Sincronización en Microcontroladores en Red: Implementación y Evaluación

Fernando G.Tinetti[1], Ricardo A. López, Marcelo E. Gómez, Sebastián P. Wahler

Departamento de Informática Sede Trelew, Facultad de Ingeniería - UNPSJB
III-LIDI, Facultad de Informática - UNLP
[1]Investigador Asistente Comisión de Investigaciones Científicas de la Prov. de Bs. As.
fernando@info.unlp.edu.ar, lopez.ricardoa@gmail.com

Resumen. Este artículo se enfoca en el problema de sincronización de relojes de microcontroladores en red, teniendo en cuenta la utilización de la red de microcontroladores para adquisición de datos y control. Es importante remarcar que el problema de sincronización de relojes sigue siendo un área activa de investigación en el contexto de los sistemas distribuidos. Esta temática, en el contexto de microcontroladores interconectados, puede aprovechar lo ya estudiado pero también considerando que existen restricciones y aplicaciones específicas, características. Parte de los temas de investigación pueden ser utilizados casi directamente para docencia y los temas más avanzados son el objeto de estudio específico para investigación. Los problemas como la evaluación de la sincronización, tanto analítica como experimental, son de vital importancia en lo relacionado con los sistemas SCADA (*Supervisory Control And Data Acquisition*).

Palabras Clave: Sincronización de Relojes, Redes de Microcontroladores, Sistemas de Tiempo Real, Sistemas SCADA.

1 Introducción

La problemática de sincronización tiene como escenario una red de bajo costo implementada con Microcontroladores bajo norma RS485 (fig. 1). Los dispositivos Microcontroladores consignado en la figura, tienen la misión de interactuar con el campo. La cantidad de nodos responde a variaciones y configuración particular de cada uno y la complejidad del sistema a tratar. Todos estos dispositivos se encuentran interconectados mediante una red multipunto en velocidades relativamente bajas: 9600, 28800, 57600 bps. Existe un dispositivo destacado, denominado Concentrador cuya misión primordial es la de ser puente entre la red multipunto de concentradores, y otra red que sería de mayor velocidad y normalizada, a efectos de multiplicar la cantidad de dispositivos a interconectar [8].

La adquisición de datos, es una de las tareas definidas para el sistema. Por ello, cierto tipo de datos (analógicos o digitales) que sean definidos como eventos deben tener en su registro, una marca de tiempo (*timestamp*) asociada, para que pueda ser posible crear una secuencia de ellos y establecer una relación causa y efecto. Un

evento se define como un cambio de estado de su variable asociada. En particular, un evento digital sería el pasaje de 0 a 1 ó de 1 a 0. Para posibilitar su registro, debe estar almacenado su estado anterior dado que los microcontroladores provocan una interrupción ante el cambio de estado de un puerto. La interrupción ejecuta la rutina de servicio asociada en unos pocos microsegundos y en ese tiempo, se lee el valor del reloj de tiempo real (RTC: *Real Time Clock*) contenido en la unidad, se determina nuevo estado de la variable y todo este conjunto se almacena en una pila secuencial de eventos destinada a tal fin.

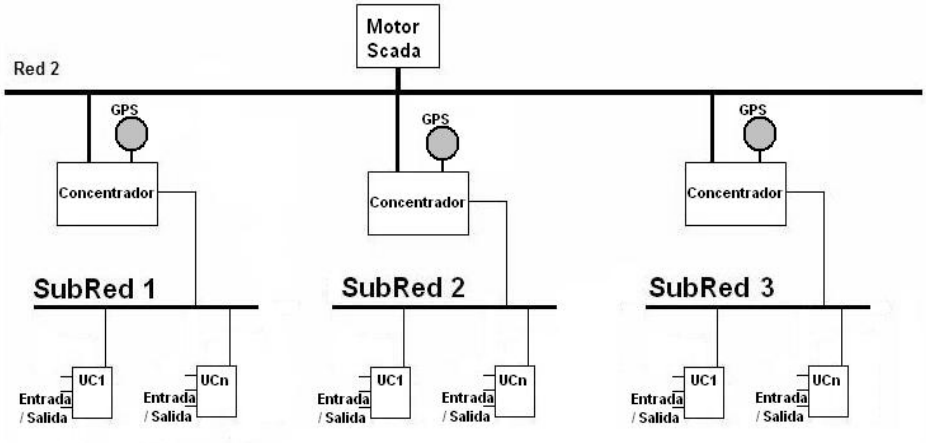

Fig. 1. Contexto de Sincronización de Microcontroladores Interconectados.

En la sincronización de sistemas de una única CPU (o microcontrolador) no se requiere ninguna consideración especial en el diseño del software, ya que existe un reloj único que proporciona en forma regular el tiempo en cada momento. Los sistemas distribuidos en cambio, tienen un reloj por cada CPU del sistema, por lo que es fundamental una coordinación entre todos los relojes para tener una hora única. Por esto, se debe definir alguna representación de tiempo y además los dispositivos UC1, …, UCn de la figura deben aceptar sincronización horaria por un dispositivo externo. Un factor de mérito deseable entonces, es la dispersión de hora que presentan cada una de las unidades respecto a un reloj patrón. Este factor se denomina exactitud, debe ser acotado y es normal obtener un valor de este que esté por debajo de 1 milisegundo (1ms). La sincronización así definida haría posible presentar una secuencia de eventos ocurridos en un ambiente (industrial u otro similar), vinculado a microcontroladores del SCADA, con consistencia independientemente de la unidad que hubiere registrado cada evento. La exactitud de esta secuencia, será determinada por la convergencia que guardaren los RTC de cada una de las unidades a través del tiempo. Es por ello que la sincronización es elemento vital en la presentación de los resultados o en la evaluación del estado del sistema que es presentado al usuario.

Los osciladores de cada microcontrolador son ligeramente diferentes, típicamente alrededor de 20 partes por millón (ppm) y como consecuencia todos los relojes sufren un desfase y deben ser sincronizados periódicamente. La sincronización no es trivial, porque se realiza a través de mensajes por la red, cuyo tiempo de transmisión puede

ser variable debido principalmente, a variabilidad en el dispositivo que sincroniza por un lado y las propias derivas de cristales de cada unidad por el otro, los cuales conjugadamente llevan a sesgos apreciables en un momento dado. Dado que todos los dispositivos de la red tienen la deriva de hora aludida, debe elegirse uno de ellos como "patrón" de hora y el resto de los individuos de la red debe ser sincronizado mediante algún procedimiento con la hora que indica ese patrón. De esta forma, se llega a lo que se da en llamar *Sincronización Interna* [5] [6].

En base al alcance de la distribución física, similar tipo de tecnología y necesidad de contar con una cantidad acotada (alrededor de 30) de dispositivos en una zona o área de trabajo, se definió que el candidato ideal para la tarea de patrón o master de la sincronización sea el Concentrador. Ya se observó oportunamente que este dispositivo ocupa un lugar preponderante en la red debido a su actuación como puente o *gateway* entre dos redes de distinta velocidad y alcance. Por ello y como puede apreciarse en la fig. 1, en esta condición se podrán tener "islas" o subredes, sincronizadas por un Concentrador, con alguna exactitud a determinar. Dado que cada concentrador puede tener asociado un dispositivo que otorgue el UTC (Tiempo Universal Coordinado), como ser un GPS, podría lograrse la llamada Sincronización Externa de todo el sistema. No obstante, éste no es todavía el objetivo de este trabajo, aunque sería uno de los objetivos futuros.

2 Desarrollo

Sin lugar a dudas, una de las primeras líneas de estudio se enfocó al análisis caso por caso de las estrategias de sincronización existentes en el contexto de sistemas distribuidos [6]. Dicha estrategia es esencial para resolver problemas tales como ordenamiento de eventos (envío y recepción de mensajes, lanzamiento de eventos, etc). Era esperable llegar a una propuesta de sincronización que, por un lado tenga en cuenta las limitaciones de capacidad de los microcontroladores, pero no obstante por el otro que aproveche las ventajas de un sistema que podría considerarse de tiempo real estricto, con cotas de tiempo de respuesta a los eventos conocidas de antemano. Esta característica puede considerarse propia de los sistemas basados en microcontroladores [1] [2], donde en muchos casos la cantidad de eventos por unidad de tiempo es acotada y el software que se ejecuta es sencillo de analizar y también estimar en cuanto a tiempo de ejecución. Una vez decidida la estrategia de sincronización, es importante el análisis exhaustivo y lo más formal posible desde dos puntos de vista:

- Requerimientos impuestos a los microcontroladores y a la red de interconexión de los mismos. Esto debe tener en cuenta que los microcontroladores no se dedican exclusivamente a la sincronización.
- Cotas de error de sincronización, ya que toda sincronización tiene en sí misma una definición y especificación de error.

A efectos de obtener factores de mérito de la solución, ha sido importante la implementación y experimentación de un entorno real. Como mínimo, es necesario contar con resultados experimentales para comparar y analizar, en función de los cuales avanzar hacia una implementación probada. A continuación se hace referencia

a aspectos generales de la problemática de sincronización, combinados con aspectos de la implementación particular, efectuada con microcontroladores de la línea de un fabricante del mercado.

2.1 Concentrador

El concentrador es un elemento que fue definido inicialmente para operar entre dos redes y además tiene capacidad de comunicarse mediante otro canal serie bajo norma RS232. El Concentrador gestiona las comunicaciones entre un centro de control (donde reside el Motor SCADA de la Fig. 1) y las unidades de control. Para la implementación de referencia que se describe en esta sección, este dispositivo está compuesto por un hardware y un software específicos. El hardware está basado en el microcontrolador PIC 18F97J60 de Microchip, unidad con la ventaja de poseer embebido un módulo Ethernet. El software del Concentrador fue concebido sobre la base de la técnica de programación denominada *multitarea colaborativa*. Por lo tanto, la estructura del Concentrador es la de un gran programa con un único hilo de ejecución, que invoca secuencialmente a diferentes tareas (colaborativas) de manera iterativa. Las tareas colaboran con el fin común, resolviendo su trabajo en un tiempo apropiado o resolviendo parte de su trabajo y entregando el control. De esta forma, se asegura que todas las tareas tendrán oportunidad de realizar su trabajo.

El reloj de tiempo real que tiene el Concentrador para mantener su propia hora es obtenido mediante un temporizador (*timer*) hardware interno alimentado directamente por un cristal de cuarzo de 32768 Hz. Este reloj es tomado como "patrón" de sincronización interna para la subred subyacente y dada su exactitud inherente (20 ppm), se logran mediciones para sucesos dentro de la misma CPU con exactitud por debajo de décimas de milisegundos.

2.2 Estrategia de Sincronización Interna

Dado que los microcontroladores a sincronizar son de mediana o baja capacidad, se decidió una característica importante de la red de interconexión, casi directamente orientada hacia la posibilidad de sincronización: posibilidad de broadcast físico de datos. Por un lado, la gran mayoría de las redes físicas más populares cuentan con esta posibilidad, por ejemplo EIA 485 (RS485) [7] y Ethernet [4]. Por otro lado, esta permite que haya un controlador que establezca la hora en toda la red con una única transferencia de datos por la red. De manera prácticamente simultánea, todos los microcontroladores reciben la información necesaria para la sincronización. Se decide implementar una estrategia de sincronización en el cual el dispositivo que arbitra el flujo en cada subred multipunto (cada Concentrador), sea el encargado de enviar en forma periódica y entrelazada con los mensajes de datos, un mensaje de sincronismo en broadcast para todas las unidades de la subred. Con esta metodología, cada unidad de la subred que recibe el mensaje de sincronismo efectúa el ajuste de su reloj interno.

En una primera aproximación, el envío de hora a la red se realizó dentro de la rutina principal del concentrador, enviando cada uno de los bytes como una tarea más dentro del ciclo de tareas colaborativas, es decir por encuesta (*polling*) de la UART.

Para ello, se construye un paquete de Puesta en Hora (PeH), que contiene desde el año actual hasta una fracción de segundo de cuatro cifras significativas (décimas de milisegundo), el que se comienza a enviar desde su primer byte a la red RS485, inmediatamente después de obtenida la hora local. Enviado el primer byte, se van enviando los sucesivos (hasta un total de 15 bytes integrantes del paquete) a medida que la UART está disponible (con encuesta). Enviado el último byte, se inicia un temporizador de algunos segundos, terminado este se envía otra nueva PeH y así sucesivamente. Este intervalo del temporizador es regulable entre 1 y 255 segundos y en el experimento inicial se fijó en 5 segundos. Con parámetros del mensaje 9600, N, 8, 1 (velocidad en bps, paridad, largo de palabra, bits de stop) se cumple:

$$\Delta t = N \times 10 \times 1/Vt \quad (1)$$

$$h_{pdo} = h_o + \Delta t \quad (2)$$

donde:

Δt = tiempo de transmisión del mensaje en segundos.
N = Cantidad de bytes del paquete de PeH.
Vt = Velocidad de transmisión en bps, del canal RS485.
h_o = Hora inicial (hora enviada en la PeH).
h_{pdo} = Hora PeH a imponer al destino.

Mientras que la ecuación (1) otorga el tiempo que el mensaje tarda en llegar al destino, la ecuación (2) otorga la hora corregida a partir de la hora patrón h_o extraída desde el concentrador.

En la implementación se pudo observar que con esta técnica de inclusión de la sincronización en entre las tareas estándares del Concentrador, se obtiene un ajuste de hora con un delta de tiempo que resultó con fluctuaciones por encima del valor teórico que se obtiene de la ecuación (1). Aunque esa variación es pequeña, impacta linealmente sobre la hora con que se corrige la hora local en cada UC como se aprecia por la ecuación (2). Las mediciones efectuadas con esta técnica arrojaron una fluctuación que oscilaba en +/- 1 ms, teniendo en cuenta que el valor teórico que resulta de la ecuación (1) para los parámetros del mensaje consignados es de 15,625 ms. Independientemente del tiempo que tarde el mensaje, la variación en el instante de tiempo que resulta de la modalidad de envío del mismo impacta directamente sobre lo que se obtiene en cada una de las UC sincronizadas. En efecto, entre cada uno de los bytes integrantes del paquete de PeH existen pequeños tiempos variables que se suman al tiempo total Δt de transmisión, con el que se ajusta finalmente a la llegada a la UC. Este efecto puede apreciarse en la fig.2a). Para resolver el problema de la falta de regularidad en el envío de los bytes de sincronización, fue necesario cambiar el esquema de tareas colaborativas permitiendo que dentro de las mismas el envío de puesta en hora se realice con una cadencia fija, y ese Δt pase a un valor cuasi-constante, con fluctuaciones acotadas e inferiores a las logradas por encuesta, tal como se muestra en la fig. 2 b). La implementación de esta solución consiste en cambiar la forma en que se envía el comando de puesta en hora, y resolverlo mediante interrupciones. Al llegar el instante correspondiente a la emisión por un comando de PeH por parte del concentrador, inmediatamente se ejecutará la rutina correspondiente a tomar la hora, armar el paquete correspondiente y listo éste activar el mecanismo de interrupciones de la UART a efectos que cada byte sea requerido por interrupciones [10].

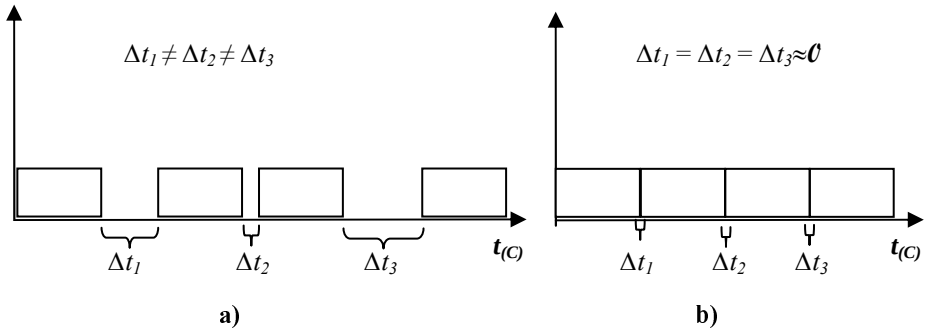

Fig. 2. Secuencia de Bytes de PeH: a) por Polling, b) por Interrupciones.

La interesante particularidad de este mecanismo hace que los tiempos existentes entre byte y byte del mensaje de PeH sean muy cercanos a cero y sistemáticamente iguales, convirtiendo el mensaje asincrónico por naturaleza en cuasi-sincrónico y con una duración mesurable y predecible tal cual se aprecia en la fig.2b. De esta forma es posible saber con mayor seguridad con qué frecuencia se enviará el comando PeH y una vez que llegue el paquete a las unidades de la red, éstas podrán sumar a sus relojes propios un Δt, que ahora pasa a ser una magnitud fija y conocida.

3 Escenario de Prueba y Resultados Obtenidos

La fig. 3 muestra el pseudocódigo del software que se ejecuta en cada UC de la red a sincronizar, que no deja de ser similar a la mayoría del software de control embebido: una iteración con un conjunto de evaluaciones para registrar cambios de estado y procesar comandos de control.

```
(1)  Configurar/Inicializar (timer/s, UART/s, etc.)
(2)  Loop forerever Do
(3)  If (hay un trama a procesar) Then
(4)     Procesar trama
(5)  End if
(6)  If (hay un comando recibido a procesar) Then
(7)     Procesar recepción de un comando
(8)     If (hay respuesta de comando a procesar) Then
(9)        Preparar la respuesta del comando
(10)       Iniciar la respuesta del comando
(11)    End If
(12) End If
(13) Registrar estado
(14) End Loop
```

Fig. 3. Pseudocódigo del programa de una UC.

Lo relacionado con las tramas (línea 3 de la fig. 3) corresponde al procesamiento de

las comunicaciones y lo relacionado con los comandos (línea 6 de la fig. 3), corresponde al sistema SCADA en general y con el mecanismo de sincronización en particular, dado que uno de los tipos de comandos a procesar es el de sincronización. Para este comando cada una de las UC debe asignar como nueva hora local la recibida en el comando (en este caso no habrá mensaje de respuesta).

Con el fin de demostrar las mejoras que suponen teóricamente los cambios propuestos en la descripción anterior se realizaron una serie de pruebas en un entorno de trabajo que respeta la filosofía esbozada en la fig.1, con tres unidades de control y un concentrador. Además, una Terminal de diagnóstico oficia de Centro de Control y emite mensajes mediante un protocolo desarrollado para tal fin. Dicho entorno se presenta en la fig.4. Esta placa prototipo contiene tres microcontroladores de la familia Microchip (PIC 16F877 y 16F876 [11]): UC1, UC2 y UC3 que son las *unidades principales*. Contienen CPU, capacidad de memoria y dispositivos de E/S suficientes como para las pruebas de software del sistema. En la fig. 4 se destaca como ejemplo para la UC1, un detalle de la periferia que poseen las unidades principales: a) Entradas analógicas b) Entradas digitales y c) Leds como salidas digitales.

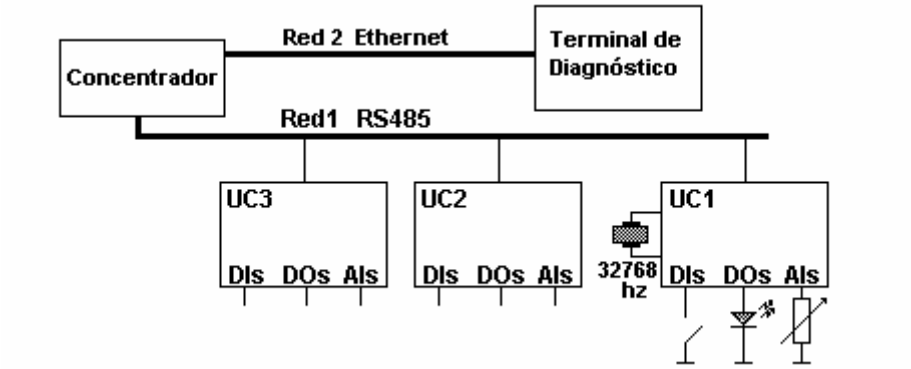

Fig. 4. Esquema del Ambiente de Evaluación para Sincronización.

Por otro lado, se ha tomado una placa ya construida sobre la que se ha desarrollado el software del Concentrador sobre una arquitectura basada en un microcontrolador de la línea microchip 18F97J60 con conexión Ethernet embebida. El concentrador está vinculado a la Terminal de diagnóstico montado sobre una PC mediante una conexión Ethernet. El concentrador emite cada 5 segundos un comando de PeH cuya estructura resumida es la siguiente:

Denominación	**Preámbulo de comunicaciones**	**Año, día, Hora, Segundo**	**Fracción de segundo**	**BCC**
Cant. de bytes	**5**	**4**	**1**	**2**

Donde:

- El preámbulo de comunicaciones incluye la cantidad de bytes del mensaje (15) y las direcciones de origen y destino de UCs (en este caso FFh: Broadcast).

- En un bloque de 4 bytes se codifica el año, el día dentro del mismo, la hora y el segundo dentro de la misma.
- En un bloque de dos bytes se codifica la fracción de segundo.
- El bloque final del mensaje de 2 bytes es el Check-Sum de todo el paquete.

Todo el software involucrado se implementó de manera efectiva. Las pruebas más relevantes realizadas son:

- En la llegada a una UC, la misma calcula el sesgo existente entre su hora local y la que impone el mensaje de PeH (un valor positivo indica un adelanto de la UC). Este valor se calcula con la precisión apuntada de 4 cifras de fracción de segundo.
- El Terminal por otro lado y con intervalos de 500 ms recaba el estado de cada UC. Entre esos valores de estado de una UC se incluye el sesgo calculado en la PeH anterior y un número identificador de cada PeH a efectos de comparar sesgos correspondientes para cada UC. Estos valores son presentados en la Terminal y luego comparados en una planilla de cálculo.

En el experimento se tomaron 4896 muestras durante un intervalo de 40 minutos. Se compararon los sesgos correspondientes entre sí y el resultado de caso peor arrojó que, las tres UCs diferían entre sí a lo sumo en una unidad, como se presenta en las muestras 417 a 419 de la tabla 1 para las UCs de direcciones 02, 03 y 04.

Tabla 1. Muestras de Mensajes con Contenido de Tiempos para Evaluación.

Nro. Muestra	Trama del mensaje de respuesta de cada UC	Sesgo (Hex)	Sesgo (Dec)
417	**02**-00 09 02 **B3 FF FC** 03 0A 08 01 06 0C 00 2F 00 F0 01	FFFC	-4
418	**03**-00 09 00 **B3 FF FA** 03 0A 08 01 06 0C 00 27 00 E4 01	FFFA	-6
419	**04**-00 09 02 **B3 FF FB** 03 0A 08 01 06 0C 00 24 00 E0 01	FFFB	-5
3728	**03**-00 09 00 **33 FF F5** 03 0A 05 01 06 0C 00 07 00 E6 01	FFF5	-11
3729	**04**-00 09 02 **33 FF F7** 03 0A 05 01 06 0C 00 04 00 E4 01	FFF7	-9
3730	**02**-00 09 02 **33 FF F7** 03 0A 05 01 06 0C 00 2F 00 F4 01	FFF7	-9

El cálculo se condice con una fracción de 1/32768 de segundo hacia un lado u otro de la considerada como centro en este caso. Esto es un sesgo de +/- 0.03 ms entre las unidades. Si se toma el intervalo que media entre cada PeH de 5 s = 5000 ms se tiene un valor de exactitud entre muestras:

```
P = +/- 0.03ms / 5000ms = 6 x 10^-6 = +/- 6 ppm
```

Lo cual se condice con los valores esperados máximos de 20 ppm que asegura el fabricante. Sin embargo, se ha observado que si bien las mediciones de un conjunto correspondiente no tienen un divergencia entre si mayor a +/- 1 unidad, cuando se compara todo el conjunto de 4896 muestras entre sí, se observa -como caso peor-, un sesgo máximo de -11 (este caso se observó en la muestra 3728 de la tabla 1). No se observaron sesgos positivos de las UCs respecto al concentrador, esto es: en todas las medidas el concentrador resultó estar coincidente o atrasado hasta un valor negativo máximo de 11 unidades respecto a cualquier UC. Situándose entonces la media en un

sesgo de -5.5 unidades. Si se calcula la excursión máxima del sesgo, se obtiene un valor de exactitud global para el sistema de broadcast de:

```
P = +/- 0.03ms * (11 - 0)/2 = +/- 0.16ms
```

Esa media negativa de las medidas del sesgo determinadas experimentalmente, (cuando se esperaría que estuviera centrada en cero), se debe probablemente a un error sistemático en el concentrador, ocasionado por alguna de sus rutinas, que podría estar introduciendo un retraso de su reloj. Esto se supone así, debido a que el Concentrador cuenta con un cristal de 32768Hz, de similares características a los que poseen las UCs que tienen una exactitud garantizada como la apuntada y además demostrada experimentalmente mediante la alta convergencia de las mediciones. No obstante los importantes factores de mérito obtenidos a partir de la sincronización en broadcast, es un tema de investigación consecuente en las rutinas de los microcontroladores las causas del comportamiento apuntado.

4 Conclusiones y Trabajos Futuros

Se ha llegado a una definición e implementación bien definida de sincronización de microcontroladores de baja gama interconectados por una red de muy bajo costo. El valor de exactitud de Puesta en Hora mediante broadcast es altamente satisfactorio. El sesgo en cualquiera de las unidades respecto al dispositivo que oficia de patrón otorga valores comparables a la propia exactitud del patrón (20 ppm). Este resultado permite a las unidades de una subred de microcontroladores sincronizados mediante esta técnica, elaborar un registro de eventos consistente en relación causa–efecto, aún cuando estos eventos son registrados por unidades diferentes.

Como trabajo futuro resta efectuar la sincronización con un patrón externo de mayor exactitud que provea el UTC, logrando así la sincronización externa de una subred y, de hecho, todas las UCs del sistema distribuido completo. Siguiendo en esta línea, la sincronización lograda para las distintas subredes permitiría relacionar eventos ocurridos en cualquier punto del sistema global con una exactitud a determinar.

Referencias

1. S. F. Barrett, D. J. Pack, Microcontrollers Fundamentals for Engineers and Scientists, Morgan & Claypool Publishers, 2006, ISBN: 1598290584.
2. F. M. Cady, Microcontrollers and Microcomputers: Principles of Software and Hardware Engineering, Oxford University Press, 1997, ISBN: 0195110080.
3. Free Software Foundation, Inc., GNU General Public License, www.gnu.org/copyleft/gpl.html
4. Institute of Electrical and Electronics Engineers, Local Area Network - CSMA/CD Access Method and Physical Layer Specifications ANSI/IEEE 802.3 - IEEE Computer Society, 1985.
5. A. S. Tanenbaum, Computer Networks, 4th Edition, Prentice Hall Ptr, ISBN 0130661023, 2002.

6. A. S. Tanenbaum, M. van Steen, Distributed Systems: Principles and Paradigms, Prentice Hall, 2nd Edition, ISBN 0132392275, 2006.
7. Telecommunications Industry Association, “Application Guidelines for TIA/EIA-485-A”, TIA/EIA Telecommunications Systems Bulletin, 1998.
8. F. G. Tinetti, R. A. López , "Ambiente de Desarrollo y Puesta en Marcha de Sistemas Basados en Microcontroladores", IX Workshop de Investigadores en Ciencias de la Computación, Universidad Nacional de La Patagonia San Juan Bosco (UNPSJB), Trelew, Chubut, Argentina, Mayo 3-4 de 2007.
9. F.G. Tinetti, R.A. López, S.P. Wahler, “Sincronización Microcontroladores Interconectados: Evaluación de Factibilidad y Detalles de Implementación”, X Workshop de Investigadores en Ciencias de la Computación 2008, Fac. de Ingeniería, Universidad Nacional de La Pampa, Gral. Pico, La Pampa, Argentina, Mayo 5-6 de 2008, pp. 209 - 213.
10. Microhip Technology Inc., Tutorial USART - Using in Asynchronous Mode, 2001. Disponible en http://ww1.microchip.com/downloads/en/DeviceDoc/usart.pdf.
11. Microhip Technology Inc., PICmicro™ Mid-Range MCU Family Reference Manual, 1997. Disponible en http://ww1.microchip.com/downloads/en/DeviceDoc/33023a.pdf