Revista Educación en Ingeniería
ISSN 1900-8260
Junio de 2009 • N°. 7 • Pp 137-145 • Publicada en línea por la Asociación Colombiana de Facultades de Ingeniería -ACOFI- www.acofi.edu.co
Enviado: 30/04/2009 • Aprobado: 29/05/2009

TARJETA DE ADQUISICIÓN DE DATOS DE BAJO COSTO MULTIPLATAFORMA ORIENTADA A LA ENSEÑANZA DE LOS SISTEMAS DE CONTROL EN TIEMPO REAL

Alexander López Parrado
Universidad del Quindío, Armenia (Colombia)

Resumen

Este artículo presenta una descripción de dUQx, una tarjeta de adquisición de datos y control de bajo costo, desarrollada en el programa de ingeniería electrónica de la Universidad del Quindío, Armenia (Colombia). dUQx dispone de una conexión USB, es basada en un único microcontrolador ATmega168p y construida en su mayoría utilizando herramientas de desarrollo software libres. Adicionalmente dUQx puede ser accedida desde los sistemas operativos Linux®, QNX Neutrino® y Windows® utilizando los lenguajes de programación C y Java a partir de la misma interfaz de programación de aplicaciones. Teniendo en cuenta que dUQx es una herramienta pedagógica, también se permite su utilización desde MATLAB® sobre el sistema operativo Windows®. dUQx ha sido utilizada como equipo de laboratorio en la formación de los estudiantes en las áreas de automatización y sistemas en tiempo real del programa de ingeniería electrónica de la Universidad del Quindío.

Palabras clave: Adquisición de datos, sistemas de control, sistemas operativos, sistemas en tiempo real.

Abstract

This paper presents a description of dUQx, a low-cost data acquisition and control card developed within the electronics engineering program at the Universidad del Quindío, Armenia (Colombia). dUQx has an USB connection, uses a single microcontroller ATmega168p and was built using free software development tools. Additionally dUQx can be accessed from Linux®, QNX Neutrino® and Windows® operating systems using the programming languages C and Java from the same application programming interface. As dUQx is a pedagogic tool, it can be acceded from MATLAB® on the Windows® operating system. dUQx has been used as laboratory equipment in the training of students in the areas of automation and real time systems of the electronics engineering program at the Universidad del Quindío.

Keywords: Data acquisition, control systems, operating systems, real time systems.

Introducción

Una de las grandes preocupaciones de los docentes y estudiantes en los programas de formación en ingeniería electrónica tiene que ver con los equipos y materiales disponibles para la realización de prácticas de laboratorio; adicionalmente la formación basada en créditos académicos (Ministerio de Educación Nacional, 2001) requiere que el estudiante pueda adquirir algunas de estas herramientas a bajos costos facilitando el trabajo independiente en su hogar.

La tarjeta de adquisición de datos es uno de los equipos más utilizados por los ingenieros electrónicos en formación en las áreas de instrumentación, control automático y procesamiento digital de señales; el costo de un equipo de estos puede variar significativamente dependiendo del tipo de conexión, PCI (PCI Interest Group, 1998) o USB (Universal Serial Bus Group, 2007), la máxima frecuencia de muestreo soportada y la licencia del software de control para el computador.

Para construir una tarjeta de adquisición de datos de bajo costo orientada a la enseñanza se deben considerar dos aspectos importantes. El primero viene dado por la creciente popularización del computador portátil al interior de los centros de formación en ingeniería por parte de los estudiantes y docentes, esta tendencia hace atractiva la conexión USB para el trabajo en el laboratorio y el hogar. El segundo aspecto tiene que ver con el software de control, ya que en la mayoría de los programas de ingeniería electrónica se cuenta con licencias de paquetes como LabView® (National Instruments®, 2003) y MATLAB® (Mathworks®, 2008) cuyos costos pueden superar varios cientos de dólares, en el caso de MATLAB® los instructores pueden utilizar gratuitamente una versión estudiantil para propósitos académicos y suministrarla a sus estudiantes; una alternativa gratuita se encuentra en las aplicaciones de software libre GNU (Smith, 2009), sin embargo su instalación y utilización puede resultar tediosa y restringirse únicamente a sistemas operativos como Linux®.

En el mercado existen múltiples opciones para tarjetas de adquisición de datos con conexión USB, sin embargo en muchas universidades, como es el caso de la Universidad del Quindío, son utilizadas las fabricadas por las corporaciones estadounidenses National Instruments® y LabJack®, en particular son populares las referencias NI USB-6008 (National Instruments®, 2005) y LabJack U12 (LabJack®, 2004) de ambos fabricantes respectivamente. El modelo NI USB-6008 puede alcanzar frecuencias de muestreo de hasta 10 kHz cuando se capturan bloques de datos, en el caso de captura de una única muestra la frecuencia de muestreo máxima es de 150 Hz. Para el modelo LabJack U12 la frecuencia de muestreo máxima cuando se capturan bloques de datos es de 8192 Hz y para una sola muestra de 50 Hz. El costo de ambas tarjetas de adquisición de datos en Estados Unidos, al momento de escritura de este artículo, es de US$185 para el modelo NI USB-6008 y de US$129 para el modelo LabJack U12. Debe tenerse en cuenta que la captura usando bloques de datos es útil cuando se trata de sistemas de instrumentación, sin embargo, para el caso de sistemas de control en tiempo real se requiere procesar una muestra al tiempo de tal forma que no se introduzca un retardo de transporte indeseado (Burns et al., 2003) considerando lo anterior, en el contexto de los sistemas de control en tiempo real el indicador de interés es la frecuencia de muestreo máxima de una única muestra.

La tarjeta de adquisición de datos dUQX se presenta como una alternativa de muy bajo costo respecto a las mostradas anteriormente ya que se compone únicamente de un microcontrolador de gama media AVR de 8 bits ATmega168p, este microcontrolador es de fácil adquisición en Colombia a un costo aproximado de US$6. dUQx utiliza una conexión USB 1.1 de 1.5 Mbps (Universal Serial Bus Group, 2007) que permite frecuencias de muestreo de hasta 10 kHz usando bloques de datos, o de 349 Hz para una única muestra; el software de control de dUQx es de fácil utilización e instalación y es distribuido de forma gratuita bajo la licencia GPL (Smith, 2009) para los sistemas operativos Windows®, Linux® y QNX Neutrino®; en el caso de Windows® se provee un sencillo toolbox para MATLAB®. dUQx puede ser accedida en estas mismas plataformas usando lenguaje C a partir de la misma interfaz de programación de aplicaciones (API), de igual manera usando lenguaje Java. La posibilidad de acceso desde el sistema operativo QNX Neutrino

permite a dUQx ser utilizada en aplicaciones de control e instrumentación de tiempo real duro.

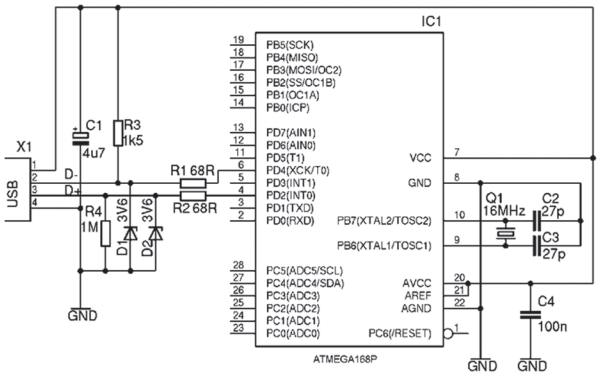
Metodología

Para definir los requerimientos que debe satisfacer dUQx se partió de las necesidades de los estudiantes y de las asignaturas en el área de automatización y la asignatura diseño de sistemas en tiempo real del programa de ingeniería electrónica de la Universidad del Quindío; se consideraron aspectos como: el costo de fabricación y el costo de los componentes electrónicos para los estudiantes, facilidad de acceso desde el computador en múltiples sistemas operativos de propósito general y de tiempo real usando software libre, acceso desde MATLAB® o LabView® y API en los lenguajes C y Java. Estos requerimientos permitieron elegir las mejores alternativas para el diseño de dUQx usando el lenguaje de descripción de sistemas SysML (Weilkiens, 2006), a continuación se describen los resultados más importantes tanto para el hardware como para el software.

Arquitectura del Hardware

Como se ha indicado anteriormente, el hardware de dUQx está compuesto principalmente de un microcontrolador ATmega168P (Atmel, 2008). Este microcontrolador de 8 bits y gama media de arquitectura AVR cuenta con 16 Kbytes de memoria FLASH y 1 Kbyte de memoria SRAM; dispone de un conversor análogo a digital (ADC) de 10 bits por aproximaciones sucesivas y 6 entradas análogas multiplexadas. La Figura 1 muestra el circuito básico de dUQx.

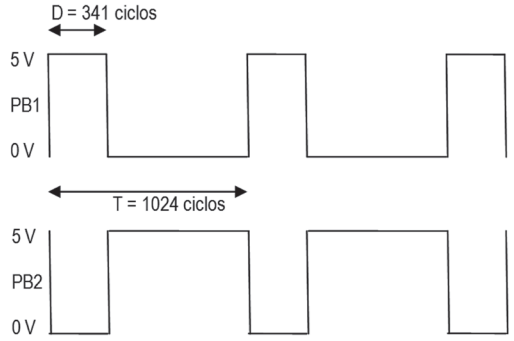
Figura 1. Circuito básico de dUQx.



La conexión USB sólo requiere el uso de dos pines de propósito general del microcontrolador, los componentes eléctricos y electrónicos son de muy bajo costo y se restringen a resistencias, condensadores, oscilador en base a cristal y diodos zenner. Los pines 23 a 28 son destinados como entradas análogas de dUQx.

Debido a que no se tiene disponible un conversor digital a análogo (DAC) en el microcontrolador, dUQx emula la generación de señales análogas mediante dos señales PWM (Pulse Width Modulation) complementarias de una frecuencia de 16 MHz/1024 = 15625 Hz, esto con el fin de proveer una resolución total de 10 bits para los cambios en el ciclo útil de la señal. La Figura 2 muestra las señales PWM generadas por dUQx, en este caso el valor de la componente de DC de la señal en el pin PB1 es 5 v x 341 / 1024 = 1.67 v y en el pin PB2 es de 5 v – 1.67 v = 3.33 v. Las señales PWM complementarias son generadas por los pines 14 y 15.

Figura 2. Señal PWM generada por dUQx.



Los pines no utilizados del microcontrolador son destinados para un puerto digital de propósito general bidireccional de 10 bits dado por los pines 2, 3, 5, 11, 12, 13, 14, 19, 18 y 17.

Dada la simplicidad del hardware de dUQx la mayor parte de su funcionalidad recae en el firmware programado en el microcontrolador y al software de control del computador.

Firmware de dUQx

El firmware programado en el microcontrolador de dUQx es basado en el proyecto Virtual USB de la

compañía de desarrollo de software alemana Objective Development (Objective Develpment GmbH, 2009). El firmware satisface un subconjunto de la versión 1.1 de la especificación USB (Universal Serial Bus Group, 2007) para una velocidad de 1.5 Mbps.

El firmware hace de dUQx un dispositivo USB tipo vendor-specific que soporta únicamente transferencias de control a través del punto final cero (Universal Serial Bus Group, 2007), utiliza como VID (Vendor Identification) 0x016c0 y PID (Product Identification) 0x05dc (Objective Develpment GmbH, 2009). Se determinó un conjunto de 12 peticiones específicas para acceder a dUQx desde el computador, dentro de las más importantes se encuentran: DUQX_DIGITAL_CONFIGURE, configurar sentido del puerto digital; DUQX_DIGITAL_READ, leer los pines del puerto digital; DUQX_DIGITAL_WRITE, escribir en los pines del puerto digital; DUQX_ADC_READ_SINGLE, leer una sola muestra de un canal análogo; DUQX_DAC_WRITE_SINGLE, escribir una sola muestra en el canal análogo y DUQX_ADC_BUFFER_READ, leer bloque de muestras de un canal análogo. Estas peticiones son ubicadas en el campo bRequest del paquete de configuración de la transferencia de control (Universal Serial Bus Group, 2007). Con las peticiones restantes es posible modificar la frecuencia de muestreo en el modo de captura por bloques, configurar el tamaño de palabra de las muestras a 8 o 10 bits, entre otras.

Para el caso de las operaciones de lectura DUQX_DIGITAL_READ y DUQX_ADC_READ_SINGLE la transferencia de control involucra una parte de datos de entrada de longitud dos, wLength = 2, esto para transferir en dos bytes los dos bits de mayor peso y los ocho bits de menor peso del número de diez bits correspondiente a la lectura. Para el caso de DUQX_ADC_READ_SINGLE la transferencia puede ser de longitud uno, wLength = 1, cuando el ADC es configurado para muestras de 8 bits.

Las operaciones de escritura DUQX_DIGITAL_CONFIGURE, DUQX_DIGITAL_WRITE y DUQX_DAC_WRITE_SINGLE no involucran una parte de datos ya que en cualquier caso el número de diez bits es enviado en el campo wValue del paquete de configuración de la transferencia de control (Universal Serial Bus Group, 2007).

La petición DUQX_ADC_BUFFER_READ involucra una parte de datos de entrada de longitud 250 máximo gracias a una cola de 250 bytes implementada en el microcontrolador, esto equivale a 125 muestras de 10 bits o 250 muestras de 8 bits. Las solicitudes de captura en este modo deben hacerse en paquetes de hasta 250 muestras desde el computador, en este caso es posible configurar la frecuencia de muestreo desde 61 Hz hasta 8064 Hz. La cola permite almacenar hasta 4 segundos de la señal capturada de tal forma que no se pierda ninguna muestra debido a la latencia del software de control.

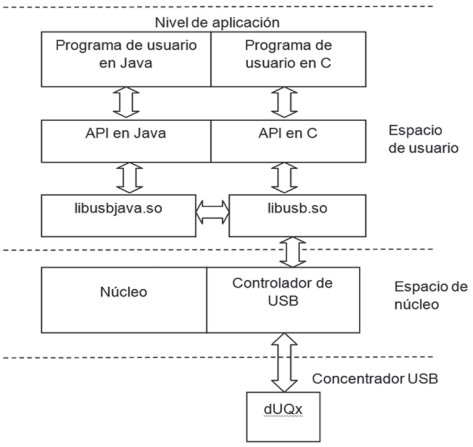
Arquitectura del software para acceso multiplataforma

Se definió una API en lenguaje C común para los sistemas operativos Linux, QNX Neutrino y Windows que permite el acceso a dUQx de forma sencilla. Igualmente para el lenguaje Java en Linux y Windows. No se realizó una implementación en Java para QNX Neutrino debido a que las máquinas virtuales comerciales para este sistema operativo no son libres, esto limitaría la distribución de forma gratuita de la especificación de dUQx.

Abstracción para Linux®

El acceso a dUQx desde Linux se logra por medio de la librería GNU libusb (Drake, 2008) tanto usando lenguaje C como Java. Esta librería permite el acceso a cualquier dispositivo USB en espacio de usuario sin necesidad de adicionar un controlador como módulo al núcleo de Linux (Salzman et al., 2003). En el caso de Java se requiere una librería dinámica adicional (University of Applied Sciences of Technology, 2008) que permite el acceso mediante código nativo a las primitivas de libusb desde Java. La Figura 3 muestra la arquitectura del software de control de dUQx bajo Linux®.

Figura 3. Arquitectura del software de control de dUQx en Linux

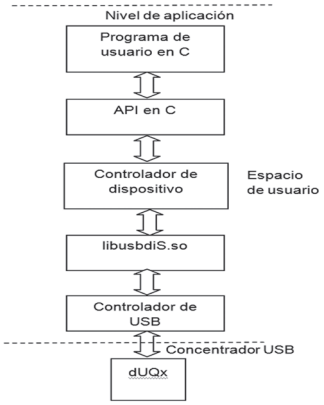


Las librerías mencionadas anteriormente son utilizadas en el espacio de usuario de Linux. La librería libusb realiza la interfaz entre el controlador genérico USB de Linux y la aplicación. Las APIs de programación en lenguaje C y Java son genéricas y no dependen del sistema operativo.

Abstracción para QNX Neutrino®

Teniendo en cuenta que la arquitectura del sistema operativo de tiempo real QNX Neutrino® es micro núcleo (QNX Software Systems GmbH, 2008), la totalidad de las aplicaciones son ejecutadas en el espacio de usuario, en este caso el controlador de dUQx fue escrito usando las primitivas de la librería dinámica libusbdiS (QNX Software Systems GmbH, 2007). La Figura 4 muestra la arquitectura del software de control de dUQx bajo QNX Neutrino.

Figura 4. Arquitectura del software de control de dUQx en QNX Neutrino



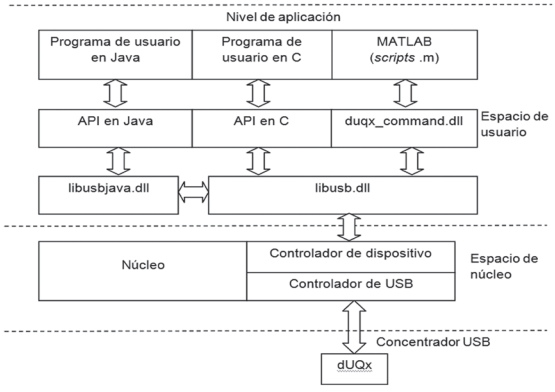
Fue necesario desarrollar un controlador para dUQx y la interfaz para la API multiplataforma en lenguaje C común a Linux® y Windows®. El controlador fue codificado para emular la API de libusb usando las primitivas de libusbdiS, bajo este nivel de abstracción es posible garantizar la ejecución multiplataforma del software de control de dUQx en lenguaje C.

Abstracción para Windows®

Desde Windows es posible acceder a dUQx usando lenguaje C, Java y MATLAB®. En cualquier caso el acceso se logra gracias a la librería GNU libusb-win32 (Meyer, 2007) la cual es derivada de libusb. De forma similar a como se hizo para Linux, se requirió de la librería libusbjava con el fin de acceder a dUQx desde Java usando el código nativo de libusb-win32.

En el caso de MATLAB, con el fin de garantizar la libre distribución de dUQx, se utilizó la herramienta GNU gnumex (Birge et al., 2008) para generar una librería dinámica con código nativo accesible desde MATLAB mediante el compilador GNU de C para Windows MinGW (MinGW Team, 2008). La Figura 5 muestra la arquitectura del software de control de dUQx bajo Windows®.

Figura 5. Arquitectura del software de control de dUQx en Windows®



El papel más importante es desempeñado por la librería libusb-win32 la cual permite el acceso desde lenguaje C y MATLAB, para este último a través de la librería duqx_command y un conjunto de scripts como parte del toolbox de dUQx. En el

caso de Windows® se requiere de un controlador de dispositivo que se adicione al núcleo de Windows®, éste fue construido fácilmente con una utilidad provista en el proyecto libusb-win32.

Análisis y discusión de Resultados

En esta sección se discutirán el conjunto de pruebas y experimentos realizados con dUQx que permitieron obtener indicadores numéricos de desempeño sobre los sistemas operativos considerados y lenguajes de programación soportados. Adicionalmente se relatan las experiencias vividas durante la asignatura diseño de sistemas en Tiempo real por los estudiantes y el docente después del uso de dUQx como herramienta pedagógica.

Pruebas de desempeño en Windows® y Linux®

Se realizaron pruebas en Windows® XP y la distribución de Linux Ubuntu® 8.04 sobre un computador con procesador Intel® Core 2 Duo de 2 GHz. Se midieron las frecuencias de muestreo máximas cuando se captura desde programas escritos en lenguaje C y Java. Para el caso del modo única muestra se utilizó el ADC a 10 bits, mientras que para el modo bloque se utilizó el ADC a 8 bits con bloques de 250 muestras cada uno.

Desempeño usando lenguaje C

En el caso de Windows®, un programa escrito en lenguaje C puede capturar muestras hasta una frecuencia máxima de 333 Hz en el modo única muestra. Cuando se captura en el modo bloque la frecuencia de muestreo alcanzó un valor de 7142 Hz.

En Linux el desempeño fue menor que el obtenido en Windows en el modo única muestra ya que la frecuencia de muestreo máxima fue de 250 Hz. Para el modo de bloque el comportamiento fue similar con una frecuencia de muestreo máxima de 7142 Hz.

Desempeño usando lenguaje Java

La velocidad en la frecuencia de muestreo usando Java en Windows® fue de 250 Hz en el modo única muestra y de 285 Hz en Linux.

Para el caso del modo bloque la frecuencia de muestreo alcanzada en Windows® fue 8064 Hz, por otra parte en Linux la frecuencia de muestreo solo alcanzó un valor de 7142 Hz.

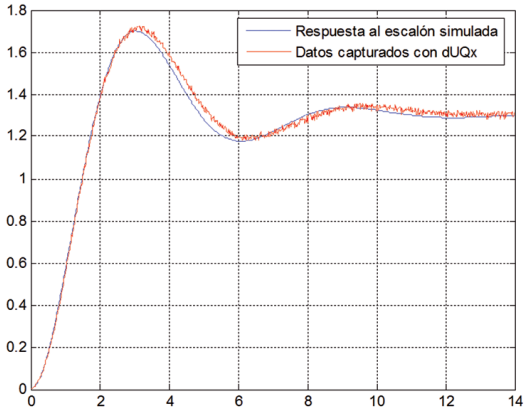
Desempeño en MATLAB®

Las pruebas en MATLAB® se realizaron bajo Windows® XP en un computador con procesador Intel Core 2 Duo de 2 GHz. Uno de los experimentos consistió en obtener la respuesta ante una señal escalón de un sistema análogo de segundo orden construido a partir de amplificadores operacionales. La ecuación (1) muestra la función de transferencia del sistema.

$$Gp(s) = \frac{1.576}{S^2 + 0.7734S + 1.212} \quad (1)$$

La señal escalón fue emulada con una de las salidas PWM de dUQx y las muestras fueron capturadas de la salida del sistema usando el modo única muestra de 10 bits. La Figura 6 muestra en color azul la respuesta al escalón obtenida mediante simulación en MATLAB, la curva roja representa la respuesta real del sistema análogo obtenida con dUQx.

Figura 6. Respuesta al escalón de un sistema análogo usando dUQx.



En este caso el periodo de muestreo es de 14 ms y se capturan 1000 muestras. Es destacable la capacidad de emulación de señales análogas que provee PWM con una frecuencia lo suficientemente alta cuando no se dispone de un DAC como es el caso del microcontrolador ATmega168p.

Desde MATLAB, los resultados experimentales mostraron que se puede alcanzar una frecuencia de muestreo de 250 Hz cuando se realiza la adquisición en el modo de única muestra. En el caso de la adquisición de un bloque de datos de 250 muestras de 8 bits cada una se alcanzó una frecuencia de muestreo de 8064 Hz.

En cualquier caso la frecuencia de muestreo máxima es mucho mayor en el modo bloque debido a que USB tiene un rendimiento mayor cuando se transfiere un bloque de datos al reducirse la sobrecarga debida a las cabeceras (Universal Serial Bus Group, 2007) además esto no sería posible sin la presencia de la cola en el firmware del microcontrolador.

Pruebas de tiempo real usando QNX Neutrino®

Gracias a la alta predictibilidad del micro núcleo de tiempo real de QNX Neutrino®, es posible garantizar tasas de muestreo constantes con desviaciones mínimas tanto en el modo de única muestra como de bloque.

Se realizó un experimento sobre un computador con procesador AMD Athlon de 892 MHz, se midió la tasa de muestreo creando un proceso de tiempo real con la prioridad más alta (QNX Software Systems GmbH, 2008) (Burns, et al., 2003). El experimento mostró que dUQx puede muestrear en el modo única muestra a una frecuencia de hasta 349 Hz. En el caso del modo bloque la frecuencia alcanzada fue de 10.1 kHz. Las frecuencias de muestreo fueron más altas que en el caso de Windows y Linux, esto es gracias a la baja latencia en la atención de interrupciones del hardware de los sistemas operativos de tiempo real como QNX Neutrino (QNX Software Systems GmbH, 2008).

Ocupación del firmware

El firmware de dUQx ocupa 3128 bytes de memoria FLASH del microcontrolador, esto representa el 19 % de los 16 Kbytes de memoria de programa disponible. En el caso de la memoria SRAM la ocupación es el 30 % del total de 1024 bytes. La memoria disponible tanto de programa y de datos permite realizar mejoras al firmware de dUQx para las futuras versiones sin necesidad de migrar a un microcontrolador de mayor gama.

Experiencias con la asignatura diseño de sistemas en tiempo real

La iniciativa para el desarrollo de la especificación de dUQx se originó durante el transcurso de la asignatura diseño de sistemas en tiempo real del programa de ingeniería electrónica de la Universidad del Quindío en el año 2008. Durante ese año se formalizó el diseño y para el primer semestre del año 2009 la especificación estuvo lista en su versión 1.0. A la fecha de escritura de este artículo dUQx se encuentra en la versión 1.2.

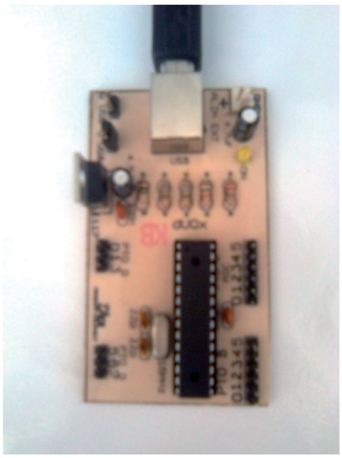
Los doce estudiantes de la asignatura diseño de sistemas en tiempo real del primer semestre del 2009 construyeron 6 tarjetas de adquisición de datos que fueron utilizadas en 2 proyectos. En el primero de ellos los estudiantes diseñaron e implementaron un sistema de control moderno bajo el enfoque de ejecutivo cíclico (Burns et al., 2003) sobre el sistema operativo de tiempo real QNX Neutrino®. El segundo proyecto consistió en un sistema de procesamiento digital de señales usando Java sobre el sistema operativo Windows. A la fecha de escritura de este artículo los estudiantes se encuentran trabajando en la implementación de un sistema de control distribuido (Burns et al., 2003) sobre Linux-RT (Molnar, 2009) que hace uso de dUQx.

Se obtuvieron resultados bastante satisfactorios desde el punto de vista académico después de la realización de los dos proyectos, ya que no hubo limitaciones en la cantidad de equipos para la asignación de las prácticas de laboratorio, de este modo los grupos de trabajo no tuvieron que ser tan numerosos lográndose así un incremento significativo en el rendimiento de los estudiantes, esto se evidenció en una reducción de aproximadamente dos semanas del tiempo de entrega de los proyectos. Los estudiantes manifestaron el haber trabajado con bastante comodidad al poder tener en sus manos una tarjeta de adquisición de datos para su uso en la universidad y el hogar, sin requerir la reservación con anterioridad del espacio de laboratorio y del propio

equipo. El hecho de haber podido adquirir la tarjeta de adquisición de datos a un costo tan bajo motivó a otros docentes en el área de automatización y control para su utilización dentro de sus asignaturas.

La Figura 7 muestra una implementación de dUQx desarrollada por uno de los estudiantes de la asignatura diseño de sistemas en tiempo real.

Figura 7. Tarjeta de adquisición de datos dUQX construida por un estudiante.



Las dimensiones de la tarjeta son 375 mm x 445 mm, se pueden apreciar los conectores disponibles para las entradas y salidas análogas así como para el puerto digital de 10 bits. El estudiante que construyó la tarjeta manifestó haber invertido cerca de US$12 en los materiales.

Conclusiones

Los resultados experimentales sobre los diferentes sistemas operativos demostraron que dUQx tiene un desempeño comparable y en algunos casos superior al exhibido por equipos de características similares y de costo mucho mayor como son las tarjetas de adquisición de datos NI USB-6008 y Labjack U12 de uso común en los laboratorios de los programas de ingeniería electrónica de muchas universidades en Colombia.

Las experiencias docentes logradas con el uso de dUQx dentro de la asignatura diseño de sistemas en tiempo real mostraron que el rendimiento y motivación de los estudiantes son incrementados cuando estos disponen de sus propios equipos para el trabajo de laboratorio en la universidad y hogar, y más aún cuando estos pueden ser adquiridos o construidos fácilmente a un bajo costo.

El uso de herramientas de desarrollo libres permite que el firmware y el software de dUQx puedan ser distribuidos libremente sin ningún costo a toda la comunidad académica de los programas de formación en electrónica de Colombia, adicionalmente la fácil consecución de los materiales y su bajo costo hace posible la construcción de dUQx a estudiantes de bajos recursos económicos.

El software de control multiplataforma y multilenguaje no restringe la utilización de dUQx a los sistemas operativos Windows® y Linux®, en este caso se dio soporte para un sistema operativo de tiempo real de amplio uso en la industria como lo es QNX Neutrino®. Para facilitar el uso de dUQx en la etapa de diseño de los proyectos de los estudiantes se provee un toolbox de libre distribución para MATLAB que facilita la realización de pruebas rápidas.

El mejor desempeño de dUQx se obtuvo en el sistema operativo de tiempo real QNX Neutrino®, en este caso las frecuencias de muestreo máximas fueron superiores a las logradas con la tarjeta de adquisición de datos NI USB-6008. Estos resultados perfilan a dUQx como una buena opción libre para la dotación de tarjetas de adquisición de datos de bajo costo en los laboratorios de los programas de ingeniería electrónica en Colombia.

Referencias

Atmel (2008). 8-bit Microcontroller with 4/8/16/32K Bytes In-System Programmable Flash ATmega48P/V ATmega88P/V ATmega168P/V ATmega328P. Consultado el 4 de julio de 2008 en http://www.atmel.com/dyn/products/product_card.asp?PN=ATmega168P.

Birge, J., Jonasson, K. and Brett M. (2008). Compiling Matlab mex files with gcc for Windows. Consultado el 13 de febrero de 2009 en http://gnumex.sourceforge.net/.

Burns. A y Wellings A. (2003). Sistemas de Tiempo Real y Lenguajes de Programación. Addison Wesley, Madrid, pp. 3 -9, 513 - 514, 573 -575.

Drake, D. Libusb-1.0 (2008). Consultado el 17 de diciembre de 2008 en http://libusb.wiki.sourceforge.net/.

LabJack (2004). LabJack U12 User's Guide. Consultado el 13 de febrero de 2009 en http://www.labjack.com/labjack_u12_downloads.php.

Mathworks (2008). MATLAB Getting Started Guide. Consultado el 14 de agosto de 2008 en http://www.mathworks.com/access/helpdesk/help/pdf_doc/matlab/.

Meyer, S. (2007). libusb-win32. Consultado el 14 de agosto de 2008 en http://libusb-win32.sourceforge.net/.

MinGW Team (2008). Minimalist GNU for Windows. Consultado el 13 de marzo de 2008 en http://www.mingw.org/.

Ministerio de Educación Nacional (2001). Sistema de Créditos Académicos. Consultado el 20 de noviembre de 2008 en http://www.mineducacion.gov.co/1621/article-87727.html.

Molnar, I. (2009). The CONFIG_PREEMPT_RT patch set. Consultado el 15 de febrero de 2009 en http://rt.wiki.kernel.org/index.php/CONFIG_PREEMPT_RT_Patch.

National Instruments (2003). LabView User Manual. Consultado el 14 de agosto de 2008 en www.ni.com/pdf/manuals/.

National Instruments (2005). User Guide and Specifications USB-6008/6009. Consultado el 13 de febrero de 2009 en http://sine.ni.com/nips/cds/view/p/lang/es/nid/14604.

Objective Development GmbH (2009). Virtual USB Port for AVR Microcontrollers. Consultado el 13 de agosto de 2008 en http://www.obdev.at/products/vusb/index.html.

PCI Special Interest Group (1998). PCI Local Bus Specification Revision 2.2. Consultado el 13 de febrero de 2009 en http://www.pcisig.com/specifications/.

QNX Software Systems GmbH & Co. KG (2007). QNX® Neutrino® Device Drivers, Universal Serial Bus (USB) Devices. Consultado el 4 de julio de 2008 en www.qnx.com/download/download/18824/.

QNX Software Systems GmbH & Co. KG (2008). QNX® Neutrino® RTOS System Architecture. Consultado el 4 de julio de 2008 en http://www.qnx.com/download/download/14695/.

Salzman, P. J, and Pomerantz, O. (2003). The Linux Kernel Module Programming Guide. Consultado el 20 de noviembre de 2008 en http://mirrors.kernel.org/LDP/guides.html.

Smith, B (2009). A Quick Guide to GPLv3. Consultado el 13 de febrero de 2009 en http://www.gnu.org/licenses/quick-guide-gplv3.html.

Universal Serial Bus Group (2007). Universal Serial Bus Specification Revision 2.0. Consultado el 20 de noviembre de 2008 en http://www.usb.org/developers/docs.

University of Applied Sciences of Technology NTB (2008). Java libusb / libusb-win32 wrapper. Consultado el 14 de agosto de 2008 en http://libusbjava.sourceforge.net/wp/.

Weilkiens, T. (2006). Systems Engineering with SysML/UML Modeling, Analysis, Design, Morgan Kaufmann, Burlington, pp. 33 – 37, 226 - 238.

Sobre el autor

Alexander López Parrado.
Ingeniero Electrónico Universidad del Quindío 2002, Magíster en Ingeniería Énfasis Ingeniería Electrónica Universidad del Valle 2009. Docente del programa de Ingeniería Electrónica e Investigador del Grupo de Procesamiento Digital de Señales y Procesadores (GDSPROC) de la Universidad del Quindío desde el año 2002. parrado@uniquindio.edu.co. Universidad del Quindío, Carrera 15 Calle 12N, Facultad de Ingeniería, Programa de Ingeniería Electrónica, Armenia (Colombia).