Una plataforma tiempo real para ejecutar algoritmos de control

A real-time platform to execute control algorithms

Mejía, Carlos[1]; Ríos, Addison[2*]; León, Leandro[1] e Hidrobo, Francisco[3]
[1]CEMISID, Dpto. de Computación [2] ; Dpto. de Sistemas de Control [3]; Dpto. de Física
Escuela de Ingeniería de Sistemas, Universidad de Los Andes
Mérida 5101, Venezuela
*ilich@ula.ve

Recibido: 29-05-2009 Revisado: 10-06-2010

Resumen

En este estudio se desarrolla una nueva plataforma de tiempo real para el PC104 basada en software libre. La idea es disponer de un sistema operativo tiempo real para soportar tareas exigentes de control de procesos. El sistema se ha construido utilizando un kernel de Linux básico y una interfaz para aplicaciones de tiempo real a partir de RealTime Application Interface for Linux *(RTAI), la cual permite desarrollar aplicaciones con estrictas exigencias de tiempo sobre cualquier sistema Linux. La construcción del sistema, en cuanto a requerimientos de hardware, se fundamenta en los dispositivos embebidos basados en la arquitectura del PC104, disponible con todas las características de entrada-salida, medios de comunicación, módulos de conexión de red, etc. A partir de esas características, se configura un nuevo* kernel *del sistema de forma óptima, el cual necesita primero ser actualizado y mejorado para soportar las aplicaciones desarrolladas sobre RTAI. Con este nuevo sistema se tiene dominio de todas las potencialidades del sistema operativo tiempo real para el diseño e implementación de sofisticados algoritmos para la supervisión y control de procesos físicos. El sistema desarrollado tiene la ventaja de ser de bajo costo, pueda implantarse de forma sencilla, no requiera licencias de software, no depende de un fabricante específico, puede operar en ambientes industriales de altas exigencias y, lo más relevante, soporte la implementación de aplicaciones de control y automatización industrial.*

Palabras clave: Sistemas empotrados, PC104, sistemas de tiempo real, linux rtai, control de procesos.

Abstract

In this paper, the development process to implement a real -time platform for the PC104 based on free software is presented. The idea is to arrange a real-time operating system to execute the process control tasks. The system has been made using a basic kernel of Linux and the Real-Time for Application Interface (RTAI), in order to develop applications with strict time limitations on any Linux system. The hardware requirements for the system are related with the embedded devices based on PC104 architecture; these support all characteristics of I/O, communications, networking, etc. Using these characteristics, we generate a new optimal kernel for the system, which needs first being updated and improved to support the applications developed on RTAI. With this real time platform, we can use all potentialities of the operating system to design and implement sophisticated algorithms for the supervision and control of physical process. The platform implementation has the advantage of being of low cost, can be made of a simple way, does not require software licenses, does not depend on a specific manufacturer, can operate in industrial environment of high exigencies and, most important, it allows the implementation of control applications and industrial automation.

Key words: Embedded systems, PC104. real-time systems, linux rtai, process control.

1 Introducción

Las técnicas de control han estado evolucionando y avanzando a medida que cambian las exigencias de producción en procesos industriales. Este avance ha sido fortalecido por el desarrollo de las tecnologías de información y comunicación (TIC), de manera que los procesos productivos pueden adaptarse a las nuevas exigencias de rendimiento, comunicación, costos, integración, e incluso a los requerimientos de protección ambiental. En este sentido, los

procesos productivos han demandado la construcción e implementación de sistemas dedicados y de aplicaciones específicas, como los sistemas empotrados ó embebidos, ya que estos tienen un menor consumo de energía, tienen mejor velocidad de respuesta, ofrecen mayor seguridad y pueden construirse por módulos según los requerimientos de producción, reduciéndose los costos.

Los sistemas empotrados tienen casi las mismas funciones que un computador personal, excepto que son construidos para aplicaciones muy específicas (Abbot, 2006). Además, estos sistemas requieren de un sistema operativo capaz de soportar exigencias de tiempo real y que sea confiable para la manipulación de datos, dando respuestas en los intervalos de tiempo establecidos. Para cumplir con esta exigencia, dichos sistemas operativos deben ser predecibles (deterministas). En el mercado existe cierta variedad de estos sistemas de tiempo real, pero en su mayoría son costosos y cualquier necesidad en particular por parte de ellos requiere un costo adicional, por lo que es mas útil recurrir a un sistema de tiempo real basado en software libre, con las ventajas asociadas en costos y en la capacidad de adaptación a necesidades específicas.

En (Gusenbauer et al., 2005) se presenta una plataforma similar solo para aplicaciones experimentales, por lo que no han tenido que reducir u optimizar el tamaño de la distribución de Linux que conforma la plataforma de tiempo real del PC104, su sistema dispone de una gran cantidad de memoria y capacidad de almacenamiento (Disco duro de 80 Gb); pero esto limita su rango de utilidad, ya que al usar dispositivos mecánicos se incrementa la posibilidad de fallos. Ese sistema presenta, además, limitaciones ya que no puede ser usado en áreas con temperaturas críticas.

Así, en este trabajo se describe la implantación de una plataforma para supervisión y control, basada en dispositivos estándar y herramientas de software abiertos (PC industrial, I/O, Sistema operativo, SoftPLC). Esta plataforma tiene la capacidad para prestar funcionalidades en cuanto a conectividad, integración, mantenibilidad y aplicaciones de supervisión y control, mediante el desarrollo de software basado en tecnologías de comunicación TCP/IP, de bases de datos y de interfaz WEB.

Con esta plataforma se intenta proveer de un soporte básico para el desarrollo de sistemas capaces de dotar de inteligencia los procesos de adquisición de grandes cantidades de variables, facilitar su procesamiento, ejecutar elaborados algoritmos de control, visualizar el comportamiento de los procesos, en un todo sin necesidad de recurrir a sofisticados ambientes especializados como los Sistemas de Control Distribuido (DCS) o los Sistemas de Adquisición de Datos y Control Supervisorio (SCADA). Al estar soportados por una filosofía de herramientas libres y de estándares abiertos, el sistema facilita su integración con otros ambientes de control y supervisión. Además, permite la escalabilidad, conectividad y mantenibilidad, a objeto de disponer de toda la inteligencia en sitio para ejecutar las funciones de control, supervisión y optimización en tiempo real. Así, el sistema que se presenta es una herramienta de bajo costo, que pueda implantarse de forma sencilla, no requiera licencias de software, no depende de un fabricante específico, opera en ambientes industriales de altas exigencias y soporte la implementación de aplicaciones de control y automatización industrial.

El resto del artículo se organiza de la siguiente manera: la sección 2 presenta los elementos básicos que dan soporte a la implantación realizada.

Luego, en la sección 3 se describe la plataforma implantada y los procesos necesarios para ponerla en funcionamiento. La sección 4 contiene la descripción y las pruebas realizadas con una aplicación que sirve de ejemplo para el uso de la plataforma.

Finalmente, en la sección 5 se presentan los comentarios concluyentes de este trabajo.

2 Soporte básico para la plataforma

2.1 Hardware

El hardware utilizado para la implantación de la propuesta se basa en el estándar PC104. Este estándar define el formato de la placa base y del bus del sistema (una adaptación del bus ISA) para un computador empotrado que puede satisfacer las necesidades de las aplicaciones embebidas e industriales. Su nombre deriva de su arquitectura PC y del conector de 104 pines (Consortium, 2003).

Seleccionando la tecnología PC104, ingenieros y programadores pueden tomar ventaja de sus conocimientos de hardware y software PC compatible para desarrollar rápidamente sistemas empotrados (for Embedded Systems & OEM Design, 1998b). Los sistemas basados en PC104 son utilizados para diversas aplicaciones, incluyendo fábricas, laboratorios, plantas de proceso, vehículos y casi cualquier otro lugar donde los dispositivos deban ser controlados por una computadora programable. Las especificaciones PC104 mecánicas y eléctricas comunes de los módulos hacen que sean intercambiables con productos de cualquiera de los fabricantes que cumpla con el estándar (Consortium, 2003).

2.1.1 Arquitectura

Para la implantación de la plataforma se ha usado un dispositivo PC104 con 4 módulos ensamblados en forma de escalera, uno sobre otro, conectados a una fuente de poder. Estos módulos están envueltos por una cubierta aislante diseñada para soportar las exigencias de la operación en campo.

- **Módulo del CPU**: corresponde a un modelo MOPSlcd7, de la empresa Kontron (AG, 2004).

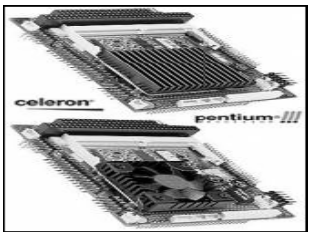

Fig. 1: Módulo del CPU

El MOPSlcd7 es una tarjeta sencilla que ofrece la oportunidad de obtener un alto rendimiento a un bajo costo. El procesador usado es un Intel Mobile Pentium III de 700 MHz.

La versión con el procesador Intel es ideal para aplicaciones que requieren un alto rendimiento en el poder de procesamiento. Este modelo es versátil, incluye un dispositivo VIA Twister-T VT8606 north bridge, un VT82C686B south bridge y un controlador gráfico integrado S3 ProSavage4.

La versión usada del MOPSlcd7 esta compuesto por lo siguiente (AG, 2004):

- • CPU (Mobile Pentium III de 700 MHz con 256 Kb de cache L2 integrada)
- • Sistema ROM (BIOS) (BIOS Phoenix, con 512Kb Flash BIOS)
- • 512Mb de SRAM
- • Controladores para acceso directo a memoria (DMA)
- • Contadores
- • Controladores de Interrupciones
- • Controladores para teclado/ratón
- • Interfaz para altavoz
- • Interfaz para disco duro IDE
- • Puertos Seriales (COM1 y COM2)
- • Puerto Paralelo (LPT1)
- • Puertos USB
- • Ethernet 10/100Base

- **Módulo de almacenamiento**: se encarga de albergar el dispositivo de almacenamiento secundario, en este caso una *Compac Flash*, lo que elimina los problemas asociados al uso de dispositivos mecánicos. El modelo de este módulo es una tarjeta PCM-3116CF de la compañía Canadiense Tri-M Systems and Engineering (for Embedded Systems & OEM Design, 1998a).

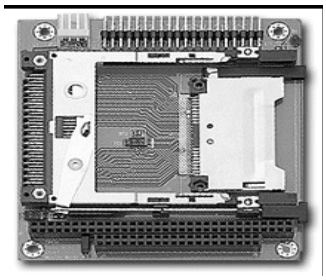

Fig. 2: PCM-3116CF

La familia de drivers PCM-3116 brinda las conveniencias de las PCMCIA y las tarjetas Compac Flash para los sistemas computacionales industriales.

Todos los modelos de PCM-3116 son compatibles con el estándar de PC104. Las conexiones pueden realizarse de 2 maneras, una a través del conector del PC104 y otra a través de IDE.

La PCM-3116CF es un módulo que permite conectar el PC con un controlador IDE a una Compact Flash de lectura/escritura. El módulo convierte los 50 pines de una Compact Flash a los 40 pines de la señal IDE.

- **Módulo de red:** permite conectar el PC104, vía ethernet, con otros dispositivos. Este módulo corresponde a un modelo PCM-3660 de la empresa Canadiense Tri-M Systems and Engineering (for Embedded Systems & OEM Design, p1995).

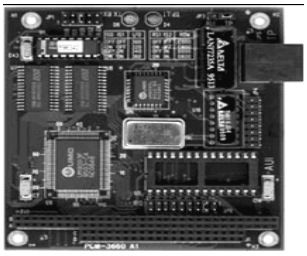

Fig. 3: PCM-3660

El PCM-3660 es un módulo de interfaz ethernet de 16 bit de alto rendimiento. El módulo detecta, automáticamente, si el dispositivo conectado es de 8 bit o de 16 bit. El PCM-3660 cumple con los estándares IEEE 802.3 (10 Mbps) CSMA/CD y es 100% compatible con NE2000.

Módulo de Comunicación Serial: permite la comunicación del PC104 con otros dispositivos, vía puerto serial, para cumplir con tareas de adquisición de datos o aplicación de técnicas de control. Este módulo corresponde a un modelo Xtreme/104 desarrollado por la empresa Canadiense Connect Tech Inc. (Inc, 2004).

El módulo Xtreme/104 ofrece 4 y 8 puertos seriales asíncronos RS-232 o RS-422/485, para conexión de distintos dispositivos para automatización industrial.

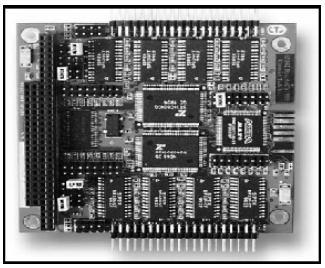

Fig. 4: Módulo Xtreme/104

- **Módulo de suministro de potencia:** proporciona la potencia eléctrica necesaria para el funcionamiento del resto del sistema. Este módulo se corresponde a un modelo

HE104 de la empresa Canadiense Tri-M Systems and Engineering (for Embedded Systems & OEM Design, 1998b).

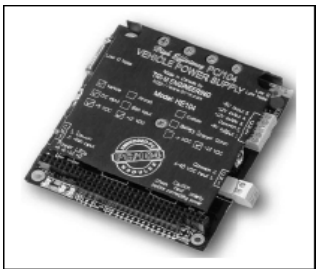

Fig. 5: Módulo HE104

- **Protección externa:** la protección externa está constituida por una armazón que envuelve todos los módulos, de tal forma de poder aislarlos del contacto con el medio y protegerlos de posibles movimientos bruscos. Este armazón corresponde a un modelo VT-6 de la empresa Canadiense Tri-M Systems and Engineering (for Embedded Systems & OEM Design, 2004).

Fig. 6: Modelo VT-6

Los VT104 es un armazón de aluminio que pueden resguardar cualquier PC104 o PC104-Plus. La sólida estructura de la pieza proporciona protección ante la vibración. Esta armazón incluye dos placas (frontal y trasera) con variadas configuraciones dependiendo de los distintos dispositivos a conectar al PC104. El VT-6 puede abarcar hasta 8 módulos. Placa Frontal:

- • Modelo: VT-EC29
- • Puertos USB: 2
- • Puertos DB9: 1
- • Puertos PS/2: 2
- • Puerto Db25: 1

Las figs. 7 y 8 muestran dos ejemplos de configuraciones para las placas frontal y traseras respectivamente.

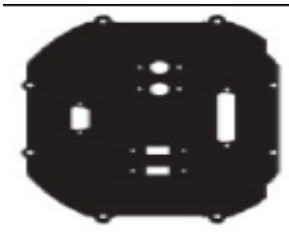

Fig. 7. Placa frontal

- Placa Trasera:
- • Modelo: VT-EC28
- • Puertos DB9: 10
- • Puertos RJ-45: 2
- • Conector de Energía: 1

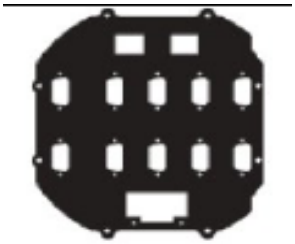

Fig. 8: Placa trasera

2.2 Sistema Operativo tiempo real

Para cumplir con los requisitos de soporte tiempo real y de software libre, se ha tomado para la implantación de la plataforma el RTAI. RTAI es una implementación de Linux para tiempo real basada en RTLinux. Añade un pequeño kernel de tiempo real bajo el kernel estándar de Linux y trata al kernel de Linux como una tarea de menor prioridad. RTAI además proporciona una amplia selección de mecanismos de comunicación entre procesos y otros servicios de tiempo real (Mantegazza, 2006a; Abbot, 2006). Adicionalmente, RTAI proporciona un API (Application Programming Interface) llamada LXRT para facilitar el desarrollo de aplicaciones de tiempo real en el espacio de usuario.

2.2.1 Arquitectura

RTAI tiene una arquitectura similar a RTLinux, al igual que éste, RTAI trata el kernel estándar de Linux como una tarea de tiempo real con la menor prioridad, lo que hace posible que se ejecute cuando no haya ninguna tarea tiempo real por ejecutar. Las operaciones básicas de las tareas de tiempo real son implementadas como módulos del kernel.

Las interrupciones originadas en el procesador, principalmente señales de error como división por cero, son manejadas por el kernel estándar, pero las interrupciones de los periféricos (como los relojes) son manejadas por RTAI Interrupt Dispatcher.

RTAI envía las interrupciones a los manejadores del kernel estándar de Linux cuando no hay tareas de tiempo real activas. Las instrucciones de activar/desactivar las interrupciones del kernel estándar son reemplazadas por macros que se enlazan con las instrucciones de RTAI. Cuando las interrupciones están desactivadas en el kernel estándar, RTAI encola las interrupciones para ser repartidas después de que el kernel estándar haya activado las interrupciones de nuevo. Adicionalmente, se puede ver en la figura 9 el mecanismo de comunicación entre procesos (IPC), que está implementado de forma separada por Linux y por RTAI. También existe un planificador (scheduler) distinto para el

kernel estándar y para RTAI.

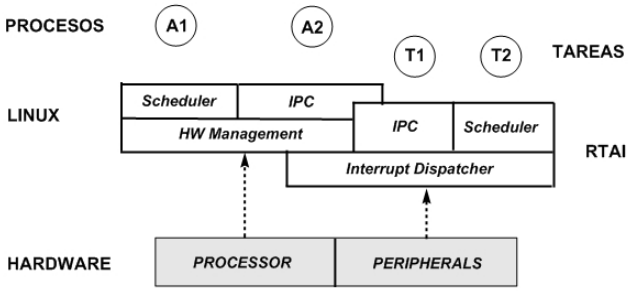

Fig. 9: Arquitectura de RTAI

2.2.2 Planificación (Scheduler)

La unidad de planificación de RTAI es la tarea. Siempre hay al menos una tarea, el kernel estándar.

Cuando las tareas de tiempo real son añadidas, el planificador da entonces mayor prioridad a éstas. El planificador proporciona servicios tales como *suspend*, *resume*, *yield*, *make periodic*, *wait until*, que son usadas en varios sistemas operativos de tiempo real (Mantegazza, 2006b).

El planificador es implementado como un módulo del kernel dedicado (contrario a RTLinux) lo que facilita la implementación de planificadores alternativos, si es necesario. Actualmente hay tres tipos de planificadores, dependiendo del tipo de máquina: Uniprocesador, Multiprocesador simétrico, Multi-Uniprocesador.

2.2.3 Características

Con el objetivo de facilitar y flexibilizar el desarrollo de aplicaciones, en RTAI han introducido diferentes mecanismos para la comunicación entre procesos (IPC), entre las tareas de tiempo real y los procesos en el espacio de usuarios. Como añadido al mecanismo IPC, RTAI proporciona servicios de manejo de memoria e hilos (threads) compatibles con Posix (Mantegazza, 2006b; Navarro, 2002).

RTAI proporciona una variedad de mecanismos para la comunicación entre procesos. Aunque los sistemas Unix proporcionan mecanismos similares a IPC para los procesos en el espacio de usuario, RTAI necesita proporcionar una implementación propia para que las tareas de tiempo real puedan usar este mecanismo y no usen el estándar del kernel de Linux. Los diferentes mecanismos de IPC están incluidos como módulos de kernel, lo que facilita la carga cuando son necesarios. Como ventaja adicional el uso de módulos para los servicios, IPC facilita el mantenimiento y la expansión.

- **Comunicación inter-procesos**: el antiguo mecanismo básico de comunicación dentro de RTAI eran los FIFOs. FIFO es un canal asíncrono, y no bloqueante, de comunicación entre los procesos de Linux y las tareas de tiempo real. La implementación RTAI de FIFO esta basada en la implementación de RTLinux, pero RTAI proporciona algunas características que no son posibles en RTLinux. Primeramente, RTAI puede lanzar señales cuando hay eventos en el FIFO (escritura de nuevos datos). Los procesos en el espacio de usuario pueden entonces crear un manejador para la señal usando los mecanismos estándar de Unix.

  Sin embargo, este mecanismo no es necesario para los procesos de usuario que quieran leer ó escribir del FIFO. Adicionalmente, puede haber múltiples lectores y escritores en el FIFO, cosa que no es posible en la version de RTLinux (Bucher, 2006; Abbot, 2006).

  Por otro lado, los identificadores FIFO pueden ser dinámicamente localizados con un nombre simbólico. Antes era necesario establecer un identificador global para el FIFO, lo que causaba problemas cuando múltiples, y posiblemente independientes, procesos lo usaban. Los semáforos es otra herramienta básica de sincronización entre procesos usada en los sistemas operativos. RTAI proporciona un API para usar semáforos, aunque cada semáforo esta técnicamente asociado a un FIFO, por tanto cada semáforo usa una entrada global del FIFO. Como añadido al servicio básico de semáforos, es posible asociar un semáforo con un reloj, el cual puede ser usado para despertar un proceso encolado en un semáforo, incluso cuando el semáforo aún está cerrado. La memoria compartida proporciona una alternativa a IPC y al paradigma FIFO cuando un modelo de comunicación diferente es requerido. La memoria compartida es un bloque común de memoria que puede ser leído ó escrito por más de un proceso en el sistema. Como los diferentes procesos pueden operar de forma asíncrona en la región de memoria, es necesario un diseño para asegurar la consistencia de los datos.

  Finalmente, quizás el método más flexible de IPC sean los mailboxes. Cualquier número de procesos pueden enviar y recibir mensaje a y desde un mailbox. Un mailbox almacena mensajes hasta un límite definido, y contrario a los FIFOs, el mailbox preserva los mensajes que están en el límite. Puede haber un número arbitrario de mailbox activos en el sistema simultáneamente. RTAI también facilita la comunicación entre procesos mediante RPC.
- **Gestión de memoria:** en las primeras versiones de RTAI la memoria tenía que ser asignada estáticamente. Sin embargo, en las actuales versiones se incluye un módulo gestor de memoria que permite la asignación dinámica de memoria por parte de las tareas de tiempo real usando una interfaz basada en una biblioteca estándar de C.
- **Hilos Posix:** RTAI tiene módulos que proporcionan la implementación de hilos basada en el estándar POSIX 1003.1c. Usando las operaciones especificadas en dicho estándar, el usuario puede manejar de manera similar a como lo hace con los hilos posix convencionales, excepto en cuanto a los conceptos de *joining* y *detaching*, (Bishop, 2004; Navarro, 2002).
- **LXRT:** es un API para RTAI que hace posible el desarrollo de aplicaciones de tiempo real en el espacio de usuario sin tener que crear módulos para el kernel. Esto es útil en

primer lugar porque el espacio de memoria destinado al kernel no esta protegido de accesos inválidos, lo que puede provocar la corrupción de datos y el mal funcionamiento del kernel de Linux. En segundo lugar, si el kernel es actualizado, los módulos necesitan ser recompilados lo que puede provocar que sean incompatibles con la nueva versión.

3 Diseño e implantación de la plataforma de tiempo real

Le meta final en el diseño de sistemas de control y supervisión de procesos es su implementación práctica. Cualquier algoritmo de control y supervisión que se adopte demanda su ejecución siguiendo rigurosos tiempos para las tareas programadas.

De allí surge la necesidad de aplicaciones que puedan ejecutarse en una plataforma tiempo real (Bishop, 2004). Es así que aplicaciones de control de procesos deben desempeñarse bajo sistemas operativos en tiempo real, lo cual garantiza el determinismo de las funcionalidades de esas aplicaciones, con relación al resto de las tareas programadas en el sistema. Nuestro interés es la elaboración de un sistema para la prestación de servicios en tiempo real. Además, se requiere que esta plataforma sea fácil de implantar en términos económicos y prácticos; es decir que no requiera licencias de software y que sus componentes puedan ser adquiridos sin mayores inconvenientes.

El sistema se construye utilizando una distribución Linux ya instalada (Fedora Core 3). Este sistema Linux existente (el anfitrión) se utiliza como punto de inicio para suministrar los programas necesarios, como un compilador, un enlazador y un intérprete de comandos. El proceso de construcción del nuevo sistema esta compuesto por 6 secciones.

3.1 Pre-instalación

31.1 Preparara una partición

El primer paso es preparar la partición que contendrá el sistema. Se crea la propia partición, se crea un sistema de archivos en ella y se monta. Para crear una nueva partición se utiliza el comando fdisk pasándole como argumento el nombre del dispositivo en el que debe crearse la nueva partición, en este caso */dev/sda1* (*compact flash*) y luego se crea la nueva partición de tipo Linux.

Para crear un sistema de archivos del tipo extendido 2 (ext2) en la partición se ejecuta: mke2fs –v /dev/sda1. Una vez creado el sistema de archivos es necesario hacer accesible la partición; Se selecciona el punto de montaje (ej. /mnt/XX). Finalmente, se monta el sistema de archivos ejecutando: mount -v /dev/sda1 /mnt/XX.

3.1.2 Instalar paquetes requeridos

En esta sección se explica qué paquetes y arctualizaciones deben descargarse para construir el nuevo sistema y cómo guardarlos en el nuevo sistema de archivos. Es necesario guardar todos los paquetes, mejoras y actualizaciones descargadas en algún sitio que esté disponible durante toda la construcción. También se necesita un directorio de trabajo en el que desempaquetar las fuentes y construirlas. Se usa */mnt/XX/sources* tanto para almacenar los paquetes y sus actualizaciones como directorio de trabajo.

Paquetes requeridos: Binutils (2.15.94.0.2.2), Coreutils (5.2.1), Glibc (2.3.4), Sysvinit (2.86), Udev (056), Grub (0.96), Bash (3.0) y GCC (3.4.3).

Aparte de los paquetes, también se necesitan varias actualizaciones. Estos actualizaciones corrigen pequeños errores en los paquetes. Los actualizaciones también hacen pequeñas modificaciones para facilitar el trabajo con el paquete.

Las siguientes actualizaciones son necesarias para construir el sistema: Bash Avoid Wcontinued Patch, BashVarious Fixes, Binutils Build From Host Running Gcc4 Patch, GCC Linkonce Patch, GCC No-Fixincludes Patch, Glibc Rtld Search Dirs Patch.

3.2 Configuaración del entorno de trabajo

Todas las herramientas requeridas se instalan bajo /mnt/XX/tools para mantenerlos separados de los programas propios del sistema anfitrión. Se crea el directorio necesario ejecutando lo siguiente como root: mkdir -v /mnt/XX/tools.

3.3 Instalación de herramientas

Esta sección describe la instalación de una serie de paquetes que forman el entorno básico de desarrollo (o herramientas principales) utilizado para construir el sistema real. La construcción de este sistema se hace en dos etapas.

La primera es construir un conjunto de herramientas independientes del sistema anfitrión (compilador, ensamblador, enlazador, librerías y unas pocas herramientas útiles).

La segunda etapa utiliza estas herramientas para construir el resto de herramientas esenciales.

Los archivos compilados en esta sección se instalan bajo el directorio */mnt/XX/tools* para mantenerlos separados de los archivos que se instalan en la siguiente sección y de los directorios de producción del anfitrión. Puesto que los paquetes compilados aquí son puramente temporales, no se desea que estos archivos afecten el futuro sistema.

3.4 Construcción del sistema

En esta sección se entra en la zona de construcción y se comienza a construir realmente el nuevo sistema. Es decir, se cambia la raíz al mini sistema Linux temporal construido, se hacen unos cuantos preparativos finales, y entonces se comienza a instalar los paquetes. Es hora de entrar en

el entorno *chroot* para iniciar la construcción e instalar el sistema final. Como usuario root, se ejecutan los siguientes comandos para realizar el cambio de entrono:

```
chroot "/mnt/XX" /tools/bin/env -i \
HOME=/root TERM="$TERM" PS1='\u:\w\$ ' \
PATH=/bin:/usr/bin:/sbin:/usr/sbin:/tools/bin \
/tools/bin/bash --login +h.
```

Es hora de crear cierta estructura en el sistema de archivos. Se crea un árbol de directorios estándar ejecutando los siguientes comandos:

```
install -dv /{bin,boot,dev,etc/opt,home,lib,mnt}
install -dv /{sbin,srv,usr/local,var,opt}
install -dv /root -m 0750
install -dv /tmp /var/tmp -m 1777
install -dv /media/{floppy,cdrom}
install -dv /usr/{bin,include,lib,sbin,share,src}
ln -sv share/{man,doc,info} /usr
install -dv /usr/share/{doc,info,locale,man}
install -dv /usr/share/{misc,terminfo,zoneinfo}
install -dv /usr/share/man/man{1,2,3,4,5,6,7,8}
install -dv/usr/local/{bin,etc,include,lib,sbin,share,src}
ln -sv share/{man,doc,info} /usr/local
install -dv /usr/local/share/{doc,info,locale,man}
install -dv /usr/local/share/{misc,terminfo,zoneinfo}
install -dv /usr/local/share/man/man{1,2,3,4,5,6,7,8}
install -dv /var/{lock,log,mail,run,spool}
install -dv /var/{opt,cache,lib/{misc,locate},local}
install -dv /opt/{bin,doc,include,info}
install -dv /opt/{lib,man/man{1,2,3,4,5,6,7,8}}
```

El árbol de directorios está basado en el estándar FHS. Para que root pueda entrar al sistema y para que el nombre "root" sea reconocido, es necesario tener las entradas apropiadas en los archivos /etc/passwd y /etc/group. Finalmente, se instalan los paquetes que componen el sistema.

3.5 Configuración del núcleo y del gestor de arranque

En esta sección se crea el archivo fstab, se configura el núcleo para el nuevo sistema y se instala el gestor de arranque (GRUB). El archivo /etc/fstab lo utilizan ciertos programas para determinar el punto de montaje de los sistemas de archivos, el orden y la lista de comprobación (por fallos de integridad) antes de montarse. Se crea una nueva tabla de sistemas de archivos. Construir el núcleo comprende varios pasos: configuración, compilación e instalación; se prepara la compilación ejecutando el siguiente comando: make mrproper. Antes de iniciar la configuración es necesario aplicar una mejora al núcleo, contenida en el paquete de RTAI, con la intención de agregarle soporte para acciones de tiempo real:

patch -p1 </usr/src/rtai.3.3/base/arch/i386/patches/hal-linux-2.6.11-i386-r12.patch.

Se configura el núcleo mediante una interfaz de menús: make menuconfig. Algunas de las características del núcleo que se toman en cuenta, para el correcto funcionamiento de RTAI, son las siguientes:

Modules:

- Loadable module support: se activa "Enable module support", "Module unloading", y "Automatic module loading 2 se desactiva"Module versioningsupport".
- Processor type and features: se selecciona el tipo de subarquitectura como (PCCompatible) y como familia de procesador un Pentium III, se activa "Preemption Model", se desactiva "Use register arguments 2 "Local APIC support on uniprocessors".
- Bus options: se mantienen los valores por omisión.

Device Drivers:

- Generic driver options: se mantienen los valores por omisi´on.
- Memory Technology Devices (MTD): no se necesita.
- Parallel port support: se desactiva. El puerto paralelo estándar no es útil en experimentos de tiempo real; a través de los drivers de Comedi se puede acceder a este puerto.
- Plug and Play support: se mantienen los valores por omisión.
- Block devices: aquí se seleccionan los distintos dispositivos.
- Network device support: se mantienen los valores por omisión.
- Input device support: se asegura que Mouse esta seleccionado.
- I2C support: se desactiva; existen reportes de dificultades cuando se usan con RTAI.
- Multimedia devices: se desactiva.
- Graphics support: se mantiene desactivado.
- Sound: se desactiva
- USB Support: se desactiva.

File systems:

- Second extended fs support: seleccionado
- Ext3 journalling file system support: seleccionado

y "Ext3 extended attributes"

- Reiserfs support: no se necesita.
- CD/ROM-DVD Filesystems: no se selecciona.
- DOS/FAT/NT Filesystems: no se necesita.

Se compila la imagen del núcleo y los módulos: make. Se instala los módulos: make modulesinstall.

Tras completar la compilación se necesitan algunos pasos adicionales para completar la instalación. Es necesario copiar varios archivos al directorio /boot:

- cp -v arch/i386/boot/bzImage /boot/lfskernel-2.6.11.12.
- cp -v System.map /boot/System.map-2.6.11.12.
- cp -v .config /boot/config-2.6.11.12.

3.6 Instalación de RTAI

Para instalar RTAI correctamente es necesario configurar el kernel de manera adecuada además de agregar las mejoras proporcionadas por los desarrolladores de RTAI correspondiente a la versión del kernel utilizada (2.6.11.12). Después de la correcta configuración debemos compilar el kernel y comenzar con la instalación de RTAI. Antes de realizar la instalación de RTAI, es necesario crear un enlace simbólico a las fuentes del núcleo, con el siguiente comando: [11].

ln -s linux-2.6.11.12-adeos linux

La documentación de RTAI recomienda construirlo fuera del árbol de las fuentes:

mkdir rtai-3.3-builddir && cd rtai-3.3-builddir make f /rtai-3.3/makefile
srctree=../rtai-3.3 oldconfig
Se realiza la configuración:
make -f ../rtai-3.3/makefile srctree=./rtai-3.3 men config
Se compila el paquete:
Make
Se instala el paquete:
Make install

Al instalar el paquete RTAI se crea el directorio /usr/realtime/ es importante agregar /usr/realtime/bin al PATH de variables en /etc/profile.

Para verificar el correcto funcionamiento de RTAI se procede a probar algunas rutinas de test que se encuentran en el directorio /usr/realtime/testsuite/kern las cuales son instaladas junto con RTAI.

3.6.1 Instalación de comedi

Comedi provee los drivers, librerías de funciones y un API para interactuar con señales provenientes del hardware de adquisición de datos.

Comedilib: Este paquete contiene las librerías necesarias para el correcto funcionamiento de los driver de comedi. Primero se configura el paquete:

./configure –sysconfdir=/etc
Se compila el paquete:
make
Se instala el paquete:
make install

Se crea los dispositivos inodos en el directorio /dev/comedi[0-3].

make dev
Comedi:
Se configura el paquete:
./configure –withlinuxdir=/ usr/src/linux -withrtaidir=/ usr/realtime
Se compila el paquete:
Make
Se instala el paquete:
Make install
Se crean los dispositivos:
Make dev

Algunos archivos hay que reubicarlos y hay que crear algunos enlaces para asegurar el correcto funcionamiento de los drivers.

cp include/linux/comedi.h /usr/include/
cp include/linux/comedilib.h /usr/include/
ln-s/usr/include/comedi.h /usr/include/linux/comedi.h
ln-s/usr/include/comedilib.h /usr/include/linux/comedilib.h
ln -s /usr/include/linux /usr/local/include/linux

El nuevo sistema está casi completo. Una de las últimas cosas por hacer es asegurarse de que puede ser arrancado.

Ahora se inicia el intérprete de comandos de grub (gestor de arranque):

grub
Se le indica a GRUB la ubicación de sus archivos stage1, 2.
root (hd0,1)
Se le indica a GRUB que se instale en el MBR de sda:
Setup (hd0, 1)

GRUB informa que ha encontrado sus archivos en /boot/grub. Esto es todo para activarlo. Se cierra el intérprete de comandos de grub:
Quit.

Finalmente se crea un archivo de "lista de menú" para definir el menú de arranque de GRUB.

Al terminar de construir el sistema se instalaron otras herramientas que prestaban ciertas utilidades al desarrollo de distintas aplicaciones sobre el PC104. Entre alguna de estas tenemos:

- Un Servidor Web (lighttpd-1.4.11)
- Un servidor de Base de Datos (sqlite-3.3.8)
- Un editor de Texto (nano-1.2.5)
- Un Cliente DHCP

Con esta plataforma base para aplicaciones a tiempo real, se puede disponer de todo un sistema de control supervisorio ya que permite incorporar:

- Mecanismos de comunicación con dispositivos de campo a través de protocolos estándares (ModBus, DNP3, etc.). Ver (González O. et al., 2009).
- Bases de datos tiempo real a partir de manejadores genéricos. Ver (Hidrobo F. *et al*., 2009).
- Ambientes de configuración y supervisión con capacidades gráficas, a través de servicios WEB. Ver (Hidrobo F. *et al*., 2009).
- Lógicas de control con algoritmos muy específicos y/o a partir de sistemas tipo SoftPLC que utilizan lenguajes de programación estándares en la industrial de PLCs. (González O. *et al*., 2009).

Así, este sistema tiene la capacidad para ser utilizado como un medio de adquisición de datos, un sistema de control directo, y un sistema supervisorio, todo en un mismo dispositivo, lo cual es difícil de obtener con tecnologías de microcontroladores tipo PIC.

4 Aplicación de prueba

Con el objetivo de probar el funcionamiento de la plataforma, se ha implementado un simulador de un proceso real. En esta aplicación se trabaja con un sistema de control en tiempo real que consiste en conectar, por medio del puerto serial, la plataforma a un PC donde se simula el comportamiento de una planta. En la Fig. 10 se muestra el sistema de control implementado.

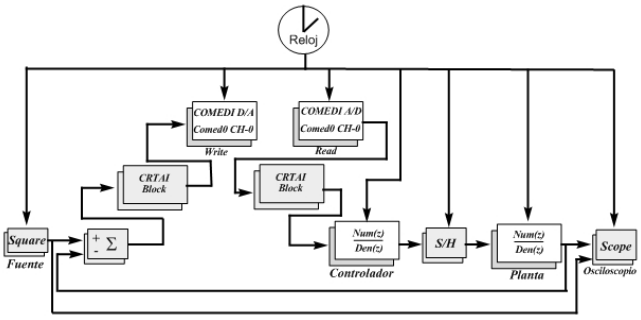

Fig. 10: Sistema de control

Este ejemplo esta basado en el tutorial de control de Matlab y Simulink: Modelado de la posición del Motor DC en Simulink (Mejía, 2006), el cual es controlado desde el PC104, es decir, los coeficientes del controlador son asignados por una rutina en el PC104 a través del puerto serial. El bloque de la planta (motor) representa una función de transferencia, el bloque controlador describe el algoritmo o mecanismo de control aplicado, el bloque para la escritura (*Write*) representa enviar datos e información por el puerto serial, el bloque de lectura (*Read*) describe la acción de tomar datos por el puerto serial. La rutina que se emplea en el PC104 para la asignación de los coeficientes del controlador, corresponde al siguiente diagrama:

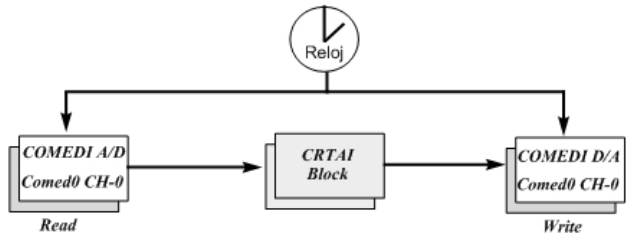

Fig. 11: Diagrama de bloques para El PC104

En realidad no se ha utilizado ningún tipo de algoritmo para el diseño del controlador, solo se han asignado los valores correspondientes al diseño realizado en el manual de Matlab para el control del motor DC. La principal utilidad de esta aplicación es demostrar la comunicación entre el PC104 y un PC en tiempo real, abriendo la posibilidad de aplicar distintas técnicas de control sobre cualquier modelo simulado o sistema físico.

La respuesta del sistema ante una entrada escalón, se puede observar en el osciloscopio virtual que suministra la herramienta rtailab (ver la Fig. 12).

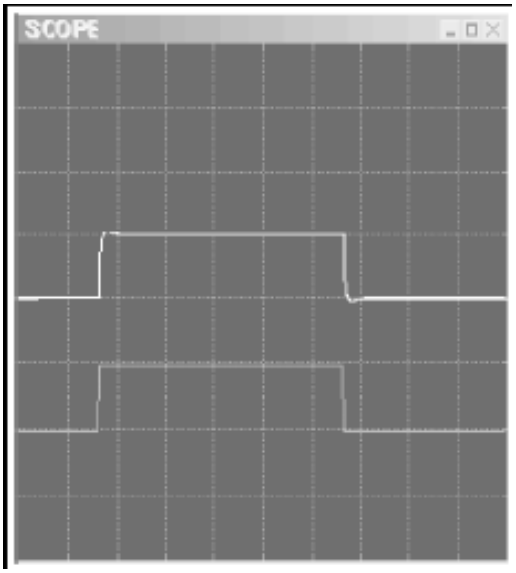

Fig. 12: Salida del sistema controlado

5 Conclusiones

En este trabajo se ha implantado una plataforma basada en el estándar PC104 que usa como sistema operativo una versión de Linux con soporte para tiempo real. Esta plataforma utiliza herramientas libres y estándares abiertos.

La plataforma implantada facilita su integración con otros ambientes de control y supervisión.

Además, permite la escalabilidad, conectividad, mantenibilidad y permite agregar inteligencia en sitio para ejecutar las funciones de control, supervisión y optimización en tiempo real, ya que se pueden incorporar elementos para la comunicación con dispositivos a través de protocolos estándares, el manejo de variables operacionales mediante una base de datos a tiempo real, servicios WEB para la configuración y supervisión, y la implementación de algoritmos de control específicos o desarrollados mediante lenguajes de programación de PLCs. Así, la plataforma que se presenta es una propuesta de bajo costo, que pueda implantarse de forma sencilla, no requiere licencias de software, no depende de un fabricante específico.

Por otro lado, se puede utilizar en ambientes industriales de altas exigencias. RTAI constituye una herramienta de gran utilidad en la implantación de sistemas empotrados de bajo costo y alta fiabilidad. Esta herramienta provee una serie de recursos que facilitan la programación y la hacen menos vulnerable a fallas.

Adicionalmente, los sistemas desarrollados con esta herramienta pueden contar con todas las bondades de un sistema operativo de propósito general como Linux, lo que permite asignar tareas críticas y no críticas en el mismo sistema dando preferencia a las primeras.

Para comprobar la efectividad de esta plataforma se desarrolló un ejemplo académico que permitió comprobar el funcionamiento de la misma.

En este caso, se ha podido implantar un esquema de control básico usando un modelo simulado que fue controlado desde el sistema implantado.

6 Agradecimientos

Este trabajo ha sido financiado por el FONACIT bajo el proyecto No. 2005000170, y por el CDCHT-ULA a través del proyecto No. I-1103-08-02-A, instituciones a quienes los autores expresan sus agradecimientos.

Referencias

Abbot D, 2006, Linux for Embedded and Real Time Applications, Elsevier.

AG KEC, 2004, Manual de Usuario del MOPSlcd7, http://www.kontron.com.

Bishop S, 2004, Introduction to Linux for Real Time Control, National Institute of Standards and Technology.

Bucher R, 2006, RTAI-Lab, Tutorial: Scilab, Comedi, and real-time control, http://www.rtai.org.

Consortium PE, 2003, PC104 Especificaciones, Versión 2.5.

for Embedded Systems & OEM Design, T. S. H. S. 1995, Manual de Usuario de la PCM-3660, http://www.tri-m.com.

for Embedded Systems & OEM Design, T. S. H. S. (1998a). Manual de Usuario de la PCM- 3116, http://www.tri-m.com.

for Embedded Systems & OEM Design, T. S. H. S. 1998b Vehicle Power Supply: Manual de Usuario, http://www.tri-m.com.

for Embedded Systems & OEM Design, T. S. H. S. 2004, Manual de Usuario del VT104, http://www.tri-m.com.

González O, Rivas Y, Hidrobo F y Ríos-Bolívar A. 2009, Desarrollo de un Sistema de Control y Supervisión de Datos Embebido en Tiempo Real: Diseño de la Versión Net-DASTM v2.0. Reporte Técnico.PDVSA-Intevep-ULA.

Gusenbauer M, Irschik H and Schlacher K, 2005, Implementation of control tasks via Linux-RTAI, Technical report, Strucutural Control Network SCN, http://www.structuralcontrol. at /RTAI webpage/ContSys.html.

Hidrobo F, Ríos-Bolívar A y Díaz G, 2009, Un Sistema de Control y Supervisión de Datos en Tiempo Real bajo Software Libre. Proc. del Congreso Colombiano de Informática. Bucaramanga, Colombia.

Inc, CT, 2004, Manual de Usuario de la Xtreme/ 104, http://www.connecttech.com.

Mantegazza P, 2006a RTAI, 3.3 Manual del Usuario, http://www.rtai.org.

Mantegazza P, 2006b, RTAI, Guía de Principiante, http://www.rtai.org.

Navarro VJ B 2002, Sistemas Inteligentes en Tiempo Real, Universidad Politécnica de Valencia.