Sistema de control automático a distancia para un actuador lineal de posicionamiento de muestras para irradiación neutrónica empleando plataformas de hardware abierto de bajo costo

Eduardo Cunya[1,*], Renzo Chan[1], Adison Pacheco[2]

[1] Instituto Peruano de Energía Nuclear, Dirección de Investigación y Desarrollo, Av. Canadá 1470, Lima 41, Perú

[2] Universidad Nacional de Ingeniería, Facultad de Ing. Eléctrica y Electrónica, Av. Túpac Amaru, km 1, Lima 25, Perú

Resumen

En el presente artículo se describe los criterios de diseño e implementación de un sistema de control automático, basado en circuitos electrónicos embebidos (microcontroladores) de diferentes arquitecturas (RISC, ARM) y el marco de trabajo de software de fuente abierta *CanFestival,* para el desarrollo de aplicaciones de usuario. Los dispositivos electrónicos son autónomos y soportan la funcionalidad requerida para la comunicación y el control a distancia del actuador lineal.

Palabras clave: CanFestival, Automatización, Control, Activación neutrónica, Accionadores, Diseño

Automatic control system of a linear actuator for positioning samples to neutron irradiation using low cost open-hardware boards

Abstract

This paper describes the design criteria and implementation of an automatic control system based on embedded electronic circuits (microcontrollers) of different architectures (RISC, ARM) and open source framework software *CanFestival*, to develop user applications. Electronic devices are autonomous and support the functionality required for communication and remote control of the linear actuator.

Keywords: CanFestival, Automation, Control, Neutron activation, Actuator, Design

1. Introducción

Un sistema de irradiación neutrónica, por normas de seguridad y protección radiológica es recomendable operarlo a una distancia segura; por eso, si el accionamiento es automático y a distancia será de mucha ayuda para el usuario. En el año 2014 se diseñó y construyó en el Instituto Peruano de Energía Nuclear (IPEN) un prototipo de intercambiador para la automatización de la técnica de análisis por activación neutrónica [1].

Actualmente, hay dispositivos programables que pueden realizar tareas simples de control, tales como la activación de un interruptor, el monitoreo de una señal eléctrica o el encendido de una alarma, etc. que son ejecutadas sin la intervención del usuario. Estas facilidades hacen muy atractivo el uso de dichos dispositivos y seleccionarlos adecuadamente para una aplicación específica, sin ser excesivamente costosos y con un mínimo de componentes periféricos.

El objetivo de este desarrollo es la implementación de un sistema de control automático para el posicionamiento de muestras que realiza el personal del Laboratorio de Análisis por Activación Neutrónica del Centro Nuclear RACSO, con la finalidad de completar la automatización de ésta técnica y posibilitar la irradiación de muestras sin la presencia del analista, como etapa previa a su envío a la posición de irradiación en el núcleo del reactor de investigación RP-10 [2].

2. Metodología

El sistema neumático (Figura 1) contemplado en [1], está compuesto por dispositivos mecánicos denominados “válvulas intercambiadoras” que se encargan de modificar la vía

[*]Correspondencia autor: ecunya@ipen.gob.pe

ISSN 1684-1662 (impreso)
ISSN 2225-2029 (en línea)

de recorrido de las muestras encapsuladas, permitiendo así tener dos modos de operación: un "Sistema de envío y retorno directo" y un "Sistema de envío con retorno temporal al hall del reactor para su decaimiento". Las válvulas se representan en el sistema neumático (Figura 1) como "EVI", y se puede observar que están presentes en diferentes etapas del proceso.

Para el control y automatización de los actuadores lineales se optó por el protocolo de comunicación CANOpen, utilizando el marco de trabajo CanFestival que se enfoca en proporcionar una biblioteca en ANSI-C, independiente de la plataforma de hardware que puede ser construida e incorporada como nodo maestro/esclavo en PCs, para la comunicación interprocesos en tiempo real y en microcontroladores. Una característica importante de CanFestival es que tiene integrado un conjunto de controladores de dispositivos (drivers) y una familia de protocolos de comunicación CAN para el sistema operativo Linux denominado S*ocketCan*, que permite a los programadores elaborar aplicaciones de red fácilmente [3].

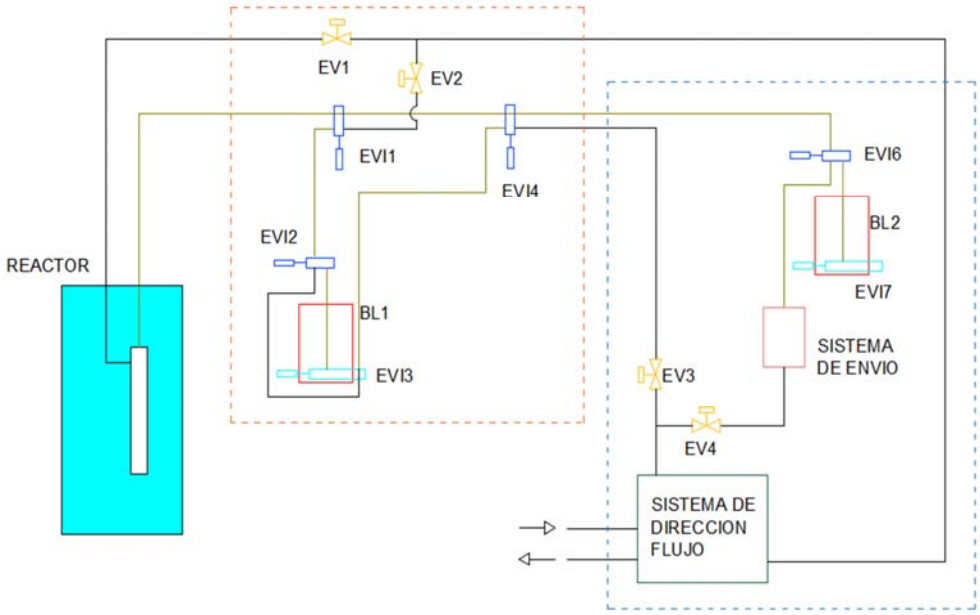

Figura 1. Representación del sistema neumático de envío–retorno de muestras para el AAN.

La tarjeta electrónica (Figura 2) utilizada para este desarrollo, como nodo maestro, es una modificación y adaptación de la tarjeta PICAN [4] para ser instalada sobre una tarjeta Raspberry PI 2 (RPI).

Figura 2. Tarjeta PICAN adaptada para ser usada en el sistema intercambiador de muestras y montada sobre un Raspberry PI 2.

A continuación, se añade al archivo /boot/config.txt del sistema de archivos de la placa RPI, las líneas de configuración:

dtoverlay=mcp2515-can0-overlay,oscillator=1600000, interrupt=22
dtoverlay=spi-bcm2835-overlay

Luego de un reinicio (*reset*) del Raspberry la interface CAN estará lista para ser usada. Después de instalar CanFestival sobre RPI, levantamos la interface con el comando Linux, "*link*", mediante:

sudo /sbin/ip/ link set can0 up type can bitrate 500000
CanOpenShell
load#libcanfestival_can_socket.so,can0 500k,1,1

Donde los parámetros para la carga son el nombre de la biblioteca, tasa de bits, ID de nodo y maestro(1)/esclavo(0). Si la orden

CanOpenShell se inicia correctamente se debe ver el mensaje de arranque:

Node_initialisation
Node_preOperational

Entonces, se puede enviar comandos NMT con los comandos *ssta/ssto/srst/scan* y para leer o escribir mensajes SDOs se hace con los comandos *rsdo* y *wsdo* [5]. Se usa CanFestival a través de una simple aplicación, "esclavosimple", primero se edita el archivo Makefile y se retira el parámetro "*-lcanlib"*, también se debe editar el archivo cabecera main.h y cambiar las siguientes líneas en la sección "Por favor seleccione las siguientes definiciones de acuerdo a sus necesidades"

#define NODE_MASTER 0x1
#define NODE_SLAVE 0x40
#defineDRIVER_LIBRARY
"libcanfestival_can_socket.so"
#define BAUDRATE 500
#define BUS 0

Entonces, se construye la aplicación con:

make mrproper
make

Y para ejecutarla se procede con:

./esclavosimple

Se conecta otro nodo CAN como esclavo, usando una tarjeta Arduino modelo UNO con un CAN BUS Shield (Figura 3) diseñado de acuerdo con el propósito del desarrollo (basado en el diseño de Seed Studio [6]), que le permite comunicarse mediante un protocolo CAN incorporando la biblioteca CANopen adaptada para esta tarjeta [7,8], de este modo se pudo *iniciar/detener* el nodo con los comandos NMT y envíe un mensaje PDO por cada mensaje SYNC que recibe. La aplicación es generada por un editor del diccionario de objetos en el que se puede configurar las partes principales del nodo.

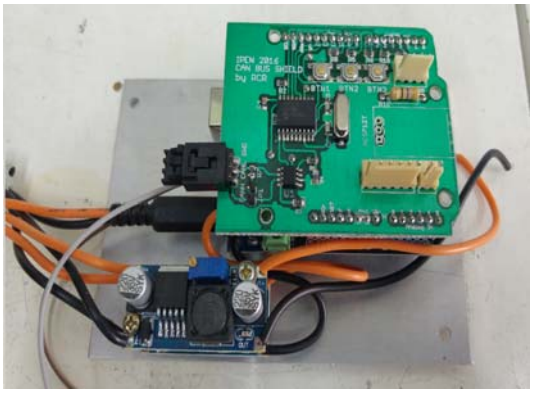

Figura 3. Nodo esclavo Arduino UNO con el shield de bus CAN.

Para poner en funcionamiento la aplicación *esclavosimple* se necesita la herramienta *objdictedit*, una GUI creada en lenguaje de programación Python. Se ejecuta sobre cualquier SO con el módulo PythonWx instalado, para ello se activa en el prompt de comandos del SO:

sudo apt-get install python-wxtools

Luego, se debe situar en el directorio *objdictgen* que contiene el código fuente y se hace:

python objdictedit.py

Enseguida, se va al File a Open y se selecciona *sclavosimple.od* se edita su diccionario de objetos y cualquier modificación debe salvarse otra vez en el archivo *od* para luego ser construido siguiendo los pasos anteriores con los nuevos cambios.

En este paso, se puede monitorear los mensajes en el bus CAN0 con el comando *candump can0* [9] y los diferentes mensajes *SDOs* transferidos entre los dos nodos indicando su estado actual (Figura 4).

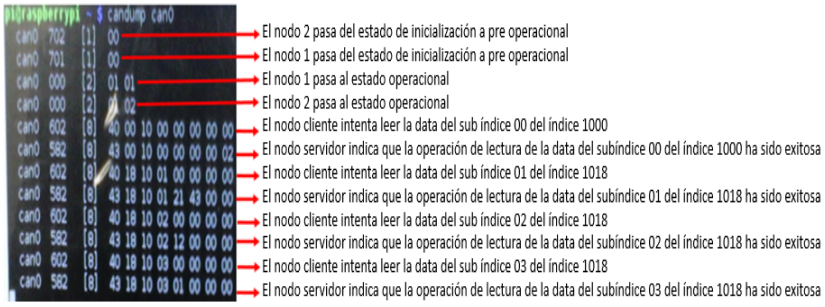

Figura 4. Mensajes CANOpen obtenidos mediante el comando candump.

ISSN 1684-1662 (impreso)
ISSN 2225-2029 (en línea)

Se pueden observar los mensajes NMT enviados por el programa de aplicación, los enviados por el nodo maestro con ID 701 en modo cliente y aquellos del nodo esclavo configurado en modo servidor con ID 702.
esclavosimple [10] es una aplicación de prueba básica donde las definiciones para el ID de nodo y la tasa de bits deberían ser leídas desde el diccionario de objetos directamente. Si es necesario agregar más funcionalidad avanzada, CanFestival tiene handles (controles) de devolución de llamada que se pueden utilizar (Figura 5).

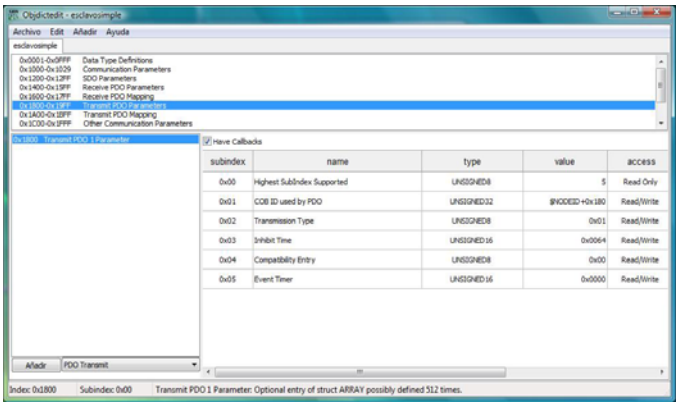

Figura 5. Editor de diccionario de objetos esclavo simple.od.

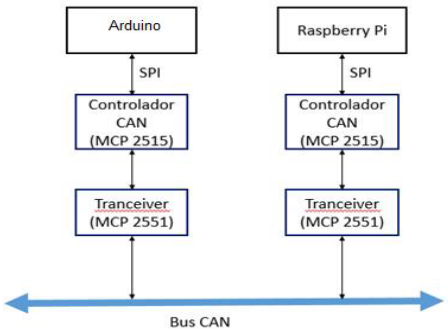

Figura 6. Diagrama de bloque de los nodos maestro y esclavo a través de Bus CAN.

Se procedió a armar la red CANopen que consta de dos nodos, tal como se muestra en el diagrama de la Figura 6. La red mínima en actual desarrollo transfiere mensajes de proceso CAN [11], consistentes en ordenes de maestro a esclavo (Figura 7) para el avance o repliegue de las válvulas intercambiadoras, que sirven para el cambio de recorrido de las muestras a ser enviadas a través del sistema neumático de irradiación del reactor nuclear RP-10 (Figura 8).

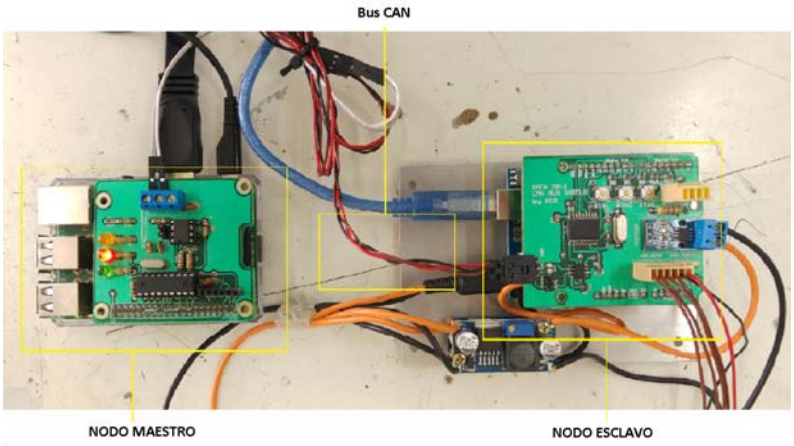

Figura 7. Implementación circuital de los nodos maestro y esclavo.

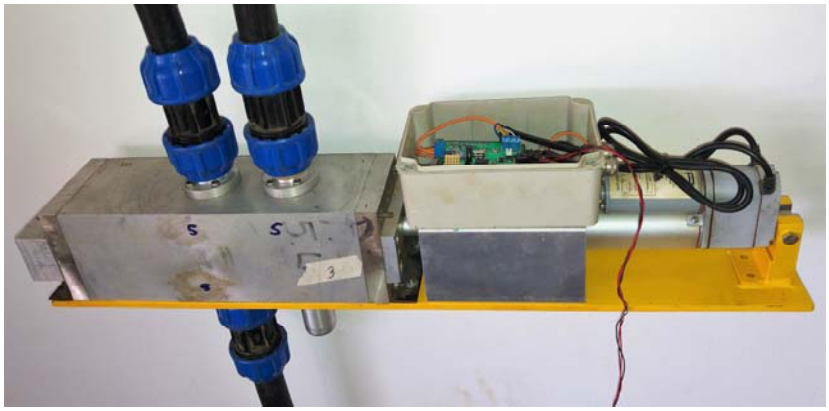

Figura 8. Actuador lineal encargado cambiar el recorrido de las muestras.

3. Resultados y Discusión

La implementación ha permitido interconectar dos plataformas hardware independientes (Raspberry y Arduino) utilizando un solo marco de trabajo de programación (CanFestival) mostrando total compatibilidad funcional para el control de los nodos, no siendo necesario el uso de PCs. Los recursos electrónicos y herramientas de programación son de fácil acceso público. Sin embargo, la desventaja está en la revisión del extenso código que involucra tanto la aplicación de usuario y la biblioteca de funciones estándar. La utilidad automatiza el manejo de la muestra antes y después del proceso de irradiación posicionándola correctamente en el puerto de envío (cabezal) y el recinto de recuperación (campana radioquímica), facilitando la programación de irradiaciones no asistidas por el usuario.

4. Conclusiones

Se realizó el monitoreo de sensores y el control de actuadores distribuidos en un proceso de forma local o remota. Los nodos pueden realizar funciones más complejas dependiendo de los recursos de hardware que integren, tales como mayor tamaño de memoria, canales de conversión sean A/D y D/A, canales de temporización, canales de modulación de ancho de pulso, etc. La tecnología actual permite obtener nodos maestro/esclavo que pueden transferir información a mayor velocidad, lo que permite, por ejemplo, implementar el control en tiempo real para ciertos procesos.

5. Agradecimientos

A la MsC. Patricia Bedregal responsable de la Subdirección de Técnicas Analíticas Nucleares y al Ing. Oscar Baltuano Subdirector de Desarrollo Tecnológico por brindarnos todos los recursos técnicos y logísticos para la implementación del sistema. Asimismo, al Ing. Javier Gago y Téc. Yuri Hernández por su ayuda en el diseño y construcción de los dispositivos mecánicos.

6. Bibliografía

[1]. Gago J, López Y, Hernández Y, Baltuano O, Urquizo R, Bedregal P. Diseño y construcción de un prototipo de intercambiador para la automatización de la técnica de análisis por activación neutrónica. Informe Científico Tecnológico. 2014; 14:153-159.

[2]. Cunya E, Baltuano O, Bedregal P. Implementación de una red de comunicación y control para instrumentos de un laboratorio de técnicas analíticas nucleares. Informe Científico Tecnológico. 2013; 13:95-109.

[3]. CAN Bus [Homepage]. eLinux [acceso 2016]. Disponible en: http://elinux.org/CAN_Bus

4]. SK Pang Electronics Ltd. [Homepage]. PiCAN2 CAN-Bus Board for Raspberry Pi 2/3 [acceso 2015]. Disponible en: http://skpang.co.uk/catalog/pican2-canbus-board-for-raspberry-pi-2-p-1475.html

[5]. Pfeiffer O, Ayre A, Keydel C. Embedded Networking with CAN and CANopen. 1st ed. California: Copperhill Technologies Corporation; 2003.

[6].Seeed Development Limited [Homepage]. CAN Bus Shield V1.2 eLinux [acceso 2015]. Disponible en: http://wiki.seeed.cc/CAN-BUS_Shield_V1.2/

[7]. Geisler J. AGCON - Arduino Generic CANopen Node [Homepage]. Hackaday [acceso 2014]. Disponible en: https://hackaday.io/project/3614-agcon-arduino-generic-canopen-node.
[8]. CAN in Automation (CiA) e.V. CiA 301. CANopen. Application layer and communication profile. Version 4.0.2; 2002. Disponible en: https://workarea.ego-gw.it/ego2/ego/itf/software/301_canopen.pdf
[9]. Dupin Francis. CANOpen Memento. 2009
[10]. ByteMe [Homepage]. CanOpen on a Raspberry PI using CanFestival [acceso 2015]. Disponible en: http://www.byteme.org.uk/2015/11/12/canopen-on-a-raspberry-pi-using-canfestival/
[11]. Kaschel H, Pinto E. Análisis protocolar del bus de campo CAN. [Internet]; 2015. Disponible en: https://www.researchgate.net/publication/265488252_ANALISIS_PROTOCOLAR_DEL_BUS_DE_CAMPO_CAN