BIBLIOTECA INFOTEC
VISTO BUENO DE TRABAJO TERMINAL

Maestría en Sistemas Embebidos (MSE)

Ciudad de México, 7 de noviembre de 2023

Unidad de Posgrados
PRESENTE

Por medio de la presente se hace constar que el trabajo de titulación:

"Plataforma prototipo de comunicación CAN Bus para intercambio de mensajes en aplicaciones automotrices"

Desarrollado por la alumna: **Giselle del Carmen Megchún Pérez,** y bajo la asesoría del **Dr. Jesús Antonio Sosa Herrera** cumple con el formato de Biblioteca, así mismo, se ha verificado la correcta citación para la prevención del plagio; por lo cual, se expide la presente autorización para entrega en digital del proyecto terminal al que se ha hecho mención. Se hace constar que la alumna no adeuda materiales de la biblioteca de INFOTEC.

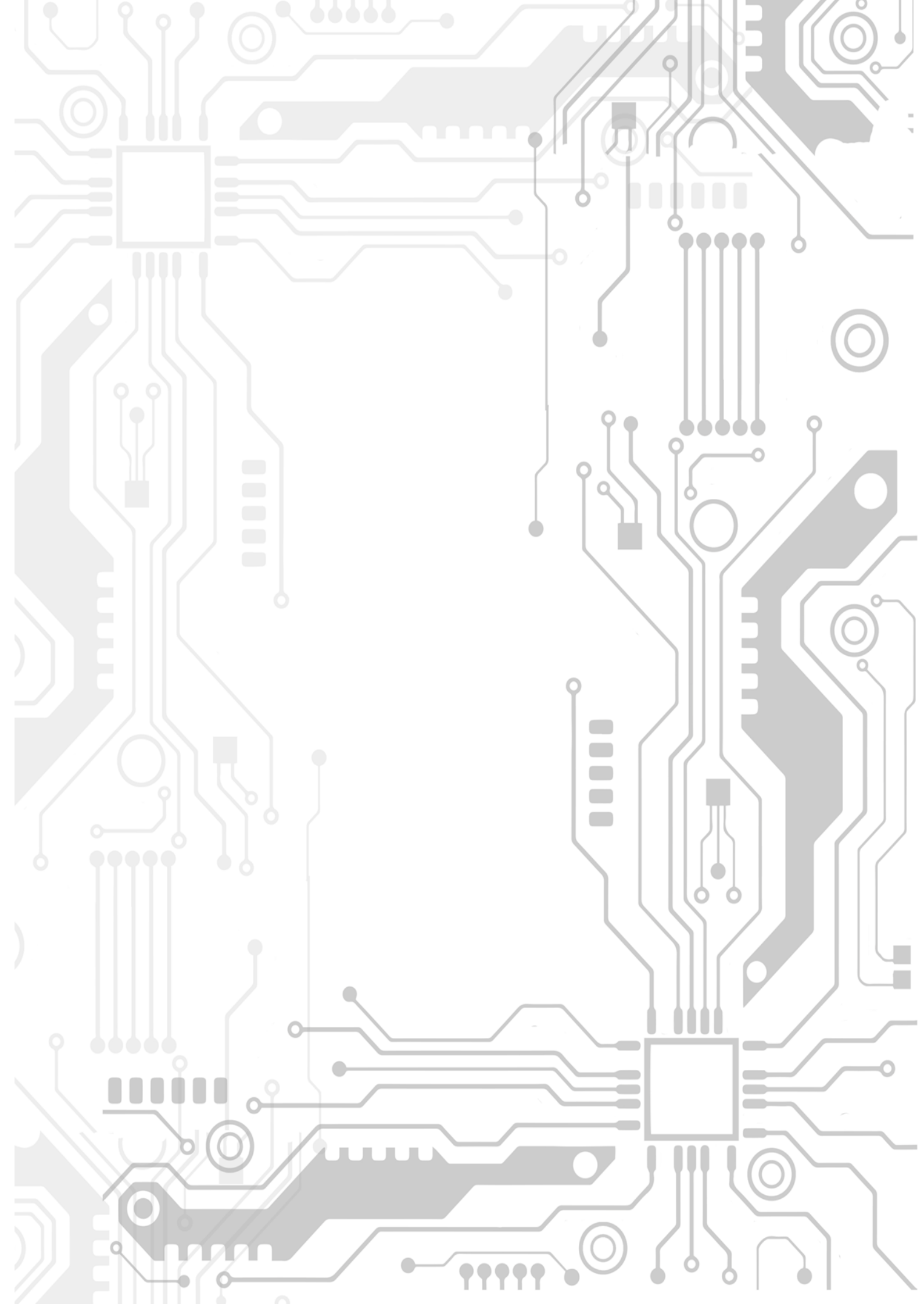
No omito mencionar, que se deberá anexar la presente autorización al inicio de la versión digital del trabajo referido, con el fin de amparar la misma.

Sin más por el momento, aprovecho la ocasión para enviar un cordial saludo.

Mtro. Carlos Josué Lavandeira Portillo
Director Adjunto de Innovación y Conocimiento

CJLP/jah

C.c.p. Felipe Alfonso Delgado Castillo.- Gerente de Capital Humano.- Para su conocimiento
Giselle del Carmen Megchún Pérez.- Alumna de la Maestría en Sistemas Embebidos.- Para su conocimiento.



MSE
MAESTRÍA EN SISTEMAS EMBEBIDOS

INFOTEC CENTRO DE INVESTIGACIÓN E INNOVACIÓN EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN

DIRECCIÓN ADJUNTA DE INNOVACIÓN Y CONOCIMIENTO

GERENCIA DE CAPITAL HUMANO

POSGRADOS

"Plataforma prototipo de comunicación CAN Bus para intercambio de mensajes en aplicaciones automotrices"

Proyecto de desarrollo tecnológico
Que para obtener el grado de MAESTRA EN SISTEMAS EMBEBIDOS

Presenta:

Giselle del Carmen Megchún Pérez

Asesor:

Dr. Jesús Antonio Sosa Herrera

Ciudad de México, Septiembre, 2020

CONACYT

INFOTEC POSGRADOS

Autorización de impresión

Agradecimientos

Durante mi formación académica profesional elaboré un plan para cuando llegara el momento de graduarme, el cuál de forma resumida consistía en lo siguiente. Haría mis practicas profesionales en un empresa fuera de mi estado natal, conseguir empleo en esa compañía, trabajar dos años para adquirir experiencia y estudiar la maestría. Olvidé mencionar que no soy tan buena estimando tiempos y esta ocasión mi intuición no me advirtió. Trabajé más de dos años y no dejé de hacerlo, estudié mientras trabajaba y me gradué poco más de un año después de completar mis créditos lo cuál estaba totalmente fuera del plan. Lo que pareciera un desastre total resultó ser una gran lección de vida. Si no hubiera sucedido tal como pasó, estas palabras no estarían escritas por la misma persona que ahora lo hace.

Agradezco a Dios por darme la oportunidad de seguir sus planes y no los míos. Por dotarme de las herramientas adecuadas para los desafíos a los que me he enfrentado y por acercarme a las personas correctas que me han inspirado, apoyado y acogido. Gracias a mis padres Yuriria y Emilio, mi hermana Majo que a pesar de la distancia han estado presentes en las buenas y en las malas. Siendo mi apoyo, mis fanáticos y mi fuente de amor incondicional.

A mi segunda familia, la cuál afortunadamente conocí al permanecer trabajando durante mis estudios de posgrado. Una madre, un padre y 3 hermanos me ofrecieron un lugar en su hogar cada fin de semana durante dos años. Me trataron como una más y se sumaron a mi propósito sin esperar nada a cambio. Gracias Tonny, Juan, Andrea, Sofi y Juanito porque hicieron de mi experiencia de posgrado una aventura muy grata. Andrea me hizo ensayar el papel de una hermana mayor, fue divertido.

A los amigos que sumé al emprender este viaje, un par de grandes personas que conocí y que valoro formen parte de mi vida hasta el momento. Gracias por su apoyo y su amistad.

Tabla de contenido

Índice de figuras

Índice de cuadros

Siglas y abreviaturas

CAN: Controller Area Network

LIN: Local Interconnect Network

ECU: Electronic Control Unit

UUT: Unit Under Test

HITL: Hardware In The Loop

HAL: Hardware Abstraction Layer

SD: Secure Digital

LCD: Liquid Crystal Display

OBD: On-Board Diagnostics

DTC: Diagnostic Trouble Code

API: application programming interface

Glosario

Bus	La ruta de conducción eléctrica a lo largo de la cual se transmiten los datos dentro de cualquier dispositivo electrónico digital.
Hardware in the loop (HIL)	Técnica en la que las señales reales de un controlador son conectadas a un sistema de pruebas que simula la realidad en tiempo real, estimulando al controlador para que actúe como ensamblado en el producto final [2].
KLine	El K-Line es un protocolo de diagnóstico que cumple con el estándar ISO 14230. Al igual que la interfaz serie estándar RS232, se basa en la tecnología de los circuitos UART (Universal Asynchronous Receiver Transmitter) típicos.
Protocolo	Un protocolo pertenece a las normas desarrolladas para regular la transmisión de datos entre redes y dispositivos que especifica el método en el que los datos son empaquetados por la fuente para ser recibidos, analizados y comprendidos por el receptor [18]. Generalmente, esto establece las reglas de comunicación entre los dispositivos. Los protocolos individuales tienen una capacidad incorporada para detectar y determinar paquetes de datos escritos en base a ese protocolo específico [17].

Sistemas distribuidos embebidos	Sistema embebido con varios elementos de procesamiento que están físicamente separados y que son conectados por una red que les permite comunicarse [19].
Stand Alone	Destinado, diseñado o capaz de usarse o funcionar solo o por separado sin que requiera conexión a otro dispositivo.
Tiempo real o Real-Time	El lapso en el que ocurre un proceso físico o de control por computadora sin demora o asíncronismo.

Introducción

La industria automotriz ha crecido constantemente en diversas partes del territorio Mexicano y ha dado lugar al gran corredor industrial, así mismo, la demanda de aportaciones tecnológicas en distintas áreas automotrices han incrementado, por ejemplo, el de pruebas.

Dentro del programa de posgrados con la industria que mantiene CONACyT a través del Centro de investigación especializado en el desarrollo de tecnologías de la información y en colaboración con el Centro de Investigación en Matemáticas, unidad Guanajuato se desarrolló el presente trabajo titulado "Plataforma prototipo de comunicación CAN Bus para intercambio de mensajes en aplicaciones automotrices". El objetivo fue desarrollar un módulo de intercambio de mensajes integrado en un sistema embebido que proporcione acceso a señales de control y señales de estado mediante el protocolo Controller Area Network.

La necesidad de proveer un dispositivo hardware in the loop para la implementación de pruebas y diagnostico se responde a tavés de la aportación de este proyecto que fue ejecutado con base al modelo del ciclo de desarrollo de un sistema embebido propuesto por INFOTEC. Se limitaron y adecuaron las etapas de dicho modelo para desarrollar este prototipo en particular. Durante el progreso se implementaron cinco etapas que incluyen desde la delimitación teorica y alcance del sistema, desarrollo de Software, desarrollo de Hardware hasta la integración y pruebas correspondientes.

El prototipo integra tecnología de STMicroelectronics dentro del desarrollo de hardware por lo que el desarrollo del Software es la principal aportación de este trabajo. Se desarrolló el sofware embebido bajo la arquitectura Hardware Abstraction Layer y con las librerias STM32 HAL con la finalidad de aprovechar el hardware embebido sin programarlo a bajo nivel. Esto nos permite interactuar desde una capa más arriba y a un nivel más general. La ventaja de lo anterior mencionado yace en poder interactuar con el hardware sin acceder a las operaciones primitivas del set de instrucciones especificas.

Los resultados obtenidos muestra el tráfico de mensajes transmitidos a través de un sistema mecatrónico que integra un vehículo y la posibilidad de mandar mensajes y monitorear los estados del sistema. El prototipo es configurable para enviar y coordinar mensajes de tipo control o diagnóstico a través de su interfaz.

En conclusión el presente trabajo describe el desarrollo tecnológico de un dispositivo útil para monitoreo y control de subsistemas vehiculares, reconfigurable y operable en campo, laboratorio y línea de producción según las necesidades y aplicaciones para los cuales se pretenda emplear.

Capítulo 1

Antecedentes y Justificación

1 Antecedentes y Justificación

Un vehículo está compuesto por múltiples subsistemas mecatrónicos que interactúan entre sí para brindar la experiencia de confort y seguridad mientras se viaja. Los controladores de cajas de velocidades automáticas; limpiadores de parabrisas; frenos ABS; estado de las bolsas de aire; vidrios, puertas y pantalla son algunos de los componentes que integran un coche y que necesitan estar comunicados entre ellos para su monitoreo y control. En la industria automotriz principalmente, los estándares de calidad aumentan y en consecuencia los sistemas de prueba y control autónomos en las estaciones de trabajo son implementadas para obtener las siguientes ventajas.

- Procesos eficientes.
- Reducción de tiempos de producción.
- Reducción de costos.
- Disminución de mermas y desperdicios.
- Versatilidad en actualizaciones y modificaciones.

Los avances tecnológicos para control en los sistemas de producción automotriz demandan el desarrollo de dispositivos que gestionan, de forma electrónica, las funciones del ambiente en la que la unidad bajo prueba se desarrollará. Procesadores capaces de facilitar la implementación de pruebas y que respondan a las constantes mejoras y cambios que se ejecutan en las líneas de producción.

“La tecnología CAN (Controller Area Network) es usado en la industria automotriz como uno de los principales protocolos seriales estandarizados en modernos y complejos sistemas electrónicos y mecatrónicos” [6]. Múltiples subsistemas son monitorizados a través de CAN Bus dentro de la arquitectura de un coche, los techos solares por ejemplo, deben ser probados y monitorizados a lo largo de su manufactura en la línea de producción durante la ejecución de movimientos como abrir y cerrar cristal; abrir o cerrar cortinillas y posición de venteo.

1.1. Dispositivos de control para pruebas

Los sistemas mecatrónicos automotrices son dispositivos complejos que exigen cumplir ciertos estándares en el proceso de producción con el fin de cumplir los requerimientos de calidad del cliente. Así mismo, los sistemas que son implementados en las estaciones de prueba deben garantizar un desempeño de alto nivel sobre comunicación en Tiempo real o Real-Time, robustez y confianza.

Las aplicaciones de pruebas que utilizan el Protocolo CAN Bus son de frecuente uso dentro de la industria automotriz, los dispositivos que trabajan bajo Protocolo CAN deben ser probados bajo las condiciones donde originalmente estarán operando por lo tanto, es necesario emular un ambiente en Tiempo real o Real-Time para su función. Para apoyar a los ingenieros de pruebas, de control y de calidad a implementar pruebas usando CAN Bus, herramientas profesionales y especializadas han sido desarrolladas. Dichos dispositivos de control, son utilizados para mandar mensajes a través del Protocolo a cualquier UUT (Unit Under Test). Estos mensajes son solicitudes a la unidad bajo prueba, en este documento hablaremos de señales comúnmente utilizadas en compañías de ensamble de subsistemas automotrices, dichas señales son las de control para ejecutar movimientos de mecanismos o las de diagnóstico, que son peticiones para monitorear el estado en la UUT [10].

Como resultado de la adaptabilidad del Protocolo CAN Bus, es bastante económico su uso e implementación, la infraestructura no es tan costosa, hablamos de un estándar libre y que se transmite a través de un Bus que consiste en dos cables. Compañías como Lipowsky Industry y Vector han desarrollado una línea entera de dispositivos de control bajo Protocolo LIN y CAN que son configurables con el software del fabricante. Son equipos capaces de monitorear e intervenir el hardware a bajo nivel dentro del Bus para mandar y recibir mensajes, así como emular condiciones de operación de una unidad bajo prueba.

Los principales módulos de un sistema en esta área de aplicación son:

- Diagnostico.
- Software embebido y comunicación con ECUs.
- Control.
- Emulación.
- Eventos IOs.

El alcance de las operaciones automotrices están en constante cambio debido a la tendencia de mejora continua, por lo tanto, los dispositivos de control para aplicaciones automotrices deben ser reconfigurados para ajustarse a las nuevas condiciones de prueba, a pesar de ello, la tarea de programar los controladores CAN resulta una tarea complicada para los ingenieros que no tienen un conocimiento del funcionamiento del Protocolo y de algún lenguaje de programación. Dichos dispositivos mencionados son robustos y de uso industrial por lo que ocupan un lugar importante en las soluciones de la industria automotriz, sin embargo, son dispositivos que orillan a los usuarios a depender del soporte de la marca y de requerir un especialista en el tema para la configuración de los sistemas de pruebas, además requieren de licencias o llaves para habilitar los puertos de comunicación limitando la distribución de las aplicaciones desarrolladas con esa tecnología. Indiscutible es la ventaja principal de los dispositivos que ofrecen Vector y Lipowsky la cual permite el acceso al control en bajo nivel de los mensajes enviados a las unidades conectadas al sistema.

Por otro lado, los dispositivos de menor costo no tienen acceso al Hardware o Software y simplemente están limitados a la lectura de mensajes, por lo cual, no son considerados dentro de la mayoría de las aplicaciones industriales.

A pesar de la madurez de los productos actualmente posicionados en el mercado para la ejecución de pruebas bajo Protocolo CAN, existen un área importante de oportunidad para mejorarlos tecnológicamente.

El prototipo en desarrollo descrito en el presente trabajo ofrece ciertas ventajas que se explican a continuación. Los sistemas actualmente en el mercado requieren, además

de contraer el equipo, adquirir las licencias de software para operar y configurar las tareas en el dispositivo. Además, licencias de firmware para activar los canales del Bus a utilizar; CAN Bus LIN Bus o KLine según sea la necesidad del cliente.

En consecuencia, promueven dependencia de las licencias del fabricante del producto y del soporte técnico que se pueda requerir, los usuarios están restringidos en relación a la distribución de sus aplicaciones. Dichas restricciones engloban el duplicado, modificación, operación e interpretación de sus aplicaciones desarrolladas bajo la plataforma y equipo del fabricante.

En el presente trabajo se abordará como tarea principal de los controladores CAN la siguiente.

- Mandar mensajes a la unidad bajo prueba para emular un ambiente de ciclo cerrado para ser capaces de tomar decisiones o monitorear el estado de nuestra ECU (Electronic Control Unit).

El prototipo terminado en este proyecto nos acerca a un primer módulo de operación similar al de los dispositivos de Lipowsky o Vector. Este módulo de comunicación bajo protocolo CAN está integrado en un sistema embebido perteneciente a la familia ST Microelectronics para validación del concepto y depuración del prototipo.

Se concluirá con la realización de ciertas pruebas en laboratorio con ciertos instrumentos monitores y analizadores especializados para afirmar un comportamiento admisible de rendimiento y asistencia en el tráfico de mensajes fiable y sin interrupciones.

1.2. Objetivos

1.2.1. Objetivo General

Desarrollar un módulo para generar mensajes integrado en un sistema embebido que proporcione acceso a señales de control y señales de estado que acepte el protocolo CAN y dichas señales puedan ser manipuladas y emitidas por software y reenviado a otros sistemas embebidos.

1.2.2. Objetivos Particulares

- Emular el estado de un sistema mecatrónico bajo protocolo CAN.
- Enviar mensajes de estado a través del software embebido a la unidad bajo prueba vía protocolo CAN.

Capítulo 2

Estado del Arte

2 Estado del Arte

2.1. Controladores CAN

Muchos desarrollos tecnológicos se enfocan en mejorar la industria automotriz, las áreas de mayor explotación son las que buscan brindar mayor comodidad en el vehículo, garantizar seguridad para disminuir los riesgos de un accidente y finalmente la eficiencia energética y protección del medio ambiente. Los campos mencionados son afrontados por unidades electrónicas de control, ECU. Al principio con una única ECU y luego con el agregado de otras. Entonces surge la necesidad de establecer una adecuada comunicación entre las distintas unidades de control de dichos procesos.

Se estima que un automóvil bien equipado actualmente usa más de 70 unidades de microcontroladores (MCU) y cada uno de ellos puede tener información que debe compartirse con otras partes de la red. Como resultado, la red en el vehículo ha creado una evolución silenciosa en la tecnología automotriz, lo que resultó en la eliminación de arneses de cableado difíciles de manejar una vez usados en circuitos de control [8].

La necesidad de interconectar estas unidades de control en la arquitectura vehicular da lugar al desarrollo de protocolos especializados para eliminar los extensos cableados. Dentro de esos protocolos desarrollados se encuentra el CAN que es uno de los más populares y utilizados en los sistema vehiculares además del LIN y K-Line.

A su vez, garantizar la intercomunicación de los sistemas internos impulsa a los proveedores de las unidades de control electrónicas a actualizar sus estrategias y métodos de pruebas demandando sistemas Stand Alone y Hardware in the loop (HIL) (hardware in the loop). Compañías han dedicado sus investigaciones y desarrollos tecnológicos al área automotriz, en especial a los dispositivos de control para pruebas.

Vector Company se ha dedicado durante 30 años al desarrollo de la electrónica automotriz. Tiene presencia en más de 30 locaciones y ha desarrollado soluciones que ayudan a los ingenieros a gestionar sus tareas más especiales. [9].

Lipowsky Industry es una empresa Alemana fundada hace 25 años, se especializan en sistemas LIN y CAN-Bus para aplicaciones multitarea en Tiempo real o Real-Time. Desarrollan y producen unidades electrónicas equipadas con microcontroladores para aplicaciones automotrices [12].

Enseguida en el Cuadro 2.1 se muestra la comparación de los principales productos que ofrecen las marcas anteriormente mencionadas y que se utilizan en la industria automotriz.

Cuadro 2.1: Comparación de dispositivos de simulación CAN Bus.

Fuente: Elaboracion propia.

Marca	Vector	Lipowsky
Línea	**VN1600**	**Baby-LIN-RM-ll**
		
Aplicaciones	Análisis de Bus simples. Tareas de diagnóstico, calibración y programación flash en el laboratorio, en el banco de pruebas, en el taller de servicio o en el vehículo.	Simulación móvil, Bus CAN y registro de datos con pantalla, teclado y batería. Permite observar y analizar el tráfico de señales en Tiempo real o Real-Time así como grabarlas en una memoria SD. Además, posibilita ejecutar macros para ser enviados en el Bus.
Descripción	Interfaces de red con USB para comunicación CAN y LIN.	Dispositivo de simulación Multibus con interfaz de I/O. Interfaces LIN, CAN-HS y CAN-LS.
Características	Alimentación suministrada por USB. Rendimiento óptimo para CANoe, CANalyzer, CANape o aplicaciones de clientes, con soporte para múltiples aplicaciones. LED para mostrar la actividad del canal y el estado del dispositivo.	Funcionalidad IO integrada. Dispositivo portátil con batería. LCD gráfico. Teclado con 12 teclas. Menús configurables. Funciona con baterías, con un tiempo de ejecución de hasta 10 horas. Se pueden almacenar y recuperar varias configuraciones en una tarjeta SD de 4 GB.

Estas compañías se especializan en desarrollar soluciones para la industria automotriz, sin embargo, es necesario señalar la tecnología desarrollada al momento corresponde a sistemas propietarios además de los precios bastante elevados, las licencias que conceden los fabricantes al comprar sus productos son restrictivas con respecto a las aplicaciones que el usuario final puede darle, sobre todo al referirnos a la distribución del desarrollo y extensiones de los mismos. No se trata de opciones open hardware ni mucho menos open software. La replica de las aplicaciones desarrolladas en dichos equipos requiere estrictamente de nuevas licencias por cada estación de trabajo complicando la distribución de los sistemas de pruebas.

En la Figura 2.1 se puede observar una propuesta económica de un dispositivo de control CAN Bus. Como se puede observar en la imagen la inversión para un solo sistema de comunicación LIN a implementar da un total $ 80,671.14 pesos mexicanos.

SERVICIOS
ELECTROMECANICOS Y
ERGONOMÍA DE SILAO S.A. DE C.V.

Blvd. Emiliano Zapata núm. 23, Colonia Noria de Sopeña, C.P. 36118, SILAO, GUANAJUATO, MÉXICO .
RFC: SEE 040303 LI3

19 de Noviembre del 2019
COT: SEE 108-1119

Cliente: Departamento de Compras
Tel: 01 -473 732 7155
CIMAT

Por medio de la presente, nos es grato saludarle y poner a su disposición nuestra propuesta económica para el suministro:

Equipo Lipowsky. Entrega en sus instalaciones.

DETALLE DEL SUMINISTRO

N°	Cant.	CONCEPTO	MONTO COTIZADO SIN IVA	IMPORTE
1	1	Option BL-MB-II MIF-LIN02 8000872 (Hardware, Puerto de expansión LIN).	$22,313.47	$22,313.47
2	3	Option-BL-MB-II-MIF-DIO as upgrade module. 8000890 (Hardware, Puerto de entradas y Salidas).	$5,243.37	$15,730.11
3	1	Baby-LIN-MB-II (Hardware, Dispositivo de simulación multibus).	$42,627.56	$42,627.56
		Incluye entrega en sus instalaciones.		
			Subtotal	**$80,671.14**

NO INCLUYE IVA

Figura 2.1: Cotización para suministro de equipo de la marca Lipowsky.
Fuente: Departamento de vinculación, Centro de Investigación en Matemáticas, 2019.

Capítulo 3

Marco Teórico

3 Marco Teórico

3.1. Definiciones Útiles

3.1.1. Fundamentos CAN

El Bus CAN (Controller Area Network) es un Bus para vehículos estandarizado y diseñado para permitir la comunicación entre los distintos subsistemas mecatrónicos de un vehículo . El protocolo CAN fue desarrollada por Robert Bosch GmbH para aplicaciones automotrices a principios de 1980 y lanzado públicamente en 1986 [5]. La especificación CAN de Bosch se convirtió en un estándar ISO (ISO 11898) en noviembre de 1993, el protocolo CAN es estandarizado bajo la norma ISO 11898 en la que se define la capa física para velocidades de transferencia de datos de hasta 1 Mbps; un Bus con tolerancia a fallos para aplicaciones de baja velocidad hasta 125Kbit/s estandarizado bajo la norma ISO 11519 y luego actualizado por la ISO 11519-2 y ampliado en 1995 para permitir identificadores de dispositivo más largos (CAN 2.0B) [11].

Una comunicación punto a punto supondría una gran cantidad de cables que aumentarían el peso y la complejidad del sistema, además restarían volumen que se puede aprovechar para otras funciones. Es por esto que se necesita una red multiplexada, es decir, una red con un circuito único que intercomunique todos los elementos con un cableado de tipo Bus.

Todos los dispositivos electrónicos que estén conectados al Bus CAN, recibirán todos los mensajes que publiquen otros dispositivos, y a su vez, todos los mensajes que envíe podrán ser leídos por cualquier elemento conectado al Bus. Además el Bus de datos CAN tiene la particularidad de que, si falla un componente, el resto del sistema continúa funcionando normalmente, reduciendo en gran medida el riesgo de un fallo total del sistema. Otra de las ventajas de este protocolo, reside en que el procesador principal delega la carga de comunicaciones a un periférico independiente que las controla. De esta manera, el procesador principal puede dedicar más recursos a realizar sus ta-

reas [5].

3.1.2. CAN Bus en la industria automotriz

Los principales campos de aplicación de la redes CAN en el automóvil incluyen [4].

- Acoplamiento de unidades de control.
- Electrónica de carrocería y de confort.
- Comunicación móvil.

Algunos de los módulos instalados, más utilizados por diferentes marcas de autos, son por ejemplo [22]:

- PCM: Control electrónico del tren motriz (Conjunto motor y transmisión).
- TCM: Control electrónico de la transmisión.
- ABS: Control electrónico sistema de frenos anti-bloqueo.
- BCM:Control electrónico de la carrocería.
- IPC: Módulo de control del cuadro de instrumentos.
- EHPS: Módulo de control del sistema asistencia electro-hidráulica.
- TPM: Sistema de monitoreo presión neumáticos.
- INMO: Sistema de control de seguridad llave codificada.
- VTD: Sistema de detección sistema anti-robo.
- EBTCM: Sistema de frenos anti-bloqueo para GM.
- DDM: Módulo de la puerta del conductor.
- RFA: Módulo Radio Frecuencia.
- IPM: Módulo de fusibles electrónicos.

3.1.3. Bus

La capa física consta de dos cables trenzados conectados a todos los transmisores y receptores: CAN High (o CANH) y CAN Low (o CANL).

Este Bus se utiliza en el automóvil como bus de comunicaciones de lo que se conoce como Sistemas distribuidos embebidos, es decir que permite las comunicaciones entre las diferentes unidades de control electrónicas que están empotradas en los sistemas que controlan. Dentro de los vehículos pueden existir varias redes CAN, aunque normalmente son dos. Se tendrá una red de alta velocidad que se encargará de comunicar en Tiempo real o Real-Time elementos como la inyección de combustible, ABS, frenado del vehículo, etc. Mientras que la red de baja velocidad se encargará de conectar los elementos menos relevantes como los sistemas de climatización, etc [4].

La topología lineal se usa comúnmente, asegurando la continuidad del intercambio de datos a pesar de un nodo dañado. Para evitar reflexión de ondas electromagnéticas, en los extremos del Bus están instaladas unas resistencias de 120 +-18 Ohm, llamadas resistencias terminales. Los datos se transmiten en forma diferencial, es decir, es necesario restar el valor de tensión del CANL al CANH para conocer la señal del bit [14]. La representación de lo mencionado se puede observar en la Figura 3.1.

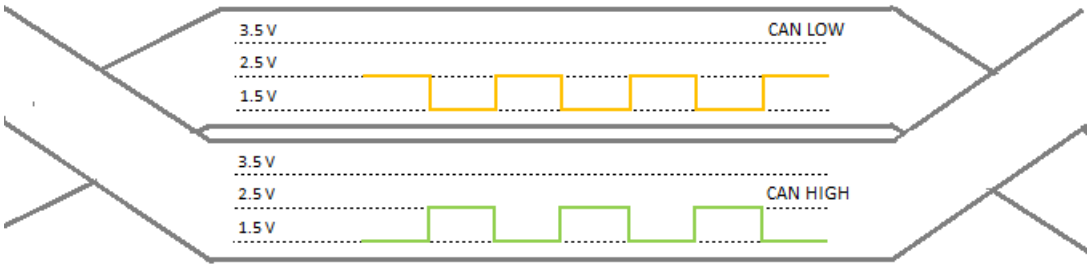


Figura 3.1: Diagrama de pulsos en transmisión de la señal CAN sobre par trenzado. Fuente:Elaboración propia.

La atenuación de una señal es proporcional a la longitud del conductor, dicha atenuación aumenta con la frecuencia y por tanto con la velocidad de transmisión. Una lon-

gitud excesiva en un conductor también aumenta su susceptibilidad al ruido.

Dado que la señal no se propaga inmediatamente, sino que tiene un cierto retardo, la longitud del Bus está limitada y depende de la velocidad de transmisión tal y como se muestra en el Cuadro 3.1 [5]

Cuadro 3.1: Relación velocidad del Bus / longitud del cable.

Fuente: Elaboración propia.

Velocidad del Bus	Tiempo del Bus	Longitud del Bus
1 $Mbit/s$	1 μs	30 $metros$
500 $Mbit/s$	2 μs	100 $metros$
250 $Kbit/s$	4 μs	250 $metros$
125 $Kbit/s$	8 μs	1000 $metros$
50 $Kbit/s$	20 μs	500 $metros$
20 $Kbit/s$	50 μs	2500 $metros$
10 $Kbit/s$	1 μs	5000 $metros$

En el estándar ISO 11898 está especificado dos niveles lógicos de transmisión que son denominados dominante y recesivo [11]:

- Dominante. La diferencia de tensión entre

$$CAN_H \rightarrow CAN_L \tag{3.1}$$

  es aproximadamente de unos 2V, con

$$CAN_H = 3.5V \quad y \quad CAN_L = 1.5V \tag{3.2}$$

- Recesivo. La diferencia de tensión entre

$$CAN_H \rightarrow CAN_LOW \tag{3.3}$$

  es aproximadamente de unos 0V, con

$$CAN_H = CAN_L = 2.5V \tag{3.4}$$

3.1.4. Estructura de Datos

Los mensajes a transmitir desde cualquier nodo o unidad de control en una red CAN no contienen la dirección del nodo emisor ni del nodo receptor. La información se descompone en mensajes asignándoles un identificador único, este identificador indica la prioridad del mensaje. El mensaje de mayor prioridad accede al Bus, donde cada nodo receptor filtra el mensaje y decide o no aceptarlo, mientras que los mensajes de menor prioridad se retransmitirán automáticamente en los siguientes ciclos de Bus [3].

El protocolo CAN contempla diferentes formatos de tramas, cada una orientada a un propósito específico dentro del funcionamiento del protocolo. Las más importantes son [16]:

- Tramas de datos. Útiles para el paso de información entre nodos, que puede ser enviada y recibida por uno o varios de ellos.
- Tramas de error.
- Tramas de petición remota.Para solicitar el envío de la información otro nodo.
- Tramas de sobrecarga. Es enviada por un nodo cuando está sobrecargado, lo que provoca que el Bus incluya un retardo extra entre tramas.

Los mensaje de tipo estándar, son los más comunes en la comunicación interna de los vehículos. Como se muestra en la Figura 3.2, constan de los siguientes campos:

- Cabecera o "Arbitration Field". El primer bit es el inicio de la trama (SOF). Este bit dominante representa el comienzo de un mensaje CAN. El siguiente es el identificador de 11 bits, que establece la prioridad del mensaje CAN. Cuanto más pequeño es el identificador, mayor es la prioridad del mensaje. El bit de solicitud de transmisión remota (RTR) es normalmente dominante, pero se vuelve recesivo cuando un nodo solicita datos de otro. El bit de extensión de identificador (IDE) es dominante cuando se envía una trama CAN estándar y no una extendida. El bit r0 está reservado y no se usa actualmente. Con la información de estos campos, los componentes electrónicos deciden si el mensaje va dirigido a ellos o

no [1,5].

- Celda de control. Es el campo de información del tipo de mensaje. Incluye información como el número de Bytes que va a tener el mensaje que oscila entre 0 y 8 bytes o el tipo de mensaje que es (estándar, extendido o remoto) [1,5].
- Campo de datos. Es donde se encuentra la información que se quiere transmitir.
- Campo de detección de errores o CRC (Cyclic Redundancy Check). Sirve para comprobar si el mensaje se ha transmitido correctamente.

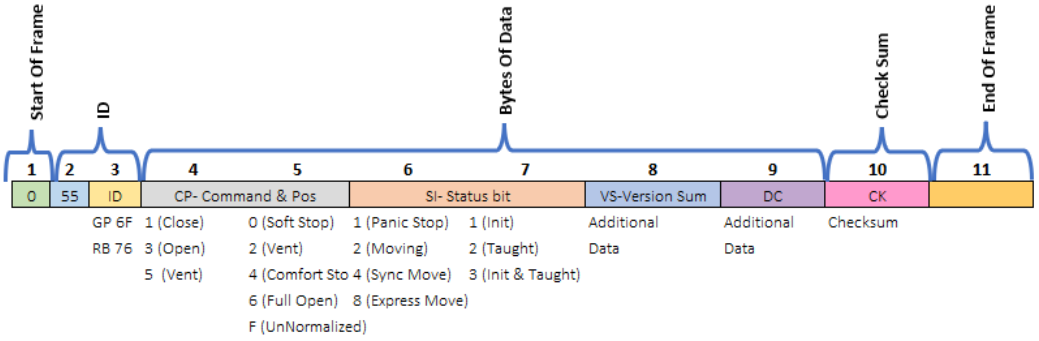


Figura 3.2: Descripción de un mensaje CAN para un techo solar VW416.
Fuente: Elaboración propia. Basado en especificaciones VW416.

3.1.5. Comunicación y Diagnóstico

Actualmente, el bus CAN es el protocolo más extendido en el sector automovilístico y es por el que circulan mensajes con los parámetros más importantes [5].

La comunicación a través de CAN está orientado a los mensajes. Los mensajes a transmitir desde cualquier nodo o unidad de control en una red CAN no contienen la dirección del nodo emisor ni del nodo receptor [3]. Cada nodo perteneciente a la red recibe el mensaje y decide si es de su interés o no. Al ser filtrado se descartan aquellos que no necesitan atención por el nodo.

El acceso al bus CAN es muy utilizado para tareas principalmente de diagnosis, ya que leyendo los mensajes que se transmiten se pueden conocer parámetros de los componentes del vehículo, y además se pueden enviar peticiones de datos específicos para saber el estado de un componente en concreto. Aunque existen algunos mensajes de

diagnosis estándar, la mayoría son específicos de cada fabricante, y en general no existe una base de datos explicando el significado de cada mensaje [5].

El diagnóstico a bordo (OBD) es el sistema de diagnóstico e informes del vehículo que le permite al usuario solucionar problemas a través de códigos de diagnóstico (DTC). Cuando se enciende la luz de revisar motor", un técnico a menudo usará un dispositivo portátil para leer los códigos del motor del vehículo. En el nivel más bajo, estos datos se transmiten a través de un protocolo de señalización, que en la mayoría de los casos es CAN [1].

El sistema OBD II al ser un sistema para el diagnóstico, su función primordial es monitorear el estado de los subsistemas que integran el vehículo.

Un conector del sistema OBD II tiene que cumplir las especificaciones según la normativa, ISO 15031-3:2004 [3].

3.1.6. Sistemas de prueba Hardware in the Loop, HIL

Es un procedimiento de prueba para verificar y validar sistemas basados en software sobre bancos de pruebas dónde existe una gran interacción de datos.

Las pruebas Hardware in the loop (HIL) consisten en simular el entorno de control y monitoreo de las señales generadas por los subsistemas de un vehículo enfocados a la unidad bajo prueba ECU.

Por ejemplo, las configuraciones de prueba pueden componer un ambiente de prueba en el que un vehículo circula a 120 km/h sobre una recta, las ventanas traseras están abiertas y el conductor activa la señal para abrir el techo solar. Los sensores conectados envían mensajes a la ECU sobre temperatura, posición, velocidad que son utilizados por el ECU para tomar una decisión sobre la velocidad de apertura o límites de apertura del techo solar para no ocasionar un daño al mismo.

Las simulaciones pueden ser tan complejas como los casos de pruebas lo establezcan. Los ingenieros de prueba utilizan este método para poder ejecutar pruebas sobre escenarios que sería muy complicado ejecutar en campo o que estadísticamente podrían

representar un riesgo alto para los operadores.

Un banco de pruebas Hardware in the loop (HIL) consta de la ECU bajo prueba, una interfaz de entradas y salidas y una interfaz de software que crea los escenarios de los casos de y analiza los resultados de cada prueba. [2].

Capítulo 4

Metodología y Desarrollo

4 Metodología y Desarrollo

4.1. Modelo de desarrollo tecnológico.

Un modelo de desarrollo tecnológico por definición se refiere al: “Uso sistemático del conocimiento y la investigación dirigidos hacia la producción de materiales, dispositivos, sistemas o métodos incluyendo el diseño, desarrollo, mejora de prototipos, procesos, productos, servicios o modelos organizativos” [13]. En la Figura 4.1 se expone el modelo de desarrollo tecnológico que ha sido seleccionado y bajo el cuál se basó el desarrollo del presente trabajo de investigación. Con base al modelo del ciclo de vida de un sistema embebido propuesto por INFOTEC se tomaron las etapas necesarias y se dirigieron hacia el alcance del objetivo del proyecto.

Figura 4.1: Modelo de desarrollo de Sistemas Embebidos.
Fuente. Elaboración propia.

4.1.1. Metodología de Desarrollo de un Sistema Embebido.

En la elaboración del proyecto final, el desarrollo de software destaca como la principal mejora con respecto a los dispositivos existentes en el mercado. En consecuencia se consideró el modelo de desarrollo en cascada que se caracteriza por dividir los procesos de desarrollo en sucesivas fases como un procedimiento lineal [13]. Este modelo es frecuentemente implementado para el desarrollo de software que al contrario que en los modelos iterativos, cada una de estas fases se ejecuta tan solo una vez. Así, en combinación con el modelo de desarrollo tecnológico las etapas y tareas se orientan al área de software.

En la práctica, se aplican diversas versiones del modelo. Los más habituales son los modelos que dividen los procesos de desarrollo en cinco fases. En ocasiones, las fases 1, 2 y 3 definidas se integran en una sola fase de proyecto a modo de análisis de los requisitos [13].

1. Análisis: planificación, análisis y especificación de los requisitos.
2. Diseño: diseño y especificación del sistema.
3. Implementación: programación y pruebas unitarias.
4. Verificación: integración de sistemas, pruebas de sistema y de integración.
5. Mantenimiento: entrega, mantenimiento y mejora.

A continuación se plantea la metodología de desarrollo a implementar para el prototipo:

1. Análisis y Diseño. “En esta etapa se realiza un análisis de diversos aspectos relacionados al proyecto a implementar. Tendencias del mercado, ideas innovadoras, acceso a clientes y usuarios finales” [3]. Requerimientos.
    - Benchmarking: Investigar los dispositivos que existen en el mercado y ocupan los primeros lugares como opción en las soluciones para la industria automotriz. Comparar la funcionalidad de los equipos e identificar las ventajas y desventajas.

- Investigar algunos desarrollos de dispositivos CAN para el monitoreo y control de sistemas automotrices.
- Definir requerimientos de Hardware.
    - El sistema tendrá comunicación con la unidad de control a través de una cable USB.
    - El sistema debe operar con un Bus CAN High Speed.
    - El dispositivo deberá ser alimentado con una entrada de voltaje de 9-24 VCD con protección de sobrevoltaje.
    - El dispositivo deberá tener como máximas dimensiones 160x100x80mm.
    - El dispositiva se conectará punto a punto con la unidad bajo pruebas.
    - El dispositivo deberá tener una interfaz dónde su pueda observar los mensajes recibidos desde la unidad bajo prueba.
- Definir requerimientos de Software.
    - El sistema será capaz de enviar mensajes (frames) desde la unidad de control a la unidad de prueba mediante el uso de protocolo CAN Bus.
    - El sistema será capaz de recibir mensajes (frames) desde la unidad bajo prueba a la unidad de control mediante el uso de protocolo CAN Bus.
    - El sistema deberá emular un maestro o un esclavo en caso de ser necesario para completar el ciclo de comunicación.
    - El sistema permitirá que los mensajes de envío a la unidad bajo prueba sean reconfigurables.
    - El sistema deberá realizar reintentos de envío de mensajes en caso de que exista una colisión en los mensajes.
    - El sistema deberá indicar cuando la unidad bajo prueba sea inaccesible.

- El sistema deberá indicar cuando exista un error en el envío de mensajes.
- El sistema deberá indicar cuando exista un error en la recepción de mensajes.
- El sistema deberá calcular el CRC (Cyclic Redundancy Check) correspondiente por cada frame enviado a la unidad bajo prueba.

Arquitectura

- Identificar la estructuras y arquitectura de operación en el envío de mensajes bajo protocolo CAN.
- Definir la arquitectura del Sistema.
- Definir los módulos de software y su arquitectura así como la interacción entre ellos.

A continuación en la Figura 4.2, se muestras algunas de las principales actividades relacionadas con las áreas de trabajo del proyecto a desarrollar. Estas actividades están orientadas a los entregables de los módulos que integraran el sistema entero.

Figura 4.2: Actividades del modelo de desarrollo.
Fuente: Elaboración propia.

Diseño

El diseño de las capas se proyectó al uso de la arquitectura HAL que ofrece las ventajas de utilizar un API independiente de hardware subyacente integrando la biblioteca estándar newlib ANSI C que proporciona las funciones de la biblioteca

estándar C. Además ofrece otros servicios de acceso para dispositivos periféricos, proporciona una interfaz estándar para acceso a dispositivos, manejo de interrupciones e instalaciones de alarmas. Otra de las ventajas de utilizar HAL es la flexibilidad para configurar e inicializar tareas antes de entrar al bucle principal. Esto nos permite tener un modulo estructurado y dividir el proceso de pruebas en fases.

- Diseño de las capas del software a nivel de comunicación e intercambio de mensajes.

2. Desarrollo de Hardware. “Se identifican todos los componentes necesarios para construir el producto y la especificación detallada del hardware, así como la realización del prototipo cubriendo todos los estándares y especificaciones definidos” [15].

   - Seleccionar periféricos de comunicación y buses: CAN Bus.
   - Seleccionar fuente de alimentación, sensores y HMI.
   - Interconexión entre periféricos.

3. Desarrollo de Software. “Se determina el software a utilizar, como el sistema operativo y el código que se ejecuta en el microcontrolador seleccionado”. Según la descripción del modelo definido por INFOTEC [15], la etapa de desarrollo de software involucra los siguientes procesos:

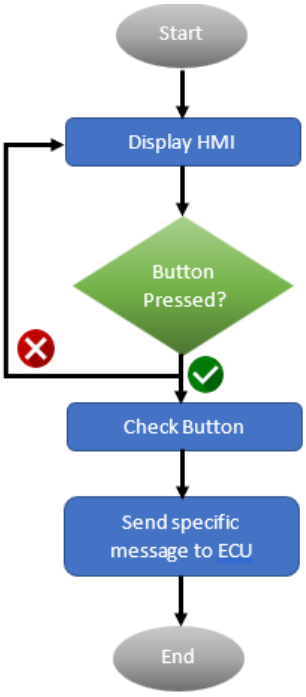
   - Estándares de Software.
     - Sistema Operativo.
       - Soporte para los controladores CAN Bus.
       - Gestión de memoria.
       - Procesamiento de datos y sincronización.
     - Firmware
       - Desarrollo de módulos de envío de datos.

    - Desarrollo de módulos de diagnostico.

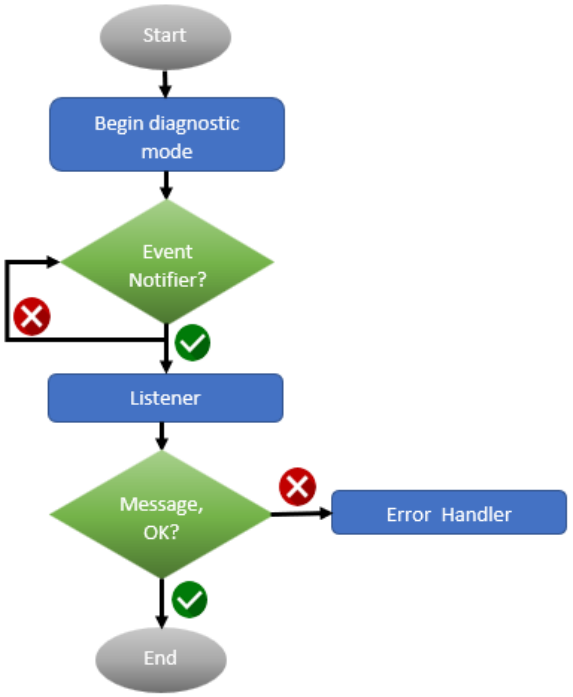
- Middleware
    - Manejo de datos en bus CAN, frecuencia de envío de tramas.

Durante la etapa de diseño de software se revisaron los tipos de telegramas a enviar: estándar, específicos del proyecto y de prueba o diagnóstico. Adicional, el algoritmo de çhecksum"para validar la redundancia de la transmisión y se propusieron los diagramas de flujo de los módulos de software a integrar en primera instancia.

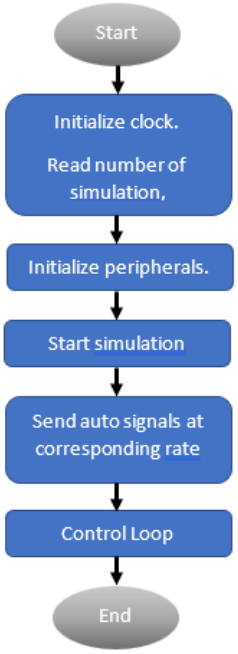
En la Figura 4.3, se pueden observar el desglose de los módulos principales en los que se basa el software de comunicación del prototipo:



(a) Módulo HMI



(b) Módulo Diagnóstico



(c) Módulo Simulación

Figura 4.3: Módulos de Software
Fuente: Elaboración propia.

4.2. Plan de mantenimiento y técnicas de mejora.

Dentro de la ingeniería del software se proporcionan soluciones técnicas que permiten abordar el mantenimiento de manera que su impacto en coste dentro del ciclo de vida sea menor. Las soluciones técnicas pueden ser de tres tipos [20]:

- **Ingeniería inversa:** Análisis de un sistema para identificar sus componentes y las relaciones entre ellos, así como para crear representaciones del sistema en otra forma o en un nivel de abstracción más elevado [20].
- **Reingeniería:** Modificación de un producto software, o de ciertos componentes, usado para el análisis del sistema existente técnicas de ingeniería inversa y, para la etapa de reconstrucción, herramientas de ingeniería directa, de tal manera que se oriente este cambio hacia mayores niveles de facilidad en cuanto a mantenimiento, reutilización, comprensión o evolución [20].
- **Reestructuración del software:** Cambio de representación de un producto software, pero dentro del mismo nivel de abstracción [20].

El objetivo de estas técnicas es proporcionar métodos para reconstruir el software, ya sea reprogramándolo, re-documentándolo, rediseñándolo, o rehaciendo algunas características del producto. La diferencia entre las soluciones descritas radica en cuál es el origen y cuál es el destino de las mismas (producto inicial y/o producto final) [20].

Gráficamente en la Figura 4.4, estas tres soluciones técnicas se enmarcan en el ciclo de vida de la siguiente manera:

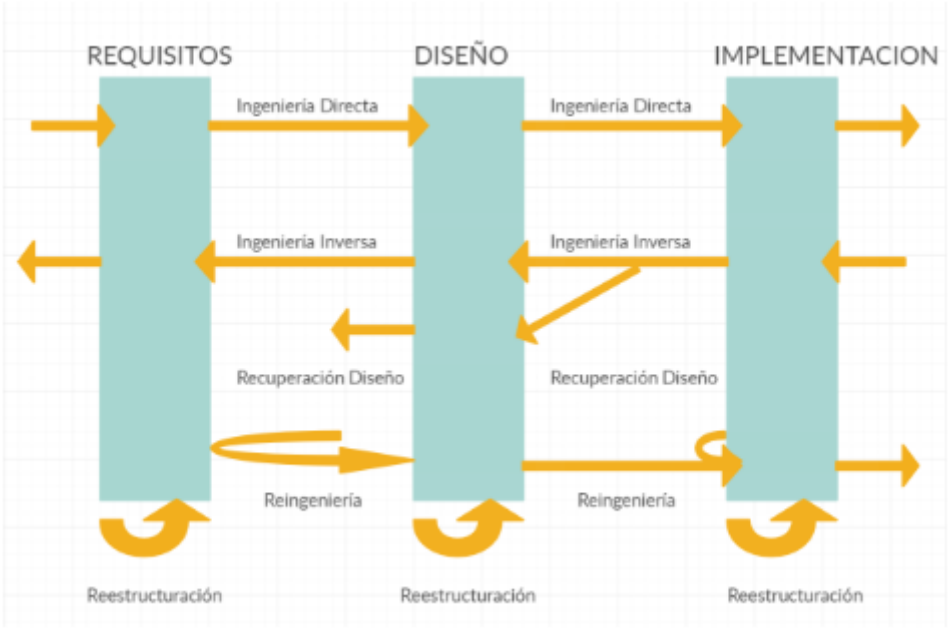


Figura 4.4: Diagrama correspondiente al plan de mantenimiento
Fuente: Elaboración propia.

Plan de mantenimiento.

Los dispositivos embebidos sistema informático compuesto por una combinación de hardware y software diseñado para realizar una función específica [7], con base a la anterior definición las tarjetas electrónicas no tienen garantía y no es posible adaptarlos a las técnicas industriales de mantenimiento predictivo preventivo y correctivo. El HW embebido justifica por su naturaleza que no se haya contemplado un plan de mantenimiento, sin embargo, ha sido considerado actualizaciones de firmware y software para mantener un hardware robusto, stand alone y con las pruebas necesarias para su operación 24/7 en estaciones de pruebas dentro de líneas de producción o laboratorio. La técnica para realizar este proceso de mantenimiento en software embebido se prevé un desarrollo con arquitectura modular para ejercer los mínimos cambios en la estructura del programa original con cada actualización del sistema. La técnica a utilizar será la de reingeniería una vez finalizado la ingeniería directa con el prototipo.

Plan de mejoras.

El objetivo general del proyecto es generar un prototipo funcional para intercambio de mensajes integrado en un sistema embebido que proporcione acceso a señales de control y señales de estado que acepte el protocolo CAN.

Los sistemas de control reprogramables que están en el mercado y con los cuáles se pretende competir ofrecen muchas más funcionalidades que el módulo de comunicación punto a punto. Dentro de las mejoras del prototipo están los módulos adicionales para complementar la funcionalidad desarrollada durante el proceso del posgrado.

Objetivos:

1. Que el módulo sea capaz de generar archivos de descripción maestro y esclavo para simulaciones Hardware In The Loop.
2. Que el dispositivo sea capaz de manipular periféricos de entrada y salida por medio de configuración y reacción por eventos dentro de la secuencia de control.
3. Una interfaz hombre máquina con apartados intuitivos de configuración.
4. Desarrollar un módulo de actualización por firmware a través de una USB.
5. Generar una interfaz remota “Dashboard” para monitoreo de estado y conectividad TCP/IP.

Descripción del plan de acción.

Dentro del plan de acción para las mejoras anteriormente mencionadas se tomará en cuenta el tiempo de desarrollo de cada módulo de software adicional y el hardware en caso de ser requerido. Trabajo inmediato posterior a finalizar el primer prototipo de comunicación punto a punto CAN Bus durante el posgrado será invertir 4 meses para el módulo de configuración de maestro y esclavo, así como lo algoritmos de emulación de condiciones de pruebas HITL.

Es indispensable tener un diseño eléctrico específico para el nuevo modelo de desarrollo de software ya que al intentar integrar periféricos es posible que no se encuentre

una tarjeta que cumpla con las especificaciones necesarias dejando como alternativa el diseño especifico de hardware y software.

4.3. Hardware

La tarjeta STM32F469 (32F469IDISCOVERY) permite a los usuarios desarrollar fácilmente aplicaciones con los MCU de alto rendimiento STM32F469 con núcleo ARM Cortex-M4 y Chrom-ART Accelerator, en la Figura 4.6 se puede apreciar una imagen de la tarjeta. El kit permite una amplia gama de casos de uso aprovechando gráficos premium por su amplia pantalla en la Figura 4.5 , audio, soporte multisensor, pantalla a color WVGA, seguridad, extensión de memoria y características de conectividad [21].

Figura 4.5: Vista superior tarjeta STM32F4 Discovery
Fuente: Hoja de datos de fabricante.STMicroelectronics.STM32F407G-DISC1,2020

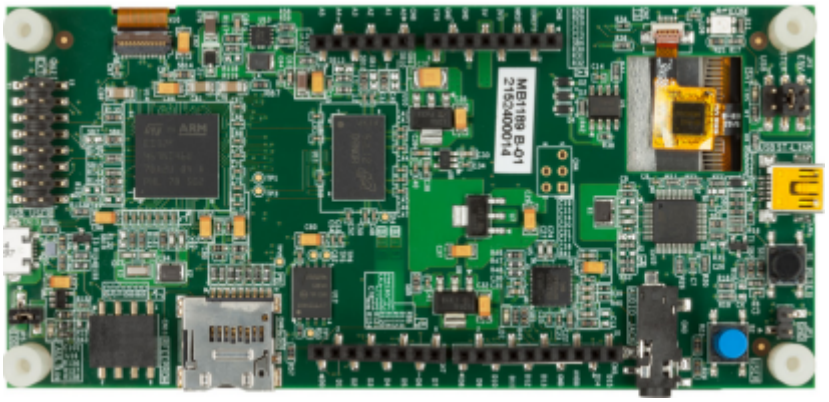


Figura 4.6: Vista inferior tarjeta STM32F4 Discovery
Fuente: Hoja de datos de fabricante.STMicroelectronics.STM32F407G-DISC1,2020

4.4. Software

Para mantener el software en ejecución durante un largo período de tiempo y mantener los sistemas de prueba flexibles y escalables es conveniente mantener una documentación actualizada con el control de cambios y errores observados durante el desarrollo, la implementación y la liberación del sistema, así como las mejoras de rendimiento. A continuación se listan algunas acciones de manteniemento a largo plaza.

1. Acciones correctivas.
    a) Solucionar errores que se generen durante largas periodos de funcionamiento del sistema.
2. Acciones preventivas.
    a) Agregar un informe de errores que permita reconocer y solucionar las fallas que se generen durante la operación del producto.
3. Acciones adaptables.
    a) Atención especial en las tecnologías de hardware, soporte en firmware, actualizaciones en los frames de comunicación.
    b) Análisis de plan de cambos, costos y recurso en tiempo.
    c) Desarrollar un API para generar uns interfaz de conectividad con el dispositivo y poder configurarlo.

Conclusiones

Conclusiones

Resultados

En la industria manufacturera, las líneas de ensamble están compuestas por varios procesos de pre-ensamble y ensamble final. La manufactura avanzada se enfoca en mejorar los procesos de operación, simplificarlos y hacerlos más robustos y eficientes. Por otro lado la manufactura de prueba orienta las estaciones de línea final para implementar bancos de prueba autónomos, instrumentar con tecnología adecuada para cubrir los requerimientos, diseñar la arquitectura de hardware así como la adquisición, tratamiento y procesamiento de señales y no solo se limita al banco de pruebas si no a los datos que se adquieren, interpretación y presentación. Así que el desarrollo del presente trabajo ha sido dirigido hacia el área de ingeniería de pruebas. La metodología implementada y las actividades planificadas durante el desarrollo del trabajo ha sido fundamental para obtener y presentar un módulo de software embebido para emitir mensajes de control y recibir mensajes de diagnostico sobre un Bus CAN. Las especificaciones de prueba señalan los frames embebidos que deben integrar el tráfico de señales. Se simula un ambiente de señales enviadas periódicamente a diferentes frecuencias a través del Bus. El final de línea prueba características de seguridad del usuario y eléctricas del módulo ensamblado como si estuviera integrado en el vehículo y el resto de los subsistemas que lo conforman. La información presentada en el cuadro 4.1 es una fracción de algunos mensajes transmitidos. Propiedad de WRSI de México. Webasto Roof Systems.

Cuadro 4.1: Especificación de simulación de tramas para E-drive y frecuencia.

Fuente. Elaboración propia.

ID	Name	DLD	**Data**								Cycle Time
339	BC F Stat1 AR	8	0	0	7B	0	0	0	0	0	100 ms
20	Ign Veh Stat AR2	8	0	C0	0	0	0	0	0	0	100 ms
339	NM EIS AR	8	0	7	0	0	0	0	0	0	2000 ms
397	Door Adj Rq Dr RL AR	2	0	0							25 ms
39B	Door Adj Rq Dr RR AR	2	0	0							25 ms
122	PN14 Stat AR2	8	87	0	0	0	0	0	0	0	100 ms

Los resultados obtenidos en el diseño de software embebido ha sido por una parte la transmisión de un paquete de señales estables a diferente frecuencia que emulan señales periféricas que se trasmiten al modulo mecatrónico de prueba, ECU.

Uso de una herramienta de software y hardware que permite supervisar el tráfico en el Bus en tiempo real y capturar todos los datos que fluyen hacia y desde el nodo punto a punto. Figura 4.7

Figura 4.7: Analizador CAN BUS, Microchip.
Fuente: Guía de usuario de fabricante. Microchip,2011.

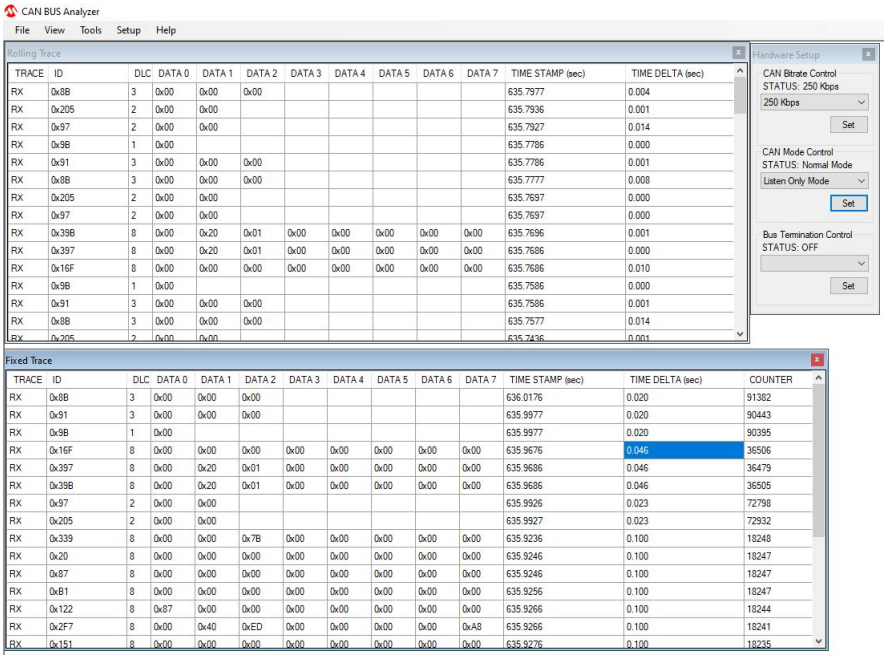
Recibir información de identificación del modulo ECU en la figura 4.8.

RX	0x59C	8	0x06	0x50	0x03	0x00	0x14	0x00	0xC8	0x00
RX	0x59C	8	0x03	0x7F	0x31	0x22	0x14	0x00	0xC8	0x00
RX	0x59C	8	0x06	0x50	0x03	0x00	0x14	0x00	0xC8	0x00
RX	0x59C	8	0x03	0x7F	0x31	0x22	0x14	0x00	0xC8	0x00

Figura 4.8: TX. Mensajes de Estado.
Fuente: Elaboración propia.

En la Figura 4.9 se muestra una captura de una prueba en modo esclavo para escribir mensajes. Canal RX a 250 Kbps, ID en hexadecimal y una frecuencia especifico para

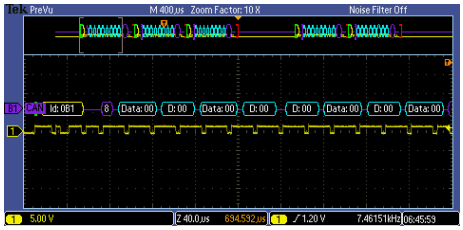
cada tipo de mensaje.



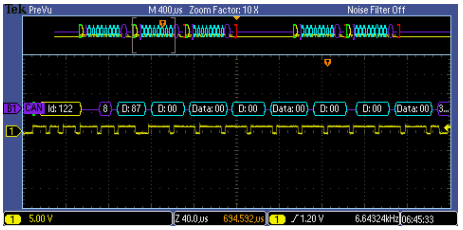
TRACE	ID	DLC	DATA 0	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7	TIME STAMP (sec)	TIME DELTA (sec)
RX	0x8B	3	0x00	0x00	0x00						635.7977	0.004
RX	0x205	2	0x00	0x00							635.7936	0.001
RX	0x97	2	0x00	0x00							635.7927	0.014
RX	0x9B	1	0x00								635.7786	0.000
RX	0x91	3	0x00	0x00	0x00						635.7786	0.001
RX	0x8B	3	0x00	0x00	0x00						635.7777	0.008
RX	0x205	2	0x00	0x00							635.7697	0.000
RX	0x97	2	0x00	0x00							635.7697	0.000
RX	0x39B	8	0x00	0x20	0x01	0x00	0x00	0x00	0x00	0x00	635.7696	0.001
RX	0x397	8	0x00	0x20	0x01	0x00	0x00	0x00	0x00	0x00	635.7686	0.000
RX	0x16F	8	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	635.7686	0.010
RX	0x9B	1	0x00								635.7586	0.000
RX	0x91	3	0x00	0x00	0x00						635.7586	0.001
RX	0x8B	3	0x00	0x00	0x00						635.7577	0.014
RX	0x205	2	0x00	0x00							635.7436	0.001

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	0x8B	3	0x00	0x00	0x00						636.0176	0.020	91382
RX	0x91	3	0x00	0x00	0x00						635.9977	0.020	90443
RX	0x9B	1	0x00								635.9977	0.020	90395
RX	0x16F	8	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	635.9676	0.046	36506
RX	0x397	8	0x00	0x20	0x01	0x00	0x00	0x00	0x00	0x00	635.9686	0.046	36479
RX	0x39B	8	0x00	0x20	0x01	0x00	0x00	0x00	0x00	0x00	635.9686	0.046	36505
RX	0x97	2	0x00	0x00							635.9926	0.023	72798
RX	0x205	2	0x00	0x00							635.9927	0.023	72932
RX	0x339	8	0x00	0x00	0x7B	0x00	0x00	0x00	0x00	0x00	635.9236	0.100	18248
RX	0x20	8	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	635.9246	0.100	18247
RX	0x87	8	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	635.9246	0.100	18247
RX	0xB1	8	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	635.9256	0.100	18247
RX	0x122	8	0x87	0x00	0x00	0x00	0x00	0x00	0x00	0x00	635.9266	0.100	18244
RX	0x2F7	8	0x00	0x40	0xED	0x00	0x00	0x00	0x00	0xA8	635.9266	0.100	18241
RX	0x151	8	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	635.9276	0.100	18235

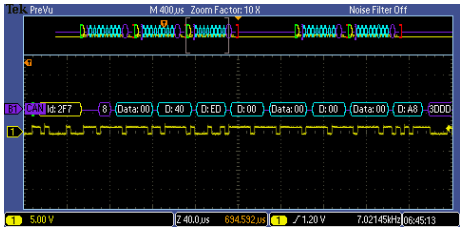
Figura 4.9: RX. Mensajes en el Bus.
Fuente: Elaboración propia.



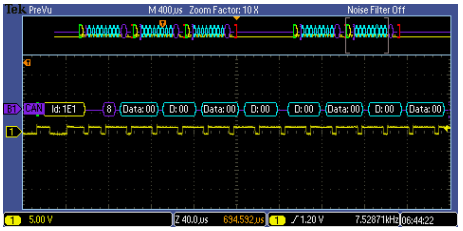
(a) ID hB1, DLC 8, 100 ms



(b) ID h122, DLC 8, 100 ms



(c) ID h2F7, DLC 8, 100 ms



(d) ID h1E1, DLC 8, 100 ms

Figura 4.10: Mensajes en Osciloscopio.
Fuente: Elaboración propia.

En la Figura 4.10 se pueden observar los mensajes enviados a través de señales eléctricas observadas en un osciloscopio TechtroniX.

Durante el desarrollo del prototipo los obstáculos mayores que se experimentó fue la no identificación de los interesados (stakeholders) y su grado de compromiso así como una planeación optimista de las actividades. Es bien sabido que las prioridades en la industria manufacturera varían súbitamente, lo que un día es prioridad numero uno al siguiente puede ser lo contrario, por razones que involucran temas comerciales, cliente, mercado o volúmenes de producción. El tener que administrar la prioridad de las actividades y cumplir con objetivos a corto plazo con los proyectos a mi cargo, redujo el tiempo dedicado al desarrollo del prototipo causando un retraso importante en las tareas programadas. Para mitigar estos hechos fue importante implementar planes de contingencia que permitieran responder de manera efectiva sin afectar actividades laborales y progresar en el desarrollo del prototipo. Dentro de las tareas más destacadas son las de establecer actividades especificas y concretas que pudiera finalizar en poco tiempo y que provocara cierta motivación en lugar de una frustrante tarea incompleta. Fueron útiles herramientas de organización y sobre todo ser riguroso con el tiempo programado en calendario solo para las actividades relacionas al proyecto. Los proyectos de desarrollo tecnológico y los que se relacionan con el ciclo de vida de un

sistema embebido tienden a tener este tipo de influencia en la organización que sin duda alguna ha dejado muchas lecciones aprendidas para futuros proyectos.

Durante la fase de desarrollo se puede señalar afrontar la poca experiencia sobre programación de microcontroladores y desarrollar la habilidad a bajo nivel, sin embargo, el soporte técnico de los asesores y la atención oportuna en proporcionar herramientas y orientación permitieron alcanzar los objetivos. La capa de hardware en la familia de controladores STM simplifica la implementación de protocolos gracias al paquete de drivers HAL incluidos.

Por ultimo, durante la fase de pruebas u validación se necesitaba simular una comunicación redundante en el sistema para evaluar que la transmisión de datos estaba siendo efectuada de manera correcta y a las frecuencias deseadas. Se intentó utilizar otra tarjeta gemela a la del prototipo con un programa espejo pero no funcionó. Se optó por incluir un analizar de señales CAN de microchip que nos permitiera observar las tablas con los datos transmitidos.

Conclusiones finales

A pesar de los avances tecnológicos en términos de sistemas embebidos capaces de asistir las pruebas HIL, existe un área de oportunidad. Las herramientas de software y hardware especializados representan a las compañías una disyuntiva entre la calidad y la seguridad de sus procesos respecto al costo y el mantenimiento de sus procesos. Las limitadas opciones en el área automotriz expone la necesidad de desarrollar tecnología que pueda aportar beneficios similares pero con tecnología menos compleja y sin limitaciones de ejecución, contrato y negocio. Los requerimientos de prueba incrementarán en términos de seguridad del producto y en privacidad para vehículos autónomos de combustión y eléctricos. Los subsistemas mecatrónicos seguirán siendo integrados a vehículos más equipados tecnológicamente y el área de pruebas necesitará herramientas escalables, mantenibles y programables. Después de una aproximación profunda al tema desde una perspectiva de investigación y desarrollo, se puede inferir que es posible y viable desarrollar una herramienta de calidad con un enfoque llave en mano.

Al final de la revisión, se considera preciso el desarrollo de diferentes módulos que complementen y sumen al trabajo presentado y la asignación de recursos para lograr un producto final, para ello es indispensable continuar trabajando para promover la implementación del prototipo en un modulo de pruebas para manufactura, en el que se consiga un desempeño óptimo, tanto en el área de los sistemas Hardware in the loop (HIL) como en los niveles y especialidades operativas, en favor del desarrollo de pruebas en menor tiempo y a menor costo.

Bibliografía

[1] ALL ABOUT CIRCUITS. Introduction to CAN (Controller Area Network). https://www.allaboutcircuits.com/technical-articles/introduction-to-can-controller-area-network/. (Accessed on 07/13/2020).

[2] APTIV, M. I. ¿qué es la prueba de Hardware In The Loop? https://www.aptiv.com/es/tendencias, marzo 2022. (Accessed on 09/10/2022).

[3] BLANCO CURIESES, F. Estudio del bus de comunicacioes CAN. http://openaccess.uoc.edu/webapps/o2/bitstream/10609/88505/8/fblancocuTFM0119memoria.pdf. (Accessed on 07/23/2020).

[4] BUITRAGO, J. Estudio del protocolo de comunicación serial bus CAN y la aplicación en la industria de vehículos de transporte. https://repositorio.utp.edu.co/server/api/core/bitstreams/ad66e6e2-7cdd-4c24-9acc-fdf440fb263a/content, 2017. (Accessed on 07/11/2020).

[5] DE MIGUEL PARAÍSO, M. Desarrollo de herramienta para comunicación con vehículo a través de CAN bus. http://hdl.handle.net/10016/23164, July 2015.

[6] DÜLGER, M. Virtual model for the simulation of the controller area network. *Global Journal of Engineering Science and Research Management* (2019), 32–38.

[7] ELECTRONICS, R. Embedded Systems Explained. Blog. https://reboundeu.com/insights/blog/embedded-systems-explained/. (Accessed on 25/02/2023).

[8] FLEMING, B. Microcontroller units in automobiles [automotive electronics]. *IEEE Vehicular Technology Magazine 6*, 3 (sep 2011), 4–8.

[9] GROUP, V. Vector Group. https://www.vector.com/us/en/. (Accessed on 07/11/2020).

[10] HUYBRECHTS, T., VANOMMESLAEGHE, Y., BLONTROCK, D., VAN BAREL, G., AND HELLINCKX, P. Automatic reverse engineering of CAN bus data using machine learning techniques. In *Advances on P2P, Parallel, Grid, Cloud and Internet Computing*. Springer International Publishing, Nov. 2017, pp. 751–761.

[11] INC., T. H. CAN bus protocol. https://www.typhoon-hil.com/documentation/typhoon-hil-schematic-editor-library/References/can_bus_protocol.html. (Accessed on 07/13/2020).

[12] INDUSTRY, L. Dispositivos de comunicación CAN. https://www.lipowsky.com/lipo/about-us/chronicle.html. (Accessed on 07/11/2020).

[13] IONOS. El modelo en cascada en el desarrollo de software ionos. https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/el-modelo-en-cascada/, 03 2019. (Accessed on 08/29/2020).

[14] JAROSŁAW, J. Y KRZYSZTOF, M. Can bus diagnostics. https://sin.put.poznan.pl/publications/details/n10886, 2014. (Accessed on 07/13/2020).

[15] JUAN CARLOS TÉLLEZ MOSQUEDA, P. A. M. Diagnóstico para la fundamentación de la Maestría en Sistemas Embebidos. https://www.infotec.mx/work/models/Infotec/Publicaciones/Diagnostico-para-undamentacion-Maestria-Sistemas-Embebidos.pdf, Febrero 2014. (Accessed on 25/02/2023).

[16] MARTÍNEZ RIQUEÑA, A. Introducción a CAN bus: Descripción, ejemplos y aplicaciones de tiempo real. http://oa.upm.es/48054/8/TFM_ADRIAN_MARTINEZ_REQUENA.pdf. (Accessed on 07/18/2020).

[17] MITCHELL, B. What Are Network Protocols? https://www.lifewire.com/definition-of-protocol-network-817949. (Accessed on 08/2021).

[18] OF ENCYCLOPAEDIA BRITANNICA, T. E. Protocol. https://www.britannica.com/technology/protocol-computer-science. (Accessed on 07/11/2020).

[19] OF ENGINEERING, R. C., AND TECHNOLOGY. Distributed Embedded Systems. https://www.rcet.org.in/uploads/academics/rohini_85806983944.pdf.

(Accessed on 25/02/2023).

[20] SICILIA, M. Técnicas del mantenimiento del software. https://cnx.org/contents/RunP_m1t@6.1:20y9xGvZ@4/T%C3%A9cnicas-del-Mantenimiento-del-Software, September 2008. (Accessed on 07/23/2020).

[21] STMICROELECTRONICS. Discovery kit with STM32F469NI MCU - STMicroelectronics. https://www.st.com/en/evaluation-tools/32f469idiscovery.html. (Accessed on 07/24/2020).

[22] TEAM, A. Multiplexado automotriz redes can bus - guía práctica electrónica. https://www.autoavance.co/blog-tecnico-automotriz/22-sistema-de-redes-y-multiplexado/. (Accessed on 07/18/2020).