Automatización de un puente grúa a escala, mediante una plataforma embebida la cual soporta multiprogramación

Camilo A. Pérez Quintero. Asesor: Iván Castillo, Co-asesor: Carlos Camargo
Universidad de Los Andes, Colombia
Diciembre de 2005, camiloapq@gmail.com , iv-casti@uniandes.edu.co , cicamargo@unal.edu.co

Resumen-**En este artículo se resume el trabajo de grado, "Diseño construcción y automatización de un puente grúa a escala, mediante una plataforma embebida la cual soporta multiprogramación", este fue realizado durante el primer semestre de 2005. Solo se tratara todo lo referente al diseño e implementación electrónico, si esta interesado en el diseño y construcción mecánico lo podrá encontrar en detalle en el documento de tesis consignado en la "Universidad de los Andes" (Bogota, Colombia) o simplemente contactando a uno de los autores. El artículo inicia con una breve descripción del modelo construido de puente grúa, luego se plantea una arquitectura base para su automatización sobre un sistema embebido el cual soporta multiprogramación. Finalmente se desarrollan algoritmos de aplicación sobre el sistema los cuales tienen la particularidad de utilizar primitivas del sistema operativo implementado, permitiendo así generar aplicaciones de multiprogramación sobre el puente grúa en tiempo real.**

Palabras Clave – Real Time System, Operating Systems Computer Architecture, Embedded systems, Game Boy Advance, XPORT.

1. INTRODUCCION

La necesidad y uso de un computador personal en casi cualquier ámbito crece cada día. Aun más el uso de pequeños computadores personales especializados los cuales se pueden denominar sistemas embebidos. En este artículo se debe entender un sistema embebido como cualquier dispositivo electrónico similar aun computador personal con la característica de que este fue diseñado para un uso particular y no para un uso general, corriendo el riesgo de entrar en controversias con esta definición ya que muchos autores denominarían a un procesador como un sistema embebido pero para el caso de este articulo entiéndase de la manera explicada anteriormente. Es así entonces como cada día encontramos más sistemas embebidos en nuestro alrededor, celulares, reproductores de música con video etc. Ese crecimiento en el uso de este tipo de dispositivos no solo en el uso personal si no también a nivel de industria es una de las principales motivaciones que impulsaron la elaboración de este trabajo. El documento se inicia con una breve descripción del diseño elaborado del puente grúa. En seguida se introduce la plataforma embebida con la cual se trabajara. Después se plantea una arquitectura solución. Siguiente a esto se propone una automatización para implementar y finalmente se generan los algoritmos necesarios para poner en marcha la automatización propuesta.

2. PUENTE GRUA

El modelo del puente grúa consiste en dos carros laterales; cada uno impulsado por un motor AC, proporcionando movimiento en el eje Y. Un carro secundario impulsado con un motor DC, proporcionando movimiento en X. Finalmente un motor DC contenido dentro del carro secundario. El cual proporcionara al sistema el movimiento en Z. La ubicación de los sensores utilizados y el tipo de estos se encuentran resumidos en las siguientes ilustraciones.

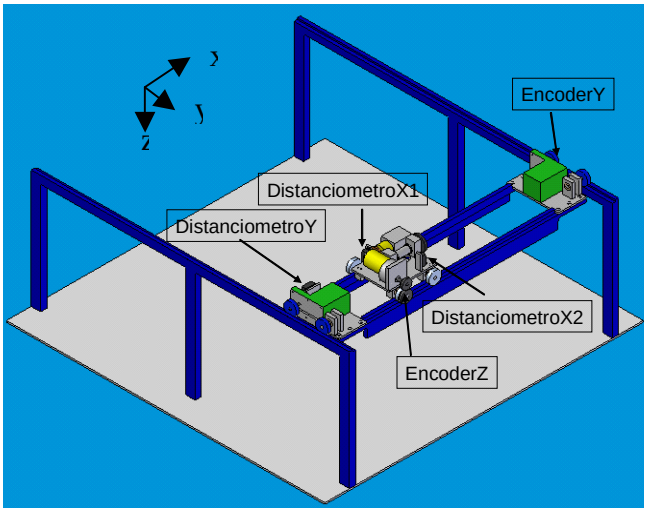


Ilustración 1. Modelo CAD Puente Grúa

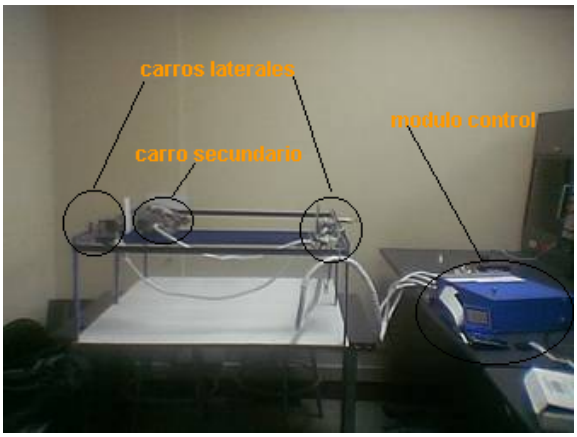
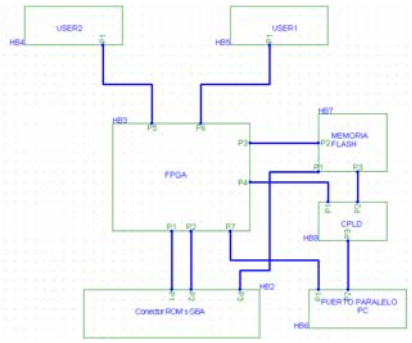


Ilustración 2. Modelo Físico Puente Grúa

3. PLATAFORMA UTILIZADA

La Xport[1] es una tarjeta de desarrollo comercial que permite convertir al Game Boy Advance (GBA) en un sistema de desarrollo embebido.
El GBA tiene un procesador ARM de 32 bits[2], con un reloj de 16.78Mhz, tiene 96Kb de memoria de video, 32Kb de memoria rápida interna RAM y 256 Kb de memoria RAM externa. Posee un "display" con resolución máxima de 240x160.
La Xport utilizada posee una FPGA de 50000 compuertas lógicas (1728 celdas lógicas), 1 CPLD, 4MB de memoria



Flash y 64

señales de usuario programables (ver ilustración).

Ilustración 3.GBA +XPORT

4. ARQUITECTURA

Teniendo claro la plataforma a utilizar, el funcionamiento de un sistema operativo [1] con multiprogramación y teniendo construido el puente grúa con su implementación sensorial el siguiente paso es plantear una arquitectura base para el sistema realizando una partición Hardware y Software. La siguiente ilustración muestra la caja negra de nuestro sistema en donde existe una interfase con el usuario final, la cual será principalmente la pantalla y botones del GBA. Las señales de entrada son los 3 distanciómetros análogos y las tres señales de los encoders. Las señales de salida son las señales de control a los dos motores DC(X,Z) y a los dos motores AC(Y).

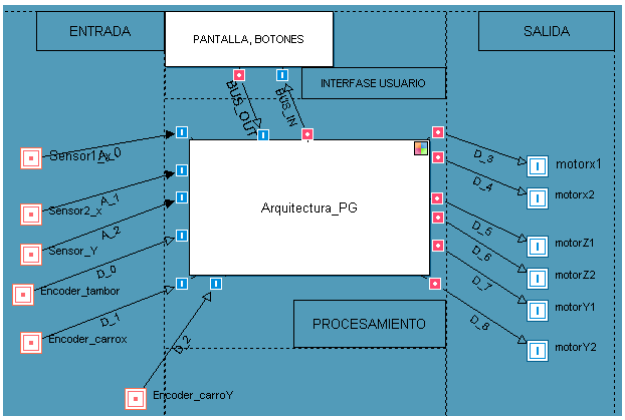


Ilustración 4.Caja Negra Sistema

Dentro de la caja negra podemos diferenciar tres bloques claros. Un bloque de conversión análoga digital para sensores de respuesta análoga, un bloque de adecuación de señal para los motores y finalmente un bloque de control.

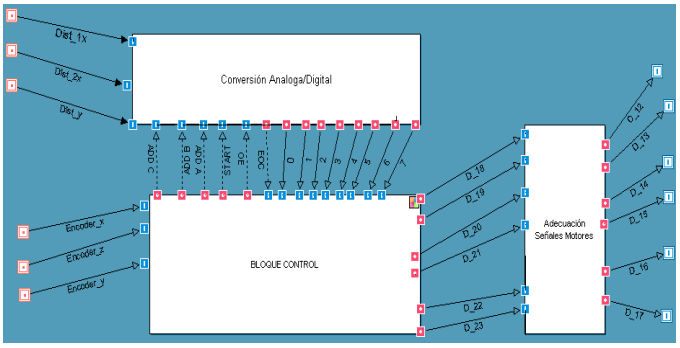


Ilustración 5.Dentro de Arquitectura PG

Dentro del bloque de control encontramos 3 bloques de hardware específicos, el GBA, la FPGA y la memoria Flash [5]. El GBA se comunicara con la arquitectura diseñada por medio de su conector de lectura de ROM´s (juegos). Las señales del conector son 32 en las cuales se encuentra un bus de datos de 2 bytes, un bus de direcciones de 24 bits, señal de lectura, escritura y una señal de interrupción (el bus de datos se encuentra multiplexado con los dos primeros bytes del bus de direcciones) [3]. La memoria Flash se utilizara para almacenar tanto el

[1] Puede encontrarse en www.charmedlabs.com
[2] Las especificaciones de este pueden ser encontradas en el sitio Web de ARM.

programa que será ejecutado como el archivo para sintetizar en la FPGA, el cual será luego utilizado por el CPLD (no se muestra en la arquitectura por claridad) para configurarla cada vez que se inicie la aplicación la FPGA.

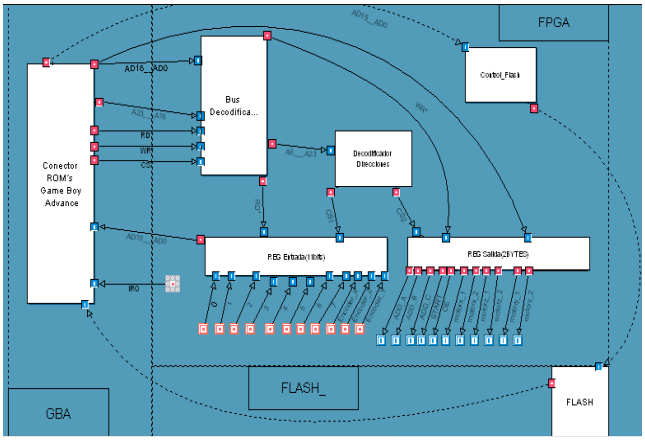


Ilustración 6.Dentro de BLOQUE CONTROL

Finalmente tenemos la FPGA, la cual nos permitirá conectar, el GBA y la memoria FLASH con el mundo exterior mediante 64 pines configurables los cuales se encuentran distribuidos en dos conectores cada uno de 34 pines (ver ilustración3).
Dentro del bloque de la FPGA se encuentra sintetizado un decodificador de bus, el cual tiene como función separar el bus datos del bus de direcciones. Un bus de direcciones el cual tendrá como entrada el bus de direcciones y de salida señales de CS, las cuales permitirán seleccionar uno de los periféricos diseñados, los periféricos son dos registros, un registro de 2 Bytes de salida el cual esta divido como se muestra en la ilustración 7, la primera parte se encarga de todo lo referente al control del conversor análogo/digital y los demás bits se encargan de las señales de control de los motores. El registro de entrada es de 11 bits el primer byte es para la lectura digital directamente del conversor y los tres bits restantes son entradas directas de los encoders. Finalmente existe una señal de interrupción que proviene directamente del conversor A/D al pin de interrupción del GBA.

SALIDA

Control Conversor A/D	Control Motores

ENTRADA

Dato ADC	Lectura Encoders

Ilustración 7. Registros

5. APLICACION

La aplicación de multiprogramación que se propone desarrollar es posicionamiento de la grúa utilizando primitivas del sistema operativo, en este caso se utilizara ECOS [2] como sistema operativo el cual es un RTOS[3]. La idea es utilizar las ventajas que proporciona un sistema operativo para implementar la automatización del puente grua en particular utilizar múltiples hilos o múltiples "threads" con el fin de implementar un algoritmo que situé en tiempo real a la grúa en una posición especifica. Se busca que el sistema implementado tenga conocimiento de la posición del puente grúa en tiempo real, es decir, mientras realiza su trayecto a un punto destino se debe mostrar por pantalla la ubicación actual.
El programa principal inicia pidiendo la coordenada a la que se desea llevar el puente grúa. Al obtener el destino el proceso principal (ver siguiente ilustración) crea 5 subprocesos los cuales correrán en paralelo y serán independientes el uno del otro.

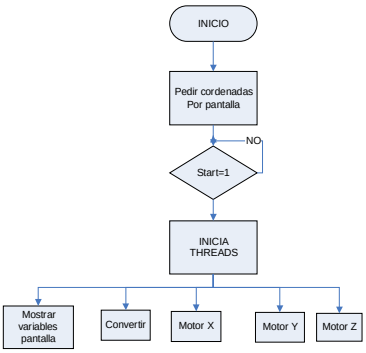


Ilustración 8. Main

Las 5 tareas o subprocesos creados son:

- Motor X, Motor Y, Motor Z los cuales se encargaran de llegar al punto deseado sobre su eje (ver ilustración 10).
- Convertir: se encarga de generar la señal de inicio de conversión para el conversor A/D.
- Mostrar: Se encarga de mostrar el valor de las variables de los sensores en la pantalla durante el desplazamiento del puente grúa.

En la siguiente ilustración se muestra los diagramas de flujo para los 5 subprocesos creados por el main (el subproceso de los motores es igual para todos).

[3] "Real Time Operating System"

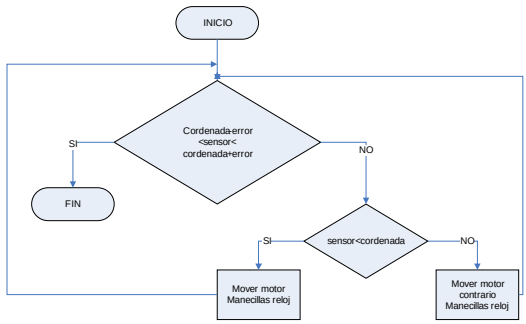


Ilustración 9.”Threads”

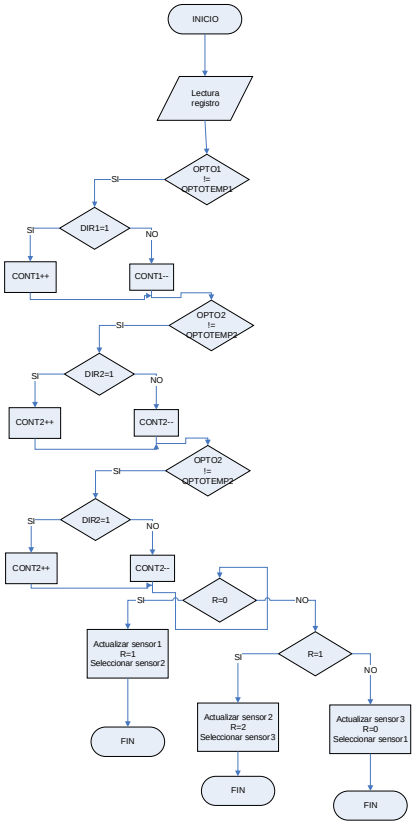


Ilustración 10.Interrupción

Existe otra tarea corriendo es la de interrupción la cual se ejecuta cada vez que se genera una interrupción, su diagrama de flujo se muestra en la ilustración anterior, se encarga de almacenar la lectura de los sensores en variables globales las cuales serán luego utilizadas por las diferentes tareas iniciadas en el proceso principal.

Finalmente se realizo el montaje físico de la arquitectura propuesta, se diseñaron los circuitos impresos tanto del bloque de conversión A/D como el de adecuación señal de motores (ver ilustración 6) y se fabrico un modulo de mando para el control del puente grúa.

Ilustración 11.Modulo de mando

6. CONCLUSIONES

- La implementación de tareas paralelas es una posible solución a la espera activa generada por programas secuenciales.
- La utilización de un sistema operativo sobre una plataforma embebida facilita la implementación de algoritmos de control sobre esta.
- La ventaja principal en el uso de un sistema operativo es la abstracción del hardware que realiza este, permitiendo al diseñador implementar algoritmos de control en lenguaje de alto nivel y a su vez proporcionándole primitivas del sistema las cuales facilitan el desarrollo de algoritmos.
- Aunque el sistema es robusto el grado de exactitud de posicionamiento esta sujeto al tipo de sensores utilizados.
- La aplicación de multiprogramación soluciona muchos problemas de la programación secuencial, pero a la vez trae problemas de sincronismo y de exclusión mutua [1].

7. REFERENCIAS

[1] A. Tanenbaum, Sistemas operativos Modernos.. Person Educación, 2003.
[2] Anthony J. Massa, Embedded Software Development with ECOS. Prentice Hall 2003.
[3] Jain, Gadre, Gameboy Advanced for Non- Gaming Applications. Embedded Systems 2004.
[4] A. Berger, Embedded System Design. CMP Books 2002.
[5] Carlos Camargo Diagramas de bloques XPORT, archivo personal.
[6] A.R.W. Edwards, Embedded System Design on a Shoestring, Newnes 2003.
[7] Michael Barr, Programming Embedded Systems in C and C++, O´Reilly 1999.