Polibits

ISSN: 1870-9044

polibits@gelbukh.com

Instituto Politécnico Nacional

México

Pimentel Cruz, Jesús; Pérez Romero, Patricia

Introducción a las Redes CAN 1a. Parte

Polibits, núm. 35, 2007, pp. 18-21

Instituto Politécnico Nacional

Distrito Federal, México

Introducción a las Redes CAN 1a. Parte

Ing. Jesús Pimentel Cruz
Ing. Patricia Pérez Romero
Profesores del CIDETEC-IPN

Este artículo se presenta como el primero de una serie de trabajos sobre el protocolo ***CAN*** (***Controller Area Network***), los cuales abarcaran desde los aspectos básicos en dicho protocolo hasta su implementación y comunicación en dos tarjetas de hardware, las cuales están diseñadas con microcontroladores *PIC* y *AVR*.

Longitud del Bus	Velocidad en Bits/seg	Tiempo Máximo de Transmisión
Hasta 25 mts.	1Mbit/s	129 μs
Hasta 100 mts.	500 Kbit/s	258 μs
Hasta 500 mts.	125 Kbit/s	1032 μs
Hasta 1000 mts.	50 Kbit/s	2580 μs
Mensaje para CAN2 de 129 bits de longitud		

Tabla 1

Introducción

El sistema CAN consta de un mecanismo de comunicación serial multiplexado, en el cual la entrega a tiempo de un mensaje es mucho más importante que el ancho de banda; las redes que trabajan con CAN lo hacen a diferentes velocidades, considerando una serie de recomendaciones practicas, tal comose indica en la **Tabla 1**. Este tipo de redes se utilizan para trabajo en ambientes muy hostiles, destacándose por ser muy robustas y seguras; originalmente fueron desarrolladas por el consorcio alemán BOSH GmbH para el intercambio de información en tiempo real entre piezas automotrices, con el objetivo de conseguir automóviles más seguros y confiables, al mismo tiempo que tuvieran una eficiencia mayor en el consumo de combustible y un menor peso. Actualmente, esta tecnología se utiliza ampliamente en la industria automovilística, pero sus aplicaciones se están extendiendo rápidamente a otros mercados, como son la industria eléctrica ó aquellos procesos de transformación con alta posibilidad de fallas críticas.La existencia de un estándar del *bus* CAN a nivel mundial posibilita que unidades de control de diferentes fabricantes se puedan comunicar entre si.

Generalidades del Bus CAN

La funcionalidad específica del Bus CAN se puede interpretar, en términos del modelo OSI, como un protocolo de dos capas, mostrado en la **Figura 1.**

• Capa Física: se usa generalmente para implementar redes CAN en una línea de bus con dos cables de conducción diferenciados.

• Control de Acceso al Medio: se contemplan tramas y arbitrajes, corrección de errores, detección y bandera de error.

• Control de Enlace Lógico: incluye aspectos de transferencia de datos, petición remota, filtrado de mensajes, notificación de sobrecarga y recuperación.

Se tiene acceso al bus serial multimaestro con un esquema CSMA/CD, (*Carrier Sense Multiple Access / Collision Detection*, Acceso Múltiple con Detección de Portadora/Detección de Colisiones) por lo que es posible que varios nodos intenten transmitir al mismo tiempo en cualquier momento, y por lo mismo se pueden dar colisiones entre mensajes simultáneos; sin embargo, cada mensaje lleva una identificación que establece su prioridad, obteniendo el acceso al bus el de mayor jerarquía, mientras que los de menor nivel esperan a que el bus se encuentre libre, mediante un arbitraje de Bus (*Bus Arbitration*).

El Bus CAN es un protocolo con prioridad al mensaje y no a la dirección; esto significa que los participantes no tienen una dirección y el mensaje se envía a todos los nodos de la red CAN. Son estos nodos los que deciden, con base en una identi-

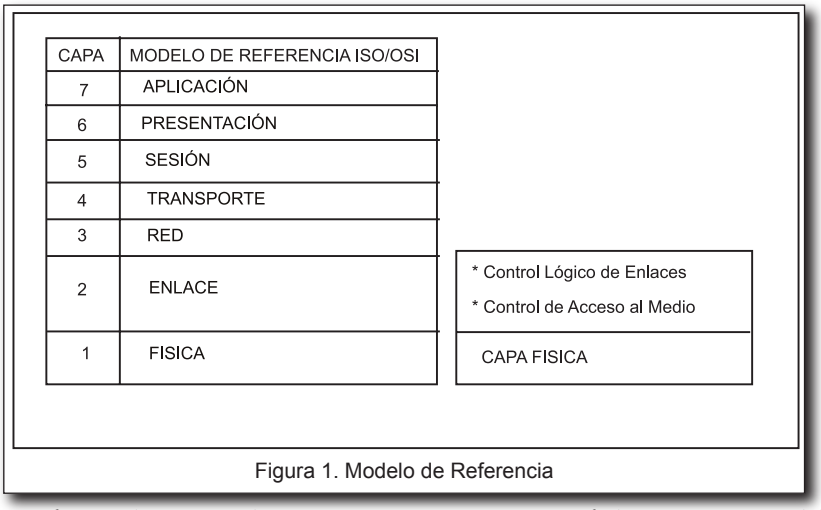

Figura 1. Modelo de Referencia

ficación, si el mensaje les sirve para procesarlo o lo desechan; así, si un nodo envía un mensaje, dicho mensaje no incluye un campo de nodo destino, dado que todos los nodos escuchan el canal, y en un momento dado varios de ellos pueden aceptar la transmisión. Las identificaciones las asigna el usuario CAN según lo crea conveniente, teniendo que cumplir algunas reglas, como la de que dos nodos no pueden transmitir con la misma identificación.

Trama de Datos

Los mensajes CAN constan de celdas donde se envían datos e información de acuerdo a las especificaciones que marca el protocolo; ver **Figura 2**, el formato del paquete incluye los campos siguientes:

• SOF (*Start of Frame*, Inicio de Trama), inicio de trama, consta de un bit siempre dominante, que indica el inicio del mensaje.

• Arbitraje, este campo es el que concede prioridad a un mensaje o a otro (dominante o recesivo); en el formato estándar tiene once bits de identificación, seguidos del bit RTR (*Remote Transmisión Request*, Petición de Transmisión Remota), que en este caso será dominante. En el formato extendido son los once bits de identificación base y dieciocho bits adicionales, y el bit SRR (*Subsitute Remote Request*, Petición de Sustituto Remoto) sustituye al RTR, siendo *recesivo.* Ver **Figura 3**.

• Campo de Control.- Este grupo está formado por dos bits reservados y cuatro bits adicionales que indican el numero de bytes de datos; el primero de estos bits, r1 (IDE, *Identifier Extension*, Extensión de Identificación) se utiliza para indicar si la trama es estándar (IDE dominante) o extendida (IDE recesivo), el segundo de estos, r0, es siempre recesivo. Los cuatro bit del DLC (*Data Length Code*, Código de Longitud de Datos), indican, en binario, el número de bytes de datos en el mensaje.

• CRC (*Cyclic Redundance Code*, Código de Redundancia Cíclica), el código de esta celda, se utiliza para verificar si se han producido errores.

• ACK (*Acknowledge,* Reconocimento) este campo de dos bits (*SLOT,* Ranura *y Delimiter,* Delimitador*)* indica si el mensaje ha sido recibido correctamente. El nodo transmisor envía los dos bits como recesivos, mientras que el nodo que lo reciba contestará enviando un bit dominante en el campo SLOT si ha recibido de manera correcta un mensaje válido.

Figura 2 Trama de Datos

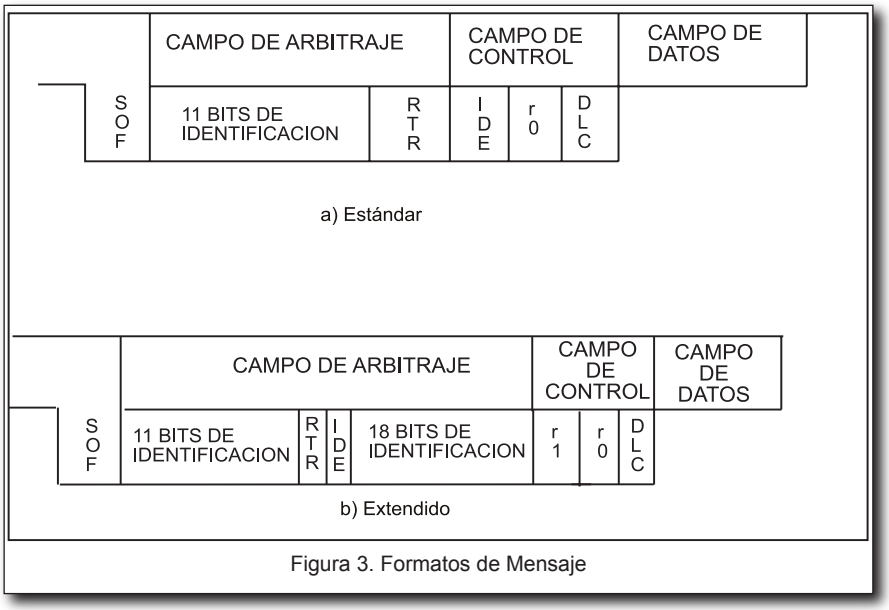

Figura 3. Formatos de Mensaje

• EOF (*End of Frame*, Fin de trama), consiste en una bandera de siete bits recesivos, en forma sucesiva, que indican el final de la trama.

• IFS (*InterFrame Space*, Espacio entre Tramas), esta consta de al menos tres bits recesivos, y durante su vigencia ningún nodo tendrá permiso para iniciar transmisiones.

Trama Remota

Los nodos pueden requerir información de otros nodos, por ello, si alguno realiza una petición en este sentido, el nodo que tiene la información envía un mensaje con la repuesta, misma que puede ser recibida por quien la solicitó y por cualquier otro nodo interesado.

El mensaje de petición remota tiene el formato que se muestra en la **Figura 4**. Este tipo de mensajes se envían con una trama que incluye la identificación del nodo requerido, a diferencia con los mensajes de datos, el bit RTR toma valor recesivo y no hay campo de datos. En el caso de que se envíe un mensaje de datos y de petición remota con la misma identificación, la trama de datos ganará el acceso al bus, ya que su RTR tiene valor dominante.

Trama de Error

Esta trama, mostrada en la **Figura 5**, es generada por cualquier nodo que detecte un error, y consiste de dos campos: indicación de de error (*error flag*) y delimitador de error (*error delimiter*), el cual consta de ocho bits recesivos continuos y permite a los nodos reiniciar la comunicación tras el error.; la indicación va a depender del estado de error del nodo que detecte dicha condición.

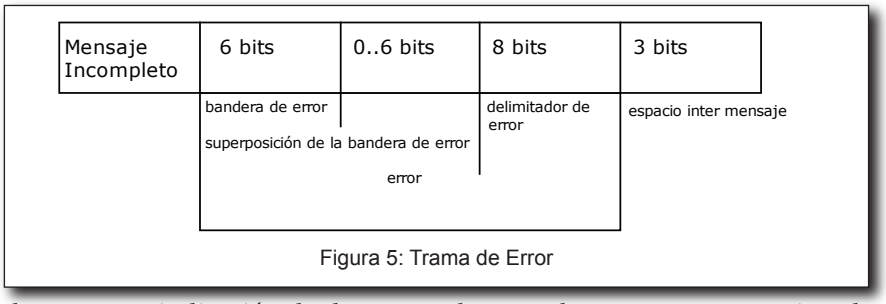

Figura 5: Trama de Error

Si un nodo en estado de error *Activo* detecta otro error en el bus, interrumpe la comunicación del mensaje en proceso, generando un nuevo mensaje de error activo, que consiste en una secuencia de seis bits dominantes sucesivos. Esta secuencia no cumple con la regla de relleno de bits y genera la trama de error en otros nodos, por lo tanto el error puede extenderse de seis a doce bits dominantes sucesivos. Se espera la llegada de un campo de limitación de error formado por los ocho bits recesivos, entonces la comunicación se reinicia y el nodo que había sido interrumpido comienza nuevamente la transmisión del mensaje.

Si un nodo en estado *pasivo* detecta un error, el nodo transmite un indicador de error pasivo, seguido nuevamente por el campo delimitador de errores. El indicador de error de tipo pasivo consiste de seis bits recesivos seguidos y por lo tanto la trama para este tipo de nodos es de catorce bits recesivos; por ello, la trama de error de tipo pasivo no afecta a nodo alguno en la red, excepto cuando el error es detectado por el mismo nodo que ésta transmitiendo. En este caso, los demás nodos detectan una violación a las reglas de relleno y transmitirán a la vez una trama de error.

El relleno de bits consiste en que cada cinco bits de igual valor se introduce uno de valor inverso. Al ser señalada una trama de error de la forma apropiada, cada nodo transmite bits recesivos

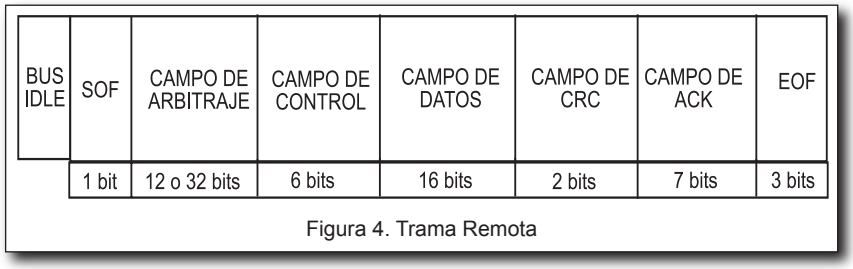

Figura 4. Trama Remota

Implementación de controladores CAN

Existen tres tipos de arquitecturas en microcontroladores: Controlador CAN Autónomo (*Stand-Alone CAN Controller)*, Controlador CAN Integrado (*Integrated CAN Controller)* y Nodo CAN de Circuito Único (*Single-Chip CAN Node)*.

Controlador CAN Autónomo

Es la arquitectura más simple, para llevar a cabo una comunicación en una red CAN se requiere:

1. Un microcontrolador.

2. Un controlador CAN para introducir el protocolo, el filtro de mensajes y todo lo necesario para las comunicaciones. Un ejemplo es el dispositivo MCP2510.

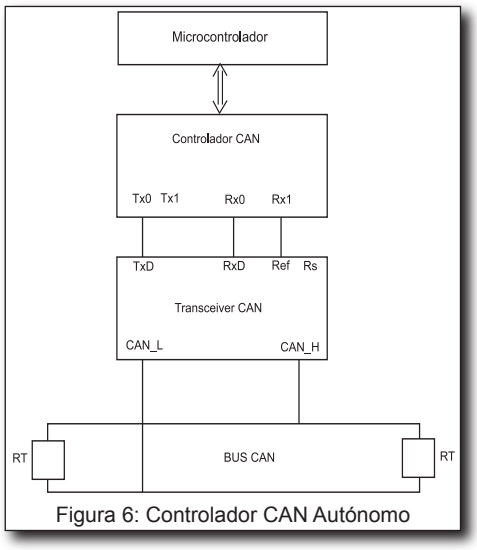

Figura 6: Controlador CAN Autónomo

3. Un transceiver CAN, esto es, un transmisor/receptor; por ejemplo, el MCP2551 desarrollado por Microchip. **Figura 6**.

Controlador CAN Integrado

Este tipo de arquitectura consiste en un microcontrolador que incluya, no sólo sus características propias sino además un módulo con la funcionalidad de un microcontrolador CAN. El transceiver se sitúa de manera separada como se puede observar en **Figura 7**.

Nodo CAN en un solo Circuito

Se trata de un chip que incluye en su interior los tres elementos necesarios para llevar a cabo las comunicaciones en un entorno CAN, ver Figura 8.

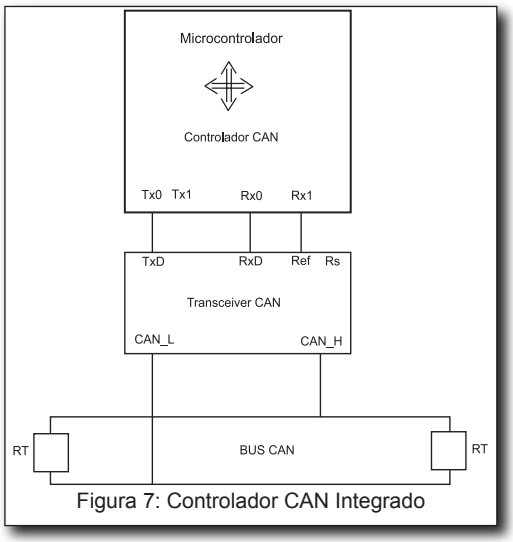

Figura 7: Controlador CAN Integrado

Bibliografía:

[1] Pfeiffer Ayre, Olaf; Keydel, Christian. *Embedded Networking*.RTC Books.

[2] Bosch GmbH, Robert. *CAN Specification version 2.0*

[3] Chamú Morales, Carlos Antonio. *Desarrollo de un sistema educativo para la enseñanza del protocolo de comunicaciones CAN.* Universidad Tecnológica de la Mixteca.

[4] Etschberger, Konrad. Controller Area Network: *Basics Protocols, Chips and Applications*; IXXAT Automotion GmbH.

[5] Lawrenz, Wolfhard. *CAN System Engineerin: From Theory to Practical Applications*. Springer

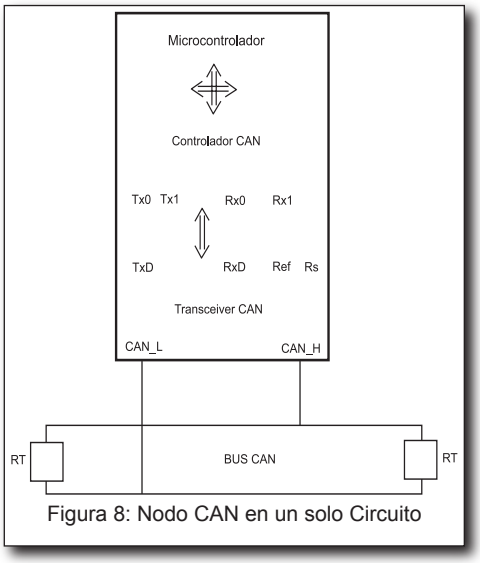

Figura 8: Nodo CAN en un solo Circuito