidad proporcionada por el VSS (Vehicle Speed Sensor)
- Si la doble tracción está activa

1.12 MICROCONTROLADORES PROGRAMABLES

Los microcontroladores programables son dispositivos electrónicos con circuitos integrados contenidos en encapsulados y se destinan a gobernar una tarea o instrucción específica previamente programada la cual es grabada en su memoria. Para desarrollar proyectos con microcontroladores programables, se requiere de herramientas específicas de software y hardware. Los microcontroladores disponen de puertos o pines que pueden ser programados como entradas o salidas. Generalmente, el fabricante del microcontrolador proporciona el software IDE para la programación (David, 2010).

1.12.1 Entorno Integrado de Desarrollo IDE

Es un software utilizado para editar, compilar y cargar el código de programación o instrucciones a la memoria del microcontrolador (Fezari & Ali, 2018).

1.12.2 Arduino

Arduino es una tarjeta electrónica con microcontrolador incorporado de software y hardware libre. El microcontrolador se programa usando el lenguaje de programación de Arduino el cual es basado en C y un entorno de desarrollo IDE. Consta de múltiples pines de entrada o salida que sirven para conectar sensores y controlar dispositivos externos en base a instrucciones programadas, algunos de estos pines tienen funciones especiales (Enríquez Herrador, 2009, p. 13).

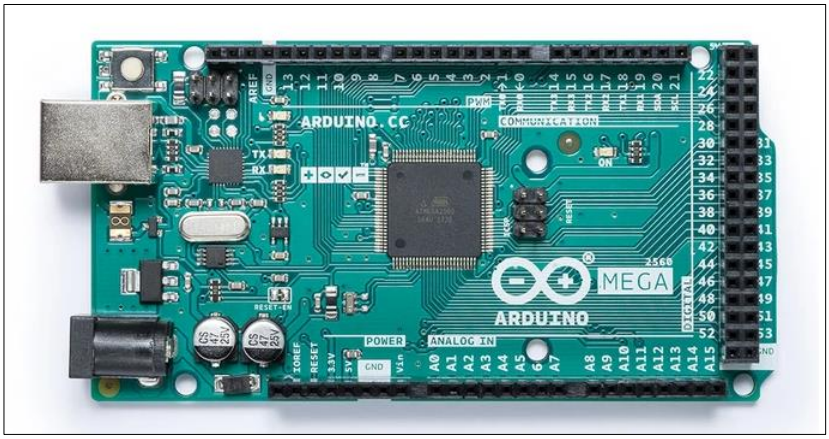


Figura 1.23 Placa de Arduino mega 2560 Rev3.

(store-usa.arduino.cc, s/f)

- **Comunicación serial: (Rx) y (Tx)** - Se utilizan para mantener una comunicación serial entre microcontroladores, siendo receptor el pin (Rx) y transmisor el pin (Tx).
- **PWM** - Proporcionan salidas PWM de 8 bits (1 byte).
- **Comunicación SPI: SCLK, MOSI, MISO, CS** - Según el sitio web (proyectoarduino.com, s/f), La comunicación SPI es una interfaz de comunicación que permite una conexión tipo Bus serial, se utiliza para mantener una interconexión entre microcontroladores a distancias cortas que requieran una gran velocidad de transmisión de datos. Este tipo de interfaz mantiene una estructura maestro esclavo, aquí un microcontrolador principal es el maestro y el resto esclavos. Esta interfaz, requiere de 4 líneas de comunicación: SLCK (Pulsos de reloj del microcontrolador maestro), MOSI (Datos del maestro al esclavo), MISO (Datos del esclavo al maestro), SS (indicador de activación del esclavo).
- **Interruptores externos** - Son pines que pueden ser configurados para disparar una interrupción, con un margen creciente o decreciente y generar un cambio de valor (Enríquez Herrador, 2009, p. 13).
- **Comunicación I2C: VCC, GND, SDA Y SCL** - Este tipo de interfaz, también mantiene una estructura maestro esclavo. Aquí los datos enviados por los maestros son de una dirección, es decir, no se espera respuesta de los esclavos. Para este tipo de interfaz se

requieren 4 líneas: VCC (para alimentación), GND (para masa), SDA (envío de datos) y SCL (sincronización del reloj).

- **AREF** - Es un voltaje de referencia que utilizan las entradas analógicas (Enríquez Herrador, 2009, p. 13).
- **Reset** - Reinicia el microcontrolador.

1.12.2.1 Tipos de memorias empleadas en Arduino

Arduino utiliza tres tipos de memorias y la capacidad de cada una dependerá del tipo de microcontrolador que tenga la placa (Enríquez Herrador, 2009).

- **EEPOROM** - Es una memoria no volátil donde se puede grabar datos desde el IDE, borrarlos y volverlos a grabar, es una memoria de solo lectura y tiene la capacidad de mantener los datos incluso después de ser reiniciado.
- **FLASH** - Es una memoria de almacenamiento amplio, van desde 1Kb a 4Mb, aquí se guarda el código de programación ya compilado. Estas memorias tienen una vida útil de 100.000 ciclos de escritura, al igual que la memoria EEPROM no es volátil.
- **SRAM -** Es una memoria volátil, el compilador crea y escribe variables aquí cuando se ejecuta.

1.12.2.2 Programación

Cualquier placa de Arduino puede ser programada utilizando el IDE de Arduino, simplemente se debe seleccionar el modelo de placa Arduino y el puerto USB al que está conectado la placa para establecer una comunicación con la computadora (Enríquez Herrador, 2009).

1.12.2.3 Protección de sobrecarga

Todas las placas de Arduino cuentan con fusibles reseteables para protección de sobrecargas por cortocircuitos de aproximadamente 1 A, lo cual hace completamente seguro trabajar con placas Arduino en sistemas embebidos.

1.13 SENSORES QUE INTERVIENEN EN EL FUNCIONAMIENTO DE SISTEMAS DE SEGURIDAD DEL VEHICULO

1.13.1 Velocidad

El sensor de velocidad tiene la finalidad de realizar mediciones de movilidad o por relación de respuesta a las frecuencias denominadas como aceleración, generando señales senoidales, mediante la amplificación de potencias y excitando vibraciones por medio del sensor de velocidad (Rueda et al., 2000).

1.13.2 Magnitud de viraje

Este sensor se encuentra situado en el centro de gravedad del vehículo, verificando los pares de giro de un cuerpo, con la finalidad de medir los pares sufridos en el eje vertical del vehículo. Este sensor está conformado por un cilindro hueco con 8 elementos piezoeléctricos, donde la mitad de ellos producen una oscilación resonante al cilindro, mientras que los otros comprueban las variaciones producidas en el sistema (Zapatería, 2007).

1.13.4 Goniométrico de dirección

Es un sensor que se encuentra alojado en la columna de la dirección y tiene la finalidad de suministrar la señal correspondiente al giro del volante, comprobando valores de giro hasta ±720° siendo 4 vueltas por volante. El funcionamiento de este sensor se basa por el paso de luz a través de una serie de ventanas (Zapatería, 2007).

1.13.5 Tiempo de frenado

El sistema de frenado se denomina un sistema crítico de un vehículo, teniendo el objetivo de reducir la velocidad o detener el vehículo. En situaciones ideales, el conductor tendrá el tiempo necesario para anticipar la necesidad de reducir la velocidad antes de producirse una colisión tras el resultado de varias pruebas el tiempo de reacción medido en maniobras de frenado presentan valores de 0.75 a 1.5 (Farfán et al., s/f).

1.13.6 Distancia de frenado

En el caso de una emergencia la distancia de frenado, generalmente se encuentra afectada por el tiempo de reacción del conductor, usualmente es las reconstrucciones de los accidentes se considera el tiempo de reacción desde la percepción del peligro hasta que todos los frenos se encuentren bloqueados (Farfán et al., s/f). Además, la desaceleración de un vehículo es la tasa de decrecimiento de su velocidad respecto al tiempo, donde la velocidad disminuirá a medida que el tiempo transcurra.

1.13.8 Presión de frenado

La presión de frenado es controlada por un sensor que se encarga de suministrar la señal de presión, presente en el circuito de frenos de la Unidad de Control Electrónica, cuando el sensor piezoeléctrico es sometido a la presión del líquido de frenos, la tensión producida será proporcional a la presión generada (Zapatería, 2007). Tomando esta señal producida como el valor de frenado en cada momento.

1.13.9 Aceleración transversal

Controlado por un sensor de aceleración transversal se encuentra ubicado en el centro de gravedad del vehículo, con la finalidad de detectar la presencia de fuerzas laterales y la intensidad de estas. Siendo estas fuerzas las que empujan al vehículo fuera de su trayectoria, estas variaciones son medidas por un sensor piezoeléctrico que trasmiten señales a la Unidad de Control Electrónico, interpretando la dimensión de viraje (Zapatería, 2007).

1.13.10 Velocidad de las ruedas

El sensor de velocidad de las ruedas se encuentra ubicada en el buje de las ruedas, con la finalidad de determinar la velocidad de giro de esta. Existen dos tipos de sensores de velocidad pasivo que no necesita alimentación para su funcionamiento, mientras que el activo necesita alimentación para funciona (Gonzales, 2019)**.**

1.13.11 Torque del volante

Este tipo de sensor se encuentra instalado en el conjunto del volante de dirección, donde posee varias pistas en forma de disco, las cuales giran y miden la resistencia del torque y posteriormente procesada por el módulo de dirección para activar el nivel de asistencia al servomotor (Martinez, 2018).

1.13. SISTEMAS DEL AUTOMÓVIL QUE INTERVIENEN EN LA SEGURIDAD DINÁMICA DEL VEHÍCULO

1.13.1 Sistema de control de frenado

Este sistema se encarga de regular la presión del frenado en cada rueda de manera individual para evitar el bloqueo de las mismas, conservando la adherencia sobre la superficie, previniendo derrapes descontrolados (Rajamani, 2012).

Este sistema está compuesto por:

- Unidad de Control Electrónico

Se encarga de operar los algoritmos matemáticos programados utilizando los valores provenientes de los sensores de velocidad de las ruedas, presión y activación de freno para comandar el sistema de frenos y de ser necesario reducir la potencia del motor comunicándose con la ECM del vehículo.

- Válvulas de control de frenado

Se encargan de repartir y distribuir la fuerza de frenado.

- Sensores de velocidad de las ruedas

Miden la velocidad de giro de cada rueda en rpm, rad/s, m/s o km/h.

1.13.2 Sistema de control de estabilidad

El control electrónico de estabilidad (ESC) también conocido como ESP (Programa electrónico de estabilidad) o VDC (Control dinámico del vehículo) de acuerdo al fabricante, tiene como misión evitar un giro descontrolado o derrape, el cual pueda provocar la pérdida de control del vehículo o volcamiento en trayectorias donde el conductor proporcione un movimiento brusco del volante (Rajamani, 2012).

Es necesario que el vehículo disponga del sistema ABS (Anti-lock Braking System) y Sistema EPS (Dirección Controlada Electrónicamente) debido a que la ECU utilizará sensores de estos sistemas para comandar el correcto funcionamiento de los actuadores que intervendrán en el control de giro para mantener una trayectoria deseada por el conductor.

Este sistema está compuesto por:

- Unidad de Control Electrónico

Se encarga de operar los algoritmos matemáticos programados utilizando los valores provenientes de los sensores de velocidad, guiñada, torque y posición del volante, aceleración transversal, para comandar los actuadores e intervenir en la estabilidad del vehículo.

- Sensor de ángulo y torque de dirección o volante

Mide el torque en N/m y posición del volante en grados proporcionados por el conductor para determinar el ángulo de la dirección del vehículo permitiéndole a la ECU inferir la trayectoria deseada.

- Válvulas de control de frenado

Se encargan de regular la presión de frenado en cada rueda.

- Sensor de ángulo de giro (Yaw sensor) y aceleración transversal del vehículo

Proporcionan información sobre los ejes de desplazamiento del vehículo (giro en grados y aceleración en el eje transversal en m/s^2 o gravedades).

- Sensor de presión de frenado

Mide la presión aplicada en Bares o PSI al sistema de frenos.

- Sensores de velocidad de las ruedas

1.13.3 Sistema de control de tracción

El sistema de control de tracción (TCS) o también llamado ASR (Sistema de Regulación Antideslizamiento) tiene como función evitar que las ruedas pierdan adherencia (patinen) durante la fase de aceleración o cuando la misma es baja, permitiendo al vehículo mantener

la adherencia en la superficie controlando la velocidad de giro de las ruedas mediante los frenos y la potencia del motor/tansmisión en todo momento (*Control Electrónico de Frenado - ABS, ESP y Control de Tracción*, s/f).

1.14 SISTEMAS DEL AUTOMOVIL QUE INTERVIENEN EN LA SEGURIDAD DE LOS OCUPANTES DEL VEHICULO

1.14.1 Sistemas de retención suplementarios

Son sistemas que se encargan de la protección de los ocupantes en caso de accidentes como: vuelco, choques: frontales, laterales y posteriores, entre otros. Permitiendo la disminución de lesiones o muertes en el caso de accidentabilidad. Elementos que interactúan con el objetivo de salvaguardar la vida del ocupante dentro del habitáculo del vehículo se denominan sistemas de retención suplementarios o también conocidos como elementos de seguridad pasiva, siendo estos el cinturón de seguridad y las bolsas de aire (Airbag). (Gámez Santana & Sanabra Loewe, 2005).

1.14.1.1 Pretensores y cinturones de seguridad

Siendo el sistema de retención suplementario más importante y antiguo, el cinturón de seguridad es un arnés que mantiene al conductor en su sitio en el caso de existir una brusca desaceleración provocada por un choque. Los primeros cinturones de seguridad eran de un solo tramo que se encargaban solo de asegurar la cadera, posteriormente se desarrolló el segundo tramo que aseguraba el tronco superior del ocupante, dando lugar a un cinturón de seguridad de tres puntos utilizado actualmente (Moriche Guerrero, 2008). El inconveniente de este elemento de seguridad pasiva; en el caso de accidentabilidad el cinturón de seguridad provocaba al ocupante una separación de las vértebras ocasionándole una parálisis.

El cinturón de seguridad funciona a base de pretensores mecánicos y pirotécnicos, además de un carrete inercial, con el paso del tiempo estos elementos son mejorados permitiendo otorgar seguridad al ocupante sin ocasionarle posibles lesiones.

1.14.1.2 Bolsas aire (AIRBAG)

En el caso de una colisión es un sistema destinado a impedir lesiones en la cabeza y el tórax, convirtiendo en un complemento para el cinturón de seguridad permitiendo una disminución de daños en el cuerpo del ocupante hasta un 30% siendo un sistema de retención suplementario (Moriche Guerrero, 2008). El sistema de airbag se basa en unas bolsas de aire ubicadas en el volante y en el salpicadero, actualmente los vehículos vienen incorporados con airbags laterales que se encuentran en la parte superior de las lunas o elevavidrios. Según la disponibilidad del vehículo también se incluyen airbags de rodillas. Al igual que el cinturón de seguridad, estas bolsas de aire hacen uso de pretensores pirotécnicos, en el caso de un accidente estos se activan en milésimas de segundo y por medio de la ECU se facilita su diagnóstico en caso de avería.

1.14.2 Sistemas ADAS

El sistema Avanzado de Asistencia a la Conducción se encarga de mejorar la seguridad automovilística minimizando riesgos de sufrir un accidente vial o colisionar con otros automóviles. Incluyendo tres tecnologías que permiten facilitar la actividad de la conducción (García Alvarez, s/f)

1.14.2.1 Sistemas de detección de peatones

Este sistema permite la detección de peatones mediante sistemas de visión por computador, siendo uno de los componentes tecnológicos que más se han desarrollado en los últimos años con el progreso de la robótica móvil aplicada a la industria automotriz destinada a la seguridad vehicular (Galarza Bravo et al., 2018). Su funcionamiento se basa en la detección de peatones mediante un radar integrado en la rejilla del vehículo y una cámara situada en el espejo retrovisor. Permitiendo un análisis de datos de los dos elementos a través de la Unidad de Control Electrónica (Cisneros Oscar, 2010).

1.14.2.2 Control de crucero adaptativo

El control de crucero adaptivo (ACC) es una versión avanzada del control de crucero estándar (CC), con la función de encargarse en mantener la velocidad prefijada del conductor, asegurándose que el vehículo mantenga una distancia segura del vehículo

delantero. Esto se lleva a cabo mediante cámaras y láseres y un radar, además, el ACC permite adaptarse a las diferentes condiciones del tráfico, evitando que el conductor vuelva a reactivar el sistema cada vez que se vea obligado a frenar (Costas et al., s/f).

1.14.2.3 Control del cambio de carril

Es un sistema que ayuda a controlar la trayectoria del vehículo evitando que este se salga del carril cuando se encuentra en circulación. Hace uso de una cámara que se encuentra delante del retrovisor para detectar de manera continua las líneas que delimitan los carriles, enviando estos datos a Unidad de Control Electrónica, para emitir una alerta al conductor por medio de señales luminosas o acústicas.

CAPÍTULO II

2 MATERIALES Y MÉTODOS

2.1 METODOLOGÍA DE LA INVESTIGACIÓN

En el presente capítulo se explica la metodología de la investigación empleada para el desarrollo del dispositivo propuesto, con la finalidad de cumplir los objetivos previamente establecidos. Por medio de una investigación documental, se recaba información considerada necesaria e importante que aporte conocimiento específico sobre temas relacionados a este proyecto. Utilizando un diagrama de flujo propuesto, se establecieron procedimientos secuenciales y resultados que se esperan obtener en base al método experimental. Una vez terminado cada proceso, se realiza una explicación en base a la interpretación de resultados usando el método analítico-explicativo. Aquí se consideran variables necesarias que tendrán o no que ser cuantificadas, por eso se considera hacer uso del método cuantitativo para cumplir con este lineamiento.

2.1.1 Enfoque investigativo

El propósito de este proyecto es desarrollar un dispositivo basado en microcontrolador Arduino para la adquisición de datos a través de la red CAN para conocer las condiciones que suscitan accidentes vehiculares, para lo cual, es necesario indagar el funcionamiento de las redes multiplexadas CAN, el protocolo OBDII y considerar variables de interés que se puede obtener de las Unidades Electrónicas de Control del vehículo para extraer valores y almacenarlos en una situación necesaria o que el dispositivo propuesto lo considere, utilizando códigos PID enviados a través de una intercomunicación entre la red CAN del vehículo y el microcontrolador programado (dispositivo propuesto).

2.1.2. Tipo de investigación

2.1.2.1 Método documental

Este proyecto empleó un método documental para recopilar toda la información teórica necesaria para el desarrollo del dispositivo propuesto, utilizando varias fuentes de información de estudio previos o relacionados. Por medio de uso de, libros, revisiones técnicas, tesis, artículos científicos, internet entre otros.

2.1.2.2 Método cuantitativo

Se utiliza el método cuantitativo, donde las variables consideradas de las Unidades Electrónicas de Control del vehículo son cambiantes y al momento del estudio los valores que arrojan deben ser medidos, comparados y almacenados, es decir, se debe considerar una relación causa y efecto con cada variable. A partir de los valores numéricos obtenidos dichas variables, se puede emitir conclusiones del estado y comportamiento del vehículo durante el accidente o hacer un estudio posterior a la toma de datos.

2.1.2.3 Método analítico

Se considera el método analítico, debido a que los resultados obtenidos por el microcontrolador programado y la red CAN deben ser interpretados para comprobar su veracidad, sentido y coherencia. Además, es necesario analizar el comportamiento del vehículo y la red ante distintas situaciones a manera de prueba con el dispositivo.

2.1.2.4 Método experimental

Este proyecto utiliza el método experimental, ya que su desarrollo conlleva observación, manipulación, ensayos, pruebas y verificaciones que se realizan conforme se desarrolla el dispositivo. Además, en la elaboración del código de programación para el microcontrolador, es necesario de manera experimental verificar constantemente la operabilidad y efectividad para evitar errores en la adquisición y almacenamiento de datos.

2.1.2.5 Método explicativo

El método explicativo es aquel que busca encontrar razones para explicar ciertos fenómenos, por eso, este proyecto considera este método debido a que utilizando el prototipo se busca responder a las interrogantes que surgirían al momento de un accidente vehicular por medio de la identificación de variables de importancia obtenidas de las Unidades Electrónicas de Control, que ayuden a responder interrogantes que surgen en un vehículo accidentado.

2.2 MATERIALES Y EQUIPOS

Para llevar a cabo el presente proyecto y la ejecución de las diversas pruebas; fue necesario el uso de un vehículo equipado con sistema OBD II y protocolo de comunicación CAN, donde se pueda implementar el dispositivo propuesto para la adquisición de los datos de la RED CAN. Para el desarrollo del dispositivo fueron necesarios los siguientes materiales:

2.2.1 CAN Bus Analyzer

El dispositivo CAN Bus Analyzer está diseñado para ser un monitor de bajo costo y fácil de usar, utilizado para el desarrollo y depurado de una RED CAN de alta velocidad (Mircrochip, 2019). Esta herramienta dispone de una amplia gama de funciones lo cual permite ser utilizada en varios segmentos del mercado que haga uso de protocolo de comunicación CAN.

Figura 2.1 Dispositivo CAN Bus Analyzer

(Mircrochip, 2019)

CAN Bus Analyzer	
Características	
Admite el protocolo CAN 2.0 e ISO11898-2	Abarca una interfaz de usuario de PC intuitiva para funciones como: filtro, registro, rastreo y transmisión.
Acceso directo a las señales CAN H y CAN L, CAN TX y CAN RX con la finalidad de obtener una depuración robusta.	Compatibilidad con el microcontrolador PIC de Microchip.
Opciones flexibles de interfaz de bus CAN.	Control de software de resistencia de terminación y pantalla LED.

Tabla 2.1 Características del CAN Bus Analyzer

Este dispositivo se conecta a un vehículo mediante un conector RS232C, además la velocidad de trasmisión de datos del analizador bus CAN es de hasta un bit.

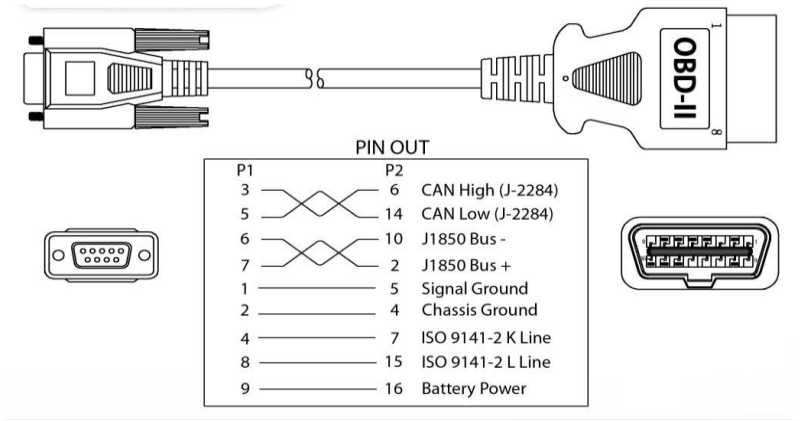


Figura 2.2 Conector RS232C a OBD II

(Mircrochip, 2019)

2.2.2 ELM 327 CAN bus scanner

El ELM 327 CAN scanner es un dispositivo utilizado para la trasmisión de datos, el cual se conecta mediante el puerto de diagnóstico OBD II con la Unidad de Control Electrónico (ECU), haciendo fácil la lectura de datos y enviarlos mediante Bluetooth a distintos dispositivos.

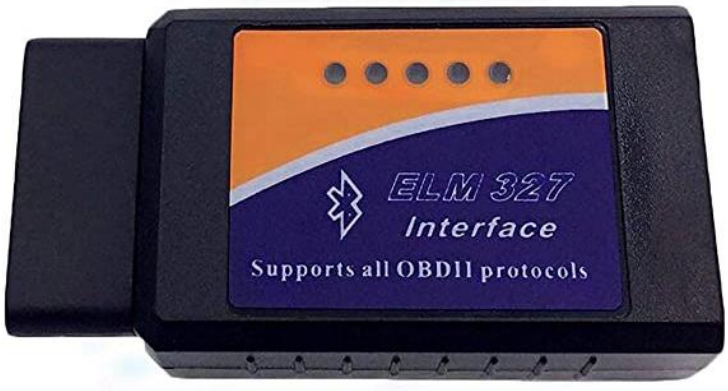


Figura 2.3 Dispositivo ELM 327 CAN bus scanner

(TodoMicro, s/f)

Además, este dispositivo puede ser usado con un microcontrolador donde se emplean diferentes librerías para interactuar con el escáner mediante comandos AT. Así mismo trabaja con diferentes protocolos de comunicación soportados por firmware v1.5 como son:

- SAE J1850 PWM
- SAE J1850 VPW
- ISO 14230-4 KWP
- ISO 9141-2
- ISO 15765-4 CAN

ELM 327 CAN bus scanner		
Características		**Funciones**
Conexión mediante Bluetooth	Conexión inalámbrica ISO 9141, KWP2000	-RPM -Temperatura del refrigerante -Cálculo del valor de carga -Ajuste de combustible a largo plazo - Presión del múltiple - Flujo de aire -Temperatura del aire -Posición del acelerador -Estado del sistema de combustible
Compatible con SAE J1850 y CAN bus	Software incluido para la palma, PDA	

Tabla 2.2 Características y funciones del dispositivo ELM 327 CAN bus scanner

2.2.3 Arduino Mega 2560

Arduino Mega 2560 es una placa basada en el microcontrolador ATmega 2560, la cual dispone de 54 entradas y salidas digitales donde 15 pueden ser usadas como salidas Pulse Wide Modulation o PWM además de 4 puertos seriales por hardware, un cristal de 16 MHz, una conexión USB, un botón reset y conectores: de alimentación y de programación ICSP (Arduino, s/f).

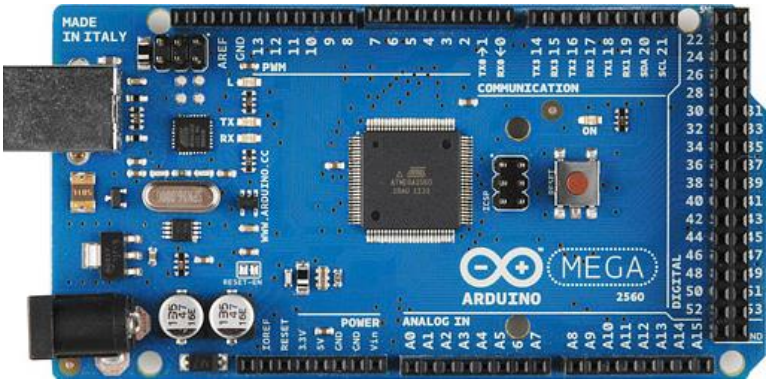


Figura 2.4 Arduino Mega 2560

(Arduino, s/f)

Arduino Mega 2560		
Características		**Funciones especializadas**
Dimensiones	**Voltajes**	- Serial: Se usa para recibir (RX) y transmitir (TX) datos en serie TTL. -Interrupciones externas: - PWM - SPI - LED - TWI
Longitud: 101.52 mm Ancho: 53.3 mm Peso: 37 g	-Voltaje de operación: 5V -Voltaje de entrada recomendado: 7-12V -Voltajes de entrada mínimo y máximo:6-20V	
Pines de conexión	**Corriente**	
-Pines de E/S digital: 54 de los cuales 15 son salida PWM. -Pines de entrada analógica: 16	-Corriente CC por cada pin E/S: 20mA -Corriente CC para el pin de 3.3V: 50mA	
-Microcontrolador: ATmega2560. -Memoria Flash: 256KB, donde 8 KB son usados por el gestor de arranque. -SRAM: 8KB -EEPROM: 4KB -Frecuencia de reloj: 16MH.		

Tabla 2.3 Características y funciones especializadas del Arduino Mega 2560

2.2.4 CAN – Bus Shield V2 MCP2515

Este shield permite a la placa Arduino comunicarse mediante el bus CAN, haciendo uso del controlador CAN MCP 2515 de Microchip con el transceptor CAN MCO2551 por medio de un estándar de 9 vías sub-D, donde este shield admite realizar sondeos entre la ECU de

información incluyendo datos como: temperatura del refrigerante, la velocidad del vehículo, rpm del motor y posición del acelerador (MICROJMP, s/f).

Figura 2.5 CAN – Bus Shield V2 MCP2515

(MICROJMP, s/f)

CAN – BUS Shield V2 MCP 2515	
Características	
Interfaz SPI de 10 MHz	Bus CAN v2.0B (1Mb/s)
Conexión CAN mediante conector sub-D de 9 pines	Conector para GPS tipo EM406
Conexión estándar para cable OBD II	Zócalo para tarjetas de memoria Micro SD
Protección de polaridad	Botón de RESET

Tabla 2.4 Características del CAN – Bus Shield V2 MCP2515

2.2.5 Modulo Micro SD lector tarjeta Arduino

Este módulo es un lector de tarjetas microSD, permitiendo la transferencia de datos hacia y desde una tarjeta microSD estándar, mediante un almacenamiento de datos. Debido a su capacidad y disposición de tamaño es utilizado para diferentes dispositivos, el módulo lleva incorporado un regulador de tensión donde puede alimentarse de 3.3 V a 5V (LLamas, 2016).

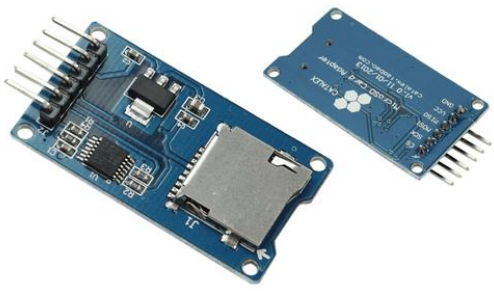


Figura 2.6 Módulo Micro SD lector tarjeta Arduino

(LLamas, 2016)

Módulo Micro SD lector tarjeta Arduino	
Características	
Voltaje de alimentación: 3.3V o 5V	Regulador de tensión incorporado 3.3V
Consumo <200mA	Interfaz:SPI
Dimensiones: 24 mm x 42 mm x 12 mm	

Tabla 2.5 Características del Módulo Micro SD lector tarjeta Arduino

2.2.6 Conector Socket OBD2

Es un conector de diagnóstico a bordo OBD2 estandarizado por la norma J1962, se utiliza mediante la interfaz del escáner de comunicación del vehículo, donde podremos tener comunicación con los diferentes módulos.

Figura 2.7 Conector Socket OBD2

(Microchip, 1999)

Además, la configuración de los pines del conector OBD2 se determina dándole utilidad al pin dependiendo del protocolo utilizado.

2.2.7 Módulo RTC DS1302

Este módulo es un reloj en tiempo real RTC (Real Time Clock), el cual consiste en un circuito integrado alimentado por una batería, permitiendo registrar en todo momento la

fecha: día, semana, mes y hora. Además, lleva integrado un cristal de cuarzo de 32.769 kHz para contar los segundos con una excelente precisión (Tecnopura, s/f).

Figura 2.8 Módulo RTC DS1302

(Tecnopura, s/f)

Módulo RTC DS1302	
Especificaciones Técnicas	
Voltaje de operación: 2V a 5.5V	Pines de conexión directa: VCC, GND, CLK, DAT y RST
Temperatura de operación: 0°C a 70°C	Interfaz de comunicación: I2C
Batería requerida: CR2032	Dimensiones: 43 mm x 22 mm x 11 mm

Tabla 2.6 Especificaciones técnicas del Módulo RCT DS1302

Su funcionamiento se basa en una interfaz de comunicación I2C, donde se establece la comunicación con tarjetas de desarrollo o microcontroladores, de tal manera este tipo de módulo son ideales para proyectos donde se requiera la utilización de hora y fecha para obtener eventos de puntualidad y exactitud a lo largo del tiempo.

2.3 PROCESOS METODOLÓGICOS

En la Figura 2.9 se presenta un flujograma, sobre el desarrollo del presente trabajo de investigación, donde se resume todos los procesos a llevar cabo para el desarrollo de un dispositivo basado en microcontrolador Arduino para la adquisición de datos a través de la red can para conocer las condiciones que suscitan accidentes vehiculares.

El flujograma inicia con una recopilación de datos bibliográficos, donde se obtiene la información necesaria para llevar a cabo el marco teórico. Considerando los temas más relevantes que de alguna forma son parte del proceso para el desarrollo de la presente investigación, posteriormente se lleva a cabo un análisis de las variables de seguridad activa

y pasiva del vehículo con la finalidad de parametrizar los tipos de datos que son necesarios para el estudio. Con esta información se realiza la selección del vehículo considerando que este cuente con protocolo de comunicación CAN.

Posterior se realiza un análisis del vehículo seleccionado haciendo uso de un escáner automotriz, con la finalidad de determinar los módulos que se encuentran disponibles en el vehículo y a su vez realizar una selección de estos, lo cuales serán de utilidad para el estudio. Cada uno de los módulos tiene un flujo de datos que brinda información como: valor cálculo de la carga, velocidad del motor, aceleración transversal, switch del freno entre otros. De tal manera se llevará a cabo la elección de estos PIDs considerando la información que ofrecen y su utilidad para el proyecto.

Para la extracción de estos flujos de datos se considera la conexión en paralelo del escáner automotriz y un analizador CAN, por medio de esta conexión se extraen los PIDs sea estándar y fabricante, considerando las variables seleccionadas para el estudio. Realizado este proceso se lleva a cabo la decodificación de los PIDs de cada una de las tramas en función de cada variable, debido que el flujo de datos que se extrae de la red CAN se presenta en formato hexadecimal, lo cual es difícil de interpretar. Posteriormente se plantea definir los rangos de disparo, considerando los datos de las aceleraciones transversales y longitudinales. Con este umbral de valores se desarrolla una interfaz para el microcontrolador con la finalidad de generar la comunicación entre el dispositivo y la red CAN. Obteniendo la lectura, escritura y almacenamiento de los datos en sistema de almacenamiento.

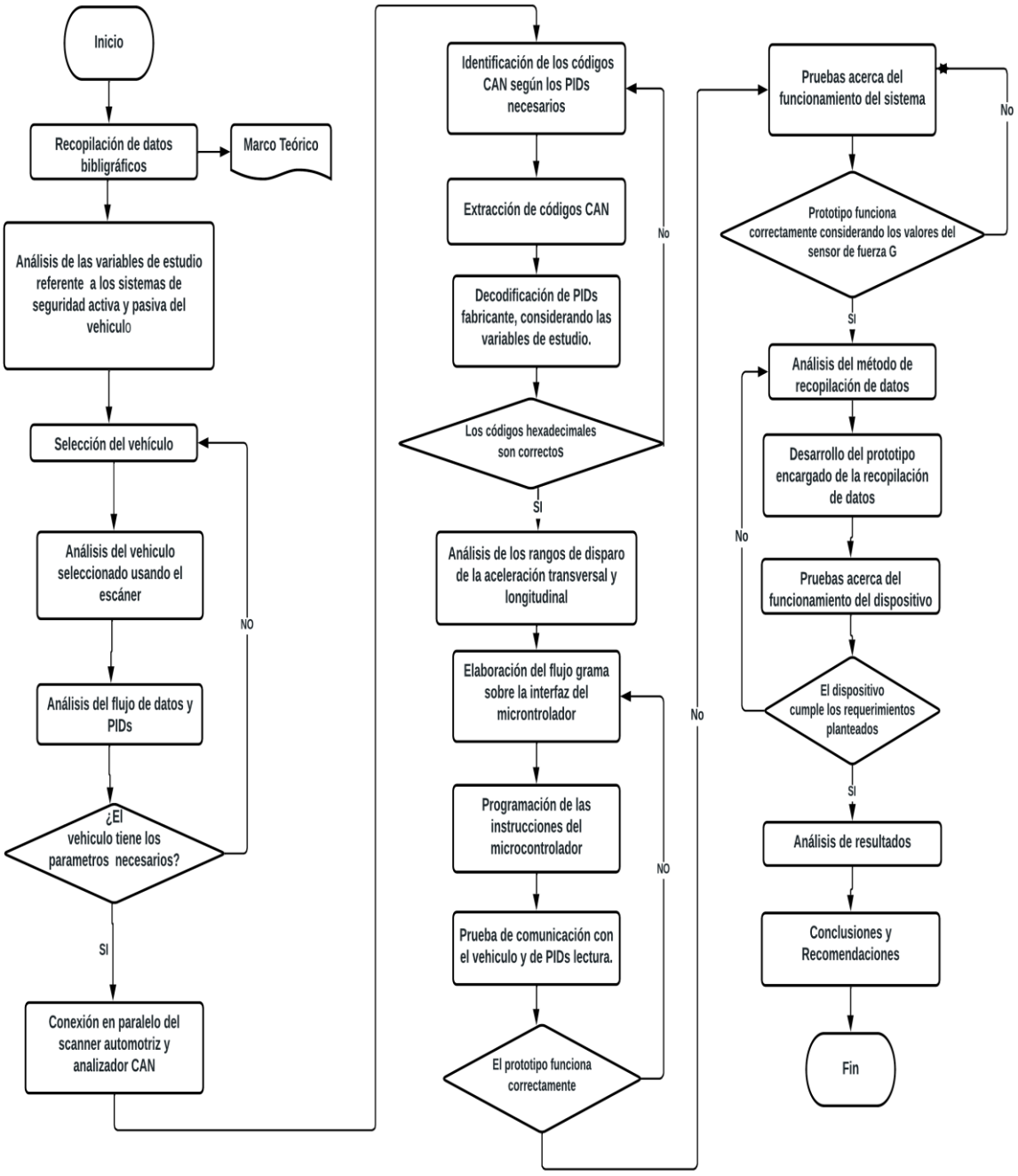


Figura 2.9 Flujograma de proceso metodológico

2.3.1 Recopilación de datos bibliográficos

Una fase inicial del presente proyecto es la recopilación de datos bibliográficos, con la finalidad de estructurar el marco teórico, debido que es fundamental conocer varios temas, lo cuales son relevantes para el desarrollo del trabajo de grado. Esta recopilación de información se ejecutó mediante el uso de la herramienta global “internet”, considerando

artículos científicos, trabajos de grado y páginas web de diversos autores. Con la finalidad de fundamentar cada uno de los temas añadidos al Capítulo I que están ligados al tema de desarrollo.

2.3.2 Análisis de las variables de estudio referente a los sistemas de seguridad activa y pasiva del vehículo.

El vehículo dispone de sistemas de seguridad activa y pasiva, de acuerdo con el tipo de funcionamiento se encargará de prevenir accidentes o reducir posibles lesiones en los ocupantes del vehículo durante el suceso. En estos sistemas de seguridad se encuentran sensores y actuadores los cuales forman parte de cada uno de ellos y son necesarios para su funcionamiento.

Por lo tanto, el vehículo seleccionado deberá constar con al menos un sistema de seguridad activa y pasiva comandados por una Unidad Electrónica de Control. Es necesario hacer uso de los distintos sensores que componen estos sistemas, para obtener datos que serán útiles posteriores a la colisión, mediante la siguiente tabla se especifica los sensores anexos a cada sistema determinando si su información es relevante para el estudio.

Sistemas de Seguridad Activa y Pasiva Controlados Electrónicamente			
Sistema	Sensores	Seguridad Activa	Seguridad Pasiva
Antibloqueo de frenado	Velocidad de las ruedas	X	
	Velocidad del vehículo		
	Switch del Freno		
	Presión del frenado		
Control Electrónico de Estabilidad	Ángulo y torque de dirección o volante	X	
	Velocidad de la rueda		
	Velocidad del vehículo		
	Presión del frenado		
	Switch del freno		
	Ángulo de giro		
	Aceleración transversal		

Dirección Electrónicamente Asistida	Ángulo y torque dirección del volante	X	
	Velocidad del vehículo		
Retención Suplementaria	Impacto		X
	Aceleración transversal		
	Ángulo de giro		
	Velocidad del vehículo		

Tabla 2.7 Sistemas de seguridad activa y pasiva

2.3.4 Selección del vehículo

Para la selección del vehículo se consideraron varios aspectos importantes, siendo el principal que el vehículo disponga de un sistema de diagnóstico a bordo OBDII, posteriormente haciendo uso de un multímetro automotriz, se midió el valor de las resistencias de terminación del par trenzado de cables de la red correspondiente a 60 ohmios, que en teoría tienen todos los vehículos con red multiplexada CAN en los pines 6 y 14 del puerto DLC. Una vez comprobado el valor de la resistencia de terminación de la red del vehículo, se utilizó un osciloscopio automotriz con dos canales para verificar las señales de la línea CAN High y CAN Low proveniente de los pines 6 y 14 respectivamente.

Identificado el conector DLC correspondiente al sistema de diagnóstico a bordo OBDII del vehículo, se conecta un escáner automotriz multimarca, con la finalidad de comprobar el protocolo de comunicación que emplea esta herramienta para comunicarse con la red del vehículo y constatar la disponibilidad de la red CAN. En este caso el protocolo de comunicación que selecciono el escáner automáticamente es el protocolo ISO 15765-4 (CAN).

Finalmente se realiza un escaneo rápido para identificar los módulos disponibles en la red del vehículo, es necesario que este cuente con al menos dos Unidades Electrónicas de Control, donde se encuentre información a través de la red CAN que resulte de utilidad para conocer las condiciones en que se encontraba operando el vehículo al momento de suscitarse un posible accidente, en este caso los módulos encontrados son: ECM (Motor), ABS (Frenos), BCM (Carrocería), AIRBAG, HVAC (Climatización), EPS (Dirección electrónica), IPDM (Modulo de Fusibles inteligente), CHASIS CONTROL , METER M&A (Tablero de instrumentos).

2.3.3 Análisis del vehículo seleccionado usando escáner

Una vez que se ha seleccionado el vehículo y determinado los módulos disponibles para el estudio, se realiza un análisis de cada uno en función de las variables que ofrecen determinando el grado de utilidad referente a la información que se considere importante. Para esto es necesario revisar el flujo de datos correspondiente a cada módulo (Modo 01 de diagnóstico), omitiendo los módulos que se consideran irrelevantes.

Módulos	Descripción	Necesario SI/NO
ENGINE	Se encuentran datos de los dispositivos electrónicos que controlan el motor como: régimen del motor, revoluciones, velocidad del vehículo, entre otros.	SI
ABS	Se encuentran datos de los dispositivos electrónicos que conforman el sistema antibloqueo del frenado como: válvula distribuidora de frenado, sensor de velocidad de las ruedas, entre otros.	NO
METER/M&A	Se encuentran variables que envían información de los diferentes módulos hacia el tablero.	NO
BCM	Se encuentran datos de los distintos que intervienen en la seguridad del vehículo.	NO
AIR BAG	Se encuentran valores específicos del sistema de retención suplementario.	NO
HVAC	Encontramos valores referentes a climatización y aire acondicionado.	NO
EPS/DAST 3	Se presentan datos del sistema de dirección electrónicamente asistida.	NO
IPDM E/R	Se encuentran datos de la fusilera inteligente.	NO
CHASIS CONTROL	Se encuentran variables que envían los diferentes módulos de los sistemas de seguridad activa y pasiva correspondientes al funcionamiento general del vehículo.	SI

Tabla 2.8 Módulos disponibles y su utilidad para el estudio

2.3.4 Análisis del flujo de datos y PIDs

Dentro del flujo de datos de los módulos escaneados, se realiza una selección de variables de utilidad en función de la información que suministra cada una de ellas para

posteriormente, identificar el PID correspondiente a las variables seleccionadas utilizando un dispositivo red CAN ANALYZER.

Módulo de Control Electrónico (ENGINE)			
Nombre	Descripción	Unidad	Necesario SI/NO
A/F ALPHA-B1	Sensor O2	%	NO
ATOM PRES SEN	Sensor MAP	V	NO
A/F S1 HTR(B1)	Sensor O2	%	NO
B/FUEL SCHDL	Ancho de pulso del Inyector	Ms	NO
ACCEL SEN1	Sensor APP	V	NO
ACCEL SEN2	Sensor APP	V	NO
AC PRESS SEN	Sistema de aire acondicionado	V	NO
A/F ADJ-B1	Sensor O_2	-	NO
A/F-S ATMSPHRC CRCT B1	Sensor O_2	-	NO
A/F-S ATMSPHRC CRCT UP B1	Sensor O_2	-	NO
AIR COND SIG	Sistema de aire acondicionado	-	NO
AIR COND RLY	Sistema de aire acondicionado	-	NO
ALT DUTY SIG	-	-	NO
COOLANT TEMP/S	Sensor ECT	deg/ree/C	NO
Battery Voltage	Voltaje de Batería	V	NO
CAL/LD VALUE	Valor calculado de la carga	%	SI
H02S2 (B1)	Sensor de O_2	V	NO
ENGINE OIL TEMP	Sensor de temperatura del aceite del motor	deg/ree/C	NO

FAN DUTY	Porcentaje de trabajo electroventilador	%	NO
ENGINE SPEED	Revoluciones del motor	rpm	SI
CLSD THL POS	Sensor TPS	-	NO
HO2S2 MNTR (B1)	Sensor de O_2	-	NO
HEATER FAN SW	Switch de la calefacción	-	NO
IDL A/V LEARN	Aprendizaje del volumen al aire en ralentí	-	NO
BRAKE SWITCH	Switch del freno		SI
FUEL PUMP RELAY	Relay de la bomba de combustible	-	NO
HO2S2 HTR (B1)	Sensor O_2 (Calefactor)	-	NO
H02 SE DIAG2(B1)	Estado Sensor O_2	-	NO
Intake temperatura sensor	Sensor IAT	C	NO
INT/V SOL (B1)	Voltaje del solenoide de arranque	%	NO
THRTL STK CNT B1	-	-	-
INJ PULSE – B1	Pulso del inyector	Msec	NO
MASS AIRFLOW	Sensor MAF	g/s	NO
TP SEN 1-B1	Sensor TPS 1	V	NO
TP SEN 2- B1	Sensor TPS 2	V	NO
MASS AIR FLOW SENSOR (Hz)	Sensor MAF (Frecuencia)	-	NO
LOAD SIGNAL	-	-	-
PW/ST SIGNAL	-	-	-
P/N POSI SW	-	-	-
START SIGNAL	Señal de encendido	-	NO
IGNITION SWITCH	-	-	-
THRTL RELAY	Relevador motor mariposa	-	NO
VHCL SPEED SE	Sensor VSS	km/h	SI
TRVL AFTER MIL	Distancia con luz MIL ON	km	SI
VEHICLE SPEED	Sensor VSS	km/h	SI

Tabla 2.9 Módulo de Control Electrónico (ENGINE)

CHASIS CONTROL			
Nombre	Descripción	Unidad	Necesario SI/NO
CAN DIAG STATUS	-	-	-
ACCELE PEDAL POSITION	Sensor APP	%	NO
BRAKE SW2	Switch del freno 2	-	NO
BRAKE SW1	Switch del freno 1	-	SI
ABS	Sistema ABS	-	SI
ABS MALF	Estado sistema ABS	-	SI
ATC1	-	-	-
ATC2	-	-	-
ATC4	-	-	-
BRAKE HOLD	Tiempo de frenado	-	SI
ATC3	-	-	-
ATC DISP	-	-	-
ATC 5	-	-	-
BRAKE HOLD DISP	-	-	-
ATC SETTING	-	-	-
AEB SETTING	-	-	-
IGN VOLT	Voltaje de encendido	V	NO
CONTROL MODULE MALF	-	-	-
FR WHEEL SPEED	Velocidad rueda frontal derecho	Rpm	SI
FL WHEEL SPEED	Velocidad rueda frontal izquierdo	Rpm	SI
RR WHEEL SPEED	Velocidad rueda posterior derecho	Rpm	SI
RL WHEEL SPEED	Velocidad rueda posterior izquierdo	rpm	SI
PRESS SENSOR	Sensor MAP	Bar	NO

EBD	-	-	-
FL TIRE DISP	-	-	-
FR TIRE DISP	-	-	-
RL TIRE DISP	-	-	-
RR TIRE DISP	-	-	-
INTERRUPT DISP	-	-	-
Vehicule speed	Sensor VSS	km/h	SI
STEERING ANG SENSOR	Ángulo de la dirección	Deg	SI
SIDE G SENSOR	Sensor de fuerza transversal	G	SI
YAW RATE SENSOR	Sensor de giro o guiñada	deg/s	SI
Throttle control	-	-	-
STOP LAMP SW	-	-	-
TCS	-	-	-
VDC	-	-	-
VDC MALF	-	-	-
VDC OFF SWITCH	-	-	-
VEHICLE DISP	-	-	-
TURN DISP	-	-	-
DECEL G SENSOR	-	G	SI

Tabla 2.10 Módulo del Chasis Control

2.3.5 Conexión en paralelo del escáner automotriz y analizador CAN.

Para identificar los PIDs que envía la herramienta de diagnóstico a los distintos módulos del vehículo, es necesario conectar un ANALIZADOR DE RED CAN (CAN ANALYZER) en paralelo al escáner por medio de un divisor (SPLITER) del conector DLC.

2.3.6 Identificación de los PIDs necesarios según el análisis del flujo de datos

De la lista de variables seleccionadas se muestran las Tablas 2.9 y 2.10, posteriormente se identificarán y clasificarán las variables que se adquieren utilizando PIDs estándar y fabricante de los módulos de control electrónico seleccionados.

2.3.6.1 Identificación de PIDs estándar

Utilizando la Tabla 1.1 se identifican los PIDs estándar y las fórmulas de conversión respectivas para las variables seleccionadas enlistadas en la Tabla mostrada a continuación.

Módulo de Control Electrónico (ENGINE)				
Nombre	Descripción	Unidad	PID	Fórmula
CAL/LD VALUE	Valor calculado de la carga	%	04	A/2.55
ENGINE SPEED	Revoluciones del motor	rpm	0C	(256A+B) /4
VHCL SPEED SENSOR	Sensor VSS	km/h	0D	A
TRVL AFTER MIL	Distancia con luz MIL ON	km	21	265A+B

Tabla 2.11 Identificación de PIDs genéricos o estándar.

Para establecer una comunicación con la computadora del vehículo y enviar el PID requerido a través de la red CAN utilizando el microcontrolador Arduino no solo es necesario conocer el PID sino también, la estructura completa e ID de la trama en la cual el PID será enviado. Se sabe que dentro de la programación propia de un escáner genérico existe dicha información, para extraerla, fue necesario utilizar una computadora con el software Hyperterminal “Hércules” y Can Analyzer de Microchip, además, un escáner genérico (ELM327) con conexión USB conectado en paralelo al DLC del vehículo con un módulo analizador de red CAN. La figura a continuación explica la conexión que se realizó entre hardware y software. Si bien es cierto las tramas CAN se envían a través de la red en grupos de bytes es decir en formatos de 1 y 0, sin embargo, el software CAN Analyzer agrupa estos datos en formato hexadecimal identificando cada parte de las tramas que viajan a través de la red para facilitar el entendimiento y compactar información, cada PID que se envía en síntesis es un byte expresado en formato hexadecimal, es decir una parte de la trama. Por esta razón fue necesario conocer el formato completo de la trama.

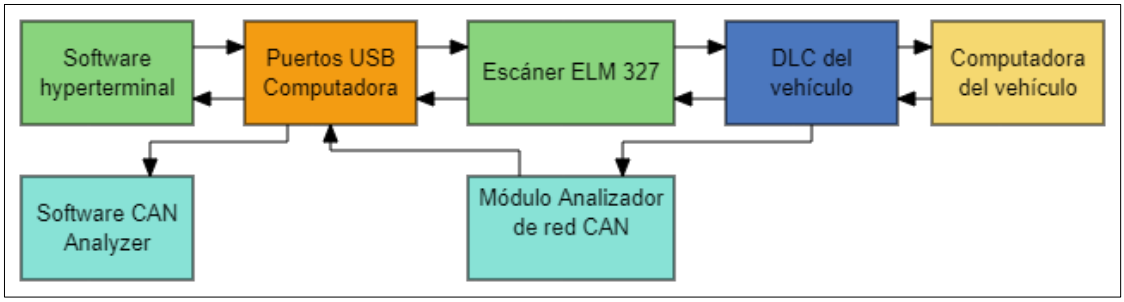


Figura 2.10 Esquema de conexión entre hardware y software al DLC del vehículo

Una vez abierto el software hyperterminal, se selecciona el puerto serial USB de la computadora a la cual estaba conectado el escáner ELM327, después en el cuadro de diálogo se escribe el comando "ATI" para confirmar la comunicación entre el escáner y la computadora, cuando el software confirma la comunicación se escribe cada PID para su análisis y mediante la tecla "enter" se envía el comando e inmediatamente el software recibe una respuesta de la computadora del vehículo. Esto se realizó con cada PID correspondiente a las variables seleccionadas de la Tabla 2.9

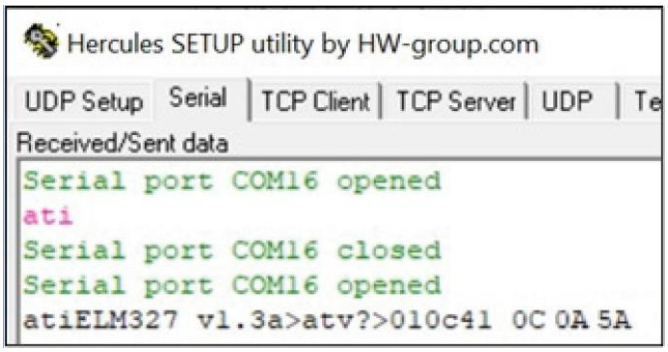


Figura 2.11 Solicitud y respuesta RPM del Motor mediante PID utilizando el hyperterminal.

Como se observa en la figura anterior, cuando se envía el PID correspondiente a RPM "01 0C" a través del hyperterminal la computadora del vehículo responde el valor de las RPM "41 0C 0A 5A" siendo los dos últimos Bits el valor expresado en Hexadecimal correspondiente a las revoluciones del motor, los cuales se les asigna la variable A y B respectivamente, luego con la fórmula de conversión a decimal definida en el estándar SAE J1939 se puede obtener el parámetro en formato decimal si así se lo requiere.

Todo evento que suceda en la red CAN del vehículo incluida la comunicación escáner-computadora será registrado en tiempo real y presentado en forma de matriz de tramas por

el software CAN Analyzer, se aprovechó esta característica del software para registrar el ID y la estructura de la trama que envía y recibe el escáner mediante el software hyperterminal a la red. La figura a continuación muestra la trama enviada y recibida por el escáner cuando se solicitó el PID de las RPM del motor mediante el software hyperterminal.

Figura 2.12 Análisis de la red utilizando el software CAN Analyzer ante la solicitud y respuesta de las RPM del Motor requeridas por el software hyperterminal mediante PID.

En la Figura 2.12 se observa la trama de solicitud y respuesta con su respectivo ID y estructura la última y penúltima fila respectivamente. Se sabe que estas filas corresponden al requerimiento del hyperterminal, ya que la última columna denominada "counter" muestra que el mensaje solo se envió y recibió una vez lo cual coincide con el número de veces que se envió la orden a través del hyperterminal. Se determinó que las tramas de datos mantienen la siguiente estructura:

ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
7DF	8	02	01	0C	00	00	00	00	00

Tabla 2.12 Estructura de la trama para el envío de PIDs estándar.

ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
7E8	8	04	41	0C	0A	5ª	00	00	00

Tabla 2.13 Estructura de la trama para recepción de datos.

ID. - Representa el identificador del mensaje, si es solicitud o respuesta y a qué módulo de la red va dirigido.

DLC. - Indica la cantidad de bytes de la trama.

DATA. - Indica la información de las condiciones de operación del vehículo utilizando un valor numérico hexadecimal.

La trama de datos tiene un identificador (ID), un indicador de longitud de datos (DLC) y ocho datos que van del cero al siete. La trama de solicitud tiene el ID "7DF" y el dato dos de esa trama lleva el PID que se requiere. La trama de recepción tiene el ID "7E8" y se sabe gracias al software hyperterminal que el dato tres y cuatro de esta trama, contienen la información de funcionamiento del vehículo a manera de respuesta a la trama de solicitud, en este ejemplo indican las revoluciones del motor.

Como se explica anteriormente, para obtener el valor de las RPM del motor en formato decimal, bastaría con asignar al dato tres y cuatro la variable A y B respectivamente y aplicar la fórmula de conversión definida por el estándar correspondiente al PID RPM, en este ejemplo la Tabla 2.11 indica que la fórmula es (256A+B) /4. Por lo tanto, se convierten A y B a decimal y se los inserta en la fórmula para obtener el valor RPM en formato decimal siendo 662 RPM, de esta manera se constata que el valor obtenido en la trama corresponde a la realidad.

2.3.6.2 Identificación de PIDs fabricante

Al igual que sucede con los PIDs genéricos, para poder enviar PIDs fabricante a través de la red también es necesario conocer el ID y la estructura completa de la trama en la que se envía el PID, sin embargo, los PIDs fabricante en esta ocasión no se conocen pues no están estandarizados como los PIDs genéricos, mucho menos se conoce su fórmula de conversión de hexadecimal a decimal. Para realizar esta tarea se hizo uso nuevamente del software CAN Analyzer, pero esta vez se utilizó un escáner original de la marca del vehículo seleccionado, en este caso la comunicación entre el escáner fabricante y la computadora del vehículo también será monitoreada por el módulo analizador de red CAN y el software CAN Analyzer en todo momento, debido a que esta herramienta es la única capaz de acceder al modo fabricante y solicitar datos en vivo mediante PIDs fabricante.

En este punto no fue necesario hacer uso del software Hyperterminal, debido a que las tramas de datos que envía el escáner original seleccionado a la red del vehículo no pueden ser controladas a voluntad como se hizo con el ELM327, utilizando el hyperterminal. Este escáner depende de su propia interfaz y su conexión es mediante Bluetooth a un dispositivo móvil. La Figura 2.13 explica la conexión que se realizó entre hardware y software.

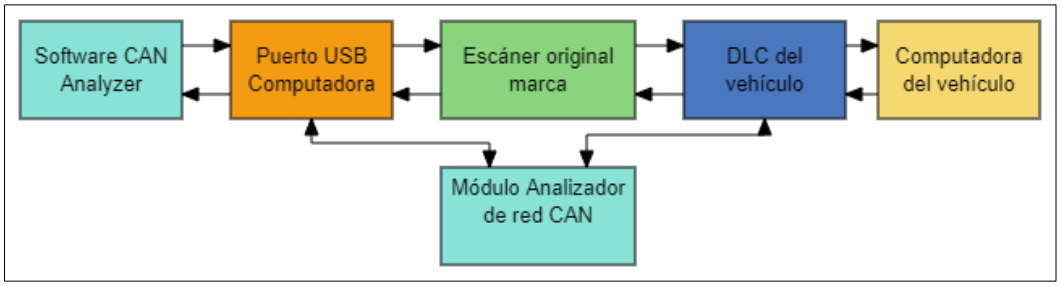
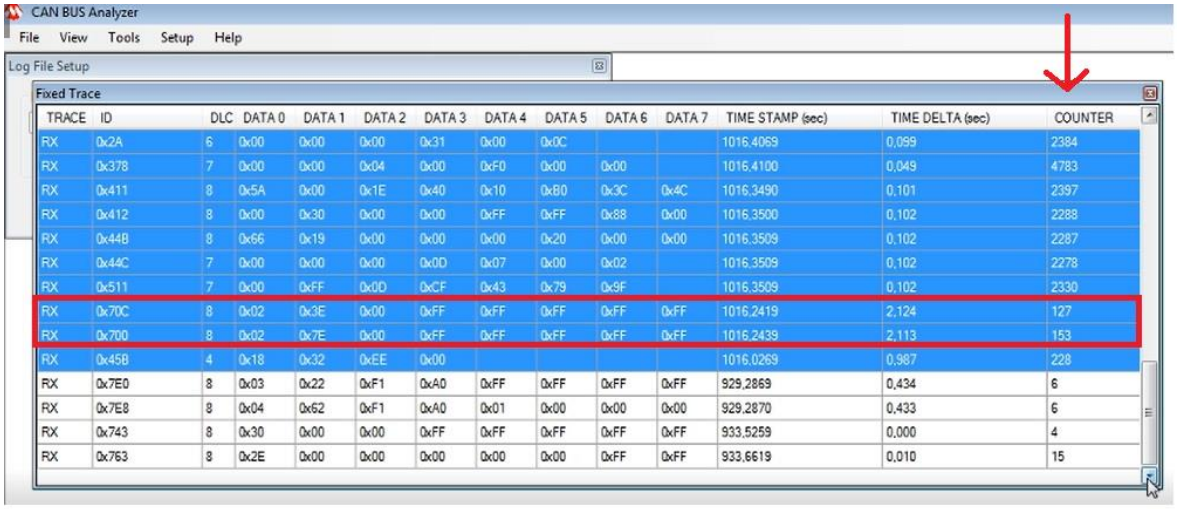


Figura 2.13 Esquema de conexión entre hardware y software al DLC del vehículo

Cabe mencionar que, el software CAN Analyzer no solo es capaz de recibir y plasmar tramas de datos sino también de transmitirlas, incluso sin la necesidad de utilizar el escáner genérico u original, solo hace falta saber que trama se va a enviar su ID, longitud y escritura. Mediante un cuadro de diálogo el usuario puede escribir la trama y enviarla a través de la red del vehículo. Además, el software ofrece la opción de grabar y guardar la matriz de tramas en formato CSV para su posterior análisis en otro software. Se utilizó esta última opción y el analizador de datos Excel, para analizar el envío de tramas que realizó el escáner fabricante al solicitar un escaneo rápido de los módulos disponibles en el vehículo utilizando su interfaz.

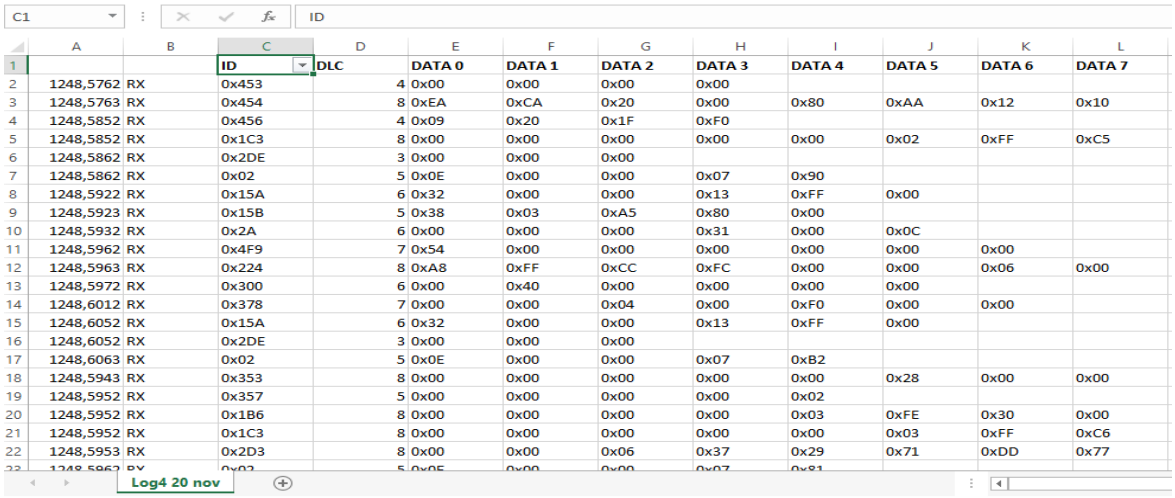
Mientras se escaneaba el vehículo por modo fabricante, se observó la matriz de tramas en el software CAN Analyzer, como se visualiza en la Figura 2.13, las tramas con el ID 0x70C y 0x700 se enviaron pocas veces (de acuerdo con el contador) durante el proceso de escaneo, entonces, se pudo asumir que esas tramas corresponden a la solicitud del escáner para la comunicación con la red del vehículo en modo fabricante. Los datos de las tramas observadas son cambiantes entorno al tiempo transcurrido y solo mantienen su ID pero sus datos varían Figura 2.14.



TRACE	ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7	TIME STAMP (sec)	TIME DELTA (sec)	COUNTER
RX	0x2A	6	0x00	0x00	0x00	0x31	0x00	0x0C			1016,4069	0,099	2384
RX	0x378	7	0x00	0x00	0x04	0x00	0xF0	0x00	0x00		1016,4100	0,049	4783
RX	0x411	8	0x5A	0x00	0x1E	0x40	0x10	0x80	0x3C	0x4C	1016,3490	0,101	2397
RX	0x412	8	0x00	0x30	0x00	0x00	0xFF	0xFF	0x88	0x00	1016,3500	0,102	2288
RX	0x44B	8	0x66	0x19	0x00	0x00	0x00	0x20	0x00	0x00	1016,3509	0,102	2287
RX	0x44C	7	0x00	0x00	0x00	0x0D	0x07	0x00	0x02		1016,3509	0,102	2278
RX	0x511	7	0x00	0xFF	0x0D	0xCF	0x43	0x79	0x9F		1016,3509	0,102	2330
RX	0x70C	8	0x02	0x3E	0x00	0xFF	0xFF	0xFF	0xFF	0xFF	1016,2419	2,124	127
RX	0x700	8	0x02	0x7E	0x00	0xFF	0xFF	0xFF	0xFF	0xFF	1016,2439	2,113	153
RX	0x45B	4	0x18	0x32	0xEE	0x00					1016,0269	0,987	228
RX	0x7E0	8	0x03	0x22	0xF1	0xA0	0xFF	0xFF	0xFF	0xFF	929,2869	0,434	6
RX	0x7E8	8	0x04	0x62	0xF1	0xA0	0x01	0x00	0x00	0x00	929,2870	0,433	6
RX	0x743	8	0x30	0x00	0x00	0xFF	0xFF	0xFF	0xFF	0xFF	933,5259	0,000	4
RX	0x763	8	0x2E	0x00	0x00	0x00	0x00	0x00	0xFF	0xFF	933,6619	0,010	15

Figura 2.14 Tramas de solicitud (0x70C) y respuesta (0x700) parara entrar al modo de escaneo por fabricante.

Por lo tanto, fue necesario grabar y guardar los datos transmitidos a la matriz en el software en formato CSV y usando el analizador de datos (EXCEL) como se muestra en la Figura 2.14.

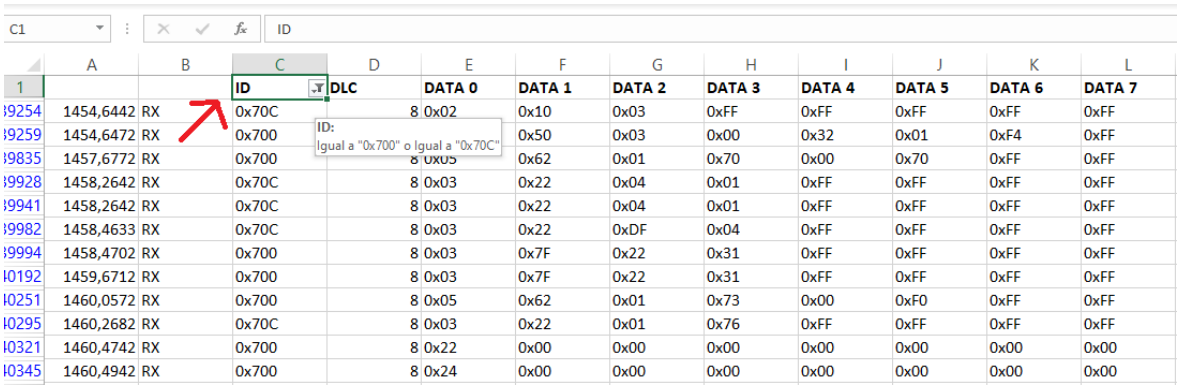


	A	B	C	D	E	F	G	H	I	J	K	L
1			ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
2	1248,5762	RX	0x453	4	0x00	0x00	0x00	0x00				
3	1248,5763	RX	0x454	8	0xEA	0xCA	0x20	0x00	0x80	0xAA	0x12	0x10
4	1248,5852	RX	0x456	4	0x09	0x20	0x1F	0xF0				
5	1248,5852	RX	0x1C3	8	0x00	0x00	0x00	0x00	0x00	0x02	0xFF	0xC5
6	1248,5862	RX	0x2DE	3	0x00	0x00	0x00					
7	1248,5862	RX	0x02	5	0x0E	0x00	0x00	0x07	0x90			
8	1248,5922	RX	0x15A	6	0x32	0x00	0x00	0x13	0xFF	0x00		
9	1248,5923	RX	0x15B	5	0x38	0x03	0xA5	0x80	0x00			
10	1248,5932	RX	0x2A	6	0x00	0x00	0x00	0x31	0x00	0x0C		
11	1248,5962	RX	0x4F9	7	0x54	0x00	0x00	0x00	0x00	0x00	0x00	
12	1248,5963	RX	0x224	8	0xA8	0xFF	0xCC	0xFC	0x00	0x00	0x06	0x00
13	1248,5972	RX	0x300	6	0x00	0x40	0x00	0x00	0x00	0x00		
14	1248,6012	RX	0x378	7	0x00	0x00	0x04	0x00	0xF0	0x00	0x00	
15	1248,6052	RX	0x15A	6	0x32	0x00	0x00	0x13	0xFF	0x00		
16	1248,6052	RX	0x2DE	3	0x00	0x00	0x00					
17	1248,6063	RX	0x02	5	0x0E	0x00	0x00	0x07	0xB2			
18	1248,5943	RX	0x353	8	0x00	0x00	0x00	0x00	0x00	0x28	0x00	0x00
19	1248,5952	RX	0x357	5	0x00	0x00	0x00	0x00	0x02			
20	1248,5952	RX	0x1B6	8	0x00	0x00	0x00	0x00	0x03	0xFE	0x30	0x00
21	1248,5952	RX	0x1C3	8	0x00	0x00	0x00	0x00	0x00	0x03	0xFF	0xC6
22	1248,5953	RX	0x2D3	8	0x00	0x00	0x06	0x37	0x29	0x71	0xDD	0x77

Figura 2.15 Matriz de tramas de la RED CAN almacenada en formato CSV sin filtro.

Posteriormente se realizó un filtrado de datos de la Figura 2.16 correspondientes a las tramas asumidas de la comunicación, con la finalidad de analizar el conjunto de datos que contenía cada trama con el mismo identificador. En la matriz de la misma figura se observó que la

trama con el ID 0x70C ocupó la primera fila, siendo esta la posible solicitud, la cual necesariamente tuvo que ser precedida por el ID 0x700 a manera de respuesta.



	A	B	C	D	E	F	G	H	I	J	K	L
1			ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
9254	1454,6442	RX	0x70C	8	0x02	0x10	0x03	0xFF	0xFF	0xFF	0xFF	0xFF
9259	1454,6472	RX	0x700	8	0x06	0x50	0x03	0x00	0x32	0x01	0xF4	0xFF
9835	1457,6772	RX	0x700	8	0x05	0x62	0x01	0x70	0x00	0x70	0xFF	0xFF
9928	1458,2642	RX	0x70C	8	0x03	0x22	0x04	0x01	0xFF	0xFF	0xFF	0xFF
9941	1458,2642	RX	0x70C	8	0x03	0x22	0x04	0x01	0xFF	0xFF	0xFF	0xFF
9982	1458,4633	RX	0x70C	8	0x03	0x22	0xDF	0x04	0xFF	0xFF	0xFF	0xFF
9994	1458,4702	RX	0x700	8	0x03	0x7F	0x22	0x31	0xFF	0xFF	0xFF	0xFF
0192	1459,6712	RX	0x700	8	0x03	0x7F	0x22	0x31	0xFF	0xFF	0xFF	0xFF
0251	1460,0572	RX	0x700	8	0x05	0x62	0x01	0x73	0x00	0xF0	0xFF	0xFF
0295	1460,2682	RX	0x70C	8	0x03	0x22	0x01	0x76	0xFF	0xFF	0xFF	0xFF
0321	1460,4742	RX	0x700	8	0x22	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0345	1460,4942	RX	0x700	8	0x24	0x00	0x00	0x00	0x00	0x00	0x00	0x00

Figura 2.16 Filtro de tramas como solicitud (0x70C) y respuesta (0x700).

Se utilizó nuevamente el software CAN Analyzer para transmitir la trama con el ID 0x70C considerada como solicitud y constatar que da como resultado la trama con el ID 0x700 tomada como respuesta. En las dos últimas filas de la matriz de tramas de la Figura 2.14, se observa que la trama con el ID 0x70C entra a la red desde el transmisor y efectivamente es respondida por la trama con ID 0x700 como se evidenció en el registro CSV cuando se ejecutó la comunicación con el escáner original y la computadora del vehículo.

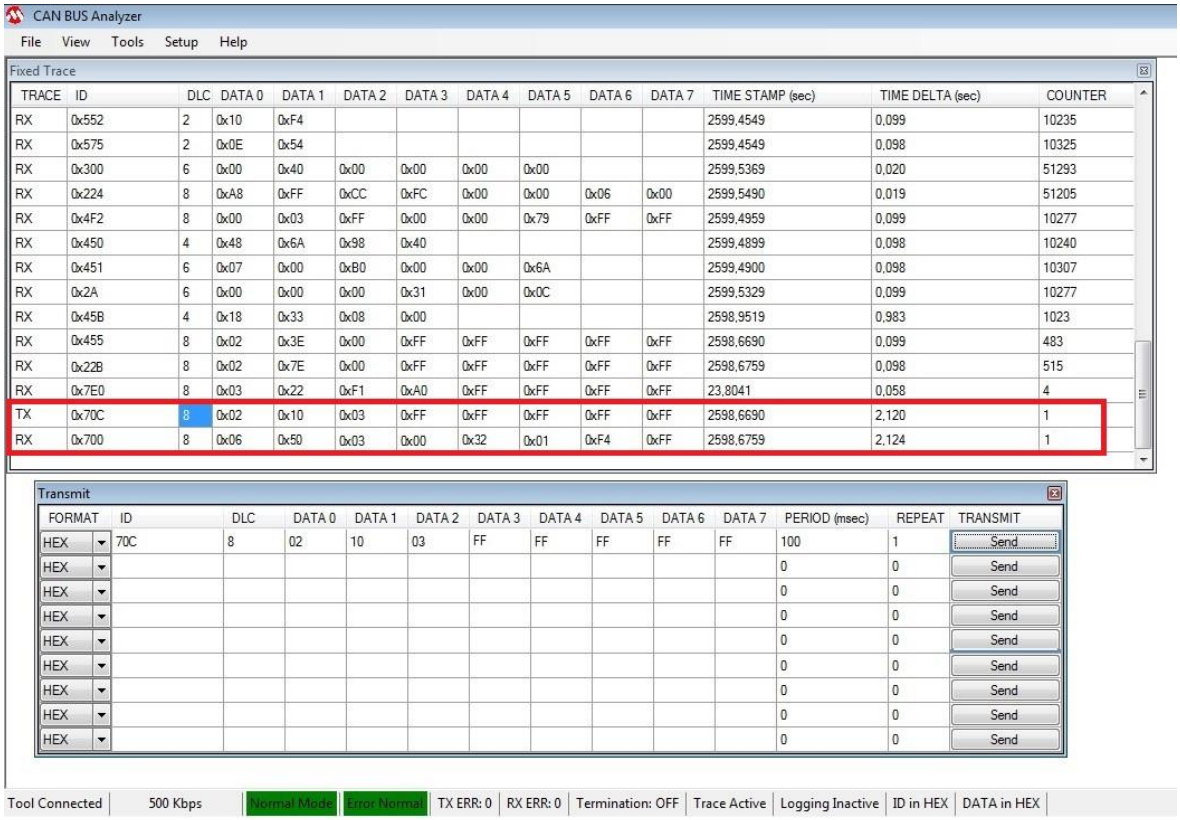


Figura 2.17 Prueba de transmisión solicitud de trama y respuesta.

Una vez filtrada las tramas en el orden correspondiente, se determinó la solicitud para un escaneo rápido de los módulos disponibles en modo fabricante, enviada por el escáner original de la marca, en la Tabla 2.14 se observa la solicitud completa. Es necesario enviar las tramas una a la vez en el orden correspondiente, para obtener acceso al igual que lo hace el escáner.

SOLICITUD PARA ESCANEO RAPIDO DE MODIULOS DISPONIBLES MODO FABRICANTE									
ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
70C	8	2	10	3	FF	FF	FF	FF	FF
70C	8	3	22	4	1	FF	FF	FF	FF
70C	8	3	22	DF	4	FF	FF	FF	FF
70C	8	3	22	1	76	FF	FF	FF	FF
70C	8	3	22	4	4	FF	FF	FF	FF
70C	8	3	22	4	7	FF	FF	FF	FF
70C	8	2	10	3	FF	FF	FF	FF	FF
70C	8	2	10	3	FF	FF	FF	FF	FF
70C	8	2	10	1	0C	FF	FF	FF	FF
70C	8	3	22	1	FF	FF	FF	FF	FF
70C	8	3	22	1	52	FF	FF	FF	FF
70C	8	3	22	1	80	FF	FF	FF	FF
70C	8	4	2F	DF	1	3	FF	FF	FF
70C	8	2	3E	0	FF	FF	FF	FF	FF

Tabla 2.14 Tramas de datos para solicitar PIDs en modo fabricante.

2.3.7 Extracción de códigos CAN

La Tabla 2.14 muestra la solicitud enviada por el escáner a través de la red para acceder al escaneo de los módulos disponibles y posteriormente al módulo denominado “CHASIS CONTROL”. A partir de ese momento, el escáner puede solicitar cualquier función de dicho módulo, por medio de su interfaz, incluida la función muestra de datos en vivo, la cual se utiliza para monitorear todos los sensores conectados al módulo en tiempo real. Utilizando la función muestra de datos en vivo se pudo solicitar el valor de cada sensor que esté conectado al módulo y analizar lo que envía el escáner a través de la red en cada solicitud que realiza, para esto se hizo uso del software CAN Analyzer nuevamente.

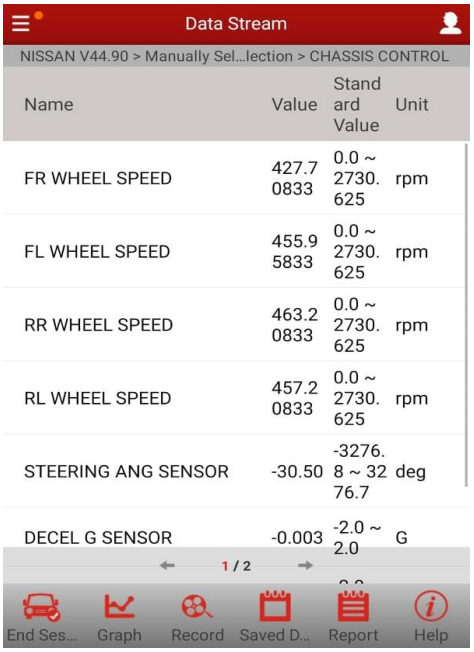


Figura 2.18 Muestra de datos en vivo del módulo CHASIS CONTROL

Las tramas denominadas como solicitudes mantienen la misma estructura para cada variable del módulo del CHASIS CONTROL, no obstante, solo el DATA 3 cambia su valor, por lo tanto, se pudo inferir que ese es el número del PID correspondiente a la variable solicitada, mientras que las respuestas son independientes a cada variable.

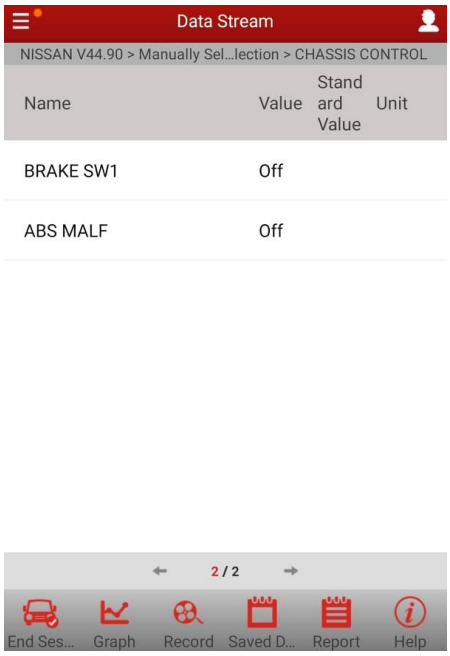


Figura 2.19 Muestra de datos en vivo del módulo CHASIS CONTROL

CHASIS CONTROL									
Nombre				**Descripción**				**Unidad**	
BRAKE SW1				Switch del freno 1				-	
				Solicitud del Switch					
ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
70C	8	3	22	1	22	FF	FF	FF	FF
				Respuesta Switch OFF					
700	8	5	61	1	22	0	C0	FF	FF
				Respuesta Switch ON					
700	8	5	62	1	22	40	C0	FF	FF

Tabla 2.15 Estructura de la Trama de solicitud y respuesta que abarca el PID del BRAKE SW1

La Tabla 2.15 muestra la solicitud y las respuestas de la variable switch del freno 1 “BRAKE SW1”, donde el DATA 3 tiene un PID “22”, mientras que las respuestas de la trama mantienen su misma estructura a diferencia del DATA 4 que muestra el estado del Switch: “0” OFF y “40” ON.

2.3.8 Decodificación de PIDs fabricante, considerando las variables de estudio

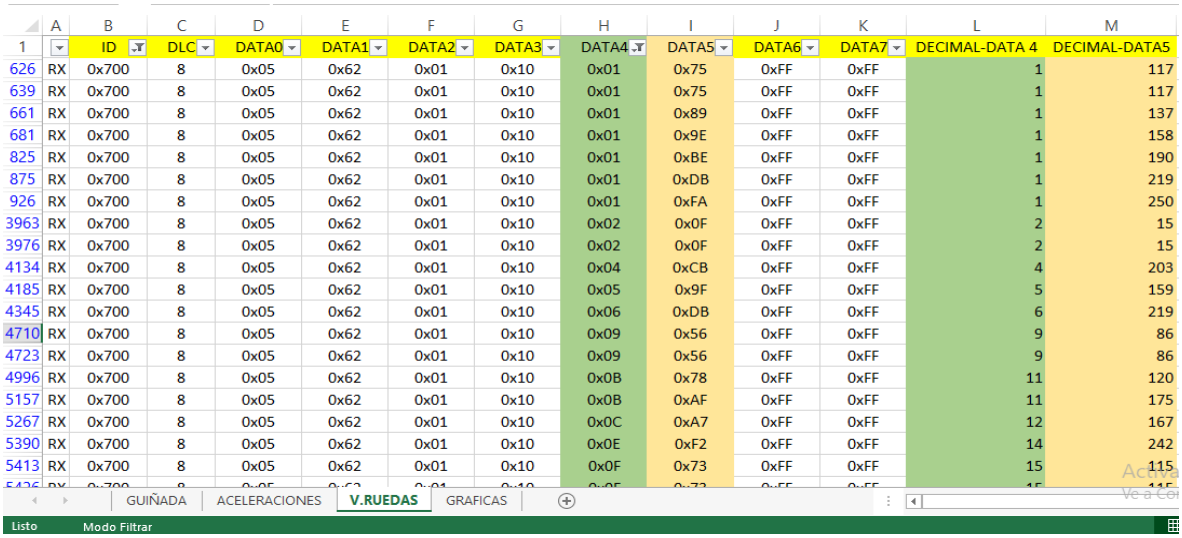
Una vez realizado el proceso de identificación de las tramas en modo fabricante, se lleva a cabo la decodificación de las tramas correspondientes a los PIDs considerando las variables de estudio, puesto que es necesario obtener un valor número decimal lógico interpretable para objeto de análisis. En el caso de la variable BRAKE SW1 su decodificación no fue necesaria, debido que el valor observado en DATA 5 solo puede tener dos estados 0x40 para ON que equivale presionado y 0x00 para OFF que equivale a no presionado, se observaron los mismos comportamientos para el estado del sistema ABS (ABS MALF).

Como se mencionó anteriormente en el DATA 4 y 5 de la trama de respuesta, se encuentran los valores numéricos correspondientes al valor de la variable solicitada en formato hexadecimal, fue necesario transformar estos valores a formato decimal para poder ser interpretado, sin embargo, una transformación directa para cada variable resultaba ilógica. Por medio de observación se infirió que el escáner original realiza una transformación de los valores de DATA 4 y 5 empleando formulas, para plasmar valores decimales entendibles por el usuario en su interfaz. Para llevar el proceso de decodificación se siguen los siguientes pasos:

- Solicitar la variable y la gráfica en tiempo real en el escáner.
- Grabar los datos con el DATA LOG.
- Extraer el data log y filtrar el ID 0x700.
- Identificar los picos de valores de la lista filtrada.
- Se analiza si las variables presentan valores positivos, negativo o ambos.
- Gracias a la observación se determinó que los valores negativos de las variables se cuentan en retroceso empezando desde el número hexadecimal 0xFF o decimal 255 para DATA 4 y 5, mientras que los valores positivos de las variables se cuentan en ascenso empezando desde 0x00. El valor máximo que puede albergar DATA 4 y 5 es 0xFF respectivamente, ya que este número es equivalente a un byte u 8 bits. Es necesario mencionar que cada DATA de la trama está compuesto máximo de un byte.
- Una vez identificado los picos, se convierten los valores de DATA 4 y 5 a decimales para su interpretación y se los compara con los valores de la gráfica del escáner, si existen una relación directa no es necesario encontrar una fórmula, en el caso de no encontrar dicha relación es necesario desarrollar una fórmula que permita la interpretación de los datos, normalmente las fórmulas mantienen la misma lógica de las fórmulas para PID estándar. Finalmente se compara los picos del DATA LOG con los picos de la gráfica del escáner con la finalidad de verificar los resultados que se obtienen.

2.3.8.1 WHEEL SPEED

En la Figura 2.20 se muestra el flujo de datos de la velocidad de las ruedas frontal y posterior, considerando que el DATA: 4 y 5 son los valores que cambian en función del incremento o disminución de las revoluciones del motor. Estos datos son convertidos a decimal para compararlos con los picos de la Figura 2.18 para llevar a cabo la deducción de la formula necesaria.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1		ID	DLC	DATA0	DATA1	DATA2	DATA3	DATA4	DATA5	DATA6	DATA7	DECIMAL-DATA 4	DECIMAL-DATA5
626	RX	0x700	8	0x05	0x62	0x01	0x10	0x01	0x75	0xFF	0xFF	1	117
639	RX	0x700	8	0x05	0x62	0x01	0x10	0x01	0x75	0xFF	0xFF	1	117
661	RX	0x700	8	0x05	0x62	0x01	0x10	0x01	0x89	0xFF	0xFF	1	137
681	RX	0x700	8	0x05	0x62	0x01	0x10	0x01	0x9E	0xFF	0xFF	1	158
825	RX	0x700	8	0x05	0x62	0x01	0x10	0x01	0xBE	0xFF	0xFF	1	190
875	RX	0x700	8	0x05	0x62	0x01	0x10	0x01	0xDB	0xFF	0xFF	1	219
926	RX	0x700	8	0x05	0x62	0x01	0x10	0x01	0xFA	0xFF	0xFF	1	250
3963	RX	0x700	8	0x05	0x62	0x01	0x10	0x02	0x0F	0xFF	0xFF	2	15
3976	RX	0x700	8	0x05	0x62	0x01	0x10	0x02	0x0F	0xFF	0xFF	2	15
4134	RX	0x700	8	0x05	0x62	0x01	0x10	0x04	0xCB	0xFF	0xFF	4	203
4185	RX	0x700	8	0x05	0x62	0x01	0x10	0x05	0x9F	0xFF	0xFF	5	159
4345	RX	0x700	8	0x05	0x62	0x01	0x10	0x06	0xDB	0xFF	0xFF	6	219
4710	RX	0x700	8	0x05	0x62	0x01	0x10	0x09	0x56	0xFF	0xFF	9	86
4723	RX	0x700	8	0x05	0x62	0x01	0x10	0x09	0x56	0xFF	0xFF	9	86
4996	RX	0x700	8	0x05	0x62	0x01	0x10	0x0B	0x78	0xFF	0xFF	11	120
5157	RX	0x700	8	0x05	0x62	0x01	0x10	0x0B	0xAF	0xFF	0xFF	11	175
5267	RX	0x700	8	0x05	0x62	0x01	0x10	0x0C	0xA7	0xFF	0xFF	12	167
5390	RX	0x700	8	0x05	0x62	0x01	0x10	0x0E	0xF2	0xFF	0xFF	14	242
5413	RX	0x700	8	0x05	0x62	0x01	0x10	0x0F	0x73	0xFF	0xFF	15	115

Figura 2.20 Data Log de WHEEL SPEED

En la Figura 2.21 se puede observar la gráfica sobre la velocidad de la rueda delantera izquierda adquirida por medio del escáner, esta se la considera importante debido que se orienta los resultados obtenidos tras el proceso de decodificación anteriormente mencionado. Los datos que se observan son las revoluciones de las ruedas en función del tiempo, para comprender como realiza el escáner el proceso de toma de datos de la matriz de tramas de WHEEL SPEED y como estos datos son registrados en la gráfica. Fue necesario realizar una aceleración para graficar un pico de la velocidad de las ruedas, debido que si se aumenta la aceleración esto será directamente proporcional con el incremento de los picos del WHEEL SPEED. En la gráfica se observa el pico mas alto que es de 280 rpm, en función de este pico obtenido se espera el mismo resultado con las fórmulas desarrolladas y para corroborar estos datos se toma en cuenta el segundo pico con un valor de 255 rpm.

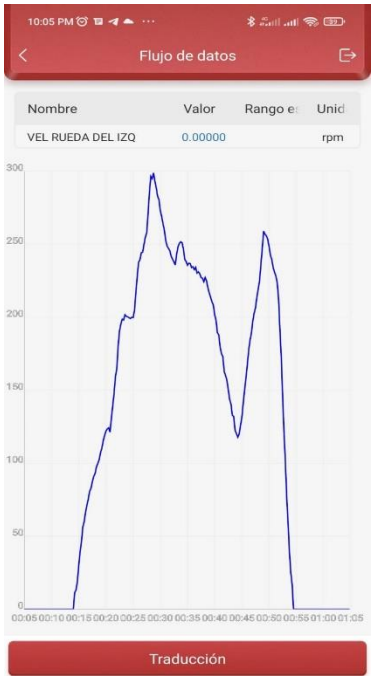


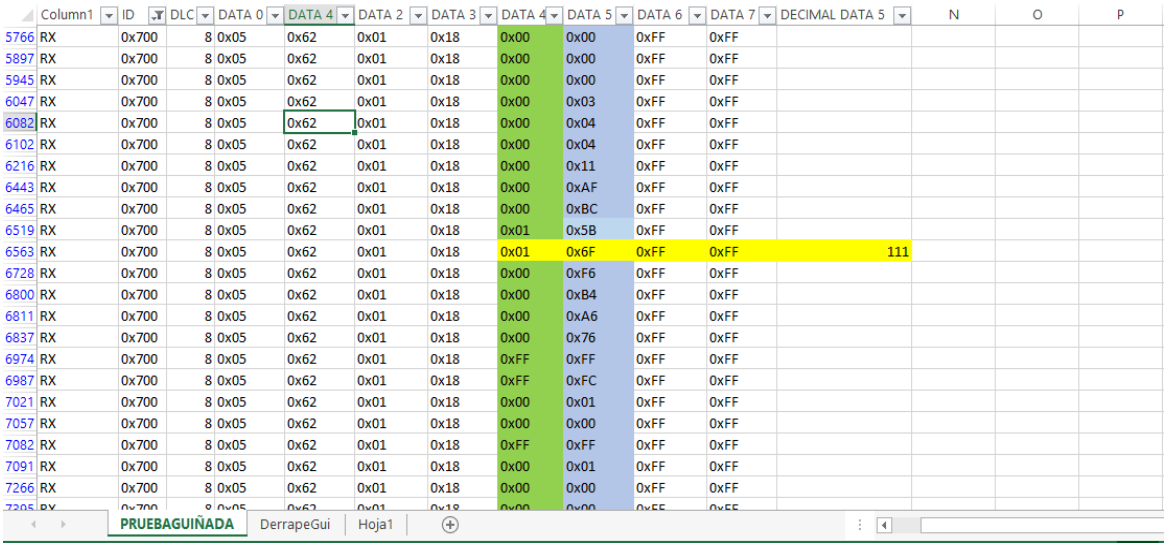
Figura 2.21 Señal del flujo de datos del escáner del WHEEL SPEED

2.3.8.2 STEERING ANG SENSOR

En el STEERING ANG SENSOR (Ángulo de dirección del volante) no fue necesario realizar un DATA LOG, debido a que los valores plasmados en el escáner coinciden directamente con la conversión decimal del DATA 4 y 5.

2.3.8.3 YAW RATE SENSOR

En la Figura 2.22 se muestra el flujo de datos del sensor de giro o guiñada, donde el DATA: 4 y 5 son las respuestas de la trama de datos, en función de esta consideración se decodifica los valores tomando en cuenta los picos de giro, siendo lo marcado con color amarillo.



	Column1	ID	DLC	DATA 0	DATA 4	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7	DECIMAL DATA 5
5766	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0x00	0xFF	0xFF	
5897	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0x00	0xFF	0xFF	
5945	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0x00	0xFF	0xFF	
6047	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0x03	0xFF	0xFF	
6082	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0x04	0xFF	0xFF	
6102	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0x04	0xFF	0xFF	
6216	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0x11	0xFF	0xFF	
6443	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0xAF	0xFF	0xFF	
6465	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0xBC	0xFF	0xFF	
6519	RX	0x700	8	0x05	0x62	0x01	0x18	0x01	0x5B	0xFF	0xFF	
6563	RX	0x700	8	0x05	0x62	0x01	0x18	0x01	0x6F	0xFF	0xFF	111
6728	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0xF6	0xFF	0xFF	
6800	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0xB4	0xFF	0xFF	
6811	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0xA6	0xFF	0xFF	
6837	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0x76	0xFF	0xFF	
6974	RX	0x700	8	0x05	0x62	0x01	0x18	0xFF	0xFF	0xFF	0xFF	
6987	RX	0x700	8	0x05	0x62	0x01	0x18	0xFF	0xFC	0xFF	0xFF	
7021	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0x01	0xFF	0xFF	
7057	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0x00	0xFF	0xFF	
7082	RX	0x700	8	0x05	0x62	0x01	0x18	0xFF	0xFF	0xFF	0xFF	
7091	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0x01	0xFF	0xFF	
7266	RX	0x700	8	0x05	0x62	0x01	0x18	0x00	0x00	0xFF	0xFF	

Figura 2.22 DATA LOG del YAW RATE SENSOR

La Figura 2.23 muestra la gráfica del flujo de datos del YAW RATE SENSOR, igual por medio de esta gráfica se pretende comprobar los resultados obtenidos tras el proceso de decodificación en función de los picos de giro que se sometió al vehículo. Sabiendo que un giro a la derecha tendremos picos positivos y a su vez un giro a la izquierda pico negativos, con esta información podemos determinar que el vehículo realizo un giro a la derecha obteniendo un pico positivo de 38 deg/s. Con este dato se puede corroborar los resultados obtenidos al aplicar la fórmula desarrollada para el sensor de giro o guiñada.

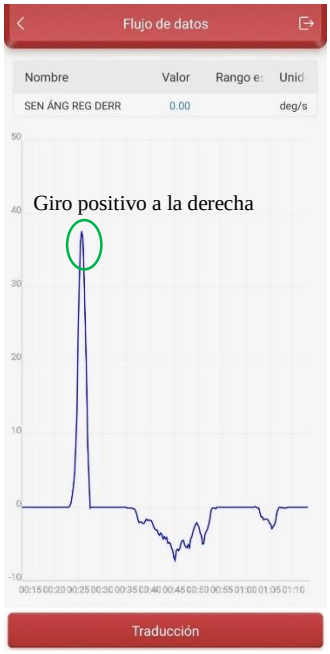


Figura 2.23 Señal del flujo de datos del YAW RATE SENSOR obtenida del escáner automotriz

2.3.8.4 DECEL G SENSOR

La Figura 2.24 muestra el flujo de datos de la aceleración y desaceleración longitudinal del vehículo, siendo DATA: 4 y 5 la respuesta de la trama de datos, no obstante, se consideran los picos de aceleración y desaceleración para la decodificación de los datos marcados.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
20055	1531,8062	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x06	0xFF	0xFF				
20099	1532,0002	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x06	0xFF	0xFF				
20137	1532,1752	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x06	0xFF	0xFF				
20199	1532,4042	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x05	0xFF	0xFF				
20352	1533,0122	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x04	0xFF	0xFF				
20402	1533,0912	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x05	0xFF	0xFF				
20432	1533,2742	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x04	0xFF	0xFF				
20551	1533,6912	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x05	0xFF	0xFF				
20704	1534,2872	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x1C	0xFF	0xFF				
20859	1534,8842	RX	0x700	8	0x05	0x62	0x01	0x15	0x01	0x70	0xFF	0xFF		Aceleración	112	
20887	1535,0582	RX	0x700	8	0x05	0x62	0x01	0x15	0x01	0xB7	0xFF	0xFF			183	
21059	1535,6812	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0xF8	0xFF	0xFF				
21247	1536,2983	RX	0x700	8	0x05	0x62	0x01	0x15	0xFE	0xEC	0xFF	0xFF		Desaceleración	236	
21357	1536,7182	RX	0x700	8	0x05	0x62	0x01	0x15	0xFE	0xB7	0xFF	0xFF			183	
21659	1537,9132	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x36	0xFF	0xFF				
21704	1538,1122	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x08	0xFF	0xFF				
21790	1538,3412	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x12	0xFF	0xFF				
22002	1539,2843	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x0C	0xFF	0xFF				
22237	1540,3982	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x0E	0xFF	0xFF				
22425	1541,5772	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x0F	0xFF	0xFF				
22530	1541,8412	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x10	0xFF	0xFF				
22750	1542,7942	RX	0x700	8	0x05	0x62	0x01	0x15	0x00	0x08	0xFF	0xFF				

GUIÑADA | ACELERACIONES | V.RUEDAS | GRAFICAS

Listo

Figura 2.24 DATA LOG del DECEL G SENSOR

Para confirmar si la fórmula desarrollada para el DECEL G SENSOR es la correcta se hace uso de la gráfica mostrada en la Figura 2.25, donde se visualizan los datos de la aceleración y desaceleración del vehículo, sabiendo que el primer pico positivo de 0.47 g representa a la aceleración y a su vez el pico negativo de -0.35 g es la deceleración que sufrió el vehículo. En función de estos dos picos se puede corroborar los resultados obtenidos tras el proceso de decodificación.

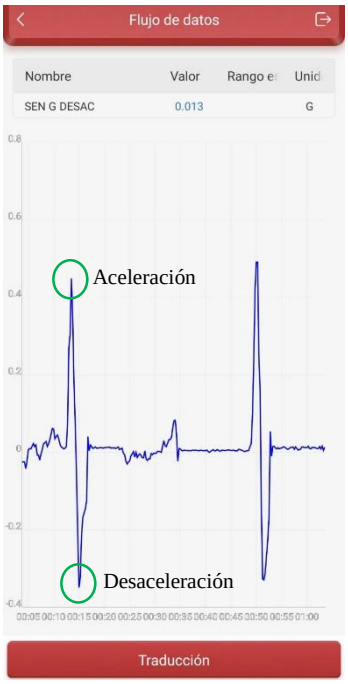


Figura 2.25 Señal del flujo de datos del DECEL G SENSOR

2.3.8.5 SIDE G SENSOR

Para el proceso de decodificación del SIDE G SENSOR (Aceleración transversal) no fue necesario realizar un DATA LOG debido que su decodificación es igual a DECEL G SENSOR. La fórmula se presentará en el apartado de resultados.

2.3.9 Análisis del umbral de valores de las aceleraciones transversal y longitudinal

Los sensores de accidente y de aceleración se encuentran directamente en la unidad de control o a su vez pueden ubicarse como si fueran satélites en el frontal o lateral del vehículo, en esta unidad de control se almacena información la cual ha sido obtenida tras realizarse varias pruebas de simulación de accidentes (HELLA, s/f). Mediante los diversos tests se puede clasificar un accidente por su grado de gravedad:

Grado	Descripción
Gravedad 0	Accidente leve
Gravedad 1	Accidente de gravedad media
Gravedad 2	Accidente grave
Gravedad 3	Accidente muy grave

Tabla 2.16 Clasificación del grado de gravedad de un accidente vehicular

(HELLA, s/f)

Según el (INEN, 2013) establece la Normativa NTE INEN 2675:2013 acerca de los requisitos de inspección de los cinturones de seguridad, menciona el funcionamiento de los retractores con seguro de emergencia y a su vez la condiciones de operación que estos deben cumplir:

Condiciones de Funcionamiento de Retractores con Seguro de Emergencia			
Desaceleración		**Aceleración**	
0,45 g	Retractor clase 4	< 0,8 g	Retractor clase 4
0,85 g	Retractor clase 4N	< 1,0 g	Retractor clase 4N

Tabla 2.17 Condiciones de funcionamiento de los retractores

(INEN, 2013)

La Tabla 2.17 muestra las condiciones de operación que debe cumplir un retractor al permitir que el cinturón de seguridad se extienda y retraiga libremente con el movimiento de los ocupantes, de tal manera el retractor debe quedar asegurado cuando la desaceleración del vehículo alcanza 0,45 g cuando se trata de retractores de la clase 4, mientras que 0,85 g cuando se trata de retractores de clase 4 N. Y no debe quedar asegurado con valores de aceleración medidos en dirección de extracción de la cinta, menos de 0,8 g cuando se aplique retractores de clase 4 y menos de 1 g con retractores de clase 4N.

Se considera la información de la Tabla 2.16 y 2.17 para el desarrollo de la programación del microcontrolador en Arduino sobre el umbral de valores de las aceleraciones trasversales y longitudinales, tomando en cuenta el valor referencial de 0,45 g para el funcionamiento del retractor en desaceleración, valor establecido por la normativa NTE INEN 2675:2013.

Con la finalidad de realizar la adquisición, registro y almacenamiento de los datos, siempre y cuando se supere las 0.60 g, valor que se consideró para el funcionamiento del dispositivo.

De tal manera si el vehículo supera las 0.60 g (gravedad leve), inmediatamente se ejecutará el almacenamiento y registro de datos, de las variables que se consideraron para este estudio y a su vez siendo almacenadas en una tarjeta SD para su posterior lectura.

2.3.10 Elaboración del diagrama de flujo sobre la programación del microcontrolador

La Figura 2.26 muestra un diagrama de flujo del proceso a llevar a cabo para el desarrollo de la programación del microcontrolador.

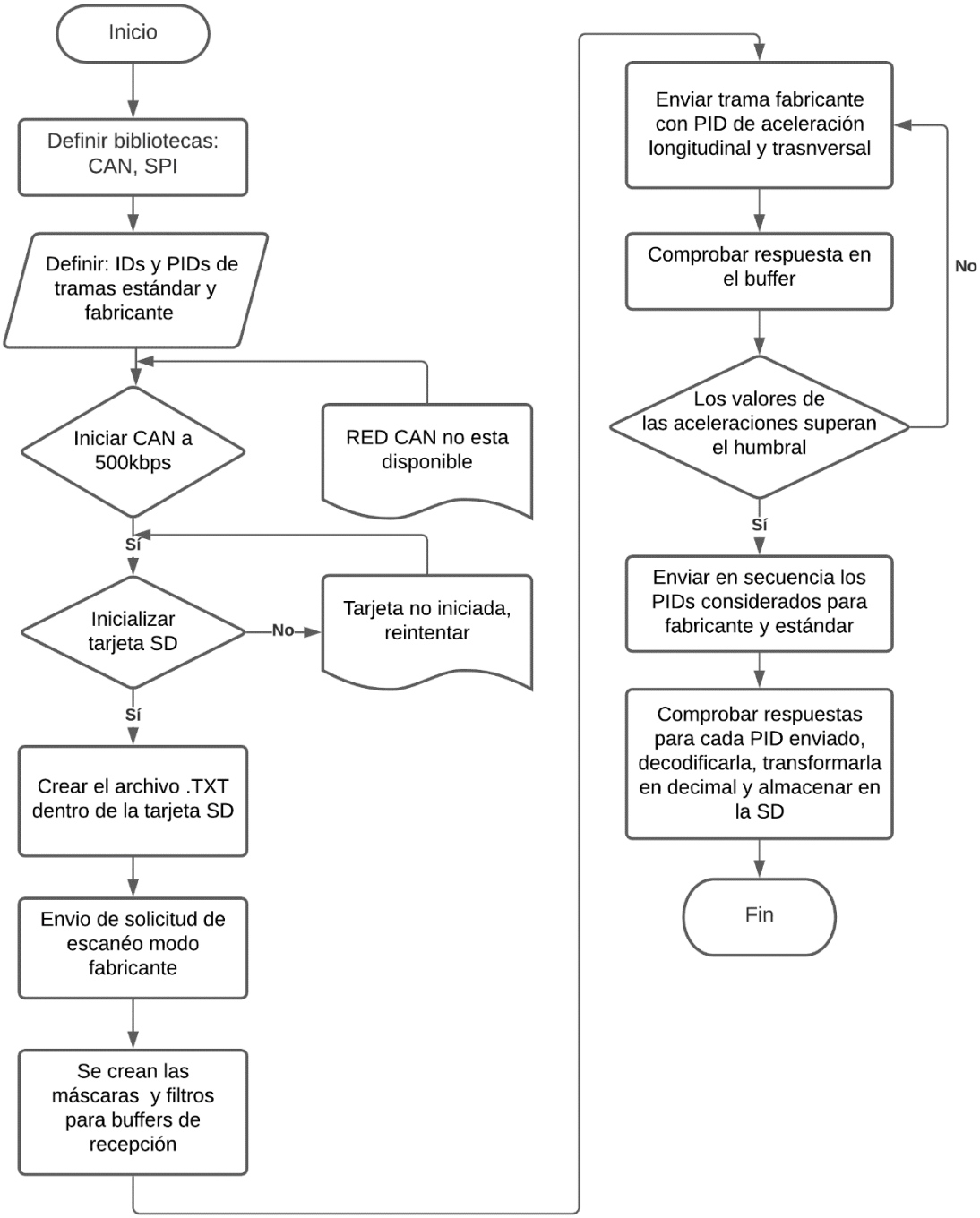


Figura 2.26 Diagrama de flujo de la programación del microcontrolador

2.3.11 Programación de las instrucciones del microcontrolador

El desarrollo de la programación se lo realizo en el software Arduino IDE para la escritura y lectura de los datos de la red CAN. Anterior a esto es necesario considerar las librerías que son compatibles con el dispositivo CAN Bus Shield el encargado de llevar a cabo la

comunicación del Arduino con la red CAN del vehículo, además de librerías que permitan el almacenamiento y registro de datos.

Las librerías que fueron usadas para llevar a cabo la programación del microcontrolador fueron las siguientes:

- CAN Bus Shield V2 master library
- SD library
- RTClib library
- SPI library

Las variables que son declaradas estarán entorno a los PIDs sea fabricantes o genérico, considerando las variables de estudio del módulo ENGINE y CHASIS CONTROL mostradas.

PIDs declarados			
Módulo CHASIS CONTROL (Fabricante)	**Valor Hexadecimal**	**Módulo ENGINE (Genérico)**	**Valor Hexadecimal**
BRAKE SW1	0x22	ENGINE SPEED	0x0C
ABS MALF	0x25	CAL/LD VALUE	0x04
FR/FL/RR/RL WHEEL SPEED	0x0F/0x10/0x11/ 0x12	VEHICULE SPEED	0x0D
STEERING ANG SENSOR	0x13	TRVL AFTER MIL	0x21
SIDE G SENSOR	0x16		
YAW RATE SENSOR	0x18		
DECEL G SENSOR	0x15		

Tabla 2.18 PIDs declarados fabricantes y genéricos

A continuación, se especifican los puntos clave para llevar a cabo el proceso de programación:

- Se define el inicio y la velocidad del monitor serial para posteriormente verificar la comunicación del microcontrolador con el vehículo a través de la computadora.

- Mostrar un mensaje de la inicialización de la comunicación del microcontrolador con el módulo RTC, módulo CAN y el módulo SD. En el caso de existir comunicación continuar.
- En el caso de fallar la inicialización de la comunicación entre los dispositivos añadir un mensaje de fallo de comunicación.
- Definir las máscaras y filtros de los buffers de recepción o aceptación de tramas.
- Envío de solicitud de escaneo modo fabricante.
- Se solicita de manera infinita los valores de las aceleraciones longitudinal y transversal.
- Comprobación de las respuestas de las solicitudes enviadas.
- Comparación de las respuestas por medio de un bucle “if” con los valores de disparo o umbrales máximos establecidos.
- Si la condición del bucle es verdadera iniciar la grabación de datos con el registro de fecha y hora, mediante funciones para envió y recepción de tramas, donde se enviará en secuencia los PIDs considerados para fabricante y estándar.
- Comprobar respuestas para cada PID enviado, decodificarla transformarla en decimal y almacenarla en la tarjeta SD.

2.3.12 Prueba de comunicación con el vehículo y de PIDs lectura.

Finalizada la programación del microcontrolador se llevó a cabo la prueba de comunicación, del dispositivo desarrollado con la red CAN del vehículo, donde se verificó su correcto funcionamiento mediante el Monitor Serial. En la Figura 2.27 se muestra el Monitor Serial de Arduino donde se evidencia la prueba de comunicación entre los dispositivos y la red CAN del vehículo fue exitosa.

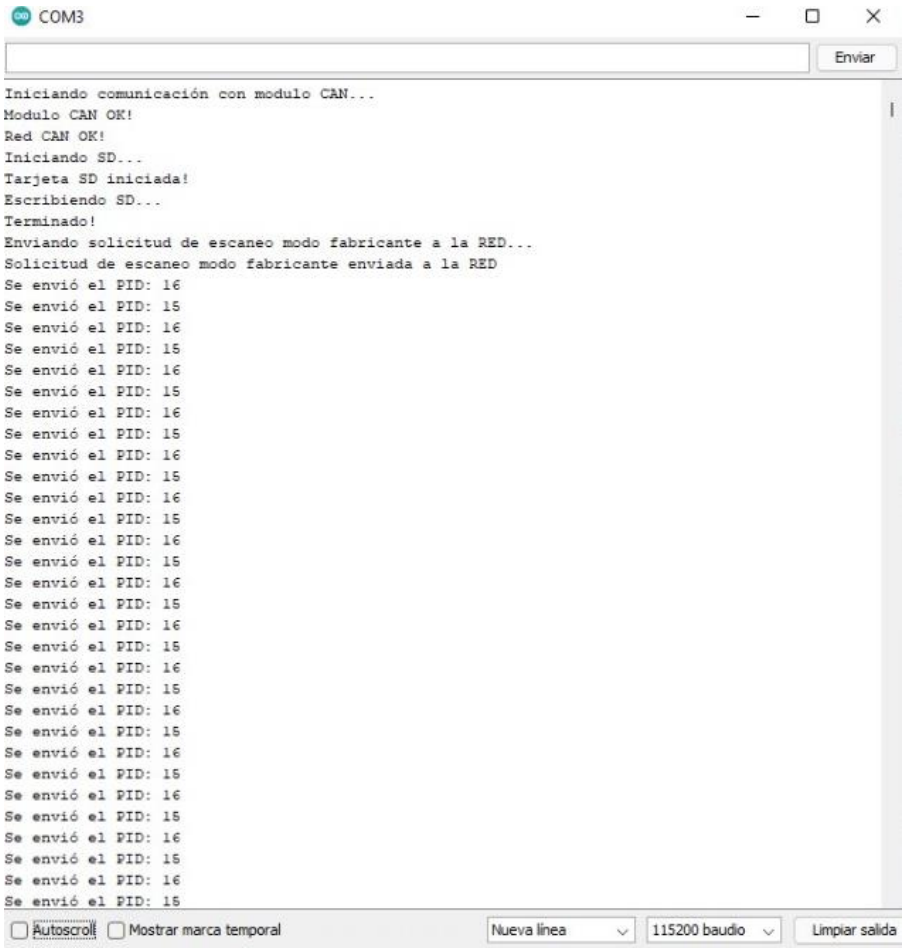


Figura 2.27 Comunicación entre el microcontrolador Arduino y la red CAN del vehículo

2.3.13 Análisis del método de recopilación de datos

Para la recopilación de datos fue necesario hacer uso de un Módulo Micro SD, el cual permite el almacenamiento de datos mediante un archivo.txt, que se generó previo a instrucciones en la tarjeta microSD para la posterior lectura de los datos. La Figura 2.28 muestra el archivo generado por las variables almacenadas.

```
------------------------------------------------------------------------------
RED CAN OK!
Nuevo registro de datos de evento iniciado:
Solicitud de escaneo modo fabricante enviada a la RED
14/2/2022 16:52:5

----------------------------------------------------------------------
ID: 0x700

16,     0,      17

0,      23

Aceleración transversal:
Izquierda 0.0230 g

----------------------------------------------------------------------

----------------------------------------------------------------------
ID: 0x700

22,     40,     C0

64,     192

Switch de freno ON

----------------------------------------------------------------------

----------------------------------------------------------------------
ID: 0x700

25,     20,     E0

32,     224
```

Figura 2.28 Registro y almacenamiento de datos en formato .txt.

2.3.14 Desarrollo del prototipo encargado de la recopilación de datos

El desarrollo del prototipo encargado de la recopilación de datos a través de la red CAN se lo llevó a cabo con los módulos que enlistan a continuación:

- Arduino Mega 2560
- CAN – Bus Shield V2
- Conector Socket OBD2
- Módulo RTC DS1302
- Módulo Micro SD

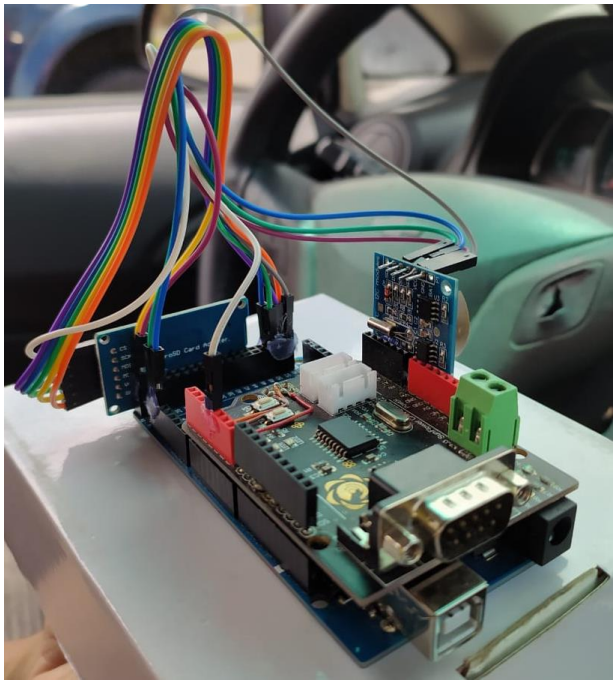


Figura 2.29 Microcontrolador encargado de la adquisición y almacenamiento de datos

La Figura 2.29 muestra el concepto final del prototipo para la adquisición de datos de la red. A través del conector DLC el dispositivo se comunica con el vehículo mediante el microcontrolador previamente programado, utilizando de por medio el microcontrolador CAN (CAN Bus Shield) el cual le permitirá comunicarse a través de la red. Gracias a esta conexión se logró extraer los datos necesarios en función de las variables que se seleccionaron previamente. Mediante el Módulo Micro SD se permite el almacenamiento y registro de datos en un archivo .txt, el cual facilita la interpretación de los datos grabados, mientras que el Módulo RTC se registrara la hora y fecha del evento. Cabe recalcar que este proceso se ejecutará siempre y cuando el vehículo supere las 0.60 gravedades de las aceleraciones trasversal y longitudinal. Finalmente, mediante cables macho-hembra se permite las conexiones entre todos los módulos, respetando sus pines de comunicación.

2.3.15 Pruebas acerca del funcionamiento del sistema

Una vez desarrollado el prototipo se llevaron a cabo las pruebas de funcionamiento del sistema, para esto se consideró un lugar apropiado donde no exista tráfico vehicular, debido a que se somete al vehículo a maniobras bruscas controladas, utilizando los frenos para forzar la grabación de datos, cuando el umbral de las aceleraciones supere las 0.60 gravedades. Una

vez superado el umbral inmediatamente el microcontrolador debe solicitar los datos a la red CAN de los PIDs fabricantes y genéricos de las variables seleccionadas para su posterior registro y almacenamiento de datos en la tarjeta SD.

CAPÍTULO III

3 RESULTADOS Y DISCUSIONES

3.1 Resultados del análisis de las variables de estudio referente a los sistemas de seguridad activa y pasiva del vehículo

Una vez analizado los sensores que forman parte de cada sistema de seguridad, se concluyó que algunos sistemas comparten información de varios sensores a través de la red, es decir que dos o más sistemas de seguridad pueden depender de un solo sensor. Por esta razón es común encontrar en un flujo de datos de cada Unidad Electrónica de Control del vehículo información del mismo sensor. La ventaja de tener las computadoras conectadas en una misma red es que se puede analizar únicamente una de ellas que contenga la información de todos los sensores a analizar.

Sistemas de Seguridad Activa y Pasiva que Comparten Sensores	
Sensores	**Sistemas de seguridad dependientes**
Presión del frenado	Control Electrónico de Estabilidad y Antibloqueo del Freno
Ángulo de giro	Retención Suplementaria, Control Electrónico de Estabilidad y Dirección Electrónicamente Asistida
Aceleración transversal	Retención Suplementaria y Control Electrónico de Estabilidad
Switch del freno	Control Electrónico de Estabilidad y Antibloqueo del Freno
Velocidad de las ruedas	Control Electrónico de Estabilidad y Antibloqueo del Freno
Velocidad del vehículo	Retención Suplementaria, Control Electrónico de Estabilidad, Dirección Electrónicamente Asistida, Antibloqueo del Frenado
Ángulo y torque de dirección o volante	Control Electrónico de Estabilidad y Dirección Electrónicamente Asistida

Tabla 3.1 Sistemas de seguridad activa y pasiva que comparten sensores

3.2 Resultado del vehículo seleccionado

El vehículo que cumple con los parámetros necesarios para el desarrollo del proyecto es el Nissan Kicks, el cual esta implementado con Sistema de Diagnostico a Bordo OBDII con protocolo de comunicación CAN, como se observa en la presente imagen.

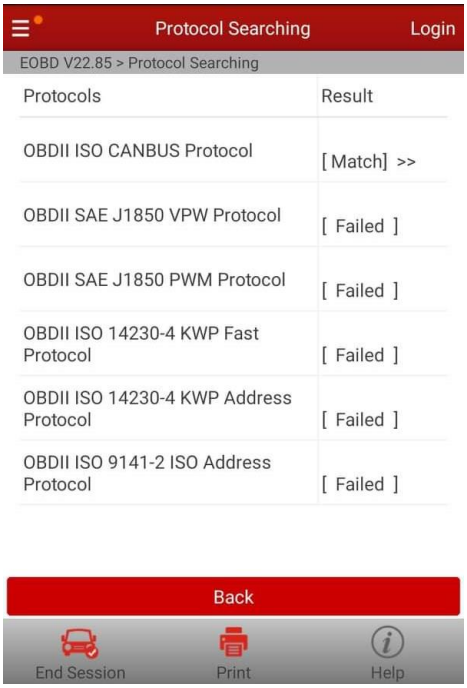


Figura 3.1 Enlace del escáner con el protocolo CAN BUS

Mediante un escaneo rápido en modo fabricante se encontraron los módulos que se presentan en la siguiente imagen

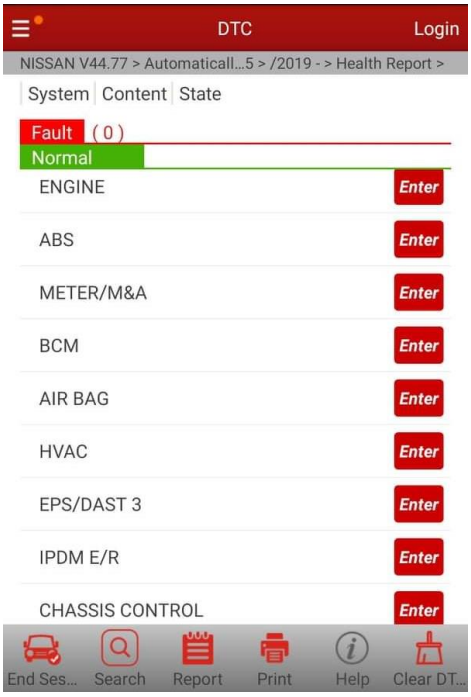


Figura 3.2 Módulos encontrados en el vehículo seleccionado

3.3 Análisis del vehículo seleccionado usando escáner

Los módulos ENGINE y CHASIS CONTROL se consideraron necesarios para la adquisición del flujo de datos.

Módulos	Descripción
ENGINE	Se encuentran datos de los dispositivos electrónicos que controlan el motor como: régimen del motor, revoluciones, velocidad del vehículo, entre otros.
CHASIS CONTROL	Se encuentran variables que envían los diferentes módulos de los sistemas de seguridad activa y pasiva correspondientes al funcionamiento general del vehículo.

Tabla 3.2 Descripción de los módulos ENGINE y CHASIS CONTROL

3.4 Análisis del flujo de datos y PIDs

Para la selección de los PIDs, se tomó en cuenta el tipo de información que estos suministran y su utilidad para el estudio; estos PIDs seleccionados se los puede observar en las Tablas 3.3 y 3.4 de los módulos: ENGINE y CHASIS CONTROL.

Módulo de Control Electrónico (ENGINE)		
Nombre	Descripción	Unidad
CAL/LD VALUE	Valor calculado de la carga	%
ENGINE SPEED	Revoluciones del motor	rpm
BRAKE SWITCH	Switch del freno	
VHCL SPEED SE	Sensor VSS	km/h
TRVL AFTER MIL	Distancia con luz MIL ON	km
VEHICLE SPEED	Sensor VSS	km/h

Tabla 3.3 Módulo de Control Electrónico (ENGINE)

En la Tabla 3.4 se muestran las variables que fueron seleccionadas para la adquisición de las tramas, se menciona que variables como: ABS y BRAKE HOLD no se consideraron debido que las tramas de requerimiento son las mismas a otras variables, siendo ABS para ABS MALF y BRAKE HOLD para BRAKE SW1.

CHASIS CONTROL		
Nombre	Descripción	Unidad
BRAKE SW1	Switch del freno 1	-
ABS MALF	Estado del sistema ABS	-
FR WHEEL SPEED	Velocidad rueda frontal derecho	Rpm
FL WHEEL SPEED	Velocidad rueda frontal izquierdo	Rpm
RR WHEEL SPEED	Velocidad rueda posterior derecho	Rpm
RL WHEEL SPEED	Velocidad rueda posterior izquierdo	rpm
STEERING ANG SENSOR	Ángulo de la dirección	Deg

SIDE G SENSOR	Sensor de fuerza transversal	G
YAW RATE SENSOR	Sensor de giro o guiñada	deg/s
DECEL G SENSOR	Sensor de aceleración y desaceleración longitudinal	G

Tabla 3.4 Módulo del Chasis Control

3.6 Identificación de los códigos CAN según los PIDs necesarios

3.6.1 Extracción de los PIDs estándar

Mediante el software hyperterminal se lleva a cabo una comunicación con el escáner donde podemos extraer el flujo de datos de las diferentes variables que fueron seleccionadas del módulo ENGINE, cabe mencionar que estos datos se los considera como estándar.

ENGINE - CAL/LD VALUE									
Solicitud									
ID	**DLC**	**DATA 0**	**DATA 1**	**DATA 2**	**DATA 3**	**DATA 4**	**DATA 5**	**DATA 6**	**DATA 7**
7DF	8	02	01	04	00	00	00	00	00
Respuesta									
ID	**DLC**	**DATA 0**	**DATA 1**	**DATA 2**	**DATA 3**	**DATA 4**	**DATA 5**	**DATA 6**	**DATA 7**
7E8	8	04	41	04	00	00	00	00	00

Tabla 3.5 Estructura de la trama solicitud y respuesta del CAL/LD VALUE

ENGINE – TRVL AFTER MIL									
Solicitud									
ID	**DLC**	**DATA 0**	**DATA 1**	**DATA 2**	**DATA 3**	**DATA 4**	**DATA 5**	**DATA 6**	**DATA 7**
7DF	8	02	01	21	00	00	00	00	00
Respuesta									
ID	**DLC**	**DATA 0**	**DATA 1**	**DATA 2**	**DATA 3**	**DATA 4**	**DATA 5**	**DATA 6**	**DATA 7**
7E8	8	04	41	21	00	00	00	00	00

Tabla 3.6 Estructura de la trama solicitud y respuesta del TRVL AFTER MIL

ENGINE – VEHCL SPEED SENSOR									
Solicitud									
ID	**DLC**	**DATA 0**	**DATA 1**	**DATA 2**	**DATA 3**	**DATA 4**	**DATA 5**	**DATA 6**	**DATA 7**
7DF	8	02	01	0D	00	00	00	00	00
Respuesta									
ID	**DLC**	**DATA 0**	**DATA 1**	**DATA 2**	**DATA 3**	**DATA 4**	**DATA 5**	**DATA 6**	**DATA 7**
7E8	8	04	41	0D	00	00	00	00	00

Tabla 3.7 Estructura de la trama solicitud y respuesta del VHCL SPEED Sensor

Al igual que las Tablas: 3.5, 3.6 y 3.7 muestra la solicitud de las revoluciones del motor, la velocidad del vehículo mediante el sensor VSS y la distancia del Luz Mil, cabe mencionar que este proceso fue el mismo para cada una de las variables seleccionadas del módulo ENGINE, considerando el PID estándar que muestra la Tabla 1.1 referente a cada variable.

3.2 Extracción de los PIDs fabricantes

La extracción de los PIDs fabricantes se la llevo a cabo mediante el CAN Analyzer, donde, mediante él envió de una solicitud se entra al módulo CHASIS CONTROL y por medio de una trama independiente a cada variable se ingresaba al flujo de datos de esta.

La Tabla 3.8 presenta la solicitud y la respuesta del estado del sistema ABS “ABS MALF”, donde las tramas mantienen su estructura, pero difieren algunos DATAS como es: el DATA 3 tiene un PID “25” de la solicitud mientras que el DATA 4 tiene un PID “0” y el DATA 5 tiene un PID “E0” referente a la respuesta. Cabe mencionar que esta distribución es igual para todas las variables, como se muestra en las siguientes tablas:

CHASIS CONTROL									
Nombre				**Descripción**				**Unidad**	
ABS MALF				Estado sistema ABS				-	
				Solicitud ABS MALF					
ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
70C	8	3	22	1	25	FF	FF	FF	FF
				Respuesta					
700	8	5	62	1	25	0	E0	FF	FF

Tabla 3.8 Estructura de la Trama de solicitud y respuesta que contiene el PID del ABS MALF

La Tabla 3.9 muestra la desaceleración y aceleración longitudinal del vehículo “DELCE G SENSOR”, donde el ID 70C representa la solicitud de la trama con el DATA 3 que indica el valor de las condiciones de operación del vehículo en un valor hexadecimal siendo el PID 15 encargado de indicar estas condiciones de operación. Mientras que el ID 700 representa la trama de la respuesta que presenta valores en el DATA: 4 y 5 de “0” debido que el vehículo se mantuvo inmóvil.

CHASIS CONTROL									
Nombre				**Descripción**				**Unidad**	
DELCE G SENSOR				Desaceleración y aceleración longitudinal				-	
				Solicitud desaceleración y aceleración longitudinal					
ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
70C	8	3	22	1	15	FF	FF	FF	FF
				Respuesta					
700	8	5	62	1	15	0	0	FF	FF

Tabla 3.9 Estructura de la Trama de solicitud y respuesta que contiene el PID del SIDE G SENSOR

La Tabla 3.10 expone la aceleración transversal del vehículo “SIDE G SENSOR”, donde: el DATA 3 tiene un PID “16” de solicitud, el DATA 4 tiene un PID “FF” y el DATA 5 tiene un PID “FF” de respuesta. Además, los datos de la trama de respuesta no varia debido que el vehículo se mantuvo en cero revoluciones.

CHASIS CONTROL									
Nombre				**Descripción**				**Unidad**	
SIDE G SENSOR				Aceleración Transversal				-	
Solicitud aceleración transversal									
ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
70C	8	3	22	1	16	FF	FF	FF	FF
Respuesta									
700	8	5	62	1	15	FF	FF	FF	FF

Tabla 3.10 Estructura de la Trama de solicitud y respuesta que contiene el PID de la Aceleración Transversal

Las Tablas: 3.11, 3.12, 3.13 y 3.14 muestran las tramas de las velocidades de las ruedas frontal posterior izquierdo y derecho, respectivamente. Estas mantendrán la misma estructura y solo cambiará los DATA: 3 del PID de solicitud, 4 y 5 de los PIDs de respuesta.

CHASIS CONTROL									
Nombre				**Descripción**				**Unidad**	
FR WHEEL SPEED				Velocidad rueda frontal derecho				rpm	
Solicitud velocidad rueda frontal derecho									
ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
70C	8	3	22	1	0F	FF	FF	FF	FF
Respuesta									
700	8	5	62	1	0F	0	0	FF	FF

Tabla 3.11 Estructura de la Trama de solicitud y respuesta que contiene el PID del FR WHEEL SPEED

CHASIS CONTROL									
Nombre				**Descripción**				**Unidad**	
FL WHEEL SPEED				Velocidad rueda frontal izquierdo				rpm	
Solicitud velocidad rueda frontal izquierdo									
ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
70C	8	3	22	1	10	FF	FF	FF	FF
Respuesta									
700	8	5	62	1	10	0	0	FF	FF

Tabla 3.12 Estructura de la Trama de solicitud y respuesta que contiene el PID del FL WHEEL SPEED

CHASIS CONTROL									
Nombre				**Descripción**				**Unidad**	
RR WHEEL SPEED				Velocidad rueda posterior derecho				rpm	
Solicitud velocidad rueda posterior derecho									
ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
70C	8	3	22	1	11	FF	FF	FF	FF
Respuesta									
700	8	5	62	1	11	0	0	FF	FF

Tabla 3.13 Estructura de la Trama de solicitud y respuesta que contiene el PID del RRWHEEL SPEED

CHASIS CONTROL									
Nombre				**Descripción**				**Unidad**	
RL WHEEL SPEED				Velocidad rueda posterior izquierdo				rpm	
Solicitud velocidad rueda posterior izquierdo									
ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
70C	8	3	22	1	12	FF	FF	FF	FF
Respuesta									
700	8	5	62	1	12	0	0	FF	FF

Tabla 3.14 Estructura de la Trama de solicitud y respuesta que contiene el PID del RR WHEEL SPEED

La Tabla 3.15 muestra la trama del STERING ANG SENSOR “Ángulo de dirección”, donde el DATA 3 es la solicitud, mientras que los DATA 4 y 5 son las respuestas. Esta estructura se mantiene para la Tabla 3.16 solo que cambian los PIDs de cada DATA correspondiente a cada ID de la trama.

CHASIS CONTROL									
Nombre				**Descripción**				**Unidad**	
STERING ANG SENSOR				Angulo de dirección				Deg	
Solicitud velocidad rueda posterior izquierdo									
ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
70C	8	3	22	1	13	FF	FF	FF	FF
Respuesta									
700	8	5	62	1	13	0	0	FF	FF

Tabla 3.15 Estructura de la Trama de solicitud y respuesta que contiene el PID del STERING ANG SENSOR

CHASIS CONTROL									
Nombre				**Descripción**				**Unidad**	
YAW RATE SENSOR				Sensor de giro o guiñada				deg\s	
				Solicitud ángulo de dirección					
ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7
70C	8	3	22	1	18	FF	FF	FF	FF
				Respuesta					
700	8	5	62	1	18	0	0	FF	FF

Tabla 3.16 Estructura de la Trama de solicitud y respuesta que contiene el PID del YAW RATE SENSOR

3.7 Formulas obtenidas tras el proceso de decodificación de los PIDs fabricante considerando las variables de estudio

Realizado el proceso de decodificación a continuación, se muestra las fórmulas correspondientes a cada una de las variables seleccionadas:

3.7.1 WHEEL SPEED

El proceso de decodificación del WHEEL SPEED (Velocidad de las ruedas) se lo llevo a cabo, convirtiendo los datos en decimal y posteriormente a binario obteniendo la siguiente formula:

$$WS = \frac{(A * 10000) + (B + 256) * 10000}{1000} \quad (1)$$

Donde:

WS: Velocidad de las ruedas

A: DATA 4

B: DATA 5

Según los datos capturados y registrados del Data Log se realiza un ejemplo de aplicación de la formula para obtener la velocidad de las ruedas en formato decimal interpretable. Donde el valor de DATA 4 es: 0x1B y el valor de DATA 5: 0x0D, transformando estos valores hexadecimales a decimales se obtiene:

DATA 4: 27

DATA 5: 13

Con la aplicación de la fórmula:

$$WS = \frac{(27 * 10000) + (13 + 256) * 10000}{1000}$$

$$WS = 270\ rpm$$

Se obtiene el valor de la velocidad de las ruedas, siendo 270 rpm.

3.7.2 STEERING ANG SENSOR

La obtención de la formula del ángulo de dirección del volante es:

$$SAS = \frac{(A * 1000 + B)}{100} \quad (2)$$

Donde:

SAS: Ángulo de dirección del volante

A: DATA 4

B: DATA 5

3.7.3 SIDE G SENSOR

El proceso de decodificación de la aceleración transversal se la llevo a cabo considerando los valores del DATA: 4 y 5. En la Tabla 3.17 se muestra la función de DATA 4 como respuesta del SIDE G SENSOR, donde 0x00 es la aceleración positiva (izquierda) y 0xFF la aceleración negativa (derecha) del vehículo. Por lo tanto, las Ecuaciones 3 y 4 muestran el resultado de la decodificación siendo SGSn (Aceleración transversal negativa) y SGSp (Aceleración transversal positiva).

DATA 4 (Hexadecimal)	DATA 4 (Decimal)	Descripción del DATA 4
0x00	0	Aceleración negativa
0xFF	255	Aceleración positiva

Tabla 3.17 Función del DATA 4

(3)

$$SGSn = \frac{256 * A + B}{1000}$$

$$SGSp = \frac{(256 * (255 - A) + (255 - B))}{1000} \quad (4)$$

3.7.4 YAW RATE SENSOR

El resultado obtenido de la decodificación de los datos del sensor de giro o guiñada se muestra en la siguiente tabla:

DATA 4 (Hexadecimal)	DATA 4 (Decimal)	Descripción del DATA 4
0x00	0	Giro positivo (+) a la derecha
0xFF	255	Giro negativo (-) a la izquierda

Tabla 3.18 Función del DATA 4

La Tabla 3.18 muestra la función del DATA 4 como respuesta del YAW RATE SENSOR, siendo este el que determina la dirección del giro del vehículo, donde 0x00 en decimal (0) es el giro positivo a la derecha mientras que 0xFF en decimal (255) giro negativo a la izquierda esto se lo puede comprobar en la Figura 2.23. La Ecuación 5 y 6 muestran el resultado de la decodificación, donde YRSd es (Giro positivo) mientras que YRSi (Giro negativo).

$$YRSd = \frac{256 * A + B}{10} \quad (5)$$

$$YRSi = \frac{256 * (255 - A) + (255 - B)}{10} \quad (6)$$

3.7.5 DECEL G SENSOR

El proceso de decodificación de las aceleraciones y desaceleraciones longitudinales fue el siguiente:

DATA 4 (Hexadecimal)	DATA 4 (Decimal)	Descripción del DATA 4
0x00	0	Aceleración
0xFF	255	Desaceleración

Tabla 3.19 Función del DATA 4

La Tabla 3.19 muestra la función de DATA 4 como respuesta del DELCEL G SENSOR, donde 0x00 es la aceleración y 0xFF la desaceleración del vehículo. Las Ecuaciones 7 y 8 muestran las fórmulas siendo estas el resultado tras el proceso de decodificación, donde DGSa es la (Aceleración longitudinal) de vehículo, mientras que DGSd es la (Desaceleración longitudinal).

$$DGSa = \frac{256 * A + B}{1000} \tag{7}$$

$$DGSd = \frac{(256 * (255 - A) + (255 - B))}{1000} \tag{8}$$

3.9 Resultados de las pruebas de funcionamiento del sistema

La Figura 3.3 se muestra el resultado del funcionamiento de una de las pruebas que se realizó al sistema, donde observamos primero la comunicación del dispositivo con la red CAN del vehículo y posterior el registro de la fecha y hora del evento. Por otro lado, se muestra el valor hexadecimal de 0x700 de la trama recibida correspondiente al ID, en las siguientes dos filas se visualiza los DATA 3, 4 y 5, donde se muestra el parámetro de identificación enviado y los valores correspondientes a la respuesta respectivamente. Donde podemos analizar variables como aceleración trasversal, switch del freno y el sistema ABS. En función de estas variables se pudo determinar que el vehículo tuvo una aceleración trasversal (izquierda) un valor de 0,0240 g siendo una fuerza lateral mínima, valores considerados prescindibles, los cuales no entran en el rango donde puede suscitarse un accidente, a su vez se muestra que el

switch del freno se encontraba presionado (ON) y el sistema ABS de igual manera se encontraba activado evitando que los neumáticos tengan perdida de adherenca sobre la calzada tras el proceso de frenado. Además, se muestra las otras variables que corresponde aceleración longitudinal con un valor de -0.7480 g siendo una fuerza longitudinal mínima, considerando que este es el rango del umbral para que se active al registro y almacenamiento de datos de parte del dispositivo, mientras que la velocidad de la rueda delantera derecha presenta un valor de 234.10 rpm.

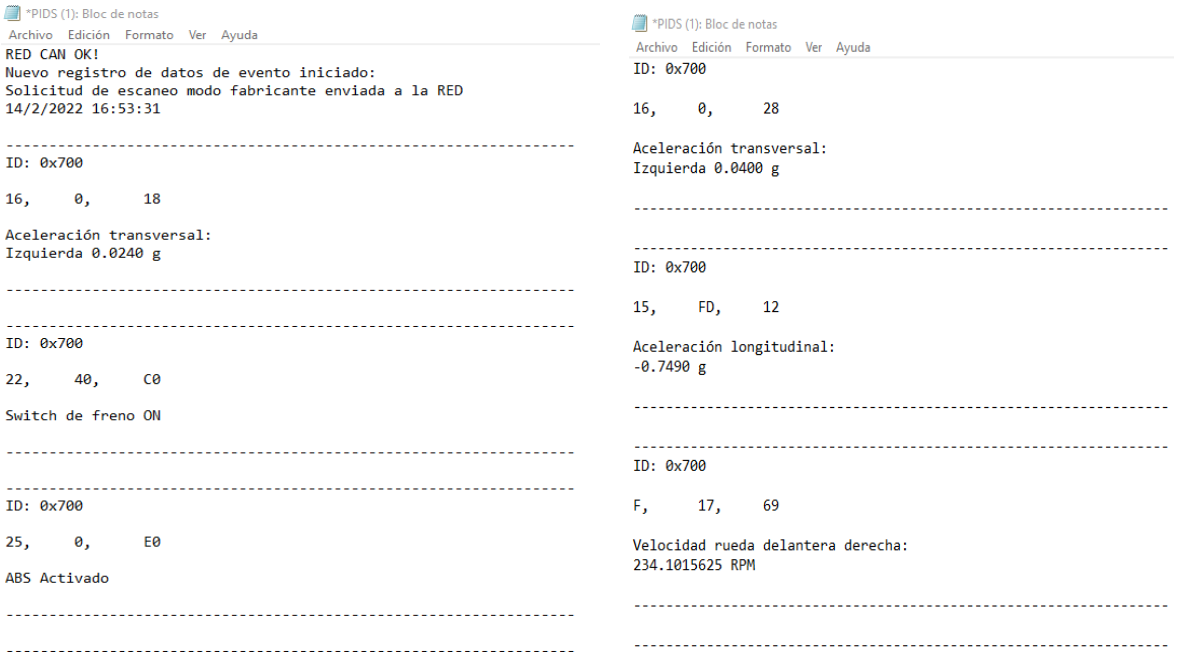


Figura 3.3 Registro de datos en formato .txt

Además, la Figura 3.4 muestra los otros valores referentes a las velocidades de las ruedas delantera izquierda y posterior derecha e izquierda, en función de estas velocidades y de los datos obtenidos del ángulo de dirección y giro o guiñada se puede considerar si el vehículo sufrió de subviraje o sobreviraje. Sabiendo que el subviraje es el derrape de las ruedas delanteras cuando tomamos una curva, mientras que el sobreviraje se lo considera como el derrape de las ruedas traseras. Observamos que el volante en función del dato obtenido del ángulo de dirección tuvo un leve giro a la 0.89000° y el vehículo tuvo un giro sobre su eje vertical de 0.3000° dato obtenido de la variable de ángulo de giro o guiñada, son valores que permiten analizar el comportamiento dinámico lineal o no lineal del vehículo, a su vez este análisis dinámico se completa con la salida de información presentada en las velocidades de las ruedas.

Si el comportamiento del vehículo es no lineal, es decir no responde a los datos enviados por los sensores de dirección y giro o guiñada, estas son posibles causas para efectuar un accidente vehicular.

Figura 3.4 Registro de datos en formato .txt

Mientras que la Figura 3.5 muestra los últimos datos de las variables: la carga del motor y la distancia con LUZ MIL. Los valores que se visualizan son netamente informativos para saber el estado del vehículo, si este al momento de suscitarse el accidente presentaba fallas en el sistema electrónico. En la prueba de funcionamiento obtuvimos un valor de 0 de la LUZ MIL debido que el vehículo no presentaba fallas con el sistema electrónico.

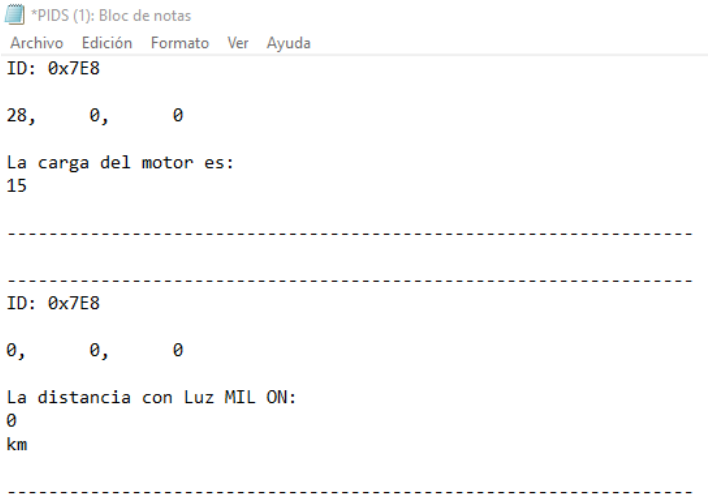


Figura 3.5 Registro de datos en formato .txt

CAPÍTULO IV

4 CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

- Se determinó que los sistemas de seguridad activa y pasiva del vehículo comparten información de uno o más sensores para su funcionamiento, es decir todos los sistemas mantienen al menos una variable relacionada como: la velocidad en cada una de las ruedas y la velocidad del vehículo, para el caso del Control Electrónico de Estabilidad y el Sistema ABS. Por consiguiente, las velocidades de las ruedas y la velocidad del vehículo son factores clave para predecir una parte del comportamiento dinámico del vehículo.
- Los módulos ENGINE y CHASIS CONTROL son los que presenta las variables que se consideraron importantes para predecir el comportamiento dinámico del vehículo en el caso de suscitarse un accidente, debido que ofrecen información relevante acerca de los sistemas de seguridad activa y pasiva del vehículo. Estas variables son: valor calculado de la carga, revoluciones del motor, switch del freno, sensor VSS, distancia de luz mil, estado del sistema ABS, velocidades de las ruedas, ángulo de dirección, aceleraciones longitudinales y transversales y sensores de giro o guiñada.
- Una red CAN puede ser hackeada a través de un analizador CAN y un escáner automotriz, puesto que estas herramientas permiten extraer un flujo de datos hexadecimales, siendo estas las tramas de los PIDs sean estándar o fabricantes. Estos datos pueden ser filtrados con la finalidad de ser resumidos, para obtener las solicitudes de ingreso a cada una de las variables de interés y a su vez adquirir la respuesta de estas variables en función de una trama constituida por bytes de información.
- La decodificación de los PIDs es posible siempre y cuando se haga uso de un escáner automotriz y un analizador CAN, ya que estas herramientas permiten obtener un flujo de datos y graficas en tiempo real. Identificando los valores picos de esos datos y logrando un análisis de cada uno de ellos, donde se facilita desarrollar una fórmula para obtener el dato en formato decimal, a través de los PIDs hexadecimales recibidos.
- La comunicación del microcontrolador y la red CAN es posible gracias al desarrollo de una interfaz constituida por funciones, las cuales permitan la extracción,

decodificación y almacenamiento de los PIDs. Además del uso de librerías que sean compatibles con el dispositivo CAN Bus Shield. Permitiendo el almacenamiento de los datos en el caso de suscitarse una posible colisión.

- Superando el rango de valores de las aceleraciones transversales y longitudinales específicamente las 0,60 gravedades. Es posible la lectura del flujo de datos de la red CAN, por medio del microcontrolador y posteriormente la escritura y almacenamiento de los datos en la tarjeta SD.

4.2 RECOMENDACIONES

- Para futuros estudios se recomienda la implementación de gráficas del flujo de datos de las variables que se consideran importantes para conocer las condiciones que suscitan el accidente vehicular, de la manera mediante estas graficas que presentan los datos de los sistemas de seguridad activa y pasiva permitan que la información sea más interpretable y se llegue con el tiempo a obtener directamente de la lectura y escritura de los datos de la red CAN, un informe pericial. El cual muestre toda la información para concluir inmediatamente las causas del accidente.
- Se aconseja que el almacenamiento del flujo de datos adquiridos de la red CAN en el caso de suscitarse un accidente vehicular, sean estos enviados y guardados en la nube, con la finalidad que esta información sea accesible para el usuario sin la necesidad de disponer del vehículo en ese momento. Este sistema se llevaría a cabo siempre y cuando el vehículo tenga accesibilidad a una red de internet, de tal modo los datos estarían almacenados en la tarjeta SD, hasta que el vehículo disponga de una red y pueda subir la información.
- En el futuro, la implementación de cámaras de seguridad y micrófonos en el vehículo, podrían añadirse al grupo de variables que se consideran para la adquisición de los datos en la red CAN, de tal manera se podrían extraer la información de la cámara de seguridad y estos videos ser almacenadas en la nube junto a los datos almacenados en la tarjeta SD, con el objetivo de mejorar la información que se puede obtener del vehículo en el caso de suscitar un siniestro.
- Para futuros trabajos de investigación se recomienda realizar investigaciones para el hackeo de otros protocolos de comunicación, debido que no todos los vehículos trabajan con CAN ISO 15765-4, siendo este el protocolo que se utilizó para este

estudio. De tal manera mucho de los vehículos usan protocolos como: SAE J1939 y SAE J1850 y actualmente no se tiene información acerca del hackeo de la red de estos protocolos. Llevando a cabo esta posible investigación, se podría llegar al objetivo, que la mayor parte de los vehículos dispongan de un microcontrolador encargado de adquirir los datos de la red en caso de un posible accidente vehicular.

5 REVISIÓN BIBLIOGÁFRICAS

A.G.P.S. (s/f). *OBD2 – AGPS*. Recuperado el 5 de octubre de 2021, de https://www.agps.cl/obd2/

Andrews, D. F., & Limpert, R. (2013). *ELECTRONIC CONTROL MODULE DATA IN LARGE TRUCK COLLISON ANALYSIS*.

Arduino. (s/f). *Arduino - ArduinoBoardMega2560*. Recuperado el 27 de enero de 2022, de https://www.arduino.cc/en/Main/ArduinoBoardMega2560

Automotriz, I. y M. (2020). *¿Qué es el conector DLC OBD II y cuál es su función? - INGENIERÍA Y MECÁNICA AUTOMOTRIZ*. https://www.ingenieriaymecanicaautomotriz.com/que-es-el-conector-dlc-obd-ii-y-cual-es-su-funcion/

Black boxes/ in-vehicle data recorders | Mobility and transport. (s/f). Recuperado el 28 de julio de 2021, de https://ec.europa.eu/transport/road_safety/specialist/knowledge/esave/esafety_measures_known_safety_effects/black_boxes_in_vehicle_data_recorders_en

Blasco, V. (s/f). *SISTEMA DE DIAGNOSTICO OBD II ¿Qué es el OBD?*

Cantos Calderón, M. E., Ulco Ulco, S. P., Cantos Calderón, M. E., & Ulco Ulco, S. P. (2016). *Estudio cuantitativo de los sistemas de comunicación en vehículos de la región andina mediante norma ISO 9141*. https://repositorio.uide.edu.ec/handle/37000/1728

Carpio Guartambel, C. P. (2013). *Manual de procedimientos para interactuar entre protocolos de comunicación automotriz*. http://dspace.uazuay.edu.ec/handle/datos/2210

César, J., & López, O. (2014). *DISEÑO DE ESCÁNER AUTOMOTRIZ OBDII MULTIPROTOCOLO*.

Cisneros Oscar. (2010, junio). *Los Sistemas de Detección de Peatones* . http://www.centro-zaragoza.com:8080/web/sala_prensa/revista_tecnica/hemeroteca/articulos/R44_A8.pdf

Control Electrónico de Frenado - ABS, ESP y Control de Tracción. (s/f). Recuperado el 8 de noviembre de 2021, de http://motorenmarcha.com/control-de-traccion-de-frenado/

Corrigan, S. (2002). *Introduction to the Controller Area Network (CAN)*. https://www.rpi.edu/dept/ecse/mps/sloa101.pdf

Costas, A., Cerdeira-Corujo, M., Barreiro, A., Delgado, E., & Baños, A. (s/f). *Control basado en Reset para seguimiento de consigna en el sistema de Control de Crucero Adaptativo*.

David, A. (2010). *MICROCONTROLADORES PIC FUNDAMENTOS Y APLICACIONES UN ENFOQUE DIDÁCTICO*.

Dimaté Cáceres, J. M., & González Castillo, P. M. (2013). Diseño de una interfaz gráfica en Labview para el diagnostico de vehículos por medio de OBD2. *instname:Universidad Pontificia Bolivariana*. https://repository.upb.edu.co/handle/20.500.11912/911

Enríquez Herrador, R. (2009). *Guía de Usuario de Arduino*.

Estadísticas siniestros de tránsito – Agencia Nacional de Tránsito del Ecuador – ANT. (2021). https://www.ant.gob.ec/?page_id=2670

Esteban del Castillo Pérez. (2019). *Equipo de diagnostico para vehiculos OBDII*. http://deeea.urv.cat/public/PROPOSTES/pub/pdf/2606pub.pdf

EU funded projects - Veronica I+II - Squaris s.p.r.l. (s/f). Recuperado el 28 de julio de 2021, de http://www.squaris.com/about-us/eu-funded-projects/veronica-iii

Farfán, L. C., Patiño, A., Andro, M., Soto, C., Manuel, J., Farfán, C., Guerra Zapata, L., Antonio, J., Herrera, L., Huarcaya, A. N., Alejandra, M., Calderón, S., & Jhonatan, R. (s/f). *SISTEMA SENSORIAL DE FRENADO AUTOMÁTICO EN AUTOMÓVILES UNIVERSIDAD DE PIURA Electrónica General-IME SISTEMA SENSORIAL DE FRENADO AUTOMOTIZADO EN AUTOMÓVILES*.

Fezari, M., & Ali, D. (2018). *Integrated Development Environment "IDE" For Arduino*. https://www.researchgate.net/publication/328615543

Galarza Bravo, M., Flores Calero, M., Galarza Bravo, M., & Flores Calero, M. (2018). DETECCIÓN DE PEATONES EN LA NOCHE USANDO FASTER R-CNN E IMÁGENES INFRARROJAS. *Ingenius. Revista de Ciencia y Tecnología* , *20*, 48–57. https://doi.org/10.17163/INGS.N20.2018.05

Gallegos, M. (2015). *Redes multiplexadas: CAN bus de datos*.

Gámez Santana, J. L., & Sanabra Loewe, M. (2005). *Revisión y mejora de los actuales ensayos no destructivos para la evaluación del comportamiento de los sistemas de retención en caso de vuelco*. https://upcommons.upc.edu/handle/2099.1/2786

García Alvarez, R. M. (s/f). *El Sistema ADAS ayuda a prevenir accidentes de tráfico*. Recuperado el 8 de noviembre de 2021, de https://www.seguridad-vial.net/vehiculo/seguridad-pasiva/156-el-sistema-adas-ayuda-a-prevenir-accidentes-de-trafico-a-los-conductores

García, M., & Coello, M. (2017). *Propuesta de Metodología para el Peritaje en Accidentes de Tránsito para la red vial Estatal E35 correspondiente a la Provincia del Cañar.*

Gonzales, M. (2019). *Sensores en los sistemas de seguridad del automóvil*. https://core.ac.uk/download/pdf/228074031.pdf

HELLA. (s/f). *Sistema airbag: Estructura y funcionamiento | HELLA*. Recuperado el 16 de febrero de 2022, de https://www.hella.com/techworld/es/Informacion-Tecnica/Electricidad-y-electronica-del-automovil/Sistema-airbag-3083/

Ilakkiya B, & Vanitha V. (2016). *A SURVEY ON ENGINE CONTROL UNIT*. *2*. www.ijariie.com21

INEN. (2013). *NORMA TÉCNICA ECUATORIANA NTE INEN 2675:2013 CINTURONES DE SEGURIDAD. REQUISITOS E INSPECCIÓN Primera edición*.

Ingeniería y mecánica automotriz. (s/f). *¿Qué es una Central Gateway y cómo funciona?* Recuperado el 4 de octubre de 2021, de https://www.ingenieriaymecanicaautomotriz.com/que-es-una-central-gateway-y-como-funciona/amp/

Instituto Mexicano del Transporte. (s/f). *Revisión documental del protocolo CAN como*

herramienta de comunicación y aplicación en vehículos. Recuperado el 29 de septiembre de 2021, de https://imt.mx/archivos/Publicaciones/PublicacionTecnica/pt474.pdf

Johansson, K. H., Törngren, M., & Nielsen, L. (2005). Vehicle Applications of Controller Area Network. *SpringerLink*, 1–25. https://doi.org/10.1007/0-8176-4404-0_32

Jorge Sánchez Carrizo. (2017). *Simulador de una ECU y diagnóstico mediante CAN y OBD II*. https://ruidera.uclm.es/xmlui/bitstream/handle/10578/15618/TFG Jorge Sanchez Carrizo.pdf?sequence=1

Lie, A., Tingvall, C., Krafft, M., & Kullgren, A. (2006). The Effectiveness of Electronic Stability Control (ESC) in Reducing Real Life Crashes and Injuries. *https://doi.org/10.1080/15389580500346838*, *7*(1), 38–43. https://doi.org/10.1080/15389580500346838

LLamas, L. (2016). *Leer y escribir en una tarjeta SD o micro SD con Arduino*. https://www.luisllamas.es/tarjeta-micro-sd-arduino/

Luna, M. P. C. (2016). *CUESTIONARIO SOBRE REDES MULTIPLEXADAS - PDF Free Download*. https://docplayer.es/20114282-Cuestionario-sobre-redes-multiplexadas.html

Martinez, F. (2018). *"INTRODUCCIÓN A LA DIRECCIÓN ELECTRÓNICA" FABIÁN MARTINEZ-ASESOR E INSTRUCTOR AUTOMOTRIZ Lima, 03 de agosto de 2018*.

Mauricio, R., & Velandia, C. (2014). *PA133-05 SISTEMA MÓVIL DE SONDEO PREVENTIVO DE VEHÍCULOS CON SOPORTE OBDII PARA MEJORAR LA VIDA ÚTIL DEL AUTOMOTOR*. http://pegasus.javeriana.edu.co/~PA133-05-PMovVidaAutomotor/

Microchip. (1999). *Controller Area Network (CAN) Basics*. https://weble.upc.edu/asig/ea/practica4/00713a.pdf

MICROJMP. (s/f). *CAN-BUS Shield [DEV-13262] :: Micro JPM*. Recuperado el 27 de enero de 2022, de https://www.microjpm.com/products/can-bus-shield/

Mircrochip. (2019). *APGDT002 Herramienta de análisis de BUS CAN - Tecnología de microchip | Ratonero*. https://www.mouser.ec/new/microchip/microchip-apgdt002-tool/

Moriche Guerrero, M. (2008). Estudio del efecto del cinturón de seguridad y del airbag en el ocupante de un vehículo en caso de colisión. *Manuel Morrichue Guerrero*. https://e-archivo.uc3m.es/handle/10016/5535

Pacheco, J. (2011). *Monitoreo de habitos de manejo por medio de una red CAN automotriz* [Universidad de las Américas Puebla]. http://catarina.udlap.mx/u_dl_a/tales/documentos/lmt/pacheco_h_je/

Perez, A. (2017). *Derecho Ecuador - INFORMES PERICIALES EN MATERIA DE TRÁNSITO*.

proyectoarduino.com. (s/f). *El bus SPI con Arduino uso y configuración*. Recuperado el 5 de octubre de 2021, de https://proyectoarduino.com/arduino-bus-spi/

Racines, P., & López, L. (2016). *Estudio de los Procesos Técnicos de Peritajes en la*

Accidentabilidad Vehicular en el Distrito Metropolitano de Quito. Universidad Internacional del Ecuador.

Rajamani, R. (2012). Electronic Stability Control. *Springer*, 201–240. https://doi.org/10.1007/978-1-4614-1433-9_8

Rueda, A. R., Pineda, G. S., & Rojas Ramírez, S. R. (2000). *IX Reunión acional de Análisis de Esfuerzos Análisis dinámico de sensores de velocidad*.

Ruiz Campoverde, J. G. (2013). *Análisis del funcionamiento y aplicación de las redes de comunicación multiplexadas en vehículos automotrices*. http://dspace.uazuay.edu.ec/handle/datos/3276

S.A. (2021). *5 síntomas de un módulo de control del motor (ECM) defectuoso, ubicación y costo de reemplazo - OkNoticias*. https://oknoticias.website/5-sintomas-de-un-modulo-de-control-del-motor-ecm-defectuoso-ubicacion-y-costo-de-reemplazo/

Spiegel, A. (2015). *A Soft ECU Approach to Develop a Powertrain Control*. The Ohio State University.

Tecnopura. (s/f). *Módulo RTC DS1302 Reloj en tiempo real para Arduino - Tecnopura*. Recuperado el 9 de febrero de 2022, de https://www.tecnopura.com/producto/modulo-rtc-ds1302-reloj-en-tiempo-real-para-arduino/

TodoMicro. (s/f). *Scanner automotriz bluetooth ELM327 ODB2 V1.5*. Recuperado el 27 de enero de 2022, de https://www.todomicro.com.ar/instrumentos-de-medicion-y-prueba/557-scanner-automotriz-bluetooth-elm327-odb2-v15.html

Villamar Aguirre, I. D. (2008). *Estudio y Análisis de los Sistemas de Diagnóstico en los automóviles modernos, Sistemas OBD*. http://dspace.uazuay.edu.ec/handle/datos/208

What is an Automobile Black Box | ExpertLaw. (s/f). Recuperado el 28 de julio de 2021, de https://www.expertlaw.com/library/accidents/auto_black_boxes.html

Wouters, P. I. J., & Bos, J. M. J. (2000). Traffic accident reduction by monitoring driver behaviour with in-car data recorders. *Accident Analysis and Prevention*, *32*. www.elsevier.com/locate/aap

Zapatería, Ó. S. (2007). *Mecánica y Electricidad*. www.centro-zaragoza.com

6 ANEXOS

ANEXO 1

6.1 Código de programación

```
#include <SPI.h>
#include <SD.h>
#define CAN_2515
#include <Wire.h>
#include <RTClib.h>
RTC_DS1307 rtc;
#include "mcp2515_can.h"
const int SPI_CS_PIN = 10;
const int CAN_INT_PIN = 2;
mcp2515_can CAN(SPI_CS_PIN);
#define SSpin 4 //selector de esclavo al pin 22
#define CAN_ID_PID_STD        0x7DF
#define CAN_ID_PID_FABRIC     0x70C
#define PID_ENGINE_SPEED      0x0C
#define PID_CAL_LD_VALUE      0x04
#define PID_TRVL_AFTER_MIL    0x21
#define PID_VHCL_SPEED_VSS    0X0D
#define PID_BRAKESW1          0x22
#define PID_ABSMALF           0x25
#define PID_SIDE_G            0x16
#define PID_DELC_G            0x15
#define FRWS                  0X0F
#define FLWS                  0X10
#define RRWS                  0X11
#define RLWS                  0X12
#define STERING_ANGL          0X13
#define YAW_RS                0X18
```

```
unsigned char PID_ENTRADA;
unsigned char ExtraerPID = 0;
unsigned char len = 0;
unsigned char buf[8];
File archivo;
void setup() {
SERIAL_PORT_MONITOR.begin(115200);
if (!rtc.begin()){
Serial.println("Modulo RTC falló");
while(1); }
 //rtc.adjust(DateTime(__DATE__,__TIME__));  // Solo ejecutar una vez
SERIAL_PORT_MONITOR.println("Iniciando comunicación con modulo CAN...");
while(CAN_OK != CAN.begin(CAN_500KBPS)) {
SERIAL_PORT_MONITOR.println("Falló Modulo CAN");
SERIAL_PORT_MONITOR.println("Reintentando...");
delay(1000);
      }
SERIAL_PORT_MONITOR.println("Modulo CAN OK!");
 set_mask_filt(); //define máscaras y filtros de buffers de recepción
 //ENVIARTRAMAcheck(PID_ENGINE_SPEED);
 //CanChequear();
SERIAL_PORT_MONITOR.println("Red CAN OK!");
SERIAL_PORT_MONITOR.println("Iniciando SD...");
while(!SD.begin(SSpin)){
SERIAL_PORT_MONITOR.println("Falló! Tarjeta micro SD no iniciada");
SERIAL_PORT_MONITOR.println("Esperando la disponibilidad del almacenamiento...");
delay(1000);
       }
SERIAL_PORT_MONITOR.println("Tarjeta SD iniciada!");
archivo = SD.open("PIDs.txt", FILE_WRITE);
```

```
   SERIAL_PORT_MONITOR.println("Escribiendo SD...");
   if (archivo) {
   archivo.println("-------------------------------------------------------------------------");
   archivo.println("RED CAN OK!");
   archivo.println("Nuevo registro de datos de evento iniciado:");
   archivo.close();
   SERIAL_PORT_MONITOR.println("Terminado!")
   } else{
        SERIAL_PORT_MONITOR.println("Falló! No se pudo escribir tarjeta micro SD...");
        return;
        }


             // --------------------Envío de solicitud fabricante----------------------
  SERIAL_PORT_MONITOR.println("Enviando solicitud de escaneo modo fabricante a la RED..."
solicitud();
archivo = SD.open("PIDs.txt", FILE_WRITE);
archivo.println("Solicitud de escaneo modo fabricante enviada a la RED");
archivo.close();
SERIAL_PORT_MONITOR.println("Solicitud de escaneo modo fabricante enviada a la RED");
}
void loop() {
ENVIARTRAMAFab(PID_SIDE_G);
if (CAN_MSGAVAIL == CAN.checkReceive()) {
CAN.readMsgBuf(&len, buf);
int filtro =  CAN.getCanId();
if (filtro == 1792){
SERIAL_PORT_MONITOR.println("\r\n------------------------------------------------------------");
SERIAL_PORT_MONITOR.print("Obteniendo datos de ID: 0x");
SERIAL_PORT_MONITOR.println(CAN.getCanId(), HEX)
```

```
int data0104 = (buf[4]);

int data0105 = (buf[5]);

for (int i = 3; i == 5; i++) {

SERIAL_PORT_MONITOR.println("\r\n-----------------------------------------------------------
-------");

SERIAL_PORT_MONITOR.print("0x");           // QUITAR TODO PRINT SERIAL en el
codigo final, unicamente se conserva archivo.print

SERIAL_PORT_MONITOR.print(buf[i], HEX);

SERIAL_PORT_MONITOR.print("\t");

 }

SERIAL_PORT_MONITOR.println();

for (int a = 3; a == 5; a++) {

SERIAL_PORT_MONITOR.println("\r\n-----------------------------------------------------------
-------");

SERIAL_PORT_MONITOR.print("0x");

SERIAL_PORT_MONITOR.print(buf[a]);

SERIAL_PORT_MONITOR.print("\t");

SERIAL_PORT_MONITOR.println();

}

float resultadodec = 0.0;

float resultadoac = 0.0;

int CA10104=0;

int CA10105=0;

int CA101024=0;

int CA101025=0;

if(data0104>=125 && data0104<=255){

CA10104 = 255 - data0104;

CA10105 = 255 - data0105;

}else {

CA10104=0;

CA10105=0;

}
```

```
if(data0104>=0 && data0104<=10){
CA101024 = data0104;
CA101025 = data0105;
}else {
CA101024=0;
CA101025=0;
}
resultadoac = (256.0*CA101024+CA101025)/1000.0;
resultadodec = (256.0*CA10104+CA10105)/1000.0;
if(resultadodec >= 0.60 && resultadodec <= 10 ||  resultadoac >= 0.60 && resultadoac <= 10){
DateTime fecha = rtc.now();
archivo = SD.open("PIDs.txt", FILE_WRITE);
archivo.print(fecha.day());
archivo.print("/");
archivo.print(fecha.month());
archivo.print("/");
archivo.print(fecha.year());
archivo.print(" ");
archivo.print(fecha.hour());
archivo.print(":");
archivo.print(fecha.minute());
archivo.print(":");
archivo.println(fecha.second());
archivo.close();
//1
ENVIARTRAMAFab(PID_BRAKESW1);
taskCanRecv();
//2
ENVIARTRAMAFab(PID_ABSMALF);
taskCanRecv();
//3
```

```
ENVIARTRAMAFab(PID_SIDE_G );
taskCanRecv();
//4
ENVIARTRAMAFab(PID_DELC_G);
taskCanRecv();
//5
ENVIARTRAMAFab(FRWS);
taskCanRecv();
//6
ENVIARTRAMAFab(FLWS);
taskCanRecv();
//7
ENVIARTRAMAFab(RRWS);
taskCanRecv();
//8
ENVIARTRAMAFab(RLWS);
taskCanRecv();
//9
ENVIARTRAMAFab(STERING_ANGL);
taskCanRecv();
//10
ENVIARTRAMAFab(YAW_RS);
taskCanRecv();
//1
ENVIARTRAMAStd(PID_ENGINE_SPEED);
taskCanRecv2();
//2
ENVIARTRAMAStd(PID_CAL_LD_VALUE);
taskCanRecv2();
//3
ENVIARTRAMAStd(PID_TRVL_AFTER_MIL);
```

```
taskCanRecv2();
//4
ENVIARTRAMAStd(PID_VHCL_SPEED_VSS);
taskCanRecv2();
}
}
}
ENVIARTRAMAFab(PID_DELC_G);
if (CAN_MSGAVAIL == CAN.checkReceive()) {
CAN.readMsgBuf(&len, buf);
int filtro2 = CAN.getCanId();
if(filtro2 == 1792){
// Para comprobar:
SERIAL_PORT_MONITOR.println("\r\n----------------------------------------------------------------");
SERIAL_PORT_MONITOR.print("Obteniendo datos de ID: 0x");
SERIAL_PORT_MONITOR.println(CAN.getCanId(), HEX);
int data0204 = (buf[4]);
int data0205 = (buf[5]);
int i;
for (i == 3; i == 5; i++) {
SERIAL_PORT_MONITOR.println("\r\n----------------------------------------------------------------");
SERIAL_PORT_MONITOR.print("0x");          // QUITAR TODO PRINT SERIAL en el codigo final, unicamente se conserva archivo.print
SERIAL_PORT_MONITOR.print(buf[i], HEX);
SERIAL_PORT_MONITOR.print("\t");
}
SERIAL_PORT_MONITOR.println();
for (int a = 3; a == 5; a++) {
SERIAL_PORT_MONITOR.println("\r\n----------------------------------------------------------------");
```

```
SERIAL_PORT_MONITOR.print("0x");
SERIAL_PORT_MONITOR.print(buf[a], DEC);
SERIAL_PORT_MONITOR.print("\t");
SERIAL_PORT_MONITOR.println();
}
float resultadoec2 = 0.0;
float resultadoac2 = 0.0;
int CA101042=0;
int CA101052=0;
int CA1010242=0;
int CA1010252=0;
if(data0204>=125 && data0204<=255){
CA101042 = 255 - data0204;
CA101052 = 255 - data0205;
}else {
CA101042 = 0;
CA101052 = 0;
}
if(data0204>=0 && data0204<=10){
CA1010242 = data0204;
CA1010252 = data0205;
}else {
CA1010242=0;
 CA1010252=0;
}
resultadoac2 = (256.0 * CA1010242 + CA1010252)/1000.0;
resultadoec2 = (256.0 * CA101042 + CA101052)/1000.0;
```

```
if(resultadoec2 >= 0.60 && resultadoec2 <= 10 ||  resultadoac2 >= 0.60 && resultadoac2
<= 10){
DateTime fecha = rtc.now();
archivo = SD.open("PIDs.txt", FILE_WRITE);
archivo.print(fecha.day());
archivo.print("/");
archivo.print(fecha.month());
archivo.print("/");
archivo.print(fecha.year());
archivo.print(" ");
archivo.print(fecha.hour());
archivo.print(":");
archivo.print(fecha.minute());
archivo.print(":");
archivo.println(fecha.second());
archivo.close();
//1
ENVIARTRAMAFab(PID_BRAKESW1);
taskCanRecv();
 //2
ENVIARTRAMAFab(PID_ABSMALF);
taskCanRecv();
//3
ENVIARTRAMAFab(PID_SIDE_G );
taskCanRecv();
//4
ENVIARTRAMAFab(PID_DELC_G);
taskCanRecv();
//5
ENVIARTRAMAFab(FRWS);
taskCanRecv();
//6
```

```
 ENVIARTRAMAFab(FLWS);
taskCanRecv();
//7
ENVIARTRAMAFab(RRWS);
taskCanRecv();
//8
ENVIARTRAMAFab(RLWS);
taskCanRecv();
//9
ENVIARTRAMAFab(STERING_ANGL);
taskCanRecv();
//10
ENVIARTRAMAFab(YAW_RS);
taskCanRecv();
//1
ENVIARTRAMAStd(PID_ENGINE_SPEED);
taskCanRecv2();
//2
ENVIARTRAMAStd(PID_CAL_LD_VALUE);
taskCanRecv2();
//3
ENVIARTRAMAStd(PID_TRVL_AFTER_MIL);
taskCanRecv2();
//4
ENVIARTRAMAStd(PID_VHCL_SPEED_VSS);
taskCanRecv2();
}
```

```
}
}
}
void CanChequear() {
unsigned char len = 0;
unsigned char buf[8];
if (CAN_MSGAVAIL != CAN.checkReceive()) {
CAN.readMsgBuf(&len, buf);
int data02 = (buf[2]);
int data03 = (buf[3]);
while(data03==0){
Serial.println("Red CAN Falló");
Serial.println("Reintentando...");
delay(1000);
}
}
}
void ENVIARTRAMAcheck(unsigned char PIDStd) {
unsigned char tmp[8] = {0x02, 0x01, PIDStd, 0x00, 0x00, 0x00, 0x00, 0x00}; // Asigna un
véctor con longitud 8
CAN.sendMsgBuf(CAN_ID_PID_STD, 0, 8, tmp); //Envio de la trama
}
void ENVIARTRAMAStd(unsigned char PIDStd) {
unsigned char tmp[8] = {0x02, 0x01, PIDStd, 0x00, 0x00, 0x00, 0x00, 0x00};
SERIAL_PORT_MONITOR.print("Se envió el PID: ");
SERIAL_PORT_MONITOR.println(PIDStd,HEX);
CAN.sendMsgBuf(CAN_ID_PID_STD, 0, 8, tmp); //Envio de la trama
}
void ENVIARTRAMAFab(unsigned char PIDFab) {
unsigned char tmp[8] = {0x03, 0x22, 0x01, PIDFab, 0xFF, 0xFF, 0xFF, 0xFF};
SERIAL_PORT_MONITOR.print("Se envió el PID: ");
SERIAL_PORT_MONITOR.println(PIDFab, HEX);
```

```
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp); //Envio de la trama
}
void taskCanRecv() {
unsigned char len = 0;
unsigned char buf[8];
unsigned char dataswitch[1];
if (CAN_MSGAVAIL == CAN.checkReceive()) {                // check if get data
CAN.readMsgBuf(&len, buf);    // read data,  len: data length, buf: data buf
archivo = SD.open("PIDs.txt", FILE_WRITE);
int data03 = (buf[3]);
int data04 = (buf[4]);
int data05 = (buf[5]);
// int data03St =(buf[3]); suspendidas porque se definieron arriba y aplica para caso std y fabricante
// int data04St =(buf[4]);
SERIAL_PORT_MONITOR.println("\r\n------------------------------------------------------------");
SERIAL_PORT_MONITOR.print("Obteniendo datos de ID: 0x");
SERIAL_PORT_MONITOR.println(CAN.getCanId(), HEX);
int i;
for (i = 3; i == 5; i++) {
SERIAL_PORT_MONITOR.println("\r\n------------------------------------------------------------");
SERIAL_PORT_MONITOR.print("0x");           // QUITAR TODO PRINT SERIAL en el codigo final, unicamente se conserva archivo.print
SERIAL_PORT_MONITOR.print(buf[i], HEX);
SERIAL_PORT_MONITOR.print("\t");
}
SERIAL_PORT_MONITOR.println();
for (int a = 3; a == 5; a++) {
SERIAL_PORT_MONITOR.println("\r\n------------------------------------------------------------");
```

```
SERIAL_PORT_MONITOR.print("0x");
SERIAL_PORT_MONITOR.print(buf[a], DEC);
SERIAL_PORT_MONITOR.print("\t");
SERIAL_PORT_MONITOR.println();
}
if (archivo) {
SERIAL_PORT_MONITOR.println("escribiendo SD...");
archivo.println("\r\n----------------------------------------------------------------");
archivo.print("ID: 0x");
archivo.println(CAN.getCanId(), HEX);
archivo.println();
archivo.print(buf[3],HEX);
archivo.print(",\t");
archivo.print(buf[4],DEC);
archivo.print(",\t");
archivo.println(buf[5],DEC);
archivo.println();
}
float resultado=0;
int CA104 = 0;
int CA105 = 0;
switch (data03){
case 34: //BRAKE SWITCH
 if(data04==64){
archivo.println("Switch de freno ON");
}
if(data04==0){
archivo.println("Switch de freno OFF");
}
break;
```

```
case 37:  //ABS MALF
if(data04==0){
archivo.println("ABS ok");
}
if(data04==64){
archivo.println("ABS malf");
}
break;
case 21: //DECL G
archivo.println("Aceleración longitudinal:");
if(data04==0){
resultado = data05/1000.0;
archivo.print(resultado,4);
archivo.println(" g");
}
if(data04 >= 1 && data04 < 20){
resultado = (256.0*data04+data05)/1000.0;
archivo.print(resultado,4);
archivo.println(" g");
}
if(data04 == 255){
CA105 = 255 - data05;
resultado = CA105/1000.0;
archivo.print("-");
archivo.print(resultado,4);
archivo.println(" g");
}
if(data04 <= 254 && data04 > 234){
CA104 = 255 - data04;
CA105 = 255 - data05;
resultado = (256.0*CA104+CA105)/1000.0;
```

```
archivo.print("-");
archivo.print(resultado,4);
archivo.println(" g");
}
break;
case 22: //SIDE G
archivo.println("Aceleración transversal:");
if(data04==0){
resultado = data05/1000.0;
archivo.print("Izquierda ");
archivo.print(resultado,4);
archivo.println(" g");
}
if(data04 >= 1 && data04 < 20){
resultado = (256.0*data04+data05)/1000.0;
archivo.print("Izquierda ");
archivo.print(resultado,4);
archivo.println(" g");
}
if(data04 == 255){
CA105 = 255 - data05;
resultado = CA105/1000.0;
archivo.print("Derecha -");
archivo.print(resultado,4);
archivo.println(" g");
}
if(data04 <= 254 && data04 > 234){
CA104 = 255 - data04;
CA105 = 255 - data05;
resultado = (256.0*CA104+CA105)/1000.0;
archivo.print("Derecha -");
```

```
archivo.print(resultado,4);
archivo.println(" g");
}
break;
case 15: //FRWS
archivo.println("Velocidad rueda delantera derecha:");
resultado = (data04*10000.0 + (data05/256.0)*10000.0)/1000.0;
archivo.print(resultado,7);
archivo.println(" RPM");
break;
case 16: //FLWS
archivo.println("Velocidad rueda delantera izquierda:");
resultado = (data04*10000.0 + (data05/256.0)*10000.0)/1000.0;
archivo.print(resultado,7);
archivo.println(" RPM");
break;
case 17: //RRWS
archivo.println("Velocidad rueda posterior derecha:");
resultado = (data04*10000.0 + (data05/256.0)*10000.0)/1000.0;
archivo.print(resultado,7);
archivo.println(" RPM");
break;
case 18: //RLWS
archivo.println("Velocidad rueda posterior izquierda:");
resultado = (data04*10000.0 + (data05/256.0)*10000.0)/1000.0;
archivo.print(resultado,7);
archivo.println(" RPM");
break;
case 19: //STEERING ANG
archivo.println("Ángulo de dirección volante:");
if(data04 >= 0 && data04 <= 60){
```

```
resultado = (data04*1000.0 + data05)/100.0;
archivo.print("Derecha ");
archivo.print(resultado,5);
archivo.println(" grados");
}
if(data04 >= 195 && data04 <= 255){
CA104 = 255 - data04;
CA105 = 255 - data05;
resultado = (CA104*1000.0 + CA105)/100.0;
archivo.print("Izquierda ");
archivo.print(resultado,5);
archivo.println(" grados");
}
break;
case 24: //YAW RATE
archivo.println("Ángulo de giro guiñada:");
if(data04<=100){
resultado = (256*data04+data05)/10.0;
archivo.print(resultado,4);
archivo.println(" grados/segundos");
}
if(data04>=100 && data04<=255){
CA104 = 255 - data04;
CA105 = 255 - data05;
resultado = (256*CA104+CA105)/10.0;
archivo.print(resultado,4);
archivo.println(" grados/segundos");
}
break;
}
archivo.println("\r\n----------------------------------------------------------------");
```

```
archivo.close();
}
}
void taskCanRecv2() {
unsigned char len = 0;
unsigned char buf[8];
unsigned char dataswitch2[1];
if (CAN_MSGAVAIL == CAN.checkReceive()) {               // check if get data
CAN.readMsgBuf(&len, buf);    // read data,  len: data length, buf: data buf
archivo = SD.open("PIDs.txt", FILE_WRITE);
int data02st = (buf[2]);
int data03st = (buf[3]);
int data04st = (buf[4]);
SERIAL_PORT_MONITOR.println("\r\n-----------------------------------------------------------
-------");
SERIAL_PORT_MONITOR.print("Obteniendo datos de ID: 0x");
SERIAL_PORT_MONITOR.println(CAN.getCanId(), HEX);
int i;
for (i = 3; i == 5; i++) {
SERIAL_PORT_MONITOR.println("\r\n-----------------------------------------------------------
-------");
SERIAL_PORT_MONITOR.print("0x");
SERIAL_PORT_MONITOR.print(buf[i], HEX);
SERIAL_PORT_MONITOR.print("\t");
}
SERIAL_PORT_MONITOR.println();
for (int a = 3; a == 5; a++) {
SERIAL_PORT_MONITOR.println("\r\n-----------------------------------------------------------
------");
SERIAL_PORT_MONITOR.print("0x");
SERIAL_PORT_MONITOR.print(buf[a], DEC);
SERIAL_PORT_MONITOR.print("\t");
```

```
SERIAL_PORT_MONITOR.println();
}
if (archivo) {
SERIAL_PORT_MONITOR.println("escribiendo SD...");
archivo.println("\r\n----------------------------------------------------------------");
archivo.print("ID: 0x");
archivo.println(CAN.getCanId(), HEX);
archivo.println();
archivo.print(buf[2],HEX);
archivo.print(",\t");
archivo.print(buf[3],DEC);
archivo.print(",\t");
archivo.println(buf[4],DEC);
}
float resultado2=0;
int Carga = 0;
int Revoluciones =0;
int VSS = 0;
int LuzMIL = 0;
switch (data02st){
case 4: //Calculo de la carga
Carga = data03st/2.55;
archivo.println("La carga del motor es:");
archivo.print(Carga);
archivo.print(" porciento");
break;
case 12: //Revoluciones del motor
Revoluciones  = ((256*data03st)+data04st)/4;
archivo.println("Las revoluciones del motor son:");
archivo.println(Revoluciones);
archivo.print(" RPM");
```

```
break;
case 13: //Sensor VSS
VSS = data03st;
archivo.print("La velocidad del vehículo es: ");
archivo.print(VSS);
archivo.println(" km/h");
break;
case 33: //Luz MIL
LuzMIL = ((256*data03st)+data04st);
archivo.println("La distancia con Luz MIL ON:");
archivo.print(LuzMIL);
archivo.println("km");
break;
}
archivo.println("\r\n----------------------------------------------------------------");
archivo.close();
}
}
void solicitud(){
unsigned char tmp[8] = {0x02, 0x10, 0x03, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF};
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp);
// 2
unsigned char tmp1[8] = {0x03, 0x22, 0x04, 0x01, 0xFF, 0xFF, 0xFF, 0xFF};
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp);
// 3
unsigned char tmp2[8] = {0x03, 0x22, 0xDF, 0x04, 0xFF, 0xFF, 0xFF, 0xFF};
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp);
// 4
unsigned char tmp3[8] = {0x03, 0x22, 0x01, 0x76, 0xFF, 0xFF, 0xFF, 0xFF};
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp);
// 5
```

```
unsigned char tmp4[8] = {0x03, 0x22, 0x04, 0x04, 0xFF, 0xFF, 0xFF, 0xFF};
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp);
// 6
unsigned char tmp5[8] = {0x03, 0x22, 0x04, 0x07, 0xFF, 0xFF, 0xFF, 0xFF};
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp);
// 7
unsigned char tmp6[8] = {0x02, 0x10, 0x03, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF};
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp);
// 8
unsigned char tmp7[8] = {0x02, 0x10, 0x01, 0x0C, 0xFF, 0xFF, 0xFF, 0xFF};
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp);
// 9
unsigned char tmp8[8] = {0x03, 0x22, 0x01, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF};
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp);
//10
unsigned char tmp9[8] = {0x03, 0x22, 0x01, 0x52, 0xFF, 0xFF, 0xFF, 0xFF};
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp);
// 11
unsigned char tmp10[8] = {0x03, 0x22, 0x01, 0x80, 0xFF, 0xFF, 0xFF, 0xFF};
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp);
// 12
unsigned char tmp11[8] = {0x04, 0x2F, 0xDF, 0x01, 0x03, 0xFF, 0xFF, 0xFF};
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp);
// 13
unsigned char tmp12[8] = {0x02, 0x3E, 0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF};
CAN.sendMsgBuf(CAN_ID_PID_FABRIC, 0, 8, tmp);
}
void set_mask_filt() {
CAN.init_Mask(0, 0, 0x7FC);
CAN.init_Mask(1, 0, 0x7FF);
```

```
CAN.init_Filt(0, 0, 0x7E8);
CAN.init_Filt(1, 0, 0x7E8);
CAN.init_Filt(2, 0, 0x700);
CAN.init_Filt(3, 0, 0x700);
CAN.init_Filt(4, 0, 0x700);
CAN.init_Filt(5, 0, 0x700);
}
```