ESCUELA POLITÉCNICA DEL EJÉCITO

FACULTAD DE INGENIERÍA ELECTRÓNICA

PROYECTO DE GRADO PARA LA OBTENCIÓN DEL TÍTULO EN INGENIERÍA ELECTRÓNICA

ANÁLISIS E INVESTIGACIÓN SOBRE LAS CARATERÍSTICAS FUNDAMENTALES DEL PROTOCOLO CONTROLLER AREA NETWORK (CAN).

IVÁN ALEJANDRO VERGARA VARGAS

QUITO – ECUADOR

2005

CERTIFICACIÓN

Certificamos que el presente proyecto de grado: “Análisis e investigación sobre las características fundamentales del Protocolo Controller Area Network (CAN)”, fue desarrollado en su totalidad por el señor Iván Alejandro Vergara Vargas, bajo nuestra dirección.

Atentamente,

Ing. Byron Navas	Ing. Víctor Proaño
DIRECTOR	**CODIRECTOR**

AGRADECIMIENTO

A Dios por ser mi guía y brindarme esta oportunidad.

A mis padres por haberme inculcado valiosos valores, por haber sido un excelente ejemplo de responsabilidad y principalmente por haberme brindado su absoluta confianza durante el desarrollo del proyecto.

A mi hermana y a mi cuñado por su constante apoyo.

A mi primo Pablo Ramos, Vanesa Vargas y Fernanda Díaz, y en general a todos mis amigos por su ayuda y colaboración.

A la Escuela Politécnica del Ejército, a los profesores, y en especial, a mi Director de tesis, Ing. Byron Navas y Codirector Ing. Víctor Proaño, por haberme instruido a lo largo de mi carrera, y guiado correctamente en la elaboración de mi proyecto de grado.

Y a todos quienes, de una manera u otra, colaboraron con el desarrollo del presente proyecto.

DEDICATORIA

A mis padres, a mi hermana, a mi cuñado, Dr. Fabrisio Ortiz y a mi sobrinita Emily

PROLOGO

El protocolo CAN o Controller Area Network fue originalmente diseñado para utilizarlo en la industria automotriz y tuvo una gran acogida por los fabricantes de microcontroladores más importantes a nivel mundial, en la actualidad CAN es el protocolo más utilizado para aplicaciones en sistemas de comunicación de datos de vehículos debido a su alta fiabilidad y excelente relación costo-eficacia, esto se aplica a un amplio rango de sistemas móviles como autos, aviones, trenes o ascensores.

Los sistemas de comunicación basados en el protocolo CAN también juegan un papel muy importante en la realización de sistemas distribuidos en cualquier tipo de aplicación, desde las máquinas de café hasta sistemas quirúrgicos.

En los últimos años CAN ha emigrado hacia aplicaciones desarrolladas no solamente en el área vehicular sino también en diversos campos de automatización y control industrial creando la necesidad de un protocolo abierto estandarizado de alto nivel que proporcione un sistema de intercambio de mensajes fiable junto con medios apropiados para detección, configuración, operación de nodos y administración de red.

Se han desarrollado varios protocolos de alto nivel para la implementación de sistemas que operen bajo CAN tales como SAEJ139, DeviceNet y el más popular en la actualidad CANopen.

Casi todos los proveedores principales de módulos de entrada/salida, sensores inteligentes y dispositivos de automatización en general ofrecen productos que incluyen alternativas que trabajan de acuerdo a las especificaciones CanOpen o DeviceNet, ya que ambas soluciones son muy fiables y rentables para la transmisión de datos en tiempo real y garantizan la interoperabilidad e intercambiabilidad de dispositivos bajo un perfil estandarizado.

Por estas razones el presente proyecto de investigación tiene por objetivo principal brindar las bases teóricas necesarias para los diseñadores y usuarios de sistemas de comunicación basados en el protocolo CAN.

En el CAPITULO 1 se realiza una introducción a las características principales de un protocolo de comunicaciones enfocadas al protocolo CAN y el papel que este desempeña dentro del modelo OSI.

En el CAPÍTULO 2 se presentan todas las características relacionadas a la capa de enlace de datos del protocolo CAN incluyendo temas como tipos de tramas, tipos de detección de errores, confinamiento de fallas, protocolo CAN extendido entre otros.

El CAPITULO 3 presenta temas relacionados a la capa física tales como la señalización física, medios de transmisión, topología de red, medio de acceso al bus, estándares de capa física, versiones del protocolo CAN, entre otros.

El CAPITULO 4 presenta el protocolo CAN enfoca las diferentes alternativas de implementación existentes en el mercado, es decir los diferentes controladores, microcontroladores y transceivers clasificándolos según sus fabricantes.

En el CAPITULO 5 se analizan los diferentes protocolos complementarios o de alto nivel que se pueden utilizar para la aplicación del protocolo CAN tales como CANopen, DeviceNet, SAE J1939, sus principales características y diferencias.

El CAPITULO 6 trata los temas principales relacionados al diseño de redes CAN, implementación de nodos y criterios de selección.

En el CAPITULO 7 se muestran algunas aplicaciones del protocolo CAN en diferentes campos.

El CAPITULO 8 contiene las conclusiones y recomendaciones de la investigación.

ÍNDICE

DATASHEETS. Las hojas de especificaciones de algunos controladores y microcontroladores CAN, se encuentran en el cd adjunto al documento de tesis: Atmel T89C51CC02, Infineon 82C900, Nacional Semiconductor 884BC, Microchip PIC18F6585, Dallas DS80C390, Intel 82527, Motorola MC68HC05X16, Philips PCA 82C250. Transceiver Microchip MCP2551.

CAPÍTULO 1

INTRODUCCIÓN

Controller Area Network o CAN es uno de los protocolos más utilizados en la actualidad para transferencia de mensajes en ambientes distribuidos, tiene una arquitectura de bus multi-master o multimaestra. Entre sus cualidades más significativas proporciona características de respuesta en tiempo real y control de fallas ya sea en la recepción de mensajes o en un mal funcionamiento de los nodos. Dentro del modelo OSI, su estructura se basa en dos capas: la capa física y la capa de enlace de datos y además se vale de protocolos de alto nivel para la capa de aplicación.

Es importante analizar las características y fundamentos generales de un protocolo de comunicación para luego profundizar en relación al protocolo CAN.

1.1 COMUNICACIÓN DE DATOS EN EL DOMINIO DE CAMPO

La automatización en la actualidad viene buscando una descentralización de procesamiento de datos e interfaces, el uso de sistemas de comunicación de datos en forma serial en lugar de las tecnologías de instalación eléctricas convencionales garantizan una mayor flexibilidad con respecto a modificaciones y actualizaciones, también proporcionan una considerable reducción de costos de planificación e instalación en muchas áreas de automatización industrial. Además del ahorro en costos de hardware e instalación, existen posibles ahorros sustanciales con respecto a costos de diseño, mantenimiento y operación.

El término dominio de campo representa la base que soporta a un sistema de automatización o a una planta industrial y está directamente relacionado con el proceso técnico en sí, por lo tanto, el dominio de campo abarca todos los dispositivos y equipos (dispositivos de campo) que interactúan directamente con el proceso a ser monitoreado o

controlado, de ahí que los dispositivos de campo no son solo los dispositivos utilizados para el proceso técnico tales como PLCs, PCs, sensores e instrumentos de medición; sino también, el equipo que influye directamente en el proceso como interrruptores, equipos reguladores, equipos de mando y operación y equipos de despliegue de información.

Los sistemas de comunicación que realizan la distribución de dispositivos electrónicos y controlan su funcionamiento se llaman buses de campo.

El protocolo CAN ha desempeñado un papel muy importante en la tecnología de sistemas de bus de campo vía serial, empezando en los vehículos de motor y actualmente en una variedad de aplicaciones industriales. Aunque existe una gran diversidad de sistemas de bus de campo disponibles, sólo pocas soluciones han ganado las posiciones sustanciales en el mercado y pueden ser consideradas como normas industriales.

1.2 EL MODELO DE REFERENCIA PARA LA COMUNICACIÓN DE DATOS

El modelo de referencia utilizado en la actualidad para especificaciones protocolares es el modelo OSI desarrollado por la Organización de Estandarización Internacional (ISO) para apoyar el desarrollo y aplicación de protocolos de comunicación abiertos.

Según dicho modelo, los sistemas de comunicación de datos se describen como un modelo jerárquico que consiste en siete capas de funcionalidad diferente; cada capa de comunicación proporciona sus servicios a la capa inmediatamente superior y utiliza los servicios proporcionados por la capa inmediatamente inferior sin la necesidad de conocer ningún aspecto relacionado con las capas más inferiores, es decir que la funcionalidad de las capas inferiores está oculta y es transparente para las capas superiores.

1.2.1 Estructura del Modelo OSI

Tiene una estructura multinivel de manera que cada nivel (capa) resuelve solo una parte del problema de la comunicación, con funciones especificas.

Cada nivel se comunica con su homologo en los otros nodos, usando un mensaje a través de los niveles inferiores. La comunicación entre niveles se define de manera que un nivel N utilice los servicios del nivel N-1 y proporcione servicios al nivel N+1. Entre los diferentes niveles existen interfaces llamadas "puntos de acceso" a los servicios.

En cada nivel, se incorpora al mensaje un formato de control. Este elemento de control permite que un nivel en el receptor se entere de que el emisor le está enviando un mensaje con información.

Cualquier nivel puede incorporar un encabezado al mensaje. Por esta razón se considera que un mensaje está constituido por dos partes: el encabezado y la información

Los elementos activos que se encuentran en cada una de las capas (software o hardware) toman el nombre de entidades y cuando estas se encuentran en la misma capa se las conoce como entidades pares. Dichas entidades pares se comunican entre sí siguiendo una serie de reglas conocidas como “protocolos par a par”; dentro de estas reglas se utiliza los SAP o puntos de acceso a los servicios de una capa inferior, los cuales poseen una dirección que los identifica. Existe una IDU (unidad de datos de interface) que se la utiliza a través del SAP y que a su vez contiene una SDU (unidad de datos de servicio) además de alguna información de control, necesarias para que las capas inferiores realicen su trabajo.

Los servicios que proporcionan las capas entre sí son los siguientes:

Solicitud.- Cuando una capa N recibe una primitiva de solicitud envía información a través de la red a su homologo y cuando llega a su homologo (*equivalente*) de capa N, esta capa produce una primitiva de indicación.

Indicación.- Es un mensaje que va de una capa N a una capa N+1 en respuesta a una solicitud. Este aviso puede ser de una solicitud para establecer una comunicación. Al recibir la capa N+1 la indicación produce una respuesta.

Respuesta.- Es un mensaje que siempre va de la capa N+1 a la capa N y se genera a raíz de una solicitud.

Confirmación.- Es el mensaje que va de una capa N a una capa N+1 en consecuencia para confirmar una solicitud pedida previamente.

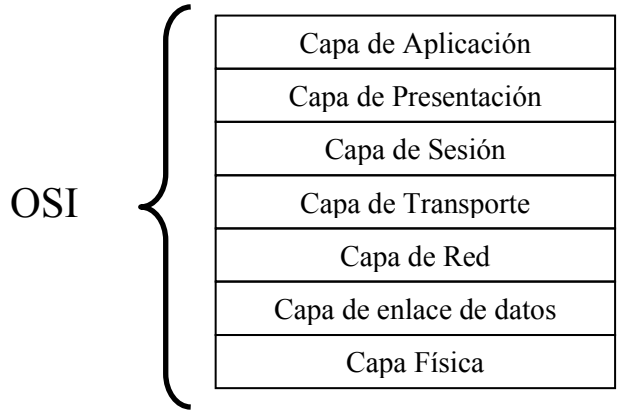


Figura. 1.1. Estructura del modelo OSI

El modelo OSI está constituido por 7 capas: Física, Enlace, Red, Transporte, Sesión, Presentación y Aplicación, y las funciones principales de cada una se describen a continuación:

1.2.1.1 Capa Física

La capa física abarca el conjunto físico propiamente dicho del que consta toda comunicación. Sus principales características son las siguientes:

Mecánicas: Relaciona las propiedades físicas de las interfaces utilizadas en el medio de transmisión. A veces, incluye la especificación de los conectores que unen los diferentes circuitos.

Eléctricas: Relaciona la representación de los bits en términos de niveles de tensión y la tasa de transmisión de datos. Maneja voltajes y pulsos eléctricos.

Funcional: Especifica las funciones realizadas por los circuitos individuales entre un sistema y el medio de transmisión.

De procedimiento: Especifica la secuencia de eventos por los que se intercambia un flujo de bits a través del medio físico.

1.2.1.2 Capa de Enlace de Datos

La capa de enlace de datos está encargada de realizar el enlace físico seguro y proporciona las definiciones para la transmisión de datos entre los nodos de una red y los medios de activación y desactivación del enlace.

El principal servicio proporcionado por la capa de enlace de datos es el de detección de errores y control.

Así con un protocolo que utilice una capa de enlace de datos completamente operacional, la capa adyacente superior puede suponer transmisión libre de errores en el enlace.

Otra característica muy importante de la capa de enlace de datos es el Control y Acceso al Medio conocido como MAC el cual es necesario en caso de que dos o más nodos deseen acceder al bus al mismo tiempo, el MAC se encarga de evitar éstas posibles colisiones ya sea impidiendo problemas en el paso de bits o reconociendo conflictos de acceso de tal forma que el problema se resuelva sin que ningún dato quede perdido.

1.2.1.3 Capa de Red

La capa de red controla la comunicación lógica a través de los sistemas de comunicación físicos mediante algún tipo de red de comunicación.

Libera a las capas superiores de la necesidad de tener conocimiento sobre la transmisión de datos subyacente y las tecnologías de conmutación utilizadas para conectar los sistemas.

En esta capa, el sistema está envuelto en un diálogo con la red para especificar la dirección de destino y solicitar ciertas facilidades de la red, como por ejemplo el tipo de prioridad.

Esta capa está encargada de establecer una red de conmutación de circuitos o de conmutación de paquetes y en muchos casos requiere el uso de técnicas de interconexión entre redes. Los paquetes individuales pueden llegar en un orden que no es idéntico a la sucesión de la transmisión original; por lo tanto, la capa de la red sólo es responsable del transporte apropiado de cada paquete. La capa de transporte de la estación receptora pondrá los paquetes recibidos de nuevo en el orden secuencial correcto.

1.2.1.4 Capa de Transporte

La capa de transporte proporciona un mecanismo para intercambiar datos entre sistemas finales. El servicio de transporte orientado a conexión asegura que los datos se entregan libres de errores, en secuencia y sin pérdidas o duplicados.

Si el intercambio de datos entre dos aplicaciones tuviera un dominio de 200 bytes; dado que las capas 1 y 2 del protocolo CAN pueden transmitir paquetes de 8 bytes únicamente; la capa de transporte estaría encargada de cortar el dominio en paquetes más pequeños antes de transmitirlos. En el lado receptor los paquetes se unirían para conformar el dominio de datos original.

La capa de transporte puede estar relacionada con la optimización del uso de los servicios de red y proporcionar una calidad del servicio solicitada. El tamaño y la complejidad del protocolo de transporte dependen de que tan seguras o inseguras sean las redes y sus servicios.

1.2.1.5 Capa de Sesión

La capa de sesión proporciona mecanismos para controlar el diálogo entre aplicaciones en sistemas finales. En muchos casos habrá poca o ninguna necesidad de la capa de sesión,

pero para algunas aplicaciones, estos servicios si se utilizan. Los principales servicios proporcionados por la capa de sesión son los siguientes:

Disciplina de Diálogo:

- Simultánea en dos sentidos o fullduplex.
- Alternada en los dos sentidos o semi-duplex.

Agrupamiento:

- El flujo de información se puede marcar para definir grupos de datos por ejemplo inicio y finalización de cuentas.

Recuperación:

- La capa de sesión puede proporcionar un mecanismo de puntos de comprobación, de forma que, si ocurre algún tipo de fallo entre puntos de comprobación, la entidad de sesión puede retransmitir todos los datos desde el último punto de comprobación.

1.2.1.6 Capa de Presentación

La capa de presentación define el formato de los datos que se van a intercambiar entre las aplicaciones y ofrece a los programas de aplicación un conjunto de servicios de transformación de datos.

La capa de presentación define la sintaxis utilizada entre entidades de aplicación y proporciona los medios para la selección y las subsecuentes modificaciones de la representación utilizada.

Algunos ejemplos de los servicios específicos que se podrían realizar en esa capa son los de compresión, descompresión y encriptado de datos.

1.2.1.7 Capa de Aplicación

La capa de aplicación proporciona un medio a los programas de aplicación para que accedan al entorno OSI.

Esta capa contiene funciones de administración y generalmente mecanismos útiles para admitir aplicaciones distribuidas. Además, se considera que residen en esta capa las aplicaciones de uso general como: transferencia de ficheros, correo electrónico y acceso terminal a computadores remotos.

Uno de los protocolos más conocidos para los sistemas de automatización es el MAP o Protocolo de Automatización Industrial, el cual proporciona servicios de lectura y escritura de variables, carga de programa y configuración de datos, o comienzo y finalización de programas. En el caso de CAN existen protocolos de aplicación como el CanOpen o el DeviceNet los cuales serán analizados en el capítulo 5.

1.2.2 El Protocolo CAN y el Modelo OSI

Debido a que el Modelo de Referencia OSI mantiene un armazón para cualquier tipo de sistema de comunicación complejo, algunos sistemas no utilizan necesariamente las siete capas del modelo. Las capas no utilizadas en cualquier protocolo se denominan capas nulas. Generalmente, algunos servicios de las capas del modelo OSI como por ejemplo el establecimiento de conexiones encima de varias redes parciales, no es pertinente para la automatización industrial o los sistemas de comunicación de automotores como CAN.

Por consiguiente sólo tres capas (capa física, capa de enlace de datos, capa de aplicación) son las pertinentes para la comunicación de datos en protocolos para automatización industrial como se muestra en la figura 1.2.

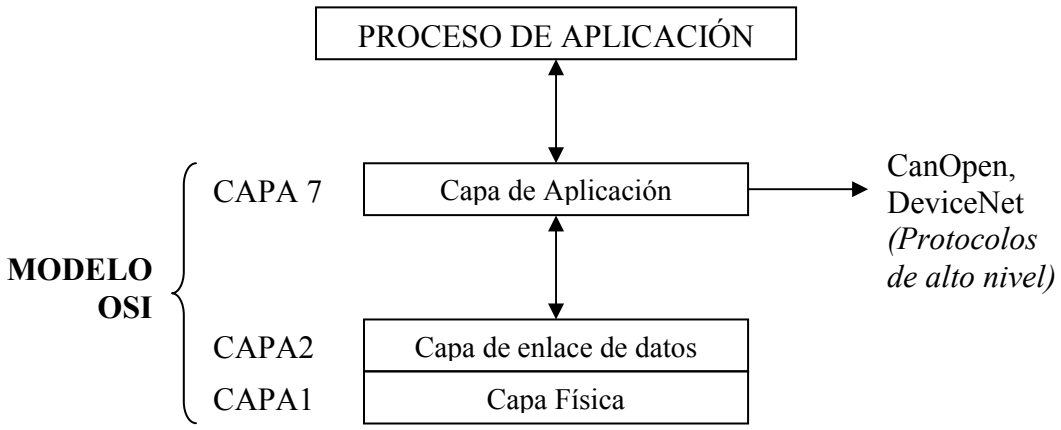


Figura. 1.2. Capas del Modelo OSI que intervienen en CAN

Existen protocolos como CanOpen encargados de proporcionar aplicaciones para el protocolo CAN (capa 7), pero CAN como tal, utiliza las capas 1 y 2 del modelo OSI.

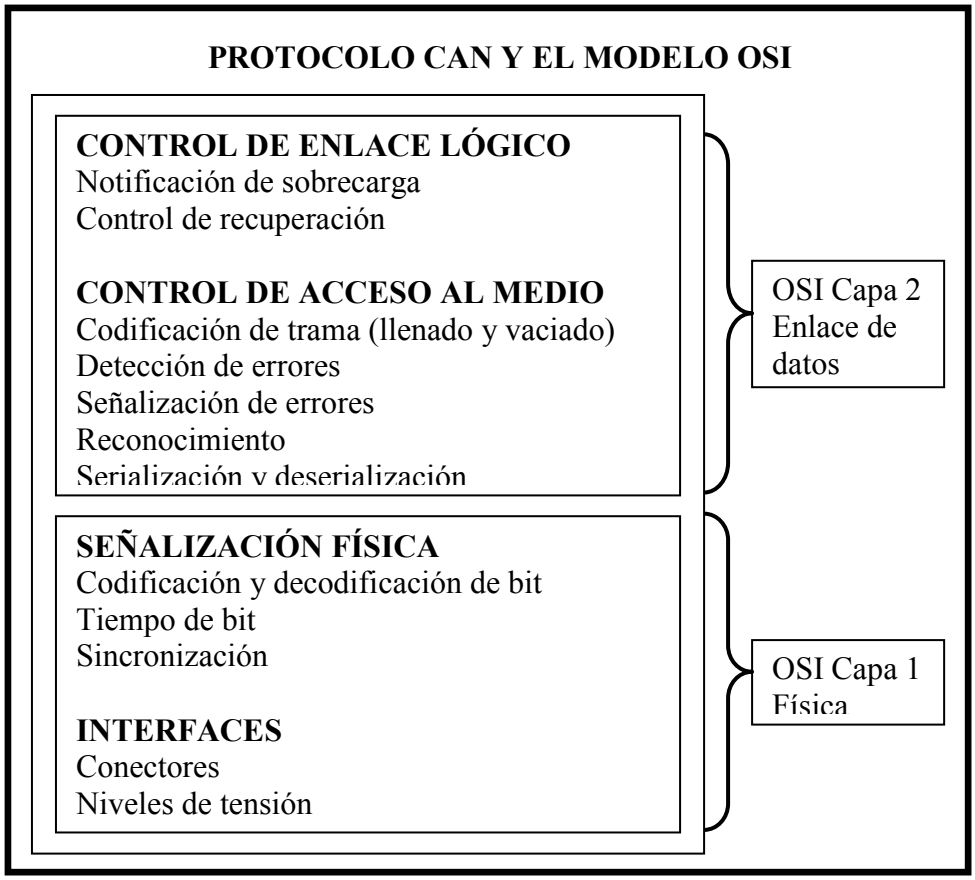


Figura. 1.3. Características de las capas 1 y 2 del Modelo OSI aplicadas a CAN

La funcionalidad de estas 2 capas se detalla en las especificaciones CAN dadas principalmente por las normas ISO 11898. Por lo general toda la capa 2 o de enlace de datos y la parte superior de la capa 1 incluyendo la Señalización Física están implementados en un controlador de comunicación. La parte inferior de la capa 1 o física se encuentra implementada en otro chip llamado transceiver.

La separación en dos circuitos integrados se da debido a que las funciones que involucran medios físicos y medios lógicos son tan diferentes que las tecnologías actuales no han sido capaces de integrar su funcionalidad entera en un solo chip.

Entre las características técnicas del transceiver se encuentra el control de altos voltajes y corrientes mientras que el controlador se encarga de operar a 5 Voltios constantes, mantener bajas las corrientes involucradas en los circuitos lógicos, etc.

1.3 MODELOS CLIENTE/SERVIDOR Y PRODUCTOR/CONSUMIDOR

El modelo cliente/servidor describe siempre una relación de comunicación uno-uno entre el cliente y el servidor mientras que el modelo productor/consumidor describe una comunicación uno-varios entre un productor y uno o varios consumidores.

La mayoría de sistemas de comunicaciones modernos utilizados en la actualidad para automatización industrial soportan estos dos modelos de comunicación.

En la figura 1.4. se muestra la interacción existente en el modelo cliente/servidor donde un proceso N (cliente) requiere un servicio específico mediante una primitiva de *requerimiento*. Dicha primitiva de requerimiento es indicada al proceso M mediante una primitiva de *indicación*. Una vez ejecutado el requerimiento, el servidor muestra el resultado mediante una primitiva de *respuesta.* Para concluir la transmisión, el cliente es informado de una ejecución exitosa mediante una primitiva de confirmación.

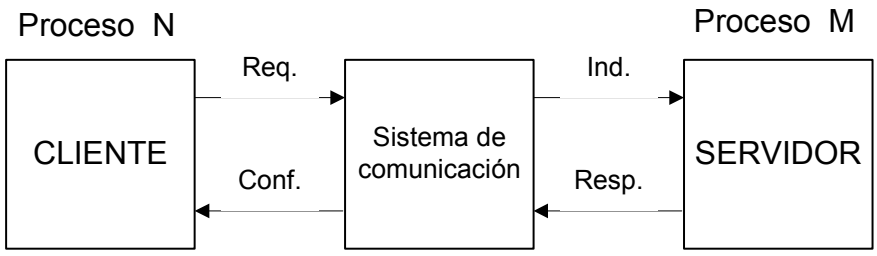


Figura. 1.4. Interacción del Modelo Cliente/Servidor

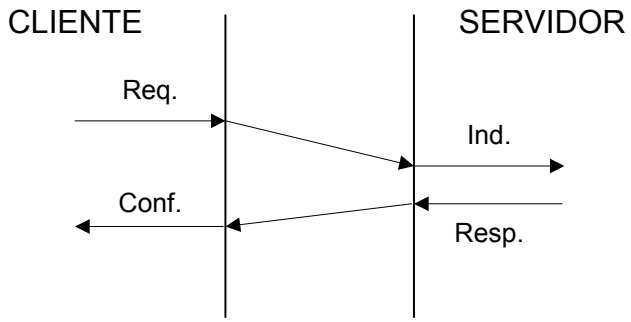


Figura. 1.5. Diagrama de secuencia de un servicio confirmado cliente/servidor

Los servicios de comunicación mediante los cuales se transfieren datos en forma de mensajes multicast o broadcast forman parte del modelo Productor/Consumidor.

En la figura 1.6. se muestra el diagrama de secuencia del modelo productor/consumidor, en el cual los servicios que brinda el productor son ofrecidos a otros procesos mediante una primitiva de indicación, los otros procesos deciden si aceptan o no los servicios ofrecidos y si los aceptan no existe ninguna primitiva de confirmación.

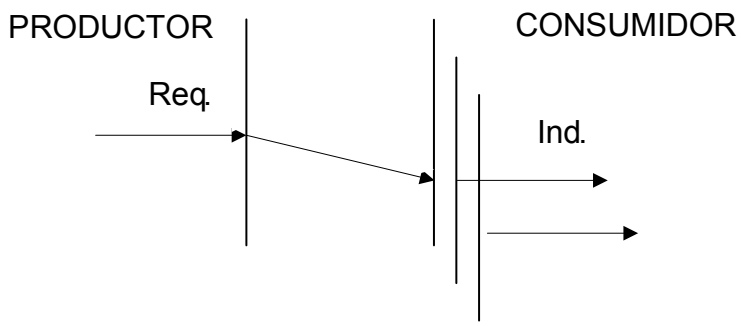


Figura. 1.6. Diagrama de secuencia de un servicio productor/consumidor

1.4 CLASIFICACIÓN DE PROTOCOLOS SEGÚN SU TIPO DE ORIENTACIÓN

1.4.1 Protocolos orientados a nodos

En este tipo de protocolos el intercambio de datos se basa en un direccionamiento de nodos, generalmente la trama de datos transmitidos por el medio de comunicación contiene la información específica y a veces la dirección del nodo involucrado. De esta forma la trama es enviada a un nodo o a un grupo de nodos específico.

Se reservan direcciones especiales para la transmisión de tramas que son dirigidas a un grupo de nodos o a todos los nodos, esto toma el nombre de broadcasting

Casi todos los sistemas de comunicación de datos convencionales están basados en el principio de direccionamiento de nodos.

1.4.2 Protocolos orientados a mensajes

En estos protocolos el intercambio de datos en cambio se basa en identificadores de mensajes, es decir que un mensaje transmitido por un nodo se lo reconoce mediante un único identificador específico.

De ésta forma es decisión únicamente de los nodos si desean aceptar el mensaje, ya sea un solo nodo, varios nodos o ningún nodo, por esto la transmisión de datos orientada a mensajes representa un principio diferente a la transmisión orientada a nodos.

Como el transmisor del mensaje normalmente no conoce quien es el receptor, la confirmación del mensaje transmitido no se da adecuadamente, por esta razón los protocolos orientados a mensajes utilizan el método de señalización de error en lugar del método de reconocimiento, los dos métodos se explicarán en la sección 1.6, que trata del control y verificación de errores.

1.5 CONTROL DE ACCESO AL MEDIO MAC

El Control de acceso al medio (MAC) se usa para arbitrar qué transmisor en una red gana el acceso a los medios de comunicación de la transmisión. Además impide que dos o más nodos intenten transmitir datos al mismo tiempo, de esta forma las señales de una transmisión no interfieren entre sí.

MAC define el tipo de señalización utilizada en la línea de comunicación, que permite a múltiples dispositivos compartir el mismo cable, comunicarse sobre él y no interferir con otro dispositivo.

El control de acceso al medio en el que un bus de campo se basa determina muchas características del sistema como su comportamiento en tiempo real. Por esta razón, el tipo de control de acceso al medio puede ser un criterio firme para la selección de un protocolo de comunicación.

El MAC generalmente puede ser clasificado en dos: Control de Acceso al Medio determinístico y Control de Acceso al Medio no controlado o arbitrario. El acceso determinístico se distingue de dos maneras: el derecho de acceso al bus se asigna por una entidad central (Maestro) o descentralizadamente mediante un acuerdo entre nodos pasando un token o señal de nodo a nodo. El acceso arbitrario se caracteriza

principalmente por que cualquier nodo puede acceder en cualquier momento siempre que el bus se encuentre libre.

Métodos Determinísticos

Con un acceso al bus determinístico se define claramente que solo un nodo puede acceder al bus, esto implica que no existen colisiones provocadas por el acceso simultáneo de más de un nodo, además los métodos de control determinísticos ya sean centralizados o descentralizados utilizan una habilitación muy simple para la reacción del sistema.

El mayor inconveniente de los sistemas controlados centralizadamente es que si la entidad central (*maestro*) falla, el sistema completo también fallará.

La realización de métodos de acceso determinados descentralizados es más compleja, pero estos métodos son más flexibles y generalmente permanecen operables incluso si un nodo falla o se desactiva.

Métodos Arbitrarios

Con los métodos de acceso al bus arbitrarios, los nodos pueden acceder al bus cuando se encuentre inactivo es decir varios nodos pueden requerir un acceso simultáneo al bus, este método se llama Acceso Múltiple con Detección de Portadora (CSMA).

Los diferentes métodos de acceso aleatorio al bus son clasificados en dos: Métodos con colisiones y Métodos sin colisiones y se los aplica dependiendo si una colisión de mensajes transmitidos puede ocurrir o no.

El método en el cual existen colisiones de mensajes pero pueden ser detectadas se llama CSMA/CD (Collision Detect - Detección de Colisión); y el método en que se minimizan las colisiones o se evitan por completo se llama CSMA/CA (Collision Avoidance – Anulación de colisión).

Durante un acceso al bus completamente libre de colisiones, los nodos descubren el acceso simultáneo al bus antes de la transmisión de los mensajes actuales durante una fase de arbitraje. Basado en las prioridades del mensaje, sólo el nodo que debe transmitir el mensaje de prioridad más alta permanece en el bus.

En un sistema de comunicación arbitrario, pero no en el libre de colisión, sólo se detecta el acceso simultáneo al bus después de una longitud específica de la trama. Después de ser detectada es señalada por el nodo involucrado y el conflicto es resuelto por una apropiada estrategia de seguridad estadística.

El protocolo CAN se ha desarrollado como un protocolo con acceso al bus arbitrario completamente libre de colisiones y basado en el funcionamiento mediante mensajes prioritarios, éste concepto se estudiará detalladamente como "Arbitraje de Bus" en el capítulo 2 sección 2.1.

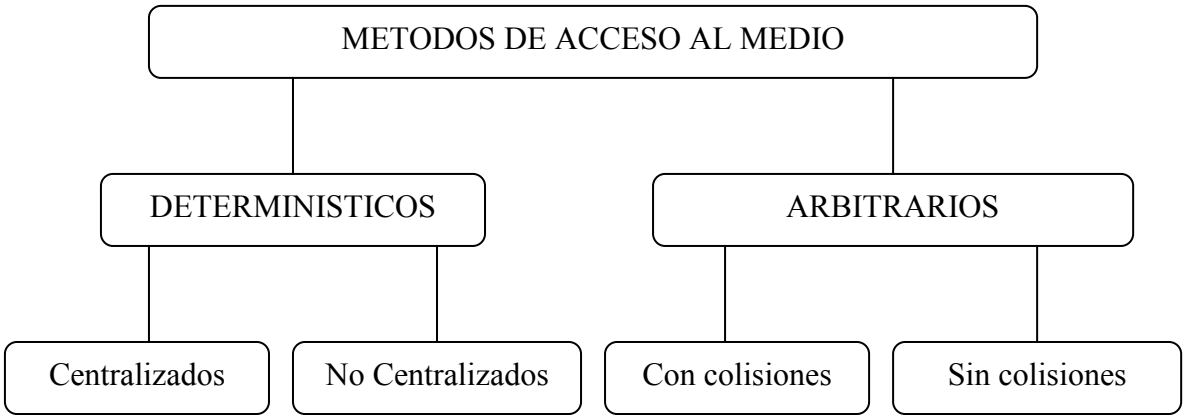


Figura. 1.7. Clasificación de los métodos de Acceso al Medio

Los siguientes métodos de acceso al bus son los más utilizados en los sistemas de comunicación de datos aplicados al dominio de campo:

- MAESTRO ESCLAVO

- DELEGATED TOKING

- TOKEN PASSING
- TDMA
- CSMA/CD
- CSMA/CA

1.5.1 Maestro-Esclavo

El acceso al bus está controlado por una entidad central llamado maestro el cual imparte permisos de transmisión a los diferentes esclavos.

En una topología en anillo, para que un nodo pueda transmitir debe recibir permiso del nodo central o maestro a través de un mensaje de sondeo. Este permiso va pasando secuencialmente de estación en estación a lo largo de todo el anillo. Cada estación puede transmitir cuando recibe el permiso y encuentra el anillo vacío. Al finalizar su transmisión pasa el permiso a la estación siguiente. El inconveniente de esta técnica reside en la necesidad de que la comunicación entre dos nodos pase por la estación central.

En la topología en bus se halla la técnica de sondeo en la que por turnos, la estación central pregunta a cada estación si tiene necesidad de transmitir información. En caso afirmativo, la estación preguntada recibe permiso para transmitir, de tal forma que la información pasará por la estación central para, desde ella, dirigirse a la estación destino.

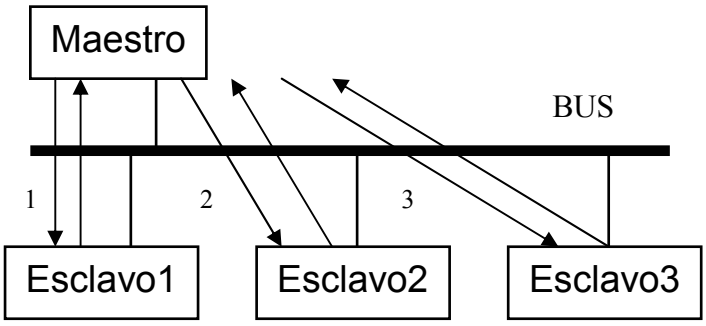


Figura. 1.8. Configuración Maestro-esclavo

Además se trata de un método en el que pueden incluirse prioridades con facilidad, haciendo que las estaciones con mayor prioridad sean interrogadas con mayor frecuencia que el resto. Estas técnicas centralizadas tienen el inconveniente de que, si falla la estación central, toda la red se queda sin servicio. Pueden aparecer turnos de espera excesivamente largos si el número de estaciones es grande.

1.5.2. Delegated Token

Según este principio, una entidad central (maestro) induce a los nodos responsables de transmitir algún mensaje requerido a hacerlo en un horario de tiempo predefinido. Los mensajes transmitidos pueden aceptarse entonces por otros nodos interesados.

Como se muestra en la figura 1.9. el árbitro del bus (B) delega el derecho de acceso al bus a los nodos transmitiendo una señal (token) de demanda de mensaje; hacia todos los nodos; la cual es idéntica al identificador del mensaje requerido. El nodo que posee la información requerida (Tx) la envía hacia todos los nodos.

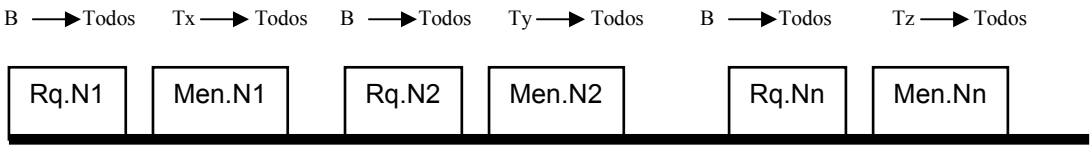


Figura. 1.9. Secuencia de mensajes con acceso al bus delegated token

Este tipo de sistema puede considerarse como una base de datos distribuida con un número específico de variables. Después de que una variable requerida por el árbitro del bus se envía por un nodo, el derecho de acceso al bus queda nuevamente en posición del árbitro del bus.

Este sistema de comunicación es un sistema distribuido de mensajes centralizado, en el que cada nodo puede proporcionar mensajes a los otros nodos formando un sistema de comunicación todos con todos como se muestra en la figura 1.10.

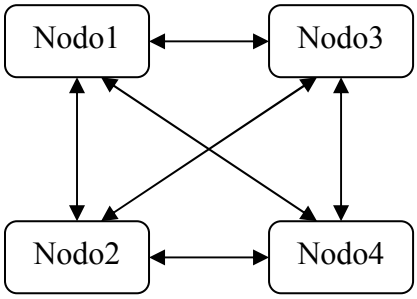


Figura. 1.10. Estructura de comunicación todos con todos

1.5.3. Token Passing

De acuerdo con este principio, el derecho de acceso al bus se traslada de estación a estación en forma de un mensaje específico (token) lo que también se conoce como paso de testigo, tan pronto como una estación ha recibido dicho mensaje utiliza el bus para realizar un ciclo de intercambio de mensajes con otras estaciones por un periodo limitado de tiempo, todas las estaciones de la red intervienen en la circulación del paquete de información. Esta técnica se puede aplicar a topologías de red en anillo (Token ring) o en bus (Token bus).

Token Ring

Se basa en una pequeña trama o testigo que como ya se indicó circula a lo largo del anillo. Se utiliza un bit que indica el estado del anillo (libre u ocupado) y cuando ninguna estación está transmitiendo, el testigo simplemente circula por el anillo pasando de una estación a la siguiente. Cuando una estación desea transmitir, espera a recibir el testigo modificando el bit de estado del anillo de libre a ocupado e inserta a continuación la información a enviar junto con su propia dirección y la de la estación destino. El paquete de datos circula por el anillo hasta llegar a la estación receptora que copia su contenido y lo vuelve a poner en circulación incluyendo una marca de recepción, de tal forma que, cuando vuelve a llegar a la estación emisora, ésta lo retira de la red y genera un nuevo testigo libre. La principal ventaja de esta técnica es su alto rendimiento. Su adaptabilidad a la topología en anillo y el ser un método de transmisión unidireccional hacen que su eficiencia sea mayor que las redes que utilizan el método de pase de testigo en bus. Entre sus desventajas

se encuentran la disminución de la eficiencia en caso de baja carga del sistema debido a la circulación del testigo y los procesos de mantenimiento de la red.

Token Bus

En esta técnica, las estaciones del bus o árbol forman un anillo lógico, es decir, a las estaciones se les asigna una posición lógica en una secuencia ordenada y circular. Cada estación conoce la identidad de su estación antecesora y de su sucesora dentro del anillo lógico. En este caso, la ordenación física es totalmente independiente e irrelevante a la ordenación lógica. La estación que recibe el testigo tiene garantizado el derecho de acceso al medio de transmisión de tal forma que al finalizar su transmisión o terminar el tiempo asignado, pasará el testigo a la siguiente estación en la secuencia lógica. Las redes con pase de testigo proporcionan mejores posibilidades de gestión de red.

1.5.4. Acceso Múltiple por división de tiempo (TDMA)

Mediante este método, se proporcionan ranuras de tiempo fijo de acceso al bus para cada nodo y al contrario que en token passing, los nodos no son llamados mediante un testigo (token) sino que acceden al bus en instantes de tiempo predefinidos para cada uno, el prerequisito para este tipo de comunicación es un alto sincronismo entre la referencia de tiempo local y los nodos individuales. En la figura 1.11. se muestra este concepto, cada nodo tiene un instante de tiempo "t" para transmitir su información y por lo general el tiempo es siempre el mismo.

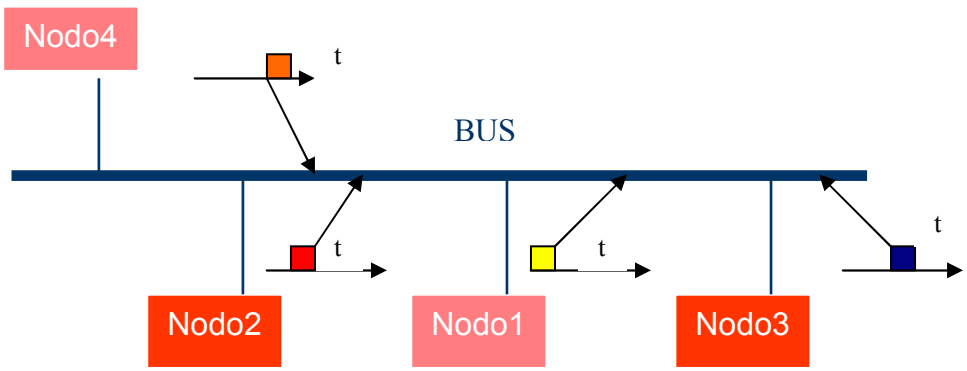


Figura. 1.11. Acceso Múltiple por división del tiempo

Esta técnica es más utilizada en el campo de las telecomunicaciones vía satélite y no ha sido muy tomada para sistemas de comunicación industrial de bus de campo.

1.5.5. Acceso Múltiple con Detección de Portadora (CSMA)

Se basa en que un nodo puede acceder al bus cuando se encuentre inactivo, el proceso es particularmente ventajoso para la comunicación de mensajes porque en este caso una transmisión de la trama no se inicializa a menos que sea necesario.

Las estaciones solicitan transmitir de forma aleatoria y no existe control para determinar el orden de transmisión, por tanto, todas las estaciones deben competir por el derecho al acceso. La técnica de acceso aleatorio múltiple con detección de portadora es el método más usado en las topologías en bus y árbol. Esta técnica consiste en que una estación que desee transmitir, primero escucha el medio para determinar si hay otra transmisión en proceso. Si es así, la estación tras un período de espera lo vuelve a intentar hasta que encuentre el medio de transmisión libre.

Cuando el medio está libre, la estación emisora vuelve a retransmitir la información hasta que la confirmación se produzca. Uno de los motivos porque una información no llegue a su destino puede ser la aparición de colisiones debido a que varias estaciones solicitan transmitir al mismo tiempo luego de que el bus quede libre.

1.5.5.1.- Acceso Múltiple con Detección de Portadora y Detección de Colisiones (CSMA/CD)

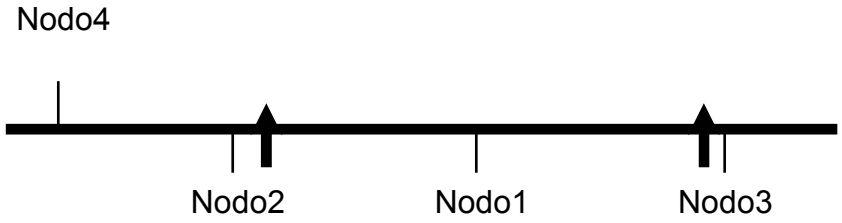


Figura. 1.12. Acceso Múltiple con Detección de Portadora y Detección de Colisiones

Es una técnica mejorada consiste en controlar la aparición de colisiones añadiendo a la técnica CSMA las siguientes reglas:

- Si se detecta una colisión, se cesa inmediatamente la transmisión y se envía una pequeña trama de consenso de colisión (JAM) para asegurar que todas las estaciones se han enterado de la existencia de la colisión.

- Tras esperar un determinado período de tiempo, se vuelve a intentar la transmisión usando CSMA.

Las colisiones están generalmente resueltas introduciendo los períodos de espera aleatorios antes de intentar la retransmisión, como el éxito de un acceso al bus retransmitido tampoco puede garantizarse, este método puede ser problemático para las aplicaciones críticas con respecto al tiempo.

En la figura 1.13. se ilustra el principio de CSMA/CD cuando dos nodos quieren acceder al bus simultáneamente.

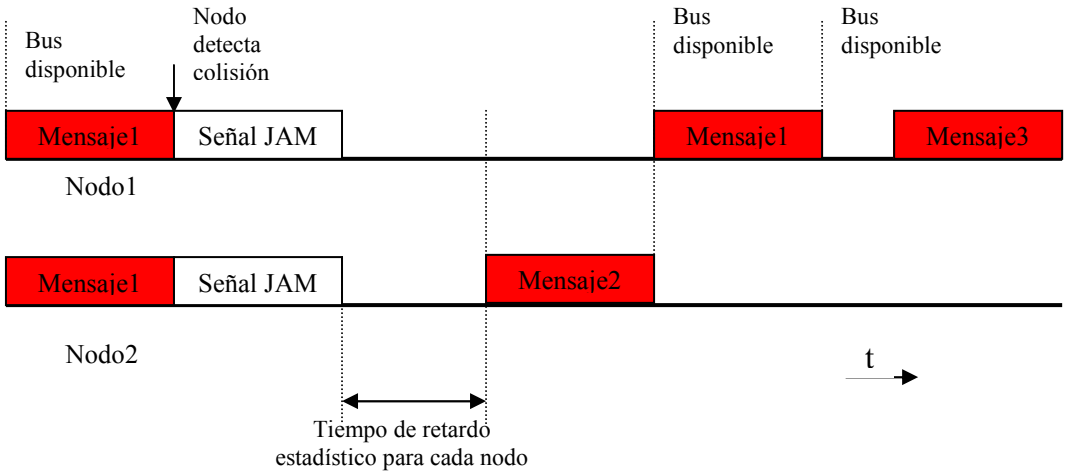


Figura. 1.13. Principio utilizado por el acceso al bus CSMA/CD

El nodo 1 es el primero en detectar la colisión por lo cual transmite una señal llamada “señal de jam”. Ambos nodos se retiran del bus y respectivamente esperan un tiempo aleatorio estadístico hasta que un acceso al bus vuelva a empezar.

En el ejemplo mostrado, el tiempo de espera del nodo 2 es más corto que el tiempo de espera del nodo 1 de tal forma que el nodo 2 pueda ocupar el bus sin perturbaciones ocasionadas por el nodo 1.

En la figura 1.14. se muestra un diagrama de flujo sobre el funcionamiento de CSMA/CD

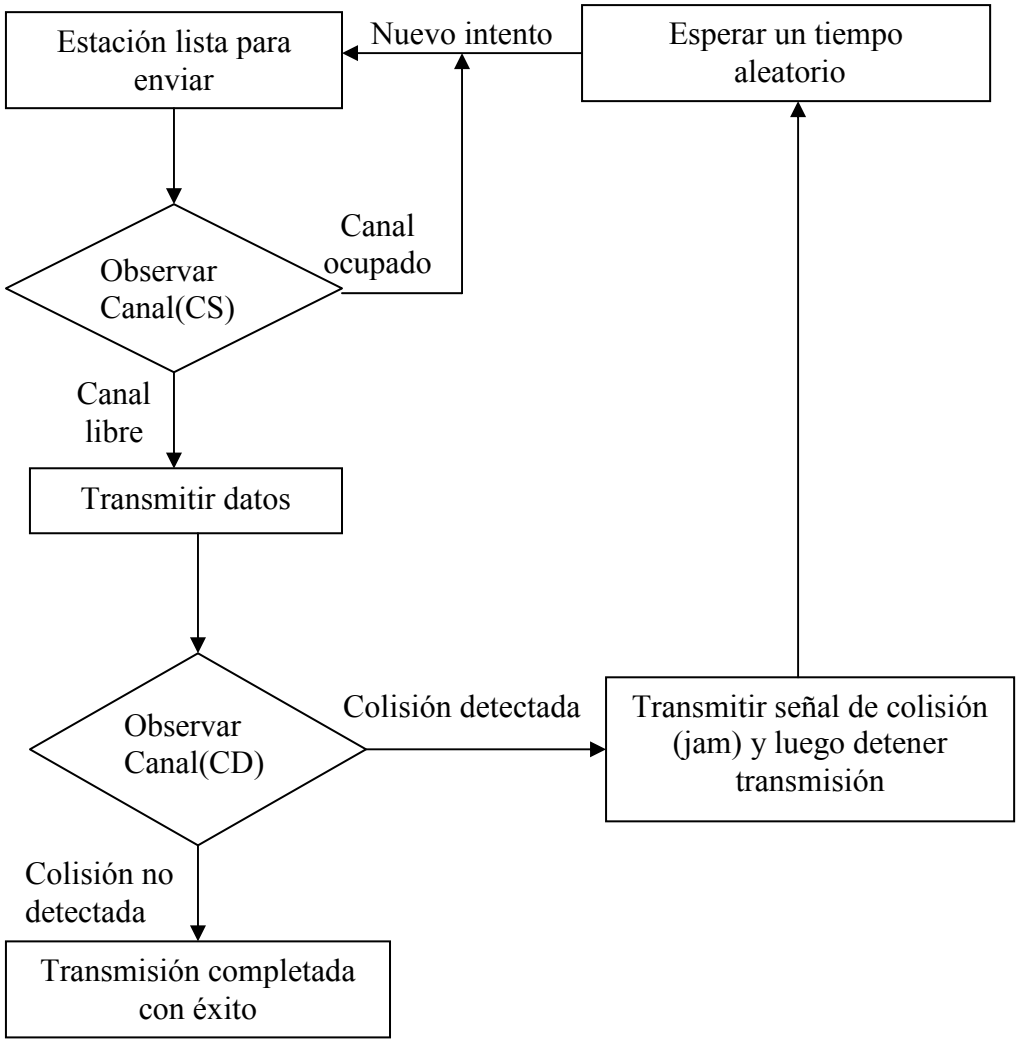


Figura. 1.14. Funcionamiento CSMA/CD

1.5.5.2.- Acceso Múltiple con Detección de Portadora y Anulación de Colisiones (CSMA/CA)

Al contrario que el método CSMA/CD, en CSMA/CA se anula cualquier colisión de mensajes. A pesar de que también es posible que inicialmente varios nodos empiecen transmitiendo un mensaje simultáneamente, se asegura durante una fase de arbitraje que sólo un nodo, que contiene el mensaje de mayor prioridad, será el que se mantiene en el bus hasta el final de la fase.

Durante la fase de arbitraje, los nodos que querían transmitir simultáneamente transmiten los identificadores del mensaje bit por bit en el bus y verifican para cada bit transmitido si el nivel de prioridad presente en el bus empareja el nivel de prioridad de ellos. Si éste no es el caso, entonces el nodo sabe que otro nodo ha empezado la transmisión de un mensaje de prioridad más alto simultáneamente.

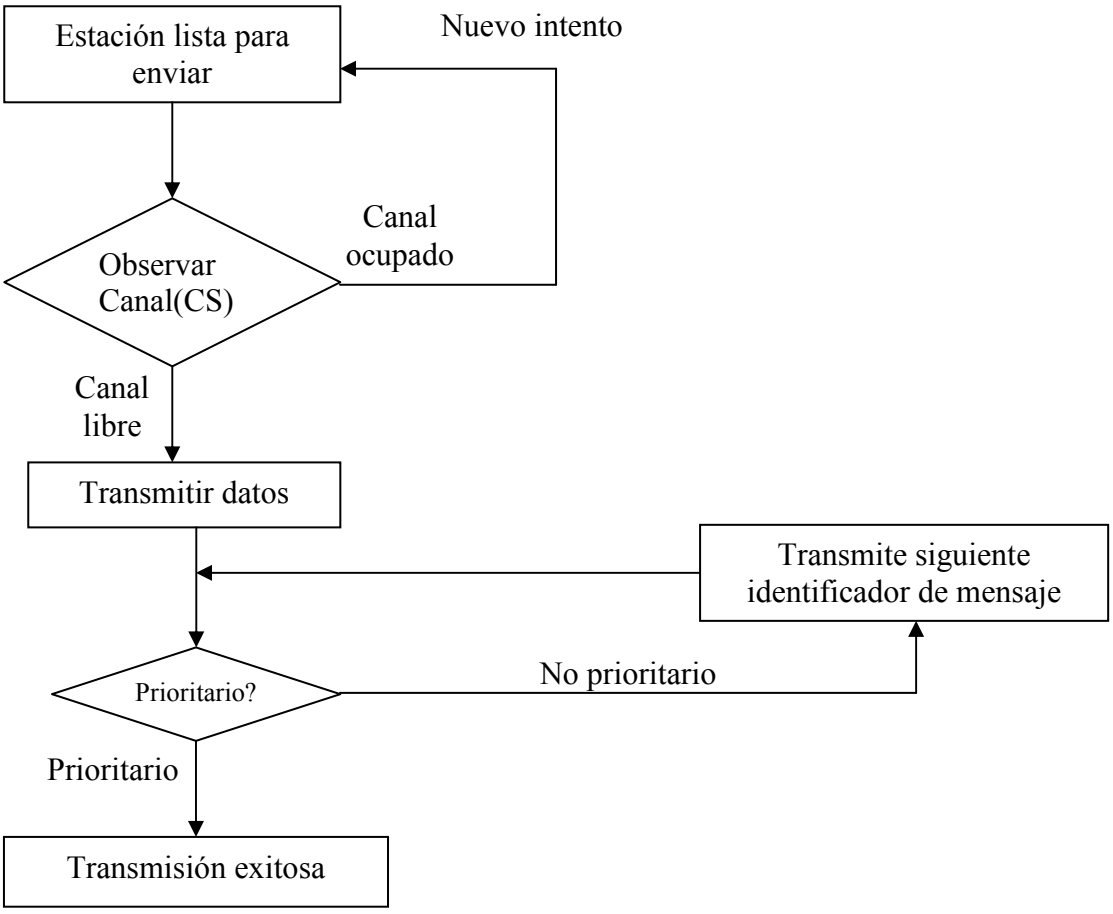


Figura. 1.15. Funcionamiento CSMA/CA

1.6 CONTROL Y VERIFICACIÓN DE ERRORES

Durante una transmisión de señales digitales puede existir anomalías o errores en uno o varios bits de la información transmitida, esto puede ocurrir debido a varias causas como la interferencia de señales inductivas y capacitivas o variaciones de voltajes entre el transmisor y el receptor.

El grado de probabilidad de que ocurran errores está dado por el ambiente electromagnético del sistema de transmisión de datos, las distancias del conexionado y las medidas de protección utilizadas.

Los errores en una transmisión de datos no pueden ser totalmente prevenidos por lo cual se ha desarrollado técnicas para que en cada nodo se verifique si la información recibida es correcta, y en caso de existir errores detectarlos y posteriormente corregirlos.

Los métodos utilizados comúnmente para detección de errores se basan en la transmisión adicional de información en la trama de datos, la cual permite al receptor darse cuenta si la información es correcta o errada con un alto porcentaje de credibilidad para esto es necesario que el receptor envíe una señal al transmisor cuando la información es errada para que este la reenvíe.

Figura. 1.16. Principio básico de verificación de información

La trama de verificación de información enviada por el transmisor es generada de acuerdo a reglas específicas, el receptor aplica las mismas reglas a los datos recibidos y posteriormente lo compara con la trama de verificación de información, de esta manera el

receptor puede detectar errores en la transmisión si la secuencia recibida difiere de la secuencia calculada.

Existen varios tipos de tramas o secuencias de verificación de datos, debido a que no todas garantizan una transmisión completamente libre de errores, cada una con diferente grado de complejidad, por ejemplo la técnica más sencilla es la "verificación de paridad" la cual solo puede detectar un número impar de bits errados

Una referencia para la capacidad de detección de errores es conocida como la distancia Hamming. Esta medida indica la mínima cantidad de bits errados que se pueden detectar sin ninguna posibilidad de que alguno de ellos se pase por alto, una distancia Hamming de cuatro significa que se van a detectar tres bits errados por mensaje fiablemente, para una aplicación de bus de campo se requiere una distancia Hamming de al menos cuatro según cálculos realizados a partir de hechos experimentales.

El mecanismo de control de errores más importante en los sistemas de comunicación serial es el "Código de Redundancia Cíclica" (CRC). En este mecanismo se consideran a los bits a ser transmitidos en una trama de datos como coeficientes binarios de un polinomio con un grado apropiado. Este polinomio se divide por un Generador Polinomial de un grado específico conocido por el receptor, el residuo de esta división es transmitido como una secuencia de verificación de datos (FCS- Frame Check Sequence). El receptor lleva a cabo la misma operación y compara el resultado con la FCS recibida; si las dos FCS son idénticas no existe error en la transmisión de lo contrario la trama recibida es errónea. Otras técnicas de control de error como por ejemplo la suma de todos los bytes consecutivos de una trama de datos, son menos efectivas.

Generalmente, el largo de la trama de verificación de datos transmitida adicionalmente depende de la longitud de la trama de datos, desafortunadamente, la longitud de dicha trama reduce la eficiencia del protocolo debido a que la información enviada adicionalmente es información redundante. Los métodos aplicados para el control de error dependen básicamente de si el protocolo está orientado a nodos o a mensajes, cuando el protocolo está orientado a nodos el método toma el nombre de Control de Error Pasivo y

cuando está orientado a mensajes toma el nombre de Señalización de Error, que como se verá en la próxima sección, es el caso de CAN.

1.6.1 Control de Error Pasivo

En este método, en seguida de la transmisión de un mensaje, el transmisor espera una respuesta o reconocimiento por parte del receptor por un periodo de tiempo máximo especificado, si el receptor detecta un error en la transmisión, ignora el mensaje recibido y no envía dicho reconocimiento. La desventaja de este tipo de Control es que el transmisor tiene que esperar el periodo de tiempo máximo en caso de que exista un error, además no se puede enviar un mensaje que vaya dirigido a un grupo de nodos porque se necesitaría varios reconocimientos. Este principio es utilizado por PROFIBUS, AS-I INTERBUS.

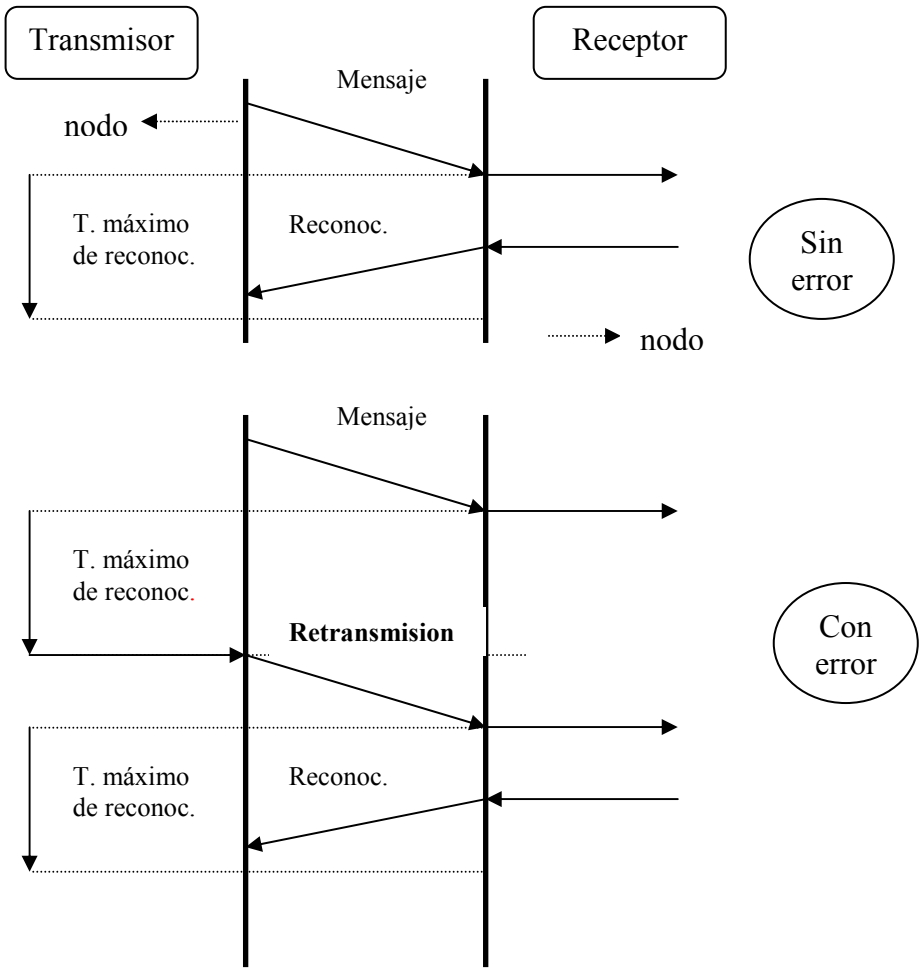


Figura. 1.17. Principio de Control de error pasivo

1.6.2 Señalización de Error

Como ya se dijo los protocolos orientados a mensajes utilizan este principio, en este caso el receptor indica la recepción de un mensaje erróneo transmitiendo una señal de error específica, cuando el transmisor recibe dicha señal automáticamente empieza la retransmisión del mensaje. Con este método es posible obtener tiempos de recuperación muy cortos. La ventaja de este método es que todos los nodos reciben la señal de error, este método es utilizado por el protocolo CAN.

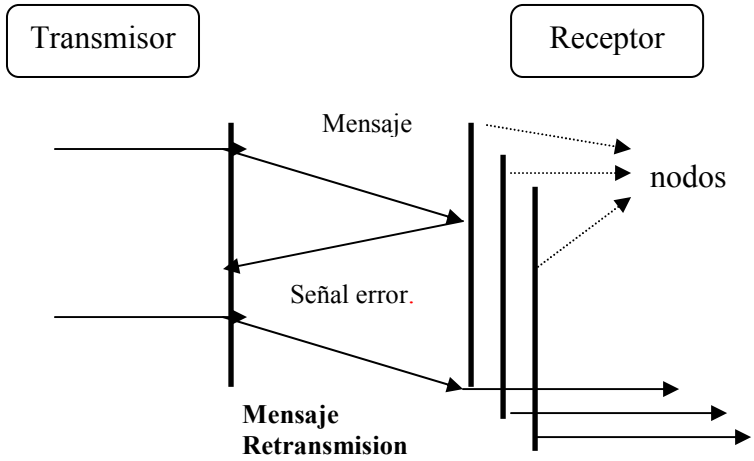


Figura. 1.18. Principio de Señalización de Error

1.7 TOPOLOGÍAS DE RED

Una de las características importantes sobre las redes en un sistema de comunicación de datos como CAN es su topología; en ella se describe la estructura física de conexión entre los diferentes nodos de la red.

Estrella: Todos los nodos son conectados a un nodo central mediante conexiones punto a punto

Anillo: Se caracteriza por una conexión punto a punto direccionada formando una cadena cerrada

Bus: Todos los nodos tienen una conexión común al mismo medio

Árbol: Se da cuando de un elemento o nodo se ramifican nuevos nodos

TOPOLOGÍA	VENTAJAS	DESVENTAJAS
ESTRELLA	Cada nodo tiene su propia conexión al nodo central Integración simple de nuevos nodos Fácil diseño con medios de transmisión ópticos	Longitud larga de conexiones si los nodos están en línea geográfica El nodo central requiere N interfaces para N nodos La comunicación se da solo mediante el nodo central y si este falla no puede existir comunicación
ANILLO	Redes extensas debido a que cada nodo provee regeneración de señal Excelente para medios de transmisión ópticos Identificación simple de nodos de acuerdo a su posición en el anillo	El sistema total falla si uno de los nodos falla La operación del sistema se interrumpe si se debe integrar un nuevo nodo
BUS **(utilizada por CAN)**	Costos de cableado bajos Conexión simple de nodo La falla de un nodo no provoca falla en otros nodos	Tiene una longitud de bus y numero de nodos limitados si no se usan repetidores Dificulta la implementación con medios ópticos Requiere identificadores de nodos
ARBOL	Excelente adaptación a los requerimientos geográficos. Costos de instalación y cableado mínimos.	Por lo general se necesitan elementos activos para la ubicación de las ramas

Tabla. 1.1. Ventajas y desventajas de las principales topologías de red

1.8 CARACTERÍSTICAS PRINCIPALES DEL PROTOCOLO CAN

Implementaciones de Hardware CAN

Existen dos implementaciones, estas permiten administrar el uso del bus en función de las necesidades de cada nodo.

BASIC CAN: En esta implementación existe una intervención del Controlador Host para tratar con los mensajes del bus CAN. Cada nodo transmitirá tan solo cuando se produzca un evento en alguna de las señales que lo conciernen. El almacenamiento de mensajes en transmisión y recepción se da mediante una pequeña cantidad de buffers dedicados (ya sea a transmisión o a recepción).

FULL CAN: En este tipo de implementación, no es necesaria la interrupción del Controlador Host para llevar a cabo la recepción y transmisión de mensajes, de esta manera se reduce la carga al microcontrolador. Se lo utiliza en aplicaciones con alta exigencia en cuanto a frecuencia de actualización y seguridad. El almacenamiento de mensajes en transmisión y recepción se da mediante una mayor cantidad de buffers programables ya sea como receptores o transmisores.

Las características de estos dos tipos de implementaciones se detallarán en el capítulo 4 sección 4.2.1.

Especificaciones CAN (CAN 2.0A y CAN 2.0B)

Existen dos especificaciones para CAN las cuales están más orientadas hacia los requisitos de fabricación de controladores CAN. La diferencia entre estas dos especificaciones se encuentra principalmente en el formato del identificador de mensajes. CAN2.0A o protocolo CAN estándar define sistemas CAN con 11 bits identificadores (2048 mensajes) y CAN 2.0B o protocolo CAN extendido especifica una nueva trama que incluye 29 bits identificadores (512 millones de mensajes) además de la versión estándar. (Capítulo 2 sección 2.7).

Las tramas de los mensajes son los elementos básicos de transmisión y van de un nodo emisor a uno o varios nodos receptores. Las tramas se dividen en siete campos diferentes, cada uno de ellos con una función específica. Dichos campos de bits y tipos de trama se detallan en el capítulo 2 sección 2.2.

Topología, Velocidad de Transmisión, Numero de Nodos

CAN se basa en una topología de bus lineal, mediante repetidores o routers se puede establecer una topología de árbol o estructuras jerárquicas y aumentar el número de nodos.

El número de nodos que se pueden manejar dependen de la capacidad de los controladores empleados, no está definido por el protocolo; pero normalmente son 32 nodos con manejadores de línea estándar.

La velocidad de transmisión está entre 5Kbps y 1 Mbps y está en relación a la distancia de cobertura de la red como se analiza en el Capítulo 3 sección 3.2.5.

Protocolo orientado a mensajes

El protocolo CAN, como se vio en la sección 1.2.4, está basado en la identificación de mensajes transmitidos mediante un identificador de mensajes mas no por direccionamiento del receptor del mensaje, de esta manera todos los nodos verifican si el mensaje transmitido es pertinente o no para cada uno de ellos por lo tanto un mensaje puede ser aceptado por ninguno, uno, varios (broadcasting) o por todos los nodos.

Prioridad de mensajes

El identificador del mensaje simultáneamente determina su prioridad con respecto al acceso al bus, es decir el mensaje tendrá acceso de acuerdo a su importancia.

Este método resuelve las situaciones de carga excesiva del bus, ya que asegura que los mensajes particularmente importantes obtengan acceso al bus a pesar de que este se encuentre en fases de capacidad de transporte limitada; esta ventaja se consigue en muy pocos sistemas de comunicación de datos.

Capacidad Multimaestra

El acceso al bus no está controlado por una única entidad central o maestro ya que cualquier nodo puede transmitir un mensaje en el momento en que encuentre libre al bus y en caso de que varios nodos tengan que transmitir un mensaje al mismo tiempo, el derecho de acceso es para el nodo que tenga el mensaje de mayor prioridad. Este método produce una carga promedio del bus mucho más baja que otros sistemas de acceso al bus de tipo determinísticos.

Control de acceso al medio CSMA/CR (colisión resolution) y arbitraje de bus libre de pérdida

Como el acceso al bus en el protocolo CAN toma lugar aleatoriamente es posible que varios nodos quieran acceder al mismo tiempo al bus; en otros sistemas aleatorios de acceso al bus, los mensajes transmitidos son destruidos y la solución del conflicto requiere de un nuevo esfuerzo por acceder al bus basado en una conveniente estrategia.

El acceso al bus CAN se denomina CSMA/CR (arbitraje no destructivo). Es un método de acceso en el cual se garantiza que el mensaje con mayor prioridad tomará ventaja del bus y que al existir mensajes simultáneos no sea necesario a ningún momento la destrucción de mensajes.

El arbitraje de bus del protocolo CAN y la prioridad de mensajes se detalla en el Capitulo 2 sección 2.1.

CR o Resolución de colisión se basa en una topología eléctrica que aplica una función lógica determinista a cada bit, que se resuelve con la prioridad del nivel definido como bit de tipo dominante.

El bit dominante se define como el equivalente al valor lógico '0'

El bit recesivo se define como el equivalente al valor lógico '1'

El arbitraje de bus trata de una función AND de todos los bits transmitidos simultáneamente con el nivel actual del bus; cada transmisor escucha continuamente el nivel presente en el bus, y se retira cuando ese nivel no coincide con el que dicho transmisor envía. Mientras hay coincidencia en este nivel la transmisión continúa, finalmente el mensaje con identificador de máxima prioridad sobrevive y los demás nodos reintentarán la transmisión lo antes posible.

El nodo que encuentra libre el bus inicia transmitiendo 1 bit de inicio, 11 bits identificadores para CAN V2.0A o 29 bits para CAN 2.0B llevando el nombre y la prioridad del mensaje, luego 6 bits de control, de 0 a 8 bytes de datos, 16 bits CRC, 2 bits de reconocimiento (ACK) y finalmente 7 bits de fin de trama y 3 bits Inter-trama. En el Capítulo dos se detallan los diferentes campos pertenecientes a la trama de datos CAN.

TRAMA ESTÁNDAR Y EXTENDIDA

SOF(1) Start of Frame	ID (11) Identi-fiers	CONT(6) Control	DATOS (0-64) Data	CRC(16) Cyclic redundancy code	ACK(2) Acknow-ledgment	EOF(10) End of Frame

SOF(1)	**ID (29)**	CONT(6)	DATOS (0-64)	CRC(16)	ACK(2)	EOF(10)

Figura. 1.19. Tramas Estándar y Extendida de CAN

Longitud corta de trama de datos

La longitud máxima de datos en un mensaje CAN está limitada a 8 bytes, esta longitud es suficiente para los requerimientos de comunicación de datos en vehículos y a nivel de entrada/salida en sistemas de automatización. En estos campos de aplicación, la transmisión de tramas de datos más largas no se requiere tanto como el soporte de una alta tasa de transmisión con mensajes cortos.

La transmisión de tramas de datos largas en el protocolo CAN se da únicamente en casos excepcionales como en la configuración o parametrización de nodos o en la carga de códigos de programa, pero los protocolos de aplicación como CanOpen o DeviceNet poseen servicios de transmisión fragmentada para estos casos.

La transmisión de tramas de mensajes cortas es particularmente importante cuando se da en ambientes pesados propensos a distorsiones, ya que la probabilidad de una coincidencia de transmisión de mensajes con una perturbación es proporcional a la longitud del mensaje transmitido; de esta forma, la corta longitud de mensajes en el protocolo CAN permiten que la transmisión se de incluso en ambientes electromagnéticos difíciles con alto índice de distorsión.

Alta integridad de datos y corto tiempo de recuperación

CAN proporciona diversos mecanismos de detección de errores logrando una alta probabilidad de detección y corrección de mensajes erróneos, si se detecta un error automáticamente se origina una retransmisión; y en oposición a los protocolos orientados a nodos, la detección y señalización de errores se dan en un tiempo muy corto.

Detección y desactivación de nodos defectuosos

El protocolo CAN cuenta con un monitoreo continuo de los nodos y de sus funciones específicas, cuando un nodo excede una tasa de error promedio predefinida perturbando la comunicación continua de los datos, se toman medidas para que se limite el acceso al nodo afectado o se lo desconecte de la red para poder corregirlo.

Tiempo de latencia corto para los mensajes prioritarios

El esquema de acceso al bus CAN, su pequeña longitud máxima de mensajes y su principio de prioridad de mensajes permiten que se obtenga un tiempo de latencia muy corto para los mensajes con prioridades altas. El tiempo máximo de latencia para el mensaje de mayor prioridad en una red CAN con mensajes de 8 bytes es de 130 bit times.

También pueden asegurarse los tiempos de latencia definidos por mensajes prioritarios extensos, tomando medidas adicionales en un protocolo sobrepuesto a la capa de la aplicación.

Codificación de bit

La codificación de bit en CAN se consigue mediante la técnica NRZ "No Return to Zero" y se detalla en el Capítulo 3.

Sincronización de bit

La sincronización en la comunicación entre los nodos de la red se consigue mediante el método conocido como bit stuffing (relleno de bit). El protocolo CAN garantiza que después de 5 bits subsecuentes en la trama, se produce un cambio en el flujo de bits transmitidos dando inicio a una nueva resincronización del generador de reloj de los nodos.

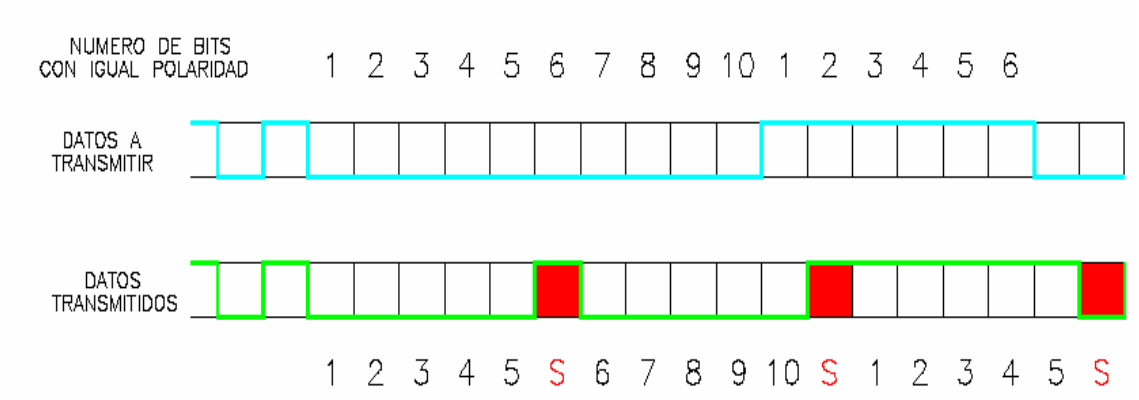


Figura. 1.20. El principio del Bit Stuffing

Cuando se han transmitido 5 bits consecutivos de un mismo nivel (0 o 1) se añade un sexto bit con el nivel opuesto a la cadena de salida. El receptor eliminará este sexto bit al recibirlo.

El método de sincronización y resincronización de bits se detalla en el Capítulo 3 sección 3.2.

Características de línea conductora:

CAN proporciona varios modos y características eléctricas para manejar las líneas de comunicación:

Modo diferencial o balanceado: Esta técnica requiere de dos líneas de señal para la comunicación y una para tierra. Las líneas se manejan con señales complementarias, el signo de la diferencia de las señales en ambas líneas determina el valor lógico del bit ya sea “1” o “0”. Esta técnica proporciona alta inmunidad al ruido.

Modo desbalanceado (+ NRZ): Esta técnica sólo requiere una línea de señal y una de tierra. Este modo es muy susceptible al ruido y se utiliza para costos y velocidades muy bajas.

Modo de comunicación DC: Esta técnica se utiliza en la comunicación acoplada por transformadores

Sincronización entre nodos

Después de una transmisión y recepción satisfactoria de un mensaje, se puede dar una interrupción en todos los nodos participantes al mismo tiempo. Esta interrupción puede ser aprovechada para reajustar los relojes de la aplicación en el ambiente de control distribuido. La sincronización de los nodos es muy importante en aplicaciones de control en tiempo real.

Detección y corrección de errores a nivel de mensajes

La poderosa técnica de tratamiento de errores utilizada en CAN se basa en el mecanismo CRC de 15 bits con una distancia Hamming de 6, asegurando la detección de 5 bits erróneos por mensaje. Además existen extensos mecanismos de detección de errores construidos dentro de CAN, como detección de error de forma, detección de error read-after-write y chequeo de paridad de bits.

La corrección del error se consigue mediante una retransmisión del mensaje erróneo luego de que el nodo detector del mensaje haya enviado por el bus una trama de error hacia todos los otros nodos.

La estandarización internacional

Las normas internacionales ISO-WS 11898 partes 1, 2 y 3 especifican a CAN como un protocolo de capas 1 y 2 para la comunicación interna de vehículos para aplicaciones de velocidades bajas y altas.

En la norma ISO98-1 se especifica un plan de prueba de confirmación y cumplimiento de la norma ISO-WS 11898 para cualquier aplicación de redes CAN

Las normas que especifican protocolos de alto nivel para aplicaciones industriales en general están dadas como protocolos de capa 7 por CAL (CAN Aplication Layer) en [CIA-200], por CanOpen en [CIA30x-40x] basada en CAL y por ODVA en [DNSP98] para DeviceNet

Productos existentes

En la actualidad CAN se ha convertido en una tecnología habitual en la industria, y numerosas firmas como INTEL, MOTOROLA, PHILIPS, SIEMENS, NSC, etc, fabrican y distribuyen productos compatibles con este protocolo de comunicaciones.

Entre los productos existentes se tienen:

- Controladores CAN, que gestionan las comunicaciones a través de este protocolo.

- Módulos CAN integrados en el mismo chip del microcontrolador. Existen versiones CAN con los microcontroladores más populares del mercado.

- Controladores CAN independientes que permiten a microcontroladores no incluidos en la anterior categoría comunicarse a través del CAN.

- Tarjetas de conexión con PCs.

- Software y herramientas diversas de monitorización de sistemas CAN, útiles tanto en la fase de diseño y simulación como en la de prueba.

CAPITULO 2

CAPA DE ENLACE DE DATOS DEL PROTOCOLO CAN

Como ya se explicó anteriormente la capa de enlace de datos según el modelo de referencia OSI es la encargada de realizar el control de acceso al medio, el enlace lógico de datos y los métodos de confinamiento y detección de errores.

2.1 CONTROL DE ACCESO AL MEDIO CAN Y ARBITRAJE DEL BUS

CAN utiliza un método de Acceso al Bus arbitrario es decir que cualquier nodo accede al bus cuando este se encuentra libre; como la transmisión de nodos se da de una forma asincrónica pueden existir varios nodos que empiecen una transmisión de una trama simultáneamente.

CAN trabaja mediante dos estados lógicos: recesivo y dominante. El estado recesivo es semejante al estado lógico “1” y el estado dominante es semejante al estado lógico “0”.

El estado del bus CAN se define mediante una operación idéntica a la de una compuerta AND como se muestra en la figura 2.1.

De esta forma se explica que el bus está libre solo cuando ningún nodo está transmitiendo.

NODO			BUS
1	2	3	
D	D	D	**D**
D	D	R	**D**
D	R	D	**D**
D	R	R	**D**
R	D	D	**D**
R	D	R	**D**
R	R	D	**D**
R	R	R	**R**

Bus libre

"1" = R = bit recesivo (nodo no transmisor)

"0" = D = bit dominante (bus ocupado por un nodo transmisor)

Figura. 2.1. Estados lógicos del bus CAN

Para que un nodo inicie una transmisión debe poner un bit de inicio de trama en estado dominante (SOF o Start of Frame).

Si un nodo quiere iniciar una transmisión de una trama y el bus está ocupado debe esperar a que finalice el "campo de intermisión" del nodo que está transmitiendo en ese momento, lo que indicará que el bus se encuentra nuevamente libre. El campo de intermisión se describe más adelante en la sección 2.2.5.

Si el bus se encuentra libre pero existen varios nodos que quieren acceder simultáneamente se inicia la fase de arbitraje del bus en la cual se transmiten uno a uno los bits del identificador de mensaje de todos los nodos desde el bit más significativo y se comparan con el nivel actual del bus; el identificador de mensaje consta de 11 bits en la versión estándar o 29 bits en la versión extendida.

Cuando un nodo transmite un bit recesivo y al mismo tiempo monitorea un bit dominante en el bus detiene su transmisión actuando solo como nodo receptor ya que entiende que un mensaje con mayor prioridad ha accedido al bus y se queda en espera de que el bus se encuentre nuevamente libre.

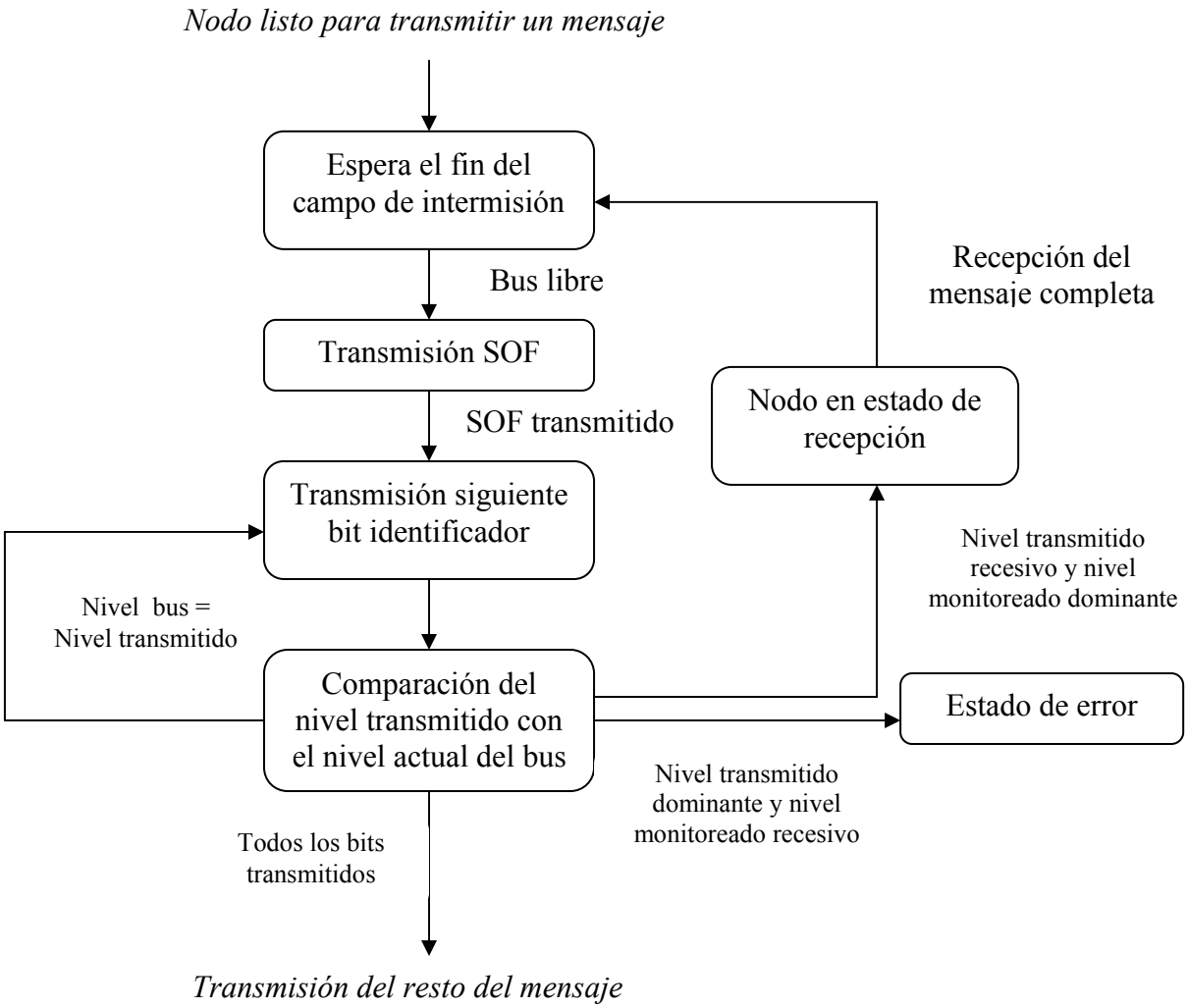


Figura. 2.2. Proceso de arbitraje del bus CAN

El nodo que gana el acceso al bus se convierte en el nuevo maestro del sistema hasta que otro nodo gane el acceso al bus, mediante esta fase de arbitraje de bus se asegura que solamente un nodo pueda estar actuando como transmisor en el bus y que a su vez sea el nodo que contiene el mensaje de mayor prioridad.

La figura 2.3. muestra el proceso de arbitraje del bus CAN cuando tres nodos requieren transmitir información al mismo tiempo.

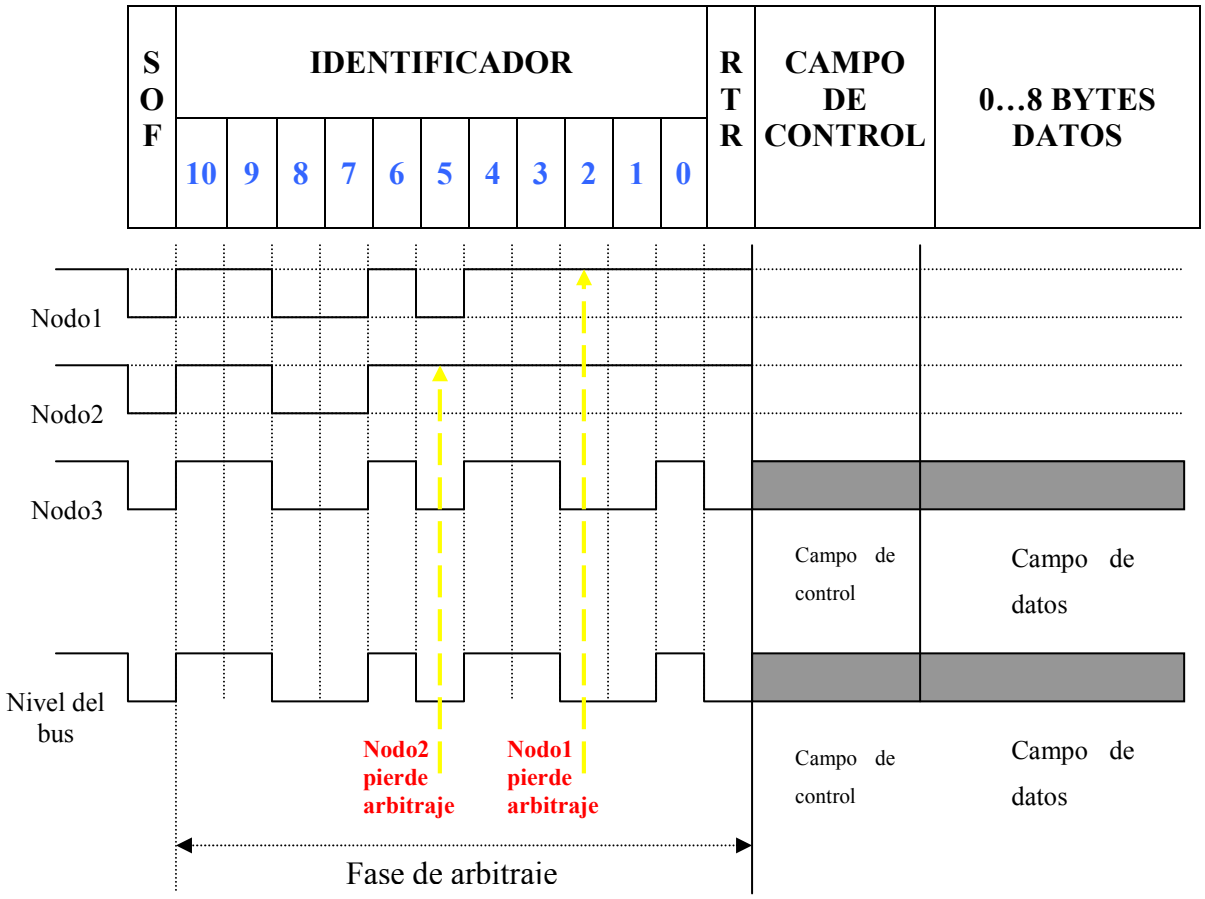


Figura. 2.3. Ejemplo de proceso de arbitraje del bus CAN con 3 nodos simultáneos

En caso de que un nodo desee transmitir una trama de datos y al mismo tiempo un nodo realice una transmisión de una trama remota (requerimiento de datos, sección 2.2) pero los dos del mismo mensaje entonces la colisión no se puede resolver únicamente mediante el identificador del mensaje ya que es el mismo para los dos nodos, por esta razón existe el bit RTR “Remote Transmisión Request Frame” el cual se define como dominante para una trama de datos y recesivo para una trama remota de esta manera se resuelve la colisión.

2.2 TIPOS DE TRAMAS EN EL PROTOCOLO CAN

El protocolo CAN define 4 tipos de tramas cada una con sus funciones y características específicas.

Trama de Datos: Esta trama contiene datos específicos, los cuales se envían desde un transmisor hacia uno o varios receptores; incluye un bit RTR en estado dominante para diferenciarla de una trama remota y es la trama que más circula en una red CAN.

Trama Remota: Es la trama utilizada por un nodo para solicitar una trama de datos determinada y contiene el identificador (ID) del mensaje en cuestión, además se debe especificar exactamente la longitud del campo de datos del mensaje que se va a recibir.

La única diferencia con la trama de datos es su bit RTR que se encuentra en estado recesivo.

Trama de Error: Este tipo de tramas son enviadas por los nodos que detecten un error en la transmisión o recepción de un mensaje y se genera solo durante la transmisión de dicho mensaje, la trama correspondiente al mensaje es invalidada y posteriormente retransmitida.

Trama de sobrecarga: La trama de sobrecarga se puede generar solamente en el espacio entre tramas lo que la diferencia de la trama de error. La utilizan los nodos para pedir un tiempo adicional antes del inicio de la siguiente trama, un nodo puede generar un máximo de 2 tramas de sobrecarga.

Adicionalmente a estas cuatro tramas se genera un espacio entre una trama de datos o remota y la precedente; este espacio está previsto para realizar procesamientos internos en los nodos antes del comienzo de la próxima trama, y toma el nombre de INTERTRAMA

Cada trama está constituida por diferentes campos de bits los cuales se detallan a continuación:

2.2.1 Trama de Datos

La trama de datos está constituida por 7 campos: Campo de inicio de trama (SOF), Campo de arbitraje, Campo de control, Campo de datos, Campo CRC, Campo de reconocimiento y Campo de fin de trama (EOF), como se muestra en la figura 2.4.

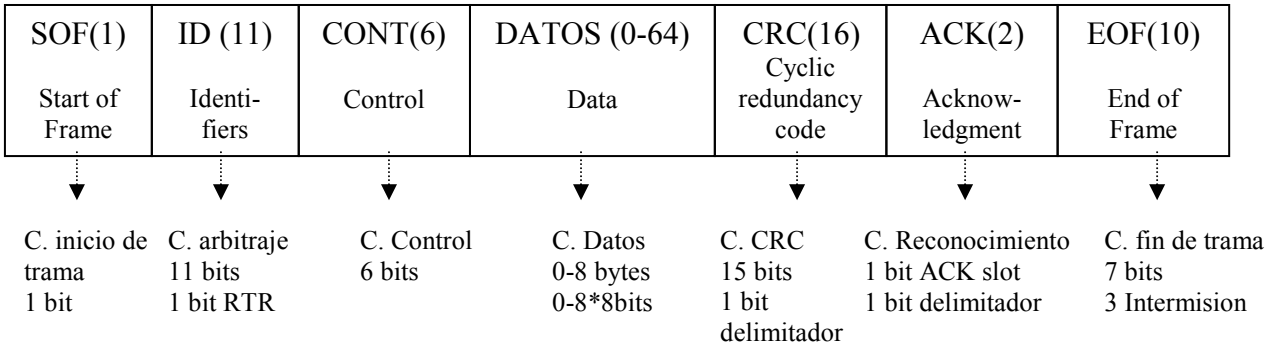


Figura. 2.4. Trama de Datos

2.2.1.1 Campo de inicio de trama

El campo de inicio de trama está constituido por un solo bit (SOF = Start of Frame) en estado dominante el cual indica el inicio de una trama de datos. Este bit puede ser transmitido cuando el bus se encuentre libre nuevamente en caso de que haya estado utilizado por otro nodo o cuando este desocupado.

2.2.1.2 Campo de arbitraje

El campo de arbitraje está constituido por un grupo de bits identificadores (ID) los cuales determinan la prioridad del mensaje y por un bit RTR el cual; según su estado; determina si una trama es de datos (0: estado dominante) o es una trama remota (1: estado recesivo).

En la versión estándar 2.0A existen 11 identificadores (2048 mensajes posibles) y en la versión extendida 2.0B existen 29 identificadores (512 millones de mensajes posibles), y se transmiten desde el bit más significativo es decir desde ID10 a ID0 o desde ID28 a ID0. El estado de estos bits determinan la prioridad del mensaje.

Si el bit RTR está en estado dominante “0” se trata de una trama de datos y si se encuentra en estado recesivo “1” se trata de una trama remota, esto debido a que una trama de datos tiene mayor prioridad que una trama remota.

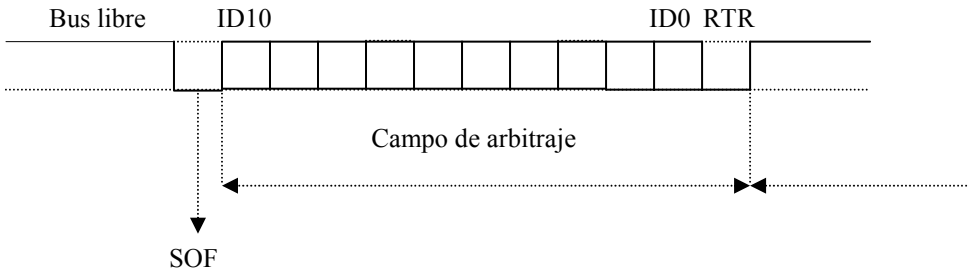


Figura. 2.5. Campo de arbitraje de una trama de Datos

2.2.1.3 Campo de Control

El campo de control está constituido por 6 bits, el primero (IDE = Identifier Extensión bit) distingue entre una trama base o una trama extendida, para una trama base este bit es dominante y para una extendida es recesivo.

El segundo bit (r0) está reservado para futuras extensiones del protocolo CAN y se transmite dominante mientras su función no se defina.

Los cuatro últimos bits (DLC3 – DLC0) se utilizan para indicar el numero de bytes que se van a transmitir en el campo de datos y puede variar de 0 a 8 bytes.

Se puede tener casos especiales en que se realizan transmisiones de más de 8 bytes para propósitos específicos.

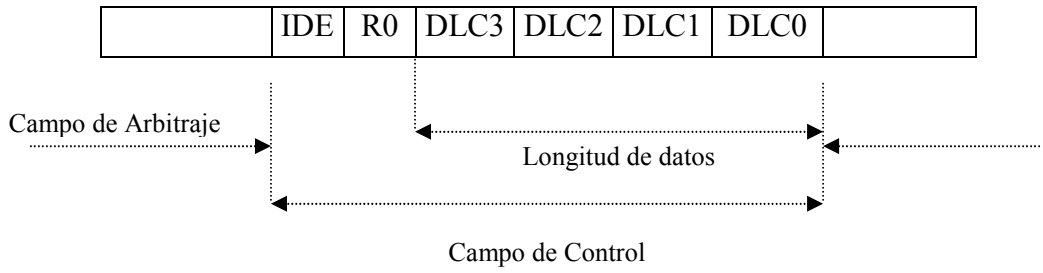


Figura. 2.6. Campo de Control de una trama de Datos

2.2.1.4 Campo de Datos

Este Campo se utiliza para transmitir los datos requeridos por algún nodo dentro de la trama CAN la longitud máxima es de 8 bytes y se transfieren desde el bit más significativo hasta el bit menos significativo.

2.2.1.5 Campo CRC (Código de Redundancia Cíclico)

El Campo del Código de Redundancia Cíclico está compuesto por una secuencia de 15 bits de comprobación, esta secuencia se obtiene mediante una operación polinomial que se detalla más adelante en la sección 2.5 de Detección de Error. Además consta de un bit delimitador en estado recesivo que, como su nombre lo indica, sirve para reconocer el fin del código CRC.

Por medio de la sucesión de bits CRC un receptor puede verificar si la trama de bits recibida es errónea debido a diferentes perturbaciones.

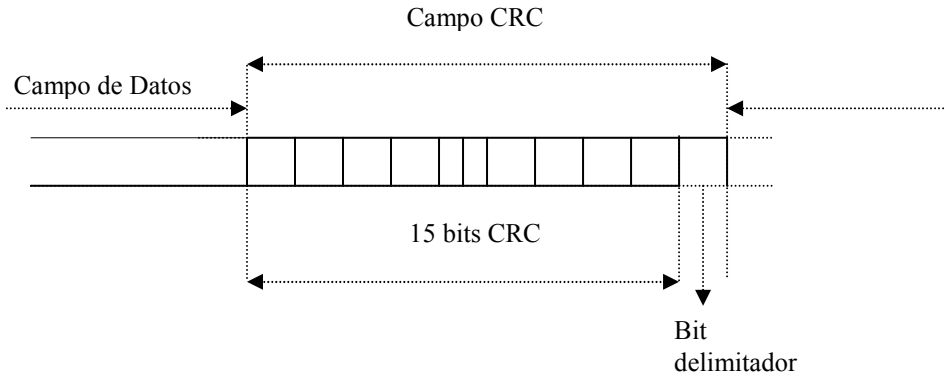


Figura. 2.7. Campo CRC

2.2.1.6 Campo de Reconocimiento ACK

Este campo está compuesto por dos bits llamados Slot o ranura de reconocimiento y un bit delimitador. Estos dos bits son enviados por un transmisor en estado recesivo.

El nodo transmisor de una trama espera un reconocimiento por parte de por lo menos un nodo receptor, de tal forma que, cualquier nodo que haya recibido correctamente la trama

lo informa al nodo transmisor, cambiando el bit recesivo enviado por dicho transmisor a un estado dominante en el Slot ACK.

Un bit dominante en el Slot ACK indica al transmisor que por lo menos uno de los nodos ha tenido una recepción correcta de la trama transmitida.

Cuando un nodo detecta un error en la transmisión mediante el campo CRC, es señalado mediante una trama de error que viene solo después del reconocimiento, esto quiere decir que un reconocimiento de recepción de mensaje no significa necesariamente una transmisión libre de error.

El bit delimitador siempre se encuentra en estado recesivo y viene a continuación del Slot ACK.

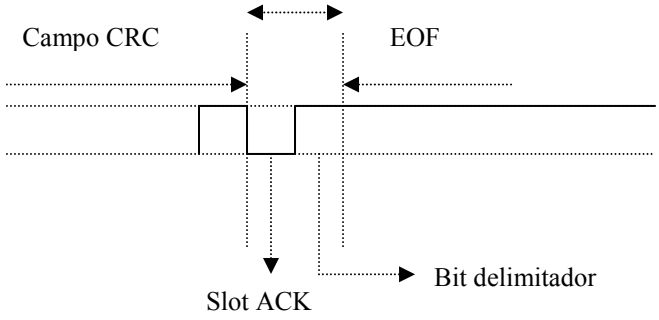


Figura. 2.8. Campo ACK

2.2.1.7 Campo de fin de trama (EOF)

Cada trama de datos y trama remota se delimitan por una sucesión de banderas de siete bits recesivos. Junto con el bit delimitador de ACK recesivo se produce una sucesión total de ocho bits recesivos al final de una trama de datos o remota.

Un nodo no sincronizado puede confundir a un bit dominante de reconocimiento ACK como un bit de SOF debido al nivel dominante de los dos, por esta razón se insertan 6 bits que generen un error de relleno e indiquen que el bit recibido fue un ACK. Además se

inserta un bit adicional ya que se puede dar un error local en un receptor a último momento. Estos siete bits unidos al delimitador ACK dan un total de ocho bits recesivos que indican una transmisión libre de error.

2.2.2 Trama Remota

Una trama remota es la responsable de realizar una petición de una trama de datos específica mediante el identificador correspondiente.

Solo un nodo es el responsable de transmitir un mensaje con un identificador específico pero uno o varios nodos pueden recibir el mensaje; la trama de datos enviada corresponde al mismo identificador del mensaje que la trama remota envió previamente.

La trama de datos y la trama remota tienen el mismo formato pero tienen dos diferencias fundamentales:

La primera diferencia se da por el bit RTR (RTR dominante corresponde a una trama de datos y RTR recesivo corresponde a una trama remota). Así, una trama remota que arbitra simultáneamente con una trama de datos con el mismo identificador pierde el arbitraje ya que el nivel dominante es el que gana el derecho de acceso al bus. La segunda radica en que no existe valor en el campo de datos.

El principio del ciclo de demanda de datos se ilustra en figura 2.9. El nodo A realiza un requerimiento de una trama de datos mediante un identificador_X en la trama remota. El nodo B es el responsable de transmitir dicha trama de datos ya que es el nodo que maneja el identificador_X y la transmite tanto al nodo A como a todos los nodos que estén interesados en dicha trama.

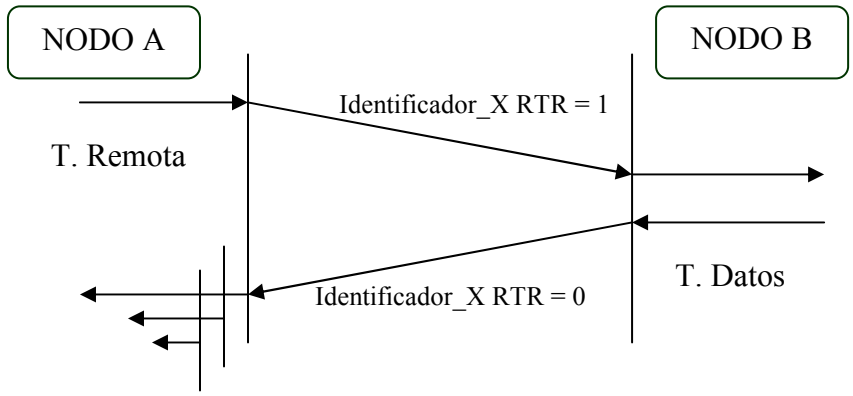


Figura. 2.9. Principio de un ciclo de demanda de datos

2.2.3 Trama de Error

La detección de cualquier error durante una transmisión o recepción de una trama de datos o remota es señalada mediante una trama de error que intencionalmente viola la regla de "bit stuffing" o relleno de bit que se explicará más adelante en la sección 2.5.1. Esta trama de error provoca que el nodo transmisor inicie una retransmisión del mensaje.

La detección de un error durante la transmisión de una trama de error o de sobrecarga también genera una nueva trama de error.

Una trama de error está constituida por dos campos de bits. El primer campo se da por la superposición de banderas de error transmitidas por uno o varios nodos. El segundo campo es una sucesión de ocho bits recesivos que indican el fin de la trama de error y se los conoce como Delimitador de Error.

2.2.3.1 Banderas de Error

Dentro de CAN existen dos tipos de error a señalar que se clasifican mediante una tasa de detección de error obtenida por medio de contadores, éste mecanismo se explicará detalladamente en la sección 2.6 para el confinamiento de fallas. Los dos tipos de errores mencionados son: error activo y error pasivo.

ERROR ACTIVO: Los nodos con una tasa de detección de error por debajo de un cierto límite están en estado de "error activo" y señalan los errores detectados transmitiendo una "bandera de error" activa que consiste en seis bits dominantes.

Un nodo que detecta un error activo lo señala transmitiendo una bandera de error activa primaria. Esta sucesión de bits viola las reglas de llenado de bits (bit stuffing) aplicadas a todos los campos desde SOF hasta el delimitador de CRC o destruye el formato fijo de ACK o EOF.

En consecuencia, todos los otros nodos también descubren una condición de error y lo señalan transmitiendo una bandera de error secundaria. Dependiendo del punto en el tiempo en el que los otros nodos descubren la condición de error se produce una secuencia entre seis y doce bits dominantes en el bus.

ERROR PASIVO: Los nodos con un promedio de tasa de detección de error que excede un límite fijo tienen un problema local y por lo tanto tienen derecho limitado a la señalización del error. Esto se conoce como estado de "error pasivo". Para señalar un error pasivo el nodo encargado transmite una secuencia de seis bits recesivos

Por lo tanto un nodo que señala un error pasivo puede informar al resto de nodos que ha destruido una trama enviada hacia sí mismo pero no está en la capacidad de eliminar tramas recibidas correctamente por otros nodos.

Los nodos transmiten las banderas de error activo inmediatamente después de la detección de la condición de error (error de bit, error de relleno, error de reconocimiento, error de forma) es decir en el siguiente intervalo de bits excepto en el caso de error de CRC en el cual la bandera de error comienza en el bit siguiente al delimitador ACK para no distorsionar las funciones de reconocimiento

2.2.3.2 Delimitador de Error

El delimitador de error consta de 8 bits recesivos los cuales son transmitidos enseguida de la bandera de error. El nodo detector del error transmite el primer bit recesivo delimitador y monitorea el bus hasta que se reconozca un bit recesivo, entonces transmite los siete bits recesivos restantes.

Con este mecanismo un nodo puede determinar si fue el primero en detectar el error y transmitir la bandera de error. Si este es el caso, la transmisión del primer bit recesivo delimitador no es el inmediato debido a la señalización de error secundaria enviada por los otros nodos. Este mecanismo es la base para el confinamiento de nodos erróneos y se puede entender claramente en la figura 2.10. en la cual se detecta un error de bit en un nodo que actúa como transmisor.

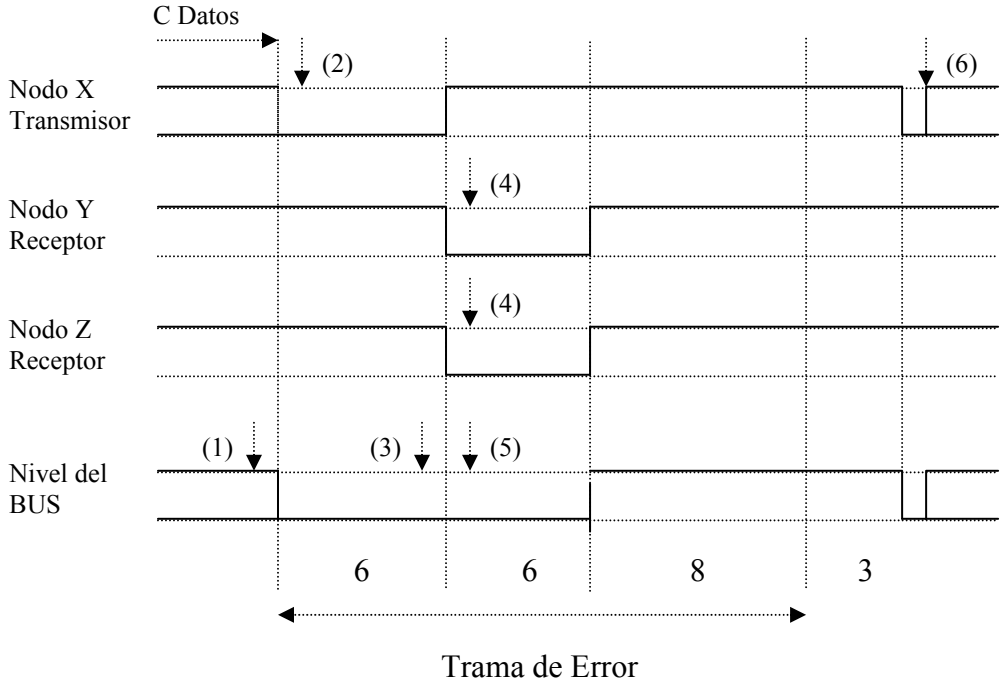


Figura. 2.10. Formación de una trama de error detectada por un nodo transmisor.

En la figura 2.10. el nodo X (transmisor) detecta un error de bit en el campo de datos en el punto (1), señala este error con el próximo bit transmitiendo una bandera de error activa (2). Después de recibir el sexto bit de la bandera del error (3) los otros nodos del bus (Y, Z)

detectan un error de relleno de bit y también transmiten una bandera de error (4) desde el próximo bit. Esto produce una bandera de error con una longitud total de doce bits.

Después de la transmisión de su propia bandera de error, todos los nodos empiezan transmitiendo un nivel recesivo (delimitador) hacia el bus. El nodo X habrá empezado a transmitir primero lo cual le indica que fue el primero en señalar el error. El primer bit recesivo junto con lo siete siguientes bits recesivos forman el delimitador de error con una longitud total de ocho bits.

Después de la transmisión de tres bits en el campo de intermisión, el transmisor empieza a retransmitir los datos mediante un nuevo arbitraje (6).

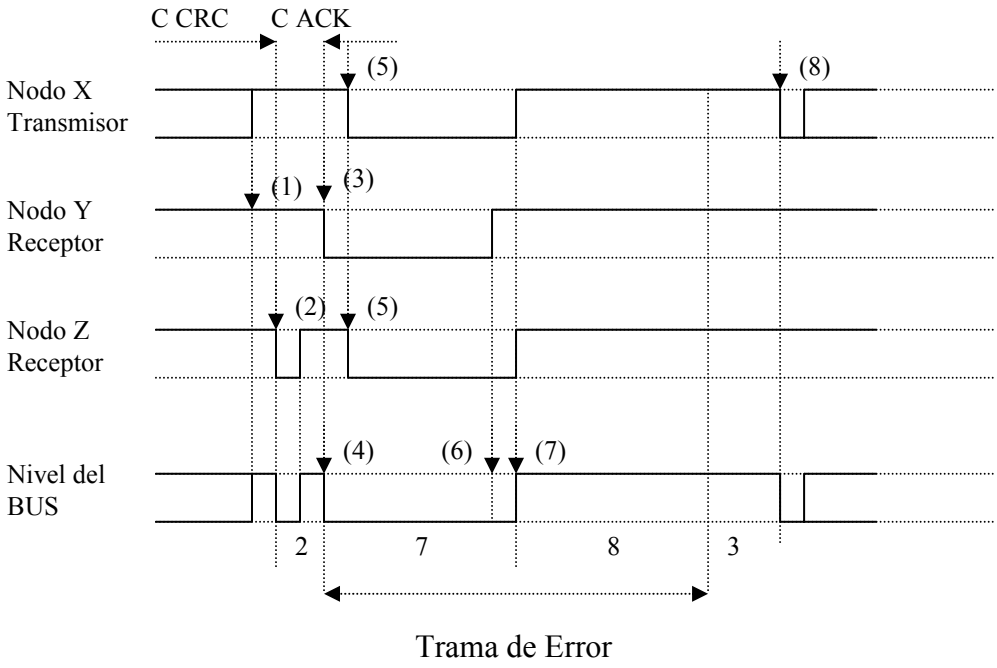


Figura. 2.11. Formación de una trama de error detectada por un nodo receptor.

En la figura 2.11. el nodo Y detecta un error por la secuencia CRC (1). El nodo Z ha recibido correctamente la trama y por consiguiente la reconoce en el Slot ACK (2).

Como el mecanismo de reconocimiento debe conservarse, el nodo Y no señala el error descubierto hasta después del campo ACK (3). Los Nodos X y Z descubren un error de forma en el punto (4) porque están esperando el delimitador final recesivo y señalan el

error en el próximo bit (5). Así, una bandera del error activa se forma en el bus con una longitud total de siete bits dominantes. El nodo Y descubre en (6) que fue el primero en señalar el error. Después de la transmisión de ocho bits delimitadores de error y tres bits de intermisión el nodo transmisor inicia una retransmisión del mensaje con un nuevo arbitraje.

Tanto cuando se detecta un error por un transmisión como cuando se detecta un error por un nodo receptor los tiempos de recuperación de error (23 y 20 bits respectivamente) son sumamente bajos en comparación con otros protocolos de bus de campo, esta ventaja de CAN se da debido a su tipo de detección de errores mediante el método de la señalización de error en lugar del control de error pasivo.

2.2.4 Trama de Sobrecarga.

La trama de sobrecarga tiene el mismo formato que una trama de error activo. Este tipo de trama sólo puede generarse durante el espacio entre tramas lo cual la diferencia de una trama de error, que sólo puede ser transmitida durante la transmisión de un mensaje.

La trama de sobrecarga consta de dos campos, el Indicador de Sobrecarga, y el delimitador. El indicador de sobrecarga consta de 6 bits dominantes que pueden ser seguidos por los generados por otros nodos, dando lugar a un máximo de 12 bits dominantes. El delimitador es de 8 bits recesivos.

Una trama de sobrecarga puede ser generada por cualquier nodo que debido a sus condiciones internas no puede iniciar la recepción de un nuevo mensaje. De esta forma retrasa el inicio de transmisión de un nuevo mensaje. Un nodo puede generar como máximo 2 tramas de sobrecarga consecutivas para retrasar un mensaje. Otra razón para iniciar la transmisión de una trama de sobrecarga es la detección por cualquier nodo de un bit dominante en los 3 bits de intermisión.

Por estas razones una trama de sobrecarga generada por un nodo dará lugar a la generación de tramas de sobrecarga por los demás nodos dando lugar a un máximo de 12 bits dominantes de indicador de sobrecarga y 8 bits recesivos delimitadores.

2.2.5 Espacio Intertramas

El Espacio Intertramas o espacio entre tramas separa una trama de cualquier tipo de la siguiente trama de datos o remota. El espacio entre tramas ha de constar de, al menos, 3 bits recesivos los cuales se denominan intermisión. Una vez transcurrida esta secuencia, un nodo en estado de error activo puede iniciar una nueva transmisión o el bus permanecerá en reposo.

Un nodo en estado de error pasivo deberá esperar una secuencia adicional de 8 bits recesivos antes de poder iniciar una transmisión, los cuales se denominan "suspend transmission" o transmisión suspendida. De esta forma se asegura una ventaja en inicio de transmisión a los nodos en estado activo frente a los nodos en estado pasivo.

Durante el campo de bits de intermisión sólo es posible la señalización de una trama de sobrecarga transmitiendo las banderas correspondientes. La transmisión de una trama de datos o remota no puede iniciar durante este intervalo.

Con los tres bits de intermisión, los siete delimitadores de trama (EOF) y el delimitador ACK, la sucesión de bits recesivos entre dos tramas tiene una longitud total de por lo menos once bits. Después de esta secuencia, el bus se encuentra nuevamente libre.

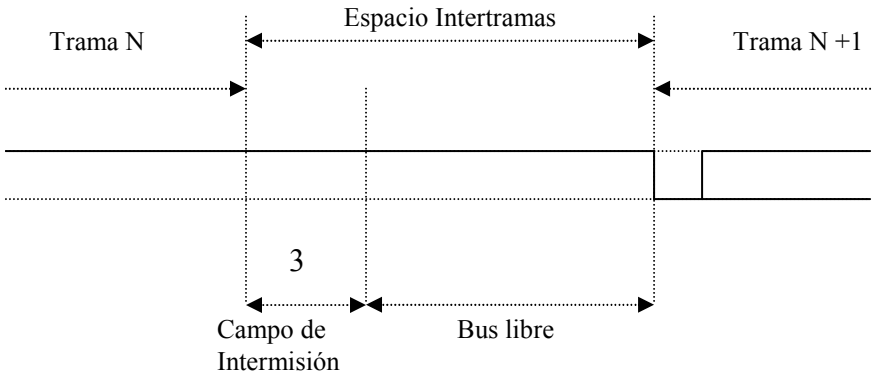


Figura. 2.12. Espacio Intertramas (nodos en estado de error activo)

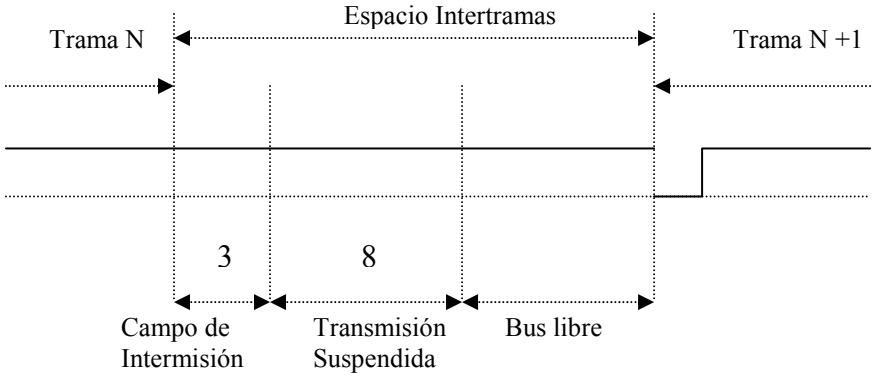


Figura. 2.13. Espacio Intertramas (nodos en estado de error pasivo)

2.3 VALIDACIÓN DE TRAMAS

Las condiciones para que una trama se tome como válida son diferentes para el transmisor y el receptor de la trama. Para un transmisor una trama es válida cuando ningún error se ha señalado hasta el último bit del campo EOF.

Los receptores aceptan una trama como válida cuando no han descubierto ningún indicio de error como el principio de una bandera de error o errores provocados por otros nodos antes del final del campo EOF.

Si sin embargo, debido a un error local, el sexto bit de la bandera EOF es dominante, el nodo receptor afectado rechaza la trama transmitida. Para eliminar la inconsistencia de los datos existente debido a que otros nodos pudieron haber aceptado la trama transmitida, el campo de EOF de una trama CAN posee un séptimo intervalo de tiempo de bit.

Durante este intervalo, un receptor con un error local en el sexto bit del campo EOF puede señalarlo generando una trama de error y así obliga al transmisor a retransmitir la trama.

Es decir que durante el último bit de la bandera de EOF, existe una posibilidad de inconsistencia de datos entre los receptores y el transmisor, esto significa que los nodos

receptores que ya han recibido correctamente la primera trama recibirán nuevamente la misma trama.

De igual forma que cuando se detecta un bit dominante al final del delimitador de error o de sobrecarga, cuando un receptor descubre al último bit de EOF como un bit dominante genera una trama de sobrecarga. Este hecho no provoca una retransmisión de la trama recibida, pero corrige la sincronización entre nodos.

Si tanto un receptor como un transmisor descubren un bit dominante al final del campo EOF debido a un error global, entonces la trama es válida para el receptor, pero no para el transmisor. Como se repite la transmisión, el receptor recibe la trama dos veces.

2.4 CODIFICACIÓN DE BIT

El protocolo CAN representa físicamente a los bits de una trama mediante el código NRZ (No retorno a cero). Esto significa que en cada tiempo de bit el nivel del bit generado es o dominante o recesivo. Por lo tanto en caso de tener una secuencia larga de bits del mismo nivel se puede dar una falta de sincronización y no existe ningún bit que pueda servir de flanco para la resincronización de los receptores.

Esta desventaja de la codificación NRZ puede eliminarse cuando se inserta un bit adicional de nivel complementario "Bit de Relleno" a la secuencia de bits del mismo nivel enviada por el transmisor. Entonces la secuencia de bits mantiene un número suficiente de flancos para una resincronización en los nodos receptores. Los receptores quitan los bits adicionales insertados por el transmisor antes de completar el proceso. Este proceso es conocido como Bit Stuffing o Relleno de Bit.

En las tramas de datos y remota, el segmento de trama SOF, el campo de arbitraje, campo de control, campo de datos y la sucesión CRC están codificadas por el método de relleno de bit. Siempre que un transmisor detecte cinco bits consecutivos en la secuencia, inserta un bit de nivel complementario automáticamente. Las tramas de error y sobrecarga, así como, los campos de bits restantes de las tramas de datos o remota es decir el

delimitador CRC, campo ACK y EOF son de formato fijo y no utilizan el mecanismo de relleno de bit.

Como ya se dijo la longitud de la bandera de error en la trama de error es de 6 bits de igual valor; esto se da debido a la definición de bit de relleno (5 bits de igual valor) por lo tanto una violación a la regla de relleno de bit (6 bits de igual valor) también puede ser considerada como un indicativo de una condición de error y se da en los campos de bits que no trabajan bajo el bit o relleno de bit.

Así se concluye que el relleno de bit tiene dos funciones: resincronización de nodos receptores y señalización de error.

2.5 DETECCIÓN Y TRATAMIENTO DE ERRORES

2.5.1 Mecanismos de detección de error

CAN provee los siguientes mecanismos de detección de error**:**

Bit Check (verificación de bit)

La verificación de bit se da mediante la supervisión del nivel actual del bus y el nivel transmitido, si es diferente se da un error de bit; este método sirve para detectar errores globales. Además se detectan todos los errores locales en el transmisor.

Los cambios o sobre escrituras de bits durante la fase de arbitraje, el Slot ACK o en la bandera de error pasiva, no son tomados como errores de bit.

Frame Check (verificación de trama o de forma)

En las tramas CAN existen campos de bits con formatos fijos específicos. Un error de trama es detectado cuando alguno de estos campos contiene uno o varios bits ilegales, o fuera de su formato específico.

CRC check (verificación de redundancia cíclica)

La verificación de redundancia cíclica se da para asegurar la integridad de los datos recibidos por un nodo. Este método ofrece una buena garantía a los sistemas de transmisión de datos debido a su alta probabilidad de detección de errores.

En la verificación de redundancia cíclica, cada trama es representada como una trama polinomial la cual es dividida por un generador polinomial definido. El residuo de esta división es la secuencia CRC transmitida por el bus como parte de la trama. En el receptor se realiza la misma operación, es decir, se divide la trama polinomial recibida por el generador polinomial; si el nuevo residuo es igual al residuo recibido en la trama, la transmisión está libre de errores.

Los coeficientes de la trama polinomial son formados por los valores de los bits (0/1) de la secuencia formada por el campo SOF, campo del arbitraje, campo de control y el campo de datos.

La secuencia de verificación de 15 bits usada en el protocolo CAN se basa en el código BCD, un código cíclico bien especializado para las longitudes de trama menores a 127 bits. El generador polinomial de grado 15 aplicado está dado por:

$$x^{15} + x^{14} + x^{10} + x^{8} + x^{7} + x^{4} + x^{3} + 1$$

ACK check (verificación de reconocimiento)

Un nodo transmisor espera la llegada de un bit dominante en el slot ACK por parte de por lo menos uno de los nodos receptores. Un reconocimiento perdido, es decir un nivel recesivo en el Slot ACK, se toma como un error de reconocimiento.

Stuff rule check (verificación de la regla de llenado de bits)

Cada nodo detecta una violación de la regla de llenado de bit (bit sttuffing) en cuanto recibe el sexto bit consecutivo del mismo nivel en algún campo de la trama. Por consiguiente, cada nodo que detecta una bandera de error la interpreta como una violación a la regla de relleno de bit y genera su propia trama de error.

2.5.2 Capacidad de detección de error

La capacidad para descubrir errores en una transmisión de datos depende en gran parte de los mecanismos de detección de error aplicados y varía ampliamente entre los diferentes protocolos.

Las condiciones de interferencia electromagnética bajo las cuales opera el sistema de transmisión de datos y la habilidad de detección de errores del sistema es esencial para la valoración de la integridad de los datos transmitidos.

Una medida estadística de la integridad de datos transferidos en un sistema de transmisión se denomina “probabilidad del error residual”. Esta medida especifica la probabilidad de tramas erróneas no detectables.

Debido a la combinación de los diferentes mecanismos de detección de error descritos anteriormente, la capa de enlace de datos del protocolo CAN garantiza un alto nivel de detección de error. Todos los errores que se dan de forma global son detectados por el proceso de monitoreo de bits desarrollado por el transmisor de un trama. Estos errores se refieren a todas las perturbaciones que se pueden dar en el bus debido a la interferencia electromagnética y se presentan en todos los nodos que conforman la red. Además, el transmisor también es capaz de detectar todos los disturbios locales. Los errores que solo ocurren localmente en los receptores son detectados mediante la secuencia de 15 bits verificada por todos los receptores.

Gracias a todos estos mecanismos de detección de errores se ha logrado un Distancia Hamming de seis en el protocolo CAN. En los protocolos de bus de campo convencionales la distancia Hamming normalmente es de cuatro o menos.

Mediante una serie de pruebas, operaciones y análisis que incluyen parámetros tales como la taza de error de trama (determinada según la perturbación del ambiente en el que se desarrolla la red), promedio de carga en el bus, tasa de transmisión de datos y longitud del mensaje; se ha concluido que la probabilidad de error residual en el protocolo CAN es de $4,7 * 10^{-14}$, esto significa que dentro de un rango de 10^{13} mensajes puede ocurrir un promedio de 2 mensajes erróneos no detectados. En un promedio estadístico este evento ocurre cada 2000 años.

Los valores de los parámetros utilizados son los siguientes:

- Taza de transmisión de bit = 500kbits/seg.
- Promedio de carga en el bus = 25%.
- Promedio de longitud de mensaje = 80 bits.
- Tiempo de operación por año = 2000 horas.
- Promedio de taza de error = 10^{-3}

Como ya se dijo un nodo señala un error detectado mediante la transmisión de una bandera de error. Cuando se detecta un error ya sea de bit, de forma (trama), de relleno o de reconocimiento; la transmisión de la bandera de error se da desde el siguiente intervalo de bits, mientras que cuando el error detectado es mediante el código CRC la transmisión de la bandera de error empieza con un retardo de dos tiempos de bits; es decir, después del Slot ACK. Esto se realiza para conservar el mecanismo de reconocimiento.

El protocolo CAN tiene estas ventajas sobre otros protocolos de bus de campo ya que la señalización de error es muy rápida y por lo tanto la retransmisión de la trama errónea se da tan pronto como sea posible, mientras que en los otros protocolos por lo general la retransmisión de dicha trama errónea se da después de un tiempo de reconocimiento de error predefinido.

2.6 CONFINAMIENTO DE FALLAS

El reemplazo de las tecnologías de instalación eléctrica convencionales por un sistema de bus de transmisión en serie puede ocasionar que un nodo de la red defectuoso bloquee el sistema entero. Particularmente, el principio de señalización de error aplicado por el protocolo CAN implica este riesgo ya que un nodo defectuoso puede generar banderas de error continuamente. Para eliminar este riesgo, se proporciona un mecanismo que puede descubrir a un nodo defectuoso y desconectarlo de la red o del bus.

Para este propósito el protocolo CAN proporciona un contador de error transmisor y un contador de error receptor para cada nodo. Los valores de dichos contadores se incrementan en un valor específico en cada transmisión o recepción de una trama errónea y decrementan en uno con cada transmisión o recepción de una trama satisfactoriamente.

En caso de detectar un error, el incremento de los contadores de error es mayor que el decremento en caso de una transmisión exitosa. El valor del incremento depende del momento en el que un nodo detecta el error o si un nodo fue el primero en descubrirlo.

Esto puede llevar a un incremento muy notable en los contadores durante un cierto período de tiempo, sin embargo los contadores pueden ser reducidos nuevamente durante una fase más larga con una tasa de error muy baja.

De esta forma los valores de los contadores de error son una medida de la taza de error de la transmisión.

Cuando un nodo detecta un error local, la transmisión de la bandera de error realizada por dicho nodo causa una señalización de error en el resto de nodos de la red. Según el tiempo de realización de dicha bandera, los nodos reconocen cual de ellos fue el primero en señalar el error. Para el confinamiento de fallas, los contadores de error de transmisión y recepción del primer nodo en señalar el error se incrementan en una mayor cantidad (un valor de ocho y nueve respectivamente) que los contadores de los nodos que señalaron el error posteriormente. Si un nodo es continuamente el primero en señalar un error durante un largo periodo de tiempo se puede asumir que tiene problemas y se convierte en un nodo defectuoso.

Cuando se exceden límites específicos en los contadores de error, se toman las medidas necesarias para restringir el efecto del nodo defectuoso dentro de la red. Estas medidas van desde prohibir la transmisión de banderas de error activo, cuando se alcanza el nivel de cuenta de error específico, hasta prohibir al nodo cualquier interacción con el bus cuando se alcanza el nivel de cuenta de error aun más alto.

Dependiendo del valor que tengan los contadores de error, un nodo se puede encontrar en tres estados:

- Error Activo
- Error Pasivo
- Fuera de bus

Un nodo de la red en estado de error activo toma parte en la comunicación del bus y envía una bandera de error activa cuando detecta un error, destruye la trama que se estaba transmitiendo, viola la regla de relleno de bit e impide a otros nodos que acepten la trama errónea.

Un nodo de la red en estado de error pasivo ya ha aumentado relativamente los valores de los contadores de error y ha supervisado una taza de error significativamente alta

durante un largo período de tiempo. Dicho nodo aún se comunica con el bus señalando el error pero ya no mediante una bandera activa sino mediante una bandera pasiva la cual no afecta al resto de nodos en la red.

Además, un nodo en estado de error pasivo debe esperar ocho tiempos de bit (Campo de Transmisión Suspendida) antes de intentar la retransmisión de la trama detectada como errónea.

Estas dos medidas aseguran que un nodo que ya ha estado detectando una alta taza del error durante un largo período de tiempo no puede interferir con la comunicación del resto de nodos de la red.

Un nodo de la red en estado Fuera de Bus no puede tener ninguna influencia en el bus, no puede enviar ningún tipo de tramas ni reconocimientos y por lo tanto tampoco afectará al resto de nodos de la red.

Después de un reset o configuración de nodos, un nodo está en estado de error activo. Si uno de los dos contadores de error excede del valor 127, el nodo entra en estado de error pasivo. Cuando los dos contadores de error vuelven a un valor menor a 128 el nodo entra nuevamente en estado de error activo. Un nodo se desconecta totalmente del bus e ingresa a un estado fuera de bus cuando el valor del contador de error transmisor excede de 255. Estas consideraciones se resumen en la figura 2.14.

El nodo puede salir de estado “fuera de bus” y volver al estado de error activo solo cuando los contadores de error se reinician en cero (reset) y se reconfiguran los nodos además se debe haber detectado 128 secuencias de 11 bits recesivos consecutivos.

Esta medida asegura que un nodo recientemente reiniciado que continúe con errores no vuelva a perturbar inmediatamente la comunicación después de un reset.

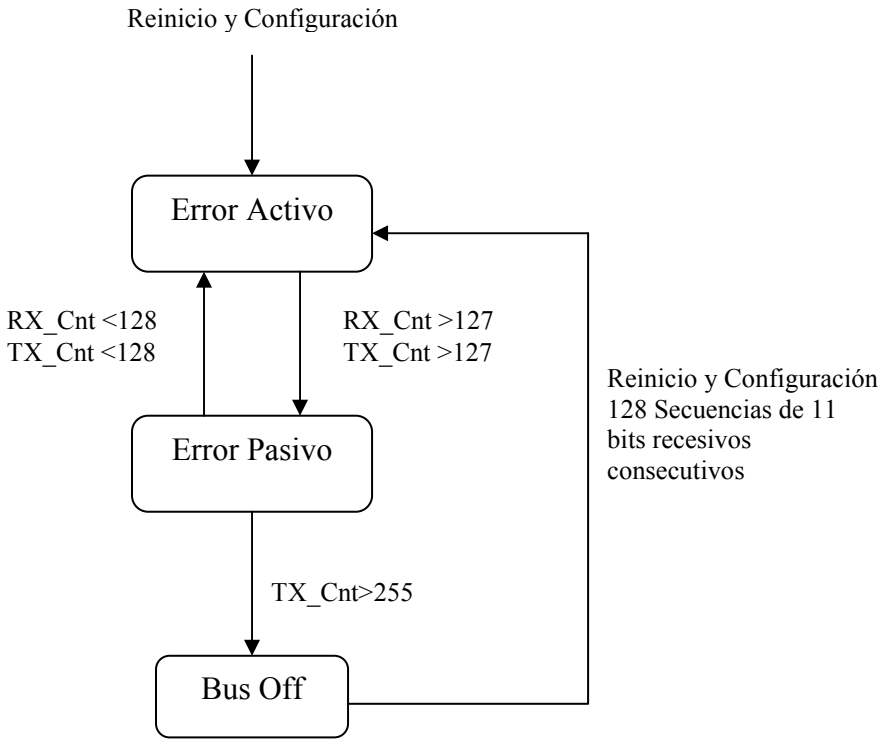


Figura. 2.14. Diagrama de Estados de Error de nodo CAN

Las reglas establecidas para el confinamiento de fallas tienen por objetivo determinar que nodo detecta errores continuamente y además es el primero en detectarlos; así como otros casos particularmente severos de error. Por otro lado, puede haber una alta tasa de error temporal en el bus causada por motivos externos, como por ejemplo altas interferencias electromagnéticas temporales; por esta razón, las reglas también deben proporcionar una reducción en los contadores de error con una tasa de error reducida. Esto se logra disminuyendo los contadores correspondientes en cada transmisión o recepción de una trama correctamente.

Según las reglas establecidas para el confinamiento de fallas, el contador de error transmisor de un nodo incrementa en ocho cuando el nodo detecta un error durante la transmisión de una falla pero decrece solo en uno cuando el nodo transmitió la trama exitosamente. Esto significa que una red CAN permanece funcional cuando se han transmitido por lo menos ocho tramas correctamente antes de transmitir una trama errónea.

2.6.1 Reglas establecidas para el Confinamiento de Fallas

Las reglas establecidas para el funcionamiento de los contadores CAN de transmisión y recepción están dadas por la norma ISO99-1 y son las siguientes:

1) Cuando un nodo receptor detecta un error, su contador de error receptor incrementará en uno excepto si el error detectado fue un error de bit generado durante el envío de una bandera de error activa o una bandera de sobrecarga. En este caso se aplica la regla 5.

2) Cuando un nodo receptor es el primero en detectar un error después de la transmisión de una bandera de error, su contador de error receptor aumenta adicionalmente en ocho. Este es el caso en el que el receptor detecta un bit dominante después de enviar una bandera de error debido a una bandera de error secundaria sucesiva.

3) Cuando un nodo transmisor envía una bandera de error, su contador de error transmisor incrementa en ocho, esto se da cuando un transmisor detecta un error de bit o no recibe un reconocimiento.

La siguiente excepción se aplica a esta regla:

- Si el transmisor está en error pasivo, además dicho transmisor detecta un error de reconocimiento y no detecta un bit dominante mientras envía su bandera de error pasiva, entonces su contador de error transmisor no aumenta. Esta excepción previene que un nodo transmisor en una red sin nodos adicionales se ponga en estado Fuera de Bus debido a una falta de reconocimiento.

4) Si un transmisor detecta un error de bit mientras envía un error activo o una bandera de sobrecarga, su contador transmisor aumenta en ocho.

5) Si un receptor detecta un error de bit mientras envía un error activo o una bandera de sobrecarga, su contador receptor aumenta en ocho.

6) Cualquier nodo tolera siete bits dominantes consecutivos después de enviar una bandera de error activa, pasiva o de sobrecarga. Después de detectar el bit dominante número 14 consecutivo (en el caso de una bandera de error activa o de sobrecarga) o después de detectar el octavo bit dominante después de una bandera de error pasiva entonces después de cada sucesión de 8 bits dominantes consecutivos adicionales cada nodo transmisor aumenta en 8 su contador respectivo y cada receptor incrementa en 8 su contador respectivo.

7) Después de la transmisión exitosa de una trama, el contador de error transmisor del nodo correspondiente disminuye en uno, a menos que ya se encuentre en cero.

8) Después de la recepción exitosa de una trama el contador de error receptor disminuye en uno si su valor se encuentra entre 1 y 127, si su valor es 0 se mantiene en 0 y si su valor es mayor a 127 se pondrá en un valor entre 119 y 127.

Nodo Receptor

Generalmente el contador de error receptor se incrementa en nueve cada vez que se detecta primero un error y en uno cuando detecta el error simultáneamente con otros nodos. El contador disminuye en uno cada vez que se recibe exitosamente una trama.

Cuando la cuenta del error excede a 96 se indica una señal de alta taza de error en el controlador, pero el nodo sigue en estado de error activo

Cuando la cuenta del error excede el valor 127, el nodo pasa a un estado de error pasivo.

Un nodo receptor no puede ingresar al estado Fuera de Bus ya que cuando el contador excede el valor 127, espera a la próxima recepción exitosa de una trama y dicho contador se coloca en un valor inferior a 127 como se explica en la regla 8

Nodo Transmisor

El contador transmisor se incrementa en ocho para cada detección de error y disminuye en uno para cada transmisión de trama exitosa. Si la cuenta excede el valor 96 se da una indicación de tasa de error alta en el controlador, si la cuenta excede 127 el nodo pasa a un estado de error pasivo y si la cuenta excede el valor 255 el nodo pasa al estado Fuera de Bus.

2.7 PROTOCOLO CAN EXTENDIDO

Existe una versión extendida del protocolo CAN que surge a raíz de la necesidad de abarcar un mayor número de mensajes dentro de la red. Esto se consiguió mediante la adición de bits identificadores en el campo de arbitraje de las tramas. La versión 2.0A tiene 11 bits identificadores de mensajes alcanzando un máximo de 2048 mensajes posibles y la versión extendida 2.0B tiene 29 bits identificadores de mensajes los cuales pueden abarcar hasta 512 millones de mensajes diferentes.

Según la Especificación CAN 2.0 B, los formatos de trama base extendidos son compatibles, es decir que los dos formatos de tramas pueden existir simultáneamente en la misma red y nodo.

Los dos formatos se distinguen por el IDE (Bit de Extensión de Identificador) que se ubica en el campo de control de la trama. En el formato extendido, los 29 bits identificadores se dividen en dos secciones, los 11 bits identificadores base (Base-ID) y los 18 bits identificadores extendidos (Extended-ID). A continuación de los IDs base se encuentra el bit SRR (Substitute Remote Request Bit) y el IDE (Identifier Extensión Bit) como parte del campo de arbitraje.

El bit RTR del formato base se sustituye por el bit SRR transmitido en estado recesivo en el formato extendido. Esto implica que una trama de datos (RTR dominante) transmitida en el formato base prevalece sobre la transmisión de una trama de datos con IDs idénticos en el formato extendido.

El bit IDE se utiliza para distinguir entre los formatos base y extendidos. En el formato base se transmite dominante y en el extendido se transmite recesivo.

Los seis bits del campo de control son ligeramente diferentes en el formato base y el extendido, el campo de control del formato base contiene el bit IDE (dominante), el bit r0 (dominante) y 4 bits para la longitud de código de datos. El formato extendido contiene dos bits reservados r0 y r1 (dominantes) además de los 4 bits de longitud de código de datos. En la figura 2.15. y 2.16. se pueden apreciar claramente las diferencias entre los formatos base y extendido.

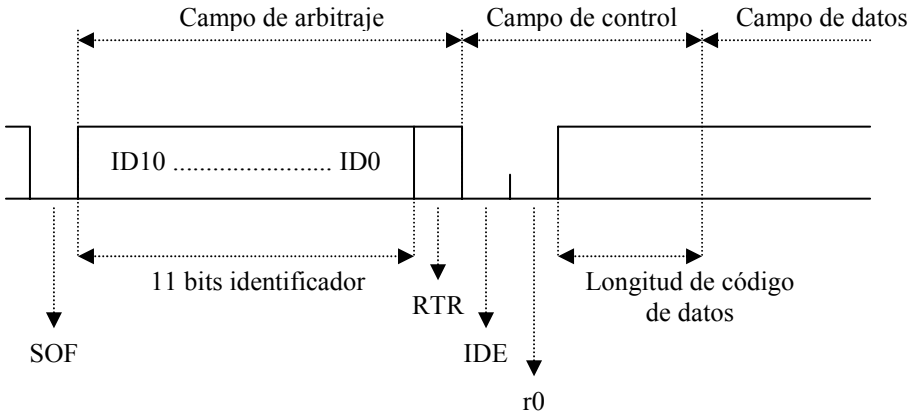


Figura. 2.15. Especificación CAN 2.0 B versión estándar, campo de arbitraje y control

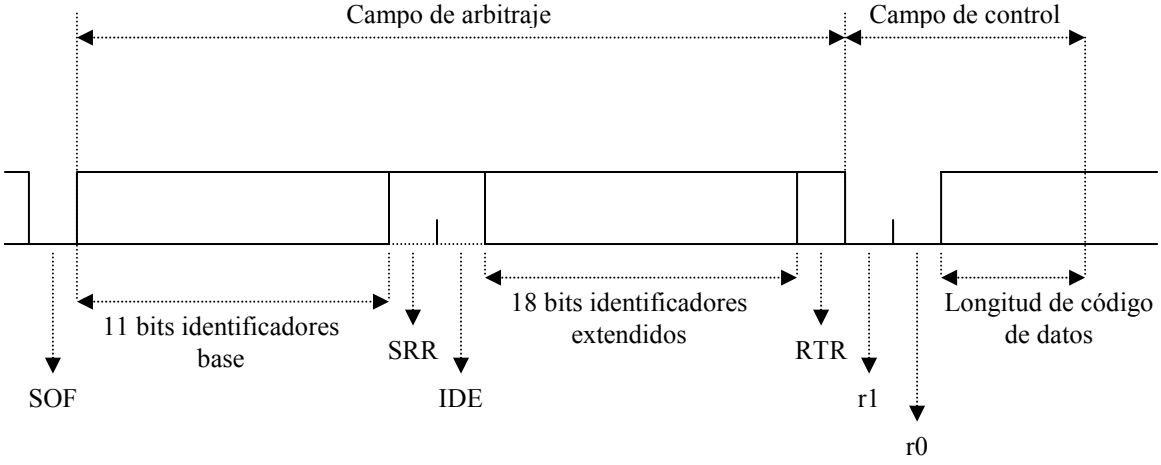


Figura. 2.16. Especificación CAN 2.0 B versión extendida, campo de arbitraje y control

CAPITULO 3

CAPA FÍSICA DEL PROTOCOLO CAN

La capa física del protocolo CAN cubre aspectos relacionados a la transmisión física de datos entre los nodos de la red, cualquier capa física CAN debe ser capaz de soportar los dos estados o niveles que maneja el protocolo CAN es decir el nivel dominante y el nivel recesivo.

El medio de transmisión debe estar en estado recesivo cuando ningún nodo de la red ha transmitido un nivel dominante y si uno o varios nodos han transmitido un nivel dominante el medio de transmisión debe estar en estado dominante.

La señalización física válida para cualquier aplicación CAN se encuentra en el estándar ISO898-1 [ISO99-1]. El estándar ISO898-2 [ISO99-2] especifica características de acceso al medio a velocidades altas, es decir, alrededor de 1 Mbps y el estándar ISO898-3 [ISO99-3] para velocidades bajas, es decir, de 250 Kbps o inferiores.

3.1 ESTÁNDARES DE LA CAPA FÍSICA DEL PROTOCOLO CAN

Además de los estándares ISO898 existen otros estándares adicionales con funciones más específicas como el CIA DS-102 o los estándares de alto nivel como CanOpen y DeviceNet

La parte de la señalización física del protocolo CAN se define por el estándar ISO898-1 y se la analizará a profundidad en la sección 3.2

La tabla 3.1. muestra las características principales de los diferentes estándares y se puede apreciar las diferencias existentes entre ellos, más adelante se detallarán uno por uno:

	ISO898-2	**CIA DS-102**	**CANopen**	**DeviceNet**	**SAEJ2411**
Veloc. De transmisión	De 5Kbps a 1Mbps	De 10Kbps a 1Mbps	CIA DS-102	125Kbps 250Kbps 500Kbps	33,3Kbps 83,3 Kbps
Long. bus (máxima velocidad)	40 mts	25 mts	CIA DS-102	100 mts	-
Nodos	-	32	127	64	32
Conectores definidos	-	SUB-D9	SUB-D9 RJ10 RJ45	Ministyle Microstyle Openstyle	-
Resistencia de fin de línea	120 Ohm	ISO 11898-2	ISO 11898-2	121 Ohm	-
Rango de voltajes	-2 a 7 (V)	ISO 11898-2	ISO 11898-2	ISO 11898-2	0 a 4.1 (V)
Nivel de bit	Dominante y recesivo según el Vdif.	ISO 11898-2	ISO 11898-2	ISO 11898-2	Dominante: 0 V Recesivo: 4.1 V

Tabla. 3.1. Características principales de los estándares CAN

3.1.1 Estándar ISO898-2

Como se observa en la tabla 3.1, esta norma especifica características fundamentales para otros estándares desarrollados para las redes CAN en la automatización industrial, dentro de las cuales se encuentran las siguientes:

- CAN funciona a una tasa de datos de 1 Mbps y su máxima longitud de bus a dicha velocidad es de 40 m.

- Trabaja bajo un modo diferencial o balanceado mediante dos líneas independientes de señal: CAN LOW (CAN_L) y CAN HIGH (CAN_H).

- Impedancia característica de la línea de 120 Ohm.

- Rango de voltajes entre –2V (CAN_L) y 7V (CAN_H).

- Retardo de propagación: 5ns/m.

Para determinar el estado del bus se trabaja mediante un voltaje diferencial Vdif que se define como: $V_{DIF} = V_{CAN\_H} - V_{CAN\_L}$.

Si V_{DIF} se encuentra entre –500 mV y 50 mV el bus se encuentra en nivel RECESIVO.

Si V_{DIF} se encuentra entre +1.5 V y 3 V el bus se encuentra en nivel DOMINANTE.

En la figura 3.1. se muestran los niveles de bus *nominales* de acuerdo al estándar 118982

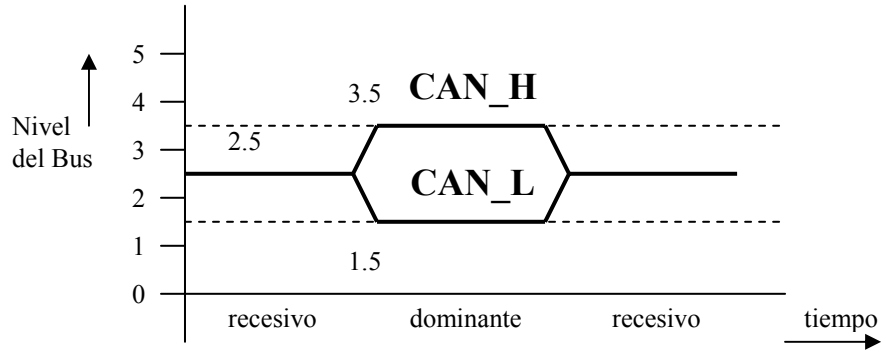


Figura. 3.1. Niveles de bus nominales para ISO 11898-2

Un nodo detecta un estado recesivo si el voltaje de CAN_H no es mayor al voltaje de CAN_L más 0.5V. Si el voltaje de CAN_H es al menos 0.9V mayor que el voltaje de CAN_L se detecta un nivel dominante.

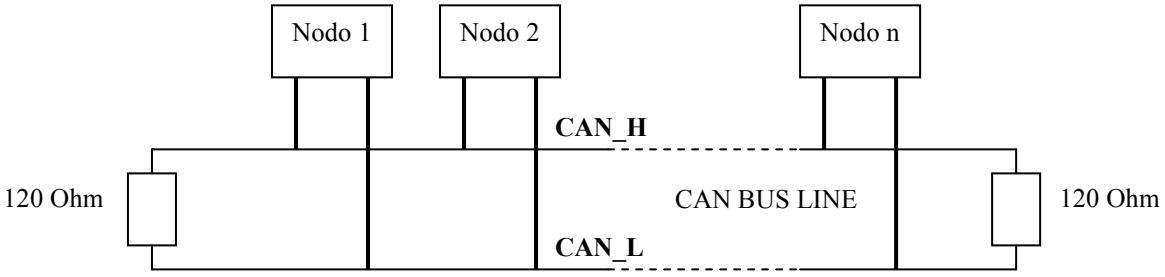


Figura. 3.2. Topología CAN según ISO 11898-2

3.1.2 Estándar CIA DS-102

La realización de redes abiertas CAN están apoyadas por las recomendaciones DS-102 de CIA (CAN in Automatization) y se basan en las recomendaciones ISO 11898-1 e ISO 11898-2 pero se incluyen características más detalladas. La tasa de datos entre 10 Kbps y 1 Mbps tomando en cuenta los parámetros de tiempos de bit de la tabla 3.2.

Tasa de Datos	Máxima Longitud de línea	Tiempo de bit Nominal	Número de tiempos del quántum	Longitud de tiempo del quántum	Punto de muestreo
1 Mbps	25 m	1 us	8	125 ns	6 tq
800 Kbps	50 m	1.25 us	10	125 ns	8 tq
500 Kbps	100 m	2 us	16	125 ns	14 tq
250 Kbps	250 m	4 us	16	250 ns	14 tq
125 Kbps	500 m	8 us	16	500 ns	14 tq
50 Kbps	1000 m	20 us	16	1.25 us	14 tq
20 Kbps	2500 m	50 us	16	3.125 us	14 tq
10 Kbps	5000 m	100 us	16	6.25 us	14 tq

Tabla. 3.2. Recomendaciones de tiempos de bit y longitudes de línea CIA DS-102

Uso de conectores con asignación de pines específicos, el SUB-D9 es utilizado para aplicaciones industriales y se define de la siguiente manera:

Conector SUB-D9

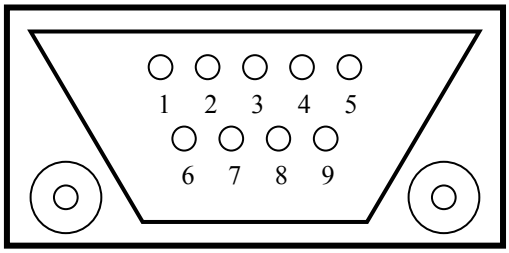


Figura. 3.3. Pines del conector CAN SUB-D9

PINES	SEÑAL	DESCRIPCIÓN
1	---	Reservado
2	CAN_L	Señal CAN LOW
3	CAN_GND	Tierra CAN
4	---	Reservado
5	CAN_SHLD	CAN Shield
6	GND	Tierra CAN opcional
7	CAN_H	Señal CAN High
8	---	Reservado
9	CAN_V+	Voltaje Externo Opcional

Tabla. 3.3. Asignación de pines del conector SUB-D9

El voltaje externo opcional tanto en el conector SUB-D9, como en el resto de conectores que se utilizan en los diferentes estándares que se analizan más adelante, por lo general se utilizan para alimentar transceivers.

3.1.3 Estándares de alto nivel

Los protocolos de alto nivel como CanOpen o DeviceNet tienen como fin garantizar la mayor interoperabilidad e intercambiabilidad de dispositivos para lo cual se apoyan en

estándares propios de cada uno pero siempre tomando como base a los estándares ISO11898.

3.1.3.1 CANopen

Con CanOpen se pueden implementar hasta 127 nodos, su medio de transmisión especifica dos líneas en cuyos extremos se colocan dos resistencias de fin de línea de una impedancia específica. Todos los dispositivos CanOpen deben soportar tasas de datos desde 20 Kbps y recomiendan la longitud del bus de acuerdo a la norma CIA DS102.

CanOpen puede utilizar el conector de 9 pines (serial) de acuerdo a la norma CIA DS 102 y pueden utilizar otros conectores como el conector multipolo, RJ10, RJ45.

Conector RJ10

El conector RJ10 se define de la siguiente manera:

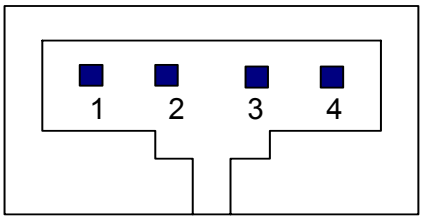


Figura. 3.4. Pines del conector RJ10

PINES	SEÑAL	DESCRIPCIÓN
1	CAN_V+	Reservado
2	CAN_H	Señal CAN High
3	CAN_L	Señal CAN Low
4	CAN_GND	Tierra

Tabla 3.4. Asignación de pines del conector RJ10

Conector RJ45

El conector RJ45 se define de la siguiente manera:

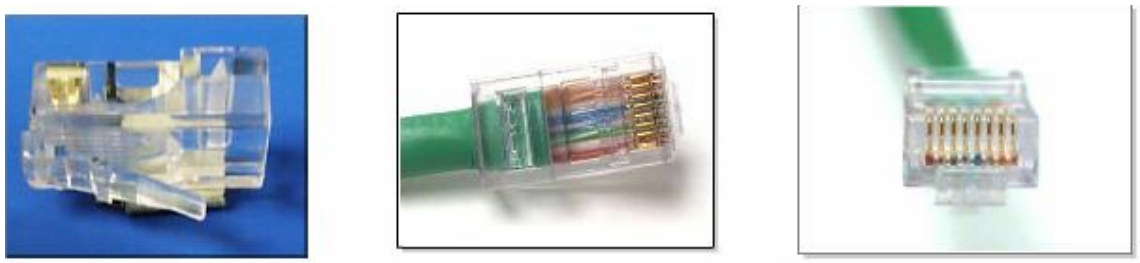


Figura. 3.5. Conector RJ45

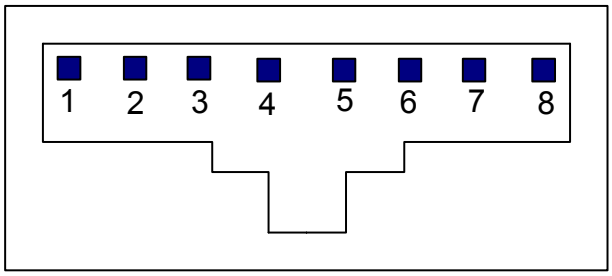


Figura. 3.6. Pines del Conector RJ45

PINES	SEÑAL	DESCRIPCIÓN
1	CAN_H	Señal CAN HIGH
2	CAN_L	Señal CAN LOW
3	CAN_GND	Tierra CAN
4	---	Reservado
5	---	Reservado
6	CAN_SHLD	CAN Shield
7	GND	Tierra CAN opcional
8	CAN_V+	Voltaje Externo Opcional

Tabla. 3.5. Asignación de pines del conector RJ45

3.1.3.2 DeviceNet

La representación de la capa física de DeviceNet representa una extensión del estándar ISO 11898.

DeviceNet especifica redes hasta de 64 nodos con topología de bus. Las líneas del bus son terminadas en ambos extremos con una impedancia de 127 Ohm ± 1%.

De acuerdo a DeviceNet se pueden trabajar en tres tasas de datos: 125Kbps, 250Kbps y 500Kbps, y trabaja solo bajo tres tipos de conectores Open-style, Mini-style y Micro-style.

La configuración de los pines de cada conector se muestra a continuación ya sea para CanOpen o DeviceNet.

Conector Mini-style

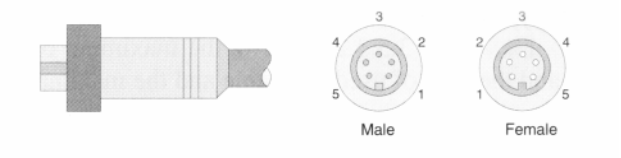


Figura. 3.7. Conector CAN mini-style de 5 pines

PINES	SEÑAL	DESCRIPCIÓN
1	CAN_SHLD	Señal Opcional
2	CAN_V+	Voltaje externo auxiliar
3	CAN_GND	Tierra CAN
4	CAN_H	Señal CAN HIGH
5	CAN_L	Señal CAN LOW

Tabla. 3.6. Descripción de pines del conector Mini-style

Figura. 3.8. Conector tipo T mini/mini/mini

Conector Open Style

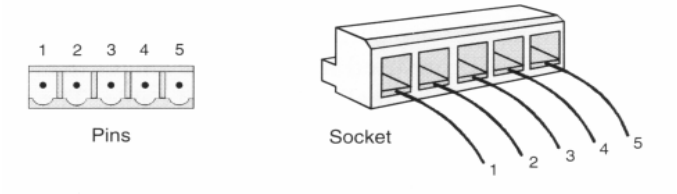


Figura. 3.9. Pines Conector CAN Open-style

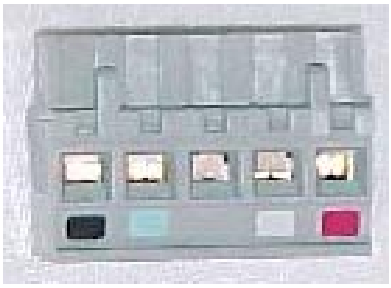


Figura. 3.10. Conector Open-Style

PINES	SEÑAL	DESCRIPCIÓN
1	CAN_GND	Tierra CAN
2	CAN_L	Señal CAN LOW
3	CAN_SHLD	Señal Opcional
4	CAN_H	Señal CAN HIGH
5	CAN_V+	Voltaje externo auxiliar

Tabla. 3.7. Descripción de pines del conector Open-style

Conector Micro Style

El micro-style el cual tiene la misma configuración que el conector Mini-style.

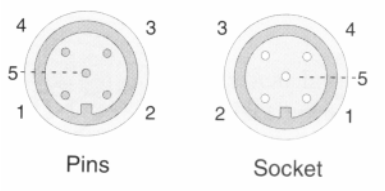


Figura. 3.11. Pines Conector CAN Micro-style

Figura. 3.12. Conector tipo T mini/mini/micro

3.1.4 Estándar SAE J2411

El estándar SAE J2411 fue desarrollado por General Motors como una alternativa para aplicaciones de redes CAN mediante una sola línea con bajos requerimientos en relación a velocidad de transmisión y largo del bus, sus parámetros básicos son:

Comunicación mediante una sola línea de bus y dos modos de comunicación: normal y alta velocidad. Trabaja a una tasa de datos nominal de 33.3 Kbps en modo normal.

Para programas de diagnósticos se puede trabajar a una tasa de datos nominal de 83.3 Kbps en modo de alta velocidad manejado mediante software.

Trabajan 32 nodos por red.

La aplicación principal de las redes CAN de una sola línea se da en los vehículos de motor la cual proporciona gran comodidad dentro del vehículo. Debido a su baja velocidad de transmisión de datos su topología no está limitada a una longitud pequeña.

La figura 3.13. muestra los niveles de bus nominales de acuerdo al estándar SAE J2411.

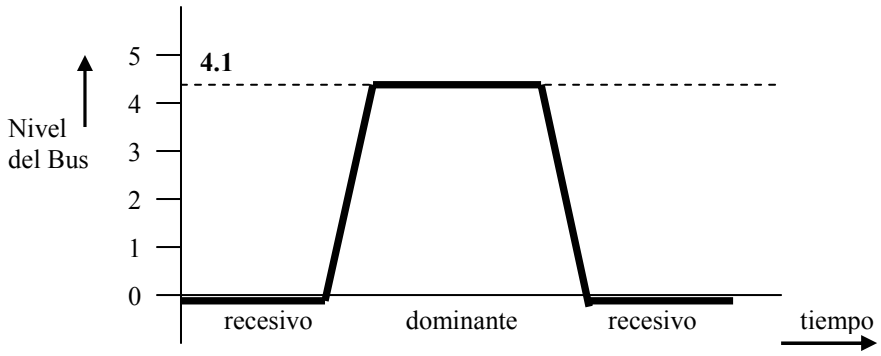


Figura. 3.13. Niveles de bus nominales de acuerdo a SAE J2411

El estándar SAE J2411 incluye una capacidad selectiva de nodos de tipo sleep lo que permite la comunicación normal entre varios nodos mientras otros nodos se encuentran en estado de reposo (sleep).

3.2 SEÑALIZACIÓN FÍSICA

3.2.1 Representación del bit

Los métodos de codificación de bit se caracterizan por el número de slots o periodos de tiempo necesarios para representar un bit, en el caso de CAN se utiliza el código NRZ el cual necesita un solo slot de tiempo ya que el nivel de la señal permanece constante durante todo el tiempo de bit; esto no ocurre en otros mecanismos de codificación tales como el Manchester el cual necesita dos slots de tiempo debido al cambio de nivel existente durante todo el tiempo de bit, esto se visualiza en la figura 3.14.

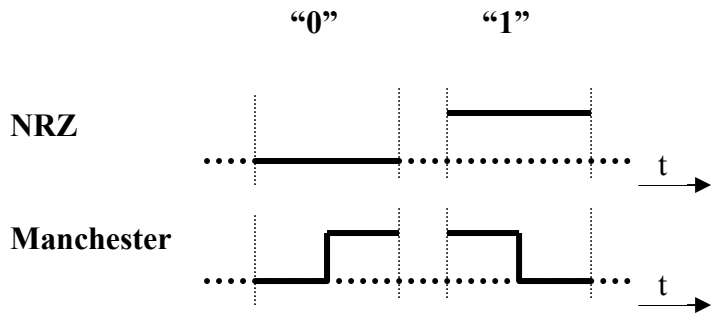


Figura. 3.14. Representación de bit según la codificación NRZ y Manchester

La ventaja del código Manchester es que siempre existe una señal disponible para la sincronización de bits debido al cambio de nivel existente en cada representación del bit, este punto se denomina flanco. Esto no ocurre en el código NRZ cuando se presentan varios bits del mismo nivel ya que se mantendrá dicho nivel sin cambio alguno. Por otro lado el código Manchester puede trabajar máximo a un poco más de la mitad de la velocidad de transmisión del código NRZ.

Debido a que con la codificación NRZ la señal puede permanecer constante durante un largo periodo de tiempo dependiendo de los datos transmitidos, se deben tomar medidas para garantizar que no se exceda un máximo intervalo de tiempo permitido entre dos flancos es decir entre dos señales adecuadas para la resincronización de bits.

Este problema se puede resolver mediante el método de bit stuffing o relleno de bit en el cual se inserta un bit de valor complementario cuando se detecta 5 bits de un nivel constante creando un nuevo flanco; esta medida garantiza la mayor capacidad de transporte de información posible con una sincronización suficiente para su correcto funcionamiento.

El método de relleno de bit se aplica a las tramas de datos y remotas en los campos SOF, arbitraje, control, datos y CRC; mientras que las tramas de error y sobrecarga y los campos restantes de las tramas remota y de datos como son el ACK, EOF y el delimitador CRC tienen formatos fijos por lo cual no trabajan bajo el método de relleno de bit.

3.2.2 Tiempo y Sincronización de bit

El protocolo CAN también difiere de otros protocolos en la transmisión de datos síncrona en lugar de la transmisión asíncrona, esto implica una mayor eficacia y al mismo tiempo mejor sincronización.

En la transmisión síncrona existe un solo bit de inicio de trama lo cual normalmente no es suficiente para mantener un punto de muestreo de bit del receptor bien sincronizado con el transmisor.

Para que un receptor mantenga un correcto muestreo de bit al final de una trama es necesaria una resincronización continua o detección del período del reloj de la señal recibida, dicha resincronización que representa la detección del periodo de reloj de la señal recibida es posible mediante los flancos producidos por cambios de nivel en la señal o en caso de que se tenga un nivel constante mediante el flanco que provoca el bit de relleno insertado en la secuencia.

Los segmentos de tiempo durante un intervalo de bit necesarios son 4 y se representan en la figura 3.15.

- Segmento de Sincronización (Sync_Seg)
- Segmento de Retardo de Propagación (Prop_Seg)
- Segmento de fase 1 (Phase_Segl)
- Segmento de fase 2 (Phase_Seg2)

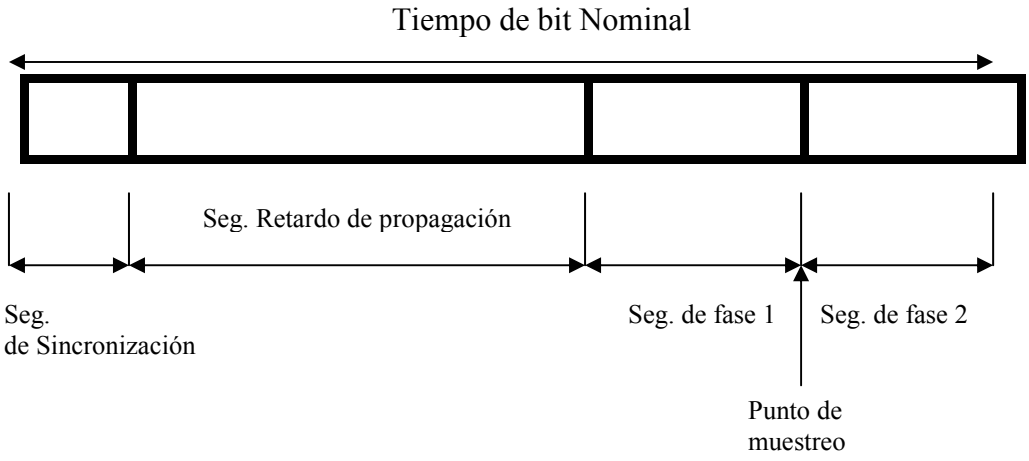


Figura. 3.15. Segmentos que conforman un tiempo de bit

Segmento de Sincronización

Para la sincronización de bits se ha apartado un segmento de tiempo dentro del intervalo de bit, dicho segmento se coloca al inicio del intervalo y se lo conoce como segmento de sincronización.

Segmentos de fase

Debido a las diferencias entre los osciladores del transmisor y receptor es posible que se presente una desincronización entre dos flancos. Este tiempo debe ser compensado mediante pequeños nuevos tiempos llamados segmentos de fase, los cuales se colocan antes y después de un punto de muestreo nominal dentro del intervalo de bit.

Segmento de Retardo de Propagación

Se debe aumentar un segmento de tiempo adicional a los segmentos de resincronización y fase, este segmento toma el nombre de segmento de retardo de propagación y se utiliza para compensar el tiempo que demora en propagarse una señal a lo largo de la línea del bus desde el transmisor al receptor y regresar nuevamente al transmisor (bit de reconocimiento ACK dominante) así como también retardos de señal internos en los diferentes nodos.

La duración de dicho segmento debe ser al menos dos veces mayor al máximo retardo existente entre los osciladores del transmisor y el receptor.

La longitud de los segmentos de tiempo ubicados dentro del intervalo de bit se especifica mediante múltiplos de una unidad básica de tiempo denominada ***tiempo del quántum (t_q) o quanta,*** derivada del periodo del oscilador **(tq = 1/fosc)**. Además se ha fijado como 1 t_q al tiempo que demora el segmento de sincronización dentro del intervalo de bit.

Tiempos Máximos de Duración de los Segmentos de Bit

Los diferentes segmentos que conforman un tiempo de bit pueden ser programados dentro de los siguientes límites:

Sync_Seg: 1 tiempo del quántum

Prop_Seg: de 1 a 8 tiempos del quántum

Phase_Seg1: de 1 a 8 tiempos del quántum

Phase_Seg2: de 1 a 8 tiempos del quántum pero nunca menor al tiempo de procesamiento de información.

Tiempo de procesamiento de la información

El tiempo necesario para calcular el próximo nivel de bit a partir del punto de muestreo se denomina tiempo de procesamiento de información y siempre es menor o igual a 2 tiempos del quantum. El cálculo se refiere a la determinación de un nivel ya sea dominante o recesivo mediante el voltaje diferencial.

Error de Fase

El error de fase se da cuando los flancos para la sincronización no coinciden con el segmento de sincronización (Sync_Seg) y esta dado por los tiempos del quantum existentes desde dicho segmento hacia el flanco como se ve en las figuras 3.16. y 3.17. en donde ***e*** representa el error de fase.

Por lo tanto los valores que puede tomar el error de fase "e" son los siguientes:

e = 0: Si el flanco cae dentro de SYNC_SEG.

e > 0: Si el flanco cae después de SYNC_SEG y antes del punto de muestreo.

e < 0: Si el flanco cae antes del SYNC_SEG y después del punto de muestreo del bit anterior.

Dependiendo del valor del error de fase, se alargan o se acortan los segmentos de fase, si e > 0 el Segmento de fase 1 se alarga en la misma cantidad de tiempos de quantum que marca el error de fase y si e < 0 el segmento de fase 2 se acorta en la misma cantidad de tiempos de quantum que marca el error de fase, los dos casos se muestran en las figura 3.16. y 3.17. para un valor de e = 1.

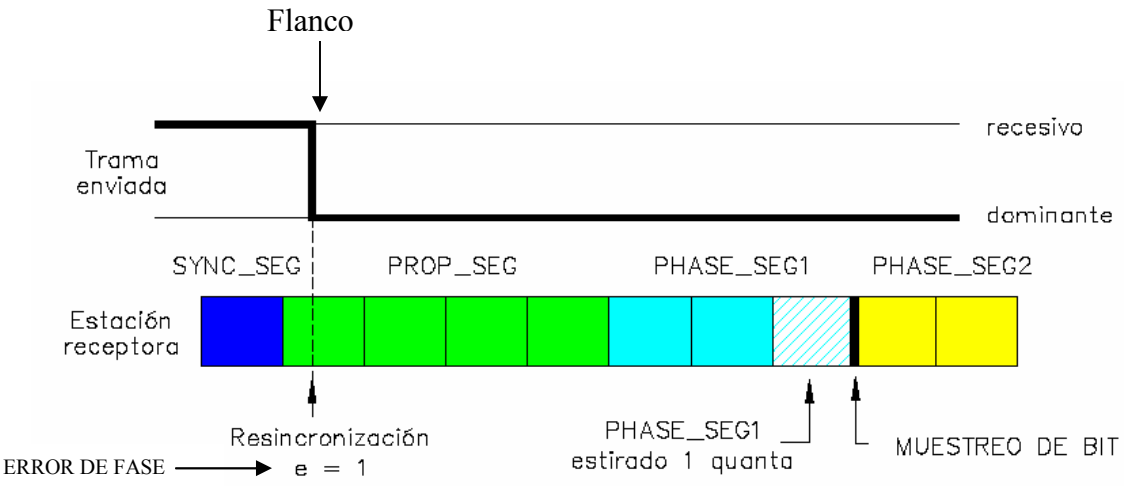


Figura. 3.16. Principio de resincronización cuando el flanco del transmisor cae después del segmento de sincronización de la estación receptora

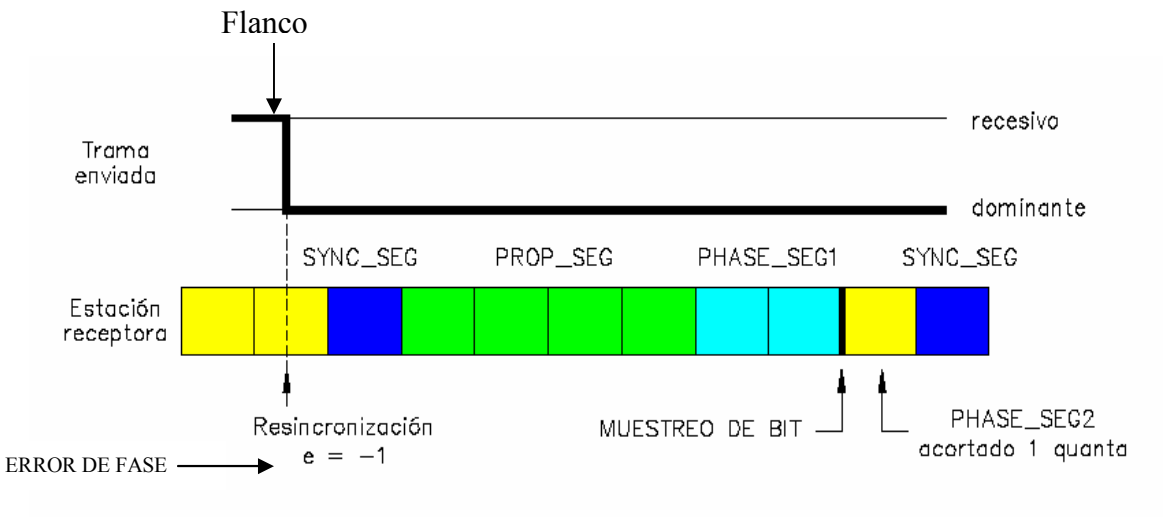


Figura. 3.17. Principio de resincronización cuando el flanco del transmisor cae antes del segmento de sincronización de la estación receptora

Mediante los segmentos de fase 1 y 2 antes y después del punto nominal de muestreo se reserva un lapso de tiempo para la reubicación del punto actual de muestreo durante la resincronización.

Como se aprecia en las figuras 3.16. y 3.17. la resincronización solo se da en los flancos generados durante un cambio de nivel de bit de estado recesivo a estado dominante. Los flancos son detectados mediante una comparación del nivel actual del bus en cada tiempo de quántum con el nivel del bus del punto de muestreo anterior.

Ancho de salto para resincronización (RJW)

El RJW se define como el tiempo máximo, en quantums, que puede ser alargado el PHASE_SEG1 o acortado el PHASE_SEG2 y puede programarse entre 1 y el mínimo entre 4 y el segmento de fase 1: min (4,PHASE_SEG1). Si el error de fase es menor a RJW según su valor positivo o negativo causará un alargamiento o disminución de los segmentos de fase respectivamente, si RJW es igual al error de fase, los segmentos permanecen como se han programado y la resincronización será idéntica a la sincronización anterior y en caso de que la magnitud del error de fase sea mayor a RJW, la

resincronización no puede compensar el error de fase completamente por lo que permanece un error de fase.

Reglas para sincronización

Existen tres reglas principales desarrolladas exclusivamente en la norma [ISO99-1]:

1) Solo existe una sincronización entre dos puntos de muestreo dentro de un tiempo de bit.

2) Solo una señal de flanco generada por un cambio de nivel de recesivo a dominante será utilizada para sincronización.

3) Se puede dar una sincronización en un flanco generado por un cambio de nivel de estado recesivo a dominante durante un periodo de tiempo en el que el bus se encuentre desocupado es decir durante el espacio Inter.-tramas con excepción del primer bit del campo de intermisión.

3.2.3 Simplificación de los Segmentos del Bit

A pesar de que ninguna norma o estándar lo especifica, existen en el mercado algunos controladores CAN que simplifican la programación mediante una combinación de los segmentos PROP_SEG y PHASE_SEG1 conformando un solo segmento denominado TSEG1, en este caso el segmento PHASE_SEG2 toma el nombre de TSEG2 como se muestra en la figura 3.18.

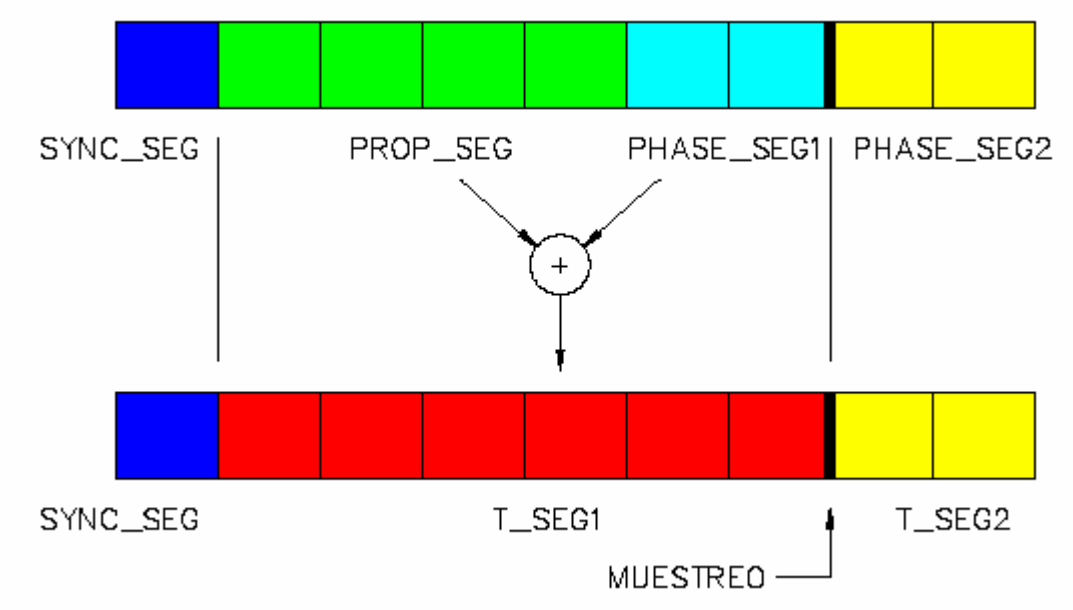


Figura. 3.18. Simplificación de los segmentos

$$T\_SEG1 = PROP\_SEG + PHASE\_SEG1 \quad \textbf{(3-1)}$$

$$T\_SEG2 = PHASE\_SEG2 \quad \textbf{(3-2)}$$

3.2.4 Dimensionamiento de los Segmentos de Tiempo de Bit

El dimensionamiento de los segmentos de tiempo de bit se deben determinar de acuerdo a las características del sistema de comunicación de datos, intervienen principalmente dos parámetros que son la longitud del bus y la tasa o velocidad de transmisión de datos, se puede necesitar una máxima longitud de bus a una determinada tasa de datos o una máxima tasa de datos a una determinada longitud de bus, en estos casos se deben mantener los tiempos de los segmentos de fase al mínimo.

Cuando los segmentos de fase se mantienen en el mínimo, el único valor de error de fase que se puede corregir es el de $|e| = 1$ durante una resincronización, los requerimientos para su buen funcionamiento incluyen osciladores de cuarzo de alta precisión (tolerancias menores al 0.1%).

Para sistemas de automatización industrial generalmente se necesitan este tipo de sistemas con una máxima longitud de bus y máxima tasa de datos como se muestra en la figura 3.19.

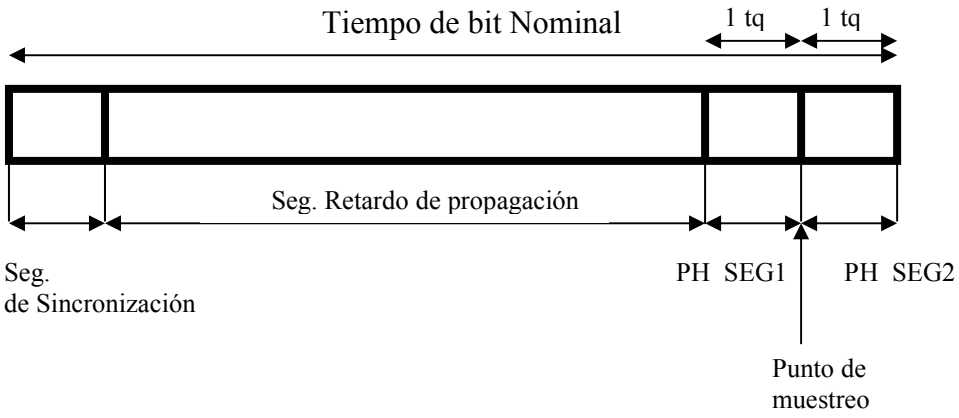


Figura. 3.19. Dimensionamiento de segmentos de bit para condiciones máximas de longitud de bus y tasa de transmisión de datos

Si los requerimientos del sistema de comunicación implican velocidades de transmisión bajas y longitudes de bus cortas los segmentos de tiempo pueden incrementarse, consiguiendo hasta un máximo RJW donde se puede corregir un error de fase |e| = 4.

En este caso se puede utilizar osciladores más baratos como son los de cerámica. Estas características se aplican a sistemas utilizados para el confort electrónico en los automóviles.

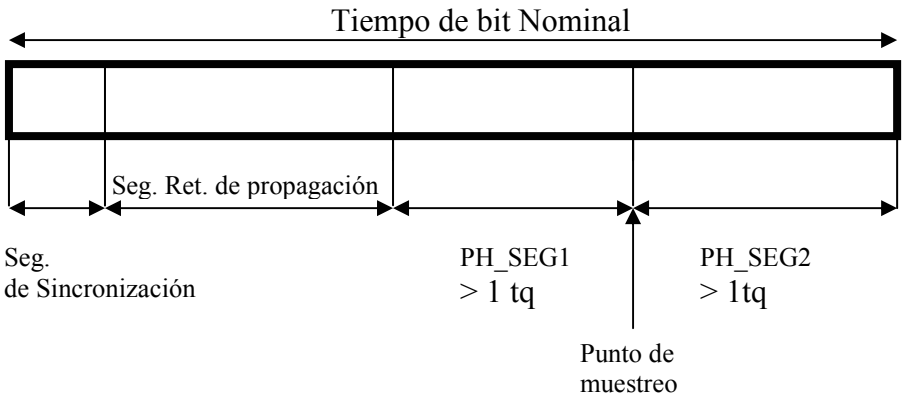


Figura. 3.20. Dimensionamiento de segmentos de bit para condiciones mínimas de longitud de bus y tasa de transmisión de datos

Consideraciones para el dimensionamiento de los segmentos de bit

Inicialmente se debe considerar los requerimientos de tasa de transmisión tiempo del quantum dependiendo del reloj del sistema:

$t_q = 1/f\_reloj$ **(3-3)**

El tiempo de bit debe ser un múltiplo entero del período del reloj y viene dado por:

$t_{bit} = 1/\text{tasa de transmisión}$ **(3-4)**

$t_{bit} = n * t_q$ donde n = 4…25 tiempos del quántum t_q

El valor de n indica que el tiempo de bit siempre va a ser un múltiplo del tq

Un acercamiento para determinar los parámetros que conforman el tiempo de bit es la designación de la longitud del segmento de propagación, para lo cual se debe tomar en cuenta la máxima longitud del bus y el máximo tiempo de retardo interno existente en los nodos, el retardo resultante del recorrido total del bus debe ser convertido en tiempos de quántum redondeado al múltiplo más cercano de t_q

Como la longitud del segmento de sincronización siempre es de 1 t_q entonces:

$t_{Sync\_Seg} = t_q$ **(3-5)**

$t_{phase\_seg1} + t_{phase\_seg2} = (t_{bit} - t_{prop\text{-}seg} - t_q)$

Se selecciona un número similar para los dos segmentos de fase de preferencia el segmento de fase 2 mayor al segmento de fase 1, generalmente:

$t_{phase2} = t_{phase1} + t_q$

La mínima longitud nominal del segmento de fase 2 no debe ser menor al tiempo de procesamiento de la información que dependiendo del controlador CAN varía entre 0 y 2

El RJW se coloca en su máximo valor es decir 4 para aplicaciones de características bajas o menor que 4 a medida que los requerimientos aumenten.

Generalmente se obtiene los valores de los segmentos simplificados t_{TSEG1} y t_{TSEG2} como se explicó en la sección 3.2.3 para programar en dos registros del microcontrolador utilizado.

Ejemplo:

Determinar los parámetros correspondientes al tiempo de bit a programar para obtener la máxima velocidad de transmisión (1 Mbit/s) y la máxima longitud del bus (40m con un retardo de 220 ns) de una red CAN si la frecuencia del reloj es 10 MHz, el retardo interno de cada nodo es de 50 ns y el retardo del circuito receptor es de 30 ns.

t_q = 1/f_reloj = 1/10MHz = 100 ns

t_{bit} = 1/tasa de transmisión = 1/1Mbit = 1000 ns; como $t_{bit} = n*t_q$ entonces n = 10

$t_{Synch\_Seg} = t_q$

t_{Prop_Seg} = (50 + 30 + 220)*2 = 600 ns = 6*tq (el factor 2 se da debido al recorrido de ida y vuelta de la señal)

$t_{Phase\_Seg1} = 1*t_q$ = 100 ns (tiempo mínimo para obtener las condiciones máximas de transmisión que sugiere el problema)

$t_{Phase\_Seg2} = t_{Phase\_Seg1} * t_q = 2*t_q$ = 200 ns

t_{SJW} = mínimo entre t_{Phase_Seg1} y (4*tq) = t_{Phase_Seg1} =100 ns

Finalmente de (3-1) y (3-2):

$$\mathbf{t_{TSEG1} = t_{Prop\_Seg} + t_{Phase\_Seg1} = 700ns}$$

$$\mathbf{t_{TSEG2} = t_{Phase\_Seg2} = 200ns}$$

Estos dos valores serán programados en los registros correspondientes del microcontrolador utilizado, se debe tomar en cuenta el uso de un oscilador de cuarzo para las características de alta velocidad y máxima longitud del bus.

Se puede partir de estos valores y modificarlos en caso de tener diferentes características de tasa de transmisión y longitud de bus hasta obtener resultados óptimos.

Para características de baja velocidad y corta longitud de bus se pueden utilizar osciladores cerámicos y calcular los valores de $\mathbf{t_{TSEG1}}$ **y** $\mathbf{t_{TSEG2}}$ a partir de $t_{Phase\_Seg1} = 4*t_q$ y $t_{sjw} = 4*t_q$

3.2.5 Relación entre Longitud de Bus y Tasa de Datos

La asignación de un valor para el segmento de retardo de propagación (Propo_Seg) define las características de longitud de bus y tasa de datos. Existe un parámetro muy importante a tomarse en cuenta que es el ***máximo tiempo de retardo entre dos nodos td,*** tomando en cuanta que sean los dos nodos más lejanos dentro de la red.

En la figura 3.21. se observa el tiempo de retardo td existente entre dos nodos N y M, en donde el nodo N inicia la transmisión de un nivel recesivo al tiempo t1, este nivel es alcanzado por el nodo M después de un pequeñísimo tiempo de retardo td en el tiempo t2. Si los nodos N y M son los más lejanos dentro de la red, el tiempo td resulta ser el máximo tiempo de retardo.

Basados en la existencia de este tiempo de retardo, si el nodo M transmite un nuevo nivel de señal (dominante) en el tiempo t2, el nodo N va a reconocer este nivel al tiempo t3 es decir después de que transcurra un nuevo tiempo td. Esto significa que hasta este tiempo, el nodo N no puede saber si el nivel de señal que transmitió fue retenido en el bus o si se reemplazó por un nivel dominante del nodo M como es el caso.

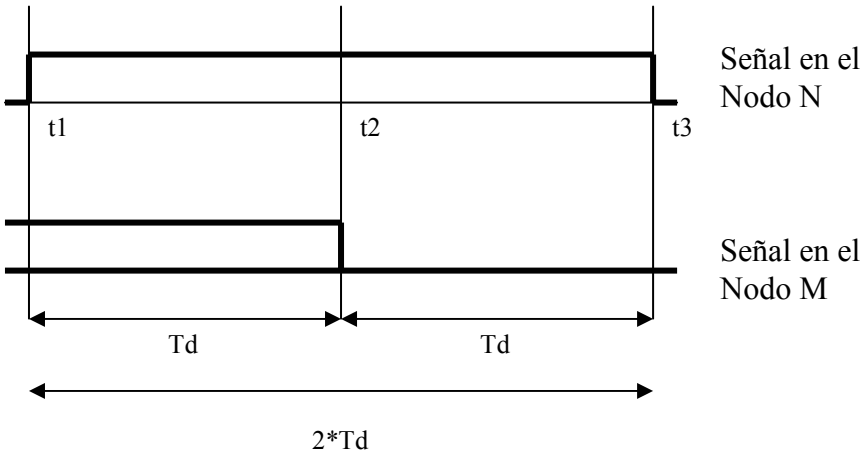


Figura. 3.21. Tiempo de retardo de propagación entre los nodos más lejanos de una red CAN

Por lo tanto el tiempo del segmento de retardo de propagación t_{Prop_Seg} debe ser mayor o igual al doble de Td:

$$t_{Prop\_Seg} \geq 2^* td \qquad \textbf{(3-6)}$$

$$td = t_{el} + t_L$$

$$t_{el} = t_{CAN} + t_{Tranceiver}$$

$$t_L = L^*(t^*p)$$

donde:

t_{Prop_Seg} = Tiempo del segmento de retardo de propagación

td = Tiempo máximo de propagación entre dos nodos

t_{el} = Tiempo de retardo de la señal eléctrica

t_{CAN} = Tiempo de retardo del controlador CAN

$t_{Transceiver}$ = Tiempo de retardo del Transceiver

t_L = Tiempo de retardo debido a la linea de transmisión

L_L = Longitud de la línea de transmisión

tp* = Tiempo de propagación específico del medio

La longitud máxima de un bus de dos líneas se obtiene con un tiempo de propagación específico de señal tp* de 5ns/m aproximadamente y a partir de (3-6) se obtiene:

$t_{Prop\_Seg} = 2 * td$

$0.5\ t_{Prop\_Seg} = td$

$0.5\ t_{Prop\_Seg} = (t_{el} + t_L)$

$0.5\ t_{Prop\_Seg} = (t_{el} + Lmax*(tp*))$

$5*Lmax = 0.5\ t_{Prop\_Seg} - t_{el}$

$$L_{MAX} = \frac{0.5 * t_{Prop\_seg} - t_{el}}{5ns/m} \qquad \textbf{(3-7)}$$

Se puede asumir que el tiempo del retardo de propagación es un factor x del tiempo de bit o Tasa de datos por lo cual se puede resumir:

$t_{Prop\_Seg} = x * t_{Bit}$

$$t_{Prop\_Seg} = \frac{x}{BitRate} \quad \textbf{(3-8)}$$

Por lo tanto de (3-7) y (3-8) se obtiene:

$$L_{MAX} = \frac{\frac{x}{2 * BitRate} - t_{el}}{5ns / m} \quad \textbf{(3-9)}$$

De la ecuación (3-9) se puede obtener la tabla 3.8. tomando en cuenta valores de x=0.85 y t_{el} = 300ns y graficados en la figura 3.22.

Tasa de Datos (Kbits/s)	Longitud del Bus (m)
1000	25
950	29
900	34
850	40
800	46
750	53
700	61
650	71
600	82
550	95
500	110
450	129
400	153
350	183
300	223
250	280
200	365
150	507
100	790
50	1640

Tabla. 3.8. Datos de Longitud de bus y Tasa de Datos x=0.85 y t_{el}=300ns

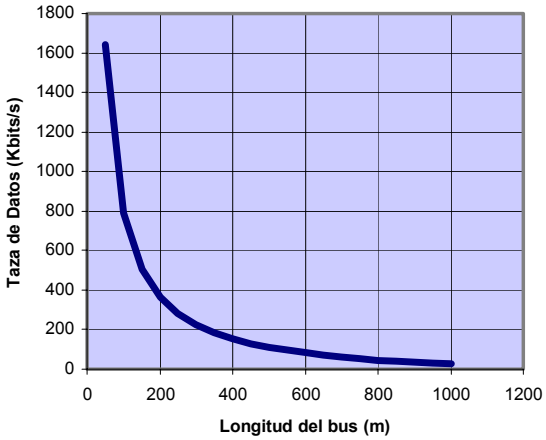


Figura. 3.22. Gráfica de datos tabulados en tabla 3.8.

El gráfico obtenido, a pesar de que se tomaron valores específicos de x y te1, indica el comportamiento general de la velocidad de transmisión con relación a la longitud del bus.

De la fórmula anterior se puede concluir que para longitudes de bus mayores a 100 mts se considera como regla:

$$\text{Bit Rate}_{MAX}\,(\text{Mbit/s}) * L_{MAX}\,(\text{m}) \leq 60 \qquad \textbf{(3-10)}$$

3.3 MEDIOS DE TRANSMISIÓN

El principal aspecto relacionado a los medios de transmisión del protocolo CAN es poder representar un estado recesivo o un estado dominante, el bus debe mantener un estado recesivo cuando todos los nodos se encuentren en dicho estado, y el bus debe tener un estado dominante cuando al menos un nodo se encuentre en estado dominante.

Estos principios se realizan mediante los medios de comunicación eléctricos u ópticos, en los eléctricos según sus niveles de voltaje y en los medios ópticos según el nivel de luz; se interpreta un nivel recesivo como oscuro y un nivel dominante como luminoso.

3.3.1 Medios de Transmisión Eléctricos

Existen tres principales medios de transmisión eléctricos: mediante un par de alambres (par trenzado) manejados diferencialmente con un retorno común utilizado normalmente en aplicaciones industriales; mediante una línea de bus de un solo alambre utilizado en los vehículos principalmente y existe también una solución en la que las señales CAN y la alimentación del sistema actúan en las mismas líneas.

Bus de dos líneas

Permite la transmisión de señales diferenciales.

Las líneas del bus deben terminar a cada lado por una resistencia de un valor recomendado de 120 Ohm para evitar reflexiones de señal, además la transmisión se garantiza a pesar de los bajos niveles de señal.

Además, la interferencia inducida electromagnéticamente puede compensarse por un par trenzado de alambres.

Las normas de capa física CAN más importantes están basadas en un bus de dos líneas.

Mediante diferentes tipos de transceivers que se rigen a diferentes normas se pueden obtener las interfaces adecuadas para diferentes implementaciones.

Bus de una sola línea

Se aplica principalmente para vehículos de motor con una estructura totalmente electrónica.

Se tiene una tierra común disponible para todos los nodos, por lo que se puede utilizar para aplicaciones aisladas.

Como el bus de una sola línea está expuesto a la radio interferencia inducida eléctricamente cuando no está protegido, se necesita un cambio de nivel de señal más grande para mejorar la relación señal-ruido.

El bus CAN de una sola línea se especifica exclusivamente por la norma SAE J2411 y de igual manera se encuentran transceivers para esta aplicación en el mercado.

Transmisión de señales CAN y de alimentación en las mismas líneas

Varios sistemas de buses de campo como el AS-I o PROFIBUS-PA trasmiten la alimentación del sistema en conjunto con los datos sobre las mismas líneas de bus.

Estas características de transmisión también serían deseables para el protocolo CAN pero la transmisión simultánea de poder y datos resulta problemática para el arbitraje de bus que utiliza CAN.

Se han desarrollado soluciones para este tipo de transmisión pero no han dado resultados favorables debido al incremento de costos principalmente debido a la inclusión de circuitos complejos.

3.3.1.1 Parámetros importantes para medios de transmisión eléctricos.

Existen tres parámetros fundamentales que se deben tomar en cuenta principalmente cuando se tiene una gran longitud de bus y son:

- Tiempo de propagación específico
- Sección transversal del cable
- Impedancia de línea

Tiempo de propagación específico

La velocidad de propagación específica de una señal que viaja en líneas libres de pérdida está dada por V_p:

$$V_p = \frac{c}{\sqrt{\mu_r * \varepsilon_r}} \qquad \textbf{(3-11)}$$

Donde

c = Velocidad de la luz ($3*10^8$ m/s)

μ_r = Permeabilidad relativa (determinada por la inductividad específica de la línea)

ε_r = Dielectricidad relativa (determinada por la capacitancia específica de la línea)

El tiempo de propagación de una señal específico está entre 5 y 7 ns/m. Para lograr una tasa de datos alta a una longitud de línea dada se debe mantener una alta velocidad en la línea.

En la tabla 3.9. se muestran las máximas tasas de datos que se pueden obtener para los diferentes valores de tp* con una longitud de 1000 m

Tiempo de propagación específico tp* (ns/m)	**Máxima Tasa de Datos (Kbps)**
5.0	80
5.5	73
6.0	67
6.5	62
7	58

Tabla. 3.9. Máximas tasas de datos para diferentes valores de tp* con una longitud de 1000 m

Sección transversal del cable

Si se tiene líneas de bus largas es muy importante tomar en cuenta la caída de voltaje. La sección transversal del cable a utilizarse está determinada por la caída de voltaje en el nivel de la señal entre los dos nodos más distantes sobre la línea, así como, la unión de las resistencias de entrada de todos los receptores conectados.

La caída de voltaje máxima sobre la línea debe ser tal que el nivel de la señal se interprete fiablemente por cualquier receptor cuando el transmisor envía la señal a un nivel mínimo.

La resistencia distribuida en la línea toma el nombre de R_X y la resistencia total de entrada de los receptores toma el nombre de R_{INTOT} como se muestra en la figura 3.23.

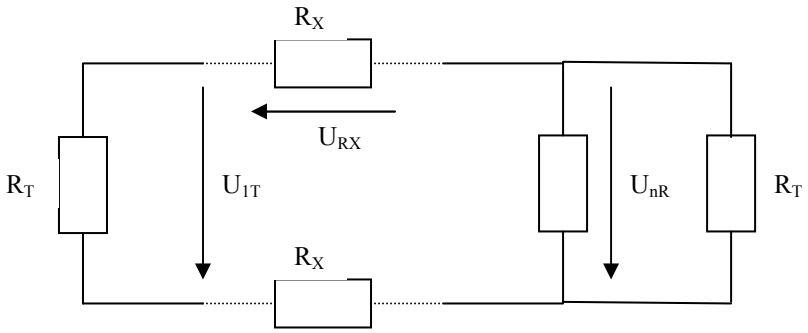


Figura. 3.23. Modelo para obtener el voltaje visto a través del nodo 1 en uno de los extremos de la red hacia el nodo *n* receptor al otro extremo de la red compuesta por n nodos.

El nivel del voltaje U_{nR} en el nodo más lejano de la red (nodo n) mediante un divisor de voltaje está dado por:

$$U_{nR} = U_{1T} * \frac{R_{INTOT} \parallel R_T}{2 * Rx + R_{INTOT} \parallel R_T} \qquad \textbf{(3-12)}$$

donde:

U_{nR} = Voltaje visto por el nodo n

U_{1T} = Voltaje proveído por el nodo 1

R_{INTOT} = Resistencia total de entrada en los receptores

R_T = Resistencia de terminación de línea

R_x = Resistencia distribuida en la línea

Además la resistencia total de entrada en los receptores está dada por:

$$R_{INTOT} = \frac{R_{IN}}{n-1} \quad \textbf{(3-13)}$$

Donde:

R_{IN} = Resistencia de entrada del receptor

n = Número de nodos

Por lo tanto:

$$R_{INTOT} \parallel R_T = \frac{R_T * \frac{R_{IN}}{n-1}}{R_T + \frac{R_{IN}}{n-1}}$$

Desarrollando (3-12):

$$2 * R_x + R_{INTOT} \parallel R_T = U_{1T} * \frac{R_{INTOT} \parallel R_T}{U_{nR}}$$

$$2 * R_x = U_{1T} * \frac{R_{INTOT} \parallel R_T}{U_{nR}} - R_{INTOT} \parallel R_T$$

$$R_x = U_{1T} * \frac{U_{1T}(R_{INTOT} \parallel R_T) - U_{nR}(R_{INTOT} \parallel R_T)}{2 * U_{nR}}$$

Finalmente:

$$R_x = \frac{R_{INTOT} \parallel R_T (U_{1T} - U_{nR})}{2 * U_{nR}} \quad \textbf{(3-14)}$$

Donde:

$$R_x = \rho \frac{L}{A} \quad \textbf{(3-15)}$$

ρ = Resistividad específica de la línea

L = Longitud de la línea

A = Sección transversal de la línea

Reemplazando (3-15) en (3-14) se obtiene finalmente el valor de la sección transversal requerida:

$$A = \frac{2 * \rho * L * U_{nR\min}}{(U_{1T\min} - U_{nR\min}) * R_{INTOT} \parallel R_T} \quad \textbf{(3-16)}$$

Si se toma en cuenta un margen de seguridad del 10% para el mínimo voltaje en el receptor y la mínima resistencia de terminación, se pueden tomar los siguientes valores para el cálculo de la sección transversal:

$U_{1Tmin} = 1.5V$

$U_{nRmin} = 1.05V$

$R_{Tmin} = 100\Omega$

$\rho = 0.0178\ \Omega m/mm^2$

Impedancia de línea

La impedancia de línea está dada por:

$$Z = \sqrt{\frac{L*}{C*}} \qquad \textbf{(3-17)}$$

Donde L* es la inductancia específica y C* la capacitancia específica de las líneas; estos dos parámetros dependen del diámetro y la distancia de las líneas.

Para eliminar reflexiones de señal en el fin de las líneas se debe colocar resistencias de fin de línea en ambos extremos. Dicha resistencia R = Z tiene típicamente el valor de 120Ω.

Debido a la pequeña longitud de trama de datos y la gran capacidad de detección de errores, CAN está preparado para actuar en ambientes con alta interferencia electromagnética. El uso de protecciones y de pares trenzados no se requiere a menos que existan interferencias electromagnéticas extremadamente altas.

3.3.2 Medios de transmisión ópticos

El uso de medios de transmisión ópticos principalmente la fibra óptica plástica cada día toma mayor importancia dentro de las redes CAN debido a su excelente eficacia y a la disponibilidad creciente de tecnología para transmisión óptica. El comportamiento eléctricamente neutro de los medios ópticos es muy útil en aplicaciones potencialmente explosivas.

Debido al acoplamiento directo dentro del medio óptico las líneas de transmisión y de recepción deben proporcionarse por separado, además cada línea receptora debe estar acoplada externamente con cada línea transmisora para asegurar el correcto monitoreo de bits requerido por el protocolo CAN, esta función puede ser implementada mediante un acoplador de tipo estrella.

El uso de acopladores de estrella pasivos es posible con un número pequeño de nodos, ya que están limitados a una división de la potencia transmitida para el número de líneas receptoras, su estructura se muestra en la figura 3.24.

La atenuación de potencia en los acopladores tipo estrella dependen del número de puntos de acople

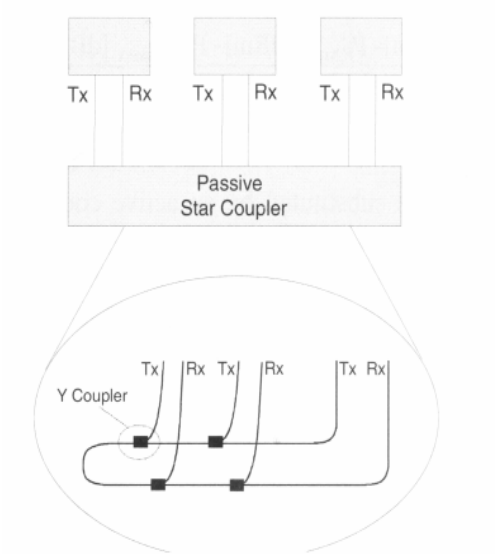


Figura. 3.24. Estructura de una red CAN óptica con acoplador de estrella pasivo

La extensión máxima de una red óptica depende de la potencia disponible y de la potencia perdida debido a los componentes de la red.

La extensión máxima Lopt de una red óptica con un acoplador pasivo tipo estrella resulta de la atenuación a lo largo de la red y puede calcularse según (3-18).

Principalmente la longitud máxima de la red depende de la potencia mínima del transmisor P_{TXmin}, de la máxima atenuación a lo largo de la fibra óptica P_{TLmax} y de la atenuación debido al acoplador pasivo tipo estrella así como de la mínima potencia requerida por el receptor $P_{Rxmin.}$

$$L_{opt}[m] = \frac{P_{TX\,\min}[dBm] - P_{RX\,\min}[dBm] - P_{ACP\,\max}[dB]}{P_{TL\,\max}[dB/m]} \quad \textbf{(3-18)}$$

Si el acoplador pasivo se reemplaza por un acoplador activo entonces ya no es necesario tomar en cuenta la atenuación debido al acoplador, en este caso la longitud de la red depende de la potencia disponible para conectar el nodo transmisor con el acoplador de estrella activo hacia el receptor o conectar el nodo receptor con el acoplador de estrella activo hacia el transmisor.

Normalmente el uso de una sola conexión óptica punto a punto es suficiente como una extensión de la red CAN, es decir se utiliza esta conexión en los sectores de la red que se encuentran ubicados en sectores potencialmente explosivos. El uso de puentes o repetidores de fibra óptica facilitan este tipo de conexiones, de esta forma se puede mantener una buena relación costo-eficacia utilizando una combinación de red eléctrica y óptica.

3.4 TOPOLOGÍA DE RED

Como ya se ha explicado en las secciones anteriores es necesario eliminar las posibles reflexiones de señal en los extremos de las líneas eléctricas mediante la conexión de una resistencia de fin de línea, pero también se debe tomar en cuenta las reflexiones en las líneas que conectan los nodos con el bus. Estas reflexiones se evitan mediante el uso de líneas conectoras cortas en una topología de bus sencilla como muestra la figura 3.25.

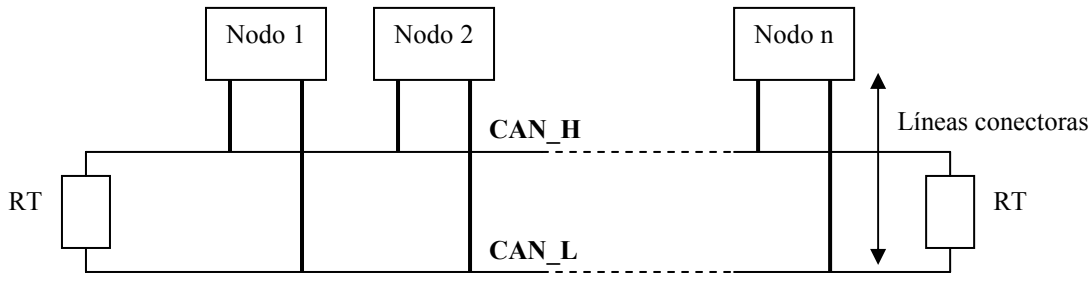


Figura. 3.25. Topología de bus de una red CAN

El estándar ISO99-2 especifica una topología como se muestra en la figura 3.25. con resistencias de fin de línea de 120Ω para una tasa de datos de 1Mbps, con una longitud de bus de 40 mts. y líneas conectoras de 30 cm. Para tasas de datos menores se pueden obtener longitudes mayores de bus y de líneas conectoras.

En aplicaciones industriales a menudo no es posible conectar un nodo hacia el bus mediante líneas conectoras cortas. Mediante una configuración apropiada de los parámetros del tiempo de bit se pueden minimizar las reflexiones de señal ocasionadas por las líneas conectoras, atenuándolas mediante el segmento de retardo de propagación

Una regla para estimar la máxima longitud de las líneas conectoras L_{DC} está dada por (3-19).

$$L_{DC} < \frac{t_{\Pr op\_Seg}}{50 * t_p} \qquad \textbf{(3-19)}$$

Donde:

t_p = Retardo de propagación específico

$t_{PropSeg}$ = Tiempo del segmento de retardo de propagación

De esta manera la longitud máxima acumulada de las líneas conectoras resulta:

$$\sum_{i=1}^{n} L_{DC\,i} < \frac{t_{\Pr opSeg}}{50 * t_p} \qquad \textbf{(3-20)}$$

En diversas aplicaciones no es suficiente con estas medidas ya que se puede adaptar el segmento de retardo de propagación de tal forma que se obtenga la máxima longitud de líneas conectoras y aún así no se obtiene los resultados deseados para aplicaciones específicas. El uso de repetidores, puentes o gateways permiten sobrepasar la limitación

de la topología de bus CAN básica y adaptar una nueva topología para aplicaciones específicas.

Repetidores

La función de un repetidor es transferir una señal eléctrica desde un segmento físico del bus hacia otro segmento regenerando la señal en dicha transferencia. Así un repetidor divide el bus en dos segmentos físicamente independientes.

Con respecto a la propagación de la señal, un repetidor introduce un tiempo de señal de propagación adicional igual al tiempo de retardo.

El tiempo de retardo adicional introducido por un repetidor es aproximadamente de 250 a 350 ns logrando una máxima longitud de línea posible de 50 a 70 mts por repetidor.

Existen en el mercado repetidores de fibra óptica muy modernos que pueden lograr una distancia máxima de 1 Km como el que se muestra en la figura 3.26.

Figura. 3.26. Repetidor CAN de fibra óptica

En el ejemplo de la figura 3.27. se tiene una red CAN de 15 nodos que necesitan una longitud de línea total de 440 mts, con esta longitud de bus la red puede actuar a una velocidad de 150Kbps. Si en la red se colocan repetidores cada 50 mts de longitud, se puede lograr una nueva longitud total de 290 mts y trabajar a una velocidad de 400

Kbps. De esta forma se ha optimizado tanto la longitud como la velocidad de la red como se muestra en la figura 3.28.

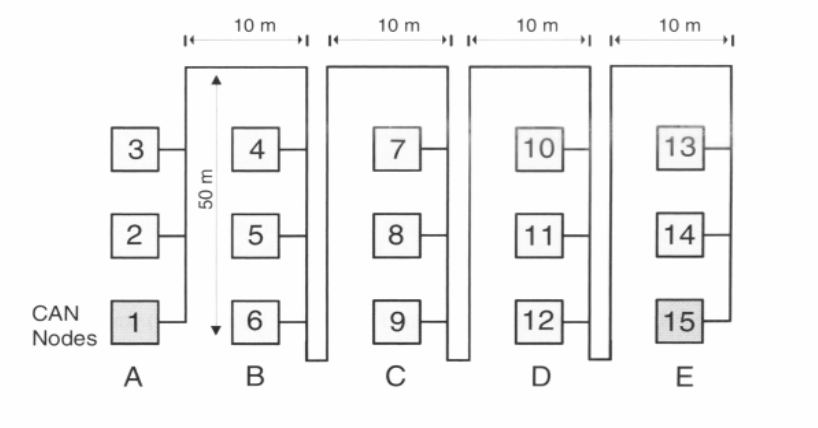


Figura. 3.27. Conexión básica de una red CAN

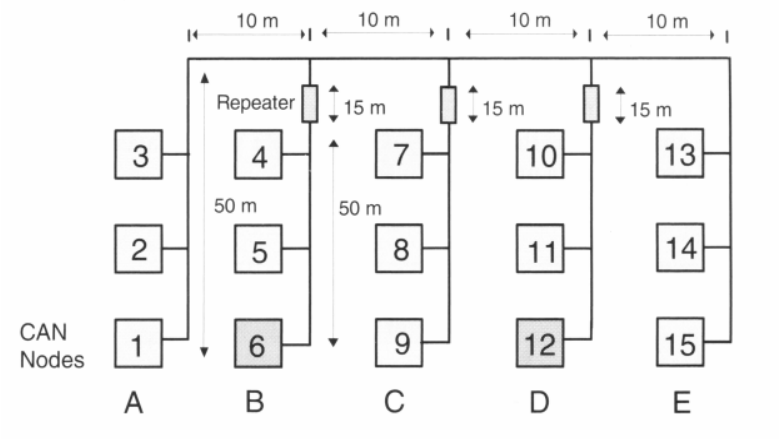


Figura. 3.28. Conexión optimizada de una red CAN mediante el uso de repetidores

La combinación de varios repetidores en una sola unidad llamada acoplador de estrella también se encuentra disponible en el mercado. Este tipo de conexiones permiten una óptima realización de una red CAN con una topología tipo estrella.

Puentes

Los puentes son dispositivos que actúan en la capa de enlace de datos, ya que proveen una función de almacenamiento y envío de mensajes o partes de mensajes en transmisiones independientes entre segmentos de red.

Los puentes difieren de los repetidores ya que almacenan y envían mensajes mientras que el repetidor almacena señales eléctricas lo cual implica una regeneración de la señal

.

Gateways

La función principal de un gateway es la de conectar redes de diferentes protocolos desarrollando el traslado de unidades de datos entre sistemas de comunicaciones diferentes. Por lo general un gateway actúa en la capa 7 según el modelo OSI es decir en la capa de aplicación.

En el mercado se encuentran disponibles diferentes tipos de gateways CAN entre CAN/CanOpen/DeviceNet, RS232/RS485, Interbus, Profibus o Ethernet/TCP-IP.

En la figura 3.29. se muestra un gateway CAN-Ethernet/TCP/IP mediante el cual es posible conectar una red basada en CAN con una basada en Ethernet/TCP/IP a una estación (PC) para labores de control y monitoreo.

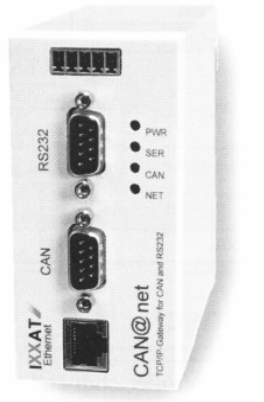


Figura. 3.29. Gateway CAN-TCP/IP para diferentes aplicaciones

En la figura 3.30. se muestra una red CAN con una estructura compleja mediante el uso de repetidores, puentes y gateways.

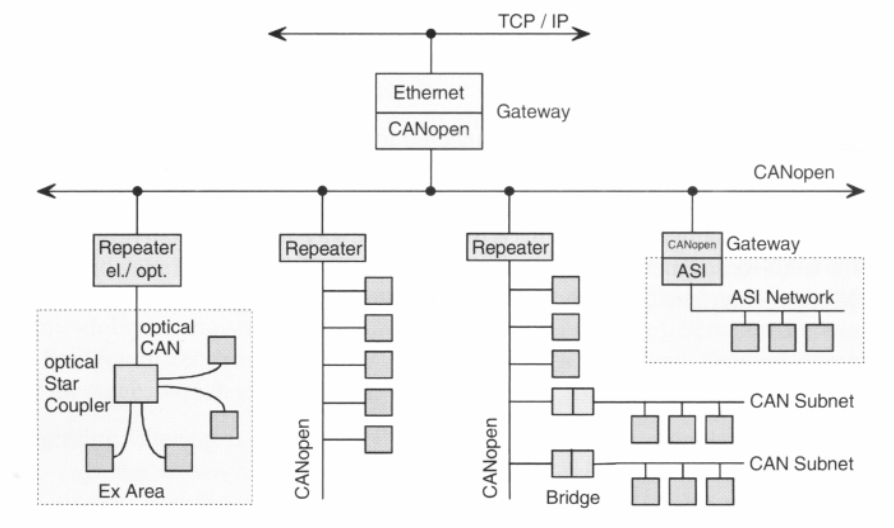


Figura. 3.30. Topología de red mediante el uso de puentes, repetidores y gateways

3.5 CONEXIÓN DEL BUS HACIA EL MEDIO

La conexión entre el medio físico del bus y el controlador o microcontrolador CAN se da mediante circuitos integrados transeptores de señal o transceivers y se los encuentra en gran cantidad en el mercado principalmente para conexión de bus de dos líneas diferencial.

Básicamente el transceiver está compuesto por un amplificador transmisor y un amplificador receptor, mientras que, el controlador CAN consta de un comparador en la recepción y un amplificador a la salida.

Además de estas características principales, un transceiver debe proporcionar otras funciones en el transmisor y en el receptor.

En el lado transmisor:

- Proveer suficiente capacidad de salida
- Proteger al chip controlador contra sobrecarga
- Reducir la radiación electromagnética

En el lado receptor:

- Proveer un nivel de señal recesivo definido
- Proteger al comparador de entrada del chip controlador
- Proveer suficiente sensibilidad de entrada
- Detectar errores en el bus tales como líneas rotas o corto circuitos

Otra función de la unidad de interfaz de bus o transceiver puede ser la del aislamiento galvánico del nodo o del bus; esta función se realiza principalmente mediante el uso de optoacopladores.

3.6 MEDIDAS PARA MEJORAR LA COMPATIBILIDAD ELECTROMAGNÉTICA (EMC)

Existen dos formas de mejorar la EMC en redes CAN:

1) Inmunidad contra interferencia electromagnética inducida
2) Reducción de la potencia electromagnética emitida

La EMC tiene como fin lograr que el receptor detecte apropiadamente los niveles de señal diferencial bajo condiciones de ruido.

Uno de los métodos más importantes para mejorar la EMC es el uso de líneas trenzadas y cubiertas. Esta medida ofrece un gran grado de protección independientemente de la tasa de datos y el número de nodos de la red.

Cuando se trabaja a altas frecuencias principalmente; se puede mejorar la EMC mediante una división de las resistencias de fin de línea. Cada resistencia de fin de línea es dividida en dos resistencias de igual valor y en la mitad de estas dos resistencias se conecta a tierra por medio de un capacitor como muestra la figura 3.31.

Generalmente los valores son:

$Rt/2 = 62\Omega$

C_K = 10 nF...100nF

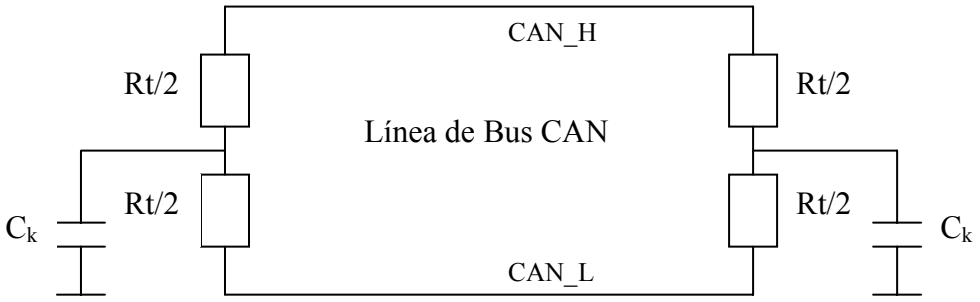


Figura. 3.31. Circuito para mejorar la EMC mediante la división de los terminadores de bus

CAPITULO 4

CONTROLADORES CAN

En la actualidad existe una gran variedad de controladores CAN de diferentes fabricantes como Philips, Intel, Microchip, Motorola, Infineon, Dallas, Atmel, Mitsubishi, Fujitsu, etc.

Todas las funciones para el procesamiento que debe realizar un controlador CAN así como las funciones de soporte del protocolo son realizadas por el controlador CAN y proveídas al HOST CONTROLLER, que se define como el procesador de la aplicación.

Las funciones que debe realizar un controlador CAN se refieren principalmente a las características mencionadas en la capa 1 y 2 del Protocolo CAN como son las siguientes:

- Arbitraje de Bus y Filtraje de mensajes
- Serialización y De-serialización de las tramas a ser enviadas o recibidas
- Cálculo y verificación de la secuencia de redundancia cíclica
- Señalización y detección de errores
- Construcción de los formatos del mensaje CAN
- Inserción y anulación de los bits de relleno “stuff bits”
- Generación y verificación del bit de reconocimiento

- Sincronización de la cadena de bits recibida

4.1 FILTRAJE DE MENSAJES

Como ya se ha visto en los capítulos anteriores, las redes CAN se basan en el principio de broadcasting. Esto significa que todos los mensajes transmitidos en el bus son accesibles para todos los nodos y recibidos por sus controladores CAN. En muchos casos, un nodo en particular está interesado solamente en uno o pocos mensajes de todos los enviados por el bus. Por esta razón es muy apropiado implementar un mecanismo de filtraje de mensajes en el controlador CAN, el cual asegure que el Controlador Host está informado de la recepción de un nuevo mensaje si es actualmente pertinente para el nodo en cuestión. Este mecanismo se llama filtro de admisión y su función se implementa de diferentes formas en los controladores disponibles en el mercado.

Como parte del filtraje del mensaje, se decide mediante los identificadores del mensaje recibido, si éste debe ser almacenado en los buffers de recepción correspondientes del controlador. Para esto el controlador CAN debe estar informado de todos los identificadores de mensajes en los cuales el nodo está interesado.

4.2 CLASIFICACIÓN DE LOS CONTROLADORES CAN

Los diferentes tipos de controladores pueden referirse principalmente a tres categorías:

1) Según su estructura: BasicCAN y FullCAN.

2) Según su longitud del identificador del mensaje: Standard CAN y Extended CAN.

3) Según su grado de integración: Stand Alone CAN y Microcontroladores con controladores CAN integrados.

4.2.1 Controladores CAN según su estructura

Las estructuras BasicCAN y FullCAN están orientadas hacia el tipo de tratamiento o manipulación de mensajes que utiliza un controlador CAN.

4.2.1.1 BasicCAN

Su característica principal es el almacenamiento de todos los mensajes en un buffer de transmisión y un buffer de recepción.

En la ruta de recepción se ejecuta un filtro de admisión antes de que el mensaje sea almacenado en el buffer de recepción (buffer 0). Además, dicho buffer proporciona un nuevo buffer de "imagen" (buffer 1) el cual permite que durante la lectura del mensaje recibido por el procesador, el siguiente mensaje pueda ser recibido al mismo tiempo, por lo cual, se debe asegurar que un mensaje ha sido almacenado antes de que un tercer mensaje sea recibido de lo contrario, se perdería el tercer mensaje.

En el caso de que un nodo BasicCAN se encuentre recibiendo mensajes secuencialmente debido a la rápida transmisión de los otros nodos, no se presenta un problema ya que en el peor de los casos es decir cuando el primer mensaje tiene 8 bytes (recibido en el buffer 0) y el segundo tiene 0 bytes (recibido en el buffer 1), es decir se dispone del mínimo tiempo para leer el mensaje de máxima longitud en el buffer 0 antes de que vuelva a llegar un mensaje en el mismo buffer.

El tiempo mínimo que se demora en llegar el segundo mensaje (59 us a una velocidad máxima de 1Mbps ya que existiría un mínimo de 59 bits incluyendo los 3 bits de los espacios intertrama) es suficiente para que el controlador haya leído y guardado 10 bytes de longitud del mensaje, como el primer mensaje máximo puede tener 8 bytes, nunca existirá pérdida.

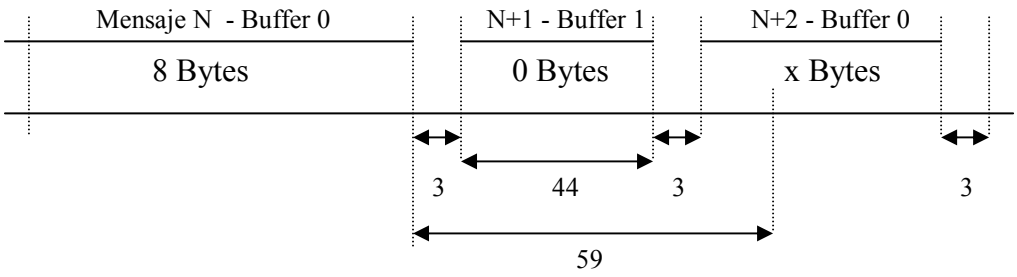


Figura. 4.1 Transmisión de varios mensajes secuenciales BasicCAN en el peor de los casos

La figura 4.1. muestra este caso. Los 59 bits se generan debido a que se transmiten 44 bits que corresponden a los campos de la trama tales como SOF, ID, CRC, ACK, etc, con una longitud de 0 en el campo de datos; 3 bits de espacios intertrama en los dos extremos del mensaje y 9 bits adicionales del tercer mensaje que corresponden al bit de SOF (inicio de trama) y los 8 bits más significativos del campo ID (identificadores), los cuales son necesarios antes de realizar la lectura del mensaje.

Con respecto a la recepción de una trama remota, como el almacenamiento de mensajes en BasicCAN se da en la memoria del Controlador Host, el controlador CAN debe pasar el requerimiento de mensaje remoto hacia el Controlador Host para poder responder. Por último debe proveer el mensaje requerido escribiéndolo en el buffer de transmisión para enviarlo hacia los nodos que lo requieran.

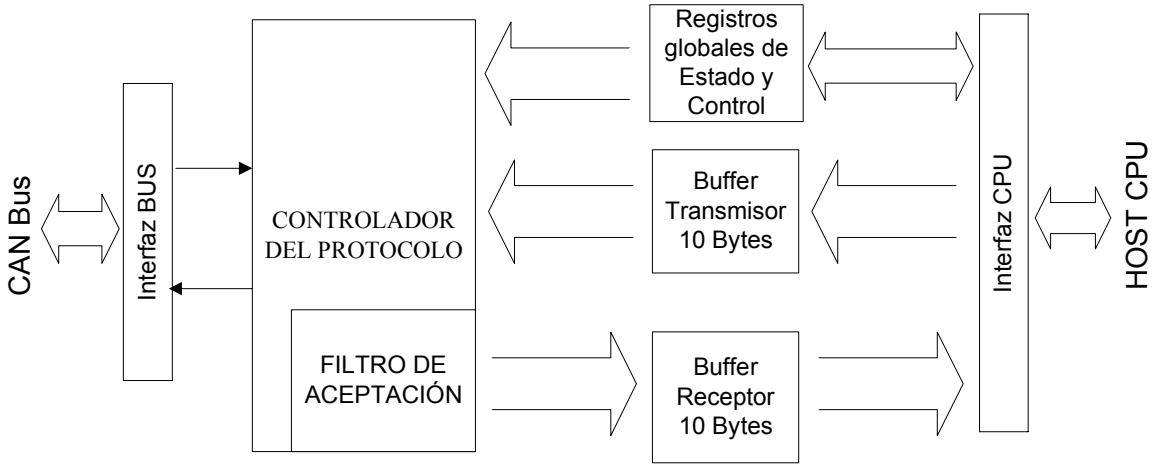


Figura. 4.2. Diagrama de bloques de un Controlador BasicCAN

La figura 4.2. muestra el diagrama de bloques de un controlador CAN y la manera en que interactúan sus registros de Estado y control y sus buffers con el Controlador Host.

4.2.1.2 FullCAN

A diferencia de BasicCAN; el principio de FullCAN define un número de buffers receptores o transmisores. De esta manera cada mensaje a ser recibido o transmitido tiene su propio buffer de mensaje asociado.

La asignación de buffers con los diferentes mensajes es desarrollada mediante el Controlador Host en una fase de configuración inicial. El Host asigna los buffers pero no se encarga de la lectura y almacenamiento de todos los mensajes ya que esta tarea es única del controlador CAN.

Si un buffer de recepción ya contiene un mensaje y un nuevo mensaje con el mismo identificador es recibido, el mensaje anterior es sobre escrito, de tal forma que, los buffers siempre contienen el mensaje actual transmitido con un identificador específico. Este tipo de manipulación de mensajes es ventajoso especialmente cuando se tiene varios mensajes de identificadores diferentes dentro de periodos cortos de tiempo.

Como el proceso de almacenamiento de mensajes no necesita del Host sino solo del controlador CAN, el Procesador Host no se sobrecarga por realizar esta actividad. En BasicCAN se da lo contrario ya que el Controlador Host está forzado a copiar cada mensaje desde el buffer de recepción hacia su lugar de almacenamiento en seguida de su recepción.

En el caso de recibir una trama remota, en FullCAN, es posible que el controlador CAN responda inmediatamente al requerimiento de mensaje sin ninguna intervención de procesamiento del Controlador Host, sino solo una actualización de los datos requeridos.

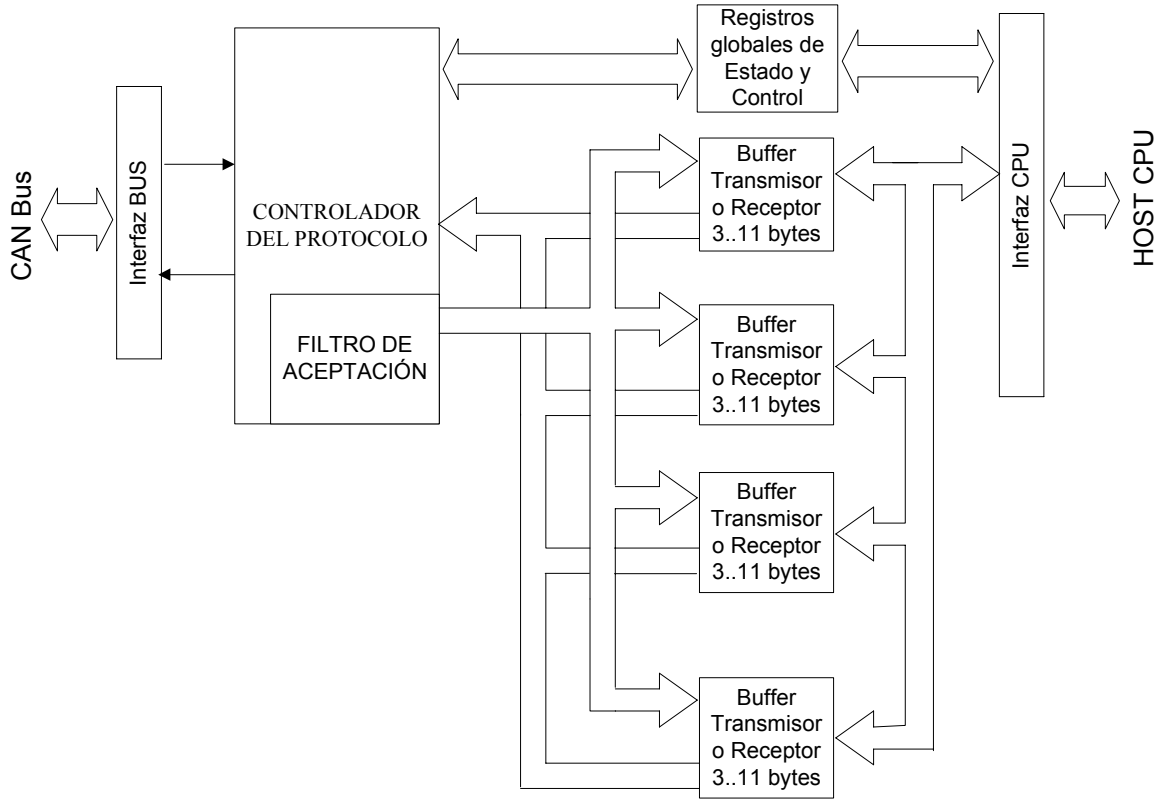


Figura. 4.3 Diagrama de bloques de un Controlador FullCAN

Áreas de aplicación de BasicCAN y FullCAN

El concepto de un Controlador FullCAN parece ser el principio de implementación más poderoso debido al aliviamiento o poca sobrecarga del Controlador Host y el almacenamiento de mensajes residente en el controlador.

Por otro lado BasicCAN es limitado en relación a FullCAN por lo que permite una implementación menos costosa con una buena efectividad.

El principio original de la implementación FullCAN está primeramente limitado con respecto al máximo número de mensajes a ser recibidos por un nodo, lo cual no es una limitación para el principio BasicCAN. Por estas razones, los dos principios de implementación han sido incluidos en la segunda generación de microcontroladores.

El uso del principio FullCAN es necesario y conveniente para altas tasas de datos, pequeñas cantidades de mensajes recibidos por nodo y Controladores Host menos potentes. Se debe tener en cuenta que para aliviar al Controlador Host se debe realizar una sobre escritura de mensajes.

En caso de que los nodos deban ser capaces de recibir una larga cantidad de mensajes o si es tolerable una alta carga en el Controlador Host se puede utilizar el principio de BasicCAN.

Puede ser utilizado uno o los dos principios de implementación CAN en la misma red ya que el principio de funcionamiento del protocolo es el mismo; solo varía la forma de manipular y almacenar los mensajes.

4.2.2 Controladores CAN según la longitud del identificador del mensaje

La clasificación de controladores CAN según la longitud del identificador del mensaje está relacionada con las versiones del protocolo CAN. La versión 2.0A maneja la trama estándar y la versión 2.0B maneja tanto la versión estándar como la versión extendida del protocolo CAN. Las dos versiones se diferencian principalmente por la longitud del identificador del mensaje; en la versión estándar se manejan 11 bits identificadores que pueden abarcar 2048 tipos de mensajes diferentes mientras que en la versión extendida se manejan 29 bits identificadores que pueden abarcar hasta 512 millones de mensajes diferentes como muestra la tabla 1.1.

Versión 2.0A		**Versión 2.0B**			
CAN Estándar		CAN Estándar		CAN Extendido	
11 bits	2048 mensajes	29 bits	2048 mensajes	29 bits	512` mensajes

Tabla. 4.1. Versiones CAN

4.2.3 Controladores CAN según el Grado de Integración

Según el grado de integración de los controladores CAN se encuentran dos tipos: Controladores CAN Stand Alone y Controladores CAN integrados en microcontroladores.

4.2.3.1 Controladores CAN Stand Alone

El término Stand Alone se aplica cuando cierto dispositivo, en este caso un controlador, dispone de una funcionalidad específica, en este caso la comunicación CAN, y no dispone de otras características de procesamiento de datos como la mayoría de microcontroladores.

Los Controladores CAN Stand Alone únicamente proveen las características de almacenamiento de mensajes. Los registros y buffers de mensajes están normalmente mapeados en espacios de memoria y direccionados como chips de memoria RAM normales. La figura 4.4. muestra el diagrama de bloques de un nodo CAN basado en un Controlador Stand Alone.

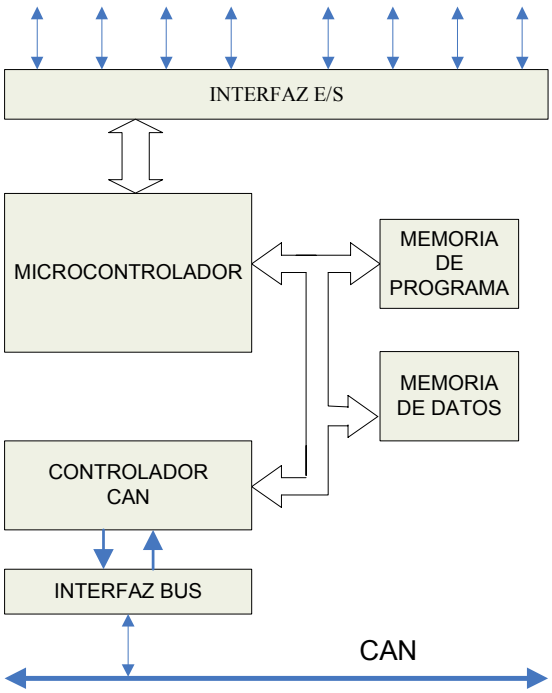


Figura. 4.4. Diagrama de bloques de un nodo CAN basado en un Controlador Stand Alone

4.2.3.2 Controladores CAN integrados en microcontroladores

Los microcontroladores CAN han tomado gran importancia en la actualidad ya que la mayoría de fabricantes de microcontroladores están incluyendo variantes con interfaces CAN integradas.

Los microcontroladores con una interfaz CAN integrada son convenientes para nodos que conforman una sola red CAN y los espacios físicos son reducidos, además deben tener una suficiente capacidad para desarrollar funciones de control local y procesamiento de datos.

Los microcontroladores con **2 o más módulos CAN** han tomado gran importancia en los casos en que se deben enlazar varias redes CAN en un solo sistema, la industria automotriz ha desempeñado un papel muy importante en este aspecto ya que se realizan dos tipos de sistemas, uno de alta velocidad relacionado a las funciones del motor del vehículo y uno de baja velocidad relacionado con el cuerpo del vehículo y su confort electrónico. Este tipo de sistemas tiene una cantidad moderada de nodos y normalmente utiliza más de una red CAN para conformar el sistema vehicular completo.

Para conectar dos o más redes CAN, la conexión de los nodos debe proporcionar una función de puente. Normalmente el puente también desarrolla filtros de mensajes entre las redes conectadas con el fin de reducir la carga del bus en los diferentes segmentos de red o para transferir solo los mensajes que son de interés para los segmentos de red receptores.

La figura 4.5. muestra el diagrama de bloques de un nodo CAN basado en un microcontrolador con un controlador CAN integrado.

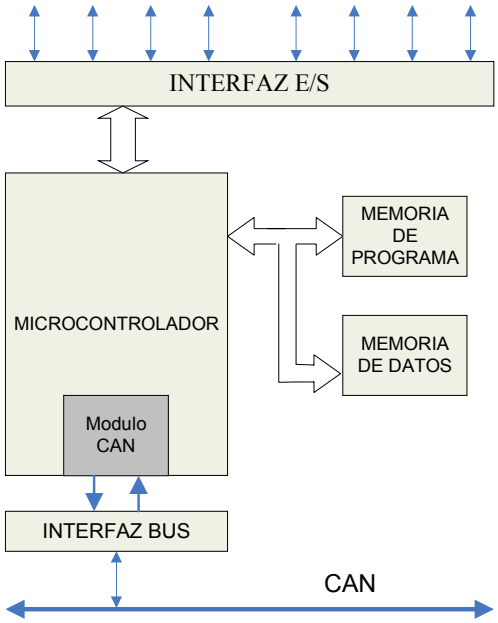


Figura. 4.5. Diagrama de bloques de un nodo CAN basado en un microcontrolador con un módulo CAN integrado

Registros de propósitos especiales

En todos los controladores CAN ya sea de tipo Stand Alone o microcontroladores con CAN integrado, deben existir una serie de registros además de los buffers de mensajes para la configuración, monitoreo y control de todo el controlador CAN. Dichos registros se utilizan principalmente para las siguientes funciones:

- Indicación del estado actual del controlador como son: reset, transmisión, recepción o estado de error.

- Control del estado del controlador en la fase de inicialización luego de un reset general.

- Configuración de los modos de operación del controlador tales como la activación y desactivación de las interrupciones.

- Configuración de los parámetros de tiempo.

- Configuración de la sección del transceiver interna.

Adicionalmente pueden existir otros registros dependiendo de las características especiales que ofrezca el controlador.

4.3 CONTROLADORES CAN "STAND ALONE" DISPONIBLES EN EL MERCADO

Los primeros controladores CAN desarrollados fueron el Philips PCA82C200 (que posteriormente dio lugar al SJA1000) y el Intel 82527. Estos controladores son los ejemplos más populares de controladores CAN tipo Stand Alone.

4.3.1 Philips SJA1000

El primer controlador CAN diseñado por Philips fue el PCA82C200. Los crecientes requerimientos de las redes CAN crearon la necesidad de que el controlador CAN realice una contribución sustancial para la reducción de las limitaciones de tiempo real en el Procesador Host, si éste debe almacenar temporalmente varios mensajes en una cola.

Para abastecer estos requerimientos Philips creó el controlador CAN SJA1000 (figura 4.6.) que es totalmente compatible con el PCA82C200 en términos de software. Dicho controlador junto con el INTEL 82527 se convirtieron en la base para el desarrollo de diversos controladores y microcontroladores CAN de otros fabricantes; de igual forma los mecanismos de filtraje de mensajes, análisis de error y sus diversas aplicaciones se tomaron como base para el resto de controladores y microcontroladores CAN.

El controlador SJA1000 provee una arquitectura extendida BasicCAN que soporta protocolos CAN de alto nivel y especialmente contiene un buffer receptor de 64 bytes, el cual reduce considerablemente las funciones requeridas por el Controlador Host.

El SJA1000 además permite trabajar bajo la versión extendida 2.0B del protocolo CAN mediante el cambio de un bit designado para esta función. Dicho modo de funcionamiento en el SJA1000 toma el nombre de modo PeliCAN.

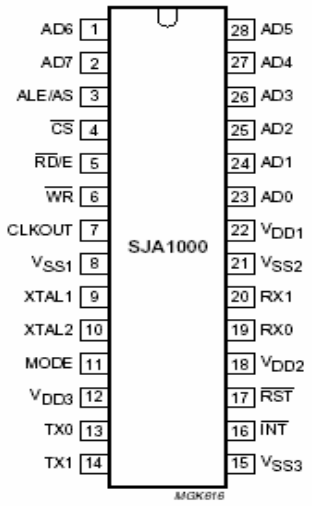


Figura. 4.6. Controlador CAN Stand Alone Philips SJA1000

4.3.1.1 Modo BasicCAN

En el modo BasicCAN existe un solo buffer de transmisión y uno de recepción mediante los cuales se pueden transmitir o recibir mensajes respectivamente.

El buffer receptor se puede ver como una cola o estructura FIFO (First In First Out) en la cual solo el primer mensaje recibido en cada caso puede ser visto por el Controlador Host y cada vez que se ingresa un mensaje el último es eliminado.

Los dos buffers están compuestos por 10 bytes de los cuales dos son utilizados para la descripción de mensajes en los que se incluyen los 11 bits identificadores, 4 bits que determinan la longitud del campo de datos y un bit (RTR) que determina el tipo de trama ya sea de datos o remota. Los 8 bytes restantes son utilizados para el campo de datos.

La figura 4.7. muestra el esquema general de la memoria del controlador CAN Philips SJA y la figura 4.8. muestra el formato de los buffers de mensajes del mismo controlador, toda la información contenida en estos buffers es transferida entre el Host y el Controlador CAN.

El control de acceso a los buffers de mensajes es desarrollado mediante los bits correspondientes al registro global de estado/control que se encuentra ubicado en los primeros 9 bytes del mapa de memoria como muestra la figura 4.7.

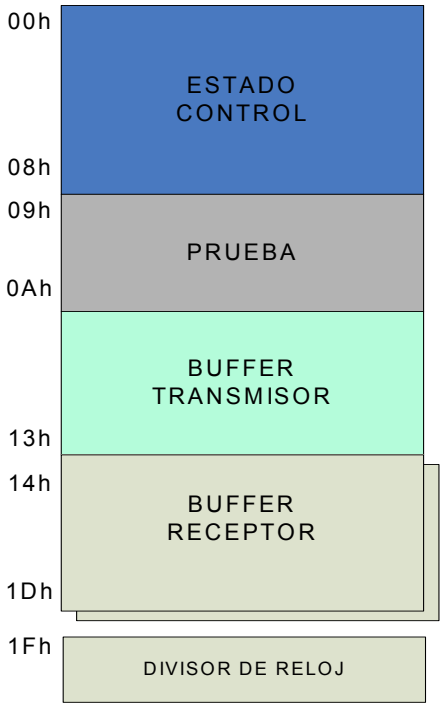


Figura. 4.7. Diseño de memoria del Controlador Philips SJA 1000

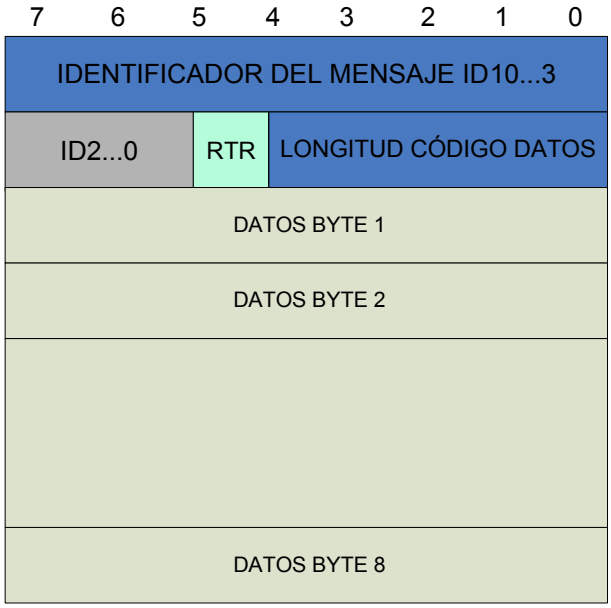


Figura. 4.8. Formato de los buffers del Controlador Philips SJA1000

Registros de Estado/Control

Los Registros de Estado/Control están compuestos por 9 bytes, cada byte representa un registro con funciones específicas asociadas. Los 9 registros son los siguientes: Control, Comando, Estado, Interrupción, Código de Admisión, Máscara de Admisión, Tiempo de Bus 0, Tiempo de Bus 1 y Control de Salida.

Para lograr la comunicación se deben especificar los parámetros relacionados con el tiempo de bit. Dichos parámetros se encuentran en los **Registros Tiempo de Bus 0 y 1** como se muestra en la figura 4.9. Dentro de los parámetros del tiempo de bit se debe programar todos los segmentos de tiempo involucrados mediante la conformación de los segmentos Tseg1 (Prop_Seg + Phase_Seg1) y Tseg2 (Phase_Seg2) como se detalla en el Capítulo 3 sección 3.2.3

El valor del punto de muestreo (Simple Point) también es tomado en cuenta en el Registro de Tiempo de Bus 1. Tseg1 y Tseg2 resultan ser un múltiplo del periodo del oscilador (como se detalla en el Capítulo 3) cuya longitud es determinada mediante el llamado BRP (Baud Rate Prescaler) en el Registro de Tiempo de Bus 0. En dicho registro también interviene el ancho de salto de sincronización (SJW).

El controlador SJA1000 tiene integrado internamente un transceiver, que consta principalmente de dos entradas Rx0 y Rx1, un comparador, un circuito para la lógica de control de salida y dos salidas Tx0 y Tx1. El transceiver puede ser configurado de diferentes maneras. El tipo de salida es configurado mediante los bits de modo (M0, M1) en el **Registro Control de Salida**, existen 3 modos posibles:

- Normal Output
- Clocked Output
- Biphase Output

R. Control	Test Mode	Sync		Overrun Int. Enable	Error Int. Enable	Tx. Int. Enable	Rx. Int. Enable	Rst. Req.
R. Comando				Goto Sleep	Clear Overrun Status	Rel. Rx. Buffer	Abort Tx.	Tx. Request
R. Estado	Bus Status	Error Sstatus	Tx Status	Rx Status	Tx. Comp Status	Tx. Buffer Acces	Data Overrun	Rx. Buffer Status
R. Interrupción				Wake Up Int.	Overrun Int.	Error Int.	Tx. Int.	Rx. Int
Codigo de Admisión	AC 7	AC 6	AC 5	AC 4	AC 3	AC 2	AC 1	AC 0
Máscara Admisión	AM 7	AM 6	AM 5	AM 4	AM 3	AM 2	AM 1	AM 0
Tiempo de Bus 0	SJW 1	SJW 0	BRP 5	BRP 4	BRP 3	BRP 2	BRP 1	BRP 0
Tiempo de Bus 1	SAM	TSEG2 2	TSEG2 1	TSEG2 0	TSEG1 3	TSEG1 2	TSEG1 1	TSEG1 0
Control de Salida	OC TP1	OC TN1	OC POL1	OC TP0	OC TN0	OC POL0	OC Mode1	OC Mode0

Figura. 4.9. Registros de Estado y Control Philips SJA 1000

Normal output

La cadena de bits es transmitida en las dos salidas Tx0 y Tx1. Para esto se pueden configurar el modo de operación de los transistores (PullUp, PullDown o PushPull) así como la polaridad de las salidas.

Clocked output:

La cadena de bits es transmitida mediante una salida y la señal del reloj mediante la otra. La frecuencia del reloj es determinada mediante los parámetros de los registros de tiempo de Bus.

Bi_phase output

En este modo, los bits dominantes del mensaje son enviados alternativamente en una de las dos salidas.

Mediante los bits restantes del Registro de Control de Salida (figura 4.9.) se pueden especificar el comportamiento de las dos salidas Tx0 y Tx1. Los bits OCPOL0 y OCPOL1 determinan si un bit dominante (o recesivo) como una señal de nivel bajo o alto. OCTP0 y OCTP1 determinan si la salida asociada es manejada activamente a un nivel alto o no. De igual forma OCTN0 y OCTN1 determinan si la salida es manejada activamente hacia un nivel bajo.

La entrada del comparador del transceiver interno puede ser omitida, de esta manera se pueden recibir las señales directamente desde un transceiver externo, lo cual disminuye el tiempo de retardo de señal interno del controlador.

Filtraje de Mensajes

Este controlador permite un mecanismo simple de filtraje de mensajes, en dicho mecanismo no se puede especificar los mensajes individuales a ser aceptados por el

controlador pero si permite la especificación de un rango de identificadores de mensajes limitado. De esta forma solo los mensajes que se encuentren dentro del rango serán aceptados y enviados al buffer para su almacenamiento.

Antes de que el controlador sea inicializado, se deben configurar dos registros: Registro de Código de Admisión y el de Máscara de Admisión.

El **Registro de Código de Admisión** contiene los 8 bits más significativos (ID10-ID3) del identificador del mensaje para que dicho mensaje tenga la opción de pasar el filtro y ser tomado en cuenta por el controlador.

El **Registro de Máscara de Admisión** determina los bits del registro de código de admisión que realmente son relevantes para el filtraje de mensajes, es decir determina cuales bits del Registro de Código de Admisión deben coincidir exactamente con el identificador de un mensaje recibido para ser aceptado por el nodo. En éste registro los bits relevantes son los ceros “0” y los irrelevantes son los unos “1”.

Como el filtro solo ocurre en los 8 bits más significativos del identificador del mensaje, los 3 bits (ID2-ID0) no se involucran en el filtro dando un total de 8 posibles combinaciones por lo cual el controlador no aceptará un solo mensaje sino por lo menos 8 mensajes en caso de que todos los bits del filtro hayan sido relevantes.

En caso de que todos los bits del registro de máscara de admisión se ponen en “0” entonces pasarán el filtro todos los mensajes que tengan el valor 0 en los identificadores del 0 al 8, pero en caso de que un bit del registro de máscara se ponga en uno como en la figura 4.10, este es irrelevante, por lo tanto van a pasar todos los mensajes con identificadores del 0 al 15.

Si se requiere que todos los mensajes posibles pasen el filtro es decir 2048 mensajes; se deben poner en uno todos los bits del registro de máscara de admisión o sea en el valor 255, de esta manera, todos los bits son irrelevantes y no existiría ningún proceso de

filtraje. Este valor es importante para nodos que realizan funciones centrales que por lo general necesitan conocer sobre todos los mensajes existentes.

Para utilizar el filtraje de mensajes BasicCAN del controlador SJA1000, los identificadores, mediante los cuales los nodos van a transmitir sus mensajes deben estar asignados de acuerdo a cierto patrón.

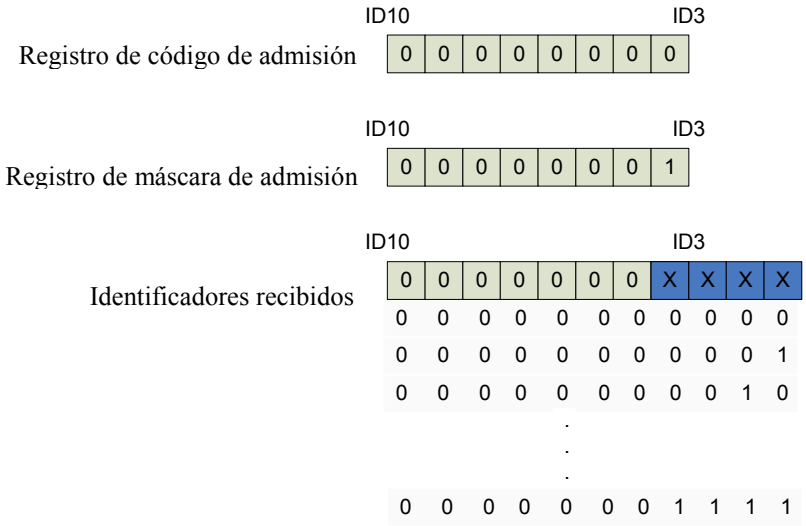


Figura. 4.10. Principio de filtraje de mensajes en BasicCAN

Control de transmisión

Después de configurar los parámetros de tiempo de bus, control de salidas y filtro de admisión, el controlador puede iniciar sus funciones. El inicio o reinicio del controlador está hecho mediante el estado del bit Rst.Req (Reset Request) en el **Registro Global de Control.**

Con el inicio del controlador, éste recibe todos los mensajes transmitidos en el bus y acepta los mensajes dentro de su buffer receptor en caso de que estos hayan pasado el filtro de admisión. Si se tiene algún mensaje disponible para el Controlador Host, se determina mediante el estado del bit de estado de buffer receptor en el registro global de

estado. El Controlador Host puede entonces leer el mensaje del buffer receptor que dicho mensaje sea distorsionado por la llegada de nuevos mensajes.

Como ya se ha explicado el buffer de recepción actúa como una estructura FIFO (First In First Out) en la que se puede acceder solamente al mensaje más reciente. Si el Controlador Host ya ha leído un determinado mensaje lo indica mediante el estado del bit "release receive buffer" en **el Registro de Comandos**.

Cuando existe un desbordamiento o sobrecarga de datos en la estructura FIFO de manera que el Controlador Host no alcanza a leer todos los mensajes provocando la pérdida de algunos de ellos, se indica un estado de sobrecarga mediante el bit "Data Over-run" en el **Registro de Estado.**

Para transmitir un mensaje, éste es copiado en el buffer transmisor y se configura automáticamente el bit "Transmisión Request" en el Registro de Comandos. Si el controlador fue capaz de enviar el mensaje, se configuran los bits "Transmisión complete status" (Estado de Transmisión Completa) y "Transmit buffer Access" (Acceso a Buffer Transmisor) en el Registro de Estado. La señalización de una transmisión exitosa mediante estos dos bits se hace necesaria ya que se puede abortar una transmisión de un mensaje que ya ha accedido al buffer transmisor pero no se envió debido a la transmisión de otros mensajes de mayor prioridad por los otros nodos.

Para este propósito se configura el bit de aborto de transmisión "transmissión abort" en el Registro de Comandos, el cual facilita el funcionamiento de prioridad de mensajes ya que indica si un mensaje ha accedido al buffer transmisor (bit de Acceso a Buffer Transmisor) pero no se ha completado la transmisión (bit de Estado de Transmisión Completa) debido a la existencia de mensajes con mayor prioridad en el bus. Cuando el bit de aborto de transmisión vuelve a su estado original, se vuelve a leer el bit de acceso al buffer transmisor y se completa la transmisión del mensaje que quedó en espera siempre y cuando no exista nuevamente mensajes de mayor prioridad en el bus. Finalmente la transmisión se verifica mediante el bit de estado de transmisión completa.

Si se producen continuamente errores durante la transmisión o recepción de un mensaje, el valor de los contadores de error (Capítulo 2) incrementan y el bit "Error Status" (Estado de error) es configurado automáticamente por el controlador en el Registro de Estado. Este bit indica una tasa de error elevada en el nodo es decir que uno de los contadores está rebasando el límite permitido es decir más de 96. En caso de que los contadores de error lleguen a su valor crítico máximo es decir 255, el nodo se pone en estado de "Bus off", en este caso el nodo no actúa más en el bus y el controlador lo representa configurando el estado del bit "Buss Off Status" .

Interrupciones

Todos los eventos descritos anteriormente como son la recepción de un mensaje, sobrecarga de datos, la transmisión exitosa de un mensaje, los excesos en los contadores de error, la transición al estado Buss Off y el evento "Wake Up" en el modo Sleep del controlador pueden ser notificados al Controlador Host por medio de interrupciones.

La habilitación individual de cada una de las interrupciones se da mediante la configuración de los bits de habilitación asociados a cada interrupción en el **Registro de Control**. En caso de que se produzca una interrupción el Controlador Host puede determinar su tipo mediante la lectura del registro de interrupciones.

4.3.1.2 Modo PeliCAN

El modo PeliCAN del controlador Philips SJA1000 soporta la especificación CAN 2.0B por lo tanto transmite y recibe mensajes en el formato CAN extendido. Para esta tarea su comportamiento es el mismo que en el modo BasicCan pero con las siguientes diferencias:

- Tamaño de buffers de 13 bytes.
- Filtro de admisión más poderoso con código de admisión y máscara de admisión de 4 bytes de tal forma que todos los bits identificadores participan en el proceso de filtraje.

- Dos modos de filtros:

1) Single Filter Mode: Se filtran los 4 bytes del identificador del mensaje incluyendo el bit RTR

2) Dual Filter Mode: Se filtran los 4 bytes del identificador del mensaje dividiéndolos en dos grupos de 2 bytes cada uno. El mensaje será aceptado cuando haya pasado por uno de los dos filtros.

4.3.2 Intel 82527

El controlador de comunicaciones serial Intel 82527 desarrolla características de comunicación de acuerdo a las especificaciones CAN 2.0A y 2.0B para BasicCAN y FullCAN. El controlador Intel 82527 desarrolla todas las funciones de comunicación serial tales como transmisión y recepción de mensajes, filtrado de mensajes e interrupciones con una mínima interacción del Controlador Host. Todas las funciones soportan las tramas de mensajes en formato estándar (11 bits identificadores) y extendido (29 bits identificadores).

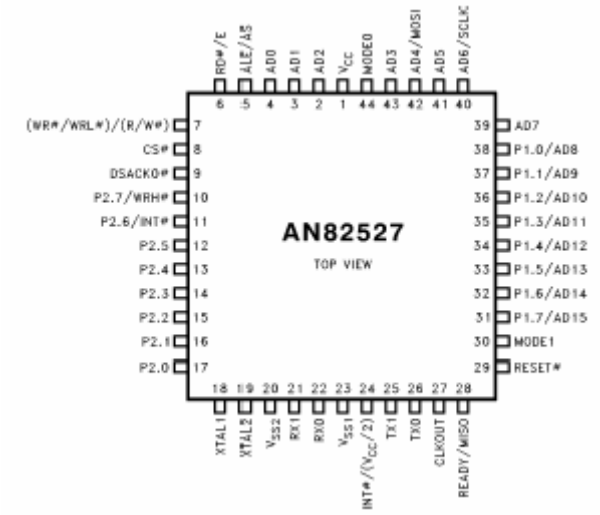


Figura. 4.11. Controlador Intel 82527

Este dispositivo tiene una amplia capacidad de buffers de almacenamiento que consiste en 15 buffers de una longitud fija de 8 bytes, de los cuales, 14 funcionan como una implementación FullCAN es decir que pueden ser configurados como buffers de transmisión o recepción. Se dispone además de un buffer que actúa según el principio BasicCAN. Dicho buffer incluye el llamado buffer de fondo que permite recibir un nuevo mensaje mientras el primero está siendo leído.

Antes de desarrollar cualquier transferencia de mensajes, se deben configurar los parámetros de tiempo de bus en el **Registro Configuración de Bus** en el cual se puede anular las entradas del comparador interno, que regula los niveles de la señal, para conectar directamente la entrada Rx0 con los niveles TTL desde un transceiver externo, lo cual se puede realizar para mayor seguridad Además se puede configurar si la llegada de un bit dominante tiene un nivel alto o bajo.

El controlador CAN Intel 82527 está formado por seis bloques funcionales: Interfaz lógica CPU, Control Lógico CAN, Buffers de mensajes RAM, Puertos de entrada/salida y salida de reloj como muestra la figura 4.12.

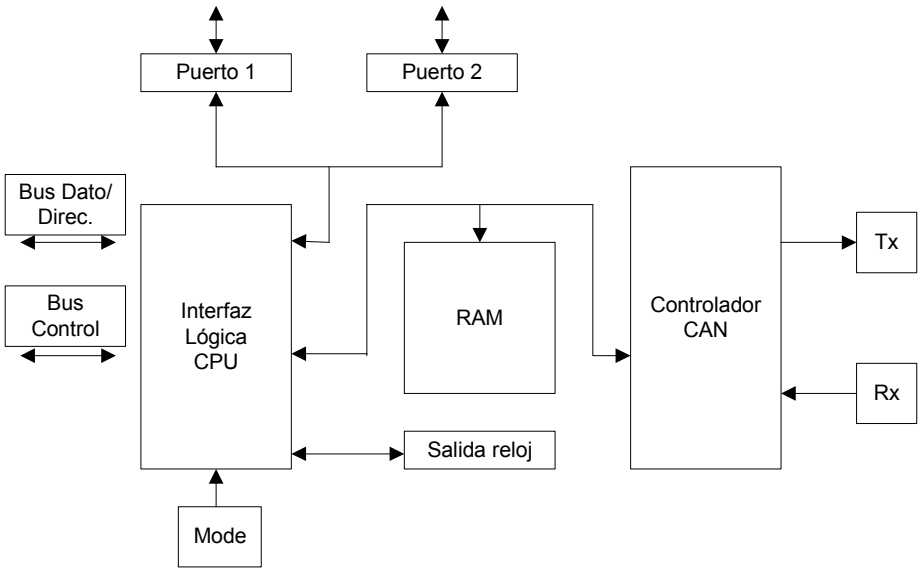


Figura. 4.12. Bloques funcionales del controlador Intel 82527

La memoria RAM tiene 256 bytes. El almacenamiento de mensajes se da en 15 objetos de mensajes o buffers que como ya se indicó tienen 8 bytes cada uno y además registros dedicados a funciones específicas.

Filtraje de Mensajes

El controlador Intel 82527 contiene un registro de máscara global mediante el cual se puede especificar los bits del mensaje recibido que deben coincidir con el identificador del mensaje configurado para ser aceptado dentro del buffer del mensaje asociado.

Se puede especificar la recepción ya sea de un solo mensaje o de un grupo de mensajes. Para la admisión de mensajes mediante el buffer adicional (#15) se dispone de un registro de máscara local. Los dos registros pueden contener 11 o 29 bits según el tipo de formato CAN escogido.

.

Cuando llega un mensaje, éste pasa primero por el filtro global, en caso de no ser aceptado pasa al filtro local y si no es aceptado por ninguno de los dos filtros, el mensaje es rechazado por el nodo.

En el proceso de filtraje de mensajes, los bits identificadores relevantes están marcados como “1” y los irrelevantes como “0”.

También se deben configurar los Registros de Arbitraje de cada buffer en los que se colocan las posiciones de los bits relevantes del mensaje.

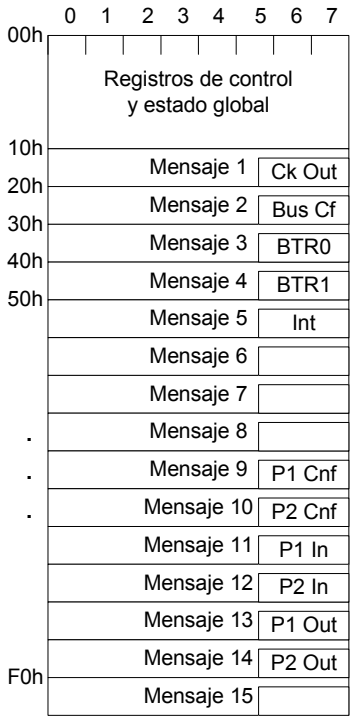


Figura. 4.13. Mapa de memoria del Controlador Intel 82527

Análogamente con el Philips SJA100, el registro de código de admisión realiza la misma función que los registros de arbitraje en el Intel 82527.

Control de transmisión

Una ventaja de este controlador es que se pueden reconfigurar los buffers de mensajes sin necesidad de que el controlador se ponga en un estado en el que deje de recibir mensajes. Los bits necesarios para controlar la transmisión de mensajes y el acceso a los buffers se encuentran ubicados de una manera especial en los registros descritos en cada buffer.

Para cada función (requerimiento de transmisión, estado del mensaje, etc) se proveen un par de bits en los registros de control como muestra la figura 4.14.

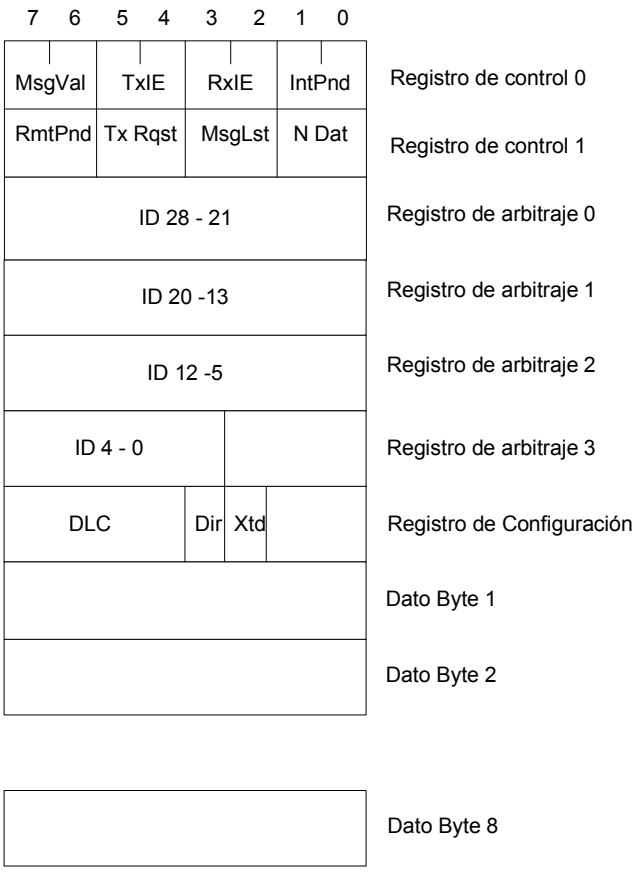


Figura. 4.14. Buffer de mensaje Intel 82527

En estos casos la combinación “10” significa activación y la combinación “01” significa desactivación.

Para la configuración del buffer de mensajes primeramente son desactivados “01” los bits MsgVal en el registro de control 0. Después de que el identificador del mensaje ha sido introducido en los **Registros de Arbitraje 0 a 3** se debe especificar la dirección del dato (buffer transmisor o receptor), la longitud del código de dato y el tipo de protocolo (estándar o extendido) en el **Registro de Configuración**.

En el **Registro de Control 0** se pueden habilitar las interrupciones de transmisión o recepción mediante las parejas de bits TxIE y RxIE. Para la transmisión de un mensaje se debe generar la interrupción de la transmisión completada y además la interrupción de la recepción de la trama remota correspondiente. Es decir que se pueden generar las interrupciones de transmisión y recepción al mismo en un buffer de mensajes.

Buffer Transmisor

En el buffer transmisor se debe activar el bit CPUupd mientras no se introduzca el dato a ser transmitido para informar al controlador CAN que el dato no es consistente todavía y no se puede enviar en respuesta a una trama remota. Una vez que la actualización del campo de datos ha sido finalizada, el bit CPUupd es desactivado y la transmisión es requerida mediante la activación del bit TxRqst.

Luego de una transmisión satisfactoria, el controlador resetea el bit TxRqst, genera una interrupción de transmisión (si ésta fue habilitada) y activa el bit IntPnd, el cual debe ser desactivado por el Controlador Host al reconocer la interrupción. Si el mensaje fue requerido por una trama remota y no pudo ser transmitido inmediatamente, se activa temporalmente el bit RmtPnd y se desactiva nuevamente tan pronto como la transmisión haya finalizado.

Buffer Receptor

En el buffer receptor, el bit NewDat indica si un nuevo mensaje ha sido recibido o no. Antes de leer el dato, este bit debe ser desactivado por el Controlador Host. En caso de que el bit NewData sea activado nuevamente después de la lectura de un mensaje significa que un nuevo mensaje fue recibido durante la lectura, en este caso el dato leído puede resultar una mezcla entre el dato anterior y el nuevo por lo cual debe repetirse la lectura. Este proceso es indicado mediante el bit MsgLst o Message Lost (pérdida de mensaje). La transmisión de una trama remota es un proceso muy común en el principio FullCAN y es señalada mediante la activación del bit Transmit Request (Requerimiento de transmisión) en el **Registro de Control 1** del buffer receptor.

La identificación más rápida de eventos se da mediante las interrupciones en el **Registro de Interrupción**, en el cual se indica mediante un índice desde que buffer fue transmitido un mensaje o mediante que buffer fue recibido un nuevo mensaje. Los 14 buffers FullCAN están definidos mediante los índices 3 a 16, el buffer 15 BasicCAN se define con el índice 2 y el **Registro de Estado** se define con el índice 1. Cuando se

encuentra un índice cero en el registro de interrupciones quiere decir que no existe ninguna interrupción pendiente.

Interrupt	Register Value (hex)
none	0
Status Register	1
message object 15	2
message object 1	3
message object 2	4
message object 3	5
message object 4	6
message object 5	7
message object 6	8
message object 7	9
message object 8	AH
message object 9	BH
message object 10	CH
message object 11	DH
message object 12	EH
message object 13	FH
message object 14	10H

Figura. 4.15. Índices de buffers en los que se producen interrupciones

Otros Controladores Stand Alone

A continuación se detallan otras marcas disponibles en el mercado de controladores y microcontroladores CAN en las que se nombran sus características principales ya que su funcionamiento, ya sea en FullCAN o BasicCAN, se basa en el funcionamiento de los controladores detallados anteriormente es decir el Philips SJA1000 e INTEL 82527.

Las características de arbitraje, filtraje de mensajes, recepción y transmisión de mensajes mediante buffers, reconocimientos, errores, sincronización, tiempos de bit, etc funcionan de la misma forma y generalmente solo varia el mapa de memoria es decir las localizaciones de los registros principales tanto en los siguientes controladores como en los microcontroladores que se presentarán con sus características principales en la sección 4.4.

4.3.3 Texas Instrument TI-CAN

- Este controlador soporta aplicaciones CAN de formato tanto estándar como extendido.

- Capacidad para 16 buffers de mensajes. Cada uno con su respectivo identificador y filtro de máscara.

- El controlador TI-CAN se puede dividir en dos capas: la capa de hardware denominada módulo CAN-HP, la cual provee la implementación de bajo nivel del protocolo CAN y la capa de software que controla las cadenas de datos y provee la implementación del protocolo extendido.

- La capa de hardware (módulo HP) provee características como generación de reloj transmisor y receptor, tiempos de bit y sincronización, detección de inicio de trama, relleno de bit, codificación y decodificación, detección de error y arbitraje.

- La capa de software se desarrolla mediante el RPP (Procesador de protocolo RISC) con acceso a la capa de software mediante los registros CAN-HP.

- Incluye un módulo de interrupciones en tiempo real.

- 256 bytes de memoria RAM de datos.

- 256 bytes de memoria RAM de comunicación.

- 2 Kbytes de memoria de programa ROM.

4.3.4 Microchip MCP2510

- Provee una interfaz SPI o Serial Peripheral Interface. El acceso a los buffers de mensajes y a los otros registros del controlador CAN solo es posible mediante esta interfaz.

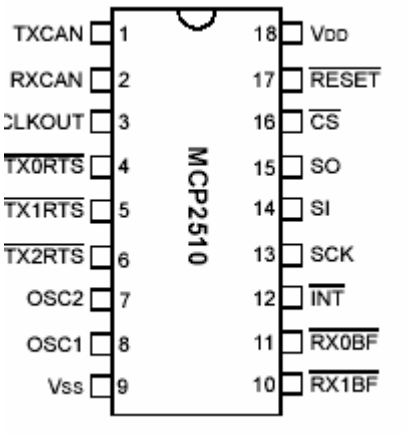


Figura. 4.16. Controlador CAN Microchip MCP2510

- Soporta implementaciones FullCAN de formatos estándar y extendidos.

- Contiene dos buffers receptores y tres transmisores.

- El controlador tiene tres pines especiales para la transmisión muy rápida de mensajes y son los pines “request to send” o requerimiento de envío.

- Los dos buffers receptores proveen diversos filtros de mensajes, el buffer de recepción 0 tiene dos registros de código de admisión y un registros de máscara de admisión; el buffer receptor 1, tiene cuatro registros de código de admisión y uno de máscara de admisión.

- Si un mensaje recibido pasa un filtro de mensajes, es transferido dentro del correspondiente buffer de mensajes.

- Además este controlador tiene un modo de operación “loopback” en el cual los mensajes transmitidos no son enviados por el bus sino solamente entre sus propios buffers receptores.

4.3.5 Infineon 82C900

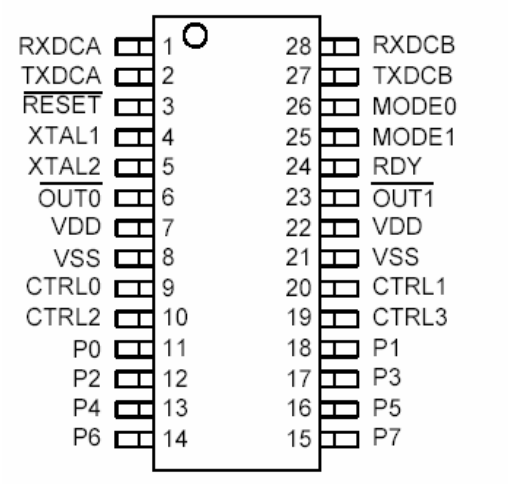


Figura. 4.17. Controlador CAN Infineon 82C900

- Este controlador pertenece a una nueva generación de controladores CAN, ya que contiene dos controladores FullCAN incorporados que trabajan de acuerdo a CAN 2.0B (formatos estándar y extendidos), los dos controladores pueden operar independientemente o pueden ser utilizados como puentes para la transferencia de tramas remotas o de datos entre dos redes CAN.

- Se pueden asignar hasta 32 mensajes a uno de los dos controladores CAN.

- Los buffers de mensajes pueden estar concatenados como una estructura FIFO con 2,4,8,16 o 32 mensajes.

- Transferencia automática de mensaje entre los dos buses CAN sin la intervención del Controlador Host.

- Cada buffer de mensajes contiene su propio registro de máscara de admisión.

- Interfaz SPI.

- Generación de interrupciones.

- Distinción e indicación de errores de bus.

- Modo “Listen only” para monitoreo de bus.

4.4 MICROCONTROLADORES CON CONTROLADORES CAN INCLUIDOS

Los microcontroladores CAN además de proveer características de almacenamiento y manipulación de mensajes CAN mediante un módulo controlador CAN, poseen la inteligencia para desarrollar funciones de control local y procesamiento de datos con diferentes aplicaciones como contadores, temporizadores, conversores A/D o D/A, modulación de señales o pulsos (PWM), uno o varios puertos de entrada/salida, entradas analógicas y digitales, etc. El funcionamiento de los módulos controladores CAN dentro del microcontrolador es similar a los controladores CAN “Stand Alone” pero siempre con pequeñas diferencias como número de buffers, tipo de filtraje de mensajes, formatos del protocolo, etc.

Generalmente estos microcontroladores cuentan con nuevos registros para labores de direccionamiento, acceso y envío de datos que actúan como enlace entre el módulo controlador CAN y el CPU del microcontrolador.

A continuación se nombran algunos microcontroladores CAN de diferentes fabricantes con sus características fundamentales.

4.4.1 Philips P8xC592

- Fue el primer microcontrolador con un módulo CAN incluido. Derivado de la familia 8051 de microcontroladores de 8 bits.

- Su módulo CAN trabaja de acuerdo al principio BasicCAN y su funcionalidad es idéntica al controlador CAN Stand Alone Philips SJA1000.

- Trabaja bajo la especificación CAN 2.0A (formato estándar).

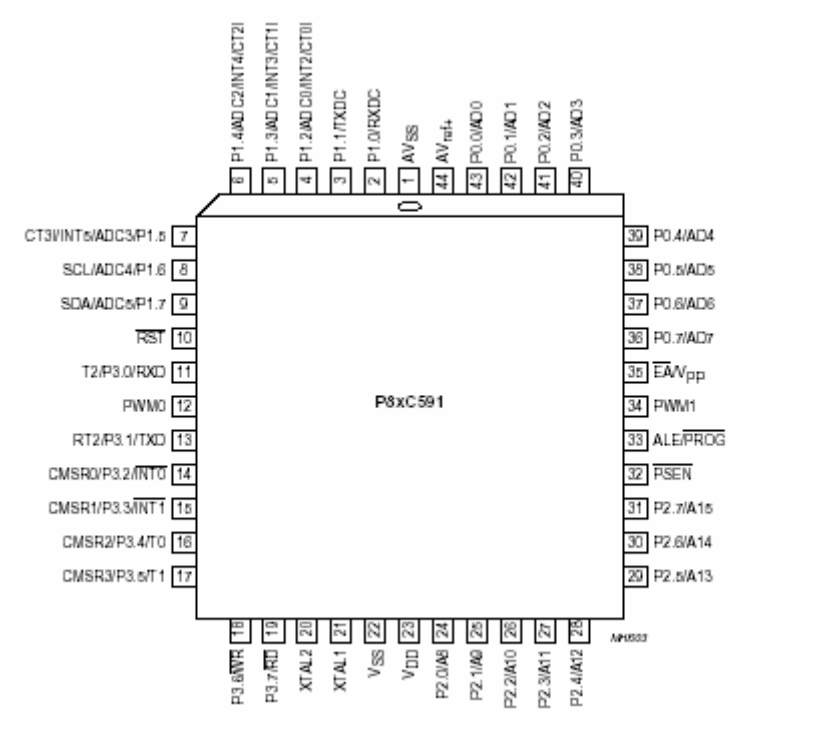


Figura. 4.18. Microcontrolador CAN Philips P8xC592

- Los otros dos registros habilitan la lectura directa del Registro de interrupción y de Estado del controlador CAN así como la escritura en el Registro de Comando de dicho controlador (figura 4.19.).

- El CPU del microcontrolador puede acceder al módulo controlador CAN mediante cuatro registros de funciones especiales. Dos de estos registros sirven para la transferencia de datos. En el primero, CANADR, se ingresa la dirección del registro del controlador CAN a ser leído o escrito. El otro, CANDAT, provee el dato asociado a la transferencia.

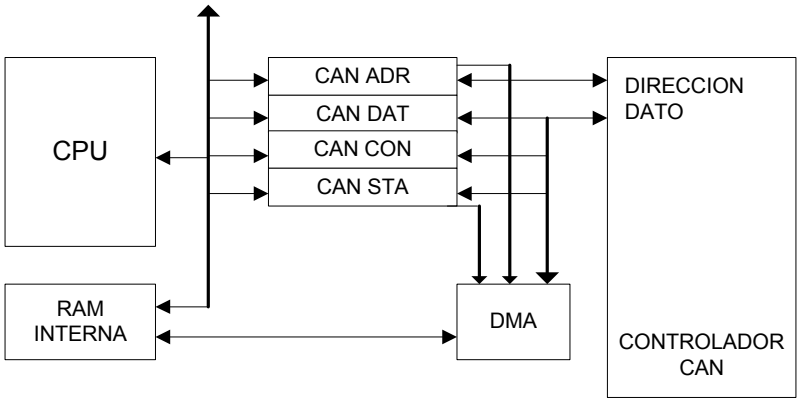


Figura. 4.19. Interfaz del controlador CAN Philips 80C592

- Se proveen dos mecanismos con respecto al tiempo necesario para copiar un mensaje recibido desde el buffer receptor a la RAM del microcontrolador:

  1) *Modo "Incremento autodireccionado"*: en este modo, la dirección inicial del buffer se provee una sola vez en el registro de direcciones, posteriormente se puede llevar a cabo 10 accesos de lectura consecutivos en el registro de datos. Después de cada lectura del registro de datos, el registro de direcciones se incrementa automáticamente.

  2) Acceso Directo a Memoria (DMA): en este modo existe un canal entre el buffer de recepción o transmisión con la memoria RAM del microcontrolador. Esta operación toma pocos ciclos de instrucción. Es posible transferir hasta 10 bytes entre el controlador CAN y la memoria RAM del microcontrolador dentro de dos ciclos de instrucción.

- Tiempo de ciclo por instrucción de 1 us.

- 256 Bytes de RAM interna.

- 64 Kbytes de espacio de direcciones tanto para código como para datos.

- Cuatro bancos de registros con 8 registros cada uno.
- Dos timers de 16 bits.
- Un timer de 16 bits con funciones de comparación y captura.
- Dos salidas PWM con 8 bits de resolución.
- 15 Interrupciones con 2 niveles de prioridad.
- 5 puertos de entrada/salida de 8 bits.
- Convertidor análogo/digital de 10 bits.
- Comunicación UART full duplex.
- Watchdog Timer.

4.4.2 Infineon C515C/C505C

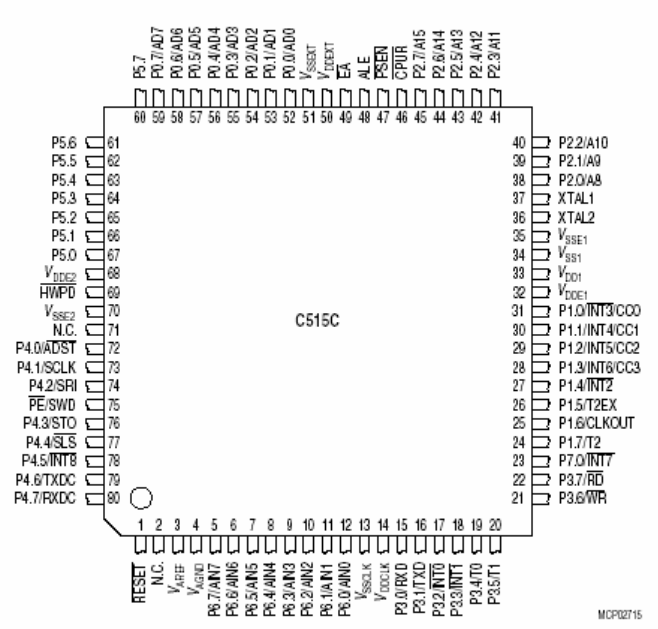


Figura. 4.20. Microcontrolador CAN Infineon C515C

- Con respecto al funcionamiento del módulo CAN interno de este microcontrolador corresponde al 82527 de Intel (sección 4.3.2) y es direccionado mediante una memoria externa RAM.

- Controlador FullCAN con 15 buffers de mensajes.

- CPU de 8 bits.

- Tiempo de ciclo de instrucción de 600 ns.

- 64 Kbytes de ROM interna.

- 256 bytes de RAM interna.

- 15 interrupciones con 4 niveles de prioridad.

- 3 temporizadores/contadores de 16 bits.

- Módulo PWM.

- 4 canales de comparación y captura.

- Convertidor A/D de 10 bits.

- 49 pines de entrada/salida.

- WatchdogTimer programable.

- Comunicación UART FullDuplex.

4.4.3 Dallas DS80C390

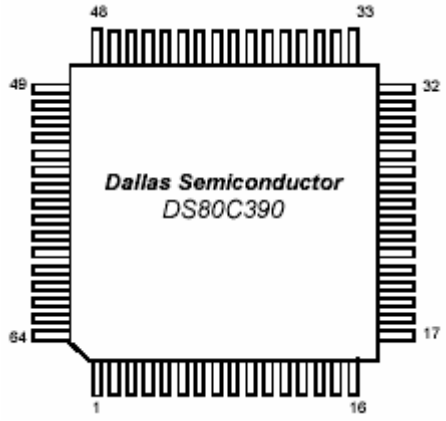


Figura. 4.21. Microcontrolador CAN Dallas DS80C390

- Contiene dos módulos controladores CAN implementados de acuerdo a la especificación 2.0B.

- 15 buffers de mensajes de los cuales 14 pueden ser programados como transmisores o receptores (FullCAN) y un buffer que puede ser programado solo como receptor (BasicCAN).

- Mecanismos de filtraje de mensajes para 11 y para 29 bits identificadores.

- Tiempo de ciclo de instrucción de 100 ns a un reloj de 40 MHz.

- 4 Kbytes de memoria SRAM interna.

- Espacio de direccionamiento para 4 Kbytes de memoria RAM externa y 4 Mbytes de memoria ROM externa.

- 16 interrupciones.

4.4.4 Mitsubishi M37630M4

- Microcontrolador de 8 bits con módulo CAN incluido de acuerdo al principio BasicCAN, con un buffer transmisor y un buffer receptor.

- Funcionamiento de acuerdo a la especificación 2.0B tanto para trama estándar como extendida.

- El filtro de admisión de mensajes comprende todos los bits identificadores del mensaje.

- 16 Kbytes de memoria interna ROM.

- 512 bytes de memoria interna RAM.

- 36 pines de entrada/salida.

- Tres temporizadores de 8 bits.

- Dos temporizadores de 16 bits con funciones PWM.

- Conversor A/D de 8 bits.

- Comunicación UART Full Duplex.

- Interface Serial Sincrona.

- Watchdog Timer.

- Modo sleep.

4.4.5 National Semiconductor COP684BC/COP884BC

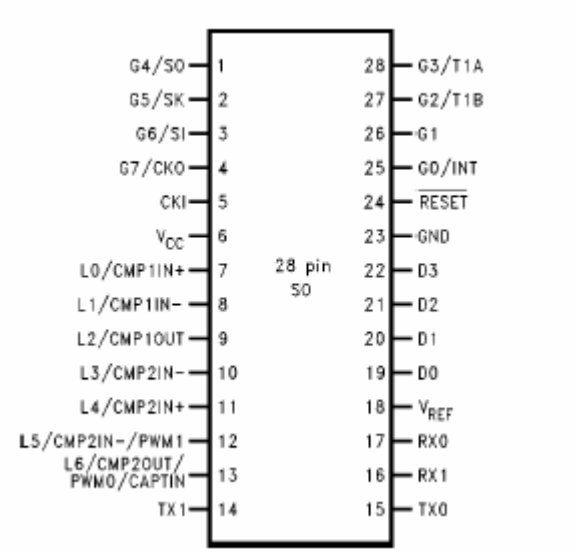


Figura. 4.22. Microcontrolador CAN Nacional Semiconductor COPP648BC

- Microcontrolador de 8 bits con módulo BasicCAN.
- El módulo CAN soporta las especificaciones 2.0A y 2.0B es decir para formatos de trama estándar y extendidos.
- Trabaja con aplicaciones que requieren bajas velocidades en el módulo CAN.
- Contiene un buffer transmisor y un buffer receptor (BasicCAN).
- El dispositivo es capaz de generar una interrupción tan pronto como un byte haya sido transmitido o recibido.
- Tiempo de ciclo por instrucción de 1 us a un reloj de 10 MHz.
- 2048 bytes de memoria ROM interna.
- 64 bytes de memoria RAM interna.

- 2 comparadores.

- Modulo PWM de 8 bits.

- Un temporizador de 16 bits con módulos de comparación y captura incluidos.

4.4.6 Motorola MC68HC05Xx

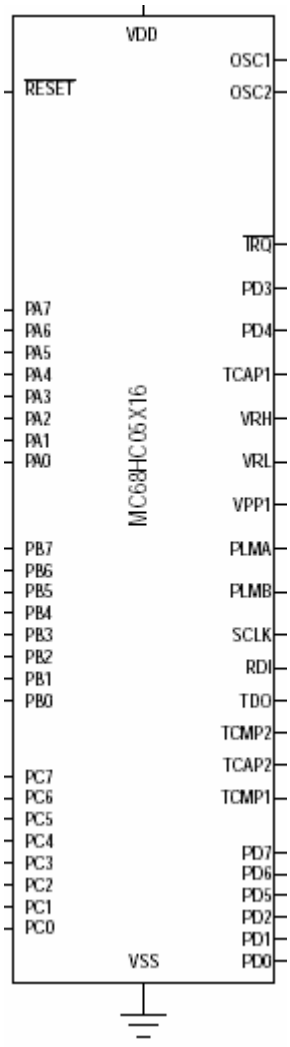


Figura. 4.23. Microcontrolador CAN Motorola MC68HC05X16

- El modulo CAN integrado en ésta familia de microcontroladores trabaja de acuerdo al principio BasicCAN y soporta solamente la versión 2.0A del protocolo CAN es decir con formato estándar de 11 bits identificadores.

- El módulo consta básicamente de un buffer transmisor, un buffer receptor y los registros de estado y control.

Motorola MC68C05X4

- 8 Kbytes de espacio de direccionamiento.

- 176 bytes de memoria interna RAM.

- 4 Kbytes de memoria interna ROM.

- Un timer con funciones de comparación y captura.

- Watchdog Timer.

- Dos puertos de entrada/salida de 8 bits.

Motorola MC68HC05X16

Además de las características del MC68C05X4 contiene las siguientes mejoras:

- 16 Kbytes de espacio de direccionamiento.

- 352 bytes de memoria interna RAM.

- 15 Kbytes de memoria interna ROM.

- 256 bytes de memoria interna EEPROM.

- Tres puertos de entrada/salida de 8 bits.

- Un puerto de entrada de 8 bits digital o análogo.

- Un conversor A/D con resolución de 8 bits.

- Comunicación UART Full duplex

Motorola MC68HC05X32

Además de las características del MC68C05X16 contiene las siguientes ampliaciones:

- 32 Kbytes de espacio de direccionamiento.

- 528 bytes de memoria interna RAM.

- 31 Kbytes de memoria interna ROM o EEPROM.

- Cuatro puertos de entrada/salida de 8 bits para voltajes de 5 a 20 Voltios.

4.4.7 Motorola MC68HC08AZx

- La familia HC08AZx resulta mucho más poderosa que la HC05 en un factor de cinco lo cual permite que se desarrollen aplicaciones fácilmente.

- Además, el set de instrucciones es extendido, lo cual permite que el lenguaje de programación sea más eficiente.

- Las características adicionales en el módulo CAN que provee este microcontrolador con respecto a los microcontroladores Motorola anteriormente nombrados se da en la posibilidad de intercambiar mensajes entre los formatos estándar y extendido.

- En la parte receptora se provee un solo buffer de mensajes pero se incluye un proceso de filtraje extendido. Pueden ser seleccionados tres tipos de filtros:

  1) Filtro de 32 bits en el que se incluyen los 29 bits identificadores de los mensajes de formato extendido.

  2) Dos filtros independientes de 16 bits cada uno que contienen los 11 identificadores del formato estándar o los 14 identificadores más significativos en el formato extendido.

  3) Cuatro filtros identificadores de 8 bits los cuales comprenden los 8 bits más significativos del formato estándar o extendido.

- Permite una conexión entre el módulo temporizador y el módulo CAN que permite conocer el tiempo exacto en el que llegan los mensajes mediante un registro de captura.

- Ciclo de tiempo por instrucción de 125 ns a 16 MHz.

- 64 Kbytes de espacio de direccionamiento.

- 1 Kbyte de memoria interna RAM.

- 512 bytes de memoria interna ROM.

- Bus de datos/direcciones externo.

- Dos temporizadores de 16 bits con módulos de comparación o captura y funciones de modulación PWM en un total de 6 canales.

- Un conversor de A/D de 8 bits de resolución.

- 48 pines de entrada/salida.
- Seis interrupciones externas.
- Comunicación UART Full duplex.
- Comunicación Serial (SPI – Serial Peripheral Interface)

Existen otros microcontroladores miembros de esta familia que difieren en las siguientes características:

- 68HC08AZ32: 32 Kbytes de ROM, 1 Kbyte de RAM.
- 68HC08AZ24: 24 Kbytes de ROM, 768 bytes de RAM.
- 68HC08AZ32: 16 Kbytes de ROM, 512 bytes de RAM.
- 68HC08AT32: 32 Kbytes de Flash, 1,2 Kbyte de RAM

4.4.8 Fujitsu MB90F5xx

- Microcontrolador de 16 bits con controlador CAN integrado, una implementación FullCAN versión 2.0B con 16 buffers de mensajes que pueden ser configurados como transmisores o receptores.
- Contiene dos filtros de máscara con 32 bits cada uno que pueden ser asignados a buffers receptores individuales.
- Ciclo de instrucción de 62,5 ns a un reloj de 16 MHz.
- 128 Kbytes de ROM o Flash dependiendo de la versión.

- 4 Kbytes de RAM interna.
- 2 módulos de comunicación UART Full Duplex.
- Interfaz serial síncrona (SPI).
- Conversor A/D de 8 o 10 bits de resolución según la versión.
- Unidad de captura de 8 canales.
- Unidad de comparación de 4 canales.
- Generador de pulsos programable de 4 canales.
- 8 Interrupciones externas.

4.4.9 Fujitsu MB90F594

Este microcontrolador contiene un módulo CAN de las mismas características que toda la familia como se indicó en la sección 4.4.8, pero con las siguientes diferencias:

- 256 Kbytes de ROM o Flash.
- 6 Kbytes de RAM interna.
- 3 módulos de comunicación UART Full Duplex.
- Unidad de captura de 6 canales.
- Unidad de comparación de 6 canales.
- Generador de pulsos programable de 6 canales.

- Temporizador de 16 bits.

- Watchdog Timer.

- Controlador de motor de pasos de 4 canales.

4.4.10 Hitachi SH7055

- Microcontrolador de 32 bits que contiene dos controladores CAN llamados módulos HCAN.

- El módulo HCAN trabaja de acuerdo a la versión 2.0 B conteniendo 16 buffers de mensajes.

- 15 buffers de mensajes pueden ser configurados como transmisores o receptores de acuerdo al principio FullCAN. Estos buffers no contienen ningún tipo de filtro de admisión ya que solo aceptan los mensajes con identificadores que coincidan con todos los identificadores configurados.

- Un buffer de mensajes es solo para recepción de acuerdo al principio BasicCAN y contiene un filtro de máscara de admisión lo cual habilita la recepción de grupos de mensajes.

- El procesamiento de los mensajes a ser transmitidos pueden darse de acuerdo a dos estrategias:

    1) Los buffers transmisores son procesados de acuerdo a su posición en la memoria del módulo.

    2) Según la prioridad del mensaje. En este caso el HCAN debe asegurar que al comienzo de una fase de arbitraje, siempre se transmita primero el mensaje de mayor prioridad.

- Ciclo de instrucción de 25 ns 40 MHz.
- 4 Gbytes de espacio de direcciones.
- Bus de datos externo de 8/16 bits.
- 512 Kbytes de FLASH.
- 32 Kbytes de SRAM.
- 9 interrupciones externas.
- 115 interrupciones internas con 16 niveles de prioridad.
- 4 canales DMA.
- Unidad de comparación y captura.
- Generador PWM.
- Contador de eventos.
- Generador de pulsos.
- Watchdog Timer.
- Comunicación UART full duplex de 5 canales.
- Convertidor A/D de 12 bits con 32 canales.
- 149 pines de entrada/salida.

4.4.11 ST Microelectrónics 72F561

- Microcontrolador de 8 bits que incluye un módulo BasicCAN llamado beCAN que soporta las versiones 2.0 A y 2.0 B del protocolo.

- Proporciona diferentes mecanismos de filtraje de mensaje para reducir la carga en el CPU:

  1) Un filtro de 32 bits para filtrar mensajes en formato extendido incluyendo el bit RTR.

  2) Dos filtros de 16 bits aplicados a los identificadores de formato estándar y el bit RTR.

  3) Cuatro filtros de 8 bits aplicados al byte más significativo del formato estándar.

  4) Una combinación de un filtro de 16 bits y dos filtros de 8 bits

  Las características principales del módulo CAN son las siguientes:

- Dos buffers transmisores con prioridad configurable.

- Un buffer receptor con estructura FIFO de tres estados.

- Captura de tiempo en la transmisión o recepción del bit de inicio SOF

En general el microcontrolador proporciona las siguientes características:

- Tiempo de ciclo de instrucción de 125 ns a 8 MHz.

- De 8 a 64 Kbytes de memoria de programa FLASH.

- 2Kbytes de RAM.
- 14 Interrupciones.
- Temporizador de 16 bits (PWM y generador de pulsos).
- Temporizador de 8 bits (Watchdog Timer y generador de pulsos).

4.4.12 Philips XA C3

- Microcontrolador de 16 bits con módulo CAN integrado.
- El módulo CAN trabaja bajo el principio FullCAN en la versión 2.0B es decir para formato estándar y extendido.
- Contiene 32 buffers de mensajes y 32 filtros de código de admisión y máscara de admisión de 32 bits.
- Compatible con el Philips SJA1000.
- Frecuencia de reloj de 2 MHz.
- 39 interrupciones.
- 32 Kbytes de memoria interna EPROM.
- 1024 bytes de memoria interna RAM.
- Tres temporizadores/contadores.
- Cuatro puertos de 8 bits.
- WatchdogTimer.

- Comunicación UART e Interfaz SPI.

4.4.13 Atmel T89C51CC01/02/03

Los microcontroladores Atmel T89C51CC01/02/03 incluyen un controlador CAN de acuerdo al principio FullCAN y soportan las versiones 2.0A y 2.0 B del protocolo, es decir en formatos estándar (11 bits identificadores) y extendido (29 bits identificadores).

Con respecto al controlador CAN, los microcontroladores T89C51CC01 y T89C51CC03 tienen 15 buffers de mensajes mientras que el T89C51CC02 incluye únicamente 4 buffers de mensajes o también llamados objetos de mensajes.

Una característica especial de los buffers de mensajes es su forma de programar ya que cada buffer puede ser configurado ya sea como transmisor, receptor o como parte un conjunto de buffers receptores que conforman una estructura FIFO, esto se muestra en la figura 4.25.

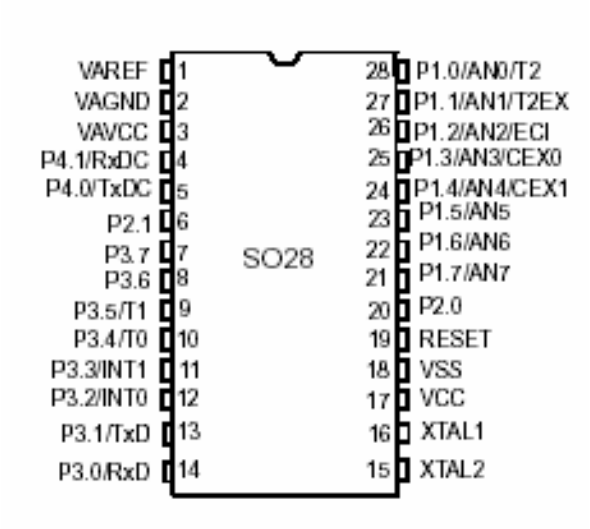


Figura. 4.24. Microcontrolador CAN Atmel T89C51CC51

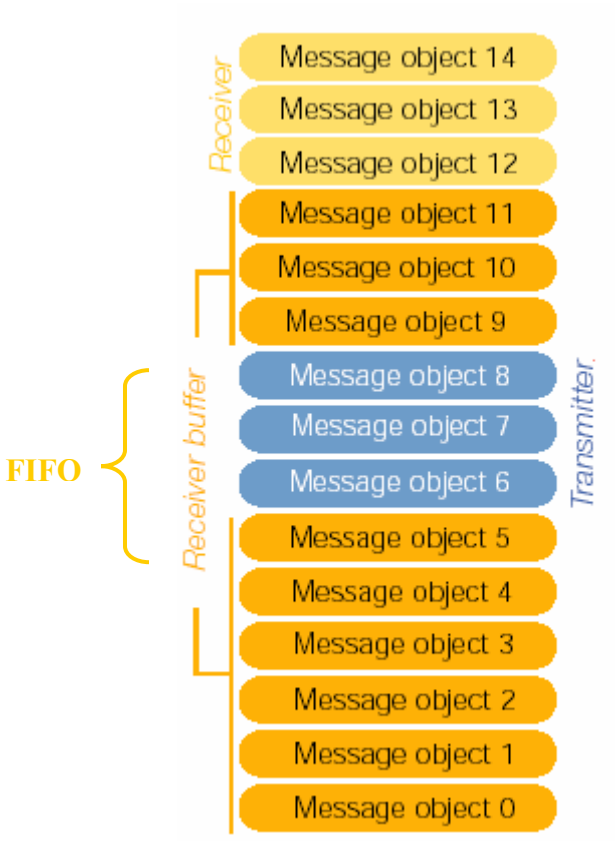


Figura. 4.25. Buffers (objetos) de mensajes ATMEL T89C51CC02/03

Cada buffer tiene sus propios registros de filtros y máscaras de admisión para mensajes ya sea de 11 o de 29 bits identificadores para reducir la carga del Procesador Host. Además cada buffer contiene 8 registros de 1 byte cada uno para el almacenamiento del campo de datos de los mensajes así como registros de 16 bits de control y estado general de cada buffer.

En la tabla 4.2. se pueden observar las características principales y diferencias entre los 3 microcontroladores ATMEL correspondientes a esta familia

	T89C51CC02	T89C51CC01	T89C51CC03
Memoria Flash	16 Kb	32Kb	64Kb
Memoria EEPROM	2Kb	2 Kb	2 Kb
Memoria RAM	0.5 Kb	1.2 Kb	2.2 Kb
Controlador CAN	SI	SI	SI
Objetos de mensajes CAN	4	15	15
UART	SI	SI	SI
Temporizadores	3 de 16 bits	3 de 16 bits	3 de 16 bits
Canales PCA	2	5	5
Conversor ADC	8 can. De 10 bits	8 can. de 10 bits	8 can. de 10 bits
Watchdog Timer	21 bits	21 bits	21 bits
Frecuencia de operación	40MHz	40MHz	40MHz
Pines de E/S	20	34	34
Voltaje de alimentación	3 – 5.5 V	3 – 5.5 V	3 – 5.5 V
Temperatura	-40 a 85 °C	-40 a 85 °C	-40 a 85 °C

Tabla. 4.2. Características de procesador de los microcontroladores ATMEL T89C51CC01/02/03

Adicionalmente el controlador CAN de los microcontroladores ATMEL proporcionan un temporizador de captura CAN o CANTIMER de 16 bits, el cual es usado para capturar el momento preciso en el que un mensaje fue enviado o recibido. Este contador inicia su labor tan pronto como el bit ENA del registro de control CANGCON es habilitado.

Los microcontroladores ATMEL adicionalmente pueden generar hasta 14 interrupciones que son controladas mediante registros específicos.

Mediante la configuración de los bits que conforman los diferentes registros relacionados al módulo CAN se pueden realizar las diferentes funciones tales como configuración de buffers de mensajes CAN, transmisión y recepción de mensajes, filtros y máscaras de admisión de mensajes, tasa de datos, sincronización, tiempos de bit, detección de errores, reconocimientos, interrupciones, etc. Los mapas de memorias, SFRs o registros de funciones especiales y tablas de funcionamiento de los diferentes bits se pueden encontrar en las hojas de especificaciones (datasheet) de los microcontroladores ATMEL de esta familia.

4.4.14 Microchip PIC18F6585/8585/6680/8680

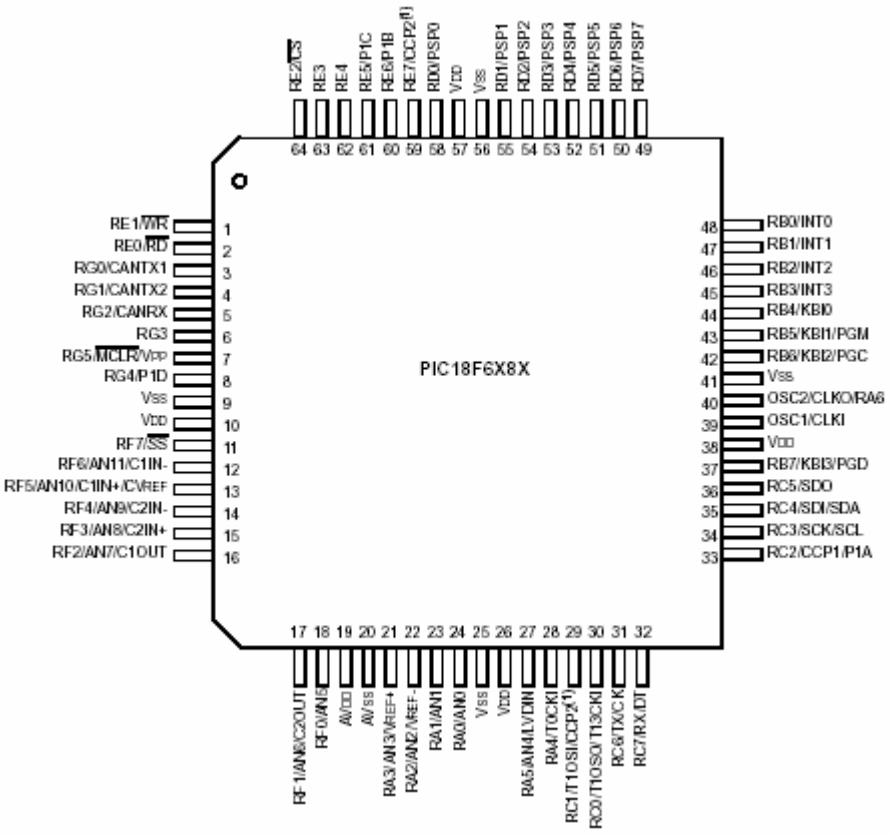


Figura. 4.26. Microcontrolador CAN Microchip 18F6X8X

Los microcontroladores Microchip PIC18F6585/8585/6680/8680 incluyen un módulo denominado ECAN y son muy utilizados para aplicaciones CAN por lo que se profundizará un poco más en relación a los microcontroladores nombrados anteriormente.

Las siguientes características son comunes a los 4 microcontroladores Microchip con módulo ECAN:

- Frecuencia de operación: 40 MHz.
- Memoria EEPROM: 1024 bytes.
- Memoria de datos: 3328 bytes.
- Numero de interrupciones: 29
- Temporizadores/Contadores: 4
- Módulos PWM, comparación y captura: 4
- Set de 75 instrucciones 75.

Existen algunas diferencias dentro de la familia de dichos microcontroladores:

- Los 18F6x8x tienen 64 o 68 pines y los 18F8x8x tienen 80 pines.
- El 18Fx585 tienen 48 kbytes de memoria FLASH y los 18F8x8x tienen 64 Kbytes.
- Los 18F6x8x tienen un conversor A/D de 12 canales y los 18F8x8x tienen un conversor A/D de 16 canales.
- Los 18F6x8x tienen 7 puertos de entrada/salida y los 18F8x8x tienen 9 puertos.

4.4.14.1 Modulo ECAN

Figura. 4.27. Módulo ECAN de Microchip

Las características principales de los módulos ECAN son comunes a todos los microcontroladores de esta familia:

- Tasa de transmisión de mensajes hasta de 1Mbps.
- Trabaja de acuerdo a la especificación 2.0B del protocolo CAN con formatos estándar y extendidos.
- Tres buffers dedicados para transmisión con prioridad.
- Dos buffers dedicados para recepción.
- Seis buffers programables para transmisión o recepción.
- Tres filtros de admisión de 29 bits.
- Tres modos de operación

Modos de operación ECAN

Estos microcontroladores pueden trabajar bajo 6 modos de operación

1) Configuration Mode (Modo de Configuración)

El módulo CAN tiene que ser inicializado antes de su activación poniéndolo en modo de configuración mediante un nivel alto en el bit OPMODE2. Un requerimiento de microcontrolador de modo de configuración se indica mediante el bit REQOP2. Nunca se puede aceptar un requerimiento de configuración durante la transmisión de un mensaje.

La función principal del Modo Configuración es la de programar algunos registros:

- Registros de configuración.
- Registros de selección de modos funcionales.
- Registros de tiempo de bit.
- Registros de filtro de admisión y máscara de admisión.
- Registros de control de filtro y máscaras.
- Registros de selección de filtro y máscara.

2) Disabled Mode (Modo Desactivado)

En el modo desactivado, no se puede realizar ninguna transmisión o recepción de mensajes, ninguna interrupción queda habilitada y los contadores de errores mantienen sus valores. Si los bits del registro REQOP<2-0> se encuentran en el valor 001, el módulo ingresa en el modo desactivado después de esperar que el bus se encuentre libre mediante la detección de 11 bits recesivos consecutivos.

La interrupción señalada mediante el bit WAKIF, que indica si hay o no actividad en el bus CAN, del registro PIR 3 es la única que continúa habilitada en este modo y la que

permite que el módulo regrese al modo de funcionamiento anterior, es decir, al modo desactivado.

3) Normal Mode (Modo Normal)

Este es el modo más común de operación de todos los microcontroladores de esta familia, en el cual constantemente monitorean el bus, generan reconocimientos, detectan errores, etc. Solo en este modo un nodo está en capacidad de enviar un mensaje.

4) Listen Only Mode (Modo solo Receptor)

En este modo, el módulo sigue recibiendo todos los mensajes incluyendo los que contienen errores. Generalmente es utilizado para aplicaciones de monitoreo y detección de tasa de datos. En este modo no se puede realizar ninguna transmisión, incluyendo señales de reconocimiento y banderas de error.

Este modo es activado mediante los bits de requerimiento de modo en el registro CANCON.

5) Modo Loopback

En este modo se permite la transmisión interna de mensajes desde los buffers transmisores hacia los receptores sin la intervención de mensajes a través del bus CAN.

Este modo es utilizado generalmente para actividades de prueba y no se puede enviar ningún tipo de mensaje hacia otros nodos.

6) Error Recognition Mode (Modo de Reconocimiento de Error)

El módulo puede ser configurado para ignorar todos los errores y recibir cualquier tipo de mensajes. Este modo se activa mediante la configuración de los bits del registro RXM.

Modos funcionales del módulo ECAN

Existen tres modos funcionales del módulo ECAN adicionales a los modos de operación:

1) Modo 0 (Legacy Mode)

Los recursos disponibles en el modo 0 son los siguientes:

- Tres buffers transmisores TXB0, TXB1 y TXB2.
- Dos buffers receptores RXB0 y RXB1.
- Dos máscaras de admisión, una para cada buffer receptor RXM0 y RXM1.
- Seis filtros de admisión, dos para RXB0 y cuatro para RXB1, estos son: RXF0, RXF1, RXF2, RXF3, RFX4, RXF5

2) Modo 1 (Enhanced Legacy Mode)

Los recursos disponibles en el modo 1 son los siguientes:

- Tres buffers transmisores TXB0, TXB1 y TXB2.
- Dos buffers receptores RXB0 y RXB1.
- Seis buffers programables ya sea para transmisión o recepción.
- 16 filtros de admisión RXF0 – RXF15.
- Dos máscaras de admisión, RXM0 y RXM1.

- Seis filtros de admisión, dos para RXB0 y cuatro para RXB1, estos son: RXF0, RXF1, RXF2, RXF3, RFX4, RXF5.

3) Modo 2 (Enhanced FIFO Mode)

En este modo dos o más buffers receptores pueden ser utilizados para conformar un buffer receptor FIFO (First In First Out), es decir que no existe una relación entre un buffer receptor y un registro de filtro de admisión, sino que cualquier filtro que sea habilitado y enlazado al buffer receptor FIFO genera una admisión del mensaje y conforma parte de la estructura FIFO.

La longitud de la estructura FIFO siempre consta de RXB0, RXB1 y los otros buffers (B0-B5) programables dependiendo del primer buffer programado como transmisor. Si se programa B4 como transmisor la longitud de la estructura FIFO será de 6 (RXB0,RXB1,B0,B1,B2,B3) es decir que la longitud puede variar entre 2 y 8 buffers receptores.

Los recursos disponibles en el modo 2 son los siguientes:

- Tres buffers transmisores TXB0, TXB1 y TXB2.
- Dos buffers receptores RXB0 y RXB1.
- Seis buffers programables como transmisores o receptores (B0 – B5).
- 16 filtros de admisión RXF0-RXF15.
- Dos máscaras de admisión, RXM0 y RXM1.

4.4.14.2 Buffers de mensajes ECAN

Existen tres tipos de mensajes CAN: los buffers dedicados a transmisión, los buffers dedicados a recepción y los buffers programables como transmisión o recepción. Todos ellos constan de registros de control, de longitud de datos, de identificación y de datos. Además todos ocupan un espacio de memoria RAM de 14 bytes.

4.4.14.5 Registros del módulo ECAN

Existen una gran cantidad de registros mapeaedos en la memoria del microcontrolador relacionados al módulo ECAN, cada uno de ellos contiene 8 bits de los cuales algunos pueden ser reservados o simplemente no utilizados. Mediante la configuración, ya sea automática o por programación de los bits pertenecientes a los diferentes registros, se pueden realizar las diferentes actividades concernientes al protocolo CAN tales como asignación de modos de operación, modos funcionales, buffers de mensajes CAN, transmisión y recepción de mensajes, filtros y máscaras de admisión de mensajes, tasa de datos, sincronización, tiempos de bit, detección de errores, reconocimientos, interrupciones, etc.

Los diferentes registros pertenecientes al módulo ECAN se dividen en 7 secciones:

1) Registros de control y estado general

2) Registros de buffers dedicados a transmisión

3) Registros de buffers dedicados a recepción

4) Registros de buffers programables como transmisores o receptores

5) Registros de control de tasa de datos

6) Registros de control de entrada/salida

7) Registros de control y estado de interrupciones

1) Registros de control y estado general

Estos registros indican el estado operacional de todas las funciones del modulo ECAN:

CANCON: Registros de control CAN

CANSTAT: Registro de estado CAN

ECANCON: Registro de control mejorado CAN

COMSTAT: Registro de estado de comunicación

2) Registros de buffers dedicados a transmisión

Estos registros realizan el control de los buffers transmisores y todas las funciones que ellos desempeñan:

TXBnCON: Control de buffer n ($0 \leq n \leq 2$) Transmisor

TXBnSIDH: Registros identificadores estándar de buffer n ($0 \leq n \leq 2$) transmisor (highbyte)

TXBnSIDL: Registros identificadores estándar de buffer n ($0 \leq n \leq 2$) transmisor (lowbyte)

TXBnEIDH: Registros identificadores extendidos de buffer n ($0 \leq n \leq 2$) transmisor (highbyte)

TXBnEIDL: Registros identificadores extendidos de buffer n ($0 \leq n \leq 2$) transmisor (lowbyte)

TXBnDm: Registros de campo de dato m ($0 \leq n \leq 7$) de buffer transmisor n ($0 \leq n \leq 2$)

TXBnDLC: Registros de longitud de código de datos de buffer transmisor n ($0 \leq n \leq 2$)

TXERRCNT: Registro de contador de error transmisor

3) Registros de buffers dedicados a recepción

Estos registros realizan el control de los buffers receptores y todas las funciones que ellos desempeñan:

RXB0CON: Registro de control de buffer Receptor 0

RXB1CON: Registro de control de buffer Receptor 1

RXBnSIDH: Registros identificadores estándar de buffer receptor n ($0 \leq n \leq 1$) (highbyte)

RXBnSIDL: Registros identificadores estándar de buffer receptor n ($0 \leq n \leq 1$) (lowbyte)

RXBnEIDH: Registros identificadores extendidos de buffer n ($0 \leq n \leq 1$) receptor (highbyte)

RXBnEIDL: Registros identificadores extendidos de buffer n ($0 \leq n \leq 1$) receptor (lowbyte)

RXBnDm: Registros de campo de dato m ($0 \leq n \leq 7$) de buffer receptor n ($0 \leq n \leq 1$)

RXBnDLC: Registros de longitud de código de datos de buffer receptor n ($0 \leq n \leq 1$)

RXERRCNT: Registro de contador de error receptor

4) Registros de buffers programables como transmisores o receptores

Los registros que controlan los buffers programables como transmisor o receptor tienen exactamente las mismas características y funcionalidades que los registros de buffers dedicados a recepción (BnCON, BnSIDH, BnSIDL, BnEIDH, BNEIDL, BnDL, BNDLC).

De igual forma los registros que controlan los filtros de admisión de mensajes: (RXFnSIDH, RXFnSIDL, RXFnEIDH, RXFnEIDL, RXMnSIDH, RXFnSIDL, RXFnEIDH, RXFnEIDL).

SDFLC: Registro contador de longitud de filtros

rxfconn: Registro n de control de filtro receptor

rxfbconn: Registro n de control de buffer de filtro receptor

Msel0: Registro selector de máscara 0

Msel1: Registro selector de máscara 1

Msel2: Registro selector de máscara 2

Msel3: Registro selector de máscara 3

5) Registros de control de la tasa de datos

BRGCON1: Regitro de control de tasa de datos 1

BRGCON2: Regitro de control de tasa de datos 2

BRGCON3: Regitro de control de tasa de datos 3

6) Registros de control de entrada/salida del modulo ECAN

CIOCON: Registro de control I/O

7) Registros de control de interrupciones

PIR3: Registro de bandera de interrupción

PIE3: Registro de habilitación de interrupción

IPR3: Registro de prioridad de interrupción

TXBIE: Registro de habilitación de interrupción en buffer transmisor

BIE0: Registro de habilitación de interrupción de buffer

En las hojas de especificaciones (datasheet) de los microcontroladores se puede encontrar las tablas necesarias para la funcionalidad de los diferentes bits correspondientes a los diferentes registros así como el mapa de memoria y registros.

4.5 PROGRAMAS DE AYUDA PARA CÁLCULO DE TIEMPOS DE BIT CAN DE ACUERDO AL CONTROLADOR UTILIZADO.

En el mercado existen algunos programas o herramientas que realizan labores de cálculo de segmentos de tiempo de bit dependiendo del controlador escogido, es decir mediante el ingreso de parámetros como la frecuencia del oscilador, velocidad de transmisión, retardos en los diferentes dispositivos, etc. El software automáticamente realiza el cálculo de los valores en hexadecimal de los segmentos de tiempo TSEG1 y TSEG2 que conforman el intervalo de bit del protocolo y que serán programados en los registros correspondientes al tiempo de bit según el controlador escogido. El software CANTIME es el más utilizado.

4.5.1 Software CANTIME

En primer lugar el software permite elegir el controlador que se desea utilizar entre los que se encuentran los fabricantes más importantes de controladores CAN como son Infineon, Intel, Motorola, Philips y ST Microelectronics como se muestra en la figura 4.28.

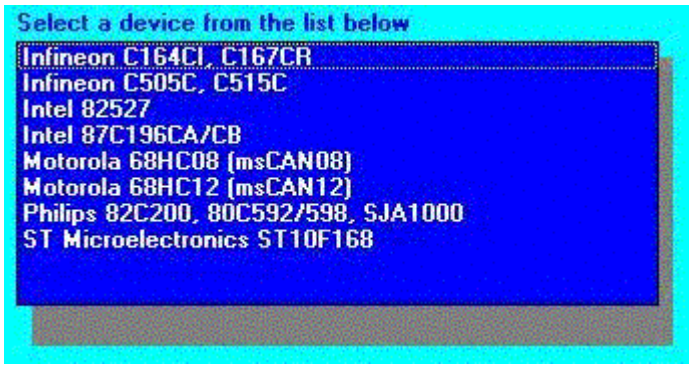


Figura. 4.28. Software CANTIME. Selección de controlador

Posteriormente el software permite ingresar parámetros como los siguientes:

- Frecuencia del oscilador (cristal)
- Velocidad de bus
- Tolerancia en la velocidad del bus
- Retardo de propagación debido al cable utilizado
- Retardo de propagación debido al controlador utilizado
- Retardos debido al transceiver utilizado

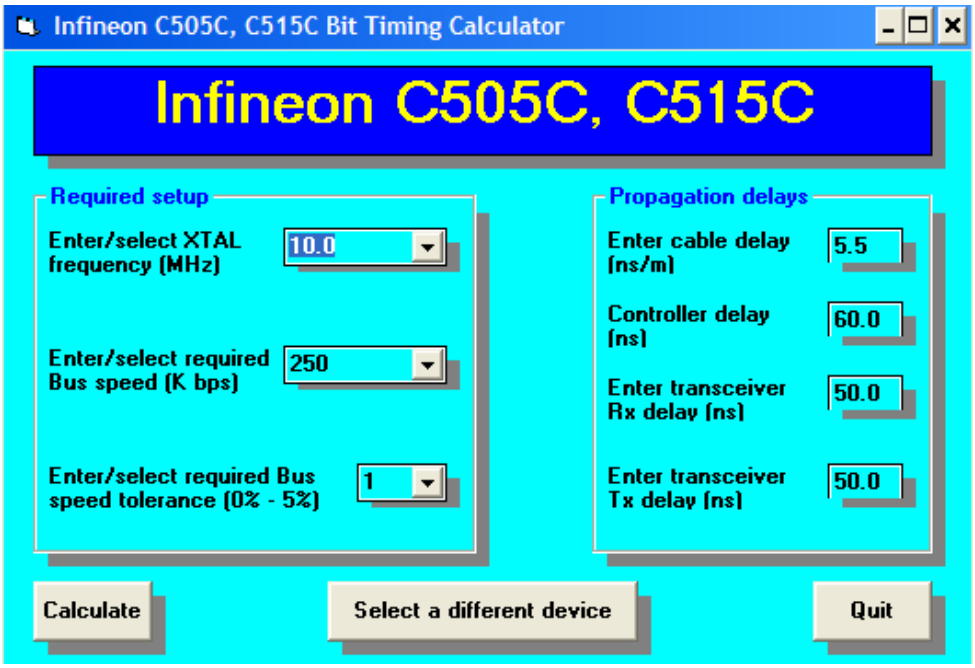


Figura. 4.29. Software CANTIME. Configuración de parámetros para cálculo de tiempos de bit

En el ejemplo de la figura 4.29. se seleccionó el controlador Infineon C505C a una frecuencia de 10 MHz, velocidad de bus de 250 Kbps. Los valores más comunes para retardo del cable y de transceiver son de 5.5 y 50 ns/m respectivamente y el valor de retardo del controlador es de 60 ns/m el cual se configura automáticamente en el software.

Luego de configurar los diferentes parámetros y enviar la orden de cálculo el software entrega diferentes resultados de valores posibles a programarse en los registros correspondientes a TSEG1 y TSEG2 (BTR0 y BTR1) con sus respectivos valores de longitud de bus en metros (Max.Bus) y de salto de ancho de sincronización (SJW). Esto se muestra en la figura 4.30.

BTR0(hex)	BTR1(hex)	Samples	Spl%	SJW	Max.Bus(m)	Kbps
04	14	1	75	1	198	250
03	16	1	80	1	225	250
04	23	1	62	1	152	250
44	23	1	62	2	107	250
03	25	1	70	1	189	250
43	25	1	70	2	152	250
01	2F	1	85	1	261	250
41	2F	1	85	2	243	250

Figura. 4.30. Software CANTIME. Resultados de cálculo de TSEG1 y TSEG2

4.6 TRANSCEIVERS CAN

Para acoplar las señales que viajan por el bus (CAN_H y CAN_L) a las entradas de los controladores CAN Stand Alone o a las entradas de los módulos CAN incluidos en los diferentes microcontroladores, se utilizan dispositivos llamados transceivers que son circuitos integrados disponibles en el mercado. Se debe tomar en cuenta que algunos controladores ya traen un transceiver interno.

Dentro de un transceiver existe la circuitería necesaria para realizar el acople de señales, mediante un comparador, la señal diferencial existente entre las líneas del bus CAN son convertidas a una señal de nivel lógico correspondiente RxD. De igual forma una salida del módulo CAN del microcontrolador es conectada al transceiver, el cual mediante una circuiteria apropiada (driver) envía la señal hacia el bus.

Otro parámetro importante en los transceivers CAN es el rango de voltajes que soporta el dispositivo. La mayoría de transceivers ofrecen un amplio rango de voltajes y garantizan de esta forma una transmisión de señales sin distorsión inclusive en ambientes con interferencia electromagnética alta.

La operación de un transceiver generalmente requiere de una configuración apropiada del registro de control de salida del microcontrolador utilizado.

Además de proveer los voltajes de referencia adecuados, los transceivers pueden desarrollar otras funciones como protección de sobrecarga y cortocircuito, reconocimiento de errores en el bus, etc.

Una característica importante de los transceivers CAN es la reducción de la interferencia electromagnética denominada “slope control”, mediante el aumento o reducción de tiempos de las señales CAN_H y CAN_L.

Muchos controladores CAN requieren una señal de entrada diferencial, para este fin, el transceiver provee una salida de voltaje de referencia que puede ser conectada al controlador CAN mediante una circuitería mínima adicional.

4.6.1 Transceivers CAN disponibles en el mercado.

Existen diversos transceivers disponibles en el mercado los cuales se dividen generalmente de acuerdo a su velocidad es decir si son para redes CAN de alta velocidad o de baja velocidad.

4.6.1.1 Microchip MCP2551

Uno de los transceivers más utilizados es el de MICROCHIP MCP2551 el cual es un transceiver de alta velocidad que soporta transferencias de datos hasta de 1Mbps y sus características principales son las siguientes:

- Trabaja bajo la norma ISO 11898.
- Apropiado para sistemas de 12 y 24 voltios.
- Slope Control (reducción de interferencia electromagnética).
- Protección contra daños debido a corto circuito.

- Protección contra transitorios de alto voltaje.

- Se pueden conectar hasta 112 nodos.

El transceiver MCP2551, como la gran mayoría de transceivers CAN, consta de 8 pines que se muestran en la figura 4.3. y su descripción en la tabla 4.3:

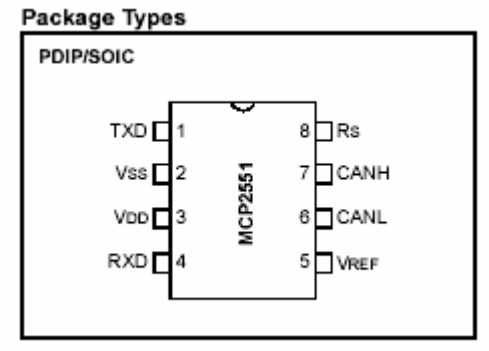


Figura. 4.31. Presentación MCP2551

Numero de Pin	Nombre del Pin	Función
1	TxD	Transmisión de datos (IN)
2	Vss	Tierra
3	Vdd	Voltaje de alimentación
4	RxD	Recepción de datos (OUT)
5	VREF	Voltaje de referencia
6	CAN_L	Voltaje de nivel bajo CAN (I/O)
7	CAN_H	Voltaje de nivel alto CAN (I/O)
8	RS	Slope Control

Tabla. 4.3. Descripción de pines MCP2551

En la tabla 4.4. se muestran las características principales de algunos transceivers de diversos fabricantes que trabajan a altas velocidades bajo la norma ISO 11898-2

Así mismo en la tabla 4.5. se muestran las características principales de algunos transceivers de diversos fabricantes que trabajan a bajas velocidades bajo las normas ISO 11898-3, SAE J2411 o ISO 11992.

Fabricante	**Infineon TLE6250**	**Philips 80C250**	**Philips 80C251**	**Philips TJA 1050**	**Temic Si9200E**	**Unitrode UC5350**
Velocidad (Mbps)	1	1	1	1	1	1
Slope Control	No	Variable	variable	No	no	Variable
Rango Voltaje (V)	-2...+7	-7...+12	-7...+12	-7...+12	-2...+7	-25...+18
Retardo (ns)	150	170	170	150	120	100
Consumo en estado dominante (mA)	70	70	80	75	70	70
Consumo en estado recesivo (uA)	10	170	250	170	170	1000

Tabla. 4.4. Transceivers CAN de alta velocidad

Fabricante	**Infineon TLE6250**	**Philips 80C250**	**Philips 80C251**	**Philips TJA 1050**	**Temic Si9200E**	**Unitrode UC5350**
Velocidad (Kbps)	125	100	125	125	125	250
Especificación	ISO 11898-3	SAE J2411	ISO 11898-3	ISO 11898-3	ISO 11898-3	ISO 11992
Consumo en estado dominante (mA)	10	9	5	35	27	26
Consumo en estado recesivo (uA)	55	40	30	75	50	50

Tabla. 4.5. Transceivers CAN de baja velocidad

CAPITULO 5

PROTOCOLOS DE ALTO NIVEL

5.1 INTRODUCCIÓN

Muchas tecnologías de redes de comunicaciones están disponibles “on-chip” es decir mediante circuitos integrados o microcontroladores. Los microcontroladores carecen de un protocolo de alto nivel estándar dominante. Todas las interfaces de tipo serial proveídas en microcontroladores solo incluyen algunas de las características de la capa de enlace de datos. Esto implica que se dispone de funciones para transmitir y recibir datos pero no se define otras características importantes como variables de proceso o el tipo de dato que se está transmitiendo. En el momento en que se especifica características como el tipo de datos, administración de red o identificadores de mensajes a ser utilizados para servicios específicos, se define un protocolo de alto nivel.

Muchas funciones adicionales son necesarias o al menos muy utilizadas para la implementación de sistemas distribuidos más sofisticados. En particular, servicios para la transmisión de bloques de datos de más de ocho bytes (protocolo CAN), funciones de administración de red apropiadas, la asignación transparente de identificadores de mensajes o la configuración de parámetros para nodos específicos.

Una característica muy importante en la tecnología de automatización es la interoperabilidad e intercambiabilidad de dispositivos de diferentes fabricantes; para esto es necesario proveer una descripción estandarizada de funcionalidades de estos dispositivos y definir perfiles de dispositivos estandarizados.

5.2 PROTOCOLOS DE ALTO NIVEL BASADOS EN CAN

Los protocolos de alto nivel CAN se han desarrollado de acuerdo a las características y funcionalidades que son capaces de ofrecer. Los principales son CAL, CanOpen, DeviceNet y SAEJ1939.

Los protocolos de alto nivel desarrollados para aplicaciones en redes CAN dan su primer paso con CAL (CAN Aplication Layer), los servicios definidos por CAL pueden ser aplicados de acuerdo a requerimientos específicos de una aplicación. Es decir que CAN proporciona una plataforma de comunicación para la implementación optimizada de aplicaciones distribuidas específicas.

Sin embargo para la implementación de sistemas de automatización distribuidos estandarizados se necesitan definiciones adicionales a una plataforma de comunicación estandarizada. El objetivo más importante es la provisión de sistemas del mismo tipo y comportamiento para reducir el esfuerzo necesario en labores de diseño, prueba, instalación y mantenimiento. Esto se consiguió mediante la estandarización de funciones de dispositivos ya que es la condición esencial para la intercambiabilidad de dispositivos de diferentes fabricantes. Los protocolos de alto nivel que cumplen estas características son DeviceNet representado por ODVA (Open DeviceNet Vendors Asociation) y el más utilizado en la actualidad CanOpen definido por CIA (CAN in Automation).

El protocolo estandarizado basado en CAN SAE J1939 se desarrolló específicamente aplicaciones dentro del campo de vehículos comerciales sin embargo está tomando fuerza en otros tipos de aplicaciones. También especifica una capa de aplicación estandarizada mediante un sistema de numeración. Estos números son transmitidos con cada mensaje dentro de los 29 bits identificadores.

En las siguientes secciones se describen las características principales de cada protocolo de alto nivel CAN profundizando un poco más en el protocolo CanOpen por ser el más utilizado en la actualidad.

5.3 CAL (CAN APLICATION LAYER)

CAL habilita la descripción de aplicaciones distribuidas mediante objetos de comunicación estandarizados y define protocolos correspondientes al intercambio de estos objetos. Además ofrece tipos de datos estándar y reglas para la codificación de bits para la transmisión. También proporciona características de administración de red y configuración de parámetros para nodos específicos. CAL trabaja bajo el modelo Cliente – Servidor.

5.3.1 Arquitectura CAL

La arquitectura CAL, como se muestra en la figura 5.1, consta de 4 elementos de servicio básicos:

1) CMS (CAN Message Specification)

2) NMT (Network Management)

3) DBT (Identifier Distributor)

4) LMT (Layer Management)

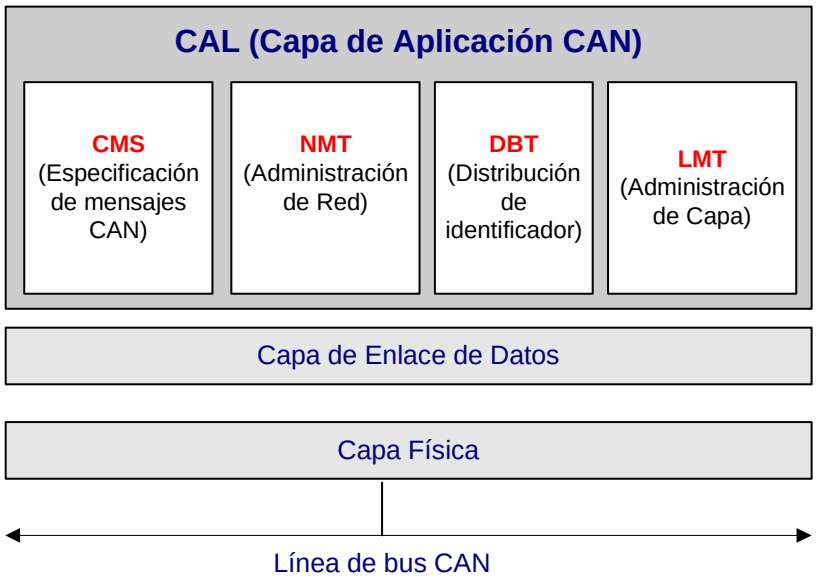


Figura. 5.1. Sistema de comunicación CAN mediante CAL

5.3.1.1 CMS (CAN Message Specification)

Mediante este elemento de servicio, los objetos de comunicación estandarizados son definidos para la descripción de funciones y designados para procesos de aplicación. Los objetos de comunicación se dividen en variables, eventos y dominios.

Variables:

Las variables pueden ser de tipo solo lectura, solo escritura o de lectura/escritura.

Las variables de tipo solo lectura pueden ser utilizadas por el cliente para leer el dato de un nodo remoto, este dato será el enviado por el servidor en el último servicio de "actualización de variables" ejecutado.

Las variables de tipo solo escritura pueden ser utilizadas por el cliente para proporcionar datos o realizar una petición (con el fin de ejecutar un comando) a uno o más servidores

Las variables de tipo lectura/escritura permiten al cliente recolectar el dato actual desde el servidor o pedir al servidor que ejecute un comando y ser informado del resultado de la ejecución.

Eventos:

Los eventos pueden ser utilizados para monitorear comportamientos específicos como un valor de temperatura excediendo un cierto límite. La ocurrencia del evento es detectada por el servidor y notificada al cliente. Un evento puede proporcionar información adicional sobre las razones por las que se provocó dicho evento.

Dominios:

Los dominios básicos pueden ser utilizados para transferir un bloque largo arbitrario de datos desde un cliente a un servidor o viceversa. Los dominios “multilpexados” se pueden utilizar para transferir conjuntos de datos múltiples entre un cliente y un servidor o viceversa.

Un dominio es transferido como una secuencia de segmentos; previo a la transferencia se da una fase de inicialización y preparación por parte del cliente y el servidor.

5.3.1.2 NMT (Network Management)

El elemento NMT o Administración de Red se dedica principalmente a las labores de configuración, inicialización y monitoreo de nodos. Para guardar los identificadores de mensajes, NMT se basa en un esquema lógico maestro - esclavo, donde un nodo del sistema desarrolla la función de nodo maestro.

Los servicios ofrecidos por NMT son:

- Control de módulo.
- Control de Error.
- Configuración

A través del control de módulo, el maestro NMT inicializa a los esclavos NMT que quieren tomar parte en la aplicación y permite que se comuniquen entre ellos. En el servicio de control de error el maestro detecta fallas remotas en la red CAN.

El servicio de configuración permite activar desactivar propiedades de los nodos como parámetros, datos o códigos ejecutables.

5.3.1.3 DBT (Identifier Distributor)

Este elemento de servicio provee la asignación dinámica de identificadores de mensajes en los objetos de comunicación lógica durante la inicialización del sistema.

5.3.1.4 LMT (Layer Management)

Mediante este servicio, se pueden configurar diversos parámetros por medio de un nodo (LMT-maestro) y transmitirlos hacia otros nodos (LMT- esclavos) a través de la red.

Este servicio permite la configuración de direcciones a los esclavos, las cuales incluyen el nombre del nodo, el identificador del nodo y los parámetros de tiempo de la capa física.

En el modo de configuración, el direccionamiento de un nodo está basado en el nombre del fabricante, nombre del producto y número de serie.

5.3.2 Modelo Cliente - Servidor

Como ya se indicó CAL describe una interacción entre procesos de aplicación por medio del modelo cliente – servidor. Un servicio CAL por lo tanto es requerido mediante una “primitiva de requerimiento”. Una “indicación de servicio”, informa a un proceso de aplicación remoto que un cierto servicio CAL ha sido requerido. Mediante el “servicio de confirmación”, el resultado es transmitido en la “respuesta de servicio” como positivo o negativo.

Existen cuatro tipos de servicios distinguidos por CAL:

1) Servicios Locales: La aplicación puede requerir un servicio a ser transportado localmente por CAL mediante un requerimiento.

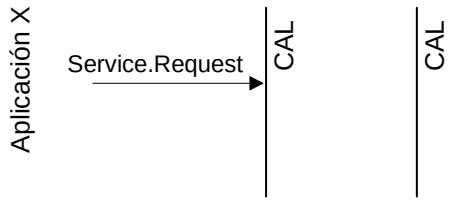


Figura. 5.2. Servicio Local CAL

2) Servicio proveedor inicial: Mediante este servicio se informa a la aplicación que un evento es detectado por CAL mediante una primitiva de indicación.

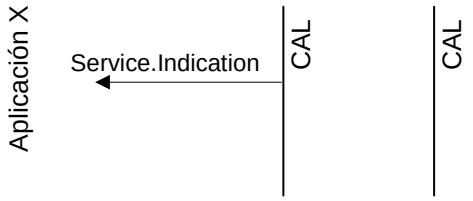


Figura. 5.3. Servicio proveedor inicial CAL

3) Servicio no confirmado: El requerimiento de un servicio no confirmado por una aplicación (cliente) causa que CAL transmita el mensaje de requerimiento correspondiente a los otros nodos (servidores). El requerimiento de servicio es indicado a las aplicaciones en los otros nodos mediante un servicio de indicación.

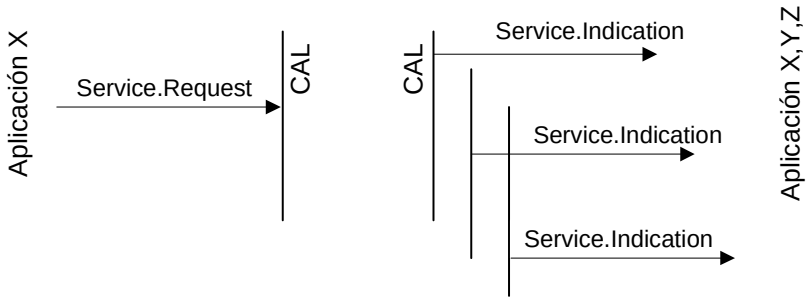


Figura. 5.4. Servicio no confirmado CAL

4) Servicio Confirmado: Mediante un servicio confirmado, una aplicación (cliente) puede pedir a un nodo específico remoto (servidor) que provea cierta información. La aplicación de servidor indica el resultado mediante un servicio de respuesta (positivo o negativo). La respuesta del servidor es transmitida por CAL mediante mensajes CAN al nodo que la requería (cliente) y se realiza una primitiva de confirmación a la aplicación del cliente.

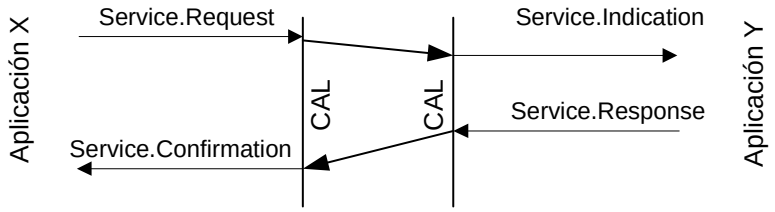


Figura. 5.5. Servicio Confirmado CAL

5.4 CANOPEN

5.4.1 Introducción

CANopen es un estándar perteneciente a CIA (CAN In Automatization o Automatización en CAN) y está basado en el "comunication profile" o perfil de comunicación, el cual especifica los mecanismos de comunicación base y sus definiciones.

Los diferentes tipos de dispositivos más utilizados en la tecnología de automatización industrial como módulos de entrada/salida digitales y análogos, manejadores (drivers), controladores, dispositivos programables, etc; se definen en los "device profiles" o perfiles de dispositivos.

En CANopen se pueden obtener hasta 127 nodos, cada uno de ellos tiene una dirección específica, es decir que CANopen realiza un direccionamiento de nodos (0 - 127) además de manejar identificadores de mensajes.

Existen diferentes conceptos que se deben conocer antes de incursionar en las diferentes aplicaciones que ofrece CANopen.

5.4.2 El Diccionario de Objetos (DO)

La parte fundamental de cualquier nodo CANopen es el Diccionario de Objetos (DO), que es una tabla de búsqueda con un índice de 16 bits (0000h - ffffh) que permite hasta 65.536 entradas y un subíndice de 8 bits (00h - ffh), lo cual permite hasta 256 subentradas en cada índice. Cada entrada puede proporcionar una variable de cualquier tipo y longitud.

El Diccionario de Objetos contiene una descripción de la configuración y funcionalidad del nodo CANopen que lo contiene y puede ser leído y escrito por otros nodos CANopen. Adicionalmente el DO es utilizado para almacenar información específica de aplicación que es utilizada por el nodo. Esta información también puede ser utilizada por otros nodos de la red.

Mediante la escritura de datos en las entradas del Diccionario de Objetos de un nodo, dicho nodo puede ser instruido para realizar una operación de cualquier tipo por ejemplo muestrear la temperatura actual de cierto proceso y colocar el dato muestreado en su diccionario de objetos para ser leído por otros nodos.

Mediante la lectura de datos en las entradas del Diccionario de Objetos de un nodo, otros nodos pueden encontrar información sobre sus operaciones. Dependiendo de los requerimientos del diseño de la red, la cantidad de información puede variar de aplicación a aplicación es decir que un nodo puede presentar información descriptiva completa o información mínima según los requerimientos. Sin embargo existe información dentro del Diccionario de Objetos que es obligatoria y se debe presentar.

Todos los procesos y datos relacionados con la comunicación se encuentran almacenados como entradas en direcciones predefinidas del Diccionario de Objetos.

Para las entradas que almacenan un solo valor existe una sola subentrada con el subíndice 00h. Las entradas que almacenan más de un valor deben tener una subentrada o subíndice por cada valor. Además en el subíndice 00h debe constar el número de valores a almacenar.

Ejemplo:

La entrada de índice 2000h almacena un solo valor de 8 bits

Indice 2000h Subíndice 00h = valor 8 bits

La entrada de índice 2001h almacena dos valores de 8 bits

Indice 2000h Subíndice 00h = 2

Indice 2000h Subíndice 01h = primer valor 8 bits

Indice 2000h Subíndice 02h = segundo valor 8 bits

Cada nodo debe tener su propio Diccionario de Objetos. No todas las entradas del DO son utilizadas.

La tabla 5.1. muestra la organización del Diccionario de Objetos

Rango de Índices	Descripción
0000h	Reservado
0001h – 0FFFh	Tipo de Datos
1000h – 1FFFh	Entradas de comunicación
2000h – 5FFFh	Especificaciones de fabricante
6000h – 9FFFh	Parámetros de perfil de dispositivos
A000h – FFFh	Reservado

Tabla. 5.1. Organización del Diccionario de Objetos

El DO no solo proporciona una manera de asociar variables con un valor de índice y subíndice sino también especifica el tipo de dato. Debe notarse que en las direcciones correspondientes a tipos de datos no se puede realizar ningún almacenamiento de variables; las entradas a partir de 1000h son utilizadas para el almacenamiento.

5.4.3 Perfiles de Dispositivos (Device Profiles)

Existen entradas obligatorias que todos los nodos CANopen deben soportar. Estas incluyen un objeto de identidad mediante el cual el nodo puede identificarse a sí mismo y un objeto de error para reportar estados de error potenciales. Adicionalmente se pueden especificar entradas opcionales. Los perfiles de dispositivos son especificaciones añadidas que describen todos los parámetros de comunicación y entradas del diccionario de objetos que son soportadas por un cierto tipo de módulo CANopen. Algunos perfiles están disponibles para módulos genéricos de entrada/salida, codificadores y otros dispositivos.

El nodo maestro puede tener acceso de lectura al objeto de identidad de cualquier nodo esclavo mediante el uso de los SDO o Service Data Object (sección 5.4.5). Como contestación, este recibe un SDO con la información sobre el perfil de dispositivo que conforma el módulo. Asumiendo que el maestro conoce las entradas de objetos que son definidas por el perfil del dispositivo en particular, también conoce las entradas del DO que son soportadas por el nodo y se puede acceder directamente.

5.4.4 Hojas Electrónicas de Datos (EDS)

Las Hojas de Datos Electrónicas “EDS” ofrecen una forma estandarizada de especificar entradas del Diccionario de Objetos. Cualquier fabricante de un módulo CANopen entrega el módulo junto con un archivo, el cual en diseño es similar a un “.ini” utilizado en los sistemas operativos Microsoft Windows. Estos archivos se denominan EDS.

Un maestro o herramienta de configuración CANopen corriendo en un PC que tiene una tarjeta CAN puede cargar directamente las EDS en el juego de dispositivos reconocidos.

Una vez que se encuentra el dispositivo, el maestro buscará sus correspondientes Hojas de Datos Electrónicas, y al reconocerlas, el maestro conocerá las entradas del DO que soporta el dispositivo.

Los Perfiles de Dispositivos o Device Profiles, especifican el mínimo de entradas que deben ser soportadas por un dispositivo de acuerdo al perfil. Sin embargo, las EDS especifican solo los objetos que están relacionados a cierto fabricante o a un sub-tipo de módulo.

5.4.5 Service Data Objects (SDO)

Los Objetos de Servicio de Datos o SDOs están relacionados a la obtención de un canal directo para la comunicación. El maestro puede ser capaz de leer y/o escribir en las entradas del DO de todos los nodos conectados a la red.

CANopen soporta un método de comunicación en el cual se puede acceder a los DO de los nodos mediante requerimientos de lectura/escritura. Estos requerimientos toman el nombre de SDOs.

Por defecto solamente un nodo llamado maestro o administrador tiene el derecho de iniciar activamente este modo de comunicación SDO. Sin embargo existen formas de que otros nodos requieran el uso de los canales de comunicación SDO para labores de análisis o configuración.

La comunicación SDO por defecto es de tipo maestro/esclavo o requerimiento/respuesta donde el maestro posee todos los canales de comunicación y tiene un canal disponible para cada nodo en el sistema. Solo el nodo que posee un canal puede enviar un requerimiento SDO de lectura o escritura hacia otro nodo (y su diccionario de objetos) el cual enviará una respuesta SDO confirmando el acceso de lectura o escritura.

La comunicación SDO también permite realizar transferencias segmentadas, las cuales permiten transmitir las entradas del diccionario de objetos sin importar su tamaño. Si el

contenido no encaja en un mensaje simple, este es segmentado automáticamente y transferido mediante múltiples mensajes.

Como ya se dijo, la metodología SDO permite al maestro tener acceso de lectura y escritura a todas las entradas del DO de todos los nodos pertenecientes a la red. Debido a que tanto los procesos como la configuración de datos forman parte del DO, el dato procesado puede ser actualizado mediante transferencias SDO.

Por definición los SDOs siempre contienen una longitud de mensajes de 8 bytes inclusive si el SDO contiene solo 1 byte de datos o un simple reconocimiento sin ningún dato procesado.

Para realizar una comunicación SDO se deben realizar los siguientes pasos:

1) El maestro envía un requerimiento de lectura SDO a un nodo de entrada.

2) El nodo de entrada contesta con una respuesta SDO y el dato requerido.

3) El maestro envía un requerimiento de escritura SDO a un nodo de salida.

4) El nodo de salida confirma con una respuesta SDO.

Para implementar un canal de comunicaciones punto a punto, deben ser reservados dos identificadores de mensajes CAN que se encuentran dentro del COB ID o identificador de objeto de conexión. Estos identificadores tienen que ver con el requerimiento de envío de un nodo específico y la respuesta enviada por el nodo. La figura 5.6. muestra los identificadores de mensajes que se utilizan por defecto.

El identificador de mensajes que se utiliza para enviar un requerimiento hacia un nodo específico se calcula sumando el ID del nodo (1 – 127) a la dirección base 600h. Las direcciones 601h hasta la 67Fh son utilizadas para proveer 127 canales desde un cliente hacia 127 servidores. El identificador de mensajes utilizados para enviar una respuesta

desde cada nodo hacia el cliente que envió el requerimiento se calcula sumando el ID del nodo (1 - 127) a la dirección base 580h. De esta manera se obtienen 127 canales para enviar respuestas desde 127 servidores hacia el cliente en las direcciones 581h hasta 5FFh.

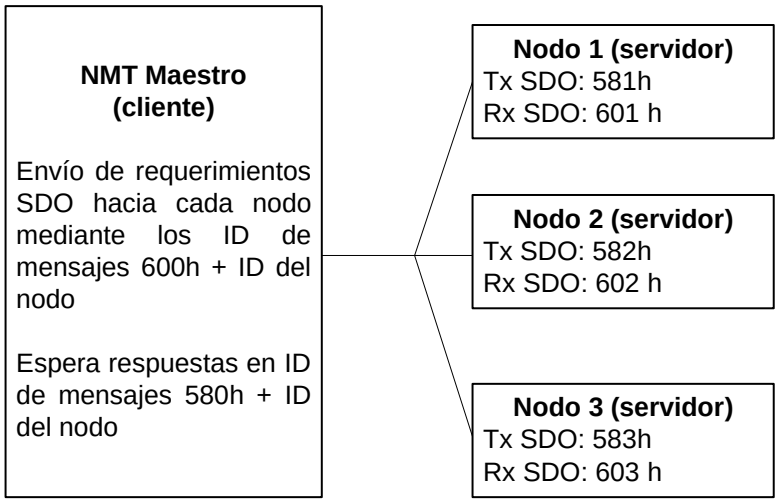


Figura. 5.6. Identificadores de mensajes por defecto para comunicación SDO

En el esquema utilizado para la asignación de identificadores de mensajes se permite un solo cliente en la red y es el que tiene acceso a todas las entradas del DO de todos los nodos que integran la red.

5.4.5.1 Contenido de los mensajes SDO

Cada mensaje de requerimiento y respuesta SDO contienen 8 bytes de datos de los cuales el primero es llamado byte especificador. Los bits de dicho byte especifican si la acción es de lectura, escritura o aborto de mensaje. Otros bits son utilizados para especificar si la transferencia es completa o segmentada en la cual se dividen los datos en segmentos y se envían en múltiples mensajes.

Los bytes 2 a 4 contienen el multiplexor, es decir la combinación de 16 bits del índice y 8 bits de subíndice identificando la entrada del diccionario de objetos que es accedida con el correspondiente SDO. Se envía primero el byte más bajo de los 16 bits del índice luego

el byte más alto de los 16 bits de índice y finalmente el byte correspondiente a los 8 bits de subíndices.

Los bytes restantes (5 a 8) son utilizados para transmitir el dato. Si el dato es de 4 bytes o menor es parte del mensaje completo, de lo contrario, el resto del dato se envía en nuevos mensajes (segmentados).

5.4.6 Process Data Objects (PDO)

Para la mayoría de aplicaciones, la comunicación SDO no es suficientemente eficiente para manejar el intercambio de datos de procesos reales. Como CAN soporta el concepto de comunicación multi-maestra con prioridad de mensajes, es necesario un método de comunicación más directo para permitir un acceso más eficiente a los datos de proceso.

Los PDOs implementan una solución para colocar múltiples variables de proceso del DO en un solo mensaje CAN de hasta 8 bytes. Mediante el proceso de mapeo PDO, cualquier entrada del diccionario puede ser colocada como un dato PDO con la única limitación de no contener más de 8 bytes.

El máximo número de bits disponibles en un PDO es de 64, es decir que en un PDO pueden ser mapeadas un total de 64 entradas del DO si cada entrada tiene un solo bit de largo como se muestra en la tabla 5.2.

SUBINDICE	NOMBRE	TIPO DE DATO
0	Número de entradas	UNSIGNED8
1	Primera entrada mapeada del OD	UNSIGNED32
2	Segunda entrada mapeada del OD	UNSIGNED32
3	Tercera entrada mapeada del OD	UNSIGNED32
4	Cuarta entrada mapeada del OD	UNSIGNED32
---	---	
64	64ava entrada mapeada del OD	UNSIGNED32

Tabla .5.2. Parámetros de mapeo PDO

En el DO el área de índices desde 1600h hasta 17FFh es reservada para los parámetros de mapeo RPDO y el rango de índices desde 1A00h hasta 1BFFh es reservada para los parámetros de mapeo TPDO.

En la figura. 5.7. se muestra un ejemplo de un mapeo PDO donde un nodo de entrada CANopen soporta dos variables de entradas digitales de 8 bits cada una y dos variables de entradas análogas de 16 bits cada una. De acuerdo con el perfil de dispositivos (device profile) para módulos de entrada/salida genéricos, las entradas del DO en el índice 6000h almacenan las dos variables de entrada digitales de 8 bits y en el índice 6401h se almacenan las dos variables análogas de 16 bits.

INDICE	SUBINDICE	TIPO	DESCRIPCION
1A00h			Mapeo - primer PDO Transmisor
	0	Unsigned8	4
	1	Unsigned32	6000 01 08h
	2	Unsigned32	6000 02 08h
	3	Unsigned32	6401 01 10h
	4	Unsigned32	6401 02 10h
6000h			Datos de proceso - entradas digitales
	0	Unsigned8	2
	1	Unsigned8	entrada digital de 8 bits
	2	Unsigned8	entrada digital de 8 bits
6401h			Datos de proceso - entradas análogas
	0	Unsigned8	2
	1	Unsigned16	entrada análoga de 16 bits
	2	Unsigned16	entrada análoga de 16 bits

TPDO1	D IN 1	D IN 2	A IN 2	A IN 2
	Byte 1	Byte 2	Bytes 3, 4	Bytes 5, 6

Figura. 5.7. Ejemplo de mapeo PDO

Las entradas del OD con índice 1A00h especifican el mapeo PDO, indicando bit por bit cuales entradas del OD son utilizadas para el primer PDO transmisor (TPDO1). Cada entrada inicia utilizando el primer bit libre disponible en el PDO y ocupa tantos bits como requiera.

El primer subíndice (0h) de la entrada 1A00h indica el número de entradas mapeadas en el PDO en este caso 4. El segundo subíndice (1h) de la entrada 1A00h contiene los valores 6000 01 08h es decir realiza el mapeo de la entrada de índice 6000h, subíndice 1, y 8 bits de longitud de la primera variable digital de entrada. El tercer subíndice (2h) de la entrada 1A00h contiene los valores 6000 02 08h es decir realiza el mapeo de la entrada de índice 6000h, subíndice 2, y 8 bits de longitud de la segunda variable digital de entrada. De igual forma el cuarto subíndice (3h) de la entrada 1A00h contiene los valores 6401 01 10h es decir realiza el mapeo de la entrada de índice 6401h, subíndice 1, y 16 bits (en hexadecimal 10h) de longitud de la primera variable análoga de entrada, y de igual forma se realiza con la segunda variable análoga de entrada.

En el ejemplo los bits restantes del TPDO1 (bytes 7 y 8) permanecen sin utilizarse.

5.4.6.1 Enlaces PDO

La comunicación POD por defecto es similar a la comunicación mediante SDOs. El maestro es el único nodo que recibe objetos de datos de proceso transmisores o TPDOs y es el único que puede enviar objetos de datos de proceso receptores o RPDOs a los esclavos como se muestra en la figura 5.8.

En CANopen el COB ID o identificador de objetos de conexión es el que contiene el identificador de mensajes CAN y algunos bits adicionales de configuración como habilitación y desactivación de PDOs.

Existe un único identificador de mensajes COB ID para cada TPOD y uno para cada RPOD.

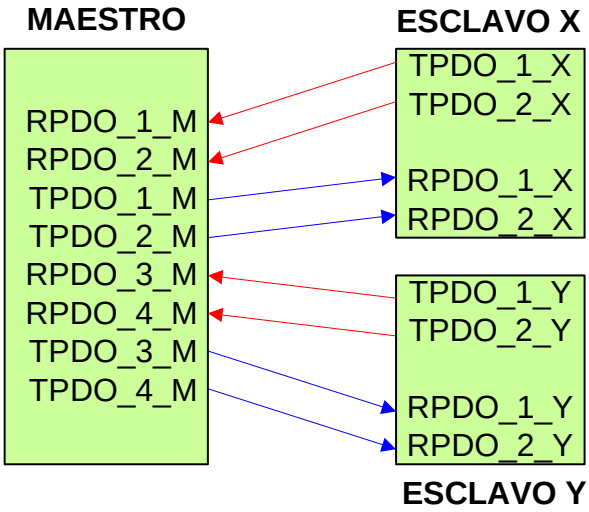


Figura. 5.8. Enlace PDO por defecto

Durante el ciclo de inicialización y configuración el enlace PDO puede ser cambiado. El maestro puede informar a uno o varios módulos de salida que pueden escuchar directamente a un TPDO específico proveniente de cualquier módulo de entrada. Es decir que cada nodo puede ser informado de las tramas de mensajes pertinentes para él y las que deben ser ignoradas de acuerdo a su identificador.

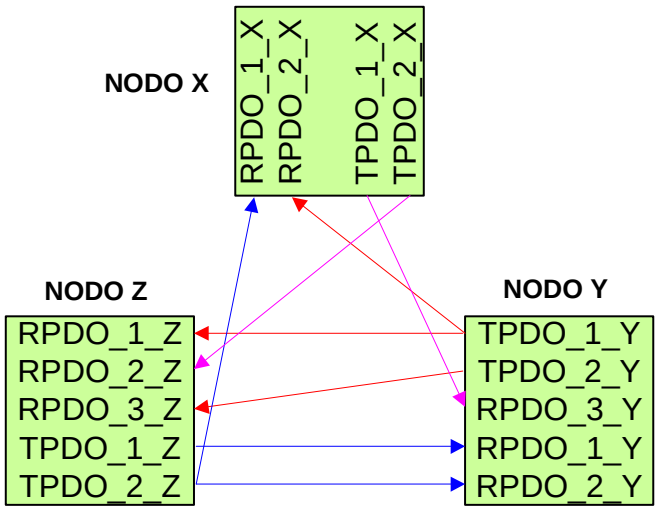


Figura. 5.9. Enlace directo PDO optimizado

Una vez que el enlace se ha realizado y la red entra en su estado operacional, el maestro no necesita estar involucrado en el proceso de comunicación de datos y se puede enfocar en otras aplicaciones como administración de red.

Dadas estas características, la red puede quedar configurada como una arquitectura multi-maestra (figura 5.9.) donde cada nodo puede enviar mensajes (TPDOs) en el momento que desee mediante identificadores de mensajes y de igual forma los nodos interesados los recibirán (RPDOs) tal como implementa el protocolo CAN.

5.4.6.2 Modos de activación PDO

Dentro de CANopen existen 4 modos de activación (trigger) para el funcionamiento de los PDOs:

1) Event driven

Si un dispositivo de entrada reconoce un cambio de estado (COS por sus siglas en inglés *change of state*) en cualquiera de sus entradas, éste actualiza el dato en el DO y transmite el PDO. Este modo permite los tiempos de respuesta más rápidos.

Si un TPDO contiene un juego de entradas digitales, un evento se define como cualquier cambio en dichas entradas y el TPDO será transmitido tan pronto como se haya dado el evento.

En caso de que se de una entrada cambie constantemente, el TPDO debe ser transmitido continuamente ya que en seguida de transmitir un TPDO ya existe un nuevo cambio en la entrada. CANopen maneja este caso introduciendo un temporizador de inhibición *"inhibit timer"* configurable a múltiplos de centenares de microsegundos. Después de que se ha transmitido un TPDO, el temporizador debe expirar para que el TPDO pueda ser transmitido nuevamente. En caso de que el temporizador sea configurado en 10 milisegundos, el peor caso es que se transmita el TPDO cada 10 milisegundos.

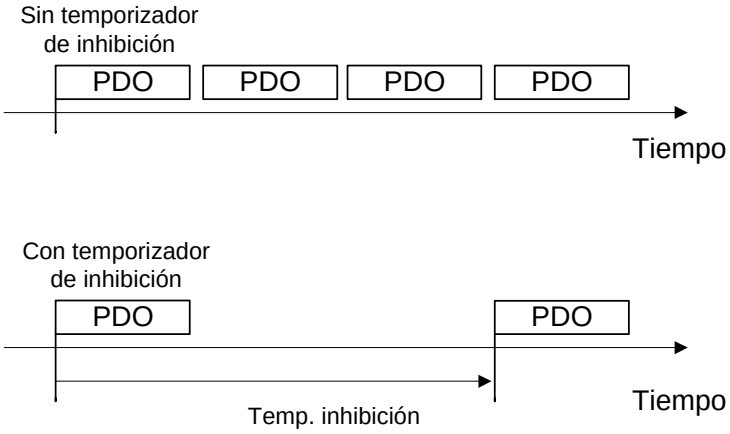


Figura. 5.10. Temporizador de inhibición

2) Time driven

El PDO puede ser configurado para transmitir en bases de tiempos fijas mediante un temporizador de eventos "*Event timer*" el cual es local en cada nodo CANopen. El temporizador es especificado en milisegundos; por ejemplo si se lo configura en 50 milisegundos, el TPDO será transmitido cada 50 milisegundos.

La comunicación Time Driven mejora el desempeño de la red pero a la vez ocasiona una mayor carga en los nodos, ya que los TPDOs se transmiten inclusive si no existe un cambio de estado en las entradas.

3) Individual polling

Utilizando una características regular CAN, el requerimiento de un mensaje mediante una trama de datos remota permite que un PDO sea transmitido solo si el dato es requerido específicamente por un nodo.

4) Synchronized

La idea principal de este modo es la de proveer movimientos orientados a sistemas como robots donde las diferentes entradas y salidas son "paralelizadas" y deben estar perfectamente sincronizadas.

Para el único propósito de sincronización CANopen utiliza una señal llamada SYNC la cual es un mensaje específico que no contiene ningún dato. La figura 5.11. muestra como es sincronizado el dato de un sensor, por ejemplo un sensor que realiza mediciones de posición del brazo de un robot.

Los sensores leen constantemente sus datos de entrada y guardan una copia actual del mensaje en el buffer transmisor. Al recibir el mensaje SYNC, todos los sensores detienen la actualización del buffer y empiezan a transmitir el dato.

La sincronización de las salidas (actuadores) trabajan de manera similar. Una vez que la unidad de procesamiento de datos tiene nuevos valores para sus salidas o actuadores, los transmite serialmente por la red tomando en cuenta la prioridad de mensajes. Los actuadores reciben el mensaje guardando el dato recibido en sus buffers receptores sin aplicar el dato a sus salidas. Ellos esperan a la siguiente señal SYNC y solo después de la recepción de dicha señal aplican el dato recibido en sus buffers a sus salidas en forma paralela.

Terminología utilizada en la sincronización

SYNC COB ID

Es un parámetro configurable individualmente en todos los nodos que soportan comunicación sincronizada por lo que puede haber más de una señal SYNC en el sistema. Especifica cual identificador de mensajes es utilizado con la señal SYNC.

SYNC PRODUCER

Es un solo nodo capaz de producir la señal SYNC. Por lo general este nodo es el maestro o el nodo encargado de la configuración del sistema sin embargo no siempre debe ser así.

COMMUNICATION CYCLE PERIOD

El período de tiempo de comunicación es un período de tiempo en microsegundos en el cual va a ocurrir la señal SYNC. Los nodos que soportan comunicación sincronizada tienen disponible este valor en la entrada 1006h del OD

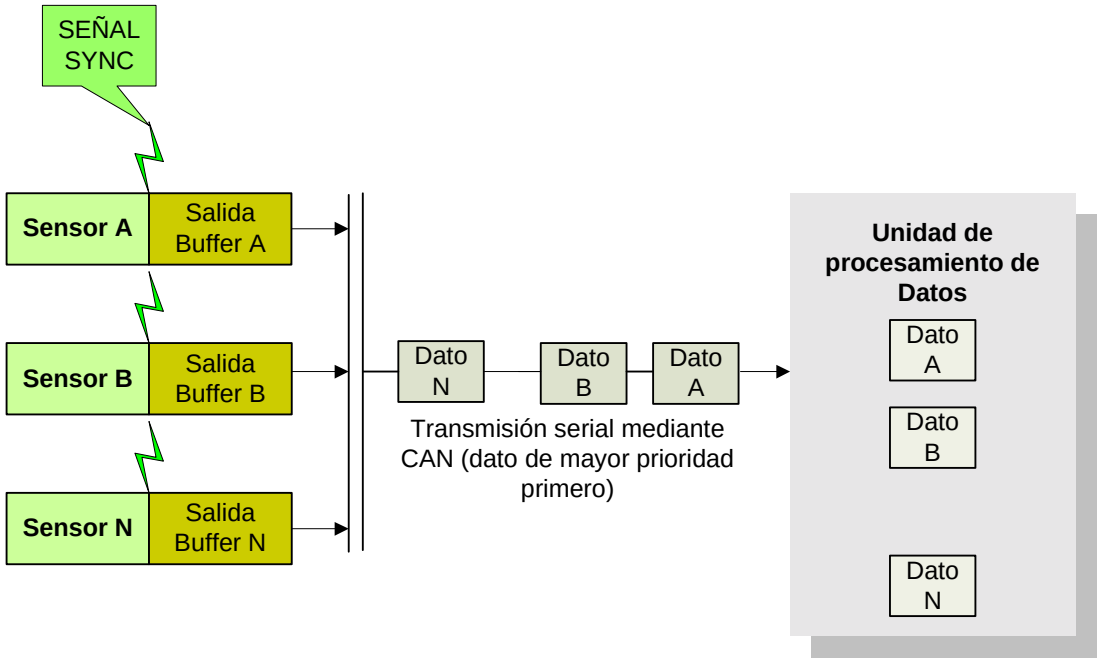


Figura. 5.11. Comunicación sincronizada para sensores

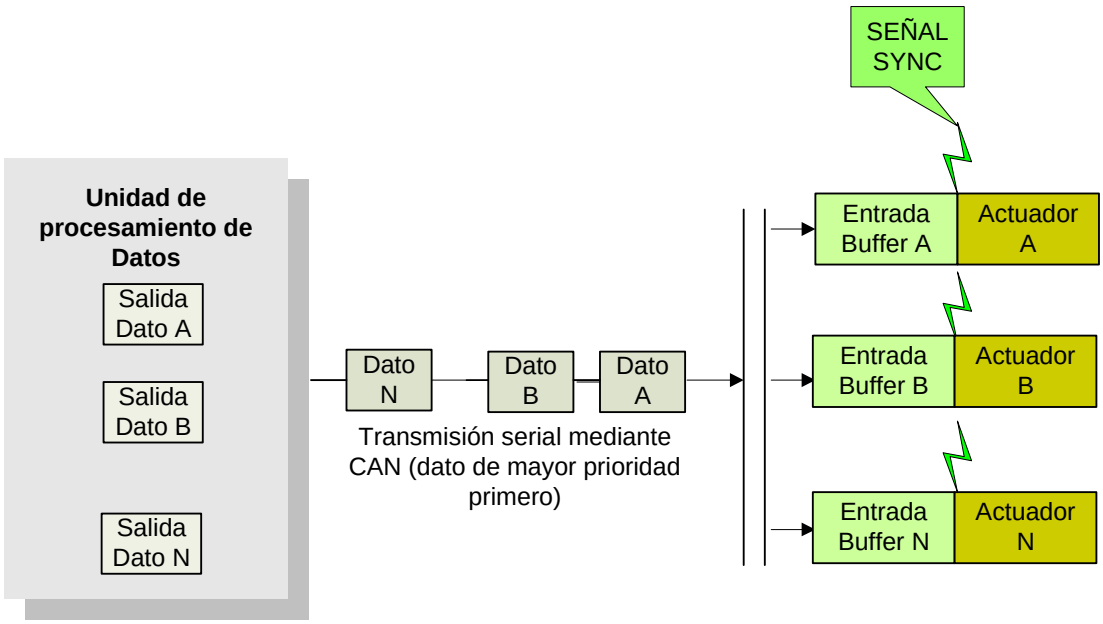


Figura. 5.12. Comunicación sincronizada para actuadores

5.4.6.3 Identificadores de mensajes PDO

Dentro de la red, un PDO no es más que un mensaje con un identificador de mensaje respectivo y hasta 8 bytes de datos. En CANopen casi todo es configurable, esto incluye cual identificador de mensaje (COB ID) es utilizado para cada PDO.

Existe implementado un uso por defecto de los identificadores de mensajes que determinan cual COB ID debe ser utilizado por defecto por un nodo.

La tabla 5.3. muestra el rango de identificadores de mensajes asignados a los PDOs:

IDENTIFICADOR CAN			
Desde	**Hasta**	**Objeto de comunicación**	**Comentario**
0h	-	Servicios NMT	Para Maestro NMT
80h	-	Mensaje SYNC	Para productor SYNC
81h	FFh	Mensaje de emergencia	Para nodos 1 – 127
100h	-	Mensaje de tiempo de captura	Para productor de tiempo de captura
181h	1FFh	1er TPDO	Para nodos 1 – 127
201h	27Fh	1er RPDO	Para nodos 1 – 127
281h	2FFh	2do TPDO	Para nodos 1 – 127
301h	37Fh	2do RPDO	Para nodos 1 – 127
381h	3FFh	3er TPDO	Para nodos 1 – 127
401h	47Fh	3er RPDO	Para nodos 1 – 127
481h	4FFh	4to TPDO	Para nodos 1 – 127
501h	57Fh	4to RPDO	Para nodos 1 – 127
581h	5FFh	SDO transmisor	Para nodos 1 – 127
601h	67Fh	SDO receptor	Para nodos 1 – 127
701h	77Fh	Control de error NMT	Para nodos 1 – 127

Tabla. 5.3. Rango de identificadores PDO

5.4.6.3 Parámetros de comunicación RPDO

En el diccionario de objetos, el área de índices desde 1400h hasta 15FFh es reservada para los parámetros de comunicación RXPDO, es decir que se puede obtener un máximo de 512 RPDOs configurados en el diccionario de objetos de un nodo CANopen. Los parámetros del primer RPDO (RPDO1) se localizan en el índice 1400h, los del segundo RPDO (RPDO2) se localizan en el 1401h y así sucesivamente. Los parámetros para cada RPDO son accesibles mediante subíndices como se muestra en la tabla 5.4. a continuación:

Subíndice	Nombre	Tipo de dato
0	Número de entradas	UNSIGNED8
1	COB ID	UNSIGNED32
2	Tipo de transmisión	UNSIGNED8
3	Tiempo de Inhibición	UNSIGNED16
4	Reservado	UNSIGNED8
5	Temporizador de eventos	UNSIGNED16

Tabla. 5.4. Parámetros RPDO

El número de entradas de un RPDO puede ser 5 si se utiliza el temporizador de eventos (event timer) pero la configuración más popular no la utiliza por lo que el número de entradas suele ser 2.

El COB ID o identificador de objeto de conexión es el identificador del mensaje CAN utilizado por el RPDO. Este parámetro determina cual mensaje CAN es recibido e interpretado como un RPDO.

El tipo de transmisión determina si el RPDO va a ser procesado inmediatamente después de la transmisión o si el nodo necesita esperar una señal de sincronización (SYNC) antes de procesar el dato recibido.

El tiempo de inhibición no es utilizado por los RPDOs y si se lo implementa debe tener el valor de cero.

5.4.6.4 Parámetros de comunicación TPDO

De igual forma que los RPDOS, en el diccionario de objetos existe un área de índices destinada a los parámetros de comunicación de los TPDOs y va desde 1800h hasta 19FFh, es decir que se puede obtener un máximo de 512 TPDOs configurados en el diccionario de objetos de un nodo CANopen. Los parámetros del primer TPDO (TPDO1) se localizan en el índice 1800h, los del segundo TPDO (TPDO2) se localizan en el 1801h y así sucesivamente.

Los parámetros para cada TPDO son accesibles mediante subíndices como se muestra en la tabla 5.5. a continuación:

Subíndice	Nombre	Tipo de dato
0	Número de entradas	UNSIGNED8
1	COB ID	UNSIGNED32
2	Tipo de transmisión	UNSIGNED8
3	Tiempo de Inhibición	UNSIGNED16
4	Reservado	UNSIGNED8
5	Temporizador de eventos	UNSIGNED16

Tabla. 5.5. Parámetros RPDO

El número de entradas de un TPDO es de 5. Las entradas de 0 hasta 2 son obligatorias mientras que la 3 y 5 son opcionales.

El COB ID al igual que en los RPDOs indican el identificador del mensaje CAN utilizado cuando se transmite un TPDO.

El tipo de transmisión selecciona la forma en que se va a transmitir el TPDO (event driven, time driven, etc).

Para una transmisión de tipo “COS” o cambio de estado (change of state), en la cual se producen cambios de estado constantes, debe haber una transmisión continua de TPDOs. En ese caso, el tiempo de inhibición especifica el periodo de tiempo que debe pasar antes de que un TPDO sea retransmitido nuevamente. El mínimo tiempo entre dos transmisiones de un TPDO se da en múltiplos de 100 microsegundos.

Para una transmisión “time driven”, el temporizador de eventos especifica el período de tiempo entre cada transmisión. Dicho tiempo se especifica en milisegundos. Se puede utilizar una combinación del tiempo de inhibición y el tiempo de eventos para lograr una aplicación orientada tanto a cambios de estado como a eventos.

5.4.7 Tipos de Datos CANopen

Como se vio en la tabla 5.1. los tipos de datos están colocados en el rango de entradas 0001h – 0FFFh pero a su vez pueden ser clasificadas por tipos de datos predefinidos y definidos por el fabricante así como por tipos de datos estándar y complejos. La tabla 5.6. muestra dicha clasificación con sus rangos de entradas:

Rango de Índices	Descripción
0001h – 001Fh	Tipo de datos estándar
0020h – 0023h	Tipo de datos complejos pre-definidos
0024h – 003Fh	Reservadas
0040h – 005Fh	Tipo de datos complejos (fabricante)
0060h – 007Fh	Tipo de datos estándar (device profile)
0080h – 009Fh	Tipo de datos complejos (device profile)
00A0h – 025Fh	Tipo de datos módulos de dispositivo múltiple
0260h – 0FFFh	Reservadas

Tabla. 5.6. Almacenamiento de tipos de variables en el diccionario de objetos

Los tipos de datos de módulos de dispositivos múltiples almacenan tipos de datos tanto estándar como complejos cuando se está utilizando más de un perfil de dispositivos.

5.4.7.1 Tipos de datos estándar

La tabla 5.7 muestra los tipos de datos estándar junto con sus descripciones y direcciones donde están definidas.

Tipo de dato estándar	Descripción	Almacenamiento en índice OD
Bolean	Un bit (0 o 1) indicador de verdadero o falso	0001h
Integer8	Entero con signo (8 bits)	0002h
Integer16	Entero con signo (16 bits)	0003h
Integer24	Entero con signo (24 bits)	0010h
Integer32	Entero con signo (32 bits)	0004h
Integer40	Entero con signo (40 bits)	0012h
Integer48	Entero con signo (48 bits)	0013h
Integer56	Entero con signo (56 bits)	0014h
Integer64	Entero con signo (64 bits)	0015h
Unsigned8	Entero sin signo (8 bits)	0005h
Unsigned16	Entero sin signo (16 bits)	0006h
Unsigned24	Entero sin signo (24 bits)	0016h
Unsigned32	Entero sin signo (32 bits)	0007h
Unsigned40	Entero sin signo (40 bits)	0018h
Unsigned48	Entero sin signo (48 bits)	0019h
Unsigned56	Entero sin signo (56 bits)	001Ah
Unsigned64	Entero sin signo (64 bits)	001Bh
Real32	Numero flotante simple de precisión. 32 bits	0008h
Real64	Numero flotante doble de precisión. 64 bits	0011h

Visible_String	Cadena de texto con caracteres ASCII imprimibles	0009h
Octet_String	Arreglo de enteros sin signo de 8 bits	000Ah
Unicode_String	Arreglo de enteros sin signo de 16 bits	000Bh
Time_of_day	Hora y día	000Ch
Time_Difference		000Dh
Domain	Bloque de datos	000Fh

Tabla. 5.7. Tipos de dato estándar

El tipo de dato Domain es un bloque de datos de una aflicción específica que puede tener cualquier longitud deseada.

Ejemplo:

Si se desea almacenar el valor actual de temperatura en el diccionario de objetos se lo puede hacer en la dirección 2000h utilizando un tipo de dato REAL32 entonces:

Índice 2000h Subíndice 00hTipo de dato REAL32.

5.4.7.2 Tipos de datos complejos

Los tipos de datos complejos son aquellos que contienen uno o más tipos de datos estándar agrupados, permitiendo que se construyan juegos de datos. Generalmente se utilizan para describir entradas que contienen subentradas con diferentes tipos de datos y se utilizan frecuentemente.

Existen cuatro tipos de datos complejos definidos en las especificaciones CANopen y se muestran en la tabla 5.8.

Tipo de dato	Descripción	Índice en el OD
PDO_COMUNICATION_PARAMETER	Graba los parámetros de comunicación de un PDO	0020h
PDO_MAPPING	Graba los parámetros de mapeo de un PDO	0021h
SDO_PARAMETER	Graba los parámetros de comunicación de un SDO	0022h
IDENTITY	Graba los parámetros de identidad	0023h

Tabla. 5.8. Tipos de dato estándar

Ejemplo:

Si se desea almacenar los detalles de un mensaje de error se puede necesitar el número de error y el texto del mensaje de error por lo que se necesitan dos variables una de tipo UNSIGNED16 y una de tipo VISIBLE_STRING. En el OD podemos asignar por ejemplo las direcciones 2000h – 200Fh para mensajes de error es decir para almacenar 16 mensajes de error entonces:

Índice 2000h

Subíndice 00h Almacena el valor 2 (número de subentradas)

Subíndice 00hTipo de dato UNSIGNED16

Subíndice 00hTipo de dato VISIBLE_STRING

5.4.8 Entradas de comunicación

Las entradas de comunicación de un DO describen la mayoría de aspectos relacionados a la comunicación CANopen utilizada por un nodo. Muchas de las entradas pueden ser escritas permitiendo la configuración de un nodo por otros nodos de la red. Las entradas

ocupan un rango de índices de 1000h – 1FFFh en el OD. La tabla 5.9. muestra todas las entradas de comunicación con sus respectivos índices:

INDICE	NOMBRE
1000h	Tipo de dispositivo
1001h	Registro de error
1002h	Registro de estado de fabricante
1003h	Campo de error predefinido
1005h	COB ID SYNC
1006h	Periodo del ciclo de comunicación
1007h	Longitud de la ventana síncrona
1008h	Nombre del dispositivo (fabricante)
1009h	Versión de hardware (fabricante)
100Ah	Versión de software (fabricante)
100Ch	Guard Time
100Dh	Factor de tiempo de vida
1010h	Parámetros de almacenamiento
1011h	Parámetros de restauración por defecto
1012h	COB ID Time
1013h	Tiempo de captura
1014h	COB ID EMCY
1015h	Tiempo de Inhibición (EMCY)
1016h	Tiempo de latencia (consumidor)
1017h	Tiempo de latencia (productor)
1018h	Objeto de identidad
1200h – 127Fh	Parámetros SDO servidor
1280h – 12FFh	Parámetros SDO Cliente
1400h – 15FFh	Parámetros de comunicación RxPDO
1600h – 17FFh	Parámetros de mapeo RxPDO
1800h – 19FFh	Parámetros de comunicación TxPDO
1A00h – 1BFFh	Parámetros de mapeo TxPDO

Tabla. 5.9. Entradas de comunicación

5.4.8.1 Entradas Obligatorias:

Las entradas obligatorias son las que deben ser implementadas por un nodo necesariamente para que compagine con CANopen y son las siguientes:

Device Type (1000h)

Es un valor de 32 bits que contiene en una forma limitada algunas características del nodo. Por ejemplo si el nodo es un módulo digital de entrada/salida y si están implementadas.

Registro de Error (1001h)

Es un valor de 8 bits que puede indicar si varios errores genéricos han ocurrido en el nodo. Ejemplo: error de comunicación, error de temperatura, etc.

Guard Time (100Ch)

Es un valor de 16 bits que indica con que frecuencia, el requerimiento de guardia de nodo, es transmitido por el maestro o recibido por el nodo. Este mecanismo permite conocer a los nodos si otro nodo específico continúa con vida en la red y está habilitado para comunicarse. Esta entrada se debe utilizar si no se está utilizando la entrada *heartbeat time* (tiempo de latencia).

Factor de tiempo de vida (100Dh)

Es un valor de 8 bits que trabaja con el tiempo de guardia (guard time). Especifica cuantos múltiplos de dicho tiempo deben pasar dentro de una transmisión o recepción antes de que se genere una condición de error. Esta entrada se debe utilizar si no se está utilizando la entrada *heartbeat time* (tiempo de latencia).

Producer heartbeat time (productor de tiempo de latencia (1017h)

Si no se utiliza una guardia de nodos, se debe implementar los *heartbeats*. Esta entrada especifica que tan a menudo el nodo debe transmitir mensajes de latencia. Se configura en cero para deshabilitar la entrada en caso de que se esté utilizando el tiempo de guardia.

Objeto de identidad (1018h)

Esta entrada provee información de identificación del nodo. Debe contener al menos el identificador del vendedor (ID vendor) CAN. También puede contener el código del producto que lo identifica, el número de revisión y el número de serie.

Indice	Subíndice	Tipo	Descripción
1000h	0	Unsigned32	Tipo de dispositivo
1001h	0	Unsigned8	Registro de error
1017h	0	Unsigned16	Tiempo de latencia
1018h			Objeto de identidad
	0	Unsigned8	4 (número de subentradas)
	1	Unsigned32	ID vendedor
	2	Unsigned32	Código de producto
	3	Unsigned32	Número de revisión
	4	Unsigned32	Numero de serie

Figura. 5.13. Entradas del diccionario de Objetos (OD) obligatorias

5.4.9 Entradas específicas de fabricante

Esta sección del DO utiliza el rango de índices 2000h – 5FFFh. Si se requiere una aplicación con almacenamiento de datos o configuración de operaciones que están fuera de cualquier estándar CANopen, se deben ubicar en ésta sección.

Ejemplo:

Un nodo puede realizar las funciones de un reloj de tiempo real. Se puede tener disponible el tiempo actual en una de las entradas del OD para que otros nodos de la red lo puedan leer. Esta aplicación se puede obtener definiendo los siguientes parámetros en la sección de entradas específicas de fabricante:

Índice 2000h

Subíndice 00h Almacena el valor 3 (número de subentradas)

Subíndice 01h Horas (UNSIGNED16)

Subíndice 02h Minutos (UNSIGNED8)

Subíndice 03h Segundos (UNSIGNED8)

5.4.10 Parámetros del perfil de dispositivos

Una vez que un nodo es implementado, el diseñador del nodo debe especificar que servicios de comunicaciones va a utilizar y cómo lo va a hacer. La especificación CANopen CIADS301 provee una variedad de estos servicios de comunicaciones. Un perfil de dispositivos especifica variables de datos de procesos que conoce un nodo, su configuración por defecto y sus parámetros de comunicación.

Existen perfiles propietarios como CIA (CAN IN AUTOMATION) que estandarizó perfiles de dispositivos es decir tipos de nodos específicos como módulos de entrada/salida genéricos. Las especificaciones son publicadas para varios tipos de dispositivos para poder implementarlos, ellos utilizan las entradas del OD localizadas en la sección de Parámetros de Perfil de Dispositivos.

Ejemplo:

En el perfil de dispositivos CIA DS 401 para módulos genéricos de entrada/salida; la entrada del OD 6000h permite hasta 2032 entradas digitales a ser leídas.

Índice 6000h

Subíndice 00h 1 -254 (número de subentradas)

Subíndice 01h Lectura entradas 1 – 8

Subíndice 02h Lectura entradas 9 – 16

Subíndice 03h Lectura entradas 17 – 24

Etc.

5.4.11 Lectura de especificaciones CANopen

La especificación CANopen CIADS301 contiene términos que se deben entender adecuadamente para el correcto funcionamiento de las redes, entre ellos los siguientes:

1) Index (índice) – El índice en el diccionario de objetos

2) Object or Object Code (código de objeto) – El tipo de objeto

NULL – No existe campo de datos

DOMAIN – Una variable con largas cantidades de datos

DEFTYPE – Define un tipo de datos estándar

DFSTRUCT – Define un tipo de datos complejo

VAR – Un valor simple

ARRAY – Una entrada con más de un subíndice, con cada subíndice (excepto 00h) teniendo el mismo tipo de dato

RECORD – Una entrada con más de un subíndice, con cada subíndice (excepto 00h) teniendo diferentes tipos de dato

3) Name (Nombre) – El nombre de la entrada

4) Type or Data Type (tipo de dato) – Tipo de dato

5) Acc. or Access Atributes (atributos de acceso) – Atributos de lectura y/o escritura

RW – Acceso de lectura y escritura

WO – Solo escritura

RO – Solo lectura

CONST – Solo lectura, dato constante

6) M/O or Category (categoría) – Indica si la entrada es obligatoria u opcional

Obligatory – Debe ser implementada

Opcional – Puede ser implementada si se desea

Condicional – Debe ser implementada en caso de que se den ciertas características o se implementen otras ciertas entradas.

5.4.12 Administración de red (NMT – Network management)

Cada nodo CANopen esclavo está en capacidad de encontrarse en diferentes estados operacionales. La figura 5.14. muestra dichos estados. Algunas transiciones se dan automáticamente por el nodo esclavo pero otras necesitan la recepción de un mensaje NMT Maestro. Dicho mensaje contiene el nuevo estado en el que debe ingresar el nodo y puede ser enviado ya sea a un nodo en especial o a todos los nodos de la red simultáneamente.

Al encender el sistema un nodo esclavo CANopen empieza en un estado llamado *Power-ON Reset* y continúa hacia un estado de inicialización (initialization). Dicho estado inicializa la aplicación entera así como las interfaces CAN/CANopen. Al final de la inicialización, el nodo trata de transmitir un mensaje de *boot-up* (inicio). Tan pronto como se ha transmitido el mensaje de inicio satisfactoriamente, el nodo cambia a un estado Pre-operacional.

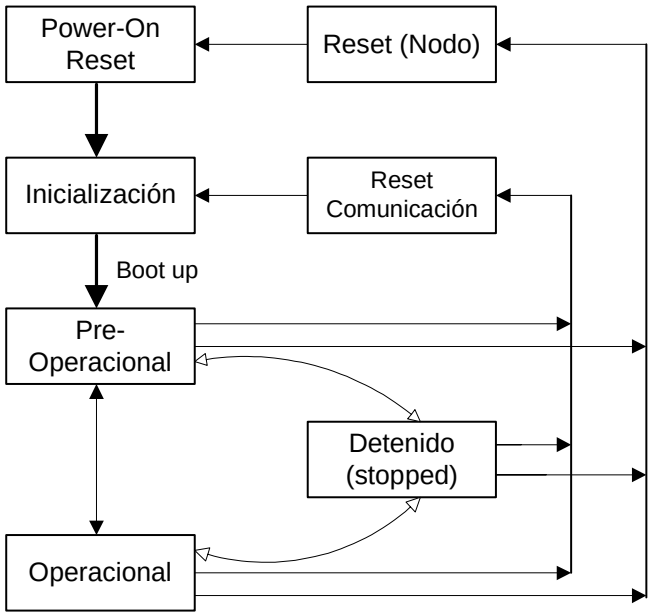


Figura. 5.14. Estados operacionales NMT

Utilizando el mensaje NMT Maestro, el nodo maestro puede colocar a los nodos esclavos entre tres estados principales: Pre-operacional, Operacional y Detenido y además es capaz de enviar requerimientos de dos tipos de reset. Cuando un nodo esclavo recibe la

orden de un reset de comunicación, el nodo resetea todas las interfaces de comunicación CAN/CANopen. Cuando un nodo esclavo recibe la orden de un reset de nodo, se resetean todos los periféricos y software relacionado al nodo.

La diferencia principal entre los estados NMT es que no en todos los estados se puede acceder a todos los tipos de comunicación CANopen, la tabla 5.10. muestra el tipo de comunicación que puede desarrollar un nodo en los diferentes estados.

	Inicialización	**Pre-operacional**	**Operacional**	**Detenido**
Boot-up	*			
SDO		*	*	
Emergencia		*	*	
SYNC/TIME		*	*	
Guardia de nodo		*	*	*
PDO			*	

Tabla. 5.10. Tipo de comunicaciones de los estados NMT

En el estado de inicialización, un nodo solo puede producir un mensaje de boot-up y no consume ningún tipo de mensaje.

En el estado pre-operacional, un nodo participa activamente en todas las comunicaciones relacionadas con SDOs, Emergencias, Tiempos de captura y guardia de nodo.

Existe una sola diferencia entre el estado pre-operacional y el operacional. El estado operacional incluye las comunicaciones de tipo PDO y solo en este estado el nodo se encuentra en toda su capacidad de ejecutar todas las funciones de entrada o salida para las que fue diseñado.

En un estado “Detenido”, un nodo detiene todas sus comunicaciones excepto un mínimo de servicios NMT.

5.4.12.1 Guardia de nodo y Heartbeat (latido)

Un nodo CANopen debe implementar los servicios ya sea de la guardia de nodo o del Heartbeat (latido). En la actualidad se recomienda utilizar el servicio de heartbeat en lugar de la guardia de nodo (node guarding) ya que consume menor ancho de banda y es más flexible y seguro.

En el servicio de Guardia de nodo, el correcto funcionamiento de los nodos esclavos es responsabilidad del nodo maestro. Es decir que si un nodo esclavo no responde dentro de un intervalo de tiempo específico, el maestro asume que el nodo esclavo perdió la información y toma medidas apropiadas como la reinicialización de sistema. Además todos los nodos realizan un monitoreo de los mensajes NMT enviados por el maestro por lo que pueden recibir una indicación de que se ha perdido la conexión con el maestro.

Las opciones de monitoreo se hacen más flexibles utilizando el servicio de latido de nodo. En este método, cada nodo esclavo transmite por sí mismo un mensaje CAN (heartbeat) de 1 byte de longitud que contiene el estado NMT actual del nodo. El tiempo que transcurre entre cada mensaje de latido CAN es configurable en milisegundos. (entrada [1017h,00h] UNSIGNED16). En la figura 5.15. se muestra dos mensajes de latido producidos constantemente por dos nodos esclavos cada uno con su respectivo tiempo.

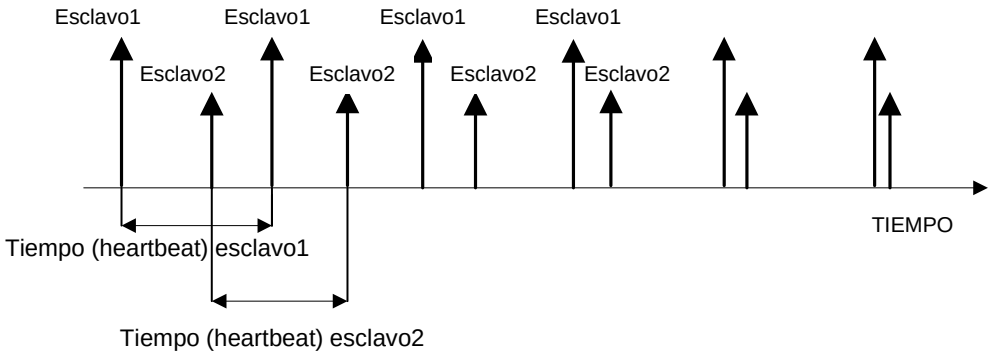


Figura. 5.15. Servicio Heartbeat

Cada nodo puede decidir cuales latidos desea monitorear. Una práctica común es que un nodo monitoree los latidos de los nodos que participan directamente en la comunicación. Esto permite que un nodo (productor) que transmite un PDO escuche los latidos de todos los nodos consumidores del mensaje para asegurarse que ellos continúan en estado operacional.

Otro aspecto muy importante es la seguridad que brinda el método ya que ningún nodo se vuelve esencial para escuchar los latidos en el sistema. En el método de Guardia de nodo el nodo maestro es esencial y si este falla o se desconecta, todos los nodos se verán afectados.

5.4.12.2 Emergencias (EMCY)

Cada nodo esclavo CANopen tiene asignado un mensaje de emergencia referido normalmente como EMCY. El identificador CAN utilizado para emergencias resulta de la suma entre 80h y el ID del nodo CANopen (1 - 127). El nodo 5 utilizará el identificador 85h para transmitir mensajes de emergencias.

Un mensaje de emergencia siempre contiene 8 bytes de datos, dentro de los cuales, los 2 primeros bytes son utilizados para el código de error CANopen (códigos definidos para voltajes, corrientes, temperaturas, hardware, software, sobrecargas, etc). El tercer byte contiene una copia del registro de error. Los 5 bytes restantes están disponibles para códigos de error específicos del diseñador.

En general las emergencias son reportadas una sola vez y se anulan en el momento en que se recibe otro mensaje de emergencia. Por ejemplo si un nodo reporta un mensaje de emergencia debido a un exceso de temperatura en cierto proceso, el mensaje se transmite una sola vez y continua latente hasta que el nodo envíe otro mensaje de emergencia indicando que la temperatura volvió a su rango normal.

5.4.13 Ejercicios

1) Qué entradas del OD del nodo 5 deben implementarse si el nodo necesita:

- **Producir un latido de 250ms**
- **Monitorear el latido del nodo 7 (producido cada 500ms)**
- **Monitorear el latido del nodo 9 (producido cada 1000ms)**

Resumen de entradas del OD que controlan el mecanismo de heartbeat

[1016h, 00h] Tiempo de latido de consumidor, número de elementos del arreglo

[1016h, xxh] Entrada de 32 bits para cada latido monitoreado

[1017h, 00h] Tiempo de latido de productor en milisegundos

Solución:

Escribir el valor 250d en la entrada del [1017h, 00h] del DO del nodo 5

Para monitorear el latido de un nodo, el tiempo del consumidor debe ser mayor al tiempo del productor del latido. Un valor razonable es de 1.5 o 2 veces el tiempo del productor.

Escribir el valor de 2 (2 latidos a monitorear) en la entrada [1016, 00h]

Escribir el valor 750d (1.5 veces 500) en la entrada [1016, 01h]

Escribir el valor 1500d (1.5 veces 1000) en la entrada [1016, 02h]

2) El nodo 5 necesita ser configurado para escuchar directamente los TPDO1 y TPDO2 transmitidos por defecto del nodo 6. El RPDO1 y RPDO2 del nodo 5 debe ser utilizado para recibir el TPDO1 y TPDO2 del nodo 6.

Resumen de las entradas del diccionario utilizadas para controlar el enlace PDO con sus respectivos identificadores por defecto

[1400h, 01h] COB ID del RPDO1 (por defecto 200h + ID del nodo)

[1401h, 01h] COB ID del RPDO2 (por defecto 300h + ID del nodo)

[1402h, 01h] COB ID del RPDO3 (por defecto 400h + ID del nodo)

[1403h, 01h] COB ID del RPDO4 (por defecto 500h + ID del nodo)

[1800h, 01h] COB ID del TPDO1 (por defecto 180h + ID del nodo)

[1801h, 01h] COB ID del TPDO2 (por defecto 280h + ID del nodo)

[1802h, 01h] COB ID del TPDO3 (por defecto 380h + ID del nodo)

[1803h, 01h] COB ID del TPDO4 (por defecto 480h + ID del nodo)

Solución:

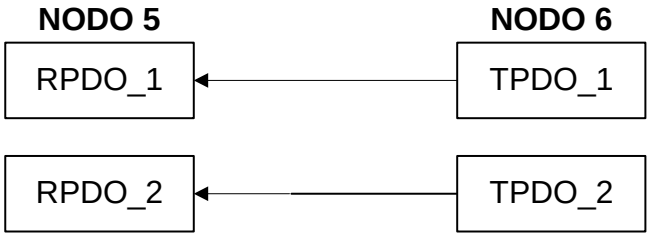


Figura. 5.16. Ejemplo 2

El identificador CAN utilizado por defecto por el nodo 6 para su TPDO0 y TPDO1 es 186h y 286h respectivamente. (180h y 280h + ID 6)

Escribir el valor 186h (390d) en la entrada del DO [1400h, 01h] del nodo 5

Escribir el valor 286h (646d) en la entrada del DO [1401h, 01h] del nodo 5

3) El nodo 42 (2Ah) transmite 2 valores análogos de 16 bits en su TPDO2, el nodo 45 (2Dh) transmite 2 valores análogos de 16 en su TPDO3. Realizar las siguientes configuraciones:

- **El nodo 31 (1Fh) debe recibir en su RPDO2 el TPDO2 del nodo 2Ah y debe recibir en su RPDO3 el TPDO3 del nodo 2Dh**

- **El nodo 1Fh tiene un arreglo de 4 valores [6411, 01h-04h]. Configurar el mapeo del RPDO2 y RPDO3 para que los valores de RPDO2 vayan en [6411, 01h-02h] y los valores de RPDO3 vayan en [6411, 03h-04h]**

Resumen de las entradas del diccionario utilizadas para controlar el mapeo PDO

[1600h, 00h] Mapeo RPDO1, número de entradas mapeadas

[1600h, xxh] Índice, subíndice y longitud (en bits) de la entrada mapeada

[1601h, 00h] Mapeo RPDO2, número de entradas mapeadas

[1601h, xxh] Índice, subíndice y longitud (en bits) de la entrada mapeada

[1A00h, 00h] Mapeo TPDO1, número de entradas mapeadas

[1A00h, xxh] Índice, subíndice y longitud (en bits) de la entrada mapeada

[1A01h, 00h] Mapeo TPDO2, número de entradas mapeadas

[1A01h, xxh] Índice, subíndice y longitud (en bits) de la entrada mapeada

Solución:

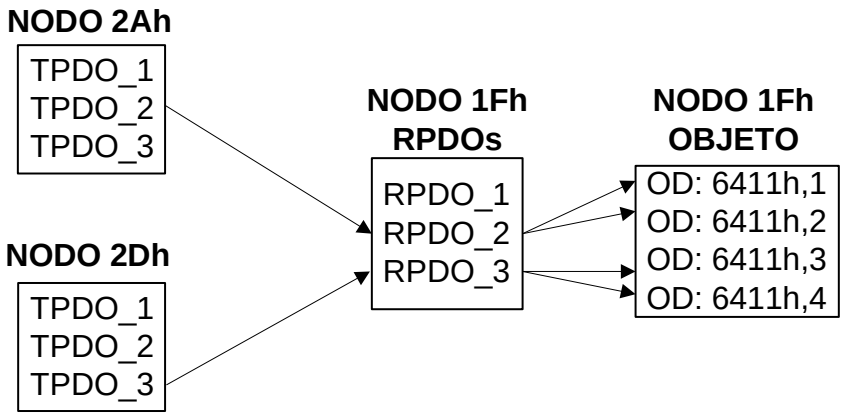


Figura. 5.17. Ejemplo 3

Para cambiar los parámetros PDO, un PDO normalmente necesita ser deshabilitado. Esto puede ser logrado mediante la configuración del bit 31 en el COB ID.

Primera solución:

- El valor del identificador por defecto utilizado por el nodo 2Ah para su TPDO2 es 2Aah (280h + ID 2Ah). El valor por defecto del identificador para usar el TPDO3 del nodo 2Dh es 3ADh (380h + ID 2Dh).

Para configurar el nodo 1Fh para recibir los TPDOs:

Escribir el valor 2AAh en la entrada [1401h, 01h] del DO del nodo 1Fh

Escribir el valor 3ADh en la entrada [1402h, 01h] del DO del nodo 1Fh

Para configurar el mapeo:

Escribir el valor `0’ en la entrada [1601, 00h] del DO del nodo 1Fh (informa al nodo que el mapeo será cambiado)

Escribir el valor de 2 en la entrada [1601h, 00h] del DO del nodo 1Fh (el número total de entradas mapeadas es de 2)

Escribir el valor 64110110h en la entrada [1601h, 01h] del DO del nodo 1Fh (primera entrada mapeada para el RPDO2, Índice 6411, Subíndice 01h, longitud 10h para UNSIGNED16)

Escribir el valor 64110210h en la entrada [1601h, 02h] del DO del nodo 1Fh (segunda entrada mapeada para el RPDO2, Índice 6411, Subíndice 02h, longitud 10h para UNSIGNED16)

Escribir el valor `0’ en la entrada [1602, 00h] del DO del nodo 1Fh (informa al nodo que el mapeo será cambiado)

Escribir el valor de 2 en la entrada [1602h, 00h] del DO del nodo 1Fh (el número total de entradas mapeadas es de 2)

Escribir el valor 64110310h en la entrada [1602h, 01h] del DO del nodo 1Fh (primera entrada mapeada para el RPDO2, Índice 6411, Subíndice 03h, longitud 10h para UNSIGNED16)

Escribir el valor 64110410h en la entrada [1602h, 02h] del DO del nodo 1Fh (segunda entrada mapeada para el RPDO2, Índice 6411, Subíndice 04h, longitud 10h para UNSIGNED16)

5.5 DEVICENET

El protocolo DeviceNet fue desarrollado por Rockwell Automation como un bus de campo abierto basado en el protocolo CAN y diseñado a nivel de sensores, actuadores y unidades de control asociadas. Es muy utilizado en Asia y Estados Unidos. En los últimos años también se han empezado a desarrollar muchas aplicaciones DeviceNet en Europa.

DeviceNet especifica características tanto de capa física como de capa de aplicación, dentro de la capa física define conectores, tipos de cables y longitudes de acuerdo a la velocidad de transmisión de datos y con respecto a la capa de aplicación trabaja mediante un sistema basado en un modelo de objetos.

DeviceNet puede trabajar con una cantidad hasta de 64 nodos dentro de la red a tres velocidades de transmisión: 125Kbps, 250 Kbps y 500 Kbps.

Los cables de una red DeviceNET conducen tanto las señales de la red como la alimentación de los dispositivos. El amplio rango de aplicaciones crearon la necesidad de disponer diversos tipos de cables según la aplicación y se han divido según la velocidad de transmisión en tres tipos como muestra la tabla 5.5. Todos los cables tienen cuatro hilos y una tierra excepto el cable llano que no dispone de tierra, y han sido diseñados para ambientes industriales.

Velocidad de Transmisión	**Longitud de Línea Principal**		
	Cable grueso	**Cable Delgado**	**Cable llano**
125 Kbps	500 mt	100 mt	420 mt
250Kbps	250 mt	100 mt	200 mt
500Kbps	100mt	100 mt	100 mt

Tabla. 5.11. Recomendaciones de longitud de cables DeviceNet

Como en otros protocolos, la función principal del protocolo DeviceNet es la de intercambiar datos entre dispositivos con unidades de control asociadas y al igual que en los protocolos basados en redes CAN se hace una distinción entre los mensajes de mayor prioridad y los de menor prioridad.

DeviceNet describe el intercambio de datos entre dispositivos de acuerdo al modelo de comunicaciones Productor-Consumidor y en base a los principios del protocolo CAN (broadcasting). Un nodo DeviceNet transmisor produce un dato en la red y el nodo receptor DeviceNet consume el dato desde la red.

5.5.1 Modelo de Objetos

DeviceNet describe todos los datos y funciones de un dispositivo utilizando un modelo de objetos. Este método permite que un dispositivo sea visto como una colección de objetos individuales, dichos objetos individuales solo pueden ser accesibles en la red mediante los objetos de conexión. En la figura 5.18. se muestra un resumen de los principales objetos definidos por DeviceNet.

Dentro de este contexto, un objeto representa una descripción abstracta de un componente dentro de un dispositivo. El objeto es determinado según sus datos o atributos (estado del objeto, número de serie del dispositivo, datos de proceso como temperatura, presión, etc), sus funciones o servicios (lectura o escritura) y su comportamiento dentro de la red (respuestas a eventos internos o externos).

Los datos y servicios de un objeto son direccionados de acuerdo al siguiente orden jerárquico:

- Dirección del dispositivo (MAC ID): Identificador de control de acceso al medio.

- Identificador de Clase (Class ID): Una clase define la descripción de todos los atributos, servicios y comportamientos de varios objetos del mismo tipo.

- Identificador de Instancia (Instante ID): Representa un objeto existente en un dispositivo que pertenece a cierta clase.

- Identificador de Atributo (Attribute ID): Identifica el proceso que representa un objeto como temperatura, presión, posición, etc.

- Código de servicio (Service Code): Identifica una lectura o escritura de un determinado atributo.

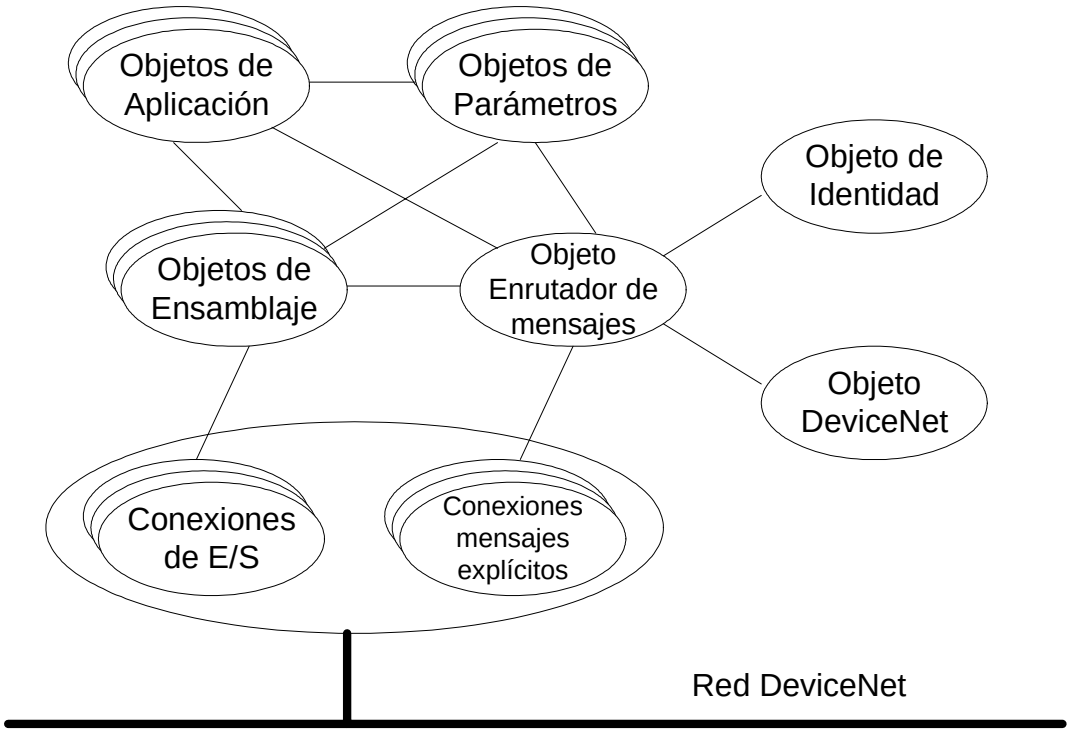


Figura. 5.18. Modelo de objetos DeviceNet

Las clases, instancias y atributos generalmente son definidas en DeviceNet por identificadores, cada identificador es un valor entero de 8 bits que permite hasta 256 identificadores disponibles. Los servicios en cambio se definen por un identificador o valor entero de 7 bits que permite hasta 128 identificadores disponibles.

Los identificadores de clases, instancias, atributos y servicios no son disponibles en su totalidad para el usuario. Algunos de ellos son reservados para futuras expansiones DeviceNet o para uso específico de vendedores.

DeviceNet tiene tres tipos generales de objetos:

1) Objetos de comunicación

2) Objetos de sistema

3) Objetos de aplicación

5.5.2 Objetos de comunicación

Estos objetos definen y administran el intercambio de mensajes dentro de la red DeviceNet.

Objetos DeviceNet

Los objetos DeviceNet (código de clase 03h) deben ser soportados por todos los dispositivos y definen la conexión física de los dispositivos a la red. Esto incluye la dirección del dispositivo (MAC ID) y la velocidad de transmisión.

Objetos Enrutadores de Mensajes

Los objetos enrutadores (código de clase 02h) también deben ser soportados por todos los dispositivos DeviceNet. Estos permiten el acceso a cada clase e instancia en el dispositivo mediante mensajes explícitos.

Objetos de conexión

Los objetos de conexión (código de clase 05h) definen:

- La ruta de conexión del dato mediante mensajes de e/s (clase, instancia, atributo).
- La longitud del dato a ser producido/consumido.
- El identificador CAN utilizado para la conexión.
- El tipo de activación o disparo (trigger) para e/s de mensajes:

  Activación de Aplicación: La aplicación decide cuando se transfiere el dato.

  Activación Cíclica: El dato es transferido en ciclos o intervalos de tiempo configurados.

  Activación por eventos: El dato es transferido tan pronto como se haya producido un cambio de estado (evento).

  Activación de Red: El nodo produce un dato y lo envía cuando hubo un requerimiento por parte de otro nodo.

5.5.3 Objetos de sistema

Los objetos de sistema definen datos o funciones generales que son útiles para dispositivos virtuales.

Objetos de identidad

Los objetos de identidad (código de clase 01h) contienen los atributos que identifican claramente a un nodo dentro de la red como son el ID de vendedor, tipo de dispositivo y código de producto.

Objetos de parámetros

Los objetos de parámetros (código de clase 0Fh) especifican una interfaz opcional disponible para toda la configuración de datos o parámetros del dispositivo. La característica principal de estos objetos es el gran número de atributos que habilita el usuario para una configuración conveniente del dispositivo utilizando una herramienta portátil. Durante este proceso se definen valores límites, divisores, factores y unidades físicas.

Algunos dispositivos DeviceNet proveen hojas electrónicas de especificaciones o EDS (electronic data sheet) que contienen la misma información en código ASCII.

5.5.4 Objetos de Aplicación específica

Objetos de aplicación

Los objetos de aplicación son definidos por el fabricante del dispositivo. Para aplicaciones de automatización típicas se encuentran disponibles varios objetos de aplicación simples como entradas/salidas discretas, entradas/salidas análogas o unidades de control de motores.

Objetos de ensamblaje

Los objetos de ensamblaje (código de clase 04h) permiten al usuario la opción de mapear datos de atributos o instancias diferentes de varias clases en un solo atributo o instancia de un objeto de ensamblaje. Este mapeo generalmente es utilizado para

entrada/salida de mensajes con el fín de maximizar el control del intercambio de datos en la red.

5.5.5 Uso de identificadores CAN

DeviceNet está basado en el protocolo CAN estándar por lo tanto utiliza 11 bits identificadores de mensajes. Mediante 11 bits identificadores se puede obtener hasta 2048 mensajes pero 6 bits son suficientes para identificar un dispositivo ya que la red DeviceNet está limitada a 64 dispositivos. Estos 6 identificadores son conocidos como MAC ID.

Dependiendo del modelo del identificador, se han divido en cuatro grupos de mensajes como muestra la tabla 5.12.:

ID de conexión o ID CAN (bits 10:0)											Utilizado por:
10	9	8	7	6	5	4	3	2	1	0	
0	ID Mensaje				ID MAC Fuente						Grupo de Mensajes 1
1	0	ID MAC						ID Mensaje			Grupo de Mensajes 2
1	1	ID Mensaje			ID MAC Fuente						Grupo de Mensajes 3
1	1	1	1	1	ID Mensaje						Grupo de Mensajes 4
1	1	1	1	1	1	1	x	x	x	x	Identificadores CAN Inválidos

Tabla. 5.12. Definición de grupos de mensajes

En DeviceNet el identificador CAN es conocido como ID de Conexión. Este comprende el identificador del grupo del mensaje, el identificador de mensaje y el ID MAC tanto fuente como destino. La importancia o prioridad del mensaje depende del ID Mensaje.

Grupo de mensajes 1

Dispone de 1024 identificadores CAN (10 bits) y dentro de este grupo el usuario puede manejar hasta 16 mensajes por dispositivo de acuerdo a su prioridad (ID Mensaje). Después del identificador de mensajes se transmite los identificadores MAC o dirección del dispositivo fuente.

Si dos dispositivos transmiten un mensaje al mismo tiempo, el que tenga mayor prioridad será el que gane el acceso al bus. Si dos dispositivos envían un mensaje con el mismo identificador entonces el nodo con menor identificador ganará el acceso al bus.

Grupo de mensajes 2

Dispone de 512 identificadores CAN (9 bits), en este grupo la prioridad de acceso al bus está definida primeramente por el dispositivo de destino (MAC ID) y después por el identificador del mensaje.

Grupo de mensajes 3

Tiene la misma estructura que el grupo de mensajes 1, con el mismo tipo de prioridad pero se resta un bit a la parte del identificador del mensaje.

Grupo de mensajes 4

Finalmente este grupo no incluye ningún direccionamiento ni del dispositivo fuente ni del dispositivo destino y la prioridad por lo tanto se relaciona únicamente al identificador de mensaje.

5.5.6 Modelo de Comunicaciones

DeviceNet utiliza un modelo de comunicaciones orientado a conexiones, esto significa que un dato no puede acceder a la red mientras no se establezca una conexión entre objetos.

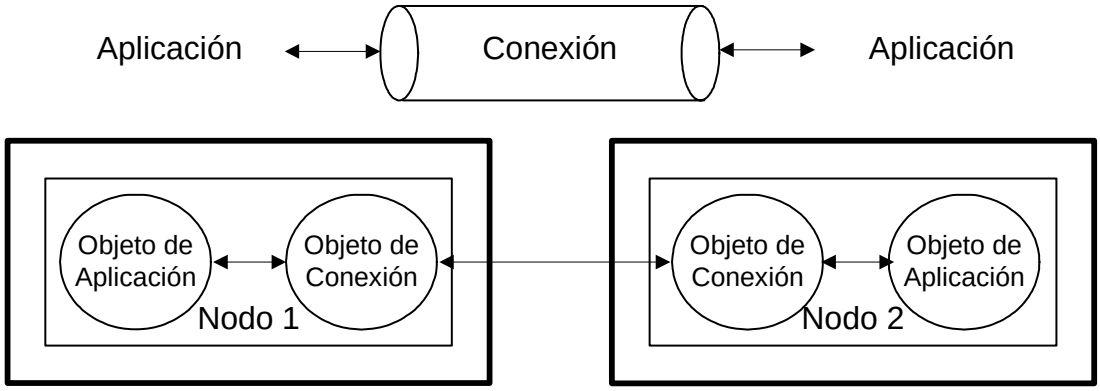


Figura. 5.19. Modelo de comunicaciones DeviceNet

Los mensajes a transmitirse en una red DeviceNet pueden ser de dos tipos: mensajes explícitos y mensajes de entrada/salida.

Mensajes explícitos

Los mensajes explícitos son utilizados para un intercambio general de datos entre dispositivos mediante la red. Datos de configuración de prioridad, datos de administración general y algunos datos de diagnóstico pueden ser transferidos mediante este sistema.

Este tipo de comunicación siempre se da punto a punto en un sistema cliente/servidor por lo cual, un requerimiento de un cliente debe ser reconocido mediante una respuesta del servidor.

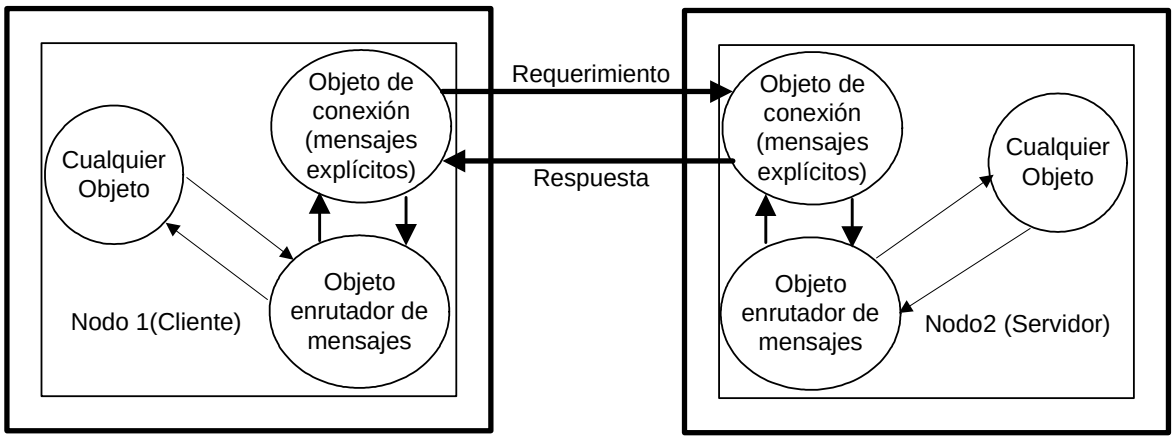


Figura. 5.20. Conexión de mensajes explícitos

Mensajes de entrada/salida

Los mensajes de entrada/salida se utilizan para el intercambio de datos de proceso y aplicaciones de mayor prioridad mediante la red CAN.

La comunicación mediante mensajes de entrada/salida se da mediante el modelo de comunicaciones productor/consumidor es decir que los datos de e/s siempre son transferidos desde una aplicación o nodo consumidor hacia uno o varios nodos consumidores.

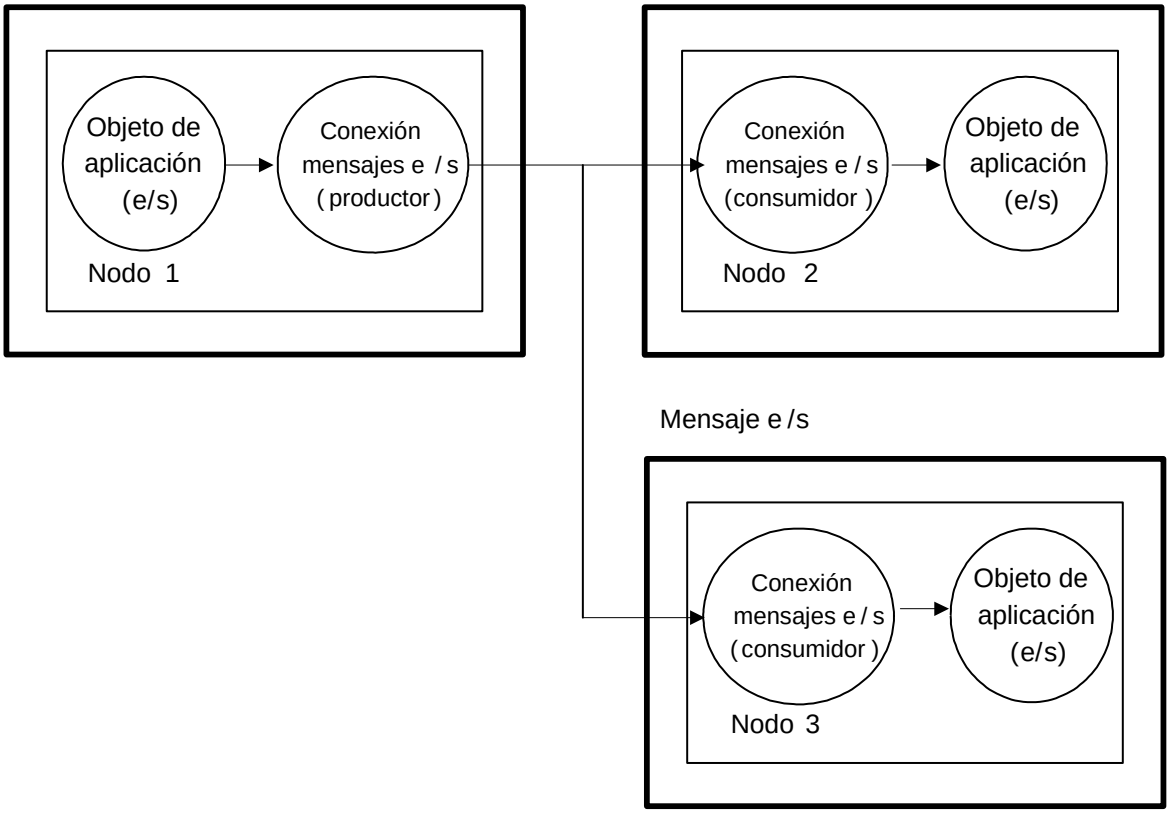


Figura. 5.21. Conexión de mensajes de entrada/salida

5.6 SAE J1939

En el sector de vehículos comerciales, los protocolos de comunicación serial estandarizados para la interacción entre unidades de control electrónicas y componentes de poder ha sido utilizado hace mucho tiempo. El protocolo principal desarrollado para este fin fue el J1708/J1587 inventado por SAE en Estados Unidos el cual trabajaba a bajas velocidades (9.6Kbps) y utilizaba la interfaz serial utilizada comúnmente en la mayoría de microcontroladores. Dicho protocolo fue la base para el SAE J1939 que trabaja bajo el protocolo CAN a mayores velocidades.

El uso de un protocolo de comunicaciones serial en el área vehicular trae varias ventajas como son:

- Los fabricantes de componentes deben implementar un solo componente.
- Los fabricantes de vehículos comerciales pueden utilizar componentes de diferentes proveedores.
- La interoperabilidad entre componentes individuales está asegurada, es decir que los componentes de varios fabricantes pueden operar juntos sin realizar ninguna modificación.

Mediante SAE J1939 es posible transmitir valores de medidas, datos de control, y datos de configuración. Además es posible leer o borrar datos de diagnósticos de componentes individuales y llevarlos a calibración.

En este protocolo no se define solo el tipo de transmisión, estructura de mensajes, segmentación, control de flujo, etc. Sino también se determina exactamente el contenido de cada uno de los mensajes que fluyen en la red.

En los protocolos de comunicación SAE J1587 (base para SAE J1939) se trabajaba bajo el concepto de direccionamiento de nodos. Para cumplir con este requerimiento y mantener la compatibilidad de SAE J1939 con las versiones anteriores, se describe la capa

de enlace de datos basada en la versión 2.0B del protocolo CAN en donde normalmente se utiliza solo el formato extendido y algunas veces el formato base en aplicaciones específicas.

Se utiliza el identificador de mensajes CAN extendido es decir los 29 bits identificadores de tal forma que una parte del identificador es utilizada para indicar la dirección del nodo fuente y el nodo de destino (direccionamiento de nodos) y otra parte del identificador se utiliza para determinar la prioridad del mensaje de acuerdo al protocolo CAN.

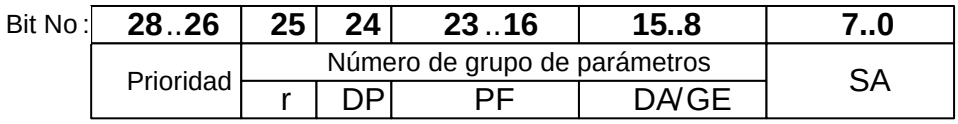
SAE J1939 opera a una velocidad de transmisión de 250 Kbps las cuales posibilitan una transmisión de 1850 mensajes por segundo aproximadamente. El número máximo de nodos es de 30 y la máxima longitud de bus es de 40 mts.

5.6.1 Estructura del mensaje

Básicamente un mensaje CAN consta de 29 bits identificadores, información sobre la longitud de los datos y los datos propiamente dichos (hasta 8 bytes).

El identificador CAN se divide en tres partes:

- Prioridad
- Número de grupo de parámetros (PGN)
- Dirección fuente del mensaje



r: reservado

DP: Pagina de datos

PF: Formato [PDU]

DA: Dirección de destino

GE: Extensión de grupo

SA: Dirección Fuente

Figura. 5.22. Estructura del identificador CAN de acuerdo a SAE J1939

5.6.1.1 Prioridad

Los tres bits más significativos del identificador CAN son designados para determinar la prioridad del mensaje con respecto al arbitraje de bus. Estos tres bits pueden colocarse entre 0 (alta prioridad) y 7 (baja prioridad). Dentro de la especificación, los mensajes de control son configurados en 3 y el resto de mensajes en 6. Se pueden asignar prioridades adicionales a mensajes específicos.

5.6.1.2 Número de grupo de parámetros

El grupo de parámetros especifica los siguientes componentes:

Bit reservado

Se dispone de un bit de reserva para futuras ampliaciones del protocolo y se lo debe colocar en cero.

Bit de página de datos (DP)

El bit DP saca la cantidad de campos de formatos PDU que vienen a continuación es decir que se pueden tener dos campos PDU.

Formato PDU (PF):

Mediante 8 bits de campo de formato PDU se especifican servicios de comunicación relacionado a la transmisión orientada ya sea a nodos o a mensajes.

Formato PDU 1: Transmisión orientada a nodos.

Formato PDU 2: Transmisión orientada a mensajes

Campo específico PDU (DA/GE):

El campo específico PDU se utiliza para designar la función de 8 bits.

Si el tipo de formato PDU es orientado a nodos, los 8 bits se utilizan para indicar la dirección de destino del mensaje (DA). En el lado receptor el filtraje de mensajes se realiza mediante la dirección incluida en el identificador. De esta manera se puede obtener una comunicación par a par entre dos nodos donde se involucran un solo nodo fuente y un solo nodo receptor. Además es posible enviar la dirección 2555 de tal forma que el mensaje sea enviado hacia todos los nodos de la red (broadcasting).

Si el formato PDU indica una transmisión orientada a mensajes, los 8 bits indicarán una extensión de grupo (GE) que es un cierto grupo de parámetros. El filtraje de mensajes se da en el lado receptor mediante este grupo de `parámetros incluidos en el identificador de tal forma que el mensaje puede ser recibido por uno o varios nodos interesados en él (multicasting).

5.6.1.3 Dirección fuente del mensaje

La dirección fuente del mensaje está representada por 8 bits e indica el nodo que ha producido el mensaje, todos los nodos tienen una única dirección por lo que no se podría dar colisiones incluso si dos nodos enviaran un mensaje de la misma prioridad ya que las direcciones serían diferentes.

5.6.1.4 Campo de Datos

Los datos de un mensaje asignados al Grupo de Parámetros (PGN) son transmitidos mediante un campo con una longitud máxima de 8 bytes (mensaje CAN). Si los datos de un Número de Grupo de Parámetros es mayor a 8 bytes, el bloque de datos es fragmentado, es decir transmitido en forma de diversos mensajes CAN. SAE J1939 permite una transmisión segmentada hasta de 1785 bytes.

5.6.2 Tipos de mensajes

Existen 5 tipos de mensajes dentro del Grupo de Parámetros según SAE J1939:

- Comandos
- Requerimientos
- Respuestas
- Reconocimiento
- Funciones

5.6.2.1 Comandos

Los comandos son utilizados para llevar a cabo una función en uno o más nodos de destino. Se puede utilizar el formato PDU1 (orientado a nodos) o el formato PDU2 (orientado a mensajes).

5.6.2.2 Requerimientos

Mediante mensajes de requerimientos específicos, es posible realizar peticiones de datos concretos a uno o más nodos remotos de la red. En el protocolo SAE J1939, una trama de

datos con un valor de formato PDU específico (234) en el identificador, implica un requerimiento de datos, es decir no trabaja con la trama remota del protocolo CAN.

El dato requerido está especificado dentro del Número de Grupo de Parámetros (PGN) en el campo de datos.

<table>
<tr><th colspan="5">Identificador (29 bits)</th><th>DLC</th><th colspan="3">Campo de Datos</th></tr>
<tr><td rowspan="2">Prioridad</td><td colspan="4">PGN</td><td rowspan="2">SA</td><td rowspan="2">3</td><td>d0</td><td>d1</td><td>d2</td></tr>
<tr><td>r</td><td>DP</td><td>PF</td><td>DA</td><td colspan="3">PGN Requerimiento</td></tr>
</table>

r: reservado
DP: Pagina de datos
PF: Formato PDU (234)
DA: Dirección de destino
SA: Dirección Fuente
DLC: Código de longitud de datos

Figura. 5.23. Formato de un requerimiento SAE J1939

Un mensaje de requerimiento puede ser global cuando DA = FFh (255) o puede ser hacia un nodo específico. El mensaje de respuesta debe ser enviado por el nodo diseccionado en el requerimiento.

5.6.2.3 Respuesta

Este tipo de mensaje se da en consecuencia o respuesta a un mensaje de requerimiento o comando.

5.6.2.4 Reconocimientos

Un mensaje de reconocimiento (ACK) se da por un nodo para advertir a todos los nodos de la red sobre la recepción de un requerimiento. El mensaje de reconocimiento siempre es

en forma de broadcast es decir se envía a todos los nodos de la red (DA=FFh) y tiene el valor de formato PDU 232. Un ACK es un reconocimiento positivo y el NACK es un reconocimiento negativo, la diferencia entre estos dos se da mediante un byte de control en el campo de datos 0=ACK, 1=NACK, 2= Acceso denegado por razones de seguridad.

<table>
<tr><th colspan="6">Identificador (29 bits)</th><th>DLC</th><th colspan="8">Campo de Datos</th></tr>
<tr><td rowspan="2">Prioridad</td><td colspan="4">PGN</td><td rowspan="2">SA</td><td rowspan="2">8</td><td>d0</td><td>d1</td><td>d2</td><td>d3</td><td>d4</td><td>d5</td><td>d6</td><td>d7</td></tr>
<tr><td>r</td><td>DP</td><td>PF</td><td>DA</td><td>Ctr</td><td>GFV</td><td>FFH</td><td>FFH</td><td>FFH</td><td colspan="3">PGN</td></tr>
</table>

ctrl. = 0: ACK

ctrl. = 1: NACK

ctrl. = 0: Acceso denegado

GFV: Valor de Función de Grupo

Figura. 5.24. Estructura de un mensaje de reconocimiento SAE J1939

Además del byte de control “ctrl” el mensaje de reconocimiento envía un byte con el PGN del mensaje reconocido y un byte de valor de función de grupo “GFV” mediante el cual se puede asignar un reconocimiento a un mensaje de funciones de grupo.

5.6.2.5 Funciones de grupo

El tipo de mensajes “Funciones de grupo” se utiliza para identificar funciones especiales. Estas incluyen mensajes propietarios, funciones de administración de red y funciones para la transmisión de bloques largos de datos (multi-paquetes). Para las funciones de grupo se asignan PGNs específicos. En este caso el PGN no indica el dato a ser transmitido sino la función o tipo de servicio.

Transmisiones fragmentadas

Los bloques de datos mayores a 8 bytes de longitud deben ser transferidos de una manera fragmentada. En el protocolo SAE J1939 se asegura que dicho proceso funcione en

su totalidad es decir que se divida el bloque total de datos a transmitirse en bloques más pequeños en forma de mensajes CAN. Cuando se han recibido todos los segmentos de datos, se reordena y se consigue el bloque total de datos inicial.

El protocolo soporta dos tipos de transmisión fragmentada:

- Transmisión orientada a nodos fragmentada.
- Transmisión de tipo broadcast fragmentada.

Transmisión fragmentada orientada a nodos

Este tipo de transmisión se basa en tres pasos principales:

1) Establecer la conexión
2) Transmitir los datos
3) Desconexión

Para establecer la conexión entre dos nodos, el nodo iniciador envía un mensaje RTS (Request to send), hacia el nodo de destino. El nodo de destino envía un mensaje CTS (Clear to send) si recibió el mensaje RTS. Si la conexión es fallida el nodo de destino envía un mensaje CA (Conection Abort).

Una vez que el nodo transmisor recibió el mensaje CTS, empieza a enviar los segmentos de datos con mensajes DT (Data Segment). Cada tres mensajes DT, el nodo receptor envía un nuevo mensaje CTS. Al finalizar la transmisión, se debe enviar un mensaje EOM (End of Message Acknowledge) como reconocimiento de fin de transmisión de paquetes.

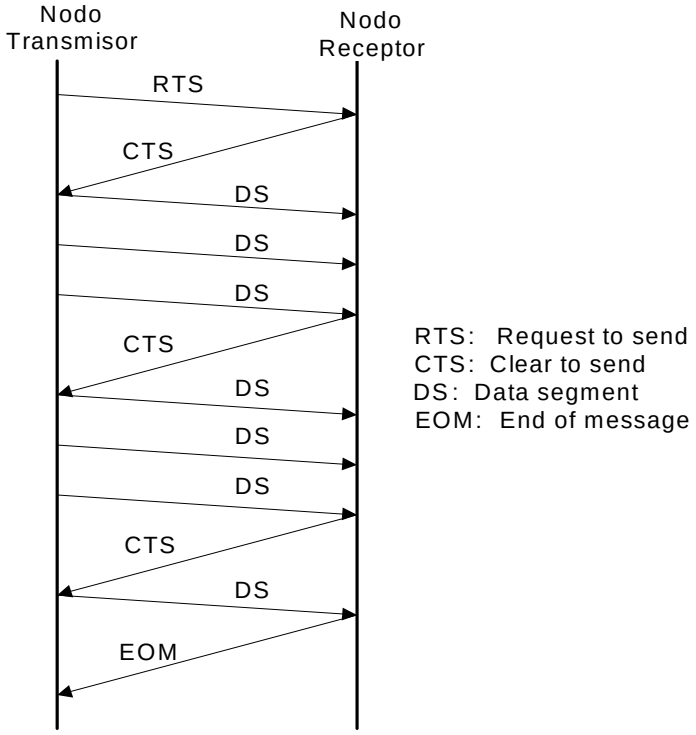


Figura. 5.25. Secuencia de una transferencia fragmentada orientada a nodos

do	d1	d2	d3	d4	d5	d6	d7
Ctrl = 10h	Número de bytes de datos		Num de segm	Reserv	PGN		

Campo de datos del mensaje RTS

do	d1	d2	d3	d4	d5	d6	d7
Ctrl = 11h	Número de segmentos		Seg #	Reserv	PGN		

Campo de datos del mensaje CTS

do	d1	d2	d3	d4	d5	d6	d7
Ctrl = FFh	Reservado				PGN		

Campo de datos del mensaje CA

do	d1	d2	d3	d4	d5	d6	d7
Seg #	Dato						

Campo de datos del mensaje DT

do	d1	d2	d3	d4	d5	d6	d7
Ctrl = 13h	Número de bytes de recibidos		Num de segm	Reserv	PGN		

Campo de datos del mensaje EOM_ACK

Figura. 5.26. Campo de datos de mensajes de la fragmentación orientada a nodos

En la figura 5.26. se muestran los campos de datos de cada tipo de mensaje, donde principalmente varia el byte de control “ctrl” según el mensaje utilizado.

Transmisión fragmentada tipo broadcast

Con este tipo de transmisión, no es necesario establecer una conexión directa con el nodo receptor del mensaje, se anuncia el inicio de una transmisión mediante el mensaje BAM (Broadcast Announce Message), este mensaje contiene el número de bytes, número de segmentos y grupo de parámetros del grupo de datos a enviar como se muestra en la figura 5.27.

do	d1	d2	d3	d4	d5	d6	d7
Ctrl = 20h	Número de bytes de datos		Num de segm	Reserv	PGN		

Figura. 5.27. Campo de datos mensaje BAM SAE J1939

No existe ninguna respuesta al mensaje BAM por parte de los nodos receptores. La transmisión del primer segmento de datos se da después de un rango de tiempo entre 50 y 200 ms a partir del mensaje BAM. De igual manera, el tiempo de retardo entre los segmentos de datos es de 50 a 200 ms como muestra la figura 5.28.

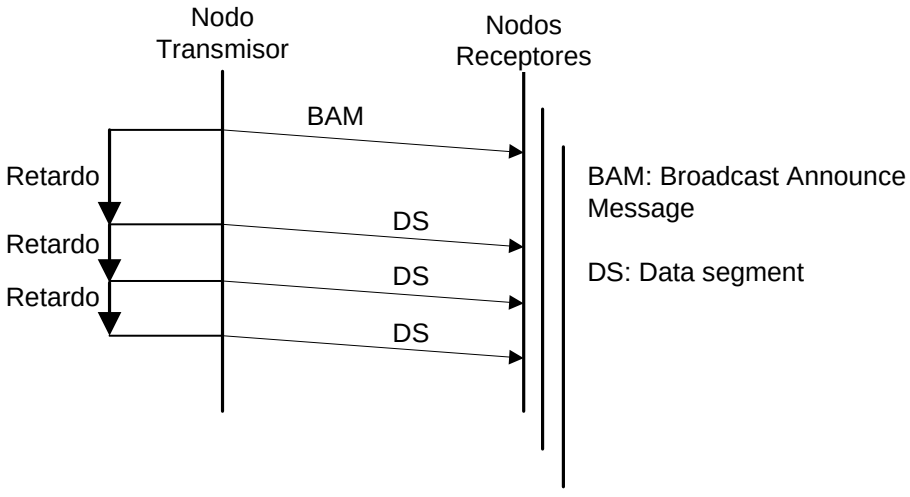


Figura. 5.28. Secuencia de mensajes de una transmisión fragmentada tipo broadcast

CAPITULO 6

ASPECTOS RELACIONADOS A LA IMPLEMENTACION DE SISTEMAS CAN

6.1 ALTERNATIVAS DE IMPLEMENTACIÓN CAN

Existen algunas alternativas de diseño e implementación de redes CAN, en un comienzo se implementaron redes CAN basadas en la capa física y de enlace de datos del protocolo según el estándar ISO. Con la disponibilidad de varios protocolos de alto nivel y perfiles de dispositivos como CAL, CanOpen y DeviceNET, ahora es posible utilizar servicios de capa 7 y habilitar una comunicación abierta, así como asegurar la intercambiabilidad de dispositivos mediante el uso de perfiles predefinidos.

En general existen tres alternativas de implementación de redes CAN:

1) Implementación basada en servicios de capa 2 o enlace de datos.

2) Implementación basada en protocolos de alto nivel estandarizados de capa 7.

3) Implementación basada en protocolos de alto nivel y perfiles de dispositivos.

6.1.1 Implementación basada en servicios de capa 2

La capa de enlace de datos del protocolo CAN provee principalmente al usuario una comunicación con servicios de requerimiento y envío de mensajes CAN; donde el proceso de aplicación opera directamente mediante identificadores de mensajes CAN

En este tipo de implementación, los servicios de administración de red y otros servicios adicionales (envío de mensajes largos) deben proveerse por medio de las aplicaciones por

si mismas, es decir, el diseñador se encarga de desarrollarlas formando servicios únicos para un determinado proceso (comandos, parámetros, etc).

Los sistemas basados en este tipo de implementación no cumplen con los requerimientos de sistemas de comunicaciones abiertos. Por otro lado estos servicios permiten la implementación de sistemas específicos optimizados donde los requerimientos de comunicaciones abiertas no son de vital importancia.

6.1.2 Implementación basada en protocolos estandarizados de capa 7

Este tipo de implementación provee un completo acoplamiento entre los procesos de comunicación y aplicación, y una estructura muy clara con respecto al software.

Los protocolos estandarizados de alto nivel permiten realizar un tipo de comunicación abierta es decir que brindan las mismas bases para cualquier aplicación.

Un protocolo de alto nivel usualmente incluye una serie de características especiales como administración de red, tipos de datos, funciones de prueba y consistencia de objetos de comunicación definidos y un uso transparente de identificadores.

6.1.3 Implementación basada en perfiles estandarizados

Mediante la disponibilidad de comunicaciones estandarizadas y perfiles de dispositivos, se pueden obtener extensas definiciones para cada dispositivo y funciones ya especificadas. Mediante estas definiciones, la implementación de una aplicación en un dispositivo también esta predefinida. Las interfaces de datos así como las funciones y comportamiento del sistema están definidos en el perfil.

Como resultado, la estructura de comunicación y los posibles procesos de un sistema se encuentran predeterminados.

En la figura 6.1. se muestran los diferentes tipos de implementación de redes CAN. La parte A señala las características de comunicación de una implementación pura de capa 2, la parte B señala las características para una implementación de capa 7 (CAL) y la parte C para una implementación mediante perfiles de dispositivos (CanOpen, DeviceNet)

Los bloques que se muestran totalmente en blanco definen características que deben ser desarrolladas por el programador.

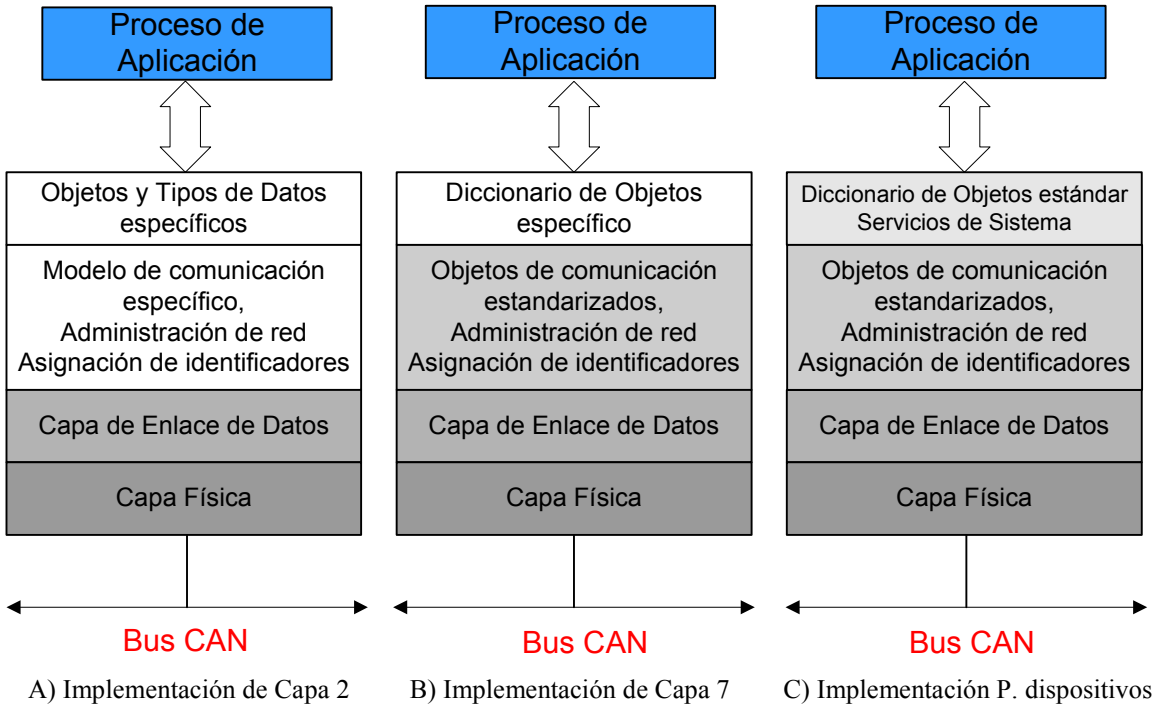


Figura. 6.1. Alternativas de implementación CAN

Criterio de selección de Protocolos de alto nivel y Perfiles de Dispositivos

Además de las definiciones de protocolos como el SAE J1939 que son desarrolladas para aplicaciones muy específicas (área automotriz), existen protocolos de alto nivel universales diseñados por compañías u organizaciones como CIA (CAN in automation) que desarrolló CAL y CanOpen u OVDA (Open DeviceNet Vendors Association) que desarrolló DeviceNet.

CanOpen y DeviceNet (capítulo 5) son las soluciones estandarizadas más comunes para la implementación de sistemas basados en CAN. Estos protocolos definen además perfiles de dispositivos para varios dispositivos utilizados típicamente en la tecnología de automatización, de esta manera ofrecen al usuario gran flexibilidad de implementación de sistemas en diferentes campos de aplicación.

La interfaz al programa de aplicación (diccionario de objetos en CanOpen y modelo de objetos en DeviceNet), la cual está estandarizada para cada aplicación, permite una fácil integración de componentes como módulos de E/S, sensores o controladores.

Debido a los mecanismos definidos en CanOpen y DeviceNet, los sistemas maestro/esclavo se pueden poner en fácil operación y rápidamente. Además, se pueden desarrollar sistemas con características plug and play debido a su esquema predefinido de asignación de identificadores.

Si las funciones de CanOpen o DeviceNet definidas en las especificaciones están implementadas de manera correcta, estos protocolos crean mayor demanda en los microcontroladores con respecto al tamaño de código, en comparación a las implementaciones puras de capa 2. En particular, los objetos de administración y la posibilidad de enviar arreglos de variables o datos (PDO u Objetos de ensamblaje) representan un código bastante grande y por lo tanto mayores tiempos de ejecución de código.

La especificación CAL solo define procesos de comunicación generales requeridos en sistemas distribuidos. Esto incluye servicios (como se analizó en el Capítulo 5) para la transmisión de datos de aplicaciones, y para control y verificación de nodos de la red a nivel de comunicación. CAL no define ningún objeto de comunicación para dispositivos específicos. Por lo tanto el usuario es libre de elaborar sus propias aplicaciones específicas y puede adaptar la comunicación del sistema a sus requerimientos. Esta característica está bastante limitada cuando se utilizan perfiles de dispositivos ya que se encuentra definido el acceso a los datos y el momento en que debe ocurrir.

CAL es particularmente conveniente para la implementación de soluciones de sistemas específicos en ciertos campos de aplicación como la tecnología médica y de medición, así como para la implementación de sistemas de control cerrados con unidades inteligentes descentralizadas. CAL representa una menor extensión de código en comparación a CanOpen y DeviceNet.

6.2 Diseño de Redes basadas en CAN

El diseño de sistemas de comunicación orientado a mensajes como CAN está basado en el número de mensajes específicos de aplicación del sistema. El diseño de una red CAN normalmente empezará con el análisis del número y tipo de mensajes de aplicación (datos o señales periódicos o no periódicos) así como el análisis de los mensajes transmisores y receptores. Como resultado del análisis se puede conocer el número de mensajes específicos con su tipo, la relación productor-consumidor, la frecuencia y el tiempo de latencia de los mensajes.

Mediante la agrupación de mensajes en grupos apropiados, y determinando el modo de transmisión (requerimientos remotos, cíclicos u orientado a evento), los mensajes son mapeados en un nuevo sistema de grupos de mensajes.

Los mensajes CAN (objetos CAN) del sistema son determinados por la asignación de identificadores al grupo de mensajes.

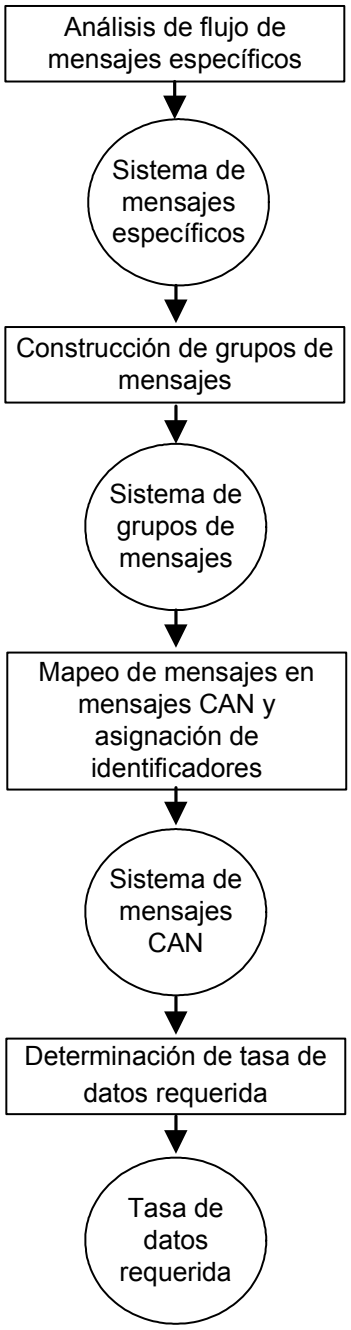


Figura. 6.2. Diseño sistemático de redes CAN

6.2.1 Mapeo de mensajes dentro de grupos de mensajes

Una red CAN se basa en una comunicación mediante requerimientos remotos de mensajes. Además de esta característica, la capacidad multimaestra permite la generación de mensajes CAN orientada a eventos en cada nodo.

Un útil agrupamiento de mensajes individuales específicos en grupos de mensajes es de vital importancia en la red CAN para minimizar la cantidad de mensajes CAN y la carga del bus.

La figura 6.3. muestra el principio de agrupamiento de mensajes individuales en un solo grupo de mensajes dentro del campo de datos de la trama CAN.

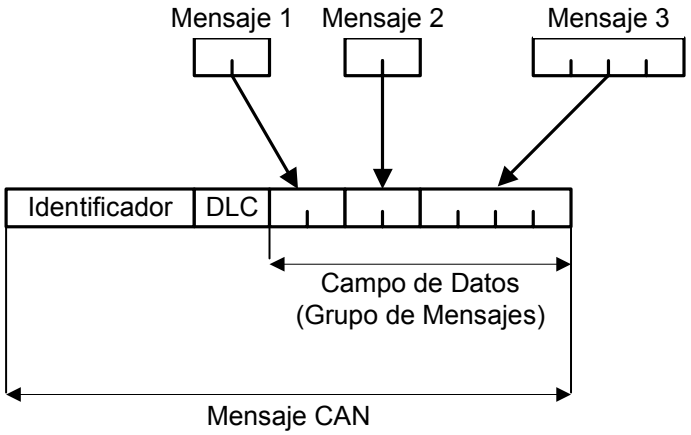


Figura. 6.3. Agrupamiento de mensajes específicos en un grupo de mensajes

También es posible el múltiple uso de mensajes CAN para la transmisión de información diferente (multiplexación), insertando un subidentificador dentro del mensaje CAN.

Este principio permite transmitir un largo número de parámetros mediante un simple mensaje CAN, utilizando un solo identificador. Este método se provee por medio de la "variable multiplexada" en CAL y los PDOs multiplexados en CANopen.

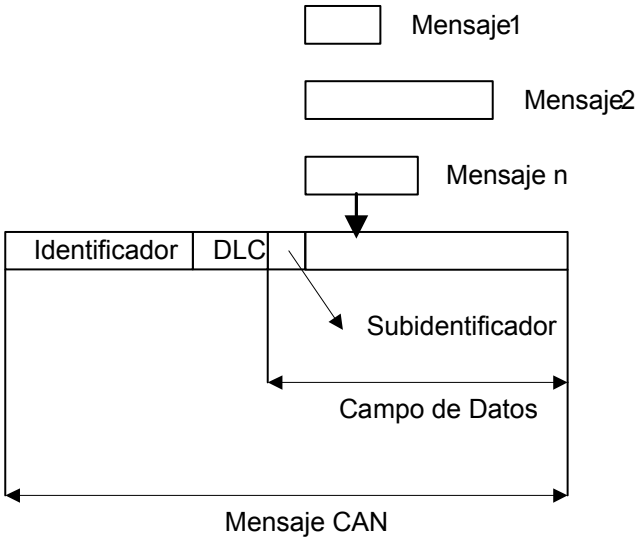


Figura. 6.4. Principio de uso múltiple de mensajes por medio de sub-identificadores

6.2.2 Asignación de grupos de mensajes en mensajes CAN

En el protocolo CAN versión estándar se puede obtener un total de 2048 mensajes diferenciados por su identificador de 11 bits. Existen 2 estrategias principales para la asignación de identificadores de mensajes CAN:

1) Asignación de identificadores orientada a la prioridad del mensaje.

2) Asignación de identificadores orientada al proceso.

Asignación de Identificadores orientada a la prioridad

Mediante los identificadores de mensajes se puede determinar la prioridad de acceso al bus y el máximo tiempo de latencia de un mensaje (sección 6.2.3). La estrategia es simple y consiste en la asignación de identificadores de mensajes de acuerdo a su prioridad y depende del diseñador de la red establecer los dispositivos o funciones de carácter relevante para asignar a ellos los identificadores de mayor prioridad.

Asignación de Identificadores de mensajes orientada al proceso

La asignación de mensajes también puede ocurrir desde el punto de vista del proceso en sí. Es decir que se divide al identificador del mensaje CAN en grupos de bits, los cuales se codifican de acuerdo a las funciones que se van a realizar y al direccionamiento del nodo fuente del mensaje o del nodo destino del mensaje. Este tipo de estrategia es aplicada por el protocolo SAE J1939 (Capítulo 5).

La asignación de identificadores en grupos de mensajes generalmente ya es desarrollada por el diseñador del sistema con una implementación CAN de capa 2 y es codificada en el programa de aplicación.

CanOpen y DeviceNet tienen esquemas de asignación de identificadores predefinidos, de acuerdo a estos esquemas, los identificadores son asignados a diferentes mecanismos de comunicación basados en el número del nodo. Además ofrecen la posibilidad de una libre configuración de asignación de identificadores como parte de la instalación del sistema o permite una modificación de identificadores dentro de ciertos límites.

6.2.3 Comportamiento en tiempo Real de las Redes CAN

Debido al uso común del medio de comunicación (sistema de bus serial) por parte de todos los nodos transmisores, existe un tiempo de retardo adicional causado por la transmisión de información. En sistemas de comunicación con accesos de bus determinísticos (Capítulo 1), el tiempo de retardo adicional es determinado ya sea por el máximo tiempo que demora el testigo (token) en recorrer toda la red en sistemas token-passing o por el tiempo de ciclo en sistemas maestro-esclavo.

A pesar de que CAN está basado en un sistema con acceso al medio arbitrario descentralizado, se puede garantizar un comportamiento equivalente a los sistemas con accesos al medio determinísticos. Sin embargo, se debe asegurar que los mensajes con alta prioridad no ocupen continuamente el bus. Un método simple para limitar la continua transmisión de mensajes con alta prioridad es insertar un intervalo de tiempo apropiado *(tiempo mínimo de inhibición)* entre las transmisiones de uno o varios mensajes de alta

prioridad. Por lo tanto dentro de este tiempo es posible la transmisión de mensajes de baja prioridad.

En los protocolos de alto nivel (CAL, CANopen, DeviceNet) se tienen mecanismos que aseguran que los mensajes de alta prioridad puedan volver a ocupar el bus solo después del intervalo de tiempo de inhibición. Esto no tiene ninguna repercusión en el tiempo de latencia de los mensajes de alta prioridad pero reduce la tasa de transmisión posible de dichos mensajes.

La figura 6.5. muestra el peor de los casos relacionado a la transmisión de mensajes de alta prioridad, es decir cuando todos los mensajes de mayor prioridad requieren transmitir al mismo tiempo. La figura se basa en un sistema con 16 mensajes de alta prioridad cada uno de 2 bytes de longitud (longitud de trama con un máximo de 73 bits). Si se transmite a una velocidad de 1 Mbps, el tiempo de transmisión de cada mensaje será de 73 us. Los 16 mensajes de alta prioridad se transmitirán en un total de 1.168 ms. La consideración de una ventana de tiempo adicional para la transmisión de mensajes CAN de baja prioridad es necesaria solo en los sistemas con un elevado promedio de carga de bus por parte de los mensajes de alta prioridad. En este caso sería apropiado un tiempo de inhibición de 1.5 ms. El sistema puede asegurar un tiempo de latencia menor a 1.5 ms para los 16 mensajes de alta prioridad incluyendo un tiempo para la transmisión de mensajes de baja prioridad (332us).

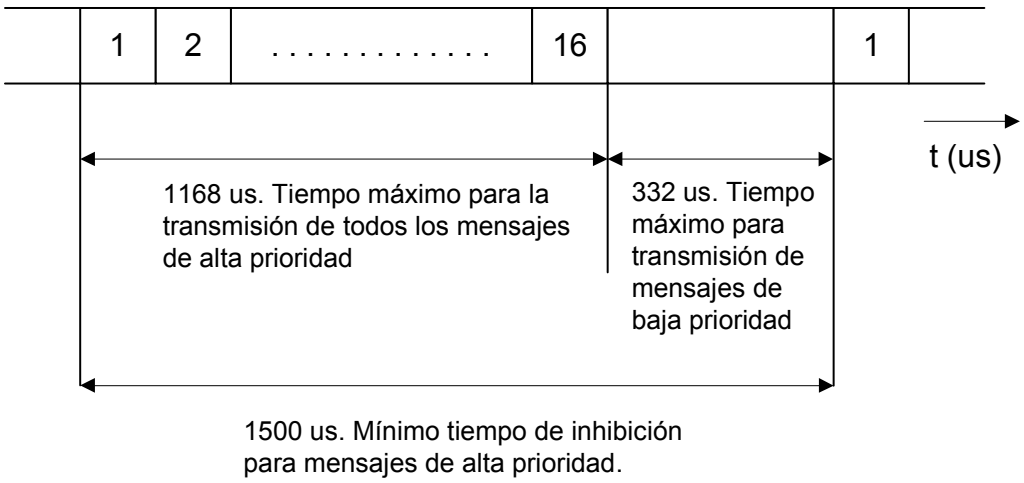


Figura 6.5. Estimación del máximo tiempo de latencia para mensajes CAN.

6.2.4 Determinación de la Tasa de Datos Requerida

Como el máximo tiempo posible de latencia de un mensaje CAN se determina mediante la duración de la transmisión de todos los mensajes CAN de alta prioridad en conjunto, la tasa de datos requerida está normalmente determinada según los tiempos de latencia requeridos.

Aunque el protocolo CAN permite una tasa de datos máxima de 1 Mbps, es aconsejable operar el sistema solo con la tasa de datos necesaria en relación a los requerimientos de tiempo de latencia. Esto se debe dar ya que con el aumento de la tasa de datos también aumentan los requerimientos de procesamiento en los nodos y la red se vuelve más propensa a la interferencia electromagnética.

Una limitante para la tasa de datos también es la extensión de la red, que se analiza en el Capítulo 3.

6.3 Implementación de nodos CAN

Existen tres tipos de implementación de nodos CAN basados en los diferentes tipos de controladores presentados en el Capítulo 4:

1) Nodo de entrada/salida.

2) Nodo con microcontrolador.

3) Nodo con interfaz de sistema (Host).

6.3.1 Nodo de entrada/salida

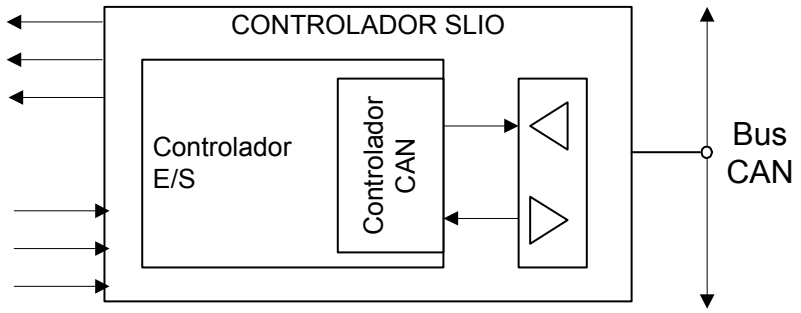


Figura 6.6. Nodo de entrada/salida

Es la forma más simple de implementar un nodo CAN y consiste en un enlace serial de entrada/salida conocido como SLIO (Serial Linked I/O) y un transceiver. Los nodos de este tipo son utilizados como interfaces de proceso (funciones de e/s) mediante el bus CAN.

Una solución SLIO presenta una desventaja relacionada con los costos por canal de entrada/salida, en comparación a una solución con microcontrolador la cual provee un gran número de canales de e/s, mayor flexibilidad y permite pre-procesamiento de datos.

6.3.2 Nodo con microcontrolador

En este tipo de implementación existen dos variantes que se muestran en la figura 6.7, ya que el controlador CAN puede encontrarse dentro del microcontrolador (controlador integrado) o fuera de él (controlador stand alone). Además del enlace de procesos mediante el bus CAN, los nodos con microcontrolador también permiten la implementación de funciones de procesamiento descentralizadas en forma de nodos inteligentes.

El uso de implementaciones mediante microcontroladores con controlador CAN Stand Alone ha disminuido en los últimos años debido a que la gran mayoría de microcontroladores están siendo fabricados con controladores CAN integrados y se ha convertido en una característica básica como la interfaz serial o I2C.

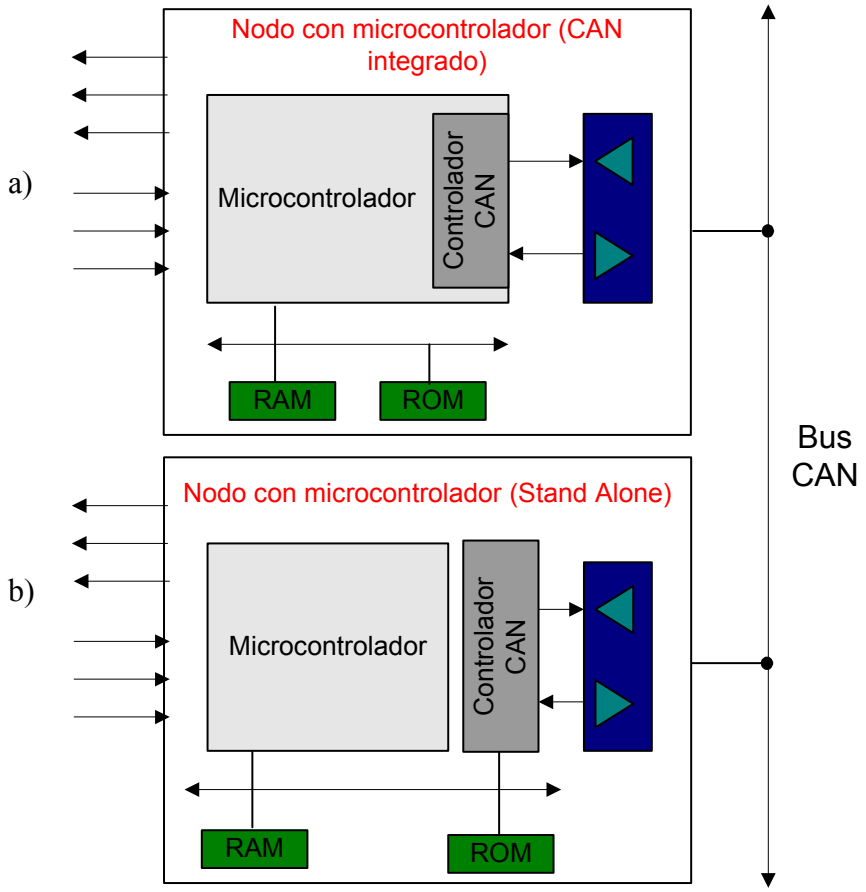


Figura. 6.7. Nodo con microcontrolador a) CAN integrado, b) CAN Stand Alone

6.3.3 Nodo con interfaz de sistema (Host)

Existen dos tipos de implementaciones de nodos con interfaz de sistema Host, los cuales dependen de las funciones que realizan:

1) Interfaz Pasiva

2) Interfaz Activa

6.3.3.1 Interfaz Pasiva

El primer tipo de implementación con interfaz de sistema se muestra en la figura 6.8. y sirve para conectar el bus CAN a un sistema host como PCs o PLCs mediante un

controlador CAN. Este tipo de nodo generalmente consiste en una tarjeta de interfaz y un sistema host. En la tarjeta de interfaz CAN, el controlador desarrolla solo las conexiones entre el bus CAN y el bus paralelo o serial del sistema host. El sistema host accede a los registros del controlador mediante su interfaz y controla la comunicación directamente mediante el bus CAN.

Las funciones necesarias para la comunicación como el ingreso de mensajes a ser enviados por el controlador y la lectura de los mensajes recibidos desde el controlador, debe ocurrir bajo el control del Host (CPU).

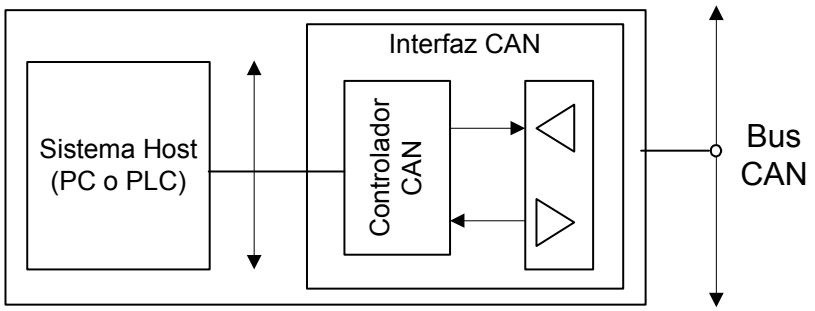


Figura. 6.8. Nodo con interfaz pasiva de sistema (Host)

Es necesaria una apropiada interfaz lógica que sea responsable de la adaptación de la interfaz Host a los requerimientos del bus CAN.

Especialmente en el bus ISA de un PC existe una manera de obtener acceso al controlador CAN: el acceso mapeado en memoria.

Acceso mapeado en Memoria

Los registros del controlador CAN son mapeados en espacios de direcciones de memoria del PC. El acceso al controlador CAN ocurre mediante un acceso normal de memoria directo.

Normalmente el espacio de memoria libre de un PC para este tipo de aplicaciones se encuentra en el área de segmentos de memoria desde C800hex hasta EFFF hex. Dependiendo de la implementación de la lógica de direccionamiento, las direcciones básicas de la tarjeta de interfaz CAN-PC pueden ser seleccionadas dentro de los segmentos de memoria mencionados en cantidades variables (4Kbytes, 8Kbytes, 16Kbytes).

Como los tiempos de bus de un PC pueden ser mayores que los de un controlador CAN, la interfaz CAN debe ser capaz de requerir estados de espera por parte de la unidad central de proceso (CPU) del PC.

6.3.3.2 Interfaz Activa

El segundo tipo de implementación con interfaz de sistema Host se muestra en la figura 6.9. y es conveniente para intereses de bus en las cuales el sistema Host debe ser relevado de las características de comunicación tanto como se pueda. Es decir para aplicaciones con gran carga de bus o grandes velocidades de transmisión de datos. Para lograr estas características (funciones de interfaz y procesamiento) se dispone de un microcontrolador incluido en la tarjeta de interfaz CAN.

El Sistema Host pasa los requerimientos de transmisión a la tarjeta de interfaz, la cual se encarga de relevar a dicho sistema mediante el microcontrolador además de procesar los mensajes CAN. Una vez procesados los mensajes la tarjeta los envía hacia el sistema Host sin necesidad de que este realice nuevos procesamientos.

Además de estas características, la tarjeta de interfaz CAN también permite desarrollar funciones de procesamiento de tiempo real críticas.

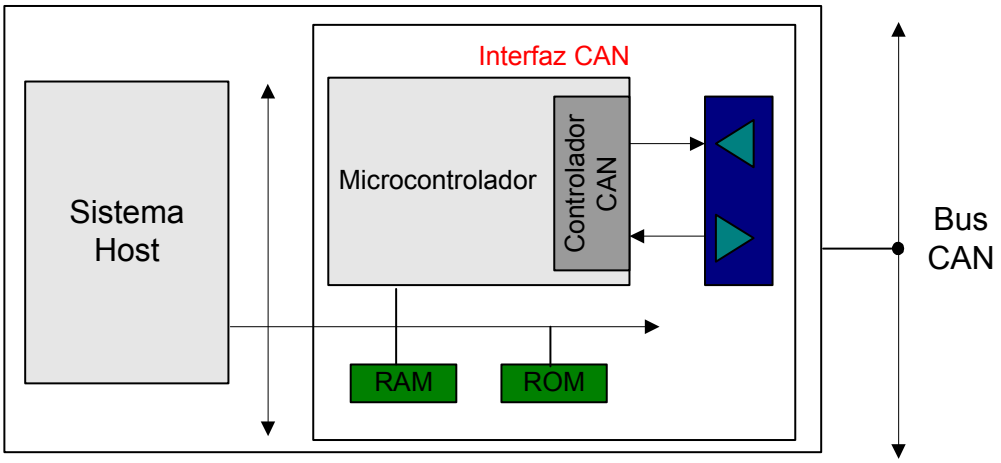


Figura. 6.9. Nodo con interfaz activa de sistema (Host), microcontrolador incluido en tarjeta

En sistemas que trabajan en base a protocolos de alto nivel como CANopen y DeviceNet, los sistemas Host generalmente desarrollan funciones maestras a nivel de administración de red y a nivel de aplicación. Esto significa que diferentes mensajes CAN tienen que ser enviados, recibidos y administrados y que se deben distribuir diferentes procesos de comunicación. Cuando se utiliza una interfaz CAN pasiva, éstas funciones deben ser provistas por el programa de aplicación. Esto significa que el proceso de aplicación Host se vuelve más complejo, más complicado de mantener y más difícil de probar.

Cuando se utiliza una interfaz CAN activa, todas las características específicas de comunicación por ejemplo en sistemas CANopen y DeviceNet, pueden ser asignadas a la interfaz y desarrolladas por el microcontrolador acoplado al programa de aplicación del sistema Host.

En sistemas con un promedio alto de carga del bus y alta velocidad de transmisión, el uso de interfaces CAN pasivas significa un incremento de carga del procesador del sistema Host sobre todo con el uso de controladores BasicCAN. En el caso de las tarjetas activas, el microcontrolador incluido en la tarjeta de interfaz desarrolla las operaciones de tiempo real críticas del controlador CAN y permite un claro y completo desacoplamiento entre los procesos de comunicación y de aplicación.

En este caso, una interfaz conveniente de capa 2 puede ser implementada para el sistema Host en el sistema de microcontrolador local. Esta desarrolla filtros de admisión de mensajes recibidos, provee tiempos de captura, almacena en espacios de memoria y proporciona colas de transmisión de diferente prioridad.

Dependiendo de las características y la estructura del hardware del sistema Host, el uso de un Puerto Dual RAM (DPRAM) o de una memoria FIFO es generalmente apropiado como una interfaz entre el controlador de red y el sistema Host. También es posible implementar una conexión indirecta de la interfaz CAN activa mediante otra interfaz serial o paralela presente en el sistema Host.

Interfaz DPRAM (Puerto Dual RAM)

El uso de la interfaz DPRAM o Memoria Compartida presupone que el sistema Host tiene un suficiente espacio para direccionamiento libre disponible, ya que el Host CPU tiene acceso directo a toda el área de direccionamiento de la DPRAM.

La interfaz DPRAM ofrece la ventaja de que es direccionada por el CPU como un sistema de memoria normal y puede ser implementada en forma de “interfaz de datos” entre procesos de comunicación y aplicación. Las áreas de datos (variables de entrada y salida) del proceso de aplicación pueden ser mapeadas directamente en la DPRAM. En el otro lado de la DPRAM, el sistema de comunicación actualiza las salidas de proceso reales mediante el bus CAN, o los requerimientos remotos o la actualización de entradas de procesos controladas por eventos en la DPRAM.

Interfaz FIFO

Si existen pocas direcciones disponibles en el sistema Host para acceso a la interfaz CAN, es recomendable utilizar una memoria tipo FIFO como una interfaz entre el sistema Host y la interfaz CAN.

Una memoria FIFO bidireccional provee dos canales de transmisión (colas), y pueden almacenar varios bytes incluso hasta varios kbytes por cola. La memoria FIFO normalmente ocupa solo una o dos direcciones en cada lado del sistema. Si se ocupa una sola dirección, el módulo FIFO automáticamente distingue entre un acceso de lectura o de escritura y direcciona el canal de transmisión correspondiente. Algunos módulos FIFO contienen registros adicionales como registros de configuración y estado.

La velocidad de transmisión de datos a la que puede funcionar una memoria FIFO es menor a la velocidad de un DPRAM. La principal desventaja de este tipo de interfaz es que se necesita información de protocolo adicional, la cual se tiene que intercambiar entre los dos sistemas mediante la memoria FIFO para distinguir entre comandos y datos.

Interfaces Serial y Paralela

La conexión de una interfaz CAN mediante una interfaz paralela o serial tiene la ventaja de que el sistema Host no tiene que proveer ninguna ranura o slot de expansión.

La desventaja de utilizar este tipo de interfases es la baja velocidad de transmisión de datos entre el sistema Host y la interfaz CAN. Si la velocidad de transmisión de datos es un requerimiento indispensable es preferible utilizar tarjetas de interfaz activas en este tipo de sistemas.

El sistema microcontrolador local permite una suficiente velocidad de servicio del controlador CAN y eficientes procesamientos de datos.

La comunicación puede ocurrir en modo full duplex mediante la interfaz serial o en modo semiduplex mediante la interfaz paralela.

6.3.4 Criterio de selección del controlador CAN

En el mercado existe una gran cantidad de controladores CAN de diferentes tipos y fabricantes. El principal aspecto a definir en la selección de un controlador es la arquitectura BasicCAN o FullCAN.

El concepto BasicCAN está basado en una simple interfaz para pasar los mensajes CAN entre el microcontrolador y el controlador CAN. Para el envío y recepción de mensajes se tienen disponible un buffer respectivamente por lo tanto el usuario no tiene que determinar en que buffer fue recibido un mensaje o desde que buffer fue enviado. Mediante el mecanismo de un segundo buffer (oculto) receptor se puede leer un mensaje mientras otro está siendo recibido. Con este simple mecanismo, las operaciones requeridas para la lectura de mensajes recibidos se reducen el mínimo. Sin embargo, el concepto BasicCAN representa la creación de un proceso de filtraje para la admisión de mensajes. Los protocolos de alto nivel implementan un filtraje de admisión mediante software. Con un filtraje mediante hardware existe confusión entre los identificadores que deben ser recibidos por todos los nodos y los identificadores que deben ser recibidos por un cierto nodo.

Los controladores CAN que trabajan bajo BasicCAN no son capaces de responder automáticamente un requerimiento remoto. Cuando esta función es requerida se debe implementar mediante software. El protocolo DeviceNet no trabaja con tramas remotas y el protocolo CANopen puede utilizar mensajes de requerimientos remotos de una manera opcional.

Los controladores FullCAN tienen varios buffers de mensajes que pueden ser configurados como transmisores o receptores. Estos pueden ser dispuestos para recibir mensajes específicos en cada buffer y no necesitan de un filtro de admisión. Además pueden responder automáticamente requerimientos remotos. El tiempo de procesamiento en este tipo de arquitectura es mayor.

Muchos controladores que incluyen FullCAN trabajan de manera similar al INTEL 82527 el cual tiene 15 buffers de mensajes de los cuales 14 trabajan como buffers programables para recepción o transmisión y el último trabaja solo como buffer receptor en forma de BasicCAN. Este buffer toma todos los mensajes que no han sido recibidos en ninguno de los 14 buffers anteriores e incluye un simple filtro de admisión.

Desde el punto de vista de programación, un controlador FullCAN es particularmente conveniente en caso de que todos los mensajes a ser recibidos pueda ser configurados en los primeros 13 buffers de tal forma que el buffer 15 no sea utilizado. En caso de que el buffer 15 tenga que ser utilizado debido a la gran cantidad de mensajes CAN a ser recibidos, se debe crear un filtro de admisión y es más recomendable utilizar un controlador BasicCAN. El uso de un controlador BasicCAN es simple y requiere de pocas operaciones.

6.4 RESUMEN DE PASOS A TOMAR EN CUENTA PARA EL DISEÑO DE UNA RED CAN

Resumiendo los aspectos importantes para el diseño e implementación de una red CAN se pueden presentar tres pasos:

1) Tipo de Implementación:

- Implementación de capa 2.
- Implementación de capa 7.
- Implementación de capa 7 con perfiles de dispositivos

Las características, ventajas y desventajas de los tipos de implementaciones se presentan en la tabla 6.1.

2) Tipo de Nodo:

- Nodo de entrada/salida CAN (SLIO).
- Nodo con microcontrolador CAN.
- Nodo con tarjeta de interfaz CAN.

Las características, ventajas y desventajas de los tipos de nodos CAN se presentan en la tabla 6.2.

3) Arquitectura del Controlador CAN:

- Arquitectura BasicCAN.
- Arquitectura FullCAN.

Las características, ventajas y desventajas de las arquitecturas de controladores CAN se presentan en la tabla 6.3.

TIPO DE IMPLEMENTACIÓN	CARACTERÍSTICAS PRINCIPALES				
	Funciones	**Protocolo de Alto Nivel**	**Usos**	**Ventajas**	**Desventajas**
CAPA 2	- Capa Física - Capa de Enlace de Datos	-	- Aplicaciones pequeñas - Demostraciones	Menor costo en equipos.	Desarrolla solo de funciones básicas
CAPA 7	- Capa Física - Capa de Enlace de Datos - Administración de Red - Asignación de identificadores - Objetos de comunicación	CAL	- Área Médica - Técnicas de Medición	Funciones de administración de red.	Mayor tiempo de ejecución (programas)
		SAE J1939	- Área Automotriz		
CAPA 7 (Perfiles Estandarizados)	- Capa Física - Capa de Enlace de Datos - Administración de Red - Asignación de identificadores - Objetos de comunicación - Diccionario de objetos - Servicios de Sistema	CANopen	- Maquinaria - Sistemas de producción - Control de Procesos - Transporte	Seguridad y confiabilidad Integración de componentes Flexibilidad en campos de aplicación	Mayor costo en equipos Complejidad del protocolo
		DeviceNet	- Maquinaria - Sistemas de Producción - Control de Procesos - Transporte		

Tabla. 6.1. Características, ventajas y desventajas de las implementaciones CAN

TIPO DE NODO		CARACTERÍSTICAS PRINCIPALES		
		COMPONENTES	**VENTAJAS**	**DESVENTAJAS**
NODO DE ENTRADA/SALIDA		- Controlador de e/s (SLIO) - Controlador CAN - Transceiver	- Menor Costo - Mayor Velocidad	- No hay procesamiento de datos - Pocos canales de entrada/salida
NODO CON MICROCON-TROLADOR	**Contr. Stand Alone**	- Controlador CAN - Microcontrolador - Transceiver - Memoria RAM,ROM	- Procesamiento de daos - Más canales de e/s - Reduce carga de microcontrolador	- Mayor espacio físico - No existe sistema Host
	Contr. CAN incluido	- Microcontrolador CAN - Transceiver - Memoria RAM, ROM	- Procesamiento de daos - Más canales de e/s - Menor espacio físico	- Mayor carga al microcontrolador - No existe sistema Host
NODO CON TARJETA DE INTERFAZ	**Pasiva**	- Controlador CAN - Transceiver - Sistema Host	- Host controla funciones de comunicación CAN (requerimientos, procesamiento de mensajes) - Utiliza distintas interfaces (ISA,PCI) - Menor espacio físico	- Mayor carga al Sistema Host - Procesos de aplicación más complejos
	Activa	- Microcontrolador CAN - Transceiver - Memoria RAM, ROM - Sistema Host	- Reduce la carga del procesador del sistema (releva al Host) - Utiliza distintas interfaces (ISA,PCI) - Los procesos de comunicación y de aplicación se diferencian claramente.	- Mayor espacio Físico - Mayor tiempo de ejecución

Tabla. 6.2. Características, ventajas y desventajas de los tipos de nodos CAN

ARQUITECTURA	VENTAJAS	DESVENTAJAS
BasicCAN	- Operaciones de lectura y escritura reducidas al mínimo - Menor tiempo de procesamiento de mensajes - Menor Costo	- Un solo buffer de mensajes transmisor y receptor más un buffer de respaldo - Difícil de programar debido a la necesidad de filtros de mensajes - No responde a requerimientos remotos
FullCAN	- Varios buffers de mensajes programables como transmisores o receptores - Fácil de programar ya que no es necesario crear filtros - Responde a requerimientos remotos	- Mayor tiempo de procesamiento de mensajes - Mayor costo

Tabla. 6.3. Ventajas y desventajas de las arquitecturas de controladores CAN

Existe una técnica muy importante dentro del diseño e implementación de redes CAN y es el agrupamiento de mensajes en nuevos grupos de mensajes.

Se recomienda seguir los siguientes pasos para realizar el agrupamiento:

1) Analizar la cantidad de mensajes que van a circular en la red.

2) Analizar el tipo de mensajes que van a circular en la red.

3) Analizar la longitud de los mensajes que van a circular en la red.

4) Construir grupos de mensajes de acuerdo al análisis realizado anteriormente, es decir de acuerdo a las conveniencias y requerimientos de la red. Varios mensajes pueden quedar organizados en uno solo siempre y cuando no se supere el máximo valor de 8 bytes de datos.

5) Asignar identificadores a los diferentes grupos de mensajes.

CAPITULO 7

APLICACIONES DESARROLLADAS CON EL PROTOCOLO CAN

7.1 INTRODUCCIÓN

El protocolo CAN no es simplemente un concepto teórico ya que se ha estado implementando hace muchos años en una gran cantidad de aplicaciones. En un principio se desarrolló para mejorar los sistemas de comunicación dentro de los vehículos y posteriormente incursionó en un amplio campo de aplicaciones.

Una característica importante y determinante para la aceptación del protocolo CAN es la posibilidad de realizar comunicaciones CAN abiertas, las cuales permiten que se utilicen productos de diferentes fabricantes en la misma red.

CAN ha incursionado en el transporte, medicina, agricultura, y en general en muchos campos donde la automatización se pueda aplicar. Se puede resumir que CAN se ha implementado en los siguientes campos:

- Vehículos de pasajeros
- Vehículos de propósitos especiales
- Sistemas de transporte masivo
- Maquinarias e Ingeniería Mecánica
- Fábricas y control de procesos

- Automatización de edificios

7.2 CAN EN VEHÍCULOS

Los vehículos de hoy contienen centenares de circuitos, sensores, actuadores y otros componentes eléctricos. La comunicación entre dichos circuitos o dispositivos es indispensable para el correcto funcionamiento del vehículo. Por ejemplo, para realizar el encendido y apagado de las luces debe haber comunicación entre el tablero disponible para el conductor y los faros propiamente dichos. Esta comunicación generalmente se implementaba mediante un alambre por cada dispositivo lo cual se conoce como conexión punto a punto. Si se suman todas las posibles combinaciones de interruptores, sensores, motores, y otros dispositivos eléctricos dentro de los vehículos, el número resultante de conexiones punto a punto y la instalación eléctrica especializada es enorme. Esto representa un costo elevado y una gran complejidad en la instalación del sistema.

Un método conocido como MULTIPLEXING resuelve este tipo de problemas, ya que es un método para conectar módulos electrónicos distribuidos y transferir datos entre ellos, mediante un bus de datos serial, de esta forma se reduce el número de alambres combinando las señales en un solo cable (bus). El protocolo CAN, desarrollado por Robert Bosch, es un sistema de bus serial muy apropiado para este tipo de comunicaciones.

Originalmente, como ya se explicó, CAN fue desarrollado para aplicaciones en vehículos de motor. En los carros de pasajeros existen dos redes CAN como muestra la figura 7.1. El bus de color verde conecta la red de baja velocidad y el bus de color rojo conecta la red de alta velocidad. Este sistema toma el nombre de Multiplexado CAN, (Multiplexing CAN).

Parece seguro que, al menos, existirán dos buses multiplexados e independientes en los coches del mañana.

La red de alta velocidad opera en el rango de 125 Kbps hasta 1 Mbps. Mediante dicha red se conecta los dispositivos principales del vehículo que requieren de un control en

tiempo real, por lo tanto mayor velocidad de operación, y realiza el control de las unidades encargadas de la seguridad activa. Esta red abarca elementos que incluyen el control de motor, sistema ABS (anti-blocking system) y ASR (anti-slipping control), caja automática, suspensión activa, control de frenos, Air Bag, cinturones de seguridad entre otros.

La red de baja velocidad opera a velocidades menores a los 125 Kbps y conecta al bus los dispositivos auxiliares marginales y relacionados al confort del conductor y los pasajeros. Dichos dispositivos no representan un grave riesgo en caso de mal funcionamiento. Esta red se dedica a la transferencia de información general por lo que no necesitan una velocidad alta de transmisión y abarca dispositivos como: control de luces, aire acondicionado, encendido, asientos reclinables, vidrios eléctricos, apertura remota del tapón de gasolina, antena eléctrica, cargador de CDs, etc.

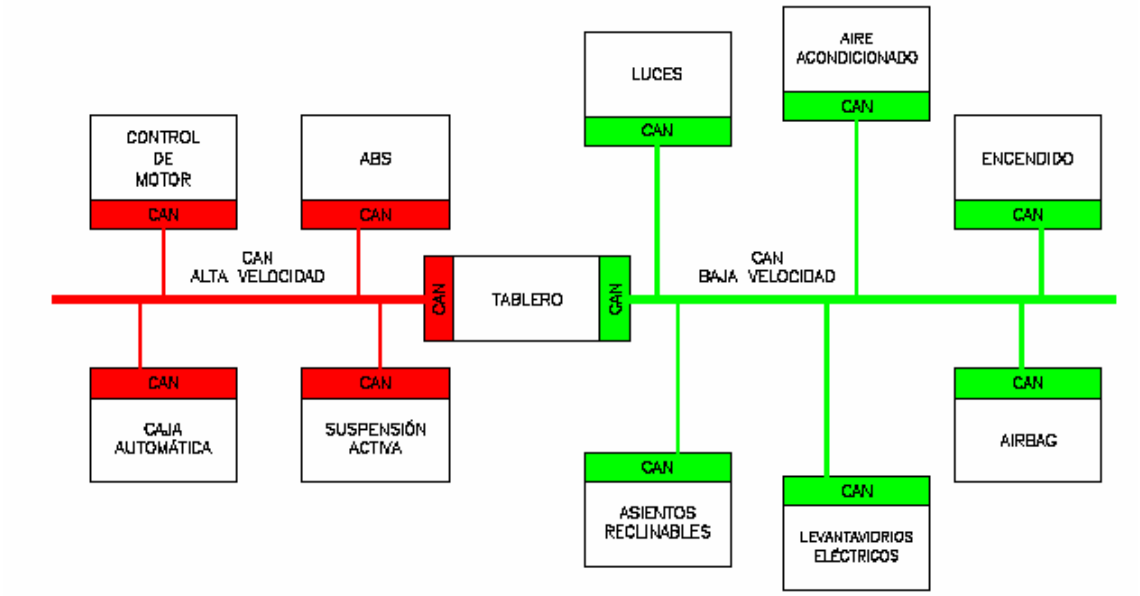


Figura. 7.1. Diagrama de conexión de las redes CAN de alta y baja velocidad dentro de un vehículo

Figura. 7.2. Conexión de las redes CAN de alta y baja velocidad

Términos representados en la figura 7.2:

- ECM: Engine Control Module (Módulo de control de llantas)
- TCM: Transmission Control Module (Módulo de control de transmisión)
- ETM: Electric Transmission Module (Módulo de transmission eléctrica)
- SAS: Stability Active System (Sistema de estabilidad activa)
- SRS: Security Retention System (Sistema de retención de seguridad)
- PDM: Passenger Door Module (Módulo de puerta de pasajero)
- DDM: Driver Door Module (Módulo de puerta de conductor)

- RTI: Ramp Travel Index (flexibilidad en la suspension y chasis de un vehículo)

- CCM: Monitor component Comprehensive

Por lo general se incorporan dos tarjetas CAN en un vehículo: la una contiene la circuitería necesaria para la red de alta velocidad se coloca en la parte delantera del vehículo y la otra que contiene la circuitería de la red de baja velocidad se coloca en la parte trasera del vehículo.

La figura 7.3. muestra las tarjetas delantera y trasera estándar de un vehículo. Dichas tarjetas corresponden un entrenador de comunicaciones del automóvil (simulador) diseñado por a la marca LUCAS-NULLE, el cual es muy semejante a la realidad.

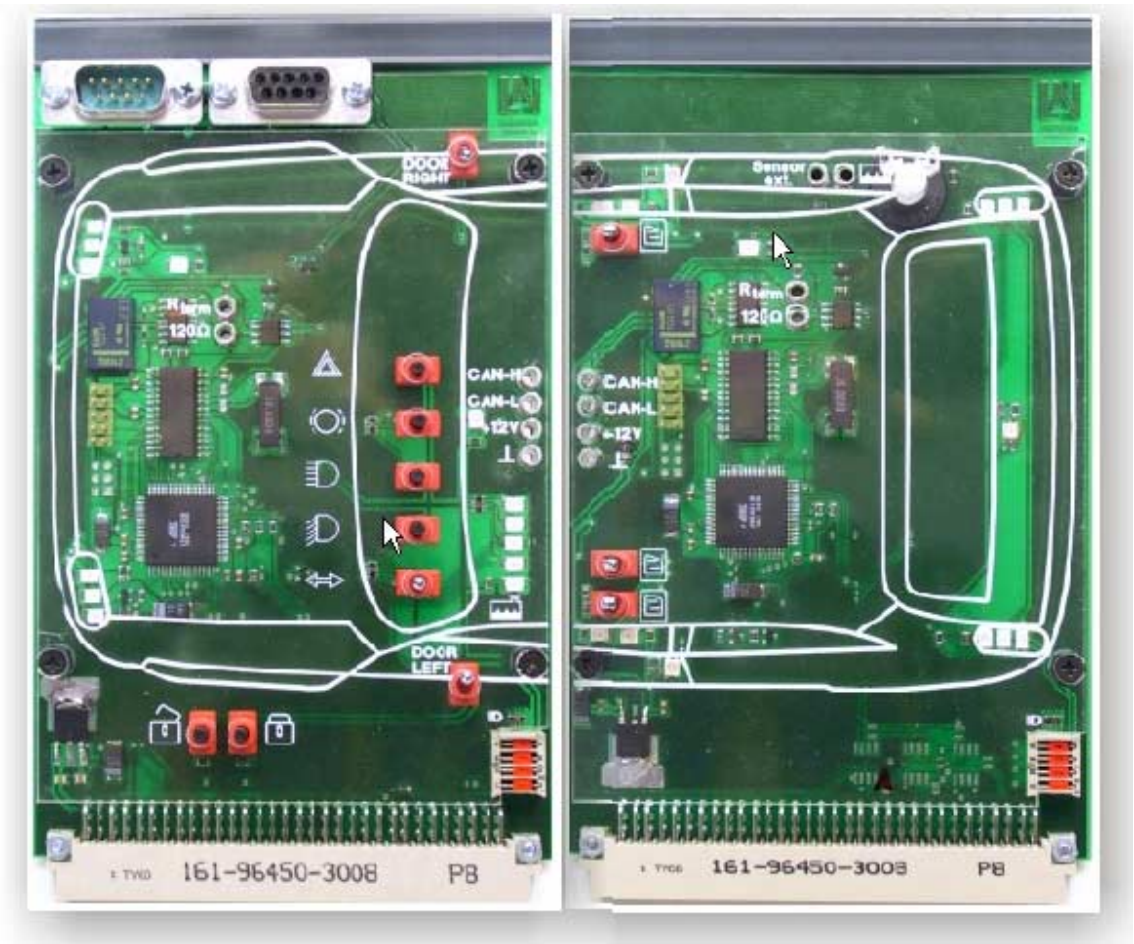


Figura. 7.3. Tarjetas de interfaz CAN para red de alta y baja velocidad (LUCAS-NULLE).

Las dos tarjetas se colocan en paneles y al igual que todos los sensores y actuadores, se interconectan mediante los terminales CAN_H y CAN_L del bus. Adicionalmente las tarjetas incluyen terminales de +12V y de conexión a tierra como muestra la figura 7.4.

Cada tarjeta contiene un microcontrolador con interfaz CAN incluida o de lo contrario un microcontrolador y un controlador CAN Stand Alone. Según la tarjeta (parte delantera o trasera del auto), se configura el microcontrolador para que trabaje ya sea a alta o a baja velocidad.

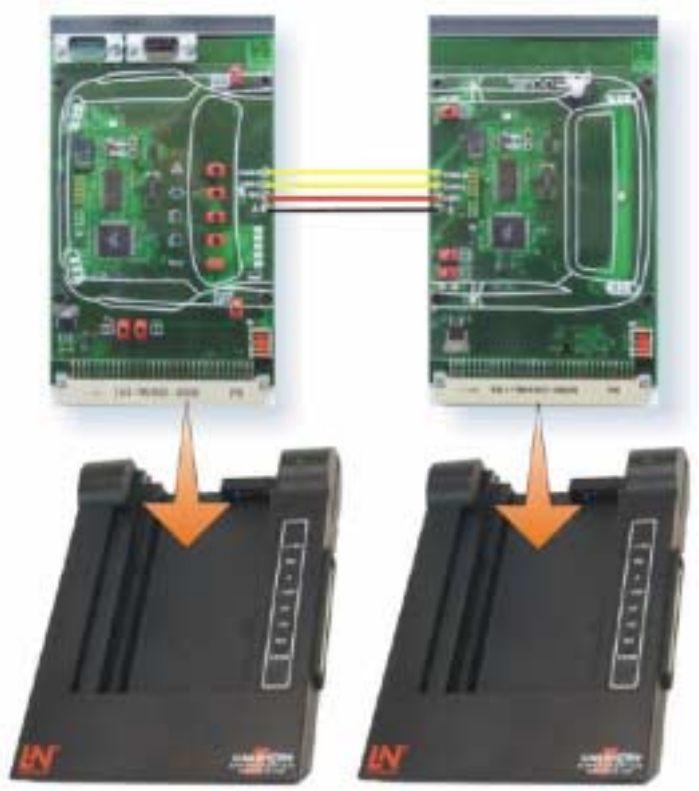


Figura. 7.4. Conexión entre las tarjetas de interfaz CAN para red de alta y baja velocidad

La tarjeta de red de alta velocidad tiene dos puertos de interfaz CAN con conectores DB9 (configuración en el capítulo 3), desde dichos conectores se realiza la conexión al panel de control (figura. 7.5.) y a su vez desde dicho panel se conecta a una PC para realizar la simulación CAN.

Figura. 7.5. Tarjetas CAN delantera y trasera.

Algunas de las características que poseen las tarjetas CAN dentro de los vehículos son las siguientes:

- Microcontrolador
- Controlador CAN Stand Alone
- Tranceptor CAN
- CAN de alta velocidad o de baja velocidad seleccionable
- Terminales para CAN_H, CAN_L, +12V y tierra
- Puerto CAN serial DB9
- Leds para simulación de iluminación, cerradura electrónica y reserva de combustible en el depósito
- Sensor para control de temperatura
- Elementos de entrada para realizar varias funciones del automóvil (interruptores)

La implementación de un bus CAN dentro de los vehículos no solo proporciona una disminución de cableado en el interior del vehículo, también proporciona facilidad de instalación y mejor control de todos los elementos funcionales del vehículo. Incluso el bus CAN puede ser una herramienta de ayuda al conductor del vehículo.

Red de alta velocidad CAN

A continuación se describen algunos elementos que conforman la red de alta velocidad:

El control de transmisión automática permite maniobrar el vehículo rápidamente y con seguridad en el tráfico vehicular, su consumo de gasolina es el mismo que en un sistema de transmisión manual.

Figura. 7.6. Sistema de transmisión CAN (caja de cambios)

La ECU (Engine Control Unit) o unidad de control de motor que se muestra en la figura 7.7. proporciona una regulación en tiempo real tanto en la inyección de gasolina como en los tiempos de ignición, esto garantiza un máximo desempeño del motor con un mínimo consumo de gasolina.

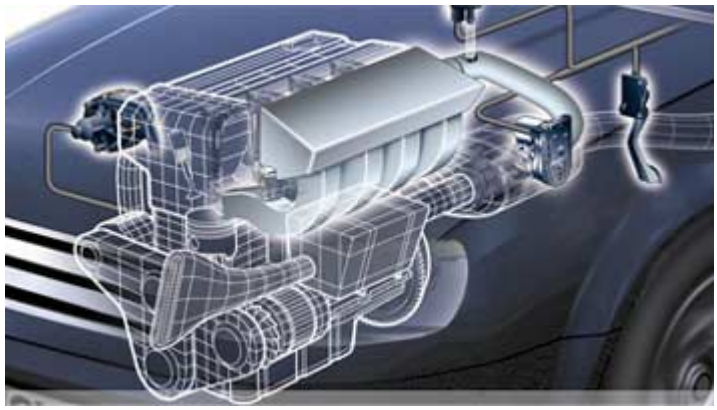


Figura. 7.7. Unidad de control de motor CAN

La potencia entregada a las llantas influye en la vida del motor de un vehículo, si la potencia del motor no es transferida a la carretera completamente, el control anti-deslizamiento (ARS) no solo controla el frenado y el giro de las llantas sino también reduce la potencia del motor mediante el bus CAN.

La unidad ASR también es activada si el conductor repentinamente presiona el acelerador cuando el vehículo esta atravesando una curva peligrosa en una carretera resbalosa. De igual forma el sistema actúa como estabilizador (figura 7.8.) del vehículo reduciendo el torque de frenado inclusive cuando existen cambios repentinos de trayecto o carga.

Figura. 7.8. Control de torque de frenado en curvas cerradas

Todas estas acciones se realizan en el sistema mediante diferentes sensores (velocidad, nivel, etc) y actuadores (válvulas, motores, etc) que se encuentran conectados al bus CAN.

Una característica importante de la red de alta velocidad es el sistema de bolsa de aire que proporciona gran seguridad en el momento en el que el vehículo sufra un impacto fuerte ya que están colocados sensores de aceleración (figura 7.9.) en diferentes lugares los cuales activan instantáneamente la bolsa de aire (figura 7.10.). Este sistema ha salvado muchas vidas alrededor del mundo.

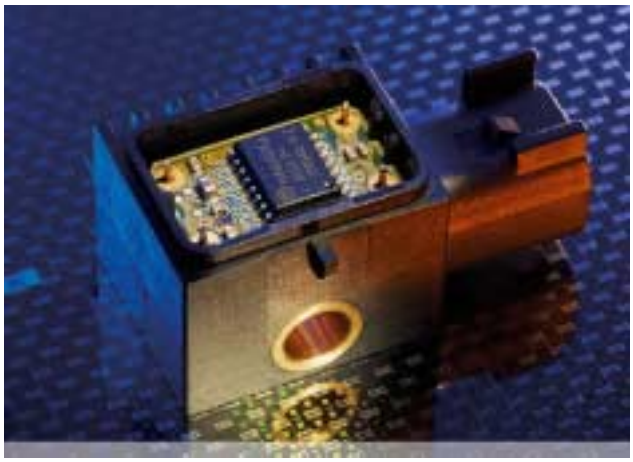


Figura. 7.9. Sensor de aceleración para bolsas de aire

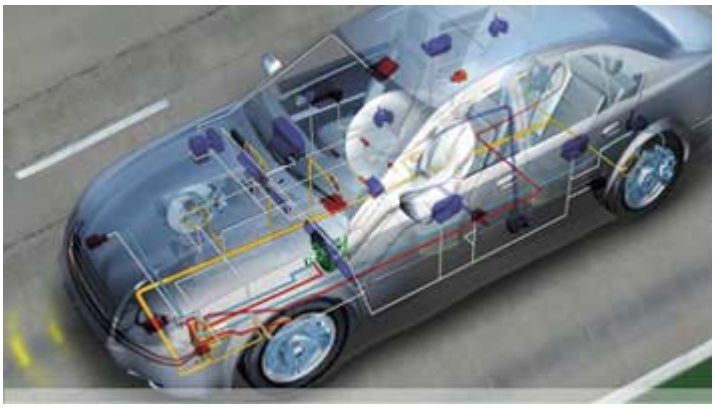


Figura. 7.10. Sistema Airbag (bolsa de aire)

Red de baja velocidad CAN

La red de baja velocidad influye en los dispositivos que proporcionan comodidad y confort al usuario (figura 7.11.) tales como: vidrios eléctricos, asientos reclinables, intensidad de luces automáticas, control de parlantes que regulan el volumen en función de la velocidad, calefacción en función de la temperatura, aire acondicionado, plumas que regulan su velocidad en función de la cantidad de lluvia, etc. Es decir, esta red no actúa sobre los dispositivos que proporcionan el funcionamiento en sí del vehículo como el motor, ruedas, transmisión, etc.

Figura. 7.11. Control de sensores y actuadores de la red de baja velocidad

Las redes CAN en el campo vehicular se seguirán incrementando en el futuro utilizando conceptos como el multiplexado, el cual, define varias redes de diferentes velocidades en los vehículos según los dispositivos que interconectan y sus requerimientos.

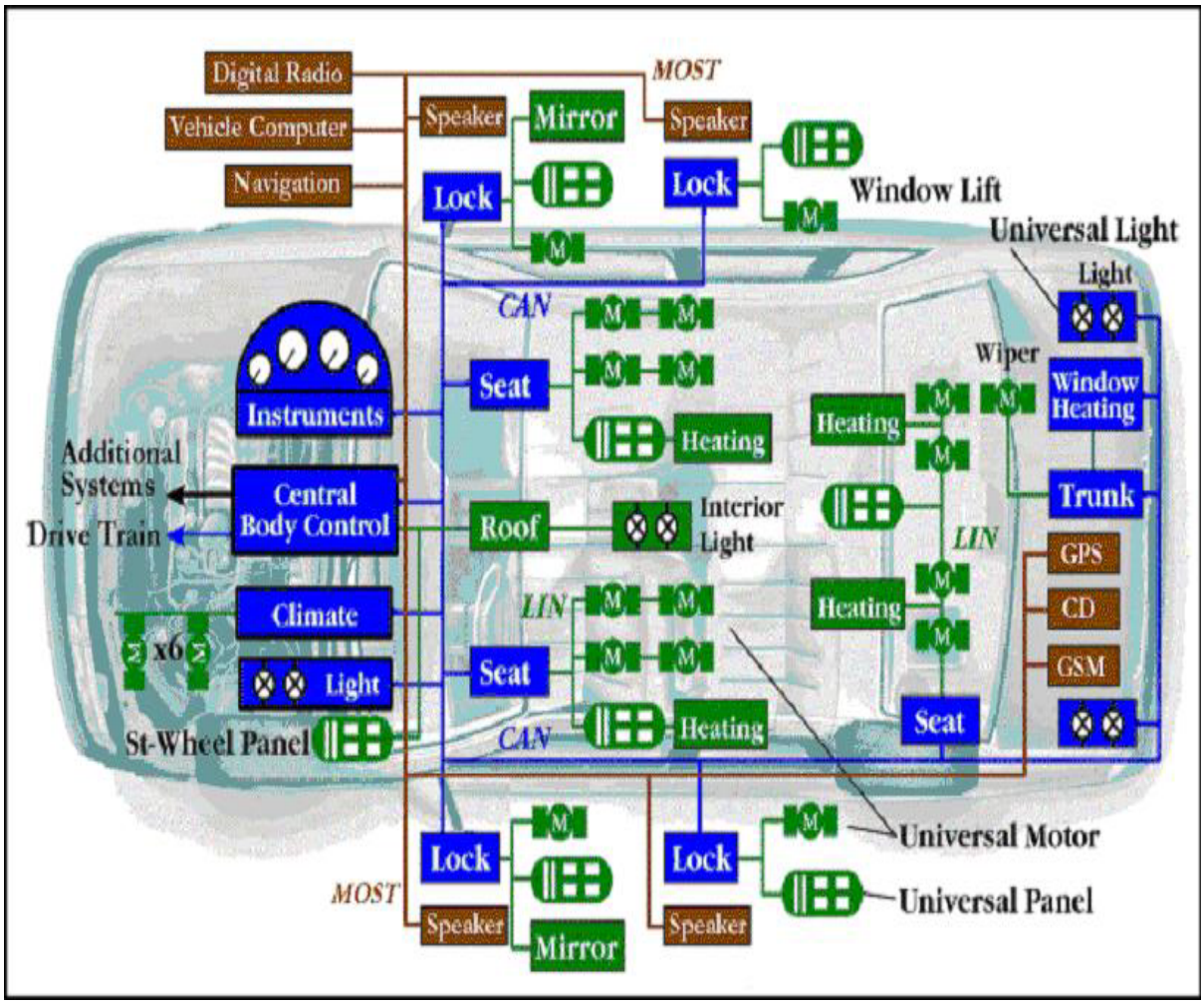


Figura. 7.12. Multiplexado de buses en los automóviles

7.3 CAN EN VEHÍCULOS DE PROPÓSITOS ESPECIALES

Los vehículos en la industria de la construcción como tractores, excavadoras, etc; así como los vehículos forestales y agrícolas utilizan el protocolo CAN. Este tipo de vehículos son llamados “máquinas sobre ruedas” y utilizan CAN tanto en el funcionamiento del vehículo en sí como en el control de los componentes del trabajo mecánico.

En los vehículos de la industria de construcción, CAN realiza el control de las palancas de mano y de pie para transferir los datos a los diferentes controladores y actuadores (bombas, etc) los cuales mantendrán los diferentes motores en los valores deseados.

Figura. 7.13. CAN en vehículos de propósitos especiales

Las unidades de control manejan las revoluciones del motor, velocidad de conducción, valores de presión y temperatura y transfieren estas señales al computador del panel central el cual despliega la información en pantallas en el momento en que el usuario presiona diferentes pulsadores o conmuta switches. En este campo de aplicación se han utilizado protocolos de alto nivel como son SAE J1939 y CANopen.

7.4 CAN EN VEHÍCULOS DE TRANSPORTE MASIVO

Vehículos sobre rieles

Una de las primeras aplicaciones CAN en el campo del transporte masivo se da en los vehículos sobre rieles, como son: trenes, ferrocarriles, metros, etc.

Figura. 7.14. CANopen en vehículos sobre rieles.

En los vehículos sobre rieles existe una gran aplicación de CANopen sobre todo en países europeos. El perfil de aplicación CANopen para la red de integración en vehículos ferroviarios conecta los sistemas subalternos fabricados independientemente por varias compañías. Las interfaces de los sistemas subalternos estandarizados permiten la integración simplificada del sistema, la cual implica un ahorro en los recursos financieros y humanos.

Los dispositivos CANopen incluyen el panel de control del conductor, el control de tracción, el suministro de poder principal, control de la puerta, etc.

Vehículos Marítimos

En la electrónica marítima CAN se aplica en las redes de dispositivos internos de barcos, submarinos, buques, etc. Existen muchos dispositivos en el área marítima dedicados a conexiones CAN.

CANopen en la electrónica marítima se utiliza para el control de potencia, control del generador, bombas de carga y válvulas. CANopen proporciona un protocolo que facilita y asegura la interoperabilidad de sistemas y equipos marítimos.

Vehículos aéreos

CAN es utilizado en el backbone o cableado vertical de muchos aviones, es decir todos los sensores, sistemas de navegación e investigación y los PCs instalados en la cabina del piloto.

Dentro de las aplicaciones CAN se encuentran también los datos producidos en vuelo como estaciones de voz y video para ayuda y recomendaciones a la tripulación desde la cabina del piloto.

7.5 CAN EN MÁQUINAS ESTACIONARIAS E INGENIERÍA MECÁNICA

Las redes CAN no solamente se encuentran instaladas en las llamadas “máquinas sobre ruedas” sino también en “máquinas estacionarias”, desde máquinas simples de café hasta complejos sistemas ubicados en las salas de operaciones de hospitales.

CAN es una solución ideal para el enlace de componentes de automatización que son controlados, regulados y medidos utilizando diferentes tipos de sensores y actuadores. Esto se da debido a su alta velocidad de transmisión (1Mbps a 40 mts), la inmunidad a la interferencia y principalmente su habilidad de trabajar en tiempo real.

Dispositivos como las máquinas textiles, de papel, impresoras, máquinas de empaquetamiento, etc. Necesitan un sistema de bus en tiempo real para la conexión de sus subsistemas inteligentes.

Figura. 7.15. CAN en máquinas estacionarias (máquina de café)

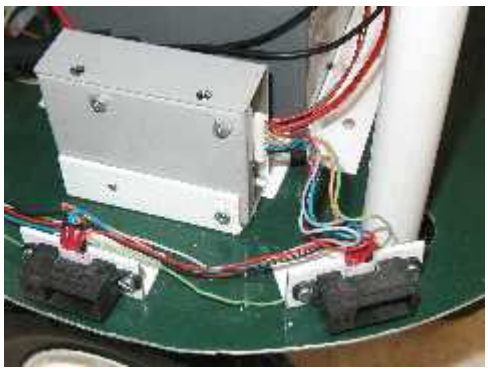


Figura. 7.16. CAN en Robots

CAN también es adecuado para sistemas de bus sensor-actuador para robots y controladores CNC. La figura 7.16. muestra un robot que realiza la lectura de 6 sensores infrarrojos de distancia.

CAN en Técnicas Médicas

El uso de CAN ha sido muy difundido en técnicas médicas, la razón principal para esta aplicación es su alta confiabilidad. Se han desarrollado máquinas estáticas en el campo de la medicina como son máquinas para tomografías, rayos X, sillas para dentistas, etc.

Las exigencias médicas dentro del cuarto de operaciones se han incrementado mucho en los últimos años, una buena solución para cumplir con dichas exigencias dentro del plano técnico es aplicar una red CAN.

El desarrollo empezó con extensos análisis de los procesos del OR (operation room) o cuarto de operaciones; estos análisis proporcionaron la información esencial sobre los procesos individuales en donde era posible una optimización. Un ejemplo de este tipo de análisis es Sios (Siemens Integrated OR System). Siemens habilitó el desarrollo de un sistema con funcionamiento centralizado y reunió los requisitos esenciales de varias disciplinas quirúrgicas (la cirugía general, ginecología, la urología, la ortopedia).

La red CAN controla todos los elementos esenciales de un OR como son las luces, la mesa, insuflador, cámaras de video, equipos de radiografía, equipos de ultrasonido, rayos X, teléfonos, etc.).

La comunicación entre Sios (figura 7.17.) y los componentes individuales está basada en la tecnología CANopen. La ventaja de la red CANopen es que se puede conectar un número casi ilimitado de componentes pertenecientes al OR, con una estabilidad y transmisión de datos muy confiable.

Figura. 7.17. CAN en la medicina (OR)

7.6 CAN en Fábricas y Control de Procesos

En la automatización de fábricas y control de procesos, CAN es utilizado para interconectar máquinas, unidades de control de procesos, PLCs, PCs, sensores, actuadores y subsistemas de producción. Las aplicaciones típicas incluyen conveyors, almacenamiento de datos de producción y sistemas configurables.

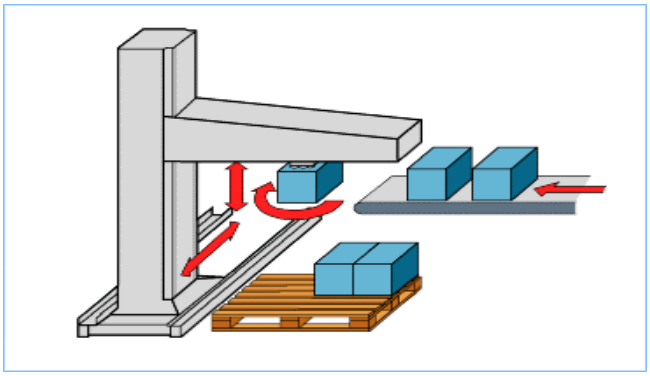


Figura. 7.18. CAN en sistemas de producción

Es prácticamente un requisito el uso de protocolos de alto nivel en este tipo de sistemas ya que se tiene una gran demanda para conexiones plug-and-play lo que representa, además de su gran confiabilidad, una alta intercambiabilidad e interoperabilidad de dispositivos.

Figura. 7.19. CAN en el área de ensamblaje de piezas automotrices

Figura. 7.20. CAN en sistemas (maquinaria) de empaquetamiento

Figura. 7.21. CAN en sistemas (maquinaria) de almacenamiento

Una de las aplicaciones es en la industria de ensamblaje, CAN se ha utilizado en gran cantidad en las industrias de ensamblaje de piezas automotrices (figura 7.19.), este tipo de sistemas necesitan altas precisiones, velocidades y requieren de sistemas confiables para una producción eficiente. CAN (CANopen y DeviceNet) proporciona facilidad para obtener este tipo de precisión.

Existen muchas otras aplicaciones en el campo de la industria y los procesos de producción que éstas implican. Otros ejemplos pueden ser industrias embotelladoras de pintura, sistemas de almacenamiento, sistemas de empaquetamiento, etc.

7.7 CAN EN AUTOMATIZACIÓN DE EDIFICIOS

El protocolo CAN en la automatización de edificios se utiliza para manejar aspectos como la calefacción, aire acondicionado, ventilación, puertas, etc .También se utiliza en el control de sistemas de alarma, sistemas anti-incendios, sistemas de control de iluminación, equipo de estudio que incluye control de audio y video, etc.

Se han incluido otras aplicaciones dentro del campo de automatización de edificios o edificios inteligentes sobre todo en países europeos, por ejemplo en los ascensores, gradas eléctricas, puertas eléctricas, control de equipos de dormitorio, refrigeradores, congeladores, etc.

7.8 SOFTWARES DE APLICACIÓN CAN

En el mercado existen diferentes programas de aplicación principalmente para protocolos de alto nivel basados en CAN.

Programas CANopen:

- CANopen Master Software and Configuration Tool
- CANopen Magic

- Micro CANopen
- proCANopen
- CANopenoe
- CANopen Configuration Studio
- canAnalyser
- CANopen EDS Editor

Softwares DeviceNet:

- RSNetWorks
- RSNetworks MD
- DeviceNet Monitor
- RSLinx

A continuación se detalla las características principales de algunos programas CanOpen y DeviceNet.

7.8.1 CANopen Master Software and Configuration Tool

El programa CANopen Master Software proporciona al usuario las funciones necesarias para controlar y manejar una red CANopen entera.

CANopen Master software and Configuration Tool contiene dos herramientas de aplicación: CANopen Master Software y VANopen Konfigurator.

CANopen Master Software comprende la inicialización, la supervisión, el control y la sincronización de una red CANopen.

Características de CANopen Master Software:

- NMT maestro o esclavo
- Administración SDO
- Señal SYNC servidor o cliente
- PDOs cíclicos, síncronos y asíncronos
- Manejo de errores

CANopen Konfigurator es una herramienta integrada en el software, la cual realiza la configuración de los esclavos en una red CANopen de una manera muy sencilla. El usuario tiene una interfaz gráfica para cada nodo en la red CANopen en dónde realizará la configuración de todos los nodos.

Características de CANopen Konfigurator:

- GUI o interfaz de usuario gráfica para realizar la configuración.
- Vista de árbol de los dispositivos conectados a la red CANopen.
- Nombres de dispositivos individuales.
- Activación o desactivación de la Guardia de Nodo (node guarding) o Latido (hearthbeat) para cada nodo.
- Mapeo automático de PDOs con valores por defecto.

- Los PDOs pueden tener nombres individuales y su scceso se realiza a travéz de dichos nombres.

7.8.2 PCANopenMagic

El software PCANopen MAGIC es fácil de usar y representa una gran utilidad para acceder y controlar los nodos en una red CANopen.

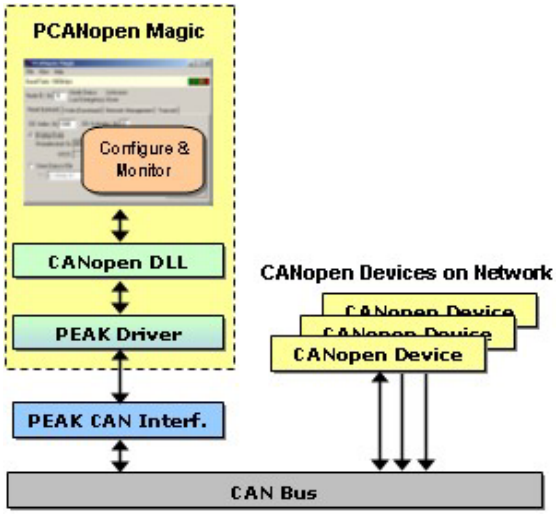


Figura. 7.22. Componentes del software PCANopen Magic

Permite accesos de lectura y escritura a los datos de proceso y a las variables de configuración de nodos CANopen.

Este software soporta múltiples interfaces CAN ya sea mediante el puerto PCI, ISA, USB, puerto paralelo o serial de un PC. Un solo nodo o todos los nodos pueden ponerse en los estados Operacional, Pre-operacional, detenidos o de reset.

Se pueden cargar datos en un nodo seleccionado manualmente o bajarlos desde un archivo, de igual forma cualquier dato de un nodo seleccionado puede ser leido por otro nodo y guardado en un archivo.

Permite tener una indicación del estado actual de cualquier nodo mediante los sistemas de guardia de nodo y heartbeat. Además escucha mensajes de emergencia enviados por cualquier nodo.

La carga de bus, errores y funciones de transmisión y recepción se indican mediante LEDs en la ventana principal del software e indican cuando PCANopen Magic está transmitiendo y recibiendo.

Permite configurar las entradas del diccionario de objetos de una menera sencilla y práctica, de igual forma permite configurar los parámetros de los SDOs y los PDOs y el mapeo de sus respectivos parámetros.

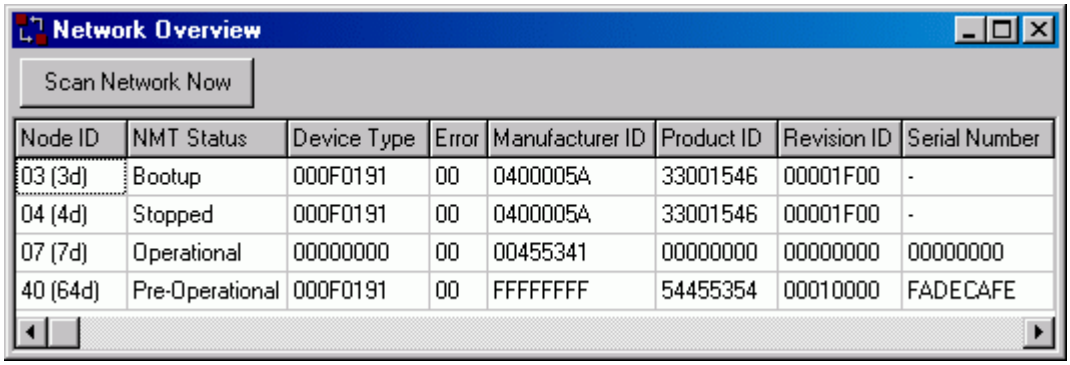


Figura. 7.23. Entradas obligatorias del diccionario de objetos PCANopen Magic

7.9 DISPOSITIVOS QUE FORMAN PARTE DE UNA APLICACIÓN CAN

Como ya se ha mencionado existen muchos dispositivos que pueden formar parte de una red CAN tales como PCs, PLCs, sensores, arrancadores, HMIs, variadores, etc.

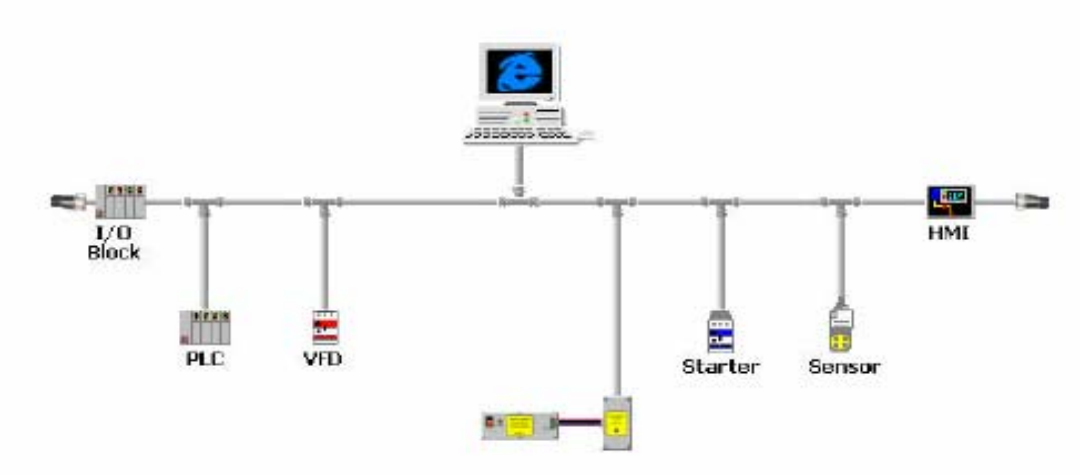


Figura. 7.24. Red CAN

Algunos de estos dispositivos se muestran en las figuras a continuación:

Figura. 7.25. Controladores Lógicos Programables

Figura. 7.26. Módulos de entrada/salida

Figura. 7.27. Variadores de frecuencia o velocidad

Figura. 7.28. Arrancadores

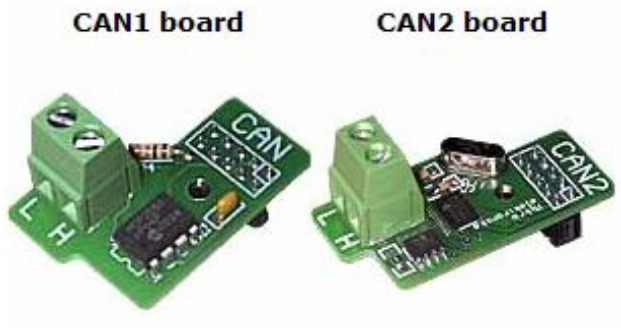


Figura. 7.29. Tarjetas de interfaz CAN

7.10 COMPARACIÓN DEL PROTOCOLO CAN CON OTROS PROTOCOLOS

La tarea de comprar redes que funcionen bajo diferentes protocolos resulta muy ambiciosa y ambigua debido a los numerosos parámetros que se necesitan para especificar una red y las complejas interdependencias que existen entre dichos parámetros. Siempre depende de la aplicación y sus características para la selección de un protocolo.

Por lo general se debe analizar los requerimientos de la aplicación a desarrollar y posteriormente elaborar una lista de parámetros relacionada a dicha aplicación, la cual describirá las características de los protocolos bajo discusión.

Una vez elaborados los requerimientos de la aplicación y los parámetros que implica, se consigue visualizar el protocolo más conveniente. Pero en la mayoría de casos, se convierte en una difícil tarea ya que cada protocolo consigue un alto grado de conveniencia en diferentes aspectos, por lo cual se debe analizar el parámetro más relevante dentro de la red que se va a desarrollar. Como por lo general las redes no dependen de un solo parámetro sino de un conjunto de parámetros relacionados entre sí, el análisis se vuelve más complejo.

A continuación se muestran algunos de los posibles requerimientos relacionados a las diferentes aplicaciones:

Características orientadas al mercado

- Área de Aplicación
- Volumen de Mercado
- Costo
- Proyección a futuro

- Disponibilidad de elementos
- Cantidad de instalaciones ya existentes

Capacidad del Sistema

- Eficiencia del Protocolo
- Protocolos de alto nivel disponibles, herramientas de hardware y software
- Características del tipo de mensajes
- Transparencia en la codificación de datos
- Tiempo de Latencia
- Tolerancia a fallas
- Características de Acceso al Medio
- Topología de Red
- Extensión máxima de red
- Velocidad de transmisión
- Cantidad de datos por mensaje
- Número de repetidores por segmento

Características Técnicas

- Medio de Red
- Tipo y número de conductores
- Poder de alimentación
- Tasa de detección de errores
- Consumo de potencia
- Rango de temperatura
- Codificación de bit
- Frecuencia de oscilador
- Técnicas de sincronización
- Tipos de protección

Además de estos parámetros se debe tomar en cuenta otros relacionados a las características de producción, así como las técnicas de prueba y monitoreo existentes. Esto se relaciona a la metodología y herramientas existentes para el desarrollo, pruena de producción, diagnóstico, monitoreo, etc.

En la tabla. 7.1. se muestra un ejemplo de una comparación entre algunos protocolos como son Bitbus, INTERBUS-S, Profibus, FIP, LON y el protocolo CAN.

	Bitbus	**CAN**	**Interbus**	**Profibus**	**FIP**	**LON**
Área de Aplicación	Control. Industrial	Vehículos Control Industrial	Control. Industrial	Control. Industrial	Control. Industrial	Domótia
Compañía	Intel	Bosch	Phoenix Contact	Profibus Consortium	Schneider	Echelon
Ciudad	USA	Alemania	Alemania	Alemania	Francia	USA
Topología	Bus	Bus	Anillo	Bus	Bus	Bus
Num. de nodos	28	-	64	32	-	64
Long. (mt) a máx vel. (Kbps)	1200 (62.5)	40 (1000)	400 (300)	1200 (93.75)	25 (1000)	500 (1250)
Máx. Velocidad (Kbps)	62.5	1000	300	95	1000	1250
Acceso al Medio	Maestro-Esclavo	Multi-Maestro	Slot de Tiempo	Token + Maestro-Esclavo	Maestro - Esclavo	Multi-Maestro
Control Descentralizado	No	Si	No	Parcial	No	Si
Núm. Bytes de Datos	128	8	128	246	128	246

Tabla. 7.1. Comparación de Protocolos

CAPITULO 8

CONCLUSIONES Y RECOMENDACIONES

8.1 CONCLUSIONES

Una vez finalizado el análisis e investigación de las características fundamentales del Protocolo CAN, se ha llegado a las siguientes conclusiones:

- La arquitectura multi-maestra de CAN, su tipo de acceso al medio (CSMA/CA) y su sistema de prioridad de mensajes, presentan una gran ventaja en la comunicación adecuada entre nodos, ya que cualquiera de ellos puede transmitir en el momento requerido, mientras el bus se encuentre libre, es decir, no es necesaria la participación de un dispositivo central. Además, anula las colisiones mediante un sistema de identificadores de mensajes, el cual asegura que el nodo con mensajes de mayor prioridad gane el acceso al bus.

- CAN es un protocolo que proporciona alta seguridad y confiabilidad en la transmisión de datos, debido a sus diversos mecanismos de detección y corrección de errores, así como su capacidad de reconocimiento y desactivación automática de nodos defectuosos.

- La transmisión de datos en una forma balanceada, es decir, mediante dos líneas de comunicación CAN_H y CAN_L, y la representación de bits mediante un voltaje diferencial permite que CAN tenga una alta inmunidad al ruido e interferencia electromagnética. Por esta razón este sistema de comunicación alcanza grandes distancias y se puede instalar en ambientes con altas interferencias.

- La disponibilidad de protocolos de alto nivel como SAE J1939, CAL, CANopen y DeviceNet, aseguran mayor confiabilidad ya que brindan excelentes características de administración de red, control y monitoreo de nodos. Además, proporcionan mayor intercambiabilidad e interoperabilidad debido a su sistema de perfiles de dispositivos estandarizados.

- El método de relleno de bits propio de CAN, además de ser un mecanismo para detección de errores, asegura que los nodos continúen perfectamente sincronizados incluso cuando se transmiten varios bits del mismo nivel insertando un bit de valor complementario, el cual sirve como flanco para resincronización.

- El sistema de contadores de error, transmisores y receptores permite ubicar a cada nodo en uno de los tres estados de error (activo, pasivo o fuera de bus), lo cual asegura la desactivación automática de nodos que cometan errores constantemente excluyéndolos totalmente de la comunicación dentro de la red.

- El mecanismo de detección de errores CRC tiene una distancia Hamming de 6, lo cual brinda mayor seguridad e integridad en la transmisión de datos.

- En el mercado existe una amplia gama de controladores o microcontroladores CAN. Esto demuestra la importancia y gran proyección a futuro del protocolo, ya que los fabricantes más importantes de dichos dispositivos en la actualidad como: Microchip, Intel, Motorola, entre otros; incluyen interfaces CAN.

- El protocolo CAN brinda una gran flexibilidad con respecto al tipo de implementación, ya que presenta diferentes alternativas en el uso de interfaces, arquitectura de controladores y diseño de nodos.

- El hecho de que CAN transmita un campo de datos relativamente pequeño (máximo 8 bytes) dentro de la trama, contribuye a la obtención de mayores velocidades de transmisión (hasta 1 Mbps).

- Los diferentes tipos de tramas CAN (remota, datos, error y sobrecarga) dan mayor funcionalidad al protocolo. Los campos de bits incluidos en dichas tramas se elaboran de acuerdo a las necesidades del nodo (transmisión, requerimiento, estado de error y carga de bus).

- El protocolo CAN ha demostrado ser de gran utilidad en sistemas de bus de campo, prueba de ello es el amplio campo de aplicaciones en el que se ha desarrollado, desde pequeñas máquinas estacionarias hasta complejos sistemas de producción y control de procesos. El uso de CAN en los sistemas de comunicación dentro de los vehículos ha tenido un gran impacto, tanto es así, que inclusive se ha creado la norma de que los vehículos trabajen bajo CAN a partir del año 2008.

8.2 RECOMENDACIONES

De la experiencia obtenida luego de desarrollar el análisis, se puede citar las siguientes recomendaciones con respecto a la implementación de redes CAN:

- Si bien es cierto, los protocolos de alto nivel proporcionan mejores características de administración de red, seguridad, control y monitoreo de nodos, pero su aplicación es aconsejable en redes destinadas a tener un largo tiempo de vida, por lo cual, para fines didácticos, de prueba de funcionamiento y demostración del protocolo; se recomienda utilizar una implementación pura de capa 2 por su menor tiempo de vida. Además, los gastos en una implementación con protocolos de alto nivel no serían justificables para este tipo de fines.

- El uso de una arquitectura de controladores FullCAN es aconsejable porque representa un menor tiempo de programación al eliminar la necesidad de filtros de mensajes, ya que, cada mensaje es asignado a su propio buffer transmisor o receptor. Sin embargo, se debe tener la precaución de que el número de mensajes a transmitirse en la red, no exceda al número de buffers disponibles en el controlador utilizado; de lo contrario, es recomendable el uso de una arquitectura BasicCAN.

- Cuando se implementa tipos de nodos con microcontrolador y la aplicación requiere de procesamientos de datos extensos, se recomienda utilizar un controlador Stand Alone para reducir la carga al microcontrolador. De la misma manera, cuando se utiliza un nodo con tarjeta de interfaz y sistema Host, se recomienda utilizar una tarjeta activa (microcontrolador incluido), para reducir la carga del sistema Host y diferenciar claramente los procesos de comunicación y aplicación.

- Se recomienda utilizar el método balanceado de cuando se requiera aplicaciones a distancias considerables, de lo contrario CAN presenta la alternativa de transmitir mediante una sola línea de forma desbalanceada. Para esto se debe tener la precaución de configurar correctamente las salidas del microcontrolador utilizado.

- Es recomendable utilizar el software CANTime para la asignación de parámetros de tiempos de bit ya que es bastante confiable y se pueden evitar posibles fallas de cálculo. Si los requerimientos son máximos, es decir, altas velocidades de transmisión y largas longitudes de bus es recomendable utilizar osciladores de cuarzo. De lo contrario se pueden utilizar osciladores cerámicos.

- Se podría realizar una implementación de redes CAN a nivel de sensores y actuadores en la ESPE para actualizar sistemas como el CIM, inclusive adquiriendo dispositivos estandarizados CANopen o DeviceNet ya que sería una red destinada a tener un largo tiempo de vida.

REFERENCIAS BIBLIOGRÁFICAS

- ETSCHBERGER, Konrad, **Controller Area Network**, Primera Edición, Ed. IXXAT Automation GmbH, Alemania 2005, Pgs 1-408.

- PFEIFFER, Olaf, AYRE, Andrew, KEYDEL, Christian, **Embedded Networking with CAN and CANopen, Primera Edición,** ED. RTC Books, USA 2003, Pgs 1-507.

- JAWRENZ, Wolfhard, **CAN System Engineering From Theory to Practical Applications**, Primera Edición, Ed. Springer, Alemania 1997, Pgs 1-465.

- FARSI, Mohammad, **CANopen Implementation: Applications to Industrial Networks,** Primera Edición, USA 2001, Pgs 1-225.

- www.cia.com, CAN in Automation

- www.can.bosch.com, Fundamentos del Protocolo CAN

- www.CANopen.com, Descripción del protocolo CANopen

- www.ixxat.com, Dispositivos CAN e Información general del protocolo

- www.fieldbus.com, CAN como un bus de campo

- www.ni.com, Dispositivos CAN Nacional Instruments

- www.amazon.com, Libros dedicados al protocolo CAN

- www.Kvaser.ser.com, Información General
- www.CANopensolutions.com, Implementaciones CANopen
- www.canAnalyser.com, Software CAN
- www.crucial.com, Información general

ANEXOS

ANEXO 1

DESCRIPCIÓN DE PINES Y CARACTERÍSTICAS PRINCIPALES DE CONTROLADORES CAN STAND ALONE PHILIPS SJA1000, INTEL 82527, ININEON 82C900

CONTROLADOR PHILIPS SJA1000

Descripción de pines:

5 PINNING

SYMBOL	PIN	DESCRIPTION
AD7 to AD0	2, 1, 28 to 23	multiplexed address/data bus
ALE/AS	3	ALE input signal (Intel mode), AS input signal (Motorola mode)
$\overline{CS}$	4	chip select input, LOW level allows access to the SJA1000
$\overline{RD}$/E	5	$\overline{RD}$ signal (Intel mode) or E enable signal (Motorola mode) from the microcontroller
$\overline{WR}$	6	$\overline{WR}$ signal (Intel mode) or RD/$\overline{WR}$ signal (Motorola mode) from the microcontroller
CLKOUT	7	clock output signal produced by the SJA1000 for the microcontroller; the clock signal is derived from the built-in oscillator via the programmable divider; the clock off bit within the clock divider register allows this pin to disable
V_{SS1}	8	ground for logic circuits
XTAL1	9	input to the oscillator amplifier; external oscillator signal is input via this pin; note 1
XTAL2	10	output from the oscillator amplifier; the output must be left open-circuit when an external oscillator signal is used; note 1
MODE	11	mode select input 1 = selects Intel mode 0 = selects Motorola mode
V_{DD3}	12	5 V supply for output driver
TX0	13	output from the CAN output driver 0 to the physical bus line
TX1	14	output from the CAN output driver 1 to the physical bus line
V_{SS3}	15	ground for output driver
$\overline{INT}$	16	interrupt output, used to interrupt the microcontroller; $\overline{INT}$ is active LOW if any bit of the internal interrupt register is set; $\overline{INT}$ is an open-drain output and is designed to be a wired-OR with other $\overline{INT}$ outputs within the system; a LOW level on this pin will reactivate the IC from sleep mode
$\overline{RST}$	17	reset input, used to reset the CAN interface (active LOW); automatic power-on reset can be obtained by connecting $\overline{RST}$ via a capacitor to V_{SS} and a resistor to V_{DD} (e.g. C = 1 μF; R = 50 kΩ)
V_{DD2}	18	5 V supply for input comparator
RX0, RX1	19, 20	input from the physical CAN-bus line to the input comparator of the SJA1000; a dominant level will wake up the SJA1000 if sleeping; a dominant level is read, if RX1 is higher than RX0 and vice versa for the recessive level; if the CBP bit (see Table 49) is set in the clock divider register, the CAN input comparator is bypassed to achieve lower internal delays if an external transceiver circuitry is connected to the SJA1000; in this case only RX0 is active; HIGH is interpreted as recessive level and LOW is interpreted as dominant level
V_{SS2}	21	ground for input comparator
V_{DD1}	22	5 V supply for logic circuits

Note

1. XTAL1 and XTAL2 pins should be connected to V_{SS1} via 15 pF capacitors.

Diseño de Memoria del Controlador:

CAN ADDRESS	SEGMENT	OPERATING MODE		RESET MODE	
		READ	WRITE	READ	WRITE
0	control	control	control	control	control
1		(FFH)	command	(FFH)	command
2		status	–	status	–
3		interrupt	–	interrupt	–
4		(FFH)	–	acceptance code	acceptance code
5		(FFH)	–	acceptance mask	acceptance mask
6		(FFH)	–	bus timing 0	bus timing 0
7		(FFH)	–	bus timing 1	bus timing 1
8		(FFH)	–	output control	output control
9		test	test; note 2	test	test; note 2
10	transmit buffer	identifier (10 to 3)	identifier (10 to 3)	(FFH)	–
11		identifier (2 to 0), RTR and DLC	identifier (2 to 0), RTR and DLC	(FFH)	–
12		data byte 1	data byte 1	(FFH)	–
13		data byte 2	data byte 2	(FFH)	–
14		data byte 3	data byte 3	(FFH)	–
15		data byte 4	data byte 4	(FFH)	–
16		data byte 5	data byte 5	(FFH)	–
17		data byte 6	data byte 6	(FFH)	–
18		data byte 7	data byte 7	(FFH)	–
19		data byte 8	data byte 8	(FFH)	–
20	receive buffer	identifier (10 to 3)	identifier (10 to 3)	identifier (10 to 3)	identifier (10 to 3)
21		identifier (2 to 0), RTR and DLC	identifier (2 to 0), RTR and DLC	identifier (2 to 0), RTR and DLC	identifier (2 to 0), RTR and DLC
22		data byte 1	data byte 1	data byte 1	data byte 1
23		data byte 2	data byte 2	data byte 2	data byte 2
24		data byte 3	data byte 3	data byte 3	data byte 3
25		data byte 4	data byte 4	data byte 4	data byte 4
26		data byte 5	data byte 5	data byte 5	data byte 5
27		data byte 6	data byte 6	data byte 6	data byte 6
28		data byte 7	data byte 7	data byte 7	data byte 7
29		data byte 8	data byte 8	data byte 8	data byte 8
30		(FFH)	–	(FFH)	–
31		clock divider	clock divider; note 3	clock divider	clock divider

Registros de Control y Estado:

REGISTER	BIT	SYMBOL	NAME	VALUE	
				RESET BY HARDWARE	SETTING BIT CR.0 BY SOFTWARE OR DUE TO BUS-OFF
Control	CR.7	–	reserved	0	0
	CR.6	–	reserved	X	X
	CR.5	–	reserved	1	1
	CR.4	OIE	Overrun Interrupt Enable	X	X
	CR.3	EIE	Error Interrupt Enable	X	X
	CR.2	TIE	Transmit Interrupt Enable	X	X
	CR.1	RIE	Receive Interrupt Enable	X	X
	CR.0	RR	Reset Request	1 (reset mode)	1 (reset mode)
Command	CMR.7	–	reserved	note 3	note 3
	CMR.6	–	reserved		
	CMR.5	–	reserved		
	CMR.4	GTS	Go To Sleep		
	CMR.3	CDO	Clear Data Overrun		
	CMR.2	RRB	Release Receive Buffer		
	CMR.1	AT	Abort Transmission		
	CMR.0	TR	Transmission Request		
Status	SR.7	BS	Bus Status	0 (bus-on)	X
	SR.6	ES	Error Status	0 (ok)	X
	SR.5	TS	Transmit Status	0 (idle)	0 (idle)
	SR.4	RS	Receive Status	0 (idle)	0 (idle)
	SR.3	TCS	Transmission Complete Status	1 (complete)	X
	SR.2	TBS	Transmit Buffer Status	1 (released)	1 (released)
	SR.1	DOS	Data Overrun Status	0 (absent)	0 (absent)
	SR.0	RBS	Receive Buffer Status	0 (empty)	0 (empty)
Interrupt	IR.7	–	reserved	1	1
	IR.6	–	reserved	1	1
	IR.5	–	reserved	1	1
	IR.4	WUI	Wake-Up Interrupt	0 (reset)	0 (reset)
	IR.3	DOI	Data Overrun Interrupt	0 (reset)	0 (reset)
	IR.2	EI	Error Interrupt	0 (reset)	X; note 4
	IR.1	TI	Transmit Interrupt	0 (reset)	0 (reset)
	IR.0	RI	Receive Interrupt	0 (reset)	0 (reset)

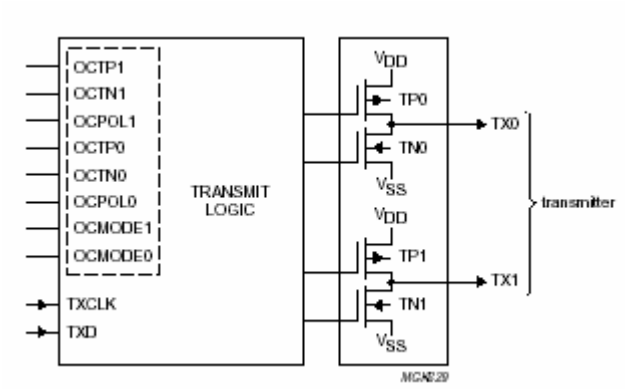
Registros de Control y Estado (continuación):

REGISTER	BIT	SYMBOL	NAME	VALUE	
				RESET BY HARDWARE	SETTING BIT CR.0 BY SOFTWARE OR DUE TO BUS-OFF
Acceptance code	AC.7 to 0	AC	Acceptance Code	X	X
Acceptance mask	AM.7 to 0	AM	Acceptance Mask	X	X
Bus timing 0	BTR0.7	SJW.1	Synchronization Jump Width 1	X	X
	BTR0.6	SJW.0	Synchronization Jump Width 0	X	X
	BTR0.5	BRP.5	Baud Rate Prescaler 5	X	X
	BTR0.4	BRP.4	Baud Rate Prescaler 4	X	X
	BTR0.3	BRP.3	Baud Rate Prescaler 3	X	X
	BTR0.2	BRP.2	Baud Rate Prescaler 2	X	X
	BTR0.1	BRP.1	Baud Rate Prescaler 1	X	X
	BTR0.0	BRP.0	Baud Rate Prescaler 0	X	X
Bus timing 1	BTR1.7	SAM	Sampling	X	X
	BTR1.6	TSEG2.2	Time Segment 2.2	X	X
	BTR1.5	TSEG2.1	Time Segment 2.1	X	X
	BTR1.4	TSEG2.0	Time Segment 2.0	X	X
	BTR1.3	TSEG1.3	Time Segment 1.3	X	X
	BTR1.2	TSEG1.2	Time Segment 1.2	X	X
	BTR1.1	TSEG1.1	Time Segment 1.1	X	X
	BTR1.0	TSEG1.0	Time Segment 1.0	X	X
Output control	OC.7	OCTP1	Output Control Transistor P1	X	X
	OC.6	OCTN1	Output Control Transistor N1	X	X
	OC.5	OCPOL1	Output Control Polarity 1	X	X
	OC.4	OCTP0	Output Control Transistor P0	X	X
	OC.3	OCTN0	Output Control Transistor N0	X	X
	OC.2	OCPOL0	Output Control Polarity 0	X	X
	OC.1	OCMODE1	Output Control Mode 1	X	X
	OC.0	OCMODE0	Output Control Mode 0	X	X
Transmit buffer	–	TXB	Transmit Buffer	X	X
Receive buffer	–	RXB	Receive Buffer	X; note 5	X; note 5
Clock divider	–	CDR	Clock Divider Register	00000000 (Intel); 00000101 (Motorola)	X

Formato de Buffers:

CAN ADDRESS	FIELD	NAME	BITS							
			7	6	5	4	3	2	1	0
10	descriptor	identifier byte 1	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5	ID.4	ID.3
11		identifier byte 2	ID.2	ID.1	ID.0	RTR	DLC.3	DLC.2	DLC.1	DLC.0
12	data	TX data 1	transmit data byte 1							
13		TX data 2	transmit data byte 2							
14		TX data 3	transmit data byte 3							
15		TX data 4	transmit data byte 4							
16		TX data 5	transmit data byte 5							
17		TX data 6	transmit data byte 6							
18		TX data 7	transmit data byte 7							
19		TX data 8	transmit data byte 8							

Transceiver Interno:



Configuración del tipo de salida:

OCMODE1	OCMODE0	DESCRIPTION
0	0	bi-phase output mode
0	1	test output mode; note 1
1	0	normal output mode
1	1	clock output mode

CONTROLADOR INTEL 82527

Descripción de pines:

Pin	Description
V_{SS1}	Ground (0V) connection must be shorted externally to a V_{SS} board plane to provide digital ground.
V_{SS2}	Ground (0V) connection must be shorted externally to a V_{SS} board plane to provide ground for analog comparator.
V_{CC}	Power connection must be shorted externally to +5V DC to provide power to the entire chip.
XTAL1	Input for an external clock. XTAL1 (along with XTAL2) are the crystal connection to an internal oscillator.
XTAL2	Push-pull output from the internal oscillator. XTAL2 and XTAL1 are the crystal connections to an internal oscillator. If an external oscillator is used, XTAL2 must be floated or not be connected. XTAL2 must not be used as a clock output to drive other CPUs.
CLKOUT	Programmable clock output. This push-pull output may be used to drive the oscillator of the CPU.
RESET #	Warm Reset: (V_{CC} remains valid while RESET # is asserted), Reset # must be driven to a low level for 1 ns minimum. Cold Reset: (V_{CC} is driven to a valid level while Reset # is asserted), Reset # must be driven low for 1 ns minimum measured from a valid V_{CC} level. No falling edge on the Reset pin is required during a cold reset event.
CS #	A low level on this pin enables the CPU to access the 82527.
INT # or (V_{CC}/2)	The interrupt pin is an open collector output (requires external pullup resistor) to the CPU. V_{CC}/2 is the power supply for the ISO low speed physical layer. The function of this pin is determined by the MUX bit in the CPU Interface Register (Address 02H) when the DcR1 bit (Address 2FH) is set: when MUX = 1 and DcR1 = 1: then pin 24 = V_{CC}/2, pin 11 = INT # when MUX = 0: then pin 24 = INT #
RX0 RX1	Inputs from the CAN bus line(s) to the input comparator. A recessive level is read when RX0 > RX1. A dominant level is read when RX1 > RX0. When the CoBy bit (Bus Configuration register) is programmed as a "1", the input comparator is bypassed and RX0 is the CAN bus line input.
TX0 TX1	Serial push-pull data output to the CAN bus line. During a recessive bit, TX0 is high and TX1 is low. During a dominant bit, TX0 is low and TX1 is high. TX0/TX1 suggestion: Unlike the Intel 82526, the 82527 TX0 and TX1 output drivers can not be individually programmed to transmit either recessive or dominant bits; this is fixed as described in the TX0/TX1 definition. If 82527 and 82526 devices are not communicating on a CAN bus, the problem may be due to TX0/TX1 configuration differences. Reversing the TX0/TX1 connections for either device may allow these devices to communicate.
Ports1/2	Port1 and Port2 pins are weakly held high until the Port configuration registers have been written (locations 9FH and AFH respectively).

Descripción de pines (continuación):

Pin	Description					
AD0–AD15	The functions of these pins are defined below.					
		8-Bit Intel Multiplexed	**8-Bit Non-Intel Multiplexed**	**16-Bit Intel Multiplexed**	**8-Bit Non-Multiplexed**	**Serial Interface**
	AD0	AD0	AD0	AD0	A0	ICP
	AD1	AD1	AD1	AD1	A1	CP
	AD2	AD2	AD2	AD2	A2	CSAS
	AD3	AD3	AD3	AD3	A3	STE
	AD4	AD4	AD4	AD4	A4	MOSI
	AD5	AD5	AD5	AD5	A5	Unused
	AD6	AD6	AD6	AD6	A6	SCLK
	AD7	AD7	AD7	AD7	A7	Unused
	AD8	Port 1.0	Port 1.0	AD8	D0	Port 1.0
	AD9	Port 1.1	Port 1.1	AD9	D1	Port 1.1
	AD10	Port 1.2	Port 1.2	AD10	D2	Port 1.2
	AD11	Port 1.3	Port 1.3	AD11	D3	Port 1.3
	AD12	Port 1.4	Port 1.4	AD12	D4	Port 1.4
	AD13	Port 1.5	Port 1.5	AD13	D5	Port 1.6
	AD14	Port 1.6	Port 1.6	AD14	D6	Port 1.6
	AD15	Port 1.7	Port 1.7	AD15	D7	Port 1.7
P2.0 P2.1 P2.2 P2.3 P2.4 P2.5 P2.6/ INT#	Port 2 function in all CPU interface modes. P2.6 is INT# when MUX = 1 in the CPU Interface register (02H).					
P2.7/ WRH#	P2.7 is WRH# in 16-bit multiplexed mode (Mode1).					
Mode0/ Mode1	These pins select one of the four parallel interfaces: **Mode1** **Mode0** 0 0* 8-bit multiplexed Intel 0 1 16-bit multiplexed Intel 1 0 8-bit multiplexed non-Intel 1 1 8-bit non-multiplexed * Note: If upon reset, Mode0 = Mode1 = 0, RD# = 0 and WR# = 0, then the serial interface mode is entered. Mode0 and Mode1 pins are internally connected to weak pulldowns. These pins will be pulled low during reset if unconnected. Following reset, these pins float.					
ALE/ AS	ALE used for Intel CPU Interface Modes 0 and 1. AS used for non-Intel modes. Except Mode 3, this pin must be tied high.					

Descripción de pines (continuación):

Pin	Description
RD #	RD # used for CPU Interface Modes 0 and 1.
E	E used for non-Intel modes. Except Mode 3 Asynchronous, this pin must be tied high.
WR # /WRL # R/W #	WR # used for Intel CPU Interface Modes 0 and 1. (WRL # function in Mode1, 16-bit Mode.) R/W # used for CPU Interface Mode3.
READY/ MISO	READY is an open-drain output to synchronize accesses from the CPU to the 82527 for CPU Interface Modes 0 and 1. MISO is the serial data output for the serial interface mode.
DSACK0 #	DSACK0 # is an open-drain output to synchronize accesses from the CPU to the 82527 for CPU Interface Mode3. DSACK0 # is often used as a pulldown output with a 3.3 kΩ pullup resistor and a 100 pF load capacitance. An open-drain output is used because many peripherals may be connected to the DSACK0 # line. The 82527 specifies a TCHKH timing (CS # high to DSACK0 # high) equal to 55 ns, however a 3.3 kΩ resistor will not sufficiently charge the line when DSACK0 # is floated by the 82527. To meet this timing, the 82527 has an active pullup that drives the DSACK0 # output until it is high, and then the pullup is turned off. Therefore, the pullup is only active for a short time.

Mapa de Memoria:

00H	Control Register
01H	Status Register
02H	CPU Interface Reg.
03H	Reserved
04–05H	High Speed Read
06–07H	Global Mask - Standard
08–0BH	Global Mask - Extended
0C–0FH	Message 15 Mask
10–1EH	**Message 1**
1FH	CLKOUT Register #
20–2EH	**Message 2**
2FH	Bus Config. Reg. #
30–3EH	**Message 3**
3FH	Bit Timing Reg. 0 #
40–4EH	**Message 4**
4FH	Bit Timing Reg. 1 #
50–5EH	**Message 5**
5FH	Interrupt Register
60–6EH	**Message 6**
6FH	Reserved
70H–7EH	**Message 7**
7FH	Reserved
80–8EH	**Message 8**
8FH	Reserved
90–9EH	**Message 9**
9FH	P1CONF #
A0–AEH	**Message 10**
AFH	P2CONF #
B0–BEH	**Message 11**
BFH	P1IN
C0–CEH	**Message 12**
CFH	P2IN
D0–DEH	**Message 13**
DFH	P1OUT
E0–EEH	**Message 14**
EFH	P2OUT
F0–FEH	**Message 15**
FFH	Serial Reset Address

CONTROLADOR INFINEON 82C900

Descripción de pines:

Pin Definition

Table 1 Pin Definitions and Functions

Symbol	Pin Number	I/O [1)]	Function
RXDCA	1	I	**Receiver Input of CAN Node A** Receiver input of CAN node A, connected to the associated CAN bus via a transceiver device.
TXDCA	2	O	**Transmitter Output of CAN Node A** TXDCA delivers the output signal of CAN node A. The signal level has to be adapted to the physical layer of the CAN bus via a transceiver device.
RXDCB	28	I	**Receiver Input of CAN Node B** Receiver input of CAN node B, connected to the associated CAN bus via a transceiver device.

Descripción de pines (continuación):

Table 1 Pin Definitions and Functions (cont'd)

Symbol	Pin Number	I/O [1]	Function
TXDCB	27	O	**Transmitter Output of CAN Node B** TXDCB delivers the output signal of CAN node B. The signal level has to be adapted to the physical layer of the CAN bus via a transceiver device.
$\overline{\text{RESET}}$	3	I	**Reset** A low level on this pin resets the device.
RDY	24	O	**Ready Signal** Output signal indicating that the standalone device is ready for data transfer.
CTRL0	9	I/O	**Control 0** MODE0=0: **Chip Select $\overline{\text{CS}}$** Input used as Chip Select for the device. MODE0=1: **Select Slave $\overline{\text{SLS}}$** MODE1=0: Input used to enable SSC action when active. MODE1=1: Output used to select a slave when active.
CTRL1	20	I/O	**Control 1** MODE0=0: **Address Latch Enable or Address Strobe, ALE or AS** Input used for latching the address from the multiplexed address/data bus. MODE0=1: **Serial Channel Clock SCLK** Input/output of the SSC clock. MODE1=0: Clock input MODE1=1: Clock output
CTRL2	10	I/O	**Control 2** MODE0=0: **Write or Read/Write, $\overline{\text{WR}}$ or R/$\overline{\text{W}}$** MODE1=0: Input used as write signal $\overline{\text{WR}}$ MODE1=1: R/$\overline{\text{W}}$=0: Data transfer direction = write MODE1=1: R/$\overline{\text{W}}$=1: Data transfer direction = read MODE0=1: **Master Transmit Slave Receive MTSR** MODE1=0: Serial data input MODE1=1: Serial data output

Descripción de pines (continuación):

Table 1 Pin Definitions and Functions (cont'd)

Symbol	Pin Number	I/O [1]	Function
CTRL3	19	I/O	**Control 3** MODE0=0: **Read or Read/Write Enable, $\overline{RD}$ or E** MODE1=0: Input used as read signal $\overline{RD}$ MODE1=0: Read/write enable MODE0=1: **Master Receive Slave Transmit MRST** MODE1=0: Serial data output MODE1=1: Serial data input
P7 **P6** **P5** **P4** **P3** **P2** **P1** **P0**	15 14 16 13 17 12 18 11	I/O	**Parallel Bus** MODE0=0: **8-bit Address/ Data Bus AD[7:0]** Address and data bus AD7..AD0 in 8-bit multiplexed modes. MODE0=1: **8-bit parallel I/O Port IO[7:0]** Programmable 8-bit general purpose I/O-port IO7..IO0.
$\overline{OUT0}$ [2]	6	O	**Output Line 0** The logic 0 level at this pin indicates an interrupt request to the external host device if selected as interrupt output. The interrupt line will be active if there is a new pending interrupt request for interrupt node 0 (according to register GLOBCTR). If selected as clock output, the functionality is defined by register CLKCTR.
$\overline{OUT1}$	23	O, open drain	**Output Line 1** The logic 0 level at this pin indicates an interrupt request to the external host device. The interrupt line will be active if there is a new pending interrupt request for interrupt node 1 (according to register GLOBCTR).
MODE0 [3]	26	I/O, open drain	**Interface Selection** Pin MODE0 selects whether the on-chip SSC or an 8-bit multiplexed bus are used to access the TwinCAN device. MODE0=0: 8-bit multiplexed address/data bus MODE0=1: on-chip SSC After registering the initial state of MODE0 with the rising edge of the reset signal, the respective pin can be used as additional general purpose or special function I/O line according register IOMODE4.

Descripción de pines (continuación):

Table 1 Pin Definitions and Functions (cont'd)

Symbol	Pin Number	I/O [1]	Function
MODE1	25	I/O, open drain	**Interface Mode Selection** Pin MODE1 determines the access mode of the host device. MODE0=0: **8-bit multiplexed bus** MODE1=0: Infineon / Intel mode, ($\overline{RD}$, $\overline{WR}$) MODE1=1: Motorola mode, (R/$\overline{W}$, E) MODE0=1: **On-chip SSC** MODE1=0: SSC is slave, host device is master MODE1=1: SSC is master, external serial EEPROM is slave After registering the initial state of MODE1 with the rising edge of the reset signal, the respective pin can be used as additional general purpose or special function I/O line according register IOMODE4.
XTAL1	4	I	**XTAL1** Input of the inverting oscillator amplifier and input to the internal clock generation circuit. When the 82C900 device is provided with an external clock, XTAL1 should be driven while XTAL2 is left unconnected. Minimum and maximum high and low pulse width as well as rise/fall times specified in the AC characteristics must be respected.
XTAL2	5	O	**XTAL2** Output of the inverting oscillator amplifier.
V_{SS}	21, 8	0V	**Ground**, both pins must be connected.
V_{DD}	22, 7	+5V	**Power Supply**, both pins must be connected.

ANEXO 2

DESCRIPCIÓN DE PINES Y CARACTERÍSTICAS PRINCIPALES DE MICROCONTROLADORES CAN DALLAS DS80C390, PHILIPS P8xC592, ATMEL T89C51CC01

MICROCONTROLADOR DALLAS DS80C390

Descripción de pines:

PIN		NAME	FUNCTION
LQFP	PLCC		
8, 22, 40, 56	17, 32, 51, 68	V_{CC}	**+5V**
9, 25, 41, 57	1, 18, 35, 52	GND	**Digital Circuit Ground**
46	57	ALE	**Address Latch Enable, Output.** When the $\overline{MUX}$ pin is low, this pin outputs a clock to latch the external address LSB from the multiplexed address/data bus on Port 0. This signal is commonly connected to the latch enable of an external transparent latch. ALE has a pulse width of 1.5 XTAL1 cycles and a period of four XTAL1 cycles. When the $\overline{MUX}$ pin is high, the pin will toggle continuously if the ALEOFF bit is cleared. ALE is forced high when the device is in a reset condition or if the ALEOFF bit is set while the $\overline{MUX}$ pin is high.
45	56	$\overline{PSEN}$	**Program Store Enable, Output.** This signal is the chip enable for external ROM memory. $\overline{PSEN}$ provides an active-low pulse and is driven high when external ROM is not being accessed.
47	58	$\overline{EA}$	**External Access Enable, Input.** This pin must be wired to GND for proper operation.
26	36	$\overline{MUX}$	**Multiplex/Demultiplex Select, Input.** This pin selects if the address/data bus operates in multiplexed ($\overline{MUX}$ = 0) or demultiplexed ($\overline{MUX}$ = 1) mode.
2	11	RST	**Reset, Input.** The RST input pin contains a Schmitt voltage input to recognize external active-high reset inputs. The pin also employs an internal pulldown resistor to allow for a combination of wired-OR external reset sources. An RC circuit is not required for power-up, as the device provides this function internally.
3	12	$\overline{RSTOL}$	**Reset Output Low, Output.** This active-low signal is asserted: When the processor has entered reset through the RST pin, During crystal warmup period following power-on or stop mode, During a watchdog timer reset (2 cycles duration), During an oscillator failure (if OFDE = 1), Whenever $V_{CC} \leq V_{RST}$.
23	33	XTAL2	**XTAL1, XTAL2.** Crystal oscillator pins support fundamental mode, parallel resonant, and AT-cut crystals. XTAL1 is the input if an external clock source is used in place of a crystal. XTAL2 is the output of the crystal amplifier.
24	34	XTAL1	
55	67	AD0/D0	**AD0–7 (Port 0), I/O.** When the $\overline{MUX}$ pin is wired low, Port 0 is the multiplexed address/data bus. While ALE is high, the LSB of a memory address is presented. While ALE falls, the port transitions to a bidirectional data bus. When the $\overline{MUX}$ pin is wired high, Port 0 functions as the bidirectional data bus. Port 0 cannot be modified by software. The reset condition of Port 0 pins is high. No pullup resistors are needed.
54	66	AD1/D1	
53	65	AD2/D2	
52	64	AD3/D3	
51	63	AD4/D4	
50	62	AD5/D5	
49	61	AD6/D6	
48	59	AD7/D7	

Descripción de pines (continuació):

PIN		NAME	FUNCTION
LQFP	PLCC		
58–64, 1	2–8, 10	P1.0–P1.7	**Port 1, I/O.** Port 1 can function as an 8-bit bidirectional I/O port, the nonmultiplexed A0–A7 signals (when the $\overline{MUX}$ pin = 1), and as an alternate interface for internal resources. Setting the SP1EC bit relocates RXD1 and TXD1 to Port 5. The reset condition of Port 1 is all bits at logic 1 through a weak pullup. The logic 1 state also serves as an input mode, since external circuits writing to the port can overdrive the weak pullup. When software clears any port pin to 0, a strong pulldown is activated that remains on until either a 1 is written to the port pin or a reset occurs. Writing a 1 after the port has been at 0 activates a strong transition driver, followed by a weaker sustaining pullup. Once the momentary strong driver turns off, the port once again becomes the output (and input) high state.
			Port — **Alternate Function**
58	2	A0	P1.0 — T2 External I/O for Timer/Counter 2
59	3	A1	P1.1 — T2EX Timer/Counter 2 Capture/Reload Trigger
60	4	A2	P1.2 — RXD1 Serial Port 1 Input
61	5	A3	P1.3 — TXD1 Serial Port 1 Output
62	6	A4	P1.4 — INT2 External Interrupt 2 (Positive Edge Detect)
63	7	A5	P1.5 — $\overline{INT3}$ External Interrupt 3 (Negative Edge Detect)
64	8	A6	P1.6 — INT4 External Interrupt 4 (Positive Edge Detect)
1	10	A7	P1.7 — $\overline{INT5}$ External Interrupt 5 (Negative Edge Detect)
35	46	A8 (P2.0)	**A15–A8 (Port 2), Output.** Port 2 serves as the MSB for external addressing. The port automatically asserts the address MSB during external ROM and RAM access. Although the Port 2 SFR exists, the SFR value never appears on the pins (due to memory access). Therefore, accessing the Port 2 SFR is only useful for MOVX A, @Ri or MOVX @Ri, A instructions, which use the Port 2 SFR as the external address MSB.
36	47	A9 (P2.1)	
37	48	A10 (P2.2)	
38	49	A11 (P2.3)	
39	50	A12 (P2.4)	
42	53	A13 (P2.5)	
43	54	A14 (P2.6)	
44	55	A15 (P2.7)	
4–7, 10–13	13–16, 19–22	P3.0–P3.7	**Port 3, I/O.** Port 3 functions as an 8-bit bidirectional I/O port and as an alternate interface for several resources found on the traditional 8051. The reset condition of Port 1 is all bits at logic 1 through a weak pullup. The logic 1 state also serves as an input mode, since external circuits writing to the port can overdrive the weak pullup. When software clears any port pin to 0, the device activates a strong pulldown that remains on until either a 1 is written to the port pin or a reset occurs. Writing a 1 after the port has been at 0 activates a strong transition driver, followed by a weaker sustaining pullup. Once the momentary strong driver turns off, the port once again becomes the output (and input) high state.
			Port — **Alternate Function**
4	13		P3.0 — RXD0 Serial Port 0 Input
5	14		P3.1 — TXD0 Serial Port 0 Output
6	15		P3.2 — $\overline{INT0}$ External Interrupt 0
7	16		P3.3 — $\overline{INT1}$ External Interrupt 1
10	19		P3.4 — T0 Timer 0 External Input
11	20		P3.5 — T1/XCLK Timer 1 External Input/External Clock Output
12	21		P3.6 — $\overline{WR}$ External Data Memory Write Strobe
13	22		P3.7 — $\overline{RD}$ External Data Memory Read Strobe

Descripción de pines (continuación):

PIN		NAME	FUNCTION
LQFP	PLCC		
34–27	45, 44, 42–37	P4.0–P4.7	**Port 4, I/O.** Port 4 can function as an 8-bit, bidirectional I/O port, and as the source for external address and chip enable signals for program and data memory. Port pins are configured as I/O or memory signals via the P4CNT register. The reset condition of Port 1 is all bits at logic 1 via a weak pullup. The logic 1 state also serves as an input mode, since external circuits writing to the port can overdrive the weak pullup. When software clears any port pin to 0, the device activates a strong pulldown that remains on until either a 1 is written to the port pin or a reset occurs. Writing a 1 after the port has been at 0 will activate a strong transition driver, followed by a weaker sustaining pullup. Once the momentary strong driver turns off, the port once again becomes the output (and input) high state.
			Port — Alternate Function P4.0 — $\overline{CE0}$ Program Memory Chip Enable 0 P4.1 — $\overline{CE1}$ Program Memory Chip Enable 1 P4.2 — $\overline{CE2}$ Program Memory Chip Enable 2 P4.3 — $\overline{CE3}$ Program Memory Chip Enable 3 P4.4 — A16 Program/Data Memory Address 16 P4.5 — A17 Program/Data Memory Address 17 P4.6 — A18 Program/Data Memory Address 18 P4.7 — A19 Program/Data Memory Address 19
34 33 32 31 30 29 28 27	45 44 42 41 40 39 38 37		
21–14	31–27, 25–23	P5.0–P5.7	**Port 5, I/O.** Port 5 can function as an 8-bit, bidirectional I/O port, the CAN interface, or as peripheral enable signals. Setting the SP1EC bit will relocate the RXD1 and TXD1 functions to P5.3-P5.2 as described in the *High-Speed Microcontroller User's Guide: DS80C390 Supplement*. The reset condition of Port 1 is all bits at logic 1 via a weak pullup. The logic 1 state also serves as an input mode, since external circuits writing to the port can overdrive the weak pullup. When software clears any port pin to 0, the device activates a strong pulldown that remains on until either a 1 is written to the port pin or a reset occurs. Writing a 1 after the port has been at 0 will activate a strong transition driver, followed by a weaker sustaining pullup. Once the momentary strong driver turns off, the port once again becomes the output (and input) high state.
			Port — Alternate Function P5.0 — C0TX CAN0 Transmit Output P5.1 — C0RX CAN0 Receive Input P5.2 — C1RX CAN1 Receive Input (optional RXD1) P5.3 — C1TX CAN1 Transmit Output (optional TXD1) P5.4 — $\overline{PCE0}$ Peripheral Chip Enable 0 P5.5 — $\overline{PCE1}$ Peripheral Chip Enable 1 P5.6 — $\overline{PCE2}$ Peripheral Chip Enable 2 P5.7 — $\overline{PCE3}$ Peripheral Chip Enable 3
21 20 19 18 17 16 15 14	31 30 29 28 27 25 24 23		
	9, 26, 43, 60	N.C.	**Not Connected.** Reserved. These pins are reserved for use with future devices in this family and should not be connected.

MICROCONTROLADOR PHILIPS P8xC592

Descripción de pines:

SYMBOL	PIN		DESCRIPTION
	QFP44	PLCC44	
$\overline{RST}$	4	10	**Reset:** A Input to reset the P8xC591. It also provides a reset pulse as output when Timer T3 overflows.
P3.0to P3.7			**Port 3 (P3.0 to P3.7)**: 8-bit programmable I/O port lines; Port 3 can sink/source 4 LSTTL inputs. Port 3 pins serve alternate functions as follows:
P3.0/RXD	5	11	**RXD**: Serial input port for UART; **T2:** T2 event input
P3.1/TXD	7	13	**TXD**: Serial output port for UART; **RT2:** T2 timer reset signal. Rising edge triggered.
P3.2/$\overline{INT0}$/CMSR0	8	14	**$\overline{INT0}$**: External interrupt input 0; **CMSR0:** Compare and Set/Reset output for Timer T2.
P3.3/$\overline{INT1}$/ CMSR1	9	15	**$\overline{INT1}$**: External interrupt input 1; **CMSR1:** Compare and Set/Reset output for Timer T2.
P3.4/T0/CMSR2	10	16	**T0**: Timer 0 external interrupt input; **CMSR2:** Compare and Set/Reset output for Timer T2.
P3.5/T1/CMSR3	11	17	**T1**: Timer 1 external interrupt input; **CMSR3:** Compare and Set/Reset output for Timer T2.
P3.6/$\overline{WR}$	12	18	**$\overline{WR}$**: External Data Memory Write strobe;
P3.7/$\overline{RD}$	13	19	**$\overline{RD}$**: External Data Memory Read strobe. During reset, Port 3 will be asynchronously driven resistive HIGH. Port 3 has four modes selected on a per bit basis by writing to the P3M1 and P3M2 registers as follows: **P3M1.x P3M2.x Mode Description** 0 0 Pseudo-bidirectional (standard c51 configuration default) 0 1 Push-Pull 1 0 High impedance 1 1 Open drain
XTAL2	14	20	**Crystal pin 2:** output of the inverting amplifier that forms the oscillator. Left open-circuit when an external oscillator clock is used.
XTAL1	15	21	**Crystal pin 1:** input to the inverting amplifier that forms the oscillator, and input to the internal clock generator. Receives the external oscillator clock signal when an external oscillator is used.
V_{SS}	16	22	**Ground;** circuit ground potential.
V_{DD}	17	23	**Power supply;** power supply pin during normal operation and power reduction modes.

Descripción de pines (continuación):

SYMBOL	PIN		DESCRIPTION
	QFP44	PLCC44	
P2.0/A08 to P2.7/A15	18 to 25	24 to 31	**Port 2 (P2.0 to P2.7)**: 8-bit programmable I/O port lines; **A08 to A15**: High-order address byte for external memory. Alternate function: High-order address byte for external memory (A08-A15). Port 2 is also used to input the upper order address during EPROM programming and verification. A8 is on P2.0, A9 on P2.1, through A12 on P2.4. During reset, Port 2 will be asynchronously driven HIGH. Port 2 has four output modes selected on a per bit basis by writing to the P2M1 and P2M2 registers as follows: **P2M1.x P2M2.x Mode Description** 0 0 Pseudo-bidirectional (standard c51 configuration default) 0 1 Push-Pull 1 0 High impedance 1 1 Open drain
$\overline{PSEN}$	26	32	**Program Store Enable** output: read strobe to the external Program Memory via Ports 0 and 2. Is activated twice each machine cycle during fetches from external Program Memory. When executing out of external Program Memory two activations of $\overline{PSEN}$ are skipped during each access to external Data Memory. $\overline{PSEN}$ is not activated (remains HIGH) during no fetches from external Program Memory. $\overline{PSEN}$ can sink/source 8 LSTTL inputs. It can drive CMOS inputs without external pull-ups.
ALE/$\overline{PROG}$	27	33	**Address Latch Enable** output. Latches the low byte of the address during access of external memory in normal operation. It is activated every six oscillator periods except during an external Data Memory access. ALE can sink/source 8 LSTTL inputs. It can drive CMOS inputs without an external pull-up. To prohibit the toggling of ALE pin (RFI noise reduction) the bit A0 (SFR: AUXR.0) must be set by software; see Table 4. **$\overline{PROG}$**: the programming pulse input; alternative function for the P87C591.
$\overline{EA}$/V_{PP}	29	35	**External Access** input. If, during reset, $\overline{EA}$ is held at a TTL level HIGH the CPU executes out of the internal Program Memory. If, during reset, $\overline{EA}$ is held at a TTL level LOW the CPU executes out of external Program Memory via Port 0 and Port 2. $\overline{EA}$ is not allowed to float. $\overline{EA}$ is latched during reset and don't care after reset. **V_{PP}**: the programming supply voltage; alternative function for the P87C591.
P0.0/AD0 to P0.7/AD7	30 to 37	36 to 43	**Port 0**: 8-bit open-drain bidirectional I/O port. During reset, Port 0 is HIGH-Impedance (Tri-State). **AD7 to AD0**: Multiplexed Low-order address and Data bus for external memory. During these accesses internal pull-ups are activated. Port 0 can sink/source up to 8 LSTTL inputs.
AV_{ref+}	38	44	**Analog to Digital Conversion Reference Resistor:** High-end.
AV_{SS}	39	1	**Analog ground.**

Descripción de pines (continuación):

SYMBOL	PIN		DESCRIPTION
	QFP44	PLCC44	
P1.0 to P1.4 P1.5 to P1.7	40 to 44 1 to 3	2 to 6 7 to 9	**Port 1:** 8-bit I/O port with a user configurable output type. The operation of Port 1 pins as inputs or outputs depends upon the port configuration selected. Each port pin is configured independently. Port 1 also provides various special functions as described below:
P1.0	40	2	**RXDC:** CAN Receiver input line.
P1.1	41	3	**TXDC:** CAN Transmit output line. During reset, Port P1.0 and P1.1 will be asynchronously driven resistive HIGH, P1.2 to P1.7 is High-Impedance (Tri-state).
P1.2 to P1.4	42 to 44	4 to 6	**CT0I/INT2 / CT1I/INT3 / CT2I/INT4:** T2 Capture timer inputs or External Interrupt inputs. **ADC0 to ADC2:** Alternate function: Input channels to ADC.
P1.5 to P1.7	1 to 3	7 to 9	**ADC3 to ADC5:** Input channels to ADC:
P1.5	1	7	**CT3I/INT5:** T2 Capture timer input or External Interrupt inputs.
P1.6	2	8	**SCL:** Serial port clock line I^2C. Push-pull or pseudo bidirectional modes is not implemented at I^2C.
P1.7	3	9	**SDA:** Serial data clock line I^2C.Push-pull or pseudo bidirectional modes is not implemented at I^2C. Port 1 has four modes selected on a per bit basis by writing to the P1M1 and P1M2 registers as follows: **P1M1.x P1M2.x Mode Description** 0 0 Pseudo-bidirectional (standard c51 configuration default [2]) 0 1 1 0 Push-Pull [2] 1 1 High impedance Open drain Port 1 is also used to input the lower order address byte during EPROM programming and verification. A0 is on P1.0, etc.
$\overline{PWM0}$	6	12	**Pulse Width Modulation:** Output 0.
$\overline{PWM1}$	28	34	**Pulse Width Modulation:** Output 1.

MICROCONTROLADOR ATMEL T89C51CC01

Descripción de pines:

Pin Name	Type	Description
VSS	GND	Circuit ground
VCC		Supply Voltage
VAREF		Reference Voltage for ADC
VAVCC		Supply Voltage for ADC
VAGND		Reference Ground for ADC
P1.0:7	I/O	Port 1: Is an 8-bit bi-directional I/O port with internal pull-ups. Port 1 pins can be used for digital input/output or as analog inputs for the Analog Digital Converter (ADC). Port 1 pins that have 1's written to them are pulled high by the internal pull-up transistors and can be used as inputs in this state. As inputs, Port 1 pins that are being pulled low externally will be the source of current (I_{IL}, See section "Electrical Characteristic") because of the internal pull-ups. Port 1 pins are assigned to be used as analog inputs via the ADCCF register (in this case the internal pull-ups are disconnected). As a secondary digital function, port 1 contains the Timer 2 external trigger and clock input; the PCA external clock input and the PCA module I/O. P1.0/AN0/T2 Analog input channel 0, External clock input for Timer/counter2. P1.1/AN1/T2EX Analog input channel 1, Trigger input for Timer/counter2. P1.2/AN2/ECI Analog input channel 2, PCA external clock input. P1.3/AN3/CEX0 Analog input channel 3, PCA module 0 Entry of input/PWM output. P1.4/AN4/CEX1 Analog input channel 4, PCA module 1 Entry of input/PWM output. P1.5/AN5 Analog input channel 5, P1.6/AN6 Analog input channel 6, P1.7/AN7 Analog input channel 7, It can drive CMOS inputs without external pull-ups.
P2.0:1	I/O	Port 2: Is an 2-bit bi-directional I/O port with internal pull-ups. Port 2 pins that have 1's written to them are pulled high by the internal pull-ups and can be used as inputs in this state. As inputs, Port 2 pins that are being pulled low externally will be a source of current (IIL, on the datasheet) because of the internal pull-ups. In the T89C51CC02 Port 2 can sink or source 5mA. It can drive CMOS inputs without external pull-ups.

Descripción de pines (continuación):

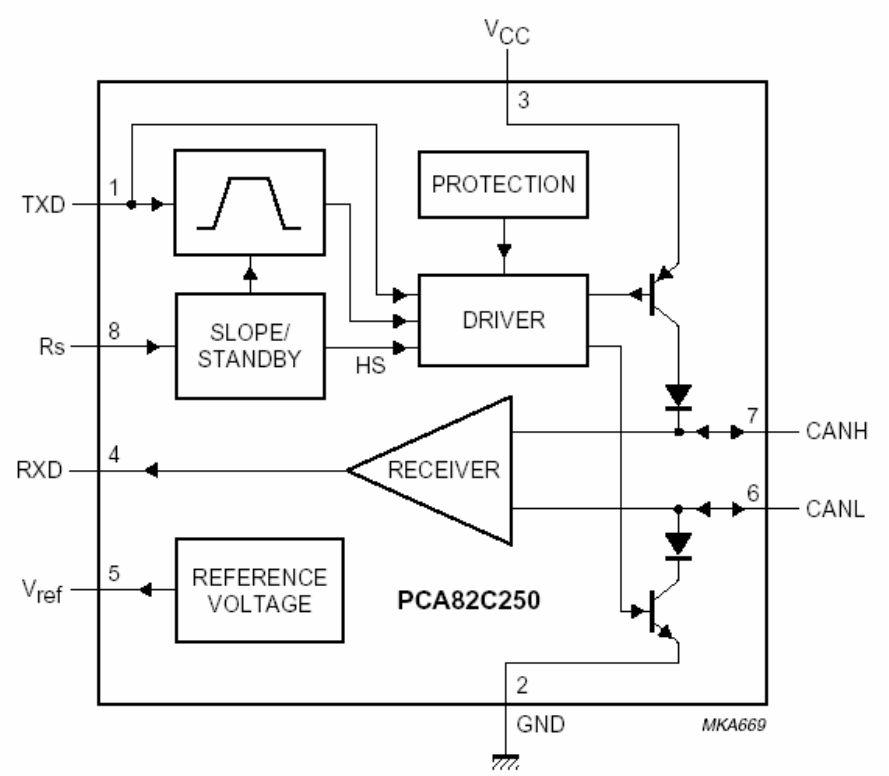
Pin Name	Type	Description
P3.0:7	I/O	Port 3: Is an 8-bit bi-directional I/O port with internal pull-ups. Port 3 pins that have 1's written to them are pulled high by the internal pull-up transistors and can be used as inputs in this state. As inputs, Port 3 pins that are being pulled low externally will be a source of current (I_{IL}, See section 'Electrical Characteristic') because of the internal pull-ups. The output latch corresponding to a secondary function must be programmed to one for that function to operate (except for TxD and $\overline{WR}$). The secondary functions are assigned to the pins of port 3 as follows: P3.0/RxD: Receiver data input (asynchronous) or data input/output (synchronous) of the serial interface P3.1/TxD: Transmitter data output (asynchronous) or clock output (synchronous) of the serial interface P3.2/$\overline{INT0}$: External interrupt 0 input/timer 0 gate control input P3.3/$\overline{INT1}$: External interrupt 1 input/timer 1 gate control input P3.4/T0: Timer 0 counter input P3.5/T1: Timer 1 counter input P3.6: Regular I/O port pin P3.7: Regular I/O port pin
P4.0:1	I/O	Port 4: Is an 2-bit bi-directional I/O port with internal pull-ups. Port 4 pins that have 1's written to them are pulled high by the internal pull-ups and can be used as inputs in this state. As inputs, Port 4 pins that are being pulled low externally will be a source of current (IIL, on the datasheet) because of the internal pull-up transistor. The output latch corresponding to a secondary function RxDC must be programmed to one for that function to operate. The secondary functions are assigned to the two pins of port 4 as follows: P4.0/TxDC: Transmitter output of CAN controller P4.1/RxDC: Receiver input of CAN controller. It can drive CMOS inputs without external pull-ups.
RESET	I/O	Reset: A high level on this pin during two machine cycles while the oscillator is running resets the device. An internal pull-down resistor to VSS permits power-on reset using only an external capacitor to VCC.
XTAL1	I	XTAL1: Input of the inverting oscillator amplifier and input of the internal clock generator circuits. To drive the device from an external clock source, XTAL1 should be driven, while XTAL2 is left unconnected. To operate above a frequency of 16 MHz, a duty cycle of 50% should be maintained.
XTAL2	O	XTAL2: Output from the inverting oscillator amplifier.

ANEXO 3

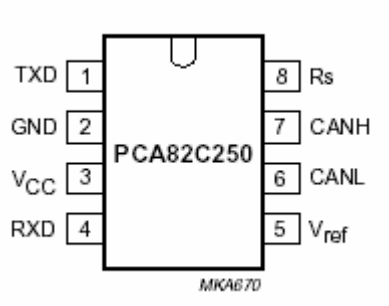
DESCRIPCIÓN DE PINES Y CARACTERÍSTICAS PRINCIPALES DE LOS TRANSCEIVERS PCA82C250 Y MCP2551

TRANSCEIVER PHILIPS PCA82C250

Diagrama de Bloques:



Configuración de pines:

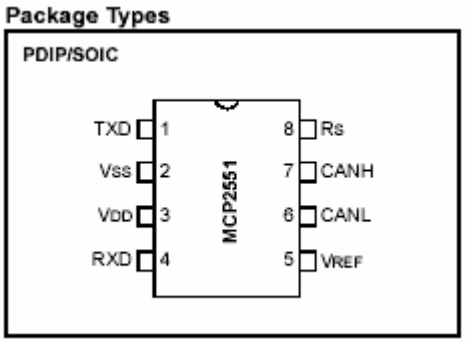


Descripción de pines:

SYMBOL	PIN	DESCRIPTION
TXD	1	transmit data input
GND	2	ground
V_{CC}	3	supply voltage
RXD	4	receive data output
V_{ref}	5	reference voltage output
CANL	6	LOW-level CAN voltage input/output
CANH	7	HIGH-level CAN voltage input/output
Rs	8	slope resistor input

TRANSCEIVER MICROCHIP MCP2551

Configuración de pines:



Descripción de pines:

Numero de Pin	Nombre del Pin	Función
1	TxD	Transmisión de datos (IN)
2	Vss	Tierra
3	Vdd	Voltaje de alimentación
4	RxD	Recepción de datos (OUT)
5	VREF	Voltaje de referencia
6	CAN_L	Voltaje de nivel bajo CAN (I/O)
7	CAN_H	Voltaje de nivel alto CAN (I/O)
8	RS	Slope Control

INDICE DE FIGURAS

INDICE DE TABLAS

GLOSARIO

Bit Unidad más pequeña de información, representa la unidad base en la comunicación digital mediante dos estados “0” o “1”.

Broadcast Mensaje destinado a todos los dispositivos pertenecientes a una red.

Bus Representa una topología de red en la cual existe una vía o canal de transmisión compartido por todos los dispositivos conectados a la red.

CAN Controller Area Network

Controlador Dispositivo encargado de gestionar la información y controlar periféricos.

Dominante En el Protocolo CAN se llama dominante al bit cuyo estado lógico es “0”. El mensaje cuyo identificador contenga en sus posiciones más significativas bits dominantes gana una fase de arbitraje y acceso al bus.

Evento Representa un cambio de estado en una entrada/salida o un determinado sensor u actuador

Hardware Comprende toda la parte física de un sistema.

Host Dispositivo encargado del procesamiento de datos dentro de la aplicación

Identificador Conjunto de bits utilizado para representar un determinado mensaje dentro de la red, es único para cada mensaje y determina su prioridad.

Interfaz Conexión que permite la correcta comunicación entre dos o más dispositivos.

Latencia: Tiempo de retardo que puede tener un mensaje sin acceder inmediatamente al medio de red.

Mensaje Conjunto de datos que transmite un medio de comunicación.

Multicast Mensaje destinado a varios dispositivos que pertenecen a una red.

Periférico Dispositivo externo que permite realizar funciones adicionales a un dispositivo.

Polling Interrogación que hace un sistema maestro de forma regular a cada uno de los dispositivos esclavos que recomunican con él.

Protocolo Conjunto de reglas estandarizadas que controlan el funcionamiento y transmisión de información en un tipo de comunicación.

Recesivo En el Protocolo CAN se llama recesivo al bit cuyo estado lógico es “1”. El mensaje cuyo identificador contenga en sus posiciones más significativas bits recesivo pierde una fase de arbitraje y acceso al bus.

Red Grupo de dispositivos interconectados que pueden transferir información entre ellos.

Software Comprende toda la parte lógica de un sistema relacionada con programas o conjunto de instrucciones grabadas en una memoria.

Transceiver Dispositivos que permiten acoplar dos tipos de señales o las cambian de magnitud.

Unicast Mensaje destinado a un solo dispositivo de la red.

SANGOLQUÍ, 9 de Agosto de 2005

ELABORADO POR:

_____

Iván Alejandro Vergara Vargas

El Decano

El Secretario Académico

_____

Tnt. Crnl. Marcelo Gomez

_____

Dr. Jorge Carvajal