Ingeniería Energética Vol.XXXII 2/ 2011 Abril – Julio p. 65 – 73 ISSN 1815 – 5901

Programación de un sistema de adquisición de datos utilizando el sistema embebido DNP/1110

Programming of a data acquisition system using the embedded system dil/netpc DNP/1110

Josnier Ramos - Guardarrama
Mercedes Canasi - Piñeiro
Raynel Díaz - Santos

Recibido: Octubre del 2010
Aprobado: Noviembre del 2010

Resumen/ Abstract

En el trabajo se presenta una alternativa económica de un sistema de adquisición de datos (SAD) basado en el kit de desarrollo DIL/NetPC DNP/1110, el cual esta formado por un microcontrolador de INTEL SA-1110 StrongARM que trabaja a 206 MHz y que tiene incorporado un controlador de Ethernet que permite desarrollar aplicaciones con el uso de la red. El sistema embebido dispone de un sistema operativo Linux, kernel versión 2.4.18. Como parte del diseño se incorporan de forma compacta los elementos necesarios para que el SAD sea capaz de muestrear ocho señales capturadas simultáneamente. De particular interés resulta la programación, presentándose dos variantes de solución: cuando se ejecuta la aplicación con un software que corre en el área de usuario y cuando el control se logra con un módulo del sistema operativo específico para el trabajo del SAD. El sistema está soportado por la integración de software libre y propietario.

Palabras clave: módulos de linux, sistema de adquisición de datos, sistema embebido.

This work focuses on an economic alternative for developing a data acquisition system (DAS) which is made up for The DIL/NetPC DNP/1110, which provides a very compact Intel 206 MHz SA-1110 StrongARM-based low power embedded controller with TCP/IP stack and web server for high-speed embedded networking applications. The embedded system has an operating system Linux, kernel version 2.4.18. The built data acquisition system has the necessary elements to take charge of governing the sampling process of eight signals. Of particular interest it is the programming, being presented two solution variants: when the application is executed with software that it runs in user's area and when the control is achieved with a specific build module of the operating system for the work of the DAS.. The system is supported by the integration of free software and property software.

Keywords: acquisition data system, embedded systems, linux module.

INTRODUCCIÓN

Los sistemas de control en tiempo real, que debido a sus amplias ventajas utilizan procesamiento digital, llevan implícito un sistema de adquisición de datos [1- 3]. En el mercado se ofrecen diversas tarjetas de este tipo, en su mayoría para ser trabajadas a través de una computadora personal.El desarrollo alcanzado por los sistemas empotrados es bien conocido, así como las ventajas que ofrecen en cuanto a recursos de hardware y herramientas de programación. Un sistema de este tipo con controlador de Ethernet incluido, brinda la posibilidad de ejecutar un

control a distancia, o lo que es lo mismo es posible disponer de un dispositivo que integra elementos que participan en la captación de las señales con aquellos que permiten transmitir a cualquier usuario conectado a una red de PC la información muestreada en tiempo real.

A continuación se detalla el SAD diseñado y construido con el empleo del kit de desarrollo DIL/NetPC DNP/1110 [4], el cual constituye un pequeño servidor con un sistema operativo Linux instalado, con el objetivo de utilizarlo en diferentes aplicaciones en la rama de la electroenergética. Una primera variante de programación en ANSI C se presenta, almacenada y ejecutada desde la memoria Flash en la zona de usuario y posteriormente su conversión a módulo del kernel para lograr una simetría lo más perfecta posible en lo que a período de muestreo de las señales se refiere.Utilizando protocolos de comunicación como http o socket es posible solicitar la información capturada y procesada en cualquiera de las variantes.

HARDWARE DEL SISTEMA DE ADQUISICIÓN DE DATOS

El DIL/NetPC constituye el núcleo del SAD y es el encargado de dirigir el proceso de muestreo y almacenar los valores discretizados para su posterior procesamiento. Consta de un microprocesador Intel StrongArm SA-1110 de 32 bit, memoria RAM, Flash y un controlador Ethernet. El Kernel del sistema operativo Linux es versión 2.4.18, la cual presenta servidores Web, FTP, y Telnet remoto vía Ethernet [4].Todo esto le aporta una potencialidad elevada al sistema para cualquier aplicación.

Sus datos fundamentales son:

CPU a 206 MHz de velocidad de reloj.

32 MByte de SDRAM y 16 MByte de FLASH.

Interfase Ethernet de 10/100 Mbps.

Dos puertos Serie de 16550 kbps.

20 Líneas Paralelas de Propósito General Entrada/Salida a alta velocidad.

Bus de Expansión de 8 bit de Entrada/Salida.

5 Entradas de Interrupciones y 4 Salidas de Chip Select.

Temporizador Programable “Watchdog”.

Tensión de Alimentación 3.3 V

El DIL/NetPC permite la conexión de dispositivos externos a través de líneas de puerto y del bus de expansión (figura 1). Los periféricos que se conecten al bus de expansión se direccionan por los CS1-4.

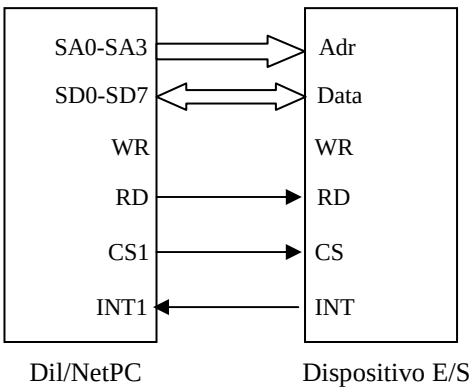


Fig. 1 Conexión de dispositivos por el bus de expansión.

Se requiere utilizar un conversor A/D que admita señales bipolares y que aporte al SAD una alta frecuencia de muestreo. Se selecciona el AD7891-1, el cual es un conversor de 12-bit, monolítico completo LC^2MOS que está combinado con un multiplexor de 8 canales en la entrada, de alta velocidad de conversión (2.3 μs), 2.5 V de CD de referencia interna y amplificador track/hold, con una opción de estructura de la interfaz, o sea, la transferencia de datos en forma serie o paralelo. Las partes operativas funcionan con el suministro de solo 5V de CD y entradas analógicas de ±5V y ±10V [5]. Conociendo el mapa de memoria del sistema embebido (Tabla I) se decidió conectar los 8 bit bajos del AD7891 a través del bus de expansión direccionado por el CS1 y los cuatro bits de mayor peso por líneas E/S paralelas**.**

Tabla 1

Mapa de memoria del DIL/NetPC DNP/1110

Direcc. Física	***Direcc. Virtual***	***Descripción***	***Acceso***
0x00000000- 0x07FFFFFF	0xE8000000- 0xEFFFFFFF	FLASH 16 Mbyte	L/E
0x20000000- 0x20FFFFFF	0xF6000000- 0xF6FFFFFF	Controlador Ethernet	L/E
0x30000000- 0x30FFFFFF	No tiene	Señal del Chip Select CS1	L/E
0x31000000- 0x31FFFFFF	No tiene	Señal del Chip Select CS2	L/E
0x32000000- 0x32FFFFFF	No tiene	Señal del Chip Select CS3	L/E
0x33000000- 0x33FFFFFF	No tiene	Señal del Chip Select CS4	L/E
0x80000000- 0xB7FFFFFF	0x80000000- 0xB7FFFFFF	Registros internos SA-1110	L/E
0xC0000000- 0xC7FFFFFF	0x00000000- 0x07FFFFFF	SDRAM 32 Mbyte	L/E

Para capturar seis señales simultáneamente se dota al sistema del circuito de muestro y retención LF398 [6], uno por canal, gobernados por una línea de puerto. En la fig. 2 se muestra el esquema general del SAD. La tarjeta de circuito impreso construida al efecto e incorporada al sistema embebido consta además de los convertidores de DC-DC NMH0515SC y NMH0505SC [7] que garantizan los niveles de tensión requeridos por los diferentes circuitos integrados y que se alimentan de la fuente original del DIL/NetPC.

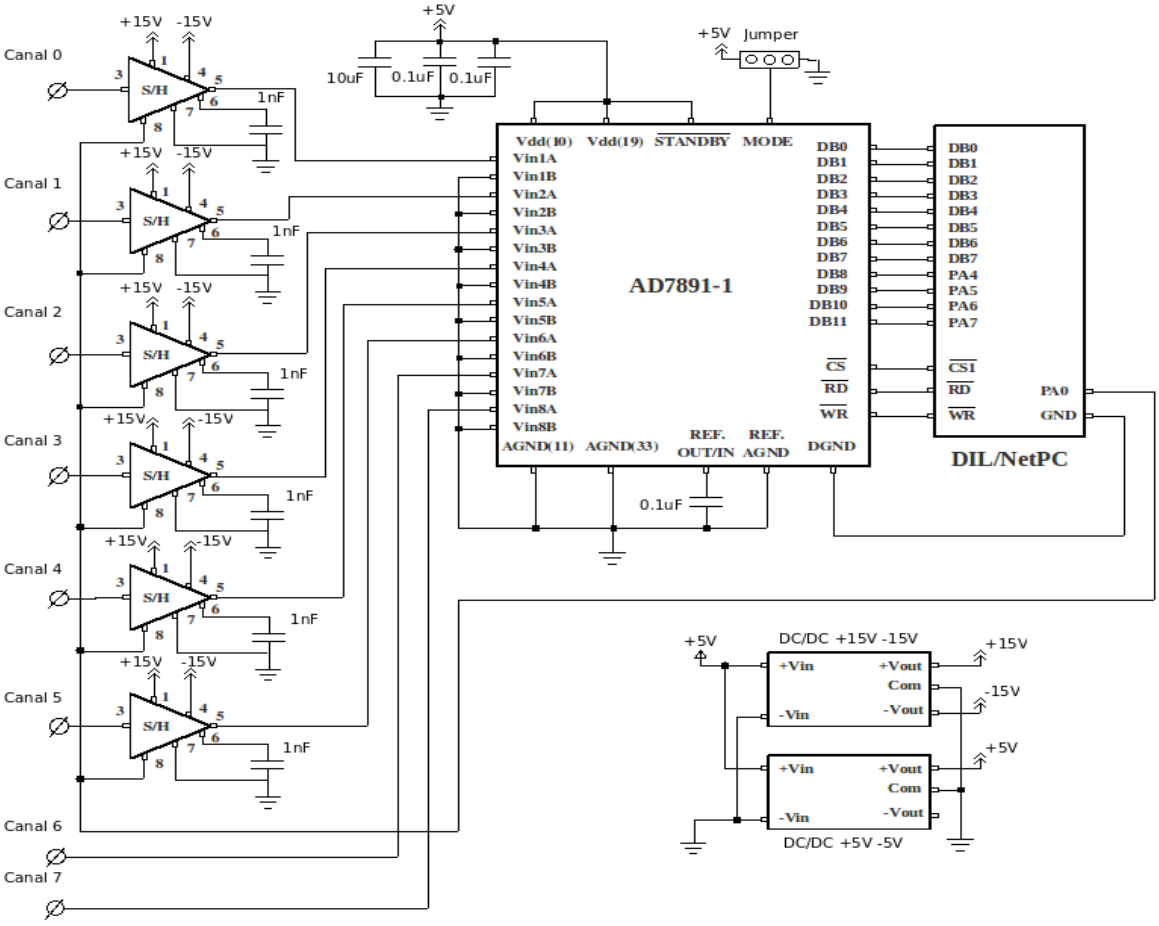


Fig. 2 Esquema de las conexiones de los elementos del SAD.

PROGRAMACIÓN DEL SISTEMA DE ADQUISICIÓN DE DATOS

Para poder interactuar en tiempo real con un hardware conectado físicamente al sistema empotrado y sin tener que recompilar el kernel de Linux es necesario contar con un software apropiado. Este debe ser capaz de dar una respuesta en su procesamiento que evite la existencia de demoras por encima de las requeridas por el hardware. Los requisitos de tiempo están fijados por el tiempo de conversión máximo del conversor A/D, por el tiempo máximo necesario para la correcta operación de los “samples and hold” y por la cantidad de canales a convertir. Para ello, el software debe poder medir los intervalos de tiempo necesarios lo más preciso posible. Lo anterior se cumple para cualquier variante de programación de que se disponga, siendo un elemento común.

El lenguaje de programación usado es el ANSI C, entre muchos de los posibles motivos porque este es el utilizado en la creación del núcleo del sistema operativo.

Los programas de la zona de usuario son los más habituales para la mayoría de los programadores. Este se ejecuta con los privilegios del usuario que lo ejecuta y con los recursos que este tiene asignado. Esto no ocurre así con un módulo del kernel [8 - 9], ya que este tiene los más altos privilegios del sistema, además de tener a su alcance todos los recursos disponibles.

Algunas de las características más notables en los módulos es que no se puede hacer uso de la biblioteca estándar de C, son de un ambiente concurrente por lo que es necesario tener cuidado con los requerimientos de

sincronización, además de que los medios de depuración son casi inexistentes. Es importante prestar atención a la administración de recursos. Un programa de usuario si se bloquea se puede finalizar con el administrador de procesos, si se bloquea un módulo del kernel, se congela todo el sistema. Se requiere de las fuentes del kernel y sus respectivos parches para poder compilar un módulo.

El programa que se ejecuta en la zona de usuario llamado “ADQUISICION” se encarga de dirigir el proceso de muestro, enviando un pulso por una línea de puerto a todos los S/H, lo que significa la orden de adquirir las señales en todos los canales. Al retirar la señal de los S/H se comienza la conversión análogo-digital de forma secuencial.
En el proceso iterativo las muestras son almacenadas en una estructura de datos declarada en memoria RAM. Una vez finalizado el muestreo se vuelcan los valores capturados en un fichero de datos ASCII. En la figura 3 se muestra el algoritmo de “ADQUISICION”.

A CONTINUACIÓN SE DETALLAN ALGUNOS ELEMENTOS TENIDOS EN CUENTA EN LA PROGRAMACIÓN DEL ALGORITMO

- Se invoca la función “mapmemory” para poder acceder a los registros de trabajo de los puertos.
- Adicionalmente se llama a “mapmemory” para montar la zona de memoria correspondiente a CS1 en el fichero de registro del sistema. Se debe tener en cuenta que el procesador se comunica con el conversor AD7891-1 a través del bus de expansión.
- Los tiempos de espera necesarios para fijar la señal del S/H y para el tiempo de conversión se logran a través de lazos “for” nulos. Una precisión del orden de pocos μseg no se logra con funciones de temporización del lenguaje C.
- Se requiere declarar una variable de 12 bits para almacenar el dato convertido, o sea que su rango se ajuste a la resolución del conversor, lo que permite obtener adecuadamente el módulo y el signo de la muestra adquirida.
- Para direccionar el canal que se desea convertir se utiliza la variable Configuración que contiene los parámetros fijos de la palabra de control del conversor A/D, adicionándole la dirección del canal a partir de operadores lógicos del lenguaje C.
- La función “gettimeofday” se utiliza para registrar el tiempo en que se fijan las señales para su posterior conversión. Esta función retarda la ejecución del programa disminuyendo la frecuencia de muestreo, si en el futuro no resulta indispensable dicho parámetro debe eliminarse lo que permitiría muestrear 648 puntos en 3 ciclos.
- Los valores almacenados en arreglos son transferidos a un fichero de texto (“file_Ua”) al finalizar los 3 ciclos de muestreo exceptuando el primero que se desecha por comprobarse experimentalmente no válido.

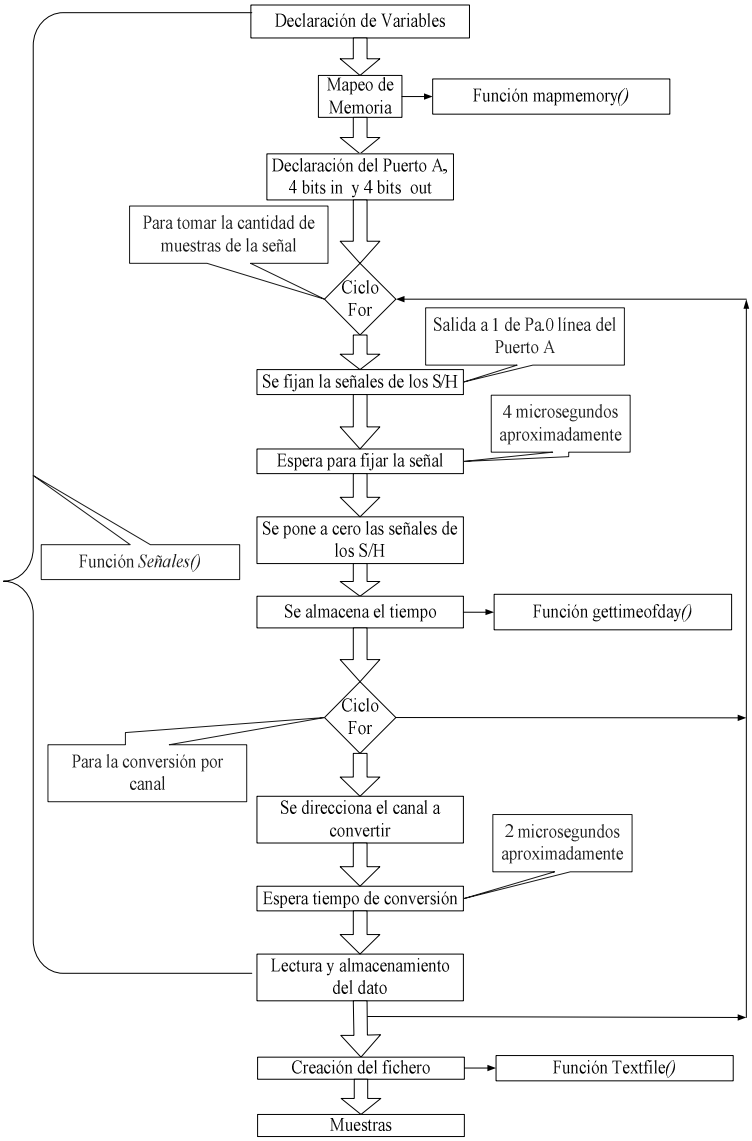
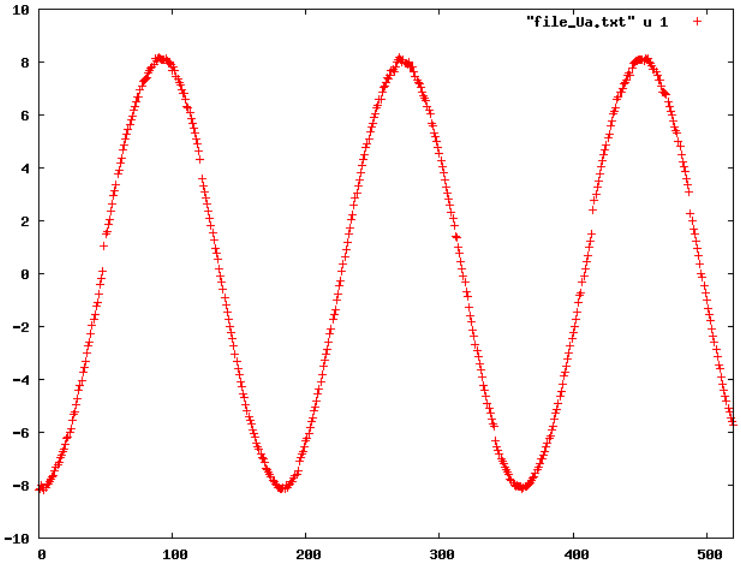


Fig. 3 Algoritmo del programa "ADQUISICION".

El uso de ciclos "for" para la medición de intervalos de tiempo resulto el mecanismo más sencillo y práctico para tiempos tan pequeños. Se comprobó experimentalmente que usando las tradicionales funciones del lenguaje C así como el uso de los ciclos, que no se garantiza el ajuste de tiempo predeterminado. El origen de este problema está dado en que los tiempos a medir se encuentran en el orden de la ejecución de las funciones, además, como el sistema operativo Linux es multitarea, se encuentran corriendo otros procesos que introducen retardos aleatorios. En el gráfico 1 se muestra el resultado de adquirir la señal del canal cero con 512 muestras, donde se observa la influencia de la inexactitud en los tiempos medidos.



Gráfica. 1 Muestras del canal cero con el programa de la zona de usuario.

Cuando se desea adicionar un segmento de código al kernel del sistema la variante elemental consiste en agregar ficheros fuentes al árbol del kernel y recompilar el mismo. De hecho el proceso de configurar el kernel está dado por la selección de los ficheros que serán incluidos en su compilación. No obstante, es posible adicionar ficheros durante la ejecución del sistema. El segmento de código que se incorpora al kernel de esta manera se conoce como módulos del kernel recargables. Los módulos recargables se pueden utilizar en un sin número de funciones pero básicamente se emplean en controladores de dispositivos, controladores del sistema de ficheros y en llamadas al sistema.

Ventajas en el uso de los módulos:

- No se requiere recompilar el kernel periódicamente lo que significa ahorro de tiempo y evitar la posibilidad de errores durante la reinstalación del mismo. El kernel base que se posee trabajando permanece intocable durante un largo período de tiempo.

- Los módulos representa una ayuda en el diagnóstico de los problemas del sistema. Un controlador de un dispositivo que se encuentra en el kernel base si presenta problemas puede detener el sistema y resulta muy difícil conocer la causa, sin embargo si el mismo controlador está configurado como módulo el problema se presenta después que el sistema está cargado y aunque lo detenga se puede precisar a qué se debe, eliminarlo temporalmente mientras se repara y continuar con el sistema en uso.

- No ocupan memoria hasta tanto sean utilizados, sin embargo todas las partes del kernel base se cargan y permanecen en memoria todo el tiempo.

- El mantenimiento y depuración de los módulos es relativamente rápido a partir del uso de algunos comandos.

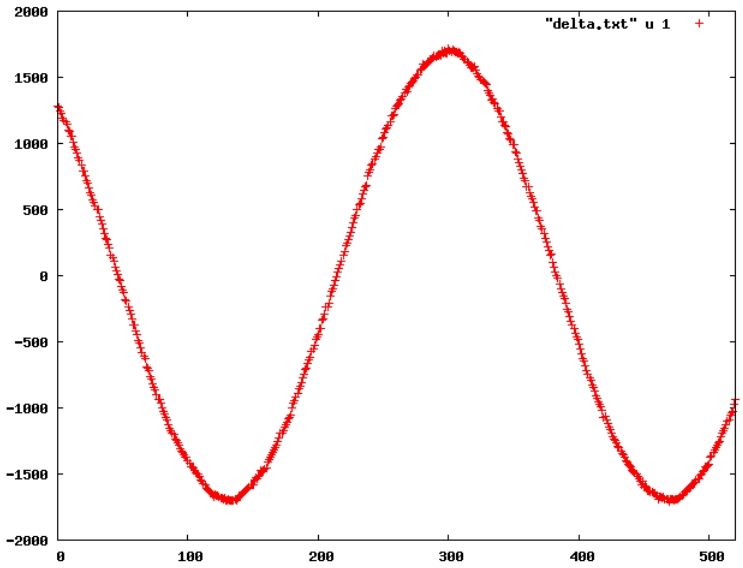
El módulo implementado es de tipo carácter, que no es más que aquel en donde el acceso es mediante un flujo de bytes, mecanismo similar al de un fichero. Los módulos de este tipo son los más sencillos para su comprensión. En este tipo de programación, a diferencia de la tradicional, no se crean los programas desde cero, sino que se parte de un esqueleto pre-diseñado.

Si nos paramos a ver el código en detalle de un módulo [8-9], no encontramos una función “main” sino dos llamadas “init_module” y “cleanup_module” que son ejecutadas al cargar y al descargar el módulo de memoria respectivamente, aunque desde Linux 2.6 hay algo más de libertad en este aspecto y podemos llamarlas como queramos e indicarlo mediante dos macros: “module_init” y “module_exit”.

Existen cuatro secciones fundamentales en el código, las cuales se ejecutan en etapas bien diferenciadas. Como el mecanismo de acceso al módulo es muy semejante al de un fichero como se dijo anteriormente, las secciones de código responden a cada una de las operaciones con ficheros. Entonces en cada parte se realiza lo siguiente:

- Apertura del fichero:- En esta sección se va llevando la cuenta de las veces que se abre el fichero, ello permite que se lleve un control del acceso al hardware para evitar que se intente utilizar por más de una petición simultánea. Esto se logra mediante un incremento de un contador.
- Lectura del fichero:- En esta sección se controla todas las operaciones del hardware encaminadas a obtener todas las muestras de los canales analógicos. Se tiene presente las declaraciones de las variables temporales y de la estructura de datos donde se almacena la información y consta del mecanismo para su transferencia de la zona de memoria del kernel a la zona de memoria del usuario.
- Escritura del fichero:- Esta sección se encuentra reservada para futuras implementaciones en la operación de otras líneas de puerto de propósito general.
- Cierre del fichero:- Usado para realizar decrementos al contador involucrado en la cantidad de accesos al módulo. Lo que indica si ya se puede realizar otra operación en el hardware sin interferir en una anterior.

Las temporizaciones se realizaron usando la función “udelay” brindada por el núcleo, gracias a la mejora en los privilegios y los recursos del sistema. Como se puede ver en el gráfico 2, se obtuvieron 520 muestras en el canal cero, donde no se observa saltos en la continuidad de la señal producto de la mejora de precisión en los tiempos medidos



Gráfica. 2 Muestras del canal cero con el módulo.

CONCLUSIONES

Como ventajas de la programación del módulo para controlar el hardware, se logra mejorar la resolución de la señal, aumentado la cantidad de muestras por ciclo. Ello tiene influencia directa sobre el procesamiento digital de la señal, llegándose a una mejor precisión y al uso de filtros antialias más sencillos. Se tiene un mayor acceso a los recursos, disminuyendo la inexactitud en el tiempo hasta valores despreciables.

Debido que la programación de los módulos es bastante diferente de la tradicional, en un principio es más difícil de llevar a cabo, a ello se le suma las difíciles condiciones de depuración, las cuales se realizan casi totalmente a partir de log del sistema operativo. Requiere prestar una mayor atención a la administración de los recursos, para evitar fallas catastróficas.

REFERENCIAS

[1] *Sistema Operativo*. [Consultado el: http://es.wikipedia.org/wiki/SO] enero del 2009.

[2] W. SMITH, S. Digital Signal Processing.*Digital Signal Processing* En 2da ed . California Technical Publishing, Enero 1999, Cap. ISBN 0-9660176-6-8.

[3] *Adquisición de Datos*. de http://es.wikipedia.org/wiki/Adquisici%C3%B3n_de_datos].Enero del 2009.

[4] *DIL/NetPC DNP/1110 Starter Kit User Manual". SSV Embedded Systems*. [Consultado el: http://www.dilnetpc.com/d1110um.pdf.] Septiembre del 2009.

[5] *LC2MOS 8-Channel, 12bit High Speed Data Acquisition System AD7891". Analog Devices*. de www.analog.com/static/imported-files/data_sheets/AD7891.pdf].Septiembre del 2010.

[6] *Sample-and-Hold amplifiers LF198, LF298 y LF398". Philips Semiconductors Linear Products.* de http://www.datasheetcatalog.com/datasheets_pdf/L/F/3/9/LF398N.shtml].Septiembre del 2010.

[7] *NMH Series: Isolated 2W Dual Output DC/DC Converters". C & D Technologies.* de http://es.farnell.com/murata-power-solutions/nmh0515sc/converter-dc-dc-sil-2w-15v/dp/1021491].Septiembre del 2010.

[8] RUBINI, A. A. C., JONATHAN *Linux Device Drivers* 2da ed, O´Reilly. ed. [Consultado el: http://www.oreilly.com/catalog/linuxdrive2/chapter/book. de].Septiembre del 2010

[9] POMERANTZ., P. J. S. Y. O. *The Linux Kernel Module Programming Guide* [Consultado el: http://tldp.org/LDP/lkmpg/2.6/html/.] Septiembre del 2010.

AUTORES

Josnier Ramos Guardarrama

Ingeniera Electricista, Facultad Eléctrica, Instituto Superior Politécnico José Antonio Echeverría,Cujae,La Habana, Cuba
e-mail: josnier@electrica.cujae.edu.cu

Mercedes Canasí Piñeiro

Ingeniera Electricista.Master en Ingeniería,Facultad Eléctrica,Instituto Superior Politécnico José Antonio Echeverría,Cujae, La Habana, Cuba.
e-mail: mcanasi@electrica.cujae.edu.cu

Raynel Díaz Santos

Ingeniero Electricista ,Facultad Eléctrica ,Instituto Superior Politécnico José Antonio Echeverría, Cujae. La Habana, Cuba.

e-mail: raynelds@electrica.cujae.edu.cu