DISEÑO Y CONSTRUCCIÓN DE INTERFACES SECUNDARIAS Y COMUNICACIONES DEL VEHÍCULO ELÉCTRICO DE LA UPB

Juan C. VÉLEZ GIRALDO, Daniel A. ARROYAVE MOLINA

Trabajo de grado para optar al título de Ingeniero Electrónico

Director(es)
José Valentín RESTREPO LAVERDE
MSc. Ingeniero Electrónico

Universidad Pontificia Bolivariana
Escuela de ingeniería
Facultad de ingeniería eléctrica y electrónica
Ingeniería electrónica
Medellín
2013

Dedicatoria

Principalmente dedicamos este trabajo a nuestros padres y hermanos, que por medio de su apoyo incondicional en todo momento durante el desarrollo de nuestra carrera profesional nos inspiraron a ser mejores cada día. Gracias por ayudarnos a cumplir nuestras metas y a realizarnos como profesionales y personas de bien, por sus consejos en los momentos difíciles y por compartir con nosotros los logros obtenidos.

Agradecimiento

Primero agradecerles a nuestros familiares por la oportunidad de poder estudiar y por el apoyo económico que nos brindaron durante la carrera.

A Valentín Restrepo y Pilar Álzate por compartir su tiempo, experiencia y conocimiento con nosotros.

Agradecerles también a Armando Bohórquez, Andrés Emiro y Mauricio Figueroa por su apoyo durante la elaboración de este proyecto.

Agradecerles a todos nuestros amigos y compañeros en especial a Javier Arismendy, que han sido un apoyo incondicional y a los técnicos del laboratorio Jorge, Mauro y Guille.

A todos nuestros más sinceros agradecimientos.

Contenido

Lista de Figuras

Glosario

CAN: *Controller Area Network*. Protocolo de comunicación basado en topología bus para la transición de mensajes en entornos distribuidos.

TTL: *Transistor-Transistor Logic*. Es una tecnología de construcción de circuitos electrónicos digitales. Los componentes fabricados con tecnología TTL las señales de entrada y salida del dispositivo son transistores bipolares.

PCB: *Printed Circuit Board*. Es una superficie constituida por caminos o pistas de material conductor laminadas sobre una base no conductora.

PLC: *Programmable Logic Controller*. Es una computadora utilizada para automatizar procesos electromecánicos.

SEÑAL DE ACK: en inglés *acknowledgement*. En las comunicaciones es un mensaje que el destino envía al origen para confirmar la recepción de una información.

SEÑAL DE CRC: comprobación de redundancia cíclica es un código de detección de errores usado frecuentemente en redes digitales y en dispositivos de almacenamiento para detectar cambios accidentales en los datos.

LCD: *Liquid Cristal Display*. Tecnología utilizada en monitores de computadoras, televisores, cámaras digitales, etc. que permite una pantalla más delgada y plana, además de una excelente definición.

TFT: *Thin-film Transistor*. Es un tipo especial de transistor de efecto campo que se fabrica depositando finas películas de un semiconductor activo así como una capa de material dieléctrico y contactos metálicos sobre un sustrato de soporte.

TRANSCEIVER: es un dispositivo que cuenta con un transmisor y un receptor que comparten parte de la circuitería o se encuentran dentro de la misma caja.

MAC: *Media Access Control*. Se conoce también como dirección física, y es única para cada dispositivo.

NODO: en electricidad, es un punto de conexión entre dos o más elementos de un circuito.

BUS: es una topología que se caracteriza por tener un único canal de comunicaciones al cual se conectan los diferentes dispositivos permitiendo compartir el mismo canal para comunicarse entre sí.

LATENCIA: Es la suma de retardos temporales dentro de una red informática, donde dicho retardo es producido por la demora en la propagación y transmisión de paquetes.

Resumen

En este trabajo se mostrara la realización de un tablero para vehículo eléctrico. También se hablara de las diferentes variables a medir y los sensores a utilizar. Además, de cómo se hará la comunicación entre ellos y el procesamiento de las señales obtenidas, también de cómo serán visualizadas estas.

Se documentara los diferentes procesos y desarrollos llevados a cabo como lo son por ejemplo los circuitos que se utilizaran tanto para cada sensor como para el microprocesador central, además de los circuitos de la pantalla donde se implementar al desarrollo del tablero.

Se hablara de los diferentes formas o protocolos de comunicación existentes para este tipo de aplicaciones y se hará énfasis en el protocolo de comunicación CAN (Controller Area Network), que será el que se utilizara para el desarrollo del proyecto.

Palabras clave: Conversor A/D, rata de baudios, circuitos, protocolos de comunicación.

Abstract

This paper shows the performance of a board for electric vehicles. Also spoke of the different variables to be measured and the sensors used. Also, how will the communication between them and processing the signals obtained, also how these will be displayed.

Different processes are documented and developments made such as for example the circuits both to be used for each sensor to the central microprocessor, and circuits which implement the display board development.

They talk about the different forms or existing communication protocols for such applications and will focus on the communication protocol CAN (Controller Area Network), which will be the one used for the development of the project.

Keywords: A/D converters, baud rates, circuits, communication protocols.

INTRODUCCIÓN

Con el desarrollo en los últimos años de la tecnología y su incursión en el mundo automovilístico, vemos el crecimiento del uso de nuevos implementos y periféricos en la evolución de los automóviles, con sistemas inteligentes de frenado, dirección asistida y otras más de confort como el GPS y sistemas inteligentes que convierten el hecho de manejar un automóvil en un mundo de comodidades para el usuario. Además del gran auge que se está tomando las grandes ciudades en el mundo con el transporte eléctrico, por su gran favorabilidad y cuidado con el medio ambiente, se dio la idea de poder realizar e incursionar en un estudio sobre este nuevo mundo de la movilidad eléctrica (BOSCH, 2002). En este caso el diseño y desarrollo de un tablero para un vehículo eléctrico, ya que este es fundamental para el vehículo, dado que por medio de este el conductor tiene la información en tiempo real de la mayoría de variables y estados en los que se encuentra el vehículo. El propósito de este proyecto es implementar un tablero para un vehículo eléctrico de la UPB, por medio de una analogía con el tablero de un vehículo de combustión interna se puede llegar a tener un prototipo.

La característica principal de este proyecto es poder tomar las variables relevantes del sistema para así, realizar un proceso de acondicionamiento y poder entregarlas al conductor del vehículo eléctrico como es debido. Las variables a mostrar son velocidad del vehículo, temperatura del motor y la carga de las baterías que sería análoga a la cantidad de combustible en un vehículo de combustión interna. De igual forma, variables que aunque no son mostradas en el tablero son de importancia para el control y seguridad, como la presión de las bombas del sistema de refrigeración y la corriente de las baterías para poder estudiar también el consumo y rendimiento de estas, éstas variables son distribuidas a otros sistemas para su utilización. Además de la incorporación de diferentes alarmas y señales para la seguridad del conductor y sus pasajeros y para la autonomía del vehículo.

Es necesario mencionar que este proyecto hace parte de un proyecto aún más grande que está realizando la facultad de eléctrica y electrónica de la Universidad Pontificia Bolivariana, que trata sobre la factibilidad de conversión de un vehículo de combustión interna a un vehículo eléctrico.

Durante el desarrollo del proyecto se describirá los términos y tecnologías importantes para el desarrollo de este como lo son los sensores para la toma de las variables a medir, para la comunicación entre éstos, la computadora principal y la visualización de estas (Escamilla Arroyo, 2005)por parte del conductor, esto estará descrito en el primer capítulo.

También se explicará el desarrollo de todos los sistemas que irán incorporados para cumplir dichas funciones dentro del vehículo, tanto en hardware como en software, esto se verá en el capítulo 2 y 3.

En el capítulo 4 se mostrará los resultados obtenidos durante el desarrollo de este prototipo, las principales dificultades y las soluciones implementadas.

OBJETIVOS

Objetivo general

Diseñar un tablero electrónico para la indicación de variables importantes para la conducción del vehículo eléctrico.

Objetivos específicos

- Diseño de un sistema de comunicación robusta por medio del protocolo de comunicación CAN (Controller Area Network)
- Seleccionar el conjunto de variable que el tablero debe mostrar para el conductor del vehículo
- Construir un tablero prototipo con el sistema de comunicación y los sensores que se requieran
- Entregar un diseño óptimo para la implementación de este en el proyecto en escala de prototipo de producto terminado

1. MARCO TEORICO

1.1. Controller Area Network (CAN)

Es un protocolo serial basado en topología bus que conecta diferentes dispositivos como sensores, actuadores y varios sistemas de control por medio de un bus y fue desarrollado por la firma alemana Bosch a mediados de los 80. Esto debido a la expansión de dispositivos electrónicos en los vehículos que llevó a la necesidad de buscar una alternativa diferente a la que se tenía, ya que esta utilizaba conexión punto a punto y el cableado en los vehículos se iba siendo más extenso, complicado y costoso. (Ran, Junfeng, & Haiying, 2010)

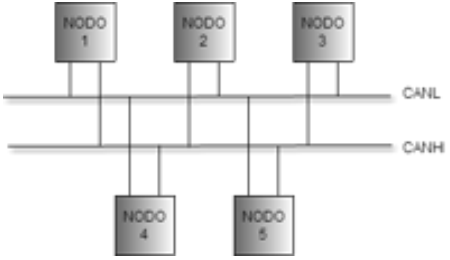


Figura 1. Topología del bus CAN.

A estos dispositivos, se les conoce como nodos. Ver Figura 1. Esto implica mayor control y registro de los instrumentos conectados a la red y una reducción de costos de desarrollo, para el caso en específico que se está tratando en la industria automotriz. Aunque en un principio este protocolo fue desarrollado por las necesidades que se presentaban en la industria automotriz, hoy en día esta es usada en muchas aplicaciones industriales debido a su comodidad, eficiencia y su baja tasa de errores. (Marino & Schmalzel, 2007)

CAN es un protocolo orientado a mensajes dándole las siguientes características (Escamilla Arroyo, 2005):

- Prioridad de mensaje: esto permite a cada nodo perteneciente a la red identificar qué tipo de mensaje se está transmitiendo por la red y poder filtrarlo para saber si actúa o no.
- Garantía en tiempos de latencia: la longitud del mensaje transmitido por CAN tiene como máximo 8 bytes de longitud garantizando así una baja latencia.
- Carrier Sense Multiple Access/ Collision Detection with Arbitration on Message Priority (CSMA/CD with AMP): Con esto se tiene un control sobre los mensajes trasmitidos por la red evitando colisiones por medio de prioridades que son asignadas a cada uno de los mensajes. Con esto se busca dar una prioridad a los mensajes que se tendrá en cuenta solo en el caso de que dos nodos traten de enviar un mensaje al mismo tiempo.

- Configuración flexible: gracias a que el protocolo es orientado a mensajes, es decir que se asigna un ID único, se puede seguir agregando nodos en un futuro sin necesidad de tener que volver a configurar el software o hardware.

En general este protocolo ofrece muchas ventajas a la hora de ser implementado, además cuenta con un excelente mecanismo de detección de errores y una alta inmunidad a la interferencia electromagnética. (Escamilla Arroyo, 2005)

1.2. Capas del bus CAN

Capa física: Como se mencionó anteriormente el protocolo trabaja con una topología de bus, este consiste en un par de cables trenzado que en sus extremidades son colocadas unas resistencias que representan la impedancia característica de la línea, la mayor distancia que se puede colocar es de un kilómetro, aunque si se desea agregar más longitud a la línea se pueden usar repetidores. Los nodos conectados a esta línea deben de interpretar dos niveles (Calva Cuenca, 2010):

- Dominante (0)
- Recesivo (1)

En la Tabla 1 podemos observar la relación entre longitud del bus y la velocidad de transmisión.

La capa física es la encargada de transferir los mensajes entre cada uno de los nodos que componen la red. Se definen aspectos como los niveles de la señal, la codificación y la sincronización.

Tabla 1. Relación entre longitud y velocidad de transferencia.

Longitud del bus (m)	*Velocidad de transmisión(* Kbits/s*)*
40	1000
300	500
600	100
1000	50

Los valores que se muestran en la Tabla 2, son los correspondientes a los rangos de voltaje definidos para los bits recesivos y dominantes respectivamente para los canales CANH y CANL de la red.

Tabla 2. Valores nominales de los voltajes en el bus CAN.

Parámetro	Dominante	Recesivo
VCAN_L	2.5	3.5
VCAN_H	2.5	1.5
Vdiff	0	2

Hay que tener en cuenta que "el mínimo valor de VCAN_H es determinado por el mínimo valor de VCAN_L más el valor de Vdiff. El máximo valor de VCAN_L es determinado por el máximo valor de VCAN_H menos el mínimo valor de Vdiff. Vdiff es la diferencia de voltaje entre VCAN_H y VCAN_L." (ISO 11898-1, 2003)

Capa de enlace:

La capa de enlace o capa de transferencia representa el kernel del protocolo CAN, esta presenta los mensajes recibidos por medio del bus a los filtros previamente configurados y también es la encargada de recibir los mensajes que serán trasmitidos por el bus. (Bosch, 1991)

Esta capa se puede dividir en un control de enlace lógico (LLC) y en un control de acceso al medio (MAC). En el LLC encontramos (Ekiz, Kutlu, & Powner, 1996):

- Filtros de aceptación.
- Notificaciones de sobrecarga.
- Gestión de recuperación.

Mientras que en la MAC encontramos:

- Encapsulación de datos.
- Des encapsulación de datos.
- Codificación del frame.
- Gestión del acceso al medio.
- Detección de error.
- Señalización de error.
- Acknowledgment
- Serialización / Descerialización. (Provencher, 2012)

1.3. Formato de las tramas

El mensaje del protocolo CAN está definido en dos formatos:

- Formato estándar
- Formato extendido

Estos dos formatos en su composición son parecidos la mayor diferencia entre ellos es el tamaño del ID, ya que en el formato estándar es de 11 bits, mientras que en el formato extendido es de 29 bits.

Se pueden encontrar diferentes tipos de tramas de mensajes que puede permitir ser enviada por una red CAN. Los tipos de tramas más utilizados en el protocolo CAN son (Fresno, 2004):

Trama de datos: es la más utilizada comúnmente, esta es la que utiliza un nodo para colocar información en el bus, y tiene un tamaño de dato de hasta 8 bytes.

En el formato que definió Bosch cuando creo el protocolo, dijo que el mensaje se descompone en diferentes tamaños, en los cuales se definen algunos conceptos para que la comunicación sea efectiva. Estos mensajes son introducidos en el bus con una cadencia entre los 7 y los 20 ms dependiendo de la velocidad del área y del módulo que lo introduce (fresno, 2004).

Figura 2. Formato trama de datos estándar.

Como se observa en la Figura 2, esta trama está conformada por:

- Campo de inicio del mensaje: El mensaje inicia con un bit dominante.
- Campo arbitraje: en este se encuentran los 11 bits del identificador donde también va la prioridad del mensaje, también se encuentra el bit RTR que indica si el mensaje lleva datos (RTR = 0) o no (RTR = 1).
- Campo de control: en este se da información acerca de cuáles son las características del campo de datos. Cuando IDE = 0 indica que es una trama estándar, si por el contrario es IDE = 1 indica que es extendida, el bit RB0 siempre está en estado recesivo y los últimos 4 bits que componen el DLC indican el número de bytes contenidos en los datos. Ver Tabla 3.

Tabla 3. Formato del campo DLC según la longitud del dato.

Hora	T (°C)	ω (RPM)	Q (L/min)	I (A)
1:00	22.1	1345	1.34	34.2
2:15	22.0	1200	2.45	12.0
2:18	22.3	1100	2.53	11.1
5:34	22.1	3450	0.99	10.3
8:07	22.2	0	0.00	0.1

- Campo de datos: en este campo se manda la información del mensaje y tiene un tamaño máximo de 64 bits.
- Campo de aseguramiento: Código de redundancia cíclica que genera el transmisor por la división módulo 2 de todos los bits precedentes del mensaje, incluyendo los de relleno si existen, por el polinomio generador: X15 + X14 + X8 + X7 + X6 + X5 + X4 + X3 + X1 + 1, el resto de esta división es el código CRC transmitido. Los receptores comprueban este código. En los primeros 15 bits son utilizados para la detección de errores, mientras que el último bit siempre es recesivo.
- Campo de confirmación: estos son siempre transmitidos como recesivos. Todos los nodos que reciben el mismo CRC modifican el primer bit del campo por un dominante, de forma que el transmisor reconoce que el mensaje ha sido recibido correctamente por alguno de

estos. Si por el contrario ninguno cambia este bit, el transmisor lo interpreta como un error.

- Campo de final de mensaje: este indica el fin de la trama poniendo 7 bits recesivos. El protocolo CAN coloca cada 5 bits del mismo valor un bit contrario a este (es decir si hay cinco "0" seguidos el coloca un "1" y viceversa) para así no perder la sincronización, estos bits son llamados **bit stuffing**.

- Espacio entre tramas: este indica el tiempo entre tramas.

Trama remota: Esta trama a diferencia de la anterior no lleva datos, simplemente el identificador es el del nodo al que se le solicita la información, el RTR es recesivo. El nodo que contenga el ID especifico deberá responder con una trama de datos, la trama remota conlleva dato. (Fresno, 2004)

Trama de error: esta es enviada por un nodo cuando este detecta un error en la red para informar de este a los demás nodos. (Fresno, 2004)

- Indicador de error: si un nodo estando en estado de error activo detecta un error en el bus interrumpe la comunicación y genera un indicador de error activo que consiste en 6 bits dominantes sucesivos, esta no cumple con la regla del **bit stuffing** y también genera que otros nodos también manden este tipo de trama y se espera el delimitador de error para reiniciar la transmisión. Por el contrario si el nodo está en estado de error pasivo y este detecta un error, transmite un indicador de error pasivo que consta de 6 bits recesivos seguidos seguido por el delimitador de error, por lo tanto la trama consta de 14 bits recesivos.

- Delimitador de error: consta de 8 bits recesivos consecutivos y permite a los nodos reiniciar la comunicación.

Trama de sobrecarga: esta es generada por un nodo para solicitar un tiempo de más antes del próximo envió de información. Esta trama tiene el mismo formato de una trama de error activo, con la diferencia de que esta solo puede ser transmitida en el espacio entre tramas. Esta trama consta de dos partes, el indicador de sobrecarga y el delimitador. El indicador de sobrecarga tiene una longitud de 6 bits dominantes y el delimitador consta de una longitud de 8 bits.

Esta trama es generada por un módulo cuando por alguna razón, este no puede recibir algún mensaje. De esta forma se retrasa él envió de datos. Un módulo tiene la posibilidad de generar hasta 2 mensajes de sobrecarga. (Fresno, 2004)

Intertrama: es un espacio entre tramas (bien sean remotas o de datos) para permitir que los demás nodos si tienen que hacer algún procesamiento interno lo puedan hacer. Este es el que se presenta entre dos tramas sucesivas, este debe contar con al menos 3 bits recesivos. Al haber transcurrido esta secuencia y el modulo que transmite se encontraba en un error activo, se reinicia la transición. Si por el contrario este se encuentra en un error pasivo,

este deberá esperar una secuencia de 8 bits recesivos para poder reiniciar la transmisión. (Fresno, 2004)

1.4. Tipos de errores

Al recibir una trama de error es importante poder saber cuáles son los errores más comunes para así ahorrar tiempo en la búsqueda y solución de estos. Los errores que se presentan en una red CAN son:

- Error de CRC: los códigos de redundancia ciclada generados tanto por el transmisor como por el receptor no coinciden. El nodo receptor descarta la trama y trasmite una trama error.
- Error de ACK: detectado por el transmisor. Este indica que ningún nodo recibió el mensaje. No se genera trama de error.
- Error de forma: se produce si se detecta un bit dominante en los siguientes campos:
    - Delimitador de CRC.
    - Delimitador de ACK.
    - Fin de trama.
- Error de bit: el bit transmitido es diferente al monitoreado. Se transmite una trama de error y se retransmite la trama. Solo lo hace el nodo transmisor.
- Error de stuffing: se detectan 6 bits consecutivos de igual polaridad entre el comienzo y el delimitador CRC. Se envía una trama de error.

1.5. Bit stuffing

El bit stuffing es un método de codificación para la resincronización cuando se usa la representación de NRZ (Non Return to Zero). Esta técnica es muy usada en telecomunicaciones, y consiste en insertar un “0” cuando en la trama de datos existen un bloque de transmisión conformado por cinco “1” seguidos, también para el caso contrario cuando el bloque está conformado por una seguidilla de cinco “0” seguidos se agrega un “1”. Esto sirve para no perder la sincronización en el bus. (Navet)

1.6. Medición y visualización

Ya se mencionó el modo de comunicación entre cada una de las interfaces que componen el sistema, pero no se descrito cuáles son estas y como están conformadas.

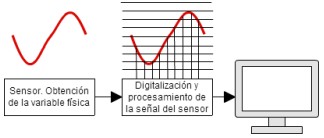


Figura 3. Esquema para la obtención y visualización de las variables a medir.

Estas interfaces son las encargadas de la recolección de la información que es recolectada por medio de los sensores que son colocados en el vehículo, con el fin de realizar una realimentación al usuario del comportamiento de cada uno de los sistemas que componen un vehículo, para el caso un vehículo eléctrico. Es importante conocer este comportamiento de sistemas como el motor y su controlador, las baterías y todos los demás sistemas complementarios que garantizan un correcto funcionamiento del vehículo eléctrico.

Este modelo para la obtención y posterior visualización de los datos consta de tres grandes partes que se observan en la Figura 3: la obtención de las señales análogas por medio de los sensores, la digitalización y procesamiento de las señales obtenidas de los sensores y por último el dispositivo de visualización de estas.

Variables físicas a medir y sensores: durante el desarrollo histórico y la evolución de los automóviles, ha ido creciendo la importancia de la utilización de diferentes sensores para la medición de los diferentes sistemas que comprenden un vehículo.

Esta red de sensores puede tomarse como una analogía a los órganos sensoriales de los seres vivos, ya que gracias a estos podemos tener una interpretación del comportamiento interno del vehículo. Estas señales han llegado a tener gran importancia en las diferentes funciones del sistema como pueden ser las de mando, seguridad, confort, etc.

Para empezar se define el concepto de sensor. Este término se refiere a un dispositivo que produce una señal relacionada a la que se está midiendo. En la mayoría de casos, los sensores toman una medida de una variable física o química y la transforman en una variable eléctrica, bien sea voltaje, corriente o resistencia. Algunos de los términos que se emplean para el funcionamiento del sensor y que deben ser tenidos en cuenta a la hora de realizar cualquier desarrollo con estos son: rango y margen. El primero indica los valores máximos y mínimos en que puede variar la variable a medir, el segundo es la resta entre esos valores. También tenemos el error. Este indica la diferencia que puede existir entre el valor verdadero y el valor que obtenemos de la medida. (Bolton, 2001)

El desarrollo de sistemas de medición como elementos periféricos de un vehículo, constituyen las interfaces entre el vehículo y los diferentes sistemas que lo componen. Por ejemplo con sistemas como el frenado, la transmisión, conducción, navegación, unidad de control, entre otras. (BOSCH, 2002)

Los sensores que son utilizados en los automóviles pueden clasificarse en tres grandes categorías:

- Funcionalidad: su aplicación dentro del automóvil está dada a la regulación y mando.
- Seguridad: están divididos en dos partes, la primera. Destinados a la protección del vehículo en caso de robo. La segunda. Protección en caso de alguna falla interna del vehículo.

- Vigilancia: estos son con los que tiene contacto el usuario y la mayoría son visualizados en el tablero del vehículo.

Al respecto de los sensores que son usados en las aplicaciones de la industria automotriz, se deben tener en cuantas ciertas exigencias. Estos deben ser de alta fiabilidad, bajo costo de fabricación, duras condiciones de funcionamiento, compactibilidad y alta precisión. (BOSCH, 2002)

Algunos de los sensores que podemos encontrar en un vehículo son: (BOSCH, 2002)

- Sensor de presión
- Sensor de masa de aire
- Sensor de velocidad de rotación
- Transmisor de posición del pedal
- Sensor de par
- Sensor de aceleración
- Sensor de nivel de combustible
- Sensor de temperatura

Ahora, el enfoque son los principales sensores que se deben utilizar en un vehículo eléctrico, para ello se debe tener en cuenta cuales son las variable de mayor importancia, y así poder escoger los sensores correctos.

Los sensores que se tienen para un vehículo eléctrico son:

- Sensor de temperatura
- Sensor de presión
- Sensor de voltaje
- Sensor de corriente
- Sensor de velocidad

Hay que tener en cuenta que estos sensores son para el funcionamiento básico del vehículo eléctrico.

En un vehículo se pueden colocar los sensores que se quieran o se crean necesarios según las necesidades del desarrollo.

Conversión análogo – digital y procesamiento: Como es bien sabido al trabajar con microprocesadores, toda la información que estos manejan es digital, entonces al momento de interactuar con medidas tomadas de variables físicas que son análogas se llega a la necesidad de realizar una interfaz entre el mundo análogo con el digital. Esta transformación de lo análogo a lo digital es realizada por un sistema llamado conversor análogo – digital (ADC por sus sigla en inglés).

Para realizar esta conversión se realiza mediante dos procesos, el primero es la cuantificación, este proceso se encarga de muestrear la entrada análoga y le asigna un valor o estado, este valor o estado va a depender de la cantidad de bits que posea en ADC. Es decir, si el ADC con el que se trabaja es de 12 bits va a tener 4096

estados. Entre más bits de resolución tenga el conversor, más aproximada será la conversión a la señal análoga original. (Rosa, sin año).

El segundo proceso es el muestreo o *sampling*, esto es, tomar valores discretos de la señal en instantes de tiempo igualmente espaciados. Para garantizar una conversión correcta es necesario tener una frecuencia de muestreo adecuada, por ello la frecuencia de muestreo sea como mínimo el doble que la frecuencia de la señal a muestrear, como lo establece el teorema de Nyquist. (Huiercán & Ignacio, si año)

1.7. Tecnología TFT – LCD touch

TFT (*Thin Film Transistor*) y LCD (*Liquid Crystal Display*), esta tecnología es una variante de la LCD, ya que esta implementa la tecnología TFT para mejorar la calidad de la imagen. El TFT es un tipo de transistor, fabricado por medio de unas películas finas de un material semiconductor, material dieléctrico y unos contactos sobre un sustrato de soporte. El semiconductor más usado para la fabricación de estos es el silicio, además de este también encontramos semiconductores como por ejemplo el seleniuro de cadmio y oxido de zinc. También existen unos transistores fabricados por medio de materiales orgánicos. Al integrar los transistores en los paneles LCD, lo que se busca es la reducción de la diafonía entre los pixeles y mejorar la estabilidad de la imagen. Cada pixel que conforma la pantalla tiene un transistor integrado. (Wikipedia, 2013)

Una pantalla táctil, tiene la característica que mediante un toque en algún punto de su superficie, esta reacciona como una entrada de datos mediante aquel toque, con el cual se solicita alguna información o se desea realizar cualquier otra actividad con la pantalla. Para el desarrollo de estas pantallas existen diferentes tipos de tecnologías las dos más utilizadas y populares son:

Resistiva: La construcción de una pantalla táctil resistiva es por medio de una sucesión de capas con una pequeña separación. Al realizar algún toque en la superficie de la pantalla ósea, en la superficie de la capa exterior, se produce un contacto entre las capas en un punto. Esto lo que hace es producir un cambio en la corriente eléctrica que permite saber en dónde fue el punto de contacto. Estas pantallas son las más usadas porque tienen la ventaja de no ser afectadas por el polvo y son muy exactas, pero tiene la desventaja de que al utilizar varias capas para su construcción son un poco más opacas que las demás.

Capacitiva: Esta pantalla está cubierta con una material compuesto por óxido de indio y estaño para la conducción de la corriente eléctrica. También está compuesto por un sensor que detecta los cambios de inductancia. Al realizar el toque en la pantalla se realiza un cambio de capacitancia con lo cual se envía la información al procesador y este calcula el punto de contacto. A diferencia de las resistivas, estas tienen que ser tocadas por un dispositivo conductivo. Las pantallas capacitivas tienen una alta claridad, pero son más costosas que las resistivas. (cristobaldominguez, sin año)

2. DESARROLLO DEL HARDWARE

En el diseño y posterior construcción de las interfaces secundarias y comunicaciones del vehículo eléctrico de la UPB, se utilizarán módulos en cada uno de los sensores que se tendrán en el vehículo eléctrico. Estos módulos están diseñados para que se intercomuniquen por medio de una red CAN y cada uno de estos está conformado por un controlador CAN y un *transceiver* CAN. Durante el desarrollo del capítulo se explicará cómo fue el desarrollo tanto de hardware como de software de cada uno de los módulos pertenecientes a los sensores y de la tarjeta principal que es la encargada de obtener los datos enviados por cada uno de los módulos y mostrarlos al usuario.

2.1. Desarrollo de los nodos

En el desarrollo en el hardware de los nodos, se tiene un microprocesador, un controlador CAN y un *transceiver*, como se observa en la Figura 4, sin importar el fabricante, tipo de microprocesador o tipo de transceiver, esta configuración es estándar y por lo tanto se pueden configurar diferentes familias de estos componentes y deberán comunicarse perfectamente. En esta configuración también se puede dar que el controlador CAN ya este incorporado dentro del microprocesador que se va a utilizar. Para el caso de este desarrollo se eligieron microprocesadores que ya contienen el controlador CAN. En el desarrollo del hardware se utilizan dos familias distintas de microprocesadores y *transceiver*, para un solo nodo se diseñó un hardware con un microprocesador de Freescale de la familia Coldfire v1 y un *transceiver* Micrichip, para los demás nodos se utilizó un microprocesador de Microchip de la familia 18x y un *transceiver* también Microchip. Ambos microprocesadores tiene la característica de tener incluido el controlador CAN.

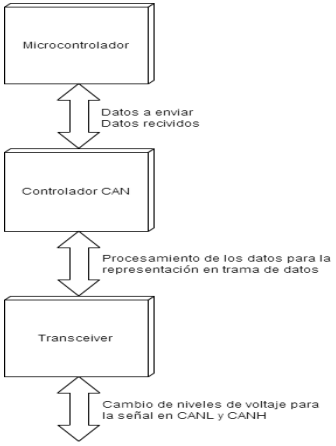


Figura 4. Componentes de un nodo CAN.

2.2. Nodo principal

Este nodo es desarrollado con el fin del manejo de los datos que se recepcionan de los demás nodos, los cuales posteriormente serán enviados a un módulo donde serán presentados

gráficamente al usuario. S dice que este nodo es la transición entre la obtención de las variables y la lectura por parte del usuario. Aunque se sabe que un bus CAN es multimaestro, este nodo se puede considerar el nodo central del desarrollo, ya que a este es el encargado del manejo y distribución de los datos que son obtenidos por los demás nodos.

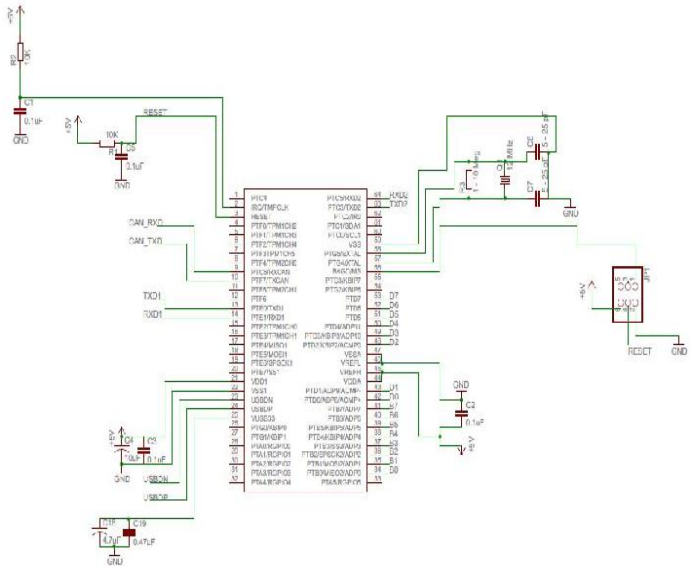


Figura 5. Conexiones básicas micro controlador MCF51JM128VHL.

Este módulo fue desarrollado con un microprocesador MCF51JM128VHL de Freescale de la familia Coldfire V1. Las conexiones para los diferentes periféricos que se usan en este y para su correcto funcionamiento se pueden ver en la Figura 5. Éste ya tiene integrado el controlador CAN y a continuación se nombran las características que ofrece este módulo CAN del MCF51JM128VHL (Freescale, 2009)

- CAN en protocolo 2.0 A/B.
    - Manejo de tramas estándar y extendidas.
    - Longitud de datos desde 0 hasta 8 bytes.
    - Velocidad de hasta 1Mbps.
    - Soporte de tramas remotas.
- Dos filtros de máscara de 32 bits (4 de 16 bits u 8 de 8 bits), de identificación flexible.
- Soporta loopback para auto chequeo.
- Modo de programación de solo escucha, para monitoreo del bus CAN.
- Fuente de reloj del CAN programable.
- Registros para la inicialización global del periférico CAN.

Para el correcto funcionamiento del periférico CAN de este nodo, se implementó el diseño del circuito que se ve en la Figura 6.

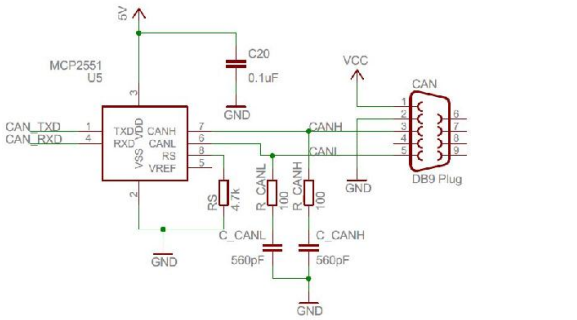


Figura 6. Conexiones para el funcionamiento del periférico CAN en el micro controlador MCF51JM128VHL.

Como se observa en la Figura 6, la conexión al bus del nodo se realiza por medio un dispositivo llamado *transceiver*, este es el MCP2551 del fabricante Microchip. El MCP2551 es un dispositivo de alta velocidad CAN y es el encargado de la interface entre el controlador CAN y el bus. También está protegido contra corto circuitos o transientes que se puedan presentar en el bus.

El MCP2551 ofrece tres modos de operación: *high speed*, *slope control y standby*. Estos modos de operación son configurados por medio de la terminal Rs del dispositivo (Microchip, 2010)

El modo en que se está operando en este caso en el de *slope control*, ya que en el pin de RS se está colocando una resistencia a tierra. Tanto el slope control como el *high speed* tienen la ventaja de reducir las EMI limitando las subidas y bajadas de los canales CAN_L y CAN_H. Pero en el modo de slope control, el *slew rate* (SR) es controlado por medio de la resistencia que es colocada en el diseño que para el caso es de 4.7KΩ, aunque en este caso no tenga relevancia ya que como observamos en la Figura 7 el SR se empieza a ver modificado a valores muchas más grandes de esta resistencia.

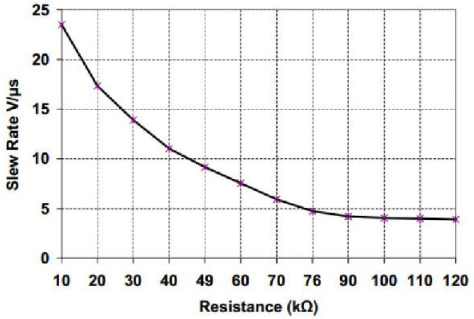


Figura 7. Slew rate vs valor resistencia de slope control. (Microchip, 2010)

Algunas de las características que este dispositivo ofrece son:

- Soporta una velocidad de hasta 1Mbps.
- Cumple con los requerimientos de la capa fisica del estandar ISO 11898.

- Pueden estar conectados hasta 112 nodos en el mismo bus.
- Alta inmunidad ante ruido presente en el bus diferencial.

En el desarrollo del nodo se tiene en cuenta la inclusión de dos terminales de comunicación serial, esto con el fin de poder transmitir datos a módulos como por ejemplo el implementado por medio de un PLC (*Programmable Logic Controller*) (Pérez, 2013) que es el encargado del control y protección del motor eléctrico del vehículo. También se debe comunicar con el módulo de visualización del cual se hablara más adelante, con el fin de poder mostrar en tiempo real el comportamiento de las variables del vehículo al conductor. Como se observa en la Figura 8, se usó el chip MAX232 del fabricante MAXIM, este es el encargado de convertir las señales de la comunicación serial a señales compatibles con los niveles TTL. También se observa la implementación del hardware necesario para su correcto funcionamiento (Freescale, 2009).

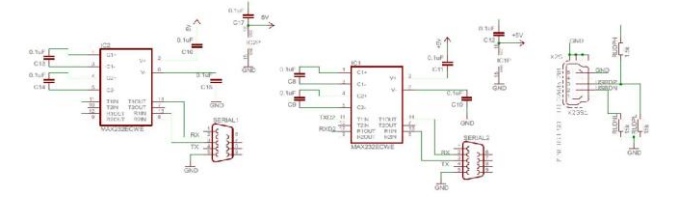


Figura 8. Hardware implementado para la comunicación serial y USB.

Se utilizó la comunicación serial para este parte del desarrollo debido a que era la mejor opción, ya que la otra alternativa que ofrece el PLC es Ethernet, esto debido a que los módulos mencionados anteriormente, con los cuales este se comunica no tienen forma de implementar un nodo CAN que pueda ser unido al bus, además de su gran facilidad en la implementación tanto en hardware como en software.

El puerto de comunicaciones USB fue implementado para que en caso de ser necesario realizar una comunicación más aparte de las mencionadas, este pueda ser utilizado. El desarrollo de hardware de este no es complicado ya que el MCF51JM128VHL ya tiene su controlador USB y lo único que agrego fueron las resistencias de pull-down y el conector USB.

Además de lo mencionado anteriormente, el diseño también incluye unos puertos de entrada que serán utilizados para la conexión de diferentes suiches de uso general por si se necesita alguna conexión adicional. También cuenta con los pines de programación, con lo cual se puede modificar el software si es necesario más fácilmente.

Todo el diseño de este nodo fue desarrollado en el software Cadsoft EAGLE v6.4, en la Figura 9 se observar el diseño de la PCB.

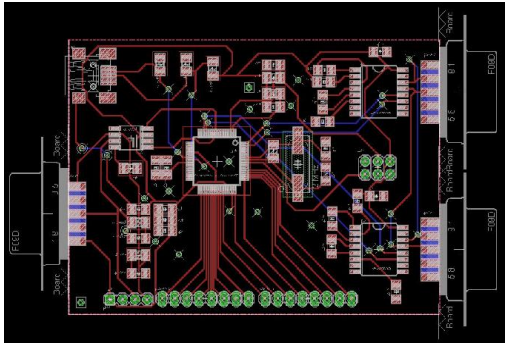


Figura 9. Diseño de la PCB del nodo.

2.3. Nodos secundarios

En el diseño de los nodos que serán utilizados para la recolección de la información de los sensores, se utiliza un microprocesador de la empresa Microchip. Éste es el PIC18F25K80. Éste cuenta con la tecnología ECAN que es perteneciente a todos los dispositivos de Microchip.

Al igual que con el nodo principal, este microprocesador ya tiene incorporado el controlador CAN, por lo que al diseño del hardware solo hubo que añadir el *transceiver* para su funcionamiento. El *transceiver* usado también es perteneciente a la empresa Microchip y es el MCP2551. El diseño de todas las PCB utilizadas para le recolección de la información proveniente de los sensores, es el mismo, solo cambian en la parte de la conexión del sensor.

Describamos las características que ofrece el PIC18F25K80 en el controlador CAN que ofrece que es conocido como ECAN. Es un controlador que soporta las versiones de CAN 2.0A y CAN 2.0B que están definidas en las especificaciones de BOSCH. Algunas de las características que ofrece este dispositivo son: (Microchip, 2011)

- Trama estándar y extendida.
- Hasta 8 bytes para los datos.
- Velocidad programable de hasta 1 Mbit/sec.
- Loopback programable para auto chequeo.
- Fuente de reloj programable.
- Tres buffers de transmisión.
- Dos buffers de recepción.

Una de las partes fundamentales del proyecto es la obtención de las diferentes variables de importancia para el vehículo eléctrico. Para ello se van a utilizar diferentes sensores, para cada uno de éstos se describirá el diseño e implementación del hardware para su funcionamiento.

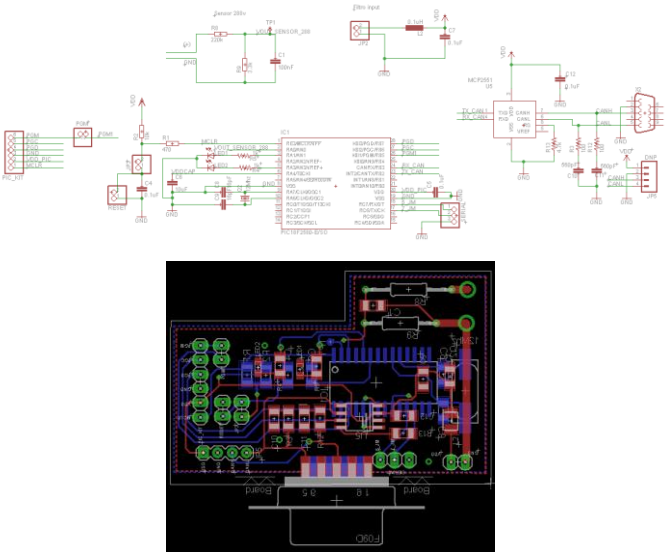


Figura 10. Diseño esquemático y de la PCB del sensor de voltaje.

Voltaje: La importancia de tomar la medida del voltaje en las baterías del vehículo eléctrico radica en su analogía con el nivel de combustible en un vehículo de combustión interna. Al igual que pasa con la gasolina en un vehículo de combustión interna, las baterías son las encargadas del suministro de energía para el motor eléctrico, además, se encarga de alimentar varios de los demás sistemas que conformaran el vehículo eléctrico. Por ello la importancia de tener el control de esta variable por parte del usuario.

Para la medición de esta variable se decidió implementar un sensor que consta de un divisor de voltaje con el cual se hace una trasformación de los 320 voltios que aportan las baterías, a 5 voltios que utiliza para su posterior lectura el ADC del microprocesador. El diseño se puede observar en la Figura 10 (Alzate, 2013)

Temperatura: Otra de las variables a tener en cuenta, ya que al igual que como pasa en un vehículo de combustión interna, el motor tiene una temperatura límite de operación. Lo que nos indica que si la temperatura comienza a tener valores demasiado elevados que pueden llegar al límite o sobrepasarlos, el usuario del vehículo tenga la suficiente información para saber que algo no está funcionando correctamente. Esto con el fin de la protección del motor y el vehículo en general. Pero lo más importante la seguridad del conductor y los demás pasajeros. Para el caso del vehículo eléctrico se debe manejar un límite de temperatura de 60 grados Celsius.

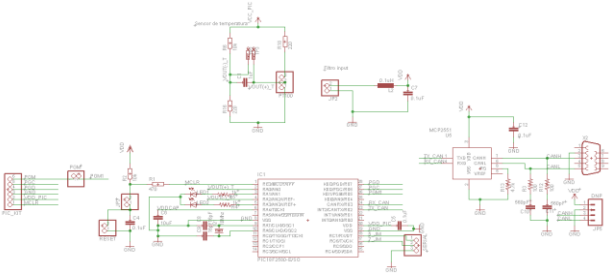
Para la lectura de esta variable, se trabaja con una PT100, éste es un sensor resistivo, el cual presenta cambios en su valor de resistencia cuando se presentan cambios en la temperatura.

Además, también presenta un comportamiento bastante lineal, lo que facilita su lectura. Para este sensor fue necesaria la implementación de un puente de *wheatstone* para poder convertir esos cambios de resistencia en cambios de voltaje para luego realizar la recolección de la información por medio del ADC del microprocesador. El diseño se puede observar en la Figura 11. (Alzate, 2013)

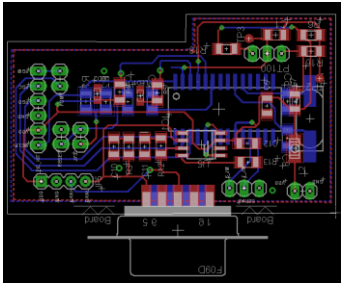
Se implementaron dos sensores de temperatura independientes, uno para tomar la medida del motor eléctrico y el otro para tomar la medida del controlador del motor.

Presión: en los vehículos de combustión interna, este tipo de variable es muy importante, por lo que tiene diferentes tipos de sensores de presión distribuidos para la medida de sistemas como el de inyección de gasolina, entre otros. A diferencia del caso anterior, en el vehículo eléctrico solo se tomaran medidas de presión en el sistema de refrigeración, aunque en un vehículo eléctrico no se manejan casi fluidos, esta medida es importante, ya que se tendrá el control de las bombas que hacen parte de éste sistema con el fin de evitar un mal funcionamiento de éste, dado que va asociada a la temperatura del motor.

Los sensores usados para la medición de presión son los MPX5100ap de Motorola, ideal para el trabajo con microprocesadores, da una medida absoluta de presión entre 2.18 y 16.68 psi (Motorola, 2004). El diseño se puede observar en Figura 12.



(a)



(b)

Figura 11. Diseño esquemático(a) y de la PCB (b) del sensor de temperatura.

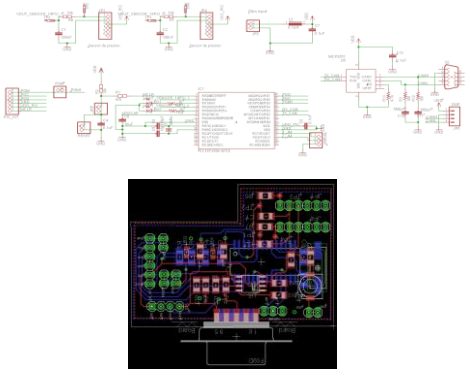


Figura 12. Diseño esquemático y de la PCB del sensor de presión.

Corriente: La medición de esta variable es dado más al estudio del comportamiento del vehículo eléctrico en cuanto a rendimiento y consumo, para futuros mejoramientos que se puedan hacer para poder asegurar una mayor vida útil de las baterías y un mayor rendimiento en kilometraje del vehículo. Con estas mediciones de corriente en las baterías y también en los demás sistemas, se podrá ver con mayor facilidad que elementos son los más críticos para el vehículo en cuestión de consumo y así tener bases para poder ir desarrollando nuevas alternativas. Para las lecturas de corriente se utilizaran dos sensores, el primero para la lectura de los diferentes sistemas que conforman el vehículo, el cual soporta una lectura de hasta 30 amperios. Éste sensor pertenece a la empresa Allegro, su referencia es ACS713. Para la lectura de la corriente en el sistema de baterías se utiliza el sensor L01Z200S05 de la empresa Tamura, este me proporciona lecturas de hasta 200 amperios y es adecuada para el sistema de baterías. Ambos sensores son de efecto hall y el diseño del hardware fue realizado para que entregaran una salida de 0 a 5 voltios según los cambios de corrientes, estas lecturas serán tomadas por el ADC del microprocesador para su posterior transmisión. (Alzate, 2013)
Los diseños de los sensores de corriente se pueden observar en la Figura 13 la utilizada con el sensor Allegro y en la Figura 14 con el sensor Tamura.

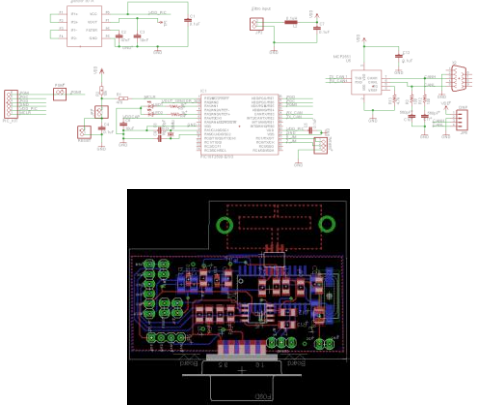


Figura 13. Diseño esquemático y de la PCB del sensor de corriente de 30 amperios.

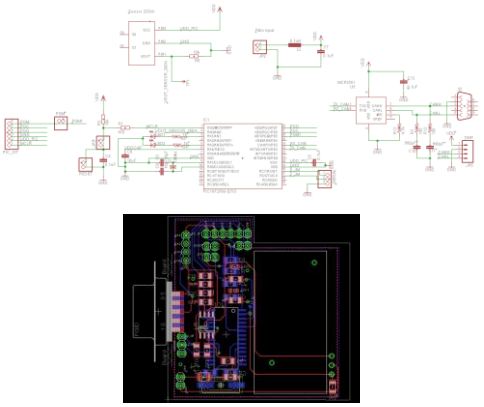


Figura 14. Diseño esquemático y de la PCB del sensor de corriente de 200 amperios.

2.4. Módulo de visualización

Para la implementación del módulo que servirá como el tablero del vehículo eléctrico, en el cual se podrán observar todas las variables mencionadas anteriormente en la descripción de los nodos secundarios, se utilizara una pantalla TFT – LCD con una resolución de 320 x 240, el modelo de ésta es NHD-3.5-320240MF-ATXL#-T-1 de la empresa NEWHAVEN DISPLAY, y tiene la característica de ser touch resistiva. Estas a diferencia de las pantallas que poseen el touch capacitivo necesitan ser presionadas con mayor fuerza para poder detectar el punto de contacto.

El controlador que se utiliza para esta pantalla es el SSD1963 del fabricante SOLOMON SYSTECH. Éste es un controlador de display que soportan 24 bits de color y resoluciones de hasta 864x480. Como no se encontró la forma de conseguir el componente en ningún proveedor, se optó por conseguir una board de desarrollo basada en este controlador el modelo de éste es NHD-3.5-320240MF-22 de la empresa NEWHAVEN DISPLAY, tanto la pantalla como la board se pueden observar en la Figura 15.

Figura 15. Lcd-tft y board controladora.

Para el desarrollo de este módulo también fue necesario el desarrollo de una interfaz por medio de un microprocesador para la inicialización del controlador del display y él envío de la información al display, ya que la tarjeta de desarrollo tiene un puerto para la interfaz con éste, donde recibe los cambios de estado para los pines de control los cuales se manejan para la

escritura y lectura en el display. En la Figura 16 podemos observar el esquemático de éste y en la Figura 17 observamos el diseño de la PCB. La información viaja por medio de un protocolo 8080 donde es enviada a través de un bus paralelo de 8 bits. El microprocesador a usar es un Freescale MCF51JM128VHL de la familia Coldfire V1, el mismo usado en la implementación del nodo CAN principal, ya que este ofrece la cantidad de pines necesarios para el desarrollo de la interfaz con el controlador del display.

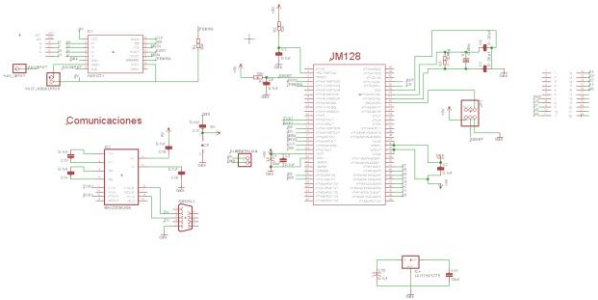


Figura 16. Esquemático de la interfaz del microprocesador para la conexión con el controlador del display.

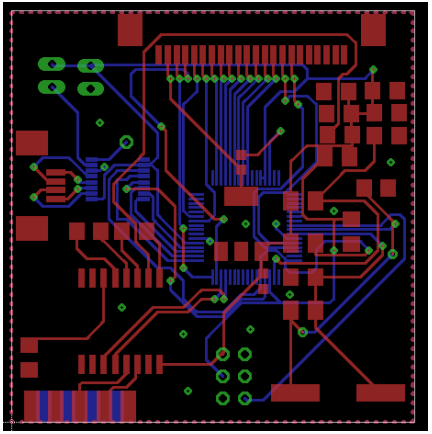


Figura 17. Diseño PCB de la interfaz del microprocesador con el controlador del display.

3. DESARROLLO DEL SOFTWARE

El desarrollo del software se realizó en dos distintos ambientes de programación, esto por lo nombrado anteriormente que se utilizaron dos familias distintas de microprocesadores en los nodos pertenecientes al bus CAN. Esta parte del desarrollo fue el más complicado y en donde aparecieron los mayores problemas, tales como:

- La configuración de las velocidades de trabajo de los controladores CAN, para que trabajasen todos a la misma.
- Comprender el funcionamiento del ID, los filtros y las máscaras.

Durante el desarrollo de esta sección se ira adentrando más en estos problemas y las soluciones desarrolladas. También se detallara detenidamente la configuración de cada uno de los módulos que componen los controladores CAN y su función, para el correcto funcionamiento de estos.

3.1. Nodo principal

Para el desarrollo del software del nodo que fue diseñado con un controlador CAN integrado en el microprocesador MCF51JM128VHL perteneciente a Freescale, se usó la herramienta Codewarrior(r) V1.3. Esta herramienta es basada en eclipse y el desarrollo del software de los dispositivos se hace por medio del lenguaje C. CodeWarrior nos ofrece un entorno de trabajo muy práctico y fácil de usar.

A la hora de la programación del dispositivo se usó el sistema de programación BDM-Multilink, el cual no solo permite la programación del dispositivo, sino también el poder hacer debug y conocer el comportamiento del desarrollo en tiempo real, esto para la solución de errores que se puedan presentar.

3.2. Configuración del protocolo CAN en el nodo principal

Es recomendable seguir un orden a la hora de realizar la configuración del módulo CAN. Los registros que se configuran en este módulo son:

- Registros de control.
  - CANCTL0
  - CANCTL1
- Registros de configuración del bit timming.
  - CANBTR0
  - CANBTR1
- Registros de filtros de aceptación.
  - CANIDAC
  - CANIDAR0-7

- CANIDMR0-7

- Registros de interrupción.
  - CANRIER
  - CANTIER

Para la configuración del módulo, lo primero que se debe realizar es la inicialización. En éste el modulo pierde la sincronización y cancela todas las recepciones o transmisiones que se estén realizando en ese momento. En este modo se resetean y llevan a su valor por defecto los registros CANCTL0, CANRFLG, CANRIER, CANTFLG, CANTIER, CANTARQ, CANTAAK, Y CANTBSEL. Además se configuran los registros CANBTR0, CANBTR1, CANIDAC, CANIDAR, CANIDMR.

Para entrar al modo de inicialización se debe colocar en "1" el bit INITRQ del registro CANCTL0, este es el encargado de hacer el request para que el modulo entre en éste modo. Luego se debe esperar la bandera INITAK que pertenece al registro CANCTL1, que indica que el módulo entro en modo de inicialización. Al entrar en éste modo el módulo ya está listo para que se le puedan configurar los demás registros.

Los registros CAN0CTL1, CAN0BTR0, CANBTR1, son los encargados de la configuración del bit time.

Para la configuración de estos tiempos se tiene que tener en cuenta como está conformado cada bit, éstos están conformados por varios segmentos y a su vez estos segmentos están conformados por una unidad llama ``time quanta" (Tq). Por medio de un preescaler se puede configurar un clock de Tq desde el registro CANCLK.

Para el desarrollo se va a implementar una velocidad de transmisión de 250KHz, para lograr esta velocidad tenemos que configurar los segmentos que conforman un Bit Time.

Los segmentos que conforman estos Bit Time se dividen en:

- SYNC_SEG
- Time Segment 1 (TS1)
- Time Segment 2 (TS2)

Los cuales son configurados en los registros CANBTR0 y CANBTR1. Por medio de (1) se puede hallar el valor del preescaler necesario según las configuraciones escogidas para obtener la velocidad que se desea.

$$BitTime = \frac{PreescalerValue}{f_{CANCLK}} * NofTQ \quad (1)$$

Donde NofTQ = Number of TimeQuanta

Para hallar el valor de NofTQ se tiene a (2).

$$NofTQ = 1 + TS1 + TS2, \quad (2)$$

Para la recepción, se utilizan registros de máscaras y de aceptación. El cual tiene la tarea de crear un filtro por el cual pasan todos los mensajes que llegan al nodo, con el propósito de

desechar los mensajes que no son de interés para el nodo en particular y recepcionar los que si van dirigidos a él.

La configuración se realiza por medio de los registros CANIDMR0 - CANIDMR7 para los bits de máscaras. Estas son las encargadas de seleccionar que bits pertenecientes al ID del mensaje que llega al nodo serán comparados con los bits de aceptación de los filtros ya configurados en el nodo receptor, que para este caso están configurados en los registros CANIDAR0 - CANIDAR7.

Como se defino anteriormente, este nodo es el que recibe toda la información proveniente del bus, por lo que los filtros que utiliza para la recepción de mensajes se dejaron desactivados. Lo que implica que todos los mensajes que lleguen a él serán recepcionados sin ninguna restricción.

Para la desactivación de estos es necesario que todos los bits de los registros CANIDMR0 - CANIDMR7 estén en "1'' (Freescale, 2009), ya que esto permite al receptor recibir el mensaje sin tener que filtrarlo. En la Tabla 4 se puede observar mejor como es el funcionamiento de los filtros.

Para la inicialización del módulo CAN se utiliza la función "CANInit", en la cual se realizan todas las configuraciones necesarias para el funcionamiento de este.

En esta función encontramos la configuración de la velocidad a trabajar, de las máscaras, de los filtros y los modos de operación.

Para el desarrollo se escogió una velocidad de transmisión de 250KHz, las máscaras fueron configuradas para que el nodo recepcione todos los mensajes que son enviados a través del bus, ya que la función de esta es la recolección de toda la información enviada por los sensores. Dada la configuración anteriormente mencionada de las máscaras no importa como estén configurados los filtros ya que no serán utilizados en la recepción de mensajes.

Para el envío de mensajes CAN por medio de este nodo, se tiene la función "CAN0SendFrame", esta recibe por parámetros un ID a utilizar, un dato a enviar, el tamaño del dato y una prioridad.

Tabla 4. Manejo de filtros

Bit de mascara	*Bit de filtro*	*Bit ID*	*Resultado*
1	X	X	Aceptado
0	0	0	Aceptado
0	0	1	Rechazado
0	1	0	Rechazado
0	1	1	Aceptado

3.3. Nodos secundarios

Para el desarrollo del software de los nodos se utilizó la herramienta de programación MPLAB v 1.85 proporcionada por Microchip. Esta herramienta es basada en eclipse y el desarrollo del software de los dispositivos se hace por medio del lenguaje C.

A la hora de la programación de los dispositivos se usó el sistema de programación Pickit3, el cual no solo permite la programación del dispositivo, sino también el poder hacer debug y conocer el comportamiento del desarrollo en tiempo real. Esto para la solución de errores que se puedan presentar.

En los nodos secundarios, recolectamos la información tomada por los diferentes sensores que serán utilizados para el desarrollo del proyecto. Esta información obtenida será procesada por medio de los conversores análogo – digital que posee cada micro procesador utilizado, para así digitalizar la señal que es recibida desde los sensores para poder ser transmitida por el bus CAN.

En estos nodos se realizaran todos los procesamientos necesarios de la información obtenida de los sensores para que así el dato transmitido por el bus no tenga que ser procesado por el nodo principal, ahorrando así tiempo de ejecución en el nodo principal.

El primer registro en configurar es:

- CANCON

Este es el registro de control del módulo y su función es la de proporcionar el modo de operación de este. El registro debe estar programado en modo de configuración, para así poder configurar los demás registros como lo son las máscaras y filtros. Luego de esto se debe llevar a el modo en que se piensa trabajar, para este caso será en modo de operación normal. También se puede configurar en modo loopback si se necesita hacer algún chequeo.

Para la transmisión se tiene los registros:

- TXBnCON
- TXBnSIDH
- TXBnDLC

Para la recepción tenemos los siguientes registros a configurar:

- RXB0CON
- RXB1CON
- RXBnSIDH
- RXBnDLC

Y así en la Figura 18 se muestra el diagrama de flujo que sigue el código implementado en el nodo principal.

3.4. Configuración del protocolo CAN en los nodos secundarios

Empezaremos por describir como es la configuración del *baud rate* en estos procesadores. Al igual que con el nodo principal se va a manejar una velocidad de transmisión de 250KHz.

Para la configuración de esta velocidad se tienen los registros de bit timing configuration que son:

- BRGCON1
- BRGCON2
- BRGCON3

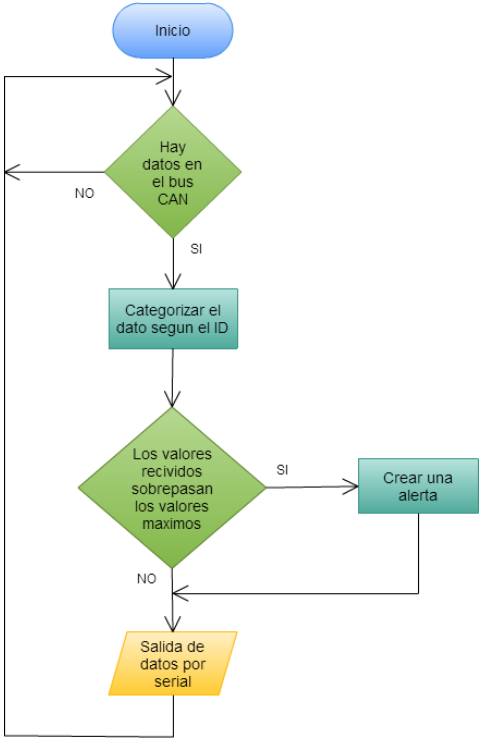


Figura 18. Diagrama de flujo del código implementado en el nodo principal.

Al configurar estos registros se están modificando los parámetros que componen un bit de trasmisión, los cuales son:

- Synchronization Segment (Sync_Seg)
- Propagation Time Segment (Prop_seg)
- Phase Buffer Segment 1 (Phase_seg1)
- Phase Buffer Segment 2 (Phase_seg2)

Al asignar los valores de estos y por medio de (3) y (4) se logra obtener la velocidad de trabajo deseada.

$$Nominal\ Bit\ Time = T_Q * (Sync\_Seg + Prop\_Seg + Phase\_Seg1 + Phase\_Seg2) \quad (3)$$

$$T_Q = (2 * (BRP + 1))/Fosc(MHz) \quad (4)$$

Al configurar los filtros y máscaras para estos, se encontró que a diferencia del microprocesador usado en el nodo principal, un bit en “0” en la máscara permite el paso del sin comparación en los filtros como se muestra en la Tabla 5.

Para la configuración de los filtros y las máscaras se usan los registros:

- RXMnSIDH para las máscaras.

- RXFnSIDH para los filtros.

Para la implementación del periférico CAN en estos nodos, se utilizó la librería CAN18xx8, en donde se proveen las funciones para la utilización del protocolo para dispositivos Microchip de la familia PIC18F.

Esta librería nos proporciona funciones que facilitan el uso del módulo CAN como lo son:

- CANInitialize: con esta se configura el modulo, los filtros y las mascaran.
- CANSetOperationMode: selecciona el modo con el cual se pienza trabajar, puede ser modo normal, modo loopback, modo de configuración entre otros.
- CANSetBaudRate: esta se utiliza para la configuración de la velocidad de trabajo del módulo.
- CANIDToRegs: asigna el ID que el usuario da al nodo a los registros del módulo.
- RegsToCANID: al contrario del anterior, asigna el valor de los registros a un ID.
- CANSetMask: configura las máscaras del módulo.
- CANSetFilter: configura los filtros del módulo.
- CANSendMessage: la función se asigna un ID, un dato, el tamaño del dato y una prioridad a los registros del módulo que serán trasmitidos por el bus.
- CANReceiveMessage: recibe los mensajes trasmitidos por el bus.

3.5. Código final de los nodos secundarios

Además de la implementación del protocolo CAN en cada uno de estos nodos, se utilizaron los conversores análogo-digital disponibles en los microprocesadores para el procesamiento de la información obtenida por medio de los sensores.

Tabla 5. Manejo de filtros

Bit de mascara	*Bit de filtro*	*Bit ID*	*Resultado*
1	X	X	Aceptado
0	0	0	Aceptado
0	0	1	Rechazado
0	1	0	Rechazado
0	1	1	Aceptado

Como se observa en la Figura 19, el diagrama de flujo que sigue el código implementado en estos nodos lo que hace es verificar la disponibilidad del bus para enviar un mensaje, en la lectura del conversor análogo digital, lo que se tiene es la digitalización de la señal obtenida del sensor por medio de un conversor de 12 bits lo que me da un valor máximo de 4096, por esta razón se debe realizar un procesamiento de la señal para así convertir este valor que me envía el conversor y convertirlo por medio de operaciones matemáticas a unidades de presión, temperatura, voltaje, corriente, etc. Luego de realizar este procesamiento y obtener el valor en las unidades necesarias es enviado este dato por medio

del bus al nodo principal, que como ya se menciono es el encargado de gestionar estos datos.

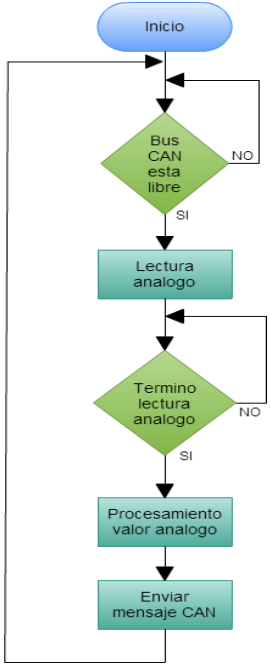


Figura 19. Diagrama de flujo del código implementado en los nodos secundarios.

Tabla 6. Identificadores para cada una de la variables para definir prioridad

Variable	Identificador
Temperatura Motor	0x102
Temperatura Controlador	0x103
Presión Bomba A	0x104
Presión Bomba B	0x105
Corriente 30 Amperios	0x107
Corriente 200 Amperios	0x108
Voltaje baterías	0x106

En la Tabla 6 se observa los identificadores asignados a cada uno de los nodos, la prioridad se asignó desde el más relevante hasta el menos relevante.

4. RESULTADOS

Durante el desarrollo del prototipo se siguieron diferentes pasos para su elaboración. Tenemos la consulta previa que se realizó para escoger las variables a medir y que tipo de sensores se utilizarían para esto, luego el diseño de los diferentes nodos y su armado y por ultimo encontramos el desarrollo para la unión de cada uno de estos módulos con el sistema de visualización. En la Figura 20 se observa el diagrama del sistema completo, se observa que además de la implementación del protocolo CAN, se tiene comunicación tanto con la interfaz gráfica, como con el PLC que será el encargado del control del vehículo.

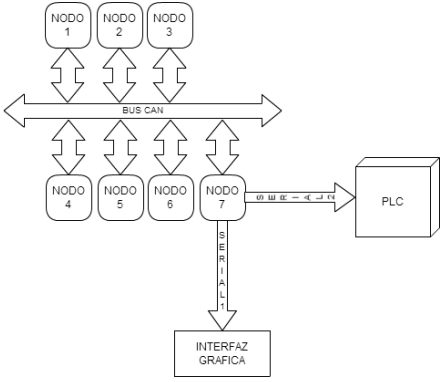


Figura 20. Diagrama de las conexiones del sistema.

En la Figura 21 se muestra dispositivo terminado que actúa como nodo principal.

Figura 21. PCB terminada.

En la Figura 22, se puede observar el nodo encargado de la toma de la temperatura, se tiene dos de estos, uno para la temperatura del motor y otro para la temperatura del controlador. En la Figura 23 se muestra el sensor de presión, para este caso también se implementaron dos nodos ya que se desea saber el comportamiento en presión de las dos bombas que usa el sistema de refrigeración.

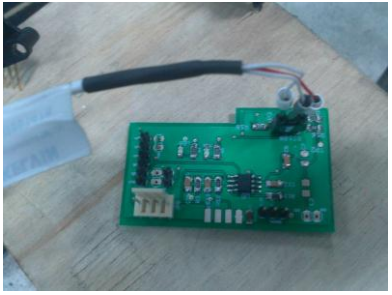


Figura 22. Nodo temperatura.

Figura 23. Nodo presión.

En la Figura 24 y en la Figura 25 se muestran los sensores de corriente. En la Figura 26 se muestra el sensor de voltaje.

Figura 24. Nodo corriente 200 amperios.

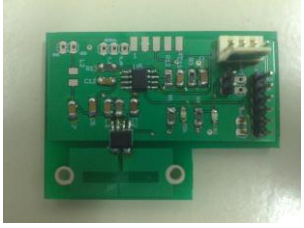


Figura 25. Nodo corriente 30 amperios.

4.1. Protocolo CAN

Luego de tener los sensores y los dispositivos listos, se procedió a la programación de cada uno de los microprocesadores, cada una de las diferentes funciones que tenían que realizar.

El primer problema que se presentó durante el desarrollo del prototipo fue la comunicación de todos los módulos por medio del protocolo CAN. Para la comprobación del correcto funcionamiento de estos, lo primero que se realizó después de tener configurados los periféricos CAN en cada microprocesador, fue hacer una prueba en modo loopback, para ver si estos estaban bien inicializados, comprobar el envío y recepción de mensajes de cada uno de estos. En esta parte del procedimiento no se presentaron problemas ni con el chip de Freescale ni con el de Microchip.

Figura 26. Nodo voltaje.

Luego se realizó la comunicación entre el nodo principal y uno de los nodos secundarios, presentándose él primer problema, ya que nunca llegaba el mensaje enviado desde el nodo secundario al principal ni viceversa. Por ende se optó conectar por separados cada una de las familias, se conectaron dos periféricos CAN de Freescale y se encontró que funcionaba perfectamente, no se presentaba ningún problema ni en la transmisión ni en la recepción. Al contrario de lo que paso al conectar dos nodos secundarios que trabajan con chips de Microchip, entre estos no funcionaba el protocolo, no había ni envío ni recepción de datos. El problema se estaba presentando en la velocidad, ya que estaba transmitiendo demasiado rápido debido a que se tenía activado el PLLx4, que lo que hace es multiplicar la velocidad del cristal con el que trabaja el microprocesador por 4. Al realizar el cambio de velocidad los nodos empezaron a trabajar correctamente entre ellos.

Al comprobar por separado que ambas familias estaban bien inicializadas en su protocolo, se procedió a programar la misma velocidad de 250kbits en cada uno de ellos, esto indica que la duración de cada bit en el tiempo es de 4µs.

Al configurar esta velocidad en cada uno de los nodos, se procedió a conectar el nodo principal con un nodo secundario, pero aun así no funcionaba la comunicación, por lo que se procedió a verificar la velocidad de cada uno por medio de un osciloscopio.

Al realizar esta prueba se pudo observar que había una diferencia significativa en la velocidad de cada uno, lo que no permitía la sincronización ya que el nodo principal estaba trabajando a una velocidad de 3.98µs y el nodo secundario estaba a una velocidad de 4.2µs. El comportamiento de la comunicación debido a esta diferencia se puede observar en la Figura 27. En ésta prueba se puso a transmitir a uno de los nodos y el otro a recibir y luego se

cambió el orden con el fin de ver la velocidad de transmisión de cada uno, transmitiendo el mismo ID y el mismo dato. De color rojo tenemos la transmisión del nodo construido con un microprocesador Freescale y de azul tenemos la transmisión del nodo construido con un microprocesador Microchip. Como se observa se pierde el sincronismo entre ambos.

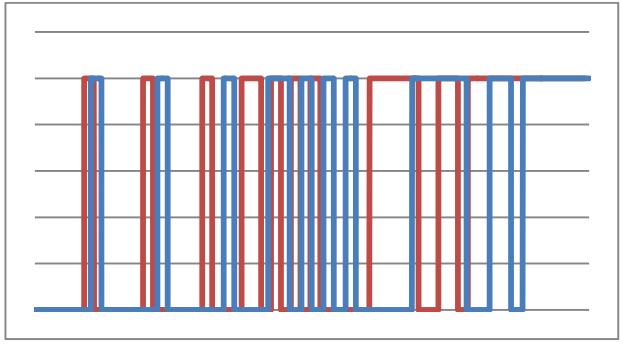


Figura 27. Comparación entra las velocidades entre los chips de Freescale y Microchip.

Se realizaron de nuevo las configuraciones de velocidad en cada uno de los nodos con el fin de que ambos transmitieran a la misma velocidad. Para ello se tuvo que configurar la opción del PLL en cada uno para tener mayor precisión a la hora de calcular el preescaler.

Al realizar estos cambios se consiguió que las velocidades en los nodos fuera igual. Esto nos indica el poco margen de error en la velocidad que se presenta en un bus CAN, y la importancia de saberlo al implementar un protocolo de estos con diferentes familias de dispositivos, también al tratar de conectar un nuevo nodo a un protocolo ya implementado en el cual no se conoce ninguna de las características de construcción.

Al tener implementado y funcionando el protocolo, se puede observar en la Figura 28 el comportamiento de este. Analizando ésta, se tiene que línea roja representa el dato transmitido por medio del nodo secundario. De color azul tenemos el bus, que como se observa es el mismo que el dato que se envía por el nodo secundario con una excepción, al final de la trama se puede observar que se coloca un bit dominante en el bus, esto es debido a la respuesta por parte del nodo principal de que recibió el mensaje correctamente, éste es el *acknowledge*, que se encuentra de color negro, ésta línea pertenece a la transición del nodo principal.

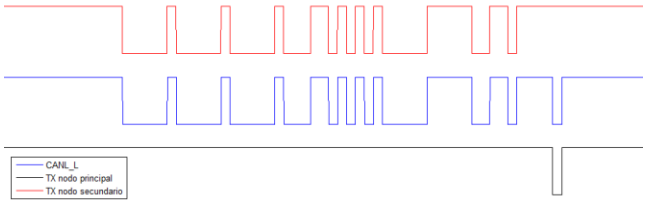


Figura 28. Trama de datos protocolo CAN.

4.2. Obtención de las variables y digitalización

Para la obtención y posterior digitalización de las variables a medir, se utilizó el conversor análogo – digital del microprocesador PIC18F25K80. Éste ADC tiene una resolución de 12 bits.

Para la visualización de las variables se usaron las ecuaciones características que se obtuvieron de la caracterización de cada uno de los sensores. Esta ecuación relaciona el voltaje del ADC, que para este caso es 5 voltios, con la respectiva variable a medir, bien sea corriente, temperatura, presión, etc. Para mayor información dirigirse a (Alzate, 2013).
Se realizaron varias pruebas para la medición de estas obteniendo resultados aceptables con errores pequeños. Con excepción de la temperatura, ya que a ésta está siendo afectada por el ruido del sistema y la medida es muy ruidosa como se observa en la Figura 29.

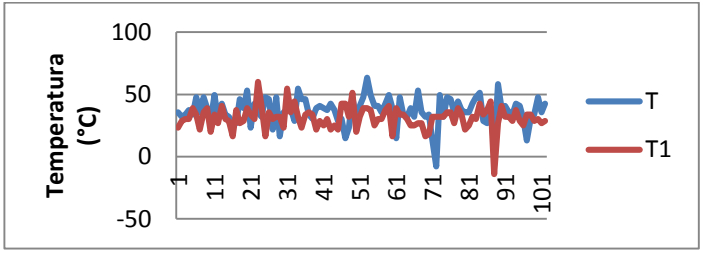


Figura 29. Resultado obtenido de la lectura de temperatura.

4.3. Visualización

Para la parte de visualización se trabajó con una pantalla TFT-LCD y una tarjeta controladora a base del chip SSD1963. En esta parte no se presentaron mayores problemas. Se desarrolló una interfaz gráfica en la cual se muestren las variables deseadas, además de algunas alarmas. En la Figura 30 se muestra el resultado del funcionamiento de la pantalla. Se pueden crear diferentes formas de visualización de los datos, este solo sería una idea de las posibles formas de visualización.

Figura 30. Visualización en la pantalla TFT-LCD.

5. CONCLUSIONES

Se diseñó e implemento una red de comunicaciones CAN funcional y robusta capaz de controlar la comunicación entre los diferentes subsistemas como lo son los sensores, que conforman un vehículo eléctrico, de forma segura, rápida, con poco error y eficiente con posibilidad de expansión. Además se construyó la interfaz gráfica que funciona como tablero, en el cual se mostraran los diferentes datos que se obtiene por medio de la red CAN implementada.

Se realizó el desarrollo de los diferentes nodos que conforman la red CAN, los cuales son los encargados de la recolección de los datos que se obtiene mediante los sensores. La escogencia de los sensores se realizó teniendo en cuenta cuales son las variables prioritarias para el estudio del comportamiento del vehículo eléctrico.

Al participar en la conversión de un vehículo de combustión interna a un vehículo eléctrico por medio de este proyecto se nota como cambian la necesidad de la instrumentación interna del automóvil, ya para un vehículo de combustión se necesita variables como nivel de gasolina, nivel de líquido de frenos y para el motor eléctrico se necesita medir variables como la presión de la bombas de refrigeración, voltaje en la baterías entre otras. Pero también tienen sus similitudes ya que en ambos diseños necesita estar monitoreando la temperatura para evitar que fallas en el sistema.

El protocolo CAN mostro ser una muy buena opción cuando se necesita hacer un desarrollo de comunicación que sea rápido y robusto ya que debido a la implementación se necesita fidelidad de la información y este desarrollo está expuesto a unas condiciones a las cuales puede haber mucho ruido electromagnético y mecánico y la lectura de los sensores no es la mejor con otros tipos de comunicación como la seria o la inalámbrica entre otras. Aun así se usa comunicación serial para en algunos de las comunicaciones pues no son de largo alcance como lo puede ser entre los sensores y la tarjeta principal

REFERENCIAS

Alzate, P. D. (2013). *Dispositivo para telemetría de vehículo eléctrico.* Medellín

Bolton, W. (2001). *Mecatrónica. Sistemas de control electrónico en ingenieria mecánica y eléctrica.* Alfaomega.

BOSCH. (2002). *Los sensores en el automóvil.*

Bosch, R. (1991). *CAN Specification*.

Calva Cuenca, J. (2010). *Diseño e Implementación de Protocola CAN paea el Control de un Módulo de Red de Sensores.* Quito.

cristobaldominguez. (sin año). Recuperado el 6 de 09 de 2013, de http://www.cristobaldominguez.com/ficheros/pantalla%20tactil.pdf

Ekiz, H., Kutlu, A., & Powner, E. (1996). Desing and Implementation of a CAN / CAN Bridge. *IEEE*.

Freescale. (2009). Obtenido de http://www.freescale.com/: http://cache.freescale.com/files/32bit/doc/ref_manual/MCF51JM128RM.pdf?fpsp=1

Fresno, J. A. (2004). Obtenido de http://deeea.urv.cat/public/PROPOSTES/pub/pdf/561pub.pdf

Huiercán, & Ignacio, J. (si año). Recuperado el 06 de 11 de 2013, de http://quidel.inele.ufro.cl/~jhuircan/PDF_CTOSII/ad03.pdf

ISO 11898-1. (2003). *Road vehicles — Controller area network.*

Marino, A., & Schmalzel, J. (2007). Controller Area Network for In - Vehicle Law Enforcement Applications.

Microchip. (2010). Recuperado el 2013, de http://ww1.microchip.com/downloads/en/DeviceDoc/21667f.pdf

Microchip. (2011). Recuperado el 2013, de http://ww1.microchip.com/downloads/en/DeviceDoc/39977f.pdf

Motorola. (2004). Recuperado el 10 de 11 de 2013, de http://pdf.datasheetcatalog.com/datasheet/motorola/MPX5100D.pdf

Navet, N. (s.f.). COntroller area network. CANs use within automobiles. *IEEE Potentials*.

Pérez, J. E. (2013). *Control de sub-sidtemas para el proyecto de conversión del vehículo eléctrico.* Medellín.

Provencher, H. (Abril de 2012). *Controller Area Network For Vehicles*.

Ran, L., Junfeng, W., & Haiying, W. (2010). Design Method od CAN bUS Network Communication Structure for Electric Vehicle.

Rosa, J. J. (sin año). Recuperado el 06 de 11 de 2013, de http://www2.uca.es/grup-invest/instrument_electro/ppjjgdr/Electronics_Instrum/Electronics_Instrum_Files/temas/T11_CAD.pdf

Wikipedia. (8 de 11 de 2013). Recuperado el 10 de 11 de 2013, de http://es.wikipedia.org/wiki/Thin-film_transistor

AUTOR

Juan Carlos VÉLEZ GIRALDO, egresado del programa de Ingeniería Electrónica de la Universidad Pontificia Bolivariana. Participación en el día técnico entre 2007 y 2011, en el FISE 2011. Integrando de los semilleros de Automática y Diseño A+D y del semillero de electrónica ambos de la Universidad Pontificia Bolivariana. Participo en la XII olimpiada robótica 2011.

Daniel Alberto Arroyave Molina, bachiller del colegio de la universidad pontificia bolivariana 2005. Estudiante de Decimo semestre de ingeniería electrónica en la universidad pontificia bolivariana, integrante del semillero de microelectrónica en la UPB desde el segundo semestre de 2011.

.